ADDALY · THE AI CLASSROOM

Design, Before Any Software

Hierarchy, type, colour and spacing — the part of design you can do with a pencil, before any software.

---

7 modules · 74 lessons · 25 figures · 7 case studies · 61,212 words · about 11 hours

Dr Mustafa Mun
Author and editor

ABOUT THIS BOOK

Design, Before Any Software, published by Addaly, 2026. Author and editor: **Dr Mustafa Mun**.

The manuscript was drafted with the assistance of a large language model and was edited and published under his name. That is stated plainly here rather than left to be discovered, because a reader is entitled to know how a book they are trusting was made.

This book is the printed form of the course of the same name at addaly.com/learn/design-foundations. The website is the living version and is corrected first; where the two differ, the website is right.

Free to read, free to download, free to print, and free to teach from. There is no paid edition and there never will be. If you paid for this, somebody has sold you something that was given away.

Every diagram in it is drawn from data by code, not generated as a picture, so the numbers on the page are the numbers in the argument. Answers to every exercise, test and examination question are at the back, with a note on why each wrong option attracts people — those notes are the teaching, not the marking.

Contents

1. How seeing actually works

Predict where a stranger's eye will land on a layout in the first two seconds and in what order, change that order by adjusting one named channel to a stated magnitude, and say why a difference below that magnitude reads as a mistake rather than as a rank

2. Space, grouping and the grid

Set every gap in a layout from a single stated spacing scale, name the relationship each gap encodes and the grouping cue carrying it, lay out a column grid from its arithmetic rather than by eye, and say where an optical adjustment must override the alignment tool

1. The gap is doing the talking
2. When two grouping cues disagree
3. Every gap comes from a list
4. Space is content you have not been billed for
5. You can feel a misalignment you cannot see
6. Where the alignment tool is wrong
7. A grid is arithmetic, not taste
8. Vertical rhythm, and why the strict version fails on screen
9. Padding belongs to the thing, margin belongs to the relationship
10. How big a thing has to be to hit
11. Balance is a comparison of weights
12. Case study: The admission form that cost two hundred hours to correct
13. Module test

3. Type, mechanically

Set a page of text a stranger can read comfortably — measure, leading, size scale, alignment, figures and emphasis each chosen from stated numbers — say why each number is where it is, and name what changes when the same page is set in a non-Latin script

1. Your font is fine, your line is too long
2. The four measurements that decide how a typeface behaves
3. Four sizes, chosen once
4. Choosing a typeface, and what the licence actually says
5. Two typefaces, and why you probably want one
6. Rivers, rags and where the line breaks
7. Letter spacing, and the sizes type was drawn for
8. Capitals, italics and the cost of emphasis
9. Figures, dashes and the details nobody teaches
10. The grey of a paragraph
11. What breaks when the text is not in English
12. Make it pop is a symptom report
13. Case study: The scholarship leaflet whose Hindi was clipped
14. Module test

4. Colour, measured

Build a palette from perceptual lightness ramps rather than from hue, state the contrast ratio of every text pair and the criterion it is being judged against, name the specific conditions under which that ratio overstates legibility, and give any colour-carried meaning a second non-colour encoding

1. A palette will not save an unreadable page
2. The lightness slider is not lightness
3. What a contrast checker is actually computing
4. Where the contrast standard is wrong, and what is unsettled
5. What colour blindness actually removes
6. Colour is never the only signal
7. Greys are a design decision
8. Red means danger, except where it does not
9. Dark mode, built rather than inverted
10. Why your blue printed purple
11. How many colours a design needs
12. Case study: The grey that passed in the office and failed in the sun
13. Module test

5. Composition and the image

Crop and place an image so that the subject, the four frame edges, the empty space and any text over it all support one reading, justify the crop by what it removes, and produce a single asset that survives re-cropping to square, portrait and vertical without losing its subject

1. The rule of thirds is a nudge, not a law
2. Why a near-touch is worse than an overlap
3. A crop is a decision about what leaves
4. Building a path through the image
5. Six ways to make a flat surface have depth
6. The space between things has a shape
7. One image, four crops
8. Two focal points is none
9. Text over a picture you do not control
10. Repetition, and the one that breaks it
11. Case study: Two teachers, one hoarding, and the crop nobody wanted to make
12. Module test

6. Real screens, real thumbs, real paper

Produce a layout that works at 360 CSS pixels one-handed in daylight, at 400% zoom, and on paper with correct bleed and resolution, naming for each constraint the specific number it imposes and which design decisions change because of it

1. Design it at 390 pixels or find out the hard way
2. What a pixel is, and how wide phones actually are
3. Breakpoints come from the content, not from devices
4. Where a thumb can actually go
5. Why 10pt is fine in print and terrible on screen
6. The five screens nobody designs
7. Motion, and the people it makes ill
8. The accessibility requirements that are design decisions
9. The designer decides how heavy the page is
10. Bleed, trim and the things paper does
11. Case study: The bus booking site that worked everywhere except the bus stand
12. Module test

7. Making the decision once

Convert a set of one-off screens into named tokens and a small component set, decide for any new element whether it is a variant of something existing or a new thing, specify every state a component can be in, and say at what point a project is large enough for the system to be worth its maintenance cost

1. A token is a decision with a name
2. Names that survive the next change
3. Variant, or a new thing
4. Every state you did not draw still ships
5. Icons are a set, not a collection
6. The afternoon version
7. Specifications that survive contact
8. Consistency is not uniformity
9. When not to build one
10. Case study: The milk round app, and the system built three weeks too early
11. Module test

Design, Before Any Software: final examination

The paper that opens the next course. Answers to everything follow it.

MODULE 1

How seeing actually works

Design is the management of a very small patch of sharp vision and a very large patch of blur. This block covers what the eye can resolve where, how big a difference has to be before it registers as deliberate, what makes a shape read as an object rather than as background, and what the eye-tracking research does and does not support. Everything later in the course is an application of this block.

> BY THE END OF THIS MODULE YOU CAN
>
> Predict where a stranger's eye will land on a layout in the first two seconds and in what order, change that order by adjusting one named channel to a stated magnitude, and say why a difference below that magnitude reads as a mistake rather than as a rank

---

1 · 9 min

Your poster is not ugly, it is unranked

> THE ONE THING TO KEEP
>
> A design has a hierarchy only if things differ enough to be ranked at a glance, and promoting one element means demoting another.

The wrong diagnosis

Someone makes their first event poster. The band name, the date, the venue, the ticket price and a QR code all sit there at roughly the same size, in roughly

the same weight, spread evenly across the page. It looks wrong. They are told they do not have an eye for it, or that they lack taste, and they believe it.

That diagnosis is almost always false. The fault is not taste. Every element on that poster is claiming to be the most important thing, so none of them is. The reader's eye arrives, finds five equal candidates, and gives up. Nothing about colour, typeface or software fixes that, which is why the second version usually looks the same as the first with a gradient on it.

The eye does not scan, it jumps

Human vision is not a camera panning across a page. The eye moves in jerks — saccades — landing three or four times a second, and during the jump itself you are effectively blind. Sharp detail only exists in a tiny central patch, about the width of your thumbnail at arm's length. Everything else is a blur with colour and shape in it.

So a viewer does not "take in" your layout. They land on three to five spots, in some order, and build an impression from those. Design is the business of choosing those spots and that order.

What pulls a landing, roughly strongest first:

- **Size difference.** The blunt instrument. Reliable and boring.
- **Contrast against the background.** A dark thing on a pale field, or the reverse.
- **Isolation.** The underrated one. A small item alone in a wide empty area beats a large item in a crowd.
- **Position**, relative to the direction the reader starts from.
- **Saturation**, then **weight**, then **shape oddity** — one round thing among rectangles.

Notice that four of those six are relative. They exist only by comparison with the rest of the page. That is the whole idea.

Reading direction is a weak cue, and that is good news

A reader of English or Hindi starts at the top left. A reader of Urdu, Arabic or Hebrew starts at the top right. Position matters, but it is a weaker cue than size, contrast and isolation — which means a layout built on those three works for both audiences without redesign. Build the hierarchy out of the strong cues and position becomes a refinement rather than a dependency.

One more thing worth unlearning: the F-pattern. It is a real observation from eye-tracking of text-heavy pages with no visual structure, not a law of vision. On a page with a clear hierarchy, people look at the big thing first. That is the point.

The channels that pull a landing, strongest first

Size difference	The blunt instrument. Reliable, and boring.
Contrast against the background	A dark mass on a pale field, or the reverse
Isolation	A small item alone beats a large item in a crowd
Position	Weaker, and it depends on the script the reader starts from
Saturation	Real, and the first channel to fail in sunlight
Weight	Two steps, or it is not a step at all
Shape oddity	One round thing among rectangles

Four of these exist only by comparison with the rest of the page, which is why emphasis is a relationship rather than a property you apply.

Name first, second, third out loud

Before you open any software, write three lines on paper:

1. What must a stranger see in one second?
2. What should they see next?
3. What is everything else?

On a concert poster: band name / date and venue / all the rest. On a shop sign: what you sell / where you are / opening hours. On a CV: your name and what you do / your last two jobs / everything else.

If you cannot name them, the design cannot show them. That is the actual reason layouts drift: when you have not decided, every element gets nudged up a little each time you look at it.

That is the second failure mode, and it deserves a name: **emphasis inflation**. Each round of feedback adds emphasis to one more item. The phone number goes bold. Then the price goes bold and red. Then the date goes bold, red and bigger, so the band name goes bigger again. After four rounds everything is loud, which is identical to everything being quiet.

The rule that stops it: **to promote one thing, demote another.** Emphasis is zero-sum. If your fix is only ever additive, you are not designing a hierarchy, you are inflating one.

Make the difference obvious or do not make it

An 18pt heading over 16pt body text does not read as a rank. It reads as a mistake. A difference that a viewer cannot detect at a glance costs you the same effort as a difference they can, and buys you nothing.

Use steps you can see. A workable set for a page of text: body 16, subhead 20, heading 32, display 48. Each step is roughly 1.25 to 1.5 times the last. Rule of thumb: if you have to compare two items side by side to tell which is bigger, they are the same size as far as the reader is concerned.

The same applies to weight, to colour, to spacing. Half a difference is worse than none, because it looks accidental.

Four free ways to test it, one of which is your eyelids

- **Squint** until detail disappears. What survives is your hierarchy. If nothing survives, there isn't one.
- **Greyscale it.** In Photopea, GIMP or Krita, pull saturation to –100 (Image, then Adjustments or Colors, then Hue-Saturation). If the structure collapses, you built hierarchy out of colour, and colour is the least reliable channel you have.
- **Shrink it.** Zoom to 25 per cent, or take a screenshot on your phone and pinch it down to thumbnail size. The squint test by other means, and it

needs no software.

- **Three seconds.** Show it to a person for three seconds, take it away, ask what they remember. More honest than any opinion given with the page still in front of them.

Today

Take something you have already made. Write down what you want seen first, second and third. Then screenshot it, hand it to someone for three seconds, and ask them what they saw. Where their answer and your list disagree, you have found the work.

BEFORE YOU MOVE ON

A gig poster has the band name, date, venue and ticket price all set at 40pt bold, and the client says it looks cluttered. A colleague suggests setting the band name at 44pt. Why will that not fix it?

A. A four-point difference is below what a viewer registers as a rank, and making one element first requires demoting the others, because emphasis exists only by comparison.

B. 40pt is simply too large for a poster of that size, so everything should be reset smaller to leave room to breathe.

C. Bold at that size is too heavy to read, so the band name needs a display typeface to separate it from the rest.

D. The elements need more space between them; once the spacing is even, the hierarchy will follow.

2 · 9 min

What your peripheral vision is actually for

THE ONE THING TO KEEP

Outside the central patch, vision keeps luminance blobs, motion and gross shape and throws away detail and anything with close neighbours, so a landmark has to be a large isolated blob rather than a small precise mark.

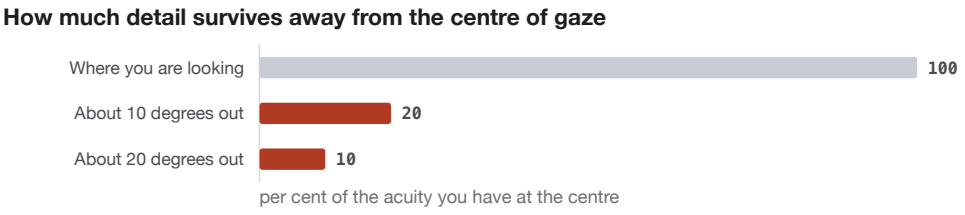
The blur is not a lesser version of the sharp bit

Outside the small central patch where you see detail, vision does not simply get fuzzier. It becomes a different instrument with different jobs. Understanding what that instrument can and cannot do tells you which parts of a layout can act as landmarks and which parts are, for practical purposes, invisible until somebody looks straight at them.

Acuity falls off faster than most people expect. At about 10 degrees away from where you are looking — roughly a hand's width at arm's length — you resolve around a fifth of the detail you resolve at the centre. At 20 degrees it is closer to a tenth. On a laptop screen at normal distance, that means while you read this sentence, the top corners of the window are being seen at about the resolution of a road sign a hundred metres away.

Crowding is the real limit, not size

The more useful fact is this: in peripheral vision, things interfere with each other. A letter that is easily large enough to identify on its own becomes unreadable when other letters sit beside it. This is called crowding, and it follows a rough rule — interference comes from anything within about half the distance to the point you are fixating.



Ten degrees is roughly a hand's width at arm's length. Crowding removes more again: a word that is legible alone out there is not legible inside a paragraph.

So at 10 degrees out, everything within about 5 degrees of a target is smearing it. A single large word in empty space is legible out there. The same word inside a paragraph is not. This is why isolation works as an emphasis channel, and why it works even better than size: isolation does not just make something bigger, it removes the flankers that were destroying it.

The practical version:

- A 16px grey label in a corner cannot be a landmark. Nobody will find it without a deliberate search.
- A 40px dark heading with 32px of clear space around it is a landmark from anywhere on the page.
- A 16px icon in a row of 16px icons is unfindable peripherally. The same icon at 24px, filled rather than outlined, with a gap either side, is findable.

Filled shapes beat outlined ones out there for the same reason. A 1.5px outline is exactly the kind of high-frequency detail the periphery throws away; a solid blob is exactly what it keeps.

What the periphery is good at

Three things, all of them coarse:

1. **Luminance blobs.** Large areas of light or dark, and their approximate position.
2. **Motion.** Peripheral motion detection is excellent, and largely involuntary.
3. **Gross shape and large areas of colour.** Enough to tell a circle from a bar, or a red region from a grey one.

The motion point deserves attention because it is the one designers most often get wrong on purpose. An animated banner in the corner of a page is not competing politely for attention; it is triggering a reflex evolved to notice things moving at the edge of your vision. That is why auto-playing carousels and looping adverts are so widely hated, and it is why operating systems ship a reduce-motion setting. If you make something move in the periphery, you have taken a fixation whether the reader wanted to give you one or not.

You cannot inspect your own periphery

Here is where most people get stuck. The obvious way to check whether a corner label is findable is to look at the corner — at which point it lands in the fovea and looks perfectly clear. The instrument destroys the measurement.

The usable proxy is a blur. Export the layout as an image and apply a Gaussian blur at roughly 1.5% of the image width: on a 1200px-wide screenshot, that is a radius of about 18 pixels. Free tools do this in one menu:

- **GIMP**: Filters > Blur > Gaussian Blur, set both radii.
- **Photopea** (runs in a browser, no install): Filter > Blur > Gaussian Blur.
- **Krita**: Filter > Blur > Gaussian Blur.

What you are looking for in the blurred image is which shapes still exist as distinct blobs and where they sit. Anything that has dissolved into the background is something a reader will only ever find by deliberate search. That may be fine — a legal footer should dissolve — but it should be your decision rather than an accident.

> *The blur test asks a different question from the squint test in the next lesson. The blur asks what can be found. The squint asks what comes first.*

The honest limitation

This is a model of the optics and early visual processing, not of the person. A reader who has come to your page specifically to find the unsubscribe link will find a 12px grey link in the footer, because deliberate search overrides all of it — they will scan systematically until the fovea lands on it. Peripheral design

governs the unguided glance, which is most first encounters and almost no second ones.

It also varies. Peripheral acuity and crowding both worsen with age, and a reader with macular degeneration has the opposite problem — the central patch is the one that fails, and the periphery is all they have. Neither is a reason to abandon the model; both are reasons to keep real text at real sizes rather than treating the whole design as a pattern of blobs.

BEFORE YOU MOVE ON

A dashboard has a row of eight 16px outlined icons with 4px between them. Users report they cannot find the settings icon without hunting. Enlarging it to 20px does not help. Why not?

A. Outlined icons are a stylistic mistake and filled icons are always more usable, regardless of the layout around them.

B. 16px is below the minimum legible icon size, so no arrangement of icons that small can work.

C. Users are scanning strictly left to right, and the settings icon sits too far along the row for a quick scan to reach it before attention moves elsewhere on the screen.

D. The neighbouring icons are crowding it, so extra size does not help until the flankers are moved away or removed.

3 · 8 min

The squint test, done properly

THE ONE THING TO KEEP

Convert a design to luminance greyscale and blur it slightly: if three masses do not appear in a clear order, the fault is structural and no amount of colour work will repair it.

Take the colour away and see what is left

The fastest diagnostic in design costs nothing and takes about twenty seconds. Convert the design to greyscale and look at it. What you are inspecting is the value structure — the pattern of light and dark — which is the layer the eye resolves first and the layer that survives every bad viewing condition you will ever meet.

Most layouts that look wrong are wrong here, and the colour work sitting on top of the fault is wasted effort. This is worth doing before you have chosen a single hue.

The procedure, with numbers

1. **Flatten to greyscale using luminance, not lightness.** The distinction matters. A luminance conversion weights green heavily and blue barely — roughly 21% red, 72% green, 7% blue — because that is how much each contributes to perceived brightness. A naive average of the three channels turns a saturated yellow and a saturated blue into similar greys, which is precisely the error you are trying to detect.

- **GIMP**: Colors > Desaturate > Desaturate, mode Luminance. - **Photopea**: Image > Adjustments > Black and White. - **Krita**: Filter > Adjust > Desaturate, choosing Luminosity. - **Your phone**, for checking anything on screen: Android Settings > Accessibility > Colour correction, or iOS Settings > Accessibility > Display and Text Size > Colour Filters > Greyscale.

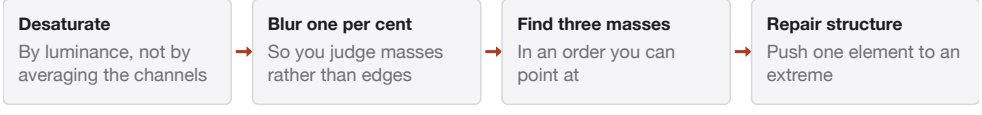
1. **Blur it slightly**, about 1% of the width, so you are judging masses rather than edges.
2. **Look for three blobs in an order.** Not five, not one. You should be able to point at the darkest or lightest mass, then the second, then the third.

What the failures look like

Uniform mush. Everything sits in the same middle band of grey. This is the commonest result and it means there is no hierarchy to fix with colour, because colour is not going to introduce a value difference that is not there. The repair is to push something to an extreme: take the heading to near-black or the background to near-white, or invert a block.

The wrong thing is loudest. A decorative photograph is the darkest mass and the headline is second. The eye will land on the photograph. Either the photograph is the message, in which case the headline needs to move onto it or away from it, or it needs to be lightened.

The squint test, in four steps



Colour work done before this step gets thrown away, because hue cannot introduce a lightness difference that is not there.

Two things tie for first. Two elements at the same value, similar size, in different places. The eye picks one at random and the reading order becomes unpredictable between viewers.

A number to hold on to: if the mean grey of your primary element and the mean grey of what surrounds it differ by less than about 20 levels out of 255, the difference will not register at a glance. You can measure this rather than guess — the colour picker in any editor gives you the value, and GIMP's Windows > Dockable Dialogs > Histogram shows you the mean of a selection directly.

Why this works when other checks do not

Value contrast is the most robust signal you have. It survives:

- **Sunlight on a phone**, where saturation collapses and everything washes towards mid-grey.
- **Colour blindness**, which alters hue discrimination and leaves lightness largely intact.
- **Cheap screens** with narrow gamut and poor calibration, which is most screens your readers own.
- **Photocopying, faxing and newsprint**, which still matter more than anyone in a design conversation expects.
- **Peripheral vision**, which is a luminance channel first.

If the structure holds in grey, it holds. That is a stronger claim than almost anything else you can say about a layout for free.

What it cannot catch

Be clear about the limits, because people over-trust this test once it has saved them a few times.

- It will not catch a **measure problem**. A paragraph set 140 characters wide has a perfectly good value structure and is still miserable to read.
- It will not catch **colour-only meaning**. Red and green status dots of the same lightness look identical in grey, which correctly warns you that a colour-blind reader is stuck — but a design that passes the squint test can still fail here if the dots differ in lightness while carrying meaning only a colour name explains.
- It will not catch **semantic errors**. A beautifully ranked layout that promotes the wrong message is still wrong, and no filter will tell you.
- It says nothing about **whether the palette is pleasant**. It is a structural test, not an aesthetic one.

Build it into the loop

The reason this test gets skipped is that it arrives too late. By the time the design is finished, converting it to grey feels like an audit of work you have already committed to. Do it at the first rough instead, when the design is four grey boxes and a line of text, and it costs nothing to change.

A useful habit: design the first version entirely in greyscale, with no hue at all, and only introduce colour once the grey version ranks correctly. Everything you then add is decoration on a structure that already works, rather than an attempt to rescue one that does not.

BEFORE YOU MOVE ON

A designer desaturates a layout by averaging the red, green and blue channels equally, and concludes the yellow call-to-action and the blue background have good value separation. On a real screen the button nearly disappears. What went wrong?

A. An equal-channel average is not perceived brightness — green contributes about three-quarters of it and blue almost none, so the conversion flattened a real difference and invented a false one.

B. Averaging is correct but the monitor is uncalibrated, so the on-screen result differs from the file.

C. Yellow and blue sit opposite each other on the colour wheel, so they are guaranteed to separate in lightness as well as in hue, which means the greyscale check was never going to add anything useful here.

D. The blur step was skipped, and without it the test reports edges rather than masses.

4 · 9 min

How big a difference has to be

THE ONE THING TO KEEP

Every difference you introduce must clear a perceptual threshold — roughly 1.2x for size, two weight steps, 15 to 20 points of lightness, double the gap — because a difference below it carries no information and reads as a production error.

Small differences read as errors

A heading at 17px above body text at 16px does not create a hierarchy. It creates the impression that something went wrong in production. The reader cannot tell whether the difference is deliberate, and a difference that might be an accident carries no information.

This is the single most common technical fault in beginner work, and it is entirely fixable with arithmetic. The rule is that every difference you introduce has to clear a threshold, and below that threshold you are better off with no difference at all.

The thresholds, channel by channel

These are working numbers rather than laws of nature, drawn from what holds up in practice on ordinary screens at ordinary distances.

Size. A step needs to be at least 1.2 times the size below it to read as deliberate, and 1.25 is more comfortable. From a 16px body, that is 20px minimum for the next step up. In practice most designers use a modular scale — pick a ratio and multiply repeatedly:

```
ratio 1.25 from 16px:  16 → 20 → 25 → 31 → 39 → 49
ratio 1.333 from 16px: 16 → 21 → 28 → 38 → 51 → 67
```

Round to whole pixels. The point of a scale is not mathematical purity; it is that every size on the page is guaranteed to clear the threshold against every other size, without you checking each pair.

Weight. On most screen typefaces, 400 against 500 is invisible at body size — the two render nearly identically after hinting and anti-aliasing. 400 against 600 reads. 400 against 700 reads clearly. If a family only offers Regular and Bold, you have one weight step, not three.

Lightness. Two greys need roughly 15 to 20 points of difference on a 0-100 lightness scale before the difference is unambiguous rather than suspicious. Text at 40% grey beside text at 45% grey looks like a rendering bug.

Spacing. A gap that separates groups should be at least twice the gap inside them. 8px inside and 12px outside is ambiguous; 8px inside and 24px outside is not. This one has the most slack — going to three times is usually better, never worse.

Position. A 4px indent reads as a misalignment. An indent needs to be at least a full space unit — 16px or more — to read as an indent.

Why thresholds exist at all

Perception of most quantities is proportional rather than absolute. The difference you notice depends on what you are comparing against: adding 1px to a 16px letter is a 6% change and invisible, while adding 1px to a 4px border is a 25% change and obvious. This is why design scales are multiplicative rather than additive, and why a spacing system of 4, 8, 12, 16, 24, 32, 48, 64 works better than 4, 8, 12, 16, 20, 24, 28, 32 — the first keeps the proportional gap roughly constant as the numbers grow, the second lets it collapse.

There is a second reason, and it is about trust rather than optics. Work that contains many small unexplained differences reads as careless even when the reader cannot name why. Three sizes used consistently looks designed. Eleven sizes, several of them one pixel apart, looks like a document that six people edited.

The counter-rule: sometimes go far past the threshold

Clearing the threshold is the minimum, not the target. A poster read from three metres away needs its primary element at maybe four or five times the body size, not 1.25 times. A price on a product card might be twice the size of everything else. Threshold arithmetic tells you the floor; the job tells you the ceiling.

A difference either clears the threshold or reads as a fault

Reads as a rank	Reads as a production error
20px heading over 16px body, a ratio of 1.25	17px heading over 16px body
Weight 400 against weight 700	Weight 400 against weight 500
Lightness 40 against lightness 60	Lightness 40 against lightness 45
8px inside a group, 24px around it	8px inside a group, 12px around it
A 16px indent	A 4px indent

Half a difference costs the same effort as a whole one and buys nothing, because a reader cannot tell whether it was deliberate.

The failure at the other end is real but rarer: a heading so large it stops being read as text and becomes a texture, or a value difference so extreme the page looks like two unrelated designs stapled together.

How to audit a design for this

List every distinct value you have used, per channel, and count them.

```
sizes:    13, 14, 15, 16, 18, 20, 22, 24, 32
weights:  400, 500, 600, 700
gaps:     4, 6, 8, 10, 12, 16, 18, 20, 24, 32
```

That list is a diagnosis on its own. Nine sizes for a page that needs four. Ten gap values where five would do. The repair is to map each stray value onto the nearest value you intend to keep and delete the rest, which usually improves the design immediately and always makes it easier to change later.

Browsers make this auditable for free: open developer tools on any page you have built, and the computed styles panel gives you the actual rendered values rather than the ones you believed you had set.

The honest caveat

These numbers are thresholds for a reader with ordinary vision, on a reasonable screen, at ordinary distance, paying attention. Every one of those conditions can fail. A reader at 200% browser zoom is seeing proportions unchanged, so ratios survive — which is an argument for ratios. A reader in bright sunlight is losing lightness contrast, so value steps need more headroom than the minimum. When in doubt, exceed the threshold rather than sit on it; nothing is ever ruined by a hierarchy that is slightly too clear.

BEFORE YOU MOVE ON

A team standardises its spacing on 4, 8, 12, 16, 20, 24, 28, 32 pixels and finds that the large gaps still look ambiguous while the small ones feel fine. What explains the pattern?

A. The scale has too many steps, and any eight-step spacing system will produce ambiguity somewhere.

B. Large gaps need to be round multiples of the body text size, and 20 and 28 are not.

C. The steps are a constant 4px apart, so the proportional difference shrinks as the numbers grow — 4 to 8 is double, while 28 to 32 is a 14% change that falls below the threshold.

D. Gaps above 16px should always be expressed in a relative unit such as rem rather than in pixels, so that they scale with the reader's chosen font size and stay proportionate at 200% zoom.

5 · 8 min

Emphasis is zero-sum

THE ONE THING TO KEEP

Emphasis is a fixed budget on any single view, so making one element more prominent always means demoting another, and the answer to make this bigger is the question what comes down.

You cannot add importance, only redistribute it

A client asks for the phone number to stand out more. You make it bigger and bolder. A week later they ask for the logo to stand out more, so you make that bigger too. Then the offer. Then the opening hours. Six rounds later every element is large and bold, and the design communicates less than the first draft did.

Nothing went wrong in any individual round. The error is in the model. Emphasis is not a property you apply to an element; it is a relationship between an element and everything around it. There is a fixed amount of it on any given surface, and every time you give some to one thing you take it from the rest.

This is the most useful single idea in the course, because it converts an unanswerable request — make this stand out — into an answerable one: what comes down.

Three ranks, and a floor

A workable budget for one screen or one page:

- **One primary.** The single thing you want landed on first. Exactly one.
- **Two or three secondaries.** Found on the second pass, clearly below the primary and clearly above the rest.

- **Everything else at the floor.** Body text, labels, legal, navigation in its resting state.

The temptation is always to promote. Resist it by demoting instead: if something needs to rank higher, the faster and cheaper move is usually to take a channel away from whatever is currently competing with it, rather than to pile a channel onto it.

Worked example. A product page has a price at 24px bold black, an Add to Basket button in solid brand colour, a promotional badge in red, and a 20px bold black delivery promise. Four things fighting. The repair is not a bigger button. It is: delivery promise to 16px regular grey, badge to a quiet outline, price to 24px semibold. The button now wins without having changed at all.

Three primary buttons is no primary button

The pattern is visible everywhere once you know it. A form with Save, Save and Continue, and Publish, all in solid blue. A dashboard where every card has a coloured header. A homepage with four hero banners in a rotating carousel.

In each case somebody made a local decision that was individually defensible and collectively self-cancelling. The carousel is the purest example: it exists because several teams each needed the top slot, and the compromise gives every one of them a slot that most readers will never see, because the second slide arrives after most people have scrolled past.

Emphasis wears out

A channel used everywhere stops being a channel. If every heading on a site is in the brand orange, orange means heading, not important — and to make something important you now need a fourth channel you have not spent yet.

This is also why the resting state matters as much as the emphasised one. A button that is emphasised has to sit against buttons that are not. If every interactive element is filled and coloured, you have spent your loudest channel on the mere fact of interactivity, which is a poor use of it — most interfaces are

better served by one filled button per view and outlined or plain-text treatments for everything else.

How to hold the line in a meeting

The request to make something bigger is almost never about size, and arguing about size is a losing game. Two sentences that work:

> *If we promote the delivery promise, the button has to come down. Which of those two do you want someone to see first?*

And, when everything is already promoted:

> *We have four things at the top rank, so the reader is choosing at random which one they see first. Can we pick the one we would want if we only got one?*

This usually resolves it, because the underlying disagreement was never visual. It was that two people wanted different things to be most important, and nobody had said so out loud.

A test you can run in ten seconds

Show the design to someone for three seconds, take it away, and ask what they saw. Not what it said — what they saw. If three people give three different answers, you do not have a primary. If they all name the same element and it is the one you intended, the budget is spent correctly.

This is cheap enough to do with whoever is sitting near you, and it is the only check in this lesson that tests the design rather than your reasoning about it.

The limit of the rule

One primary per *view*, not per document. A long page is a sequence of views, and each screenful can have its own primary — that is how a well-structured article works, with one dominant element in each section as you scroll. The budget resets as the reader moves. What it never does is expand to cover four things in the same glance.

There are also surfaces that genuinely have two peers: a comparison of two plans, a before and after, a choice between exactly two options. Those are real, and the answer is symmetry rather than hierarchy — make them deliberately equal, so the reader understands the tie is intentional rather than unresolved.

BEFORE YOU MOVE ON

A long scrolling landing page has six sections, each with its own large heading and one bold call to action. A reviewer objects that this breaks the one-primary rule. Is the reviewer right?

A. Yes, because a document can have only one primary element and the remaining five headings should be reduced to secondary rank.

B. No, because the budget applies to one glance, and a scrolling page is a sequence of views that each get their own primary as the reader moves through it.

C. No, because calls to action are exempt from the emphasis budget and can repeat as often as conversion requires.

D. Yes, unless the six sections are given six different accent colours, which would let the reader tell one section from another and would keep the overall ranking of the page intact.

6 · 9 min

What makes a shape read as an object

THE ONE THING TO KEEP

The eye decides what is object and what is background before it ranks anything, and space is a stronger device for making that decision than fills, borders or shadows — in that order.

Every mark is either a thing or the space behind a thing

Before the eye can rank anything, it has to decide what the objects are. This happens early, fast and below the level of deliberate thought, and it is called the figure-ground decision. A shape that the eye assigns as figure gets treated as a thing with edges, a location and a meaning. A shape assigned as ground gets treated as nowhere — it has no shape at all, as far as perception is concerned.

You can feel this with the classic faces-or-vase image. Whichever region you read as the figure appears to have a boundary; the other region appears to extend behind it, edgeless. The two readings never happen at once. This is not an illusion so much as a demonstration of a decision your visual system makes about every part of every image.

In design the practical question is mundane: how do I make this card, panel or button read as a distinct object? There are four answers, and they are not equal.

The four devices, ranked by strength

1. Space. An element surrounded by a clear gap is an object, with no other treatment at all. This is the strongest and cheapest device, and the one beginners use least, because it consumes area and looks like doing nothing. A block of text with 48px of nothing around it is unmistakably a unit.

2. A fill. A change of background. This is strong, but only in proportion to the lightness difference. A card at 98% lightness on a 100% background is invisible on most phones in daylight; the same card at 96% on a 100% background is a real object. Measure it rather than trusting your calibrated monitor in a dim room — a 2% difference is about 5 levels out of 255, and a great many screens will not resolve it at all.

3. A border. Weaker than people expect. A 1px line at 10% contrast against its surroundings does less work than a 4% lightness fill, because the border is a thin high-frequency detail and the fill is an area. Borders come into their own when you need an edge that survives on a coloured background, or when you want to define an object without changing its brightness.

4. A shadow. The weakest and the most abused. A shadow says this object floats above that surface, which is a depth claim rather than a grouping claim. Real shadows are large, soft and low in opacity — something like 0 to 4 pixels down, 12 pixels of blur, at 8% black, not a hard 2px offset at 50%. A heavy shadow reads as a sticker, and a page of heavy shadows reads as a page of stickers.

The common failure is to reach for devices 3 and 4 while leaving device 1 unused. A card with a border and a shadow, crammed 8px from its neighbour, is still hard to read as a separate thing. Give it 32px of space and it does not need either.

Ambiguity is the fault, not weakness

A figure-ground relationship fails when it is ambiguous rather than when it is subtle. Two panels of equal area and near-equal lightness, side by side, both trying to be the object, produce a layout that feels unstable in a way viewers cannot name. Nothing is wrong with a quiet distinction; something is wrong with a contested one.

Watch for it in these places:

- **Inverted sections.** A dark band across a light page reads as figure — a large object — not as a section of the ground. That is often what you want, but it means the dark band is now competing at the top of your hierarchy.

- **Images with pale edges.** A photograph that fades to white on a white page has no boundary and reads as several disconnected objects.
- **Sidebars the same colour as the content.** If the only thing distinguishing them is a 1px rule, the eye keeps re-deciding which is the main region.

A rule of thumb for nesting

When you put a card inside a section inside a page, each level needs a visible relationship, and the usual mistake is to keep stepping the lightness in the same direction until you run out. Three levels of grey, each 3% apart, fail. Two work.

The reliable pattern is to alternate rather than accumulate: page light, section slightly darker, card back to light — or to change device between levels, using space for the section and a fill for the card. Depth in interfaces is almost never more than two levels deep, and a design that needs four is usually telling you the information architecture is wrong.

Checking it

The blur test from earlier in this block is the right instrument. Blur the layout and ask, of each region: is this a blob, or is it part of the field? If a card dissolves, it is ground, whatever the border says.

A second check, free and fast: turn the screen brightness down to about 30% and look at it in a bright room. Every marginal fill disappears. What remains is what a reader on a bus in the afternoon is working with, and it is a more honest account of your design than the one on your desk.

BEFORE YOU MOVE ON

A designer distinguishes a card from the page with a 1px border at 8% contrast and a 2px hard shadow at 40% black, packed 8px from the neighbouring card. On a phone in daylight the cards read as one block. What is the most effective single change?

A. Darken the border to 30% contrast so the edge survives the glare.

B. Give each card a distinct background hue so that the cards are told apart by colour rather than by any edge treatment at all.

C. Increase the shadow to 6px offset at 60% black so the cards clearly float.

D. Increase the gap between cards to 32px.

7 · 9 min

What the eye-tracking research actually says

THE ONE THING TO KEEP

The F-pattern describes how people skim a page with no hierarchy rather than prescribing where to put things, so the correct response to it is to give the eye structured landing points instead of reinforcing the shape.

A famous heatmap, widely misread

In 2006 a usability study published eye-tracking heatmaps of people reading web pages, and the aggregate images looked like the letter F: a heavy band across the top, a shorter band lower down, and a vertical stripe down the left. The F-pattern entered design folklore within months, and it is still taught as a layout rule — put the important things along the F.

This is a misreading, and it is worth understanding precisely, because the underlying research is good and the conclusion drawn from it is not.

What the study found

The participants were scanning text-heavy pages they had no strong intention to read carefully. The F shape is what happens when someone skims dense prose: they read the first line or two of a block, read progressively less of each subsequent line, and eventually drop to sampling only the first few words down the left edge.

So the F is a description of a behaviour under specific conditions — unstructured text, low commitment, no visual hierarchy to guide the eye. The authors were clear that it is a symptom of a page that gives the reader nothing to navigate by. Designing to reinforce the F is designing to make skimming permanent.

The same body of work makes the opposite recommendation: break the F by providing structure. Subheadings, short paragraphs, bulleted lists, bold on the first few words of a paragraph, and images with genuine information in them all pull fixations off the left edge and into the content.

What is better supported

Several findings from eye-tracking research are much more robust than the F, and more useful:

- **Entry point.** Readers of left-to-right scripts tend to begin near the top-left of the content area. Readers of Arabic, Urdu or Hebrew mirror this. It is a tendency, easily overridden by a strong visual element elsewhere — a large dark mass in the lower right will be fixated first, whatever the script.
- **Faces attract fixations**, early and reliably, and the direction a face is looking pulls attention along with it. A model looking at your headline measurably increases fixations on that headline compared with the same model looking at the camera.
- **Banner blindness is real.** Regions that look like advertising — right-hand column, rectangular, coloured, near the top — get systematically skipped, including when they contain content the reader is actively hunting for. Styling your own content like an advert makes it invisible.

- **Text beats images for attention when the reader has a task.** Decorative photography of the kind used to fill space is routinely skipped. An image that carries information — a chart, a product, a diagram — is not.
- **Numerals attract fixations in prose.** A figure written as 47% pulls the eye where forty-seven per cent does not.

The honest limits of heatmaps

This is where the field gets oversold, and knowing the limits will save you from arguments you cannot win with data.

A heatmap records where the eye pointed. It does not record what was understood, what was believed, or what was decided. A long fixation can mean interest or confusion, and the trace cannot tell you which. Aggregate heatmaps also hide individual variation: a smooth orange blob may be an average of twelve people who each looked at completely different things, and the average path may be one that nobody actually took.

Sample sizes in published eye-tracking work are usually small — tens of people, not thousands — and the equipment historically required a lab, a chin rest and a calibration step, which selects for a particular kind of participant behaving in a particular way. Webcam-based tools have made this cheaper and much noisier.

Finally, almost all of the widely cited web findings come from desktop screens in the 2000s and 2010s. Phone reading has a different geometry, a different holding posture and a different scroll behaviour, and the F does not transfer cleanly to a 390-pixel column where nearly every line is short.

What to do with all this

Use the research as a set of constraints, not a layout template.

1. **Do not rely on position alone.** Position is the weakest of the emphasis channels and the most culturally variable. Size, value contrast and isolation do the heavy lifting and work in every script.

2. **Assume skimming and design against it.** Structure the page so there is something to land on every few centimetres: a subheading, a pulled number, a short list.
3. **Do not style content like advertising.** Avoid the shape, position and treatment that readers have learned to ignore.
4. **Point faces at what matters**, if there is a face.
5. **Test with people rather than predicting.** Three people, three seconds each, asked what they saw, will tell you more about your specific layout than any published heatmap of somebody else's.

> *The useful summary of forty years of research: readers are looking for a reason to stop. Give them one, in a place you chose.*

BEFORE YOU MOVE ON

A team moves its key message into the top-left of the page and bolds the left-hand words of every paragraph, citing the F-pattern. Engagement does not improve. What is the flaw in the reasoning?

A. The F-pattern applies only to desktop screens, and the change would have worked if the site were not responsive.

B. Top-left is the entry point only for left-to-right scripts, so the change helps some readers and harms others.

C. The F is what skimming looks like when a page offers no structure, so reinforcing it entrenches skimming instead of giving the eye reasons to stop.

D. Bold text is too weak an emphasis channel on its own, and the key message would have needed a substantial size increase and a colour change before any reader registered it as a rank.

8 · 8 min

Weight is not area, and centre is not the middle

THE ONE THING TO KEEP

Visual weight is area times darkness times detail, not size, and both balance and centring should be judged on where the mass sits rather than where the bounding box is.

Two shapes of the same size can pull differently

A 200-pixel square of pale grey and a 200-pixel square of near-black occupy identical area and exert wildly different pull. That much is obvious. What is less obvious is that a 60-pixel square of near-black can outweigh the 200-pixel pale one, and that a dense paragraph of small text can outweigh a large photograph.

Visual weight is roughly the product of three things: how much area a mark covers, how far it sits from its background in lightness, and how much detail it contains per unit of area. A small, dark, busy element is heavy. A large, pale, empty one is light.

This matters because balance in a layout is a comparison of weights, not of sizes, and because designers routinely try to balance a photograph against a block of text by matching their dimensions, which is the wrong quantity.

Estimating weight without guessing

The blur test gives you a direct read. Blur the layout heavily — 3% of the width rather than 1% — until nothing but masses remain, and then squint. The blobs that remain darkest and largest are the heavy ones.

For a numeric version, most editors will tell you the mean value of a selection. In GIMP, select a region and open Windows > Dockable Dialogs > Histogram,

which reports the mean. A region with a mean of 60 out of 255 is heavy; a region with a mean of 235 is light. Multiply mean darkness by area and you have a rough weight you can compare across elements. This is cruder than the eye but it settles arguments.

A counter-intuitive consequence worth internalising: **a dense paragraph of 14px text is a mid-grey mass**, not a light one. Text at normal settings averages somewhere around 70 to 80% lightness when blurred — the black letters and the white gaps average out. That is why a column of body copy can hold its own against a photograph, and why increasing line spacing lightens a paragraph as surely as changing its colour does. Typographic colour, covered later in the type block, is exactly this property.

The optical centre

Put something in the exact vertical middle of a frame and it will look slightly low. This is consistent enough that typographers and framers have compensated for it for centuries: the optical centre sits a few per cent above the geometric one.

The practical numbers:

- For a title in a frame or on a title page, place the centre of the text block at about 45% of the height rather than 50%. On a 1000px frame, that is roughly 50 pixels higher than the mathematical centre.
- For traditional book pages, the margin at the bottom is set larger than the top — often by a third or more — for the same reason.
- For a single icon or glyph inside a circular or square button, the same principle applies vertically, and additionally the glyph's own optical centre rarely matches its bounding box.

Why it happens is not fully settled. The usual explanation is that we compensate for perspective and for the expectation that objects sit on a ground plane; whatever the cause, the effect is reliable and easy to test on yourself. Draw a dot at exactly half the height of a tall rectangle and it will look like it has sunk.

Optical centring of glyphs and shapes

Inside a button or an avatar, the mathematical centre of a bounding box is frequently wrong:

- **A play triangle** in a circular button looks off-centre when its bounding box is centred, because the triangle's mass is towards the flat left edge. Shift it right by roughly 5 to 8% of its width.
- **Round letters** such as o, e and c are drawn slightly taller than flat ones such as x and z, overshooting the baseline and the x-height by around 1 to 1.5%, so that they look the same size. This is the typeface designer compensating on your behalf.
- **A single capital letter** used as an avatar sits low if you centre it on its bounding box, because the box includes descender space the letter does not use.

The rule that covers all of these: centre the mass, not the box. Software centres the box because that is what it can compute. You are checking the mass, because that is what a reader sees.

When the tool and the eye disagree

They will disagree often, and the eye wins. This is not mysticism; it is that the alignment tool is solving a geometric problem and the reader is solving a perceptual one. The two coincide for simple rectangles and diverge for anything with uneven mass distribution — type, icons, photographs with a bright sky at the top, logos with a tail.

The workflow that survives this: align mathematically first, because it gets you to within a pixel or two for free, then look, then nudge. Record the nudge as a deliberate offset rather than leaving it as an untracked hand adjustment, so that the next person does not helpfully re-centre it and undo your work.

BEFORE YOU MOVE ON

A play-arrow glyph inside a circular button is centred perfectly by the layout tool, yet every reviewer says it looks shifted left. What is happening?

A. The button's circle is not truly circular, so the glyph is centred against a slightly asymmetric container.
B. The triangle's mass sits towards its flat left edge, so centring its bounding box leaves the perceived centre of mass to the left of the circle's centre.
C. The reviewers are seeing the optical centre effect, which shifts perceived position and needs a vertical rather than horizontal correction.
D. The glyph needs to be substantially larger, because a small shape inside a much larger container always appears displaced towards whichever edge of that container happens to be nearest to it.

9 · 8 min

Shape is a channel you are probably not using

THE ONE THING TO KEEP

Gross shape survives peripheral vision, greyscale, glare and colour blindness, so one deviant silhouette in a field of rectangles is a cheap and robust emphasis — provided it stays rare enough to mean something.

Everything is a rounded rectangle

Open almost any interface built in the last decade and inventory the shapes. Cards: rounded rectangles. Buttons: rounded rectangles. Input fields, badges, modals, images, avatars: rounded rectangles, with the avatars occasionally promoted to circles. The corner radius varies between 4 and 16 pixels and that is the whole vocabulary.

This is not a disaster. Rectangles tile efficiently, respect grids and hold text well, which is why they won. But it means an entire emphasis channel is sit-

ting unspent, and a channel nobody is using is the cheapest one to use.

What shape contrast buys

Put one circle in a field of rectangles and the circle is found instantly, from the periphery, at any size, in greyscale, by a colour-blind reader, in sunlight. Very few devices survive that many conditions.

The mechanism is the one from earlier in this block: peripheral vision keeps gross shape. It discards the 8px corner radius that distinguishes your card from your button, but it keeps the difference between a blob that is round and a blob that is straight-sided.

Practical uses:

- **Avatars as circles** among rectangular content. This is so standard it has stopped registering as a design decision, but it is one, and it works.
- **A single pill-shaped primary button** on a page of square-cornered secondary ones.
- **A diagonal or a curve** cutting across a layout of horizontals and verticals. One diagonal in an otherwise orthogonal design is loud; a design full of diagonals is noise.
- **A silhouette that is not a rectangle** — a cut-out product photograph on a plain ground beats the same product in a rectangular crop, because the object's own outline becomes a shape rather than being hidden inside a box.

That last one is the most under-used device available to a small business making its own materials, and it is free: remove the background from a product photograph and place it on flat colour. GIMP does this with the Foreground Select tool or Layer > Transparency > Add Alpha Channel plus the Free Select tool; Photopea has a one-click Magic Cut in the browser; Krita has a comparable selection workflow. None of them requires a subscription.

Shape carries meaning, weakly and locally

There is a long tradition of claiming that shapes have inherent emotional content — circles are friendly, triangles are aggressive, squares are stable. Treat this carefully. There is some experimental support for a general preference for curved over angular contours, and it is weak, culturally variable, and swamped by context. A rounded button in a hostile interface is not friendly.

What is more defensible is the learned conventions, which are strong precisely because they are learned:

- A triangle pointing right means play, almost universally on screens.
- A circle among rectangles means person, in most interfaces.
- A rounded-corner rectangle that is more rounded than everything around it means tag or pill or status.
- An octagon in red means stop, in the many countries that adopted the Vienna Convention road signs — and this is a good example of a convention that feels universal and is not.

Build on conventions your audience actually holds, and do not assume that a shape means what a moodboard told you it means.

The corner radius question

Since rectangles are the vocabulary, the radius is where most of the decisions get made, and there is one technical point worth knowing.

A radius should be consistent in its *relationship* to the element, not in its absolute value. If a card at 400px wide has a 12px radius and a button at 40px tall also has a 12px radius, the button looks much rounder, because 12px is 30% of its height and 3% of the card's width. Systems handle this by defining two or three radius tokens — say 4px for small controls, 8px for cards, and fully rounded for pills — rather than one value everywhere.

The second point: when you nest a rounded object inside another, the inner radius should be the outer radius minus the padding, or the two curves will not run parallel. An 8px-radius card with 8px of padding wants a 0px-radius

element inside it, not another 8px one. Most people never notice this consciously and everyone notices it unconsciously.

Restraint

Shape contrast works because it is rare. One deviant shape per view is an emphasis; five are a style, and a style is not an emphasis. If you introduce a circle to mark the primary action, do not also use circles for decoration, bullets and icon backgrounds in the same view, or you have spent the channel on nothing.

The same logic covers illustration, texture and any other formal device: the first one is information, the fourth is wallpaper.

BEFORE YOU MOVE ON

A design system defines a single 12px corner radius for every element. On the built product, the 40px-tall buttons look like pills while the 400px cards look almost square. Why?

A. The buttons are being rendered at a different device pixel ratio from the cards, so the declared radius value is not actually being applied consistently across the two kinds of element.

B. Radius is perceived relative to the element's smallest dimension, so 12px is nearly a third of the button's height and a few per cent of the card's width.

C. A 12px radius is simply too large for buttons and too small for cards, so the system should use 8px throughout instead.

D. Buttons carry a fill and cards do not, and a filled shape always reads as rounder than an outlined one of the same geometry.

10 · 8 min

Content that attracts attention whatever you do to it

THE ONE THING TO KEEP

Faces, numerals, a reader's own identifying words and any motion attract fixations regardless of styling, so inventory what is already pre-emphasised on a surface before you allocate size, value and isolation around it.

Some things are pre-emphasised

Everything in this block so far has treated attention as something you allocate through formal choices: size, value, isolation, shape. That model is incomplete, because certain kinds of content pull fixations regardless of how they are styled, and if you ignore them your carefully budgeted hierarchy gets overruled by the content itself.

This is the most common reason a layout tests badly despite being formally correct. The designer ranked the elements; the reader's visual system ranked the meanings.

Faces

Human faces attract fixations early and reliably, including faces that are small, low-contrast, partially occluded or cartoonish. Face detection appears to be a specialised and fast process, running before you have consciously identified anything. Newborns orient towards face-like arrangements of two dots above a line within days of birth, and adults will find faces in plug sockets and clouds.

Two consequences for design:

1. **You cannot out-style a face.** A 60-pixel photograph of a person in the corner of a page will be looked at before a 40-pixel headline. If you want

the headline first, the face has to move, shrink drastically, or be removed.

2. **Gaze direction transfers.** A face looking at something pulls attention towards it. This is well-replicated: participants shown a model looking at a headline fixate the headline sooner and more often than participants shown the same model looking at the camera. A model looking straight out engages the viewer and holds them; a model looking at your product moves them along.

The practical rule is to point faces into the layout rather than out of it, and to treat any face as already occupying the top of your emphasis budget.

Numerals

A figure set as 47% is fixated where forty-seven per cent is not. Digits are visually distinct from the surrounding lowercase letters — different shapes, usually taller, often with more open counters — and they carry the promise of a specific fact, which is exactly what a skimming reader is hunting for.

This is the cheapest device in this lesson. If a paragraph contains a number worth knowing, set it as a numeral. If a page has one fact that should survive skimming, a large numeral will survive it. It is also why pricing, statistics and dates get skimmed accurately from pages where the prose is not read at all.

Be careful with the corollary: a page full of numerals has no attracting numerals, and a long table is a field of digits where nothing stands out. That is a job for alignment and tabular figures, covered in the type block.

Your own name, and words that name you

A reader's own name is detected in text with unusual reliability, and so, more weakly, are words that describe them — their city, their job title, their language, their team. This is the mechanism behind personalised subject lines working, and it is also the mechanism behind them feeling intrusive once readers learn that a machine inserted the name.

The design point is narrow but real: a word that identifies the reader is a strong emphasis channel you did not have to style. Use it where it is honest — naming the city a shop is in, naming the course somebody is enrolled on —

and be aware that using it where it is not honest is the recognisable texture of spam.

Motion, and why it is a weapon

Covered earlier as a peripheral-vision effect, but it belongs on this list: motion in the periphery captures attention involuntarily. It cannot be budgeted alongside the other channels because it does not compete with them, it overrides them.

This is why the ethical line sits here more clearly than anywhere else in visual design. An auto-playing video beside an article is not asking for attention; it is taking it, from a reader who cannot decline without leaving. Operating systems and browsers now expose a reduce-motion preference precisely because so many products spent this channel carelessly, and respecting that preference is a two-line change in any modern stylesheet.

What this means for the budget

Add a step to the emphasis budget from earlier: before you allocate anything, inventory what is already pre-emphasised on the surface.

```
faces present?      → already rank 1, wherever they sit
numerals present?   → strong pull inside prose
motion present?     → overrides everything, including rank 1
reader's own data?  → strong pull, easily overused
```

Then allocate your formal channels around that inventory rather than in spite of it. If a page has a large photograph of a person and you need the headline read first, the honest options are to remove the photograph, crop out the face, or accept that the face is the headline and write accordingly.

The fights that go on for weeks in design reviews are very often this: one person allocating formal emphasis, another person responding to content emphasis, and neither saying which they mean.

BEFORE YOU MOVE ON

A landing page has a 500px photograph of a smiling person looking straight at the camera, beside a headline the team wants read first. Enlarging the headline by 40% does not change what testers report seeing first. What is the most defensible next move?

A. Add motion to the headline, since motion overrides every other channel including a face.
B. Increase the headline again, and keep increasing it, until it clearly outweighs the photograph in both area and darkness.
C. Crop the face out, shrink the photograph substantially, or turn the model to look at the headline.
D. Desaturate the photograph so the headline wins on colour contrast.

11 · 10 min

Nine checks that find what is wrong with a layout

THE ONE THING TO KEEP

Diagnose in order — hierarchy, value, spacing, alignment, inventory — because a fault in structure cannot be fixed with colour, and colour work done first gets thrown away.

Why you cannot see your own work

"It looks bad and I do not know why" is the most common sentence in design, and the reason is not inexperience. It is that you built the thing. You know what every element means, which item is the important one, and what the client asked for. Your reader has none of that, and your memory of your intention sits between you and what is actually on the page.

Diagnosis is a set of procedures for getting the information without the memory. It is boring, mechanical and reliable, which is exactly what you want.

Run them in this order

The order matters. Fixing colour before structure is painting a wall that is not straight — the paint is fine and you will do it twice.

1. Squint, or shrink to a thumbnail. Question: is there one thing you see first? If three things arrive together, that is a hierarchy fault. Fix by demoting, not promoting.

2. Desaturate it. Question: does the structure survive in grey? If the layout falls apart, you built your hierarchy out of hue, and hue is the channel that fails in sunlight, on cheap screens, in a photocopy and for one man in twelve.

3. Measure two gaps. Take any group. Is the gap *inside* it as large as, or larger than, the gap *around* it? This is a proximity fault, and it is the most common single defect in beginner work. Headings with equal space above and below are the classic instance.

4. Count the left edges. Draw the vertical alignment lines with a thin rectangle. More than three or four, and each extra one needs a reason. Usually two of them are the same intended line, three pixels apart.

5. Take an inventory. Count typefaces, type sizes, weights, colours, corner radii, border widths. Rough ceilings for one piece: two typefaces, four sizes, three weights, five colours, one corner radius. Over those numbers, you are not using variety, you are accumulating leftovers — often from a template that arrived with three fonts already in it.

6. Compare the outer margin with the biggest inner gap. If the margin is smaller, the content looks like it is sliding off the edge, because the strongest boundary on the page is in the wrong place.

7. Look for trapped space. A pocket of emptiness in the middle of a group, wider than the group's own margin, splits it in two whether you meant it to or not.

8. Check the four edges and the tangents. Anything cut at a joint, anything just touching something else, anything that ends three pixels short of an edge it was meant to meet.

9. Read the words. Honestly, this is half of it. A great deal of what presents as a design problem is forty words of copy that should be twelve. No arrangement of forty words is as strong as twelve good ones, and cutting them is free.

A worked autopsy

A real kind of example: a flyer for a small restaurant's opening offer.

What was on it — a photo of the food filling the top half; the restaurant name in a script face; "30% OFF FIRST WEEK" in a bold condensed face; the address centred under a line; the phone number in the same size as the address; opening hours; a border drawn around the address block; four typefaces, three of them inherited from the template.

Running the checks:

- **Squint:** the photo, the name and the offer all arrive together. No first thing.
- **Grey:** the offer disappears into the photo, because it was only separated by being red.
- **Gaps:** the address block has more air inside it than between it and the hours.
- **Edges:** the name is centred, the offer is left-aligned, the address is centred again. Three axes in one column.
- **Inventory:** four typefaces, six sizes, four colours.
- **Margin:** 6mm outer margin, 12mm gaps inside.

Three edits fixed most of it, and none of them added anything:

1. The photo dropped to a background, its contrast reduced, so it stopped competing.
2. The offer became roughly twice the size of everything else and the restaurant name became second — because a person walking past needs the offer

first and the name second, which is a business decision, not a taste one.

3. The address, phone and hours collapsed into one small left-aligned block, one typeface, one size, and the border around it was deleted, since space was already doing that job.

No new colours. No new fonts. Nothing added.

The fix is usually subtraction

The reflex when something looks unfinished is to add: a border, a gradient, a drop shadow, an accent colour, a divider line. Every addition is a new competitor for attention and a new thing to align.

Reverse the reflex. Take something away and look. If removing an element does not damage the message, it was decoration, and decoration on an unranked layout makes the ranking harder to see rather than easier.

The limitation worth being honest about

Some layouts cannot be fixed by arranging them, because the content is wrong. If five things are genuinely equally important, no design makes five things first. What you are being asked to do is impossible, and pushing pixels around is a way of avoiding saying so.

The response is not a better grid. It is a conversation: what can be cut, what can move to a second page, a second screen, or the back of the card. That is a content decision wearing a design costume, and recognising it is a large part of being useful rather than decorative.

Today

Take something you made a month ago — far enough back that you have half-forgotten it. Run all nine checks and write the faults down **before** you change anything. The written list is the point; it stops you fixing the first thing you notice and calling it done.

BEFORE YOU MOVE ON

A flyer looks wrong. The designer's first move is to change the colour palette and add a drop shadow to the heading. Why is that the wrong order?

A. Drop shadows are dated and palettes are a matter of taste, so both should be left until the end.

B. A palette change is a larger job than a spacing change, so it belongs later in the schedule.

C. Colour should not be touched before the client has approved a direction, because colour is subjective.

D. Colour and effects sit on top of structure; if nothing outranks anything or the gaps encode the wrong groups, new colour hides the symptom and the work has to be redone once the structure changes.

CASE STUDY · MODULE 1

The vaccination poster with five first things

A district health office in Nashik, preparing a poster for a two-day measles and rubella camp at eleven anganwadi centres.

The camp had a budget for 4,000 posters and one week to get them pasted. A junior officer with a laptop and a free browser editor made the first version in an afternoon, and it contained everything anyone had asked for.

Across the top: the department's logo, the state emblem, and a photograph of the district health officer. Below that, in a decorative script, "Measles and Rubella Free Nashik". Then the dates, the words "free of cost", a list of the eleven centres in two columns, a helpline number, a QR code, and a line of small text naming the scheme the funding came from. Every element was large. Several were in red, because red had been described in the meeting as urgent.

What the review meeting was about

The poster was circulated, and the comments arrived in the usual shape. The programme officer wanted the helpline bigger. The communications cell wanted the logo bigger. Somebody wanted "free of cost" in a bigger red. Nobody said the poster was hard to read, because nobody looks at a poster they have already read four times.

An intern on placement from a polytechnic did two things that cost nothing. She took a screenshot, desaturated it, and blurred it. Everything came back as one even grey field with a darker rectangle where the photograph was. Then she printed three copies, showed them to three people in the corridor for three seconds each, took them away, and asked what they had seen.

One said "a government thing". One said "something about vaccination". One named the district health officer, who she recognised. Nobody said a date. Nobody said a place.

The decision

The fix was arithmetic, and it was also political. A poster read from four metres, by a parent walking past with a child on one hip, can carry one thing first and two things second. The intern's proposal was:

- **First:** the dates, at roughly four times the size of everything else.
- **Second:** the words "free vaccination" and the name of the local centre, with a blank space left for each centre's name to be overprinted.
- **Everything else** at the floor: logo small in a corner, helpline small, QR code small, scheme credit small, and the photograph removed.

Removing the photograph was the part that stopped the room. It was not decoration in the ordinary sense. The photograph was there because it had been there on the last four campaign posters, and because taking it off was a statement somebody would have to make to the officer whose photograph it was.

The cost on the other side was concrete and countable. The blurred image had already shown that the photograph was the darkest, largest mass on the poster, and a face attracts fixations whatever is done to it. Leaving it in meant the strongest element on a poster about a date was a person's face. Eleven centres, two days, a target of 9,000 children, and one chance for a parent to catch the information from across a road.

The programme officer put the trade-off in a single sentence in the meeting: if the photograph stays, the date does not get read; if the date does not get read, the camp is under-attended, and the photograph is on a poster about a camp that failed.

There was a second, smaller decision hiding inside the first. The eleven centres could not all be named at readable size on one poster. Naming them all meant a list at 14 points that nobody would read at four metres. Printing eleven versions, each naming one centre, cost more in plate setup and made the print run harder to manage with a week to go.

STOP HERE

Before turning the page: what would you have done, and what would it have cost if you were wrong? Write it down. The point of the next section is to compare.

WHAT HAPPENED

The photograph came off, and the district health officer was told why before the file went to the press rather than after. The dates went to roughly 120mm tall on an A3 sheet, with "free vaccination" second and a blank band left in the lower third. Eleven short runs were printed with the centre name and its timing overprinted in that band, which cost about eleven per cent more than one run of 4,000 and removed the unreadable list entirely.

The three-second test was repeated in the corridor with the new version. All three people named the dates. Two named the words "free vaccination". None of them mentioned the logo, which was the intended result rather than a failure.

Attendance against target was reported at the end of the camp as higher than the previous year's, which the office attributed partly to the poster and partly to better ASHA follow-up. Nobody could separate the two, and the honest conclusion drawn in the debrief was narrower: the poster now said one thing at four metres, and the previous one had said nothing at all.

The reviewers all asked for something to be made bigger. Why did granting each of those requests make the poster worse?

Emphasis is a relationship between an element and everything around it, not a property of the element, so there is a fixed amount of it on one surface. Every promotion demoted the rest, and after four rounds every item was loud, which is perceptually identical to every item being quiet. The productive response to 'make this bigger' is the question 'what comes down'.

The intern's blur test and the three-second test answered different questions. What were they?

The blur test asks what can be found: it shows which elements survive as distinct masses and where they sit, which is what a reader's peripheral vision is working with before they look at anything. The three-second test asks what actually arrives first in a real person's head, including content effects such as a recognisable face that no formal analysis of size and contrast would predict.

Somebody argues that removing the photograph was a design preference, not a finding. What evidence in the case answers that?

Faces attract fixations early and reliably regardless of styling, and the blurred image showed the photograph was also the largest dark mass on the sheet. Both the formal channel and the content channel put it at rank one on a poster whose message was a date. The three-second test then confirmed it behaviourally: a viewer named the officer and nobody named a date.

MODULE 1 · TEST

Check what stuck

6 questions, one right answer each. This one is for you, not for a record: answers and the reason behind each are at the back. If two questions from the same module go wrong, re-read that module before the exam rather than after.

1 A poster sets its heading at 17 pixels above body text at 16 pixels. Why does this fail to create a hierarchy?

A. The reader cannot tell whether the difference was meant, so it carries no rank

B. Seventeen is not a value on a standard modular scale, so browsers round it inconsistently

C. A heading must be at least twice the body size before it is legible as a heading

D. Both are set in the same typeface, so size alone can never separate them

2 A product page has the price at 24px bold black, a solid-colour Add to Basket button, a red promotional badge and a 20px bold delivery promise. The client wants the button seen first. Which change is most likely to achieve it?

A. Make the button larger and more saturated than it is now

B. Set the delivery promise to 16px regular grey and the badge to a quiet outline

C. Move the button to the top left, because that is where the eye enters the page

D. Add a drop shadow and a small animation so that the button separates from the page

3 The squint test specifies a luminance conversion to greyscale rather than an average of the red, green and blue channels. Why does the difference matter?

A. A luminance conversion also darkens the image, which makes the masses easier to rank
B. It keeps the image's colour profile intact, which an average discards
C. An average turns a saturated yellow and a saturated blue into similar greys
D. Averaging is only available in paid software, so the free path needs luminance

4 Isolation is described as a stronger emphasis channel than its cost suggests. What is the mechanism?

A. Peripheral vision resolves an isolated item at full central acuity
B. An isolated item is fixated first because readers begin at whitespace
C. Isolation increases the item's contrast against the background
D. Isolation removes the neighbouring marks that were smearing it

5 A card has a 1px border and a hard 2px drop shadow, sits 8 pixels from its neighbour, and still does not read as a separate object. What is the strongest repair?

A. Give it 32 pixels of space and delete the border and shadow
B. Thicken the border to 2px and darken its value
C. Add a second, lighter border inside the first to imply an edge
D. Increase the shadow's blur and its opacity

6 What does the F-shaped heatmap from eye-tracking studies of web pages actually describe?

A. The order in which readers prefer to receive information on any kind of page
B. How people skim dense prose that gives them no structure to navigate by
C. A layout template that puts the important elements along the F
D. The path the eye takes when a page has a strong visual hierarchy

MODULE 2

Space, grouping and the grid

Space is the cheapest material in design and the one beginners spend least. This block turns spacing from a feeling into a system: which grouping cue wins when two of them disagree, a scale that makes every gap a deliberate decision, the arithmetic of columns and gutters, where optical alignment departs from mathematical alignment, and how big a thing has to be before a thumb can hit it.

> BY THE END OF THIS MODULE YOU CAN
>
> Set every gap in a layout from a single stated spacing scale, name the relationship each gap encodes and the grouping cue carrying it, lay out a column grid from its arithmetic rather than by eye, and say where an optical adjustment must override the alignment tool

---

12 · 9 min

The gap is doing the talking

> THE ONE THING TO KEEP
>
> The eye assigns things to groups by distance before it reads a word, so the space inside a group must be obviously smaller than the space around it.

The most common fault in beginner work

A form has a label above each field. The label sits 14 pixels above its box. The next label sits 16 pixels below that box. People keep typing their phone number into the email field, and nobody can say why.

The reason is that the gaps are lying. To the eye, a label 14 pixels from one field and 16 from another belongs to neither, or to both. The words are correct. The spacing says something different, and the spacing is read first.

This is worth taking seriously because it costs nothing to fix and it is invisible to the person who built the thing. You know which label goes with which field, because you typed them in that order. Your reader has only the distances.

Grouping happens before reading

Psychologists working in Berlin in the 1920s catalogued the ways vision assembles a scene into objects before any conscious thought. Three of them do most of the work in layout:

- **Proximity.** Things close together are one thing.
- **Similarity.** Things that look alike are one kind of thing.
- **Common region.** Things inside the same boundary, or on the same patch of colour, are one thing.

These are not preferences. They are how the visual system delivers the page to you. By the time you have read a word, the grouping has already happened. That is why a wrong gap beats correct words: the words arrive second.

One number, and it is not subtle

The gap inside a group must be visibly smaller than the gap around it. Not slightly smaller. Obviously smaller — aim for a factor of three or four.

Concretely:

- Form label 4 pixels above its input; 32 pixels to the next field.
- Caption 8 pixels under its photo; 48 pixels to the next photo.
- A heading 8 pixels above its paragraph; 40 pixels below the previous paragraph.

Look at that last one, because it catches nearly everyone. Software defaults put equal space above and below a heading. A heading belongs to the text under-

neath it, not the text above it. Move it down towards its own paragraph and a page of text becomes navigable without a single other change.

The failure mode has a name too: **uniform spacing**. When everything sits 16 pixels from everything else, the eye receives no grouping information at all and reads a flat list of unrelated items. This is why a tidy, evenly spaced layout can feel harder to use than a scruffy one — the scruffy one at least has accidental groups.

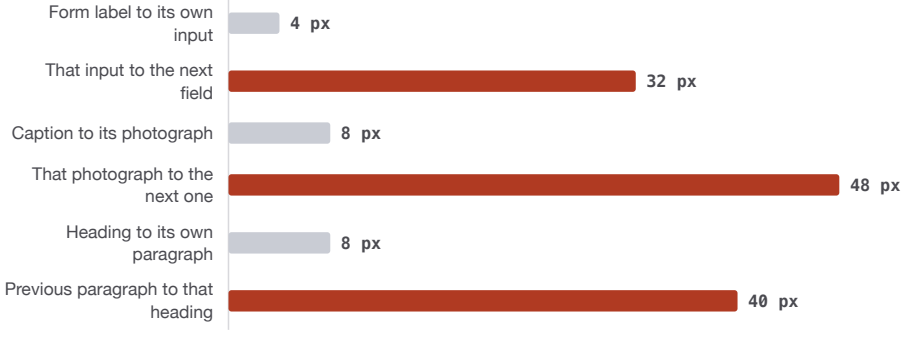
Repetition is the difference between designed and assembled

Repetition means: the same kind of thing gets the same treatment, every time. Every second-level heading identical. Every card the same padding. Every icon the same weight and size. Every gap drawn from the same short list of numbers.

This is not decoration and it does not require talent. It is a decision to stop inventing. Most work that reads as amateur is not badly drawn; it is inconsistently drawn — three different card paddings because three cards were made on three different evenings.

The mechanism is worth holding on to. Repetition builds an expectation in the reader, and an expectation is the only thing a break can be measured against. If nothing on your page is consistent, you have no way to signal that something is special. You spent your signal before you needed it.

The gap inside a group against the gap around it



Every pair differs by a factor of four or more. Equal space above and below a heading is the software default, and it attaches the heading to the wrong text.

Contrast means clearly different, not slightly different

The counterpart rule: if two things are not the same, make them clearly not the same. Two greys eight per cent apart. Two sans-serifs with similar proportions. A 15 pixel gap and an 18 pixel gap. Each of these reads as sloppiness rather than distinction, because the viewer's first hypothesis for a small difference is that you failed to line something up.

Either commit or collapse them. Two greys that are nearly the same should become one grey.

Boxes are the last resort, not the first

The instinct when a group is not reading as a group is to draw a border around it. Understand what that costs: a border is a line, a line is a graphic element, and now the page has more elements competing for attention than it did before. Panels stacked inside panels are how a clean layout turns into a control room.

Try these in order:

1. Move the items closer together.
2. Add space around the whole group.
3. Give the group a faint background tint — common region without a line.
4. Only then, a border.

Most groups resolve at step one or two.

Trapped space

One more failure that is hard to see until you are told: whitespace stranded in the middle of a group, bigger than the space at the group's edge. A pocket of emptiness between two columns, wider than the page margin. The eye reads that pocket as a divider and splits the group in half. Space at the edge of a group holds it together; space inside it pulls it apart.

Today

Take any layout you have made. Screenshot it. Open it in Photopea, GIMP or your phone's photo editor, and use the rectangular selection tool to measure two gaps: the gap inside a group and the gap that surrounds it. Almost every beginner file has at least one place where the inside gap is equal to or larger than the outside gap. Fix that one, and look again.

BEFORE YOU MOVE ON

A signup form sets each label 14 pixels above its input box, and 16 pixels between one input box and the next label. Users keep typing into the wrong field. What is the mechanism?

A. The labels are too small to read at a glance, so people are guessing from field position instead.

B. Each input needs a visible border so that the field is contained and unambiguous.

C. The gap inside each label-and-field pair is almost identical to the gap between pairs, so the eye gets no grouping information and can attach a label to either field.

D. The form is too long, and splitting it into steps would stop people losing their place.

13 · 9 min

When two grouping cues disagree

THE ONE THING TO KEEP

Grouping cues have different strengths — a bounding region beats distance, distance beats colour — so a proximity error cannot be repaired with similarity, and a group that distance cannot express needs a boundary rather than a shared hue.

Grouping is a vote, not a rule

Proximity, similarity and common region each push elements into groups. Most of the time they agree, and nobody has to think about it. The interesting cases — and almost every genuinely confusing layout — are the ones where two cues point at different answers, and you need to know which one wins.

A worked example. A form has eight fields in two visual columns. The labels sit close above their inputs, which groups by proximity. The designer then colours the four required fields' labels red, which groups them by similarity. A reader now sees two incompatible structures: four label-and-field pairs, and one scattered set of four red things. Both readings are available, the eye keeps flipping between them, and the form feels harder than it is.

The rough order of strength

Working from the experimental literature and from what survives on real screens, the cues rank roughly like this, strongest first:

1. **Uniform connectedness.** Elements that are physically joined — sharing a continuous region, a background patch, a border, or connected by a line — group before anything else. A box beats everything.
2. **Proximity.** Distance is the next strongest and the one you control most easily.

3. **Similarity.** Shared colour, shape, size or weight. Real, but it loses to both of the above.
4. **Continuity.** Elements that fall on a smooth path, especially a shared edge, read as a sequence.
5. **Closure.** The eye completes implied shapes and boundaries that are not drawn.
6. **Common fate.** Elements that move or change together group; only available where there is motion.

The practical consequence: **you cannot fix a proximity error with colour.** If a label is closer to the wrong field, making it the right colour will not rescue it. Move it. Conversely, if you need a group that distance cannot express — because the items are scattered across a table — reach for common region, not similarity, because a shared background patch is a stronger cue than a shared hue.

Continuity and closure are the ones you can exploit

These two are under-used, and they save ink.

Continuity means a list of items sharing a left edge reads as one column even when the gaps between them are large. That is why a well-aligned layout can afford generous space without falling apart, and why a ragged left edge forces you to compensate with boxes and rules you would not otherwise need.

Grouping cues, in rough order of strength

Uniform connectedness	A shared region, a border, a joining line. A box beats everything.
Proximity	Distance, and the cue you control most easily
Similarity	Shared colour, shape, size or weight. It loses to both above.
Continuity	Elements on a smooth path, especially a shared edge
Closure	The eye completes boundaries nobody drew
Common fate	Things that move together, where there is motion at all

A proximity error cannot be repaired with colour. Where distance cannot express a group, reach for a common region rather than a shared hue.

Closure means you rarely need to draw every boundary. A table with no lines at all still reads as a table, because aligned columns imply the rules. The standard advice to remove every vertical rule from a table and most of the horizontal ones works because closure does the job for free, and because each line you remove is one fewer thing competing for the eye. Try it on your next table: delete all the rules, keep the alignment, add a single rule under the header row. It almost always improves.

Common fate appears in interfaces as shared motion. Items that slide in together are read as one set. This is why a staggered animation, where list items appear one after another, actively weakens the grouping it was meant to decorate.

A diagnostic

For each visual group you think exists, write down which cue is carrying it. Then look for the ones carried by a single weak cue.

```
card header + body     → common region (the card)        strong
filter chips row       → proximity + similarity
strong
required-field markers → similarity only                  weak
footer links           → proximity + continuity           ade-
quate
related products       → similarity only, items scattered weak
```

The two weak rows are where readers will report confusion. The repair is to add a stronger cue, not a second weak one: put the related products in a bounded region, and move the required marker next to the thing it modifies.

When you want ambiguity

Occasionally two readings are both correct and you want both available: a grid of photographs that reads as rows and as columns, a table you might scan either way. Here the answer is to make the competing cues deliberately equal in strength — identical gaps in both directions, no boxes — so the ambiguity looks like a property of the content rather than an unresolved argument.

What fails is *unequal* competition: a 20px horizontal gap against a 24px vertical one. That is close enough that neither reading wins and far enough apart that the eye notices something is wrong. Either make them equal or make them obviously different.

The honest limitation

The grouping principles come from demonstrations made in Berlin in the 1910s and 1920s, and for most of their history they were exactly that: compelling pictures rather than measured laws. Later work has put numbers on some of them — proximity in particular behaves predictably as distances change — but the relative strengths above are a working ordering rather than a constant of nature. They shift with the display, the density and what the viewer is doing.

Treat them as a checklist for generating hypotheses about why a layout confuses people, and then test the layout on somebody. The checklist will point you at the right three candidates; it will not tell you which of the three it was.

BEFORE YOU MOVE ON

A dashboard shows twelve metric tiles in a 4x3 arrangement. Three related tiles, scattered across different rows, are tinted blue to show they belong together. Users consistently fail to see them as a set. What is the mechanism?

A. The blue tint is too subtle, and a more saturated blue with a coloured border around each tile would make the relationship register.

B. Grouping by colour only works for two items, and any set of three or more requires an explicit label naming the relationship.

C. Similarity is a weaker cue than the proximity of the grid, which is grouping each tile with its neighbours instead.

D. Blue is a receding colour in the layout, so the tinted tiles sit behind the others and lose the attention needed to be read as a group.

14 · 8 min

Every gap comes from a list

THE ONE THING TO KEEP

Every gap in a design should be a member of one multiplicative scale with a meaning attached to each step, so that spacing becomes a statement about relationships rather than a series of independent guesses.

The fault you can find in ten seconds

Open any layout made without a system and measure the gaps. You will find something like 6, 9, 10, 14, 15, 18, 22, 25, 30 pixels, with no two the same and no reason behind any of them. Each was set by dragging something until it looked right, in a moment, against whatever was next to it at the time.

This is the most reliable cause of work that looks amateur without looking obviously wrong. No single gap is a mistake. The absence of a pattern is the mis-

take, because the reader is constantly asked to decide whether a 14px gap and a 15px gap mean different things, and the answer keeps being no.

The scale

Pick a base unit and build a ramp on it. Four pixels is the usual base, for a practical reason: it divides cleanly at the device pixel ratios real phones use, so nothing lands on a half pixel and blurs.

```
4   8   12   16   24   32   48   64   96
```

Nine values, which is more than most designs need. Notice the shape: small steps at the bottom, doubling at the top. That is deliberate, and it follows from the threshold lesson in the first block — a difference registers in proportion to what it is compared with, so a constant 4px step would become invisible by the time you reach 40 and 44.

Some systems use a strict 8px base with a 4px half-step, which is where the widely used 8-point grid comes from. Others use a modular ratio. The specific ramp matters far less than the fact that there is one.

Assign meanings, not just numbers

A scale becomes useful when each step owns a job:

- **4** — space between a glyph and its label, inside a badge.
- **8** — inside a group: label to input, icon to text.
- **16** — between related items in a list; page gutter on a phone.
- **24** — between groups within a section.
- **32 / 48** — between sections.
- **64 / 96** — between major regions, or above a section on a large screen.

With meanings attached, the question when you are laying something out stops being how big should this gap be and becomes what is the relationship here, which has an answer you can defend to someone else.

Implementing it for free

In CSS, custom properties make the scale the only source of spacing values:

```
:root {
  --space-1: 0.25rem;  /*  4px */
  --space-2: 0.5rem;   /*  8px */
  --space-3: 1rem;     /* 16px */
  --space-4: 1.5rem;   /* 24px */
  --space-5: 2rem;     /* 32px */
  --space-6: 3rem;     /* 48px */
}
.field { margin-bottom: var(--space-4); }
.field label { margin-bottom: var(--space-2); }
```

Using `rem` rather than `px` means the whole scale grows when a reader increases their base font size, which is the behaviour you want and which a pixel scale silently refuses to give them.

In drawing tools, set the document grid to the base unit and enable snapping. Inkscape: File > Document Properties > Grids, with Snap to grid on. Krita and GIMP both offer a configurable grid under View. Figma's free tier has layout grids and an 8px nudge setting. None of this requires a paid tool.

The audit

List every distinct gap value in a finished design and count them. More than about six on a single page means you have gaps that were never decided. Map each stray onto the nearest scale value and delete it. Almost always the design improves immediately, because several accidental near-differences collapse into either a real difference or none.

In a browser, developer tools give you this for free: inspect an element and the box model diagram shows the computed margin and padding, which is what shipped rather than what you meant.

Where to break it

The scale governs relationships between objects. It does not govern optical corrections, which are the subject of a later lesson in this block. If a 16px gap between an icon and its label looks like 18px because the icon's artwork has built-in padding, you fix that with a 14px value and a note explaining why.

The distinction is worth stating precisely, because it is the difference between a system and a superstition:

> *Structural spacing comes from the scale. Optical corrections are exceptions, they are small, and each one has a written reason.*

A design with three documented exceptions is a system. A design with thirty undocumented ones is where you started.

What it does not buy you

A spacing scale makes a layout consistent. It does not make it well grouped — you can apply the scale perfectly and still put 24px inside a group and 16px around it, which is the error from the previous lesson expressed in tidier numbers. The scale is a vocabulary. Grouping is what you say with it.

BEFORE YOU MOVE ON

A team adopts a spacing scale of 4, 8, 12, 16, 20, 24, 28, 32, 36, 40 and finds the layout still looks arbitrary. What is the most likely reason?

A. A scale needs to be built on multiples of the body font size rather than on a fixed pixel base, so the values should be derived from 16px.

B. Ten evenly spaced steps offer almost no distinguishable gap sizes at the top, so choices there remain effectively arbitrary.

C. The base unit is too small, and rebuilding the same additive scale on an 8px base would give the design the structure it lacks.

D. Pixel values cannot produce a consistent rhythm on screens with different device pixel ratios, so the scale must be expressed in relative units before it will read as deliberate.

15 · 8 min

Space is content you have not been billed for

THE ONE THING TO KEEP

Space is a working mechanism — isolation, decrowding and grouping — rather than an unfilled area, but it is paid for in scrolling on small screens, so spend it on the boundaries that carry meaning and compress the ones that do not.

The argument you will have

Somebody senior looks at a layout and says there is too much white space, we are wasting the page, can we bring things up. This is one of the most common conversations in design and one of the least productive, because both parties are arguing about the wrong quantity.

The quantity that matters is not how much of the page is covered. It is how much of the page gets read. Those move in opposite directions past a certain density, and the crossover arrives earlier than most people expect.

What space actually does, mechanically:

- It is the **isolation channel** from the first block — the only way to make something prominent without adding any ink to it.
- It **removes the crowding** that destroys peripheral legibility, which is why an isolated item can be found and a packed one cannot.
- It carries the **grouping information** that tells a reader what belongs with what.

None of that is decoration. Remove the space and you have removed three working mechanisms and gained some square centimetres.

Macro and micro

Two different budgets, and beginners almost always get one right and the other wrong.

Micro whitespace is the space between words, between lines, between a label and its field, inside a button. Defaults handle most of it tolerably.

Macro whitespace is the space between major regions: around the content column, above a section, between a header and the first thing under it. This is the one that gets starved, because it is the one that visibly costs area, and it is the one that does the most work.

If a layout feels cramped, check the macro budget first. Doubling the space above each section heading often fixes a page that somebody was about to redesign.

The printed precedent

Books made when printing was expensive still gave up enormous margins. In a well-made book the text block covers roughly half the page, and the outer and bottom margins are the largest. Classical page constructions such as the Van de Graaf canon place the text block at a fixed proportion of the page so that the margins hold consistently across sizes.

These were not extravagant decisions by people with paper to spare. They are decisions by people who knew the text block is what you look at and the margin is what makes it a block rather than a field. That reasoning transfers directly: on screen, the content column is the design, and the viewport is the page.

The real cost, which is not area

Here is where the honest version departs from the usual advice. Whitespace is not free on a phone. It costs scrolling, and scrolling costs attention and thumb work. A layout with 96px between every element on a 390-pixel-wide screen turns an article into a half-hour of swiping, and readers do abandon it.

This is a genuine trade-off with no universal answer. What resolves it in practice:

- **Keep horizontal breathing room; compress vertical rhythm on small screens.** A 16px side gutter is not negotiable; a 96px gap between sections can become 48px.
- **Spend space on the boundaries that matter and take it from the ones that do not.** A large gap above a section heading is worth more than an equally large gap between every paragraph.
- **Let the space scale with the screen.** The same design does not want the same numbers at 390px and 1440px, and a scale expressed in relative units gets some of that for free.

How to make the case

Arguing for space on aesthetic grounds loses to arguing for content on commercial grounds, every time. Two arguments that work better:

1. **The five-second test.** Show both versions to three people for five seconds each and ask what they saw. The dense version usually produces vaguer answers. That is a result, not an opinion.
2. **Name what the space is doing.** Not this looks better with more air, but this 48px gap is the only thing telling the reader that the pricing section has ended. If you remove it, the two sections merge.

The second is the more useful habit, because it forces you to check that the space really is doing something. Sometimes it is not, and then the senior person is right.

The failure at the other end

Disconnection. When gaps between related items grow past the point where proximity still binds them, the group dissolves and the reader is left with scattered orphans. A caption 64 pixels below its photograph belongs to nothing.

The check is the same one as ever: the gap inside a group must be visibly smaller than the gap around it. Generous space applied uniformly is not gen-

erous, it is uninformative. What you are buying with space is contrast between distances, and if every distance grows together you have paid and bought nothing.

BEFORE YOU MOVE ON

A designer responds to the complaint that a page feels cramped by increasing every gap in the layout by 50%. The page now scrolls much further and still feels cramped. What went wrong?

A. The increase was too small to clear the perceptual threshold, and gaps needed to double before the change would register as deliberate.

B. Scaling every gap equally preserves the ratios between them, so the grouping contrast that makes a layout feel open is unchanged.

C. Vertical space should never be increased on a scrolling page, and the extra room should have been taken from the horizontal gutters instead.

D. The page needed less content rather than more space, since no spacing change can rescue a layout carrying too many elements for its area.

---

16 · 9 min

You can feel a misalignment you cannot see

THE ONE THING TO KEEP

Every element should share an edge with something else, and where the alignment tool and your eye disagree, trust your eye.

Why "something is off" is a real perception

Human vision is unusually good at one specific task: judging whether two lines are aligned. This ability — vernier acuity — is finer than the resolution of the eye itself. People can reliably detect an offset several times smaller than

the width of a single light-sensing cell in the retina, an effect vision researchers call hyperacuity.

So when a client says the page feels off but cannot point at anything, they are not being vague. They are reporting a measurement your visual system took without telling you about it. Three items with left edges at 40, 40 and 43 pixels will produce that feeling in everyone who looks, and nobody will name it.

Good news, because misalignment is the cheapest fault to fix and needs no taste at all.

Alignment is about edges, and mostly the left one

Every element on a page should share an edge with something else. Not a vague sense of "roughly under" — an actual shared line you could draw.

Left edges do most of the work in a left-to-right script, because the eye's return sweep at the end of every line lands there. A consistent left edge gives that sweep a target. A ragged one makes the reader search each time, a few milliseconds at a time, which is exactly the kind of cost that gets reported as "it is tiring" rather than "line 3 starts three pixels late".

Right edges matter far less, which is why justified text is optional and often harmful: on a narrow column it stretches word spacing until pale rivers run down the paragraph. On a phone-width column, never justify.

Centring is a commitment, not a default

Centred text has no straight left edge at all. That is fine for three lines on a title card or an invitation, where the shape itself is the point. It is bad for a paragraph, and it is worse when it is inconsistent.

The single most common amateur tell in layout: a block where the heading is centred and the body underneath is left-aligned. Pick one axis for a group and hold it. If you centre, centre the whole group, including the little bits — the date, the caption, the price.

A grid is a short list of allowed positions

A grid is nothing more than a decision about where things may start, made once, so you stop making it repeatedly and inconsistently.

The minimum useful version: choose three or four horizontal positions and one gap value, and never invent another. On paper that is a fold: halve a sheet, halve it again, and you have four columns and every combination of them.

Twelve columns became conventional for a plain arithmetic reason: twelve divides evenly by two, three, four and six, so the same grid gives you halves, thirds, quarters and sixths without fractions. That is the entire justification.

Do the same for space. Pick a scale — 4, 8, 16, 24, 32, 48, 64 — and use only those values. You stop deliberating over whether a gap should be 18 or 21, and you get repetition for free.

Free tools: Figma's free tier and Penpot (open source, and it runs on your own machine) both have layout grids and spacing tokens; Inkscape has snapping and guides. A pencil and a folded sheet is not a downgrade — it is faster for deciding structure, and structure is what you are deciding.

Whitespace is a boundary, not a luxury

Margin is what tells a reader where the content ends and the world begins. The commonest crowding fault is an outer margin smaller than the biggest gap inside the layout — the content then appears to be sliding off the page, because the strongest boundary is in the wrong place.

But be honest about the limit here, because design writing is not. Whitespace is not a virtue in itself. A railway departures board, a price list, a newspaper front page and a currency-trading terminal are all dense on purpose, and adding air to any of them makes them worse. Density is legitimate when the reader came to compare many items at once. Whitespace is a tool for grouping and ranking, not a way to look expensive.

Where the maths is wrong and your eye is right

Alignment tools measure bounding boxes. Vision measures apparent mass. These disagree in predictable places, and knowing them is one of the more useful things in this course:

- **A circle must overshoot.** Put a circle and a square of the same height side by side and the circle looks smaller, because it has less area near its top and bottom. Full stops, bullet points and round logo marks are usually drawn two to five per cent larger than the square shapes they sit with.
- **A triangle centred in a circle looks left of centre.** The classic play button. The triangle's visual mass sits behind its bounding-box centre, so it needs a nudge right, roughly an eighth of its width. Every media player in the world does this.
- **Punctuation hangs outside.** Opening quotation marks and bullets set flush with the text edge make the edge look broken, because they are mostly air. Hang them slightly outside and the edge reads straight.
- **Text vertically centred in a button sits low**, because the box includes descender space that most words never use. Nudging up a pixel or two fixes it.

The general rule: when the number and the eye disagree about alignment, the eye wins. It is the eye that will be looking at it.

Today

Open a layout you have made and draw the alignment lines on top of it — a thin rectangle in any editor, or a marker on a printout. Count the vertical lines. If there are more than four, each one after the fourth needs a reason. Usually two of them are the same line, three pixels apart.

BEFORE YOU MOVE ON

A play button is drawn as a triangle centred inside a circle using the align tools, so it is mathematically dead centre. Everyone who sees it says it looks shifted left. What is going on?

A. The circle is not truly circular — the alignment tool is measuring a bounding box that includes the stroke width.

B. A triangle carries its visual mass behind its bounding-box centre, so mathematical centring reads as left-shifted; it needs nudging right by roughly an eighth of its width.

C. The triangle is too large for the circle, and reducing it will remove the impression of imbalance.

D. Screens round shapes to whole pixels, so a shape landing on a half-pixel appears displaced.

17 · 9 min

Where the alignment tool is wrong

THE ONE THING TO KEEP

Layout tools align bounding boxes while readers align visible mass, so icons beside text, letters against rules, punctuation at a margin and text in a button all need small documented corrections that the tool will otherwise undo.

The tool aligns boxes; the reader aligns shapes

Every layout tool aligns bounding boxes, because a bounding box is a rectangle it can compute. A reader aligns the visible mass of the marks. For plain rectangles these coincide, and for almost everything else they do not. The gap between them is where optical alignment lives, and it accounts for a surprising share of the difference between work that looks professional and work that looks nearly right.

Here are the specific cases, with the sizes of the corrections, so you know how much to nudge rather than guessing.

Overshoot in letterforms

A lowercase o is drawn slightly taller than a lowercase x. A capital O extends slightly above the cap line and below the baseline. A pointed A does the same. These overshoots are typically 1 to 1.5% of the size — about half a pixel at 40px, one and a half pixels at 100px — and they exist because a curve that stopped exactly at the line would look small.

You rarely have to do anything about this in text; the typeface designer has already done it. It matters when you set a single large letter or a numeral as a graphic element, or when you align a letterform against a drawn rectangle. Then you are working with the letter's outline, not its metrics, and you align what you see.

Hanging punctuation

A paragraph that begins with an opening quotation mark has a visual left edge further right than a paragraph that begins with a letter, because the quote mark is a small shape high up with white space beneath it. The traditional fix is to hang it outside the measure, so the letters line up and the punctuation sits in the margin.

The same applies to bulleted lists, where the bullet should hang and the text should align, and to hyphens at a right margin.

CSS has `hanging-punctuation: first;` and support is still limited to Safari at the time of writing, so in practice you do it with a negative indent:

```
blockquote p:first-child { text-indent: -0.4em; }
```

This is a good example of an honest limitation: the correct mechanism exists in the specification, is not widely implemented, and the workaround is approximate because the right offset depends on the typeface.

Icons beside text

An icon in a 24px box beside 16px text will look low when both boxes are centred. The reason is that text's optical centre is around the x-height, which sits below the centre of the line box, while the icon's mass usually fills its box evenly. The correction is small and consistent: shift the icon up by roughly 1 pixel at body size, 2 at larger sizes.

The second icon problem is optical size. A circle, a square and a triangle drawn to the same bounding box look like three different sizes — the circle smallest, the square largest. Icon sets built properly compensate by letting shapes exceed the nominal grid: in most well-made sets, circles are drawn a few per cent larger than squares. If you are mixing icons from two sources, this is the first thing that will look wrong and the hardest to name.

Text inside a button or a box

Set equal top and bottom padding on a button and the text will look as though it has sunk. The line box includes descender space that most words do not use, so the visible mass sits high in its own box and the space below it therefore looks larger.

The correction: reduce the bottom padding by 1 to 2 pixels at body size, or set the line height explicitly and pad against that. If your button contains a word with a descender — Category, Apply — the effect is smaller, which is why the same button can look right on one label and wrong on another.

The same is true of headings. A 24px gap below a heading is optically larger than a 24px gap above it, because the heading's line box carries half-leading above the caps. Headings almost always want more space above than below, and a symmetric setting reads as though the heading belongs to the paragraph before it.

Centring things with tails

A logo with a descender, a swash or a single tall ascender has its mass somewhere other than its bounding-box centre. So does any wordmark with an ampersand or a trailing full stop. Centre the mass. The practical method is to blur

the thing heavily until it is a single blob, note where the blob's centre falls, and align that.

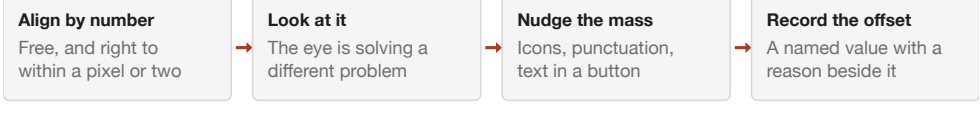
The workflow that survives other people

Align mathematically first, because it is free and gets you within a pixel or two. Then look. Then nudge. Then — and this is the part that gets skipped — record the nudge as a deliberate value rather than leaving it as a hand adjustment.

In code that means a named offset with a comment. In a design file it means a note on the component. Without it, the next person sees an element that is 1px off the grid, helpfully corrects it, and quietly undoes work they could not see the reason for. This happens constantly, in both directions, and the only cure is writing down why.

> *An optical correction with a reason attached is craft. The same correction with no reason attached is indistinguishable from a mistake, and will be treated as one.*

The workflow that survives the next person



An optical correction with a reason attached is craft. The same correction with nothing attached is indistinguishable from a mistake, and gets tidied away.

BEFORE YOU MOVE ON

A button with 12px padding on all four sides looks fine with the label Apply but looks top-heavy with the label Save. What explains the difference?

A. Save is a shorter word, so the button is narrower and its proportions make the vertical padding read differently.
B. The two labels use different letterform widths, so the optical centre of the word shifts and the padding needs adjusting per label.
C. Save has no descender, so its visible mass sits higher in a line box that still reserves descender space below it.
D. Capital S has more overshoot than capital A, which lifts the whole word slightly within its line and makes the space below look larger than it is.

18 · 9 min

A grid is arithmetic, not taste

THE ONE THING TO KEEP

A column grid is defined by the exact relationship between margin, column and gutter, and its value is that repeated positioning decisions get made once — it removes small errors and arguments rather than producing quality.

What a grid actually is

A grid is a set of vertical lines that elements snap to, so that repeated decisions get made once. That is the whole idea. It is not a source of beauty, it is not a guarantee of anything, and a poster designed by one person in an afternoon does not need one.

What a grid buys is coordination. When four people are laying out forty pages, or when a component will appear in six different contexts, a shared set of positions means nobody has to negotiate each placement. That is why grids came

out of newspaper and magazine production and why they matter more the more people are involved.

The arithmetic

Three quantities define a column grid:

- **Margin** — the space between the content area and the edge of the page or viewport.
- **Column** — the width of one unit.
- **Gutter** — the space between adjacent columns.

The relationship is exact:

```
content width = (n × column) + ((n − 1) × gutter)
```

Worked: a 1200px content area with 12 columns and 24px gutters gives

```
column = (1200 − 11 × 24) / 12 = (1200 − 264) / 12 = 78px
```

So an element spanning 4 columns is `4 × 78 + 3 × 24 = 384px`. Not 400, not a third of 1200. If you place things by eye at round numbers, you are not on the grid, and the misses will be a few pixels — exactly the size that vernier acuity detects and nobody can name.

In CSS you never do this arithmetic yourself:

```
.grid {
  display: grid;
  grid-template-columns: repeat(12, 1fr);
  gap: 24px;
}
.feature { grid-column: span 4; }
```

The browser computes the column width, including the fractional pixels, and it stays correct when the container resizes. In a drawing tool, set guides once and save them with the document. Inkscape does this under Edit > XML or by

dragging from the rulers; Scribus has a proper page-guide manager with a column and gutter dialogue built in.

Why twelve

Twelve divides by 2, 3, 4 and 6. That gives you halves, thirds, quarters and sixths from one grid, which covers nearly every layout anyone asks for. Sixteen is also common and gives halves, quarters and eighths but no thirds, which is why teams that adopt it end up hand-placing anything three-across.

On a phone, twelve columns is theatre. At 390px wide with a 16px margin each side, a twelve-column grid gives columns under 20px, narrower than a single character. Use four columns, or none at all — a single column with a 16px gutter on each side is a perfectly respectable grid and is what most phone layouts really are.

Margins are not decoration

The margin is the outer whitespace, and on screen it has a job beyond aesthetics: it keeps content away from the physical edge, where rounded corners, camera cut-outs, curved glass and the operating system's own gesture areas live. A 16px minimum side margin on a phone is the floor, and modern layouts add the safe-area inset on top of it:

```
padding-inline: max(16px, env(safe-area-inset-left));
```

In print the equivalent constraint is harder: anything within about 3mm of a trimmed edge may be cut off, because paper shifts in the guillotine. Covered properly in the block on physical output.

Beyond columns

Two extensions worth knowing:

- A **modular grid** adds horizontal divisions as well, producing cells rather than columns. Useful for catalogues, dashboards and anything with a repeating card.

- A **compound grid** overlays two column systems — say a 3 and a 4 — so that some elements align to thirds and others to quarters while both share the same outer margins. This gives variety without abandoning structure, and it is how most editorial layouts get their rhythm.

Both are tools for a specific problem. Reaching for them because a layout feels dull is usually a sign that the hierarchy, not the grid, is the issue.

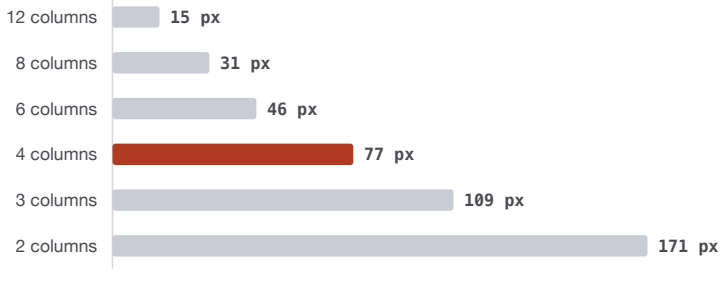
The failure modes

Grid worship. Elements forced to span whole columns when the content wanted a different width. A photograph cropped to fit a grid it did not need to fit. The grid serves the content.

Partial adoption. Half the page on the grid and half placed by hand is worse than neither, because the eye picks up the near-alignments and reports them as errors.

A grid with no margin discipline. Twelve tidy columns and elements that wander into the outer margin at random. The outer edge is the most visible alignment on the page.

Column width on a 390-pixel phone, 16px margins and gutters



The content width is 358 pixels. Twelve columns gives a column narrower than one character, which is why four columns, or one, is what a phone layout really is.

The honest summary: a grid removes a class of small errors and a class of arguments. It does not make anything good. Every dull layout on a twelve-column grid is evidence for that, and there are a great many of them.

BEFORE YOU MOVE ON

A designer building on a 1200px, 12-column grid with 24px gutters places a feature block at 400px wide, reasoning that it should span a third of the width. The page has faint misalignments throughout. Why?

A. A four-column span is 384px, because the span includes three gutters but not the two that would sit outside it.

B. A third of 1200px is 400px only if the margins are excluded, and the calculation should have started from the viewport rather than the content area.

C. Twelve columns cannot produce exact thirds at this width, so the block should have been placed on a compound grid with a three-column overlay.

D. Gutters should be subtracted from each column individually rather than from the total, giving a column width of 74px and a four-column span of 296px.

19 · 8 min

Vertical rhythm, and why the strict version fails on screen

THE ONE THING TO KEEP

A strict baseline grid is achievable in print and brittle on screen because of half-leading, arbitrary content heights, platform form controls and reflow, so keep the part readers perceive — a small consistent set of vertical values tied to the line height — and drop the rest.

The idea

A baseline grid is a set of evenly spaced horizontal lines — say every 24 pixels — that every line of text sits on. Body text, captions, headings and list items all land on the same lines, so that two columns side by side have their text in register and the page has an even vertical beat.

In print this is real craft and it is achievable. It is also old: in letterpress and in fine book printing, register-true setting meant the lines on the back of a leaf fell exactly behind the lines on the front, so that thin paper did not show a shadow of misaligned text. InDesign has a baseline grid; so does Scribus, which is free and open source, under File > Document Setup > Guides. LibreOffice Writer offers register-true paragraph styles.

Why it mostly does not survive the web

Four mechanisms break it, and it is worth knowing them precisely rather than concluding that web typography is simply careless.

1. The half-leading model. In CSS, when the line height exceeds the font's natural line box, the surplus is split equally above and below the text. Where the baseline falls inside that box depends on the font's internal metrics — ascent, descent and line gap — which differ between families and are not exposed to you in any convenient way. Change the typeface and every baseline moves by an amount you cannot easily predict.

2. Arbitrary content heights. An image is whatever height it is. A video embed, a code block, a map, a third-party widget — none of them is a multiple of 24 pixels, and a single one pushes everything after it off the grid for the rest of the page.

3. Form controls. Inputs, selects and buttons are rendered partly by the operating system and have platform-specific metrics. A select on Android is not the same height as one on iOS.

4. Reflow. The page is a different width on every device, so line counts change, and anything whose height depends on wrapping changes with it.

You can fight all four. People do, with `line-height` set in fixed multiples, `calc()` on every margin, and images forced into fixed-ratio boxes. The cost is real: brittle layouts, awkward image sizing, and a lot of arithmetic maintained by hand for a benefit almost no reader can articulate.

What to keep

The valuable part of vertical rhythm survives without the strict grid, and it is this: **vertical space should come from a small, consistent set of values that relate to the line height**.

Concretely, if your body line height is 24px:

- Use 24 and 48 as the main vertical gaps, with 8 and 16 for tight relationships.
- Give every text element a line height that is a clean multiple where you can — 24px body, 24px captions at a smaller size, 48px for a large heading.
- Set headings with more space above than below, because a heading belongs to what follows it.

This gets you the even beat that readers actually perceive — a page whose vertical spacing feels considered — without the brittleness. Nobody has ever noticed that a paragraph after a photograph is three pixels off a grid. Everybody notices a page where the gaps are 19, 26, 31 and 44.

Where the strict version is still worth it

- **Print, where the page is fixed.** A book, a report, a programme, a menu. Scribus will hold the grid and the result is visibly better in multi-column settings.
- **Multi-column text on a fixed-width screen**, such as a kiosk or a generated PDF.
- **Long documents with side notes**, where a note must align with the line it annotates.

Everywhere else, the honest answer is that strict baseline alignment is a cost with a small and mostly invisible return, and that the people who tell you otherwise are usually showing you a screenshot of a page with no images in it.

A note on line height inheritance

One practical trap while you are setting this up. A unitless line height — `line-height: 1.5` — is inherited as a ratio and recalculated against each element's own font size. A line height with units — `line-height: 24px` — is inherited as a fixed value, so a 32px heading inside that container gets 24px lines and the letters collide.

Set a unitless value on the body and override it explicitly on headings. This is the single most common cause of headings whose lines overlap, and it costs one character to fix.

BEFORE YOU MOVE ON

A site sets line-height: 24px on the body element to enforce a baseline grid. Headings at 32px then render with their lines overlapping. What is the mechanism?

A. The heading font has larger internal ascent and descent metrics than the body font, so its line box exceeds the declared height.

B. A line height with units is inherited as a fixed value, so 32px text is being given 24px of line box.

C. Half-leading splits the surplus equally above and below, and at 32px there is no surplus left to split, which collapses the spacing.

D. Baseline grids require the line height to be a multiple of the font size, and 24 is not a multiple of 32, so the grid cannot resolve and falls back to the font's default.

20 · 8 min

Padding belongs to the thing, margin belongs to the relationship

THE ONE THING TO KEEP

Padding is part of an object's identity and travels with it, while margin describes a relationship that belongs to the layout, so components should carry internal padding and let their container set the gaps between them.

A distinction that decides how reusable your work is

Every element has space inside it and space around it. The distinction sounds like a technical detail in CSS, and it is actually a design decision with consequences that show up months later.

The rule:

> ***Padding is part of the object.*** *It travels with the thing wherever it goes, because it is what makes the thing look like itself.*

> ***Margin is part of the relationship.*** *It belongs to the layout, because it describes how this object sits next to that one.*

A card has 24px of padding because a card with 4px of padding is not the same card. A card has 32px below it because *in this layout* it sits above a section heading. Move the card into a sidebar and the padding should follow; the 32px should not.

The component that cannot be reused

Here is the failure. Somebody builds a card and gives it `margin-bottom: 32px` so that a stack of them looks right. It does look right, in that one layout. Then:

- The card goes into a grid, where the gap is handled by the grid, and every card now has 32px of dead space underneath.
- The last card in a list has a trailing 32px that nothing balances, so somebody adds a rule for `:last-child`.
- A different page wants 48px, so somebody adds a modifier class.
- Six months later the card has four spacing overrides and nobody can say what its natural spacing is.

Every one of those steps was reasonable. The original decision was the mistake: the card claimed ownership of a relationship it was not part of.

Let the container own the gaps

The modern answer is that a container sets the spacing between its children, and children carry no outer margin at all:

```
.card { padding: var(--space-4); }       /* the object */
.card-list { display: grid; gap: var(--space-5); }  /* the re-
lationship */
```

This also sidesteps margin collapsing, which is a genuine source of confusion: adjacent vertical margins in normal document flow merge into the larger of the two rather than adding, and a parent's margin can collapse through to a child. `gap` does none of that. It puts the stated space between things and nothing else.

The same logic applies away from code. In a design file, a component's spec should state its internal padding. The spacing between instances belongs to the layout that contains them, and a component that ships with an external margin baked in is a component that will be fought with.

Padding is measured to the visible mass

The optical point from earlier returns here. If a button has 12px of padding measured against the line box, the space above the letters looks like about 8px

and the space below like about 16px, because the line box reserves descender room the word may not use.

So padding around text is not simply a number. Two working rules:

- Set the line height explicitly on the element you are padding, so you know what you are measuring against.
- Expect to shave 1 to 2 pixels off the bottom padding at body size, more at display sizes.

For icons the opposite problem appears: most icon artwork ships with transparent padding built into the file, so a 24px icon may only be 20px of visible mark. Two icons from different sets in the same 24px box will look like different sizes. Check the artwork, not the box.

The nesting rule

When a padded object sits inside another padded object, the two paddings add up, and the result is usually too much. A section with 32px of padding containing a card with 24px gives 56px between the card's content and the section's edge, which reads as a mistake.

The usual resolution is that one level of nesting owns the padding and the other owns nothing. Pick the level that is visible — if the card has a background, it needs the padding; if the section has the background, the card can be flush.

The related detail from the shape lesson: an inner rounded rectangle nested inside an outer one wants an inner radius equal to the outer radius minus the padding. With 16px of padding inside a 16px radius, the inner element should have square corners.

Why this is a design lesson and not an engineering one

Because the decision is about what a thing *is*. Deciding that a card owns its padding is deciding that the padding is part of the card's identity, and that its position in a layout is somebody else's business. That is a piece of design thinking, and it is the point where a set of screens starts to become a system.

BEFORE YOU MOVE ON

A card component is built with 24px padding and 32px bottom margin. It looks right in the original stacked list. What is the first thing that breaks when the card is reused in a grid?

A. The grid gap and the card's margin add together, leaving uneven vertical space and a trailing gap after the final row.
B. The card's 24px padding collides with the grid's own padding, producing 48px of combined inset that looks like an error.
C. Margin collapsing merges the card's bottom margin with the grid container's margin, so the spacing disappears entirely.
D. The card loses its rounded corners against the grid's background, because an inner radius must equal the outer radius minus the padding for the curves to run parallel.

21 · 9 min

How big a thing has to be to hit

THE ONE THING TO KEEP

Visual size and target size are separate decisions — aim for 44 to 48 pixels of target with at least 8 pixels between neighbours — and because distance and width both sit inside a logarithm, widening a target buys as much as halving the distance to it.

Visual size and target size are two different numbers

A close button may be a 16-pixel cross. Its target — the region that responds to a tap — should be about 44 pixels square. These are separate decisions, and treating them as one is the commonest cause of an interface that feels fiddly on a phone.

The numbers, from the platform guidance and the accessibility standard:

- **iOS**: 44 by 44 points minimum.
- **Android / Material**: 48 by 48 density-independent pixels.
- **WCAG 2.2**, success criterion 2.5.8 Target Size (Minimum), level AA: 24 by 24 CSS pixels, with exceptions where spacing compensates, where the target is inline in a sentence, or where the presentation is essential.
- **WCAG 2.1**, criterion 2.5.5 Target Size, level AAA: 44 by 44 CSS pixels.

The AA floor of 24 is lower than either platform's guidance, which surprises people. It is a floor for conformance, not a recommendation. Design to 44 or 48 and you satisfy everything.

A fingertip's contact patch is roughly 8 to 10 millimetres, and the point a person believes they are touching is typically a few millimetres above the centre of the contact patch. That is where these numbers come from: 44 points is about 9mm on a typical phone.

Fitts's law, and what it actually says

The time to reach a target is proportional to the logarithm of the distance divided by the target's width:

```
time ≈ a + b · log2( 2D / W )
```

The useful consequence is the shape of that relationship, not the constants. Because both distance and width sit inside a logarithm, **doubling the width buys the same improvement as halving the distance**. Width is usually far cheaper to change than distance, which is why enlarging a target is the first move and rearranging a layout is the second.

A second consequence, on desktop only: the pointer stops at the edge of the screen, so a target against the edge is effectively infinitely wide in that direction, and a corner is infinite in two. This is why the macOS menu bar at the very top and the Windows Start button in the corner are fast to hit, and why a toolbar placed one pixel away from the screen edge throws that advantage away. On touch, the effect does not apply — there is no pointer to stop — and

edges are actively worse because of the operating system's own swipe gestures.

Spacing matters as much as size

Two 48px targets with no gap between them are not safe. The error case is not missing the target, it is hitting the wrong one, and the consequence of hitting Delete instead of Archive is much worse than the consequence of missing both.

A working minimum is 8 pixels of clear space between adjacent targets, and more where the actions differ in consequence. Put destructive actions somewhere other than beside their most common neighbour, and give them a confirmation if the gap cannot be made.

WCAG 2.2's spacing exception formalises this: a target smaller than 24px can still conform if a 24px circle centred on it does not overlap the circle of any adjacent target.

Implementing a large target on a small mark

You do not have to make the icon bigger. Three free approaches:

```
/* 1. Padding on the interactive element */
.icon-button { padding: 14px; }   /* 16px icon + 28px = 44px */

/* 2. A pseudo-element that extends the hit area */
.small-link { position: relative; }
.small-link::after {
  content: ''; position: absolute; inset: -12px;
}

/* 3. Minimum size, letting the icon centre itself */
.icon-button { min-width: 44px; min-height: 44px;
  display: inline-grid; place-items: center; }
```

The first is usually the right one, because padding is part of the object, which is the rule from the previous lesson.

Testing it honestly

A mouse on a desktop will hit anything. The bug is invisible until somebody uses a thumb while walking. Two checks that cost nothing:

- **Open the page on an actual phone and use it one-handed**, while standing up, not while sitting at your desk holding it with both hands.
- **In browser developer tools**, device emulation shows layout but not accuracy. It will not find this class of fault, and believing it does is how the fault ships.

The counter-argument

Larger is not automatically better. Fitts's law describes the cost of hitting a target, and it applies just as well to targets you do not want hit. A giant delete button is a hazard, and an interface where every control is 60 pixels tall pushes real content off the screen and forces scrolling that costs more than the taps it saved.

The rule that resolves it: size targets in proportion to how often they are used and how recoverable they are. Frequent and safe gets large and close. Rare and destructive gets small, far away, and confirmed.

BEFORE YOU MOVE ON

A toolbar of 20px icons packed 2px apart passes a desktop review but generates complaints about wrong taps on phones. The team doubles each icon's visual size to 40px, keeping the 2px gaps. Complaints about wrong taps continue. Why?

A. 40px is still below the 44px platform minimum, so the targets remain too small for a fingertip's contact patch.

B. Fitts's law says distance matters more than width, so the icons needed to be moved closer to the thumb rather than enlarged.

C. The error is hitting the neighbour rather than missing the target, and 2px of separation leaves no margin for an imprecise touch.

D. Enlarging the icons without enlarging their padding leaves the interactive region unchanged, so the tap areas are the same as before.

22 · 8 min

Balance is a comparison of weights

THE ONE THING TO KEEP

Balance compares visual weights multiplied by their distance from an axis rather than matching sizes, and a composition that is slightly off balance reads as an error, so either resolve it properly or push the imbalance far enough to be obviously deliberate.

Two ways to balance a page

Symmetry balances by mirroring. Put the same thing at the same distance on both sides of an axis and the result is stable, formal and slightly static. This is the right answer more often than modern design writing admits: certificates, invitations, title pages, official notices, anything where the message is this is serious and this is settled.

Asymmetric balance balances by weight. A large pale element near the axis can balance a small dark element far from it, in the same way a light weight on a long lever arm balances a heavy one on a short arm. The result is more dynamic and much harder to get right, because you are judging a product of two quantities rather than matching one.

The lever analogy is close enough to be useful. Weight, from the first block, is roughly area times darkness times detail. Distance from the axis multiplies it. So:

```
left side:  a 240px pale photograph, mean lightness 85%, 120px
from centre
right side: a 70px block of dark bold text, mean lightness 25%,
320px from centre
```

These can balance, and no measurement will tell you exactly. But the model tells you which direction to move when it is wrong, which is the part you actually need.

The near-miss problem

The worst outcome is neither balanced nor deliberately unbalanced. A layout that is 5% off balance reads as a mistake, in exactly the way a 17px heading above 16px body text reads as a mistake. The threshold rule applies to composition as well as to type.

So commit. Either resolve the balance properly, or push the imbalance far enough that it is obviously a choice — an element hard against one edge with a large void opposite it, not an element drifting slightly right of centre.

This is the most common fault in first posters and first slides: everything is nearly centred, nearly balanced, nearly aligned. The fix is rarely subtle adjustment. It is usually to pick a direction and go much further.

Tension is balance you have spent on purpose

An unbalanced composition creates tension, which is useful when the subject wants it: urgency, movement, instability, a before-and-after where the before

is meant to feel wrong. A figure placed hard against the right edge, looking off the page, with empty space behind, is unsettling, and that is a tool.

Two things keep tension from reading as an accident:

1. **Magnitude.** Far past the threshold, as above.
2. **Consistency.** If the composition is deliberately weighted left, everything else in the piece should agree — the type alignment, the crop, the direction of any implied motion. A layout weighted left with centred type is a layout in an argument with itself.

Checking it

Three free checks, in increasing order of reliability.

Flip it horizontally. Mirror the whole design. A composition you thought was balanced often collapses when mirrored, which tells you it was relying on the reading direction rather than on weight. This also catches the near-miss, because the near-miss moves to the other side and you notice the movement.

Blur it heavily and look at the blobs. From the first block. Balance is a property of the masses, and once the detail is gone you can see the masses.

Turn it upside down. The oldest trick in drawing instruction. Inverting the image suppresses your recognition of what the objects are and leaves you looking at shapes and weights. It is startlingly effective, and it takes one keystroke in any editor.

Balance is not centring

A final distinction, because the two get confused constantly. Centring is placing an object on an axis. Balance is a relationship between everything on both sides of that axis.

A centred headline over an off-centre photograph is centred and unbalanced. A left-aligned headline with a heavy image on the right can be perfectly balanced and contains nothing centred at all. Layouts that centre everything by default are not balanced; they have simply declined to make the decision, and the tell is that they usually feel inert without anybody being able to say why.

> *Symmetry is a decision. Centring everything is the absence of one, and readers can feel the difference even though they will describe it as the design being boring.*

BEFORE YOU MOVE ON

A poster has a large pale illustration slightly left of centre and a small dark headline slightly right of centre. Reviewers say it feels wrong but cannot say why. What is the most likely diagnosis?

A. The illustration is too pale to balance any text, so it should be darkened until its visual weight matches the headline's.

B. Both elements are near the axis, so neither the symmetry nor the imbalance is decisive and the result reads as a near-miss.

C. Pale illustrations always need to sit on the right in left-to-right layouts, because the eye finishes its sweep there and expects the lighter element last.

D. The headline is too small relative to the illustration, and enlarging it until the two occupy similar areas would resolve the composition.

CASE STUDY · MODULE 2

The admission form that cost two hundred hours to correct

A polytechnic in Coimbatore admitting about 1,400 students a year, with a printed application form and an identical web version.

The office had a recurring complaint that everybody treated as a fact of life: a large share of application forms arrived filled in wrongly. Phone numbers in the email row. The guardian's name in the applicant's name field. The community certificate number in the box for the transfer certificate number. Two clerks spent roughly the six weeks of the admission season telephoning applicants to correct entries, and the office had come to describe this as "the students are careless".

A lecturer who taught a first-year design elective asked for a blank form and measured it.

What the measurements said

The form had been laid out in a word processor, and every gap in it was a gap somebody had made by pressing Return. Measured, the vertical distances were 14, 16, 15, 18, 16, 22 and 14 points, in no pattern. The most important measurement was the one that repeated: a label sat about 14 points above its own box, and the next label sat about 16 points below that box.

To a reader, those two distances are the same. The label therefore belonged to both fields, or to neither, and which one a person chose depended on where their eye happened to land. The words on the form were correct. The spacing said something different, and the spacing is read first.

There were two further faults. Four groups of fields — personal, academic, community, contact — were separated by exactly the same space that separated the fields within them, so the form read as one undifferentiated list of twenty-three boxes. And the outer margin of the sheet was 8mm while the largest internal gap was 22 points, so the content looked as though it was sliding off the page.

The decision

The repair was not difficult to describe. Label 4 points above its own box, 24 points to the next field, 40 points between groups, a 16mm outer margin, and every one of those values drawn from a single scale of 4, 8, 16, 24, 40.

The problem was that the form had to remain one sheet of A4, printed both sides. It already was, at the current spacing, with the back side used for the declaration and the checklist of enclosures. Adding the space the grouping needed pushed the form to three sides, which means two sheets.

The numbers were available. The polytechnic printed about 6,000 forms a season for walk-in applicants and district outreach camps. A second sheet added roughly nine rupees per applicant in paper, printing and the heavier envelope, which came to somewhere near fifty-four thousand rupees a year, in a budget where that was a line somebody would have to defend.

Against that sat an estimate the office had never made. The two clerks reckoned they corrected about four hundred forms a season, at fifteen to twenty minutes each once the telephoning, the re-checking and the re-entry were counted. That is between one hundred and one hundred and thirty hours, twice over, in the busiest six weeks of the year, and it did not include the applicants whose forms were rejected outright and who did not call back.

The registrar's objection was not that the arithmetic was wrong. It was that the printing cost was a visible line item he would be asked about, and the clerks' hours were already paid for and therefore invisible. Two real costs, one of them countable in a budget and the other countable only if somebody chose to count it.

STOP HERE

Before turning the page: what would you have done, and what would it have cost if you were wrong? Write it down. The point of the next section is to compare.

WHAT HAPPENED

The head of department authorised a single season as a trial rather than a permanent change, which made the decision reversible and got it approved. The form went to two sheets with the spacing scale applied, the four groups separated by 40 points, and the guardian's details moved under a bounded heading of their own rather than sitting adjacent to the applicant's.

The clerks were asked to keep a tally rather than an impression. Corrections fell from a counted 411 the previous season to 96. The three commonest errors — phone in the email row, guardian's name in the applicant's row, and the two certificate numbers swapped — accounted for most of the drop.

The web version took an afternoon and cost nothing, because there was no paper constraint at all; the same scale was applied with a container setting the gaps between fields. The office kept the two-sheet printed form the following year without discussion, and the registrar's later summary of it was that the paper had been the cheaper thing all along and nobody had ever put the two numbers next to each other.

Why could this form not have been repaired by making the labels bold, or by colouring the required fields?

The fault was a proximity error, and proximity is a stronger grouping cue than similarity. Bolding or colouring a label adds a similarity cue while leaving the distances that attach it to the wrong field unchanged, so the two cues then disagree and the reader gets two competing structures instead of one wrong one. Distance errors have to be fixed with distance.

The registrar's objection was about which cost was visible. How would you frame the argument without relying on his goodwill?

Convert the invisible cost into the same unit as the visible one and present both. The clerks' correction work was countable once somebody counted it: about four hundred forms at fifteen to twenty minutes, twice over, in the season's busiest weeks. Proposing a single reversible season, with a tally kept, also removes the need to win the argument in advance, because it turns a claim into a measurement.

The outer margin was 8mm while the largest internal gap was 22 points. Why does that combination matter on its own?

The strongest boundary on a page should be the one between the content and the edge of the sheet. When an internal gap is larger than the outer margin, the strong-

est boundary sits inside the layout, so the content reads as sliding off the page and the internal gap reads as a divider splitting the form. Fixing the margin costs nothing and repairs a fault most readers feel without being able to name.

MODULE 2 · TEST

Check what stuck

6 questions, one right answer each. This one is for you, not for a record: answers and the reason behind each are at the back. If two questions from the same module go wrong, re-read that module before the exam rather than after.

1 A form's labels sit closer to the field below them than to their own field. A reviewer suggests colouring each label to match its field instead of moving anything. Why will that not work?

A. The labels would then need a border as well, which adds elements to the page

B. Colour is invisible to one man in twelve, so it can never carry a grouping

C. Proximity is a stronger cue than similarity, so the distances keep winning

D. Similarity groups only items that are identical in every other respect

2 Why is a spacing ramp of 4, 8, 12, 16, 24, 32, 48, 64 better than one of 4, 8, 12, 16, 20, 24, 28, 32?

A. Because the second contains more values than any one design will use

B. Because the second is not a modular scale, so its values cannot be tokenised

C. Because every step in the first is divisible by eight, which suits real device pixel ratios

D. Because the first keeps the proportional gap roughly constant as the numbers grow

3 A card component ships with a bottom margin of 32 pixels so that a stack of cards looks right. What is the underlying mistake?

A. The card has claimed a relationship that belongs to the layout
B. Margin cannot be expressed as a token, so the value drifts over time
C. Padding and margin render identically on screen, so the choice is arbitrary
D. Adjacent margins collapse, so the visible gap is always half of what was set

4 Fitts's law puts both the distance to a target and the target's width inside a logarithm. What follows for design?

A. The time to reach a target grows in proportion to the distance travelled
B. Doubling a target's width buys about as much as halving the distance to it
C. Any target further than a thumb's length away must be at least twice as wide
D. Moving a target closer always helps more than making it larger

5 Why does a strict baseline grid survive in print and break down on the web?

A. Screens cannot render the sub-pixel positions that a baseline grid needs
B. Browsers round every line height to a whole pixel, and the error accumulates down a page
C. Images, embeds, form controls and reflow are not multiples of the grid
D. CSS has no property capable of setting a baseline grid at all

6 A layout has a 12-pixel outer margin and a 32-pixel gap between two internal groups. What does a reader experience?

A. Nothing, provided every gap comes from the same spacing scale
B. The internal gap reads as generous, which makes the layout feel expensive
C. The page feels cramped, which is repaired by reducing the internal gaps
D. The content looks as though it is sliding off the page

MODULE 3

Type, mechanically

Typography is the part of design with the most measurable answers and the most folklore attached. This block replaces the folklore: how to compare two typefaces at the same apparent size, how to build a size scale, when to justify and why rivers form, what tracking is for, which figures to use in a table, and what breaks when the text is in Devanagari or Urdu rather than English.

> BY THE END OF THIS MODULE YOU CAN
>
> Set a page of text a stranger can read comfortably — measure, leading, size scale, alignment, figures and emphasis each chosen from stated numbers — say why each number is where it is, and name what changes when the same page is set in a non-Latin script

---

23 · 10 min

Your font is fine, your line is too long

> THE ONE THING TO KEEP
>
> Set the line length to 45-75 characters before you touch the typeface, because most "hard to read" is a measure problem wearing a font costume.

The problem people misdiagnose

A page of text feels tiring. The instinct is that the font is wrong, or too small, so the typeface gets swapped and the size goes from 16 to 18. It does not help, and now the page holds less.

The usual culprit is not the typeface. It is the **measure** — the length of the line, counted in characters. On a wide screen with no width limit, a paragraph can run 140 characters across. Reading it is genuinely harder work.

At the end of a line your eye performs a return sweep: a long jump back and to the left onto the start of the next line. That jump is ballistic and slightly imprecise, so the longer the line, the more often it lands on the line above or below the one it wanted. You re-read a line, or skip one and lose the thread. You do not notice this happening. You notice being tired.

The numbers that matter

- **45 to 75 characters** per line for continuous text, spaces included. Around 66 is the traditional target for a printed book.
- **35 to 50** on a phone.
- Fix it with a maximum width, not a smaller font — roughly 30 to 34em in CSS terms.

To measure: count the characters in one full line. Ten seconds, and the best-spent ten seconds in this course.

Line spacing follows from measure. Body text on screen wants roughly 1.4 to 1.6 times the font size. Longer lines need more, because the return sweep needs a bigger target. Headlines want less — around 1.0 to 1.2 — because at large sizes the default gap opens into a visible hole.

Point size measures a box, not the letters

This is what makes font sizes untrustworthy. A type size measures the invisible body the letters sit on, inherited from the metal block in a printing press. It does not measure the letters.

What you actually perceive as size is the **x-height**: the height of a lowercase x, and therefore of most of the letters in any word. Typefaces vary enormously here. Set Helvetica and EB Garamond both at 16 pixels and the Garamond looks noticeably smaller, because it has a small x-height and long ascenders. Neither is wrong; the number is just not measuring what you think it is.

Two consequences:

- When you swap a typeface, **re-set the size by eye** against the old one. Do not assume 16 is 16.
- Large x-heights hold up better at small sizes and on poor screens. Small x-heights read as more formal and need a size bump and more leading.

What a typeface actually communicates

Less than designers claim and more than clients think.

What is reasonably reliable is **association through exposure**. A typeface carries whatever the reader has seen it doing. Comic Sans reads as a school noticeboard because that is where people met it. Times New Roman reads as an unstyled document, because for twenty years it appeared when nobody chose anything. Impact reads as a meme caption. None of this is in the letter-forms; it is in the reader's history.

There is also **era**. Heavy slab serifs come out of nineteenth-century wood-type advertising and still read as loud and physical; geometric sans-serifs built from circles and straight lines come out of 1920s European modernism and still read as rational and a bit cold.

What is not reliable is the claim that a typeface is friendly, trustworthy or premium. The research behind those assertions is thin and does not travel across cultures or scripts. The safest formulation: **a typeface can be clearly wrong more reliably than it can be clearly right.** A legal notice in a rounded comic face is wrong, and everyone agrees. Which of two competent sans-serifs is more trustworthy is a conversation with no answer.

Why two or three typefaces is the working limit

The limit is usually stated as a taste rule. It is not; it is an information rule.

A change of typeface signals that the kind of thing you are reading has changed — heading, not body; quotation, not commentary. With two typefaces you have one such signal, which is generally enough. With five, a change of typeface no longer means anything, because it happens constantly. You have spent the signal and received nothing.

You do not need extra families to get contrast. Within one family you have regular, semibold, bold and italic — four clearly distinct voices from one file, guaranteed to fit together because they were drawn together.

For a serif and a sans that pair without guesswork, use a superfamily: IBM Plex Sans and Serif, Source Sans and Source Serif, Noto Sans and Noto Serif. All free.

Free, and legally free

Google Fonts, Fontshare and Open Foundry give away professional typefaces under the SIL Open Font License, which permits commercial use. A genuine free path, not a compromised one.

Two warnings.

Licence. Sites advertising "1000 free fonts" often host commercial fonts someone simply uploaded. Using one in paid client work is a real exposure for you and your client. Look for the actual licence; SIL Open Font License is the safe one to see.

Script coverage. This matters more than most courses admit. Many popular Latin typefaces contain no Devanagari, Bengali, Tamil, Telugu, Thai or Arabic at all. If you set a Hindi headline in one, the text silently falls back to whatever the device has, and your careful design breaks on the phones of the people you made it for. Check the language coverage before you commit — Google Fonts lets you filter by script, and the Noto family exists to cover the ones others skip.

Today

Take one paragraph you have written and count the characters in a full line. If it is over 75, set a maximum width until it lands between 60 and 70, set the line spacing to 1.5, and read it again without changing anything else.

BEFORE YOU MOVE ON

Body text on a wide desktop page is set in a 16px serif at 1.5 line spacing, and readers report it is tiring. Bumping the size to 18px changed nothing. What is the most likely cause?

A. Serif faces are harder to read on screen than sans-serifs, so the family should be swapped.

B. 1.5 line spacing is too tight at that size, and 2.0 would give the eye more room to travel.

C. The typeface has a small x-height, so 16px is effectively smaller than it looks and 18px was not a big enough jump.

D. The line is running well past 75 characters, so the return sweep at the end of each line frequently lands on the wrong line; the fix is a maximum width, not a larger size.

24 · 9 min

The four measurements that decide how a typeface behaves

THE ONE THING TO KEEP

A typeface's behaviour is governed by measurable properties — x-height ratio, aperture, stroke contrast, descender length and character differentiation — and matching point size between two faces does not match their apparent size or their legibility.

Why two fonts at 16px are different sizes

Set the same sentence in Verdana and in EB Garamond, both at 16 pixels. The Verdana looks substantially larger. Nothing is wrong with either; the point size measures the invisible body the letters sit on, and the letters occupy different proportions of that body.

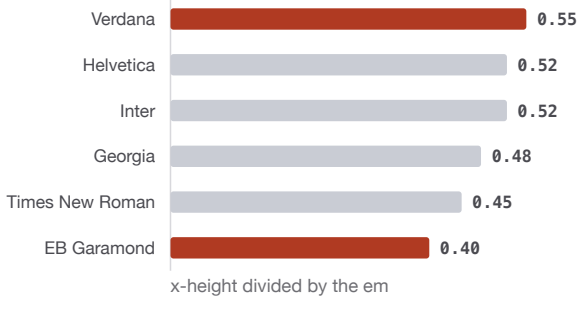
The number that governs this is the **x-height ratio** — the height of a lowercase x divided by the em. Approximate values:

```
Verdana         0.55
Helvetica       0.52
Inter           0.52
Georgia         0.48
Times New Roman 0.45
EB Garamond     0.40
```

So Garamond at 16px has an x-height of about 6.4 pixels and Verdana about 8.8 — a 38% difference in the part you actually read. This is why a design that looks fine in one face falls apart when somebody substitutes another at the same declared size.

The arithmetic to correct it: to match a 0.52 face at 16px using a 0.40 face, you want `16 × 0.52 / 0.40 = 20.8px`, so set it at 21. CSS can do this for you with `font-size-adjust: 0.52`, which tells the browser to scale whatever face it ends up using so its x-height matches that ratio. Support arrived late and is now broad; it is particularly useful for fallback fonts, which is the case where the substitution is out of your hands.

x-height as a fraction of the em, at one point size



Set at 16 pixels, Garamond gives an x-height of about 6.4 pixels and Verdana about 8.8. The declared size is the same and the part you read is 38 per cent apart.

Aperture, and why some faces fail at small sizes

Aperture is how open the gap is at the end of a stroke — the mouth of a c, the opening of an s or an a. Closed apertures, where the terminals curl back to-

wards the body, look tidy at display sizes and turn c into o at 12 pixels. Humanist faces with open apertures stay legible smaller and on poor screens.

This is a real, checkable property rather than a matter of preference. Print a word containing c, e, s and a at 11px, look at it from a metre away, and see which letters have collapsed.

Stroke contrast

Contrast here means the difference between the thick and thin strokes of a letter. A Didone face such as Playfair Display has extreme contrast: hairlines that are lovely at 60 pixels and vanish at 14, especially on a low-density screen or on newsprint where ink spreads.

The rule that follows: **high-contrast faces are display faces**. Low-contrast faces survive small sizes, bad printing and bad screens. If you are choosing one face for a whole project and some of it will be small, choose the low-contrast one.

Descender and ascender length

Long descenders are beautiful and they eat line height. A face with long descenders set at 1.4 line height will have its p and g very close to the capitals of the line below. Faces drawn for screens — Inter, Source Sans, Public Sans — have short descenders and large x-heights precisely because they were designed to work at tight line spacing in small sizes.

The practical consequence: line height is not a property of the size, it is a property of the size and the face together. When you swap a typeface, re-check the leading.

Letters that must not be confused

One more property, and it matters more than any aesthetic question in certain jobs: how well the face distinguishes **I, l and 1**, and **O and 0**.

In Helvetica, capital I and lowercase l are the same vertical bar. In a reference number, a password, a tracking code, a part number or a one-time code, that

is a real failure with a real cost. Faces that solve it do so deliberately — a tail on the l, a slash or a dot in the zero, serifs on the capital I:

- **Inter** offers an alternate l with a tail, and a slashed zero, both as OpenType features.
- **IBM Plex Mono**, **JetBrains Mono** and **Source Code Pro** distinguish all of them by default, and all three are free and openly licensed.
- **Atkinson Hyperlegible**, released free by the Braille Institute, was designed specifically to maximise the difference between similar characters for low-vision readers.

If your interface shows codes, use one of these for the code even if the rest of the page is set in something else. This is one of the few places where mixing faces is not a style decision but a correctness one.

How to compare two faces honestly

Not by looking at the name, and not by setting the alphabet. Set the same real paragraph of your own content in both, adjusted to matching x-height rather than matching point size, at the size and line length you will actually use. Then print it, or look at it on a phone.

Most typeface comparisons are made at 60 pixels on a large monitor, which is a test of how the face looks and not of how it works. The two questions have different answers surprisingly often.

BEFORE YOU MOVE ON

A site swaps its body font from Inter to EB Garamond at the same 16px and the same 1.5 line height. Readers report the text is harder to read. Which explanation accounts for the most of it?

A. Serif faces are harder to read on screen than sans-serif faces, so the substitution was always going to reduce legibility regardless of the settings.

B. Garamond's longer descenders need more line height than 1.5, so the lines are colliding and the block reads as a solid dark mass rather than as a series of separate lines.

C. EB Garamond has higher stroke contrast, so its hairlines are disappearing at 16px on ordinary screens.

D. Garamond's x-height is around 0.40 of the em against Inter's 0.52, so the letters are roughly a quarter smaller although the declared size is identical.

25 · 8 min

Four sizes, chosen once

THE ONE THING TO KEEP

Build four or five text sizes from a base and a ratio so that every pair clears the perceptual threshold, compress the ratio on small screens, and carry the rest of the hierarchy with weight, colour and space rather than with more sizes.

The list you should be able to write down

A finished design should have four or five text sizes and you should be able to name all of them from memory. If you cannot, the page has sizes in it that nobody decided.

A typical honest set:

```
12px  captions, legal, metadata
16px  body
20px  lead paragraph, card titles
28px  section headings
40px  page title
```

Five values. Every piece of text on the page is one of them. That constraint is the whole technique, and it does more for coherence than any choice of typeface.

Building the scale

Pick a base — almost always 16px, because it is the browser default and the size most readers have implicitly agreed to — and a ratio, then multiply.

```
1.2   (minor third)   16 → 19 → 23 → 28 → 33 → 40
1.25  (major third)   16 → 20 → 25 → 31 → 39 → 49
1.333 (perfect fourth) 16 → 21 → 28 → 38 → 51 → 67
1.5   (perfect fifth)  16 → 24 → 36 → 54 → 81 → 122
```

Round to whole numbers. The musical names are borrowed and decorative; there is no evidence that a page set on a perfect fourth is more harmonious than one set on 1.3. What the ratio genuinely buys you is that **every size is guaranteed to clear the perceptual threshold against every other size** — which, from the first block, is the thing that makes a difference read as a rank rather than as an error.

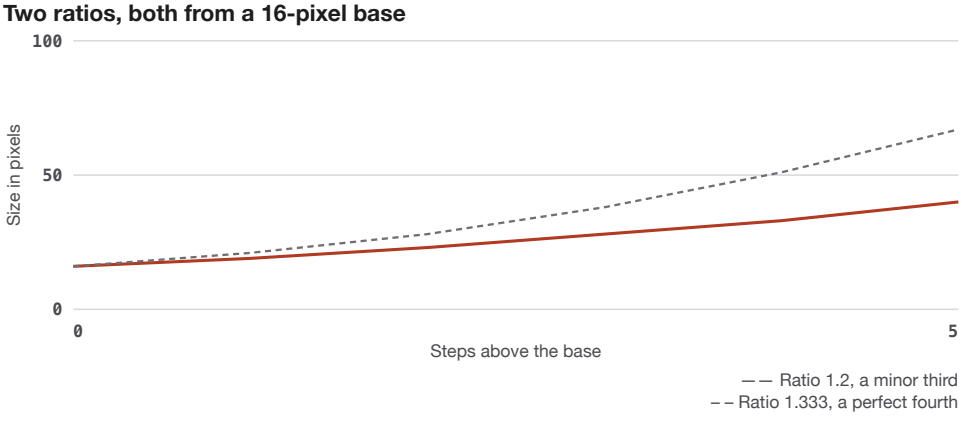
Go down as well as up. From 16 at 1.25, the step below is 12.8, which rounds to 13. Many designers set captions at 14, which is not on the scale and is close enough to 16 to look accidental.

The scale has to change size with the screen

A 1.333 scale looks excellent at 1440 pixels and ridiculous at 390. A 51px page title on a phone occupies 13% of the screen width per character and forces a two-word line. The correct response is not a different design but a compressed

ratio on small screens: use around 1.2 on a phone and let it open up as the viewport grows.

CSS does this in one declaration per size:



Both clear the perceptual threshold at every step. The larger ratio opens up fast, which is why a 1.333 scale is compressed towards 1.2 on a phone.

```
h1 { font-size: clamp(1.75rem, 1.2rem + 2.5vw, 3rem); }
```

That reads: never below 28px, never above 48px, and in between it grows with the viewport. Including a `rem` term in the middle value matters — a value expressed purely in `vw` ignores the reader's own font-size setting, which is an accessibility failure that passes every visual review.

Why fewer sizes is better

Three reasons, in order of how much they matter:

1. **Each size means something.** With five sizes, a reader learns within one screen that 28px means new section. With eleven, 28 and 26 both appear and neither means anything.
2. **Differences clear the threshold.** Sizes one or two pixels apart read as production errors.
3. **It is cheaper to change.** Five values in one place, adjusted once, versus forty declarations across the project.

The audit is the same as for spacing: list every distinct size on a page and count. More than six on a single screen is a finding.

Size is not the only rank

The commonest misuse of a type scale is trying to express every level of hierarchy through size alone, which ends with six heading sizes and a page that looks like a ransom note.

Weight, colour and space are all available. A subheading can be the same size as the body text, set in semibold, with 32px above it and 8px below. That is three channels doing the work of one size step, and it reads more clearly than a 17px heading ever will.

In long-form text this is the normal solution: two or three sizes for the whole document, with weight and space carrying the structure beneath the top level.

The honest limitation

A modular scale is a convenience, not a law. Real typographic settings break it constantly: a price that has to be exactly as wide as a card, a logotype that was drawn at a fixed size, a legal notice a regulator specifies in points.

The workable position is that the scale is the default and departures are deliberate and documented, which is the same rule as for spacing. What you are avoiding is not the existence of an off-scale value; it is a page with nine of them and no memory of why.

BEFORE YOU MOVE ON

A designer sets h1 { font-size: 5vw } so the title scales with the viewport. It looks correct on every device tested. What has been broken?

A. Viewport units are relative to the layout viewport rather than the content column, so the title will not stay proportionate inside a constrained container.
B. The title has no minimum, so on a very narrow screen it will render too small to function as a title at all.
C. A size expressed purely in viewport units ignores the reader's own font-size setting, so somebody who has increased it gets no change.
D. 5vw does not correspond to any step on a modular scale, so the title will not clear the perceptual threshold against the sizes around it at most viewport widths.

26 · 9 min

Choosing a typeface, and what the licence actually says

THE ONE THING TO KEEP

Choose a typeface by the job it must do — available weights, real italics, script coverage, figure behaviour — before aesthetics, and check the licence, because a desktop licence usually does not cover web embedding and design protection itself varies by country.

Four questions, before any aesthetic judgement

Most typeface selection goes wrong before taste enters, because a face was chosen for how a heading looked and then asked to do a job it was not drawn for. Four questions filter out most mistakes:

1. **How many weights and styles does it have?** A family with Regular and Bold gives you one weight step. A family with 300 through 800 plus

italics gives you a hierarchy without a second typeface. This single question eliminates more candidates than any other.

2. **Does it have real italics?** A true italic is a separate design with a cursive skeleton, not a slanted roman. If the family has no italic file, browsers and word processors will synthesise one by shearing the letters, and it looks like exactly what it is.
3. **What scripts and characters does it cover?** If any of your content will be in Hindi, Tamil, Urdu or Bengali, or contains currency symbols, mathematical signs or accented names, check before committing. Missing glyphs fall back to another font mid-word.
4. **What are the figures like?** Does it have tabular numerals for tables? Does it distinguish 0 from O?

Only then look at whether you like it.

Text faces and display faces

A display face is drawn for large sizes: fine detail, tight default spacing, dramatic contrast, sometimes very short descenders. At 15 pixels it falls apart. A text face is drawn for small sizes: sturdier strokes, open apertures, generous spacing, and it looks bland and slightly wide when you blow it up.

This is not a hierarchy of quality. It is a specification. Using a display face for body text is the commonest typographic error in first projects, and it is usually made by someone who picked the face from a specimen shown at 90 pixels.

Where to get good type free

The free landscape is genuinely good now, and several of these are used in professional work daily.

- **Google Fonts** — over 1,500 families, nearly all under the SIL Open Font Licence, downloadable for local use as well as served from the CDN.
- **Fontshare**, from the Indian Type Foundry — high-quality families free for commercial use, with much stronger Indic coverage than most Western sources.

- **Ek Type** — a Mumbai foundry releasing open-licensed Devanagari and multi-script families, including Mukta.
- **Velvetyne** and **The League of Moveable Type** — smaller, more experimental, all open.
- **Open Foundry** — a curated index of open-licensed faces with proper specimens.
- **The system font stack** — costs zero bytes, renders natively, and is what most operating systems' own interfaces use.

The paid industry names are worth knowing because job advertisements name them: Monotype, Linotype, Hoefler, Commercial Type, Klim. You do not need any of them to do good work, and a great deal of professional work is set in Inter, Source Sans or IBM Plex.

Licensing, plainly and honestly

A font is software, and you are licensed to use it under specific terms. The main categories:

- **SIL Open Font Licence (OFL).** Free to use commercially, embed, modify and redistribute. Two real conditions: you may not sell the font on its own, and if you modify a font you must rename it if the original carries a Reserved Font Name. This covers most of Google Fonts.
- **Apache 2.0.** Used by some Google-released families such as Roboto. Similar permissions, different formalities.
- **Commercial desktop licences.** Usually cover installing the font on a set number of machines for producing artwork. They frequently do **not** cover embedding the font in a website, an app or a PDF sent to a client. Web use is typically a separate licence sold by pageview.

The practical trap: a font that came with your operating system or with a piece of design software is licensed for your use of that software, not for redistribution. Handing a client a logo file with the font outlined is fine; handing them the font file is usually not.

Where it is genuinely unsettled. Whether a typeface *design* can be protected at all differs by country. In the United States the letterform designs

themselves are generally not copyrightable while the font software is, which is why lookalike fonts are legal there. In much of Europe a typeface design can be registered and protected. There is no single global answer, the boundaries are litigated rather than settled, and this is not legal advice — if a project's budget depends on it, ask a lawyer in the relevant jurisdiction.

The related question of whether typefaces in a training set create rights in generated letterforms is newer and less settled still. Anybody telling you it is resolved is describing one jurisdiction's current position as though it were the world's.

The default that is hard to beat

If you do not have a reason for a particular face: one well-drawn sans with many weights, for everything, with the size scale and spacing doing the work. It is not a compromise. A great deal of the best interface typography of the last decade is exactly this.

BEFORE YOU MOVE ON

A studio buys a desktop licence for a commercial typeface, uses it in a client's brand, and uploads the font files to the client's website so the headings render correctly. What is the most likely problem?

A. The font cannot be used commercially at all under a desktop licence, so the brand work itself is unlicensed.

B. Typeface designs are not copyrightable anywhere, so no licence restriction on the letterforms can be enforced against either the studio or the client in any jurisdiction.

C. The font must be renamed before it is served from a different domain, because the original carries a Reserved Font Name that applies to redistribution.

D. Desktop licences typically cover producing artwork on licensed machines but not embedding the font on a website, which is usually a separate web licence.

27 · 8 min

Two typefaces, and why you probably want one

THE ONE THING TO KEEP

One family with many weights is the default and a second typeface needs a stated reason; when you do pair, cross a class boundary, match x-height and apparent weight, and write down which face does which job.

The default is one

One family with a full range of weights will carry almost any project. Regular for body, semibold for subheadings, bold for the page title, light for a large display line, italic for emphasis. That is five distinct typographic treatments from one file set, all guaranteed to sit together because one designer drew them.

A second typeface is a decision that needs a reason. Good reasons exist — a serif for long reading against a sans for interface labels, a monospace for code, a distinctive display face for a masthead that would be tiring in a paragraph. A vague sense that one font is boring is not a reason; it is usually a symptom of a hierarchy problem, and adding a face will not fix it.

If you do pair, pair across a class boundary

The mechanism is the threshold rule again. Two faces that are slightly different read as a mistake; two that are clearly different read as a decision.

So pair across categories — a serif with a sans, a sans with a monospace, a geometric sans with a humanist serif. Do not pair two humanist sans faces, or two transitional serifs. The reader cannot tell whether the difference is intentional, and the pair will look like a font failed to load somewhere.

What to match when you pair:

- **x-height.** Two faces with wildly different x-heights at the same point size will look like different sizes. Adjust the point size of one until the lowercase letters are the same height.
- **Apparent weight.** A regular in one face may be visibly heavier than a regular in another. Compare the two at the same size and adjust the weight rather than assuming the names mean the same thing.
- **Width and rhythm.** A condensed face beside a wide one produces a texture difference that competes with the hierarchy.

Superfamilies remove the problem

Several families ship multiple classes designed to work together, with shared metrics and shared skeletons. This is the lowest-risk pairing available and all of these are free:

- **IBM Plex** — Sans, Serif, Mono, Condensed, plus coverage for several Indic scripts.
- **Source** — Source Sans, Source Serif, Source Code, from Adobe under the OFL.
- **Noto** — Sans and Serif across an extraordinary range of scripts, which matters if your content is multilingual.
- **Roboto** — Sans, Slab, Mono, Condensed.
- **Fira** — Sans, Sans Condensed, Code.

Using a superfamily is not a lesser choice. It is the choice a lot of professional work makes, because the compatibility is guaranteed rather than eyeballed.

The test

Set the same short phrase in both faces, at matched x-height, one directly above the other, at the size each will actually be used. Then look at it small, and then blurred.

Three outcomes:

- **Clearly different, and comfortable together.** A pairing.

- **Nearly the same.** Not a pairing. Drop one.
- **Fighting.** Usually a mismatch of apparent weight or width rather than of class. Adjust before abandoning.

This takes about two minutes and replaces a large amount of published advice about which fonts go with which, most of which is one person's taste presented as a rule.

Assign jobs, then hold the line

The part that fails in practice is not the choice of faces but the discipline afterwards. Write down which face does what:

```
Display face → page titles only, 40px and above
Text face    → everything else, including all headings below
40px
Mono         → codes, reference numbers, tabular data
```

Without that, the display face migrates into subheadings, then into buttons, and after three months the distinction has dissolved and both faces are doing everything. The pairing has stopped carrying information, which was the only thing it was for.

The honest note on cost

Each additional family costs bytes. A single weight of a full-coverage web font is commonly 15 to 40 kilobytes in WOFF2; a family with four weights and matching italics is easily 200 kilobytes, and with Devanagari or CJK coverage it can be far larger. On a slow connection that is a visible delay before the text appears, or a visible jump when it swaps in.

Subsetting to the characters and scripts you actually use, and serving with `font-display: swap`, mitigates it. The real mitigation is using fewer families, which is where this lesson started.

BEFORE YOU MOVE ON

A team pairs Open Sans with Lato and reviewers keep saying the headings look like a rendering bug rather than a design choice. What is the underlying issue?

A. Both are humanist sans faces with similar skeletons, so the difference is too small to read as deliberate.
B. The two families have different x-heights, so the headings appear at a different size from the one that was specified in the design.
C. Lato lacks a true italic, so any emphasised text in the headings is being synthesised by shearing the roman and looks wrong.
D. Open Sans and Lato have different apparent weights at the same numeric value, so the specified semibold renders heavier in one face and the hierarchy inverts between the two.

28 · 9 min

Rivers, rags and where the line breaks

THE ONE THING TO KEEP

Justification absorbs its adjustment in word spaces, so a narrow measure with few spaces per line produces rivers; justify only at 60 characters and up with hyphenation properly enabled, and remember that hyphenation silently fails without a declared language.

Why justified text goes wrong

Justified text has flush edges on both sides. To achieve that, the setting engine stretches or compresses the spaces between words until the line fills the measure exactly.

The number of spaces available to absorb that adjustment is roughly the number of words on the line. On a 70-character measure there might be twelve words and therefore eleven spaces, so each absorbs a small share. On a 35-

character phone measure there might be five words and four spaces, so each has to stretch four or five times as far.

When several consecutive lines have wide spaces in nearby positions, they connect into a pale channel running down the paragraph. These are **rivers**, and once you have seen one you will see them everywhere. They are distracting in a specific way: they compete with the horizontal reading motion by offering a vertical path.

The rule that follows is mechanical, not stylistic:

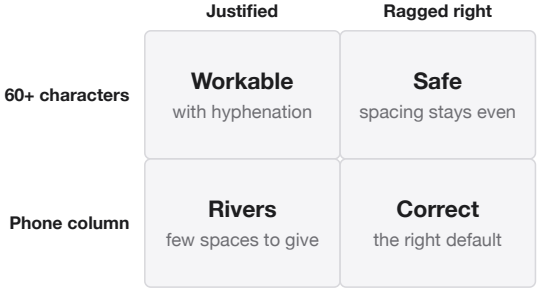
> *Justify only at a wide measure — 60 characters and up — and only with hyphenation enabled. Never justify at phone width.*

Hyphenation is the missing half

Hyphenation gives the engine somewhere else to absorb the adjustment: it can break a long word rather than stretch every space on the line. Justification without hyphenation is the setting that produces the worst rivers, and it is the default in most web pages and word processors.

On the web, hyphenation requires two things and people usually do one of them:

Justified or ragged, at two measures



Justification absorbs its adjustment in the word spaces, so the fewer spaces a line has, the further each one must stretch. A 35-character line has about four.

```
p { hyphens: auto; }
```

```
<html lang="en-GB">
```

The `lang` attribute is not optional. Hyphenation is language-specific — the rules for breaking English words are not the rules for German or Hindi — and without a declared language the browser has no dictionary to consult and will silently do nothing. This is one of the most common invisible faults on the web: the CSS is present, the attribute is missing, and nobody notices that hyphenation never ran.

Ragged is usually right

Ragged-right setting — flush left, uneven right edge — keeps word spacing constant, which keeps the paragraph's texture even. It is the correct default for screens and for narrow measures.

A good rag has a shape. What to look for:

- **No line dramatically shorter than its neighbours.** A rag that varies between 90% and 100% of the measure looks accidental; one varying between 75% and 100% looks like a rag.
- **No accidental shape.** If the right edge forms a wedge, a curve or a staircase over several lines, the eye reads it as an object.
- **No stack of identical endings.** Three consecutive lines ending in the word and, or two consecutive hyphens, both draw attention.

You adjust a rag by changing the measure slightly, by inserting a non-breaking space to force two words to stay together, or by rewriting. In long web text you cannot control it, which is fine — this is a fine-typography concern for headlines, pull quotes and print, not for flowing body text that reflows on every device.

Headlines are where line breaks matter most

A headline that breaks badly changes the sentence. Consider:

```
The committee rejected the plan to
close the library
```

against

```
The committee rejected the plan
to close the library
```

The second breaks at a phrase boundary and reads in one pass. CSS now has a declaration for this:

```
h1, h2 { text-wrap: balance; }
p { text-wrap: pretty; }
```

`balance` evens the line lengths across a short block, which is what you want for headings of up to about four lines. `pretty` leaves the lines alone but avoids leaving a single word on the last line. Both are recent, both degrade harmlessly where unsupported, and `balance` is deliberately limited to short blocks because the computation is expensive.

Widows and orphans

Two terms that get swapped constantly. The conventional definitions:

- An **orphan** is the first line of a paragraph left alone at the bottom of a column or page.
- A **widow** is the last line of a paragraph left alone at the top of the next one.

A single word alone on the last line of a paragraph is also commonly called a widow, or a runt. All three are worth avoiding because they look like a mistake in the production rather than a property of the text.

In CSS, `orphans` and `widows` take effect only in paged contexts — printing and multi-column layouts — and do nothing in ordinary continuous flow. In Scribus, LibreOffice Writer and any word processor, the equivalent setting

lives in the paragraph controls and does work. This is a case where print tools are simply better equipped than the browser, and it is worth knowing which environment you are in before you promise anybody a fix.

Never centre a paragraph

Centred text has no straight left edge, so the return sweep at the end of each line has no target and has to search. Over three lines it is a stylistic choice. Over ten it is a real cost, paid by every reader, for nothing.

BEFORE YOU MOVE ON

A developer adds hyphens: auto to a justified article and the rivers remain exactly as before. The CSS is definitely applied. What is the most likely cause?

A. Hyphenation only applies to ragged-right text, so it has no effect once justification is switched on.

B. The measure is too wide for hyphenation to find break opportunities, so the engine still has to stretch spaces to fill each line.

C. No lang attribute is set, so the browser has no hyphenation dictionary and does nothing.

D. Hyphenation requires an explicit hyphenate-limit-chars value, and without one the browser applies a default that refuses to break any word in ordinary body text.

29 · 8 min

Letter spacing, and the sizes type was drawn for

THE ONE THING TO KEEP

Default letter spacing is drawn for one size, so display type wants negative tracking and all-caps labels want 6 to 10 per cent positive — and a font with an optical size axis adjusts the letterforms themselves rather than only the gaps.

Three different things with similar names

Kerning adjusts the space between one specific pair of letters. AV, To, Ta and Wa need it, because the shapes nest. Typeface designers ship kerning tables with the font, and the browser uses them by default in most cases; `font-kerning: normal` makes it explicit.

Tracking, or letter-spacing, applies the same adjustment uniformly across a run of text. This is the one you control.

Optical size is the idea that a typeface should be drawn differently for different sizes. It is the one most people have never heard of, and it explains the other two.

Why large type needs tighter tracking

A typeface's default spacing is designed for a particular size — usually somewhere in the range of running text. When you scale the letters up to 60 pixels, the spaces scale up with them. But apparent gaps do not grow the way they measure: at large sizes the same proportional space reads as a hole.

So headings need negative tracking and small text needs positive. Working numbers:

```
60px display    letter-spacing: -0.02em to -0.03em
40px heading    letter-spacing: -0.01em to -0.02em
16px body       letter-spacing: 0
12px caption    letter-spacing: 0.01em
12px ALL CAPS   letter-spacing: 0.06em to 0.10em
```

The all-caps line is the one to memorise. Capitals are spaced for use as initials inside lowercase words, not for running together in a word of their own, so a caps label set at default tracking looks cramped. Adding 6 to 10 per cent is the standard correction, and a caps label without it is one of the most recognisable marks of untrained typesetting.

Use `em` rather than pixels so the tracking scales with the size.

What optical size actually was

In metal type, every size was cut separately. A punchcutter making 6-point type gave it sturdier strokes, larger counters, shorter extenders and looser spacing, because at 6 points the ink spreads and the eye needs help. The same face at 48 points got finer hairlines, tighter spacing and more delicate detail.

Photocomposition and then digital type threw this away: one outline, scaled to any size. That is why a heading in a digital face often looks slightly loose and slightly clumsy, and why small text in the same face looks slightly fragile. You have been compensating by hand with tracking ever since, without necessarily knowing that is what you were doing.

Variable fonts brought it back as a real axis. A font with an `opsz` axis contains a design space, and the renderer interpolates the appropriate drawing for the size in use:

```
body { font-optical-sizing: auto; }
```

That is the default in modern browsers where the font supports it, and it applies the correct design automatically. Families with a genuine optical size axis include Roboto Flex, Source Serif 4, Newsreader and Literata — all free. When

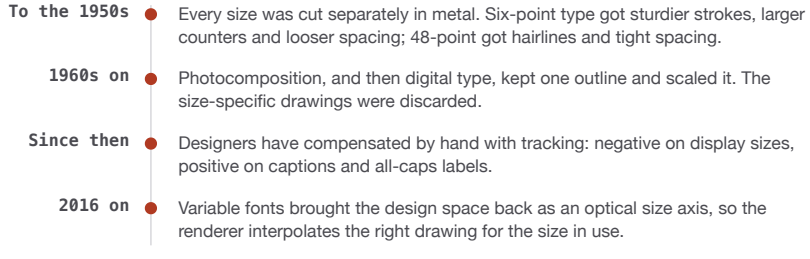
a face has this, you need much less manual tracking, because the shapes themselves are being adjusted rather than just the gaps.

When to touch kerning by hand

Almost never in running text. The font's tables are better than your judgement across thousands of pairs.

There is one real exception: **a logotype or a short display line**, where five or six words are going to be reproduced thousands of times and the pairs are visible at large size. There you set the letters individually, and the usual method is to work on the *spaces* rather than the letters — adjust until each gap looks like it contains the same amount of air as its neighbours, which is a judgement about area, not distance. A common training exercise is to set a word upside down, where you cannot read it and therefore cannot be distracted from the shapes.

Optical size: thrown away, then recovered



Where a face carries the axis, font-optical-sizing set to auto applies it. Roboto Flex, Source Serif 4, Newsreader and Literata all have one, and all are free.

Inkscape, Krita and Scribus all support manual letter-by-letter kerning; so does any vector editor. This is not a paid-tool capability.

Where it goes wrong

- **Tracking body text to fit.** Adding letter-spacing to make a paragraph fill a space damages the word-shape recognition that reading depends on. Change the measure or the size instead.
- **Negative tracking at small sizes.** Below about 14 pixels, letters are already close to touching at typical screen resolutions, and negative tracking

makes them collide.

- **Tracking scripts that join.** In Devanagari the letters sit under a connecting headline, and in Arabic they join cursively. Adding letter spacing breaks the joins and can make the text unreadable rather than merely ugly. Set `letter-spacing: normal` explicitly for those blocks if a global rule is applying tracking site-wide.

That last one catches people out constantly on multilingual sites, because the rule was written for a Latin heading and applied to everything.

BEFORE YOU MOVE ON

A multilingual site applies letter-spacing: 0.05em to all headings for a refined look. The English headings improve; the Hindi headings become hard to read. What is happening?

A. Devanagari has a larger x-height equivalent than Latin, so the same tracking value is proportionally much larger and opens holes between the letters.

B. Devanagari letters sit under a connecting headline, and added tracking breaks that continuous stroke apart.

C. Hindi fonts lack kerning tables, so the added tracking accumulates with unresolved pair spacing and compounds the gaps.

D. The tracking is applied in em units, and Devanagari fonts define a different em square, so the computed spacing is several times larger than intended.

30 · 8 min

Capitals, italics and the cost of emphasis

THE ONE THING TO KEEP

Capitals flatten the word shapes that reading depends on, so reserve them for short labels set with CSS rather than typed, and emphasise with the weakest device that works — italic before bold before colour — because stacking them cancels the emphasis.

Why all-caps is slow to read

A word in lowercase has a distinctive outline: ascenders sticking up, descenders hanging down, a profile you can partly recognise before resolving the letters. Set the same word in capitals and the outline becomes a rectangle. Every word becomes the same shape, differing only in length.

Classic reading studies found continuous text in capitals to be read measurably more slowly than the same text in mixed case — the figures vary with the method, commonly in the region of 10 to 20 per cent. For a one- or two-word label the effect is negligible, because you are recognising a short known string rather than reading. For a paragraph it is real, and for a long headline it is enough to notice.

The practical division:

- **Fine in caps**: buttons, short labels, table headers, a two-word eyebrow above a headline, an acronym.
- **Avoid caps**: sentences, long headlines, anything over about four words, and any body text at all.

Set caps with CSS, not with the caps lock key

If a label should display as uppercase, write it in sentence case and transform it:

```
.eyebrow { text-transform: uppercase; letter-spacing: 0.08em; }
```

This matters for three practical reasons. The underlying text stays readable when somebody searches it or copies it. Some screen readers pronounce a string of capitals letter by letter, so a genuinely uppercase word can be read out as individual letters. And you can change the decision later without editing the content.

The same applies to small capitals. Use `font-variant-caps: small-caps` — or `all-small-caps` for a string that is already uppercase — rather than setting capitals at a smaller size. Real small caps are drawn with the correct stroke weight; scaled-down capitals are visibly thinner than the text around them, which is why faked small caps always look slightly wrong.

Real italics, and fake ones

A true italic is a separate design. The letters have a different skeleton — a single-storey a, a cursive f with a descender, entry and exit strokes that imply a pen. A true oblique, in some sans families, is a deliberately redrawn slanted roman.

What you get when the italic file is missing is neither: the renderer shears the roman outlines by a fixed angle. Circles become ellipses on a slant, stems thicken inconsistently, and the result looks cheap in a way readers notice without diagnosing.

This is why the earlier checklist asks whether a family has italics. If it does not, use weight or colour for emphasis instead of asking the software to invent a style.

An emphasis ladder

Emphasis inside running text has a natural order, from lightest to heaviest:

1. **Italic.** Almost invisible in the texture of the paragraph until you reach it. The correct default for a title, a term being introduced, or a word carrying stress.

2. **Bold.** Visible from across the room, which means it changes the typographic colour of the whole block. Correct for a term a scanning reader is hunting for; wrong for stress inside a sentence.
3. **Colour.** Strong, and on the web it collides with the convention that coloured text is a link.
4. **Background highlight.** Loudest, and best reserved for search results or a genuine marker.

The rule is the same as the emphasis budget from the first block: use the weakest device that does the job. A sentence with a bold word, an italic word and a coloured word in it has three levels of emphasis and therefore none.

Underline deserves its own line: on the web, reserve it for links. An underlined non-link is tried constantly and produces a stream of failed clicks. In print, underlining is a typewriter convention that italics replaced, and it damages descenders.

The combination that always looks wrong

Bold, italic, underlined, in colour, in capitals, all at once. This appears in almost every first design, and the reason it looks bad is not that the individual devices are bad. It is that five channels stacked on one word tell the reader that the author could not decide which mattered, and the word is now so loud that the sentence around it stops existing.

One channel, at full strength, is the correct answer nearly every time.

A note on emphasis in other scripts

Italic is a Latin convention with a specific history. Devanagari has no italic tradition, and slanting it produces something that reads as a distortion rather than as emphasis. CJK typography does not italicise either; emphasis is conventionally marked with a heavier weight, or with emphasis dots set beside or above the characters, which CSS supports as `text-emphasis`.

If your content is multilingual, an emphasis rule written as make it italic will quietly produce nonsense in half your languages. Weight and colour transfer

across scripts; slant does not.

BEFORE YOU MOVE ON

A team types its navigation labels in capital letters directly in the content, then reports that a screen reader announces HOME as individual letters. Switching to text-transform fixes it. Why?

A. Screen readers ignore CSS entirely, so any text-transform value leaves the announced string unchanged from what was written.

B. text-transform applies the change at render time, so the underlying text remains Home and is announced as a word.

C. Screen readers apply a different pronunciation dictionary to styled text than to literal text, and the styled version matches a known word.

D. A string of capitals in the content is treated as an acronym by the accessibility layer, and only an explicit aria-label can override that classification once it has been made.

31 · 8 min

Figures, dashes and the details nobody teaches

THE ONE THING TO KEEP

Use tabular figures for anything that changes in place or sits in a column, old-style figures inside prose, the correct dash for the job, and a locale-aware formatter for digit grouping — because Indian grouping differs from Western and hard-coding one is a decision to be wrong somewhere.

The counter that jiggles

A live timer counts down: 0:59, 0:58, 0:57. The digits shift left and right by a pixel or two on every tick, and the whole element twitches. A price column in a

table has its digits almost but not quite aligned.

Both have the same cause. Most typefaces ship **proportional figures**, where a 1 is narrower than a 0, because that looks better inside a sentence. For anything that changes in place, or anything in a column, you want **tabular figures**, where every digit occupies the same width:

```
.timer, .price-column { font-variant-numeric: tabular-nums; }
```

One declaration. It fixes a class of small ugliness that people otherwise try to solve with fixed widths and monospace fonts.

Lining and oldstyle

A second distinction, independent of the first. **Lining figures** are all the same height as the capitals. **Oldstyle** or text figures have varying heights with ascenders and descenders, so 3, 4, 5, 7 and 9 descend and 6 and 8 ascend.

Oldstyle figures sit more comfortably inside lowercase prose, because they have the same up-and-down profile as the letters around them. Lining figures stand out, which is right in a table, in an all-caps setting, or where a number is the point.

```
p { font-variant-numeric: oldstyle-nums; }
table { font-variant-numeric: lining-nums tabular-nums; }
```

Not every font contains both sets. Check before you specify, because a request for a variant the font lacks silently does nothing.

Dashes, and the three of them

- **Hyphen** `-` joins words and breaks them at line ends: well-known, co-operate.
- **En dash** `–` marks ranges and connections: 1998–2004, the London–Delhi route. The width of an n.

- **Em dash** `—` marks a break in a sentence — like this one. The width of an m.
- **Minus sign** `−` is a fourth character, wider than a hyphen and aligned with the figures. `−5°C` uses it; `-5°C` is using a hyphen.

British practice typically uses a spaced en dash for the parenthetical break where American practice uses a closed-up em dash. Both are correct within their conventions; what is not correct is a hyphen standing in for either.

Quotes and primes

Typographic quotes curl and point: ‘single' and "double". The straight marks ' and " are typewriter substitutes, and they have their own correct use as **prime** and **double prime** for feet and inches, minutes and seconds: 5′ 11″.

So a shop sign reading 6" Sandwich is using a prime correctly and an apostrophe in a name like O'Brien written with a straight mark is not. Word processors convert automatically. Web content generally does not, and the straight apostrophe is the single commonest typographic error on the internet.

Indian digit grouping

Worth its own section for this audience, because getting it wrong is immediately visible to a large number of readers.

The Indian numbering system groups the last three digits and then in pairs: 1,20,000 rather than 120,000; 1,00,00,000 for a crore. A page that displays Indian rupee amounts in Western grouping looks foreign in a way that is hard to unsee.

This is a formatting decision, not a font one, and JavaScript handles it:

```
new Intl.NumberFormat('en-IN').format(1200000)   // "12,00,000"
new Intl.NumberFormat('en-IN', {
  style: 'currency', currency: 'INR'
}).format(1200000)                                //
"₹12,00,000.00"
```

The same API handles every other locale's conventions, including the comma-as-decimal-separator used across much of Europe. Hard-coding a grouping format is a decision to be wrong somewhere.

Spaces that should not break

Some pairs must not be split across a line: a number and its unit, an honorific and a name, a figure and its currency symbol.

```
10 kg    ₹1,200    Dr Sharma    Figure 4
```

A non-breaking space also does not collapse, which occasionally matters. The related character, a **thin space**, is what typesetters use inside a large number in some conventions, and a **hair space** is thinner still.

The ellipsis and the degree sign

Three full stops in a row are not an ellipsis. The character … has its own spacing and will not break across a line. Likewise the degree sign ° is not a superscript o, and the multiplication sign × is not a lowercase x — 1920 × 1080 against 1920 x 1080 is a small difference that a designer's eye catches instantly.

None of this is decoration. Each of these details exists because somebody needed the distinction and the typewriter could not make it. The characters came back; most people just never found out.

BEFORE YOU MOVE ON

A dashboard shows a live count that visibly twitches left and right as the number changes. A developer switches the element to a monospace font and the twitching stops. What was the actual fault, and what is the better fix?

A. The element had no fixed width, so the container was resizing on each update; setting an explicit width would solve it without changing the typeface.
B. The font was using oldstyle figures whose varying heights created the impression of movement; requesting lining figures would fix it.
C. The font's proportional figures have different widths, and font-variant-numeric: tabular-nums fixes it without a typeface change.
D. The browser was re-shaping the text run on each update and applying kerning between the digits, which monospace fonts avoid by having no kerning pairs at all.

32 · 8 min

The grey of a paragraph

THE ONE THING TO KEEP

The even grey texture of a text block comes from weight, size, spacing and leading acting together, so check it by blurring or inverting the page and fix the specific cause of each patch rather than adjusting the settings at random.

A block of text is a texture

Blur a well-set paragraph and it becomes an even field of grey. Blur a badly set one and you see patches: dark clots where something is bold or cramped, pale holes where spacing has opened up, and channels running down where word spaces have aligned.

That overall greyness is called **typographic colour**, and the word colour here has nothing to do with hue. It is the density of ink per unit of area, and it

is produced by four things together: the weight of the strokes, the size, the letter and word spacing, and the line height.

Even colour is the mark of competent setting. It is also easy to check and easy to fix, which makes it one of the highest-return habits in this course.

How to check it

Three methods, all free and all taking seconds:

1. **Blur.** Export the block and apply a Gaussian blur of about 3 to 4 pixels at body size. Even grey is good.
2. **Turn it upside down.** Rotate the page 180 degrees. You can no longer read it, so you stop attending to the words and start seeing the texture. Any editor does this in one command.
3. **Squint at it from two metres.** The cheapest, and surprisingly reliable.

What you are looking for is uniformity. Not lightness — a dense, dark paragraph can have perfectly even colour — but the absence of patches.

The five things that ruin it

Rivers, from the justification lesson. Vertical pale channels. Fix by un-justifying, widening the measure, or enabling hyphenation.

A bold word mid-paragraph. Bold creates a dark clot visible from across the room, which is exactly what it is for when a scanner is hunting for a term, and exactly wrong when you only meant to stress a word in a sentence. Use italic for stress.

Acronyms in capitals. A paragraph containing NHS, HTML and PDF has three dark blocks in it, because capitals are taller and denser than lowercase. The traditional fix is small capitals, which have the same weight as the lowercase around them:

```
abbr { font-variant-caps: all-small-caps; letter-spacing:
0.03em; }
```

Where the font has no real small caps, setting acronyms about 8 to 10 per cent smaller than the body gets most of the benefit.

Links in a bright colour. A paragraph with six blue links has six attention-grabbing marks in it and reads as a list of links with some connective text. This is a real trade-off rather than a rule — the links have to be findable — and the usual resolution is a link colour close in lightness to the body text, distinguished by hue and underline rather than by being much darker or lighter.

Line height that is too tight or too loose. Too tight and the lines merge into a dark mass; too loose and the lines stop being a paragraph and become separate objects. The 1.4 to 1.6 range for body text on screen exists because that is where the horizontal band of a line is clearly separate from its neighbours without floating free.

Paragraph separation: indent or space, never both

A new paragraph is signalled either by a first-line indent or by a space between paragraphs. Both together is redundant, and it is the most common setting in documents made by people who have not thought about it.

- **Indent** is the book convention. It costs no vertical space, which is why books use it. The first paragraph after a heading is not indented, because there is nothing before it to separate from.
- **Space** is the screen convention. It suits short paragraphs and scanning.

One or the other. If you use space, a full blank line is usually too much: something like 0.75 of the line height reads as a separation without breaking the block apart.

Measure, size and colour interact

A setting is not a list of independent decisions. Increase the measure and you need more line height, because the return sweep has further to travel. Increase the line height and the block gets lighter, so it may need a slightly heavier weight to hold its own against a photograph beside it. Reduce the size and you need slightly more letter spacing.

This is why copying a set of numbers from an article does not work reliably: the numbers were a solution to somebody's combination of typeface, size, measure and medium. What transfers is the method — set it, blur it, look at the texture, adjust the thing that is causing the patch.

Colour in the other sense

One genuine overlap with hue: grey body text. Setting text at 60% grey instead of near-black lightens the typographic colour and is a common way to make a page feel calmer. It is also the single most common cause of failing contrast requirements, covered in the next block. If you lighten text for texture, check the ratio afterwards, because the texture argument and the legibility requirement pull in opposite directions and only one of them is measurable.

BEFORE YOU MOVE ON

A paragraph blurs to an even grey except for three dark spots. The text contains the acronyms NHS, PDF and HTML. What is the mechanism, and what is the conventional fix?

A. Acronyms are usually bolded by the content management system's default styles, and removing that emphasis restores the even texture.

B. The acronyms are being letter-spaced by an uppercase rule, and the extra tracking concentrates the strokes into a visible block.

C. Capitals in most faces are drawn with heavier stems than the lowercase, so setting them at the same size produces more ink and the fix is to reduce their stroke weight with a lighter font variant.

D. Capitals are taller and denser than the lowercase around them, and small capitals at matching weight remove the clots.

33 · 10 min

What breaks when the text is not in English

THE ONE THING TO KEEP

Line height, letter spacing, measure and the italic convention are all Latin defaults that degrade silently in other scripts — Devanagari and Indic scripts need more vertical room and no tracking, Urdu Nastaliq needs far more still, and the only reliable check is a native reader looking at real text in the real layout.

Almost every default you have been given is a Latin default

The 1.5 line height, the 45-to-75 character measure, the type scale, the italic convention, the letter-spacing rules — all of them were worked out for the Latin alphabet, and most of them degrade silently in other scripts. Nothing warns you. The page renders, the text is legible enough that a reviewer who does not read the script sees nothing wrong, and a reader who does read it can immediately tell that nobody checked.

This matters here because a large share of this course's readers work in Hindi, Marathi, Bengali, Tamil, Telugu, Urdu or Punjabi, often alongside English on the same page.

Devanagari

Used for Hindi, Marathi, Nepali, Sanskrit and others. Three structural facts change your settings:

The shirorekha. A horizontal headline runs along the top of the letters and joins the word together. This is why letter-spacing must be left at normal — tracking breaks the line into fragments, as covered earlier.

Matras above and below. Vowel signs stack above the headline and below the baseline, and conjunct consonants stack vertically too. The total vertical extent of a line is therefore much greater than the extent of a Latin line at the

same size, and a line height tuned for Latin will clip or collide. Devanagari generally wants roughly **1.6 to 1.8** where Latin wants 1.4 to 1.5.

Apparent size. At the same point size, Devanagari and Latin do not look like the same size, and which looks bigger depends on the two fonts. On a mixed Hindi-and-English page you often need a size adjustment for one of them, which `font-size-adjust` or a separate rule can supply.

Free, well-made Devanagari families, all openly licensed: **Noto Sans Devanagari** and **Noto Serif Devanagari**, **Mukta** and **Baloo** from Ek Type, **Hind** and the Devanagari families on Fontshare from the Indian Type Foundry, and **Tiro Devanagari** for text setting.

Arabic and Urdu

Right to left, which changes layout direction and not only text direction — the whole interface mirrors, including navigation, icons that imply direction, and progress. In CSS this is `dir="rtl"` on the element plus logical properties (`margin-inline-start` rather than `margin-left`), which then work in both directions from one stylesheet.

Letters change shape depending on position in the word — initial, medial, final or isolated — so the text is shaped by the font at render time and cannot be treated as a sequence of independent glyphs. Letter-spacing is again destructive.

Urdu is conventionally set in **Nastaliq**, a style with a steeply sloping baseline where each word cascades down and to the left. This needs dramatically more line height than anything in the Latin world — commonly 2.0 or more — and a font built for it, such as **Noto Nastaliq Urdu**. Setting Urdu in a Naskh font is legible but reads, to an Urdu reader, roughly the way an English paragraph set entirely in a condensed display face reads to you.

Numbers inside right-to-left text run left to right, which the bidirectional algorithm handles automatically and which produces genuinely confusing results when a string mixes scripts and somebody has hard-coded a direction.

Other Indic scripts

Tamil, Telugu, Kannada, Malayalam, Bengali, Gujarati, Odia and Gurmukhi each have their own metrics and their own stacking behaviour. Bengali has a headline like Devanagari; Tamil has fewer conjuncts and a wider, rounder texture; Malayalam has many. What is common to all of them is that they need more vertical room than Latin and that their glyph shaping cannot be interfered with.

Noto covers all of them, which is what the project was built for — the name is short for no more tofu, tofu being the empty rectangle a renderer shows when it has no glyph for a character.

CJK

Chinese, Japanese and Korean have no spaces between words, so lines may break almost anywhere, and there are rules about which characters may not begin or end a line. There is no italic tradition; emphasis is conventionally a weight change or emphasis dots beside the characters, which CSS supports as `text-emphasis`. Glyphs occupy a full em square, so line length is counted differently and a 66-character measure is not a meaningful target.

The font stack, and what actually happens

A font stack is evaluated per character, not per element:

```
body {
  font-family: 'Inter', 'Noto Sans Devanagari', system-ui,
sans-serif;
}
```

Latin characters come from Inter; Devanagari characters, which Inter does not contain, fall through to Noto. This works well and is the normal approach. It also means a mixed sentence is set in two typefaces, so the two need compatible weight and apparent size or the sentence will look assembled.

Set the `lang` attribute correctly on the element. It selects the right glyph forms where a font serves several languages, drives hyphenation, and tells

screen readers which pronunciation to use. A Hindi paragraph inside an English page needs `lang="hi"` on that paragraph.

The honest position

There is no automatic fix for any of this. The defaults are Latin, the tooling is better for Latin, most published guidance is about Latin, and the failures are invisible to a reviewer who cannot read the script.

What works is unglamorous: get real text in the target script from somebody who reads it, put it in the actual layout at the actual size, and have that person look at it. Placeholder text in the wrong script tells you nothing, and machine-translated text often contains shaping that a native reader would never write. One reader, ten minutes, will find more than a week of your own inspection.

BEFORE YOU MOVE ON

A site sets all headings with letter-spacing: 0.04em and line-height: 1.3. The English headings look refined; the Hindi headings look broken and the matras are clipped. Which pair of changes addresses the causes?

A. Increase the font size for Devanagari and reduce the tracking by half, so the extra size compensates for the vertical stacking.

B. Set letter-spacing to normal and raise the line height to about 1.7 for the Devanagari text.

C. Switch the Devanagari text to a serif family with shorter matras and keep both existing values, since the tracking is a brand decision.

D. Apply font-size-adjust so the two scripts match in apparent size, and add overflow visible so the matras are no longer clipped by their container.

34 · 9 min

Make it pop is a symptom report

THE ONE THING TO KEEP

Treat client feedback as a description of a symptom and do the diagnosis yourself, and use mirroring and a five-second test to see your own work as a stranger would.

Clients describe, they do not prescribe

"Make it pop." "It needs more energy." "Can the logo be bigger?" "I do not like the blue."

The beginner move is to hear an instruction and execute it. The logo gets bigger, the blue becomes green, the heading gets a shadow, and the client is still unhappy, now with a worse layout.

These sentences are symptom reports. They are the client telling you, in the only vocabulary they have, that something did not work when they looked at it. They are usually right that something is wrong and usually wrong about what. Translating symptom into cause is the job.

Three separate acts, belonging to different people: **description** (what it feels like — theirs), **diagnosis** (what is causing it — yours), **prescription** (what to change — yours). Trouble comes from accepting their prescription and skipping your diagnosis.

The common translations

Honest, from the mechanisms in the earlier lessons:

- **"Make it pop."** Almost always insufficient hierarchy — nothing arrives first — or insufficient contrast. Occasionally too little whitespace around the main element. Almost never a gradient, and never a drop shadow.

- **"The logo should be bigger."** Usually means "I could not tell whose this was in the first second". Size is one fix; isolation is usually a better one, and a small mark alone in clean space reads as more confident than a large one wedged into a corner.
- **"It feels cheap."** Usually inconsistency: four typefaces, uneven gaps, a stretched photograph, three corner radii. Run the inventory check.
- **"It needs more energy."** Usually a crop, a diagonal, and a real contrast jump — not more colours. Adding colours reduces energy, because it removes the difference that made one thing loud.
- **"I do not like the blue."** Sometimes exactly that, and sometimes the blue is the only thing they have a word for.

The two questions that do the most work:

1. "When someone sees this for two seconds, what should they walk away knowing?"
2. "What are they walking away with now?"

The gap between those two answers is the actual brief, and it is often the first time anyone has written it down.

Never ask whether they like it

"Do you like it?" has no answer that helps you, and it invites a person with no training to redesign your work out loud. Ask a task question instead: "which of these two gets the price across faster?" Taste has no answer; a task has one, and the client is genuinely the better judge of their own customers.

Show two or three options. One invites line edits, because there is nothing to compare it against; eight invites a request for a combination of four and six. Never include an option you do not want chosen — people choose it, and then you have to build it.

Critiquing your own work

You cannot see your own layout because you remember making it. These are the ways round that, in rough order of value, and every one of them is free.

- **Sleep on it.** Time weakens your memory of the intention, so more of what is actually on the page reaches you.
- **Mirror it.** Flip the whole thing horizontally. This is the single most effective trick in the list: it breaks the reading habit, so you stop reading and start seeing shape, and alignment and balance faults jump out immediately. Every tool has it — Image, then Transform, then Flip Horizontal in GIMP, Photopea and Krita. Flip it back to work.
- **Thumbnail it, desaturate it, print it**, and look at it on a phone you did not design on.
- **The five-second test.** Send it to a friend, tell them they get five seconds, then ask two questions: what was it about, and what were you supposed to do next. This takes one message and gives you more than an hour of staring.

Taking criticism without losing the day

Separate "this does not work" from "you are not good". The first is information about an artefact; the second is a story you added. Ask for the symptom, not the fix: "what were you looking for when you got stuck?" beats "what would you like me to change?"

You are allowed to defend a decision once, with a reason, and then let it go. If three different people trip over the same element, they are not all wrong about it, however clear it is to you.

Where AI fits in this, and where it does not

You can send a screenshot to a multimodal model and ask what it sees first, second and third. That is a cheap approximation of the five-second test, available at any hour, and it will reliably catch contrast failures, alignment drift and text that is too small.

Two honest limits. It agrees too easily — ask it to name three problems, ranked, rather than "is this good", because "is this good" returns a compliment. And it does not know your audience or what the thing is for, so it can tell you the eye lands on the wrong element and not whether the right element

was chosen. Getting useful answers out of a model is its own skill: prompting and evaluating AI cover it.

If any imagery in the piece was generated rather than photographed or drawn, it needs a visible label on the work itself. Who owns generated output is genuinely unsettled and differs by country — AI images sets out the state of the disagreement, and a lawyer in your country is the person who answers it for your contract.

Today

Take your most recent piece. Write down what you want seen first, second and third. Mirror it and look for one thing you had not noticed. Then send it to one person for five seconds and compare their answer with your list. Where those two disagree is your next hour of work.

BEFORE YOU MOVE ON

A client says "make the logo bigger" on a poster where the logo is already large and well placed. What is the most useful next move?

A. Enlarge it as asked, since the brief belongs to the client and they are paying for the work.

B. Explain that enlarging it will break the hierarchy, and decline the change.

C. Ask what they could not tell in the first second; "bigger" usually reports that the sender was not identifiable fast enough, which isolation and placement often fix better than size.

D. Produce eight versions at different logo sizes and let them pick one.

CASE STUDY · MODULE 3

The scholarship leaflet whose Hindi was clipped

A small education trust in Patna, printing 20,000 bilingual leaflets about a state scholarship with a fifteen-day application window.

The trust had one designer, working on a laptop with free software, and a printer who needed the file by Thursday evening to deliver on Monday. The leaflet was A5, folded, English on one side and Hindi on the other, with the eligibility rules, the documents needed, the deadline and the trust's helpline.

The English side was set in a well-drawn geometric sans the designer had used before, at 16 points with a line height of 1.4. It looked clean. The Hindi side was set in the same typeface, at the same size, with the same line height, because that was the obvious way to make the two sides look like one leaflet.

What went wrong, and why nobody saw it

The typeface contained no Devanagari at all. When the Hindi text was pasted in, the software silently fell back to whatever the system had, which was a default Devanagari face with different proportions. Nothing warned anybody. The text rendered, the layout held, and the proof looked acceptable at the size it was viewed on screen.

A volunteer who reads Hindi looked at a printed proof and found two faults in about ninety seconds.

First, the line height. Devanagari stacks vowel signs above the headline and below the baseline, and conjunct consonants stack vertically as well, so a line occupies far more vertical space than a Latin line at the same size. At 1.4 the matras of one line were touching the headline of the line below, and in three places on the folded panel a vowel sign had been clipped by the edge of a tinted box behind the text.

Second, the letter spacing. A site-wide rule the designer used for all body text added a small positive tracking, which is correct for Latin captions and wrong

here: Devanagari letters sit under a connecting headline, and adding space between them breaks the line into fragments. The volunteer's description was that the words looked as though they had been pulled apart.

Neither fault was visible to the designer, who does not read Devanagari. Both were immediately visible to somebody who does.

The decision

It was Thursday afternoon. The fixes were not large: set the Hindi in an openly licensed Devanagari family, raise the line height to about 1.7, set letter spacing explicitly to normal for that block, and re-flow the panel, which would change where the text broke and therefore the fold.

The estimate was a full working day, because re-flowing the Hindi changed the panel depth, which changed the English side's alignment, which meant re-proofing both. That pushed the file to Friday evening and delivery to Wednesday.

The cost of the delay was specific. The application window was fifteen days and had already started. Two days of the distribution plan — the district outreach van and the Saturday camp at two schools — would be lost, and the trust estimated those two days at somewhere between two and three thousand leaflets reaching families directly rather than sitting in a bundle.

The cost of shipping on Thursday was also specific, and harder to put a number on. About seventy per cent of the intended readers would read the Hindi side and not the English one. What they would receive was a leaflet with clipped vowel signs and broken words — legible with effort, and carrying the unmistakable signal that the organisation asking for their documents had not checked its own Hindi.

The trustee who had to decide put it plainly: two thousand families who do not get a leaflet, against fourteen thousand who get one that tells them the trust does not read their language.

STOP HERE

Before turning the page: what would you have done, and what would it have cost if you were wrong? Write it down. The point of the next section is to compare.

WHAT HAPPENED

The trust delayed. The Hindi was re-set in Mukta at 17 points with a line height of 1.7 and letter spacing at normal, and the English side moved to a matching weight of the same family's Latin so the two sides no longer had to be reconciled by eye. The printer accepted the Friday file and delivered on Wednesday.

The outreach van lost two days. The Saturday camp was rescheduled to the following weekend and reached fewer people than planned. The trust's own count put the shortfall at about 1,800 leaflets distributed directly, some of which were absorbed by a longer counter-distribution at the two schools later.

The change the trust kept was procedural rather than typographic. Every bilingual piece since then has had one rule attached to it: a person who reads the script looks at real text, in the real layout, at the real size, before the file goes to a printer. The volunteer who found both faults in ninety seconds is now listed on the production checklist, which is the cheapest line on it.

Both faults passed every check the designer ran. What property of the failure made it invisible to them?

The defaults being applied were Latin defaults, and they degrade silently in other scripts. A font stack falls through per character, so the missing Devanagari coverage produced a substitution rather than an error; a line height tuned for Latin ascenders and descenders produced collisions rather than a warning; and tracking that is correct for a Latin caption breaks the shirorekha that joins a Devanagari word. Nothing renders as broken, so a reviewer who does not read the script sees nothing wrong.

Why was raising the line height not enough on its own?

The line height fixed the collisions and the clipping, but the letter spacing was a separate fault with a different mechanism. Devanagari letters are joined by a head-

line, so positive tracking fragments the word itself rather than merely loosening it. The two faults were independent, and a global rule written for Latin body text was the cause of the second one, which is why the repair had to set letter spacing explicitly to normal for that block.

The trustee framed the decision as two thousand families against fourteen thousand. What does that framing get right, and what does it leave out?

It gets right that both options had a real, countable cost, which is what made it a decision rather than a correction. What it leaves out is that the two costs are not the same kind: the missing leaflets were a one-off loss in one campaign, while a leaflet that signals the trust does not check its own Hindi affects how the next leaflet is received. The procedural change the trust adopted afterwards is the part that addresses the second cost.

MODULE 3 · TEST

Check what stuck

6 questions, one right answer each. This one is for you, not for a record: answers and the reason behind each are at the back. If two questions from the same module go wrong, re-read that module before the exam rather than after.

1 A page of text feels tiring and its lines run about 120 characters. Why does raising the font size from 16 to 18 pixels not help?

A. Because the return sweep still travels too far and lands on the wrong line

B. Because a typeface's x-height is the only thing that affects how large text feels

C. Because line height has to be increased at the same time as the size

D. Because 18 pixels is not a value on a modular scale

2 You swap a typeface with an x-height ratio of 0.52 for one at 0.40, keeping the declared size at 16 pixels. What happens?

A. It looks the same size but less legible, because point size measures the letters

B. It looks smaller, because the x-height falls from about 8.3 pixels to 6.4

C. It looks identical in size, because both are set at the same point size

D. It looks larger, because a smaller x-height means longer ascenders

3 Why does justified text produce rivers at a 35-character measure but not at 70?

A. Narrow columns use a smaller point size, so the spaces are proportionally larger

B. Justification stretches the letterforms rather than the spaces at narrow widths

C. Short lines contain fewer word spaces, so each must stretch much further

D. Short measures force more hyphenation, and the hyphens align down the page

4 A stylesheet sets hyphens: auto and no hyphenation appears anywhere. What is the most likely cause?

A. Hyphenation applies only to justified text, and this text is ragged
B. The text sits inside a flex container, where hyphenation is not applied
C. The typeface contains no hyphen glyph, so the browser suppresses the break
D. The document has no lang attribute, so the browser has no dictionary

5 Why is a paragraph set in capitals measurably slower to read than the same paragraph in mixed case?

A. Every word becomes a rectangle, so the outline the eye uses has gone
B. Capitals have smaller counters, which fill in at body sizes
C. Screen readers pronounce capitals letter by letter, which slows everybody
D. Capitals are wider, so the measure runs past 75 characters

6 A multilingual site has one rule for emphasis: set it in italic. What breaks?

A. Right-to-left scripts slant in the other direction, so the rule inverts itself
B. Devanagari and CJK have no italic tradition, so the text reads as distorted
C. Italic is unavailable in variable fonts, so the axis is ignored
D. Italics drop contrast below 4.5 to 1 in most typefaces

MODULE 4

Colour, measured

Colour is where design has the most measurable answers and the most unreferenced folklore. This block gives you the arithmetic — the luminance formula behind every contrast checker, why the lightness slider in HSL is not lightness, what colour vision deficiency actually removes — and it is equally clear about where the standards are contested, where cultural meaning has no universal answer, and where a screen cannot tell you what will come off a press.

> BY THE END OF THIS MODULE YOU CAN
>
> Build a palette from perceptual lightness ramps rather than from hue, state the contrast ratio of every text pair and the criterion it is being judged against, name the specific conditions under which that ratio overstates legibility, and give any colour-carried meaning a second non-colour encoding

35 · 10 min

A palette will not save an unreadable page

THE ONE THING TO KEEP

Build palettes by moving lightness rather than hue, check every text pair against a contrast number, and never let colour be the only thing carrying a meaning.

Why palette generators disappoint

The usual first move is to visit a colour-palette site, spin the wheel until five swatches look pleasant together, and paste them in. The result is often worse than three greys would have been.

The reason is mechanical. Most of those tools vary **hue** while holding lightness roughly constant, because that is what makes a strip of swatches look harmonious. Convert the result to greyscale and the five colours collapse into nearly the same tone. You now have a page where nothing is lighter or darker than anything else: no ranking, and text sitting on its background at about two to one.

Hue is the least useful of the three properties you control. Lightness is the most useful, and it is the one those tools hold still.

Use hue, saturation, lightness — and mostly move lightness

The working model is HSL: hue (which colour), saturation (how much of it), lightness (how close to white or black). Photoshop's picker is HSB, sometimes called HSV, where the third slider is brightness and full brightness gives the pure colour rather than white. Photopea, GIMP and Krita offer both. Know which you are looking at, because the same number means different things.

A method that takes two minutes:

1. Pick **one** hue you like.

2. Build a five-step ramp by moving lightness only: roughly 95, 80, 55, 30, 12. Drop saturation a little at the extremes so the pale end is not milky and the dark end not muddy.
3. Add **one** accent hue, well separated on the wheel, reserved for the single thing you want people to act on.

That is a complete palette. It has hierarchy built into it, because it was built out of the property that produces hierarchy.

Contrast is a number you can check in twenty seconds

Text legibility is not a matter of opinion. There is a standard ratio, computed from the relative luminance of two colours, that runs from 1:1 (identical) to 21:1 (black on white).

The thresholds worth memorising, from the Web Content Accessibility Guidelines:

- **4.5:1** for normal body text.
- **3:1** for large text (roughly 24px, or 18.7px bold) and for interface components and meaningful graphics, such as the border of an input box or a line on a chart.
- **7:1** if you are aiming at the stricter AAA level.

Some numbers to make this concrete. Mid-grey **#767676** on white is 4.54:1 and passes. #777777 on white is 4.48:1 and fails. That is one hex step. The habit of setting secondary text in a light grey because it looks refined usually fails by a margin that small, and the person it fails for is on a bus in the sun.

More: pure yellow **#FFFF00** on white is about 1.07:1 — effectively invisible, which is why yellow highlighter works on paper and yellow text does not work anywhere. Pure blue **#0000FF** on white is about 8.6:1 and is fine. The same blue on black is about 2.4:1 and fails badly. So "blue is readable" is not a fact about blue. Contrast is always a property of a pair.

Free ways to check:

- The WebAIM Contrast Checker, a web page that works in a phone browser.
- Chrome and Firefox developer tools show the ratio inside the colour picker when you inspect text.

Why 4.5:1 is a floor and not a goal

Your design is being read in conditions you did not test.

A phone at 30 per cent brightness in daylight loses most of the difference between a pale grey and white. Cheap panels have poorer contrast than the screen you designed on and often a colour cast. And eyes age: the lens yellows and the pupil narrows, so a sixty-year-old retina receives roughly a third of the light a twenty-year-old's does.

None of this is an argument for ugly high-contrast design. It is an argument for treating a marginal pass as a fail: at 4.6:1 you have no headroom for any of the conditions above.

Colour vision deficiency is a constraint, not a charity

Roughly one man in twelve and one woman in two hundred has some form of colour vision deficiency, most commonly a reduced ability to separate red from green. In a room of thirty, assume someone cannot read your red-versus-green chart.

The fix is not "choose friendlier colours". Blue and orange are genuinely easier to tell apart than red and green, but swapping hues only moves the problem: some deficiency defeats every hue pair, and none survive a black-and-white photocopy.

The actual rule: **never encode information in hue alone.** Add a second channel.

- A red cross and a green tick differ in shape as well as colour. Two coloured dots do not.
- Label chart lines at the end of each line instead of colour-matching them to a legend.

- Give the two states different lightness as well as hue, or add a pattern, icon or position as the second signal.

To test free: Coblis, a browser-based colour blindness simulator you upload an image to; GIMP has display filters for several deficiency types under View, Display Filters. Or the cheapest test of all — desaturate the image, and see whether the meaning survives.

Dark mode is not the same colours inverted

Pure white text on pure black is uncomfortable for many people and genuinely bad for readers with astigmatism, where bright text on a dark field blurs and smears — an effect called halation. The common fix is to back off both ends: something near **#E6E6E6** on something near **#121212**.

And check your ratios again: a colour that passed comfortably on white often fails on a dark background.

Today

Take a design you have made. Desaturate a copy and ask whether the hierarchy survives. Then pick your two most-used text colours and run them through a contrast checker against their actual backgrounds. If either comes in under 5:1, darken it until it does not.

BEFORE YOU MOVE ON

A dashboard shows each server as a small red or green dot, and both dots pass a 4.5:1 contrast check against the page background. Some users still misread the status. What is the mechanism?

A. The status is carried by hue alone, so a reader with red-green colour vision deficiency gets no signal at all; contrast against the background says nothing about telling two states apart from each other.

B. The dots are too small — contrast ratios are defined for text and need a larger area to have any effect.

C. Red and green should be replaced with blue and orange, which are safer for colour blindness.

D. Small graphics should be checked at the stricter 7:1 level rather than 4.5:1.

36 · 9 min

The lightness slider is not lightness

THE ONE THING TO KEEP

HSL lightness is a position in the RGB cube rather than a measurement of brightness, so yellow and blue at the same L differ more than tenfold in luminance — build ramps in a perceptually uniform space such as OKLCH, or measure every swatch.

Two colours, same number, nothing alike

Take `hsl(60, 100%, 50%)` and `hsl(240, 100%, 50%)`. Both claim 50% lightness. The first is a blazing yellow, the first thing you see on any page. The second is a deep blue you could set white text on.

Measure them properly and the gap is enormous. Relative luminance runs from 0 for black to 1 for white, and pure yellow comes out at about 0.93 while

pure blue is about 0.07 — more than a twelvefold difference between two colours the model says are equally light.

This is not a bug in HSL so much as a statement about what HSL was for. It is a convenient reparameterisation of RGB that makes colour pickers easy to build, and its L is a geometric position inside the RGB cube, not a measurement of how bright something looks.

Why blue is dark and yellow is bright

The eye is not equally sensitive to all wavelengths. Sensitivity peaks in the green region, around 555 nanometres in daylight vision, and falls away steeply towards both ends of the spectrum. That is why the luminance formula weights the channels so unevenly:

```
L = 0.2126·R + 0.7152·G + 0.0722·B
```

Green contributes nearly three-quarters of perceived brightness and blue about seven per cent. Yellow, which is red plus green at full strength, inherits almost all of it. Blue has almost none to inherit.

Everything downstream follows from this. It is why a yellow warning banner needs black text and a blue one needs white. It is why a naive greyscale conversion that averages the channels is wrong, as covered in the first block. And it is why a five-swatch palette that varies hue at constant HSL lightness collapses into an unusable mush of near-identical greys.

The model that does not lie: OKLCH

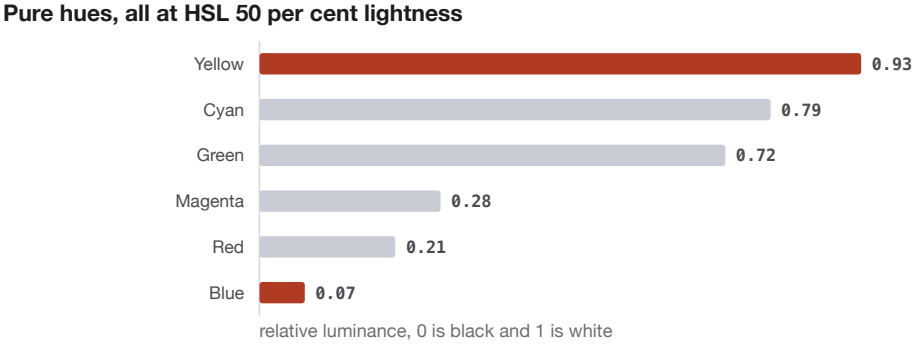
A perceptually uniform colour space is one where equal numeric steps correspond to roughly equal perceived differences. OKLCH is the one now available directly in CSS, and it has three components:

- **L** — perceptual lightness, 0 to 1. A colour at L 0.5 looks mid-grey whatever its hue.
- **C** — chroma, how colourful. 0 is grey; the practical maximum varies by hue.

- **H** — hue angle, 0 to 360.

```
:root {
  --brand-500: oklch(0.55 0.18 255);
  --brand-300: oklch(0.75 0.12 255);
  --brand-700: oklch(0.38 0.16 255);
}
```

This buys two things that are genuinely difficult otherwise.



Yellow carries more than twelve times the light of blue while the model reports both as equally light. That is why a yellow banner wants black text and a blue one white.

A ramp with even steps. Move L by 0.08 each time and every step looks like the same size of change. In HSL the steps at the pale end are visually tiny and the steps at the dark end are enormous, which is why hand-built HSL ramps always need fiddling.

Categorical colours that match in lightness. For a chart, you want six colours that differ in hue and are the same brightness, so no series looks more important than another. Hold L and C constant, vary H, and you have it. In HSL this is impossible without measuring every swatch.

Browser support for `oklch()` arrived across all the major engines in 2023, and Chrome, Firefox and Safari developer tools all show OKLCH values in their colour pickers now.

The honest limitation

OKLCH can describe colours that a screen cannot show. Ask for `oklch(0.7 0.35 145)` and you are requesting a green more saturated than sRGB contains. The browser then maps it into gamut, and different browsers have historically clipped differently — which means the colour you get may not be the colour you specified, and it may not match between devices.

Practical guardrails: keep chroma modest, around 0.1 to 0.2 for interface colours, which stays comfortably inside sRGB for most hues. Check anything vivid on a real screen. And if you need certainty, specify a fallback:

```
.button { background: #2f6df6; background: oklch(0.55 0.18
255); }
```

The second declaration wins where it is understood and is ignored where it is not.

What to take away

If you build palettes in HSL — and most tools still do — you are not obliged to abandon them. What you must do is stop trusting the L number and start measuring. Every browser's developer tools will give you a contrast ratio, which is a luminance measurement in disguise, and the greyscale check from the first block gives you the same information visually.

The rule that survives whichever model you use: **a palette is a set of lightness decisions with hues attached, not a set of hues with lightness attached.** Build the ramp first, in a space that measures lightness honestly, and choose the hue afterwards.

BEFORE YOU MOVE ON

A designer builds a six-colour chart palette in HSL, holding lightness at 55% and varying hue evenly around the wheel. In greyscale the yellow series nearly disappears and the blue one looks almost black. Why?

A. Six hues is too many for one chart, and any categorical palette beyond four will contain pairs that converge in greyscale.
B. The saturation was left at 100%, and fully saturated colours always diverge in luminance regardless of the lightness setting.
C. HSL lightness is a geometric position in the RGB cube, so equal L values produce wildly different luminance across hues.
D. Greyscale conversion applies a gamma curve that HSL does not account for, so the ranking of the swatches changes when the colour is removed.

37 · 9 min

What a contrast checker is actually computing

THE ONE THING TO KEEP

A contrast ratio is the linearised, sensitivity-weighted luminance of two colours compared with a flare constant added to both, capped at 21:1 — and the 4.5:1 threshold is a conformance floor rather than a guarantee that text is comfortable to read.

The number, and where it comes from

Every contrast checker you have used reports a ratio between 1:1 and 21:1. Here is the whole computation, so it stops being magic.

Step one: linearise each channel. Screen values are gamma-encoded, so 128 is not half the light of 255. Convert each of R, G and B from 0-255 to 0-1, then:

```
c_lin = c / 12.92                    if c ≤ 0.03928
c_lin = ((c + 0.055) / 1.055) ^ 2.4   otherwise
```

Step two: weight and sum, using the sensitivity weights from the previous lesson:

```
L = 0.2126·R_lin + 0.7152·G_lin + 0.0722·B_lin
```

Step three: take the ratio, with a constant added to both sides:

```
ratio = (L_lighter + 0.05) / (L_darker + 0.05)
```

Black against white gives `(1 + 0.05) / (0 + 0.05) = 21`. That is the maximum, and it is why no checker ever reports more.

The 0.05 is not decoration. It models ambient light reflecting off the screen — flare — which never lets a real black be truly black in a real room. Without it, black on black would compute as an infinite ratio.

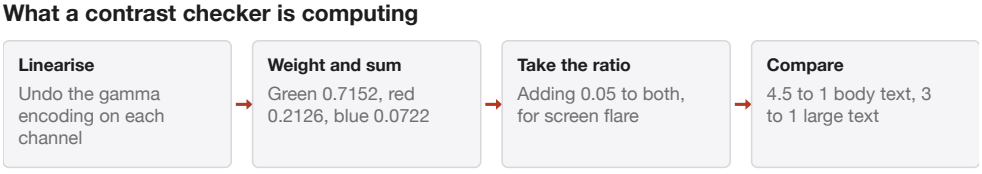
The thresholds

The Web Content Accessibility Guidelines set these, and they are the numbers that appear in law in many countries:

- **1.4.3 Contrast (Minimum), level AA** — 4.5:1 for normal text, 3:1 for large text.
- **1.4.6 Contrast (Enhanced), level AAA** — 7:1 normal, 4.5:1 large.
- **1.4.11 Non-text Contrast, level AA** — 3:1 for user interface components and meaningful graphics: a button's border, a form field's outline, an icon carrying information, the bars of a chart.

Large text has a precise definition: at least 18 point, or 14 point bold. In CSS pixels at the usual 96dpi assumption, that is 24px, or 18.66px bold. Not 18px. The three-quarters of a pixel matters if you are claiming conformance.

Two numbers worth memorising because they come up constantly:



The flare constant models ambient light bouncing off the glass. It is why black on black computes as 1 to 1 rather than infinite, and why the scale stops at 21 to 1.

- **#767676 on white** is 4.54:1 — the lightest neutral grey that passes AA for body text on a white background.
- **#949494 on white** is 3:1 — the lightest that passes for large text and for interface components.

Any grey lighter than #767676 for body text on white fails, whatever it looks like on your monitor.

Where to check, free

- **WebAIM Contrast Checker** in a browser, which also tells you which criteria pass.
- **Chrome and Firefox developer tools**: open the colour picker on any text element and the ratio is displayed, with a line on the picker showing where the passing region begins.
- **Firefox's Accessibility inspector** will audit a whole page for contrast issues in one pass.
- **Chrome's Lighthouse** audit reports failures across a page, and runs from the same panel.

None of this requires a paid tool or a plugin.

Two traps

Opacity. Text at `rgba(0,0,0,0.6)` has no contrast ratio of its own; it depends entirely on what is behind it. Change the background from white to a pale grey card and the ratio drops without anything in the text changing. Checkers cannot evaluate semi-transparent text without knowing the composite, and many people check the declared colour rather than the rendered one. Compute the blended value, or set solid colours.

Text over images. There is no single ratio, because the background varies across the image. The criterion still applies, and the usual solutions are a solid scrim behind the text, a gradient overlay, or placing the text in a plain region. Check the worst pixel, not the average.

Working the formula once, by hand

Do it once and you will never be mystified by a checker again. Take mid-grey `#808080` on white.

Each channel is 128/255 = 0.502. That is above 0.03928, so linearise with the power form:

```
((0.502 + 0.055) / 1.055) ^ 2.4
= (0.528) ^ 2.4
= 0.2158
```

All three channels are equal, so the weights sum to 1 and the luminance is 0.2158. White has a luminance of 1. The ratio is therefore

```
(1 + 0.05) / (0.2158 + 0.05) = 1.05 / 0.2658 = 3.95
```

So `#808080` on white is about 3.95:1 — it passes for large text and fails for body text, which surprises almost everybody the first time. The grey that looks like a perfectly reasonable secondary text colour is not one.

Notice what the power of 2.4 does: it pushes mid-range values down hard. A colour that is numerically halfway between black and white carries only about a fifth of white's light. That single non-linearity is why intuition about contrast is unreliable and why the check is worth running rather than estimating.

What passing does and does not mean

Passing 4.5:1 is a floor for a conformance requirement, not a claim that the text is comfortable. A 12px light-weight face at exactly 4.5:1 satisfies the criterion and is genuinely hard to read; a 17px semibold at 7:1 is easy. The criterion cannot see the difference, and the next lesson is about exactly that gap.

For most interface text the practical target is comfortably above the minimum — 7:1 or better for body text costs you nothing and removes an entire class of complaint.

BEFORE YOU MOVE ON

A team checks its body text colour of rgba(0,0,0,0.62) against a white page background, gets 5.7:1, and ships. Users on the settings page, where content sits on pale grey cards, report the text is hard to read. What went wrong?

A. Semi-transparent text has no fixed ratio, so the value measured against white does not hold once the same text sits on a grey card.

B. The checker measured the declared colour rather than the rendered one, and browsers apply a subtle antialiasing that lowers effective contrast on non-white backgrounds.

C. 5.7:1 passes AA but not AAA, so the text was always below the level needed for comfortable body reading on any background.

D. Pale grey cards reduce the flare constant in the ratio formula, because less ambient light is reflected from a grey surface than from a white one.

38 · 9 min

Where the contrast standard is wrong, and what is unsettled

THE ONE THING TO KEEP

The WCAG 2 contrast formula ignores size, weight and polarity and misscores very dark and very light pairs, so meet it as the enforceable floor while adding headroom for small or light text and checking dark mode separately — APCA addresses these gaps but is a draft, not a requirement.

A standard worth following and worth understanding

The WCAG 2 contrast formula is the basis of accessibility law in a great many countries. You should meet it. You should also know its limits, because a designer who believes the number is the whole truth will ship text that conforms and is still hard to read — and will argue about it from a position that cannot be defended.

There are four known problems.

1. It cannot see the typeface

The formula takes two colours. It knows nothing about size, weight, letterform, or how much of the glyph is actually ink.

So a 12px Light face and a 17px Semibold face at identical colours produce an identical ratio, and one of them is comfortable while the other is not. The large-text carve-out at 24px or 18.66px bold is the standard's only acknowledgement of this, and it is a single crude step rather than a curve.

The practical response: treat 4.5:1 as the floor for body text at 16px regular, and require more headroom as the text gets smaller or lighter. A 12px caption at 4.5:1 is technically conformant and practically poor.

2. It treats light-on-dark and dark-on-light as equivalent

Black on white and white on black compute to the same 21:1. They do not look the same.

Light text on a dark background suffers from **halation**: the bright glyph appears to bleed outwards into the dark field, thickening the strokes and filling the counters. It is worse for readers with astigmatism, which is a substantial share of adults, and worse again on OLED screens where the pixels beside a lit glyph are genuinely off.

WCAG 2 has no term for polarity at all. Two identical ratios, two different reading experiences, and the standard cannot distinguish them.

3. It overrates very dark pairs

The 0.05 flare constant assumes a fixed amount of ambient light reflecting from the screen. That assumption dominates at the dark end.

Take `#000000` against `#333333`. The computed ratio is about 3.7:1, which passes for large text. Look at it in a bright room and the two are nearly indistinguishable — the real flare in daylight is far higher than the constant assumes, and it swamps the difference. Dark-on-dark pairs consistently score better than they read.

4. It overrates very light pairs in the other direction

The same constant has the opposite effect near white: pairs of pale colours can be visually distinguishable while scoring poorly, which is why designers sometimes find the standard forbids a combination that works perfectly well in practice for non-text elements.

APCA, and why this is not resolved

The Accessible Perceptual Contrast Algorithm was developed to address exactly these failures. It is polarity-aware, it accounts for font size and weight, and it returns a lightness contrast value on a different scale rather than a ratio.

Here is the honest state of it, as of now:

- It has been under consideration for **WCAG 3**, which is a working draft with no adoption date and which will not replace WCAG 2 quickly even after it is published.
- Legal requirements in the EU, the UK, the US, India and elsewhere currently reference WCAG 2.0 or 2.1. Conformance means WCAG 2.
- APCA's own thresholds have been revised during development, and tools implementing it have disagreed with each other at various points.
- Some accessibility practitioners argue it is a clear improvement; others have raised concerns about complexity and about whether it is better for all users or only on average.

This is genuinely contested territory, being worked out in public by people who disagree in good faith. Anybody who tells you that WCAG 2 is simply wrong, or that APCA is simply the answer, is overstating a live disagreement.

What to actually do

1. **Meet WCAG 2 AA.** It is the requirement, and it catches the great majority of real failures.
2. **Exceed it for small or light text.** Headroom is free.
3. **Look at the result on a real device**, including in bright light and in dark mode, and believe your eyes when they disagree with a passing number.
4. **For dark mode, do not assume a passing light-mode pair inverts.** Recheck, and expect to need a slightly lighter font weight to compensate for halation.
5. **Use APCA as a second opinion**, not as a conformance claim.

> *The number is a floor that can be enforced. It is not a ceiling and it is not a measurement of comfort. Both statements are true at once, and holding both is the difference between following a standard and understanding one.*

BEFORE YOU MOVE ON

A dark-mode theme inverts a light-mode pair that scored 8:1. The computed ratio is unchanged, but testers say the dark version looks blurry and the letters feel thicker. What is the most likely explanation?

A. The inverted pair falls foul of the flare constant, which overrates dark-on-dark combinations and is inflating the reported 8:1.
B. Dark mode requires a separate AAA threshold of 7:1, and an 8:1 pair only just clears it, leaving no headroom for small text.
C. Halation makes light glyphs bleed into a dark field, thickening strokes and filling counters, and the formula has no term for polarity.
D. The typeface's hinting is optimised for dark-on-light rendering, so the stem weights are being adjusted incorrectly when the polarity reverses.

39 · 9 min

What colour blindness actually removes

THE ONE THING TO KEEP

Common colour vision deficiency collapses red-green distinctions while leaving lightness discrimination intact, so a palette built on lightness steps is already largely robust — and a simulator showing a distinction disappearing is conclusive while one showing it surviving is not.

Not black and white

The common mental picture — that a colour-blind person sees in greyscale — describes achromatopsia, which is extremely rare. What almost everybody means by colour blindness is a shift or a loss in one of the three cone types, and it leaves most colour vision intact while collapsing specific distinctions.

The categories:

- **Deuteranomaly** — reduced green sensitivity. By far the commonest.
- **Deuteranopia** — green cones absent.
- **Protanomaly / protanopia** — reduced or absent red sensitivity. Also darkens reds noticeably.
- **Tritanomaly / tritanopia** — blue-yellow, rare, and usually acquired rather than inherited.
- **Achromatopsia** — no colour discrimination at all. Very rare.

The first four are the red-green family, and they account for the overwhelming majority.

The prevalence figures, honestly

The number usually quoted is about 8% of men and 0.5% of women, or roughly one man in twelve. That figure comes from studies of populations of northern European descent, and it is not universal: measured rates are lower among men of African and Asian descent, with published figures commonly in the 3-5% range depending on the population and the screening method.

The inheritance explains the sex difference. The genes for the red and green photopigments sit on the X chromosome, so a man with one affected copy is affected, while a woman generally needs two.

What this means practically: for a product with a hundred thousand users, several thousand of them are affected, in every market. The exact figure varies; the design consequence does not.

What converges

The distinctions that collapse are the ones between colours that differ mainly in the red-green direction and match in lightness:

- A red and a green of similar lightness — the classic status dot pair.
- Red text on a green background, or the reverse.
- Orange against yellow-green.
- Purple against blue, in some types.

- Pink against grey.

The distinction that survives everything is **lightness**. This is the payoff for building palettes by moving lightness rather than hue: a palette built that way is already largely robust, without anybody having thought about colour vision at all.

Checking it, free

Simulation is built into the tools you already have:

- **Chrome DevTools**: the Rendering panel has Emulate vision deficiencies, offering protanopia, deuteranopia, tritanopia and achromatopsia.
- **Firefox DevTools**: the Accessibility inspector includes the same simulations.
- **GIMP**: View > Display Filters > Colour Deficient Vision, which applies to the canvas while you work.
- **Coblis**, a free browser tool, for uploading a flat image.

Run the design through deuteranopia first, since it is the commonest. Then look for anything where information has disappeared rather than merely changed colour.

The honest limitation on simulators. They model dichromacy — the complete absence of a cone type — reasonably well. They model anomalous trichromacy, which is the much more common condition, only approximately, and they represent an average rather than any individual. A simulator that shows a distinction surviving is weaker evidence than one that shows it disappearing. Treat a failure as conclusive and a pass as encouraging.

Designing so it does not matter

The systematic answer is in the next lesson: never let colour be the only carrier of meaning. Beyond that, three specific habits:

What common colour vision deficiency removes

Distinctions that collapse	Distinctions that survive
A red and a green of similar lightness	Lightness, in every common type
Red text on a green ground	Shape, and an icon
Orange against yellow-green	The word itself
Purple against blue, in some types	Position in a list or a column
Pink against grey	Pattern or hatching

This is the payoff for building a palette by moving lightness. A ramp made that way is already largely robust, before anybody has thought about colour vision at all.

1. **Separate in lightness as well as hue.** If a red and a green must sit together as categories, make one clearly darker. This alone solves most cases.
2. **Avoid the red-green pair for opposed meanings** where an alternative exists. Blue and orange is a widely used substitute that survives every common deficiency, and it is why so many data visualisations use it.
3. **Use a tested categorical palette for charts.** The Okabe-Ito palette was designed for exactly this and is freely usable:

```
#000000  #E69F00  #56B4E9  #009E73
#F0E442  #0072B2  #D55E00  #CC79A7
```

Eight colours that remain distinguishable under the common deficiencies. ColorBrewer, also free, offers sequential and diverging schemes with colour-blind-safe filters built in.

One thing not to do

Do not build a separate colour-blind mode as the solution. It requires the user to know they need it, find it, and turn it on, and it leaves the default design broken for everybody who does not. Fix the default. A mode may be a useful extra; it is not a substitute.

BEFORE YOU MOVE ON

A chart uses six hues at matched lightness for six data series, and a deuteranopia simulation shows two of them merging. The team's proposed fix is to add a colour-blind mode toggle. What is the stronger objection?

A. Simulators only model dichromacy approximately, so the merge shown may not occur for real users and no change is warranted until it is confirmed with them.

B. Six categories exceeds what any categorical palette can keep distinguishable, so the chart needs restructuring rather than recolouring.

C. The matched lightness is the cause, and separating the series in lightness fixes the default for everyone without requiring anybody to find a setting.

D. A toggle changes only the rendered colours and cannot alter the underlying legend, so the chart would still be unreadable once the mode was enabled by a user who knew to look for it.

40 · 8 min

Colour is never the only signal

THE ONE THING TO KEEP

Colour must always be the second encoding of something already carried by lightness, shape, text or position, because colour fails under deficiency, greyscale, sunlight and bad screens — and the check is simply to use the product in greyscale.

One rule, applied everywhere

WCAG 1.4.1 puts it plainly: colour must not be the only visual means of conveying information, indicating an action, prompting a response or distinguishing an element.

That is an accessibility requirement, and it is also just good design. Colour fails in more circumstances than any other channel: colour vision deficiency, greyscale printing, sunlight on a phone, a cheap projector, a monochrome e-reader, a screenshot pasted into a document, a bad monitor. Anything carried by colour alone is carried by the least reliable channel you have.

The rule is not do not use colour. Colour is fast and pleasant and it should do a lot of work. The rule is that colour must always be the second encoding of something that is already encoded another way.

The common failures, and the repairs

Status dots. A green circle and a red circle of similar lightness, meaning healthy and failing. Repairs, in ascending order of robustness: make the two differ in lightness; change the shape as well, so one is a filled circle and the other a triangle or a cross; add a word.

Form validation. A field turns red. A colour-blind user sees a field that looks slightly different; a user in sunlight sees nothing. The complete version is a coloured border **and** an icon **and** a message in text next to the field. The message also has to say what is wrong — Invalid is a colour change in words.

Links in body text. Blue text among black text is a colour-only signal. This is why underlines exist, and why removing them for cleanliness is a real regression. If you remove the underline, you owe the reader something else: a border on hover is not enough, because it appears only after the reader has already found the link.

Charts with a legend. A legend maps colours to names in one place and asks the reader to hold the mapping while looking somewhere else. Direct labelling — putting the series name at the end of its own line — removes both the colour dependency and the memory task. Where a legend is unavoidable, add a second distinction: different line styles, different markers, or pattern fills on bars.

Diff views. Red and green backgrounds for removed and added lines. This is close to the worst case, and it is why good diff tools also use `+` and `−` marks in the gutter.

Required fields. A red asterisk is two signals — colour and a glyph — which is adequate. A field label that is simply red is not.

Selected state. A chip that turns blue when selected. Add a tick, a border weight change, or a filled versus outlined treatment.

The redundancy toolkit

When you need a second encoding, these are the options, roughly from cheapest to strongest:

- **Lightness** — a darker and a lighter version rather than two hues.
- **Shape or icon** — a tick and a cross, a circle and a square.
- **Text** — the word. Unambiguous, costs space, and translates.
- **Position** — always first in the list, always in the same column.
- **Pattern or texture** — hatching on a chart bar. Works in print and in greyscale.
- **Weight or size** — heavier for the selected item.

For most interface work, lightness plus a glyph solves nearly everything.

The two-second test

There is a check that catches almost all of these and takes no setup at all: **turn the design to greyscale and ask whether you can still use it.**

This is the squint test from the first block doing a second job. If a status is unreadable in grey, it is unreadable for some share of your users right now. If a chart becomes guesswork, so does the chart in the printed report somebody will circulate.

On a phone: Android Settings > Accessibility > Colour correction, or iOS Settings > Accessibility > Display and Text Size > Colour Filters > Greyscale. Leave it on for an hour of ordinary use and you will find things in your own product that no audit found.

Where the rule has limits

Colour alone is acceptable where the information is genuinely decorative, or where the colour is a redundant restatement of something already unambiguous. A brand colour on a header carries no information. A paint chart, a colour picker or a photograph of a garment is about colour, and no substitution is possible — what is required there is a text name or code alongside, which is why every clothing retailer lists the colour name next to the swatch.

That last case is the clearest illustration of the whole principle: even the products whose entire subject is colour do not rely on colour alone.

BEFORE YOU MOVE ON

A team removes underlines from in-text links for a cleaner look and adds an underline on hover. Which objection identifies the actual accessibility failure?

A. Hover styles do not exist on touch devices, so phone users never see the underline at all.

B. The link colour may not meet 3:1 against the surrounding body text, which is the ratio required between a link and adjacent text.

C. Hover arrives only after the reader has already located the link, so colour is the sole signal at the moment of finding it.

D. Underlines are required by WCAG for all links regardless of contrast, so removing them is a conformance failure with no available mitigation.

41 · 8 min

Greys are a design decision

THE ONE THING TO KEEP

Neutrals cover most of an interface, so build them as a perceptually even ramp tinted slightly towards one hue, keep every grey at the same temperature, and spread the dark end further than the arithmetic suggests because small differences compress near black.

Most of your page is grey

Count the surface area of a finished interface. Text, borders, backgrounds, dividers, disabled states, secondary labels — the overwhelming majority of the pixels are some kind of neutral, and the brand colour appears on perhaps two per cent of them.

Designers spend hours on that two per cent and pick the greys from a default palette. It is the wrong allocation of attention. The neutral ramp is the single most-used part of any palette, and it is where a design either feels coherent or feels like a template.

Pure grey looks dead

`#808080` is exactly equal parts red, green and blue, and beside any coloured element it looks lifeless, slightly dirty, and disconnected from the rest of the page.

The fix is to tint the neutrals — to pull them very slightly towards a hue, usually the brand hue or its complement. Very slightly is the operative phrase. In OKLCH terms, a chroma of 0.005 to 0.02 is the useful range: enough that the grey relates to the palette, not enough that anybody would call it blue.

```
:root {
  --grey-50:  oklch(0.98 0.004 255);
  --grey-100: oklch(0.95 0.006 255);
  --grey-300: oklch(0.85 0.008 255);
  --grey-500: oklch(0.62 0.010 255);
  --grey-700: oklch(0.44 0.012 255);
  --grey-900: oklch(0.22 0.010 255);
}
```

The hue angle stays constant and the chroma rises slightly through the middle, which is where a tint is most visible and most useful.

Warm and cool, and what they actually do

A neutral tinted towards blue reads as cool, clinical, technical. One tinted towards orange or yellow reads as warm, paper-like, softer. This is one of the few claims in colour folklore that survives contact with practice, probably because it is a comparison within a page rather than an absolute claim about a hue.

What it is not is a rule about your subject matter. A warm neutral on a financial product is not wrong; it is a decision that makes the product feel like a document rather than a terminal. Choose deliberately and stay consistent, because mixing warm and cool greys in one interface produces a specific kind of muddiness that is very hard to diagnose after the fact — the page looks slightly dirty and nobody can say which element is at fault.

The diagnostic: put every grey in the design side by side as swatches, large. Mixed temperatures become obvious immediately and are invisible when the greys are scattered across a layout.

How many steps

Nine or ten is the working standard, usually numbered 50 through 900. That gives you:

- **50-100** — page and card backgrounds.
- **200-300** — borders, dividers, disabled backgrounds.

- **400-500** — placeholder text, disabled text, icons.
- **600-700** — secondary text.
- **800-900** — primary text.

What matters is that the steps are perceptually even, which is exactly what a perceptually uniform space gives you and what an HSL ramp does not. Built in HSL, the steps at the light end are visually almost identical and the steps at the dark end jump.

The dark end needs more room

One practical asymmetry. Near black, small lightness differences are harder to see than the same differences near white — and ambient light in a real room, the flare from the contrast formula, washes out the dark end further.

So a ramp that is even by the numbers may still look compressed at the bottom in practice. In dark interfaces this matters a great deal, because three or four of your surface levels all live down there. Expect to spread the dark end further apart than the arithmetic suggests, and check it in a bright room rather than a dim one.

The borders problem

A specific, common failure: borders set from the neutral ramp at a step that looks right on a large monitor in a dark office, and which disappears entirely on a phone in daylight.

A 1px border needs more contrast than a filled area to remain visible, because it is thin. Interface components carrying information require 3:1 under WCAG 1.4.11, and a decorative divider does not — but a divider nobody can see is not doing its job either. If a border is structural, give it at least 3:1 against both surfaces it separates. If it keeps disappearing, replace it with a change of background, which is the stronger figure-ground device anyway.

BEFORE YOU MOVE ON

A dark theme uses four surface levels built from an evenly spaced lightness ramp. On a laptop in a dim room the levels are clearly distinct; on a phone outdoors three of them look identical. What explains it?

A. The ramp was built in HSL, so the steps are only nominally even and the dark end is genuinely compressed in the numbers.
B. Ambient light washes out the dark end, so differences that are even by the numbers become indistinguishable in bright conditions.
C. Phone screens have a narrower colour gamut, so the four surface values are being clipped to fewer distinct outputs.
D. Dark themes require each surface level to differ by at least 3:1 in contrast under WCAG 1.4.11, and an even ramp cannot deliver that across four levels.

42 · 8 min

Red means danger, except where it does not

THE ONE THING TO KEEP

There is no reliable universal mapping from hue to meaning — red is danger in one market and auspicious in another, and stock-market colour conventions are inverted across East Asia — so rely on conventions you establish consistently inside your own product and always back them with a non-colour signal.

The chart you have seen is folklore

Search for colour psychology and you will find a diagram assigning emotions to hues: blue is trust, green is growth, yellow is optimism, red is urgency. These charts circulate endlessly in marketing material and almost none of them cite anything.

The research that does exist is much more modest. There are measurable effects — some evidence that red affects performance on certain tasks, some on colour preference varying with what a colour is associated with in a person's environment — and the effects are small, context-dependent, and nothing like strong enough to support choose blue because your users will trust you.

What is much better supported is that **learned associations are strong and local**. A colour means what a person's environment has taught them it means, which is why the honest version of this lesson is about conventions rather than psychology.

Where the conventions genuinely are shared

A few mappings are widely held because an institution taught them across many countries:

- **Red for stop and green for go** in traffic, because the Vienna Convention on Road Signs and Signals standardised it across a large number of signatory countries.
- **Red for hot and blue for cold** on taps, thermostats and weather maps, across most of the world.
- **Yellow and black hazard striping** in industrial settings.

Even these are not universal, and the traffic case is the clearest example of a convention that feels like a fact because it was legislated.

Where the conventions diverge

A handful of concrete cases, chosen because each will bite somebody building for an international audience:

- **Red.** Danger and error in Western interfaces. In China, red is the colour of luck, celebration and prosperity. In India, red is auspicious, and it is the traditional colour of a bride's sari. Marking something red to mean broken sits awkwardly against that, and marking a celebration red is natural in one market and alarming in another.

- **White.** Weddings and purity in much of Europe. Mourning and funerals in parts of East and South Asia.
- **Green.** Growth and safety widely; also carries specific religious significance in Islam, which is worth knowing before using it decoratively in a product aimed at Muslim-majority markets.
- **Stock market direction.** Green for a rise and red for a fall in Europe, the Americas and India. Inverted in China, Japan and South Korea, where red is a rise. A financial product that ships one convention worldwide will read as reporting the opposite to a large number of users.

That last one is the most useful example in this lesson, because it is a case where an ordinary designer would never think to check and where the consequence is not offence but a factual misreading.

The position that holds up

There is no reliable universal mapping from hue to meaning. What there is:

1. **Conventions your specific audience holds**, which you find out by asking people from that audience rather than by reading a chart.
2. **Conventions you establish inside your own product**, which are strong precisely because you taught them and because they are consistent across every screen.
3. **A second encoding**, from the previous lesson, which is what makes the whole question less dangerous. If the failure state is red *and* carries a cross icon *and* says Failed, then a reader for whom red does not mean failure still gets the message.

That third point is the practical resolution. You cannot research every cultural association of every colour for every market. You can make sure that no meaning rests on the association alone.

Internal consistency beats cleverness

The most common real-world failure is not cultural at all. It is a product where red means error on one screen, a destructive action on another, a sale price on a third, and the brand colour on a fourth.

That is four meanings for one signal, and the result is that the signal carries none of them. Assign each semantic colour one job, write it down, and defend it:

```
danger  → destructive actions and error states only
warning → recoverable problems requiring attention
success → completed actions
info    → neutral system messages
brand   → identity and the single primary action
```

Five roles, no overlaps. A sale price is not danger; it needs its own treatment or none.

This is entirely within your control, it costs nothing, and it does more for comprehension than any amount of consulting colour meaning charts for the markets you are entering.

BEFORE YOU MOVE ON

A trading app built in London launches in Japan using green for price rises and red for falls. Local users report the numbers look wrong at a glance even though the figures are correct. What is the best resolution?

A. Keep the colours consistent worldwide, since an internal convention is stronger than a cultural one once users have learned the product.

B. Follow the local market convention for direction and carry the direction in an arrow and a sign as well as in colour.

C. Replace both colours with a single neutral treatment, since any colour choice will be wrong in some market and colour should not carry meaning at all.

D. Add a setting that lets each user choose their preferred colour convention, defaulting to the one used in the market where the account was opened.

43 · 9 min

Dark mode, built rather than inverted

THE ONE THING TO KEEP

A dark theme is a second design with its own colour values — a near-black surface rather than black, elevation expressed by lightness rather than shadow, lightened and desaturated brand colours, and a slightly lighter font weight to offset halation.

Inverting does not work

The cheap approach is a filter that flips every lightness value. It produces a theme where the brand colour is wrong, the photographs are negatives, the shadows are inverted into glows, and the text is pure white on pure black — which is the one combination you specifically do not want.

A dark theme is a second design with its own decisions. There are five of them and they are all small.

1. Not pure black, and not pure white

White text on `#000000` produces the strongest halation, the smearing effect from the contrast lesson where bright glyphs appear to bleed into the dark field. It is worse for readers with astigmatism and worse on OLED screens, where the pixels around a lit glyph are genuinely switched off and the transition is absolute.

The widely used baseline is a very dark grey rather than black — Material Design specifies `#121212` for the base surface — with text below full white. A common approach is white at around 87% opacity for primary text, 60% for secondary, which lands near `#DEDEDE` and `#999999` on that surface.

Those particular numbers are one team's answer rather than a law. The principle behind them is not: back off both extremes.

2. Elevation runs the other way

In a light theme, a raised surface casts a shadow and gets darker at its edges. In a dark theme a shadow is invisible, because there is nothing darker for it to be.

So elevation is expressed by lightness instead: the higher a surface sits, the lighter it is. A dialogue floats above a card, which floats above the page, and each step is a few points lighter. This is why dark themes need more neutral steps at the dark end than light themes need at the light end, and why the compression discussed in the neutrals lesson bites hardest here.

3. Saturated colours have to change

A brand blue that works on white is often close to unreadable on a dark surface, and a fully saturated colour on a dark ground appears to vibrate — an effect strong enough that two adjacent saturated colours on black can be genuinely uncomfortable to look at.

The adjustment is to **lighten and desaturate** for dark mode. A brand at `oklch(0.50 0.20 255)` on white might become `oklch(0.72 0.13 255)` on a dark surface: lighter so the contrast works, less chromatic so it stops buzzing.

This means a real dark theme has its own colour values, not a filter applied to the light ones. CSS makes maintaining two sets straightforward:

```
:root { color-scheme: light dark; }
.button {
  background: light-dark(oklch(0.50 0.20 255), oklch(0.72 0.13
255));
}
```

4. Text looks heavier on dark

Because of halation, the same font at the same weight appears bolder on a dark background. A page set in a 400 weight on white can look like 500 on dark.

The correction is to drop a weight step, or where a variable font is available, to shave the weight axis slightly:

```
@media (prefers-color-scheme: dark) {
  body { font-weight: 350; }
}
```

This is a genuinely small change that removes a widespread and rarely-diagnosed sense that the dark version looks clumsier than the light one.

5. Images and shadows need handling

Photographs on a dark surface often look too bright, and a common treatment is a slight brightness reduction. Logos and illustrations supplied as dark artwork on transparency vanish entirely, so they need a light variant. Shadows should be replaced by surface lightness, as above, or used only at very low opacity for a subtle edge.

Is dark mode better?

This is less settled than the enthusiasm suggests, and it is worth being accurate.

For people with ordinary vision in ordinary lighting, most reading studies find **dark text on a light background is read at least as well as, and often better than, the reverse** — the same halation effect is the usual explanation. The claimed benefits for eye strain and for sleep are weaker than commonly stated; screen brightness and ambient light matter more than polarity.

What is well supported is that dark themes are clearly better for some people: those with light sensitivity, some migraine sufferers, some people with certain visual conditions, and anybody reading in a genuinely dark room. Battery saving on OLED is real and modest.

So the defensible position is: offer both, respect the system setting by default with `prefers-color-scheme`, and build the dark theme properly rather than

treating it as a filter over the real one. Neither theme is the correct one. Both are, for different readers.

BEFORE YOU MOVE ON

A team ships dark mode by applying a CSS filter that inverts lightness across the whole page. Beyond the photographs appearing as negatives, what is the most significant structural problem?

A. Inverted lightness puts text at pure white on pure black, which maximises halation and is the one combination to avoid.

B. Brand colours are inverted to their opposites on the hue wheel, so the product's identity colour becomes a different hue entirely.

C. Elevation inverts, so raised surfaces become darker than the page and the depth ordering of the interface reverses.

D. The inversion applies to the whole document, so any element that was already dark, such as a code block, becomes light and breaks the theme's consistency.

---

44 · 9 min

Why your blue printed purple

THE ONE THING TO KEEP

Screens add light and presses subtract it, so saturated blues, greens and oranges simply do not exist in CMYK; soft-proof in Scribus or GIMP against the printer's profile, use rich black for solids and pure K for small text, and approve colour from a printed sample rather than a screen.

Two completely different physical processes

A screen emits light. Start with black and add red, green and blue until you reach white. This is **additive** colour, and sRGB describes it.

Paper reflects light. Start with white paper and add inks that subtract wavelengths — cyan, magenta, yellow and black — until you approach a muddy dark. This is **subtractive** colour, and CMYK describes it.

They are not two notations for the same thing. They are two different sets of achievable colours, and neither contains the other.

The gamut problem, concretely

The colours sRGB can show and CMYK inks can print overlap substantially, but sRGB reaches much further into saturated blues, greens and oranges. The specific failures are predictable:

- **Vivid blue** — the classic. A saturated screen blue such as `#0000FF` has no CMYK equivalent, and the press produces the nearest it can, which reads as purple or as a flat navy. Every designer meets this once.
- **Bright green** — screen greens are far outside the four-colour gamut and print noticeably duller.
- **Orange and vivid red** — printable but weaker than they look on screen.
- **Neon anything** — not reproducible in four-colour process at all.

The colour did not change on the way to the press. It was never printable, and the screen had no way to tell you.

Soft proofing, free

Soft proofing simulates on screen what a given press and paper will produce. It is not exact — your monitor is still emitting light — but it will show you the gamut collapse before it costs you a print run.

- **Scribus** has full colour management with ICC profiles, a gamut warning, and proper PDF/X export. It is free and open source and it is the tool to use for anything going to a printer.
- **GIMP**: View > Colour Management > Soft-Proofing, with a CMYK profile loaded and Out of Gamut Warning enabled.
- **Krita** and **Inkscape** both support ICC profiles, with Inkscape's CMYK support historically the weakest of the set.

The paid industry tools are InDesign and Illustrator, named here because job advertisements name them. Scribus produces press-ready PDF/X files, and commercial printers accept them.

You need a CMYK profile to proof against. Your printer should tell you which one they use; common defaults are `FOGRA39` for coated stock in Europe and `US Web Coated SWOP` in the United States. Free profiles are available from the ECI and Adobe.

Rich black, and the small text trap

A large area printed with 100% black ink alone looks washed out — a dark grey rather than a black — because a single ink layer cannot reach full density. Printers use a **rich black**: black plus percentages of the other inks, something like C40 M30 Y30 K100, which gives a genuinely deep black.

The critical exception: **small text must be 100% K only**. Four inks on a press are applied by four plates, and they never register perfectly. On a large area a fraction of a millimetre of misregistration is invisible; on 9-point text it produces coloured fringes around every letter and a blurred appearance.

So: rich black for large solids, pure K for text and thin rules. This single rule accounts for a large share of the difference between print work that looks professional and print work that does not.

Spot colours

Where a colour must be exact — a brand mark, a corporate identity — process printing is unreliable, because every press and paper shifts it slightly. A **spot colour** is a pre-mixed ink applied by its own plate, specified from a physical reference system, of which Pantone is the best known.

Two practical consequences. A spot colour can reach outside the CMYK gamut, so brands with a vivid identity colour often specify one. And each spot colour adds a plate, which adds cost — which is why two-colour printing, one spot plus black, is a real and economical option for stationery and simple work.

Pantone references are proprietary and the physical books are expensive. For most work you do not need one; for a brand identity that will be printed at scale, the studio or the printer will have one.

The rule that saves you

Never approve a colour from a screen for something that will be printed in quantity.

Look at a printed sample. That means a swatch book, a previous job printed on the same stock, or a proof from the actual printer. Paper changes everything — the same ink on uncoated stock is duller and spreads more than on coated — and no monitor simulates paper.

A digital proof from the printer costs a small amount and settles arguments permanently. For anything above a few hundred copies, it is the cheapest insurance available, and the one step that experienced people never skip.

BEFORE YOU MOVE ON

A poster is printed with all its text, including 9pt captions, set in a rich black of C40 M30 Y30 K100. The large headline looks excellent; the captions look fuzzy and slightly coloured at the edges. Why?

A. Four inks are laid by four plates that never register perfectly, and at small sizes that misregistration shows as coloured fringing.

B. Rich black is outside the CMYK gamut at small sizes, so the press substitutes the nearest printable value inconsistently across the sheet.

C. The total ink coverage of 200% exceeds what the paper can absorb, so the ink spreads and blurs the smaller characters.

D. The caption text was set in a high-contrast typeface whose hairlines cannot survive at 9pt, and the colour fringing is an unrelated artefact of the proofing process.

45 · 8 min

How many colours a design needs

THE ONE THING TO KEEP

One neutral ramp, one brand ramp and three or four semantic colours covers most design work, because each additional colour reduces the information carried by every existing one — data visualisation is the genuine exception, and there the palette should be a tested colour-blind-safe set.

The complete palette for most products

Here is a palette sufficient for the great majority of interfaces and printed material:

- **One neutral ramp**, nine or ten steps, slightly tinted.
- **One brand hue ramp**, five or six steps.
- **Three or four semantic colours** — danger, warning, success, and perhaps info — each with a light background variant and a dark text variant.

That is it. Two hues plus the semantic set. Almost every design that feels chaotic has more than this, and almost every design that feels coherent has this or less.

The reason is the emphasis budget from the first block. A second brand colour does not add a second accent; it halves the strength of the first. With five brand colours, nothing is an accent and the palette has become decoration.

The whole palette, for most products

One neutral ramp	Nine or ten steps, tinted very slightly towards one hue
One brand ramp	Five or six steps, covering the interactive states
Three or four semantic colours	Danger, warning, success, and perhaps info

Two hues plus the semantic set. A second brand colour does not add a second accent; it halves the strength of the first.

The 60/30/10 heuristic

A widely repeated split: 60% of the surface in a dominant neutral, 30% in a secondary, 10% in an accent.

It comes from interior design and it has no research behind it. What it is useful for is as a corrective, because the instinct of a beginner is roughly 40/40/20 and the result is a page where the accent has no force. If you are unsure, aim at 60/30/10 and you will overshoot in the right direction.

Do not treat it as a specification. Plenty of excellent work is 90/8/2, and a great deal of interface design is closer to 95/4/1 once you count the actual pixels.

Where more colours are genuinely required

Data visualisation. Categorical series need distinguishable colours, and there the constraints are different and well studied:

- Roughly **six to eight categories** is the practical maximum before colours stop being reliably distinguishable, whatever palette you use. Beyond that, group the tail into Other, or use a different encoding.
- The palette must survive colour vision deficiency, which means a tested set rather than one you assembled. Okabe-Ito's eight-colour palette and ColorBrewer's qualitative schemes are both free and both designed for this.
- **Sequential** data — a quantity increasing — wants a single hue varying in lightness, not a rainbow. The rainbow scale is actively misleading because its perceived lightness is not monotonic, which creates false boundaries in the data where the hue changes fast.
- **Diverging** data — deviation either side of a midpoint — wants two hues meeting at a light neutral. Blue to orange is the standard colour-blind-safe pair.

Category systems that users genuinely navigate by colour: a calendar with six colleagues' schedules, a transit map, a file-type system. These earn their colours because the colour is the primary index and the set is learned over time.

Illustration and photography, which are not palettes and should not be constrained by one.

Building the brand ramp

From a single brand hue, the ramp serves every state you need:

```
50   pale background for a selected row or a tinted banner
100  hover background on a light surface
300  borders and dividers in brand colour
500  the primary action, the main brand appearance
700  hover and pressed states on the primary action
900  brand-coloured text on a pale brand background
```

Six values, one hue. Notice that the interactive states come out of the ramp rather than being invented separately, which is what keeps a hover state looking related to the thing it is a state of.

The audit

Same as for spacing and type. Export every distinct colour value in the design as a row of swatches and look at them together.

What this reveals, reliably:

- Two greys that were meant to be the same.
- Three blues that are nearly identical because they came from three sources.
- A semantic red and a brand red that clash.
- A palette with fourteen values where six were intended.

The repair is mechanical: map each stray onto the nearest intended value and delete it.

The restraint argument, stated properly

Using fewer colours is not minimalism or taste. It is that **a colour only carries information if it is rare enough to mean something**. Every addi-

tional colour in a system reduces the information carried by all the others, in exactly the way every additional emphasis reduces the emphasis of the rest.

So the question to ask of any proposed new colour is not whether it looks good with the others. It is: what will this colour mean, and what is currently meaning that?

BEFORE YOU MOVE ON

A dashboard uses a rainbow colour scale to show a single continuous quantity from low to high. A reviewer objects. What is the strongest technical reason?

A. Rainbow scales use too many hues, and any sequential scale should be limited to the six to eight categories that remain distinguishable.

B. Rainbow scales are not colour-blind safe, so a tested qualitative palette such as the Okabe-Ito eight-colour set should be used in place of the continuous ramp.

C. Perceived lightness across a rainbow is not monotonic, so the eye reads boundaries where the hue turns quickly rather than where the data changes.

D. A rainbow scale has no defined midpoint, so readers cannot tell which values sit above and below the average without consulting the legend.

CASE STUDY · MODULE 4

The grey that passed in the office and failed in the sun

A health survey app used by about 240 ASHA workers across three blocks of rural Maharashtra, filling in household forms outdoors.

The app was built by a four-person team in Pune for a state-funded programme. The interface followed a brand palette produced by an agency two years earlier: a strong blue for actions, near-black for primary text, and a light grey for secondary text, hints and field labels.

The grey was documented in the brand sheet as the "supporting text" colour. Checked against white, it measured 4.62 to 1. That passes the AA requirement for body text, which is 4.5 to 1, and every audit the team ran came back green.

The complaint nobody could reproduce

Field reports had a recurring line that the programme manager could not make sense of: workers said they "could not see the small writing". Nobody could reproduce it. On the team's monitors the grey was perfectly legible, and on a phone at full brightness indoors it was fine.

A developer took a phone out to a supervisor's meeting in Ahmednagar in March, at about eleven in the morning, and opened the app in the open courtyard where most of the workers were sitting. The primary text was readable. The grey was gone. Not faint — gone, indistinguishable from the white field it sat on, at the brightness the phone had settled on after a morning outside.

Three things had stacked. Phone screens in daylight lose most of the difference between a pale grey and white. The handsets in use were inexpensive Android devices with narrower gamut and poorer contrast than the team's own phones. And the audience included a substantial number of workers over fifty, whose eyes receive materially less light than a young team's.

The grey had a headroom of 0.12 above the requirement, and every one of those three conditions ate more than that.

The decision

The repair was one line in a stylesheet. The secondary text colour moved from the brand grey to a considerably darker one at about 7.4 to 1, and the field hints, which were the worst case, moved to the primary text colour with a smaller size carrying the distinction instead.

The obstacle was governance. The brand palette had been signed off by the department's communications committee, and the grey was in a document with an approval reference on it. Changing an approved colour meant a fresh note, a committee that met monthly, and — the team was told informally — a likely request to see the whole palette again, which would reopen questions about the blue that nobody wanted reopened.

The estimate was three weeks at best and a quarter at worst.

Against that sat the field cost. The hints under the fields were where the form explained what went in them: which date, whose name, what unit. A worker who cannot read the hint fills the field from memory of the training, and the commonest errors in the incoming data matched exactly the fields whose hints carried the most information. The programme was in its baseline survey, and the baseline is the number every later comparison is made against.

The team's lead put the two costs next to each other in the note she wrote: wait for the committee and collect a quarter of a baseline survey with a known data-quality fault, or change an approved colour without approval and answer for it afterwards.

> **STOP HERE**
>
> Before turning the page: what would you have done, and what would it have cost if you were wrong? Write it down. The point of the next section is to compare.

WHAT HAPPENED

The team wrote the note, and while it was in the queue they shipped the change as an accessibility defect rather than a design change, on the argument

that a marginal pass under laboratory conditions is not a pass under field conditions and that the app had a legal obligation to be usable. The release notes said exactly that, in one sentence, which meant nobody could later claim it had been done quietly.

The committee's response, when it came six weeks later, was to accept the change and to ask that the brand sheet be amended to carry two greys: the original for marketing surfaces and a darker one specified for interfaces. That was a better outcome than the team had asked for and had not occurred to anybody before the incident.

The habit that came out of it was smaller and more useful than the colour change. The team bought one inexpensive second-hand Android handset, kept it on the desk, and made a rule that anything before release is looked at on that phone, outdoors, once. The first month it found two more contrast failures, a tap target under a system gesture bar, and a loading spinner that was invisible against the sky.

The grey passed 4.62 to 1 and was still unreadable in the field. Does that mean the contrast standard is wrong?

No. It means the threshold is a conformance floor rather than a statement that the text is comfortable. The formula takes two colours and knows nothing about ambient light, screen quality, the size and weight of the text, or the reader's age. A pass with 0.12 of headroom leaves nothing for conditions the check does not model, which is why a marginal pass should be treated as a fail and small or light text given headroom well above the minimum.

Why were the field hints the worst case rather than merely one more instance?

The hints were the only place that explained what each field wanted, so the colour was the sole carrier of information that had no second encoding anywhere. Unreadable secondary text elsewhere costs polish; an unreadable hint costs the datum. The repair reflected that by moving hints to the primary text colour and carrying the visual distinction with size instead, rather than simply darkening the grey a little.

What did the cheap handset on the desk buy that an audit tool did not?

Automated checks evaluate declared colours against declared backgrounds under ideal assumptions, and they found nothing here because the pair technically passed. A real device outdoors tests the conditions the formula excludes: screen quality, brightness under sunlight, glare, and the physical size of targets under a thumb. In the first month it found three classes of fault, none of which any checker reports.

MODULE 4 · TEST

Check what stuck

6 questions, one right answer each. This one is for you, not for a record: answers and the reason behind each are at the back. If two questions from the same module go wrong, re-read that module before the exam rather than after.

1 Pure yellow and pure blue are both written as hsl(h, 100%, 50%). Why can one carry white text and the other cannot?

A. Contrast checkers weight warm hues lower than cool ones
B. Saturation at 100 per cent always reduces contrast
C. Their relative luminance differs by more than tenfold
D. Yellow occupies a smaller region of sRGB than blue does

2 The contrast formula adds 0.05 to both luminances before dividing. What is that constant for?

A. It compensates for the gamma encoding of the channels
B. It fixes the 4.5 to 1 threshold used for body text
C. It prevents a division by zero in the implementation
D. It models ambient light reflecting off the screen

3 A heading is set at 18 pixels bold and the designer claims the 3 to 1 large-text threshold. Is that right?

A. No, because large text means 24px, or 18.66px when bold
B. Yes, because bold strokes carry more ink and therefore more contrast
C. Yes, because any heading counts as large text
D. No, because bold text of any size must meet 4.5 to 1

4 A chart distinguishes six series by colour and maps them in a legend. What is the most robust repair?

A. Choose a tested colour-blind-safe palette such as Okabe-Ito
B. Label each line at its own end
C. Raise the saturation of every series
D. Add a pattern fill behind each legend swatch

5 In a dark theme, why is a raised surface made lighter rather than given a shadow?

A. Because OLED screens cannot render soft shadows at low opacity
B. Because dark themes use a different shadow colour
C. Because there is nothing darker for a shadow to be
D. Because shadows are disabled by prefers-color-scheme

6 A saturated screen blue came back from the press as a flat purple. What happened?

A. The monitor was uncalibrated, so the blue on screen was already purple
B. Rich black under the artwork pulled the blue towards magenta
C. The file was exported in RGB rather than CMYK, so the press had to guess at it
D. That blue was never printable, so the press produced the nearest it could

MODULE 5

Composition and the image

Most design work includes a picture, and most pictures arrive as somebody else's rectangle. This block is about what you do with them: where the crop should fall and what it removes, why a tangent is uncomfortable, how to build depth on a flat surface, how to keep text legible over an image you do not control, and how to make one asset survive being re-cropped by four different platforms.

> BY THE END OF THIS MODULE YOU CAN
>
> Crop and place an image so that the subject, the four frame edges, the empty space and any text over it all support one reading, justify the crop by what it removes, and produce a single asset that survives re-cropping to square, portrait and vertical without losing its subject

---

46 · 9 min

The rule of thirds is a nudge, not a law

> THE ONE THING TO KEEP
>
> Decide first whether the picture wants balance or tension, then check the four edges — the thirds grid is only useful for stopping you centring a horizon.

What composition is actually deciding

Beginners treat composition as the question of where to put the subject. It is simpler than that: composition is the decision about what is inside the frame and what is not. Everything else is downstream of that.

Which is why the most powerful compositional move is also the cheapest. Crop. It needs no equipment, works on a phone, and changes the meaning of a picture rather than its size.

The rule of thirds, honestly assessed

The idea comes from a 1797 book of landscape-painting advice by John Thomas Smith, who suggested dividing a picture into thirds and putting the horizon on one of the lines rather than the middle. It now appears as a grid overlay in every camera app.

What it genuinely buys you: it stops you putting the horizon dead centre. A horizon in the middle splits the frame into two halves that compete and neither wins, which produces the flat landscape everyone has taken. Nudging it up or down decides which half the picture is about — worth the grid overlay on its own.

What it does not buy you: there is no perceptual evidence that the four intersections are special "power points". They are a by-product of dividing by three, not a discovery about vision. Related claims about the golden ratio are mostly retrofitted: the analysis is drawn on top of the painting afterwards, and you can draw it on almost anything.

Where it becomes cargo cult: switching on the grid and shoving every subject to an intersection regardless of what the picture is doing. Symmetry is a strategy, not a failure. Passport photos, most logos, a temple façade and a portrait meeting the lens straight on are dead centre on purpose: centre means stillness and confrontation.

The better version of the rule: decide whether this picture is about **balance** or **tension**. Balance goes centre. Tension goes off-centre. Then commit, because a subject one-fifth off-centre reads as a mistake in either direction.

The edge of the frame is an active element

The frame is not a neutral container: where it cuts changes what the viewer sees.

- **Cut a limb at a joint** — wrist, elbow, ankle — and it reads as amputation. Cut mid-forearm or mid-shin and it reads as continuing out of the picture. Same with a building: cut at a corner it looks truncated, cut across a wall it carries on.
- **Tangents** are the quiet killer. An edge that just touches another edge welds two unrelated things into one shape: a lamp post growing out of a head, a horizon line meeting a shoulder exactly. Move it clearly apart or clearly overlap. Almost-touching is the only bad option.
- **Check all four edges before you export.** Ten seconds, and it catches most of what makes an image look accidental.

Lead room, and why gaze steers attention

Give a subject space in the direction they are moving or looking. A person walking towards the left edge with the empty space behind them looks trapped. A face looking out of the near edge sends the viewer's attention straight out of the picture, because attention follows gaze — one of the most automatic things human vision does.

This is also why a thirds crop can make a portrait worse: push the face to the right-hand line while it is looking right, and the rule has removed the space the picture needed.

Lines, space and cropping

Leading lines are real but weaker than photography blogs suggest, and they only work if the line **arrives somewhere**. A road receding into empty sky is a line to nothing. What a line provides is continuity, and continuity is a grouping cue: it says the thing at this end and the thing at that end belong together. Use it to connect where the eye enters to the thing you want it to reach.

Negative space is the other half, and the trick is to look at it directly rather than through the subject. A free way to force this: open the image in Photopea, GIMP or Krita, select the subject and fill it flat black. What remains is the shape you actually composed. A scatter of thin slivers means the picture is cluttered however good the subject is; two or three clean shapes means it is working.

Cropping is a claim, not a resize

The same photograph cropped tight on a face says intimacy and pressure — you are close and cannot see what else is happening. Cropped wide, it says context, scale, and often isolation. Neither is more truthful. Both are choices, and pretending a crop is neutral is how photographs say things the moment did not.

The practical version is aspect ratio: a composition does not survive reframing. A wide 16:9 image cropped square for a feed and then to 9:16 loses everything at the sides, which is usually where the context was. Either design the crops you will actually need, or leave empty margin the reframe can eat. More on cutting for different frames in video editing and visual storytelling.

Two things that are not composition questions but arrive at the same moment. If a recognisable person is in your frame, their permission to use the image is a separate matter from whether it is well composed, and it is not a style question. And if any part of the image was generated rather than photographed, it needs a label — several countries, India among them, now require visible labelling of synthetic media, and the requirement is about what a viewer can see, not what sits in the file.

Today

Take one photograph you like and make three crops: one tight, one wide, one that changes which part of the scene is the subject. Look at them side by side and write one sentence for each about what it now says. You will use this more than any grid overlay.

BEFORE YOU MOVE ON

A photographer re-crops a portrait so the subject sits on a rule-of-thirds intersection, looking towards the nearer edge of the frame. It feels worse than the centred original. What is the most likely reason?

A. The rule of thirds applies to landscapes rather than portraits, so it should not have been used here.
B. Cropping reduced the resolution, so the image now reads as lower quality.
C. The subject's gaze now runs straight out of the frame with no space in front of it, so attention leaves the picture; the grid does not override the need for lead room.
D. Centred compositions are always stronger for faces, because faces are symmetrical.

47 · 8 min

Why a near-touch is worse than an overlap

THE ONE THING TO KEEP

A tangent removes the occlusion information the visual system uses to order depth, so two contours that nearly touch fuse into one ambiguous shape — either overlap them clearly or separate them clearly, and never crop a figure at a joint.

The lamp post growing out of the head

Everybody has seen the photograph where a pole rises from behind someone and appears to sprout from their skull. It is funny, and it is an instance of a general fault that also produces effects nobody finds funny and nobody can name.

A **tangent** is where two edges just touch, or barely overlap, or come within a hair of each other. The specific kinds:

- A contour meeting the frame edge exactly, so the subject appears glued to the border.
- A horizon line passing through a subject at a point where it looks attached.
- Two objects whose outlines kiss without clearly overlapping.
- A figure cropped exactly at a joint — the ankle, the wrist, the knee, the neck.
- A corner of one rectangle landing on the corner of another.

The mechanism

Occlusion — one thing covering part of another — is the strongest depth cue the visual system has. It is unambiguous and it works at any distance: if A hides part of B, A is in front. Almost everything you know about the spatial arrangement of a scene starts there.

A tangent removes exactly that information. When two contours meet without clear overlap, there is no evidence about which is in front. The visual system has to resolve an ambiguity with nothing to resolve it from, and two things happen. The depth reading becomes unstable, which is mildly unpleasant in a way people describe as the image looking flat or odd. And because the contours are continuous, the gestalt principle of continuity fuses them into a single shape — which is precisely why the pole and the head become one object.

This is the near-miss problem from the first block, in another domain. Clear overlap is fine. Clear separation is fine. The narrow band between them is the fault.

Cropping at joints

The joint case deserves its own note because it comes up in every portrait and every product shot.

Cropping a figure at the ankle, wrist, elbow, knee or neck reads as amputation. Cropping mid-calf, mid-forearm or mid-thigh does not. The usual explanation is that the visual system segments a body at its joints, so a cut at a joint coincides with a boundary the brain was already treating as a division and reads as a removal, while a cut through a limb's middle reads as the frame ending.

Whatever the mechanism, the convention is old, consistent across photography and painting, and reliable. Crop through the middle of a segment, never at the hinge.

Overlap, separate, or the narrow band in between

Reads correctly	Reads as a fault
One contour clearly covering another	Two outlines that just touch
A gap wider than the smaller stroke	A three-pixel clearance
A figure cropped mid-forearm or mid-shin	A figure cropped at wrist, ankle or neck
Two edges aligned exactly, on purpose	A shape three pixels off an exact line

A tangent removes the occlusion that tells the eye which object is in front, and continuity then fuses the two contours into a single shape.

Fixing it

The repair is almost always a small movement, which is what makes tangents worth learning to spot — they are cheap to fix and invisible until they are pointed out.

- **In a photograph you are taking**: move the camera 30 centimetres left. That is usually the whole fix.
- **In a photograph you are given**: crop slightly differently, or reposition the image within its frame so the contour clears the edge by a visible margin.
- **In a layout**: either pull the elements apart until there is real space, or push them together until there is real overlap. Do not settle at a two-pixel gap.

The amount matters. A three-pixel clearance is still a tangent. Aim for a gap that is obviously a gap — at least the width of the smaller element's stroke, and usually much more.

The inspection routine

After a composition is otherwise finished, spend thirty seconds on this:

1. **Look at each of the four edges in turn**, along its whole length, ignoring the middle of the image. This is unnatural and that is the point — the

centre is where you have been looking all along.

2. **Find every place where two outlines come close** and ask, for each, whether you can tell which is in front.
3. **Check every crop of a figure** against the joints.

Doing this deliberately, as a separate pass, catches far more than looking at the image as a whole, because as a whole you see the subject and the tangents live at the periphery.

Where a tangent is a tool

Deliberate tangency is a real device. Two shapes that align exactly along an edge can read as one continuous form, which is how a great deal of logo construction works and how a layout can imply a shape that is not drawn. A subject cropped hard at the frame edge, decisively rather than marginally, reads as energetic and is a standard editorial treatment.

The difference is commitment, and it is the same distinction as between a balanced composition and one that is 5% off. Exact alignment is a decision. Three pixels away from exact alignment is a mistake. There is nothing useful in between.

BEFORE YOU MOVE ON

A product photograph shows a bottle whose left edge sits two pixels from the frame edge. A reviewer says it looks stuck to the border. The designer moves it to four pixels. It still looks wrong. What is the principle being missed?

A. The bottle should be cropped by the frame edge instead, since a deliberate crop reads as energetic and avoids the question entirely.

B. Product photographs require a minimum margin proportional to the subject's height, and four pixels is far below any usable proportion.

C. Any gap below the perceptual threshold still reads as contact, so the clearance has to be obviously a gap rather than merely non-zero.

D. Frame edges create tangents only with curved contours, so the fix is to rotate the bottle slightly so its edge is no longer parallel to the border.

48 · 8 min

A crop is a decision about what leaves

THE ONE THING TO KEEP

A crop is a decision about what to remove rather than what to keep, it cannot change perspective because that was fixed at capture, and it spends resolution permanently — so crop last, for a named output, and never over the original.

Ask the other question

When people crop, they ask what should be in the frame. The more useful question is what is being removed, because that is the decision that changes the picture's meaning.

A wide photograph of a vegetable market is about a market. Crop to two hands and a note passing between them and it is about a transaction. Crop to one vendor's face and it is about a person. The same file, three subjects, and the difference was made entirely by subtraction.

This reframing is not wordplay. It changes what you look at while cropping — from the subject, which you were already looking at, to the context, which is where the decision actually lives.

Three passes

A habit that produces better crops than adjusting one rectangle until it looks right:

1. **Crop for the subject.** The tightest frame that still contains what the picture is about.
2. **Crop for the context.** The widest frame that adds information rather than clutter.
3. **Choose between them deliberately**, and be able to say what the middle option gains or loses against each.

Most weak crops are the accidental middle: too tight to explain anything, too loose to be about anything.

What a crop cannot do

Cropping changes framing. It does not change perspective.

Perspective is fixed at the moment of capture by where the camera was. A portrait taken from twenty centimetres away has a distorted nose because of the distance, and cropping the same file cannot produce the portrait you would have got from two metres away with a longer lens. Likewise a building photographed from below has converging verticals, and cropping does not straighten them — that needs a perspective correction, which is available free in GIMP's Perspective tool and in Krita, and which costs resolution and introduces stretching.

So when a photograph is wrong because the camera was in the wrong place, cropping is not the repair. Reshoot if you can, and if you cannot, choose a crop that makes the distortion look intentional rather than one that half-hides it.

The resolution budget

Cropping throws pixels away permanently, and it is worth being able to do the arithmetic rather than discovering the problem at the printer.

For print, the working figure is **300 pixels per inch**. So:

```
print size in inches = pixels ÷ 300
```

A 12-megapixel phone photograph is roughly 4000 × 3000 pixels, which prints at 13.3 × 10 inches. Crop to a quarter of the area and you have 2000 × 1500, which prints at 6.7 × 5 inches. Crop to a tenth of the area and you are at roughly 1265 × 950, which is a 4-inch print and nothing larger.

For screens the budget is far looser. A full-width hero image on a phone needs around 1200 pixels wide to look sharp on a high-density display; on a desktop, around 2400. A crop that is useless for a poster is often perfectly adequate for

a web page, which is why the same asset library can serve one and not the other.

Upscaling, including with machine-learning upscalers, invents detail that was not recorded. It can look good and it is not the same as having the pixels. For anything where accuracy matters — a product, a document, a face that somebody will recognise — treat invented detail as a risk rather than a solution.

Keep the original

Crop last, for a specific output, and never over the master file. This sounds obvious and it is the most commonly broken rule in practice, usually because somebody cropped in a phone gallery app that overwrites.

The workflow that survives:

- Keep originals untouched in one place.
- Produce crops as exports, named for their purpose — `hero-16x9.jpg`, `card-4x5.jpg`, `thumb-1x1.jpg`.
- In GIMP, Photopea or Krita, do the crop on a layer mask or as a canvas-size change in a saved working file, so the original pixels remain in the document.

You will be asked for a different crop. You will always be asked for a different crop.

The crop as a composition decision

One last point that connects this to the rest of the block. When you crop, you are not only choosing content — you are setting the aspect ratio, deciding where the subject falls relative to the edges, and creating the negative space. All three are composition decisions being made by the same rectangle.

That is why cropping well is a skill rather than a trim, and why a photograph that looked ordinary can become good with one decisive crop. What you removed was doing damage.

BEFORE YOU MOVE ON

A 12-megapixel photograph, roughly 4000 by 3000 pixels, needs to fill an A5 flyer at roughly 6 by 8 inches at 300dpi. How much of the original area can be cropped away?

A. None — 4000 by 3000 is already below the 1800 by 2400 pixels required, so the image is unusable at that size without upscaling.
B. Roughly half, since the flyer needs about 1800 by 2400 pixels and the original supplies 4000 by 3000.
C. Almost all of it, because 300dpi applies only to line art and photographs remain acceptable down to 150dpi at any size.
D. About three-quarters, because the flyer's area in square inches is roughly a quarter of what the file could print at 300dpi across its full dimensions.

49 · 8 min

Building a path through the image

THE ONE THING TO KEEP

The eye follows the smoothest available path, and gaze direction, rows of objects and value edges create lines the picture never drew — so give a facing subject space to look into, specify the intended path before checking it, and never leave a strong line ending at nothing.

Lines the picture does not contain

A road, a fence, a railing or a shadow gives the eye a track to follow. Those are real lines and they are easy to spot. The more useful ones are implied, and they are everywhere:

- **Gaze direction.** Where a person in the image is looking. The strongest implied line available, and the one from the faces lesson in the first block.

- **A row of similar objects.** Three windows in a line read as a line, by continuity.
- **A pointing hand, an outstretched arm, the direction a vehicle faces.**
- **The direction of motion**, including implied motion in a still image.
- **An edge between two areas of different value**, which is a line whether or not anything drew it.

The mechanism is continuity from the grouping block: the eye follows the smoothest available path. Give it a path and it takes it. Give it several crossing paths and it stalls.

Lead room

A subject looking or moving towards the frame edge, with the edge close behind their gaze, feels compressed and slightly wrong. Give them space in the direction they are facing — traditionally called lead room or nose room in photography and film — and the composition relaxes.

This is not decoration. The gaze creates an implied line, that line has to go somewhere, and if it hits the edge immediately the eye is ejected from the image. Space in front of the subject lets the implied line resolve inside the frame.

The practical version: when a subject faces right, place them left of centre, not right. It is the opposite of what an untrained crop tends to do, because the instinct is to centre the face and leave the space behind.

Designing the path on purpose

For a composed piece — a poster, a slide, an advertisement — the eye path is something you specify rather than discover:

```
entry     → where the eye lands first
primary   → the message
secondary → the supporting detail
exit      → what you want them to do
```

Then check whether the composition supports that order. The honest way to check is to sketch the intended path as an arrow over a print or a screenshot, then show the piece to somebody for three seconds and ask what they saw first, second, third. Where the answer disagrees with the arrow, the composition is telling a different story from the one you meant.

The commonest disagreement: the exit is not on the path at all. The call to action sits below everything, outside the flow, reached only by a reader who has decided to keep going.

Diagonals, horizontals and verticals

The conventional associations — diagonals suggest movement, horizontals calm, verticals formality — are worth knowing and worth holding loosely.

What is mechanically true is narrower and more useful: a diagonal crosses more of the frame than a horizontal or vertical of the same length, so it moves the eye further and connects more regions. It also sits at odds with the rectangular frame and with almost every other element in a typical layout, so it has contrast of form working for it, which is the shape channel from the first block.

The associative claims beyond that are conventions rather than perceptual facts, and they are strongest where a body of work has taught them. Treat them as available conventions, not as forces.

What ruins a path

Three things, in order of frequency:

Competing paths. Two strong lines crossing at a shallow angle leave the eye without a clear route. One should dominate.

A path leading out of the frame. A line running to a corner takes the eye with it. Corners are exits, which is why a strong diagonal running exactly into a corner is usually a fault and why compositions often deflect a line before it reaches one.

A path with nothing at the end. A line that draws the eye to an empty area, a crop edge, or a piece of background is a promise that is not kept. If a line is strong, something has to be waiting where it arrives.

The check

Blur the image heavily, as in the first block, and trace where the masses lead. Lines survive blurring surprisingly well, because a line is a large-scale luminance structure rather than a fine detail. What you see in the blurred version is close to what somebody gets in the first half-second, before they have looked at anything deliberately.

If the blurred version has no path in it, the image will feel static no matter how good the subject is.

BEFORE YOU MOVE ON

A poster places a photograph of someone looking sharply to the right at the right-hand edge of the layout, with the headline and call to action on the left. Testers report the poster feels uncomfortable and they miss the call to action. What is the mechanism?

A. The headline is competing with the face for first fixation, so the call to action is reached third and gets skipped by readers who stop after two elements.

B. The gaze creates an implied line that exits the frame immediately, so the eye is ejected rather than led towards the text.

C. Right-to-left eye movement is unnatural for readers of English, so the layout is asking for a reading order the audience cannot perform.

D. The photograph sits at the frame edge, which creates a tangent between the subject's contour and the border and destabilises the depth reading of the whole composition.

50 · 8 min

Six ways to make a flat surface have depth

THE ONE THING TO KEEP

Occlusion is by far the strongest depth cue and costs nothing, aerial perspective — less contrast, less saturation, closer to the background lightness — is the one you can apply deliberately anywhere, and shadows work only when every shadow in a design agrees about one light direction.

The cues, in order of strength

A screen and a sheet of paper are flat. Depth in both is a set of cues the visual system reads, and they are not equally powerful. Knowing the order tells you which one to reach for and why the weak ones keep failing.

1. Occlusion. One thing covering part of another. Unambiguous, works at any distance, and stronger than every other cue combined — if occlusion says A is in front, nothing overrides it. In flat design this is the most useful cue by a wide margin and the most underused: one card overlapping another establishes order instantly, with no shadow required.

2. Relative size. Two things known to be the same size, one smaller, so it is further away. Depends on the viewer knowing the real sizes.

3. Height in the frame. Objects further away sit higher in the visual field, up to the horizon. This is why a small object placed high reads as distant and the same object placed low reads as small.

4. Texture gradient. A repeating texture — cobbles, tiles, a crowd — compresses with distance. Useful in photographs, rarely constructible in layout.

5. Aerial perspective. Distant things are lower in contrast, less saturated, lighter, and slightly blue, because you are looking through more atmosphere. This is the one you can apply deliberately to any flat artwork and it is enor-

mously effective: push background elements towards the background colour and pull contrast out of them.

6. Blur. A shallow depth of field separates a subject from its background. Real in photographs; simulated by `filter: blur()` in CSS or a Gaussian blur in GIMP or Krita.

Shadow appears nowhere in this list as a primary cue, and that is deliberate. A shadow is a depth cue only if the light is consistent, which brings us to the failure.

Depth cues, strongest first

Occlusion	One thing covering another. Unambiguous, free, and stronger than the rest combined.
Relative size	Needs the viewer to know the real sizes
Height in the frame	Further things sit higher, up to the horizon
Texture gradient	A repeating texture compressing with distance
Aerial perspective	Less contrast, less saturation, nearer the background lightness
Blur	Shallow depth of field, photographed or simulated

Shadow is not on this list. A shadow becomes a depth cue only while every shadow in the design agrees about one light direction.

The light has one direction

A shadow tells you where an object sits relative to a surface only because your visual system assumes a single light source, and by default assumes it is above. That assumption is strong enough that a pattern of circles shaded light-on-top reads as bumps and the same pattern shaded dark-on-top reads as dents — turn the page upside down and they swap.

So shadows in a design must all agree. One light direction, one angle, one softness, applied across everything. An interface where cards cast a shadow downwards, a modal casts one in all directions, and a button has a shadow going up is not a design with depth; it is a design with three contradictory claims about a light that does not exist, and the reader's depth reading collapses.

Practical settings for an interface, assuming light from above:

```
--shadow-sm: 0 1px 2px  rgb(0 0 0 / 0.06);
--shadow-md: 0 4px 12px rgb(0 0 0 / 0.08);
--shadow-lg: 0 12px 32px rgb(0 0 0 / 0.12);
```

Note what these have in common: the vertical offset and the blur grow together, and the opacity stays low. A real shadow from a light directly above is soft and faint. A hard 50% shadow at a 45-degree offset is not a shadow; it is a sticker effect, and it was fashionable in 1998 for that reason.

Aerial perspective in a layout

This is the cue most worth learning to apply consciously, because it costs nothing and works everywhere.

To push something back: reduce its contrast against the background, reduce its saturation, and move its lightness towards the background's lightness. To bring something forward: do the opposite.

Concretely, for a decorative illustration behind content, taking it to 40% opacity does all three at once against a light background. For a background photograph behind text, reducing contrast by pulling the blacks up is more effective than reducing brightness, because it is the contrast that carries the foreground reading.

This is also the principle behind disabled states, which are pushed back, and behind a modal dimming the page behind it.

Depth in interfaces is shallow

A last constraint. However many cues are available, an interface almost never needs more than two levels of depth: the page, and a thing above the page. Occasionally three, with a modal above a card above a page.

A design that seems to need four or five levels is usually telling you that the information architecture is wrong — that too many things are trying to be con-

textual at once. The honest fix is structural rather than visual, and adding a fifth shadow value will not supply it.

BEFORE YOU MOVE ON

An interface uses five shadow styles picked individually for each component: cards cast downwards, modals cast evenly in all directions, and a floating button casts slightly upwards. Users describe the interface as cluttered rather than layered. What is the underlying fault?

A. Five levels of depth exceeds what an interface can support, and the design should be reduced to two or three regardless of how the shadows are drawn.

B. Shadows are a weak depth cue, so the design should be using occlusion instead and no shadow arrangement would have worked.

C. Shadows read as depth only under the assumption of a single light source, and contradictory directions leave no coherent depth ordering to read.

D. The shadow opacities are too high for the offsets used, so each element reads as a sticker rather than as a raised surface.

---

51 · 8 min

The space between things has a shape

THE ONE THING TO KEEP

The leftover space in a layout is itself a shape that readers see, so evaluate it directly by filling every element with flat black and looking only at the white — and place the largest area of empty space beside the most important element rather than distributing it evenly.

You designed it whether you meant to or not

Place three objects on a page and you have made four shapes: the three objects, and the space around and between them. That fourth shape is being

seen. It is usually the largest shape on the page, and in most amateur work nobody decided it.

This is the figure-ground idea from the first block turned into a working method. Instead of asking whether an element reads as an object, you ask what the leftover looks like — and you evaluate it as a shape in its own right.

The exercise that teaches it

Take a finished layout. Make a copy. Fill every element with flat black and make the background white. Now look only at the white.

What you are looking for:

- **Is the white a coherent shape, or a set of awkward slivers?** Narrow tapering gaps between elements are the commonest fault and read as tension without purpose.
- **Are the channels consistent?** The white between columns should form clean vertical bands, not a zigzag.
- **Are there trapped pockets?** A small isolated area of white surrounded by content looks like an omission.
- **Does the white have a direction?** Good negative space usually moves — it flows around the composition rather than pooling randomly.

This takes two minutes in any editor and it finds problems that hours of looking at the elements will not, because when you look at the elements you see the elements.

Active and passive space

A useful distinction. **Passive** negative space is the leftover: margins, the gap between a heading and a paragraph, the room around a photograph. It does its job by being consistent and by carrying the grouping information from the spacing block.

Active negative space is space that has been shaped deliberately so that its own form contributes. The space between two figures that forms a recognisable shape. The counter of a letter used as an element. The gap in a wordmark

that becomes an arrow. Interface work uses very little of this, and posters, packaging and identity work use a great deal.

The distinction matters because the two are evaluated differently. Passive space is correct when nobody notices it. Active space is correct when somebody does.

Counters, and the typographic version

Inside letterforms the same principle has a name: the **counter** is the enclosed or partly enclosed space in a letter — the hole in an o, the gap in a c, the two chambers of a g.

Counter size is why some faces survive at small sizes and on bad screens and others fill in. It is why a condensed face becomes unreadable before a normal-width one does: condensing squeezes the counters first. And in the kerning work from the type block, what you are judging when you space letters by eye is the balance between the counters and the spaces between the letters — an area comparison, not a distance one.

Interfaces: the channel problem

The concrete version for screen layout. In a grid of cards, the negative space forms vertical channels between the columns and horizontal ones between the rows. Those channels are a visible structure, and they work when they are straight and consistent.

They break when cards have different heights and the bottom edges do not align, leaving a ragged horizontal channel; when one card spans two columns and interrupts a vertical channel without clearly replacing it; and when the gap is set from margins on the cards rather than from a grid gap, so it varies.

The last one is the padding-and-margin rule from the spacing block showing its consequences: when the container owns the gaps, the channels are uniform by construction.

Where the space should be largest

A distribution rule that is worth more than it looks: **the largest area of empty space should be adjacent to the most important element**, because isolation is an emphasis channel and this is how you spend it.

In practice this means a poster's empty space belongs next to the headline, not distributed evenly around the edges; a card's empty space belongs around the primary action, not shared out among the metadata; a photograph's empty space belongs on the side the subject is looking towards.

Evenly distributed empty space is the default, and like all defaults it communicates nothing. Space that is deliberately uneven tells the reader where to look, for free, in a channel that survives greyscale, sunlight and every form of colour deficiency.

BEFORE YOU MOVE ON

A poster has even margins on all four sides, evenly distributed gaps between its five elements, and a clear hierarchy of type sizes. Reviewers say it looks flat and institutional. What change would most directly address it?

A. Reduce the margins so the composition feels denser, since even spacing at a large scale is what makes the poster feel empty and formal.

B. Introduce a second type size step between the headline and the body, so the hierarchy has more levels to carry the composition.

C. Concentrate the empty space beside the headline and tighten it elsewhere, so isolation reinforces the existing hierarchy.

D. Break the symmetry by moving every element slightly off its current axis, which introduces the tension that even placement removes.

52 · 9 min

One image, four crops

THE ONE THING TO KEEP

Design in the tallest ratio you need and keep everything load-bearing inside the intersection of all the crops you will be asked for, because platform crops, overlay positions and compression all change without notice and margin is the only durable defence.

The ratios you will be asked for

```
1:1     1080 × 1080   square feed posts, avatars, thumbnails
4:5     1080 × 1350   the tallest most feeds allow for a still
9:16    1080 × 1920   stories, reels, shorts, full-screen ver-
tical
16:9    1920 × 1080   video, slides, most screens, YouTube
3:2     e.g. 6000 × 4000   what most cameras produce
2:3     posters, book covers
1:1.414 A-series paper
```

The A-series entry is worth a sentence because the reason is elegant and practical. A4 is 210 × 297 millimetres, a ratio of one to the square root of two. That ratio is the only one that stays the same when you halve the sheet along its long edge — so A4 folds to A5, A5 to A6, and a layout designed for one scales to any other without reproportioning. Nothing in the imperial paper sizes works this way, which is why scaling a US Letter document to half size distorts its margins.

Designing once for several crops

The normal situation is one photograph or one composition that has to appear square in a feed, vertical in a story, and wide on a website. There are two approaches and only one of them scales.

The approach that does not: make each version by hand. It works for three assets and collapses at thirty, and every later change has to be made three times.

The approach that does: design in the tallest ratio you need, mark the safe area for the widest, and keep everything load-bearing inside the intersection.

Concretely, for a 9:16 frame of 1080 × 1920 that must also crop to 1:1 and 4:5:

- The **1:1 crop** takes the central 1080 × 1080 — so 420 pixels come off top and bottom.
- The **4:5 crop** takes the central 1080 × 1350 — 285 off top and bottom.
- The **safe intersection** is therefore the central 1080 × 1080.

Put the subject, the headline and anything else that must survive inside that central square. Everything outside it is atmosphere that some crops will lose.

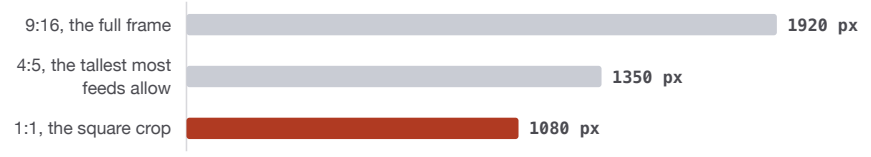
Platform overlays

Separate from the crop, platforms draw their own interface over your image: a username and caption across the bottom, action buttons up the right side, a progress bar and a close button at the top.

The working margins for a 1080 × 1920 vertical frame are roughly 250 pixels clear at the bottom, 150 at the top, and 200 down the right-hand side. Those numbers move — they are a platform's current design, not a standard — but the order of magnitude is stable, and designing to them costs nothing.

The honest limitation: platforms change their crops, their overlay positions and their compression without notice, and they do not announce it to designers. Every published set of exact safe-area pixel values goes stale. The durable defence is margin: keep critical content away from all four edges by a visible amount, and it survives whatever the platform does next.

One 1080 by 1920 frame, three crops



Each crop is taken from the centre, so the square loses 420 pixels off the top and bottom and the 4:5 loses 285. Everything load-bearing goes in the central 1080.

Compression, briefly

An asset uploaded to a social platform is re-encoded. Fine detail, subtle gradients and thin light text suffer most, because that is what lossy compression discards first.

Two practical consequences. Text over an image should be reasonably large and solid rather than thin and pale, because thin pale strokes are exactly what gets mangled. And a large flat gradient — a sky, a brand background — will often come back with visible banding, which is worth checking on the platform rather than in your editor.

Uploading at the platform's stated dimensions rather than larger generally gives a better result, because you control the downscale rather than their encoder.

The container ratio matters too

One thing that catches people out on the web. If a card is 16:9 and the uploaded photograph is 3:2, something has to give: the image is either letterboxed, stretched, or cropped. CSS decides this with one property:

```
.card img { aspect-ratio: 16 / 9; object-fit: cover; object-po-
sition: center 30%; }
```

`cover` crops rather than distorting, which is almost always what you want, and `object-position` lets you bias the crop — `center 30%` keeps the upper third, which is where faces usually are in a portrait photograph.

Without this, the browser either distorts the image or lets the card's height vary with each upload, which breaks the channels from the negative-space lesson. Setting an explicit aspect ratio on image containers also prevents layout shift while images load, which is a measurable performance improvement and not only a visual one.

BEFORE YOU MOVE ON

A team designs a campaign image at 1080 by 1920 with the headline placed 120 pixels from the bottom. It looks correct in the design file. On the platform, the headline is partly hidden. What is the most likely cause?

A. The platform compressed the image, and thin pale text at that size is the first thing a lossy encoder discards.

B. The platform cropped the frame to 4:5, which removes 285 pixels from the bottom and takes the headline with it.

C. The platform's own caption and controls are drawn over roughly the bottom 250 pixels of the frame.

D. The safe intersection for a 9:16 asset is the central square, so anything below 1500 pixels is outside it and cannot be relied upon in any context.

53 · 8 min

Two focal points is none

THE ONE THING TO KEEP

Two elements of similar prominence leave the eye oscillating, so demote one, merge them into a single group, or commit to an exact symmetry — and the diagnostic is not that viewers disagree with you but that they disagree with each other about what they saw first.

The oscillation

A photograph with two faces of similar size and similar prominence. A layout with two large images of equal weight. A slide with a chart on the left and a photograph on the right, both demanding attention.

In each case the eye goes to one, then the other, then back. It never settles, and the viewer's report is that the image feels busy, or unresolved, or that they are not sure what they are looking at. This is the emphasis budget from the first block, applied to pictures: two things at the top rank means the rank is not occupied.

Three resolutions

1. Demote one. The most common and usually the right answer. The available moves, roughly in order of how much they change the image:

- **Crop** so one is partly out of frame. A subject cut by the edge is clearly secondary.
- **Reduce its size**, which requires a real difference — 20% is not enough, as always.
- **Reduce its contrast** against the background, pushing it back with the aerial-perspective cue.
- **Blur it**, which is the photographic version of the same thing.

- **Turn it away.** For a face, having one subject look at the other resolves the competition and makes the pair a relationship rather than a standoff.

2. Merge them. Make the two into one group. Overlap them so occlusion binds them, place them inside a common region, or bring them close enough that proximity groups them. Two faces that touch are one subject; two faces with a metre of space between them are two subjects.

3. Commit to a pair. Sometimes there genuinely are two peers — a before and after, two plans being compared, two people of equal standing. Then make the symmetry exact and obvious: identical size, identical treatment, equal distance from a central axis.

This is the decisive option, and it works because a perfect tie reads as a statement while an imperfect one reads as an unresolved argument. The failure is the near-miss: two subjects at 90% and 100% of each other's prominence.

The competing background

A variant that is easy to miss. The second focal point is often not another subject but a piece of background: a bright window behind a person, a saturated red object in the corner, a patch of sky.

The test is the blur test. Blur the image heavily and see how many bright or dark blobs remain. If there are two, you have two focal points whether or not one of them is a person.

The repairs are the same as above, and one more that only applies to backgrounds: **burn it down**. Selectively darkening a bright background region is a standard photographic move and is free in GIMP or Krita with a soft brush on a darkening layer, or with a gradient mask.

In layouts rather than photographs

The same fault in interface and print work usually looks like this: two cards with coloured headers, two buttons with solid fills, a hero image beside a hero heading, both at maximum.

The resolution is the same demotion. What makes it harder is that somebody has usually argued for each of them separately, so the conversation has to happen at the level of the page rather than the element. The sentence that helps is the one from the emphasis budget: which of these two should somebody see first, if they only see one.

A worked case

A charity wants a poster showing a volunteer and a beneficiary. The photograph has both at similar size, similar distance from the camera, both facing the lens. Everyone involved finds it flat and nobody can say why.

The options, and what each costs:

- **Crop so the volunteer is cut at the frame edge.** Cheapest, strongest, and it makes a statement about who the poster is about — which somebody may object to, and that objection is the real conversation.
- **Turn one to look at the other.** Requires a reshoot, and produces the best image, because it converts two competitors into a relationship with a direction the eye can follow.
- **Blur the rear figure.** Available after the fact if there is any depth separation at all, and it reads as a photographic choice rather than as a design intervention.
- **Darken the background behind one of them**, pushing that figure back with aerial perspective while leaving both fully visible. The most subtle option and the one that survives the most stakeholder scrutiny.

Notice that none of these is make one bigger. Scaling one subject in a photograph is not available, and in a layout it needs a difference large enough to clear the threshold, which is usually larger than anyone is willing to go.

When two is actually right

A composition can carry two subjects when they are at clearly different depths — one in front, one behind, with occlusion and contrast ordering them — or when one is plainly the subject and the other is context that has been pushed back.

The distinction is that in a working two-subject image you can answer immediately which one is primary. In a failing one you cannot, and neither can anybody else, and the three-second test will produce a split of answers across viewers. That split is the diagnostic: not that people disagree with you, but that they disagree with each other.

BEFORE YOU MOVE ON

Three testers each name a different element as the first thing they saw in a layout. The designer concludes the testers were not paying attention. What is the better interpretation?

A. Three testers is too small a sample to conclude anything about first fixation, and the result should be discarded until more data is available.

B. The disagreement between viewers is itself the evidence that no element occupies the top rank.

C. The testers were reporting what interested them rather than what they saw first, so the question needs rewording before the result means anything.

D. First-fixation reports are unreliable without eye-tracking equipment, and a heatmap would show that all three testers actually looked at the same element.

54 · 9 min

Text over a picture you do not control

THE ONE THING TO KEEP

An image has no single contrast ratio, so the worst pixel under the text governs legibility — and because the image is usually unknown in advance, a solid panel or a multi-stop gradient scrim is the robust choice and a text shadow alone is not.

Why this is harder than it looks

A contrast ratio is computed between two colours. An image has thousands. So there is no single ratio for white text over a photograph, and the requirement still applies — which means, in practice, that the **worst pixel under the text** governs whether it is legible.

This would be manageable if you chose every image. Usually you do not. The image comes from a content management system, a user upload, a client's library, or a stock search performed by somebody else six months from now. The design has to work for an image nobody has seen yet.

That constraint rules out the most elegant solution and forces one of the robust ones.

Five approaches, from best to worst

1. Put the text in a plain region of the image. Costs nothing, looks best, and only works when you control the image. A sky, a wall, a shallow-focus background. If you are art-directing a specific photograph, brief for it: ask for space in the frame.

2. A solid panel behind the text. The most reliable option available. The ratio becomes computable because the text sits on a known colour. It looks deliberate rather than apologetic, and it is what newspapers and posters have

always done. The objection is that it covers part of the image, which is only an objection if the image is more important than the words.

3. A gradient scrim. A gradient from transparent to dark, over the region carrying the text. The standard compromise, and it works because it is strongest exactly where the text is and disappears where it is not.

```
.hero::after {
  content: '';
  position: absolute;
  inset: 0;
  background: linear-gradient(
    to top,
    rgb(0 0 0 / 0.75) 0%,
    rgb(0 0 0 / 0.45) 25%,
    rgb(0 0 0 / 0) 55%
  );
}
```

Three stops rather than two matters: a straight two-stop gradient has a visible linear ramp that reads as a band. Easing it with an intermediate stop makes it look like light rather than like a rectangle.

4. A full overlay. A flat dark layer over the whole image at 40 to 60%. Reliable, and it dulls the photograph everywhere including where nothing needed dulling. Acceptable when the image is atmosphere rather than content.

5. Text shadow. The weakest, and the one reached for first. A shadow adds a dark halo around each glyph, which helps marginally against a light background and makes the text look muddy at any size. It also fails completely on a busy background, where the halo merges with the detail. Use it as a small addition to one of the above, never on its own.

A sixth option worth knowing: `backdrop-filter: blur(12px)` blurs whatever is behind an element, which turns a busy background into a smooth one under the text while leaving the rest of the image sharp. Support is now broad, it is expensive to render on low-end devices, and it needs a translucent background colour alongside it to guarantee the ratio.

Checking the worst case

A routine that takes a minute:

1. Find the lightest pixel in the region under white text, using the colour picker in any editor.
2. Compute the ratio against your text colour.
3. Repeat with the darkest pixel if your text is dark.

Then do it again with the worst plausible image rather than the chosen one — a white sky, a snow scene, a photograph of a document. If the design holds for that, it holds.

For a system where images are uploaded by users, the honest answer is that a scrim tuned to a nice photograph will fail on somebody's picture of a whiteboard, and the panel is the correct choice.

Placement

Two practical points that reduce how much scrim you need in the first place.

Put the text where the image is already quiet. Even with an unknown image, the bottom of a photograph is more often darker and less detailed than the middle, which is why so many cards put their text there.

Keep text out of the centre. The centre is where the subject usually is, so text there covers the subject and sits on the busiest region. Top and bottom bands are safer on both counts.

The accessibility position

WCAG applies to text over images exactly as it applies to text on a flat background, and there is no exemption for decorative photography. If the text is content, it needs its ratio.

What the guidelines do not require is that the image be unaltered. A scrim is a legitimate solution, a panel is a legitimate solution, and moving the text off the image entirely is a legitimate solution. The one thing that is not available

is deciding that a photograph is more important than whether people can read the words on it.

BEFORE YOU MOVE ON

A card component places white text over user-uploaded photographs with a two-stop gradient scrim tuned against the sample images. Complaints arrive about unreadable cards. What is the most defensible fix?

A. Add a text shadow beneath the white text, which will preserve the photographs while restoring legibility on the lighter uploads.

B. Replace the scrim with a solid panel behind the text, so the ratio is computable regardless of what is uploaded.

C. Increase the scrim's opacity until it passes against a pure white background, which is the worst case any upload can present.

D. Add a third stop to the gradient so it eases rather than ramping linearly, which removes the visible band and strengthens the darkest region under the text.

---

55 · 8 min

Repetition, and the one that breaks it

THE ONE THING TO KEEP

Repetition builds a prediction and a single deviation violates it cheaply, which makes consistency the thing that gives emphasis its force — so break a pattern once, at the place you want attention, and never twice in the same way.

Prediction is what makes a break loud

A grid of twelve identical cards sets up an expectation within about the third card. Once the eye has predicted the pattern, it stops examining each instance and starts sampling. That is efficient, and it is why a well-repeated structure is restful to look at.

It is also why a single deviation is so loud. The eye is no longer inspecting; it is checking predictions, and a violated prediction is the cheapest possible thing to notice. One double-width card in a grid of twelve is found instantly, from the periphery, in greyscale, at any size.

This makes repetition-plus-break one of the strongest emphasis devices available, and it is nearly free — you did not add anything, you only made one thing different.

The mechanism is the same one that finds typos

A typo stands out in a well-set paragraph and hides in a badly set one. The reason is identical: even typographic colour and consistent spacing establish a prediction, and the misspelling violates it.

This generalises usefully. **The more consistent a design is, the more powerful each deviation becomes.** A design with no consistency has nothing to break, which is why chaotic layouts cannot emphasise anything — there is no baseline against which a difference registers. Restraint is not the opposite of emphasis; it is what makes emphasis possible.

One break per pattern

The rule follows directly from the emphasis budget. A grid with one double-width card has a highlight. A grid with four double-width cards has a second pattern, and neither pattern emphasises anything.

The number is genuinely one, in most cases. Two can work when they are clearly different from each other as well as from the field — a large card and a card with an inverted colour scheme are two different kinds of break, so they do not merge into a rule. Two identical breaks always do.

Kinds of rhythm

Repetition does not have to be even.

- **Regular.** Equal elements at equal intervals. Calm, and the easiest baseline to break.

- **Alternating.** Two treatments taking turns — the striped table, the alternating image-left and image-right sections down a landing page. It establishes a prediction with a period of two, and breaking it is correspondingly louder.
- **Progressive.** Intervals that grow or shrink — spacing that opens up as a page descends, type that steps down through a hierarchy. This is what a type scale is: a progressive rhythm in size.
- **Random.** Not a rhythm. Deliberate randomness in a layout almost always reads as an absence of decisions rather than as a choice, because viewers have no way to distinguish the two.

Repetition across a whole piece

The version that matters most is not within one screen but across many: the same kind of thing always gets the same treatment.

Every section heading looks alike. Every card has the same corner radius. Every destructive action is the same colour. Every photograph is cropped to the same ratio.

This is the beginning of a design system, and the next block is about formalising it. What matters here is the perceptual reason it works, which is that consistency is what lets a reader stop examining. A user who has learned that this shape means a button does not have to work out whether each new one is clickable. Every inconsistency withdraws a little of that, and the cost is paid in attention on every screen, forever.

The failure at the other end

Repetition without any break is monotonous, and it is a real fault rather than a safe default. A page of forty identical rows has no entry point, no rest, and nothing for the eye to hold on to — which is why long lists get section headers, why long documents get pull quotes, and why a wall of identical cards benefits from one that is different.

The diagnostic question is whether the eye has anywhere to land. Blur the page. If the blurred version is a uniform texture with no features, you have

repetition and no rhythm, and the repair is to break the pattern once, deliberately, at the place you most want somebody to look.

BEFORE YOU MOVE ON

A landing page alternates image-left and image-right sections down its length. The team adds a third variant, a centred full-width section, and uses it three times among nine sections. The page now feels disorganised. Why?

A. Three variants is one too many for a page of nine sections, and the layout should be reduced to a single repeating treatment throughout.

B. The centred sections are full-width, so they break the grid's vertical channels and the negative space between sections becomes inconsistent.

C. Repeating the variant three times turns it into a second pattern, so neither pattern establishes a prediction the other can break.

D. Alternating layouts establish a prediction with a period of two, and inserting a third treatment at irregular intervals changes the period to something the eye cannot resolve.

CASE STUDY · MODULE 5

Two teachers, one hoarding, and the crop nobody wanted to make

A coaching centre in Kota with two founding teachers, commissioning one photograph to serve a roadside hoarding, a square feed post and a vertical story.

The centre had grown from one room to four batches, and the two founders taught physics and chemistry between them. For the new session they paid for a photographer, a day's shoot, and a designer to produce the campaign.

The brief was to use one photograph across everything, because the budget would not stretch to three shoots and because the two founders wanted the same image on the hoarding, the feed and the printed handbill outside the railway station.

The photograph delivered was competent and had a specific problem. Both teachers stood at similar distance from the camera, at similar size, both facing the lens, one slightly left of centre and one slightly right, with the centre's board behind them. Everybody who saw it said it felt flat, and nobody could say why.

The diagnosis

The designer ran two checks. She blurred the image heavily and found two masses of almost identical weight, plus a third from a bright window behind the right-hand figure. Then she showed the composition to five people for three seconds each and asked what they had seen first.

The answers split: two said the left teacher, two said the right, one said the window. That split was the finding. Viewers disagreeing with the designer is an opinion; viewers disagreeing with each other about what they saw first is a measurement, and it says there is no primary at all.

There was a second problem underneath the first. The image had to crop to 16:9 for the hoarding, 1:1 for the feed and 9:16 for the story. With the two fig-

ures placed near the left and right thirds, the vertical crop could not contain both without cutting one at the shoulder, and the square crop put a figure hard against each edge.

The options, and what each cost

The designer put four on the table with the cost of each stated.

Crop so one teacher is cut by the frame edge. Cheapest, strongest, and available immediately from the file they already had. It makes one founder secondary and one primary, permanently, on a hoarding their students' parents drive past.

Reshoot with one teacher turned to look at the other. Produces the best image, because it converts two competitors into a relationship with a direction, and it also solves the crop, because the pair can then be composed closer together. It costs a second shoot day and pushes the campaign past the date the hoarding contract started, which was already paid for.

Blur or darken behind one figure. Subtle, survives scrutiny, available after the fact. It still leaves two figures at similar size, so the oscillation is reduced rather than removed, and it does nothing at all for the vertical crop.

Commit to an exact symmetry. Place both at identical size, identical treatment, equal distance from a central axis, looking straight out. A deliberate tie reads as a statement rather than an unresolved argument. It is the only option that treats the two founders equally, and it makes the hoarding static and the 9:16 crop nearly impossible, because an exact pair needs horizontal room the vertical frame does not have.

The decision was never really a visual one. Two people who had built the centre together had to agree which of them the campaign would be about, or agree to spend money and time so that neither had to be.

STOP HERE

Before turning the page: what would you have done, and what would it have cost if you were wrong? Write it down. The point of the next section is to compare.

WHAT HAPPENED

They reshot, and the extra day was cheaper than either founder had expected once the photographer was already engaged. The new frame placed the chemistry teacher three-quarters towards the left, looking across at the physics teacher, who faced the camera. Occlusion did the rest: a slight overlap of one shoulder in front of the other ordered the two figures in depth and bound them into a single group, so the eye stopped oscillating and read the pair as one subject with a direction.

The composition was built in 9:16 with everything load-bearing inside the central square, so the 1:1 and 16:9 crops came out of the same file without loss. The hoarding ran four days late against the contract and the vendor did not charge for it.

The unexpected result was in the handbills. The designer had assumed the crop discipline was a production convenience. What it actually removed was the three rounds of argument that had happened in every previous campaign about whose face was larger, because the question had been settled once in the frame rather than repeatedly in the crops.

The five-person test produced a split answer. Why is that more useful than the designer's own judgement that the image felt flat?

A split tells you the failure mode rather than that there is one. If viewers disagree with the designer, that is a difference of opinion about what should be first. If viewers disagree with each other about what they saw first, no element is winning the first fixation, which is the definition of two focal points and therefore none. It converts an unnamed feeling into a specific diagnosis with known repairs.

Why did the reshoot solve the aspect-ratio problem as well as the focal-point problem?

Both problems came from the same arrangement: two figures at similar prominence, spread towards the left and right thirds. Turning one to look at the other and overlapping the shoulders let them be composed close together as one group, which meant everything load-bearing could sit inside the central square that all three crops share. A composition does not survive reframing unless it was designed inside the intersection of the crops it has to serve.

The exact-symmetry option treated the two founders equally. Why was it still rejected?

A deliberate tie is a legitimate resolution and reads as a statement rather than an argument, but it has to be exact to work, and an exact horizontal pair needs width the 9:16 frame does not have. It would have solved the emphasis problem while making the vertical crop impossible, which was one of the three outputs the single photograph had to serve.

MODULE 5 · TEST

Check what stuck

6 questions, one right answer each. This one is for you, not for a record: answers and the reason behind each are at the back. If two questions from the same module go wrong, re-read that module before the exam rather than after.

1 Why does a near-touch between two contours read worse than a clear overlap?

A. It removes the occlusion that orders the two in depth
B. It creates a moire pattern where the two edges interfere
C. It breaks the rule of thirds by splitting the frame unevenly
D. The gap is too small to be rendered on a low-density screen

2 A portrait taken from twenty centimetres away has a distorted nose. Can cropping repair it?

A. Yes, if the crop is taken from the centre of the frame
B. No, because perspective was fixed by where the camera stood
C. Yes, provided enough resolution remains after the crop
D. No, but a perspective correction restores the original geometry

3 A card shows white text over photographs uploaded by users. Which treatment is the robust choice?

A. Raising the text to a bold weight so the strokes survive
B. A text shadow, which adapts to whatever sits behind it
C. A solid panel behind the text
D. A two-stop gradient tuned against a sample photograph

4 Where should the largest area of empty space in a poster go?

A. In the centre, so that the elements frame it
B. Along the bottom, underneath the call to action
C. Distributed evenly around all four edges
D. Beside the most important element

5 A grid of twelve identical cards is given four double-width cards to create emphasis. What happens?

A. The four become a second pattern, and nothing is emphasised

B. The emphasis is four times stronger than with a single one

C. The grid becomes unbalanced and needs a fifth to restore symmetry

D. The eye reads the twelve small cards as the break instead

6 One 1080 by 1920 composition must also crop to 1:1 and 4:5 from the centre. Where does the headline have to sit?

A. Centred vertically in the 9:16 frame, then re-placed for each crop

B. Inside the central 1080 by 1080

C. In the lower 250 pixels, clear of the platform's buttons

D. Anywhere inside the 4:5 crop, which is the tallest most feeds allow

MODULE 6

Real screens, real thumbs, real paper

A design lives somewhere: a 360-pixel phone held one-handed on a bus, a 200% browser zoom, a sheet of 130gsm paper coming off a press with 3mm trimmed off every edge. This block covers the numbers those surfaces actually have — device widths and pixel ratios, thumb reach, zoom and reflow requirements, page weight, bleed and paper — and what each one forces you to decide differently.

> BY THE END OF THIS MODULE YOU CAN
>
> Produce a layout that works at 360 CSS pixels one-handed in daylight, at 400% zoom, and on paper with correct bleed and resolution, naming for each constraint the specific number it imposes and which design decisions change because of it

56 · 9 min

Design it at 390 pixels or find out the hard way

THE ONE THING TO KEEP

Design at 360-390 pixels wide and test on a cheap phone in daylight, because shrinking a desktop layout hides the faults that expanding a phone layout exposes.

The order most people work in is backwards

The standard sequence is: build it comfortably on a large monitor, admire it, then near the end check what it looks like on a phone and patch the damage. For a majority of the people who will ever see your work, the patched version is the only version that exists.

Reverse it. Design at phone width first and expand outwards. The reason is not moral, it is diagnostic. Scaling a desktop layout down hides hierarchy faults, because every element shrinks together and their relative ranks are preserved even though none of them is legible any more. Scaling a phone layout up exposes faults, because you have to decide what deserves the extra room. The direction that forces decisions is the useful one.

Design at the width most people actually have

Common Android phone widths land around 360 to 412 CSS pixels. An iPhone is typically 390 or 393. If you draw at 360 or 390 and it works, it works nearly everywhere; the reverse is not true.

You do not need an expensive tool to do this. Figma's free tier and Penpot both ship phone frame presets. So does Canva's free tier. In a browser, Chrome and Firefox both have a device toolbar built in — press F12, then the device icon — where you can set an exact width and throttle the connection to something slow, which is the second half of the test.

The numbers that are load-bearing

- **16 pixels minimum for body text on the web.** Not a style preference: iOS Safari zooms the whole page when a text field with a font size below 16px receives focus. A type decision therefore becomes a layout bug, and it is one of the most common ones on the mobile web.
- **Tap targets.** Apple's guidelines say 44 by 44 points; Google's Material guidance says 48 by 48; the WCAG 2.2 minimum is 24 by 24 CSS pixels. Under those, people miss, and on a touch screen a miss is not a small annoyance — it often triggers the wrong thing.
- **Thumb reach.** Held one-handed, the easy zone is the lower half and the side of your dominant thumb. The hardest point on a large phone is the opposite top corner. Put the main action low. Do not put a destructive action next to it.
- **Safe areas.** Notches, rounded corners, the home indicator bar, and the browser chrome that appears and disappears as you scroll. Content that reaches the very edge of your artboard will be clipped or covered on some real device.

What survives the shrink, concretely

- **A logo with a tagline locked inside it does not.** At a 48-pixel avatar the tagline is a grey smudge, and at a 16-pixel favicon so is the wordmark. Draw a second, simplified mark for small sizes. Every large brand has one.
- **Three-column layouts do not.** They stack, and stacking changes reading order. Decide the stacked order deliberately rather than accepting whatever the source order happens to be.
- **Wide tables do not.** A table with six columns becomes a horizontal scroll nobody scrolls. Turn each row into a small card with labelled values.
- **Thin type weights do not.** A hairline weight that looks elegant at 60 pixels disintegrates at 14 on a screen being viewed at an angle.
- **Pale grey text does not**, for the reason in the next section.

Daylight is a contrast problem, and it is the real test

Your design was approved in a room, on a bright screen, indoors. Your reader is outside, at 30 per cent brightness, with a smeared screen protector. In those conditions a combination that just passed 4.5:1 disappears.

The test costs nothing: take a screenshot, walk outside with your phone, and look at it. That is the whole procedure, and it will change your greys permanently. It is also why a contrast floor is a floor rather than a target.

Weight is money for your reader

A 3MB hero image is not just slow. On a metered connection it is a cost the reader pays, and on a slow one it is a blank rectangle they will not wait for.

- Export at 2x, not 3x. The visible difference on a phone is small and the file is roughly half.
- Use WebP, or a JPEG at quality 75 to 80. Between quality 80 and 100 the difference is invisible at phone size and the file is often three times bigger.
- Resize before you export. A 4000-pixel-wide photo displayed 390 pixels wide is 99 per cent waste.

Free tools for all of this: Squoosh, a browser page that compresses images on your own machine; GIMP's export dialogue, which shows the resulting file size as you move the quality slider; Photopea's Save for Web. On a phone, most gallery apps can resize on export.

Test on the cheapest device you can borrow

Not your phone. Your phone is fast, new and bright, and it is not representative of the person you are designing for. Borrow the oldest, cheapest handset in the room and open your work on it. This single habit catches more real problems than any amount of theory, and it is free.

Today

Screenshot something you have made, open it on a phone, and take it outside. Then hold the phone in one hand and try to reach every interactive element

with your thumb without shifting your grip. Write down what you could not read and what you could not reach.

BEFORE YOU MOVE ON

A signup page looks correct in the browser's phone preview, but on a real iPhone the whole page jumps and zooms in the moment someone taps the email field. What is the most likely cause?

A. The keyboard shrinks the visible viewport and the layout has no fixed height, so it reflows.

B. The input's font size is below 16 pixels, and iOS Safari zooms the page when a field with text smaller than that receives focus.

C. The viewport meta tag is missing, so the page renders at desktop width and is then scaled to fit.

D. The tap target is smaller than 44 points, so the browser magnifies the area to disambiguate the tap.

57 · 9 min

What a pixel is, and how wide phones actually are

THE ONE THING TO KEEP

A CSS pixel is an angular reference rather than a physical size, device pixel ratio means raster assets need 2x or 3x export while vectors need none, and 360 CSS pixels is the width to design at because a layout that fits at 393 can still break there.

A CSS pixel is not a physical thing

This trips up nearly everybody once. A CSS pixel is a reference unit defined by the angle it subtends at a typical viewing distance, not by a fixed physical size.

The specification's nominal value is 1/96 of an inch at arm's length, which is about 0.26 millimetres — but that is the value for a screen viewed at roughly 60 centimetres.

On a phone held 30 centimetres away, the browser's CSS pixel is physically smaller. An iPhone 15 reports 393 CSS pixels across a screen about 71 millimetres wide, so one CSS pixel is about 0.18 millimetres there. The 44-pixel touch target from the spacing block is therefore about 8 millimetres of glass, not 11.

This is intentional rather than broken. What matters for perception is angular size — how much of your visual field something occupies — and a phone held closer needs fewer millimetres for the same angle. It does mean that reasoning about pixels as physical distances is unreliable, and that anything with a genuinely physical requirement, such as a touch target, should be checked on a device rather than on a ruler.

Device pixel ratio

The hardware has more pixels than the CSS layer admits to. Device pixel ratio is the multiplier:

```
iPhone 15        393 CSS px wide,  DPR 3  → 1179 hardware px
budget Android   360 CSS px wide,  DPR 2  →  720 hardware px
1080p laptop    1536 CSS px wide,  DPR 1.25 (Windows at 125%
scaling)
MacBook Pro     1512 CSS px wide,  DPR 2
```

Two consequences that decide real work:

Raster assets need to be exported at 2x or 3x. A logo displayed at 120 CSS pixels wide needs a 360-pixel file to look sharp on a DPR-3 phone. An image exported at 1x looks soft on almost every modern device.

Vector assets do not. An SVG logo or icon is resolution-independent and sharp at any ratio, at a fraction of the file size. This is the single strongest argument for drawing interface graphics as vectors, and Inkscape does it free.

The widths to design at

Mobile viewport widths cluster tightly. 360 and 390 to 393 CSS pixels cover the overwhelming majority of phones in use, and 360 is the safe floor — it is what a great many Android devices in India, Africa and Latin America report.

Design at **360**. If it works there it works at 393. The reverse is not reliable: a layout that fits at 393 can break at 360, and the break is usually a fixed-width element or a long unbreakable string.

On desktop the common widths are far more spread out, and the useful number is not the screen width but the content width, which you are choosing anyway — the measure rule from the type block caps a text column at roughly 30 to 34em regardless of how wide the monitor is.

The 100vh problem

A specific, extremely common bug. `height: 100vh` on a mobile browser refers to the viewport height with the browser's address bar hidden. While the bar is showing — which is most of the time — the visible area is smaller, so the bottom of a supposedly full-height section is cut off, taking the button with it.

The modern units solve it:

```
.hero { min-height: 100svh; }  /* smallest: bars showing */
.hero { min-height: 100lvh; }  /* largest: bars hidden */
.hero { min-height: 100dvh; }  /* dynamic: changes as bars move
*/
```

`svh` is the safe default for anything that must be fully visible. `dvh` is correct for a full-screen overlay but causes the layout to resize as the user scrolls, which can look unsettled.

Safe areas

Phones have rounded corners, camera cut-outs and a home indicator bar. The operating system exposes the insets, and a layout that ignores them puts content under the notch or under the swipe bar:

```
.footer-bar {
  padding-bottom: max(16px, env(safe-area-inset-bottom));
}
```

The `max()` means you get your normal padding on devices with no inset and the correct clearance on devices with one.

Testing honestly

Browser device emulation is good for layout and useless for two things it appears to test: touch accuracy, and how anything looks in real light. Both need the actual device.

The cheapest useful test rig is one inexpensive Android phone, bought second-hand, kept on the desk, and used outdoors. It will find contrast failures, target failures and performance problems that no emulator reports, because emulation runs on your fast machine in your dim room with your mouse.

BEFORE YOU MOVE ON

A full-screen section is set to height: 100vh with a button at its bottom edge. On desktop it is fine; on phones the button is often cut off. What is the mechanism?

A. The device pixel ratio makes the section render taller in hardware pixels than the CSS height declares, pushing the button past the physical screen edge.

B. The safe-area inset at the bottom of the device is overlapping the button, and env(safe-area-inset-bottom) would restore the clearance.

C. 100vh measures the viewport with the browser bars hidden, so the visible area is smaller whenever the address bar is showing.

D. Mobile browsers apply a minimum font size that expands the section's content beyond the declared height, so the button is pushed out of view.

58 · 8 min

Breakpoints come from the content, not from devices

THE ONE THING TO KEEP

Put breakpoints where your content measurably breaks rather than where devices sit, prefer intrinsic rules and container queries over enumerated cases, and design the small screen first because a narrow column forces the hierarchy decisions a wide one lets you avoid.

Naming breakpoints after phones ages badly

A stylesheet with `$tablet: 768px` was written when there was a tablet at 768 pixels. There are now devices at every width from 320 to 3840, foldables that change width mid-session, split-screen multitasking, and browser windows people resize for no reason.

The method that does not age: **widen and narrow the browser slowly, and put a breakpoint wherever the layout stops working.** Not where a device sits. Where your content breaks.

This produces breakpoints at odd numbers — 540, 720, 960, 1180 — and that is the sign you did it right. Round numbers usually mean somebody copied them.

What breaking looks like

The specific failures to watch for as you resize:

- **The measure exceeds 75 characters.** Time to cap the column or add a second one.
- **The measure drops below about 35 characters.** Time to drop a column.

- **A heading breaks into a bad shape** — one word on the second line, or a wrap mid-phrase.
- **A horizontal layout has less than about 200 pixels per item.** Cards stop working somewhere around there.
- **Navigation runs out of room.** Usually the first thing to break on the way down.
- **A table starts scrolling horizontally.** Which means it needs a different presentation, not a smaller font.

Each of these is a measurement you can take, which is what makes this a technique rather than an aesthetic judgement.

Container queries change the unit of the question

A media query asks how wide is the window. That is the wrong question for a component, because the same card might appear in a wide main column, a narrow sidebar and a three-across grid, all on the same page at the same viewport width.

Container queries ask how wide is the space this component has been given:

```
.card-wrap { container-type: inline-size; }

@container (min-width: 28rem) {
  .card { display: grid; grid-template-columns: 8rem 1fr; }
}
```

The card becomes horizontal when its own container is wide enough, wherever that container happens to be. This has been available across the major browsers since 2023, and it is the most significant change to responsive layout in a decade — because it means a component can carry its own responsive behaviour and be reused without the page having to know.

Many layouts need no breakpoints at all

Before reaching for a query, check whether the layout can simply be intrinsic:

```
.grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(16rem, 1fr));
  gap: var(--space-4);
}
```

That reads: as many columns as fit, each at least 16rem, sharing the space equally. One declaration, no breakpoints, correct at every width including ones that do not exist yet. Combined with `clamp()` for type sizes from the type block, a surprising amount of responsive design needs no media queries whatsoever.

The general principle: prefer a rule that describes the constraint over a rule that enumerates the cases.

Small first, for a design reason

Mobile-first and desktop-first are code organisation preferences and neither is meaningfully better as CSS. What is not merely a preference is designing the small version first, and the reason is nothing to do with implementation.

A 360-pixel column forces prioritisation. There is room for one thing at a time, so you have to decide what comes first, second and third — which is the hierarchy work from the first block, made unavoidable. A 1440-pixel canvas lets you avoid the decision by putting four things side by side, and the resulting design has no hierarchy because nobody ever had to choose.

Going small to large is therefore addition: you have a working priority order and you are deciding what can now sit beside what. Going large to small is subtraction under pressure, usually the afternoon before a deadline, and it is where features get hidden behind a menu because nobody can face rethinking the order.

The honest limit

Some content genuinely does not fit a phone. A complex comparison table, a dense schematic, a spreadsheet, a seat map.

The answer there is not a clever responsive trick; it is a different presentation for the small screen — a card per row instead of a table, a filtered subset with a link to the full version, a pinch-zoomable image with explicit controls. Pretending otherwise produces a table at 9 pixels that nobody can read, which is worse than admitting the constraint and designing for it.

BEFORE YOU MOVE ON

A card component switches to a horizontal layout at a 768px media query. In the main column it looks right; in a 300px sidebar on a wide desktop it also goes horizontal and becomes cramped. What is the correct fix?

A. Add a more specific media query for the sidebar context, so the card stays vertical when it appears there.

B. Use a container query so the card responds to its own available width rather than to the window's.

C. Raise the breakpoint above 768px so the card only goes horizontal on genuinely wide screens where sidebars are not used.

D. Replace the media query with an intrinsic grid using auto-fit and minmax, so the card's internal layout adapts without any query being needed at all.

59 · 8 min

Where a thumb can actually go

THE ONE THING TO KEEP

The top-left corner is the hardest point on a large phone for most people, so frequent actions belong in the easily reached bottom while destructive ones need separation — and because people switch grips constantly, nothing essential may sit where a single grip cannot reach it.

Phones outgrew hands

A phone with a 4-inch screen could be operated entirely with one thumb. A phone with a 6.7-inch screen cannot: the top corner opposite the holding hand is genuinely out of reach without regripping, and the whole top third requires a stretch that makes the phone unstable.

The reach map, for a right-handed one-thumb grip:

- **Easy**: the bottom half, and the lower-right region.
- **Stretch**: the middle, and the upper-right.
- **Hard or impossible**: the top-left corner and the top edge.

Mirror it for a left hand. Which means the top-left corner — historically where the back button, the menu and the logo all live — is the hardest point on the screen for roughly nine people in ten.

What the industry did about it

This explains a decade of interface changes that otherwise look like fashion:

- **Bottom navigation bars** replacing top tab bars in most mobile apps.
- **Safari moving its address bar to the bottom** of the screen on iPhone.

- **Reachability** on iOS and one-handed mode on Android, which slide the whole screen down on a gesture.
- **Bottom sheets** replacing dialogues that used to appear centred or at the top.
- **Primary actions pinned to the bottom** of forms and flows rather than sitting at the end of the content.

Each of those is the same finding applied.

The numbers, honestly

The widely cited figure is that about half of phone use is one-handed with the thumb. It comes from an observational study published in 2013 that watched 1,333 people using phones in public. That work is genuinely useful and it is also twelve years old and predates phones of the current size.

What has held up in later work is the more important finding: **people switch grips constantly**, within a single session — one-handed thumb, one-handed with the other hand steadying, two-handed with both thumbs, phone on a table. Nobody uses one grip.

So the design conclusion is not optimise for the thumb zone. It is:

1. **Put frequent actions within easy reach**, because that is where they are cheapest.
2. **Do not put anything essential where it cannot be reached**, because some share of uses will be one-handed.
3. **Do not assume a grip**, which means avoiding gestures that only work from one side and layouts that only make sense held one particular way.

Consequence, not recessive

The reach map interacts with the destructive-action rule from the touch-target lesson. Easy to reach is also easy to hit by accident, especially while walking.

So the allocation is two-dimensional:

```
frequent + safe        → bottom centre, large
frequent + destructive → bottom, but separated and confirmed
rare + safe            → top, or behind a menu
rare + destructive     → top or in a submenu, and confirmed
```

A delete button in the easiest-to-hit position on the screen is a design that will generate support tickets, however well it scores on reachability.

Gestures are not targets

One useful distinction. A pull-to-refresh gesture starts at the top of the screen and is perfectly comfortable one-handed, because a swipe can begin anywhere the thumb can touch and does not require precision. Reach constraints apply to **targets**, which require accuracy, far more than to **gestures**, which do not.

This is why the top of the screen is a reasonable place for a scrollable content area and a poor place for a small button.

The caveat is that gestures are invisible. A swipe-to-delete that nobody discovers is not a feature, and every gesture needs a discoverable equivalent somewhere — which is usually a button, which then needs a reachable position, which brings you back to the map.

The layout that follows

Put together, the reach map and the grip variation produce a fairly specific shape for a phone screen:

- **Bottom**: primary navigation, the main action of whatever screen you are on, and anything used repeatedly.
- **Middle**: content, which is scrolled rather than tapped, and where any gesture works comfortably.
- **Top**: the title, contextual information, and at most one control — which should have an equivalent lower down if it matters.

This is close to an inversion of the desktop convention, where navigation sits along the top because a mouse reaches everywhere equally and the top is

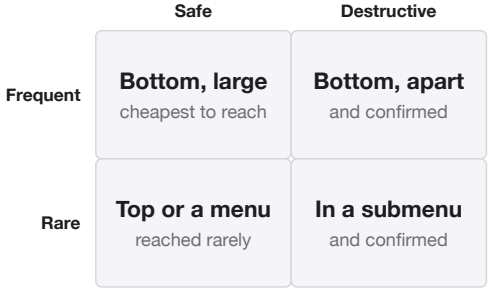
where the eye starts. Carrying the desktop arrangement onto a phone is the commonest cause of an app that feels awkward without anybody being able to say why.

One caveat about the bottom edge itself. The lowest strip of the screen belongs to the operating system on both platforms — the home indicator on iOS, the gesture bar on Android — so a control flush against the bottom edge competes with a system swipe. The safe-area padding from earlier in this block is what keeps them apart.

Testing it

The check costs nothing and is skipped almost universally: **use your own design on a phone, standing up, one-handed, while holding something in the other hand.**

Where an action belongs on a phone



Easy to reach is also easy to hit by accident, especially while walking. A delete button in the easiest position on the screen will generate support tickets.

Sitting at a desk with the phone in both hands is not a test of reach, and neither is a simulator. Ten minutes of genuinely one-handed use will tell you more about your interface's bottom half than any amount of annotating a heat map.

BEFORE YOU MOVE ON

A team moves every control on a phone screen into the bottom third to maximise reachability, including Delete account, which now sits beside Save. What have they got wrong?

A. Reach research is based on a 2013 study of smaller phones, so the bottom-third assumption does not hold for current devices and the controls should be distributed evenly.

B. Easily reached is also easily hit by accident, so a destructive action placed there needs separation and confirmation rather than optimal reachability.

C. Placing every control in one region violates the touch-target spacing minimum, since the available area cannot hold them all at 48 pixels with 8-pixel gaps.

D. Controls in the bottom third collide with the home indicator's safe area inset, so any control placed there will intercept the system's swipe gesture instead of registering a tap.

60 · 8 min

Why 10pt is fine in print and terrible on screen

THE ONE THING TO KEEP

Print builds a 10pt letter from hundreds of dots while a screen builds it from about ten pixels, so screen minimums are larger — 16px body, and exactly 16px on form inputs because iOS auto-zooms below that — and font smoothing declarations change stroke weight rather than sharpness.

The resolution gap

A commercial press lays ink at something like 2,400 dots per inch. A standard-density screen has 96 pixels per inch, and a high-density phone has the equivalent of two or three hundred.

So a 10-point letter in print is built from hundreds of dots in each direction and can carry fine serifs, hairline strokes and delicate curves. The same letter at 10 pixels on a 1x screen has about ten pixels of total height, of which perhaps five are the x-height. There is no room for a hairline. There is barely room for a stem and a counter.

That is the whole reason screen minimums are larger than print minimums, and it is why advice imported from print typography — 9pt captions, high-contrast faces, tight leading — produces unreadable web pages.

Working minimums: 16px for body text on screen, 12px absolute floor for anything, against 9 to 10pt comfortably readable in print and 7pt as the floor for legal small print.

The iOS zoom trap

A concrete consequence that catches people constantly. On iOS, Safari automatically zooms the page when a user taps into a form input whose font size is below 16px. The layout jumps, the user loses their place, and zooming back out is manual.

The fix is to set inputs at 16px or larger. Not a rounding error — exactly 16 or above:

```
input, select, textarea { font-size: 16px; }
```

This single rule removes an irritation that appears on a large share of forms on the web.

Hinting and antialiasing

Hinting is a set of instructions inside the font telling the rasteriser how to snap stems onto the pixel grid at small sizes, so that two stems that should be the same width do not end up one pixel and two. It mattered enormously at 96dpi. At device pixel ratios of 2 and 3 it matters much less, because there are enough pixels that a slight misalignment is invisible.

Antialiasing fills the edge pixels with intermediate values so curves do not look like staircases. Two kinds exist: greyscale, and subpixel, which uses the separate red, green and blue elements within each pixel to gain three times the horizontal resolution. Subpixel rendering produced visibly sharper text at 1x and produced colour fringing if the screen's subpixel order differed or the screen was rotated. Apple removed subpixel antialiasing from macOS in Mojave, because high-density displays made it unnecessary.

One practical note. The declaration `-webkit-font-smoothing: antialiased` is widely copied into stylesheets as a fix. What it actually does on macOS is force greyscale antialiasing, which makes text visibly **thinner** than the default. On a light background that often looks worse, not better. On a dark background it partly offsets the halation thickening from the colour block, which is sometimes a genuine reason to use it — but apply it knowing that it is a weight change, not a sharpening.

What you cannot control

A longer list than most designers expect:

- The operating system's rendering engine, which differs between Windows, macOS, Android and Linux.
- Whether the user has increased their default font size, which they are entitled to do and which your layout must survive.
- Browser zoom level.
- Screen density and subpixel arrangement.
- Whether a web font loaded at all, and what the fallback did to the layout while it did not.

The design conclusion is to build with margin rather than with precision. A layout that only works at exactly 16px with exactly one typeface at exactly 100% zoom is a layout that breaks for a substantial number of real readers.

The font loading moment

One last screen-specific concern. While a web font downloads, the browser shows either nothing or the fallback face. `font-display: swap` shows the

fallback immediately and swaps when the font arrives, which is almost always the right choice — invisible text is worse than text that changes.

The swap produces a visible reflow if the fallback has different metrics, and that jump is exactly what `size-adjust` and `font-size-adjust` in an `@font-face` block exist to reduce. Matching the fallback's x-height to the web font's makes the swap nearly invisible, which is the x-height arithmetic from the type block doing practical work.

BEFORE YOU MOVE ON

A designer copies a caption style from a print layout — 9pt, a high-contrast serif, tight leading — onto a web page and finds it nearly unreadable on a standard-density laptop. Which explanation is most complete?

A. Serif faces are unsuited to screens, and the caption should be set in a sans-serif at the same size.

B. 9pt is below the 12px absolute floor for screen text, so no typeface or spacing choice could rescue it at that size.

C. At around 96 pixels per inch there are too few pixels to render hairline strokes, so the face's thin strokes drop out and the tight leading merges the lines.

D. Print measurements in points do not translate to CSS pixels, so the caption is rendering at roughly two-thirds of its intended size and needs converting at 96dpi first.

61 · 9 min

The five screens nobody designs

THE ONE THING TO KEEP

Every screen ships in at least six states — empty, one, too many, loading, error and offline — and the undesigned ones default to a blank rectangle and the words something went wrong, so produce all six before calling a screen finished.

The happy path is one state out of six

Almost every design review looks at a screen full of plausible content. The screens that actually determine how the product feels are the other ones:

1. **Empty.** No items yet. The first thing every new user sees.
2. **One item.** Where a grid designed for twelve looks broken.
3. **Too many.** Two hundred rows, a name of sixty characters, a number with nine digits.
4. **Loading.** Between the tap and the content.
5. **Error.** The request failed.
6. **Offline or no permission.** Which is an error with a different remedy.

Each of these ships whether or not it was designed. Undesigned, they ship as a blank rectangle, a broken grid, truncated text, a spinner, and the words Something went wrong.

The empty state is a first impression

The screen a new user sees before they have done anything is, by definition, the empty state — and it is usually the last screen anybody designs, if it gets designed at all.

A useful empty state does three things: says what belongs here, says why there is nothing, and offers the action that fills it. That is a sentence and a button. The illustration is optional and frequently takes the place of the sentence, which is the wrong trade.

Distinguish between empty types, because they need different responses:

- **Nothing yet** — invite the first action.
- **Nothing matching this filter** — offer to clear the filter, and say what was filtered.
- **Nothing because something failed** — that is an error, not an empty state, and saying No results when the request failed is a lie the user will act on.

Loading, and the thresholds that matter

The response-time thresholds have been stable since they were first characterised in the 1960s and repeated in usability work since:

- **Under about 100 milliseconds** feels instantaneous. No indicator. Adding a spinner here makes the interface feel slower, because a flash of spinner reads as a stutter.
- **Up to about 1 second** keeps the user's flow of thought unbroken. A subtle indicator at most.
- **Up to about 10 seconds** is the limit of held attention. A clear indicator is required, and ideally something that shows progress.
- **Beyond 10 seconds** the user will switch to something else. You need a progress bar with a real estimate, and ideally a way to leave and be notified.

The practical rule that follows: do not show a loading indicator immediately. Delay it by around 300 milliseconds, so a fast response never flashes one.

Skeletons versus spinners. A skeleton screen shows grey blocks in the shape of the content to come. The argument for it is that it communicates the layout and reduces the sense of waiting; the evidence is largely industry testing rather than strong independent research, and results are mixed. What is

uncontroversial is that a skeleton must match the real layout, or the content shifts when it arrives, which is worse than a spinner was.

Error messages

An error message has three jobs and most manage none: say what happened, say why if you know, say what to do next.

```
Bad:  Something went wrong.
Bad:  Error 500.
Good: We could not save your changes because the connection
dropped.
      Your text is still here. Try again.
```

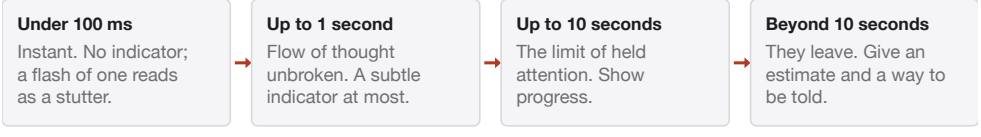
The last line of that is the part that matters most and is omitted most often. A user who does not know whether their work survived will assume it did not.

Never blame the user, never show a stack trace to somebody who cannot act on it, and never use red alone — the colour rule from the previous block applies here as much as anywhere.

Design with hostile content

The third state on the list deserves a specific habit. Design with the longest realistic values, not the convenient ones:

Response time, and what each band needs



Delay any indicator by about 300 milliseconds, so a fast response never shows one at all.

- A name that is sixty characters, or two characters.
- A price of 1,20,00,000 rather than 99.
- A title that wraps to four lines.

- A translated string. German runs roughly 30% longer than English on average; Tamil and Hindi strings are frequently longer too, and a button sized to fit Save in English will not fit its translation.

Using John Smith and Item name in a mockup is not neutral; it is a decision to test the easiest case only. Substituting the longest plausible values takes two minutes and finds most truncation bugs before they exist.

The checklist

Before any screen is considered finished, produce it in all six states. It takes far less time than it sounds — most are a variation on the same layout — and it converts a class of bug that normally surfaces in production into an hour of design work.

BEFORE YOU MOVE ON

A team adds a spinner that appears the instant any request starts. Most requests complete in under 200 milliseconds, and users report the interface feels unstable. What is happening?

A. The spinner animates in the periphery and captures attention involuntarily, so frequent appearances become a stream of interruptions.

B. Responses under 100 milliseconds feel instantaneous, so an indicator that flashes and vanishes reads as a stutter rather than as feedback.

C. A spinner communicates no progress information, and a skeleton screen matching the content layout would remove the instability.

D. The spinner is being rendered before the layout has settled, so it causes a layout shift each time it appears and again when it is removed.

62 · 8 min

Motion, and the people it makes ill

THE ONE THING TO KEEP

Motion overrides every other attention channel, so keep it to continuity, feedback and direction at 100 to 400 milliseconds with ease-out — and honour reduced-motion by replacing movement with a fade, because large-scale motion causes real symptoms rather than mild annoyance.

Motion is the loudest channel you have

From the first block: peripheral motion detection is involuntary. A moving element does not compete for attention with size and contrast; it overrides them. That makes motion powerful and makes it the channel most easily misused.

Used well, motion does three specific jobs:

- **Continuity.** Showing that this panel came from that button, so the user does not have to re-orient.
- **Feedback.** Confirming that a tap registered.
- **Direction.** Indicating that content came from the right, so going back means going left.

Anything else is decoration, and decoration in the most intrusive channel available is a poor trade.

The durations

Working numbers, which are consistent across the major platform guidelines:

```
100–150ms   a small local change: hover, a checkbox, a colour
shift
200–250ms   an element entering or leaving: a menu, a tooltip
250–350ms   a panel or sheet crossing much of the screen
300–400ms   a full-screen transition
```

Beyond about 500 milliseconds an animation stops feeling responsive and starts feeling like a wait. The most common mistake in first attempts is a 1-second transition, which looks impressive once and is infuriating on the fifti-eth use.

Distance scales duration. A thing moving 40 pixels should take less time than a thing moving 400. Using one duration for everything makes short movements feel sluggish and long ones feel rushed.

Easing

Linear motion looks mechanical because nothing in the physical world starts and stops instantly.

- **Ease-out** for things entering: fast at first, settling at the end. This is the default for most interface motion, because the user sees the result quickly and the settle feels natural.
- **Ease-in** for things leaving: slow start, accelerating away.
- **Ease-in-out** for things moving from one place to another on screen.
- **Linear** only for continuous motion with no start or end — a spinner, a progress bar.

```
.panel { transition: transform 250ms cubic-bezier(0.2, 0, 0,
1); }
```

That curve is a strong ease-out: quick departure, long gentle settle. It is a good default and worth keeping as a token rather than retyping.

Reduced motion is a medical accommodation

This is the part that matters most and is understood least. Large-scale motion — parallax, zooming transitions, content sliding across the whole screen, autoplaying video — can trigger genuine symptoms in people with vestibular disorders: dizziness, nausea, disorientation lasting well beyond the animation. This is not a preference about taste.

Operating systems expose the user's setting and CSS reads it:

```
@media (prefers-reduced-motion: reduce) {
  *, *::before, *::after {
    animation-duration: 0.01ms !important;
    transition-duration: 0.01ms !important;
    scroll-behavior: auto !important;
  }
}
```

That blanket rule is a reasonable safety net and a crude one. The better version is to keep the transition and change its nature: **replace movement with a fade.** A panel that fades in rather than sliding in still communicates that something appeared, still provides feedback, and does not move across the visual field. Reduced motion means less movement, not no change.

The worst offenders, in order: full-page parallax scrolling, transitions that scale an element across most of the screen, and automatically playing background video.

Anything that moves for more than five seconds needs a stop

WCAG 2.2.2 requires that any moving, blinking or scrolling content that starts automatically and lasts more than five seconds can be paused, stopped or hidden by the user.

That covers carousels, autoplaying video, animated backgrounds and marquee text. It is a requirement rather than a suggestion, it is frequently ignored, and the control is usually a single button.

The related point from the emphasis block: an autoplaying carousel is usually a symptom of an unresolved argument about what should be first, expressed as motion. Solving the argument removes the accessibility problem at the same time.

A test

Turn reduced motion on in your own operating system and use your product. Android: Settings > Accessibility > Remove animations. iOS: Settings > Accessibility > Motion > Reduce Motion. macOS: System Settings > Accessibility > Display > Reduce motion. Windows: Settings > Accessibility > Visual effects > Animation effects.

If the product becomes unusable — panels appearing with no indication, states changing with no feedback — then the motion was carrying information that nothing else carried, which is the same colour-alone failure in a different channel.

BEFORE YOU MOVE ON

A site honours prefers-reduced-motion by setting all transition durations to zero. A user with the setting enabled reports that panels now appear and disappear with no indication that anything happened. What is the better implementation?

A. Keep the durations but reduce the distance each element travels, so the motion is small enough not to trigger symptoms.

B. Reduce the durations to around 50 milliseconds rather than zero, which keeps the transition perceptible while minimising the movement.

C. Replace the movement with a fade, so the change is still signalled without anything crossing the visual field.

D. Apply the reduced-motion rule only to large-scale animations such as parallax and full-screen transitions, and leave panel transitions at their normal durations.

63 · 9 min

The accessibility requirements that are design decisions

THE ONE THING TO KEEP

Focus visibility, 400% reflow, agreement between visual and source order, persistent labels and functional alt text are design decisions rather than implementation details — and automated checkers find roughly a third of accessibility issues, so the rest needs a person with a keyboard and a free screen reader.

Not all of it is engineering

Contrast and target size have already appeared, because they are colour and spacing decisions. Several other accessibility requirements are equally design decisions rather than implementation details, and they are the ones most often discovered late.

Focus has to be visible

Every interactive element needs a visible indication when it has keyboard focus. Without it, somebody navigating by keyboard — because of a motor impairment, because they use a screen reader, or because they are fast at forms — has no idea where they are.

The common cause of failure is a single declaration:

```
:focus { outline: none; }   /* removes the indicator and pro-
vides nothing */
```

This appears in a great many stylesheets because the default outline was considered ugly. The correct move is to replace it, not remove it:

```
:focus-visible {
  outline: 2px solid currentColor;
  outline-offset: 2px;
}
```

`:focus-visible` is the important part: it applies the indicator for keyboard focus and not for mouse clicks, which removes the original aesthetic objection entirely. The indicator itself needs 3:1 contrast against what is behind it, under the non-text contrast criterion.

Zoom and reflow

Two separate requirements, often confused.

1.4.4 Resize Text (AA) — text must be resizable to 200% without loss of content or function. A layout with fixed-height containers and text in pixels fails this, because the text grows and the container does not.

1.4.10 Reflow (AA) — content must be presentable at a width equivalent to 320 CSS pixels without requiring scrolling in two directions. In practice this means 400% zoom on a 1280-pixel screen, and the test is exactly that: zoom a desktop browser to 400% and see whether you have to scroll sideways to read a line.

The design consequence is that a page has to genuinely reflow rather than merely shrink. A fixed two-column layout with a sidebar fails; a layout that stacks fails nothing. Anyone who has built the phone layout properly has already satisfied most of this, which is another argument for designing small first.

Visual order and DOM order must agree

Keyboard focus moves through the document in source order. CSS can reorder elements visually with `order`, `row-reverse`, `grid-area` or absolute positioning — and when it does, the focus order no longer matches what is on screen.

The result is a keyboard user tabbing from the top of the page to something in the middle, then back to the top, with the focus ring apparently jumping at random. It passes every visual review, because visually nothing is wrong.

The rule: if you reorder visually, reorder the source to match. Reordering is fine as a responsive convenience when the two orders agree at every breakpoint; it is a defect when they do not.

Placeholders are not labels

A form field whose only label is placeholder text has three problems: the label disappears as soon as the user types, so nobody can check what they entered; placeholder text is usually low-contrast by default and frequently fails the contrast requirement; and assistive technology handling of placeholders is inconsistent.

Every field gets a visible, persistent label. The floating-label pattern, where the placeholder moves up and becomes a label on focus, is an acceptable compromise; a field with nothing above it is not.

While you are there: the label must be clickable and tied to the field, which doubles the target size for free.

Alt text describes function, not appearance

An image's alternative text should convey what the image is doing in the page.

- A photograph illustrating an article: describe what it shows, briefly.
- A logo that links home: the alt text is the site name, because that is its function.
- An icon inside a button that already has visible text: `alt=""`, because the icon is redundant and announcing it twice is noise.
- A purely decorative image: `alt=""`, which tells assistive technology to skip it entirely.
- A chart: the alt text cannot carry the data. Provide the figures nearby, as a table or a summary sentence.

The commonest error is not missing alt text but unhelpful alt text — `image`, `photo`, the file name — which is worse than nothing because it cannot be skipped.

Automated tools find a minority

Be honest about this. Automated accessibility checkers — Lighthouse, axe, WAVE, all free — reliably catch missing alt attributes, contrast failures, missing form labels and some structural problems. Published estimates of their coverage vary and commonly land around a third of issues, and the tools' own documentation says as much.

They cannot tell you whether alt text is meaningful, whether the focus order makes sense, whether an error message is actionable, or whether the interface is usable with a screen reader. That needs a person, and the screen readers are free: NVDA on Windows, VoiceOver built into macOS and iOS, TalkBack on Android, Orca on Linux.

Spending twenty minutes navigating your own product with the keyboard alone, and another twenty with VoiceOver or NVDA, finds more than any audit tool will.

BEFORE YOU MOVE ON

A responsive layout uses CSS order to move a promotional panel above the main content on narrow screens while leaving the source order unchanged. Visual review passes. What breaks?

A. Screen readers announce elements in visual order, so the promotional panel is read twice — once in its source position and once in its visual one.

B. Keyboard focus follows source order, so the focus ring jumps between the visual top and middle of the page with no apparent logic.

C. Reordering with CSS order removes the elements from the accessibility tree, so the panel becomes invisible to assistive technology entirely.

D. The reordered panel now fails the reflow requirement at 400% zoom, because its position no longer corresponds to a single-column stacking order that can be scrolled in one direction.

64 · 9 min

The designer decides how heavy the page is

THE ONE THING TO KEEP

Page weight is set by design choices — how many images, how many font weights, how large the hero — and a 3MB image exceeds the ten-second attention limit on a connection many readers have, so choose the weight deliberately rather than letting export defaults choose it.

Performance is chosen at the design stage

A developer can compress an image. A developer cannot decide that the page does not need a full-bleed photographic hero, three web font families, an icon library and an animated background. Those are design decisions, and they set the floor for how fast the page can ever be.

This matters here because much of this readership, and much of their audience, is on a mid-range Android phone on a connection that is not the one in your office.

The arithmetic

Download time is size divided by throughput, and the numbers are unforgiving:

```
3 MB hero image on a 1.6 Mbps connection   ≈ 15 seconds
300 KB hero image on the same connection   ≈ 1.5 seconds
```

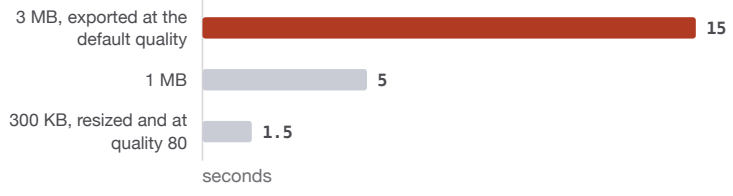
From the loading-state lesson: ten seconds is the limit of held attention. A 3MB hero image exceeds it on its own, before any other asset has loaded, on a connection that a large number of people have.

And on a metered plan the bytes are money. A visitor on a prepaid data pack is paying for your decorative background.

Images: the four decisions

Format. WebP is typically 25 to 35% smaller than JPEG at comparable quality and is supported everywhere that matters. AVIF is smaller again and slower to encode. PNG is for images that need transparency or crisp flat areas, not for photographs. SVG is for anything drawn rather than photographed, and it is usually a few kilobytes.

Download time for a hero image on a 1.6 Mbps connection



Ten seconds is the limit of held attention. A 3 MB hero exceeds it on its own, before a single other asset has begun to load.

Dimensions. Serving a 2400-pixel image to a 390-pixel phone wastes roughly 85% of the bytes. `srcset` lets the browser choose:

```
<img src="hero-800.webp"
     srcset="hero-400.webp 400w, hero-800.webp 800w, hero-
1600.webp 1600w"
     sizes="(max-width: 600px) 100vw, 800px"
     width="1600" height="900" alt="...">
```

The `width` and `height` attributes matter separately: they let the browser reserve the space before the image arrives, which prevents the layout shifting under the reader's finger.

Quality. JPEG or WebP at quality 75 to 80 is usually indistinguishable from 95 at roughly a third of the size. Exporting at maximum quality is a default nobody chose.

Whether the image is needed at all. The cheapest image is the one you did not use. A flat colour, a gradient, or a small SVG pattern costs a fraction of a photograph and frequently serves the layout better.

Free tools for all of this: **Squoosh** runs in a browser and shows a side-by-side comparison with file sizes as you move the quality slider. **GIMP** and **Krita** export with quality controls. **ImageMagick** and **cwebp** do it in batch from the command line.

Fonts

Each weight and style is a separate file. A family with four weights plus matching italics is eight files, easily 200 kilobytes in WOFF2 for Latin alone, and far more with Devanagari or CJK coverage.

The reductions, in order of effect:

1. **Use fewer weights.** Two is usually enough. This is the same restraint argument as everywhere else.
2. **Subset to the characters you need**, which for a Latin site removes a large fraction of most fonts.
3. **Use a variable font**, where one file covers a whole weight range and is often smaller than three static weights.
4. **Use the system font stack**, which costs nothing at all.

Layout shift

A specific and widely felt failure: content moves while the page is loading, and the reader taps the wrong thing because the button they were aiming at has been pushed down by an image that just arrived.

The causes are all reservations that were not made: images without dimensions, web fonts with different metrics from the fallback, advertisements or embeds inserted into the flow, content appearing above what is already visible.

The repairs are all design decisions: set aspect ratios on image containers, match fallback font metrics, reserve space for anything that will arrive later, and never insert content above the reader's current position.

The honest trade-off

This is not an argument that every page should be text on a white background. Images communicate, typefaces carry identity, and a page that is fast and says nothing has not succeeded either.

The argument is that weight is a design variable like any other, with a cost that somebody pays, and that it should be decided rather than defaulted. A hero photograph that genuinely carries the message is worth 300 kilobytes. The same photograph used because the layout looked empty is not worth anything, and on a slow connection it is worth less than nothing, because the reader left before the text arrived.

The test: open your own work on a mid-range phone, on mobile data, away from your office. Once. It changes what you design.

BEFORE YOU MOVE ON

A page serves one 2400-pixel-wide hero image at JPEG quality 95, with no width or height attributes. Which single change gives the largest reduction in bytes for the typical phone visitor?

A. Add width and height attributes so the browser can reserve space and avoid layout shift while the image loads.

B. Convert the file from JPEG to WebP, which is typically 25 to 35% smaller at comparable quality.

C. Serve a 400-pixel version to phones via srcset, since the 2400-pixel file wastes most of its bytes on a 390-pixel screen.

D. Reduce the export quality from 95 down to 80, a setting that is usually visually indistinguishable from the original at roughly a third of the resulting file size.

65 · 9 min

Bleed, trim and the things paper does

THE ONE THING TO KEEP

Trimming lands within about a millimetre of the mark, so artwork needs 3mm of bleed beyond the trim and nothing important within 3 to 5mm inside it — and paper finish, binding method and viewing distance each change the design before the file is exported.

Paper gets cut, and the cut moves

Printing happens on sheets larger than the finished piece, and the stack is trimmed with a guillotine. Paper shifts slightly in the stack, so the cut lands within about a millimetre of where it should — sometimes more.

That single fact produces the two numbers every print job needs:

- **Bleed: 3mm.** Any colour or image that should reach the edge must extend 3 millimetres beyond the trim line. Without it, a cut that lands 0.5mm outside leaves a white sliver along the edge.
- **Safe margin: 3 to 5mm inside the trim.** Nothing important — text, logos, faces — within that band, because a cut landing inside eats it.

So an A5 flyer at 148 × 210mm is set up as a 154 × 216mm document with the trim marked at 3mm in from each edge and the content kept 6 to 8mm from the paper's eventual edge.

Resolution, and the distance rule

For anything held in the hand, **300 pixels per inch at final size** is the standard. The arithmetic from the cropping lesson: divide pixel dimensions by 300 to get inches.

For large format the requirement drops, because the piece is viewed from further away and angular resolution is what matters. A roll-up banner viewed

from two metres is fine at 150dpi; a billboard seen from fifty metres is routinely produced at 10 to 20dpi at full size. A common working method is to build large-format artwork at a fraction of final size — a tenth, say — at 300dpi, and let the printer scale it.

Text and logos should be **vector** wherever possible, because vectors have no resolution and print at the press's full precision. This is why a logo supplied as a 500-pixel PNG looks soft on a poster and the same logo as an SVG or PDF does not.

Paper changes the design

Weight, in grams per square metre:

```
 80 gsm   office copier paper
120–170   flyers, leaflets, posters
250–350   business cards, postcards, covers
```

Finish matters more than weight for how the design looks. **Coated** stock has a sealed surface, so ink sits on top: colours stay saturated, blacks are deep, fine detail holds. **Uncoated** stock absorbs ink, which spreads slightly — colours are duller, blacks are greyer, and fine hairlines and small reversed-out text thicken and can fill in.

The design consequences are concrete. On uncoated stock, avoid high-contrast typefaces at small sizes, avoid white text below about 8pt on a dark ground, and expect colours to print perhaps 10 to 15% duller than the same file on coated. If you know the stock in advance, design for it.

Binding eats the inner margin

- **Saddle-stitched** booklets, folded and stapled, suffer **creep**: the inner pages push outwards as the stack thickens, so the outer margins of the centre pages get trimmed narrower. The printer usually compensates, and you should ask.
- **Perfect-bound** books are glued at the spine, and a portion of the inner margin disappears into the curve. Inner margins need to be larger than

outer ones — commonly 5 to 10mm more — which is the opposite of what an untrained layout does.

- **Spiral and wire binding** punch holes, so nothing goes within about 12mm of the bound edge.

Producing the file

The deliverable is nearly always **PDF/X**, a constrained PDF for print: fonts embedded, colours in a defined space, transparency flattened where required.

- **Scribus** is free and open source, has full colour management, bleed and trim setup, and exports proper PDF/X. It is the tool to reach for.
- **Inkscape** handles single-page vector artwork and can export PDF with bleed configured manually.
- **GIMP** is a raster editor and is the wrong tool for a page layout, though it is fine for preparing the images that go into one.
- The paid industry tools are **InDesign**, **Illustrator** and **QuarkXPress**, and **CorelDRAW**, which is what a great deal of the Indian print, signage and garment trade actually runs on.

Embed fonts, or convert type to outlines if the printer asks. Outlining guarantees the shapes and makes the file uneditable, so keep an editable master.

Ask first, every time

The single most useful habit in print work: **get the printer's specification sheet before you start the layout.**

Every printer differs on bleed, on preferred colour profile, on maximum ink coverage, on whether they want crop marks, on how they handle spot colours. Five minutes of asking prevents a re-export the night before a deadline, and printers would much rather answer the question than fix the file.

And for any run above a few hundred, pay for a proof. It costs a small amount, it is the only honest look at how the colour will land on that paper, and it is the step that experienced people never skip.

BEFORE YOU MOVE ON

A perfect-bound booklet is laid out with equal 15mm margins on all four sides of every page. When it arrives, the text on inner pages runs uncomfortably close to the spine. What was missed?

A. Saddle-stitch creep pushes the inner pages outwards as the stack thickens, narrowing the margins nearest the fold.

B. The 3mm bleed was not applied to the spine edge, so the trim has removed part of the inner margin on every page.

C. Glued binding takes part of the inner margin into the curve of the spine, so it needs to be larger than the outer margin.

D. Equal margins are a symmetry decision that suits screen layouts, and printed pages conventionally use a larger bottom margin, which would have left room at the spine.

CASE STUDY · MODULE 6

The bus booking site that worked everywhere except the bus stand

A private bus operator in Madurai running twelve routes, with a booking site built by a two-person web shop.

The site was well liked in the office. It opened with a full-width photograph of a new coach on a coastal road, taken by the owner's nephew, with the search form below it. On the shop's machines it loaded instantly and looked expensive.

Bookings through the site were about a fifth of what the operator had expected, and the counter staff had an explanation nobody had tested: people in Madurai prefer to book at the counter.

The morning at the stand

One of the developers spent a Tuesday morning at the bus stand with a second-hand phone he had bought for two thousand rupees, on a prepaid connection, standing in the open. He tried to book a ticket the way a passenger would.

The hero photograph was 3.4 megabytes. On the connection available at the stand, at around 1.6 megabits per second with a crowd of people on the same cell, the page showed a white rectangle for about fifteen seconds before anything appeared. Ten seconds is the limit of held attention. He tried it eleven times over the morning and it never once completed inside that.

When it did load, the photograph occupied the whole of the first screen, so the search form was below the fold and the page opened on a picture of a bus rather than on the thing the page was for.

The date field used a font size of 14 pixels, so tapping into it made the browser zoom the whole page and the layout jumped. The "Search buses" button sat at the top right of the form, which is the hardest point on a large phone to reach with one thumb — and at a bus stand, one hand is holding a bag.

The seat map was a table of six columns. At 360 pixels it scrolled sideways, which nobody discovered, so passengers thought only the first two seats were available.

Finally, the price and the departure time were set in a light grey that had looked refined in the office. In the sun, on that phone, neither was visible.

The decision

The list of repairs was short and none of them was difficult. The argument was about the photograph.

The developer proposed replacing the hero with a flat colour band carrying the operator's name and the search form, and moving the coach photograph further down the page, resized to 280 kilobytes and served at the size it was displayed. That takes the first screen from 3.4 megabytes to something under 200 kilobytes and puts the form at the top.

The owner objected, and the objection was not vanity. The competing operator's site had a large photograph. The coach was new, it was the main thing distinguishing the service from three cheaper operators on the same routes, and the owner's view was that a site that opened with a coloured band and a form would look like a government page and cost him the customers who were choosing on comfort.

Both costs were real. A page that loads in fifteen seconds loses the passengers who are standing at the stand with a bag and a departure in forty minutes. A page that shows nothing of the coach loses the passengers who are choosing between operators the night before, on a fast connection at home, where the photograph is the whole argument.

STOP HERE

Before turning the page: what would you have done, and what would it have cost if you were wrong? Write it down. The point of the next section is to compare.

WHAT HAPPENED

The compromise was to serve the form first and the photograph second, in that order in the page, rather than to choose between them. The first screen became a solid colour band with the operator's name, the search form, and a single line about the new fleet. The coach photograph moved immediately below, at 280 kilobytes, with width and height attributes so the layout no longer shifted when it arrived.

The rest went in the same release: form inputs at 16 pixels, the search button moved to the bottom of the form and made full width, the seat map turned into a labelled card per row, greys darkened, and the date field given a tap target of 48 pixels.

Measured at the stand on the same cheap phone two weeks later, the first screen was usable in about two seconds. Bookings through the site roughly doubled over the following two months, which the operator's counter staff attributed to a poster at the stand that went up at the same time. Nobody could separate the two effects, and the developers did not claim to.

The owner's own summary was the useful one. He said the photograph was still the reason people chose his buses, and it had simply been standing in the doorway.

The counter staff explained low bookings as a local preference for the counter. What made that explanation hard to dislodge?

It was consistent with the evidence available in the office, where the site always worked. The faults were all conditional on circumstances nobody in the office experienced: a slow shared connection, a cheap handset, sunlight, and one free hand. Browser device emulation reproduces the layout and none of those conditions, so a plausible story about customer behaviour survived until somebody stood at the bus stand and tried to book a ticket.

Which of the faults found that morning were design decisions rather than implementation defects?

Nearly all of them. The weight of the first screen was set by the decision to open with a full-bleed photograph; the input size that triggers the iOS zoom is a type decision; the position of the search button is a layout decision made against the

thumb reach map; the six-column seat table is a decision to carry a desktop presentation onto a phone; and the light greys were a palette decision. A developer can compress an image, but cannot decide the page does not need one.

Why was serving the form first and the photograph second a better answer than either side's original position?

Because the two costs applied to different readers in different conditions. The passenger at the stand needs the form inside a few seconds on a poor connection; the passenger choosing at home the night before needs to see the coach. Ordering the page so the form arrives first and the photograph follows serves both, and resizing the photograph to 280 kilobytes means the second reader pays a fraction of the original weight for the same argument.

MODULE 6 · TEST

Check what stuck

6 questions, one right answer each. This one is for you, not for a record: answers and the reason behind each are at the back. If two questions from the same module go wrong, re-read that module before the exam rather than after.

1 A full-height section's button is cut off on a real phone but not in device emulation. Which unit fixes it?

A. 100lvh, which measures the largest viewport
B. 100vh, with a fixed pixel fallback
C. 100svh
D. 100dvh, which is always the safest

2 Why must a form input on the web be set at 16 pixels or larger rather than 14?

A. 14 pixels fails the 4.5 to 1 contrast requirement for small text
B. Android renders text below 16 pixels without antialiasing
C. The 24-pixel minimum target size cannot be met below 16-pixel type
D. iOS Safari zooms the page when a smaller field is focused

3 Your breakpoints land at 540, 720 and 1180 pixels rather than at named device widths. What does that indicate?

A. They were placed where the content measurably broke
B. They will need re-tuning whenever a new phone is released
C. The layout is using container queries rather than media queries
D. They were copied from a framework's defaults

4 A product honours prefers-reduced-motion by setting every animation duration to almost zero. What is lost?

A. Nothing, because removing motion is the whole requirement
B. The feedback and continuity the motion was carrying
C. The ability to meet the five-second pause requirement
D. Hover states, which depend on transition durations to appear

5 A stylesheet contains outline: none on :focus because the default ring looked untidy. What is the correct repair?

A. Remove focus styles and rely on the visible position of the cursor

B. Restore the browser's default outline everywhere

C. Replace it with a styled indicator on :focus-visible

D. Apply the outline on hover instead, which is less intrusive

6 An A5 flyer at 148 by 210mm has a photograph running to the edge. What does the artwork need?

A. A 3mm white border, so that a misaligned cut is not visible

B. A crop mark at each corner, which is sufficient on its own

C. Nothing further, provided the photograph is exported at 300dpi

D. 3mm of artwork beyond the trim on every edge

MODULE 7

Making the decision once

The scales in the previous blocks were each a way of deciding something once instead of repeatedly. This block joins them up: named tokens, components and the variants worth having, the states every component ships in whether or not you drew them, icons as a set rather than a collection, and an honest account of what a design system costs and when a project is too small to want one.

> BY THE END OF THIS MODULE YOU CAN
>
> Convert a set of one-off screens into named tokens and a small component set, decide for any new element whether it is a variant of something existing or a new thing, specify every state a component can be in, and say at what point a project is large enough for the system to be worth its maintenance cost

66 · 8 min

A token is a decision with a name

THE ONE THING TO KEEP

A token is a named design decision, and the structure that matters is two layers — primitive values and semantic roles pointing at them — because retheming means remapping the semantic layer rather than editing every component.

The problem a name solves

A value used in forty places is forty opportunities to disagree. Somebody types 23 instead of 24. Somebody uses a slightly different blue because they picked it off a screenshot. Six months later the design contains four greys that were meant to be one, and nobody can change the brand colour without a search that will miss three instances.

A **token** is a design decision given a name, so that the decision exists in one place and every use refers to it.

```
:root {
  --space-4: 1.5rem;
  --colour-action: oklch(0.55 0.18 255);
  --radius-md: 8px;
  --text-lg: 1.25rem;
}
```

That is the entire mechanism. Everything else in this lesson is about which decisions deserve names and how to structure them.

Two layers, and why the second one matters

The important structural idea is that tokens come in two kinds, and collapsing them is the commonest mistake.

Primitive tokens are the raw scale. They describe values and nothing else.

```
--blue-100: oklch(0.93 0.04 255);
--blue-500: oklch(0.55 0.18 255);
--blue-700: oklch(0.42 0.16 255);
--grey-900: oklch(0.22 0.01 255);
```

Semantic tokens describe jobs, and they point at primitives.

```
--colour-action:        var(--blue-500);
--colour-action-hover:  var(--blue-700);
--colour-text-primary:  var(--grey-900);
--colour-surface-raised: var(--grey-50);
```

Components use only the semantic layer. Nothing in a button refers to `--blue-500`; it refers to `--colour-action`.

The payoff arrives the first time somebody asks for a dark theme, a rebrand, or a white-label version for a client. You remap the semantic layer — a dozen lines — and every component follows. Without the second layer you are editing every component, and you will miss some.

```
@media (prefers-color-scheme: dark) {
  :root {
    --colour-action:       var(--blue-300);
    --colour-text-primary: var(--grey-100);
    --colour-surface-raised: var(--grey-800);
  }
}
```

Note what changed: only the mapping. The primitives are the same colours; they are simply assigned different jobs.

What deserves a token

The categories that reliably earn one:

- **Colour** — the full ramps, plus semantic roles.

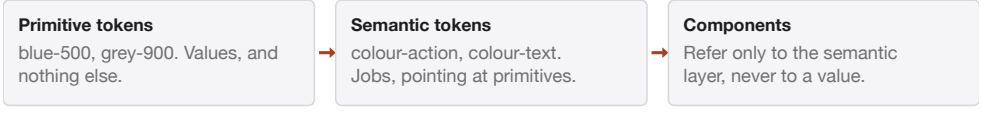
- **Spacing** — the scale from the spacing block.
- **Type** — sizes, line heights, weights, family stacks.
- **Radius** — two or three values, as discussed in the shape lesson.
- **Shadow** — a small set, all agreeing about the light direction.
- **Duration and easing** — from the motion lesson.
- **Border width** — usually just 1px and 2px, and worth naming so a focus ring is consistent.
- **Breakpoints**, if you use media queries.

The test for anything else: **will this value appear again, and would I want it to change everywhere at once?** If both are yes, name it. If either is no, leave it inline. A one-off 37px offset in a single illustration is not a token, and making it one adds a name nobody will ever look up.

What tokens do not do

Be clear about this, because token systems attract more enthusiasm than they deserve. Tokens make a design **consistent** and **changeable**. They do not make it good.

Two layers, and why the second one matters



A dark theme, a rebrand or a white-label version is then a dozen lines remapping the middle layer. Without it you edit every component, and you will miss some.

A badly proportioned layout built entirely from tokens is a badly proportioned layout that is easy to restyle. Every judgement from the earlier blocks — hierarchy, grouping, measure, contrast — still has to be made, and a token system will happily encode a bad decision and propagate it to two hundred screens with perfect consistency.

That is worth saying plainly because teams reach this stage and feel that the design work is now done. It is the opposite: the tokens are the vocabulary, and what you say with them is still entirely up to you.

The alias trap

There is one way this structure goes wrong, and it is worth naming because it looks like good practice while it happens.

Somebody creates a semantic token for every single use, so the file fills with entries like `--colour-settings-page-heading` and `--space-profile-card-top`. Each is used once. The layer has stopped being a set of roles and become a list of places.

The symptom is that adding a screen requires adding tokens. A semantic layer should be roughly stable while the number of screens grows, because roles recur — text, muted text, surface, raised surface, border, action, danger — and screens do not create new ones. If yours grows linearly with the product, the names are describing locations rather than jobs.

The repair is to ask, for each token, what would break if this were used somewhere else. If the answer is nothing, the token was a location and should be replaced by the role it actually plays.

Tokens for things that are not colours

Spacing, radius and type are the obvious ones. Three that get forgotten and cause visible inconsistency:

- **Border width.** Usually 1px and 2px. Naming them means a focus ring and a selected state agree without anybody checking.
- **Z-index.** A small named ladder — dropdown, sticky header, modal, toast — prevents the arms race where every new overlay is given 9999 and the next one 10000.
- **Maximum widths.** The measure cap from the type block, the content column, the modal width. These are decisions that recur and get re-guessed constantly.

Where they live

For most work, a single CSS file of custom properties is enough, needs no build step, and works in every browser. That file is a design system, and for a

great many projects it is the correct amount of one.

Larger setups keep tokens in a platform-neutral format — usually JSON — and generate CSS, Swift, Kotlin and Android XML from the same source, so a colour changed once reaches the web app, the iOS app and the Android app. Style Dictionary is the common open-source tool for that transformation, and it is worth knowing the pattern exists before you need it.

Do not start there. Start with the CSS file.

BEFORE YOU MOVE ON

A team defines tokens such as --blue-500 and uses them directly in every component. When a dark theme is requested, they find themselves editing each component. What structural change would have prevented it?

A. Define a parallel set of dark tokens such as --blue-500-dark and reference the correct set in each component based on the active theme.

B. Store the tokens in a platform-neutral JSON source and generate the CSS from it, so that a single change propagates automatically to every output the build produces.

C. Use a perceptually uniform colour space for the primitives, so each value can be algorithmically inverted in lightness to produce its dark equivalent.

D. A semantic layer naming roles rather than values, so components reference the role and only the mapping changes per theme.

67 · 8 min

Names that survive the next change

THE ONE THING TO KEEP

Name tokens and components by the role they play rather than by how they currently look, number ramps rather than labelling them small and large, and order words from general to specific so autocomplete groups them — then test the names on somebody who did not write them.

The name outlives the value

`--colour-red-error` is a token that was named twice and will be wrong once. When the error colour becomes orange because red tested badly, the name is a lie that stays in the codebase for years, and every new person has to learn that red means orange here.

The rule is straightforward and the discipline is not: **name by role, never by appearance.**

```
Good: --colour-danger        Bad: --colour-red
Good: --colour-surface-raised Bad: --colour-light-grey
Good: --button-primary        Bad: --button-blue
Good: --card-interactive      Bad: --card-with-shadow
```

The exception is the primitive layer from the previous lesson, which is *supposed* to describe values, because that is its job. `--blue-500` is a correct primitive name and an incorrect semantic one.

Numbers beat sizes

For any ramp, number the steps. Do not use light, medium and dark, because within a month you will need one between light and medium and there is no name for it.

```
--grey-50  --grey-100 --grey-200 ... --grey-900
--space-1 --space-2 --space-3 ... --space-8
```

The 50-to-900 convention is borrowed from font weights and has the useful property that there is always room to insert: 150 sits between 100 and 200, 250 between 200 and 300.

T-shirt sizes — xs, sm, md, lg, xl — are common and have two real problems. They run out, which produces `xxxl` and then `xxxxl`. And they have no arithmetic: you cannot reason that `--space-6` is twice `--space-3`, because `lg` and `sm` do not relate.

Where t-shirt sizes genuinely work is for a small closed set that will not grow — component sizes, where small, medium and large are the whole space and always will be.

Order the words from general to specific

A consistent word order makes tokens findable by typing, because an editor's autocomplete groups by prefix.

```
--colour-text-primary
--colour-text-secondary
--colour-text-inverse
--colour-surface-base
--colour-surface-raised
--colour-border-subtle
```

Type `--colour-text-` and every text colour appears. Name them the other way round — `--primary-text-colour` — and they scatter across the list alphabetically, which sounds trivial and is the difference between a system people use and one they work around.

The general pattern that most published systems converge on:

```
[category]-[element]-[role]-[state]

--colour-button-primary-hover
--space-card-inset
--text-heading-lg
```

You do not need every slot every time. You do need the order to be the same every time.

Naming components

Same principle, one step up. A component's name should describe what it is for, not what it looks like.

`Callout` survives being redesigned from a yellow box to a bordered panel. `YellowBox` does not. `Toast` and `Snackbar` are both appearance-neutral and both widely understood; `BottomPopup` describes a position that may change.

The harder version of this is naming by **role in the interface** rather than by visual treatment: `ActionBar` rather than `StickyFooter`, `FieldHint` rather than `SmallGreyText`.

The honest part: naming is genuinely hard

There is no consensus in the industry. Major published design systems disagree about almost every convention above — singular versus plural, hyphens versus dots, whether to include the category prefix, whether `background` or `surface` or `bg`. Reading three systems will give you three answers, all defensible.

What matters far more than which convention you pick is that you pick one and apply it everywhere. A system with an idiosyncratic but consistent naming scheme is usable. A system that follows best practice in three places and something else in a fourth is not.

Naming things that come in pairs

One recurring difficulty deserves a concrete answer. A colour used as a background needs a partner colour for the text that sits on it, and the pair has to stay together or somebody will put dark text on a dark action colour.

The convention that works is an `on-` prefix, borrowed from Material Design and now widespread:

```
--colour-action      the surface
--colour-on-action   the text and icons that sit on it
--colour-danger
--colour-on-danger
```

The name states the relationship, so the pairing survives being copied into a component by somebody who was not thinking about contrast. This is a small convention that prevents one of the most common accessibility regressions, which is a themed background whose foreground was never updated with it.

Do not encode the value in the name

A related trap: `--space-16` meaning sixteen pixels. It reads as helpful and it makes the value impossible to change, because a token called `--space-16` set to 20px is a lie, and everybody who reads the name will assume 16.

Number the **position on the scale**, not the value at that position. `--space-3` can be 16px today and 18px after a density review, and every name remains true.

The same applies to type: `--text-lg` survives a change; `--text-20` does not.

The test that actually works

Ask somebody who did not build the system to find the right token for a task. Give them a concrete job — the colour for a destructive button's hover state, the space between a label and its input — and watch.

If they find it on the first attempt, the names work. If they have to open the file and read the whole list, the names do not, and the fix is usually word order

or a missing category prefix rather than the individual names.

Do this once, early, with three people. Renaming tokens later is expensive because every usage moves, so the twenty minutes spent testing names before they spread is the best-value twenty minutes in the whole exercise.

BEFORE YOU MOVE ON

A team names its tokens --primary-text-colour, --secondary-text-colour and --raised-surface-colour. New designers repeatedly fail to find the right token and end up hard-coding values. What is the most likely cause?

A. The names describe roles rather than values, so a designer choosing a colour visually has no way to tell which token produces the shade they want.

B. The word order puts the specific part first, so related tokens scatter alphabetically instead of grouping under a shared prefix.

C. The token set is missing a primitive layer, so there is nothing for designers to fall back on when no semantic token fits their case.

D. British and American spellings of colour are both in use across the system, so autocomplete matches only half the set on any given search.

68 · 9 min

Variant, or a new thing

THE ONE THING TO KEEP

Make a variant when the structure and purpose are the same and only the appearance differs, and a new component when the content model differs — then define axes and rules rather than drawing permutations, and use slots to keep the property count from doubling.

The question that comes up every week

Somebody needs a button that is smaller, has an icon, sits on a dark background and is used for a destructive action. Is that a variant of the existing button, or a new component?

Getting this wrong in one direction produces a library of ninety components that are all nearly the same. Getting it wrong in the other produces one component with forty properties that nobody can use correctly.

The test that resolves most cases:

> ***Same structure and same purpose, different appearance or emphasis → variant.***

> ***Different structure or different content model → new component.***

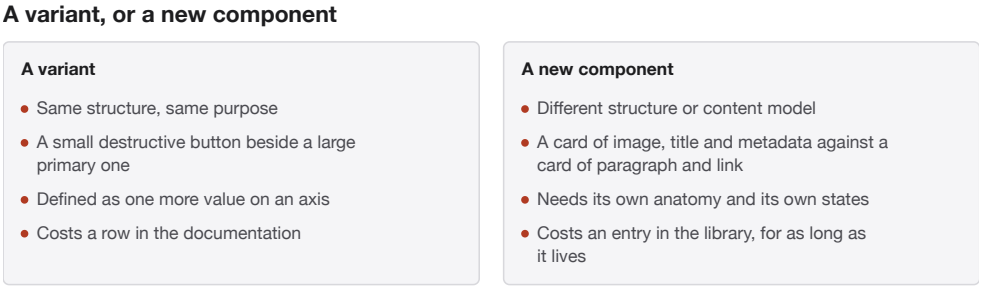
A small destructive button has the same structure as a large primary one — a container, an optional leading icon, a label — and the same purpose, which is to trigger an action. Variant.

A card that contains an image, a title and three metadata rows has a different content model from a card that contains a paragraph and a link. Those are two components, even though both are rectangles with padding.

Axes, not permutations

A button with 4 appearance variants, 3 sizes, 8 states and an optional icon has $4 \times 3 \times 8 \times 2 = 192$ combinations. You cannot draw 192 buttons, and you should not try.

What you define instead is the **axes** and the **rules for each axis**:



Four appearances, three sizes, eight states and an optional icon is 192 combinations. Define the axes and the rules, and draw only the ones that change the structure.

```
variant:  primary | secondary | ghost | danger
size:     sm (32px) | md (40px) | lg (48px)
state:    default | hover | focus | active | disabled | loading
icon:     none | leading | trailing | icon-only
```

Then document how each axis behaves independently, and draw only the combinations that are genuinely non-obvious — usually icon-only, disabled, and loading, because those change the structure rather than just the colour.

This is also the argument for keeping the axis count low. Every new boolean property doubles the space. A component with nine boolean properties has 512 configurations, most of which nobody has ever looked at, and several of which are certainly broken.

Slots keep the count down

The technique that prevents most component sprawl: give a component a **slot** — a region that accepts arbitrary content — rather than a property for every possible piece of content.

A card with a `header` slot, a `body` slot and a `footer` slot serves fifty layouts. A card with `title`, `subtitle`, `imageUrl`, `badgeText`, `badgeColour`, `metaLine1`, `metaLine2`, `ctaLabel` and `ctaHref` serves exactly one, and the next requirement adds a tenth property.

The trade-off is real: slots give up control, so a card with a free body slot can contain something that looks wrong. The usual resolution is slots for layout regions and properties for the things the system genuinely wants to govern, such as the card's own padding and elevation.

Anatomy: name the parts

For every component, name its parts. This sounds like bureaucracy and it is what makes the conversations possible.

```
Button anatomy:  container · leading icon · label · trailing
icon · focus ring
Field anatomy:   label · control · hint · error message · re-
quired marker
Card anatomy:    surface · media · header · body · footer
```

With names, a review comment can be the gap between the leading icon and the label is 4px and should be 8. Without them it is the icon thing looks squashed, which produces a different fix each time.

Signs a component should be split

- It has a property that changes several other properties' meanings.
- Half its properties are only valid when another property has a particular value.
- People keep detaching or copying it instead of using it.
- Its documentation needs the word except.
- Its name contains and.

That last one is reliable enough to be a rule of thumb. `SearchAndFilterBar` is two components.

Signs it should not have been split

The opposite failure is just as common and harder to see, because a large library feels like progress.

- Two components with the same anatomy and different colours.
- A component used exactly once, ever.
- Three components that were copied from each other and have since drifted.

The audit: list every component and its usage count. The ones used once are candidates for deletion or for merging back into whatever they were copied from. A library where a third of the entries are used once is not a system; it is a folder.

BEFORE YOU MOVE ON

A card component has accumulated properties: title, subtitle, imageUrl, badgeText, badgeColour, metaLine1, metaLine2, ctaLabel and ctaHref. A new design needs a card with a chart in place of the image. What is the better response?

A. Add a chartData property alongside imageUrl, with a rule that exactly one of the two must be supplied.

B. Create a ChartCard component, so each card type has a property set matched to its own content.

C. Replace the content properties with a media slot and a body slot that accept arbitrary content.

D. Keep imageUrl and render the chart to an image server-side, so the existing component continues to work without modification.

69 · 9 min

Every state you did not draw still ships

THE ONE THING TO KEEP

A component ships in every state whether or not it was designed, hover and focus serve different audiences and never substitute for each other, and a disabled control that cannot be read or explained should usually be an enabled control that explains what is missing on activation.

The list

A component exists in more conditions than the one in the mockup. For anything interactive:

```
default         resting
hover           pointer over it (does not exist on touch)
focus-visible   keyboard focus (does not exist for mouse users)
active          being pressed
disabled        not available
loading         working
selected        currently chosen, in a set
error           in an invalid condition
read-only       shown but not editable
visited         for links
```

Every one of these ships whether or not somebody designed it. Undesigned, they ship as the browser's defaults, which are inconsistent across platforms, or as nothing at all, which means the interface does not respond to being used.

Hover and focus are different audiences

A point that gets conflated constantly. **Hover does not exist on touch devices.** Any information revealed only on hover is invisible to most of your users, which is why tooltips as the only explanation of an icon is a failure and why hover-only menus are a recurring accessibility complaint.

Focus does not appear for mouse users, and should not — a focus ring after every click was the original reason people removed focus styles. `:focus-visible` gives the ring to keyboard users and not to mouse clicks, which resolves the conflict entirely and is the correct selector to use.

So the two states serve different people, need different treatments, and neither substitutes for the other.

Disabled is the one that goes wrong

Disabled controls are conventionally shown at low contrast to signal unavailability, and WCAG explicitly exempts disabled controls from the contrast requirements. That exemption is real, it is also widely criticised, and it produces a genuine problem: a disabled button that nobody can read is a control whose label cannot be checked, by anybody, including people with perfect vision in bright light.

The deeper problem is that disabled controls communicate nothing about **why**. A Submit button that is greyed out tells the user they cannot proceed and not what to fix. They then hunt.

Three better patterns, in order of how often they apply:

1. **Keep the control enabled and explain on activation.** The user presses Submit, and the form shows what is missing and moves focus there. This is the strongest pattern for forms and it is now common.
2. **Say why next to the control.** If it must be disabled, put the reason in text beside it, not in a tooltip that touch users cannot reach.
3. **Hide it entirely**, where the action is not merely unavailable but irrelevant in this context.

If you do keep a disabled state, make it legible anyway. The exemption permits low contrast; nothing requires it.

Loading, and the width problem

A button that replaces its label with a spinner changes width, which shifts everything beside it. The user's finger is already moving.

Fix it by reserving the width: keep the label in place at reduced opacity and overlay the spinner, or set an explicit minimum width so the shortest state and the longest state are the same size. This is the layout-shift point from the previous block, at component scale.

Also decide what the button does while loading. The correct answer is almost always that it stops accepting presses, because the common failure is a user tapping three times and submitting three orders.

Error is a component state, not a screen

An invalid field needs: a changed border, an icon, a message, and — from the colour block — none of those carrying the meaning alone. The message goes next to the field, not at the top of the form, because a message at the top makes the user hunt for which field it refers to.

The field should also keep what the user typed. Clearing an invalid field is the single most disliked behaviour in form design, and it is still common.

Specify them together

The practical method is a small table of decisions per component, drawn once:

```
Button, primary
  default   surface: --colour-action        label: --colour-on-
action
  hover     surface: --colour-action-hover
  active    surface: --colour-action-active, translate-y 1px
  focus     ring: 2px --colour-focus, offset 2px
  disabled  surface: --colour-action-muted, cursor default
  loading   label at 0% opacity, spinner centred, width locked
```

Six rows. It takes ten minutes, it removes a week of small inconsistencies later, and every value in it is a token from the earlier lesson rather than a number.

BEFORE YOU MOVE ON

A signup form disables Submit until every field is valid. Users report getting stuck without knowing why. The team proposes adding a tooltip on the disabled button explaining what is missing. What is the flaw?

A. Disabled controls are exempt from contrast requirements, so the tooltip's text would not be required to meet 4.5:1 and would be unreadable for the same users who are already stuck.

B. A tooltip requires hover, so touch users — most of them — cannot reach the explanation at all.

C. The explanation belongs at the top of the form rather than on the button, so the user sees every outstanding problem in one place before acting.

D. Disabled buttons should never be used in forms under any circumstances, so the tooltip is treating a symptom of a pattern that has to be removed entirely.

70 · 8 min

Icons are a set, not a collection

THE ONE THING TO KEEP

Use one icon set, because mismatched grids, stroke weights and detail levels are visible even when nobody can name them — and label icons wherever there is room, since comprehension without a label is poor for everything outside a small well-learned set.

Why mixed icons look wrong

Download three icons from three sources and put them in a row. They will not match, and the mismatch will be visible to people who cannot name any of the reasons.

The reasons are specific and measurable:

- **Different grids.** One set is drawn on 24 pixels, another on 20, another on 16 and scaled up.
- **Different stroke weights.** 1.5px against 2px is a visible difference in apparent weight.
- **Different corner treatments.** Rounded caps and joins against square ones.
- **Different levels of detail.** One set draws a camera with a lens and a flash, another with a rectangle and a circle.
- **Different optical sizing.** One set lets a circle exceed the nominal grid so it looks the same size as a square; another does not.

Use one set. This is the single decision that makes an interface's icons look designed rather than assembled, and it costs nothing.

The grid and the stroke

The common convention is a 24 × 24 grid with a 20 × 20 live area, leaving 2 pixels of padding on each side, and a 2-pixel stroke. At 16 pixels the stroke usually drops to 1.5.

Two consequences worth knowing:

Optical size overrides the grid. As covered in the alignment lesson, a circle drawn to exactly 20 pixels looks smaller than a square drawn to exactly 20 pixels. Well-made sets let round shapes exceed the live area by a few per cent. If you draw your own icons, do the same, and judge by eye rather than by the guide.

Stroke weight should relate to the text it sits beside. A 1px icon next to 600-weight text looks anaemic. The icon's apparent weight should be close to the stem weight of the accompanying type, which usually means 1.5 to 2 pixels at body size and heavier beside a bold label.

Free sets worth knowing

All of these are genuinely free for commercial use, and all are permissively licensed — but check for yourself rather than taking this list on trust, because

licences change.

- **Lucide** — a community fork of Feather, ISC licence, clean and consistent.
- **Phosphor** — MIT, six weights including a filled variant, unusually large.
- **Material Symbols** — Apache 2.0, a variable font with weight, fill and optical size axes.
- **Tabler Icons** — MIT, very large set on a consistent grid.
- **Remix Icon** — Apache 2.0, includes line and fill pairs.
- **Bootstrap Icons** — MIT.

One caution that catches people out: several popular sets described as free carry attribution requirements. Font Awesome's free icons are licensed under CC BY 4.0, which requires attribution. That is a mild condition and it is still a condition, and it applies to work you hand a client.

Delivering them

Inline SVG is usually correct: it can inherit `currentColor`, it can be styled by CSS, and it adds no network request.

```
<svg width="24" height="24" viewBox="0 0 24 24" fill="none"
     stroke="currentColor" stroke-width="2" aria-hidden="true">
  <path d="M5 12h14M12 5l7 7-7 7"/>
</svg>
```

`stroke="currentColor"` is the important part: the icon takes the colour of the surrounding text automatically, including in dark mode and in every state.

Icon fonts were the standard approach for years and have real problems: they fail in a specific and ugly way when the font does not load, screen readers sometimes announce the private-use character, and they cannot be multi-coloured. They still work, and there is rarely a reason to choose one now.

Icons are rarely self-explanatory

This is the honest part, and it contradicts a lot of practice.

Studies of icon comprehension consistently find that recognition without a label is poor for most icons. A small set is genuinely well understood across populations — a magnifier for search, a house for home, a printer, a play triangle, a rubbish bin for delete, a plus for add. Beyond that, comprehension falls off quickly, and icons for abstract concepts such as settings, share, filter or archive are guessed rather than known, with the guesses varying by platform and by age.

The common hamburger menu is the frequently cited case: it is now widely learned, and studies through the 2010s repeatedly found that labelling it Menu improved discovery.

The design conclusion:

1. **Label icons wherever there is room.** Icon plus text beats either alone.
2. **Where an icon must stand alone**, it needs an accessible name regardless — `aria-label` on the button — because a screen reader announces nothing for an unlabelled glyph.
3. **Decorative icons beside text get** `aria-hidden="true"`, so the label is not announced twice.
4. **Do not invent an icon for an abstract concept** and expect it to be understood. It will not be.

BEFORE YOU MOVE ON

An interface uses icon-only buttons throughout, with tooltips on hover giving each one's name. A usability test finds users guessing wrong about several. The team's proposal is to redesign the icons to be clearer. Why is this unlikely to work?

A. Recognition without a label is poor for abstract concepts whatever the drawing, so a clearer icon does not produce a shared meaning.

B. The tooltips are the real problem, since they require hover and are unreachable on touch, so moving them to persistent text beside each icon resolves the comprehension issue.

C. The icons were probably drawn on inconsistent grids with mismatched stroke weights, and the resulting visual inconsistency is what users are struggling to interpret.

D. Icon-only buttons lack accessible names, so assistive technology announces nothing and the test participants using it had no information to work from at all.

71 · 8 min

The afternoon version

THE ONE THING TO KEEP

A working design system for a small project is about thirty tokens and six components, all derived from decisions already made in the earlier blocks — and the step that proves it is rebuilding one real screen using nothing else.

Most of the value, very little of the work

Published design systems are enormous because they come from companies with dedicated teams and thousands of screens. The version that helps a solo designer or a three-person team is about one per cent of that, and it takes an afternoon.

Here is the whole thing.

Step 1: the token file

One CSS file. No build step, no tooling, works everywhere.

```
:root {
  /* neutrals */
  --grey-50:  oklch(0.98 0.004 255);
  --grey-100: oklch(0.95 0.006 255);
  --grey-200: oklch(0.90 0.008 255);
  --grey-400: oklch(0.72 0.010 255);
  --grey-600: oklch(0.54 0.012 255);
  --grey-800: oklch(0.32 0.010 255);
  --grey-900: oklch(0.20 0.008 255);

  /* brand */
  --brand-300: oklch(0.75 0.12 255);
  --brand-500: oklch(0.55 0.18 255);
  --brand-700: oklch(0.42 0.16 255);

  /* semantic */
  --colour-text:       var(--grey-900);
  --colour-text-muted: var(--grey-600);
  --colour-surface:    white;
  --colour-raised:     var(--grey-50);
  --colour-border:     var(--grey-200);
  --colour-action:     var(--brand-500);
  --colour-danger:     oklch(0.52 0.19 28);

  /* space */
  --space-1: 0.25rem; --space-2: 0.5rem;  --space-3: 1rem;
  --space-4: 1.5rem;  --space-5: 2rem;    --space-6: 3rem;

  /* type */
  --text-sm: 0.8125rem; --text-md: 1rem;
  --text-lg: 1.25rem;   --text-xl: 1.75rem; --text-2xl: 2.5rem;
  --leading-tight: 1.2; --leading-normal: 1.55;

  /* form */
  --radius-sm: 4px; --radius-md: 8px; --radius-full: 999px;
  --shadow-md: 0 4px 12px rgb(0 0 0 / 0.08);
  --dur-fast: 150ms; --ease-out: cubic-bezier(0.2, 0, 0, 1);
}
```

That is roughly thirty decisions. Every one of them came from an earlier block in this course: the lightness ramp, the spacing scale, the modular type scale, two radius values, one shadow agreeing about the light, one duration.

Step 2: six components

Not sixty. These six cover the overwhelming majority of what a small product or a set of marketing pages needs:

1. **Button** — three variants, two sizes, all six states.
2. **Text field** — with label, hint and error.
3. **Card** — surface, padding, optional media slot.
4. **Heading set** — three levels, with space above and below defined.
5. **Link** — in-text and standalone.
6. **Badge or tag** — because every product grows one.

Specify each with the state table from the previous lesson. Two hours.

Step 3: rebuild one real screen using only these

This is the step that makes the exercise worth doing, and the step people skip.

Take an existing screen and rebuild it using nothing but the tokens and the six components. You will immediately discover:

- Values you need that are not in the scale, which are either missing tokens or accidental values in the original.
- Components you need that are not in the six, which tells you what the seventh should be.
- Places where the original screen used four greys where the system offers two, and looks no worse for the reduction.

The third finding is the common one, and it is the system paying for itself on its first day.

The tools, and what they cost

For the tokens and components: a text editor and a browser. Nothing else is required, and a single HTML page showing every component in every state is a perfectly respectable component gallery.

For visual design files:

- **Penpot** — open source, free, self-hostable, with components and design tokens. The genuine free alternative.
- **Figma** — free tier with limits on files and version history; the industry default and what job advertisements name.
- **Inkscape** — excellent for vector artwork, not a UI design tool.
- **Sketch** is paid and macOS only; **Adobe XD** has been discontinued.

For a rendered component gallery: Storybook is the named industry tool and is free and open source. A hand-written HTML page does the same job for a small project with none of the setup.

Step 4: write down the six rules you keep

One short page, and it matters more than it looks. The tokens say what values exist; this says how they are used.

A realistic example:

```
1. Body text is --text-md at --leading-normal, capped at 34em.
2. One primary action per view. Everything else is secondary or
ghost.
3. Space between groups is at least three times the space in-
side them.
4. Interactive targets are at least 44px, with 8px between
neighbours.
5. Any colour-carried meaning also has an icon or a word.
6. Motion is 150-300ms, ease-out, and respects reduced-motion.
```

Every one of those is a decision from an earlier block, written down so it does not have to be re-argued. Six lines is the right length: a page of forty rules will not be read, and these six catch most of what goes wrong.

What this does not require

No build step. No framework. No component library dependency. No design tool subscription.

That is worth stating plainly, because the visible ecosystem around design systems is expensive and elaborate, and it can make the whole activity look like something that needs funding. It does not. A stylesheet of custom properties, six components written in plain HTML and CSS, an HTML page showing them, and six rules in a text file is a complete and genuinely useful design system, and it runs on a laptop that cannot install anything.

What you have not built

Be clear about the boundary. An afternoon produces tokens and six components. It does not produce contribution guidelines, a versioning policy, a deprecation process, cross-platform token generation, usage analytics or a governance model.

You do not need any of those yet, and the last lesson in this block is about recognising when you do.

BEFORE YOU MOVE ON

A solo designer builds a token file and six components in an afternoon, then rebuilds an existing screen with them and finds the screen needs five greys where the system offers three. What is the most useful interpretation?

A. The system is too restrictive for real work, and the neutral ramp should be expanded until it covers every value the existing screens use.

B. The two extra greys are probably accidental values, and the screen should be checked to see whether it looks worse with three.

C. The ramp was built with too few steps, and a full 50-to-900 ramp of nine values would have avoided the mismatch entirely.

D. The rebuild should be abandoned, since a system proven inadequate on its first real screen will fail on the remainder and needs more design work before it is applied anywhere.

72 · 8 min

Specifications that survive contact

THE ONE THING TO KEEP

Hand over token names rather than pixel measurements, specify the rules for long content and narrow containers rather than only providing pictures, and keep documentation beside the code or generated from it, because stale documentation is worse than none.

Redlines are dead labour

The traditional handoff was an annotated screenshot: arrows and numbers marking every margin, every size, every hex code. It takes hours, it is obsolete the moment anything changes, and the numbers get transcribed by hand into code where they become untraceable magic values.

If you have tokens, hand over token names instead.

```
Bad:   padding 24px, gap 8px, radius 8px, #2F6DF6, 16px/1.55
Good:  padding var(--space-4), gap var(--space-2),
       radius var(--radius-md), background var(--colour-
action),
       text var(--text-md)/var(--leading-normal)
```

The second version is shorter to write, unambiguous, and survives a rebrand without being rewritten. It also makes deviations visible: if a developer has to ask what token 18px is, the answer is that 18px is a mistake.

What a component specification must contain

Six sections, none of them long:

1. **Anatomy** — the named parts, from the components lesson.
2. **Variants and sizes** — the axes, with the token values for each.

3. **States** — the table from the states lesson.
4. **Spacing** — internal padding and gaps, as tokens. Not external margins, which belong to the layout.
5. **Content rules** — maximum label length, what happens when it is exceeded, whether it wraps or truncates, what the empty case is.
6. **Accessibility notes** — the accessible name, the keyboard behaviour, the focus treatment, anything that is not obvious from the visual.

The fifth is the one most often missing and most often needed. Truncate at one line with an ellipsis is a design decision, and if nobody makes it, the implementation will make it differently in three places.

Write the rules, not the pictures

A picture of a component in one configuration is not a specification, because implementation always encounters a case the picture did not show. What is the longest label? What if there is no icon? What happens at 320 pixels?

Rules answer those. Pictures do not, and a beautiful specification made entirely of pictures generates a constant stream of questions that the designer then answers ad hoc, inconsistently, over the following weeks.

A useful heuristic: for every visual you provide, write one sentence about what happens when the content is longer than shown, and one about what happens when the container is narrower.

Documentation goes stale, and stale documentation lies

This is the honest limitation, and it is severe. A specification that is out of date is worse than no specification, because it is wrong with authority. Somebody will build from it, and the discrepancy will be found in review, and the time spent is worse than if they had asked.

Two things keep documentation alive, and only two:

Proximity. Documentation that lives beside the code gets updated when the code changes. Documentation in a separate tool that requires a different login does not.

Use. Documentation that is the actual source — a component gallery rendered from the real components, so the examples cannot drift — cannot go stale, because it is generated from the thing it documents. Storybook does this and is free; a hand-written HTML page that imports the real stylesheet does most of it.

A static PDF of specifications is stale within a month, always. If that is the only option available, keep it to the decisions rather than the details, because decisions age more slowly than pixel values.

The handoff conversation

The document is not the handoff. A fifteen-minute conversation covering three questions prevents most of the problems a document cannot:

1. **What is fixed and what is flexible?** Which values are load-bearing and which were chosen because something had to be chosen. Developers routinely preserve an irrelevant 3px offset with great care while altering the one measurement that mattered, because nothing told them which was which.
2. **What happens at the edges?** The longest content, the narrowest screen, the slowest connection, the error.
3. **What should you come back to me about?** Set the threshold explicitly, or you will either be consulted about everything or about nothing.

Keep it short

A component specification that runs to eight pages will not be read. One that fits on a screen will.

The discipline is to document the decisions somebody could not infer, and to leave out everything they could. A button is a rectangle with a label in it; nobody needs that written down. What the button does when the label is forty characters long is not inferable, and that is what the page is for.

BEFORE YOU MOVE ON

A designer hands a developer an annotated screenshot with every measurement in pixels and every colour as a hex code. The build matches the screenshot exactly. Six months later a rebrand changes the accent colour and the component does not follow. Why?

A. The hex codes were transcribed as literal values rather than references to a token, so nothing connects them to the brand colour that changed.

B. The screenshot documented one configuration only, so the developer had no rules for what should change under a rebrand and reasonably left the values fixed.

C. The specification lacked a state table, so the accent colour was only applied to the default state and the rebrand updated states that had never been implemented.

D. Annotated screenshots go stale within a month, so the documentation the developer built from no longer described the component by the time the rebrand happened.

73 · 8 min

Consistency is not uniformity

THE ONE THING TO KEEP

Consistency means the same things look the same while uniformity means everything does, so break the system only for moments that must not feel routine, genuinely new content, a different context or a timed experiment — and require every exception to be resolved rather than merely recorded.

Two things that sound alike

Consistency means the same things look the same. A destructive action is always the same colour; a card always has the same radius; a section heading always has the same space above it. This is what lets a reader stop examining,

and it is the property that makes every deviation meaningful, as the repetition lesson established.

Uniformity means everything looks the same. It is what you get when a system is applied without judgement, and it produces work that is competent, coherent and completely inert — no hierarchy between screens, no sense that any moment matters more than another, nothing an eye wants to land on.

A system is meant to produce the first. Left unsupervised, it produces the second.

When breaking the system is correct

Four situations, and they are narrower than most people would like:

1. A moment that must not feel routine. An irreversible deletion. A payment confirmation. The completion of something that took the user an hour. These should look different from the ordinary flow, because feeling the same as everything else is precisely the failure.

2. A genuinely new content type. The system has no pattern because nothing like this has existed before. Build the new thing, use it, and then decide whether it becomes a pattern.

3. A different audience or context. A marketing landing page and the product's settings screen have different jobs. Sharing tokens is right; sharing every component is not. Most organisations run a looser system for marketing and a stricter one for the product, and that is a reasonable arrangement rather than a failure of discipline.

4. A deliberate experiment. Testing whether a different treatment performs better. This is a break with an expiry date attached.

What is not on the list: this screen felt boring, and the designer wanted to do something interesting. That is the uniformity complaint, and the answer is usually that the hierarchy is flat rather than that the system is wrong.

How to break it without dissolving it

Three conditions, and all three matter:

- **Deliberately.** Somebody decided, and can say why.
- **Documented.** The exception is recorded where the system lives, so the next person knows it was a choice.
- **Resolved.** Within some agreed period, the exception either becomes a pattern or is removed. Exceptions that are neither are how a system decays.

That third condition is the one that gets dropped and the one that determines whether the system survives two years.

The signs of drift

Systems fail gradually and the symptoms are recognisable:

- People **copy and detach** components rather than using them.
- A component's usage count stops growing while the number of screens grows.
- Values appear that are not in the scale, in increasing numbers.
- Somebody maintains a private stylesheet of overrides.
- New joiners ask which of these two patterns should I use, and get different answers.

Each of these is information rather than misbehaviour. People route around a system when it costs them more than it saves, which means a system nobody uses is usually a system that was too strict, too incomplete or too hard to find — not a team that lacks discipline.

The response to drift is an audit and a conversation, not an instruction.

Over-strict is a real failure mode

This is worth saying because the literature is overwhelmingly about the opposite risk. A system that forbids everything produces designers who spend their effort negotiating rather than designing, and work that is worse than it would

have been with no system at all, because the constraints were binding in the wrong places.

The symptom is that people are asking permission for things nobody should need permission for — a one-off illustration, the exact crop of a photograph, the wording of a label.

A good system constrains the things that benefit from consistency and says nothing about the rest. Spacing, colour, type and component behaviour benefit enormously. What a photograph depicts, how an illustration is drawn, and what a headline says do not, and a system that governs them has overreached.

The periodic audit

Once a quarter, or whenever it feels wrong, do the three audits from earlier blocks together: every distinct spacing value, every distinct type size, every distinct colour, plus every component and its usage count.

The list is the conversation. Values that appear once get deleted or adopted. Components used once get merged or removed. Exceptions get resolved.

An hour, four times a year, is what keeps a system from becoming a document describing a product that no longer exists.

BEFORE YOU MOVE ON

A design system is well documented, but designers increasingly copy and detach components rather than using them. The lead proposes stricter enforcement in review. What is the more likely diagnosis?

A. The team is producing uniformity rather than consistency, and detaching is how designers reintroduce variation the system has removed.
B. The documentation has gone stale, so designers are detaching because the documented behaviour no longer matches what the components actually do.
C. Exceptions have not been resolved into patterns, so the library is missing the components those exceptions should have become and designers are rebuilding them each time.
D. The system costs more than it saves for these cases, so detaching is information about a gap rather than a discipline problem.

74 · 8 min

When not to build one

THE ONE THING TO KEEP

Build the three scales always, and everything beyond them only once the same decision has been made three times or a second person has joined — because a system is a product with ongoing maintenance cost, and most published advice is written for organisations a hundred times larger than the project in front of you.

Systems slow down the first three screens

Building tokens and components before you know what the product is means deciding things you cannot yet decide. The first three screens take longer, and several of the decisions turn out to be wrong, and unpicking them costs more than making them fresh would have.

This is the part that published design-system advice rarely says, because that advice comes almost entirely from organisations with dedicated system teams, hundreds of screens and dozens of contributors. Their conclusions are correct for their situation and are routinely applied to situations a hundred times smaller.

Where the breakeven sits

Two triggers, either of which is enough:

The same decision has been made three times. Three buttons drawn separately, three card layouts, three spacing choices for the same relationship. Three is the point at which the next one is cheaper to systematise than to repeat — before that, you are guessing at what the pattern will be.

A second person joins. The moment more than one person is making the decisions, the cost of disagreement exceeds the cost of writing them down. A solo designer holds the system in their head and it works; two people holding two systems in two heads does not.

Before either trigger, the correct amount of system is: a spacing scale, a type scale and a colour ramp, which cost half an hour and pay back immediately. Those three are always worth it. Components, documentation and governance are not, yet.

Projects that should not have one

- **A one-off poster, flyer or invitation.** It will be made once.
- **A single landing page.** Consistency within one page is achieved by looking at the page.
- **An unvalidated product**, where the screens will be thrown away. Systematising a design that is about to be replaced is work performed on a thing that will not exist.
- **A prototype meant to answer a question.** Its job is to be tested, not to be maintained.

In each of these, the three scales are still worth having and everything beyond them is overhead.

A system is a product with users

The cost that surprises people is not building it but keeping it. A design system has users — your own team — and like any product it needs:

- **An owner.** Somebody whose job includes it. Without one it becomes nobody's, which is the commonest way systems die.
- **A way to request changes**, and a decision about who says yes.
- **A changelog**, so people can tell what changed and when.
- **A deprecation path**, because components do get replaced and the old ones have to be removed rather than merely discouraged.
- **Migration work**, which is real and is usually underestimated. Changing a token's value touches every screen using it, and somebody has to check them.

A rough figure for a small team: maintaining a modest system costs the equivalent of a day or two a month, indefinitely. That is affordable and it is not free, and budgeting zero for it is why so many systems are abandoned within a year of launch.

The failure that ends most systems

One person builds it, in their own time, because they care. It is good. They leave, or move team, or get pulled onto something urgent.

Nobody else understands the conventions, changes stop being made, the product moves on, and within six months the system describes a product that no longer exists. People stop consulting it, which is rational, and then it is genuinely dead.

The defence is unglamorous: keep it small enough that somebody else can hold it, document the decisions rather than only the outputs, and make sure at least two people have made changes to it.

The opposite cost

None of this is an argument against systems. At two hundred screens with no system, every change is a hunt, every new screen is an argument, the product looks like five products, and onboarding a designer takes months.

That cost is larger than the maintenance cost, considerably, and it arrives whether or not anybody budgeted for it.

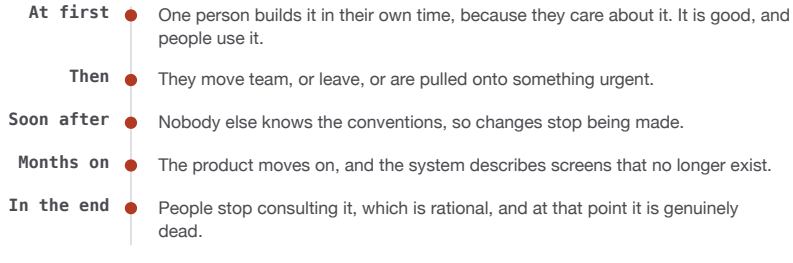
The judgement

Match the system to the project:

```
one artefact            three scales, nothing else
one page                three scales
a small product         three scales + six components + one
page of notes
a product with a team   the above + a gallery + an owner + a
changelog
many products           tokens in a neutral format, governance,
versioning
```

Most readers of this course are in the first three rows, and most published advice is written for the last. Knowing which row you are in is the whole skill, and building a row-five system for a row-two project is a common and expensive mistake that looks like professionalism while it is happening.

How a design system usually dies



The defence is unglamorous: keep it small enough that somebody else can hold it, write down the decisions rather than only the outputs, and make sure two people have changed it.

BEFORE YOU MOVE ON

A two-person studio is building a prototype to test whether a product idea is worth pursuing. They spend the first week building tokens, twelve components and a documented gallery. What is the strongest objection?

A. Twelve components is too many for a first pass, and the six core components would have delivered most of the value in a fraction of the time.

B. A two-person team has already crossed the second-person trigger, so a system is justified, but it should have been built after the first three screens rather than before them.

C. The screens exist to answer a question and will be discarded, so systematising them is work on something that will not exist.

D. A system needs an owner, a changelog and a deprecation path, and a two-person studio cannot sustain a day or two a month of maintenance alongside product work.

CASE STUDY · MODULE 7

The milk round app, and the system built three weeks too early

A two-person product team in Pune building a delivery app for a dairy co-operative with about 400 daily rounds.

The first version of the app had four screens: the round list, a customer, a delivery entry, and a day summary. It was built in five weeks by one designer and one developer, and it worked.

The designer had read a good deal about design systems, and before the second phase began she proposed three weeks to build one: a full token set, a component library of eighteen components, a documented naming convention, and a rendered gallery. The developer was in favour. The founder asked what it would delay, and the answer was the route-planning screens, which were the reason the co-operative had commissioned the work.

The argument, stated fairly

The case for building it then was that the product was about to grow from four screens to perhaps thirty, and that every screen built without a system would have to be revisited. The designer had already seen the early signs: two card treatments that were nearly the same, three different greys for secondary text, and a button whose padding differed by two pixels between two screens because it had been drawn twice.

The case against was that nobody yet knew what the route-planning screens needed. A route screen has a map, a reorderable list, a time window per stop, and an offline state, and none of those had an equivalent in the four screens that existed. Eighteen components designed before that work would contain several that were guesses, and unpicking a guessed component after it has been used in ten places is more expensive than not having had it.

The founder's own cost was the plainest. Three weeks was a third of the remaining budget for the phase, spent on something the co-operative would never see and could not evaluate.

The compromise, and the trigger that settled it

What they agreed was narrower than either position. They spent half a day, not three weeks, on the three things that were already decided and already being got wrong: a spacing scale, a type scale and a colour ramp with a small semantic layer on top. Those three came out of decisions the design had already made, so nothing in them was a guess.

Everything else waited on a rule they wrote down: a component gets built when the same decision has been made three times, or when a second person starts making it.

A third thing was agreed at the same time and took twenty minutes: one page listing the six rules the team kept, such as one primary action per view, and space between groups at least three times the space inside them. None of those were new decisions either. They were decisions the four existing screens had already made, written down so that they did not have to be re-argued when the screens multiplied.

Both triggers fired within two months, and they fired for different things. The button reached three variants across the route screens and became a real component with a state table. The card did not — the three card-like things turned out to have different content models and stayed separate, which the designer said later she would have merged into one over-configured component if she had systematised in week one. Then a contract designer joined for the offline states, and within a week the naming convention stopped being optional, because two people were inventing names for the same roles.

> **STOP HERE**
>
> Before turning the page: what would you have done, and what would it have cost if you were wrong? Write it down. The point of the next section is to compare.

WHAT HAPPENED

The route-planning screens shipped on time, and the half-day scales did most of what the three-week system would have done for consistency. Of the eigh-

teen components originally proposed, eleven were eventually built. Three of the remaining seven were never needed, and four turned out to be different from the way they had been sketched, which was the concrete version of the argument about guessing.

The token file stayed one CSS file of custom properties with no build step for the whole of the first year. It was replaced by a generated set only when an Android app appeared and the same colours had to reach two platforms.

The cost that surprised them was maintenance rather than construction. Once eleven components existed, keeping them cost the equivalent of a day or two a month, which nobody had budgeted and which came out of the designer's time. The founder's later view was that the half-day decision had been right and the number the team had been missing all along was not the cost of building a system but the cost of owning one.

The designer's evidence for building early — two card treatments, three greys, a button drawn twice — was real. Why did it not justify the three weeks?

Those faults are exactly what the three scales fix, and the three scales cost half a day. The expensive part of the proposal was the eighteen components and the documentation around them, and nothing in the evidence spoke to those. The correct amount of system before the triggers fire is a spacing scale, a type scale and a colour ramp, because those encode decisions already made rather than decisions not yet made.

Why did the card turn out differently from the button?

The button variants had the same structure and the same purpose and differed only in appearance and emphasis, which is the definition of a variant axis, so one component with rules served all three. The three card-like things had different content models — different parts, in different arrangements — which makes them separate components however similar their outlines are. Building a card component in week one would have merged three content models into one over-configured thing with a property for every field.

What does the team's later finding about maintenance add to the decision they made?

It shows the relevant cost was never only the construction cost. A system is a product with users, and once eleven components existed somebody had to own them, answer questions, deprecate things and migrate usages, at roughly a day or two a month indefinitely. Budgeting zero for that is the commonest way systems are abandoned, and it is an argument for building the smallest system that removes the repeated decisions rather than the largest one the team can imagine.

MODULE 7 · TEST

Check what stuck

6 questions, one right answer each. This one is for you, not for a record: answers and the reason behind each are at the back. If two questions from the same module go wrong, re-read that module before the exam rather than after.

1 Why should a button refer to a token called colour-action rather than to one called blue-500?

A. Because a retheme then remaps one layer instead of every component
B. Because semantic names compress better in a stylesheet
C. Because primitives cannot be used inside a var() declaration
D. Because primitive token names are reserved for the build tool to generate

2 A token file grows a new semantic entry for each screen: colour-settings-page-heading, space-profile-card-top. What has gone wrong?

A. The naming lacks a state slot, so every use needs its own entry
B. The names describe locations rather than roles
C. The file now needs a build step to stay maintainable
D. The primitives are missing, so the semantics point at raw values

3 A card of image, title and three metadata rows sits beside a card of a paragraph and a link. One component or two?

A. Two, but only if their paddings differ as well
B. One, with a variant property, because both are padded rectangles
C. Two, because the content models differ
D. One, with slots, because slots remove the need to decide

4 A Submit button is greyed out until the form is valid. Why is that usually the weaker pattern?

A. Disabled controls cannot receive keyboard focus in any browser
B. Greying out fails because grey is a colour-only signal
C. Disabled controls are exempt from contrast rules, which is a legal risk
D. It tells the user they cannot proceed but not what to fix

5 During handoff a developer asks which token corresponds to 18 pixels. What does the question tell you?

A. The 18 pixels was a mistake, because it is not on the scale
B. The developer is measuring a screenshot instead of opening the file
C. The specification should have given hex values rather than token names
D. The scale needs an extra step between 16 and 24

6 A solo designer is making a one-off flyer that will be printed once. How much system is worth building?

A. A full token file plus a documented naming convention
B. A spacing scale, a type scale and a colour ramp
C. None, because consistency on one artefact comes from looking at it
D. Tokens and six components, so that the next flyer is faster

EXAMINATION

Design, Before Any Software: final examination

This paper covers the whole course: how seeing works and how to rank a layout, space and grouping and the grid, type set from numbers rather than adjectives, colour measured against a contrast criterion, composition and the image, real screens and real paper, and turning one-off decisions into tokens and components. Each attempt draws twenty-five questions from a larger pool, with every module represented. The pass mark is seventy per cent, and passing opens the next course in the track. There is no timer: read each question as slowly as you like. If you do not pass, you may retake after thirty minutes and the paper will be drawn fresh, so it will not be the same questions. A teacher can open the next course for you at any time regardless of this paper, and that is recorded as an override rather than as a pass.

On the website a sitting is 25 questions drawn from this pool and the pass mark is 70%. There is no time limit, there and here. Passing opens the next course in the track.

1 1. How seeing actually works

Design is described as choosing three to five landing spots and the order they arrive in. Which fact about vision is that built on?

A. The eye moves in jerks and resolves sharp detail only in a small central patch

B. Readers scan a page from the top left to the bottom right at a steady rate

C. The retina resolves a whole page at once and the brain discards most of it afterwards

D. People keep looking at whatever is largest until they have understood it

2 1. How seeing actually works

The blur test and the squint test are run on the same design. What different questions do they answer?

A. The blur measures contrast ratios; the squint measures spacing relationships
B. The blur asks what can be found; the squint asks what comes first
C. The blur checks colour accuracy; the squint checks alignment
D. The blur tests printed output; the squint tests screen output

3 1. How seeing actually works

Every heading on a site is set in the brand orange. What has that cost?

A. Headings become harder to distinguish from links
B. The palette has spent the three semantic colours it was allowed
C. Orange now means heading rather than important
D. The brand colour will fail contrast on a white background

4 1. How seeing actually works

Why can motion not be budgeted alongside size, contrast and isolation?

A. It cannot be measured, so it has no place in a budget
B. It is disabled for a large share of readers by default
C. It applies only to interfaces and not to printed or static work
D. It overrides them rather than competing with them

5 1. How seeing actually works

A layout has a photograph of a person looking straight out at the camera, beside a headline you want read first. What does the research on gaze support?

A. Turning the model to look at the headline increases fixations on it
B. Moving the photograph below the headline removes the competition
C. Reducing the photograph's saturation makes the face rank below the text
D. Faces attract attention only when they are larger than the surrounding type

6 1. How seeing actually works

A title is placed at exactly half the height of a frame and looks slightly low. What is the usual correction?

A. Increase the line height so the block occupies more of the frame
B. Move it up to about 45 per cent of the height
C. Move it down to about 55 per cent of the height
D. Leave it, because the geometric centre is correct by definition

7 1. How seeing actually works

A designer introduces a circle to mark the primary action, then uses circles for bullets, icon backgrounds and a decorative motif on the same view. What has happened?

A. The corner radius tokens are no longer consistent with the cards
B. Shape contrast fails whenever more than two shapes are present
C. The shape channel has been spent without buying an emphasis
D. The circles now form a grouping by similarity that overrides proximity

8 1. How seeing actually works

The nine diagnostic checks are specified in a fixed order, starting with hierarchy and value. Why does the order matter?

A. The later checks need measurements produced by the earlier ones
B. Readers perceive a layout in that order, so the checks mirror the experience
C. Alignment problems become invisible once a design has been desaturated
D. Colour cannot repair a fault in structure, so that work is wasted

9 2. Space, grouping and the grid

A group of elements has a pocket of empty space in its middle that is wider than the space at the group's edge. What does the eye do with it?

A. It reads the pocket as a divider and splits the group in two
B. It reads the pocket as emphasis and fixates the nearest element
C. It ignores it, because internal space carries no grouping information
D. It reads the whole group as a single object, because the outer edge dominates

10 2. Space, grouping and the grid

A table is redrawn with every vertical rule and most horizontal rules removed, keeping only the alignment and one rule under the header. Why does it still read as a table?

A. Proximity is stronger in a grid than in any other arrangement
B. Closure completes the boundaries that were not drawn
C. Similarity groups the cells because they share a typeface
D. Common fate binds the rows because they scroll together

11 2. Space, grouping and the grid

Twelve columns became the conventional grid. What is the actual justification?

A. Twelve is the largest number of columns a 1200-pixel layout can hold
B. Twelve columns produce a gutter that matches an eight-pixel spacing scale
C. Twelve divides evenly by two, three, four and six
D. Twelve columns match the twelve-point typographic pica

12 2. Space, grouping and the grid

A 1200-pixel content area has twelve columns and 24-pixel gutters, so each column is 78 pixels. How wide is an element spanning four columns?

A. 400 pixels, which is one third of the content width
B. 312 pixels, four columns with no gutters added
C. 408 pixels, four columns and four gutters
D. 384 pixels

13 2. Space, grouping and the grid

A card list sets its gaps with gap on the container rather than with a bottom margin on each card. Besides reuse, what does that avoid?

A. Margin collapsing, where adjacent margins merge into one
B. Sub-pixel rounding, which makes the last card a pixel short
C. The need for a spacing scale, since gap enforces its own rhythm
D. Layout shift while images inside the cards are still loading

14 2. Space, grouping and the grid

Two 48-pixel targets sit directly beside each other with no gap. Why is that still a fault?

A. A finger's contact patch is larger than 48 pixels, so neither can be hit reliably
B. The error is hitting the wrong target rather than missing both
C. Targets must be square, and adjacency makes them effectively rectangular
D. The combined target exceeds the maximum size the platform guidance allows

15 2. Space, grouping and the grid

A composition is about five per cent off balance. What is the recommended response?

A. Add a small element on the light side to even the weights
B. Leave it, because slight asymmetry is what makes a layout feel alive
C. Resolve it properly, or push the imbalance much further
D. Centre everything, which removes the question

16 2. Space, grouping and the grid

Text vertically centred in a button with equal top and bottom padding looks as though it has sunk. Why?

A. Buttons render their text one pixel low on most platforms
B. The cap height is always larger than the x-height, which pulls the mass up
C. Padding is measured from the border box, which includes the border width
D. The line box reserves descender space that most words do not use

17 3. Type, mechanically

A display face is used for body text and falls apart at 15 pixels. What distinguishes a display face from a text face?

A. It is drawn for large sizes: fine detail, tight spacing, dramatic contrast
B. It contains fewer weights, so it cannot carry a hierarchy
C. It has no italic, so emphasis has to be carried by weight
D. It is licensed for headlines only, and the licence excludes running text at any size

18 3. Type, mechanically

You have decided to use two typefaces. Which pairing is most likely to read as a mistake?

A. A slab serif for headings with a humanist sans for body text
B. Two humanist sans faces with similar proportions
C. A geometric sans with a transitional serif
D. A serif for long reading with a monospace for code

19 3. Type, mechanically

A countdown timer's digits shift left and right by a pixel or two on every tick. What is the repair?

A. Give the element a fixed width in pixels
B. Switch the figures from lining to oldstyle
C. Set the element in tabular figures
D. Set the element in a monospace typeface

20 3. Type, mechanically

A page shows rupee amounts as 1,200,000 to readers in India. What is the correct fix?

A. Insert the commas manually in the Indian pattern
B. Use a typeface with Devanagari coverage for the figures
C. Set the figures in tabular numerals so the grouping aligns
D. Format the number with a locale-aware formatter

21 3. Type, mechanically

A paragraph containing NHS, HTML and PDF has three visibly dark patches in it. What is the traditional repair?

A. Set the acronyms in real small capitals
B. Set the acronyms in a lighter weight of the same face
C. Add positive letter spacing to the whole paragraph
D. Replace the capitals with lowercase and rely on context

22 3. Type, mechanically

A headline breaks with one word alone on its second line. Which declaration is aimed at that problem for a short block?

A. overflow-wrap: break-word
B. text-wrap: balance
C. text-wrap: pretty
D. hyphens: auto

23 3. Type, mechanically

Devanagari set at a line height of 1.4 collides with the line below. What is the structural reason?

A. The shirorekha adds a fixed number of pixels to every line
B. Devanagari fonts are drawn on a larger em square than Latin fonts of the same size
C. Vowel signs stack above the headline and conjuncts stack below the baseline
D. Devanagari glyphs have a larger x-height ratio than Latin ones

24 3. Type, mechanically

A studio has a desktop licence for a commercial typeface and wants to use it on a client's website. What is the usual position?

A. Any purchased licence covers all uses by the purchaser
B. Web use is covered provided the font is subsetted
C. Web embedding is permitted if the client is credited in the source
D. A desktop licence generally does not cover web embedding

25 4. Colour, measured

An interface mixes greys tinted towards blue with greys tinted towards orange. What does a reader notice?

A. The page looks slightly dirty and no element can be blamed
B. The warm greys appear to advance and the cool greys recede
C. Contrast ratios drift, because tinting changes relative luminance
D. Nothing, because a tint below 0.02 chroma is imperceptible

26 4. Colour, measured

A neutral ramp is built with mathematically even lightness steps and the dark end still looks compressed. Why?

A. Screens apply a gamma curve that compresses the shadows on output
B. Differences near black are harder to see, and room flare widens the gap
C. Dark values are quantised more coarsely in eight-bit colour
D. Perceptually uniform spaces are only accurate in the upper half of the range

27 4. Colour, measured

Secondary text is declared as black at 60 per cent opacity. Why can a contrast checker not evaluate it from that declaration?

A. Semi-transparent text is exempt from the contrast criteria
B. Checkers require hex values and cannot parse an alpha channel
C. Its ratio depends entirely on what is composited behind it
D. Opacity is applied after the contrast calculation in the rendering pipeline

28 4. Colour, measured

Grey #767676 on white measures 4.54 to 1 and #777777 measures 4.48. What does that pair illustrate?

A. Contrast ratios are unreliable at the light end of the scale
B. Rounding in the checker makes results below 5 to 1 untrustworthy
C. Any grey lighter than mid-grey fails for body text on a white background
D. The threshold is a cliff, and a refined grey often lands just under it

29 4. Colour, measured

A financial product uses green for a rise and red for a fall in every market it ships to. Where does that go wrong?

A. In China, Japan and South Korea, where red conventionally marks a rise
B. In Europe, where the convention is inverted relative to India
C. Everywhere, because red and green cannot be told apart by a large share of readers
D. Nowhere, provided both colours meet the non-text contrast requirement

30 4. Colour, measured

A design is run through a deuteranopia simulator and the distinction between two states survives. How strong is that evidence?

A. Strong, provided the simulator is run at the device's real brightness
B. Weaker than a failure, because simulators model the common case loosely
C. Conclusive, because deuteranopia is the most severe of the common deficiencies
D. Irrelevant, because simulators model printing rather than vision

31 4. Colour, measured

A map shades a single increasing quantity with a rainbow scale. What is the specific defect?

A. Rainbow scales cannot be reproduced in CMYK
B. It requires a legend to be read, which a sequential scale does not need
C. Its perceived lightness is not monotonic, so false boundaries appear
D. It uses too many hues to remain colour-blind safe

32 4. Colour, measured

A print job sets nine-point body text in a rich black built from all four inks. What goes wrong on press?

A. The text prints noticeably lighter than the surrounding solids
B. Four-ink coverage exceeds the maximum the paper can absorb at that size
C. The text cannot be trapped, so it drops out on uncoated stock
D. Slight misregistration produces coloured fringes around every letter

33 5. Composition and the image

What does the rule of thirds genuinely buy a photographer?

A. It stops the horizon landing dead centre, which splits the frame
B. It places the subject on one of four points that attract fixations
C. It produces the proportions found in classical painting
D. It guarantees the composition survives a crop to any other ratio

34 5. Composition and the image

A subject is facing towards the right edge of the frame. Where should they be placed?

A. Right of centre, with the empty space behind them for context
B. Left of centre, so the gaze has somewhere to travel
C. Right of centre, on the thirds line they are facing
D. Dead centre, so neither edge is favoured

35 5. Composition and the image

A 4000 by 3000 pixel photograph is cropped to a quarter of its area. What is the largest sharp print at 300 pixels per inch?

A. About 3.3 by 2.5 inches
B. Unchanged, because resolution is set at export rather than by the crop
C. About 6.7 by 5 inches
D. About 13.3 by 10 inches

36 5. Composition and the image

Which change pushes a decorative illustration into the background of a flat layout?

A. Add a soft drop shadow beneath it
B. Reduce its size and move it higher in the frame
C. Apply a Gaussian blur, which is the only cue available without perspective
D. Reduce its contrast and saturation towards the background's lightness

37 5. Composition and the image

An interface has cards casting a shadow downwards, a modal casting one in all directions and a button with a shadow above it. What is the result?

A. The depth reading collapses, because there is no single light
B. The modal reads as furthest forward, because its shadow is largest
C. Nothing, because shadows are decorative rather than informative
D. The design gains three levels of depth, which is one more than needed

38 5. Composition and the image

A designer fills every element of a layout with flat black on a white background and looks only at the white. What is being evaluated?

A. The value structure, which is the same as the squint test
B. The shape of the negative space, and whether it is coherent
C. The contrast ratio of each element against the background behind it
D. Whether the layout survives a greyscale photocopy

39 5. Composition and the image

Five viewers are shown a two-subject photograph for three seconds and give three different answers about what they saw first. What does that split diagnose?

A. The viewers disagree with the designer about the intended reading order
B. The background is more prominent than either subject
C. There is no primary, because two subjects share the same rank
D. The test was too short to produce a reliable result

40 5. Composition and the image

An image is uploaded to a social platform and comes back with the fine text looking mangled. What is the usual cause?

A. The platform converted the image to a smaller colour space
B. The image was uploaded at larger than the platform's stated dimensions
C. The platform applied its own sharpening before compressing
D. Lossy re-encoding discards thin pale strokes first

41 6. Real screens, real thumbs, real paper

Why is 360 CSS pixels the recommended width to design at rather than 393?

A. A layout that fits at 393 can still break at 360, and not the reverse
B. 360 is the width at which iOS reports its viewport
C. 360 divides evenly by a four-pixel spacing scale, and 393 does not at all
D. Device pixel ratio is always 2 at 360 and 3 at 393

42 6. Real screens, real thumbs, real paper

A logo is displayed 120 CSS pixels wide on a phone with a device pixel ratio of 3. What does the asset need?

A. A 360-pixel raster file, because vectors cannot be used for logos
B. A 360-pixel raster file, or an SVG at any size
C. A 120-pixel raster file, because CSS pixels are what is displayed
D. A 240-pixel raster file, since 2x is sufficient on all devices

43 6. Real screens, real thumbs, real paper

How do you test the reflow requirement, which asks that content be presentable at a width equivalent to 320 CSS pixels?

A. Set the browser's minimum font size to 200 per cent and reload
B. Resize the window to 320 pixels and measure the tap targets
C. Zoom a 1280-pixel browser to 400 per cent and check for sideways scrolling
D. Open the page on the narrowest phone available and check nothing is clipped

44 6. Real screens, real thumbs, real paper

A responsive layout uses CSS order to move a panel visually, and keyboard focus now appears to jump around the page. What is the rule?

A. Add tabindex values to force the focus into the visual order
B. Avoid CSS order entirely and use absolute positioning instead
C. Set the panel to display: contents so it is removed from the focus order
D. If you reorder visually, reorder the source to match

45 6. Real screens, real thumbs, real paper

A form field's only label is its placeholder text. Which of these is a genuine problem with that?

A. The label disappears as soon as the user types, so nothing can be checked
B. Placeholder text cannot be translated, so the whole field is untranslatable
C. Placeholder text is never announced by any assistive technology
D. A placeholder prevents the field from receiving keyboard focus

46 6. Real screens, real thumbs, real paper

An icon sits inside a button that already carries visible text. What should its alt or aria treatment be?

A. Give it the file name, which is better than an empty value
B. Mark it hidden, because announcing it twice is noise
C. Describe the icon's appearance, so the reader knows what is shown
D. Repeat the button's label, so the association is explicit

47 6. Real screens, real thumbs, real paper

Why should a loading indicator be delayed by about 300 milliseconds rather than shown immediately?

A. Screen readers cannot announce an indicator that appears instantly
B. Three hundred milliseconds is the threshold at which attention is lost
C. A fast response then never flashes one, which would read as a stutter
D. It gives the request time to fail before anything is drawn

48 6. Real screens, real thumbs, real paper

A 120-page book is perfect-bound. How should the inner margins be set?

A. Equal to the outer ones, so the spread looks symmetrical
B. Smaller than the outer ones, to maximise the text area
C. Equal to the bleed, since the spine is trimmed like any other edge
D. Larger than the outer ones, because the spine takes part of them

49 7. Making the decision once

A token is named colour-red-error. What will go wrong?

A. When the colour becomes orange, the name is a lie that stays
B. The name is too long for autocomplete to group it usefully
C. Red cannot be used for error states in every market
D. Two words describing colour make the token ambiguous to parse

50 7. Making the decision once

Why number a ramp 50, 100, 200 and so on rather than naming the steps small, medium and large?

A. A numeric name can be parsed into a value by the build tool
B. There is always room to insert a step between two numbers
C. Numbers are shorter, which keeps stylesheets smaller
D. T-shirt sizes are reserved for component sizing by convention

51 7. Making the decision once

What problem does an on- prefix solve, as in colour-action and colour-on-action?

A. It distinguishes semantic tokens from primitives in autocomplete
B. It records which tokens have been checked for contrast
C. It keeps a background and its foreground paired when themed
D. It marks tokens that may only be used on interactive elements

52 7. Making the decision once

Why is a token called space-16, meaning sixteen pixels, a trap?

A. Numbers in a name prevent the token from being generated for other platforms

B. It conflicts with the primitive layer, which already numbers by value

C. Sixteen-pixel spacing is too large to be the third step on a scale

D. A value change makes the name untrue, and readers trust the name

53 7. Making the decision once

Which of these is the most reliable sign that a component should be split in two?

A. Its name contains the word and

B. It has more than three variants

C. It is used on fewer than five screens

D. Its documentation is longer than one page

54 7. Making the decision once

Three icons taken from three sources sit in a row and look wrong. Which differences are doing it?

A. Licensing, which forces a different attribution line for each one

B. Grid size, stroke weight, corner treatment and level of detail

C. Colour profiles, which differ between vector formats

D. File format, since SVG and icon fonts render differently

55 7. Making the decision once

Why is a component gallery generated from the real components better than a written specification document?

A. It is the only format that can show every state of a component

B. It satisfies the accessibility requirement for component documentation

C. It cannot go stale, because it is produced by the thing it documents

D. It removes the need to name the parts of each component

56 7. Making the decision once

Which is a legitimate reason to break the design system?

A. A screen that the designer finds dull to look at

B. A component that would take longer to reuse than to redraw

C. A stakeholder who prefers a different treatment on their page

D. A moment that must not feel routine, such as a deletion

Answers

Each entry gives the correct option, then what each wrong one reveals about how somebody was thinking. The second part is worth more than the first: a wrong answer here is a named misreading, not a mistake.

Lesson checks

1. Your poster is not ugly, it is unranked — A

B — Treats crowding as an absolute size problem when the fault is that no element outranks any other.

C — Reaches for a new typeface to create a rank that a size difference has not yet been asked to create.

D — Confuses grouping with ranking — even spacing tells the eye nothing about which item comes first.

2. What your peripheral vision is actually for — D

A — Takes a true observation about peripheral vision and turns it into an unconditional style rule, ignoring that the neighbours are the dominant problem.

B — Treats it as an absolute size floor when a 16px icon alone in white space is perfectly findable.

C — Blames reading order for a failure that persists wherever in the row the icon sits, because the cause is interference from adjacent shapes rather than position.

3. The squint test, done properly — A

B — Blames the display for a conversion error that would reproduce identically on a perfect monitor.

C — Confuses hue opposition with lightness difference; complementary colours can sit at nearly the same luminance.

D — Identifies a real step of the procedure but not the one that caused this failure, which occurred before any blur.

4. How big a difference has to be — C

A — Blames the count when a multiplicative eight-step scale would work; the fault is the spacing between steps, not the number of them.

B — Invents a typographic constraint that does not govern whether two gaps are distinguishable from each other.

D — Raises a genuine responsive concern that has no bearing on whether two fixed gaps look different from one another.

5. Emphasis is zero-sum — B

A — Applies the budget to the whole document when it governs a single glance; the reader never sees all six at once.

C — Invents an exemption for a category of element; a repeated call to action within one view would compete exactly like anything else.

D — Reaches for differentiation between peers when the question is about ranking within a view, and adds a channel that would weaken the hierarchy further.

6. What makes a shape read as an object — D

A — Strengthens the weakest device; a thin line is high-frequency detail that peripheral vision and poor viewing conditions discard regardless of its contrast.

B — Solves separation with the one channel that collapses first in daylight and fails for colour-blind readers.

C — Makes a depth claim louder rather than a grouping claim clearer, and produces a page of stickers.

7. What the eye-tracking research actually says — C

A — Correctly notes that the research was desktop-era but treats the shape as a valid rule in its original setting, which it was not.

B — Raises a real but secondary issue; the change would underperform for left-to-right readers too.

D — Reaches for more emphasis on the same elements when the problem is the absence of structured landing points across the page.

8. Weight is not area, and centre is not the middle — B

A — Blames the container for an effect that reproduces on a perfect circle, because the asymmetry is in the glyph.

C — Applies the optical-centre rule, which is a vertical phenomenon, to a horizontal complaint.

D — Invents a size-dependent displacement effect in place of the straightforward mass distribution of a triangle.

9. Shape is a channel you are probably not using — B

A — Reaches for a rendering explanation for an effect that is purely geometric and would appear identically on any screen.

C — Swaps one universal value for another, which reproduces the same proportional mismatch at a different scale.

D — Attributes the difference to the fill treatment rather than to the ratio between the radius and the element's height.

10. Content that attracts attention whatever you do to it — C

A — Correctly identifies that motion wins, and spends the most intrusive channel available on a problem that has quieter solutions.

B — Keeps escalating a formal channel against a content effect that does not compete on those terms.

D — Reduces one attribute of the image while leaving intact the face detection that is doing the attracting.

11. Nine checks that find what is wrong with a layout — D

A — Objects to the techniques as fashion, rather than to where they sit in the sequence of diagnosis.

B — Orders the work by how much effort each step takes rather than by what depends on what.

C — Treats the ordering as an approval process question instead of a structural dependency between decisions.

12. The gap is doing the talking — C

A — Treats a grouping failure as a legibility failure, when the labels are perfectly readable.

B — Reaches for a drawn boundary when the relationship is already being communicated — wrongly — by distance.

D — Assumes the problem is the quantity of content rather than the spacing that encodes what belongs to what.

13. When two grouping cues disagree — C

A — Escalates the same weak cue; a stronger colour still loses to the grid's proximity structure that says rows and columns are the groups.

B — Invents a numeric limit; the difficulty is the competing proximity structure, not the count.

D — Reaches for a claim about advancing and receding hues in place of the competition between grouping cues.

14. Every gap comes from a list — B

A — Raises a reasonable convention that would produce a nearly identical list and leave the underlying problem untouched.

C — Changes the base while keeping the additive shape that caused the problem, so the top of the scale stays undifferentiated.

D — Names a real accessibility benefit of relative units and misattributes an appearance problem to the unit rather than to the shape of the ramp.

15. Space is content you have not been billed for — B

A — Applies the threshold rule to absolute size when the failure is that all the ratios between gaps were preserved.

C — Turns a genuine trade-off about scroll cost into a rule that would strip out the macro spacing doing the most work.

D — Jumps to an inventory problem when the diagnosis is that the distribution of space, not the amount of content, was left untouched.

16. You can feel a misalignment you cannot see — B

A — Hunts for a measurement error when the measurement is correct and it is the perception that differs from it.

C — Reads the problem as a size relationship rather than a centre-of-mass one, so shrinking it changes nothing.

D — Blames rendering for an effect that persists at any resolution, in vector output and in print.

17. Where the alignment tool is wrong — C

A — Attributes a vertical effect to the button's width, which does not change where the letters sit in the line box.

B — Correctly notes that letterforms differ but locates the effect horizontally, when the complaint is about vertical position.

D — Invokes a real effect that operates at a fraction of a pixel, far too small to account for a visible imbalance.

18. A grid is arithmetic, not taste — A

B — Introduces the margin as the error source when the miscalculation happens entirely inside the content area.

C — Claims twelve does not divide by three, when it does; the error is in how gutters are counted.

D — Applies a plausible-sounding alternative arithmetic that double-counts the gutters and produces a span far too narrow.

19. Vertical rhythm, and why the strict version fails on screen — B

A — Names a real source of baseline unpredictability that is not what causes lines to collide at a fixed 24px.

C — Invokes the half-leading model correctly but treats a shortage of surplus as the cause rather than the symptom of an inherited fixed value.

D — Invents a fallback behaviour; the browser applies the inherited 24px exactly as declared, which is precisely the problem.

20. Padding belongs to the thing, margin belongs to the relationship — A

B — Describes the nesting problem, which is real but concerns padding rather than the margin that travelled with the component.

C — Applies margin collapsing to grid items, where it does not occur, and predicts the opposite of the observed symptom.

D — Transplants the nested-radius rule into a situation about spacing between siblings rather than a shape inside a shape.

21. How big a thing has to be to hit — C

A — Focuses on a four-pixel shortfall against the guideline when the errors are caused by adjacency rather than by target size.

B — Inverts the relationship: distance and width sit symmetrically inside the logarithm, and width is the cheaper of the two to change.

D — Describes a plausible implementation slip, but the stated change was to the rendered size of the controls themselves.

22. Balance is a comparison of weights — B

A — Assumes weight must be matched element for element, ignoring that distance from the axis multiplies it.

C — Invents a directional convention in place of an account of where the masses sit relative to the axis.

D — Treats balance as an equality of areas rather than of weight multiplied by lever arm, and would work only by accident.

23. Your font is fine, your line is too long — D

A — Repeats a screen-versus-print claim that no longer holds at modern screen resolutions, and never looks at the column width.

B — Treats leading as an independent setting rather than something that has to follow from the length of the line.

C — Correctly identifies a real x-height effect, then applies it to a problem that is being caused by the width of the column.

24. The four measurements that decide how a typeface behaves — D

A — Repeats a claim that has not held up in reading research; the difference here is measurable and specific to the two faces' proportions.

B — Identifies a genuine secondary effect while missing that the letters themselves became substantially smaller.

C — Names a real property of the face that matters below about 12px and contributes little at 16.

25. Four sizes, chosen once — C

A — Names a real constraint of viewport units that affects layout proportion rather than the reader's ability to control text size.

B — Identifies a genuine gap that clamping would also fix, but the more serious failure applies at every width.

D — Applies the threshold rule to a value that in fact varies continuously and will usually be far from the body size.

26. Choosing a typeface, and what the licence actually says — D

A — Overstates the restriction; desktop licences ordinarily cover producing commercial artwork on the licensed machines.

B — Takes one jurisdiction's position on letterform designs and applies it to the font software, which is licensed separately and is protected.

C — Imports a Reserved Font Name condition from the Open Font Licence into a commercial licence where it does not apply.

27. Two typefaces, and why you probably want one — A

B — Names a real matching step, but a size mismatch would read as a size problem rather than as a rendering fault.

C — Raises a genuine selection criterion that would only affect italic text, not the headings generally.

D — Describes a real pairing adjustment that would produce an inconsistent hierarchy rather than the impression that a font failed to load.

28. Rivers, rags and where the line breaks — C

A — Inverts the relationship: hyphenation matters most in justified setting, because it gives the engine an alternative to stretching spaces.

B — Reverses the mechanism; wide measures are where justification works best and narrow ones where it fails.

D — Invents a mandatory companion property; the limit values are optional refinements, not a precondition.

29. Letter spacing, and the sizes type was drawn for — B

A — Reaches for a proportion argument when the damage is structural rather than a matter of degree.

C — Blames absent kerning data for an effect caused by interrupting a visually continuous element of the script.

D — Invents a differing em definition; the em is a fixed reference and the same 0.05em applies identically.

30. Capitals, italics and the cost of emphasis — B

A — Overstates the separation; the point is that the underlying text is now sentence case, not that CSS is invisible to assistive technology.

C — Invents a dictionary distinction based on styling; the difference lies entirely in the characters present in the content.

D — Correctly notes that short capital strings are often read as initialisms but prescribes an override that the stated fix did not use.

31. Figures, dashes and the details nobody teaches — C

A — Treats a glyph-width issue as a container issue; the jitter persists inside a fixed container because the digits themselves differ in width.

B — Identifies the wrong numeric variant — oldstyle affects vertical profile, not the horizontal advance that causes jitter.

D — Attributes the shift to kerning between digits, which is rare in practice, rather than to the differing advance widths of proportional figures.

32. The grey of a paragraph — D

A — Assumes an applied bold weight when capitals produce the density on their own, with no emphasis applied.

B — Reverses the effect of tracking, which would lighten a run of capitals rather than darken it.

C — Attributes the density to stem weight rather than to the greater height and closed area of capital letterforms.

33. What breaks when the text is not in English — B

A — Treats clipping as a size problem, which changes the proportions without giving the line box more room, and leaves the shirorekha still broken.

C — Looks for a font that tolerates hostile settings instead of correcting settings that interrupt a connecting stroke.

D — Fixes the visible symptom of clipping by letting glyphs escape their box, which then collides with the line above instead.

34. Make it pop is a symptom report — C

A — Accepts a symptom report as a specification, which delivers the request but not the outcome the request was about.

B — Defends the artefact instead of finding out what the comment was actually caused by.

D — Substitutes volume for diagnosis, and hands the client a decision they have no basis on which to make.

35. A palette will not save an unreadable page — A

B — Recasts a colour-coding problem as a size problem, and misreads what the contrast figure actually measures.

C — Swaps one hue pair for another instead of adding a second channel such as shape, position or a label.

D — Applies a stricter version of the wrong test, treating a legibility threshold as if it governed distinguishability.

36. The lightness slider is not lightness — C

A — Raises a real limit on distinguishable categories, which does not explain why two specific series sit at opposite extremes of lightness.

B — Correctly notes that high chroma widens the spread while implying that reducing saturation alone would equalise luminance across hues.

D — Blames the conversion step for a difference that is already present in the colours themselves before any conversion.

37. What a contrast checker is actually computing — A

B — Correctly separates declared from rendered colour but blames antialiasing rather than the composite against a different background.

C — Treats an AAA shortfall as the cause when the same setting was reported as acceptable on the white pages.

D — Invents a variable flare constant; the 0.05 term is fixed and does not respond to the background colour.

38. Where the contrast standard is wrong, and what is unsettled — C

A — Applies the dark-pair overrating correctly as a concept but to a high-contrast pair where the constant has little influence.

B — Invents a polarity-specific threshold; WCAG 2 applies the same numbers in both directions.

D — Attributes a perceptual effect to font hinting, which does not vary with the colours applied to the text.

39. What colour blindness actually removes — C

A — Inverts the reliability of simulation evidence: a disappearing distinction is the conclusive result, not the doubtful one.

B — Names a real limit that sits nearer eight than six, and that tested palettes such as Okabe-Ito already address.

D — Assumes a toggle would be implemented incompletely, which is an implementation detail rather than the objection to the approach.

40. Colour is never the only signal — C

A — Names a real gap in the hover approach that would still leave the desktop case defensible, which it is not.

B — Cites a genuine supplementary requirement for colour-only links, which is a mitigation rather than the failure itself.

D — Overstates the requirement; the guidelines permit alternatives, which is exactly why the timing of the hover cue matters.

41. Greys are a design decision — B

A — Names a real failure mode of HSL ramps that would also show up in the dim room, where the levels were distinct.

C — Blames gamut for a difference in viewing conditions; the same phone in a dim room would show the levels apart.

D — Applies the non-text contrast criterion to decorative surface elevation, where it does not set the requirement.

42. Red means danger, except where it does not — B

A — Applies the internal-consistency principle to a domain convention users already hold from every other financial source they read.

C — Overcorrects the never-colour-alone rule into never-colour, discarding a fast signal that works when it is backed by another.

D — Offers configuration in place of a correct default, which leaves the misreading in place for everyone who never opens settings.

43. Dark mode, built rather than inverted — C

A — Names a real consequence of the approach, and the surface treatment is repairable by choosing different endpoints.

B — A lightness inversion changes lightness rather than hue, so the brand colour shifts in brightness but does not become its opposite.

D — Identifies a genuine irritation of blanket inversion that affects isolated components rather than the depth model of the whole interface.

44. Why your blue printed purple — A

B — Treats rich black as a gamut problem when it is a combination of four in-gamut inks; the issue is how they align on the sheet.

C — Raises a real ink-limit concern that would affect large solids far more than fine text, and at a coverage below typical limits.

D — Blames the typeface for an effect that would appear in any face set in four inks at that size.

45. How many colours a design needs — C

A — Applies the categorical distinguishability limit to continuous data, where the quantity is read from position on a ramp rather than by identifying discrete colours.

B — Correctly flags a deficiency problem but prescribes a qualitative palette for data that is continuous rather than categorical.

D — Describes a property of diverging scales, which is not what continuous low-to-high data requires.

46. The rule of thirds is a nudge, not a law — C

A — Turns a heuristic's limits into a categorical rule about subject matter instead of asking what this picture needs.

B — Attributes a compositional discomfort to a technical property a viewer will not notice at normal viewing size.

D — Replaces one blanket rule with another blanket rule, rather than deciding whether this picture wants balance or tension.

47. Why a near-touch is worse than an overlap — C

A — Offers one of the two valid commitments without explaining why the small gap fails, and forces a treatment the image may not want.

B — Invents a proportional rule in place of the perceptual point that the gap must be unambiguously readable as a gap.

D — Restricts tangency to curves when parallel straight edges produce the effect just as strongly.

48. A crop is a decision about what leaves — B

A — Compares the numbers in the wrong orientation and concludes a shortfall where there is headroom.

C — Halves the requirement on a rule of thumb that applies to large-format work viewed at a distance, not to a hand-held flyer.

D — Reasons from total printable area rather than from the pixel dimensions the required orientation actually demands.

49. Building a path through the image — B

A — Correctly identifies that faces outrank headlines while missing that the gaze is also pushing attention off the layout.

C — Treats reading direction as a hard constraint when a strong implied line readily overrides it in either direction.

D — Applies the tangent principle to a placement problem that concerns where an implied line points rather than where a contour touches.

50. Six ways to make a flat surface have depth — C

A — Names a real structural guideline that would not by itself resolve contradictory light directions among however many levels remain.

B — Correctly ranks the cues and concludes that shadows cannot work, when a consistent set reads perfectly well.

D — Describes a real way shadows go wrong that concerns their individual styling rather than their disagreement with one another.

51. The space between things has a shape — C

A — Treats the amount of space as the fault when the uniformity of its distribution is what removes the emphasis.

B — Adds hierarchy in a channel that is already working, leaving the spatial distribution untouched.

D — Reaches for a small asymmetry that lands in the near-miss band rather than a deliberate redistribution of space.

52. One image, four crops — C

A — Names a real compression effect that degrades text rather than concealing it behind something.

B — Applies the correct crop arithmetic to a platform that displays vertical assets full-frame rather than cropping them.

D — Applies the multi-crop intersection rule to a case where only the vertical version is being published.

53. Two focal points is none — B

A — Applies a sampling standard appropriate to measuring a rate to a signal that is informative precisely because it is a disagreement.

C — Raises a real limitation of self-report while dismissing a spread of answers that a design with a clear primary would not produce.

D — Assumes instrumentation would overturn the finding, when aggregate heatmaps are themselves averages that can hide exactly this kind of variation.

54. Text over a picture you do not control — B

A — Reaches for the weakest device, which produces a muddy halo and fails entirely against detailed light backgrounds.

C — Solves for the worst case by darkening every image so heavily that the photograph stops being visible at all.

D — Applies a real refinement to gradient construction that improves appearance without guaranteeing a ratio against an unknown image.

55. Repetition, and the one that breaks it — C

A — Prescribes eliminating variation altogether, which removes the rhythm rather than restoring the prediction it depends on.

B — Names a real spatial consequence that would not by itself make the page read as disorganised if the variant appeared once.

D — Describes the periodicity accurately but locates the fault in the irregularity of the spacing rather than in the repetition of the deviation.

56. Design it at 390 pixels or find out the hard way — B

A — Names a genuine mobile behaviour that changes the available height, not the zoom level of the page.

C — Picks a real cause of bad mobile rendering that would be visible from first paint, not only on focusing a field.

D — Confuses a target-size guideline with a browser behaviour that guideline does not trigger.

57. What a pixel is, and how wide phones actually are — C

A — Confuses the hardware scaling factor, which affects sharpness rather than layout extent, with the viewport measurement.

B — Names a real inset that accounts for a few tens of pixels rather than the height of a browser address bar.

D — Invokes a text-inflation behaviour that affects text sizing rather than the interpretation of the viewport unit.

58. Breakpoints come from the content, not from devices — B

A — Enumerates one more case in a system that will need a new case for every context the card is placed in.

C — Moves the threshold without changing the quantity being measured, so the same mismatch reappears at a different width.

D — Reaches for the right general preference, but auto-fit governs how many items sit in a track rather than how one component arranges its own parts.

59. Where a thumb can actually go — B

A — Correctly dates the source while drawing the wrong conclusion; larger phones make the reach constraint stronger rather than weaker.

C — Raises a spacing constraint that may or may not bind, and misses the consequence-based reason the placement is wrong.

D — Names a real inset that occupies a thin strip rather than the bottom third, and that correct padding resolves.

60. Why 10pt is fine in print and terrible on screen — C

A — Repeats a blanket claim about serifs rather than identifying the resolution that makes this particular face fail.

B — Correctly identifies that the size is too small while ignoring that the face's hairlines fail well above that floor.

D — Raises a genuine unit conversion that would enlarge the text slightly without addressing why the strokes themselves disappear.

61. The five screens nobody designs — B

A — Correctly identifies that peripheral motion captures attention while missing that the brevity of the appearance is what reads as a stutter.

C — Substitutes one indicator for another when the fault is showing any indicator at all for a response this fast.

D — Describes a plausible implementation defect rather than the timing principle that makes the indicator inappropriate here.

62. Motion, and the people it makes ill — C

A — Assumes the trigger scales down safely with distance, when the reliable accommodation is to remove the traversal of the visual field.

B — Retains the movement and merely accelerates it, which does not address why the motion was a problem.

D — Draws a sensible-sounding line that leaves the user receiving motion they explicitly asked not to receive.

63. The accessibility requirements that are design decisions — B

A — Inverts which order assistive technology follows and invents a duplication that does not occur.

C — Attributes a removal from the accessibility tree to a property that only affects layout position.

D — Invokes the reflow criterion, which concerns whether content fits a narrow width rather than the sequence elements appear in.

64. The designer decides how heavy the page is — C

A — Fixes a real and separate problem that changes the reading experience without reducing the bytes transferred at all.

B — Delivers a genuine saving that is smaller than serving an appropriately sized file, since the image remains six times wider than needed.

D — Names a large and real saving that still leaves the browser downloading an image six times wider than the screen can show.

65. Bleed, trim and the things paper does — C

A — Describes a real effect belonging to a different binding method, and one that narrows the outer rather than the inner margin.

B — Applies bleed to an edge that is bound rather than trimmed, and would account for a millimetre rather than the loss observed.

D — Cites the traditional page canon, whose larger bottom and outer margins address optical balance rather than binding loss.

66. A token is a decision with a name — D

A — Doubles the primitive layer while leaving components responsible for knowing which theme is active, which is the work being avoided.

B — Solves distribution across platforms rather than the absence of a role layer that themes can be swapped at.

C — Assumes a dark theme is a lightness inversion, which the colour block established it is not.

67. Names that survive the next change — B

A — Argues against role naming itself, which is the convention that makes tokens survive change, and misses the findability problem.

C — Identifies a real structural gap that would cause designers to reference primitives rather than to hard-code arbitrary values.

D — Names a genuine consistency hazard that the given examples do not actually exhibit.

68. Variant, or a new thing — C

A — Adds a property whose validity depends on another property's value, which is one of the signs a component has outgrown its shape.

B — Splits into components with identical anatomy and different content slots, which produces the duplication the split was meant to avoid.

D — Preserves the interface by degrading the content into a raster image, losing interactivity and accessibility to avoid a structural change.

69. Every state you did not draw still ships — B

A — Extends the exemption from the control to an associated tooltip, and treats contrast as the obstacle rather than reachability.

C — Moves the message somewhere that forces the user to hunt for which field each item refers to.

D — Hardens a useful default into an absolute, when the real objection is that the chosen explanation is unreachable.

70. Icons are a set, not a collection — A

B — Identifies the right remedy for the wrong reason, treating reachability of the tooltip as the cause rather than the absence of a visible label.

C — Applies the set-consistency point, which governs whether icons look coherent rather than whether their meaning is understood.

D — Raises a genuine and separate defect that would not explain sighted users guessing wrong about what an icon means.

71. The afternoon version — B

A — Treats the existing screens as the specification, which imports the accidental values the system was built to remove.

C — Assumes more steps is the answer without asking whether the original screen's greys were distinguishable or intended.

D — Reads a normal and expected finding as a failure, when the point of the rebuild is to surface exactly this kind of gap.

72. Specifications that survive contact — A

B — Correctly notes that pictures under-specify, but a rebrand does not depend on configuration rules — it depends on where the value lives.

C — Introduces a missing-states problem that would produce inconsistency between states rather than a component that ignores a rebrand.

D — Applies the staleness point to a document that was accurate when it was used; the failure occurred in what it encoded, not when.

73. Consistency is not uniformity — D

A — Names the right pair of concepts and assigns the behaviour to a desire for variety rather than to an unmet need.

B — Contradicts the premise that the system is well documented, and would produce confusion rather than deliberate detaching.

C — Describes a genuine decay mechanism that presumes exceptions were being documented in the first place, which detaching bypasses entirely.

74. When not to build one — C

A — Trims the scope of an activity whose timing is the problem, not its size.

B — Applies the trigger correctly while missing that prototype screens are meant to be discarded whatever the team size.

D — Cites a real ongoing cost that would apply once the product were validated, rather than the reason this particular work is premature.

Module test — 1. How seeing actually works

1. A

A size step has to be at least about 1.2 times the size below it before it reads as deliberate. Below that threshold the difference might be an accident, and a difference that might be an accident carries no information — it reads as a production error rather than as a rank. The cost of the small step is identical to the cost of a large one, and it buys nothing.

2. B

Emphasis is a relationship between an element and everything around it, so there is a fixed amount of it on one surface. Four things are currently competing at the top rank. Demoting the two that are cheapest to demote makes the button win without changing the button at all, which is faster than promoting it and does not spend a further channel.

3. C

The eye is far more sensitive to green than to blue, so perceived brightness weights the channels roughly 21 per cent red, 72 per cent green and 7 per cent blue. An unweighted average flattens that, and a blazing yellow and a deep blue come out as nearly the same grey — which is precisely the fault the test exists to expose.

4. D

In peripheral vision, nearby marks interfere with each other — crowding — and the interference comes from roughly anything within half the distance to the point you are fixating. A word that is perfectly legible alone out there is unreadable inside a paragraph. Isolation does not merely enlarge the target; it deletes the flankers that were destroying it.

5. A

The four figure-ground devices are not equal. Space is the strongest and cheapest, then a fill, then a border, then a shadow — and a shadow is a claim about depth rather than about grouping. The common failure is reaching for the two weakest devices while the strongest one sits unused because it consumes area and looks like doing nothing.

6. B

The F is what happens when somebody skims text-heavy pages with no visual structure and low commitment: they read less of each successive line and end up sampling down the left edge. It is a symptom of a page that gives the reader nothing to navigate by, so designing to reinforce it makes the skimming permanent. The same research recommends breaking it with structure.

Module test — 2. Space, grouping and the grid

1. C

Grouping cues have different strengths: a shared region beats proximity, and proximity beats similarity. Adding a colour to a layout whose distances are wrong does not replace the wrong reading, it adds a second and weaker one, so the reader is offered two incompatible structures and the eye flips between them. A distance error is repaired with distance.

2. D

A difference registers in proportion to what it is compared against, not in absolute terms. Four pixels added to a 4px border is obvious; four pixels added to a 40px gap is invisible. A ramp with small steps at the bottom and doubling at the top keeps each step perceptible, while a constant 4px step stops carrying information as the numbers grow.

3. A

Padding is part of the object and travels with it, because it is what makes the thing look like itself. Margin describes how this object sits beside that one, which is the container's business. A component that owns an external margin accumulates overrides — a last-child rule, a modifier for a different page — until nobody can say what its natural spacing is.

4. B

Because both quantities sit inside the same logarithm, a factor of two applied to either has the same effect on the predicted time. Width is usually far cheaper to change than distance, since enlarging a target is a padding change and moving it is a layout argument. That is why widening is the first move and rearranging is the second.

5. C

Four mechanisms break it: the half-leading model puts the baseline somewhere that depends on font metrics you cannot see, arbitrary content heights push everything after them off the grid, platform form controls have their own metrics, and reflow changes line counts on every device. What survives is the useful part — a small consistent set of vertical values tied to the line height.

6. D

The margin is what tells a reader where the content ends and the world begins, so it should be the strongest boundary on the page. When an internal gap exceeds it, the strongest boundary sits inside the layout: the content appears to slide off the edge, and the internal gap starts reading as a divider that splits the group in two.

Module test — 3. Type, mechanically

1. A

At the end of each line the eye makes a ballistic jump back and down, and the longer the line, the more often that jump lands on the line above or below the one it wanted. The reader re-reads or skips without noticing, and reports being tired. The repair is a maximum width of roughly 30 to 34em, which brings the measure to 45 to 75 characters.

2. B

Point size measures the invisible body the letters sit on, inherited from the metal block, not the letters themselves. What you perceive as size is the x-height, because that is the height of most letters in a word. To match the old face you would need about 21 pixels, or you can hold the ratio with font-size-adjust, which also rescues a fallback font.

3. C

The setting engine absorbs the whole adjustment in the word spaces, and the number available is roughly the number of words on the line. A 70-character line has about eleven spaces sharing the work; a 35-character line has about four, so each stretches four or five times as far, and the wide gaps on successive lines connect into a pale vertical channel.

4. D

Hyphenation is language-specific — the rules for breaking English words are not the rules for German or Hindi — so the browser consults a dictionary chosen by the declared language. With no language declared it has nothing to consult and silently does nothing. The CSS is present, nothing reports an error, and the feature never runs.

5. A

A lowercase word has a distinctive profile made by its ascenders and descenders, and part of recognition happens on that shape before the letters are resolved. Capitals flatten every word to the same rectangle, differing only in length. For a two-word label the cost is negligible, because you are recognising a short known string; over a paragraph it is real.

6. B

Italic is a Latin convention with a specific history. Devanagari has no italic tradition, and slanting it produces something that reads as a distortion rather than as stress; CJK typography marks emphasis with a heavier weight or with emphasis dots beside the characters. Weight and colour transfer across scripts; slant does not.

Module test — 4. Colour, measured

1. C

The L in HSL is a geometric position inside the RGB cube, not a measurement of brightness. The eye peaks in the green region, so luminance weights green at 0.7152 and blue at 0.0722. Yellow is red plus green at full strength and inherits almost all of it, landing near 0.93, while blue sits near 0.07 — the same declared lightness, twelvefold apart in light.

2. D

Flare: in a real room, light bounces off the glass, so a real black is never truly black. The constant is also why the scale caps at 21 to 1 rather than running to infinity, and why very dark pairs score better than they read — in daylight the actual flare is far higher than the constant assumes, and it swamps the difference.

3. A

Large text has a precise definition: at least 18 point, or 14 point bold, which is 24 CSS pixels or 18.66 pixels bold at the usual assumption. Eighteen pixels bold falls just under the bold figure, so the 4.5 to 1 requirement applies. Three-quarters of a pixel sounds like pedantry and decides whether a conformance claim is true.

4. B

Direct labelling removes two problems at once: the colour dependency, and the memory task of holding a mapping while looking somewhere else. A tested palette is a genuine improvement and still fails a greyscale photocopy or a bad projector, because the information is still carried by hue alone. Colour has to be the second encoding of something, not the first.

5. C

In a light theme a raised surface casts a shadow because there is somewhere darker for the shadow to go. On a near-black surface there is not, so elevation is expressed by lightness instead: the higher a surface sits, the lighter it is. That is why dark themes need more neutral steps at the dark end, where small lightness differences are hardest to see.

6. D

A screen adds light and a press subtracts it, and sRGB reaches much further into saturated blues, greens and oranges than four-colour process can. The colour did not change on the way to the press; it never existed there. Soft-proof against the printer's profile in Scribus or GIMP, and approve colour from a printed sample rather than a screen.

Module test — 5. Composition and the image

1. A

Occlusion is the strongest depth cue the visual system has: if A hides part of B, A is in front, unambiguously. With no clear overlap there is no evidence about order, so the depth reading becomes unstable, and continuity then fuses the two continuous contours into a single shape. Overlap clearly or separate clearly; the narrow band between is the fault.

2. B

A crop changes framing and nothing else. Perspective is decided at capture by the camera's position, so no rectangle drawn afterwards produces the portrait a longer lens from two metres would have given. A perspective correction can straighten converging verticals at a cost in resolution and stretching, which is a different operation and not a restoration.

3. C

An image has thousands of colours and therefore no single ratio, so the worst pixel under the text governs legibility. When the image is unknown in advance, a scrim tuned on a pleasant photograph will fail on somebody's picture of a whiteboard. A panel puts the text on a known colour, which makes the ratio computable rather than hopeful.

4. D

Isolation is an emphasis channel, and this is how it gets spent. Evenly distributed empty space is the default and therefore communicates nothing. Space that is deliberately uneven tells the reader where to look for free, in a channel that survives greyscale, sunlight, a photocopy and every common colour vision deficiency.

5. A

A break is loud because the eye has stopped inspecting and started checking a prediction, and a violated prediction is the cheapest possible thing to notice. Repeat the break four times and it becomes a rule of its own: two patterns, and neither emphasises anything. One break per pattern, and two only when they are clearly different kinds of break.

6. B

The square crop is the tightest of the three, taking the central 1080 and discarding 420 pixels from the top and bottom, so the safe region is the intersection of all the crops rather than any one of them. Designing to the 4:5 crop puts the headline outside the square. Platform overlays are a further margin on top of that, not a substitute for it.

Module test — 6. Real screens, real thumbs, real paper

1. C

The vh unit refers to the viewport with the browser bars hidden, so while the address bar is showing — which is most of the time — the visible area is smaller and the bottom of the section is off screen. svh is the smallest viewport and the safe default for anything that must be fully visible. dvh changes as the bars move, which resizes the layout mid-scroll.

2. D

A type decision becomes a layout bug: the whole page zooms when the field receives focus, the user loses their place, and zooming back out is manual. It is one of the most common faults on the mobile web and one declaration removes it. The threshold is exactly 16 — 15.9 triggers it.

3. A

Odd numbers are the sign of a breakpoint found by widening and narrowing the window until something failed: the measure ran past 75 characters, a heading wrapped into a bad shape, or cards dropped below about 200 pixels each. Round numbers named after devices usually mean somebody copied them, and they age as soon as the devices change.

4. B

Reduced motion means less movement, not no change. The blanket rule is a reasonable safety net and a crude one; the better version replaces movement with a fade, so a panel still announces that it appeared and a tap is still confirmed. If the product becomes unusable with the setting on, the motion was carrying information nothing else carried.

5. C

The original objection was that a ring appeared after every mouse click. :focus-visible applies the indicator for keyboard focus and not for mouse clicks, which removes the objection entirely while keeping the indicator for the people who navigate without a pointer. The indicator itself needs 3 to 1 against whatever sits behind it.

6. D

Paper shifts in the stack, so the guillotine lands within about a millimetre of the mark and sometimes further. Without bleed, a cut that falls half a millimetre outside the line leaves a white sliver down one edge. The companion number is the safe margin: nothing important within 3 to 5mm inside the trim, because a cut can fall inwards too.

Module test — 7. Making the decision once

1. A

Primitives describe values and semantic tokens describe jobs, pointing at primitives. When a dark theme, a rebrand or a white-label version arrives, you remap a dozen lines in the middle layer and every component follows. Collapse the two layers and the same change means editing every component that used the raw value, and you will miss some.

2. B

A semantic layer should stay roughly stable while the number of screens grows, because roles recur — text, muted text, surface, border, action, danger — and new screens do not invent new ones. If the layer grows in step with the product, each entry is naming a place. The test is to ask what would break if a token were used somewhere else; if nothing would, it was a location.

3. C

Same structure and same purpose with a different appearance is a variant; a different content model is a new component, however similar the outlines look. Slots genuinely help one component serve many layouts, but they do not merge two content models — they hide the difference behind a region that accepts anything, which is how a card ends up with nine properties.

4. D

The user is told the door is shut and left to hunt for the reason. The stronger pattern in most forms is to keep the control enabled and explain on activation: press Submit, show what is missing, and move focus to it. If a disabled state stays, make it legible anyway — the contrast exemption permits low contrast, it does not require it.

5. A

Handing over token names rather than measurements makes deviations visible, which is most of the point. A value with no token behind it is either a deliberate exception with a written reason, or an accident — and a question from the person implementing it is the cheapest way to find out which, before it is set in code as an untraceable number.

6. B

The three scales cost about half an hour and pay back on the first artefact, because they remove the accidental near-differences that make work look amateur. Everything beyond them waits for a trigger: the same decision made three times, or a second person joining. Building further on a piece that will be made once is work performed on a thing that will not recur.

Design, Before Any Software: final examination

1. A ***1. How seeing actually works***

Vision is not a camera panning across a page. The eye lands three or four times a second, and during each jump you are effectively blind. Sharp detail exists only in a patch about the width of a thumbnail at arm's length, so a viewer builds an impression from a handful of fixations rather than from the layout as a whole.

2. B ***1. How seeing actually works***

Blurring simulates what peripheral vision keeps, so it tells you which elements survive as distinct blobs and can therefore be found by an unguided glance. Desaturating and squinting tells you the value structure and therefore the rank. An element can be findable and still not be first, and a design can have a first thing that dissolves at the edge of vision.

3. C ***1. How seeing actually works***

A channel used everywhere stops being a channel. Once orange marks every heading, it carries the meaning heading and nothing else, and making something genuinely important now needs a fourth device that has not been spent. This is the same mechanism as emphasis inflation: the more surfaces a signal covers, the less any one of them receives.

4. D ***1. How seeing actually works***

Peripheral motion detection is involuntary. A moving element does not enter the ranking with the other channels; it takes a fixation from a reader who had no opportunity to decline. That is why operating systems expose a reduced-motion setting, and why an autoplaying element beside an article is a different kind of decision from a large headline.

5. A ***1. How seeing actually works***

Gaze direction transfers: viewers shown a model looking at a headline fixate that headline sooner and more often than viewers shown the same model looking at the camera. A face is already at the top of the emphasis budget whatever its size, so the productive move is to point it into the layout rather than to try to out-style it.

6. B ***1. How seeing actually works***

The optical centre sits a few per cent above the geometric one, consistently enough that typographers and picture framers have compensated for centuries — which is also why a traditional book page has a larger bottom margin than top. On a 1000-pixel frame the correction is roughly 50 pixels. Centre the mass, not the bounding box.

7. C ***1. How seeing actually works***

Shape contrast works because it is rare: one deviant silhouette in a field of rectangles is found from the periphery, in greyscale, at any size. Five circles are a style, and a style is not an emphasis. The first instance of any formal device is information and the fourth is wallpaper.

8. D ***1. How seeing actually works***

Fixing colour before structure is painting a wall that is not straight: the work is fine and it will be done twice. Hue cannot introduce a lightness difference that is not there, so a hierarchy fault survives every palette. Diagnose hierarchy, then value, then spacing, then alignment, then inventory.

9. A ***2. Space, grouping and the grid***

Space at the edge of a group holds it together; space inside it pulls it apart. Trapped whitespace larger than the group's own margin is read as a boundary, whether or not one was intended, which is why a wide channel between two columns of a single block splits it into two blocks.

10. B ***2. Space, grouping and the grid***

The eye completes implied shapes and boundaries, so aligned columns imply the rules without anybody drawing them. That is why deleting rules almost always improves a table: the structure survives on closure and continuity, and every removed line is one fewer element competing for attention.

11. C ***2. Space, grouping and the grid***

It is arithmetic and nothing else: twelve gives halves, thirds, quarters and sixths from one grid without fractions. Sixteen columns give halves, quarters and eighths and no thirds, which is why teams that adopt sixteen end up hand-placing anything three across. On a phone, twelve columns is theatre.

12. D ***2. Space, grouping and the grid***

A span covers its columns plus the gutters between them, not around them: four columns and three gutters, so 4 times 78 plus 3 times 24, which is 384. Placing the element at a round 400 instead puts it 16 pixels off the grid — exactly the size of error that vernier acuity detects and nobody can name.

13. A ***2. Space, grouping and the grid***

Adjacent vertical margins in normal flow merge into the larger of the two rather than adding, and a parent's margin can collapse through to a child, which is a real source of gaps that are not the number anybody set. The gap property does none of that: it puts the stated space between things and nothing else.

14. B ***2. Space, grouping and the grid***

Size solves accuracy; spacing solves the consequence of a miss. With no clear space between them, a slightly low tap lands on the neighbour, and hitting Delete instead of Archive is much worse than hitting nothing. A working minimum is eight pixels of clear space, and more where the two actions differ in consequence.

15. C ***2. Space, grouping and the grid***

The threshold rule applies to composition as much as to type: a near-miss reads as a mistake, in the same way a 17-pixel heading over 16-pixel body text does. Either balance the weights, or commit to the imbalance far enough that it is obviously a choice. There is nothing useful in the narrow band between.

16. D ***2. Space, grouping and the grid***

The visible mass of a word sits high inside its own line box, because the box includes room for descenders that a word such as Save never uses. The space below therefore looks larger than the space above. Shave one to two pixels off the bottom padding at body size, and note that the same button looks right on a label that does have a descender.

17. A ***3. Type, mechanically***

It is a specification rather than a ranking of quality. A text face is drawn with sturdier strokes, open apertures and generous spacing so that it survives small sizes, and looks bland when enlarged. The error is usually made by somebody who chose a face from a specimen shown at 90 pixels.

18. B ***3. Type, mechanically***

The threshold rule again: two faces that are slightly different read as an error, because the reader's first hypothesis for a small difference is that something failed to load. Pair across a class boundary so the difference is obviously deliberate, and match x-height and apparent weight so the pair does not look like two sizes.

19. C ***3. Type, mechanically***

Most typefaces ship proportional figures, where a 1 is narrower than a 0, because that sits better inside a sentence. Anything that changes in place, or sits in a column, wants tabular figures, where every digit occupies the same width. One declaration of font-variant-numeric fixes a class of small ugliness that people otherwise attack with fixed widths.

20. D ***3. Type, mechanically***

Indian grouping takes the last three digits and then pairs: 12,00,000 rather than 1,200,000. This is a formatting decision rather than a typographic one, and Intl.NumberFormat with the en-IN locale handles it, along with every other locale's conventions including the comma-as-decimal used across much of Europe. Hard-coding one grouping is a decision to be wrong somewhere.

21. A ***3. Type, mechanically***

Capitals are taller and denser than the lowercase around them, so a run of them makes a clot in the paragraph's even grey. Real small capitals are drawn at the correct stroke weight and sit in the texture; scaled-down capitals look visibly thinner, which is why faked small caps always seem slightly wrong. Where the face has none, about eight to ten per cent smaller gets most of the benefit.

22. B ***3. Type, mechanically***

balance evens the line lengths across a short block, which is what a heading of up to about four lines needs, and it is deliberately limited to short blocks because the computation is expensive. pretty is the body-text counterpart: it leaves the lines alone and avoids a single word on the last line. Both degrade harmlessly where they are not supported.

23. C ***3. Type, mechanically***

The total vertical extent of a Devanagari line is much greater than a Latin line at the same size, because matras sit above the headline and conjunct consonants stack downwards. Leading tuned for Latin ascenders and descenders therefore clips or collides. Roughly 1.6 to 1.8 is the working range, and Urdu set in Nastaliq commonly needs 2.0 or more.

24. D ***3. Type, mechanically***

A desktop licence typically covers installing the font on a stated number of machines to produce artwork. Embedding it in a website, an app or a distributed PDF is usually a separate licence, often sold by pageview. The related trap is handing a client the font file itself, which a desktop licence rarely permits even though handing over outlined artwork is fine.

25. A ***4. Colour, measured***

Mixed temperatures produce a muddiness that is very hard to diagnose after the fact, because the greys are scattered across a layout rather than sitting side by side. The diagnostic is to pull every grey in the design out as large swatches in a row, where a mismatch is obvious immediately.

26. B ***4. Colour, measured***

Near black, the same numeric step is less visible than it is near white, and ambient light bouncing off the screen — the flare the contrast formula models with a constant — washes out the dark end further. Dark interfaces feel this hardest, because three or four surface levels all live down there. Spread the dark end further than the arithmetic suggests, and check it in a bright room.

27. C ***4. Colour, measured***

Semi-transparent text has no colour of its own. Move the same text from a white page onto a pale grey card and the ratio drops without anything in the text changing. Either compute the blended value against the real background, or set solid colours — and be aware that many people check the declared colour rather than the rendered one.

28. D ***4. Colour, measured***

One hex step separates a pass from a fail, which is why the habit of lightening secondary text until it looks refined so often fails by a margin nobody can see on a good monitor. It is also the argument for headroom: at 4.6 to 1 there is nothing left for sunlight, a cheap panel or an older reader's eyes.

29. A ***4. Colour, measured***

Stock-market direction is the most useful example in the whole culture discussion, because an ordinary designer would never think to check it and the consequence is not offence but a factual misreading. The general position holds: there is no reliable universal mapping from hue to meaning, so back every colour-carried meaning with a second signal.

30. B ***4. Colour, measured***

Simulators model dichromacy — the complete absence of a cone type — reasonably well, and anomalous trichromacy, which is much more common, only approximately, and they represent an average rather than any individual. Treat a distinction that disappears as conclusive and a distinction that survives as encouraging.

31. C ***4. Colour, measured***

Where the hue changes fast, the perceived lightness jumps, and readers see an edge in the data that the data does not contain. Sequential quantities want one hue varying in lightness; a quantity diverging either side of a midpoint wants two hues meeting at a light neutral, with blue to orange as the standard colour-blind-safe pair.

32. D ***4. Colour, measured***

Four inks are laid by four plates and they never register perfectly. On a large solid, a fraction of a millimetre of misalignment is invisible; on nine-point text it fringes every letter and blurs the whole block. The rule is rich black for large solids and 100 per cent K alone for small text and thin rules.

33. A ***5. Composition and the image***

A horizon in the middle divides the frame into two halves that compete and neither wins. Nudging it decides which half the picture is about, and that alone justifies the grid overlay. There is no perceptual evidence that the intersections are special points; they are a by-product of dividing by three.

34. B ***5. Composition and the image***

Gaze is the strongest implied line available, and an implied line has to resolve inside the frame. With the edge close in front of them, the line ejects the viewer from the picture immediately. This is also why a thirds crop can make a portrait worse: pushing a right-facing face onto the right-hand line removes the space the picture needed.

35. C ***5. Composition and the image***

A quarter of the area is half the linear dimensions, so 2000 by 1500 pixels, and dividing by 300 gives 6.7 by 5 inches. Cropping spends resolution permanently, which is why you crop last, for a named output, and never over the original. Upscaling invents detail rather than recovering it.

36. D ***5. Composition and the image***

Aerial perspective is the cue you can apply deliberately to any flat artwork: distant things are lower in contrast, less saturated and nearer the background's lightness. Dropping an element to 40 per cent opacity over a light ground does all three at once, which is also the principle behind disabled states and behind a modal dimming the page.

37. A ***5. Composition and the image***

A shadow tells you where something sits only because the visual system assumes a single light source, by default above. Three contradictory claims about a light that does not exist cancel each other, and the elements stop being ordered in depth at all. One direction, one angle, one softness, applied everywhere.

38. B ***5. Composition and the image***

Three objects on a page make four shapes, and the fourth is usually the largest and rarely decided. Looking only at the white finds narrow tapering slivers, ragged channels between columns and trapped pockets — problems that hours of looking at the elements will not surface, because when you look at the elements you see the elements.

39. C ***5. Composition and the image***

The diagnostic is not that viewers disagree with the designer, which is an opinion, but that they disagree with each other, which is a measurement. Two elements of similar prominence leave the eye oscillating and never settling, so which one a given person lands on is effectively random.

40. D ***5. Composition and the image***

Every upload is re-encoded, and fine detail, subtle gradients and thin light text are exactly what lossy compression throws away. Text over an image therefore wants to be reasonably large and solid rather than thin and pale, and large flat gradients often return with visible banding — worth checking on the platform rather than in your editor.

41. A ***6. Real screens, real thumbs, real paper***

Mobile viewport widths cluster tightly, and 360 is the safe floor reported by a great many Android devices across India, Africa and Latin America. Designing at the floor means the layout holds everywhere above it; designing at 393 leaves a class of breakage — a fixed-width element, a long unbreakable string — undiscovered.

42. B ***6. Real screens, real thumbs, real paper***

The hardware has three times as many pixels as the CSS layer admits to, so a 1x export looks soft on almost every modern device. A vector is resolution-independent and sharp at any ratio for a fraction of the bytes, which is the strongest argument for drawing interface graphics as vectors — and Inkscape does it free.

43. C ***6. Real screens, real thumbs, real paper***

Four hundred per cent zoom on a 1280-pixel screen gives the equivalent of a 320-pixel viewport, and the criterion is that reading a line should not require scrolling in two directions. A fixed two-column layout with a sidebar fails; a layout that genuinely stacks passes. Anybody who built the phone layout first has already satisfied most of it.

44. D ***6. Real screens, real thumbs, real paper***

Focus moves through the document in source order, so a visual reorder makes the focus ring appear to jump at random for anybody navigating by keyboard. It passes every visual review, because visually nothing is wrong. Reordering is fine when the two orders agree at every breakpoint and is a defect when they do not.

45. A ***6. Real screens, real thumbs, real paper***

Three faults stack: the label vanishes at the moment the user needs it to check what they entered, placeholder text is usually low-contrast by default and often fails the contrast requirement, and assistive-technology handling of placeholders is inconsistent. Every field needs a visible, persistent label, which is also clickable and therefore doubles the target for free.

46. B ***6. Real screens, real thumbs, real paper***

Alt text conveys what an image is doing in the page rather than what it looks like. A decorative icon beside its own label is redundant, so it takes aria-hidden or an empty alt and is skipped. The commonest error is not missing alt text but unhelpful alt text, which is worse than none because it cannot be skipped.

47. C ***6. Real screens, real thumbs, real paper***

Under about a hundred milliseconds a response feels instantaneous, and putting a spinner there makes the interface feel slower rather than faster, because a brief flash reads as a stutter. The bands above it need an indicator, and past ten seconds — the limit of held attention — they need a real estimate and a way to leave.

48. D ***6. Real screens, real thumbs, real paper***

A glued spine curves, and a portion of the inner margin disappears into the curve — commonly five to ten millimetres more is needed inside than outside. This is the opposite of what an untrained layout does. Saddle-stitched booklets have their own version of the problem, creep, which pushes the centre pages outwards as the stack thickens.

49. A ***7. Making the decision once***

Name by role, never by appearance: the name outlives the value. The exception is the primitive layer, where describing the value is the job, so blue-500 is a correct primitive name and an incorrect semantic one. Once a name and its value disagree, every new person has to learn that red means orange here.

50. B ***7. Making the decision once***

T-shirt sizes run out, which produces xxxl and then xxxxl, and they carry no arithmetic — you cannot reason that lg is twice sm. Numbered steps borrowed from font weights leave room to insert 150 between 100 and 200. The one place t-shirt sizes work is a small closed set, such as component sizes, that will not grow.

51. C ***7. Making the decision once***

A colour used as a background needs a partner for the text that sits on it, and the pair has to travel together or somebody will put dark text on a dark action colour. Naming the relationship makes the pairing survive being copied into a component by a person who was not thinking about contrast — which is one of the commonest accessibility regressions.

52. D ***7. Making the decision once***

Number the position on the scale, not the value at that position. A token called space-16 that is set to 20 pixels is a lie, and everybody who reads the name assumes sixteen. space-3 can be 16 pixels today and 18 after a density review, and the name remains true. The same applies to type: text-lg survives a change and text-20 does not.

53. A ***7. Making the decision once***

SearchAndFilterBar is two components. The related signals are a property that changes the meaning of other properties, properties that are only valid when another has a particular value, documentation that needs the word except, and people copying the component instead of using it. Variant count on its own is not a signal, because axes are how a component is meant to grow.

54. B ***7. Making the decision once***

The reasons are specific and measurable: one set drawn on a 24-pixel grid and another on 20, 1.5-pixel strokes against 2-pixel, rounded caps against square, a camera drawn with a lens and a flash against one drawn as a rectangle and a circle, and different optical sizing of round shapes. Using one set is the single decision that fixes all of them.

55. C ***7. Making the decision once***

Stale documentation is worse than none, because it is wrong with authority and somebody will build from it. Only two things keep documentation alive: proximity, so that it is updated when the code is, and use, so that the examples are the real components and cannot drift. A static PDF of specifications is stale within a month, always.

56. D ***7. Making the decision once***

The legitimate cases are narrow: a moment that must not feel routine, a genuinely new content type, a different audience or context such as marketing against the product, and a time-limited experiment. Boredom is not on the list, and the answer to it is usually that the hierarchy is flat rather than that the system is wrong. Every exception has to be deliberate, documented and eventually resolved.