Engineer

Responsive design

a layout that adapts to whatever screen it is read on

What it is

Responsive design is the practice of building one page whose layout adapts to whatever screen it is read on, from a narrow phone held upright to a wide monitor, instead of building a separate site for each kind of device. The longer spelling, responsive web design, is the title of the article that named it: Ethan Marcotte's piece in A List Apart, published on 25 May 2010. He drew the idea from what he described as an emergent discipline called responsive architecture, in which walls and installations move and change shape as people come near them.

Marcotte called fluid grids, flexible images and media queries "the three technical ingredients for responsive web design". A fluid grid sizes its columns in proportions of the space available rather than in fixed pixels, so the layout stretches and shrinks with the window. A flexible image is never allowed to grow wider than the column that holds it. A media query lets a stylesheet change its mind at the points where proportions alone stop working: when three columns become too narrow to read, for example, the page switches to one.

The Media Queries Level 4 specification, a W3C Candidate Recommendation Draft dated 19 February 2026, defines a media query as a logical expression with one of two values, true or false: a test of some aspect of the browser or device that is showing the document. The test responsive layouts lean on is `width`, which for a screen is the width of the viewport, scroll bar included. Lengths in a query are the same CSS lengths as anywhere else, so a pixel there is a CSS pixel, not a physical one, and an `em` is measured from the browser's initial font size, which the reader can change, never from any size the page itself sets.

None of this works on a phone unless the page asks for it. Without a viewport meta element, mobile browsers lay a page out as if the screen were a desktop window, typically 980 to 1024 CSS pixels wide, and then shrink the result, so every media query sees a desktop. The CSS Viewport draft calls that a mitigation for pages designed only for large screens, unnecessary for pages that work well on small ones, and the line `width=device-width, initial-scale=1` in the example below is how a page says it is one of them.

Use the concept in a decision

Put “What is responsive design?” to work in a real decision. We can help relate the concept to the system you are building or buying. Interfaces that load quickly, work on a phone and behave the same in every browser your customers happen to use.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Directed full-stack GIS map development — full-stack mapping, from database to browser

Directed full-stack GIS map development Get in touch →

Why it matters

Marcotte’s argument was about maintenance as much as appearance. Rather than tailoring disconnected designs to an ever growing number of devices, he proposed treating them as facets of the same experience. One page, one address and one stylesheet mean that a correction reaches every reader at once, and a new size of screen is served by the same site, because the layout was never written for a particular size in the first place.

Zoom is the reason many teams underestimate. Zooming in makes the CSS pixel bigger, so the viewport, counted in CSS pixels, gets narrower. The Web Content Accessibility Guidelines 2.2, a W3C Recommendation dated 12 December 2024, require at level AA in Success Criterion 1.4.10 Reflow that content can be used without scrolling in two dimensions when the available width is 320 CSS pixels, and they spell out that this is what a window 1280 CSS pixels wide becomes at 400% zoom. At that starting width, every media query on the page sees a reader who zooms a laptop browser to 400% as using a 320-pixel viewport. The W3C’s explanation of the criterion observes that sites commonly use responsive design to fit smaller viewports, and that content which fits a small viewport helps people who zoom in on larger devices, not only people on phones.

Bytes are the third reason. The HTML Standard lists three separate reasons to offer several files for one image: choosing by viewport, art direction, and choosing by image format. A photograph shown across a phone needs far fewer pixels than the same photograph across a large monitor, and with `srcset` and `sizes` the browser can fetch the smaller file without any script.

Width is not the only thing a page can respond to. Media Queries Level 4 also describes the primary pointing device: `(pointer: coarse)` matches a pointing device of limited accuracy, such as a touch screen, and `(hover: none)` matches a primary pointing device that cannot hover, or no pointing device at all. Marcotte's article already suggested using media queries to give links bigger targets on small screens.

What it looks like in practice

A list of cards that shows one column on a phone, two on a tablet held sideways and three on a wide screen starts with the viewport line and an image the browser can choose for itself:

```
<meta name="viewport" content="width=device-width, initial-scale=1" />

<img
  src="pool-400.jpg"
  srcset="pool-400.jpg 400w, pool-800.jpg 800w, pool-1600.jpg 1600w"
  sizes="(max-width: 40em) 100vw, (max-width: 64em) 50vw, 33vw"
  width="1600"
  height="900"
  alt="A tide pool at low water"
/>
```

The stylesheet carries the layout:

```
.cards {
  display: grid;
  grid-template-columns: 1fr;
  gap: 1rem;
}

@media (width >= 40em) {
  .cards {
    grid-template-columns: repeat(2, 1fr);
  }
```

```
}

@media (width >= 64em) {
  .cards {
    grid-template-columns: repeat(3, 1fr);
  }
}

.cards img {
  max-width: 100%;
  height: auto;
}
```

The grid is the fluid part: `1fr` columns share whatever width there is, so nothing has a fixed size. The two media queries are the switches, and because they are written in `em`, they follow the reader's default text size. Someone who has set a larger default font gets the single column at window widths where others already see two, which is exactly what the longer lines would need. The last rule keeps every image inside its column.

The `sizes` attribute tells the browser, before it has laid anything out, how wide the image will be shown: the first condition that is true wins, and the last value applies when none is. At a viewport 39em wide, the first condition is true, so the image counts as 39em wide; at 41em, the second one is, and the image counts as half the viewport, 20.5em. The slightly wider window needs the smaller file, because the layout has changed, and the HTML Standard works through the same effect in its own example. From those numbers and the `w` descriptors in `srcset`, the browser works out how dense each file would be on this screen and may pick any of them, taking the screen's pixel density, the zoom level and possibly the network into account. The `width` and `height` attributes let it allocate space for the image before the file has been downloaded.

A portfolio entry on making the site's browser checks faster shows the same idea turned into a test. In that work, which was carried out in August 2026, the checks that drive a headless browser were given one shared table of twenty-two window shapes, derived from every media query the stylesheet actually contains, and the layout check, which looks at the page at every window shape the design draws, went from 95 seconds to 405 as a result. The shapes come from the stylesheet, not from a list of devices, so the breakpoints the design actually has are the ones the pages are checked at.

Where it goes wrong

Breakpoints that ask about the device. Marcotte's 2010 examples queried `max-device-width`, the width of the whole screen. Media Queries Level 4 now deprecates `device-width`, `device-height` and `device-aspect-ratio`, says authors must not use them, and notes that they are often used as a proxy for detecting mobile devices, while they say nothing about how much space the page actually has. Query `width` instead, and put each breakpoint where the content stops working, not where a popular phone happens to end.

A gap between two breakpoints. A stylesheet with `(max-width: 320px)` for one layout and `(min-width: 321px)` for the next looks complete, but the specification points out that a viewport between 320 and 321 pixels wide matches neither, and fractional widths are real: they come from screens with non-integer pixel densities and from zoom. At those widths, neither block of styles applies. Use the range syntax, `(width <= 320px)` and `(width > 320px)`, so that every width falls into exactly one of them.

Zoom switched off. A viewport line with `user-scalable=no`, or with a `maximum-scale` below 2, fails the W3C's test rule for Success Criterion 1.4.4 Resize Text, which asks that text can be enlarged to 200 percent. The rule notes that desktop browsers pay no attention to the element, and that on mobile most current browsers disregard it out of the box or give the reader an accessibility setting to zoom regardless, so the people still affected are those on older mobile browsers. Leave both properties out and make the layout survive zoom instead. A smaller font in a narrow breakpoint needs the same care: the W3C's explanation of Reflow allows it, but only if the reader can still get text to twice its normal size in some way.

Images that ignore the layout. With `w` descriptors in `srcset` and no `sizes`, the HTML Standard makes the browser assume the image fills the whole viewport, so a picture shown in a third of the screen may download a file about three times wider than it needs. Percentages are not allowed in `sizes`; use `vw`. And when `srcset` uses `w` descriptors, browsers that understand `srcset` ignore `src`, so changing only `src` changes nothing for most readers. Write `sizes` to match the CSS, and change both together when the layout changes. Choosing images in a script is no fix: the standard warns that it can download the image twice, or not at all for readers with scripting turned off.

One wide element pushes the page sideways. A data table, a long line of code or an embed with a fixed width can be wider than 320 CSS pixels and make the whole page scroll horizontally, so that every paragraph has to be read by scrolling back and forth. The W3C's explanation of Reflow excepts content that needs two dimensions, such as data tables, maps and video, but not the paragraphs around them. Put the table in its own container that scrolls by itself with `overflow-x: auto`, and give images and video `max-width: 100%`, so that only the part that needs it scrolls.

Taking a narrow screen for a touch screen. A laptop zoomed to 400% is narrow and has a mouse; a large tablet is wide and has no hover at all. Layout belongs to `width`, and target sizes and hover effects belong to `pointer` and `hover`. Media Queries Level 4 advises authors to consider `any-pointer` and `any-hover` as well, since a device can have more than one way to point.

Further reading

- Responsive Web Design — Ethan Marcotte's article of 25 May 2010 that named the approach and its three ingredients.
- Media Queries Level 4 — the `width` feature, the range syntax, relative units, pointer and hover, and why `device-width` is deprecated.
- HTML Standard: Images — `srcset`, `sizes`, the three reasons for several files, and the worked 29em and 32em example.
- Understanding Success Criterion 1.4.10: Reflow — why 320 CSS pixels, how zoom relates to it, and which content is excepted.
- Meta viewport allows for zoom — the W3C test rule for `user-scalable` and `maximum-scale`, with its note on browser support.
- CSS Viewport Module Level 1 — the viewport meta element, its properties and why mobile browsers assume a desktop width.

Glossary

Choose for your actual constraints

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

See what it costs Get in touch →

Make the terminology practical

Ask how “What is responsive design?” applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/responsive-design/