1

Understanding Flexible Layouts

The term "flexible layouts" can mean different things to different people, so let's make sure we're all on the same page before we dive into designing and building them. In this chapter, you'll learn the defining characteristics of each of the main layout types—fixed-width, liquid, elastic, and hybrid—described in the Introduction. We'll focus on the benefits and pitfalls of each so that you can decide which type might be right for your particular site.

Types of Layouts

We can group web-page layouts into three categories based on how their width is set: fixed-width, liquid (or fluid), and elastic. It's also possible to combine these layouts into hybrid layouts by mixing units of measurements; each column of the design can use a different unit. In any of these four types of layouts, any number of columns or aesthetic themes is possible; the type simply establishes how the browsing device determines how wide to make the layout appear to the user.

Fixed-width: Rigid Pixels

Fixed-width layouts are the designs you're most used to seeing—and probably making, since you're reading this book to learn the alternatives. The width of the overall layout of a fixed-width design is set to a value in pixels that's decided by the designer. Usually, the designer chooses a width based on one of the common screen resolutions, such as 800 by 600 or 1024 by 768.

Fixed-width designs are rigid: they don't change size based on any variations in the user's setup (**Figure 1.1**). This can allow you to design a graphically rich site that holds together well and looks consistent across a variety of user setups. If you have done your homework on the target audience of your site, you can design a layout that fits nicely in the majority of users' browser windows, and you can make sure the lines of text are set at an optimal width for ease of reading—at least, if you assume a couple things.

FIGURE 1.1 The same fixed-width layout in two differently sized browser windows.

FIXED-WIDTH DESIGNS ARE NOT EVIL!

This chapter does a fair amount of beating up on fixed-width designs and a lot of singing the praises of liquid and elastic designs. This is simply because that's what the purpose of the book is: promoting and teaching flexible layout techniques, and what they have to offer. But I want to stress that I don't think fixed-width designs are "wrong;" they are definitely appropriate in certain situations, as we'll discuss in more detail at the end of this chapter.

SCREEN RESOLUTION DOES NOT EQUAL BROWSER WINDOW SIZE

The biggest problem with fixed-width layouts is that they essentially depend on you making a guess as to what width will work well for the largest number of your users. Even if your web statistics software can tell you the screen resolution of each of your users—heck, even if you're making an intranet and are certain that only a single resolution will be used—it's simply not the case that screen resolution matches the browser window width all the time. Some people don't browse with their browser window maximized (admittedly a small number, but growing as monitors and resolutions increase in size). Also, some people use browser sidebars that can take away hundreds of pixels from the available width.

Fixed-width designs are always going to result in some segment of your audience seeing a design that is either too wide for their windows (necessitating the dreaded horizontal scrolling) or too narrow (leaving oceans of space on one or both sides of the layout). And based on my experience with user testing, many people get almost as distressed about "wasted space" in their browser as they do about horizontal scrolling!

RESEARCH ON BROWSER WINDOW SIZES

In October 2006, Thomas Baekdal published a very interesting report, "*Actual Browser Sizes*," on his web site, www.baekdal.com. He gathered three months' and five sites' worth of data on both screen resolutions and browser window sizes. The report states that while the majority of the tested users maximized their browsers—or at least came close—a significant number did not. For instance, users with 1024-by-768 resolutions—the most common by far—maximized about 80 percent of the time. His conclusion was that "in order to support 95% of your visitors, you need to design for a maximum size of 776x424px"—even though he found that only five percent of his users had 800-by-600 screens. Check out the full report at http://baekdal.com/reports/actual-browser-sizes.

NOT EVERYONE USES 16-PIXEL TEXT

If you know the size of the text you're working with, you can choose a fixed width to optimize the number of characters that appear on a line, or the line *length*, to aid readability. Print designers do this all the time; Robert Bringhurst's famous book *The Elements of Typographic Style* recommends line lengths of 45 to 75 characters, based on years of research on readability of printed text. More recent research into the readability of onscreen text has shown that longer line lengths, from 75 to 100 characters, result in faster reading speeds (though many of the tested users say they prefer shorter lines).

However, on the web, we can't know our users' text sizes. The default size for browsers nowadays is 16 pixels, and the vast majority of your users will leave their text set at this default. However, some users do change the default, or set up user style sheets to format text in a way that makes it easier for them to read. Even users who leave the text at the default have the option of bumping up the size on a per-page basis if a particular page of text is difficult to read (even text you set in pixels is resizable, except in Internet Explorer 6 and earlier). So, if you optimize line lengths for 16-pixel text, you may be optimizing readability for the majority of your visitors, but not for all. Don't get me wrong; designing for the majority is a good thing. Just don't fool yourself into thinking that "majority" is the same as "all."

Another problem is that these line-length studies don't take into account different disabilities that may lead certain groups to prefer much shorter or longer line lengths. Although I agree that it's often our job as designers to set things up for our users in the best way for them because they don't know what's best themselves, there are times when we need to trust them to be better informed than we are about what will best meet their needs. At the very least, we can optimize the design for what we think will help the majority of our users, but leave open the possibility for individual users to adapt our design to better meet their needs—an advantage that print and more rigid media do not enjoy.

Liquid or Fluid: Adapts to the Viewport

Liquid layouts, also known as fluid layouts, change in width based on the size of the user's viewport. Liquid layouts built with CSS may or may not have any width assigned to them. If they don't have a width assigned, they will fill up the user's viewport no matter how big or small it is (**Figure 1.2**).

■ **NOTE:** *Viewport* is a generic term for the viewable area of a page in the user's device. It's preferred over *window* because, after all, not every device uses windows (for example, mobile phones).

FIGURE 1.2 The same liquid layout changes width based on the browser window size.

If a designer does assign a width to a liquid layout, it will be measured in percentages, not in pixels. The percentage refers to the portion of the viewport it takes up.

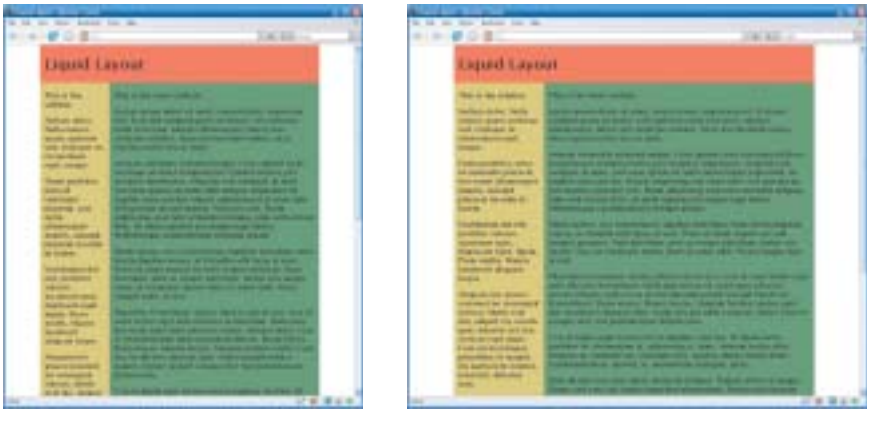

FIGURE 1.3 A liquid layout always adapts to the browser window size, even if you assign a smaller percentage width so that it never takes up the full width of the browser window.

TAKING ADVANTAGE OF SCREEN REAL ESTATE

When a liquid layout changes in size, all of the content within it—and often the background images as well—has to shift around on the page to fill up the space. As long as the content can wrap, this flexibility prevents horizontal scrollbars from appearing and makes full use of the screen real estate available on each user's device.

Once the content can no longer wrap, due to the fixed widths of images and other content, a horizontal scrollbar will finally appear, but this will happen only on the very narrowest of screens or on sites with very large fixed-width content. Using a liquid layout instead of a fixed-width one, it's much less likely that a user will miss important content hidden by a horizontal

scrollbar. Also, for users with very large viewports, more content will be visible on the page at once, decreasing the amount of vertical scrolling they have to do.

RESPECTING USER PREFERENCES

A liquid layout allows you to stop guessing at what works for your users and instead let them choose what page widths best meet their needs. There's no longer a need for a "best viewed at 1024 by 768" type of disclaimer on your home page. Even if a user can change his resolution to meet this requirement, the chances are slim to none that he's going to do so to accommodate your site. He's set that resolution for a reason: either he has no other choice given the constraints of his device, or it's the resolution that he enjoys or finds the most useful (for instance, some users stick with 800 by 600 because it makes everything bigger, which is easier to read). With liquid layouts, you don't need to worry about this anymore. Liquid layouts just work in a larger range of viewing scenarios, respecting users' preferences for how they like to view the web.

IMPROVING READABILITY

Horizontal scrollbars are the sworn enemy of readability. After all, scrolling continually back and forth to read across several lines of text does not make for the most enjoyable reading experience. With a liquid layout, horizontal scrollbars almost never happen, because users can size the window however they like to produce lines of text that they find the most comfortable to read and understand. Preferences for line length can vary by age, disability, and browsing device, so leaving widths adjustable can help a much broader range of people read your content efficiently than setting one fixed width might.

■ **NOTE:** One of the main arguments against liquid layout, actually, is that it decreases readability due to overly long lines of text on very large browser windows. This can certainly be the case when a liquid layout is implemented poorly or in certain user scenarios. We'll talk more about this challenge later in the chapter.

DEALING WITH HANDHELD DEVICES

While liquid layouts increase the range of sizes at which your web site can look good and work well, you're still probably going to need to set up a separate style sheet for handheld devices such as mobile phones and PDAs. It's just not possible to design something that works as well at 200 pixels wide as at 1200 (unless you're going for the old-school, plain-text look).

INCREASING ACCESSIBILITY FOR PEOPLE WITH DISABILITIES

Some users have disabilities that make line length even more essential for successfully reading and understanding content. If a user has a visual impairment that requires her to make her text size very large, she may prefer to browse with a very large window to allow more words to fit across each line. In a narrow, fixed-width layout, large text may allow only two or three words to fit on every line, making reading more difficult and resulting in a huge amount of vertical scrolling.

Other types of visual impairments may necessitate the use of screen magnifying software, which shows only a small, highly zoomed portion of the window to the user at one time. People who use screen magnifiers may prefer to make their windows very narrow so that the entire width of each line of text fits within their small, magnified area of the screen and they don't have to keep pushing the magnified view back and forth horizontally to read.

FIGURE 1.4 If the browser window is wide, screen magnifying software may not be able to show the entire line length of text within the magnified overlay, so the user has to push the overlay right and left to read the text (left). If the browser window is narrow, and the text width adjusts accordingly, the entire line of text can fit in the magnified overlay (right), making reading faster and easier.

Line length can also play a role in comprehension. For instance, many people with dyslexia find it easier to read and understand text with short line lengths.

Elastic: Adapts to the Text Size

Elastic layouts change in width based on the text size set in the user's browsing device. A user who has set a larger default text size will see a page where not only is the text bigger, but the entire layout is bigger proportionally than that seen by people with the default text size (**Figure 1.5**). If a user changes his text size while viewing the site, the entire layout width will also change proportionally, either wider or narrower, depending on whether he increased or decreased the text size.

FIGURE 1.5 The same elastic layout in two browser windows of the same size, but using different text sizes.

Like fixed-width layouts, elastic layouts always have a width assigned to them, but that width is set in a unit of measurement called an *em*. If you've used CSS for formatting text, you're probably familiar with ems (but we'll go over this in more depth when we start actually building elastic layouts). One em is equal to the font height, which in turn equals roughly two characters in width, since most characters aren't nearly as wide as they are tall. By setting the width of your layout in ems, you're essentially telling the design to be as wide as a certain number of text characters. That means when the text gets bigger, the whole layout has to widen as well to stay equal to the same number of now-larger text characters. It works in reverse, too, of course: smaller text sizes make the layout narrower.

THE DIFFERENCE BETWEEN ZOOMING AND TEXT RESIZING

In many browsers, you can use the zoom feature to make all pages act like elastic layouts. Zooming is not the same as resizing text, which is a separate browser function. Browser zoom functions scale images as well as text, as if you were truly moving closer into, or magnifying, the entire page. Think of it like changing the magnification on a PDF in Adobe Acrobat or Adobe Reader.

While all layout types zoom when you use a browser's zoom function, elastic ones also zoom when you use the text-resizing function as well as when the user starts out with a larger or smaller default text size. However, images do not change size in elastic layouts, as they do when you zoom a layout using the browser feature, unless you explicitly set them to. We'll cover how to do this in Chapter 9.

BROWSER SUPPORT FOR ZOOMING AND TEXT RESIZING

Resizing text is a good way to put the flexibility of your pages to the test, but not every browser handles it in the same way. Here are the basics on how to resize text as well as zoom entire page layouts in the major browsers.

TABLE 1.1 How to Resize Text or Zoom

BROWSER	HOW TO RESIZE TEXT	HOW TO ZOOM
Internet Explorer 7	View > Text Size or View > Text Zoom, and then choose a size keyword (e.g., Larger)	Ctrl + + (plus sign) to zoom in Ctrl + - (minus sign) to zoom out
Internet Explorer 6 and earlier	View > Text Size, and then choose a text size keyword (e.g., Larger)	Not available
Firefox 3	Ctrl/Command + + (plus sign) to increase Ctrl/Command + - (minus sign) to decrease	Deselect Zoom Text Only (View > Zoom > Zoom Text Only) Ctrl/Command + + (plus sign) to zoom in Ctrl/Command + - (minus sign) to zoom out
Firefox 2 and earlier	Ctrl/Command + + (plus sign) to increase Ctrl/Command + - (minus sign) to decrease	Not available
Opera	Not available on a per-page basis	Ctrl/Command + + (plus sign) to zoom in Ctrl/Command + - (minus sign) to zoom out
Safari	Ctrl/Command + + (plus sign) to increase Ctrl/Command + - (minus sign) to decrease	Ctrl/Command + + (plus sign) to zoom in Ctrl/Command + - (minus sign) to zoom out Available only as part of Mac OS system-wide zoom feature; not built into Safari on Windows.

Elastic layouts are the rarest type of layout because before CSS became usable for page layout, they were simply impossible to create with tables (the page-layout mechanism used before CSS and still in heavy use today). Since many browsers now allow you to "zoom" pages, I don't think that designers will increase their use of elastic layouts, which behave similarly to zooming. Nevertheless, there are definite benefits to elastic layouts.

INCREASED TYPOGRAPHIC CONTROL

Elastic layouts give you more control over where text falls in relation to other design components on the page. In other words, your design proportions stay intact. In a fixed-width layout, if a user increases his text size, the text has to wrap onto more lines and make its container taller (or overflow, which is even worse). Your meticulously crafted headline that fit so perfectly on one line might now be awkwardly broken between two lines, or a piece of text that needed to be near a certain image may have moved. The same thing can happen with liquid layouts when a user narrows her browser window. Elastic layouts can keep the same number of words and characters that appear on each line consistent no matter the size of the user's text or window.

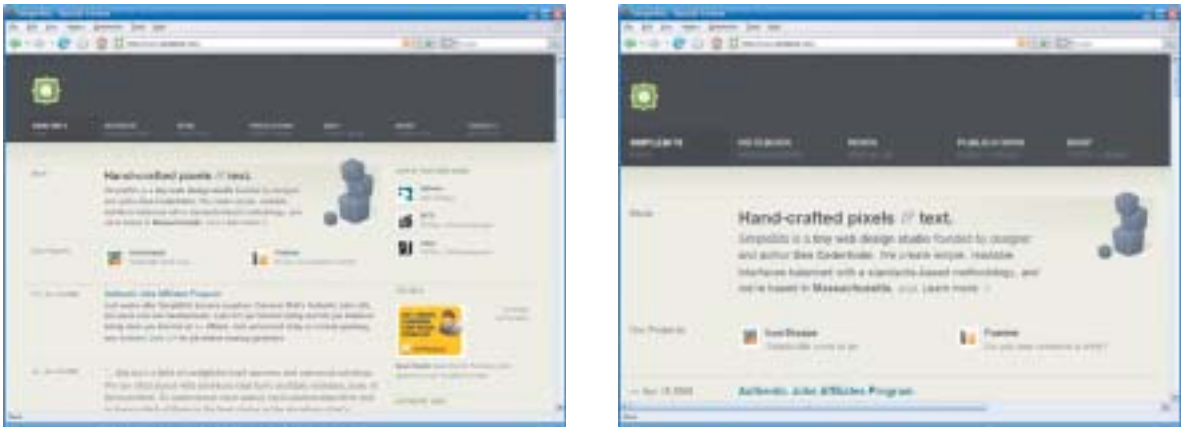

FIGURE 1.6 Dan Cederholm's SimpleBits web site features an elastic design, so as you increase your text size, the text doesn't wrap any differently. Notice that the blurb text at the top of the page keeps the same number of characters on each line when the text is small (left) or large (right).

IMPROVING READABILITY THROUGH STANDARD LINE LENGTHS

With widths set in number of characters per line, you can choose line lengths that optimize readability. As mentioned earlier, line lengths of 75 to 100 characters usually result in increased reading speed (though not necessarily increased comprehension or comfort) over shorter line lengths.

As we saw with liquid layouts, however, not all users prefer the same line lengths. Certain types of disabilities, as well as device limitations and age, may make the line lengths you choose less ideal or downright problematic for

LINE-LENGTH RESEARCH

While more recent research shows that longer line lengths are better for onscreen text than the shorter lengths that have been traditional to print media, it's hard to draw conclusions that are much more concrete than this. I think this is an area where we'll continue to see new standards emerge as we get more extensive research into online reading performance and preference.

The most recent line-length study, "The Effects of Line Length on Reading Online News" by A. Dawn Shaikh, makes an interesting read and provides figures from the conclusions of several older studies. The article was published in the July 2005 issue of the Usability News newsletter (http://psychology.wichita.edu/surl/usabilitynews/72/LineLength.htm). Another good overview of the results of line-length studies through 2002 is "Optimal Line Length: Research Supporting How Line Length Affects Usability" by Dr. Bob Bailey (www.webusability.com/article_line_length_12_2002.htm).

some people. In fact, the most recently published study found that users preferred either the shortest length tested (30 percent preferred 35 characters per line) or the longest length tested (another 30 percent preferred 95 characters per line). Also, many studies have shown that users say they prefer shorter lines even though the testers found they read faster with longer lines.

It's a delicate balancing act: do you trust users to set up the ideal browsing environment for their needs, or, knowing that most users have no idea what line length would be most comfortable or result in the best performance for them, do you optimize it in the way you think will work best for the majority? And if you do choose to use an elastic layout to optimize line length, do you optimize it to the users' preference or to their performance, since it appears they often don't match? There's no right answer, which is why there is no one type of layout that is right for all sites. We'll cover how to choose which one may be best for your site later in the chapter.

INCREASING ACCESSIBILITY

While the readability improvements that we've discussed can affect everyone, they can have an even greater impact on people with disabilities, such as visual impairments that don't warrant the use of a screen reader but do require larger than normal text. People with motor impairments might also use larger text in order to have larger than normal link text, to make it easier for them to "target." And some people, such as those with tunnel vision, might actually prefer smaller than normal text so that more content can fit within their range of vision. Designing layouts that stay proportional at all these different sizes can really help a wide variety of people.

Hybrid Layouts

You don't have to stick with one of the "big three" types of layouts. You can create countless hybrid versions simply by mixing units of measurement or limiting the flexibility range of a liquid or elastic layout.

MIXING UNITS OF MEASUREMENT

Most web layouts are built using the idea of columns, whether or not the columns are explicitly visible in the design. Each column can have its own unit of measurement and be thought of individually as fixed-width, liquid, or elastic. Mix them together, and you've got a *hybrid layout*. For example, a common type of hybrid layout has a fixed-width sidebar with a liquid main content area (**Figure 1.7**).

FIGURE 1.7 The same hybrid layout (fixed sidebar, liquid main column) in two differently sized browser windows. Note that the sidebar is the same width in both windows, while the main content column adapts to fill the remaining viewport space.

RESOLUTION-DEPENDENT LAYOUTS

A resolution-dependent layout is a page that uses JavaScript to switch the CSS and thus the layout of the elements on the page by detecting browser window size. It's kind of like a fixed-width and liquid hybrid, because although each of the possible layouts might be a fixed width, which fixed width the browser chooses is based on window size. It can keep users from seeing horizontal scrollbars and it makes the best use of the available horizontal space, so the design never looks awkwardly stretched out or squished together.

The problem with these types of pages is that you have to create at least two and possibly several different layouts. It can be quite difficult and time-consuming to create a design that can adapt that radically. But it's certainly a cool effect that can enhance the usability as well as the design aesthetic, if you're up for the challenge.

You can see a number of examples of resolution-dependent layouts linked from http://clagnut.com/blog/1663. Teaching the JavaScript necessary to create such layouts is beyond the scope of this book, but two popular scripts, as well as more information on how the technique works, are freely available at http://themaninblue.com/writing/perspective/2006/01/19 and http://alistapart.com/articles/switchymclayout.

Some day you may be able to use CSS 3's media queries feature to do a similar thing without having to ever get JavaScript involved. See www.w3.org/TR/css3-mediaqueries for more information.

Usually, a design that has even one liquid column is called liquid, even if parts are fixed-width; likewise for hybrid layouts that are partially elastic. We'll stick to calling them hybrid layouts throughout this book and reserve the terms *liquid* and *elastic* for layouts that are 100 percent liquid or elastic.

Hybrid layouts offer many of the advantages of liquid and elastic layouts without some of the disadvantages. But they're sometimes tricky to build due to the math involved—quick, what's 200 pixels plus 80 percent? It's impossible to know, of course—and if you don't know, how can you set the width on the container that's supposed to hold these two elements? Luckily, there are usually ways to avoid or work around these challenges, as we'll go over in Chapter 6 when we build some hybrid layouts.

LIMITING FLEXIBILITY WITH MINIMUM AND MAXIMUM WIDTHS

Another way to create a functionally hybrid layout is to limit the amount of a liquid or elastic layout's flexibility by setting minimum and maximum widths. The CSS properties `min-width` and `max-width` allow you to set limits on how far a flexible layout will expand or contract. When a flexible design hits its minimum or maximum width, it essentially becomes a different type of layout—whatever type corresponds with the unit of measurement you used for the `min-width` or `max-width` value.

For example, you might give a liquid layout a minimum width in pixels to keep the images inside the layout from overflowing when the viewport is too narrow. The page would act like a liquid layout as the window was narrowed, until it reached the `min-width` value, at which point it would snap to that value and no longer budge, now acting like a fixed-width layout.

Even though using `min-width` and `max-width` essentially creates hybrid layouts, they're usually still called by whatever type they would fall into if the minimum and maximum widths weren't there. It's pretty rare that you'll make a liquid or elastic layout without using either `min-width` or `max-width`, or both, so we'll still call these layouts liquid and elastic in order to keep from calling practically everything hybrid.