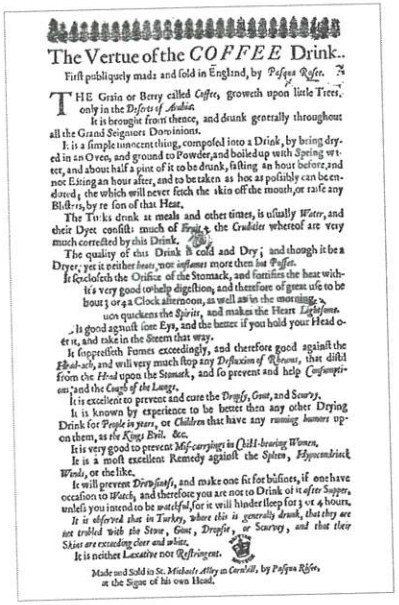
4 RESPONSIVE ADVERTISING

FOR ALMOST AS LONG AS we've printed on paper, we've had advertising. Longer, even: campaign slogans and advertisements have been discovered on the walls of Pompeii. In ancient Egypt, papyrus was often pounded into sales messages and hung prominently. But once we figured out that whole "paper" thing, advertising *really* took off. The earliest printed advertisement is a handbill for wares from tenth-century China (**FIG 4.1**).

I'm sure the advertisements with which you're most familiar are from the printed page—specifically, the display ads that appear in magazines, newspapers, and other periodicals. But those began modestly. Take a look at the first known advertisement for coffee, which appeared in the pages of the *Publick Adviser* in the seventeenth century (**FIG 4.2**). It was a text-only affair, featuring an understated (and rather poetic) testimonial for a roaster's services. A century later, if you were to squint at the front page of the Times of London, you might notice ads for shipping merchants nestled among the columns (**FIG 4.3**). But over time, of course, printed display ads evolved beyond their humble beginnings to become more, well, flamboyant in nature (**FIG 4.4**).

FIG 4.1: From stone walls to handbills, from papyrus to newsprint, advertising's been around for some time. (http://bkaprt.com/rdpp/04-01/; http://bkaprt.com/rdpp/04-02/; http://bkaprt.com/rdpp/04-03/)



The Vertue of the *COFFEE* Drink..

First publiquely made and sold in England, by *Pasqua Rosee*.

THE Grain or Berry called *Coffee*, groweth upon little Trees, only in the *Deserts of Arabia*.

It is brought from thence, and drunk generally throughout all the Grand Seignors Dominions.

It is a simple innocent thing, composed into a Drink, by being dryed in an Oven, and ground to Powder, and boiled up with Spring water, and about half a pint of it to be drunk, fasting an hour before, and not Eating an hour after, and to be taken as hot as possibly can be endured; the which will never fetch the skin off the mouth, or raise any Blisters, by reason of that Heat.

The Turks drink it meals and other times, is usually *Water*, and their Dyet consists much of *Fruit*, the *Crudities* whereof are very much corrected by this Drink.

The quality of this Drink is cold and Dry; and though it be a Dryer, yet it neither *heats*, nor *inflames* more then hot *Posset*.

It scoureth the Orifice of the Stomack, and fortifies the heat within it, is very good to help digestion, and therefore of great use to be taken an hour, or half an hour before meat.

It quickens the *Spirits*, and makes the Heart *Lightsome*.

It is good against sore Eyes, and the better if you hold your Head over it, and take in the Steem that way.

It suppresseth Fumes exceedingly, and therefore good against the *Head-ach*, and will very much stop any *Defluxion of Rheums*, that distil from the *Head* upon the Stomach, and so prevent and help *Consumptions*, and the *Cough of the Lungs*.

It is excellent to prevent and cure the *Dropsie*, *Gout*, and *Scurvy*.

It is known by experience to be better then any other Drying Drink for *People in years*, or *Children* that have any *running humours* upon them, as the *Kings Evil*. &c.

It is very good to prevent *Miscarryings in Child-bearing Women*.

It is a most excellent Remedy against the *Spleen*, *Hypochondriack Winds*, or the like.

It will prevent *Dropsies*, and make one fit for business, if one have occasion to *Watch*, and therefore you are not to Drink of it *after Supper*, unless you intend to be *watchful*, for it will hinder sleep for 3 or 4 hours.

It is observed that in Turkey, where this is generally drunk, that they are not troubled with the Stone, Gout, Dropsie, or Scurvy, and that their Skins are exceeding clear and white.

It is neither Laxative nor Restringent.

Made and Sold in St. *Michaels Alley* in *Cornhill*, by *Pasqua Rosee*, at the Signe of his own Head.

FIG 4.2: In its earliest days, advertising often preferred prose to pictures, as seen in London's *Publick Adviser* (http://bkaprt.com/rdpp/04-04/; http://bkaprt.com/rdpp/04-05/).

The Times.

THURSDAY, DECEMBER 4, 1788

FIG 4.3: Other than some visual ornaments and slightly more adventurous typesetting, display advertisements still feel fairly understated a century later (http://bkaprt.com/rdpp/04-06/).

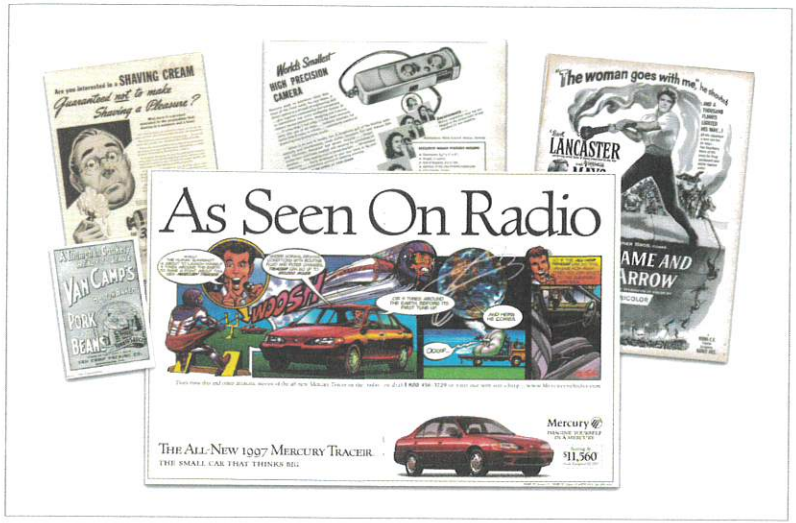
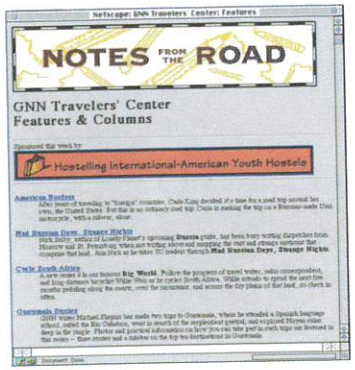


FIG 4.4: Of course, ads got a little more ... vivid over time. (http://bkaprt.com/rdpp/04-07/; http://bkaprt.com/rdpp/04-08/; http://bkaprt.com/rdpp/04-09/; http://bkaprt.com/rdpp/04-10/; http://bkaprt.com/rdpp/04-11/)

When the web came along, it's only natural that we borrowed the advertising practices that seemed to work for print-based publishers. The trajectory for digital ads was, in many ways, similar to print. From simple, modest banners to the complex interactive ads of today, the design of digital advertising has evolved into its own distinct practice (**FIG 4.5-4.6**). For good or ill, much of our medium is supported by advertising—and this presents a unique challenge for responsive layouts. Because when it comes to responsive design, digital advertising is one of the elephants in the room: after all, most ads on the web are fixed-width.

I know, I'm as shocked as you are. But it's true! Take, for example, the Interactive Advertising Bureau (IAB), the consortium responsible for defining most standards for online advertising. If you read their guidelines for desktop (http://bkaprt.com/rdpp/04-12/) or mobile (http://bkaprt.com/rdpp/04-13/), you'll see that each entry in the list—the 300×250 "medium rect-



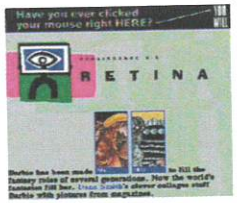


FIG 4.5: Love them or loathe them, these early banner ads from O'Reilly and HotWired helped kickstart the digital advertising industry—and, eventually, the rise of ad blockers.

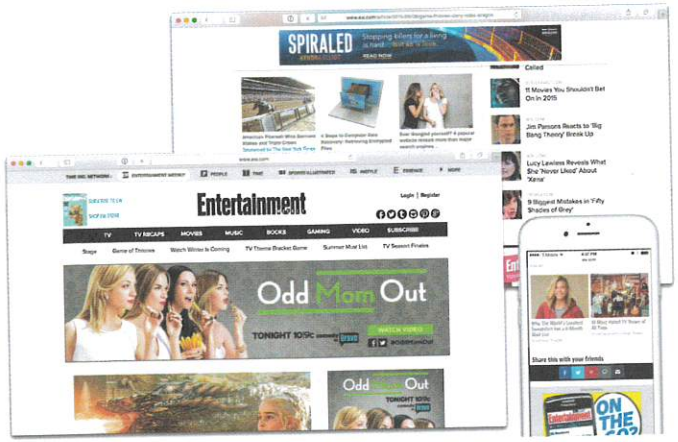


FIG 4.6: Banners, videos, and rollovers—oh my!

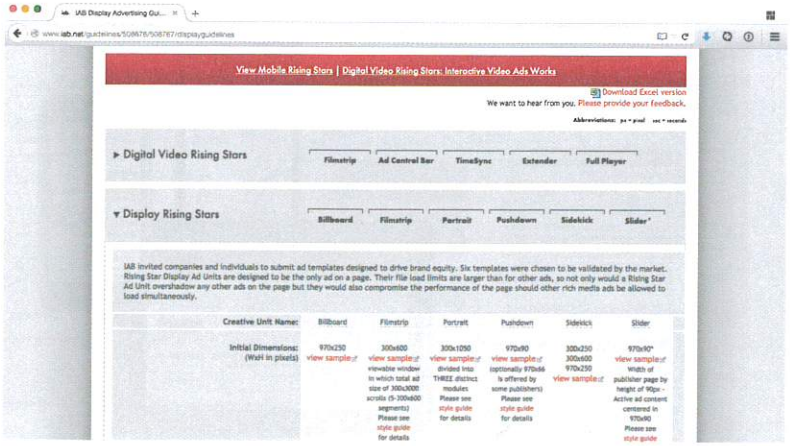


FIG 4.7: You can have ads in any shape you want. (As long as you like pixels.)

angle" ad, the 160×600 "skyscraper" ad, and so on—has a specific width and height, defined in perfectly inflexible pixels (**FIG 4.7**).

I mention this not because I hate pixels. (Much.) But this presents a challenge to responsive designers. Though our layouts have become more flexible, responsive, and device-agnostic, most standard advertisement sizes are still defined in specific, fixed dimensions. So, how are we supposed to incorporate them into decidedly fluid designs?

I'll jump to the punchline: there's no perfect answer quite yet. Responsive advertising is still very much a work in progress, but there are a number of emerging patterns we can use. Let's take a look.

(Hang on: aren't punchlines supposed to be funny?)

CONDITIONAL LOADING

Smashing Magazine, an online publication for web designers and developers, launched a striking new responsive site in 2012 (**FIG 4.8**). Featuring an airy palette alongside considered, elegant typography, the Elliott Jay Stocks-designed site is a joy to read

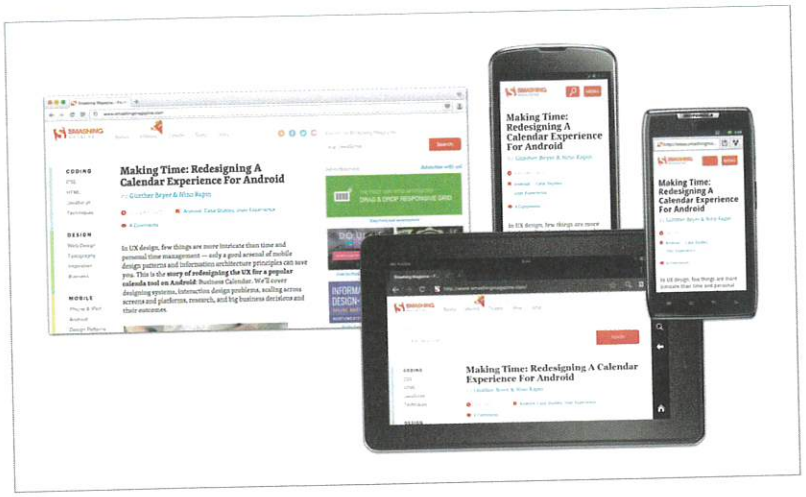


FIG 4.8: *Smashing Magazine,* sporting a stately responsive design.

on any device. If your browser is on a reasonably wide display, you'll see a fair number of advertisements within the fixed-width sidebar on the right side of the screen. But if you open the site on a smaller display—like a phone or small tablet—the ads aren't visible. Below a certain point, they're completely hidden.

Here's a quick look at the CSS that creates the effect:

```
.sb {
  display: none;
}
@media screen and (min-width: 63.75em) {
  .sb {
    display: block;
  }
}
```

The block that contains the advertisements—a `div` with a class of `.sb`—is set to `display: none` by default. But above a viewport width of `63.75em`, or approximately `1020px`, the sidebar's

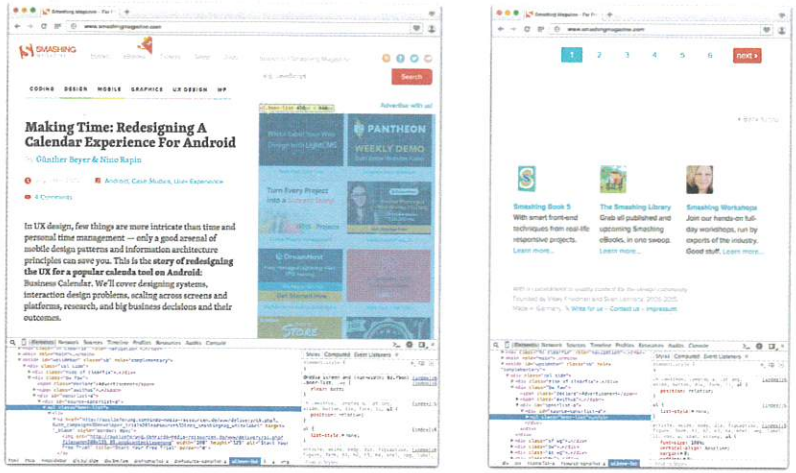


FIG 4.9: *Smashing*'s ads are hidden from view, but still loaded.

`display` property is set to `block`, allowing it to reappear on the right edge of the design, ads triumphantly in view.

Seems like a reasonable approach, no? After all, below a certain width, the ads would be incredibly difficult to integrate into the responsive layout. But if you poke around in your browser's inspector, you'll see the ads are still loaded: the code to display and download the ads still runs (FIG 4.9). There are simply a few lines of CSS to hide them from view.

There are a number of challenges with hiding content when it doesn't fit. From an advertising standpoint, this might mean widescreen readers are subsidizing the experience for those on smaller displays. (Assuming, of course, that hidden ads aren't counted as "viewed" by their advertisers.) And as we discussed before, extra (but hidden) code can introduce needless overhead into our designs. If a certain class of users—mobile, tablet, desktop, or otherwise—won't benefit from a certain piece of content, simply hiding that information with CSS adds extra weight to the page that won't benefit the reader.

Beyond the ads themselves, there are other potentially important pieces of content within that sidebar—a number of

FIG 4.10: It's not just ads: there are other pieces of great content hidden from small screens.

promotional blurbs for various books, a newsletter subscription form, and so on—that are hidden from smaller screens (**FIG 4.10**). These are, I hasten to add, not criticisms of *Smashing Magazine:* hiding content that "doesn't fit" is a common technique on many responsive sites. But whether we're designing text, video, imagery, or advertising, we should be looking for opportunities to *simplify* our designs, rather than *suppressing* information. A better approach would be to load only what we need at any given viewport, rather than hiding the excess with CSS. More specifically, we can start by identifying the ads best suited for each breakpoint, and then load them only if the design can accommodate them.

Back in Chapter 2, we took a brief look at responsive navigation that used conditional loading to *(ahem)* load more complex menus conditionally: say, when a viewport was above a certain width. Currently, the markup for the sidebar element—our `.sb` div—is included directly in the page, and hidden with CSS:

```
<div class="sb">
  <!-- Code for sidebar -->
  …
</div>
```

In theory, we could use the Ajax-Include pattern (http://bkaprt.com/rdpp/02-08/) to load the markup for the sidebar conditionally, by removing the content and moving it to an external file—say, `sidebar-contents.html`:

```
<div class="sb"
  data-append="/include/sidebar-contents.html"
  data-media="(min-width: 63.75em)"></div>
```

This is only a sketch, but it shows how the Ajax-Include pattern might work. The `data-append` attribute points to the URL of our snippet, which contains the content to be appended to the `div`; `data-media`, however, says the snippet should be loaded *if* our viewport is `63.75em` or wider. Otherwise, if the viewport is smaller than that threshold, the `div` remains empty.

FIG 4.11: Ads may appear underneath the lead stories on smaller screens.

RETHINKING THE HIERARCHY

While redesigning their site in 2011, the *Boston Globe* came up with a pattern to address their responsive advertising problem: namely, that the placement of an advertisement would be determined by the width of the page. When the site was a single column, ads could be inserted into sensible points in the content flow. A large ad might appear immediately after the lead stories on the homepage—but only on narrower screens (**FIG 4.11**). As the layout widened to two columns, the ad would move from its initial position and stick to the top of that new column. Similarly, when a third column appeared at the widest breakpoint, the ad would shift again (**FIG 4.12–13**).

I was part of the team working on the redesign, and we all felt the *Globe*'s suggested pattern was a downright novel approach to making their ads responsive. However, it did require a slight departure from their traditional method of inserting ads into pages. Historically, producers would insert some JavaScript in a page's HTML, like so:

```
<script>insertAd( 'MAIN_AD' );</script>
```

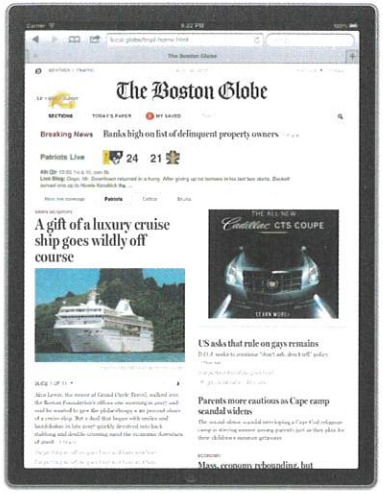



FIG 4.12: At a wider breakpoint, the ad's promoted to the second column.

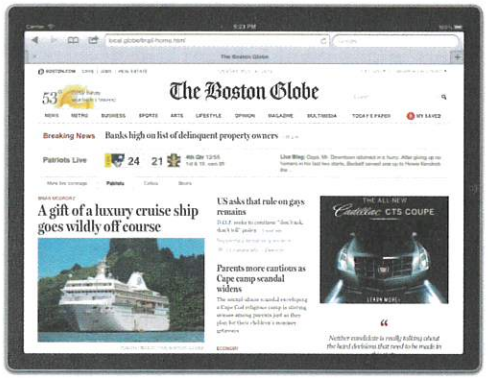


FIG 4.13: As the design gets wider still, the ad moves to the third column.

It looks fairly straightforward, because it's designed to be: the `insertAd()` function is tasked with inserting an ad of some type (specifically, `MAIN_AD`) at that specific point in the layout. But that simplicity's short-lived. Once that code is run by the browser, it often turns into complex-looking JavaScript—specifically, a series of `document.write()` statements:

```
<script>
document.write( »
  '<script src="ad-load.js"></script>' );
document.write( »
  '<style>.ad { border: 1px solid; … }</style>' );
…
</script>
```

It's a bit more complicated, but the spirit's the same: those `document.write()` statements are responsible for inserting the `MAIN_AD` advertisement into the design, along with any JavaScript and CSS files it requires. This inline approach is very common on the web, and it definitely got points for reliability, because you knew exactly where each ad would be placed. But unfortunately, it doesn't work for a more responsive solution like the one outlined by the *Globe,* in which an ad could appear in multiple potential locations.

Complicating matters is `document.write()` itself. First of all, it's terrible for performance: while the browser downloads all the external images, styles, and assets required to render the ad, any content on the page after those `document.write()` statements is prevented from loading (**FIG 4.14**). The effect on the user's experience can be terrible, especially on lower-powered devices or slower networks. What's more, once content's been written into the page with `document.write()`, it can't be moved around with JavaScript. If we had used this method, our ad would have been locked into place, making `document.write()`-generated content incompatible with our responsive advertising pattern.

To make our ads responsive-friendly, our first step was to remove all inline JavaScript. Instead, we looked at all the areas where ads could *potentially* appear—underneath the lead stories, at the top of the block for the second column, and then at the top of the third column—and inserted an empty `div` into each location:

```
<div data-adname="MAIN_AD" class="ad-slot-a"></div>
…
<div data-adname="MAIN_AD" class="ad-slot-b"></div>
```

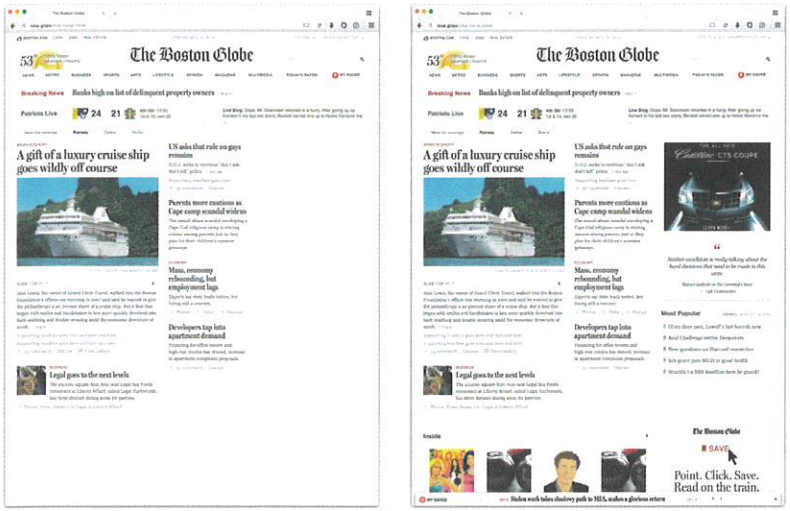



FIG 4.14: `document.write()`: great for inserting content at a precise point in the design; not so great at performance.

```
…
<div data-adname="MAIN_AD" class="ad-slot-c"></div>
```

While each `div` is completely empty, it does have two pieces of descriptive information attached to it. The first is `data-adname`, an HTML5 `data-` attribute, which contains the name of the ad it will eventually contain. (I am, like, a genius at naming things.) The other snippet of metadata is a humble `class` attribute, which allows us to distinguish each ad container from its siblings.

Pretty modest markup, but this was the foundation for our responsive advertising pattern. In the examples below, we'll be using a `class` attribute as a kind of "hook" to apply simple styles—namely, by selectively hiding or showing each block at different breakpoints. With that `display` toggle in place, we can write some light JavaScript to not just insert the ads, but to shuttle them from one position to the next:

1. The script begins by looping through all `divs` that share a `data-adname` value, and looking for the first one that's set to `display: block`.
2. Once it's found, our JavaScript inserts the ad into that slot.
3. Whenever the browser window resizes or the device's orientation changes, the JavaScript starts the process over again: looking for the visible block, and moving the ad into that container.

By only showing one container at each layout breakpoint, our JavaScript can place the ad into the appropriate container, making our ads breakpoint-sensitive. We begin by showing only the ad block that appears immediately after the lead stories—that is, the container with a `class` of `ad-slot-a`:

```
.ad-slot-a {
  display: block;
}
.ad-slot-b,
.ad-slot-c {
  display: none;
}
```

We've hidden `ad-slot-b` and `ad-slot-c` from view, so our JavaScript loops through all of the `MAIN_AD` containers, and sees that only `ad-slot-a` is visible. And since the `div` is set to `display: block`, our script inserts the ad into that container, like so:

```
<div data-adname="MAIN_AD" class="ad-slot-a">
  <a class="ad" href="http://example.com/">
    <img src="http://example.com/ad-main.gif"
      alt="" />
  </a>
</div>
…
```



```
<div data-adname="MAIN_AD" class="ad-slot-b"></div>
…
<div data-adname="MAIN_AD" class="ad-slot-c"></div>
```

Once our viewport gets a little wider—around `30em`—the second column becomes available. At that point, we'll update our CSS slightly to *only* show `ad-slot-b`, the second of our three ad containers:

```
@media (min-width: 30em) {
  .ad-slot-b {
    display: block;
  }
  .ad-slot-a,
  .ad-slot-c {
    display: none;
  }
}
```

With `ad-slot-a` hidden, our JavaScript runs again, and notices that `ad-slot-b` is visible. As a result, our script inserts the ad into that container:

```
<div data-adname="MAIN_AD" class="ad-slot-a"></div>
…
<div data-adname="MAIN_AD" class="ad-slot-b">
  <a class="ad" href="http://example.com/">
    <img src="http://example.com/ad-main.gif"
      alt="" />
  </a>
</div>
…
<div data-adname="MAIN_AD" class="ad-slot-c"></div>
```

Then, at the widest breakpoint, we could hide all of our containers, *except* the one atop the rightmost column—`ad-slot-c`:

```
@media (min-width: 50em) {
  .ad-slot-c {
    display: block;
  }
  .ad-slot-a,
  .ad-slot-b {
    display: none;
  }
}
```

With these two simple rules in place, our JavaScript will—you guessed it—move the ad into our third and final container:

```
<div data-adname="MAIN_AD" class="ad-slot-a"></div>
…
<div data-adname="MAIN_AD" class="ad-slot-b"></div>
…
<div data-adname="MAIN_AD" class="ad-slot-c">
  <a class="ad" href="http://example.com/">
    <img src="http://example.com/ad-main.gif"
      alt="" />
  </a>
</div>
```

With our modest CSS toggle and some lightweight JavaScript, our ad is finally getting a properly responsive treatment (**FIG 4.15**). It's never resized or clipped, but it is repositioned to maintain its visibility and make the best use of the space available. This approach isn't limited to the *Boston Globe*'s responsive design. In fact, this pattern evolved into Filament Group's AppendAround library (http://bkaprt.com/rdpp/04-15/), which allows responsively-minded designers to shuttle any content—not just advertisements—from one container to another.

Repositioning advertisements within a responsive design is quickly becoming a standard for many publishers. Several of Vox Media's responsive sites, including Vox.com, have adopted this pattern (**FIG 4.16**). Their approach, however, is slightly different. According to Jesse Young, a member of Vox Media's

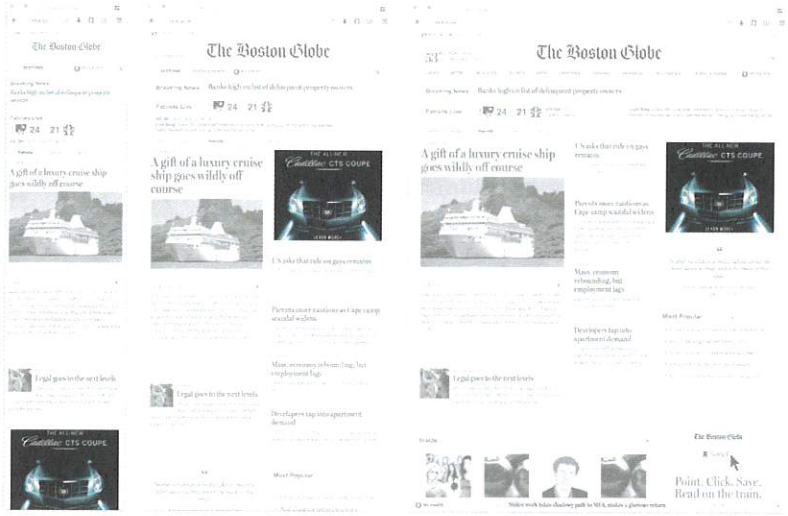


FIG 4.15: Lightweight JavaScript and CSS, combined to shuttle an ad around the page.

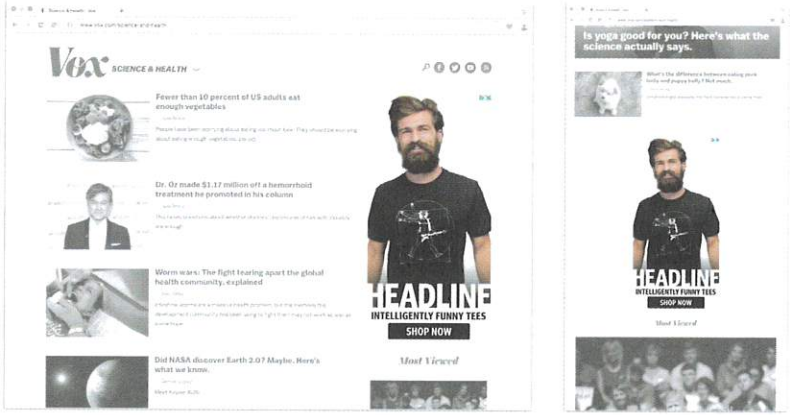


FIG 4.16: Vox Media rotates content responsively around the ad, not the other way around.

product team, they've elected to move not the ads, but everything else (http://bkaprt.com/rdpp/04-16/):

> *Ad placement is slightly trickier. We may want the banner to appear only on the right side of the screen and nowhere else. To ensure this, we use JavaScript for repositioning. However, it's worth noting that we don't actually reposition the ad but the content around it, instead. Because once an ad has rendered, performing DOM manipulation directly on it creates unwanted behaviors like creating tracking inaccuracies or causing the ad to disappear.*

Here's the thing, though: when it comes to responsive advertising, layout is the easy part. In many ways, we've got bigger challenges ahead.

A NEED FOR NEW MODELS

Some time ago, designer Mark Boulton took a step back from problems of layout, and described a number of the deeper challenges with making our advertising responsive (http://bkaprt.com/rdpp/04-17/):

> *Here's the problem as I see it:*
> - *A large number of sites rely on advertising for revenue. Many of those sites will benefit from a responsive web design approach.*
> - *Web advertising is a whole other industry.*
> - *Ad units are fixed, standardised sizes.*
> - *They are commissioned, sold and created on the basis of their size and position on the page*
> - *Many ads are rich (including takeovers, video, pop-overs, flyouts and interactions)*

We've already discussed some of the layout problems, including that advertisements are fixed and inflexible, and aren't usable across various device classes and screen sizes. But Boulton gets to the root of some deeper, business-related issues—namely, the advertising industry operates independently from the rest of the web, and still considers the sale of digital ads in print-centric, position-specific terms.

In a pair of essays on the topic, designer and art director Roger Black approached the problem from another standpoint—namely, that the *business* of online advertising is far from ready for the web's multi-device nature (http://bkaprt.com/rdpp/04-18/, http://bkaprt.com/rdpp/04-19/):

> *Web, tablet and mobiles are sold and served separately, and there are not analytics services that can yet follow a multi-platform campaign. Right now the only way to get responsive advertising is a custom sell, and custom creative...[T]here is no single way to buy and insert adaptive ads across the platforms. The Interactive Advertising Bureau, which has worked over the years to promote standard sizes for ads for the desktop web, doesn't even list mobile ad sizes with its web ad units.*

Black was writing about the problem in 2011, but the underlying issues haven't changed much. Many advertising networks still think of "mobile," "tablet," and "desktop" as distinct products to be managed and sold, making it difficult for companies to coordinate ad campaigns across multiple device types. This problem will only get more complicated over time, of course—soon, "mobile," "tablet," and "desktop" won't be the only categories we're designing for. (And they shouldn't be.) According to research published by Google, this siloed approach desperately needs to catch up with our multiscreen reality (http://bkaprt.com/rdpp/04-20/). People rarely begin and end a particular workflow on one device; instead, we might begin shopping on our phones, before completing the checkout on our tablets or laptops.

Like our responsive layouts, our ads need to become not just more fluid in shape, but also in delivery. And while the advertising industry has yet to modernize their layout standards or business practices, many organizations have opted to try and fix responsive advertising internally, by designing and developing custom-built, more flexible ad formats in-house (**FIG 4.17**). According to Vox Media's Trei Brundrett, this approach didn't

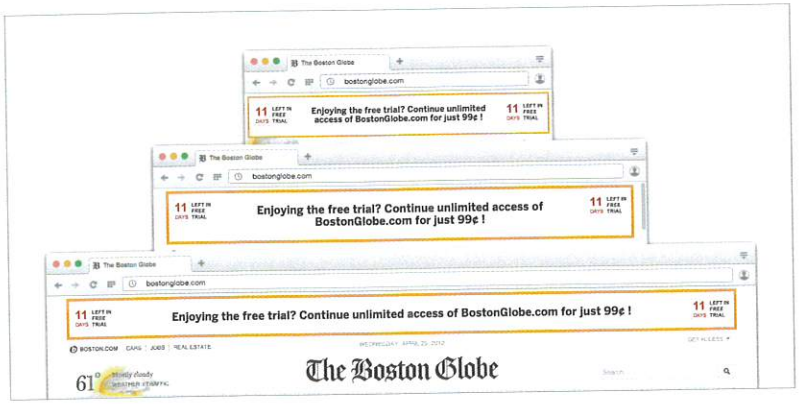


FIG 4.17: Many publishers, like the *Boston Globe,* have been bucking industry standards and started designing flexible advertisements in-house.

just yield more responsive-friendly ads for their websites—it created ads that were less intrusive *and* more profitable (http://bkaprt.com/rdpp/04-21/):

> *Our guiding principle is that advertising is part of the total user experience...It turns out that a great user experience with your advertising integrated with what you're building, your advertising performs better. It performs better for everybody.*

There's something appealing about that formula. Rather than seeing digital advertising as somehow incompatible with an elegant, reader-friendly design, publishers are suggesting that flexible, responsive-friendly ads result in happier advertisers *and* readers.

The digital advertising team at the *Guardian* describes their cross-device advertising experiments in similar terms, saying their new responsive ad units are "better for advertisers, better for the *Guardian* and better for our readers" (http://bkaprt.com/rdpp/04-23/). Since their site sees a massive amount of diversity in screen sizes and device classes each month—"6000 different types [and] counting"—the *Guardian* created a few flexible versions of standard ad sizes (**FIG 4.18**). To do so, they broke each

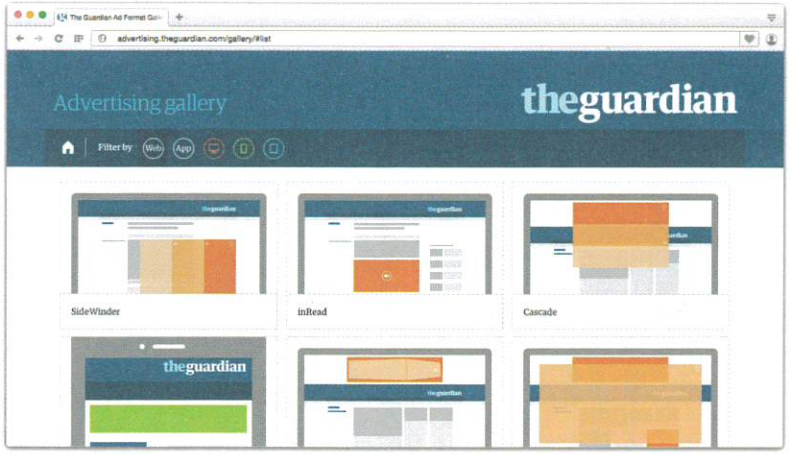


FIG 4.18: Absent an industry standard, the *Guardian* created a number of responsive ad units in-house (http://bkaprt.com/rdpp/04-22/).

advertisement into its component parts, and treated them like a small-scale responsive design:

> *To build this unit, we abstracted the various elements of an advert: the background, the subject image, the branding and the call to action. These are then populated individually into the HTML5 ad unit to allow the unit to respond best to the space available.*

By treating their advertisements as small-scale responsive layouts instead of fixed, inflexible blocks, publishers like the *Guardian* are able to reposition these elements within a flexible, responsive canvas.

Of course, not every site has the resources to design their own responsive-friendly ad formats and sell them to prospective advertisers. Thankfully, Monotype has built demos of various responsive ad formats with lightweight, standards-based technologies—each one designed for flexibility from the outset (**FIG 4.19**). And Google—that scrappy little search engine—has released a responsive unit for its AdSense advertising service,

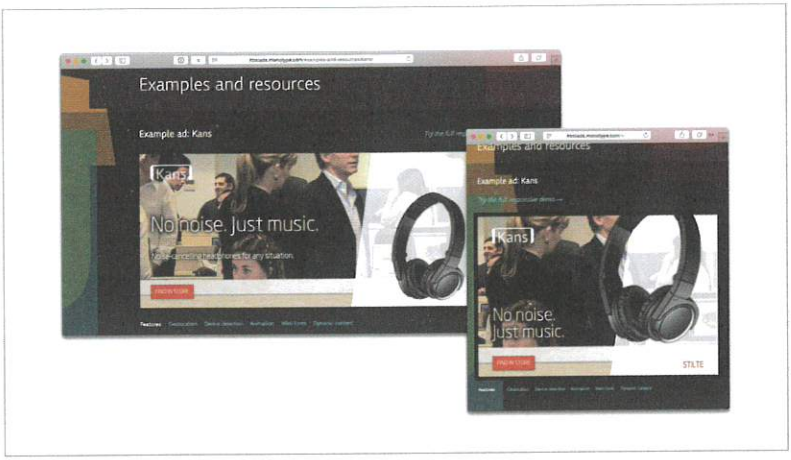


FIG 4.19: Monotype's various responsive ad formats are wonderful proofs-of-concept (http://bkaprt.com/rdpp/04-24/).

which should help make responsive advertising more broadly accessible (http://bkaprt.com/rdpp/04-25/).

Responsive advertising is still very much in its early days. As of this writing, the advertising industry hasn't made much progress on truly cross-device advertising, focusing instead on contractual language to improve ad visibility, and investing heavily on designing distinct ad formats for "mobile" and "desktop" (http://bkaprt.com/rdpp/04-26/). And publishers are painfully aware of this gap, as Peter Bale, CNN International Digital's vice president, noted recently (http://bkaprt.com/rdpp/04-27/):

> *The ad industry has not fully come down the pipe yet in terms of responsively designed ads that will particularly work to the same level of monetisation on any device—there is a lag there. We have to move ahead of that and that's very difficult.*

Until that "lag" disappears, it seems it's up to us to address user-facing issues like performance or layout—and to come up with our own ways of making advertising lightweight, flexible, and responsive.