SMASHING MAGAZINE

Essentials Of Mobile Design

Imprint

Version 1: September 2012

ISBN: 978-3-943075-42-7

Cover Design: Ricardo Gimenes

PR & Press: Stephan Poppe

eBook Strategy: Talita Telma Stöckle

Technical Editing: Talita Telma Stöckle, Andrew Rogerson

Idea & Concept: Smashing Media GmbH

ABOUT SMASHING MAGAZINE

Smashing Magazine is an online magazine dedicated to Web designers and developers worldwide. Its rigorous quality control and thorough editorial work has gathered a devoted community exceeding half a million subscribers, followers and fans. Each and every published article is carefully prepared, edited, reviewed and curated according to the high quality standards set in Smashing Magazine's own publishing policy. Smashing Magazine publishes articles on a daily basis with topics ranging from business, visual design, typography, front-end as well as back-end development, all the way to usability and user experience design. The magazine is — and always has been — a professional and independent online publication neither controlled nor influenced by any third parties, delivering content in the best interest of its readers. These guidelines are continually revised and updated to assure that the quality of the published content is never compromised.

ABOUT SMASHING MEDIA GMBH

Smashing Media GmbH is one of the world's leading online publishing companies in the field of Web design. Founded in 2009 by Sven Lennartz and Vitaly Friedman, the company's headquarters is situated in southern Germany, in the sunny city of Freiburg im Breisgau. Smashing Media's lead publication, Smashing Magazine, has gained worldwide attention since its emergence back in 2006, and is supported by the vast, global Smashing community and readership. Smashing Magazine had proven to be a trustworthy online source containing high quality articles on progressive design and coding techniques as well as recent developments in the Web design industry.

About this eBook

Designing for Mobile can be very complex – it requires many skills such as programming, usability, typography, creating applications… all of this without forgetting the necessary ingredient of visual appeal. This eBook "Essentials of Mobile Design" will give you an overview of the basic features you need to know for designing beautiful and useful Mobile interfaces and apps.

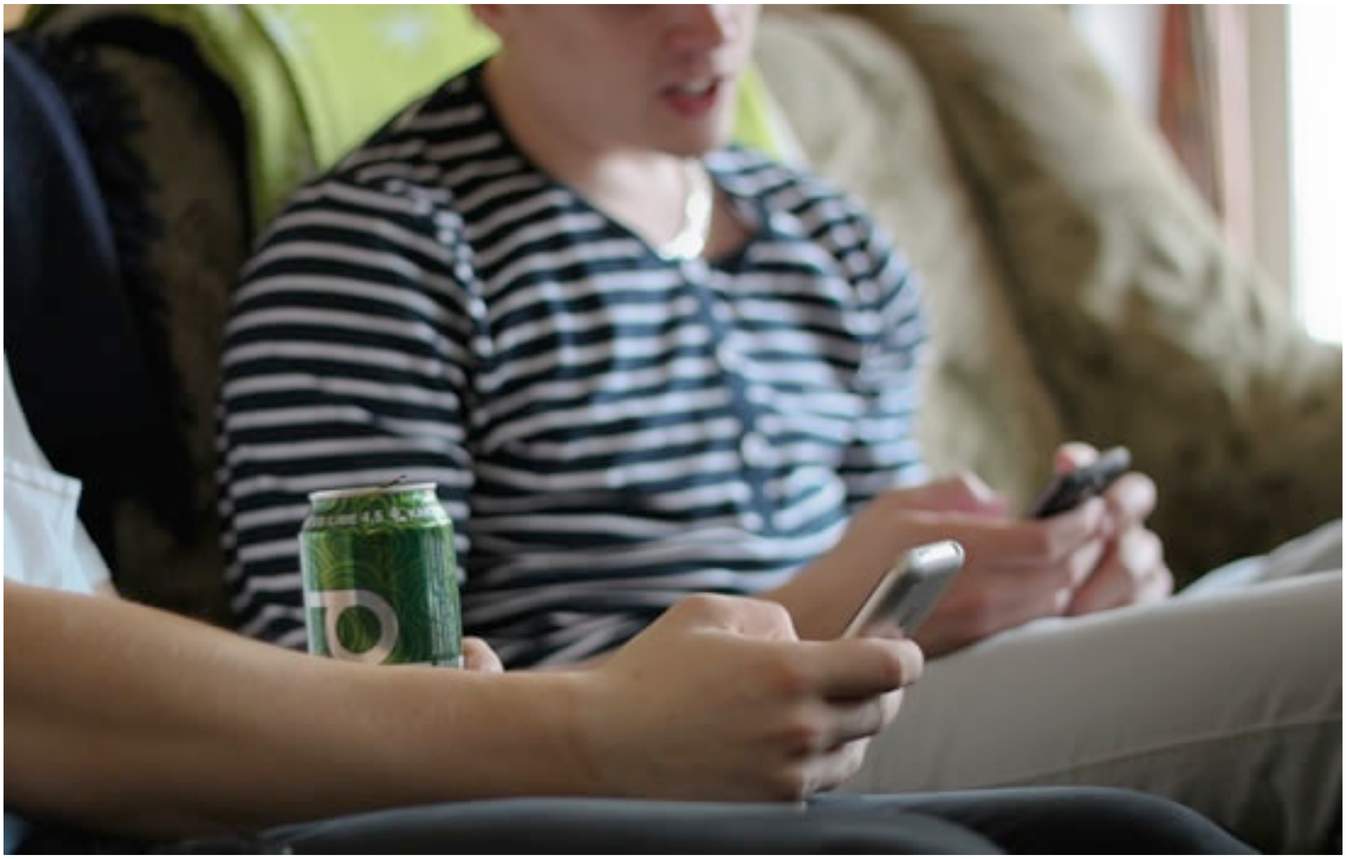
Table of Contents

Not Your Parent's Mobile Phone: UX Design Guidelines For Smartphones

Tim R. Todish

In your pocket right now is the most powerful "remote control" (as Drew Diskin put it) that has ever existed. It is no ordinary remote control. It can harness everything that all of the previous mass media (television, radio, Internet, etc.) can do. People aren't using them just for simple entertainment or for phone calls. They have become the hub of our personal lives.

Smartphones are what younger generations know as just phones. The iPad (aka *the* tablet) is giving your grandma's PC a run for its money. You certainly are holding some amazing futuristic technology in your hands. It will be even better tomorrow, though, so why does it matter to us or to users? Moore's Law tells us, in effect, that these things will continue to become capable of more than anything our minds can think up.



(Image: Denis Dervisevic)

It’s no longer just about the evolving power and capabilities of these devices. It’s about us and how we, too, are changing. The user’s expectation of a great experience is the new standard. It falls to us as UX professionals to apply our skills to make this happen on the vast array of devices out there. It’s not always easy, though. The mobile realm has some unique constraints and offers some interesting opportunities. While covering all of the nuances of mobile UX in one article would be impossible, we’ll cover some fundamentals and concepts that should move you in the right direction with your projects.

Mobile Constraints

The mobile realm has many constraints. Here are several of them, along with thoughts on what to keep in mind as you come upon them.

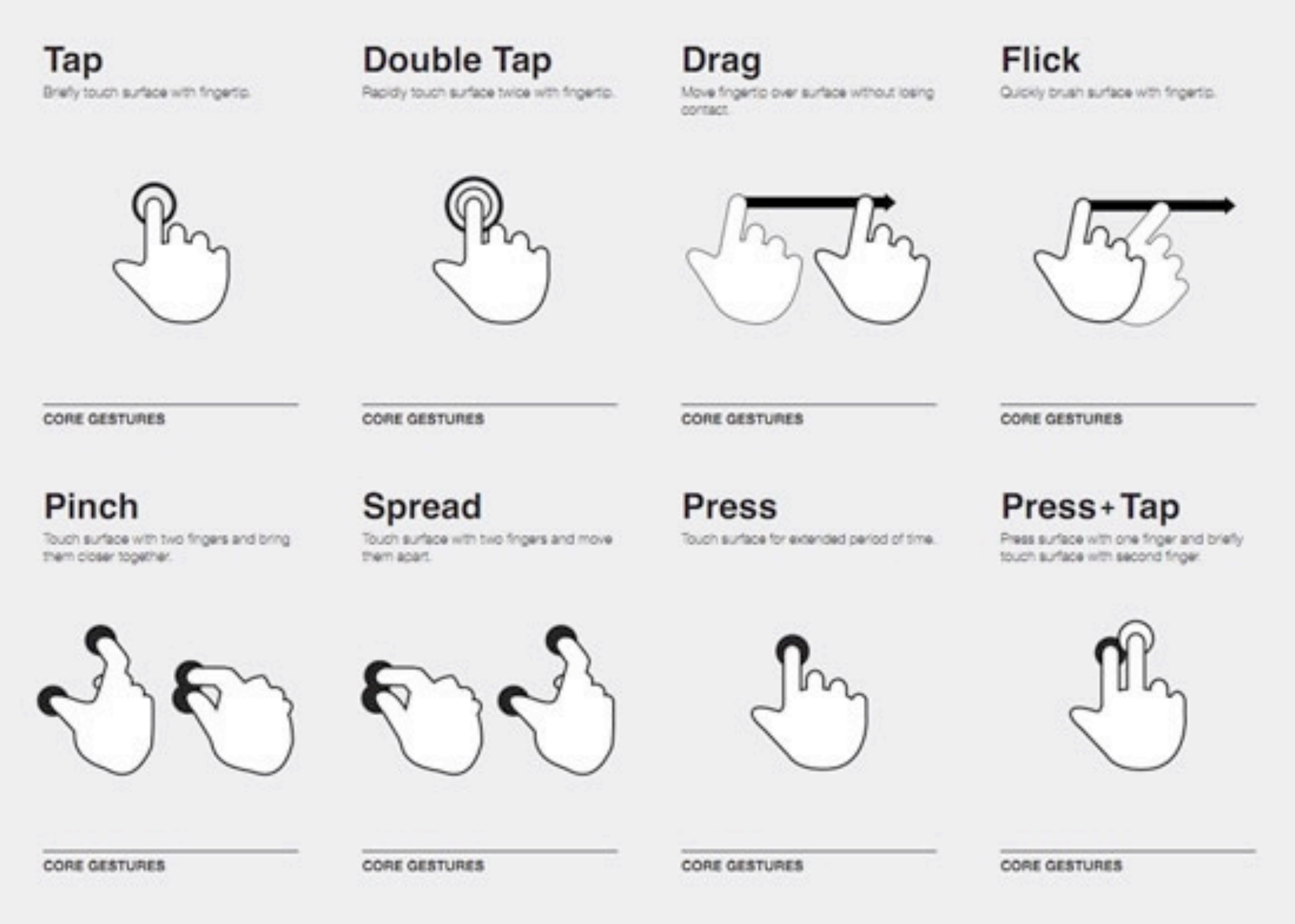
FORM FACTOR

The most obvious constraint going from desktop to mobile is screen size. Mobile screens are smaller. A lot smaller. You need to seriously consider this when designing and developing your application. Antony Ribot makes a good point in his presentation, "Mobile UX: The Intricacies of Designing For Mobile Devices," when he says, "Mobile is not about making things smaller." It's much more than that. We need to consolidate what's on the screen. Boil the application down to the most critical functions and content, and then lay them out strategically in the available screen space. For example, action buttons should go in the lower third of the screen, where they are most easily tappable.

INPUT METHODS

Another obvious constraint is the absence of or difference in certain input mechanisms, and the addition of others. First, there's no mouse. No mouse means no hover states. It also means that there must be some other means of clicking and navigating content. In most cases, this other means is the user's finger. This difference in input method can be quite exciting because it opens the door to new possibilities with various gestures. Many standards are forming around these new gesture capabilities: pinch to zoom, swipe to scroll, etc. Take the time to include support for these gestures in your application. In addition, think of new gestures that you could add to enhance interactivity.

Discovering new gestures can be a powerful experience for users. It adds a sense of excitement, mystery and achievement—"Hey, I just figured out something new!" Take care, though, not to change the function of standard gestures unless you have a very good reason to do so, or else you will cause unnecessary confusion and frustration in users.


Tap
Briefly touch surface with fingertip.
CORE GESTURES
Double Tap
Rapidly touch surface twice with fingertip.
CORE GESTURES
Drag
Move fingertip over surface without losing contact.
CORE GESTURES
Flick
Quickly brush surface with fingertip.
CORE GESTURES
Pinch
Touch surface with two fingers and bring them closer together.
CORE GESTURES
Spread
Touch surface with two fingers and move them apart.
CORE GESTURES
Press
Touch surface for extended period of time.
CORE GESTURES
Press+Tap
Press surface with one finger and briefly touch surface with second finger.
CORE GESTURES

(Touch Gesture Cards (PDF): Luke Wroblewski)

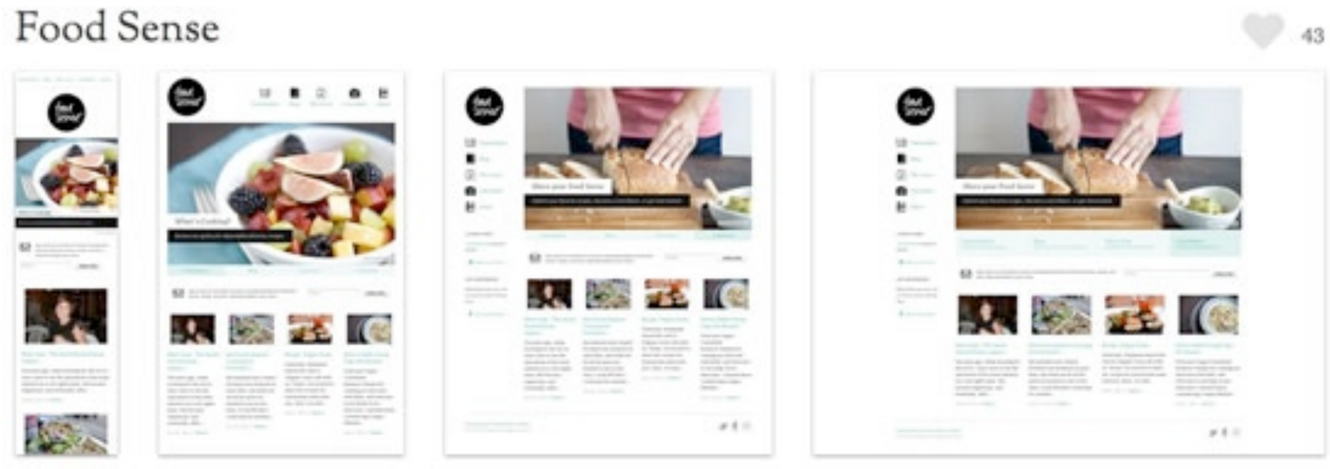
One other caveat: consider the type of application you're developing before getting too fancy with gestures. If it will be highly utilitarian in nature, then keeping things simple and straightforward would be best. If the application is for a specific task, then users will want to complete it as quickly and easily as possible. They don't have the time or desire to discover new interactions.

TECHNICAL CONSTRAINTS

While the capabilities of these devices improve with each new release it, keep in mind their limitations. Things like battery life and processing power are important to consider. Draining the battery or bringing the device to its knees with memory leaks or processor-intensive operations is a surefire way to destroy the user experience. This is why testing on the device early and often is imperative. Simulators cannot be trusted.

DATA TRANSFER AND PRICING

This will not be an issue for users who have unlimited data plans or who work on Wi-Fi networks. Unfortunately, unlimited plans are becoming increasingly rare. So, be sensitive to the amount of data you are transferring to and from your application. Keep the sizes of assets to a minimum, while maintaining quality. Don't transfer data unnecessarily. For example, implement delta updates whenever possible (i.e. update only the data that has changed since the last transfer).



(Images: Mediaqueri.es and Food Sense)

Much has been said recently about Responsive Web Design. This approach does create some challenges with minimizing data transfer. Jason Grigsby has a very good write-up on the specifics. To summarize, CSS media queries — part of the magic sauce of responsive design — do almost nothing to lessen the overhead of data transfer to mobile devices. Resizing or hiding unwanted images still requires the full images to be downloaded to the browser. In addition, resources such as JavaScript libraries might be downloaded to mobile devices without even being enabled for users.

Good General Practices

What follows are some good general principles to keep in mind when designing and developing mobile applications.

MOBILE FIRST

Luke Wroblewski has a great post on the "Mobile First" methodology. In a nutshell, focusing on mobile first puts your mind in the right place. It forces you to focus on and prioritize the most important features and content in your application. It also extends your abilities by offering new tools and services that are not available in a traditional desktop environment. By approaching your project with the mobile-first mentality, you will start off on the right foot.

BEHAVIORS AND ARCHETYPES

Build on the behaviors and archetypes that your users are already accustomed to. This will go a long way to reducing the learning curve of your application. If your application responds predictably to a user's interaction, then the user will immediately become more comfortable.

This applies to more than general behaviors and archetypes. You will want to use design patterns that are specific to your target devices. This means building multiple interfaces for various devices and platforms, which is extra work; but it will pay off in the long run because users will appreciate that your application behaves in the manner they've come to expect from their device. For example, iOS design patterns dictate that tabbed navigation be located at the bottom of the screen, whereas Android devices have it along the top.

As with most good UX principles, if done properly, the user won't even notice, while their increased comfort level will encourage them to continue exploring the application. Which brings us to our next practice.

ENCOURAGE EXPLORATION

The more that users feel comfortable with and enjoy your application, the more likely they will explore it. You may want to lead them down certain paths or provide a few cues or coach marks on how certain things work, but still allow your users to "discover." I'm not suggesting that you make the application complicated or ambiguous; rather, for example, if there are multiple ways to perform an action, one more obvious and traditional and the other a quick and easy gesture, then the user might come to prefer the second option once they discover it. Such solutions improve the overall experience if they prove to be quicker and more efficient than traditional interactions.

PROVIDE IMMEDIATE FEEDBACK

We've all witnessed our less computer-savvy peers clicking violently and repeatedly on a button trying to force it to do whatever they so desperately want to achieve. Touchscreen's only add to this anxiety because they don't provide that tactile response that we've been conditioned to expect from tapping on a keyboard or clicking with a mouse. Providing some indication that the application has registered the user's interaction is critical, whether it's a small bounce at the end of a scrollable region or a subtle color change at the tap of a button. This not only compensates for the lack of tactile response, but assures users that something is happening even if the screen isn't updating immediately due to slow network traffic or some processor-intensive operation.

CONTEXT

(Image: S. Diddy)

Another glaring difference between mobile and desktop applications is context. With a desktop application, you can be relatively certain that it is being used in a particular environment. With mobile, all bets are off. This gives us some exciting opportunities: location-based services, on-the-spot social networking, the opportunities are vast.

It also raises some unique problems. Do your research to determine the context in which the majority of people will be using your application.

If you're targeting on-the-go users, then you'll want to build the application for speed: bold, obvious, stripped-down selectors and a streamlined workflow. If your application is more akin to a breakfast-table browser, then content will probably be more important to the user, but they may have only one hand free to navigate, while the other cradles their morning coffee. These are just two examples; the point is that your mobile application could be used in any number of contexts, and you will need to take the time to figure out how to provide the best experience to the user in their context.

One other thing to consider is the device(s) that you are targeting. Research suggests that a majority of tablet owners use their device mostly at home. Only 21% take their device with them on the go, compared to 59% of smartphone users who consult their device while out and about.

IDEATE IN THE WILD

(Image: Niall Kennedy)

I'm borrowing this one directly from Rachel Hinman because she is spot on. The best way to determine context and to conduct research is to immerse yourself in the environments in which your application will be used.

Hang out where your target audience hangs out. If possible, do the things they do, go where they go. This will serve a couple purposes. First, it could give you ideas for great applications to build. Maybe you'll observe common pain points and come up with a solution to alleviate them. Or, if you already have an idea for an application, you could gain valuable insight into how the application might be (or is being) used in the wild. We'd be surprised quite often by the difference between how we intend for our application to be used and how it is actually being used. This information can help us iterate our ideas and continually improve the application.

Conclusion

The way mobile devices are being used is changing all the time, and users are increasingly expecting exceptional experiences from the applications they use. While the mobile world has many constraints, its many more opportunities make building mobile applications a worthwhile venture. Keep in mind the constraints, and focus on mobile first when beginning your project.

Remember that innovative features and cutting-edge design aren't as valuable to users as we may think. Users are concerned with getting the information they need through a sometimes limited connection, or perhaps getting accustomed to typing on a screen without any tactile feedback. Not everyone has an iPad… yet.

Talk to real people, follow common archetypes, and keep the context of your target users in mind. These guidelines should help you create a great experience in your mobile application.

Why We Shouldn’t Make Separate Mobile Websites

Bruce Lawson

There has been a long-running war going on over the mobile Web: it can be summarized with the following question: “Is there a mobile Web?” That is, is the mobile device so fundamentally different that you should make different websites for it, or is there only one Web that we access using a variety of different devices? Acclaimed usability pundit Jakob Nielsen thinks that you should make separate mobile websites. I disagree.

Jakob Nielsen, the usability expert, recently published his latest mobile usability guidelines. He summarizes:

> *“Good mobile user experience requires a different design than what’s needed to satisfy desktop users. Two designs, two sites, and cross-linking to make it all work.”*

I disagree (mostly) with the idea that people need different content because they’re using different types of devices.

Firstly, because we’ve been here before, in the early years of this century. Around 2002, the huge UK supermarket chain Tesco launched Tesco Access—a website that was designed so that disabled people could browse the Tesco website and buy groceries that would be delivered to their homes.

It was a great success—heavily stripped down, all server-generated (as in, those days screen readers couldn't handle much JavaScript) and it was highly usable. One design goal was "to allow customers to purchase an average of 30 items in just 15 minutes from login to checkout." In fact, from a contemporary report, (cited by Mike Davis), "many non-disabled customers are switching from the main Tesco site to the Tesco Access site, because they find it easier and faster to use!" It also made Tesco a lot of money: "Work undertaken by Tesco.com to make their home grocery service more accessible to blind customers has resulted in revenue in excess of £13m per annum, revenue that simply wasn't available to the company when the website was inaccessible to blind customers."

However, some blind users weren't happy. There were special offers on the "normal" Tesco website that weren't available on the access website. There were advertisements that were similarly unavailable—which was a surprise; whereas most people hate advertisements, here was a community complaining that it wasn't getting them.

The vital point is that you never know better than your users what content they want. When Nielsen writes that mobile websites should "cut features, to eliminate things that are not core to the mobile use case; [and] cut content, to reduce word count and defer secondary information to secondary pages," he forgets this fact.

Tesco learned this:

> *"We have completely redesigned Access so that it is no longer separate from our main website but is now right at the center of it, enabling our Access customers to enjoy the same features and functionality available on the standard grocery website. As part of this work we have had to retire the old Access website."*

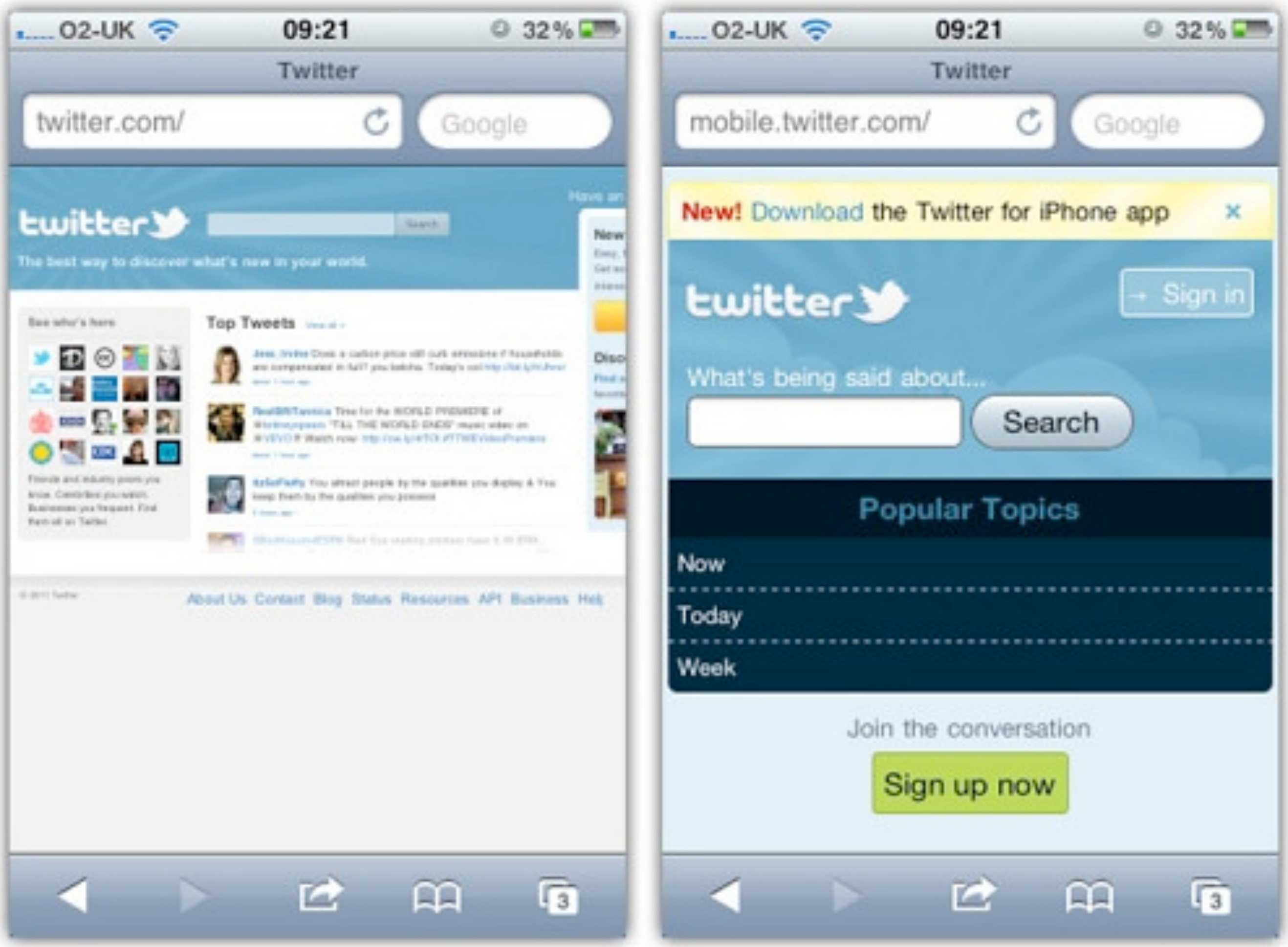
Nielsen writes:

> *"Build a separate mobile-optimized site (or mobile site) if you can afford it … Good mobile user experience requires a different design than what's needed to satisfy desktop users. Two designs, two websites, and cross-linking to make it all work."*

From talking to people in the industry, and from my own experience of leading a dev team, I've found that building a separate mobile website is considered to be a cheaper option in some circumstances—there may be time or budgetary constraints. Sometimes teams don't have another option but creating a separate website due to factors beyond their control.

I believe that this is not ideal, but for many it's a reality. Re-factoring a whole website with responsive design requires auditing content. And changing a production website with all the attendant risks, then testing the whole website to ensure it works on mobile devices (while introducing no regressions in the desktop website)—all this is a huge task. If the website is powered by a CMS, it's often cheaper and easier to leave the "desktop website" alone, and implement a parallel URL structure so that www.example.com/foo is mirrored by m.example.com/foo, and www.example.com/bar is mirrored by m.example.com/bar (with the CMS simply outputting the information into a highly simplified template for the mobile website).

The problem with this approach is Nielsen's suggestion: "If mobile users arrive at your full website's URL, auto-redirect them to your mobile website." The question here is how can you reliably detect mobile browsers in order to redirect them? The fact is: you can't. Most people attempt to do this with browser sniffing—checking the User Agent string that the browser sends to the server with every request. However, these are easily spoofed in browsers, so they can't be relied upon, and they don't tell the truth, anyways. "Browser sniffing" has a justifiably bad reputation, so is often renamed "device detection" these days, but it's the same flawed concept.



On mobile, Twitter.com automatically forwards users to a separate mobile website.

More troublesome is that there are literally hundreds of UA strings that your detection script needs to be aware of in order to send the visitor to the "right" page. The list is ever-growing, so you need to constantly check and update your detection scripts. And of course, you only know about a new User Agent string after it turns up in your analytics—so there will be a period between the first visitor arriving with an unknown UA and your adding it to your detection scripts (in which visitors will be sent to the wrong website).

Despite all this work to set up a second parallel website, you will still find that some visitors are sent to the wrong place, so here I agree with Nielsen:

> *"Offer a clear link from your full site to your mobile site for users who end up at the full site despite the redirect … Offer a clear link from your mobile site to your full site for those (few) users who need special features that are found only on the full site."*

Missing out features and content on mobile devices perpetuates the digital divide. As Josh Clark points out in his rebuttal:

> *"First, a growing number of people are using mobile as the only way they access the Web. A pair of studies late last year from Pew and from On Device Research showed that over 25% of people in the US who browse the Web on smartphones almost never use any other platform. That's north of 11% of adults in the US, or about 25 million people, who only see the Web on small screens. There's a digital-divide issue here. People who can afford only one screen or internet connection are choosing the phone. If you want to reach them at all, you have to reach them on mobile. We can't settle for serving such a huge audience a stripped-down experience or force them to swim through a desktop layout in a small screen."*

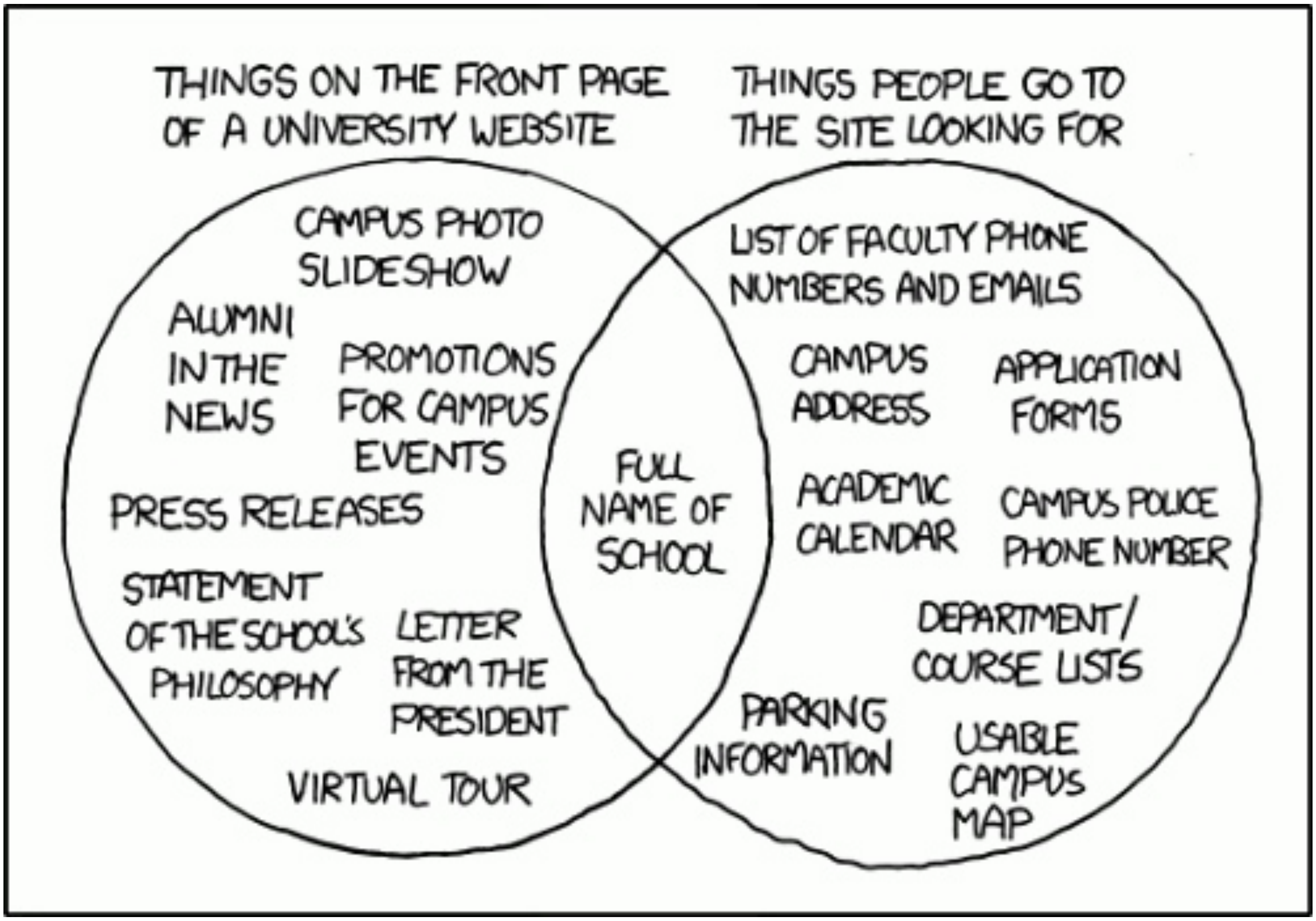
The number of people only using mobile devices to access the Web is even higher in emerging economies. Why exclude them?

Mobile Usability

I also agree with Nielsen when he writes:

> *"When people access sites using mobile devices, their measured usability is much higher for mobile sites than for full sites."*

But from this he draws the wrong conclusion, that we should continue making special mobile websites. I believe that special mobile websites is like sticking plaster over the problem; we generally shouldn't have separate mobile websites, anymore than we should have separate screen reader websites. The reason many "full websites" are unusable on mobile phones is because many full websites are unusable on any device. It's often said that your expenditure rises as your income does, and that the amount of clutter you own expands to fill your house however many times you move to a bigger one. In the same way, website owners have long proved incontinent in keeping desktop websites focussed, simply because they have so much room. This is perfectly illustrated by the xkcd comic:


THINGS ON THE FRONT PAGE OF A UNIVERSITY WEBSITE
THINGS PEOPLE GO TO THE SITE LOOKING FOR
CAMPUS PHOTO SLIDESHOW
ALUMNI IN THE NEWS
PROMOTIONS FOR CAMPUS EVENTS
PRESS RELEASES
STATEMENT OF THE SCHOOL'S PHILOSOPHY
LETTER FROM THE PRESIDENT
VIRTUAL TOUR
FULL NAME OF SCHOOL
LIST OF FACULTY PHONE NUMBERS AND EMAILS
CAMPUS ADDRESS
APPLICATION FORMS
ACADEMIC CALENDAR
CAMPUS POLICE PHONE NUMBER
DEPARTMENT/ COURSE LISTS
PARKING INFORMATION
USABLE CAMPUS MAP

A Venn diagram showing "Things on the front page of a university website" and "Things people go to the site looking for." Only one item is in the intersection: "Full name of school." Image source: xkcd.

As I wrote on the website The Pastry Box on April 13th:

> *"The mobile pundits got it right: sites should be minimal, functional, with everything designed to help the user complete a task, and then go. But that doesn't mean that you need to make a separate mobile site from your normal site. If your normal site isn't minimal, functional, with everything designed to help the user complete a task, it's time to rethink your whole site.*

> *“And once you’ve done that, serve it to everyone, whatever the device.”*

In a previous article, Nielsen wrote in September 2011 that he dropped testing usability with featurephones:

> *“Our first research found that feature phone usability is so miserable when accessing the Web that we recommend that most companies don’t bother supporting feature phones.*
>
> *“Empirically, websites see very little traffic from feature phones, partly because people rarely go on the Web when their experience is so bad, and partly because the higher classes of phones have seen a dramatic uplift in market share since our earlier research.”*

This is a highly westernized view. Many people can't afford smartphones, so they use feature phones running proxy browsers (such as Opera Mini), which move the heavy lifting to servers. This is often the only way that underpowered featurephones can browse the Web. Statistics from Opera's monthly State of the Mobile Web report (disclosure: Opera is my employer) shows that lower-end feature phones still dominate the market in Eastern Europe, Africa and other emerging economies—see the top 20 handsets worldwide for 2011 that accessed Opera Mini. Since February 2011, the number of unique users of Opera Mini has increased 78.17% and data traffic is up 142.79%.

A caveat about those statistics: not every user of Opera Mini is a featurephone user in developing countries. They're widely used on high-end smartphones in the West, too, as we know that they are much faster than built-in browsers, and users really want speed.

Nielsen's dismissal of feature phones reminds me of some attitudes to Web accessibility in the early 2000′s. His assertion that companies shouldn't support feature phones because they see little traffic from feature phones is the classic accessibility chicken and egg situation: we don't need to bother with making our website accessible, as no-one who visits us needs it. This is analogous to the owner of a restaurant that is up a flight of stairs saying he doesn't need to add a wheelchair ramp as no-one with a wheelchair ever comes to his restaurant. It's flawed logic.

Developing Usable Websites For All Devices

The W3C Mobile Web best practices say:

> *"One Web means making, as far as is reasonable, the same information and services available to users irrespective of the device they are using. However, it does not mean that exactly the same information is available in exactly the same representation across all devices. The context of mobile use, device capability variations, bandwidth issues and mobile network capabilities all affect the representation. Furthermore, some services and information are more suitable for and targeted at particular user contexts."*

There will always be edge cases when separate, mobile-specific websites will be a better user experience, but this shouldn't be your default when approaching the mobile Web. For a maintainable, future-friendly development methodology, I recommend that your default approach to mobile be to design one website that can adapt to different devices with viewport, Media Queries and other technologies that are often buzzworded "Responsive Design."

Combining these techniques in a smart way with progressive enhancement allows your content to be viewed on any device (and with richer experiences available on more sophisticated devices), with the possibility of accessing device APIs such as geolocation, or the shiny new getUserMedia for camera access.

Although many other resources are available, I've written "Mobile-friendly: The mobile web optimization guide" which you'll hopefully find a useful starting point.

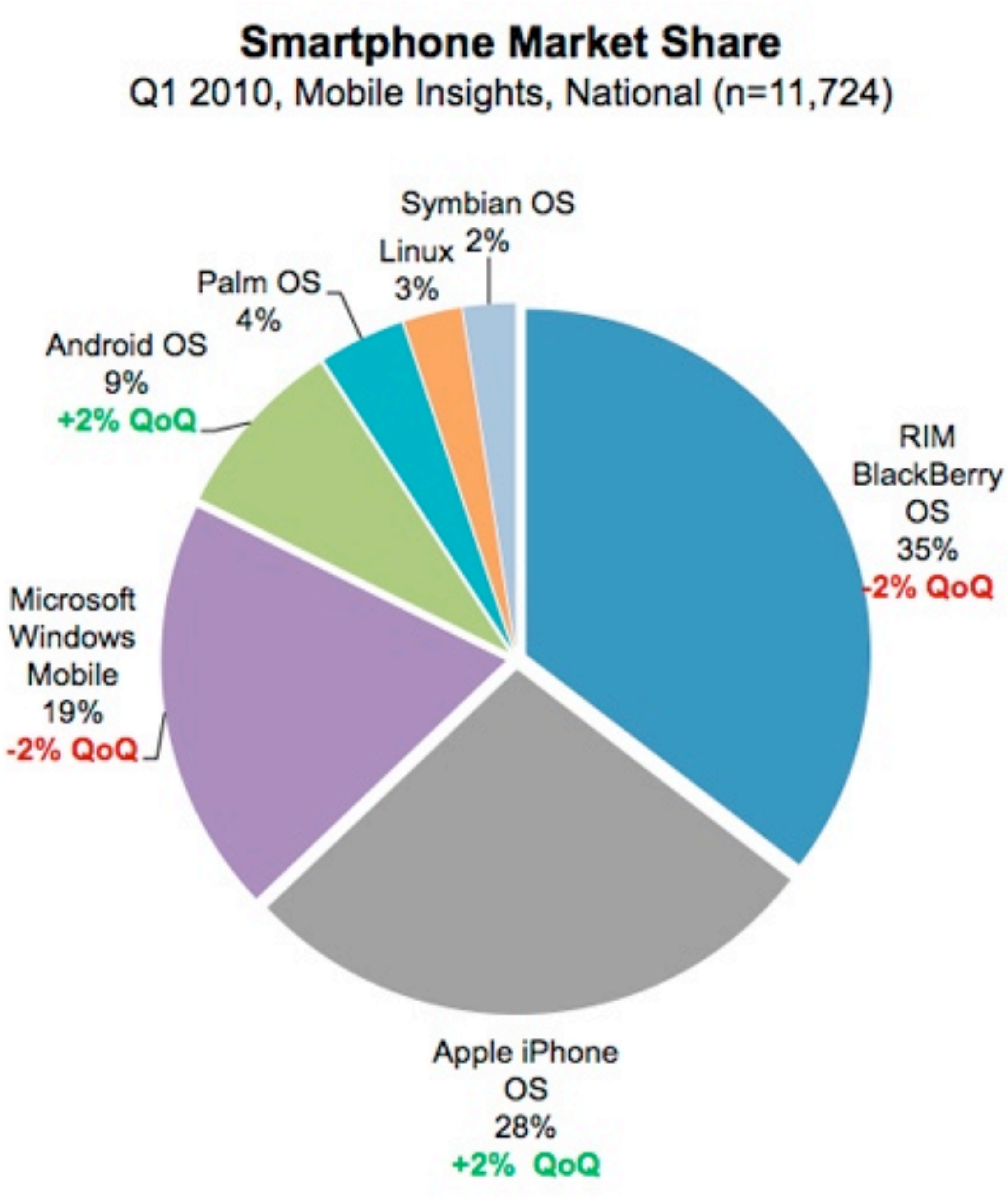
How To Build A Mobile Website

Jon Raasch

Over the past few years, mobile web usage has considerably increased to the point that web developers and designers can no longer afford to ignore it. In wealthy countries, the shift is being fueled by faster mobile broadband connections and cheaper data service. However, a large increase has also been seen in developing nations where people have skipped over buying PCs and gone straight to mobile.

Unfortunately, the mobile arena introduces a layer of complexity that can be difficult for developers to accommodate. Mobile development is more than cross-browser, it should be cross-platform. The vast number of mobile devices makes thorough testing a practical impossibility, leaving developers nostalgic for the days when they only had to support legacy browsers.

In addition to supporting different platforms, each device may use any number of mobile web browsers. For instance, an Android user could access your site using the native Android browser, or could have also installed Opera Mini or Firefox Mobile. It's fine as long as the smartphone uses a progressive web browser (and it's safe to say that most browsers are progressive nowadays), but it doesn't have to.



Source: Nielsen Study, Image credit

The mobile web reintroduces several issues that have been largely ignored in recent years. First, even with 4G networks, bandwidth becomes a serious issue for mobile consumers. Additionally, mobile devices have a significantly reduced screen size, which presents screen real estate issues that have not existed since the days of projection monitors. Combine these issues with cross-platform compatibility problems, and it isn't hard to see how mobile development is a lot like 'stepping backwards in time'. So let's tackle these issues one at a time and create a road map for mobile web development:

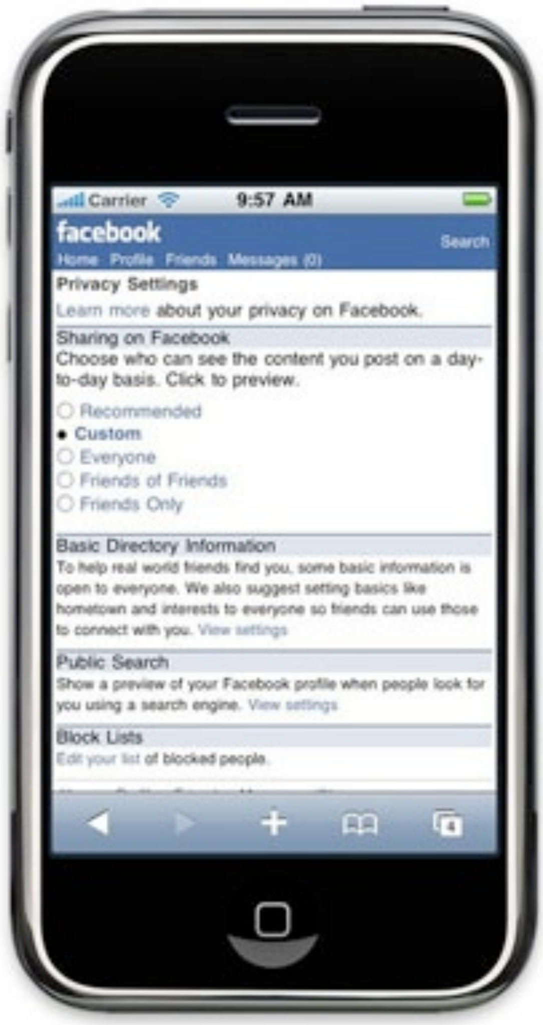
- How To Implement Mobile Stylesheets
- What To Change With Mobile Stylesheets
- Beyond Stylesheets
- Special Concerns For iPhone / iPad

How To Implement Mobile Stylesheets

The first step to adding mobile support to a website is including a special stylesheet to adjust the CSS for mobile devices:

SERVER-SIDE METHODS & THE UA STRING

One approach to including mobile stylesheets involves detecting the user agent string with a server-side language such as PHP. With this technique, the site detects mobile devices and either serves an appropriate stylesheet or redirects the user to a mobile subdomain, for instance m.facebook.com. This server-side approach has several advantages: it guarantees the highest level of compatibility and also allows the website to serve special mark-up/content to mobile users.



Privacy Settings

Learn more about your privacy on Facebook.

Sharing on Facebook

Choose who can see the content you post on a day-to-day basis. Click to preview.

○ Recommended
● Custom
○ Everyone
○ Friends of Friends
○ Friends Only

Basic Directory Information

To help real world friends find you, some basic information is open to everyone. We also suggest setting basics like hometown and interests to everyone so friends can use those to connect with you. View settings

Public Search

Show a preview of your Facebook profile when people look for you using a search engine. View settings

Block Lists

Edit your list of blocked people.

Large image

While this technique is perfect for enterprise level websites, there are practical concerns that make it difficult to implement on most sites. New user agent strings come out almost daily, so keeping the UA list current is next to impossible. Additionally, this approach depends on the device to relay its true user agent. Even though, browsers have spoofed their UA string to get around this type of detection in the past. For instance, most UA strings still start with "Mozilla" to get through the Netscape checks used in the 90′s, and for several years Opera pretended to be IE. As Peter-Paul Koch writes:

> *"It's an arms race. If device detection really catches on, browsers will start to spoof their user agent strings to end up on the right side of the detects."*

CLIENT-SIDE METHODS & MEDIA QUERIES

Alternately, the easiest approach involves detecting the mobile device on the client side. One of the earliest techniques for including mobile stylesheets involves taking advantage of the stylesheet's media type, for instance:

```
<link rel="stylesheet" href="site.css" media="screen" />
<link rel="stylesheet" href="mobile.css" media="handheld" />
```

Here we've included two stylesheets, the first **site.css** targets desktops and laptops using the **screen** media type, while the second **mobile.css** targets mobile devices using **handheld**. While this would otherwise be an excellent approach, device support is another issue. Older mobile devices tend to support the **handheld** media type, however they vary in their implementation: some disable the **screen** stylesheets and only load **handheld**, whereas others load both.

Additionally, most newer devices have done away with the **`handheld`** distinction altogether, in order to serve their users fully-featured web pages as opposed to duller mobile layouts. To support newer devices, we'll need to use media queries, which allow us to target styles to the device width (you can see another practical adaptation of media queries in Ethan Marcotte's article Responsive Web Design). Since mobile devices typically have smaller screens, we can target handheld devices by detecting screens that are 480px and smaller:

```
<link rel="stylesheet" href="mobile.css" media="only screen
and (max-device width:480px)"/>
```

While this targets most newer devices, many older devices don't support media queries, so we'll need a hybrid approach to get the largest market penetration.

First, define two stylesheets: `screen.css` with everything for normal browsers and **`antiscreen.css`** to overwrite any styles that you don't want on mobile devices. Tie these two stylesheets together in another stylesheet **`core.css`**:

```
@import url("screen.css");
@import url("antiscreen.css") handheld;
@import url("antiscreen.css") only screen and
(max-device-width:480px);
```

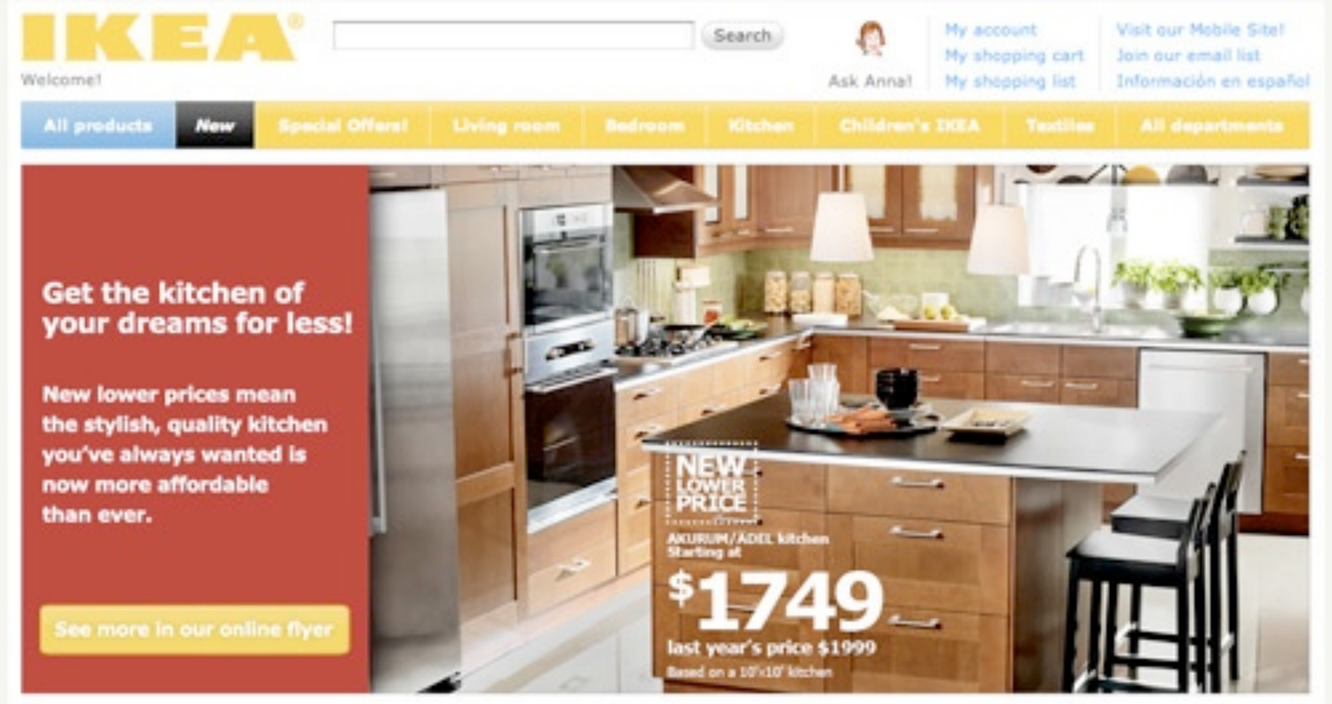
Finally, define another stylesheet **`handheld.css`** with additional styling for mobile browsers and link them on the page:

```
<link rel="stylesheet" href="core.css" media="screen"/>
<link rel="stylesheet" href="handheld.css" media="handheld,
only screen and (max-device-width:480px)"/>
```

While this technique reaches a large market share of mobile devices, it is by no means perfect. Some mobile devices such as iPad are more than 480 pixels wide and will not work with this method. However, these larger devices arguably don't need a condensed mobile layout. Moving forward, there will likely be more devices that don't fit into this mold. Unfortunately, it is very difficult to future-proof mobile detection, since standards are still emerging.

Besides device detection, the media query approach also presents other issues. Mainly, media queries can only style content differently and provide no control over content delivery. For instance, a media query can be used to hide a side column's content, but it cannot prevent that mark-up from being downloaded by your users. Given mobile bandwidth issues, this additional HTML should not simply be ignored.

USER INITIATED METHOD



Get the kitchen of your dreams for less!

New lower prices mean the stylish, quality kitchen you've always wanted is now more affordable than ever.

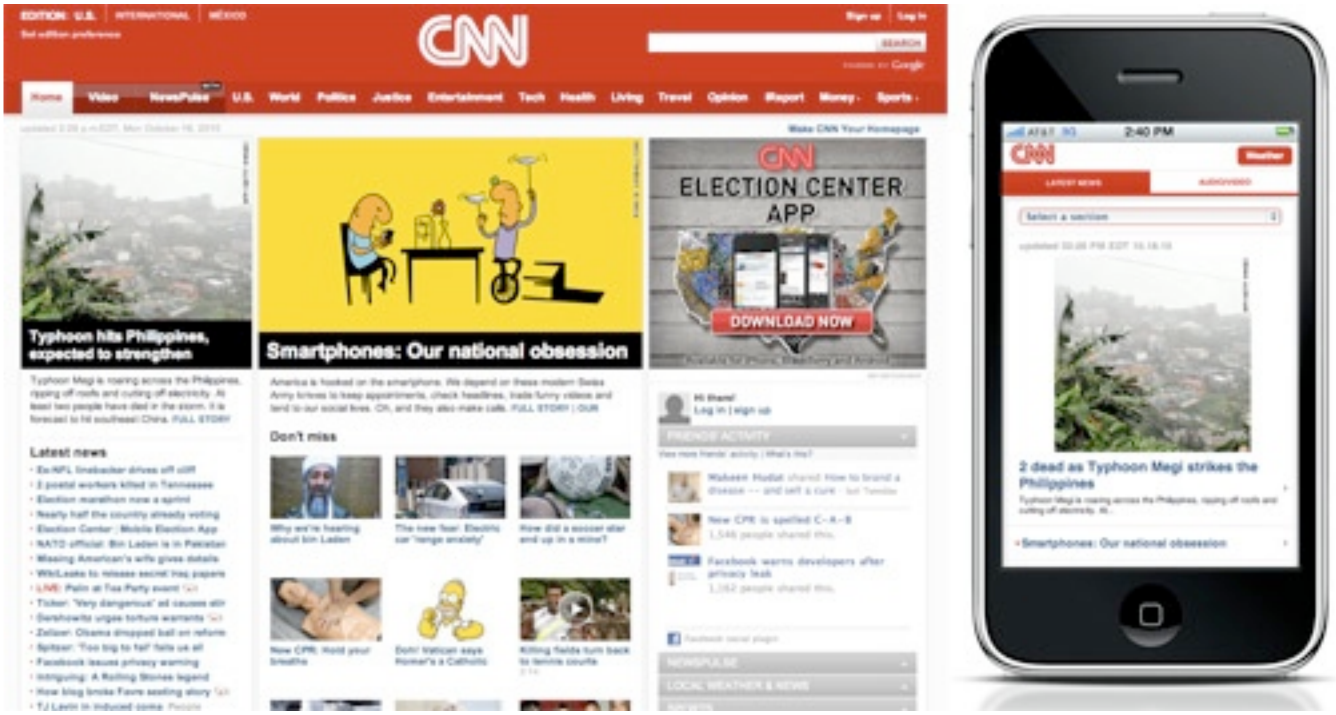
Considering the difficulties with mobile UA detection and the pitfalls of media queries, some companies such as IKEA have opted to simply allow the user to decide whether to view the mobile version of their website. While this has the clear disadvantage of requiring more user interaction, it is arguably the most fool-proof method and also the easiest to accomplish.

The site contains a link that reads “Visit our mobile site” which transports the user to a mobile subdomain. This approach has some drawbacks. Of course, some mobile users may miss the link, and other non-mobile visitors may click it, since it is visible regardless of what device is being used. Even though, this technique has the advantage of allowing the user to make the mobile decision. Some users prefer a condensed layout that is optimized for their device, whereas other users may prefer to access the entire website, without the restrictions of a limited mobile layout.

What To Change With Mobile Stylesheets

Now that we’ve implemented mobile stylesheets, it’s time to get down to the nuts and bolts of which styles we actually want to change.

INCREASE & ALTER SCREEN REAL ESTATE


CNN
Typhoon hits Philippines, expected to strengthen
Smartphones: Our national obsession
ELECTION CENTER APP
DOWNLOAD NOW
Latest news
Don't miss
2 dead as Typhoon Megi strikes the Philippines
Smartphones: Our national obsession

The primary goal of mobile stylesheets is to alter the layout for a smaller display. First and foremost this means reducing multi-column layouts to single columns. Most mobile screens are vertical, so horizontal space becomes even more "expensive" and mobile layouts can rarely afford more than one column of content. Next, reduce clutter throughout the page by setting **`display: none;`** on any less important elements. Finally, save additional pixels by reducing margins and padding to create a tighter layout.

REDUCE BANDWIDTH

Another goal of mobile stylesheets is to reduce bandwidth for slower mobile networks. First make sure to remove or replace any large background images, especially if you use a background image for the whole site. Additionally set **`display: none`** on any unnecessary content images.

If your site uses images for buttons or navigation, consider replacing these with plain-text / CSS counterparts. Finally if you'd like to force the browser to use the alternate text for any of your images, use this snippet (and use JavaScript to add the **`as-text`** class for **`img`** and make sure that **`alt-`**attributes are properly defined in your markup):

```
img.as-text { content: attr(alt); }
```

OTHER CHANGES

Besides addressing screen size and bandwidth concerns, there are a few additional changes that should be made in any mobile stylesheet. First, you can improve readability by increasing the font size of any small or medium-sized text. Next, clicking is generally less precise on mobile devices, so make sure to increase the clickable areas of any important buttons or links by setting **`display: block`** and adding padding to the clickable elements.

Additionally, floated elements can cause problems for mobile layouts, so consider removing any floats that aren't absolutely necessary. Remember that horizontal real estate is especially expensive on mobile, so you should always opt for adding vertical scrolling as opposed to horizontal.

Finally, mouseover states do not work with most mobile devices, so make sure to have proper definitions of **`:active-`**states. Also, sometimes it may be useful to apply definitions from the already defined **`:hover`** states to the **`:active`** states. This pseudo-class is displayed when the user clicks an item, and therefore will work on mobile devices. However this only enhances the user experience and should not be relied on for more important elements, such as drop-down navigation. In these cases it is best to show the links at all times in mobile devices.

Beyond Stylesheets

In addition to mobile stylesheets, we can add a number of special mobile features through mark-up.

CLICKABLE PHONE NUMBERS

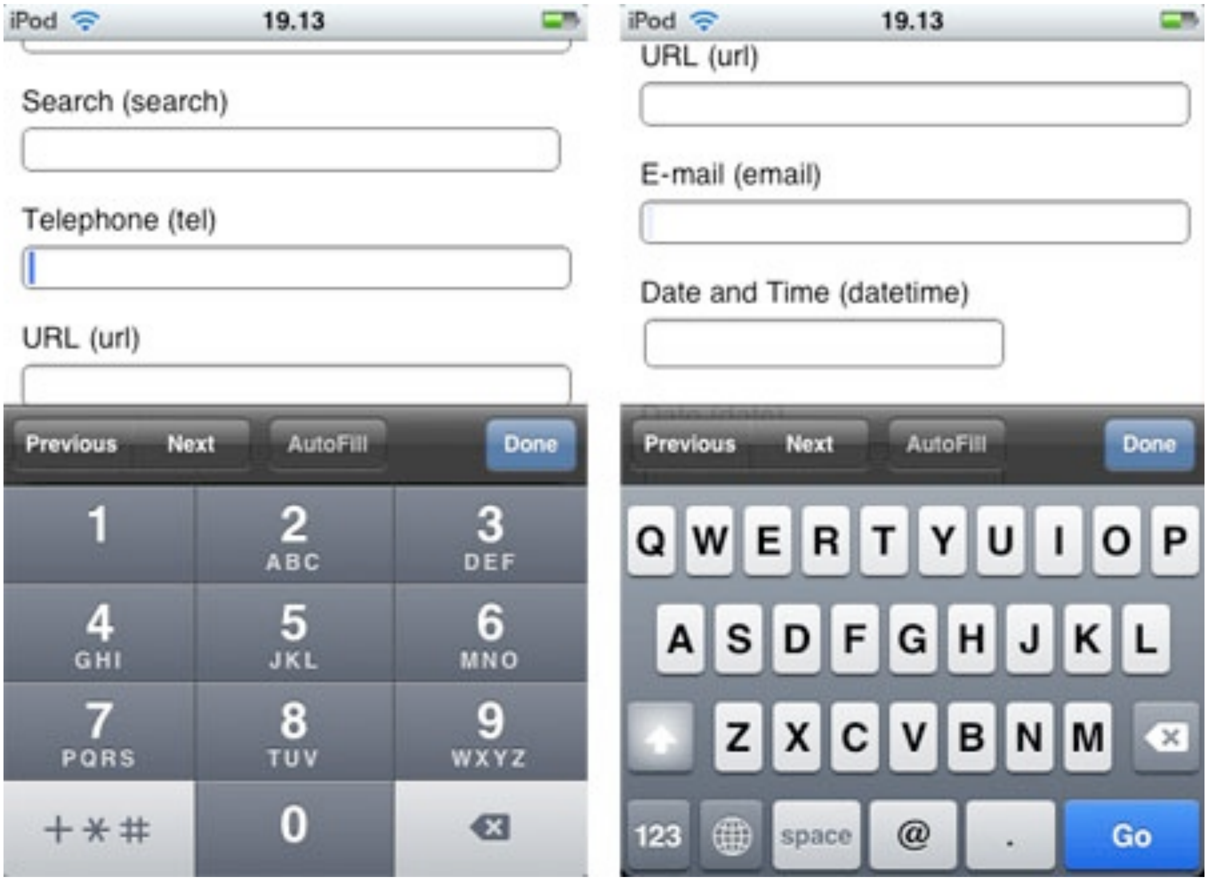
First, most handheld devices include a phone, so let's make our phone numbers clickable:

```
<a href="tel:15032084566" class="phone-link">(503) 208-4566</
a>
```

Now mobile users can click this number to call it, however there are a few things to note. First, the number in the actual link starts with a 1 which is important since the web is international (1 is the US country code).

Second, this link is clickable whether or not the user has a mobile device. Since we're not using the server-side method described above, our best option is to simply hide the fact that the number is clickable via CSS. So use the **`phone-link`** class to disable the link styling in your screen stylesheet, and then include it again for mobile.

SPECIAL INPUT TYPES



When it comes to mobile browsing, another concern is the difficulty of typing compared to a standard full-sized keyboard. But we can make it easier on our users by taking advantage of some special HTML5 input types:

```
<input type="tel" />
<input type="email" />
```

These input types allow devices such as iPhone to display a **contextual keyboard** that relates to the input type. In the example above `type="tel"` triggers a numeric keypad ideal for entering phone numbers, and `type="email"` triggers a keypad with `@` and `.` buttons.

HTML5 input types also provide in-browser validation and special input menus that are useful in both mobile and non-mobile browsing. Furthermore, since non-supportive browsers naturally degrade to view these special input types as **`<input type="text" />`**, there's no loss in using HTML5 input types throughout your websites today.

See a complete list of HTML5 input types. You can find some information about the current browser support of HTML5 input attributes in the post HTML5 Input Attributes & Browser Support by Estelle Weyl.

VIEWPORT DIMENSIONS & ORIENTATION

When modern mobile devices render a webpage, they scale the page content to fit inside their viewport, or visible area. Although the default viewport dimensions work well for most layouts, it is sometimes useful to alter the viewport. This can be accomplished using a **`<meta>`** tag that was introduced by Apple and has since been picked up by other device manufacturers. In the document's **`<head>`** include this snippet:

```
<meta name="viewport" content="width=320" />
```

In this example we've set the viewport to `320`, which means that 320 pixels of the page will be visible across the width of the device.

The viewport meta tag can also be used to disable the ability to resize the page:

```
<meta name="viewport" content="width=320,user-
scalable=false" />
```

However, similar to disabling the scrollbars, this technique takes control away from the user and should only be used for a good reason.

Additionally, it is possible to add certain styles based on the device orientation. This means that different styles can be applied depending on whether the user is holding their phone vertically or horizontally.

To detect the device orientation, we can use a media query similar to the client-side device detection we discussed earlier. Within your stylesheet, include:

```
@import url("portrait.css") all and
(orientation:portrait);
@import url("landscape.css") all and
(orientation:landscape);
```

Here **`portrait.css`** styles will be added for vertical devices and the **`landscape.css`** will be added for horizontal.

However orientation media queries have not been adopted by all devices, so this is best accomplished with the **`max-width`** media query. Simply apply different max-width queries for the different orientation widths you want to target. This is a much more robust approach, since presumably the reason to target different orientations is to style for different widths.

Special Concerns For iPhone / iPad

With a market share of 28% and estimates of as much as 50% of mobile browsing going through iPhone, it makes sense that developers make special accommodations for the mobile giant.

NO FLASH

Regardless of Apple's ethics, the reality is that iPhones do not play Flash unless they are jailbroken. Fortunately, there are alternatives to Flash, and iPhone's issues with this technology are often easy to get around. The main use for Flash in modern websites is Flash video, which can easily be circumvented using HTML5 video. However since older browsers don't support HTML5, make sure to include a Flash backup for non-supportive browsers (this is why the whole debate about Flash vs. HTML5 is a bit pointless, because you can actually offer both to your users and the user's device will pick up the one it can render automatically).

Beyond video, it is usually best to use JavaScript to accommodate any simple functionality. JavaScript libraries such as jQuery make it easy to build rich interactive applications without Flash. Regardless of your desire to support iPhone, these JavaScript apps typically have a number of additional advantages over Flash alternatives.

Finally, certain applications are simply too hard to recreate with HTML5 and Javascript. For these, iPhone users will have to be left out, however make sure to include appropriate alternate content.

A spoof of Adobe's "We Love Apple" campaign, where the heart is replaced by the broken plugin icon.

OTHER SHORTCOMINGS

Besides Flash, there are a few additional caveats to supporting iPhones and iPads.

First, iPhone does not support **`<input type="file" />`**, since it does not have an accessible internal file structure. While most mobile devices connect to a computer as an external hard-drive, Apple has taken steps to ensure that the iPhone file structure remains obfuscated.

Next, iPhone will only cache files that are **25 kb** or less, so try to keep any reused files under this restriction. This can be a bit counter-intuitive, as it often means breaking out large image sprites and concatenated JavaScripts into smaller chunks. However be careful to serve these files only to iPhone, or it will cause extra HTTP requests in all other browsers.

Finally, when it comes to **`@font-face`** font embedding, iPhone's Mobile Safari doesn't fully support it and supports the **SVG file format** instead. However, SVG fonts are only supported by Chrome, Opera and iPhone, so we'll need a hybrid approach to target all browsers. In addition to the SVG, we'll need an .otf or .ttf for Firefox and Safari, as well as an EOT for IE (IE has actually supported **`@font-face`** since IE4).

After obtaining the necessary files, tie them all together with the appropriate CSS:

```
@font-face {
    font-family: 'Comfortaa Regular';
    src: url('Comfortaa.eot');
    src: local('Comfortaa Regular'),
         local('Comfortaa'),
         url('Comfortaa.ttf') format('truetype'),
         url('Comfortaa.svg#font') format('svg');
}
```

For more information, read this article on cross-platform font-face support.

SPECIAL IPHONE / IPAD ENHANCEMENTS

Despite iPhone's various shortcomings, the device offers a wonderfully rich user experience that developers can leverage in ways not possible with older mobile devices.

First, there are a variety of JavaScript libraries that can be used to access some of the more advanced functionality available in iPhone. Take a look at Sencha Touch, jQTouch and iui. These three libraries allow you to better interface with the iPhone, and also work on similar devices such as Android. Additionally, keep an eye on the much anticipated jQuery Mobile which has just been released in alpha.

Next, the App Store isn't the only way to get an icon on your users' iPhones: you can simply have them bookmark your page. Unfortunately the default bookmark icon is a condensed screen shot of the page, which doesn't usually look very good, so let's create a special iPhone icon. Also check the Icon Reference Chart by Jon Hicks for further details.

Start by saving a 57 x 57 pixel PNG somewhere on your website, then add this snippet within your **`<head>`** tag:

```
<link rel="apple-touch-icon" href="/customIcon.png"/>
```

Don't worry about rounded corners or a glossy effect, iPhone will add those by default.

Conclusion

As the worldwide shift to mobile continues, handheld device support will become increasingly important. Hopefully this article has left you with both the desire and toolset necessary to make mobile support a reality in your websites.

Although mobile occupies a significant chunk of global web browsing, the technology is still very much in its infancy. Just as standards emerged for desktop browsing, new standards are emerging to unify mobile browsers. This means that the techniques described in this article are only temporary, and it is your responsibility to stay on top of this ever-changing technology.

In fact, the only thing in web development that remains constant is the perpetual need to continue learning!

Making It A Mobile Web App

Kim Pimmel

Ask any interactive agency nowadays what their clients are asking for when they need a mobile experience — the answer will inevitably be “an iPhone and/or an iPad app.” Native Apple apps are a hot commodity, and in today’s mobile application ecosystem, mobile web apps are not sexy. In fact, many people don’t even realize they are even an option. In certain cases, an iPhone/iPad app will be the right solution for their needs.

However, there are some situations where it may become a short-term win, but eventually a long-term loss. Mobile web apps offer a good number of advantages over native apps; and though they face some design, development and deployment challenges, they are a powerful cross platform, scalable and affordable solution.

INCREASING FRAGMENTATION

Mobile apps are all the rage. There are a slew of startups targeting the iPad, countless entrepreneurs hacking together the next killer iPhone app, and it seems as though every big company has released an app of some sort. With the increasing penetration of Android phones, developers are scrambling to port their software.

But what about deploying to Windows Phone 7, Blackberry and Symbian? Who wants to study yet another SDK, learn another language, and go through yet *another* app submission process? Who will continue to keep the code up to date for all these platforms as each one splinters into new incarnations, releases new hardware and OS updates. Fragmentation is a costly long-term investment. And people are beginning to realize that native apps are not a sustainable long-term solution for all their needs.

GAP StyleMixer application for both iOS & Android

THE MOBILE WEB IS EVERYWHERE

As the native mobile app market becomes increasingly fragmented, it is becoming clear that there needs to be a solution which can re-use code and designs across platforms, and which eases deployment headaches. But why invent a new solution, when it already exists on every device out there: the Web. Webkit is gaining ground as the de facto standard for rendering web content, but even Webkit isn't appropriate for every application. It wouldn't be recommended for experiences that need complex graphics rendering, require hooks into specific hardware such as camera or accelerometer, or have hefty media requirements.

Though these constraints will change over time. But for all other apps that don’t need these features, using the mobile web frees developers to use their web technology of choice, so long as it will render on mobile browsers. Design and develop once, deploy everywhere. With smart design and code, a single web app could render appropriately on differing resolutions and screen sizes, and respond accordingly to touch, 5-way or cursor. Indeed, frameworks for mobile web app development already exist, such as Sencha Touch.

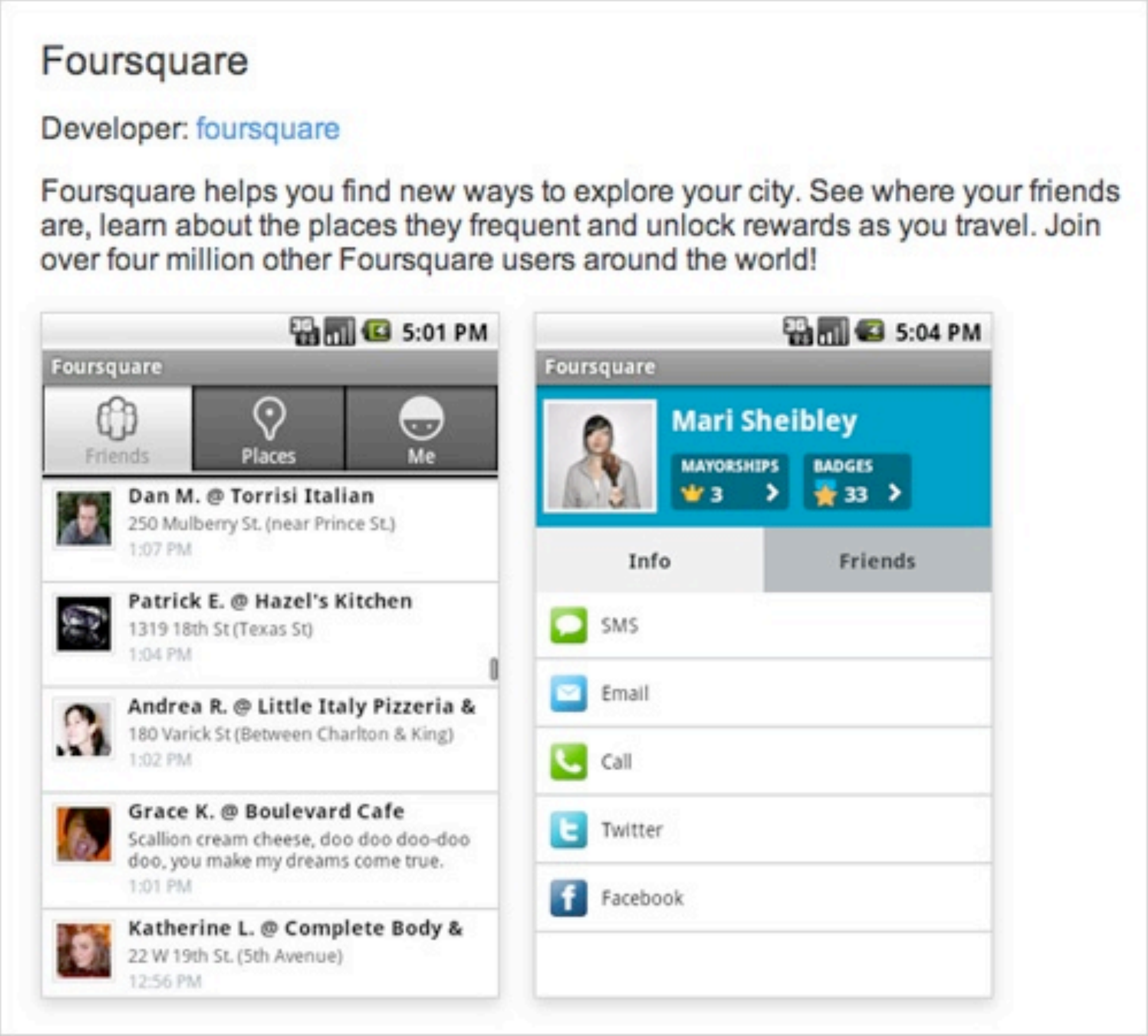
Sencha Touch Mobile Web Development Framework

OLD NEWS

Desktop web apps are far from a new idea — Rich Internet apps have been around for a while. Google has been pushing in this direction for years, creating a broad suite of online tools, primarily for the desktop, with an increasing focus on mobile. However, web apps have been slow to gain traction in the mobile space. Even with Apple promoting mobile web apps as the next best thing on their 1st generation iPhone in 2007, the focus is still squarely on native apps. And the primary reason for this is due to the overwhelming success of Apple's (native) App Store.

THE APP STORE MODEL

Apple's App Store was not the first to distribute native applications to mobile phones, but they proved it was a viable model, and launched the concept into popular culture. It's this same model that would be necessary to make a mobile web app ecosystem successful.



Foursquare

Developer: foursquare

Foursquare helps you find new ways to explore your city. See where your friends are, learn about the places they frequent and unlock rewards as you travel. Join over four million other Foursquare users around the world!

Dan M. @ Torrisi Italian
250 Mulberry St. (near Prince St.)
1:07 PM

Patrick E. @ Hazel's Kitchen
1319 18th St (Texas St)
1:04 PM

Andrea R. @ Little Italy Pizzeria &
180 Varick St (Between Charlton & King)
1:02 PM

Grace K. @ Boulevard Cafe
Scallion cream cheese, doo doo doo-doo doo, you make my dreams come true.
1:01 PM

Katherine L. @ Complete Body &
22 W 19th St. (5th Avenue)
12:56 PM

Foursquare App in Google's Android Market

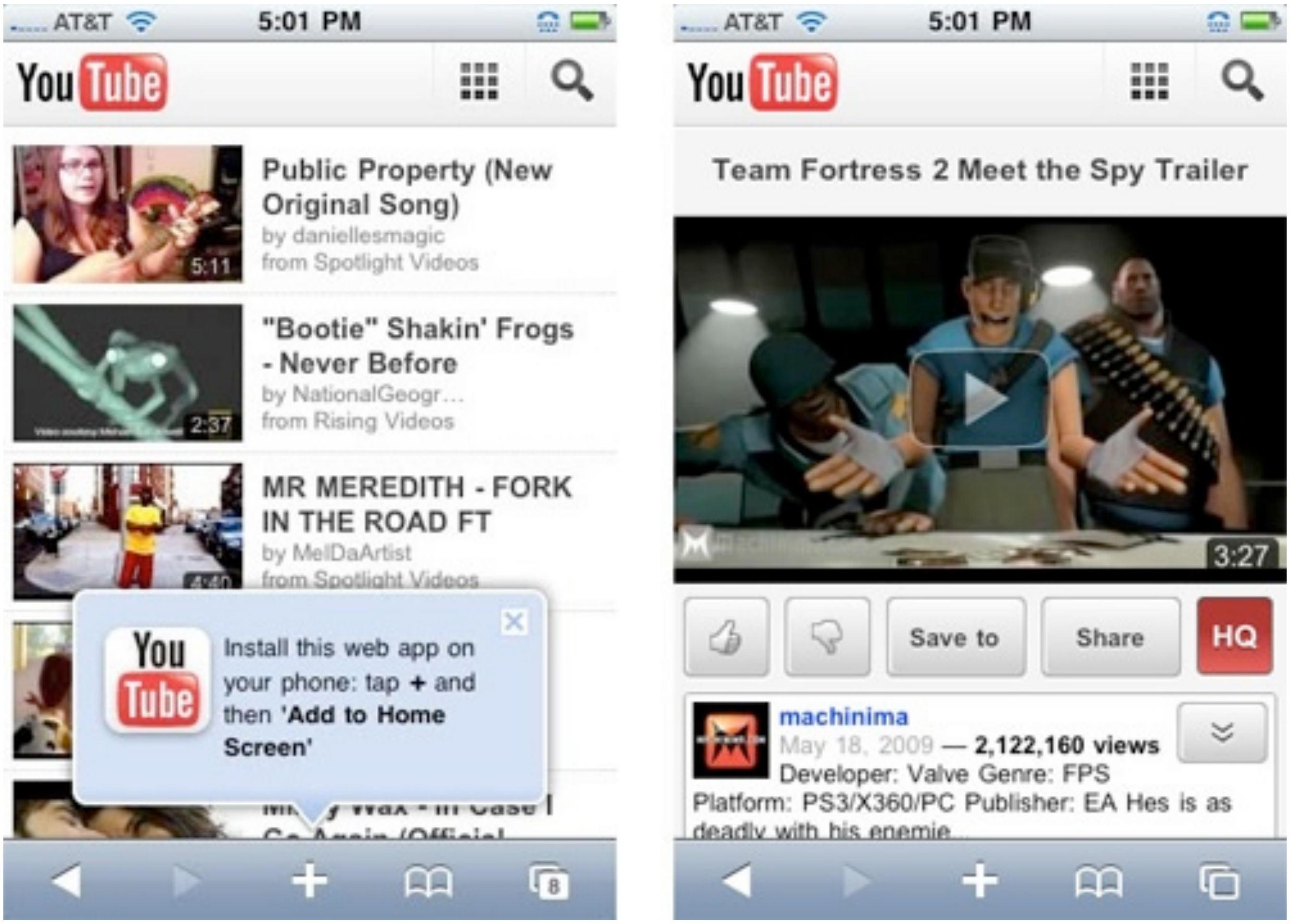
As a consumer, it's more appealing to go to one trusted online outlet for stuff than to waste time searching the web for the same thing, and putting yourself at risk of being hacked. Mobile web apps need a consolidated storefront for much the same reasons.

First, it’s easy to find apps when they are indexed, categorized, and searchable in one place. Second, a robust community of users exposing app popularity, contributing ratings and writing reviews makes it easier to evaluate your choices. Third, when I’ve decided to buy a game such as Plants and Zombies, I want to be sure my purchase will be a safe one — something a robust app store from a recognized company should offer. And since a web app is cross-platform, you could play it on your Android phone, your iPad, and your desktop — all with a single purchase. Buy once, use anywhere. It’s magic!

As a business or developer creating web apps, a centralized web app store provides benefits over doing it solo. Most importantly, it provides a source of monetization. This is the key to driving adoption of a web app ecosystem, as without revenue, businesses and developers will stick with money-making native apps. It’s also a marketing channel, allowing for easy discovery and promotion. Another potential benefit of using a web app storefront would be the APIs to help developers deal with authentication, licensing and other technical hurdles of digital distribution.

IT’S POSSIBLE NOW

A great majority of native apps could be deployed today as full featured mobile web apps. The HTML5 family of technologies allow for refined typography, animation, streaming video, offline storage, and the list goes on. Probably the most high profile web app to date is the Youtube mobile site, which delivers a comparable experience to the native apps they have built.



Install this web app on your phone: tap + and then **'Add to Home Screen'**

May 18, 2009 — **2,122,160 views**
Developer: Valve Genre: FPS
Platform: PS3/X360/PC Publisher: EA Hes is as deadly with his enemie…

Youtube Mobile Web Experience

REAL WORLD CHALLENGES

As with any innovation, there are big questions that need to be answered. The most obvious is the issue of cross-platform compatibility. Building a robust and rich cross-platform mobile web app experience would benefit from HTML5 technology support, but currently RIM and Microsoft's mobile offerings use their own standards. This weakens the des/dev once, deploy anywhere story; but is by no means a dealbreaker. Web developers have long dealt with coding to accommodate troublesome browsers, and this would be a similar case.

Another challenge in the ‘deploy anywhere’ scenario arises when you look at how a given design translates across devices with varying resolutions, form factors and input methods. Application designers will need to approach this problem by targeting several key resolution/form factor combinations, similar to what is recommended by the Android SDK. Depending on what device an app is being run on, the design, layout and functionality may differ significantly. This can be solved using a combination of intelligent design and careful development.

Last but not least is the problem of providing consistent, quality user experiences in this new application space. We’ve seen how the Android’s app offerings often leave much to be desired in terms of visual design and usability while Apple has been more successful in defining quality experiences. Providing a set of best practices, design patterns, and components for designers would go a long way towards the creation of quality mobile web app experiences that would win over consumers. As mobile web apps gain credibility, we will see more offering such as Sencha Touch and Sproutcore that provide solid web development and experience frameworks.

THE INEVITABLE VICTORY OF THE WEB BROWSER

Web applications as ‘the next big idea’ might never happen — but in the coming years, more and more websites will have mobile incarnations that look a lot like applications. You’ll be swiping through articles, pinching photos, and flicking pests off your Farmville plot — all in your mobile browser. And people won’t even realize that in the end, the next generation mobile web won.

A Study Of Trends In Mobile Design

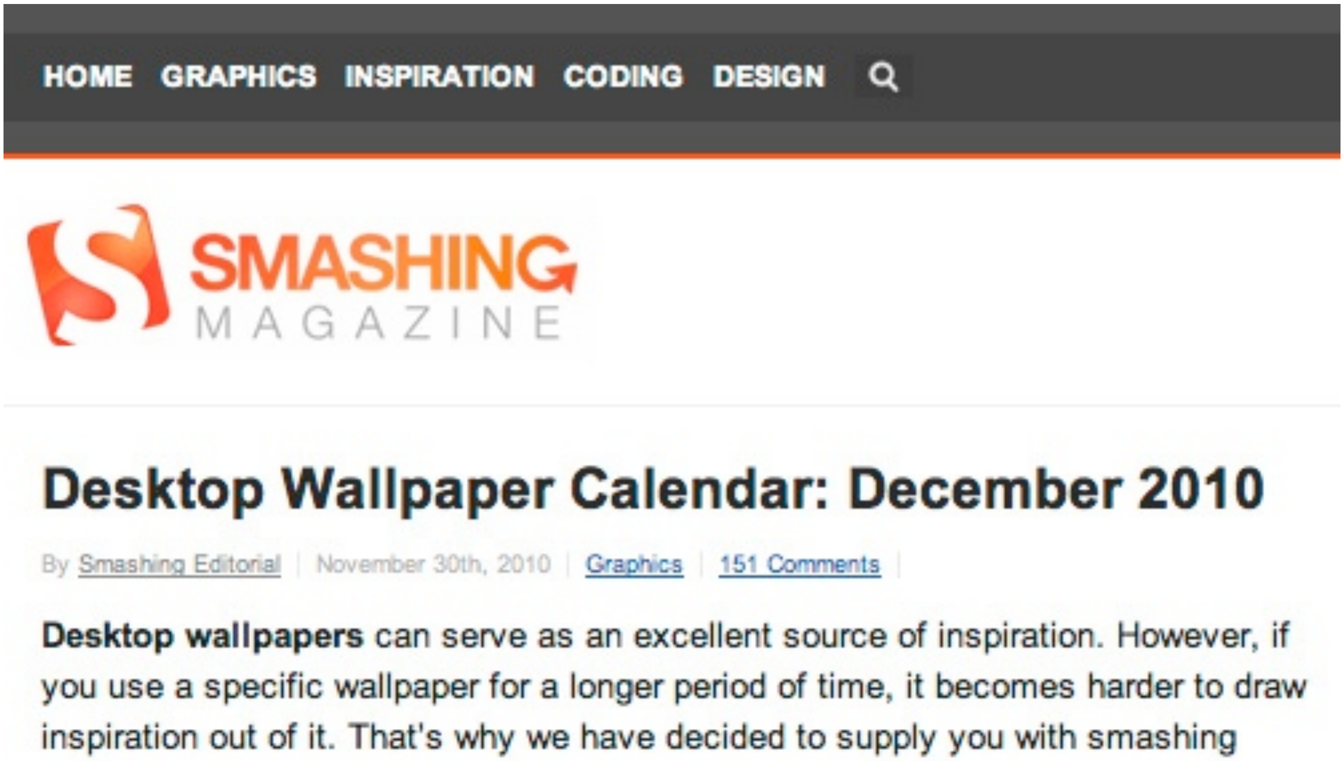
Alexander Dawson

The industry has evolved in many ways, but one particular area has affected how we build websites to a greater extent than any other. The surge in Web-enabled mobile devices has forged a subculture of visitors who require the adaptation of our websites to meet their needs. While mobile design is still in its infancy (and little primary research on mobile trends exist), we need to observe how this now-critical element of our industry is evolving, and the patterns which exist from current development efforts.

The aim of this article is to showcase the variety of methods in which some of today's most popular websites provide an interactive and (hopefully) useful mobile experience for their end users. There are plenty of big names which were analyzed, such as Facebook and Amazon, and you'll see plenty of useful graphs to draw some inspiration from. With statistics and some really interesting revelations on the diversity of modern design, you can be excited about the future of mobile Web design!

What is This Anyway?

To determine some best practices and common trends within the ever growing field of mobile Web design, a study was conducted to analyze how popular websites deal with important factors relating to the information architecture and design implementations within mobile design. Because this research could have covered any number of variables, it's important to state that this study isn't going to answer every one of your questions–but it hopefully may help you to learn a few things!



Desktop Wallpaper Calendar: December 2010

By Smashing Editorial | November 30th, 2010 | Graphics | 151 Comments |

Desktop wallpapers can serve as an excellent source of inspiration. However, if you use a specific wallpaper for a longer period of time, it becomes harder to draw inspiration out of it. That's why we have decided to supply you with smashing

Figure 1: Mobile design has become an essential component of a successful, compatible website. Smashing Magazine has a mobile version, too.

Of the websites that are measured and accounted for within the study, we not only examined how these websites deal with mobile devices and how visitors are served a mobile experience, but also if they had non-Web features, such as mobile apps, that can play a useful role in the process. In addition, some of the layout conventions, design choices, coding levels, and some useful features specific to mobile websites, have been measured (to their existence, their conformance level, and the method they undertake).

> ***Note:*** *While a great deal of effort has been put into making this study as accurate as possible, the number of variables being considered may result in anomalies. Factors such as websites without mobile experience have been accounted for (as has bias–to the greatest extent possible, during the study).*

Examining the Variables

Before presenting the results of this study, it's important to account for a few of the variables and methodology that came into play to explain how the research and results were formed. Using this information you can hopefully recognize the limitations of this particular study, and if you really feel adventurous, perhaps you could expand on the subject and conduct some research of your own to see how the results apply under different situations. It could make for interesting reading!

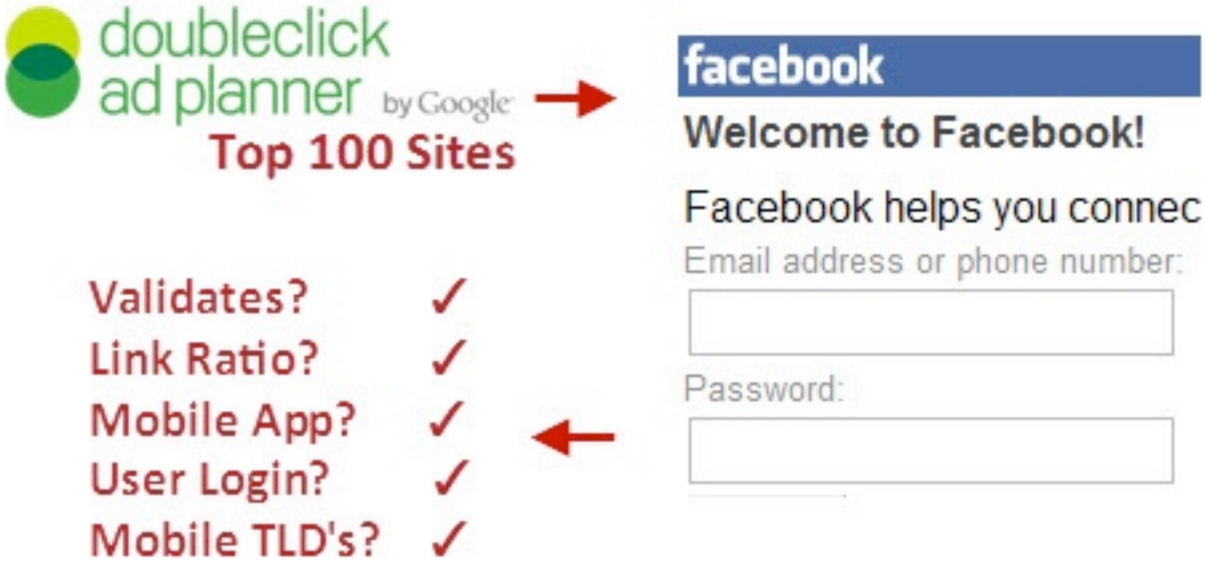


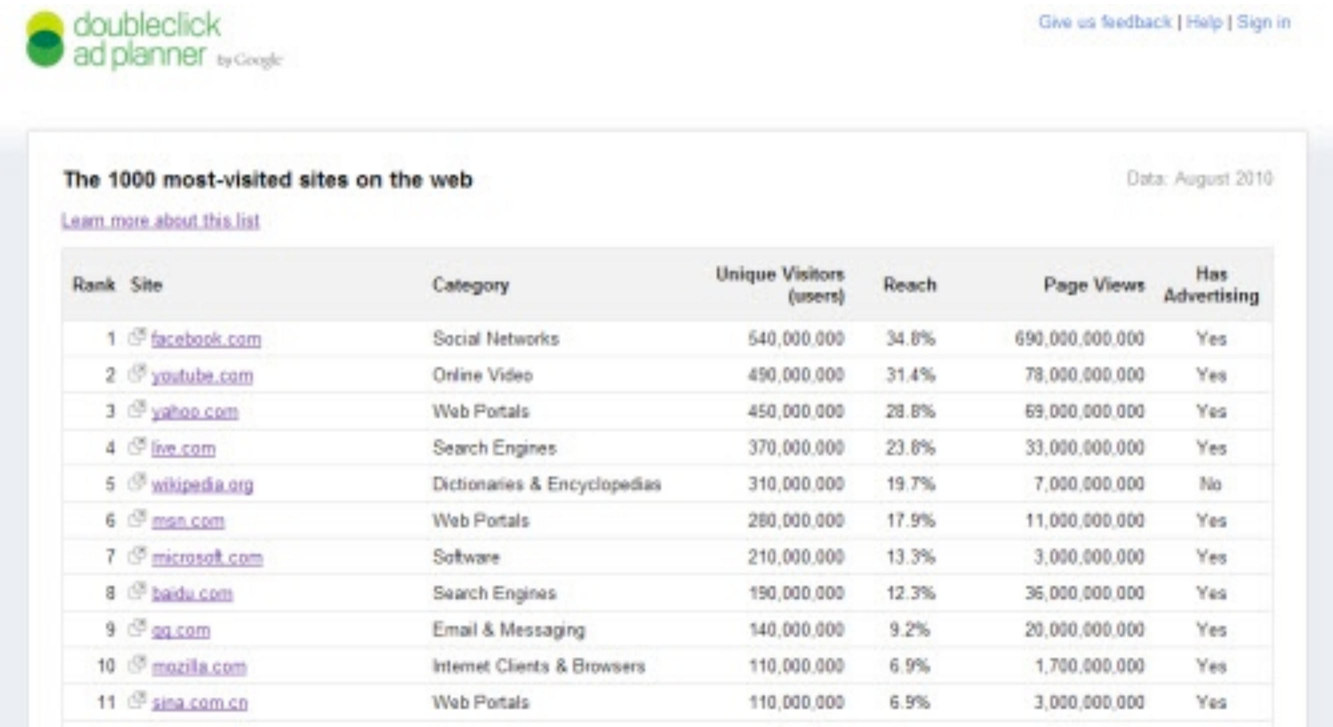
Figure 2: The above chart explains the process this study undertook to uncover its results

Regarding the methodology for this study, the protocol begin by selecting an independent group of sites (outsourced) and variables to test against–many of which had never been examined on such a scale (or in such depth) previously. The approval for which variables were included had to meet certain criteria; firstly, the results had to be interesting (and could affect how the mobile design situation is seen), and also had to be statistically significant (we don't want to state the obvious).

> ***Note:*** *Some variables were dropped from the final analysis due to a lack of conclusive, useful results. An example of this is the support for orientation (portrait and landscape modes); as a result of how browsers deal with the layout, this became a non-issue, seeing 100% native support.*

SITE SELECTION PROTOCOL

Picking a group of websites to analyze is, of course, a critical part of the process. To eliminate the potential for bias or for focusing on a niche, it was decided that the top 100 websites depicted through Google's AdPlanner list would be a suitable candidate. While the list held 1,000 websites and the study could easily be expanded, we used the initial 100 websites to receive a good sample and to provide enough variety for the baseline results. This level is used as the qualifier for statistically significant results.



The 1000 most-visited sites on the web

Data: August 2010

Learn more about this list

Rank	Site	Category	Unique Visitors (users)	Reach	Page Views	Has Advertising
1	facebook.com	Social Networks	540,000,000	34.8%	690,000,000,000	Yes
2	youtube.com	Online Video	490,000,000	31.4%	78,000,000,000	Yes
3	yahoo.com	Web Portals	450,000,000	28.8%	69,000,000,000	Yes
4	live.com	Search Engines	370,000,000	23.8%	33,000,000,000	Yes
5	wikipedia.org	Dictionaries & Encyclopedias	310,000,000	19.7%	7,000,000,000	No
6	msn.com	Web Portals	280,000,000	17.9%	11,000,000,000	Yes
7	microsoft.com	Software	210,000,000	13.3%	3,000,000,000	Yes
8	baidu.com	Search Engines	190,000,000	12.3%	36,000,000,000	Yes
9	qq.com	Email & Messaging	140,000,000	9.2%	20,000,000,000	Yes
10	mozilla.com	Internet Clients & Browsers	110,000,000	6.9%	1,700,000,000	Yes
11	sina.com.cn	Web Portals	110,000,000	6.9%	3,000,000,000	Yes

Figure 3: The Google AdPlanner top 1,000 websites list is the unbiased resource used to study from

VARIABLES (PER CATEGORY)

The results of this study wouldn’t be anything without its variables. When deciding what to test against, the focus became twofold: how the mobile experience is activated, and how that experience functions. How visitors are directed to a mobile experience became worthy of attention due to the increasing number of implementations that exist. Secondly, it was vital to test those pages to ensure they accounted for speed, bandwidth, display size, touch screens, and other best practices.

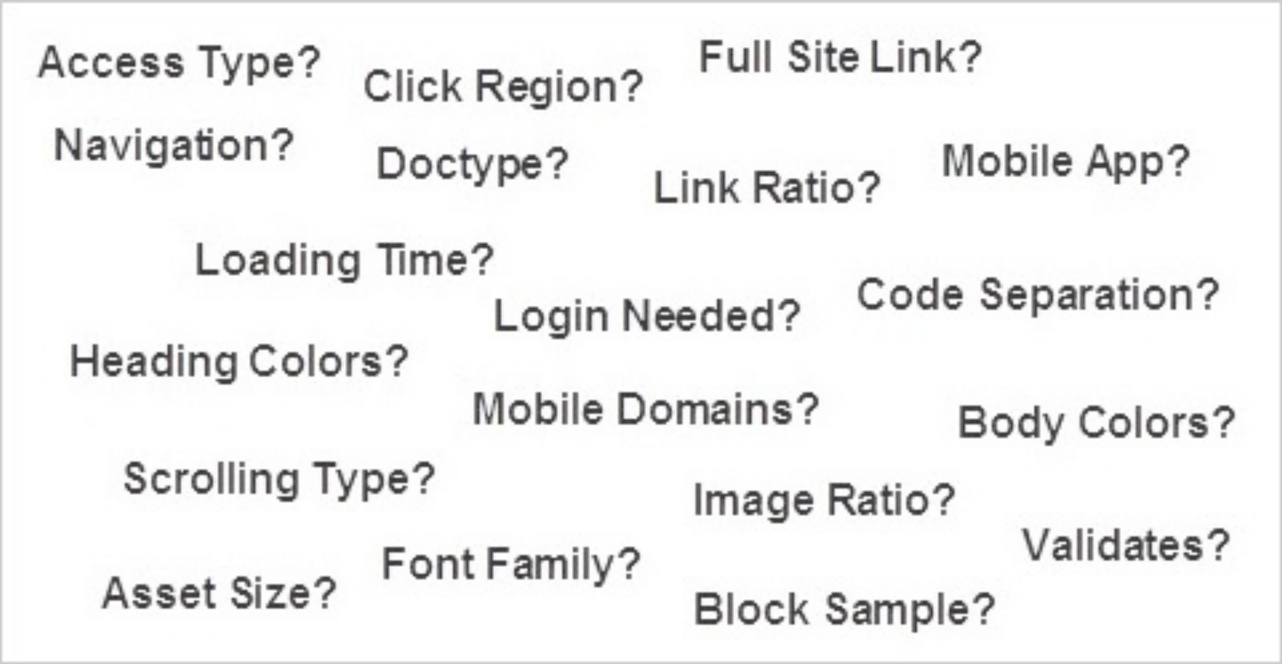


Figure 4: Plenty of variables were considered to give you some informative results to learn from

MARGINS FOR ERROR

As with any study, there is always a potential for error or bias to occur. To avoid as many of these issues as possible not only were the websites independently sourced, but all testing was undertaken on a desktop machine, several handheld devices, and a number of emulators (this occurred on every website). Following this practice reinforced the results (avoiding erroneous numbers), and the testing was verified on two separate dates to ensure that the experience resulted from consistent practices.

Figure 5: Emulators were used to expand the number of physical devices included

And Our Survey Says...the Results

Now that all of the basics (as to how the study was undertaken) have been explained, it's time we get down to the really interesting stuff–the results! You should be prepared for plenty of charts and graphs, and some of the results may be really surprising. We've broken down each type of result into its own subsection and have provided various ways the results can be interpreted, so hopefully the findings of this study will be quite apparent. Without any further delay, let's get down to business!

METHOD OF ACCESS

Within this test it was important to establish whether a mobile experience was made apparent to a user immediately. When a user proactively visits a website they want to be made aware that their device is either supported, or that the regular website will load. In this test we not only examined whether a redirection or device detection occurred, but also if the mobile experience was provided on the standard website (rather than a subdomain), and if a regular PC could switch to the mobile version too.

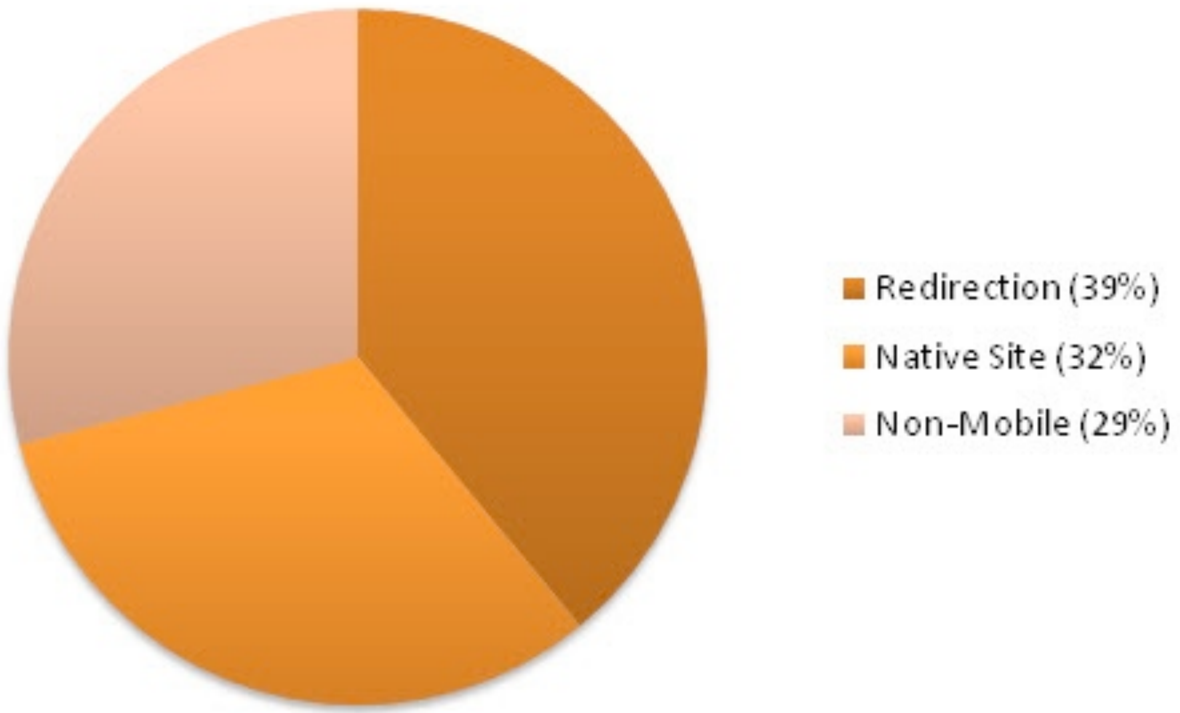


Figure 6: The proportion of automatic redirects against desktop websites with optional mobile support

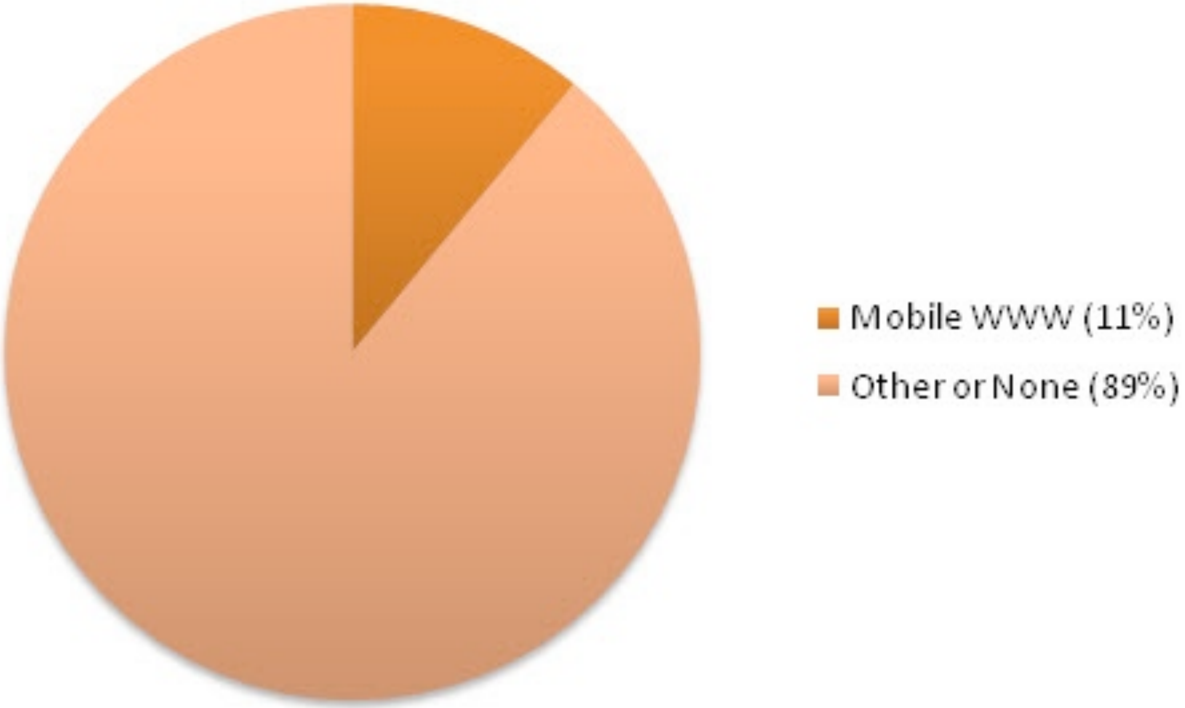


Figure 7: The distribution of websites which implemented a mobile design on the WWW subdomain

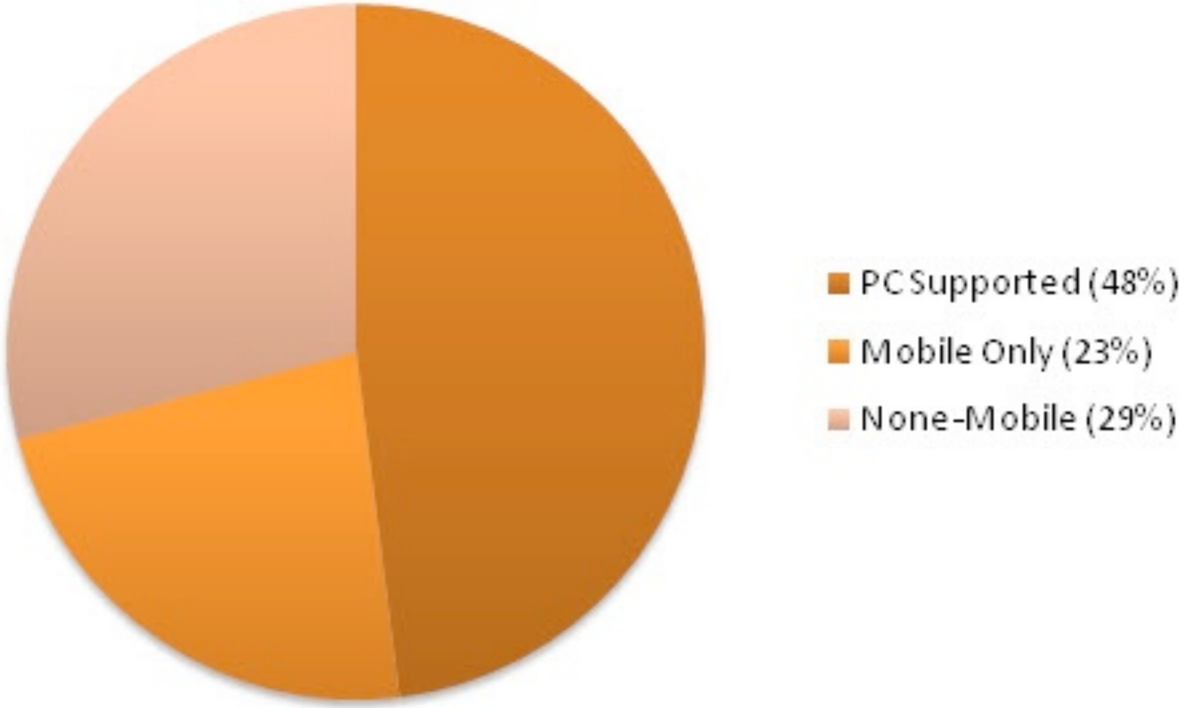
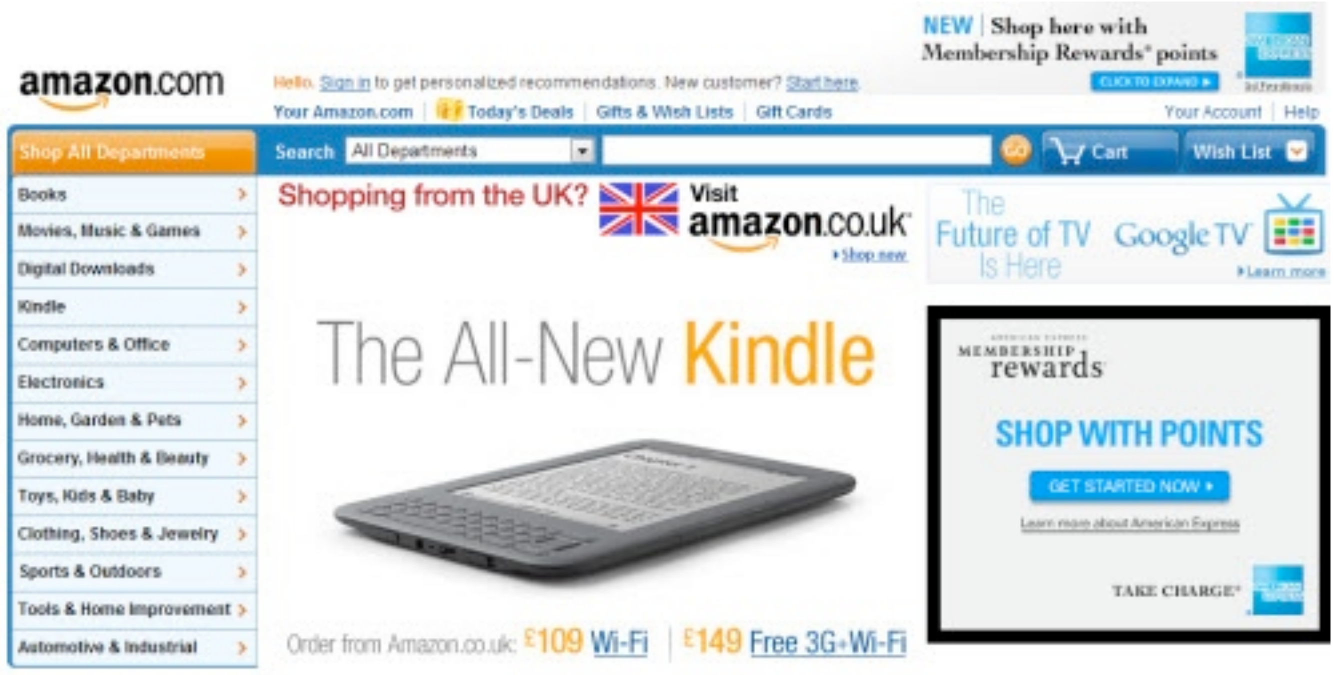


Figure 8: How mobile-friendly websites dealt with desktop PC users trying to access the mobile website



Hello. Sign in to get personalized recommendations. New customer? Start here.

Figure 9: While Amazon's .mobi website stays mobile friendly, the primary m.amazon.com redirects PC users to the original website.

While the number of websites that employed a script to detect and redirect visitors to a mobile experience wasn't as high as one might have expected (as many believe that the mobile website should come first to boost loading times), an interesting trend occurred where all but one website (answers.com) employing a mobile version of the experience (on the WWW subdomain) forced the redirection of PC users back to the desktop view without allowing entry to access the mobile edition (if they wished to).

MOBILE TLD USAGE

This second test aimed to determine the use of TLD conventions in mobile design. In order to keep the scope of the results as strict as possible, only subdomains (such as m.) and the .mobi TLD were considered (along with dedicated mobile websites). As such, directory paths on the WWW domain such as "/mobile/" were not considered. The possible implications of this result could showcase the popularity for the .mobi TLD, and trends which may exist in the use of subdomains on mobile websites.

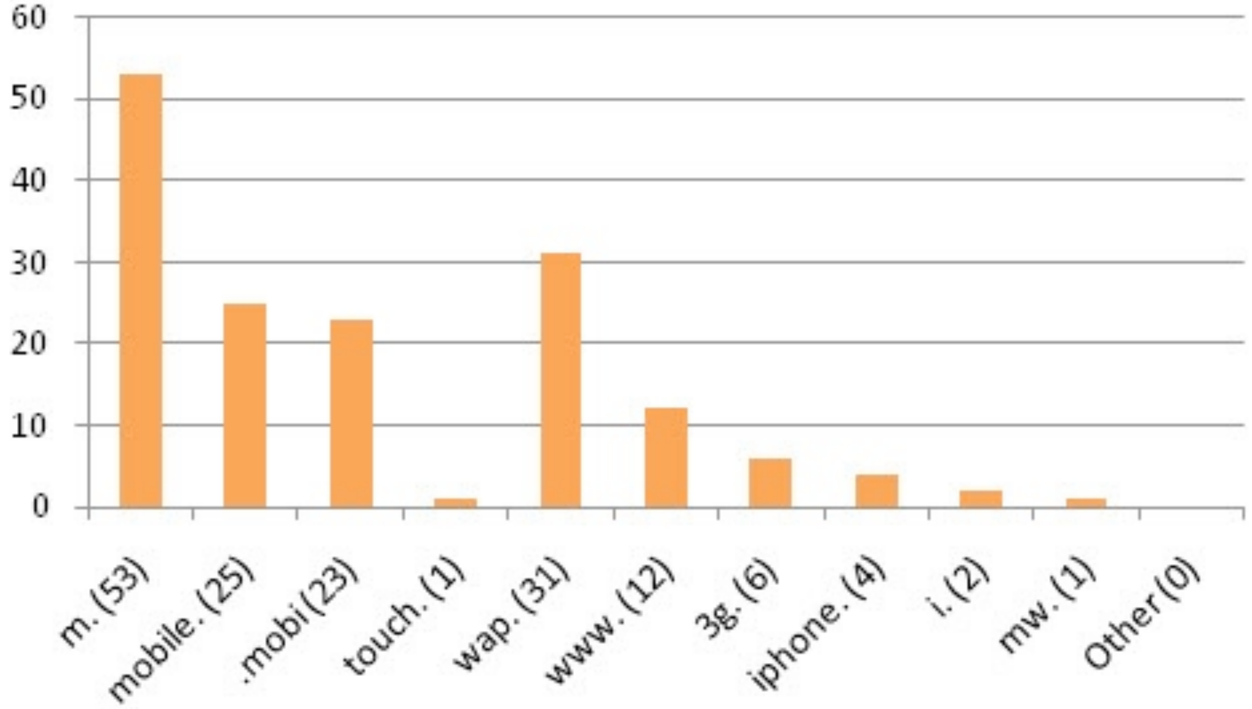


Figure 10: The distribution of mobile TLD subdomains, including the level of extension popularity

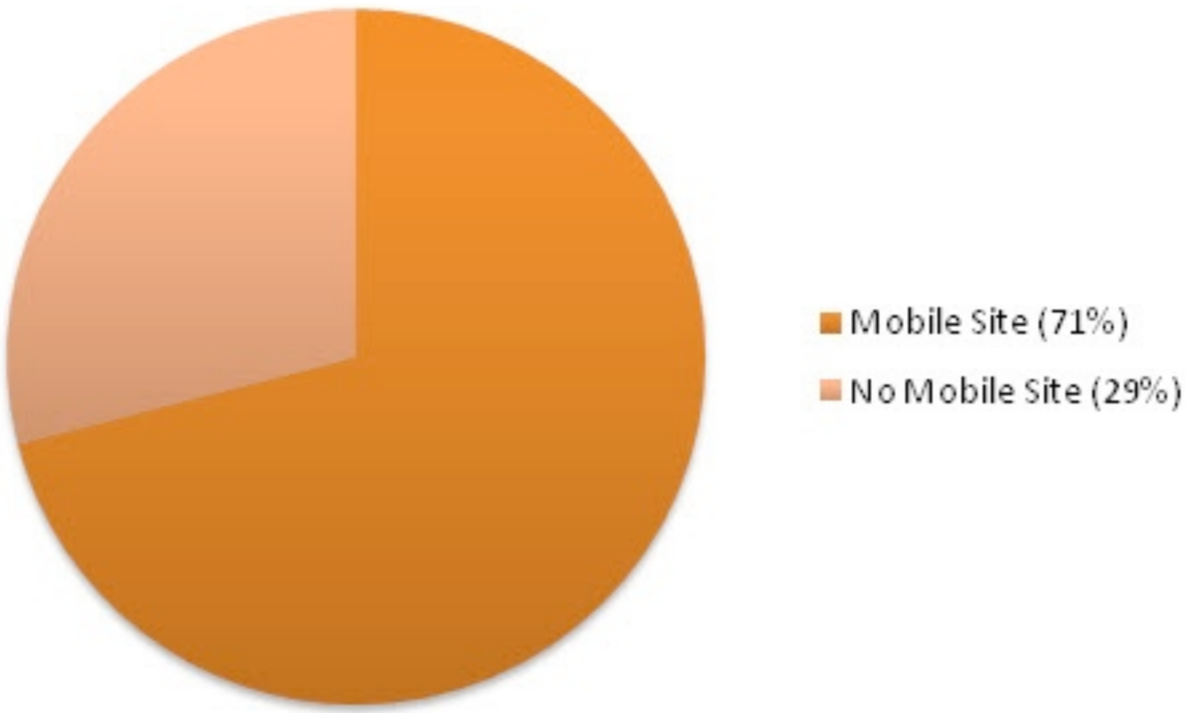


Figure 11: The number of websites which offered no mobile-optimized experience in any form

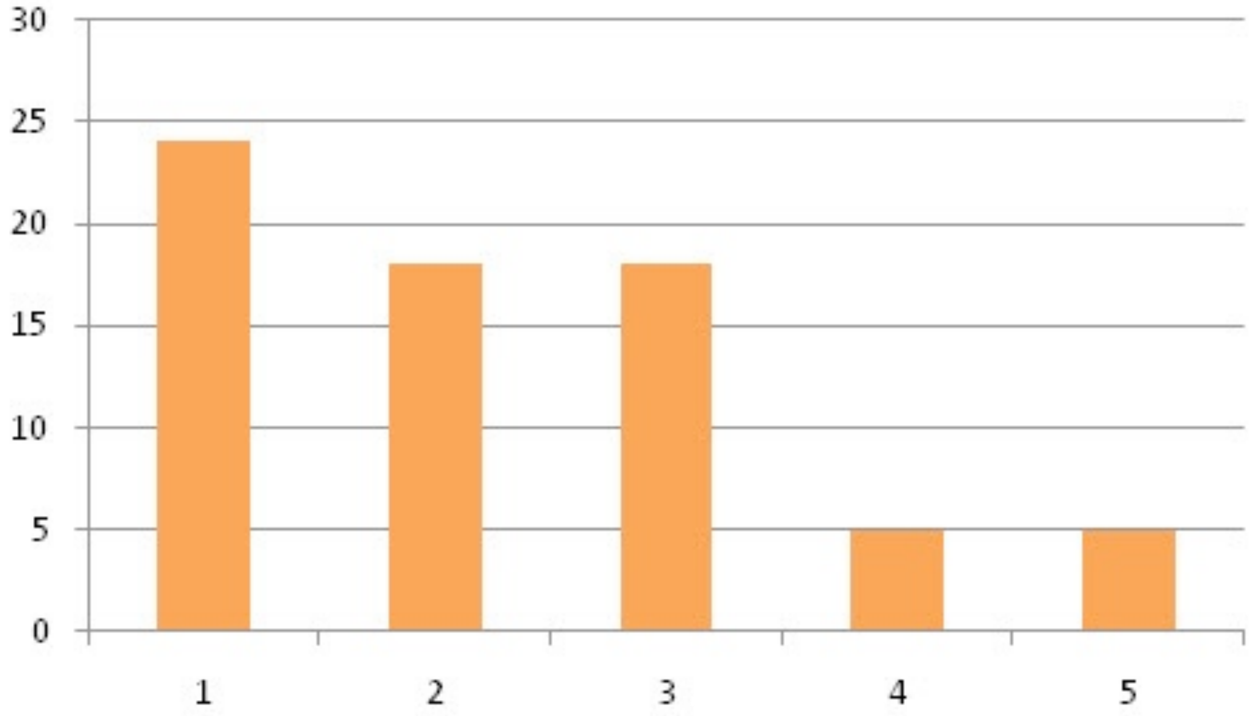


Figure 12: A graph showcasing the number of subdomains supported per website (on average)

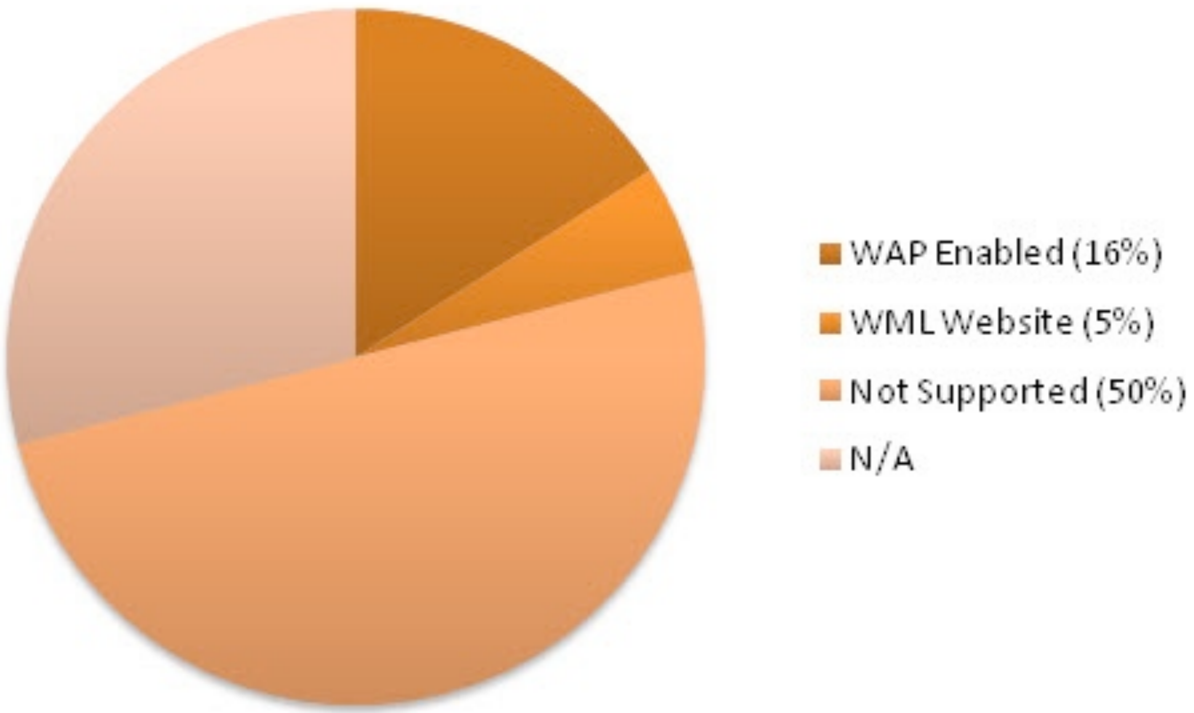


Figure 13: The levels of which WAP- or WML-enabled devices are supported or offered access

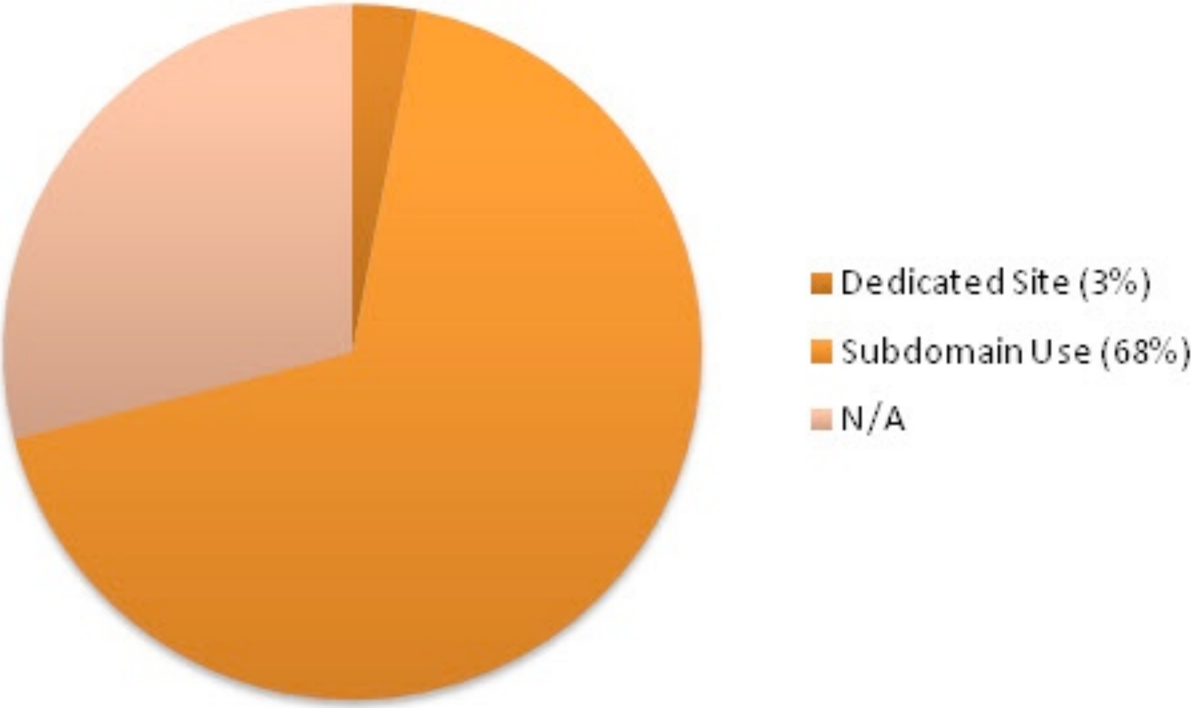
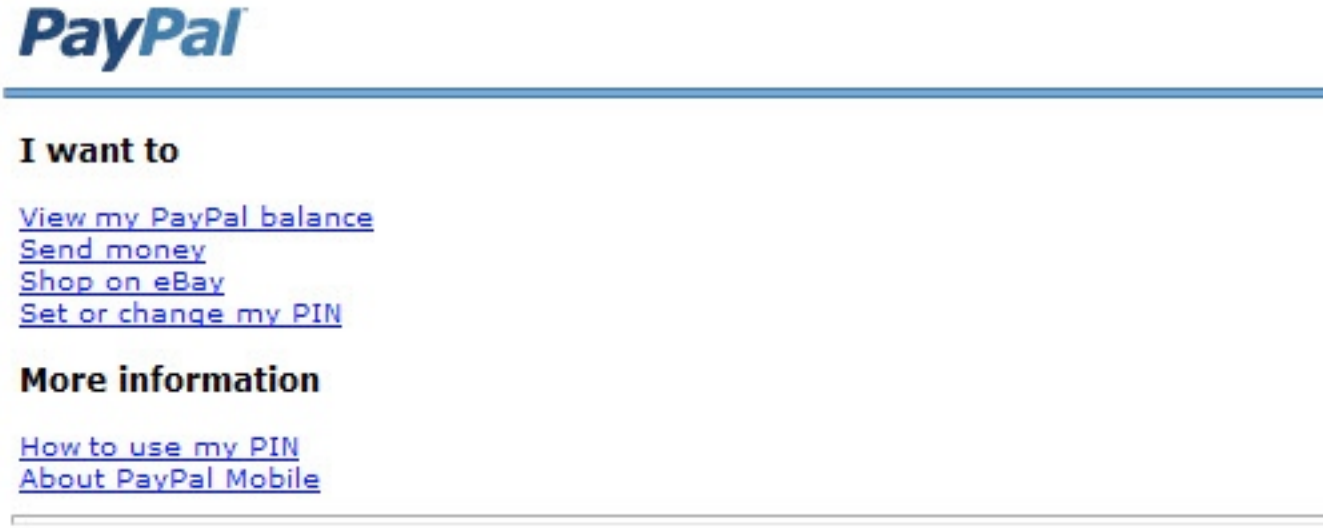


Figure 14: The percentage of websites which redirect the user to a separate mobile TLD



I want to

View my PayPal balance
Send money
Shop on eBay
Set or change my PIN

More information

How to use my PIN
About PayPal Mobile

Figure 15: PayPal makes use of a mobile website that uses the m., mobile and .mobi TLD domains

Interestingly, while many sites offered some form of mobile experience and some redirected the visitor, the use of .mobi extensions was much lower than expected. In addition, an unusual trend made itself apparent. Unlike other nations' mobile websites, Asian ones trended toward using "3G" in preference to "m" or "mobile" (as used elsewhere). It's worth mentioning in addition that only Apple devices were lucky enough to receive a dedicated website using the "touch" or "i" subdomain.

MOBILE PHONE APPS

While this test was not so much about the type of code rendered in browsers, it seemed prudent to determine how many websites in the surveyed list provided a mobile application for devices such as iDevices, Android, or Blackberry. The results of this test simply look for the presence of a mobile app; the platform itself is not taken into account. With so many apps having Web connectivity, the results of this test really push the barriers to access in finding how mobile-friendly a website is.

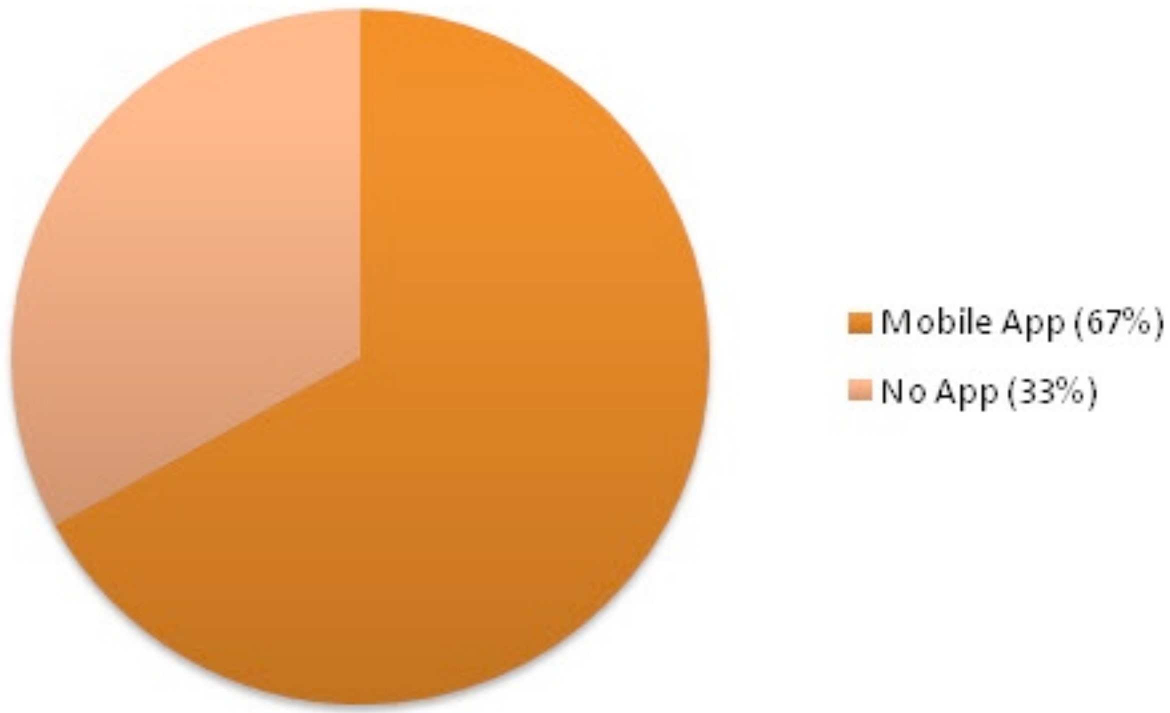


Figure 16: The number of websites that had mobile phone-specific apps (for any platform)

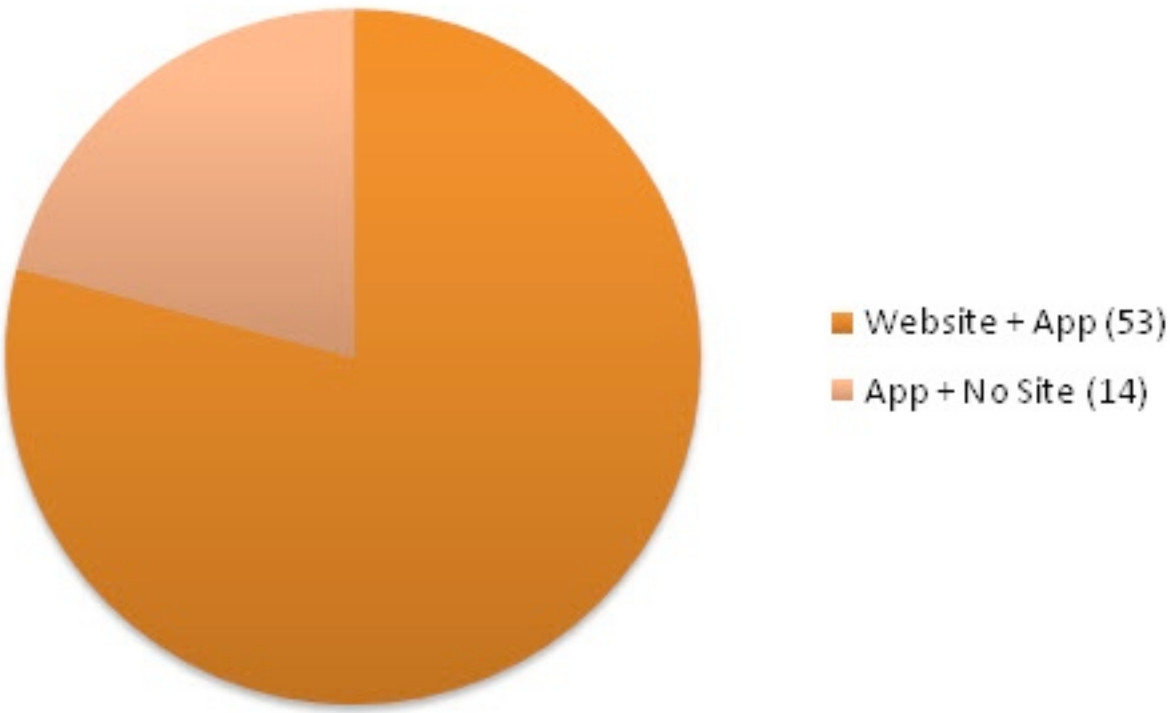
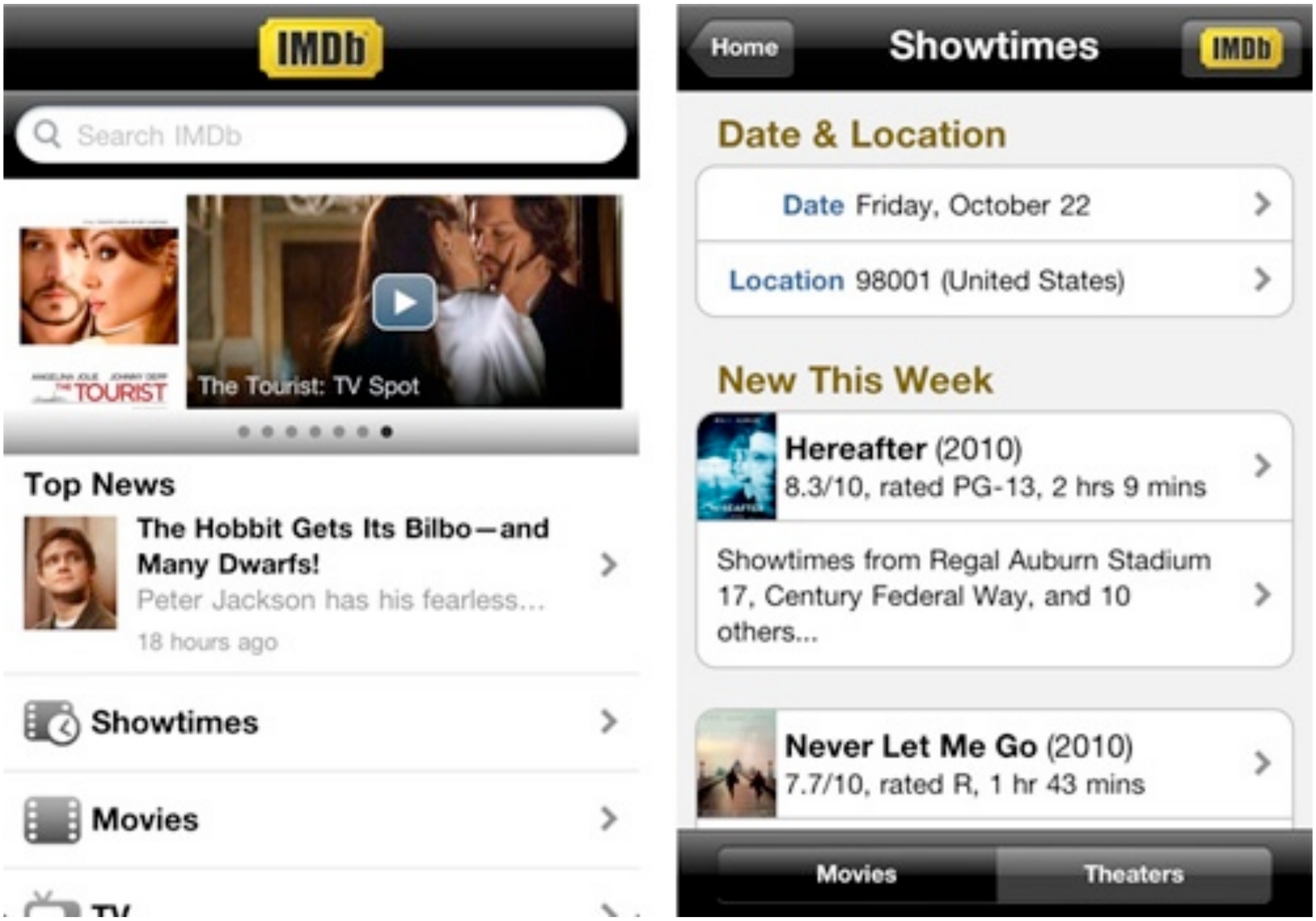


Figure 17: The percentage of websites with a mobile app, but without a mobile-friendly website



Top News

The Hobbit Gets Its Bilbo—and Many Dwarfs!
Peter Jackson has his fearless...
18 hours ago

Date & Location

Date Friday, October 22

Location 98001 (United States)

New This Week

Hereafter (2010)
8.3/10, rated PG-13, 2 hrs 9 mins

Showtimes from Regal Auburn Stadium 17, Century Federal Way, and 10 others...

Never Let Me Go (2010)
7.7/10, rated R, 1 hr 43 mins

Figure 18: The Internet Movie Database has a mobile website and an iPhone app to further improve the mobile experience

While it’s not surprising that many of the top 100 websites had a mobile app, due to the popularity and the widespread use of their services, what did become apparent and rather interesting is that some of the listed websites which did not have a mobile-friendly website (in any form) still had an app. This particular trend seems to indicate that the transition toward a mobile-friendly set of services, is down to the heightened demand for apps (which unlike mobile websites, have no fallback to function on).

AVERAGE LOADING TIME

In the next test, we felt it was important to measure the loading times of each website to see how mobile experiences account for the bandwidth restraints and temperamental speeds of the average Web user. For the purposes of this study, each of the times were registered over a Wi-Fi connection (not a 3G or Edge stream due to some emulator usage, to ensure balanced results) and were done using an uncached format; therefore, the loading time would be accurate to include any external resources.

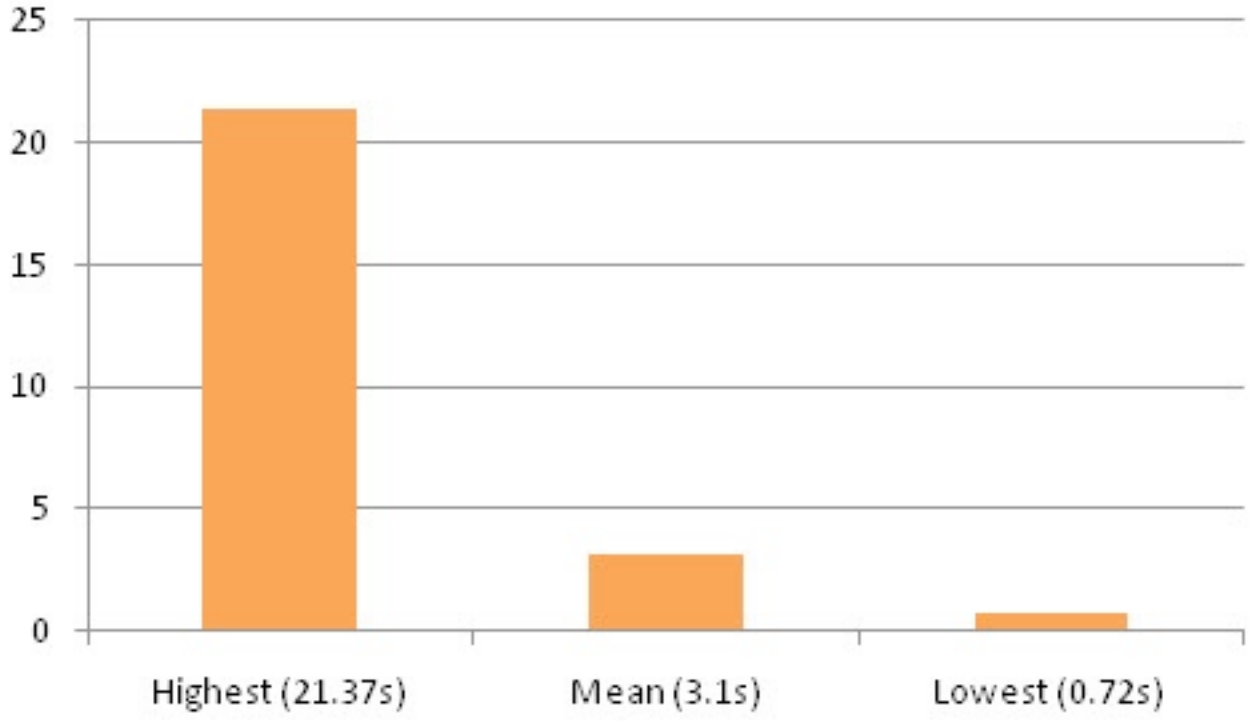


Figure 19: The highest, mean, and lowest loading times as calculated from an uncached "cold boot."

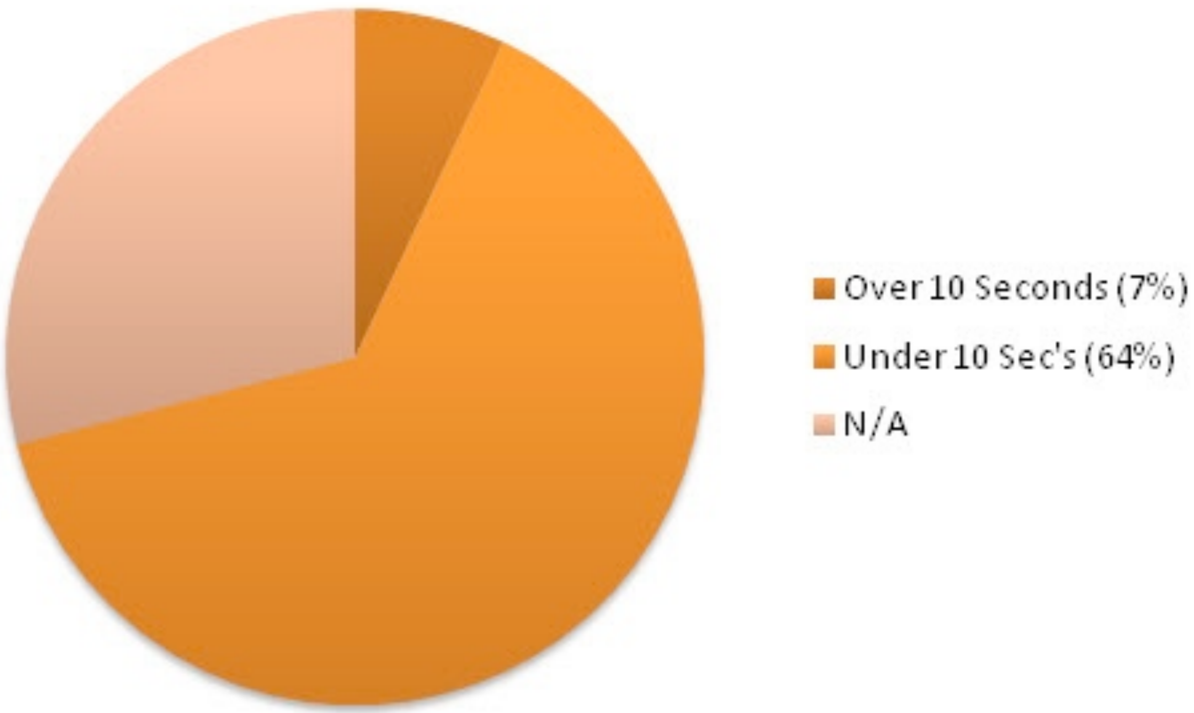


Figure 20: The percentage of websites requiring more than 10 seconds loading time on a Wi-Fi connection

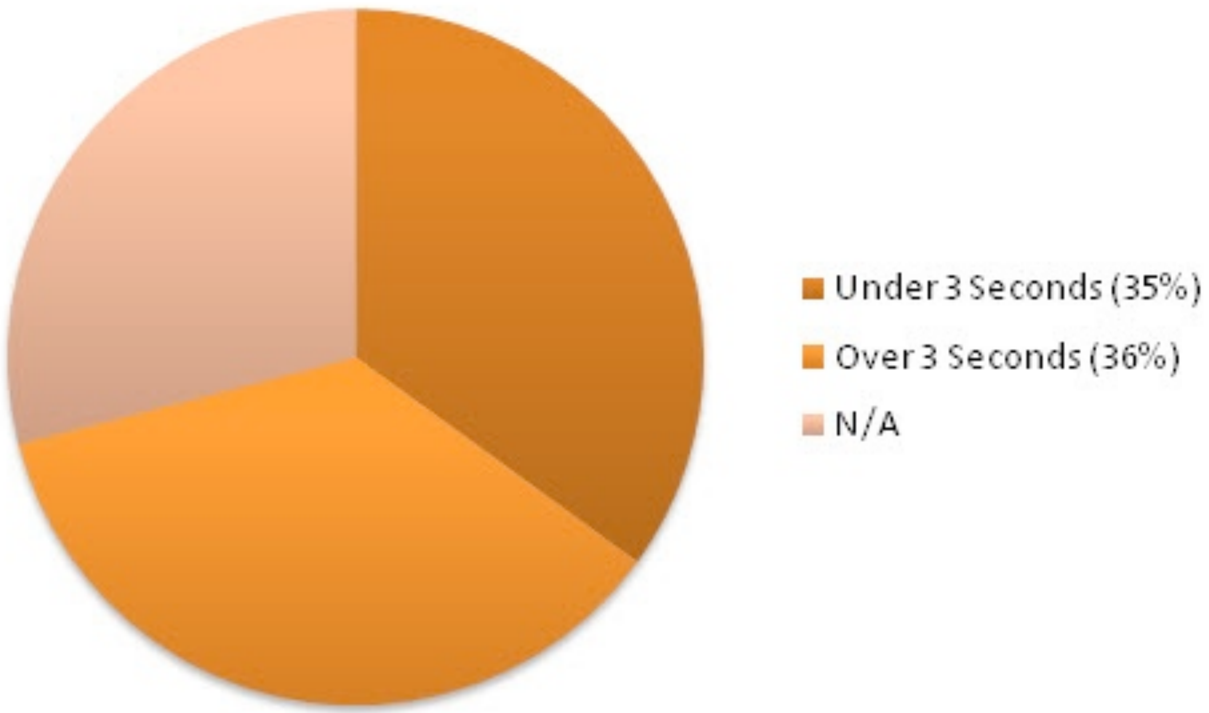
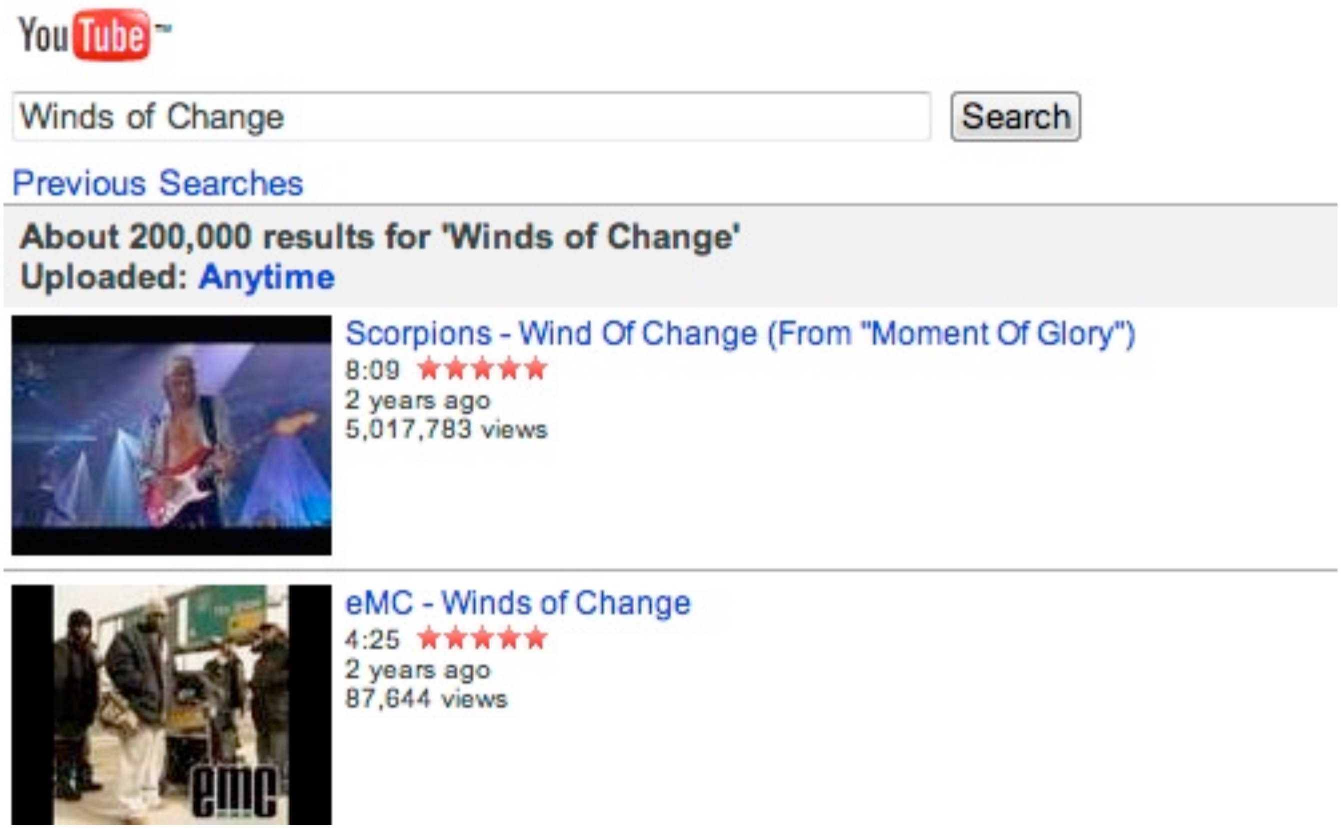


Figure 21: The percentage of websites requiring less than 3 seconds loading time on a Wi-Fi connection


Winds of Change

Search

Previous Searches

About 200,000 results for 'Winds of Change'
Uploaded: **Anytime**

Scorpions - Wind Of Change (From "Moment Of Glory")
8:09 ★★★★★
2 years ago
5,017,783 views

eMC - Winds of Change
4:25 ★★★★★
2 years ago
87,644 views

Figure 22: YouTube had one of the fastest loading times for any mobile-oriented website

While caching will produce substantially quicker loading times, the results were fairly conclusive that the loading time of each website was pretty evenly spread, with only a small number of websites having load times of over 5 seconds (often in such cases it was a result of slow server response, rather than of the physical content being transferred). As visitors obviously don't want to wait for long periods of time to get their data, a speedy and effective loading time has become critical to the design process.

HOME PAGE ASSET SIZE

Along with the time it took to load a page, in equal measure it immediately became obvious that the size of the files and any loaded external resources should be measured. With many mobile Internet plans having capped services and international browsing potentially becoming prohibitively expensive (by the megabyte), it seemed only fair to determine how each website was optimized and whether the amount of uncached data loaded was as small, or big, as a regular desktop-oriented website.

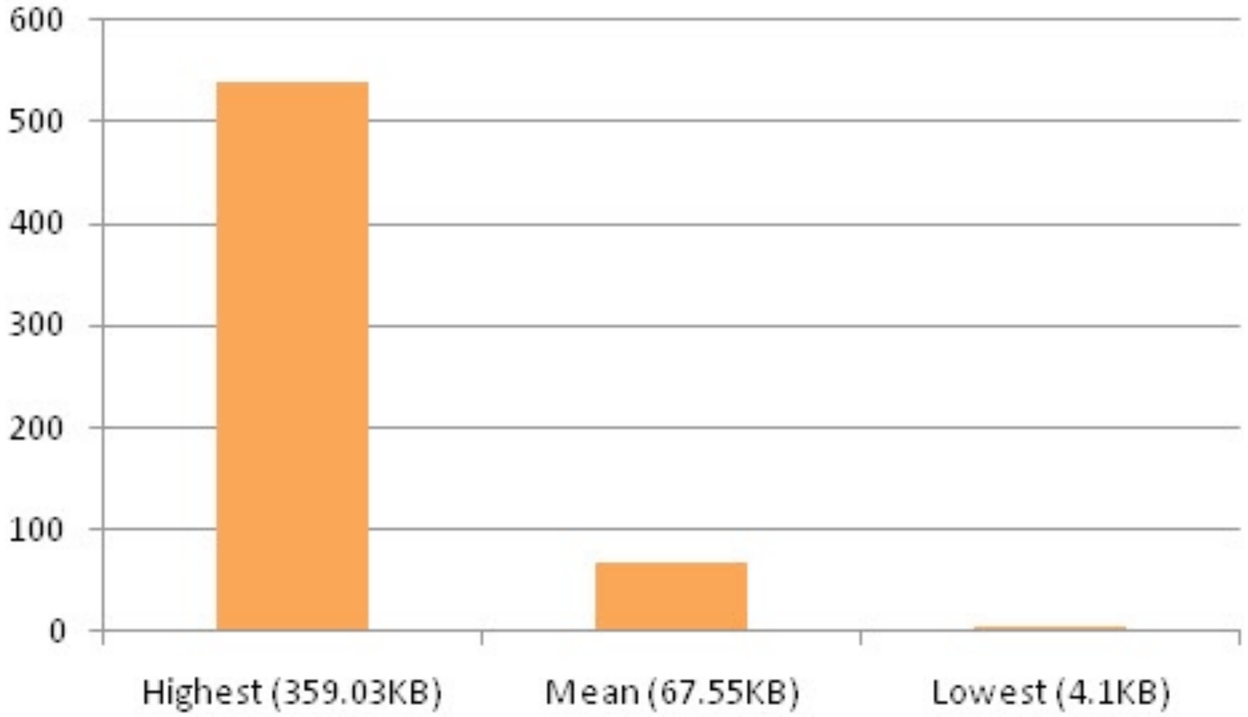


Figure 23: The highest, mean, and lowest file (and asset) size as calculated with caching turned off

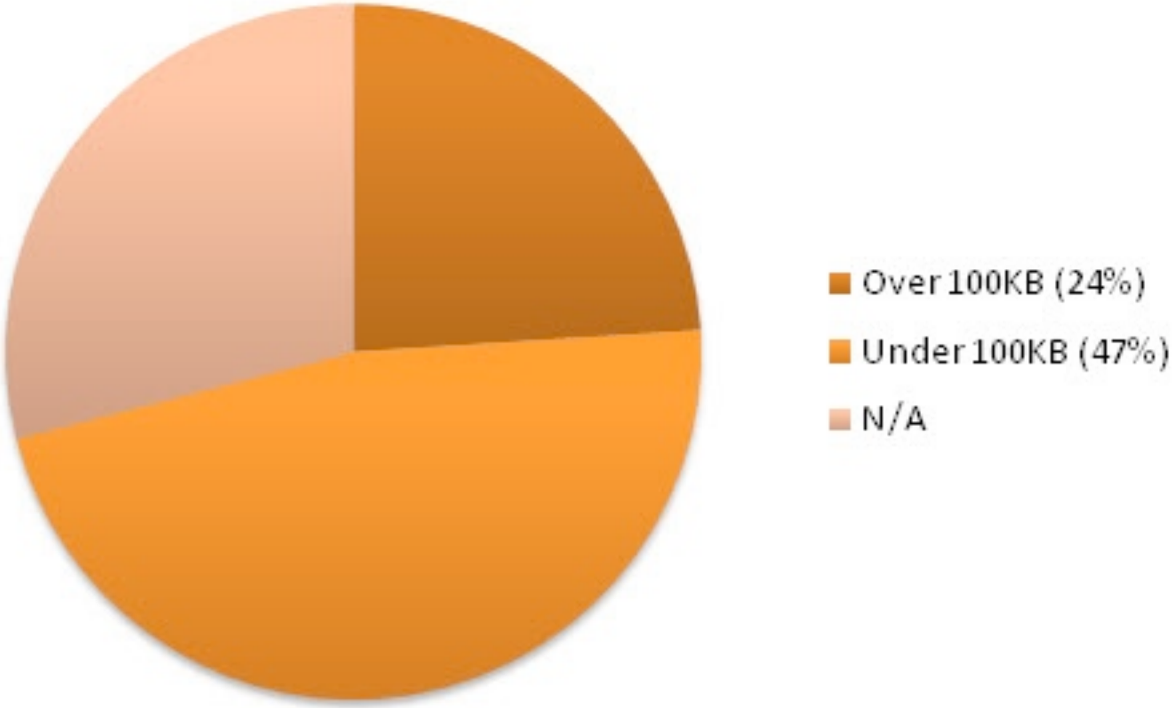


Figure 24: The percentage of websites with a total uncached asset size of more than 100KB

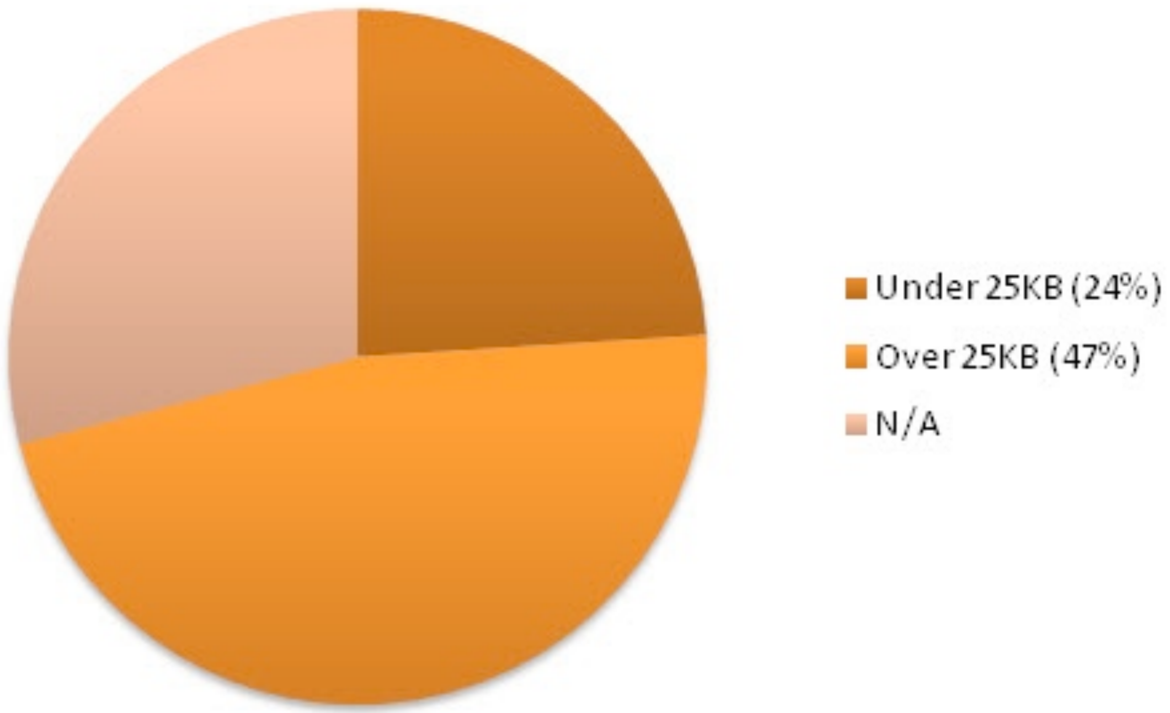
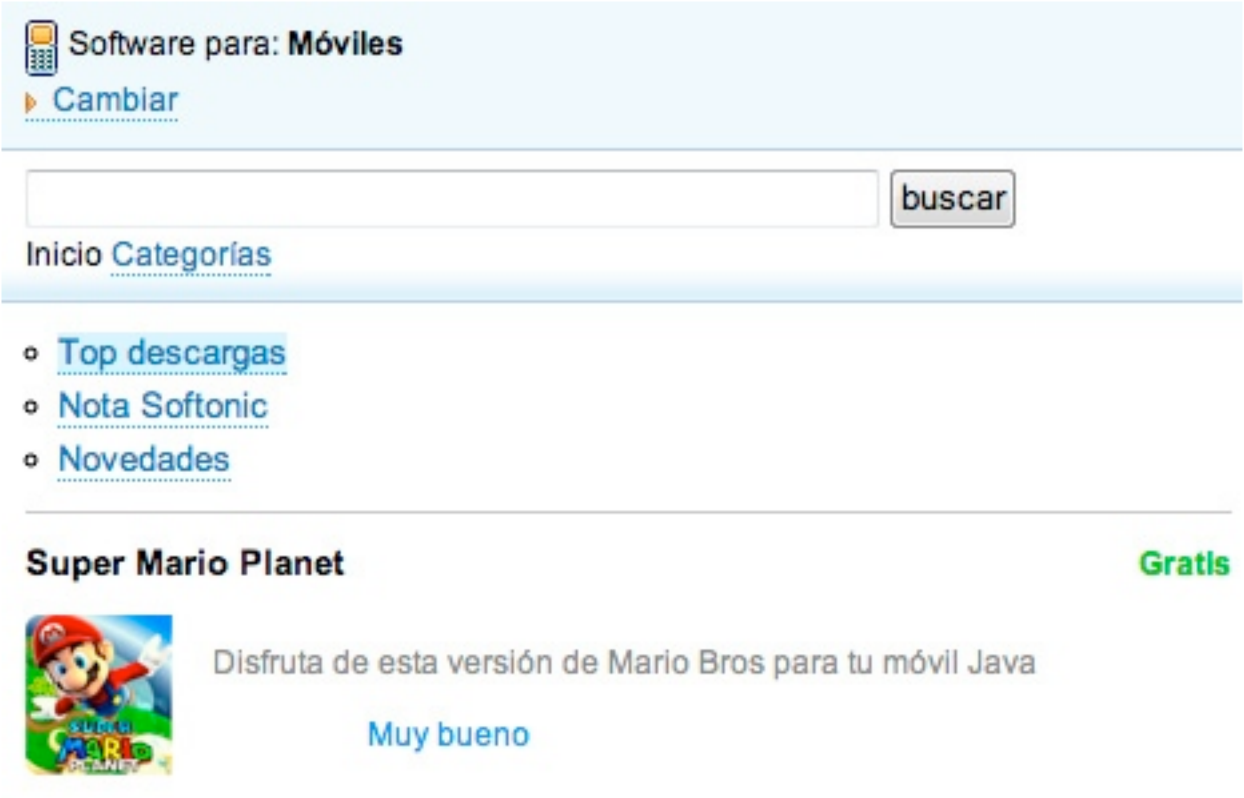


Figure 25: The percentage of websites with a total uncached asset size of less than 25KB



Super Mario Planet

Disfruta de esta versión de Mario Bros para tu móvil Java

Figure 26: Softonic had the largest file size of any mobile site — quite a surprising result!

The results of this test are rather interesting. While many websites (especially WAP-oriented ones for less capable devices) required less than 25KB to function, which seems like an acceptable level–a large number of websites supposedly providing a mobile experience required over 100KB, some even as high as 0.5MB! More interesting is the coincidence that the percentage of websites that have asset sizes of fewer than 25KB, matched (exactly) the percentage requiring over 100KB.

DOCTYPE DECLARATIONS

When producing a website, one of the initial questions we ask ourselves is, what language or version of that language will best meet our contents needs. The debates about HTML vs. XHTML have been endless, and as many mobile variants exist (the mobile profiles for XHTML and WML), it became apparent that statistics related to the types of DTD most often used could be of benefit to readers. This test therefore examined the Doctype of the front page to see if any patterns of usage exist.

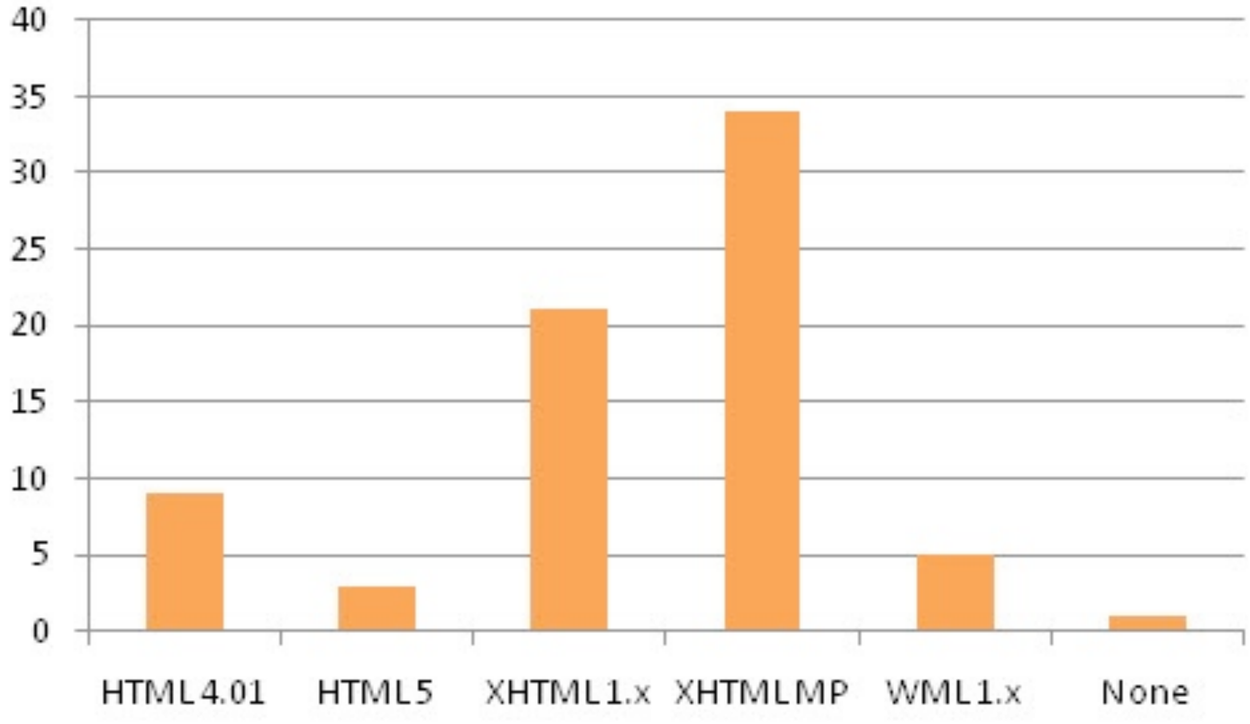


Figure 27: The distribution of mobile website Doctypes, based on language versions and profiles

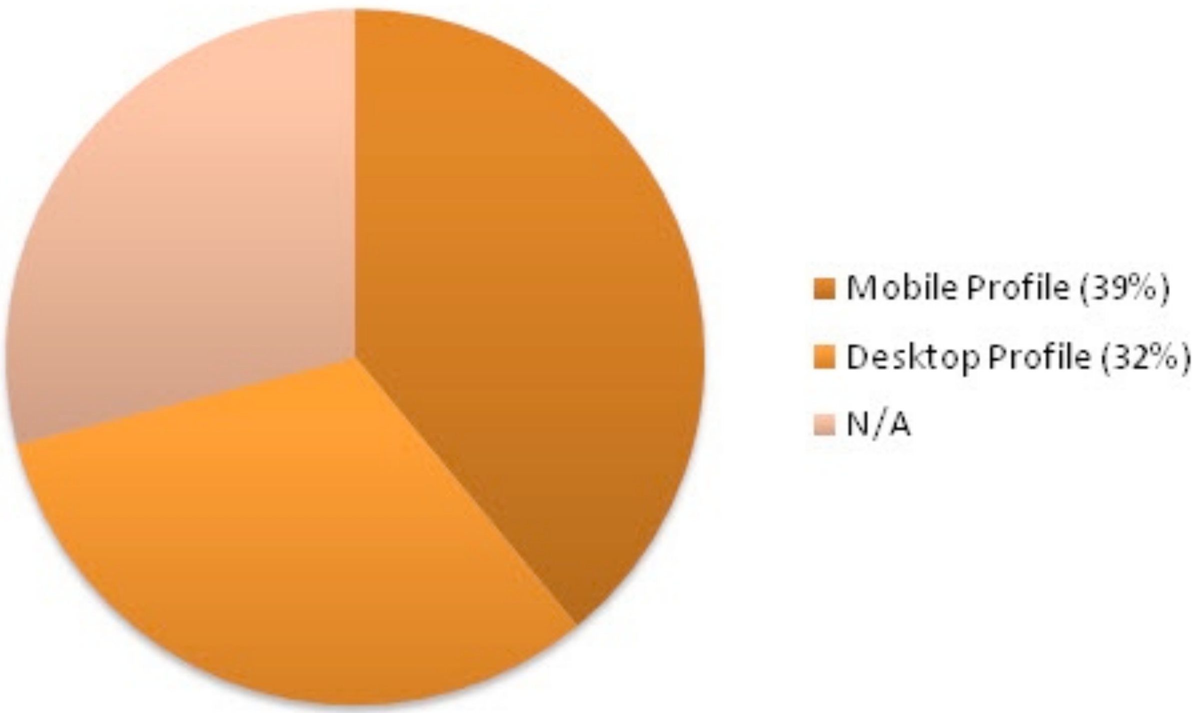


Figure 28: The percentage of websites using a mobile-specific profile in preference to desktop profiles

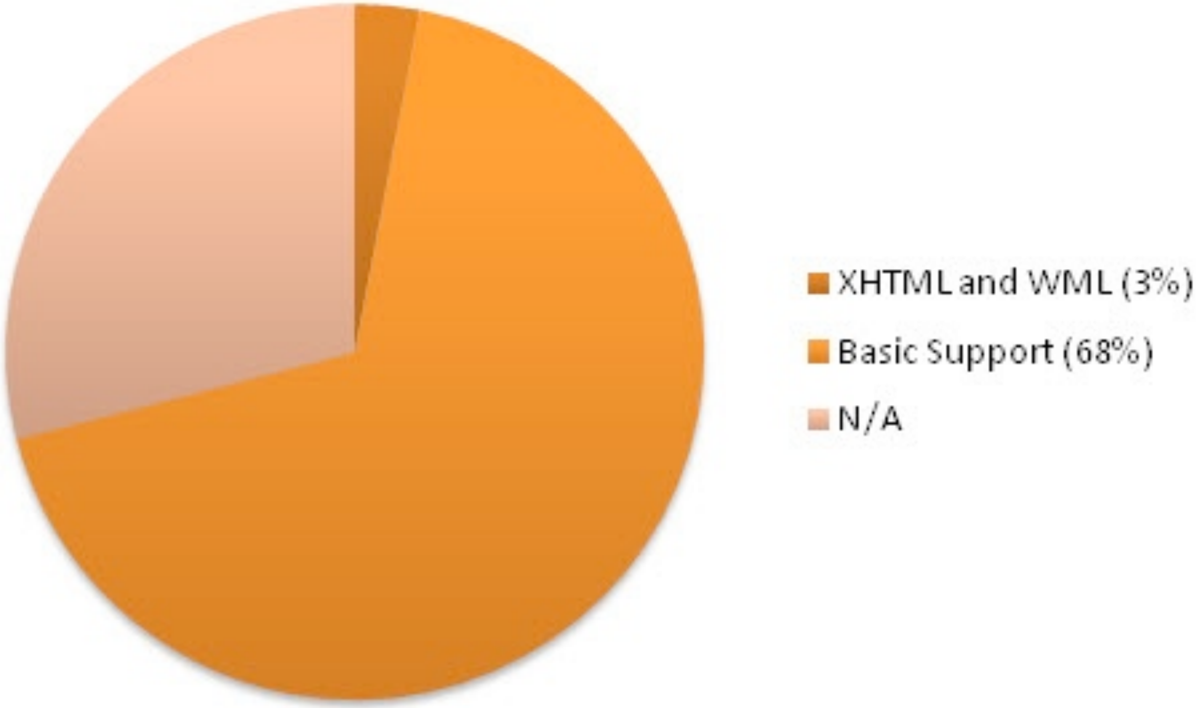
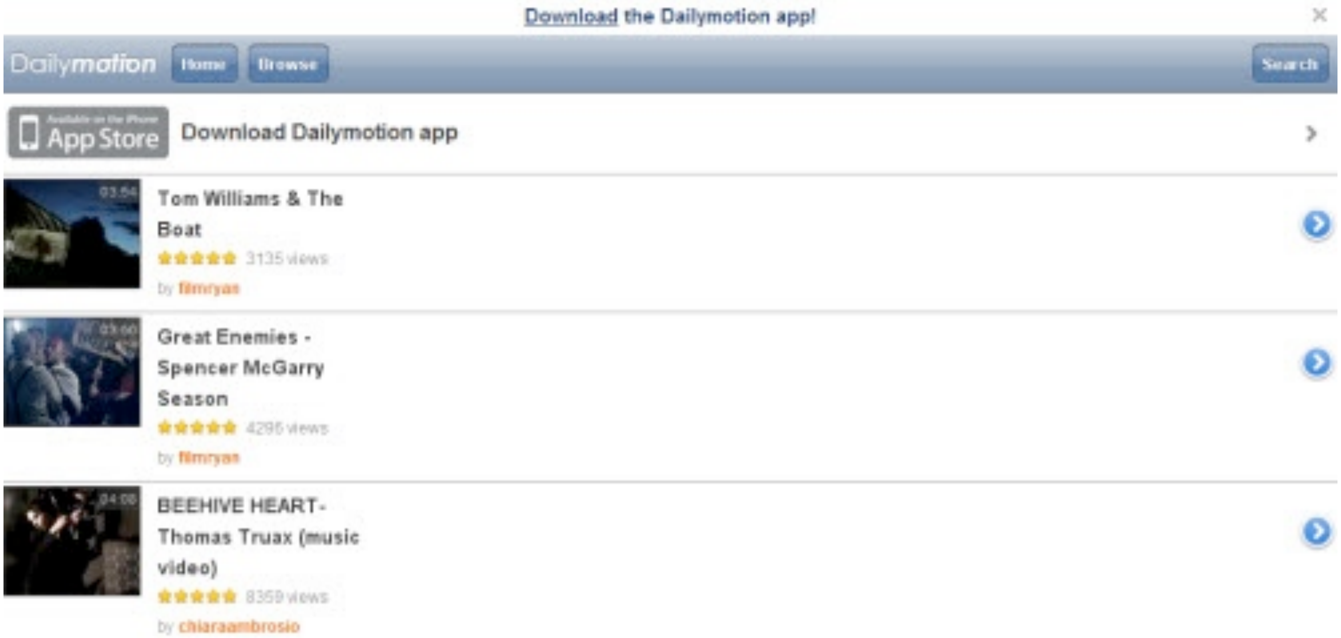


Figure 29: The percentage of websites that support both a WML and full XHTML experience



Download the Dailymotion app!

Download Dailymotion app

Tom Williams & The Boat
★★★★★ 3135 views
by filmryan

Great Enemies - Spencer McGarry Season
★★★★★ 4295 views
by filmryan

BEEHIVE HEART- Thomas Truax (music video)
★★★★★ 8359 views
by chiaraambrosio

Figure 30: Dailymotion's iPhone friendly website was one of the few leveraging HTML5

The results show conclusively that XHTML is currently more popular than HTML (or HTML5). This could be down to HTML's lack of a mobile profile (used in most cases), though it can be noted that many smartphones supported HTML and XHTML. In addition, the spread of mobile vs. desktop profile usage was fairly similar. This questions the need for mobile profiles, if the full specs are supported. In addition, Facebook's iPhone-friendly website was the only one which failed to provide any Doctype at all!

CODE VALIDATION

With the semantic Web and the need for our industry to maintain standards, this test followed an earlier study by Jeffrey Zeldman in which a large number of websites were put through a simple "pass or fail" test against validation. While Zeldman's research focused on the Alexa top 100, this piece of course used a different set of data. While standards aren't the be-all and end-all of design, this test was included to provide additional comparisons as to the stage conformity to semantics were at.

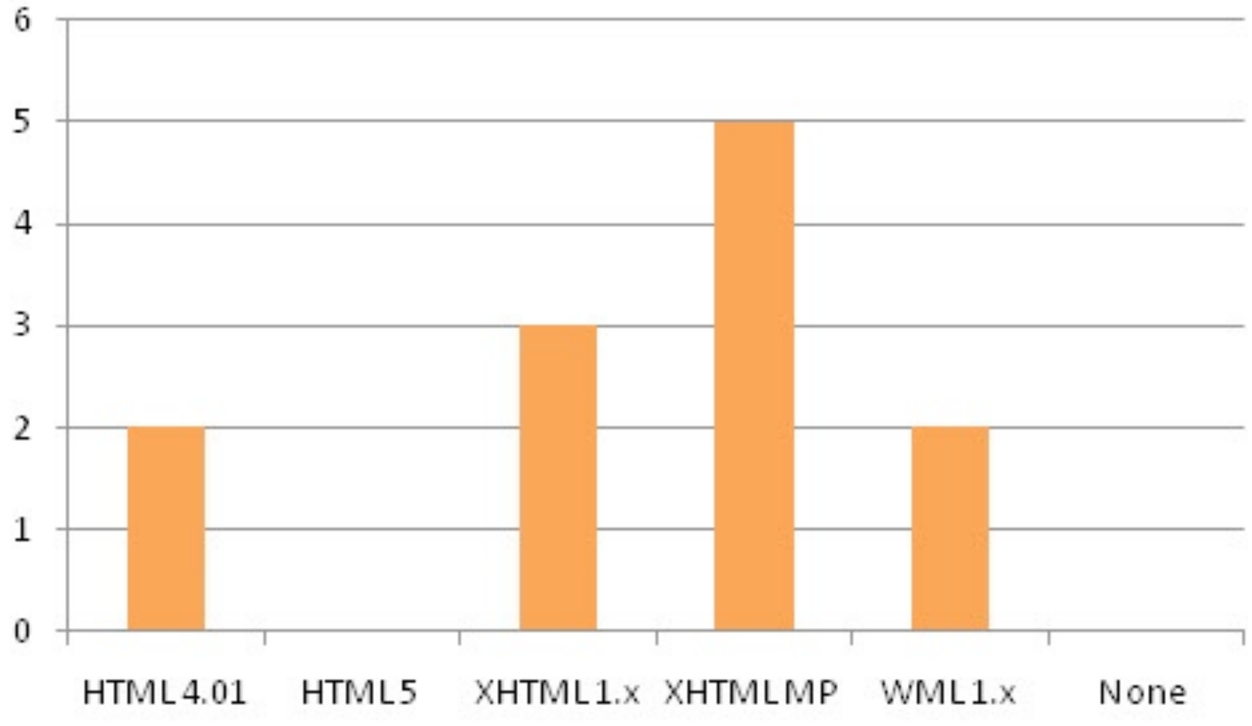


Figure 31: Distribution of website validation (in the case of CSS, proprietary properties were ignored)

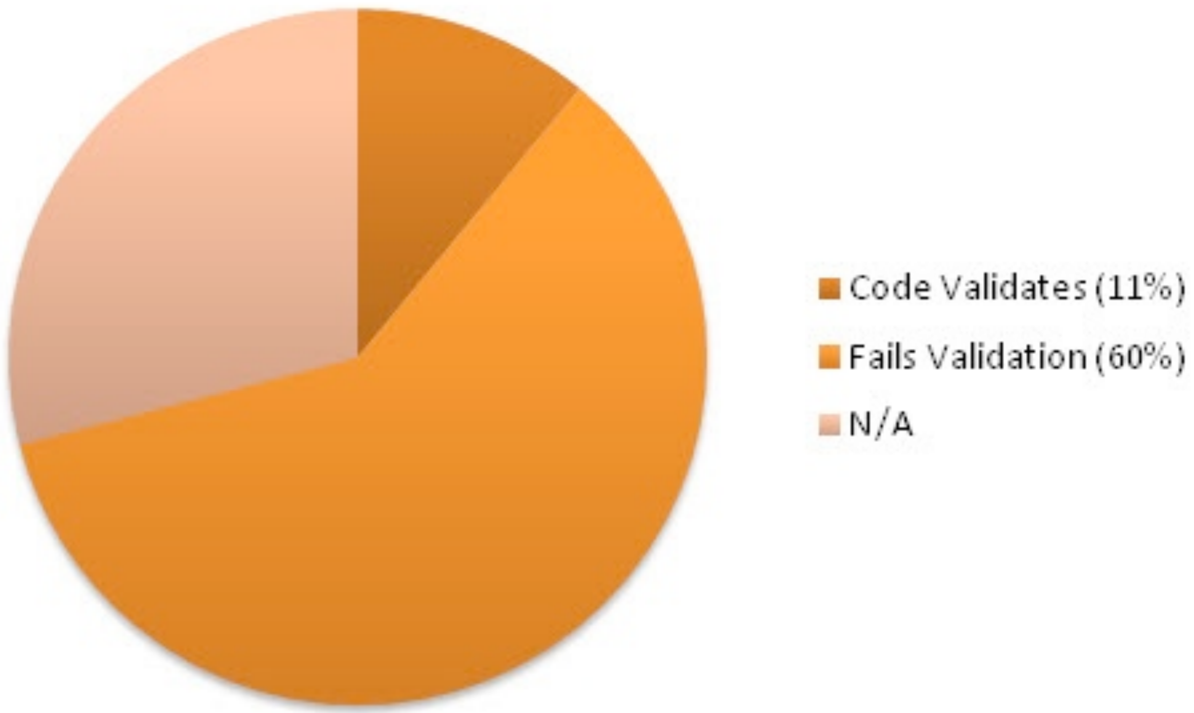
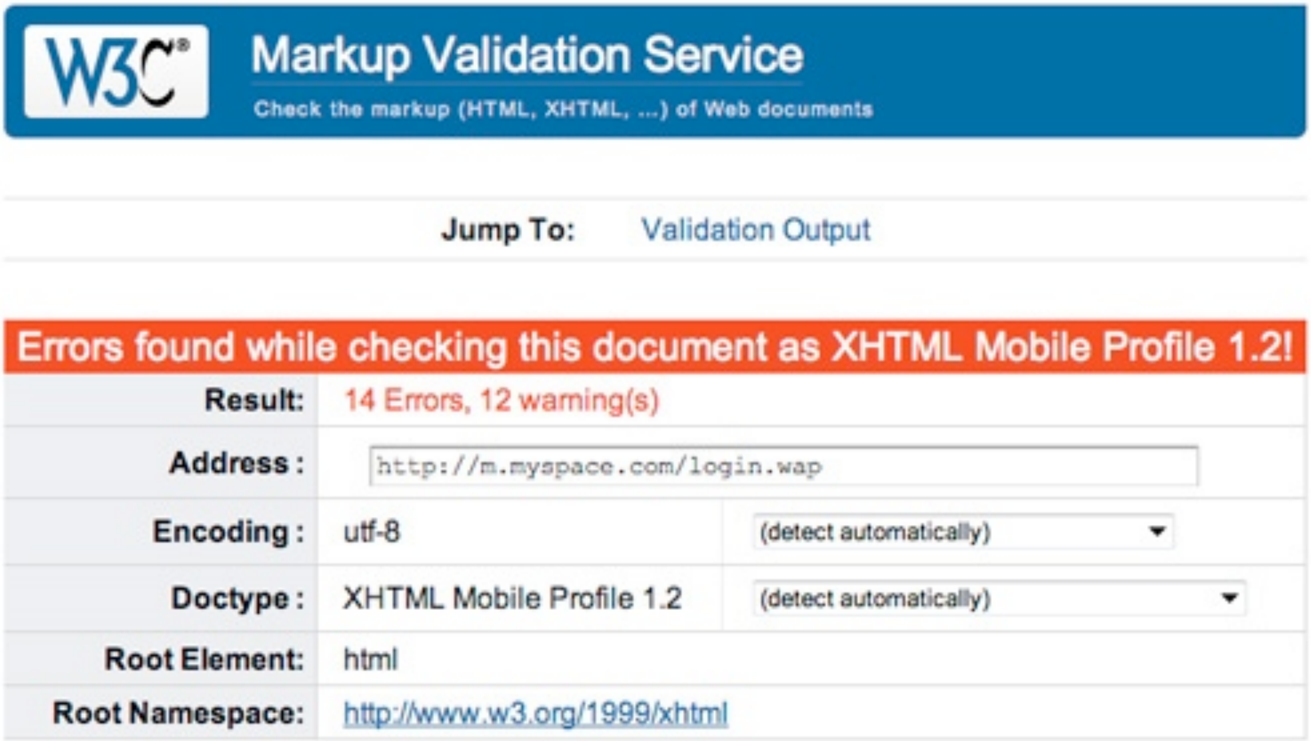


Figure 32: The percentage of top 100 websites with a front page that validates against the DTD chosen



Markup Validation Service

Check the markup (HTML, XHTML, ...) of Web documents

Errors found while checking this document as XHTML Mobile Profile 1.2!

Result:	14 Errors, 12 warning(s)	
Address :	http://m.myspace.com/login.wap	
Encoding :	utf-8	(detect automatically)
Doctype :	XHTML Mobile Profile 1.2	(detect automatically)
Root Element:	html	
Root Namespace:	http://www.w3.org/1999/xhtml	

Figure 33: The MySpace website wasn't unusual, in that its source code didn't validate.

Because this checkpoint focused only on the front page of mobile-friendly websites, the results are not as complete as they could be. They do instead provide a good indication as to the care being given to mobile experiences. Unfortunately, the results seem to correlate with Zeldman's research in that the overwhelming majority failed to meet the standards for the DTD they chose to conform too. This seems to reinforce the fact that design is more important to those designers, than optimizations or quality.

CODE SEPARATION

The next test that was carried out links quite heavily into a few of those that were previously carried out. The separation of structure, style, and behavior has been shown to have benefits in reducing file sizes (due to avoiding repeat coding, and to cache advantages). It therefore seemed right to see not only if websites did separate their style or behavior, but also if they took advantage of CSS3 or jQuery in the mobile design to provide a more dynamic behavior within the website layout.

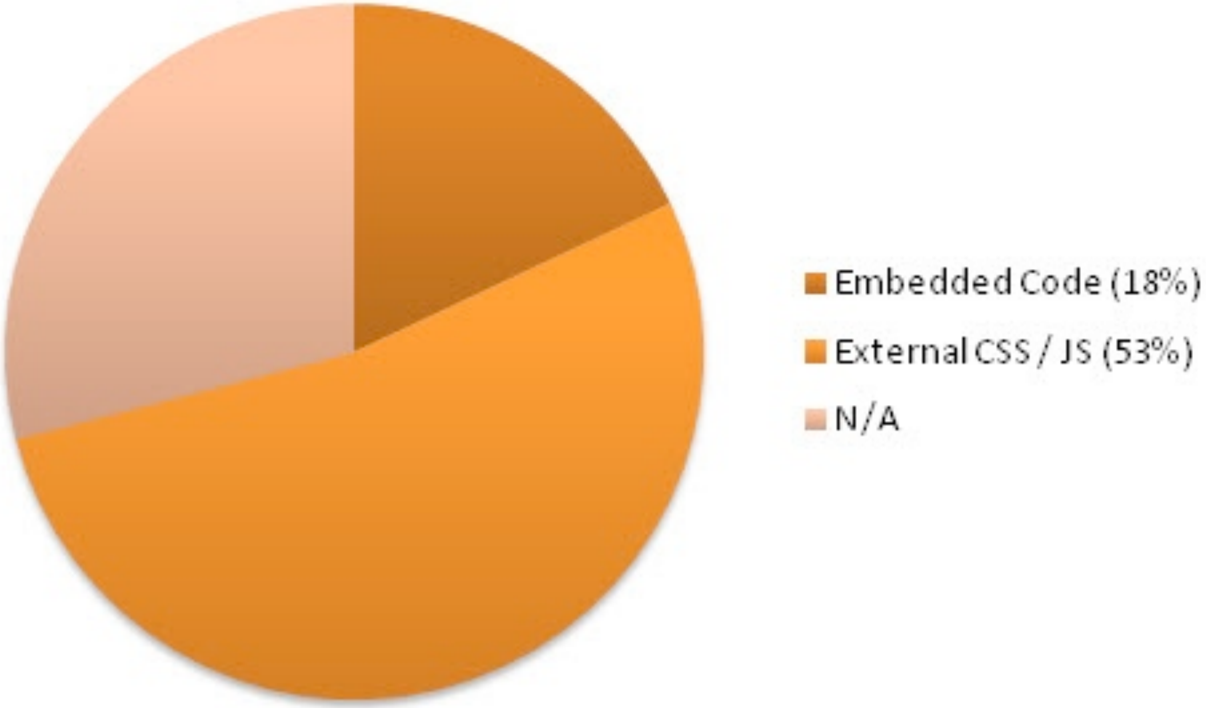


Figure 34: The percentage of websites using embedded inline style rather than external files

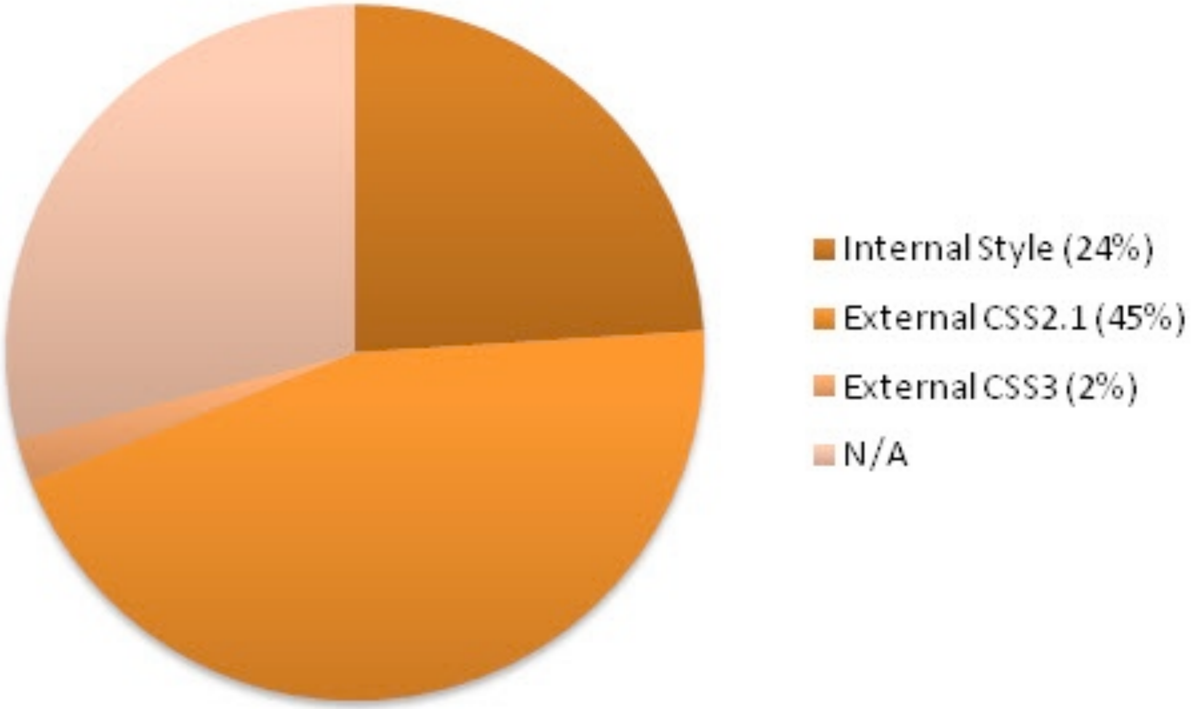


Figure 35: The percentage of websites using external CSS2.1 or CSS3 within their designs

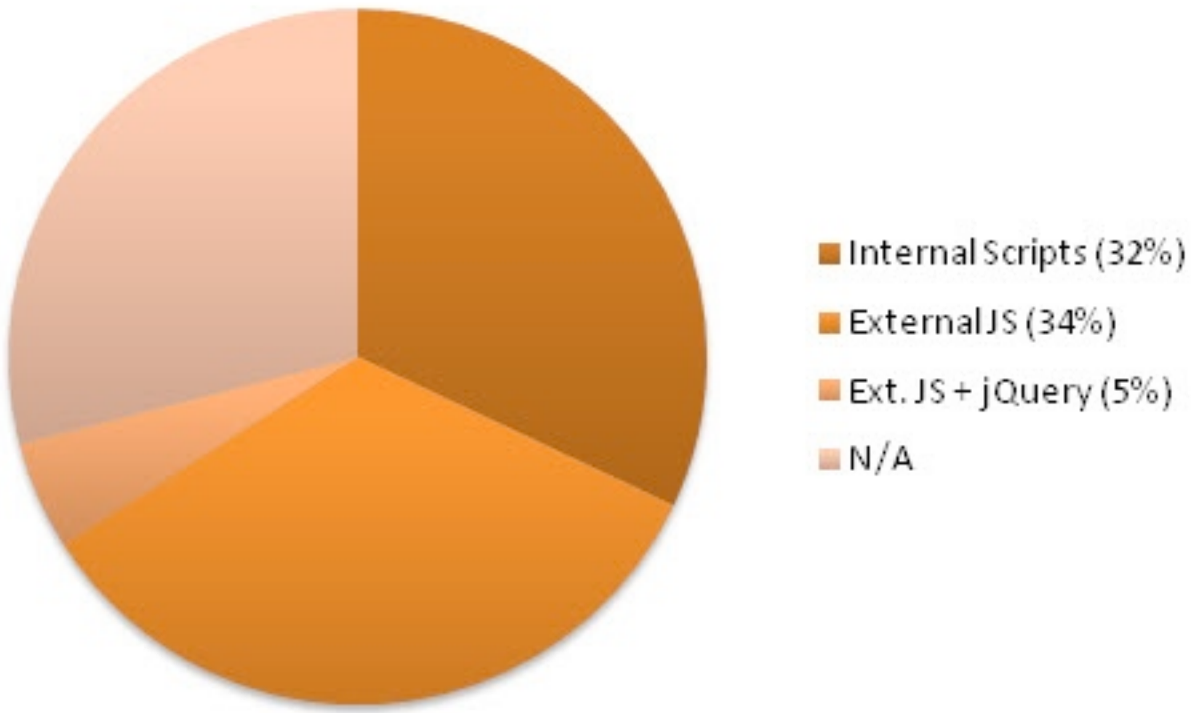
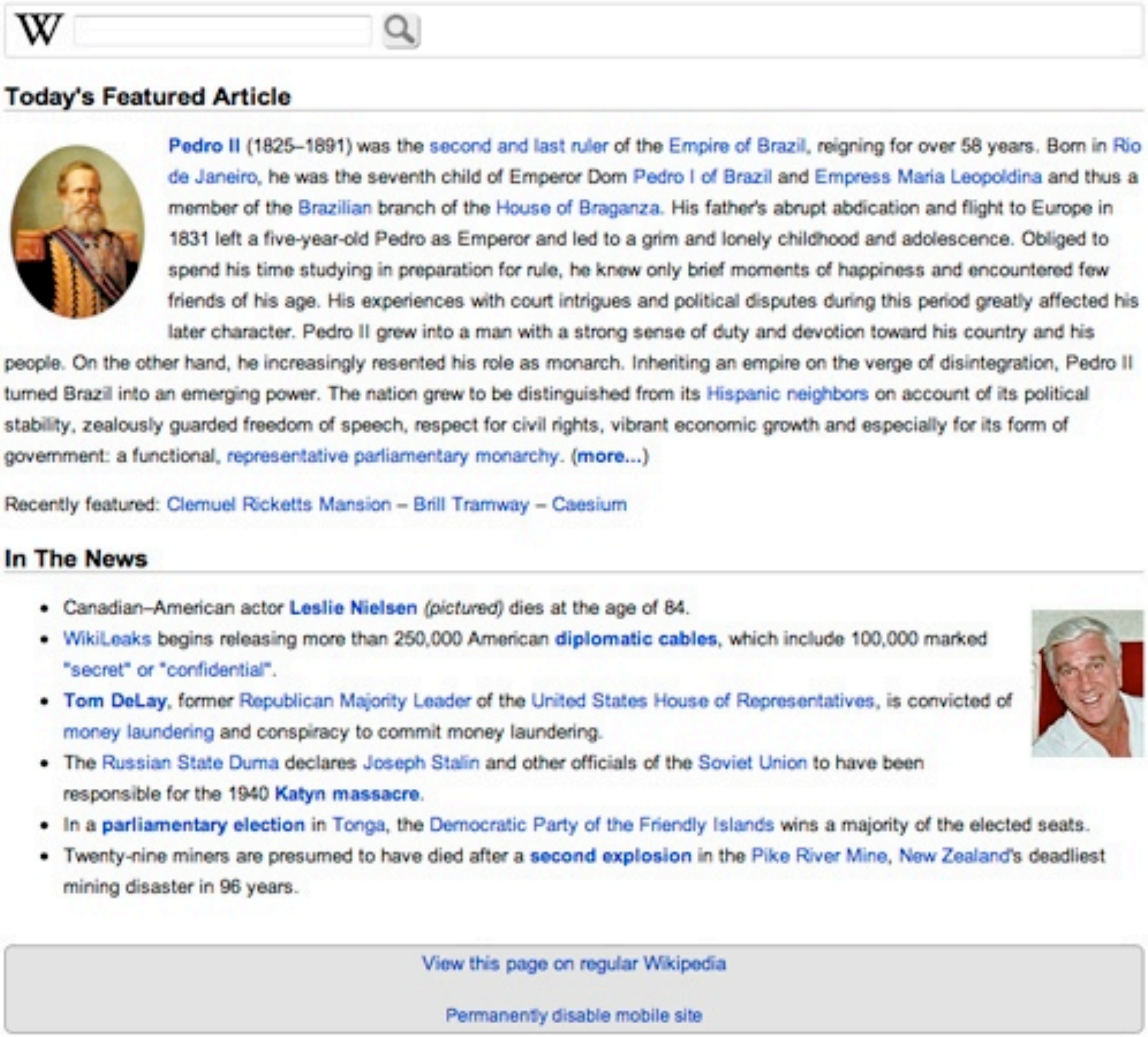


Figure 36: The percentage of websites using external JavaScript or jQuery within their designs



Today's Featured Article

Pedro II (1825–1891) was the second and last ruler of the Empire of Brazil, reigning for over 58 years. Born in Rio de Janeiro, he was the seventh child of Emperor Dom Pedro I of Brazil and Empress Maria Leopoldina and thus a member of the Brazilian branch of the House of Braganza. His father's abrupt abdication and flight to Europe in 1831 left a five-year-old Pedro as Emperor and led to a grim and lonely childhood and adolescence. Obliged to spend his time studying in preparation for rule, he knew only brief moments of happiness and encountered few friends of his age. His experiences with court intrigues and political disputes during this period greatly affected his later character. Pedro II grew into a man with a strong sense of duty and devotion toward his country and his people. On the other hand, he increasingly resented his role as monarch. Inheriting an empire on the verge of disintegration, Pedro II turned Brazil into an emerging power. The nation grew to be distinguished from its Hispanic neighbors on account of its political stability, zealously guarded freedom of speech, respect for civil rights, vibrant economic growth and especially for its form of government: a functional, representative parliamentary monarchy. (**more...**)

Recently featured: Clemuel Ricketts Mansion – Brill Tramway – Caesium

In The News

- Canadian–American actor **Leslie Nielsen** *(pictured)* dies at the age of 84.
- WikiLeaks begins releasing more than 250,000 American **diplomatic cables**, which include 100,000 marked "secret" or "confidential".
- **Tom DeLay**, former Republican Majority Leader of the United States House of Representatives, is convicted of money laundering and conspiracy to commit money laundering.
- The Russian State Duma declares Joseph Stalin and other officials of the Soviet Union to have been responsible for the 1940 **Katyn massacre**.
- In a **parliamentary election** in Tonga, the Democratic Party of the Friendly Islands wins a majority of the elected seats.
- Twenty-nine miners are presumed to have died after a **second explosion** in the Pike River Mine, New Zealand's deadliest mining disaster in 96 years.

View this page on regular Wikipedia
Permanently disable mobile site

Figure 37: Wikipedia makes use of the jQuery library within its mobile-orientated website.

While you would have thought that most websites would immediately break the style and behavior from their mobile pages in order to improve performance, a moderate number of the websites did have all of their code directly tied into the page. In addition, only a rare number of those mobile websites took advantage of the jQuery framework, and an equally small number made use of CSS3 media queries to dynamically scale the layout. The numbers for CSS3 usage were, predictably, similar to HTML5 usage.

FONT FAMILY TYPES

Typography is an essential element of the Web, and of how information is visualized. This particular test was created to not only see which Websafe typefaces are being implemented on the Web, but to see what font families are being used. In addition, any websites which rely on the default typefaces by not providing a font stack (or which have a stack with multiple typefaces), would be noted. The elements for which this test was created were not only based on headings, but on all manner of content in the page.

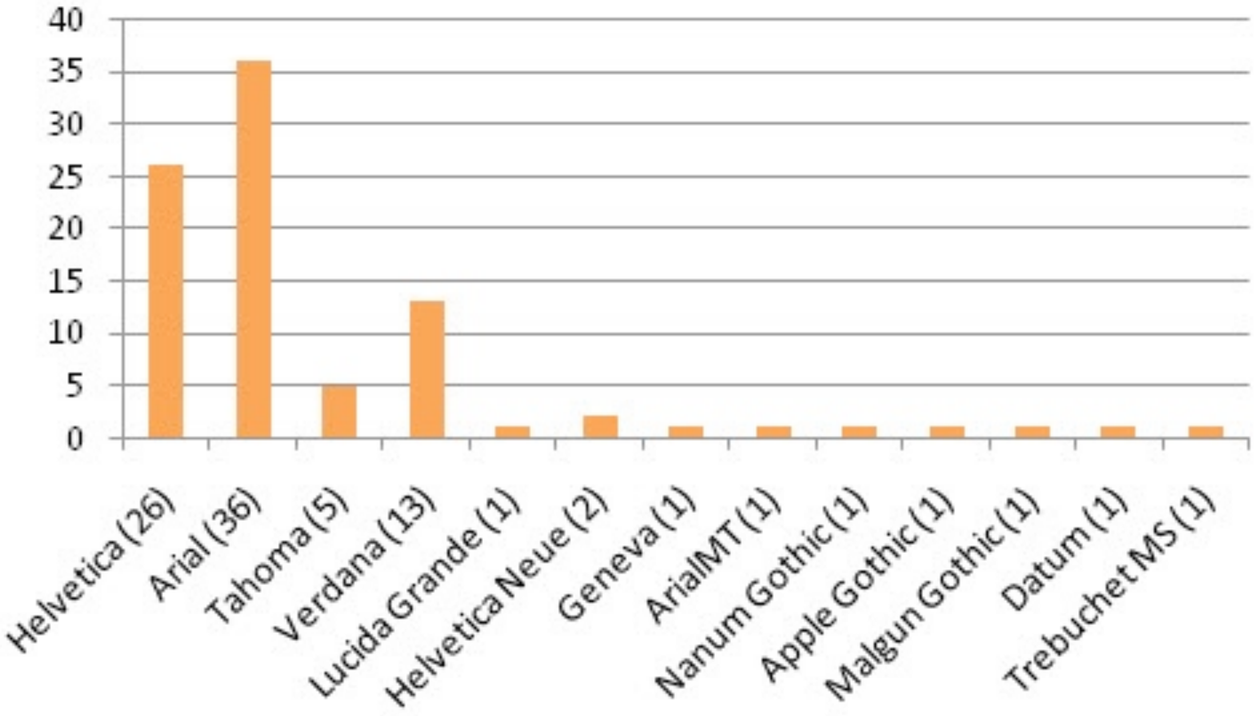


Figure 38: A distribution showcasing the typefaces used within the primary font stack

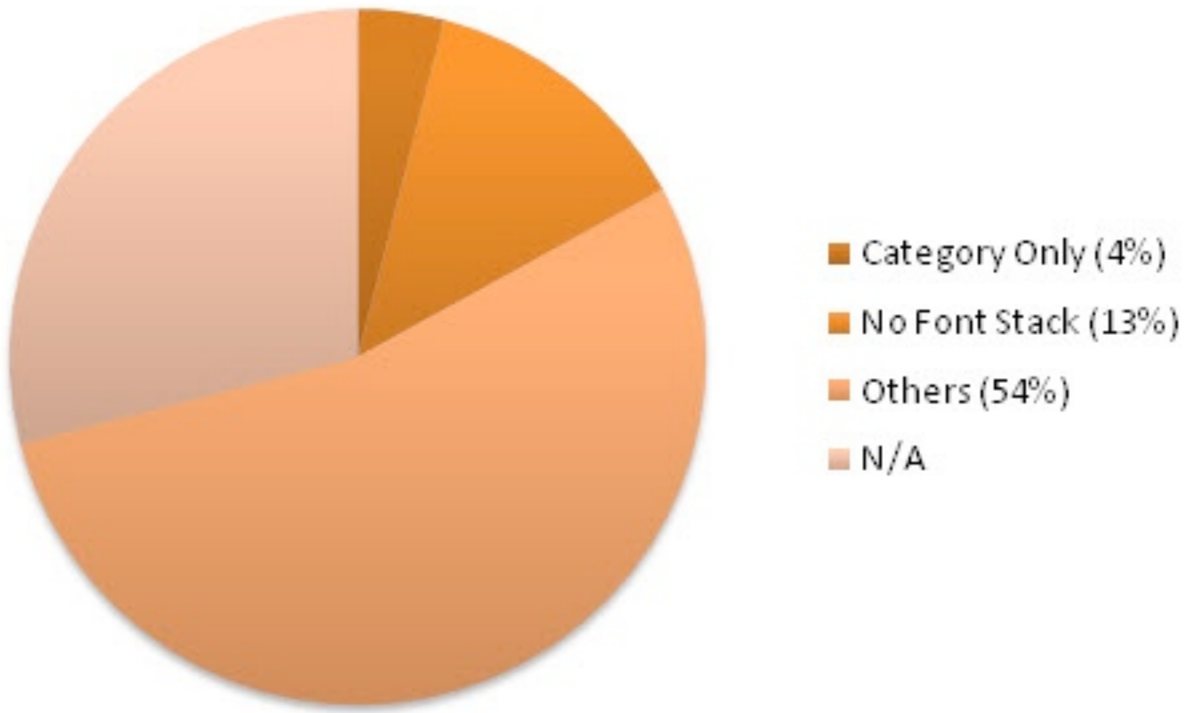


Figure 39: The percentage of mobile designs with no font stack declared (using defaults)

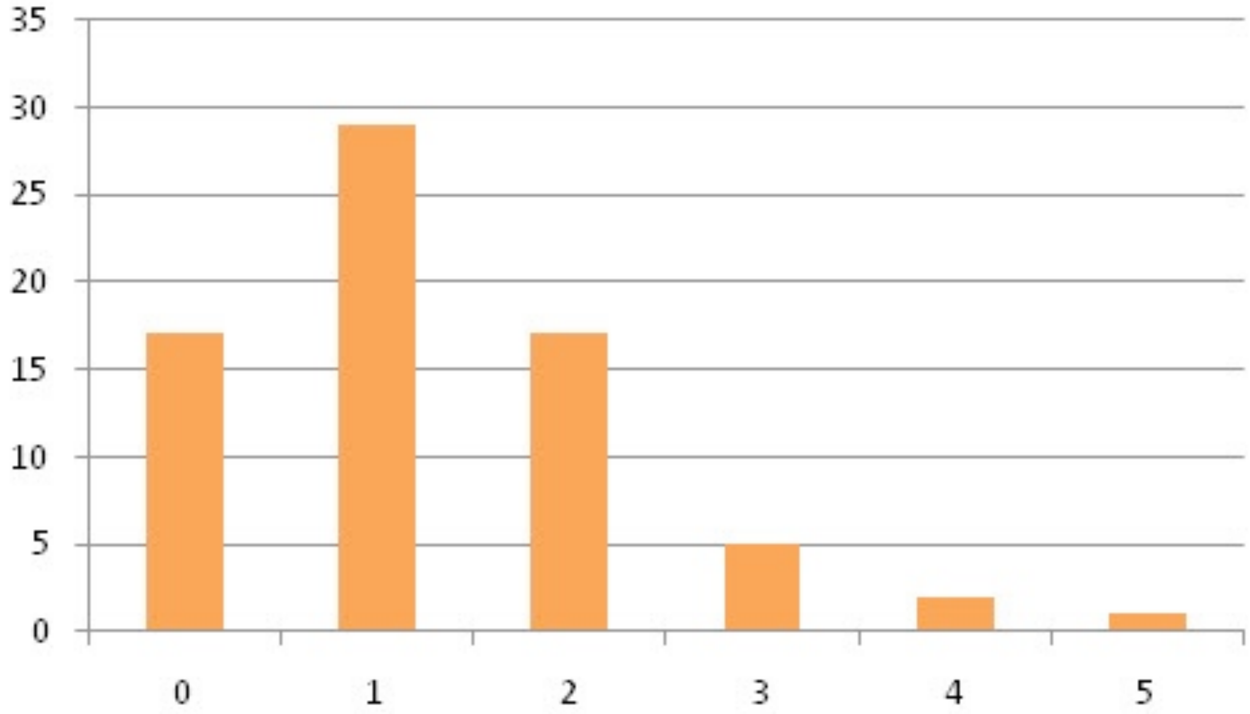
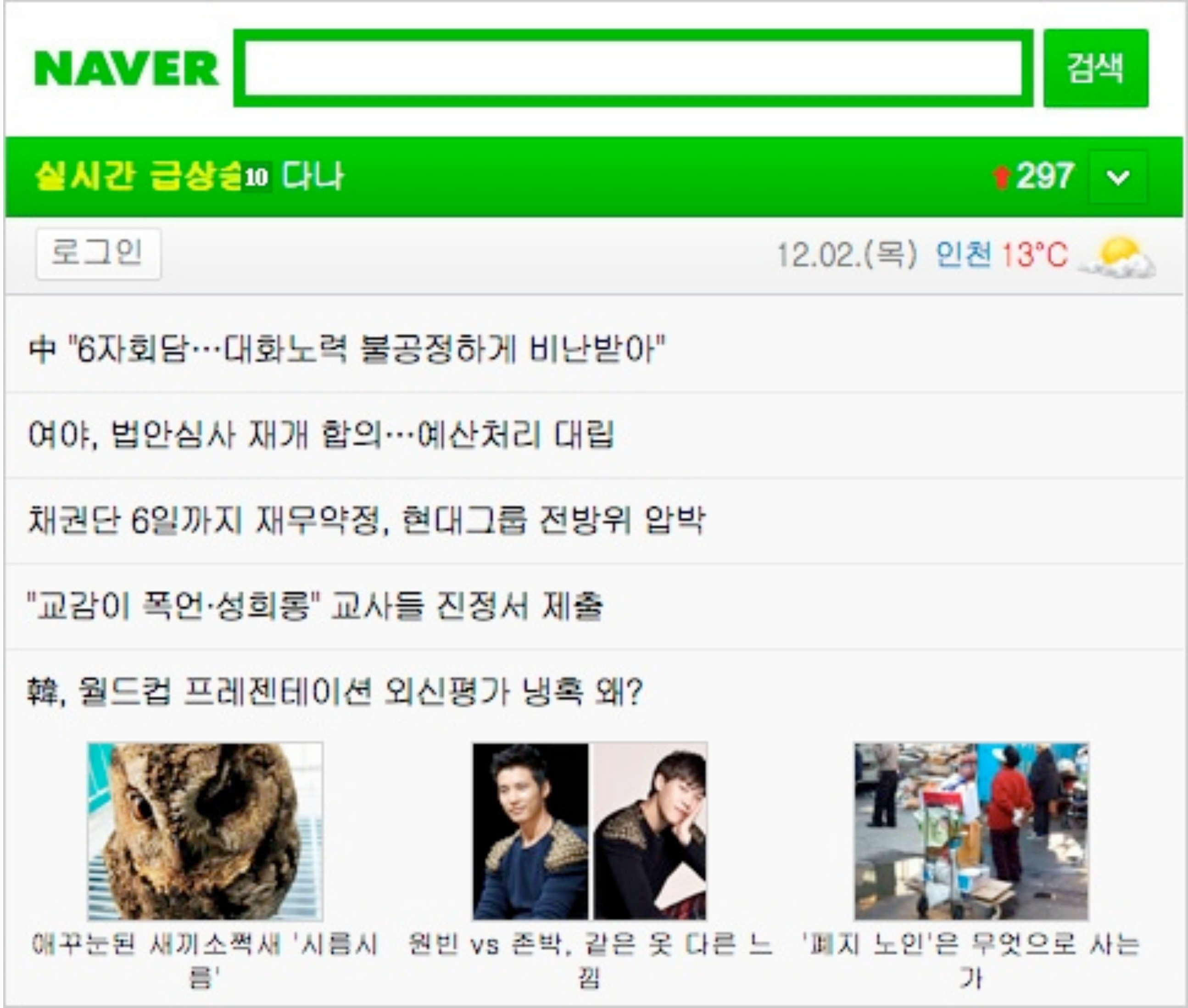


Figure 40: A graph showcasing the number of fonts declared within a family, per website



中 "6자회담…대화노력 불공정하게 비난받아"

여야, 법안심사 재개 합의…예산처리 대립

채권단 6일까지 재무약정, 현대그룹 전방위 압박

"교감이 폭언·성희롱" 교사들 진정서 제출

韓, 월드컵 프레젠테이션 외신평가 냉혹 왜?

애꾸눈된 새끼소쩍새 '시름시름'

원빈 vs 존박, 같은 옷 다른 느낌

'폐지 노인'은 무엇으로 사는가

Figure 41: m.naver.com was slightly more adventurous with non-Websafe typeface usage

The results are quite surprising. In every case where a font family was declared, the category of typefaces used was always sans serif (for both headings and body text). In no instances did serif or another classification get used, and in some cases no font family was declared at all (which could be due to inconsistencies in available fonts for such devices). In addition, the number of occasions where no font stacks were built–resulting in the use of system defaults alone–was fairly significant.

HEADING CONTRASTS

Since the evolution of the cellphone, the ability to have color screens with as much depth and clarity as a desktop PC (using high definition graphics) has increased the ability for us to give our headings and content the colors of the rainbow, both in the foreground and in the background. This test was added to the study in order to see if any trends existed, the range to which color is used within headings, and to determine whether mobile websites made use of background effects such as gradient colors.

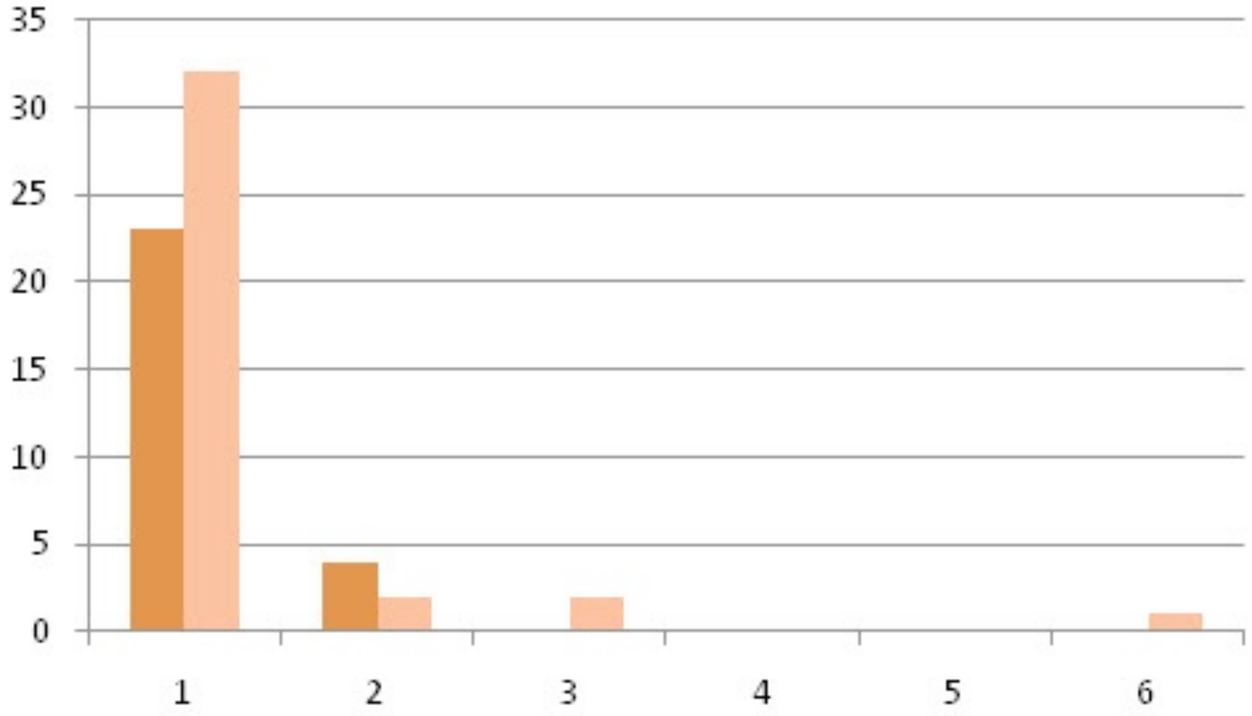


Figure 42: Showcasing the number of colors used within the foreground or background of headings

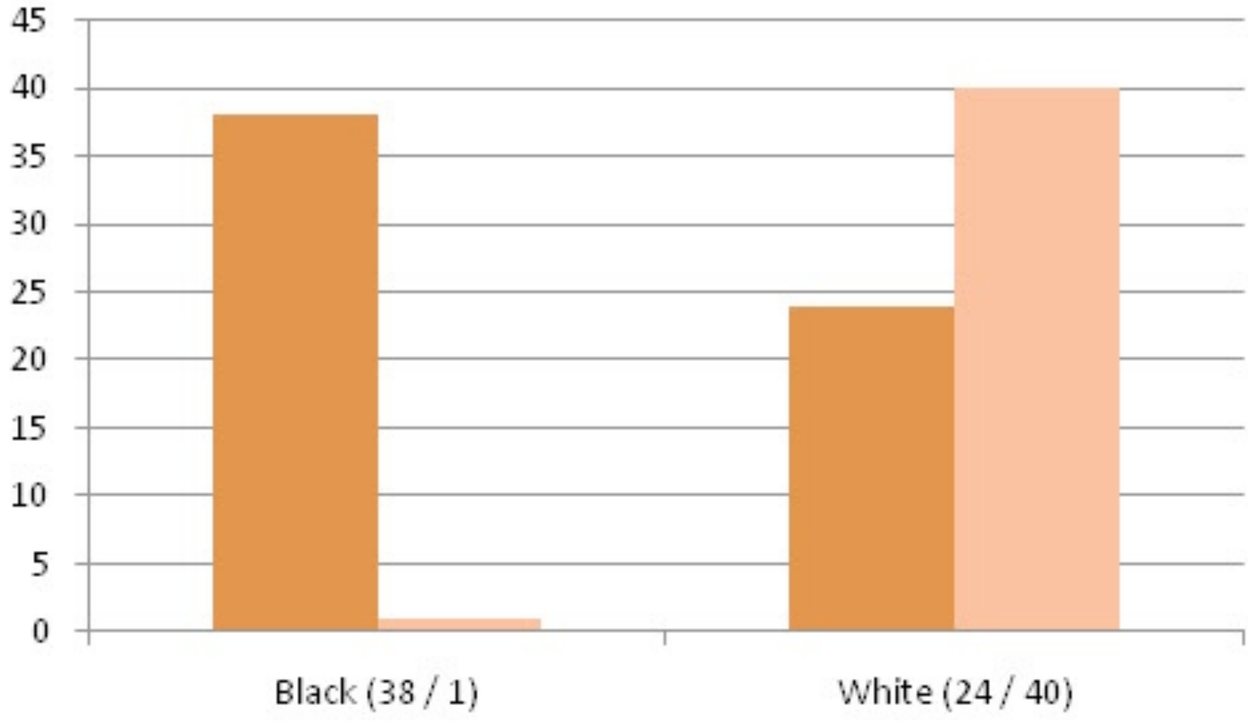


Figure 43: A distribution of websites using a black foreground or white background in headings

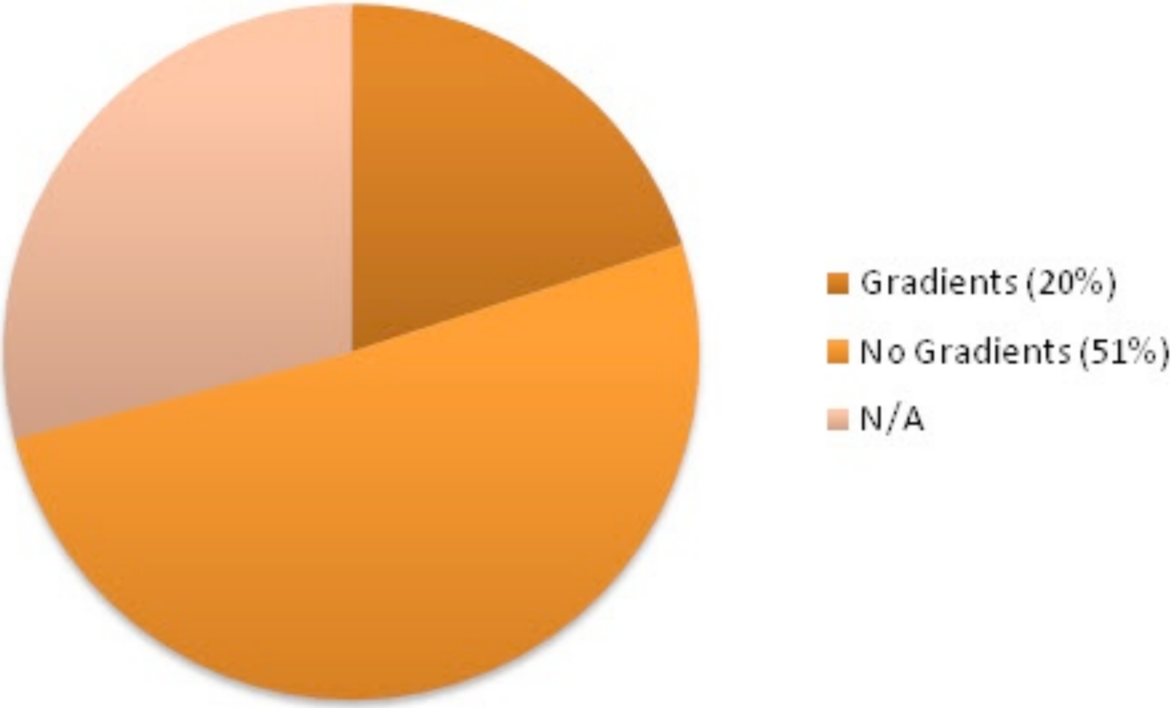
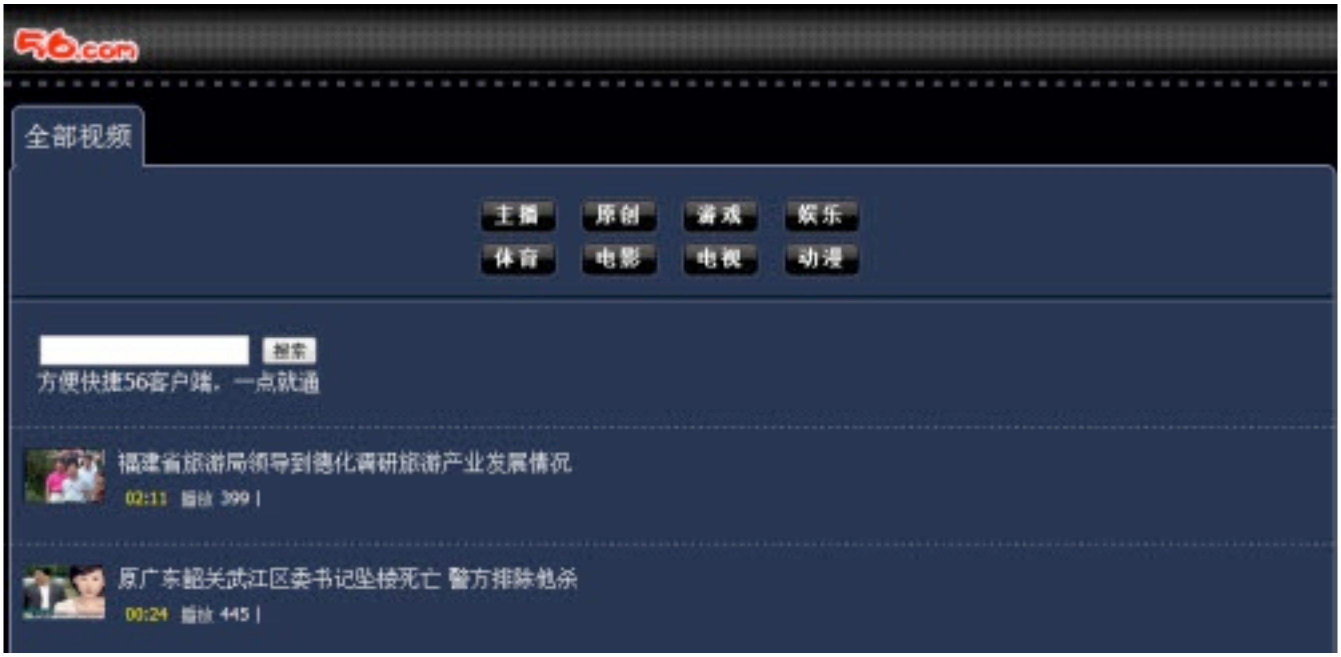


Figure 44: The percentage of mobile websites using a gradient background within a header



方便快捷56客户端，一点就通

福建省旅游局领导到德化调研旅游产业发展情况
02:11 播放 399 |

原广东韶关武江区委书记坠楼死亡 警方排除他杀
00:24 播放 445 |

Figure 45: 56.com made use of a wide range of colors and contrasts within it's headings

As you may have expected, the majority of websites showcased in the list made use of either black or white as the primary colors (in certain cases, shades of grey were used). This stands to reason as a method to keep the contrast ratio high on small screen devices, in order to boost the readability of the content. Another trend uncovered was the preferred use of gradients within the background of headings to give them an added layer of texture, instead of using solid colors or an increased text size.

BODY CONTENT CONTRASTS

The ability to showcase color in our designs is central to how we subtly influence our visitors. Carrying on from the previous test targeted at headings, this one involved examining the colors used within non-heading level elements and determining which colors they used. As this has a big impact within accessibility and the general scope of design, a mixture of visible foreground and background variants were recorded (though as with the previous test, the results were limited to the front page).

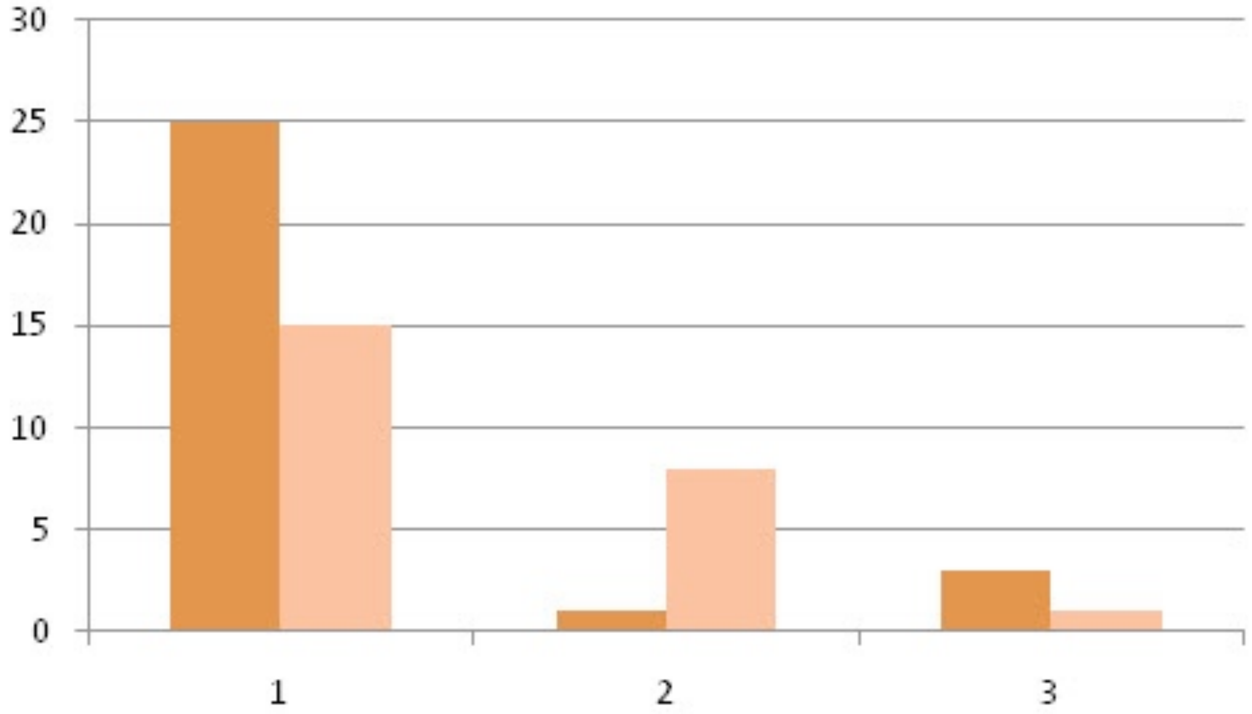


Figure 46: Showcasing the number of colors used within the foreground or background of body content

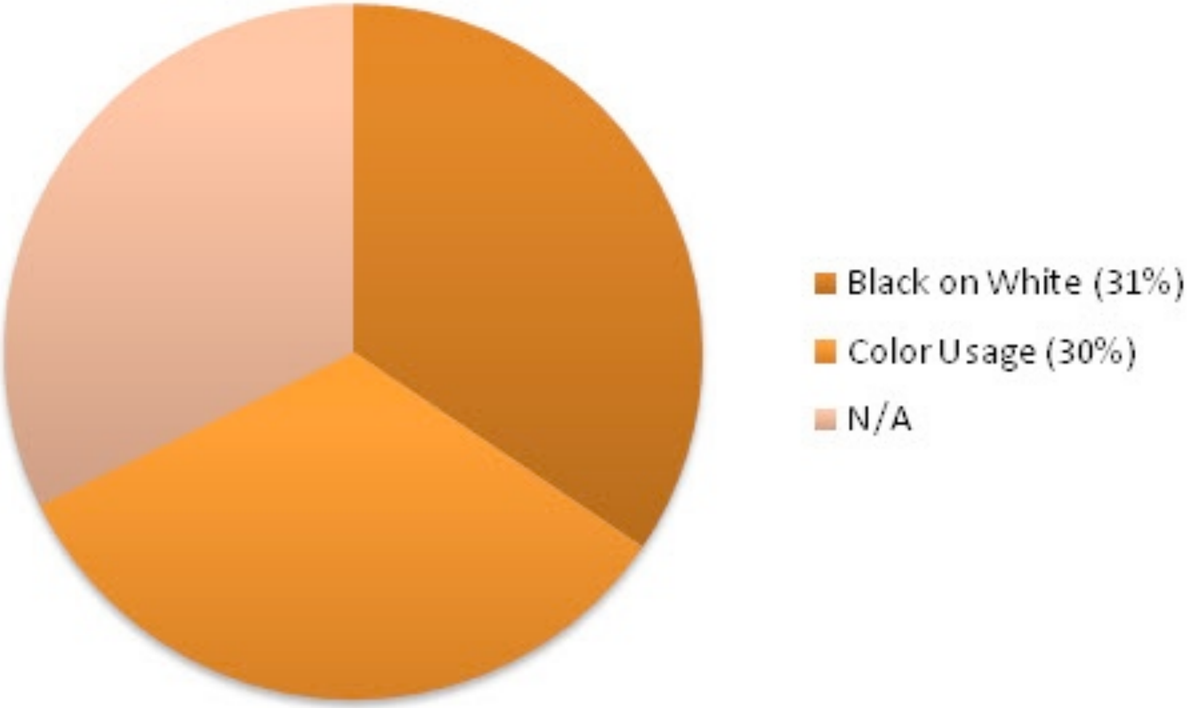
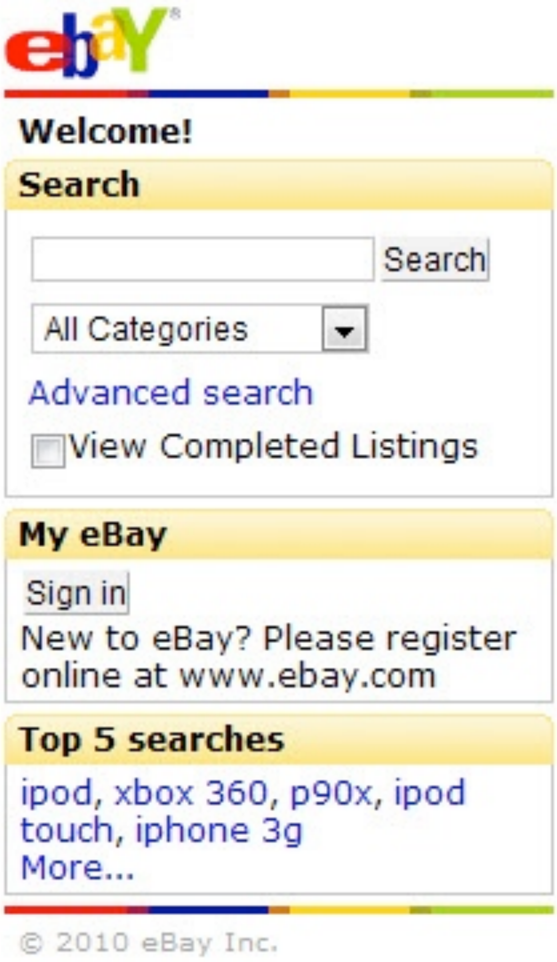


Figure 47: The percentage of websites using a black foreground on a white background in body content



Welcome!

Search

Search

All Categories

Advanced search

□View Completed Listings

My eBay

Sign in

New to eBay? Please register online at www.ebay.com

Top 5 searches

ipod, xbox 360, p90x, ipod touch, iphone 3g

More...

© 2010 eBay Inc.

Figure 48: eBay uses a one-paneled black and white contrast ratio for the body text, like most websites

Within the previous set of results for headings, black and white were shown to be dominant within the text in order to maintain a visible level of contrast. This set of results, as you might expect, follows the same trend and uses less additional colors in the palette. While the color contrasts were only built up from a sample of each website's front page (and then contrasted against each other for the comparison), the results clearly show that a sensible approach to text visibility has been maintained.

AUTHENTICATION REQUIRED

Many places require visitors to login before full access to a service is granted. On most desktop websites, a bunch of information is provided without a user needing to be registered (explaining features or contact details). However, an interesting trend seemed to exist in which mobile users were expected to have an account and know what the service did, as if the mobile website were simply a bonus feature of the service (thereby only offering a login form). The aim of this test was to see if this was the case.

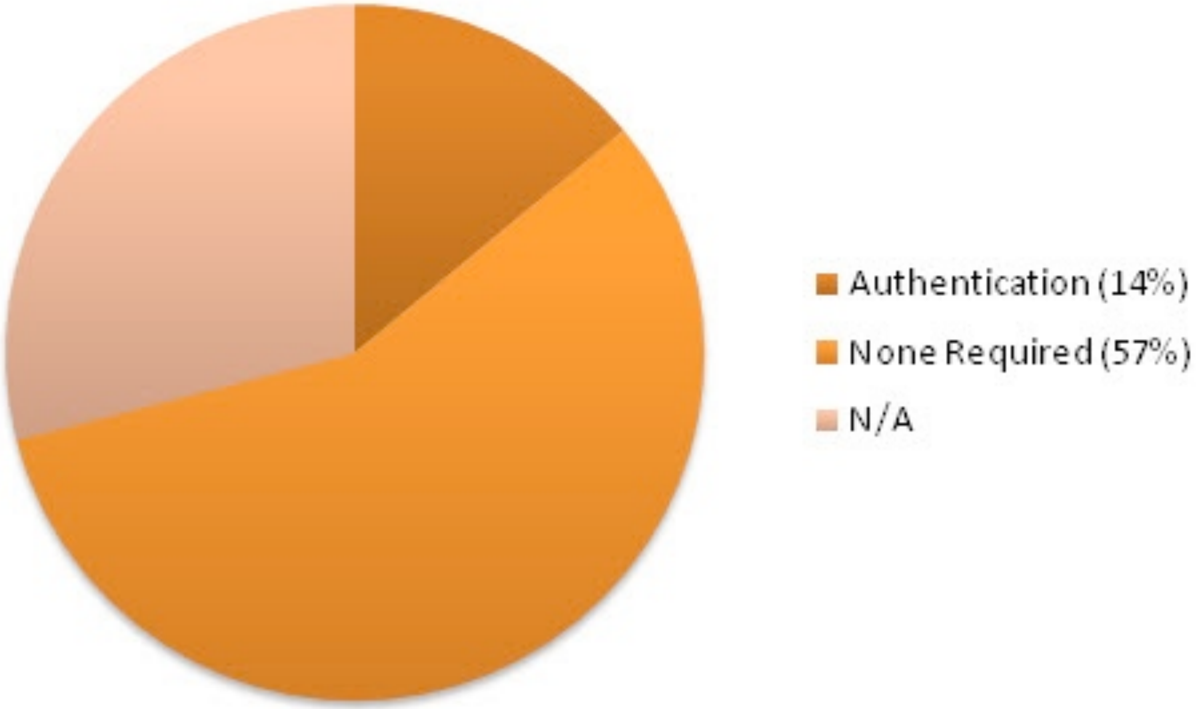
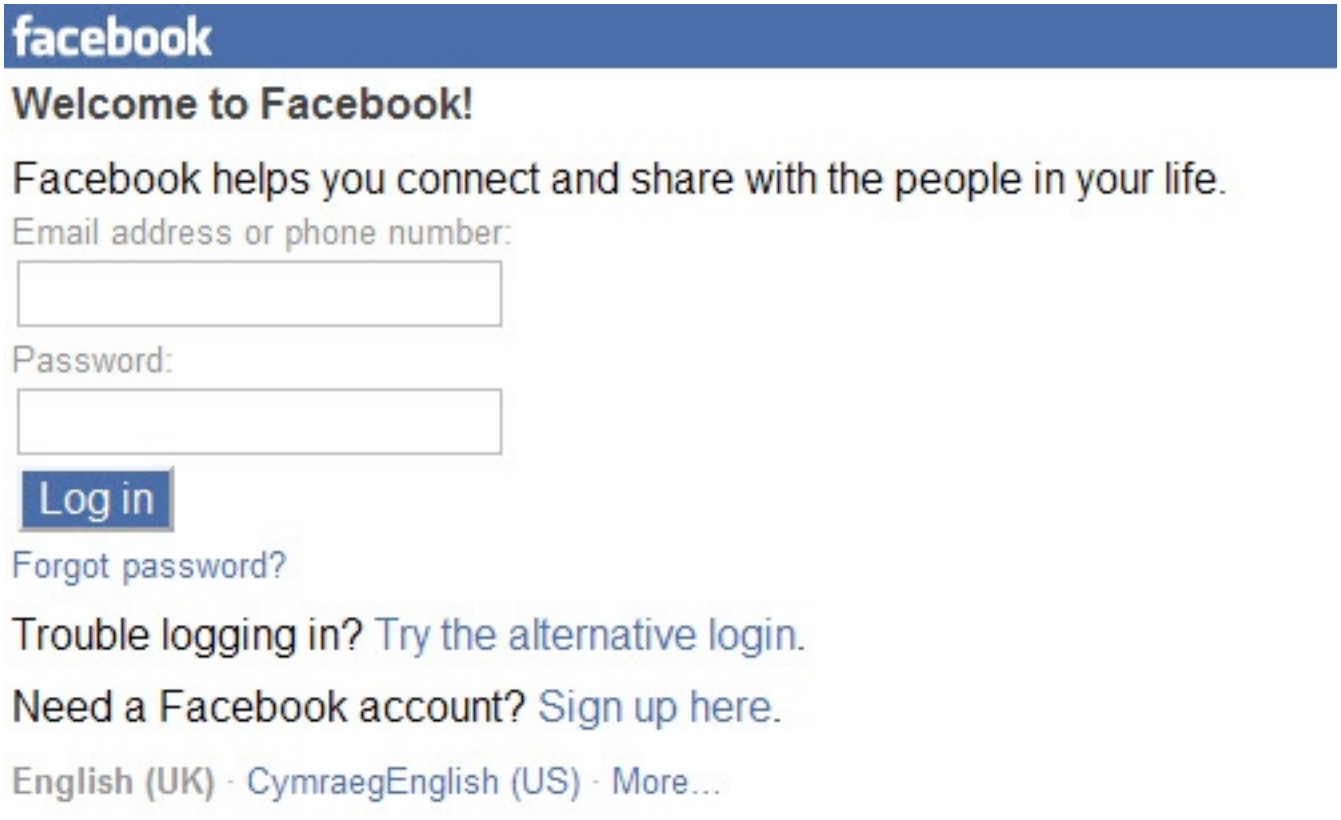


Figure 49: The number of websites requiring the visitor to login before accessing pages



facebook

Welcome to Facebook!

Facebook helps you connect and share with the people in your life.

Email address or phone number:

Password:

Log in

Forgot password?

Trouble logging in? Try the alternative login.

Need a Facebook account? Sign up here.

English (UK) · CymraegEnglish (US) · More…

Figure 50: Facebook requires you to authenticate before accessing the website

While it's understandable that some websites do not require users to authenticate themselves–and this means that the number of recorded "forced logins" will be drastically reduced–the results of this test are quite shocking. While it should be considered bad practice to require logins on a website without any description of the service being present (for mobile-only traffic), a number of popular websites which were contained within the top 100 list proved positive for doing this, as they had no useful site information!

RETURNING TO A FULL WEBSITE

The next test depended entirely on the mobile website's ability to follow common requirements and return the visitor back to a non-optimized website upon request. While a website's mobile experience will be enough for some, it's important to realize that some people may not adjust well to new a UI or may request functionality that only exists on the main website, and offering a fallback mechanism is worthwhile. The test also aimed to uncover any naming conventions used to represent this link (if one exists).

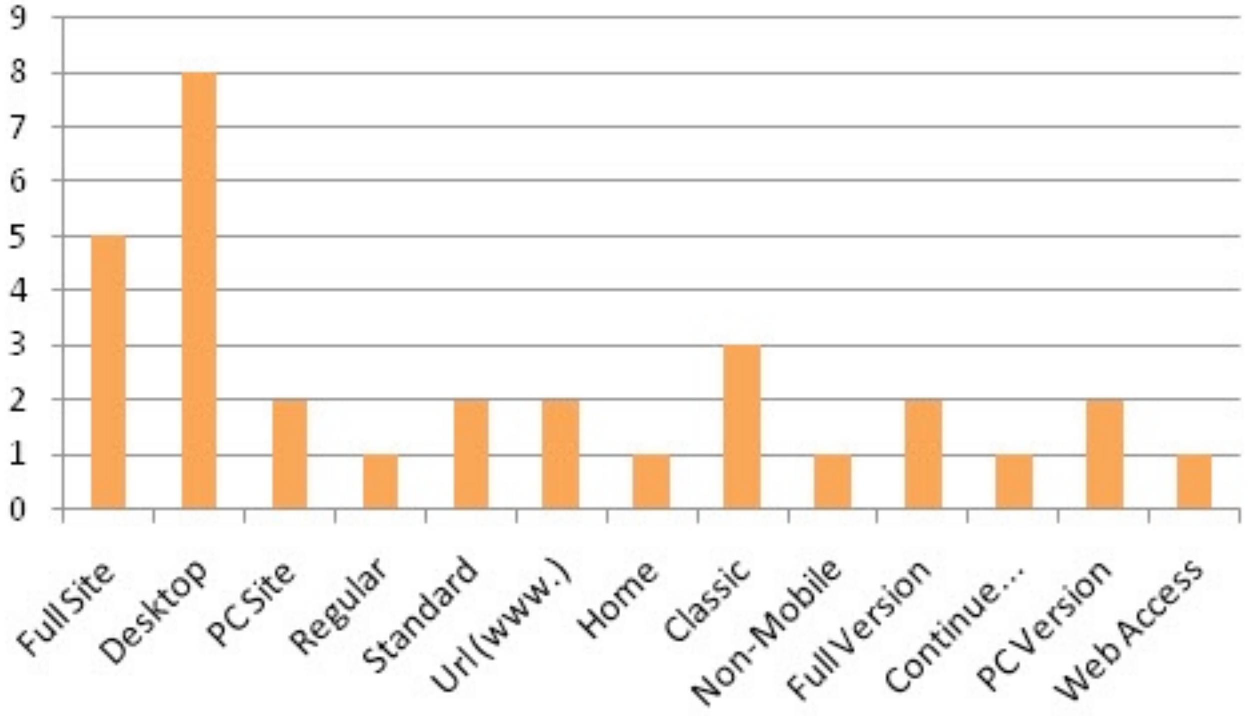


Figure 51: A graph of the various naming conventions used in links toward a full website link

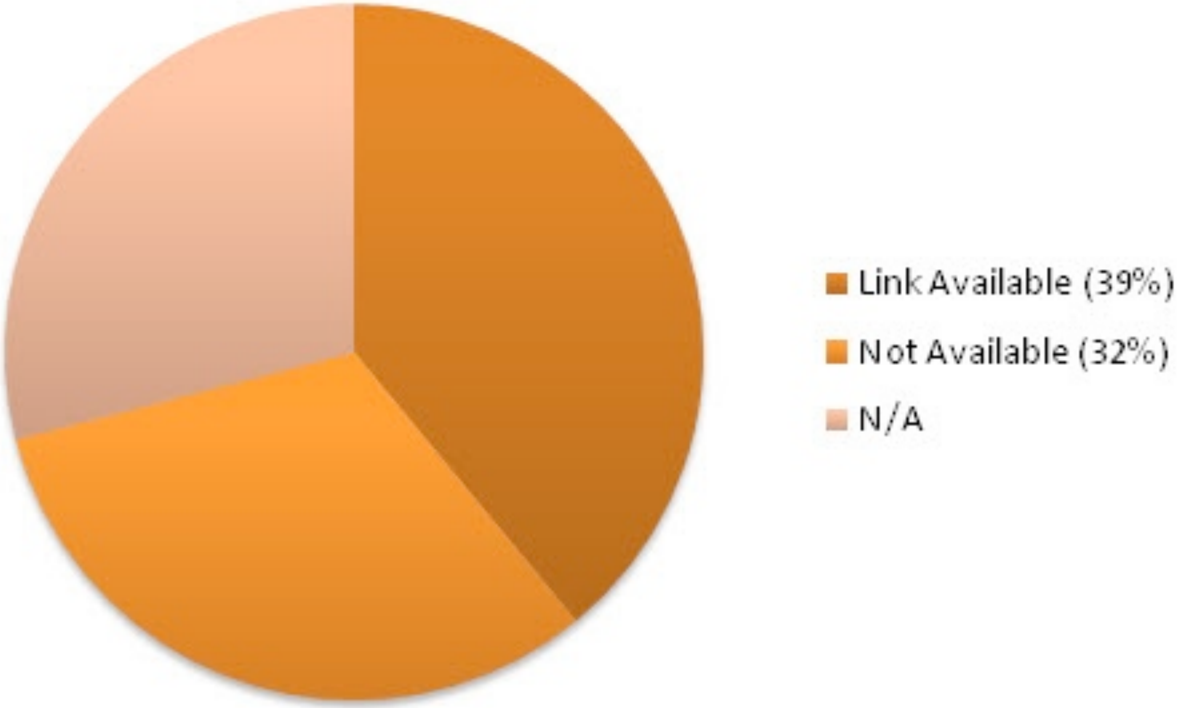
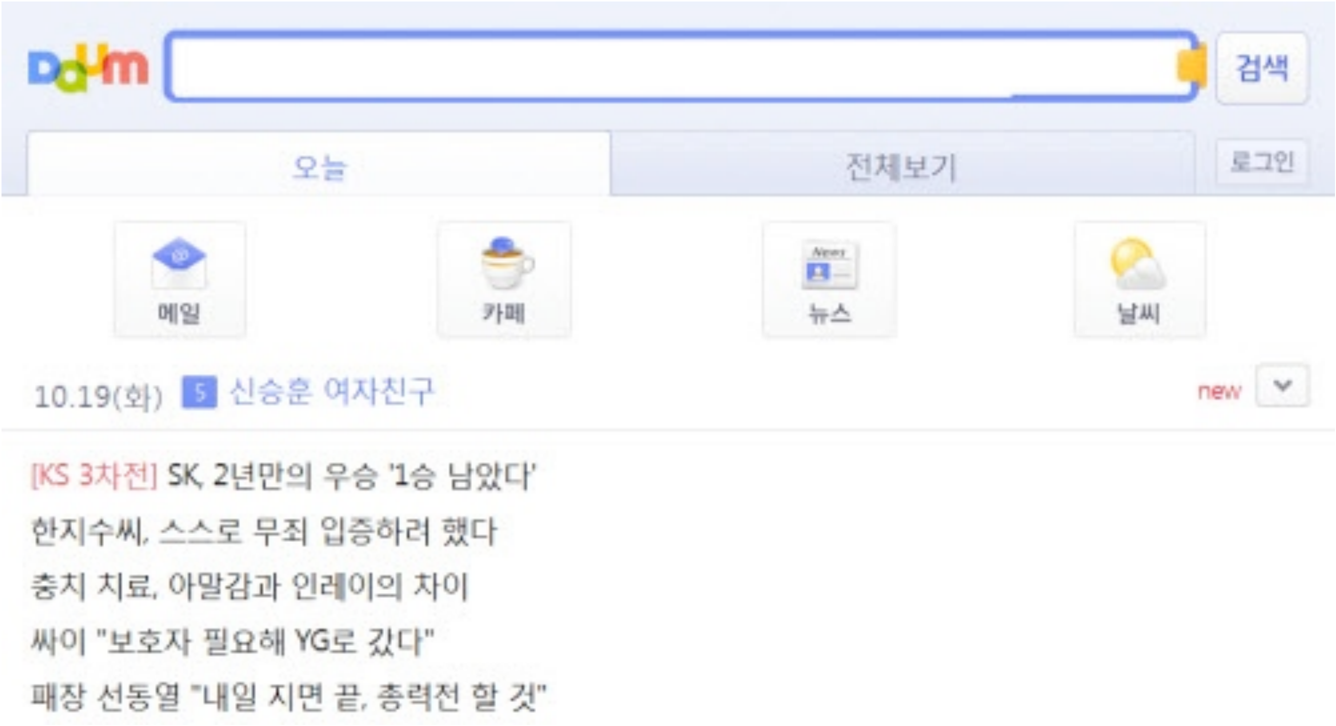


Figure 52: The percentage of websites which do not have a link returning you to the full website



[KS 3차전] SK, 2년만의 우승 '1승 남았다'

한지수씨, 스스로 무죄 입증하려 했다

충치 치료, 아말감과 인레이의 차이

싸이 "보호자 필요해 YG로 갔다"

패장 선동열 "내일 지면 끝, 총력전 할 것"

Figure 53: Daum.net allows gaining access to the full website by clicking the "PC Version" link

In the results, some websites offering mobile versions of their services used common words in their links to indicate returning to the main website, such as “full site” or even “desktop version.” In these cases, no common preference could be found even though usage was spread fairly evenly. What is quite surprising however is that many who force-redirected visitors to the mobile website, refused to link or to allow access the full website. This in turn limited the ability of mobile visitors to access some services.

NAVIGATION CONVENTIONS

This test was focused upon what could be considered one of the most important elements of a website. A successful navigation scheme can be the difference between an easy-to-use interface, and a complex and potentially unusable website. When carrying out this test, four types of navigation conventions were checked as to whether they were being used on the home page (most sites used more than one): text links, icon-based navigation, image links, and special menus (such as dropdowns or panels).

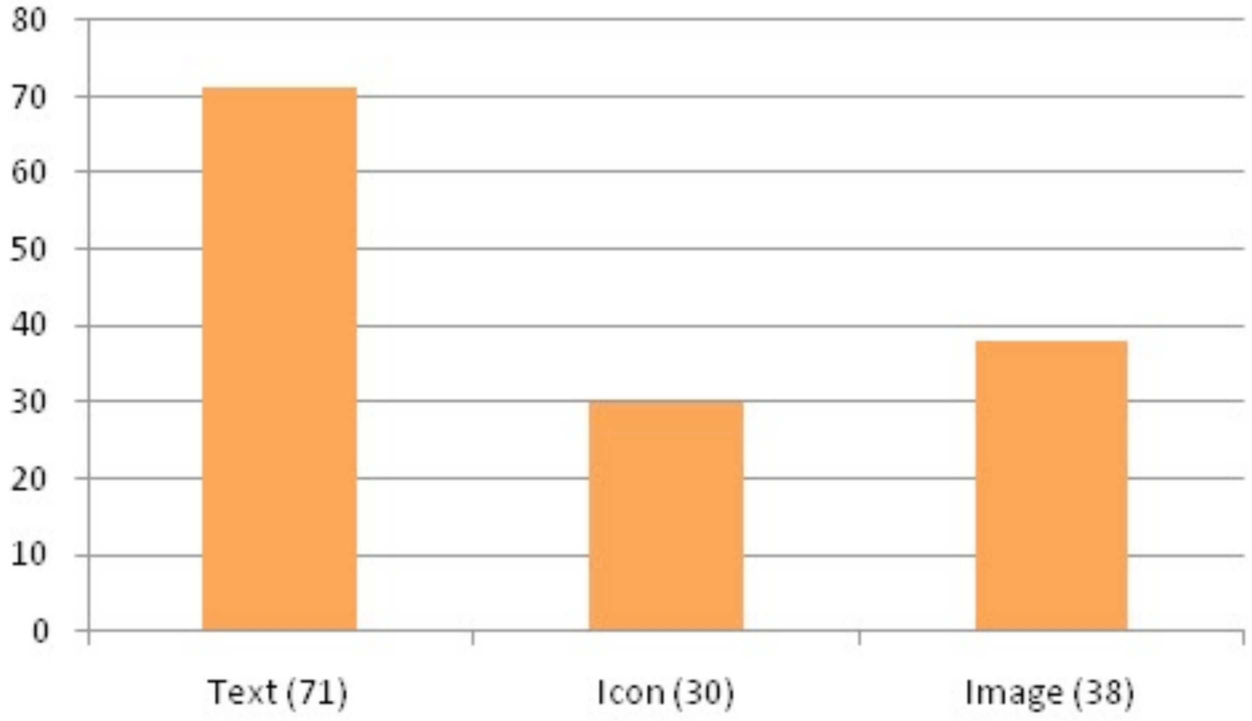


Figure 54: A distribution of websites making use of text, icon, or image-based navigation styles

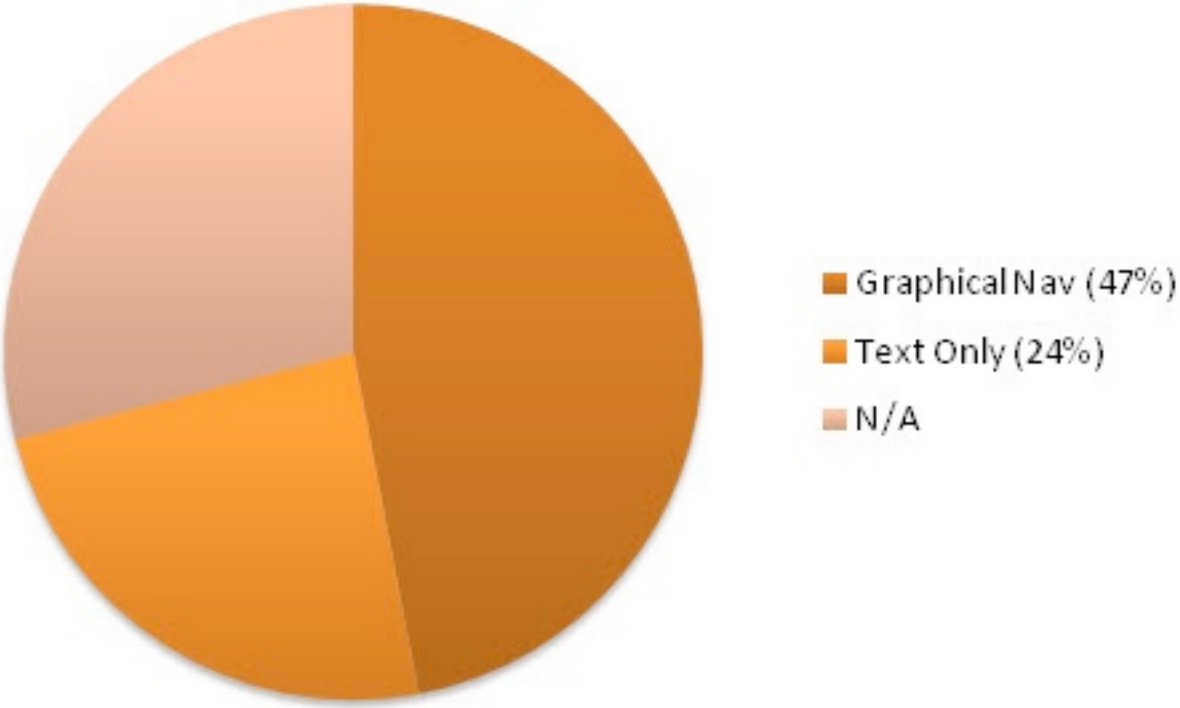


Figure 55: The percentage of icon- and image-based navigation against conventional text

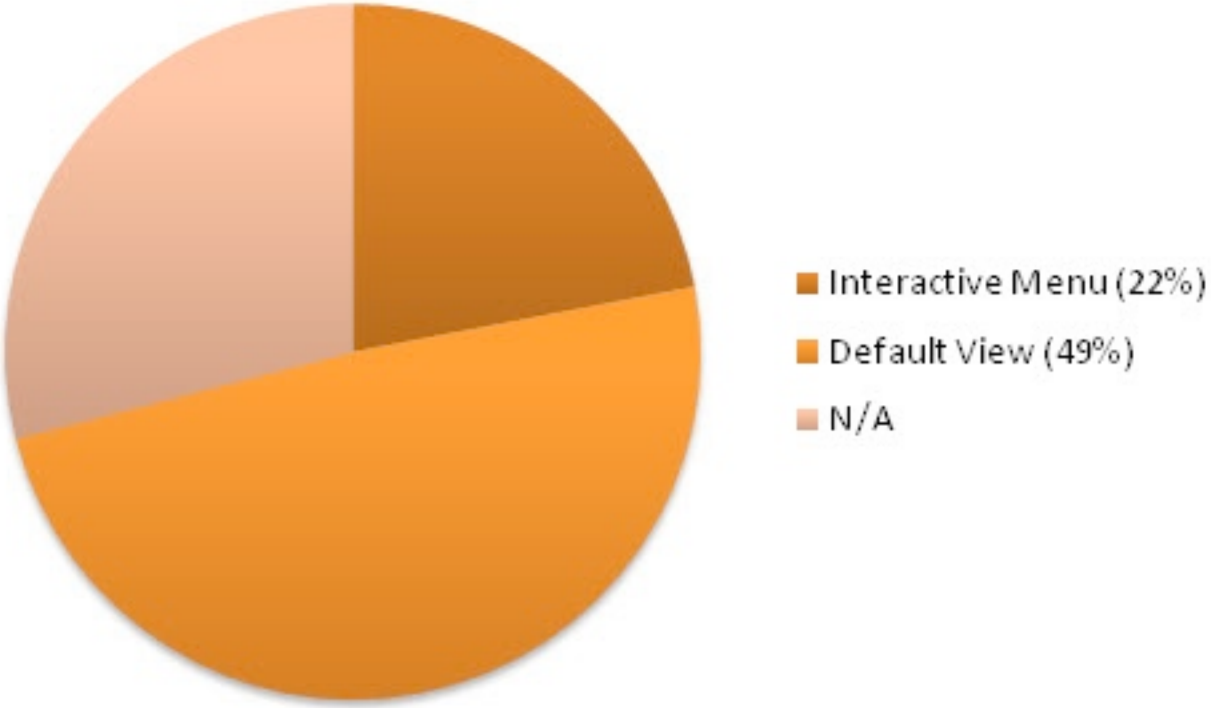
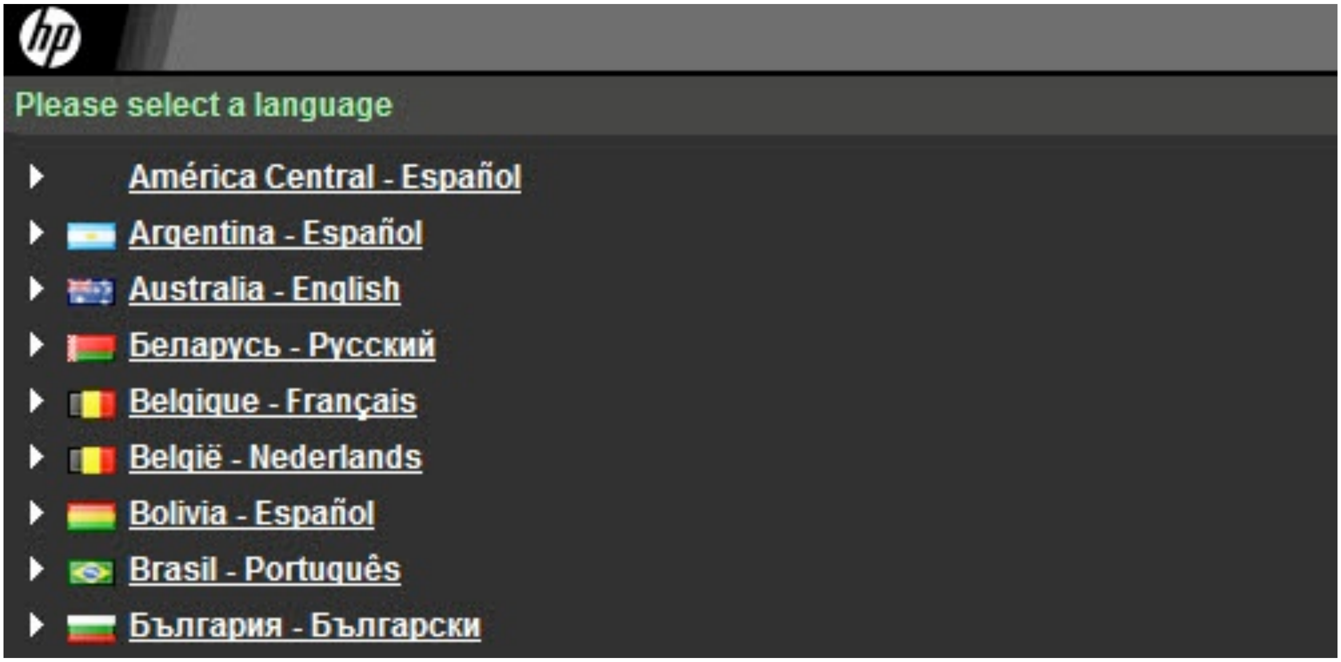


Figure 56: The percentage of websites which employ the use of menus (or content on demand)



Please select a language

- América Central - Español
- Argentina - Español
- Australia - English
- Беларусь - Русский
- Belgique - Français
- België - Nederlands
- Bolivia - Español
- Brasil - Português
- България - Български

Figure 57: HP's mobile website enlists a menu to determine your country of origin

The results of this test were fairly conclusive in that every single website (as you might imagine) had standard text anchor links, though some menus were formed of plain text. In addition, image-based menu options were quite popular and were actually used more often than icon-based navigation–which, for devices that depend on icons representing apps or services, was quite a surprise. Special menus also saw some popularity, especially when used with content-on-demand scrolling mechanisms.

HOME PAGE LINK RATIO

Having a relatively small amount of space can be a real problem in mobile devices, and therefore the potential for confusion as to navigation options, and how many links are provided, can potentially increase the difficulty for users to know the extent of where their choices will lead. If a website has too many links (or too few), the choice can quickly overwhelm a user's sense of perspective. As such it seemed worthwhile to see how many options were provided in the form of links on a single page.

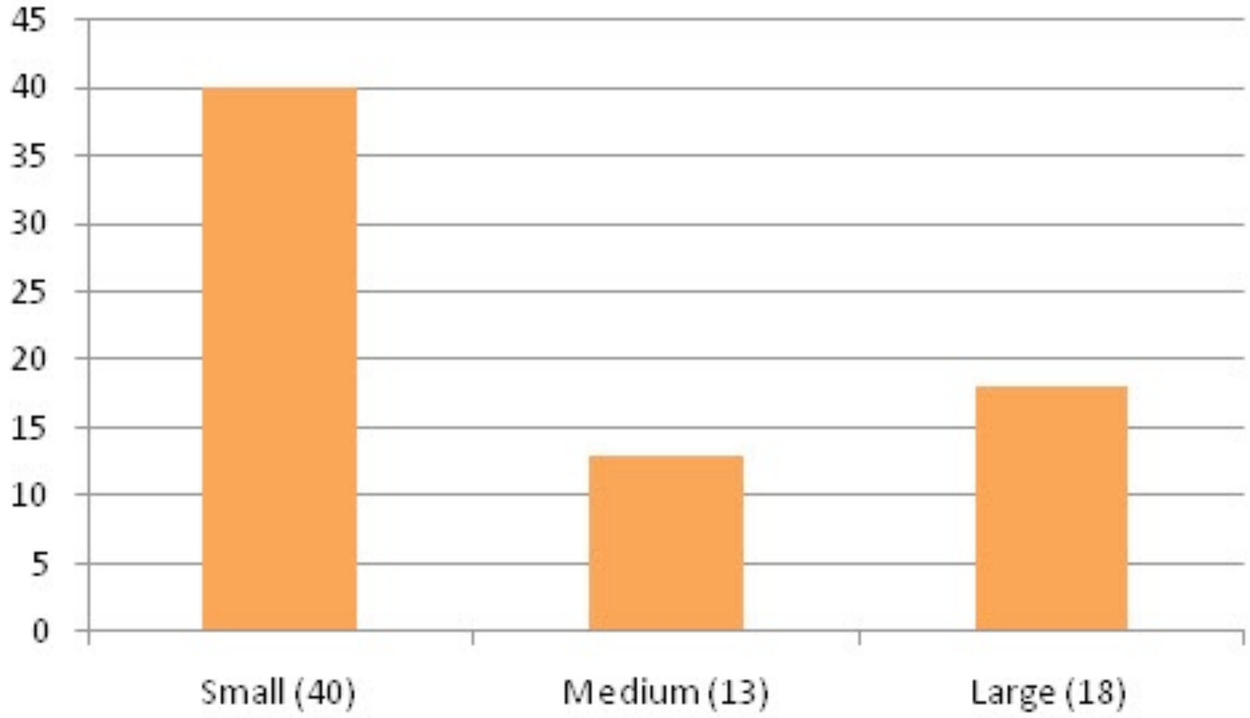


Figure 58: The number of websites using a small, medium, or large number of links on the front page

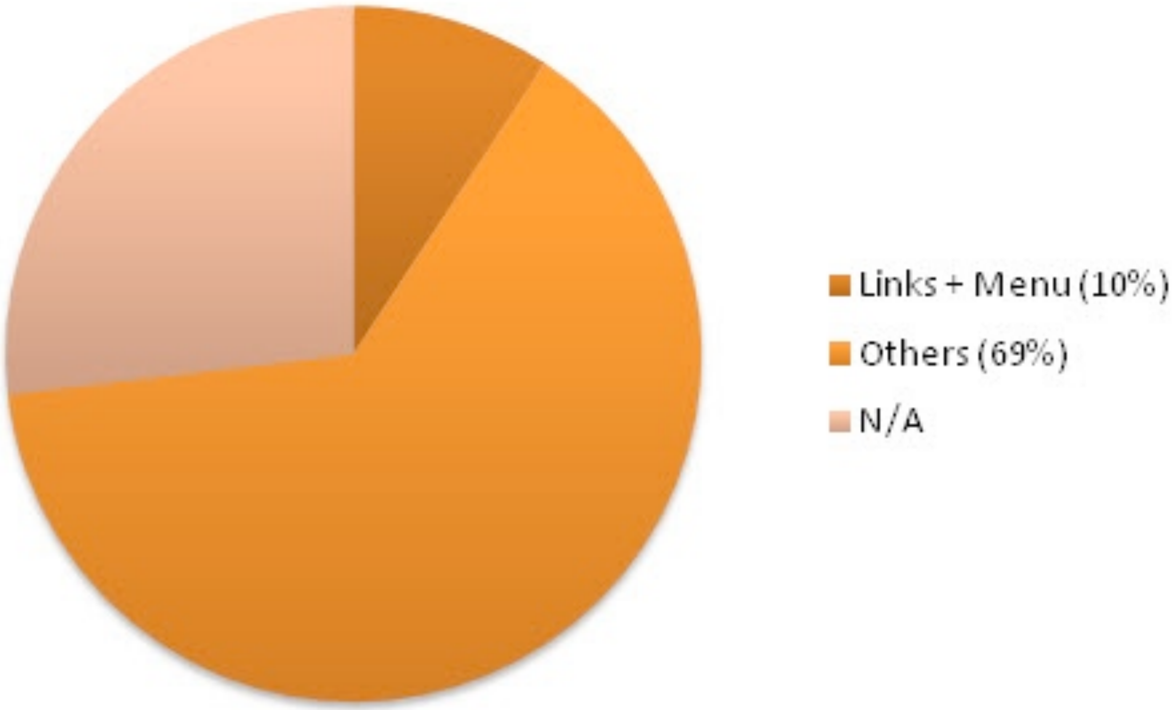


Figure 59: The percentage of websites using a large number of links as a result of a menu convention

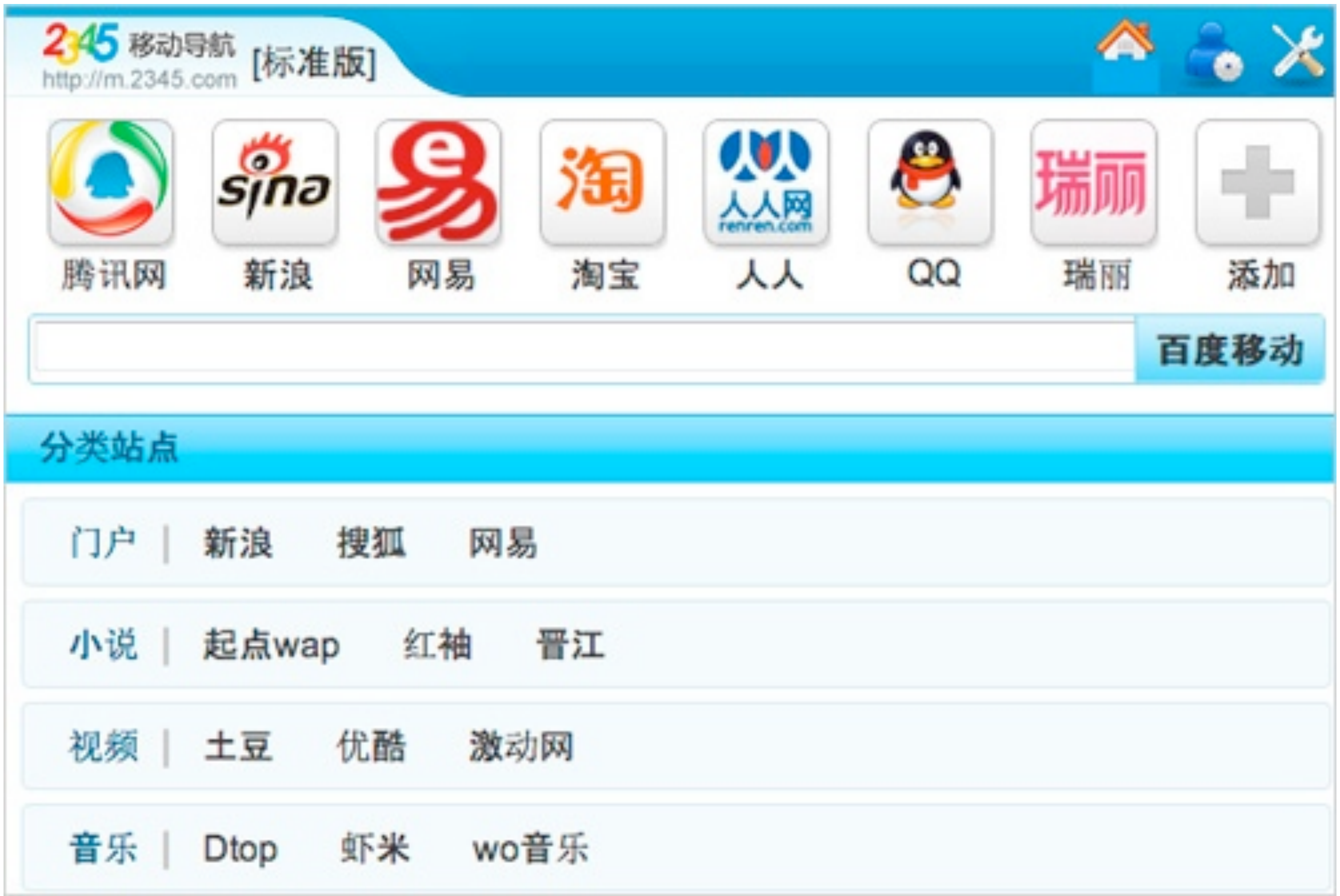


Figure 60: 2345.com has loads of links within its home page, but is that really a good thing?

When testing the websites, the results were categorized based on the number of links that existed in the front page of a website (these were labeled as small, medium, or large as per the link ratio). For the websites being measured, the results indicated that fewer links were visible on a mobile website than on the desktop variant (even when complex layouts were used). This seems to follow conventions, in that a reduced viewing space should hold less information to avoid unnecessary scrolling, and to reduce complexity.

GOLDEN IMAGE RATIO

Images on a mobile website can be a tricky affair, as these devices often require more bandwidth than anything else and therefore can reduce the loading times. Within this test, the trend of using small, medium, and large as a representation of the number of images on the page were used, similar to the link testing. Primarily focused upon the home page, the amount required to fall into each category was reduced as most websites will naturally hold more text links than images (per page, on average).

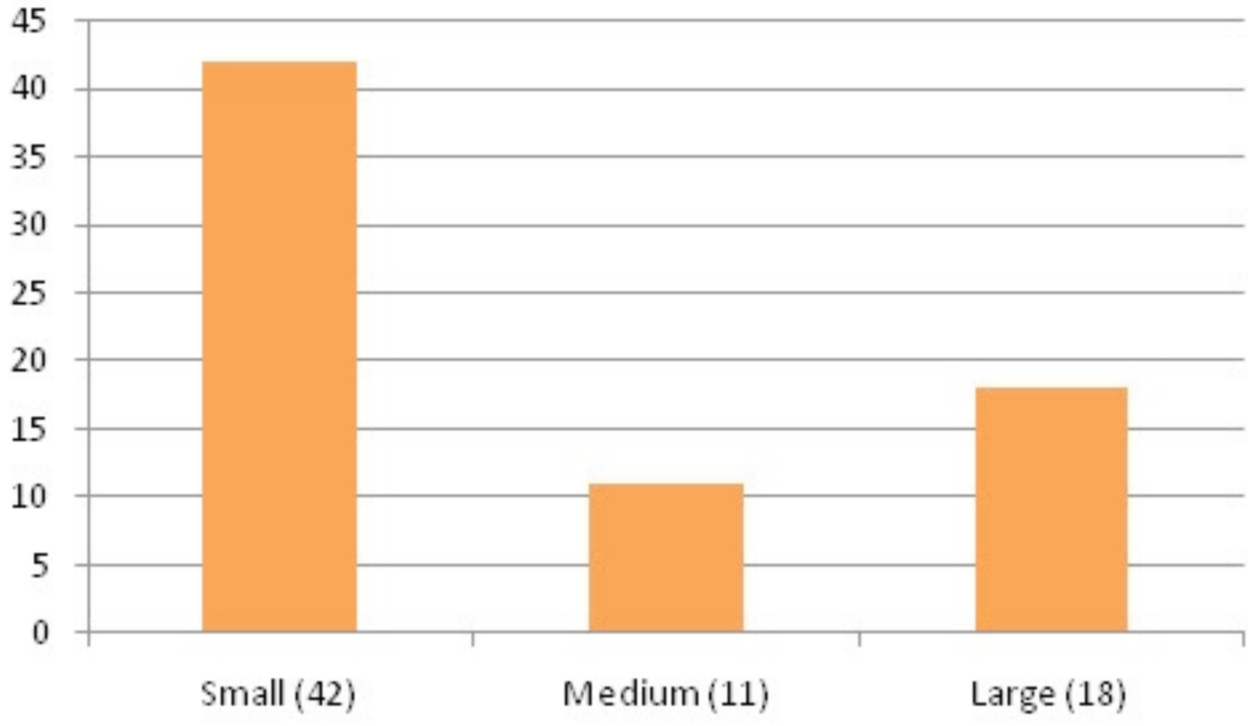


Figure 61: The number of websites using a small, medium, or large number of images on the front page

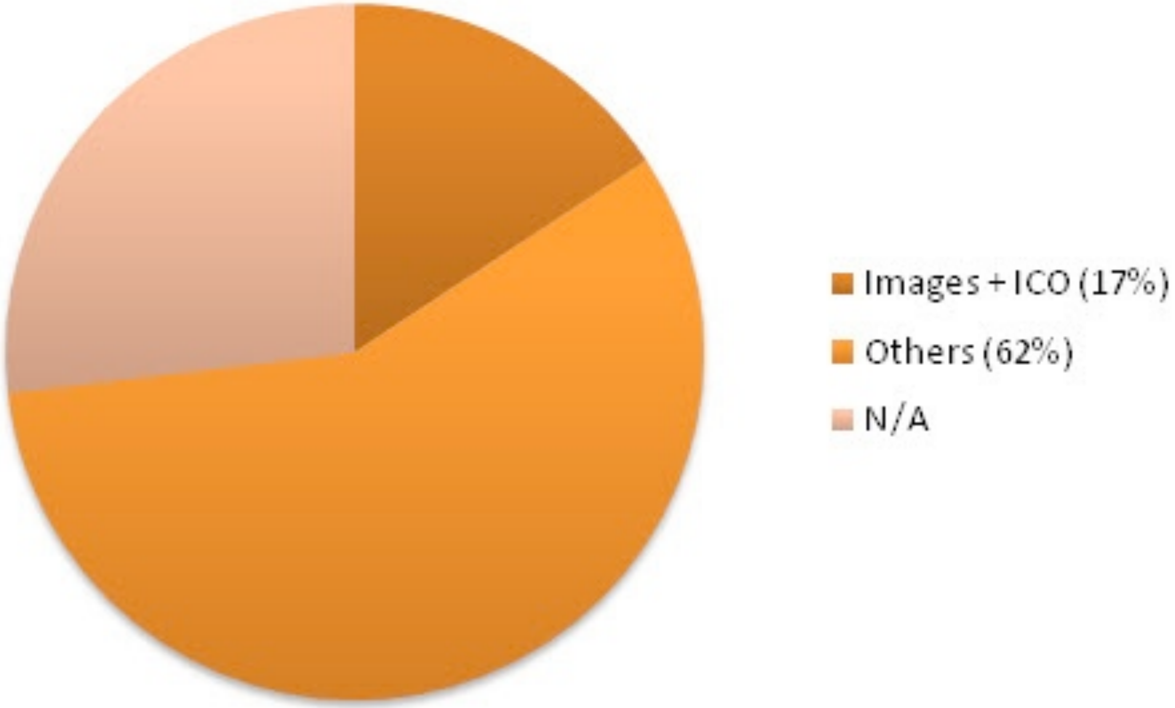


Figure 62: The percentage of websites using a large number of images resulting from icon navigation

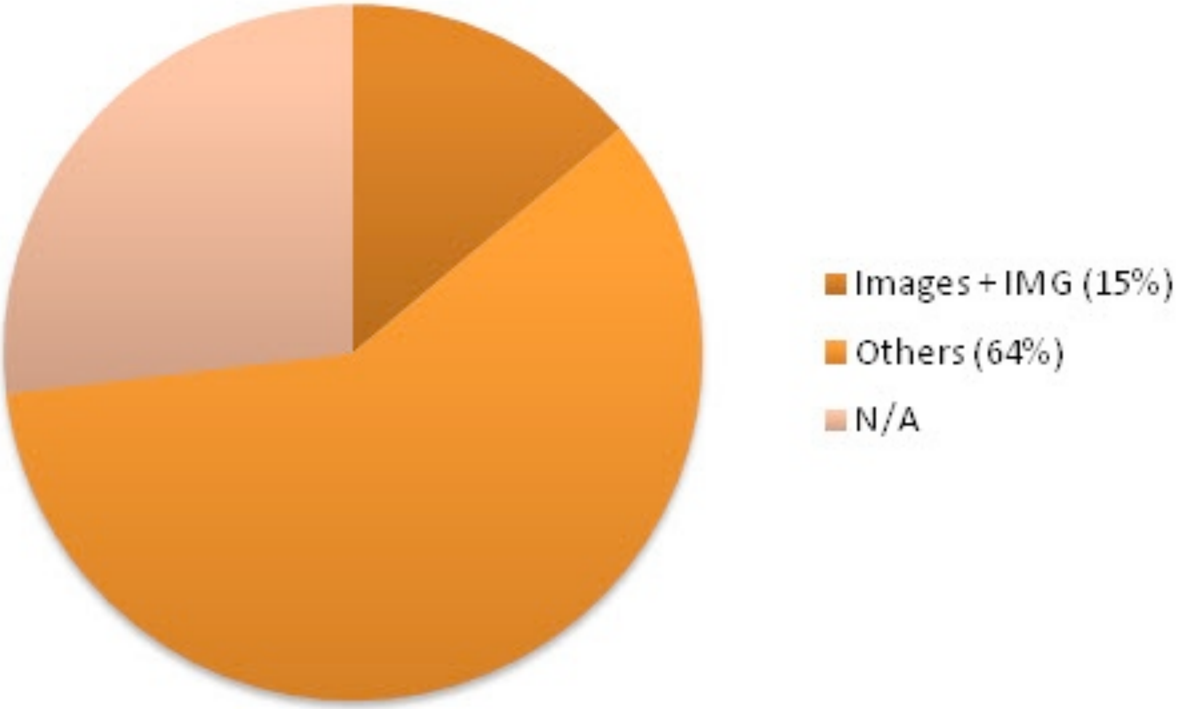
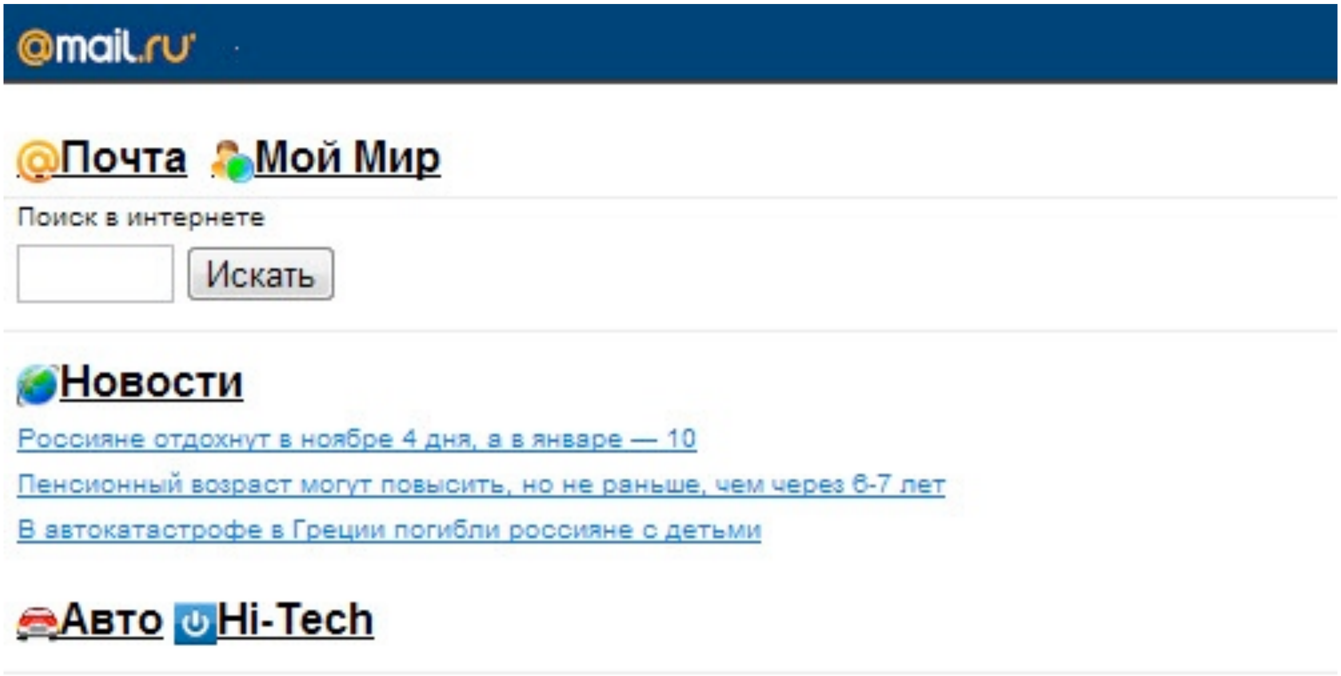


Figure 63: The percentage of websites using a large number of images resulting from image navigation



@mail.ru

@Почта Мой Мир

Поиск в интернете

Искать

Новости

Россияне отдохнут в ноябре 4 дня, а в январе — 10

Пенсионный возраст могут повысить, но не раньше, чем через 6-7 лет

В автокатастрофе в Греции погибли россияне с детьми

Авто Hi-Tech

Figure 64: Russian Mail.ru uses icons and images to highlight specific link usage within the page

As you might imagine, on mobile websites the results backed up many assumptions that mobile websites will have not only fewer images on the page, but will have smaller images presented on the screen (to match the kind of experience available on small screens). This in turn helped reduce the file sizes in earlier tests (and the speed benchmark). What makes this result particularly interesting is that websites (such as Flickr) which oriented around images, also followed this reductionist trend.

LINK CLICK REGION

The purpose of this test was to go beyond the mere presence of links (and images with links) and to examine how large the click region actually was on the page. With mobile devices and small screens supporting touch sensitivity, there is a need to keep link click regions as large as possible to ensure the usability and accessibility of such devices (to help with big, imprecise fingers). The scale used followed small, medium, or large, and focused upon the link click's size compared to other elements on the page.

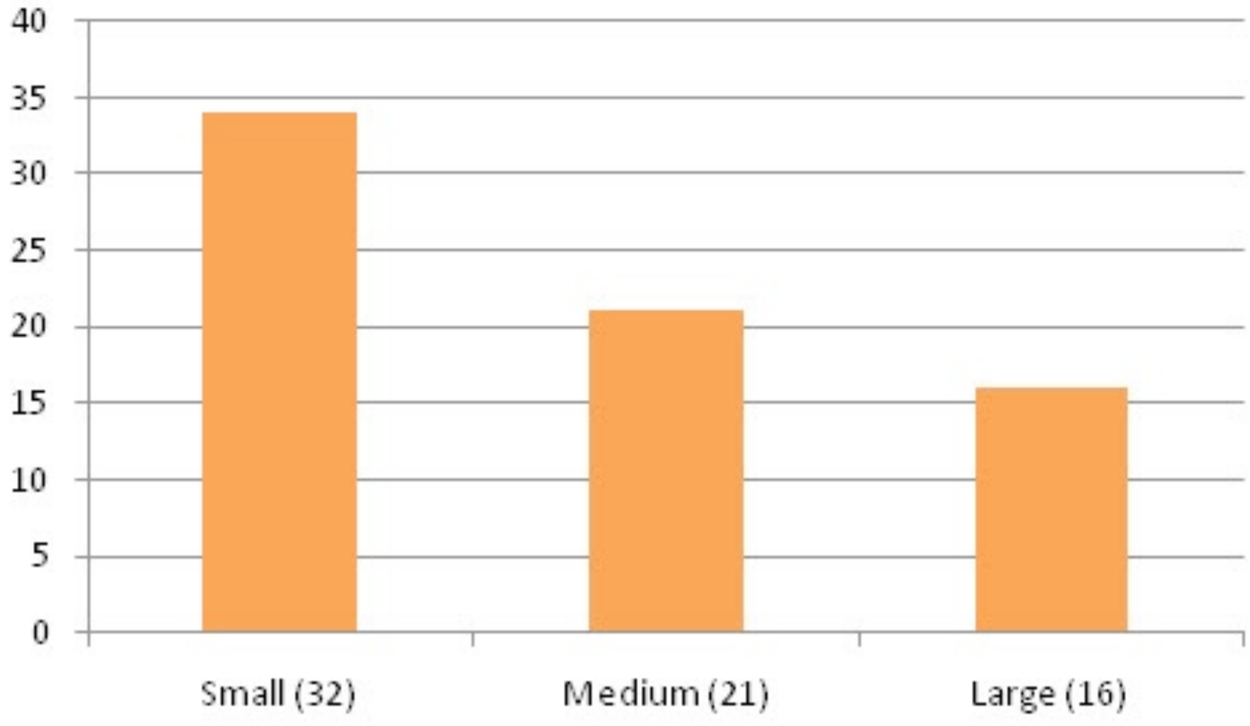


Figure 65: A graph with the average size of a website's link click region (small, medium, or large)

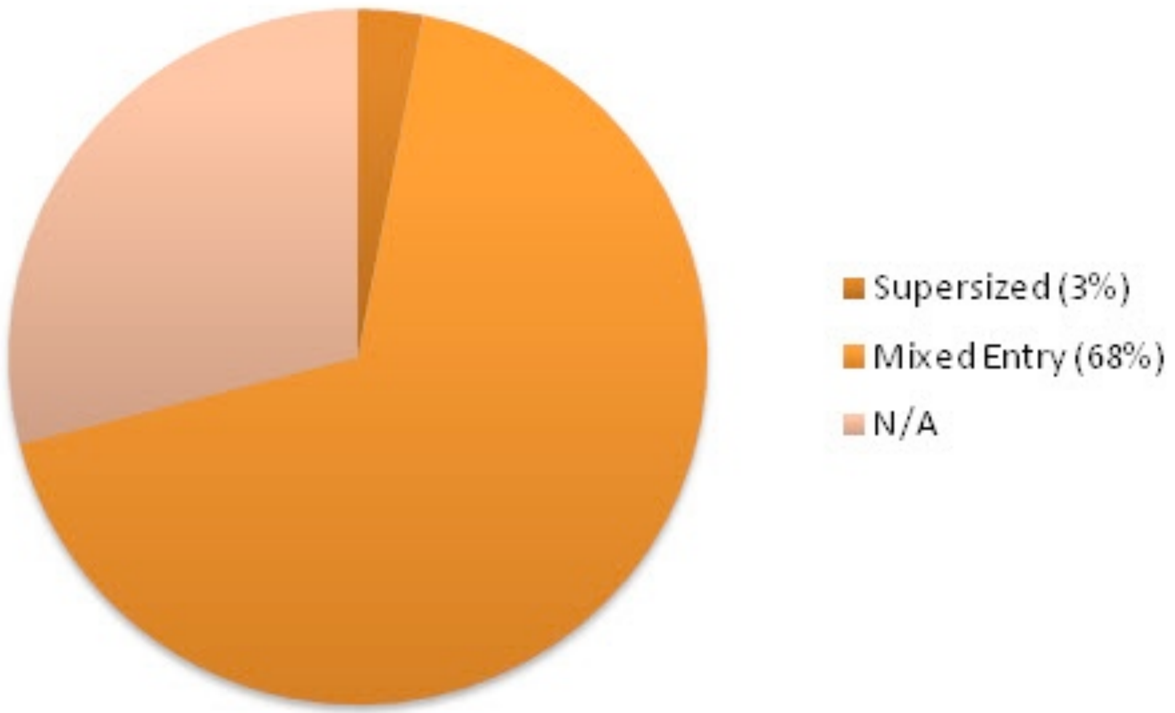
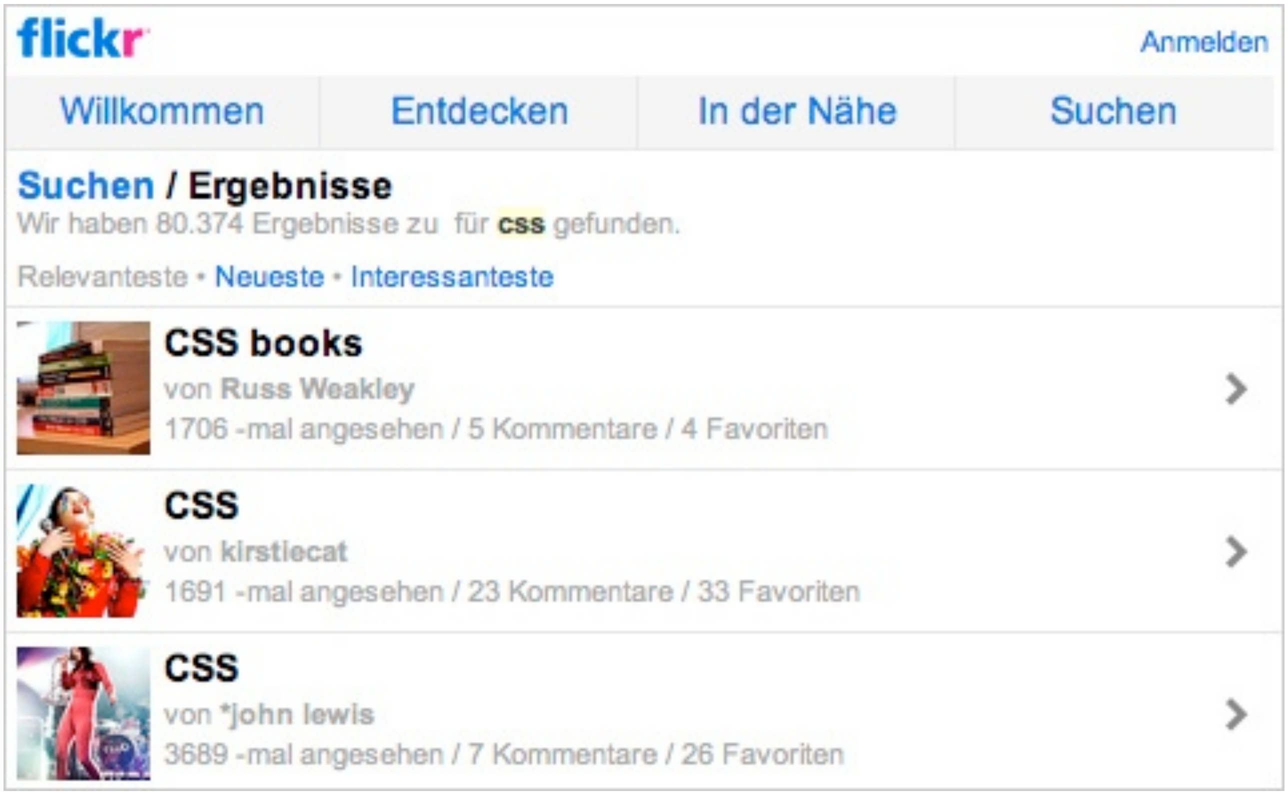


Figure 66: The percentage of websites with a large number of links, images, and large click regions



flickr Anmelden

Willkommen | Entdecken | In der Nähe | Suchen

Suchen / Ergebnisse

Wir haben 80.374 Ergebnisse zu für **css** gefunden.

Relevanteste • Neueste • Interessanteste

CSS books
von Russ Weakley
1706 -mal angesehen / 5 Kommentare / 4 Favoriten

CSS
von kirstiecat
1691 -mal angesehen / 23 Kommentare / 33 Favoriten

CSS
von *john lewis
3689 -mal angesehen / 7 Kommentare / 26 Favoriten

Figure 67: Flickr uses image links to provide a more visible click ratio, for easier usability

The results of this particular test were evenly spread, showcasing a good number of websites using large- to medium-sized links that ensured navigating could be accomplished without an accident occurring. While this trend was especially present in more image-oriented websites (as linking often occurred in image thumbnails), there was also an interesting trend in which Asian or Russian websites were more likely to have smaller click regions (fewer characters in words also reduced the click region further).

BLOCK PLACEMENT RATIOS

The next test looked at the visual layout of the page and examined how the design was broken down into either physical or visual blocks of information. It's worth stating that all of the websites measured used a single column layout, but that the way in which links and segments of content are split may give the visual indication of separation, which was worth examining further. As mobile websites require a lot more scrolling than desktop websites, the separation of blocks becomes more critical to visual identity.

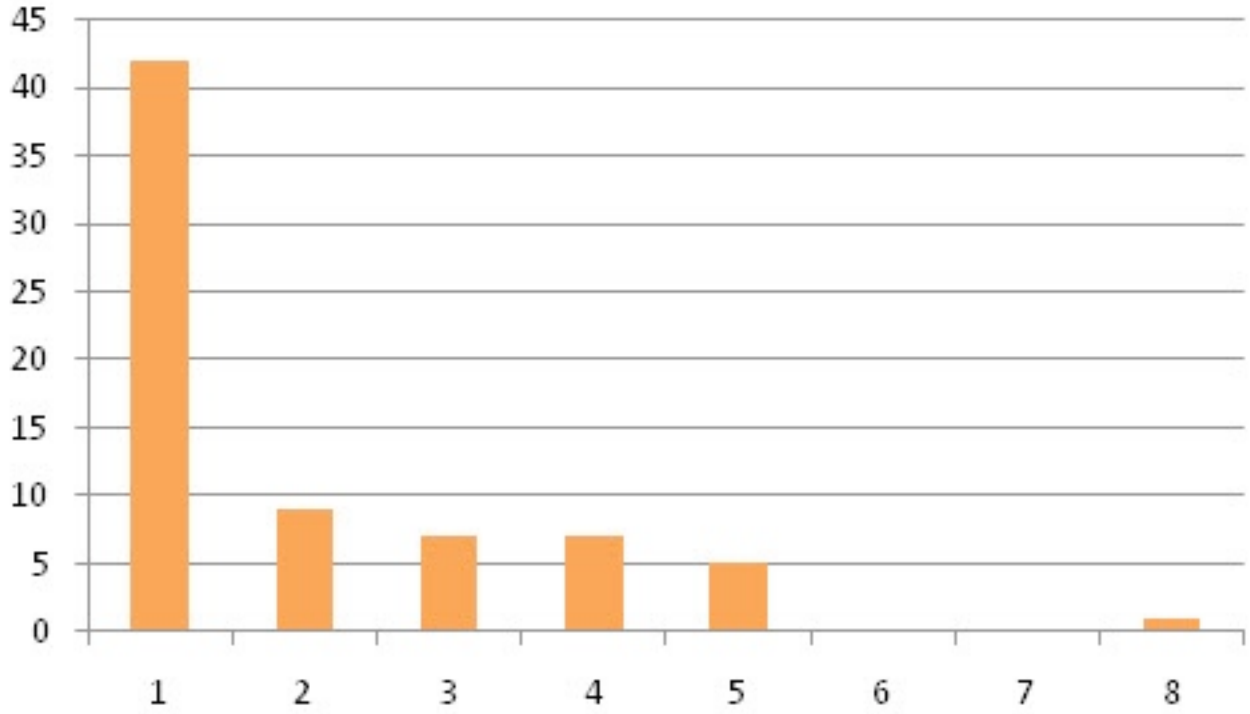


Figure 68: Distribution of horizontal elements (columns or table cells) within the viewport

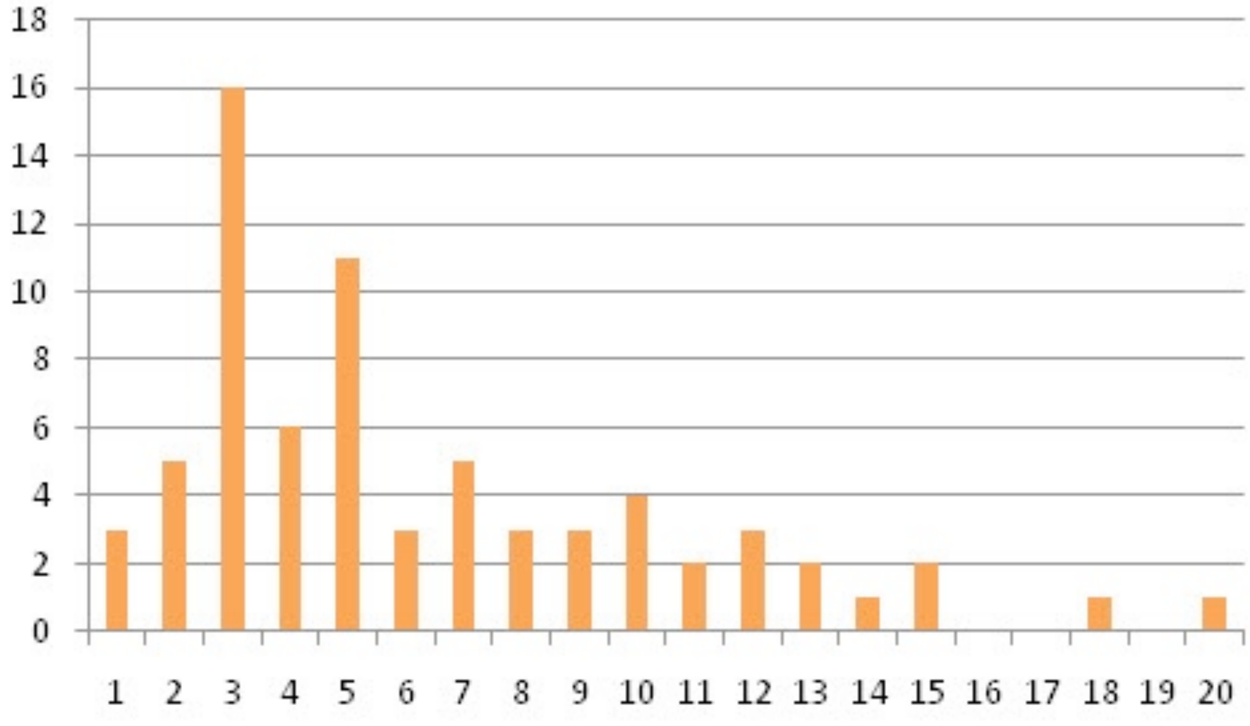


Figure 69: Distribution of vertical panel segments (headings or separated rules) per page

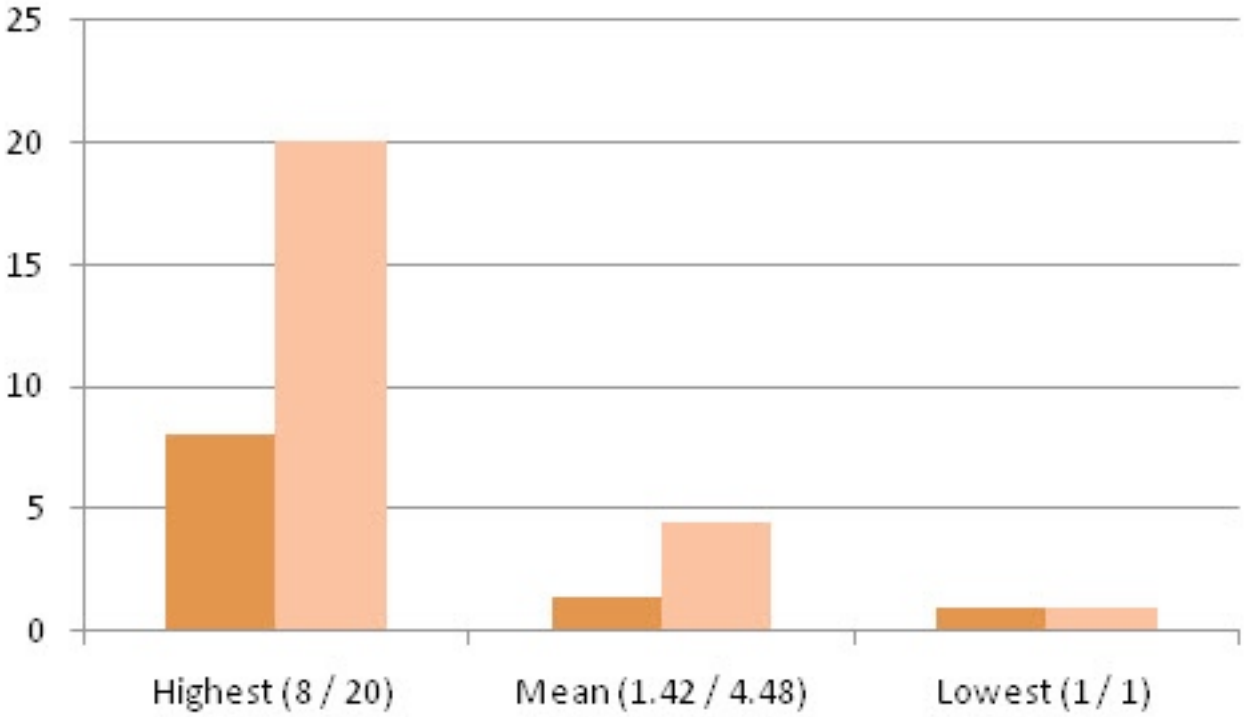
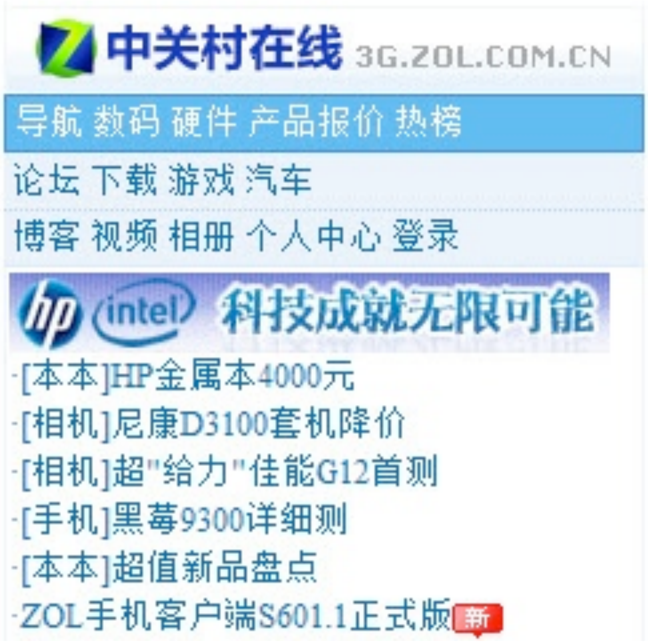


Figure 70: The highest, mean, and lowest range of blocks, both in column and panel distribution



中关村在线 3G.ZOL.COM.CN

导航 数码 硬件 产品报价 热榜

论坛 下载 游戏 汽车

博客 视频 相册 个人中心 登录

hp intel 科技成就无限可能

·[本本]HP金属本4000元
·[相机]尼康D3100套机降价
·[相机]超"给力"佳能G12首测
·[手机]黑莓9300详细测
·[本本]超值新品盘点
·ZOL手机客户端S601.1正式版新

Figure 71: The Zol website makes use of visible blocks, which separate content by heading

Interestingly, while the majority of websites held no horizontal panes (including table cells, navigation links, and other side-by-side elements), a number of websites defied convention and had anywhere between two and eight points of reference within a single line on the viewport. This visual separation was also backed up by how a number of websites had vertical breaks, which split the content into separate sections. Most mobile websites' readability was increased due to such content organization.

SCROLLING EXPERIENCE

The final test was included as a representation of how scrolling is experienced within the mobile website. The variables analyzed included the number of fingers required to initiate a scroll (as when content scrolls using an overflow, two fingers rather than one are required), and whether the window supported the fancy flourishes of jQuery, JavaScript, or CSS3 in providing paneled scrolling–where panes of content would rotate at specific intervals, or sections could be scrolled by request.

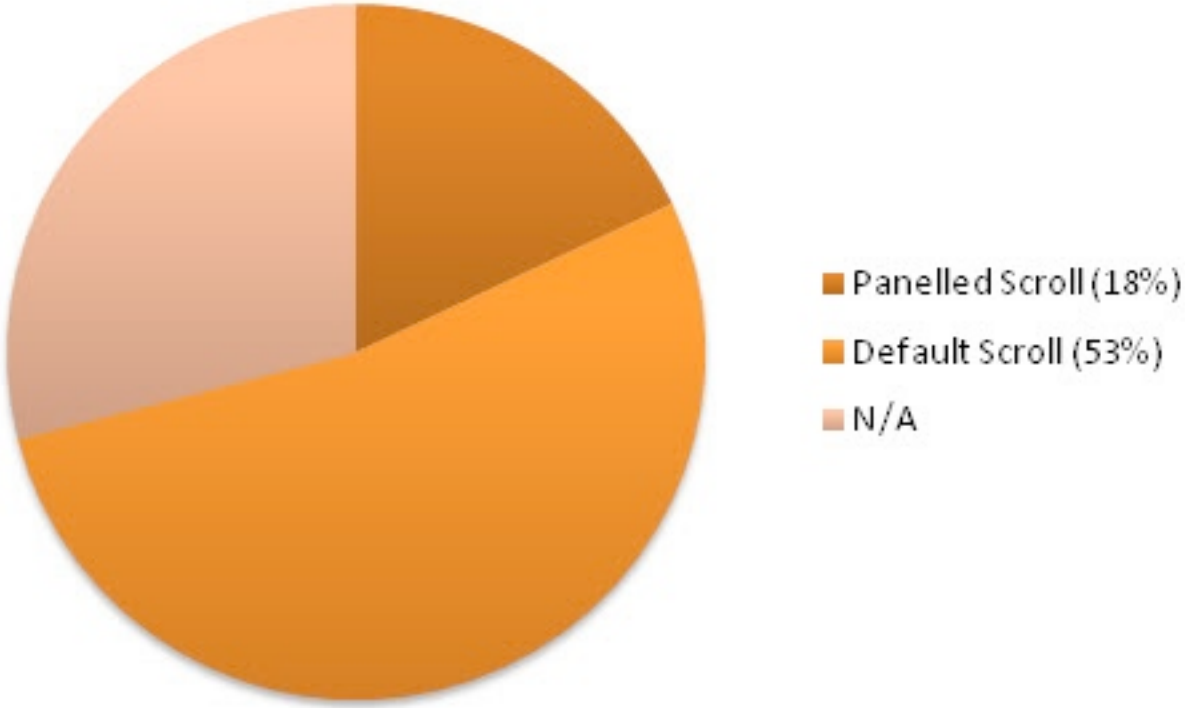


Figure 72: The percentage of mobile websites using paneled navigation over conventional scrolling

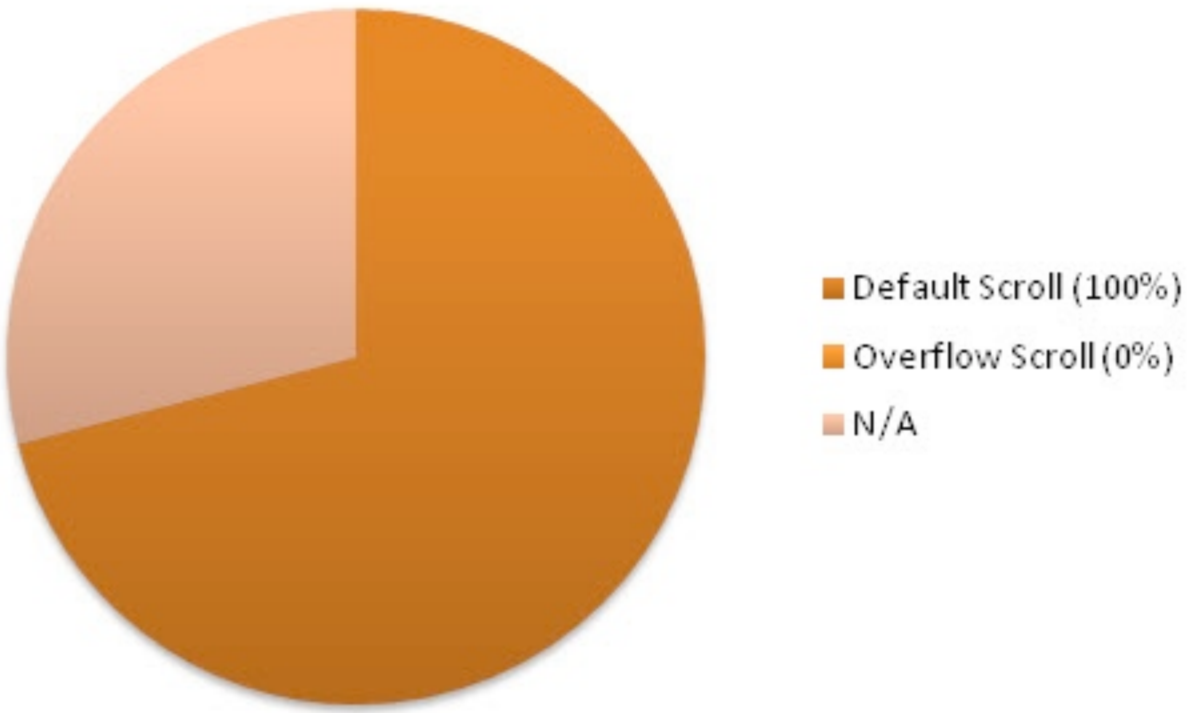
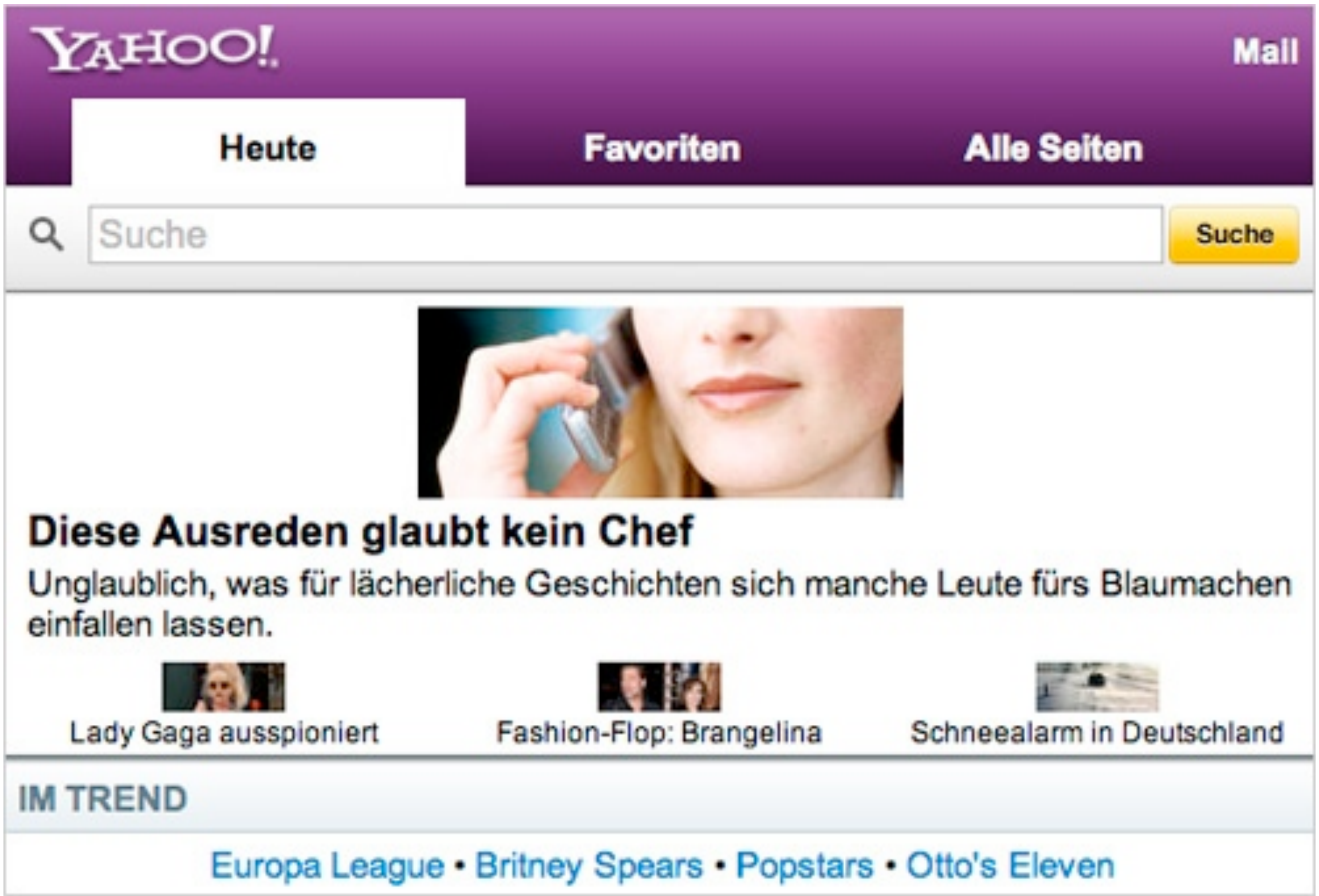


Figure 73: The percentage of overflow-based scrolling in preference to a full page scrolling effect



Diese Ausreden glaubt kein Chef

Unglaublich, was für lächerliche Geschichten sich manche Leute fürs Blaumachen einfallen lassen.

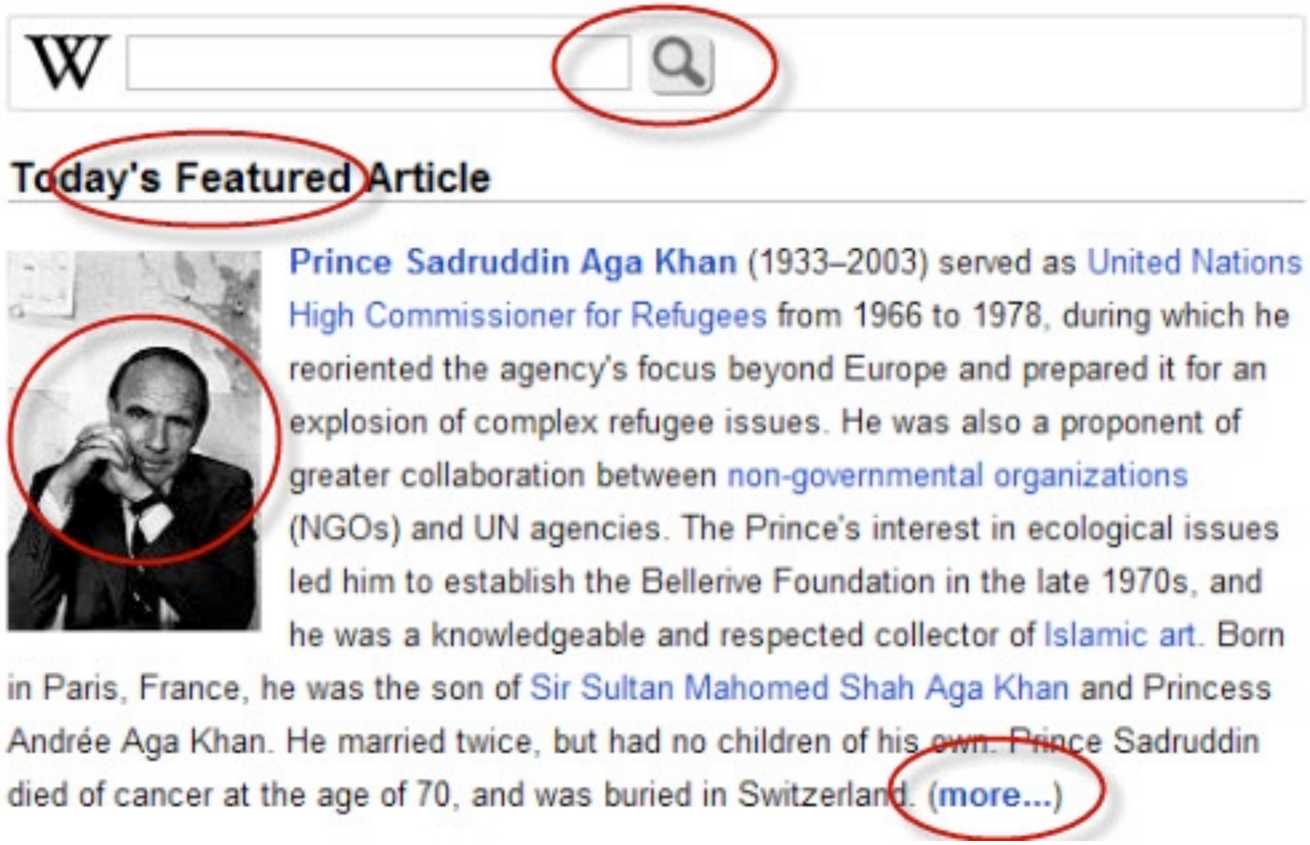
Lady Gaga ausspioniert
Fashion-Flop: Brangelina
Schneealarm in Deutschland
IM TREND
Europa League • Britney Spears • Popstars • Otto's Eleven

Figure 74: Yahoo splits it's content into several panels which can be activated on-demand

The results of this test show a general lack of paneling in general (which is representative of the lack of "flourish" scripting in most mobile websites). This may have been partly due to the nature of the websites being studied (portfolio or showcase websites are more likely to have paneled mechanisms). What is interesting however is that to increase the simplicity of the designs (as none of the sites used overflow text), they also maintained a liquid layout to reduce the complexity of required scrolling.

Going Beyond the Basics

While the sample used did constitute a wide array of Web presences, it’s worth noting that different niches may produce varying levels of results. For example, a study of the same variables upon websites which primarily act as a designer’s portfolio (as denoted in the results) will obviously account for different usage needs, and the experience may therefore vary. That said, the trends which the websites employ do bear a resemblance to many others (and therefore can be deemed as reliable).



Today's Featured Article

Prince Sadruddin Aga Khan (1933–2003) served as United Nations High Commissioner for Refugees from 1966 to 1978, during which he reoriented the agency's focus beyond Europe and prepared it for an explosion of complex refugee issues. He was also a proponent of greater collaboration between non-governmental organizations (NGOs) and UN agencies. The Prince's interest in ecological issues led him to establish the Bellerive Foundation in the late 1970s, and he was a knowledgeable and respected collector of Islamic art. Born in Paris, France, he was the son of Sir Sultan Mahomed Shah Aga Khan and Princess Andrée Aga Khan. He married twice, but had no children of his own. Prince Sadruddin died of cancer at the age of 70, and was buried in Switzerland. (**more...**)

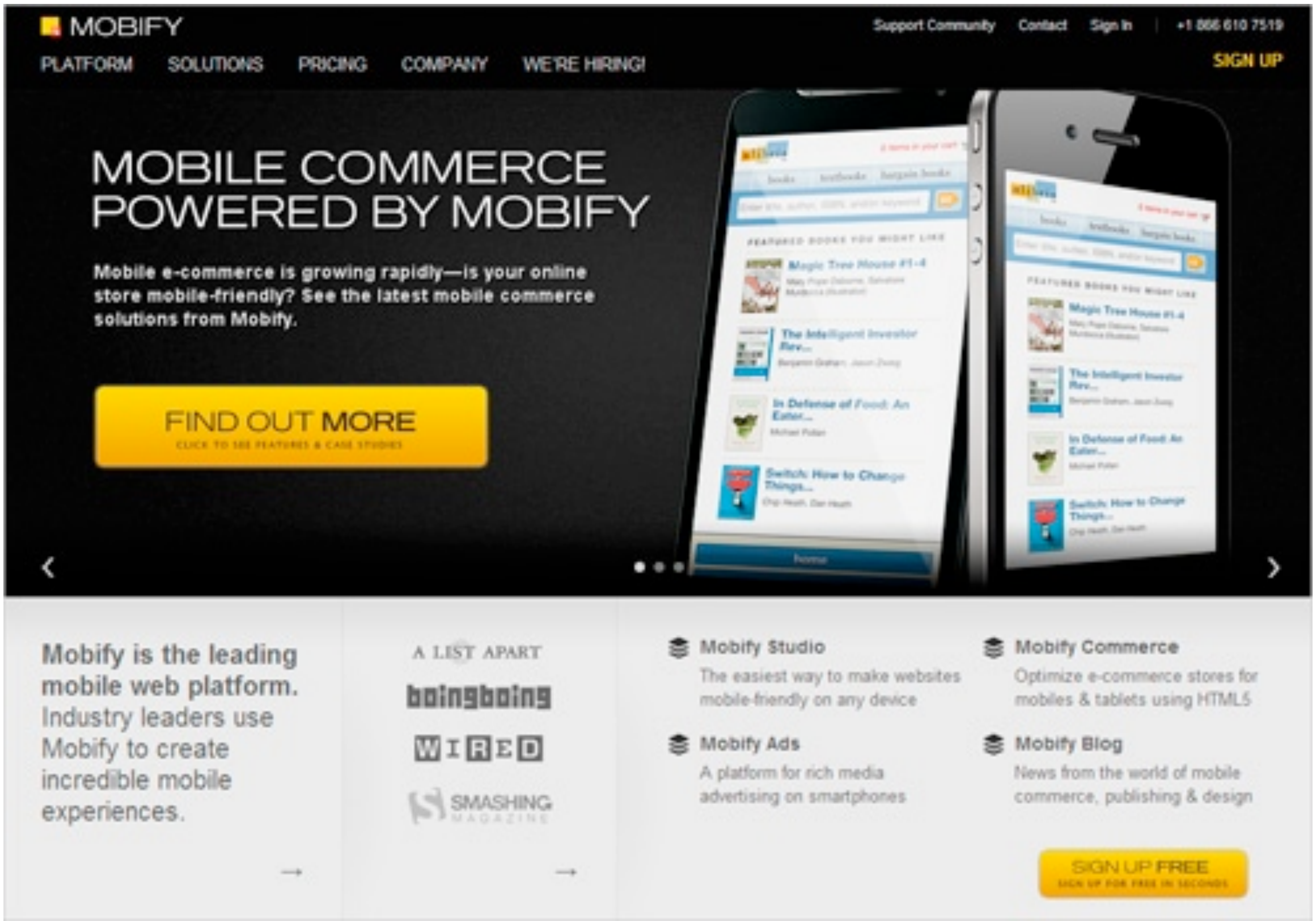
Figure 75: Different websites will follow different trends, but they still have comparative similarities

To ensure that the results were not simply a result of the top 100 websites' own popularity bias, an extra 10 websites were browsed too (their results weren't logged as it was an exercise of conformance) in order to see the accuracy of the top 100 findings. Within these were a mixture of varying website types, but the overwhelming majority of websites followed similar practices, or made changes which would logically make sense under the usage scenario. Therefore, the trends do seem to apply consistently.

> ***Note:*** *In addition to the above, it's worth stating that a couple of the websites within the top 100 listings were either subdomains, or international versions of a website which has already been mentioned. While this could prove an issue for repeat conventions, the results varied enough to qualify their inclusion.*

COMMON FACTORS AT PLAY

The common factors which were widely implemented included the trend of using scripts to redirect mobile-friendly users, in preference to giving visitors a choice (which is interesting, as it constitutes a double-edged sword in terms of usability). In addition, the trend of subdomains also followed a well-trodden pattern in order to be recognized easily by end users. A final example, which showcases the common factors, is the unfortunate issue that few of the websites' code validated!



MOBILE COMMERCE POWERED BY MOBIFY

Mobile e-commerce is growing rapidly—is your online store mobile-friendly? See the latest mobile commerce solutions from Mobify.

Mobify is the leading mobile web platform. Industry leaders use Mobify to create incredible mobile experiences.

Mobify Studio
The easiest way to make websites mobile-friendly on any device

Mobify Commerce
Optimize e-commerce stores for mobiles & tablets using HTML5

Mobify Ads
A platform for rich media advertising on smartphones

Mobify Blog
News from the world of mobile commerce, publishing & design

Figure 76: Some websites actually generate their mobile layouts using automated tools like Mobify

MODELS FOR OTHER GENRES

Expanding this study across a different cohort of website types may have varying differences as to how trends are implemented. Examples include the increased use of CSS3 and “flourishes” in portfolio websites, a reduced number of mobile apps on small business websites, and increases in the amount of required authentication if the study were applied primarily to social networks. In addition, if a website is more content-focused (like Smashing Magazine), an increase in file sizes will be an obvious side effect.

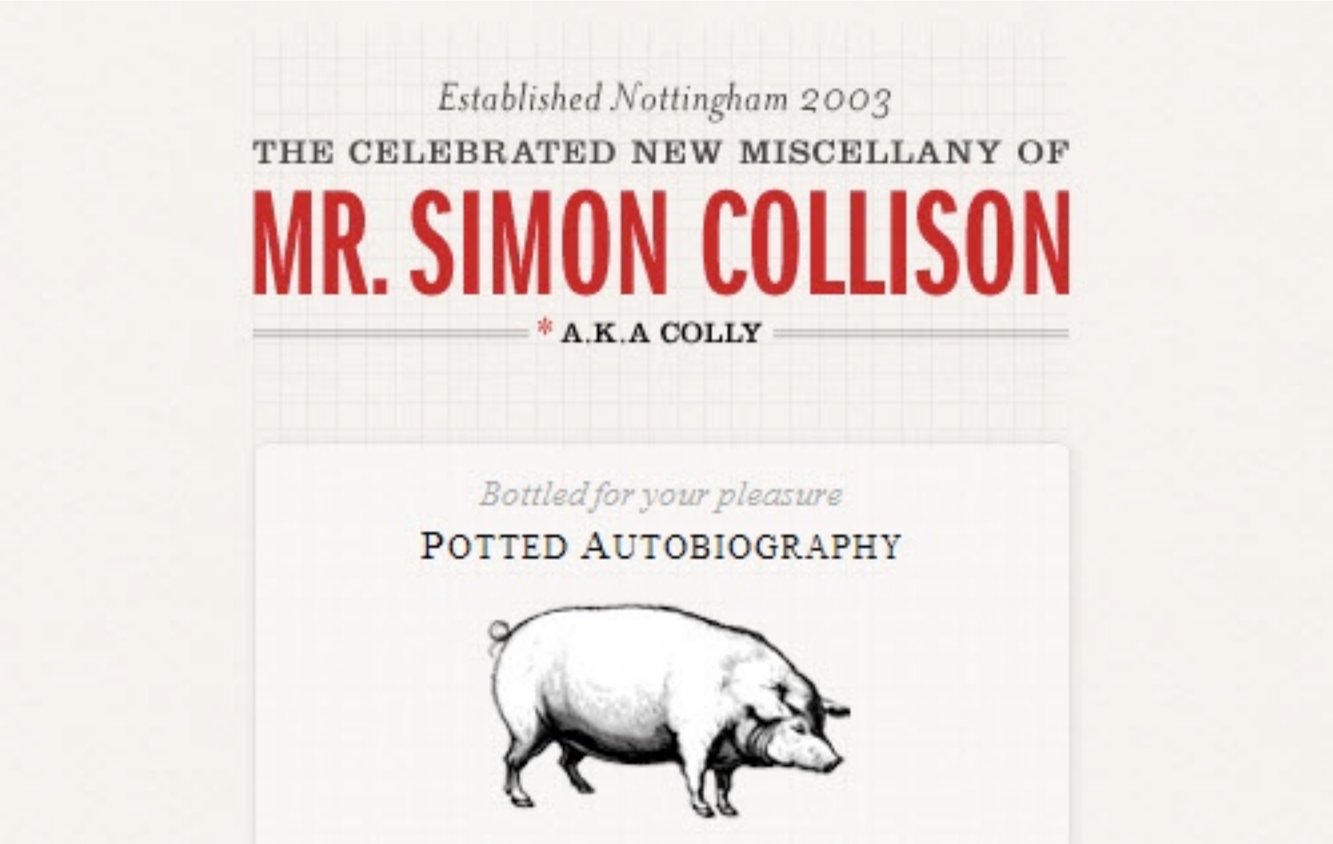
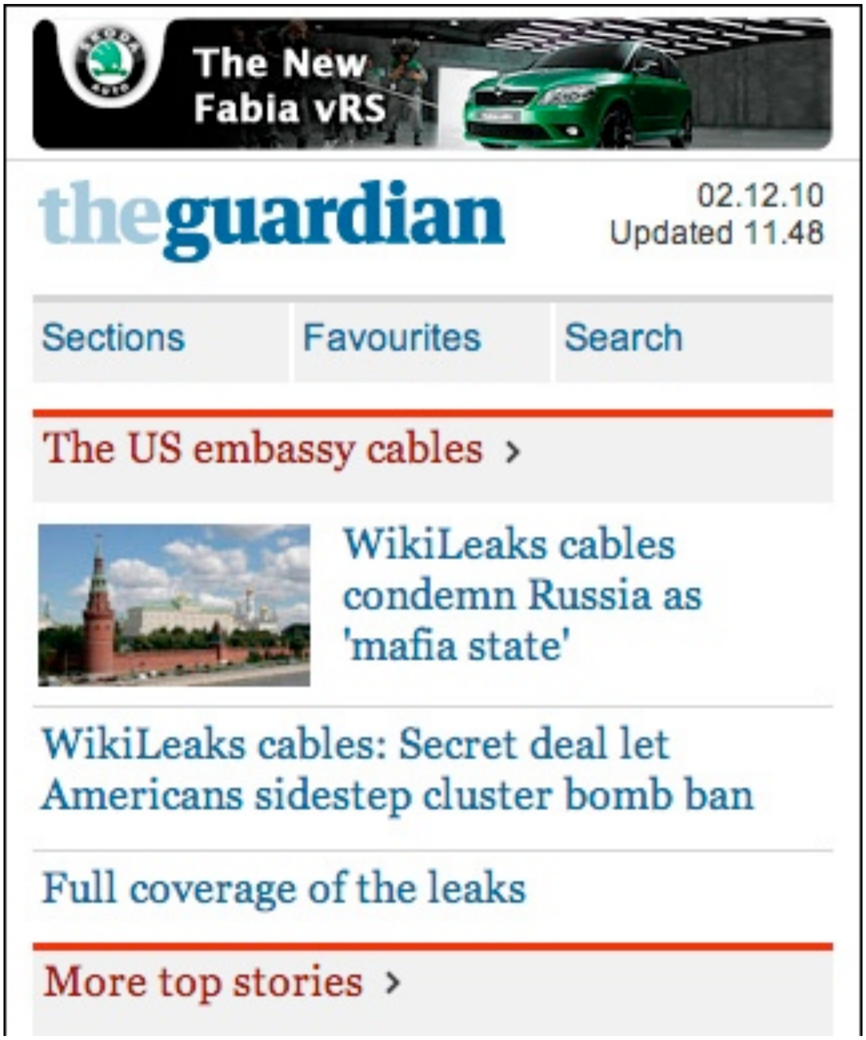

Established Nottingham 2003
THE CELEBRATED NEW MISCELLANY OF
MR. SIMON COLLISON
* A.K.A COLLY
Bottled for your pleasure
POTTED AUTOBIOGRAPHY

Figure 77: Professional designers such as Simon Collison may use CSS3 media queries to account for display sizes

POTENTIAL IMPROVEMENTS

Taking this study forward, improvements could be made. While the websites were tested using a range of real mobile devices including an Apple iPhone4, a Blackberry device, a Nokia device, a phone using WinMo, and a couple of phones using Android, a greater level of compatibility could be established if an increased number of devices were used (excluding emulators). In addition, a larger range of websites may boost the general accuracy of the results (such as if all 1,000 websites were tested).



theguardian

02.12.10
Updated 11.48

Sections | Favourites | Search

The US embassy cables ›

WikiLeaks cables condemn Russia as 'mafia state'

WikiLeaks cables: Secret deal let Americans sidestep cluster bomb ban

Full coverage of the leaks

More top stories ›

Figure 78: Additional studies could be undertaken on particular niches, like newspaper websites. Example: The Guardian.

The Future of Mobile Design

As interesting as these results are, hopefully they will inspire you not just to follow design trends, but to seek them out. While the Web may still be in its infancy, mobile design is literally only a few years old. This means that trends will likely evolve ever more rapidly, as more usability studies and additional research are carried out to determine how user needs are being catered for. The future of mobile Web design is an up and coming industry, and we need to pay it plenty of attention.

As an increasing number of handheld devices appear (with different renderers and viewport sizes), the way we design is being affected by frameworks like jQuery, the advancement of standards such as HTML5 and CSS3, and the way applications are gradually being pulled toward the cloud. While this study showcases that rudimentary trends exist for mobile designs (such as single column layouts), it will be an interesting next few years as more designers account for the booming mobile audience.

How To Market Your Mobile Application

Michael Flarup

App Store is a competitive environment. Against more than 140,000 apps, all screaming for attention, how do you make sure your app gets its time in the spotlight? What does it take to get good media coverage? How do you get people to talk about your app—and, ideally, how do you get them to buy it and show it to their friends?

Following the simple rules laid out below, you will increase your chances in the battle for fame and glory. These tips might seem rudimentary or in-your-face obvious, but they are so often neglected in the heat of the moment.

Be Unique

One of the easiest ways to stand out in the App Store is to create an app that is unique. Sure, that makes sense. Yet still thousands and thousands of apps are uninspired, shovelled out by tired developers looking for a quick buck.

If you want to stick it to the man, make sure that you are either:

1. The first developer in your category of product, or
2. Reinventing the existing category with something unique.

If you’re just improving something that’s already available, your battle to market it will be uphill.

SPIN AN EXISTING CATEGORY

At this point in the history of the App Store, very few apps create new categories. So unless you’re sitting on a revolutionary new idea, focus your attention on a unique spin of an existing category. So many things can be re-imagined with little effort. Look at your competitors and flick on your child-like consumer filter. What cool feature for this category is missing? How can you take advantage of the iPhone’s interface, accelerometer, GPS or multi-touch functionality to create a package that delivers a unique experience in this category?

A unique feature will make your app stand a head taller in the crowd and raise eyebrows. And that’s exactly the effect you want if you intend to sell apps in the App Store.

- Think, plan and build with the intention of creating something unique. From the conceptual drafts to the final marketing, keep iterating the unique aspects of your product.
- Ask yourself if you are merely improving on someone else’s idea. If it already exists in the app store, the battle to market it will be uphill.
- Try some shortcuts to create something unique, such as mixing categories; thinking of new ways to use the accelerometer, GPS, proximity sensor and multi-touch gestures; storytelling; etc.
- If you’re competing in a saturated market, do the exact opposite of the leader.

Be Tweetable

Getting people to talk about your app is imperative for success. The more people talk, the more exposure your app will get, which will hopefully translate into sales. If your app is unique, you're halfway there—people will talk about it just because of its uniqueness. But how do you encourage people to start up conversations about your product?

LEARN TO PITCH

I'm sure you've pitched your app to at least a dozen co-workers and puzzled family members. You know the ins and outs of your elevator speech, the highs and lows, the big sells of your product and the hard-to-understand parts. If you want your app to succeed, you will need to teach that pitch to the rest of the world.

BE INTERESTING

Make the conversation about your app easy and engaging. Make it so that people want to tweet about it. Tweetability—if no one has yet, I'm trademarking that word—refers to how well a product or message would move on Twitter. The Twitter network, with its millions of users, has a particular personality and disposition. Despite the diversity of people using the service, talking about it like a homogenous mass still makes sense in many ways. Some of the most successful apps are easily shared through social media. Imagine the twittersphere chattering in chipmunk voices, "Hey, guys. Check this out!" Instantly gratifying app + high tweetability = free exposure.

Even if your app isn't instantly gratifying or playfully humorous, you can still compose a tweet that is highly tweetable. Just think of what you would retweet yourself. How would you sell your app in 140 characters?

- Play to your strengths. Write good copy. And have a solid, useful and attractive landing page.
- Find the human angle. Are there any amusing and beneficial reasons why people would use your app?
- Have a memorable tagline. Sum up your app's purpose in one line.

Cater To Blogs

Social media and the blogosphere are not isolated from each other. Like ripples in a pond, the more people tweet about your app, the more likely you'll hit a big blog.

Review blogs and tech websites are part of the App Store's eco-system, and while the exact effect they have on sales is debatable, the traffic and buzz they generate are worth pursuing.

THINK LIKE MEDIA

To get good media coverage, you need to think like the media. How good a story is your app? Obviously, the law of uniqueness makes a difference here, but your app should also be easy to write about. First, provide a free press package that anyone can download. Supply people with the material they need to talk about your app. Give them a high-res version of the icon, screenshots and press-related texts.

Don't be stingy with the promo keys either—in fact, dispense them liberally. Promo keys are cheap marketing collateral and a way for you to put your app in the hands of peer leaders. Throw the keys at your favorite blog, and invite them to give some away for free in a raffle. If you can find a category-specific blog, you've got a direct line to your target customers. It's a great way to reach a new audience and strengthen your relationships and reputation.

BLOGS ARE LIKE KIDS IN A SCHOOLYARD

While they may not want to hear this, blogs are a bit like kids in a schoolyard. If you can get the cool kids to talk about you, chances are that other blogs will pick up the story and throw you on their front page. Getting on review and media websites is vital to your marketing success, because they are less transient than tweets. Reviews stay there and bring in traffic for months.

- Give out promo codes to blogs without hesitation.
- Have an extensive and easily accessible press package.
- Don't be afraid to ask individuals to endorse your app.
- Try to crack category-specific blogs. If you're making a wine app, contact wine blogs.

Control The Hype

App sales thrive on hype. Learn to control the hype, and you will have mastered the product launch. Hype will always be partly out of your hands, but the rules mentioned above will help you put things in motion. But hype will amount to nothing if it's for a poor product. While there is truth to the saying that there is no such thing as bad publicity, hype can backfire and harm your efforts to generate hype in future.

HYPE EARLY

Start hyping early. If you know you have a unique product, let people in on the secret before the launch. Having an interesting "Coming soon" website can do this, by building a mailing list and getting Google juice for your domain.

MAKE YOUR WEBSITE GREAT

Needles to say, your app should have its own website. To make any of the rules above work, you will need a point of reference, somewhere to send the masses. Make the website interesting; show the app in action, and think outside the box. Make the website an extension of your app, and you will have yet another great tool in your marketing toolbox.

LAUNCH BIG

When you launch, make it big. Send out the triumphant newsletter, and hit all social media. Have you or your team write up blog posts, and pull every lever and handle in your network. Hype is all about critical mass: the first wave you set in motion will give you instant feedback on how to adjust your hype machine.

Maintaining hype is all about introducing new venues in which to exhibit your app. Get a steady stream of review websites to cover your app. Give away promo keys on Twitter, and serve new content on your website. Obviously, if you can get into the “What’s hot” or “New and noteworthy” sections of the App Store, you’ve made it far.

In the end, hype is part luck and part skill. The best way to balance the two is to keep asking yourself whether you can do anything else to add value, mystery, polish or spin to your product. Rely on your own judgement: what would excite you about this app if it were made by another developer?

- Give out promo codes on Twitter and in the blogosphere.
- Run contests related to your app. Give away prizes that make sense for your category.

- Boost popularity by timing the launch of your app to coincide with a live event or trending topic. Climate-related apps spiked around the COP15 Climate Summit in Copenhagen.
- Release your app with a big bang. Hold an online or live event. Attract visitors in creative ways, by building a game or puzzle or just throwing a contest or giveaway.

Example: Being Awesome In A Saturated Market

To illustrate the application of these rules, let's take a play-by-play look at one successful app. For the sake of convenience, let's just call it "Awesome app."

Awesome app is a weather-forecasting app. This is a classic scenario: a re-thinking of an established category. I can't think of a more tired and saturated market than weather apps, making this an excellent example of being able to re-invent and compete if we have the right frame of mind.

UNIQUE SPIN

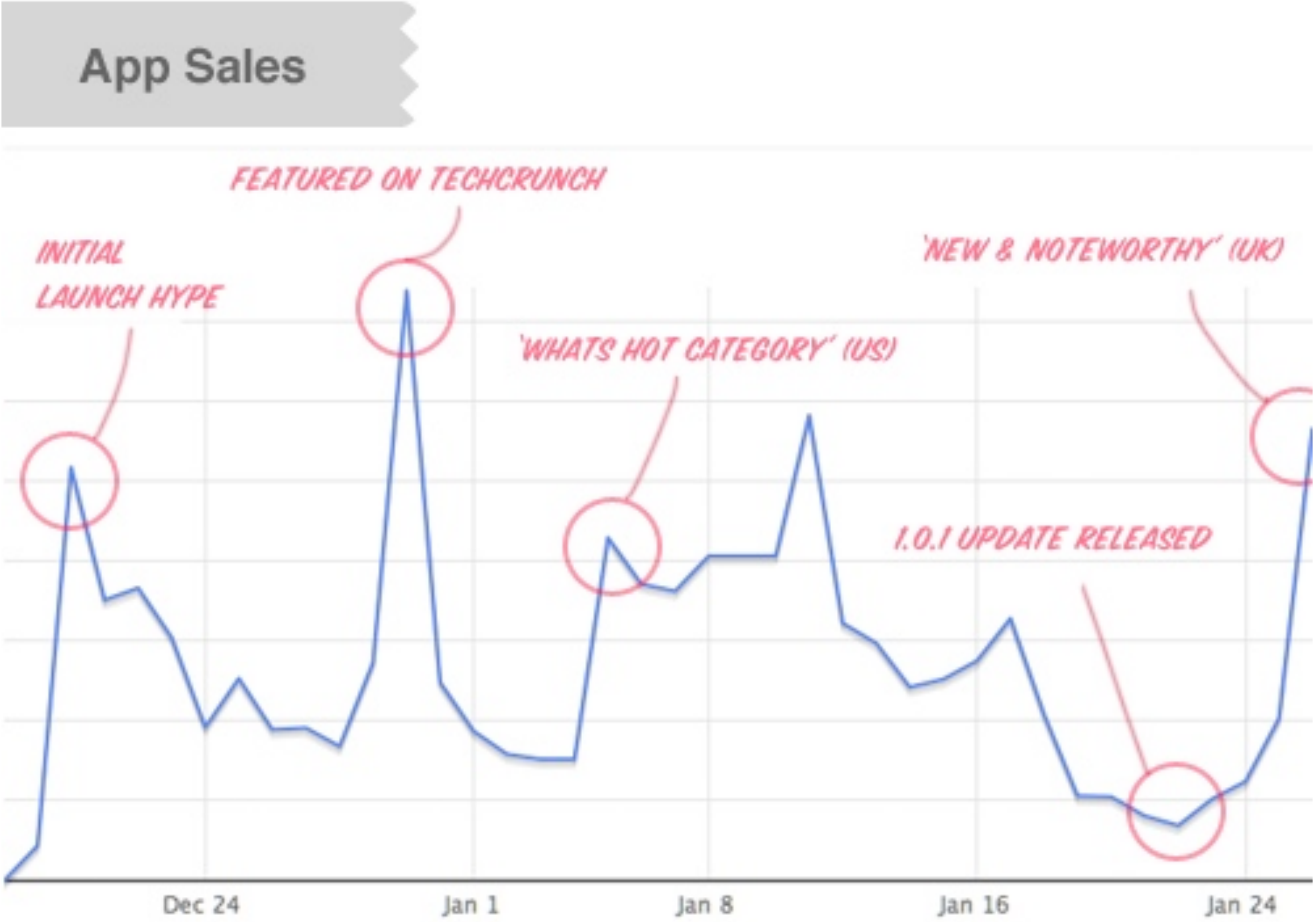
The Awesome app reverse-engineers the trend of offering up increasingly detailed and advanced weather data. Instead, it trims down functionality and focuses on the very playful and human idea of exploring the weather visually, by swiping through a virtual forecast. It builds uniqueness right into the very concept and goes in the opposite direction of the market leaders.

EARLY HYPE, BIG LAUNCH

Prior to launch, the website for Awesome app presents a "Coming soon" page that collects close to a thousand confirmed emails. A teaser video of the interface generates some buzz and earns the app a nomination in the App Star awards. The app launches at the end of December 2009. The release newsletter goes out; a more elaborate version of the website, with video and screenshots, goes up; and the developers make as much noise as they possibly can in their networks.

REVIEW WEBSITES

As soon as sales get a lift from the early launch hype, emails are sent out to various review websites offering promo keys. Reviews started flowing in, and chatter about the app is monitored on Twitter, where developers offer help and follow up on questions. A "Making the app" video is posted that gives existing customers something to enjoy (and that humanizes the team), highlighting user recommendations.



The website for Awesome app gets some wind of its own by being featured in various design blogs for its modern use of CSS animations, contributing hype that doesn't have anything to do with the app itself.

PICKED UP BY LARGER WEBSITES

A week and a half after launch, larger websites such as TUAW started showing interest. And coverage peaks with a TechCrunch article, which ripples out to LifeHacker and other major websites. More than a month in and we're still seeing continued interest in the app; it has gathered hundreds of five-star reviews in the App Store and has been featured in both "New and noteworthy" and "What's hot."

WHAT WORKED?

What worked for Awesome app was a combination of the marketing rules discussed above:

- It was sufficiently unique in a crowded market to spark interest and be seen as a "good story."
- The idea of a "visual weather forecast" was easy to convey and was presented in a way that gave it high tweetability.
- It was completely the opposite of what leading competitors were doing.
- The team started hyping early with a "Coming soon" page. It was appealing enough for people to tweet about it, and it eventually attracted visitors not only because of the app but because of the design of the website.
- A press package with everything you could want was freely available on the website, making it easy for blogs to write about the app.

Parting Thought

Not a single dime was spent on marketing it, yet the Awesome app reached tens of thousands of people. If you have a unique product and apply some of the ideas above, you too can secure free exposure for your beloved app. It's a rather democratic and honest process because you are required to re-invent apps by adding unique features. Marketing then becomes all about making it easier and more interesting for people to talk about and share your creation.

As with most other things in life, there’s no surefire way to create a successful app. But keeping in mind some of the things we’ve talked about here—both at the conception and the execution stage—will put you in a position to build awareness of your application much more easily.

A Foot On The Bottom Rung: First Forays Into Responsive Web Development

Gavyn McKenzie

Responsive design is the hottest topic in front-end Web development right now. It's going to transform the Web into an all-singing, all-dancing, all-devices party, where we can access any information located anywhere in the world. But does responsive design translate well from the text-heavy Web design blogosphere to the cold hard reality of commercial systems?

Rumors came through our office grapevine that management was looking to revamp our mobile presence. There was talk of multiple apps being built externally that could be used on some of the major mobile devices. Our team had been getting familiar with responsive design and put forward a proposal of doing away with device-specific apps and developing a single responsive website that could be served to both desktop and mobile users. After a few hasty demos and prototypes, the idea was accepted and we began work.

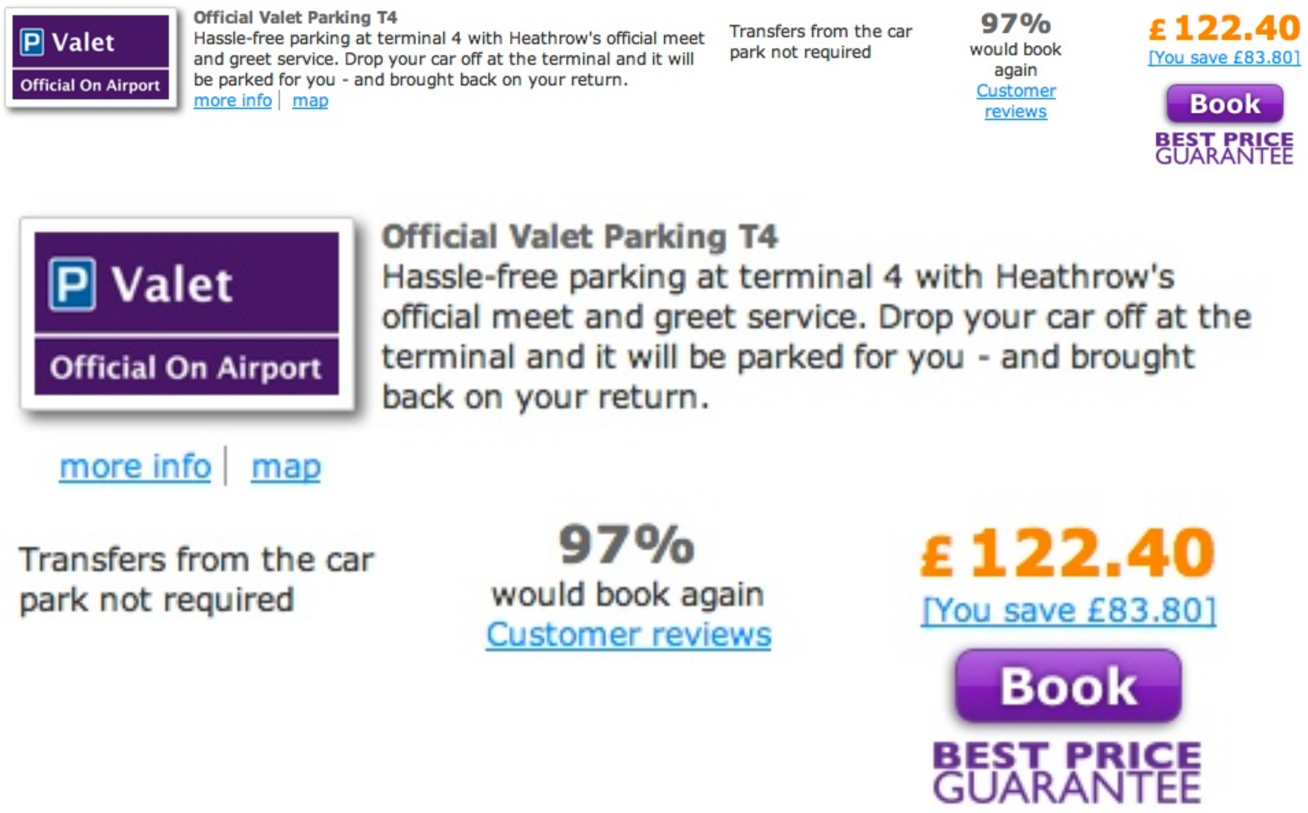
The brief: make our current website, Airport-Hotels.uk, responsive while retaining the existing layout for users on browsers of 1000 pixels and up.

The following is what we picked up along the way.

Starting With Desktop Is OK

The general consensus now seems to be “mobile first.” I agree. Starting with a single(ish)-column mobile website is the easiest way to get your CSS off to a great start. However, we use an external design agency, so the time and cost of a new mobile-first design was not feasible. It was left to the front-end developer to translate the existing design onto screens of smaller dimensions.

The solution was to break up the website into smaller blocks (or nuggets), which could then be positioned differently as the browser’s width increased. This led to our first base media query, which contained the main branding elements, with minimal layout information. Because the nuggets were of a fairly fixed size, we had a foundation for creating a grid for each of our major media queries. Anything that wasn’t deemed to be a “nugget,” such as a larger block of text, would be responsive and fill in the gaps that the nuggets couldn’t.



Official Valet Parking T4
Hassle-free parking at terminal 4 with Heathrow's official meet and greet service. Drop your car off at the terminal and it will be parked for you - and brought back on your return.

Transfers from the car park not required

97% would book again

£122.40
[You save £83.80]

Official Valet Parking T4
Hassle-free parking at terminal 4 with Heathrow's official meet and greet service. Drop your car off at the terminal and it will be parked for you - and brought back on your return.

more info | map

Transfers from the car park not required

97% would book again
Customer reviews

£122.40
[You save £83.80]
Book
BEST PRICE GUARANTEE

While this method is not as good a practice as “mobile first,” it does have the advantage of being faster and cheaper than a full redesign. And you pick up great knowledge along the way for when resources do become available for something more substantial.

Less Is More

When getting your feet wet with media queries, you’re tempted to go all out, but do you need to? Theoretically, you could serve a completely different design to each “major” screen resolution. While this would be spectacular and self-satisfying, maintaining it would be a nightmare. We ended up using the default media queries in Andy Clarke’s 320 and Up framework, containing four breakpoints (1382 pixels was not in the brief). Looking back now, we could have removed at least one of those queries, possibly two.

We've been gathering statistics in the weeks since the website's release, and by far the majority of our customers are running browsers either of 320 × 480 pixels or on full desktops. We could hit over 85% of our user base by focusing on these resolutions, while cutting down on development time and maintenance.

This was especially evident on our availability page, which easily contains the most information of any of the pages in our booking flow. In the end, rather than try to serve the perfect design to each device width, we moved much of the CSS for the largest media query to the size below: less maintenance, less fuss, and more time to work on the UX (and, importantly to the business, to make bookings).

Ability Sniffing Is Not Enough

When I first saw tools like Modernizr, I thought they would change everything. I suppose they have, but don't rely on them too much. Mobile browsers have more inconsistencies than any desktop I have ever seen. Even WebKit-based browsers can render things completely differently. With debugging tools at a minimum, it's like we've been thrust back into the pre-developer toolbar era of IE bug fixing. Luckily, that's one of my favorite things.

Exploring this strange new world of bugs became one of the major aspects of the project. A few of my favorites are highlighted below. Hopefully, they won't trip you up.

CSS COLUMNS

I love CSS columns. I had been wanting to use them for a while; but, other than small personal projects, nothing with appropriate content came up. While trying to work out the best layout for our website on a 320-pixel device, I realized that, rather than generating columns using floats or inline blocks, we could reduce the layout CSS to just a few lines by creating CSS columns. With most major mobile Web browsers being based on WebKit and Opera, this seemed to be a fairly reasonable solution and appeared to lay out everything perfectly. Great!

Here is the original code for the 320-pixel media query:

```
.product {
    -moz-column-count: 2;
    -moz-column-gap: 5px;
    -webkit-column-count: 2;
    -webkit-column-gap: 5px;
    column-count: 2;
    column-gap: 5px;
}
```

And here is the updated solution (roughly speaking — the actual code was much longer):

```
.product>div {
    width: 49%;
    float: left;
    margin: 0.5%;
}
```

Unfortunately, the **`column`** specification isn't quite ready yet. On BlackBerrys and some HTC Android phones, our form elements (specifically, the buttons) became unclickable. The layout was perfect — we checked that the CSS was accepted with Modernizr, and all the links worked — and yet you couldn't click the "Book" button. Back to the drawing board with that one.

We ended up using a more standard float-based column layout.

CSS GRADIENTS

Gradients were another excellent instance of browser idiosyncrasies. We used a lot of CSS gradients in this redevelopment to replace some images. This should have saved the user's bandwidth and made redesigns and maintenance faster.

On WebOS (with a WebKit-based browser), though, CSS gradients would render as completely black unless used on a form input element. It was baffling. In the end, we figured out that it was a bug in the implementation of **`-webkit-linear-gradient`**. We've learned that the bug has been fixed in the upcoming version, so this might not be an issue in the future.

Here is the offending CSS:

```
.ppcHeader {
    background: #73bff1; /* Old browsers */
    background: -moz-linear-gradient(left, #73bff1 0%, #009ff7
100%); /* FF3.6+ */
    background: -webkit-gradient(linear, left top, right top,
color-stop(0%,#73bff1), color-stop(100%,#009ff7)); /*
Chrome,Safari4+ */
    background: -webkit-linear-gradient(left, #73bff1
0%,#009ff7 100%); /* Chrome10+,Safari5.1+ */
```

```
    background: -o-linear-gradient(left, #73bff1 0%,#009ff7
100%); /* Opera11.10+ */
    background: -ms-linear-gradient(left, #73bff1 0%,#009ff7
100%); /* IE10+ */
    background: linear-gradient(left, #73bff1 0%,#009ff7
100%); /* W3C */
    margin-bottom: 20px;
}
```

(Bear in mind that CSS gradients add a heavy load to the browser's rendering engine, so if you are using a lot of them, switching them off for mobile might be wise.)

JAVASCRIPT ON BLACKBERRY 5.0 AND OPERA MINI

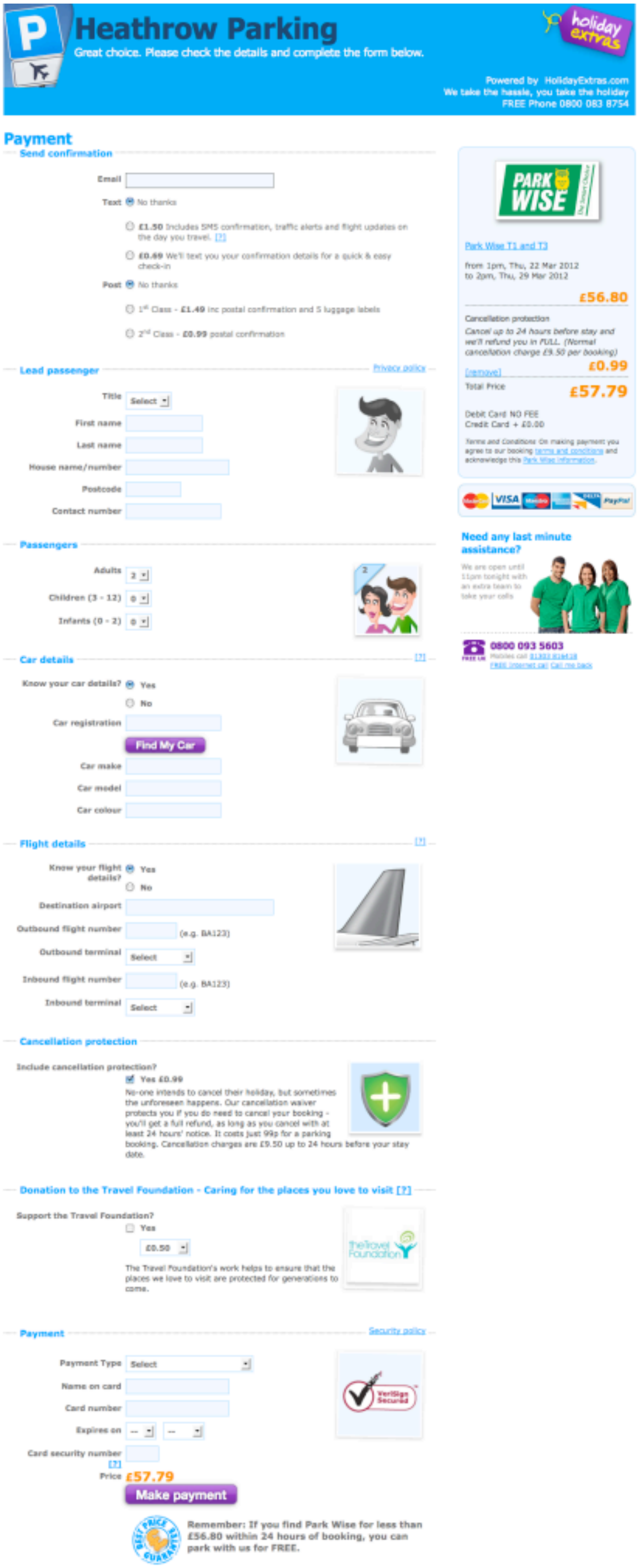
Basically, JavaScript does not work on Blackberry 5.0. BlackBerry tries, but it's so inconsistent and buggy that it's not worth it. We were reliably advised by Peter-Paul Kochs to just resort to device sniffing and to turn off any JavaScript. This is another reason to make sure your websites are progressively enhanced by falling back to non-JavaScript versions.

Meanwhile, Opera Mini works fine with JavaScript, but each of a website's pages is rendered on Opera's servers and then essentially compressed into a big image before being sent to the mobile device. This is great for the user because it can reduce bandwidth to 10% of the normal browsing experience. On the other hand, if you have **`onkeyup`** validation in your forms, this can be a problem because each call to the JavaScript would require refreshing the entire page from the server.

Forms Drop Users

This was and still is one of the major problems with mobile browsing on our e-commerce website. In order to make a purchase on an average website, the user has to fill in a lot of information: names, addresses, credit-card details, the list goes on and on. While typing on mobile has gotten much easier, navigating large forms is a frustrating and laborious process.

Our mockup payment page had 22 form inputs that needed some kind of interaction. These were required either to make a successful booking, to provide information to the product supplier after booking or for our own sales and data purposes.



Great choice. Please check the details and complete the form below.

Powered by HolidayExtras.com
We take the hassle, you take the holiday
FREE Phone 0800 083 8754

Payment

Send confirmation

Email

Text ● No thanks

○ **£1.50** Includes SMS confirmation, traffic alerts and flight updates on the day you travel. [?]

○ **£0.69** We'll text you your confirmation details for a quick & easy check-in

Post ● No thanks

○ 1st Class - **£1.49** inc postal confirmation and 5 luggage labels

○ 2nd Class - **£0.99** postal confirmation

Lead passenger Privacy policy

Title: Select

First name

Last name

House name/number

Postcode

Contact number

Passengers

Adults: 2

Children (3 - 12): 0

Infants (0 - 2): 0

Car details [?]

Know your car details? ● Yes ○ No

Car registration

Find My Car

Car make

Car model

Car colour

Flight details [?]

Know your flight details? ● Yes ○ No

Destination airport

Outbound flight number (e.g. BA123)

Outbound terminal: Select

Inbound flight number (e.g. BA123)

Inbound terminal: Select

Cancellation protection

Include cancellation protection?

☑ Yes £0.99

No-one intends to cancel their holiday, but sometimes the unforeseen happens. Our cancellation waiver protects you if you do need to cancel your booking - you'll get a full refund, as long as you cancel with at least 24 hours' notice. It costs just 99p for a parking booking. Cancellation charges are £9.50 up to 24 hours before your stay date.

Donation to the Travel Foundation - Caring for the places you love to visit [?]

Support the Travel Foundation?

☐ Yes

£0.50

The Travel Foundation's work helps to ensure that the places we love to visit are protected for generations to come.

Payment Security policy

Payment Type: Select

Name on card

Card number

Expires on: -- --

Card security number [?]

Price £57.79

Make payment

Remember: If you find Park Wise for less than £56.80 within 24 hours of booking, you can park with us for FREE.

Park Wise T1 and T3

from 1pm, Thu, 22 Mar 2012
to 2pm, Thu, 29 Mar 2012

£56.80

Cancellation protection

Cancel up to 24 hours before stay and we'll refund you in FULL. (Normal cancellation charge £9.50 per booking)

[remove] £0.99

Total Price £57.79

Debit Card NO FEE
Credit Card + £0.00

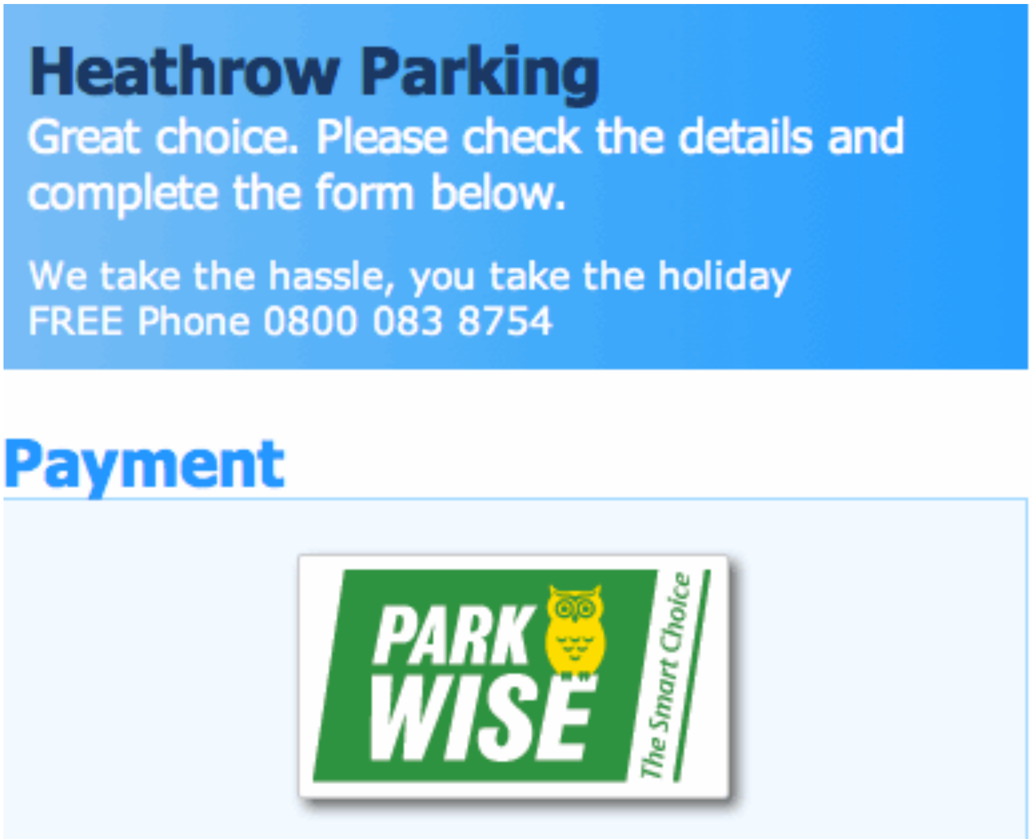
Terms and Conditions On making payment you agree to our booking terms and conditions and acknowledge this Park Wise information.

Need any last minute assistance?

We are open until 11pm tonight with an extra team to take your calls

0800 093 5603
Mobiles call [illegible]
FREE internet call Call me back

Airport hotels payment form, a large screen view. Large version.



Heathrow Parking

Great choice. Please check the details and complete the form below.

We take the hassle, you take the holiday
FREE Phone 0800 083 8754

Payment

Airport hotel's payment form, a small screen view. Large version.

The question became (as it always seems to be with mobile), what could we remove and what did we have to keep? Well, we tried to take the middle path, which is currently in development or might even be live by the time you read this.

We chose to split our payment process into two stages. Because our users can save more on their purchase by booking early, our first payment stage asks for the very minimum of information required in order to confirm a booking: name, car registration and credit-card details. This gives the user the best price available and chalks up another booking for us. Part two of the payment process is to gather the rest of the information required to “complete” the booking. This second stage can be filled out at the user’s convenience, either immediately or later on using our online booking management system. This eases any frustration caused by having to fill out a large form.

Good UI != Good UI

A good user interface means something completely different on mobile devices — and even tablets for that matter. Many of the user-friendly features we have implemented on our desktop website would just be bad ideas on these smaller mouse-less devices.

LIGHTBOXES

Lightboxes were all the rage a few years ago. They were a convenient and pretty way to display a small amount of content or something that wasn’t worth loading a new page for. In IE 7 and up, you can position lightboxes using **`position: fixed`**, which is great. On mobile devices, though, browsers do not implement **`position: fixed`**, or they implement it in an odd way to prevent non-mobile-ready websites from not working at all. This will ruin any lightboxes.

We recommend just loading a new page for lightbox content: less JavaScript, easier and fast. A new tab would also be fine, but due to the infancy of tabbed browsing on mobile devices, maintaining the flow is probably a better idea for now.

HOVERS

Content that is only visible via hovering obviously doesn't work on touchscreens. What used to be an easy way to hide content while keeping it accessible has become a bit of a nightmare to deal with. We tried just removing the hover and showing the content, to see what would happen. The iPhone actually handles hovers fairly well, translating them into tap events. On Android, you need to click and hold for a little while to prevent the default action of clicking the link (our links are anchor-tag-based).

In the end, modifying the code that handles the hovers (assuming it's JavaScript) and adding a tap event seemed to be the best solution. This allowed us to preserve the design's aesthetic, while keeping as much functionality as possible for mobile users.

```
if( document.createTouch ) {
    this.addEvent(c[j],'tap',this.tipOver,false);
} else {
    this.addEvent(c[j],'mouseover',this.tipOver,false);
    this.addEvent(c[j],'mouseout',this.tipOut,false);
}
```

DATE PICKER

Our date-picker calendar was one of the biggest hurdles to overcome in the UI. We have a text input that allows the user to enter a date. Prior to the date-picker, our solution was a dynamically generated select box, but that caused confusion among many users because they might have remembered the day of the week they were flying on but not the date. So, we added the jQuery UI Datepicker to make filling in the search form one step easier.

However, what was one step forward for convenience on the desktop was one step back on mobile. Focusing the text input would open both the date picker and the phone's keyboard, thus obscuring the date picker.

Our next option was to use the HTML5 date input. Because this element became available so recently, browsers are still playing catch up, and implementations vary wildly. It's just as rough on desktop, with Firefox still rendering it as a text input, Chrome adds an up/down selector and forces the date format, while Opera actually renders a calendar just like the jQuery UI Datepicker. This solution still requires the date-picker JavaScript, but it forces the format, which can actually make it less user-friendly. While the concept is great and the solution will be great once the bugs are ironed out, we found that the date type input is not yet ready for commercial use in this fashion.

Our eventual solution (not yet live) was to use a JavaScript "touch event" query to generate a more mobile-friendly date picker than the standard jQuery UI one. This creates an iOS-styled triple drop-down menu for day, month and year and is user-friendly on mobile devices. The no-JavaScript backup can be either a text input or a select drop-down menu. Have a look at the code for yourself.

Fix IE First

The final point, which reflects the complexity of mobile development, is how to fit old versions of IE into this new technology. IE 8 and below ignores media queries, which presents a rather sticky problem when your entire website is based on them. There are several solutions to this, which we'll explore below.

JAVASCRIPT POLYFILLS

I can think of two great JavaScript polyfill options for media queries. The first is Respond.js, which continually monitors the browser's width, parses the CSS and then serves the correct styles for that width. This is a great solution if you need the website to respond on IE 8 and below. The main issue is the time between the document loading and the JavaScript kicking in; the website is initially displayed using the base style sheet, usually the mobile view, before it "jumps" to the full desktop version. Obviously, this doesn't look great on a desktop monitor, and if the user is on an old browser, then their computer and Internet connection will probably be slow, too, which means that the jump time could be even longer.

The other JavaScript option here is the Chrome frame, which achieves the same end and has the same disadvantages. This solution isn't bad, but just not right for our implementation.

INCLUDE ALL MEDIA QUERIES

This is one of my favorite options for responsive websites and is also used in the latest version of the 320 and Up boilerplate. Create a separate CSS file for each device width; and for IE, serve them all to the user, with no media queries. With a mobile-first approach and a couple of fixed widths in your IE style sheet, this will serve the full-sized version of the website to users of outdated browsers. This solution is fast, simple and easy to maintain.

A SEPARATE IE STYLE SHEET ENTIRELY

Finally, given the right conditions, you could just write a completely separate IE style sheet, full of conditional comments to load the full desktop version of the website. Theoretically, this need only contain small amounts of layout information; but given that many of these styles will be reproduced in your media queries for wider widths, it can cause maintenance issues down the line. Duplicating code is never a good idea, which makes me wary of this solution.

Interestingly, we used this solution in the end, but with a twist. We used a PHP plugin in our template files to combine, compress and cache our CSS files. Due to some issues with the cache in IE, we were already generating a separate cached CSS file for IE users. We added a couple of lines to the PHP file to strip out media queries entirely as it combines and compresses the CSS. This method delivers the results of the “include all media queries” solution, while allowing the option for inline media queries in the style sheet. Because of the way we organized the CSS, this turned out to be the best solution for the project.

Conclusion

After all that, we finally have the first version of our responsive booking flow. I like to think that this epitomizes “mobile-ready.” We aren’t necessarily mobile-optimized, but our feet are on the bottom rung of a tall ladder that climbs to a great system that works perfectly on all devices. This is the starting point, if you will.

Was it worth it? It’s been a long journey, with a lot of head scratching and learning on our feet fast, but that’s what Web development is about, and I wouldn’t have it any other way. You can’t be perfect the first time round, and you don’t have to be. The point is that this technology is ready now, and the sooner you start using it, the better prepared you’ll be for the mobile market as it comes running at you. In the next few years, we’re hoping to see JavaScript network APIs that will allow Web users to add purchases directly to their monthly phone bill. I expect the mobile e-commerce market will explode at that point. Will you be ready?

From Monitor To Mobile: Optimizing Email Newsletters With CSS

Ros Hodgekiss

HTML email has a reputation for being a particularly tough design medium. So tough, in fact, that many designers regard coding and testing even the simplest email design to be almost as bad as fixing display quirks in Internet Explorer 6, and only slightly better than a tooth extraction. So, it's with much courage that I tell you today about using CSS in email newsletters: what works, where it's going and what you should do next.

After reading this article, you will hopefully come away with a few ideas on how to start coding email designs with improved readability and usability when viewed in Web, mobile and email desktop clients alike. Also included are a variety of resources to get you on the right path with using CSS in email.

Then again, the shaky state of email standards today may convince you that plain-text email is the way to go. The choice is yours.

CSS In HTML Email: The Good, The Bad And The Mobile-Ready

If you're about to embark on your first HTML email coding job, then you probably come from a Web background and are keen to flex a little CSS3 muscle, get a little JavaScript action happening, drop those shadows…

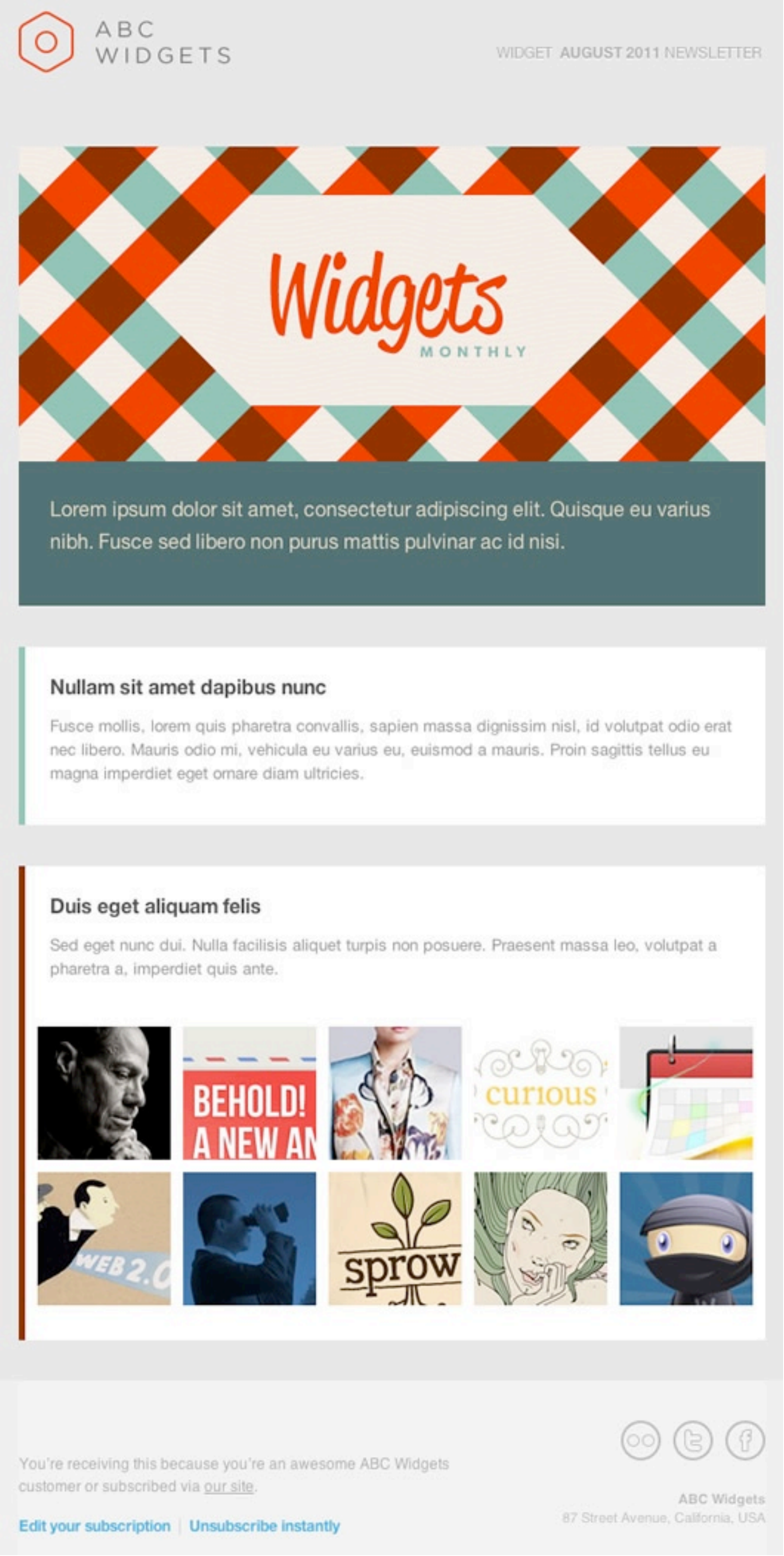
Not so fast. As any old hat to the email game can tell you, what works on the Web and what works in email are about as far apart as Sydney and Stockholm. For the most part, this is because pretty much every email client has its own way of doing things; while there are perhaps half a dozen browsers to test against when coding a Web page, there are literally dozens of email clients, many of which feature radically different CSS implementations.

But before you give up hope, here's some advice to get you through the night:

- A lot of CSS properties (such as **`font, color`** and **`border`**) work fine across many of the most popular email clients. Once you know which ones they are, you can tailor your designs accordingly.
- When a CSS property doesn't work so well, there are often workarounds (such as using **`cellpadding`** in tables instead of **`padding`**).
- When workarounds don't exist, focus on graceful degradation.
- Your design will **never look exactly the same** across all email clients, no matter how you use CSS. Once you (and your clients) make peace with this, I guarantee you will sleep better at night.

- Keep it simple. The less complicated your design and layout, the less likely something will go wrong. Favor table layouts over `div`s, and make sure your message is readable (which means text, not images).

At this point, you may be saying to yourself, “Well, this all just sounds too hard.” So, perhaps I should remind you how beautiful an HTML email can look, with just a sprinkling of CSS:



ABC
WIDGETS

WIDGET **AUGUST 2011** NEWSLETTER

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque eu varius nibh. Fusce sed libero non purus mattis pulvinar ac id nisi.

Nullam sit amet dapibus nunc

Fusce mollis, lorem quis pharetra convallis, sapien massa dignissim nisl, id volutpat odio erat nec libero. Mauris odio mi, vehicula eu varius eu, euismod a mauris. Proin sagittis tellus eu magna imperdiet eget ornare diam ultricies.

Duis eget aliquam felis

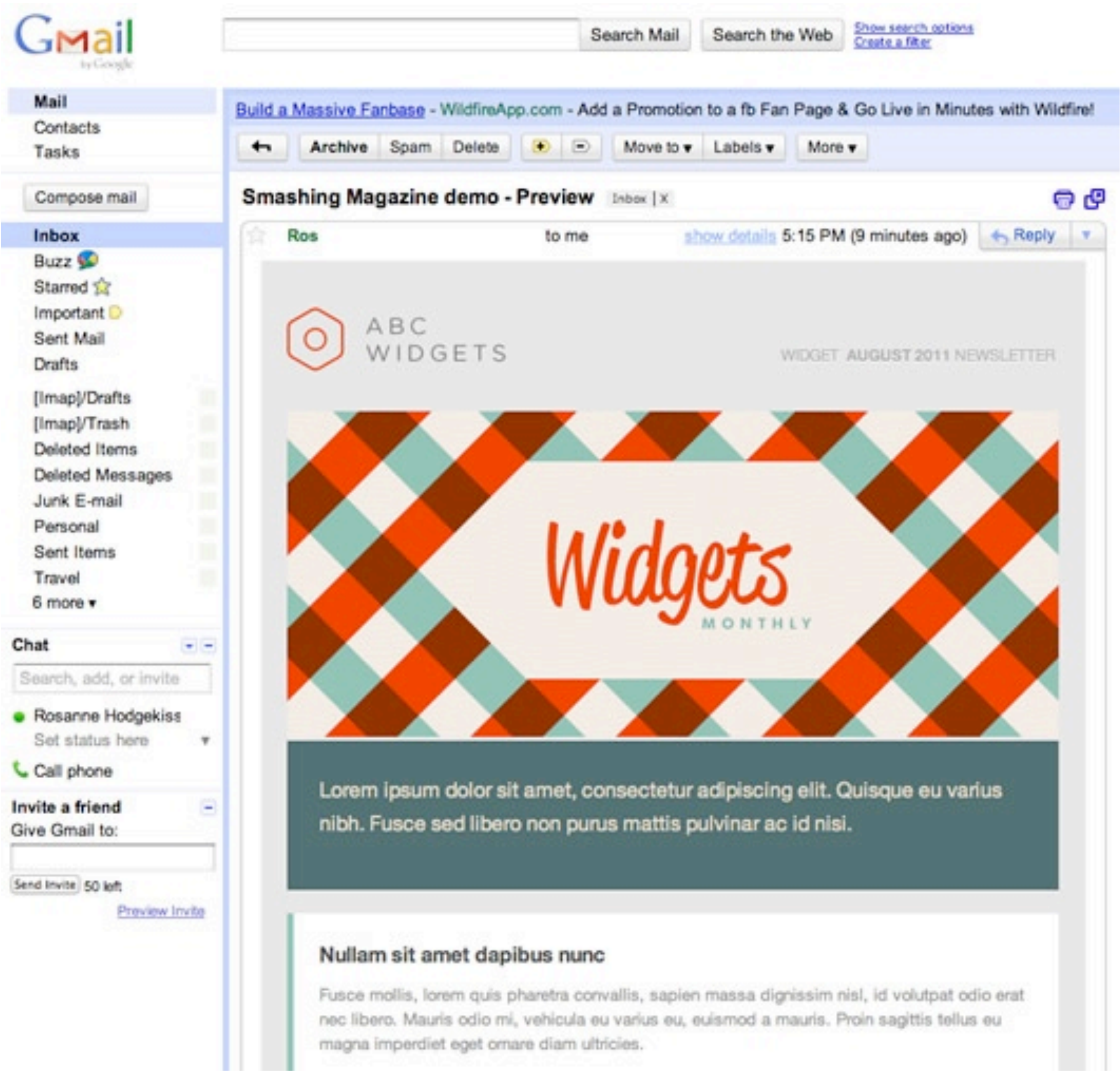
Sed eget nunc dui. Nulla facilisis aliquet turpis non posuere. Praesent massa leo, volutpat a pharetra a, imperdiet quis ante.

You're receiving this because you're an awesome ABC Widgets customer or subscribed via our site.

Edit your subscription | Unsubscribe instantly

ABC Widgets
87 Street Avenue, California, USA

For a more realistic view of what this design will look like in the inbox, here's the same email in Gmail and Outlook 2007. Both are notoriously tricky email clients to work with:



Build a Massive Fanbase - WildfireApp.com - Add a Promotion to a fb Fan Page & Go Live in Minutes with Wildfire!

Smashing Magazine demo - Preview

Ros to me show details 5:15 PM (9 minutes ago)

ABC WIDGETS

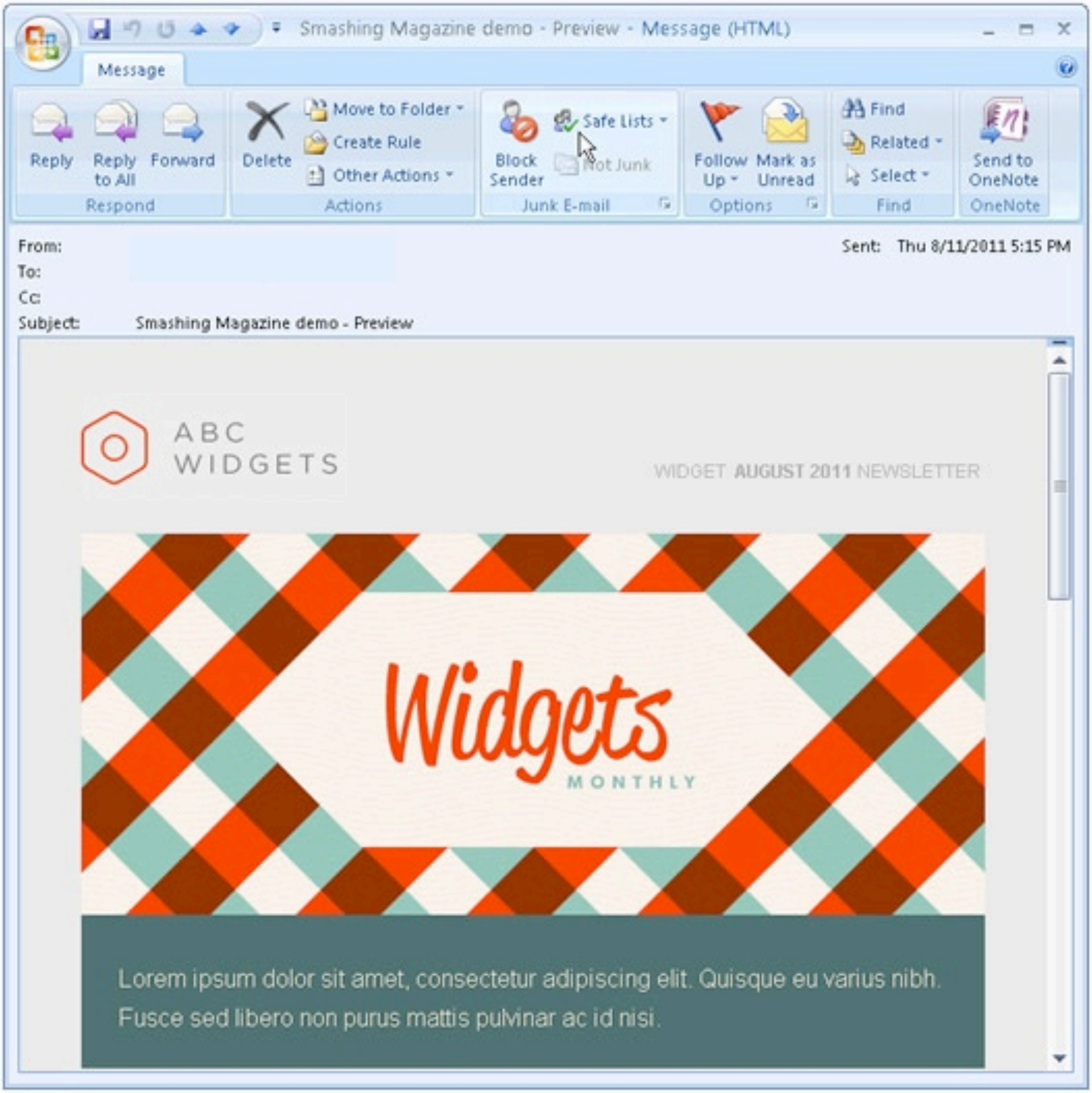
WIDGET AUGUST 2011 NEWSLETTER

Widgets MONTHLY

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque eu varius nibh. Fusce sed libero non purus mattis pulvinar ac id nisi.

Nullam sit amet dapibus nunc

Fusce mollis, lorem quis pharetra convallis, sapien massa dignissim nisl, id volutpat odio erat nec libero. Mauris odio mi, vehicula eu varius eu, euismod a mauris. Proin sagittis tellus eu magna imperdiet eget ornare diam ultricies.



From:
To:
Cc:
Subject: Smashing Magazine demo - Preview

Sent: Thu 8/11/2011 5:15 PM

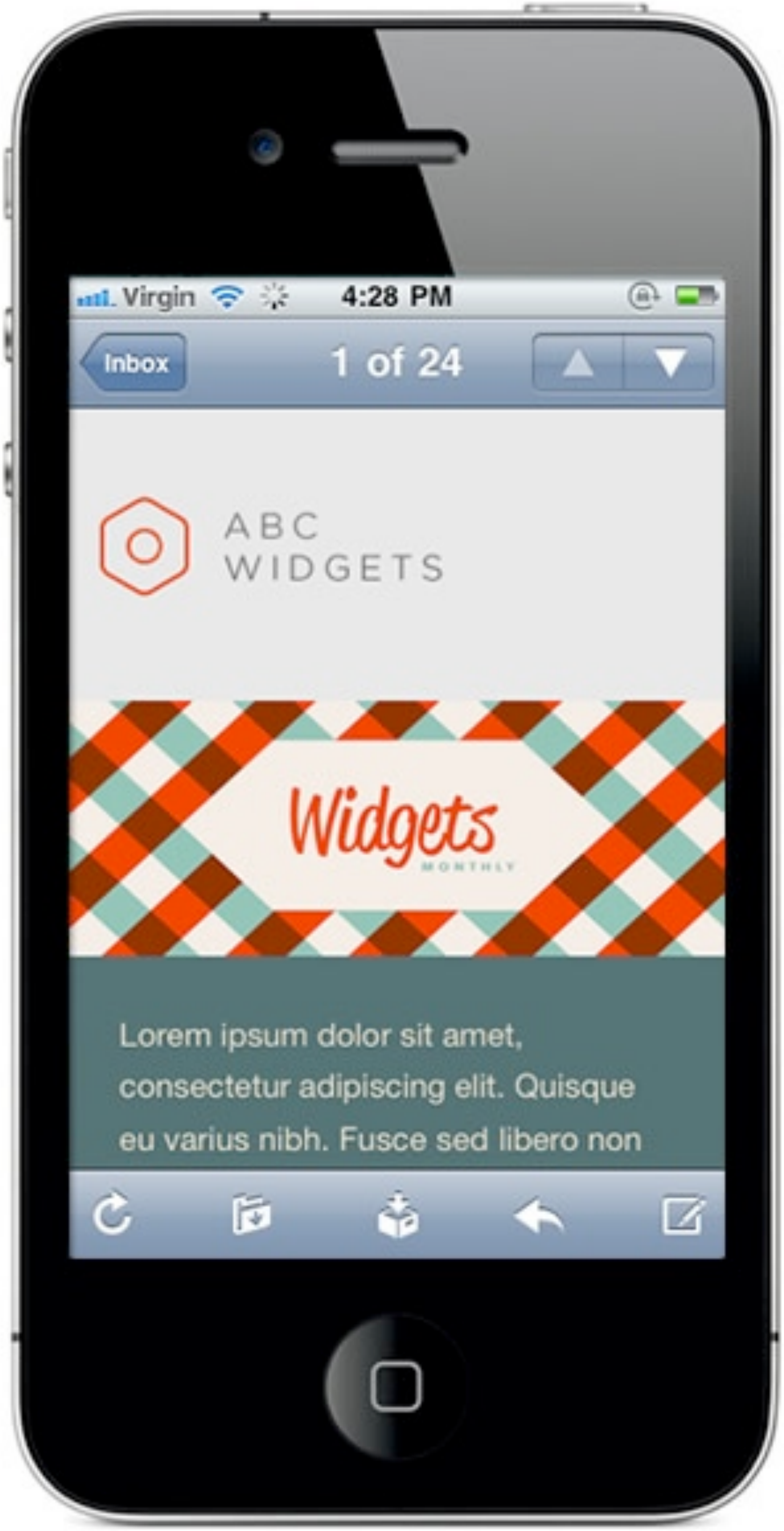
ABC WIDGETS

WIDGET **AUGUST 2011** NEWSLETTER

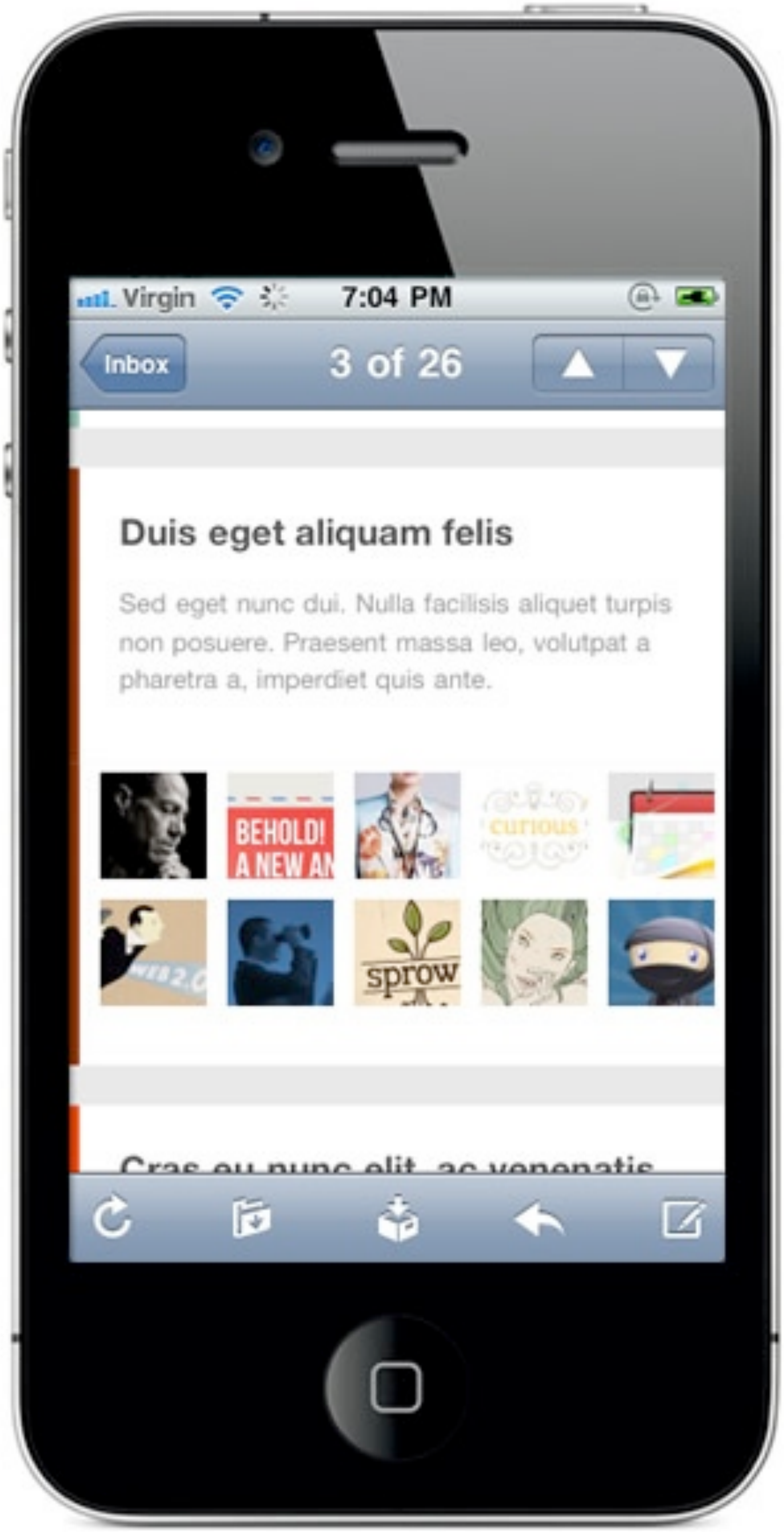
Widgets MONTHLY

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque eu varius nibh. Fusce sed libero non purus mattis pulvinar ac id nisi.

See? Ain’t so bad after all. But what’s more exciting is how you can use CSS to adapt an HTML email for optimal display on mobile devices. Here’s the same newsletter as viewed on the iPhone:



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque eu varius nibh. Fusce sed libero non

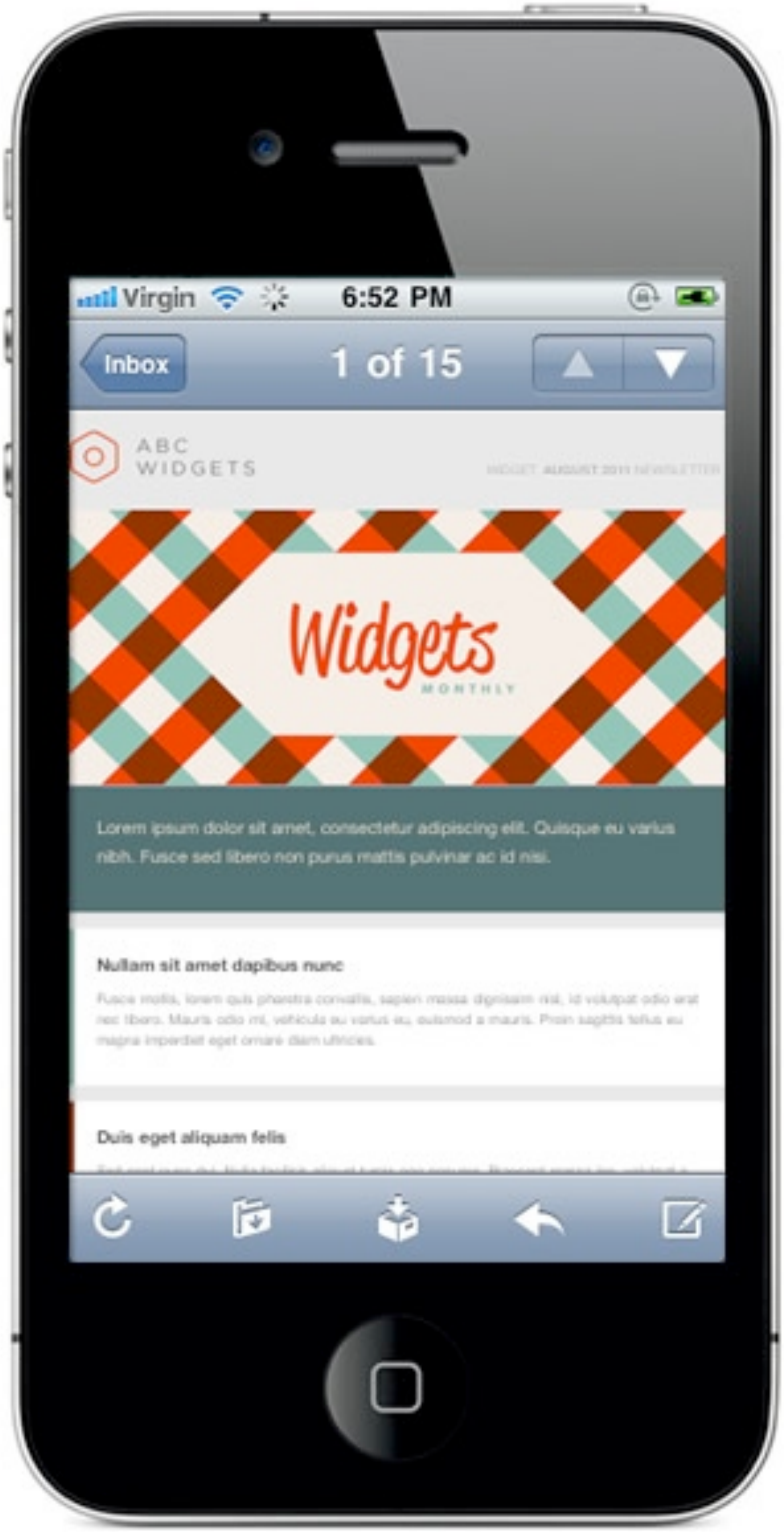


Duis eget aliquam felis

Sed eget nunc dui. Nulla facilisis aliquet turpis non posuere. Praesent massa leo, volutpat a pharetra a, imperdiet quis ante.

Cras eu nunc elit, ac venenatis

For comparison, let’s look at the same email newsletter without mobile-specific CSS:

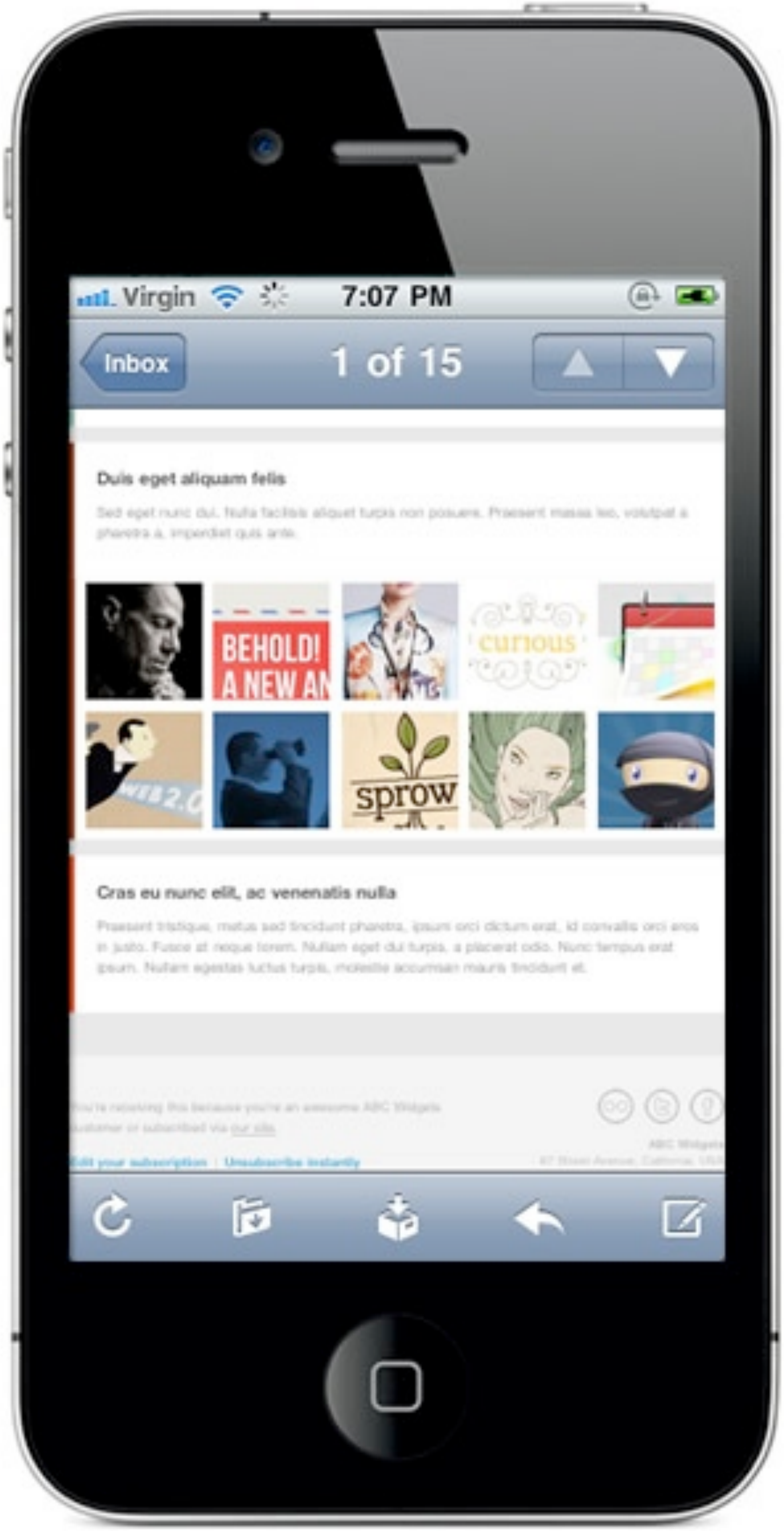


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque eu varius nibh. Fusce sed libero non purus mattis pulvinar ac id nisi.

Nullam sit amet dapibus nunc

Fusce mollis, lorem quis pharetra convallis, sapien massa dignissim nisl, id volutpat odio erat nec libero. Mauris odio mi, vehicula eu varius eu, euismod a mauris. Proin sagittis tellus eu magna imperdiet eget ornare diam ultricies.

Duis eget aliquam felis



Duis eget aliquam felis

Sed eget nunc dui. Nulla facilisis aliquet turpis non posuere. Praesent massa leo, volutpat a pharetra a, imperdiet quis ante.

Cras eu nunc elit, ac venenatis nulla

Praesent tristique, metus sed tincidunt pharetra, ipsum orci dictum erat, id convallis orci eros in justo. Fusce at neque lorem. Nullam eget dui turpis, a placerat odio. Nunc tempus erat ipsum. Nullam egestas luctus turpis, molestie accumsan mauris tincidunt et.

ou're receiving this because you're an awesome ABC Widgets ustomer or subscribed via our site.

dit your subscription | Unsubscribe instantly

ABC Widgets
87 Street Avenue, California, USA

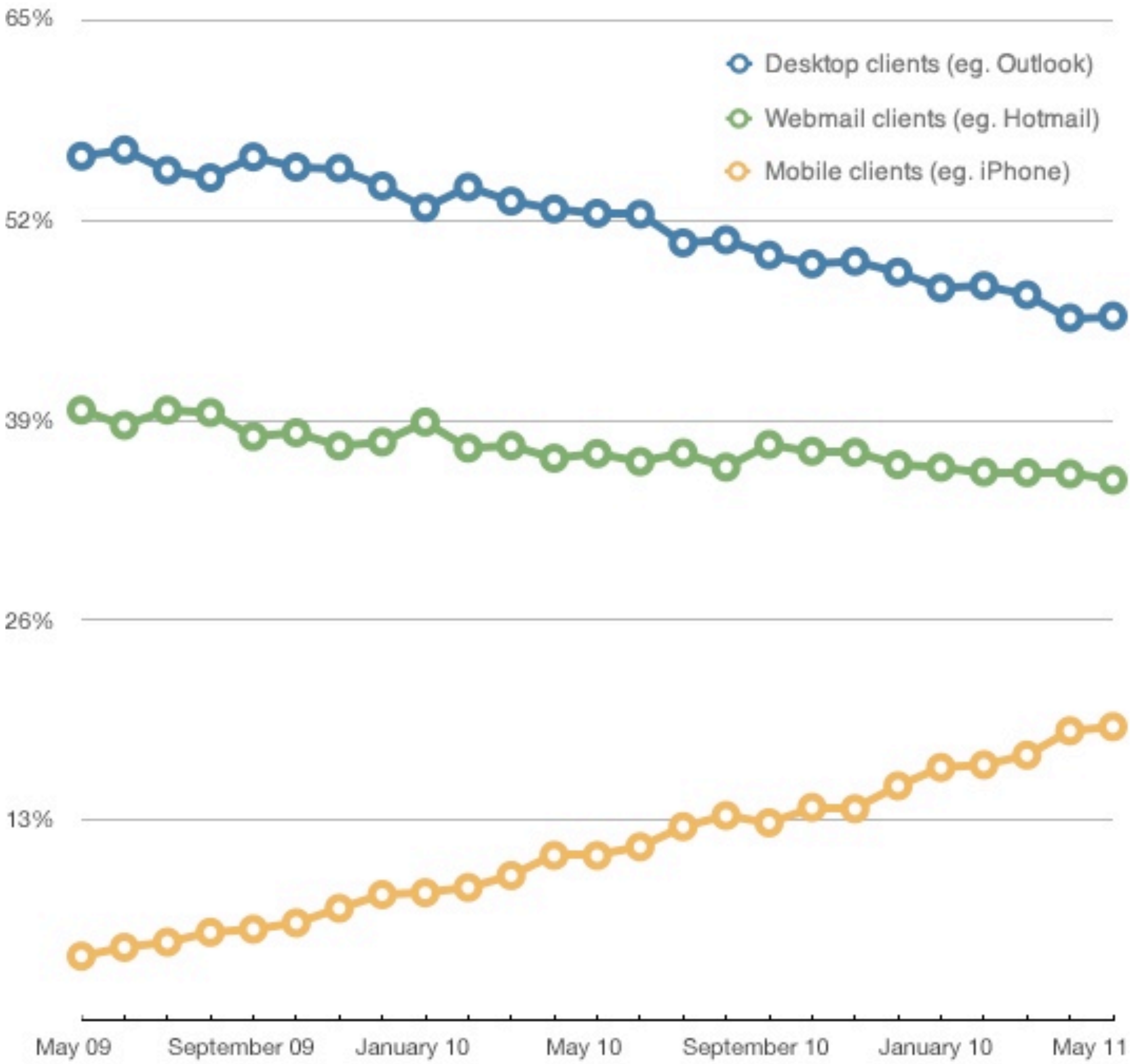
The change might seem subtle, but applying a mobile-specific style sheet has improved the readability of the email considerably. It has also allowed us to remove a lot of visual clutter (like the "Widget August 2011 Newsletter" text) that would have taken up valuable real estate on a small screen.

So, we've gone from an email layout that requires a lot of pinching and zooming, to one that can be easily read with a linear scrolling motion, using CSS alone. We'll look at how you can apply similar improvements to your email campaigns in a moment. But first, let's start with some of the fundamentals of using CSS in your HTML email designs.

The State Of CSS Support In Email Today

When I mentioned that a lot of CSS properties out there work fine in many email clients, I wasn't trying to be vague. Thankfully, the research into what works and what doesn't has already been done. You need only skim this guide to CSS support in email clients to see what properties are within and off limits. Or just know that most of your CSS rendering troubles will come from Outlook 2010, Lotus Notes and Gmail, due to their refusal to do anything useful with CSS style sheets.

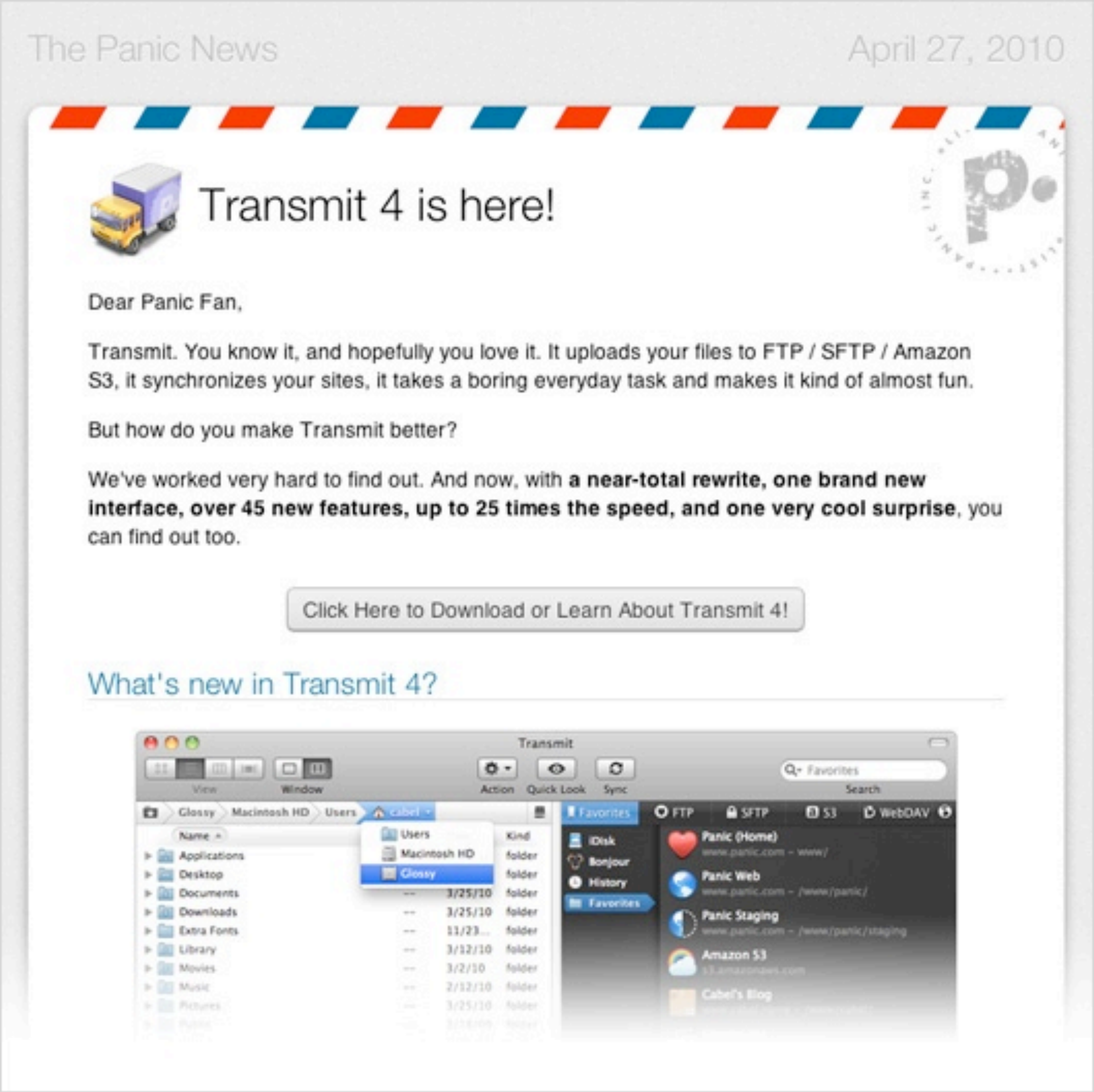
These issues are nothing new. Indeed, the battle for market share between email clients that play nice with CSS versus those that don't has been pitched for years now. But progress is being made. Looking at the data from over 3 billion emails sent, we found that mobile email clients have gained ground dramatically with the growth of mobile professionals. In fact, one in five emails are now opened on a mobile device. Here is how desktop, Web and mobile email clients have fared comparatively over the last two years:



Desktop, Web and mobile email client market share, 2009 to 2011.

Mobile’s ascent is great news for email designers everywhere for one reason: roughly 75% of mobile email is read on an iOS device. iOS devices use the Webkit rendering engine, which means they can display really nice-looking HTML emails.

Our friends at Panic (the creators of such popular Mac titles as Coda and Transmit) were well aware of this when they got started on their email announcements. As purveyors of Mac software, they can pretty much always count on their emails being read in Webkit-powered email clients like Apple Mail and the iPhone. As a result, they’ve been able to pull a lot of CSS3 trickery out of the toolbox, including **`border-radius`** and **`text-shadow`**:



The Panic News April 27, 2010

Transmit 4 is here!

Dear Panic Fan,

Transmit. You know it, and hopefully you love it. It uploads your files to FTP / SFTP / Amazon S3, it synchronizes your sites, it takes a boring everyday task and makes it kind of almost fun.

But how do you make Transmit better?

We've worked very hard to find out. And now, with **a near-total rewrite, one brand new interface, over 45 new features, up to 25 times the speed, and one very cool surprise**, you can find out too.

Click Here to Download or Learn About Transmit 4!

What's new in Transmit 4?

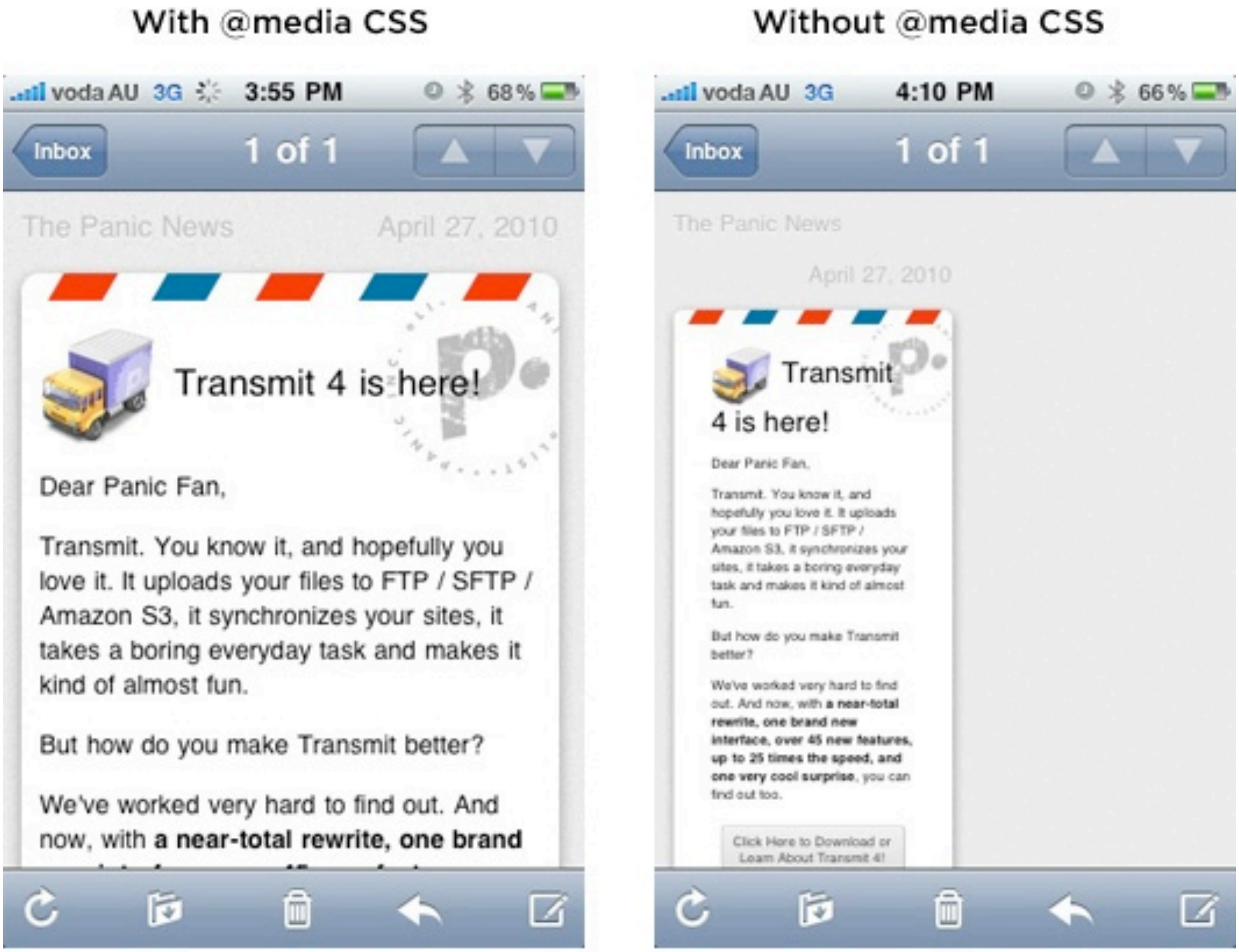
But what really impressed me was their use of CSS3 animation. Check out the mesmerizing glowing button effect in this video:

To sate your curiosity, here’s the code they used to achieve this effect:

```
-webkit-animation-name: 'glow';
-webkit-animation-duration: .7s;
```

```
  -webkit-animation-iteration-count: infinite;
  -webkit-animation-direction: alternate;
  -webkit-animation-timing-function: ease-in-out;
```

And you thought HTML email was a boring medium? Well, to temper things a bit, CSS3 still has very limited support beyond a handful of Webkit-powered email clients, so use it with discretion. But with that in mind, let's look at Panic's newsletter again, this time on the iPhone. For comparison, here it is both with and without a **`@media`** query (which calls the mobile style sheet):



The Panic News April 27, 2010

Transmit 4 is here!

Dear Panic Fan,

Transmit. You know it, and hopefully you love it. It uploads your files to FTP / SFTP / Amazon S3, it synchronizes your sites, it takes a boring everyday task and makes it kind of almost fun.

But how do you make Transmit better?

We've worked very hard to find out. And now, with **a near-total rewrite, one brand**

The Panic News

April 27, 2010

Transmit 4 is here!

Dear Panic Fan,

Transmit. You know it, and hopefully you love it. It uploads your files to FTP / SFTP / Amazon S3, it synchronizes your sites, it takes a boring everyday task and makes it kind of almost fun.

But how do you make Transmit better?

We've worked very hard to find out. And now, with **a near-total rewrite, one brand new interface, over 45 new features, up to 25 times the speed, and one very cool surprise**, you can find out too.

Click Here to Download or Learn About Transmit 4!

Let’s now look at how you can optimize your email newsletters for small displays as well.

Inside An HTML Email Builder’s Toolkit

Before we go charging in and dropping **`@media`** queries by the dozen, let’s make sure we have the tools for the job. We’ll need the following:

- An email marketing service like MailChimp or CampaignMonitor to send HTML email campaigns (you can’t do this from desktop or Web email clients like Gmail — sorry);
- Code editing software, such as Coda or Dreamweaver;
- A mobile device to test on (like an iPhone or Google Nexus);
- A variety of Web and desktop email accounts to test, either set up individually or automated via a service such as Litmus;
- An intermediate-level understanding of HTML and CSS;
- A lot of patience.

In addition, you’ll need a way to move your CSS inline, for the benefit of clients like Gmail, which strip out the **`head`** section of HTML email code. Many email marketing apps can do this automatically when you import your campaign, but if yours doesn’t, then you can use a service like Premailer or MailChimp’s Automatic CSS Inliner Tool for the job. Note that Premailer currently strips out **`@media`** queries, so you’ll have to paste them back in prior to testing your design.

Now that you’re armed and dangerous, let’s launch into the theory and code behind mobile email design.

Using CSS To Optimize Your Email For Mobile Devices

If you've created mobile style sheets for the Web or have read into responsive design, then you probably know a bit about **`@media`** queries. If not, then here's the skinny: they allow you to provide targeted CSS style sheets that are triggered when a device's capabilities match specific criteria. For example, you could use a media query to specify that a couple of lines of CSS be applied exclusively to devices with a display width of up to 480 pixels, in order to make your website or email easier to read on these screens.

The following is a snippet of code from the earlier newsletter, telling mobile email clients and browsers to scale down the width of the email layout to 300/325 pixels wide, so that it fits comfortably on displays that are no wider than 480 pixels (i.e. the width of an iPhone in landscape mode):

```
@media only screen and (max-device-width: 480px) {
   …
   table[class=table], td[class=cell] { width: 300px !
important; }
   table[class=promotable], td[class=promocell] { width:
325px !important; }
   …
}
```

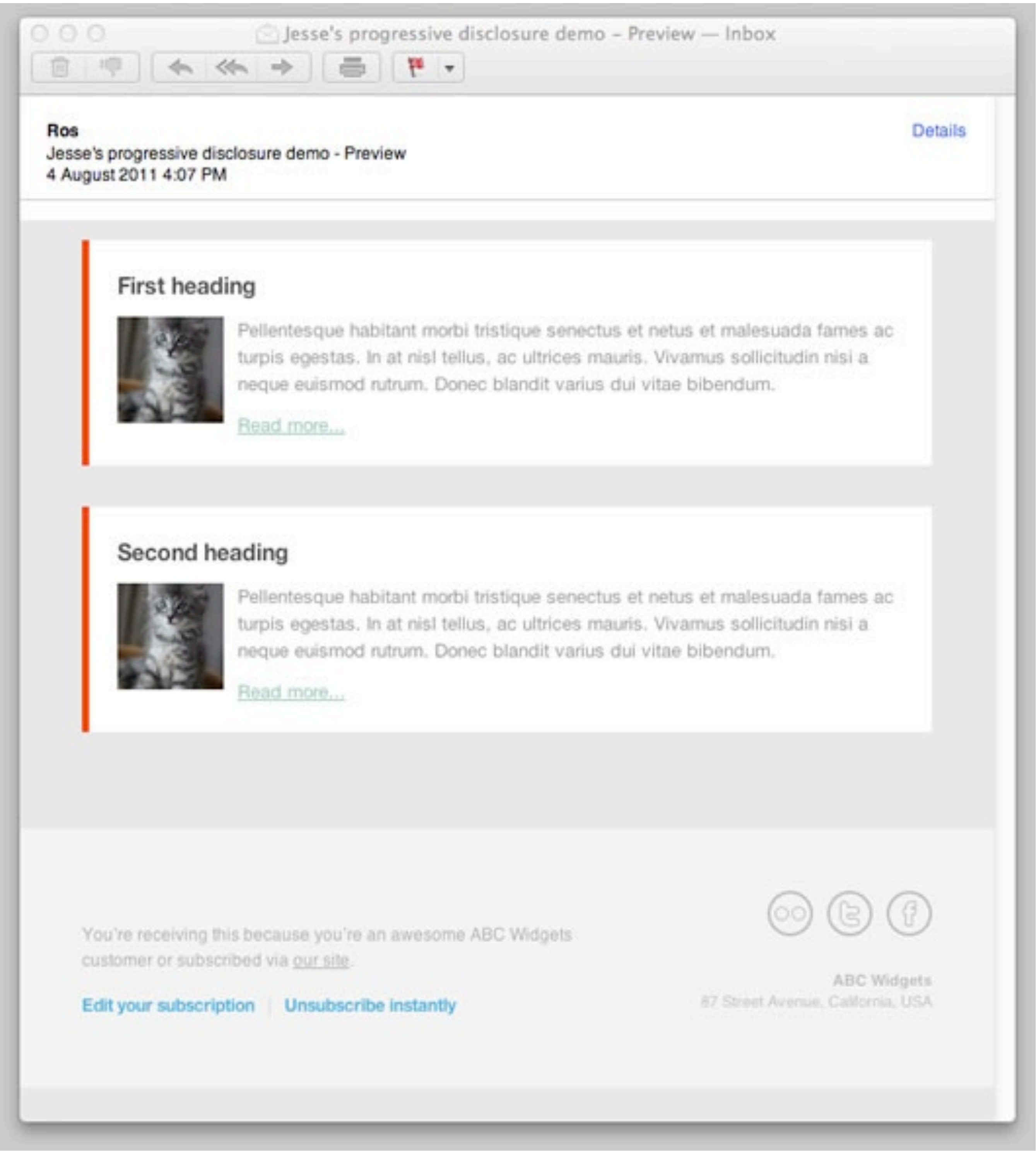
Note how we're using attribute selectors here. This is to prevent Yahoo! Mail Beta from accidentally calling this style sheet.

One thing to note at this point is that, while you can shrink the dimensions of a design (or hide bits, as we'll do later) using an **`@media`** query, the user will still be downloading all of the content. Adding mobile-specific styles shouldn't be confused with providing a slimmer or faster-loading version of your content. Mobile-specific styles just make the content easier to read.

The great news is that mobile email clients such as iOS Mail and Android's default mail client follow **`@media`** queries to the letter. So, for example, you could pop one into a style sheet in the head of your HTML email code to transform the design into an easily readable, mobile-friendly layout like the one pictured above. You can also do things like shrink the dimensions of images, and use **`display: none`** to hide visual elements that don't work on small screens. We'll be using the latter CSS property in the tutorial below to make a lot of email content fit into a very small space.

Using Progressive Disclosure To Shrink Content On Mobile Devices

Now that you have the fundamentals of developing mobile-friendly email designs down pat, let's learn about an advanced mobile email design technique called progressive disclosure. This involves hiding content behind an interactive element, such as a button or link, and then displaying it on click (or tap). In the context of mobile email, the content might be paragraphs of text that require a lot of scrolling — which, you may know from personal experience, is hard to do. Here is an email design with multiple articles:



Ros
Jesse's progressive disclosure demo - Preview
4 August 2011 4:07 PM

First heading

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. In at nisl tellus, ac ultrices mauris. Vivamus sollicitudin nisi a neque euismod rutrum. Donec blandit varius dui vitae bibendum.

Read more...

Second heading

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. In at nisl tellus, ac ultrices mauris. Vivamus sollicitudin nisi a neque euismod rutrum. Donec blandit varius dui vitae bibendum.

Read more...

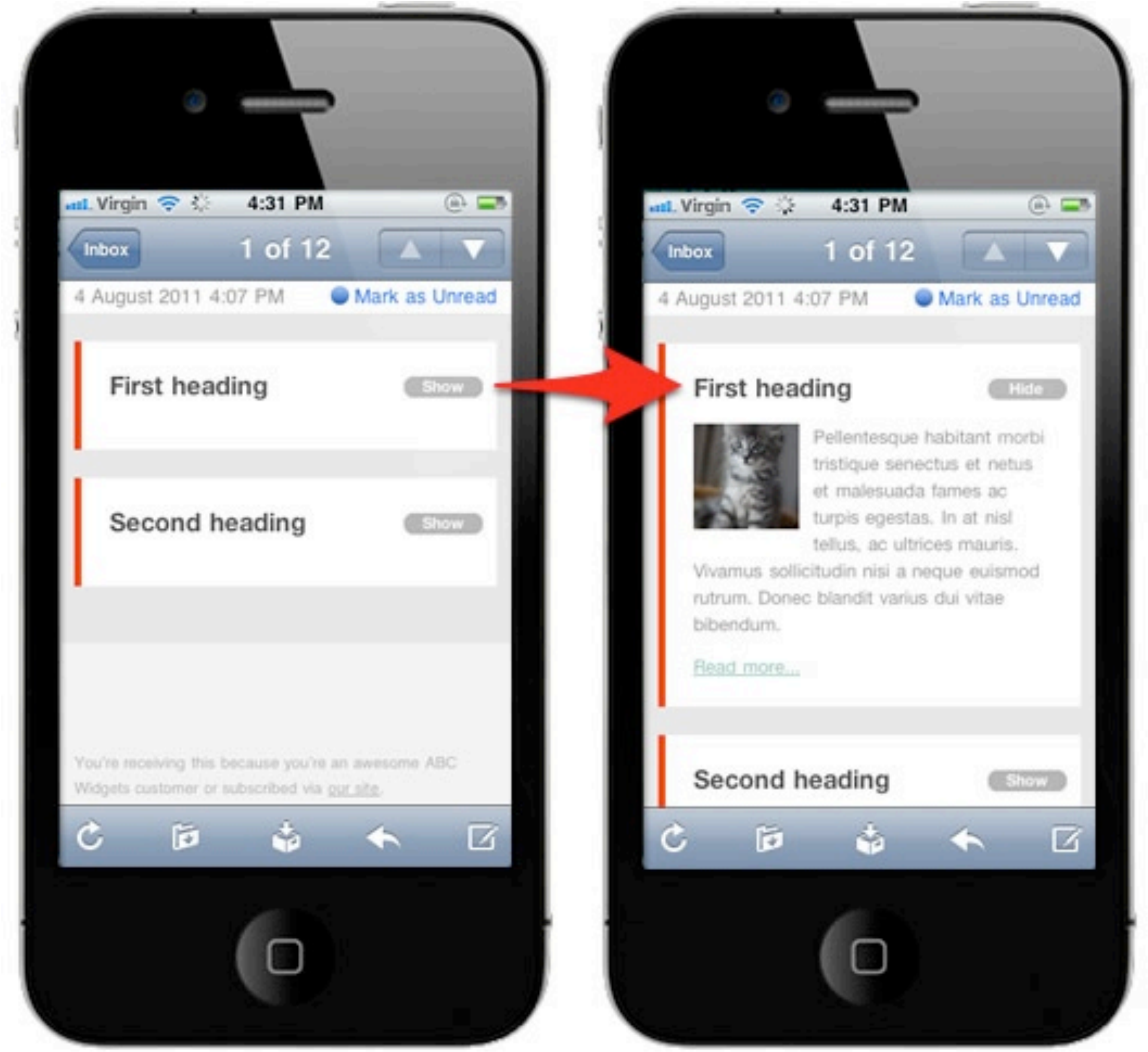
You're receiving this because you're an awesome ABC Widgets customer or subscribed via our site.

Edit your subscription | Unsubscribe instantly

ABC Widgets
87 Street Avenue, California, USA

The problem here is that even if the email design is optimized for mobile, only the first article will be visible without scrolling. So, if you want your readers to know that there are two or more articles, you could apply progressive disclosure. After all, the mobile version of Wikipedia uses it, and a lot of mobile apps use it, so why not apply it to email?

First, here's a shot of what we're going to achieve on mobile devices:



You're receiving this because you're an awesome ABC Widgets customer or subscribed via our site.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. In at nisl tellus, ac ultrices mauris. Vivamus sollicitudin nisi a neque euismod rutrum. Donec blandit varius dui vitae bibendum.

Notice how the articles are shown and hidden using a toggle button? To achieve this, let's walk through some fairly straightforward code. First, there's the code for the individual articles:

```
<td>
   <h4><a href="http://yourdomain.com" class="link">First
heading</a></strong></h4>
   <a href="#" class="mobilehide">Hide</a> <a href="#"
class="mobileshow">Show</a>
   <div class="article">
      <p class="bodytext">
         <img src="kitten.jpg" style="float: left;"
>Pellentesque habitant morbi…
      </p>
      <a href="http://yourdomain.com">Read more…</a>
   </div>
</td>
```

Take note of the classes **`mobilehide, mobileshow`** and **`article`**. These will be handling most of the action.

Then, in a regular CSS style sheet (i.e. one not enclosed by a **`@media`** query), we'll use **`display: none`** to hide our show/hide button in desktop and Web email clients. Here's the code snippet for that:

```
a.mobileshow, .mobilehide {
display: none;
color: #fff;
}
```

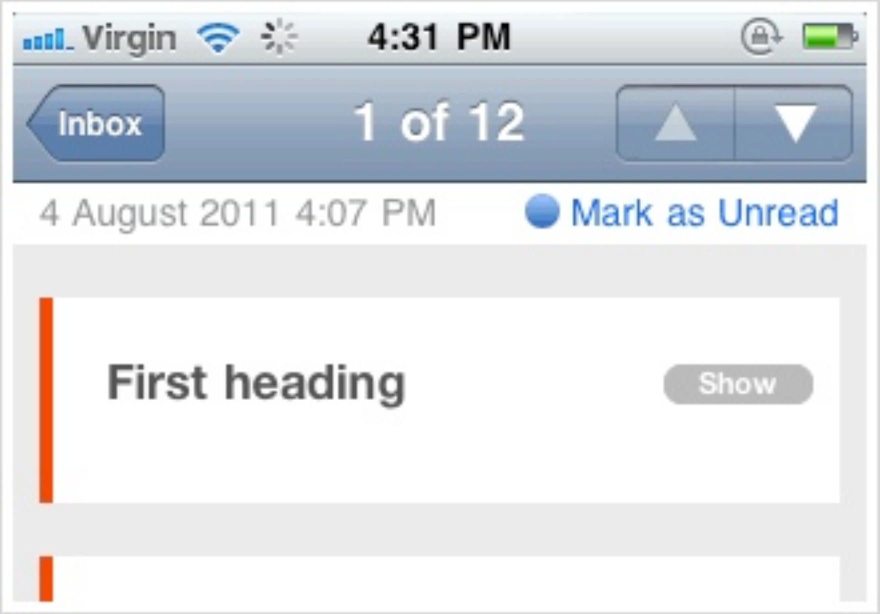
Yes, so we’ve had to “white them out” for the benefit of Gmail and Android Gmail, which don’t support **`display: none`**. Thankfully, the buttons are unresponsive in those clients.

Finally, onto the mobile-specific CSS in the **`@media`** query, which I’m sure you’ve been waiting for. Here, we’ll style the show/hide buttons and handle the swapping of states when the show/hide button is tapped:

```
@media only screen and (max-device-width: 480px) {
   …
    a[class="mobileshow"], a[class="mobilehide"] {
      display: block !important;
      color: #fff !important;
      background-color: #aaa;
      border-radius: 20px;
      -webkit-border-radius: 20px;
      -moz-border-radius: 20px;
      padding: 0 8px;
      text-decoration: none;
      font-weight: bold;
      font-family: "Helvetica Neue", Helvetica, sans-serif;
      font-size: 11px;
      position: absolute;
      top: 25px;
      right: 10px;
      text-align: center;
      width: 40px;
   }
   div[class="article"] {
      display: none;
   }
   a.mobileshow:hover {
      visibility: hidden;
   }
   .mobileshow:hover + .article {
      display: inline !important;
```

```
        }
        ...
    }
```

With this added to your HTML email code, you should be able to toggle the display of articles in your email design by tapping the show/hide button, like so:



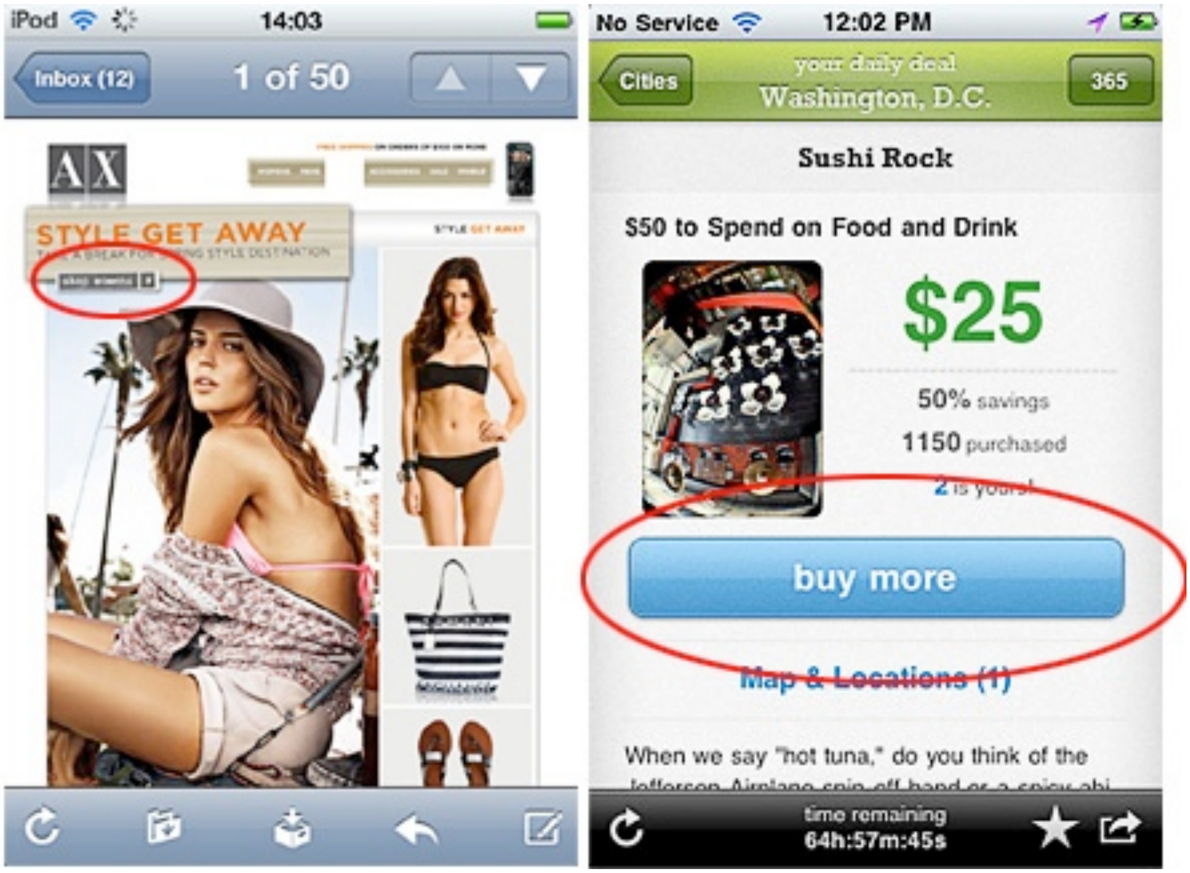
Despite the imperfections in this implementation of progressive disclosure, we've found that this technique works like a charm in iPhone Mail and Android Mail. In mobile clients that don't play so nice with CSS and `@media` queries (i.e. Android Gmail, BlackBerry and Windows Mobile 7), the text will display, but the show/hide button conveniently does not. For more detailed information on using progressive disclosure, CampaignMonitor's article "Optimizing HTML Email for Mobile Using Progressive Disclosure" is well worth a read.

Well, that's enough information on mobile email design to elevate you to black-belt status. For a more hands-on look at mobile email design, I highly recommend the tutorial, "Make Your HTML Email 5× More Mobile-Friendly."

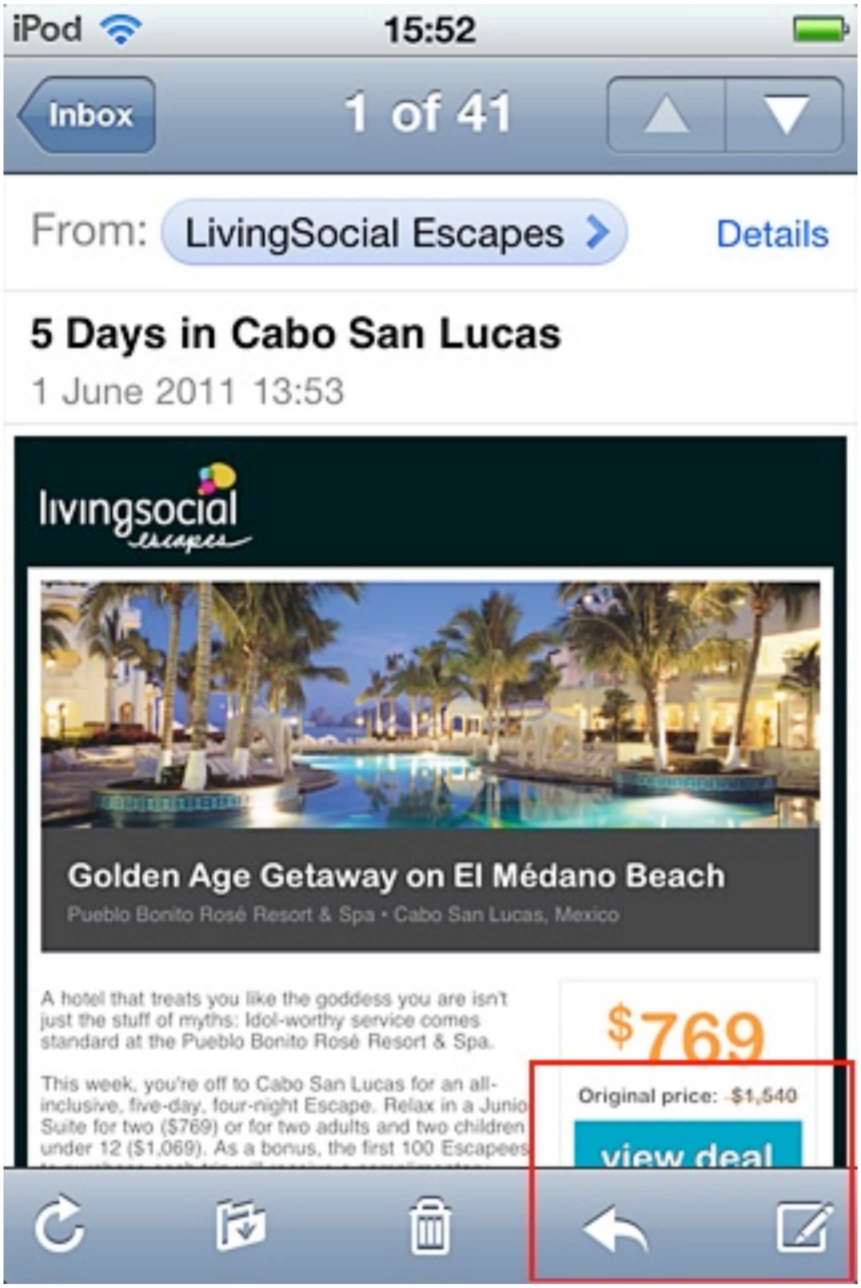
And read “Mobile Email Design in Practice” to get started downloading a mobile-optimized template.

What’s Next For HTML Email Designers?

So, while CSS-unfriendly desktop clients and Web email clients like Gmail will always be here to rain on our parade, the good news is that the rise of mobile email has meant that we may soon be at liberty to create more Web-like email experiences. It has also meant that optimizing your newsletters for handheld and touch displays has gone from being a “nice thing to have” to a given. This doesn’t just affect email newsletters at the code level, but it also changes the way we display design elements. For example, in the following two mobile designs, which do you think is the more effective call-to-action (CTA) button?



Or consider the CTA in the following mobile-friendly email. If it’s not visible “above the fold,” as they say, then will it be seen at all? Or worse, will recipients end up accidentally tapping the toolbar instead?



5 Days in Cabo San Lucas

1 June 2011 13:53

Golden Age Getaway on El Médano Beach

Pueblo Bonito Rosé Resort & Spa • Cabo San Lucas, Mexico

A hotel that treats you like the goddess you are isn't just the stuff of myths: Idol-worthy service comes standard at the Pueblo Bonito Rosé Resort & Spa.

This week, you're off to Cabo San Lucas for an all-inclusive, five-day, four-night Escape. Relax in a Junio Suite for two ($769) or for two adults and two children under 12 ($1,069). As a bonus, the first 100 Escapees

Original price: ~~$1,540~~

If designers aren't asking these questions, they sure will be soon. You need only visit Style Campaign's blog (which provided the examples above) to grasp the importance of solid mobile design.

Here are a few other important things to consider when designing adaptive layouts:

- Single-column layouts that are no wider than 500 to 600 pixels work best on mobile devices. They're easier to read, and if they fall apart, they'll do so more gracefully.
- Links and buttons should have a minimum target area of 44 × 44 pixels, as per Apple guidelines. Nothing sucks more than clouds of tiny links on touchscreen devices.
- The minimum font size displayed on iPhones is 13 pixels. Keep this in mind when styling text, because anything smaller will be upscaled and could break your layout. Alternatively, you could override this behavior in your style sheet.
- More than ever, keep your message concise, and place all important design elements in the upper portion of the email, if possible. Scrolling for miles is much harder on a touchscreen than with a mouse.

Now it’s your turn to design wicked HTML email newsletters that, with a dash of CSS, look just as effective on the small screen as they do in your Web browser or desktop inbox. I have no doubt that your readers will appreciate the effort.

Resources To Help You Make The Most Of CSS In Email

With funky CSS support and coding practices from circa 1994, designing HTML emails might seem like rocket science. Thankfully, quite a bit of solid documentation exists on effective HTML email design, so below is my recommended reading list if you want to take your newsletters to the next level.

How To Use CSS3 Media Queries To Create a Mobile Version of Your Website

Smashing Editorial

CSS3 continues to both excite and frustrate web designers and developers. We are excited about the possibilities that CSS3 brings, and the problems it will solve, but also frustrated by the lack of support in Internet Explorer 8. This article will demonstrate a technique that uses part of CSS3 that is also unsupported by Internet Explorer 8. However, it doesn't matter as one of the most useful places for this module is somewhere that does have a lot of support — small devices such as the iPhone, and Android devices.

In this article I'll explain how, with a few CSS rules, you can create an iPhone version of your site using CSS3, that will work now. We'll have a look at a very simple example and I'll also discuss the process of adding a small screen device stylesheet to my own site to show how easily we can add stylesheets for mobile devices to existing websites.

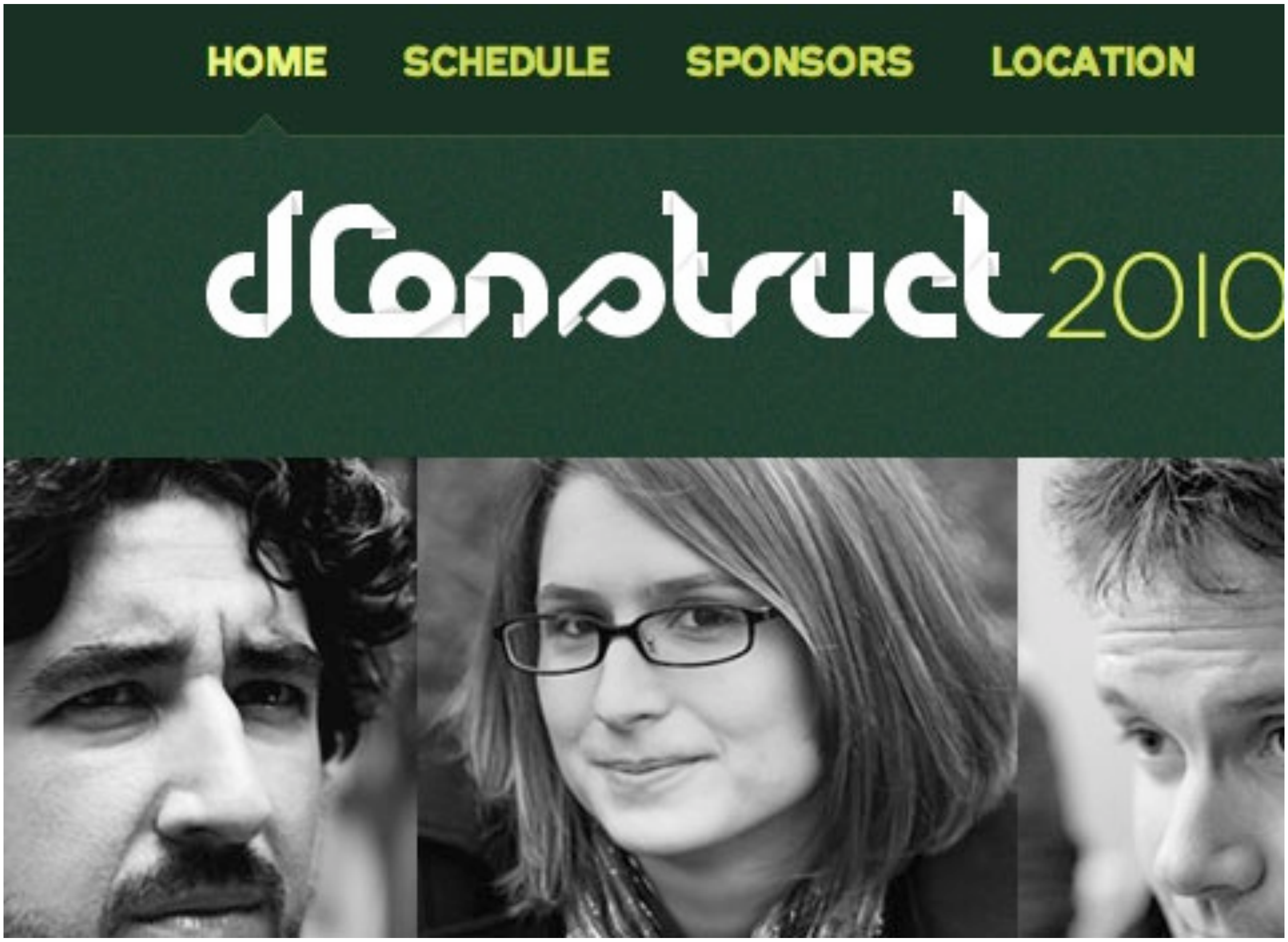
Media Queries

If you have ever created a print stylesheet for a website then you will be familiar with the idea of creating a specific stylesheet to come into play under certain conditions – in the case of a print stylesheet when the page is printed. This functionality was enabled in CSS2 by *media types*. Media Types let you specify a type of media to target, so you could target print, handheld and so on. Unfortunately these media types never gained a lot of support by devices and, other than the print media type, you will rarely see them in use.

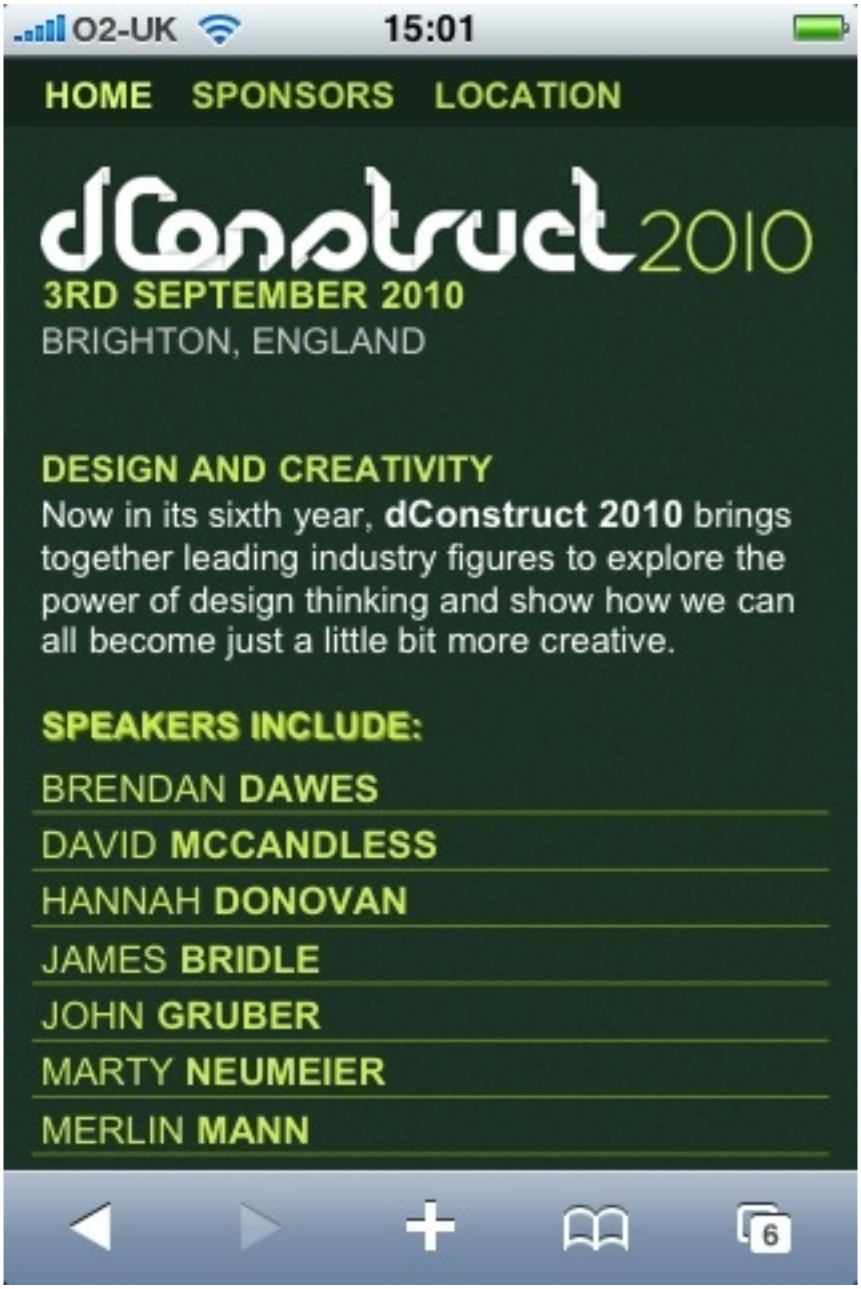
The Media Queries in CSS3 take this idea and extend it. Rather than looking for a *type* of device they look at the *capability* of the device, and you can use them to check for all kinds of things. For example:

- width and height (of the browser window)
- device width and height
- orientation – for example is a phone in landscape or portrait mode?
- resolution

If the user has a browser that supports media queries then we can write CSS specifically for certain situations. For example, detecting that the user has a small device like a smart phone of some description and giving them a specific layout. To see an example of this in practice, the UK web conference dConstruct has just launched their website for the 2010 conference and this uses media queries to great effect.



The dConstruct 2010 website in Safari on a desktop computer



dConstruct 2010

3RD SEPTEMBER 2010

BRIGHTON, ENGLAND

DESIGN AND CREATIVITY

Now in its sixth year, **dConstruct 2010** brings together leading industry figures to explore the power of design thinking and show how we can all become just a little bit more creative.

SPEAKERS INCLUDE:

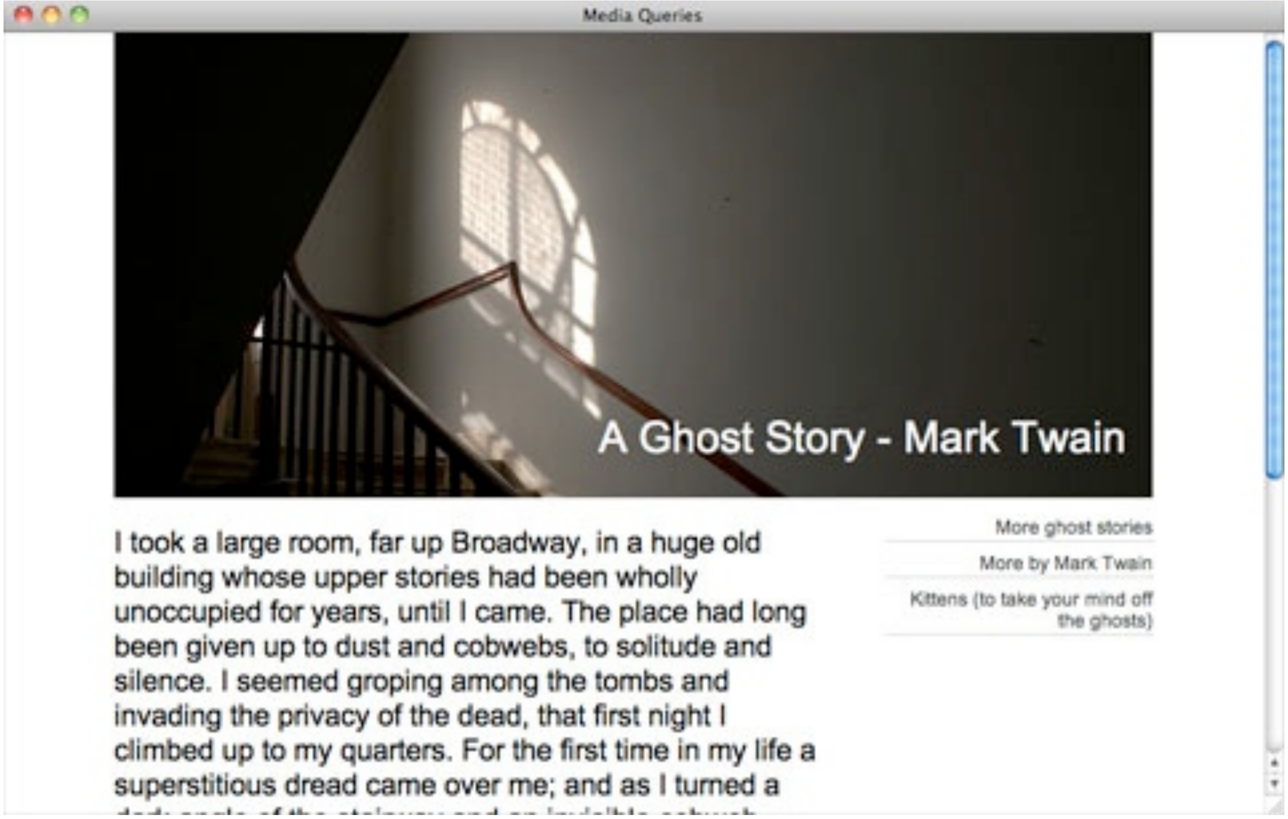
BRENDAN **DAWES**
DAVID **MCCANDLESS**
HANNAH **DONOVAN**
JAMES **BRIDLE**
JOHN **GRUBER**
MARTY **NEUMEIER**
MERLIN **MANN**

The dConstruct 2010 website on an iPhone

You can see from the above example that the site hasn't just been made smaller to fit, but that the content has actually been re-architected to be made more easy to access on the small screen of the device. In addition many people might think of this as being an iPhone layout – but it isn't. It displays in the same way on Opera Mini on my Android based phone – so by using media queries and targeting the device capabilities the dConstruct site caters for all sorts of devices – even ones they might not have thought of!

Using Media Queries to create a stylesheet for phones

To get started we can take a look at a very simple example. The below layout is a very simple two column layout.



A Ghost Story - Mark Twain

I took a large room, far up Broadway, in a huge old building whose upper stories had been wholly unoccupied for years, until I came. The place had long been given up to dust and cobwebs, to solitude and silence. I seemed groping among the tombs and invading the privacy of the dead, that first night I climbed up to my quarters. For the first time in my life a superstitious dread came over me; and as I turned a

A very simple two column layout

To make it easier to read on a phone I have decided to linearize the entire design making it all one column, and also to make the header area much smaller so readers don't need to scroll past the header before getting to any content.

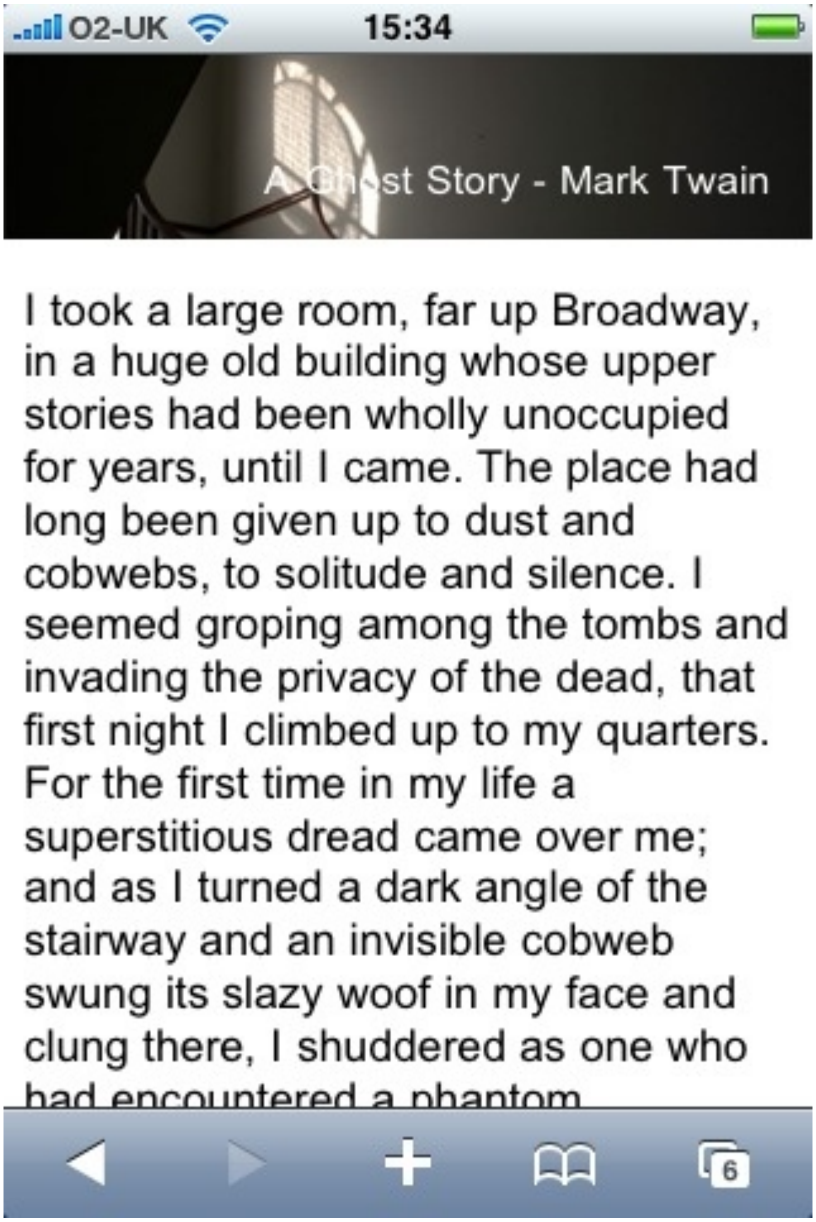
The first way to use media queries is to have the alternate section of CSS right inside your single stylesheet. So to target small devices we can use the following syntax:

```
@media only screen and (max-device-width: 480px) {

}
```

We can then add our alternate CSS for small screen and width devices inside the curly braces. By using the cascade we can simply overwrite any styles rules we set for desktop browsers earlier in our CSS. As long as this section comes last in your CSS it will overwrite the previous rules. So, to linearize our layout and use a smaller header graphic I can add the following:

```
@media only screen and (max-device-width: 480px) {
    div#wrapper {
        width: 400px;
    }

    div#header {
        background-image: url(media-queries-phone.jpg);
        height: 93px;
        position: relative;
    }

    div#header h1 {
        font-size: 140%;
    }

    #content {
        float: none;
        width: 100%;
    }

    #navigation {
        float:none;
        width: auto;
    }
}
```

In the code above I am using an alternate background image and reducing the height of the header, then setting the content and navigation to float none and overwriting the width set earlier in the stylesheet. These rules only come into play on a small screen device.



A Ghost Story - Mark Twain

I took a large room, far up Broadway, in a huge old building whose upper stories had been wholly unoccupied for years, until I came. The place had long been given up to dust and cobwebs, to solitude and silence. I seemed groping among the tombs and invading the privacy of the dead, that first night I climbed up to my quarters. For the first time in my life a superstitious dread came over me; and as I turned a dark angle of the stairway and an invisible cobweb swung its slazy woof in my face and clung there, I shuddered as one who had encountered a phantom

My simple example as displayed on an iPhone

Linking a separate stylesheet using media queries

Adding the specific code for devices inline might be a good way to use media queries if you only need to make a few changes, however if your stylesheet contains a lot of overwriting or you want to completely separate the styles shown to desktop browsers and those used for small screen devices, then linking in a different stylesheet will enable you to keep the CSS separate.

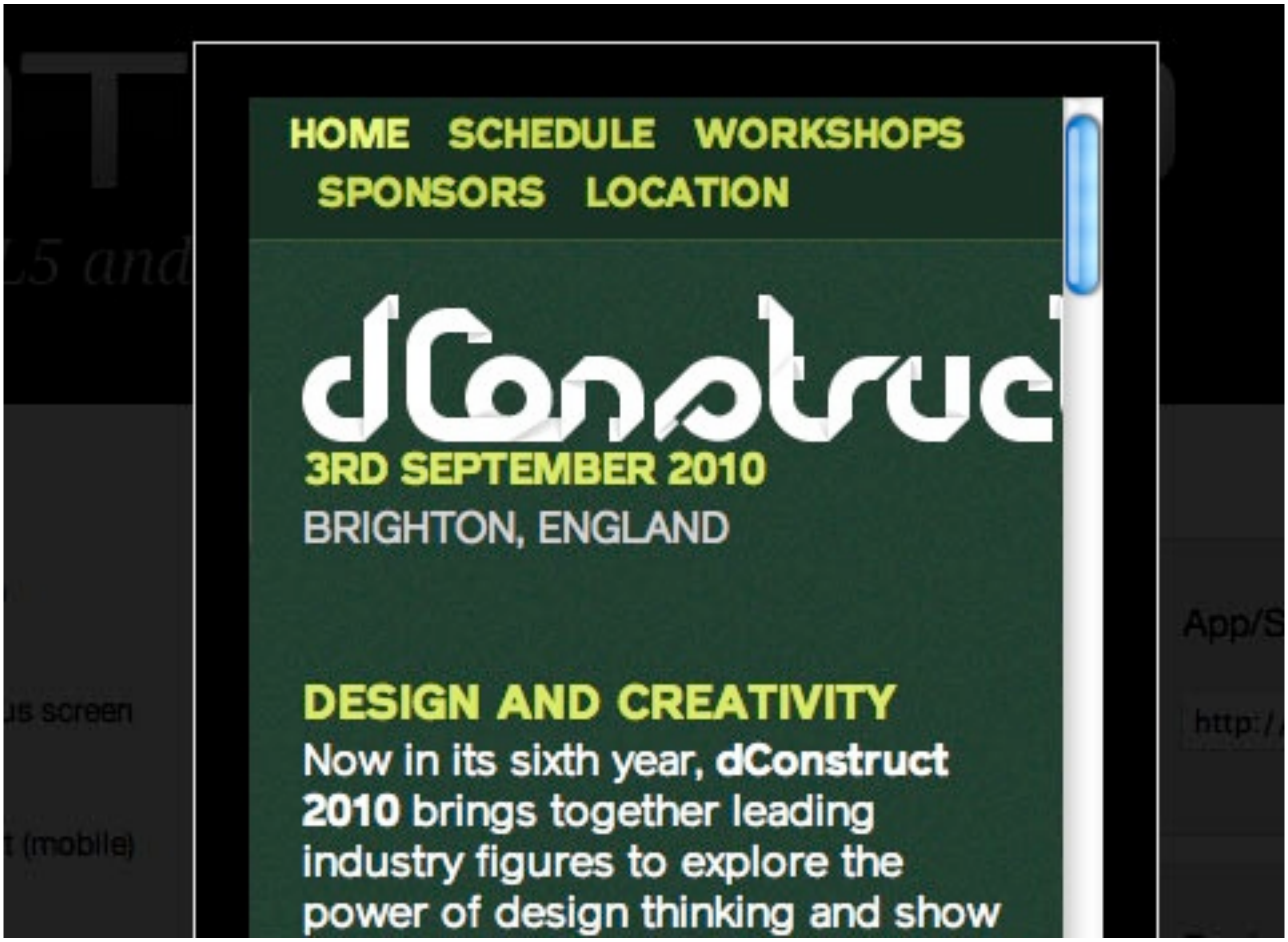
To add a separate stylesheet after your main stylesheet and use the cascade to overwrite the rules, use the following.

```
<link rel="stylesheet" type="text/css" media="only screen and
(max-device-width: 480px)" href="small-device.css" />
```

Testing media queries

If you are the owner of an iPhone, Android device or other device that has a browser which supports media queries you can test your CSS yourself. However you will need to upload the code somewhere in order to view it. What about testing devices you don’t own and testing locally?

An excellent site that can help you during the development process is ProtoFluid. This application gives you a form to enter your URL – which can be a local URL – and view the design as if in the browser on an iPhone, iPad or a range of other devices. The screenshot below is the dConstruct site we looked at earlier as seen through the iPhone view on ProtoFluid.



3RD SEPTEMBER 2010
BRIGHTON, ENGLAND

DESIGN AND CREATIVITY

Now in its sixth year, **dConstruct 2010** brings together leading industry figures to explore the power of design thinking and show

The dConstruct 2010 website in ProtoFluid

You can also enter in your own window size if you have a specific device you want to test for and know the dimensions of it's screen.

To use ProtoFluid you need to slightly modify the media query shown earlier to include max-width as well as max-device-width. This means that the media query also comes into play if the user has a normal desktop browser but using a very tiny window.

```
   @media only screen and (max-width: 480px), only screen and
  (max-device-width: 480px) {

   }
```

After updating your code to the above, just refresh your page in the browser and then drag the window in and you should see the layout change as it hits 480 pixels. The media queries are now reacting when the viewport width hits the value you entered.

You are now all ready to test using ProtoFluid. The real beauty of ProtoFluid is that you can still use tools such as FireBug to tweak your design, something you won't have once on the iPhone. Obviously you should still try and get a look at your layout in as many devices as possible, but ProtoFluid makes development and testing much simpler.

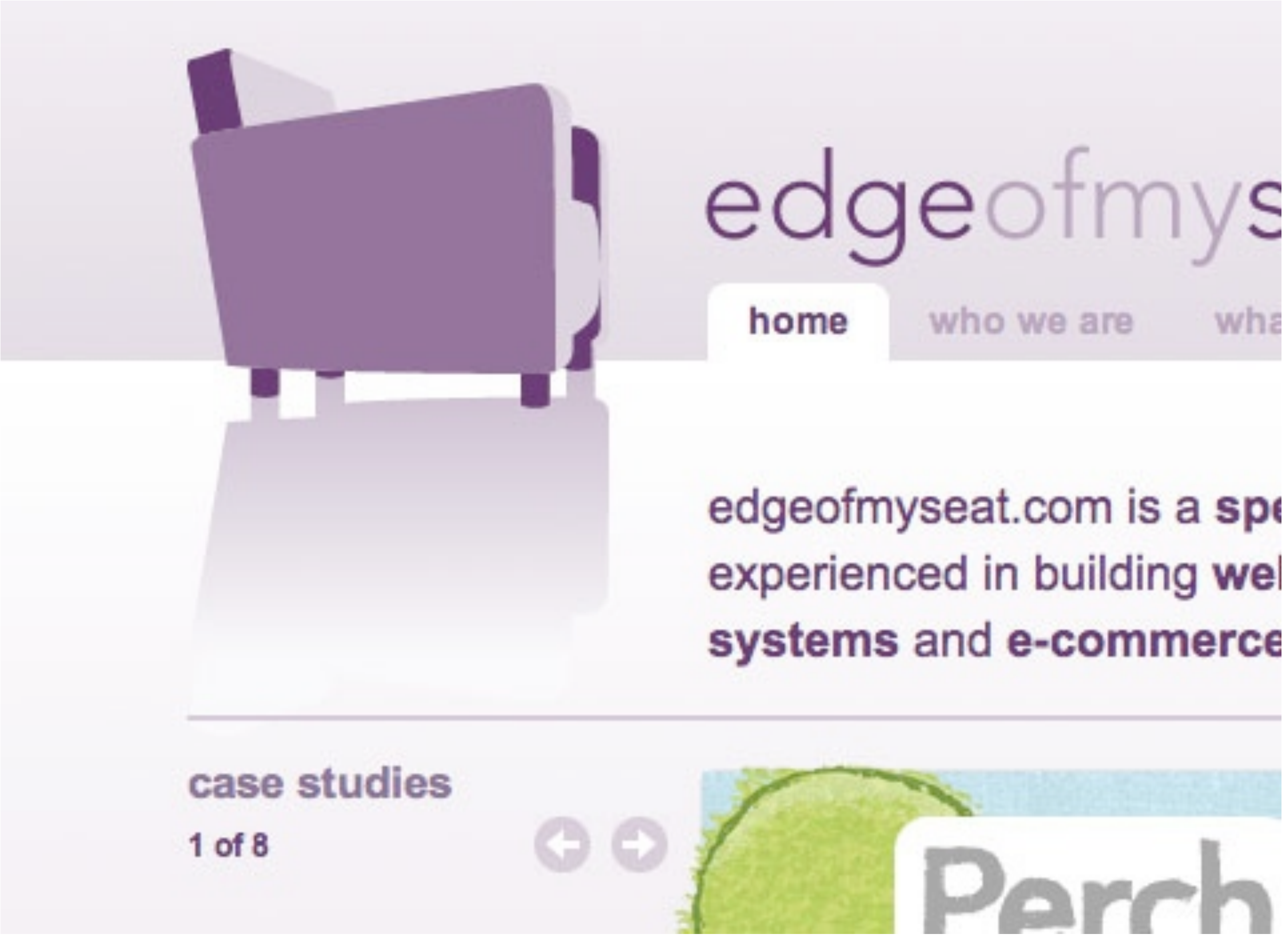
Note that if you don't want your site to switch layout when someone drags their window narrow you can always remove the max-width part of the query before going live, so the effect only happens for people with a small device and not just a small browser window.

Retrofitting an existing site

I have kept the example above very simple in order to demonstrate the technique. However this technique could very easily be used to retrofit an existing site with a version for small screen devices. One of the big selling points of using CSS for layout was this ability to provide alternate versions of our design. As an experiment I decided to take my own business website and apply these techniques to the layout.

THE DESKTOP LAYOUT

The website for my business currently has a multi-column layout. The homepage is a little different but in general we have a fixed width 3 column layout. This design is a couple of years old so we didn't consider media queries when building it.



edgeofmyseat.com is a spe
experienced in building wel
systems and e-commerce

My site in a desktop browser

ADDING THE NEW STYLESHEET

There will be a number of changes that I need to make to linearize this layout so I'm going to add a separate stylesheet using media queries to load this stylesheet after the current stylesheet and only if the max-width is less than 480 pixels.

```
  <link rel="stylesheet" type="text/css" media="only screen
and (max-width: 480px), only screen and (max-device-width:
480px)" href="/assets/css/small-device.css" />
```

To create my new stylesheet I take the default stylesheet for the site and save it as small-device.css. So this starts life as a copy of my main stylesheet. What I am going to do is go through and overwrite certain rules and then delete anything I don't need.

SHRINKING THE HEADER

The first thing I want to do is make the logo fit nicely on screen for small devices. As the logo is a background image this is easy to do as I can load a different logo in this stylesheet. I also have a different background image with a shorter top area over which the logo sits.

```
body {
    background-image: url(/img/small-bg.png);
}

#wrapper {
    width: auto;
    margin: auto;
    text-align: left;
    background-image: url(/img/small-logo.png);
    background-position: left 5px;
    background-repeat: no-repeat;
    min-height: 400px;
}
```

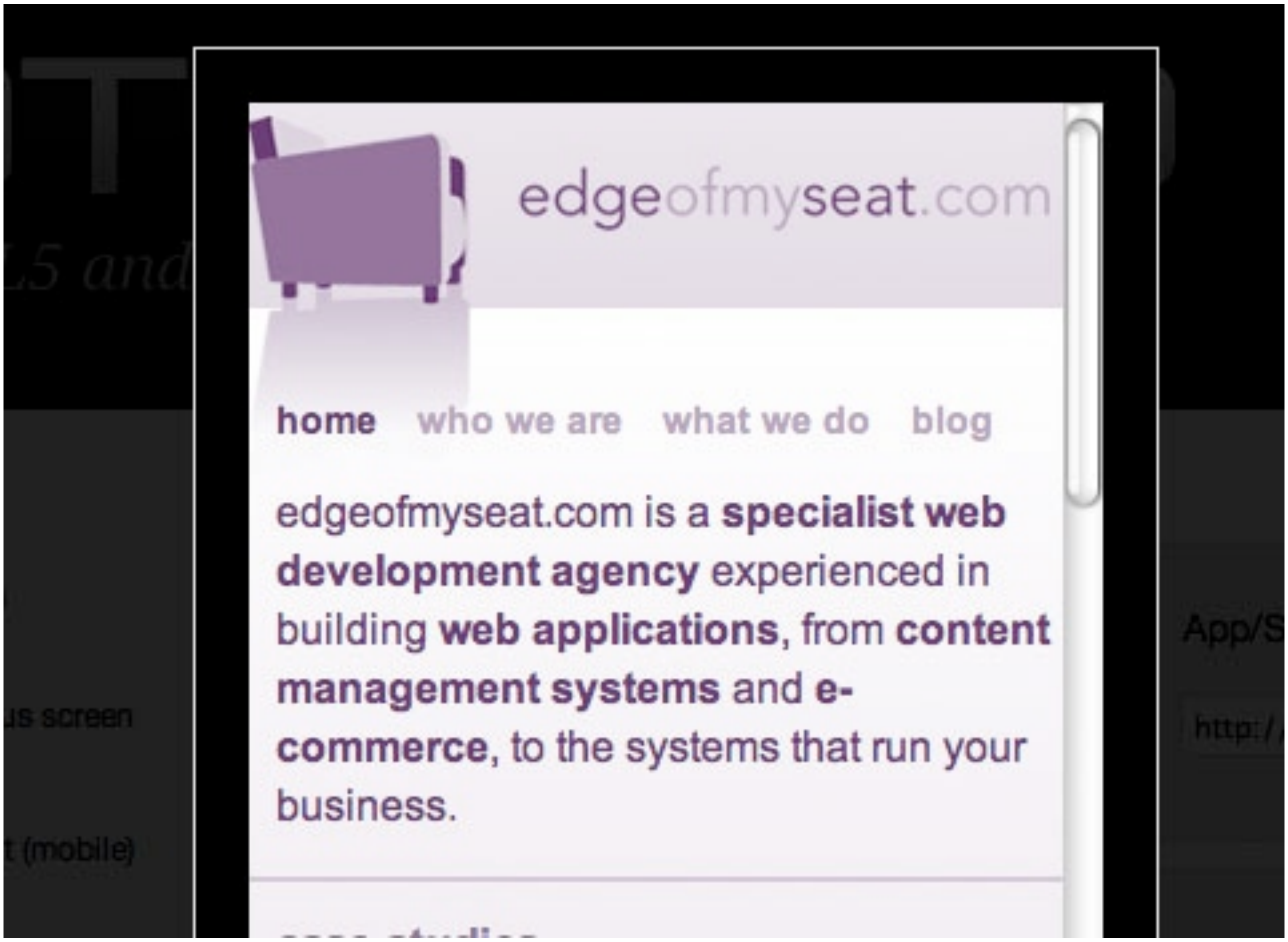
LINEARIZING THE LAYOUT

The next main job is to linearize the layout and make it all one column. The desktop layout is created using floats so all I need to do is find the rules that float the columns, set them to float: none and width:auto. This drops all the columns one under another.

```
.article #aside {
    float: none;
    width: auto;
}
```

TIDYING UP

Everything after this point is really just a case of looking at the layout in ProtoFluid and tweaking it to give sensible amounts of margin and padding to areas that now are stacked rather than in columns. Being able to use Firebug in ProtoFluid makes this job much easier as my workflow mainly involves playing around using Firebug until I am happy with the effect and then copying that CSS into the stylesheet.



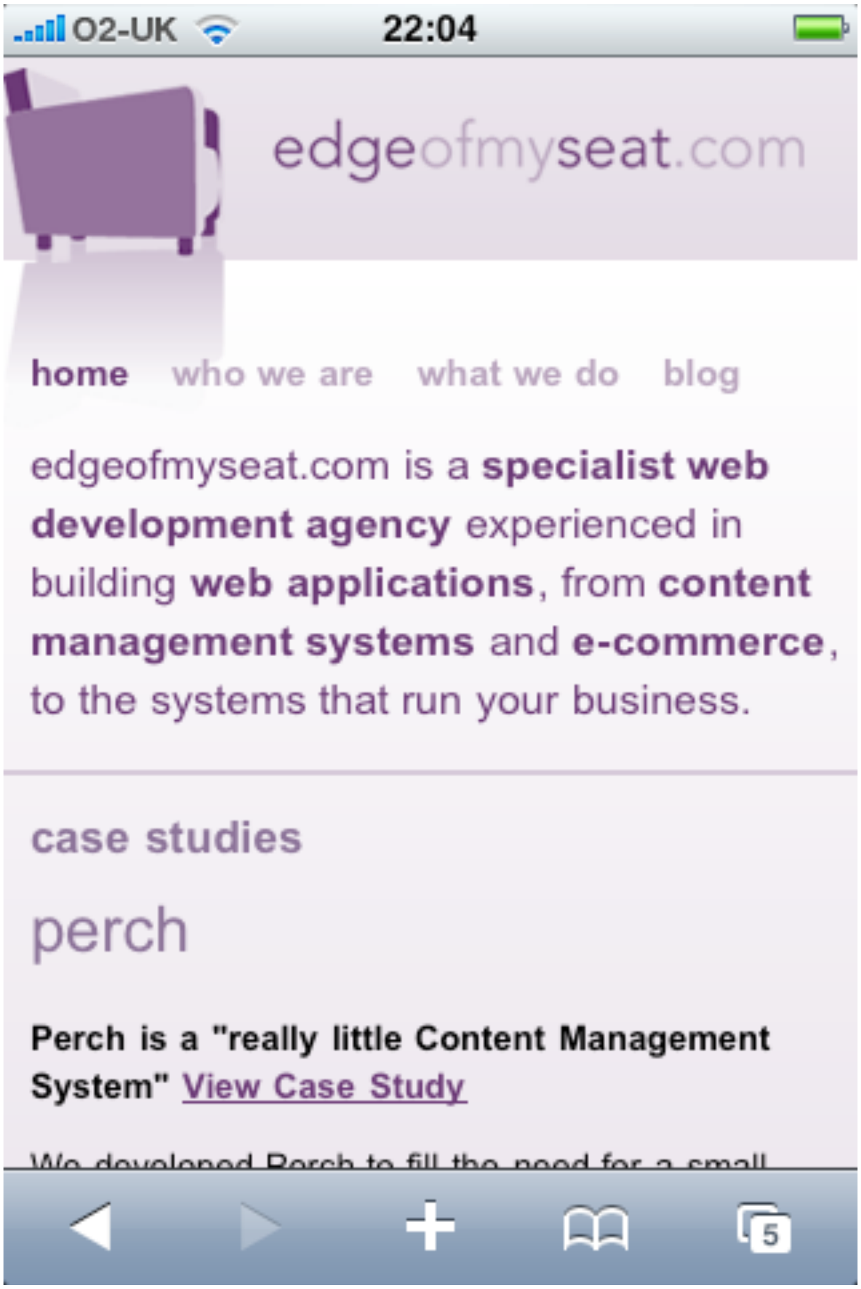
edgeofmyseat.com is a **specialist web development agency** experienced in building **web applications**, from **content management systems** and **e-commerce**, to the systems that run your business.

Testing the site using ProtoFluid

TESTING IN AN IPHONE

Having created my stylesheet and uploading it I wanted to check how it worked in a real target device. In the iPhone I discovered that the site still loaded zoomed out rather than zooming in on my nice readable single column. A quick search on the Safari developer website gave me my answer – to add a meta tag to the head of the website setting the width of the viewport to the device width.

```
     <meta name="viewport" content="width=device-width" />
After adding the meta tag the site now displays zoomed in one
the single column.
```



edgeofmyseat.com is a **specialist web development agency** experienced in building **web applications**, from **content management systems** and **e-commerce**, to the systems that run your business.

case studies

perch

Perch is a "really little Content Management System" View Case Study

We developed Perch to fill the need for a small

The site as it now displays on an iPhone

This was a very simple retrofit to show that it is possible to add a mobile version of your site simply. If I was building a site from scratch that I would be using media queries on, there are definitely certain choices I would make to make the process simpler. For example considering the linearized column orders, using background images where possible as these can be switched using CSS – or perhaps using fluid images.

Our desktop layout features a case studies carousel on the homepage, this wasn't easy to interact with on a touch screen device and so I checked using JavaScript if the browser window was very narrow and didn't launch the carousel. The way this was already written meant that the effect of stopping the carousel loading was that one case study would appear on the screen, which seems a reasonable solution for people on a small device. With a bit more time I could rewrite that carousel with an alternate version for users of mobile devices, perhaps with interactions more suitable to a touch screen.

Providing support for Media Queries in older browsers

This article covers the use of media queries in devices that have native support. If you are only interested in supporting iPhone and commonly used mobile browsers such as Opera Mini you have the luxury of not needing to worry about non-supporting browsers. If you want to start using media queries in desktop browsers then you might be interested to discover that there are a couple of techniques available which use JavaScript to add support to browsers such as Internet Explorer 8 (and lower versions) and Firefox prior to 3.5. Internet Explorer 9 will have support for CSS3 Media Queries.

More inspiration

To see more interesting use of Media Queries have a look at Hicksdesign where Jon Hicks has used Media Queries to not only provide a better experience for mobile device users, but also for regular web browser users with smaller windows. Also, have a look at Simon Collison's website and Ed Merritt's portfolio for other examples of this technique.

Try it for yourself

Using Media Queries is one place you can really start to use CSS3 in your daily work. It is worth remembering that the browsers that support media queries also support lots of other CSS3 properties so your stylesheets that target these devices can also use other CSS3 to create a slick effect when viewed on an iPhone or other mobile device.

Creating Mobile-Optimized Websites Using WordPress

Rachel McCollin

“Mobile Web design.” Unless you’ve been hiding under a bush for the last 18 months, you’ll know that it’s one of the hottest topics in the industry at the moment. Barely a week goes by without new tips being unveiled to help us hone our skills in making websites work as well — and as fast — as possible on mobile devices.

If you own or have designed a WordPress website for the desktop and are considering going mobile, the process can be fairly daunting. You probably know of responsive design and might have heard of the mobile-first approach developed by Luke Wroblewski, which entails planning the content and design for mobile devices first and then desktops second, rather than the other way round.

But if your WordPress website has a desktop theme in which everything is set in pixels, then the thought of adopting a responsive design might have you running for the hills.

It doesn’t have to be that way.

Here are four ways to make your WordPress blog or website mobile-friendly, ranging from the quick and dirty to the complex but potentially very beautiful. As well as outlining the pros and cons of these methods, we’ll include information on plugins that will help without actually doing all the work for you, and we’ll provide some code that you can use for a responsive design.

Plugins: The Quick Way To Make Your Content Mobile-Friendly

Designing for content is increasingly becoming more common than squeezing content into a pixel-perfect design, as documented on Smashing Magazine.

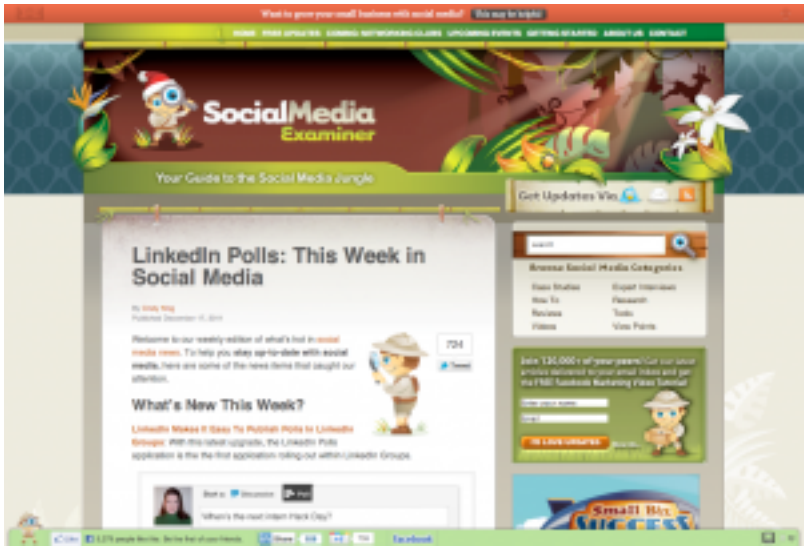
If your website is more about content than design (say you run a blog that is content-heavy and designed for reading), then you won't be too fussed about what your website looks like on mobile devices. You just want people to be able to read it without having to zoom in, move the viewport around or generally tie themselves up in knots until they decide to leave.

If this is the case, then a simple plugin might do the trick. Below are some plugins to consider.

WPTOUCH

WPtouch, which comes in free and premium versions, strips out your existing theme and displays your content and not much else, but the result is user-friendly, robust and easy to read.

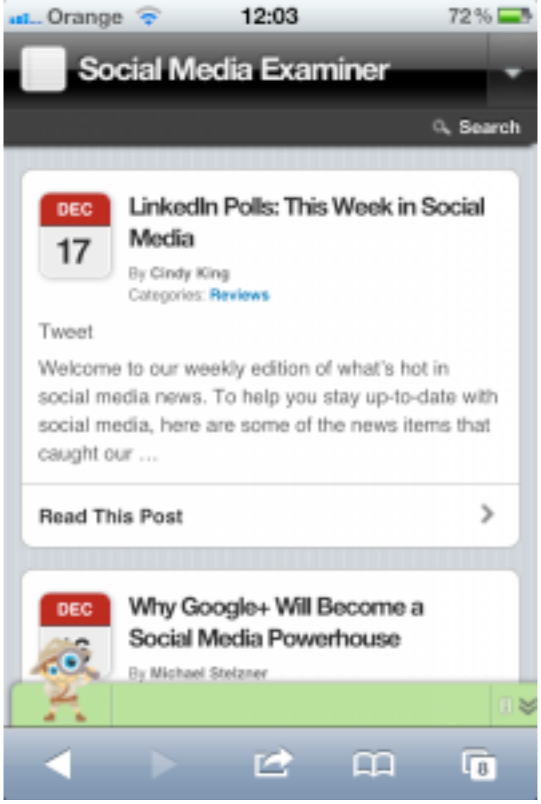
WPtouch is widely used on websites, including Stephen Fry's blog and Social Media Examiner. You can see below how the plugin renders those two websites. The premium version has options to modify the colors and some styles, including a bespoke menu at the bottom of the screen, as seen on Social Media Examiner.



LinkedIn Polls: This Week in Social Media

What's New This Week?

Social Media Examiner desktop design



DEC 17

LinkedIn Polls: This Week in Social Media

By Cindy King

Categories: Reviews

Tweet

Welcome to our weekly edition of what's hot in social media news. To help you stay up-to-date with social media, here are some of the news items that caught our …

Read This Post

DEC

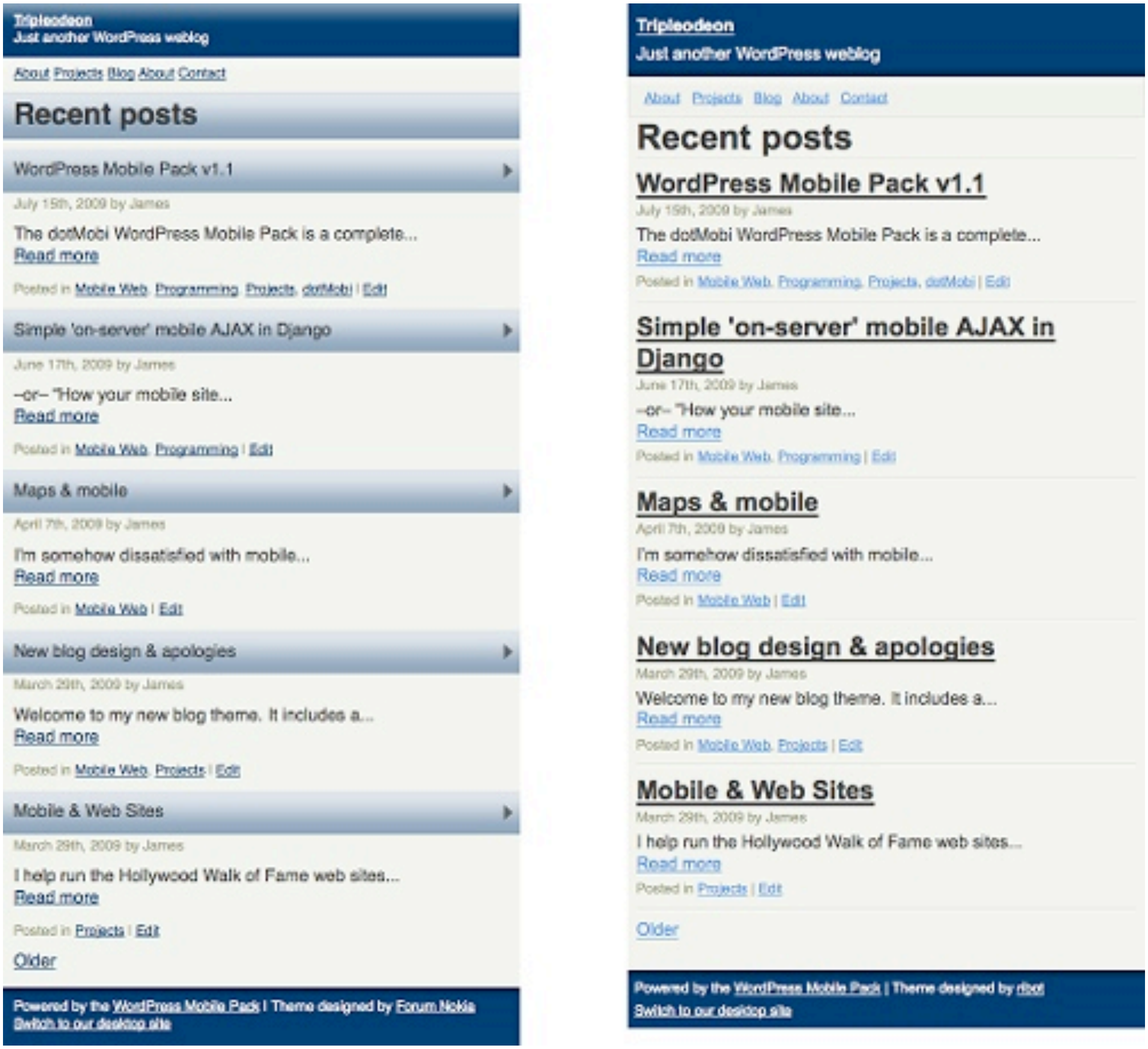
Why Google+ Will Become a Social Media Powerhouse

By Michael Stelzner

Social Media Examiner mobile design, using WPtouch

WORDPRESS MOBILE PACK

The WordPress Mobile Pack has some color options and can be used as a mobile switcher if you want a completely different theme for mobile devices. It also has a mobile interface for editing posts, although this has been superseded to some extent by the WordPress apps for iOS and Android.



Tripleodeon
Just another WordPress weblog

About Projects Blog About Contact

Recent posts

WordPress Mobile Pack v1.1
July 15th, 2009 by James
The dotMobi WordPress Mobile Pack is a complete...
Read more
Posted in Mobile Web, Programming, Projects, dotMobi | Edit

Simple 'on-server' mobile AJAX in Django
June 17th, 2009 by James
–or– "How your mobile site...
Read more
Posted in Mobile Web, Programming | Edit

Maps & mobile
April 7th, 2009 by James
I'm somehow dissatisfied with mobile...
Read more
Posted in Mobile Web | Edit

New blog design & apologies
March 29th, 2009 by James
Welcome to my new blog theme. It includes a...
Read more
Posted in Mobile Web, Projects | Edit

Mobile & Web Sites
March 29th, 2009 by James
I help run the Hollywood Walk of Fame web sites...
Read more
Posted in Projects | Edit

Older

Powered by the WordPress Mobile Pack | Theme designed by Forum Nokia
Switch to our desktop site

Tripleodeon
Just another WordPress weblog

About Projects Blog About Contact

Recent posts

WordPress Mobile Pack v1.1
July 15th, 2009 by James
The dotMobi WordPress Mobile Pack is a complete...
Read more
Posted in Mobile Web, Programming, Projects, dotMobi | Edit

Simple 'on-server' mobile AJAX in Django
June 17th, 2009 by James
–or– "How your mobile site...
Read more
Posted in Mobile Web, Programming | Edit

Maps & mobile
April 7th, 2009 by James
I'm somehow dissatisfied with mobile...
Read more
Posted in Mobile Web | Edit

New blog design & apologies
March 29th, 2009 by James
Welcome to my new blog theme. It includes a...
Read more
Posted in Mobile Web, Projects | Edit

Mobile & Web Sites
March 29th, 2009 by James
I help run the Hollywood Walk of Fame web sites...
Read more
Posted in Projects | Edit

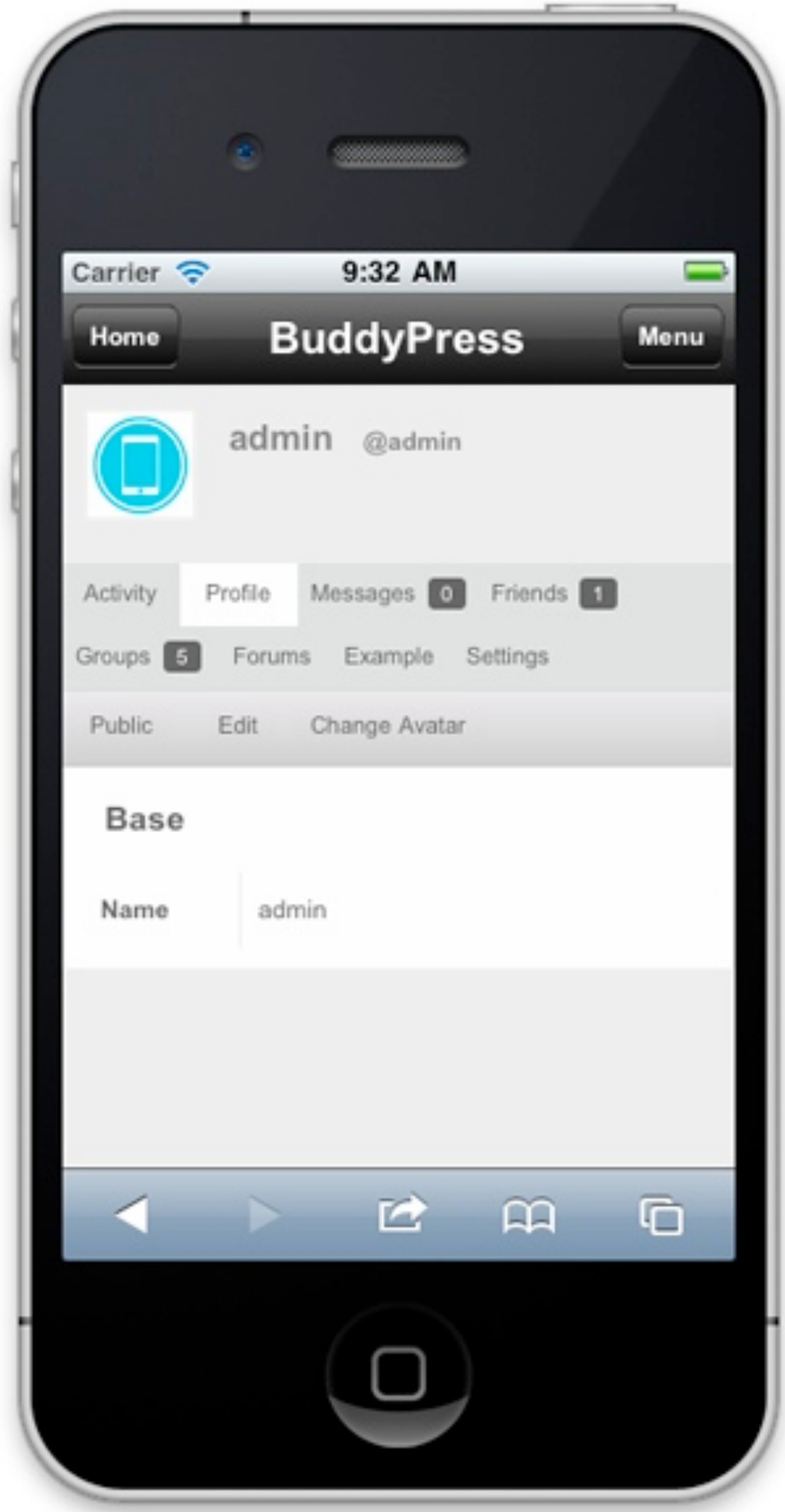
Older

Powered by the WordPress Mobile Pack | Theme designed by ribot
Switch to our desktop site

WordPress Mobile Pack screenshots

BUDDYPRESS MOBILE

If your website runs BuddyPress, then you'll need a plugin to ensure that none of its functionality is lost on mobile devices. BuddyPress Mobile has theming options, and you can edit the style sheet to make the mobile design your own.



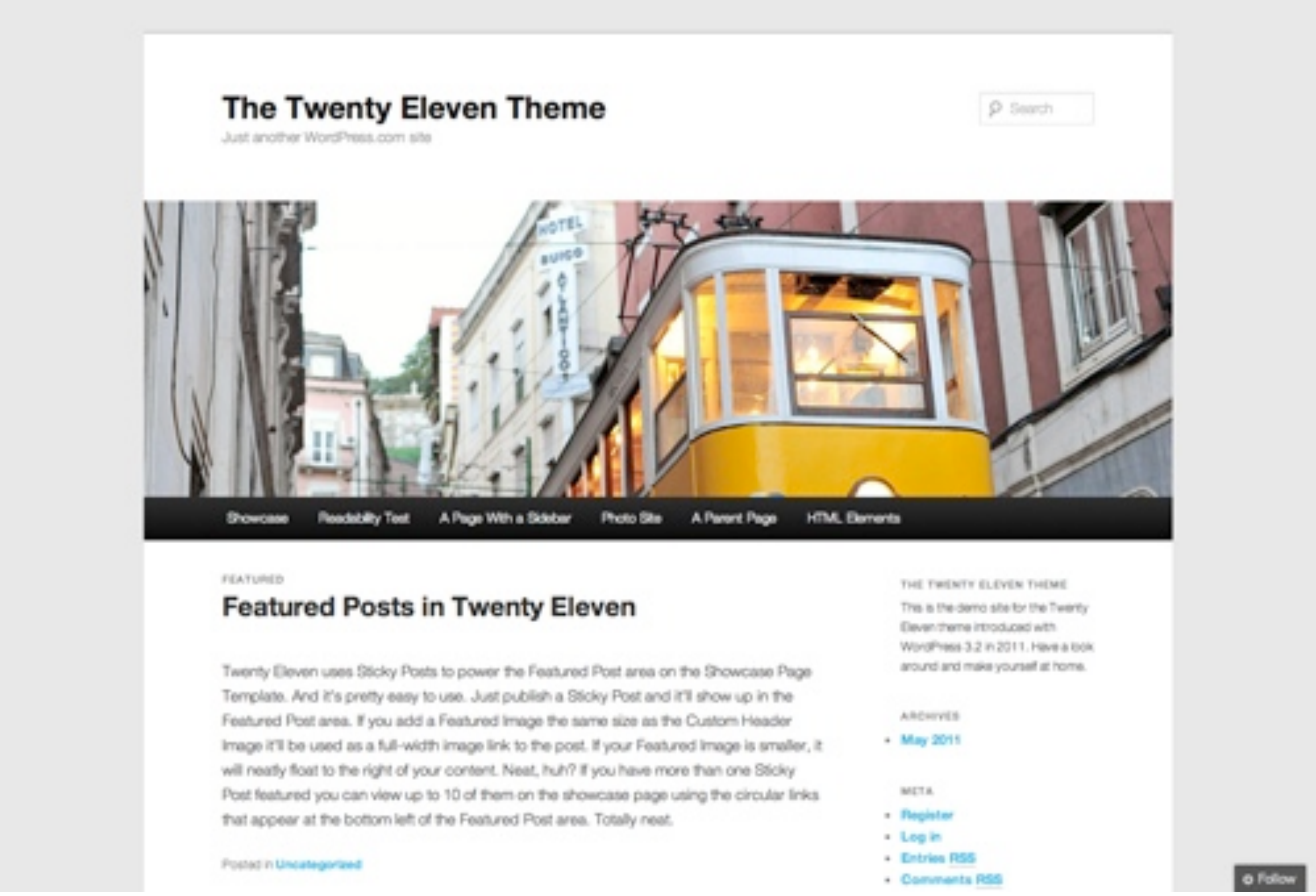
Name admin

BuddyPress Mobile

Mobile Themes: The Next Level Up

If you want a consistent design across desktop and mobile, but you don't yet have a theme or you want to develop one, then a mobile theme might be the answer.

More and more mobile themes have sprung up over the last year. In particular, Twenty Eleven, WordPress' default theme since version 3.0, is responsive enough for many websites.



The Twenty Eleven Theme

Just another WordPress.com site

FEATURED

Featured Posts in Twenty Eleven

Twenty Eleven uses Sticky Posts to power the Featured Post area on the Showcase Page Template. And it's pretty easy to use. Just publish a Sticky Post and it'll show up in the Featured Post area. If you add a Featured Image the same size as the Custom Header Image it'll be used as a full-width image link to the post. If your Featured Image is smaller, it will neatly float to the right of your content. Neat, huh? If you have more than one Sticky Post featured you can view up to 10 of them on the showcase page using the circular links that appear at the bottom left of the Featured Post area. Totally neat.

Posted in Uncategorized

THE TWENTY ELEVEN THEME

This is the demo site for the Twenty Eleven theme introduced with WordPress 3.2 in 2011. Have a look around and make yourself at home.

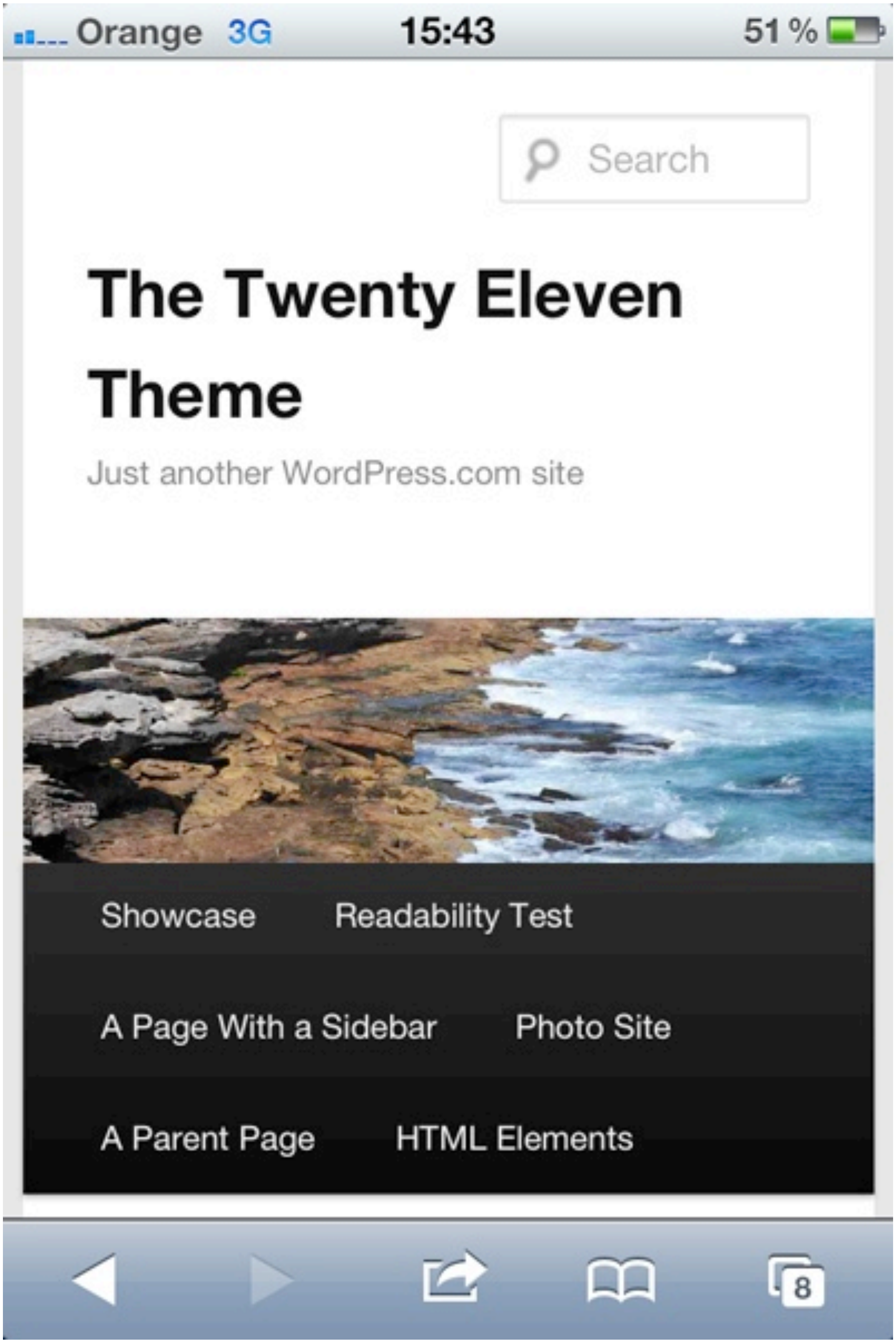
ARCHIVES

- May 2011

META

- Register
- Log in
- Entries RSS
- Comments RSS

Twenty Eleven on the desktop



The Twenty Eleven Theme

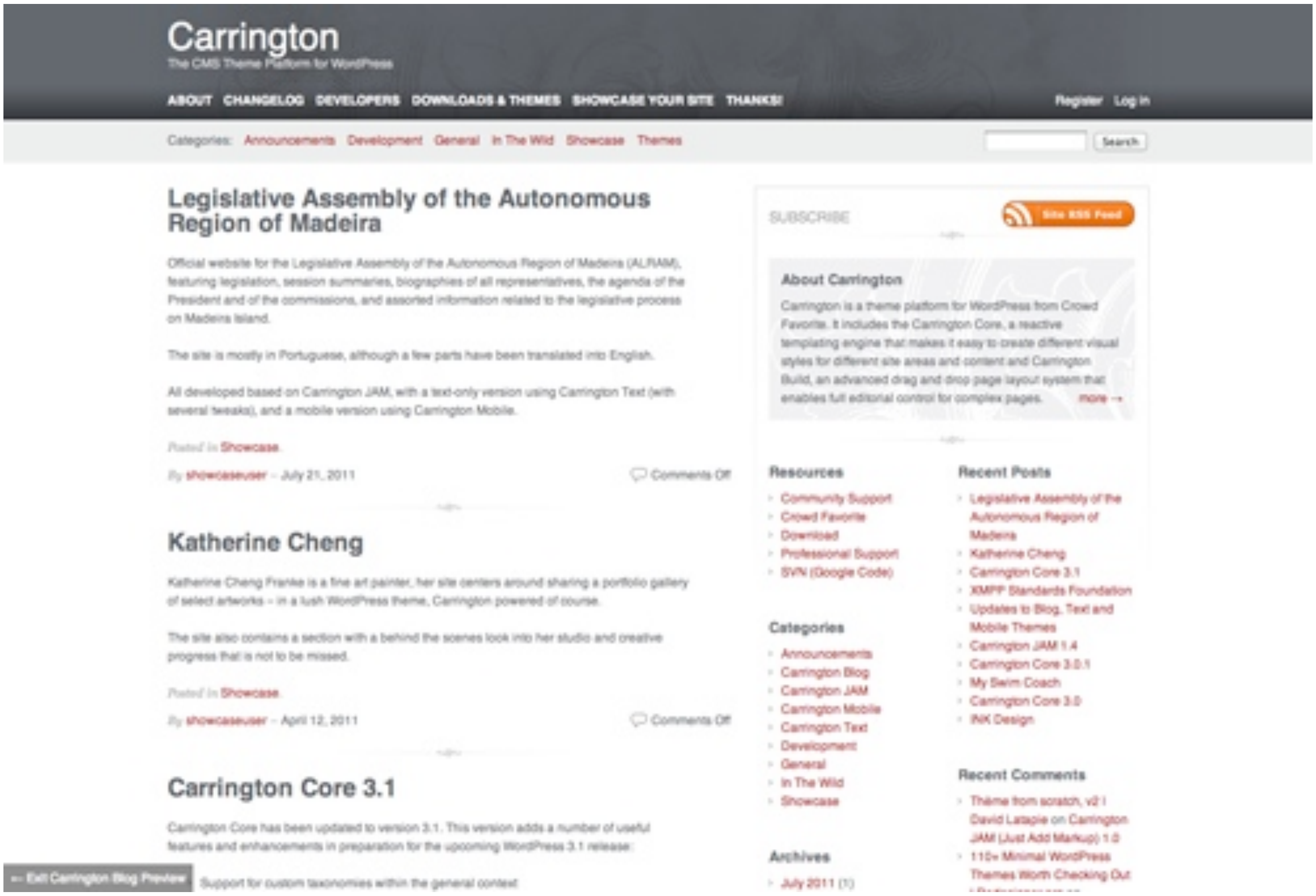
Just another WordPress.com site

Twenty Eleven on mobile

Below are some other themes that include a mobile or responsive style sheet.

CARRINGTON

The Carrington family of themes can be used as parent themes. You can edit the CSS and functions to suit your needs, and it has a mobile version.



Legislative Assembly of the Autonomous Region of Madeira

Official website for the Legislative Assembly of the Autonomous Region of Madeira (ALRAM), featuring legislation, session summaries, biographies of all representatives, the agenda of the President and of the commissions, and assorted information related to the legislative process on Madeira Island.

The site is mostly in Portuguese, although a few parts have been translated into English.

All developed based on Carrington JAM, with a text-only version using Carrington Text (with several tweaks), and a mobile version using Carrington Mobile.

Posted in Showcase.

By showcaseuser – July 21, 2011 Comments Off

Katherine Cheng

Katherine Cheng Franke is a fine art painter, her site centers around sharing a portfolio gallery of select artworks – in a lush WordPress theme, Carrington powered of course.

The site also contains a section with a behind the scenes look into her studio and creative progress that is not to be missed.

Posted in Showcase.

By showcaseuser – April 12, 2011 Comments Off

Carrington Core 3.1

Carrington Core has been updated to version 3.1. This version adds a number of useful features and enhancements in preparation for the upcoming WordPress 3.1 release:

Support for custom taxonomies within the general context

About Carrington

Carrington is a theme platform for WordPress from Crowd Favorite. It includes the Carrington Core, a reactive templating engine that makes it easy to create different visual styles for different site areas and content and Carrington Build, an advanced drag and drop page layout system that enables full editorial control for complex pages. more →

Resources

- Community Support
- Crowd Favorite
- Download
- Professional Support
- SVN (Google Code)

Categories

- Announcements
- Carrington Blog
- Carrington JAM
- Carrington Mobile
- Carrington Text
- Development
- General
- In The Wild
- Showcase

Archives

- July 2011 (1)

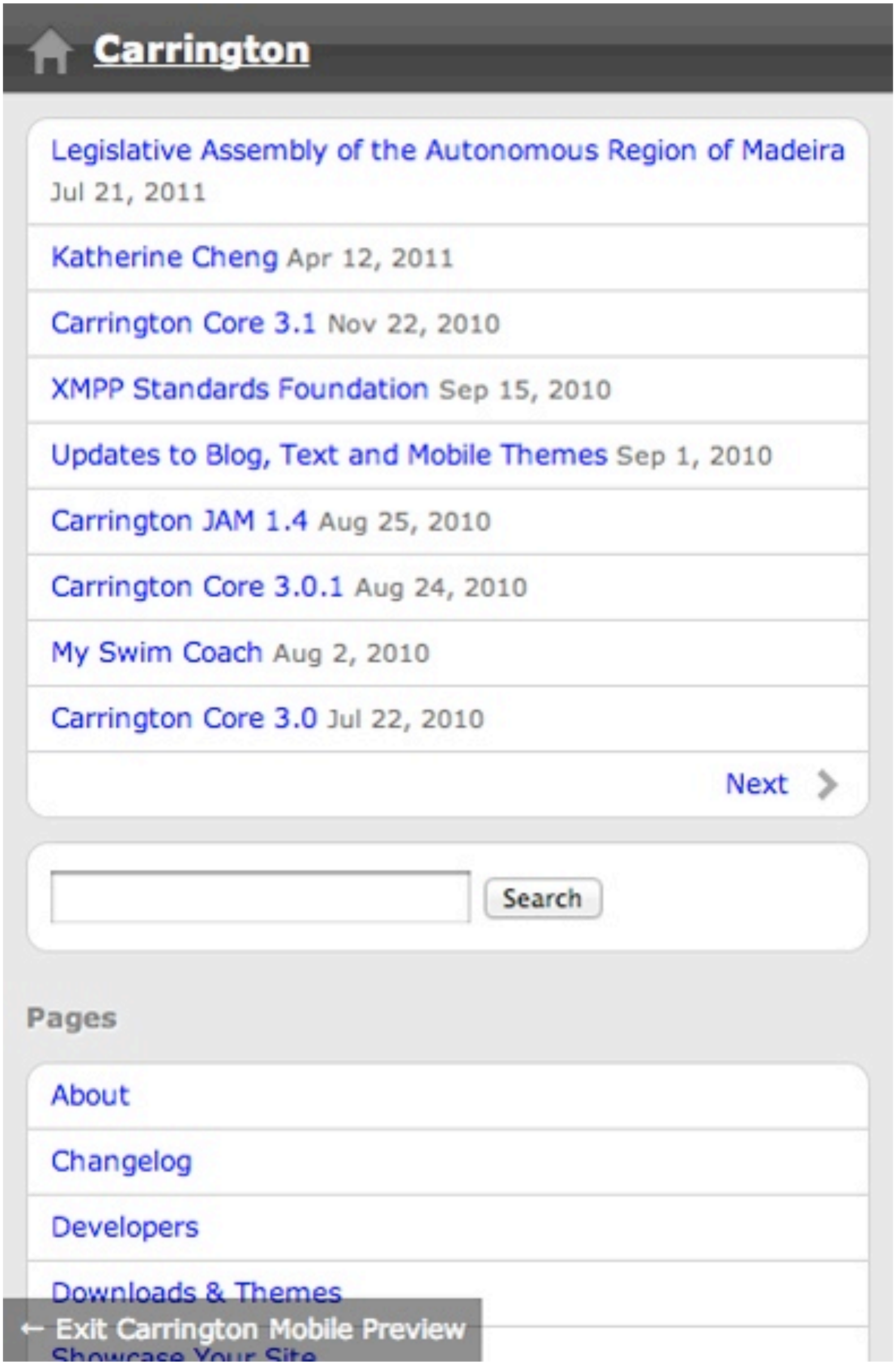
Recent Posts

- Legislative Assembly of the Autonomous Region of Madeira
- Katherine Cheng
- Carrington Core 3.1
- XMPP Standards Foundation
- Updates to Blog, Text and Mobile Themes
- Carrington JAM 1.4
- Carrington Core 3.0.1
- My Swim Coach
- Carrington Core 3.0
- INK Design

Recent Comments

- Thème from scratch, v2 | David Latapie on Carrington JAM (Just Add Markup) 1.0
- 110+ Minimal WordPress Themes Worth Checking Out

Carrington on desktop



Carrington

Legislative Assembly of the Autonomous Region of Madeira Jul 21, 2011

Katherine Cheng Apr 12, 2011

Carrington Core 3.1 Nov 22, 2010

XMPP Standards Foundation Sep 15, 2010

Updates to Blog, Text and Mobile Themes Sep 1, 2010

Carrington JAM 1.4 Aug 25, 2010

Carrington Core 3.0.1 Aug 24, 2010

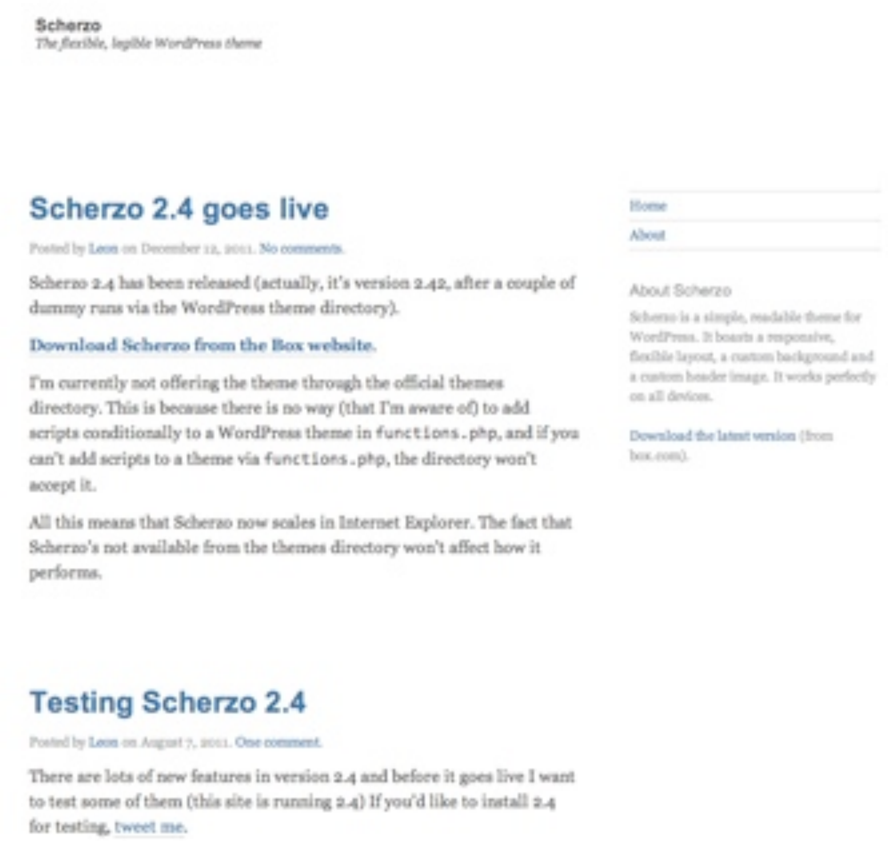
My Swim Coach Aug 2, 2010

Carrington Core 3.0 Jul 22, 2010

Carrington on mobile

SCHERZO

Scherzo is clean and minimalist and would be great to use as a parent theme. It uses a mobile-first responsive design.



Scherzo
The flexible, legible WordPress theme

Scherzo 2.4 goes live

Posted by Leon on December 12, 2011. No comments.

Scherzo 2.4 has been released (actually, it's version 2.42, after a couple of dummy runs via the WordPress theme directory).

Download Scherzo from the Box website.

I'm currently not offering the theme through the official themes directory. This is because there is no way (that I'm aware of) to add scripts conditionally to a WordPress theme in `functions.php`, and if you can't add scripts to a theme via `functions.php`, the directory won't accept it.

All this means that Scherzo now scales in Internet Explorer. The fact that Scherzo's not available from the themes directory won't affect how it performs.

Testing Scherzo 2.4

Posted by Leon on August 7, 2011. One comment.

There are lots of new features in version 2.4 and before it goes live I want to test some of them (this site is running 2.4) If you'd like to install 2.4 for testing, tweet me.

Home

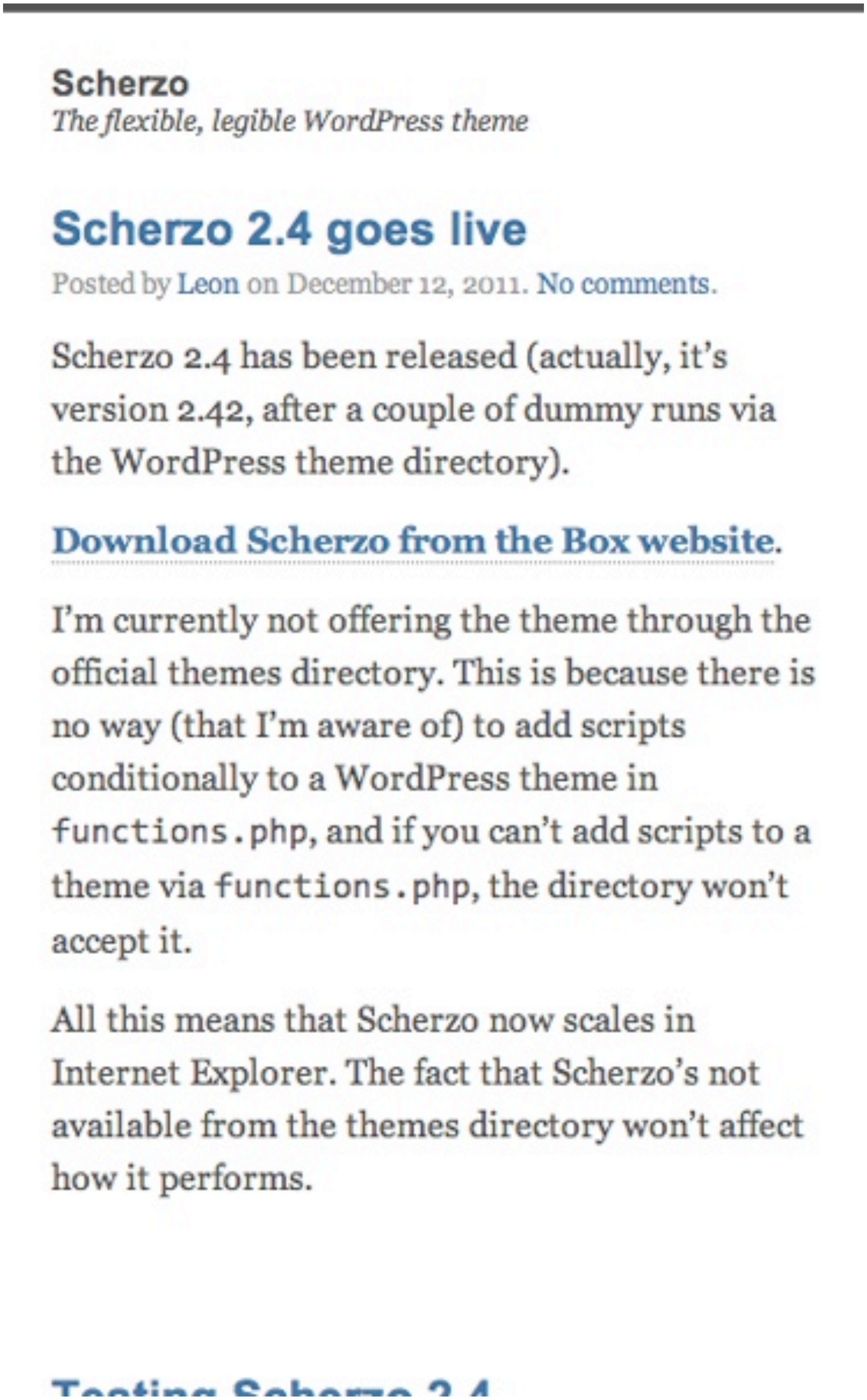
About

About Scherzo

Scherzo is a simple, readable theme for WordPress. It boasts a responsive, flexible layout, a custom background and a custom header image. It works perfectly on all devices.

Download the latest version (from box.com).

Scherzo on desktop



Scherzo

The flexible, legible WordPress theme

Scherzo 2.4 goes live

Posted by Leon on December 12, 2011. No comments.

Scherzo 2.4 has been released (actually, it's version 2.42, after a couple of dummy runs via the WordPress theme directory).

Download Scherzo from the Box website.

I'm currently not offering the theme through the official themes directory. This is because there is no way (that I'm aware of) to add scripts conditionally to a WordPress theme in `functions.php`, and if you can't add scripts to a theme via `functions.php`, the directory won't accept it.

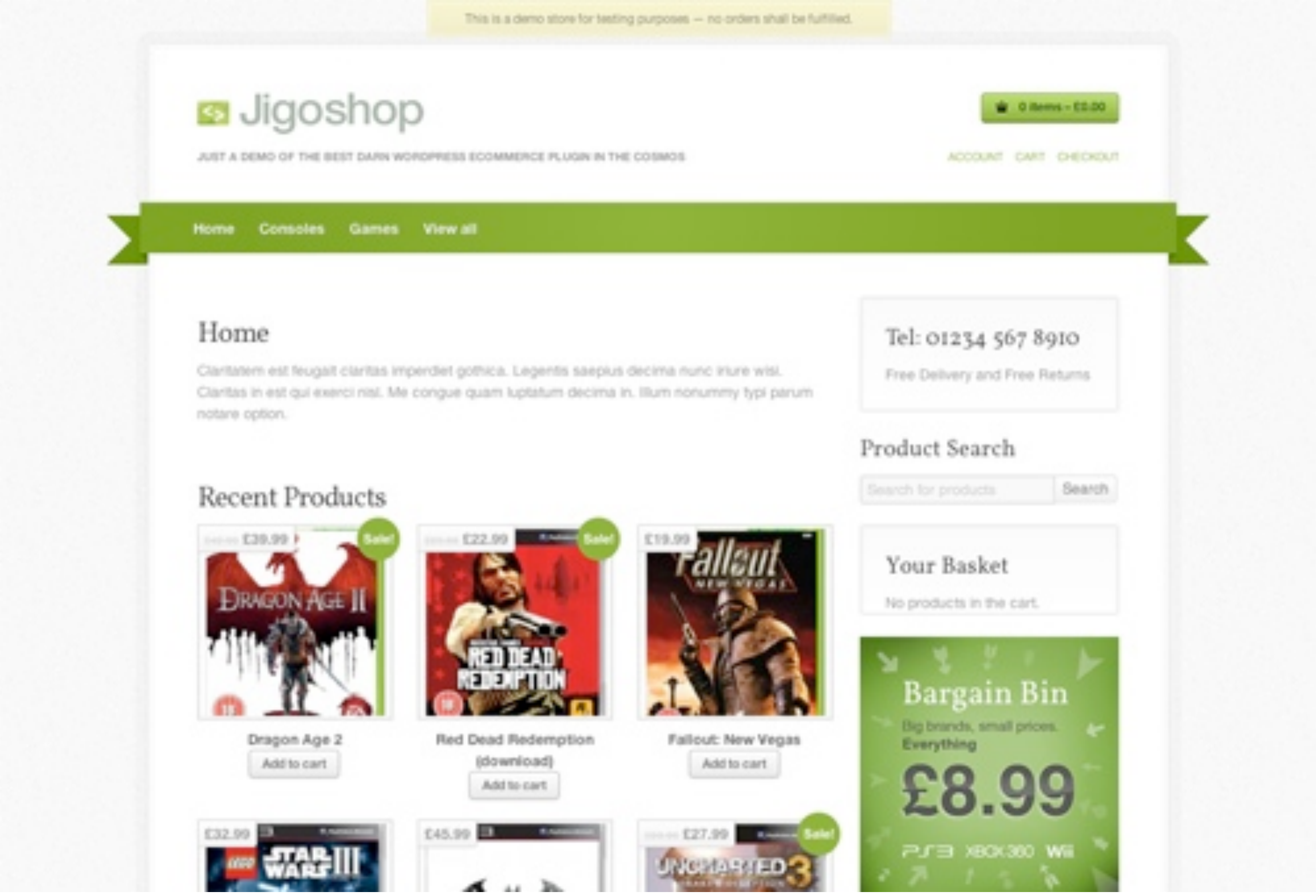
All this means that Scherzo now scales in Internet Explorer. The fact that Scherzo's not available from the themes directory won't affect how it performs.

Testing Scherzo 2.4

Scherzo on mobile

JIGOSHOP

E-commerce websites are trickier to make mobile-friendly, but Jigoshop can help. It’s a full e-commerce plugin and theme, with a responsive layout that can be tweaked to suit your design.



This is a demo store for testing purposes — no orders shall be fulfilled.

Home

Claritatem est feugait claritas imperdiet gothica. Legentis saepius decima nunc iriure wisi. Claritas in est qui exerci nisl. Me congue quam luptatum decima in. Illum nonummy typi parum notare option.

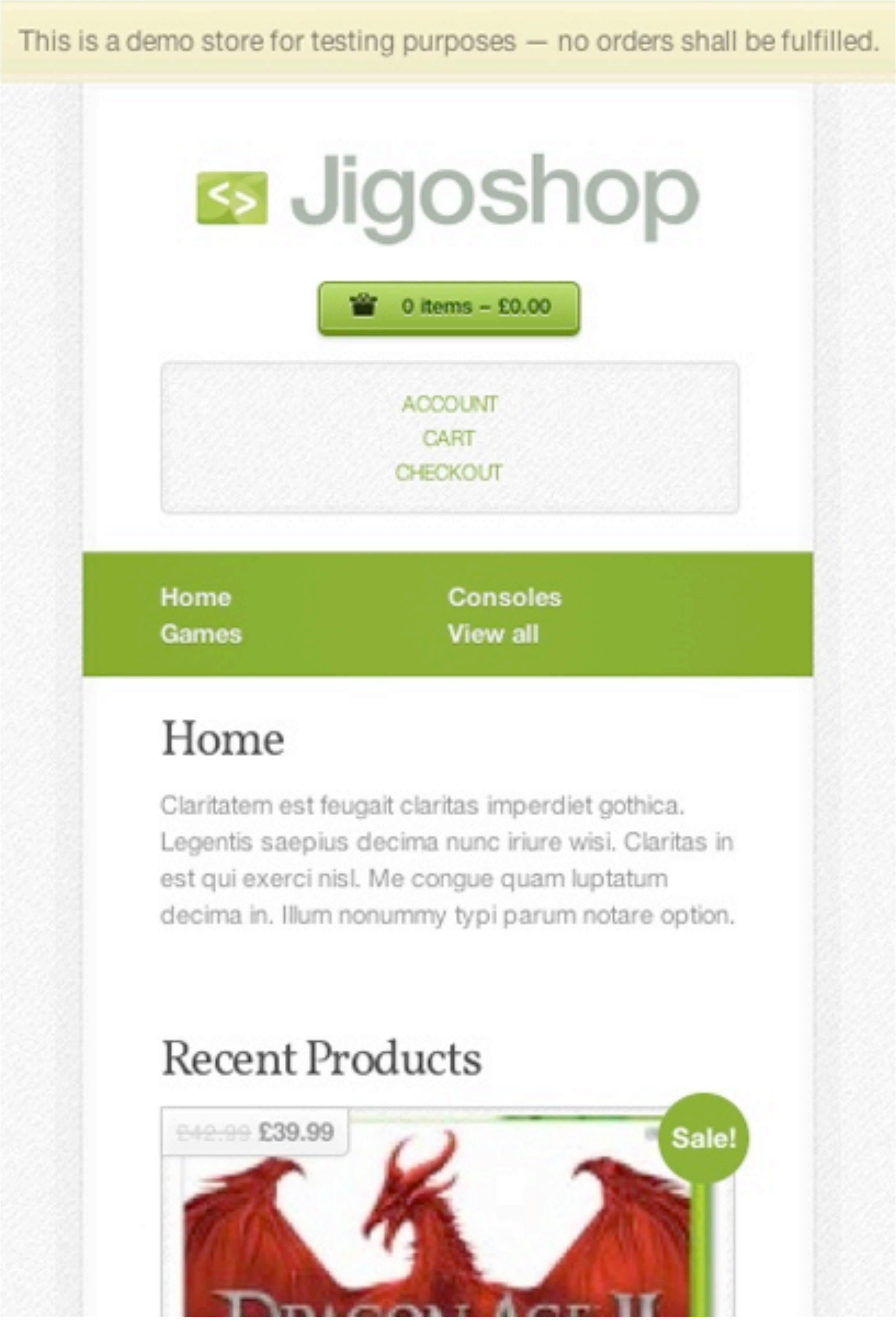
Tel: 01234 567 8910

Free Delivery and Free Returns

Your Basket

No products in the cart.

Jigoshop on desktop



This is a demo store for testing purposes — no orders shall be fulfilled.

Home

Claritatem est feugait claritas imperdiet gothica. Legentis saepius decima nunc iriure wisi. Claritas in est qui exerci nisl. Me congue quam luptatum decima in. Illum nonummy typi parum notare option.

Recent Products

Jigoshop on mobile

A Different Theme For Mobile Devices

In the days before responsive design gained traction, websites commonly had two versions: desktop and mobile. The mobile version might have been on an `m.` subdomain or have a `.mobi` extension. Some websites out there still do this, mainly huge news websites that serve different content depending on the device.

Fewer WordPress administrators are choosing to do this now, but if you do want to go down this route, then serving two versions of your website from the same database is possible, by using a mobile switcher.

Here are two plugins that make this possible:

- WordPress Mobile Pack
  This tool, already mentioned above as a theme that makes your website mobile-friendly, can also be used as a mobile switcher, detecting mobile devices and using a separate theme of your choice.
- WPtap Mobile Detector
  This targets mobile devices and enables the theme of your choice.

Using one of these plugins enables you to develop a completely separate theme for mobile devices, with its own layout, navigation and content structure.

Or, Finally, Make Your Current Theme Responsive

If you don't want to throw out your existing theme, then the best way to give mobile users an experience that is at least visually similar to the desktop version is to build responsiveness into your theme.

A responsive theme contains media queries in the theme's style sheet to define CSS that applies only to devices of a specified maximum or minimum width. A truly responsive theme has a fluid layout that adapts to mobile devices and larger screens to some extent already, but with some extra styling to make the layout optimal for mobile devices.

1. DEFINING THE MEDIA QUERIES

To get started, you will need to define media queries in the style sheet. Most of the styles already in your style sheet apply to desktop and mobile, so you only need to add CSS that is different for mobile devices. This will go at the end of your theme's style sheet.

Start by defining the screen width you are developing for. There are two main approaches to this:

1. Start with the narrowest screen width you are targeting (which will usually be mobile phones in portrait orientation); add all of the CSS needed for this screen width; and then add successive media queries for wider screens. This is known as the mobile-first approach, and it has the benefit of making websites faster on mobile devices because only the CSS needed for those devices is loaded.

2. Start with the widest screen width (usually desktop monitors) and work down, adding CSS that applies to each screen width in turn. While this might slow down loading on mobile devices, it has the advantage of

working in IE 8 and below, which doesn’t understand media queries. At the moment, most websites are developed this way because they involve making an existing desktop design responsive, so this is the approach we’ll cover here.

A media query consists of three main parts:

1. The **@media** rule;

2. The media type (the most common being **print** and **screen** — we’ll use **scree**n);

3. The maximum width of the screen you are targeting.

You could have a media query to target mobile phones (and other small devices such as the iPod Touch) in portrait orientation that have a width of 320 pixels:

```
@media screen and (max-width: 320px) {

}
```

The CSS to be applied to that screen width and any screen narrower than it would be written between the braces.

An alternative to the **`@media`** rule would be to create a linked style sheet with the CSS for each screen width. But I don't do that because it adds another server request with the potential to slow the website down; and managing all of the styles becomes harder if they're in more than one place.

Here are other media queries for commonly targeted screen sizes:

- **`(max-width: 480px)`**
  Works for mobile devices in either portrait or landscape mode, because they are 480 pixels wide in landscape orientation but are still narrower than this maximum width in portrait.

- **`(max-width: 780px)`**
  Works for iPads and other large tablets in portrait mode and any screens narrower than them.

- **`(max-width: 1024px)`**
  Works for iPads in both orientations, as well as for small desktop browsers.

You can run one media query after another so that each change you make applies to the screen size you're querying, plus any widths queried further down in the style sheet. In this case, you would work with wider screens first. For example:

```
@media screen and (max-width: 480px) {

}
```

If you are ignoring tablets, you would include this media query first and add any CSS for mobile phones in both portrait and landscape modes (for example, any changes to graphics or text size). You would then follow it with this:

```
@media screen and (max-width: 320px) {
}
```

Here, we're adding any styles that apply only to phones in portrait mode (such as layout changes). You don't need to repeat the CSS that applies to both landscape and portrait modes because this will still apply. In the same way, you don't need to repeat any styles that will stay the same for desktop views because they will cascade down from the earlier parts of the style sheet.

2. MAKING THE LAYOUT RESPONSIVE

Phew! So, now we've defined media queries, and we're ready to roll with some mobile-friendly CSS. Below are the main things you will need to work on for a standard WordPress website. Let's assume your website's markup is similar to that of the Twenty Eleven theme (i.e. **`html → body → header`** (or div **`#header) → #main → #content → #primary → #secondary → footer`** (or div **`#footer`**). You might need to substitute your own elements and IDs for the ones in the examples below.

Overall width of website

You'll need to change this so that it displays correctly. Add the following code between the braces of your first media query:

```
body {
width: 100%;
float: none;
}
```

This ensures that the website's `body` fills the width of the device and removes any floats. At this point, you might also want to change the background image if there is one (more on that shortly).

You will now have the following code at the bottom of your style sheet:

```
@media screen and (max-width: 480px) {
   body {
   width: 100%;
   float: none;
   }
}
```

Width of content and sidebar

In portrait mode in particular, there isn't room for a sidebar to the right of the main content. Add the following code to the media query relating to devices with a maximum width of 320 pixels:

```
#content, #primary, #secondary {
width: 100%;
float: none;
margin: 10px 0;
}
```

Footer content, especially widgets

If your footer has widget areas or other elements with floats applied, you will need to override them for mobile devices in portrait mode.

If you want the footer widgets to be full width in both landscape and portrait modes, then simply add **`footer.widget-area`** to the CSS for the sidebars and content.

However, you might want the widget areas to be laid out side by side in landscape mode, depending on how many you have. In that case, you'll need to do the following:

1. Work out the percentages for the widths, padding and margins (some box-model maths for you!);
2. Add the relevant code to your media query for devices with a maximum width of 480 pixels;
3. Add a separate query for devices with a maximum width of 320 pixels after the one you're working on, with the following code:

```
footer .widget-area {
width: 100%;
float: none;
margin: 10px 0;
}
```

You might also need to adjust the text alignment and borders and padding, depending on your existing theme. Margins should be set to 0 on the left and right; suit them to your theme at the top and bottom, but generally they should be smaller than in the desktop version.

Image sizes

The images in your design might still break the layout or break out of their containing elements, making your website shrink when viewed on a mobile device. There is an easy fix for this:

```
body img {
max-width: 100%;
}
```

This will ensure that images are never wider than their containing element. You might need to tweak the CSS if images sized further up in the style sheet have greater specificity.

However, this solution isn't ideal. The images might look smaller, but mobile devices will still have to download their full sizes, which will slow down response times and possibly lose visitors, as well as annoy users on expensive data plans (more of them are out there than you might think). There a number of solutions to this, some of which you will find in this roundup of articles on responsive images. You may recall the mobile-first approach mentioned earlier; one benefit of this approach is that it serves different-sized image files to devices based on screen width.

Text size

So, our layout is working, and everything displays nicely. But now that the website is narrower, the text might appear huge. We'll need to adjust the text's size with the following code:

```
body {
font-size: 60%;
line-height: 1.4em;
}
```

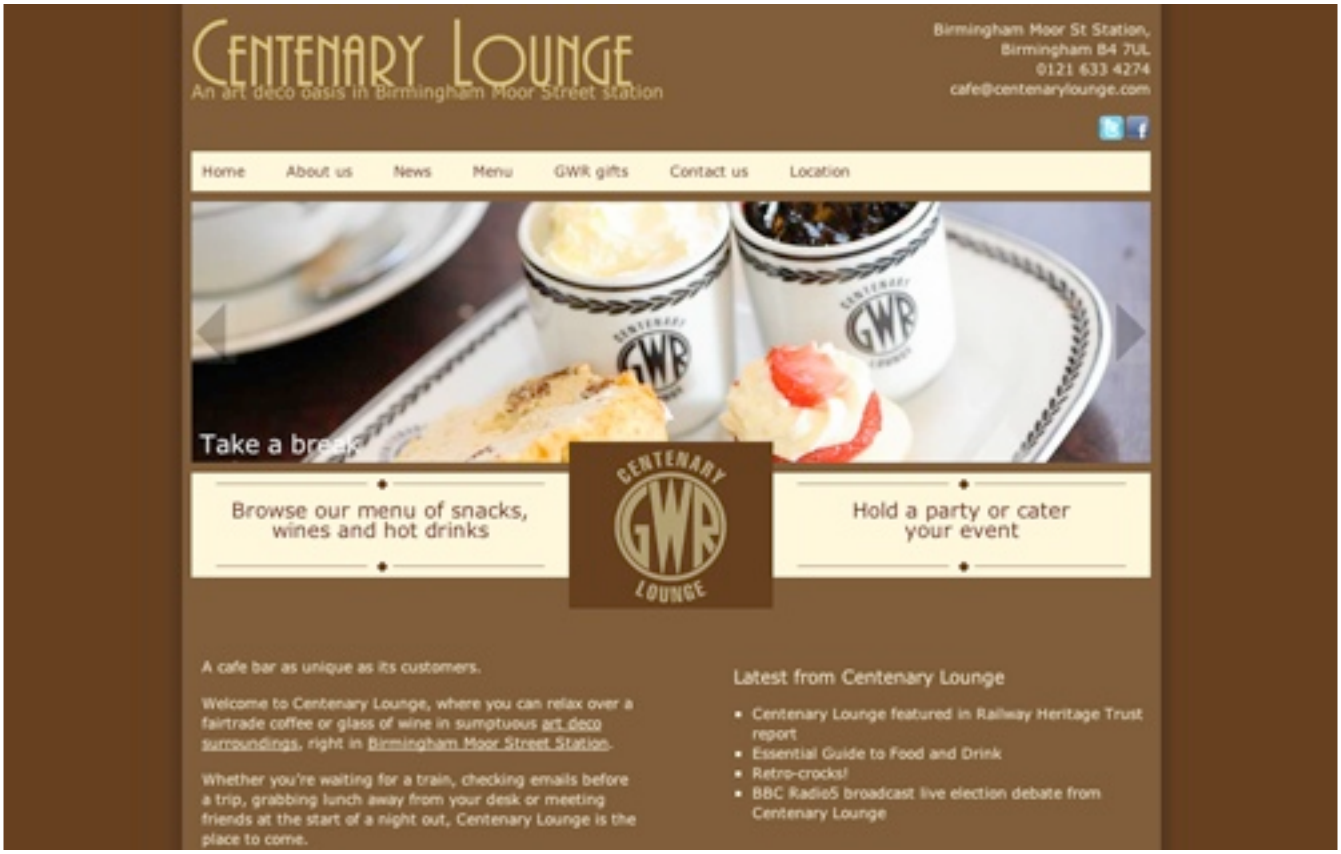
This sets the font size as a percentage of the size set for it further up in the style sheet.

4. CHANGING THE NAVIGATION MENUS AND CREATING AN APP-LIKE INTERFACE

Sometimes mobile users will want to access specific content; for example, visitors to a store's website will want to find the store's location easily, and visitors to an e-commerce website will want to shop with a minimum of clicks (or taps). Sometimes you might want to adjust the navigation to make the website look more like an app.

Here are some methods you can follow to do this:

- Use CSS to turn menu items that are visible on the desktop into drop-down menus, using code similar to what you would use to create a second-level drop-down menu on a desktop website.
- Use conditional PHP or a plugin such as Mobble to display a different menu depending on the device, as seen on the website that I developed for Centenary Lounge:



Centenary Lounge
An art deco oasis in Birmingham Moor Street station
Birmingham Moor St Station,
Birmingham B4 7UL
0121 633 4274
cafe@centenarylounge.com
Home
About us
News
Menu
GWR gifts
Contact us
Location
Take a break
Browse our menu of snacks, wines and hot drinks
Centenary GWR Lounge
Hold a party or cater your event

A cafe bar as unique as its customers.

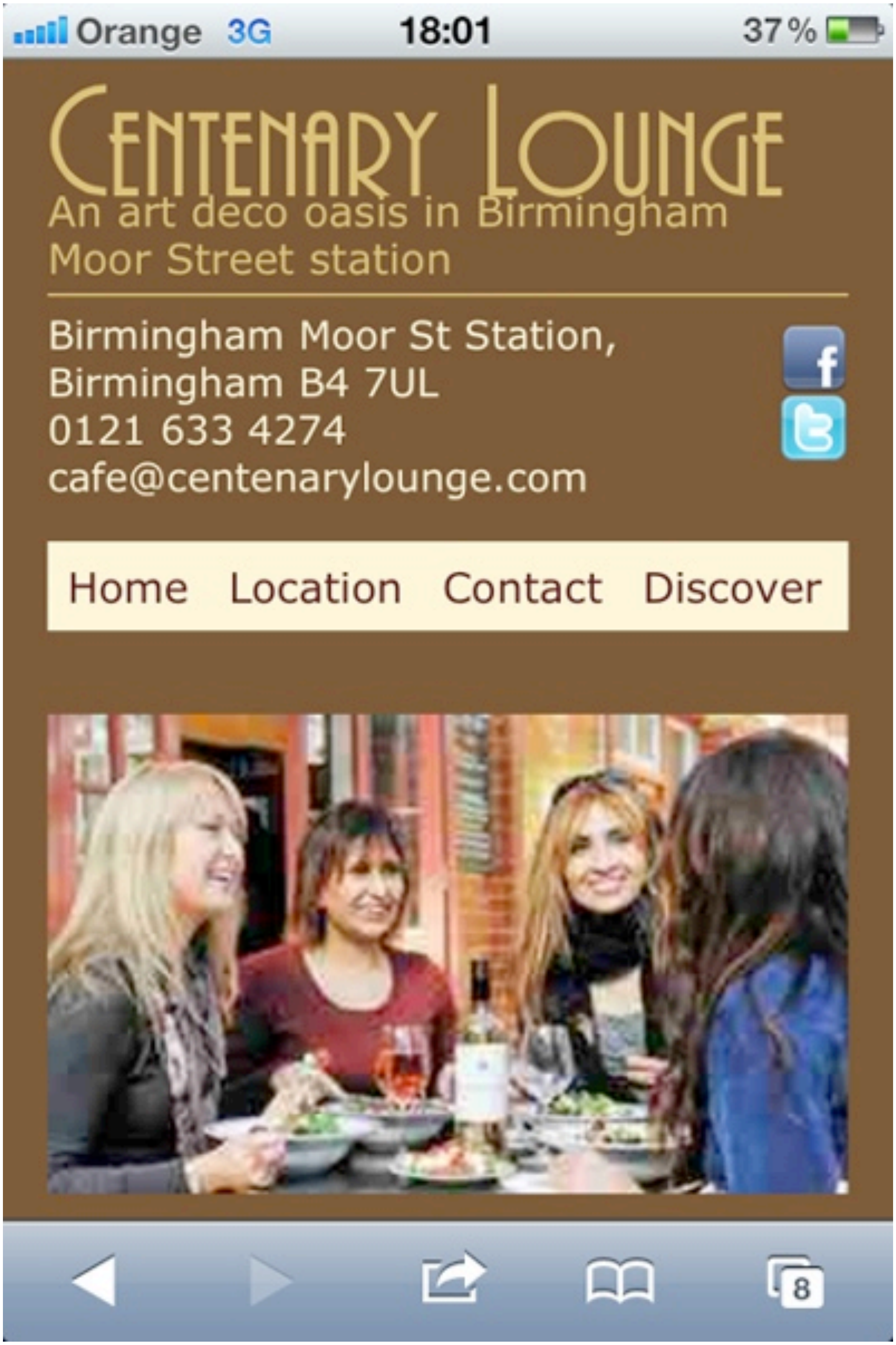
Welcome to Centenary Lounge, where you can relax over a fairtrade coffee or glass of wine in sumptuous art deco surroundings, right in Birmingham Moor Street Station.

Whether you're waiting for a train, checking emails before a trip, grabbing lunch away from your desk or meeting friends at the start of a night out, Centenary Lounge is the place to come.

Latest from Centenary Lounge

- Centenary Lounge featured in Railway Heritage Trust report
- Essential Guide to Food and Drink
- Retro-crocks!
- BBC Radio5 broadcast live election debate from Centenary Lounge

Centenary Lounge desktop website



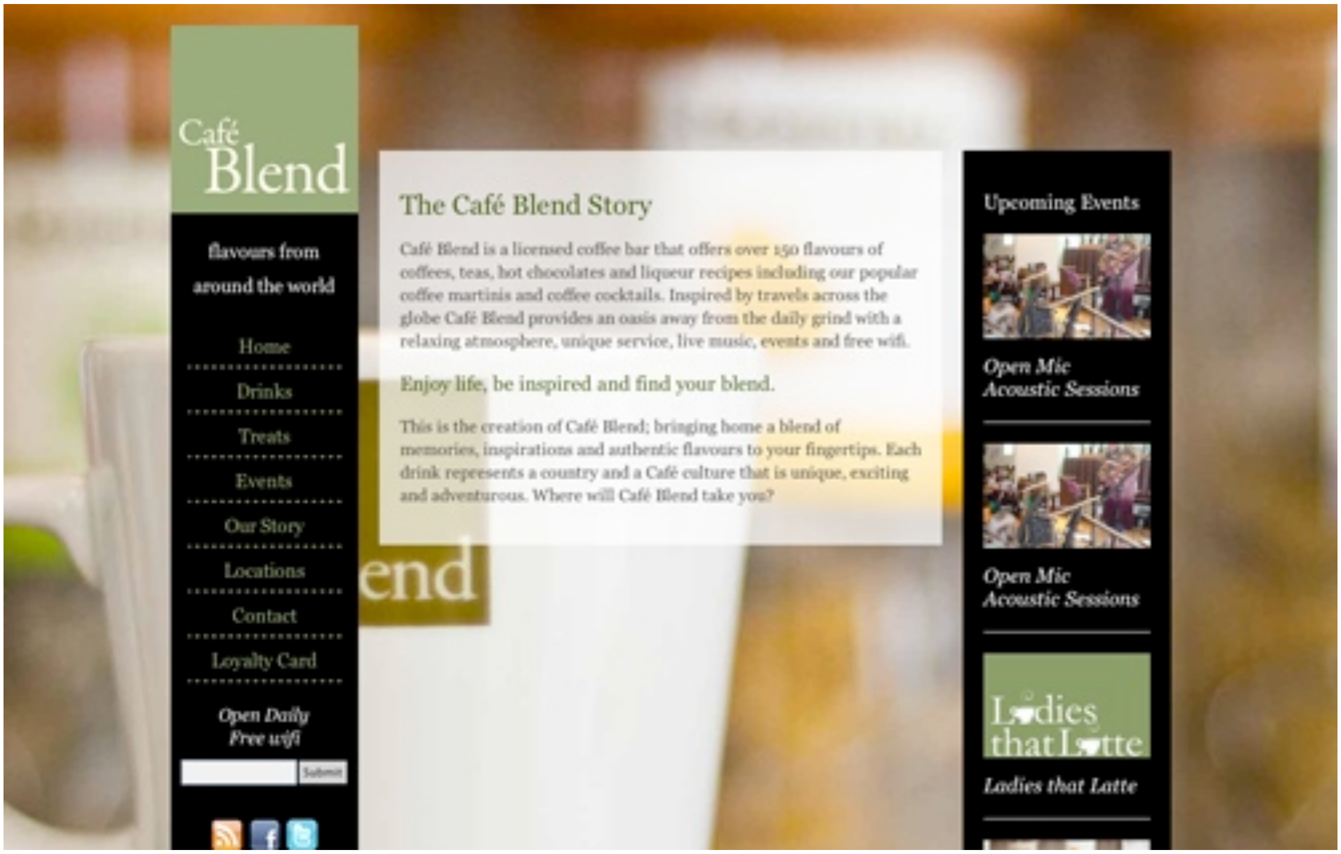
CENTENARY LOUNGE

An art deco oasis in Birmingham Moor Street station

Birmingham Moor St Station,
Birmingham B4 7UL
0121 633 4274
cafe@centenarylounge.com

Centenary Lounge mobile website

- Use CSS to display menu items as a vertical list of buttons to give the website an app-like look, such as on Cafe Blend:



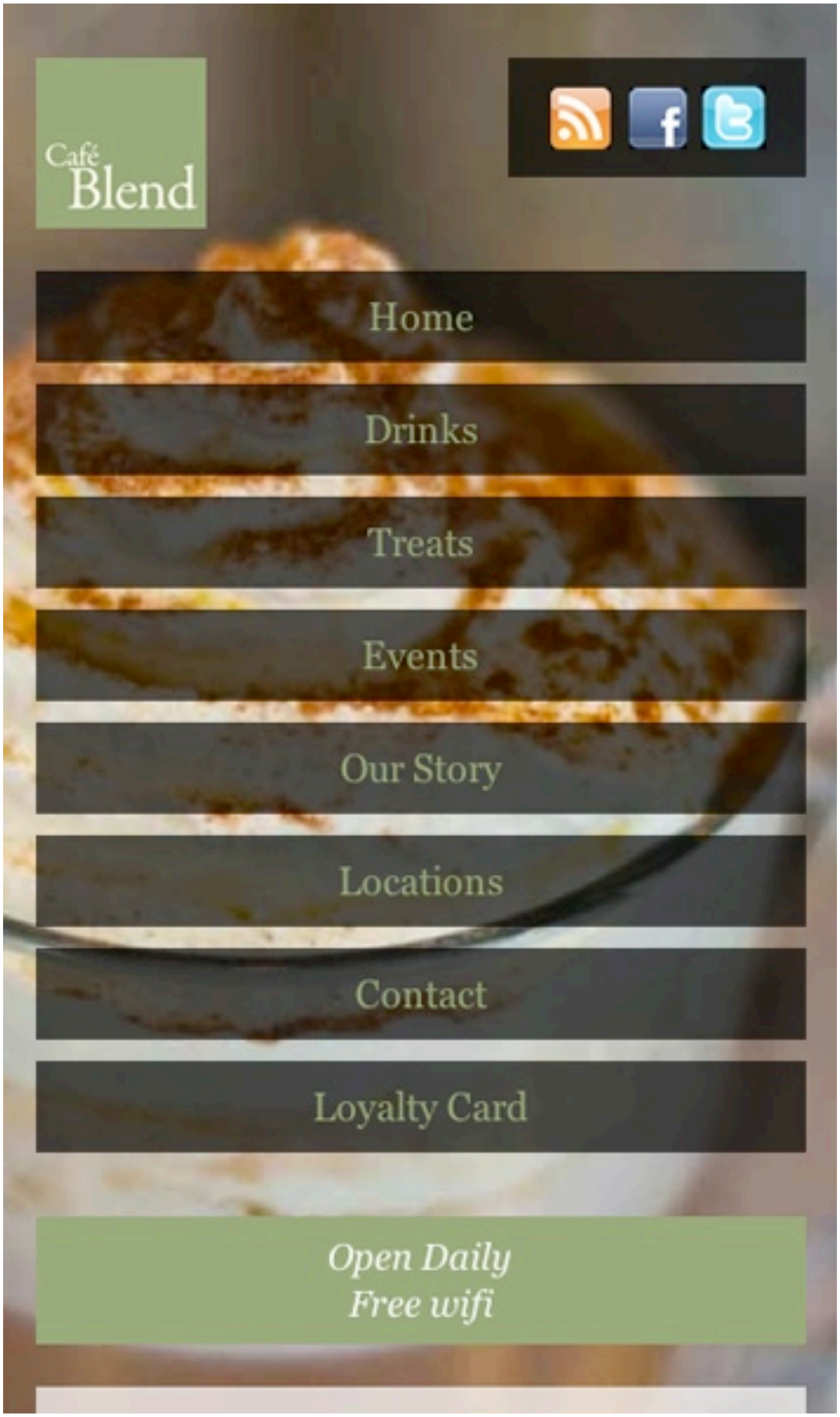
The Café Blend Story

Café Blend is a licensed coffee bar that offers over 150 flavours of coffees, teas, hot chocolates and liqueur recipes including our popular coffee martinis and coffee cocktails. Inspired by travels across the globe Café Blend provides an oasis away from the daily grind with a relaxing atmosphere, unique service, live music, events and free wifi.

Enjoy life, be inspired and find your blend.

This is the creation of Café Blend; bringing home a blend of memories, inspirations and authentic flavours to your fingertips. Each drink represents a country and a Café culture that is unique, exciting and adventurous. Where will Café Blend take you?

Cafe Blend desktop website



Cafe Blend mobile website

- Use a plugin such as Dropdown Menus to display menu items as a drop-down walker, freeing up screen real estate.
- Use background images combined with media queries and floats, to create a grid of visual buttons for your navigation, giving the home page an app-like feel.
- Use fixed positioning to fix the navigation to the bottom of the screen, minimizing the need for scrolling, as seen earlier on Social Media Examiner.

The possibilities are limited only by your imagination and creativity!

5. A PROBLEM!

You've added the media queries above, but your smartphone still displays the desktop version. Don't worry! This is because many smartphones use a virtual viewport that is equal to the width of a small desktop, which prevents desktop-designed websites from breaking when rendered in the browser. This can be easily fixed by placing the following code in the **`head`** of each page. Because yours is a WordPress website, you need to add it only once, to the **`header.php`** theme file:

```
<meta name="viewport" content="width=device-width">
```

What this does is tell the phone to treat the size of the screen as its actual size, not the virtual size… if that makes sense.

Summary

Here's what we've looked at in this article:

- Four different ways to make a WordPress website mobile-friendly: with a plugin, with a prebuilt responsive theme, with a separate mobile theme, and by making the existing theme responsive;
- Media queries for responsive design and how they target different device widths;
- Some common styles to make a WordPress website responsive in its layout, images and text.

As you can see, no one option is necessarily the best; it will depend on the website, on the budget and on the time and capability of those involved. Over time, most mobile-friendly WordPress websites will have responsiveness built into them, instead of using a separate theme, mobile website or plugin.

Hopefully this article has given you a starting point to make your WordPress website mobile-friendly. This is just the beginning of the possibilities. To further develop your mobile website, you might want to consider a mobile content strategy; a mobile-first design; APIs and native device functionality to create an even more app-like experience; and more.

Enjoy!

About The Authors

Alexander Dawson

Alexander is a freelance web designer, author and recreational software developer specializing in web standards, accessibility and UX design. As well as running a business (HiTechy) and writing, he spends time in Twitter, SitePoint's forums and other places, helping those in need.
Website: HiTechy
Twitter: Follow the author on Twitter

Bruce Lawson

Bruce Lawson evangelises open web technologies for Opera. He co-authored Introducing HTML5, the best-selling book on HTML5 that has just been published in its second edition. He blogs at brucelawson.co.uk.
Website: http://www.introducinghtml5.com
Twitter: Follow the author on Twitter

Gavyn McKenzie

Gavyn McKenzie likes downhill skateboarding, pondering semantics, painting with CSS and fine gastronomy. He is currently employed as a front-end web developer at Holiday Extras and dabbles in freelance work for interesting clients.
Website: http://www.gavyn-mckenzie.co.uk
Twitter: Follow the author on Twitter

Jon Raasch

Jon Raasch is a front-end web developer / UI designer with endless love for jQuery, CSS3 and performance tuning. Follow him on Twitter or read his blog.
Website: Jon Raasch Web Design & Dev
Twitter: Follow the author on Twitter

Kim Pimmel

Kim is user experience designer who creates engaging and innovative experiences on Adobe's XDCE team. When not pushing pixels for Adobe, he shoots photos, plays with physical computing, and makes DJ mixes.
Website: http://www.kimpimmel.com
Twitter: Follow the author on Twitter

Michael Flarup

Michael Flarup is a Copenhagen based interface designer & iconist. When he’s not freelancing and blogging out of PixelResort.com he’s creating iPhone apps with his young upstart company Robocat.
Website: Pixelresort
Twitter: Follow the author on Twitter

Rachel McCollin

Rachel manages Compass Design, a Birmingham-based agency providing web design and development and social media training and support. Her aim for 2012 is for every new site she develops to be fully responsive.
Website: Compass Design
Twitter: Follow the author on Twitter

Ros Hodgekiss

Ros Hodgekiss is the Community Manager at Campaign Monitor, the popular email marketing web app for designers and their clients. When not uncovering new email hacks for the blog or helping out customers, she's either shooting arrows, paddling madly, or making a nuisance of herself on Twitter.
Website: http://campaignmonitor.com
Twitter: Follow the author on Twitter

Smashing Editorial

Sven (sl), Vitaly (vf), Iris (il), Jan (jc) and Esther (ea) love high-quality content and care about little details. They also believe that good content and design are crafts worth sharpening.

Tim R. Todish

Tim Todish is a UX technologist with a passion for mobile & portable devices. He has been in the industry for over 10 years creating engaging experiences for clients both large and small across all industries.