

DEVICE SIZE MATTERS

Responsive Web Design With Physical Units

By Radu Chelariu^{1 1}

Published on March 21st, 2013 in Responsive Design^{2 2} with 44 Comments

This post should be titled “Getting Ahead of Yourself.” “...By a Few Years,” actually. Here’s the deal: at the time I’m writing this, early 2013, there’s no way to accurately design for the Web using physical units, nor will there be for a very long time. But there is a way to design while knowing the physical characteristics of the device — or, at least, there will be in the very near future.



Different devices can have a similar screen resolution, yet entirely different physical factors. iPad (1st generation) has the diagonal size of 9.7", the resolution 1024 × 768 and 132 ppi. Kindle Keyboard 3G has the diagonal size of 6", also the resolution 768 × 1024, yet 212 ppi. Image source: [kodomut](#).

It’s called the “resolution media query”, and it’s been in the specification for media queries for some time. However, while it has been in the spec, that doesn’t mean anyone has actually implemented it yet. Fortunately, [WebKit is leading the way](#) and pushing for this feature to be implemented. So, how will we use this nifty little feature, exactly? Here’s how.

The Thin Line Between Queries

First off, I posit that there will be only one use case for a resolution-only media query. Something along the lines of

```
@media (min-resolution: 250dpi) {  
  
}
```

has, at this time, only one good use: swapping out low- for high-resolution images. I've tried imagining other uses and, as far as I can tell, there just aren't any. But resolution is not what we, as Web designers, are truly interested in. Since we are designing for humans, shouldn't we be thinking about the physical side of human data consumption and designing using this kind of a metric? And in a perfect world we could simply say `width: 1in` and have a one-inch wide element, regardless of the device. Unfortunately, we live in a digital world in which the physical and digital pixels are not the same. We need something to bridge the gap. That something is the resolution media query.

Good. Now that that's out of the way, let me show you how this one little piece of code can make so much difference that your head will promptly explode. (I take no responsibility for actual blown heads as a result of this post.)

Let's compare two media-query declarations:

```
@media (min-resolution: 341dpi) and (min-width: 767px) >;; {  
  
}
```

and

```
@media (max-resolution: 131dpi) and (min-width: 767px) >;; {  
  
}
```

At first glance, this doesn't seem like much of a separation, right? Wrong. The numbers I've used are specific to the HTC Windows Phone 8X (the first snippet) and the iPad 2 (the second snippet). By using the resolution query, one can basically separate physically small devices from large devices.

As it currently stands, a query that looks like `@media (min-width: 767px){ }` will affect both the HTC and the iPad, with no other possibility of separation, because both have a resolution that is 768 pixels wide. In fact, the iPad has a lower resolution, at 1024×768 , whereas the HTC is 1280×768 . In case you haven't realized yet, the problem with all of this is that the iPad is a 10-inch device, while the HTC is a 4.3-inch one. That's less than half the physical size!

By using the resolution media query together with a width query, we can distinguish between physically small and large devices to adjust design elements and layouts accordingly. While, as mentioned above, screen resolution isn't *really* what we are interested in since we use logical breakpoints in responsive design, it *is* useful to know whether a site is being displayed on a small or a large physical display — e.g. to increase font size or rearrange design elements in the layout. But where do we draw the line between small and large? Quite simply, we can't. Each of us has to draw the line, possibly on a project-by-project basis, between “This is a small device” and “This is a large device.” As far as ballpark numbers go, I've done a few calculations and have developed a theorem that should give you a clearer idea of how this works more or less.

The Physical Size Inquiry Non-Exhaustive Theorem (PSINET)

Here's the theory: In a combined query, if the ratio between the smaller of the width and height and the resolution, called a **PSINET** score, is higher than **5**, then the result falls in the category of a physically large device. If the resulting number is lower than 5, then it is a physically small device. Devices that score very close to 5 are considered to be medium-sized, close to the physical size of an A4 sheet of paper (21 × 29.7 cm).

Here's a non-exhaustive list of devices to test the formula above. I've listed each device's score according to the formula, along with its diagonal size, resolution and density, and PSINET score.

PHYSICALLY LARGE DEVICES

Device name	Diagonal size (inches)	Resolution	PPI	PSINET score
Apple iMac	27	2560 × 1440	109	13.00
Sony Vaio F	16.4	1920 × 1080	134	8.05
Apple MacBook Pro RD	13	2560 × 1600	227	7.04

PHYSICALLY SMALL DEVICES

Device name	Diagonal size (inches)	Resolution	PPI	PSINET score
Sony PSP	4.3	480 × 272	128	3.75
Kindle Keyboard 3G	6	768 × 1024	212	3.62

Kindle Fire	7	1024 × 600	169	3.55
Samsung Galaxy S	4	480 × 800	160	3.00
Samsung Galaxy NoteII	5.5	720 × 1280	267	2.69
Samsung Galaxy S IV	5	1080 × 1920	441	2.62
HTC Butterfly	5	1080 × 1920	441	2.62
Samsung Galaxy Grand I9082	5	480 × 800	187	2.56
Palm Pre	3.1	480 × 320	186	2.5
Sony Xperia Z	5	1920 × 1080	443	2.43
Samsung Galaxy SIII	4.8	720 × 1280	306	2.35
LG Nexus 4 E960	4.7	768 × 1280	318	2.41
Nokia Lumia 920	4.5	1280 × 768	332	2.31
HTC One	4.7	1080 × 1920	469	2.30
HTC One X	4.7	720 × 1280	312	2.30
HTC Desire HD	4.3	480 × 800	217	2.21
BlackBerry Q10	3.1	720 × 720	328	2.19
BlackBerry Z10	4.2	768 × 1280	355	2.16
Motorola Droid X	4.3	854 × 480	228	2.10
Sony Ericsson S	4.3	720 × 1280	342	2.10
Motorola RAZR i XT890	4.3	540 × 960	256	2.10
iPhone 5	4	640 × 1136	326	1.96
Apple iPod Touch	3.5	960 × 640	326	1.96
Nokia Lumia 620	3.8	480 × 800	246	1.95
HTC Wildfire	3.2	240 × 320	125	1.92
Nokia Lumia 710	3.7	800 × 480	252	1.90
Motorola Defy	3.7	854 × 480	265	1.81

LG Optimus One	3.2	320 × 480	180	1.77
Nokia N96	2.8	240 × 320	143	1.67
Sony Ericsson W810i	1.9	176 × 220	148	1.18

MEDIUM-SIZED DEVICES

Device name	Diagonal size (inches)	Resolution	PPI	PSINET score
Apple iPad (1 & 2)	9.7	1024 × 768	132	5.81
Apple iPad (3rd Gen)	9.7	2048 × 1536	264	5.81
Amazon Kindle DX	9.7	824 × 1200	150	5.49
Acer Iconia Tab A500	10.1	800 × 1280	149	5.36
Samsung Galaxy Tab	10.1	1280 × 800	149	5.36
Motorola Xoom	10.1	1280 × 800	149	5.36
Asus Transformer Pad Infinity	10.1	1920 × 1200	224	5.35
Microsoft Surface	10.1	1366 × 768	148	5.18
Asus VivoTab RT TF600T	10.1	1366 × 768	155	4.95
iPad Mini	7.9	768 × 1024	162	4.74
Amazon Kindle Fire HD	8.9	1920 × 1200	254	4.72

Is this method of determining device size foolproof? Hardly — that’s why it’s a theorem. It’s based on solid reasoning and empirical evidence and has come about by using the scientific method, but it is not a rule, law or axiom. Take it with a pinch of salt (or, better yet, a truckload of NaCl) and refine it. It is a theorem, a proposition, to be remembered in the future when the resolution media query and our work with it become a mainstay of the Web.

BREAKING THE THEOREM

Like any self-respecting follower of the scientific method, I’ve tried to break my own theorem. Thus, I imagined a freak of a device, 2 inches long and 20 inches wide, putting its diagonal size at 20.09 inches, with a 240 × 40 pixel display, yielding a resolution of just 11.94 PPI. It gets a PSINET score of 2.01, which puts it well into the

small device category, even though it’s almost half a meter long. The reason is simple: it’s that 2-inch-wide dimension. Because the PSINET score ignores the longer of the device’s physical width and height, the greater the difference between those two dimensions, the less accurate the PSINET score will be. Sure, this beast of a device is unlikely to ever become reality, but it’s worth understanding the reasons why it would break the theorem.

Device name	Diagonal size (inches)	Resolution	PPI	PSINET score
ThinLong	20.09	24 × 240	11.94	2.01

Real-World Applications

Apart from the obvious visual changes and tweaks mentioned above, there are other ways to use the resolution media query.

Enter [Enquire.js](#). For those of you who haven’t heard of it, it’s a very nice JavaScript library that helps you execute particular scripts on matching media queries.

We could use Enquire.js or even just [window.matchMedia](#), which is a native JavaScript method, to differentiate between mobile, tablet and computer users much more reliably than by using width queries alone. Here’s a not-very-polite example using Enquire.js:

```
enquire.register("screen and max-resolution: 150dpi and max-width: 300px", function()
{
    alert('My, what a small screen you have there, Grandma!')
});
```

Combining media query types with CSS and using a resolution-aware JavaScript library is just the right formula to give us real future-proof control over what I call the “physical Web.” Imagine being able to view a priceless sculpture located in a museum halfway across the Earth on a 1:1 ratio on any device, or shopping for an engagement ring online and seeing exactly how big that 24-carat diamond is. The real-world applications, pun intended, are nearly endless.

In our world of responsive Web design, we’d very much like to provide the best experience to users, whatever their device. In light of the sans-resolution media query above, that task becomes less of a challenge and more a windmill fight. Assigning blame is pointless, because none of us can do anything to change the current ecosystem of devices. Manufacturers will continue to put out devices with resolutions and pixel densities that they’ve pulled out of their butts, and that’s fine — that’s their business. Staying on top of the situation by providing us designers with the tools we need (but can’t easily build ourselves) to create the best user experience possible is the job of browser makers, and I salute the good people at WebKit for spearheading the implementation of the resolution media query.

FOOTNOTES:

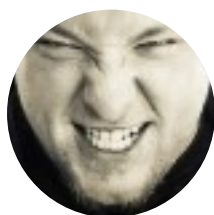
⁰ Radu Chelariu - <http://mobile.smashingmagazine.com/author/radu-chelariu/?rel=author>

¹ Responsive Design - <http://mobile.smashingmagazine.com/tag/responsive-design/>

FOOTNOTES:

⁰ Radu Chelariu - <http://mobile.smashingmagazine.com/author/radu-chelariu/?rel=author>

¹ Responsive Design - <http://mobile.smashingmagazine.com/tag/responsive-design/>



Art director, focused on eCommerce design and development and a fan of flat skeumorphism. Sometimes blogger, novice speaker, creator of the HTML5 Starter Pack, Adobe Certified Expert in Photoshop and member of the now defunct Smashing Network.

With a commitment to quality content for the design community.

Made in Germany. 2006-2013. <http://www.smashingmagazine.com>