# Introduction to media queries – Part 1: What are media queries?

The rapid spread of mobile devices has turned the world of web design upside down. Users no longer view web content only on traditional desktop systems, but are increasingly using smartphones, tablets, and other devices with a wide range of dimensions. The challenge for web designers is to ensure that their websites look good not only on a big screen, but also on a tiny phone and everything in between.

Media queries are an excellent way to deliver different styles to different devices, providing the best experience for each type of user. A part of the CSS3 specification, media queries expand the role of the media attribute that controls how your styles are applied. For example, it's been common practice for years to use a separate style sheet for printing web pages by specifying media="print" . Media queries take this idea to the next level by allowing designers to target styles based on a number of device properties, such as screen width, orientation, and so on. Figures 1–3 demonstrate media queries in action. They all show the same web page as viewed in a desktop browser, on a tablet, and on an iPod touch.



Figure 1. When viewed on a desktop, the page has a two-column layout.

In the desktop version, the page has a fixed-width, two-column layout. But when the same page is viewed on a tablet, the sidebar is moved below the main content.

<link href="css/phone.css" rel="stylesheet" type="text/css" media="only screen and (min-width: 0px) and (max-width: 649px)" >

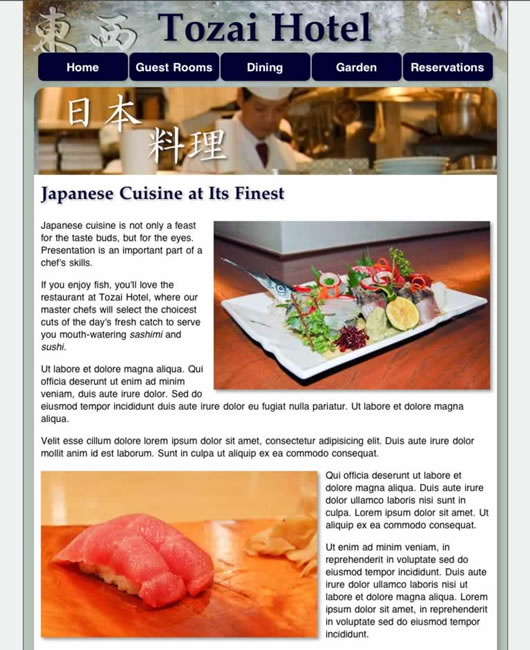


Figure 2. The layout switches to a single column to fit the narrower screen of a tablet

When viewed in an iPod touch, the menu is rearranged and the images are scaled down. The different styles are delivered to each device using media queries.



Figure 3. Media queries restyle the page to fit the constraints of a much smaller screen.

This article provides an overview of media queries, including examples that will help jumpstart your own multiscreen website development.

**Note:** Be sure to also check out the following videos on Adobe TV: CSS3 media queries with Dreamweaver and How media queries can make designing for mobile devices and different screens easier.

**Media query example**

A simple example of a media query might look like this:

<link href="css/phone.css" rel="stylesheet" type="text/css"

media="only screen and (max-width: 400px)" >

In this example, the media query has been added to the <link> tag. As you will see later, you can also use media queries within style sheets. The media attribute is where the query actually resides. This one says:

only screen and (max-width: 400px)

The meaning should be fairly obvious: apply this style sheet only to a device that has a screen and only if the browser window is no wider than 400 pixels. You can see from the file name phone.css that this particular query is intended for phone styles. Now that you have seen an example, let's take a closer look at media queries and the device features that you can use to control where styles are applied.

**Media query support and features**

Media queries are supported in Internet Explorer (IE) 9+, Firefox 3.5+, Safari 3+, Opera 7+, as well as on most modern smartphones and other screen-based devices. Although older versions of IE don't support media queries, you can—and should—start using them now. Strategies for dealing with older browsers are discussed later in this article. Table 1 lists the device features that can be used in media queries:

**Table 1.** Media features for setting conditions in media queries

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Value | Min/Max | Description |
| width | Length | Yes | Width of display area |
| height | Length | Yes | Height of display area |
| device-width | Length | Yes | Width of device |
| device-height | Length | Yes | Height of device |
| orientation | portrait orlandscape | No | Orientation of device |
| aspect-ratio | Ratio (w/h) | Yes | Ratio of width to height, expressed as two integers separated by a slash (e.g., 16/9) |
| device-aspect-ratio | Ratio (w/h) | Yes | Ratio of device-width to device-height |
| color | Integer | Yes | Number of bits per color component (if not color, the value is 0) |
| color-index | Integer | Yes | Number of entries in the output device's color lookup table |
| monochrome | Integer | Yes | Number of bits per pixel in the monochrome frame buffer (if not monochrome, the value is 0) |
| resolution | Resolution | Yes | Density of pixels of output device, express as integer followed by dpi (dots per inch) or dpcm (dots per centimeter) |
| scan | progressiveor interlace | No | Scanning process used by TV devices |
| grid | 0 or 1 | No | If set to 1, the device is grid-based, such as a teletype terminal or phone display with only one fixed font (all other devices are 0) |

The first five features in Table 1 ( width , height , device-width , device-height , and orientation ) are the most useful. You can prefix most features with min- and max- to indicate minimum and maximum values, such as min-width and max-width . The Min/Max column in Table 1 indicates which features can be modified in this way.

**width, device-width, and viewport**

Perhaps one of the most confusing aspects of media queries is the difference between width and height and the equivalent values prefixed by device- . In the case of desktop and laptop computers, the difference is easy to understand: width and height refer to the size of the browser viewport, whereas device-width and device-height refer to the dimensions of the monitor. Not everybody runs his or her browser full-screen, so width andheight are the measurements that you need to use.

Mobile browsers fill the available screen, so you might expect width and device-width to be the same. Unfortunately, that's not always the case. Most smartphones, including Android, iPhone, and Windows Phone 7, setwidth to a nominal viewport approximately 1,000 pixels wide (in an iPhone and iPod touch, it's 980 pixels, Windows Phone 7 uses 1024 pixels). Figure 4 shows how an iPod touch normally displays the example page in the previous figures.

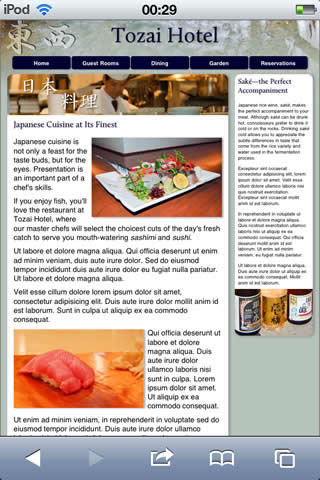


Figure 4. By default, modern mobile devices scale web pages to fit a notional viewport.

Even though the style sheet attached to the page uses media queries to serve different styles depending on the values ofmin-width and max-width , the iPod touch ignores the styles and displays the desktop version because the viewport is considered to be 980 pixels wide.

To confuse matters even further, the iPhone, iPod touch, and iPad don't take orientation into account when calculating width, whereas other devices do.

Thankfully, there's a simple solution to this confusion. Apple devised a new <meta> tag, which has been widely adopted by other mobile device manufacturers and has been incorporated into specification likely to be approved by the World Wide Web Consortium (W3C). To ensure a level playing field with all devices that support media queries, just add the following line to the <head> of each web page:

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

This tells mobile devices to treat the viewport as being the same width as the physical width of the device. What's more, it tells the iPhone, iPod touch, and iPad to take orientation into account when calculating width. As a result, you can usewidth in media queries safe in the knowledge that it means what you think it does.

**Note:** Many mobile devices, notably the iPhone 4 and iPad 2, have high resolution displays with a much higher pixel density than desktop and laptop monitors. This doesn't affect the way that you calculate pixel sizes in CSS. The CSS specification says that if the pixel density of the output device is very different from that of a typical computer display, the browser should rescale pixel values. Some developers now refer to pixel measurements as *CSS pixels*.

**How to write media queries**

To add a media query to the media attribute, you set one or more conditions using the media features in Table 1. You specify the value for a media feature after a colon in the same way as for a CSS property. Each condition is wrapped in parentheses and added to the media declaration using the keyword and . For example:

media="screen and (min-width: 401px) and (max-width: 600px)"

Media queries are Boolean: they will either be true or false. If the entire statement is true, the style sheet is applied. If false, it will be ignored. So when using the above query, all parts must be true for the style sheet to be applied. In other words, it will apply only to screens between 401 and 600 pixels wide.

Some media features, such as color , monochrome , and grid , can be used as conditions without specifying a value. For example, the following targets all color visual displays:

media="screen and (color)"

**Specifying alternatives**

There is no or keyword to specify alternative media features. Instead, you list alternatives as a comma-separated list like this:

media="screen and (min-width: 769px), print and (min-width: 6in)"

This applies styles to screens wider than 769 pixels or print devices using paper at least 6 inches wide.

**Specifying negative conditions**

To specify a negative condition, you can precede the media declaration with the keyword not like this:

media="not screen and (monochrome)"

You can't use not in front of an individual condition. The keyword *must* come at the beginning of the declaration, and it negates the whole declaration. So, the preceding example applies to all devices except monochrome screens.

**Hiding media queries from older browsers**

The media queries specification also provides the keyword only , which is intended to hide media queries from older browsers. Like not , the keyword must come at the beginning of the declaration. For example:

media="only screen and (min-width: 401px) and (max-width: 600px)"

Browsers that don't recognize media queries expect a comma-separated list of media types, and the specification says they should truncate each value immediately before the first nonalphanumeric character that isn't a hyphen. So, an old browser should interpret the preceding example as this:

media="only"

Because there is no such media type as only , the style sheet is ignored. Similarly, an old browser should interpret the following as media="screen" :

media="screen and (min-width: 401px) and (max-width: 600px)"

In other words, it should apply the style rules to *all* screen devices, even though it doesn't know what the media queries mean. Unfortunately, IE 6–8 failed to implement the specification correctly. Instead of applying the styles to all screen devices, it ignores the style sheet altogether.

In spite of this behavior, it's still recommended to prefix media queries with only if you want to hide the styles from other, less common browsers.

**Dealing with older versions of Internet Explorer**

The lack of support for media queries in IE 6 through IE 8 is not a problem. Simply create a basic set of styles that are served to all browsers without using media queries, and use the media queries to provide an enhanced experience for visitors using more advanced browsers.

Alternatively, use an Internet Explorer conditional comment to serve a special set of rules to older versions of IE like this:

<!--[if lt IE 9 & !IEMobile]>

<link href="iestyles.css" rel="stylesheet" type="text/css">

<![endif]-->

**Using media queries with @import and @media**

In addition to using media queries in <link> tags when attaching an external style sheet, you can use them with@import and @media . The basic syntax is the same. For example, the following imports a style sheet and applies the styles to devices with a screen that's no wider than 400 pixels:

@import url("phone.css") only screen and (max-width:400px);

Media queries can also be used within a style sheet like this:

@media only screen and (max-width:400px) {

#navbar {

float: none;

width: 400px;

}

}

**Testing media queries**

Testing your code is important. Testing for tablets and smartphones complicates testing since ideally you would have an array of devices at your disposal. Luckily, you can test most queries without possessing the actual device. It's always better to have the real thing in front of you, but for the purposes of this article, using a simple browser will help you understand how queries work.

The ZIP file that accompanies this article contains a sample file (mediaqueries.html) with three different designs attached. Use the sample file to go through the following resize test. With the browser window fully open, you can see the basic site design (see Figure 1). Make the browser window narrower and notice the changes as you pass the dimensions for various devices. As you pass the dimensions specified in the code, the style should change to the tablet (Figure 2) or phone size (Figure 3). The header image becomes smaller for tablets, for example, and the pods below take on a different flow. For phone style, the main center image disappears, replaced by a menu of larger vertical buttons.

The browser checks media queries every time the window changes, when you resize it, for instance, or when you flip the phone from portrait to landscape orientation.

**Media queries in Dreamweaver CS5.5**

Dreamweaver CS5.5 improves upon the media query support that Adobe introduced with the 11.0.3 update for Dreamweaver CS5. The new Media Queries dialog box helps you create and maintain media queries for a single page or for an entire site (see Figure 5).

You can access the Media Queries dialog box from any of the following places:

* Modify > Media Queries
* Multiscreen Preview panel > Media Queries button
* Multiscreen dropdown menu > Media Queries
* Options menu in the upper-right corner of the CSS Styles panel
* Context menu in the CSS Styles panel

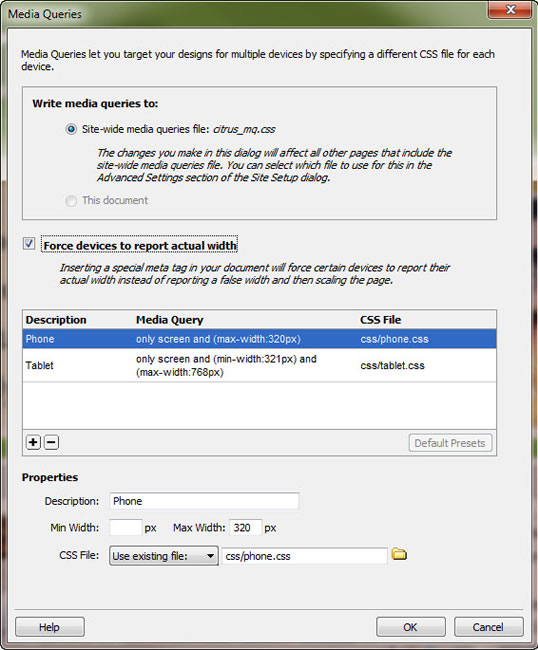


Figure 5. The Media Queries dialog box in Dreamweaver CS5.5

Using this new dialog box, you can:

* Create multiple media queries.
* Attach media queries to the current page, which adds links to a CSS file or files.
* Create a site-wide media queries file, which adds links to a single, centralized CSS file that imports other device-specific CSS files

When using the site-wide option, you can link to existing CSS files, or create new ones from the dialog box itself.

The Media Queries dialog box will also create a comment above the query with whatever text you add as the description.

Additionally, you can choose to add the viewport <meta> tag to the document that will force the device to report its actual dimensions, rather than the size of the nominal viewport. Selecting this option helps prevent unwanted scaling due to misreported sizes.

**Note:** For now, the Media Queries dialog box only reads and writes queries with at least one min-width and max-width value. Additionally, the dialog box edits these only min- and max- values. You can manually edit other query parameters; but min-width and max-width are the most commonly used values for targeting devices.

**Presets and site definition**

The Media Queries dialog box also provides a preset option at the bottom of the media query list area. Click this Default Presets button and Dreamweaver automatically creates:

* Three media queries for Tablet, Phone and Desktop
* min-width and max-width pre-populated with common starting values.

The Default Presets button provides a quick way to get started with media queries.

Another convenient way to manage the site-wide media query file is via the Site Setup dialog box. There is now a field called Site-wide Media Query File, in Advanced Settings > Local Info which specifies your site-wide media query file. The file listed here appears as the site-wide option whenever you open the Media Queries dialog box. For new (or even existing) files, all you need to do is select that option to apply the site-wide media query file to the current page.

**Note:** Changing this file will not automatically update all the site files.

The Media Queries dialog box offers a great way of simplifying the creation and management of your media queries.

Media queries are already the preferred way of targeting devices with different styles. As shown above, they are not only for smartphones, but can be used to control the look and feel of any screen.