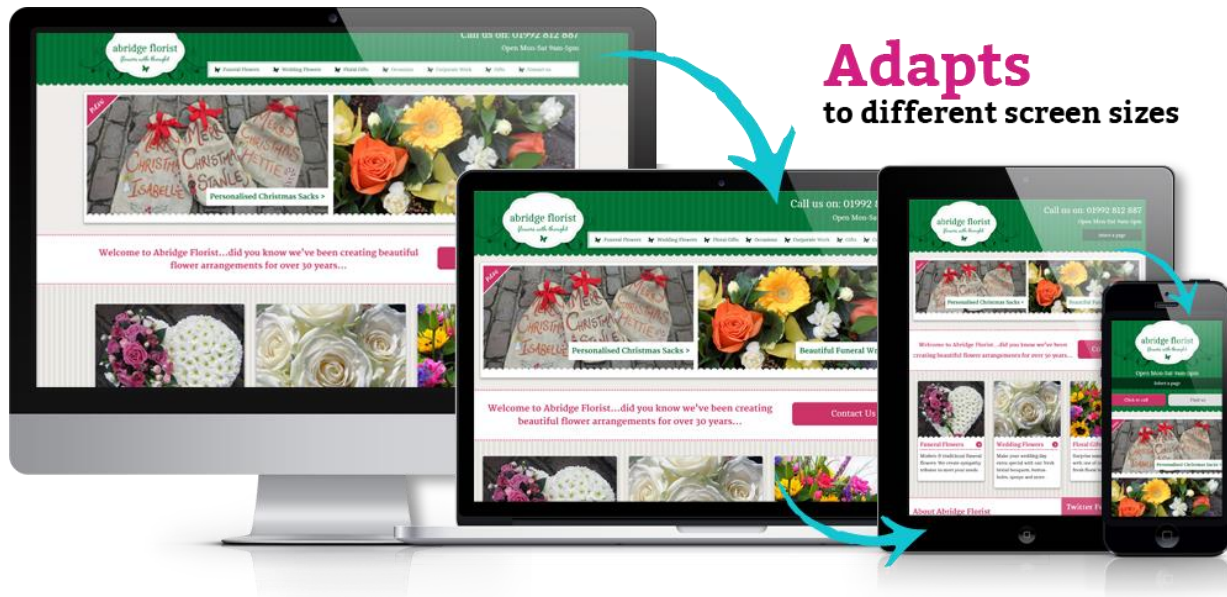


Responsive design gives web-based graphical user interfaces the flexibility to display content with fluidity between devices otherwise constrained by varying resolutions. Ultimately, responsive helps with ease of content readability and content navigation. Resolution is defined as the number of distinct pixels that may be displayed, and is displayed as height x width (e.g., 1024 x 768, which indicates 1024 pixels in horizontal width by 768 pixels in vertical height for total display size).

An example of resolution constraints is as follows:



Desktop resolution is typically 1024x768

iPad resolution is 1024x768

iPad mini resolution is 768x1024

iPhone 6 resolution is 1334x750

iPhone 4/4s resolution is 640x960

Not all sites are developed with responsive design tactics applied. However, there is an easy way to tell if a site is device-agnostic. Open the website in your browser window, then drag the size of your browser window to a much smaller horizontal width. If responsive design tactics are applied, then you should see content automatically resize to fit the new height and width. Further, depending on the techniques applied, you may see the header, logo, footer, and navigation switch out at varying resolutions to accommodate finger swiping versus mouse cursor engagement.

In this example, notice how the logo resizes, and navigation adjusts to accommodate the various width constraints:



Another way to identify websites that apply responsive design tactics is to view the source code of the page. There are a number of scripted methods to create fluid content. For instance, you might create an inline style with a float and max-width parameter, which enables text to resize dynamically. Many scripted tactics require meta viewport as follows:

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

The viewport relates to the relative size of the browser or device through the use of length units such as viewport width (vw), viewport height (vh), minimum of viewport height and width (vmin), and maximum of viewports height and width (vmax).

Because resolutions must be defined in relativistic sizes, another way to identify responsive design in an implementation is to look for @media tags with a minimum width of 120 pixels and a maximum width of 420 pixels, or variations of these minimum- and maximum-width dimensions. Typically, designs allow for one type of header at higher widths and another type with larger icons for finger swiping at lower widths. This requires that certain styles, images, and layouts only be triggered within a specified relative range. As a result, you may see one style sheet invoked at a max-width of 420 pixels (with included minimum boundary defined) but an entirely different style sheet invoked up to a max-width of 768 pixels (with minimum boundary defined). Here is what a full responsive design implementation looks like with relativistic resolution ranges:

```
<style type="text/css">
@media (min-width: 481px) and (max-width: 768px) {
  #banner { width:740px; }
  #banner img { max-width:740px; max-height:222px; }
  #main { width:740px; }
  #main-content { width:450px; float:left; }
  #widget-container { width:200px; float:right; }
  .widget-content { width:160px; }
}
```

```
@media (min-width: 321px) and (max-width: 480px) {  
  #banner { width:450px; }  
  #banner img { max-width:450px; max-height:135px; }  
  #main { width:450px; }  
  #main-content { width:400px;}  
  #widget-container { width:400px; }  
  .widget-content { width:120px; margin:5px; float:left;}  
  .widget-text { display:none; }  
}
```

```
@media (max-width: 320px) {  
  #banner { width:275px; }  
  #banner img { max-width:275px; max-height:83px; }  
  #main { width:250px; }  
  #main-content { width:250px;padding:0px;}  
  #widget-container { width:250px; padding:0px; }  
  .widget-content { width:250px; margin:5px;}  
  .widget-text { display:none; }  
}  
</style>
```

It is important to note that many developers do not create responsive design scripts from scratch. Instead, they download and inject a boilerplate such as Gridiculous, [stuffandnonsense.co.uk](http://stuffandnonsense.co.uk), Skeleton, Less Framework, Gumby Framework, and many more.