How to make a website mobile ready and “Responsive” in 6 steps.

First, let’s review the three most common types of layouts being used in Web Design.

**Fixed design** uses fixed layouts using specific pixel dimensions like 960px.

**Fluid design** uses layouts that adjust to a screen using % sizing like 100%.

**Responsive Web Design (RWD)** uses media queries to adjust a layout by allowing CSS for different screen resolutions and RWD typically uses three techniques for making webpages adapt well across many screen resolutions: fluid design, media queries, and flexible images.
**Fixed design** using a container style with a fixed width (Non responsive). This is what most students are used to using for their layouts.

```
<div class="container">
layout tool
</div>
```

CSS for a typical fixed-width layout:
```
.container {
  width:960px;
  background:#ccc;
  margin:0 auto;
  padding:20px;
}
```

OR instead use a wrapper

```
<div id="wrapper">
layout tool
</div>
```

CSS for a typical fixed-width layout:
```
#wrapper {
  width:960px;
  background:#ccc;
  margin:0 auto;
  padding:20px;
}
```

Smartphones / tablets display the fixed design, scaling to fit the page to the devices screen resolution. Users can zoom in and out in order to better view the content.
Recommended CSS default values and resets

Before we get started let's make sure we have the following in our CSS in order to get the best results from our RWD examples.

```css
html   { font-size:100%; }
html, body, ul, li, 
    { margin: 0;
      padding: 0; }
.container { margin: 0 auto; }
```
Step 1 Setting the viewport

Use Meta Viewport Element To Identify Desired Screen Size

The viewport is used with mobile browsers can render web pages without a problem and scale them so they fit nicely inside the phone browser’s visible screen area or viewport — the user can zoom in on areas of interest. The viewport meta tag tells the device at what scale to render the page.

Put the following code inside the head

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

good explanation about the viewport declaration
https://developer.mozilla.org/en-US/docs/Mozilla/Mobile/Viewport_meta_tag
for official specs on the viewport checkout
http://www.w3.org/TR/css-device-adapt/
simple explanation
http://webdesign.tutsplus.com/tutorials/quick-tip-dont-forget-the-viewport-meta-tag--webdesign-5972
**Step 2 Add media queries**

Add media Queries for the desired screen resolutions using min-width and max-width.

```css
@media screen and (max-width: 000px) { style { property:value} }
```

Examples of media queries with properties and values, goes in side CSS

```css
@media screen and (max-width: 640px) { body { font-size:1rem; }     p { color:black; } }
@media screen and (max-width: 800px) { body { font-size:1.2rem; }    p { color:red; } }
@media screen and (max-width:1024px) { body { font-size:1.5rem; }    p { color:black; } }
```

Paragraph text color in the max-width: 800px example will be red when the screen is scaled between 641 and 800px.

You can also hide items or change how they are displayed

```css
@media screen and (max-width: 640px) { .footer { display: none; }    nav { display: block; } }
```

For official specs on media queries

http://www.w3.org/TR/css3-mediaqueries/

Most popular screen resolutions to possibly target?

- max-width 640
- max-width 800
- max-width 1024
- max-width 1280
- min-width 1281
Step 3 Incorporating fluid design into your layout

Fluid design uses layouts that adjust to a screen using % sizing like 100%

```html
#wrapper {
  width: 85.3%;
  background:#ccc;
  margin:0 auto;
  padding:1.67%;
}
```

The following is an example of how to choose what % based on your target vs actual or context display. This is not required just suggestive on how best to pick a %

/* 1024px / 1200px = 85.3 */
/* target / context = % or target screen size / wrapper size = % in fluid measurement */
Step 4 Setting images to adjust their size to be scalable

```css
img {
  max-width: 100%;
  height: auto;
}
```

OR define only certain images

```css
.imgResponsive {
  max-width: 100%;
  height: auto;
}
```
Step 5 Test “Responsive” layouts

An important component of responsive design, media queries and fluid design is to actually test your page. Test using the following link:

http://mattkersley.com/responsive/

And finally check an uploaded website to see if it is validated as mobile friendly by google.

Step 6 Add fallbacks for older browsers

To make sure our RWD works in future browsers, we should also implement CSS Device Adaptation, which will be placed in the CSS stylesheet, instead of within the HTML document. For ie10 add this inside css file but not in the html

```css
@viewport{
  zoom: 1.0;
  width: extend-to-zoom;
}
@-ms-viewport{
  width: extend-to-zoom;
  zoom: 1.0;
}
```

Also add these meta tags above the viewport tag for some browser issues

```html
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
<meta name="HandheldFriendly" content="true">
```

IE=edge means IE should use the latest (edge) version of its rendering engine

chrome=1 means IE should use the Chrome rendering engine if installed

Tells the browser if the page has been built to be viewable on a small screen. Really just a true or false event, as it offers no granular control over the width of the document. Used by older obscure phones and possibly windows Phone?

It is also recommended that you add a fallback for older browsers that do not understand rem sizing by adding px sizing to each property. I would list this before the rem styling like this.

```css
{ body {
  font-size:14px;  /*fallback*/
  font-size:1rem;
}
```

-end of rwd steps
Notes screen resolutions, css3 sizing, and references
Which screen resolutions should you target? checkout example of iOS devices
http://www.iosres.com/

various common displays on iOS mobile devices
640
750
768
800
1024
1080
1242
1280
1536
1960

CSS3 notes about sizing
Also note CSS3 introduces a few new units, including the rem unit, which stands for "root em". The rem unit is relative to the root—or the html—element. That means that we can define a single font size on the html element and define all rem units to be a percentage of that without compound problems associated with em.

html { font-size: 62.5%; }
body { font-size: 1.4rem; } /* =14px */
h1   { font-size: 2.4rem; } /* =24px */

References
Making a website responsive in 3 easy steps
http://www.catswhocode.com/blog/making-a-website-responsive-in-3-easy-steps

Mobile Web Application Best Practices
http://www.w3.org/TR/mwabp/#bp-viewport

What's responsive Web design all about?
http://arstechnica.com/business/2012/05/17/whats-responsive-web-design-all-about/

-end