MDDN352 2019.03.06

Week 01 Workshop Technical Exercise

Responsive Page

Make a simple text based web page that scales to different monitor sizes and shows the current viewport height and width.

Take notes on your techique and research links and add the text to your page.

You can try different approaches like.

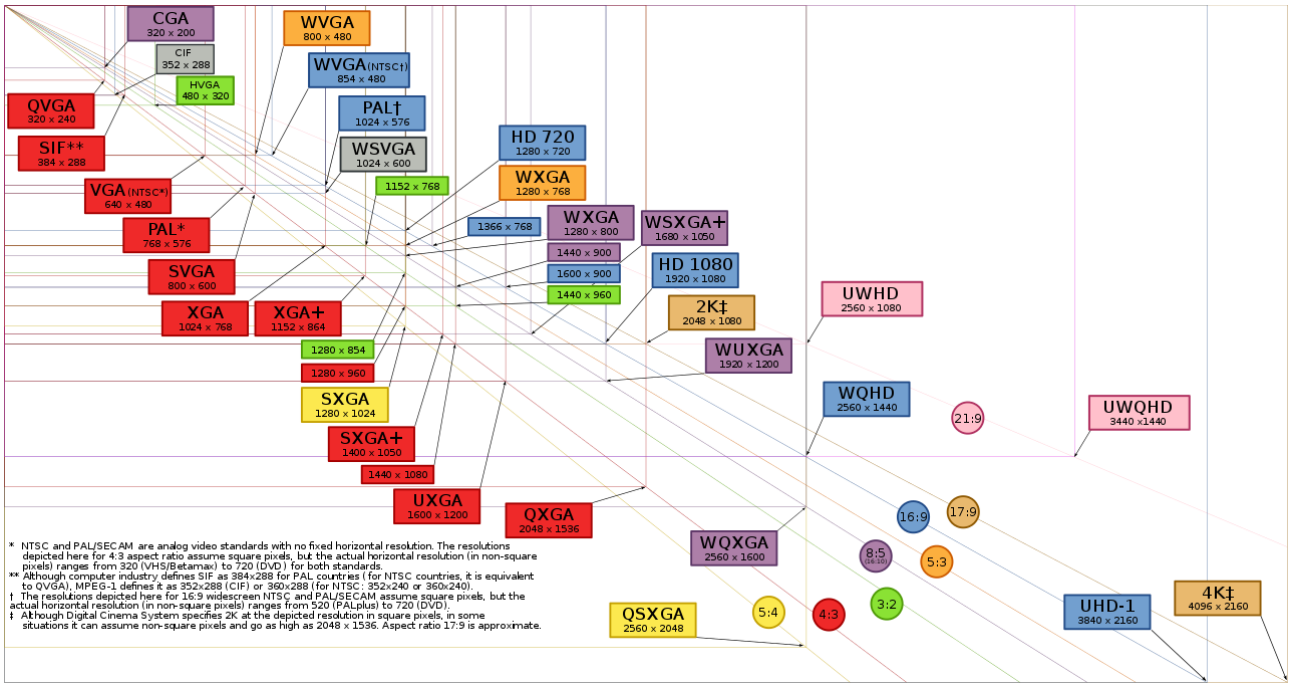
1. CSS @media directive
2. javascript var w = window.innerWidth
3. jQuery(window).width()jQuery(window).width()

Example research CSS @media

The move from computer based browsing to mobile devices and tablets has resulted in a proliferation of screen sizes and orientations. High definition screens like the Apple Retina displays on phones iPads and desktop displays complicates things further.

Look at the pixel dimensions of a number of screens.

https://en.wikipedia.org/wiki/List_of_common_resolutions



In the early days of online media on phones we would create separate versions of sites for mobile phones and touch devices like tablets.

CSS using the @media directive a browser could choose particular directives based on its rendering context. Here are some using print, screen and handheld.

```
@media print {
    body { font-size: 10pt }
}
@media screen {
    body { font-size: 13px }
}

@media handheld {
    body { font-size: 10px }
}
@media screen, print {
    body { line-height: 1.2 }
}
```

Look at the difference between http://m.trademe.co.nz and http://trademe.co.nz

Now with hi-res phones having higher resolutions than many laptops we cannot assume anything about the screen dimensions of the viewing device or the orientation, and it is not practical to create multiple versions to suit all devices.

We therefore use responsive design techniques to adjust the layout fluidly depending on screen resolution.

Some examples of responsive layout. Try these on a laptop browser, a mobile device or tablet, with and without auto rotate.

http://wellington.govt.nz http://mediaqueri.es/ https://responsivedesign.is/examples/

Look for other examples of responsive web design - make your own list of successful layouts.

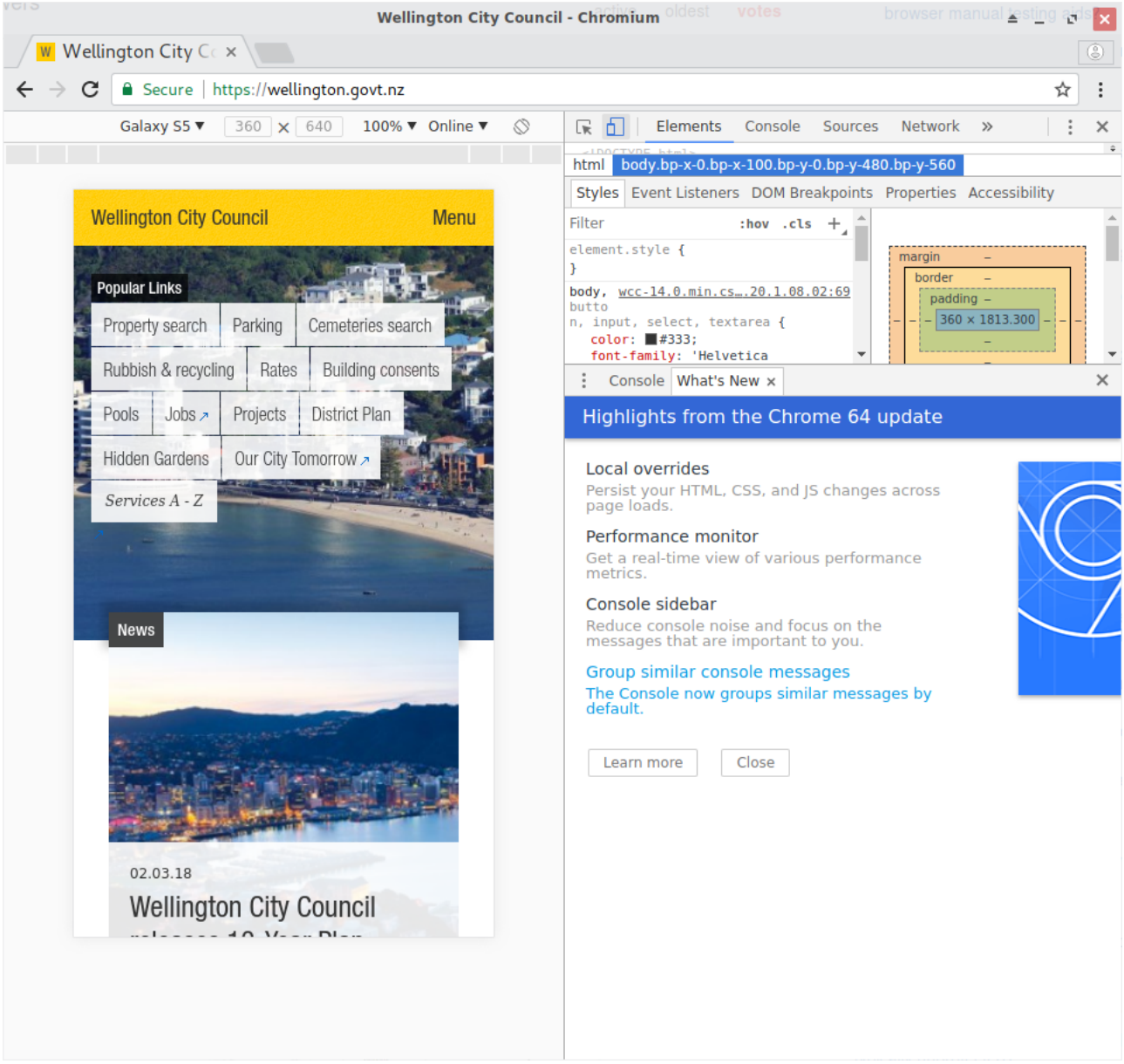
Read this article. Guidelines for responsive design

Responsive browser views

Use the device views in chrome and firefox to test different resolutions on different devices.

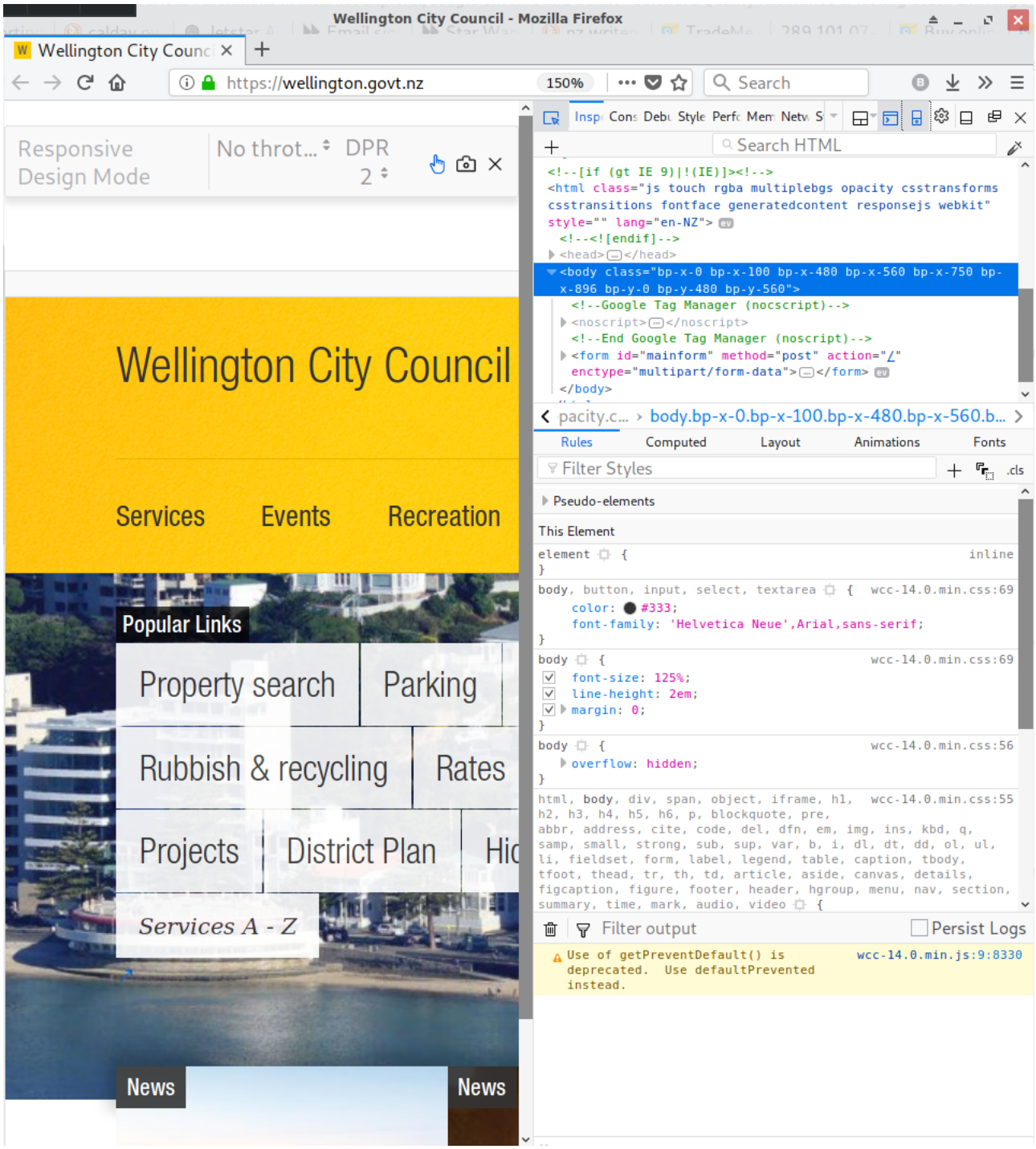
Chromium

`rightclick > inspect` or `CTRL-SHIFT-I` then `Toggle device toolbar` or `CTRL-SHIFT-M`



Firefox

`menu > web developer > inspector` then `responsive design mode` or `CTRL-SHIFT-M`



Task - breaking flow with @media

One way of reflowing a page at different screen widths is using this CSS directive `@media screen and (max-width: 480px)`

```
@media screen and (max-width: 480px) {
    .nav  {
    display: block;
    }
}
```

This is a CSS directive block that will activate when the width of the browser is less that 480px.

Try making a simple list based nav bar that responds to the media width.

Make a simple list as a nav bar for a web page

- home
- contact
- article

Use CSS to make it flow horizontally.

```
.nav ul{
    display: inline
}
```

Try rewriting this using the `@media screen and (max-width: 480px)` directive to change the menu style to :block as you rescale the browser to make it flow vertically.

Try making a text field that changes font size in em at different width break points.

Here's another tutorial that you can try as an exercise.

http://www.creativebloq.com/responsive-web-design/build-basic-responsive-site-css-1132756

Download demo zip archive to your desktop and unzip.

Experiment with changing aspects of the CSS to see how it works.

Alternatively try this jquery based method:

Responsive Jquery Example

Task - Changing menus to forms for small screens.

On phones and other small screens, menu items become unclickable. One approach is to replace them with forms which can then be selected using the built in form feature of smart phone operating systems.

Try this tutorial:

https://css-tricks.com/convert-menu-to-dropdown/

Duplicate your nav menu using the `<select>` tag to create a pulldown menu.

```
<nav>

  <ul>
    <li><a href="index.html" class="active">Home</a></li>
    <li><a href="page1.html">page 1</a></li>
    <li><a href="page2.html">page 2</a></li>
    <li><a href="page3.html">page 3</a></li>

  </ul>

  <select>
    <option value="" selected="selected">Select</option>

    <option value="index.html">Home</option>
    <option value="page1.html">Page 1</option>
    <option value="page2.html">Page 2</option>
    <option value="page3.html">Page 3</option>
  </select>

</nav>
```

Use CSS to hide the select by default:

```
nav select {
  display: none;
}
```

.. and make it visible below 960px:

```
@media (max-width: 960px) {
  nav ul     { display: none; }
  nav select { display: inline-block; }
}
```