**Responsive Web Design – HTML**

A Web site deigned to look good on any device that accesses it.

A responsive design automatically adjusts(or reflows) content for different screen sizes and viewports.

HTML and CSS can be used to automatically resize, shrink, enlarge or hide elements on a website so that it looks good on all devices.

**The Viewport**

The viewport is the term for the visible area of a webpage on a device.

This has become a major consideration when designing websites and Apps as device screen sizes vary so much.

Websites should look as good on a mobile phone as they do on a desktop PC.

A user should be able to resize a webpage and the content should adjust or reflow automatically.

A picture containing text, indoor, bunch, different

Description automatically generated

**Setting the viewport**

You can use following tag to set the viewport of your page:

<meta name="viewport" content="width=device-width, initial-scale=1.0">

This tag goes inside the head tags e.g.:

Text

Description automatically generated



Sets the width of the page to follow the screen-width of the device.



Sets the initial zoom level when the page is first loaded by the browser.

Setting the viewport gives the browser instructions on how to control the dimensions and scaling of the webpage.

Graphical user interface, application

Description automatically generated

Text

Description automatically generated

Both are screenshots of the iPhone X with and without the viewport meta tag. Otherwise, all the other elements on the page (apart from some text) are the same.

**Responsive Images**

A responsive image will scale as the browser size changes.

If the image has a CSS width property of 100% it will be responsive and resize as the viewport changes.

For example:

<img src="my\_fish.png" **style="width:100%;"**>

The problem with this is that although the image will scale down and become smaller than it originally was, it could also scale up to be much bigger than its original size. This could cause image quality problems.

**The max-width property.**

Setting the max-width property instead of the width property to 100% will result in the image scaling down when required, but not scaling up above its original size.

<img src="my\_fish.png" style="**max-width:100%;**height:auto;"

Questions:

Describe responsive design in your own words.

What is a viewport and why is it important?

Where does the meta tag containing the viewport information go in a HTML page?

Why is it better to use the max-width property than the width property?

**Create**

Make a new folder for your responsive design tasks.

* Take a screen snip of the image below and save it as a PNG file in this folder.

A fish swimming in water

Description automatically generated with low confidence

* Create a HTML file in the folder that contains the image and a paragraph of text about it.
* The image should be responsive and not exceed its original size.
* The viewport should be set.
* Screen shot your page viewed on an iPhone X

**Swap Images depending on browser width**

You can change the image depending on the size of the browser using the HTML element <picture>

Text

Description automatically generated

Graphical user interface, application

Description automatically generated

Text

Description automatically generated with medium confidence



**Task 2**

* Create a HTML file that displays 3 different images depending on the size of the browser window.
* Your file should also have a heading and a paragraph of text describing when the image size will change.
* To check the size of an image right click on the file, select properties and then click the details tab.
* Screen shot each of the images in the browser window.

**Responsive Text**

You may have noticed that although the images resized the text stayed the same size. You can set the text to resize/scale as well.

You can set the text size with a ‘vw’ unit. This stands for ‘viewport width’

Example:

<h1 style="**font-size:10vw**">Responsive text is ace!</h1>

Viewport is the browser window size. 1vw = 1% of viewport width.

If the viewport is 50cm wide, 1vw is 0.5cm.

**Task 3**

* Make the heading in task 2 responsive text.
* Screen shot the different sizes of text.

**Grid-View**

Web pages can be made using a grid-view. This view divides the page into columns.

Any amount of columns can be used but a responsive grid view will often have 12.

A picture containing table

Description automatically generated

You can then use these columns to control the content on the page.

A picture containing chart

Description automatically generated

The 12 columns should add up to 100%. To find the width of each column we divide 100 by 12 to arrive at 8.33%.

This responsive grid will resize as the browser window resizes.

**Writing the CSS and HTML for the Responsive Grid-View.**

There are some CSS properties that should be added:

Text

Description automatically generated

All HTML elements should have box-sizing set to border-box. This ensures that the padding and border are included in the total width and height of the elements.

Find out more about this property: <https://www.w3schools.com/css/css3_box-sizing.asp>

**Defining the columns**

Text

Description automatically generated

Each column is designed as a class as above.

The col-1 class represents a span of 1 column.

The col-2 class represents a span of 2 columns.

The col-3 class represents a span of 3 columns etc.

All of these column classes should float to the left and have a padding of 15px.

Text

Description automatically generated

The last important piece of CSS to include is:

Text

Description automatically generated

This prevents other elements being placed as if the columns do not exist. This style clears the flow.

**Full CSS and HTML:**

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Each row in the design should be wrapped in a <div>.

The columns inside the rows should also be wrapped in <div>s and should always add up to 12.

The row in the example above is made up of 2 columns **col-3 25% and col-9 75%.** This add up to **100%.**

**Task 4**

* Create a RWD Grid-View with 12 columns.
* You can use the code above as a template.
* Add 2 more rows to the design.
* One row should have 3 columns with content in each, the columns do not have to all be equal in width but they must add up to 100%.
* The other row should be just 1 column with content.
* Add CSS to put a border around your columns so you can see them clearly in the browser.

**Media Queries.**

The web site you have just made is responsive but when viewed on a small screen it did not look very good.

Media Queries allow you to create different layouts depending on the size of the viewport. These are called **Breakpoints.**

They can also be used to detect other environmental things e.g. the user is using a touchscreen not a mouse.

It uses the @media rule to include a block of CSS properties, only if a certain rule/condition is true.

This consists of:

* **A media type**, which tells the browser what kind of media this code is for (e.g. all or screen)
* **A media rule** @media this is a test that must be passed for the CSS rules contained in a block to be applied. You can also think of it like a condition that must be True.
* **CSS rules**, these rules will be applied if the tests are passed and the media type is correct.

**Mobile First Design**

You should always design your website to be viewed on a mobile device first before you design for a larger desktop or tablet.

The CSS needs to be changed so that the style changes when the width gets **larger than 768px** not smaller. This change makes our design **mobile first.**

**To do this we add the following CSS:**

Text

Description automatically generated

The default is that all the columns are 100%. This is for a mobile phone so that all the columns are on top of each other instead of side by side.

The line:

Text

Description automatically generated

is the media rule concerning the screen. If the screen width is larger than 768px the CSS inside the block will be processed. This is for a desktop screen.

**Task 5**

* Add media queries to your website so that it displays properly on a phone and a desktop.

So far we have considered a mobile then a desktop. We can insert as many breakpoints as we like.

Lets add a break point for a tablet in the middle of these two sizes.

Chart, bar chart

Description automatically generated

The new media query for this breakpoint should specify a new set of classes for any device larger than 600px and smaller than 768px.

The new set of media queries are shown below:

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, text

Description automatically generated

It looks like there are 2 sets that are exactly the same but one is called:

col-s-# and the other just col-#.

This allows you to specify different column widths for each media query in your HTML.

**Example:**

Text

Description automatically generated

The code above works as follows:

**For Desktop:**

Graphical user interface, text

Description automatically generated

The first and third section (<div>) will span 3 columns, the middle section will span 6 columns.

**For Tablets:**

Graphical user interface, text, application, email

Description automatically generated

The first section will span 3 columns, the second will span 9, and the third section will span 12 and is displayed below the other 2.

**For Mobile:**

Graphical user interface, text, application

Description automatically generated

All the columns are set to 100% width.

**Task 6**

* Add additional media queries so that there are now 3 breakpoints

**Orientation**

A media query can be used to change the layout of a page depending on the orientation of the browser.

Graphical user interface, text

Description automatically generated

Text

Description automatically generated

**Hide Elements**

Media queries can be used to decide which elements are visible on different screen sizes. For example, elements can be hidden on smaller screens.

Text

Description automatically generated with medium confidence

Any <div> with the class ‘gone’ will be hidden if the screen is smaller than 600px.

**Font Size**

Media queries can be used to change the font size of an element depending on the size of the device screen.

Text

Description automatically generated

Reference:

<https://www.w3schools.com/cssref/css3_pr_mediaquery.asp>

**Task 7**

Add additional media queries:

* to change the background colour of your <h1> element when the device screen is in landscape.
* to adjust the font size to match the device size.

A screenshot of a computer

Description automatically generated with medium confidence