Workshop: Introduction to HTML

King's College School

June 2019

1 Introduction

In this workshop, we will learn to make a personal web-page you can view in the browser (like internet explorer or chrome). If you wanted to, you could then deploy the website to a **web server** on the internet for everyone to view.

2 Basics

Websites are created using `html` files.

Html is a markdown language. A markdown language is like a programming language. But instead of telling a computer what to do, it tells a computer how to format a document. In the case of HTML, it tells the browser (Chrome, Internet Explorer) how to display a web page.

The basic building block of html are **tags**. Tags tell the computer how to show something. There is always an opening tag and a closing tag. An opening tag looks like `<this>` and a closing tag looks like `</this>`. The content goes in between the opening and closing tag.

Example: This is an example of a `p` tag. `p` stands for paragraph:

```
<p>Here's some text</p>
```

This will create a paragraph containing the text "Here's some text".

To create more complex structures, we can compose tags. For instance, to have bold text inside a paragraph, we would use the following:

```
<p>Here's some<b>BOLD</b> text</p>
```

This would output "Here's some **BOLD** text".

There are a few required, special tags to every html document:

`<!DOCTYPE html>` should be the first thing on the page. This tells the browser that it's reading a html file. Note that this tag is special: It is the only tag that does not requrie a closing tag!

`<html></html>`: Everything except for the doctype tag goes between these two tags.

`<head> </head>`: Between these two tags is where we put *site metadata*. This is content that will not be shown in the page, but is required. For instance, the title that will be displayed on the tab at the top of the page can be set using the `<title>Title goes here</title>` tag inside the `<head>` section.

`<body> </body>`: Between these two tags is where the actual content of the page goes. Anything that should be displayed goes here.

Task: Create a new html file on your computer. Double-click it to open it in the browser. You should see an empty page. Open the same document with your text editor. Add the required tags to make the html document work properly. Add a title inside the head section and a paragraph to the body. Refresh the page in the browser, and you should see your changes.
If you get stuck, don't be afraid to use google to find help, ask a classmate or ask me.

3 Some common tags

The following tags are used commonly: This tag creates a heading: `<h1></h1>`
This tag creates a smaller heading: `<h2> </h2>`. This goes on all the way down to `<h6>`
This tag creates a paragraph: `<p> </p>`
The `<div></div>` tag (divider tag) is used to group elements. This will be useful later when we learn about styles.

Task: Add a heading with your name to the page. Underneath, add a paragraph with one or two sentences about yourself - maybe talk about your favorite food, activity or programming language?

4 Tag attributes

Tags can also have **attributes**. Attributes can provide additional information. They look like this `<tag attribute="something"> </tag>`.
This can be useful for the following tags:
`<img src="/path/to/image"></img>`. This will create an image on the screen, and the `src` attribute specifies which image to load.
`<a href="google.com">This is a link</a>`. This creates a link. Clicking on the text between the tags will take you to the website specified by the `href` attribute.

5 Some more tags

`<a></a>` tag: The anchor tag is used to create links. It takes the `href` attribute to know where the link should point to.
`<img></img>` tag: The imgage tag is used to add images. It takes the `src` attribute to know where the image should be fetched from. The src tag can be any link to an image on the internet, or a local file on your computer.

Task: Find an image you like, either on your computer or on google. Copy the URL of the image and add it to your page using an `<img>` tag.

After adding your image, you may notice that it is much too large, or much too small. We will cover how to change the size in the next section.

6 Styles

You can also name certain tags on the screen by using the `id` attribute. For instance, write `<h2 id="my-title">Here's a title</h2>`. After you have given an element an id, you can change its **style**. Changing an element's style is done using the style tags:
In the **head section** (not body) of your html file, put the following:

```
<style>
    #my-title {
        color: "red";
    }
</style>
```

The style tag defines styles. Inside the style tag, use the hash symbol and name of your element to select it. Then, inside the brackets, you can set properties like color and size.

The language that you use inside the `style` tags is not html - it is called CSS (Cascading Style Sheet). It consists of a selector (in this case `my-title`), and after the brackets, a list of styles to apply to that selector, separated by semi colon. You can find a list of different styles you can change by using google.

Hint: If you would like to use the same style for multiple elements, you can use a `class` attribute instead. Multiple elements can have the same class attribute name. Then, select that class using `.classname` in CSS to set styles.

Task: Using google, find a list of colors you can specify in CSS. Then, change the colour of your title to something you like.

Task: Add an id to your image tag. Then, using CSS, set the `width` property of the image to `200px`. This should fix your image if it is too small or too large.

7 Linking to external stylesheets

Typing all of your CSS styles in the head section of your HTML file is messy. For this reason, we can **link to external stylesheets**. This is done using a `<link>` tag. The link tag takes three attributes:
`rel=`, this should be set to `'stylesheet'`
`type=`, this should be set to `'text/css'`
`href=`, this should be set to the link to the stylesheet. Again, like for the `img` tag, this can be a local file on your computer or any URL in the internet.

Task: Cut all of the styles you created from the HTML file, and delete the style tag. Create a new file called `index.css` in the same folder you created your `html` file in. Paste all of your old styles into the newly created `css` file.
In the head section of your `HTML` file, add a `link` tag. For the `href` attribute, set the value to `'./index.css'`. The `'./'` tells the browser to look for this file in the same folder as the HTML file.

8 More CSS Selectors

So far, we have used `id` and `class` to select different elements on the page. However, you can also select elements using simply their tag name. For instance, using the following will change the colour of all `p` tags:

```
p {
  colour: 'green';
}
```

Task: Some elements also have default styles: Select the `body` tag and set the css-property 'margin' to 0. You should see the default margin disappear. We will set our own layout in section 10.

9 Changing fonts

The default font looks quite ugly. You can link fonts to your html file, also by using the `link` tag, just like linking to stylesheets. Once you have linked to a font, you can the use it in CSS like this (replace yourSelector with some selector):

```
yourSelector {
  font-family: 'YourLinkedFontName';
}
```

Task: Go to fonts.google.com. Find a font you like, and click the little + icon. There should be a pop-up with a `<link>` tag you can copy into your html file. Paste the tag into the html, and set all of your text and headings to use your chosen font.

10 Using divs for layout

So far, the layout is doesn't really work: Everything is cluttered over to the side. By surrounding your content with a div tag, you can lay out your content better.

Task: Create a new `div` tag, and give it an id of `container`. Set the following styles for your `container` div:

```
.container {
  position: relative;
  left: 0;
  right: 0;
  margin-left: auto;
  margin-right: auto;
  max-width: 600px;
}
```

This should center your content on the screen.

The main CSS properties you use for layout are the `position, display, margin` and `padding` properties. However, layout can become quite complicated. You can find out more about these properties using google.

11 Extensions

1. Can you find a way to align your title to the center? Tip: use `text-align`.
2. Add a caption to the image. Use a div to group the image and the caption together.
3. Can you figure out how to add rounded borders to the image, like on the example website?
4. Can you figure out how to create a bullet-point list, like on the example website?