

# Coding Workshop

---

## Creating Your Online Presence

Mathijs de Jong

# Outline of the Presentation

1. Introduction
2. Installing Basic Software
3. What Will Your Website Look Like?
4. Dealing with Files
5. HTML Basics
6. CSS Basics
7. JavaScript Basics

# What is Web Development?

- The work involved in developing a web site for the internet or an intranet.
- Three kinds of web developer specializations:
  - Front-end developer
  - Back-end developer
  - Full-stack developer
- Our focus: front-end development

# Installing Basic Software

- Requirements:
  - A computer
  - A text editor
    - » Sublime Text 3
    - » Notepad (Windows)
    - » Atom
    - » Notepad++
  - A web browser
    - » Google Chrome
    - » Mozilla Firefox
    - » Microsoft Edge
  - A graphics editor
    - » GIMP
    - » Adobe Photoshop
    - » Sketch

# What Will Your Website Look Like?

- Before you start writing the code, you should plan it first
  1. What is your website about?
  2. What information are you presenting on the subject?
  3. What does your website look like?
- Next, grab pen and paper and sketch out roughly how you want your site to look
- **TIP:** Use templates!

# Dealing with Files

- Create a folder for your website
- Inside the folder you should include:
  1. ***index.html***: This file will generally contain your homepage content, that is, the text and images that people see when they first go to your site.
  2. ***images*** folder: This folder will contain all the images that you use on your site.
  3. ***styles*** folder: This folder will contain the CSS code used to style your content.
  4. ***scripts*** folder: This folder will contain all the JavaScript code used to add interactive functionality to your site.

# Creating Your First Website

1. Open up your *index.html* file, and insert the following code into the file exactly as shown.

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <h3>Hello World!</h3>
  </body>
</html>
```

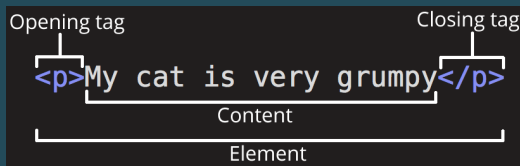
2. Save your HTML file, then load it in your web browser (double-click the file). You should see your new web page displaying "Hello World!".

# HTML — HTML Basics

- **Hypertext Markup Language**
- Structure a web page and its content
- A ~~programming~~ markup language
- Series of elements that enclose, or wrap, different parts of content to make it appear or act a certain way.

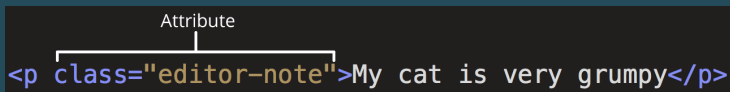


# HTML — Anatomy of an HTML Element



1. **The opening tag:** This consists of the name of the element (in this case, `p`), wrapped in opening and closing angle brackets. This states where the element begins or starts to take effect.
2. **The closing tag:** This is the same as the opening tag, except that it includes a *forward slash* before the element name. This states where the element ends.
3. **The content:** This is the content of the element, which in this case is just text.
4. **The element:** The opening tag, the closing tag, and the content together comprise the element.

# HTML — Attributes

A diagram showing an HTML tag with an attribute. The text is `<p class="editor-note">My cat is very grumpy</p>`. A bracket above the `class="editor-note"` part is labeled 'Attribute'. The `class` part is highlighted in blue, and the `"editor-note"` part is highlighted in orange.

```
graph TD
    A[Attribute] --- B[ ]
    B --- C[class="editor-note"]
```

- Attributes contain extra information about the element that you don't want to appear in the actual content
- `class` is the attribute *name*
- `editor-note` is the attribute *value*

# HTML — Anatomy of an HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <h3>Hello World!</h3>
  </body>
</html>
```

- `<!DOCTYPE html>` — The doctype.

# HTML — Anatomy of an HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <h3>Hello World!</h3>
  </body>
</html>
```

- `<html></html>` — This element wraps all the content on the entire page and is sometimes known as the root element.

# HTML — Anatomy of an HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <h3>Hello World!</h3>
  </body>
</html>
```

- `<head></head>` — This element acts as a container for all the stuff you want to include on the HTML page that isn't the content you are showing to your page's viewers.

# HTML — Anatomy of an HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <h3>Hello World!</h3>
  </body>
</html>
```

- `<meta charset="utf-8">` — This element sets the character set your document should use to UTF-8, which includes most characters from the vast majority of written languages.

# HTML — Anatomy of an HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <h3>Hello World!</h3>
  </body>
</html>
```

- `<title></title>` — This sets the title of your page, which is the title that appears in the browser tab the page is loaded in.

# HTML — Anatomy of an HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <h3>Hello World!</h3>
  </body>
</html>
```

- `<body></body>` — This contains all the content that you want to show to web users when they visit your page, whether that's text, images, videos, games, playable audio tracks, or whatever else.



# HTML — Images



```

```

# HTML — Marking Up Text

- Headings
  - Heading elements allow you to specify that certain parts of your content are headings or subheadings.
  - HTML contains 6 heading levels, `<h1>`–`<h6>`.
- Paragraphs
  - `<p>` elements are for containing paragraphs of text; you'll use these frequently when marking up regular text content.
- Lists
  - *Unordered lists* are for lists where the order of the items doesn't matter, such as a shopping list. These are wrapped in a `<ul>` element.
  - *Ordered lists* are for lists where the order of the items does matter, such as a recipe. These are wrapped in an `<ol>` element.
  - Each item inside the lists is put inside an `<li>` (list item) element.

# HTML — Links

- To add a link, we need to use a simple element — `<a>` — "a" being the short form for "anchor".
- To make text within your paragraph into a link, follow these steps:
  1. Choose some text. We chose the text "Mozilla Manifesto".
  2. Wrap the text in an `<a>` element.
  3. Give the `<a>` element an href attribute.
  4. Fill in the value of this attribute with the web address that you want the link to link to:

```
<a href="https://www.mozilla.org/en-US/about/manifesto/">Mozilla Manifesto</a>
```

# HTML - Marking up a Letter Exercise

- Now that you are familiar with HTML, you can start practicing with it
- Before you get started, you might want to read the [HTML Basics](#) article
- Try to work on the [marking up a letter exercise](#)
- Check out the articles in the [HTML — Structuring the Web](#) section to find more detailed explanations on how to solve the marking up a letter exercise

# CSS — CSS Basics

- Cascading Style Sheets
- Code used to style your web page
- A programming markup style sheet language
- Used to apply styles selectively to elements in HTML documents

# CSS — Apply Your CSS to Your HTML

## 1. External stylesheet

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My CSS experiment</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <h1>Hello World!</h1>
    <p>This is my first CSS example</p>
  </body>
</html>
```

```
h1 {
  color: blue;
  background-color: yellow;
  border: 1px solid black;
}
```

```
p {
  color: red;
}
```

# CSS — Apply Your CSS to Your HTML

## 2. Internal stylesheet

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My CSS experiment</title>
    <style>
      h1 {
        color: blue;
        background-color: yellow;
        border: 1px solid black;
      }

      p {
        color: red;
      }
    </style>
  </head>
  <body>
    <h1>Hello World!</h1>
    <p>This is my first CSS example</p>
  </body>
</html>
```

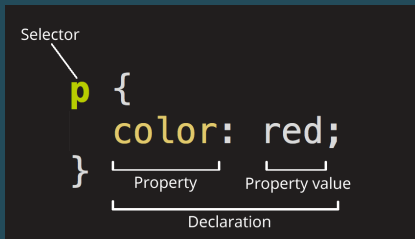
# CSS — Apply Your CSS to Your HTML

## 3. Inline styles

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My CSS experiment</title>
  </head>
  <body>
    <h1 style="color: blue;background-color: yellow;border: 1px solid black;">Hello World!</h1>
    <p style="color:red;">This is my first CSS example</p>
  </body>
</html>
```



# CSS — Anatomy of a CSS Ruleset

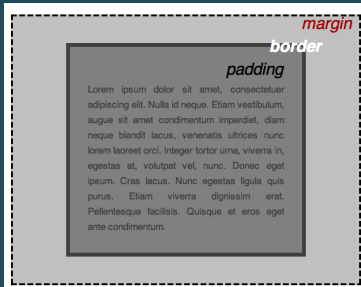


- The whole structure is called a *rule set*
- *Selector* — Selects the element(s) to be styled
- *Declaration* — Specifies which of the element's properties you want to style
- *Properties* — Ways in which you can style a given HTML element
- *Property value* — Chooses one out of many possible appearances for a given property

# CSS — Different Types of Selectors

Selector name	What does it select	Example
Element selector (sometimes called a tag or type selector)	All HTML element(s) of the specified type.	<code>p</code> Selects <code>&lt;p&gt;</code>
ID selector	The element on the page with the specified ID. On a given HTML page, you're only allowed one element per ID (and of course one ID per element).	<code>#my-id</code> Selects <code>&lt;p id="my-id"&gt;</code> or <code>&lt;a id="my-id"&gt;</code>
Class selector	The element(s) on the page with the specified class (multiple class instances can appear on a page).	<code>.my-class</code> Selects <code>&lt;p class="my-class"&gt;</code> and <code>&lt;a class="my-class"&gt;</code>
Attribute selector	The element(s) on the page with the specified attribute.	<code>img[src]</code> Selects <code>&lt;img src="myimage.png"&gt;</code> but not <code>&lt;img&gt;</code>
Pseudo-class selector	The specified element(s), but only when in the specified state (e.g. being hovered over).	<code>a:hover</code> Selects <code>&lt;a&gt;</code> , but only when the mouse pointer is hovering over the link.

# CSS — Boxes, Boxes Everywhere



- CSS layout is based principally on the *box model*
- Each of the blocks taking up space on your page has properties like this:
  - *padding* — the space just around the content
  - *border* — the solid line that sits just outside the padding
  - *margin* — the space around the outside of the element

## CSS — Style Your Page Exercise

- Now that you are familiar with CSS, you can start practicing with it
- Before you get started, you might want to read the [CSS Basics](#) article
- Try to work your way through the [W3School CSS exercises](#)
- Check out the articles in the [CSS — Styling the Web](#) section to find more detailed explanations on how to solve the CSS exercises

# JavaScript — Make Your Web Page Dynamic

- Programming language that adds interactivity to your website
- Much more powerful and complexer than HTML and CSS
- External script

```
<script src="scripts/main.js"></script>
```

- Learn by doing with the [CodeAcademy — Introduction To JavaScript](#) course

# Time to Start Working on Your Own Website

Time to start working on your own project! However, before you get started, you might want to go over the following articles.

1. [What will your website look like?](#)
2. [Dealing with files](#)
3. [HTML basics](#)
4. [CSS basics](#)
5. [JavaScript Basics](#)
6. [Publishing via Github](#)
7. [JavaScript — A splash into JavaScript](#)
8. [How the Web works](#)

# Make Your Own Personal Website

- That's it! Now you know how the Web works and how you can add your content to it
- What better way to do this than by creating your own personal website?
- You can check out the following links for inspiration:
  - [Medium — 25 Best Personal Website Design Examples and Resources for Your Inspiration](#)
  - [Portfolio — Best Portfolio Websites](#)
- *TIP:* Start simple and work in sprints!