ACADEMIND

Browser Instructions: HTML, CSS & JavaScript

Option A

Write the instructions (i.e. HTML, CSS & JavaScript code) themselves

Option B

Write code that generates the browser instructions (HTML, CSS, JavaScript) dynamically

Core Technologies

HTML

The content and structure of the displayed page

CSS

The styling of the displayed page and its content

JavaScript

Interactivity that might be needed on the displayed page

ACADEMIND

How To Create A Website?

Your computer can act as a temporary development server

A Browser

Any browser works but Chrome and Firefox provide particularly good development support

+

A HTML File

A regular text file that just happens to have “.html” as a file extension and contains HTML code

ACADEMIND

You don’t need a fancy computer or special operating system!

ACADEMIND

HTML is a “Markup Language”

Instructs the browser about content, its structure and its meaning

Why HTML Elements?

<h1 style="font-size: 16px">...</h1>

...

Screenshot

Screenshot

Google

Blind

Developer

Understanding HTML Elements

Elements “tell” the browser which kind of content is displayed (headings, text, images, lists, buttons, ...)

HTML **Element**

<h1>Hello World!</h1>

Element **Content**

Opening Tag

HTML **Tags**

Closing Tag

< ELEMENT NAME >

ACADEMIND

Which HTML Elements Exist?

Which HTML Elements Exist?

There are many available HTML elements

Because there are a lot of things you can describe

You will learn about lots of important HTML elements in this course

+

You can then also use resources like MDN to look up all available elements (e.g. for niche use cases)

Why HTML Elements?

Without extra annotation, content often has no clear meaning

It's like writing just plain, unformatted text in Word

"Telling" the browser that something is a title / subtitle / image / ... allows the browser to present that content correctly

- To regular visitors of your website
- To search engine crawlers
- To visitors using assistive technologies (e.g. screen readers)

ACADEMIND

Working with Colors

There are >16mn colors to choose from in CSS

Colors can be defined in different ways

Color Keywords	Hexadecimal Values	RGB Values	HSL Values
red, green, blue, ...	#fa923f	rgb(250, 146, 63)	hsl(27, 95, 61)
Limited set of pre-defined colors	3 two-digit pairs (red, green, blue)	3 values for red, green & blue	3 values for hue, saturation & lightness
	Shorthand: #ffcc00 ➔ #fc0	Every value is between 0 and 255	

Working with Colors

You can define your own colors – there are over 16mn colors to choose from!

Colors can be defined for **various CSS properties** and in **different ways**

background-color
border-color
color
...

Hex Code
RGB
HSL

ACADEMIND

Defining Colors

Hex Code	RGB	HSL
Every color has its own unique hexadecimal identifier (can be split into red, green, blue “parts”)	Every color is created by combining a **r**ed, **g**reen and **b**lue decimal value	Every color is created by combining a **h**ue, **s**aturation and **l**ightness
#fa923f *(i.e. red: fa => 250, ...)*	rgb(250, 147, 63)	hsl(27, 95, 61)
6 characters between 0 and f each (can be shortened to 3 characters if each pair is equal: #ffcc00 ➔ #fc0)	Every value must be between 0 and 255. It’s the decimal version of hex numbers: #fa923f ➔ rgb(250, 147, 63)	Hue is an angle and hence must be between 0 and 359, saturation and lightness are percentages (0-100)

ACADEMIND

CSS Sizes & Size Units

Some CSS properties expect **numeric values** (e.g. a size value for the font-size property)

For CSS properties that expect a size (dimension), you got different options for defining that size

Absolute	Relative	
px	rem	%
A device-independent pixel on the screen	A base-font-size relative unit	Relative to the parent value

Working With Global CSS & CSS Selectors

A CSS Selector

In this case: A "**Type Selector**"
(selects by element type)

```
p {
  color: #534b4b
}
```

A CSS Property + Value

In this case: The color property and a hex color code

A CSS Rules

The Anatomy Of A Valid HTML Document (Page)

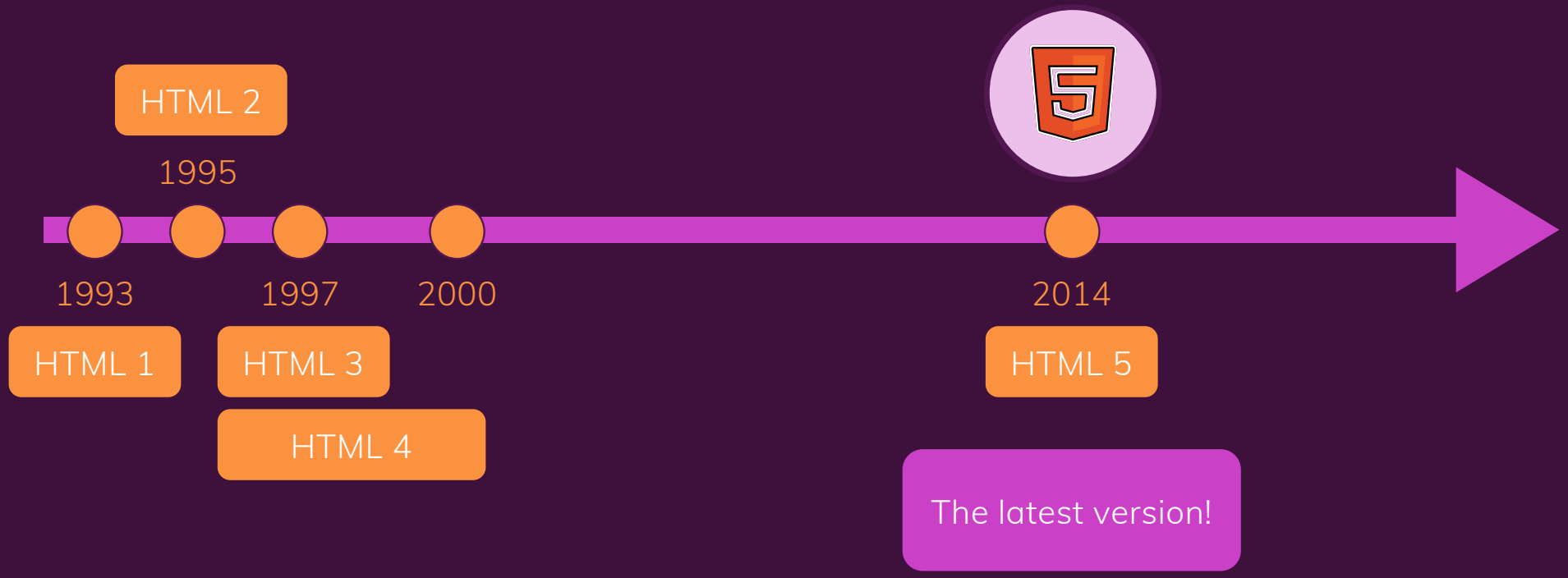
```
<!DOCTYPE html>
<html>
  <head>
    <title>My Page</title>
  </head>
  <body>
    <h1>Welcome!</h1>
  </body>
</html>
```

HTML Version

Page Metadata

Page Content

The History Of HTML



HTML Elements – Good To Know

Most HTML Elements have **opening** (`<...>`) and **closing** (`</...>`) tags

Element names ("`h1`") are **case insensitive**: "`h1`" and "`H1`" both work, but sticking to **lowercase only is the standard**

Choosing the **correct element** for expressing the content **matters**: It allows the browser to **display and describe** (e.g. to search engines or assistive technologies) **your content correctly**

HTML Element Attributes

Attribute Name

Attribute Value

```
<a href="https://google.com">My Best Friend</a>
```

HTML **Attribute**

Adds extra configuration to an element

In this case: The target location of the anchor tag (link)

ACADEMIND

Void Elements ("Self-Closing Elements")

```
<img src="me.jpg" alt="That's me!">
```

Some HTML elements are void – which means:
They have no content

They are configured with attributes only

ACADEMIND

Time To Practice!

1. Add the **base HTML document skeleton** + structure to the new about.html page
2. **Move the image** from the index.html to the about.html document (Bonus points: Create the image on about.html before removing it from index.html – i.e. don't cut & paste but re-create and remove)
3. Add a **title above the image** on the about.html page and add **some text below the image** (pick a proper HTML element)
4. On about.html, **add a link which links back** to the index.html page