

# FUTURA

# LA SCUOLA PER L'ITALIA DI DOMANI



Finanziato  
dall'Unione europea  
NextGenerationEU



Ministero dell'Istruzione  
e del Merito



**Italiadomani**  
PIANO NAZIONALE DI RIPRESA E RESILIENZA

DOCENTE	Shadi Lahham
Corso	Web Developer
Unità Formativa	Programmazione - Javascript e Typescript
Argomento	Specificato nel titolo della slide successiva

Intervento realizzato da  
**ITS**  
TECNOLOGIE  
DELL'INFORMAZIONE  
E DELLA COMUNICAZIONE

COESIONE  
ITALIA 21-27  
PIEMONTE



Cofinanziato  
dall'Unione europea



REGIONE  
PIEMONTE

# HTML CSS JS

## Introduction

Shadi Lahham - Web development

# Structure

# Web page Structure

## **Content**

Text, Media

## **HTML**

Structure

## **CSS**

Presentation

## **Javascript**

Logic/Interactivity

# Boilerplates

# HTML Boilerplate

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Website Title</title>
  <meta name="description" content="My new wonderful website">
  <meta name="author" content="Mister X">
  <link rel="stylesheet" href="./css/styles.css?v=1.0">
</head>

<body>
  <div>My Website</div>
  <!-- end of the body -->
  <script src="./js/scripts.js"></script>
</body>
</html>
```

# The Doctype

The first thing on an HTML page is the doctype, which tells the browser which version of the markup language the page is using.

**For XHTML 1.0 Strict:**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"  
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

**For HTML4 Transitional:**

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
    "http://www.w3.org/TR/html4/Loose.dtd">
```

**For modern HTML5:**

```
<!doctype html>
```

# The html Element

```
<!doctype html>  
<html lang="en">  
  
</html>
```

Represents top-level element of an HTML document  
Also referred to as the root element

All elements must be descendants of `<html>`



# The head Element - character encoding

UTF-8 is a character encoding capable of encoding all possible characters, or code points, defined by Unicode. The encoding is variable-length and uses 8-bit code units.

**XHTML and HTML4:**

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

**HTML5:**

```
<meta charset="utf-8">
```

[Metadata - Wikipedia](#)

# The head Element

```
<head>
  <meta charset="utf-8">
  <title>Website Title</title>
  <meta name="description" content="My new wonderful website">
  <meta name="author" content="Mister X">
  <link rel="stylesheet" href="/css/styles.css?v=1.0">
</head>
```

## Quick Exercise:

What is this for?

?v=1.0

How does the browser cache work?

# The body Element

```
<body>
  <div>My Website</div>
  <!-- end of the body -->
  <script src="./js/scripts.js"></script>
</body>
```

XHTML and older:

```
<script src="./js/scripts.js" type="text/javascript"></script>
```

HTML5:

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

The `<script>` element

- used to define a client-side script - JavaScript
- either contains Javascript code or points to an external script file via the src attribute

# HTML Boilerplate - Complete picture

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Website Title</title>
  <meta name="description" content="My new wonderful website">
  <meta name="author" content="Mister X">
  <link rel="stylesheet" href="./css/styles.css?v=1.0">
</head>

<body>
  <div>My Website</div>
  <!-- end of the body -->
  <script src="./js/scripts.js"></script>
</body>
</html>
```

# URL

Uniform Resource Locator

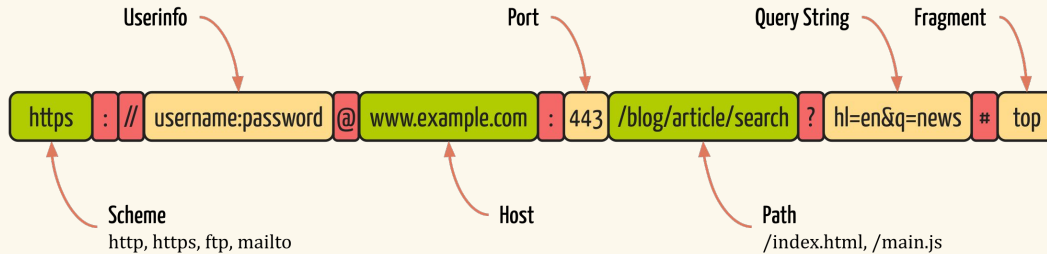
# Components of a URL

**Uniform Resource Locator:** an address for locating a unique resource on the net like a file or an app

## The components of a URL

### Main components

Some of the components shown here are simplified and some are optional



# URL example

`https://www.example.com:8080/path/to/resource?key=value#section`

- **scheme** (e.g., http, https)
- **host** (e.g., www.example.com)
- **port** (optional, e.g., :80, :443)
- **path** (optional, e.g., /path/to/resource)
- **query string** (optional, e.g., ?key=value)
- **fragment** (optional, e.g., #section)

# Resources

Inline, embedded & external



# Inline CSS

```
<body>  
  <p style="color:red;text-align:center;">Hello</p>  
  <p style="color:orange;text-align:center;">Nice to meet you</p>  
</body>
```

- no separation of concerns
- no reusability since it applies to a single element only
- limited caching, larger HTML file and slower load times
- no selectors or media queries
- hard to read and maintain code

**Never use inline CSS**

# Embedded CSS

```
<head>
  <style>
    p {
      color: red;
      text-align: center;
    }
  </style>
</head>
```

- no separation of concerns
- limited reusability; single HTML file only
- limited caching, larger HTML file and slower load times
- hard to read and maintain code

**Never use embedded CSS**

# External CSS

**index.html**

```
<head>  
  <link rel="stylesheet" href="./css/style.css">  
</head>
```

**style.css**

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

- good separation of concerns
- reusable and modular
- browser caching benefits and faster load times
- easy to maintain and collaborate

**Always use an external CSS**

# Inline Javascript

```
<body>  
  <button onclick="console.log('Hello, world!')">Click Me</button>  
</body>
```

- no separation of concerns
- no reusability since it applies to a single element only
- limited caching, larger HTML file and slower load times
- hard to read and maintain code

**Never use inline Javascript**

# Embedded Javascript

```
<head>  
  <script>  
    console.log('Hello, World!');  
  </script>  
</head>
```

- no separation of concerns
- limited reusability; single HTML file only
- limited caching, larger HTML file and slower load times
- hard to read and maintain code

**Never use embedded Javascript**

# External Javascript

index.html

```
<body>  
  <!-- end of the body -->  
  <script src="./js/main.js"></script>  
</body>
```

main.js

```
console.log('Hello, World!');
```

- good separation of concerns
- reusable and modular
- can manage script load order and dependencies
- browser caching benefits and faster load times
- easy to maintain and collaborate

**Always use an external Javascript**

# Structure & loading

speed optimization

# File and folder structure

```
<body>
  <!-- end of the body -->
  <script src="./js/main.js"></script>
</body>
```



```
<body>
  <!-- end of the body -->
  <script src="./myScripts/file.js"></script>
</body>
```





# Local vs remote Javascript

**Remote:**

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/normalize.css">
```

**Local:**

```
<link rel="stylesheet" href="./css/styles.css">
```

**Remote:**

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
```

**Local:**

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

# Script placement

```
<!doctype html>
<html lang="en">
<head>
  <script src="./js/earlyLoadingScript.js"></script>
</head>

<body>
  <h1>Introduction</h1>
  <p>Welcome to our service ... </p>
  <!-- end of the body -->
  <script src="./js/postDOMScript.js"></script>
</body>

</html>
```

**note:** the `<script>` element blocks the browser from proceeding with reading the remaining HTML content until the JavaScript code has been loaded and executed

# Async & defer

```
<script defer src="./js/first.js"></script>  
<script defer src="./js/second.js"></script>
```

## defer attribute:

- doesn't block browser
- script loads in background and executes after HTML parsing
- maintains script execution order so **first.js** runs before **second.js**




```
<script async src="./js/big.js"></script>  
<script async src="./js/small.js"></script>
```

## async attribute:

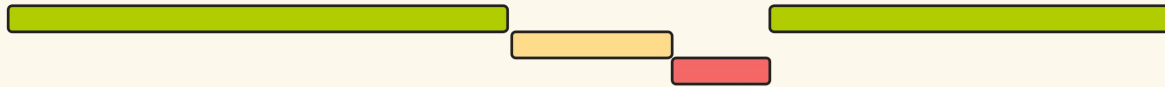
- doesn't block browser
- script loads asynchronously and executes as soon as it's ready
- no guarantee on script execution order so **small.js** might run before **big.js**

# Async & defer

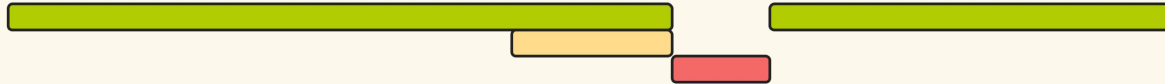
## Regular script vs defer vs async

html parsing   
js downloading   
js executing 

`<script>`



`<script async>`



`<script defer>`



© Shadi Lahham

# Compatibility

for older browsers

# HTML5 Shiv and Polyfills

```
<head>
  <!--[if lt IE 9]>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.js"></script>
  <![endif]-->
</head>
```

## **Polyfill:**

fallback code which makes modern functionality available in older browsers for compatibility  
Loading Polyfills is no longer a common practice

Specifically, the HTML5 shiv above is for older browsers that don't understand HTML5  
You don't need to use this on modern sites and apps

[Polyfill - MDN definition](#)

[What is a Polyfill?](#)

# Html Element with conditional comments

```
<!doctype html>
<!--[if lt IE 7]>      <html class="no-js lt-ie9 lt-ie8 lt-ie7"> <![endif]-->
<!--[if IE 7]>        <html class="no-js lt-ie9 lt-ie8"> <![endif]-->
<!--[if IE 8]>        <html class="no-js lt-ie9"> <![endif]-->
<!--[if gt IE 8]><!--> <html class="no-js">
</html>
```

You might see the above example in older code for compatibility reasons  
You don't need to use this on modern sites and apps

[Conditional comment - Wikipedia](#)

Your turn



# 1.Boilerplate

Quickly Read a few of the following pages

- [HTML5 Template](#)
- [Basic HTML5 Template](#)
- [Basic HTML boilerplate](#)

Using the information in this lesson and the pages above, write your own HTML boilerplate that you think is best. Name it index.html

Remember to test your file on the [The W3C Markup Validation Service](#)

Create a folder named **01-boilerplate** with your solution

## 2.New JS

Build your first Javascript project

- Write your index.html file from scratch
- Add a main.js file that writes your name to the console

Create a folder named **02-new-js** with your solution

*Note: all files should be in [kebab-case](#) ([italiano](#))*

[JavaScript Debugging](#)

[Console Overview | Tools for Web Developers](#)

# 3.The cache

Remember the line?

```
<link rel="stylesheet" href="./css/styles.css?v=1.0">
```

- What does `?v=1.0` do?
- How does the browser cache work?

Create a folder named **03-the-cache**

Inside the folder create a **.txt** or **.doc** or **.md** file with your answers

*Note: all files should be in kebab-case (italiano)*

# References

[HTML doctype declaration](#)

[HTML link](#)

[HTML meta](#)

Validate your code:

[The W3C Markup Validation Service](#)

Check browser compatibility:

[Can I use... Support tables for HTML5, CSS3, etc](#)

# References

URL components

[URL Syntax](#)

[Understanding the Components of a URL](#)

# © Copyright & Attribution

Unless otherwise stated, all materials are © 2017–2025 [Shadi Lahham](#)

For personal use only. May not be shared or reproduced without written permission

Brief excerpts may be used with proper attribution

External links and resources are copyrighted by their respective owners