INFS3202/7202 – Web Information Systems

Lecture Week 2: Creating and Deploying Web Applications

Dr Aneesha Bakharia (Senior Lecturer, EECS)
Email: a.bakharia1@uq.edu.au

Contents

Reminder - Self-paced Tutorials

Orientation Week (O Week)

Build Content | Assessments | Tools | Partner Content

Programming Background Survey

Please take a moment to fill out this short survey on your programming background. The data will be used to better customise the course material and practicals to your unique learning needs and programming background.

Take the Programming Background Survey

Self-Regulated Learning Questionnaire

Enabled: Statistics Tracking

The SRL-O questionnaire is a tool for you to check how well you're managing your studies and get feedback. It looks at your study habits, how motivated you are, and how you handle challenges while learning. By answering its questions, you'll get personalized tips on how to study better and make the most of this course.

Take the Self-Regulated Learning Questionnaire

Introduction to HTML Tutorial (Optional)

Attached Files: Introduction to HTML Tutorial (1.135 MB)
Solution (1.286 KB)

Introduction to CSS (Optional)

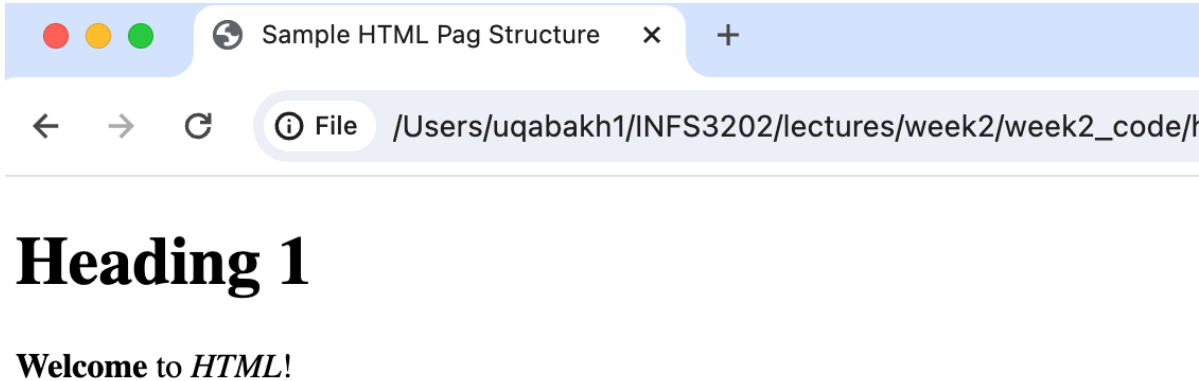
Attached Files: Introduction to CSS Tutorial (1.111 MB)
Solution (2.59 KB)

HTML Recap

Structure of HTML

example_html.html > ...

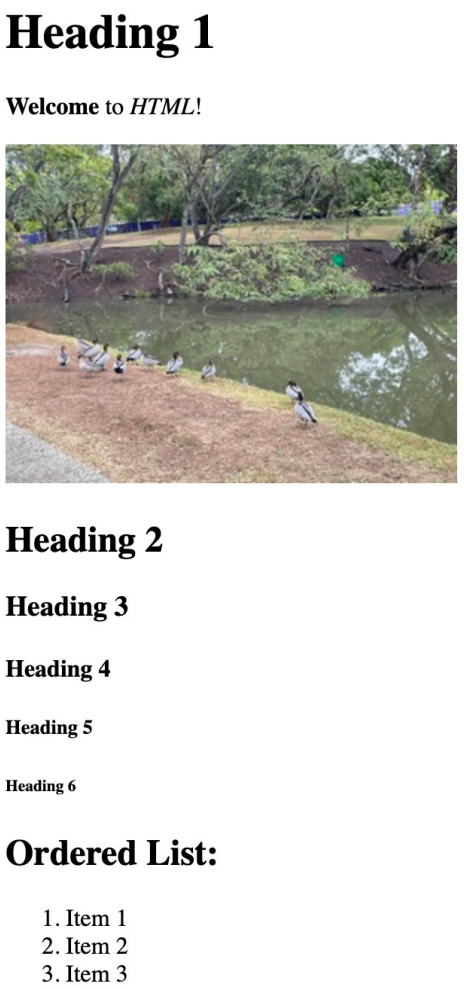
```
1   <!DOCTYPE html>
2   <html>
3   <head>
4     <title>Sample HTML Pag Structure</title>
5   </head>
6   <body>
7
8     <h1>Heading 1</h1>
9
10    <p><b>Welcome</b> to <i>HTML</i>!</p>
11
12  </body>
13  </html>
```



Code: week2_code/html_intro/example_html.html

HTML Common Tags

- Headings
- Paragraphs
- Lists (ordered and unordered)
- Images
- Tables

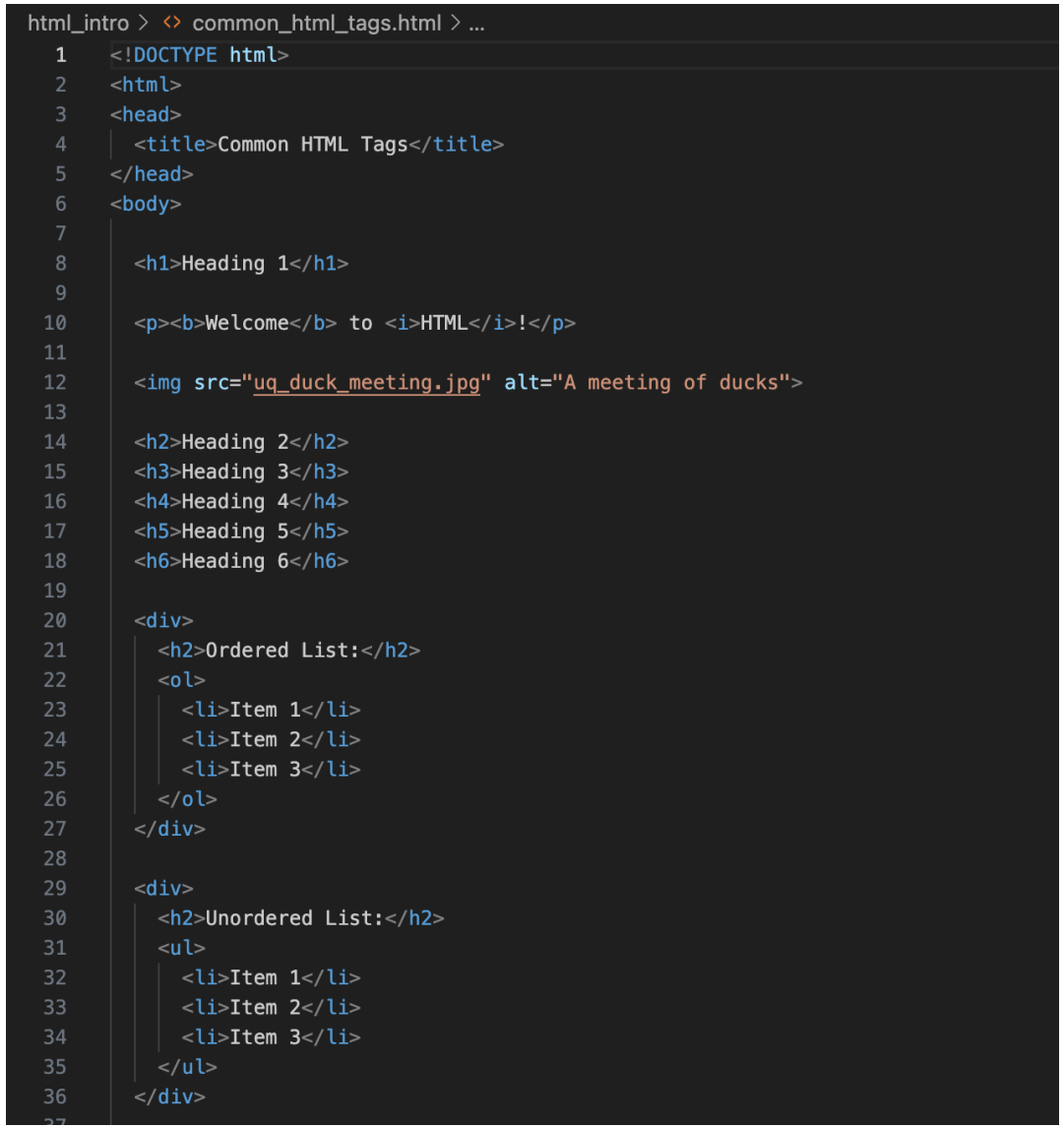


Code: week2_code/html_intro/common_html_tags.html

HTML Common Tags

html_intro > common_html_tags.html > ...

```
1   <!DOCTYPE html>
2   <html>
3   <head>
4     <title>Common HTML Tags</title>
5   </head>
6   <body>
7
8     <h1>Heading 1</h1>
9
10    <p><b>Welcome</b> to <i>HTML</i>!</p>
11
12    <img src="uq_duck_meeting.jpg" alt="A meeting of ducks">
13
14    <h2>Heading 2</h2>
15    <h3>Heading 3</h3>
16    <h4>Heading 4</h4>
17    <h5>Heading 5</h5>
18    <h6>Heading 6</h6>
19
20    <div>
21      <h2>Ordered List:</h2>
22      <ol>
23        <li>Item 1</li>
24        <li>Item 2</li>
25        <li>Item 3</li>
26      </ol>
27    </div>
28
29    <div>
30      <h2>Unordered List:</h2>
31      <ul>
32        <li>Item 1</li>
33        <li>Item 2</li>
34        <li>Item 3</li>
35      </ul>
36    </div>
```



Heading 1

Welcome to *HTML*!

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

Ordered List:

1. Item 1
2. Item 2
3. Item 3

Code: week2_code/html_intro/common_html_tags.html

HTML Common Tags

```
html_intro > common_html_tags.html > ...
 2  <html>
 6  <body>
37
38    <div>
39      <h2>Table of Harry Potter Books:</h2>
40      <table border="1">
41        <thead>
42          <tr>
43            <th>Title</th>
44            <th>Description</th>
45          </tr>
46        </thead>
47        <tbody>
48          <tr>
49            <td>Harry Potter and the Philosopher's Stone</td>
50            <td>Harry discovers he is a wizard and attends Hogwarts for the first time.</td>
51          </tr>
52          <tr>
53            <td>Harry Potter and the Chamber of Secrets</td>
54            <td>Harry returns to Hogwarts and investigates a series of mysterious petrifications.</td>
55          </tr>
56          <tr>
57            <td>Harry Potter and the Prisoner of Azkaban</td>
58            <td>Harry learns about his family's past and encounters a dangerous escaped prisoner.</td>
59          </tr>
60        </tbody>
61      </table>
62    </div>
63
64  </body>
65  </html>
66
67
```

Table of Harry Potter Books:

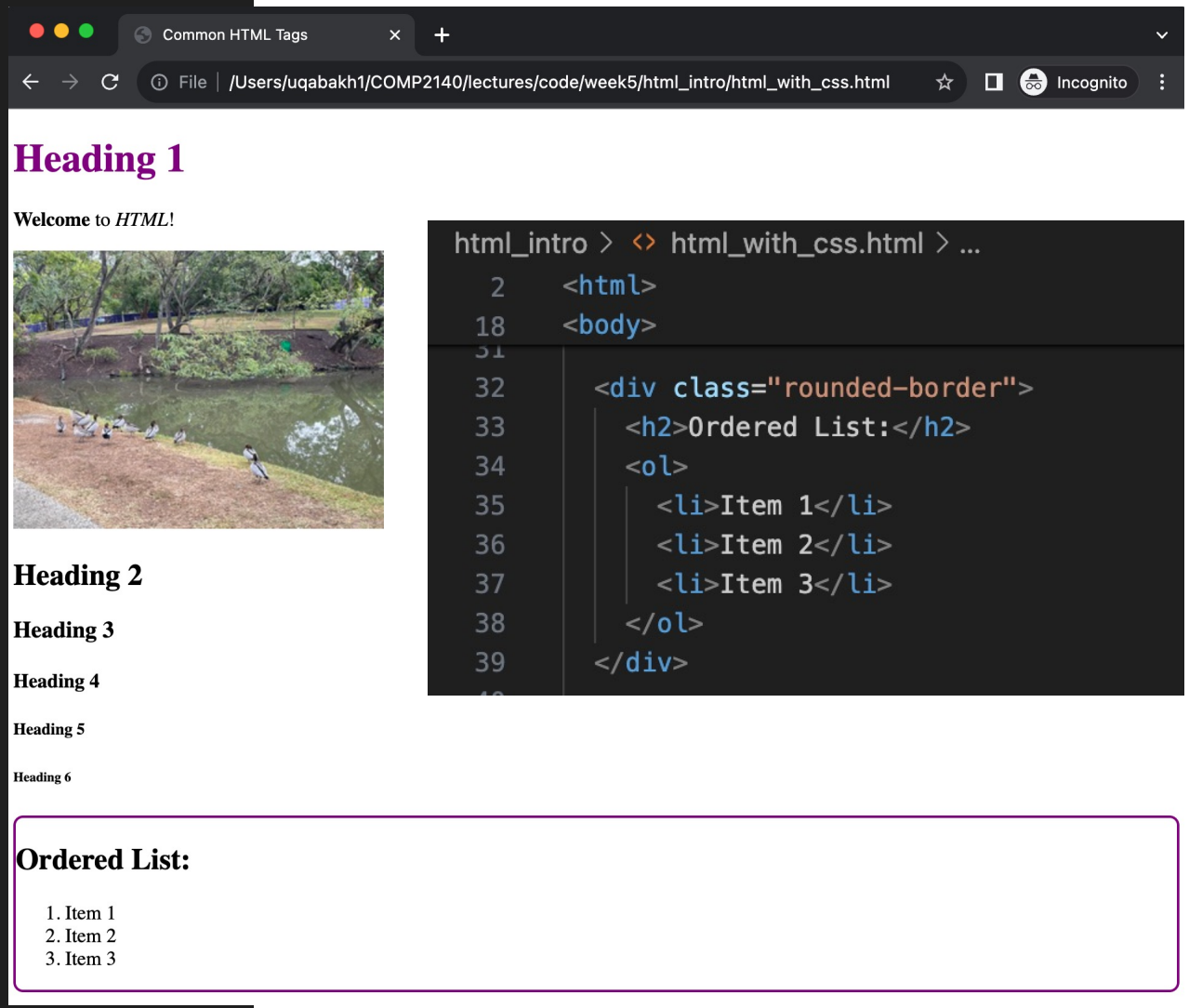
Title	Description
Harry Potter and the Philosopher's Stone	Harry discovers he is a wizard and attends Hogwarts for the first time.
Harry Potter and the Chamber of Secrets	Harry returns to Hogwarts and investigates a series of mysterious petrifications.
Harry Potter and the Prisoner of Azkaban	Harry learns about his family's past and encounters a dangerous escaped prisoner.

Code: week2_code/html_intro/common_html_tags.html

HTML with CSS

html_with_css.html > html

```
1   <!DOCTYPE html>
2   <html>
3
4   <head>
5     <title>Common HTML Tags</title>
6     <style>
7       h1 {
8         color: purple;
9       }
10
11      .rounded-border {
12        border: 2px solid purple;
13        border-radius: 8px;
14      }
15    </style>
16  </head>
17
18  <body>
19
20    <h1>Heading 1</h1>
```



html_intro > html_with_css.html > ...

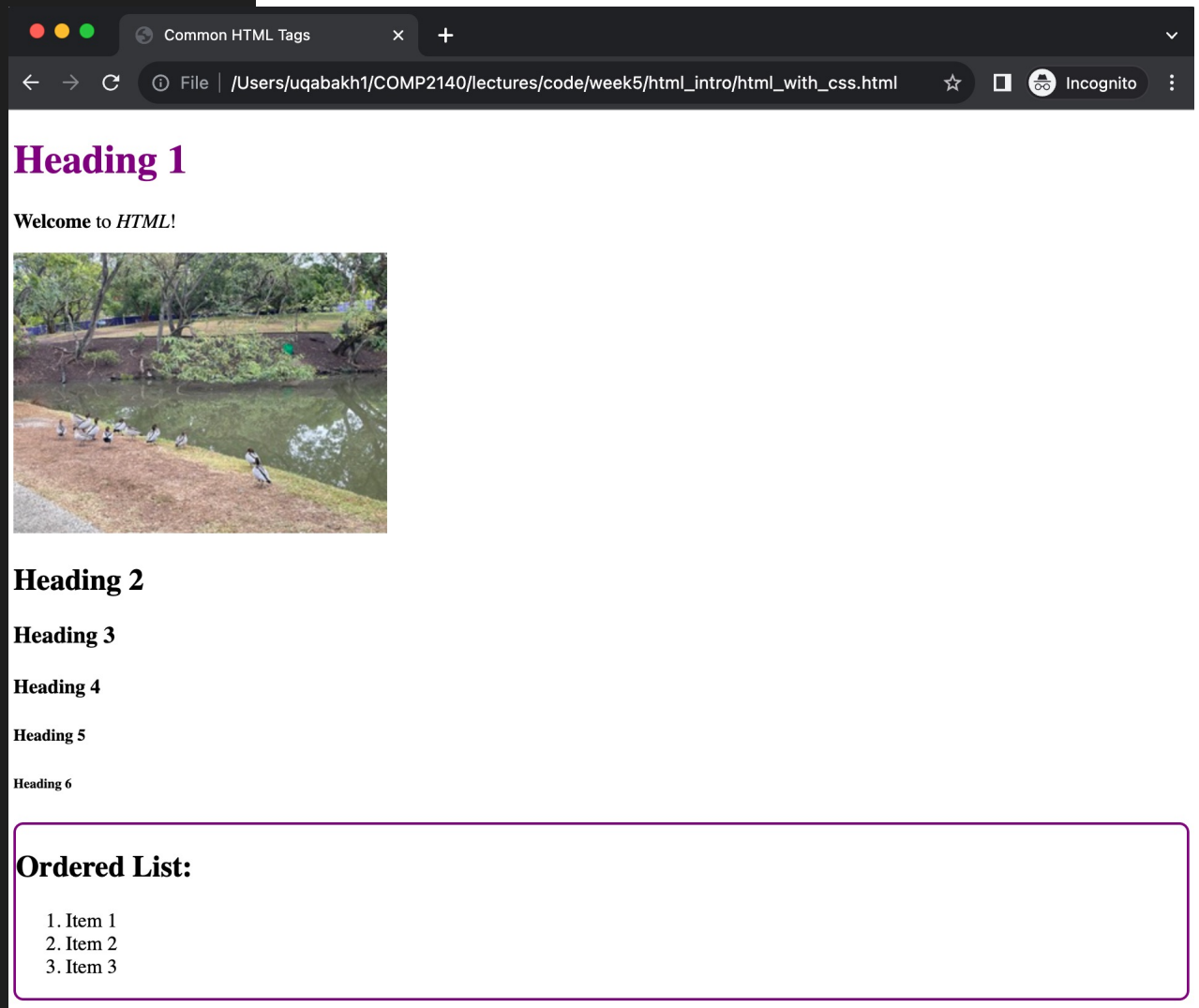
```
2   <html>
18  <body>
32    <div class="rounded-border">
33      <h2>Ordered List:</h2>
34      <ol>
35        <li>Item 1</li>
36        <li>Item 2</li>
37        <li>Item 3</li>
38      </ol>
39    </div>
```

Code: week5/html_intro/html_with_css.html

HTML with CSS

html_with_css.html > html

```
1   <!DOCTYPE html>
2   <html>
3
4   <head>
5     <title>Common HTML Tags</title>
6     <style>
7       h1 {
8         color: purple;
9       }
10
11      .rounded-border {
12        border: 2px solid purple;
13        border-radius: 8px;
14      }
15    </style>
16  </head>
17
18  <body>
19
20    <h1>Heading 1</h1>
```



Code: week5/html_intro/html_with_css.html

Cascading Style Sheets – Why Cascading?

- Cascading Style Sheets (CSS) refers to the priority scheme used to determine which style rules apply to elements in the web document if multiple rules conflict or overlap.
- This cascading mechanism allows multiple style sheets to influence the look of a document, with a well-defined order of precedence determining how conflicts are resolved.
- If two rules have the same specificity, the last rule declared in the CSS is applied.

```
/* External stylesheet */
body {
    color: red;
}

/* Embedded or inline stylesheet later in the document */
body {
    color: blue;
}
```

```
<body>
    This text will be blue because the blue style comes after the red in the cascade.
</body>
```

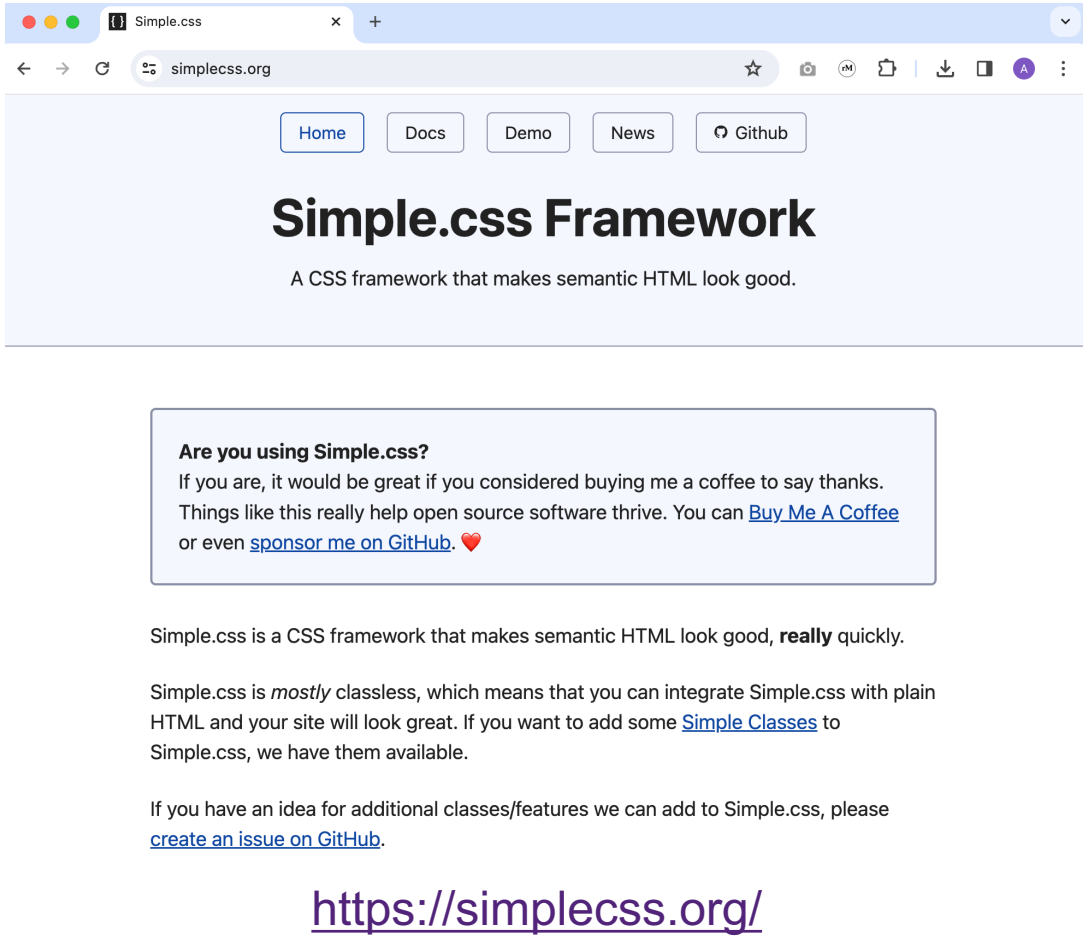
Classless CSS Frameworks – Simple.css

Advantages of Classless CSS Frameworks:

- **Simplicity and Speed**: Rapid development of websites by minimizing the need for writing custom CSS or applying specific classes to HTML elements.
- **Learning Curve**: Great for beginners because they don't require learning a system of classes.
- **Lightweight**: Due to their minimalist nature, classless frameworks tend to have a smaller file size.

Disadvantages:

- **Flexibility and Features**: Classless frameworks may not offer the same level of flexibility or component variety as more extensive frameworks like Bootstrap or Tailwind CSS.
- **Customization Limitations**: While customization is possible, extensive modifications might require adding more CSS, which could negate the benefits of using a minimalist framework.



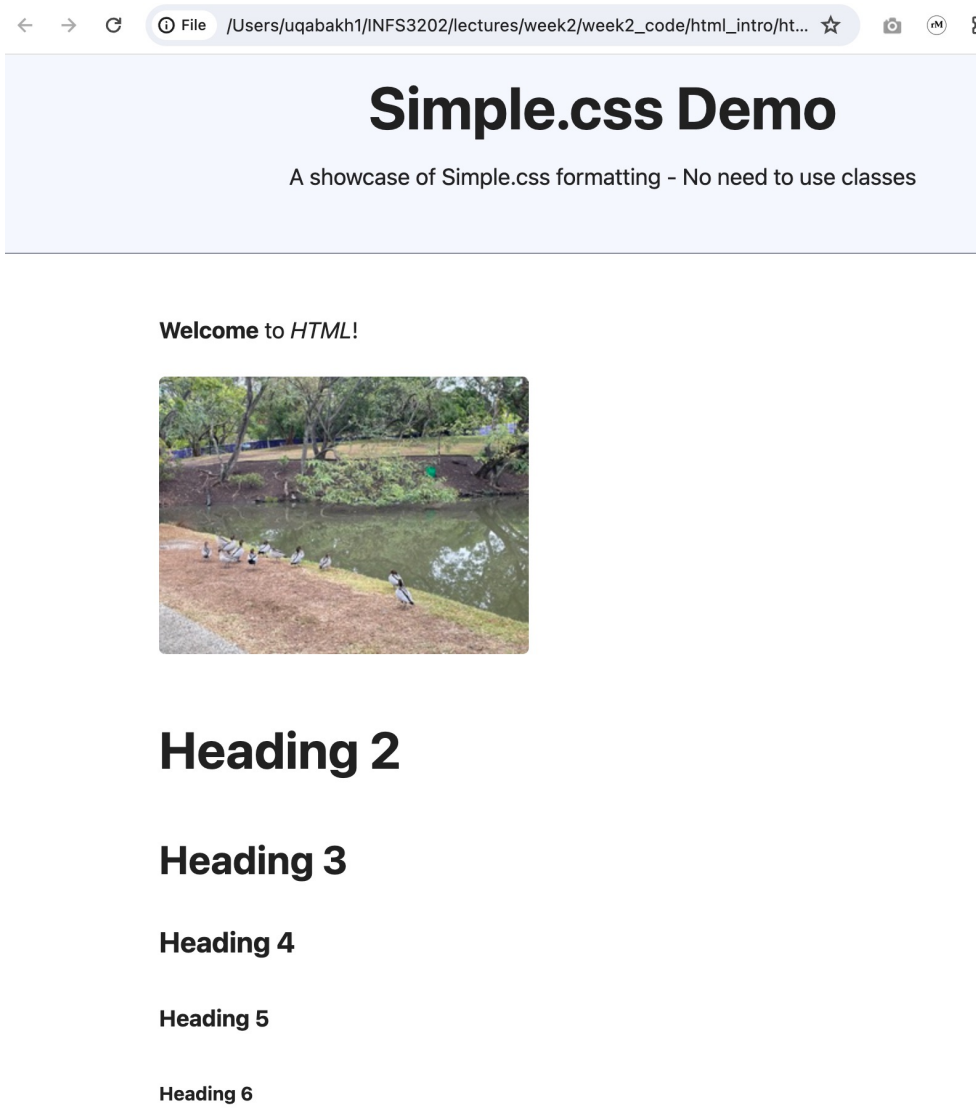
https://simplecss.org/

Classless CSS Frameworks – Simple.css

html_simplecss_classless_stylesheet.html

html_intro > html_simplecss_classless_stylesheet.html > html > body

```
1   <!DOCTYPE html>
2   <html>
3   <head>
4     <title>Common HTML Tags</title>
5     <link rel="stylesheet" href="https://cdn.simplecss.org/simple.min.css">
6   </head>
7   <body>
8     <header>
9       <h1>Simple.css Demo</h1>
10      <p>A showcase of Simple.css formatting - No need to use classes</p>
11    </header>
12
13    <main>
14      <p><b>Welcome</b> to <i>HTML</i>!</p>
15
16      <img src="uq_duck_meeting.jpg" alt="A meeting of ducks">
17
18      <h2>Heading 2</h2>
19      <h3>Heading 3</h3>
20      <h4>Heading 4</h4>
21      <h5>Heading 5</h5>
22      <h6>Heading 6</h6>
23
```



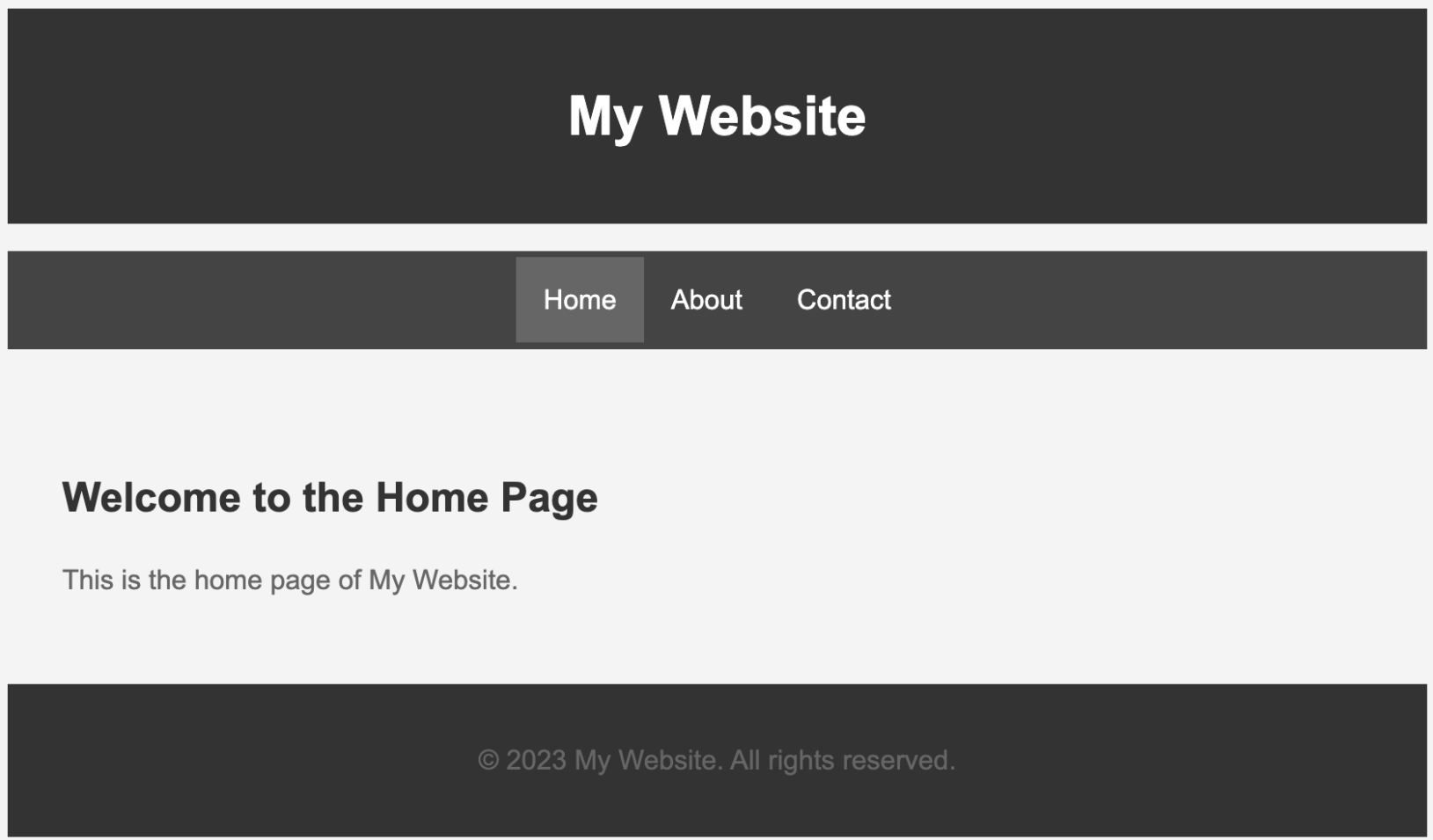
Code: week2_code/html_intro/html_simplecss_classless_stylesheet.html

Links - The <a href> tag

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home - My Website</title>
</head>

<body>
    <header>
        <h1>My Website</h1>
    </header>

    <nav>
        <ul>
            <li><a href="index.html" class="active">Home</a></li>
            <li><a href="about.html">About</a></li>
            <li><a href="contact.html">Contact</a></li>
        </ul>
    </nav>

    <main>
        <h2>Welcome to the Home Page</h2>
        <p>This is the home page of My Website.</p>
    </main>
```

- The <a> tag, also known as the anchor tag, is used in HTML to create hyperlinks that navigate to other web pages or resources.
- The href attribute within the <a> tag specifies the destination URL that the link points to.
- The href value can be an absolute URL, a relative URL, or a URL fragment (used for scrolling to specific sections within the same page).
- **Absolute URL**: A complete web address, including the protocol (e.g., http://, https://), domain, and path.
  Example: Visit
- **Relative URL**: A partial web address that is relative to the current page's location. Relative URLs are often used for links within the same website.
  Example: Learn More
- The target attribute can be used to control how the linked content is displayed when clicked eg _blank opens the link in a new tab or window.
- The **<a>** tag can be used to create text links, image links, or any content that can be clicked to navigate.

A simple HTML Website



- Uses a custom external stylesheet
- Has a nav bar with links

Code: week2_code/html_simple_website/index.html

A simple HTML Website

html_simple_website > index.html > ...

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Home - My Website</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <header>
    <h1>My Website</h1>
  </header>

  <nav>
    <ul>
      <li><a href="index.html" class="active">Home</a></li>
      <li><a href="about.html">About</a></li>
      <li><a href="contact.html">Contact</a></li>
    </ul>
  </nav>

  <main>
    <h2>Welcome to the Home Page</h2>
    <p>This is the home page of My Website.</p>
  </main>

  <footer>
    <p>&copy; 2023 My Website. All rights reserved.</p>
  </footer>
</body>
</html>
```

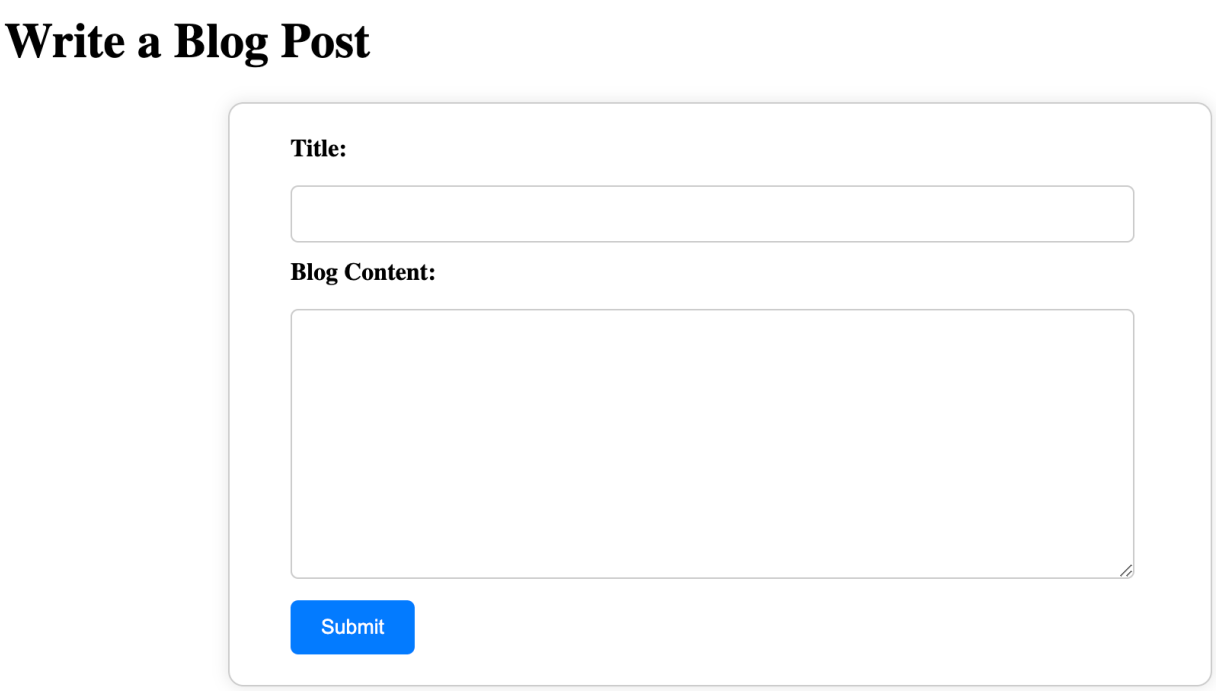
- Uses a custom external stylesheet
- Has a nav bar with links

Code: week2_code/html_simple_website/index.html

A Simple Form

simple_form.html > html

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Blog Post Form</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <h1>Write a Blog Post</h1>

  <form action="/submit-blog" method="post">
    <div>
      <label for="title">Title:</label>
      <input type="text" id="title" name="title" required>
    </div>
    <div>
      <label for="blog">Blog Content:</label>
      <textarea id="blog" name="blog" rows="10" cols="50" required></textarea>
    </div>
    <div>
      <button type="submit">Submit</button>
    </div>
  </form>

</body>
</html>
```



Code: week2_code/html_forms_recap/simple_form.html

A Simple Form

simple_form.html > html

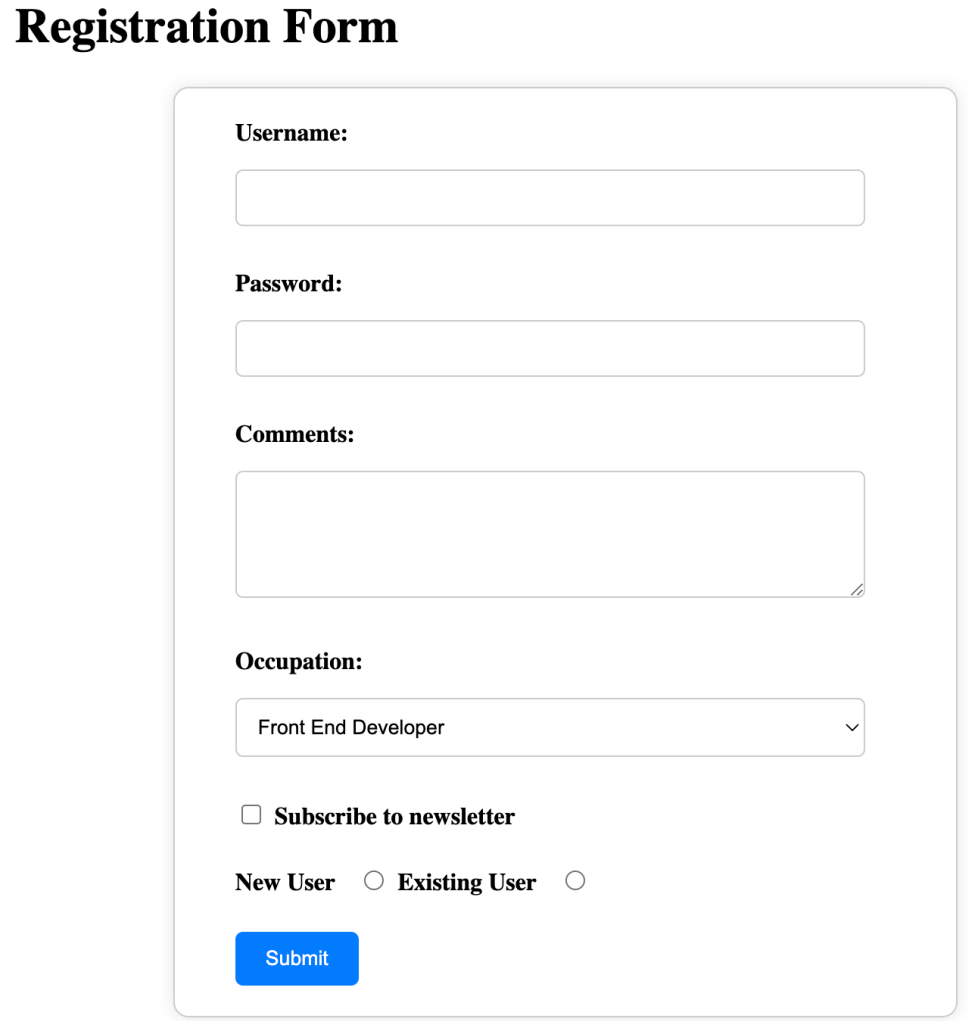
```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Blog Post Form</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <h1>Write a Blog Post</h1>

  <form action="/submit-blog" method="post">
    <div>
      <label for="title">Title:</label>
      <input type="text" id="title" name="title" required>
    </div>
    <div>
      <label for="blog">Blog Content:</label>
      <textarea id="blog" name="blog" rows="10" cols="50" required></textarea>
    </div>
    <div>
      <button type="submit">Submit</button>
    </div>
  </form>

</body>
</html>
```

- Each of elements can be included within a **<form>** element
- Use the **action** and **method** attributes of the **<form>** element to specify where and how the form data should be submitted
- **<input>** is a single line text field
- **<textarea>** is a multiline text field
- **<button>** is a button and with the type attribute set to ‘submit’ will send the values entered into the form to the action path of the form

Code: week2_code/html_forms_recap/simple_form.html

A Registration Form

```
28    <!-- Select Dropdown -->
29    <label for="occupation">Occupation:</label>
30    <select id="occupation" name="occupation">
31      <option value="frontEndDeveloper">Front End Developer</option>
32      <option value="backEndDeveloper">Back End Developer</option>
33      <option value="fullStackDeveloper">Full Stack Developer</option>
34      <option value="dataScientist">Data Scientist</option>
35      <option value="designer">Designer</option>
36    </select>
37    <br><br>
38
39    <!-- Checkbox -->
40    <input type="checkbox" id="subscribe" name="subscribe" value="yes">
41    <label for="subscribe">Subscribe to newsletter</label>
42    <br><br>
43
44    <!-- Radio Buttons -->
45    <label for="new">New User</label>
46    <input type="radio" id="new" name="userType" value="new">
47
48    <label for="existing">Existing User</label>
49    <input type="radio" id="existing" name="userType" value="existing">
50    <br><br>
51
52    <!-- Submit Button -->
53    <button type="submit">Submit</button>
```



Code: week7/html_forms_recap/registration_form.html

Forms – method Get vs Post

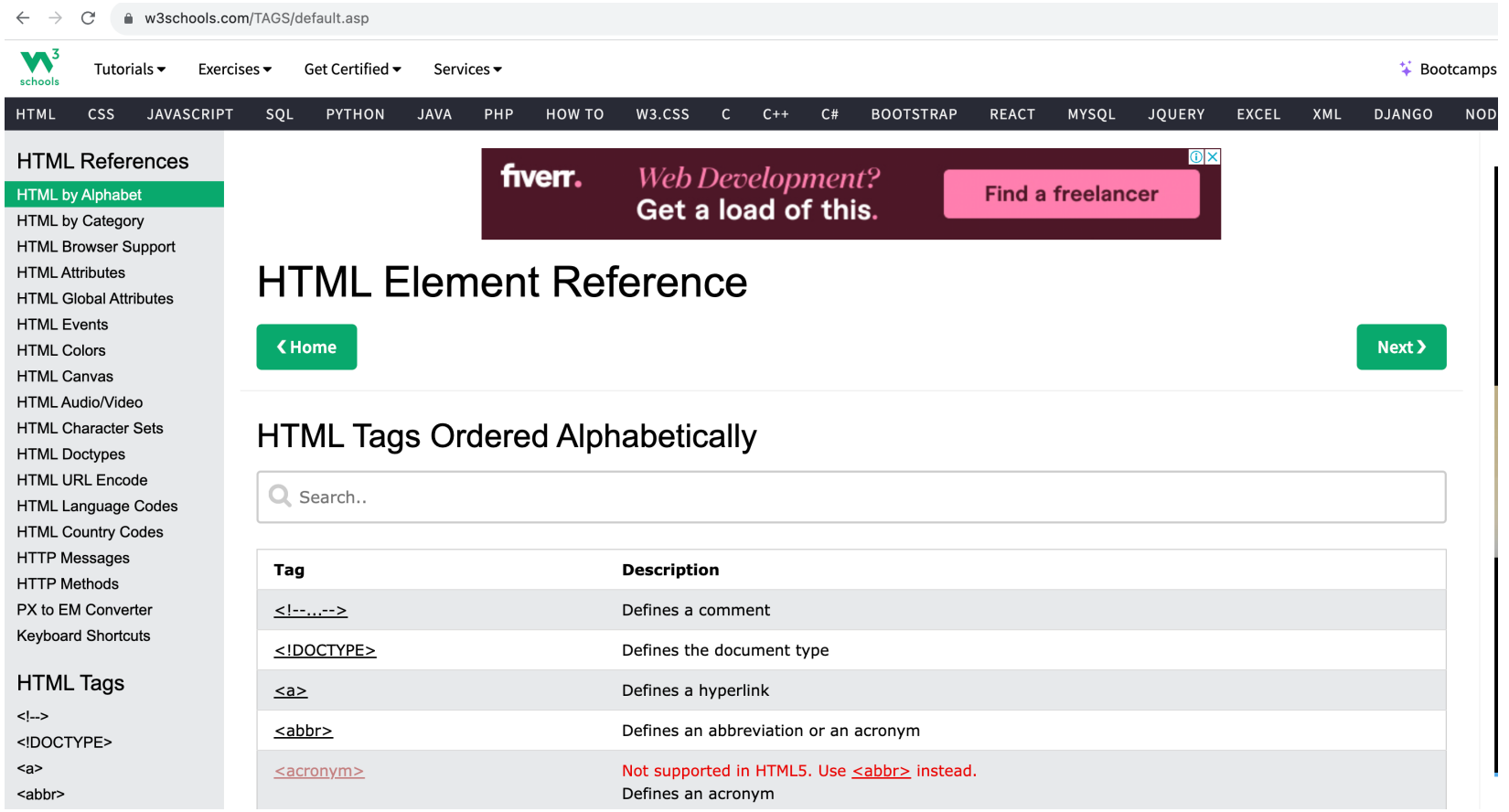
GET Method

- **Data Appended to URL**: Sends data in the URL itself, as key-value pairs in the query string.
- **Read Operations**: Typically used for retrieving data, not modifying it.
- **Limited Data Size**: Length restrictions due to URL length limitations (usually up to 2048 characters).

POST Method

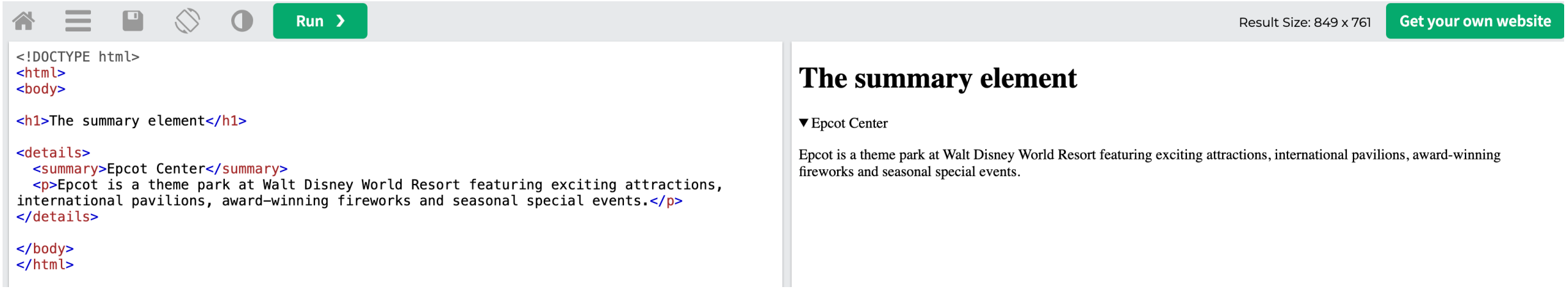
- **Data in Body**: Sends data in the request body, allowing for larger and more complex payloads. No inherent size limitation like with GET.
- **Write Operations**: Typically used for sending data to be processed or stored on the server.
- **Encoding**: GET uses URL encoding, whereas POST allows for multiple data types (e.g., **application/json**, **multipart/form-data**).

W3Schools - Reference



https://www.w3schools.com/TAGS/default.asp

Exploring HTML Tags – Detail & Summary



- The **<details>** and **<summary>** tags are HTML elements used together to create an interactive widget that users can click to hide or show more content.
- This feature is particularly useful for implementing FAQs, collapsible lists, or any content that benefits from a hide/show interaction to keep the page clean and improve user experience by not overwhelming them with information all at once.

Exploring HTML Tags – Ruby and RT

- The **<ruby>** & **<rt>** tags are part of HTML5 and are used to provide ruby annotations.
- Ruby annotations are small texts that appear above or next to the base text, primarily used in East Asian typography to show the pronunciation or to provide a short annotation of the base text.
- These tags are particularly useful for languages like Japanese, Chinese, and Korean, where they can help readers understand the text's pronunciation.

PHP

PHP – Hypertext Preprocessor

What is PHP?

- PHP: Hypertext Preprocessor
- Server-side scripting language designed for web development
- Can be embedded into HTML

Why Use PHP?

- Widely used and open-source
- Efficient for building dynamic web pages
- Compatible with various databases
- Large community and extensive documentation

PHP – Basic Syntax

- Starts with **<?php** and ends with **?>**
- PHP statements end with a semicolon (**;**)

- A PHP file normally contains **HTML tags**, and some **PHP scripting codes**.
- The file extension is ".php"

```php
<?php
echo "Hello, World!";
?>
```

Note:

In PHP, we use
// to make a one-line comment
or to make a comment block
/*
and
*/

PHP – History

Rasmus Lerdorf

PHP

- is a general-purpose scripting language for web development.
- was originally created by Rasmus Lerdorf in 1993 and released in 1995.
- was originally an abbreviation of Personal Home Page, but now it is PHP: Hypertext Preprocessor.

The standard PHP interpreter, powered by the Zend Engine, is free software.

Andi Gutmans

Zeev Suraski

W3Techs reports:

- "PHP is used by 77.8% of all the websites.” as of January 2023;
- Only 8% of PHP users use the currently supported 8.x versions;
- Most use unsupported PHP 7, more specifically 7.4,
- Even PHP 5 has 23% of the use (not supported with security updates, known to have serious security vulnerabilities).

PHP 7 (unsupported, not safe) vs PHP 8 (faster, safer, new features)

https://en.wikipedia.org/wiki/PHP
https://www.php.net/manual/en/migration80.new-features.php

PHP – Variables

- **Variables are "containers" for storing information.**
- **Rules for PHP variable names:**
- Start with **$** (e.g., $name = “Aneesha”)
- Must **begin** with a letter or the underscore character
- Can only contain **alpha-numeric** characters and underscores
- Should not contain spaces
- **Case sensitive** ($Name ≠ $name)
- PHP supports several data types:
  Strings, Integers, Floats, Booleans, Arrays, etc.

```php
<?php
$text = "Learning PHP"; // String
$number = 7; // Integer
$price = 10.5; // Float
?>
```

```php
<?php
$age = array("Peter" => "35", "Ben" => "37", "Joe" => "43");
echo "Ben is " . $age['Ben'] . " years old.";
?>
```

PHP – String Concatenation

- String concatenation in PHP is the process of joining two or more strings together into one single string.
- PHP uses the dot (.) operator for string concatenation.
- Useful generating HTML content with embedded PHP variables, combining user input into messages, or assembling file paths and URLs from separate parts.

```php
$greeting = "Hello, ";
$name = "John";
$message = $greeting . $name;
echo $message; // Outputs: Hello, John
```

PHP does not use + to join strings together.

PHP – Control Structures – If statements

if Statement

- The if statement is used to execute a block of code if a specified condition is true.
- It can be combined with an else statement to execute a code block if the condition is false.
- You can also use elseif to specify a new condition to test if the first condition is false.

```php
<?php
$age = 20;
if ($age >= 18) {
    echo "You are an adult.";
}
?>
```

```php
$age = 20;

if ($age >= 18) {
    echo "You are an adult.";
} elseif ($age >= 13) {
    echo "You are a teenager.";
} else {
    echo "You are a child.";
}
```

PHP – Control Structures – Loops

- PHP supports while, for and foreach loops

```php
$counter = 0;
while ($counter < 10) {
    echo $counter;
    $counter++;
}
```

```php
<?php
// Define an associative array
$fruits = array("apple" => "Green", "banana" => "Yellow", "cherry" => "Red");

// Start the unordered list
echo "<ul>";

// Loop through the array and render each item as a list item
foreach ($fruits as $fruit => $color) {
    echo "<li>$fruit is $color</li>";
}

// End the unordered list
echo "</ul>";
?>
```

```php
for ($counter = 0; $counter < 10; $counter++) {
    echo $counter;
}
```

PHP – Operators

Operator	Name	Description
x + y	Addition	Sum of x and y
x - y	Subtraction	Difference of x and y
x * y	Multiplication	Product of x and y
x / y	Division	Quotient of x and y
x % y	Modulus	Remainder of x divided by y
- x	Negation	Opposite of x
a . b	Concatenation	Concatenate two strings

Assignment	Same as...
x = y	x = y
x += y	x = x + y
x -= y	x = x - y
x *= y	x = x * y
x /= y	x = x / y
x %= y	x = x % y
a .= b	a = a . b

Operator	Name	Description
x == y	Equal	True if x is equal to y
x === y	Identical	True if x is equal to y, and they are of same type
x != y	Not equal	True if x is not equal to y
x <> y	Not equal	True if x is not equal to y
x !== y	Not identical	True if x is not equal to y, or they are not of same type
x > y	Greater than	True if x is greater than y
x < y	Less than	True if x is less than y
x >= y	Greater than or equal to	True if x is greater than or equal to y
x <= y	Less than or equal to	True if x is less than or equal to y

Operator	Name
x and y	And
x or y	Or
x xor y	Xor
x && y	And
x || y	Or
! x	Not

https://www.w3schools.com/php/php_operators.asp

PHP – Working with Forms

```
<!-- HTML form -->
<form action="welcome.php" method="post">
Name: <input type="text" name="name">
<input type="submit">
</form>

<?php
// PHP script in welcome.php
echo "Welcome " . $_POST['name'];
?>
```

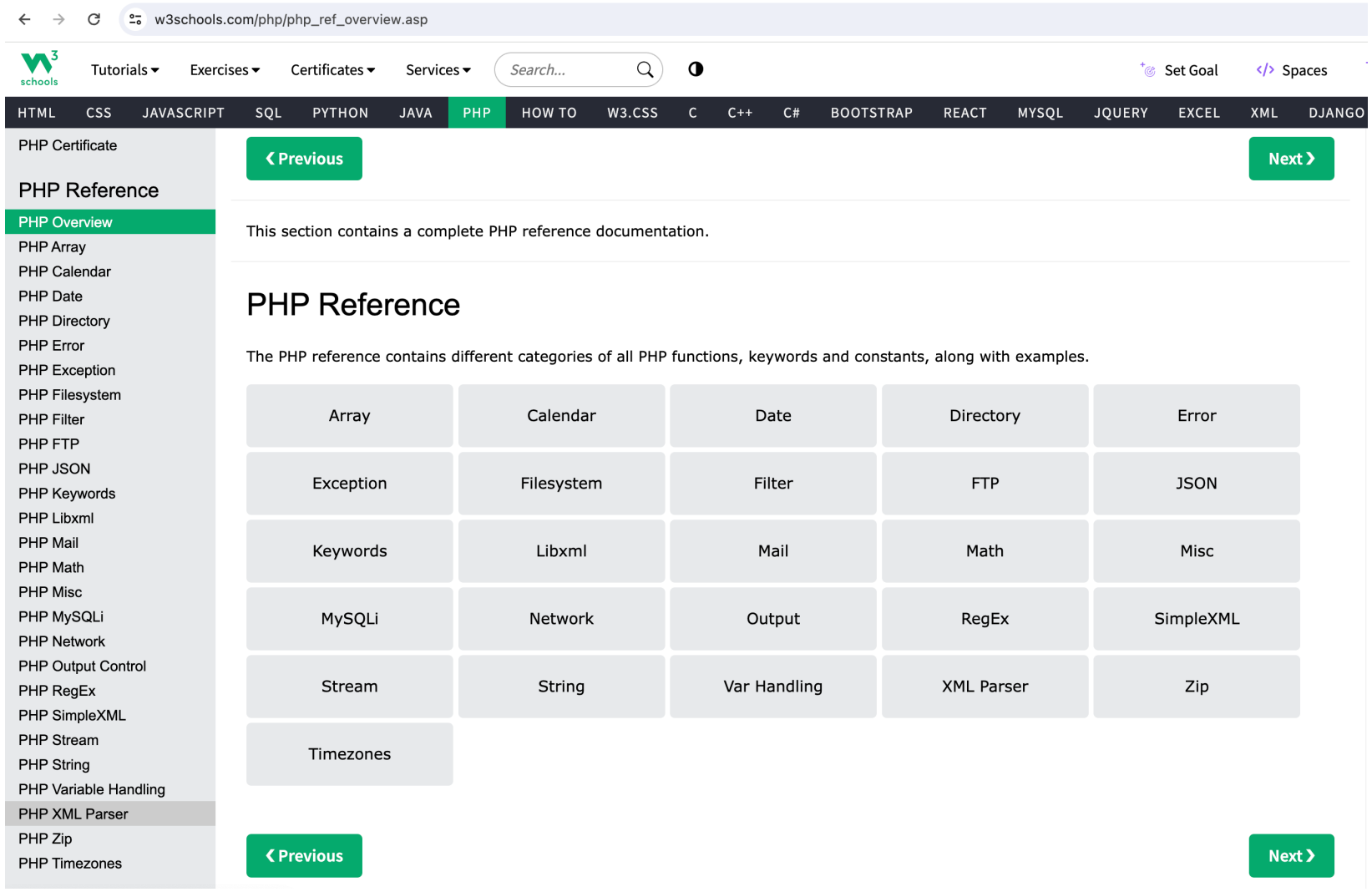
Getting data from a form using $_GET or $_POST

PHP – Built-in SuperGlobals

- **Superglobal variables in are predefined variables that are accessible from anywhere in a PHP script.**
- **Superglobal variables are automatically available in all scopes, including functions and methods,**
- **Superglobal variables provide a simple way to access common values or data structures,**
  - e.g. HTTP headers, form data, session data, server environment variables, and more.
- **Superglobal variables examples:.**
  - $GLOBALS
  - $_SERVER
  - $_GET
  - $_POST
  - $_FILES
  - $_COOKIE
  - $_SESSION
  - $_REQUEST
  - $_ENV

http://au2.php.net/manual/en/language.variables.superglobals.php

PHP – Built-in Functions

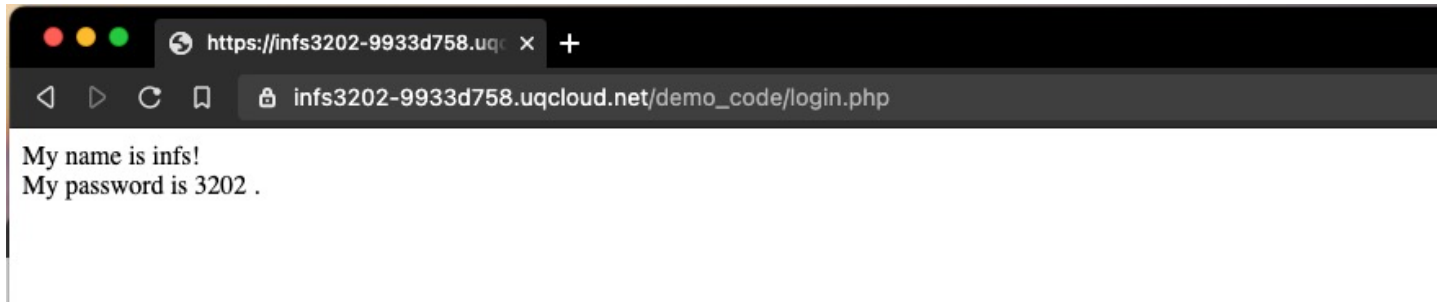


https://www.w3schools.com/php/php_ref_overview.asp

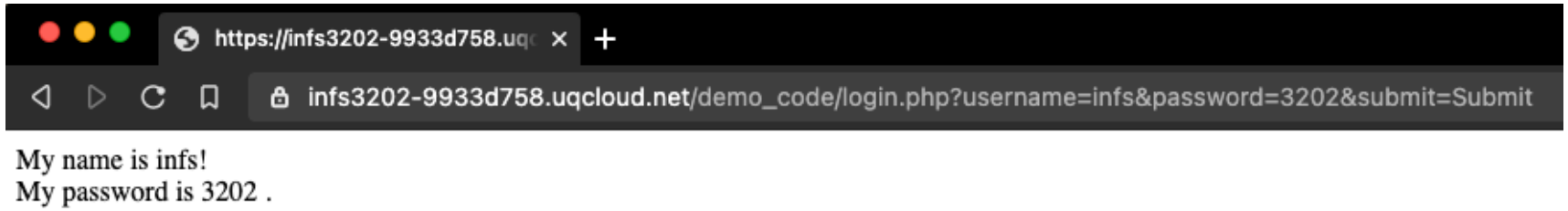
PHP – Forms – Get vs Post

- **Differences between _GET and _POST**

- _POST



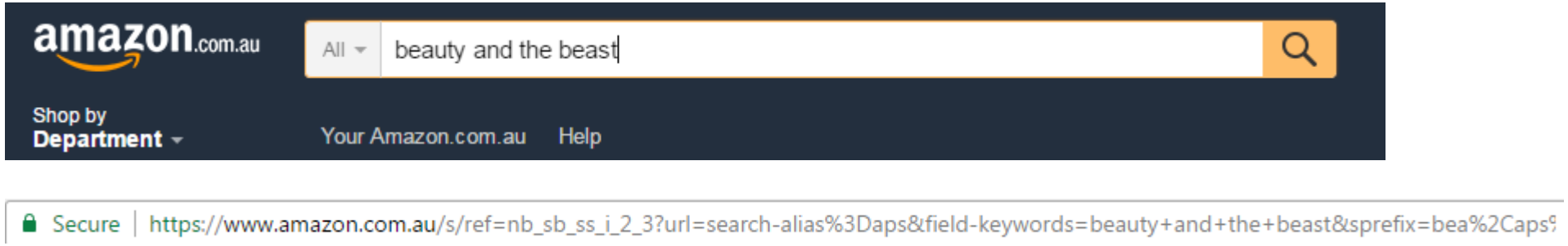
- _GET



Information sent from a form with the GET method is visible to everyone (it will be displayed in the browser's address bar) and has limits on the amount of information to send.

PHP – When to use Get

- **When to use the method="get"**
- This method should not be used when sending passwords or other sensitive information!
- This method is not suitable for very large variable values.
- This method is case insensitive (all characters in lower case).
- This **can be useful** for example where you want to be able to **bookmark** a page with specific query string values.



PHP – Functions

- **A function will be executed by a call to the function.**
- The PHP script defines a function **showName** that takes two parameters, **$given_name** and **$family_name**, and echoes them with a space in between and an exclamation mark, followed by a break (**
**).

```
1   <html>
2   <body>
3
4   <?php
5   function showName($given_name, $family_name)
6   {
7       echo $given_name . " " . $family_name . "!</br>";
8   }
9
10  echo "His name is ";
11  showName("Jack", "Black");
12  ?>
13
14  </body>
15  </html>
16
```

PHP – Classes

- Classes are the blueprint for creating objects.
- They encapsulate data for the object and methods to manipulate that data.
- Using classes and objects allows you to implement the principles of object-oriented programming (OOP).

```php
class Car {
    // Properties
    public $color;

    // Methods
    function setColor($color) {
        $this->color = $color;
    }

    function getColor() {
        return $this->color;
    }
}
```

PHP – Classes - Properties

- A class is defined using the **class** keyword followed by the name of the class and a pair of curly braces **{}** that contain the properties and methods of the class.
- Class names should start with a capital letter by convention.
- **Properties**
  - Properties (also known as attributes or fields) are variables inside a class. In the example above, $color is a property of the Car class.
  - Properties define the data that objects will store. Visibility keywords (public, protected, private) define the scope of the properties.

```php
class Car {
    // Properties
    public $color;

    // Methods
    function setColor($color) {
        $this->color = $color;
    }

    function getColor() {
        return $this->color;
    }
}
```

PHP – Classes - Methods

- **Methods**
  - Methods are functions defined inside a class.
  - They are used to set, get, or manipulate the object's properties.
  - In the Car class example, setColor and getColor are methods.
  - Methods can also have visibility keywords (public, protected, private) to define the scope of the properties.

```php
class Car {
    // Properties
    public $color;

    // Methods
    function setColor($color) {
        $this->color = $color;
    }

    function getColor() {
        return $this->color;
    }
}
```

PHP – Classes - Visibility

- **public:**
  The property or method can be accessed from anywhere.
- **protected:**
  The property or method can be accessed within the class and by classes derived from that class.
- **private:**
  The property or method can ONLY be accessed within the class.

```php
class Car {
    // Properties
    public $color;

    // Methods
    function setColor($color) {
        $this->color = $color;
    }

    function getColor() {
        return $this->color;
    }
}
```

PHP – Classes - Visibility

Creating Objects

To create an instance of a class (i.e., an object), use the new keyword followed by the class name:

Accessing Properties and Methods

Once you have an object, you can access its properties and methods using the arrow operator ->.

Here's how to use the **setColor** and **getColor** methods:

```
3   $myCar = new Car();
4
5   $myCar->setColor("blue");
6   echo $myCar->getColor(); // Outputs: blue
7
```

PHP – Classes – Constructor Method

- A special method called a constructor (**__construct()**) can be defined in a class.
- It is automatically called when an object is created.
- Constructors are typically used to initialize properties or perform setup tasks.

```php
class Car {
    public $color;

    function __construct($color) {
        $this->color = $color;
    }

    function getColor() {
        return $this->color;
    }
}

$myCar = new Car("red");
echo $myCar->getColor(); // Outputs: red
```

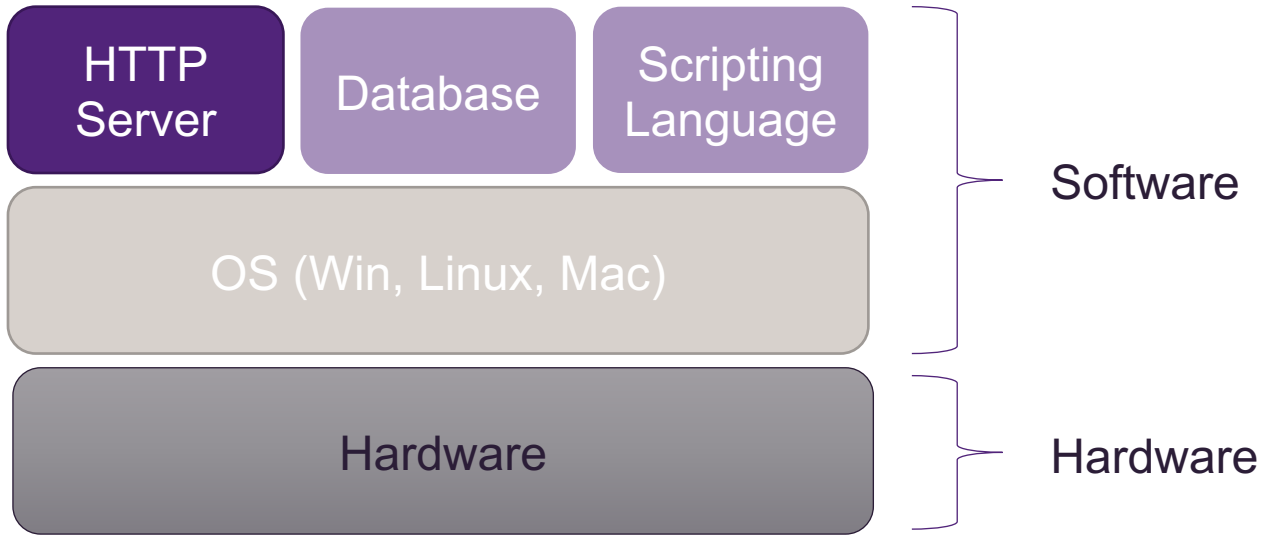
UQCloud & NGINX Web Server

Elements of a Web Server

The basic elements of a web server includes

- **Hardware**
- **operating system**
- **http server**

The addition of a database and scripting language extend a server's capabilities

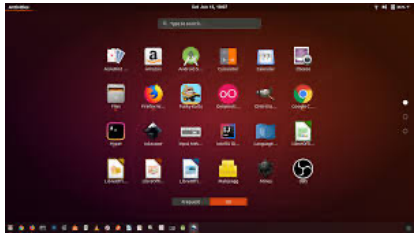


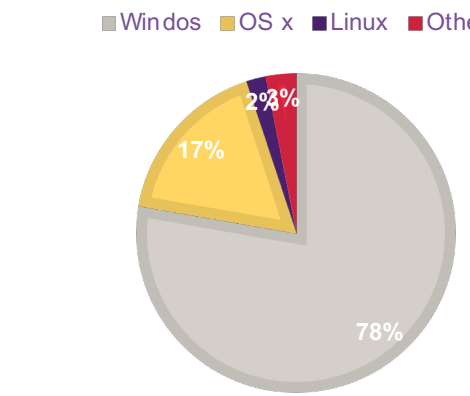
Operating System - Linux

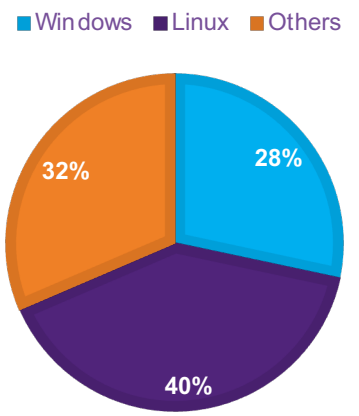
A operating system is what allows you to interact with the applications and hardware that make up your computer. It facilitates resource allocation to your applications, and communication between hardware and software.

Typically, operating systems for servers fall under three categories:

- Linux based
- Windows based
- and Mac based





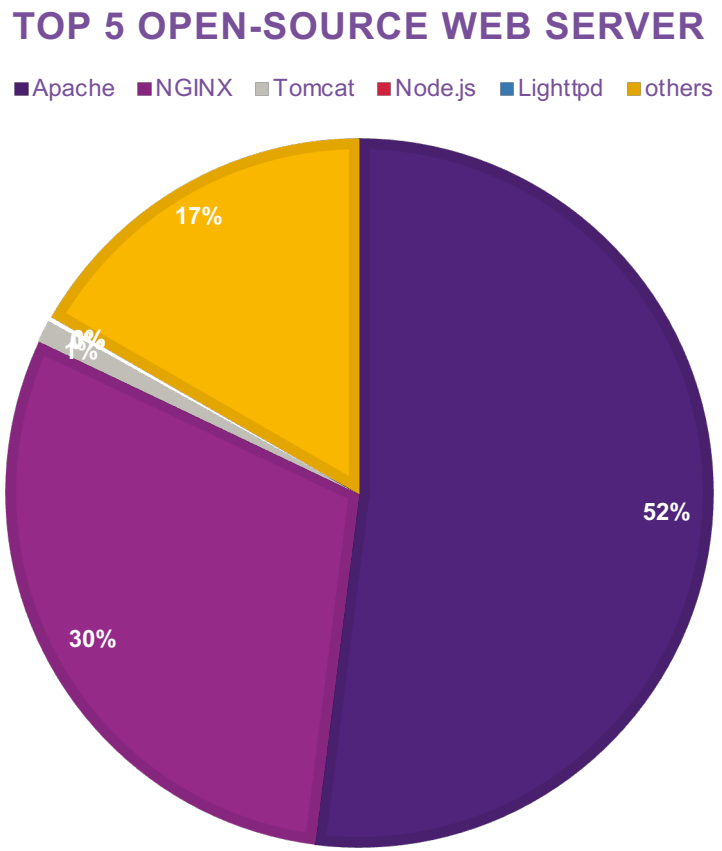


https://gs.statcounter.com/os-market-share/desktop/worldwide

Web Server software

Top five open source web server

- Apache HTTP Server
- NGINX
- Apache Tomcat
- Node.js
- Lighttpd



UQCloud

INFS3201/7202 Lab 1: UQCloud, HTML and PHP

What is UQCloud?

Welcome to this lab on accessing and setting up your UQCloud zone. UQCloud Zones is a powerful cloud service that offers personal Virtual Machines (VMs) for students, providing a flexible and customizable development and deployment environment.

UQCloud Zones is a service that provides you with a personal Virtual Machine (VM) running a Linux operating system. This environment is particularly useful for developers, researchers, or students who require a separate instance for their projects or experiments.

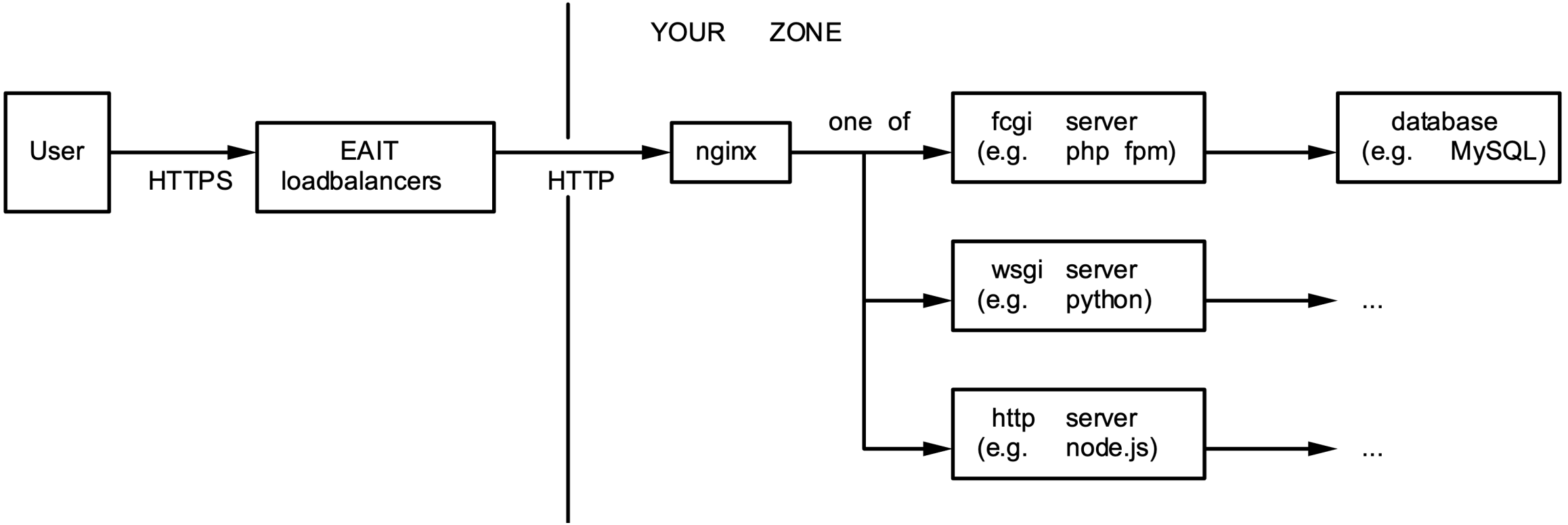
> ⓘ **Note**
>
> Zones are a specific kind of virtualisation technology. Virtualisation is the process of making one computer "pretend" to be multiple (usually smaller) computers—in the case of UQcloud, we do this in order to allow large servers to be broken up into small units for hosting websites and applications which do not need all of the available capacity on a large server. A zone functions as though it was a small computer-controlled completely by you (you get "root" access), which you can use to run programs or store data. It has a permanent network connection, protected power circuits with battery and generator backup, and redundancy to guard against disk failure and other problems. It is very similar in concept to a Virtual Private Server (VPS), cloud instance, or Virtual Machine (VM) hosted by a commercial hosting provider like Amazon AWS. One of the key advantages of a zone is its position within the UQ network, which allows it to make easy use of facilities like UQ authentication or the ability to communicate with other internal devices and systems inside the University. A completed guide can be found here and you are recommended to read it through carefully as many other UQ courses are using it.

Key Features of UQCloud Zones:

- **Personal Virtual Machine**: You get a dedicated VM to work on, which means you can configure it to your liking, install software, and run services without affecting others.
- **Nginx Webserver**: By default, the VM comes with the Nginx webserver installed. Nginx is known for its high performance, stability, and rich feature set.
- **Service Management**: UQCloud Zones allows you to turn on various services as per your requirement, such as PHP, MySQL, etc., making it a versatile platform for web development.

UQCloud and Student Zones

- UQCloud gives each student a Zone, which is a server that runs NGINX where various services can be enabled eg PHP, databases, etc…



Connecting to your Student Zone

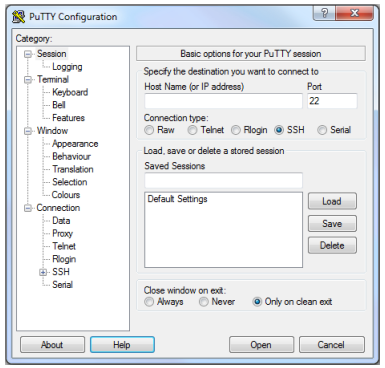
Full step by step instructions in Week 2, Lab 1 exercise.

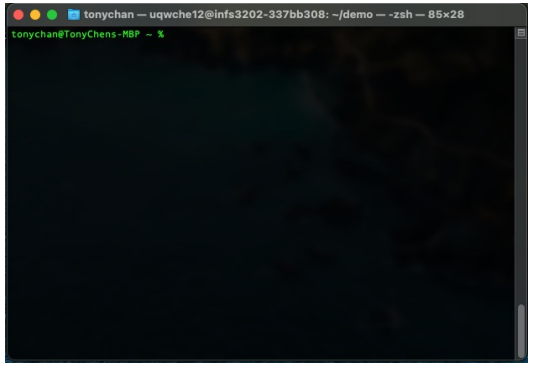
Accessing zones* - VPN required from outside the UQ network (vpn.uq.edu.au)

You can find your Zones web address and server name here: http://coursemgr.uqcloud.net/infs3202

SSH

All zones may be accessed for administration purposes through the commandline, though SSH access uses UQ usernames and passwords.





SFTP

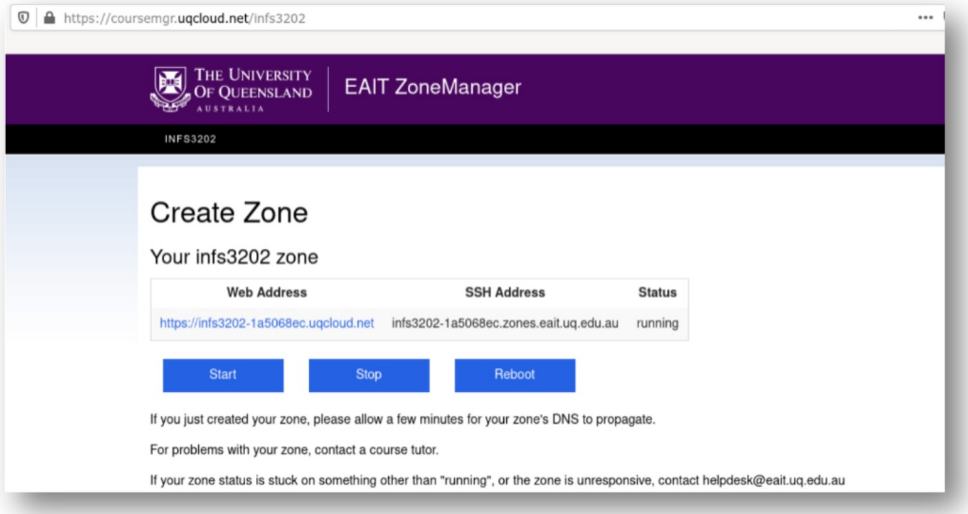
With a UQ user account, you can also access your zone over SFTP using a client such as WinSCP or FileZilla to upload or download files.

https://stluc.manta.uqcloud.net/xlex/public/zones-guide.html#accessing_zones

Connecting to your Student Zone

Connecting to your Zone

You can find your Zones web address and server name here:
http://coursemgr.uqcloud.net/infs3202



The status of the virtual server zone must be checked, and if it is found to be inactive, the “Start” button should be clicked. Two addresses are then provided: - The Web Address: is the public-facing address that can be used to access the user’s web pages. - The Server name (SSH Address): is a private address that allows the user to upload files and configure their application.

Tip

accessing the SSH Address from home requires the use of the UQ VPN. Instructions for setting up the UQ VPN can be found at the following link: https://my.uq.edu.au/information-and-services/information-technology/working-remotely/vpn-virtual-private-network

From Week 1 Lab 1

NGINX Web server

- Nginx (pronounced as "Engine-X")
- Open-source web server software designed with a focus on high performance, high concurrency, and low memory usage.
- Nginx is considered more lightweight than Apache primarily due to its event-driven architecture, which allows it to handle multiple connections more efficiently with a lower memory footprint.
- Its design choices, focusing on high performance and efficient resource use, make it an excellent option for high-traffic websites and applications.

NGINX Web server – Basic config file

- **root /var/www/html/htdocs;:**
  The root directive specifies the directory where Nginx should look for the files to serve.

- **index index.html index.htm;:**
  This specifies the default files that Nginx should look for when a directory is requested.

- After you change the configuration run:

  $ sudo systemctl reload nginx

```
http {
    server {
        listen 80; # Listen on port 80 for HTTP requests
        server_name example.com; # Replace with your domain name or use localhost

        # Document root where the files are located
        root /var/www/html/htdocs;

        # Default file to serve
        index index.html index.htm;

        # Serve files
        location / {
            try_files $uri $uri/ =404;
        }
    }
}
```

HTTP Status Codes

HTTP status codes are standardized responses that a web server sends to your browser to indicate the outcome of a requested action.

Code	Description
200 Ok	The request has succeeded. The meaning of the success depends on the HTTP method used.
201 Created	The request has been fulfilled, resulting in the creation of a new resource.
400 Bad Request	The server cannot process the request due to a client error (e.g., malformed request syntax).
401 Unauthorized	The request requires user authentication.
404 Not Found	The server has not found anything matching the Request-URI.
408 Request Timeout	The client did not produce a request within the time that the server was prepared to wait.
500 Internal Server Error	The server encountered an unexpected condition which prevented it from fulfilling the request. Usually a server-side setup issue or server-side coding error.

List of Status Codes:
https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Live UQZone Demo: VSCode and Website Editing

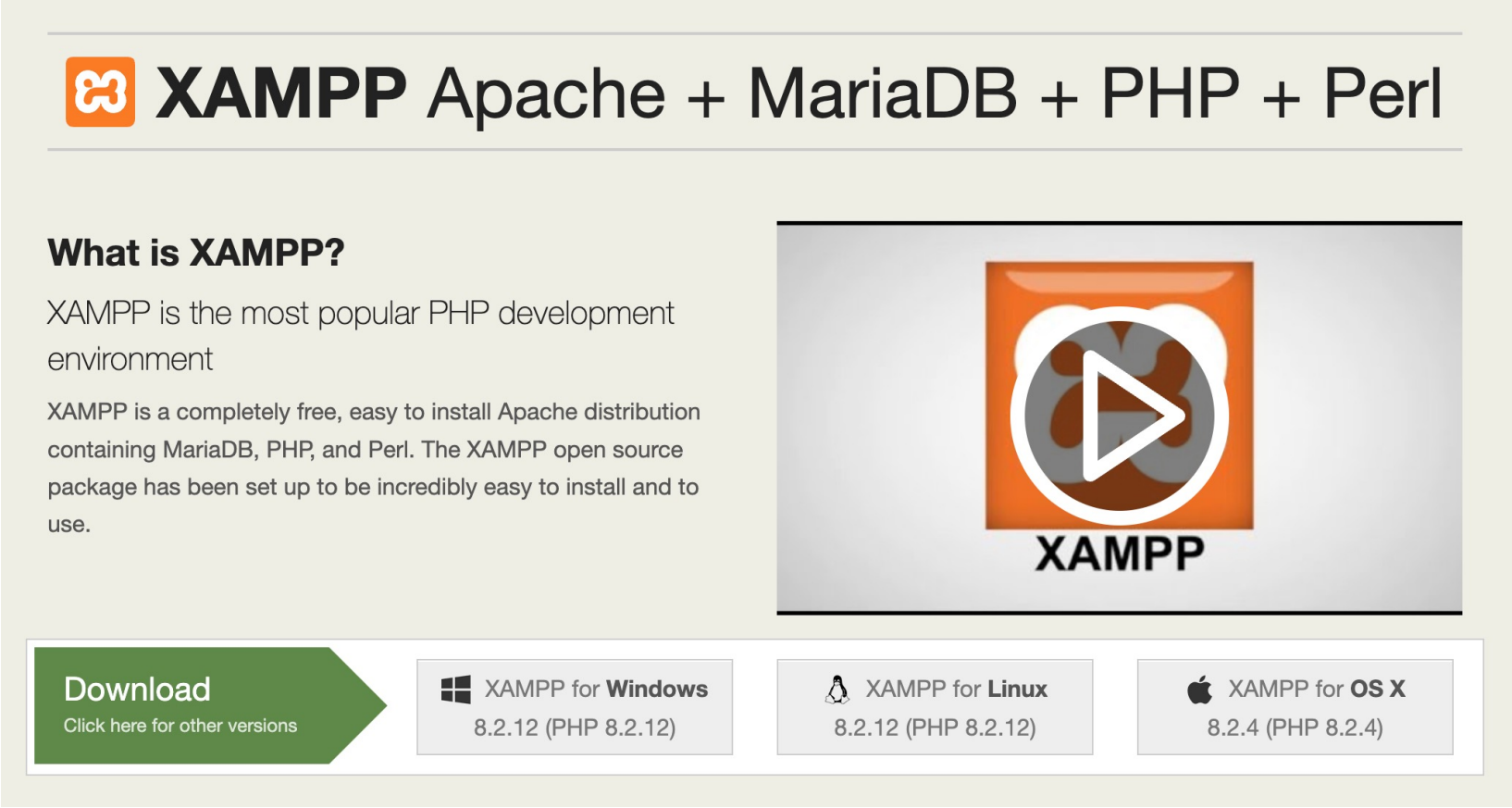
Code: week2_code/html_intro/html_simple_website_php

Local Development

XAMPP on Windows and Mac

A convenient solution :

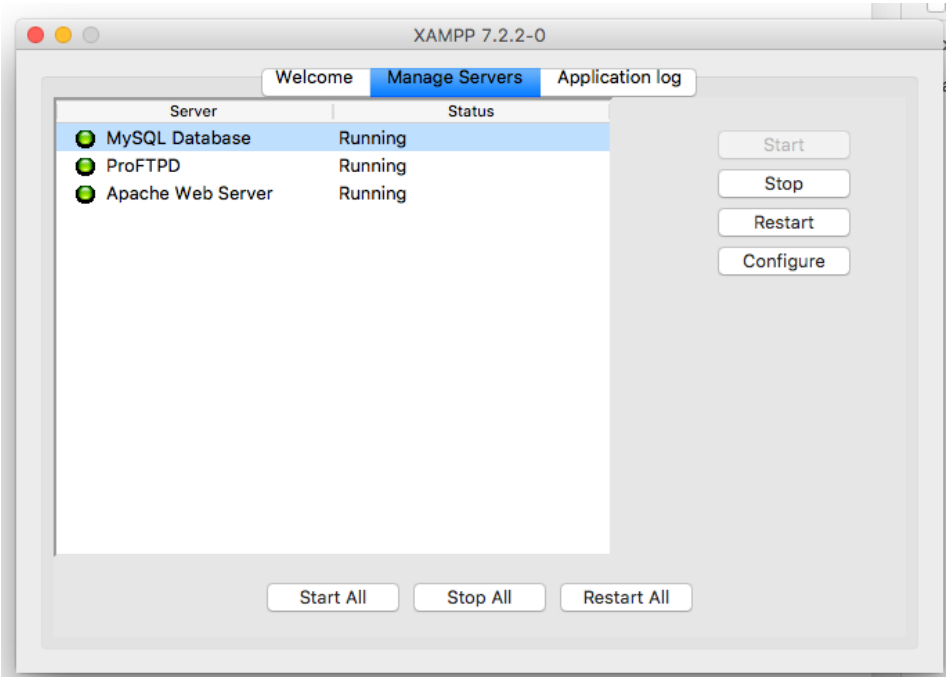
XAMPP: Stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P).
https://www.apachefriends.org/index.html

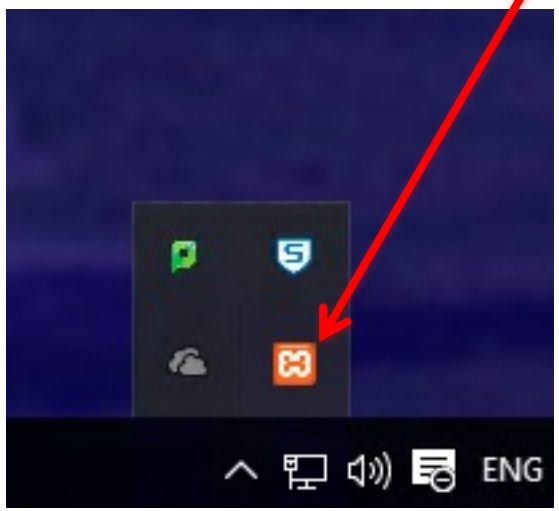


Unfortunately, no tool that packages NGINX.

In Week 3's lecture you'll be shown how to use Spark which the web server that comes with CodeIgniter.

XAMPP on Windows and Mac





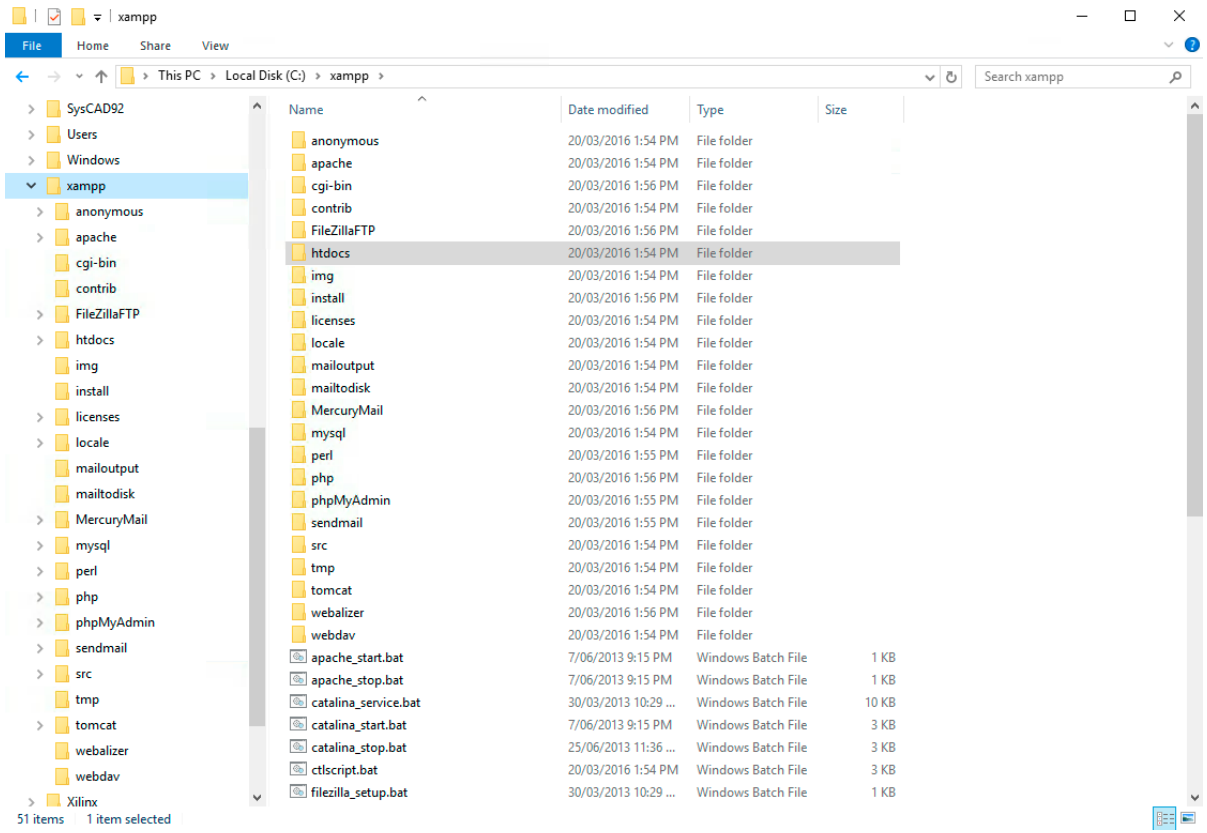
Configuring the Server

Show/Hide the XAMPP's control panel

Mac

XAMPP on Windows and Mac

Where is my server?
By Default: “C:\xampp” or on Mac /Applications/XAMPP/



Index.html

```
<html>
<head>
          <title>INFS3202</title>
</head>
<body>
          <p>Welcome to the course of INFS3202.</p>
          <p>I hope you enjoy this course.</p>
</body>
</html>
```

http://localhost/index.html

Welcome to the course of INFS3202.

I hope you enjoy this course.

Week 2: Todo

- Practicals/Labs start this week
- Weekly RiPPLE task is due in Week 2 on Friday at 3pm
- Complete the Self-paced HTML and CSS tutorials if you need a refresh
- Choose your Project topic or email a custom project to a.bakharia1@uq.edu.au before end of Week 2 to get approval

THE UNIVERSITY OF QUEENSLAND
AUSTRALIA
CREATE CHANGE

Q&A

Thank you