CS2204 Fundamentals of Internet Applications Development

Lecture 11 JavaScript Part 6

Computer Science, City University of Hong Kong

Semester B 2024-25

CW2 Reminder (Due on Apr 18th)

CW2 takes **13%** of the course assessment and will be due in two weeks on **April 18th, 23:59 PM**

Please **start as soon as possible** as they are 6 tasks in this assignment!

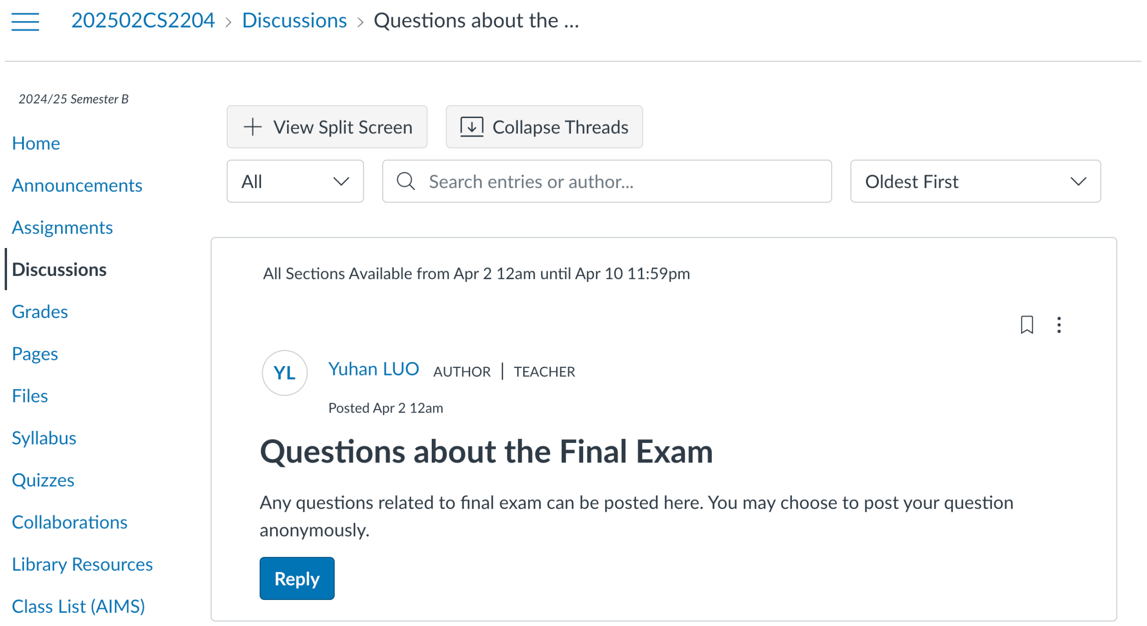
If you have any questions about assignment requirement, please **contact the student helpers for clarification.**

If your question is about how to handle a specific task or debugging, the helpers are not supposed to directly help you with that, but they may provide some hints if applicable.

Post your questions on Canvas before the last lecture (April 17th)

A discussion thread is created to collect your questions about the final exam.

The questions will be covered during the last lecture



All Sections Available from Apr 2 12am until Apr 10 11:59pm

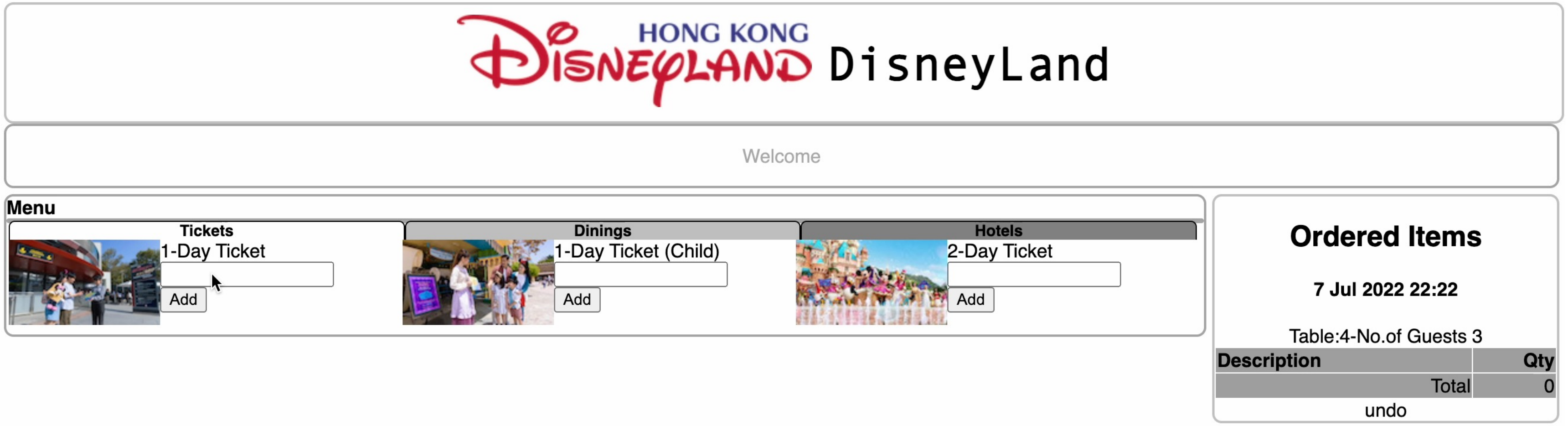
YL Yuhan LUO AUTHOR | TEACHER

Posted Apr 2 12am

Questions about the Final Exam

Any questions related to final exam can be posted here. You may choose to post your question anonymously.

CW2 Order Page Demo



DisneyLand

Welcome

Menu

Tickets	Dinings	Hotels
1-Day Ticket	1-Day Ticket (Child)	2-Day Ticket
Add	Add	Add

Ordered Items

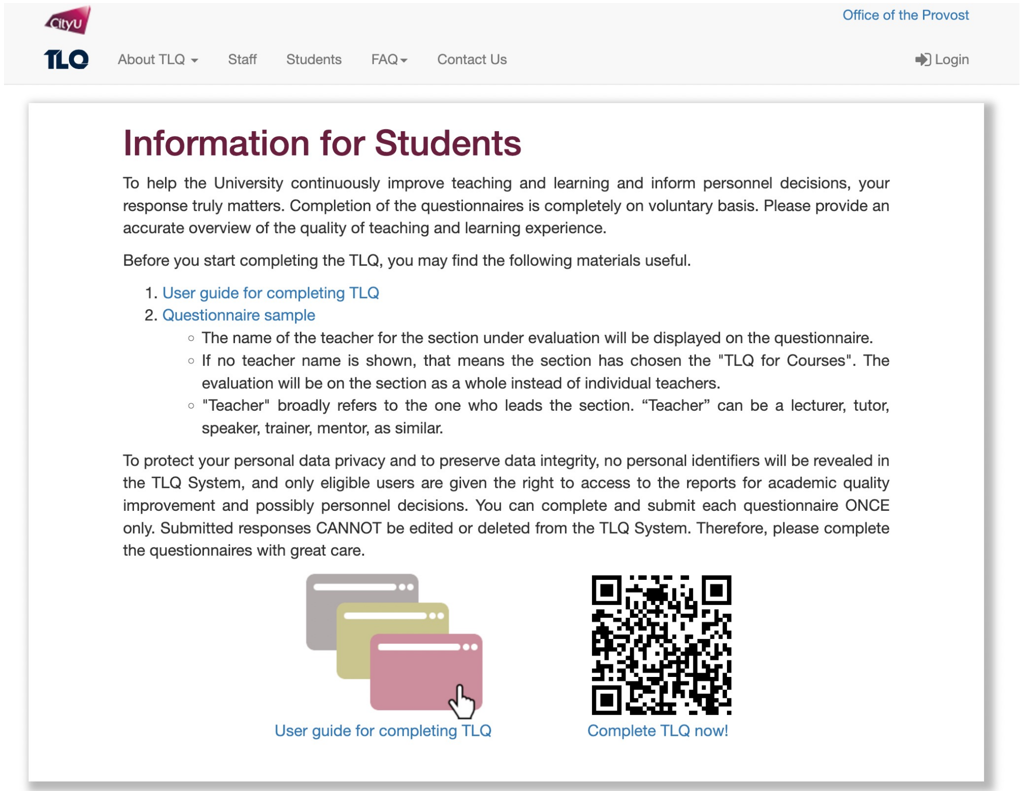
7 Jul 2022 22:22

Table:4-No.of Guests 3

Description	Qty
Total	0

undo

LOQ Reminder



Information for Students

To help the University continuously improve teaching and learning and inform personnel decisions, your response truly matters. Completion of the questionnaires is completely on voluntary basis. Please provide an accurate overview of the quality of teaching and learning experience.

Before you start completing the TLQ, you may find the following materials useful.

1. User guide for completing TLQ
2. Questionnaire sample
   - The name of the teacher for the section under evaluation will be displayed on the questionnaire.
   - If no teacher name is shown, that means the section has chosen the "TLQ for Courses". The evaluation will be on the section as a whole instead of individual teachers.
   - "Teacher" broadly refers to the one who leads the section. “Teacher” can be a lecturer, tutor, speaker, trainer, mentor, as similar.

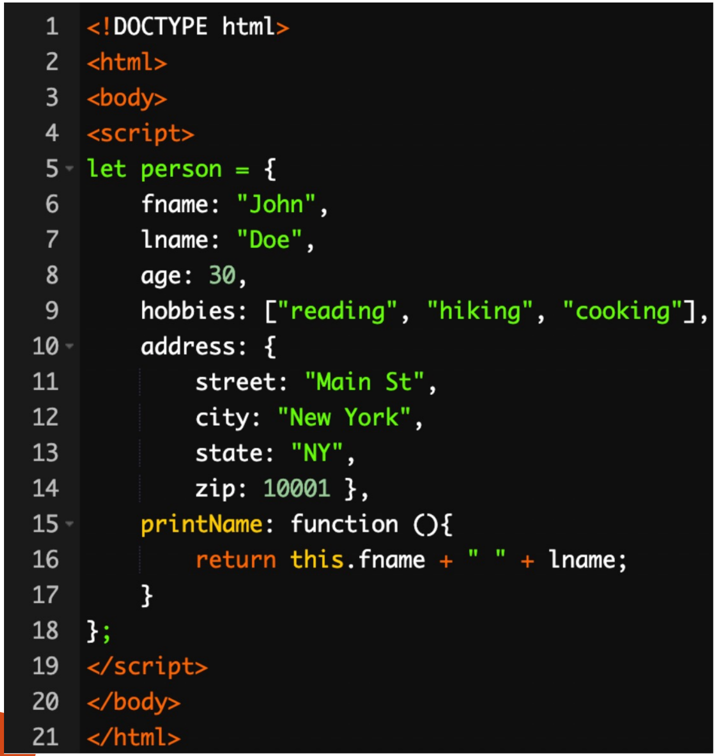
To protect your personal data privacy and to preserve data integrity, no personal identifiers will be revealed in the TLQ System, and only eligible users are given the right to access to the reports for academic quality improvement and possibly personnel decisions. You can complete and submit each questionnaire ONCE only. Submitted responses CANNOT be edited or deleted from the TLQ System. Therefore, please complete the questionnaires with great care.

User guide for completing TLQ

Complete TLQ now!

https://onlinesurvey.cityu.edu.hk/loq/

Post-Lab Quiz 8 Review



```
<!DOCTYPE html>
<html>
<body>
<script>
let person = {
    fname: "John",
    lname: "Doe",
    age: 30,
    hobbies: ["reading", "hiking", "cooking"],
    address: {
        street: "Main St",
        city: "New York",
        state: "NY",
        zip: 10001 },
    printName: function (){
        return this.fname + " " + lname;
    }
};
</script>
</body>
</html>
```

Which of the following statements are correct? (select all that apply)

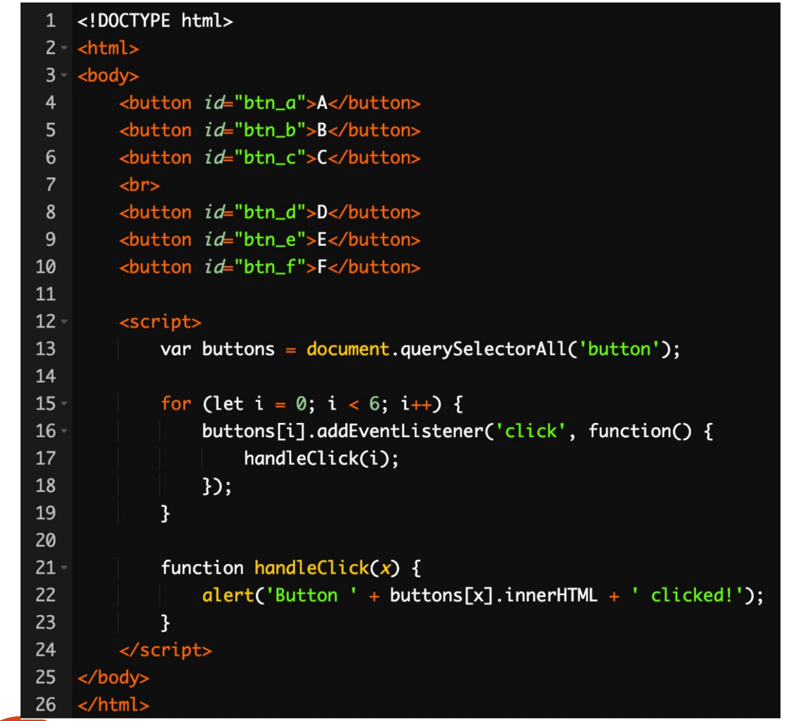
A. To display the person's full name in an alert window, one way is to call `alert(person.fname + " " + person.lname);`

B. To display the city attribute of the person in an alert window, one way is to call `alert(person.city);`

C. `alert(person.hobbies[3]);` will return "`undefined`"

D. `alert(person.printName);` is correct in syntax, but it will not return the person's full name

Post-Lab Quiz 8 Review



```
 1  <!DOCTYPE html>
 2  <html>
 3  <body>
 4      <button id="btn_a">A</button>
 5      <button id="btn_b">B</button>
 6      <button id="btn_c">C</button>
 7      <br>
 8      <button id="btn_d">D</button>
 9      <button id="btn_e">E</button>
10      <button id="btn_f">F</button>
11
12      <script>
13          var buttons = document.querySelectorAll('button');
14
15          for (let i = 0; i < 6; i++) {
16              buttons[i].addEventListener('click', function() {
17                  handleClick(i);
18              });
19          }
20
21          function handleClick(x) {
22              alert('Button ' + buttons[x].innerHTML + ' clicked!');
23          }
24      </script>
25  </body>
26  </html>
```

Which of the following statements will not change the information displayed in the alert window? (select all that apply)

A. In line #16, the keyword let can be replaced with `var`

B. In line #23, `buttons[x]` can be replaced with `this`

C. In line #17-18, the statement `function(){handleClick(i);}` can be replaced with `function(){handleClick(this);}` but at the same time, in the line #23, `button[x]` should be replaced with `x`

D. In line #13, `querySelectorAll('button')` can be replaced with `getElementsByTagName('button')`

Agenda

A brief intro to advanced web dev

- Styling libraries
- JS libraries

Programing practice

Mid-term Question review

Agenda

A brief intro to advanced web dev

- Styling libraries
- JS libraries

Programing practice

Mid-term Question review

Web Styling Frameworks

What are web styling frameworks?

Templates or libraries that help developers build **responsive** and appealing web applications

quickly and **easily**

Popular web styling frameworks

- **Bootstrap**: provides a set of pre-built UI
- **Material Design**: a design language developed by Google
- …

*** This part will not be tested in the final, but are helpful resources for you to learn about more advanced web dev in the future**

Bootstrap

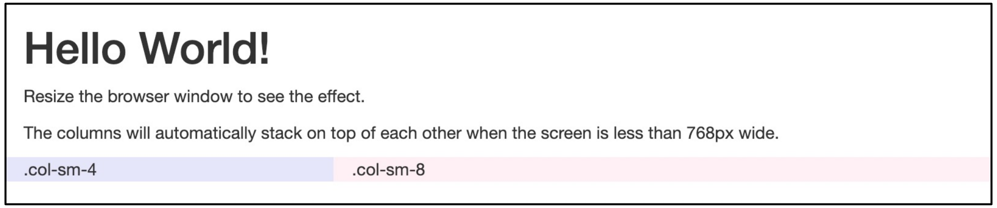
The most popular CSS Framework for developing responsive and **mobile-first** websites.

Free and open source: Link

How to incorporate Bootstrap framework?

```
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/boots
trap.min.css">
```

Bootstrap: a basic example



Hello World!

Resize the browser window to see the effect.

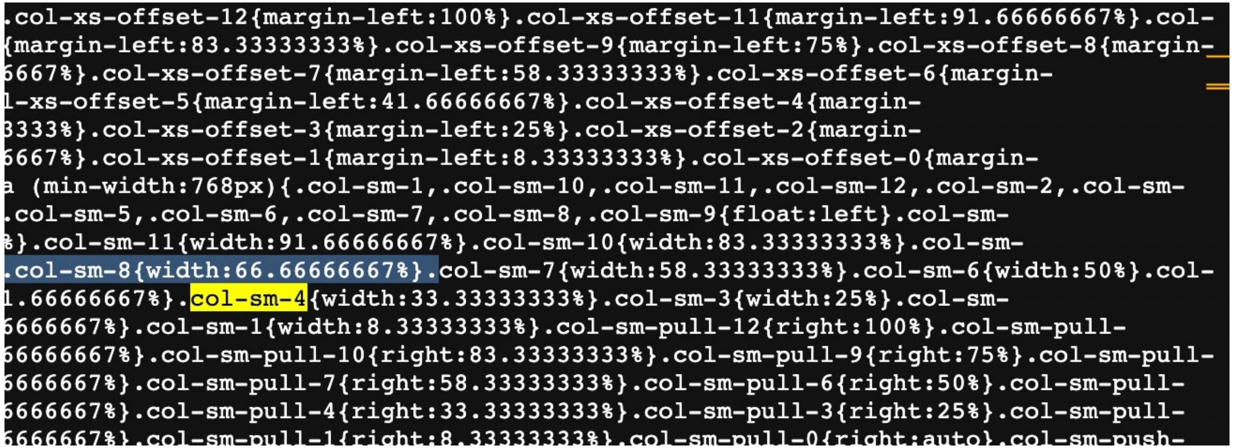
The columns will automatically stack on top of each other when the screen is less than 768px wide.

.col-sm-4 .col-sm-8

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet"
 href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
</head>
<body>

<div class="container-fluid">
  <h1>Hello World!</h1>
  <p>Resize the browser window to see the effect.</p>
  <p>The columns will automatically stack on top of each other when the screen is less
than 768px wide.</p>
  <div class="row">
    <div class="col-sm-4" style="background-color:lavender;">.col-sm-4</div>
    <div class="col-sm-8" style="background-color:lavenderblush;">.col-sm-8</div>
  </div>
</div>

</body>
</html>
```

Code Example: Lec11-01-css-bootstrap-intro.html

Bootstrap: a close look at the source



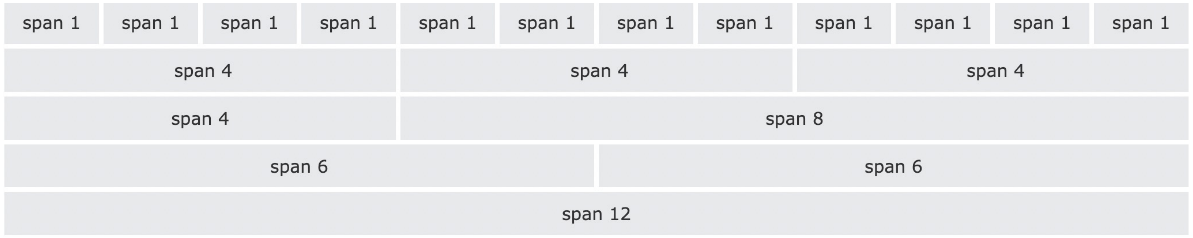
```
.col-xs-offset-12{margin-left:100%}.col-xs-offset-11{margin-left:91.66666667%}.col-
{margin-left:83.33333333%}.col-xs-offset-9{margin-left:75%}.col-xs-offset-8{margin-
6667%}.col-xs-offset-7{margin-left:58.33333333%}.col-xs-offset-6{margin-
l-xs-offset-5{margin-left:41.66666667%}.col-xs-offset-4{margin-
3333%}.col-xs-offset-3{margin-left:25%}.col-xs-offset-2{margin-
6667%}.col-xs-offset-1{margin-left:8.33333333%}.col-xs-offset-0{margin-
a (min-width:768px){.col-sm-1,.col-sm-10,.col-sm-11,.col-sm-12,.col-sm-2,.col-sm-
.col-sm-5,.col-sm-6,.col-sm-7,.col-sm-8,.col-sm-9{float:left}.col-sm-
%}.col-sm-11{width:91.66666667%}.col-sm-10{width:83.33333333%}.col-sm-
.col-sm-8{width:66.66666667%}.col-sm-7{width:58.33333333%}.col-sm-6{width:50%}.col-
1.66666667%}.col-sm-4{width:33.33333333%}.col-sm-3{width:25%}.col-sm-
666667%}.col-sm-1{width:8.33333333%}.col-sm-pull-12{right:100%}.col-sm-pull-
6666667%}.col-sm-pull-10{right:83.33333333%}.col-sm-pull-9{right:75%}.col-sm-pull-
666667%}.col-sm-pull-7{right:58.33333333%}.col-sm-pull-6{right:50%}.col-sm-pull-
666667%}.col-sm-pull-4{right:33.33333333%}.col-sm-pull-3{right:25%}.col-sm-pull-
666667%}.col-sm-pull-1{right:8.33333333%}.col-sm-pull-0{right:auto}.col-sm-push-
```

- The property of the class col-sm-4 and col-sm-8 are predefined for direct use
- The `min.css` means that all the blank spaces and line switches in the file are removed to save space

Link: https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css

Bootstrap: Grid

Intuitive 12 columns


span 1 span 1 span 1 span 1 span 1 span 1 span 1 span 1 span 1 span 1 span 1 span 1
span 4 span 4 span 4
span 4 span 8
span 6 span 6
span 12

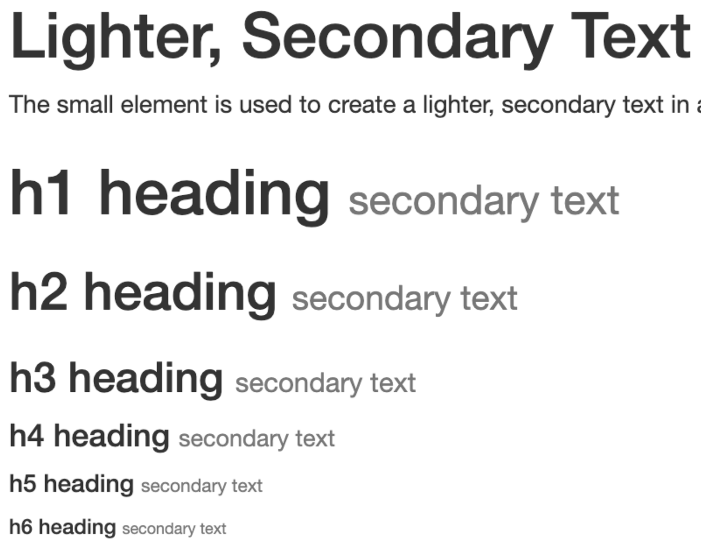
Four classes

- **xs** (for phones - screens less than 768px wide)
- **sm** (for tablets - screens equal to or greater than 768px wide)
- **md** (for small laptops - screens equal to or greater than 992px wide)
- **lg** (for laptops and desktops - screens equal to or greater than 1200px wide)

The classes above can be combined to create more dynamic and flexible layouts.

Bootstrap: Text/Typography

In Bootstrap the HTML `<small>` element is used to create a lighter, secondary text in any heading



Lighter, Secondary Text

The small element is used to create a lighter, secondary text in any heading:

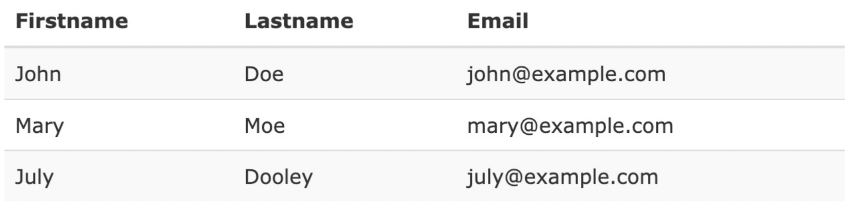
h1 heading secondary text
h2 heading secondary text
h3 heading secondary text
h4 heading secondary text
h5 heading secondary text
h6 heading secondary text

```
<div class="container">
  <h1>Lighter, Secondary Text</h1>
  <p>The small element is used to create a lighter, secondary text in
any heading:</p>
  <h1>h1 heading <small>secondary text</small></h1>
  <h2>h2 heading <small>secondary text</small></h2>
  <h3>h3 heading <small>secondary text</small></h3>
  <h4>h4 heading <small>secondary text</small></h4>
  <h5>h5 heading <small>secondary text</small></h5>
  <h6>h6 heading <small>secondary text</small></h6>
</div>
```

Code Example: Lec11-02-css-bootstrap-text.html

Bootstrap: Table

The `.table` class adds basic styling to a table
The `.table-striped` class adds zebra-stripes to



Firstname	Lastname	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com

```html
<table class="table table-striped">
  <thead>
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Email</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>John</td>
      <td>Doe</td>
      <td>john@example.com</td>
    </tr>
    <tr>
      <td>Mary</td>
      <td>Moe</td>
      <td>mary@example.com</td>
    </tr>
    <tr>
      <td>July</td>
      <td>Dooley</td>
      <td>july@example.com</td>
    </tr>
  </tbody>
</table>
```

Advantages

- Providing consistent styling
- No need to maintain
- Web developers could focus on more important functionalities

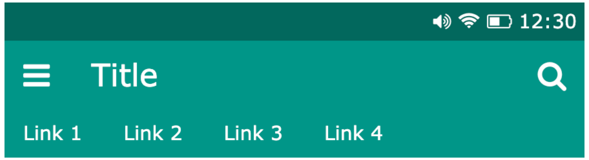
Code Example: Lec11-03-css-bootstrap-table.html

Material Design

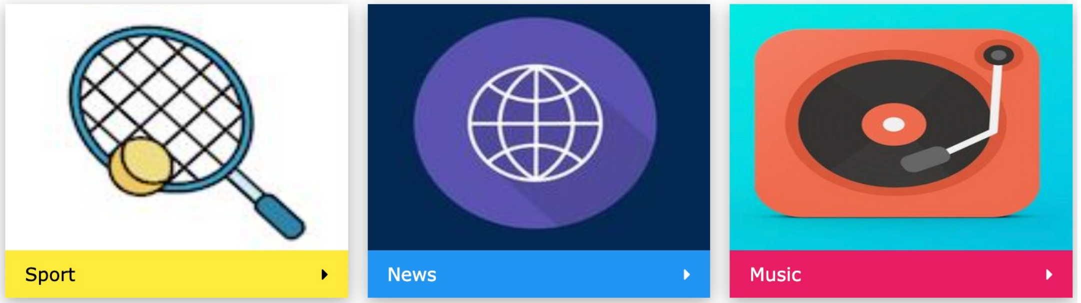
Designed by Google in 2014 and has later been adopted in many Google applications

To quickly and easily create beautiful and responsive web applications that adhere to the “material styling” looks.

Commonly used for creating web apps



Material Design Look (Colorful Cards)

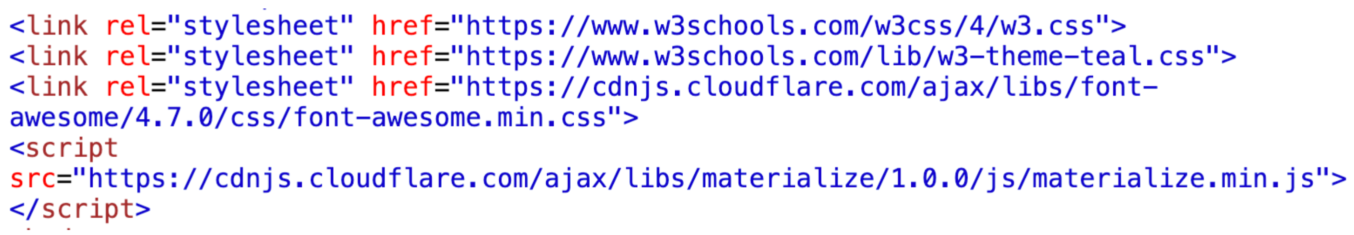


Material Design: how to use

There are several Material Design frameworks we can use

- Materialize
- Cloudflare
- W3School also provides several frameworks such as **`w3.css`**, **`w3-theme-teal.css`**

To adopt a material styling web, sometimes JS libs are also needed in order to create interactive effects

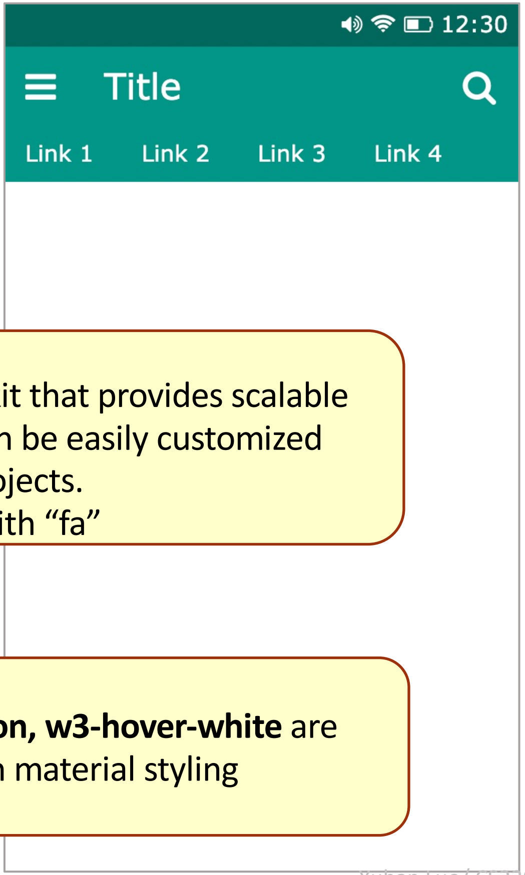


```
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet" href="https://www.w3schools.com/lib/w3-theme-teal.css">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
<script
src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js">
</script>
```

Material Design: Example

Create a mobile app style web interface

```
<!DOCTYPE html>
<html>
<title>W3.CSS</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet" href="https://www.w3schools.com/lib/w3-theme-teal.css">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
<body>

<div class="w3-container w3-padding-small w3-theme-d3">
  <div class="w3-right">
    <i class="fa fa-volume-up"></i>
    <i class="fa fa-wifi"></i>
    <i class="fa fa-battery-2"></i>
    12:30
  </div>
</div>

<div class="w3-bar w3-theme w3-xlarge">
  <a class="w3-bar-item w3-button" href="#"><i class="fa fa-bars"></i></a>
  <span class="w3-bar-item">Title</span>
  <a class="w3-bar-item w3-button w3-right" href="#"><i class="fa fa-search"></i></a>
</div>

<div class="w3-bar w3-theme">
  <a class="w3-bar-item w3-button w3-hover-white">Link 1</a>
  <a class="w3-bar-item w3-button w3-hover-white">Link 2</a>
  <a class="w3-bar-item w3-button w3-hover-white">Link 3</a>
  <a class="w3-bar-item w3-button w3-hover-white">Link 4</a>
</div>

</body>
</html>
```



Font Awesome
- a popular icon toolkit that provides scalable vector icons that can be easily customized and used in web projects.
- Class name starts with "fa"

W3-bar-item, w3-button, w3-hover-white are predefined classes with material styling

Code Example: Lec11-05-css-bootstrap-material-design.html

JS Frameworks

What are JS frameworks?

Collections of pre-written JS code that provide consistent structures for web development, with a goal of simplifying the development process.

Popular JS frameworks

- **Angular**: full-featured front-end framework for web applications
- **React**: Front-end framework for building user interfaces

*** This part will not be tested in the final, but are helpful resources for you to learn about more advanced web dev in the future**

Angular: Introduction

Origin

- Developed and maintained by Google
- First released in 2010
- Building complex and dynamic web applications

Examples of web applications

- Google Ads, Google Cloud console
- Microsoft 365 suite
- Weather.com

Angular: key components

How to apply Angular in a web page?

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
```

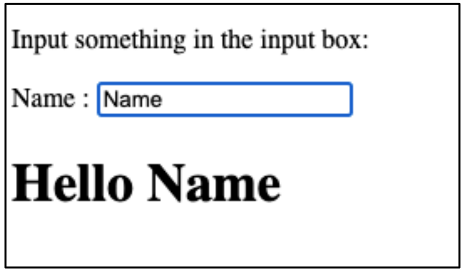
ng-directive

- **ng-app**: defines an Angular application, used to bootstrap the Angular application; if it is an empty string, Angular will disable the automatic boostrap
- **ng-model**: binds the value of HTML controls to application data
- **ng-blind**: binds application data to the HTML view

```
<div ng-app="">
  <p>Name: <input type="text" ng-model="name"></p>
  <p ng-bind="name"></p>
</div>
```

Angular: Directives (1)

```
<!DOCTYPE html>
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.
6.9/angular.min.js"></script>
<body>

<div ng-app="">

<p>Input something in the input box:</p>
<p>Name : <input type="text" ng-model="name"
placeholder="Enter name here"></p>
<h1>Hello {{name}}</h1>

</div>

</body>
</html>
```



Input something in the input box:

Name : Name

Hello Name

Code Example: Lec11-06-hello-angular.html

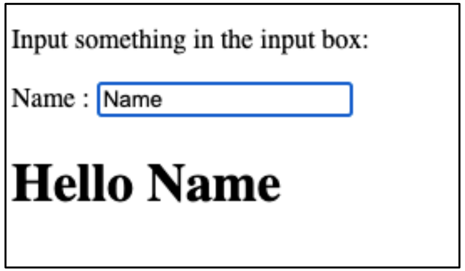
Critical thinking:
How to achieve the same effect using traditional html and JS?

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
    <p>Input something in the input box:</p>
    <p>Name : <input type="text" id="name"
    placeholder="Enter name here"></p>
    <h1 id="greeting">Hello</h1>

    <script>
        const nameInput =
        document.getElementById('name');
        const greeting =
        document.getElementById('greeting');
        nameInput.addEventListener('input', (event)
        => {
            greeting.textContent = `Hello
            ${event.target.value}`;
        });
    </script>
</body>
</html>
```

Code Example: Lec11-07-hello-traditional-JS.html

Angular: Directives (2)

```
<!DOCTYPE html>
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.
6.9/angular.min.js"></script>
<body>

<div ng-app="">

<p>Input something in the input box:</p>
<p>Name : <input type="text" ng-model="name"
placeholder="Enter name here"></p>
<h1>Hello {{name}}</h1>

</div>

</body>
</html>
```


Input something in the input box:

Name : Name

Hello Name

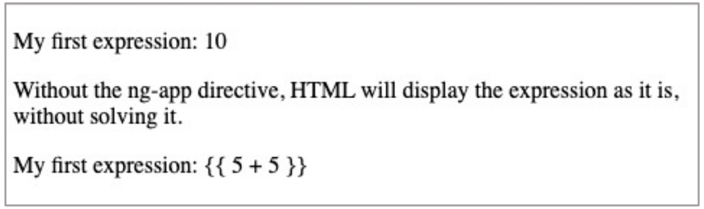
Code Example: Lec11-06-hello-angular.html

Advantages

- **ng-model directive** is used to bind the value of the input field to a variable called `{{name}}`
- No need to assign individual id or classes to read and write the HTML content
- **Two-way data binding mechanism**, allowing changes in the model to be automatically reflected in the view, and vice versa
- No traditional event handlers

Angular: Expressions

```
<!DOCTYPE html>
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.
6.9/angular.min.js"></script>
<body>

<div ng-app>
<p>My first expression: {{ 5 + 5 }}</p>
</div>

<p>Without the ng-app directive, HTML will display the
expression as it is, without solving it.</p>

<div>
<p>My first expression: {{ 5 + 5 }}</p>
</div>

</body>
</html>
```



```
My first expression: 10

Without the ng-app directive, HTML will display the expression as it is,
without solving it.

My first expression: {{ 5 + 5 }}
```

Code Example: Lec11-08-angular-expression.html

- Expressions are written inside double braces: `{{ expression }}`.
- AngularJS will resolve the expression, and return the result exactly where the expression is written in JS.
- Like JS expressions, AngularJS expressions can contain literals, operators, and variables

Critical thinking:
How to achieve the same effect using traditional html and JS?

React

Origin

- Developed by Facebook in 2013
- Building user interfaces
- Allowing reusable UI components
- Well known for its simplicity, flexibility, and performance, as well as its large and active community of developers
- React Native, extended from React, can used for mobile app development **for both Android and iOS**

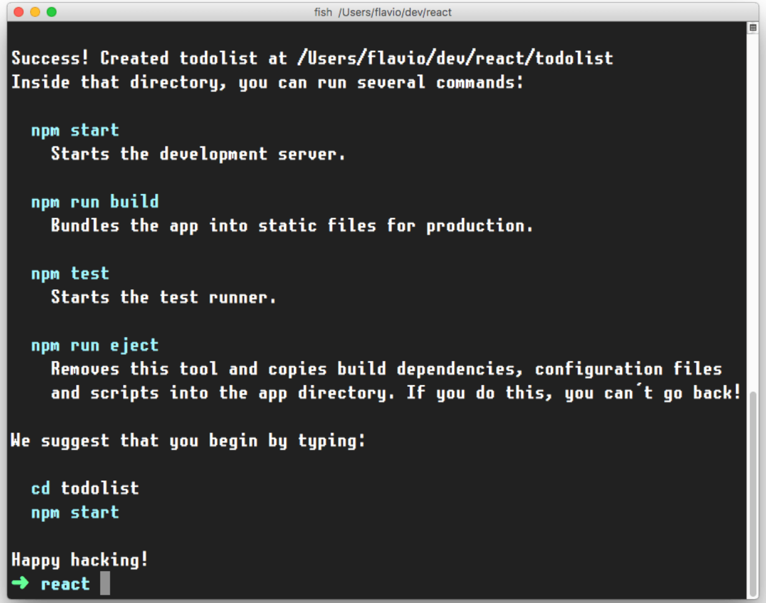
Examples of React web applications

- Facebook, Slack, Airbnb, Instagram, Netflix

React: how to run?

Requirement

- Node.js to be installed in your computer as a JS runtime environment to build the react app
- **Package manager**: npm or yarn to manage dependencies of the React app (npm is included in node.js but yarn needs to be included separately)
- **A command-line tool** to quickly set up a react project



```
Success! Created todolist at /Users/flavio/dev/react/todolist
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can´t go back!

We suggest that you begin by typing:

  cd todolist
  npm start

Happy hacking!
➜ react
```

React: test and basics

Available online react editor

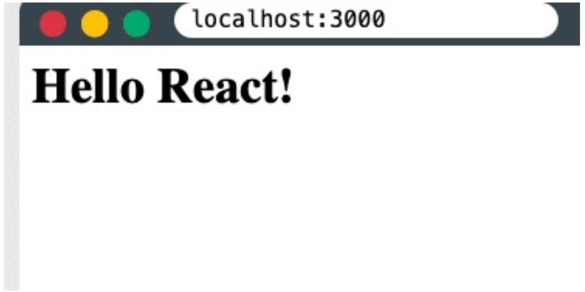
- W3Schools "Show React" Tool
- React Playground (playcode.io)

React Basics

- **React component:** a modular and reusable piece of UI that encapsulates logic and functionality.
- **ReactDOM:** used to render React Components to DOM and manipulate the UI of a web page; used to create dynamic and sophisticated web page instead of directly writing HTML and CSS code

React: Example (1)

```
import React from 'react';
import ReactDOM from 'react-dom/client';

const myFirstElement = <h1>Hello React!</h1>

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(myFirstElement);
```



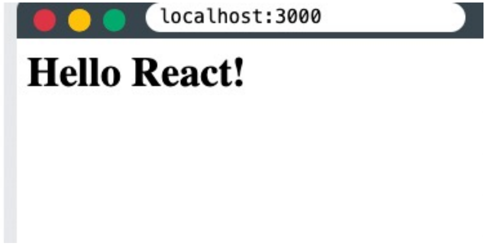
Hello React!

Steps

- Import the React and ReactDOM libraries into the current module
- Create a react element called myFirstElement
- Create a new root element with createRoot() taking a container 'root' as its argument, where the React element will be rendered

React: Example (2)

```
import React from 'react';
import ReactDOM from 'react-dom/client';

const myFirstElement = <h1>Hello React!</h1>

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(myFirstElement);
```



Hello React!

Why do we need React to create HTML elements instead of directly writing HTML?

- **Reusability**: create more maintainable and scalable UI over time
- **Data-driven UI**: build dynamic UI that responds to user actions
- **Performance**: Uses Virtual DOM to optimize performance; UI are only applied to the parts of the DOM that actually need to be changed – particularly useful for complicated UI

Agenda

A brief intro to advanced web dev

- Styling libraries
- JS libraries

Programing practice

Mid-term Question review

CSS layout and responsive design

Given the HTML code, complete the partially finished CSS to make the two `<div>` elements are displayed side by side when the screen `width` is larger than `768px` and otherwise stacked vertically. Write the CSS code in the code box below. **Do not** use "`flex`"

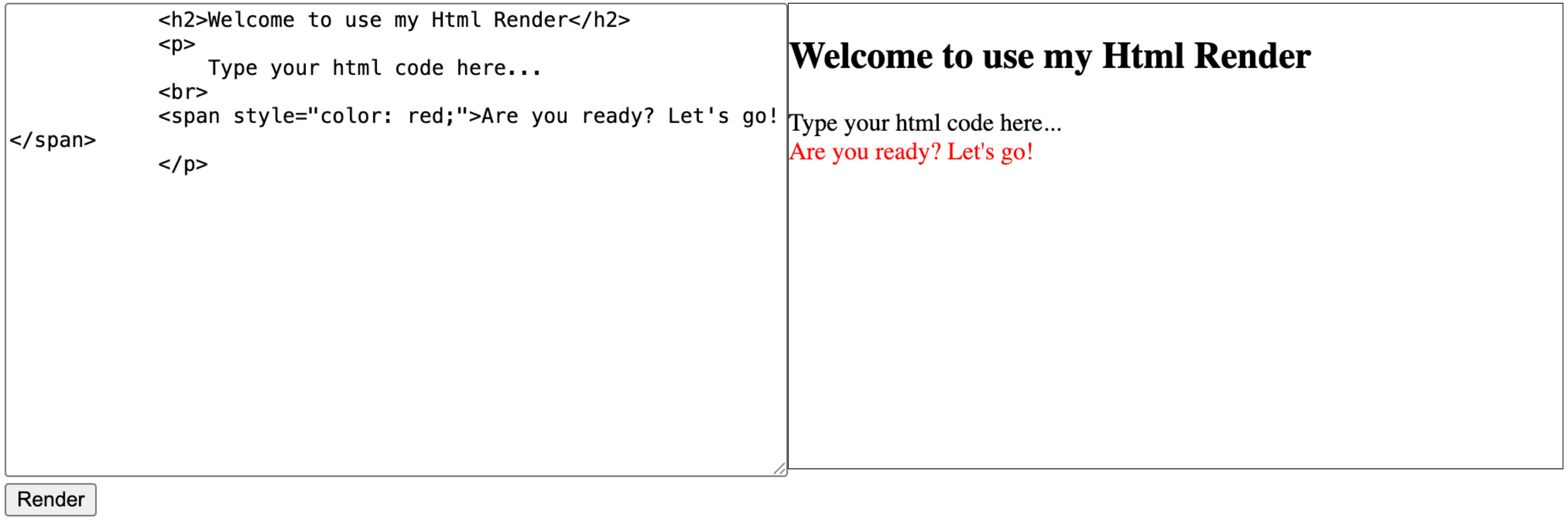
```
<body>
<div class="box"></div>
<div class="box"></div>
</body>
```

```
<style>
 div {
  border: 1px solid;
  width: 300px;
  height: 300px;
  }











</style>
```

Code Example: Lec11-09-css-styling.html

Form input and innerHTML render

How to create a webpage where users can input their html code into a textbox, and then render it to show the results of the code in another box?



```
            <h2>Welcome to use my Html Render</h2>
            <p>
                Type your html code here...
            <br>
            <span style="color: red;">Are you ready? Let's go!
</span>
            </p>
```

Welcome to use my Html Render

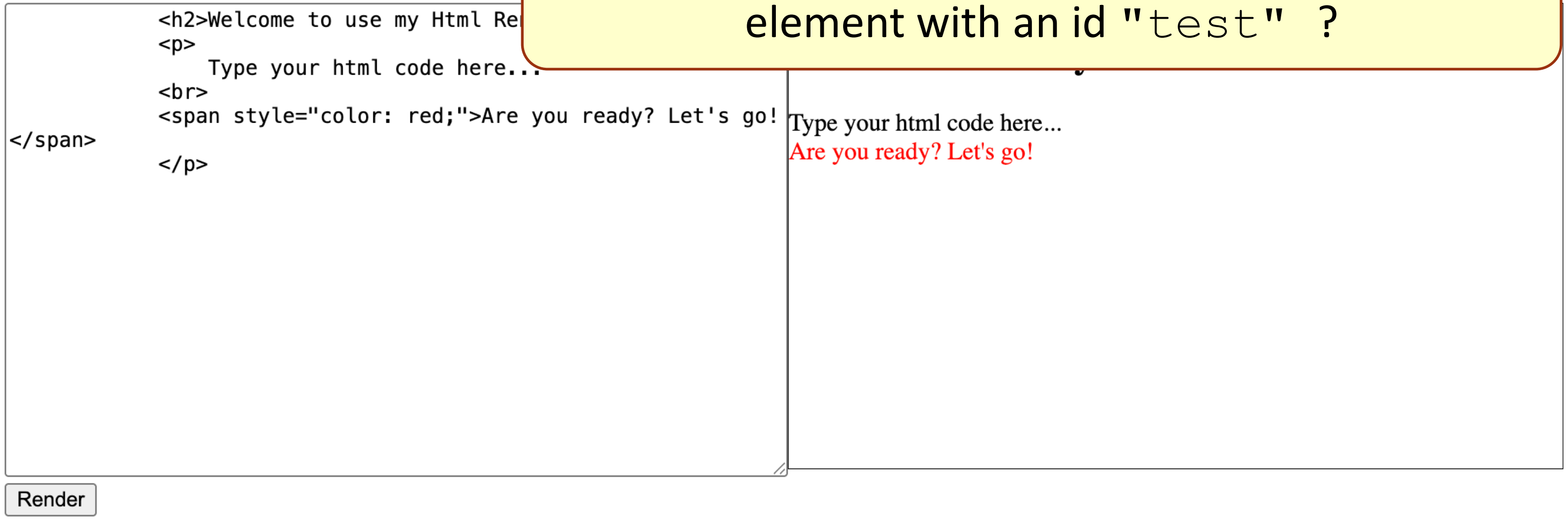
Type your html code here...
Are you ready? Let's go!

Form input and innerHTML render

First, think about how what will happen when you do sth like this:

```
document.getElementById("test").innerHTML =
<h2>Hello!</h2>;
```

Will the html tags `<h2></h2>` be shown in the element with an id `"test"` ?

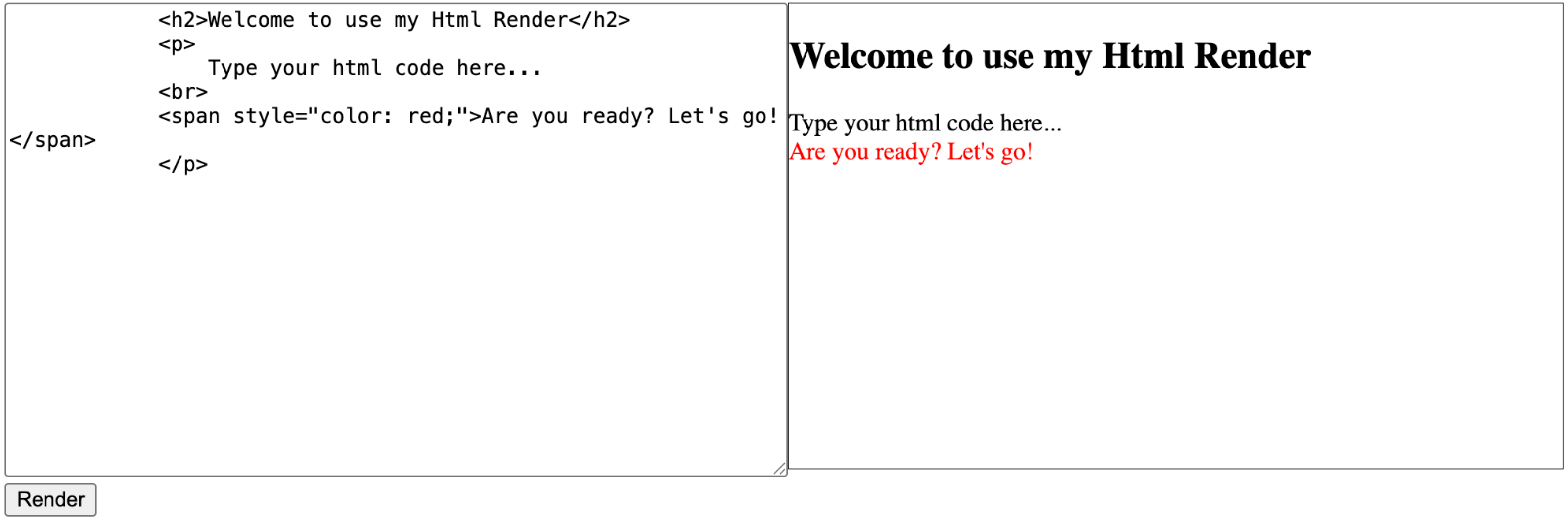


```
            <h2>Welcome to use my Html Re
            <p>
                Type your html code here...
            <br>
            <span style="color: red;">Are you ready? Let's go!
</span>
            </p>
```

Type your html code here...
Are you ready? Let's go!

Form input and innerHTML render

Since `innerHTML` will interpret the HTML code and display them accordingly, we can directly create a div and assign its `innerHTML` with the value that user entered in the textarea



```
            <h2>Welcome to use my Html Render</h2>
            <p>
                Type your html code here...
            <br>
            <span style="color: red;">Are you ready? Let's go!
</span>
            </p>
```

Welcome to use my Html Render

Type your html code here...
Are you ready? Let's go!

Form input and innerHTML render

```
<div id="codeConsole">
 <textarea id="codeConsoleCore" onkeyup="render();">
   <h2>Welcome to use my Html Render</h2>
   <p> Type your html code here...
   <br><span style="color: red;">Are you ready? Let's go!</span></p>
    </textarea>
  </div>
<div id="displayPanel"></div>
<button id="mtrender" onclick="render();"> Render </button>

<script>
function render(){
 var htmlCode = document.getElementById('codeConsoleCore').value;
 document.getElementById('displayPanel').innerHTML = htmlCode;
     }
</script>
```

Code Example: Lec11-10-JS-render.html

JavaScript Array

Complete the function `"displayFruits()"` to display all the fruits in the `"fruit-container"` every time when the page refreshes.

```
<body
onload="displayFruits();">
<div id=
"fruit-container"></div>
<button id="add">Add</button>
<button id="remove">Remove
(later in, first out)</button>
</body>
```

```
<script>
fruits = ["apple","peach","mango","grape"];

function displayFruits(){
 var s = "";




}

...

</script>
```

Code Example: Lec11-11-JS-array.html

JavaScript Array

Create a JavaScript function `add()` that allows users to add fruit into the array `"fruits"` every time the function is called.

```
<body
onload="displayFruits();">
<div id=
"fruit-container"></div>
<button id="add" onclick
="add();" >Add</button>
<button id="remove">Remove
(later in, first out)</button>
</body>
```

```
<script>
fruits = ["apple","peach","mango","grape"];

function displayFruits(){...}

function add (){




}

</script>
```

Code Example: Lec11-11-JS-array.html

JavaScript Array

Create a JavaScript function `remove()` that allows users to remove the last added fruit item from the array `"fruits"` every time the function is called.

```
<body
onload="displayFruits();">
<div id=
"fruit-container"></div>
<button id="add" onclick
="add();" >Add</button>
<button id="remove” onclick
=”remove();" >Remove (later
in, first out)</button>
</body>
```

Code Example: Lec11-11-JS-array.html

```
<script>
fruits = ["apple","peach","mango","grape"];

function displayFruits(){...}

function add (){...}

function remove (){




}
</script>
```

JavaScript Array

If removing the `onclick` attributes for the `add` and `remove` buttons, how to expand the JS code to make the click event work?

```
<body
onload="displayFruits();">
<div id=
"fruit-container"></div>
<button id="add" ~~onclick
="add();"~~ >Add</button>
<button id="remove” ~~onclick
=”remove();"~~ >Remove (later
in, first out)</button>
</body>
```

```
<script>
fruits = ["apple","peach","mango","grape"];

function displayFruits(){...}

function add (){...}

function remove (){...}









</script>
```

Code Example: Lec11-11-JS-array.html

JavaScript Array

If removing the `onclick` attributes for the `add` and `remove` buttons, how to expand the JS code to make the click event work?

```
<body
onload="displayFruits();">
<div id=
"fruit-container"></div>
<button id="add" ~~onclick
="add();"~~ >Add</button>
<button id="remove” ~~onclick
=”remove();"~~ >Remove (later
in, first out)</button>
</body>
```

```
<script>
fruits = ["apple","peach","mango","grape"];

function displayFruits(){...}

function add (){...}

function remove (){...}

</script>
```

Code Example: Lec11-11-JS-array.html

Periodically executed functions

Build a webpage which allows users to set a **countdown timer** based on a number the they enter in the input field, with two buttons to control the start (`btn_start`) and stop (`btn_stop`) of the timer

```
<body> <h1>A counter down Timer</h2>
<p>Timer: <span id="demo"></span></p>
<input type="text" id = "sec"/>
<button id="btn_start"
onclick="auto_timer();">Start</button>
<button id="btn_stop”
onclick="stop();" >Stop</button>
</body>
```

```
<script>
 var countdown, seconds = 0;
 function auto_timer(){
   ...
  }

 function count (){
   ...
 }

 function stop (){
   ...
 }
</script>
```

Code Example: Lec11-12-JS-countdown.html

Periodically executed functions

(1) Creating a timer and displaying the counting down message in real time

```
<body> <h1>A counter down
Timer</h2>
<p>Timer: <span
id="demo"></span></p>
<input type="text" id = "sec"/>
<button id="btn_start"
onclick="auto_timer();">Start</but
ton>
<button id="btn_stop”
onclick="stop();" >Stop</button>
</body>
```

```
<script>
 var countdown, seconds = 0;

 function auto timer(){









  }

 function count() {



 }
 ...
</script>
```

Code Example: Lec11-12-JS-countdown.html

Periodically executed functions

(2) Stop the timer when the `stop` button is clicked

```
<body> <h1>A counter down
Timer</h2>

<p>Timer: <span
id="demo"></span></p>

<input type="text" id = "sec"/>

<button id="btn_start"
onclick="auto_timer();">Start</but
ton>

<button id="btn_stop”
onclick="stop();" >Stop</button>

</body>
```

```
<script>
 var countdown, seconds = 0;

 function auto_timer(){ ... }

 function count() {
   seconds--;
   document.getElementById("demo”)
   .innerText = seconds;



 }

 function stop() {



   }
 ...
</script>
```

Code Example: Lec11-12-JS-countdown.html

Agenda

Programing practice

A brief intro to advanced web dev

- Styling libraries
- JS libraries

Mid-term Question review

Q3. Network topology (30%)

Which network topology features a central hub that communicates with all other nodes?

Q7. Communication channel (57%)

Which of the following is NOT a communication channel in a network context?

Q8. HTML list (62%)

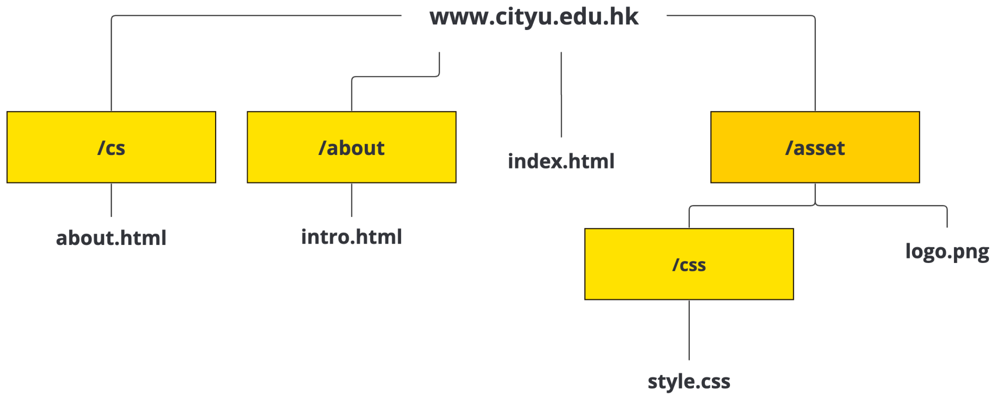
There are different types of lists we can create with HTML. Which of the following is the correct way to create an ordered list starting with “c”?

Q9. HTML image (23%)

Which of the following HTML codes makes an image a clickable link?

Q10. File path (32%)

The following is the structure of a website. Assuming that the current directory is /about, how to navigate to logo.png with relative path?



Q11. HTML form (41%)

In an HTML form, which attribute is necessary to specify the location where the form data should be sent?

Q12. HTML form input type (43%)

Which form element is used to create a multiline text input?

Q13. HTML form (43%)

Which tag is the best to group related form elements with a common usage?

Q14. HTML form input type (63%)

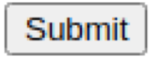
What is the main difference between checkboxes and radio buttons in an HTML form?

Q15. HTML form action (38%)

What is the primary difference between the POST and GET methods?

Q16. HTML button (52%)

Which of the following does NOT create a button that looks like the screenshot shown below?



Q17. Web accessibility (37%)

What is an essential principle to consider when creating an accessible web page?

Q19. CSS selector (63%)

Which statement about CSS Selector is true?

Q20. CSS position (35%)

Given a paragraph `<p>Hello World</p>` **in an HTML, which of the following CSS code can make it always stay in the same position (not moving with page scrolling) and on the top of all the elements in case of overlap?**

Q21. CSS box model (43%)

Which statement of box model is true?

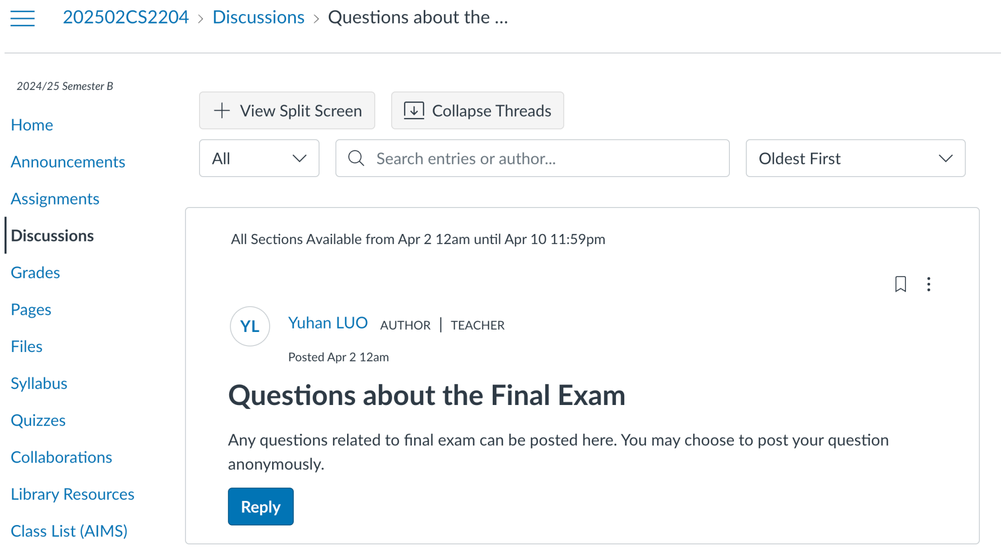
Q23. CSS comment (44%)

How to add a comment in a CSS?

For next lecture

We will review all the lectures learned in this course and continue to cover the remaining high-error-rate questions in the mid-term

If you have any questions about the final exam, do not forget to post them on Canvas



All Sections Available from Apr 2 12am until Apr 10 11:59pm

YL Yuhan LUO AUTHOR | TEACHER

Posted Apr 2 12am

Questions about the Final Exam

Any questions related to final exam can be posted here. You may choose to post your question anonymously.