Newton School
Coding Club
Bhagwan Parshuram Institute of Technology

25 HARD JS

DAY 1

Introduction to Web Development and Code Editor Setup

Introduction to HTML, CSS, JavaScript & How websites work? | Web Development Tutorials #1

HTML Tutorial: Installing VS Code & Live Server | Web Development Tutorials #2

DAY 2

HTML Basics

HTML Tutorial for Beginners: HTML Crash Course

DAY 3

Some Basic Tags in HTML

HTML Tutorial: Img and Anchor tags | Web Development Tutorials #6

HTML Tutorial: Lists and Tables | Web Development Tutorials #7

HTML Tables: Rows, Columns, and Advanced Attributes | MERN Stack Web Development || Episode - 7

DAY 4

Forms In HTML

Forms in HTML || Complete Web Dev Series 2023 || Episode - 8

FOLLOW FOR MORE

DAY 5

HTML IDs, classes and Inline, Block Elements

HTML Tutorial: Inline & Block Elements | Web Development Tutorials #9

HTML Tutorial: Ids & Classes in HTML | Web Development Tutorials #10

DAY 6

Code Tag and Entities in HTML

HTML Entities Explained | Character Entities In HTML | HTML Tutorial For Beginners | SimpliCode

https://www.codesnail.com/html-entities/ (List of Entities)

DAY 7

Semantic Tabs in HTML

Semantic HTML Tags | HTML5 Semantic Elements Tutorial

DAY 8

Assignment:

Create an odin-recipes folder and cd into it.

Iteration 1: initial structure

1. Within the odin-recipes directory, create an index.html file.
2. Fill it out with the usual boilerplate HTML and add an h1 heading “Odin Recipes” to the body.

Iteration 2: recipe page

1. Create a new directory within the odin-recipes directory and name it recipes.
2. Create a new HTML file within the recipes directory and name it after the recipe it will contain. For example lasagna.html. You can use the name of your favorite dish or, if you need some inspiration, you can find a recipe to use here.
3. For now, just include an h1 heading with the recipe’s name as its content.
4. Back in the index.html file, add a link to the recipe page you just created. Example: Under the <h1>Odin Recipes</h1> heading, write out the link like so: <a href="recipes/recipename.html">Recipe Title</a>. The text of the link should again be the recipe name.

Iteration 3: recipe page content

Your new recipe page should have the following content:

1. An image of the finished dish under the h1 heading that you added earlier. You can find images of the dish on Google or the recipe site we linked to earlier.
2. Under the image, it should have an appropriately sized “Description” heading followed by a paragraph or two describing the recipe.
3. Under the description, add an “Ingredients” heading followed by an unordered list of the ingredients needed for the recipe.
4. Finally, under the ingredients list, add a “Steps” heading followed by an ordered list of the steps needed for making the dish.

Iteration 4: add more recipes

1. Add two more recipes with identical page structures to the recipe page you’ve already created.
2. Don’t forget to link to the new recipes on the index page. Also, consider putting all the links in an unordered list so they aren’t all on one line.

DAY 9

Introduction to CSS

CSS Tutorial: Introduction to CSS | Web Development Tutorials #13

CSS Tutorial: Inline, Internal & External CSS | Web Development Tutorials #14

DAY 10

CSS Selectors

Learn Every CSS Selector In 20 Minutes

https://webdesign.tutsplus.com/the-30-css-selectors-you-must-memorize--net-16048t

DAY 11

Fonts in CSS

CSS Text and Fonts Tutorial for Beginners - Typography

DAY 12

Colors, Borders, and Backgrounds in CSS

CSS Tutorial: Colors In CSS | Web Development Tutorials #18

CSS Tutorial: Borders and Backgrounds | Web Development Tutorials #19

DAY 13

CSS Box Model

CSS "Box Model" in 40 minutes || Complete Web Dev Course || Episode - 15

DAY 14

Styling Links and Buttons in CSS

Styling Links and Buttons in CSS Explained | CSS in English | Web Development

DAY 15

CSS Display Properties

CSS Display Property in Depth: Understanding Inline, Block, and Inline-Block || Episode - 16

DAY 16

All Positions in CSS

Mastering CSS Position Property: Static, Relative, Absolute, Fixed, and Sticky || Episode - 17

DAY 17

CSS Sizes and Responsive Designs

CSS Size Units in Depth: Unlocking the Power of px, %, vw, vh, em, and rem

DAY 18

Introduction to Javascript along with Basics

What is JavaScript?

JavaScript Tutorial for Beginners: Learn JavaScript in 1 Hour

DAY 19

Strings in JavaScript

Strings in JavaScript | Web Development Tutorials #49

String Functions in JavaScript | Web Development Tutorials #50

String Functions in JavaScript | Web Development Tutorials #50

DAY 20

Arrays in JavaScript

JavaScript Arrays

JavaScript Array Filter

JavaScript Array Map

JavaScript Array Reduce

DAY 21

Introduction to Functions in JavaScript along with Basic Interaction Functions

Functions in JavaScript | Web Development Tutorials #53

JavaScript Tutorial: Interaction - Alert, Prompt, Confirm | Web Development Tutorials #54

DAY 22

Loops and Conditional Statements in JavaScript

JavaScript Tutorial: for, while, forEach, Do While Loops | Web Development Tutorials #55

JavaScript if else (tutorial)

DAY 23

Query Selectors in JavaScript and DOM Manipulation

Learn JavaScript DOM Traversal In 15 Minutes

Learn DOM Manipulation In 18 Minutes

DAY 24

Events and Event Listeners in JavaScript

Events in Javascript | chai aur #javascript

DAY 25

Classes, Objects, and JSON in JavaScript

Classes and Objects in JavaScript | JavaScript Tutorial in Hindi #76

JavaScript Tutorial: Working with JSON in JavaScript | Web Development Tutorials #62

END ASSIGNMENT

We’re going to make a simple implementation of grade-school classic “rock paper scissors”.

1. Create a blank HTML document with a script tag (Hint: it is best practice to link an external .js file). This game will be played completely from the console, so don’t worry about putting anything else in there.
2. Your game is going to play against the computer, so begin with a function called getComputerChoice that will randomly return either ‘Rock’, ‘Paper’, or ‘Scissors’. We’ll use this function in the game to make the computer play. Tip: use the console to make sure this is returning the expected output before moving to the next step!
3. Write a function that plays a single round of Rock Paper Scissors. The function should take two parameters - the playerSelection and computerSelection - and then return a string that declares the winner of the round like so: "You Lose! Paper beats Rock"
    - Make your function’s playerSelection parameter case-insensitive (so users can input rock, ROCK, RocK, or any other variation).

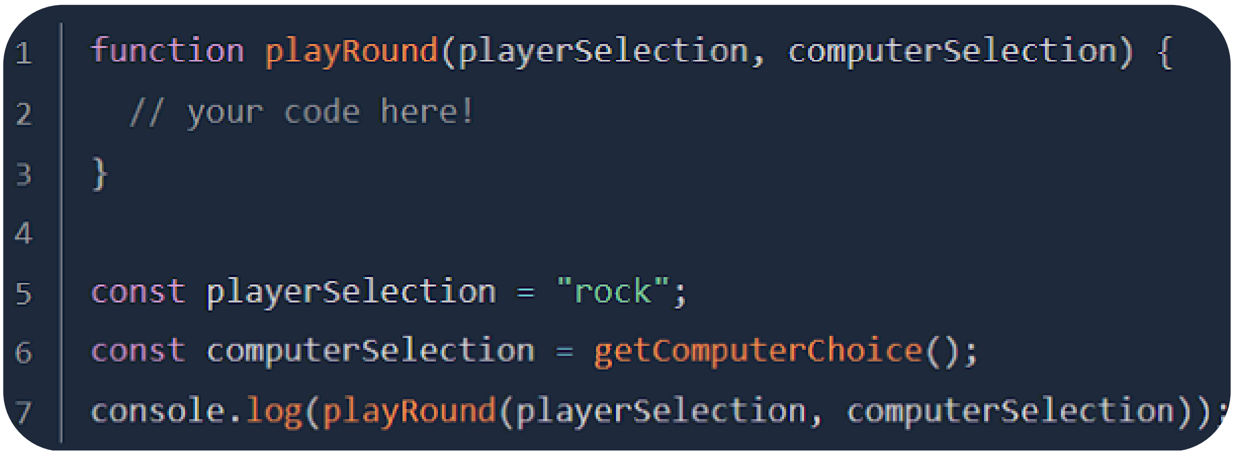
Important note: you want to return the results of this function call, not console.log(). You’re going to use what you return later on, so let’s test this function by using console.log to see the results:

- Write a NEW function called game(). Use the previous function inside of this one to play a 5-round game that keeps score and reports a winner or loser at the end.
    - You have not officially learned how to “loop” over code to repeat function calls… if you already know about loops from somewhere else (or if you feel like doing some more learning) feel free to use them. If not, don’t worry! Just call your playRound function 5 times in a row. Loops are covered in the next lesson.
    - At this point you should be using console.log() to display the results of each round and the winner at the end.
    - Use prompt() to get input from the user. Read the docs here if you need to.
    - Feel free to re-work your previous functions if you need to. Specifically, you might want to change the return value to something more useful.
    - Feel free to create more “helper” functions if you think it would be useful.

      Now we can manipulate the DOM and add a simple UI to it.

- In our UI, the player should be able to play the game by clicking on buttons rather than typing their answer in a prompt.
    - For now, remove the logic that plays exactly five rounds.
    - Create three buttons, one for each selection. Add an event listener to the buttons that call your playRound function with the correct playerSelection every time a button is clicked. (you can keep the console.log for this step)
    - Add a div for displaying results and change all of your console.log into DOM methods.
    - Display the running score, and announce a winner of the game once one player reaches 5 points.

You will likely have to refactor (rework/rewrite) your original code to make it work for this. That’s OK! Reworking old code is an important part of a programmer’s life.



```
function playRound(playerSelection, computerSelection) {
  // your code here!
}

const playerSelection = "rock";
const computerSelection = getComputerChoice();
console.log(playRound(playerSelection, computerSelection));
```