

> _Le Bateau

Created By



Faardheen Khodabuccus
2116103

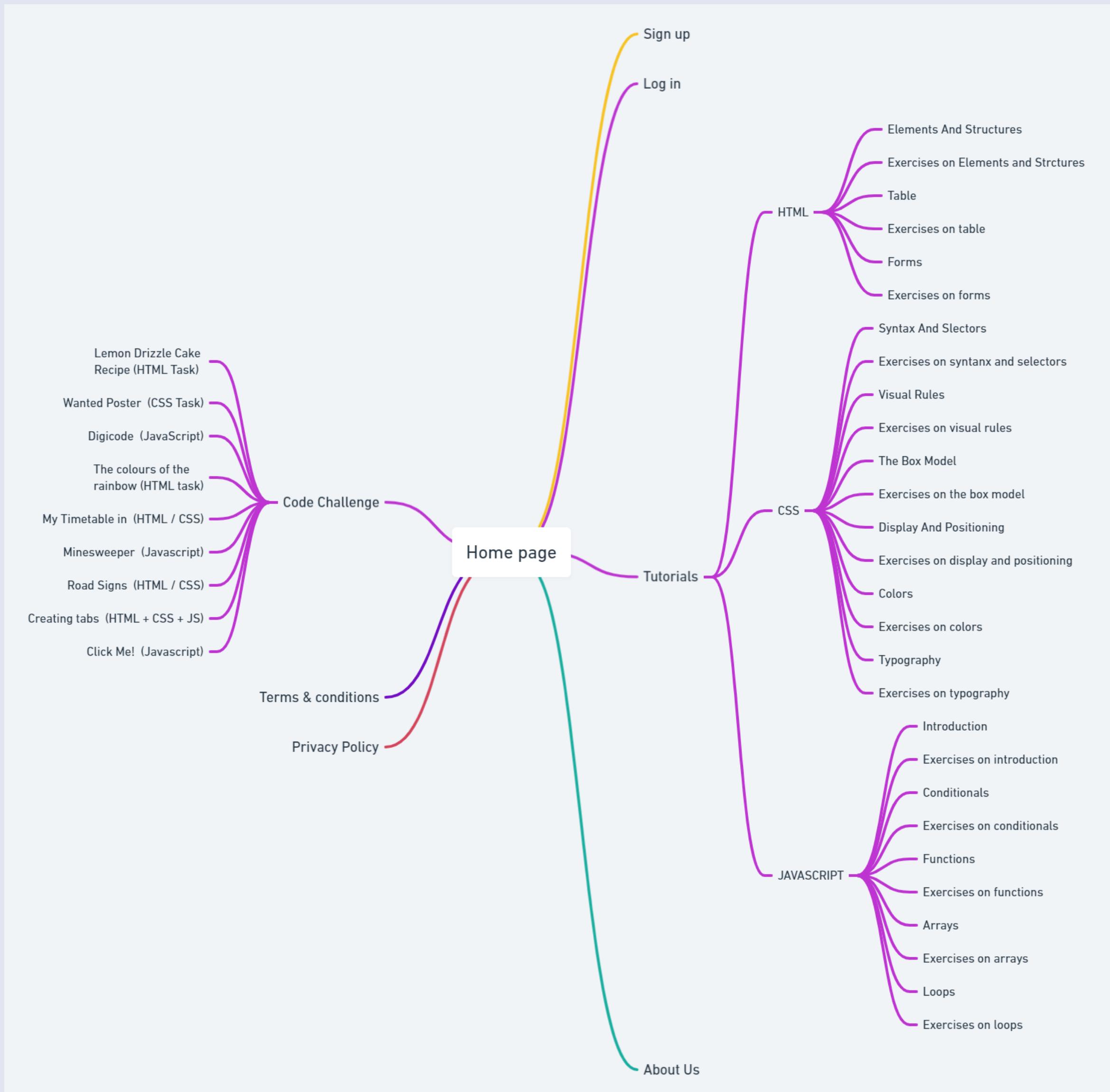


Waliyullah Muntasir
2116321

<https://lebateau.netlify.app/>

TABLE OF CONTENT

- 3 Site Map
- 7 Sign Up Page
- 9 Log In page
- 10 Tutorial page
- 11 About Us page
- 13 Html/css/Javascript Landing page
- 17 Html/css/Javascript Tutorial page
- 18 Html/css/Javascript Exercises page
- 47 Code Challenge page
- 48 Practice Code Challenge page
- 58 Terms And Conditions page
- 59 Privacy Policy page
- 60 Image Resources page



Home Page

Description

- This will be the first page that the user will interact with when viewing our website. First impression is key when a user click on a website. So, we went for modern website design and try to make it as attractive as possible through some colour pop of the buttons and also by adding some pictures.

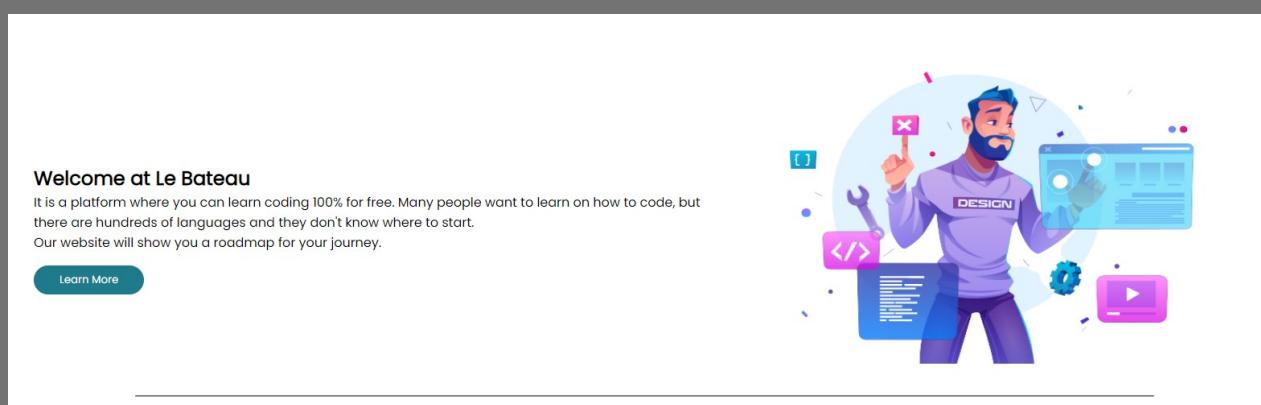
HTML 5 elements used:

- Header
- Nav
- Main
- Section
- Footer

Break-Down

> Le Bateau Tutorials Code Challenges About Us Log in Sign up

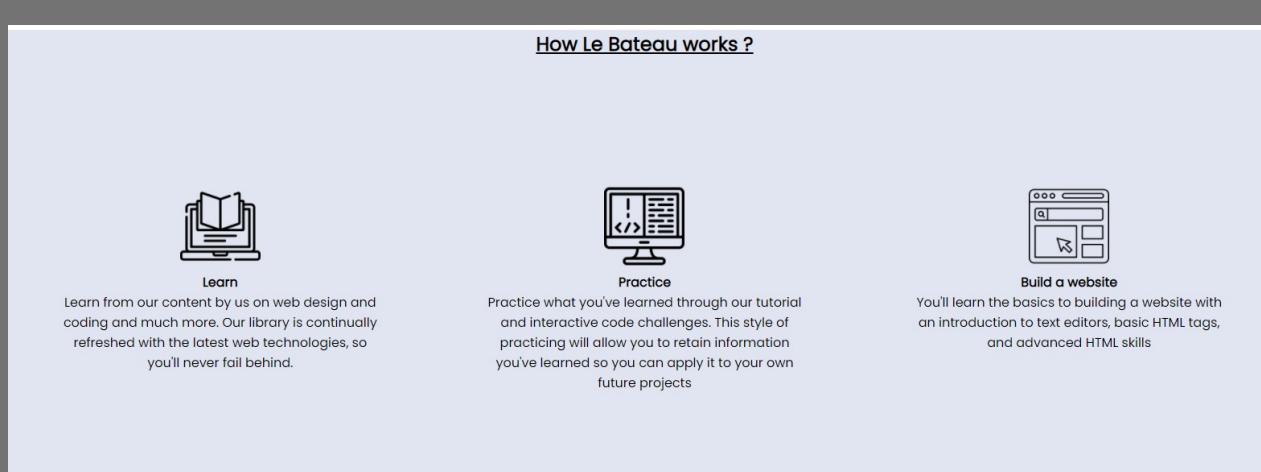
- 3 Divs are used with a class of:
 - nav-container – groups the 2 named divs below.
 - lebateau – contains a span tag.
 - links – which contains the different li tags.
- The div "nav-container" is given a display of "flex".
- The div "lebateau" is after given a style of "flex: 1;" to skew the div "links" to the right.



- A section tag is used.
- 4 Divs are imbedded inside the section with a class of:
 - hero-container – groups the 3 named divs below.
 - welcome – contains an H2 tag along with 2 p tags and a div for the "Learn More" button.
 - banner – contains the image displayed.
 - line – to draw a horizontal line as a separator
- The div "hero-container" is given a display of "flex" to display the contents horizontally, some padding to space the element and a position of relative to position the "Learn More" anchor tag.



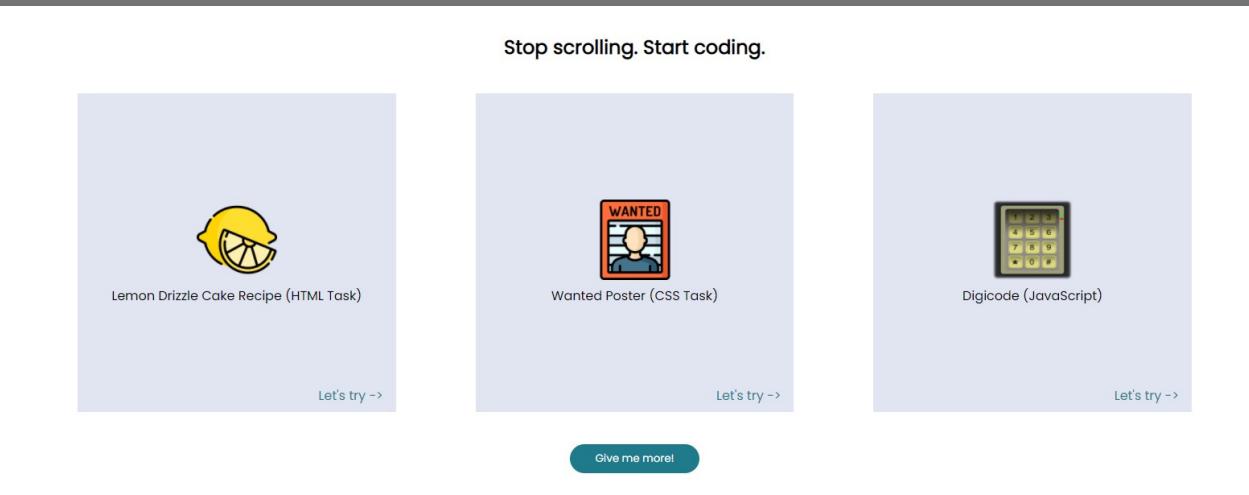
- A main tag is used.
- 4 Divs are imbedded inside the section with a class of:
 - roadmap-container – groups the 3 named divs below.
 - roadmap-txt – contains some p tags and used css to centre the texts.
 - roadmap-img – contains the roadmap image displayed.
 - start-learning btn – for the "Let's start Learning" anchor tag.



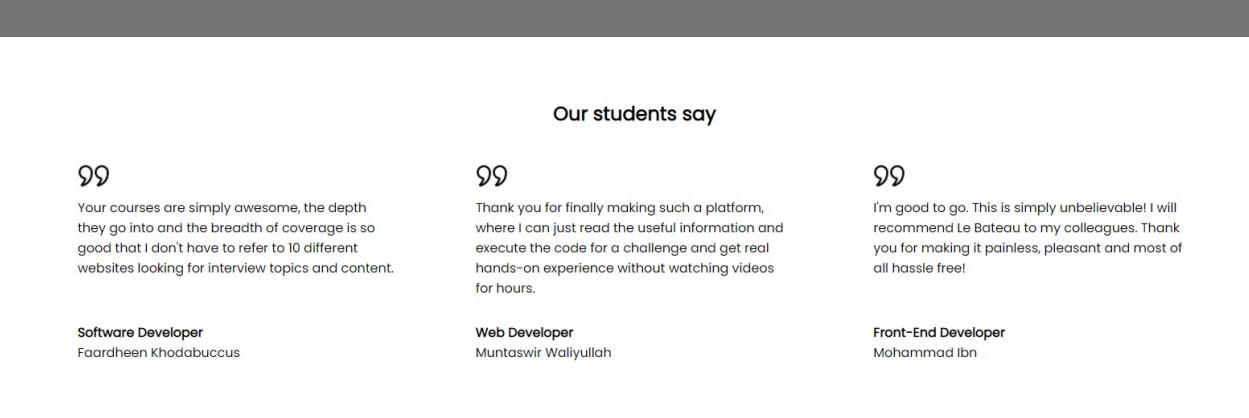
- A section tag with a class of "work" is used.
- 2 Divs are imbedded inside the section with a class of:
 - work-title – contains an H2 tag and given some css style to underline and centre the text.
 - work-container – embeds the divs listed below:
 - learn-container
 - practice-container
 - build-container
- Each of the divs embedded inside work-container consists of 2 divs:
 - i. An image container – that holds the image
 - ii. A description container – that holds the descriptions
- The div work-container is given a display of "flex" so that each embedded divs sits horizontally.
- The embedded divs are given a style property of "text-align: center" to centre everything within the div and also a width of 25%.

"Hands-on coding environments
You don't get better at swimming by watching others.
Coding is no different. Practice as you learn with live code environments inside your browser."

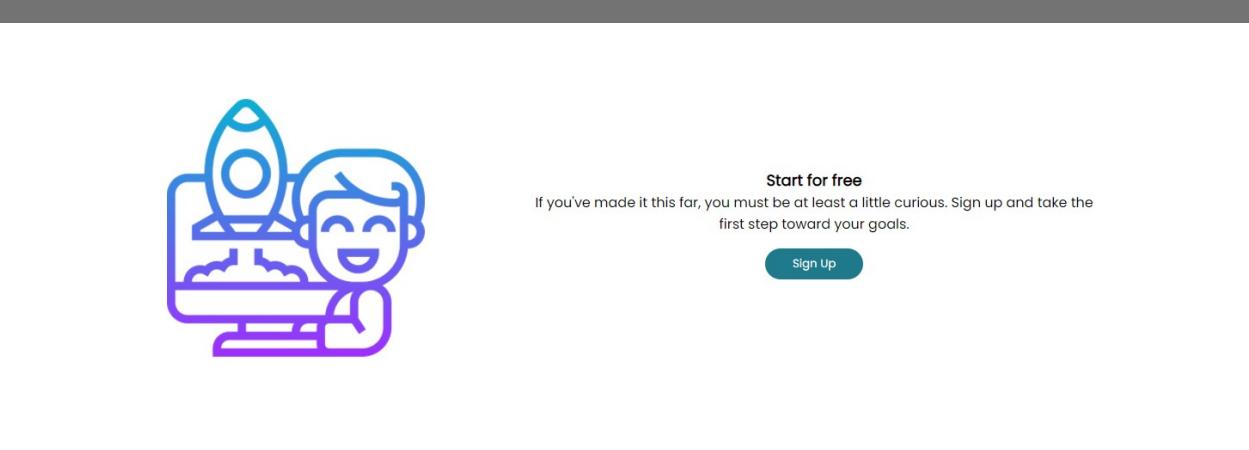
- A section tag with a class of "quote-wrapper" is used with a display of flex.
- A Div is imbedded inside the section with a class of:
 - quote – contains some p tags which are given a style property of "text-align: center" to centre the texts inside the div.



- A section tag with a class of "quiz-wrapper" is used.
- 3 Divs are imbedded inside the section with a class of:
 - quiz-heading – contains an H2 tag and given some css style to centre the text.
 - quiz-container – embeds the divs listed below:
 - lemon
 - wanted-poster
 - poker-card
 - Each of the divs embedded inside work-container consists of 3 divs:
 1. An image container – that holds the image
 2. A description container – that holds the descriptions
 3. A "let's try" container – that holds the anchor tag.
 - quiz-btn – which holds the "Give me More!" button.
- The div work-container is given a display of "flex" so that each embedded divs sits horizontally.
- The embedded divs are also given a display of flex to align everything and also given a position of relative to style the "let's try" button.
- The "let's try" container is given a position of absolute to position itself to the right of its anchor point(that is, its parent container).



- A section tag with a class of "student-corner" is used.
- 2 Divs are imbedded inside the section with a class of:
 - student-heading – contains an H2 tag and given some css style to centre the text.
 - student-container – embeds the divs listed below:
 - first-student
 - second-student
 - third-student
 - Each of the divs embedded inside work-container consists of 3 divs:
 1. An image container – that holds the quote image.
 2. A description container – that holds the descriptions.
 3. A name container – that holds the name of the student.
- The div work-container is given a display of "flex" so that each embedded divs sits horizontally



- A section tag with a class of "sign-up-wrapper" is used.
- A Div is imbedded inside the section with a class of:
 - sign-up-container – embeds 2 divs listed below:
 - sign-up-img – holds an img tag.
 - sign-up-text – holds:
 1. A span tag for the heading given a css property to make the text bold.
 2. A p tag for the descriptions.
 3. A div with a class of "button-container" – that holds the "Sign Up" button.
 - The div sign-up container is given a display of "flex" so that each embedded divs sits horizontally

- The HTML 5 footer tag is used to implement this footer.
- A Div is imbedded inside the footer with a class of:
 - footer-container – embeds:
 - An HTML 5 nav tag with a class of "footer-nav" – holds some li tags.
 - A div with a class of copyright – which holds a span tag.

- The div footer-container is given a display of "flex" so that each embedded elements sits horizontally and vertically within the divs. Also the footer-container is given a position of relative to position the copyright at the bottom.
- The li tags are given a css property of "display: inline-block" to horizontally display the lists.
- The copyright is given a position of absolute to position itself at the bottom of its anchor point(that is, its parent container).

Final Look

The screenshot shows the Le Bateau website. At the top, there's a blue header bar with links for Tutorials, Code Challenges, About Us, Privacy Policy, and Terms & Conditions. Below the header is a footer section with a light blue background. The footer contains a navigation menu with items like "HOME", "LEARN", "CHALLENGES", "PROJECTS", and "BLOG". It also features a "SIGN UP" button and a "LOG IN" button. The main content area below the footer shows a welcome message, a central illustration of a person working on a computer, and a diagram illustrating the stack of technologies used by Le Bateau. The main body of the page includes sections for learning, practicing, and building websites, along with student testimonials and a call-to-action button.

Sign Up Page

Description

- This is the Sign up page where a user can create an account at our website

The screenshot shows a sign-up form titled "Join Le bateau for free!". It has a header "Create Your Account". There are several input fields: "Enter Your First Name", "Enter Your last Name", "Enter Your Email Address", "What Is Your Profession", "Gender: Male/Female/Prefer Not to Say", "Date Of Birth: dd/mm/yyyy", "Street Address", "Apartment, suite etc (optional)", "City", "Postal code", "Enter Your Password" (with a note "*Password must be at least 8 characters long"), "Confirm Your Password", and a checkbox "I agree to the terms and conditions". Below the form are buttons for "Sign In" and "Already A Member? Log In". At the top right, there are links for "Tutorial", "Code Challenge", and "About Us".

HTML

A form tag is used to represents a document section containing interactive controls for submitting information with a name of "accountForm". The form will be process at "account.php". method="POST" is used as data cannot be exposed in the url bar.

15 input field is used:

- First name
- Last name
- Email Address
- Profession
- Gender
- Date of birth
- Street Address
- Apartment number
- City
- State
- Postal Code
- Country
- Password
- Confirm password
- checkbox

These input is implement as we want to create his account and keep a record of the user's :

- Full name
- Email address
- what is his current profession
- His gender
- His age
- Where he lives
- A password to log in to his account of his choice
- a checkbox to agree to our terms and conditions

The "required" attribute is used to make the field mandatory and the form won't submit. An error message will display on submission.

We also used `oninvalid="this.setCustomValidity()"` to display a specific message when the input data is invalid

JavaScript

We used javascript to validate the following:

- To validate if the input is blank
- To validate if first name and last name is between length of 5 to 15 character
- To validate and compare the password
- To check the users age which must be at least 10 years old to register

Regular expression is used:

- Minimum 8 characters - `{8,}`
- At least 1 upper case letter- `(?=.*[A-Z])`
- at least one lower case letter- `(?=.*[a-z])`
- At least one number- `(?=.*\d)`
- At least one special character- `(?=.*[@$!%*?&])`

`[A-Za-z\d@$!%*?&]. The characters listed inside the brackets are part of a matching-character set. For example [abcd] will match a, b, c, d, abcd.`

`^` is used to match the start of the string

`$` is used to match the end of the string

`g` is used to test against all possible matches in a string. `g` is also known global.

Our final that we used is:

`/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}/g`

For the input type of the country, we used an API from <https://restcountries.com/v2/all> where we get all the names of the countries for the drop down.

Sign Up Page

CSS

There is a div with a class called `content-wrapper` where its background has a color of `#e1e5f2`. All the headings in the sign up page is aligned to the center using `text-align: center;` with a padding of 10px.

Inside this div there is another div its class called `form-wrapper`

The `form-wrapper` div :

- is centered
- has a border radius(using `border-radius`)
- a white color background (using `background-color`)

All the input has a border radius but there are 2 different width (290px and 420px)

There is a div with a class "align". it makes the inputs inline using `flexbox(display: flex;)` and using a property called `justify-content` which defines the alignment along the main axis and distribute spaces using `space-between` (`justify-content: space-between;`)

The button is in a div and has a class called "btn". The button is centered using flexbox and with `justify-content: center;`

The button has a:

- dimension of 200px * 45px
- text color is white (`color: white;`)
- background color of `#1f7a8c`
- border radius

We also used `small` tag which displays text in a smaller size.

A div which has a class of "member " and its content is centered using flexbox.

Log in Page

Description

- This is the Log in page where the user can log in to his account

> Le Bateau

Tutorial Code Challenge About Us

Welcome back at Le Bateau

[Log in to your account to continue](#)

Email Address

Password

Log In

Not a member yet? [Sign Up](#)
[Forgot password?](#)

HTML

A form tag is used to represents a document section containing interactive controls for submitting information with a name of "login-form". The form will be process at "login.php". method="POST" is used as data cannot be exposed in the url bar.

2 input field is used:

- Email Address
- Password

The "required" attribute is used to make the field mandatory and the form won't submit. An error message will display on submission.

A span tag is used which defines an inline styleless section in a document

<hr> tag is also used to produced a horozontal line.

Javascript

We used javascript to validate the following:

- To validate if the input is blank

CSS

The is a div with a class called "main". inside this div there are another 2 div with classes called form-group and left-section. The 2 divs is inline using flexbox.

The form-group:

- Inputs:
 - have a dimension of 410px*45px
 - border radius of 5px
 - font size of 18px
- button:
 - font-size of 18px
 - background color of #1f7a8c
 - text color of white
 - dimension of 410px*45px
 - border radius of 5px

The left-section:

- flex-grow set to 1, the remaining space in the container will be distributed equally
- background color of #e1e5f2
- h2 of font size 40px
- h3 of font size 20px
- "le bateau" is in a span tag and the text color is #1f7a8c
- line
 - background color of #1f7a8c
 - dimension of 205px*10px
 - border-radius of 25px

Tutorial Page

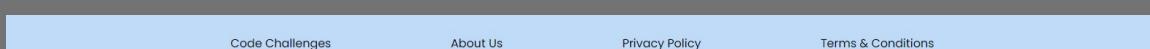
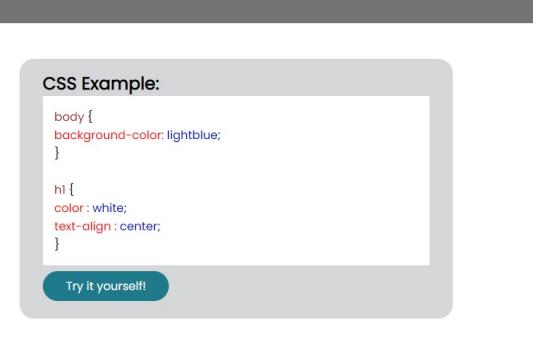
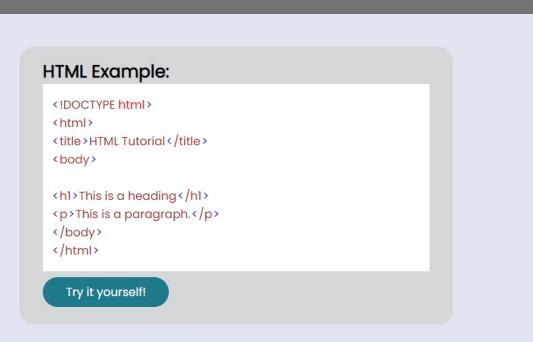
Description

- This is the tutorial landing page. A user can select on the different tutorials; when a user select on a particular topic(for example, HTML tutorial), it would redirects them to the appropriate pages.

Break-Down



- Already explained in the Home page break-down.



- Already explained in the Home page break-down.

HTML

The language for building web pages

[Learn HTML](#)

HTML Example:

```
<!DOCTYPE html>
<html>
<title>HTML Tutorial</title>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```

[Try it yourself!](#)

CSS

The language for styling web pages

[Learn CSS](#)

CSS Example:

```
body {
background-color: lightblue;
}

h1 {
color: white;
text-align: center;
}
```

[Try it yourself!](#)

JavaScript

The language for programming web pages

[Learn CSS](#)

JavaScript Example:

```
<button onClick = "myFunction()">Click Me!</button>

<script>
function myFunction() {
let x = document.getElementById( "demo" );
x.style.fontSize = '25px';
x.style.color = "red";
}

</script>
```

[Try it yourself!](#)[Code Challenges](#)[About Us](#)[Privacy Policy](#)[Terms & Conditions](#)

© 2022 Copyright • Le Bateau

About-us Page

Description

- The about-us page consists of our personal information. We made use of a [cdn](#) from 'font-awesome' to import some icons and when a user clicks on them, it will direct them to our social media accounts.
- For content purposes, we decided to add a little background story to our idea of creating this website.
- You can see more about font-awesome using this link: <https://fontawesome.com/>

Break-Down

- Already explained in the Home page break-down.

"We are all in the same boat when it comes to learning."

Faardheen Khodabuccus
Co-Founder
[f](#) [in](#) [g](#) [e](#)

Muntasir Waliullah
Co-Founder
[f](#) [in](#) [g](#) [e](#)

What inspired us towards this idea ?

The internet can be over whelming sometimes. As students, we've all been through that struggles of looking for some online resources and being completely lost afterward simply because of the loads of informations being thrown at us. Learning web development or programming in general is no different. Due to hundreds if not thousands of different programming languages out there, we're left in total confusion of what should we learn first ?; If we do start from somewhere, what should we do next ?. Although there are thousands resources on the web, none of them specifies a clear guidelines for us to follow along.

Therefore, as 2 passionate students, we've decided to take matters into our own hands. Le Bateau was founded since then to accommodate all the fellow tech lovers out there who has the same interest as we do, with a chance in developing a new skill, levelling up that skill and showing off your creative mindset !

- A section was used.
- 2 Divs are embedded inside the section with a class of:
 - heading - holds an H1 tag.
 - wrapper - holds another 2 divs with a class of:
 - profile wrapper - which consists of several nested divs.
 - about-us - holds a span tags which was used for the background story.

Faardheen Khodabuccus
Co-Founder
[f](#) [in](#) [g](#) [e](#)

Muntasir Waliullah
Co-Founder
[f](#) [in](#) [g](#) [e](#)

- The div "profile-wrapper" hold 2 divs(one for each co-founder) with a class of:
 - profile-picture
 - bio - which holds:
 - Span tags for the name and role(i.e co-founder)
 - Another div with a class of "icons" - which consists of the icons nested in anchor tags.
- Each div for the co-founder was given a display of flex to horizontally align the profile-picture and the bio.

Code Challenges About Us Privacy Policy Terms & Conditions

- Already explained in the Home page break-down.

Final Look

"We are all in the same boat when it comes to learning."



Faardheen Khodabuccus

Co-Founder

[f](#) [in](#) [g](#) [e](#)



Muntasir Waliullah

Co-Founder

[f](#) [in](#) [g](#) [e](#)

What inspired us towards this idea ?

The internet can be over whelming sometimes.

As students, we've all been through that struggles of looking for some online resources and being completely lost afterward simply because of the loads of informations being thrown at us.

Learning web development or programming in general is no different. Due to hundreds if not thousands of different programming languages out there, we're left in total confusion of what should we learn first ?; If we do start from somewhere, what should we do next ?.

Although there are thousands resources on the web, none of them specifies a clear guidelines for us to follow along.

Therefore, as 2 passionate students, we've decided to take matters into our own hands. Le Bateau was founded since then to accommodate all the fellow tech lovers out there who has the same interest as we do, with a chance in developing a new skill, levelling up that skill and showing off your creative mindset !

HTML, CSS & JavaScript tutorial landing page.

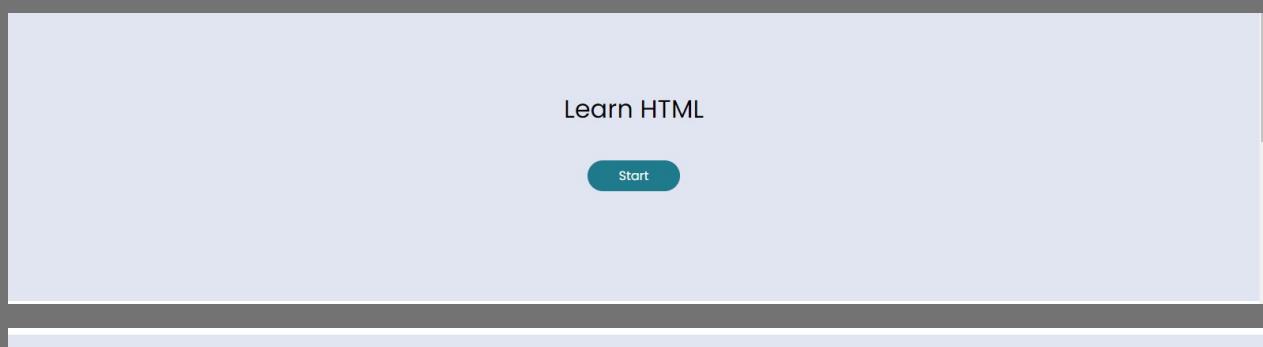
Description

- Note: the 3 tutorial landing pages were build around the same concepts and made use of the same html elements. Therefore, some parts will not be shown in the break down to avoid duplicated slides.
- Those landing pages was build mainly to give users a broad overview of what they be expected to learn.
- The landing pages mainly lists:
 - Describes a basic overview of the course.
 - The different topics that will be covered.
 - The different projects they will be able to build

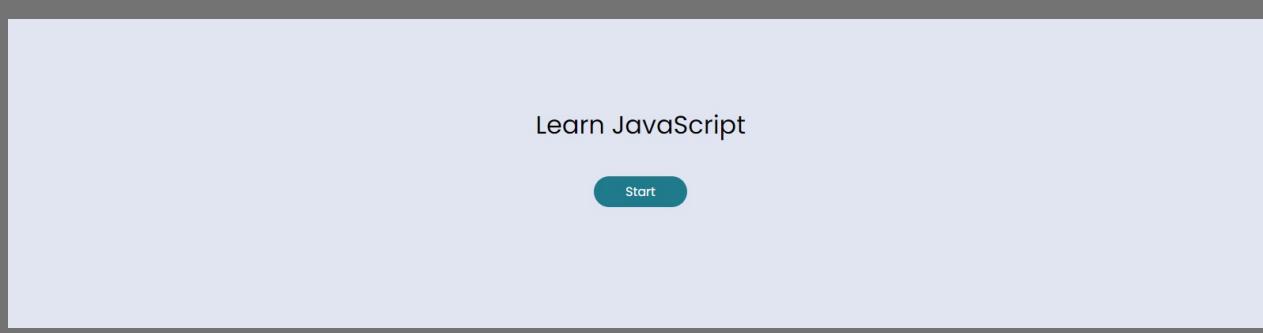
Break-Down

>_Le Bateau Tutorials Code Challenges About Us Log in Sign up

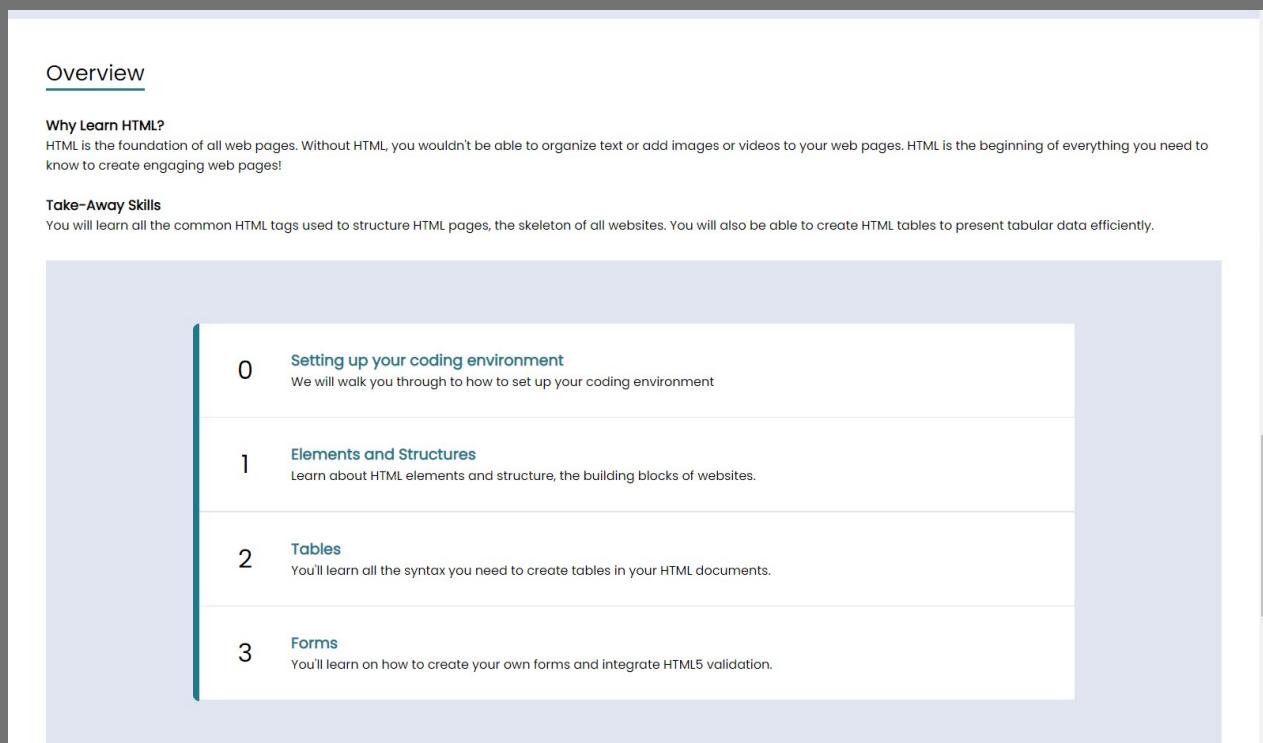
- Already explained in previous break-downs.



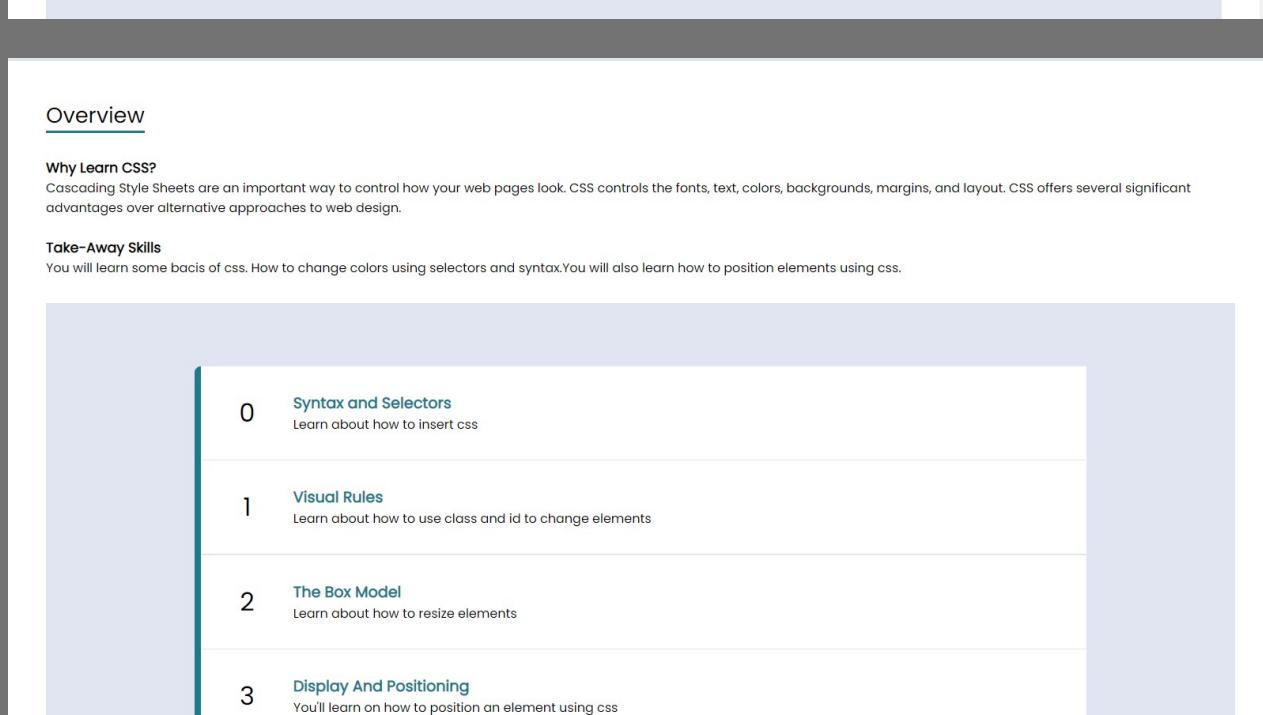
- A section was used.
- A Div is embedded inside the section with a class of "banner" which holds:
 - 2 nested divs where:
 - i.holds a span tag for the heading
 - ii.holds the anchor tag for the button



- CSS flexbox was used to horizontally and vertically align the elements.



- A section with a class of "overview-wrapper" was used.
- The section consists of:
 - A div that holds a span tag which display "Overview"
 - 2 Article Tags where each;
 - one holds:
 - 2 span tag for the heading(i.e: Why learn HTML?)
 - 2 p tag for description
 - The second holds:
 - The syllabus, which consist of an ordered lists
 - The second article was style using CSS flexbox and given a background color.
 - The li tags was given a little bit of padding.



WHAT YOU'LL BE BUILDING.



Let's try ->



Let's try ->



Let's try ->

- Already explained in previous break-downs.

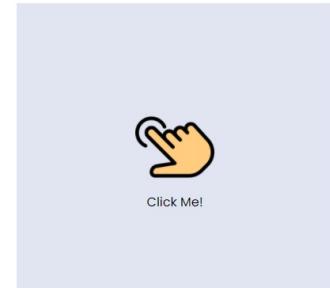
WHAT YOU'LL BE BUILDING.



Let's try ->



Let's try ->



Let's try ->

Code Challenges

About Us

Privacy Policy

Terms & Conditions

© 2022 Copyright • Le Bateau

- Already explained in the previous break-down.

Final Look

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Learn HTML

Start

Overview

Why Learn HTML?
HTML is the foundation of all web pages. Without HTML, you wouldn't be able to organize text or add images or videos to your web pages. HTML is the beginning of everything you need to know to create engaging web pages!

Take-Away Skills
You will learn all the common HTML tags used to structure HTML pages, the skeleton of all websites. You will also be able to create HTML tables to present tabular data efficiently.

0	Setting up your coding environment We will walk you through how to set up your coding environment
1	Elements and Structures Learn about HTML elements and structure, the building blocks of websites.
2	Tables You'll learn all the syntax you need to create tables in your HTML documents.
3	Forms You'll learn on how to create your own forms and integrate HTML5 validation.

WHAT YOU'LL BE BUILDING.



Let's try ->



Let's try ->



Let's try ->

Tutorials

Code Challenges

About Us

Privacy Policy

Terms & Conditions

© 2022 Copyright • Le Bateau

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Learn CSS

Start

Overview

Why Learn CSS?
Cascading Style Sheets are an important way to control how your web pages look. CSS controls the fonts, text, colors, backgrounds, margins, and layout. CSS offers several significant advantages over alternative approaches to web design.

Take-Away Skills
You will learn some basics of css. How to change colors using selectors and syntax. You will also learn how to position elements using css.

0	Syntax and Selectors Learn about how to insert css
1	Visual Rules Learn about how to use class and id to change elements
2	The Box Model Learn about how to resize elements
3	Display And Positioning You'll learn on how to position an element using css
4	Colors You'll learn on how to change colors of a specific element
5	Typography You'll learn on how to change the typography

WHAT YOU'LL BE BUILDING.



The colours of the rainbow (HTML Task)

Let's try ->



Lemon Drizzle Cake Recipe (HTML Task)

Let's try ->



Road Signs (HTML / CSS)

Let's try ->

Tutorials Code Challenges About Us Privacy Policy Terms & Conditions

© 2022 Copyright • Le Bateau

>_Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Learn JavaScript

[Start](#)

Overview

Why Learn JavaScript?
JavaScript is among the most powerful and flexible programming languages of the web. It powers the dynamic behavior on most websites, including this one.

Take-Away Skills
You will learn programming fundamentals and basic object-oriented concepts using the latest JavaScript syntax. The concepts covered in these lessons lay the foundation for using JavaScript in any environment.

- 1 Introduction**
You will learn all about JavaScript Fundamentals.
- 2 Conditionals**
Learn how to use conditional statements.
- 3 Functions**
Learn about JavaScript function syntax, passing data to functions, the return keyword, and concise body syntax.
- 4 Arrays**
You will learn about the fundamentals of arrays.
- 5 Loops**
In this course, you will learn how to use for and while loops to execute blocks of code multiple times.

WHAT YOU'LL BE BUILDING.



Minesweeper

Let's try ->



Poker Card Game

Let's try ->



Click Me!

Let's try ->

Tutorials Code Challenges About Us Privacy Policy Terms & Conditions

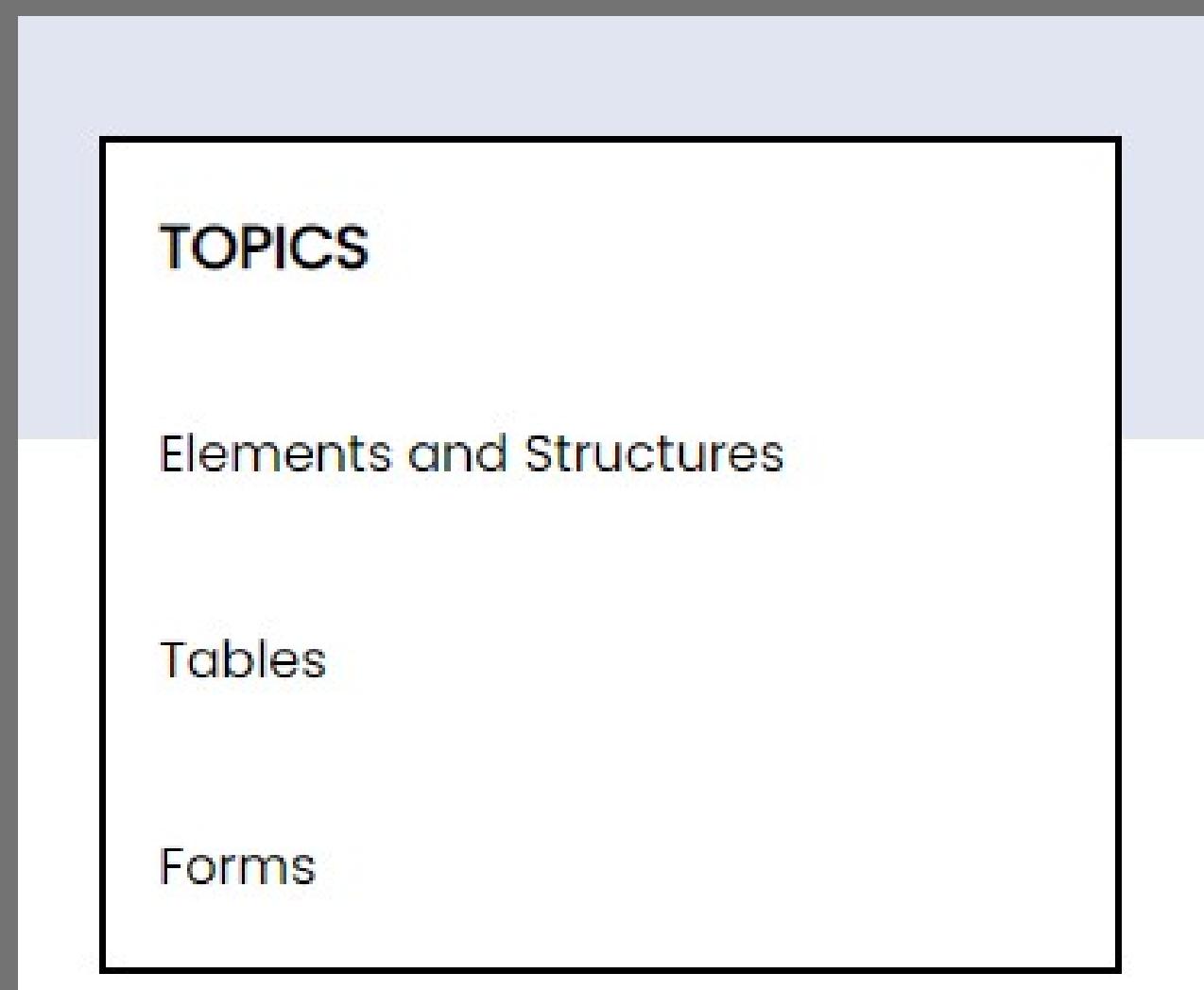
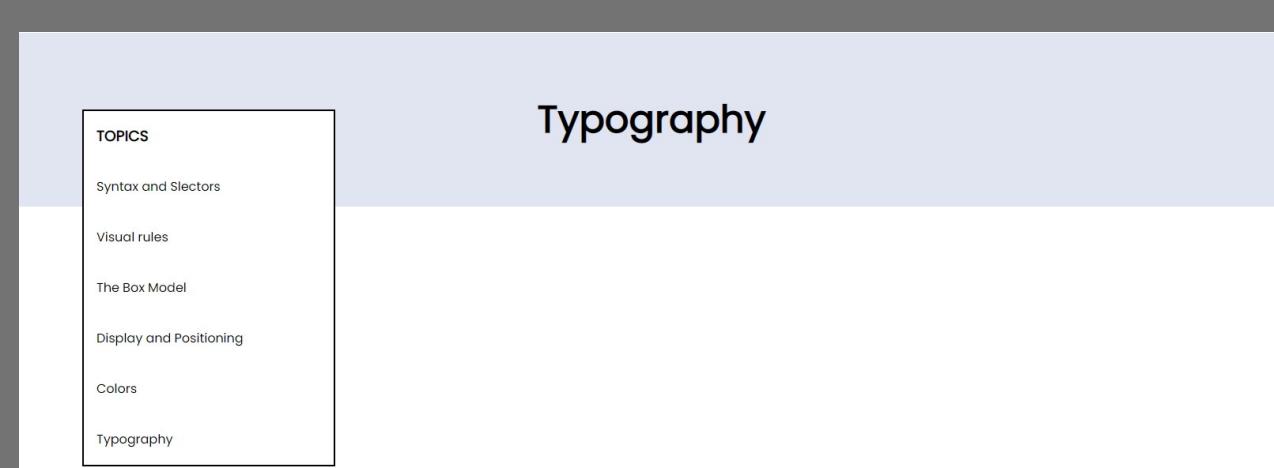
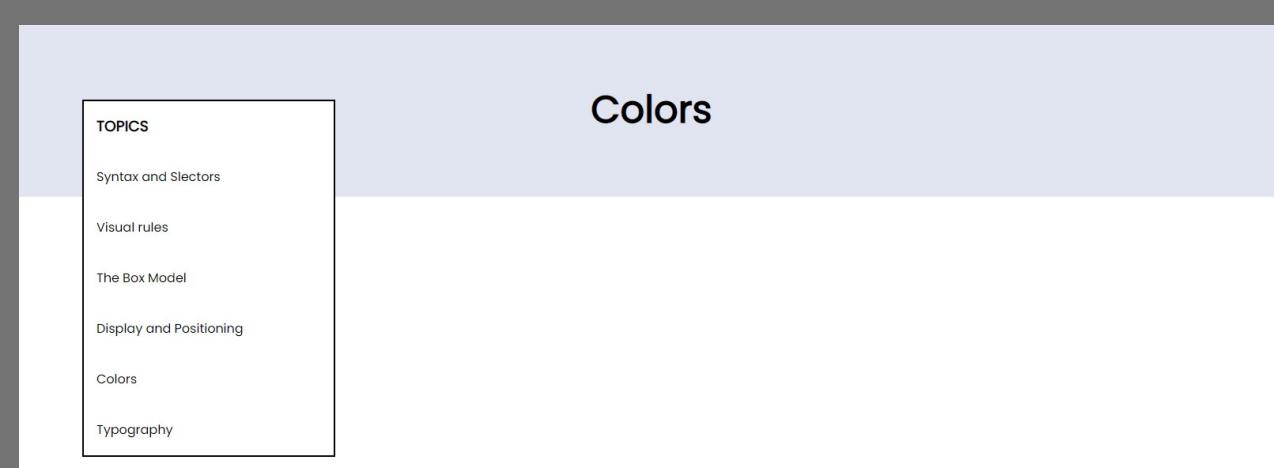
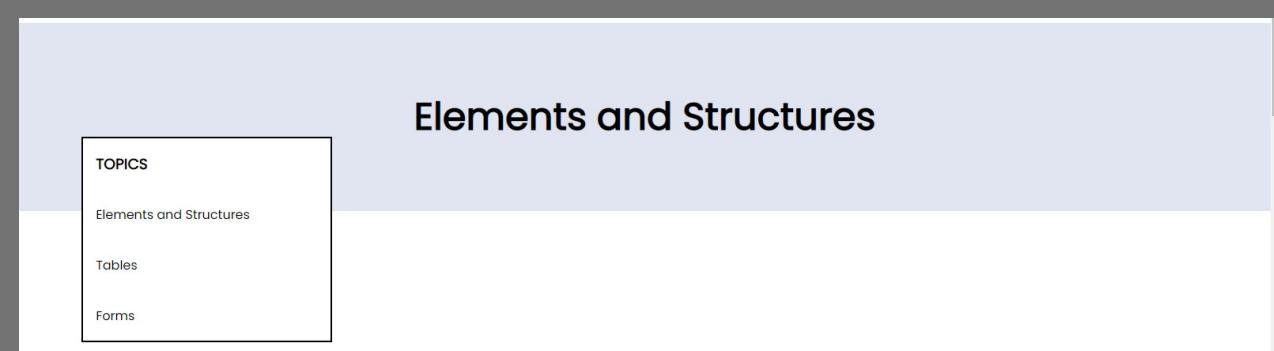
© 2022 Copyright • Le Bateau

HTML, CSS & JavaScript tutorial page.

Description

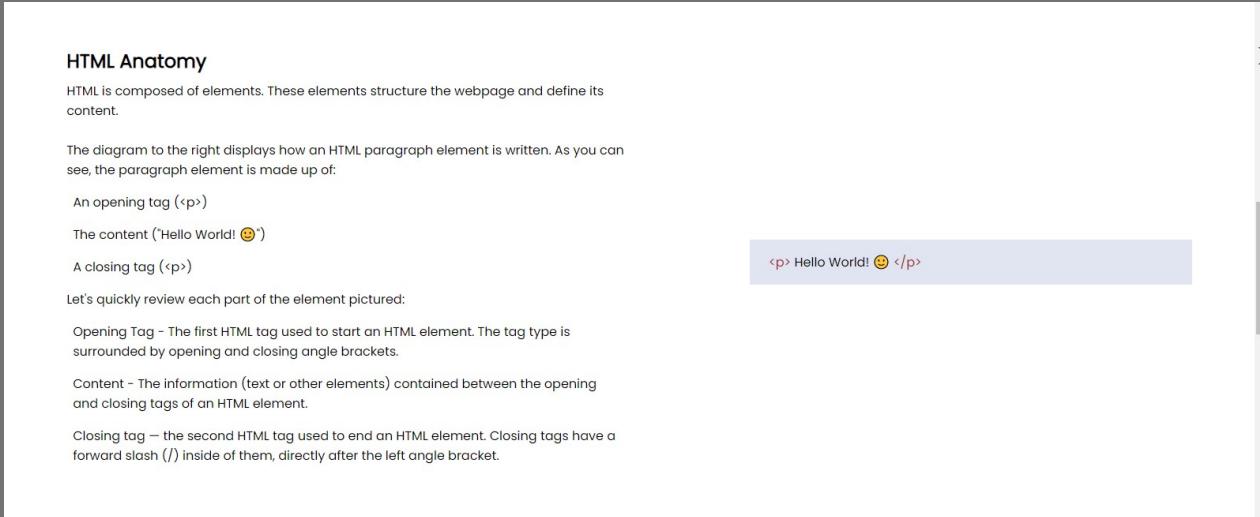
- Note: the 3 tutorial landing pages were build around the same concepts and made use of the same html elements. Therefore, some parts will not be shown in the break down to avoid duplicated slides.
- Those tutorial pages consists of the following:
 - A header(i.e Elements And Structures/Forms/Tables and so on..)
 - A menu which was style given a position of absolute.
 - The main contents
 - Some Quizzes
 - Implemented some javascript to validate the answers of the users.
 - Make use of json file to store the answers

Break-Down



- A section with a class of "header" was used.
- A Div is embedded inside the section with a class of "header-wrapper" which holds:
 - An H1 tag for the heading.
 - A div with a class of menu which holds an unordered lists of anchor tags.
- The header-wrapper was given a position of relative to help position the menu.
- CSS flexbox was used to horizontally and vertically align the elements.

- The Menu was created by using an unordered list of anchor tags
- It was given a width of 20%
- A border of 2px
- A position of absolute
- Positioned to the left by 5% and away from the top by 80%.



- An article tag was used with a display of flex
- 2 Divs are embedded inside the article with a class of:
 - Explanation which holds:
 - An H1 tag - for the title
 - A p that - To describe the title
 - code-snippet which holds:
 - span tags which illustrates the some line of codes
- The class "explanation" was given a max-width of 40% so that the text does not take up the whole screen and leaves space for the code snippet to sit along side the explanation.
- CSS flexbox was used so that the divs are horizontally and vertically align with each other.
- The same concept was then repeated for all the pages.

HTML, CSS & JavaScript Quizzes(Exercises) page.

Description

- Note: the 3 quizzes pages were build around the same concepts and made use of the same html elements and javascript file. Therefore, some parts will not be shown in the break down to avoid duplicated slides.

- The quizzes pages was build to provide users a way to practice their skills.
- It was implemented by the following:
 - Each quizzes is created using a **form** along with an **id** ... ►<form action id="2">...</form> == \$0
 - In the form there are several input fields where the users will input their answers.
 - The answers will then be validated using a javascript.
 - Each topics have their json files where the answers are stored in accordance to their form's id
 - When the user inputs their answers and click on the "Submit" button, a function will fire.
 - The function will grab the id of the form, query the value of each input fields where it will check if the value entered corresponds to the answer in the json file at the index of the form's id (for instance, if the form's id is 2, it will check the answers which are in json file at index 2).
 - Following that, if the user's answers are correct, a 'correct template' of colour green along with a 'checkmark' icon which is **animated** will be displayed.
 - On the other hand, if the answers are wrong, a 'wrong template' of colour red along with a 'cross' icon will be displayed.

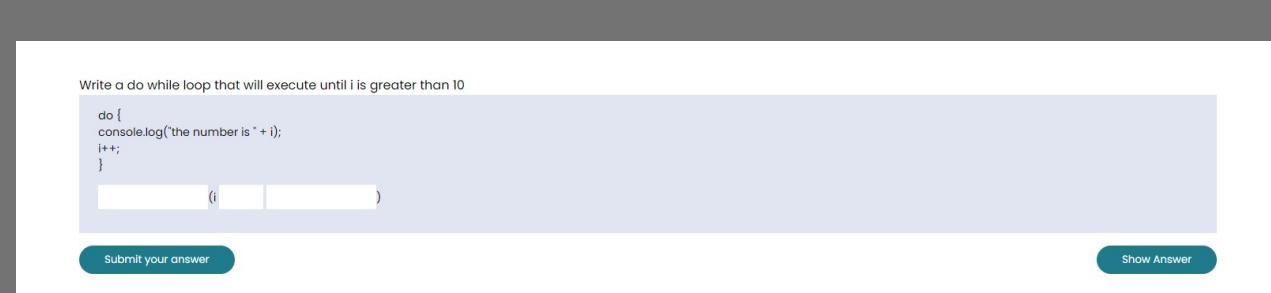


- A form is created.
- A Div is embedded inside the form with a class of "code-snippet-wrapper" which holds:
 - 4 Divs with a class of:

- correct - Holds the 'checkmark' icon and a span tag.
 - initially, the div is hidden from the DOM. Only when the answers entered corresponds to the values in the json file that the div will be displayed to the DOM.
- wrong - Holds the 'cross' icon and a span tag.
 - initially, the div is hidden from the DOM. Only when the answers entered does not corresponds to the values in the json file that the div will be displayed to the DOM.
- code-snippet - holds the input fields.
- btn-wrapper - holds the "Submit" and "Show Answer" button.



- The code-snippet was given a background-color.
- CSS flexbox was used to space the buttons in the form.



Final Look - HTML Tutorials

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Elements and Structures

TOPICS
Elements and Structures
Tables
Forms

Introduction to HTML

HTML(Hypertext Markup Language) is used to give content to a web page and instructs web browsers on how to structure that content. Learning HTML is the first step in creating websites. As we progress throughout this learning journey, we will show you how you can layer HTML with CSS and JavaScript to create aesthetically pleasing and dynamic websites.

HTML Anatomy

HTML is composed of elements. These elements structure the webpage and define its content.

The diagram to the right displays how an HTML paragraph element is written. As you can see, the paragraph element is made up of:

An opening tag (<p>)

The content ("Hello World! 😊")

A closing tag (</p>)

```
<p> Hello World! 😊 </p>
```

Let's quickly review each part of the element pictured:

Opening Tag - The first HTML tag used to start an HTML element. The tag type is surrounded by opening and closing angle brackets.

Content - The information (text or other elements) contained between the opening and closing tags of an HTML element.

Closing tag – the second HTML tag used to end an HTML element. Closing tags have a forward slash (/) inside of them, directly after the left angle bracket.

Document Type Declaration

The document type declaration, <!DOCTYPE html> is required as the first line of an HTML document. It is an instruction to the browser about what document type to expect and which version of HTML is being used, in our case, it's HTML5.

```
<!DOCTYPE html>
```

<html>HTML Element

The <html> element, is the root of an HTML document. It should always be added after the <!DOCTYPE> declaration. All the content/structure of an HTML document should be contained between the <html> tags.

```
<!DOCTYPE html>
<html>
<h1> Coding is fun </h1>
</html>
```

<head>Head Element

The <head> element contains general information about an HTML page that isn't displayed on the page itself. These informations are called as 'metadata'. The head usually includes things like the title of the HTML document and links to external stylesheets

```
<!DOCTYPE html>
<html>
<head>
<!-- Metadata is stored in this element-->
</head>
</html>
```

<title>Title Element

The <title> element contains the title of an HTML document. The title is displayed in the browser's title bar or tab in which the HTML page is displayed. The <title> element can only be contained inside a document's <head> element.

```
<!DOCTYPE html>
<head>
<title> Page Title </title>
</head>
```

The Body Element

The <body> element represents the content of an HTML document. Content inside <body> tags are rendered on the web browsers.

Note: There can only be one element in a document

```
<body>
<h1> Learn coding with Le Bateau. </h1>
</body>
```

An activity for you!

Fill in the missing keywords in the code to your right. ↗

```
<body>
    Learn coding with Le Bateau. </h1>

```

Submit your answer

Show Answer

<h1>-<h6> Heading Elements

There exists 6 levels of heading elements in HTML. The highest level being the <h1> to the lowest level being the <h6>.

```
<h1> This is an h1 Heading </h1>
<h2> This is an h2 Heading </h2>
<h3> This is an h3 Heading </h3>
...
<h6> This is an h6 Heading </h6>
```

<div>Div Element

The <div> element is used as a container that divides an HTML document into sections. Its main role is to group various HTML tags which could then be styled by applying CSS on those group of elements. <div> elements can contain flow content such as headings, paragraphs, links, images, etc.

```
<div>
    <h1> This is a Heading </h1>
    <p> What's up, doc? 😊 </p>
</div>
```

HTML Attributes

HTML attributes are used to provide additional information about HTML elements. The attribute values are added to the opening tag of an element to configure the element or change the element's default behaviour. In our example, we are giving the <p> (paragraph) element an unique identifier using the id attribute that could be referred to a CSS stylesheet for styling.

```
<p id="my-para"> Yo fellas!, Happy Learning. 😊 </p>
```

Final Look - Elements and Structures Exercises

> Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

Elements and Structures
Tables
Forms

Use the correct HTML tag to add a heading with the text "London".

<p>London is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.</p>

Submit your answer

Show Answer

Add six headings to the document with the text "Hello".

Start with the most important heading (the largest) and end with the least important heading (the smallest).

```
<html>
<body>
```



```
</body>
</html>
```

Submit your answer

Show Answer

Mark up the text with appropriate tags:

"Universal Studios Presents" is the most important heading.

"Jurassic Park" is the next most important heading.

"About" is the third most important heading.

The last sentence is just a paragraph.

Start with the most important heading (the largest) and end with the least important heading (the smallest).

 Univers Studio Presents
 Jurassic Park
 About
 On the Island of Isla Nublar, a new park has been built: Jurassic Park is a theme park of cloned dinosaurs!!

Submit your answer

Show Answer

Tables

TOPICS

Elements and Structures
Tables
Forms

<tr> Table Row Element

The table row element, <tr>, is used to add rows to a table before adding table data and table headings.

```
<table>
<tr>
...
</tr>
</table>
```

<td> Table Data Element

The table data element, <td>, can be nested inside a table row element, <tr>, to add a cell of data to a table.

```
<table>
<tr>
<td> cell one data </td>
</tr>
</table>
```

<thead> Table Head Element

The table head element, <thead>, defines the headings of columns encapsulated in the tables rows of the table.

```
<table>
<thead>
<tr>
<th> heading 1 </th>
<th> heading 2 </th> </tr>
</thead>
<tbody>
<tr>
<td> data 1 </td>
<td> data 2 </td>
</tr>
</tbody>
</table>
```

<th> Table Heading Element

The table heading element, <th>, is used to add titles to rows and columns of a table and must be enclosed in a table row element, <tr>.

```
<table>
<tr>
<th> column 1 </th>
<th> column 2 </th>
</tr>
<tr>
<td> 1 </td>
<td> 2 </td>
</tr>
</table>
```

<tbody> Table Body Element

The table body element, <tbody>, will contains all table row <tr> elements, and indicated that </tr> elements make up the body of the table.

```
<table>
<tbody>
<tr>
<td> row 1 </td>
</tr>
</tbody>
</table>
```

The rowspan Attribute

The <rowspan>attribute on a table header or table data element indicates how many rows that particular cell should span within the table. The <rowspan> value is set to 1 by default and will take any positive integer up to 65534.

```
<table>
<tr>
<th> Lecture </th>
<th> Lab </th>
</tr>
<td rowspan=2> Monday 8:30-10:30 </td>
<tr>
<td> Tuesday 13:00-14:00 </td>
</tr>
</table>
```

The colspan Attribute

Similar to <rowspan>, The <colspan> attribute indicates how many columns that particular cell should span within the table. The <colspan> value is set to 1 by default and will take any positive integer between 1 and 1000.

```
<table>
<tr>
<th> Lecture </th>
<th> Lab </th>
</tr>
<tr>
<td colspan=2> Monday 8:30-10:30 </td>
<td> Tuesday 13:00-14:00 </td>
</tr>
</table>
```

Final Look - Table Exercises

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Elements and Structures
- Tables
- Forms

Add a table row with two table headers.
The two table headers should have the value "Name" and "Age".

```
<table>
  
  
  
  
<tr>
<td>Jill Smith</td>
<td>50</td>
</tr>
</table>
```

[Submit your answer](#)

[Show Answer](#)

Add a table row at the end of the table, with two table cells.
The two table cells should have the value "Eve Jackson" and "94".

```
<table>
<tr>
<th>Name</th>
<th>Age</th>
</tr>
<tr>
<td>Jill Smith</td>
<td>50</td>
</tr>



</table>
```

[Submit your answer](#)

[Show Answer](#)

```
<table>
<tr>
<th> Name</th>
<th>Age</th>
</tr>
<tr>
<td>Jill Smith</td>
<td>50</td>
</tr>
<tr>
<td>Eve Smith</td>
<td>45</td>
</tr> </table>
```

[Submit your answer](#)

[Show Answer](#)

Forms

TOPICS

Elements and Structures
Tables
Forms

<form> Element

The HTML <form> element is used to collect and send information to an external source.

<form> can contain various input elements. When a user submits the form, information in these input elements is passed to the source which is named in the <action> attribute of the form.

```
<form>
...
</form>
```

<input> Element

The HTML <input> element is used to render a variety of input fields on a webpage including text fields, checkboxes, buttons, etc.

<input> element have a <type> attribute that determines how it gets rendered to a page.

The example code block will create a text input field on a webpage.

```
<label for="fname">First name:</label>
<input type="text" name="fname" id="fname" >
```

<label> Element

The HTML <label> element provides identification for a specific <input> based on matching values of the <input>'s <id> attribute and the <label>'s for attribute.

```
<label for="password">Password:</label>
<input type="text" name="password" id="password" >
```

<input>: Text Type

HTML <input> elements can support text input by setting the attribute type="text". This renders a single row input field that users can type text inside.

The value od the <input>'s name and value attribute of the element are sent as key-value pair when the form is submitted.

The example code block will create a text input field on a webpage.

```
<input type="text" name="username" >
```

<input>: Number Type

HTML input elements can be of type number. These input field allow the user to enter only numbers inside the field.

The example code block shows an input with a type of number and a name of balance.

```
<input type="number" name="balance" >
```

<input>: Checkbox Type

When using an HTML input element, the type="checkbox" attribute will render a single checkbox item. To create a group of checkboxes related to the same topic, they should all use the same name attribute. Since it's a checkbox, multiple checkboxes can be selected for the same topic.

```
<input type="checkbox" name="breakfast" value="waffle" > Waffle ☑
<input type="checkbox" name="breakfast" value="honey" > Honey ☑
<input type="checkbox" name="breakfast" value="coffee" > Coffee ☑
```

<input>: Number Type

HTML input elements can be of type number. These input field allow the user to enter only numbers inside the field.

The example code block shows an input with a type of number and a name of balance.

```
<input type="number" name="balance" >
```

<input>: Radio Button Type

HTML input element, can be given a type="radio" attribute. Radio buttons let a user to select ONE of a limited number of choices.

The code block shown to you allows a user to select their delivery options. The user is allowed to select only one of the option.

```
<input type="radio" name="delivery-option" value="pickup" >
<input type="radio" name="delivery-option" value="delivery" >
```

An activity for you!

In the form below, add an input field with the type "button" and the value "OK". ☕

```
<form>
<input type="button" value="OK" />
</form>
```

Submit your answer Show Answer

Final Look - Form Exercises

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Elements and Structures
- Tables
- Forms

In the form below, add two radio buttons, both with the name "fav_language".

```
<form>
 value = "html"> HTML
 value = "css"> CSS
</form>
```

[Submit your answer](#)

[Show Answer](#)

In the form below, add the following:

An input of type text with a label for "name"

An input of type number with a label for "balance"

```
<form>


</form>
```

[Submit your answer](#)

[Show Answer](#)

In the form below, add the a <button> element with the text "Click Me".

```
<form action="/action_page.php">

</form>
```

[Submit your answer](#)

[Show Answer](#)

Final Look - CSS Tutorials

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Syntax And Selectors

TOPICS

- Syntax and Selectors
- Visual rules
- The Box Model
- Display and Positioning
- Colors
- Typography

<link> Link Element

The <link> element is used to link HTML documents to external resources like CSS files. It commonly uses:

- href attribute to specify the URL to the external resource
- rel attribute to specify the relationship of the linked document to the current document
- type attribute to define the type of content being linked

```
<!-- How to link an external stylesheet with href, rel, and type attributes -->
<link href="./path/to/stylesheet/style.css" rel="stylesheet" type="text/css" />
```

Purpose of CSS

CSS, or Cascading Style Sheets, is a language that is used in combination with HTML that customizes how HTML elements will appear. CSS can define styles and change the layout and design of a sheet.

Write CSS in Separate Files

CSS code can be written in its own files to keep it separate from the HTML code. The extension for CSS files is .css. These can be linked to an HTML file using a <link> tag in the <head> section

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

Write CSS in HTML File

CSS code can be written in an HTML file by enclosing the code in <style> tags. Code surrounded by <style> tags will be interpreted as CSS syntax.

```
<head>
<style>
h1 {
color: blue;
}
</style>
</head>
```

```
<h2 style="text-align: center;"> Centered text </h2>
<p style="color: blue; font-size: 18px;"> Blue, 18-point text </p>
```

Inline Styles

CSS styles can be directly added to HTML elements by using the style attribute in the element's opening tag. Each style declaration is ended with a semicolon. Styles added in this manner are known as inline styles

Separating HTML code from CSS code

It is common practice to separate content code in HTML files from styling code in CSS files. This can help make the code easier to maintain, by keeping the syntax for each file separate, and any changes to the content or styling can be made in their respective files.

```
/* Selects all elements with class="column" */
.column {

}

/* Selects element with id="first-item" */
#first-item {
```

Class and ID Selectors

CSS classes can be reusable and applied to many elements. Class selectors are denoted with a period . followed by the class name. CSS ID selectors should be unique and used to style only a single element. ID selectors are denoted with a hash sign # followed by the id name.

```
h1, h2 {
color: red;
}
```

```
/* Selects all <p> tags */
p {
```

Groups of CSS Selectors

Match multiple selectors to the same CSS rule, using a comma-separated list. In this example, the text for both h1 and h2 is set to red.

```
.calender-cell {
color: #fff;
```

CSS Type Selectors

CSS type selectors are used to match all elements of a given type or tag name. Unlike for HTML syntax, we do not include the angle brackets when using type selectors for tag names. When using type selectors, elements are matched regardless of their nesting level in the HTML.

```
#job-title{
font-weight: bold;
```

CSS class selectors

The CSS class selector matches elements based on the contents of their class attribute. For selecting elements having calendar-cell as the value of the class attribute, a . needs to be prepended.

```
<div class="value1 value2 value3"></div>
```

HTML attributes with multiple values

Some HTML attributes can have multiple attribute values. Multiple attribute values are separated by a space between each attribute

CSS ID selectors

The CSS ID selector matches elements based on the contents of their id attribute. The values of id attribute should be unique in the entire DOM. For selecting the element having job-title as the value of the id attribute, a # needs to be prepended.

```
<style>
{
color: red;
}
</style>
```

An activity for you!

Fill in the missing keywords in the code to your right. 🚀
Set the color of all <p> elements to red.

Submit your answer Show Answer

Final Look - Syntax And Selectors Exercises

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Syntax and Selectors
- Visual rules
- The Box Model
- Display and Positioning
- Colors
- Typography

Set the color of the element with id="para1", to red.

```
<style>
  #para1 {
    color: red;
  }
</style>
<body>
  <h1>This is a heading</h1>
  <p id="para1">This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Set the color of all elements with the class colortext, to red.

```
<style>
  .colortext {
    color: red;
  }
</style>
<body>
  <h1>This is a heading</h1>
  <p>This is a paragraph</p>
  <p class="colortext">This is a paragraph</p>
  <p class="colortext">This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Change the color of all `<p>` and `<h1>` elements, to "red". Group the selectors to minimize code.

```
<style>
  h1, p {
    color: red;
  }
</style>
<body>
  <h1>This is a heading</h1>
  <h2>This is a smaller heading</h2>
  <p>This is a paragraph</p>
  <p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Visual Rules

TOPICS
Syntax and Selectors
Visual rules
The Box Model
Display and Positioning
Colors
Typography

CSS declarations

In CSS, a declaration is the key-value pair of a CSS property and its value. CSS declarations are used to set style properties and construct rules to apply to individual or groups of elements. The property name and value are separated by a colon, and the entire declaration must be terminated by a semi-colon.

```
/* CSS declaration format: property-name: value; */  
/* CSS declarations */  
text-align: center;  
color: purple;  
width: 100px;
```

Purpose of CSS

CSS, or Cascading Style Sheets, is a language that is used in combination with HTML that customizes how HTML elements will appear. CSS can define styles and change the layout and design of a sheet.

Font Size

The font-size CSS property is used to set text sizes. Font size values can be many different units or types such as pixels, section

```
font-size: 30px;
```

Background Color

The background-color CSS property controls the background color of elements.

```
background-color: blue;
```

Opacity

The opacity CSS property can be used to control the transparency of an element. The value of this property ranges from 0 (transparent) to 1 (opaque).

```
opacity: 0.5;
```

Font Weight

The font-weight CSS property can be used to set the weight (boldness) of text. The provided value can be a keyword such as bold or normal.

```
font-weight: bold;
```

Text Align

The text-align CSS property can be used to set the text alignment of inline contents. This property can be set to these values: left, right, or center.

```
text-align: right;
```

CSS Rule Sets

A CSS rule set contains one or more selectors and one or more declarations. The selector(s), which in this example is h1, points to an HTML element. The declaration(s), which in this example are color: blue and text-align: center style the element with a property and value. The rule set is the main building block of a CSS sheet.

```
h1 {  
color: red;  
text-align: center;  
}
```

Resource URLs

In CSS, the url() function is used to wrap resource URLs. These can be applied to several properties such as the background-image.

```
background-image: url( "../resources/image.png" );
```

Font-family

The font-family CSS property is used to specify the typeface in a rule set. Fonts must be available to the browser to display correctly, either on the computer or linked as a web font. If a font value is not available, browsers will display their default font. When using a multi-word font name, it is best practice to wrap them in quotes.

```
h1 {  
font-family: Verdana;  
}  
  
#page-title {  
font-family: "Courier New";  
}
```

Color Name Keywords

Color name keywords can be used to set color property values for elements in CSS.

```
h1 {  
color: aqua;  
}  
  
li {  
color: khaki;  
}
```

An activity for you!

Fill in the missing keywords in the code to your right. ↗

Set the background color to #6aa305

```
<style>  
[ ] : [ ];  
</style>
```

Submit your answer

Show Answer

Final Look - Visual Rules Exercises

> Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Syntax and Selectors
- Visual rules
- The Box Model
- Display and Positioning
- Colors
- Typography

Use the correct background property to make the background image NOT scroll with the rest of the page.

```
<style>
body {
background-image: url("img_tree.png");
[REDACTED]
}
```

</style>

[Submit your answer](#)

[Show Answer](#)

Use CSS to set the transparency of the image to 50%.

```
<style>
img {
[REDACTED]
}
</style>

<body>

</body>
```

[Submit your answer](#)

[Show Answer](#)

Use the correct font- property to style the `p` elements as "small-caps".

```
<style>
p {
background-color: rgb(255, 0, 0);
[REDACTED]
}
</style>

<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

The Box Model

TOPICS
Syntax and Selectors
Visual rules
The Box Model
Display and Positioning
Colors
Typography

CSS Margin Collapse

CSS margin collapse occurs when the top and bottom margins of blocks are combined into a single margin equal to the largest individual block margin.
Margin collapse only occurs with vertical margins, not for horizontal margins.

```
/* The vertical margins will collapse to 30 pixels instead of
adding to 50 pixels. */
.block-one {
margin: 20px;
}

.block-two {
margin: 30px;
}
```

CSS auto keyword

The value auto can be used with the property margin to horizontally center an element within its container. The margin property will take the width of the element and will split the rest of the space equally between the left and right margins.

```
div {
margin: auto;
}
```

Write CSS in Separate Files

If content is too large for its container, the CSS overflow property will determine how the browser handles the problem. section
By default, it will be set to visible and the content will take up extra space. It can also be set to hidden, or to scroll, which will make the overflowing content accessible via scroll bars within the original container.

```
small-block {
overflow: scroll;
}
```

Height and Width Maximums/Minimums

The CSS min-width and min-height properties can be used to set a minimum width and minimum height of an element's box. CSS max-width and max-height properties can be used to set maximum widths and heights for element boxes.

```
/* Any element with class "column" will be at most 200 pixels
wide, despite the width property value of 500 pixels. */
.column {
max-width: 200px;
width: 500px;
}
```

The property box-sizing of CSS box model

CSS styles can be directly added to HTML elements by using the style attribute in the element's opening tag. Each style declaration is ended with a semicolon. Styles added in this manner are known as inline styles

```
.container {
box-sizing: border-box;
}
```

CSS box-sizing: border-box

The value border-box of the box-sizing property for an element corresponds directly to the element's total rendered size, including padding and border with the height and width properties.
The default value of the border-box property is content-box. The value border-box is recommended when it is necessary to resize the padding and border but not just the content. For instance, the value border-box calculates an element's height as follows:
$$\text{height} = \text{content height} + \text{padding} + \text{border}$$

```
#box-example {
box-sizing: border-box;
}
```

An activity for you!

Fill in the missing keywords in the code to your right. ↗
Set the width of the <div> element to "200px".

```
<style>
[ ] {
[ ] : [ ];
}
</style>

<body>
<div>
Lorem ipsum dolor sit amet,
consectetur adipiscing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua.
</div>

</body>
```

Submit your answer

Show Answer

Final Look – The Box Model Exercises

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Syntax and Selectors
- Visual rules
- The Box Model
- Display and Positioning
- Colors
- Typography

Add a 2px solid red border to the <div> element.

```
<style>
  div {
    width: 200px;
    border: 2px solid red;
  }
</style>

<body>

<div>
  Lorem ipsum dolor sit amet,
  consectetur adipiscing elit,
  sed do eiusmod tempor incididunt
  ut labore et dolore magna aliqua.
</div>

</body>
```

[Submit your answer](#)

[Show Answer](#)

Add 25 pixels space between the <div> element's border and its content.

```
<style>
  div {
    width: 200px;
    border: 2px solid red;
    padding: 25px;
  }
</style>

<body>

<div>
  Lorem ipsum dolor sit amet,
  consectetur adipiscing elit,
  sed do eiusmod tempor incididunt
  ut labore et dolore magna aliqua.
</div>

</body>
```

[Submit your answer](#)

[Show Answer](#)

Add a 25 pixels space outside, to the left of the <div> element

```
<style>
  div {
    width: 200px;
    border: 2px solid red;
    padding: 25px;
    margin-left: 25px;
  }
</style>

<body>

<div>
  Lorem ipsum dolor sit amet,
  consectetur adipiscing elit,
  sed do eiusmod tempor incididunt
  ut labore et dolore magna aliqua.
</div>

</body>
```

[Submit your answer](#)

[Show Answer](#)

Display And Positioning

TOPICS

Syntax and Selectors

Visual rules

The Box Model

Display and Positioning

Colors

Typography

Fixed CSS Positioning

Positioning in CSS provides designers and developers options for positioning HTML elements on a web page. The CSS position can be set to static, relative, absolute or fixed. When the CSS position has a value of fixed, it is set/pinned to a specific spot on a page. The fixed element stays the same regardless of scrolling. The navigation bar is a great example of an element that is often set to position:fixed, enabling the user to scroll through the web page and still access the navigation bar.

```
navbar {  
position: fixed;  
}
```

CSS display property

The CSS display property determines the type of render block for an element. The most common values for this property are block, inline, and inline-block.

Block-level elements take up the full width of their container with line breaks before and after, and can have their height and width manually adjusted.

Inline elements take up as little space as possible, flow horizontally, and cannot have their width or height manually adjusted.

Inline-block elements can appear next to each other, and can have their width and height manually adjusted.

```
.container1 {  
display: block;  
}  
  
.container2 {  
display: inline;  
}  
  
.container3 {  
display: inline-block;  
}
```

CSS position: absolute

The value absolute for the CSS property position enables an element to ignore sibling elements and instead be positioned relative to its closest parent element that is positioned with relative or absolute. The absolute value removes an element entirely from the document flow. By using the positioning attributes top, left, bottom and right, an element can be positioned anywhere as expected

```
.element {  
position: absolute;  
}
```

CSS position: relative

The value relative of the CSS position property enables an element to be positioned relative to where it would have originally been on a web page. The offset properties can be used to determine the actual position of the element relative to its original position. Without the offset properties, this declaration will have no effect on its positioning, it will act as the default value static of the position property.

```
.element {  
position: relative;  
}
```

An activity for you!

Position the <h1> element to always be 50px from the top, and 10px from the right, relative to the window/frame edges.

```
<style>  
h1{  
    : ;  
    : 50px;  
    : 10px;  
}  
</style>  
<body>  
<h1> This is a heading </h1>  
<p> This is a paragraph </p>  
</body>
```

Submit your answer

Show Answer

Final Look - Display And Positioning Exercises

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Syntax and Selectors
- Visual rules
- The Box Model
- Display and Positioning
- Colors
- Typography

Position the `<h1>` element 50px from the top, relative to its normal position.

```
<style>
h1 {
    [ ] : [ ];
    [ ] : 50px
}
</style>

<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Position the `<h1>` element 50px from the top, relative to the HTML page.

```
<style>
h1 {
    [ ] : [ ];
    [ ] : 50px
}
</style>

<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Position the `<h1>` element 50px from the top, by referring to its class name.

```
<style>
[ ] {
    [ ] : relative;
    top: [ ] ;
}
</style>

<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>>
```

[Submit your answer](#)

[Show Answer](#)

Colors

TOPICS

Syntax and Selectors

Visual rules

The Box Model

Display and Positioning

Colors

Typography

Color Name Keywords

Color name keywords can be used to set color property values for elements in CSS.

```
h1 {  
color: aqua;  
}
```

```
li {  
color: khaki;  
}
```

CSS Color Alpha Values

Alpha values determine the transparency of colors in CSS. Alpha values can be set for both RGB and HSL colors by using `rgba()` and `hsla()` and providing a fourth value representing alpha. Alpha values can range between 0.0 (totally transparent) and 1.0 (totally opaque).

The CSS `transparent` value can also be used to create a fully transparent element.

```
.midground {  
background-color: rgba(0, 255, 0, 0.5);  
}
```

```
.foreground {  
background-color: hsla(34, 100%, 50%, 0.1);  
}
```

```
.transparent {  
color: transparent;  
}
```

CSS Hexadecimal Colors

CSS colors can be represented in hexadecimal (or hex) notation. Hexadecimal digits can represent sixteen different values using 0-9 and a-f.

Hexadecimal colors are composed of 6 characters—each group of two represents a value between 0 and 255 for red, green, or blue. For example `#ff0000` is all red, no green, and no blue.

When both characters of all three colors are repeated, hex colors can be abbreviated to only three values, so `#0000ff` could also be represented as `#00f`.

```
.red {  
color: #ff0000;  
}
```

```
.short-blue {  
color: #00f;  
}
```

```
.hot-pink {  
color: rgb(249, 2, 171);  
}
```

```
.green {  
color: rgb(0, 255, 0);  
}
```

CSS `rgb()` Colors

CSS colors can be declared with RGB colors using `rgb()` syntax.

`rgb()` should be supplied with three values representing red, green, and blue. These values range from 0 to 255.

```
[ ] {  
color: [ ];  
}
```

Submit your answer

Show Answer

An activity for you!

Fill in the missing keywords in the code to your right. ↗

Set the color of all `<h1>` elements to blue.

Final Look - Colors Exercises

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Syntax and Selectors
- Visual rules
- The Box Model
- Display and Positioning
- Colors
- Typography

Insert the RGBA color value for a full red background color of the `<h1>` element, with no transparency.

```
<style>
h1 {
    background-color: [REDACTED];
}
</style>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

There is also a method were we can specify the alpha channel as well as hue, saturation, and lightness, what is the name of this method?

```
<style>
h1 {
    background-color: [REDACTED] (0, 100%, 50%, 0.3);
}
</style>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Add a 25 pixels space outside, to the left of the `<div>`element

```
<style>
h1 {
    background-color: rgb(255, 0, 0);
    [REDACTED]: 0.3;
}
</style>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

TOPICS

Syntax and Selectors

Visual rules

The Box Model

Display and Positioning

Colors

Typography

Typography

CSS font-weight Property

The CSS font-weight property declares how thick or thin should be the characters of a text. Numerical values can be used with this property to set the thickness of the text. The numeric scale range of this property is from 100 to 900 and accepts only multiples of 100. The default value is normal while the default numerical value is 400. Any value less than 400 will have text appear lighter than the default while any numerical value greater than the 400 will appear bolder.

In the given example, all the `<p>` elements will appear in a bolder font.

```
/* Sets the text as bolder. */
p {
  font-weight: 700;
}
```

CSS font-style property

The CSS font-style property determines the font style in which text will appear. It accepts italic as a value to set the font style to italic.

```
.text{
  font-style: italic; }
```

CSS Fallback Fonts

The CSS font-family property can have multiple fonts declared in order of preference. In this case the fonts following the initial font are known as the fallback fonts. If the initial value of the property font-family fails to load to the webpage, the fallback fonts will be used.

```
/* Here `Arial` is the fallback font for <p>tags */
p {
  font-family: "Helvetica", "Arial";
}
```

The CSS line-height property

The CSS line-height property declares the vertical spacing between lines of text. It accepts both unitless numbers as a ratio (eg. 2) and numbers specified by unit as values (eg. 12px) but it does not accept negative numbers. A unitless number is an absolute value that will compute the line height as a ratio to the font size and a unit number can be any valid CSS unit (eg. pixels, percents, ems, rem, etc.). To set the line-height of the `<p>` elements to 10px, the given CSS declaration can be used.

```
p {
  line-height: 10px;
}
```

CSS Linking fonts

Linking fonts allow user to use web fonts in the document. They can be imported in an HTML document by using the `<link>` tag. Once the web font URL is placed within the `href` attribute, the imported font can then be used in CSS declaration.

```
<head>
<link href="https://fonts.googleapis.com/css?family=Droid%20Serif" rel="stylesheet">
</head>
```

An activity for you!

Fill in the missing keywords in the code to your right. ↗

Set the font weight of all `<p>` elements to 500.

```
p {
  [ ] : [ ];
}
```

Submit your answer

Show Answer

Final Look - Typography Exercises

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Exercises

TOPICS

- Syntax and Selectors
- Visual rules
- The Box Model
- Display and Positioning
- Colors
- Typography

Set the font for `<h1>` to "Verdana".

```
<style>
h1 {
    [ ] : Verdana;
}
</style>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Set the style of `<h1>` to "italic" text.

```
<style>
h1 {
    [ ] : [ ] ;
}
</style>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

[Show Answer](#)

Set the font size of `<h1>` to 50px.

```
<style>
h1 {
    [ ] : 50px;
}
</style>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph</p>
<p>This is a paragraph</p>
</body>
```

[Submit your answer](#)

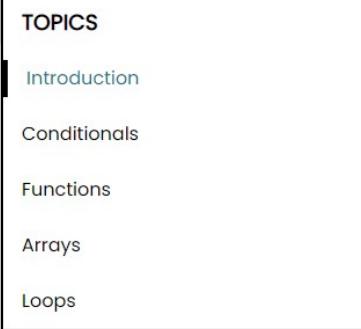
[Show Answer](#)

Final Look – JavaScript Tutorials

> _Le Bateau

Tutorials Code Challenges About Us Log in Sign up

Introduction to JavaScript



JavaScript

JavaScript is a programming language that powers the dynamic behavior on most websites. Alongside HTML and CSS, it is a core technology that makes the web run.

console.log()

The console.log() method is used to log or print messages to the console. It can also be used to print objects and other info.

```
console.log("Hi there!");
```

// Prints: Hi there!

An activity for you!

log "hello world" to the console.

```
console.log("Hello World");
```

Submit your answer

Show Answer

Single-Line Comments

In JavaScript, single-line comments are created with two consecutive forward slashes //.

```
// This lines will denote a comment
```

Multi-Line Comments

In JavaScript, multi-line comments are created by surrounding the lines with /* at the beginning and */ at the end. Comments are good ways for a variety of reasons like explaining a code block or indicating some hints, etc.

```
/*
The code below will
print "hello world"
to the console.
*/
console.log("hello world");
```

Variables

Variables are used whenever there's a need to store a piece of data. A variable contains data that can be used in the program elsewhere. Using variables also ensures code re-usability since it can be used to replace the same value in multiple places.

```
const currency = '$';
const userIncome = 85000
```

Declaring Variables

To declare a variable in JavaScript, any of these three keywords can be used along with a variable name:

var is used in pre-ES6 versions of JavaScript.

let is the preferred way to declare a variable when it can be reassigned.

const is the preferred way to declare a variable with a constant value.

```
var age;
let weight;
const numberOfFingers = 20;
```

let Keyword

let creates a local variable in JavaScript & can be re-assigned. Initialization during the declaration of a let variable is optional. A let variable will contain undefined if nothing is assigned to it.

```
let count;
console.log(count); // Prints: undefined
count = 10;
console.log(count); // Prints: 10
```

const Keyword

A constant variable can be declared using the keyword const. It must have an assignment. Any attempt of re-assinging a const variable will result in JavaScript runtime error

```
const numberOfRowsColumns = 4;
numberOfColumns = 8;
// TypeError: Assignment to constant variable.
```

String Concatenation

In JavaScript, multiple strings can be concatenated together using the + operator. In the example, multiple strings and variables containing string values have been concatenated. After execution of the code block, the displayText variable will contain the concatenated string.

```
let service = 'credit card';
let month = 'May 30th';
let displayText = 'Your' + service + 'bill is due on' + month;
console.log(displayText);
```

Template Literals

Template literals are strings that allow embedded expressions, \${expression}. While regular strings use single ' or double " quotes, template literals use backticks instead.

```
let name = "John Smith";
console.log(`Hello, ${name}`);
// Prints: Hello, John Smith.
console.log(`Billy is ${5+8} years old`);
// Prints: Billy is 13 years old.
```

Arithmetic Operators

JavaScript supports arithmetic operators for:

+ addition

- subtraction

* multiplication

/ division

% modulo

```
// Addition
5 + 5
// Subtraction
15 - 5
// Multiplication
5 * 10
// Division
10 / 5
// Modulo
10 % 5
```

Final Look - Introductional Exercises

The screenshot shows a web-based coding exercise platform. At the top, there's a navigation bar with links for Tutorials, Code Challenges, About Us, Log In, and Sign up. On the left, a sidebar titled "TOPICS" lists Fundamentals, Conditionals, Functions, Arrays, and Loops. The main content area is titled "Exercises".

Exercise 1: Create a variable called carName, assign the value Volvo to it.

```
var carName = "Volvo";
```

Exercise 2: Display the sum of 5 + 10, using two variables: x and y.var x = 5;
var y = 10;
document.getElementById("tuts").innerHTML = x + y;

Exercise 3: Alert the **remainder** when 15 is divided by 4.

```
alert(15 % 9);
```

Conditionals

TOPICS

- Introduction
- Conditionals
- Functions
- Arrays
- Loops

Control Flow

Control flow is the order in which statements are executed in a program. The default control flow is for statements to be read and executed in order from left-to-right, top-to-bottom in a program file.

Control structures such as conditionals (if statements and the like) alter control flow by only executing blocks of code if certain conditions are met. These structures essentially allow a program to make decisions about which code is executed as the program runs.

Comparison Operators

Comparison operators are used to comparing two values and return true or false depending on the validity of the comparison:

- `==` strict equal
- `!=` strict NOT equal
- `<` greater than
- `<=` greater than or equal
- `>` less than
- `>=` less than or equal

```
1 < 3 // false
3 < 1 // true
250 <= 250 // true
1 === 1 // true
1 === 3 // false
1 === T // false
```

if Statement

An if statement accepts an expression with a set of parentheses:

If the expression evaluates to a truthy value, then the code within its code body executes.

If the expression evaluates to a falsy value, its code body will not execute.

```
const isMailSent = true

if(isMailSent) {
  console.log('Mail sent to recipient');
}
```

else if Clause

After an initial if block, else if blocks can each check an additional condition. An optional else block can be added after the else if block(s) to run by default if none of the conditionals evaluated to truthy.

```
const size = 10;

if (size > 100) {
  console.log(Big);
} else if (size > 20) {
  console.log(Medium);
} else if (size > 4) {
  console.log(Small);
} else{
  console.log(Tiny);
```

else Clause

An else block can be added to an if block or series of if-else if blocks. The else block will be executed only if the if condition fails.

```
const isMailSent = true

if(isMailSent) {
  console.log('Mail sent to recipient');
} else {
  console.log('Mail Not sent to recipient');
}
```

switch Statement

The switch statements provide a means of checking an expression against multiple case clauses. If a case matches, the code inside that clause is executed.

The case clause should finish with a break keyword. If no case matches but a default clause is included, the code inside default will be executed.

Note: If break is omitted from the block of a case, the switch statement will continue to check against case values until a break is encountered or the flow is broken.

```
const food = 'salad';

switch (food) {
  case 'oyster':
    console.log('The taste of the sea 🐚');
    break;
  case 'pizza':
    console.log('So Good! 🍕');
    break;
  default:
    console.log('Enjoy your meal 🍔');
}

// Prints: Enjoy your meal 🍔
```

An activity for you!

Choose the correct comparison operator to alert true, when x is greater than y.

```
x = 10;
y = 5;
alert(x [ ] y);
```

Submit your answer

Show Answer

Final Look - Conditional Exercises

The screenshot shows a web page titled "Exercises" from the website "Le Bateau". The top navigation bar includes links for Tutorials, Code Challenges, About Us, Log in, and Sign up. A sidebar on the left is titled "TOPICS" and lists Fundamentals, Conditionals, Functions, Arrays, and Loops. The main content area contains three separate code editing and submission sections.

Exercise 1: Fix the if statement to alert "Hello World" if x is greater than y.

```
if [ ] x > y [ ] [ ]  
alert('Hello World');  
[ ]
```

Exercise 2: Fix the if statement to alert "Hello World" if x is greater than y, otherwise alert "Goodbye".

```
[ ] (x [ ] y) {  
alert('Hello World');  
} [ ] {  
alert('Goodbye');  
}
```

Exercise 3: Create a switch statement that will alert "Miam" if dinner is "burger", and "Ewww!" if dinner is "salad".

```
[ ] (dinner) {  
[ ] "Burger":  
alert("Miam")  
break;  
[ ] "Salad":  
alert("Ewww!")  
break;  
}
```

Functions

TOPICS

- Introduction
- Conditionals
- Functions
- Arrays
- Loops

Functions

Functions are one of the fundamental building blocks in JavaScript. A function is a reusable set of statements to perform a task or calculate a value. Functions can be passed one or more values and can return a value at the end of their execution. In order to use a function, you must define it somewhere in the scope where you wish to call it.

The example code provided contains a function that takes in 2 values and returns the sum of those numbers.

Control structures such as conditionals (if statements and the like) alter control flow by only executing blocks of code if certain conditions are met. These structures essentially allow a program to make decisions about which code is executed as the program runs

```
// Defining the function:  
function sum(num1,num2) {  
    return num1 + num2;  
}  
  
// Calling the function:  
sum(3,6); // 9
```

Functions Expressions

Function expressions create functions inside an expression instead of as a function declaration. They can be anonymous and/or assigned to a variable.

```
const dog = function() {  
    return 'Woof!';  
}
```

An activity for you!

Pass the variable name as a parameter for the function "hello".

```
// The parameter is name  
function sayHello(name) {  
    return 'Hello, ${name}!';  
}
```

```
let name = 'Jimmy';  
function hello(_____) {  
    console.log("Hello" + name);  
}
```

[Submit your answer](#)

[Show Answer](#)

return Keyword

Functions return (pass back) values using the return keyword. return ends function execution and returns the specified value to the location where it was called. A common mistake is to forget the return keyword, in which case the function will return undefined by default.

```
// With return  
function sum(num1,num2) {  
    return num1 + num2;  
}  
  
// Without return, so the function doesn't output the sum  
function sum(num1,num2) {  
    num1 + num2;  
}
```

Final Look - Functional Exercises

The screenshot shows a web application interface for learning functional programming. At the top, there's a navigation bar with links for Tutorials, Code Challenges, About Us, Log in, and Sign up. On the left, a sidebar titled "TOPICS" lists Fundamentals, Conditionals, Functions (which is currently selected), Arrays, and Loops. The main content area is titled "Exercises". It contains three separate code editor sections, each with a "Submit your answer" button and a "Show Answer" button.

Exercise 1: Call the function named Greetings

```
function Greetings() {  
  alert('Hello World!');  
}
```

Exercise 2: Create a function called "Greetings"

```
function Greetings() {  
  alert('Hello World');  
}
```

Exercise 3: Make the function "Greetings" return "hello"

```
function Greetings() {  
  return 'hello';  
}  
Greetings();
```

Arrays

TOPICS

Introduction

Conditionals

Functions

Arrays

Loops

Arrays

Arrays are lists of ordered, stored data. They can hold items that are of any data type. Arrays are created by using square brackets, with individual elements separated by commas.

```
// An array containing numbers
```

```
const numberArray = [0, 1, 2, 3];
```

```
// An array containing different data types
```

```
const numberArray = [1, 'chicken', false];
```

Index

Array elements are arranged by index values, starting at 0 as the first element index. Elements can be accessed by their index using the array name, and the index surrounded by square brackets.

```
// Accessing an array element
```

```
const numberArray = [100, 200, 300];
```

```
console.log(myArray[0]); // 100
```

```
console.log(myArray[1]); // 200
```

```
console.log(myArray[2]); // 300
```

Property .length

The length property of a JavaScript array indicates the number of elements the array contains.

```
const numberArray = [1, 2, 3, 4];
```

```
number.length // 4
```

An activity for you!

Declare an array, cars with the value "Ferrari", "Mercedes", "Red Bull"

```
let cars = ["", "", "", ""];
```

Submit your answer

Show Answer

Final Look – Arrays Exercises

The screenshot shows a web page titled "Exercises" from the website "Le Bateau". The top navigation bar includes links for Tutorials, Code Challenges, About Us, Log in, and Sign up. A sidebar on the left lists "TOPICS" such as Fundamentals, Conditionals, Functions, Arrays (which is highlighted in blue), and Loops.

Exercise 1: Get the value "Red Bull" from the cars array.

```
const cars = ["Ferrari", "Mercedes", "Red Bull"]  
let x = ;
```

[Submit your answer](#) [Show Answer](#)

Exercise 2: Change the second item of cars to "McLaren".

```
const cars = ["Ferrari", "Mercedes", "Red Bull"]  
 = "McLaren";
```

[Submit your answer](#) [Show Answer](#)

Exercise 3: Alert the number of items in the array "cars", using the correct Array property.

```
const cars = ["Ferrari", "Mercedes", "Red Bull"]  
alert();
```

[Submit your answer](#) [Show Answer](#)

Loops

TOPICS

- Introduction
- Conditionals
- Functions
- Arrays
- Loops**

While Loop

The while loop creates a loop that is executed as long as a specified condition evaluates to true. The loop will continue to run until the condition evaluates to false. The condition is specified before the loop, and usually, some variable is incremented or altered in the while loop body to determine when the loop should stop.

```
while(condition) {
  // code block to be executed
}

let i = 0;

while(i < 5) {
  console.log();
  i++;
}
```

Do...While Statement

A do...while statement creates a loop that executes a block of code once, checks if a condition is true, and then repeats the loop as long as the condition is true. They are used when you want the code to always execute at least once. The loop ends when the condition evaluates to false.

```
x = 0;
i = 0;

do {
  x = x + i;
  console.log(x);
  i++;
} while(i < 5);
```

For Loop

The for loop declares looping instructions, with three important pieces of information separated by semicolons :

- The initialization defines where to begin the loop by declaring (or referencing) the iterator variable
- The stopping condition determines when to stop looping (when the expression evaluates to false)
- The iteration statement updates the iterator each time the loop is completed

```
for (let i = 0; i < 4; i += 1) {
  console.log();
};

// Output: 0,1,2,3
```

Looping Through Arrays

An array's length can be evaluated with the .length property. This is extremely helpful for looping through arrays, as the .length of the array can be used as the stopping condition in the loop.

```
for (let i = 0; i < array.length; i++) {
  console.log(array[i]);
};

// Output: Every item in the array
```

Break Keyword

Within a loop, the break keyword may be used to exit the loop immediately, continuing execution after the loop body. Here, the break keyword is used to exit the loop when i is greater than 5.

```
for (let i = 0; i < 99; i++) {
  if(i > 5){
    break;
  }
  console.log();
}

// Output: 0,1,2,3
```

An activity for you!

break from the iteration when i is equal to 5

```
for (let i = 0; i < 10; i++) {
  if (i == 5) {
    // Your code here
  }
  console.log(i);
}
```

Submit your answer

Show Answer

Final Look – Loops Exercises

The screenshot shows a web-based exercise platform with a sidebar menu and three distinct code challenges.

TOPICS:

- Fundamentals
- Conditionals
- Functions
- Arrays
- Loops

Exercises:

Create a for...loop that runs 10 times.

```
let i;
[ ] ( [ ] = [ ]; [ ] > [ ]; [ ] ) {
    console.log(i);
}
```

Submit your answer **Show Answer**

Create a loop that runs as long as i is less than 10.

```
let i = 0;
[ ] (i [ ] 10) {
    console.log(i);
    i++;
}
```

Submit your answer **Show Answer**

Write a do while loop that will execute until i is greater than 10

```
do {
    console.log("the number is " + i);
    i++;
}
[ ] (i [ ] 10)
```

Submit your answer **Show Answer**

Code Challenge Page

Description

- This is a landing page where it contain all the code challenge where you can practice your coding skills

The screenshot shows a landing page for a code challenge website. At the top, there's a navigation bar with links for 'Tutorial', 'Code Challenge', 'About Us', 'Log In', and 'Sign Up'. Below the navigation, the title 'Code Challenge' is displayed, followed by a subtitle: 'Test your knowledge with code challenges Practice for your job search — or for fun. Don't worry if you get stuck.' The main content area features a 3x3 grid of cards, each representing a different coding task:

- Lemon Drizzle Cake Recipe (HTML Task)**: Icon of a lemon slice.
- Wanted Poster (CSS Task)**: Icon of a wanted poster.
- Digicode (JavaScript)**: Icon of a numeric keypad.
- The colours of the rainbow (HTML task)**: Icon of a rainbow.
- My Timetable in (HTML / CSS)**: Icon of a calendar.
- Minesweeper (Javascript)**: Icon of a bomb.
- Road Signs (HTML / CSS)**: Icon of road signs.
- Creating tabs (HTML + CSS + JS)**: Icon of a stack of tabs.
- Click Me! (Javascript)**: Icon of a hand pointing at a button.

Each card includes a 'Let's try ->' button at the bottom right. At the bottom of the page, there's a footer with links for 'Tutorial', 'Code Challenge', 'About Us', 'Privacy Policy', and 'Terms and conditions', along with a copyright notice: '© 2022 Copyright • Le Bateau'.

HTML

This menu is a bit different as we put link in a div and a use flex box to display them inline. Using flex: 1 will take the remaining space in the container

HTML Tag used in this page are :

- h1
- p
- img
- a (link)

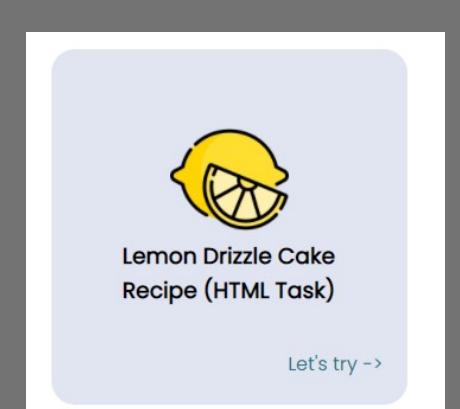
In the image tag we also used the alt attribute specifies an alternate text for an image, if the image cannot be displayed.

CSS

We used text-align: center; to center h1, h3, and p

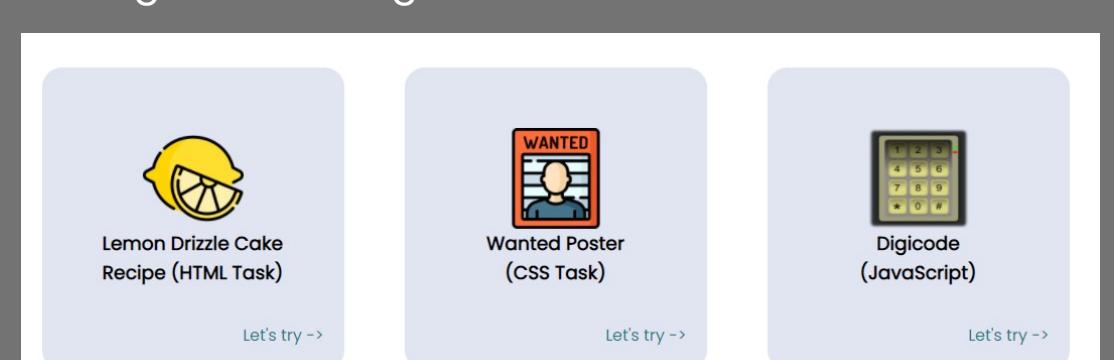
The figure at the right has a background color of #e1e5f2 and a border radius of 20px

To center the image inside the property display: block; is used



The 3 boxes in the figure below is in a div Using flexbox

- we are able to display them horizontally
- flex-wrap: wrap; allow us to make our website more responsive. It will wrap the content onto multiple lines, from top to bottom.
- justify-content: center; This allows us to defines center alignment along the main axis.



Practice Coding Pages

Description

- These are the pages where people can practice coding. We got the questions for the coding pages from: <https://www.101computing.net/html-css-javascript-challenges/>

> _ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

WANTED

Wanted Poster (CSS Task)

Let's Start

For this challenge we will create a Wanted Poster using a range of HTML and CSS skills.

CSS Selectors & CSS Properties

In order to complete this challenge, you need a good understanding of how CSS selectors work. You can learn about [CSS selectors on w3schools.com](#).

```
selector {  
    property: value;  
    property: value;  
    property: value;  
}
```

To customise the look & feel of your poster using CSS code, you will use different types of selectors such as:

- TAG
- .class
- #id
- element child-element
- element child-element:first-child

In this challenge we will use various CSS properties such as:

- font-family
- font-size
- color
- font-weight
- font-style
- text-align
- padding
- margin
- background-image
- etc

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
@import url(https://fonts.googleapis.com/css?family=Ewert);  
@import url(https://fonts.googleapis.com/css?family=Ultra);  
</style>  
</head>  
  
<body>  
<div class="poster">  
<h1>WANTED</h1>  
<p class="subheading">Dead or Alive</p>  
  
<p class="name">Billy the Kid</p>  
<p class="description">Armed & very dangerous</p>  
<h2>Reward</h2>  
<div id="price">$50.000</div>  
</body>  
</html>
```

WANTED
Dead or Alive



Billy the Kid
Armed & very dangerous

Reward
\$50.00

Note that this script will use three pictures. In case these are not displaying properly, you may have to replace their URLs, using the following addresses:
Mugshot Image: <https://www.101computing.net/wp/wp-content/uploads/mugshot.png>
Wood Texture: <https://www.101computing.net/wp/wp-content/uploads/wood.jpg>
Parchment Texture: <https://www.101computing.net/wp/wp-content/uploads/parchment.jpg>

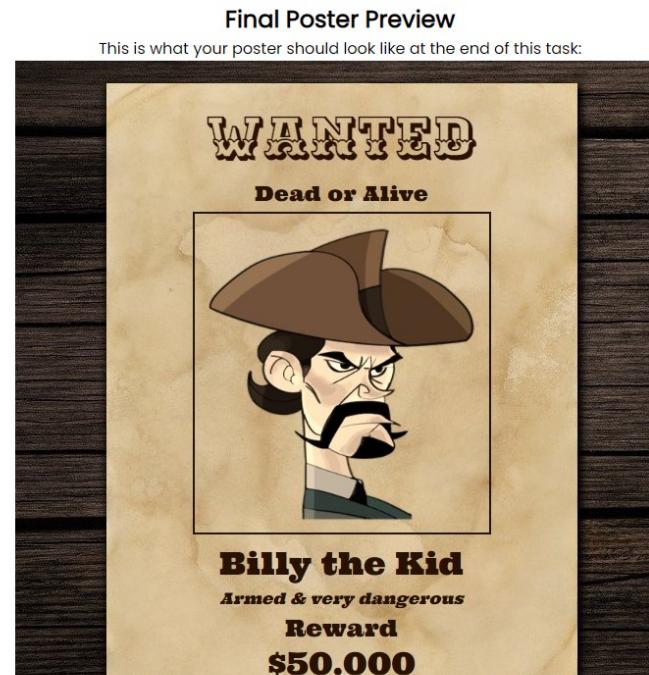
Your task

Wooden Texture Paper Texture Mugshot Wanted Typography Sepia Filter

Your first task is to apply a wooden texture to the whole page. To do so you will need to:

- Apply a background picture to the BODY of the page. The URL for this picture is the wooden texture as provided above
- Apply a 20px padding to the whole page.

Final Poster Preview
This is what your poster should look like at the end of this task:



Tutorial Code Challenge About Us Privacy Policy Terms and conditions
© 2022 Copyright • Le Bateau

Practice Coding Pages

> _ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

Road Signs in HTML / CSS

Let's Start



In this set of challenges we are going to use CSS to recreate various road signs.

Learning Objectives

By completing these challenges you will learn more about how CSS can be used to format information on the page.
We will be looking at:

- How to add borders, and border radius to create a rounded corner text frame.
- How to add a shadow to the text frame.
- How to add a gradient effect to our text frame

challenge 1

The first challenge consists of recreating this number plate using CSS:



Check the following code to see how the border was created using both the border and the border-radius CSS definitions. You can also investigate how the shadow was created using the box-shadow CSS definition.

```
<!DOCTYPE html>
<html>
<head>
<style>
#plate {
display: block;
width: 450px;
height: 100px;
background-color: #FFFF00;
box-shadow: 10px 10px 5px #888888;
border: 5px solid;
border-radius: 20px;
}

```

Challenge #2

For this challenge we are using exactly the same CSS definitions. By increasing the border-radius we are creating the round shape.

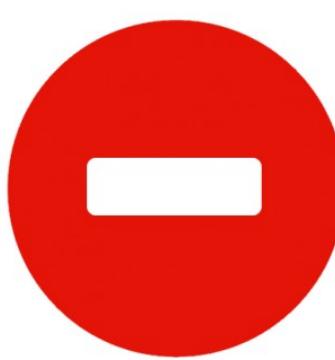


```
<!DOCTYPE html>
<html>
<head>
<style>
#road-sign {
display: block;
width: 150px;
height: 150px;
background-color: #FFFFFF;
box-shadow: 10px 10px 5px #888888;
border: 20px solid #EE0000;
border-radius: 100px;
}

```

Challenge #3

Try to recreate these road signs using CSS:

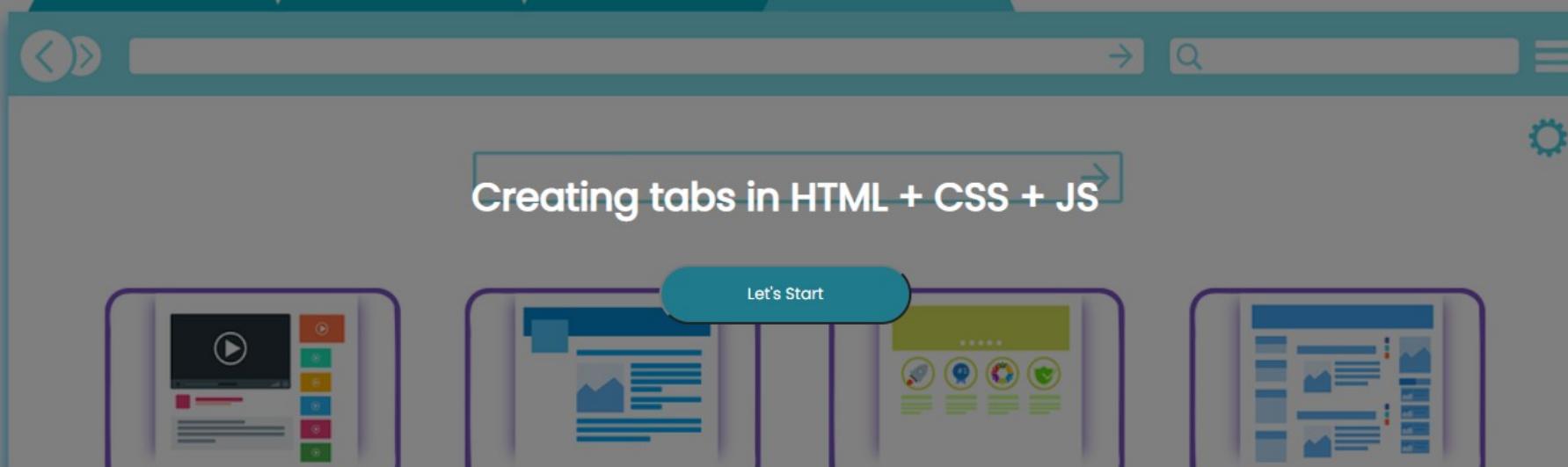


Tutorial Code Challenge About Us Privacy Policy Terms and conditions

© 2022 Copyright • Le Bateau

Practice Coding Pages

> Le Bateau Tutorial Code Challenge About Us Log In Sign Up



In previous posts we looked at how JavaScript can interact with HTML using:

- Event Handlers in HTML such as onClick, onFocus, onBlur...
- the DOM (Document Object Model) to access HTML objects using: document.getElementById("objectId")

CSS is applied to HTML objects. Using JavaScript you can change the **CSS attributes** of all your HTML objects using:

- document.getElementById("objectId").style

For instance:

- document.getElementById("myTextBox").style.backgroundColor="#FF0000";
- document.getElementById("myHeading").style.padding="25px";
- document.getElementById("myDiv").style.display="none"; //This will make the DIV tag invisible
- document.getElementById("myDiv").style.display="block"; //This will make the DIV tag visible

Let's combine all these techniques to create some tabs:

The HTML code is fairly straightforward: Each tab needs a DIV tag for the tab itself and a DIV tag for its content:

```
<div id="tab1" onClick="JavaScript:selectTab(1);>Tab 1</div >
<div id="tab2" onClick="JavaScript:selectTab(2);>Tab 2</div >
<div id="tab3" onClick="JavaScript:selectTab(3);>Tab 3</div >
<br />
<div id="tab1Content" >
This is the content to display in the first tab.
</div >
<div id="tab2Content" >
Welcome to tab 2!
</div >
<div id="tab3Content" >
Tab 3 is probably the best of the three tabs.
</div >
```

Then CSS code can be used to customise the look and feel of the tabs (Background colours, borders, padding...) But mainly CSS is used to only display the first tab's content area and hide the other two. This is done using the display attribute in CSS:

```
#tab1Content { display: block; } #tab2Content, #tab3Content { display: none; }
```

Finally the JavaScript code is used to handle the onClick event triggered by the tabs. It accesses the style attribute of each tab's content area to either hide or display the content of the tab based on which tab has been clicked.

```
function selectTab(tabIndex) { //Hide All Tabs
document.getElementById('tab1Content').style.display="none";
document.getElementById('tab2Content').style.display="none";
document.getElementById('tab3Content').style.display="none"; //Show the Selected Tab
document.getElementById('tab' + tabIndex + 'Content').style.display="block"; }
```

[Let's see all this code in action!](#)

```
<!DOCTYPE html>
<html>
<head>
<style>
#tab1, #tab2, #tab3 {
float: left;
padding: 5px 10px 5px 10px;
background: #B00098;
color: #FFFFFF;
margin: 0px 5px 0px 5px;
cursor: pointer;
border-radius: 5px;
}

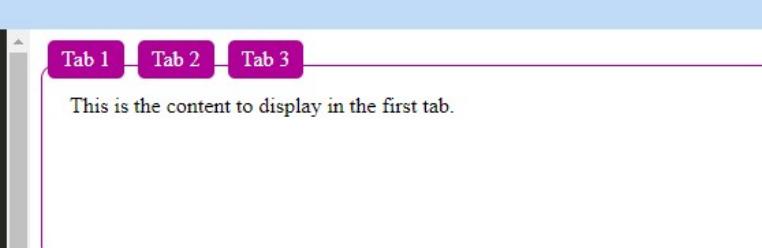
#tab1:hover, #tab2:hover, #tab3:hover {
background: #E800C9;
}

#tab1Content, #tab2Content, #tab3Content {
width: 500px;
height: 100px;
padding: 20px;
border: 1px solid #B00098;
border-radius: 10px;
}

#tab1Content {
display: block;
}

#tab2Content, #tab3Content {
display: none;
}

</style>
</head>
<body>
<div id="tab1" onClick="JavaScript:selectTab(1);>Tab 1</div >
<div id="tab2" onClick="JavaScript:selectTab(2);>Tab 2</div >
<div id="tab3" onClick="JavaScript:selectTab(3);>Tab 3</div >
```



Your challenge:

Tweak this code to add an extra tab (Tab 4) to this script. You will need to tweak the HTML code first, then the CSS then the JavaScript.

Practice Coding Pages

> _ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

My Timetable in HTML / CSS

Let's Start

Learning Objectives

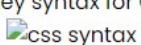
In this challenge we are learning how to draw and format a **table** on a web page using a range of **HTML tags** and **CSS properties**.

HTML Tables

First you may want to [read a bit more about how HTML tables are built](#) using the `<TABLE>`, `<THEAD>`, `<TBODY>`, `<TR>`, `<TH>` and `<TD>` HTML tags

Css Code

Remember the key syntax for CSS is as follows:



To customise the look & feel of your poster using CSS code, you will use different types of selectors such as:

- TAG
- .class
- #id
- element child-element
- element child-element:first-child

You can learn more about all these [CSS selectors](#) on w3schools.com.

In this challenge we will use various CSS properties such as:

- font-family
- font-size
- color
- font-weight
- font-style
- text-align
- padding
- margin
- background-image
- etc

My Timetable

We have created a timetable for you.

Task 1: By changing the **HTML code**, you can now change the content of this timetable: you can add your own lessons to reproduce your own timetable. You may want to change the structure of the day (e.g. Does your school day consist of 5 lessons?) by adding or deleting rows to this timetable.

Task 2: By changing the **CSS code**, you can now change the look & feel of your timetable. Change the colour scheme, the choice of fonts, the spacing (padding / margin) between table cells, etc.

```
<!DOCTYPE html>
<html>
<head>
<style>
BODY {
    font-family: Trebuchet MS;
    margin: 20px;
    background-color: #600040;
}

H1 {
    text-align: center;
    color: #FFFFFF;
}

.myTimetable {
    width: 100%;
    border-collapse: collapse;
    table-layout: fixed;
}

.myTimetable THEAD{
    color: #FFFFFF;
}

.myTimetable TBODY {
    background-color: #AAAAFF;
}

.myTimetable TD {
    border: 1px solid black;
    padding: 6px;
    text-align: center;
}
.myTimetable TBODY TR TD:first-child {
    background-color: #FF0066;
    font-weight: bold;
}

.break, .lunch {
    background-color: #FF0066 !important;
}
```

My Timetable					
	Monday	Tuesday	Wednesday	Thursday	Friday
P1	Maths A120	Art C1	English B21	Maths A120	Geography B101
P2	Science Lab1	History B104	Spanish C17	P.E. A Gym	Maths A120
Break					
P3	I.T. ICT 1	English B21	Musique C5	English B21	PSHE A24
Lunch					
P4	History B104	Drama C17	Maths A120	Geography B101	P.E. A Gym
P5	Spanish C17	Science Lab1	English B21	Science Lab1	R.E. B18

Practice Coding Pages

>_ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

The colours of the rainbow (HTML task)

Let's Start

In this blog post we are going to help Asya complete her homework research task about rainbows. Asya has collated a range of interesting facts about rainbows and has decided to submit her work as an HTML page. She has started her page but needs your help to format this page further using a range of HTML tags. To help her improve the look and feel of her page you will need to complete the six tasks listed below. But first let's look at Asya's HTML page so far:

Rainbow Facts!

Colours of the rainbow:
The 7 colours of the rainbow are: red, orange, yellow, green, blue, indigo, violet.

What Makes a Rainbow?
A rainbow is an arch of colours visible in the sky, caused by the refraction and dispersion of the sun's light by rain or other water droplets in the atmosphere.

Rainbow Myth!
There are several myths related to rainbows. Here is the most popular one:
What Makes a Rainbow? A rainbow is an arch of colours visible in the sky, caused by the refraction and dispersion of the sun's light by rain or other water droplets in the atmosphere. So to see a rainbow, you need mixed weather with a bit of sunshine and a bit of rain too! Rainbow Myth! There are several myths related to rainbows. Here is the most popular rainbow myth: There's a pot of gold at the rainbow's end. Find out more: Check the wikipedia page about rainbows to find out more.

Find out more:
Check the wikipedia page about rainbows to find out more.

Your task

Task 1: Headings Task 2: Bullet Points Task 3: Colours Task 4: Formatting text Task 5: Hyperlinks Task 6: Your turn

Task 1: Headings and Paragraphs

In order to give this page more structure we are going to add some headings, subheadings and paragraphs. We will do so using the following HTML tags:

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
<p>Paragraph</p>
```

Your task is to update Asya's code by adding some **<h1>** tags for the main heading:

- Rainbow facts!

And some **<h2>** tags for the subheading of the page:

- Colours of the rainbow
- What Makes a Rainbow?
- Rainbow Myths
- Find out more:

All other pieces of text should be displayed as paragraphs using **<p>** tags.

Show Answer

Practice Coding Pages

> Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

JavaScript: Click Me!

Let's Start

When working on HTML and JavaScript projects it is very likely that you will want to respond to events. For instance you may want to run some JavaScript code when a button is clicked.

Examples of HTML events:

- When a user clicks the mouse (onClick event)
- When a web page has loaded (onLoad event)
- When the mouse moves over an element (onMouseOver and onMouseOut events)
- When an input field gets the focus (onFocus event) or loses the focus (onBlur event)
- When an input field is changed (onChange event)
- When an HTML form is submitted (onSubmit event)
- When a user strokes a key (onKeyPress event)

To assign events to HTML elements you can use event attributes within your HTML code. For instance:

```
<INPUT type="button" value="Click Me!" onClick="JavaScript: alert('Ouch!');">
```

Let's look at a few examples:

onClick Event

```
<!DOCTYPE html>
<html>
<head>
<style>
#timeBox {
    display: block;
    width: 300px;
    height: 40px;
    text-align: center;
    border: 1px solid #FF0000;
    padding-top: 20px;
    margin-top: 20px;
}
</style>
</head>
<body>
<div id="timeBox"></div>
<script>
function displayTime() {
    var d = new Date();
    var time = d.toLocaleTimeString();
    document.getElementById("timeBox").innerHTML = time;
}
setInterval(displayTime, 1000);
</script>

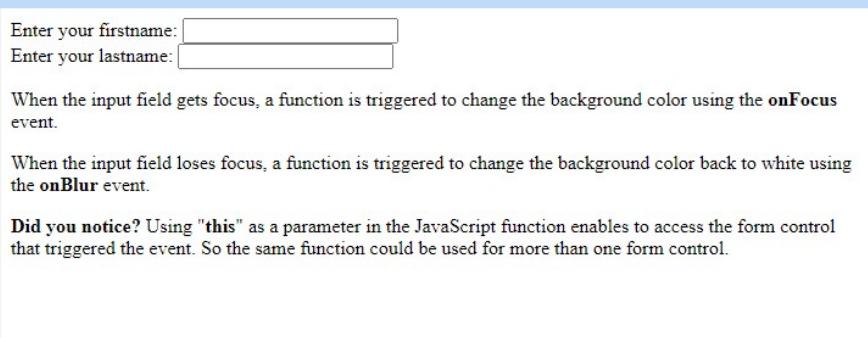
```



onFocus and onBlur Events

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
Enter your firstname: <input type="text" onFocus="JavaScript:highlight(this);>
<br/>
Enter your lastname: <input type="text" onFocus="JavaScript:highlight(this);>
<br/>
<p>When the input field gets focus, a function is triggered to change the background color using the onFocus event.
<p>When the input field loses focus, a function is triggered to change the background color back to white using the onBlur event.
<p><b>Did you notice?</b> Using "<b>this</b>" as a parameter in the JavaScript function enables to access the form control that triggered the event. So the same function could be used for more than one form control.
<script>
function highlight(x) {
    x.style.backgroundColor = "#FFFF00";
}
function resetColor(x) {
    x.style.backgroundColor = "white";
}
</script>

```

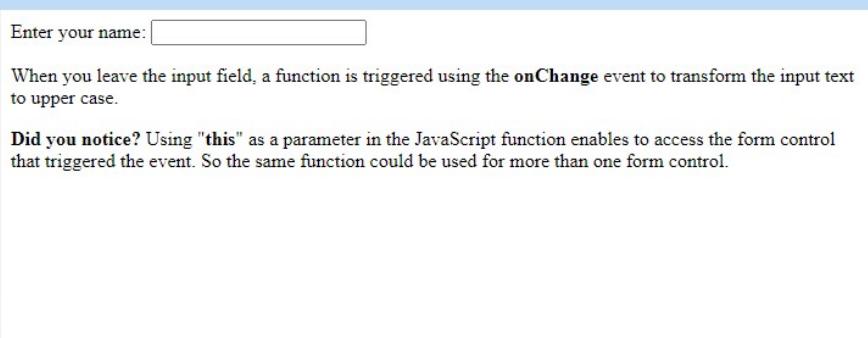


onChange Event

Did you notice? Using "this" as a parameter in the JavaScript function enables to access the form control that triggered the event. So the same function could be used for more than one form control.

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
Enter your name: <input type="text" onChange="JavaScript: changeCase(this);>
<p>When you leave the input field, a function is triggered using the onChange event.
<p><b>Did you notice?</b> Using "<b>this</b>" as a parameter in the JavaScript function enables to access the form control that triggered the event. So the same function could be used for more than one form control.
<script>
function changeCase(x) {
    x.value = x.value.toUpperCase();
}
</script>

```

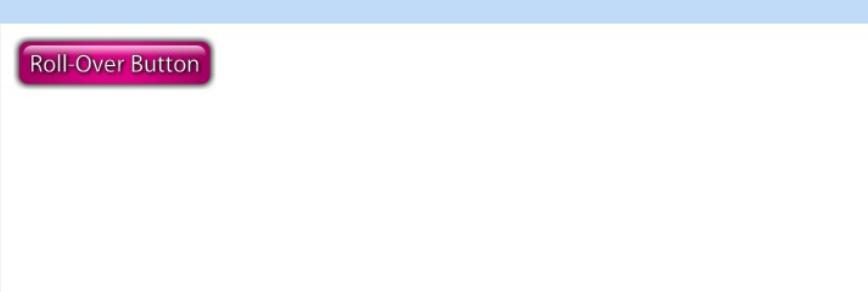


onMouseOver and onMouseOut Events

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>

<script>
function rollOver() {
    document.getElementById("myImage").src = "https://www.101computing.net/wp/wp-content/uploads/2017/02/roll_over_button_over.gif";
}
function rollOut() {
    document.getElementById("myImage").src = "https://www.101computing.net/wp/wp-content/uploads/2017/02/roll_over_button.gif";
}
</script>

```



Tutorial

Code Challenge

About Us

Privacy Policy

Terms and conditions

Practice Coding Pages

>_ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

Minesweeper in Javascript

Let's Start

In this blog post we will work on the classic game of Minesweeper. Minesweeper is a single-player puzzle video game. The objective of the game is to clear a rectangular board containing hidden "mines" or bombs without detonating any of them, with help from clues about the number of neighbouring mines in each field. The game originates from the 1960s, and has been written for many computing platforms in use today.

We have created a fully working version of the game using HTML, CSS and JavaScript. Your task will be to reverse-engineer the code to be able to improve this game further.

Full Code

```
<!DOCTYPE html>
<html>
<head>
<style>
BODY {
    background: black;
    color: #DDDDDD;
    font-family: courier new;
    text-align: center;
}

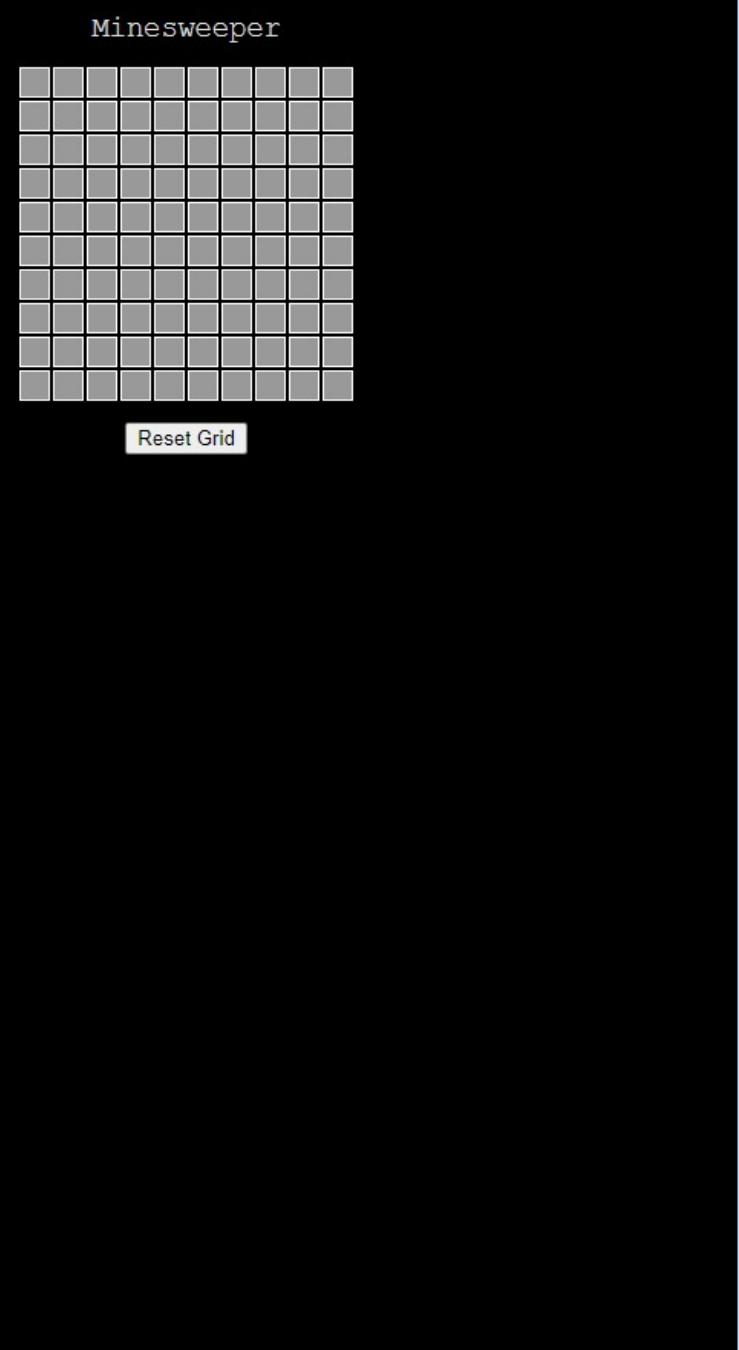
H1 {
    text-align: center;
    font-size: 14pt;
    font-weight: normal;
}
#grid {
    margin-left: auto;
    margin-right: auto;
}

#grid TR TD{
    border:1px solid white;
    background: #999999;
    width:16px;
    height:16px;
    text-align: center;
}

#grid TR TD.clicked {
    background: #333333;
}

#grid TR TD.mine {
    background: #FF0000;
}

BUTTON {
    margin: 12px;
}
```



Your Challenge

Improve this code further to:

- Calculate and display a score in real time (number of cells that have been visited).
- Allow the user to add a cross to a cell (by right clicking?) to indicate the position of a mine.
- Allow the user to choose a difficulty level or change some settings (e.g. Number of mines, size of the grid).
- Add a timer to the game.

Tutorial

Code Challenge

About Us

Privacy Policy

Terms and conditions

© 2021 Copyright • Le Bateau

Practice Coding Pages

>_ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

Digicode - CSS Challenge

Let's Start

In this challenge you are going to use **CSS** to create your own digicode keypad.

Learning Objectives

By completing this challenge you will familiarise yourself with **CSS pseudo-classes**. A pseudo-class is used to define a special state of an element.

For example, it can be used to:

- document.getElementById("objectID").style

For instance:

- Style an element when a user rolls over it.
- Style visited and unvisited hyperlinks differently.
- document.getElementById("myDiv").style.display="none"; //This will make the DIV tag invisible
- Style active elements (for instance when a user clicks on an element, it becomes active).

Let's combine all these techniques to create some tabs:

The syntax of pseudo-classes:

```
selector:pseudo-class { property:value; }
```

For instance to create a roll-over effect for hyperlinks:

```
A:hover { color:#FF0000; }
```

You can also have a pseudo-class for when a link or button is active (being clicked on):

```
A:active { font-weight:bold; }
```

[Find out more on pseudo classes](#)

Let's see the code in practice by completing this challenge.

Your Task

Tweak the code below to create your own look & feel for your "digicode":

```
<!DOCTYPE html>
<html>
<head>
<style>
BODY {
  background-color: #000000;
}

#digipad {
  width: 200px;
  text-align: center;
  margin-left: auto;
  margin-right: auto;
}

.key {
  display: inline-block;
  color: #00FF00;
  padding: 10px 15px 10px 15px;
  margin: 10px;
  box-shadow: 0px 0px 5px #00FF00;
  border-radius: 3px;
}

.key:hover {
  box-shadow: 0px 0px 5px #00FFFF;
  color: #00FFFF;
```



[Tutorial](#)

[Code Challenge](#)

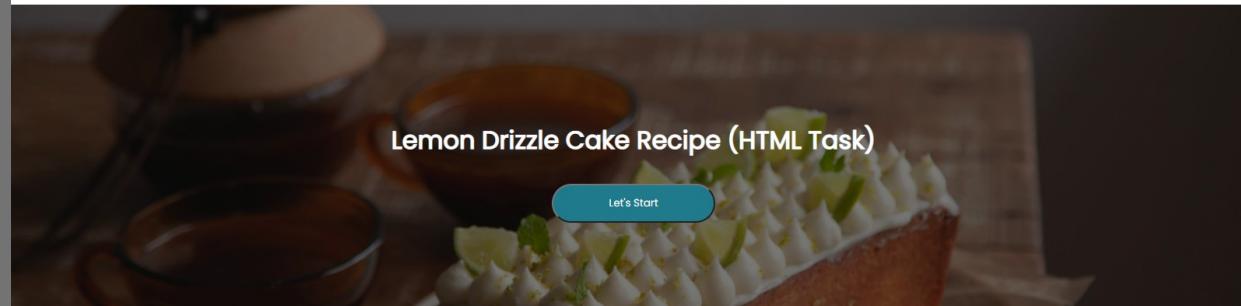
[About Us](#)

[Privacy Policy](#)

[Terms and conditions](#)

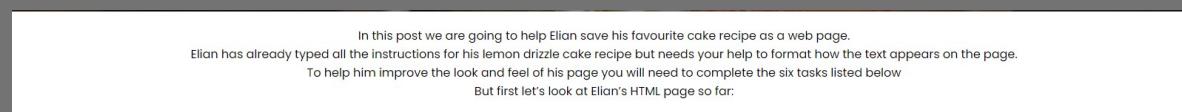
Practice Coding Pages

All the coding pages are more or less the same. So in this page we will break down this webpage.

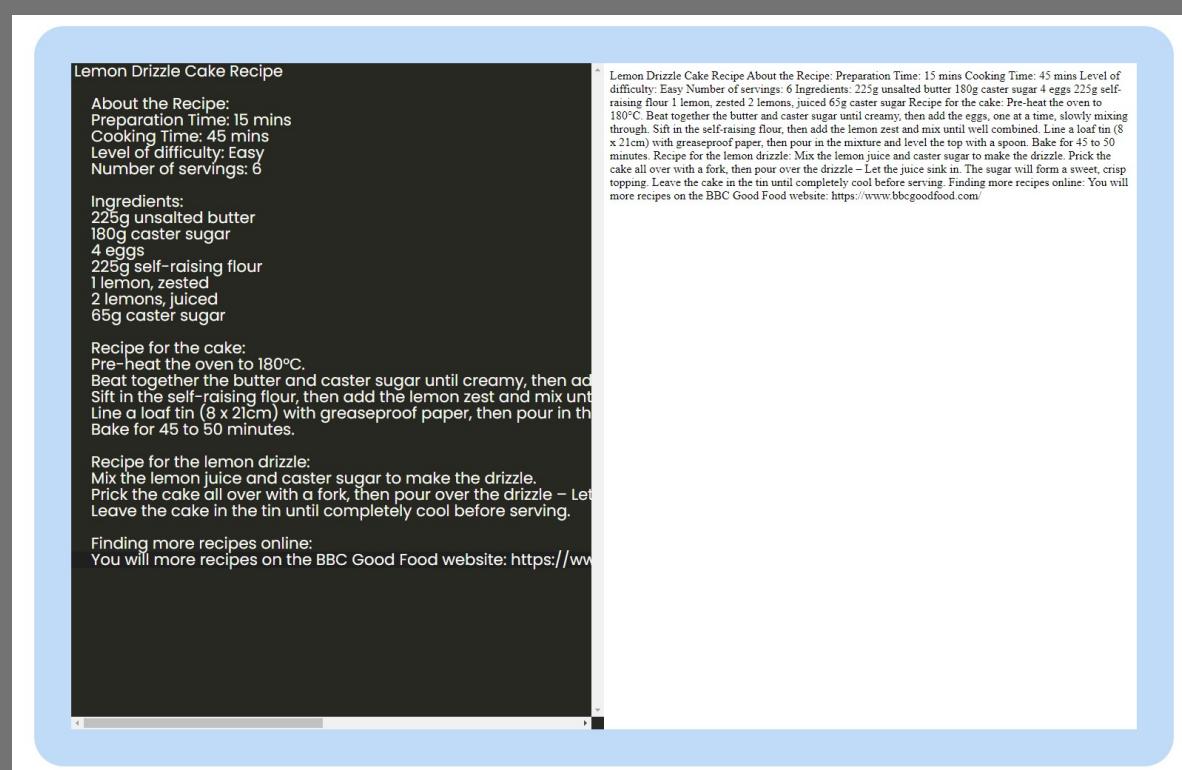


This part is in a div with a class called lemon-banner

- The h1 and button centered
 - when the button is clicked, will jump to the start of the question using an anchor tag
 - z-index is property specifies the stack order of an element
- We made this background fixed when scrolling. To achieve this we:
 - first placed a background image (background image:url("/assets/lemon-cake.jpg");)
 - Position the image using background-position: 50% 100%;
 - used background-repeat: no-repeat; so that it will not repeat the background
 - background-attachment: fixed; is property sets to the background image that will not scroll with the rest of the page
- You can also see that we put a filter on the image to make it a little bit darker. So we :
 - .lemon-banner::after {} is creates a pseudo-element that is the last child of the selected element. It is often used to add cosmetic content(content: ""); to an element with the content property.
 - And a background color of black is applied



Paragraph tag is used and is centered



This is a code-editor where we imported from <https://cdnjs.com/libraries/ace>. We followed this tutorial to initialize the code-editor: https://www.youtube.com/watch?v=jEeYPFFmMcs&t=452s&ab_channel=ShaiUI

overflow: auto; is used to add a scroll bar.

A screenshot of a tabbed interface titled "Your task". The tabs include "Task 1: Headings", "Task 2: Bullet Points", "Task 3: Colours", "Task 4: Formatting text", "Task 5: Hyperlinks", and "Task 6: Your turn". The "Task 1: Headings" tab is active. It contains a code editor with some HTML code and a "Show Answer" button.

To implement this tab we used :

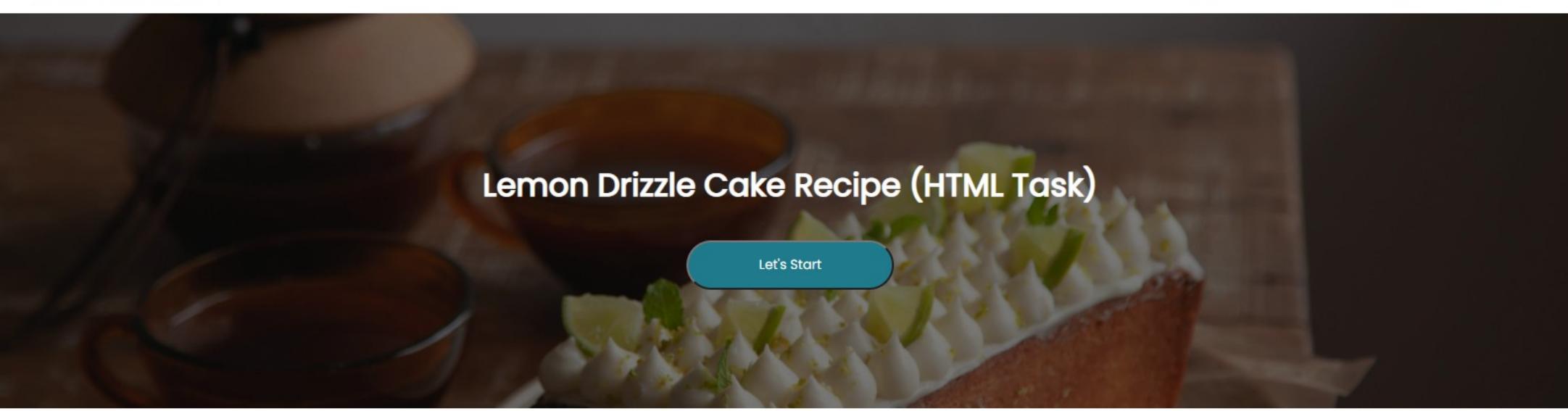
- input type of radio
- A label is used for the input type
- The name attribute is used as the tabs belong in the same group
- To hide the radio button input[type="radio"] display: none; is used
- order: 1 is used to make all the tabs becomes after all the labels
- To hide the content display:none is used
- .myTask input[type='radio']:checked + label + .tab { display: block; }
 - so, when the radio button is checked it will look for the next element which is the label and the next element which is the content of the tab
- border-top-left and right -radius is also used to make the top corners rounded

When the button is clicked, we used javascript to show the answers and hide the "show answer" button and display another button with "hide answers". When this button is clicked it will return to normal.

Final Look

>_ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up



In this post we are going to help Elian save his favourite cake recipe as a web page.
Elian has already typed all the instructions for his lemon drizzle cake recipe but needs your help to format how the text appears on the page.
To help him improve the look and feel of his page you will need to complete the six tasks listed below
But first let's look at Elian's HTML page so far:

Lemon Drizzle Cake Recipe

About the Recipe:

Preparation Time: 15 mins
Cooking Time: 45 mins
Level of difficulty: Easy
Number of servings: 6

Ingredients:

225g unsalted butter
180g caster sugar
4 eggs
225g self-raising flour
1 lemon, zested
2 lemons, juiced
65g caster sugar

Recipe for the cake:

Pre-heat the oven to 180°C.
Beat together the butter and caster sugar until creamy, then add the eggs, one at a time, slowly mixing through. Sift in the self-raising flour, then add the lemon zest and mix until well combined. Line a loaf tin (8 x 21cm) with greaseproof paper, then pour in the mixture and level the top with a spoon. Bake for 45 to 50 minutes.

Recipe for the lemon drizzle:

Mix the lemon juice and caster sugar to make the drizzle.
Prick the cake all over with a fork, then pour over the drizzle – Let the juice sink in.
Leave the cake in the tin until completely cool before serving.

Finding more recipes online:

You will find more recipes on the BBC Good Food website: <https://www.bbcgoodfood.com>

Lemon Drizzle Cake Recipe About the Recipe: Preparation Time: 15 mins Cooking Time: 45 mins Level of difficulty: Easy Number of servings: 6 Ingredients: 225g unsalted butter 180g caster sugar 4 eggs 225g self-raising flour 1 lemon, zested 2 lemons, juiced 65g caster sugar Recipe for the cake: Pre-heat the oven to 180°C. Beat together the butter and caster sugar until creamy, then add the eggs, one at a time, slowly mixing through. Sift in the self-raising flour, then add the lemon zest and mix until well combined. Line a loaf tin (8 x 21cm) with greaseproof paper, then pour in the mixture and level the top with a spoon. Bake for 45 to 50 minutes. Recipe for the lemon drizzle: Mix the lemon juice and caster sugar to make the drizzle. Prick the cake all over with a fork, then pour over the drizzle – Let the juice sink in. The sugar will form a sweet, crisp topping. Leave the cake in the tin until completely cool before serving. Finding more recipes online: You will find more recipes on the BBC Good Food website: <https://www.bbcgoodfood.com>

Your task

Task 1: Headings Task 2: Bullet Points Task 3: Colours Task 4: Formatting text Task 5: Hyperlinks Task 6: Your turn

Task 1: Headings and Paragraphs

In order to give this page more structure we are going to add some headings, subheadings and paragraphs. We will do so using the following HTML tags:

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<h3>Heading 3</h3>

<h4>Heading 4</h4>

<h5>Heading 5</h5>

<h6>Heading 6</h6>

<p>Paragraph</p>

Your task is to update Elian's code by adding some <h1> tags for the main heading

- Lemon Drizzle Cake Recipe

And some <h2> tags for the subheading of the page:

- About the Recipe:
- Ingredients:
- Recipe for the cake:
- Recipe for the lemon drizzle:
- Finding more recipes online:

All other pieces of text should be displayed as paragraphs using <p> tags.

Note that, within a paragraph, we can also force the text to go to the next line using the
 tags.

Show Answer

Tutorial

Code Challenge

About Us

Privacy Policy

Terms and conditions

Terms And conditions Page

> _ Le Bateau

Tutorial Code Challenge About Us Log In Sign Up

Terms and Conditions

Welcome to Le Bateau!

These terms and conditions outline the rules and regulations for the use of Le Bateau's Website, located at <http://www.LeBateau.com/>.

By accessing this website we assume you accept these terms and conditions. Do not continue to use Le Bateau if you do not agree to take all of the terms and conditions stated on this page.

The following terminology applies to these Terms and Conditions, Privacy Statement and Disclaimer Notice and all Agreements: "Client", "You" and "Your" refers to you, the person log on this website and compliant to the Company's terms and conditions. "The Company", "Ourselves", "We", "Our" and "Us", refers to our Company. "Party", "Parties", or "Us", refers to both the Client and ourselves. All terms refer to the offer, acceptance and consideration of payment necessary to undertake the process of our assistance to the Client in the most appropriate manner for the express purpose of meeting the Client's needs in respect of provision of the Company's stated services, in accordance with and subject to, prevailing law of Netherlands. Any use of the above terminology or other words in the singular, plural, capitalization and/or he/she or they, are taken as interchangeable and therefore as referring to same.

Cookies

We employ the use of cookies. By accessing Le Bateau, you agree to use cookies in agreement with the Le Bateau's Privacy Policy.

Most interactive websites use cookies to let us retrieve the user's details for each visit. Cookies are used by our website to enable the functionality of certain areas to make it easier for people visiting our website. Some of our affiliate/advertising partners may also use cookies.

License

Unless otherwise stated, Le Bateau and/or its licensors own the intellectual property rights for all material on Le Bateau. All intellectual property rights are reserved. You may access this from Le Bateau for your own personal use subjected to restrictions set in these terms and conditions.

You must not:

- Republish material from Le Bateau
- Sell, rent or sub-license material from Le Bateau
- Reproduce, duplicate or copy material from Le Bateau
- Redistribute content from Le Bateau

This Agreement shall begin on the date hereof. Our Terms and Conditions were created with the help of the [Terms And Conditions Template](#).

Parts of this website offer an opportunity for users to post and exchange opinions and information in certain areas of the website. Le Bateau does not filter, edit, publish or review Comments prior to their presence on the website. Comments do not reflect the views and opinions of Le Bateau, its agents and/or affiliates. Comments reflect the views and opinions of the person who post their views and opinions. To the extent permitted by applicable laws, Le Bateau shall not be liable for the Comments or for any liability, damages or expenses caused and/or suffered as a result of any use of and/or posting of and/or appearance of the Comments on this website.

Le Bateau reserves the right to monitor all Comments and to remove any Comments which can be considered inappropriate, offensive or causes breach of these Terms and Conditions.

You warrant and represent that:

- You are entitled to post the Comments on our website and have all necessary licenses and consents to do so;
- The Comments do not invade any intellectual property right, including without limitation copyright, patent or trademark of any third party;
- The Comments do not contain any defamatory, libelous, offensive, indecent or otherwise unlawful material which is an invasion of privacy;
- The Comments will not be used to solicit or promote business or custom or present commercial activities or unlawful activity.

You hereby grant Le Bateau a non-exclusive license to use, reproduce, edit and authorize others to use, reproduce and edit any of your Comments in any and all forms, formats or media.

Hyperlinking to our Content

The following organizations may link to our Website without prior written approval:

- Government agencies;
- Search engines;
- News organizations;

Online directory distributors may link to our Website in the same manner as they hyperlink to the Websites of other listed businesses; and

System wide Accredited Businesses except soliciting non-profit organizations, charity shopping malls and charity fundraising groups which may not hyperlink to our Web site.

These organizations may link to our home page, to publications or to other Website information so long as the link: (a) is not in any way deceptive; (b) does not falsely imply sponsorship, endorsement or approval of the linking party and its products and/or services; and (c) fits within the context of the linking party's site.

We may consider and approve other link requests from the following types of organizations:

- commonly-known consumer and/or business information sources;
- dot.com community sites;
- associations or other groups representing charities;
- online directory distributors;
- internet portals;
- accounting, law and consulting firms; and
- educational institutions and trade associations.

We will approve link requests from these organizations if we decide that: (a) the link would not make us look unfavorably to ourselves or to our accredited businesses; (b) the organization does not have any negative records with us; (c) the benefit to us from the visibility of the hyperlink compensates the absence of Le Bateau; and (d) the link is in the context of general resource information.

These organizations may link to our home page so long as the link: (a) is not in any way deceptive; (b) does not falsely imply sponsorship, endorsement or approval of the linking party and its products or services; and (c) fits within the context of the linking party's site.

If you are one of the organizations listed in paragraph 2 above and are interested in linking to our website, you must inform us by sending an e-mail to Le Bateau. Please include your name, your organization name, contact information as well as the URL of your site, a list of any URLs from which you intend to link to our Website, and a list of the URLs on our site to which you would like to link.

Wait 2-3 weeks for a response.

Approved organizations may hyperlink to our Website as follows:

- By use of our corporate name; or
- By use of the uniform resource locator being linked to; or
- By use of any other description of our Website being linked to that makes sense within the context and format of content on the linking party's site.

No use of Le Bateau's logo or other artwork will be allowed for linking absent a trademark license agreement.

iFrames

Without prior approval and written permission, you may not create frames around our Webpages that alter in any way the visual presentation or appearance of our Website.

Content Liability

We shall not be hold responsible for any content that appears on your Website. You agree to protect and defend us against all claims that is rising on your Website. No link(s) should appear on any Website that may be interpreted as libelous, obscene or criminal, or which infringes, otherwise violates, or advocates the infringement or other violation of, any third party rights.

Your Privacy

Please read Privacy Policy

Reservation of Rights

We reserve the right to request that you remove all links or any particular link to our Website. You approve to immediately remove all links to our Website upon request. We also reserve the right to amend these terms and conditions and its linking policy at any time. By continuously linking to our Website, you agree to be bound to and follow these linking terms and conditions.

Removal of links from our website

If you find any link on our Website that is offensive for any reason, you are free to contact and inform us any moment. We will consider requests to remove links but we are not obligated to or so or to respond to you directly.

We do not ensure that the information on this website is correct, we do not warrant its completeness or accuracy; nor do we promise to ensure that the website remains available or that the material on the website is kept up to date.

Disclaimer

To the maximum extent permitted by applicable law, we exclude all representations, warranties and conditions relating to our website and the use of this website. Nothing in this disclaimer will:

- limit or exclude our or your liability for death or personal injury;
- limit or exclude our or your liability for fraud or fraudulent misrepresentation;
- limit any of our or your liabilities in any way that is not permitted under applicable law; or
- exclude any of our or your liabilities that may not be excluded under applicable law.

The limitations and prohibitions of liability set in this Section and elsewhere in this disclaimer: (a) are subject to the preceding paragraph; and (b) govern all liabilities arising under the disclaimer, including liabilities arising in contract, in tort and for breach of statutory duty.

As long as the website and the information and services on the website are provided free of charge, we will not be liable for any loss or damage of any nature.

Tutorial Code Challenge About Us Privacy Policy Terms and conditions

© 2022 Copyright • Le Bateau

In this page we used the following tag:

- <article>
- <h2>
-

All the text is aligned to the center. We used a generator to generate the content for terms and conditions:
<https://www.termsandconditionssample.com/>

Privacy Policy Page

The screenshot shows the privacy policy page for Le Bateau. At the top, there's a navigation bar with links for Tutorial, Code Challenge, About Us, Log In, and Sign Up. The main title is "Privacy Policy for Le Bateau". Below the title, there's a paragraph about the website's privacy practices, mentioning that visitors' information is collected and recorded by Le Bateau. It also states that the policy applies to online activities and is not applicable to offline information. A "Consent" section follows, asking users to agree to the terms. The "Information we collect" section details what personal information is collected, such as name, email, and address, and how it's used for website maintenance and marketing. The "How we use your information" section lists various purposes like providing services, improving the website, and preventing fraud. The "Log Files" section explains that log files track visitors' IP addresses, browser type, and click history. The "Cookies and Web Beacons" section discusses how cookies are used to store visitor preferences and optimize the user experience. The "Advertising Partners Privacy Policies" section informs users that Le Bateau uses third-party ad servers and provides a link to their privacy policies. The "Third Party Privacy Policies" section notes that Le Bateau's policy does not apply to these partners. The "CCPA Privacy Rights (Do Not Sell My Personal Information)" section outlines consumer rights under CCPA, including the right to request disclosure of categories of personal data, deletion of personal data, and the right to opt-out. The "GDPR Data Protection Rights" section lists rights under GDPR, such as the right to rectification, erasure, and data portability. The "Children's Information" section states that Le Bateau prioritizes children's online safety and encourages parents to contact them if they find any inappropriate information. At the bottom, there's a footer with links for Tutorial, Code Challenge, About Us, Privacy Policy (which is the current page), and Terms and conditions, along with a copyright notice for 2022.

In this page we used the following tag:

- <article>
- <h2>
-

All the text is aligned to the center. We used a generator to generate the content for privacy policy:
<https://www.generateprivacypolicy.com/>

