



PLANIT
TEACHERS

Designing and Building a Simple Web Page: An Introduction to HTML, CSS, and JavaScript for 12-Year-Olds

Introduction to Web Development

Welcome to the world of web development! In this exciting project, you will learn how to design and build a simple web page using HTML, CSS, and JavaScript. You will discover how to create a web page that is both visually appealing and functional, and will have the opportunity to express your creativity and showcase your skills.

A large, empty rectangular box with a dashed border, intended for students to draw or write their introduction to web development.

Activity 1: What is Web Development?

Write a short paragraph explaining what web development is and why it's important. Draw a picture of a web page and label its different parts (header, footer, navigation menu, etc.).

A large, empty rectangular box with a dashed border, intended for students to write their paragraph and draw a labeled web page.

HTML Basics

HTML (Hypertext Markup Language) is the standard markup language used to create web pages. It provides the structure and content of a web page, including headings, paragraphs, images, and links.

Activity 2: HTML Tags

Match the following HTML tags with their meanings:

- `<h1>`: _____
- `<p>`: _____
- ``: _____
- `<a>`: _____

Create a Simple HTML Document

Create a simple HTML document using a text editor and add the following elements: a heading, a paragraph of text, and a link to your favorite website.

CSS Basics

CSS (Cascading Style Sheets) is a styling language used to control the layout and visual styling of a web page. It is used to separate the presentation of a web page from its structure, making it easier to maintain and update.

Activity 3: CSS Styles

Create a simple CSS stylesheet and add the following styles:

- Change the background color of the page to blue
- Change the font color of the heading to red
- Add a border to the paragraph of text

Apply CSS Stylesheet

Apply the CSS stylesheet to your HTML document from Activity 2.

JavaScript Basics

JavaScript is a programming language used to add interactivity and dynamic effects to a web page. It is used to create interactive elements, such as buttons, forms, and animations.

Activity 4: JavaScript Interactive Elements

Create a simple JavaScript script that changes the background color of the page when a button is clicked. Add the JavaScript script to your HTML document from Activity 2.



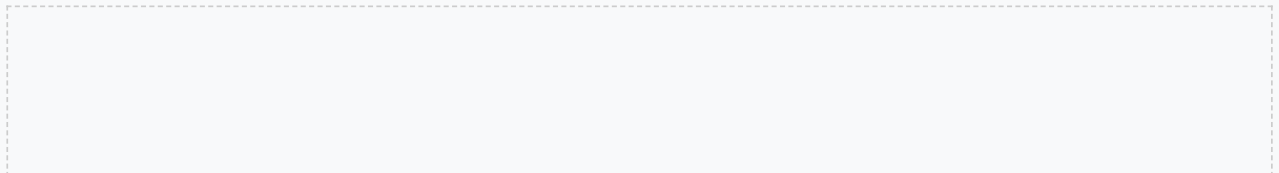
Web Page Design

A well-designed web page should be visually appealing and easy to navigate. It should include a clear structure, including a header, footer, and navigation menu.

Activity 5: Web Page Design

Create a wireframe of a web page that includes the following elements:

- Header with a logo and navigation menu
- Footer with copyright information and social media links
- Main content area with a heading, paragraph of text, and image

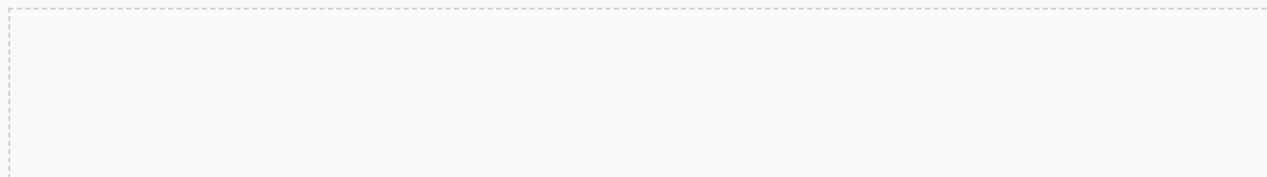


Web Development Process

The web development process involves several stages, including planning, designing, building, testing, and deploying.

Activity 6: Web Development Process

Create a flowchart that illustrates the web development process. Write a short paragraph explaining each stage of the process and why it's important.

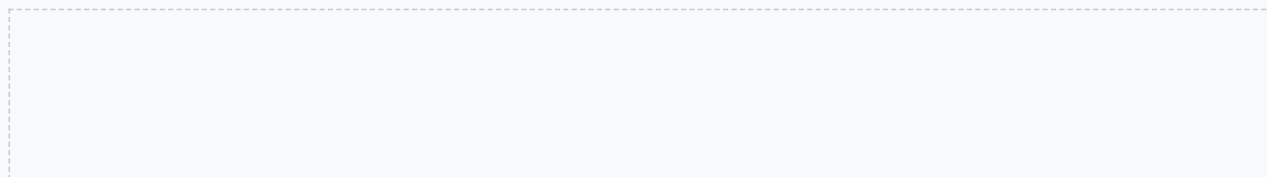


Debugging and Testing

Debugging and testing are crucial stages of the web development process. They ensure that the web page works correctly and is free of errors.

Activity 7: Debugging and Testing

Create a checklist of common errors to look for when debugging a web page. Test your web page from Activity 2 and identify any errors or areas for improvement.



Web Page Deployment

Once a web page is complete, it needs to be deployed to a server so that it can be accessed by the public.

Activity 8: Web Page Deployment

Research different web hosting options and create a list of pros and cons for each. Write a short paragraph explaining the importance of web page deployment and how it's done.

Web Development Tools and Resources

There are many tools and resources available to help with web development, including code editors, design software, and online tutorials.

Activity 9: Web Development Tools and Resources

Research different code editors and design software and create a list of their features and benefits. Write a short paragraph explaining the importance of using the right tools and resources for web development.

Conclusion

Congratulations! You have completed the project and have a solid understanding of HTML, CSS, and JavaScript. You can now design and build your own simple web page and apply your web development skills to real-world scenarios.

Activity 10: Final Project

Create a final project that showcases your understanding of HTML, CSS, and JavaScript. Write a short reflection on what you learned and what you would like to learn more about in the future.

Advanced HTML Concepts

In this section, we will explore more advanced HTML concepts, including semantic elements, microformats, and ARIA attributes. These concepts will help you create more accessible and meaningful web pages.

Example: Using Semantic Elements

Semantic elements are HTML elements that provide meaning to the structure of a web page. They help search engines and screen readers understand the content and purpose of a web page. For example, the `<header>` element is used to define the header section of a web page, while the `<footer>` element is used to define the footer section.

Activity 11: Creating a Semantic Web Page

Create a new HTML document and add the following semantic elements: `<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, `<aside>`, `<footer>`. Add some sample content to each element to demonstrate their purpose.

CSS Layout and Positioning

In this section, we will explore CSS layout and positioning techniques, including flexbox, grid, and absolute positioning. These techniques will help you create complex and responsive layouts for your web pages.

Example: Using Flexbox

Flexbox is a CSS layout mode that allows you to create flexible and responsive layouts. It is particularly useful for creating navigation menus, image galleries, and other types of content that require a flexible layout. For example, you can use the `display: flex;` property to create a flexible container, and the `flex-direction:` property to control the direction of the flex items.

Activity 12: Creating a Flexbox Layout

Create a new HTML document and add a container element with the class `flex-container`. Add five child elements with the class `flex-item`. Use CSS to create a flexbox layout that displays the child elements in a row, with equal spacing between each item.

Copyright 2023 Planit Teachers. All rights reserved.

JavaScript and DOM Manipulation

In this section, we will explore JavaScript and DOM manipulation techniques, including selecting and manipulating elements, adding event listeners, and updating the DOM. These techniques will help you create interactive and dynamic web pages.

Example: Selecting and Manipulating Elements

JavaScript provides several methods for selecting and manipulating elements, including `document.querySelector()` and `document.getElementById()`. You can use these methods to select elements and update their content, attributes, and styles. For example, you can use the `innerHTML` property to update the content of an element, or the `style` property to update the styles of an element.

Activity 13: Selecting and Manipulating Elements

Create a new HTML document and add a paragraph element with the id `my-paragraph`. Use JavaScript to select the paragraph element and update its content to "Hello, World!". Use the `console.log()` method to log the updated content to the console.

Responsive Web Design

In this section, we will explore responsive web design techniques, including media queries, flexible grids, and images. These techniques will help you create web pages that adapt to different screen sizes and devices.

Example: Using Media Queries

Media queries are a powerful tool for creating responsive web pages. They allow you to apply different styles based on the screen size, device type, and other factors. For example, you can use the `@media` rule to apply different styles for desktop, tablet, and mobile devices.

Activity 14: Creating a Responsive Web Page

Create a new HTML document and add a container element with the class `responsive-container`. Use CSS to create a responsive layout that adapts to different screen sizes. Add media queries to apply different styles for desktop, tablet, and mobile devices.

Accessibility and Usability

In this section, we will explore accessibility and usability techniques, including semantic HTML, ARIA attributes, and user testing. These techniques will help you create web pages that are accessible and usable for all users.

Example: Using ARIA Attributes

ARIA attributes provide a way to add accessibility information to HTML elements. They help screen readers and other assistive technologies understand the purpose and behavior of elements. For example, you can use the `aria-label` attribute to provide a text description of an element, or the `aria-expanded` attribute to indicate the state of a collapsible element.

Activity 15: Creating an Accessible Web Page

Create a new HTML document and add a navigation menu with the class `accessible-menu`. Use ARIA attributes to provide accessibility information for the menu items. Test the menu using a screen reader to ensure that it is accessible and usable.

Copyright 2025 Planit Teachers. All rights reserved.

Web Development Best Practices

In this section, we will explore web development best practices, including coding standards, version control, and testing. These practices will help you write clean, maintainable, and efficient code.

Example: Using Version Control

Version control systems, such as Git, provide a way to track changes to your code and collaborate with others. They help you manage different versions of your code and ensure that changes are properly tested and reviewed. For example, you can use the `git init` command to initialize a new Git repository, or the `git commit` command to commit changes to your code.

Activity 16: Using Version Control

Create a new Git repository and add a sample web page to it. Use the `git add` command to stage the changes, and the `git commit` command to commit the changes. Use the `git log` command to view the commit history.



PLANIT
TEACHERS

Designing and Building a Simple Web Page: An Introduction to HTML, CSS, and JavaScript for 12-Year-Olds

Introduction to Web Development

Welcome to the world of web development! In this exciting project, you will learn how to design and build a simple web page using HTML, CSS, and JavaScript. You will discover how to create a web page that is both visually appealing and functional, and will have the opportunity to express your creativity and showcase your skills.

Activity 1: What is Web Development?

Write a short paragraph explaining what web development is and why it's important. Draw a picture of a web page and label its different parts (header, footer, navigation menu, etc.).

HTML Basics

HTML (Hypertext Markup Language) is the standard markup language used to create web pages. It provides the structure and content of a web page, including headings, paragraphs, images, and links.

Activity 2: HTML Tags

Match the following HTML tags with their meanings:

- `<h1>`: _____
- `<p>`: _____
- ``: _____
- `<a>`: _____

Create a Simple HTML Document

Create a simple HTML document using a text editor and add the following elements: a heading, a paragraph of text, and a link to your favorite website.

CSS Basics

CSS (Cascading Style Sheets) is a styling language used to control the layout and visual styling of a web page. It is used to separate the presentation of a web page from its structure, making it easier to maintain and update.

Activity 3: CSS Styles

Create a simple CSS stylesheet and add the following styles:

- Change the background color of the page to blue
- Change the font color of the heading to red
- Add a border to the paragraph of text

Apply CSS Stylesheet

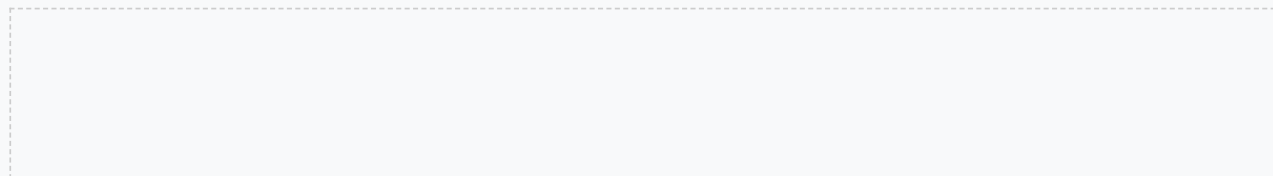
Apply the CSS stylesheet to your HTML document from Activity 2.

JavaScript Basics

JavaScript is a programming language used to add interactivity and dynamic effects to a web page. It is used to create interactive elements, such as buttons, forms, and animations.

Activity 4: JavaScript Interactive Elements

Create a simple JavaScript script that changes the background color of the page when a button is clicked. Add the JavaScript script to your HTML document from Activity 2.



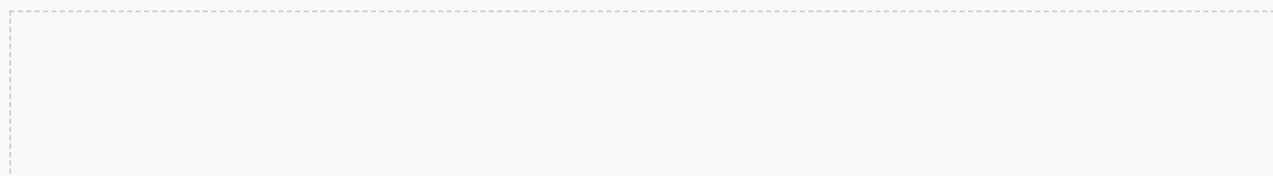
Web Page Design

A well-designed web page should be visually appealing and easy to navigate. It should include a clear structure, including a header, footer, and navigation menu.

Activity 5: Web Page Design

Create a wireframe of a web page that includes the following elements:

- Header with a logo and navigation menu
- Footer with copyright information and social media links
- Main content area with a heading, paragraph of text, and image



Web Development Process

The web development process involves several stages, including planning, designing, building, testing, and deploying.

Activity 6: Web Development Process

Create a flowchart that illustrates the web development process. Write a short paragraph explaining each stage of the process and why it's important.

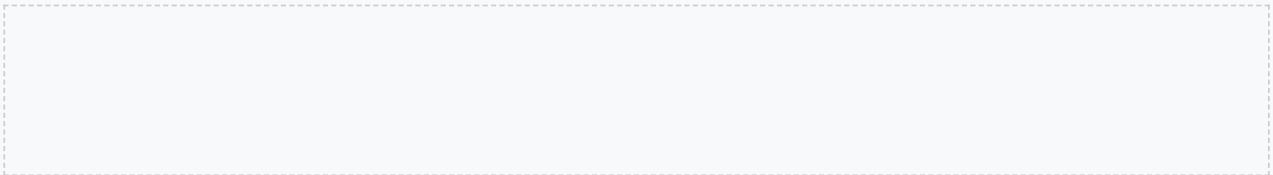


Debugging and Testing

Debugging and testing are crucial stages of the web development process. They ensure that the web page works correctly and is free of errors.

Activity 7: Debugging and Testing

Create a checklist of common errors to look for when debugging a web page. Test your web page from Activity 2 and identify any errors or areas for improvement.



Web Page Deployment

Once a web page is complete, it needs to be deployed to a server so that it can be accessed by the public.

Activity 8: Web Page Deployment

Research different web hosting options and create a list of pros and cons for each. Write a short paragraph explaining the importance of web page deployment and how it's done.

Web Development Tools and Resources

There are many tools and resources available to help with web development, including code editors, design software, and online tutorials.

Activity 9: Web Development Tools and Resources

Research different code editors and design software and create a list of their features and benefits. Write a short paragraph explaining the importance of using the right tools and resources for web development.

Conclusion

Congratulations! You have completed the project and have a solid understanding of HTML, CSS, and JavaScript. You can now design and build your own simple web page and apply your web development skills to real-world scenarios.

Activity 10: Final Project

Create a final project that showcases your understanding of HTML, CSS, and JavaScript. Write a short reflection on what you learned and what you would like to learn more about in the future.

