The_Odin_Project

December 20, 2024

Conversion to PDF: * install nbconvert * install mactex / xelatex * jupyter nbconvert –to pdf The_Odin_Project.ipynb

Conversion to PDF (with images): * Create a VSCode python virtual env in same directory * Select it as interpreter * Go to venv folder, install nbconvert * Export to HTML using VSCODE, print to PDF.

1 Foundations:

1.0.1 Working of the Internet:

Working of the internet

- Routers: Instead of connecting computers to each other in a forest of series connections, a router acts as a central point which multiplexes information into the destination computer.
- Modem: It connects your home network to your telephone line, thus connecting your home snetwork to the rest of the world's, home network, forming the internet.
- ISP: They have routers that are connected to other ISP routers.

Domain Name

- Domain Name: They are interchangable with the IP adress of a particular computer connected to the network. Ex - google can be reached via www.google.com or 142.250.190.78.
- Structure of Domain Name: subdomain.domain_name.Top_Level_Domain, ex - www.developer.mozilla.com
- Finding Domain Registrar: ex - whois google.com
- DNS request: When you type a web adress such as mozilla.com, the browser will request the DNS server the IP of the computer behind the mozilla.com, so that the browser can connect to that IP.

The Web in Operation

- Servers vs Clients : Home computers are not servers as they are not connected to the internet directly, instead are routed through ISPs and thus are called clients. Servers are directly connected.

Web elements

- Web page: A document that can be displayed on a browser.

- Website: A collection of webpages.
- Web server: A computer hosting a website.
- Search engine: A service that helps you find other web pages.
- URLS: URLs
- HTTP: HTTP

Web Server

Components: * From the hardware standpoint, the webserver stores the server software and the website files.

- From the software side, the webserver contrls how web users access the hosted file. Ex - HTTP server: Accessed from the domain name/server IP, it controls URLs and HTTP requests.
- HTTP: To access contents of the website, the browser sends across a HTTP request to the web server. The HTTP server accepts the request and sends the contents to the broswer through HTTP too. HTTP

Server Types: * Static: Computer + HTTP server. The hosted files are served as is.

- Dynamic: Computer + HTTP server + application server and databases. The application server updates the hosted files before sending content to the browser via HTTP server (it processes content on the fly).

1.0.2 Git, Github and CLI

Git basics

- Always use CLI for github
- Always clone repo from github, avoid init.
- After Pushing to github, git status should output: "Your branch is up to date with 'origin/main'. nothing to commit, working tree clean".
- For commit messages, metion the change and why the change as subject and body respecitvely in an active voice and keep a blank line between the subject and body (remove -m for this).

[ ]:
```
#File Creation:
touch filename

#Directory Creation:
mkdir dirname
mkdir -p parentdir/childdir #list nested directories

#List files:
ls
ls -l #list with details
ls -a #list hidden files

#Removing Files:
```

```
rm filename

#Removing Directory:
rm -r dirname

#Renaming a file or directory
rm oldname newname

#Moving file or directory
mv source destination

#Viewing Contents of a file:
cat filename
```

[ ]:
```
#Change directory
cd dirname

#Move to Parent Directory
cd ..

#Get current directory path
pwd

#Clearing Terminal
clear

#Go to home directory
cd ~

#Go to directory inside home directory
cd ~/Documents
```

[ ]:
```
#Initialising Repo
git init

#Clone a repo (Create on github and then clone)
git clone repo_url

#Checking status of repo
git status

#Add files for staging
git add filename
git add . #Add all files

#Commit Changes
git commit -m "Commit message"
```

```
#Check commit history
git log

#Push changes to remote repository
git push origin branch_name

#Pull changes from remote repo
git pull origin branch_name
```

1.0.3 HTML

HTML Intro

- Extra whitespaces are ommitted by HTML when rendering, but when inner HTML is accessed by JS, the extra whitespaces will be included too.
- To mention elements such as < p> without rendering, use their char reference eqvt such as &lt for < . Ex - <p>

HTML Tags

[ ]:
```
<!-- Basic Document Structure -->
<!DOCTYPE html> <!-- Defines the document type as HTML5 -->
<html> <!-- Root element of the HTML document -->
  <head> <!-- Container for metadata -->
    <title>Page Title</title> <!-- Sets the title of the webpage -->
    <meta charset="UTF-8"> <!-- Specifies the character encoding -->
    <meta name="viewport" content="width=device-width, initial-scale=1.0"> <!--␣
 ↪Responsive design meta tag -->
  </head>
  <body> <!-- Main content of the document -->

    <!-- Headings -->
    <h1>Main Heading</h1> <!-- Largest heading -->
    <h2>Subheading</h2> <!-- Smaller heading -->
    <h3>Sub-subheading</h3> <!-- Even smaller heading -->

    <!-- Text and Formatting -->
    <p>This is a paragraph.</p> <!-- Defines a paragraph -->
    <strong>Bold text</strong> <!-- Makes text bold -->
    <em>Italic text</em> <!-- Makes text italic -->
    <u>Underlined text</u> <!-- Underlines text -->
    <br> <!-- Line break -->
    <hr> <!-- Horizontal rule -->

    <!-- Links -->
    <a href="https://example.com">Visit Example</a> <!-- Hyperlink -->
    <a href="#section">Jump to Section</a> <!-- Internal link -->
```

```
  <!-- Images -->
  <img src="image.jpg" alt="Description" width="300"> <!-- Displays an image␣
↪-->

  <!-- Lists -->
  <ul> <!-- Unordered list -->
    <li>List item 1</li>
    <li>List item 2</li>
  </ul>
  <ol> <!-- Ordered list -->
    <li>First item</li>
    <li>Second item</li>
  </ol>

  <!-- Tables -->
  <table border="1"> <!-- Creates a table with borders -->
    <thead>
      <tr>
        <th>Header 1</th> <!-- Table header -->
        <th>Header 2</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>Row 1, Cell 1</td> <!-- Table data -->
        <td>Row 1, Cell 2</td>
      </tr>
    </tbody>
  </table>

  <!-- Forms -->
  <form action="/submit" method="POST">
    <label for="name">Name:</label> <!-- Label for input -->
    <input type="text" id="name" name="name"> <!-- Text input -->
    <button type="submit">Submit</button> <!-- Submit button -->
  </form>

  <!-- Media -->
  <video controls width="400">
    <source src="video.mp4" type="video/mp4">
    Your browser does not support video.
  </video> <!-- Embeds a video -->
  <audio controls>
    <source src="audio.mp3" type="audio/mp3">
    Your browser does not support audio.
  </audio> <!-- Embeds audio -->
```

```
    <!-- Semantic Elements -->
    <header>Site Header</header> <!-- Defines a header -->
    <nav>
      <ul>
        <li><a href="#">Home</a></li>
        <li><a href="#">About</a></li>
      </ul>
    </nav> <!-- Navigation menu -->
    <main> <!-- Main content of the page -->
      <article>
        <h2>Article Title</h2>
        <p>Content of the article.</p>
      </article>
    </main>
    <footer>Site Footer</footer> <!-- Footer of the page -->

    <!-- Interactive Elements -->
    <button onclick="alert('Hello!')">Click Me</button> <!-- Button with a␣
↪click event -->
    <details>
      <summary>More Info</summary>
      <p>Hidden content revealed upon click.</p>
    </details>

    <!-- Multimedia Embeds -->
    <iframe src="https://example.com" width="600" height="400"></iframe> <!--␣
↪Embeds another webpage -->
    <embed src="file.pdf" width="500" height="400"> <!-- Embeds a PDF or other␣
↪media -->

    <!-- Other Tags -->
    <span>Inline element</span> <!-- Generic inline container -->
    <div>Block element</div> <!-- Generic block container -->
    <mark>Highlighted text</mark> <!-- Highlights text -->
    <code>console.log('Hello, world!');</code> <!-- Displays code -->
    <pre>
      Preformatted text
      with line breaks.
    </pre> <!-- Preformatted text -->

  </body>
</html>
```

[ ]:
```
<!-- Anchor Tags: Hyperlinks that open on the same page -->
<a href="https://www.theodinproject.com/about">About TOP</a>
```

```
<!-- Anchor Tags Hyperlinks that open on a new page, with rel attributes␣
 ↪providing isolation between the pages,
preventing the opened page from accessing the previous page -->
<a href="https://www.theodinproject.com/about" target="_blank" rel="noopener␣
 ↪noreferrer">About The Odin Project</a>

<!-- Absoulte links connect to webpages outside the website-->
scheme://domain/path
ex - https://www.theodinproject.com/about

<!-- Relative links connect to internal webpages-->
about.html (Same directory)
pages/about.html (Pages folder is on the same directory)
./pages/about.html (Use ./ to ensure the file is searched relative to current␣
 ↪directory)

../images/images/about.html (To access through parent directory)

<!-- Images  -->
<img src="https://www.theodinproject.com/mstile-310x310.png" alt="Odin"␣
 ↪height="310" width="200">
```

1.0.4 CSS

External CSS

[ ]:
```
<head>
    <link rel="stylesheet" href="styles.css">
</head>

<body>
    <!-- The div has multiple classes and multiple ids defined. -->
    <div class="list main" id="pro forma"></div>
</body>
```

Selectors:

HTML elements on which CSS rules are applied.

Prefference for clashing rules:
* Rules with more matching class/id selectors are considered. * 1 ID > n no. of class selectors > n no. of type selector > n no. of universal selector. * ID + 1 class selector > ID * Defined rule > Inherited Rule * Rule defined last on the CSS page is the tiebreaker if all other conditions are the same.

[ ]:
```
/* Universal selectors to select all the elemetns */
* {
    color: purple;
  }
```

```
/* Type selector to select elements of a given type */
div {
  color: white;
}

/* Class selectors <div class = "alert-text">Hi</div>  */
.alert-text {
  color: red;
}

/* Id selectors: same as class selectors but cannot be reused */
#title {
  background-color: red;
}

/*Chaining selectors: If a header tag has the class subsection we can
apply rules specifically to it by: */
.subsection.header {
  color: red;
}

/*Grouping Selectors: for elements that use the same styling */
.read,
.unread {
  color: white;
  background-color: black;
}
.read {
  font-size: large;
}
.unread {
  font-size: small;
}

/*Descendant selector: If a element lies within a div of class ancestor,
and has the class content, the rule will apply only if both conditions
are satisfied */
.ancestor .contents {
  color: red;
}
```

CSS Properties

[ ]:
```
p {
/* Box Model */
margin: 20px; /* Adds 20px space around the element */
```

```
border: 2px solid black; /* 2px black solid border */
padding: 15px; /* Adds 15px space inside the element's border */
width: 200px; /* Sets the element's width to 200px */
height: 100px; /* Sets the element's height to 100px */
box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.5); /* Adds a shadow to the element */
border-radius: 10px; /* Rounds the corners of the element */

/* Typography */
font-family: Arial, sans-serif; /* Sets the font to Arial */
font-size: 16px; /* Text size is 16px */
font-weight: bold; /* Makes text bold */
line-height: 1.5; /* Sets line spacing to 1.5 times the text size */
color: red; /* Makes text red */
text-align: center; /* Centers the text horizontally */
text-decoration: underline; /* Underlines the text */
letter-spacing: 2px; /* Adds 2px spacing between characters */
word-spacing: 5px; /* Adds 5px spacing between words */
text-transform: uppercase; /* Transforms text to uppercase */
white-space: nowrap; /* Prevents text from wrapping */

/* Background */
background-color: lightblue; /* Sets the background color */
background-image: url('example.jpg'); /* Sets a background image */
background-size: cover; /* Scales the background image to cover the element */
background-repeat: no-repeat; /* Prevents the background from repeating */
background-position: center; /* Centers the background image */

/* Layout */
display: flex; /* Makes the container a flexbox */
flex-direction: column; /* Arranges flex items in a column */
justify-content: center; /* Aligns flex items vertically */
align-items: center; /* Aligns flex items horizontally */
position: absolute; /* Positions element relative to its nearest positioned␣
 ↪ancestor */
top: 10px; left: 20px; /* Moves element 10px down and 20px right */
z-index: 10; /* Places the element above others with lower z-index */
overflow: hidden; /* Hides overflowing content */
visibility: hidden; /* Hides the element (still takes up space) */
opacity: 0.5; /* Makes the element semi-transparent */

/* Animation */
transition: all 0.3s ease-in-out; /* Adds smooth transitions for property␣
 ↪changes */
transform: scale(1.2); /* Scales the element by 1.2 times */
animation: fadeIn 2s ease-in; /* Adds an animation to the element */

/* Lists */
```

```
list-style: none; /* Removes bullets from lists */
list-style-type: circle; /* Sets list markers to circles */

/* Grid */
display: grid; /* Makes the container a grid */
grid-template-columns: 1fr 1fr; /* Defines two equal-width columns */
gap: 10px; /* Adds space between grid items */

/* Miscellaneous */
cursor: pointer; /* Changes the cursor to a pointer on hover */
clip-path: circle(50%); /* Clips the element into a circular shape */
box-sizing: border-box; /* Includes padding and border in width and height␣
 ↪calculations */
user-select: none; /* Prevents text selection */
}
```

The Box Model

Think of everything on the webpage to be a box.

Padding -> Increase space between the border and the content of a box.
Border -> Adds space b/w margin and padding.
Margin -> Increases the space between the border of a box and adjacent ones.