The document covers essential HTML commands and Selenium usage for automating web tasks in Python. Here's a detailed breakdown of each section:

### 1. \*\*Basic HTML Commands\*\*

These commands (or tags) are used to structure a web page. Here are the most commonly used ones:

- `<!DOCTYPE html>`: Declares the document as an HTML5 document, letting the browser know the version of HTML in use.

- `<html></html>`: This is the root element that encloses the entire HTML content of the web page.

- `<head></head>`: Contains meta-information like the title of the page, character set, and other metadata.

```html

<head>

<title>Page Title</title>

<meta charset="UTF-8">

</head>

```

- `<body></body>`: This is where all the visible content on the webpage goes.

- \*\*Heading Tags\*\* `<h1></h1>` to `<h6></h6>`: Defines headings, with `<h1>` being the largest and `<h6>` the smallest.

- \*\*Paragraph Tag\*\* `<p></p>`: Creates a paragraph.

```html

<p>This is a paragraph.</p>

```

- \*\*Anchor Tag\*\* `<a></a>`: Creates a clickable link. The `href` attribute specifies the destination URL.

```html

<a href="https://www.example.com">Click here</a>

```

- \*\*Image Tag\*\* `<img>`: Adds an image. `src` specifies the image source, and `alt` provides alternative text.

```html

<img src="image.jpg" alt="Description">

```

- \*\*Lists\*\*:

- `<ul></ul>`: Creates an unordered list (bulleted).

- `<ol></ol>`: Creates an ordered list (numbered).

- `<li></li>`: Defines a list item.

- \*\*Table Tag\*\* `<table></table>`: Creates a table. Inside it, you use:

- `<tr>` for table rows.

- `<th>` for table headers.

- `<td>` for table data cells.

- \*\*Div and Span\*\*:

- `<div></div>`: Creates a block-level section, useful for grouping content.

- `<span></span>`: Creates an inline section of content.

- \*\*Form Tag\*\* `<form></form>`: Defines a form to collect user input.

```html

<form action="/submit" method="post">

<label for="name">Name:</label>

<input type="text" id="name" name="name">

<input type="submit" value="Submit">

</form>

```

### 2. \*\*Web Elements in HTML\*\*

Web elements are interactive components on a webpage:

- \*\*Form Elements\*\*:

- Text Box: `<input type="text">` for user text input.

- Password Box: `<input type="password">` for secure password input.

- Buttons: `<button>` and `<input type="submit">`.

- Dropdown: `<select>` for selection from options.

- Radio Buttons: `<input type="radio">` for choosing one option from a set.

- Checkboxes: `<input type="checkbox">` for selecting multiple options.

- \*\*Links and Navigation\*\*:

- `<a>` creates a hyperlink for navigation between pages.

- \*\*Media Elements\*\*:

- `<img>` for images.

- `<video>` and `<audio>` for embedding media files.

- \*\*Structural Elements\*\*:

- Paragraphs, headings, lists, and tables.

- \*\*Interactive Elements\*\*:

- `<canvas>` for drawing graphics.

- `<iframe>` for embedding other web pages within a page.

### 3. \*\*Selenium Automation with Python\*\*

Selenium is a tool for automating web tasks, and Python provides bindings to work with it. Here's a summary of common Selenium commands and code snippets.

- \*\*Opening a Browser and Searching Google\*\*:

```python

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from webdriver\_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(ChromeDriverManager().install()) # Launch Chrome

driver.get("https://www.google.com") # Go to Google

search\_box = driver.find\_element("name", "q") # Find search box by name attribute

search\_box.send\_keys("Python Selenium") # Enter search term

search\_box.send\_keys(Keys.RETURN) # Press Enter

```

- \*\*Taking a Screenshot\*\*:

```python

driver.save\_screenshot("screenshot.png") # Save screenshot of current page

```

- \*\*Finding Web Elements\*\*:

Different methods in Selenium to locate elements on the page:

1. `find\_element\_by\_id()`: Finds an element by its `id` attribute.

```python

element = driver.find\_element\_by\_id("menu-item-10")

```

2. `find\_element\_by\_name()`: Finds an element by its `name` attribute.

```python

element = driver.find\_element\_by\_name("s")

```

3. `find\_element\_by\_class\_name()`: Finds an element by its class name.

```python

element = driver.find\_element\_by\_class\_name("exampleClass")

```

4. `find\_element\_by\_tag\_name()`: Finds an element by its tag name.

```python

element = driver.find\_element\_by\_tag\_name("h1")

```

5. `find\_element\_by\_xpath()`: Finds an element using XPath.

```python

element = driver.find\_element\_by\_xpath("//h1")

```

6. `find\_element\_by\_css\_selector()`: Finds an element using a CSS selector.

```python

element = driver.find\_element\_by\_css\_selector("h1.exampleClass")

```

Each method allows you to interact with elements on the page, like clicking buttons, entering text, and more.

### Conclusion

This document provides an overview of basic HTML tags and how to interact with web pages using Selenium and Python. It covers everything from structuring a webpage to automating web tasks with code.