Front End Technologies
CSS - Day 6

Agenda

- **Sibling Selector**
- **Examples on selectors**

SIBLING SELECTOR: Sibling selector further classified into two types

- General sibling
- Adjacent sibling

Let's understand sibling selector with an example.

index.html

```html
<!DOCTYPE html>
<html>

<head>
    <title>Example</title>
    <style type="text/css">

    </style>
</head>

<body>
    <h1>H1 outside div</h1>
    <p>p1 outside div</p>
    <div>
        <h2>H2 inside div</h2>
        <p>P1 inside div</p>
        <p>P2 inside div</p>
        <h3>H3 inside div</h3>
    </div>
    <p>P2 outside div</p>
</body>

</html>
```
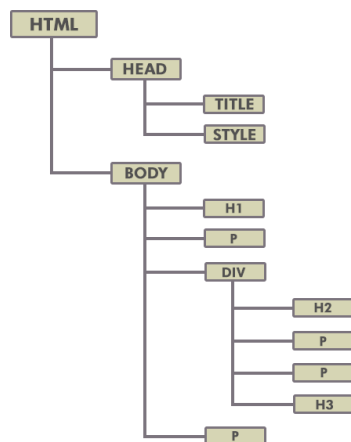
Tree representation:

In the above example, h1, p, div and p tag which is directly inside body tag is known as sibling, h2,p,p and h3 tag which is directly inside div tag is also known as sibling. These sibling which belongs to same parent is only known as General sibling. Now let's understand how to style these General sibling using css.

Example: Style all the general sibling of h1 and h2 tag which are p tag

index.html

```html
<!DOCTYPE html>
<html>

<head>
    <title>Example</title>
    <style type="text/css">
        h1 ~ p{
            background-color: orange;
        }
        h2 ~ p{
            background-color: orange;
        }
    </style>
</head>

<body>
    <h1>H1 outside div</h1>
    <p>p1 outside div</p>
    <div>
        <h2>H2 inside div</h2>
        <p>P1 inside div</p>
        <p>P2 inside div</p>
        <h3>H3 inside div</h3>
    </div>
    <p>P2 outside div</p>
</body>

</html>
```

Output:

H1 outside div

p1 outside div

H2 inside div

P1 inside div

P2 inside div

H3 inside div

P2 outside div
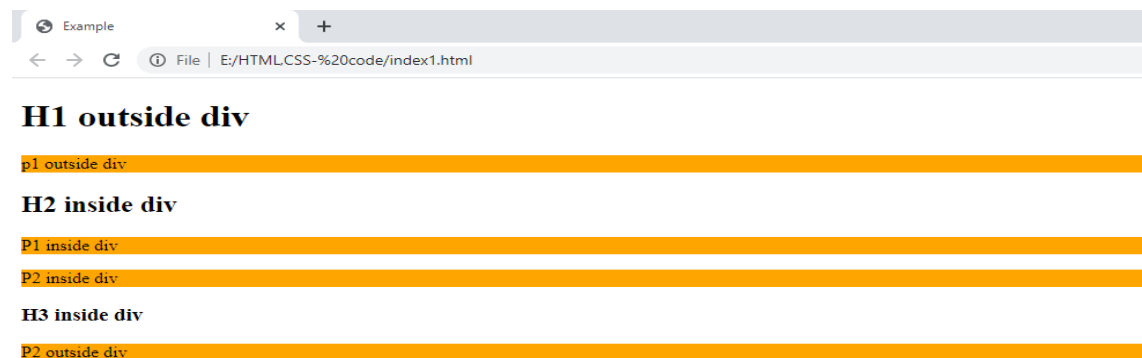
In this example, we applied styling for p tag which are **general sibling** of h1 and h2 tag. The general sibling selector selects all elements that are siblings of a specified element. The combinator used for general sibling is **~**.

Adjacent sibling: The adjacent sibling selector is used to select an element that is directly after another specific element. Sibling elements must have the same parent element, and "adjacent" means "immediately following". Let's now understand adjacent sibling with an example.

Example-1: Style the p tag which is adjacent sibling of h1 and h2 tag.

```
<!DOCTYPE html>
<html>

<head>
    <title>Example</title>
    <style type="text/css">
        h1 + p{
            background-color: orange;
        }
        h2 + p{
            background-color: cyan;
        }
    </style>
</head>

<body>
    <h1>H1 outside div</h1>
    <p>p1 outside div</p>
    <div>
        <h2>H2 inside div</h2>
        <p>P1 inside div</p>
        <p>P2 inside div</p>
        <h3>H3 inside div</h3>
    </div>
    <p>P2 outside div</p>
</body>

</html>
```

Output:

H1 outside div

p1 outside div

H2 inside div

P1 inside div

P2 inside div

H3 inside div

P2 outside div
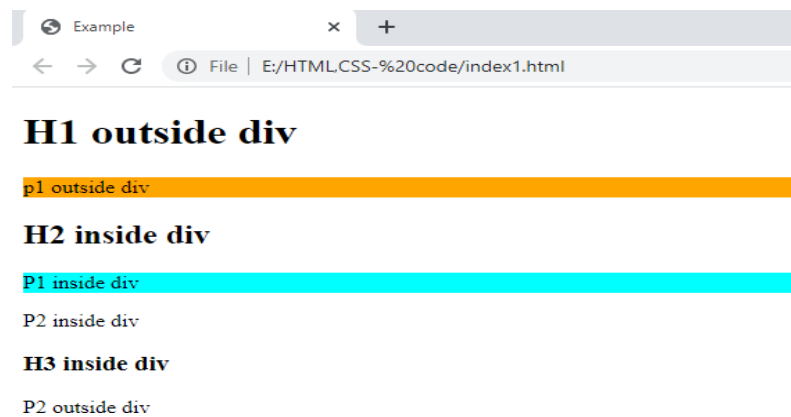
In this example, styling is applied to p tag which is immediate sibling of h1 and h2 tag. The combinatory used for adjacent sibling is +.

Example-2: color the ordered list

Index.html

```html
<!DOCTYPE html>
<html>
<head>
    <title>Example</title>
    <link rel="stylesheet" type="text/css" href="index.css">
</head>
<body>
    <h1>Lists Available on the webpage</h1>
    <ol class="firstlist">
        <li>Popular Languages</li>
        <li>Popular Frameworks</li>
        <li>Highest Paying Jobs</li>
    </ol>
    <h3>Popular Languages</h3>
    <ul>
        <li class="first">C</li>
        <li>Java</li>
        <li>Python</li>
        <li>C++</li>
        <li>C#</li>
        <li>JavaScript</li>
    </ul>
    <h4>Popular Framework</h4>
    <ul>
        <li class="first">Spring</li>
        <li>Django</li>
        <li>Node.Js</li>
        <li>.NET core</li>
        <li>Flask</li>
        <li>React.JS</li>
    </ul>
    <h3>Popular Framework</h3>
    <ul>
        <li class="first">Devops Engineer</li>
        <li>Full Stack Developer</li>
        <li>Data Scientist</li>
        <li>AI-ML Engineer</li>
        <li>Cloud Engineer</li>
        <li>Cyber Security Expert</li>
    </ul>
</body>
```
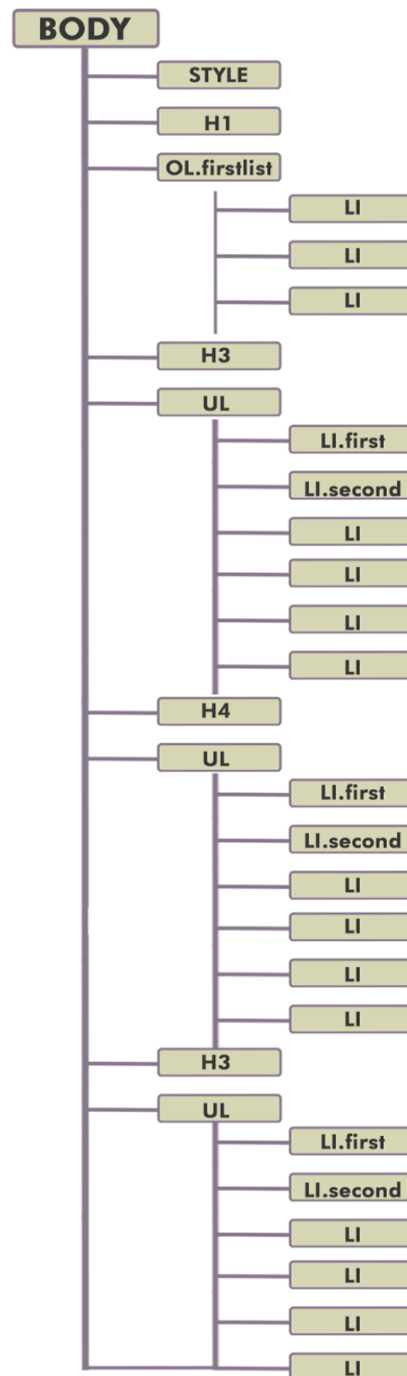
index.css

```css
ol{
    background-color: orange;
}
```

Tree Representation:

BODY
- STYLE
- H1
- OL.firstlist
  - LI
  - LI
  - LI
- H3
- UL
  - LI.first
  - LI.second
  - LI
  - LI
  - LI
  - LI
- H4
- UL
  - LI.first
  - LI.second
  - LI
  - LI
  - LI
  - LI
- H3
- UL
  - LI.first
  - LI.second
  - LI
  - LI
  - LI
  - LI

Output:

Lists Available on the webpage

1. Popular Languages
2. Popular Frameworks
3. Highest Paying Jobs

Popular Languages

- C
- Java
- Python
- C++
- C#
- JavaScript

Popular Framework

- Spring
- Django
- Node.Js
- .NET core
- Flask
- React.JS

Highest Paying Jobs

- Devops Engineer
- Full Stack Developer
- Data Scientist
- AI-ML Engineer
- Cloud Engineer
- Cyber Security Expert

Example-3: Color all the unordered list. (Refer the same index.html in example-1)

Index.css

```css
ul{
    background-color: orange;
}
```

Output:

Lists Available on the webpage

1. Popular Languages
2. Popular Frameworks
3. Highest Paying Jobs

Popular Languages

- C
- Java
- Python
- C++
- C#
- JavaScript

Popular Framework

- Spring
- Django
- Node.Js
- .NET core
- Flask
- React.JS

Highest Paying Jobs

- Devops Engineer
- Full Stack Developer
- Data Scientist
- AI-ML Engineer
- Cloud Engineer
- Cyber Security Expert
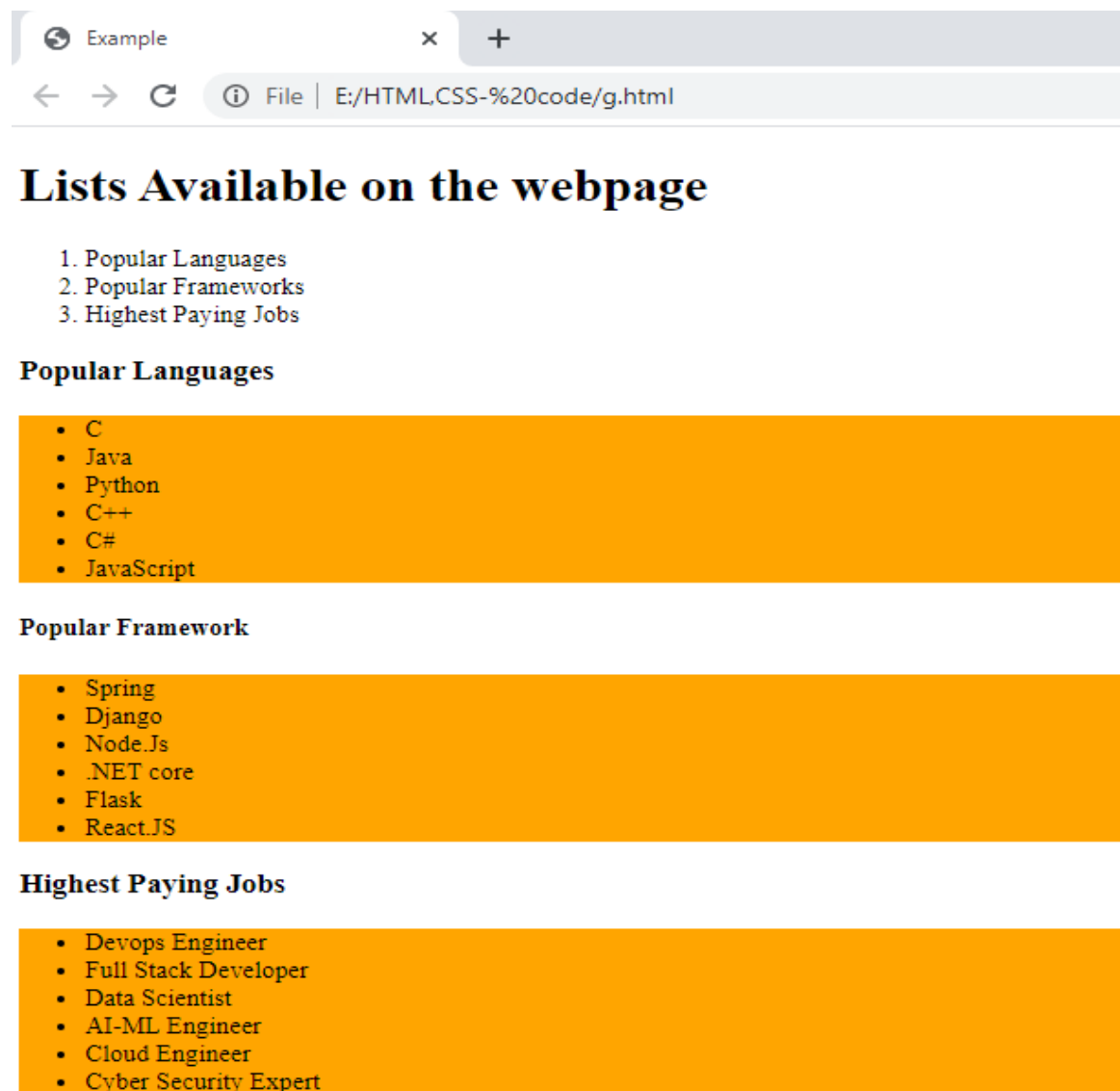
Example-4: color all li except first li inside ul. (Refer the same index.html in example-1)

Index.css

```css
.first ~ li{
    background-color: orange;
}
```

Output:

Lists Available on the webpage

1. Popular Languages
2. Popular Frameworks
3. Highest Paying Jobs

Popular Languages

- C
- Java
- Python
- C++
- C#
- JavaScript

Popular Framework

- Spring
- Django
- Node.Js
- .NET core
- Flask
- React.JS

Highest Paying Jobs

- Devops Engineer
- Full Stack Developer
- Data Scientist
- AI-ML Engineer
- Cloud Engineer
- Cyber Security Expert
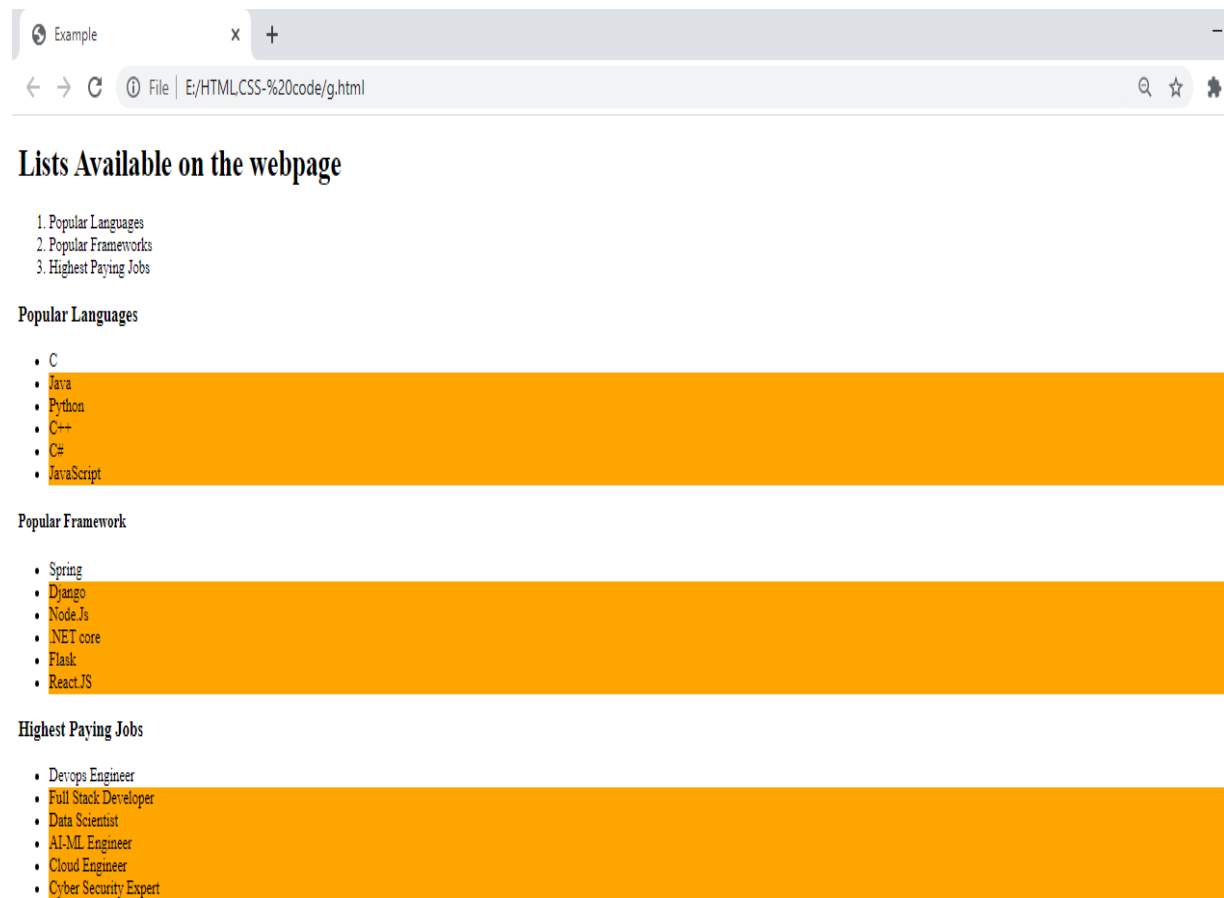
If we observe the li inside ul, the first li and other li have general sibling relationship and first li have class name thus using class selector and general sibling combinator we have styled the li except the first li.

Example-5: Color only the second li in ul

Index.css

```css
.first + li{
    background-color: orange;
}
```
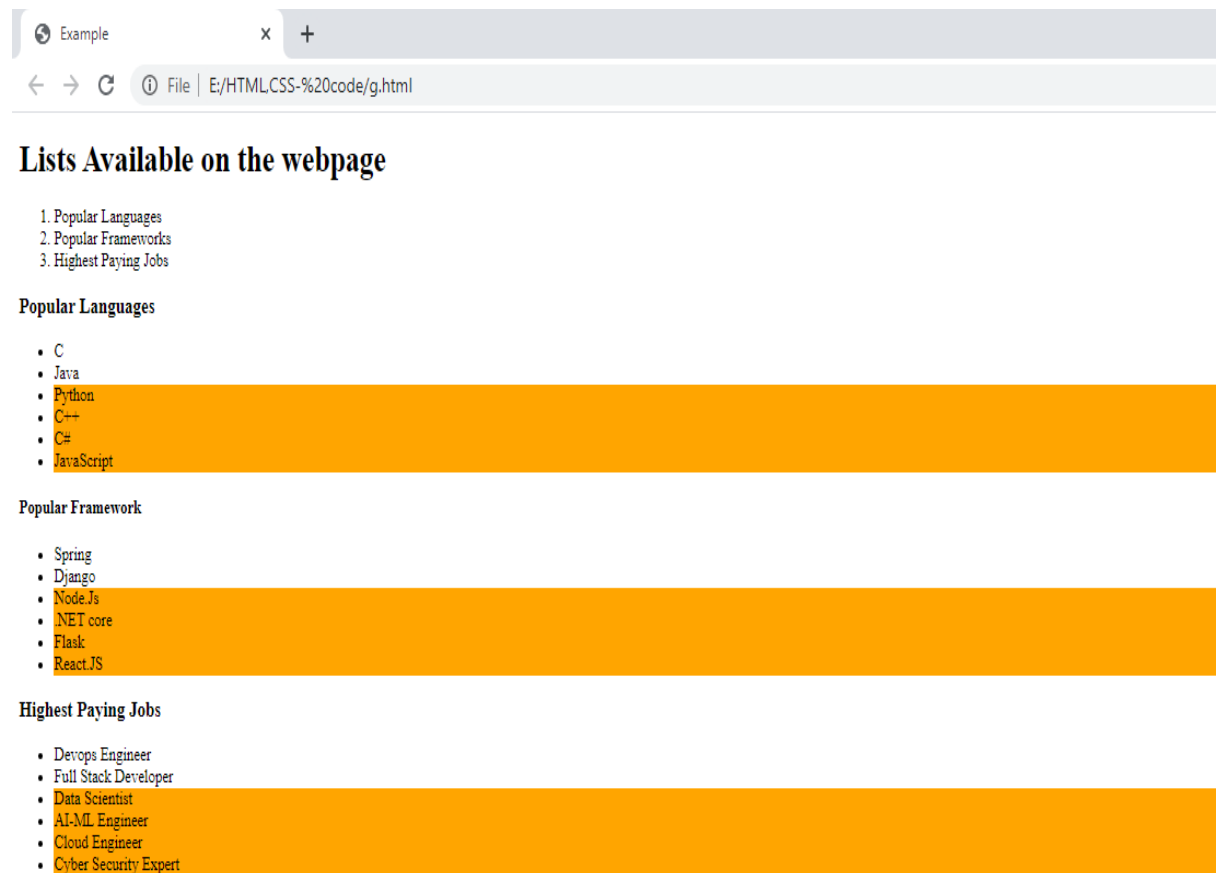
Output:
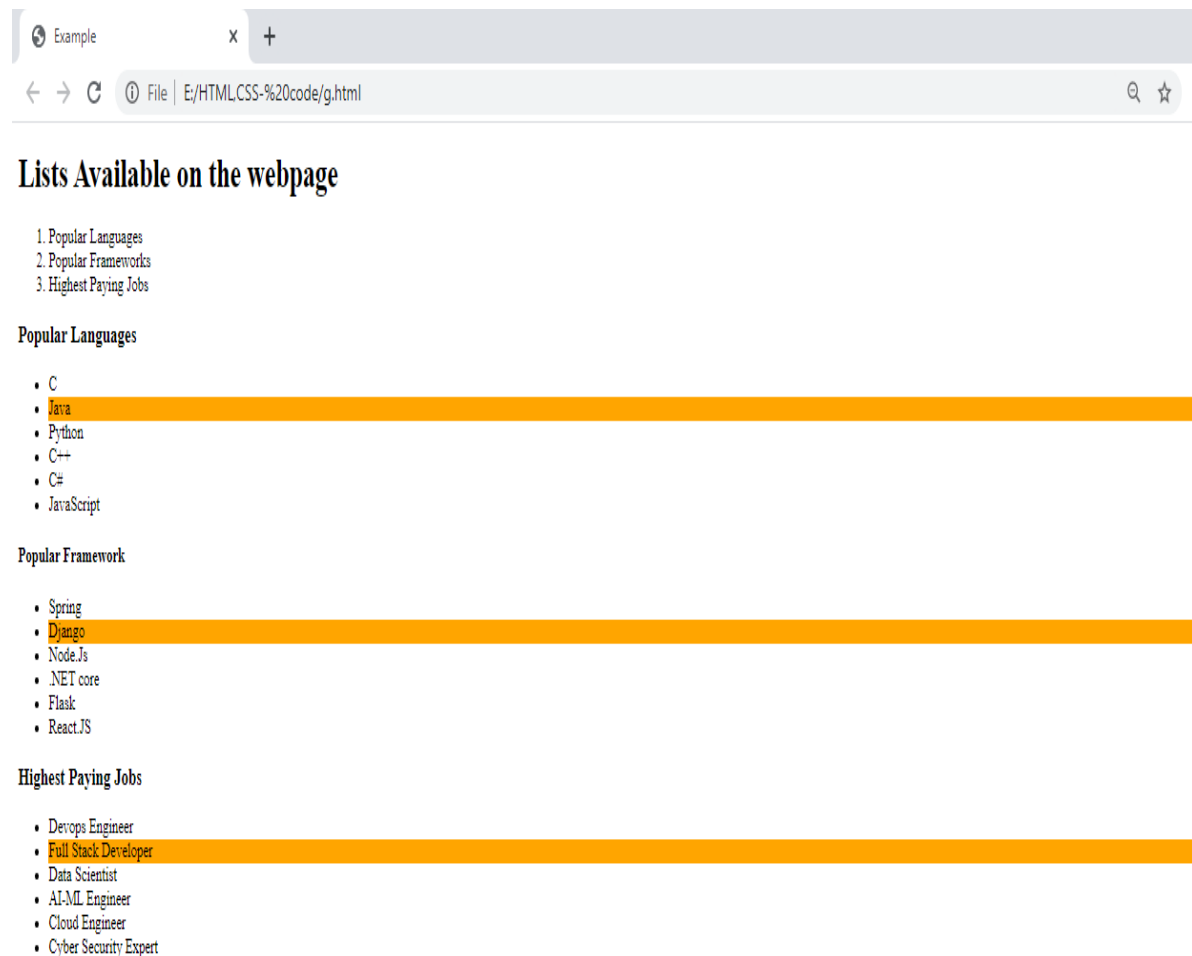
Here we want to style the second li inside ul, if we notice html code clearly there is an adjacent sibling relationship between first li and second li thus styling of second li is done using ==Adjacent sibling selector.==

Example-6: Color all li after second li in ul.

Index.css

```css
.first + li~ li{
    background-color: orange;
}
```
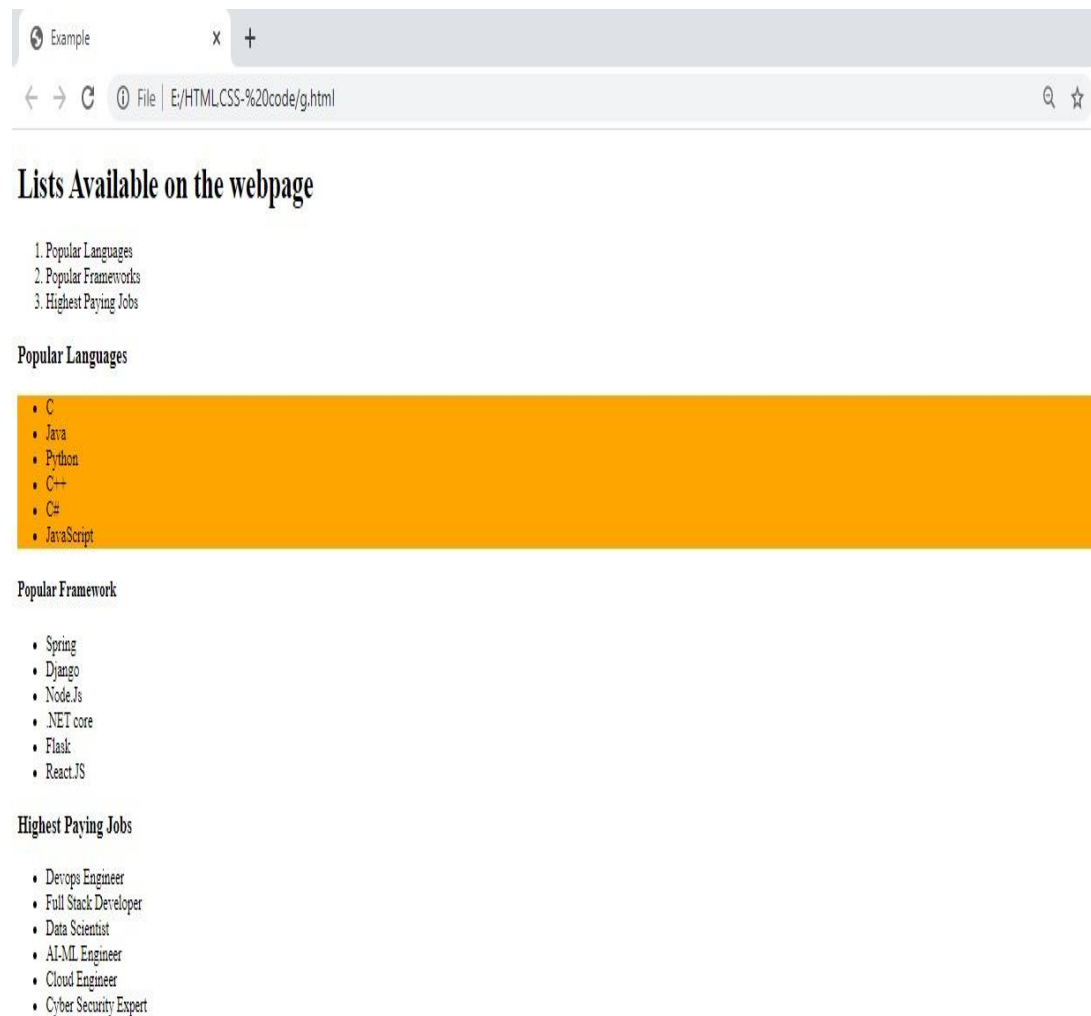
Output:

In the above example, for that we have to select second li first by using **adjacent sibling** relationship. Second li and other li have **general sibling** relationship so styling will be done using general sibling combinator.

Example-7: Color only the first ul

Index.css

```
ol + h3 + ul{
    background-color: orange;
}
```
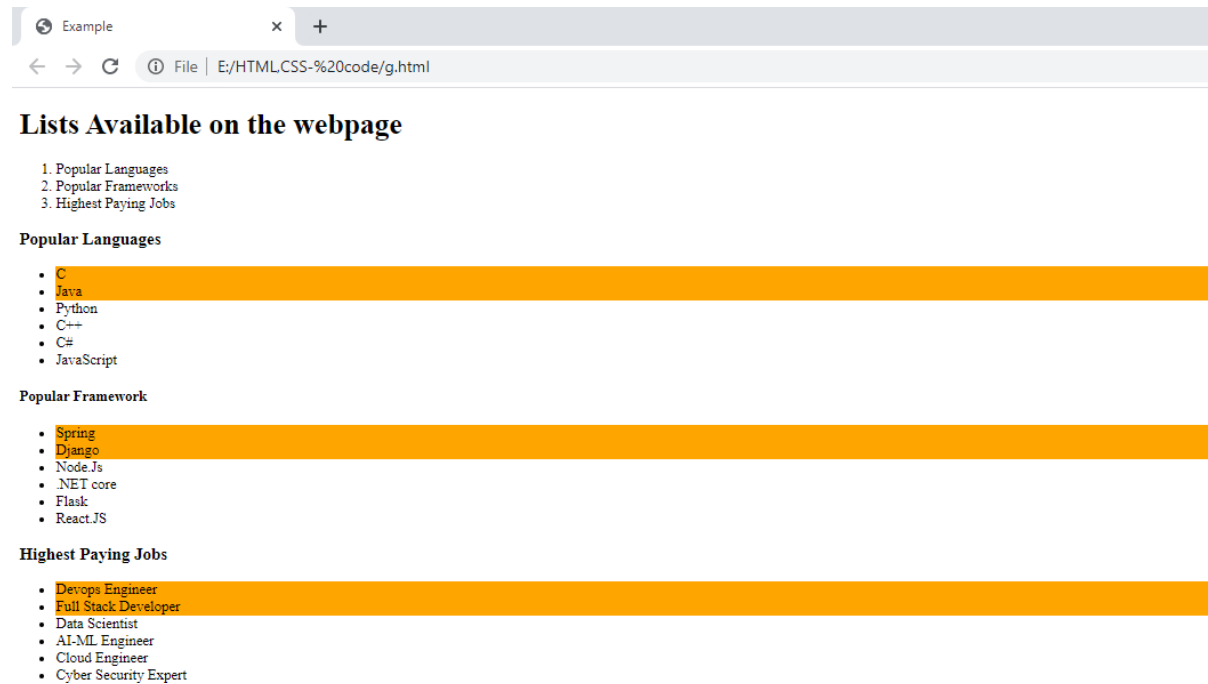
Output:

To style first ul, we have adjacent relationship between ul and h3 has there are multiple h3 and ul available. Here, we have selected first h3 by finding adjacent relationship between ol and h3 then selected first ul using adjacent combinator.

Example-8: Color only first two li in every ul.

Index.css

```css
.first,.first + li{
    background-color: orange;
}
```

Output:

Here, first li is styled using class name. Second li is styled by generating adjacent sibling relationship between first li and second li.