Cascading Style Sheet

By

Sanober Shaikh
Department of IT
TSEC

Introduction:

- Cascading Style Sheets (CSS) is a very simple designed process used to make the web pages a lot more presentable.
- CSS allows you to put styles to customize your web pages.
- It is used to control the layout of more than one web page all at once.
- All the external style sheets are stored in the form of CSS files.

Advantages:

1. Easily maintainable:
2. CSS is time-saving:
3. Superior styles to the native front end:
4. Ease with Search Engines:
5. Efficient cache storing:

Characteristics:

- A style rule consists of a selector component and a declaration block component.
- The selector is used to point to the HTML component which you want to get styled.
- Inside the declaration block, one or more declarations are contained along with semicolons.
- Every declaration which is put, has a CSS property name, a colon, and a value. For example, color is the property, and the value is red in color. Font size is the property, and the 15px is the value.

Characteristics:

- CSS declaration ends with a semicolon, and these blocks are surrounded by curly braces.
- CSS selectors are the ones that are used to find HTML elements that are based on the element name, id, attribute, class, and more.
- One unique element will be selected by the ID of an element.
- If you wish to select the particular element with a specific id, the # function along with the id attribute should be used.
- If you wish to select the elements with a specific class, the period character along with the name class should be written.

Syntax:

- Two main parts
  - Selector
  - Declaration
- Selector defines HTML element to which style is applied.
- Declaration contains css properties as well as value of these properties.
- Selector{ property:value;
  property:value;
- }

```
< style>
body
{
background-color:#d0e4fe;
}
h1
{
color:orange;
text-align:center;
}
p
{
font-family:"Times New Roman";
font-size:20px;
}
</style>
```

Types:

- **Inline**
- An inline style sheet is only used to affect the tag it is in.
- This essentially means that the small details on the page can be changed without changing the overall layout of the page or everything on the page.
- This is advantageous as if you had everything on the external pages, then you would be required to add additional tags to change details.
- Inline overrules external, which means that the small details can be changed.
- It also overrules the internal.

- **Internal**
- The internal would only be used when you wanted to add a small change in the single tag.
- This is because inline only affects the one tag that is contained within it, whereas the internal styling is put on the head of the HTML document.
- This means that if you wish to customize the page, all the required changes would be seen by just scrolling.
- The internal styling is placed inside the tags.
- Comparatively, this looks neater, simple, elegant and organized because of the separate styling and tagging.

- **External**
- External stylesheets are used to allow people to format and recreate their web pages on an entirely different document.
- This effectively means that you can have two or more workplaces, as more than one stylesheet can be embedded inside the document, thereby providing you with a much cleaner workspace.
- The stylesheet would be easily accessible in this case which is a huge advantage, but on the other hand, any changes done in the external sheet would affect all the parent sheets it is linked to.

Cascading Order

All the styles in a page will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

1. Inline style (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default So, an inline style has the highest priority, and will override external and internal styles and browser defaults.

CSS Font

- With CSS, a color is most often specified by:
  - a HEX value - like "#ff0000"
  - an RGB value - like "rgb(255,0,0)"
  - a color name - like "red“
- CSS properties used for background effects:
  - background-color
  - background-image
  - background-repeat
  - background-position

```
body {background:#ffffff url('img_tree.png') no-repeat right top;}
```

CSS font properties

- Font: sets all font properties
  - Font-family
  - Font-size: pixel
  - Font –color
  - Font-style: italic, normal
  - Font-weight:bold
  - Font-variant: small-caps

- color – specifies the color of the text
- font-size – size of font: xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger or numeric value
- font-family – comma separated font names
  - Example: verdana, sans-serif, etc.
  - The browser loads the first one that is available
  - There should always be at least one generic font
- font-weight can be normal, bold, bolder, lighter or a number in range [100 … 900]

- font:italic bold 12px verdana

- font-style: italic;
- font-weight: bold;
- font-size: 12px;
- font-family: verdana;

CSS Text

- text properties are: text-align, text-decoration, text-transform, text-indent, line-height, letter-spacing, word-spacing etc.
- Color:
- Text-align:left, right, center, justified
- Text-decoration: underline, overline, line-through
- Text-transform: uppercase, lowercase, capitalize
- Text-indent:
- Letter-spacing: used to set an extra space between the characters
- Word-spacing:

Background Properties:

- background: #FFF0C0 url("back.gif") no-repeat fixed top;

- background-color: #FFF0C0;
- background-image: url("back.gif");
- background-repeat: no-repeat;
- background-position: top;

CSS Links

- A link has four different states — link, visited, active and hover.
- These four states of a link can be styled differently through using the following anchor pseudo-class selectors.
- a:link — define styles for normal or unvisited links.
- a:visited — define styles for links that the user has already visited.
- a:hover — define styles for a link when the user place the mouse pointer over it.
- a:active — define styles for links when they are being clicked.

- a:link {
- color: #ff0000;
- text-decoration: none;
- border-bottom: 1px solid; }
- a:visited { color: #ff00ff; }
- a:hover { color: #00ff00; border-bottom: none; }
- a:active { color: #00ffff; }

CSS Lists

- Changing the marker type of the list:
- Changing the position of list marker
- list-style-position: inside;
- Using Images as List Markers

- ul { list-style: square inside url("images/bullet.png"); }

CSS Tables

- Adding borders to tables:

```
table, th, td { border: 1px solid black; }
```

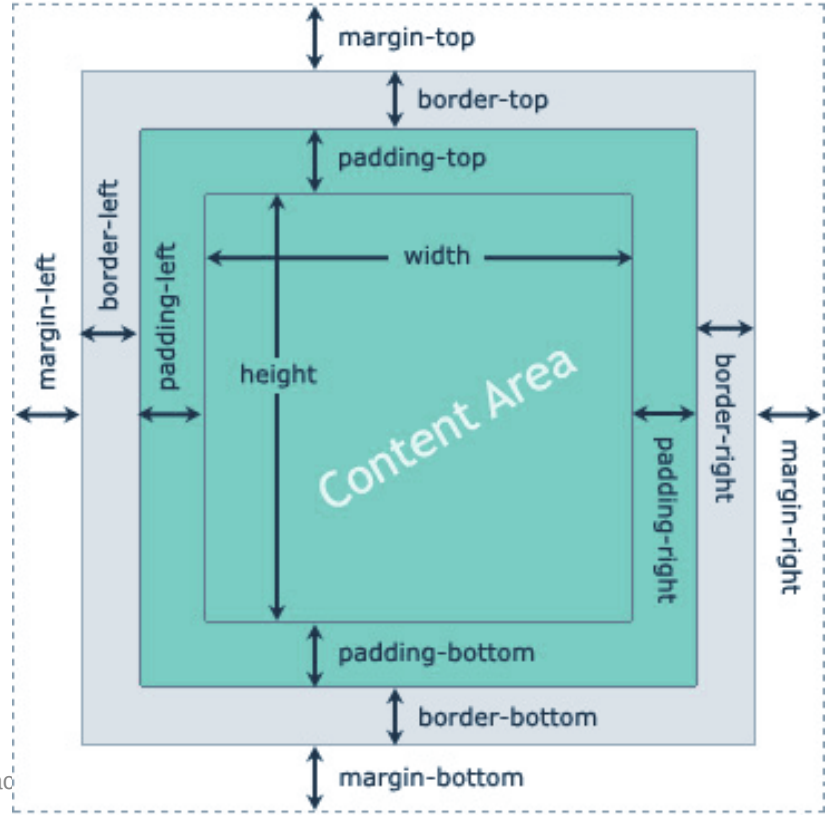
  - Collapsing Table Borders
  - Border-collapse:collapse

  - Adjusting the space inside the table
  - th, td { padding: 15px; }

- Aligning the Text Inside Table Cells
- Horizontal Alignment of Cell Contents
- th { text-align: left; }
- Vertical Alignment of Cell Contents
- th { height: 40px; vertical-align: bottom; }

CSS Box Model:

- Every element that can be displayed on a web page is comprised of one or more rectangular boxes.



1. Universal Selector

- Selects all the elements present in an HTML document. Simply use the universal selector to implement the identical rules to all the elements of an HTML.
- The asterik symbol(*) is used to represent the universal selector.

```
* { }
* {
    margin:0;
    Padding:0;
}
```

- The Type selector (Group Selector)
- Matches all the elements specified in the list with the given value to determine the elements to which the CSS rules are applied.

```
h1, h2, h3, p {font-family: sans-serif;}
```

- The Class selector:
- Allows you to apply CSS rules to the elements that carry a class attribute whose value matches with the class attribute specified in the class selector.

- The ID Selector:
- The value of ID attribute is unique within the document.
- a CSS ID selector should also be used only when you need to target one, specific and unique element on a page.

- The child selector:
- it allows you to target ALL children and grandchildren (and so on) of one or several elements.
- The > symbol is used as combinator.

```
td > b { font-family : sans-serif;}
```

- Selects all <b> elements where the parent is a <td> element.

ol > li { color: red; }

```
<ol>
  <li>abcd.com</li>
  <ul>
    <li>Oracle/PLSQL</li>
    <li>SQL</li>
    <li>Excel</li>
  </ul>
  <li>xyz.com</li>
  <li>pqr.com</li>
</ol>
```

- The Descendent Selector:
- matches an element that is descendent of another element.
- A descendent element is an element that is nested inside the other element.
- In case of descendent selector, white space is used as combinator.

```
Table b { font-family : sans-serif;}
```

- The combinator we use in a descendant selector is a whitespace character: a space, horizontal tab, carriage return, line feed, or form feed.
- Since whitespace characters are allowed around all combinators, you can include more than one whitespace character between the simple selectors in a descendant selector.

ul li { color: yellow; }

```
<ol>
  <li>abcd.com</li>
  <ul>
    <li>Oracle/PLSQL</li>
    <li>SQL</li>
    <li>Excel</li>
  </ul>
  <li>xyz.com</li>
  <li>pqr.com</li>
</ol>
```

- The Sibling Selector:
- With the general sibling CSS selector, which takes a selector, followed by a tilde character (~) and then the selector you wish to target, you can target elements by requiring the presence of another element within the same parent element.
- Another requirement is that the first part of the selector needs to be present in the markup BEFORE the targeted element, even though they are all children of the same parent

```
<style type="text/css">
 h2 ~ p { font-style: italic; }
</style>
<div id="content">
 <h1>Hello, world!</h1>
 <p>Some text here</p>
<h2>Hello, world!</h2>
<p>Some text here</p>
 <p>More text here</p>
</div>
```

- The last two paragraph tags will be in italic, but not the first,
- because it comes before the H2 element in the markup.
- Important: The sibling selector does not affect grandchildren:

- The Adjacent Sibling Selector:
- Selects all the elements that are adjacent siblings of a specified element.
- Sibling element must have same parent element.
- The word adjacent means side by side.
- No other element could exist between the adjacent sibling element.

```
h2 + p { font-family: sans:serif;}
<h2> heading</h2>
<p>para 1 </p>
<p> para 2</p>
```

- The first paragraph matches the adjacent sibling selector , h2+p .
- Because the p element is an adjacent sibling to the h2 element.
- The second paragraph does not match with the selector.
- Although it is a sibling of h2 element, it is not adjacent to the element.

CSS3 pseudo classes

- Used to add special effects to the selectors.
- Always start with colon (:).

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

Dynamic Pseudo class

- provides various types of special effect to the <a> element

```
a:link {
    color: #FF0000;
} /* unvisited link */
a:visited {
    color: #00FF00;
} /* visited link */
a:hover {
    color: #FF00FF;
} /* mouse over link */
a:active {
    color: #0000FF;
} /* selected link */
a:focus{
      color:FFFFFF;
}
```

- Language:
- The :lang() selector is used to select elements with a lang attribute with the specified value.
- With CSS you can select elements on the basis of their human language encoding.
- If a paragraph is in a particular language, this is encoded as <p lang="language code"> where language code is usually a two character standard code for common languages.
- For example, French has the language code "fr", while Chinese the code "zh".
- The above examples would be <p lang="fr"> and <p lang="zh">, for French and Chinese respectively.

UI Element States Pseudo Class

- Enables you to specify the appearance of UI elements, such as button and check box.
- Using these classes you can add a particular style to an enabled or disabled button or checked or unchecked boxes.
- :enabled :
- :disabled :
- :Checked:

The :enabled pseudo-class represents user interface elements that are in an enabled state.
:enabled{ properties }

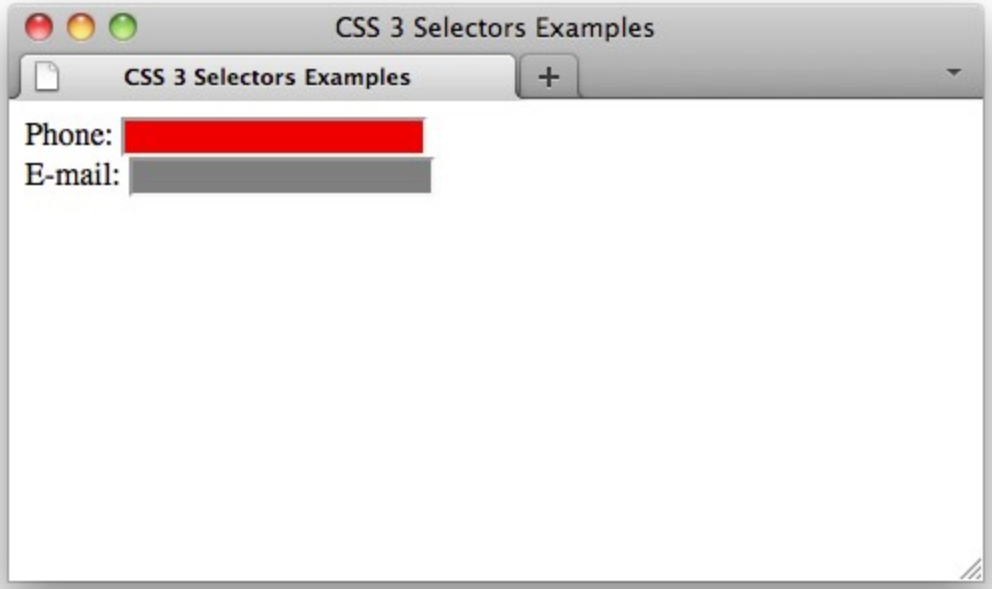
- Mostly used on form elements.

[style.css]

```
 input:enabled{
   background-color: red;
 }
 input:disabled{
   background-color: gray;
 }
```

[index.html]

```
<body>
<form>
  Phone: <input type="tel"><br>
  E-mail: <input type="email" disabled="disabled">
</form>
</body>
```


CSS 3 Selectors Examples
CSS 3 Selectors Examples
Phone:
E-mail:

The :checked pseudo-class represents input elements(only for checkboxes or radio buttons) that are in an enabled state.

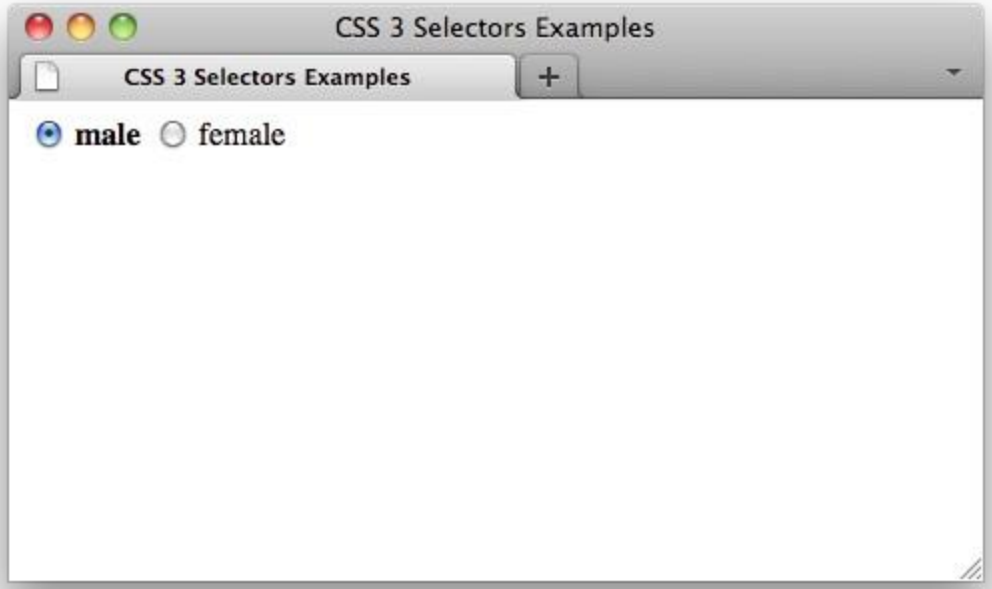
- :checked{ properties }

- [style.css]

```
 input:checked + label{
    font-weight: bold;
  }
```

- [index.html]

```
<body>
<form>
  <input type="radio" id="m" name="gender" value="male">
  <label for="m">male</label>
  <input type="radio" id="f" name="gender" value="female">
  <label for="f">female</label>
</form>
</body>
```


CSS 3 Selectors Examples
CSS 3 Selectors Examples
male
female

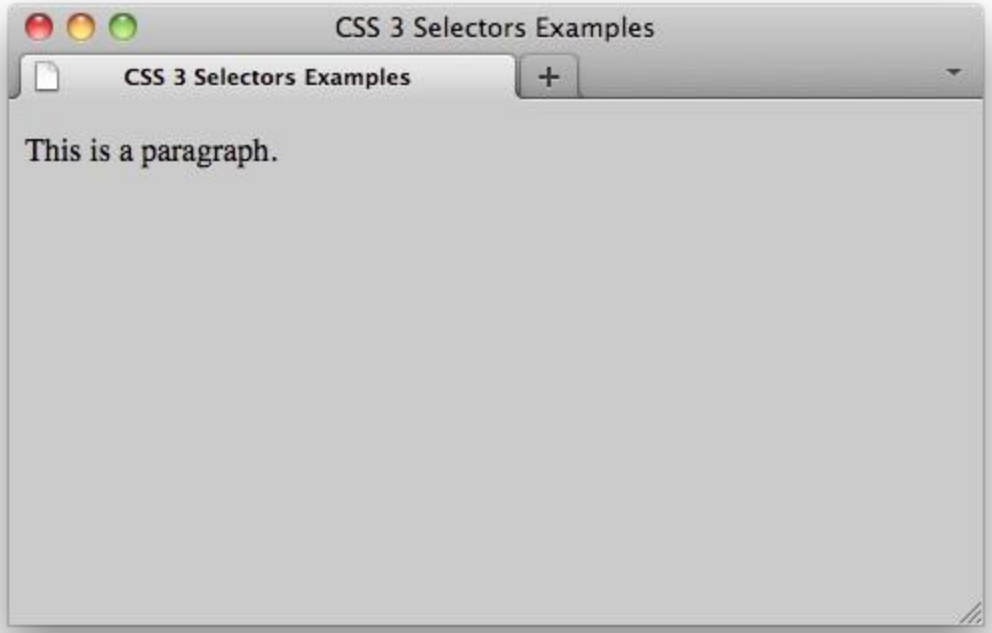
Structural Pseudo Class

- Allows the selection of elements on the basis of structure of the entire HTML document.
- Which includes the position of each element and number of times the occurrence of an element in the document.
- In HTML, this is always the HTML element.
- :root :Selects the document's root element

```
:root {
    css declarations;
}
```

- [style.css]
- :root{
- background-color: #ccc;
- }

- [index.html]
- <body>
- This is a paragraph.
- </body>



- nth-child() Selector:
- nth-child(*n*) selector matches every element that is the *n*th child, regardless of type, of its parent.
- *n* can be a number, a keyword, or a formula.

```
:nth-child(number) {
    css declarations;
}
```

Syntax

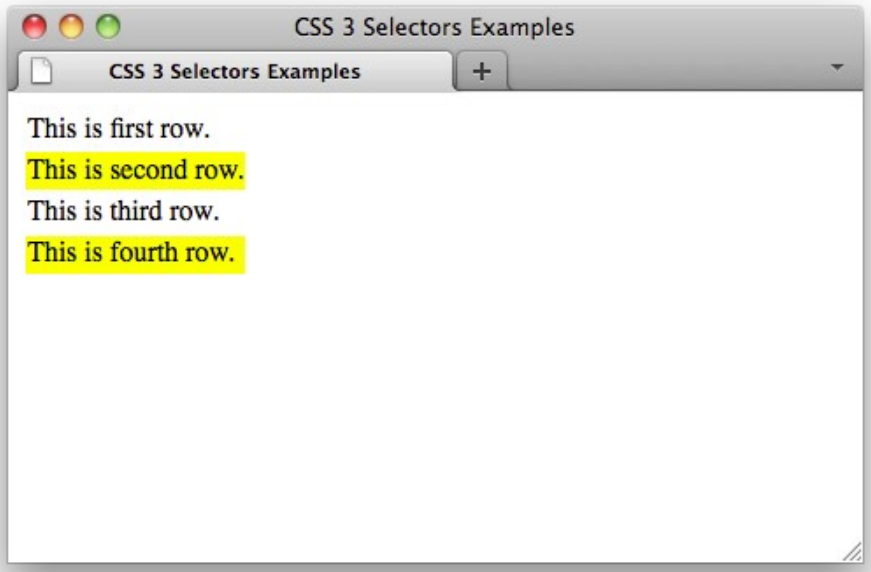
- `selector:nth-child(an+b){ properties }`

The examples of an+b are as follows:

```
:nth-child(2n) /* represents every even element */
:nth-child(even) /* same, represents every even element */
:nth-child(2n+1) /* represents every odd element */
:nth-child(odd) /* same, represents every odd element */
:nth-child(10n-1) /* represents the 9th, 19th, 29th, etc, element */
```

- [style.css]

```
tr:nth-child(2n){
  background-color: yellow;
}
```



[index.html]

```
<body>
  <table>
    <tr><td>This is 1st row.
        </td></tr>
    <tr><td>This is second
row.</td></tr>
    <tr><td>This is third
row.</td></tr>
    <tr><td>This is fourth
row.</td></tr>
  </table>
</body>
```

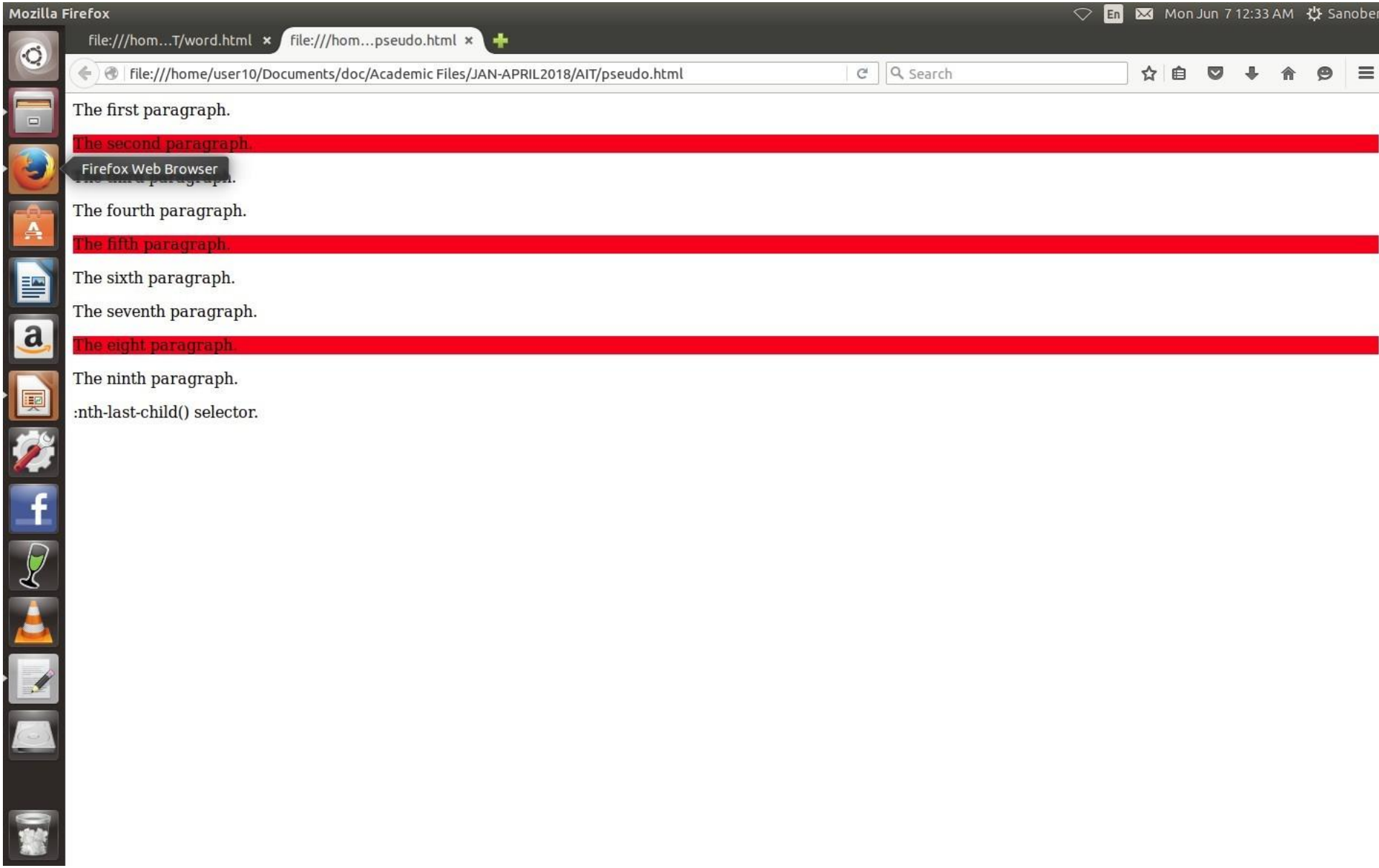
- Specify a background color for every element that is the second child of its parent:
- <style>
- p:nth-child(2) {
- background: #ff0000;
- }
- </style>
- </head>
- <body>
- <h1>This is a heading</h1>
- The first paragraph.
- The second paragraph.
- The third paragraph.
- The fourth paragraph.

```
p:nth-child(odd) {
    background: #ff0000;
}
p:nth-child(even) {
    background: #0000ff;
}
</style>
</head>
<body>
<h1>This is a heading</h1>
<p>The first paragraph.</p>
<p>The second paragraph.</p>
<p>The third paragraph.</p>
<p>:nth-child() selector.</p>
</body>
```

:nth-last-child() Selector

- The :nth-last-child(*n*) selector matches every element that is the *n*th child, regardless of type, of its parent, counting from the last child.
- *n* can be a number, a keyword, or a formula.

```
<style>
p:nth-last-child(3n+0) {
    Background-color: #ff0000;
}
</style></head>
<body>
<p>The first paragraph.</p>
<p>The second paragraph.</p>
<p>The third paragraph.</p>
<p>The fourth paragraph.</p>
<p>The fifth paragraph.</p>
<p>The sixth paragraph.</p>
<p>The seventh paragraph.</p>
<p>The eight paragraph.</p>
<p>The ninth paragraph.</p>
<p>:nth-last-child() selector.</p>
</body>
```


Mozilla Firefox
En
Mon Jun 7 12:33 AM
file:///hom...T/word.html
file:///hom...pseudo.html
file:///home/user10/Documents/doc/Academic Files/JAN-APRIL2018/AIT/pseudo.html
Search
Firefox Web Browser
The first paragraph.
The second paragraph.
The fourth paragraph.
The fifth paragraph.
The sixth paragraph.
The seventh paragraph.
The eight paragraph.
The ninth paragraph.
:nth-last-child() selector.

:nth-of-type(N)

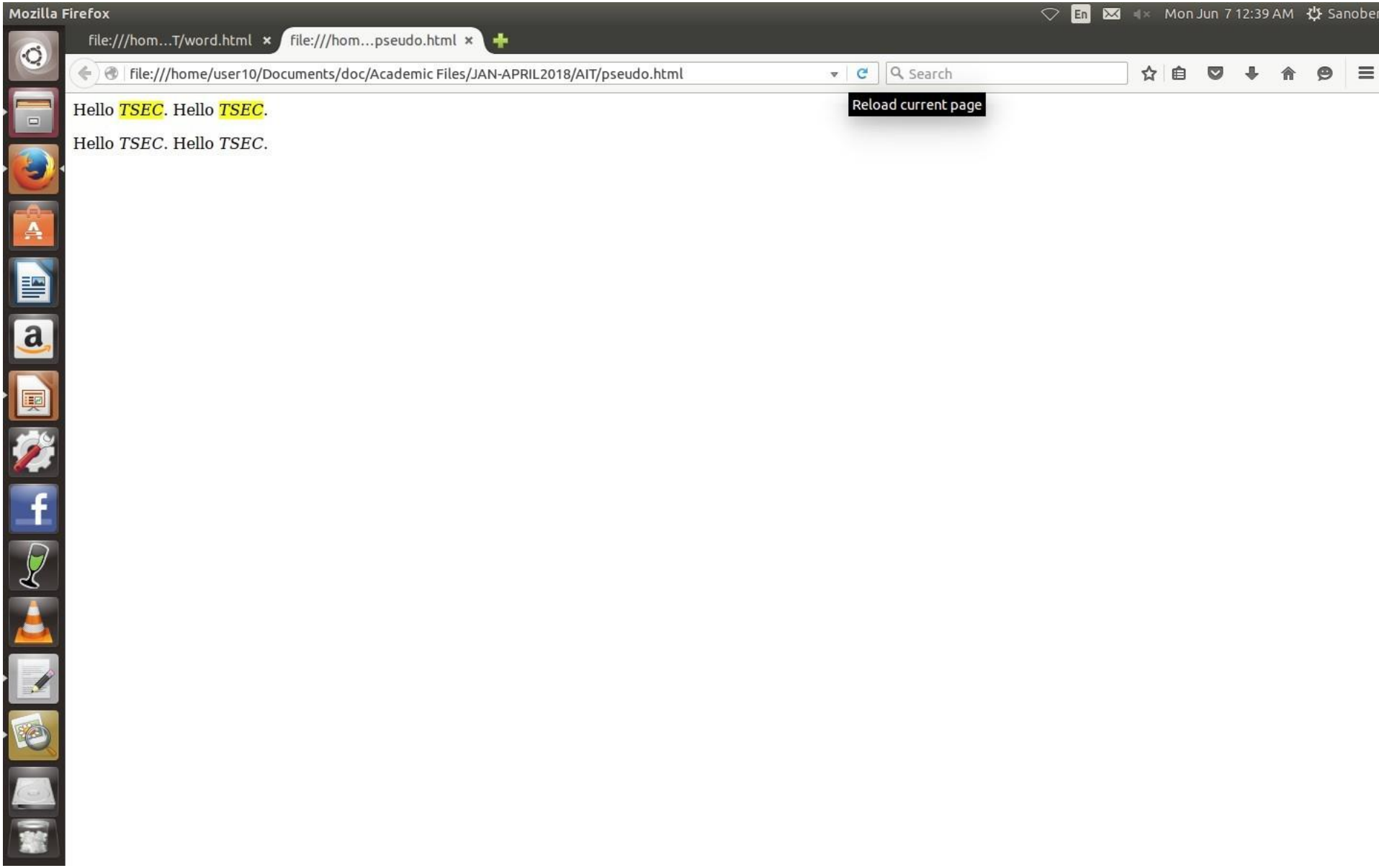
- The :nth-of-type(*n*) selector matches every element that is the *n*th child, of a particular type, of its parent.
- the :nth-child() selector to select the element that is the *n*th child, **regardless of type**, of its parent.

```
<style>
p:nth-of-type(2) {
    background: #ff0000;
}
</style></head>
<body>
<h1>This is a heading</h1>
<p>The first paragraph.</p>
<p>The second paragraph.</p>
<p>The third paragraph.</p>
</body>
```

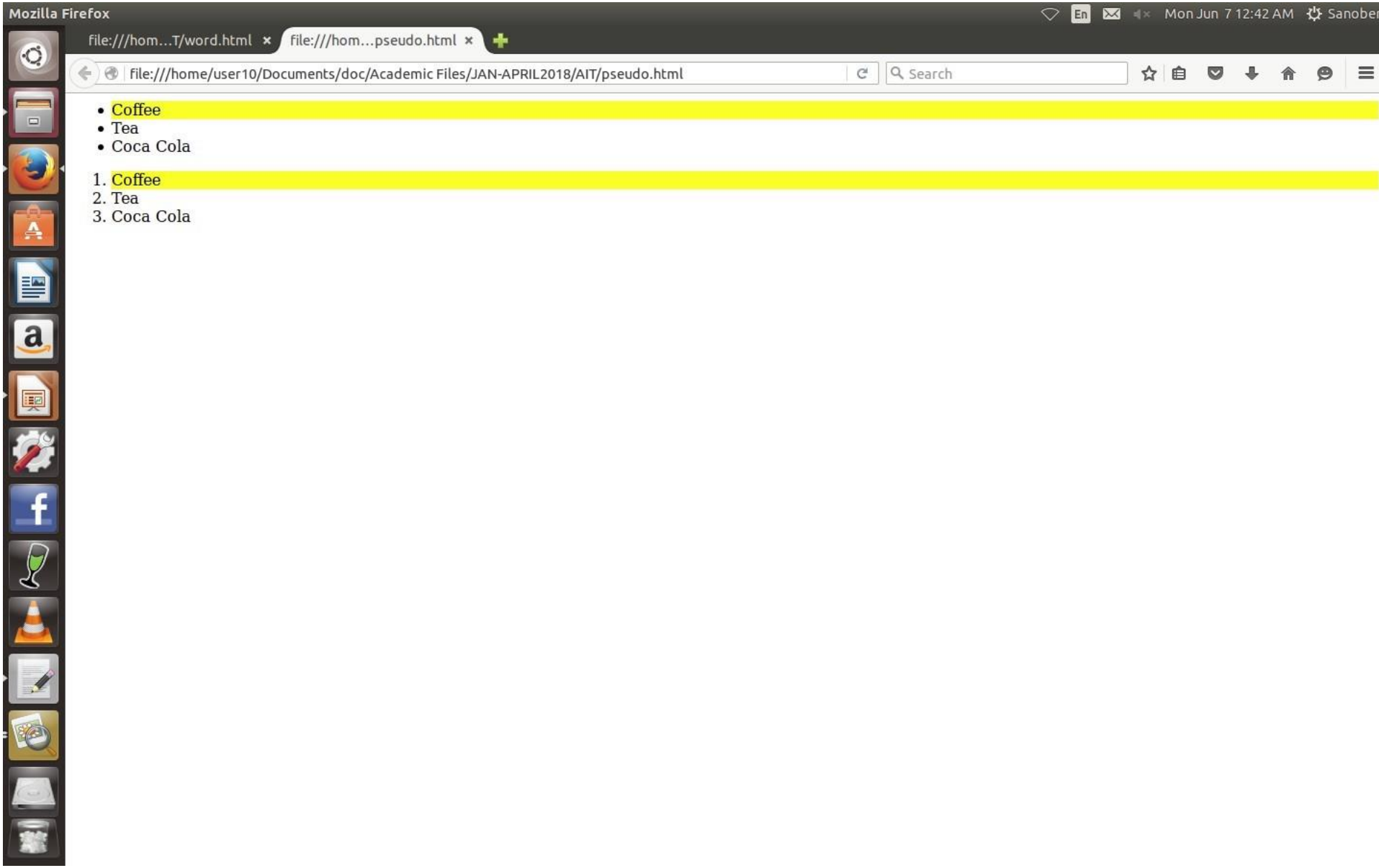
:first-child Selector

- The :first-child selector is used to select the specified selector, only if it is the first child of its parent.
- Same as :nth-child(1).

```
<style>
p:first-child i {
    background: yellow;
}
</style></head>
<body>
<p>Hello <i>TSEC</i>. Hello <i>TSEC</i>.</p>
<p>Hello <i>TSEC</i>. Hello <i>TSEC</i>.</p>
</body>
```


Mozilla Firefox
Mon Jun 7 12:39 AM
file:///hom...T/word.html
file:///hom...pseudo.html
file:///home/user10/Documents/doc/Academic Files/JAN-APRIL2018/AIT/pseudo.html
Search
Reload current page
Hello TSEC. Hello TSEC.
Hello TSEC. Hello TSEC.

```
<style>
li:first-child {
    background: yellow;
}
</style></head>
<body>
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ol>
```


Mozilla Firefox
Mon Jun 7 12:42 AM
file:///hom...T/word.html
file:///hom...pseudo.html
file:///home/user10/Documents/doc/Academic Files/JAN-APRIL2018/AIT/pseudo.html
Search
Coffee
Tea
Coca Cola
1. Coffee
2. Tea
3. Coca Cola

:first-of-type Selector

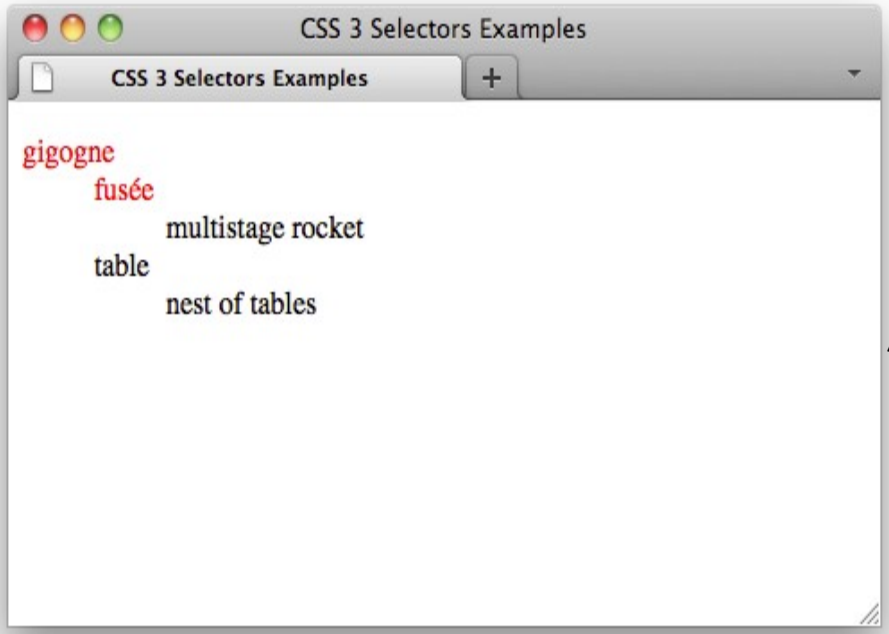
- The :first-of-type selector matches every element that is the first child, of a particular type, of its parent.
- This is the same as :nth-of-type(1).

[style.css]

```
dl dt:first-of-type{
  color: red;
}
```

[index.html]

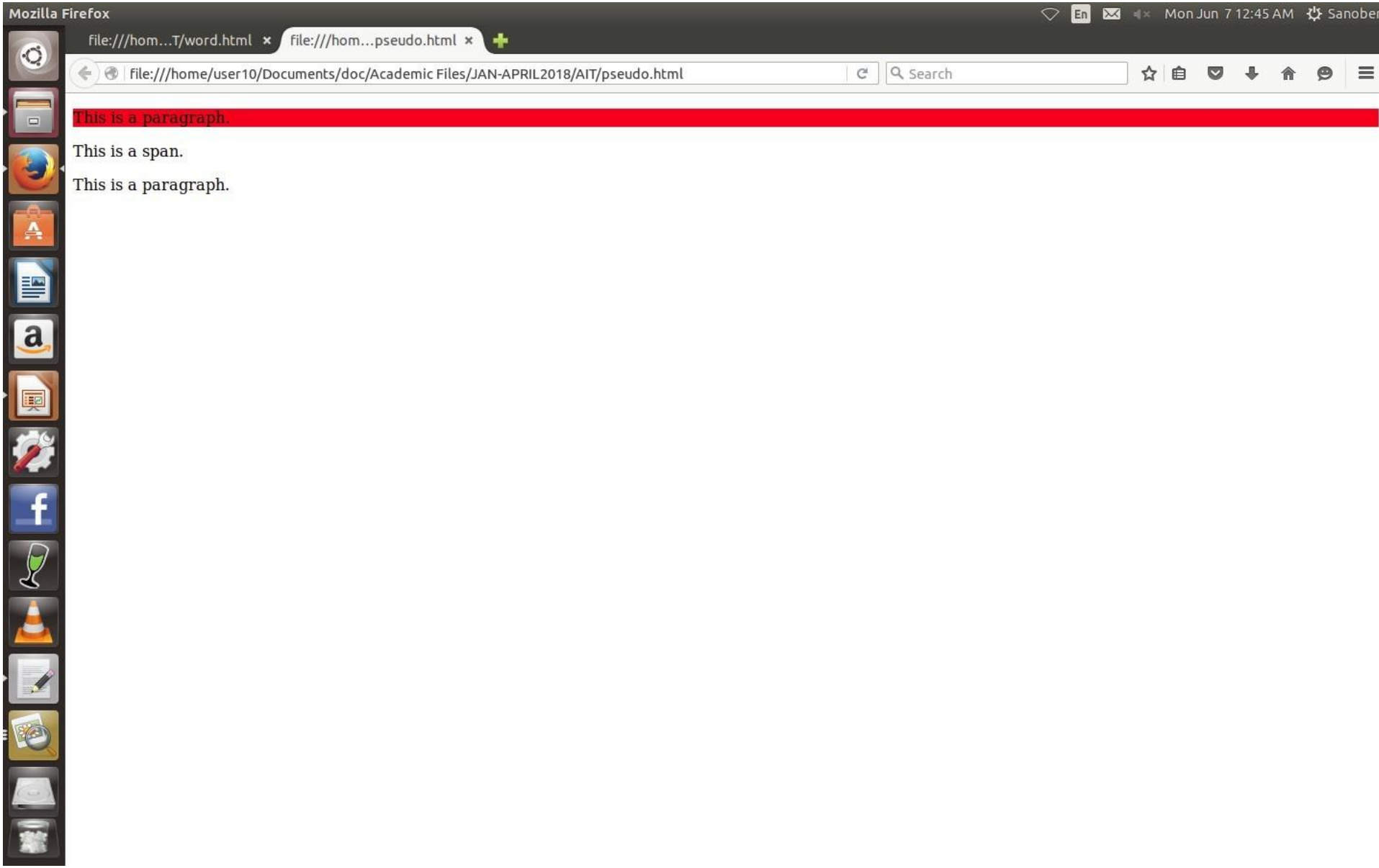
```
<body>
  <dl>
    <dt>gigogne</dt>
    <dd>
      <dl>
        <dt>fusée</dt>
        <dd>multistage rocket</dd>
        <dt>table</dt>
        <dd>nest of tables</dd>
      </dl>
    </dd>
  </dl>
/body>
```



:only-child

- The :only-child selector matches every element that is the only child of its parent.
- It targets an element whose parent element has no other element children.

```
<style>
p:only-child {
    background: #ff0000;
}
</style></head><body>
<div><p>This is a paragraph.</p></div>
<div><span>This is a span.</span><p>This is a paragraph. </p>
</div>
</body>
```


Mozilla Firefox
Mon Jun 7 12:45 AM
file:///hom...T/word.html
file:///hom...pseudo.html
file:///home/user10/Documents/doc/Academic Files/JAN-APRIL2018/AIT/pseudo.html
Search
This is a paragraph.
This is a span.
This is a paragraph.

:only-of-type

- The :only-of-type selector matches every element that is the only child of its type, of its parent.

:empty

- Selects an elements that have no child elements.

Negation pseudoclasses

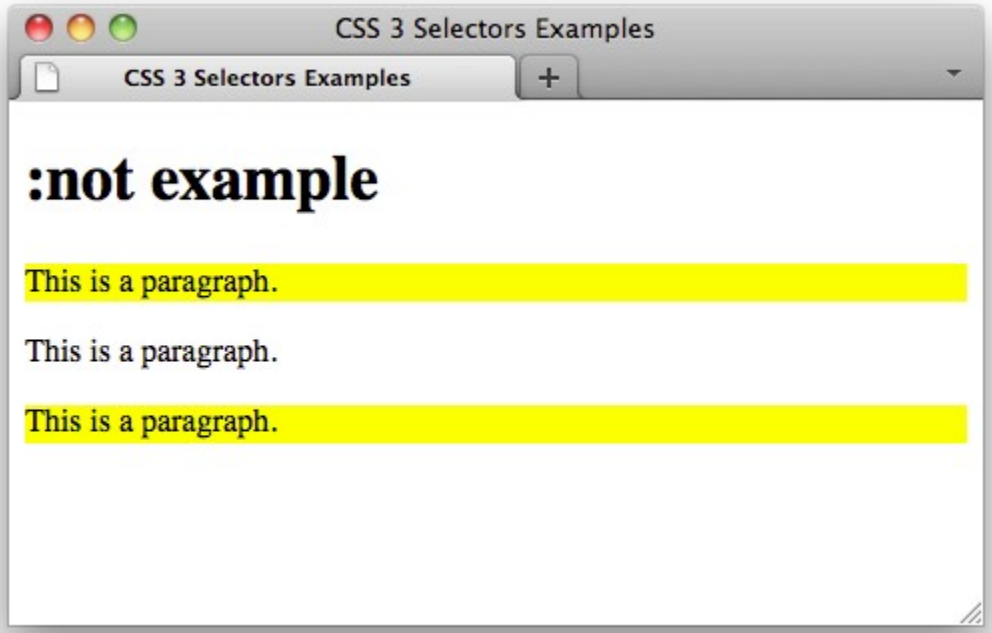
- The :not pseudo-class represents an element that is not represented by its argument.

- [style.css]

```
 p:not(#example){
   background-color: yellow;
 }
```

- [index.html]

```
<body>
  <h1>:not example</h1>
  <p>This is a paragraph.</p>
  <p id="example">This is a paragraph.</p>
  <p>This is a paragraph.</p>
</body>
```


CSS 3 Selectors Examples
CSS 3 Selectors Examples
:not example
This is a paragraph.
This is a paragraph.
This is a paragraph.

Exploring the Pseudo elements

- Allows you to apply styling rules to the subparts of elements content.
- Always starts with ::.
- Selector:: pseudo-element { property:value;}

- ::first-line : applies special styles to the first line in a block level element,such as p and div.
- ::first-letter : applies special effect to the first letter in a block level element, list item, table, caption, table cell or an inline block.
- ::before : allows you to insert content before the content of an element.
- ::after : allows you to insert content after the content of an element.

- P::first-letter{ font-size:20px;}
- P::before{content:”Note: “;}
- P::first-line{color:green;}