Scripting for Multimedia

LECTURE 6: UNDERSTANDING SELECTORS, SPECIFICITY, AND CASCADING

Defining selectors

- An element type selector is based on the name of the tag
- Creating an element type selector
  - Examples

```
/* <body> element */
body {
    background-color: white;
    color: gray;
}
/* <button> element */
button {
    background-color: white;
    color: gray;
}
```

Defining selectors

- An id selector is based on the id of the element (prefixed with #)
- Creating an id selector
  - Example

```
#btnSave {
    background-color: white;
    color: gray;
}
```

Defining selectors

- A class selector is a style with a class name, prefixed with the period (.) symbol
- It's also called a named style
- Creating a class selector
  - Example

```
.myStyle {
    background-color: white;
    color: gray;
}
```

```
<input id='txtName' name= 'txtName' type='text' class= 'myStyle' >
<button id='btnOk' class='myStyle'>Ok</button>
```

Defining selectors

- Use asterisk (*) symbol to apply a style to every element
  - Example

```
* {
    background-color: white;
    color: gray;
}
```

  - You should avoid using the universal selector because of the performance cost

Defining selectors

- Using descendant selectors
  - Change the style of elements which are descendants of another element by specifying a selector chain
  - Example

```
li a {
    text-decoration: none;
}
```

  - You can specify a selector chain with many descendant levels

```
div ol li a {
    text-decoration: none;
}
```

Defining selectors

- Using child selectors
  - Change the style of elements only if the elements are direct children of another element
  - a parent element + > + child element

```
li > a {
    text-decoration: none;
}
```

  - multiple child levels:

```
div > ol > li > a {
    text-decoration: none;
}
```

Defining selectors

- Using pseudo-class and pseudo-element selectors
  - Sometimes you may want to apply a style to something more regular than an element
  - Pseudo-class classify elements based on something other than name, attributes, or content and something that cannot be deduced from the DOM tree
  - Pseudo classes can be placed anywhere in the selector chain

Defining selectors

- List of pseudo classes (find more pseudo classes from the related materials)
  - :link `a:link`
  - :visited `a:visited`
  - :active `a:hover:active`
  - :hover `a:hover`
  - :focus `input:focus`
  - :checked `input[type='checkbox']:checked`
  - :language `p:lang(en)`
  - …

Defining selectors

- Using pseudo-class and pseudo-element selectors
  - Pseudo elements are abstractions of the document tree that provide access to information that is not directly available in the DOM tree

Defining selectors

- List of pseudo elements
  - ::first-line
    ```
    p::first-line
    ```
  - ::first-letter
    ```
    p::first-letter
    ```
  - ::before
    ```
    p::before { content: "Note:"; color: red; }
    ```
  - ::after
    ```
    p::after { content: "Done!"; color: red; }
    ```

Defining selectors

- Selectors can be grouped and applied the same style
  - Example

```
button {
    background-color: white;
    color: gray;
}
p {
    background-color: white;
    color: gray;
}
```

```
button, p {
    background-color: white;
    color: gray;
}
```

Defining selectors

- You can use the pseudo classes anywhere in your selector chain
- You can group pseudo classes
  - a:hover and a:active can be combined as a:hover:active
- You **cannot** use pseudo elements in inline style
- You **cannot** use pseudo elements in the selector chain

Defining selectors

- An adjacent selector is used select an element if it is preceded by a specific element
- The plus (+) sign denotes and adjacent selector
  - Example

```
div + h1 {
    background-color: yellow;
}
```

Defining selectors

- The subsequent sibling selector is similar to the adjacent sibling selector
  - Its search for sibling match doesn't stop at the first match
- The tilde (~) character denotes the sibling selector
  - Example

```
div ~ h1 {
    background-color: yellow;
}
```

Defining selectors

- Examples for using the subsequent adjacent sibling selector and the subsequent sibling selector

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <link href="default.css" rel="stylesheet" />
</head>
```

Defining selectors

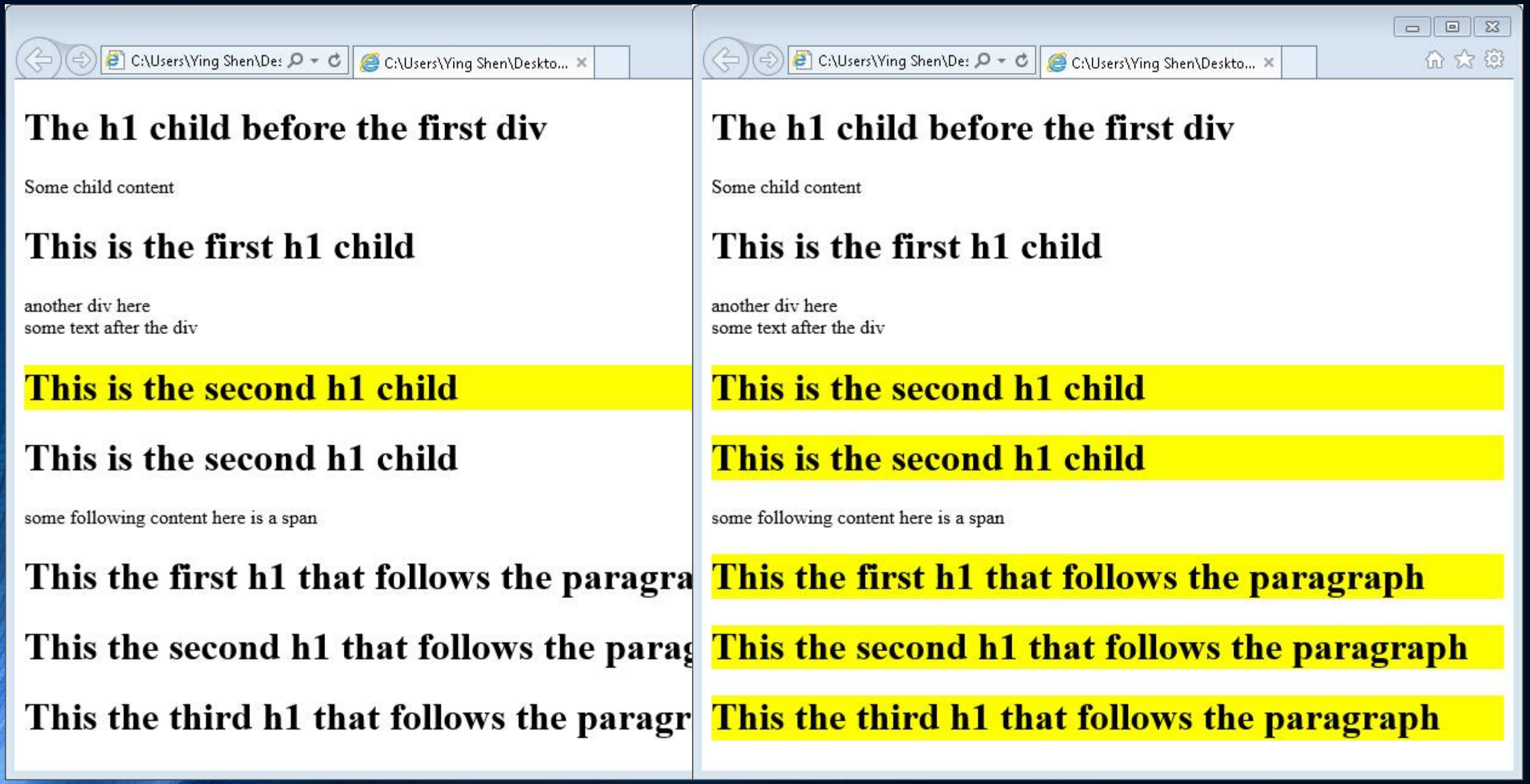
- Examples for using the subsequent adjacent sibling selector and the subsequent sibling selector

```
<body>
    <h1>The h1 child before the first div</h1>
    <div>
        Some child content
        <h1> This is the first h1 child </h1>
        <div> another div here </div>
        some text after the div
        <h1> This is the second h1 child</h1>
        <h1> This is the second h1 child</h1>
    </div>
```

Defining selectors

- Examples for using the subsequent adjacent sibling selector and the subsequent sibling selector

```
    some following content
    <span>here is a span</span>
    <h1>This the first h1 that follows the
paragraph</h1>
    <h1>This the second h1 that follows the
paragraph</h1>
    <h1>This the third h1 that follows the
paragraph</h1>
</body>
</html>
```


C:\Users\Ying Shen\Des
C:\Users\Ying Shen\Deskto...
The h1 child before the first div
Some child content
This is the first h1 child
another div here
some text after the div
This is the second h1 child
This is the second h1 child
some following content here is a span
This the first h1 that follows the paragra
This the second h1 that follows the parag
This the third h1 that follows the paragr
C:\Users\Ying Shen\Des
C:\Users\Ying Shen\Deskto...
The h1 child before the first div
Some child content
This is the first h1 child
another div here
some text after the div
This is the second h1 child
This is the second h1 child
some following content here is a span
This the first h1 that follows the paragraph
This the second h1 that follows the paragraph
This the third h1 that follows the paragraph

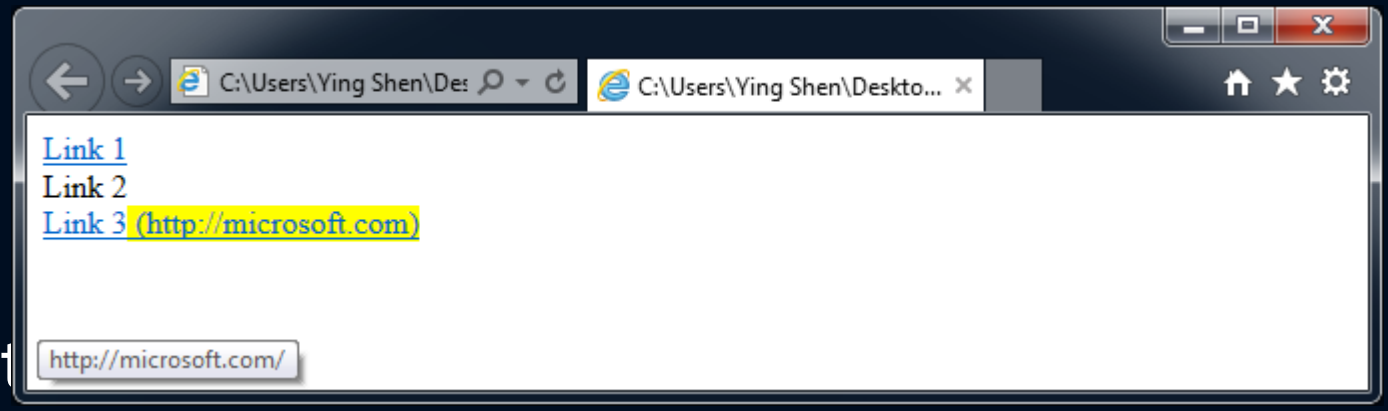
Defining selectors

- An attribute selector selects elements based on the existence of the specified attribute
  - Example

```
a[href]:hover:after {
    content: " (" attr(href) ")";
     background-color: yellow;
}
```

Defining selectors

- Examples for using the att




```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <link href="default.css" rel="stylesheet" />
</head>
<body>
    <a href='http://contoso.com'>Link 1</a><br />
    <a>Link 2</a><br />
    <a href='http://microsoft.com'>Link 3</a><br />
</body>
</html>
```

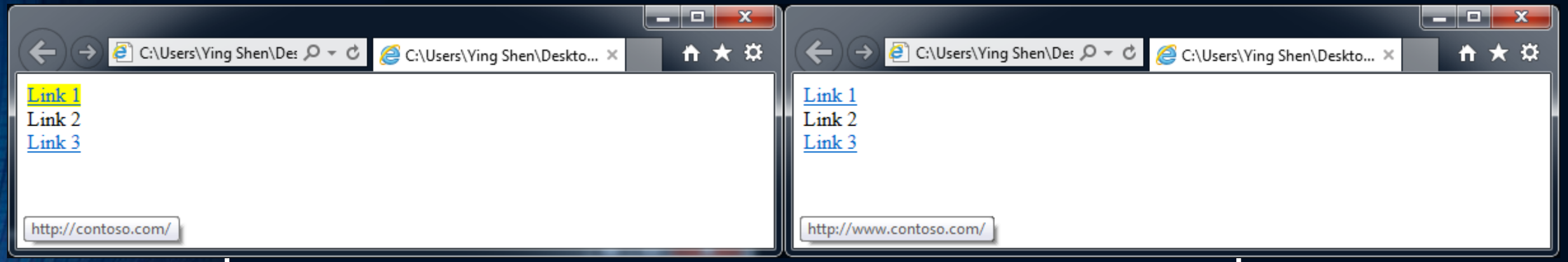
Defining selectors

- An attribute value selector selects all elements where the specified attribute has the specified value
  - Example

```
a[href='http://contoso.com']:hover {
    background-color: yellow;
}
```

Defining selectors

- Examples for using the attribute value selector




```
</head>
<body>
    <a href='http://contoso.com'>Link 1</a><br />
    <a>Link 2</a><br />
    <a href='http://www.contoso.com'>Link 3</a><br />
</body>
</html>
```

Defining selectors

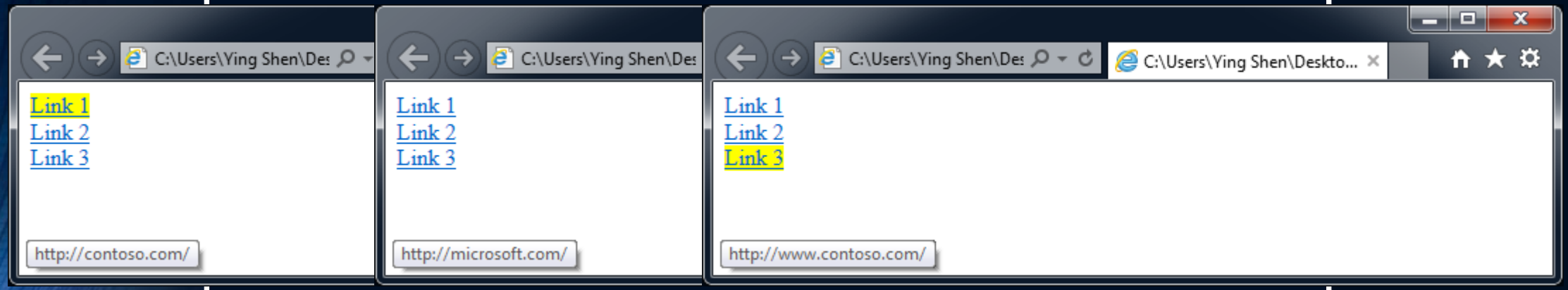
- The attribute contains value selector selects all elements that contain the specified attribute value within the specified attribute
  - Example

```
a[href*='contoso.com']:hover {
    background-color: yellow;
}
```

Defining selectors

- Examples for using the attribute contains value selector

```
<!DOCTYPE html>
```

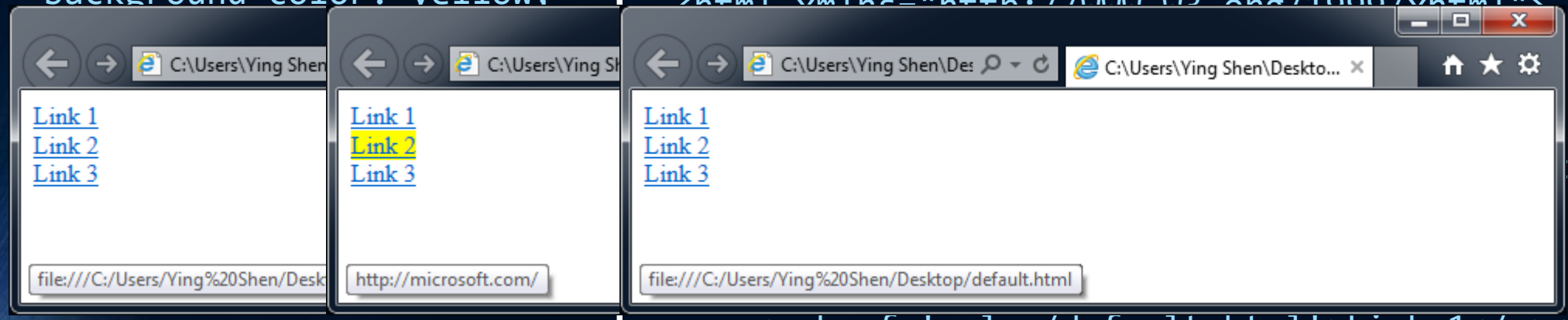



```
    <a href='http://contoso.com'>Link 1</a><br />
    <a href='http://microsoft.com'>Link 2</a><br />
    <a href='http://www.contoso.com'>Link 3</a><br />
</body>
</html>
```

Defining selectors

- Using the attribute value starts with selector

```
a[href^='http']:hover {
    background-color: yellow;
}
```

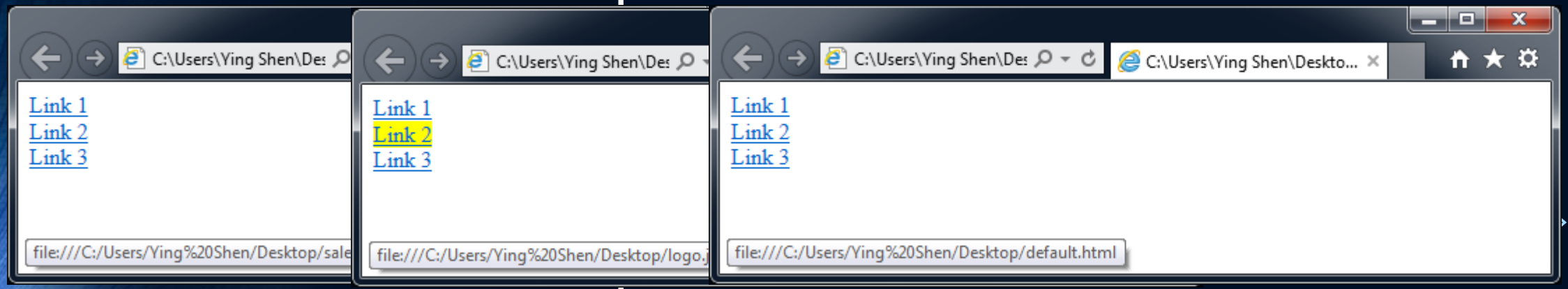



```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
                                                    " />
    <a href='sales/default.html'>Link 1</a><br />
    <a href='http://microsoft.com'>Link 2</a><br />
    <a href='default.html'>Link 3</a><br />
</body>
</html>
```

Defining selectors

- Using the attribute value ends with selector

```
a[href$='jpg']:hover {
    background-color: yellow;
}
```

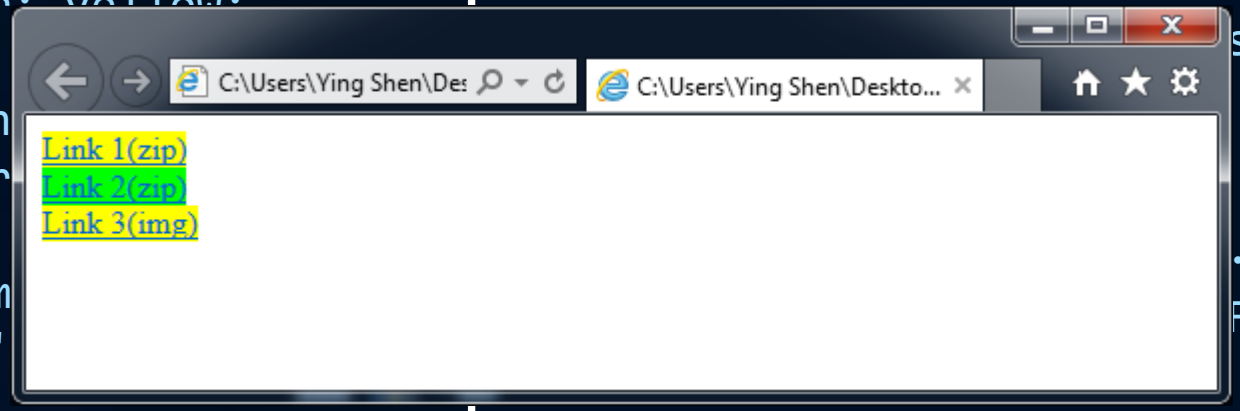



```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <a href='default.html'>Link 3</a><br />
</body>
</html>
```

Defining selectors

- Using the attribute cor

```
a[data-linktype~='externalLink'] {
    background-color: yellow;
}
a[data-linktype~='in
    background-color
}
a[data-linktype~='im
    content: '(img)'
}
a[data-linktype~='zipFile']:after {
    content: '(zip)';
}
```



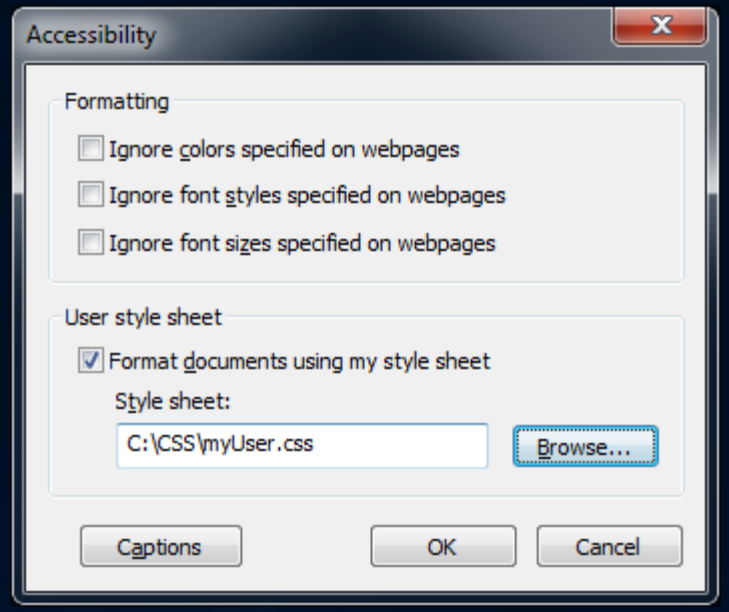

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
                                    s" rel="stylesheet" />
                                    .html' data-
                                    File'>Link 1</a><br />
                                    data-
linktype='internalLink zipFile'>Link 2</a><br />
    <a href='http://Microsoft.com/logo.jpg' data-
linktype='externalLink imageFile'>Link 3</a><br />
</body>
</html>
```

Understanding the browser's built-in styles

- Each browser has a built in style sheet, which is applied to all HTML docs before any other style sheets are applied

Extending browser styles with user styles

- Add a user-defined style sheet in IE
  - Tools | Internet Options | General | Accessibility



Working with important styles

- Use “!important” to increase the priority of user-defined style
  - Example

```
@charset 'UTF-8'] ;
body {
    background-color: white !important;
    color: black !important;
}
```

How do styles cascade?

- The order of precedence
  1. Important
  2. Specificity
  3. Textual order
- The evaluation order of style sheets
  1. Browser’s built-in style sheet
  2. User’s normal declarations in the user style sheet
  3. Author’s normal declarations in the author style sheet
  4. Author’s important declarations in the author style sheet
  5. User’s important declarations in the user style sheet

Using specificity

- The selector’ s specificity is calculated by a, b, and c
  - **a**: Record the number of id attributes
  - **b**: Record the quantity of class selectors, attributes selectors and pseudo classes
  - **c**: Record the quantity of element names

Using specificity

Lowest Precedence

Selector	a	b	c
*	0	0	0
li	0	0	1
ol + li	0	0	2
div ol + li	0	0	3
div .content	0	1	1
div .content ol + li	0	1	3
div .content ol + li .selected	0	2	3
#main	1	0	0
#main .selected	1	1	0
#main ul + li .selected	1	1	2

Highest Precedence

Understanding inheritance

- An element inherit styles from a parent element
- Using the inherit value
  - Example

```
body {
    font-size: x-large;
}
li:nth-child(even) {
    font-size: small;
}
li:nth-child(4) {
    font-size: inherit;
}
```