Web Programming

CSS (Cascading Style Sheet)

CSS (Cascading Style Sheet)

- Language that describes the look or presentation of markup languages
- Stylesheet language.

Three ways to use CSS:

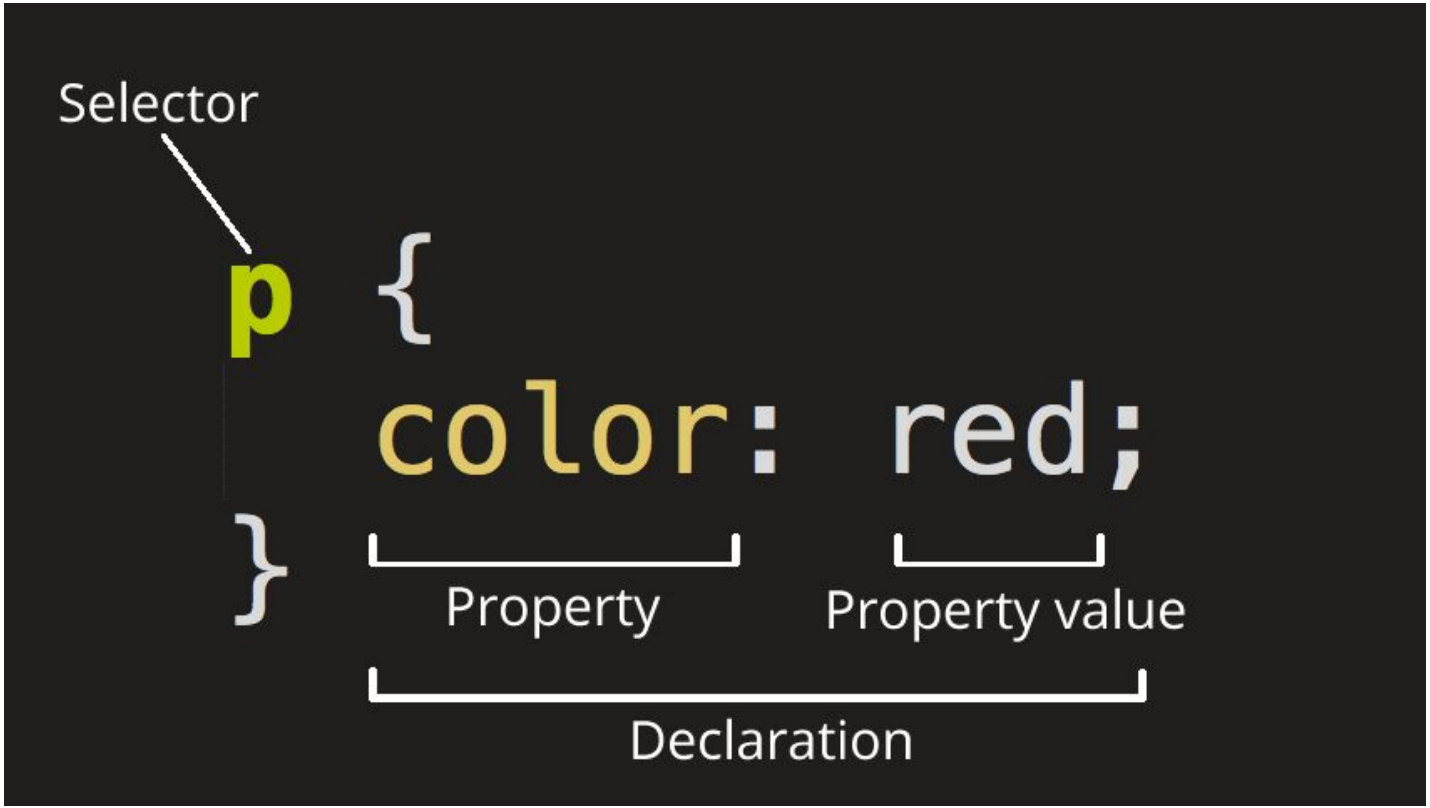
1. Via style attribute (inline style)
2. Via style tag in head
3. Via separate files

Method 1: Via style attribute (Inline Style)

```
<p style="color: red;">
   this content will be in red
</p>
```

this content will be in red

Anatomy of a CSS



Method 2: Via <style> tag

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title></title>
  <style>
    p {
      color: red;
      font-size: 20pt;
    }
  </style>
</head>
<body>
  <p>
    this content will be in red and font-size is 20pt
  </p>
</body>
</html>
```

Notice we call the tag name, followed by the curly braces and the style inside. Every style we add, we must end it with semicolon.

Using this style, all paragraph tags will have red font with font size as 20pt

Method 3: Via CSS files

What if:

- We want to add many styles?
- We put everything in HTML?
- We want to use same styles for other pages?

We could use separate file for our CSS. The extension of the file is .css, and we could add the link to the file in <head> tag

Adding .css file

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <title></title>
    <link href="style.css"
rel="stylesheet" />
</head>
<body>
    <p>
        Web Programming is < span>fun</span>!
    </p>
</body>
</html>
```

```css
/* style.css I am a comment :D */
p {
    text-decoration: underline;
    color: purple;
}

span {
    text-transform: uppercase;
}
```

Web Programming is FUN!

Selector

The target of style in HTML is called selector.

What if we add the same selector with different value?

```
/* style.css */
p {
    color: red;
}

p {
    color: blue;
}
```

Result is BLUE because the last one is used.
This is called “**Cascading**” rule

Multiple selectors

```
p, h1, div {
    color: red;
}
```

This means make all contents of paragraph, header 1 and div with red font

What if

What if we use unknown property or value?

Nothing will happen, so it is safe and our document is not formatted

```
p, h1, div {
    hohoho: red;
}
```

Types of selectors

Could we use different kind of selectors? We could and in fact we have 4 choices to use as selector in CSS:

1. **Type**: the one we have seen
2. **Attributes**: using HTML tag attribute
3. **ID**: the ID of HTML element
4. **Class**: the class of HTML element

HTML id and name attribute

Format / style certain elements in our HTML with **id** or **name** or any attribute in the HTML.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <title></title>
    <link href="style.css" rel="stylesheet" />
</head>
<body>
    <p id="myParagraph">
        this content will be in blue
    </p>

    <div name="myDiv">
        This is a content of a div
    </div>
</body>
</html>
```

```css
p[id=myParagraph] {
    color: blue;
}
div[name=myDiv] {
    color: red;
    font-weight: bold;
}
```

this content will be in blue

This is a content of a div

Besides that, we could just use id selector

- Use **#** before **id** in CSS to apply style only for elements with id “myParagraph”.
- All elements with **id=”myParagraph”** will be blue

```
#myParagraph {
    color: blue;
}
```

Class

- Use class attribute in tag to give HTML certain style.

```
<!DOCTYPE html>
<html lang="en">
<head>
 <title></title>
 <link href="style.css" rel="stylesheet"/>
</head>
<body>
 <div class="myClass">
   what color am I?
 </div>
</body>
</html>
```

```
.myClass {
  color: white;
  background-color: black;
}
```

White on Black

Selector with Class

- Add specific style to specific selector with certain class.
- Format: `selector.className`

```
p.myClass {
    color: red;
}
```

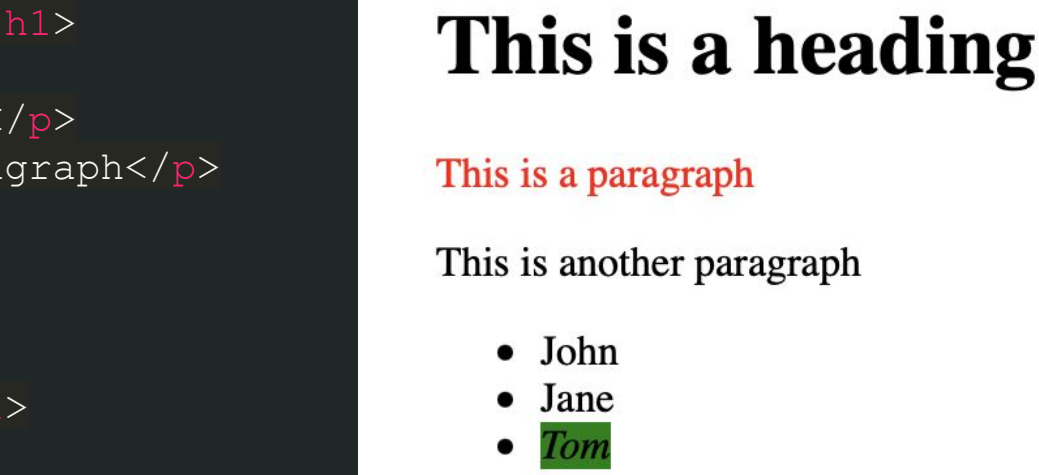
Combinator

- Make one paragraph after the heading to have red text
- Make the inside <li> has background color of green

```
<h1>This is a heading</h1>

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

<ul>
   <li>John</li>
   <li>Jane</li>
   <li><em>Tom</em></li>
</ul>
```



Combinator - Adjacent

If we want to change the style of element that comes exactly after an element, we use “**adjacent sibling combinator**” (with + sign).

```
h1 + p {
    color: red;
}
```

Combinator - Descendant

If we want to change the style of element that is exactly inside an element, we use “**descendant combinator**” (separated by space)

```
li em {
    background-color: green;
}
```

CSS based on state

Remember the hyperlink discussion? Where I stated that a link will change its color when you visit it but it didn’t? It is because we do not have distinguish style for our hyperlink element.

A hyperlink element has:

1. link (default): normal, unvisited link
2. visited: visited link
3. hover: when user hovers on the link
4. active: when user clicks on the link

Styling Links with Pseudoclass

```
<a href="https://puis.president.ac.id" target="_blank">PUIS</a>
```

```
a:link {
    color: green;
}
a:visited {
    color: black;
}
a:hover {
    background-color: red;
}
a:active {
    color: yellow;
}
```

Inheritance and Cascade

Cascade plays important roles in CSS (hence Cascading Style Sheet). This means if we have same element with same importance, the one applied is the last found in the file

```
<p>
   This will be in red
</p>
```

```
p {
   color: blue;
}

p {
   color: red;
}
```

Inheritance and Cascade

Different with cascade, inheritance in css means all of the styles applied in the higher level of element will also be applied to the children

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title></title>
    <link href="style.css" rel="stylesheet"/>
</head>
<body>
    <p>
        Because the body is blue, the paragraph inherits the
blue color except <span>this part</span>
    </p>
</body>
</html>
```

```
body {
    color: blue;
}

span {
    color: red;
}
```

NOW LET’S HAVE FUN

Let’s see what properties we could use

BUT do not limit yourself to these properties! Have fun and combine them :)

Remember: “Sky is the only limit”

Text Decoration

Property	Description	Possible Values
text-decoration-line	Decorates text with line	none underline overline line-through blink
text-decoration-color	Decorates text with color	*color*
text-decoration-style	Decorates text line with certain styles	solid double dotted dashed wavy
text-decoration-thickness	Decorates text line with certain thickness	auto from-font *number* *percentage*

CSS Shorthand

CSS is also Fun

- To simplify properties in certain cases
- From previous slide, we could use `text-decoration`

```
span {
  text-decoration-line: underline;
  text-decoration-style: wavy;
  text-decoration-color: green;
  text-decoration-thickness: 1px;
}

CSS is also <span>Fun</span>
```

```
span {
   text-decoration: underline wavy green 1px;
}

CSS is also <span>Fun</span>
```

Text Properties

Note: text in *italic* is variable

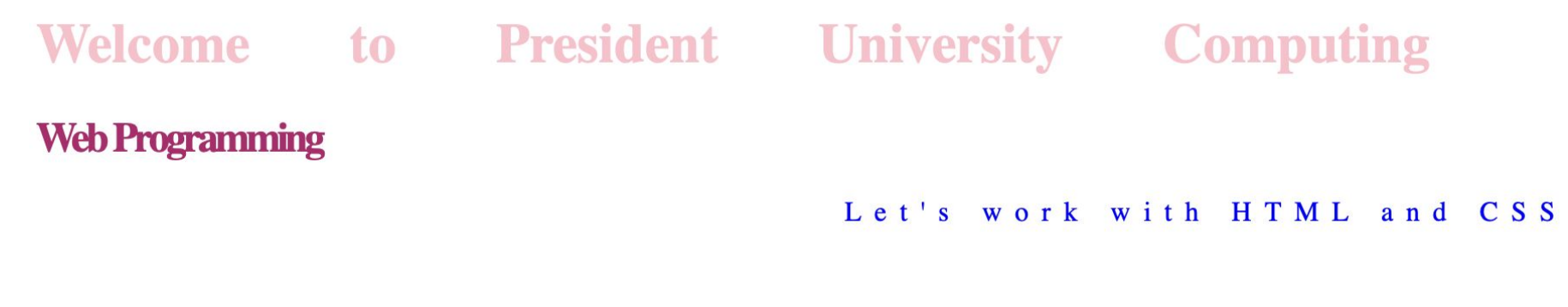
Property	Description	Possible Values
color	Set color to text	*color*
direction	Set direction of text	ltr rtl
line-height	Set distance between lines	normal *number* *length* *%*
letter-spacing	Set distance/space between letters	normal *length*
text-align	Aligns text in an element	left right center justify
text-indent	Indents first line of text in a block	*length* *%*
text-shadow	Add shadows to text	*color, X-value, Y-value, Blur* yellow, 1px, 2px, 10px
text-transform	Transform the case of text	none capitalize uppercase lowercase full-width full-size-kana
unicode-bidi	Handles the bidirectional text like Arabic, Hebrew	normal bidi-override plaintext isolate-override
white-space	Set how the white space in element is handled	normal nowrap pre pre-wrap pre-line break-spaces
word-spacing	Handles spacing between words	normal *length*

```css
h1 {
    color: pink;
    word-spacing: 50px;
}

h2 {
    color: #B31F6D;
    letter-spacing: -2px;
}

p {
    color: rgb(0, 0, 255);
    letter-spacing: 0.2cm;
    direction: rtl;
}
```

```html
<h1>Welcome to President University Computing</h1>
<h2>Web Programming</h2>
<p>Let's work with HTML and CSS</p>
```



```css
h1.headline {
    text-align: center;
    text-transform: capitalize;
    text-shadow: #f4ed30 10px 5px 1px;
}

p {
    text-indent: 10%;
    line-height: 30px;
}

div {
    width: 100px;
    white-space: break-spaces;
}
```

```html
<h1 class="headline">headline news in computing</h1>
<p>
    Lecturers for Web Programming:<br/>
    Mr. Rila<br/>
    Mr. Rusdianto<br/>
    Mr. Ronny<br/>
    Mr. Christian<br/>
    Williem<br/>
</p>
<div>Today we learn about CSS fundamentals and basic properties</div>
```

Let’s continue

https://docs.google.com/document/d/1lLeE7zlH7MKYUo7AWPXqv2CmRYQs3Pf3/edit?usp=sharing&ouid=112512321509221893227&rtpof=true&sd=true

References

https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/Getting_started

https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/CSS_basics

https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/How_CSS_is_structured#selectors

https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Cascade_and_inheritance

https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/How_CSS_works

https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks