

Lecture 5

Review

We have seen three ways to make selectors so far.

Review

We have seen three ways to make selectors so far.

By tag name:

```
[tag name]
{
  attribute 1
  ⋮
  attribute n
}
```

Review

We have seen three ways to make selectors so far.

Example:

```
span
{
    background: rgb(0,25,100);
    color: white;
}
```

All span tags will have background color `rgb(0,25,100)` and white text.

Review

We have seen three ways to make selectors so far.

By the id attribute:

```
#[id]
{
  attribute 1
  ⋮
  attribute n
}
```

Review

We have seen three ways to make selectors so far.

Example:

```
#sometag  
{  
    background: rgb(0,25,100);  
    color: white;  
}
```

Only the tag with the HTML attribute `id="sometag"` `rgb(0,25,100)` and white text.

Review

We have seen three ways to make selectors so far.

By tag name and the id attribute:

```
[tag name]#[id]
{
  attribute 1
  ⋮
  attribute n
}
```

Review

We have seen three ways to make selectors so far.

Example:

```
div#sometag  
{  
    background: rgb(0,25,100);  
    color: white;  
}
```

Only the div tag with the HTML attribute `id="sometag"` `rgb(0,25,100)` and white text.

Review

We have seen three ways to make selectors so far.

Example:

```
div#sometag
{
  background: rgb(0,25,100);
  color: white;
}
```

The tag `` is not selected because it is not a div.

Review

We have seen three ways to make selectors so far.

Example:

```
div#sometag
{
  background: rgb(0,25,100);
  color: white;
}
```

The tag `<div id="someothertag">` is not selected because it does not have the attribute `id="sometag"`

Selecting Multiple Tags

- The id attribute should be unique for all tags.
- This means that if you want to alter several instances of a tag type, you need to select each of them by id.
- Here is one (bad) way to do it:

```
#[tag 1 id], #[tag 2 id], ... , #[tag n id]  
{  
    [css attributes]  
}
```

Selecting Multiple Tags

- A comma-separated list of selectors will select any tag that matches one of the selectors in the list.
- This can be used to apply the same CSS attributes to multiple tags.
- This is generally bad because you can wind up with a huge list and you need to update it every time you add a tag.

Selecting Multiple Tags

- There is a better way to select multiple tags at once.
- There is another HTML attribute that can be used to make selectors.

The class attribute

- The class HTML attribute lets you create a set of tags that can be selected with just one selector.
- The class attribute does not need to be unique.

Here is an example:

```
<div class="something">  
</div>
```

Selecting tags by class

Now you need to add the selector to your stylesheet.

The syntax looks like this:

```
[tag name].[class name]
{
  [css attributes]
}
```

Selecting tags by class

- [tag name] is the type of tag.
- [class name] is the value of the class attribute.

This is very similar to selecting tags by id.

By class:

```
[tag name].[class name]
{
  [css attributes]
}
```

By id:

```
[tag name]#[tag id]
{
  [css attributes]
}
```

Selecting tags by class

Example:

HTML

```
<div class="red">
Welcome</div>
<div class="blue">
to</div>
<div class="red">
my</div>
<div class="blue">
web</div>
<div class="red">
page!</div>
```

CSS

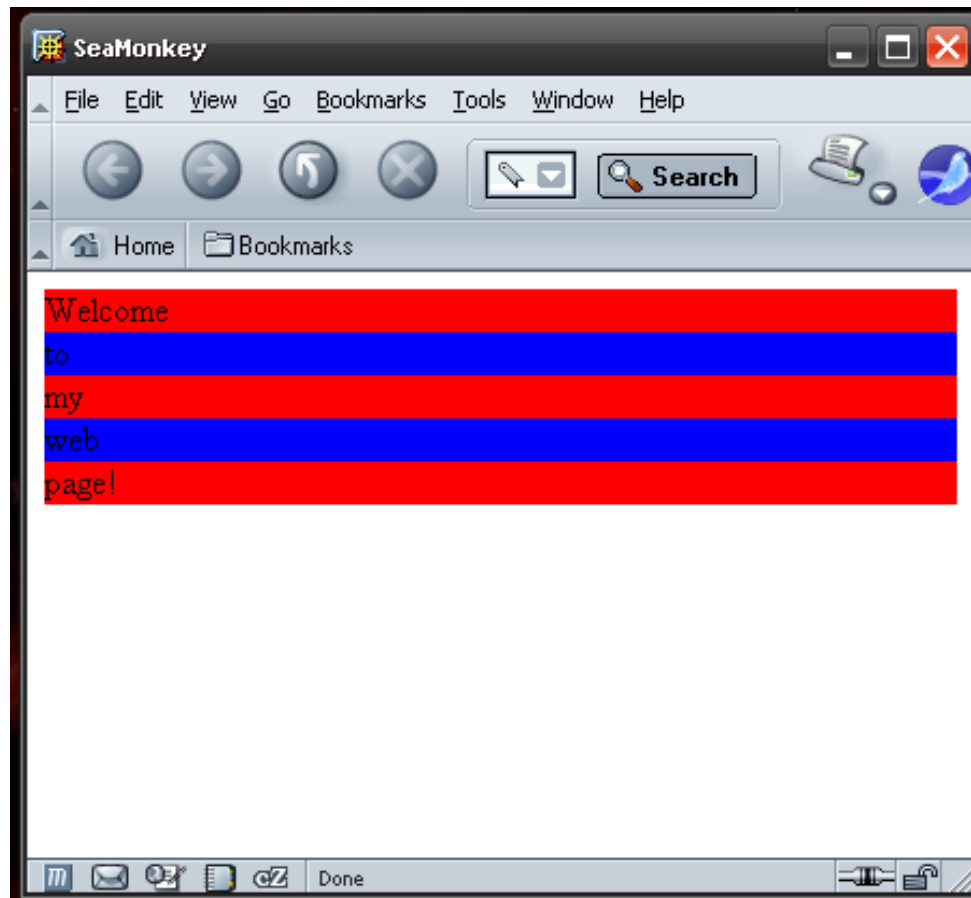
```
div.red
{
    background: red;
}

div.blue
{
    background: blue;
}
```

Selecting tags by class

Example:

Here's what you see:



Selecting tags by class

- With the id attribute, you did not need to provide the tag type in your selectors
- You do not need to specify the tag type with the class attribute either.

Selecting tags by class

Again, the syntax looks similar:

By class:

```
.[class name]
{
    [css attributes]
}
```

By id:

```
#[tag id]
{
    [css attributes]
}
```

Selecting tags by class

Consider this example:

```
<div class="class1">  
  Hello  
</div>
```

```
<span class="class1">  
  world!  
</span>
```

Selecting tags by class

The stylesheet:

```
.class1
{
    color: yellow;
}
div.class1
{
    background: red;
}
span.class1
{
    background: blue
}
```

Selecting tags by class

There are three selectors. They are `.class1`, `div.class1`, and `span.class1`.

The first selector, `.class1`, selects any tag with `class="class1"`. What tags are selected?

Selecting tags by class

The HTML:

```
<div class="class1"> ←  
  Hello  
</div>
```

This tag is selected
by .class1 because
it has class="class1"

```
<span class="class1">  
  world!  
</span>
```

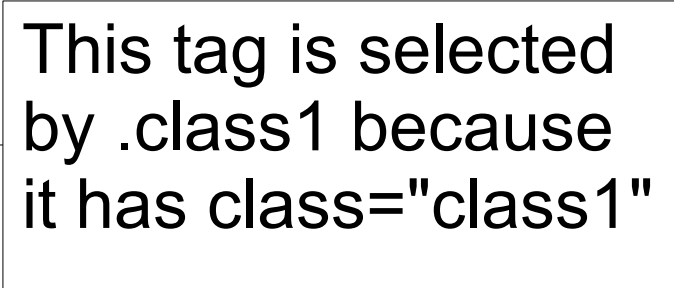
Selecting tags by class

The HTML:

```
<div class="class1">  
  Hello  
</div>
```

```
<span class="class1">  
  world!  
</span>
```

This tag is selected
by .class1 because
it has class="class1"



Selecting tags by class

There are three selectors. They are `.class1`, `div.class1`, and `span.class1`.

The second selector, `div.class1`, selects any `div` tag with `class="class1"`. What tags are selected?

Selecting tags by class

The HTML:

```
<div class="class1"> ←  
  Hello  
</div>
```

This tag is selected by div.class1 because it has class="class1" AND it is a div tag.

```
<span class="class1">  
  world!  
</span>
```

Selecting tags by class

The HTML:

```
<div class="class1">  
  Hello  
</div>
```

```
<span class="class1">  
  world!  
</span>
```

This tag is NOT selected by `div.class1` because it has `class="class1"` but it is not a `div` tag.

Selecting tags by class

There are three selectors. They are `.class1`, `div.class1`, and `span.class1`.

The third selector, `span.class1`, selects any `span` tag with `class="class1"`. What tags are selected?

Selecting tags by class

The HTML:

```
<div class="class1"> ←  
  Hello  
</div>
```

This tag is NOT selected by `span.class1` because it has `class="class1"` but it is not a `span` tag.

```
<span class="class1">  
  world!  
</span>
```

Selecting tags by class

The HTML:

```
<div class="class1">  
  Hello  
</div>
```

```
<span class="class1">  
  world!  
</span>
```

This tag is selected by span.class1 because it has class="class1" and it is a span tag.

Selecting tags by class

Let's look at the css again.

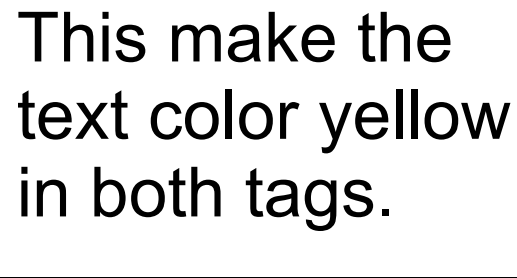
```
.class1
{
    color: yellow;
}
div.class1
{
    background: red;
}
span.class1
{
    background: blue
}
```

Selecting tags by class

Let's look at the css again.

```
.class1
{
    color: yellow;
}
div.class1
{
    background: red;
}
span.class1
{
    background: blue
}
```

This make the
text color yellow
in both tags.

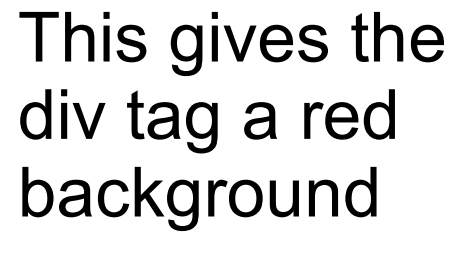


Selecting tags by class

Let's look at the css again.

```
.class1
{
    color: yellow;
}
div.class1
{
    background: red;
}
span.class1
{
    background: blue
}
```

This gives the
div tag a red
background

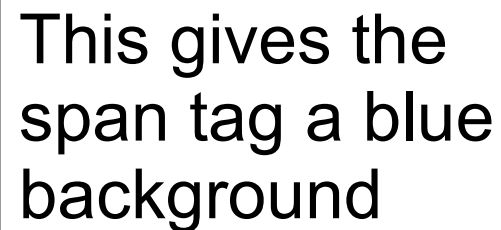


Selecting tags by class

Let's look at the css again.

```
.class1
{
    color: yellow;
}
div.class1
{
    background: red;
}
span.class1
{
    background: blue;
}
```

This gives the span tag a blue background



Precedence

Here's another important example:

HTML:

```
<div class="foo">  
Hello  
</div>  
<div class="foo" id="bar">  
World!  
</div>
```

CSS:

```
.foo  
{  
    color: green;  
    background: red;  
}  
#bar  
{  
    background: blue;  
}
```

Precedence

- The second div tag is selected by two different selectors.
- One selector selects by id.
- One selector selects by class.
- Two different background colors are given.

Precedence

- The most specific selector wins.
- id is more specific than class.
- The id selector wins.

Precedence

HTML:

```
<div class="foo">  
Hello  
</div>  
<div class="foo" id="bar">  
World!  
</div>
```

CSS:

```
.foo  
{  
    color: green;  
    background: red;  
}  
#bar  
{  
    background: blue;  
}
```

- Both divs get green text.

Precedence

HTML:

```
<div class="foo">  
Hello  
</div>  
<div class="foo" id="bar">  
World!  
</div>
```

CSS:

```
.foo  
{  
    color: green;  
    background: red;  
}  
#bar  
{  
    background: blue;  
}
```

- The first div tag gets a red background.

Precedence

HTML:

```
<div class="foo">  
Hello  
</div>  
<div class="foo" id="bar">  
World!  
</div>
```

CSS:

```
.foo  
{  
    color: green;  
    background: red;  
}  
#bar  
{  
    background: blue;  
}
```

- The second div tag gets a blue background.

Precedence

When resolving conflicts, the order of precedence is:

browser defaults	lowest
tag type selectors	.
class selectors	.
id selectors	.
the style attribute	highest

Testing Precedence

- Sometimes it is easier to test precedence than to try and work out the rules.
- Suppose you have the following selectors and you want to see how conflicts are resolved.

```
[selector a]
{
    [css attributes]
}
[selector b]
{
    [css attributes]
}
```

Testing Precedence

Here's any easy way to test it:

```
[selector a]
{
    background: red;
}
[selector b]
{
    background: blue;
}
```

Just look at a tag that is selected by both and see what its background color is.

Warning!

- Precedence only matters when there are conflicts.
- Only attributes that are overridden by a more specific selector will be ignored.
- It is not "all or nothing"
- Overriding one CSS attribute does not affect the others.

Comments

- Sometimes you'll want to make notes in your code that don't display on your Web page.
- You can do that with comments.
- The syntax looks like this

```
<!-- this text will not display -->
```

Comments

You can also add comments to your CSS file:

- The syntax looks like this

```
/* This text is ignored */
```