

- ❖ **Javascript**
- ❖ **jQuery**
- ❖ **AJAX**

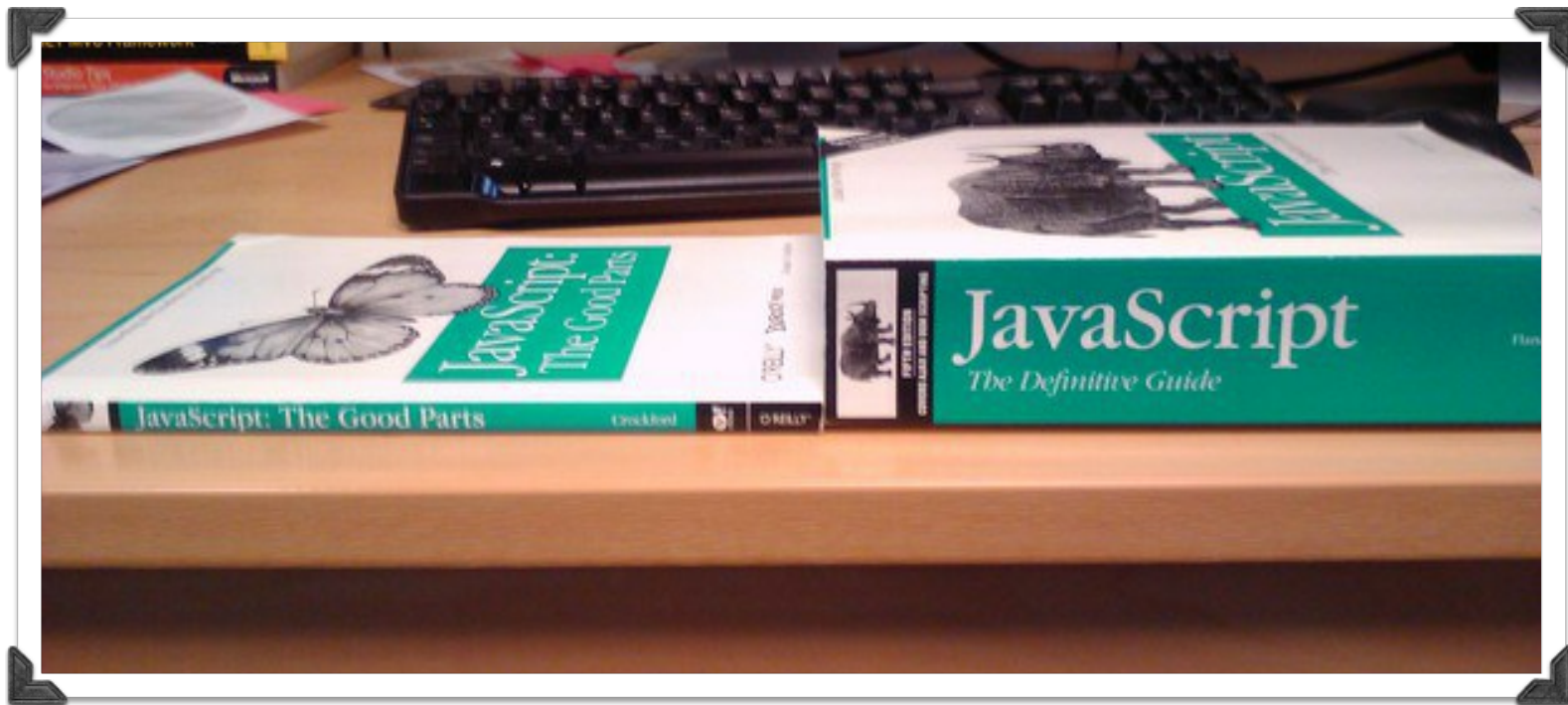
Why Javascript?

- 1. To change HTML code in the browser**
- 2. For asynchronous http requests**

Javascript 101

- ✓ **Yet Another Programming Language**
- ✓ **Has nothing to do with the Java language**
- ✓ **Same IPO Pattern**
- ✓ **Sort of object-oriented**
- ✓ **More function-oriented**
- ✓ **Runs inside the browser**

Javascript 101



Ruby vs. Javascript

```
def say_weather  
  temp = "72 degrees"  
  puts "It is #{temp}"  
end
```

Ruby vs. Javascript

```
def say_weather  
  temp = "72 degrees"  
  puts "It is #{temp}"  
end
```

```
function sayWeather() {  
  var temp = "72 degrees";  
  alert("It is " + temp);  
};
```

Ruby vs. Javascript

```
def say_weather  
  temp = "72 degrees"  
  puts "It is #{temp}"  
end
```

```
def say_weather(temp)  
  puts "It is #{temp}"  
end
```

```
function sayWeather() {  
  var temp = "72 degrees";  
  alert("It is " + temp);  
};
```

Ruby vs. Javascript

```
def say_weather  
  temp = "72 degrees"  
  puts "It is #{temp}"  
end
```

```
def say_weather(temp)  
  puts "It is #{temp}"  
end
```

```
function sayWeather() {  
  var temp = "72 degrees";  
  alert("It is " + temp);  
};
```

```
function sayWeather(temp)  
{  
  alert("It is " + temp);  
};
```


Ruby vs. Javascript

```
def say_weather  
  temp = "72 degrees"  
  puts "It is #{temp}"  
end
```

```
def say_weather(temp)  
  puts "It is #{temp}"  
end
```

```
do |temp|  
  puts "It is #{temp}"  
end
```

```
function sayWeather() {  
  var temp = "72 degrees";  
  alert("It is " + temp);  
};
```

```
function sayWeather(temp)  
{  
  alert("It is " + temp);  
};
```

Ruby vs. Javascript

```
def say_weather  
  temp = "72 degrees"  
  puts "It is #{temp}"  
end
```

```
def say_weather(temp)  
  puts "It is #{temp}"  
end
```

```
do |temp|  
  puts "It is #{temp}"  
end
```

```
function sayWeather() {  
  var temp = "72 degrees";  
  alert("It is " + temp);  
};
```

```
function sayWeather(temp)  
{  
  alert("It is " + temp);  
};
```

```
function(temp) {  
  alert("It is " + temp);  
};
```

Javascript Part 1:

Modify HTML Code in the Browser.

Javascript

HTML5 Support:

```
<script>  
    alert("Hello!");  
</script>
```

Javascript

1. Create an HTML file that includes:

```
<p>  
  Click for the current weather  
</p>
```

2. Then add this code anywhere in your HTML:

```
<script>  
  alert("Hello!");  
</script>
```

Javascript

1. Create

```
<p>  
Click  
</p>
```

2. Then add

```
<script>
```

```
    alert("Hello!");
```

```
</script>
```

Q. Does script location matter?

A. Experiment by moving the script element to different places, and observe any effects.

Obtrusive Javascript

<p>

Click here for the current weather.

</p>

How can we show an alert box only
after they click the text?

Obtrusive Javascript

HTML supports Javascript **callbacks**.

- ▶ Use an HTML **event** to start your code.
- ▶ Consider the return value from your code.

Obtrusive Javascript

```
<p onclick="alert('It is 72 degrees.');">  
  Click for the current weather  
</p>
```

Choose an HTML event and write a callback handler in Javascript.

Obtrusive Javascript

```
<a href="http://www.google.com"  
  onclick="alert('It is 72 degrees.');">  
  Click for the current weather  
</a>
```

What if we try it with an `<a>` tag?

Obtrusive Javascript

```
<a href="http://www.google.com"  
  onclick="alert('It is 72 degrees. ');  
          return false;">  
  Click for the current weather  
</a>
```

Callbacks can have return values!

Obtrusive Javascript Lab

```
<script>
  function sayWeather() {
    // Some code here
  }
</script>

<a href="http://www.google.com"
  onclick="???????">
  Click for the current weather
</a>
```

Have the callback invoke a Javascript function to show the weather.

Javascript Callbacks

```
function saySomething(greeting) {  
    alert("About to say something...");  
    greeting("Jeff");  
};
```

```
saySomething(function(name) {  
    alert("Hello " + name);  
});
```

JavaScript + The DOM

Javascript DOM Inspection

```
<h1 id="title">Wazzzzzzup!</h1>
```

```
<script>
```

```
    var e = document.getElementById("title");
```

```
    alert(e.innerHTML);
```

```
</script>
```

Javascript DOM Inspection

```
<h1 id="title">Wazzzzzzup!</h1>
```

```
<script>
```

```
  var e = document.getElementById("title");
```

```
  alert(e.innerHTML);
```

```
</script>
```

Use the Web Inspector

Javascript DOM Inspection

```
<h1 id="title">Wazzzzzzup!</h1>
```

```
<script>
```

```
  var e = document.getElementById("title");
```

```
  alert(e.innerHTML);
```

```
</script>
```

Try Changing the Order

Javascript DOM Inspection

```
<h1 id="title">Wazzzzzzup!</h1>
```

```
<script>
```

```
  var e = document.getElementById("title");
```

```
  alert(e.innerHTML);
```

```
</script>
```

Use the **.style** property to change the text color of the h1 element.

Javascript DOM Inspection

`<p>Hello</p>`

`<p>Goodbye</p>`

`<p>Wazzup</p>`

Try to change the text color of all paragraph elements.

w3schools.com

HINT: Figure out a way to select elements by tag name.

Javascript DOM Inspection

```
<p>Hello</p>
```

**Make the paragraph
disappear when the link is
clicked.**

```
<a href="#">Abracadabra!</a>
```

w3schools.com

jQuery

JQuery

```
<script src="...">  
</script>
```

**You can link to a CDN-hosted script
or a local copy.**

JQuery

jQuery()

JQuery

\$()

JQuery

`$()`

This function lets you "grab" an object from the DOM.

JQuery Selectors

<p>Hello!</p>

<p>Goodbye!</p>

JQuery Selectors

<p>Hello!</p>

<p>Goodbye!</p>

`$("p")` \Rightarrow *2 elements*

JQuery Selectors

```
<div id="header">  
  <h1>My Website</h1>  
</div>
```

`$("#header")` \Rightarrow *1 element*

JQuery Selectors

You can also select the entire DOM document:

```
$(document)
```

JQuery Selectors

You can also select the entire DOM document and call methods on it:

```
$(document).ready(...)
```

JQuery Code

\$

Let's show an alert box when the page is loaded and ready to use.

JQuery Code

```
$(document)
```

Let's show an alert box when the page is loaded and ready to use.

JQuery Code

```
$(document).ready
```

Let's show an alert box when the page is loaded and ready to use.

JQuery Code

```
$(document).ready(function() {  
    alert("The page has loaded!");  
});
```

Let's show an alert box when the page is loaded and ready to use.

JQuery Code

```
$(function() {  
    alert("The page has loaded!");  
});
```

Let's show an alert box when the page is loaded and ready to use.

Event Handlers

JQuery Event Callbacks

```
<p onclick="alert('Hello Chicago!')">Chicago</p>  
<p onclick="alert('Hello Boston!')">Boston</p>  
<p onclick="alert('Hello Detroit!')">Detroit</p>  
<p onclick="alert('Hello LA!')">LA</p>
```



JQuery Event Callbacks

```
$(function() {  
    $("p").on("click", function() {  
        alert("Hello ...!");  
    })  
});
```

```
<p onclick="alert('Hello Chicago! ')">Chicago</p>  
<p onclick="alert('Hello Boston! ')">Boston</p>  
<p onclick="alert('Hello Detroit! ')">Detroit</p>  
<p onclick="alert('Hello LA! ')">LA</p>
```

JQuery Event Callbacks

```
$(function() {  
    $("p").on("click", function(e) {  
        alert("Hello " + e.target.innerHTML);  
    })  
});
```



```
<p onclick="alert('Hello Chicago!')">Chicago</p>  
<p onclick="alert('Hello Boston!')">Boston</p>  
<p onclick="alert('Hello Detroit!')">Detroit</p>  
<p onclick="alert('Hello LA!')">LA</p>
```

jQuery UI

jQuery + Rails

jQuery in Rails

Rails 3.1+ has built-in support for jQuery Core and jQuery UI

However, jQuery UI is not included in your app by default.

//= require jquery-ui

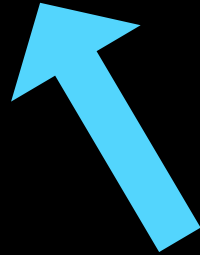
Ajax

Ajax

**Asynchronous
Javascript
And
XML**

Ajax

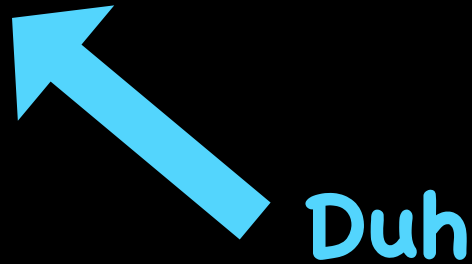
**Asynchronous
Javascript
And
XML**



User does not
have to wait

Ajax

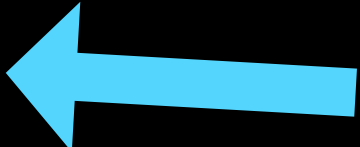
**Asynchronous
Javascript
And
XML**



Duh

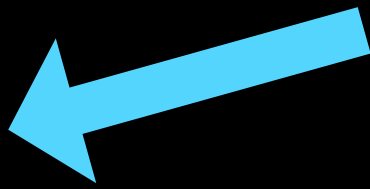
Ajax

**Asynchronous
Javascript**

And XML  **Lame**

Ajax

Asynchronous Javascript And XML



Ummm...

Nowadays it's often
Javascript, JSON, or
HTML instead.

Ajax

Asynchronous Javascript And XML

Using Ajax is completely optional.
It's primary purpose is to
(maybe) enhance your site's UX.

Ajax 3-Step Recipe

1. `:remote => true`

Add this option to any link or form to submit an AJAX request instead of a blocking request.

Ajax 3-Step Recipe

2. Respond to JS

Enhance your `respond_to` block to accept JS requests.

Ajax 3-Step Recipe

3. Generate a JS response

Generate Javascript instead of HTML.
You can use a view template if you want.