Boot Camp JavaScript

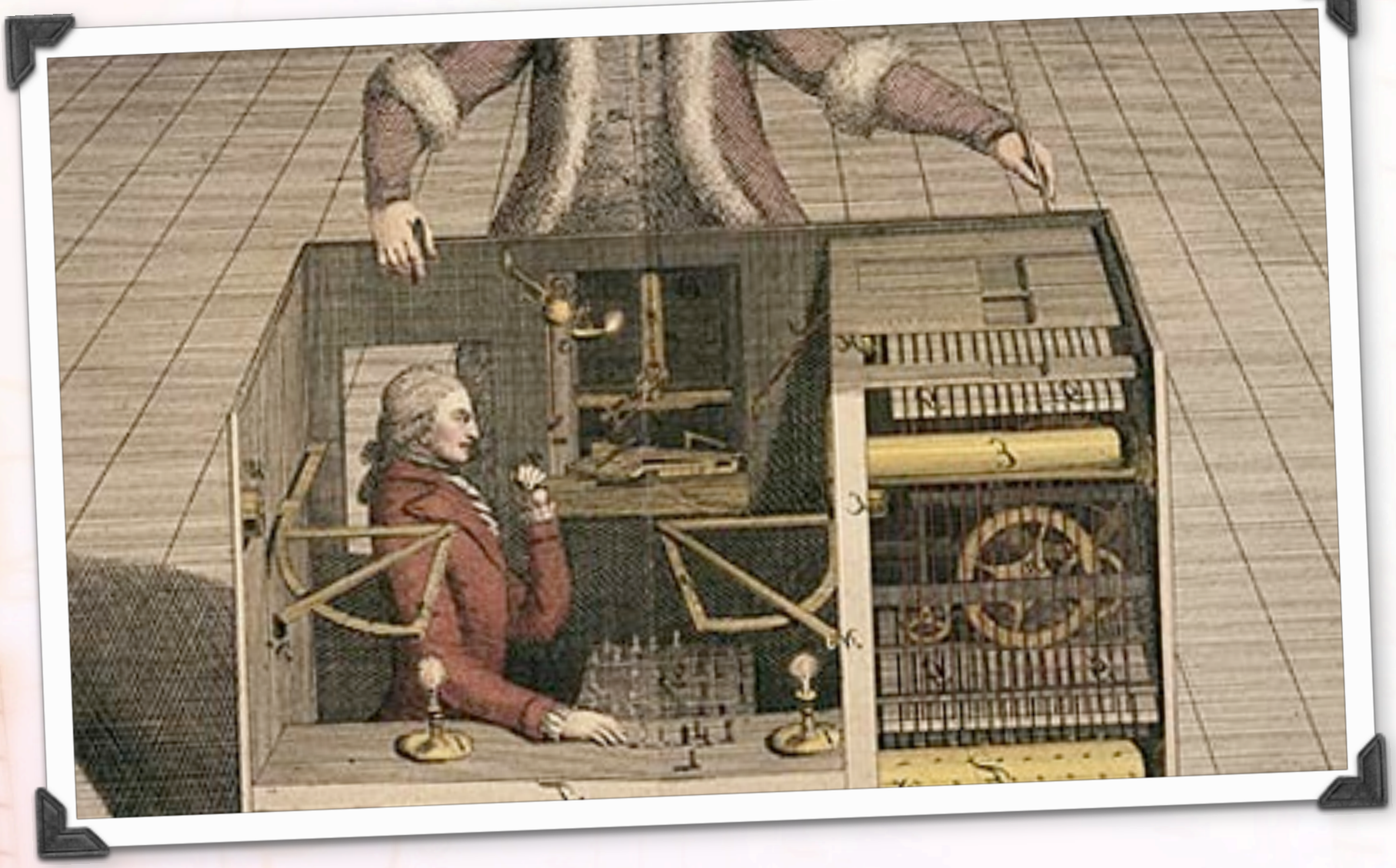
http://rix0r.nl/bootcamp

Part 2

Sioux, April 28, 2011

SIOUX

BOOT camp



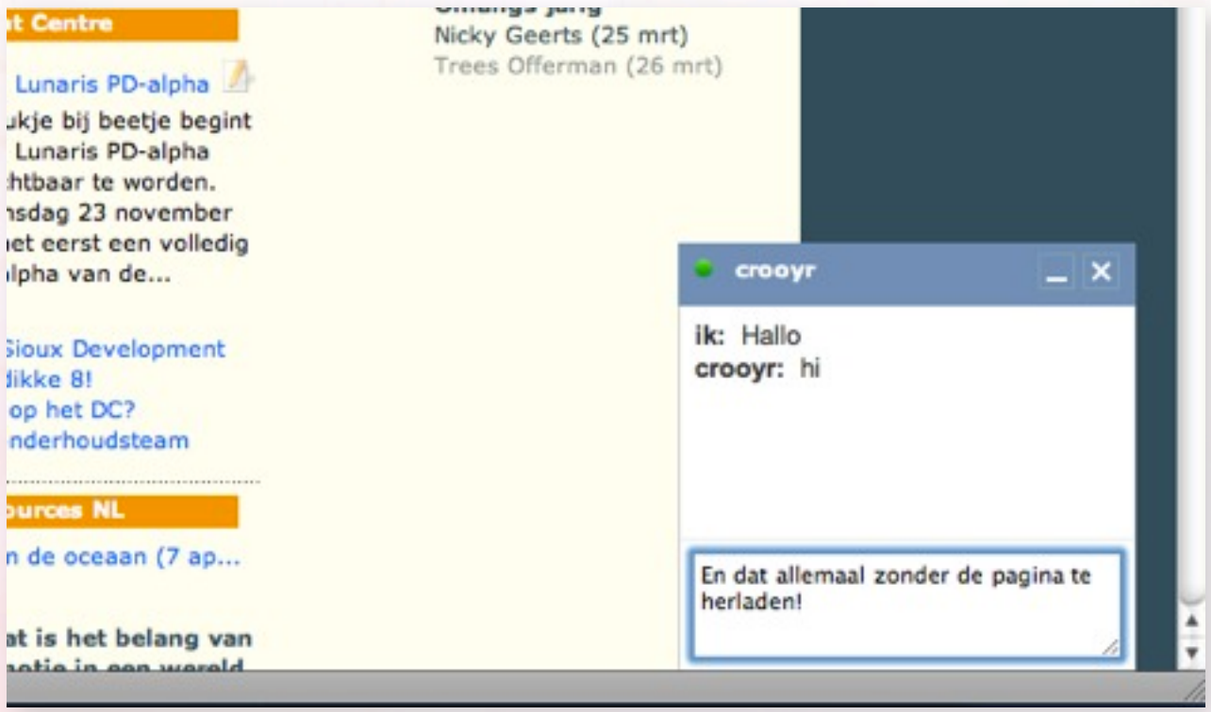
Part 2

JavaScript in the browser

Why?

Add client-side behaviour to pages
(animation, validation, autocomplete, ...)

Change page without refresh.



Build advanced web applications.

Two flavors of pages

Progressive enhancement
Static HTML page with some scripting

(autocomplete, drag&drop, ...)

Single Page Application
Empty page, add everything with JS

Web application in JavaScript, using the browser as an advanced rendering engine.

Two parts to a browser

JavaScript Engine

Rendering Engine

Two parts to a browser

JavaScript Engine

Document

Rendering Engine

Two parts to a browser

Two parts to a browser

Model

View

Popular browser pieces

	Firefox	Chrome	Safari	IE
Rendering Engine	Gecko	WebKit		Trident
JavaScript Engine	Spider Monkey	V8	Squirrel Fish	? Chakra

DOM

Using JavaScript, you call into the browser object model.

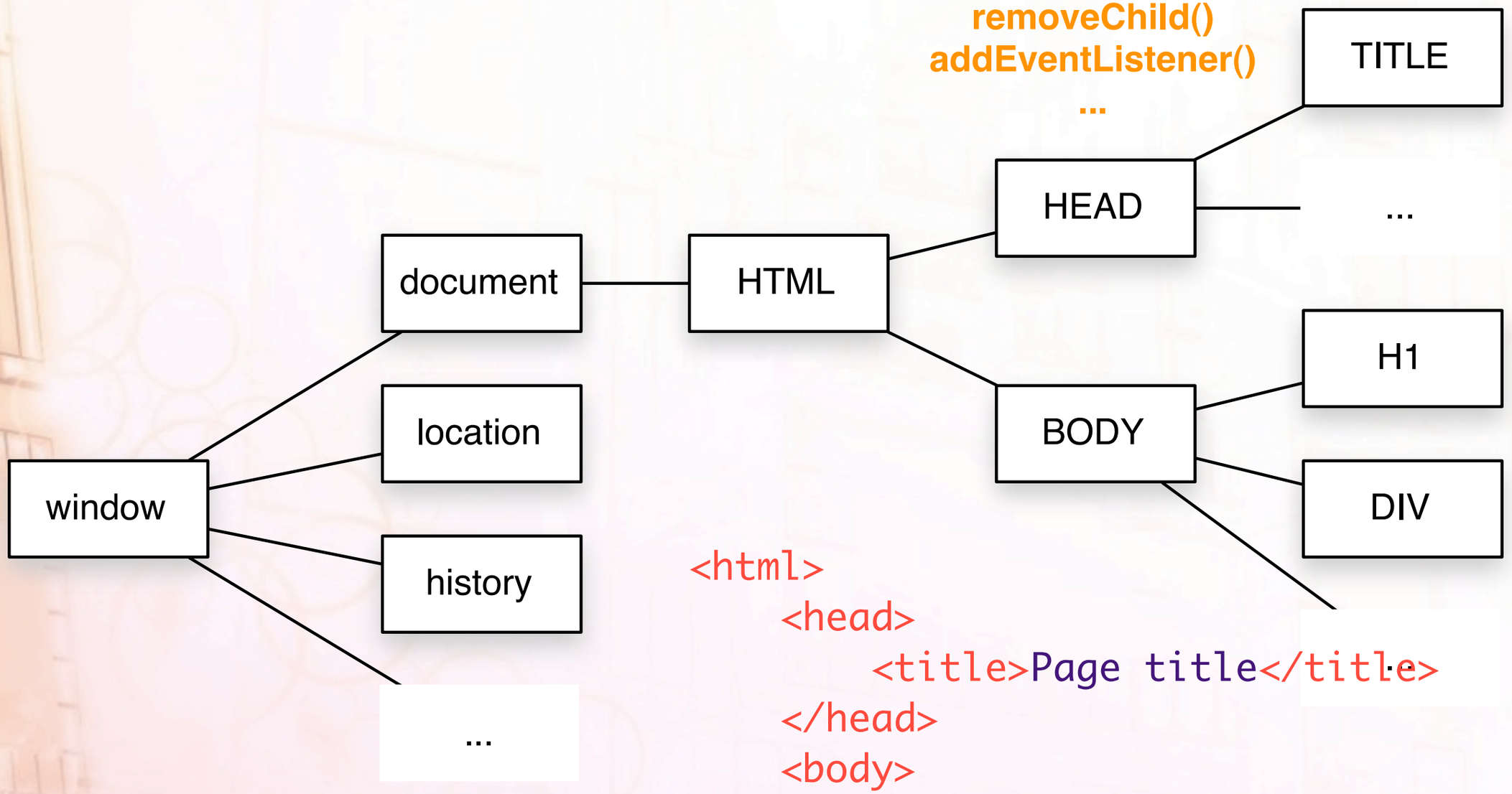
Start at global object, representing the browser: `window`

Interface to HTML document is called the Document Object Model (DOM)

Change the document via the DOM and the Rendering Engine shows the change.

The DOM

appendChild()
removeChild()
addEventListener()
...




```
<html>
    <head>
        <title>Page title</title>
    </head>
    <body>
        <h1>Header element</h1>
        <div>Some text in a DIV</div>
    </body>
</html>
```

Don’t use the DOM

In theory, it has been standardized by the W3C.

In practice, lots of details are different between browsers.

The programming model is cumbersome (Java-inspired longhand).

Use a JavaScript library!

Lots of libraries available

jQuery

Prototype/script.aculo.us

Dojo

MooTools

YUI

...

Libraries

jQuery

Designer's library: lots of existing functions for stuff you want to do.

MooTools

Developer's library: primitives for building your own extensions (class system, mixins, ...)

jQuery

`$(...)`

Almost everything is done calling this function.

What happens depends on the argument.

Manipulating DOM node

HTML

```
<div id="foo"></div>
```

Standard DOM

```
var el = document.getElementById("foo");
var text = document.createTextNode("Hello");
el.appendChild(text);
```

jQuery

```
$("#foo").append("Hello");
```

Manipulating DOM node

No relationship to <input name="...">

HTML

```
<div id="foo"></div>
```

Standard DOM

```
var el = document.getElementById("foo");
var text = document.createTextNode("Hello");
el.appendChild(text);
```

jQuery

```
$("#foo").append("Hello");
```

Manipulating DOM node

No relationship to <input name="...">

HTML

```
<div id="foo"></div>
```

Standard DOM

```
var el = document.getElementById("foo");
var text = document.createTextNode("Hello");
el.appendChild(text);
```

jQuery

```
$("#foo").append("Hello");
```

Any CSS selector works! (#id, .class)

The jQuery way

```
$("#logo").attr("src", "logo.jpg")
  .css("left", "10px")
  .toggleClass("hidden");
```

```
$("#info").text("Info here");
alert( $("#info").text() );
```

Chaining object calls(*)

Same method, different arity for **get** and **set**

* *Commonly called a "Fluent Interface"*

Animation

You can schedule a function to be called in the future.

```
window.setTimeout(function() {
    alert("This appears after a while...");
}, 10000);

window.setInterval(function() {
    alert("This popup will annoyed you...");
}, 100);
```

Animation simply comes from doing a little bit every invocation.

Exercise 5a

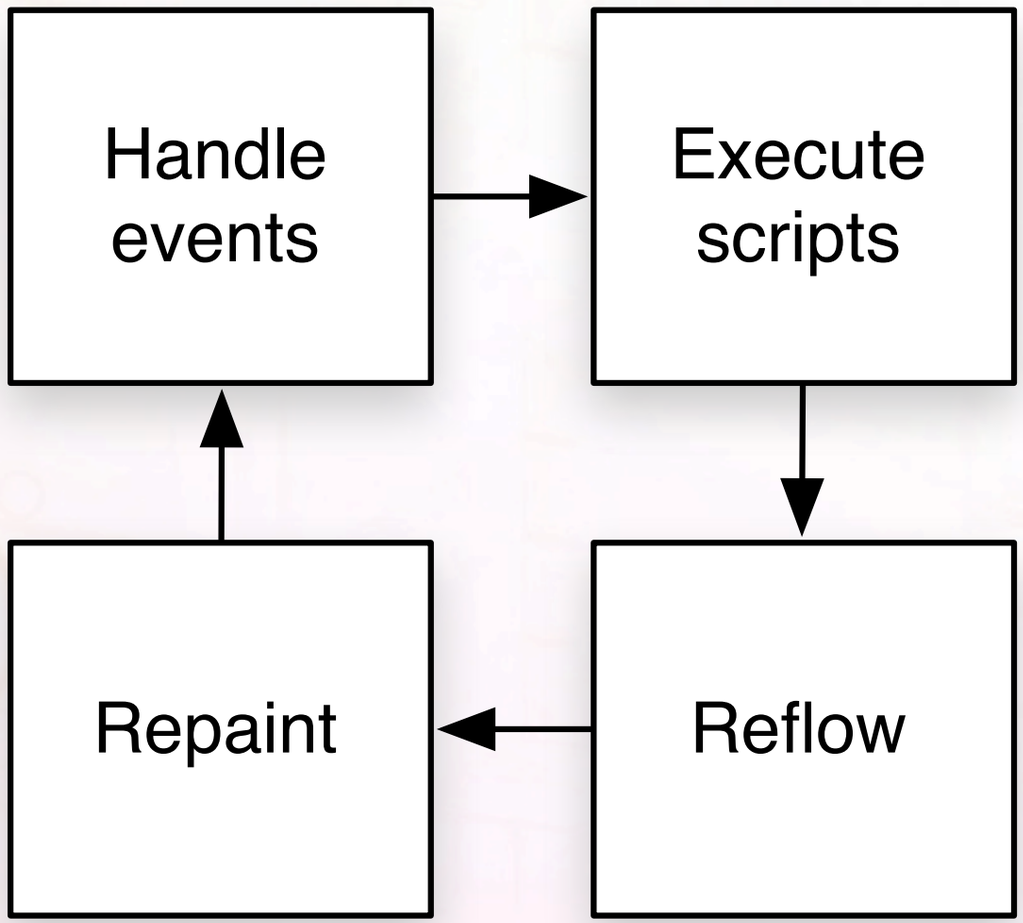
Bouncing block

Manipulate the DIV using jQuery.

To make it interesting, do it as a function of time.

You'll need to update the CSS properties `left` and `top` using the `.css()` method.

Browser loop



Event-driven

Code is only executed in response to events!

Special startup event (domready) in which a JS application hooks up other event listeners and then waits.

Hooking up events

Old-school:

```
div.onclick = function() {
    alert("Click!");
}
```

DOM2:

```
div.addEventListener('click', function() {
    alert("Click!");
}, false);
```

jQuery:

```
$(div).click(function() {
    alert("Click!");
});
```

Exercise 6

Higher/lower game

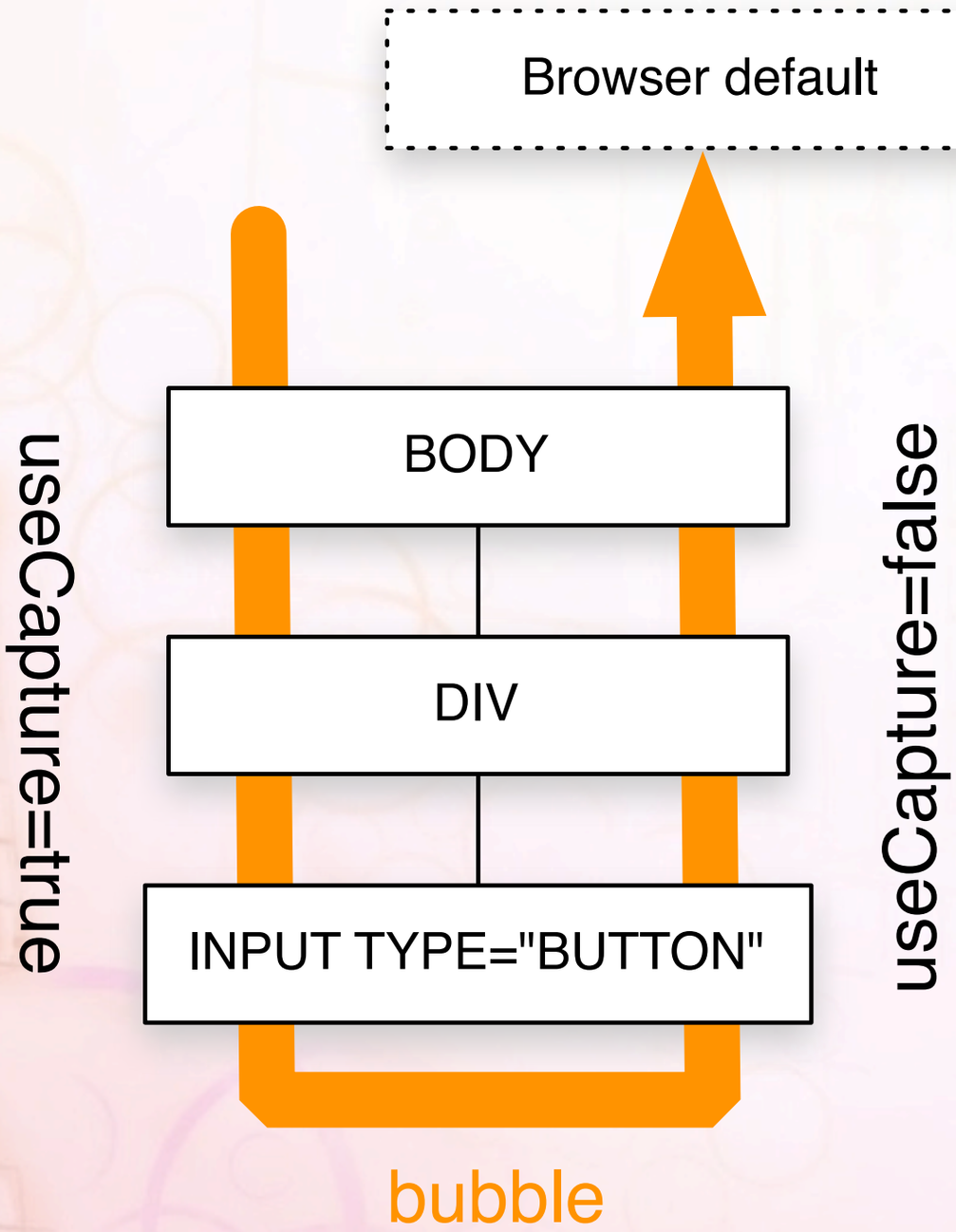
Hook up events on startup

Do calculations in response to events

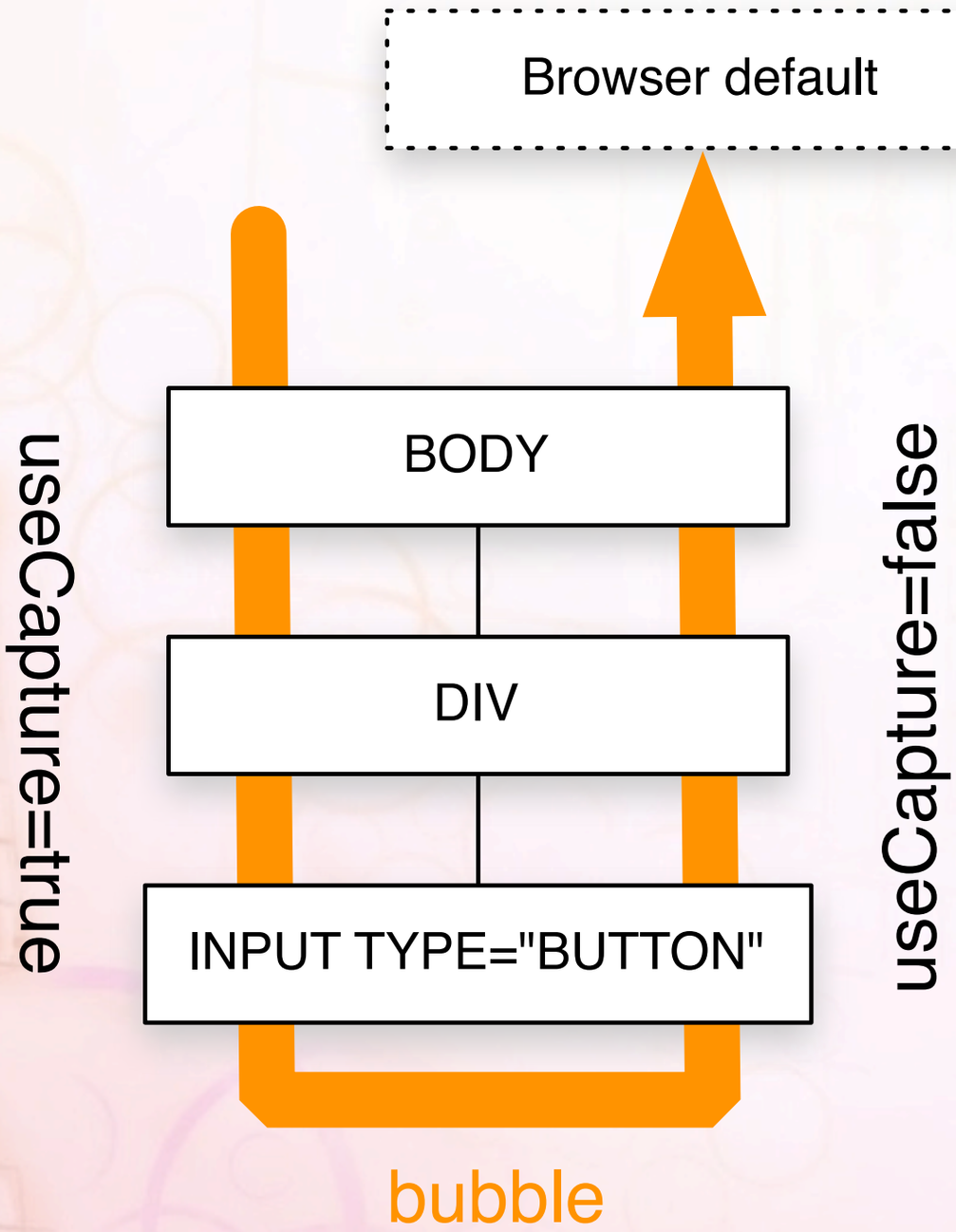
Manipulate the DOM to show results

(Hints on the appropriate jQuery functions on the web page)

Event propagation

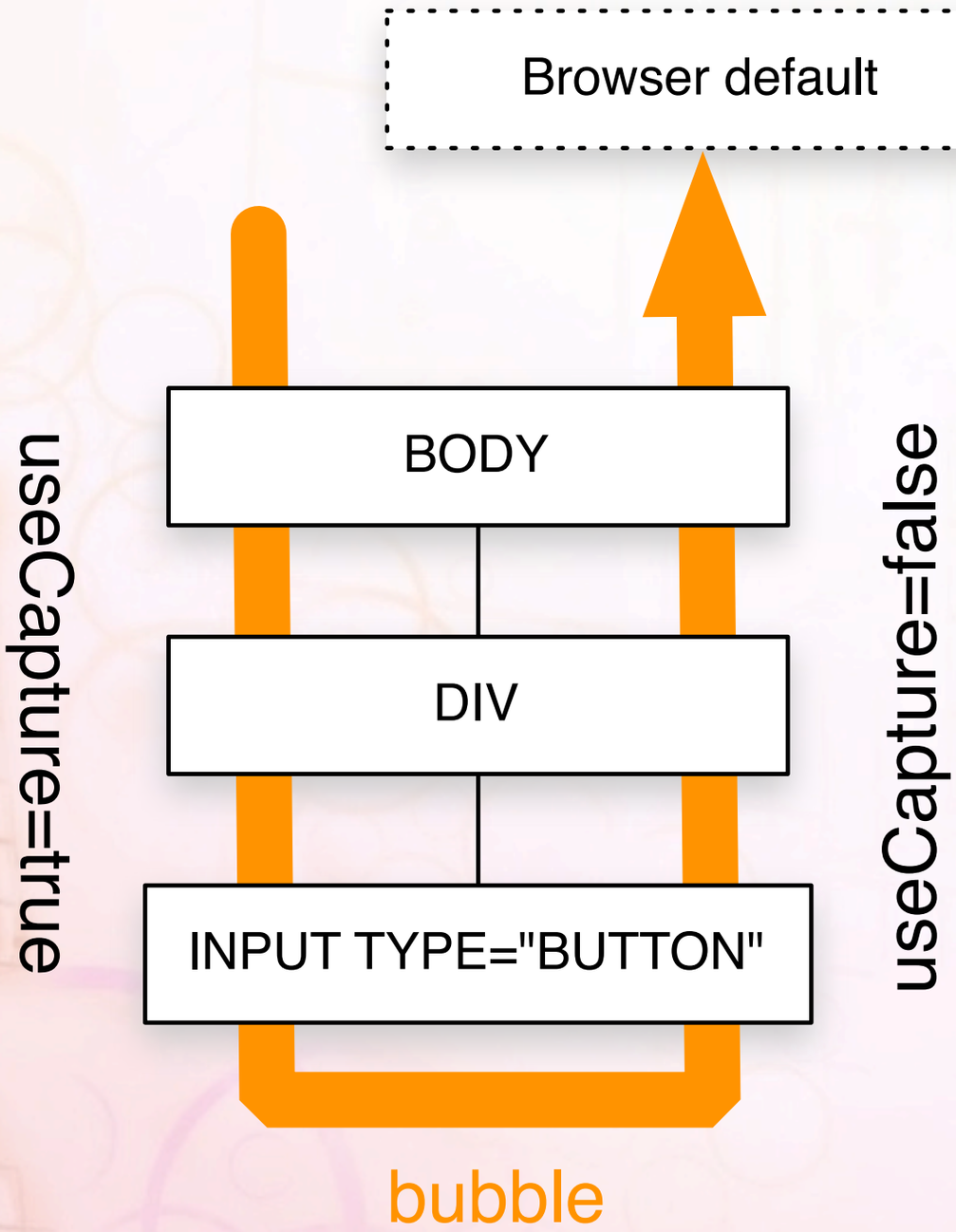


Event propagation




```
event.stopPropagation();
```

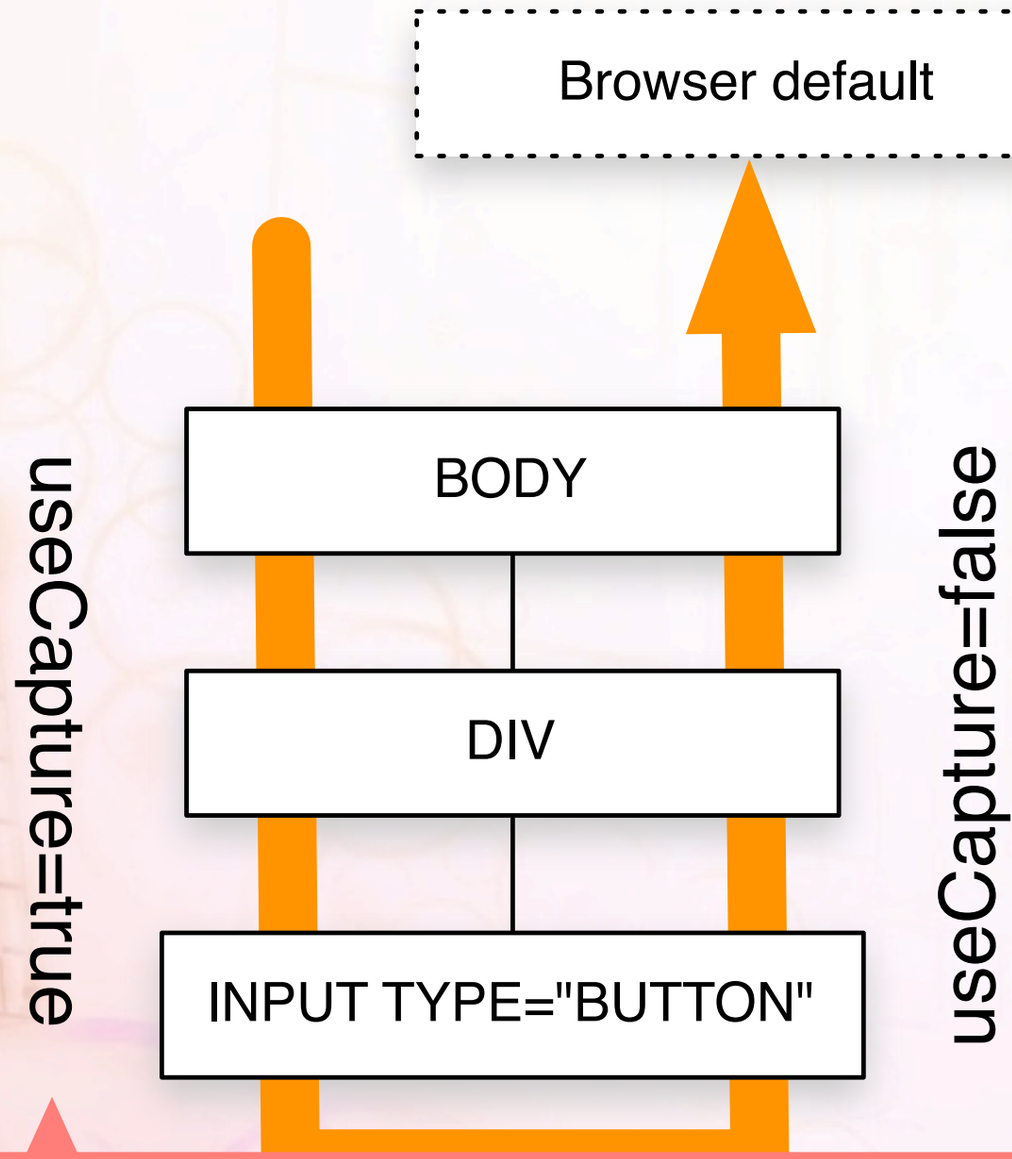
Event propagation




```
event.preventDefault();
```

```
event.stopPropagation();
```

Event propagation




```
event.preventDefault();
```

```
event.stopPropagation();
```

Only with addEventListener(). You rarely need it.

AJAX

"Asynchronous JavaScript and XML"

Just an HTTP request from within JavaScript code

Doesn't need to be XML! Can be HTML, plain text, JSON, ...

Use the result however you want (display as HTML, interpret, modify the DOM...)

Plain AJAX

```
var xhr = new XMLHttpRequest();
xhr.open('GET', 'hello.aspx', true);
xhr.onreadystatechange = function () {
    if (xhr.readyState == 4) {
        if(xhr.status == 200)
            alert(xhr.responseText);
        else
            alert("Error loading page\n");
    }
};
xhr.send(null);
```

AJAX with jQuery

```
$.get("hello.aspx", function(response) {
    alert(response);
});
```

JSON

Just literal JavaScript object notation.

```
JSON.stringify({ foo: 1, bar: 2});
>> '{"foo":1,"bar":2}'
```

And:

```
JSON.parse('{"foo":1,"bar":2}')
>> Object
```

You could `eval()` it! (But you shouldn't)

Same Origin Policy

Requests only allowed to same server as web page was served from.

Only applies to XHR calls, not to external resources included via HTML tags.

JSON-P

JSON with padding, to get around SOP.

```
<script src="http://host/data?
jsonp=myFunction">
```

Server produces the following script:

```
myFunction({ foo: 'bar' });
```

Client installs `myFunction` before calling.

Exercise 7

Retrieving a Flickr feed

Retrieve images from Flickr's public feed and display them in our browser.

Flickr is a different Origin, so we need to use JSON-P.

HTML5

Catch-all name for new browser technologies.

A bunch of them started in Google Gears, got standardized.

It even has a logo!

What's in HTML5

New markup	HTML elements (<nav>, <video>), CSS3
Client-side storage	Offline web applications, LocalStorage, Web SQL, IndexedDB
Graphics	Canvas, WebGL
New APIs	Web Workers, Web Sockets, GeoLocation

Hip HTML5 Demos

Canvas&Video support
http://alexw.me/playground/canvas.html

WebGL (requires FF4 or Chrome)
http://chrysaora.com/

Geolocation
http://html5demos.com/geo

Offline apps
http://striplijst.rix0r.nl/

Wrap-up

Summary

Small & dynamic language with a lot of flexibility

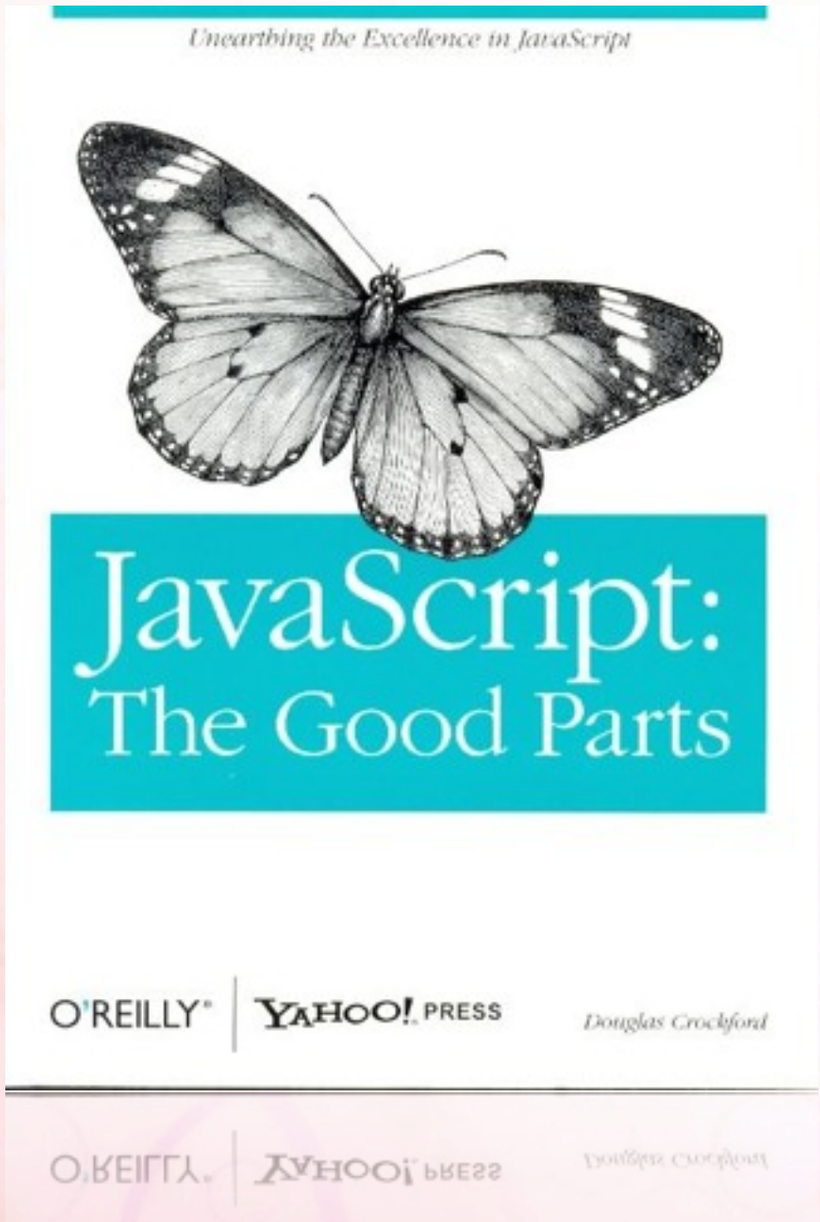
Runs in surprisingly many places

Ideally suited for asynchronous programming

Don’t go into browserland without a library!

Browsers can do pretty pimpy stuff these days.

More information

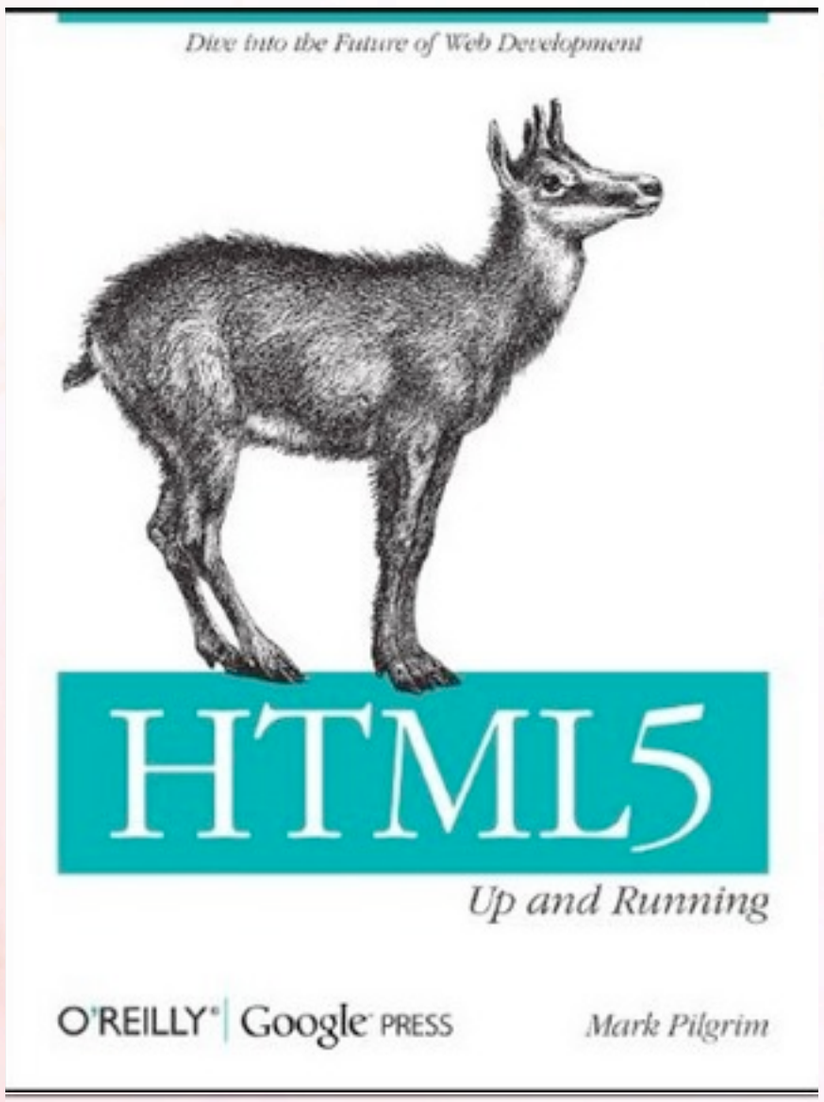


JavaScript, the Good Parts

Douglas Crockford

(Also YUI Theater)

HTML5



Dive into HTML5

Mark Pilgrim

http://diveintohtml5.org/

Mozilla Developer Network* (MDN)

Complete JavaScript reference in a nicely indexed website

http://developer.mozilla.org/

(In the process from being renamed from MDC)*

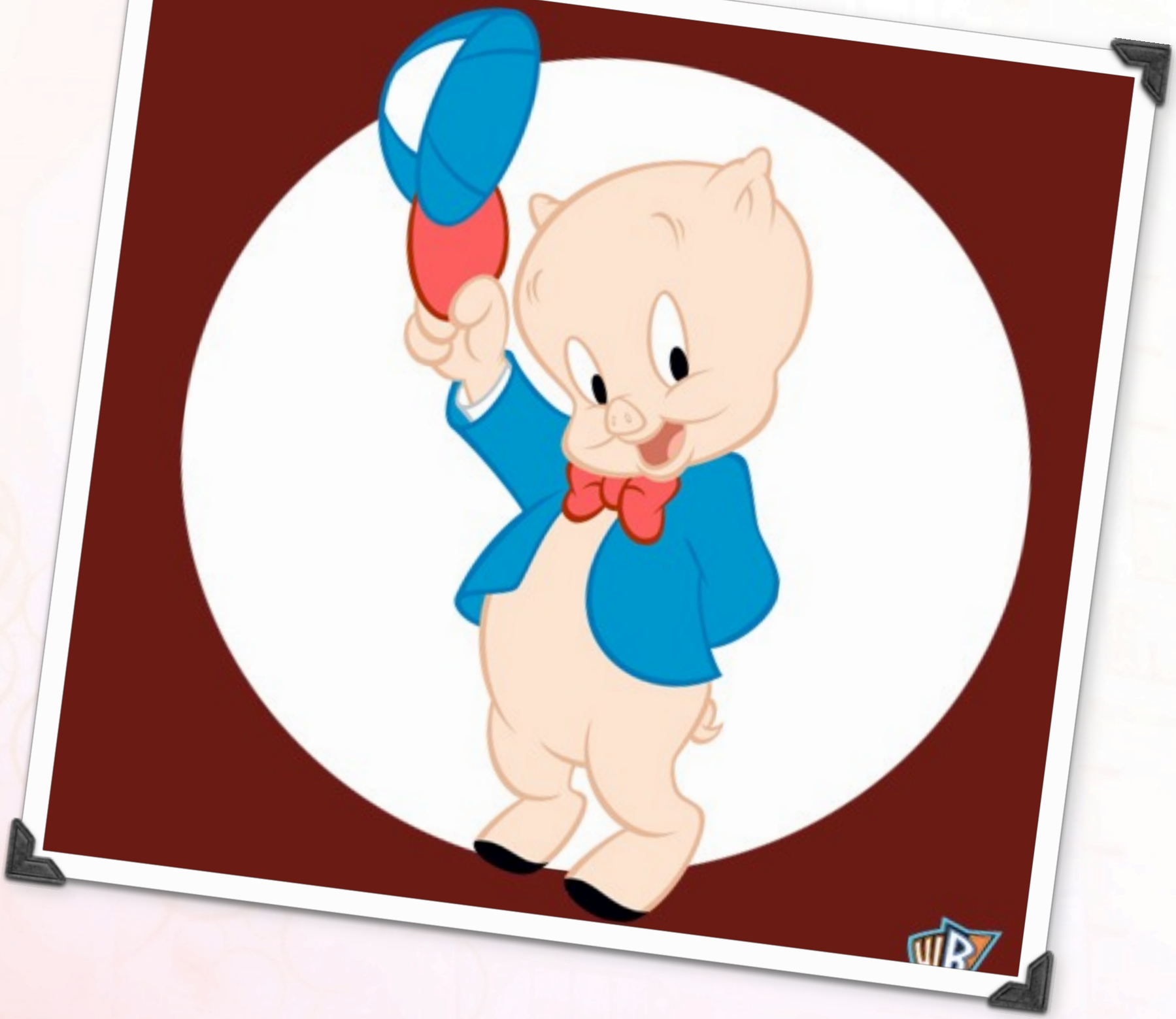
Maybe?

Essential JavaScript & jQuery Design Patterns

Freely available on the web

http://addyosmani.com/

Didn't get a chance to read it yet.



Thanks for your attention!