JavaScript Needn't Hurt!

Thomas Kjeldahl Nilsson
thomas@kjeldahlnilsson.net
linkedin.com/in/thomaskjeldahlnilsson
twitter.com/thomanil

Intro

Who Are You?

Done any JavaScript?
Dozen lines of code?
Hundreds?
Thousands?

Who Am I?

JavaScript enthusiast
Hundreds of hours
Thousands lines of code
Not an expert!

What Are We Covering Today?

Language basics
Html scripting
Good practices
Tools

Practical Stuff

History Lesson

Brendan Eich

ECMAScript

A Note on ActionScript

What Rocks?

Powerful, elegant, dynamic

Present & enabled for most users

Not confined to the browser

Small learning surface

What Sucks?

Rushed out the door
Some bad language features
Crossbrowser problems

Development Environment

Walkthrough

Language Basics

Syntax

Types

Variables

Objects

Functions

Basic Syntax

Similar to Java, C#

Operators, statements

if-else, while, for, try-catch-finally

Still in Kansas...

Types

Strings
Numbers
Booleans
Objects
Arrays

Variable Declarations

```
var x = “foo”;        // string
var x = 2;            // number
var x = true;         // boolean
var x = { };          // object
var x = [ ];          // array
```

Objects

Object Creation
Literal Form

```
var BasicProject = {

    name : "Steria Workshop",

    version : 1.2,

    getName : function() {
      return this.name;
    }

};
```

Object Creation Dynamic Form

```
var BasicProject = {};

BasicProject.name = "Steria Workshop";
BasicProject.version = 1.2;

BasicProject.getName = function() {
    return this.name;
};
```

Objects as Maps (Associative Arrays)

```
var Person = { firstname:"John", lastname:"Smith" };

Person.firstname;          // => "John" (Access using identifier)
Person["firstname"];       // => "John" (Access using variable name)
```

Arrays are Special Case of Object

```
var arr = [];   // Always declared without size.
            // Grows as needed.
arr[0] = true;
arr[3] = "john";
arr[300] = { description : "object!" };
arr[100];           // => Undefined
arr.length;     // => 301
```

Arrays and Objects Can Be Deeply Nested

```
var OuterObject = {
     innerArray : [
      innerObject : {
             deepestArray : [1,2,3]
      }
     ]
};
```

JSON

```
{"firstName":"Gustav","lastName":"Adolf",
"roles":["King of Sweden","Terrible shipbuilder"],
"other":{"birth":"9.12.1594","death":"16.11.1632"}}
```

Kind Familiar Looking

```
{
    "firstName" : "Gustav",
    "lastName" : "Adolf",
    "roles" : [
            "King of Sweden",
            "Terrible shipbuilder"
     ],
    "other" : {
            "birth" : "9.12.1594",
            "death" : "16.11.1632"
     }
}
```

JavaScript Object Literal!

```
var EvaluatedObject = {
    firstName : "Gustav",
    lastName : "Adolf",
    roles : [
        "King of Sweden",
        "Terrible shipbuilder"
     ],
    other : {
        birth : "9.12.1594",
        death : "16.11.1632"
     }
};
```

Inheritance

Prototypal Inheritance (Simplified)

```
var Employee = {name : "CEO Jack", company : "ACME Inc."};
var Janitor = Object.create(Employee);

// Janitor now looks and behaves just like its prototype, Employee
Janitor.company // =>  "ACME Inc.", falls back to prototype.company

Janitor.name = "Janitor Tim"; // Override name
Janitor.tools = ["broom", "bucket"]; // Define member variable only on child

Employee.name = "CEO Jane"; // Change name of prototype
Janitor.name; // => Still "Janitor Tim". Overriden members unaffected by prototype
    changes
```

Functions

Simple Function Definition

```
function add(a, b) {
    return a + b;
}
```

That's Just a Way of Saying:

```
var add = function(a, b) {
    return a + b;
};

// Use this consistently
// Helps you remember:
// Functions are just variables!
```

An Anonymous Function...

```
function(element) {
    // Do something with element
}
```

...Can Be Sent To Another Function

```
each(collection, function(element) {
    // Do something with current element
});
```

Example: Event Handler

```
button.onClick(function(element) {
    alert(«You clicked me!»);
});
```

Sharp Edges

Global variables

No block scope

Properly Scoped Variable

```
var getSmallNumber = function(){
    var smallNumber = 42; // Note use of var keyword
    return smallNumber;
};
```

Sloppy, Global Variable

```
var getSmallNumber = function(){
    smallNumber = 42;
    return smallNumber;
};

// Missing var prefix = smallNumber gets global scope
// Becomes available for all code
// Sticks around and pollutes namespace
```

No Block Scope

```
var x = 1;

if (true) {
var x = 123;
}

// x => 123
```

Semicolon insertion

Don't force the browser to guess!

Example

```
a = b + c
(d + e).print()
```

becomes...

```
a = b + c(d + e).print();
```

== vs ===

Quiz

```
'' == '0'              // true or false?
0 == ''                // true or false?
0 == '0'               // true or false?
false == 'false'       // true or false?
false == '0'           // true or false?
false == undefined     // true or false?
false == null          // true or false?
null == undefined      // true or false?
' \t\r\n ' == 0        // true or false?
```

How Many Did You Get?

```
'' == '0'               // false
0 == ''                 // true
0 == '0'                // true
false == 'false'        // false
false == '0'            // true
false == undefined      // false
false == null           // false
null == undefined       // true
' \t\r\n ' == 0         // true

// Why? Type coercion on right operand, that's why.
```

Instead, Use === (And !==)

```
'' === '0'              // false
0 === ''                // false
0 === '0'               // false
false === 'false'       // false
false === '0'           // false
false === undefined     // false
false === null          // false
null === undefined      // false
' \t\r\n ' === 0        // false
```

Advanced Stuff

Closures
Modules
Memoization

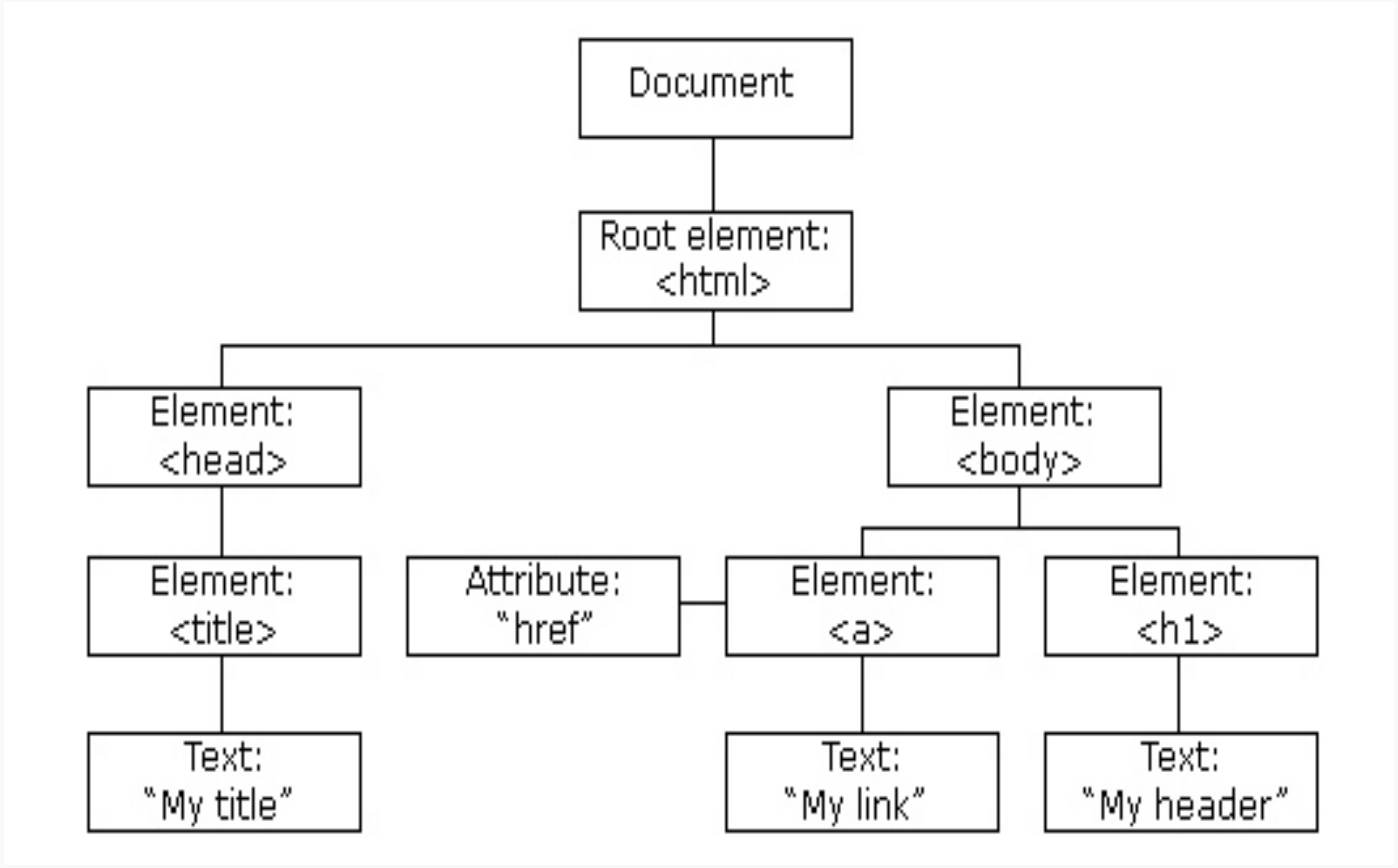
Clientside

Firebug
jQuery

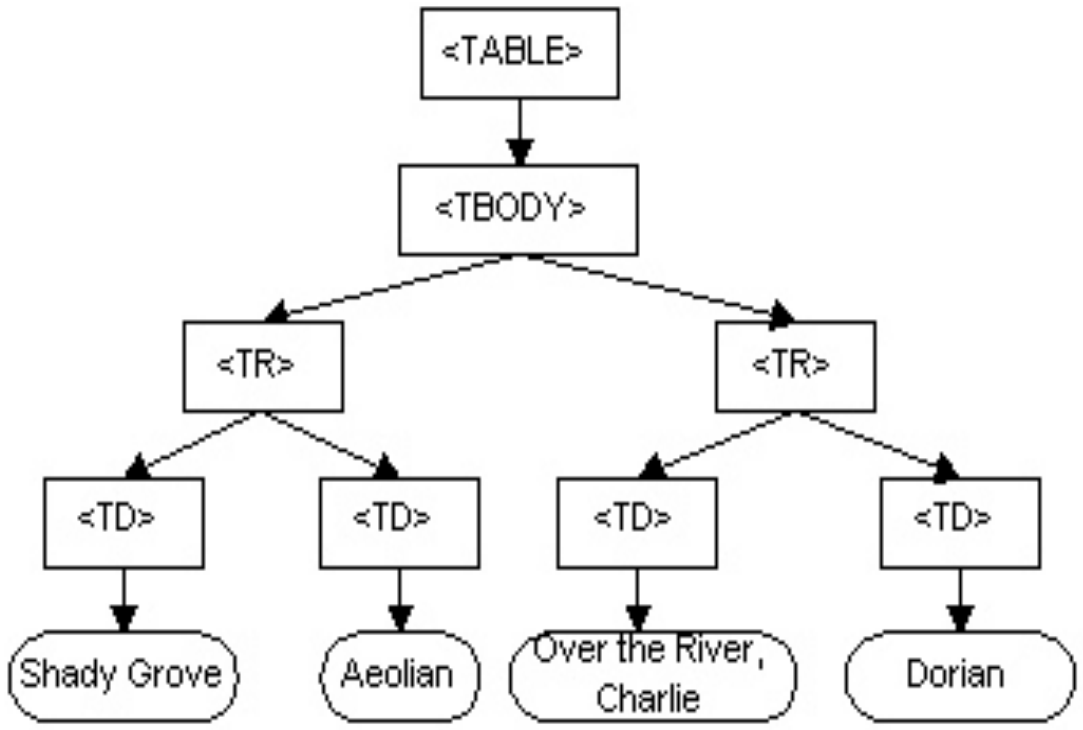
Firebug

Demo

The DOM


Document
Root element: <html>
Element: <head>
Element: <body>
Element: <title>
Attribute: "href"
Element: <a>
Element: <h1>
Text: "My title"
Text: "My link"
Text: "My header"

```
<TABLE>
      <TBODY>
            <TR>
                  <TD>Shady Grove</TD>
                  <TD>Aeolian</TD>
            </TR>
            <TR>
                  <TD>Over the River, Charlie</TD>
                  <TD>Dorian</TD>
            </TR>
      </TBODY>
</TABLE>
```



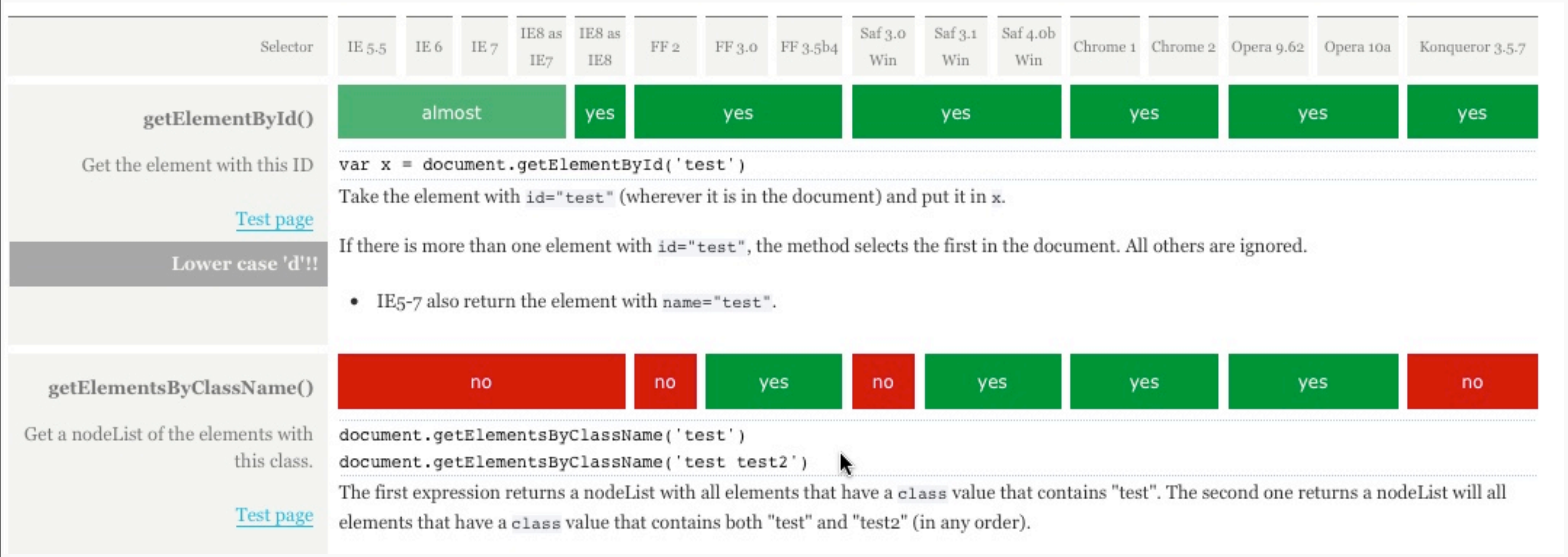
graphical representation of the DOM of the example table

DOM Scripting Basics

```
x.getElementById(id) ;                    // get the element with a specified id

x.getElementsByTagName(name);    // get all elements with a
                                                     // specified tag name

x.appendChild(node);                      // insert a child node to x
x.removeChild(node),                      // remove a child node from x

x.innerHTML = «<p>New text</p>»;   // fill element with html or text
```

Live Example

DOM Api: Crossbrowser Headache



Selector	IE 5.5	IE 6	IE 7	IE8 as IE7	IE8 as IE8	FF 2	FF 3.0	FF 3.5b4	Saf 3.0 Win	Saf 3.1 Win	Saf 4.0b Win	Chrome 1	Chrome 2	Opera 9.62	Opera 10a	Konqueror 3.5.7
getElementById()	almost	almost	almost	almost	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
getElementsByClassName()	no	no	no	no	no	no	yes	yes	no	yes	yes	yes	yes	yes	yes	no

getElementById()

Get the element with this ID

Test page

Lower case 'd'!!

```
var x = document.getElementById('test')
```

Take the element with `id="test"` (wherever it is in the document) and put it in `x`.

If there is more than one element with `id="test"`, the method selects the first in the document. All others are ignored.

- IE5-7 also return the element with `name="test"`.

getElementsByClassName()

Get a nodeList of the elements with this class.

Test page

```
document.getElementsByClassName('test')
document.getElementsByClassName('test test2')
```

The first expression returns a nodeList with all elements that have a `class` value that contains "test". The second one returns a nodeList will all elements that have a `class` value that contains both "test" and "test2" (in any order).

http://www.quirksmode.org

So Use a Framework!

jQuery

Instant Win:

Crossbrowser
Non-verbose
Traverse DOM
Manipulate DOM

Lots of Other Stuff, Too

Server communication
UI widgets and effects
each(), map(), etc
JSON parsing
+++

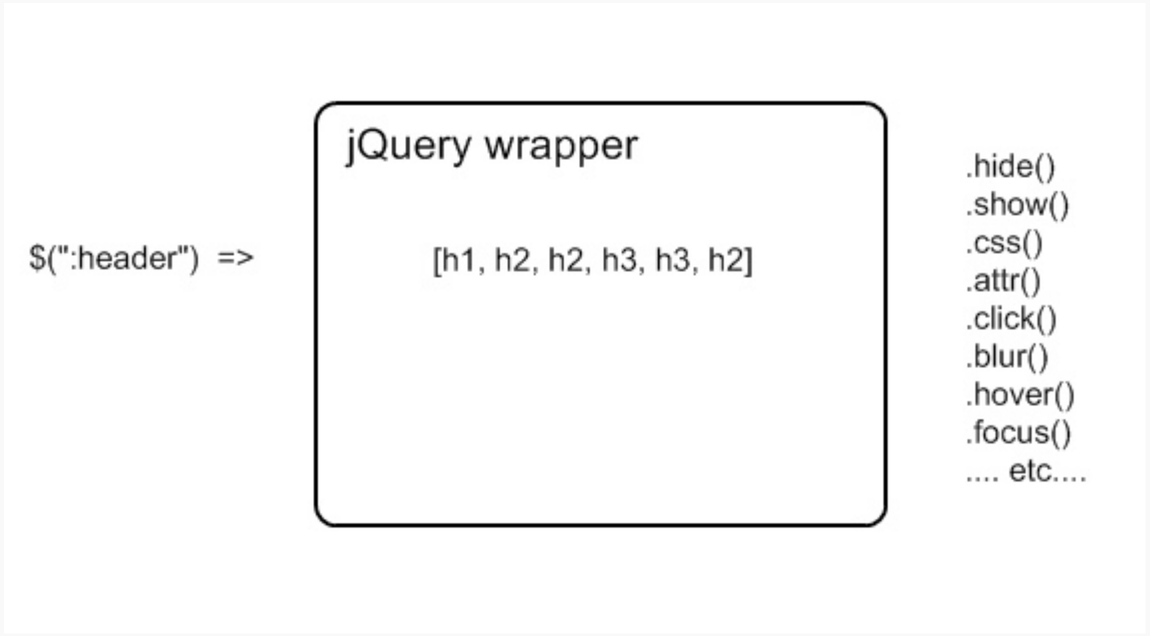
jQuery Function

```
$()           // Short form
jQuery()      // Long form
```

Find Stuff

```
$(“p”);                    // Find all paragraphs
$(“#shoppingList”);        // Find element with id 'shoppingList'
$(“.shoppingListItem”);    // Find elements with class 'shoppingListItem'
$(“:text”);                // Find all text elements
$(“:visible”);             // Find only visible elements
```

$() Wraps and Returns Matching Element(s)



Manipulate Matched DOM Elements

```
$(“p”).css(”color”, ”green”);    // Set color on all paragraph elements
$(“p”).hide();                   // Hide all paragraphs

// Make all input buttons react to mouse clicks
$(“input”).click(function(){ alert(”You clicked this button!”); });
```

Chaining

Every API call returns jQuery object
So we can chain function calls together
"Fluent API"

Chaining Example

```
// Set color on all paragraph elements, then hide them all
$(“p”).css(”color”, ”green”).hide();
```

Live Example

Prepared Example

Caution!

Use frameworks as needed

But don't depend on them!

JavaScript != jQuery

Good Practices

jsLint
Automated testing
Unobtrusive JavaScript

JsLint

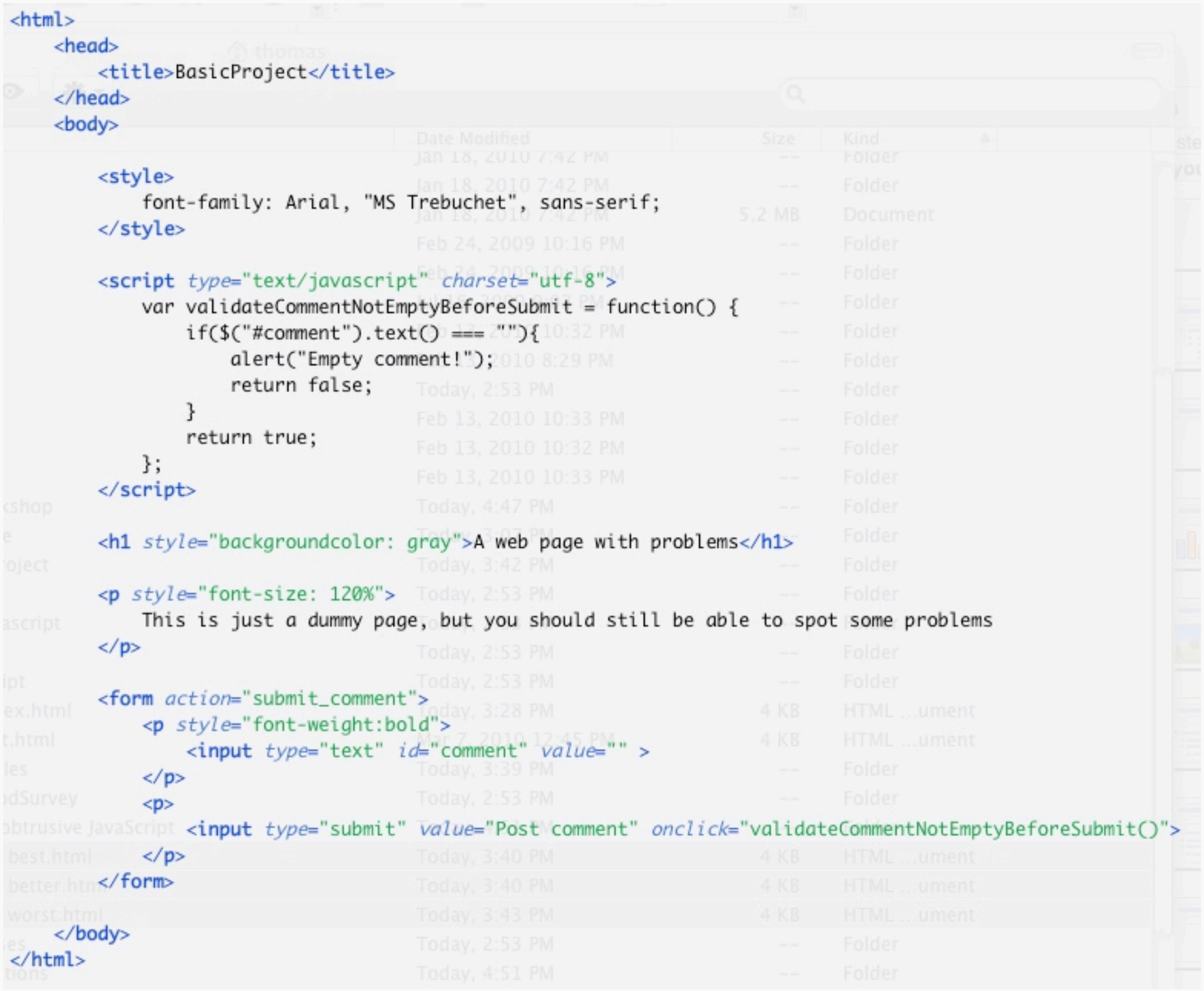
Demo

Automated Testing

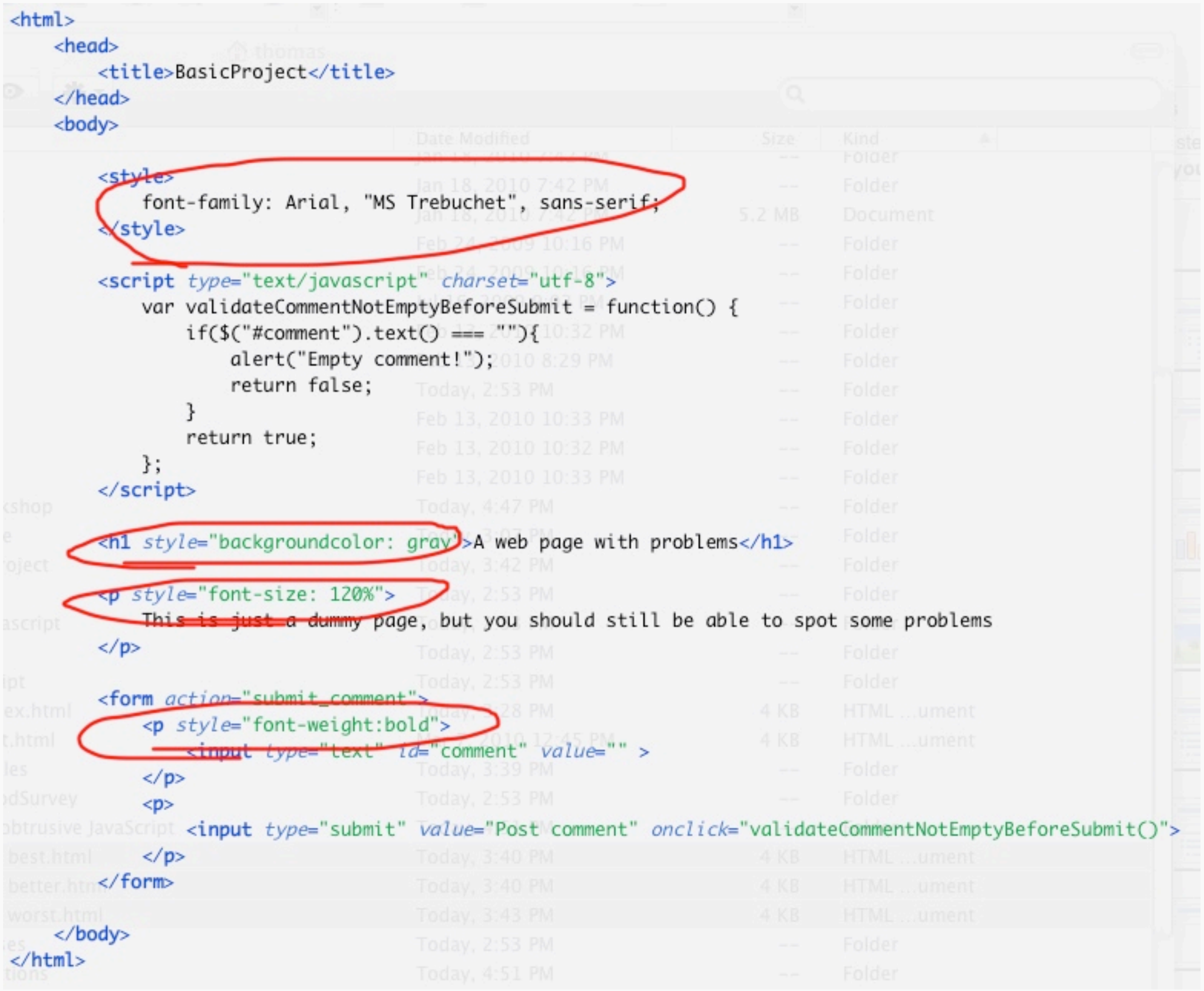
YUI Test demo

Unobtrusive JavaScript

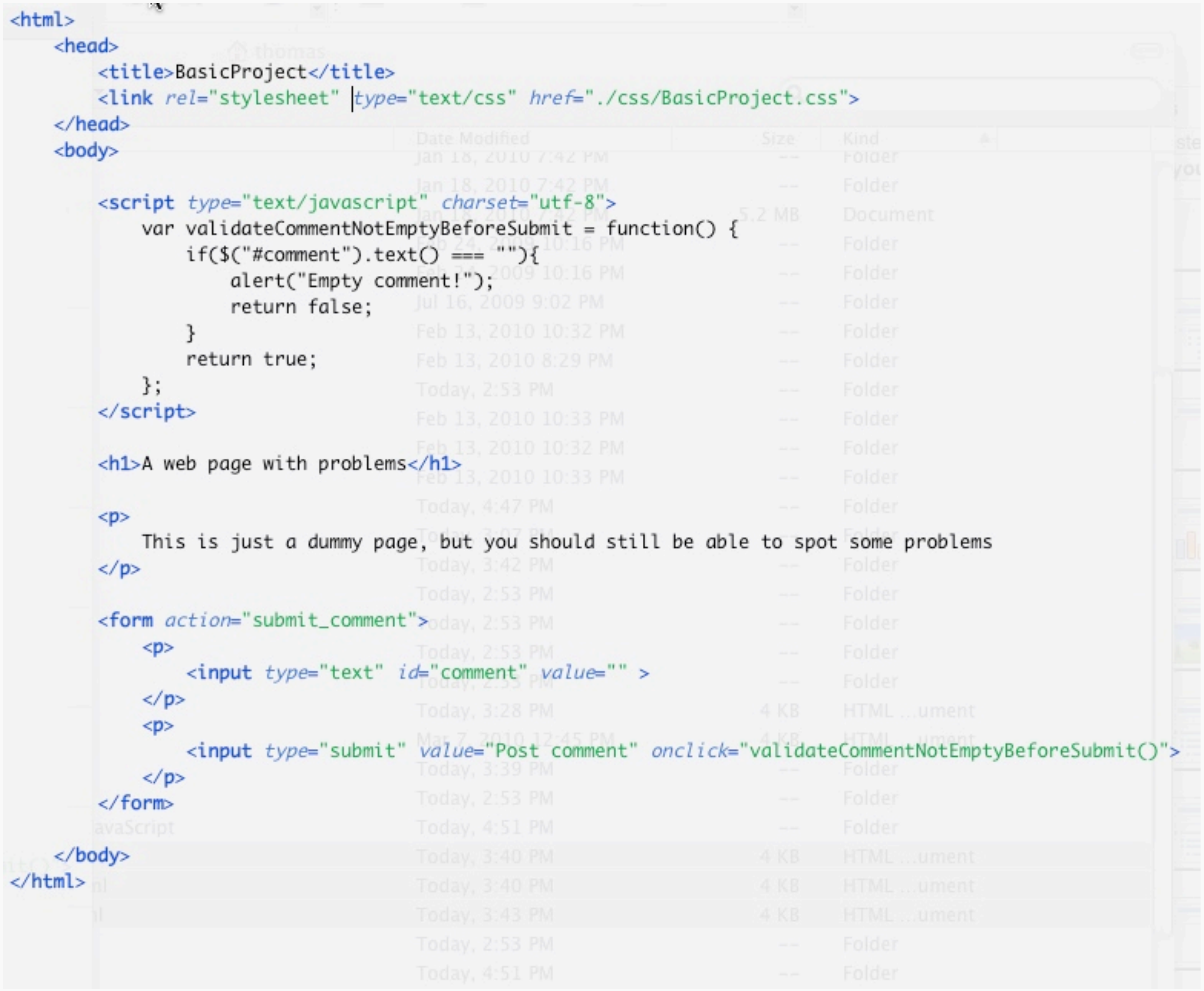
Structure vs behavior
Separate js files
Event handler setup
Namespaces
Universal Access



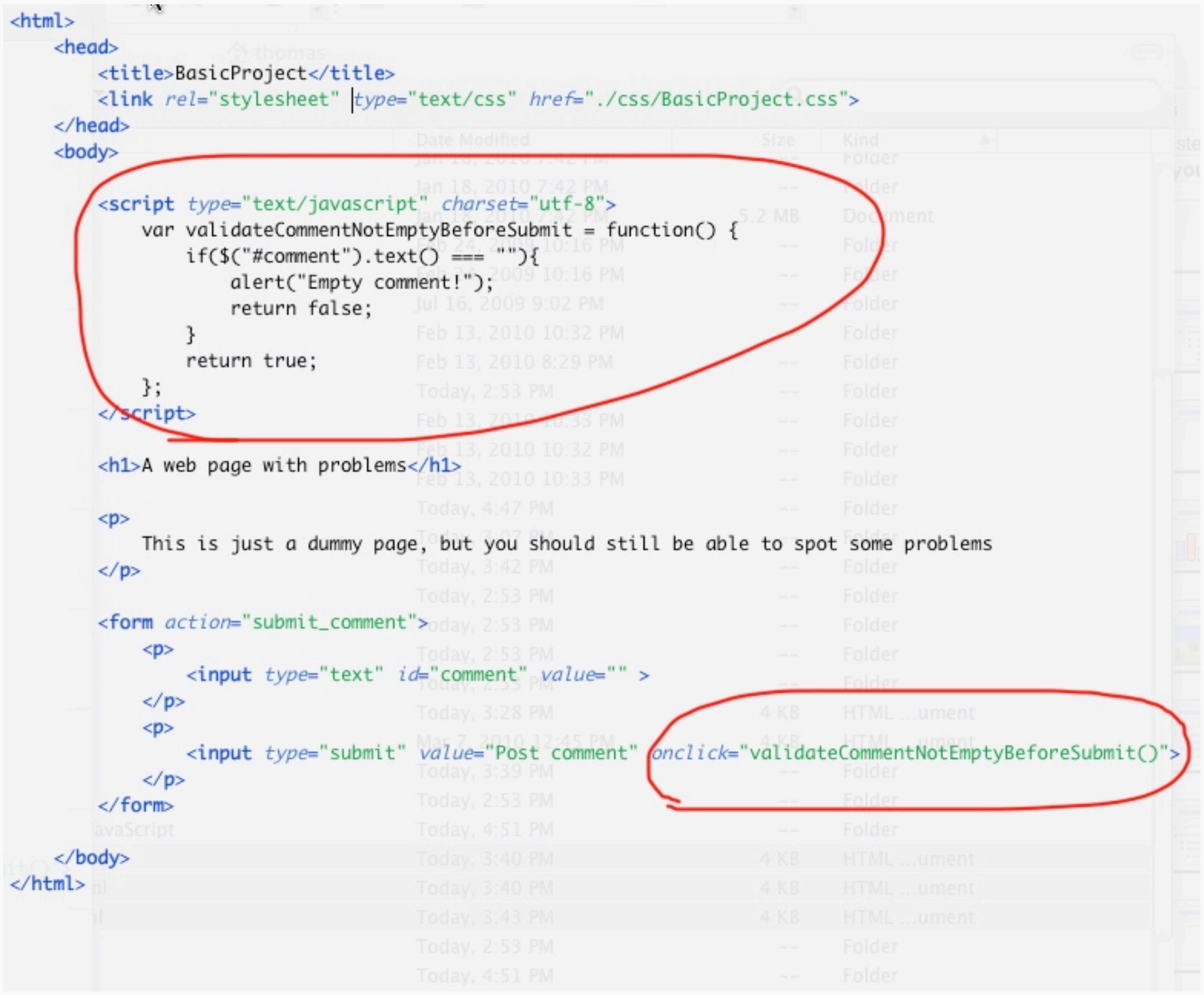
```
<html>
    <head>
        <title>BasicProject</title>
    </head>
    <body>

        <style>
            font-family: Arial, "MS Trebuchet", sans-serif;
        </style>

        <script type="text/javascript" charset="utf-8">
            var validateCommentNotEmptyBeforeSubmit = function() {
                if($("#comment").text() === ""){
                    alert("Empty comment!");
                    return false;
                }
                return true;
            };
        </script>

        <h1 style="backgroundcolor: gray">A web page with problems</h1>

        <p style="font-size: 120%">
            This is just a dummy page, but you should still be able to spot some problems
        </p>

        <form action="submit_comment">
            <p style="font-weight:bold">
                <input type="text" id="comment" value="" >
            </p>
            <p>
                <input type="submit" value="Post comment" onclick="validateCommentNotEmptyBeforeSubmit()">
            </p>
        </form>

    </body>
</html>
```



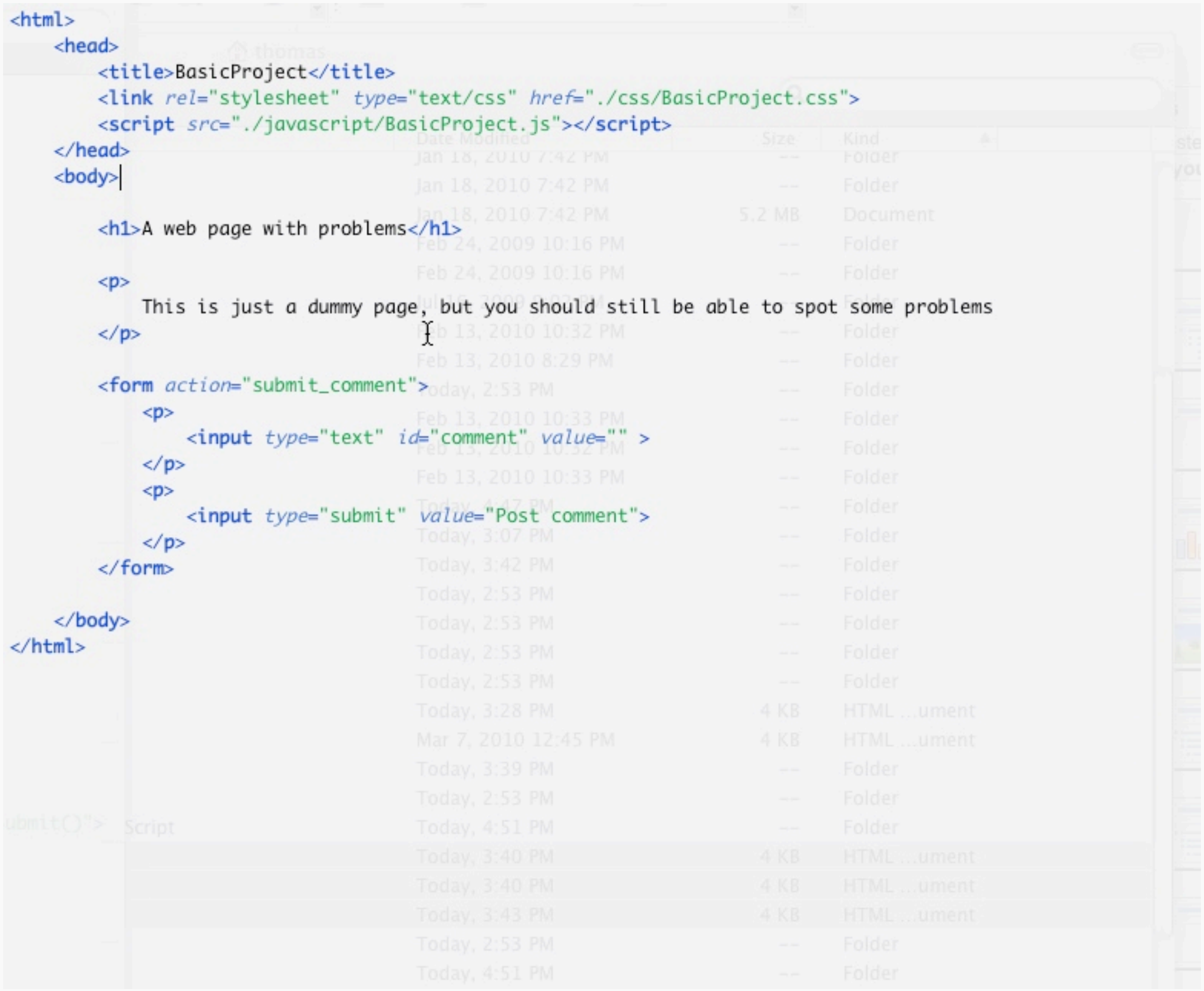
```
<html>
    <head>
        <title>BasicProject</title>
    </head>
    <body>

        <style>
            font-family: Arial, "MS Trebuchet", sans-serif;
        </style>

        <script type="text/javascript" charset="utf-8">
            var validateCommentNotEmptyBeforeSubmit = function() {
                if($("#comment").text() === ""){
                    alert("Empty comment!");
                    return false;
                }
                return true;
            };
        </script>

        <h1 style="backgroundcolor: gray">A web page with problems</h1>

        <p style="font-size: 120%">
            ~~This is just a~~ dummy page, but you should still be able to spot some problems
        </p>

        <form action="submit_comment">
            <p style="font-weight:bold">
                ~~<input type="text"~~ id="comment" value="" >
            </p>
            <p>
                <input type="submit" value="Post comment" onclick="validateCommentNotEmptyBeforeSubmit()">
            </p>
        </form>

    </body>
</html>
```



```
<html>
    <head>
        <title>BasicProject</title>
        <link rel="stylesheet" type="text/css" href="./css/BasicProject.css">
    </head>
    <body>

        <script type="text/javascript" charset="utf-8">
            var validateCommentNotEmptyBeforeSubmit = function() {
                if($("#comment").text() === ""){
                    alert("Empty comment!");
                    return false;
                }
                return true;
            };
        </script>

        <h1>A web page with problems</h1>

        <p>
            This is just a dummy page, but you should still be able to spot some problems
        </p>

        <form action="submit_comment">
            <p>
                <input type="text" id="comment" value="" >
            </p>
            <p>
                <input type="submit" value="Post comment" onclick="validateCommentNotEmptyBeforeSubmit()">
            </p>
        </form>

    </body>
</html>
```



```
<html>
    <head>
        <title>BasicProject</title>
        <link rel="stylesheet" type="text/css" href="./css/BasicProject.css">
    </head>
    <body>

        <script type="text/javascript" charset="utf-8">
            var validateCommentNotEmptyBeforeSubmit = function() {
                if($("#comment").text() === ""){
                    alert("Empty comment!");
                    return false;
                }
                return true;
            };
        </script>

        <h1>A web page with problems</h1>

        <p>
            This is just a dummy page, but you should still be able to spot some problems
        </p>

        <form action="submit_comment">
            <p>
                <input type="text" id="comment" value="" >
            </p>
            <p>
                <input type="submit" value="Post comment" onclick="validateCommentNotEmptyBeforeSubmit()">
            </p>
        </form>

    </body>
</html>
```



```
<html>
    <head>
        <title>BasicProject</title>
        <link rel="stylesheet" type="text/css" href="./css/BasicProject.css">
        <script src="./javascript/BasicProject.js"></script>
    </head>
    <body>

        <h1>A web page with problems</h1>

        <p>
            This is just a dummy page, but you should still be able to spot some problems
        </p>

        <form action="submit_comment">
            <p>
                <input type="text" id="comment" value="" >
            </p>
            <p>
                <input type="submit" value="Post comment">
            </p>
        </form>

    </body>
</html>
```

Namespace Hygiene

All your code within single object

Universal Access

Can all your users use your site?

Users without JS?

Blind users with screen readers?

Crossbrowser Dev Process

Frameworks > raw DOM

Test early, test often

Clean, disciplined code

Let’s Code!

Exercises

Wrap-up

What Did We Cover Today?

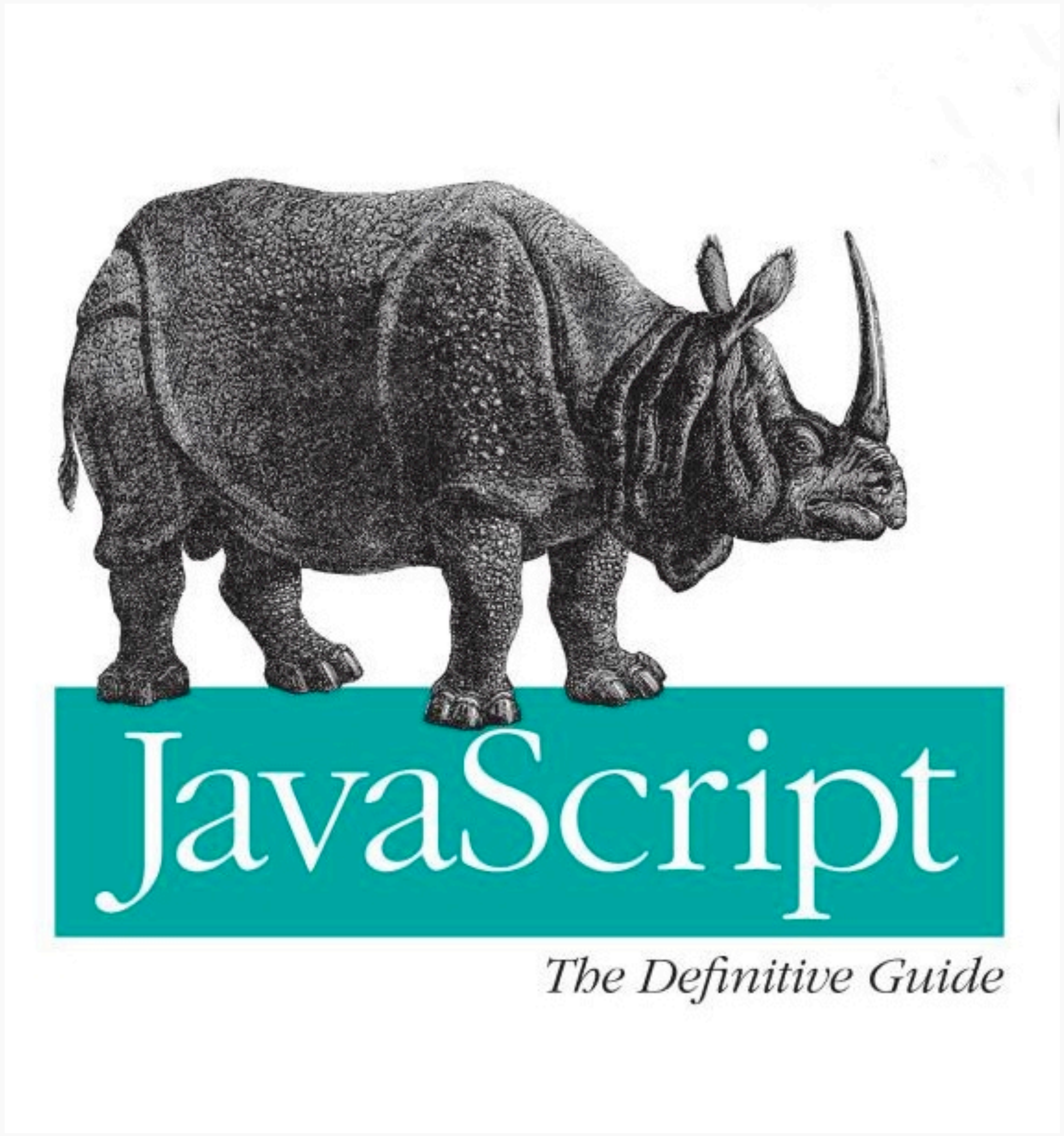
Language basics
Html scripting
Good practices
Tools

What´s Missing?

Server Communication
Performance
Security
Practice practice practice!

References & Further Studies

JavaScript:
The Good Parts


JavaScript
The Definitive Guide

Web Resources

http://javascript.crockford.com

http://cjohansen.no/javascript

http://developer.yahoo.com/yui/theater

http://ajaxian.com

Best Way to Learn:

Start Building!

http://ajaxian.com/archives/writing-a-javascript-tetris-lessons-learned-from-a-ruby-chap

Download This Workshop

http://kjeldahlnilsson.net/jsnh.zip

Slides, lecture notes, exercises
Creative Commons license
Use and distribute freely...

Q&A
Discussion

Contact Info

thomas@kjeldahlnilsson.net
linkedin.com/in/thomaskjeldahlnilsson
twitter.com/thomanil