JavaScript and DHTML

Michał Jasiński

Poznan University of Technology
Institute of Computing Science

Plan of the Lecture

Introduction

What is JavaScript?

JavaScript? What for?!!!

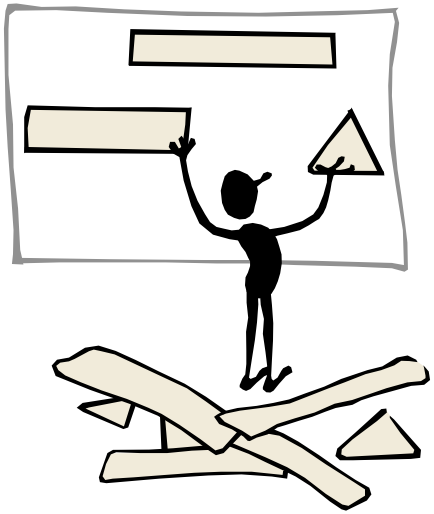
Language basics

Scripting

Document Object Model (DOM)

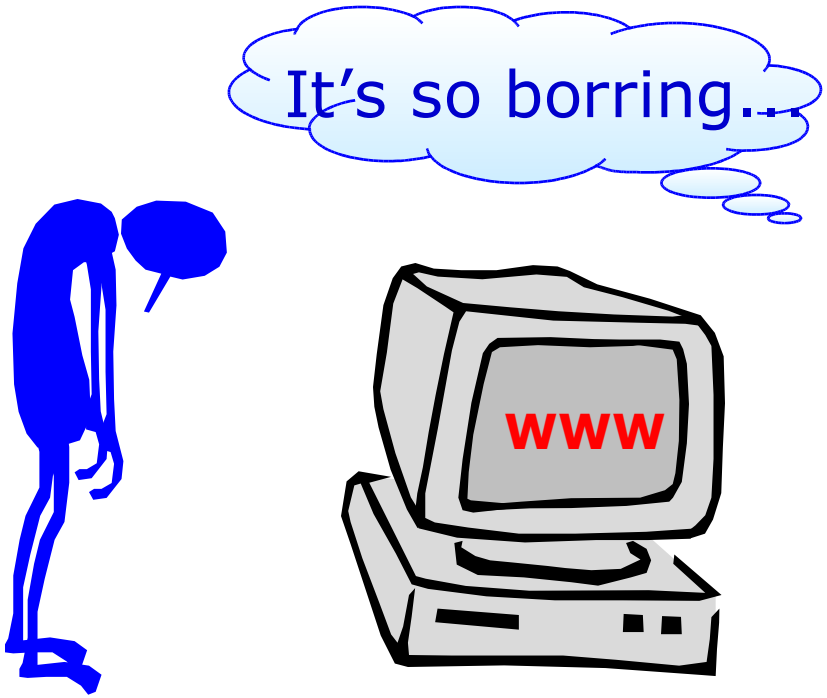
Events

Conclusions



Introduction

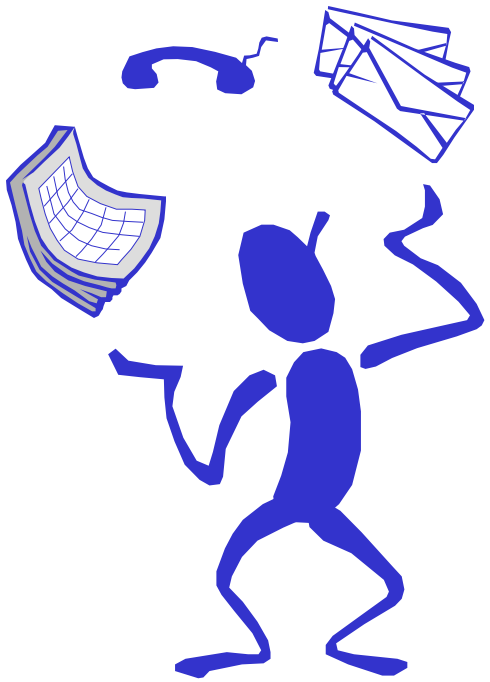
Long, long time ago...



- HTML is static
- HTML is boring
- HTML is not interactive
- HTML is not attractive
- HTML is not enough

Introduction

Looking for the summer



- CGI?
  Too "heavy"
- Helpers and plugins?
  Uncomfortable
- Java applets?
  Too complicated

Something flexible and functional?

Introduction

HIStory

- Early 1990s – Netscape Communications (Netscape Navigator + LiveScript) in early 1990's
- 1995 – Netscape Communications and Sun Microsystems:

  Java + LiveScript = JavaScript 1.0
- Micro and Soft (MS) Evolution

  JScript -> VBScript -> JavaScript

Plan of the Lecture

Introduction

What is JavaScript?

JavaScript? What for?

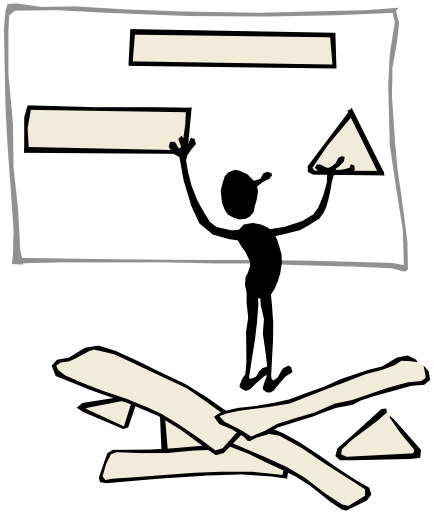
Language basics

Scripting

Document Object Model (DOM)

Events

Conclusions



What is JavaScript?

JavaScript vs. Java

JavaScript	Java
Syntax very similar to C/C++	Syntax very similar to C/C++
Object-*based*	Object-*oriented*
Code accessed from, and embedded in, HTML page	Code accessed from, but not embedded in, HTML page
Interpreted and executed on client machine	Compiled on server, then executed on client
Loose typing	Strong typing
Descends from a line of compact scripting languages –such as HyperTalk and dBASE	Descends from a line of extensive object-oriented languages –such as C++ and SmallTalk

What is JavaScript?

Object what?

- JavaScript is object-based

  Object based uses objects to fulfill a programmatic need

- Java is object-oriented

  Object oriented builds objects using existing frameworks and utilizes existing software functionality.

What is JavaScript?

Interpreted vs. compiled

- Interpreted languages are analyzed, optimized and executed at run-time.
- Compiled languages are analyzed, optimized and saved as executable files. They are executed at run-time.
- Most scripting languages are interpreted.
- Most software programs are compiled.

What is JavaScript?

Typing

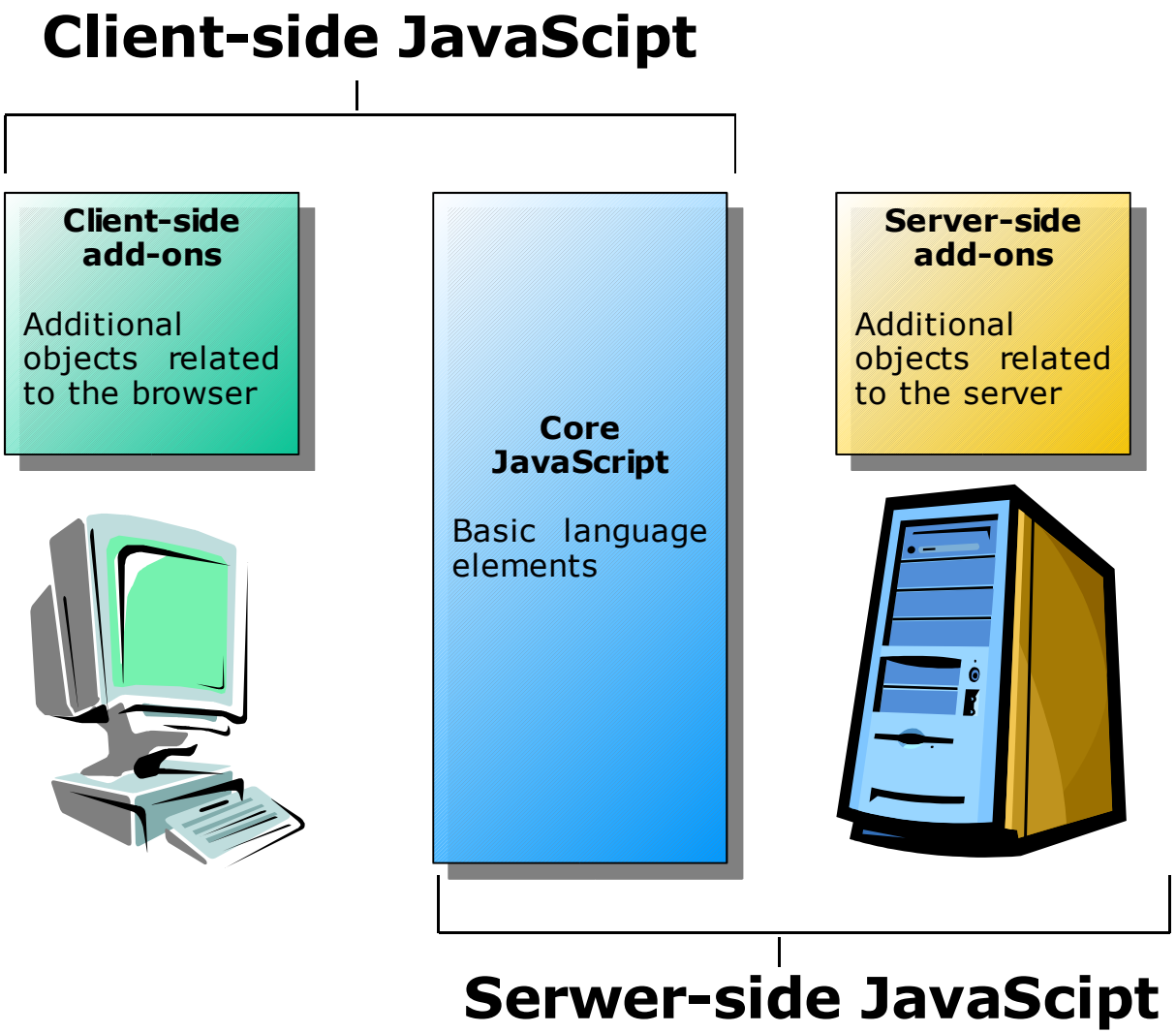
- A data type is a structure for storing a value
  - Integers, floating point, Currency, character strings, arrays, etc…
  - Complex structures, pictures, audio, video, etc...
- Loose Typing
  - Not enforcing the interaction between types
  - being **loose** about what value you can store in what data type
- Strong Typing
  - type checking during interactions is enforced
  - being **strong** in enforcing each value is stored in it's "correct" data type

What is JavaScript?

Event-driven programming

- In an event-driven application, execution does not follow a predetermined path. Instead, different code sections run in response to events. Events can be triggered by the user's actions, by messages from the system or other applications, or from inside the application itself. The sequence of events determines the sequence in which the code runs. Therefore, the path through the application's code can differ each time the program runs.
- An essential part of event-driven programming is writing code that responds to all the possible events that may occur in an application.

What is JavaScript?



What is JavaScript?

Version compatibility

- JavaScript 1.0 (NN 2.0, IE 3.0, Opera 3.0)
- JavaScript 1.1 (NN 2.0, IE 4.0, Opera 3.0)
- JavaScript 1.2 (NN 2.0, IE 4.0, Opera 3.0)
- JavaScript 1.3 (NN 4.5/4.7, IE 5.0, Opera 4.0)
- JavaScript 1.4 (n/a)
- JavaScript 1.5 (NN 6.0, IE 5.5)

What is JavaScript?

Security issues

- Remove functionality

  Cannot open remote files or network connections, cannot write to a disk file, and cannot access a user's local environment variables.

- Same origin policy

  References made in scripts can only be to the same server as the script originated.

- Script and code signing

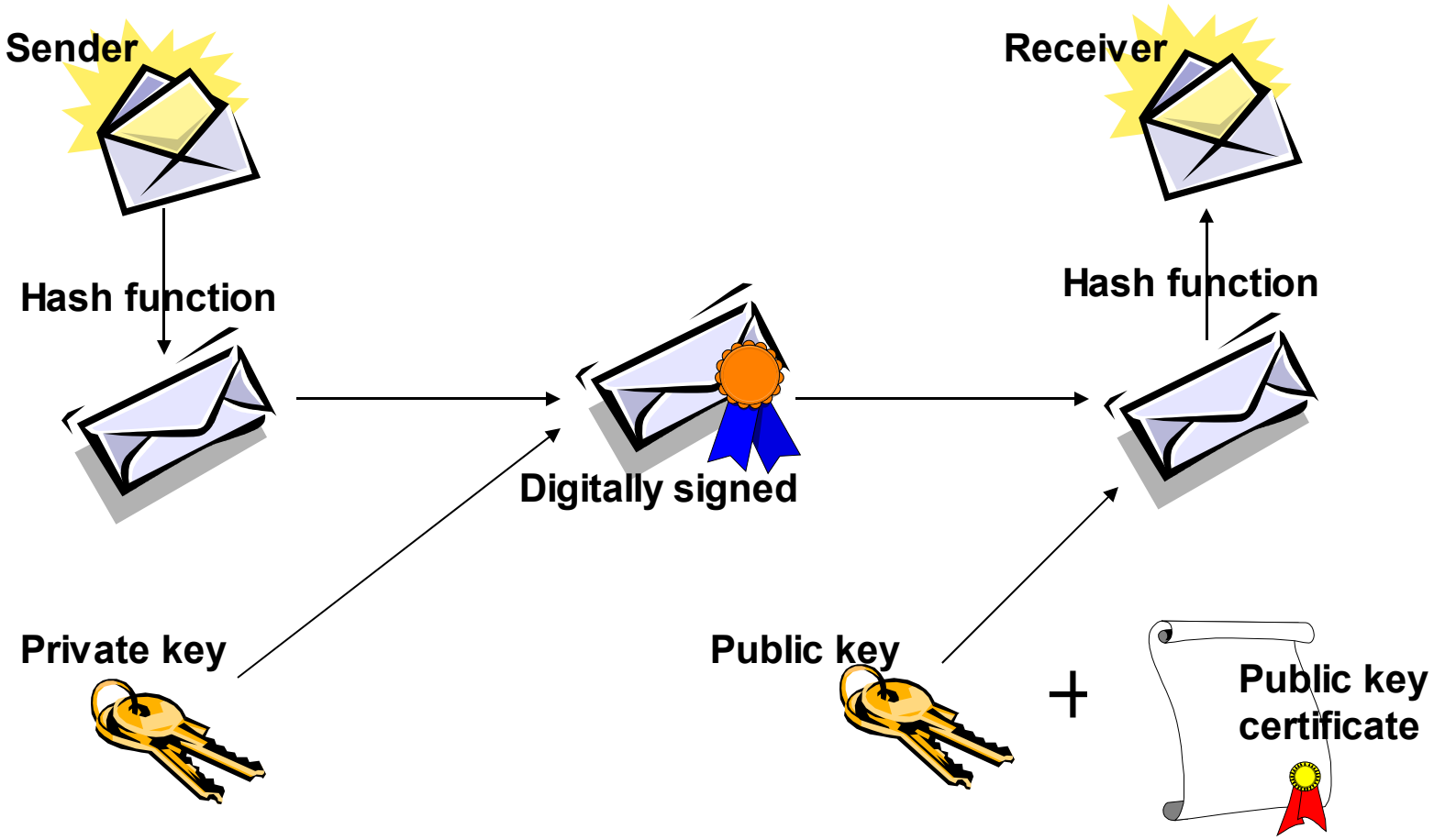
  Certificate technology is predominant form of signing

- Secure servers

  Http and https, it's the **https** hosted pages that provide the security.

What is JavaScript?

Certificates



Plan of the Lecture

Introduction

What is JavaScript?

JavaScript? What for?!!!

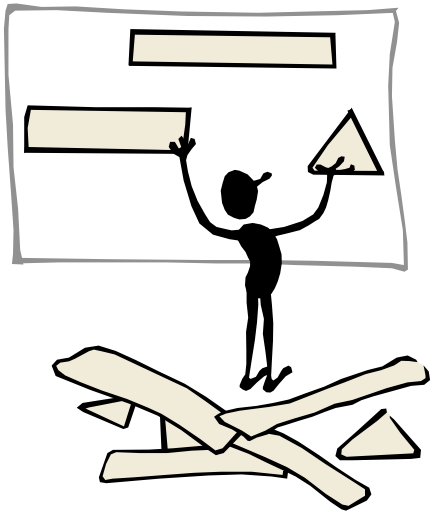
Language basics

Scripting

Document Object Model (DOM)

Events

Conclusions



JavaScript? What for?!!!

DOs and DON'Ts

- JavaScript supports standard programming constructs: values, names, literals, variables, arrays, functions, etc..
- JavaScript supports objects: built-in and user-created.
- JavaScript supports functions.
- JavaScript is an **event-driven** language.

JavaScript? What for?!!!

DOs and DON'Ts

- JavaScript cannot create a standalone executable program; its use is restricted to the Web browser.
- JavaScript cannot open remote files or network connections, cannot write to a disk file, and cannot access a user's local environment variables.
- JavaScript's syntax and runtime error-checking abilities are very limited.
- JavaScript has no integrated IDE

JavaScript? What for?!!!

Advantages and disadvantages

- Lightweight, flexible technology
- Users interaction with HTML pages
- Dynamic content
- Data validation before submitting it to the internet web server
- Managing multi-frame sites
- Visual feedback
- Data handling and formatting
- Interacting with the DOM

JavaScript? What for?!!!

Advantages and disadvantages

- Simplicity eliminates complex solutions
- Difficulty of protecting scripts source code
- Complex scripts increase webservice size
- Difficult crossbrowsing
- No adequate IDEs

JavaScript? What for?!!!

Alternatives

- JScript, VBScript
- ActiveX
- Java applets
- .NET

JavaScript? What for?!!!

IDEs

- Script libraries (i.e. AAScripter)
- Development environments (i.e. Antechinus JavaScript Editor)

Plan of the Lecture

Introduction

What is JavaScript?

JavaScript? What for?!!!

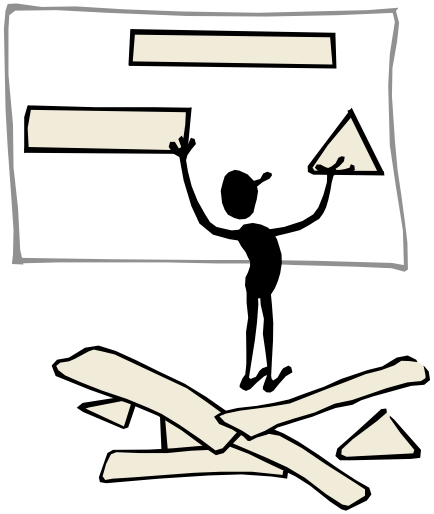
Language basics

Scripting

Document Object Model (DOM)

Events

Conclusions



Language basics

Syntax

- Case-sensitive
- Optional semicollon
- Comments

```
color = 'niebieski';
Color = biały;
x = 1; y = 2;
```

```
x = 1;
y = 2;
```

```
x = 1
y = 2
```

```
Password = 'blablabla';     // Line comment
/* Block comment */
```

Language basics

Data types

- Numbers (42, 3.14)
- Strings (‘Hello!’)
- Logloca (Boolean) (true, false)
- Null
- Undefined

JavaScript is a dynamically typed language.

Automatic data conversion.

Language basics

Numbers

```
i = -10;            // An integer number
pi = 3.14159;       // A real number
number = 5.45E-7; // A real number
oct = 047;          // An integer number (octal)
hex = 0x9f;// An integer number (hex)
```

Language basics

Strings

- `empty_string = “”;`
- `name = “Jan”;`
- `name = ‘Jan’;`
- `surname = ‘Kowalski’;`
- `price = ‘149.99’;`
- `two_lines = ‘one line \n second line’;`
- Single quotations vs. double quotations
- Special characters
- `\n`
- `\t`
- `\\`

Language basics

Variables

- Variable declaration
  - By assigning a value

    x = 10;
  - With the var keyword

    var x = 10;
- Variable scope (global vs. local)
- A variable or array element that has not been assigned a value has the value **undefined**

Functions

- functions are declared using the function keyword
- functions may take any number of operands, but produce only one output (which is returned using the return keyword)

```
function function_Name (operands){
  // commands to be executed
}
```

Language basics

Operators: Arithmetic

- +
- -
- *
- /
- %
- ++
- --
- -

Language basics

Operators: Assignement

- =
- =-
- =+
- =*
- =/
- =%

Language basics

Operators: Comparison

- ==
- !=
- ===
- !==
- >
- >=
- <
- <=

Language basics

Operators: Logic

- !
- &&
- ||

Language basics

Operators: Special

- + (‘a’ + ‘b’)
- ?: (cond ? val1 : val2)
- delete
- new

Language basics

Instructions: *if...else*

```
if (expression)
statement1
[else
statement2]

if (x == 1)
y = 2;
else
y = 3;
```

Instructions: *switch*

```
switch(expr)
{
  case label1 : instr_block_1;
                break;
  case label2 : instr_block_2;
  [default    : instr_block]
}
```

Instructions: *for*

```
for (initialize; test; increment)
statements;


for (i=0; i<3; i++)
{
  z = a[i] + b[i+1];
}
```

Instructions: *do...while*

```
do
{statements;}
while (condition);


do
{
  document.write(i);
  i += 1;
}
while (i<10);
```

Instructions: *while*

```
while (expression)
{statements;}


while (i < 3)
{
   z += x[i];
   i++;
}
```

Language basics

Instructions: *for...in*

```
for (variable in object)
Iterate through fields
of an object


for (field in my_object)
document.write("name:"+
field "value:"+my_object
[field]);
```

Language basics

Instructions: *with*

```
with(object)
Refer to methods and properties of object
without explicit mention


with(Math)
{
  x = sin(i*Math.PI)
  y = sin(i*Math.PI)
}
```

Language basics

Instructions: Other

- break;
- continue;
- return

Plan of the Lecture

Introduction

What is JavaScript?

JavaScript? What for?!!!

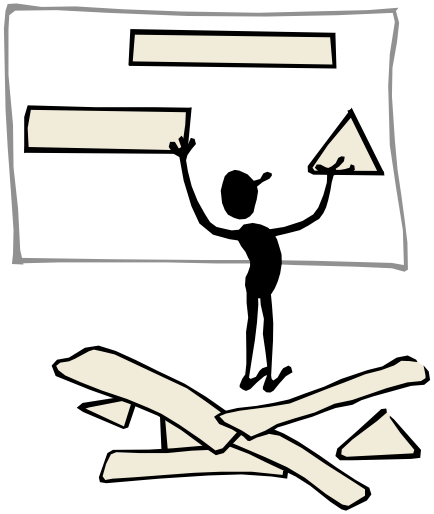
Language basics

Scripting

Document Object Model

Events

Conclusions



Simple script

```
<BODY BGCOLOR="black" TEXT="white">
  <SCRIPT LANGUAGE="JavaScript">
    lastModDate = new Date(document.lastModified)
    document.write("Last updated: " + lastModDate)
  </SCRIPT>
</BODY>
```

Simple script: Description

<SCRIPT LANGUAGE="JavaScript">

- begins the scripting block as “JavaScript”.

lastModDate = new Date(document.lastModified)

- instantiates a new date object into the lastModDate variable with the documents lastModified date.

document.write("Last updated: " + lastModDate)

- writes a concatenated text string out to the browser window.

</SCRIPT>

- ends the scripting block

Script hiding

Earlier versions of Netscape Navigator and Internet Explorer cannot recognize JavaScript. Therefore unkown-language scripts must be hidden.

```
<!--
<SCRIPT LANGUAGE="JavaScript">
     lastModDate = new Date(document.lastModified)
     document.write("Last updated: " + lastModDate)
   </SCRIPT>
-->
```

Plan of the Lecture

Introduction

What is JavaScript?

JavaScript? What for?!!!

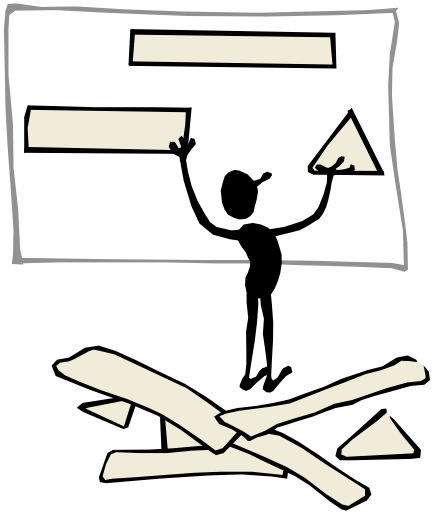
Language basics

Scripting

Document Object Model

Events

Conclusions



Document Object Model

Objects, objects, objects...

- An object has methods and properties.

- Methods perform an action

```
lastModDate = new Date
  (document.lastModified)
lastModMonth = lastModDate.getMonth()
```

- Properties are set values

```
lastModDate = new Date
  (document.lastModified)
lastModMonth = lastModDate.getMonth()
```

Document Object Model

Object models

- **DAO** - Data Access Objects
- **VBA** - Visual Basic for Applications
- **MFC** - Microsoft Foundation Classes
- **DOM** - Document Object Model
- **ADO** - Active Data Objects
- **etc**...

Document Object Model

DOM

- An object model is a set of objects that make up a system - a set of associations
- The Document Object Model (DOM) are the objects resident within the browser.
- JavaScript can access the DOM.

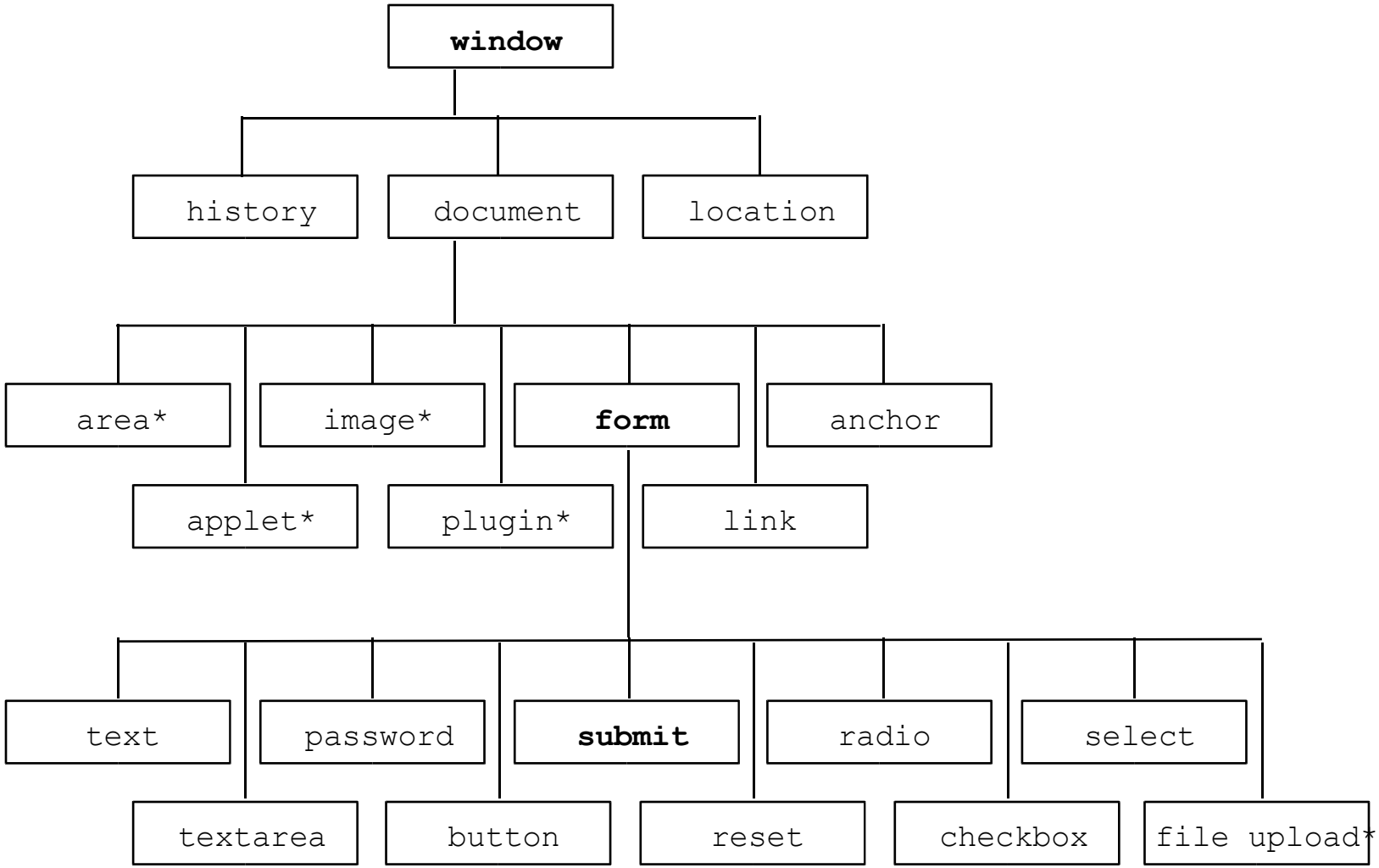
```
<HTML>
<TITLE>Using the DOM</TITLE>
<BODY>
<SCRIPT>document.write(document.title);</SCRIPT>
</BODY>
</HTML>
```

Document Object Model

DOM Structure

- Window object represents a window or frame
- Document object represents the contents
- DOM standard
  - level “0”, level “1”, level “2”, etc...
- Level “0”
  - Some support in IE v4
- Level “1”
  - Full support in IE & NS v5

Document Object Model



Plan of the Lecture

Introduction

What is JavaScript?

JavaScript? What for?!!!

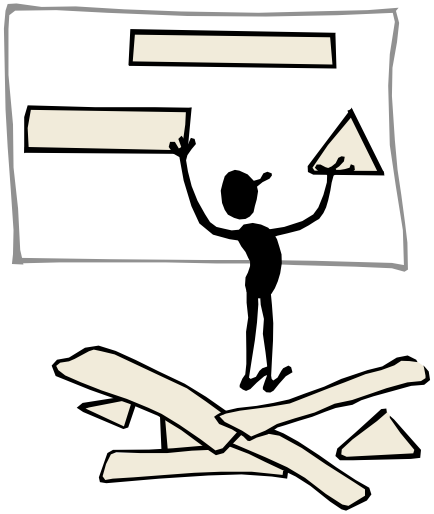
Language basics

Scripting

Document Object Model

Events

Conclusions



Events

Event-drivent programming

- Event handlers
  - Script / Code which executes in response to a particular event

```
onEvent = {some script};
onLoad = {some other script};
```

Events

Embedded vs. non-embedded

- **Embedded**

```
<HTML>
<BODY onLoad="window.defaultStatus='Welcome to JavaScript'” >
</BODY></HTML>
```

- **Non-embedded**

```
<HTML>
<SCRIPT LANGUAGE="JavaScript">
function scrollMsg() { // display scrolling text in status
   bar
  window.status = "Welcome to the joys of JavaScript . . ."
  msg = msg.substring(1, msg.length) + msg.substring(0, 1)
  setTimeout("scrollMsg()", 150) }
</SCRIPT>
<BODY onLoad="scrollMsg()” >
</BODY></HTML>
```

Events

Browser events

- Browser objects are able to respond to events
- Not all objects can respond to all events

- 12 events are defined for various JavaScript events (often buttons or text boxes). The standard syntax is:

```
<TAG onEvent="event_handler()">
```

Events

	onBlur	onChange	onClick	onFocus	Miscellaneous Events			
button	X		X	X				
checkbox	X		X	X				
file	X	X		X				
form					onSubmit	onReset		
image					onAbort	onLoad	onError	
link			X		onMouseOut	onMouseOver		
password	X	X		X				
radio	X		X	X				
reset	X		X	X				
select	X	X		X				
submit	X		X	X				
text	X	X		X				
textarea	X	X		X				
select	X	X		X				
window	X			X	onError	onLoad	onUnload	

onLoad and onUnload

- **Example**

```
<HTML>
<BODY onLoad="window.defaultStatus='Welcome to JavaScript'" >
</BODY></HTML>
```

- The onLoad event is generated whenever certain form objects are loaded into the browser window.
- The onUnload event can also be included in the <BODY> tag to trigger a function whenever a window (or frameset) is exited. onUnload is frequently put to use to load a new web page when the user exits the current page.

Events

onMouseOver and onMouseOut

- **Examples**

```
<A HREF="home.htm"
   onMouseOver="window.status='Home Base'; return true">
  <IMG SRC="home.gif" BORDER="0">
</A>
<INPUT TYPE=button onMouseOver = _
"Alert('The Clear button clears values');"_ VALUE="Clear"
   NAME="clearbutton">
```

- These events are generated whenever the user moves the mouse cursor into or out of any form element, image, or link. They find their greatest use as triggers for alert windows which inform the user of the purpose of an element.

onError and onAbort

- **Examples**

```
<IMG SRC="mypic.jpg" onAbort = "Alert('Image corrupt')">
<IMG SRC ="mypic.jpg" onAbort = "PrintAbortMess()">
```

- These two events occur when a problem is encountered during image loading. onError is triggered when an image cannot be loaded due to corruption. The onAbort event occurs only when the user aborts loading an image by pressing the browser's stop button.

onBlur and onFocus

- A ‘Blur’ is what happens when an object loses focus. Focus is used to refer to the currently selected object, which is usually the topmost object on the desktop. Hence, each time one object receives the focus (as from a mouse click) another object loses the focus and is *blurred*.

onChange and onSelect

- A selection occurs whenever the user highlights the contents of a text box or text area, often by clicking and dragging the mouse cursor over the contents of the object. A change event occurs whenever the contents of a text box or are is edited.

Events

onSubmit and onReset

- **Example**

```
<FORM onSubmit = "return(confirm('okay to submit'))">
<FORM onReset = "return(confirm('okay to reset'))">
```

- onSubmit and onReset are generated just prior to the submit() and reset() methods associated with the form object (see previous chapter for a brief review of these methods). These events are used to provide the user a last opportunity to change his choice, since the results are irreversible.

Events

onClick

- **Example**

```
<INPUT TYPE=button VALUE="Products" onClick = “window.open
('URL#index')” NAME="products">
```

- Whenever the user clicks on an object in a web page, the onClick event is generated. Unfortunately, JavaScript lacks the ability to discriminate between a right mouse button click and a left mouse button click. onClick events can be used to jump to other web pages

Conclusions

- JavaScript IS Web Programming
- Java-like, object-based scripting language
- Flexible and simple mean of making HTML pages attractive for its visitors
- Cross-browsing problems
- C-like syntax
- DOM sweet DOM
- Event-driven programming

References

Peter Rawsthorne *Introduction to programming with JavaScript*

Danny Goodman *Javascript Bible, 3rd ed.*

http://developer.netscape.com/one/javascript/

Questions?