729G26 Interaction Programming

Lecture 3

LINKÖPINGS UNIVERSITET

Lecture overview

- JavaScript
- JavaScript debugging in the web browser

LINKÖPINGS UNIVERSITET

HTML/CSS Interplay

CSS is about colors, shapes, typography and layout, but without content to apply these attributes to, CSS is nothing.

LINKÖPINGS UNIVERSITET

Aspects of CSS styling

- **Appearance**:

  Color

  Size

  Shape

  Typography

  ...

- **Layout**

  Position on the webpage

  Ordering with respect to what is on top and what is below

  Behavior when the window resizes

  ...

Structural aspects

- **The DOM**

  Which nodes are **contained** by which elements

  What is the **element type** of each node in the DOM

  Which **classes**, if any, have been assigned to each node in the DOM

  What **ID** if any, has been assigned to each node in the DOM

What can JavaScript do?

In the world of HTML and CSS, JavaScript can do **anything**.

JavaScript can be used to

- Manipulate the DOM, i.e.

Change DOM structure

Change the ID of DOM nodes

Change the classes of DOM nodes

Change contents of DOM nodes

- Manipulate the CSS

Add inline CSS to the DOM

Change the contents of the stylesheet

- Capture web browser events

changes to the web browser window (size, active/not active, URL etc)

changes originating from input devices: mouse movement, keyboard key presses, single and multi-touch events

What is JavaScript

- JavaScript is a programming language that is used in many different applications.
- JavaScript is not Java, they are completely separate beings.
  - Formally specified as ECMAScript
    http://www.ecma-international.org/publications/standards/Ecma-262.htm
  - Programming language features

    interpreted - like Python

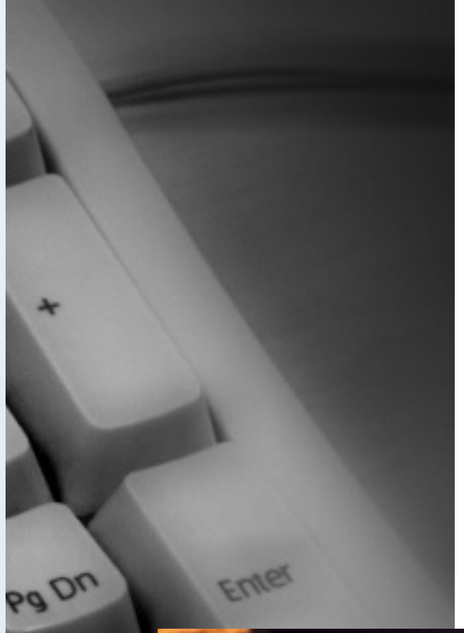
    dynamically typed - like Python

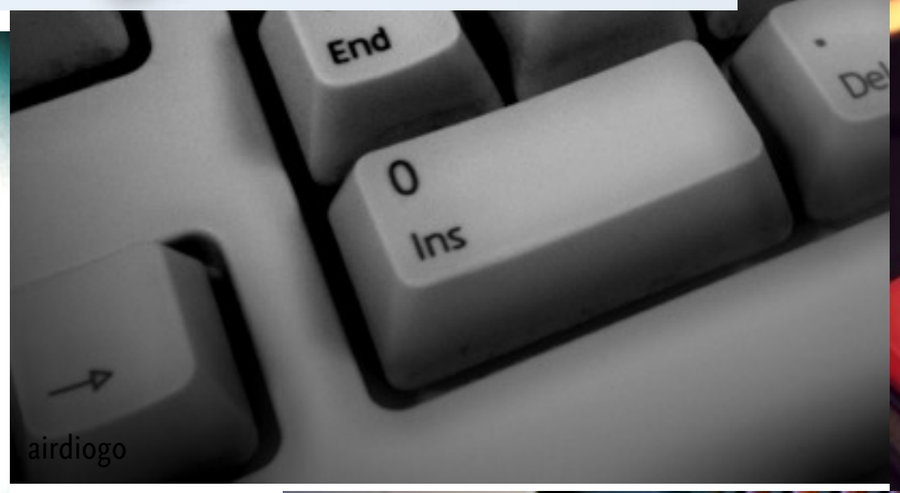
    prototype-based programming (similarities with OOP)

Where is JavaScript used

- In web browsers - client side scripting
- On servers - server-side scripting (e.g. Node.js)
- Google Chrome Extensions are written in JavaScript
- Acrobat Reader supports JavaScript in PDF files
- Photoshop, Illustrator, Dreamweaver & InDesign support scripting using JavaScript
- ActionScript, used in Adobe Flash is ECMAScript

redspotted

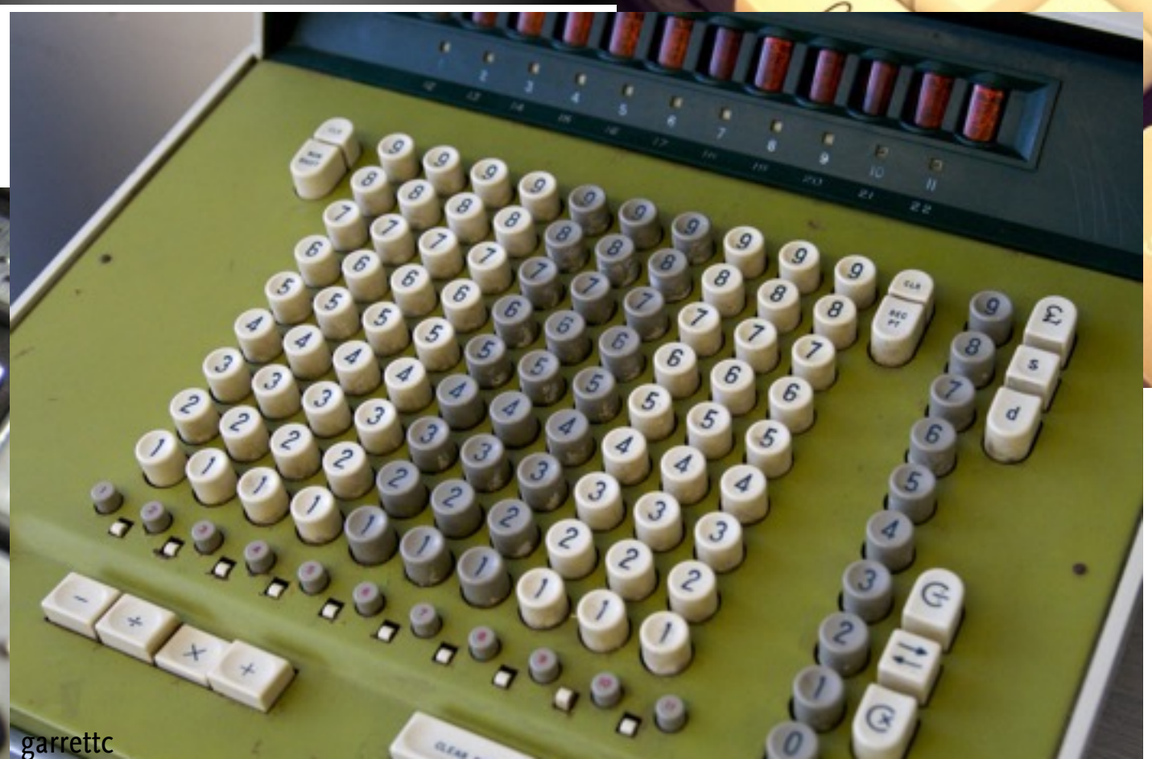




airdiogo

waltstoneburner

redspotted



garrettc

millerdial

JavaScript is JavaScript, is JavaScript?

- The language and syntax is the same
- The API we use JavaScript to talk to is different.

Google Chrome Extension

```
// Create a simple text notification:
var notification = webkitNotifications.createNotification(
  '48.png',  // icon url - can be relative
  'Hello!',  // notification title
  'Lorem ipsum...'  // notification body text
);
```

JavaScript for PDFs - annotation

```
/* The getAnnot method returns an Annotation object, an object that holds all
data of the annotation. This method takes two parameters, nPage (page number)
and cName (the name of the annot). For example, */

var a = this.getAnnot({ nPage: 2, cName: "myAnnot" });
if ( a != null ) {
    console.println("This annot has type " + a.type);
    if ( (a.type != "FileAttachment") || (a.type != "Sound") )
        console.println("The comment of this annot is " + a.contents);
}
```

Get div nodes from HTML DOM

```
var divs = document.getElementsByTagName('div');
for (var i = 0, div; div = divs[i]; i++) {
  /* Process div in some way */
}
```

JavaScript resources

http://javascript.crockford.com/javascript.html

https://developer.mozilla.org/en-US/docs/Web/JavaScript

http://docs.webplatform.org/wiki/javascript

http://www.w3schools.com/js/

Debugging in the web browser

error messages etc are shown in the console

Syntax

- Before we can do all the cool stuff, we need to learn how to speak JavaScript - syntax and vocabulary
- statements and comments
- variables
- datatypes
- functions
- conditional statements
- loops

Codecademy exercises

- Codecademy exercises to practice your JavaScript.
- Codecademy exercises should be completed by each group member.
- You will not learn JavaScript during this lecture, you will get an overview adapted to your prerequisite knowledge of programming.

Comments and statements

```
/* This is a multi-line comment
   that spans two rows */

// this is a single line comment
// each statement is followed by a semi-colon
// console.log() is used to print to the
// JavaScript console in e.g. the browser
console.log("Hello world!");
```

Variables

```
// a variable is declared using the var keyword
var myVariable;

// a variable is set using the = character
myVariable = 5;

// a variable can be declared and set at the same time
var myOtherVar = 6;

// more than one variable can be declared and set at the same time
var x, y, z;var p = 0, q = "hello", s = true;
```

Levels of nothing

- When a variable is declared, but not set
  `undefined`

- When a variable is declared and we want it to have an
  `empty value`
  `null`

- Trying to access a variable that has not been declared will result in a `ReferenceError`

Datatypes

- Numbers
- Booleans
- Strings
- Arrays
- Objects

Numbers and operators

```
// all numbers in JavaScript are stored as floating point numbers
var a = 3;
var b = 2.0;

// we can use the operators we expect from e.g. python
console.log(a+b);
console.log(a-b);
console.log(a/b);
console.log(a*n);
```

JavaScript has two boolean values

- `false`
- `true`
- The following evaluate to `false`, all other values evaluate to `true`

```
0
null
""
false
```

Strings

```
var a = "Hello";
var b = "World!";
console.log(a + " " + b);
```

Assignment operators

```
// simple assignment
var x = 1;
var y = 10;

// shorthand version of x = x + y
x += y;

// shorthand version of x = x - y
x -= y;

// shorthand version of x = x + 1
x++;

// shorthand version of x = x - 1
x--;
```

Arrays

```
// An array in JavaScript is similar to the list in python

// we can create a new, empty array in different ways
var myArray1 = new Array();
var myArray2 = Array();
var myArray3 = [];

// we can fill an array when initializing it
var myArray4 = [1, 2, "hello"];

// we can set any position in an array, empty spots will be
undefined
myArray4[10] = "what!";

// myArray4: [1, 2, "hello", undefined ? 7, "what"]
```

Objects

- JavaScript is an object oriented language.
- However: based on prototypes, rather than classes.
- Objects have properties.
  - Properties are accessed using dot notation: `object.property`
  - Objects can also have functions that can behave as methods.
  - Functions in objects are called using dot notation: `object.function()`

Functions

- named functions
- anonymous functions
- function objects

LINKÖPINGS UNIVERSITET

Function examples

```
// a named function
function helloWorld() {
    console.log("hello world");
}

// defining an anonymous function, and keeping a reference to
// its function object
anonymous = function() {
    console.log("Anonymous hello world");
}

// grabbing the function object of a named function
hello = helloWorld;
```

A note on JavaScript code style

- use camelCase for variable and function names
- start variable and function names with a lower case letter
- use meaningful names for variable and function names

LINKÖPINGS UNIVERSITET

Function demo

https://trinket.io/html/b63f83425f

LINKÖPINGS UNIVERSITET

Variable scope

```
// all variables declared outside a function are global
var myGlobalVariable = 0;

function localExample() {
    // all variables declared inside functions are local to that
    // function
        var myLocalVariable = 100;
        myGlobalVariable = 5;

    }
```

Conditional statements

```
if(a == b) {
    statement;
    statement;
} else if (a == c) {
    statement;
    statement;
} else {
    statement;
    statement;
}
```

LINKÖPINGS UNIVERSITET

Comparison operators

- equal value (with type conversion): `==`
- equal value and datatype (no type conversion): `===`
- less than: `<`
- less than or equal: `<=`
- greater than: `>`
- greater than or equal: `>=`
- different value: `!=`
- different value or different type: `!==`

Logical operators

- not: !
- and: &&
- or: ||

Loops

- while
- do while
- for
- break and continue

LINKÖPINGS UNIVERSITET

The while loop - while this is true, do this

```
while(condition) {
    //statement
    //...
}
```

The do while loop - do this, if this is true, loop

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

for loop - special while loop

```
for(i = 0; i<10; i++) {
    //statements
    //...
}
```

LINKÖPINGS UNIVERSITET

As in e.g. python, we can break and continue

- `break`: abort current loop
- `continue`: skip the rest of the current iteration and move to the next iteration

Keeping code clean is up to the programmer in JavaScript.

Be consistent, or be in trouble

Connecting the script to the content

- Two ways of adding JavaScript to an HTML file
- Use the <script> tag and inline JavaScript
- Use the <script> tag with the `src` attribute

LINKÖPINGS UNIVERSITET

Where to load the JavaScript

- The script is run when loaded.
- If the script depends on e.g. existing elements on the page, we need to load the script when those elements have been loaded.
- E.g.

  last in the `<body>` tag.

  using the `window.onload` event handler (we'll get to this later)

JavaScript and the DOM

- More about the DOM
- What can we do with the DOM using JavaScript
- How can we use this to create interaction?
- What types of interaction - input/output - can we use in a web browser
  - show/hide
  - create/destroy
  - move, position
  - change shape, color
  - click, drag, drop - elements on page and files from the computer
  - type on keyboard
  - images, movies, animations

Accessing the DOM tree

- **Elements**

  get an element using its id

  get an array of elements using their class

  accessing an element’s relatives

- **Text**

  get an element’s text content

  using innerHTML

- **Attributes**

  get an element’s attribute nodes

  get an element’s specific attribute value

Accessing elements

```
// get a single element by using its id
myElement = document.getElementById("the id");

// get an array of elements of a specified type
myElements = document.getElementsByTagName("a");
```

```
// get an elements relatives
myElement.firstChild;
myElement.lastChild;
myElement.parentNode;
myElement.nextSibling;
myElement.previousSibling;
```

Accessing element content

```
// get all HTML inside an element
allHTML = element.innerHTML;

// just get the text content of an element
theText = element.textContent;
```

Accessing attributes

```
// array access to an element's attributes
theAttributes = element.attributes;

// get the value of an element's attribute
anAttribute = element.getAttribute("attribute name");
```

Editing the DOM tree

- replace `innerHTML`
- set attributes

Change innerHTML and set an attribute value

```
// set inner HTML of an element
element.innerHTML = "new inner HTML";

// set the value of an element's attribute
element.setAttribute("attribute name", "attribute value");
```

Creating new nodes in the DOM tree

- Create
- Initialize
- Append or insert

LINKÖPINGS UNIVERSITET

Create an element node

```
// create an h1 element
heading = document.createElement("h1");

// create a paragraph element
paragraph = document.createElement("p");
```

Create text nodes

```
// create heading text
headingText = document.createTextNode("Heading 1");

// create paragraph text
paragraphText = document.createTextNode("Lorem ipsum dolor sit amet,
consectetur adipisicing elit. Aliquam, rem, adipisci, iste nisi vel
quibusdam nulla tenetur repudiandae possimus maiores inventore
doloremque asperiores aut omnis cum et harum provident suscipit!");
```

Add text nodes to elements

```
// create heading text
headingText = document.createTextNode("Heading 1");

// create paragraph text
paragraphText = document.createTextNode("Lorem ipsum dolor sit amet,
consectetur adipisicing elit. Aliquam, rem, adipisci, iste nisi vel
quibusdam nulla tenetur repudiandae possimus maiores inventore
doloremque asperiores aut omnis cum et harum provident suscipit!");

// add text to heading and paragraph
heading.appendChild(headingText);
paragraph.appendChild(paragraphText);
```

Appending and inserting elements

```
// append paragraph to body, append always places an element last
body = document.getElementsByTagName("body")[0];
body.appendChild(paragraph);

// insert an element at a specific position
bodyNodes = body.childNodes;
body.insertBefore(bodyNodes[0], heading);
```

Changing an element's CSS

- inline style
- Class

LINKÖPINGS UNIVERSITET

Set an element's style using JavaScript

```
// grab the first heading
heading = document.getElementsByTagName("h1")[0]

// change the font-family property of the heading
// note that properties with a dash ('-') are camelCased
heading.style.fontFamily = "Times"
```

LINKÖPINGS UNIVERSITET

Manipulating style with class

```
// grab the first heading
heading = document.getElementsByTagName("h1")[0]

// set the headings class
heading.className = "maintitle";

// add additional class
heading.className += " fancy";
```

User interaction - event handling

LINKÖPINGS UNIVERSITET

Elements have event handlers

- When an event is triggered in the context of an element, its event handlers are triggered
- **Event examples**

  when a mouse enters the area of an element

  when a mouse leaves the area of an element

  when the mouse is clicked on an element

  when a form field is given focus

Event handling

- Typical mouse interaction
  - `onmouseover`
  - `onmouseout`
  - `onclick`
- Typical form interaction
  - `onfocus`
  - `onblur`
  - `onsubmit`

Event handlers are references to functions

- We can set an elements event handler to a named function object or create an anonymous function object

Functions revisited

- named functions: a defined function with a name
- anonymous functions: a defined function without a name
- function objects: the actual function, when named, the name is our “handle” on the function

Function examples

```
// a named function
function helloWorld() {
    console.log("hello world");
}

// defining an anonymous function, and keeping a reference to
// its function object
anonymous = function() {
    console.log("Anonymous hellow world");
}

// grabbing the function object of a named function
hello = helloWorld;
```

Writing an event handler

```
// using an anonymous function
element.onclick = function() {
    alert("Hello World");
};

// using a function object
function hello() {
    alert("Hello World!");
}
element.onclick = hello;
```