



Digitaalisen median tekniikat

Luento 4: JavaScript

Luennot

1. Intro
2. XHTML
3. CSS
4. **JavaScript**
 - **Historia**
 - **Syntaksi**
 - **Dom**
 - **Esimerkki: kuvagalleria**
5. JavaScript-kirjastot & AJAX
6. Käytettävyys & saavutettavuus
7. Palvelinohjelmointi (PHP)
8. PHP jatkuu + pelkkiä kysymyksiä luento
9. Video- ja kuvaformaattit
10. Quirks
11. Hyviä, mutta vähän käytettyjä web-standardeja
12. Yhteenveto

HUOM.

- Kuinka moni ei ole tehnyt JavaScriptiä?
 - Ensimmäisten harjoitusten kyselyn 113 vastauksesta
 - 66 ei ole tehnyt JavaScriptiä
 - **Yli puolet!**
- Meillä on nyt vain 2h aikaa, siinä ajassa ehditään päästä vain alkuun
- Luentojen teoriaa on tiivistetty edellisistä vuosista
 - Sillä JavaScriptiä oppii vain tekemällä
 - Se ei kuitenkaan ole vaikeaa!
 - "JavaScript was influenced by many languages and was designed to look like Java, but be easier for non-programmers to work with." (wikipedia)
 - Toisella luentotunnilla: koodataan!

Historia

- JavaScriptillä ei ole mitään tekemistä Javan kanssa.
- Toistetaan vielä:
 - JavaScriptillä ei ole mitään tekemistä Javan kanssa...
...paitsi ehkä samankaltainen syntaksi.
- 1995 Brendan Eich kehittää "Mochan"
 - Nimetään myöhemmin LiveScriptiksi ja sittemmin JavaScriptiksi
- 2005 "Ajax" (Asynchronous JavaScript and XML)
 - 1999 XMLHttpRequest Internet Explorer 5.0
 - JavaScript otetaan vakavammin tekniikkana
- 2009 JavaScript moottorit kilpailevat nopeudessa

Yleinen kuvaus

- Asiakaspuolen tulkettava skriptikieli
 - Tapahtumapohjainen
 - Voi reagoida selaimen tapahtumiin, kuten klikkauksiin
 - Tyypillisiä käyttökohteita:
 - Sivun muokkausta dynaamisesti
 - Kommunikointi palvelimen kanssa
 - Lomakkeiden tarkistus ym. (käytettävyys)
- ...**Ärsyttävät efektit**

Rajoitukset

- Ei voida lukea asiakaskoneen paikallisia tiedostoja eikä muutakaan laitteistoa.
- Eikä käsitellä palvelimella sijaitsevia tiedostoja suoraan.
- ”Cross site” estetty
 - Sama alkuperä *turvallisuusmittana*
 - Sisältöä saa ladata ainoastaan samalta sivustolta, jolla skriptiä suoritetaan.

Syntaksi

- Java/C tyylinen
- Case sensitive
- Lauseen erottimena ; tai rivinvaihto
- Kommentit

```
/* Ensimmäinen JavaScript esimerkki:  
   Tervehditään käyttäjää HÄLYTYSLAATIKOLLA!  
*/  
function tervehdys(nimi) {  
    alert("Hei " + nimi);  
}  
  
// Kutsutaan tervehdystä  
tervehdys("Brendan");
```

Operaattorit

- + on kuormitettu
 - "alfa" + "beta" → "alfabeta"
 - 1 + "beta" → "1beta"
 - 1 + 2 + "alfa" → "3alfa"
- Vertailu on pääasiassa kuten Javassa
 - Merkkijonoja voi verrata samoilla operaattoreilla kuin lukuja
 - "sama"=="sama"

Kummalliset asiat mahdollisia

// nyt tehdään kummia asioita

```
function tervehdys(nimi) {  
  alert("Hei " + nimi);  
}
```

Huom! Jos et pidä näitä kummallisina, niin ei haittaa. Opit varmasti helposti JavaScriptiä silti!

// Kutsutaan tervehdystä

```
tervehdys("Brendan");
```

// Edelleen toimii, vaikka parametrit eivät täyty
tervehdys();

// määritellään tervehdys muuttujaksi

```
var tervehdys = "hei hei";
```

// ok, muuttuja on edelleen ok

```
alert(tervehdys); // "funktio" onkin nyt muuttuja!
```

// määritellään tervehdys uudestaan

```
var tervehdys;
```

```
alert(tervehdys); // arvo on edelleen "hei hei" !
```

Tietotyypit

- Luvut
 - Kokonaisluvut, liukuluvut
 - NaN = not a number
- Merkkijonot
 - "näin" tai 'näin' tai "hän sanoi 'näin'"
 - Erikoismerkit "kuten\tjavassa\n"
- Totuusarvot
 - true ja false
 - Laskennassa 1 ja 0 (true+true = 2)
 - if ("" || null || undefined || 0) { ei }
 - if ("epätyhjä" && Window && 1) { kyllä! }

Muuttujat

- Määritellään:
 - var muuttuja = "arvo"
 - var eka = 1, toka = 2
 - Voi määritellä myös ilman "var" sanaa, mutta ei tehdä näin (kohta lisää)
- Tyypittämättömät
- Tyyppi voi vaihtua ja sitä ei voi määritellä
- Tyypimuunnokset automaattisia
 - Eksplisiittiset tyypimuunnokset:
 - parseInt("20 noin") → 20
 - parseFloat("21.20 on tarkempi") → 21.20
 - parseInt("noin 20") → NaN
- typeof(muuttuja) kertoo tyyppin
 - typeof(2) → "number"
- undefined ellei alkuarvoa anneta (var muuttuja;)
 - undefined == null

Näkyvyys

- Periaatteessa funktioiden sisällä määritellyt muuttujat ovat paikallisia ja ulkopuolella määritellyt ovat globaaleja.
- Asiaa kannattaa kuitenkin kokeilla:

```
var globaali = "maailmanlaajuinen";
```

```
function kokeilu() {  
  alert(globaali); // undefined !  
  var globaali = "paikallinen";  
  alert(globaali); // alert: paikallinen  
}
```

```
kokeilu();  
alert(globaali); // alert: maailmanlaajuinen !
```

```
var globaali = "maailmanlaajuinen";
```

```
function kokeilu() {  
  alert(globaali); // maailmanlaajuinen  
  globaali = "paikallinen";  
  alert(globaali); // alert: paikallinen  
}
```

```
kokeilu();  
alert(globaali); // alert: paikallinen !
```

Funktiot

- Toisin kuin Javassa, JavaScriptissä voidaan määritellä irrallisia funktioita.
 - Eli: funktion ei tarvitse olla luokan sisällä
- Paluuarvon tyyppiä ei voida määritellä.
- Parametritkaan eivät ole niin pakollisia.

```
function kokeilu(eka,toka) {  
  if (eka) alert(eka);  
  if (toka) alert(toka);  
  
  return eka+toka;  
}
```

```
palautus1 = kokeilu("moi", "hei"); // palauttaa "moihei"  
palautus2 = kokeilu("moi"); // palauttaa "moiundefined"  
palautus3 = kokeilu(); // palauttaa NaN ! (koska undefined + undefined = NaN)
```

Oliot

- Hmm! Kielessä on ”Olion käsite”, mutta ei esimerkiksi periytymistä
- Olioista puhutaan tarkemmin seuraavalla luentokerralla
- Oliot ovat ”kokoelma nimettyjä ominaisuuksia” (kohta lisää)
- `this` kuten Javassa
- Poisto ”olioavaruudesta” `delete()`:llä

Oliot

```
function Song() {  
}
```

```
abba = new Song();
```

```
abba.name = "Waterloo";  
alert(abba.name);
```

```
var abba = { name: "Waterloo" };  
  
alert(abba.name);
```

```
var abba = new Object();  
abba.name = "Waterloo";  
alert(abba.name);
```

```
var abba = {  
  name: "Waterloo" ,  
  play: function() {  
    alert("Playing "+this.name);  
  }  
};
```

```
alert(abba.name);  
abba.play();
```

Oliot

```
function Heppa(nimi){
  this.vaihdaNimi = function(nimi){
    this.nimi = nimi;
  };

  // Jotta uuden "olion" luonnissa parametrina annettu nimi asettuu
  this.vaihdaName( name );
}

var polle= new Heppa("Polle"); // Luodaan uusi olio "Luokasta"
alert(polle.nimi); // Alertbox: Polle

polle.vaihdaNimi("Super-Polle");

alert(polle.nimi); // Alertbox: "Super-Polle"
```

Esimerkki otettu <http://www.ejohn.org> :sta jossa on paljon edistynyttä matskua

Taulukot

- Yleinen rakenne JavaScriptissä.
- Taulukon koko on dynaaminen: ei tarvitse määrittää ennalta.
- Taulukon käyttöön kannattaa tutustua.
→ Seuraava kalvo.

Taulukot

```
var taulu = new Array();
```

```
taulu[0] = 1;
```

```
taulu[1] = 2;
```

```
taulu.push(3);
```

```
alert(taulu); // alert: 1,2,3
```

```
taulu.pop();
```

```
alert(taulu); // alert: 1,2
```

```
alert(taulu.length); // alert: 2
```

```
var toinen = [3,2,1];
```

```
toinen.sort();
```

```
alert(toinen); // alert: 1,2,3
```

```
var assosiatiivinen = new Array();
```

```
assosiatiivinen["eka"] = 1;
```

```
assosiatiivinen["toka"] = 2;
```

```
alert(assosiatiivinen["eka"]); // alert: 1
```

Tapahtumat

- Tapahtumat (events) mahdollistavat käyttäjän toimintoihin ja "sivun" tapahtumiin reagoinnin.

- ``

Huom: näin eventtejä ei tulisi kuitenkaan käyttää!

- `<body onLoad="rasittavaTervetuloToivotus();">`

JavaScriptin liittäminen

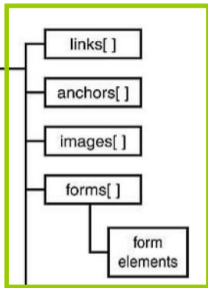
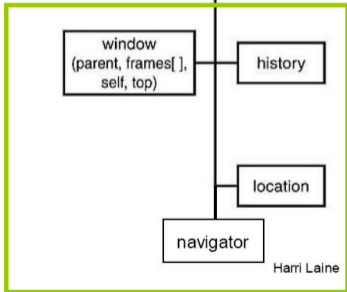
- `<script type="text/javascript" src="javascrip/ulkoinen.js"></script>`
- `<script type="text/javascript">
 <!-- piilotetaan vanhemmilta selaimilta?
 alert("hei!");
 // < huomaa JS kommentti ! -->
</script>`
- ``
- `Klikkaa`

Selainluokat

- Selain tarjoaa omia luokkia
 - window, document, history...
- Esimerkkejä:
 - window.location = "<http://www.example.com>";
 - Vaihtaa selaimen osoitekentän osoitteen ja siirtyy sivulle
 - document.body
 - Sisältää dokumentin **DOM-puun** body-oksen
- DOM-puu:
 - Document Object Model
 - Dokumentin rakenne puu-esityksenä, jota voi käsitellä JavaScriptillä tai "millä tahansa"

DOM-puu

BOM:
Browser Object Model



DOM Level 0
Document Object
Model

DOM Level 1:
Esim. document.body ja
getElementById();

(harjoituksissa 3 piirretään DOM Level 1 puu)

DOM-metodeja

- `getElementById("taulukot")`
 - Hae elementti id:n "taulukot" perusteella
- `getElementsByTagName("a")`
 - Hae elementit tagin perusteella (palauttaa taulukon)
- `createElement("p")`
 - Luo paragraph -elementin
- `elementti.setAttribute("src","kuva.jpg")`
 - Asettaa elementin attribuutin "src" arvoksi "kuva.jpg"
- Näitä tutkitaan kohta esimerkin kanssa

Työkalut

- JSLint: DEMO
 - <http://www.jslint.com>
- Firebug laajennos
- Web Developer laajennos

Testaus

- Nyt mennään rajoille!
- Jos kiinnostaa niin mm:
 - <http://ejohn.org/blog/fireunit/>
- Jos ei, niin ei tarvitse!