Web 2.0 ja uudet sovellustekniikat

Tancred Lindholm

T-110.2100 Johdatus tietoliikenteeseen
kevät 2010

Luennon sisältö

- Web 2.0: Luku/kirjoitus nettiselailu ja sosiaaliset mediat
- Nettiselailu hajautettuna järjestelmänä
- HTTP rajapintana viestien lähettämiseen
- Nettiselailun kielet: HTML, XML, Javascript
- Javascript-ohjelmointia selaimessa
- Tästä eteenpäin: Client/Server, P2P, AJAX, REST

Web 2.0

- **Netin** (eli the Word Wide Web, WWW) alkuperäinen idea tutkimusmateriaalin jakaminen tutkijoiden kesken
- 90-luvun netissä kuitenkin vain muutama tiedon julkaisija verrattuna tiedon kuluttajiin
- 2000-luvun alussa käsite Web 2.0
  - loppukäyttäjä tuottaa ja jakaa tietoa (sekä lukee että kirjoittaa Netissä)
  - Selaimessa näkyvä sivu on ennemminkin **interaktiivinen sovellus** kuin **staattinen** sivu

Web 2.0

- Viime aikoina sosiaaliset mediat nousseet suosioon
- Sosiaalisen median pääpirteet
  - Yksityishenkilö julkaisijana
  - Tiedon julkaisu ei vaadi erityistaitoja
  - Reaaliaikaista tai lähes reaaliaikaista
  - Pysyvyys vaihtelee
- Wikipedia
- Blogit
- Facebook
- Jaiku (RIP, nykyään qaiku.com)

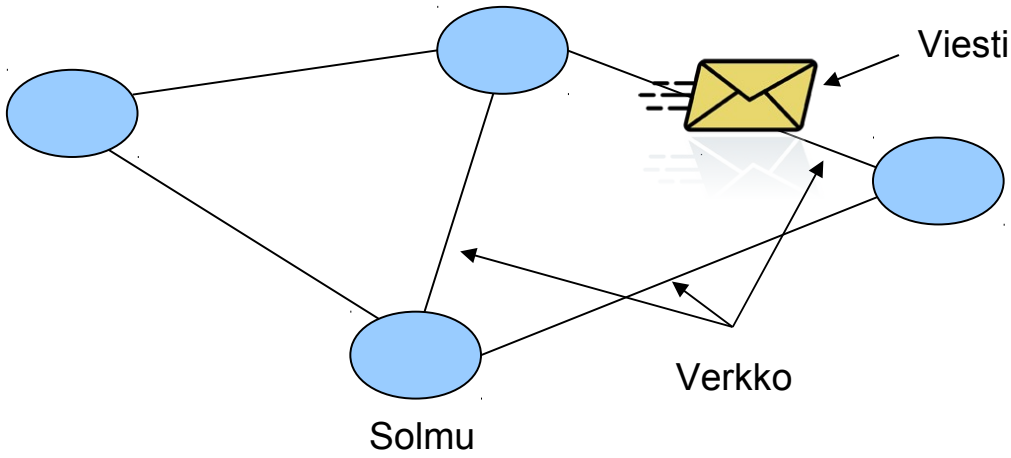
facebook

Entä miten Facebook toimii?

- Miten Web 2.0 tyylinen sovellus selaimessa (esim Facebook) toimii?
- Tästä puhutaan tänään
  - Miten selain toimii?
  - Miten sitä ohjelmoidaan?
  - Minkä muotoista dataa?
  - Entä tarvittava infrastruktuuri, ns. palvelin?

Hajautettu järjestelmä

- **Hajautettu järjestelmä** koostuu
  - **Solmuista,** joissa suoritetaan laskentaa (computational node)
  - **Tietoliikenneverkosta**, joka välittää viestejä solmujen välillä



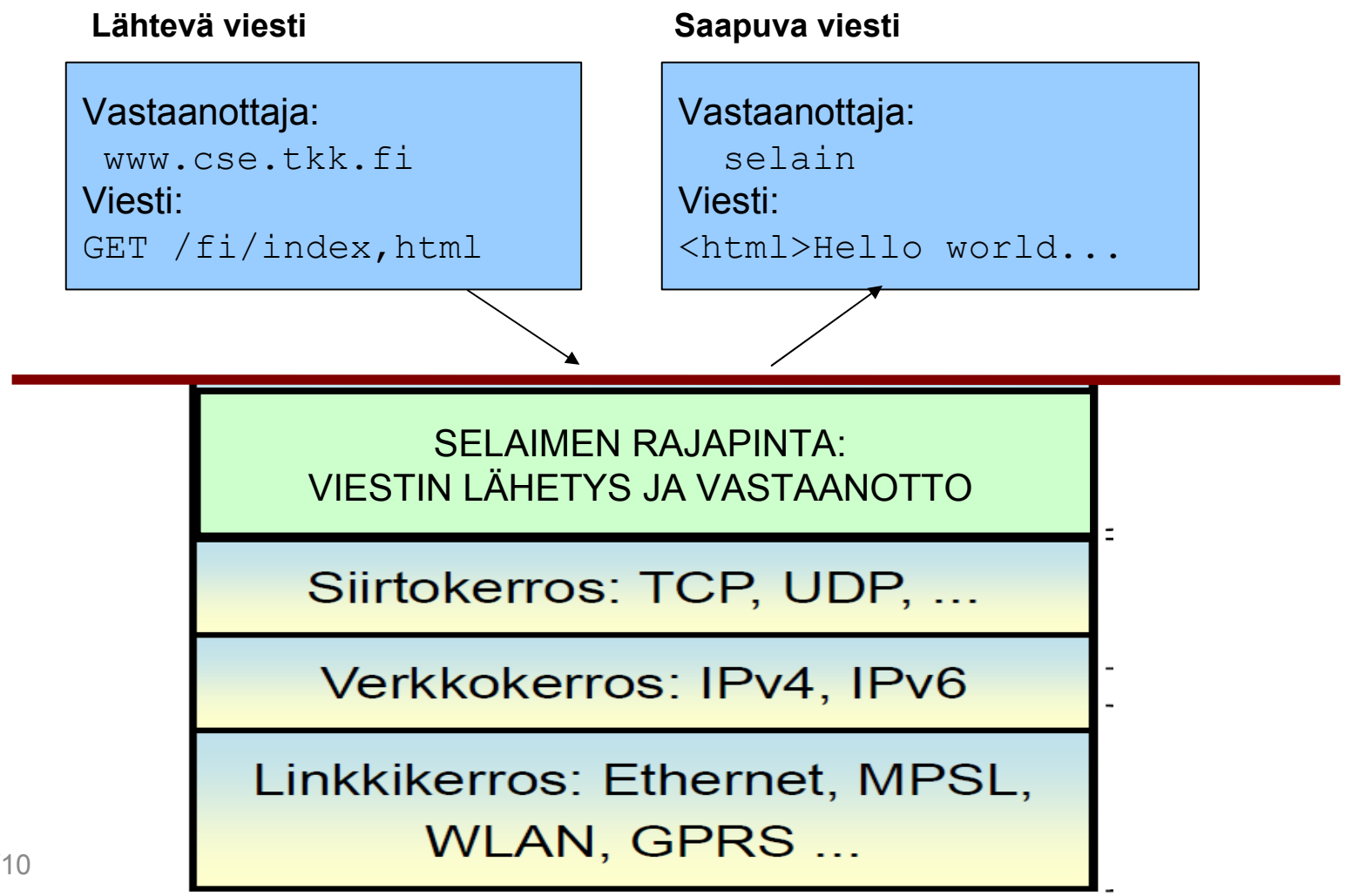
Selain ja Nettipalvelin

- Esimerkkinä hajautetusta järjestelmästä
- (Loogisesti) keskitetty solmu jossa on **palvelinohjelma** sekä tämän kanssa kommunikoivat solmut jossa on loppukäyttäjän nettiselain
- Tämä on ns asiakas/palvelin-malli (selaimet ja nettipalvelin)

Selain ja Nettipalvelin

- Nettisivun osoitteen muodoksi sovittu http://[palvelinosoite]/[sivu] esim. http://www.cse.tkk.fi/fi/index.html
- Sivulle tarkempi termi on **resurssi** (resource)
- Kun menet sivulle http://x/y
  1. Selain lähettää viestin palvelinsolmulle x, jossa pyydetään sivua y
  2. Palvelin x lähettää vastauksen jossa sivun y sisältö
  3. Selain näyttää sivun sisällön
  - Web 2.0 sivustolla ”näyttää” tarkoittaa usein ”suorittaa”

Selaimen viestirajapinta



HTTP viestirajapintana

- Dokumenttien hakeminen tehdään HTTP-standardin mukaisilla viesteillä
- HTTP = Hypertext transfer protocol (RFC 2616)
- Viestissä komento ja resurssi
  - Myös selaimeen liittyvää tietoa **otsakkeissa** (headers)
- Paluuviestissä tilannekoodi (status code) ja resurssin sisältö
  - Myös otsakkeita joissa resurssin metadataa
- Tärkein viesti on GET + resurssi

HTTP Esimerkki

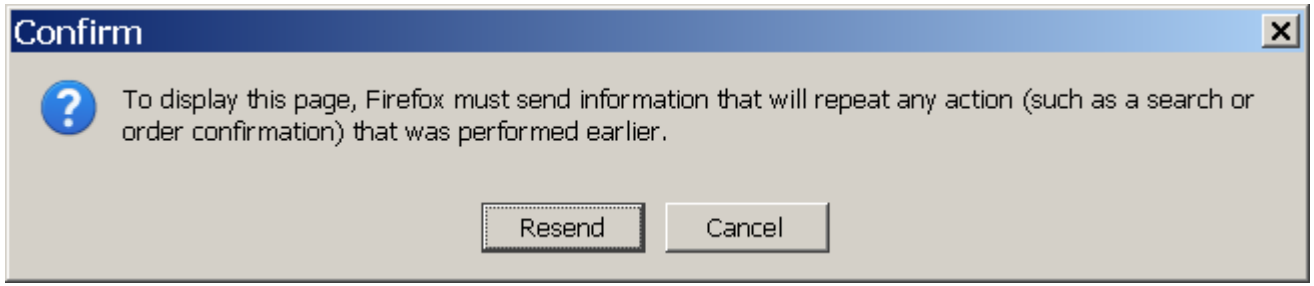
Parametri	Englanniksi	Arvo
Komento	Method	GET
Reurssi	Resource	/fi/index.html
Selain	User-Agent	Mozilla/1.0

Parametri	Englanniksi	Arvo
Tilakoodi	Status Code	200 OK
Pituus	Content-Length	11
Sisältö	Content	Hello World

- Huom: tässä mielletään viestit tietorakenteina (olioina) eikä mietitä missä muodossa ne esitetään siirtokerroksella. Tästä lisää ensi viikolla

HTTP

- Yleisimmät komennot
  - `GET` x: hae resurssi x sisältö
  - `PUT` x,y: tallenna data y resurssiin x (lue GETilla)
  - `POST` x,y: kirjoita resurssiin x **liittyen** data y (lomakkeen lähetys)
  - `GET` voidaan toistaa useamman kerran ilman sivuvaikutuksia, muita ei



HTTP

- Yleisiä statuskoodeja
  - 200 OK: kaikki meni hyvin
  - 404 Not found: resurssia ei löydy
  - 302 Moved temporarily: resurssi löytyy uudesta osoitteesta
- Yleisiä otsakkeita
  - Content-Length: resurssin koko tavuina (vastausviestin alussa)
- Muita määritetty RFC 2616

Selaimen käyttämät kielet

- Sivukuvauskieli
  - Tärkein HTML, Hypertext Markup Language
  - Myös XML-pohjaiset kielet XHTML, SVG (vektorigrafiikka), MathML (matemaattiset kaavat)
  - Laajennuksina sivuun upotetut apuohjelmat, jotka näyttävät muita datamuotoja
    - Adobe Flash
    - Java Applets
- Ohjelmointi
  - Javascript (Jscript), VB, ...
  - Javascript ohjelmakoodi HTML sivukuvauksen seassa

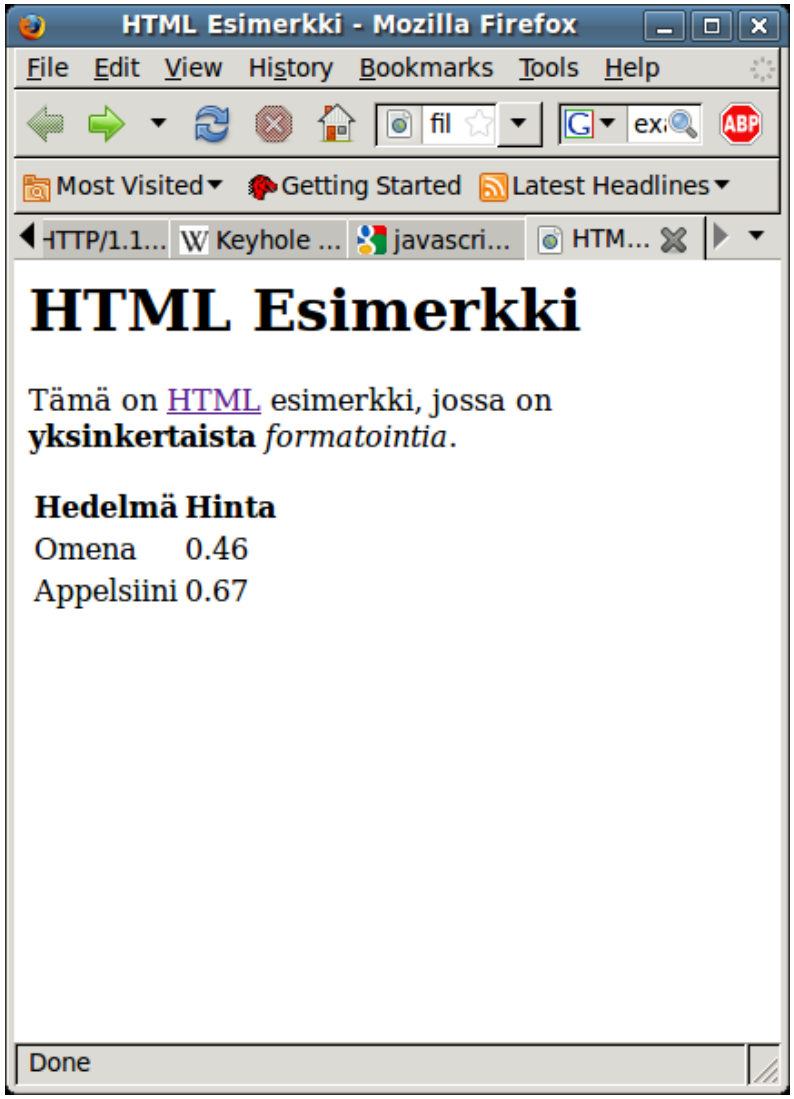
HTML

- Sivukuvauskieli, jossa tekstin ladonta (formatointi) ilmoitetaan tekstin seassa olevien **tägien** (tag) avulla
- Ladotun tekstin ”lähdekoodi”. TeX/LaTeX toinen esimerkki tästä
- Tägi muotoa <tag>...</tag>, ja se vaikuttaa alku- ja lopputägien väliseen tekstiin
- Tägiin voi liittää erinimisiä attribuutteja, joilla voi olla arvoja

  <tag attrib1=”value1” attrib2=”value2” ...>
- Tägin sisällä voi olla muita alku-/lopputägipareja
  - Tämä on väärin: Hello (sisällä ei ole tägipari)
- Standardoitu W3C:ssä ”HTML 4.01 Specification"-dokumentissa
- Kirjava historia, Netscape ja Microsoft ”browser war”, myöhemmin jonkinlainen yhteinen määritelmä

HTML

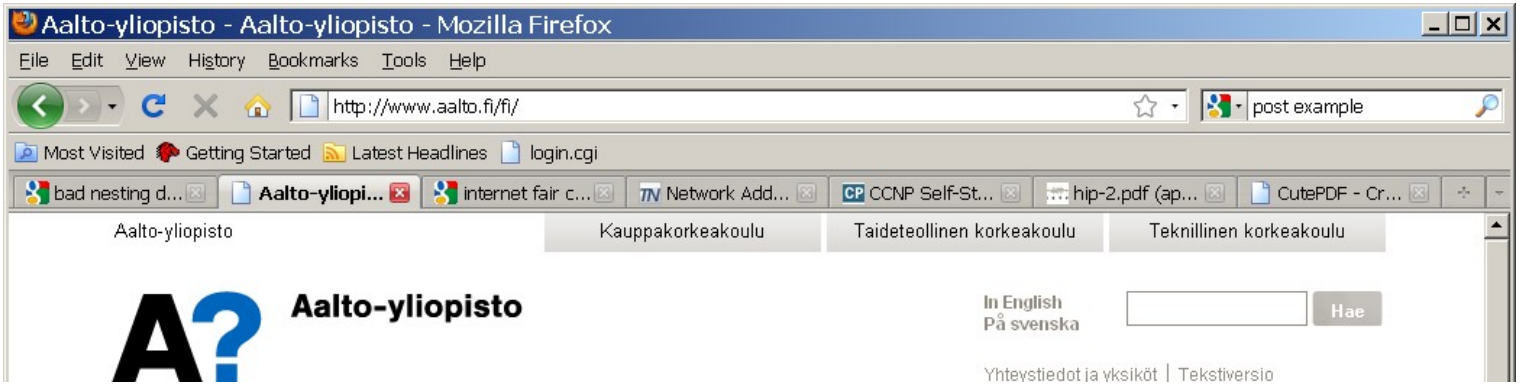
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
  <head>
    <title>HTML Esimerkki</title>
  </head>

  <body>
    <h1>HTML Esimerkki</h1>

    Tämä on
    <a href="http://www.w3c.org/TR/HTML">
              HTML</a> esimerkki,
    jossa on <b>yksinkertaista</b>
              <i>formatointia</i>.<p>

    <!-- Tämä on HTML-kommentti.
                          Alla on taulukko -->
    <table>
     <tr><th>Hedelmä</th><th>Hinta</th></tr>
     <tr><td>Omena</td><td>0.46</td></tr>
     <tr><td>Appelsiini</td><td>0.67</td></tr>
    </table>
  </body>
</html>
```



Tämä oli Web 1.0

- Seuraat linkkiä/kirjoitat osoitteen www.aalto.fi/index.html

1. Selain lähettää viestin GET index.html palvelimelle www.aalto.fi
2. Palvelin www.aalto.fi lähettää paluuviestin, jossa index.html-sivun HTML-koodi
3. Selain tulkitsee HTML koodin ja näyttää sivun



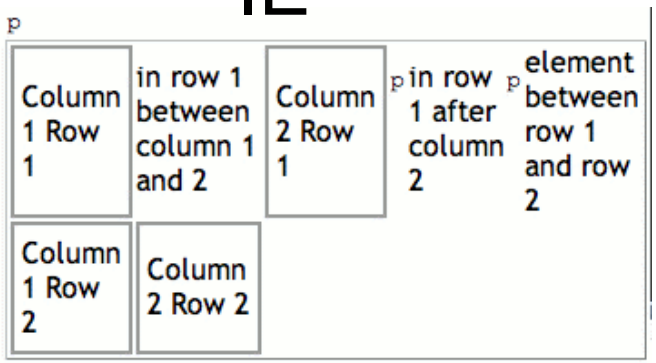
HTML:stä XML:ään

- Selaimet hyväksyvät ”rikkinäiset” HTML-sivut joissa on kielioppivirheitä
  - → Paljon HTML:ää jossa virheitä (kun kerran ei tarkisteta)
  - – Eri selaimissa eri virhetilanteiden käsittely, rikkinäiset sivut näkyvät eri lailla
- HTML koettiin ”helpoksi”
- Tarve kuvata muuta dataa kuin tekstiä
- Voisiko kuvata yleistä dataa HTML:n helppoudella, mutta ilman moniselitteisyyttä?

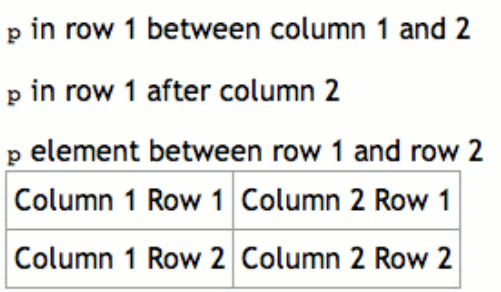
HTML:n ongelmia

```
<table summary="Example table containing unexpected elements"
border="1">
  <tr>
  <td>Column 1 Row 1</td>
  <p><code>p</code> in row 1 between column 1 and 2</p>
  <td>Column 2 Row 1</td>
  <p><code>p</code> in row 1 after column 2</p>
  </tr>
  <p><code>p</code> element between row 1 and row 2</p>
  <tr>
  <td>Column 1 Row 2</td>
  <td>Column 2 Row 2</td>
  </tr>
</table>
```

IE



Firefox

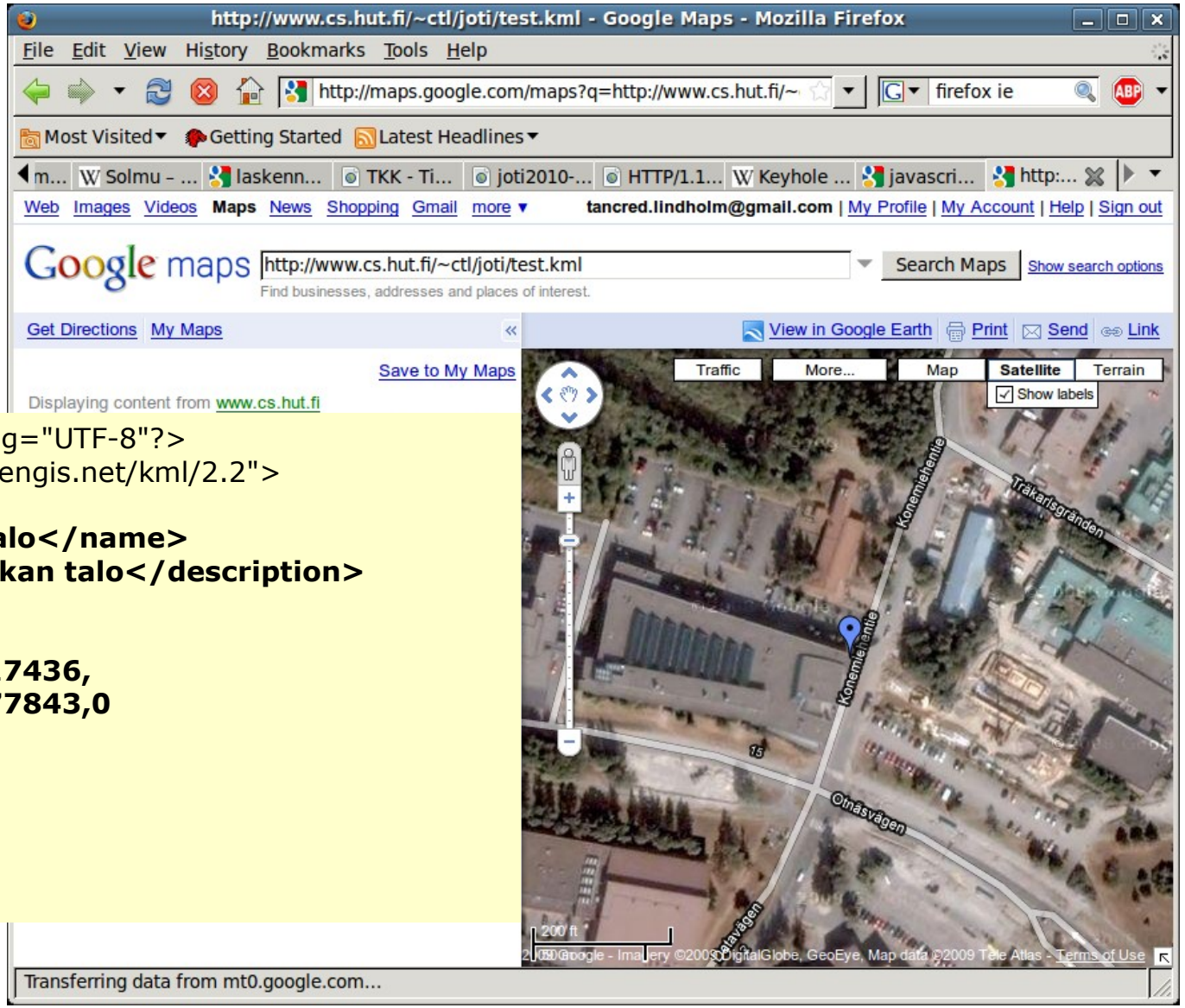


Lähde: http://www.illumit.com/Products/weblight/samples/markup_problems.shtml

Extensible Markup Language (XML)

- W3C esitti 1998 XML-kielen (pohjautuu SGML:ään)
- HTML-tyyliset tägit, jossa tägien **merkitys vaihtelee sovelluksen mukaan**
- XML:ssä sovittu että kielioppivirheitä ei hyväksytä
- XML nykyään jonkinlainen data yleinen esityskieli
  - Tekstipohjainen, tägien nimet helpottavat ymmärtämistä
  - Jokaisella alalla tarvitaan kuitenkin sopimus tägien merkityksestä (semantiikasta)
  - esim. tekstinkäsittelystä: ODT, Open Document Text file, KML karttapisteiden merkitsemiseen
- XML-kieli jossa on HTML:n tägit (ja merkitys) on nimeltään XHTML. Nykyajan selaimet tukevat tätä HTML:n rinnalla.

XML Esimerkki



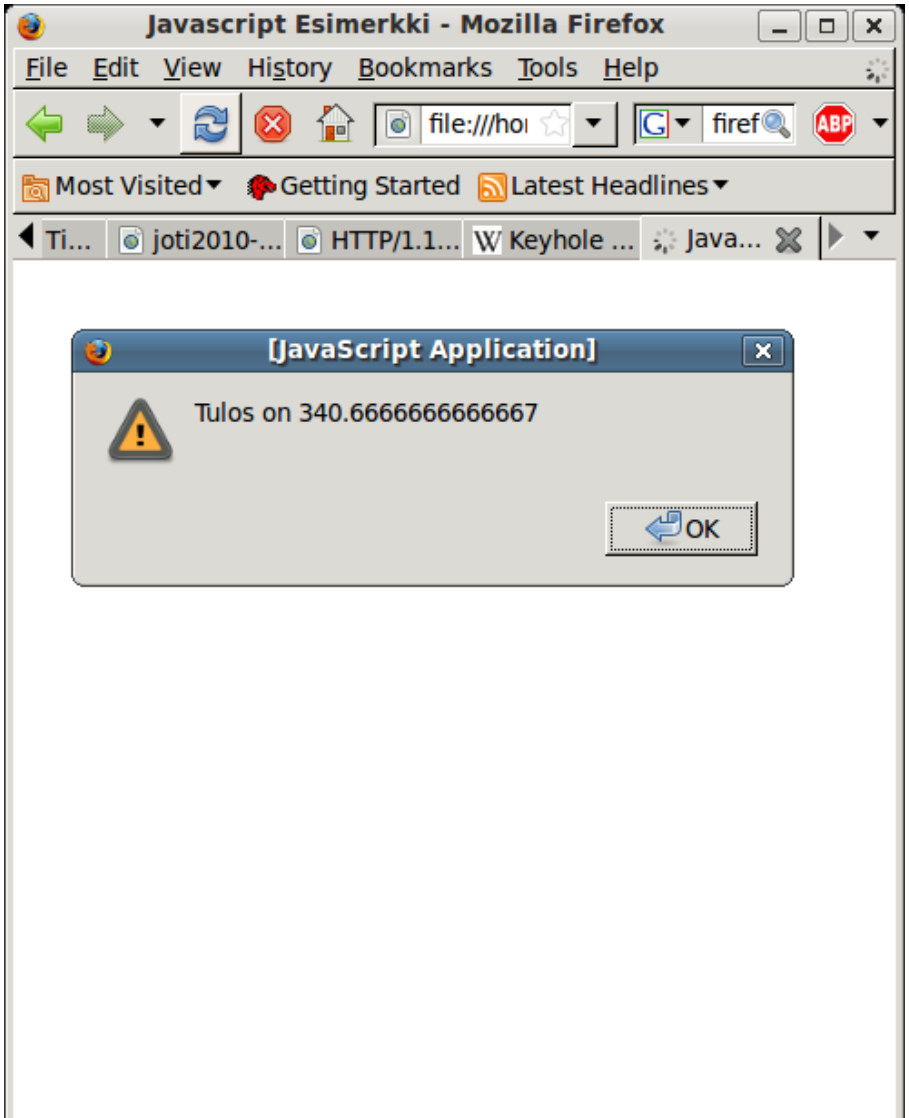

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Placemark>
  <name>Tietotekniikan talo</name>
  <description>Tietotekniikan talo</description>
  <Point>
    <coordinates>
              24.82248931417436,
              60.18696957477843,0
    </coordinates>
  </Point>
</Placemark>
</kml>
```

JavaScript

- Haluttiin dynaamisia elementtejä nettisivuihin
- Netscape kehitti 1995 JavaScript-nimisen kielen, ja lisäsi Netscape-selaimeen sille tulkki
- HTML-kieleen lisättiin <script> tägi, jonka sisällä olevaa Javascript-koodia **selain** suorittaa
- JavaScriptillä ei ole **mitään** tekemistä Javan kanssa, nimi valittiin markkinointisyistä
- Microsoft esitti tämän jälkeen Jscript-kielen jossa (lähes?) identtinen kielioppi
- C-tyylinen kielioppi
- Ajonaikainen tyypitys (”duck typing” – if it quacks like a duck....)

JavaScript Esimerkki

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD
HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Javascript
           Esimerkki</title>
  </head>

  <body>
    <script>
      var lasku = 2*137+200/3;
      alert('Tulos on ' + lasku);
    </script>
  </body>
</html>
```



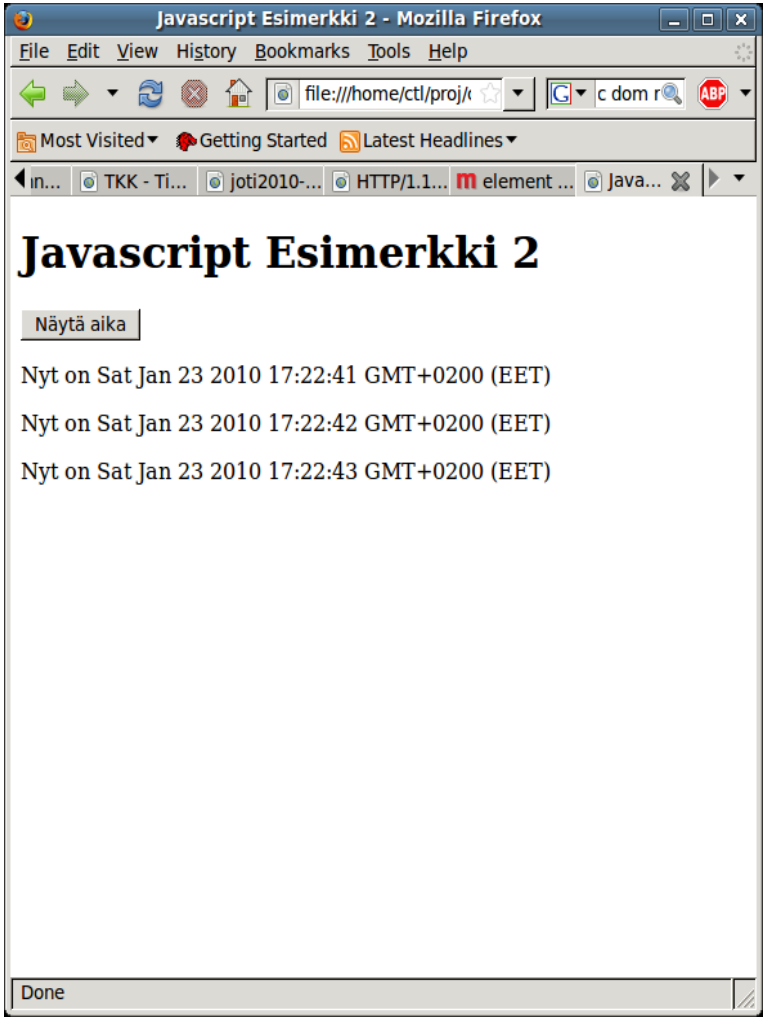
Javascript ja Selain

- Javascriptillä pystyy manipuloimaan HTML-sivun rakennetta
- Näin saadaan dynaamisesti muuttuva sivu!
- Javascriptissä HTML-sivu manipuloidaan DOM (Document Object Model) rajapinnan avulla
- HUOM: Seuraavassa jonkin verran koodia. Tärkein tässä on ajatus, eli **koodin upottaminen HTML-sivuun**, Javascript kielioppi ei ole tärkeä.
- DOM = oliomalli HTML-sivulle
- Oliomallin pikakurssi
  o.m(p1,p2,...) = sovella metodi m oliolle o parametreillä p1,p2,...

  Esim. document.write('Hello World') = Kirjoita (write) teksti "Hello World" (parametri) olioon document (HTML sivu joka selain näyttää)

Javascript DOM-esimerkki

```
...
  <body>
    <script>
     function buttonPressed() {
       // Date-olio jonka arvo on tämä hetki
       var pressTime = new Date();
       // luo uusi <p> tägi
       var paragraph =
          document.createElement('<p>');
       // <p> tägin sisälle tekstiä
       var paragraphText =  document.
          createTextNode('Nyt on ' +  pressTime);
       // Lisää uusi tägi dokumenttiin
       paragraph.appendChild(paragraphText);
       var divNode =  document.getElementById('messages');
       divNode.appendChild(paragraph);
     }
    </script>
    <h1>Javascript Esimerkki 2</h1>
    <form>
      <input type="button" onclick="buttonPressed();"
          value="Näytä aika">
    </form>
    <div id='messages'>
      <!– lisätään aikaleimat tähän tägiin sisään -->
    </div>
  </body>
</html>
```

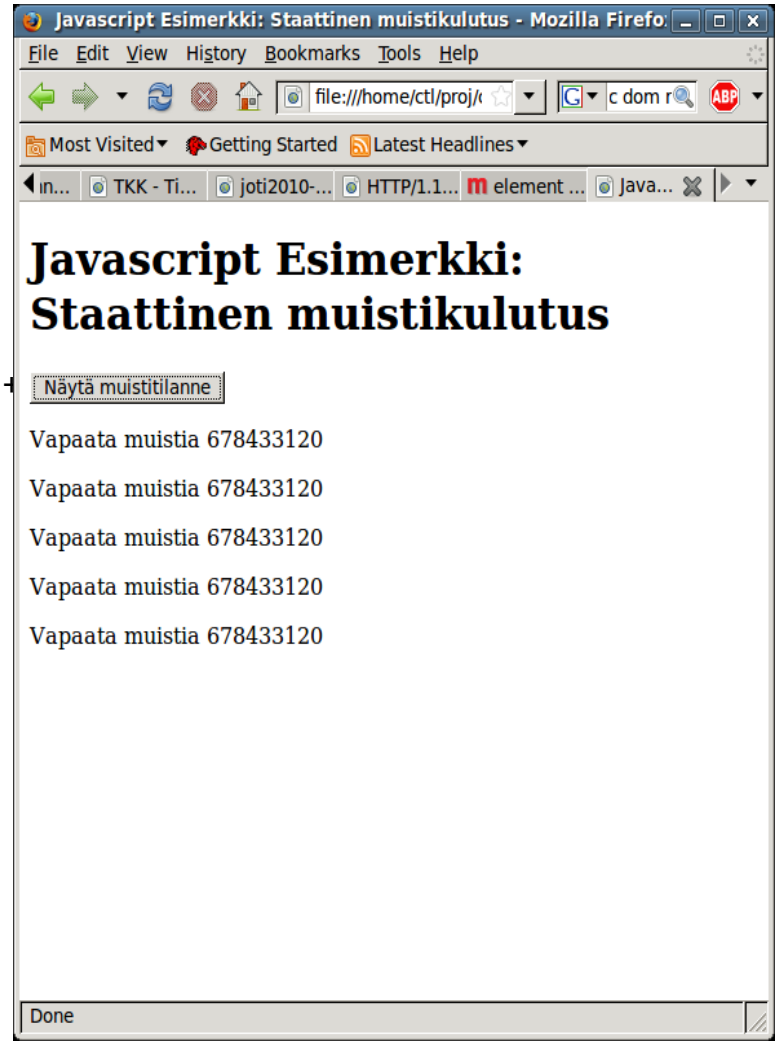


Javascript ja HTTP viestit

- Javascriptillä voi nykyään lähettää palvelimelle HTTP-pyyntöjä ja käyttää vastausta sivun päivittämiseen
- Tämä on Web 2.0 perustekniikoita
- Huom: perinteisesti (Web 1.0) HTTP-pyyntöjä ainoastaan kun siirryttiin sivusta toiseen
  - Sivua päivitettiin lataamalla se kokonaan uudelleen palvelimelta
  - Hidasta, huono käytettävyys, resurssien tuhlausta
  - Toisaalta, paljon helpompi malli ohjelmoida

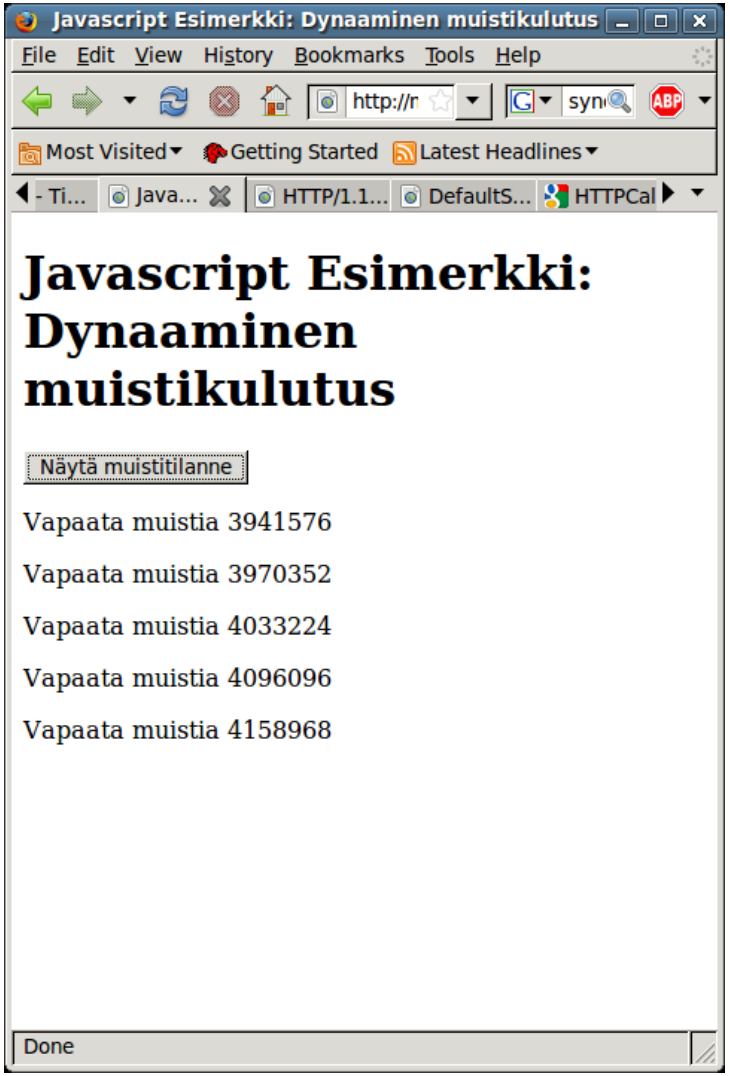
Esimerkki: Staattiset resurssit

```
....
  <body>
    <script>
      function getFreeMemory() {
        return 678433120; // Palvelin täytti tämän kun sivua haettiin
      }
      function buttonPressed() {
        var paragraph = document.createElement('<p>'); // luo kappale
        var paragraphText = document.createTextNode('Vapaata muistia ' +
          getFreeMemory());
        paragraph.appendChild(paragraphText);
        var divNode = document.getElementById('messages');
        divNode.appendChild(paragraph);
      }
    </script>
    <h1>Javascript Esimerkki: Staattinen muistikulutus</h1>
    <form>
      <input type="button" onclick="buttonPressed();"
                    value="Näytä muistitilanne">
    </form>
    <div id='messages'>
      <!-- Muistiviestit tänne -->
    </div>
  </body>
</html>
```



Esimerkki: Dynaamiset resurssit

```
...
   <script>
    function getFreeMemory() {
      var xhr = new XMLHttpRequest();
      xhr.open('GET', '/statz', false);
      xhr.send(null);
      return xhr.responseText;
    }

    function buttonPressed() {
      var paragraph = document.createElement('<p>'); // luo kappale
      var paragraphText = document.createTextNode('Vapaata muistia '
+
        getFreeMemory());
      paragraph.appendChild(paragraphText);
      var divNode = document.getElementById('messages');
      divNode.appendChild(paragraph);
    }
   </script>
   <h1>Javascript Esimerkki: Dynaaminen muistikulutus</h1>
   <form>
    <input type="button" onclick="buttonPressed();"
                value="Näytä muistitilanne">
   </form>
   <div id='messages'>
    <!-- Muistiviestit tänne -->
   </div>
  </body>
</html>
```



Javascript Demo

Etäkutsut (RPC) ja kolmannen osapuolen palveluntarjonta

- Edellisissä esimerkeissä ainoa ero oli getFreeMemory()-metodin toteutus
- Hajautetussa järjestelmässä mahdollista että ”tavallinen” metodi on toteutettu verkkoviesteillä
- Tällöin on kyseessä ns. etäkutsu (remote procedure call)
- Jotkut palveluntarjoajat (esim Google) tarjoavat Javascript-rajapinnan, jonka alla on etäkutsut kyseiseen palveluun (esim. nettihaku)

```
<script src="http://www.google.com/jsapi"
type="text/javascript"></script>
<script>
        var searchControl = new google.search.SearchControl()
        searchControl.execute("hakusana");
    // Näytä tuloksia
</script>
```

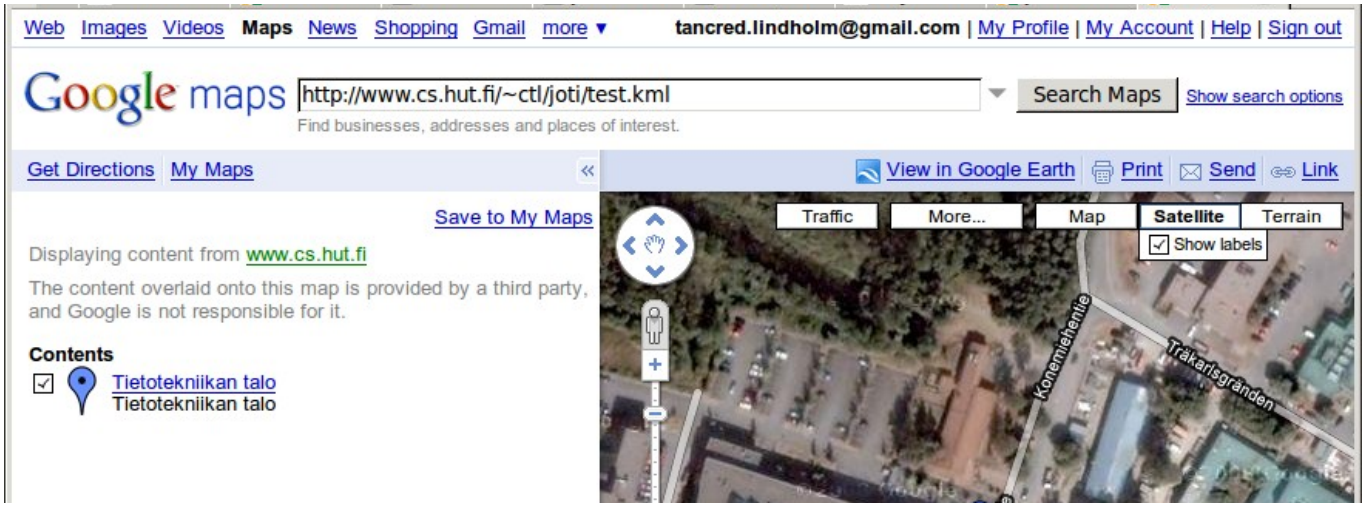
Kertaus: Web 2.0

- Seuraat linkkiä/kirjoitat maps.google.com/index.html

1. Selain lähettää viestin GET index.html palvelimelle maps.google.com
2. .... Selain tulkitsee HTML koodin ja näyttää sivun

Web 1.0

3. Käytät maps.google.com sivun interaktiivisia toimintoja, esim painat "Satellite" nappia, josta käynnistyy
4. sivussa olevaa Javascript-koodia, joka hakee etäkutsuina sivulle uutta sisältöä (esim. satelliittikartta)



HTTP Palvelin

- Entäs palvelin, miten se toimii?
- Yleiskäyttöinen ohjelmisto joka tulkitsee sisääntulevat viestit ja muuttaa ne ohjelmakielen tietorakenteiksi
- Palveluntarjoaja kirjoittaa ohjelman joka käsittelee sisääntulevat viestit tietorakenteina
- Java-kielellä yksi yleinen palvelinohjelmisto on Java Servlets
- Esimerkkinä demossa näytettiin palvelu joka palautti palvelimen vapaan muistimäärän

HTTP palvelin, esimerkki

```
void doGet(HttpServletRequest httpServletRequest,
                    HttpServletResponse response) {
            String freeMem = Runtime.getRuntime().freeMemory();
            response.setHeader("Content-Length", freeMem.length());
            response.setHeader("Content-Type", "text/plain");
            dataOut.write(freeMem);
   }
```

*) Tässä jätetty pois muutama tyyppimuunnos

Maatuska (матрёшка)

- Jatkokurssia....
- Ohjelmia jotka kirjoittavat ohjelmia
- Tavallista nettisovelluskehityksessä
- Vaikeuttaa ymmärtämistä
- Perinteisesti ohjelmia, jotka käsittelevät dataa
- Nettimaailma: ohjelmia, jotka käsittelevät dataa, joka on toisen ympäristön ohjelmaa ja dataa, joka on toisen ympäristön ohjelmaa ja dataa...
- Esim: Java-palvelinkoodi, jonka sisällä HTML, jonka sisällä JavaScript, jonka sisällä HTML

```
out.print("<script>node.innerHTML =
'It\\'s a link \\<a href=\"http://www.example.com\"\\>';");
```

Näillä eväillä...

- Asiakas/palvelin hajautettu järjestelmä
  - → Muut viestintätopologiat, P2P
- Selaimen ohjelmointia
  - → AJAX
- Palvelinohjelmointia
  - → JSP, Ruby on Rails, Cloud computing
- Palveluiden suunnittelu Internet-ympäristöön
  - → REST-arkkitehtuurityyli

Vertaisverkot (P2P)

- **Asiakas/Palvelin (client/server)** - nettiselailussa on monta asiakasta (client) jotka ottavat yhteyttä yhteen palvelimeen (server)
- **Vertaisverkko (peer-to-peer)** - toinen tapa toteuttaa hajautettu järjestelmä on käyttää monta samankaltaista solmua (peer) jotka keskustelevat toistensa kanssa ilman tiettyä topologiaa
- Näistä parhaiten tunnettuja ovat tiedostojen jakamiseen tarkoitetut ohjelmistot kuten BitTorrent
- Vertaisverkoissa ei tarvita kallista palvelininfrastruktuuria

AJAX

- AJAX = Asynchronous Javascript and XML
- Nimitys tekniikalle jolla toteutetaan dynaamiset sivut
- Esimerkkinä Facebook, sekä useimmat modernit sivustot
- Perusteet tästä nähtiin dynaamisessa resurssikulutusesimerkissä
- Tämän lisäksi
  - Asynkronisuus = suoritetaan Javascript-koodia reaktiona siihen että vastaus saapuu, eikä jäädä odottamaan vastausta (kuten esimerkissä)
  - Käytetään XML (ja myös usein ns. JSON) -muotoista dataa vastauksessa (esimerkissä ei-rakenteellinen merkkijono

Palvelinohjelmointia

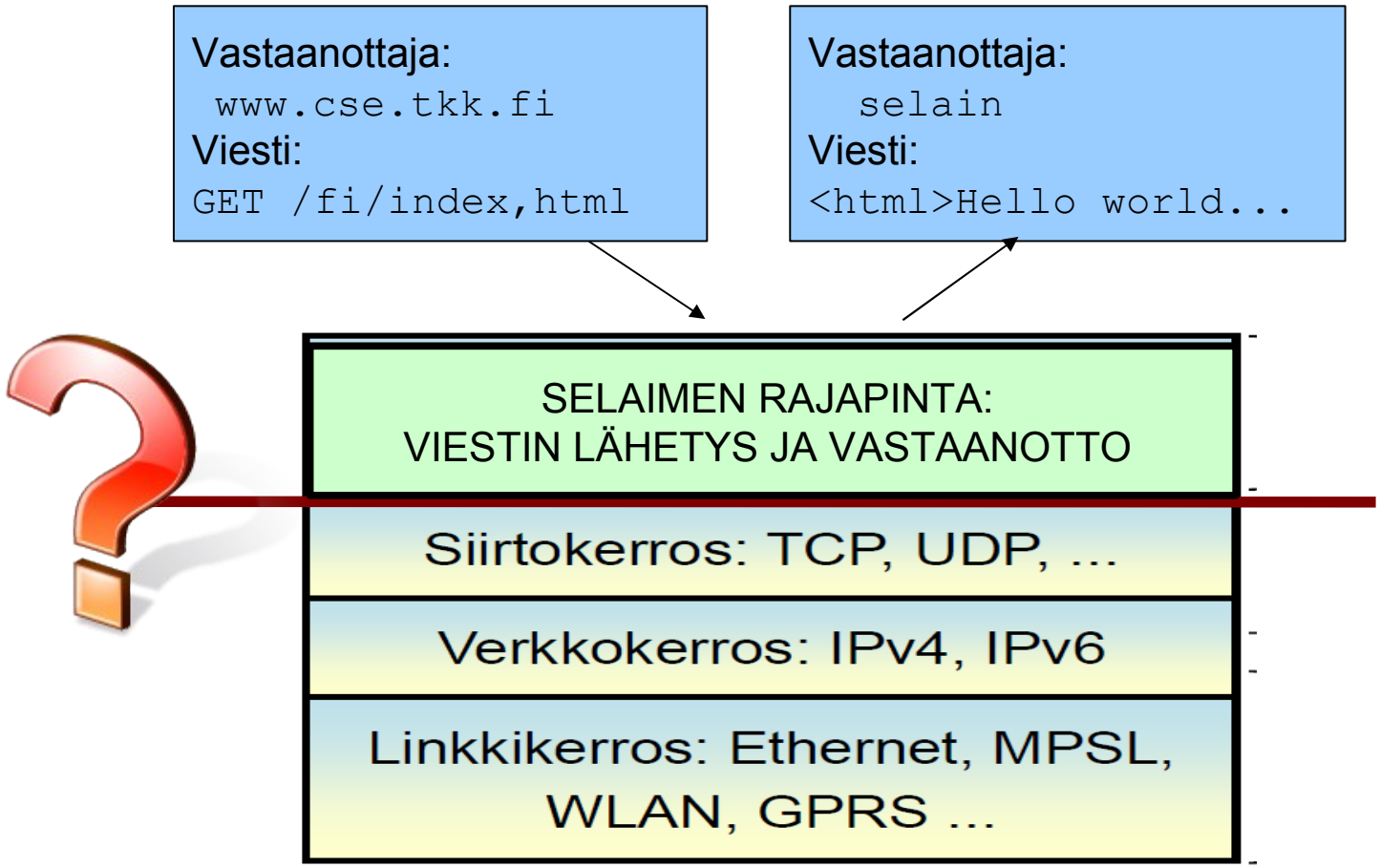
- Internet on suuri ja paha paikka
  - Miten kirjoittaa palvelinohjelmisto joka käsittelee 100.000 hakua/s?
  - Miten kirjoittaa palvelin s.e. ainoastaan oikeilla käyttäjillä on pääsy sivustoon?
  - Miten hallita ja päivittää järjestelmä, joka pitää toimia 24/7, ja joka käyttää jatkuvasti muuttuvia kolmannen osapuolen palveluja?

REST

- Osittaisen vastauksen antoi Roy Fielding väitöskirjassaan
- Internet-järjestelmien arkkitehtuurissa noudatetaan muutamaa hyväksi todettua rajoitetta
  - Viestitään täydellisiä tiloja avoimissa muodoissa
    - (esim. kokonainen dokumentti XML:nä eikä dokumentin osia suljetussa formaatissa) (Representational State)
  - Yhdenmukainen rajapinta (Uniform Interface)
  - Muita (Layering, Caching, Code-on-demand)
- Nämä rajoitteet tunnetaan nimellä REST (Representational State Transfer)

Ensi luennolla

- Tämän päivän rajapinta: viestit
- Mitä löytyy toiselta puolelta?



Yhteenveto

- Hajautettu järjestelmä josta tärkeä esimerkki netti
- Nettiselaimen ohjelmointikielet ja ohjelmointi
- Hajautettu ohjelmointi, josta tärkeä esimerkki AJAX
- Nettipalvelin
- Perusymmärrys muutamasta hypesanasta
- Voidaan kertoa omalle koiralle/kissalle miten Facebook toimii

Ekstra: staattinen sivu

- Nähtiin esimerkkisivu jossa staattinen muistimäärä
- Vapaa muistimäärä sivun HTML/Javascript-koodissa
- Palvelinohjelmisto joka generoi tämän sivun voisi näyttää esim. tälläiseltä
- Huom: Java-koodia, joka tuottaa Javascript-koodia (Maatuska-periaate)

```
void doGet(HttpServletRequest httpServletRequest,
                    HttpServletResponse response) {
  ...
  out.println("<body>");
  out.println("   <script>");
  out.println("     function getFreeMemory() {
  out.println("       return " + Runtime.getRuntime().freeMemory()  + ";")
  out.println("      }");
  ....
```