Mobile Computing

HTML5

History

- 1997 – HTML 4.0
- 1999 – XHTML 1.0 (Appendix C allows for backwards compatibility)
- 2001 – XHTML 1.1 (No backwards compatibility with HTML)
- 2004 - Web Hypertext Applications Technology (WHAT) WG is created, outside W3C, to extend HTML
- 2006 – WHAT WG rejoins W3C. Renames their work HTML 5
- 2009 – XHTML WG ends

HTML error handling

- XHTML was created because 99% of the world's webpages are believe to contain errors. Browsers are forgiving but each one handles the errors differently.
- Developers didn't adopt XHTML 1.1
- WHAT WG documented browser behaviour and defined how errors should be handled without presenting error messages to the user.

HTML 5 new features

- HTML 5 support is incremental.
- It is still being developed and each browser implements a different subset.
- Applications should test the availability of a feature.
- Major features include:
  - Canvas for drawing
  - Video and sound support
  - Local storage (key/value and SQL database)

  ...

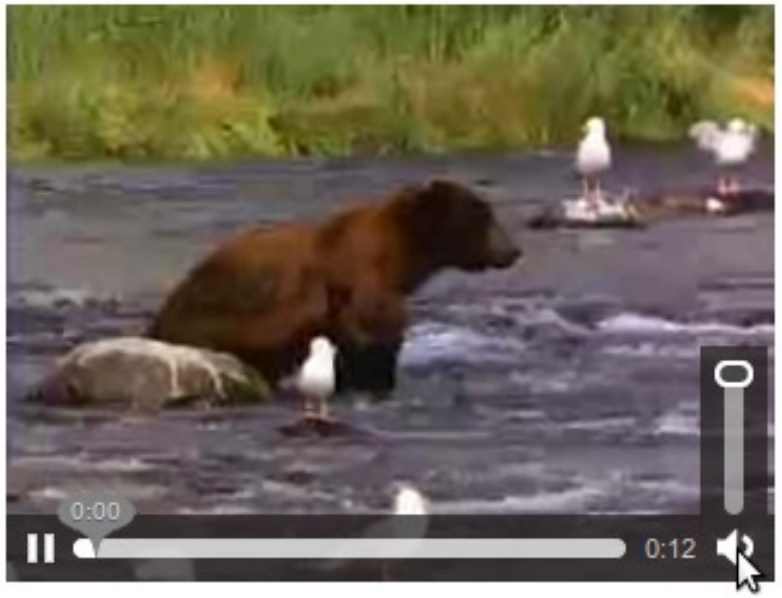
HTML 5 new features

- New features (continued)
  - New content specific elements (article, footer, header, nav, section)
  - New form elements (calendar, date, time, email, url, search)
  - Workers (threads for concurrent work)
  - Websocket
  - Work offline
  - Geolocation
- HTML 5 defines new elements, new DOM objects and corresponding JavaScript APIs

Video tag

- Video playback without plugins
- Ogg, MPEG4 and WebM formats
- Support varies among browsers
- Controllable using JavaScript
- Some declarative control: loop, hide controls, size, preload, autoplay

```
<video width="320" height="240" controls="controls">
  <source src="movie.ogg" type="video/ogg" />
  <source src="movie.mp4" type="video/mp4" />
  <source src="movie.webm" type="video/webm" />
Your browser does not support the video tag.
</video>
```



Workers

- Background tasks for multithreaded behaviour
- Allow server access and complex computation without stopping the user interface
- Communication using messages (callback methods)

```
var n = 1;
search: while (true) {
  n += 1;
  for (var i = 2; i <= Math.sqrt(n); i += 1)
    if (n % i == 0)
     continue search;
  // found a prime!
  postMessage(n);
}
```

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Worker example: One-core
computation</title>
 </head>
 <body>
  <p>The highest prime number discovered so far
is: <output id="result"></output></p>
  <script>
   var worker = new Worker('worker.js');
   worker.onmessage = function (event) {

document.getElementById('result').textContent =
event.data;
   };
  </script>
 </body>
</html>
```

Example from http://dev.w3.org/html5/workers/

websocket

- Two way communication between browser and server.
- Allows for push model. Server sends data when it is needed. More efficient than long wait.
- Uses HTTP like protocol over TCP. Protocol allows (future) HTTP servers to handle request differently.
- After initial handshake, client and server exchange messages
- WebSocket API is being developed by W3C, WebSocket protocol by IETF

Local storage

- Cookies allow data to be stored on browser but:
  - Data must be transferred in every request
  - Were designed for client identification by the server. Server holds the data.
- HTML 5 provides objects for storing key/value pairs permanently or for the duration of the session.
- Important for mobile and offline applications

```
<script type="text/javascript">
if (localStorage.pagecount)
  {
  localStorage.pagecount=Number(localStorage.pagecount) +1;
  }
else
  {
  localStorage.pagecount=1;
  }
document.write("Visits "+ localStorage.pagecount + " time(s).");
</script>
```

SQL database

- JavaScript API for storing data in a database within the browser.
- SQLite embedded database engine is used by some browsers

```
db = openDatabase('WebNotes', '1.0',
'WebNotes', 524288);

db.transaction(
  function(transaction) {
   transaction.executeSql(
    'CREATE TABLE IF NOT EXISTS ...');
  }
 );

db.transaction(
  function(transaction) {
   transaction.executeSql(
    'SELECT id,name from ...;', null,
    function(transaction, result) {
     for (var i=0; i<result.rows.length;i++){
      var row = result.rows.item(i);
      var id = row.id;
          ...
     }
    }, errorHandler);
  }
 );
```

Offline mode

- Offline webapp is a set of resources which are stored by the browser
- Manifest file lists resources. They are only refreshed if the manifest file changes.
- JS API allows webapp to detect if browser is online or offline, and when applications change.

Look & Feel APIs

- JQuery mobile framework

http://jquerymobile.com/

- jQTouch – jQuery plugin for iPhone look & feel

http://www.jqtouch.com/

- Sencha Touch – HTML 5 library for Android & iOS native look & feel

http://www.sencha.com/products/touch/

Today's class assignment

- Deploy the notes demo from[1] the course page. It uses offline mode, SQL database and jQTouch for user interface
  - Download the code and deploy it in an apache webserver (use sigma.ist.utl.pt)
  - Access the webapp with a desktop browser
  - Run the Android Virtual Device (emulator). Use the browser to access the webapp.
    - Alameda - /usr/lib/rnl-m2-cmov-plugins/android-sdk
    - TagusPark - /opt/android-sdk-linux

1 - Rick Rogers, "Developing Portable Mobile Web Applications", Linux Journal #197, September 2010

Today’s class assignment

- Sencha library examples
  - Guide & Downloads:
    http://docs.sencha.com/touch/2-0/#!/guide/getting_started

  Unzip to a webserver
  - Use the AVD's browser to access the webpage
  - Look at the examples

Today's class challenge!

- Turn the notes webapp into a todo list, where task are listed according to due date. Overdue and completed tasks should be shown differently

Useful links

- Dive into HTML 5

http://diveintohtml5.info/

- W3Schools HTML5 Tutorial

http://www.w3schools.com/html5/default.asp

- Mobile Web Application Best Practices Cards

http://www.w3.org/2010/09/MWABP/

- HTML 5 Demos

http://html5demos.com/

- Remember, Google is your friend! ☺