Development of Internet Applications

HTML 5 annd CSS 3

Ing. Michal Radecký, Ph.D.

www.cs.vsb.cz/radecky

https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5

http://html5slides-1117.appspot.com

http://www.w3schools.com/html/html5_intro.asp

Storage

```
if(typeof(Storage)!=="undefined")
 {
 // Yes!
 }
else
 {
 // Sorry! No web storage support..
 }
```

- Cookies replacement
  - Data is not a part of each request
  - Possible to store huge set of data
  - Accessible only by author/web page
  - Event-driven model
- Principle – couple key/value (string)
- **LocalStorage** – data stored for unlimited time
- **SessionStorage** – data stored for limited time defined by one session
- Access by interface (object) or indexes (keys)

Web database

- Web SQL Database
    - API for data processing on client-side based on relation DB principles (SQL)
    - No longer supported as a part of HTML 5
    - Methods: *openDatabase*, *transaction*, *executeSQL*
- IndexedDB
    - Solution to store a huge amount of structured data
    - Fast searching based on indexing
    - Synchronous and asynchronous approach
    - Objective and transactional oriented, use couple key/value (object)
    - API interface: **indexedDB**

Off-line applications

- Off-line operation of web pages using caching
- Decreasing of demands of speed and data size
- Cache Manifest (text/cache-manifest)
    - Stand-alone file includes cache rules
    - CACHE – cache specified files for further usage
    - NETWORK – specified files are never cached
    - FALLBACK – replacement for non-cached files
- Update of files
    - Cleaning of cache repository
    - Programmatically
    - Cache manifest update

Web Workers

- Implementation of „Threads“ in web page environment – run the algorithm in the background without affecting interaction with the user
- External JS files are used for WebWorkers operation – synchronic approach
- Object **Worker**
- Worker works on global level, communication is based on events and messages (*postmessage* – *onmessage*)
- No access to native objects *window*, *document*, *parent*

Web Sockets

- Advanced interface for bidirectional asynchronous communication (client – server), each side can send message during the time
- Both side implementation is necessary
- Effective usage together with WebWorkers
- Object **WebSocket**
- Implementation of events *onopen*, *onmessage*, *onclose* and method *send*

Drag & Drop

- One of fundamental users features from the desktop app domain
- It is possible to move any content element – **draggable=„true“**
- Implementation of events *ondragstart*, *ondrop*, *ondragover*
- Work with (transmitted within events) object *dataTransfer.SetData* (*GetData*)

Drag-In (File API)

- Ability to move object (file) from local computer inside the web page content
- Based on Drag & Drop approach – event *ondrop* on specified element
- Access to moved content (file) via *DataTransfer.files* (similar to process input type „file“)
- File API offers objects **File, FileList, Blob, FileReader, URL**
- File API is suitable to work with files directly inside web page, cover also the reading of the file content (text, binary, Base64)

FileSystem API

- Extends File API capabilities to write to file (**BlobBuilder**, **FileWriter**) and their organization (**DirectoryReader, FileEntry/DirectoryEntry, LocalFileSystem**)
- Based on virtual file system inside the browser sandbox – access via method *requestFileSystem*
- Suitable for Binary data (temporary or persistent) – files upload, temporary storage, file content edit, off-line working

Geo-localization

- Possibility to obtain the GPS position of the user (latitude, longitude, altitude, accuracy, speed, timestamp)
- Necessity of user permission
- Based on technical capabilities od device (GPS, Wifi, IP address)
- Object **navigator.geolocation**
- Methods *getCurrentPosition* and *watchPosition*

```
if ("geolocation" in navigator)
{
  /* geolocation is available */
}
else
{
  /* geolocation IS NOT available */
}
```

Access to hardware

- Device orientation and position in environment
- Camera and microphone
- Voice input
- Gestures
- Full-screen mode
- etc.

Graphics

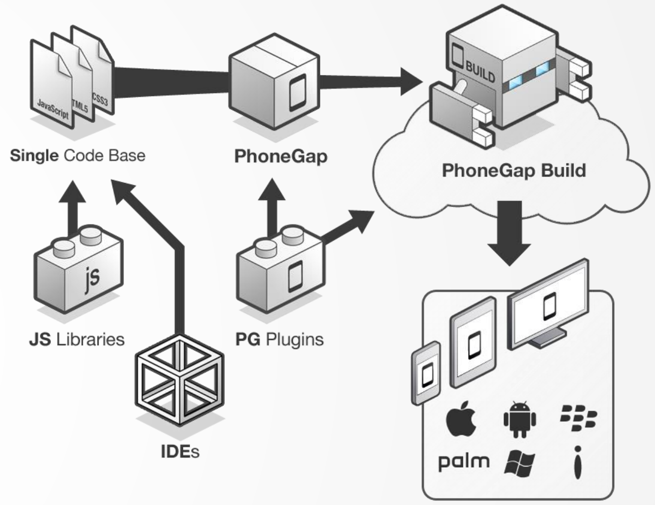
- **Bitmap graphics** – Canvas element
  - The context is operated over the element – method *getContext(„2d“)*
  - The context offers API for drawing, drawing is sequentiual
  - Animation uses methods *setTimeout* a *setInterval*. Most effective way is to use *requestAnimationFrame* – utilization of standard animation loop
- **Vector graphics** – SVG format
  - Modification of DOM – specific XML as a part of DOM
  - Ability to link visual components and CSS/JS
- **3D graphics** – WebGL technology
  - Context „webgl“
  - API is based on OpenGL approach

Specific data- attributes

- Possibility to store of specific, application related, data within standard HTML code
- Utilization of prefix **data-*** (these attributes are ignored)
- Access through property *dataset* of a given element
- Suitable for storing work or state values, settings, data for analysis, etc.

Mobile applications

- HTML 5 is suitable for implementation of native mobile apps – thanks to middleware
- Web app based on HTML5+JS+CSS is fundamental.
- It is extended by features offered by specific API (PhoneGap, Xamarin, etc.).
- The result is native cross-platform app – web browser with extended features as environment.
- The abstract layer (middleware) is used. It offers connection between app and HW/OS level.
- Camera, Geolocation, Compass, Contacts, Media, Accelerometer, Network, Notification, Storage, Filesystem



http://www.youtube.com/watch?v=wOH4aGows40