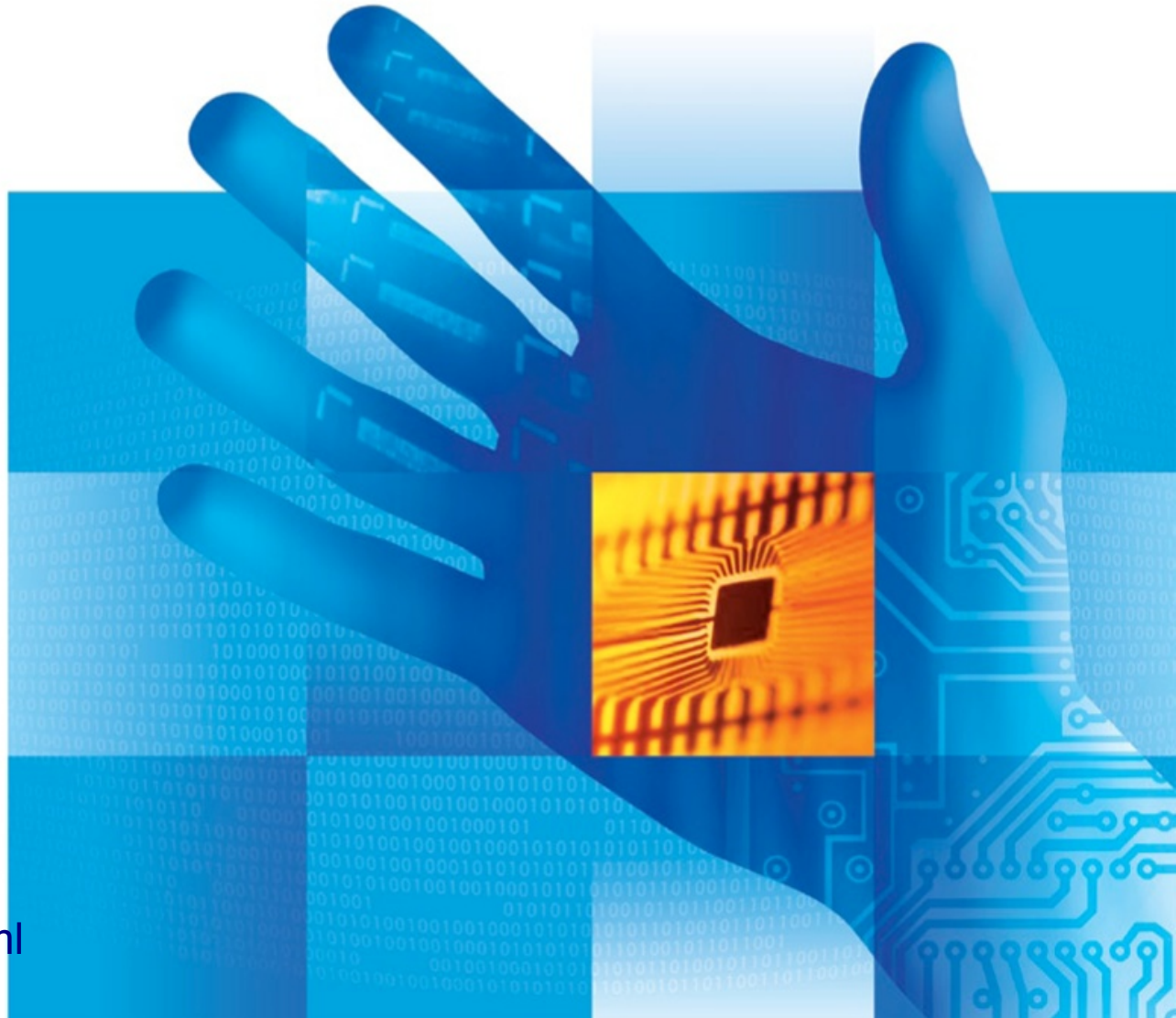




API detection AJAX (JSON) & SSE



TAMZ 1

Lab 6

See: <http://diveintohtml5.info/detect.html>

Platform & Device fragmentation

- Not all APIs and HTML-5 features are available on all platforms and/or devices
 - i.e. Android browser < Android 4.4 → no WebSockets
 - Unlike in JavaME, we can emulate some APIs in other ways → we use polyfills (web shims) in JavaScript instead
 - shim → application API compatibility workaround library
 - Apache Cordova may add native code and JS interface
 - Some sites compile compatibility and workaround lists
 - <http://html5please.com/>, <http://caniuse.com/>
- We need either to ignore missing APIs (disable parts of code) or provide a workaround
 - Server supplies browser-adjusted code directly
 - Detection is done on Client through JavaScript



API Detection on server

- Done by analyzing the sent HTTP headers and cookies
 - **User-Agent**: header for browser detection
 - e.g. in PHP: `$_SERVER['HTTP_USER_AGENT'];`
 - **Accept-Language**: detect user's preferred languages
 - e.g. to select first offered language in PHP:
`$lng = substr($_SERVER['HTTP_ACCEPT_LANGUAGE'], 0, 2);`
 - **Accept-Charset**: character sets in browser (nowadays: */*)
- May be used to supply required code modifications without client-side processing/detection
 - e.g. we can supply polyfills directly
- Some projects exist for existing frameworks
 - UA-parser framework for many programming languages, to extract information normally available in JS BOM
 - <https://github.com/tobie/ua-parser>
 - e.g. Detector (Beta) at <http://detector.dmolsen.com/> for PHP/JS, which is working with Modernizr (see later)

API Detection in JavaScript

1. Check if given API object is present:
 - if (`typeof(Storage) !== "undefined"`) { ... };
2. Check if a method is implemented/property is present:
 - if(`window.addEventListener`) { ... }
 - return `localStorage != null`
3. Check if given element is supported in a method:
 - var elem = `document.createElement('canvas');`
 - return `!!elem.getContext`; // !! – force true/false value
4. Check if some API which extends BOM is present:
 - if (`'geolocation' in navigator`) { ... } //or `navigator.geolocation`
5. Check if a built-in verification function returns true/false
 - if (`supports_video()`) { ... }
6. Check for input type support inside a function:
 - var i = `document.createElement("input");`
i.`setAttribute("type", "color");`
return `i.type !== "text";`



Is there an easier feature detection?

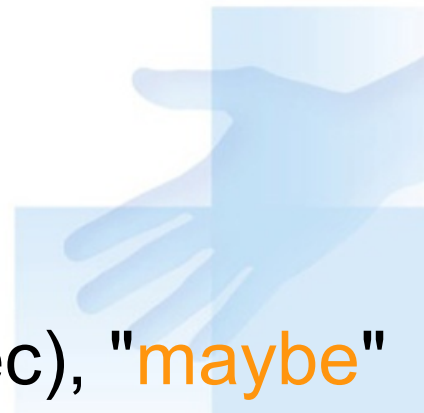
- Yes, there is – the **Modernizr** <http://www.modernizr.com/>
 - In <head> section of <html class="no-js"> add e.g.:
`<script src="js/modernizr.dev.js"></script>`
 - In your JavaScript code, you just write **Modernizr.feature**:
 - `if(Modernizr.geolocation) { ... }`
 - `if (Modernizr.input.placeholder) { ... }`
 - `if (!Modernizr.inputtypes.date) { ... }`
 - It is also possible to use Modernizr classes inside of CSS:
 - `MY_SELECTOR { ... }`
`.detected_class MY_SELECTOR { ... }`
 - Conditional script loading is possible (uses yepnope.js)
 - If you need detection of non-core features, you can use builder: <http://modernizr.com/download/>
- Tests to show the features may be executed in the browser, URL:
<http://j.mp/modernizrtest>

Extending Modernizr

- You can write your own detection routines/tests, e.g.

```
define(['Modernizr'], function( Modernizr ) {  
  Modernizr.addTest('eventsourcing', 'EventSource' in window);  
});
```
- You can conditionally load external script:
 - Polyfills (but some work only on desktop) available at: <https://github.com/Modernizr/Modernizr/wiki/HTML5-Cross-browser-Polyfills>
 - Usage of loading (includes yepnope.js):

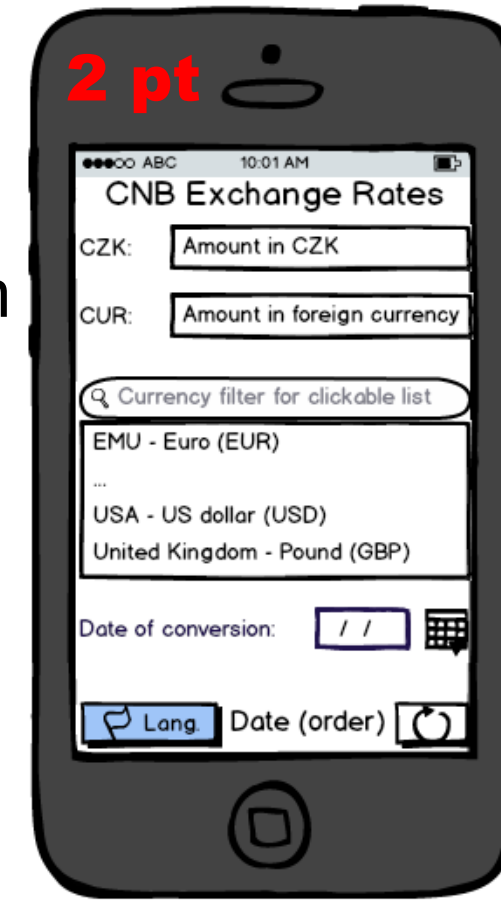
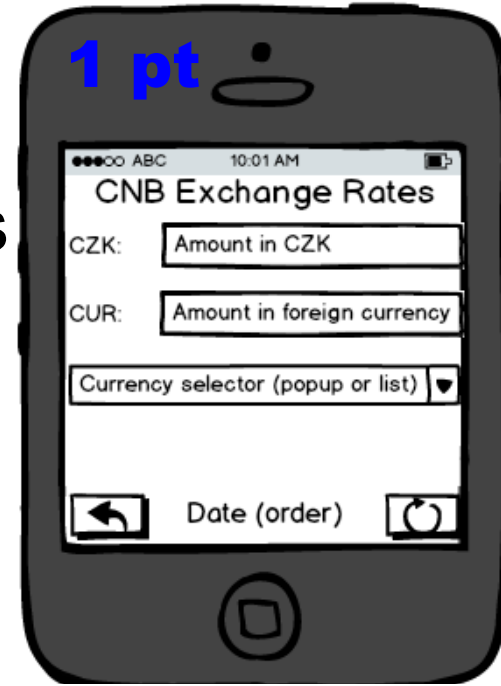
```
Modernizr.load({  
  test: Modernizr.svg,  
  yep : 'js/svg_images.js',  
  nope: 'js/png_images_polyfill.js',  
  both: [ 'js/script1.js', 'js/script2.js' ],  
  complete: function () { ... }  
});
```
- Detection results: **true**, **false**, "probably" (codec), "maybe"



Task (1-2.5 points)

Create application which will convert currencies on-the-fly (when something changes) based on current exchange rates provided by CNB (1pt):

- URL for downloading JSON(P) data to use:
http://homel.vsb.cz/~mor03/TAMZ/cnb_json.php
 - Optional arguments (+1pt when using date, lang and list with buttons to select currency):
 - `date=YYYY-MM-DD` – which date to use
 - `lang={en|cs}` – language, def.: browser
 - `sse={y|n}` – generates SSE event stream instead of JSON (+0.5b if used as well)
 - `callback=function` – JSONP callback
- The application reacts each time input changes
 - The currency label CUR: changes to current currency **code**, the selection should list **country** and **currency** names (label)
 - Exchange rate (CZK ↔ CUR) = **rate/unit**



Example of received JSON data

```
date: "2014-03-18",  
order: "54",  
data: [  
  {  
    country_label: "Australia",  
    curr_label: "dollar",  
    unit: "1",  
    code: "AUD",  
    rate: "17.967"  
  }, ... ],  
labels: ["Country", "Currency", "Amount", "Code", "Rate"],  
lang: "en", cached: false }
```

