HTML5 & CSS3

A chance to Do things Differently

Eng. Niveen Nasr El-Den
iTi

Day 2

GeoLocation

Geolocation

- The Geolocation API is one of the most exciting features of the new web standard.
- Geolocation is the art of figuring out where you are in the world and (optionally) sharing that information with people you trust.
- The ability to get device’s geographic location.
- It is set to request location once or continually.

Geolocation Facts

- HTML5 uses this API for working with maps.
- It is a new property that is added to the existing DOM browser object **navigator**
- The user must agree to share their location, and can tell the browser to remember his choice.

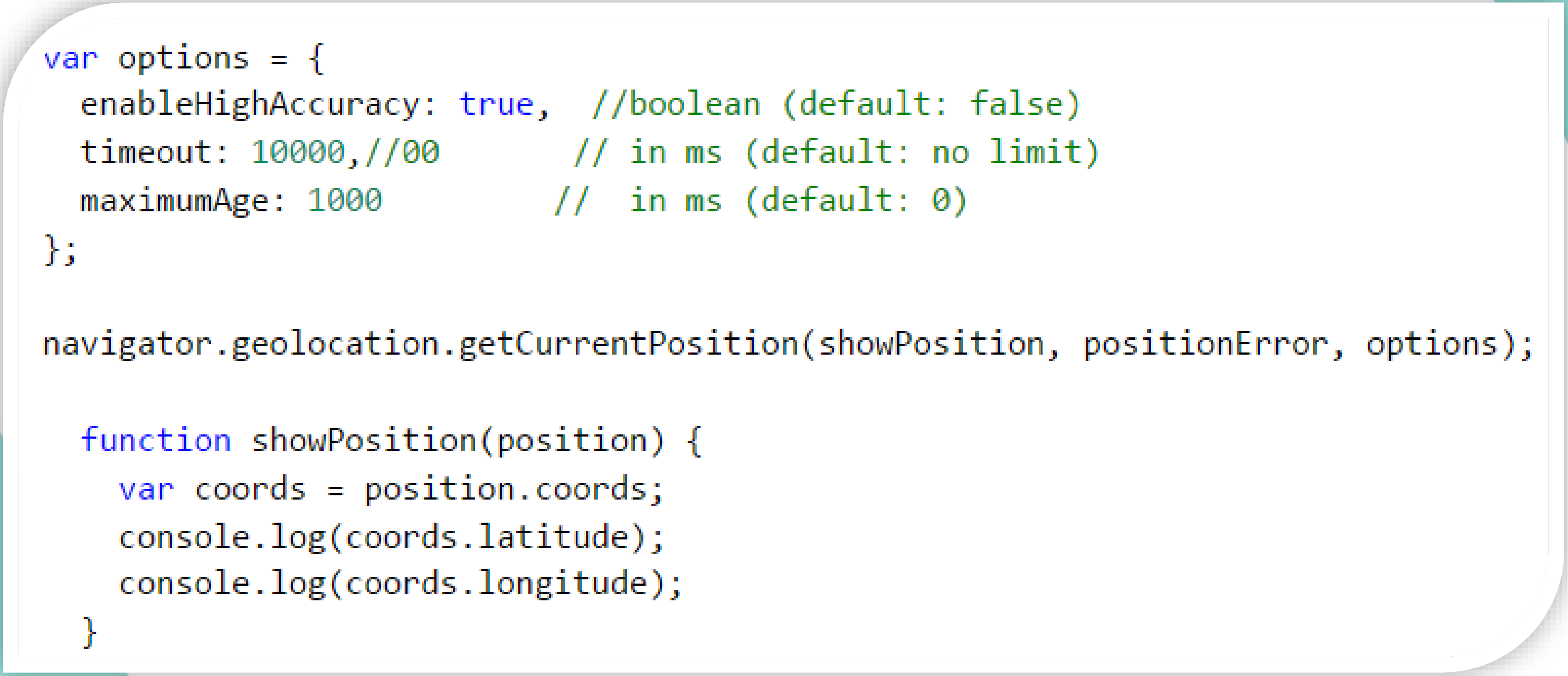
Geolocation Requesting Pattern

- To get user's current location (once)
  - navigator.geolocation.getCurrentPosition(x[,y,z])
    - x: is the onSuccess callback function where a Position object is passed in as the only invocation argument. This Position object contains a coords object which, in turn, contains our latitude and longitude, etc.. values.
    - y: is the errorHandler callback function where the object passed to this handler has code and message properties as follows:
      - 0: UNKNOWN_ERROR
      - 1: PERMISSION_DENIED
      - 2: POSITION_UNAVAILABLE
      - 3: TIMEOUT
    - z: is the options object

Location Option

- enableHighAccuracy (Boolean)
  - ➻ Attempt to gather more accurate location coordinates
  - ➻ May not do anything and cause request to take longer
  - ➻ Default **false**
- timeout (msec)
  - ➻ Determines max time allowed to calculate location
  - ➻ Default is **no limit**
- maximumAge (msec)
  - ➻ Determines how old location value may be before an attempt to refresh coordinates
  - ➻ Default is **0** (immediate recalc.)

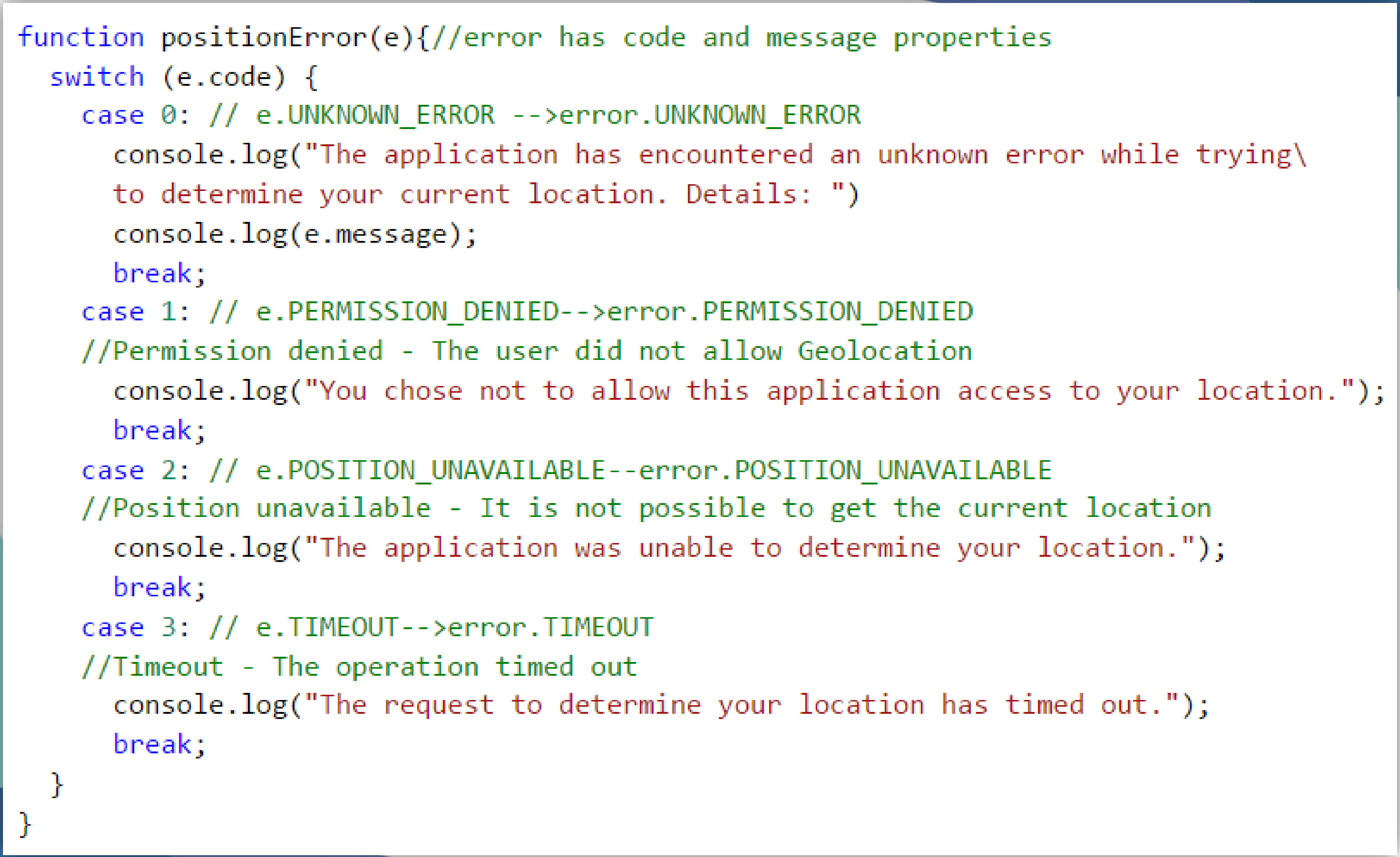
Example



```
var options = {
  enableHighAccuracy: true,  //boolean (default: false)
  timeout: 10000,//00       // in ms (default: no limit)
  maximumAge: 1000         //  in ms (default: 0)
};

navigator.geolocation.getCurrentPosition(showPosition, positionError, options);

  function showPosition(position) {
    var coords = position.coords;
    console.log(coords.latitude);
    console.log(coords.longitude);
  }
```



```
function positionError(e){//error has code and message properties
  switch (e.code) {
    case 0: // e.UNKNOWN_ERROR -->error.UNKNOWN_ERROR
      console.log("The application has encountered an unknown error while trying\
      to determine your current location. Details: ")
      console.log(e.message);
      break;
    case 1: // e.PERMISSION_DENIED-->error.PERMISSION_DENIED
    //Permission denied - The user did not allow Geolocation
      console.log("You chose not to allow this application access to your location.");
      break;
    case 2: // e.POSITION_UNAVAILABLE--error.POSITION_UNAVAILABLE
    //Position unavailable - It is not possible to get the current location
      console.log("The application was unable to determine your location.");
      break;
    case 3: // e.TIMEOUT-->error.TIMEOUT
    //Timeout - The operation timed out
      console.log("The request to determine your location has timed out.");
      break;
  }
}
```

Geolocation Requesting Pattern

- To watch location change (continual)
  - navigator.geolocation.watchPosition(x[,y,z])
    - gets the user’s current position and continually returns updated position.
  - navigator.geolocation.clearWatch()
    - used to stop “watchPosition()” running & execution.

https://www.sitepoint.com/html5-geolocation/

Web Storage APIs

Web Storage APIs

- Sometimes called DOM Storage
- Similar to http-cookies, for storing name-value pairs on the client side; but can store much larger amount of data.
- Two kinds for storing data on the client
  - localStorage
    - stores data with no expiration date
  - sessionStorage
    - stores data for one session

Web Storage APIs

- Web Storage APIs are instance of storage object, and can only store strings.
- It provide up to 5Mbytes per origin
- Same Origin Restrictions
- Stored as key/value pairs, and can only store strings
- We need to check browser support before using Web Storage APIs & add its polyfill if needed

Storage Object Methods & Properties

- Methods
  - ▻ clear()
  - ▻ getItem(‘key’)
  - ▻ setItem(‘key’,’value’)
  - ▻ removeItem(‘key’)
  - ▻ key(idx)

- Properties
  - ▻ length

localStorage

window.localStorage

- Persistent on page reloads
- Data stored locally with no expiration date.
- Avoids HTTP overhead of cookies
- Great for storing user preferences

sessionStorage

window.sessionStorage

- Data stored for only one session
- Lasts as long as browser is open
- Opening page in new window or tab starts new session
- Good for sensitive data

https://html.spec.whatwg.org/multipage/webstorage.html

Cookies
Vs.
Web Storage
?

https://gist.github.com/remy/350433

New Element Enable & Feature Detection

New Element Enable

- Earlier IE doesn’t know how to render CSS on elements that it doesn’t recognize

- HTML5 Shiv or Shim by John Resig document.createElement(“….”) for all of the used tag

https://github.com/aFarkas/html5shiv/blob/master/src/html5shiv.js

API Feature Detection

- Modernizr.js
  - Implement HTML5 Shim
  - Apply classes to <html> based on what the browser support
  - Better place its script within <head> and after<style>
  - if(!Modernizr.localstorage){
        //provide polyfill
    }

http://html5please.com/#polyfill

https://github.com/Modernizr/Modernizr/wiki/HTML5-Cross-browser-Polyfills

API Feature Detection

- Modernizr.js
  - Runs automatically, creating a ***global*** object called ***Modernizr*** that contains a set of Boolean properties for each feature it can detect.
    - Example:
      if your browser supports the video API , the Modernizr.video property will be true.
      else, the Modernizr.video property will be false
  - By default, ***Modernizr*** sets classes for all of tests on the root element.
    - i.e. adding the class for each feature when it is supported, and adding it with a no- prefix when it is not.
  - It is recommended to add no-js class to root element

API Feature Detection

http://caniuse.com/

- Conditionally loading .js file
  - Conditionizr library
    - https://conditionizr.github.io/
    - https://github.com/conditionizr/conditionizr
  - Conditionize jQuery Plugin
    - https://github.com/renvrant/conditionize.js/tree/master
    - https://www.jqueryscript.net/form/jQuery-Plugin-For-Conditional-Form-Fields-conditionize-js.html

Loading Polyfills and/or shim|shiv files is no longer a common practice to provide compatibility

MathML

MathML

- MathML is an XML vocabulary for representing mathematical expressions

- The HTML5 specification provides native support for MathML in HTML documents

- MathML provides both Presentation and Content Markup models.
  - ▻ Presentation markup tags math expressions based on how they should be displayed
    - e.g., “superscripted 2”
  - ▻ Content markup tags expressions based on the mathematical operations performed
    - e.g., “taken to the 2nd power”

MathML Presentation Markup Glossary

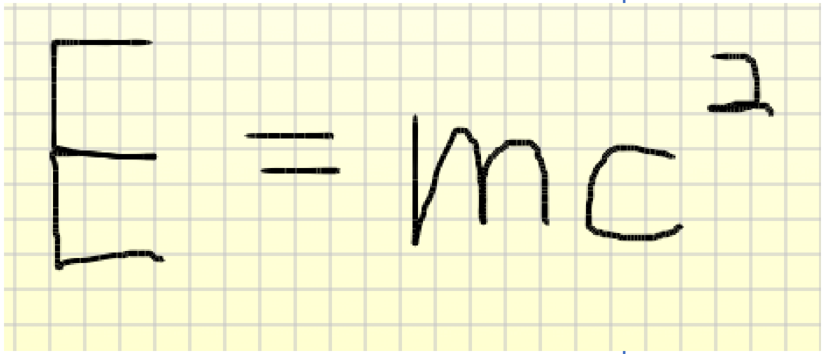
- <math> -- Root element for a mathematical expression
- <mrow> -- Element for grouping subexpressions
- <mo> -- Math operator (e.g., +, -)
- <mi> -- Math identifier (e.g., variable or constant)
- <mn> -- Number
- <mfrac> -- Fraction
- <msqrt> -- Square root
- <msup> -- Superscript
- <msub> -- Subscript
- etc..

https://developer.mozilla.org/en-US/docs/Web/MathML/Element

Converting Famous Eqn. to MathML

https://github.com/fred-wang/mathml.css

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
   <mi> E </mi>
   <mo> = </mo>
   <mi> m </mi>
   <msup>
     <mrow>
       <mi> c </mi>
     </mrow>
     <mrow>
       <mn> 2 </mn>
     </mrow>
   </msup></math>
```



SVG

SVG

- SVG stands for **S**calable **V**ector **G**raphics and it is a language for describing 2D-graphics and graphical applications in XML
- SVG is W3C standard
- HTML5 allows embedding SVG directly using **<svg>...</svg>**

SVG

https://developer.mozilla.org/en-US/docs/Web/SVG/Tutorial

- SVG would draw
  - ▻ rectangle using
    - <rect x="" y="" width=“" height=“" style=“”>
  - ▻ line using
    - <line x1=“" y1=“” x2=“" y2=““ style=“”>
  - ▻ circle using
    - <circle cx="" cy=“" r=“" stroke=““ stroke-width=“" fill=““>
  - ▻ ellipse using
    - <ellipse cx=“" cy=“" rx=“" ry=""“ style=“”>

SVG

- SVG would draw
  - path
    - <path d=“”>
  - polygon using
    - <polygon points=“”> tag
  - polyline using
    - <polyline points=“”> tag

http://tutorials.jenkov.com/svg/index.html

Assignment