HTML5 APIs

Web Tech History

- 1991 HTML
- 1994 HTML2
- 1996 CSS1 + JavaScript
- 1997 HTML4
- 1998 CSS2
- 2000 XHTML1
- 2002 Tableless Web Design
- 2005 Ajax
- 2009 HTML5

HTML5 What

- Syntactical Features (audio, video)
- Semantical Features (section, article)
- New APIs

HTML5 How

- Progressive Enhancements
- Existing web sites can move to HTML5
- Old browsers will still be able to use the page
- Modernizr: http://www.modernizr.com/

HTML5 New APIs

- Web Storage
- Web SQL
- Application Cache
- Web Workers
- Web Socket
- Desktop Notifications
- Drag & Drop
- File System API
- Geolocation
- Device Orientation
- Form Enhancements
- Audio, Video
- Canvas
- Web GL
- History API
- And More...

Modernizr: Before We Begin

- Script Loader
- Feature Detector

Hello Modernizr

```
Modernizr.load({
  test: 3 > 5,

  yep : 'http://code.jquery.com/
jquery-1.9.1.min.js',

  nope: 'http://cdnjs.cloudflare.com/
ajax/libs/zepto/1.0rc1/zepto.min.js'
});
```

More Loader Features

- Specify only “yep” or “nope”
- Provide array
- Complete callback
- Demo: http://jsbin.com/abowap/2/edit

polyfill (n): **a JavaScript shim that replicates the standard API for older browsers**

Feature Detection

- Modernizr.applicationcache
- Modernizr.canvas
- Modernizr.audio
- Modernizr.inputtypes
- Demo: http://jsbin.com/abowap/3/edit

Feature Detection + Polyfill

```
Modernizr.load({
    test: Modernizr.input.placeholder,
    nope: [
            'placeholder_polyfill.min.css',
            'placeholder_polyfill.jquery.min.combo.js'
            ]
});
```

Q & A

Offline Storage

Work Offline

Store Persistent State

Local Data Storage

- How do you store data on a client's machine ?

Cookies ?

Local Storage

- Before HTML5: cookies were used to store data on the client side
- Cookies Disadvantages:
  - Limited size (4k)
  - Sent with every request
  - Complex to use

Local Storage

- A JavaScript API for storing client side data
- Can store up to 5MB (per site)
- Simple API
- Never run out of space

Local Storage

- Store Data
  `window.localStorage.setItem("key", "value");`
- Read Data
  `window.localStorage.getItem("key");`
- Remove Data
  `window.localStorage.removeItem("key");`

Local Storage

- Can also use direct access on the storage object, so this code also works:

```
window.localStorage.username = “Jimmy”;
var username = window.localStorage.username;

console.log(‘Hello ‘ + username);
```

Local Storage

- Methods:
  - `clear()`
  - `key(idx)`
  - `length()`

Session Storage

- Interface is the same as localStorage
- Lifetime is only for the current browser window or tab (or browser session)
- Best used for temporary preferences that should not be shared between tabs

Storage Lab

- We'll implement an address book app
- Keeps a list of contacts info: name, phone number, email
- Provide add/view/delete contacts

Storage Lab

- Use the starter: https://github.com/ynonp/advanced-fed-examples/tree/master/contacts

Offline Web App

- Online-Offline Apps
- Sync with the Cloud, but can suffer a downtime (Think Google Gears)
- Can run completely offline app as a standalone
- Mobile - Save bandwidth

Offline Web App

GET MANIFEST

CACHE MANIFEST
index.html
style/main.css
script/main.js

Cache Manifest File

- Lists all the files that should be stored for offline access
- Enable with:

```
<html manifest="example.appcache">
```

example.appcache

- Required header CACHE MANIFEST
- A list of cached files

```
CACHE MANIFEST

index.html

stylesheet.css

images/logo.png

scripts/main.js
```

Demo

Caching A Website

Offline Web App

- The cache manifest is divided to sections, the default is called CACHE
- Every file listed in the default CACHE section will be cached by the browser forever (or until explicitly deleted by the user)

Offline Web App

- The NETWORK section lists files that should never be cached
- Best used for dynamic content (think gmail)

Offline Web App

- The FALLBACK section lists fallback content that should be used if network is not available
- So, if you’re trying to read a message while offline, you can get a nice formatted error page

Manifest Cache

CACHE	Cache the file locally, never refresh
NETWORK	Files should not be cached
FALLBACK	Cache a fallback content

Offline App Exercise

- A Dynamic reader with two pages
- First lists available articles, the second displays a given article
- Use your favorite server side technology and jQM
- Which files do you put in each manifest section ?

Manifest - The Good

- Can store any file locally
- Provides offline/online app functionality
- Transparent to the user

Manifest - The Bad

- Not suitable for data storage
- Complex update logic

Web SQL

- The final storage option for offline apps is the Web SQL
- Allows using a local database to better store relational data
- Best suited for hierarchical lists apps

Q & A

- Storage
- Geo Location
- Video
- Canvas

Geo Location

- Detect where your user is now
- Show nearby places
- Display location aware data

Technologies

- GPS
- A-GPS
- Cell Information
- WiFi Positioning

GPS

- Global Positioning System
- Accuracy error: 2-10m
- Requires clear sky view
- Time to position:
  5 seconds - 5 minutes

A-GPS

- Uses both GPS chip and network information to get location
- Much faster to get location

Cell Information

- Uses cell towers to calculate a device's location
- Accuracy: A block or up to some km
- Time to location: immediate

WiFi Positioning

- Detect location using a list of wireless routers in your area
- Relies on existing router databases

Location API

- Supported Platforms:
  - iPhone 3.0+
  - Android 2.0+

Location API

```
function get_location() {
  navigator.geolocation.getCurrentPosition(show_map);
}

function show_map(position) {
  var lat = position.coords.latitude;
  var long = position.coords.longitude;
  var when = position.timestamp;
  // show me the map
}
```

Location API

- `navigator.geolocation` is the entry point for all location related calls
- Location querying is async, so a callback is supplied
- User will be asked permission to share location

Location API

- iPhone browser asks user permissions before sharing location

Location API

- Location API uses callbacks to perform error handling
- When working with location, always consider errors

Location API

```
getCurrentPosition(
  successCallback,
  optional errorCallback,
  optional config
);
```

The Coord object

- `coords.latitude`
- `coords.longitude`
- `coords.altitude`
- `coords.accuracy`
- `coords.altitudeAccuracy`
- `coords.heading`
- `coords.speed`
- `timestamp`

The Coord Object

- Not all fields will be available at all times
- Sizes are specified in meters and degrees
- timestamp is a DOMTimestamp (acts like a Date object)

Location Config

- last parameter to getCurrentPosition is a config object
- there are 3 configuration options available:
  - timeout
  - maximumAge
  - enableHighAccuracy

Using Config

```
window.navigator.geolocation.getCurrentPosition(
  successCallback,
  failureCallback,

  { timeout              : 0,

    maximumAge           : 60000,

    enableHighAccuracy : false });
```

How long to wait before giving up (in ms)

Try to use a cached value aged (in ms)

Use the most accurate positioning method

Handle Errors

- The errorCallback takes a single argument called error object
- The object has two fields: error.code & error.message
- Use error.message for debug only

Next: The Map

Treasure Map
Smuggler's Cove
Parrot Haven
Glass Beach
Blue Lake
Hangman Forest
Treasure
Dead man Creek
Shipwreck Island
N
W
E
S
NW
NE
SW
SE

Show The Map

- On the iPhone, a redirect to a google maps url starts the map application
- The Good: easy to use
- The Bad:
  - user leaves the app

Google Maps API

- A JS API to display embedded maps in web sites
- Works on both desktop and mobile devices
- (Almost) Free to use

Google Maps API

- Assign a special empty div that will contain the map. recommended size of the div is entire page.
- Display the map by creating a google.maps.Map object
- Place markers on the map using google.maps.Marker
- Full documentation: http://developer.google.com/apis/maps/documentation/javascript/

Google Maps API

- Maps Workflow:
  - Choose a div
  - Create a new google.maps.Map(...)
- Remember to set minimum height on the div

Objects To Note

- google.maps.LatLng - represents a latitude/longitude pair
- Methods:
  - lat() returns the lat value
  - lng() returns the long value

Objects To Note

- google.maps.Map(element, options)
- Interesting Methods:
  - panBy(x, y) - move the map by x,y
  - panTo(latlng) - move the map to latlng
  - setZoom(zoom)

Objects To Note

- google.maps.MapOptions: configuration object
- Interesting options
  - center: latlng
  - disableDefaultUI: boolean
  - draggable: boolean
  - mapTypeId: HYBRID, ROADMAP, SATELLITE

Demo

Show Location On Map

Location Tracking

- Receive notifications about location changes
- Can use for navigation apps, sport apps, and more
- Browser must remain open

Location Tracking

- use `watchPosition` to start watching a user's position
- The method returns a watch id. Keep it. When tracking is no longer needed, use `clearWatch` with the watch id to stop.
- The callback of watchPosition will get called every time location has changed

Exercise: Where's My Car

- Write a parking reminder app
- One button titled "parked" which marks current location and keeps it in local storage
- Another button titled "find" which shows a map leading to your car
- Keep data in local storage

Q & A

- Storage
- Geo Location (iPhone, Android)
- Video & Audio (iPhone, Android)
- Canvas (iPhone, Android)

VIDEO

THE EASY WAY

Video Tag

- HTML5 introduces a <video> tag to embed videos in a web page.
- Different browsers support different video formats. The <video> tag can specify multiple formats.

Video Formats

- Video formats are like languages
- The same video can be encoded in different formats
- A browser must “speak the language” to be able to play the video

Video Converter

- Miro is a free open source video player and converter
- http://www.getmiro.com

Browser Support

- HTML5 spec does not define a video codec to use
- h.264 is the most widely supported. Plays on iPhone and Android

The Markup

```
<video poster="star.png">

  <source src="zebra.mp4" />

</video>
```

Poster image file name

Video file name. Usually mp4 for mobile devices

Limitations

- Video will start playing in a dedicated “player” window, when the user taps it
- It is not possible to auto play the video on mobile
- It is not possible to embed other content on the video on mobile

Audio Tag

- Structured like a Video tag
- iPhone and Android support MP3
- No autoplay

Lab

- Audio SoundBoard
- Use Animal Sounds: http://apps.ynonperek.com/audio.zip
- Images: http://apps.ynonperek.com/images.zip
- Use starter: https://github.com/ynonp/mobileweb-examples/blob/master/html5/SoundBoard/audioboard_starter.html

Q & A

- Storage
- Geo Location
- Video & Audio
- Canvas (iPhone, Android)

CANVAS

HTML5 Canvas

- A 2d graphics canvas for HTML apps
- Games
- Dynamic images
- Client-side painting (saves bandwidth)

Hello Canvas

- An HTML element `<canvas>` on the markup
- JS code to do actual painting

Hello Canvas

- The canvas element has no content and no border of its own
- A canvas keeps a painting context on which we perform the drawing
- First example uses `context.fillText()`

examples/canvas/intro.html

Basic Drawing

- Context object provides the drawing functions. Each takes coordinates and data.
- Drawing style is determined by setting attributes on the context object.
- Most style properties are plain text strings

Coordinate System

Drawing Rectangles

- Let's start with drawing rectangles
- note fillRect draws a filled rectangle, while strokeRect draws just the outline
- fillStyle and strokeStyle determine the colors

```
fillRect(x, y, w, h)

strokeRect(x, y, w, h)

clearRect(x, y, w, h)
```

Demo: Rects

Drawing Paths

- A composite shape on the canvas is called “path”
- Paths are made of lines, arcs, curves and rectangles
- A path can be filled or stoked. Nothing is painted until either fill() or stroke() is called

Drawing Paths

- Lines are painted with lineTo and moveTo
- stroke() performs the actual drawing

```
moveTo(x, y)

lineTo(x, y)
```

Lines Example

```
ctx.beginPath();
ctx.strokeStyle = "hsl(249, 41%, 50%)";

ctx.lineWidth = 6;
ctx.moveTo(0, 0);
ctx.lineTo(20, 10);
ctx.lineTo(0, 20);
ctx.stroke();
```

Cool Color Selector:
http://mothereffinghsl.com/

Using The Image

- It's possible to use an image drawn in a canvas for other elements
- get the image url using: `canvas.toDataURL()`
- use image url as a source attribute of an img tag
- Extra: draw on a hidden canvas

Drawing Circles

- use arc() and arcTo() to draw circles or parts of them
- Degrees are specified in radians.
  Math.PI = 180 deg
- remember to fill() or stroke()

```
arc(x, y, radius,
    startAngle,
    endAngle,
    antiClockWise)

arcTo(x1, y1,
      x2, y2,
      radius)
```

Exercise 1

- Draw the image on the right
- Use moveTo and arc
- Bonus: add colors

Exercise 2

- Draw the image on the right
- Place pins randomly across the screen

Advanced Canvas

- Transformations
- Image Filters

Transformations

- supported transformations: translation, scaling, rotation
- Transformations affect all drawing code performed after them

```
translate(x, y)

rotate(angle)

scale(x, y)
```

Translation

- Move the origin of the canvas to another point on the canvas
- The new point is now (0, 0)
- This can make drawing code simpler

Translation Example

- Use translation to paint the four painted rectangles on the right

examples/html5/canvas/translate.html

Scaling

- scales the rendering context size
- x and y are scale factors
- translate before scale to maintain position

Scale Example

examples/html5/canvas/scale.html

Rotation

- Rotates a shape around its (0, 0) point of origin
- Angle is specified in radians.
  Math.PI = 180 deg (half a circle)
  Math.PI / 4 = 45 deg
- Translate before rotate to maintain position

Example: Rotation

```
ctx.translate(75,75);

for (var i=1;i<6;i++){
  ctx.save();
  ctx.fillStyle = 'rgb('+(51*i)+','+(255-51*i)+',255)';

  for (var j=0;j<i*6;j++){ // draw individual dots
    ctx.rotate(Math.PI*2/(i*6));
    ctx.beginPath();
    ctx.arc(0,i*12.5,5,0,Math.PI*2,true);
    ctx.fill();
  }

  ctx.restore();
}
```

Example: Rotation

- The background was painted with a gradient fill
- Notice the save() and restore() at the beginning and end. Now we can put the code inside a function, and it won't affect outside code

```
ctx.save();
var grad = ctx.createLinearGradient(0, 0,
150, 0);
grad.addColorStop(0, "#888");
grad.addColorStop(1, "#eee");

ctx.fillStyle = grad;
ctx.fillRect(0, 0, 150, 150);
ctx.fill();
ctx.restore();
```

Exercise

- Draw the clock on the right
- Use rotate and translate to simplify calculations
- Bonus: Show the time

Image Filters

Drawing Images on Canvas

- It's possible to paint a png or jpeg image on a canvas using drawImage
- Canvas also lets JS code take the pixels from the image into an array. Using this array, we can transform the pixels and write the result to a new canvas
- The operation is called filtering

Image Drawing API

```
drawImage(image, x, y)
paint an entire image object on the canvas at position (x,y)



drawImage(image, x, y, w, h)
paint an entire image object on the canvas, scaling it to size (w,h)



drawImage(image, sx, sy, sw, sh, dx, dy, dw, dh)
paint a slice of the image on a canvas, can use dw,dh to scale
```

Drawing Images and File API

- Use *input type="file"* to upload a photo from the phone's camera or gallery
- Access the image and paint it on a canvas using HTML5 File API

Reading The Photo

- FileReader is used to read a photo from the input field
- *reader.readAsDataURL*(input_field) - reads the data
- reader emits *load* event when done
- Example: http://jsfiddle.net/influenztial/qy7h5/

Image Filters

- For the next few slides, we'll create a Filters object. Each filter is a function of that Filters object
- The Filters object is responsible for reading pixel data and interacting with the canvas
- A filter function takes pixel data and manipulates it

Filters Code

- When testing on desktop, some browsers will throw security exceptions if the page is read locally.

```
canvas.toDataURL()

ctx.getImageData(sx, sy, sw, sh)

ctx.putImageData(data, dx, dy)
```

- Code:

- https://github.com/ynonp/mobileweb-examples/blob/master/html5/canvas/ImageFilters/app/cutsom_filters.js

Canvas Exercise

- Implement a mobile drawing app using the canvas and touch events
- App should have a footer with a selection of colors. Tapping a color sets current color
- Tapping anywhere on screen should draw in the selected color
- Bonus: share photo on server

HTML5 Drag & Drop

HTML5 Drag & Drop

- Started back in IE5
- Still wrong

Check Support

```
if (Modernizr.draganddrop) {
  // Browser supports HTML5 DnD.
} else {
  // Fallback to a library solution.
}
```

Draggable Content

```
<div id="columns">
  <div class="column" draggable="true"><header>A</header></div>
  <div class="column" draggable="true"><header>B</header></div>
  <div class="column" draggable="true"><header>C</header></div>
</div>
```

Selections, img and anchors with href are draggable by default

Some CSS Extras

```
[draggable] {
  -moz-user-select: none;
  -khtml-user-select: none;
  -webkit-user-select: none;
  user-select: none;
  cursor: move;
}
```

And The Fun Part

- 7 Dragging Events:
  - dragstart, drag,
  - dragenter, dragleave, dragover
  - drop
  - dragend

For drop events:

cancel defaults for

`dragover` **and** `dragenter`

Demo1: Dropzone

- Prevent default from bad events
- Handle good event
- http://jsbin.com/oberec/1/edit

Alerting Drop Zone

- Use dragenter and dragleave to alert user they're in the drop zone
- Demo: http://jsbin.com/oberec/4/edit

Transferring Data

```
function handleDragStart(e) {
  e.dataTransfer.setData('text/html', this.innerHTML);
}


function handleDrop(e) {
  e.stopPropagation();
  this.innerHTML = e.dataTransfer.getData('text/html');

  return false;
}
```

Other dataTransfer

- Use e.dataTransfer.files to get data dragged from desktop

```
function handleDrop(e) {
  e.stopPropagation();
  e.preventDefault();

  var files = e.dataTransfer.files;
  for (var i = 0, f; f = files[i]; i++) {
    // Read the File objects in this FileList.
  }
}
```

Other dataTransfer

- Use setData to allow dragging “out of” the browser

```
var file = document.getElementById("dragout");

file.addEventListener("dragstart",function(evt){
    evt.dataTransfer.setData("DownloadURL",fileDetails);
},false);
```

Q & A

Thank You

- Slides By:
- Ynon Perek
- ynon@ynonperek.com
- ynonperek.com