

# Exploring HTML 5

Thierry Sans

# Geolocation

# Get GPS coordinates

```
navigator.geolocation.getCurrentPosition(success);  
  
function success(position) {  
    let lat = position.coords.latitude;  
    let long = position.coords.longitude;  
}
```

... and use Google Maps:

<https://developers.google.com/maps/documentation/javascript/examples/map-geolocation>

# Local Storage

# Local Storage (≠ cookies)

- Store key/value pairs in the browser
- Accessible from the same domain only
- Up to 10mb (on Chrome)
- Persistent

# Instructions

Push	<code>localStorage.setItem(<i>key</i>, <i>value</i>)</code>
Pull	<code>localStorage.getItem(<i>key</i>)</code>
Remove	<code>localStorage.removeItem(<i>key</i>)</code>



Drag'n Drop

# Drag & Drop can be use for

Use for

- interacting with the DOM
- uploading a file



# Drag n'Drop events

```
let holder = select_dom_element

holder.ondragstart = function(e){return false;};
holder.ondragend = function(e){return false;};
holder.ondragover = function(e){return false;};
holder.ondragenter = function(e){return false;};
holder.ondragleave = function(e){return false;};
holder.ondrop = function(e){return false;};
```

Canvas

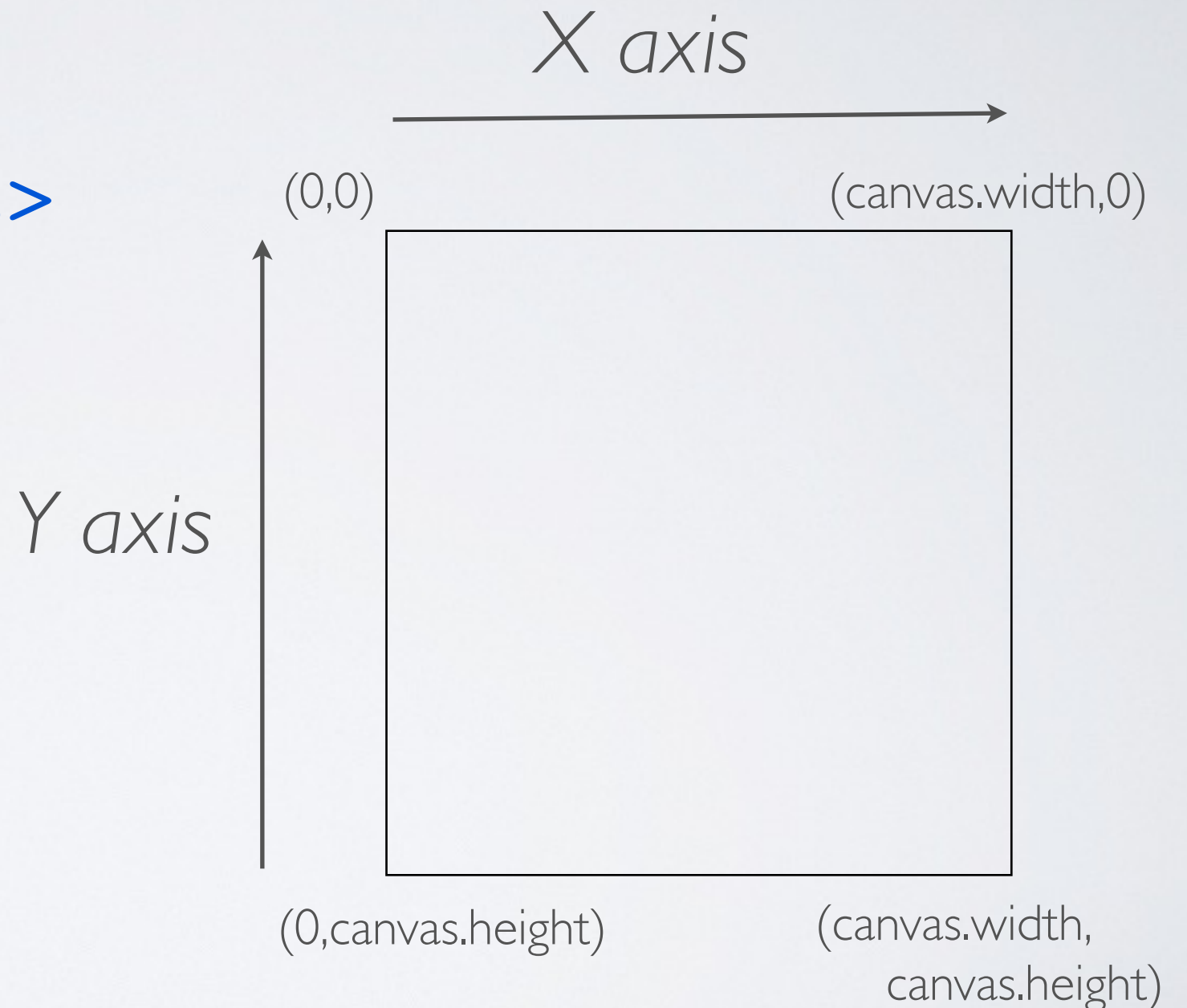
# HTML - the **canvas** tag

`<canvas></canvas>`

Specific attributes:

- **Height**
- **Width**

These are **not** the styling attributes  
`CSS.height` and `CSS.width`



# Javascript - the 2D context

The 2D context object is used for drawing

```
let canvas = $( "#myCanvas" )[0];  
let context =  
canvas.getContext( "2d" );
```

# Drawing lines

```
context.moveTo(10, 10);  
context.lineTo(50, 50);  
context.lineWidth = 2;  
context.strokeStyle = "#00FF00";  
context.stroke();
```

start-point

end-point

line width

line color

line style

Curve lines: see *arcs*, *quadratic curves* and *Beziers curves*



# Drawing shapes using the concept of *path*

```
context.beginPath(); // begin custom shape
context.moveTo(170, 80);
context.bezierCurveTo(130, 100, 130, 150, 230, 150);
context.bezierCurveTo(250, 180, 320, 180, 340, 150);
context.bezierCurveTo(420, 150, 420, 120, 390, 100);
context.bezierCurveTo(430, 40, 370, 30, 340, 50);
context.bezierCurveTo(320, 5, 250, 20, 250, 50);
context.bezierCurveTo(200, 5, 150, 20, 170, 80);
context.closePath(); // complete custom shape
context.lineWidth = 5;
context.fillStyle = "#8ED6FF";
context.fill();
context.strokeStyle = "#0000ff";
context.stroke();
```



example from *HTML5CanvasTutorial*

<http://www.html5canvastutorials.com/tutorials/html5-canvas-shape-fill/>



# Predefined shapes

rectangle

```
context.rect(topLeftCornerX, topLeftCornerY, width, height);
```

semi-circle

```
context.arc(centerX, centerY, radius, 0, Math.PI, false);
```

circle

```
context.arc(centerX, centerY, radius, 0, 2 * Math.PI, false);
```

# Drawing an existing image into a canvas

```
context.drawImage(imageObj, destX, destY, destWidth, destHeight);
```

Video

# The `video` tag

```
<video src="movie.webm" poster="movie.jpg" controls="true">  
    This is fallback content to display if the browser  
    does not support the video element.  
</video>
```

Specific attributes:

- audio
- autoplay
- controls
- height
- width
- loop
- poster
- preload
- src

see [http://www.w3schools.com/html5/tag\\_video.asp](http://www.w3schools.com/html5/tag_video.asp)

# Different video formats (yet)

webm



◎ Several formats = Several videos in your web application

```
<video poster="movie.jpg" controls>
  <source src='movie.webm' type='video/webm; codecs="vp8.0, vorbis"' />
  <source src='movie.ogv' type='video/ogg; codecs="theora, vorbis"' />
  <source src='movie.mp4' type='video/mp4; codecs="avc1.4D401E,
mp4a.40.2"' />
  <p>This is fallback content</p>
</video>
```

see browser support: [https://en.wikipedia.org/wiki/HTML5\\_video#Browser\\_support](https://en.wikipedia.org/wiki/HTML5_video#Browser_support)

# Mixing video and canvas

Exactly the same as drawing an image!

```
context.drawImage(videoObj, destX, destY, destWidth, destHeight);
```



# Getting and setting a video frame using canvas

Get the current image frame

```
let frame = myCanvasCtx.getImageData(0, 0, width, height);
```

- Set the current image frame

```
myCanvasCtx.putImageData(frame, 0, 0);
```

see <http://www.phpied.com/pixel-manipulation-in-canvas/>

# Manipulating the frame object

A frame = a matrix of pixels components

Pixel components = red green blue alpha

```
Red    = frame [(row * 4 * width) + (column * 4)];  
Green  = frame [(row * 4 * width) + (column * 4) + 1];  
Blue   = frame [(row * 4 * width) + (column * 4) + 2];  
Alpha  = frame [(row * 4 * width) + (column * 4) + 3];
```

# Example - The Green Screen Effect

How to you change the background of a video dynamically?  
like in <https://dl.dropboxusercontent.com/u/26942820/CDN/CKVideo/index.html>

see <http://tech.pro/tutorial/1281/chroma-key-video-effects-using-javascript-and-the-html5-canvas-element>

Speech

# Speech2Text - setup

```
let recognition = new webkitSpeechRecognition();

recognition.continuous = true;
recognition.interimResults = true;
recognition.lang = 'en-us';

recognition.onresult = function (e) {
    for (let i = e.resultIndex; i < e.results.length; ++i){
        console.log(e.results[i][0].transcript);
    }
};
```



# Speech2Text - control

Start

```
recognition.start();
```

Stop

```
recognition.stop();
```



# Text2Speech

```
let msg = new SpeechSynthesisUtterance( );  
msg.text = 'This is my text';  
msg.lang = 'en-us';  
speechSynthesis.speak(msg);
```

# and more ...

Camera API

Web workers (multi-threading)

Web sockets (full-duplex communication between browser and server)

Web RTC (P2P communication between browsers)

WebGL (3D)

Phone features

- <https://github.com/AurelioDeRosa/HTML5-API-demos>
- <http://www.tomg.co/gyrojs>

... and cool libraries

## Data visualization

- <http://d3js.org/>
- <http://cartodb.com/>
- <http://snazzymaps.com/>
- <http://selection.datavisualization.ch/>
- <http://www.data-mania.com/blog/19-free-applications-for-data-science/>

## Machine Learning

- <http://www.datumbox.com/machine-learning-api/>

# Natural Language Processing

- <https://www.talater.com/annyang/>
- <https://wit.ai/>
- <http://blog.mashape.com/list-of-25-natural-language-processing-apis/>

# Visualization

- <http://trackingjs.com/>
- <http://facedetection.jaysalvat.com/>
- <http://www.faceplusplus.com/>

# Web RTC

- <https://togetherjs.com/>
- <http://peerjs.com/>

# Graphics

- <http://threejs.org/>