

HTML



Web workers and geolocation

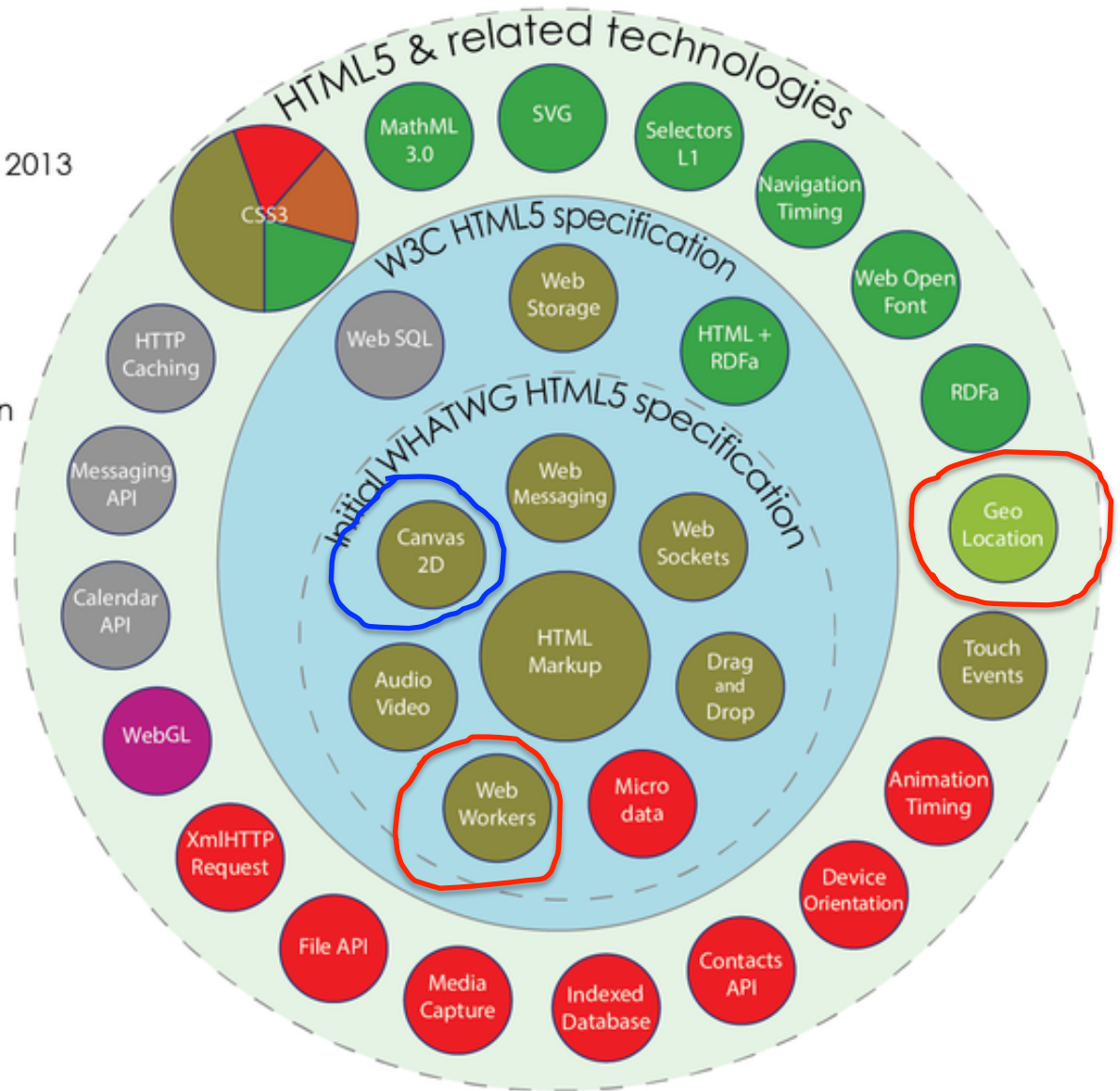
Overview

- **Web Workers**
 - Single-threaded woes
 - Web worker threads
 - Using, limitations
 - Examples
 - Fibonacci calculation
 - Mandelbrot set
- **Geolocation**
 - Privacy issues
 - JavaScript API
 - Example pages

HTML5

Taxonomy & Status on January 20, 2013

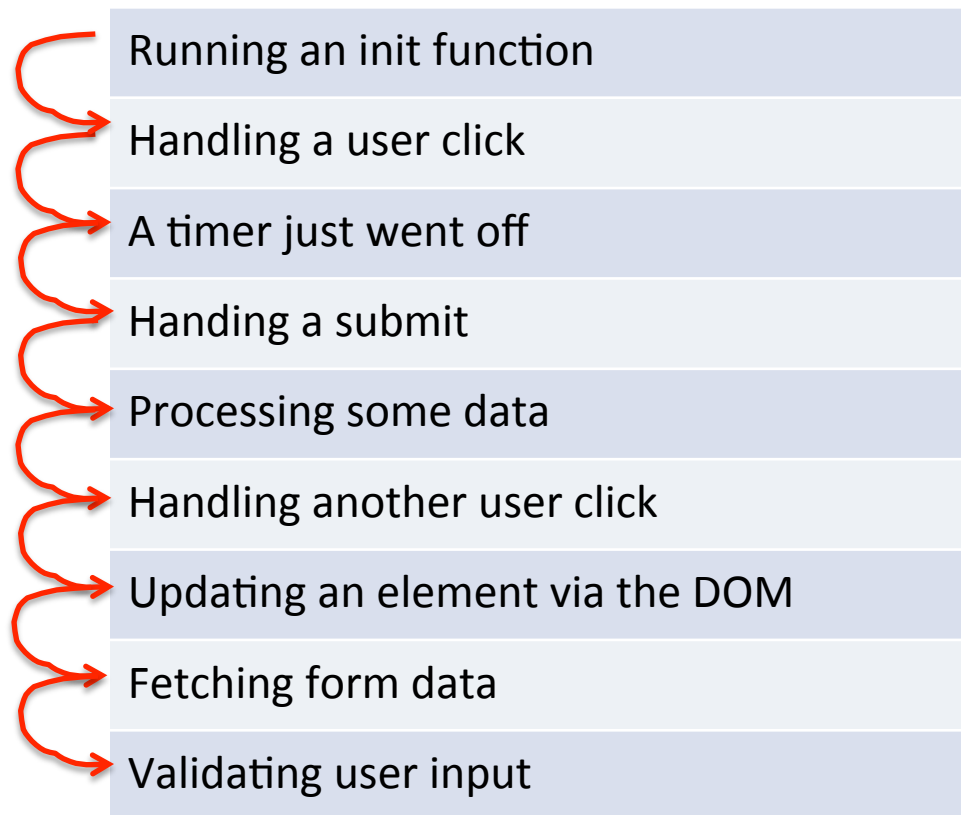
- W3C Recommendation
- Proposed Recommendation
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Deprecated



by Sergey Mavrody (cc) BY · SA

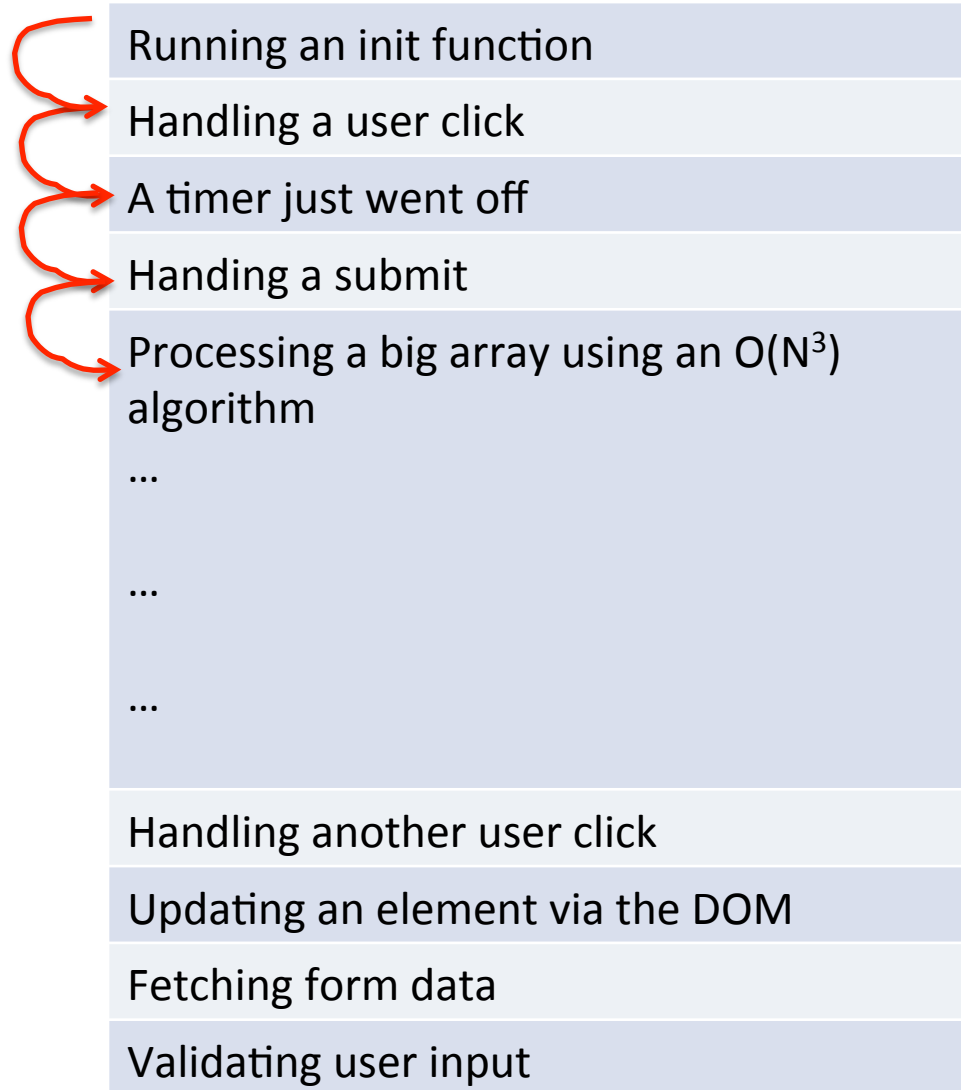
One thread to rule them all

- Prior to HTML5:
 - Single JavaScript thread running your page



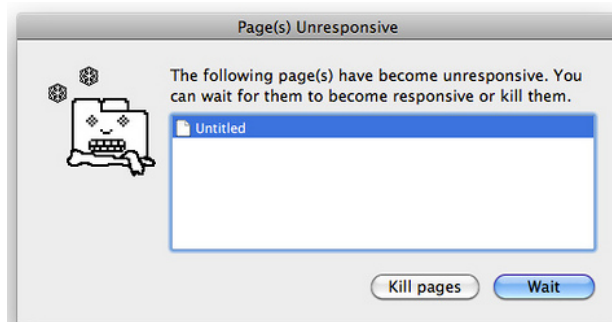
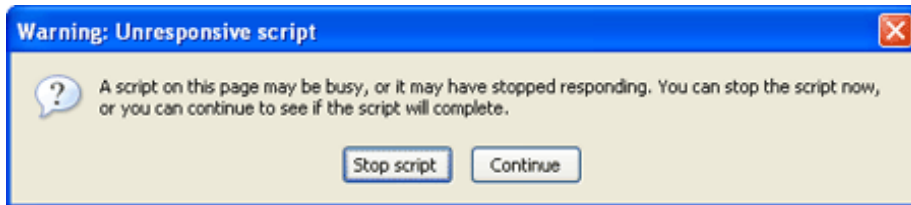
One thread to rule them all

- **Problem:** Time consuming / blocking task



Single-threaded pros/cons

- **Prior to HTML5:**
 - Single-thread running all JavaScript
 - Advantages:
 - **Easy to code** and understand
 - **No concurrency issues**
 - Disadvantages:
 - Page can become **unresponsive** to user

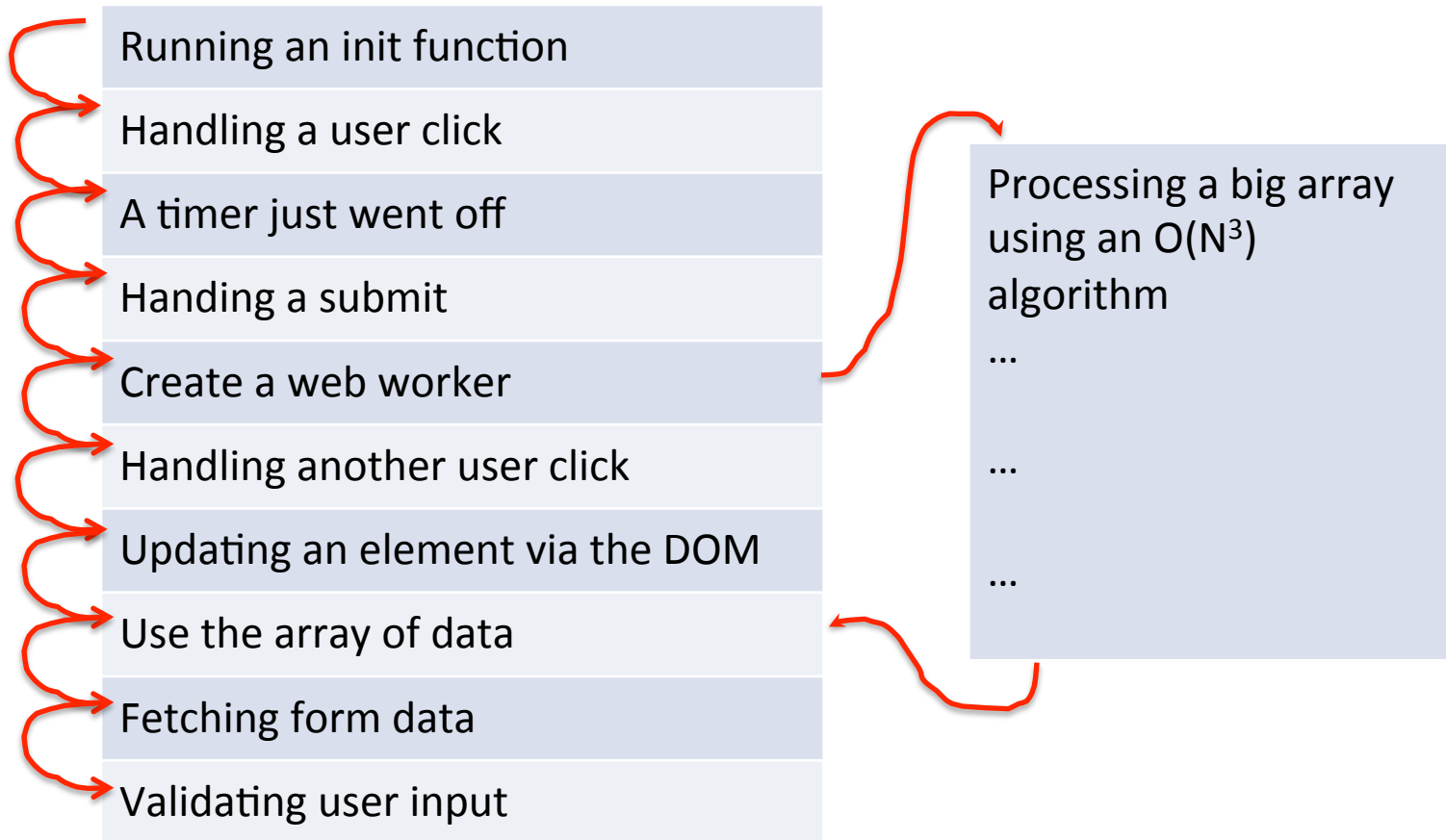


Web workers

- Multiple JavaScript threads running in browser
 - Offload time-consuming computational tasks
 - Better utilization of modern **multi-core** processors
 - Threading *may* model your problem better
 - **Simplifying your code**
- Type types
 - Dedicated workers
 - Linked to the **creating page**
 - Shared workers
 - Shared between **all pages on a domain**

One thread to rule them all

- **Problem:** Time consuming / blocking task



Web workers

- How well are they supported?

# Web Workers - Candidate Recommendation										*Usage stats: Global	
Method of running scripts in the background, isolated from the web page										Support:	74.43%
Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	BlackBerry Browser	IE Mobile	
								2.1			
								2.2			
						3.2		2.3			
						4.0-4.1		3.0			
	8.0					4.2-4.3		4.0			
	9.0		31.0			5.0-5.1		4.1			
	10.0	26.0	32.0			6.0-6.1		4.2-4.3	7.0		
Current	11.0	27.0	33.0	7.0	19.0	7.0	5.0-7.0	4.4	10.0	10.0	
Near future		28.0	34.0		20.0						
Farther future		29.0	35.0		21.0						
3 versions ahead		30.0	36.0								

<http://caniuse.com/#search=worker>

```
if (!window["Worker"])
{
    alert("No support for web workers!");
}
```

JavaScript code to test if browser supports web workers.

Web worker details

- **Creating:**
 - Worker is defined its **own JavaScript file**
- **Limitations:**
 - Workers don't have access to many things:
 - No access to **DOM**
 - No access to **variables or functions in main** program
 - No access to **many runtime objects**
 - e.g. window, document, parent
- **Process:**
 - **Main program sends message** to worker
 - Worker does work
 - **Worker sends message back** with results

Web worker example

- Goal:
 - Multithreaded Fibonacci calculator
 - Using simple (slow, wasteful) recursive algorithm
 - User types number, hits button to start calculation
 - Results appear in `<div>` as they arrive
- Single threaded version
- Multi-threaded version

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Fibonacci calculator</title>
<script>
  function goButton()
  {
    var num = document.getElementById("num").value;
    result = fib(num);
    document.getElementById("results").innerHTML +=
      "fib(" + num + ") = " + result + "<br />";
  }
  function fib(n)
  {
    if (n == 0)
      return 0;
    if (n == 1)
      return 1;
    return fib(n - 1) + fib(n - 2);
  }
</script>
</head>

<body>
<input type="text" size="10" id="num" /><br />
<input type="button" value="Go" onClick="goButton();"/>
<div id="results"></div>
</body>
</html>
```

fib1.html
Single threaded
version

NOTE:
This is a stupendously
inefficient way to
calculate this!

```
<script>
function goButton()
{
  if (!window["Worker"])
  {
    alert("No support for web workers!");
    return;
  }

  var num = document.getElementById("num").value;
  var worker = new Worker("fib.js");
  worker.onmessage = addResult;
  worker.postMessage(num);
}
function addResult(event)
{
  document.getElementById("results").innerHTML +=
    "fib(" + event.data.num + ") = " +
    event.data.result + "<br />";
}
</script>
```

fib2.html
Web Worker
version

fib.js
Worker JavaScript

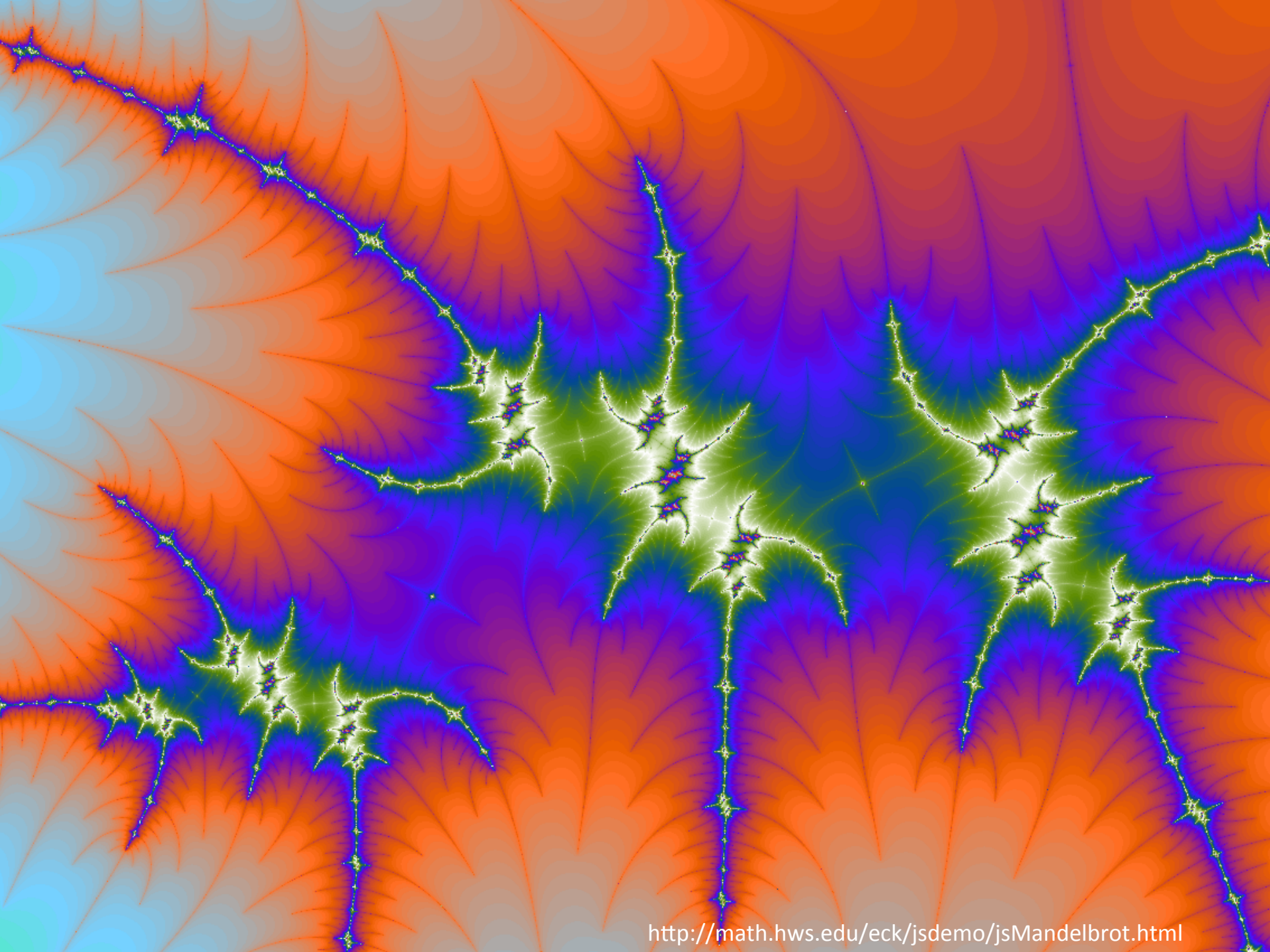
```
onmessage = startWork;

function startWork(event)
{
  var n    = event.data;
  var r    = fib(n);
  var msg = {num: n, result: r};
  postMessage(msg);
}

function fib(n)
{
  if (n == 0)
    return 0;
  if (n == 1)
    return 1;
  return fib(n - 1) +
    fib(n - 2);
}
```

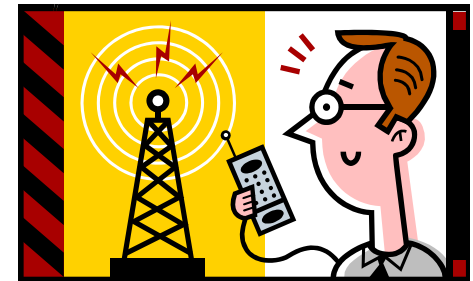
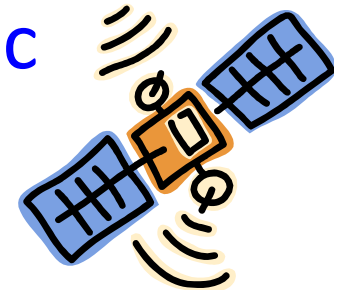
Other web worker details

- **Killing a worker**
 - `worker.terminate()` from main program
 - Worker can stop itself, `close()`
- **Importing other JavaScript files:**
 - Workers must use function: `importScripts()`
 - Also used for JSONP (JSON with padding)
 - For doing cross-domain Ajax
- **Workers can also:**
 - Spawn their own workers
 - Use `setInterval()` to schedule periodic work



Geolocation

- Actually not part of W3C HTML5 spec
 - But W3C standard
 - Widely supported
- High-level interface to device location info
 - Global Positioning System (GPS)
 - Or location inferred from network signal
 - IP address
 - RFID
 - WiFi, Bluetooth
 - GSM/CDMA cell IDs
 - User input



- One shot requests or repeated updates

Geolocation

- How well is it supported?
 - Almost every modern desktop/mobile browser

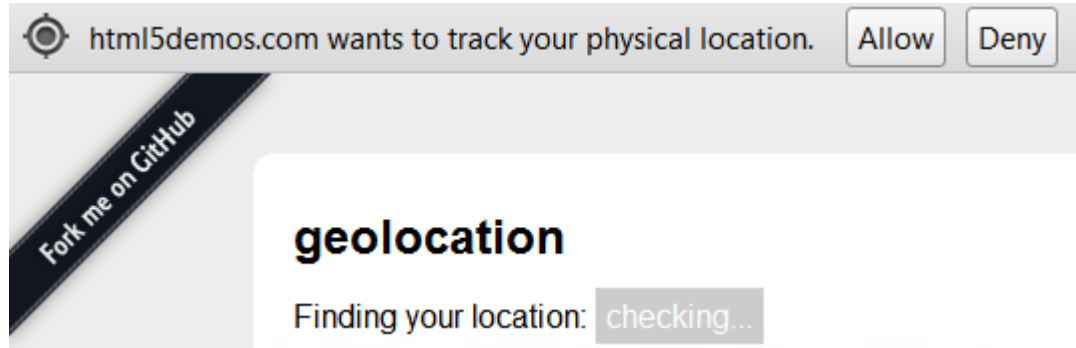
# Geolocation - Candidate Recommendation									
<i>Method of informing a website of the user's geographical location</i>									
*Usage stats: Global									
Support: 82.44%									
Partial support: 0.03%									
Total: 82.47%									
Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Blackberry Browser
								2.1	
								2.2	
						3.2		2.3	
	7.0	16.0				4.0-4.1		3.0	
	8.0	17.0				4.2-4.3		4.0	
	9.0	18.0	23.0	5.1		5.0-5.1		4.1	
Current	10.0	19.0	24.0	6.0	12.1	6.0	5.0-7.0	4.2	7.0
Near future		20.0	25.0		12.5				10.0
Farther future		21.0	26.0						

<http://caniuse.com/#feat=geolocation>

Privacy issues

- User permission required

- Browser must have express permission from user



- User can ask browser to remember permission

- WiFi access point databases

- Send AP MAC / GPS / Cell IDs to central database
- Apple, Google phones stored history on the phone
- Sometimes people's phones are an AP

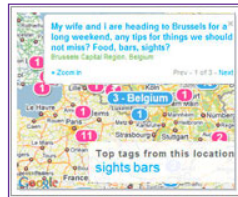
<http://petewarden.github.com/iPhoneTracker/>

Geolocation

- Often combined with the Google Maps API:

★ Google Maps API Family Home

Google Maps has a wide array of APIs that let you embed the robust functionality and everyday usefulness of [Google Maps](#) into your own website and applications, and overlay your own data on top of them:



Maps JavaScript API

Embed a Google Map in your webpage using JavaScript. Manipulate the map and add content through many services.

[Version 3](#) - [Version 2](#)



Maps API for Flash

Use this ActionScript API to embed a Google Map in your Flash-based web page or app. Manipulate the Map in three dimensions and add content through many services.

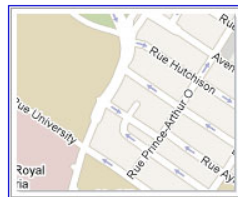
[Learn more](#)



Google Earth API

Embed a true 3D digital globe into your web page. Take your visitors anywhere on the Earth (even below the ocean) without leaving your web page.

[Learn more](#)



Maps Image APIs

Embed a fast and simple Google Maps image or Street View panorama in your web page or mobile site without requiring JavaScript or any dynamic page loading.

[Static Maps](#) - [Street View](#)



Web Services

Use URL requests to access geocoding, directions, elevation, and places information from client applications, and manipulate the results in JSON or XML.

[Learn more](#)

http://gmaps-samples-v3.googlecode.com/svn/trunk/map_events/map_events.html

<http://earth-api-samples.googlecode.com/svn/trunk/demos/drive-simulator/index.html>

<http://maps.googleapis.com/maps/api/streetview?size=600x600&location=40.720032,%20-73.988354&heading=235&sensor=false>

http://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536,-104.9847034&sensor=true_or_false

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Where am I?</title>
<script>

window.onload = getMyLocation;

function getMyLocation()
{
    if (navigator.geolocation)
        navigator.geolocation.getCurrentPosition(displayLocation,
                                                    displayError
                                                    {enableHighAccuracy: false,
                                                    timeout: Infinity,
                                                    maximumAge: 0});

    else
        alert("No geolocation support!");
}

...

</script>
</head>
<body>
<div id="location">
</div>
</body>
```

```

function displayLocation(position)
{
    var lat            = position.coords.latitude;
    var long           = position.coords.longitude;
    var accuracy       = position.coords.accuracy;
    var timestamp      = position.timestamp;

    var altitude       = position.coords.altitude;
    var altitudeAccuracy = position.coords.altitudeAccuracy;
    var heading        = position.coords.heading;
    var speed          = position.coords.speed;

    var div = document.getElementById("location");
    div.innerHTML = "Latitude: "      + lat      + "<br />";
    div.innerHTML += "Longitude: "    + long     + "<br />";
    div.innerHTML += "Accuracy: "     + accuracy + "<br />";
    div.innerHTML += "Timestamp: "    + timestamp + "<br /><br />";

    div.innerHTML += "Altitude: "     + altitude + "<br />";
    div.innerHTML += "Altitude accuracy: " + altitudeAccuracy + "<br />";
    div.innerHTML += "Heading: "      + heading  + "<br />";
    div.innerHTML += "Speed: "        + speed    + "<br />";
}

function displayError(error)
{
    var errorTypes = {0: "Unknown error", 1: "Permission denied by user",
                     2: "Position is not available", 3: "Request timed out"};
    var errorMessage = errorTypes[error.code];
    if (error.code == 0 || error.code == 2)
        errorMessage = errorMessage + " " + error.message;
    var div = document.getElementById("location");
    div.innerHTML = errorMessage;
}

```

Other features

- How fast can we get our fix?
 - geo2.html
 - Asks for high accuracy
 - Will not accept cached result, `maximumAge = 0`
 - `timeout = 10`, increase by 10ms on timeout
- Updates whenever user moves
 - geo3.html

```
function getLocation()
{
    if (navigator.geolocation)
        navigator.geolocation.watchPosition(displayLocation,
                                             displayError,
                                             {enableHighAccuracy: true,
                                              timeout: Infinity,
                                              maximumAge: 0});
    else
        alert("No geolocation support!");
}
```

geo4.html: adding Google maps

```
...
<style>
div#map
{
    margin: 5px;
    width: 400px;
    height: 400px;
    border: 1px solid black;
}
</style>

<script src="https://maps.google.com/maps/api/js?sensor=true"></script>
<script>
...
var map = null;

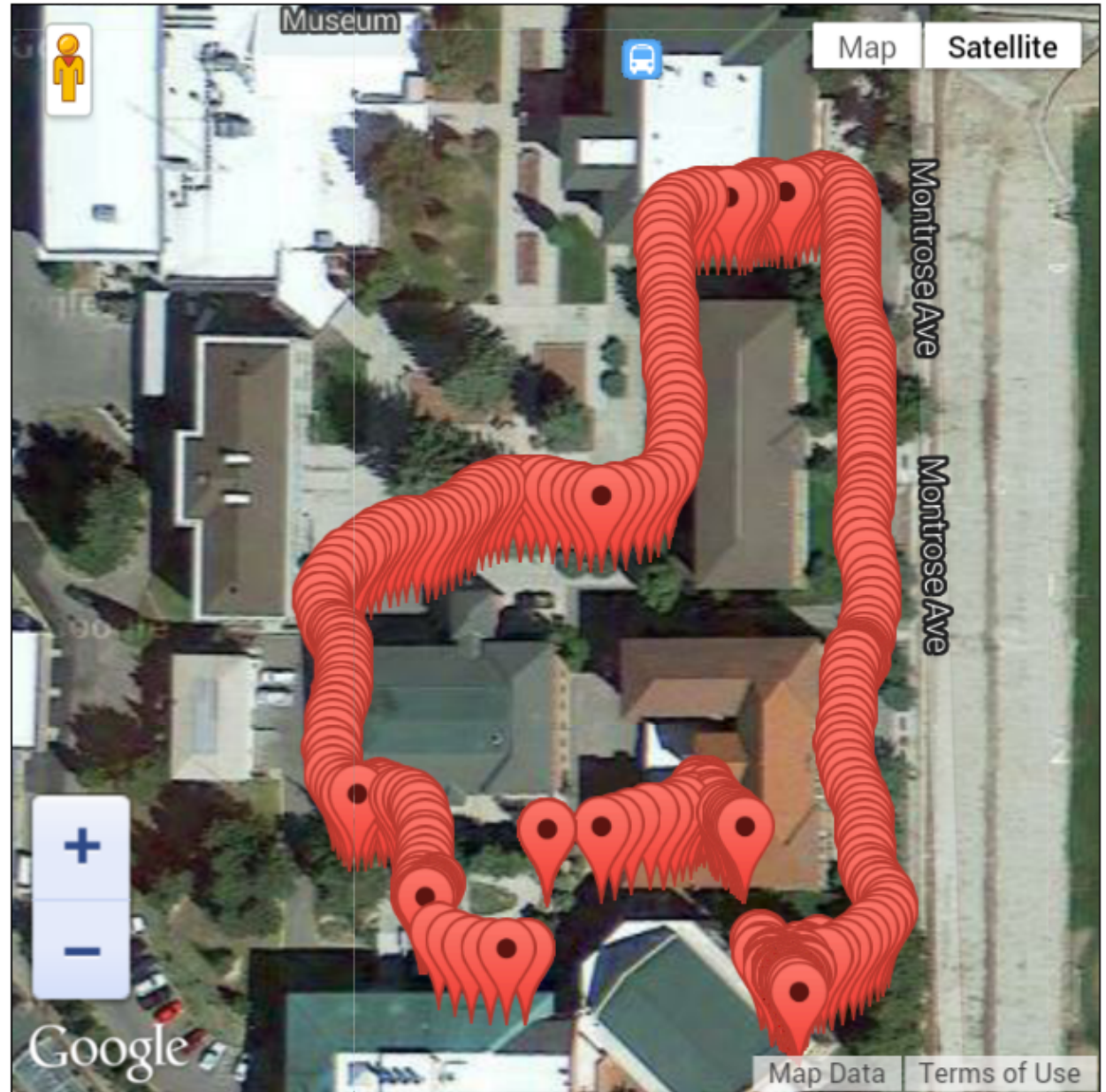
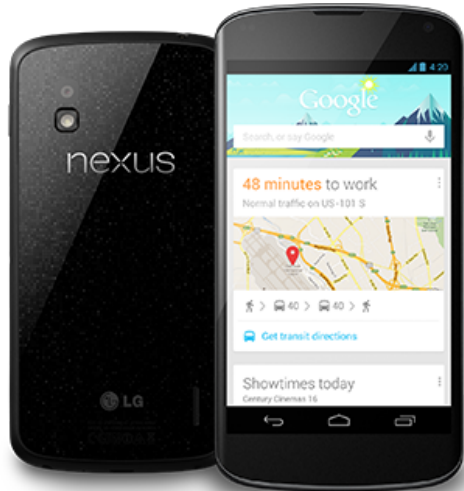
function showMap(coords)
{
    var googleLatAndLong = new google.maps.LatLng(coords.latitude, coords.longitude);
    var mapOptions = {zoom: 10, center: googleLatAndLong, mapTypeId: google.maps.MapTypeId.HYBRID};
    var mapDiv = document.getElementById("map");
    map = new google.maps.Map(mapDiv, mapOptions);
}

function displayLocation(position)
{
    ...
    showMap(position.coords);
}
...
<div id="map"></div>
```

Update #633

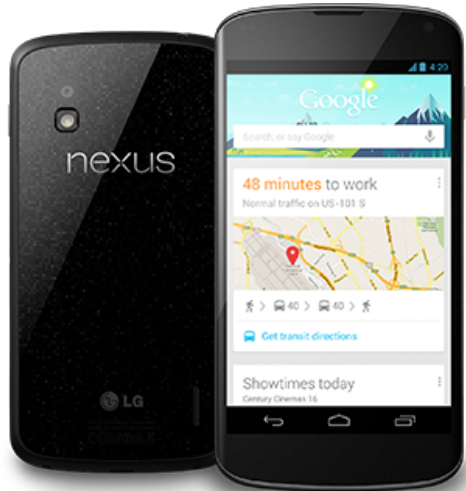
geo5.html:
adding pins

High accuracy:
GPS, Wi-Fi, and
mobile networks



geo5.html:
adding pins

Battery saving:
Wi-Fi and mobile
networks



Update #276

geo5.html:
adding pins

Wi-Fi, GPS, 3G



Conclusions

- **HTML5 + other associated APIs**
 - Enable new and different web apps
 - Location-based services
 - Making browser apps more like desktop apps
 - Threading now supported
- **Possible paper topic:**
 - **Geolocation**
 - Technical details about how it works
 - Especially without GPS
 - Location-based applications
 - Privacy issues

