2º Desarrollo de Aplicaciones Web
Curso 2014/2015

Proyecto Integrado

Aplicación Web en Tiempo Real
Seguimiento y Localización

Salvador Camacho Soto

Indice.

1 - Enlaces.

Aplicación web: http://localizacionjs.herokuapp.com/
GitHub: https://github.com/salvacam/localizacionjs
Documentación: http://salvacam.x10.mx/pi/Doc_.pdf
Presentación: http://slides.com/salvacam/localizacionjs/
App para Android: http://salvacam.x10.mx/pi/app.apk
App WindowsPhone: http://salvacam.x10.mx/pi/app.xap

2 - Objetivos.

Realizare una aplicación web, en tiempo real, para el seguimiento y localización de cualquier elemento móvil que disponga de un dispositivo GPS y conexión a internet. A través de una página web se podrá consultar en tiempo real la administración de dispositivos que tenga asignado un usuario así como la posición exacta del elemento móvil y el seguimiento de la ruta.

3 - Pre-análisis.

Para la realización del proyecto voy a usar el entorno de programación **Node.js**, para la utilización de **websockects** usare la librería de JavaScript **Socket.IO**

Para guardar los datos de posicionamiento usare una base **NoSql** para Node.js, **MongoDB**.

Para la visualización de mapas usare los mapas de **Open Street Map** a través de la librería **Leaflet.**

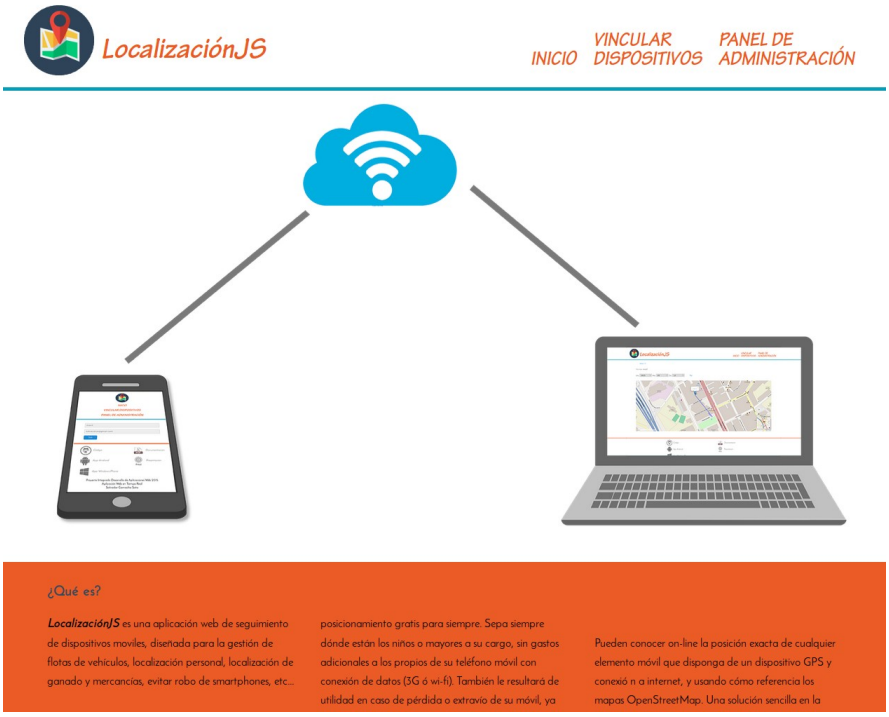
Usare la API de html5 de **geolocalización** para el envió de la posición.

4 - Estimación de costes.

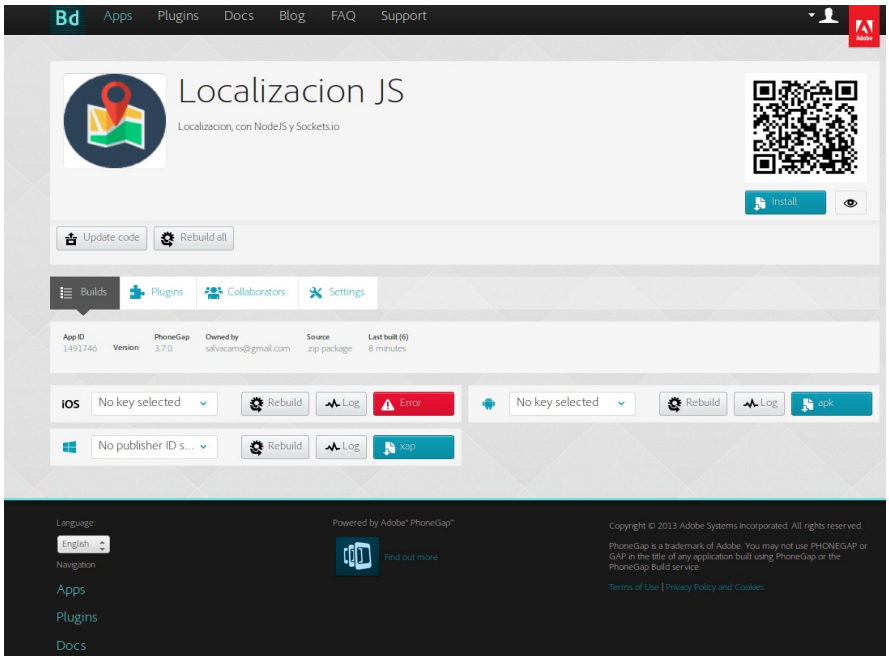
El coste económico sera cero, ya que alojare la aplicación en la plataforma Heroku donde se pueda ejecutar Node.js con una cuenta gratuita. Todas el software usado es gratuito y, en la mayoría de los casos, con Licencia Libre.

5 - Análisis.

La aplicación tendrá uso a través de la web y de los sistemas operativos móviles Android, iOS y Windows Phone. Las aplicaciones móviles las creare adaptando la web para convertirlas con el framework Phonegap.

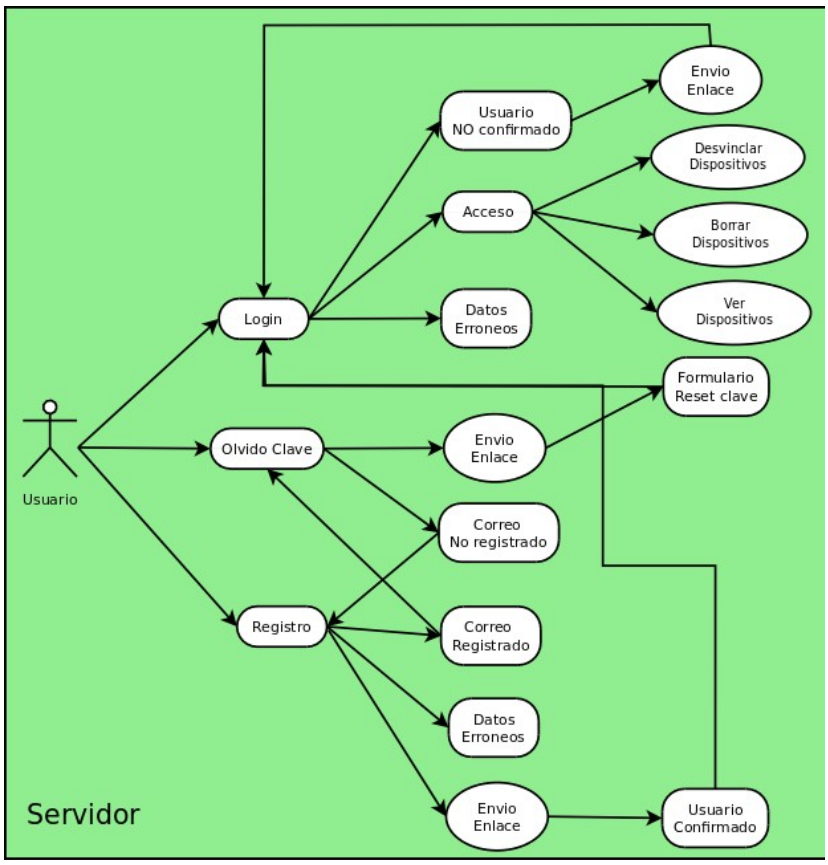


http://localizacionjs.herokuapp.com

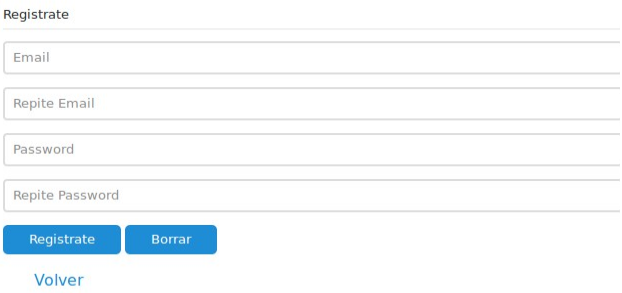


https://build.phonegap.com

El diagrama de uso del usuario



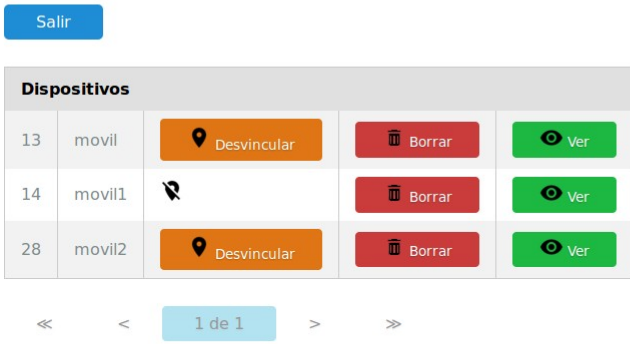
Para el uso de la aplicación se debe registrar el usuario con un correo valido, ya que se enviara un enlace para confirmar al usuario.



Formulario de registro

Solo se podrá tener un usuario por cada cuenta de correo. Si el correo ya esta en uso se mostrara un mensaje de correo en uso y un enlace para acceder al formulario de envió de enlace para cambiar la contraseña.
Se podrá cambiar la contraseña con un formulario, que se enviara el enlace al correo del usuario.

El usuario logeado, y previamente confirmado, podrá administrar los dispositivos que tenga vinculados.

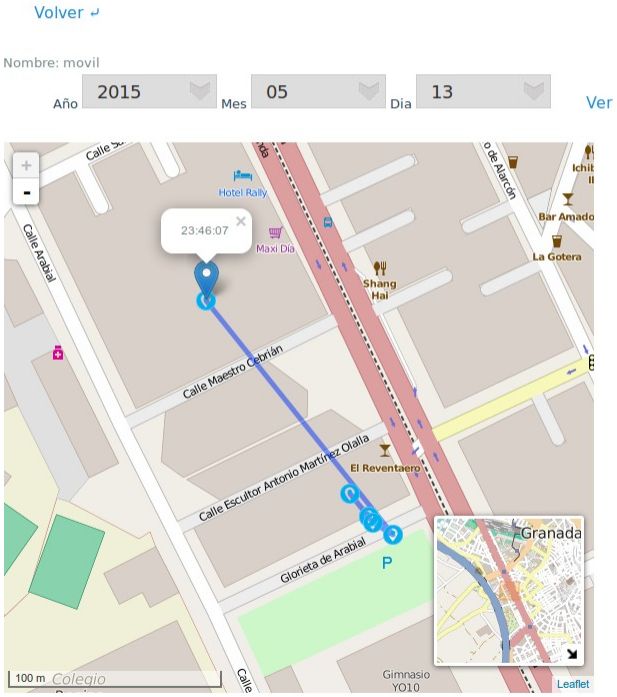


Administración de dispositivos

Esta página funciona en tiempo real, si un dispositivo se crea automaticamente se muestra en la pantalla o si se vincula o desvincula se muestra el cambio sin necesidad de recargar la página.

Se muestran los dispositivos, paginados, y se puede:

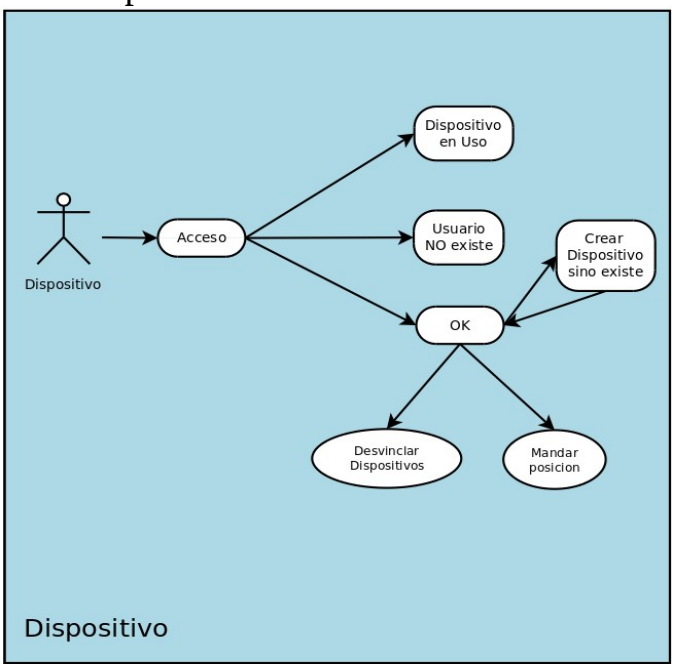
- Desvincular, para que no siga enviando su posición, si la esta enviando.
- Borrar, elimina el dispositivo de la base de datos y todas sus posiciones.
- Ver, muestra la posiciones por día.



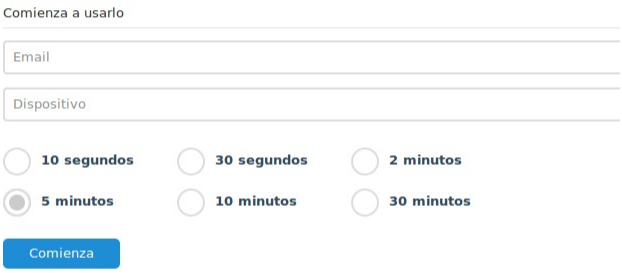
Posición por día de un dispositivo

Esta página funciona en tiempo real, si un dispositivo envía una nueva posición y se esta viendo el día en que se actualiza, se muestra en pantalla la nueva posición, se crea la polilinea y el marcador se mueve a la posición.

El diagrama de uso de un dispositivo



Para que un dispositivo envié su posición, hay que acceder al enlace dispositivo.



Formulario de dispositivo

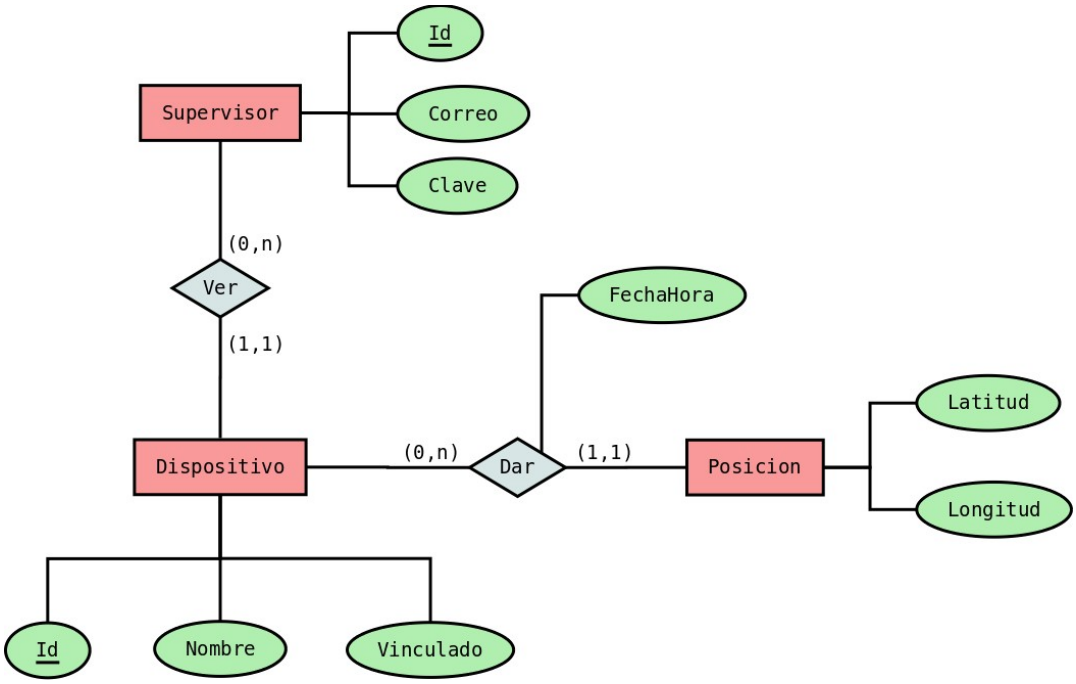
Se tiene que poner el correo del usuario que visualizara el dispositivo y un nombre para el dispositivo, el nombre del dispositivo no tiene que tenerlo el usuario vinculado (en uso en otro dispositivo) y el periodo de tiempo en que se enviara la posición. Si el nombre no existe se crea en la base de datos.

Para que el dispositivo deje de enviar su posición se puede desvincular desde el panel de administración o en el dispositivo, en el botón salir mientras esta enviando su posición.

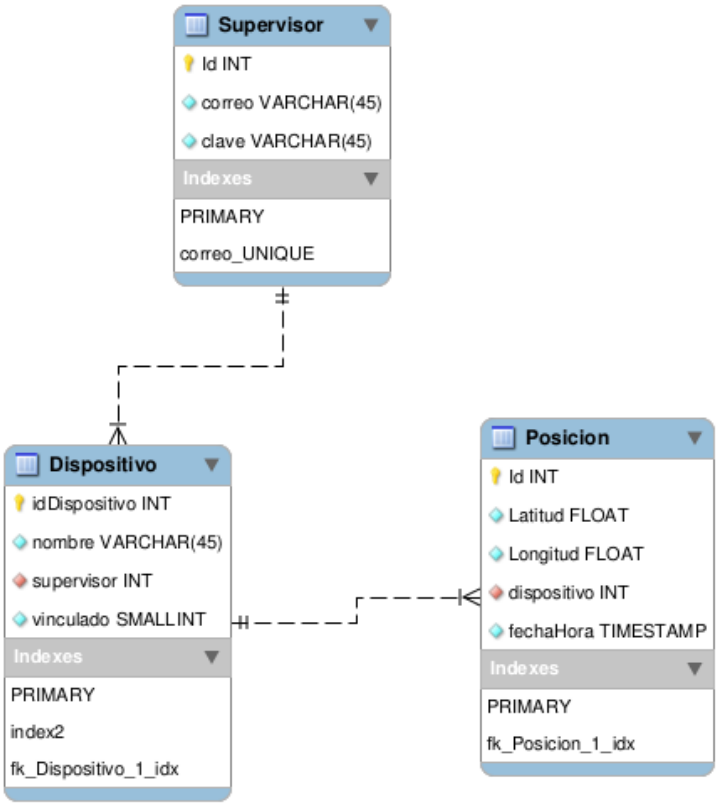
Dispositivo enviando posición

6 - Diseño.

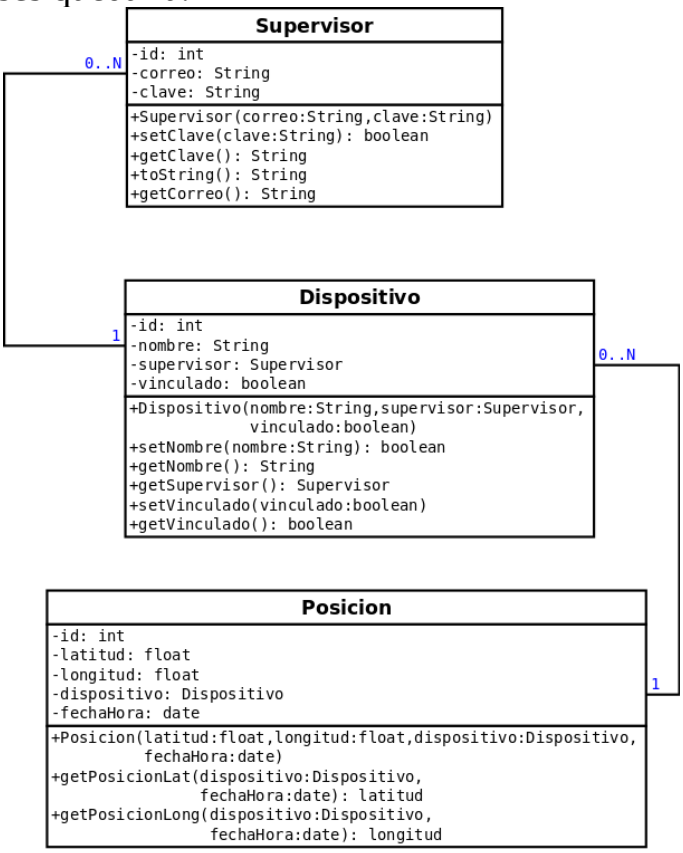
Modelo entidad relación



Tengo tres entidades, con sus relaciones, que me generan el siguiente Diagrama de Tablas (en mi caso objetos) para la base de datos.



El **diagrama de clases** quedaría.



Tendría las siguientes clases **Dispositivos, Supervisor y Posicion.**

7 - Código.

7.1 - Servidor Web.

Para tener un servidor web uso varios archivos. ***Index.js***, es la página principal, es el archivo que se ejecuta, importa varios módulos.

```
var servidor = require("./servidor.js");
var enrutador = require("./enrutador.js");
var gestor = require("./gestor.js");

var gestion = {};
gestion["/"] = gestor.principal;
gestion["/dispositivo"] = gestor.dispositivo;
...
gestion["/ver"] = gestor.ver;
gestion[".css"] = gestor.css;
gestion[".js"] = gestor.js;
gestion[".png"] = gestor.png;

servidor.iniciar(enrutador.enrutar, gestion);
```

El archivo ***enrutador.js*** según la url recibida ejecuta una determinada función del archivo ***gestor.js***

```
var url = require("url");
var path = require("path");

function enrutar(gestion, request, response, post, io) {
   var ruta = url.parse(request.url).pathname;
   if (typeof gestion[ruta] === 'function') {
      gestion[ruta](request, response, post, io);
   } else {
         var extname = path.extname(ruta);
         if(typeof gestion[extname] === 'function'){
                  gestion[extname](request, response, post);
         } else {
                  gestion['/'](request, response);
      }
   }
}
exports.enrutar = enrutar;
```

Para que cargue los archivos de estilos, las imágenes, las tipografías y los scritps. He creado la función ***recurso*** en el archivo ***gestor.js.***

```
function recurso(tipo, request, response) {
   var ruta = configuracion.datos.carpetaPublic + url.parse(request.url).pathname;
```

```
    path.exists(ruta, function(exists) {
        if (exists) {
            fs.readFile(ruta, function(error, contenido) {
                if (error) {
                    response.writeHead(500);
                    response.end();
                } else {
                    response.writeHead(200, {
                        "Content-Type": tipo
                    });
                    response.end(contenido, 'utf-8');
                }
            });
        } else {
            response.writeHead(404);
            response.end();
        }
    });
}

function css(request, response) {
    recurso("text/css", request, response);
}
...
exports.css = css;
```

Cuando una se hace una llamada a un archivo de los tipos ante mencionado se ejecuta la función especifica de cada tipo de archivo y los carga o devuelve un error 404.

7.2 – Plantillas.

Para cargar el archivo ***html***, tengo la función escribir.

```
function escribir(pagina, response) {
    response.writeHead(200, {
        'Content-Type': 'text/html'
    });
    var flujo = fs.createReadStream(configuracion.datos.carpetaPublic + pagina);
    flujo.pipe(response);
};

function principal(request, response) {
    escribir("/index.html", response);
}
function dispositivo(request, response) {
    escribir("/html/dispositivoAcceder.html", response);
}
```

Cuando carga una pagina web, se escribe el contenido de la página que se le pasa.

Para el uso de plantillas, usa el paquete ***replaceStream***.

```
function aviso(aviso, script, response) {
   response.writeHead(200, {
      'Content-Type': 'text/html'
   });
   fs.createReadStream(configuracion.datos.carpetaPublic + "/html/aviso.html")
      .pipe(replaceStream('{Aviso}', aviso))
      .pipe(replaceStream('{Scripts}', script))
      .pipe(response);
};
```

En esta función carga la web ***aviso.html***, pero sustituyendo la cadena ***{Aviso}*** por el parámetro ***aviso*** y sustituyendo la cadena ***{Scripts}*** por el parametro ***script***.

```
<!doctype html>
<html lang="es">
   <head> ...  </head>

   <body>
         <div class="header"> .... </div>
         <div class="content">
                  {Aviso}
         </div>
         <div class="footer l-box is-center"> .... </div>
   </body>
   <script>
            {Scripts}
   </script>
</html>
```

7.3 - Base de Datos.

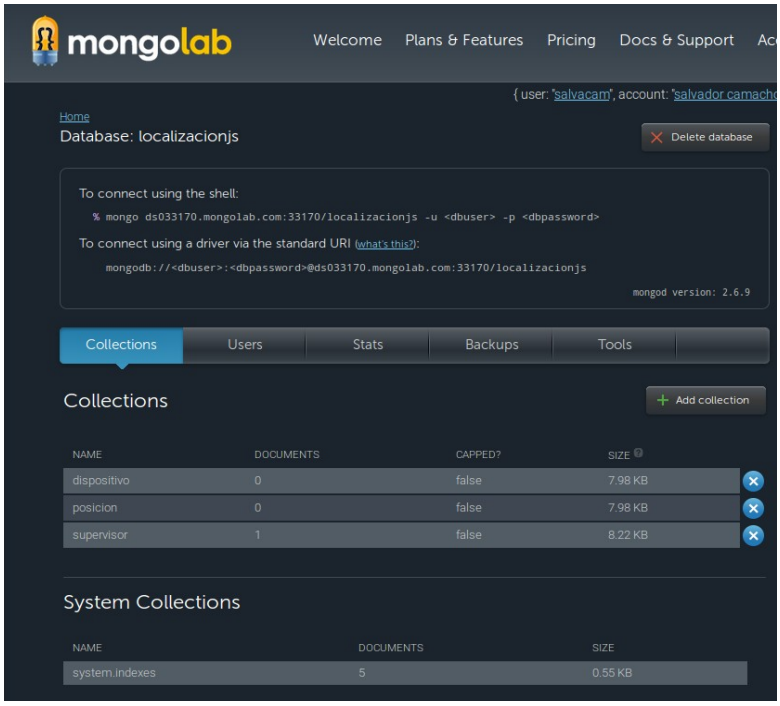
Como base de datos uso ***MongoDB***, es una base de datos orientado a objetos.

Usare el servicio **MongoLab** con una cuenta gratuita, con la que puedo tener una base de datos.

La conexión la realizo en el archivo ***gestor.js***

```
var mongo = require('mongodb').MongoClient;
var uri = 'mongodb://user:password@ds033170.mongolab.com:33170/localizacionjs';
```

Las colecciones (seria equivalente a tabla en SQL) se crean desde la web de ***MongoLab.***



Para una consulta para mostrar todo el contenido de una colección uso el método ***find()***.

```
supervisorTabla.find().toArray(function(err, docs) {
      for (var x in docs) {
      }
  });
```

Para una consulta para mostrar solo una tupla de una colección uso el método ***findOne()***.

```
supervisorTabla.findOne({
      correo: querystring.parse(post).correo
  }, function(err, item) {
          ...
      }
  });
```

Para crear una tupla en una colección uso el método ***insert()***.

```
supervisorTabla.insert({
            correo: supervisorNuevo.correo,
            clave: supervisorNuevo.clave,
            activo: supervisorNuevo.activo
        });
```

Para actualizar una tupla se usa el método ***update()***.

```
supervisorTabla.update({
                  correo: correo
```

```
            }, {
                correo: correo,
                clave: clave,
                activo: '0'
            });
```

Para borrar una tupla se usa el método ***remove()***.

```
dispositivoTabla.remove({
                _id: getParametros(request).disp
            });
```

7.4 – Correo.

Con el modulo ***nodemailer*** configurado puedo usar una cuenta de gmail.

```
var nodemailer = require('nodemailer');

var gmail = nodemailer.createTransport({
   service: 'Gmail',
   auth: {
      user: configuracion.datos.gmailCorreo,
      pass: configuracion.datos.gmailPass
   }
});
```

Para enviar correos uso la función ***enviarCorreo***.

```
function enviarCorreo(destino, opcion, enlace) {
   var asunto = "";
   var contenido = "";
   if (opcion == "activar") {
      asunto = 'Activa la cuenta';
      contenido = 'Activa tu cuenta en este ';
   } else if (opcion == "reestablecer") {
      asunto = 'Activa la cuenta';
      contenido = 'Activa tu cuenta en este ';
   }

   var mailOptions = {
      from: 'Web LocaliacionJS <programacion211@gmail.com>', // sender address
      to: destino,
      subject: asunto,
      text: contenido + enlace,
      html: contenido + enlace
   };

   gmail.sendMail(mailOptions, function(error, info) {
```

```
      if (error) {
         return;
      }
   });
}
```

7.5 – Sesión.

Para controlar la sesión del administrador, creo dos pares de llave - valor cuando se loguea

```
sessionStorage.setItem("correo", document.getElementById("correo").value);

sessionStorage.setItem("id", "' + md5(configuracion.datos.pezarana +
getHora(querystring.parse(post).correo)) + '")
```

Al volver a una página del administrador se comprueba que el parámetro id, que se pasa en la url es el par de sessionStorage que se ha creado anteriormente.

```
if (getParametros(request).id == (md5(configuracion.datos.pezarana +
getHora(getParametros(request).email))) && activo == "0") {
         ...
}else {
   aviso("Error al introducir contraseña", '<script>  sessionStorage.removeItem("correo")
</script>', response);
         }
```

Y en el cliente, en el archivo ***admin.html*** se redirecciona a la pagina de adminstrador si se esta logueado

```
if (sessionStorage.getItem("id") !== null && sessionStorage.getItem("correo") !== null) {
   window.location = "supervisor?id=" + sessionStorage.getItem("id") + "&email=" +
sessionStorage.getItem("correo");
}
```

Al salir de la sesión elimino todos los pares, en el archivo ***accesoAdmin.html***. Al usar sessionStorage también se elimina al cerrar el navegador.

```
sessionStorage.clear();
```

7.6 - Dispositivo.

El dispositivo cada periodo de tiempo indicado cuando se vincula, manda la posición al servidor con una llamada ajax, en el archivo ***dispositivo.html***.

```
ajax.open("GET",
         "http://localizacionjs.herokuapp.com/ajaxSend?disp=" +
document.getElementById("idDispositivo").value + "&lat=" + posicion.coords.latitude +
```

```
"&long=" + posicion.coords.longitude + "&date=" + fechaHora,
        true);
    ajax.onreadystatechange = function() {
      if (ajax.readyState == 4) {
        if (ajax.status == 200) {
          var ojsonBorrar = JSON.parse(ajax.responseText);
          if (ojsonBorrar.r == 1) {
            clearInterval(temp);
            alertify.confirmA("Dispositivo desvinculado por el supervisor", function(e) {
              window.onbeforeunload = function() {};
              window.location = "index";
            });
          }
          if (ojsonBorrar.r == 2) {
            clearInterval(temp);
            alertify.confirmA("No existe el dispositivo", function(e) {
              window.onbeforeunload = function() {};
              window.location = "index";
            });
          }
        }
      }
    };
    ajax.send();
```

Si el administrador ha borrado o desvinculado el dispositivo se muestra un mensaje, usando la librería ***Alertify.js***, he modificado esta librería para que muestre un confirm con solo el botón de Ok, cuando se pulsa te redirige al inicio de la aplicación.

Para desvincular y borrar dispositivos desde el panel de administración también se usa ajax, en el archivo ***accesoAdmin.html***.

```
ajax.open("GET", "http://localizacionjs.herokuapp.com/ajaxDesvincular?disp=" + id, true);
          ajax.onreadystatechange = function() {
            if (ajax.readyState == 4 && ajax.status == 200) {
              var pag = 0;
              if (window.location.search != "") {
                var str = window.location.search;
                if (str.indexOf("&pagina=") != -1) {
                  pag = str.substring((str.indexOf("&pagina=")) + 8);
                }
              }
              llamadaAjax(pag);
            }
          };
          ajax.send();
```

```
ajax.open("GET", "http://localizacionjs.herokuapp.com/ajaxBorrar?disp=" + id, true);
                ajax.onreadystatechange = function() {
                   if (ajax.readyState == 4 && ajax.status == 200) {
                      var ojsonBorrar = JSON.parse(ajax.responseText);
                      if (ojsonBorrar.r == 0) {
                         alertify.alert("Se ha borrado el dispositivo");
                         var pag = 0;
                         if (window.location.search != "") {
                            var str = window.location.search;
                            if (str.indexOf("&pagina=") != -1) {
                               pag = str.substring((str.indexOf("&pagina=")) + 8);
                            }
                         }
                         llamadaAjax(pag);
                      } else {
                         alertify.alert("No se ha podido borrar el dispositivo");
                      }
                   }
                };
                ajax.send();
```

7.7 - Socket.io

Socket.io es una librería en JavaScript para Node.js que permite una comunicación bidireccional en tiempo real entre cliente y servidor. Para ello se basa principalmente en Websocket pero también puede usar otras alternativas como sockets de Adobe Flash, JSONP polling o long polling en AJAX, seleccionando la mejor alternativa para el cliente justo en tiempo de ejecución.

En el archivo ***servidor.js*** inicio socket.io, definiendole el servidor y pasandolo como parametro al enrutador para usarlo en el archivo ***gestor.js.***

```
   var app = http.createServer(atiendePeticion).listen(port);
   var io = require('socket.io').listen(app);
...
         enrutador.enrutar(gestion, request, response, post, io);
```

En el archivo ***gestor.js*** inicio socket.io.

```
var io = require('socket.io').listen(configuracion.servidor);
```

Y defino cada vez que se envíen datos al cliente.

```
io.sockets.emit("Supervisor_" + idSupervisor, "refrescar-enviando");
                   supervisorTabla.findOne({
                      _id: idSupervisor
                   }, function(err, item) {
                      viewDispositivo(nombre, item.correo, idDisp, idSupervisor, tiempo,
```

```
response);
                        });

io.sockets.emit("Supervisor_" + idSupervisor, "refrescar-creado");

io.sockets.emit("Supervisor_" + idSup, "refrescar-desvinculado");

io.sockets.emit(getParametros(request).disp, datos);
```

En el cliente, en el archivo ***ver.js*** y ***accesoAdmin.html*** cargo la librería socket.io.js defino la url que continene la librería donde me quiero conectar.

```
<script src="http://localizacionjs.herokuapp.com/socket.io/socket.io.js"></script>
var socket = io.connect('http://localizacionjs.herokuapp.com/');
```

Después en ***accesoAdmin.html*** defino el escuchador, cuando se emite con la llave que tenga definida se ejecuta la función, los datos enviados estan en el parametro ***data***

```
socket.on("{ValorSupervisor}", function(data) {
   var pag = 0;
   if (window.location.search != "") {
      var str = window.location.search;
      if (str.indexOf("&pagina=") != -1) {
         pag = str.substring((str.indexOf("&pagina=")) + 8);
      }
   }
   llamadaAjax(pag);
});
```

Este caso es cuando se hay que refrescar los datos del panel de administración.

```
function llamadaAjax(pagina) {
   var ajax1;
   if (window.XMLHttpRequest) {
      ajax1 = new XMLHttpRequest();
   }
   ajax1.open("GET",
      "http://localizacionjs.herokuapp.com/ajaxRefrescar?correo=" + correo + "&sup=" + id
+ "&pag=" + pagina,
      true);
   ajax1.onreadystatechange = function() {
      if (ajax1.readyState == 4) {
         if (ajax1.status == 200) {
            var ojsonRef = JSON.parse(ajax1.responseText);
            if (ojsonRef.tabla != "") {
               document.getElementById("contenido").innerHTML = ojsonRef.contenido;
               document.getElementById("paginacion").innerHTML = ojsonRef.enlacesPag;
               eventos();
```

```
            }
        }
    }
  };
  ajax1.send();
}
```

En el archivo ***ver.js*** también defino el escuchador.

```
  socket.on(id, function(data) {
        ....
});
```

Actualizaría el mapa si estoy viendo el día actual.

7.8 - Mapa

Para mostrar el mapa uso la librería ***leaflet***, con la librería ***Control MiniMap*** le añado el minimapa en la esquina derecha inferior y para animar el marcador uso la librería ***AnimatedMarker***. Tengo que cargar estas librerías en el archivo ***ver.html***.

```
<script src="../js/vendor/leaflet.js"></script>
<script src="../js/vendor/Control.MiniMap.min.js"></script>
<script type="text/javascript" src="../js/vendor/AnimatedMarker.js"></script>
```

Durante la animación del marcador encontré el fallo de que se modificaba el nivel de zoom, durante cierto tiempo se separaba el marcador de la polilinea. Por lo que mientras se ejecutaba la animación desactivaba el control del zoom.

```
function animacion(poly, elemento){
      map.scrollWheelZoom.disable();
      map.doubleClickZoom.disable();
      map.zoomControl.removeFrom(map);

      $( "#ver" ).addClass("pure-button-disabled");

      animatedMarker = L.animatedMarker(poly.getLatLngs(), {
         distance: 300,  // meters
         interval: 1000, // milliseconds
         autoStart: false,
         onEnd: function(){
            map.scrollWheelZoom.enable();
            map.doubleClickZoom.enable();
            map.zoomControl.addTo(map);

            map.removeLayer(animatedMarker);

            $( "#ver" ).removeClass("pure-button-disabled");
```

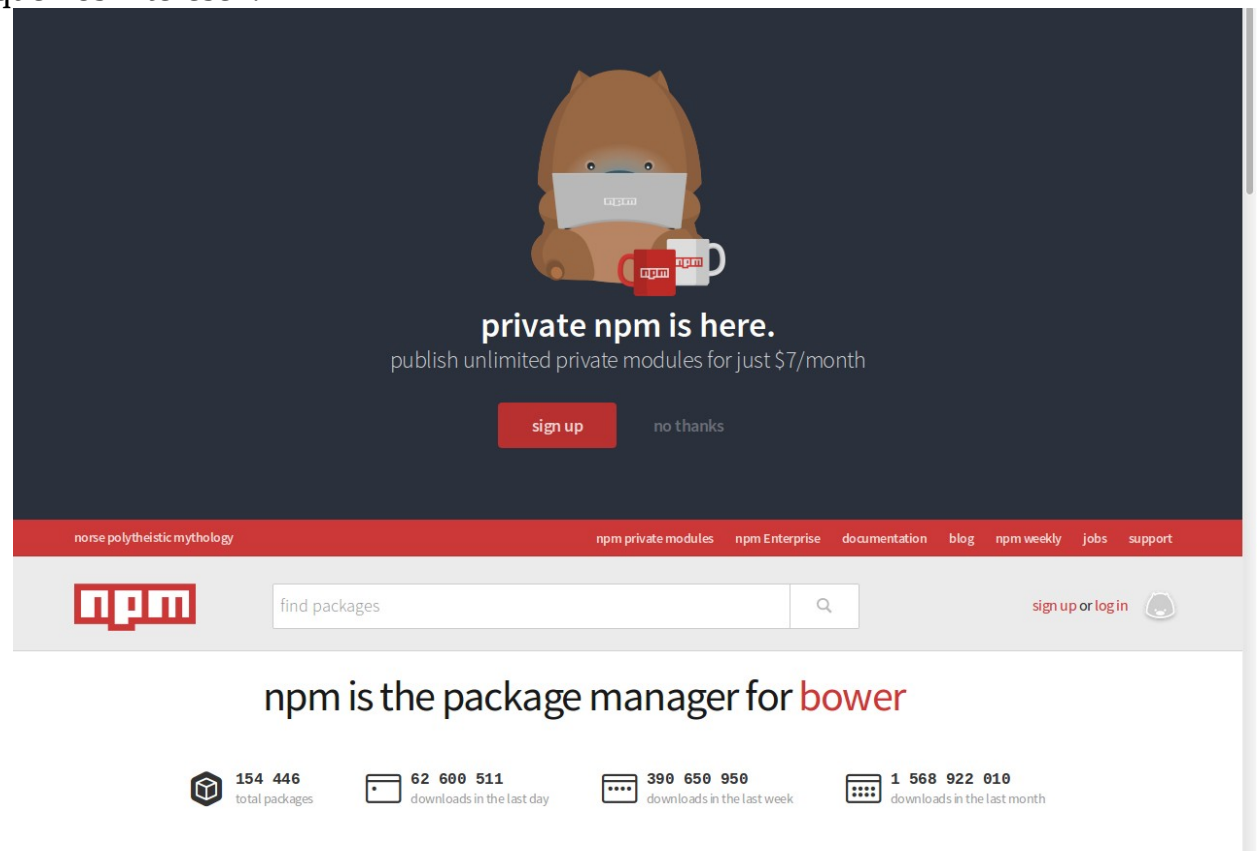
```
            lat = parseFloat(elemento.latitud);
            lon = parseFloat(elemento.longitud);
            fh = elemento.fechaHora.substring(elemento.fechaHora.length - 8);

            marcador1 = L.marker([lat, lon]).addTo(map)
               .bindPopup(fh)
               .openPopup();
         }
      });
      map.addLayer(animatedMarker);
      setTimeout(function() {
         animatedMarker.start();
      }, 500);
   }
```

8 - Implementación.

8.1 - NPM.

Node Packaged Modules es el ecosistemas de gestión de paquetes de Node.js.
Se pueden crear, compartir y reutilizar módulos para Node.js.

El sitio Web oficial de npm es http://npmjs.org . Hay podemos buscar los modulos que nos interesen.



En mi aplicación uso como paquetes

```
var replaceStream = require('replacestream');
var querystring = require("querystring");
var path = require("path");
var sha1 = require('sha1');
var md5 = require('MD5');
var Db = require('tingodb')().Db;
var nodemailer = require('nodemailer');
```

Para el uso de estos paquetes tengo que haberlos instalados previamente.

```
npm install <paquete>
```

Esto crea una carpeta ***node_modules***, dentro de la cual me crea una carpeta con el nombre del paquete que instalo.

Para evitar tener que instalar todos los paquetes uno a uno se utiliza el archivo package.json. En el apartado dependencies, especifico todos los módulos necesarios y su versión.

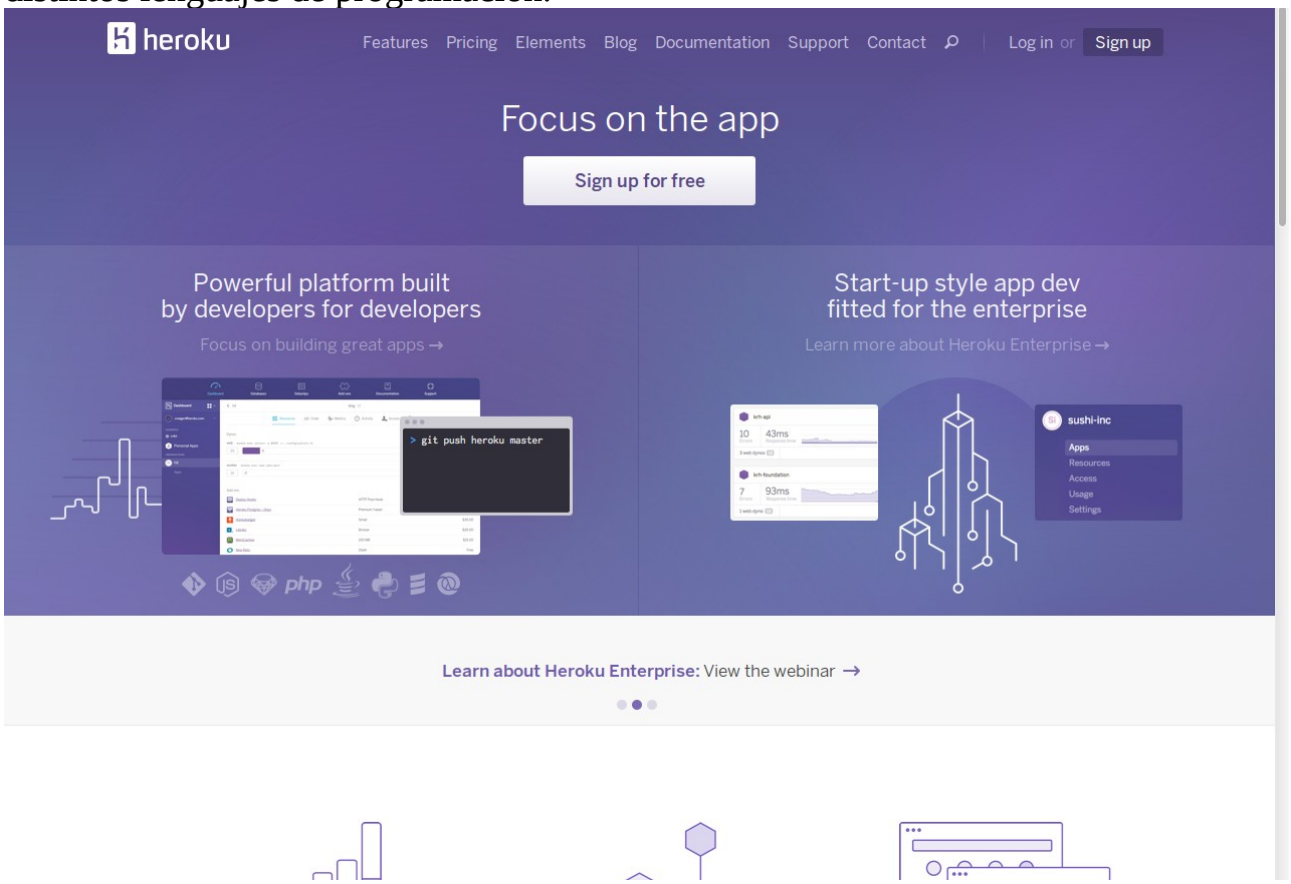
```
{
 "name": "locationJS",
 "version": "0.0.1",
 "private": true,
 "main": "index.js",
 "dependencies":{
  "tingodb":"0.3.4",
  "sha1":"1.1.0",
  "MD5": "1.2.1",
  "nodemailer": "1.3.2",
  "replacestream": "2.1.0",
  "socket.io": "1.3.5"
 },
 "devDependencies": {},
 "scripts" : {
  "start": "node index.js"
 },
 "engines": {
  "node": "0.10.x",
  "npm": "1.3.x"
 },
 "author": "Salvador Camacho Soto"
}
```

Ahora para instalar todos los módulos solo necesito ejecutar

```
npm install
```

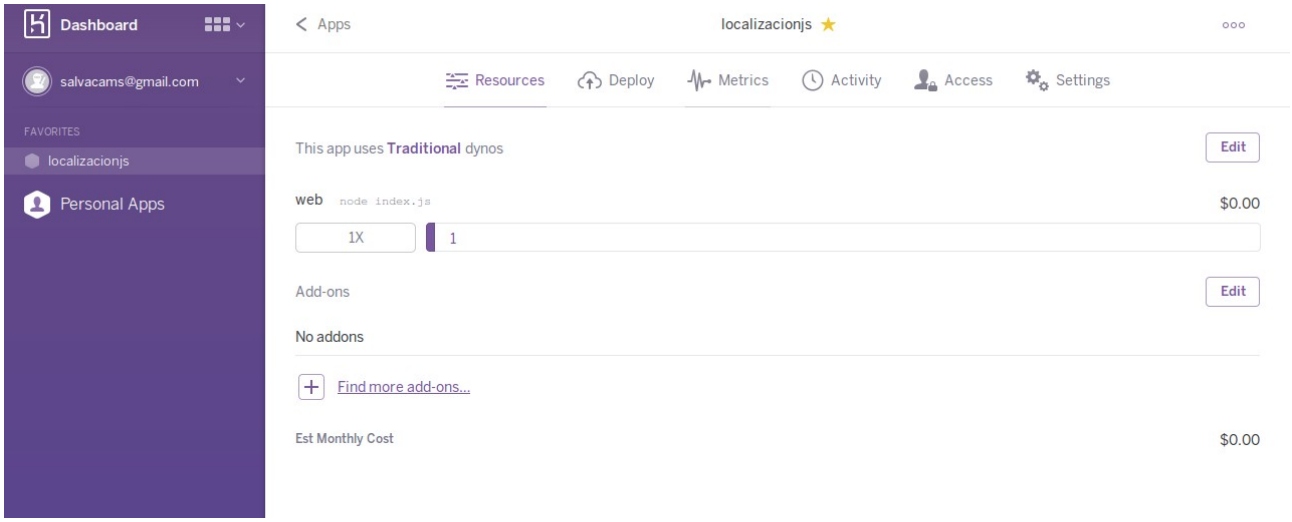
8.2 - Heroku.

Heroku es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.



Uso esta plataforma para alojar mi aplicación, con un servicio gratuito, que me ofrece un maquina virtual (1 Dynos), 300 MBytes HD. El servicio gratuito se desactiva después de 30 minutos de inactividad, y esta inactiva 6 horas por dia.

Las aplicaciones se gestionan desde linea de comando, instalando ***Heroku Toolbelt*** o desde la web accediendo al ***Dashboard.***



Para crear una aplicación con Heruku, primero hay que instalar Heroku Toolbelt, desde la página web https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction, donde hay versiones para Mac, Windows y Debian

Desde el terminal ejecutamos

```
$ heroku login
```

Donde tendremos que poner las mismas credenciales con la que estoy registrado.

Podría crear la aplicación desde terminal y re nombrarla, ya que nos da un nombre al azar.

```
$ heroku create
$ heroku apps:rename localizacionjs
```

O crearla desde la web accediendo al ***Dashboard,*** en la web https://dashboard.heroku.com/new

Para adaptar la aplicación a *Heroku*, se crea un archivo ***Procfile***. Es el fichero de arranque de *Heroku*, donde se indica el comando con el parámetro a ejecutar.

```
web: node index.js
```

En el archivo ***package.json*** se incluyen las lineas

```
 "engines": {
   "node": "0.10.x",
   "npm": "1.3.x"
 },
```

donde se indica la versión de *Node.js* y *npm* que usamos.

En el archivo ***servidor.js***, se cambia el puerto del servidores

```
var port = process.env.PORT || 8888;
```

Para ejecutar la aplicación en local, se puede usar el comando

```
$ node index.js
```

y acceder a http://localhost:8888

O ejecutar

```
$ foreman start web
```

y acceder a http://localhost:5000

Para subir la aplicación se usa git, por lo que es necesario tenerlo instalado.
En la carpeta donde tengo la aplicación ejecuto

```
$ git init
$ git add .
$ git commit -m “subida”
$ heroku git:remote -a localizacionjs
$ git push heroku master
```

Si se hace algun cambio, se ejecuta

```
$ git add .
$ git commit -m “subida 2”
$ git push heroku master
```

He creado un archivo con el nombre ***.gitignore***, cuyo contenido es los archivos y carpetas que no quiero subir al repositorio de *Heroku*, así cada vez que quiero subir algún cambio uso

```
$ git add .
```

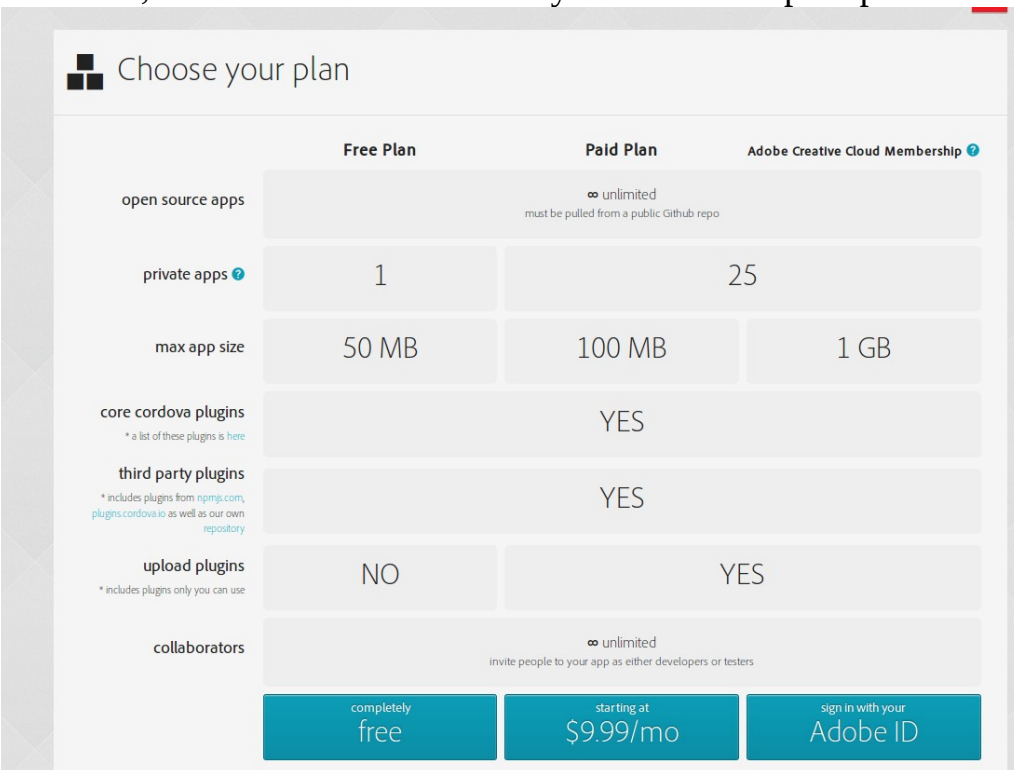
Para añadir todos los archivos, menos los contenidos en .gitignore y no tener que indicar uno a uno.

8.3 - Phonegap.

PhoneGap es un framework para el desarrollo de aplicaciones móviles, permite desarrollar aplicaciones para dispositivos móviles utilizando como herramientas JavaScript, HTML5 y CSS3.

PhoneGap Build es un servicio en la nube que permite generar los paquetes de distribuciones de cada plataforma, listo para la distribución final en cada uno de los markets de cada proveedor.

Dispone de varios planes, el plan gratuito permite una aplicación privada e ilimitadas aplicaciones libres, con un máximo de 50 MBytes de tamaño por aplicación.



Una vez creada la cuenta en https://build.phonegap.com/, tenemos que añadir la aplicación. Tenemos dos opciones, indicar un repositorio de código fuente GitHub o subir un archivo en formato ZIP que debe incluir nuestro proyecto web.

Mi aplicación para Phonegap solo tendra un archivo index.html

```
<!DOCTYPE html>
<html lang="es">
	<head>
		<meta http-equiv="content-type" content="text/html; charset=utf-8" />
		<meta name="viewport" content="width=device-width" />
		<title>Localizacion JS</title>
```

```
	</head>

	<body>
		<div id="viewsites">
			<iframe src="http://localizacionjs.herokuapp.com" id="iframeId"
marginheight="0" marginwidth="0" align="top" frameborder="0"
scrolling="no"></iframe>
		</div>
		<script>
			var height = window.innerHeight ||
document.documentElement.clientHeight || document.body.clientHeight || $
('body').innerHeight();
			document.getElementById('iframeId').style.height = height + 'px';
			document.getElementById('iframeId').style.minWidth = '100%';
		</script>
	</body>
</html>
```

Es una etiqueta iframe cuyo fuentre es la aplicación web subida a Heroku.

Empaquetado en formato zip el archivo index.html y lo subo a PhoneGap Build. Me genera un archivo .apk para Android, y un archivo .xap para Windows Phone, para generara el archivo para iOS necesito un id de desarrollador de iOS.

9 – Recursos e Implementación.

Para el uso de *Node.js*

"The Node Beginner Book" de *Manuel Kiessling*

"Node.js in 24 hours" de *George Ornbo*

Para la busqueda de modulos para Node.js

https://www.npmjs.com/

Para el envío de correo a través de Gmail con Node.js

http://www.nodemailer.com/

Para el uso de la librería *Socket.io*

http://socket.io/docs/#

Para el uso de la base de datos *TingoDB*

http://codehero.co/series/mongodb-desde-cero.html

http://docs.mongolab.com/

https://devcenter.heroku.com/articles/mongolab

Para la visualización de mapas

http://leafletjs.com/reference.html

https://github.com/openplans/Leaflet.AnimatedMarker

https://github.com/Norkart/Leaflet-MiniMap

Para el uso de Heroku

http://webaficionado.blogspot.com.es/2013/09/subiendo-apliacion-nodejs-heroku.html

https://devcenter.heroku.com/articles/getting-started-with-nodejs#introduction

Para las API de hmtl5 utilizare apuntes de clases y dos libros,

"Los Api JavaScript de HTML5" de *"Eni ediciones"*

"El gran libro de HTML5, CSS3 y JavaScript" de *Juan Diego Gauchat*