

**Desarrollo y
reutilización de
componentes de
software mediante
lenguajes de guión**



Contenido

Especificidad de los selectores	9
Ejercicio Propuesto:.....	11
Angular	17
Estructura de documentos:	17
Vista1	17
Vista2.....	17
Index.html.....	18
Inicio.html	21
Ahora el index.html	25
Menú de navegación	26
jQuery	28
AngulaJS	29
En Listas.html	31
Two-way Data Binding - AngularJS	31
Listas de Tareas	33
Directiva ngRepeat (ng-repeat)	34
Aplicar estilos	35
Agregar tareas.....	36
Restante	39
Eliminar	40
CON FIREBASE	42
Instrucciones	43
Funciones	43
Tipos de datos	43
Números	43
Cadenas de texto	44
Booleanos	44
Arrays	51

JavaScript Timing Eventos	52
El setInterval (Método).....	52
Sintaxis	52
Ejemplo	53
BUCLES	65
For	65
Número múltiplos de 3 entre el 1 y el 100	68
Do...while	69
Break y continue	70
RELOG DIGITAL	71
PROGRAMA QUE DETERMINA SI UN NÚMERO ES PRIMO	72
Acceso al DOM	73
HTML	73
CSS	73
JAVASCRIPT	73
Manipular los nodos.....	75
HTML	75
CSS.....	75
JAVASCRIPT	76
JavaScript Cookies	78
HTML	78
Set a Cookie.....	78
Get a Cookie	78
API Canvas.....	79
SVG <rect>	99
SVG <circle>.....	100
SVG <ellipse>.....	100
SVG <line>	101
SVG <polygon>	101
SVG <polyline>	103
SVG <path>	103

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

SVG <text>	105
SVG Stroke	106
SVG Drop Shadows	107
Hudir la Flota	109
JUEGO DEL AHORCADO	121
Propiedades de String	121
Métodos de String	121
Caso Práctico	125
Presupuesto online	125
Calendario	126







Especificidad de los selectores

<http://specificity.keegan.st/>

Specificity Calculator

A visual way to understand [CSS specificity](#). Change the selectors or paste in your own.

`li:first-child h2 .title`

0	0	2	2
Inline styles	IDs	Classes, attributes and pseudo-classes	Elements and pseudo-elements

+ Duplicate

`#nav .selected > a:hover`

0	1	2	1
Inline styles	IDs	Classes, attributes and pseudo-classes	Elements and pseudo-elements

+ Duplicate

La especificidad de los selectores es calculada como sigue:

- Contar el número de selectores `Id` (= a)
- Contar el número de selectores `.clase`, atributos, y pseudo-clases (= b)
- Contar el número de elementos y pseudo-elementos (= c)
- Se ignora el selector universal

(Los selectores dentro de la pseudo-clase negación `:not(X)` son contado como cualquier otro, pero la negación en sí misma no cuenta como una pseudo-clase.)

Concatenando los tres números a-b-c da la especificidad.

Ejemplos:

*	a=0 b=0 c=0 -> specificity = 0
LI	a=0 b=0 c=1 -> specificity = 1
UL LI	a=0 b=0 c=2 -> specificity = 2
UL OL+LI	a=0 b=0 c=3 -> specificity = 3
H1 + *[REL=up]	a=0 b=1 c=1 -> specificity = 11
UL OL LI.red	a=0 b=1 c=3 -> specificity = 13
LI.red.level	a=0 b=2 c=1 -> specificity = 21
#x34y	a=1 b=0 c=0 -> specificity = 100

Ejercicio Propuesto:

Para el siguiente código **HTML**:

```
<main>
  <div id="contenedor1">
    <div id="contenedor2">
      <ul class="lista">
        <li class="elemento1">Elemento1</li>
        <li class="elemento2">Elemento2</li>
        <li class="elemento3">Elemento3</li>
      </ul>
    </div>
  </div>
</main>
```

Le asignamos el siguiente **CSS**, ¿de qué color se verá el último elemento de la lista (class="elemento3")?

```
#contenedor1 > #contenedor2 .elemento3 {
  color: #000;
}

main > div > div#contenedor2 > ul > li.elemento3 {
  color: #0f0;
}

#contenedor1 .elemento3 {
  color: #f00;
}

div > ul.lista > li:last-child {
  color: #ff0;
}

ul.lista > li.elemento3 {
  color: #0f0;
}

li:last-child {
  color: #f00;
}

div > ul > li {
  color: #00f;
}
```

All CSS Pseudo Elements All CSS Pseudo Classes

Selector
<u>::after</u>
<u>::before</u>
<u>::first-letter</u>
<u>::first-line</u>
<u>::selection</u>

Selector
<u>:active</u>
<u>:checked</u>
<u>:disabled</u>
<u>:empty</u>
<u>:enabled</u>
<u>:first-child</u>
<u>:first-of-type</u>
<u>:focus</u>
<u>:hover</u>
<u>:in-range</u>
<u>:invalid</u>
<u>:lang(<i>language</i>)</u>
<u>:last-child</u>
<u>:last-of-type</u>
<u>:link</u>
<u>:not(selector)</u>
<u>:nth-child(n)</u>
<u>:nth-last-child(n)</u>
<u>:nth-last-of-type(n)</u>
<u>:nth-of-type(n)</u>
<u>:only-of-type</u>
<u>:only-child</u>
<u>:optional</u>
<u>:out-of-range</u>
<u>:read-only</u>
<u>:read-write</u>
<u>:required</u>
<u>:root</u>
<u>:target</u>
<u>:valid</u>
<u>:visited</u>

STARTER	REGULAR	PLUS	ENTERPRISE
\$7 /mo	\$19 /mo	\$79 /mo	\$125 /mo
Basic customer support for small business	Basic customer support for small business	Basic customer support for small business	Basic customer support for small business
10 projects 20 Pages 20 Emails 100 Images	15 projects 40 Pages 40 Emails 200 Images	Unlimited projects 100 Pages 100 Emails 700 Images	Unlimited projects Unlimited Pages Unlimited Emails Unlimited Images
Get started	Get started	Get started	Get started

FAQ

Frequently Ask Questions

What is Guide?

Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts. Separated they live in Bookmarksgrove right at the coast of the Semantics, a large language ocean.

What language are available?

Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts. Separated they live in Bookmarksgrove right at the coast of the Semantics, a large language ocean.

I have technical problem, who do I email?

Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts. Separated they live in Bookmarksgrove right at the coast of the Semantics, a large language ocean.

Can I have a username that is already taken?

Far far away, behind the word mountains, far from the countries Vokalia and Consonantia, there live the blind texts. Separated they live in Bookmarksgrove right at the coast of the Semantics, a large language ocean.

TREK NEPAL

A brief description of the tour.

EXPLORE TOUR

EXPLORE ANTELOPE CANYON

A brief description of the tour.

EXPLORE TOUR

SCALE SIERRA NEVADA

A brief description of the tour.

EXPLORE TOUR

BOOK YOUR NEXT ADVENTURE TODAY.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- All Inclusive Packages
- Multi-Night Stays
- Equipment Provided

BOOK A TOUR

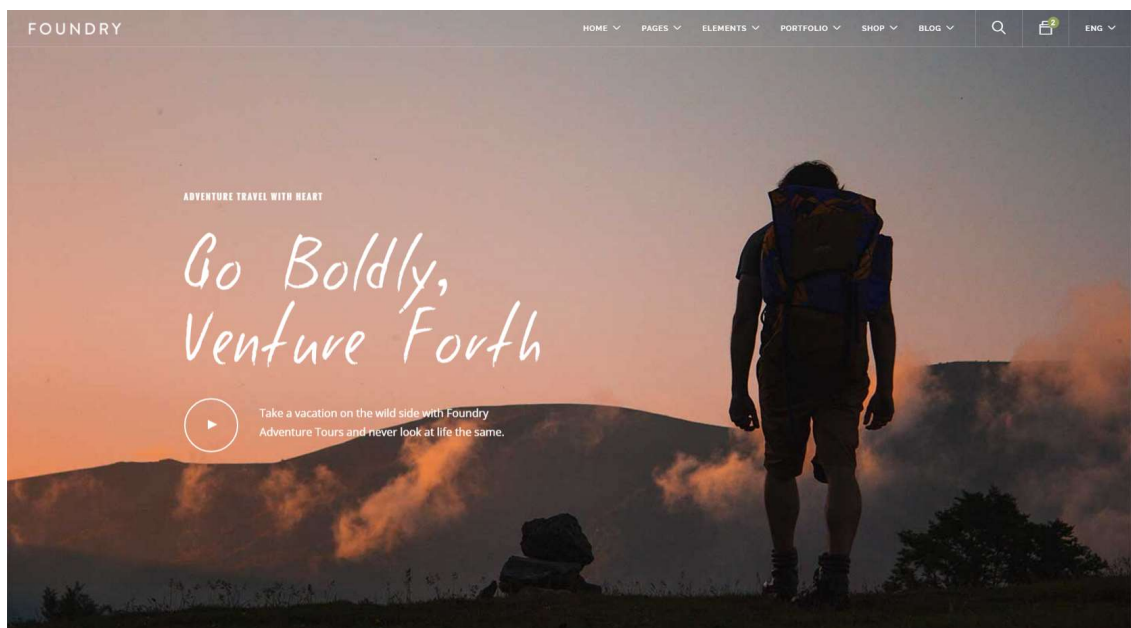
Keep it locked

EXCLUSIVE OFFERS, TRAVEL TIPS & MORE

EMAIL ADDRESS

KEEP ME INFORMED

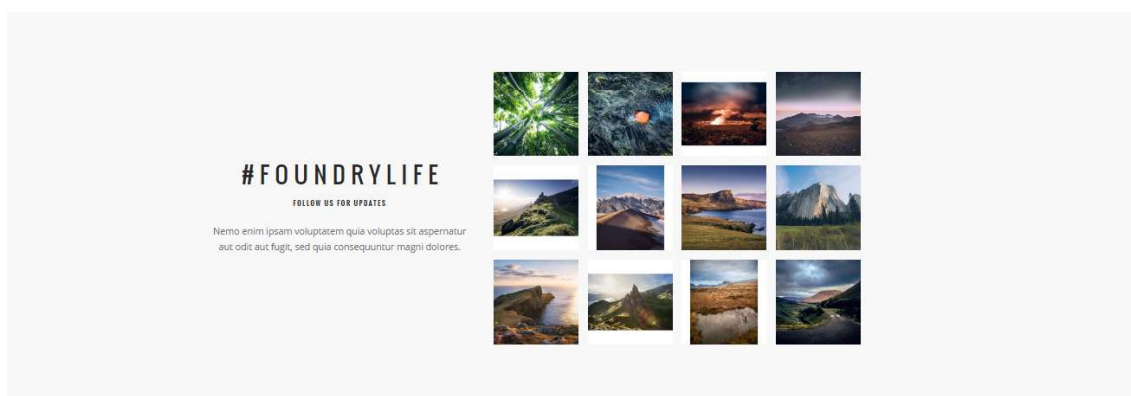
*Newsletters are sent bi-monthly



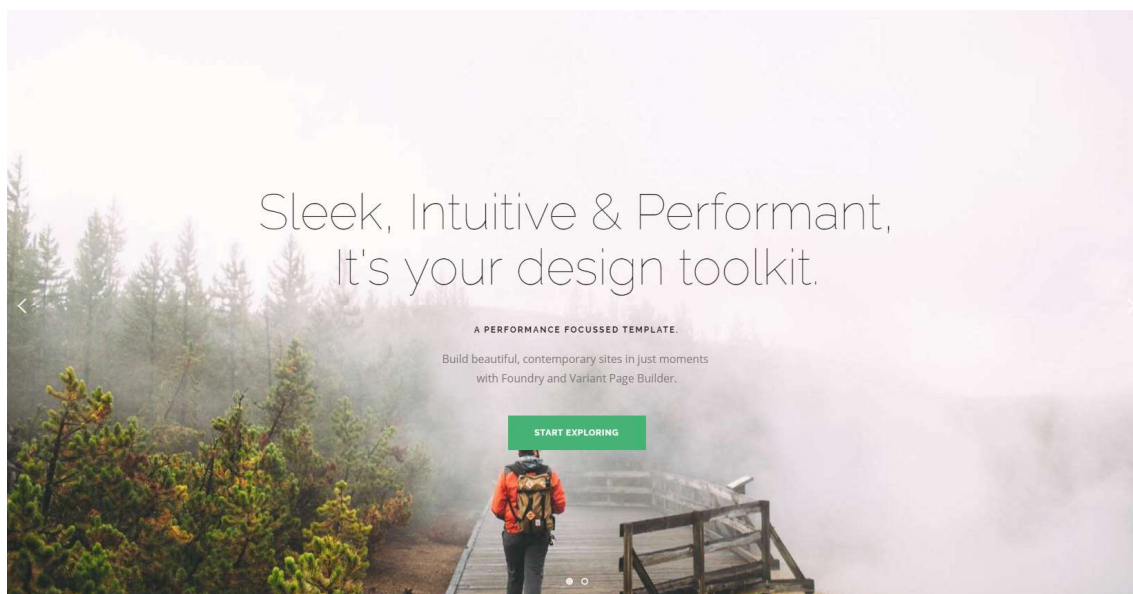
SMASH YOUR COMFORT ZONE

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores.

Card Craft

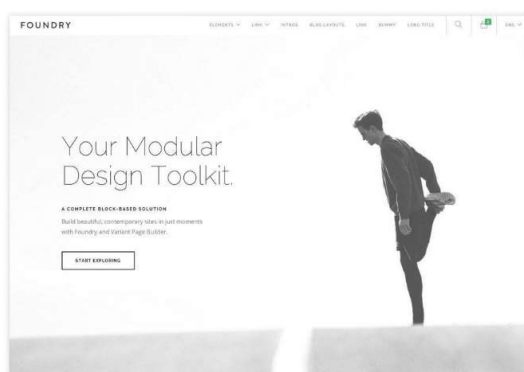


POPULAR FOUNDRY TOURS



Sleek, Flexible & Stylish.

Foundry is a remarkably complete template offering you a plethora of handcrafted design elements. Whether promoting a product, service or portfolio - Foundry's unique and flexible style has you covered.

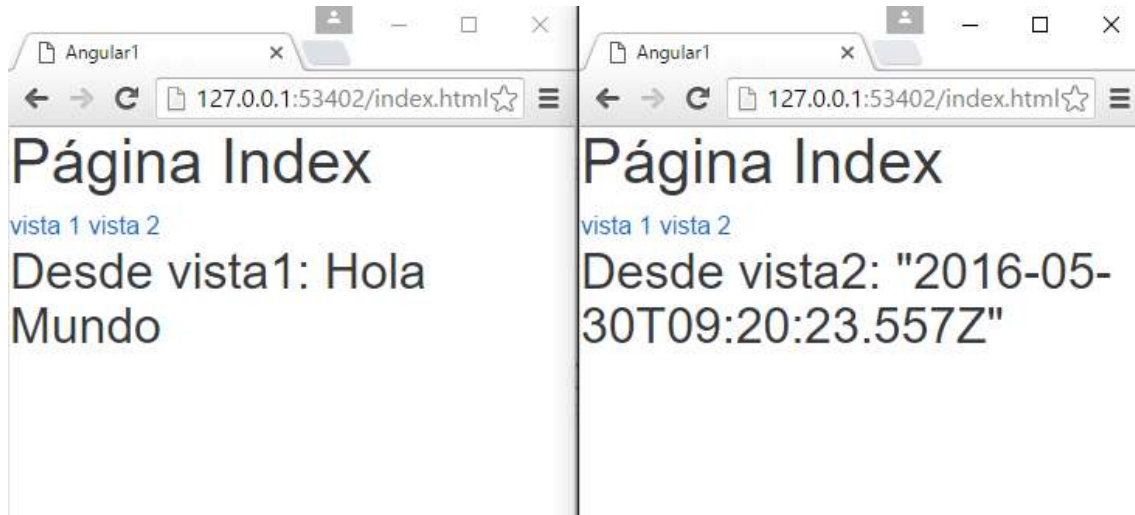


Build a slick, modern site faster than ever

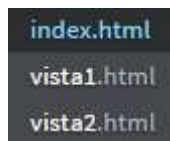
Foundry is your complete design toolkit, built from the ground up to be flexible, extensible and stylish. Building slick, contemporary sites has never been this easy!

EXPLORE FOUNDRY

Angular



Estructura de documentos:



Vista1

```
<h2>Desde vista1: {{ mensaje }}</h2>
```

Vista2

```
<h2>Desde vista2: {{ ahora }}</h2>
```

Index.html

```

</body>
</html>
<!DOCTYPE html>
<html lang="es" ng-app="miapp">
<head>
  <meta charset="UTF-8">
  <title>Angular1</title>
  <link rel="stylesheet" href="assets/css/bootstrap.min.css">
</head>
<body>

  <h1>Página Index</h1>
  <a href="#/">vista 1</a>
  <a href="#/vista2">vista 2</a>

  <ng-view></ng-view>

  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.5/angular.min.js"></scri
pt>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.5/angular-
route.js"></script>

  <script>

    'use strict'

    angular.module('miapp', ['ngRoute'])

    .config(function($routeProvider){
      $routeProvider
        .when('/',{
          controller:"Controlador1",
          templateUrl:'vista1.html'
        })
        .when('/vista2',{
          controller:"Controlador2",
          templateUrl:'vista2.html'
        })
    });

    .controller('Controlador1', function($scope){
      $scope.mensaje="Hola Mundo";
    })

    .controller('Controlador2', function($scope){
      $scope.ahora=new Date();
    });

  </script>
</body>
</html>

```

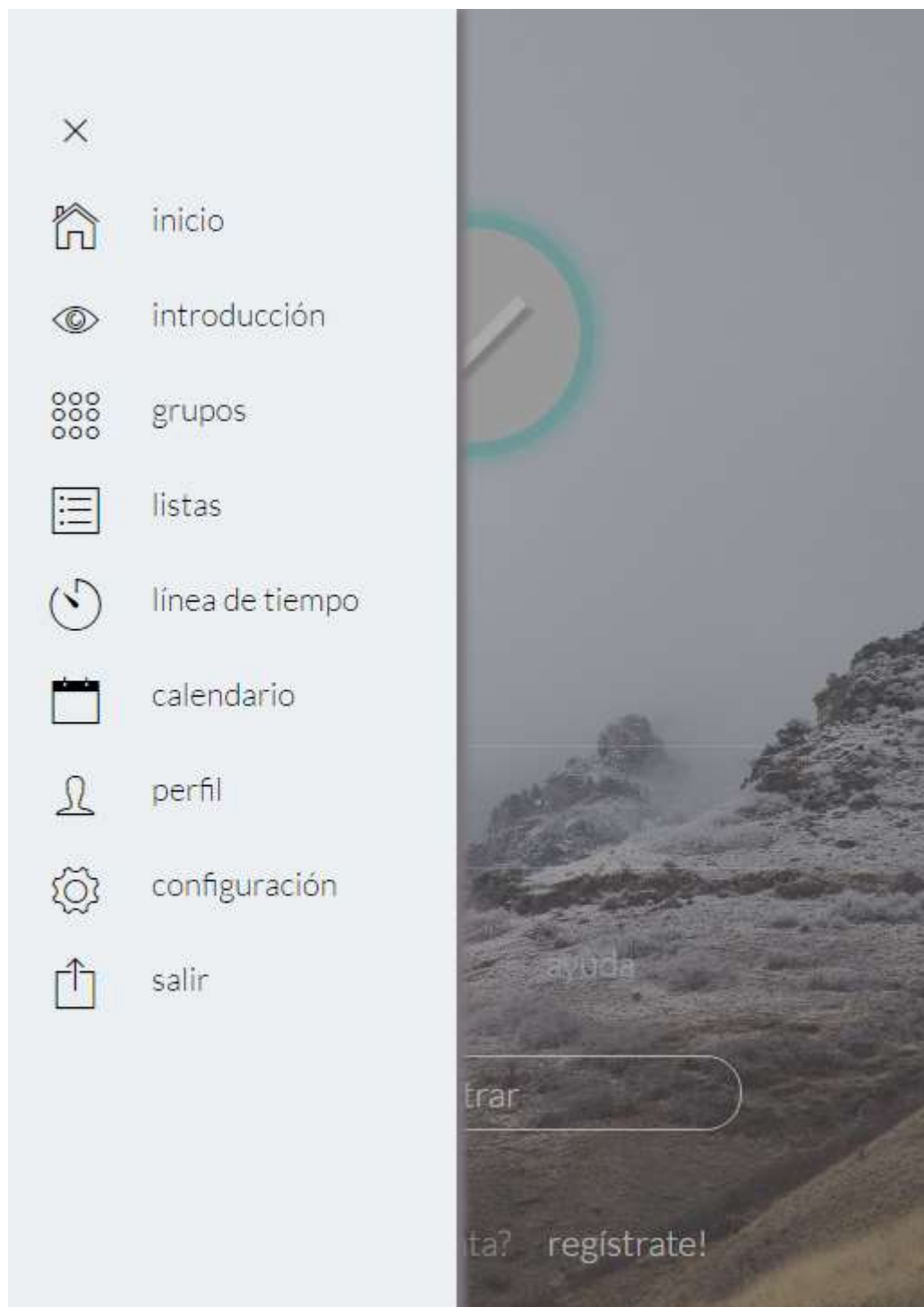
usuario

contraseña

ayuda

entrar

¿no tienes cuenta? regístrate!



Inicio.html

```
<div class="contenedor">

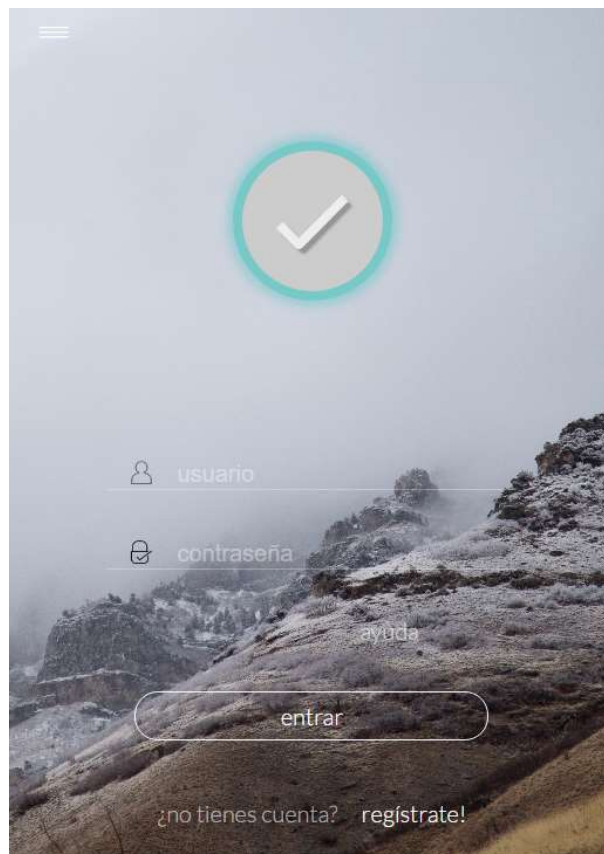
  <section class="mitad_superior">
    <div class="logo">
      <span class="ion-ios-checkmark-empty"></span>
    </div>
  </section>

  <section class="mitad_inferior">

    <input type="text" name='usuario' placeholder='usuario' />
    <input type="text" name='contraseña' placeholder='contraseña' />
    <p class="ayuda">ayuda</p>
    <a href="#/grupos" class="entrar">entrar</a>
    <p class="registrate">¿no tienes cuenta? <span>regístrate!</span></p>

  </section>

</div>
```



```
body {

background-image: url("../img/bg.jpg");
background-size: cover;
background-position: center;
background-attachment: fixed;

height: 100vh;
width: 100vw;

box-sizing: border-box;

overflow-x: hidden;
overflow-y: auto;

font-family: 'Lato';
font-weight: 300;
font-size: 18px;
}

.contenedor {
width: 100%;
height: 100vh;

display: flex;
flex-direction: column;
align-items: center;
}

.superior {
flex: 1;

display: flex;
align-items: center;
justify-content: center;
}

.inferior {
flex: 1;

width: 90%;

display: flex;
flex-direction: column;
align-items: center;
justify-content: space-around;
box-sizing: border-box;
}

header span {
display: inline-block;
width: 36px;
height: 36px;
font-size: 36px;
text-align: center;
line-height: 36px;
margin-top: 2px;
margin-left: 20px;

cursor: pointer;

color: #fff;
}
```

```
.logo {
  width: 120px;
  height: 120px;
  background: #ccc;
  border: 8px solid rgba(100,200,200, .8);
  border-radius: 50%;
  box-shadow: 0px 0px 10px 5px rgba(100,200,200,0.5);

  display: flex;
  align-items: center;
  justify-content: center;

  font-size: 150px;
  color: #eee;

  text-shadow: 4px 4px 2px rgba(150, 150, 150, 1);
}

input[type=text]{
  outline: none;
  background: transparent;
  color: #fff;
  font-size: 20px;
  border: none;
  border-bottom: 1px solid rgba(255,255,255,0.3);
  width: 350px;
  padding-left: 60px;
  box-sizing: border-box;
}

input::-webkit-input-placeholder{
  color: rgba(255, 255, 255, .4);
}

input[name='usuario']{
  background-image: url(http://icons.iconarchive.com/icons/custom-icon-
design/silky-line-user/256/user-icon.png);
  background-size: 20px;
  background-repeat: no-repeat;
  background-position: 20px 1px;
}

input[name=contraseña]{
  background-image:
url('http://icons.iconarchive.com/icons/iconsmind/outline/256/Security-Check-
icon.png');
  background-size: 20px;
  background-position: 20px 1px;
  background-repeat: no-repeat;
}
```

```
.ayuda{
  color: rgba(255, 255, 255, .6);
  font-size: 20px;
  font-weight: 300;
  text-align: right;
  padding-right: 50px;
  cursor: pointer;
  width: 50%;
  float: right;
}

.entrar {
  color: #fff;
  font-size: 20px;
  width: 300px;

  height: 40px;
  border-radius: 20px;

  border: 1px solid rgba(255, 255, 255, 1);

  display: flex;

  -ms-align-items: center;
  align-items: center;

  justify-content: center;

  cursor: pointer;

  text-decoration: none;
}

.registrate {
  color: rgba(255, 255, 255, .6);
  font-size: 20px;
  width: 500px;

  height: 40px;

  display: flex;

  -ms-align-items: center;
  align-items: center;

  justify-content: center;
}

.registrate > span {
  color: rgba(255, 255, 255, 1);
  margin-left: 20px;

  cursor: pointer;
}
```


Ahora el index.html

```
<!DOCTYPE html>
<html lang="es" ng-app="miapp">
<head>
  <meta charset="UTF-8">
  <title>Angular1</title>
  <link rel="stylesheet" href="estilos.css">
  <link rel="stylesheet"
href="http://code.ionicframework.com/ionicons/2.0.1/css/ionicons.min.css">
</head>
<body>

  <h1>Página Index</h1>
  <a href="#/">vista 1</a>
  <a href="#/vista2">vista 2</a>

  <ng-view></ng-view>

  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.5/angular.min.js"></scri
pt>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.5/angular-
route.js"></script>

  <script>

    'use strict'

    angular.module('miapp', ['ngRoute'])

    .config(function($routeProvider){
      $routeProvider
        .when('/',{
          controller:"Controlador1",
          templateUrl:'inicio.html'
        })
        .when('/vista2',{
          controller:"Controlador2",
          templateUrl:'vista2.html'
        });
    })

    .controller('Controlador1', function($scope){
      $scope.mensaje="Hola Mundo";
    })

    .controller('Controlador2', function($scope){
      $scope.ahora=new Date();
    });

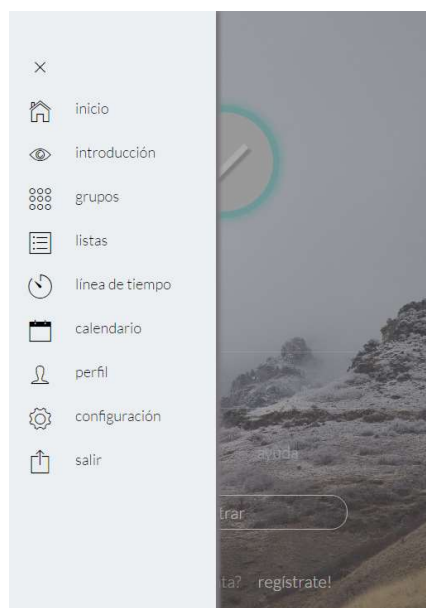
  </script>
</body>
</html>
```

Menú de navegación

```
<div class="tapar"></div>
```

```
<nav>
  <ul>
    <li><span class="ion-ios-close-empty"></span></li>
    <li><a href="#/"><span class="ion-ios-home-
outline"></span>inicio</a></li>
    <li><a href="#/introduccion"><span class="ion-ios-eye-
outline"></span>introducción</a></li>
    <li><a href="#/grupos"><span class="ion-ios-keypad-
outline"></span>grupos</a></li>
    <li><a href="#/listas"><span class="ion-ios-list-
outline"></span>listas</a></li>
    <li><a href="#/lineadetiempo"><span class="ion-ios-timer-
outline"></span>línea de tiempo</a></li>
    <li><a href="#/calendario"><span class="ion-ios-calendar-
outline"></span>calendario</a></li>
    <li><a href="#/perfil"><span class="ion-ios-person-
outline"></span>perfil</a></li>
    <li><a href="#/configuracion"><span class="ion-ios-gear-
outline"></span>configuración</a></li>
    <li><a href="#/salir"><span class="ion-ios-upload-
outline"></span>salir</a></li>
  </ul>
</nav>

<header>
  <span class="ion-ios-drag"></span>
</header>
```



```
header {
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

```

        width: 100%;
        height: 40px;
        background: ;

        position: fixed;
        top: 0;
        left: 0;
    }
    header span {
        display: inline-block;
        width: 36px;
        height: 36px;
        font-size: 36px;
        text-align: center;
        line-height: 36px;
        margin-top: 2px;
        margin-left: 20px;

        cursor: pointer;

        color: #fff;
    }
    nav {
        position: fixed;
        top: 0;
        left: -250px;
        width: 250px;
        height: 100%;

        -webkit-box-sizing: border-box;
        -moz-box-sizing: border-box;
        box-sizing: border-box;

        overflow: hidden;

        transition: all .5s;

        background: #ECEFF1;

        padding-left: 20px;
        padding-top: 40px;

        z-index: 1000;
    }
    nav.mostrar {
        left: 0;
        box-shadow: 5px 0px 5px 0px rgba(122,118,122,1);
    }
    nav span {
        font-size: 35px;
        margin-right: 25px;
        display: inline-block;
        width: 35px;

        text-align: center;
        vertical-align: middle;
    }

```

```
nav li {
  cursor: pointer;
  font-size: 18px;
  margin-bottom: 5px;
  height: 48px;
  line-height: 48px;
}

nav a {
  text-decoration: none;
  color: #000;
}

.tapar {
  width: 100vw;
  height: 100vh;
  background: rgba(100, 100, 100, .5);
  position: fixed;
  top: 0;
  left: -100vw;
  transition: all .5s;
}

.mover-tapar {
  left: 0;
}
```

jQuery

```
$(document).ready(function () {
  $('header > span').on('click', function () {
    $('nav').addClass('mostrar');
    $('.tapar').addClass('mover-tapar');
  });

  $('nav li').on('click', function () {
    $('nav').removeClass('mostrar');
    $('.tapar').removeClass('mover-tapar');
  });

  $('.tapar').on('click', function () {
    $('nav').removeClass('mostrar');
    $(this).removeClass('mover-tapar');
  });
});
```

AngularJS

```
angular.module('miapp', ['ngRoute', 'firebase'])

.config(function ($routeProvider) {
  $routeProvider

    .when('/', {
      controller: 'Controlador1',
      templateUrl: 'inicio.html'
    })

    .when('/introduccion', {
      controller: 'Controlador1',
      templateUrl: 'introduccion.html'
    })

    .when('/grupos', {
      controller: 'Controlador1',
      templateUrl: 'grupos.html'
    })

    .when('/listas', {
      controller: 'ControladorTareas',
      templateUrl: 'listas.html'
    })

    .when('/lineadetiempo', {
      controller: 'Controlador1',
      templateUrl: 'lineadetiempo.html'
    })

    .when('/calendario', {
      controller: 'Controlador1',
      templateUrl: 'calendario.html'
    })

    .when('/perfil', {
      controller: 'Controlador1',
      templateUrl: 'perfil.html'
    })

    .when('/configuracion', {
      controller: 'Controlador1',
      templateUrl: 'configuracion.html'
    })

    .when('/salir', {
      controller: 'Controlador1',
      templateUrl: 'inicio.html'
    })

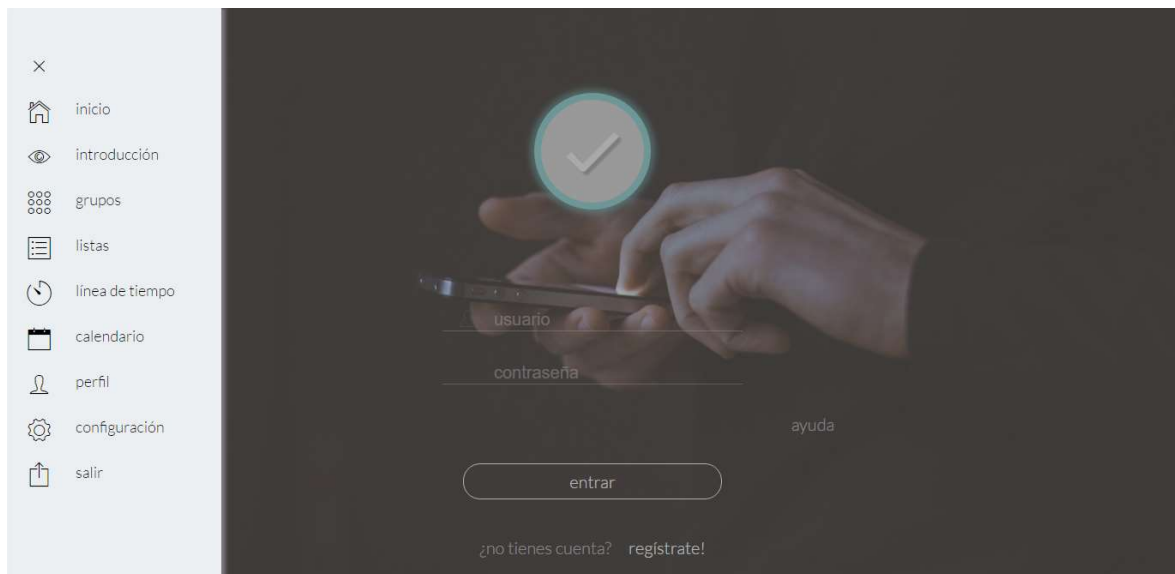
    .when('/view1', {
      controller: 'Controlador1',
      templateUrl: 'view1.html'
    })

    .when('/view2', {
      controller: 'Controlador2',
      templateUrl: 'view2.html'
    })
  });
})
```

```
.controller('Controlador1', function ($scope) {
    $scope.message = "Hello, world";
})

.controller('Controlador2', function ($scope) {
    $scope.now = new Date();
});
```

```
calendario.html
configuracion.html
grupos.html
index.html
inicio.html
introduccion.html
lineadetiempo.html
listas.html
menu.html
perfil.html
registrarse.html
salir.html
view1.html
view2.html
```



En Listas.html

Two-way Data Binding - AngularJS

```
<style>
  main {
    margin-top: 50px;
  }
</style>

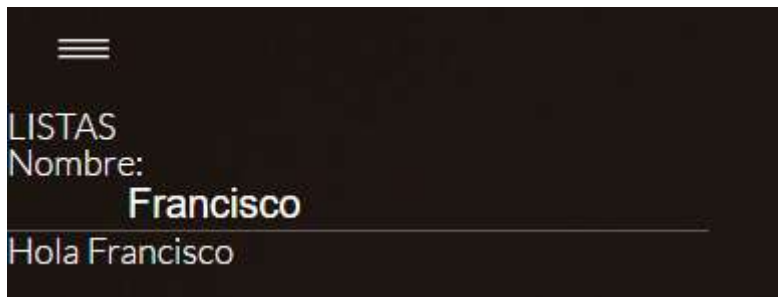
<main>

  <h1>LISTAS</h1>

  Nombre:<input type="text" ng-model="nombre">

  <h2>Hola {{ nombre }} </h2>

</main>
```



Expresiones: {{ nombre }}

Directivas: ng-model="nombre"

```
<style>
  main {
    margin-top: 50px;
  }
</style>
```

```
<main>
```

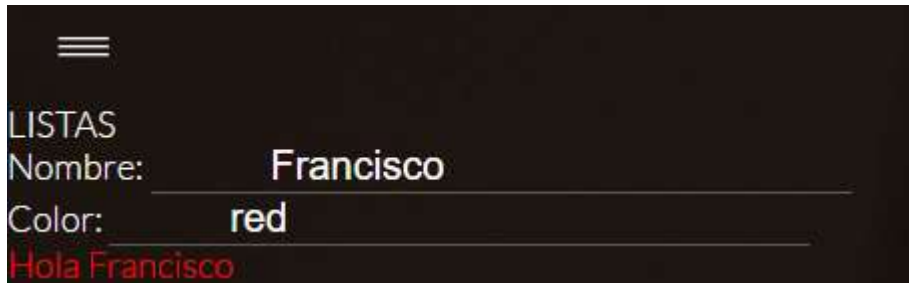
```
<h1>LISTAS</h1>
```

```
Nombre: <input type="text" ng-model="nombre"><br>
```

```
Color: <input type="text" ng-model="color">
```

```
<h2 style="color:{{ color }}">Hola {{ nombre }} </h2>
```

```
</main>
```



Listas de Tareas

La funcionalidad se la proporciona el controlador: **ControladorTareas**, que posee un **\$scope** y cualquier variable que declaremos en el **\$scope** del controlador será accesible para cualquier vista que utilice este controlador (en este caso la templateUrl listas.html).

En **index.html**

```
.when('/listas',{
    controller:"ControladorListas",
    templateUrl:'listas.html'
})

...

...

.controller('ControladorListas', function($scope){
});
```

Se va a declarar un array **tareas[]**, que van a tener una propiedad **texto**, que será la descripción de la tarea y una propiedad **hecho** que va a indicar se está realizada o pendiente.

```
$scope.tareas = [{ texto: "", hecho: true }];
```

Se añaden dos tareas:

```
.controller('ControladorListas', function($scope){
    $scope.tareas = [{texto: 'Ser Super Heroico con AngularJS', hecho: true},
        {texto: 'Crear una app con angular', hecho: false}];
});
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

En `listas.html`

```
<ul>
  <li>{{ tareas[0].texto }}</li>
</ul>
```



Directiva ngRepeat (ng-repeat)

Se va a añadir un input checkbox que va a estar seleccionado si la tarea está hecha y no si está pendiente. Y vamos a ligar el modelo de esta tarea a la propiedad `hecho` del array de tareas.

El `ng-repeat` va a iterar el array de tareas por cada tarea.

```
<ul>
  <li ng-repeat="tarea in tareas">
    <input type="checkbox" ng-model="tarea.hecho">
    <span>{{tarea.texto}}</span>
  </li>
</ul>
```



Aplicar estilos

```
<style>
  main {
    margin-top: 50px;
  }

  .hecho-true {
    text-decoration: line-through;
    color: grey;
  }
</style>

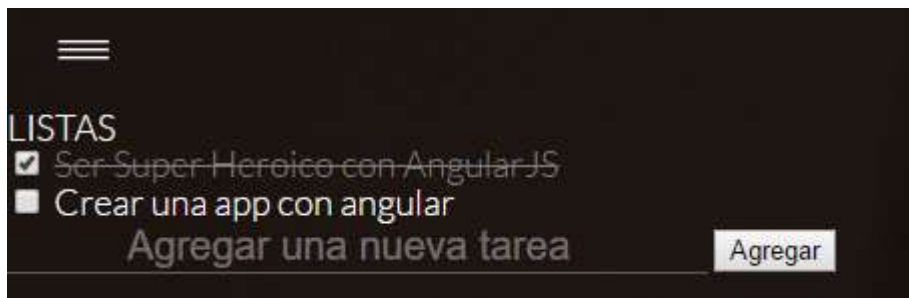
<ul>
  <li ng-repeat="tarea in tareas">
    <input type="checkbox" ng-model="tarea.hecho">
    <span class="hecho-{{tarea.hecho}}">{{tarea.texto}}</span>
  </li>
</ul>
```



Agregar tareas

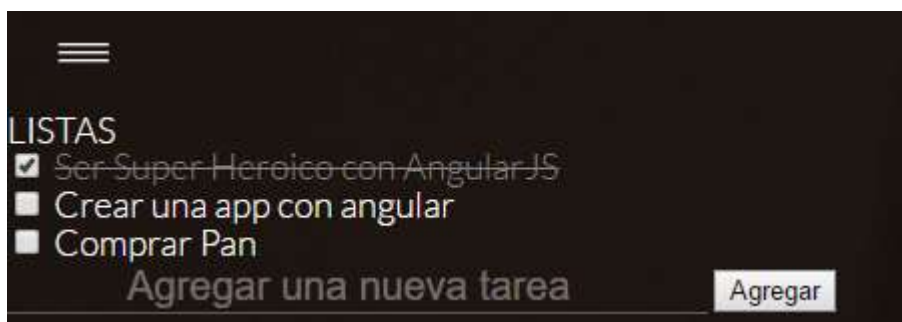
Agregamos un formulario para agregar tareas:

```
<form>
  <input type="text" size="30" placeholder="Agregar una nueva tarea" ng-model=
"textoNuevaTarea">
  <button type="button" ng-click="agregarTarea()">Agregar</button>
</form>
```

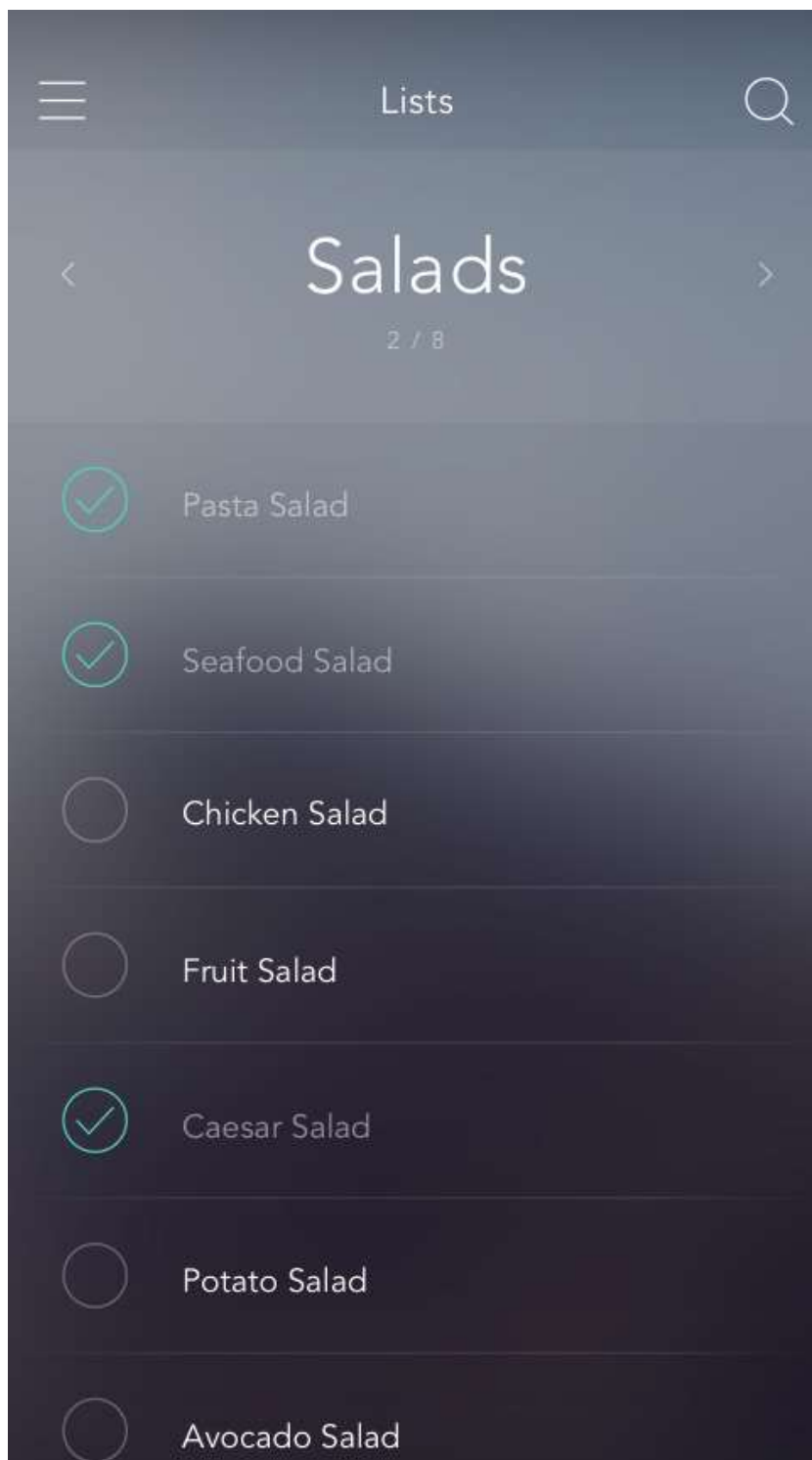



```
.controller('ControladorListas', function($scope){
  $scope.tareas = [{texto: 'Ser Super Heroico con AngularJS', hecho: true},
    {texto: 'Crear una app con angular', hecho: false}];

  $scope.agregarTarea = function(){
    $scope.tareas.push({texto: $scope.textoNuevaTarea, hecho: false})
    $scope.textoNuevaTarea = "";
  }
});
```





<https://www.invisionapp.com/do>








Sign up



NAME
Enter name

EMAIL
Enter email

PASSWORD
Enter password

PHONE
Enter phone number

Continue

ALREADY HAVE AN ACCOUNT SIGN IN

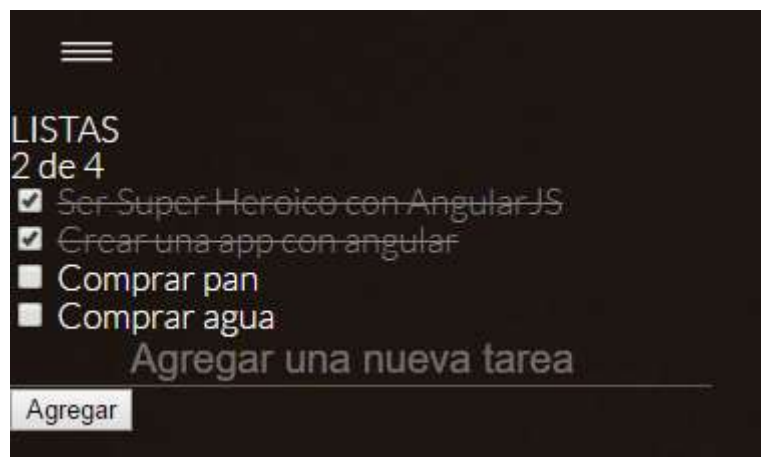
Restante

{{ restantes() }} de {{ tareas.length }}

```
.controller('ControladorListas', function($scope){
    $scope.tareas = [{texto: 'Ser Super Heroico con AngularJS', hecho: true},
                    {texto: 'Crear una app con angular', hecho: false}];

    $scope.agregarTarea = function(){
        $scope.tareas.push({texto: $scope.textoNuevaTarea, hecho: false})
        $scope.textoNuevaTarea = "";
    };

    $scope.restantes = function() {
        var cuenta = 0;
        angular.forEach($scope.tareas, function(tarea) {
            cuenta += tarea.hecho ? 0 : 1;
        });
        return cuenta;
    };
});
```



Eliminar

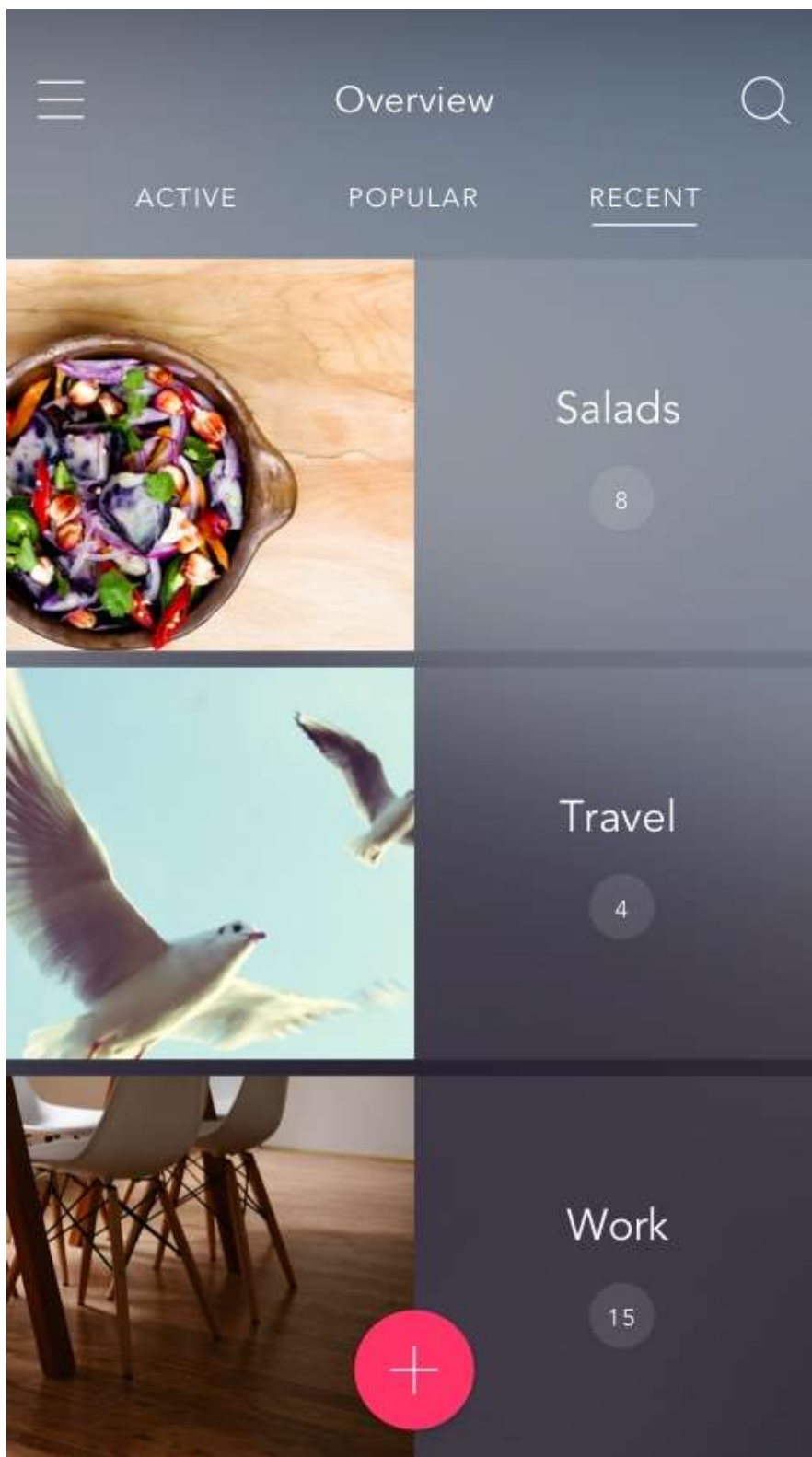
[

```
.controller('ControladorListas', function($scope){
    $scope.tareas = [{texto: 'Ser Super Heroico con AngularJS', hecho: true},
                     {texto: 'Crear una app con angular', hecho: false}];

    $scope.agregarTarea = function(){
        $scope.tareas.push({texto: $scope.textoNuevaTarea, hecho: false})
        $scope.textoNuevaTarea = "";
    };

    $scope.restantes = function() {
        var cuenta = 0;
        angular.forEach($scope.tareas, function(tarea) {
            cuenta += tarea.hecho ? 0 : 1;
        });
        return cuenta;
    };

    $scope.eliminar = function() {
        var tareasViejas = $scope.tareas;
        $scope.tareas = [];
        angular.forEach(tareasViejas, function(tarea) {
            if (!tarea.hecho) $scope.tareas.push(tarea);
        });
    };
});
```

CON FIREBASE

```
.controller('ControladorTareas',function($scope, $firebaseArray){

    var refTareas = new
    Firebase('https://applistadetarea.firebaseio.com/tareas');
    $scope.tareas = $firebaseArray(refTareas);

    $scope.agregarTarea = function () {
        $scope.tareas.$add({ texto: $scope.textoNuevaTarea, hecho: false });
        $scope.textoNuevaTarea = '';
    };

    $scope.restantes = function () {
        return $scope.tareas.length;
    };

    $scope.tareasRealizadas = function () {
        var cuenta = 0;
        angular.forEach($scope.tareas, function (tarea) {
            if (tarea.hecho)
                cuenta++;
        });
        return cuenta;
    };

    $scope.eliminar = function () {
        angular.forEach($scope.tareas, function (tarea) {
            if (tarea.hecho)
                $scope.tareas.$remove(tarea);
        });
    };

});
```

```
<script src="https://cdn.firebase.com/js/client/2.2.4/firebase.js">
</script>

<script src="https://cdn.firebase.com/libs/angularfire/1.2.0/angularfire.min.js">
</script>
```

Instrucciones

Una **instrucción** es una unidad básica de programación que representa un paso simple en un programa.

Por ejemplo:

```
alert("Hola Mundo");
```

Esta instrucción simple abre una ventana de alerta con el texto Hola Mundo

Cada instrucción acaba con ;

Funciones

Las **funciones** son conjuntos de instrucciones

Por ejemplo:

```
function holamundo(){
    alert("HOLA MUNDO");
}
```

Tipos de datos

En Javascript los tres tipos más comunes de datos son **números**, **cadenas** y **Booleanos**.

Números

Se utilizan para almacenar valores numéricos **enteros** (llamados integer en inglés) o **decimales** (llamados float en inglés). Los números decimales utilizan el carácter . (punto) en vez de , (coma) para separar la parte entera y la parte decimal:

```
var iva = 21;           // variable tipo entero
var total = 234.65;    // variable tipo decimal
```

Cadenas de texto

Para asignar el valor a la variable, se encierra el valor entre comillas dobles o simples, para delimitar su comienzo y su final:

```
var mensaje = "Bienvenido a nuestro sitio web";  
var nombreProducto = 'Producto ABC';  
var letraSeleccionada = 'c';
```

Booleanos

Una variable de tipo boolean almacena un tipo especial de valor que solamente puede tomar dos valores: true (verdadero) o false (falso).

```
var clienteRegistrado = false;  
var ivaIncluido = true;
```

1) Haz un programa de nombre **Eval1A.htm**, que sirva para restar dos números cualesquiera.

2) Haz un programa de nombre **Eval1B.htm** que sirva para dividir dos números.

3) Haz un programa de nombre **Eval1C.htm** que funcione de la siguiente forma:

- El programa nos pregunta nuestro nombre.
- El programa nos pregunta nuestra edad.
- El programa da como resultado nuestro nombre y a continuación los días que hemos vivido hasta el momento (deberás multiplicar la edad por 365).

4) Haz un programa de nombre **Eval1D.htm** que funcione de la siguiente forma:

- El programa nos pide un número.
- El programa nos muestra en una única pantalla (un único "alert"), el doble, el triple y cuádruple del número que habíamos introducido.

5) El siguiente programa tiene errores. Escríbelo (grábalo con el nombre **Eval1E.htm** en *TuCarpeta*) y corrígelo para que funcione y explica para qué sirve:

```
/* EVAL1E.HTM
var a,b;
a=prompt("Escribe la base:")
b=prompt("Escribe la altura:")
alert("Área= "+(a*b/2);
```

6) Haz un programa de nombre **Eval1F.htm** que sirva para calcular la longitud de una circunferencia y el área del círculo correspondiente.

7) Haz un programa de nombre **Eval1G.htm**, que has de grabar en *TuCarpeta*, que funcione de la siguiente forma:

- El programa nos pide nuestro nombre.
- El programa nos pide nuestro primer apellido.
- El programa nos pide en qué población vivimos.
- El programa presenta una pantalla aproximadamente igual a la siguiente:

```
=====
Hola nombre Apellido
Adiós habitante de Población
=====
```

Bucles en Javascript

For

```

////////////////////////////////////
//          BUCLES          //
////////////////////////////////////
//          Sumatorio        //
////////////////////////////////////

function Sumatorio(){
    var numero = parseInt(prompt("Introduce un número y se mostrará su
factorial"));
    var resultado = 0;

    for(var i=1; i<=numero; i++) {
        resultado += i;
    }
    alert(resultado);
    document.getElementById("resultado").innerHTML = resultado;
}

////////////////////////////////////
//          BUCLES          //
////////////////////////////////////
//          Factorial        //
////////////////////////////////////

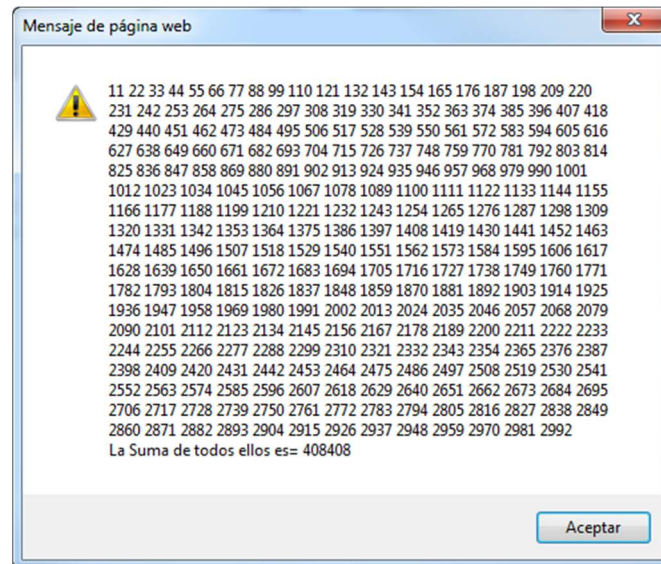
function Factorial(){
    var numero = parseInt(prompt("Introduce un número y se mostrará su
factorial"));
    var resultado = 1;

    for(var i=1; i<=numero; i++) {
        resultado *= i;
    }
    alert(resultado);
    document.getElementById("resultado").innerHTML = resultado;
}

```

}

Programa que calcula todos los **múltiplos de 11 menores de 3000** y por último nos da la **suma de todos ellos**.



```

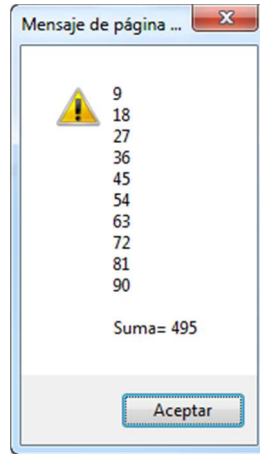
////////////////////////
//      Bucles for      //
////////////////////////
//      multiplos de 11  //
////////////////////////

function multiplos11(){
    var salida="";
    var sum=0;
    for(var multi=11;multi<3000;multi=multi+11)
    {
        salida=salida+multi+" ";
        sum=sum+multi;
    }
    alert(salida+"\nLa Suma de todos ellos es= "+sum);
}

```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

Programa que calcula los **10 primeros múltiplos del número** que queramos, por último nos da la suma de todos ellos.



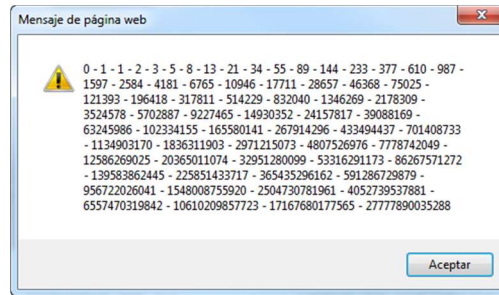
```

////////////////////////////////////
//      Bucles for              //
////////////////////////////////////
//      tablamultiplicar        //
////////////////////////////////////
function tablamultiplicar(){
    var salida="";
    var num;
    var mult;
    var sum=0;
    num=prompt("¿Tabla de multiplicar de qué número?", "");
    num=parseInt(num,10);
    for(i=1;i<=10;i++)
    {
        mult=num*i;
        salida=salida+mult+"\n";
        sum=sum+mult;
    }
    alert(salida+"\nSuma= "+sum);
}

```


CÁLCULO EN FORMA ITERATIVA

- La llamada **sucesión de FIBONACCI** es: 0, 1, 1, 2, 3, 5, 8, 13, ...
Es decir, cada término es igual a la suma de los dos anteriores.
Vamos a "programar" la sucesión de Fibonacci.



```

//////////////////////////
//          Iteraciones          //
//          Fibonacci            //
//////////////////////////

function fibonacci(){
    var anterior,ultimo,aux;
    anterior=0;
    ultimo=1;
    var solucion;
    solucion="0 - 1";
    while (ultimo<=25000000000000)
    {
        aux=anterior+ultimo;
        anterior=ultimo;
        ultimo=aux;
        if (ultimo>0) solucion=solucion+" - "+ultimo;
    }
    alert(solucion);
}

```

```

////////////////////////////////////
//                               //
////////////////////////////////////

function dni{

    var letras = ['T', 'R', 'W', 'A', 'G', 'M', 'Y', 'F', 'P', 'D', 'X',
                  'B', 'N', 'J', 'Z', 'S', 'Q', 'V', 'H', 'L', 'C', 'K', 'E', 'T'];

    var numero = prompt("Introduce tu número de DNI (sin la letra)");
    var letra = prompt("Introduce la letra de tu DNI (en mayúsculas)");
    letra = letra.toUpperCase();

    if(numero < 0 || numero > 99999999) {
        alert("El número proporcionado no es válido");
    }
    else {
        var letraCalculada = letras[numero % 23];
        if(letraCalculada != letra) {
            alert("La letra o el número proporcionados no son correctos");
        }
        else {
            alert("El número de DNI y su letra son correctos");
        }
    }
}

```

Arrays

Los **arrays** son listas de valores con índice-cero (en inglés *zero-index*), es decir, que el primer elemento del array está en el índice 0.

Un array simple

```
function matrices(){
    var marcaCoches = [ 'Audi', 'BMW', 'Mercedes', 'Renault', 'SEAT'];
}
```

Acceder a los ítems del array a través de su índice

```
function matrices(){
    var marcaCoches = [ 'Audi', 'BMW', 'Mercedes', 'Renault', 'SEAT'];
    alert("El primer elemento de marcaCoches es: " + marcaCoches[0]);
}
```

Obtener la cantidad de ítems del array

```
function matrices(){
    var marcaCoches = [ 'Audi', 'BMW', 'Mercedes', 'Renault', 'SEAT'];
    alert("El primer elemento de marcaCoches es: " + marcaCoches[0] +
        "\n" + "la longitud del array marcaCoches es: " + marcaCoches.length);
}
```

Añadir elementos a un array

```
function matrices(){
    var marcaCoches = [ 'Audi', 'BMW', 'Mercedes', 'Renault', 'SEAT'];
    alert("El primer elemento de marcaCoches es: " + marcaCoches[0] +
        "\n" + "la longitud del array marcaCoches es: " + marcaCoches.length);
    marcaCoches.push('Honda');
    var i;
    var salida = "";
    for(i in marcaCoches){
        salida += marcaCoches[i] + "\n";
    }
    alert(salida);
}
```

JavaScript Timing Eventos

Con JavaScript, es posible ejecutar un código en intervalos de tiempo especificados. Esto se llama eventos de tiempo.

Es muy fácil para los acontecimientos del tiempo en JavaScript. Los dos métodos principales que se utilizan son:

- **setInterval ()** - ejecuta una función, una y otra vez, a intervalos de tiempo especificados
- **setTimeout ()** - ejecuta una función, una vez, después de esperar un número especificado de milisegundos

El setInterval (Método)

El método setInterval () esperará un número especificado de milisegundos, y luego ejecutar una función específica, y continuará para ejecutar la función, una vez en cada intervalo de tiempo dado.

Sintaxis

```
setInterval("javascript function", milliseconds);
```

El primer parámetro de setInterval () debe ser una función.

El segundo parámetro indica la longitud de los intervalos de tiempo entre cada ejecución.

Nota: Hay 1.000 milisegundos en un segundo.

Ejemplo

Alerta "hola" cada 3 segundos:

```
setInterval(function () {alert("Hello")}, 3000);
```

A continuación se muestra un ejemplo que muestre la hora actual. El método setInterval () se utiliza para ejecutar la función una vez cada 1 segundo, como un reloj digital.

```
////////////////////////////////////
//              setInterval          //
////////////////////////////////////
var myVar = setInterval(function () {myTimer()}, 1000);
function myTimer() {
    var d = new Date();
    document.getElementById("demo").innerHTML =
d.toLocaleTimeString();
}
```

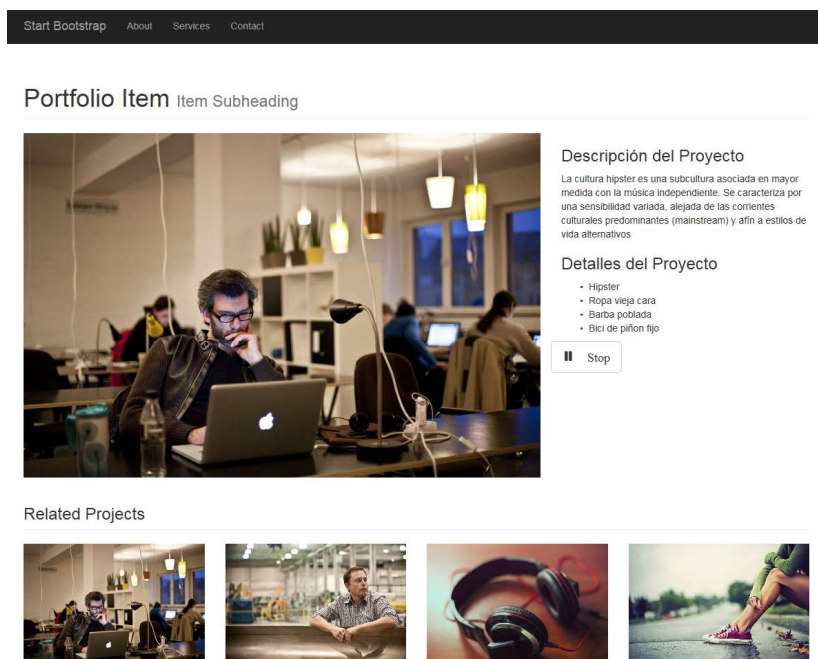
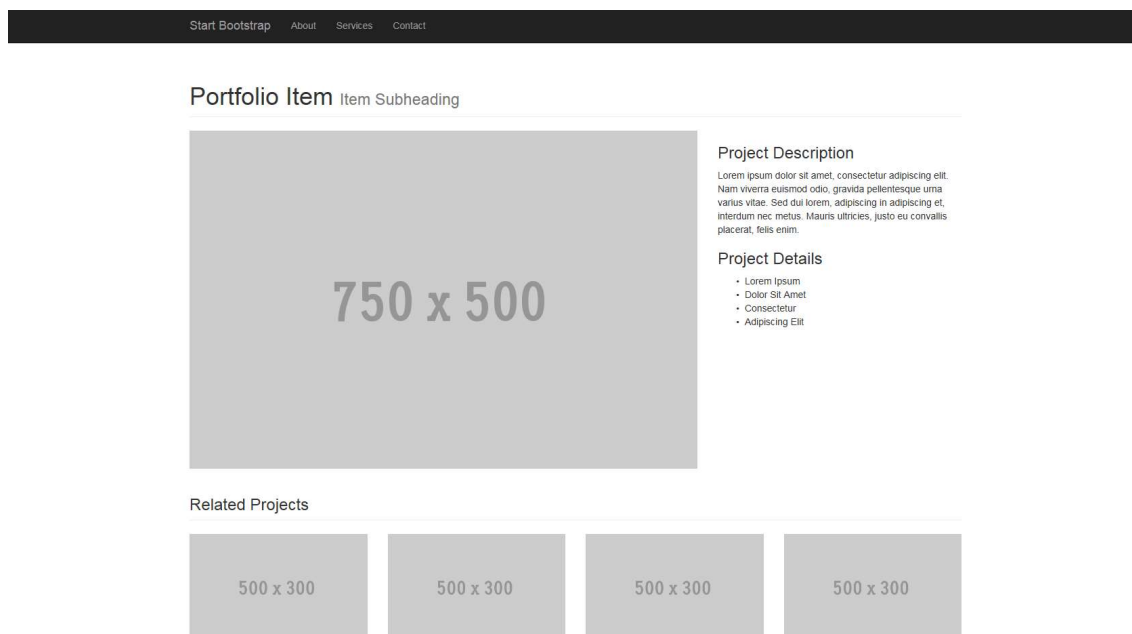
A continuación se muestra un ejemplo que muestre una serie de imágenes:

```
////////////////////////////////////
//              Imagenes            //
////////////////////////////////////
var imagenes = ['img800x500_1.jpg', 'img800x500_2.jpg', 'img800x500_3.jpg',
'img800x500_4.jpg', 'img800x500_5.jpg'];
var myVar2 = setInterval(function () {myTimer2()}, 1000);
var i = 0;
function myTimer2() {
    document.getElementById("demo").innerHTML = "<img src='img/" +
imagenes[i] + "'></img>";
    i++;
    if(i==(imagenes.length-1)){i=0;};
}
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

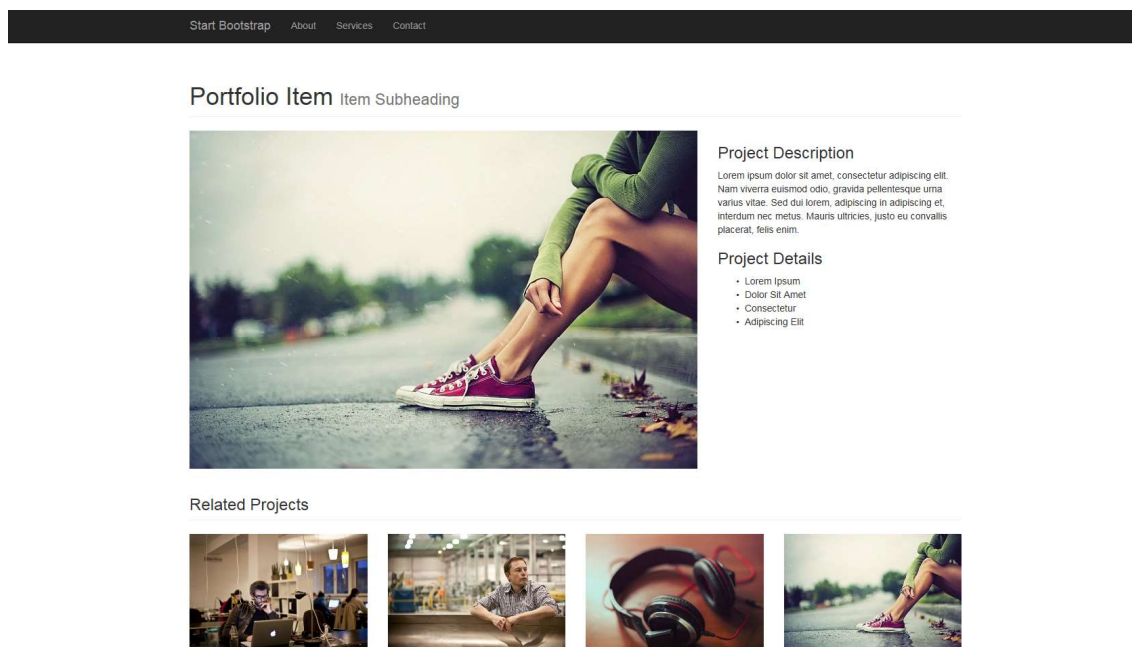
Portfolio Item.

Partiendo de la **plantilla de Start Bootstrap: Portfolio Item**



Desarrollo y reutilización de componentes de software mediante lenguajes de guión

Añadir las imágenes.



Añadir la programación:

```
var imagenes = ['img1_750x500.jpg', 'img2_750x500.jpg', 'img3_750x500.jpg', 'img4_750x500.jpg'];
```

```
var myVar2 = setInterval(function () {myTimer2()}, 4000);
var i = 0;
function myTimer2() {
document.getElementById("portfolio").innerHTML =
"<img class='img-responsive' src='img/'+ imagenes[i]+ '></img>";
i++;
if(i==(imagenes.length)){i=0;};
}
```

Y en el html

```
<div id="portfolio" class="col-md-8">
  
</div>
```

Añadir Descripción y Detalles

Desarrolllo y reutilización de componentes de software mediante lenguajes de guión

Portfolio Item Item Subheading



Descripción del Proyecto

La cultura hipster es una subcultura asociada en mayor medida con la música independiente. Se caracteriza por una sensibilidad variada, alejada de las corrientes culturales predominantes (mainstream) y afín a estilos de vida alternativos

Detalles del Proyecto

- Hipster
- Ropa vieja cara
- Barba poblada
- Bici de piñon fijo

```
var imagenes = ['img1_750x500.jpg', 'img2_750x500.jpg', 'img3_750x500.jpg', 'img4_750x500.jpg'];
```

```
var descripcion = ['La cultura hipster es una subcultura asociada en mayor medida con la música independiente. Se caracteriza por una sensibilidad variada, alejada de las corrientes culturales predominantes (mainstream) y afín a estilos de vida alternativos',
```

```
'Elon Musk (28 de junio de 1971) es un físico y emprendedor sudafricano, más conocido por ser el co-fundador de PayPal, SpaceX y Tesla Motors.',
```

```
'La música (del griego: μουσική [τέχνη] - mousikē [téchnē], "el arte de las musas") es, según la definición tradicional del término, el arte de organizar sensible y lógicamente una combinación coherente de sonidos y silencios utilizando los principios fundamentales de la melodía, la armonía y el ritmo, mediante la intervención de complejos procesos psico-anímicos.',
```

```
'Trail running es un deporte que consiste en correr "fuera de pista", por senderos de montaña, huellas, rastros o caminos secundarios, a través de montañas, cerros y montes, cruzando arroyos y ríos, con grandes trepadas y abruptas bajadas.'];
```

```
var detalles = ['<li>Hipster</li><li>Ropa vieja cara</li><li>Barba poblada</li><li>Bici de piñon fijo</li>', '<li>Elon Musk</li><li>PayPal</li><li>SpaceX</li><li>Tesla Motors</li>', '<li>Música</li><li>Musas</li>', '<li>Trail running</li><li>Montaña</li><li>Running</li><li>Resistencia</li>']
```

```
var myVar2 = setInterval(function () { myTimer2() }, 4000);
var i = 0;
function myTimer2() {
    document.getElementById("portfolio").innerHTML = "<img class='img-responsive' src='img/" + imagenes[i] + "'></img>";
```

```
document.getElementById("descripcion").innerHTML = descripcion[i];
```

```
document.getElementById("detalles").innerHTML = detalles[i];
```

```
i++;
if (i == (imagenes.length)) { i = 0; }
}
```

Html

```
<div class="col-md-4">
    <h3>Descripción del Proyecto</h3>
    <p id="descripcion">La cultura hipster ...p>
    <h3>Detalles del Proyecto</h3>
    <ul id="detalles">
        <li>Hipster</li>
        <li>Ropa vieja cara</li>
        <li>Barba poblada</li>
        <li>Bici de piñon fijo</li>
    </ul>
</div>
```

Botón Stop

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

Portfolio Item Item Subheading



Descripción del Proyecto

La música (del griego: μουσική [téchnē] - mousikḗ [téchnē], "el arte de las musas") es, según la definición tradicional del término, el arte de organizar sensible y lógicamente una combinación coherente de sonidos y silencios utilizando los principios fundamentales de la melodía, la armonía y el ritmo, mediante la intervención de complejos procesos psico-ánimos.

Detalles del Proyecto

- Música
- Musas

⏸ Stop

```
function myStopFunction() {
    clearInterval(myVar2);
}
```

```
<button      type="button"      class="btn      btn-default      btn-lg"
onclick="myStopFunction()">
<span class="glyphicon glyphicon-pause" aria-hidden="true"> Stop</span>
</button>
```

Versión 'Modo Ninja' del Botón Stop

```
function myStopFunction() {
    clearInterval(myVar2);

    document.getElementById("stopplay").innerHTML = "<button
id='stopplay'      type='button'      class='btn      btn-default      btn-lg'
onclick='myPlayFunction()'><span      class='glyphicon      glyphicon-play'      aria-
hidden='true'> Play</span></button>";
}

function myPlayFunction() {
    myVar2 = setInterval(function () { myTimer2() }, 4000);

    document.getElementById("stopplay").innerHTML = "<button
id='stopplay'      type='button'      class='btn      btn-default      btn-lg'
onclick='myStopFunction()'><span      class='glyphicon      glyphicon-stop'      aria-
hidden='true'> Stop</span></button>";
}
```

Imágenes Inferiores

Paso 1:

En el Script:

```
function cambiarA(j){
    document.getElementById("portfolio").innerHTML = "<img class='img-responsive' src='img/" + imagenes[j] + "'></img>";

    document.getElementById("descripcion").innerHTML = descripcion[j];

    document.getElementById("detalles").innerHTML = detalles[j];
}
```

En el html:

```
<div onclick="cambiarA(0)" class="col-sm-3 col-xs-6">
    <a href="#">
        
    </a>
</div>
```

FUNCIONA! → Paso 2

Paso 2

En el Script:

```
function cambiarA(j) {
    document.getElementById("portfolio").innerHTML = "<img class='img-responsive' src='img/"
    + imagenes[j] + "'></img>";
    document.getElementById("descripcion").innerHTML = descripcion[j];
    document.getElementById("detalles").innerHTML = detalles[j];
    i = j+1;
    myStopFunction();
}
```

En el html:

```
<div class="col-lg-12">
    <h3 class="page-header">Related Projects</h3>
</div>
<div onclick="cambiarA(0)" class="col-sm-3 col-xs-6">
    <a href="#">
        
    </a>
</div>
<div onclick="cambiarA(1)" class="col-sm-3 col-xs-6">
    <a href="#">
        
    </a>
</div>
<div onclick="cambiarA(2)" class="col-sm-3 col-xs-6">
    <a href="#">
        
    </a>
</div>
<div onclick="cambiarA(3)" class="col-sm-3 col-xs-6">
    <a href="#">
        
    </a>
</div>
```

Barra de Progreso

[Start Bootstrap](#) [About](#) [Services](#) [Contact](#)

Portfolio Item Item Subheading



Descripción del Proyecto

Trail running es un deporte que consiste en correr "fuera de pista", por senderos de montaña, huellas, rastros o caminos secundarios, a través de montañas, cerros y montes, cruzando arroyos y ríos, con grandes trepadas y abruptas bajadas.

Detalles del Proyecto

- Trail running
- Montaña
- Running
- Resistencia

|| Stop

Related Projects



Copyright © Your Website 2014

```
<div id="progress" class="progress">
  <div class="progress-bar progress-bar-info" role="progressbar"
    aria-valuenow="0" aria-valuemin="0" aria-valuemax="100" style="width: 0%">
    <span class="sr-only">0% Complete</span>
  </div>
</div>
```

CODIFICACIÓN

```

var imagenes = ['img1_750x500.jpg'...];

var descripcion = ['La cultura hipster ...'];

var detalles = ['<li>Hipster</li>...']

var myVar2 = setInterval(function () { myTimer2() }, 4000);
var i = 1;
function myTimer2() {
    document.getElementById("portfolio").innerHTML = "<img class='img-responsive' src='img/" + imagenes[i] + "'></img>";

    document.getElementById("descripcion").innerHTML =
descripcion[i];

    document.getElementById("detalles").innerHTML = detalles[i];

    i++;
    if (i == (imagenes.length)) { i = 0; };
}
function myStopFunction() {
    clearInterval(myVar2);
    clearInterval(myVar3);

    document.getElementById("stopplay").innerHTML = "<button
id='stopplay' type='button' class='btn btn-default btn-lg'
onclick='myPlayFunction()'><span class='glyphicon glyphicon-play' aria-
hidden='true'> Play</span></button>";

    document.getElementById("progress").innerHTML = "<div
class='progress-bar progress-bar-info' role='progressbar' aria-valuenow='0'
aria-valuemin='0' aria-valuemax='100' style='width: 0%><span class='sr-
only'>0% Complete</span>"

}
function myPlayFunction() {

```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

```

myVar2 = setInterval(function () { myTimer2() }, 4000);

myVar3 = setInterval(function () { progress() }, 40);

document.getElementById("stopplay").innerHTML = "<button
id='stopplay' type='button' class='btn btn-default btn-lg'
onclick='myStopFunction()'><span class='glyphicon glyphicon-stop' aria-
hidden='true'> Stop</span></button>";

tantoporcentaje = 0

}

function cambiarA(j) {

    document.getElementById("portfolio").innerHTML = "<img class='img-
responsive' src='img/" + imagenes[j] + "'></img>";

    document.getElementById("descripcion").innerHTML =
descripcion[j];

    document.getElementById("detalles").innerHTML = detalles[j];

    i = j+1;

    myStopFunction();

    tantoporcentaje = 0

}

var myVar3 = setInterval(function () { progress() }, 40);
var tantoporcentaje = 0;
function progress() {

    document.getElementById("progress").innerHTML = "<div
class='progress-bar progress-bar-info' role='progressbar' aria-valuenow='" +
tantoporcentaje + "' aria-valuemin='0' aria-valuemax='100' style='width: " +
tantoporcentaje + "%'><span class='sr-only'>" + tantoporcentaje + "%
Complete</span>";

    tantoporcentaje += 1;

    if (tantoporcentaje == 100) { tantoporcentaje = 0; };

}

```


BUCLES

For

Mostrar los números pares entre el 1 y el 100:

```
<script language="JavaScript" type="text/JavaScript">

function bucle(){

    document.write("<h3>Números pares entre 1 y 100...</h3>");

    for(i=1;i<=100;i++){
        if(i%2==0){
            document.write(i+", ");
        }
        if(i%10==0){
            document.write("<br />");
        }
    }
    document.write("<h3>...Hecho</h3>");
}

bucle();

</script>
```

Números pares entre 1 y 100...

2, 4, 6, 8, 10,
12, 14, 16, 18, 20,
22, 24, 26, 28, 30,
32, 34, 36, 38, 40,
42, 44, 46, 48, 50,
52, 54, 56, 58, 60,
62, 64, 66, 68, 70,
72, 74, 76, 78, 80,
82, 84, 86, 88, 90,
92, 94, 96, 98, 100,

...Hecho

Sumatorio de cada número comprendido entre 1 y 100, valor que se obtiene al sumarle a un número todos los valores comprendidos entre él y el 0.

```
<script language="JavaScript" type="text/JavaScript">

function bucle(){

    var sumatorio=0;
    document.write("<h3>Sumatorio de cada valor entre 1 y 100...</h3>");

    for(i=1;i<=100;i++){
        for(j=i;j>0;j--){
            sumatorio+=j;
        }
        document.write(sumatorio+", ");
        if(i%10==0){
            document.write("<br />");
        }
        sumatorio=0;
    }
    document.write("<h3>...Hecho</h3>");
}

bucle();

</script>
```

Sumatorio de cada valor entre 1 y 100...

1, 3, 6, 10, 15, 21, 28, 36, 45, 55,
66, 78, 91, 105, 120, 136, 153, 171, 190, 210,
231, 253, 276, 300, 325, 351, 378, 406, 435, 465,
496, 528, 561, 595, 630, 666, 703, 741, 780, 820,
861, 903, 946, 990, 1035, 1081, 1128, 1176, 1225, 1275,
1326, 1378, 1431, 1485, 1540, 1596, 1653, 1711, 1770, 1830,
1891, 1953, 2016, 2080, 2145, 2211, 2278, 2346, 2415, 2485,
2556, 2628, 2701, 2775, 2850, 2926, 3003, 3081, 3160, 3240,
3321, 3403, 3486, 3570, 3655, 3741, 3828, 3916, 4005, 4095,
4186, 4278, 4371, 4465, 4560, 4656, 4753, 4851, 4950, 5050,

...Hecho

for...in

Produce una iteración a través de todas las propiedades de un objeto.

```
<script language="JavaScript" type="text/JavaScript">

function bucle(){

    document.write("<h3>Propiedades del objeto DOCUMENT...</h3>");

    for(var propiedad in document){
        document.write("document."+propiedad+" = "+ document[propiedad]);
        document.write("<br />");
    }
}

bucle();

</script>
```

Propiedades del objeto DOCUMENT...

```
document.close = function close() { [native code] }
document.open = function open() { [native code] }
document.releaseEvents = function releaseEvents() { [native code] }
document.captureEvents = function captureEvents() { [native code] }
document.write = function write() { [native code] }
document.writeln = function writeln() { [native code] }
document.getElementsByName = function getElementsByName() { [native code] }
document.elementFromPoint = function elementFromPoint() { [native code] }
document.createEvent = function createEvent() { [native code] }
document.getSelection = function getSelection() { [native code] }
document.implementation = [object HTMLDOMImplementation]
document.clear = function clear() { [native code] }
document.getElementById = function getElementById() { [native code] }
document.getElementsByTagName = function getElementsByTagName() { [native code] }
document.createElement = function createElement() { [native code] }
document.createElementNS = function createElementNS() { [native code] }
document.createDocumentFragment = function createDocumentFragment() { [native code] }
document.createTextNode = function createTextNode() { [native code] }
document.createComment = function createComment() { [native code] }
document.createAttribute = function createAttribute() { [native code] }
document.importNode = function importNode() { [native code] }
document.attachEvent = function attachEvent() { [native code] }
document.detachEvent = function detachEvent() { [native code] }
document.addEventListener = function addEventListener() { [native code] }
document.cloneNode = function cloneNode() { [native code] }
document.normalize = function normalize() { [native code] }
document.removeEventListener = function removeEventListener() { [native code] }
document.dispatchEvent = function dispatchEvent() { [native code] }
document.contains = function contains() { [native code] }
document.insertBefore = function insertBefore() { [native code] }
document.replaceChild = function replaceChild() { [native code] }
document.removeChild = function removeChild() { [native code] }
document.appendChild = function appendChild() { [native code] }
document.hasChildNodes = function hasChildNodes() { [native code] }
document.isSupported = function isSupported() { [native code] }
document.hasAttributes = function hasAttributes() { [native code] }
```

while

Número múltiplos de 3 entre el 1 y el 100

```
<script language="JavaScript" type="text/JavaScript">

function bucle(){

    var i=1;
    document.write("<h3>Número múltiplos de 3 entre 1 y 100...</h3>");

    while(i<=100){
        if(i%3==0){
            document.write(i+", ");
        }
        if(i%10==0){
            document.write("<br />");
        }
        i++;
    }
}

bucle();

</script>
```

Número múltiplos de 3 entre 1 y 100...

3, 6, 9,
12, 15, 18,
21, 24, 27, 30,
33, 36, 39,
42, 45, 48,
51, 54, 57, 60,
63, 66, 69,
72, 75, 78,
81, 84, 87, 90,
93, 96, 99,

Do...while

```
<script language="JavaScript" type="text/JavaScript">

function bucle(){

    var k=0;
    document.write("<h3>Número pares entre 1 y ...</h3>");

    do{
        k=k+50;
        for(i=k+1-50;i<=k;i++){
            if(i%2==0){
                document.write(i+" ");
            }
            if(i%10==0){
                document.write("<br />");
            }
        }
    }while (confirm("¿Deseas ver los 50 próximos pares?"));
}

bucle();

</script>
```

Número pares entre 1 y ...

2 4 6 8 10
12 14 16 18 20
22 24 26 28 30
32 34 36 38 40
42 44 46 48 50



Break y continue

Break finaliza la ejecución de un bucle y **continue** hace saltar las sentencias posteriores a ella y evalúa de nuevo la expresión del bucle continuando con la siguiente iteración.

```
<script language="JavaScript" type="text/JavaScript">

function bucle(){

    var sumatorio=0;
    document.write("<h3>Sumatorio impares inferiores a 1000 ...</h3>");

    for(i=1;i<=100;i++){
        sumatorio=0;
        for(j=i;j>0;j--){
            sumatorio+=j;
        }
        if(i%8==0){document.write("<br />");}
        if(sumatorio>1000){break;}
        if(sumatorio%2==0){continue;}
        document.write(sumatorio+" ", " ");
    }
}
bucle();

</script>
```

Sumatorio impares inferiores a 1000 ...

1, 3, 15, 21,
45, 55, 91, 105,
153, 171, 231, 253,
325, 351, 435, 465,
561, 595, 703, 741,
861, 903,

RELOG DIGITAL

```
<script language="JavaScript" type="text/JavaScript">
```

```
function show(){
var Digital=new Date()
var hours=Digital.getHours()
var minutes=Digital.getMinutes()
var seconds=Digital.getSeconds()
var dn="AM"
if (hours>12){
dn="PM"
hours=hours-12
}
if (hours==0)
hours=12
if (minutes<=9)
minutes="0"+minutes
if (seconds<=9)
seconds="0"+seconds
document.Tick.Clock.value=hours+":"+minutes+":"+
seconds+" "+dn
setTimeout("show()",1000)
}
```

```
</script>
```

```
</head>
```

```
<body onLoad="show()">
```

```
<form name="Tick">
```

```
<input type="text" size="11" name="Clock">
```

```
</form>
```

```
|
```

```
</body>
```

```
</html>
```

10:49:38 AM

PROGRAMA QUE DETERMINA SI UN NÚMERO ES PRIMO

```
<HTML>
<SCRIPT LANGUAGE="JavaScript">
// PROG043.HTM
var num,resto;
num=prompt("Escribe un número entero","");
num=parseInt(num,10);
for (i=2;i<num-1;i++)
{
    resto=num % i;
    if ((resto==0) && (num != 2))
    {
        alert(num+" no es primo");
        break;
    }
}
alert("Si no ha aparecido un mensaje de que no es primo, entonces el número
"+num+" es primo");
</SCRIPT>
</HTML>
```


Acceso al DOM

HTML

```
<h1>Acceso a los párrafos</h1>
```

```
<div id="cuadro"></div>
```

```
<p>Este es el primer párrafo de esta página</p>
```

```
<p>Este es el segundo párrafo</p>
```

```
<p>Y este el tercero</p>
```

```
<p onclick="acceder()">Pulsa en este cuarto párrafo para volver a verlos  
todos en el recuadro de la derecha.</p>
```

```
<p onclick="borrar()">Y pulsa en el quinto para borrar el recuadro de la  
derecha.</p>
```

CSS

```
#cuadro { width: 400px; height: 200px; border: 1px black solid; float: right; }
```

```
p { font-size: 1em; font-family: arial; }
```

JAVASCRIPT

```
function acceder() {
    var texto = document.getElementById("cuadro")
    texto.innerHTML = ""
    var parrafos = document.getElementsByTagName("p")
    for (i=0 ; i<parrafos.length ; i++ ) {
        texto.innerHTML += parrafos[i].innerHTML + "<br/>";
    }
}

function borrar() {
    var texto = document.getElementById("cuadro")
    texto.innerHTML = ""
}
```



Acceso a los párrafos

Este es el primer párrafo de esta página

Este es el segundo párrafo

Y este el tercero

Pulsa en este cuarto párrafo para volver a verlos todos en el recuadro de la derecha.

Y pulsa en el quinto para borrar el recuadro de la derecha.

Este es el primer párrafo de esta página

Este es el segundo párrafo

Y este el tercero

Pulsa en este cuarto párrafo para volver a verlos todos en el recuadro de la derecha.

Y pulsa en el quinto para borrar el recuadro de la derecha.

Manipular los nodos

HTML

```
<h1>Manipulación de nodos. </h1>
```

```
<div id="cuadro">
```

```
<h4 id="primero">primer texto fijo</h4>
```

```
<h4 id="segundo">Segundo texto fijo</h4>
```

```
<h4 id="tercero">Tercer texto fijo</h4>
```

```
</div>
```

<p>Pulsa en los siguientes párrafos, para cambiar los textos de la caja de la derecha</p>

<p onclick="restaurar()">Pulsa en este párrafo para restaurar el estado inicial de la caja.</p>

<p onclick="debajo()">Insertar nuevo elemento debajo de todos</p>

<p onclick="delante()">Insertar nuevo elemento delante del segundo.</p>

<p onclick="reemplazar()">reemplazar el primer elemento de la caja por otro nuevo</p>

<p onclick="suprimir()">Suprimir el tercer párrafo</p>

<p onclick="cambiar()">Poner el segundo texto en último lugar</p>

<p onclick="copiar()">Copia del nuevo elemento, se inserta al final</p>

CSS

```
#cuadro { width: 400px; height: 250px; border: 1px black solid; float: right;
```

```
font-family: arial; overflow: auto }
```

```
p { font-size: 1em; font-family: arial;
```

JAVASCRIPT

```

var nuevoElemento = document.createElement("p");
var nuevoTexto = document.createTextNode("Nuevo elemento en la página");
var nuevo = nuevoElemento.appendChild(nuevoTexto);
function restaurar(){ //restaurar el valor inicial mediante innerHTML
    var lugar = document.getElementById("cuadro");
    lugar.innerHTML = "<h4 id='primero'>primer texto fijo</h4><h4
id='segundo'>Segundo texto fijo</h4><h4 id='tercero'>Tercer texto fijo</h4>"
}
function debajo() { //insertar elemento al final -appendChild()-
    var lugar = document.getElementById("cuadro")
    lugar.appendChild(nuevo)
}
function delante() { //Insertar elemento delante de otro -insertBefore()
    var segundo = document.getElementById("segundo")
    var padre = segundo.parentNode
    padre.insertBefore(nuevo,segundo)
}
function reemplazar() { //Reemplazar elemento con -replaceChild()-
    var primero = document.getElementById("primero")
    var padre = primero.parentNode
    padre.replaceChild(nuevo,primero)
}
function suprimir() { //Suprimir un elemento -removeChild()-
    var tercero = document.getElementById("tercero")
    tercero.parentNode.removeChild(tercero)
}
function cambiar() { //cambiar de sitio : removeChild() + appendChild()
    var segundo = document.getElementById("segundo")
    segundo.parentNode.removeChild(segundo)
    document.getElementById("cuadro").appendChild(segundo)
}

```

```
function copiar() { //copia de un nodo: cloneNode(true)
    var segundo = document.getElementById("segundo")
    copia = segundo.cloneNode(true)
    document.getElementById("cuadro").appendChild(copia)
}
```

Manipulación de nodos.

Pulsa en los siguientes párrafos, para cambiar los textos de la caja de la derecha

Pulsa en este párrafo para restaurar el estado inicial de la caja.

Insertar nuevo elemento debajo de todos

Insertar nuevo elemento delante del segundo.

reemplazar el primer elemento de la caja por otro nuevo

Suprimir el tercer párrafo

Poner el segundo texto en último lugar

Copia del nuevo elemento, se inserta al final

primer texto fijo

Segundo texto fijo

Tercer texto fijo

JavaScript Cookies

HTML

```
<input type="button" value="dime tu nombre" onclick="setCookie()"/>
```

```
<input type="button" value="tu nombre es" onclick="getCookie()"/>
```



Set a Cookie

```
function setCookie() {
    var nombre =prompt("dime tu nombre")
    document.cookie = "username=" + nombre;
}
```

Get a Cookie

```
function getCookie() {
    var name = "username=";
    var ca = document.cookie.split(';');
    for(var i = 0; i <ca.length; i++) {
        var c = ca[i];
        while (c.charAt(0)==' ') {
            c = c.substring(1);
        }
        if (c.indexOf(name) == 0) {
            alert(c.substring(name.length,c.length));
        }
    }
}
```

API Canvas

La API Canvas se hace cargo del aspecto gráfico y lo hace de una forma extremadamente efectiva. Canvas nos permite dibujar, presentar gráficos en pantalla, animar y procesar imágenes y texto, y trabaja junto con el resto de la especificación para crear aplicaciones completas.

El elemento <canvas>

Este elemento genera un espacio rectangular vacío en la página web (lienzo).

```
<div style="position: absolute; top: 50px; left: 50px;">
```

```
<canvas id="lienzo" width="500" height="300">
```

Su navegador no soporta el elemento canvas

```
</canvas>
```

```
</div>
```

Los atributos **width** (ancho) y **height** (alto) declaran el tamaño del lienzo en píxeles.

Al atributo **id**, como en otros casos, nos facilita el acceso al elemento desde el código Javascript.

getContext()

El método **getContext()** es el primer método que tenemos que llamar para dejar al elemento <canvas> listo para trabajar. Genera un contexto de dibujo que será asignado al lienzo.

```
function dibujar(){
    var theCanvas = document.getElementById('canvas');
    var context = theCanvas.getContext('2d');
}
```

```
addEventListener('load', dibujar);
```

El contexto de dibujo del lienzo será una grilla de píxeles listados en filas y columnas de arriba a abajo e izquierda a derecha, con su **origen** (el píxel 0,0) ubicado en la **esquina superior izquierda** del lienzo.

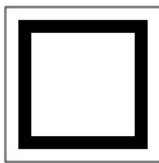
Desarrollo y reutilización de componentes de software mediante lenguajes de guión

Dibujando rectángulos

fillRect(x, y, ancho, alto) Este método dibuja un rectángulo sólido. La esquina superior izquierda será ubicada en la posición especificada por los atributos x e y. Los atributos ancho y alto declaran el tamaño.

strokeRect(x, y, ancho, alto) Similar al método anterior, éste dibujará un rectángulo vacío (solo su contorno).

clearRect(x, y, ancho, alto) Este método es usado para substraer pixeles del área especificada por sus atributos. Es un borrador rectangular.

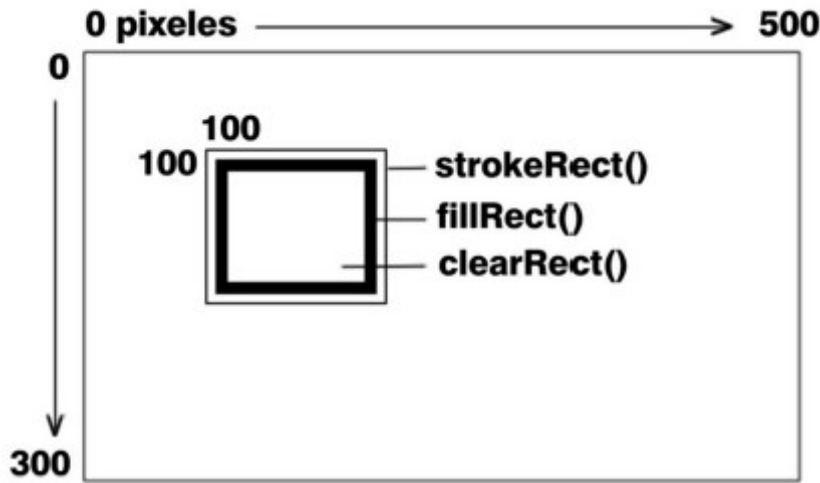


```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title></title>
    <script>

      function iniciar(){
        var elemento=document.getElementById('lienzo');
        lienzo=elemento.getContext('2d');
        lienzo.strokeRect(100,100,120,120);
        lienzo.fillRect(110,110,100,100);
        lienzo.clearRect(120,120,80,80);
      }
      window.addEventListener("load", iniciar, false);
    </script>
  </head>
  <body>
    <div style="position: absolute; top: 50px; left: 50px;">

      <canvas id="lienzo" width="500" height="300">
        Su navegador no soporta el elemento canvas
      </canvas>

    </div>
  </body>
</html>
```

El primer método usado en la función, `strokeRect(100,100,120,120)`, dibuja un rectángulo vacío con la esquina superior izquierda en la posición 100,100 y un tamaño de 120 píxeles (este es un cuadrado de 120 píxeles). El segundo método, `fillRect(110,110, 100,100)`, dibuja un rectángulo sólido, esta vez comenzando desde la posición 110,110 del lienzo. Y finalmente, con el último método, `clearRect(120,120,80,80)`, un recuadro de 80 píxeles es sustraído del centro de la figura.

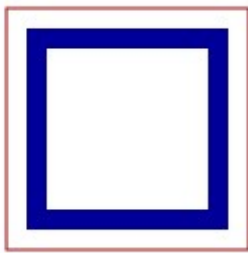
Los rectángulos son dibujados en el lienzo en la posición declarada por los atributos x e y, y uno sobre el otro de acuerdo al orden en el código (el primero en aparecer en el código será dibujado primero, el segundo será dibujado por encima del anterior, y así sucesivamente).

Colores

strokeStyle Esta propiedad declara el color para el contorno de la figura.

fillStyle Esta propiedad declara el color para el interior de la figura.

globalAlpha Esta propiedad no es para definir color sino transparencia. Especifica la transparencia para todas las figuras dibujadas en el lienzo.



```
function iniciar(){
    var elemento=document.getElementById('lienzo');
    lienzo=elemento.getContext('2d');

    lienzo.fillStyle="#000099";
    lienzo.strokeStyle="#990000";

    lienzo.strokeRect(100,100,120,120);
    lienzo.fillRect(110,110,100,100);
    lienzo.clearRect(120,120,80,80);
}

window.addEventListener("load", iniciar, false);
```

```
function pintar(){
    var elemento = document.getElementById('lienzo');
    lienzo=elemento.getContext('2d');

    color=true;
    x=0;
    y=0;
    for(var fila=1; fila<=8;fila++){
        for(var columna=1; columna<=8;columna++){
            if(color){
                color=false;

            }else{
                color=true;
            }

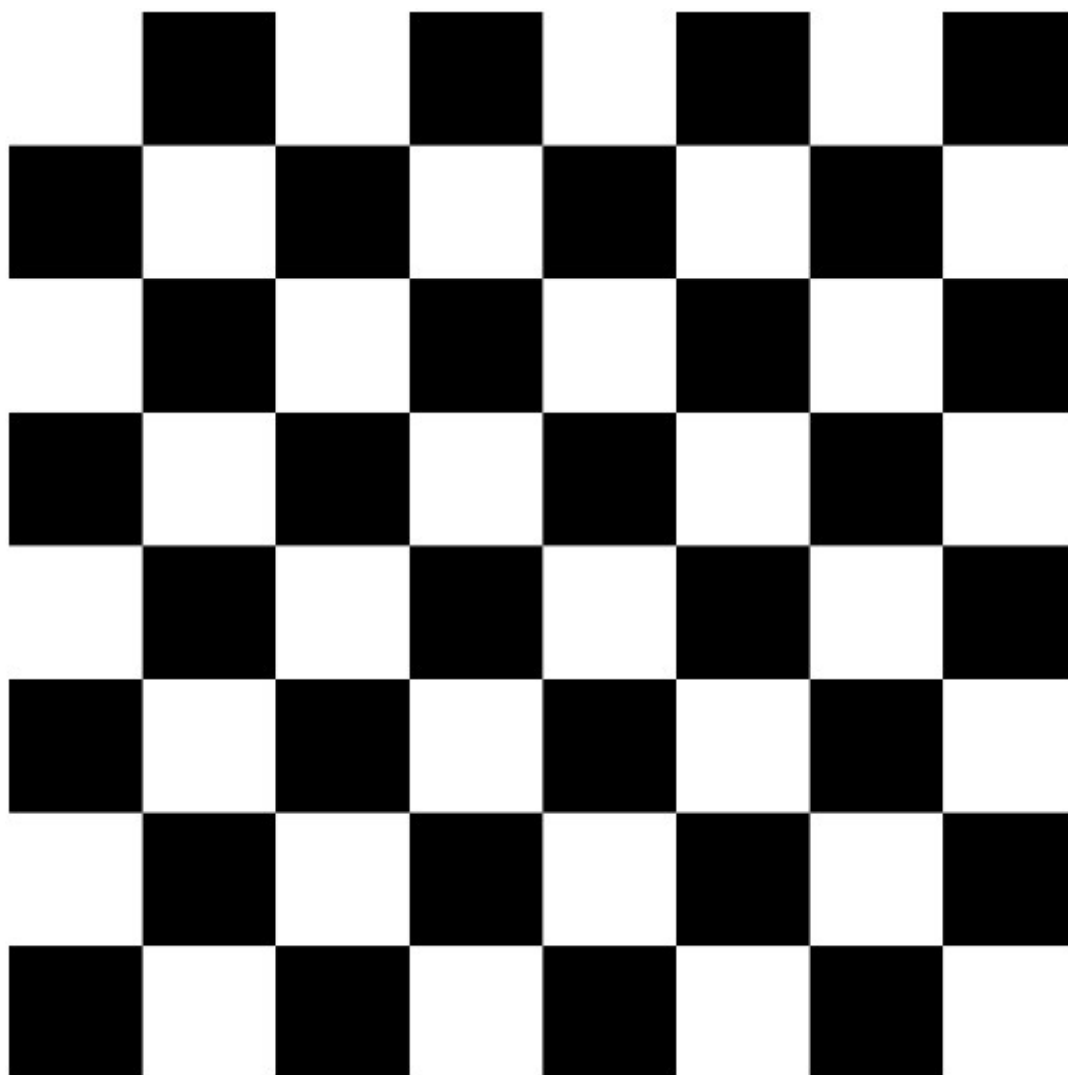
            if(color){
                lienzo.fillRect(x,y,100,100);
            }
            x+=100;

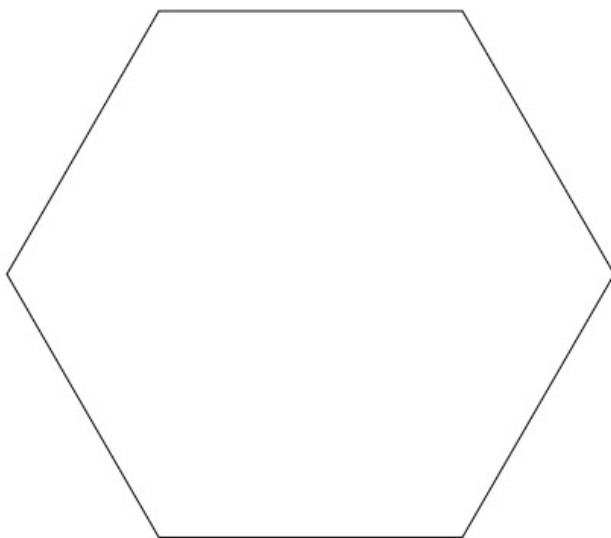
        }//FIN COLUMNA
        x=0;
        y+=100;
        if(color){
            color=false;

        }else{
            color=true;
        }
    }//FIN FILA

}

pintar();
```





<canvas id="lienzo" width="1000px" height="1000px"></canvas>

```
function dibujar(){
    var elemento=document.getElementById('lienzo');
    lienzo=elemento.getContext('2d');

    lienzo.beginPath();

        p1x=200;
        p1y=200;

        p2x=p1x+200;
        p2y=p1y+0;

        p3x=p2x+200*Math.sin(Math.PI/6);
        p3y=p2y+200*Math.cos(Math.PI/6);

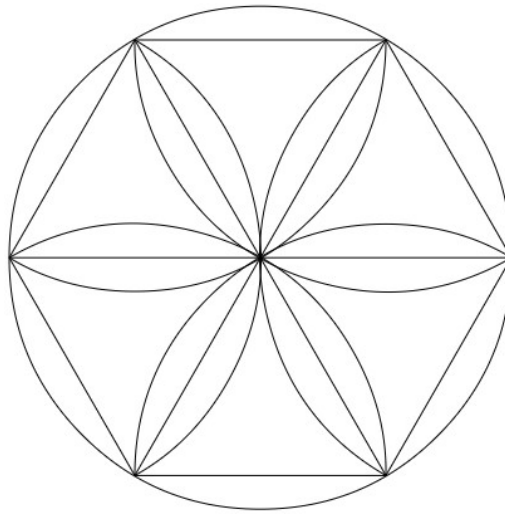
        p4x=p3x-200*Math.sin(Math.PI/6);
        p4y=p3y+200*Math.cos(Math.PI/6);

        p5x=p4x-200;
        p5y=p4y+0;

        p6x=p5x-200*Math.sin(Math.PI/6);
        p6y=p5y-200*Math.cos(Math.PI/6);

    lienzo.moveTo(p1x,p1y);
    lienzo.lineTo(p2x,p2y);
    lienzo.lineTo(p3x,p3y);
        lienzo.lineTo(p4x,p4y);
        lienzo.lineTo(p5x,p5y);
        lienzo.lineTo(p6x,p6y);
        lienzo.lineTo(p1x,p1y);
    lienzo.stroke();
}

window.addEventListener("load", dibujar, false);
```



```
function dibujar(){
    var elemento=document.getElementById('lienzo');
    lienzo=elemento.getContext('2d');

    lienzo.beginPath();

        p1x=200;
        p1y=200;

        p2x=p1x+200;
        p2y=p1y+0;

        p3x=p2x+200*Math.sin(Math.PI/6);
        p3y=p2y+200*Math.cos(Math.PI/6);

        p4x=p3x-200*Math.sin(Math.PI/6);
        p4y=p3y+200*Math.cos(Math.PI/6);

        p5x=p4x-200;
        p5y=p4y+0;

        p6x=p5x-200*Math.sin(Math.PI/6);
```

```

p6y=p5y-200*Math.cos(Math.PI/6);

pcx=p1x+200*Math.sin(Math.PI/6);
pcy=p1y+200*Math.cos(Math.PI/6);

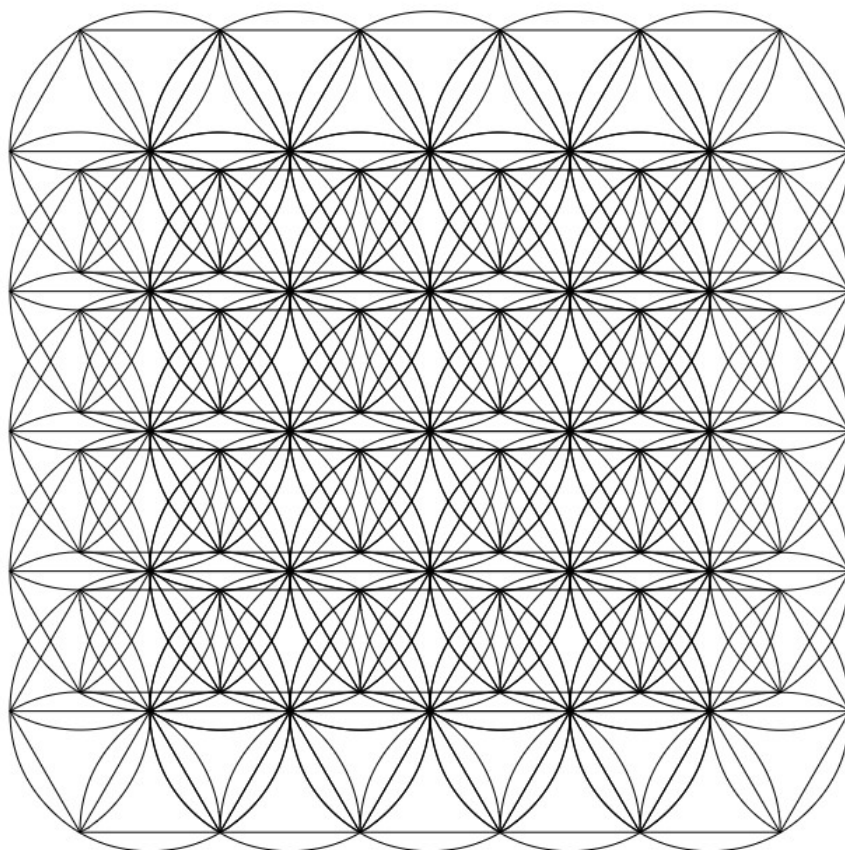
lienzo.moveTo(p1x,p1y);
lienzo.lineTo(p2x,p2y);
lienzo.lineTo(p3x,p3y);
    lienzo.lineTo(p4x,p4y);
    lienzo.lineTo(p5x,p5y);
    lienzo.lineTo(p6x,p6y);
    lienzo.lineTo(p1x,p1y);
    lienzo.lineTo(p4x,p4y);
    lienzo.moveTo(p2x,p2y);
    lienzo.lineTo(p5x,p5y);
    lienzo.moveTo(p3x,p3y);
    lienzo.lineTo(p6x,p6y);
    lienzo.moveTo(p3x,p3y);
    lienzo.arc(pcx,pcy,200,0,Math.PI*2, false);
    lienzo.moveTo(p2x,p2y);
    lienzo.arc(p1x,p1y,200,Math.PI/3*0,Math.PI/3*2, false);
    lienzo.moveTo(p3x,p3y);
    lienzo.arc(p2x,p2y,200,Math.PI/3*1,Math.PI/3*3, false);
    lienzo.moveTo(p4x,p4y);
    lienzo.arc(p3x,p3y,200,Math.PI/3*2,Math.PI/3*4, false)
    lienzo.moveTo(p5x,p5y);
    lienzo.arc(p4x,p4y,200,Math.PI/3*3,Math.PI/3*5, false);

    lienzo.moveTo(p6x,p6y);
    lienzo.arc(p5x,p5y,200,Math.PI/3*4,Math.PI/3*6, false);

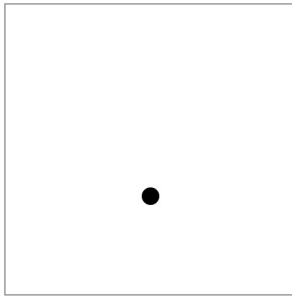
    lienzo.moveTo(p1x,p1y);
    lienzo.arc(p6x,p6y,200,Math.PI/3*5,Math.PI/3*1, false);
    lienzo.stroke();
}

window.addEventListener("load", dibujar, false);

```

API Canvas Animación I



```
function dibujar(){
    var elemento = document.getElementById('lienzo');
    lienzo = elemento.getContext('2d');
    var speed=5;
    var x=250;
    var y=10;
    setInterval(pintar,33);
    function pintar(){
        //Dibujar el rectángulo
        lienzo.fillStyle='#ffffff';
        lienzo.fillRect(0,0,elemento.width,elemento.height);
        //Dibujar el marco
        lienzo.strokeStyle='#000000';
        lienzo.strokeRect(1,1,elemento.width-2,elemento.height-2);
        //Crear la pelota
        y+=speed;
        lienzo.fillStyle='#000000';
        lienzo.beginPath();
        lienzo.arc(x,y,15,0,Math.PI*2,true);
        lienzo.closePath();
        lienzo.fill();
    }

    };//FIN dibujar()

    window.addEventListener('load', dibujar, false);
```

API Canvas Animación II

```
function dibujar(){
    var elemento = document.getElementById('lienzo');
    lienzo = elemento.getContext('2d');

    var velocidad=5;

    //Dos elementos que establecerán el inicio y el final de nuestro movimiento.
    var p1={x:20, y:250};
    var p2={x:480, y:250};

    //Calcular la diferencia entre coordenadas del segundo al primero.
    var dx= p2.x - p1.x;
    var dy= p2.y - p1.y;

    //Para determinar la distancia elevaremos al cuadrado dx y dy los sumaremos y calcularemos
    la raiz cuadrada.
    var distancia= Math.sqrt(dx*dx+dy*dy);

    //Calculamos cuantas veces hay que llamar a la funcion pintar para ir de 1 a 2.
    var tiempo=distancia/velocidad;

    //Calculamos la distancia que hay que desplazar.
    var xunidades=(p2.x - p1.x)/tiempo;
    var yunidades=(p2.y - p1.y)/tiempo;

    //Generamos un objeto dinámico bola que se encontrará en las coordenadas x e y de p1.
    var bola={x:p1.x, y:p1.y};

    //llamaremos a pintar cada 33 milisegundos.
    setInterval(pintar,33);

    function pintar(){
        //Dibujar el rectángulo
        lienzo.fillStyle='#ffffff';
        lienzo.fillRect(0,0,elemento.width,elemento.height);
    }
}
```

```
//Dibujar el marco
lienzo.strokeStyle='#000000';
lienzo.strokeRect(1,1,elemento.width-2,elemento.height-2);

//Crear la pelota
if(tiempo>0){
    tiempo--;
    bola.x +=xunidades;
    bola.y +=yunidades;
}

lienzo.fillStyle='#000000';
lienzo.beginPath();
lienzo.arc(bola.x,bola.y,15,0,Math.PI*2,true);
lienzo.closePath();
lienzo.fill();
}

};//FIN dibujar()

window.addEventListener('load', dibujar, false);
```

API Drag and Drop

Arrastrar y soltar en la web

Cuando el usuario realiza una operación de arrastrar y soltar, el **elemento origen** (el que es arrastrado) dispara estos tres eventos:

dragstart Este evento es disparado en el momento en el que el arrastre comienza. Los datos asociados con el elemento origen son definidos en este momento en el sistema.

drag Este evento es similar al evento **mousemove**, excepto que será disparado durante una operación de arrastre por el elemento origen.

dragend Cuando la operación de arrastrar y soltar finaliza (sea la operación exitosa o no) este evento es disparado por el elemento origen.

Y estos son los eventos disparados por el **elemento destino** (donde el origen será soltado) durante la operación:

dragenter Cuando el puntero del ratón entra dentro del área ocupada por los posibles elementos destino durante una operación de arrastrar y soltar, este evento es disparado.

dragover Este evento es similar al evento **mousemove**, excepto que es disparado durante una operación de arrastre por posibles elementos destino.

drop Cuando el elemento origen es soltado durante una operación de arrastrar y soltar, este evento es disparado por el elemento destino.

dragleave Este evento es disparado cuando el ratón sale del área ocupada por un elemento durante una operación de arrastrar y soltar. Este evento es generalmente usado junto con **dragenter** para mostrar una ayuda visual al usuario que le permita identificar el elemento destino (donde soltar).

Antes de trabajar con esta nueva herramienta, existe un aspecto importante que debemos considerar. Los navegadores realizan acciones por defecto durante una operación de arrastrar y soltar.

Para obtener el resultado que queremos, necesitamos prevenir en algunas ocasiones este comportamiento por defecto y personalizar las reacciones del navegador. Para algunos eventos, como **dragenter**, **dragover** y **drop**, la prevención es necesaria, incluso cuando una acción personalizada ya fue especificada.



```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Drag and Drop</title>
<style>
    #cajasoltar{
float: left;
width: 500px;
height: 300px;
margin: 10px;
border: 1px solid #999999;
}
#cajaimagenes{
float: left;
width: 320px;
margin: 10px;
border: 1px solid #999999;
}
#cajaimagenes > img{
float: left;
padding: 5px;
}
</style>
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

```
<script>
function iniciar(){
    origen1=document.getElementById('imagen');
    origen1.addEventListener('dragstart', arrastrado, false);
    destino=document.getElementById('cajasoltar');
    destino.addEventListener('dragenter', function(e){
        e.preventDefault(); }, false);
    destino.addEventListener('dragover', function(e){
        e.preventDefault(); }, false);
    destino.addEventListener('drop', soltado, false);
    }
function arrastrado(e){
    var codigo='';
    e.dataTransfer.setData("Text", codigo);
    }
function soltado(e){
    e.preventDefault();
    destino.innerHTML=e.dataTransfer.getData("Text");
    }
window.addEventListener("load", iniciar, false);
</script>
</head>
<body>
    <section id="cajasoltar">
        Arrastre y suelte la imagen...
    </section>
    <section id="cajaimagenes">
        
    </section>
</body>
</html>
```

En este nuevo ejemplo, agregamos dos funciones para el elemento destino y una para el elemento origen. Las funciones `entrando()` y `saliendo()` cambiarán el color de fondo del elemento destino cada vez que el puntero del ratón esté arrastrando un objeto y entre o salga del área ocupada por este elemento (estas acciones disparan los eventos `dragenter` y `dragleave`). Además, la función `finalizado()` será llamada por la escucha del evento `dragend` cuando el objeto arrastrado es soltado. Note que este evento o la función misma no controlan si el proceso fue exitoso o no. Este control lo deberemos hacer nosotros en el código.

Gracias a los eventos y funciones agregadas, cada vez que el ratón arrastra un objeto y entra en el área del elemento destino, este elemento se volverá verde, y cuando el objeto es soltado la imagen original es borrada de la pantalla. Estos cambios visibles no están afectando el proceso de arrastrar y soltar, pero sí están ofreciendo una guía clara para el usuario durante la operación. Para prevenir acciones por defecto del navegador, tenemos que usar el método `preventDefault()` en cada función.


```
function iniciar(){
    origen1=document.getElementById('imagen');
    origen1.addEventListener('dragstart', arrastrado, false);
    origen1.addEventListener('dragend', finalizado, false);
    soltar=document.getElementById('cajasoltar');
    soltar.addEventListener('dragenter', entrando, false);
    soltar.addEventListener('dragleave', saliendo, false);
    soltar.addEventListener('dragover', function(e){
        e.preventDefault(); }, false);
    soltar.addEventListener('drop', soltado, false);
}

function entrando(e){
    e.preventDefault();
    soltar.style.background='rgba(0,150,0,.2)';
}

function saliendo(e){
    e.preventDefault();
    soltar.style.background='#FFFFFF';
}

function finalizado(e){
    elemento=e.target;
    elemento.style.visibility='hidden';
}

    function arrastrado(e){
var codigo='';
    e.dataTransfer.setData('Text', codigo);
    }

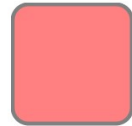
function soltado(e){
    e.preventDefault();
    soltar.style.background='#FFFFFF';
    soltar.innerHTML=e.dataTransfer.getData('Text');
}

window.addEventListener('load', iniciar, false);
```



SVG

SVG <rect>



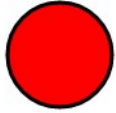
```
<svg width="400" height="110">
  <rect width="300" height="100" style="fill:rgb(0,0,255);stroke-
width:3;stroke:rgb(0,0,0)" />
</svg>
```

```
<svg width="400" height="180">
  <rect x="50" y="20" width="150" height="150"
  style="fill:blue;stroke:blue;stroke-width:5;fill-
opacity:0.1;stroke-opacity:0.9" />
</svg>
```

```
<svg width="400" height="180">
  <rect x="50" y="20" width="150" height="150"
  style="fill:blue;stroke:blue;stroke-width:5;opacity:0.5" />
</svg>
```

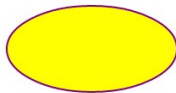
```
<svg width="400" height="180">
  <rect x="50" y="20" rx="20" ry="20" width="150" height="150"
  style="fill:red;stroke:black;stroke-width:5;opacity:0.5" />
</svg>
```

SVG <circle>



```
<svg height="100" width="100">
  <circle cx="50" cy="50" r="40" stroke="black" stroke-width="3"
  fill="red" />
</svg>
```

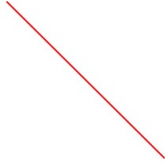
SVG <ellipse>



```
<svg height="140" width="500">
  <ellipse cx="200" cy="80" rx="100" ry="50"
  style="fill:yellow;stroke:purple;stroke-width:2" />
</svg>
```

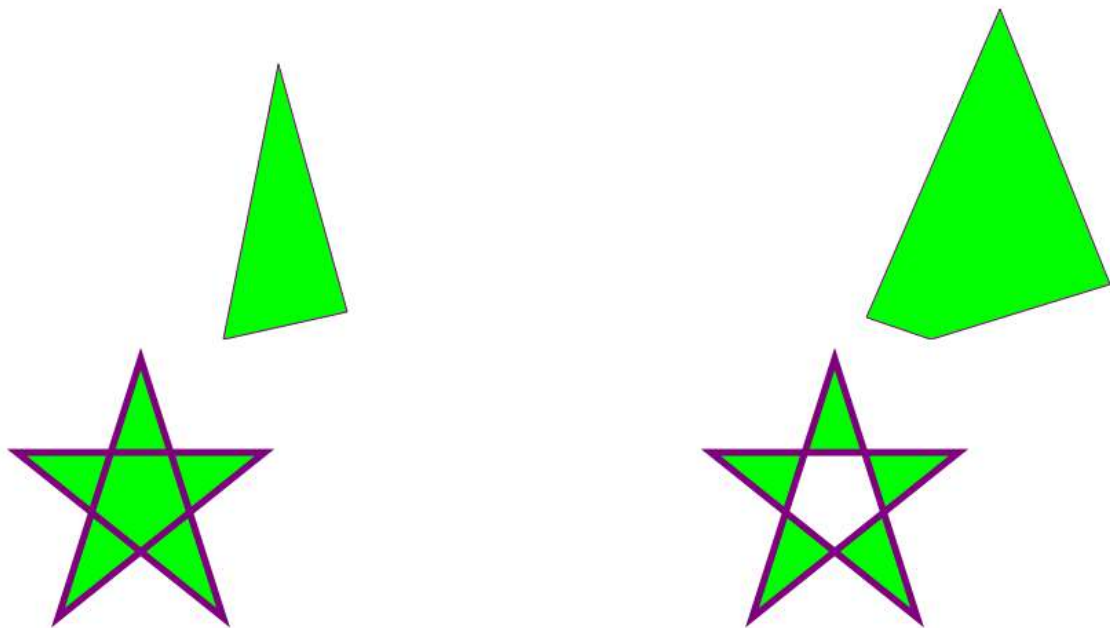
```
<svg height="150" width="500">
  <ellipse cx="240" cy="100" rx="220" ry="30" style="fill:purple" />
  <ellipse cx="220" cy="70" rx="190" ry="20" style="fill:lime" />
  <ellipse cx="210" cy="45" rx="170" ry="15" style="fill:yellow" />
</svg>
```

SVG <line>



```
<svg height="210" width="500">
  <line      x1="0"      y1="0"      x2="200"      y2="200"
  style="stroke:rgb(255,0,0);stroke-width:2" />
</svg>
```

SVG <polygon>



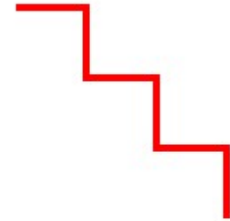
```
<svg height="210" width="500">
  <polygon      points="200,10      250,190      160,210"
  style="fill:lime;stroke:purple;stroke-width:1" />
</svg>
```

```
<svg height="250" width="500">
  <polygon points="220,10 300,210 170,250 123,234"
  style="fill:lime;stroke:purple;stroke-width:1" />
</svg>
```

```
<svg height="210" width="500">
  <polygon points="100,10 40,198 190,78 10,78 160,198"
  style="fill:lime;stroke:purple;stroke-width:5;fill-rule:nonzero;" />
</svg>
```

```
<svg height="210" width="500">
  <polygon points="100,10 40,198 190,78 10,78 160,198"
  style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;" />
</svg>
```

SVG <polyline>

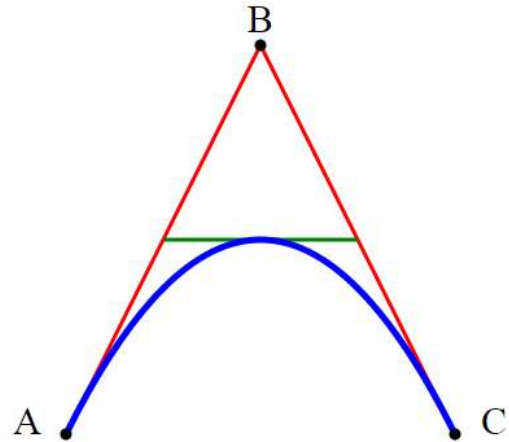
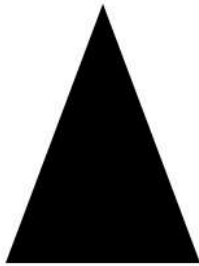


```
<svg height="200" width="500">
  <polyline points="20,20 40,25 60,40 80,120 120,140 200,180"
    style="fill:none;stroke:black;stroke-width:3" />
</svg>
```

```
<svg height="180" width="500">
  <polyline points="0,40 40,40 40,80 80,80 80,120 120,120 120,160"
    style="fill:white;stroke:red;stroke-width:4" />
</svg>
```

SVG <path>

- M = moveto
- L = lineto
- H = horizontal lineto
- V = vertical lineto
- C = curveto
- S = smooth curveto
- Q = quadratic Bézier curve
- T = smooth quadratic Bézier curveto
- A = elliptical Arc
- Z = closepath



```
<svg height="210" width="400">
  <path d="M 150 0 L 75 200 L 225 200 Z" />
</svg>
```

```
<svg height="400" width="450">
  <path id="lineAB" d="M 100 350 l 150 -300" stroke="red"
    stroke-width="3" fill="none" />
  <path id="lineBC" d="M 250 50 l 150 300" stroke="red"
    stroke-width="3" fill="none" />
  <path d="M 175 200 l 150 0" stroke="green" stroke-width="3"
    fill="none" />
  <path d="M 100 350 q 150 -300 300 0" stroke="blue"
    stroke-width="5" fill="none" />
  <!-- Mark relevant points -->
  <g stroke="black" stroke-width="3" fill="black">
    <circle id="pointA" cx="100" cy="350" r="3" />
    <circle id="pointB" cx="250" cy="50" r="3" />
    <circle id="pointC" cx="400" cy="350" r="3" />
  </g>
  <!-- Label the points -->
  <g font-size="30" font="sans-serif" fill="black" stroke="none"
    text-anchor="middle">
    <text x="100" y="350" dx="-30">A</text>
    <text x="250" y="50" dy="-10">B</text>
    <text x="400" y="350" dx="30">C</text>
  </g>
</svg>
```


SVG <text>

I love SVG!

I love SVG

Varias:

Primera.

Segunda.

Google!

```
<svg height="30" width="200">
  <text x="0" y="15" fill="red">I love SVG!</text>
</svg>
```

```
<svg height="60" width="200">
  <text x="0" y="15" fill="red" transform="rotate(30 20,40)">I
love SVG</text>
</svg>
```

```
<svg height="90" width="200">
  <text x="10" y="20" style="fill:red;">Varias:
    <tspan x="10" y="45">Primera.</tspan>
    <tspan x="10" y="70">Segunda.</tspan>
  </text>
</svg>
```

```
<svg height="30" width="200"
xmlns:xlink="http://www.w3.org/1999/xlink">
  <a xlink:href="http://www.google.com" target="_blank">
    <text x="0" y="15" fill="red">Google!</text>
  </a>
</svg>
```

SVG Stroke



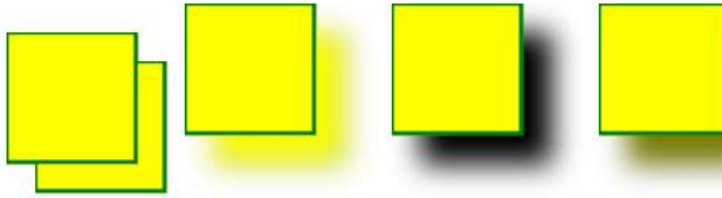
```
<svg height="80" width="300">
  <g fill="none">
    <path stroke="red" d="M5 20 1215 0" />
    <path stroke="blue" d="M5 40 1215 0" />
    <path stroke="black" d="M5 60 1215 0" />
  </g>
</svg>
```

```
<svg height="80" width="300">
  <g fill="none" stroke="black">
    <path stroke-width="2" d="M5 20 1215 0" />
    <path stroke-width="4" d="M5 40 1215 0" />
    <path stroke-width="6" d="M5 60 1215 0" />
  </g>
</svg>
```

```
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="6">
    <path stroke-linecap="butt" d="M5 20 1215 0" />
    <path stroke-linecap="round" d="M5 40 1215 0" />
    <path stroke-linecap="square" d="M5 60 1215 0" />
  </g>
</svg>
```

```
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="4">
    <path stroke-dasharray="5,5" d="M5 20 1215 0" />
    <path stroke-dasharray="10,10" d="M5 40 1215 0" />
    <path stroke-dasharray="20,10,5,5,5,10" d="M5 60 1215 0" />
  </g>
</svg>
```

SVG Drop Shadows



```
<svg height="120" width="120">
  <defs>
    <filter id="f1" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
      <feBlend in="SourceGraphic" in2="offOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
    fill="yellow" filter="url(#f1)" />
</svg>

<svg height="140" width="140">
  <defs>
    <filter id="f2" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
      <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" />
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
    fill="yellow" filter="url(#f2)" />
</svg>

<svg height="140" width="140">
  <defs>
    <filter id="f3" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceAlpha" dx="20" dy="20" />
      <feGaussianBlur result="blurOut" in="offOut" stdDeviation="10" />
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
    fill="yellow" filter="url(#f3)" />
</svg>

<svg height="140" width="140">
  <defs>
    <filter id="f4" x="0" y="0" width="200%" height="200%">
      <feOffset result="offOut" in="SourceGraphic" dx="20" dy="20" />
      <feColorMatrix result="matrixOut" in="offOut" type="matrix"
        values="0.2 0 0 0 0 0.2 0 0 0 0 0.2 0 0 0 0 1 0" />
      <feGaussianBlur result="blurOut" in="matrixOut" stdDeviation="10" />
      <feBlend in="SourceGraphic" in2="blurOut" mode="normal" />
    </filter>
  </defs>
  <rect width="90" height="90" stroke="green" stroke-width="3"
    fill="yellow" filter="url(#f4)" />
</svg>
```



Hudir la Flota

La IA es "inteligente", cuando encuentra un barco dispara alrededor.

HTML

```
<input type="button" id="startBtn" value="Empezar juego">
<div>
  <h3 id="txt1" class="hidden">Shoot'em all</h3>
  <table id="playerGamePanel" class="hidden"></table>
</div>
<div>
  <h3 id="txt2" class="hidden">Tus barcos</h3>
  <table id="iaGamePanel" class="hidden"></table>
</div>
```

CSS

```
.hidden{
  display: none;
}
div{
  float: left;
  margin-left: 18%;
}
tr, td{
  border: 1px solid black;
  background-color: LightBlue;
  height: 25px;
  width: 20px;
}
```

JAVASCRIPT

```

window.onload = function () {
    var startBtn = document.getElementById("startBtn");
    var playerGamePanel = document.getElementById("playerGamePanel");
    var iaGamePanel = document.getElementById("iaGamePanel");
    var playerGamePanelTitle = document.getElementById("txt1");
    var iaGamePanelTitle = document.getElementById("txt2");
    const GAME_SIZE = 20; //Tamaño de la tabla.
    const SHIPS = 15; //Numero de barcos por cada jugador.
    var auxName = 0;
    var iaShots = new Array(0); //Este array contendrá las coordenadas de los
    disparos que haga la IA para no disparar dos veces en el mismo sitio.
    var touch = false; //Este valor marcará true si un barco ha sido tocado,
    y false cuando acabe de hundirlo.
    var trys = 0; //Este valor lo uso para decirle al programa que tiene 4
    intentos cuando toca un barco (tiro arriba, tiro abajo, tiro derecha y tiro
    izquierda).
    var numTouch = 0; //Esto es un contador que aumentará conforme la IA vaya
    tocando el barco para asi disparar cada vez mas lejos.
    var lastX; //Cuando toque un barco, esta variable sera el valor de la x
    de ese td.
    var lastY; //Cuando toque un barco, esta variable sera el valor de la y
    de ese td.
    var iaWin = 0;
    var playerWin = 0;
    document.addEventListener("keypress", prueba, false);
    startBtn.addEventListener("click", startGame, false);

    function prueba(evento) {
        if (evento.keyCode == 13) {
            startGame();
        }
    }

    function startGame() {
        document.removeEventListener("keypress", prueba, false);
        createTable(playerGamePanel);
        createTable(iaGamePanel);
        displayButtons();
        generateShips(playerGamePanel);
        generateShips(iaGamePanel);
    }
}

```

```
function createTable(table) {
    /*
        La id de cada td la usaré para acceder directamente a ella y así
        poder hacer comprobaciones.
        El value de cada td podrá ser Agua "W" o barco "S" y nos indicará si
        hay o no un barco hay. Por defecto será agua.
        El name de cada td lo usaré para indicar que barco hay. Un valor 0
        indica agua o tocado. Por defecto será 0.
        Finalmente le asigno un evento de ratón.
    */
    for (var i = 0; i < GAME_SIZE; i++) {
        var tr = document.createElement("tr");
        for (var j = 0; j < GAME_SIZE; j++) {
            var td = document.createElement("td");
            switch (table) {
                case playerGamePanel:
                    td.id = i + "-" + j;
                    td.value = "W";
                    td.name = 0;
                    td.addEventListener("click", shoot, false);
                    break;
                case iaGamePanel:
                    td.id = i + "/" + j;
                    td.value = "W";
                    break;
            }
            tr.appendChild(td);
        }
        table.appendChild(tr);
    }
}

function displayButtons() {
    startBtn.style.display = "none";
    playerGamePanelTitle.style.display = "block";
    iaGamePanelTitle.style.display = "block";
    playerGamePanel.style.display = "block";
    iaGamePanel.style.display = "block";
}
```

```
function generateShips(table) {
    /*
        Por cada iteración del for valora cuando vale i, a partir de ese
        resultado asigna el valor del tamaño del barco. Yo voy a crear 3 barcos de tamaño
        2, 2 de 3 y 1 de 4.
        Una vez asignado el tamaño, le damos unas coordenadas y una
        dirección.
        La variable ok indica si el barco esta bien posicionado o no.
        Mientras que sea falso, generará nuevas coordenadas y llamara al
        metodo checkShipCollision() que volverá a evaluar si el barco colisiona o no.
        Finalmente, pintamos el barco en la tabla.
    */
    var shipSize = 0;
    for (var i = 0; i < SHIPS; i++) {
        if (i < 5) {
            shipSize = 1;
        } else if (i < 9) {
            shipSize = 2;
        } else if (i < 12) {
            shipSize = 3;
        } else if (i < 14) {
            shipSize = 4;
        } else {
            shipSize = 5;
        }
        var ok = false;
        var direction = Math.round(Math.random()); //0 = Vertical - 1 =
Horizontal
        var x = Math.floor(Math.random() * GAME_SIZE);
        var y = Math.floor(Math.random() * GAME_SIZE);
        while (!ok) {
            if (checkShipCollision(table, shipSize, direction, y, x)) {
                x = Math.floor(Math.random() * GAME_SIZE);
                y = Math.floor(Math.random() * GAME_SIZE);
            } else {
                ok = true;
            }
        }
        repaint(table, shipSize, direction, y, x, i);
    }
}
```



```
function repaint(table, shipSize, direction, y, x, num) {
    /*
    El for recorre el tamaño del barco.
    Pintamos la tabla asignándole value y name para la IA y con el value
    y el color para nuestro panel.
    */
    switch (table) {
        case playerGamePanel:
            for (var i = 0; i < shipSize; i++) {
                switch (direction) {
                    case 0:
                        var auxTd = document.getElementById((y + i) + "-" +
+ x);

                        auxTd.value = "S";
                        auxTd.name = num + 1;
                        break;
                    case 1:
                        var auxTd = document.getElementById(y + "-" + (x
+ i));

                        auxTd.value = "S";
                        auxTd.name = num + 1;
                        break;
                }
            }
            break;
        case iaGamePanel:
            for (var i = 0; i < shipSize; i++) {
                switch (direction) {
                    case 0:
                        var auxTd = document.getElementById((y + i) + "/" +
+ x);

                        auxTd.value = "S";
                        auxTd.style.backgroundColor = "black";
                        break;
                    case 1:
                        var auxTd = document.getElementById(y + "/" + (x
+ i));

                        auxTd.value = "S";
                        auxTd.style.backgroundColor = "black";
                        break;
                }
            }
            break;
    }
}
```

```
function checkShipCollision(table, shipSize, direction, y, x) {
    /*
        Para el case 0: (Vertical), recorreremos la tabla solo cambiando la
        coordenada Y y comprobando si el value es "S".
        Para el case 1: (Horizontal), recorreremos la tabla de manera
        "normal" y comprobando si el value es "S".
    */
    if (y + shipSize >= GAME_SIZE) {
        return true;
    }
    if (x + shipSize >= GAME_SIZE) {
        return true;
    }
    switch (table) {
        case playerGamePanel:
            switch (direction) {
                case 0:
                    for (var i = y; i < y + shipSize; i++) {
                        if (document.getElementById(i + "-" + x).value ==
"S") {
                            return true;
                        }
                        if ((i + 1) < GAME_SIZE &&
document.getElementById((i + 1) + "-" + x).value == "S") {
                            return true;
                        }
                        if ((i - 1) > 1 && document.getElementById((i -
1) + "-" + x).value == "S") {
                            return true;
                        }
                    }
                    break;
                case 1:
                    for (var i = x; i < x + shipSize; i++) {
                        if (document.getElementById(y + "-" + i).value ==
"S") {
                            return true;
                        }
                        if ((i + 1) < GAME_SIZE &&
document.getElementById(y + "-" + (i + 1)).value == "S") {
                            return true;
                        }
                        if ((i - 1) > 1 && document.getElementById(y + "-"
+ (i - 1)).value == "S") {
                            return true;
                        }
                    }
                    break;
            }
        return false;
    }
}
```

```

        case iaGamePanel:
            switch (direction) {
                case 0:
                    for (var i = y; i < y + shipSize; i++) {
                        if (document.getElementById(i + "/" + x).value ==
"S") {
                            return true;
                        }
                        if ((i + 1) < GAME_SIZE &&
document.getElementById((i + 1) + "/" + x).value == "S") {
                            return true;
                        }
                        if ((i - 1) > 1 && document.getElementById((i -
1) + "/" + x).value == "S") {
                            return true;
                        }
                    }
                    break;
                case 1:
                    for (var i = x; i < x + shipSize; i++) {
                        if (document.getElementById(y + "/" + i).value ==
"S") {
                            return true;
                        }
                        if ((i + 1) < GAME_SIZE &&
document.getElementById(y + "/" + (i + 1)).value == "S") {
                            return true;
                        }
                        if ((i - 1) > 1 && document.getElementById(y +
"/" + (i - 1)).value == "S") {
                            return true;
                        }
                    }
                    break;
            }
            return false;
            break;
    }
}

```

```
function shoot() {
    /*
        Gracias a que antes he asignado un name "unico" a cada barco ahora
        podremos saber si esta tocado o tocado y hundido.
        Cada vez que un disparo acierta, se suma 1 a playerWin.
        Finalmente compruebo si el jugador a ganado.
        Sino, llama al metodo de diparo de la IA y le quita el evento al td.
    */
    switch (this.value) {
        case "W":
            this.style.backgroundColor = "blue";
            break;
        case "S":
            /*
                Asigno a auxName el valor del name actual de la casilla (ese
                valor nos indicaba de que barco se trataba).
                y luego lo pongo en 0, que era el de por defecto para
                agua/tocado.
            */
            auxName = this.name;
            this.name = 0;
            if (isShipAlive()) {
                alert("Tocado");
                playerWin++;
            } else {
                alert("Tocado y hundido");
                playerWin++;
            }
            this.style.backgroundColor = "black";
            break;
    }
    if (playerWin == 35) {
        alert("Enhorabuena, has ganado!");
        location.reload(true);
    } else {
        iaShoot();
        this.removeEventListener("click", shoot, false);
    }
}

function iaShoot() {
    /*
        En este método genero disparos aleatorios.
        Lo primero es poner el disparo como incorrecto, y asignarle las
        coordenadas X e Y.
        Luego miramos si aun quedan intentos (los intentos se ponian en 4
        cuando acertaba). Si no quedan, tocado se pone en false y reiniciamos el contador
        de los aciertosTocados.
        Si tocado es true:
        Hacemos un switch de trys, cada case sera una opcion (a recordar,
        izquierda, derecha, arriba y abajo).
        Le asignamos el nuevo valor y comprobamos que ese td no este ya
        disparado o que este OOB.
        Si cualquiera de esos dos metodos devuelve true entonces resta 1 al
        valor de trys y vuelve a llamar al metodo por lo que en la proxima llamada me
        meterá en el siguiente case.
        Sino seguimos generando aleatorios hasta que haga un disparo no
        repetido.
        Por ultimo llamamos al método que nos comprueba que hay en el td del
        disparo.
    */
}
```

```

*/
var ok = false;
var iaShootX = Math.floor(Math.random() * GAME_SIZE);
var iaShootY = Math.floor(Math.random() * GAME_SIZE);
if (trys < 1) {
    touch = false;
    numTouch = 0;
}
if (touch) {
    switch (trys) {
        case 4: //Comprobamos la casilla de izquierda del ultimo
disparo.
            iaShootX = lastX - numTouch;
            iaShootY = lastY;
            if (shotBefore(iaShootY, iaShootX)) {
                trys--;
                numTouch = 1;
                iaShoot();
            } else if (iaShootX < 0) {
                trys--;
                numTouch = 1;
                iaShoot();
            }
            break;
        case 3: //Comprobamos la casilla de la derecha del ultimo
disparo.
            iaShootX = lastX + numTouch;
            iaShootY = lastY;
            if (shotBefore(iaShootY, iaShootX)) {
                trys--;
                numTouch = 1;
                iaShoot();
            } else if (iaShootX >= GAME_SIZE) {
                trys--;
                numTouch = 1;
                iaShoot();
            }
            break;
        case 2: //Comprobamos la casilla de arriba del ultimo
disparo.
            iaShootX = lastX;
            iaShootY = lastY - numTouch;
            if (shotBefore(iaShootY, iaShootX)) {
                trys--;
                numTouch = 1;
                iaShoot();
            } else if (iaShootY < 0) {
                trys--;
                numTouch = 1;
                iaShoot();
            }
            break;
        case 1: //Comprobamos la casilla de abajo del ultimo disparo.
            iaShootX = lastX;
            iaShootY = lastY + numTouch;
            if (shotBefore(iaShootY, iaShootX)) {
                trys--;
                numTouch = 1;
                iaShoot();
            }
    }
}

```

```

    } else if (iaShootY >= GAME_SIZE) {
        trys--;
        numTouch = 1;
        iaShoot();
    }
    break;
}
} else {
    while (!ok) {
        if (shotBefore(iaShootY, iaShootX)) {
            iaShootX = Math.floor(Math.random() * GAME_SIZE);
            iaShootY = Math.floor(Math.random() * GAME_SIZE);
        } else {
            ok = true;
        }
    }
}
checkIAShoot(iaShootY, iaShootX);
}
function isShipAlive() {
    /*
    Recorro la tabla comparando los names de los tds con el valor del
    name (es global).
    Si no hay coincidencias es que el barco esta hundido.
    */
    for (var i = 0; i < GAME_SIZE; i++) {
        for (var j = 0; j < GAME_SIZE; j++) {
            if (document.getElementById(i + "-" + j).name == auxName) {
                return true;
            }
        }
    }
    return false;
}
function shotBefore(y, x) {
    /*
    Este array guarda los disparos de la IA, asi que lo recorro
    comparandolo con las coordenadas que le paso por parametro.
    Si devuelve true es que el disparo ya esta efectuado.
    Sino lo agrega al array y devuelve false.
    */
    for (var i = 0; i < iaShots.length; i++) {
        if (iaShots[i] == y + "/" + x) {
            return true;
        }
    }
    iaShots.push(y + "/" + x);
    return false;
}
function checkIAShoot(y, x) {
    /*
    Este método se encarga junto con el switch del método iashoot() de
    darle "inteligencia".
    Lo primero es cojer el td del disparo.
    Si tocado es true (de un disparo anterior). Entonces comprueba el
    valor del value. Si es "S" lo pinta de rojo, aumenta el valor de numTouch y gana
    un punto de victoria.
    Si no es "S" lo pinta de azul, reinicia el numTouch y cambia de
    direccion (restando 1 a trys).
    */

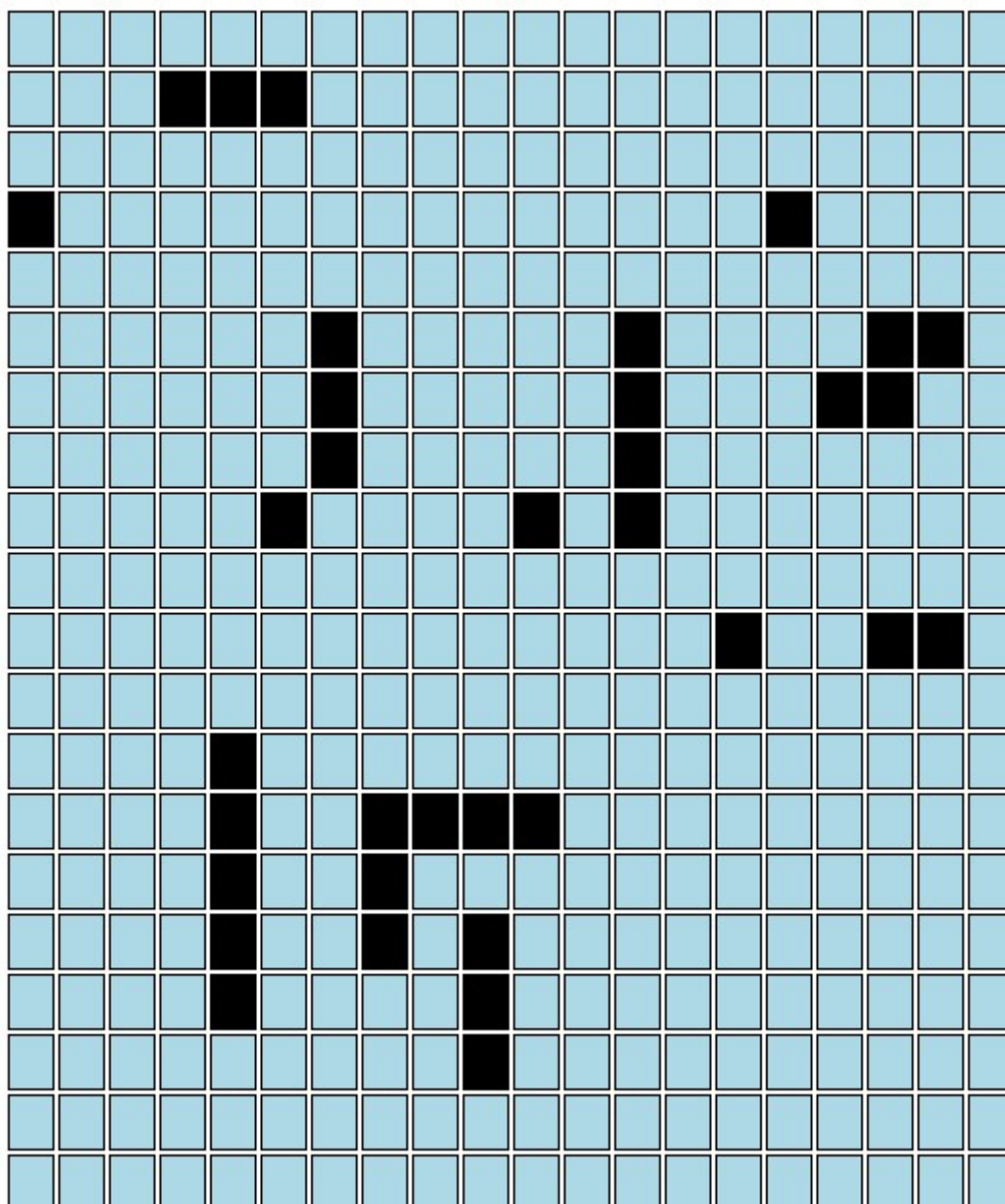
```

Si no esta tocado y falla pues no pasa nada.
Pero si no esta tocado y acierta, incia los aciertos, el numTouch y pone tocado en true, gana un punto de victoria y coge las coordenadas X e Y de la posicion, para poder hacer las comprobaciones hacia los lados.

```

*/
var shotTd = document.getElementById(y + "/" + x);
if (touch) {
    if (shotTd.value == "") {
        numTouch = 1;
        trys--;
    } else if (shotTd.value == "S") {
        shotTd.style.backgroundColor = "red";
        shotTd.value = "";
        numTouch++;
        iaWin++;
        if (iaWin == 35) {
            alert("Has perdido");
            location.reload(true);
        }
    } else {
        shotTd.style.backgroundColor = "blue";
        numTouch = 1;
        trys--;
    }
} else {
    if (shotTd.value == "S") {
        shotTd.style.backgroundColor = "red";
        shotTd.value = "";
        touch = true;
        trys = 4;
        numTouch = 1;
        lastX = x;
        lastY = y;
        iaWin++;
        if (iaWin == 35) {
            alert("Has perdido");
            location.reload(true);
        }
    } else {
        shotTd.style.backgroundColor = "blue";
        numTouch = 0;
        trys = 0;
    }
}
}
}
}

```



JUEGO DEL AHORCADO

Propiedades de String

Length

La clase String sólo tiene una propiedad: `length`, que guarda el número de caracteres del String.

Métodos de String

`charAt(indice)`

Devuelve el carácter que hay en la posición indicada como índice. Las posiciones de un string empiezan en 0.

`indexOf(carácter, desde)`

Devuelve la posición de la primera vez que aparece el carácter indicado por parámetro en un string. Si no encuentra el carácter en el string devuelve -1. El segundo parámetro es opcional y sirve para indicar a partir de que posición se desea que empiece la búsqueda.

`lastIndexOf(carácter, desde)`

Busca la posición de un carácter exactamente igual a como lo hace la función `indexOf` pero desde el final en lugar del principio. El segundo parámetro indica el número de caracteres desde donde se busca, igual que en `indexOf`.

`replace(substring_a_buscar, nuevoStr)`

Implementado en Javascript 1.2, sirve para reemplazar porciones del texto de un string por otro texto, por ejemplo, podríamos utilizarlo para reemplazar todas las apariciones del substring "xxx" por "yyy". El método no reemplaza en el string, sino que devuelve un resultante

de hacer ese reemplazo. Acepta expresiones regulares como substring a buscar.

`split(separador)`

Este método sólo es compatible con javascript 1.1 en adelante. Sirve para crear un vector a partir de un String en el que cada elemento es la parte del String que está separada por el separador indicado por parámetro.

`substring(inicio,fin)`

Devuelve el substring que empieza en el carácter de inicio y termina en el carácter de fin. Si intercambiamos los parámetros de inicio y fin también funciona. Simplemente nos da el substring que hay entre el carácter menor y el mayor.

`toLowerCase()`

Pone todas los caracteres de un string en minúsculas.

`toUpperCase()`

Pone todas los caracteres de un string en mayúsculas.

`toString()`

Este método lo tienen todos los objetos y se usa para convertirlos en cadenas.

`anchor(name)`

Convierte en un ancla (sitio a donde dirigir un enlace) una cadena de caracteres usando como el atributo name de la etiqueta `<A>` lo que recibe por parámetro.

`big()`

Aumenta el tamaño de letra del string. Es como si colocásemos en un string al principio la etiqueta `<BIG>` y al final `</BIG>`.

`blink()`

Para que parpadee el texto del string, es como utilizar la etiqueta `<BLINK>`. Solo vale para Netscape.

`bold()`

Como si utilizásemos la etiqueta ``.

`fixed()`

Para utilizar una fuente monoespaciada, etiqueta `<TT>`.

`fontColor(color)`

Pone la fuente a ese color. Como utilizar la etiqueta ``.

`fontSize(tamaño)`

Pone la fuente al tamaño indicado. Como si utilizásemos la etiqueta `` con el atributo `size`.

`italics()`

Pone la fuente en cursiva. Etiqueta `<I>`.

`link(url)`

Pone el texto como un enlace a la URL indicada. Es como si utilizásemos la etiqueta `<A>` con el atributo `href` indicado como parámetro.

`small()`

Es como utilizar la etiqueta `<SMALL>`

`strike()`

Como utilizar la etiqueta `<STRIKE>`, que sirve para que el texto aparezca tachado.

`sub()`

Actualiza el texto como si se estuviera utilizando la etiqueta `<SUB>`, de subíndice.

`sup()`

Como si utilizásemos la etiqueta `<SUP>`, de superíndice.

Caso Práctico

Presupuesto online

12 teamwebsite.com

ESTIMATED BUDGET

An estimated budget needed to complete the project

ITEM	HOUR	FEE	TOTAL
RESEARCH			
Description of the first item goes here.	7	\$29	\$203
Description of the second item goes here.	2	\$129	\$258
More description goes here.	5	\$5.65	\$28.25
RESEARCH			
Description of the first item goes here.	7	\$29	\$203
Description of the second item goes here.	2	\$129	\$258
More description goes here.	5	\$5.65	\$28.25
RESEARCH			
Description of the first item goes here.	7	\$29	\$203
Description of the second item goes here.	2	\$129	\$258
More description goes here.	5	\$5.65	\$28.25
RESEARCH			
Description of the first item goes here.	7	\$29	\$203
Description of the second item goes here.	2	\$129	\$258
More description goes here.	5	\$5.65	\$28.25
TAOTAL			\$2642.50

Short note about estimated budget here. Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum is simply dummy text of the printing and typesetting industry. Short note about estimated budget here. of the proposal goes here. Lorem Ipsum is simply dummy text of the printing and typesetting industry. .

TEAM NAME
0234 Street, 0917 NewYork, America / Fax: 012 345 6789
Email: email@team.com / Tel: 123-456-7890

Calendario

30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3
4	5	6	7	8	9	10

<input type="text"/>	<input type="text"/>	hola
<input type="text"/>	<input type="text"/>	hola <input type="button" value="calcular"/>

total

```
<form action="">
  <input type="text" id="horas1" />
  <input type="text" id="precio1" />
  <span id="resultado1">hola</span>

  <input type="text" id="horas2" />
  <input type="text" id="precio2" />
  <span id="resultado2">hola</span>

  <input type="button" id="calcular" value="calcular" onclick="calcular()" />
</form>

<p id="total">total</p>

<script>
  window.onload = function () {

    var btnCalcular = document.getElementById("calcular");
    btnCalcular.addEventListener("click", calcular, false);
    var resultado = 0;
    var total;

    function calcular() {
      var horas1 = parseInt(document.getElementById("horas1").value);
      var precio1 = parseInt(document.getElementById("precio1").value);
      var horas2 = parseInt(document.getElementById("horas2").value);
      var precio2 = parseInt(document.getElementById("precio2").value);

      resultado1 = horas1 * precio1;
      resultado2 = horas2 * precio2;
      total = resultado1 + resultado2;

      document.getElementById("resultado1").innerHTML = resultado1;
      document.getElementById("resultado2").innerHTML = resultado2;
      document.getElementById("total").innerHTML = total;
    }

  }
</script>
```

```
var d = new Date();
var d = new Date(milliseconds);
var d = new Date(dateString);
var d
= new Date(year, month, day, hours, minutes, seconds, milliseconds);
```

Date Object Methods

Method	Description
<code>getDate()</code>	Returns the day of the month (from 1-31)
<code>getDay()</code>	Returns the day of the week (from 0-6)
<code>getFullYear()</code>	Returns the year
<code>getHours()</code>	Returns the hour (from 0-23)
<code>getMilliseconds()</code>	Returns the milliseconds (from 0-999)
<code>getMinutes()</code>	Returns the minutes (from 0-59)
<code>getMonth()</code>	Returns the month (from 0-11)
<code>getSeconds()</code>	Returns the seconds (from 0-59)
<code>getTime()</code>	Returns the number of milliseconds since midnight Jan 1 1970, and a specified date
<code>getTimezoneOffset()</code>	Returns the time difference between UTC time and local time, in minutes
<code>getUTCDate()</code>	Returns the day of the month, according to universal time (from 1-31)
<code>getUTCDay()</code>	Returns the day of the week, according to universal time (from 0-6)
<code>getUTCFullYear()</code>	Returns the year, according to universal time

<code>getUTCHours()</code>	Returns the hour, according to universal time (from 0-23)
<code>getUTCMilliseconds()</code>	Returns the milliseconds, according to universal time (from 0-999)
<code>getUTCMinutes()</code>	Returns the minutes, according to universal time (from 0-59)
<code>getUTCMonth()</code>	Returns the month, according to universal time (from 0-11)
<code>getUTCSeconds()</code>	Returns the seconds, according to universal time (from 0-59)
<code>getYear()</code>	Deprecated. Use the <code>getFullYear()</code> method instead
<code>now()</code>	Returns the number of milliseconds since midnight Jan 1, 1970
<code>parse()</code>	Parses a date string and returns the number of milliseconds since January 1, 1970
<code>setDate()</code>	Sets the day of the month of a date object
<code>getFullYear()</code>	Sets the year of a date object
<code>setHours()</code>	Sets the hour of a date object
<code>setMilliseconds()</code>	Sets the milliseconds of a date object
<code>setMinutes()</code>	Set the minutes of a date object
<code>setMonth()</code>	Sets the month of a date object
<code>setSeconds()</code>	Sets the seconds of a date object
<code>setTime()</code>	Sets a date to a specified number of milliseconds after/before January 1, 1970

<code>setUTCDate()</code>	Sets the day of the month of a date object, according to universal time
<code>setUTCFullYear()</code>	Sets the year of a date object, according to universal time
<code>setUTCHours()</code>	Sets the hour of a date object, according to universal time
<code>setUTCMilliseconds()</code>	Sets the milliseconds of a date object, according to universal time
<code>setUTCMinutes()</code>	Set the minutes of a date object, according to universal time
<code>setUTCMonth()</code>	Sets the month of a date object, according to universal time
<code>setUTCSeconds()</code>	Set the seconds of a date object, according to universal time
<code>setYear()</code>	Deprecated. Use the <code>setFullYear()</code> method instead
<code>toString()</code>	Converts the date portion of a Date object into a readable string
<code>toGMTString()</code>	Deprecated. Use the <code>toUTCString()</code> method instead
<code>toISOString()</code>	Returns the date as a string, using the ISO standard
<code>toJSON()</code>	Returns the date as a string, formatted as a JSON date
<code>toLocaleDateString()</code>	Returns the date portion of a Date object as a string, using locale conventions
<code>toLocaleTimeString()</code>	Returns the time portion of a Date object as a string, using locale conventions
<code>toLocaleString()</code>	Converts a Date object to a string, using locale conventions

<code>toString()</code>	Converts a Date object to a string
<code>toTimeString()</code>	Converts the time portion of a Date object to a string
<code>toUTCString()</code>	Converts a Date object to a string, according to universal time
<code>UTC()</code>	Returns the number of milliseconds in a date since midnight of January 1, 1970, according to UTC time
<code>valueOf()</code>	Returns the primitive value of a Date object



SCSS

Variables

Pensamos en las variables como un camino para almacenar valores de estilos que reutilizaremos. Podemos almacenar cualquier valor de css. Sass utiliza el símbolo **\$** para indicar una variable.

```
$anchura:400px;
$altura:200px;
$color_base:#336699;
$grosor_borde: 3px;
$estilo_borde: solid;
$color_borde: #3366ff;

$fuente: Helvetica, sanf-serif;
$color_fuente: #333;

body{
    font: 100% $fuente;
    color: $color_fuente
}

#capa1, #capa2, #capa3{
    width: $anchura;
    height: $altura;
    background: $color_base+#333333;
    border: $grosor_borde $estilo_borde $color_borde;
}
```

Que genera el **css**:

```
body {
    font: 100% Helvetica, sanf-serif;
    color: #333; }
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

```
#capa1, #capa2, #capa3 {
  width: 400px;
  height: 200px;
  background: #6699cc;
  border: 3px solid #3366ff; }

/*# sourceMappingURL=estilos.css.map */
```

Nesting

Para un código **Html**:

```
<nav>
  <ul>
    <li><a href="">Uno</a></li>
    <li><a href="">Dos</a></li>
    <li><a href="">Tres</a></li>
  </ul>
</nav>
```

El Archivo **Sass** con nesting:

```
nav{
  ul{
    margin: 0;
    padding: 0;
    list-style: none;

    li{
      display: inline-block;

      a{
        display: block;
```

```
padding: 6px 12px;
text-decoration: none;
    }
    }
    }
}
```

El archivo **Css** que se genera:

```
nav ul {
  margin: 0;
  padding: 0;
  list-style: none; }

nav ul li {
  display: inline-block; }

nav ul li a {
  display: block;
  padding: 6px 12px;
  text-decoration: none; }

/*# sourceMappingURL=estilos.css.map */
```

Que se mostrará en el navegador:

Uno Dos Tres

Otro ejemplo:

```
#main p{
  color: #00ff00;
  width: 97%;
```

```
.redbox{
    background-color: #ff000;
    color: #000000;
}
}
```

Se compilará a:

```
#main p {
    color: #00ff00;
    width: 97%; }
#main p .redbox {
    background-color: #ff000;
    color: #000000; }
```

Referenciando al selector padre:&

Utilizando el caracter **&** inserta el selector padre, por ejemplo:

```
a{
    font-weight: bold;
    text-decoration: none;
    &:hover{
        text-decoration: underline;
    }
    body.firefox &{
        font-weight: normal;
    }
```

es compilado a:

```
a {
    font-weight: bold;
    text-decoration: none; }
a:hover {
    text-decoration: underline; }
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión


```
body.firefox a {
  font-weight: normal; }
```

& será remplazado con el selector padre que aparece en el Css.

```
#main{
  color: black;
  a{
    font-weight: bold;
    &:hover{color: red;}
  }
}
```

es compilado a:

```
#main {
  color: black; }
#main a {
  font-weight: bold; }
#main a:hover {
  color: red; }
```

Partials

Se pueden crear archivos parciales de Sass que contengan pequeñas secuencias de css que se pueden incluir en otro archivo Sass. Esto facilita el mantenimiento de archivos css de grandes dimensiones separándolos en pequeños archivos sass agrupando bajo cada uno aquellas reglas css que tengan una característica en común.

Se deben de nombrar comenzando con un guión bajo `_partial.scss`. Estos archivos sass así nombrados no se convertirán en css y si serán incluidos en otros archivos scss principales que sí se convertirán en css.

Estos archivos parciales se usan con la directiva `@import`

Por ejemplo si queremos importar un archivo `_reset.scss` dentro del archivo `estilos.scss`. El archivo `_reset.scss`:

```
html, body, div, nav, ul, li, a{
    margin: 0;
    padding: 0;
}
```

En el archivo `estilos.scss`:

```
@import 'reset';
```

Generará en el archivo `estilos.css`:

```
html, body, div, nav, ul, li, a {
    margin: 0;
    padding: 0; }
```

Koala no convierte a css los archivos scss que comienzan por guión bajo `_`

Mixins

Muy práctico en reglas css con varios prefijos de motores de navegadores, por ejemplo con la propiedad `border-radius`.

```
@mixin border-radius($radius){
    -webkit-border-radius: $radius;
    -moz-border-radius: $radius;
    -ms-border-radius: $radius;
    border-radius: $radius;
}

$radio_del_borde:30px;

div{
    @include border-radius($radio_del_borde);
}
```

Que genera la siguiente regla css:

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

```
div {
  -webkit-border-radius: 30px;
  -moz-border-radius: 30px;
  -ms-border-radius: 30px;
  border-radius: 30px; }
```

Que se aplicará en el Html:

127.0.0.1:51463/pasass/pasass1.html

Uno Dos Tres

Capa1

Capa2

Capa3

Extend

`@extend` es una de las más usadas características de Sass. Permite compartir un conjunto de propiedades Css de un selector a otro. Ayuda a hacer el código más **DRY** (don't repeat yourself). Por ejemplo:

```
.message{
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}
.sucess{
  @extend .message;
  border-color: green;
}
.error{
  @extend .message;
  border-color: red;
}
.warning{
  @extend .message;
  border-color: yellow;
}
```

Se convierte en las reglas Css:

```
.message, .sucess, .error, .warning {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333; }

.sucess {
  border-color: green; }

.error {
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

```
border-color: red; }

.warning {
  border-color: yellow; }
```

Operadores

Sass tiene operadores como **+**, **-**, *****, **/**, y **%**. Por ejemplo:

```
.container{
  width: 100%;
}

article[role='main']{
  float: left;
  width: 600px /960px *100%;
}

article[role='complementario']{
  float: right;
  width: 300px /960px *100%;
}
```

Genera las reglas Css:

```
.container {
  width: 100%; }

article[role='main'] {
  float: left;
  width: 62.5%; }
```

Desarrollo y reutilización de componentes de software mediante lenguajes de guión

```
article[role='complementario'] {  
  float: right;  
  width: 31.25%; }
```