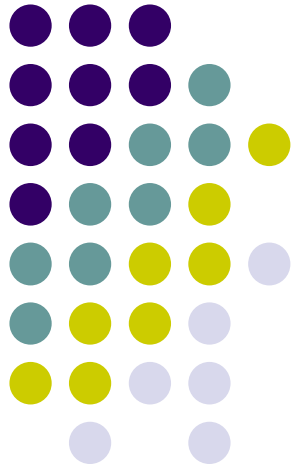
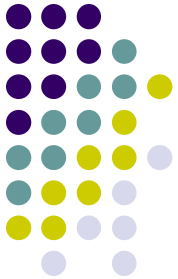
Prototype y Scriptaculous





http://prototypejs.org/learn/

API Reference

The documentation for the latest stable version of Prototype will always be located at http://api.prototypejs.org.

Tutorials

This area contains **narrative documentation** you can use to discover Prototype.

- **Defining classes and inheritance**

  Learn how to define classes and subclasses in Prototype and how to make supercalls.

- **How Prototype extends the DOM**

  Learn how Prototype adds custom methods to DOM element nodes — and how you can define your own custom methods.

- **Introduction to Ajax**

  Learn how Prototype simplifies the most common kinds of Ajax requests.

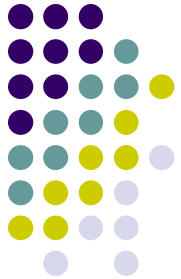
- **Introduction to JSON**

  Learn about Prototype's support for JSON encoding and decoding.

- **Event delegation**

  Learn about Prototype's support for event delegation: an advanced technique for event-driven programming.

Extender JavaScript con prototype



- Limitaciones de JavaScript
  - El mismo código funciona de forma distinta en cada browser

La librería Prototype de JavaScript añade funcionalidades a JavaScript:

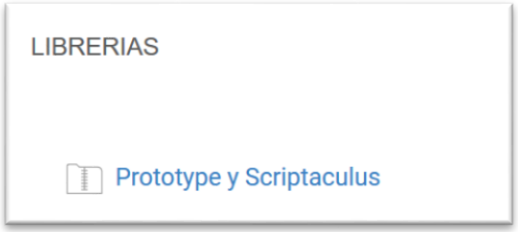
- algunas extensiones al DOM
- añade métodos para tipos pre-definidos String, Array, Date, Number, Object
- Optimiza la compatibilidad entre navegadores
- Permite que Ajax sea mas fácil

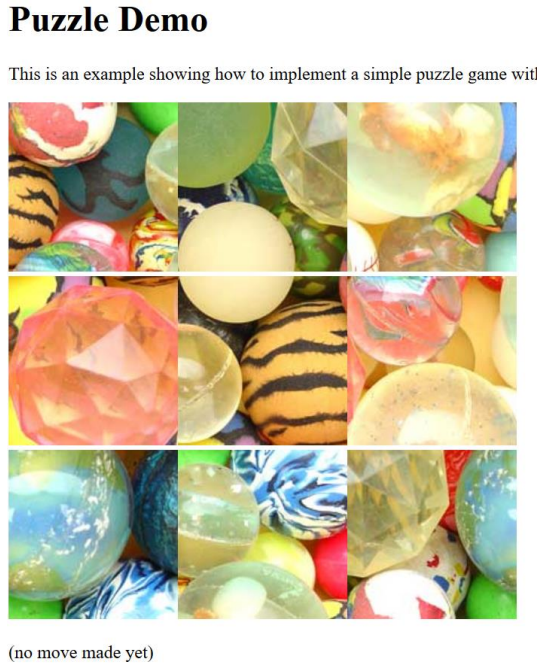
Como usar Prototype y Scriptaculus?

1) Descargar el fichero desde el campus virtual y importando el fichero:

```
<script type="text/javascript" src="lib/prototype.js"></script>
<script type="text/javascript" src="src/scriptaculous.js"></script>
```

Ver ejemplo demo.html en el repertorio





2) Importando la librería via web:

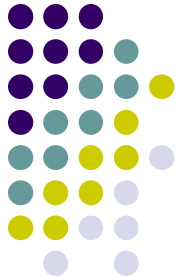
librerías prototype

```
<script src="https://ajax.googleapis.com/ajax/libs/prototype/1.7.0.0/prototype.js"
 type="text/javascript"></script>
```

librerías scriptaculus

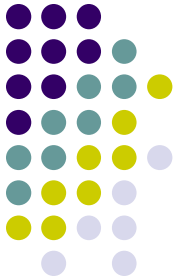
```
<script src="https://ajax.googleapis.com/ajax/libs/scriptaculous/1.9.0/scriptaculous.js">
</script>
```

Lista funcionalidades prototype



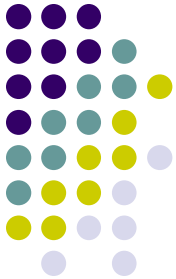
- absolutize
- addClassName
- classNames
- cleanWhitespace
- clonePosition
- cumulativeOffset
- cumulativeScrollOffset
- empty
- extend
- firstDescendant
- getDimensions
- getHeight
- getOffsetParent
- getStyle
- getWidth
- hasClassName
- hide
- identify
- insert
- inspect
- makeClipping
- makePositioned
- match
- positionedOffset
- readAttribute
- recursivelyCollect
- relativize
- remove
- removeClassName
- replace
- scrollTo
- select
- setOpacity
- setStyle
- show
- toggle
- toggleClassName
- undoClipping
- undoPositioned
- update
- viewportOffset
- visible
- wrap
- writeAttribute

Utilidades Scriptaculus



- $
- $$
- $A
- $F
- $H
- $R
- $w
- try.these
- document.getElementsByClassName

Función $



- La función $("*id*") devuelve el objecto asociado al id definido
- Equivalente a document.getElementById("*id*")
- Permite de crear código mas corto ej:

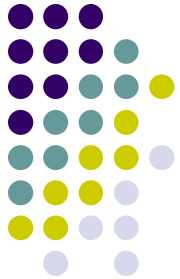
```
$("footer").innerHTML =
   $("username").value.toUpperCase();
```

Ejemplo

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<script language="javascript" type="text/javascript" src="lib/prototype.js"></script>
<script language="javascript" type="text/javascript">
	function alertDivs(){
		var divNodes = $('div1','div2');
		divNodes.each(function(node){
			window.alert(node.innerHTML);
		});
	}
</script>
<title>Dollar Method</title>
</head>

<div id="div1">Hello,</div>
<div id="div2">World!</div>

<a href="javascript:void(0)" onclick="alertDivs();">Alert Divs</a>

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

Si paso multiples IDs recupero un array de objectos de todos los elementos

Función $$

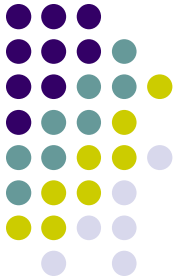


- La función $$() devuelve el elemento (o el array de elementos) asociado a la definición CSS definida.
- Podrían ser:
  - Selectores de tipos (div)
  - Selectores de atributos ([attr],[attr=value],etc…)
  - Selectores de Classes (.classname)
  - Selectores de ID (#div1)

Ejemplo

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<script language="javascript" type="text/javascript" src="lib/prototype.js"></script>
<script language="javascript" type="text/javascript">
        function alertDivs(){
                var divNodes = $$('div.helloworld');
                divNodes.each(function(node){
                                window.alert(node.innerHTML);
                });
        }
</script>
<title>Dollar Method</title>
</head>

<div id="div1" class="helloworld">Hello, World!</div>
<div id="div2">Not me?</div>

<a href="javascript:void(0)" onclick="alertDivs();">Alert Divs</a>

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

Función $A()

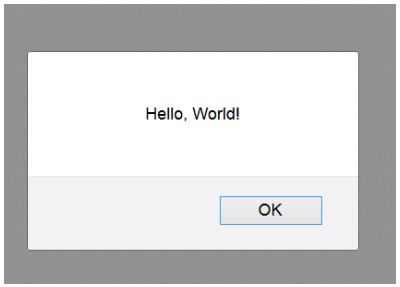


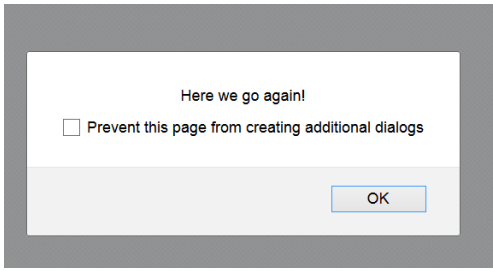
- La función $A() devuelve un array de elementos enumerables.
- Es decir remplaza por ejemplo:
  - document.images
  - document.getElementByTagName()
  - element.childNodes

Ejemplo

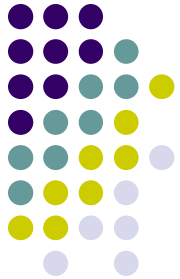
```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<script language="javascript" type="text/javascript" src="lib/prototype.js"></script>
<script language="javascript" type="text/javascript">
        function alertDivs(){
                var divs = document.getElementsByTagName('div');
                var divNodes = $A(div);
                divNodes.each(function(node){
                        window.alert(node.innerHTML);
                });
        }
</script>
<title>Dollar Method</title>
</head>

<div>Hello, World!</div>
<div>Here we go again!</div>

<a href="javascript:void(0)" onclick="alertDivs();">Alert Divs</a>

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

Hello, World!
Here we go again!
Alert Divs





Función $F()



- La función $F() devuelve el valor del controller en un form.
- El valor será un string de todos los controllers, y en el caso de “select box” de un array de valores
- Remplaza:
  - Form.Element.getValue(element)

Ejemplo

$F("*formID*")["*name*"]

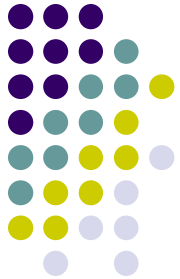
- Recupera los parámetros de un nombre definido y con un id definido

$F("*controlID*")

- La función $F devuelve el valor de un control de form con el id definido

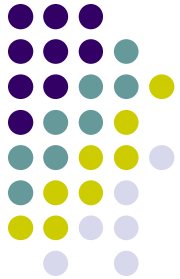
```
if ($F("username").length < 4) {
    $("username").clear();
    $("login").disable();
}
```

Función $w()



- Separa un string en un array, tratando todos los whitespace como separadores

Eventos con prototype

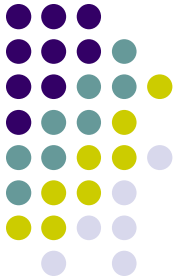


- Añade nuevos metodos para simplificar el uso de eventos:
  - observe() y stopObserving()
  - findElement()

Ejemplo observe

```
window.onload = function() {
   $("textbox").observe("mouseout", boom);
   // esta linea relaciona el evento al texto
   $("submit").observe("click", boom);      //
   esta linea relaciona el evento al botón
};

function boom() {//this se refiere al botón
   this.value = " boom";
}
```

Ejemplo 2



- Una alternativa a “window.onload:”

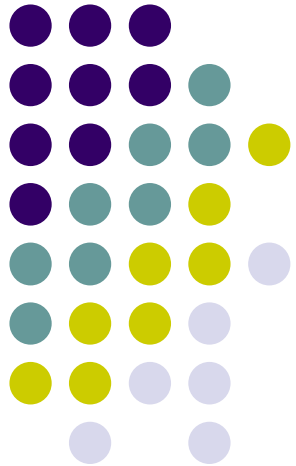
```
document.observe("dom:loaded", function() {
    // event handlers pueden ser juntados aqui,
    etc.
});
```

Ejemplo findElement

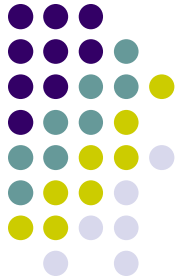
```
<html>
<head>
<title>Prototype examples</title>
   <script type="text/javascript"
   src="/javascript/prototype.js">
   </script>

<script>

Event.observe(document, 'click',
    respondToClick);

function respondToClick(event) {
  var element = Event.findElement(event,
    'P');
  alert("Hiding Tag : " + element.tagName
    );
  if ( element != document ){
    element.hide();
  }
}
</script>
</head>

<body>

   <p id="note"> Click anywhere to see
    the result.</p>

   <p id="para1">This is paragraph 1</p>
   <br />
   <br />
   <p id="para2">This is paragraph 2</p>
   <div id="division">
       This is divsion.
   </div>

</body>
</html>
```

Este ejemplo borra un cualquier paragrafo del DOM sobre el cual se clica

Scriptaculous

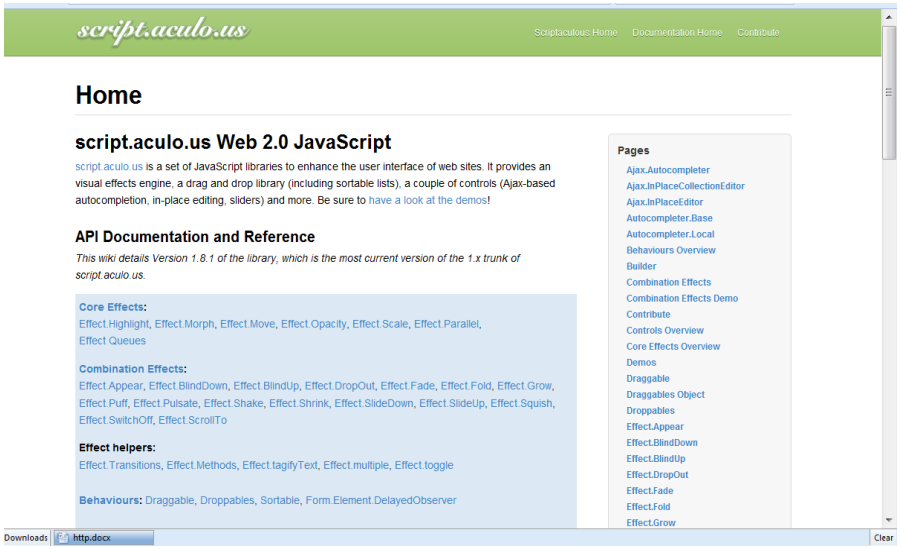


Que es scriptaculous?



- Scriptaculous : una libreria JavaScript, construida sobre Prototype, que añade:
  - Efectos visuales (animaciones, fade in/out, subrayados)
  - drag and drop
  - Características de Ajax:
    - Campos de texto que se Auto-complétanos (listas drop-down)
    - Editores In-place (un texto que se puede clicar y enviar al servidores)
- Algunas mejoras del DOM
- Etc…

- http://madrobby.github.com/scriptaculous/
- http://madrobby.github.com/scriptaculous/combination-effects-demo/



http://madrobby.github.com/scriptaculous/combination-effects-demo/

Como añadir un efecto

```
element.effectName();          // en la mayoria de los casos

new Effect.Name(element or id);   // en algunos casos

$("sidebar").shake();

var buttons = $$("results > button");
for (var i = 0; i < buttons.length; i++) {
    buttons[i].fade();
}
```

- El efecto empezara a animarse en la pantalla (de forma asíncrona) en el momento que será activado

Opciones de efectos

```
element.effectName({
    option: value,
    option: value,
    ...
});

$("my_element").pulsate({
    duration: 2.0,
    pulses: 2
});
```

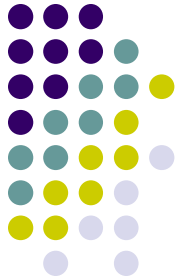
- Muchos efectos pueden ser personalizados pasando opciones adicionales (notar las {})
- Opciones : retraso, dirección, dudada, fps (Frames Per Second), transiciones)

Eventos de un efecto

```
$("my_element").fade({
    duration: 3.0,
    afterFinish: displayMessage
});
function displayMessage(effect) {
    alert(effect.element + " is done fading now!");
}
```

- Todos los efectos tienen los siguientes eventos que se pueden gestionar :
  - beforeStart, beforeUpdate, afterUpdate, afterFinish
- El evento afterFinish se activa cuando el efecto se anima
  - Se puede hacer algo cuando el elemento ha acabado (estilo, remoción)
- Cada uno de estos eventos recibe un objeto Efecto como parámetro
  - Sus propiedades son : element, options, currentFrame, startOn, finishOn
  - Algunos efectos (como Shrink) son técnicamente "effectos paralelos", por lo tanto hay que escribir effect.effects[0].element en lugar que effect.element

Drag and drop



- Scriptaculous proporciona varios objetos que suportan el drag-and-drop:
- **Sortable** **:** una lista de elemento que puede ser re-ordenada
- **Draggable** **:** un elemento que puede ser arrastrado
- Draggables : gestiona todo los objetos “Draggable” en la pagina
- Droppables : elementos hacia los cuales un elemento “Draggable” puede ser arrastrado.

Sortable

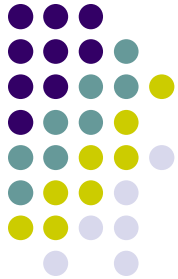
```
Sortable.create(element or id of list, {
    options
});
```

- Especifica una lista y permite de arrastrar cada elemento encima el otro
- opciones: tag, only, overlap, constraint, containment, format, handle, hoverclass, ghosting, dropOnEmpty, scroll, scrollSensitivity, scrollSpeed, tree, treeTag
- Para hacer que la lista no se pueda ordenar usar Sortable.destroy

Ejemplo sortable

```
<ol id="simpsons">
    <li id="simpsons_0">Homer</li>
    <li id="simpsons_1">Marge</li>
    <li id="simpsons_2">Bart</li>
    <li id="simpsons_3">Lisa</li>
    <li id="simpsons_4">Maggie</li>
</ol>

document.observe("dom:loaded", function() {
    Sortable.create("simpsons");
});
```

Eventos sortable



- onChange cuando un elemento de una lista se mueve en una nueva posición mientras que se arrastra
- onUpdate cuando un elemento de la lista se deja en una nueva posición

```
document.observe("dom:loaded", function() {
    Sortable.create("simpsons", {
                onUpdate: listUpdate
    });
});
```

Sortable

```
document.observe("dom:loaded", function() {
    Sortable.create("simpsons", {
                onUpdate: listUpdate
    });
});

function listUpdate(list) {
    // se puede hacer lo que se quiere aquí; efectos,
    una peticion Ajax, etc….
    list.shake();
}
```

Draggable

```
new Draggable(element or id, {
    options
});
```

- Especifica que un elemento se pueda arrastrar
- opciones: handle, revert, snap, zindex, constraint, ghosting, starteffect, reverteffect, endeffect
- Opciones del evento : onStart, onDrag, onEnd
  - Cada función handler acepta dos parámetros : el objeto Draggable, y el evento del mouse

Ejemplo Draggable

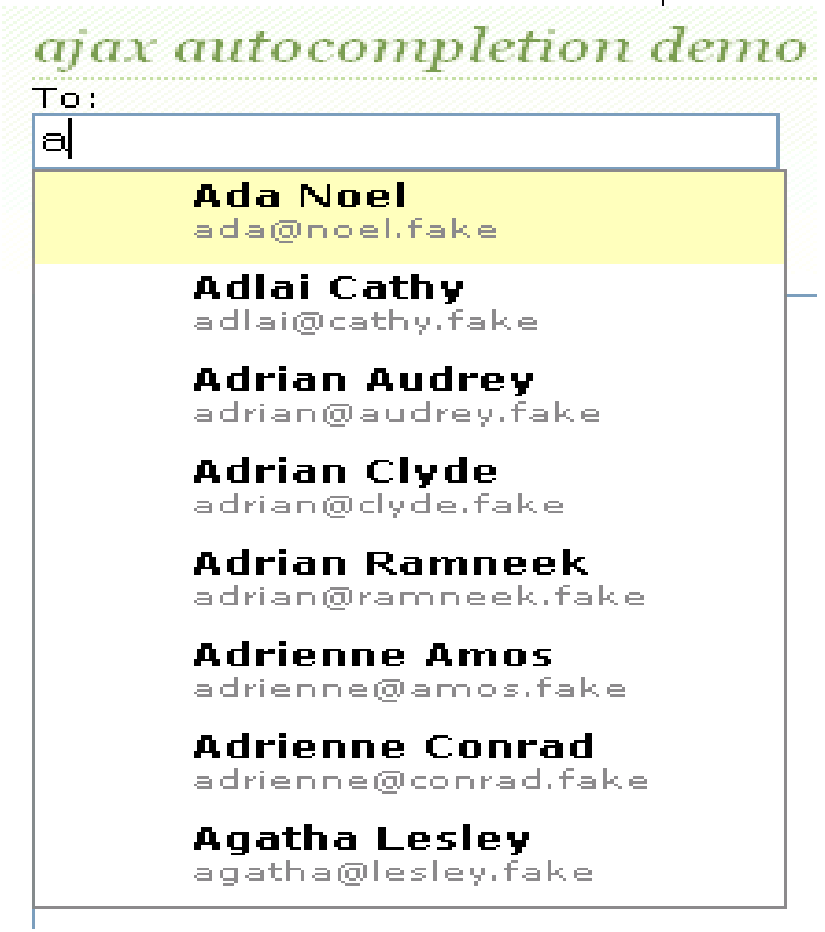
```
<div id="draggabledemo1">Draggable demo 1. Default
    options.</div>
<div id="draggabledemo2">Draggable demo 2.
    {snap: [40,40], revert: true}</div>

document.observe("dom:loaded", function() {
    new Draggable("draggabledemo1");
    new Draggable("draggabledemo2", {revert: true,
    snap: [40, 40]});
});
```

Drag/drop demo shopping

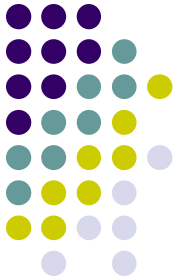
```
<img id="shirt" src="images/shirt.png" alt="shirt" />
<img id="cup" src="images/cup.png" alt="cup" />
<div id="droptarget"></div>

document.observe("dom:loaded", function() {
    new Draggable("shirt");
    new Draggable("cup");
    Droppables.add("droptarget", {onDrop: productDrop});
});

function productDrop(drag, drop, event) {
    alert("You dropped " + drag.id);
}
```

Campos que se completan Automaticamente

- Scriptaculous añade varias herramientas para hacer que un text box se auto-complete :
- Autocompleter.Local : se auto-completa a partir de un array de opciones
- Ajax.Autocompleter : analiza y visualiza una serie de opciones usando Ajax



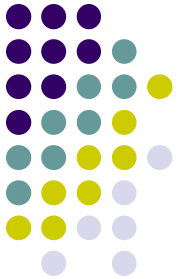
Uso de Autocompleter.Local



```
new Autocompleter.Local(
    element or id of text box,
    element or id of div to show completions,
    array of choices,
    { options }
);
```

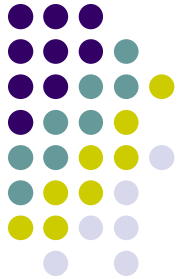
- Se tiene que crear un div inicialmente vacío para grabar las entradas del texto que se auto-completa
  - El usuario podrá seleccionar el resultado presionando las flechas Arriba/Abajo;
- Se pasan las propuestas con un array de string
- Se pasan las opciones adicionales a por medio de un cuarto parámetro entre { }
  - Opciones :choices, partialSearch, fullSearch, partialChars, ignoreCase

Ejemplo Autocompleter.Local



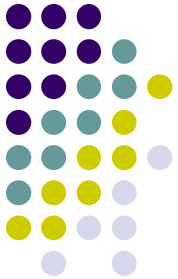
```
<input id="bands70s" size="40" type="text" />
<div id="bandlistarea"></div>

document.observe("dom:loaded", function() {
    new Autocompleter.Local(
            "bands70s",
            "bandlistarea",
            ["ABBA", "AC/DC", "Aerosmith", "America", "Bay City
    Rollers", ...],
            {}
    );
});
```

Estilos de Autocompleter.Local



```
<input id="bands70s" size="40" type="text" />
<div id="bandlistarea"></div>

#bandlistarea {
    border: 2px solid gray;
}
/* 'selected' class is given to the autocomplete item
    currently chosen */
#bandlistarea .selected {
    background-color: pink;
}
```

Sonidos



1. Sound.play("*url*");
2. Sound.disable();
3. Sound.enable();
4. Sound.play("music/java_rap.mp3");
5. Sound.play("music/wazzaaaaaap.wav");

1. Ejecuta un fichero de musica remoto
2. Desabilita todos los sonidos futuros (no para el sonido que se esta ejecutando)
3. re-habilita los sonidos despues de activar Sound.disable()
4. Ejecuta un mp3
5. Ejecuta un wav

•Para silenciar un sonido que se esta ejecutando usare Sound.play('', {replace: true});
•No se pueden sonar ficheros desde un ordenador local (tiene que ser cargado desde un sitio)
•Usa el plugin que el browser tiene instalado (ej. Quicktime)