



# Biblioteca de animación jQuery.sprately

Por [Jaime Peña Tresancos](#) [Seguir a @jpt219](#) 04 de junio de 2013  1 [Comentarios](#)  [CSS](#), [Frameworks Javascript](#)

## Cómo podemos animar sprites mediante JavaScript...y un mínimo de CSS.

En nuestro anterior artículo [Animaciones de sprites mediante CSS3 y JavaScript](#): habíamos comentado las posibilidades de crear animaciones de sprites fundamentalmente con estilos CSS3 y algo de JavaScript. Aquí lo veremos desde una nueva perspectiva –justamente la inversa-, con algunas nuevas posibilidades que por el momento no son tan fáciles –o no lo son en absoluto- de implementar con estilos.



Estas dos visiones nos ofrecerán alternativas válidas en muchos casos y en otros unas ofrecen ventajas sobre las otras, que cada programador deberá ponderar, fundamentalmente en lo referente al ámbito de las propias capacidades de animación, la optimización del código y el peso de nuestras páginas.

En el presente artículo hablaremos de:

- Animación simple de sprites
- Animación con movimientos aleatorios del sprite
- Control de la animación
- Respuesta del sprite ante clics sobre el cuerpo del documento

Comentaremos la posibilidad de animar sprites mediante JavaScript y un mínimo de CSS. Se trata de la biblioteca jQuery.sprately que facilita la animación de sprites; está basada en jQuery y tiene diversas opciones avanzadas, dentro de su sencillez de uso. El lugar de Internet de referencia y descarga es: [www.sprately.net](http://www.sprately.net)



## Funcionalidad básica de animación

Se basa en el uso del método `sprite()`, que se asocia a un elemento `div` en el cuerpo del documento HTML, donde será mostrado el sprite.

Hemos de dotar de un cierto estilo al elemento `div` para que cumpla las condiciones de contenedor del sprite y eso es todo.

Veamos el Listado 1 que pasaremos a comentar.

**Listado 1:** Animación simple de un sprite mediante el método `sprite()`

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" lang="es-es">
<title>Animated Sprite Sheet (Spritely-jQuery)</title>

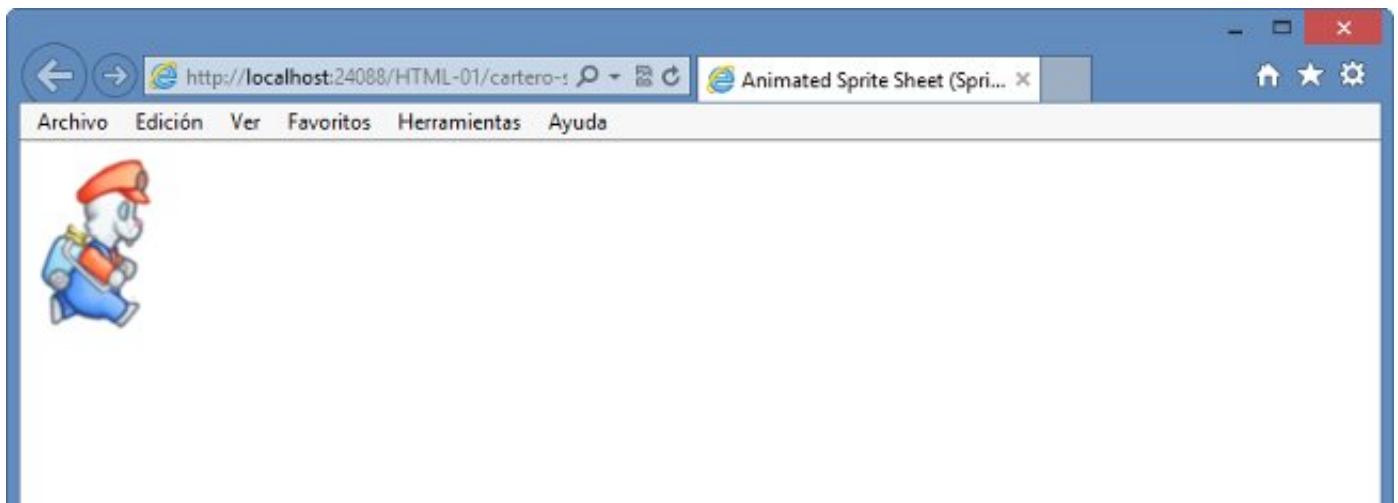
<script src="scripts/jquery-1.8.3.min.js"></script>
<script src="scripts/jquery.spritely-0.6.1.js"></script>

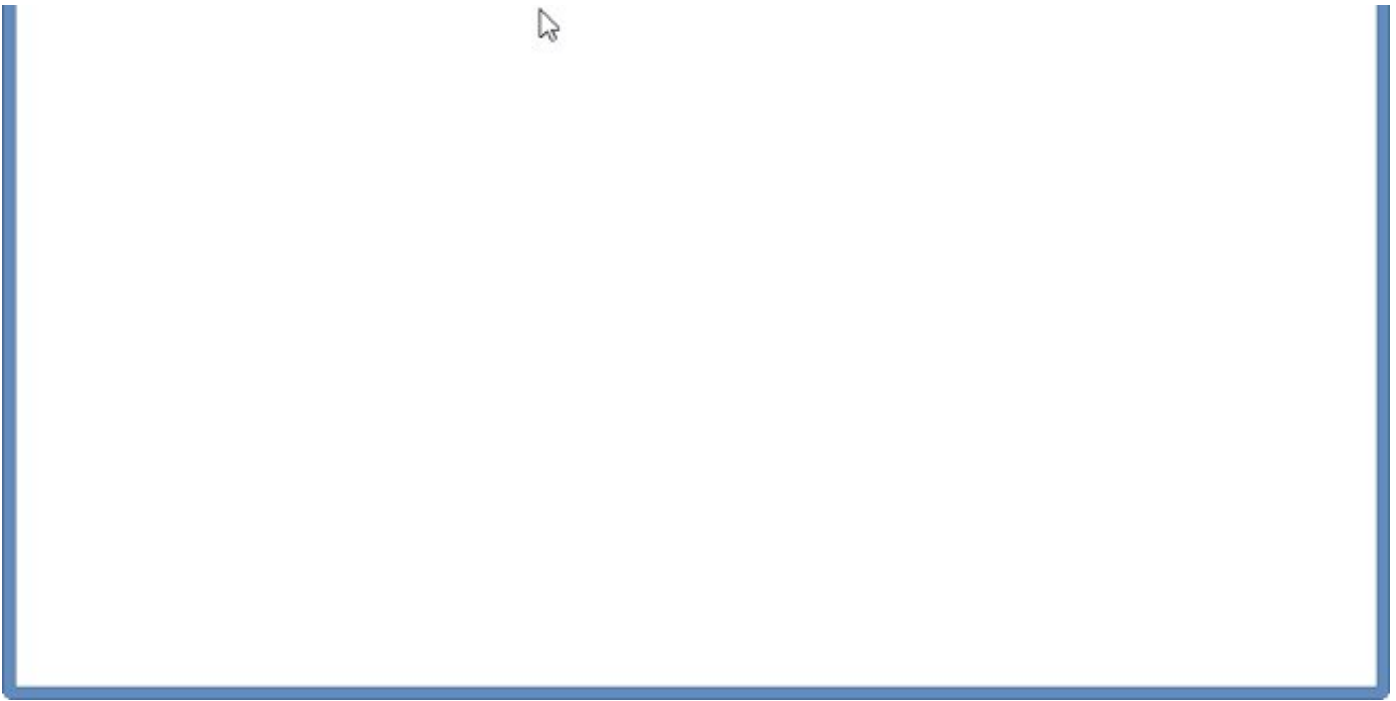
<style>
#cartero {
background: transparent url(cartero.png) 0 0 no-repeat;
position: absolute;
width: 70px;
height: 112px;
}
</style>

<script>

$(function () {
$('#cartero')
.sprite({ fps: 24, no_of_frames: 8 })
});

</script>
</head>
<body>
<div id="cartero"></div>
</body>
</html>
```





Posición del sprite animado en la posición fijada por el elemento div

Las características del sprite utilizado son:

- Número de filas con imágenes: 1
- Número de imágenes: 8
- Ancho de cada imagen: 72px
- Alto de las imágenes: 112px

Respecto al estilo del elemento div:

- Se carga la imagen como fondo transparente en la cota (0, 0), sin repetición, es decir, se cargará una única copia de la imagen
- La propiedad position se fija como absolute
- Se le dan unas dimensiones de ancho y alto que coincidan exactamente con el ancho y alto de un fotograma del sprite
- Con ello, lo que hemos creado realmente es una especie de ventana de visión de fotogramas y con el método de JavaScript –de manera transparente para nosotros- moveremos la imagen de forma que se vayan viendo las sucesivas imágenes de los fotogramas, dando la sensación de movimiento del sprite

Respecto al método sprite():

- Creamos la función que asocia el método al elemento div que contendrá el sprite
- Se da una velocidad de pase de fotogramas por segundo, en nuestro caso 24: fps: 24
- Se fija el número de fotogramas que contiene el sprite: no\_of\_frames: 8
- Ahora la función tomará las características del sprite asociado a la ventana –datos de dimensiones-, dividirá convenientemente la imagen y mostrará en el elemento div cada uno de los fotogramas sucesivamente a la velocidad especificada

# Animación aleatoria

Además de crear la animación por la presentación sucesiva de los fotogramas contenidos en el sprite, también es posible hacer que e la vez el conjunto se desplace por un área de la ventana del navegador –a la que llamaremos escenario- y que lo haga de forma aleatoria, para ello bastará llamar al método `spRandom()`.

El método `spRandom()` toma como parámetros:

- **top**: cota superior del escenario
- **bottom**: cota inferior del escenario
- **left**: borde izquierdo del escenario
- **right**: borde derecho del escenario
- **speed**: velocidad –en ms- de cada ciclo de traslación por el escenario. Opcional
- **pause**: tiempo de pausa después de cada ciclo de traslación –en ms-. Opcional

El sprite se creará en el área del elemento `div` asociado, como en el ejemplo anterior, pero inmediatamente comenzará a navegar de forma aleatoria por el escenario marcado por las cotas dadas en el método `spRandom()`.

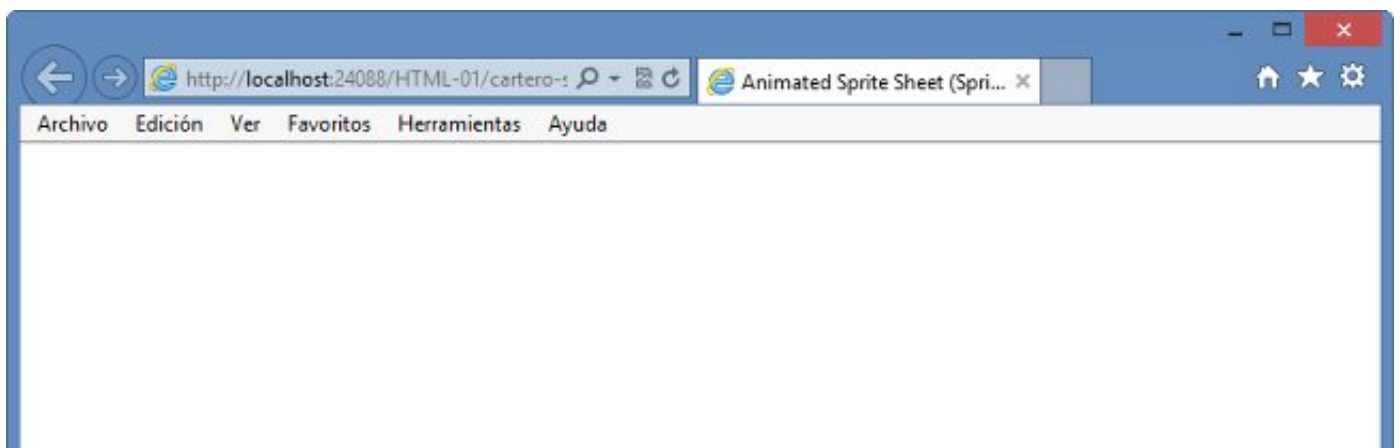
En el Listado 2 se muestra una modificación del Listado 1, en el que el sprite se moverá aleatoriamente por un escenario, en vez de permanecer estático en su posición inicial.

## Listado 2: Modo de uso del método de animación aleatoria `spRandom()`

```
<script>

$(function () {
  $('#cartero')
    .sprite({ fps: 24, no_of_frames: 8 })
    .spRandom({ top: 50, bottom: 400, left: 50, right: 520, speed: 4000, pause: 3000 })
});

</script>
```





El sprite, inicialmente en la posición que muestra la figura anterior, se desplaza aleatoriamente por un escenario prefijado

## Controles de animación

Disponemos de tres métodos predefinidos:

- **spStop()**: detiene la animación
- **spStart()**: reanuda la animación
- **spToggle()**: invierte el estado de animación

En el Listado 3 hemos programado tres botones de comando para controlar la animación del sprite, con la llamada a estos tres métodos.

**Listado 3:** Uso de botones de comando para el control de la animación

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" lang="es-es">
<title>Animated Sprite Sheet (Spritely-jQuery)</title>

<script src="scripts/jquery-1.8.3.min.js"></script>
<script src="scripts/jquery.spritely-0.6.1.js"></script>

<style>
#cartero {
background: transparent url(cartero.png) 0 0 no-repeat;
position: absolute;
```

```

top: 100px;
left: 200px;
width: 70px;
height: 112px;
}
</style>

<script>

$(function () {
$('#cartero')
.sprite({ fps: 24, no_of_frames: 8 })
});

</script>
</head>
<body>
<div id="escenario">
<div id="cartero"></div>
<div>
<button onclick="$('#cartero').spStop();">.spStop()</button>
<button onclick="$('#cartero').spStart();">.spStart()</button>
<button onclick="$('#cartero').spToggle();">.spToggle()</button>
</div>
</div>
<p>Spritely</p>
</body>
</html>

```

## Control de posición con clic del ratón

La biblioteca dispone de un método muy avanzado para hacer responder al sprite a los clics del ratón sobre el cuerpo del documento, de manera que se moverá desde su posición actual, hasta el sitio en que ha tenido lugar. Se trata de un método asociado al cuerpo del documento y que no toma ningún tipo de parámetro –flyToTap()-.

El método flyToTap() se llama con la sintaxis:

```

$('body').flyToTap();

```

En el Listado 4 se muestra un ejemplo de uso, que comentaremos posteriormente.

**Listado 4:** Ejemplo de control de posición con clics del ratón sobre el cuerpo del documento

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" lang="es-es">
<title>Animated Sprite Sheet (Spritely-jQuery)</title>

<script src="scripts/jquery-1.8.3.min.js"></script>
<script src="scripts/jquery.spritely-0.6.1.js"></script>

<style>
#cartero {
background: transparent url(cartero.png) 0 0 no-repeat;
position: absolute;
top: 100px;
left: 200px;
width: 70px;
height: 112px;
}

#escenario {
top: 0px;
left: 0px;
width: 800px;
height: 800px;
}
</style>

<script>

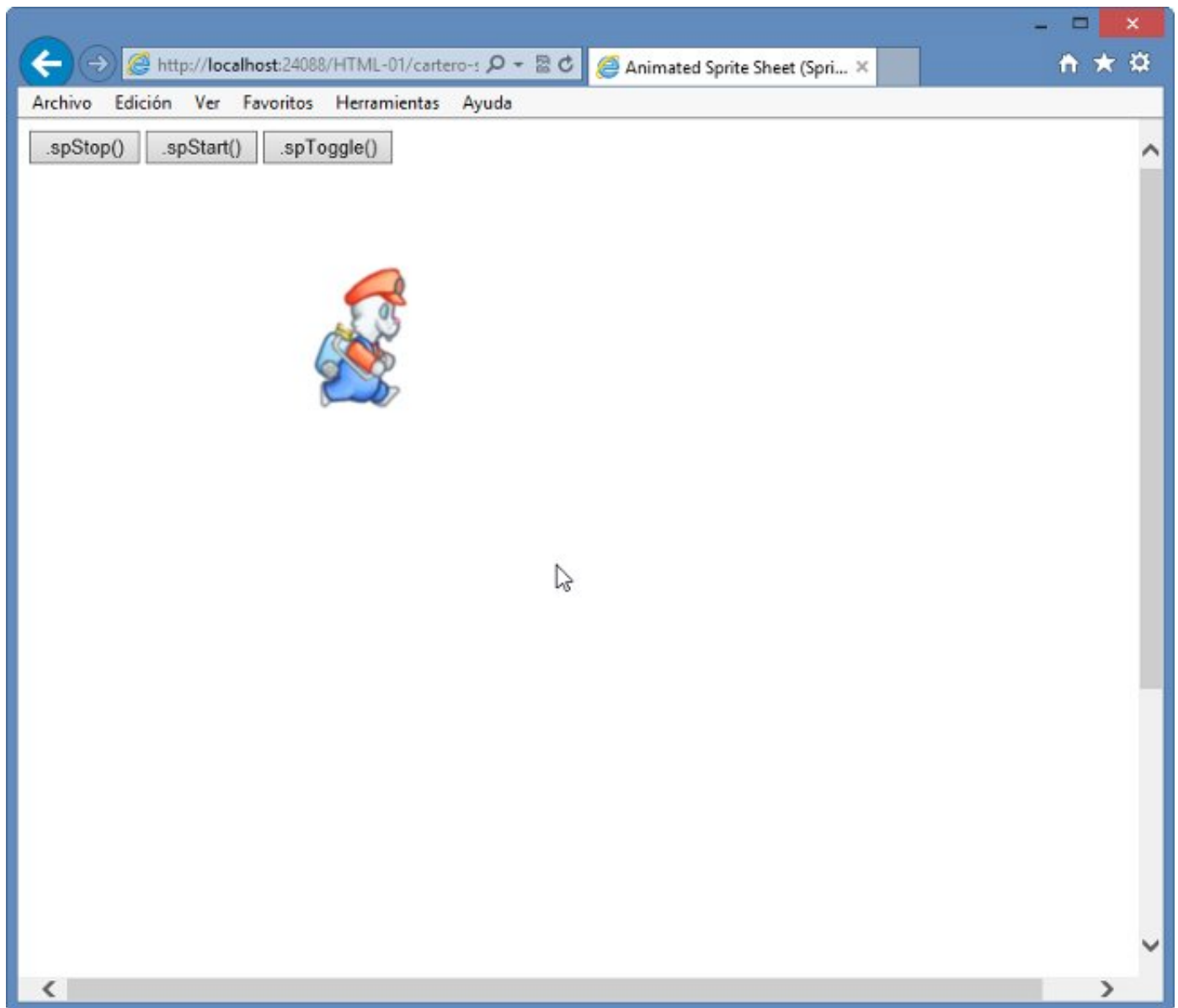
$(function () {
$('#cartero')
.sprite({ fps: 24, no_of_frames: 8 }).activeOnClick().active();
});

$(function () {
$('body').flyToTap();
});

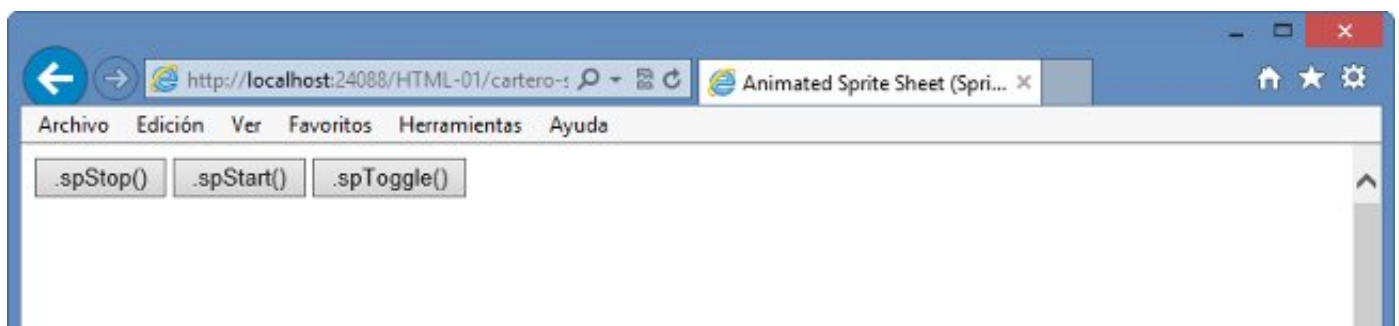
</script>
</head>
<body>
<div id="escenario">
<div id="cartero"></div>
<div>
```

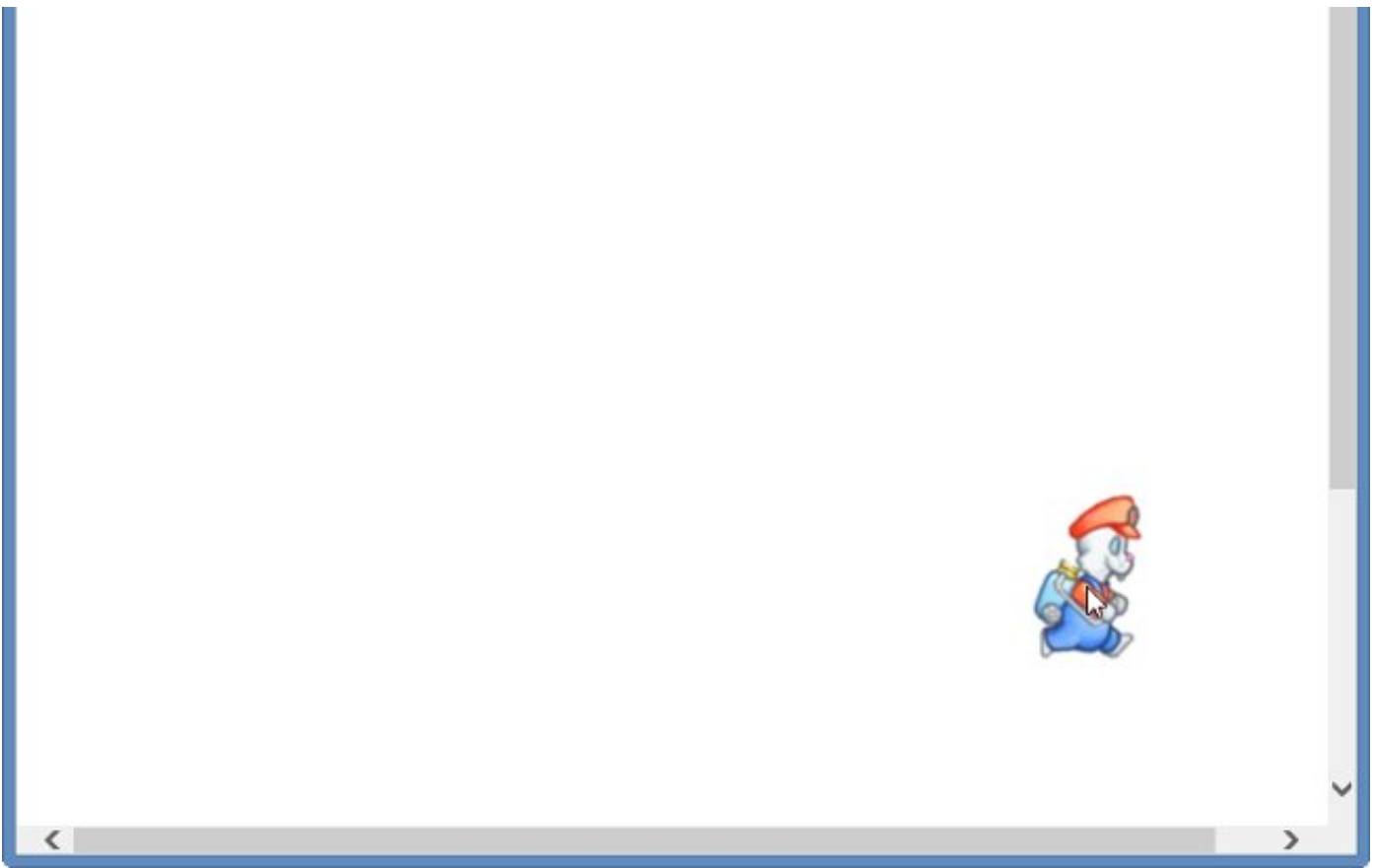


```
<button onclick="$('#cartero').spStop();">.spStop()</button>
<button onclick="$('#cartero').spStart();">.spStart()</button>
<button onclick="$('#cartero').spToggle();">.spToggle()</button>
</div>
</div>
<p>Spritely</p>
</body>
</html>
```



Posición inicial del sprite





Posición del sprite al hacer clic sobre un punto del cuerpo del documento

El código seguido en el ejemplo ha sido:

- Definir dos elementos div:
  - Uno para contener el sprite original, como en los anteriores ejemplos
  - El segundo, a falta de más elementos en la página, para que extienda el área de posibilidad de actuación de los clics del ratón
- La definición de la función `sprite()`, que ya hemos visto en los restantes ejemplos, con las llamadas a los métodos adicionales:
  - `active()`: que indica que será el sprite activado
  - `activateOnClick()`: que lo activa de facto al hacer clic sobre el escenario
- La inclusión de la nueva función `flyToTap()`, que se activará al hacer clic sobre cualquier área del documento HTML
- Los restantes elementos mantenidos son los botones de comando de control de animación, ya comentados en el apartado anterior

Un aspecto interesante del método `flyToTap()` es que de haber un método `spRandom()` implementado –véase el apartado **Animación aleatoria**–, el sprite permanecería unos segundos en el lugar en que se ha hecho clic y posteriormente volvería al lugar de animación aleatoria de forma automática.

## Conclusiones

Hemos comprobado como el uso de unas buenas bibliotecas de JavaScript pueden simplificar enormemente las tareas de cualquier programador; véase la animación de sprites, llevándonos a niveles de sofisticación relativamente elevados sin esfuerzo alguno. El tema controvertido, lo hemos apuntado al comienzo, si se trata simplemente de realizar una animación, el empleo de CSS3 será sin duda alguna ventajoso, pero si se desea una amplia compatibilidad y funciones avanzadas, bibliotecas como jQuery.sprytely serán indudablemente una solución a tomar muy en cuenta.

Esperamos que todo lo expuesto les haya servido de ayuda. Hasta nuestro próximo artículo, felices horas de programación.


## Autor

Jaime Peña Tresancos

 Seguir a @jpt219

Escritor. Colaborador habitual de revistas de tecnología y experto en nuevas tecnologías y programas Microsoft



Subir 

## Manual

Taller de CSS

← Animaciones de sprites mediante CSS3 y JavaScript

Cómo hacer con CSS3 un bocadillo de cómic o globo de historieta →

## Compartir

4

 Compartir

9

 G+1

 84

 Recomendar


71

 Tweet



 Compartir

# Comentarios

 [Enviar un comentario al artículo](#)

Jums

## Sugerencia

11/6/2013

Interesante post.

Deberíais aprovechar algún servicio que ponga en práctica vuestros ejemplos, por ejemplo, jsfiddle.net, jsbin.com, etc...

Le daría un valor adicional a vuestros posts.

Gracias por el esfuerzo.

[Marcar como spam](#)

 [Enviar un comentario al artículo](#)

## Principales

[Manuales](#)

[FAQs](#)

[En directo](#)

[Vídeos](#)

## Monotemáticos

[Desde cero](#)

[HTML](#), [CSS](#)

[Javascript](#), [Ajax](#)

[Diseño](#), [ASP](#)

## Blogging

[Actualidad](#)

[De interés](#)

[Agenda](#)



Powered by:

