**INTRODUCCIÓN**

# **Vínculos a archivos JS**

El atributo **src** admite tres tipos de ruta:

* **Absoluta**: cuando nos dirigimos a un archivo que no se encuentra en el mismo servidor que la página web.
* **Relativa a archivo**: cuando nos dirigimos a un archivo alojado en nuestro sitio web.
* **Relativa a documento**: se utilizan en los sitios web locales, es decir, en aquellos que no están publicados en un servidor web.

En nuestros ejemplos utilizaremos rutas relativas al documento. De este modo, podremos trabajar con los archivos locales.

# **Pruebas en los navegadores - Depuración**

**Chrome**: Herramientas para desarrolladores (Ctl+Mayus+I)

**Internet Explorer**: F12

**FireFox**: (Ctl+Mayus+K)

# **Principios básicos de jQuery**

jQuery es una librería.

Proporciona soluciones sencillas para gestiones habituales.

El sitio web de jQuery es [www.jquery.com](http://www.jquery.com).

JavaScript tiene otras librerías, pero jQuery aporta bastantes ventajas:

* Un tamaño pequeño.
* Soluciones sencillas para los diseñadores web. Aprovecha los conocimientos de CSS.
* Está testeada y funciona. Se usa en millones de sitios web (Amazon, Microsoft,etc).
* Es gratis.
* La comunidad de desarrolladores es muy amplia.
* jQuery permite que otros programadores generen pluggins (complementos). Podemos tener pluggins para validar formularios, añadir desplegables, generar diapositivas interactivas.
* …

jQuery no es la única librería de JavaScript. Existen librerías como mootolls.net para desarrollar potentes efectos visuales, angularjs.org que permite mejorar la creación y mantenimiento de las páginas, etc.

# **Acceso a la librería jQuery**

El código de la librería jQuery se encuentra en un archivo externo. El acceso a dicho archivo se puede llevar a cabo de varias formas:

* **Método CDN** (distribución de contenido): se trata de un sitio web que tiene el archivo y lo envía a cualquiera que lo solicite.

**Ventajas**:

* Ahorramos a nuestro servidor tiempo de acceso, ya que google, Microsoft o jQuery se encargarán de entregar el archivo a los visitantes de nuestra página web.
* Estas redes cuentan con la ventaja de tener servidores por todo el mundo, por lo que pueden recibir el archivo desde un servidor próximo a su domicilio.
* Como muchos diseñadores usan este método, es muy posible que nuestros visitantes ya tengan el archivo jQuery en la caché del navegador.

**Desventajas**:

* Los visitantes deben estar conectados a Internet para que funcione. Esto es un problema si deseamos que nuestro sitio funcione offline.
* **Añadir la librería desde jQuery.com a nuestro sitio:** nos aseguramos de que nuestro sitio web sigue funcionando cuando los servidores CDN dejen de hacerlo (poco probable. Además, nos permitiría trabajar offline.

# **Enlace a la librería jQuery publicada en un servidor CDN**

Las tres organizaciones anteriores permiten que incluyamos en nuestras páginas los archivos jQuery utilizando sus servidores.

* Microsoft:

<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.js">

</script>

* JQuery

<script src=”<http://code.jquery.com/jquery-2.2.4.min.js>”>

</script>

* Google

<script src=”[http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js](http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js%20) “>

</script>

**Notas**:

* Las versiones min de los archivos son más rápidas de cargar, pero más difíciles de leer.
* A la hora de descargar jQuery debemos fijarnos en el número de versión. El primer número indica un **cambio importante**, que puede afectar a la compatibilidad con las versiones de ciertos navegadores.
* El siguiente número indica la **distribución**. Generalmente ofrecen nuevas funciones, sobrescriben las antiguas para que funcionen mejor, etc.
* El último número hace referencia a la **corrección de errores** de alguna versión.

Para conocer qué ha cambiado de una versión a otra, merece la pena visitar la página de descargas <http://jquery.com/download>.

# **Adición de contenidos a una página**

Si añadiésemos la red de distribución de contenidos (CDN) de Google, podemos olvidarnos del **número de versión**. Si tenemos la versión 1.11 en vez de la 11.1.0, google cargará la última versión de la familia 1.11. esto es bueno ya que los cambios de revisión son mejoras que no afectan a la funcionalidad del sitio.

Si decidimos descargar el archivo en nuestro sitio, normalmente solemos alojarlo en una carpeta denominada **js**.

Una vez alojado el archivo en la carpeta, podemos empezar a añadir código que aproveche las funcionalidades de jQuery. El siguiente paso es añadir un segundo grupo de etiquetas <script>.

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js ">

</script>

<script>

$(document).ready(function(){

// código

});

</script>

**Ready** es una función nativa de jQuery que espera hasta que se cargue la página del navegador para poder ejecutar el código incluido dentro de ella.

¿Cuál es la razón de hacerlo así? Buena parte de la programación de JavaScript consiste en manipular los contenidos de la página web, como por ejemplo animar un elemento de la página, difuminar una imagen, desplegar un menú, etc.

Para hacerlo, JavaScript debe seleccionar los elementos, pero no podrá hacerlo hasta que la página esté totalmente cargada. Esto es lo que hace esta función.

**El orden** debe ser el siguiente:

* En primer lugar, el enlace al archivo jQuery.
* En segundo lugar, incluiremos el código de JavaScript después de los CSS.
* Añadiremos siempre un comentario justo detrás de cada conjunto:

$(document).ready(function(){

// código

}); // **fin de ready**

**Método abreviado** para escribir $(document).ready(function() {}

$(function){

// código

}); // fin de la función