# ARQUITECTURAS WEB

Las páginas web que puedes visitar en Internet no están todas realizadas siguiendo una misma arquitectura. Algunas de ellas no cambiarán sus contenidos ni su aspecto hasta que su diseñador realice esos cambios. En cambio, otras páginas web pueden ir variando en función de nuestra interacción con esas páginas o en función de otros aspectos externos como el paso del tiempo.

Por ello, podemos hacer una diferenciación de las diferentes arquitecturas web en función de la manera en que puedan variar sus contenidos.

## Páginas web estáticas

Es el primer tipo de páginas web que se crearon desde el origen de la web (~1991) y del que todavía podemos encontrar muchos casos en Internet.

Se basa en el **lenguaje *HTML***, el cual es interpretado por los navegadores web para mostrar la información al usuario.

Por ejemplo, el siguiente trozo de código *HTML* es el contenido de una página web:

```
<HTML>
 <HEAD>
  <TITLE>Un Titulo para el Browser de turno </TITLE>
 </HEAD>
 <BODY>
  <!-- Aqui va todo lo chachi -->
  <H1>Otro t&iacute;tulo, esta vez m&aacute;s largo. </H1>
  <P> <IMG SRC="./felix.gif" ALIGN= "MIDDLE" ALT= "EL Gato Felix">Hoola.
  <P>Esto es un parrafo con informacion
  super importante. Notese que las lineas salen pegadas aun dejando
  espacios, saltos de linea, etc. <BR> &#161 Si pongo esto
  si <STRONG>cambia </STRONG> de linea!</P>
  <P>Otro parrafo, esto ya es un poco rollo.</P>
  <H3>Pongamos un subtítulo</H3>
  <P>Por cierto, &#191 que paso con las <A HREF= "#pepe">anclas</A>?</P>
  <HR>
  <UL>
  <LI> Esto es una lista no ordenada.
  <LI> Las listas quedan mejor si tienen varios elementos.
  </UL>
  Me voy al <A HREF="http://www.iac.es/home.html">IAC</A>.
  <P>Vamos a crear un <EM>ancla </EM>, o lo que es lo mismo, un <A NAME="pepe">ancho
r.</A></P>
 </BODY>
</HTML>
```

Este código será **mostrado al usuario**, a través del **navegador web**, de la siguiente manera:


Las páginas web creadas **exclusivamente con lenguaje *HTML* no variarán** en su forma ni en su contenido mientras su desarrollador no modifique el código de la página.

Con la utilización de **otros lenguajes y tecnologías**, se pueden obtener páginas web con **contenido dinámico**, pero en cualquier caso siempre estarán integradas dentro de una **estructura básica de *HTML***. Es decir, ningún lenguaje de creación de páginas web dinámicas puede existir de forma independiente. Siempre se deben integrar de una manera u otra junto con el lenguaje HTML.

### CSS

El lenguaje de marcas CSS se puede utilizar junto con el lenguaje *HTML* para facilitar el diseño del aspecto visual que ofrecen las páginas web, aunque su uso por sí solo no permitirá crear dinamismo en las páginas web.

## Aplicaciones web

Algunos lenguajes de programación permiten integrar código junto con el lenguaje *HTML* para ofrecer **contenidos dinámicos** en las páginas web. De esta manera la web crece enormemente en posibilidades y da origen a las **aplicaciones web**. Así, los contenidos ofrecidos por las páginas web no son sólo un conjunto de información que se ofrece de forma estática, aunque

permiten la navegación entre sus contenidos, sino que se convierten en auténticas **aplicaciones informáticas** que pueden ser utilizadas desde el **navegador web**.

Se le da el nombre de *Rich Internet Application* (o Aplicación de Internet Enriquecida) a las aplicaciones web que tienen muchas de las características de las aplicaciones software de escritorio. Normalmente se ofrecen a través de un navegador específico, un complemento para los navegadores habituales, haciendo uso de *JavaScript* o mediante una máquina virtual. ~~*Adobe Flash*~~*, JavaFX, y Microsoft Silverlight* son actualmente las plataformas más comunes para ofrecer *RIA*.

## Ventajas

- **Ahorra tiempo**: Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.
- No hay problemas de **compatibilidad**: Basta tener un navegador actualizado para poder utilizarlas.
- **No ocupan espacio** en nuestro disco duro.
- **Actualizaciones** inmediatas: Como el software lo gestiona el propio desarrollador, cuando nos conectamos estamos usando siempre la última versión que haya lanzado.
- **Consumo de recursos bajo**: Dado que toda (o gran parte) de la aplicación no se encuentra en nuestro ordenador, muchas de las tareas que realiza el software no consumen recursos nuestros porque se realizan desde otro ordenador.
- **Multiplataforma**: Se pueden usar desde cualquier sistema operativo porque sólo es necesario tener un navegador.
- **Portables**: Es independiente del ordenador donde se utilice (un PC de sobremesa, un portátil...) porque se accede a través de una página web (sólo es necesario disponer de acceso a Internet). La reciente tendencia al acceso a las aplicaciones web a través de teléfonos móviles requiere sin embargo un diseño específico de los ficheros CSS para no dificultar el acceso de estos usuarios.
- La **disponibilidad** suele ser alta porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.
- Los **virus** no dañan los datos porque éstos están guardados en el servidor de la aplicación.
- **Colaboración**: Gracias a que el acceso al servicio se realiza desde una única ubicación es sencillo el acceso y compartición de datos por parte de varios usuarios. Tiene mucho sentido, por ejemplo, en aplicaciones online de calendarios u oficina.
- Los navegadores ofrecen **cada vez más y mejores funcionalidades** para crear aplicaciones web ricas (RIAs).

## Inconvenientes

- Habitualmente ofrecen **menos funcionalidades** que las aplicaciones de escritorio. Se debe a que las funcionalidades que se pueden realizar desde un navegador son más limitadas que las que

se pueden realizar desde el sistema operativo. Pero cada vez los navegadores están más preparados para mejorar en este aspecto. La aparición de HTML 5 representa un hito en este sentido. Es posible añadir funcionalidades a estas aplicaciones gracias al uso de Aplicaciones de Internet Ricas.

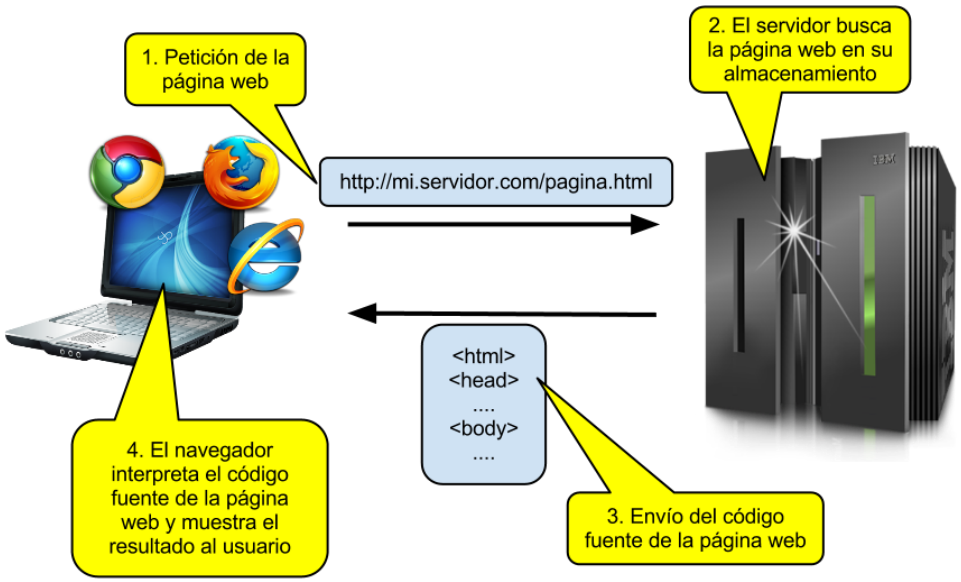
- La **disponibilidad depende de un tercero**, el proveedor de la conexión a internet o el que provee el enlace entre el servidor de la aplicación y el cliente. Así que la disponibilidad del servicio está supeditada al proveedor.

A continuación, se muestran las principales **técnicas de desarrollo** de aplicaciones web diferenciando el lugar de ejecución que puede ser en el lado del cliente (navegador) o en el servidor.

## Aplicaciones web dinámicas del lado cliente

### Lenguajes embebidos

Los casos más sencillos de desarrollo de aplicaciones web son los lenguajes de programación que son **interpretados directamente por los navegadores** (lenguajes de programación web del lado cliente) y que se encuentran embebidos dentro del código fuente *HTML* de la página web. Las páginas web que contienen código escrito en ese tipo de lenguajes, viajan a través de la red con todo ese código y el navegador, al recibir el contenido de la página web, será el responsable de interpretarlo para ofrecer el resultado de su ejecución al usuario.



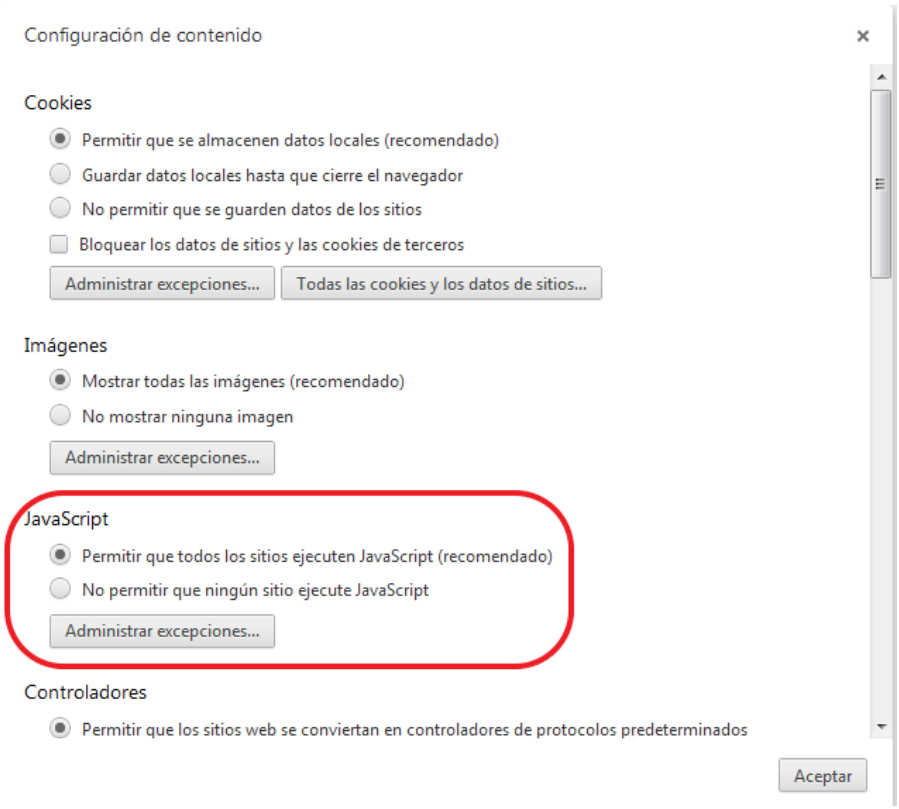

Uno de los casos más utilizados de este tipo de lenguajes es ***JavaScript***. Este lenguaje utiliza una sintaxis similar a *C* o *Java* y se integra dentro del código *HTML* debiendo estar encerrado en la etiqueta <script>.

A continuación, puedes ver un **ejemplo** con el código fuente correspondiente a una página web que muestra un reloj que se va actualizando cada segundo. Si creas una página web con este código fuente podrás comprobar que su contenido es dinámico (el reloj se actualiza de forma automática). Esto sería imposible de realizar exclusivamente con el lenguaje HTML.

```
<html>
   <body>

       <script language="javascript" type="text/javascript">
           //RELOJ 24 HORAS
           //
           //Autor: Iván Nieto Pérez
           //Este script y otros muchos pueden
           //descarse on-line de forma gratuita
           //en El Código: www.elcodigo.com

           var RelojID24 = null
           var RelojEjecutandose24 = false

           function DetenerReloj24 (){
               if(RelojEjecutandose24)
                   clearTimeout(RelojID24)
               RelojEjecutandose24 = false
           }

           function MostrarHora24 () {
               var ahora = new Date()
               var horas = ahora.getHours()
               var minutos = ahora.getMinutes()
               var segundos = ahora.getSeconds()
               var ValorHora

               //establece las horas
               if (horas < 10)
                   ValorHora = "0" + horas
               else
                   ValorHora = "" + horas

               //establece los minutos
               if (minutos < 10)
                   ValorHora += ":0" + minutos
               else
                   ValorHora += ":" + minutos

               //establece los segundos
               if (segundos < 10)
                   ValorHora += ":0" + segundos
               else
                   ValorHora += ":" + segundos

               document.reloj24.digitos.value = ValorHora
```

```
               //si se desea tener el reloj en la barra de estado, reemplazar la ant
erior por esta
               //window.status = ValorHora

               RelojID24 = setTimeout("MostrarHora24()",1000)
               RelojEjecutandose24 = true
           }

           function IniciarReloj24 () {
               DetenerReloj24()
               MostrarHora24()
           }

           window.onload = IniciarReloj24;
           if (document.captureEvents) {                      //N4 requiere invocar
la funcion captureEvents
               document.captureEvents(Event.LOAD)
           }

       </script>

       <!-- Para visualizar el reloj -->
       <p>Hora actual mostrada en cuadro de texto:</p>
       <div class="recuadro3">
           <form name="reloj24">
               <input type="text" size="8" name="digitos">
           </form>
       </div>

   </body>
</html>
```

Para que el navegador web pueda interpretar el código JavaScript es necesario que lo **soporte**. Todos los navegadores actuales más frecuentes lo soportan y ofrecen la posibilidad de poder **desactivar** la interpretación de este lenguaje en algún apartado de la configuración.



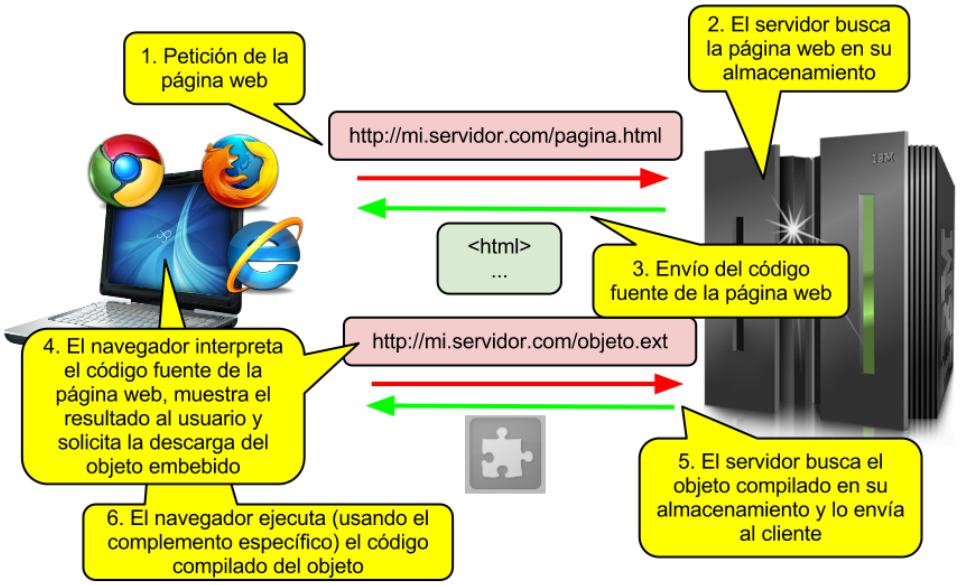

Además de *JavaScript* existen otros lenguajes similares que también se interpretan por el navegador. Por ejemplo:

- *VBScript*: Su filosofía es la misma que la de *JavaScript*, pero la sintaxis de este lenguaje está basado en *Visual Basic*.

```
<HTML>
   <HEAD>
       <TITLE>Cuadro de mensaje</TITLE>
   </HEAD>
   <BODY>
       <SCRIPT LANGUAGE = "VBScript">
           MSGBOX ("Ejemplo de mensaje")
       </SCRIPT>
   </BODY>
</HTML>
```

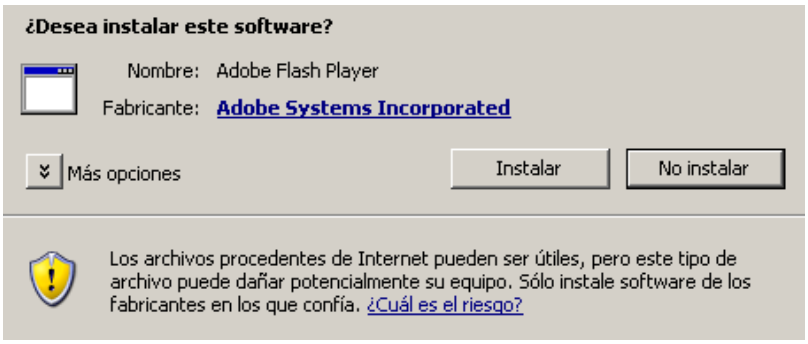
### Objetos embebidos

Otra manera de poder ejecutar aplicaciones web en el cliente (navegador web) es mediante objetos o componentes embebidos. En este caso **no se dispone del código fuente** de la aplicación como ocurría en los casos anteriores para que se ejecute en el navegador, a no ser que el desarrollador haya publicado por separado el código.




Con esta modalidad, el código fuente de la página web que se ha cargado en el cliente solicita al servidor un determinado objeto. Estos objetos consisten en una **aplicación compilada** que se habrá desarrollado con una de las tecnologías que existen para ello (a continuación, se mostrarán algunas de las más importantes). El cliente **descarga** el objeto desde el servidor que se indique y comienza la **ejecución** del mismo. Para poder ser ejecutado el objeto es necesario que el navegador disponga de una **extensión o complemento (plugin)** que lo permita. Por otro lado, conviene destacar que **no todos los navegadores** son totalmente **compatibles** con algunas de las tecnologías empleadas para integrar estos tipos de objetos en las páginas web.

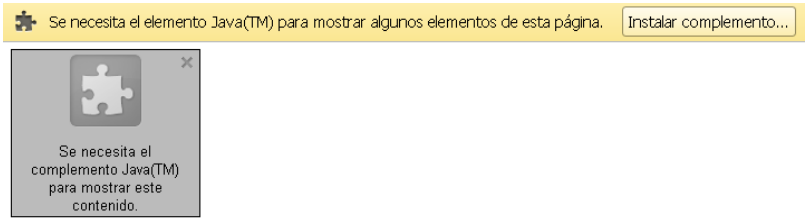
- ~~***Flash***: *Adobe Flash* es una plataforma multimedia usada para añadir animaciones, vídeo e interactividad a las páginas web. Se usa frecuentemente en publicidad, juegos, animaciones, y para el despliegue de audio y vídeo por Internet. Permite capturar las pulsaciones del usuario sobre el ratón y el teclado, así como el micrófono o la cámara. Utiliza un lenguaje de programación llamado *ActionScript*.~~
  - ~~Como ejemplo puedes visitar la web juegosjuegos.com donde puedes encontrar gran cantidad de juegos a través de la web que utilizan esta tecnología.~~
  - ~~El navegador web mostrará una información similar a la siguiente si no se dispone del complemento necesario (*Adobe Flash Player*):~~




- ~~Este es un trozo de código fuente de una página web que puede ser un ejemplo del código en el que se solicita la carga del objeto *Flash*:~~

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=7,0,0,0" width="100%" height="100%"/>
   <param name="movie"
    value="http://cdn.juegosjuegos.com/games14/9396.swf" />
   <param name="quality" value="high" />
    <param name="allowScriptAccess" value="always" />
</object>
```

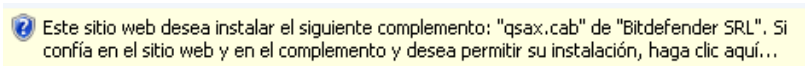
- **Applet Java**: es un *applet* (pequeña aplicación) escrito en el lenguaje de programación *Java* previamente compilado, que puede ser ejecutado por un navegador web que soporte *Java*, utilizando la máquina virtual de *Java* que debe estar instalada previamente.
  - Varios ejemplos con posibilidad de ver el código fuente de los *applets*.
  - El navegador web mostrará una información similar a la siguiente si no se dispone del complemento necesario (máquiva virtual *Java*):




  - Este es un trozo de código fuente de una página web que puede ser un ejemplo del código en el que se solicita la carga del *Applet Java*:

```
<applet codebase="." code="Clock.class" width=170 height=150>
   alt="Your browser understands the <APPLET> tag but isn't running the applet, for some reason."
   Your browser is completely ignoring the <APPLET> tag!
</applet>
```

- ***ActiveX***: Los controles *ActiveX* es una tecnología creada por *Microsoft* basada en pequeños bloques de programas que pueden utilizarse para distribuir aplicaciones a través de Internet mediante navegadores web, al igual que las tecnologías anteriores. Se pueden utilizar para recoger datos, mostrar los contenidos de determinados tipos de archivos, ver animaciones, etc.
    - Ejemplo: Bitdefender QuickScan.
    - Muchas aplicaciones *Microsoft Windows* como *Microsoft Office, Microsoft Visual Studio, y Windows Media Player* también usan controles *ActiveX* para ofrecer sus funcionalidades.
    - El navegador web mostrará una información similar a la siguiente si no se tiene instalado el control *ActiveX* solicitado:




    - Este es un trozo de código fuente de una página web que puede ser un ejemplo del código en el que se solicita la carga del control *ActiveX*:

```
<object id="CommonDialog1" width=32 height=32 classid="CLSID:F9043C85-F6F2-101A-A3C9
-08002B2F49FB"
codebase="http://activex.microsoft.com/controls/vb5/comdlg32.cab
        #Version=1,0,0,0">
</object>
```

- ***Silverlight***: *Microsoft Silverlight* es una plataforma de aplicaciones para Internet similar a lo que ofrece ~~*Adobe Flash*~~, integrando multimedia, gráficos, animaciones e interactividad. *Silverlight* es también una de las plataformas de desarrollo de aplicaciones para ~~*Windows Phone*~~.
    - Aquí puedes encontrar varios ejemplos.
    - El navegador web mostrará una información similar a la siguiente si no se tiene instalado el complemento necesario (*Microsoft Silverlight*):


    - Este es un trozo de código fuente de una página web que puede ser un ejemplo del código en el que se solicita la carga del objeto *Silverlight*:

```
<object id="silverlightBrowser" data="data:application/x-silverlight-2," type="appli
cation/x-silverlight-2"
       width="100%" height="100%">
   <param name="source" value="ClientBin/SampleBrowser.xap"/>
   <param name="background" value="white" />
</object>
```

- ***JavaFX***: Es una tecnología enfocada para el desarrollo de aplicaciones para y escritorio, soportada por *Oracle* basándose en el lenguaje de programación *Java*.
    - Aquí puedes probar ~~un ejemplo (juego BrickBreaker~~), y desde la web de Oracle puedes descargar otros ejemplos con su código fuente.
    - El navegador web mostrará una información similar a la siguiente si no se tiene instalado el complemento necesario (Máquina virtual *Java*):


    - Este es un trozo de código fuente de una página web que puede ser un ejemplo del código en el que se solicita la carga del objeto *JavaFX*:

```
<script src="http://dl.javafx.com/1.2/dtfx.js"></script>
<script>
   javafx(
       {
             archive: "BubbleBath.jar",
             width: 600,
             height: 500,
             code: "BubbleBath",
             name: "BubbleBath"
       }
   );
</script>
```

### AJAX

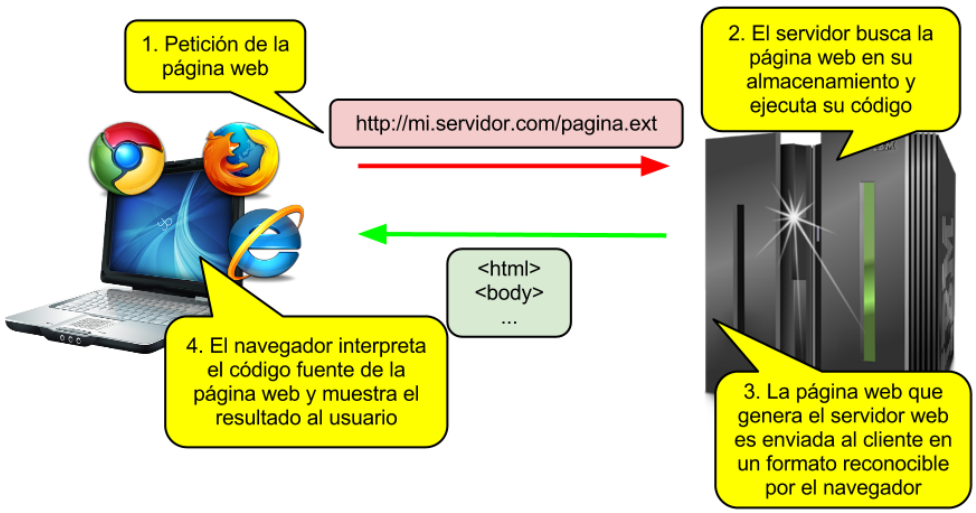
La tecnología AJAX (acrónimo de *Asynchronous JavaScript and XML*) es un grupo de técnicas de desarrollo web usadas en el lado cliente para crear aplicaciones web asíncronas. Con *AJAX*, las aplicaciones web pueden **enviar y recibir datos de un servidor de forma asíncrona** (en segundo plano) sin que interfiera en la vista o en el funcionamiento de la página web.

*AJAX* no es una sola tecnología, sino un **grupo de tecnologías**. ***HTML* y *CSS*** se usan conjuntamente para dar forma y estilo a la información. El acceso a los elementos que componen la página web (**DOM**) se realiza a través de ***JavaScript*** para mostrar la información de forma dinámica y permitir al usuario interactuar con dicha información presentada. A través de ***XML*, JSON** u otros formatos similares, se realiza el intercambio de información entre el navegador y el servidor para evitar así la recarga completa de la página web.

Para ver un ejemplo puedes consultar un *Trending Topic* de Twitter (o cualquier otro que se actualice con mucha frecuencia) y utiliza alguna herramienta que te permita ver el tráfico de red que se produce sin recargar la página (por ejemplo, el navegador Chrome dispone de la herramienta *Inspeccionar elemento* > *Network*).

## Aplicaciones web dinámicas del lado servidor

No sólo el navegador (lado cliente) puede crear dinamismo en las páginas web. El **servidor web puede devolvernos una información u otra** en una misma página web en función de la interacción del usuario o de cualquier otra circunstancia. Para ello se utilizan diversos **lenguajes de programación** que generan la página web que se enviará finalmente al cliente y en un formato entendible por el navegador (HTML, CSS, JavaScript, etc). La página web que es enviada deberá emplear los lenguajes reconocidos por el navegador, ya que los **navegadores web no reconocen los lenguajes** de programación que se emplean en los **servidores web**. En este tipo de aplicaciones es muy frecuente que el servidor web acceda a una **base de datos** para obtener los datos que finalmente formarán parte de la página web.




A continuación, se muestran algunas de las técnicas y lenguajes de programación que se pueden emplear para el desarrollo de aplicaciones web del lado servidor.

## CGI

El *Common Gateway Interface (CGI)* es un método estándar empleado por servidores web para delegar la generación del contenido web a archivos ejecutables. Dichos archivos son conocidos como ***scripts CGI***, que son aplicaciones escritas generalmente en **lenguajes de programación** que permiten crear archivos ejecutables como *C, C++, Perl, Java, Visual Basic ...* aunque es muy habitual el uso de **Perl** para el desarrollo de scripts CGI por su facilidad en el manejo de cadenas de caracteres.

Las aplicaciones CGI fueron unas de las primeras técnicas empleadas para crear contenido dinámico para las páginas web.

Seguidamente puedes ver un ejemplo de código fuente Perl que crea un script CGI para mostrar la fecha actual en una página web:

```
#!/usr/bin/perl

@months = qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec);
@weekDays = qw(Sun Mon Tue Wed Thu Fri Sat Sun);
($second, $minute, $hour, $dayOfMonth, $month, $yearOffset, $dayOfWeek, $dayOfYear,
$daylightSavings) = localtime();
$year = 1900 + $yearOffset;
$theTime = "$weekDays[$dayOfWeek] $months[$month] $dayOfMonth, $year";

print "Content-type: text/html\n\n";
print <<HTML;
<html>
<head>
<title>A Simple Perl CGI</title>
</head>
<body>
<h1>A Simple Perl CGI</h1>
<p>$theTime</p>
</body>
HTML
exit;
```

## Servlets

Un Servlet es una tecnología web basada en Java para el lado servidor, basada en recibir las respuestas del cliente y ofrecer una respuesta al servidor web. Es una clase Java de **Java EE** que utiliza la API Java Servlet, y su código suele tener **código HTML embebido**.

El **contenido generado** por el Servlet suele ser HTML pero puede generar otros tipos de datos como XML.

Para ejecutar un Servlet es necesario usar un **contenedor** web (también conocido como contenedor de Servlets) que es un componente de un servidor web que interactúa con los Servlets. El contenedor es responsable de manejar el ciclo de vida de los Servlets, mapear una URL a un Servlet en particular y

asegurarse de que el solicitante de la URL tiene los permisos de acceso adecuados.

Las **ventajas** de usar Servlets en vez de CGI es que los primeros tienen mejor rendimiento y facilidad de uso combinado con más potencia.

A continuación, puedes ver un **ejemplo** de código fuente de un Servlet que obtiene los datos introducidos por el usuario en un formulario anterior:

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class ParamServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    public void doPost(HttpServletRequest req, HttpServletResponse res) throws Servl
etException, IOException {
        // Obtenemos un objeto Print Writer para enviar respuesta
        res.setContentType("text/html");
        PrintWriter pw = res.getWriter();
        pw.println("<HTML><HEAD><TITLE>Leyendo parámetros</TITLE></HEAD>");
        pw.println("<BODY BGCOLOR=\"#CCBBAA\">");
        pw.println("<H2>Leyendo parámetros desde un formulario html</H2><P>");
        pw.println("<UL>\n");
        pw.println("Te llamas " + req.getParameter("NOM") + "<BR>");
        pw.println("y tienes "  + req.getParameter("EDA") + " años<BR>");
        pw.println("</BODY></HTML>");
        pw.close();
    }
}
```

## Lenguajes embebidos en el código HTML (PHP, ASP, JSP)

A diferencia de los lenguajes comentados anteriormente, cuyo código genera etiquetas HTML, existen otros lenguajes de programación de aplicaciones web del lado servidor cuyas **instrucciones se encuentran embebidas en el código HTML**. Es el caso de lenguajes como **PHP, ASP o JSP**.

El servidor tiene almacenadas los archivos que contienen el código correspondiente a las páginas web. Dichos archivos contienen etiquetas HTML y código en otro lenguaje que es **interpretado por el servidor**, ejecutando sus instrucciones con las que se **obtiene un código reconocible por el navegador** cliente (HTML, CSS, JavaScript, etc). El resultado final obtenido es el que se envía al cliente.

Estas son las características principales de los lenguajes más utilizados en esta categoría:

- PHP:
  - Es uno de los primeros lenguajes del lado servidor creados para ser embebido en el código HTML en lugar de llamar a un archivo externo para procesar los datos. El código es interpretado por un servidor web con un módulo procesador de PHP que genera la página web resultante. PHP se puede emplear en la mayoría de los servidores web de los principales sistemas operativos de forma gratuita al ser software libre. Se estima que se utiliza en más de 20 millones de sitios web. Por ejemplo, se usa en gestores de contenido web como *Joomla* o *Drupal*, blogs como *Wordpress*, o aulas virtuales como *Moodle*.

```
<html>
 <head>
  <title>Ejemplo PHP</title>
 </head>
 <body>
 <?php echo '<p>Hola Mundo</p>'; ?>
 </body>
</html>
```

- ASP:
  - *Active Server Pages* (ASP), también conocido como ASP clásico, fue la primera tecnología de ***Microsoft*** para generar páginas web de forma dinámica desde el lado del servidor, lanzándose como un añadido al servidor ***Internet Information Services* (IIS)**. La mayoría de los programadores crean las páginas ASP usando ***VBScript***, pero se pueden utilizar otros lenguajes.

```
  <HTML>
  <HEAD>
    <TITLE>Ejemplo</TITLE>
  </HEAD>
  <BODY>
    <% Response.Write("Hola, mundo!") %>
  </BODY>
  </HTML>
```

- ASP.NET
  - Es el sucesor de la tecnología ASP y permite a los programadores escribir código usando cualquier **lenguaje .NET** soportado, como VB.NET o C#.
  - Sólo funciona sobre el servidor de ***Microsoft* IIS**, lo que supone una desventaja respecto a otros lenguajes del lado de servidor que son ejecutables sobre otros servidores más populares.

```
<%@ Page Language="VB" ContentType="text/html" ResponseEncoding="utf-8"%>
<script runat="server">
  Sub Page_Loadp1.InnerHtml = "<b>Hola Mundo</b> "
  End Sub
</script>
<html>
  <body>
    <form runat="server">
      <p id="p1" runat="server" />
    </form>
  </body>
</html>
```

- JSP:
  - *JavaServer Pages* (JSP) es otra trecnología para crear dinámicamente páginas web de forma similar a PHP pero usando el lenguaje de programación **Java**. Para utilizarla es necesario disponer de un **contenedor de servlets** como Apache Tomcat.
  - El **funcionamiento** general de la tecnología JSP es que el Servidor de Aplicaciones interpreta el código contenido en la página JSP para construir el código Java del servlet a generar. Este servlet será el que genere el documento (típicamente HTML) que se presentará en la pantalla del navegador del usuario.
  - El **rendimiento** de una página JSP es el mismo que tendría el servlet equivalente, ya que el código es compilado como cualquier otra clase Java. A su vez, la máquina virtual compilará dinámicamente a código de máquina las partes de la aplicación que lo requieran. Esto hace que JSP tenga un buen desempeño y sea más **eficiente** que otras tecnologías web que ejecutan el código de una manera puramente interpretada.
  - JSP hereda la **portabilidad** de Java, y es posible ejecutar las aplicaciones en múltiples plataformas sin cambios. Es común incluso que los desarrolladores trabajen en una plataforma y que la aplicación termine siendo ejecutada en otra.

```
<html>
  <head>
    <title>Ejemplo PHP</title>
  </head>
  <body>
    <% out.println("Hola mundo"); %>
  </body>
</html>
```