

DESARROLLO WEB EN ENTORNO SERVIDOR
TÉCNICO EN DESARROLLO DE APLICACIONES WEB

Selección de arquitecturas y herramientas de programación

01

índice

/ 1. Introducción y contextualización práctica	3
/ 2. Modelos de programación en entornos cliente/servidor	4
/ 3. Páginas web dinámicas	5
3.1. Componentes. Capas y funciones	5
/ 4. Lenguajes de programación entorno servidor	6
/ 5. Integración con los servidores web	7
/ 6. Herramientas de programación I	8
/ 7. Herramientas de programación II	9
/ 8. Tecnologías asociadas. Instalación AMP	9
/ 9. Despliegue y puesta en marcha del entorno AMP	10
/ 10. Implementación de una nueva página con PHP	12
/ 11. Caso práctico 1: “Primera página en PHP”	13
/ 12. Node.js. Javascript en el servidor	14
/ 13. TypeScript	15
/ 14. Caso práctico 2: “Creación de un servidor web con Node.js”	16
/ 15. Resumen y resolución del caso práctico de la unidad	17
/ 16. Bibliografía	18

OBJETIVOS

Diferenciar los modelos de programación en entornos cliente y servidor.

Reconocer las ventajas que proporciona la generación dinámica de páginas web.

Identificar los principales lenguajes y tecnologías relativas a la programación web en entorno servidor.

Evaluar las herramientas de programación y desarrollo en entorno servidor.

/ 1. Introducción y contextualización práctica

El desarrollo web en entorno servidor es uno de los pilares fundamentales de la implementación web en la actualidad. La construcción de este entorno permitirá al cliente acceder a infinitos recursos servidos de forma estática o dinámica. Estos sistemas se organizan en módulos y capas que optimizan su construcción y funcionamiento.

En este primer tema, analizaremos el funcionamiento de las peticiones y respuestas sobre un servidor, centrándonos en las páginas web dinámicas, gracias a las cuales la respuesta, en cada momento, se adaptará a la petición del usuario.

Veremos diferentes herramientas y tecnologías clave para el desarrollo práctico de este módulo, desde la instalación y puesta en funcionamiento de XAMPP, hasta la creación de un servidor web utilizando Node.js, entorno que también permite utilizar JavaScript del lado del servidor y no solo para modelar las interfaces en el lado del cliente.

Escucha el siguiente audio donde planteamos la contextualización práctica de este tema. Encontrarás su resolución en el apartado «Resumen y resolución del caso práctico de la unidad».



Fig. 1. Aplicación web, aplicación de escritorio y página web.



Audio intro. "Páginas web, aplicaciones web y aplicaciones de escritorio"

<https://bit.ly/3fHrIAy>





/ 2. Modelos de programación en entornos cliente/servidor

El acceso a una página web es el resultado de un conjunto de interacciones entre varios entornos. Cuando accedemos a un sitio web a través de su URL, introducimos la dirección del sitio y la web se solicita a un servidor que, casi de forma automática, la devuelve y, de este modo, aparece en nuestro navegador.

Estamos hablando de un **entorno de programación cliente-servidor**, en el que el **cliente** se **comunica** con un **servidor** para **obtener** cierto tipo de **información**. Por ejemplo, unos datos de **consulta** o, incluso, una interfaz de aplicación personalizada en función del usuario (un claro ejemplo de esto sería la bandeja de correo personalizado de cada usuario).



Fig. 2. Diagrama petición de página web estática.

Si el acceso se realiza a páginas cuyo funcionamiento no varía y se muestran tal y como se crearon, se trata de **páginas web estáticas**. En cambio, las páginas web dinámicas sí cambian su valor en función de diferentes variables.

En el caso de las páginas estáticas, se establece una comunicación cliente-servidor en la que el usuario solicita, a través del navegador (cliente), una página a un servidor de páginas web. Ante esta petición, el servidor busca la página internamente y la recupera. Finalmente, la envía al navegador que hizo la petición para que este muestre su contenido.

Atendiendo al diagrama anterior, en el caso de las páginas estáticas, la secuencia de pasos para recuperar un sitio web es:

- El cliente introduce una URL en el navegador.
- Se modela una petición HTTP, que se envía al servidor web.
- El servidor realiza una búsqueda de recursos solicitados y los recupera.
- Finalmente, la página se envía de nuevo al cliente, como respuesta a la petición HTTP.

Como se puede observar, esta secuencia de pasos siempre va a devolver el mismo resultado. Esto es de gran utilidad cuando queremos mantener un enlace permanente a un recurso web. De esta forma, nos aseguramos de que no cambia.



Audio 1. "Diferencia entre servidor web y servidor de aplicaciones"
<https://bit.ly/2CbyTPB>





/ 3. Páginas web dinámicas

Las páginas web dinámicas muestran un contenido distinto en función de diferentes valores, como los datos de entrada, el navegador, o la autenticación del usuario. La programación de estas páginas se realiza utilizando HTML dinámico, a través del cual el recurso recibido por un usuario varía con respecto al que puede recibir otro.



Fig. 3. Diagrama petición de página web dinámica.

El funcionamiento de este tipo de páginas se realiza en función del siguiente algoritmo de pasos:

- El usuario **introduce en el navegador una URL de la página** o servicio que desea visitar.
- Se **envía una petición HTTP**, en la que se suelen incluir diferentes datos de consulta al destino.
- El **servidor envía la petición** al módulo donde se va a ejecutar el código para recuperar los datos de la consulta. Este se encargará de realizar las conexiones oportunas a los almacenes de datos y de recuperar el contenido.
- **Desde el servidor, se modela la respuesta en formato HTML**. En función de los datos recuperados, se generará un contenido u otro.
- Finalmente, **el servidor envía la página resultado al navegador** (al cliente).

Para construir este tipo de páginas, existen dos tipos de programación:

- En un fichero HTML, en el que se incluyen funciones en lenguaje JavaScript que permiten evaluar el comportamiento del usuario y hacer que la página muestre un contenido u otro en función de esta entrada. En este tipo de casos los lenguajes principales serán HTML y JavaScript.
- Existe otro tipo de páginas implementadas en lenguajes como PHP, [ASP.net](https://www.asp.net/) o Java. En este tipo de casos, las páginas mostradas al usuario son el resultado de la ejecución de un programa.

3.1. Componentes. Capas y funciones

Tras las diferentes estructuras de generación de páginas web, podemos observar que existe un conjunto de componentes clave que participan en el proceso.

- **Lenguaje de programación:** Utilizado para implementar las aplicaciones web.
- **Servidor web:** Este se encarga de recibir la petición desde el cliente, de establecer la comunicación con el módulo encargado de generar la respuesta y, finalmente, de enviar la página resultante al navegador del cliente.

- **Módulo de código:** Se integra en el servidor web y, en función de la tecnología que implemente, llevará a cabo la generación de la página respuesta. El módulo de código suele establecer una comunicación con los componentes de almacenamiento de datos, habitualmente las bases de datos.
- **Base de datos:** En el caso de ser necesario la recuperación, modificación o cualquier otra acción sobre datos, el módulo de código establece la conexión con una base de datos.
- **Los entornos de desarrollo,** sobre todo en el caso de las conexiones con el servidor, se estructuran mediante un sistema de capas, puesto que, de esta forma, las funciones entre niveles quedan claramente diferenciadas. En concreto, hablamos de funciones a nivel lógico, de tal forma que será posible alojar cada una de estas en un servidor diferente.

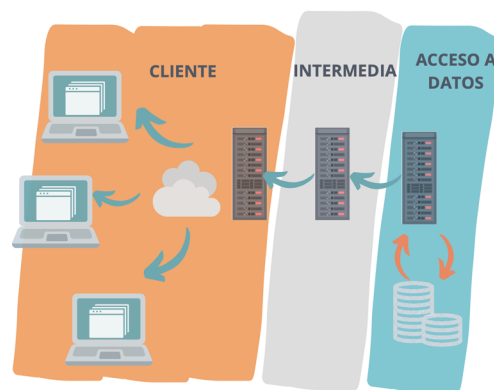


Fig. 4. Diagrama de conexión con el servidor por capas.

Los tres niveles son:

- **Capa o nivel cliente:** Aquí se encuentra todo lo relativo a la interfaz de usuario. Es la parte visible de la aplicación.
- **Capa intermedia:** En este nivel, se ubica la funcionalidad de la aplicación. Esta se encuentra en la parte del servidor.
- **Capa de acceso a datos:** Esta última capa es la encargada del almacenaje y recuperación de información o recursos.

Cada una de estas capas se ocupa de una o de varias funciones, entre las que destacamos la función del acceso, las funciones de presentación y lógica, y la función de persistencia, que se encarga de recibir la petición desde el cliente, establecer la comunicación con el módulo encargado de generar la respuesta y, finalmente, de enviar la página resultante al navegador del cliente.

/ 4. Lenguajes de programación entorno servidor

Una de las primeras tecnologías que se comenzó a utilizar en el desarrollo en entorno servidor fue CGI (Common Gateway Interfaces). Actualmente, está en desuso porque resultaba poco eficiente, ya que requería de una ejecución de programa para cada una de las peticiones de un cliente.

Actualmente, los lenguajes de Scripting son una herramienta clave para el desarrollo en entorno servidor. Estos hacen referencia a aquellos lenguajes que pueden ser embebidos con el código HTML utilizado para la implementación de páginas web. A través de estos fragmentos, se posibilita la conexión con el servidor y se generan las páginas web dinámicas.



- **PHP (Hypertext Processor):** Lo soportan casi todos los servidores web y es de código abierto. En la actualidad, es uno de los lenguajes de programación en entorno servidor más utilizados, gracias a su potencia y simplicidad. PHP permite embeber fragmentos de código desarrollado en este lenguaje en una página web HTML. Se trata de una tecnología del lado del servidor, también conocido como backend. Una de las características más importantes es que permite utilizar tanto programación estructurada como orientada a objetos.
- **JSP (Java Server Pages):** En el caso de servidores de aplicaciones empresariales, se utiliza Java como lenguaje de programación. Gracias a JSP, será posible desarrollar aplicaciones web que se ejecutan en varios servidores. Para el desarrollo de estas páginas, se utiliza HTML/XML, junto con etiquetas que permiten incluir sintaxis en Java.

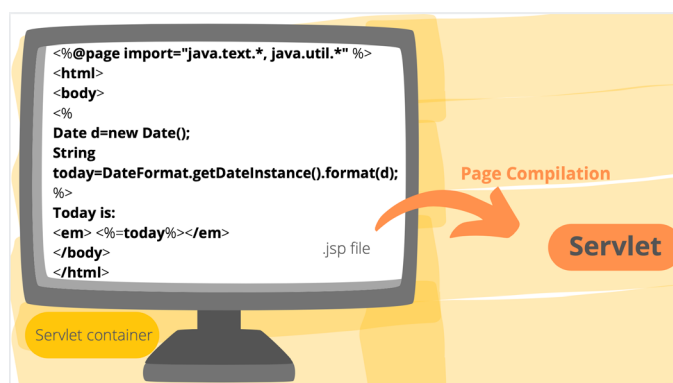


Fig. 5. Funcionamiento básico de servlet.

- **ASP (Active Server Pages):** Se trata de la tecnología desarrollada por Microsoft, encargada de crear páginas web dinámicas en el lado del servidor. Estas páginas comienzan su ejecución cuando el cliente solicita un archivo .asp al servidor.
- **Python:** Un lenguaje de programación bastante utilizado en la actualidad, que se encuentra disponible en múltiples plataformas. Además, Django, uno de los frameworks más utilizados en los servidores web más populares, está implementado en Python.

/ 5. Integración con los servidores web

La conexión entre la petición en el entorno cliente y el servidor web se lleva a cabo mediante al modelado HTTP. Para que esta conexión se establezca, el cliente debe generar una petición con unos elementos concretos y una sintaxis exacta, comprensible por el servidor. Veamos los dos sentidos del flujo de la información:

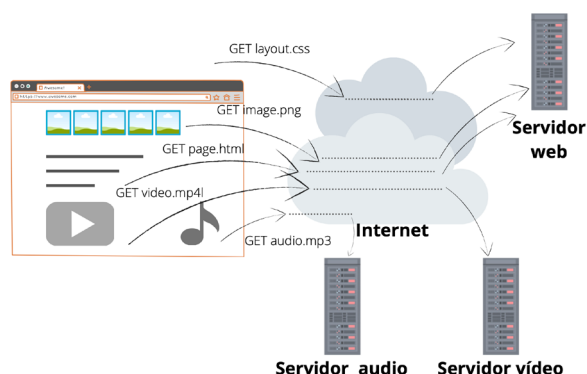


Fig. 6. Intercambio de mensajes con métodos HTTP



- **Petición cliente-servidor**

El modelado de la petición de HTTP está constituido por un método, la dirección de recursos, la versión del protocolo, las cabeceras HTTP y el cuerpo del mensaje:

- **Método HTTP:** Se utilizan dos verbos, GET, mediante el cual el cliente realiza una petición al servidor. Se utiliza para recuperar información. POST, a diferencia del anterior, se utiliza para enviar información desde el cliente al servidor.
- **URL:** La dirección del recurso solicitado.
- **Versión del protocolo HTTP:** Por ejemplo, HTTP/1.1.
- **Cabeceras HTTP:** Elemento no obligatorio que se utiliza para añadir información adicional.
- **Cuerpo de mensaje:** Se utiliza junto el método POST para añadir la información de envío al servidor.



Audio 2. "Funcionamiento del protocolo http para establecer la conexión"

<https://bit.ly/33JZv5I>



- **Respuesta servidor-cliente:** En el caso de la respuesta, se incluyen los elementos que indican a través de un código numérico y un breve mensaje. El estado de la petición, por ejemplo, 200 OK, indica que todo ha ido correctamente.

/ 6. Herramientas de programación I

Para el desarrollo web en entorno servidor, las herramientas de programación esenciales son: los navegadores, los editores de texto y algún paquete que incluye un sistema de gestión de bases de datos, servidor web, e intérprete de lenguajes de desarrollo en entorno servidor.

- Un navegador web es una aplicación que permite acceder a múltiples recursos web. Esto se realiza a través de la dirección insertada en la barra de navegación, que se sitúa en la parte superior. La petición se envía desde el cliente y se establece una conexión con el servidor. Los navegadores habituales son: Internet Explorer, Chrome, Mozilla o Safari.
- La implementación del código relativo al diseño del sitio web se puede realizar en multitud de editores de texto, desde algunos tan sencillos como NOTEPAD++, hasta otros más complejos.
- **Dreamweaver:** Es uno de los editores que admite tanto el método textual como el WYSIWYG (What You See Is What You Get), es decir, «Lo que ves es lo que obtienes», esto significa que permite programar con una presentación visual en vivo. Dreamweaver permite escribir código en todos los lenguajes de programación importantes. Está disponible para Windows y OS X.

Una de las características más importantes de esta herramienta es que dispone de vista previa. De esta manera, los desarrolladores pueden programar mientras visualizan el producto final. Además, permite confirmar el código y accesibilidad de la página, característica que puede facilitarles a los desarrolladores seguir las pautas de accesibilidad de contenido web.



Fig. 7. Logo Dreamweaver.



- **Notepad++:** Notepad++ es un editor de textos de propósito general que reconoce la sintaxis de múltiples lenguajes de programación. Es gratuito, está disponible para Windows y su código fuente se puede descargar. Los usuarios de Linux pueden usarlo a través de Wine.

Está cada vez más presente en los entornos de desarrolladores web, puesto que ofrece buenas características en comparación a lo ligero que es (ocupa poco espacio y es muy rápido). Se puede extender a través de plugins. Su interfaz es minimalista, pero los desarrolladores pueden personalizarla.



Fig. 8. Logo Notepad++.

/ 7. Herramientas de programación II

- **BEdit:** Este editor de texto para MAC OS X es uno de los editores más utilizados en este sistema operativo, puesto que supone una combinación perfecta entre la facilidad de uso, similar a la de Notepad++ y un gestor de contenido que permite estructurar el código de una forma precisa.
- **Visual Studio Code:** Es uno de los editores de texto más utilizados, compatible con múltiples lenguajes de programación y disponible para Windows, Linux y macOS. Una de sus principales características es el resaltado de sintaxis, la finalización inteligente de código. También tiene una interfaz personalizable y es gratuito.
- **NetBeans:** NetBeans es una herramienta de código abierto y es gratuita. Se trata de uno de los entornos de desarrollo más utilizados para el desarrollo de interfaces a través del lenguaje Java, aunque también permite utilizar otros lenguajes, como PHP o Python.
- **TextWrangler:** Editor de texto sencillo e intuitivo que permite programar en varios lenguajes de programación, resaltando los elementos con diferentes colores, lo cual ayuda al seguimiento del programa. Este editor se encuentra disponible para Mac OS X.

/ 8. Tecnologías asociadas. Instalación AMP

El desarrollo de aplicaciones web puede realizarse a través de varias tecnologías. En la actualidad, algunas de las más comunes son Java EE y AMP. En este módulo, nos vamos a centrar en la segunda.

La tecnología AMP (Apache + MySQL + PHP/Perl/Python) es una arquitectura de código libre que permite desarrollar todo tipo de aplicaciones web. Está disponible para todos los sistemas operativos (Mac OS X, Windows y Linux).

- **Apache:** Servidor de páginas web HTTP de código abierto, es decir, es el elemento clave para alojar los recursos web que serán recuperados a través de las peticiones oportunas.
- **MySQL:** Sistema de gestión de bases de datos, que se utiliza tanto para almacenar como para administrar diferentes tipos de datos.
- **PHP:** Lenguaje de código abierto que queda embebido en las páginas HTML y permite modelar la peticiones y posteriores respuestas de recuperación de un recurso desde el cliente hacia el servidor.

A continuación, vamos a ver de forma práctica cómo desplegar todo el entorno de desarrollo necesario para armar la tecnología AMP. En concreto, vamos a instalar XAMPP, un entorno de desarrollo que incluye un servidor web Apache, una base de datos con MySQL y el software de desarrollo en entorno servidor PHP. Este proceso es clave para poder continuar con éxito el resto del módulo.

Descarga el software completo aquí, en función del sistema operativo.



Fig. 9. Sitio oficial XAMPP.

Se inicia la instalación desde el ejecutable descargado. Selecciona todas las opciones en la siguiente pantalla y sigue pulsando Next hasta completar la instalación con Finish.

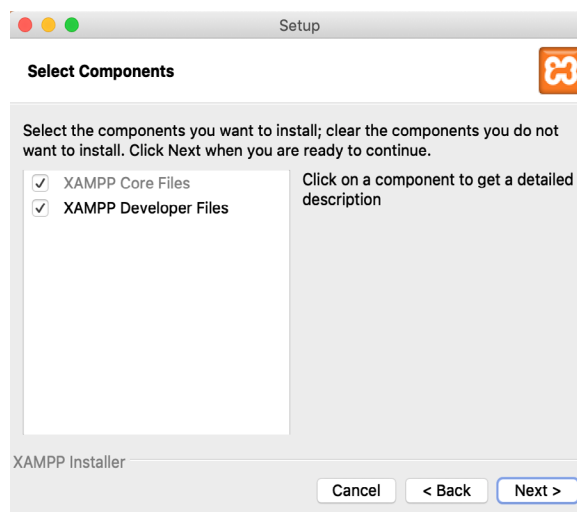


Fig. 10. Proceso de instalación. Selección de opciones.

/ 9. Despliegue y puesta en marcha del entorno AMP

Tras la instalación, debes seguir una serie de pasos para poner en funcionamiento el entorno, puesto que, por ahora, todo está apagado. Esta versión nos enlaza directamente con la carpeta de ficheros de XAMPP desde Open Application Folder. El acceso a esta carpeta es importante, puesto que aquí se colocan los archivos HTML.



Fig. 11. Interfaz aplicación XAMPP.

Desde el botón Go to Application, accedemos de forma directa a la página de la aplicación, pero es importante realizar el siguiente paso antes de acceder a este sitio, puesto que, de lo contrario, no funcionará, ya que la base de datos está apagada.

Desde los botones del menú superior, en Manage Servers, activamos, al menos, el servidor web y la base de datos. Para inicializar todo, basta con pulsar el botón Start All de la parte inferior de la pantalla.

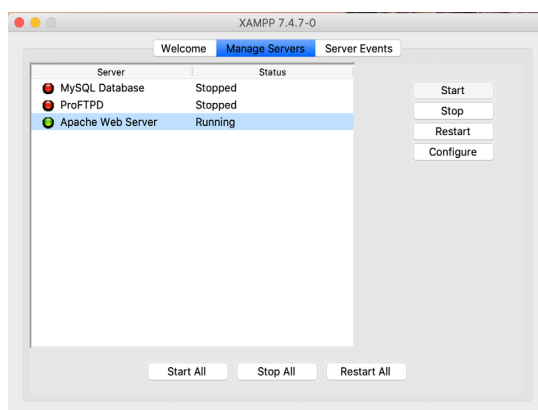
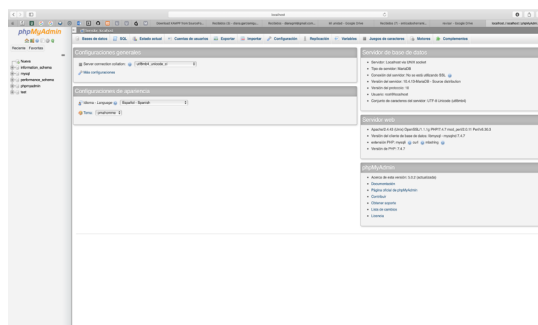


Fig. 12. Manage Servers.

Si ahora regresamos a la zona de Bienvenida (Welcome) y accedemos a Go to Application, podemos acceder sin problema a la siguiente pantalla de administración del sitio, que revisaremos en profundidad más adelante.

Fig. 13. Página de administración <http://localhost/phpmyadmin/>.



/ 10. Implementación de una nueva página con PHP

Uno de los objetivos para el despliegue de este entorno es disponer de un servidor con PHP, puesto que, de lo contrario, no podríamos probar nuestros proyectos desarrollados utilizando esta tecnología.

En temas posteriores estudiaremos en profundidad las reglas de uso de este lenguaje de programación, pero, en este apartado, solo sentaremos las bases para desplegar y probar los archivos implementados con PHP.

Hasta ahora, solo hemos accedido a la página <http://localhost/phpmyadmin/>, pero, si queremos crear un nuevo proyecto y probar su funcionamiento, seguiremos los siguientes pasos:

- Desde la carpeta htdocs, se crea una nueva carpeta para cada proyecto nuevo. Esto no es completamente necesario, puesto que podríamos crear un nuevo fichero sin necesidad de crear más carpetas, pero, de esta manera, el resultado queda mucho más organizado.
- A continuación, utilizando alguno de los editores de texto tratados en el apartado 6 y 7, implementaremos el código de desarrollo del proyecto. Dentro de las etiquetas <body> colocamos la propias de PHP, como se muestra en el siguiente esqueleto.

```
<html>
<head>...</head>
<body>
  <?php
    //CÓDIGO PHP
  ?>
</body>
</html>
```

Código 1. Código base de inclusión PHP en HTML.

Finalmente, se guarda el fichero añadiendo la extensión .php al nombre del archivo.

Desde la barra del navegador, accedemos a la URL en la que se aloja la página PHP. Si se ha situado dentro de alguna carpeta, recuerda que tienes que indicar la ruta exacta de la siguiente manera:

<http://localhost/carpeta1/carpeta2/.../nombre.php>



Vídeo 1. "Configuración básica de XAMPP. Primeros pasos PHP"
<https://bit.ly/2XLA143>





/ 11. Caso práctico 1: “Primera página en PHP”

Planteamiento: Para comprobar que el entorno para desarrollo con lenguaje PHP, XAMP, se ha instalado correctamente, se nos propone implementar un pequeño código de prueba y verificar que, al acceder al servidor, este muestra en nuestro navegador el mensaje «Hola Mundo».

Nudo: Desde un editor de texto, por ejemplo, TextWrangler, se implementa el siguiente programa. Recuerda que, para embeber el código en PHP, debemos utilizar la etiqueta de apertura `<?php` y de cierre `?>`. En este caso, para mostrar por pantalla el mensaje «Hola Mundo!!!», se utilizará función `echo`, que muestra la cadena de texto que se indica entre comillas.

```
<html>
<head>
  <title>Prueba de PHP</title>
</head>
<body>
  <?php
    echo "Hola Mundo!!!";
  ?>
</body>
</html>
```

Código 2. HolaMundo PHP + HTML.

Una vez desarrollado el pequeño fragmento de código, se almacena el fichero con el nombre `HolaMundo.php` en la carpeta `htdocs`. Para que todo funcione correctamente, es necesario activar el servidor desde la ventana principal de XAMP, concretamente, desde `Manager Server`, `Start All`.

Desenlace: En la barra del navegador, accedemos a la siguiente URL:

`http://localhost/holaMundo/holaMundo.php`

Como se puede observar, aparece el nombre de la nueva carpeta (`holaMundo`) y, a continuación, el nombre del fichero PHP (`holaMundo.php`). Si la salida es la siguiente, la instalación de todo el entorno es correcta y estamos preparados para implementar cualquier aplicación:

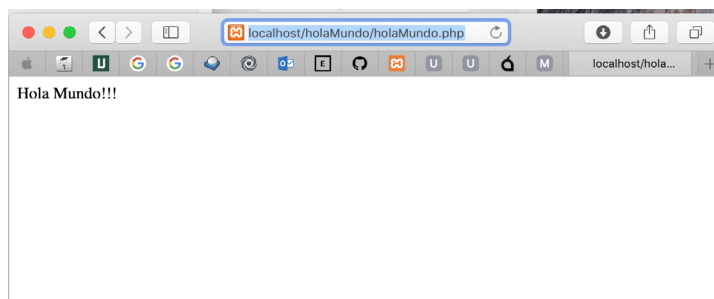


Fig. 14. Resultado código 2.

/ 12. Node.js. Javascript en el servidor

Hasta ahora, hemos hablado del desarrollo en entorno servidor a través de lenguajes de programación como PHP, pero, en la actualidad, también puede utilizarse JavaScript, más habitual en el desarrollo en cliente (frontend). No obstante, ahora también lo tenemos para la implementación del lado del servidor, en concreto, para desarrollar servidores web, aplicaciones de servidores, aplicaciones móviles...

La arquitectura Node.js se basa en el motor de JavaScript V8 y, a través del lenguaje en JavaScript y los módulos Node.js, permite desarrollar aplicaciones en el lado del servidor (backend), puesto que permite utilizar JavaScript en múltiples entornos, donde hasta ahora no era posible hacerlo.

La descarga del paquete está disponible desde su sitio web (<https://nodejs.org/en/download/>):

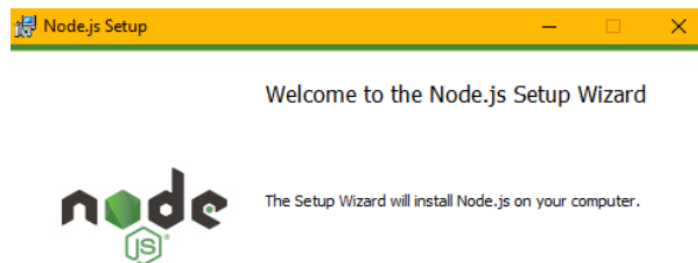


Fig. 15. Instalación Node.js.

Para realizar la instalación de Node.js en cualquier sistema operativo, tras la descarga del ejecutable correspondiente, basta con hacer clic sobre el ejecutable y se abrirá un instalador. El proceso de instalación es muy sencillo, simplemente se pulsará Next hasta completar la instalación. No necesita ninguna configuración específica.

Para comprobar si la instalación se ha realizado de forma correcta, se utilizan los comandos:

- **node --version.** Tras realizar el proceso de instalación anterior, para confirmar el resultado satisfactorio, se ejecuta esta línea por comandos y el resultado mostrará la versión de Node.js instalada.
- **npm --version.** NPM es un gestor de paquetes de JavaScript. Si al ejecutarlo se muestra una versión, está instalado. Con la instalación de Node.js, se instala también este gestor. Si se desea actualizar la versión, basta con ejecutar por línea de comandos `npm install -g npm`.

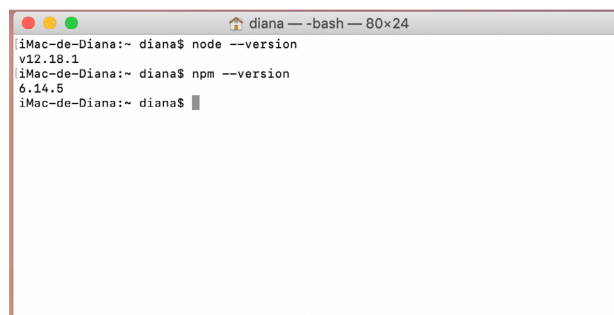


Fig. 16. Comandos instalación y resultado ejecución para Node.js.



/ 13. TypeScript

En los últimos años, JavaScript ha crecido exponencialmente, por lo que comenzaron a desarrollarse frameworks para optimizar el desarrollo de aquellas aplicaciones más grandes, como, por ejemplo, Node.js.

Mientras que en otros lenguajes de programación existen importantes entornos de desarrollo que agilizan la implementación mostrando diferentes utilidades y reduciendo el número de errores a la hora de programar, en JavaScript esto no existía.

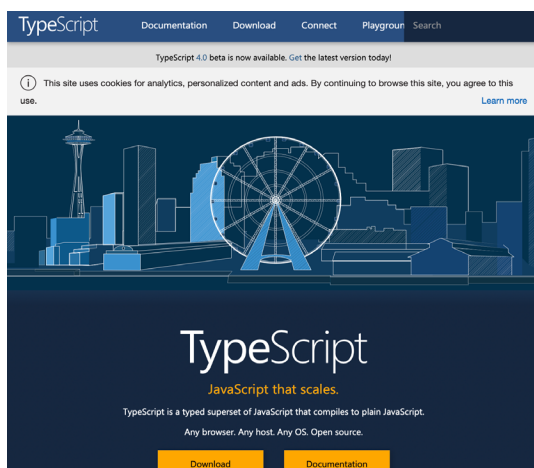


Fig17. Sitio web oficial TypeScript <https://www.typescriptlang.org>

Gracias a TypeScript, estas funcionalidades podrán incorporarse a un entorno de desarrollo, ahorrándonos diferentes errores que suelen aparecer en Javascript como, por ejemplo, que una variable se no se haya definido. Antes de la aparición de TypeScript, estos errores no eran visibles hasta que la aplicación no estaba siendo ejecutada.

En definitiva, TypeScript es JavaScript, pero añadiendo tipos de variables y tipos de métodos, entre otros.

El proceso de instalación de TypeScript es sencillo. Desde la línea de comandos del terminal se introducen la siguiente instrucción:

npm install -g typescript

Si todo ha ido correctamente, el resultado de la ejecución será algo similar a la siguiente imagen. Es posible que, para poder ejecutar la instrucción de instalación desde Linux o Mac OS X, necesites hacerlo con permisos de administrador. Para ello, basta con anteponer la palabra sudo y luego introducir la contraseña.

```
New patch version of npm available! 6.14.5 → 6.14.6
Changelog: https://github.com/npm/cli/releases/tag/v6.14.6
Run npm install -g npm to update!
```

Fig. 18. Salida por terminal tras proceso correcto de instalación TypeScript.



Vídeo 2. "Primera prueba con TypeScript en un editor de texto"
<https://bit.ly/2DLDCbe>





/ 14. Caso práctico 2: “Creación de un servidor web con Node.js”

Planteamiento: Para crear una aplicación de escritorio utilizando Node.js, no es necesario utilizar el navegador, puesto que queremos que el sistema operativo entienda JavaScript y ejecute el contenido de estos ficheros. Antes de la instalación de Node.js, sería imposible que el sistema operativo interpretara el contenido de un fichero JavaScript.

Nudo: A continuación, vamos a crear un servidor básico utilizando Node.js. En un fichero JavaScript, vamos a implementar el siguiente código, que describiremos paso a paso:

En primer lugar, importamos la librería HTTP, para cuya inclusión utilizamos require. Esta importación se almacena en una variable, como si de un objeto se tratara, lo que nos permitirá operar sobre este elemento.

```
var http = require("http");
```

Ahora, creamos el servidor web utilizando el método createServer, que recibe por parámetro una función [callback](#). Esta recibe como parámetro la petición y la respuesta.

```
http.createServer(function(req, res) {
```

A continuación, el método writeHead(...) recibe como parámetro lo que se va a mostrar en pantalla. En este caso, 200 indica que la petición se recibió de forma correcta y 'Content-type' se refiere al tipo de contenido.

```
res.writeHead(200,{ 'Content-Type': 'text/html' });
```

La función write escribirá un mensaje que se almacena en la respuesta.

```
res.write('<h2>Hola Mundo</h2>');
```

Finalmente, se cierra la respuesta utilizando el método end y se le asigna un puerto en el que el servidor estará escuchando, a la espera de recibir las peticiones.

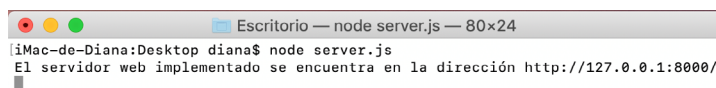
```
res.end();}).listen(8000);
```

Para concluir, se crea un log donde se indica que el servidor está en ejecución.

```
console.log('El servidor web implementado se encuentra en la dirección  
http://127.0.0.1:8000/');
```




Desenlace: Si todo ha ido bien, desde línea de comandos se ejecuta el servidor utilizando `node nombre.js` y se muestra el log indicado en el último punto. Ahora, el servidor está escuchando, por lo que, si accedemos a la dirección del localhost en el puerto señalado, nos aparecerá «Hola Mundo» en el navegador. Así, ya habríamos creado nuestro primer servidor web utilizando Node.js.



```
Escritorio — node server.js — 80x24
iMac-de-Diana:Desktop diana$ node server.js
El servidor web implementado se encuentra en la dirección http://127.0.0.1:8080/
```

Fig. 19. Creación del servidor web.

/ 15. Resumen y resolución del caso práctico de la unidad

A lo largo de este tema, hemos sentado las bases del funcionamiento de los entornos de programación cliente-servidor, en los cuales el cliente establece una comunicación a través de un modelo de peticiones con el servidor para conseguir algún tipo de información o recurso. Hemos diferenciado páginas web estáticas y dinámicas, analizando la secuencia de pasos clave que se llevan a cabo en cada sistema.

La estructura de este tipo de sistemas está formada por 4 grandes elementos: un lenguaje de programación sobre el que están implementados los sistemas y aplicaciones, un servidor web que se encarga de recibir la petición y genera la respuesta, un módulo de código que lleva a cabo la generación de la página respuesta y una base de datos.

El desarrollo de aplicaciones web puede realizarse a través de varias tecnologías. En este tema nos hemos centrado en la instalación y puesta en marcha de XAMPP (Apache + MySQL + PHP/Perl), que permite desarrollar todo tipo de aplicaciones web. El correcto despliegue de este entorno será un factor clave para toda la asignatura.

Habitualmente, JavaScript se utilizaba para el desarrollo en entorno cliente, pero, en la actualidad, se utiliza cada vez más para el entorno servidor. Es decir, ha dejado de ser exclusivo de frontend para formar parte de la implementación backend. La tecnología Node.js permite desarrollar aplicaciones en el lado del servidor, a través del lenguaje en JavaScript y los módulos Node.js.

Resolución del caso práctico de la unidad.

Como hemos escuchado en el audio, se distingue entre aplicaciones web, aplicaciones de escritorio y páginas web. La primera y la última suelen dar lugar a bastantes confusiones, sobre todo en aquellos entornos más alejados del desarrollo web.

Las páginas web muestran información al usuario de forma estática. Un ejemplo podría ser la web de un periódico.

Ejemplos claros de aplicaciones web son Gmail y todas las aplicaciones del paquete Google que se adaptan en función de la identificación del usuario. Estos son sitios web que permiten una interacción avanzada con el cliente, adaptando la interfaz, recursos, información y servicios que se muestran a cada uno de ellos.

Finalmente, una aplicación de escritorio conocida y que proporciona conexión a la web sería Microsoft Outlook.



/ 16. Bibliografía

Ganzabal, X. (2019). Desarrollo web en entorno servidor. (1.a.ed.). Madrid: Síntesis.

Vara, J. M. (2012). Desarrollo en entorno Servidor. (1.a. ed.). Madrid: Rama.

Webgrafía

Programación Lado Servidor. Recuperado de:

<https://developer.mozilla.org/es/docs/Learn/Server-side>

WUOLAC