



PHP & MySQL

Instalación y Configuración



DISTRIBUIDO POR:

CENTRO DE INFORMÁTICA PROFESIONAL S.L.

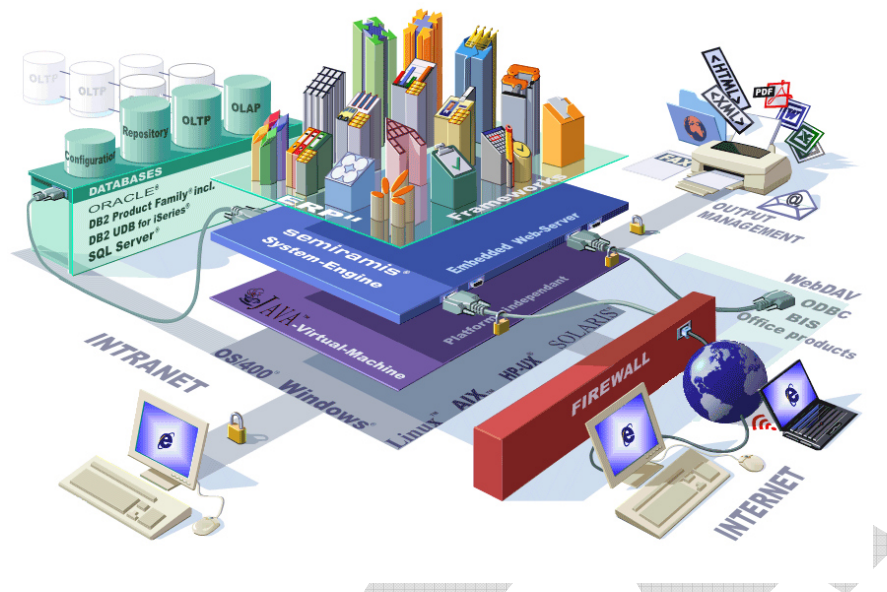
C/ URGELL, 100
08011 BARCELONA
TFNO: 93 426 50 87

C/ RAFAELA YBARRA, 10
48014 BILBAO
TFNO: 94 448 31 33

www.cipsa.net

RESERVADOS TODOS LOS DERECHOS. QUEDA PROHIBIDO TODO TIPO DE REPRODUCCIÓN TOTAL O PARCIAL DE ESTE MANUAL, SIN PREVIO CONSENTIMIENTO POR EL ESCRITOR DEL EDITOR

Fundamentos de Aplicaciones Web

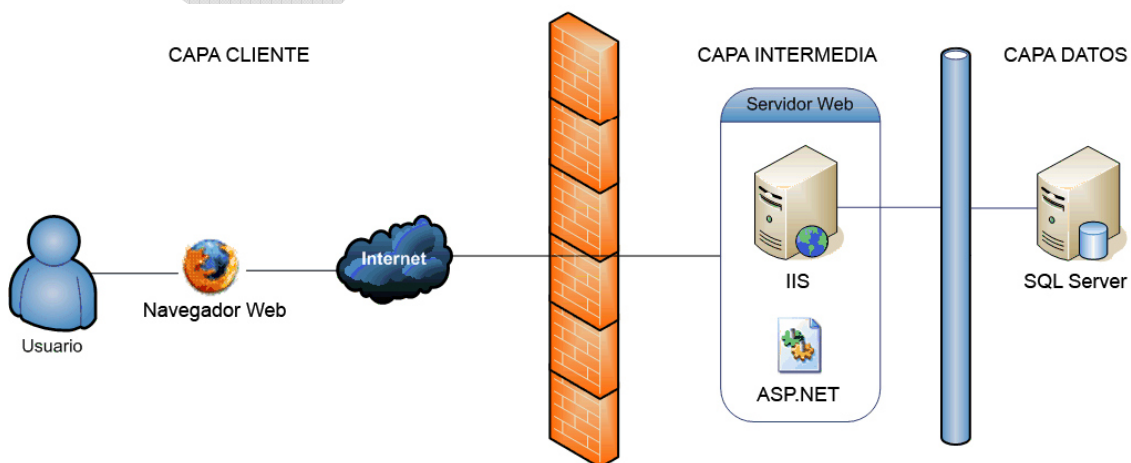


Una aplicación Web es un programa informático que se ejecuta parte en el ordenador del cliente y parte en un servidor web al que se accede mediante una red local o Internet.

Las aplicaciones web no requieren de instalación en el ordenador del cliente, ni dependen del sistema operativo instalado. Para su ejecución únicamente es necesario una conexión a la red/internet y disponer de un navegador web (*Internet Explorer, FireFox, Chrome... etc*). No dependen tampoco el dispositivo empleado: tablets, smartphones, smartTVs..., etc.

Capas de una aplicación web

Una aplicación Web puede ser utilizada por múltiples clientes al mismo tiempo a través de internet o una red local. Su estructura puede representarse en tres capas:

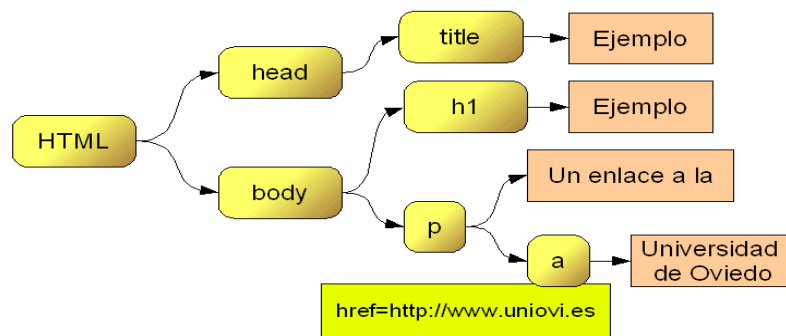


Capa Cliente

La capa cliente de una aplicación Web es la que se ejecuta en el navegador del usuario a través de páginas web. El navegador es un programa instalado en el equipo del cliente (sea un PC, Mac, Table/SmartPhone Android / Apple), que permite interactuar y navegar a través de páginas web empleando el protocolo de comunicaciones HTTP.

Una página Web es básicamente un documento de texto que contiene un código HTML que es interpretado y representado por el navegador web.

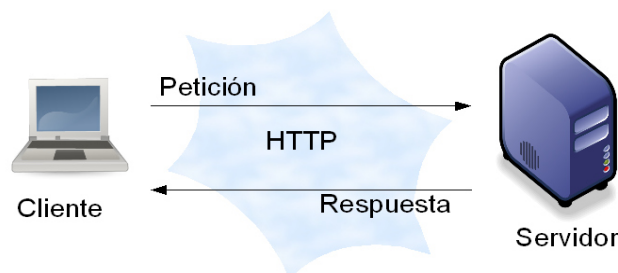
HTML es un lenguaje descriptivo de etiquetas que define el contenido y estructura de la página web. También se incluyen CSS y Javascript para definir el aspecto y la interacción. Estas etiquetas se organizan siguiendo un esquema jerárquico que organiza el contenido de la página.



Estructura de etiquetas base de documentos HTML / XHTML

El protocolo HTTP (*HyperText Transfer Protocol*) es el empleado para la comunicación entre el navegador del cliente y el servidor web. Sus funciones son las siguientes.

- **Envío de solicitud** (HTTP Request) → Solicitar una página al servidor web designado por la dirección o URL indicada en el navegador. También es posible el envío de datos introducidos previamente en formularios.
- **Recepción de respuesta** (HTTP Response) → Recepción de la página web enviada por el servidor como respuesta junto con sus contenidos (imágenes, sonidos, hojas de estilo, librerías de Javascript)



Representación visual de una petición y respuesta HTTP entre navegador y servidor

Ejemplo: Petición HTTP de un navegador solicitando a un servidor Web la página en la URL: `/localhost/hola.html`.

```
GET /localhost/hola.htm
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, .... , */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1m ... .NET CLR 3.0.04506.30
)
Host: localhost:80
Connection: Keep-Alive
```

Ejemplo: Respuesta HTTP del servidor Web con el código HTML de la página solicitada insertado.

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/5.1
X-Powered-By: ASP.NET
Date: Thu, 01 Nov 2007 17:54:34 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Mon, 02 Nov 2007 17:45:56 GMT
ETag: "04e9ace185fc51:bb6"
Content-Length: 130
<html>
    <body>
        <h1>Hello World</h1>
    </body>
</html>
```

El paquete de respuesta HTTP se compone de dos partes. Una parte inicial o *cabecera* con información sobre los datos enviados, seguida del *cuerpo / contenido*; que en este caso es el código HTML de la propia página.

Capa Cliente (o de negocio)

La capa intermedia es la que se ejecuta en el servidor Web al que acceden los usuarios a través de red o de Internet mediante su navegador.

Un *servidor Web* es un ordenador conectado a Internet con un *servicio Web* instalado.

Un *servicio web* atiende peticiones HTTP enviadas por los navegadores de los usuarios y envía las páginas web solicitadas como respuesta. Actualmente los servicio Web más conocidos son Internet Information Services de Microsoft para sistemas operativos Windows (IIS), y Apache de la Apache Software Foundation disponible para sistemas operativos Windows, Linux, Unix... etc.

Las páginas HTML deben alojarse en una ubicación determinada del servidor (*carpeta de publicación*) para ser accesibles a los usuarios. Al proceso de situar las páginas web en esta carpeta se le denomina *Publicar*. La publicación puede llevarse acabo localmente en el servidor copiando y pegando las páginas dentro de la carpeta de publicación, o remotamente subiendo las páginas al servidor mediante un servicio FTP o similar.

Capa de Datos

La capa de datos se encarga del almacenamiento permanente de la información manejada por la aplicación web. Esta información suele componerse de datos introducidos por usuarios y/o administradores de la aplicación que son mostrados y tratados como parte de su funcionamiento.

Para ello se emplean normalmente sistemas basados en *bases de datos relacionales*. Una *base de datos relacional* almacena grandes volúmenes de datos en un conjunto de tablas relacionadas que permiten realizar búsquedas eficientes.

La capa de datos puede comprender una o varias bases de datos gestionadas por un servidor gestor de base de datos (*SGBD*). Un *SGBD* administra múltiples bases de datos y permite el acceso y modificación de múltiples datos al mismo tiempo de manera eficiente y fiable. Actualmente existen múltiples SGBDs tales como *Microsoft SQL Server*, *MySQL*, *Oracle* ...etc.

Ejemplo

Supóngase una aplicación web de ventas de un comercio electrónico en la que un usuario cualquiera accede a la página de productos vendidos de una determinada categoría. El proceso al que daría lugar tal operación a través de las diferentes capas sería el siguiente:

Capa cliente	Capa intermedia	Capa datos
El usuario solicita la página de productos de una determinada categoría	El servidor recibe la solicitud, obtiene los datos de los productos de la categoría correspondiente y devuelve la página con los datos obtenidos.	El SGBD busca los productos de la categoría requerida en la base de datos de artículos y devuelve los datos a la capa intermedia

Supóngase que el usuario realiza la compra de un determinado producto:

Capa cliente	Capa intermedia	Capa datos
El usuario selecciona el producto e introduce sus datos de envío y cobro.	El servidor recibe los datos del usuario, los almacena en la base de datos para proceder a su envío y le devuelve una página indicando que el pedido ha sido registrado.	El SGBD almacena los datos del pedido incluyendo el producto solicitado y los datos de envío y cobro del usuario

Instalando un servidor de desarrollo

Antes de poder empezar a crear scripts PHP es necesario instalar un servidor de desarrollo para PHP.

Un servidor de desarrollo es un ordenador que hace de servidor web para probar y depurar las aplicaciones web que se están preparando. Habitualmente se emplean ordenadores a los que se accede dentro de una red local (no internet), o el propio ordenador en el que se programa.

Los componentes mínimos a instalar en un servidor de desarrollo PHP son:

- Un servicio web (*Apache*)
- El intérprete *PHP*
- Un servicio de base de datos (*MySQL*)
- Una aplicación web de gestión de base de datos (*PHPMyAdmin*)

Estos componentes pueden instalarse por separado o juntos instalando un entorno de desarrollo como XAMP / WAMP.

XAMPP



XAMPP es un entorno de desarrollo que integra todos los componentes necesarios para crear un servidor web PHP. Incluye los siguientes servicios y componentes:

- **Apache + PHP** (Servicio web con el interprete de *PHP* integrado)
- **MariaDB** (servicio de gestión de bases de datos libre basado en *MySQL*)
- **FileZilla** (servicio para transferir archivos remotamente mediante *FTP*)
- **Mercury** (servicio de correo electrónico)
- **Tomcat** (servicio de aplicaciones web con Java)

Es mucho más rápido y fácil de instalar y configurar que instalando los componentes por separado.

Este puede descargarse gratuitamente desde la web oficial:

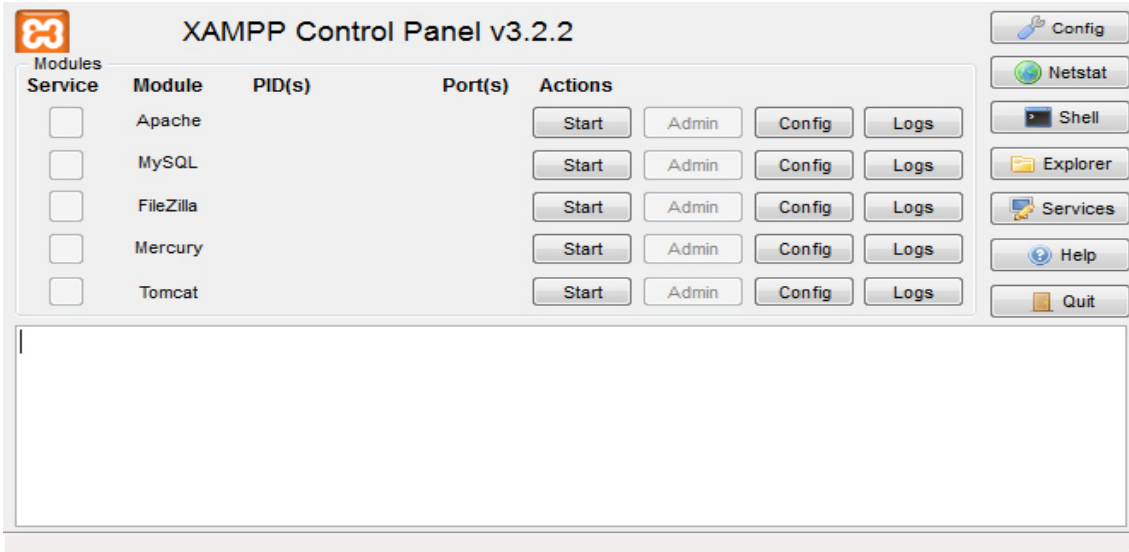
<https://www.apachefriends.org/es/index.html>

Instalación y funcionamiento de XAMPP

La instalación de XAMPP guarda de manera predeterminada todos los archivos en la carpeta:

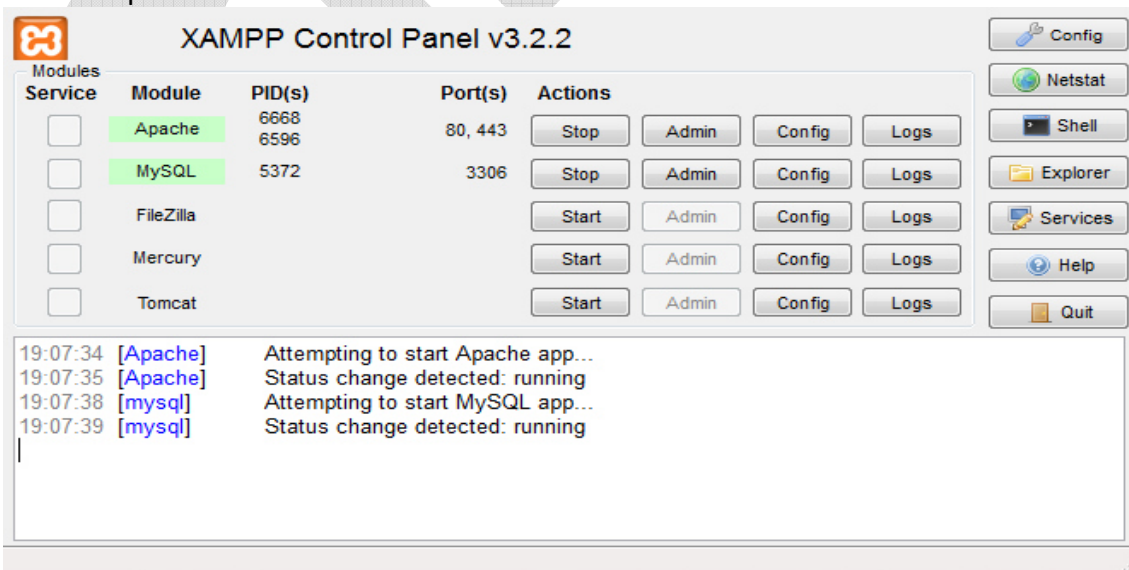
c:\xampp

Una vez instalado, la configuración y activación de los diferentes servicios puede hacerse desde el panel de control que se muestra al hacer clic sobre su acceso directo:



Panel de control de XAMPP

Cada uno de los servicios pueden activarse por separado al hacer clic sobre el botón **“Start”**. Se muestra entonces un cuadro verde indicando los puertos TCP/IP por los que atienden peticiones.



Panel de control con servidor Apache y MySQL activados.

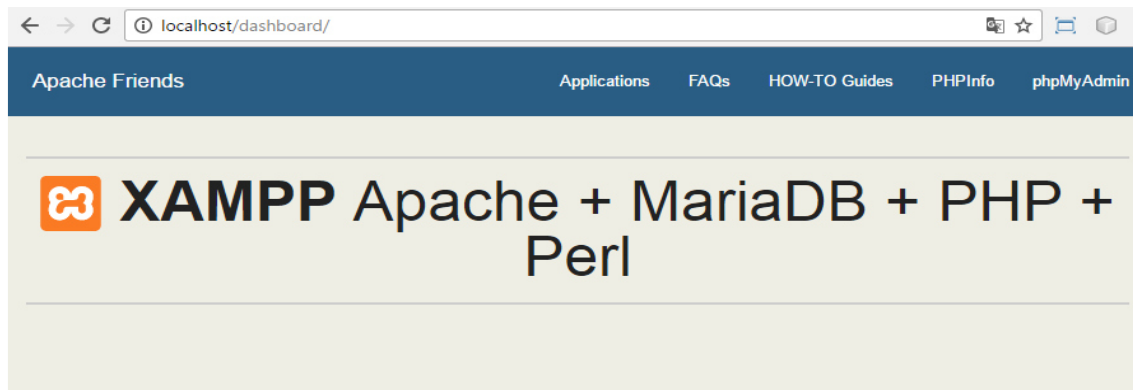
Para detener los servicios sólo es necesario pulsar el botón **“Stop”** correspondiente.

Página de presentación de XAMPP

Una vez instalado el WAMP puede probarse su funcionamiento accediendo con cualquier navegador a la dirección:

<http://localhost>

El resultado debería ser el siguiente:



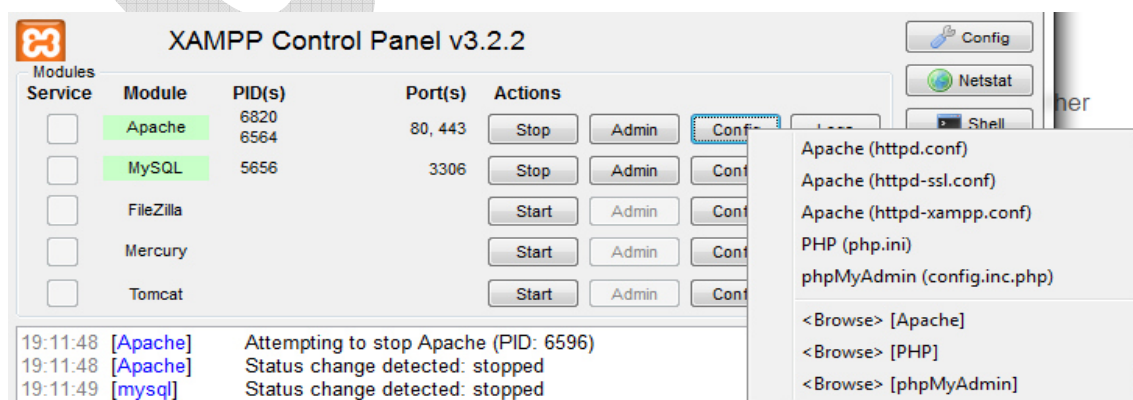
Welcome to XAMPP for Windows 5.6.15

You have successfully installed XAMPP on this system! Now you can start using Apache, MariaDB, PHP and other components. You can find more info in the FAQs section or check the HOW-TO Guides for getting started with PHP applications.

Start the XAMPP Control Panel to check the server status.

Archivos de configuración

El servicio de Apache y el intérprete de PHP se configuran mediante archivos de texto con diversas opciones. Estos pueden examinarse pulsando el botón **“Config”** y seleccionando el archivo correspondiente:



Menu contextual de archivos de configuración de XAMPP

Archivo de configuración de Apache (*httpd.conf*)

La configuración del servicio Apache se lleva acabo editando el archivo *httpd.conf* que contiene los parámetros de funcionamiento del servicio. Se trata de un archivo de texto donde que puede abrirse empleando el *Notepad* donde se encuentran todos los parámetros de funcionamiento utilizados por Apache.

```
#
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# Do not add a slash at the end of the directory path. If you point
# ServerRoot at a non-local disk, be sure to specify a local disk on the
# Mutex directive, if file-based mutexes are used. If you wish to share the
# same ServerRoot for multiple httpd daemons, you will need to change at
# least PidFile.
#
ServerRoot "C:/xampp/apache"

#
# Mutex: Allows you to set the mutex mechanism and mutex file directory
# for individual mutexes, or change the global defaults
#
# Uncomment and change the directory if mutexes are file-based and the default
# mutex file directory is not on a local disk or is not appropriate for some
# other reason.
#
# Mutex default:logs

#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
#Listen 12.34.56.78:80
Listen 80
```

Algunas líneas que comienzan con el carácter # contienen información y parámetros deshabilitados que no son utilizados por el servicio. Estos parámetros pueden rehabilitarse eliminado el carácter #.

Algunos de los parámetros globales más importantes son los siguientes:

→ **ServerRoot** "<ruta>"

Indica el directorio de instalación del servicio Apache. Sólo debería modificarse si se modifica el nombre de alguna carpeta o se cambia el servicio de directorio. Por defecto:

```
ServerRoot "C:/xampp/apache"
```

→ **Listen** <ip>:<puerto>

Listen: Indica el puerto y la dirección IP que escucha el servicio. Este parámetro puede modificarse para conseguir que el servicio sólo escuche las peticiones entrantes por una IP concreta. El puerto se modifica si ya existe otro servicio web escuchando por el puerto 80. En tal caso suele asignarse el puerto 8080.

Por defecto el servicio escucha el puerto 80 en todas las conexiones de red del equipo.

Listen 80

→ DocumentRoot "<ruta>"

Indica la ruta de la carpeta de publicación donde se almacenan las páginas web. Por defecto esta carpeta es *htdocs* presente en la carpeta de instalación del servicio. El parámetro puede modificarse si se cambia el nombre de la carpeta o se desean almacenar las páginas en otra carpeta distinta:

```
DocumentRoot "C:/xampp/htdocs"
```

→ DirectoryIndex <pag_inicio_1> <pag_inicio2> <pag_inicio3>...

Indica la secuencia de páginas de inicio que el servidor busca y muestra cuando no se solicita ninguna en concreto. Si el servicio recibe una petición sin indicar ninguna página se muestra la indicada en primer lugar. Si no se encuentra ninguna página con el nombre indicado en primer lugar, se pasa a buscar con el segundo, y así sucesivamente. En caso de no encontrarse ninguna página con los nombres indicados se envía al cliente un mensaje de error.

Por defecto la página de inicio predeterminada debe llamarse *index.html*.

```
DirectoryIndex index.php index.pl index.cgi index.asp index.shtml index.html index.htm \
default.php default.pl default.cgi default.asp default.shtml default.html default.htm \
home.php home.pl home.cgi home.asp home.shtml home.html home.htm
```

Además de los parámetros globales, existen parámetros de configuración específicos para restringir el acceso a las páginas contenidas en ciertas carpetas. Estas directivas se colocan dentro de etiquetas **<Directory>..<Directory>**:

```
<Directory "C:/xampp/htdocs">
#
# Possible values for the Options directive are "None", "All",
# or any combination of:
#   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
# The Options directive is both complicated and important. Please see
# http://httpd.apache.org/docs/2.4/mod/core.html#options
# for more information.
#
Options Indexes FollowSymLinks Includes ExecCGI
#
# AllowOverride controls what directives may be placed in .htaccess files.
# It can be "All", "None", or any combination of the keywords:
#   AllowOverride FileInfo AuthConfig Limit
#
AllowOverride All
#
# Controls who can get stuff from this server.
#
Require all granted
</Directory>
```

Publicación de una página en XAMPP

La carpeta de publicación predefinida tras la instalación de XAMPP es:

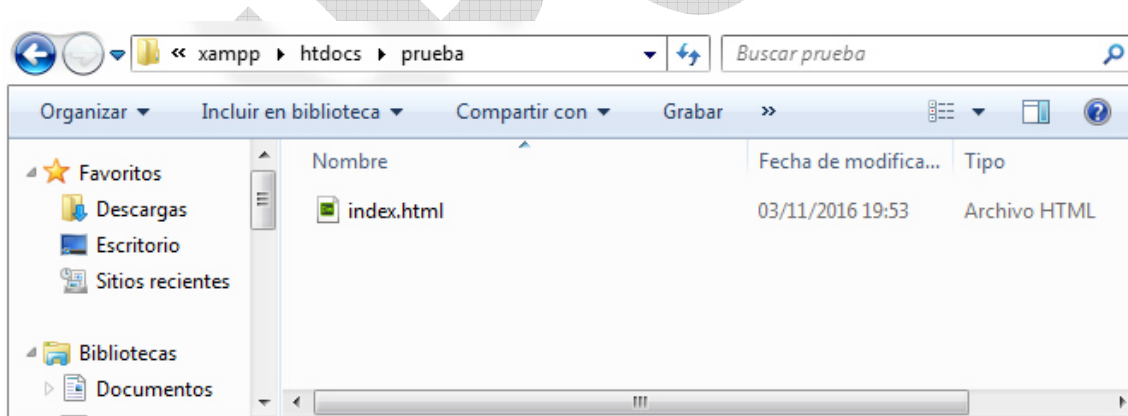
C:\xampp\htdocs

Para publicar una página web debemos crear primeramente una carpeta donde alojar nuestra página web. Por ejemplo; podemos crear la carpeta ***"prueba"***.

A continuación podemos crear una página web empleando cualquier aplicación como Dreamweaver, NetBeans, o el mismo bloc de notas con el siguiente contenido:

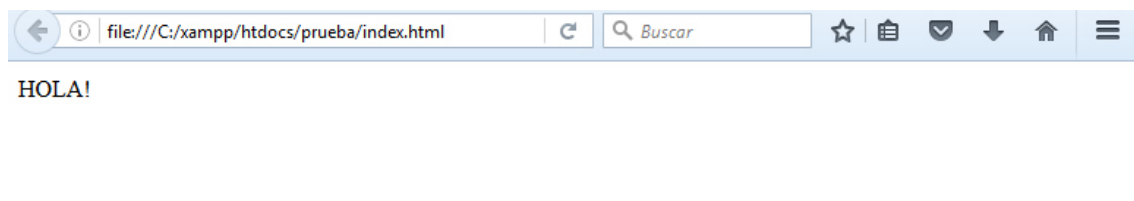
```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title></title>
<meta charset="utf-8" />
</head>
<body>
<div>
HOLA!
</div>
</body>
</html>
```

Una vez creada, la guardamos con el nombre *index.html*, dentro de la carpeta *prueba*.



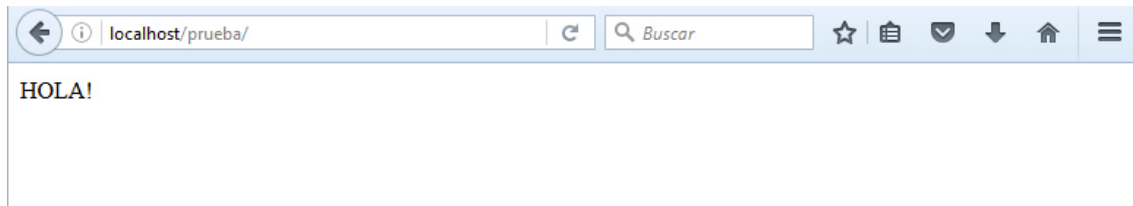
La página podemos verla ahora en el navegador de dos modos:

Accediendo a la página web como archivo indicando su ubicación local en el disco del ordenador:



Accediendo a través del servicio web indicando la URL:

<http://localhost/prueba>



Aunque no se indica la página solicitada (<http://localhost/prueba/index.html>), el servicio devuelve la página *index.html* ya que es uno de los nombres indicados como página de inicio en la directiva **DirectoryIndex** del archivo de configuración de apache.

El intérprete de PHP

XAMPP integra en el servicio de Apache el intérprete de PHP como un módulo. La configuración de Apache hace que los archivos con extensión *.php* sean procesados por el módulo PHP, y que el código HTML resultante sea enviado como respuesta al navegador del usuario.

Publicación de una página PHP

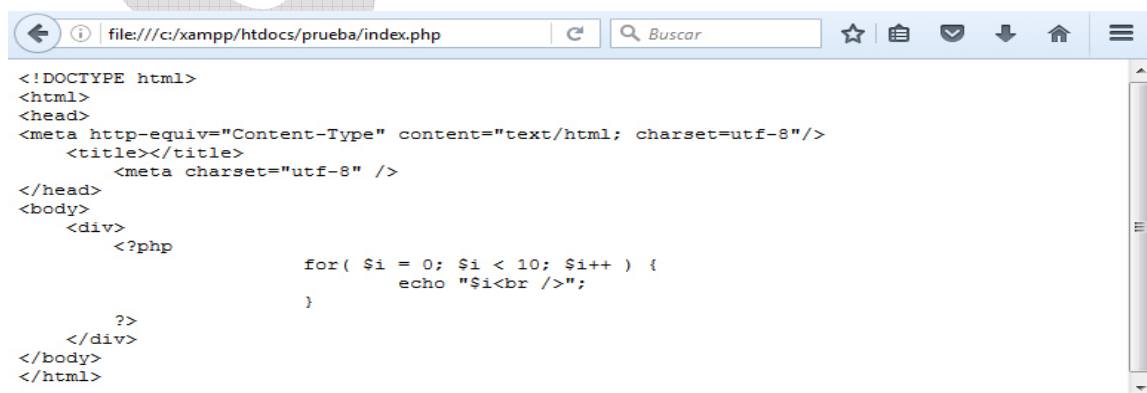
Una página web PHP es un simple archivo de texto que puede editarse de igual modo que una página web HTML.

A modo de ejemplo, modifíquese el código de la página *index.html* del ejemplo anterior añadiendo el siguiente código:

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title></title>
  <meta charset="utf-8" />
</head>
<body>
  <div>
    <?php
      for( $i = 0; $i < 10; $i++ ) {
        echo "$i<br />";
      }
    ?>
  </div>
</body>
</html>
```

A continuación guardamos el archivo pero modificando la extensión *.html*, por *.php*.

Si ahora examinamos la página con el navegador accediendo localmente al archivo observaremos lo siguiente:

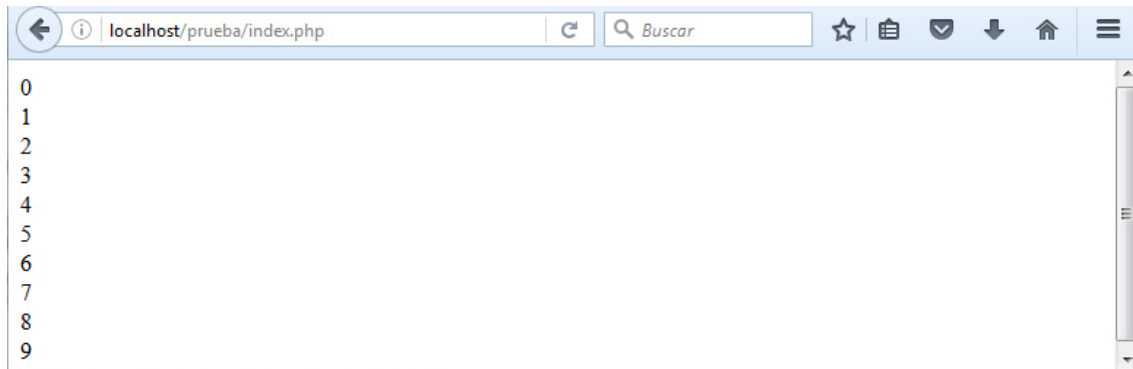


```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title></title>
  <meta charset="utf-8" />
</head>
<body>
  <div>
    <?php
      for( $i = 0; $i < 10; $i++ ) {
        echo "$i<br />";
      }
    ?>
  </div>
</body>
</html>
```

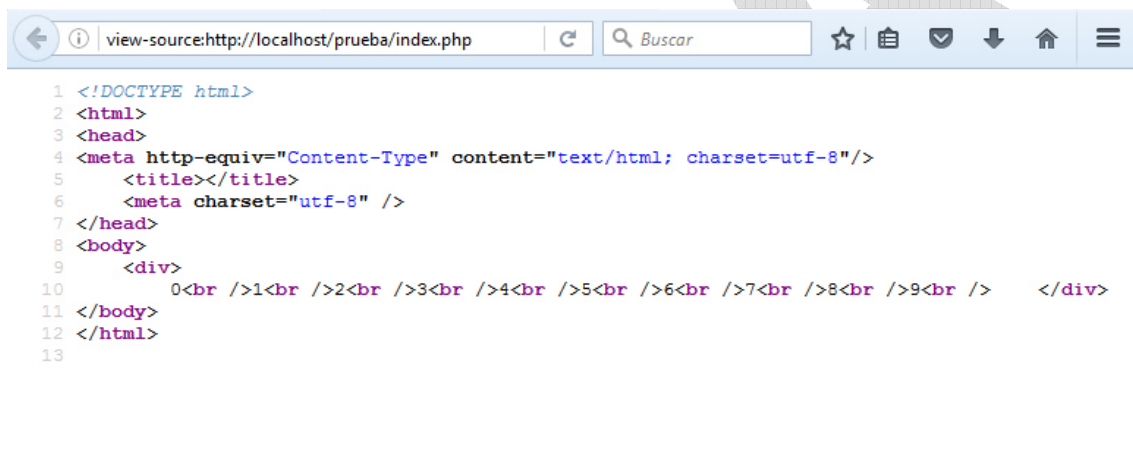
El navegador no sabe interpretar las páginas de PHP, de modo que muestra directamente el código de la misma.

Programación Web en PHP

Sin embargo, si accedemos a la página a través del servicio web indicando la URL: <http://localhost/prueba/index.php> el resultado es bien distinto:



Podría pensarse que en este caso el navegador web ha interpretado el código PHP, sin embargo; si examinamos el código HTML de la página comprobamos lo siguiente:



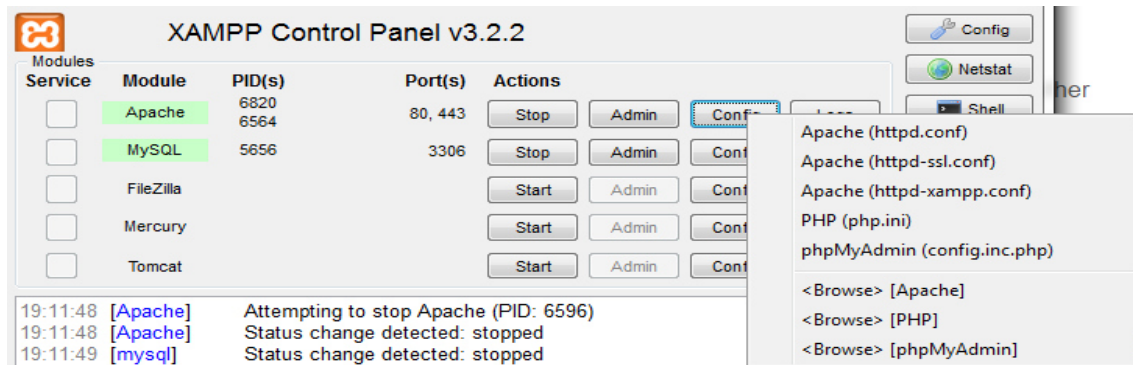
El código HTML de la página no muestra ni rastro de código PHP.

El código PHP incluido en el código original de la página *index.php*, fue procesado por el intérprete de PHP al solicitarse la página. El código HTML resultante de su ejecución es retornado al usuario como respuesta a su petición, y eso es exactamente lo que recibe el navegador:



Archivo de configuración de PHP (*php.ini*)

El intérprete de PHP consta de un archivo de configuración. En este caso el archivo se llama *php.ini* y se encuentra en la raíz de la carpeta de instalación de PHP. Este archivo es accesible desde el menú de opciones que se despliega al pulsar el botón “**Config**”.



El comando ***phpinfo()*** de PHP, genera una tabla HTML con todos los datos del motor de PHP en funcionamiento y su configuración actual.

```
; allow_call_time_pass_reference
;   Default Value: On
;   Development Value: off
;   Production Value: off

; display_errors
;   Default Value: On
;   Development Value: On
;   Production Value: off

; display_startup_errors
;   Default Value: off
;   Development Value: On
;   Production Value: off

; error_reporting
;   Default Value: E_ALL & ~E_NOTICE & ~E_STRICT & ~E_DEPRECATED
;   Development Value: E_ALL
;   Production Value: E_ALL & ~E_DEPRECATED & ~E_STRICT
```

Se trata de un archivo de texto convencional con multitud de comentarios y directivas de configuración. Estas directivas controlan desde la forma en que se ejecuta el código PHP, hasta aspectos relativos a la seguridad, el registro y visualización de errores, y la configuración de las extensiones.

Las líneas precedidas por un signo de punto y coma (;) son comentarios y parámetros deshabilitados de ejemplo. Para configurar las directivas debe emplearse la siguiente sintaxis:

directiva = valor

El nombre de la directiva es sensible a mayúsculas/minúsculas. El valor puede ser un número, una cadena de texto (entre comillas), o una constante tal como *On*, *Off*, *True*, *False*, *Yes*, *No*, *None*, *E_ALL*..., etc, según los valores admitidos por la directiva.

Algunas de las directivas más relevantes son:

display_errors

```
; display_errors
; Default Value: On
; Development Value: On
; Production Value: Off
```

Esta directiva controla si PHP muestra al usuario mensajes de error cuando el código PHP no se ejecuta correctamente. En un servidor de desarrollo puede resultar útil, pero debe deshabilitarse en un servidor de producción ya que la información mostrada podría aprovecharse para un uso malintencionado.

error_reporting

```
; error_reporting
; Default Value: E_ALL & ~E_NOTICE
; Development Value: E_ALL | E_STRICT
; Production Value: E_ALL & ~E_DEPRECATED
```

Esta directiva permite filtrar qué errores y notificaciones son mostrados al usuario de la aplicación. Los posibles valores son:

- Valor predeterminado: `E_ALL & ~E_NOTICE` → Muestra todos los tipos de errores salvo las advertencias y notificaciones.
- Valor para servidor de desarrollo: `E_ALL | E_STRICT` → Muestra errores y sugerencias de código para asegurar la compatibilidad del código entre diferentes versiones de PHP.
- Valor para servidor de producción: `E_ALL & ~E_DEPRECATED` → Muestra errores pero no las notificaciones asociadas a funciones obsoletas.

Los valores que pueden asignarse a la directiva están indicados en el propio fichero de configuración:

<code>E_ALL</code>	- All errors and warnings (includes <code>E_STRICT</code> as of PHP 6.0.0)
<code>E_ERROR</code>	- fatal run-time errors
<code>E_RECOVERABLE_ERROR</code>	- almost fatal run-time errors
<code>E_WARNING</code>	- run-time warnings (non-fatal errors)
<code>E_PARSE</code>	- compile-time parse errors
<code>E_NOTICE</code>	- run-time notices (these are warnings which often result from a bug in your code, but it's possible that it was intentional (e.g., using an uninitialized variable and relying on the fact it's automatically initialized to an empty string))
<code>E_STRICT</code>	- run-time notices, enable to have PHP suggest changes to your code which will ensure the best interoperability and forward compatibility of your code
<code>E_CORE_ERROR</code>	- fatal errors that occur during PHP's initial startup
<code>E_CORE_WARNING</code>	- warnings (non-fatal errors) that occur during PHP's initial startup
<code>E_COMPILE_ERROR</code>	- fatal compile-time errors
<code>E_COMPILE_WARNING</code>	- compile-time warnings (non-fatal errors)
<code>E_USER_ERROR</code>	- user-generated error message
<code>E_USER_WARNING</code>	- user-generated warning message
<code>E_USER_NOTICE</code>	- user-generated notice message
<code>E_DEPRECATED</code>	- warn about code that will not work in future versions of PHP
<code>E_USER_DEPRECATED</code>	- user-generated deprecation warnings

log_errors

```
; log_errors
; Default Value: Off
; Development Value: On
; Production Value: On
```

Esta directiva permite el registro de los errores en un archivo cuyo nombre por defecto es *php_errors.log*. Este nombre puede modificarse cambiando el valor asignado al parámetro *error_log* dentro del archivo de configuración.

```
; Log errors to specified file. PHP's default behavior is to leave this value
; empty.
; http://php.net/error-log
; Example:
;error_log = php_errors.log
; Log errors to syslog (Event Log on NT, not valid in Windows 95).
;error_log = syslog
```

Librerías de PHP

El intérprete de PHP también es modular al igual que PHP y permite agregarle librerías para añadir funciones adicionales a PHP. Las librerías integradas así como la configuración de sus directivas pueden consultarse en la página de información de PHP (*phpinfo*).

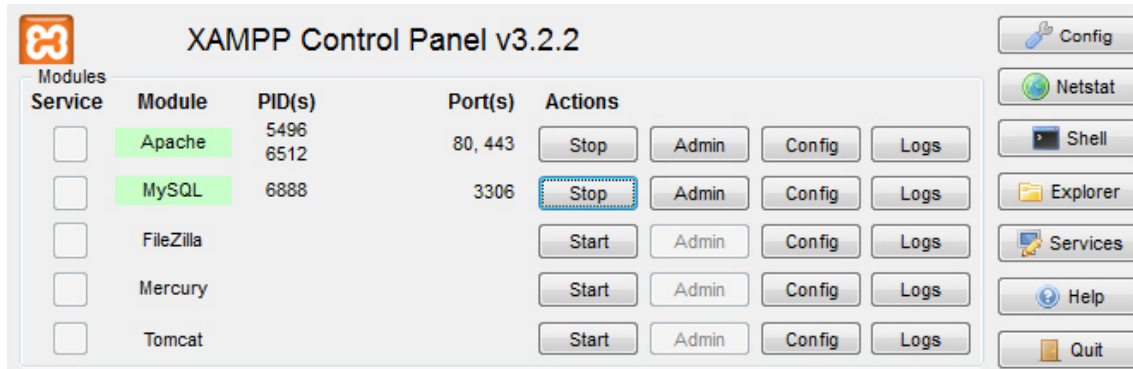
Con el entorno de desarrollo XAMPP, esta página puede consultarse indicando la dirección: <http://localhost/dashboard/phpinfo.php>

System	Windows NT ANGEL-LBNID1MGX 5.2 build 3790 (Windows Server 2003 Enterprise Edition Service Pack 1) i586
Build Date	Jul 21 2010 20:00:47
Compiler	MSVC6 (Visual C++ 6.0)
Architecture	x86
Configure Command	cscrip\inologo configure.js "--enable-snapshot-build" "--disable-isapi" "--enable-debug-pack" "--disable-isapi" "--without-mssql" "--without-pdo-mssql" "--without-pi3web" "--with-pdo-oci=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8=D:\php-sdk\oracle\instantclient10\sdk,shared" "--with-oci8-11g=D:\php-sdk\oracle\instantclient11\sdk,shared" "--enable-object-out-dir=.\obj" "--enable-com-dotnet" "--with-mcrypt=static"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\WINDOWS
Loaded Configuration File	(none)

Página de información de PHP generada por el comando phpinfo()

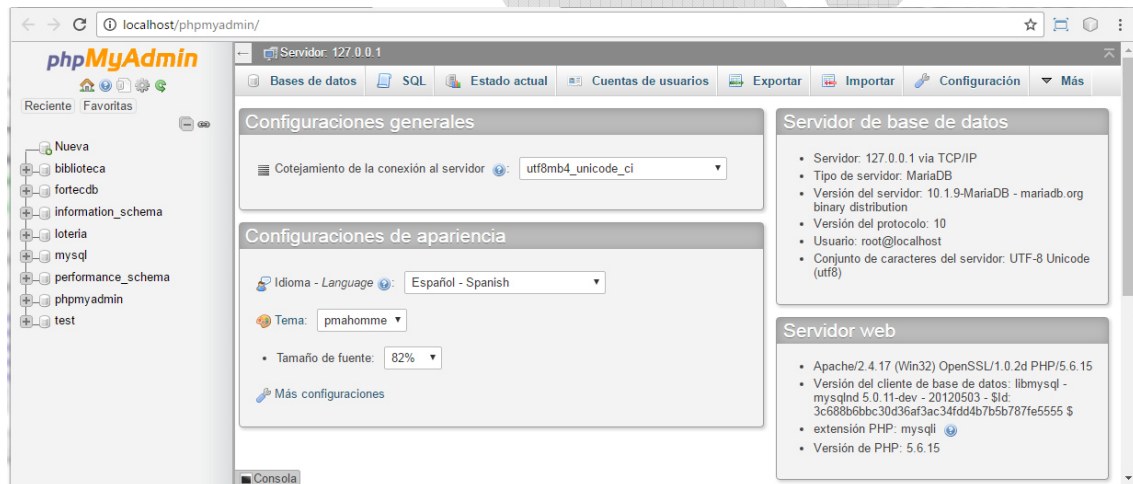
La herramienta *PHPMYAdmin*

PHPMYAdmin es una aplicación web programada en PHP que permite acceder al servicio de base de datos MySQL instalado como parte de XAMPP. Este servicio se inicia de manera independiente al servicio web de Apache + PHP, pulsando en el botón **“Start”** correspondiente en el panel de control.



Vista del panel de control de XAMPP iniciando el servicio de MySQL

El botón **“Admin”** abre el navegador y muestra la página *phpMyAdmin*.



Esta aplicación muestra las bases de datos existentes en el servicio de MySQL, y permite crear otras nuevas además de modificar su estructura y datos almacenados.

El uso de esta herramienta se verá más adelante en el curso.

Nota sobre la configuración en hospedajes comerciales

Los archivos de configuración de Apache y PHP vistos hasta ahora no están disponibles por motivos de seguridad cuando se emplea un servicio de hospedaje en internet.

En tales casos, la modificación de la configuración debe realizarse mediante el panel de control proporcionado por la propia empresa de hospedaje, o mediante archivos adicionales de permisos con extensión *.htaccess* de Apache.

(*) La configuración de propiedades de directorios mediante ficheros *.htaccess* se verá más adelante en el curso.

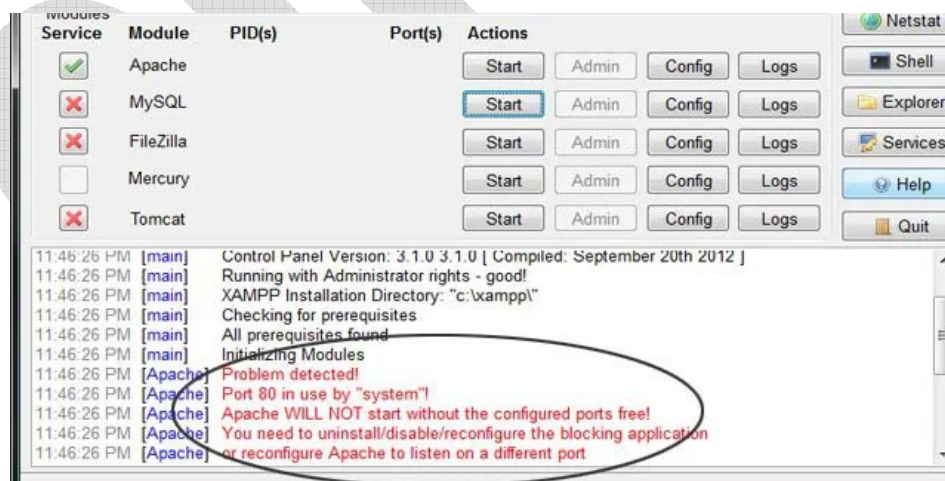
Problema de conflicto de puertos.

Todos los servicios destinados a la red o internet emplean unos puertos específicos para permitir la comunicación entre los servidores y los ordenadores de los clientes.

Los servidores web emplean por defecto el puerto 80 que está destinado a la solicitud de páginas web. Otros tipos de servicios emplean otros puertos:

MySQL 3306
FileZilla → 21

Sólo puede haber un servicio utilizando un puerto, de modo que si intentamos iniciar un servicio que emplea un puerto ya usado por otro servicio en funcionamiento, se producirá un error. Esto puede causar que algunos servicios de XAMPP no se inicien correctamente si ya existe otro servicio empleando el mismo puerto:



Panel de control de XAMPP mostrando problema de conflicto de puertos

Para solucionar este problema podemos pulsar el botón **"Netstat"** que muestra una ventana con los servicios en funcionamiento y los puertos empleados. Debemos localizar el servicio en conflicto y desactivarlo para poder iniciar XAMPP correctamente.