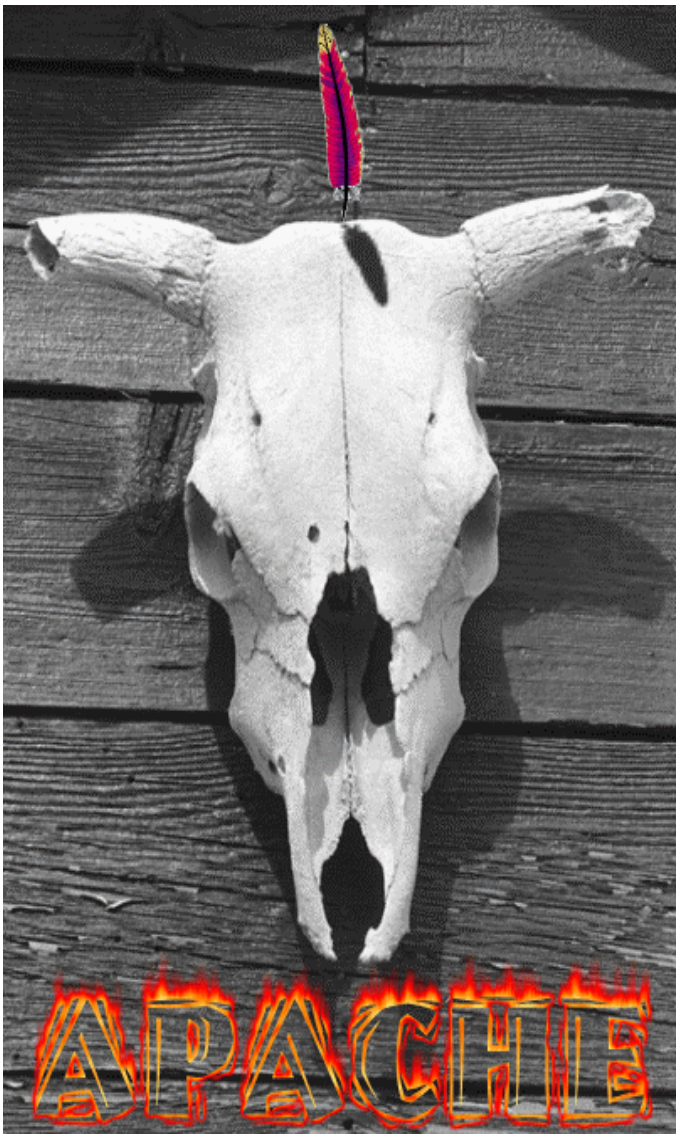
Este libro está pensado para aquellos desarrolladores que necesitan tener unos conocimientos básicos de instalación y administración del servidor Web Apache.

Existen una serie de características que convierten a Apache en uno de los servidores web más utilizados, como son el tener el código fuente abierto, mantener una evolución rápida y continuada de versiones, poder ser utilizado por desarrolladores de cualquier plataforma, y además, ser gratuito.

Apache es un servidor web multiplataforma, que permite indexación de directorios, uso de sobrenombres con las carpetas, informes configurables sobre errores http, ejecución de programas CGI y que además admite la última versión del protocolo http/1.1.



SERVIDORES PARA INTERNET CON APACHE HTTPSERVER

FRANCISCO JAVIER EGEA

GRUPO EIDOS

ISBN 84-88457-19-7

Servidores para Internet con Apache HttpServer

Fracisco Javier Egea

Responsable editorial
Paco Marín (fmarin@eidos.es)

Coordinación de la edición
Antonio Quirós (aquiros@eidos.es)

Autoedición
Magdalena Marín (mmarin@eidos.es)
Francisco Javier Egea (fjegea@eidos.es)

Grupo EIDOS
C/ Téllez 30 Oficina 2
28007-Madrid (España)
Tel: 91 5013234 Fax: 91 (34) 5017824
www.grupoeidos.com/www.eidos.es
www.LaLibreriaDigital.com

Índice

1

Introducción a Apache

Con independencia de si el lector es un administrador de sistemas o un desarrollador de software, es necesario que conozca que nos estamos refiriendo a algo más que un simple servidor web. El proyecto Apache se realiza en un gran esfuerzo de colaboración de desarrollo de software poniendo a nuestra disposición el código fuente de un servidor HTTP. El proyecto se realiza por una serie de voluntarios localizados alrededor del mundo, utilizando Internet como herramienta de comunicación, organización y desarrollo, así como toda la documentación relacionada. A estos voluntarios se les conoce como el Grupo Apache. Además cientos de usuarios siguen contribuyendo con ideas código y documentación al proyecto

Orígenes de Apache

En febrero de 1.995 el servidor web más utilizado era el creado por el NCSA (National Center for Super Computing Applications) en la Universidad de Illinois. Sin embargo, este proyecto cayó pronto en olvido y muchos webmasters habían desarrollado sus propias extensiones encontrándose con muchos problemas que se iban resolviendo poco a poco. Esto originó que un grupo de estos webmasters a través del correo electrónico se reunieran para coordinar los cambios que iban realizando a modo de parches, compartiendo toda la información creando un punto de encuentro como lista de envío en California Bay Area, naciendo así a finales de febrero el Grupo Apache, formado por ocho miembros: Brian Behlendorf, Roy T. Fielding, Rob Hartill, David Robinson, Cliff Skolnick, Randy Terbush, Robert S. Thau, Andrew Wilson. Hay que señalar también las contribuciones adicionales de: Eric Hagberg, Frank Peters, Nicolas Pioch.

Durante este tiempo NCSA reiniciaba su propio desarrollo, con Brandon Long y Beth Frank, uniéndose a mediados de marzo al Grupo Apache como miembros honorarios para que los dos proyectos pudieran compartir ideas y dificultades conjuntamente.

La primera descarga oficial de Apache se produjo en abril de 1.995.

Desde ese momento el producto resultante fue sufriendo continuas modificaciones, adaptándose a la mayor parte de los sistemas operativos. Fueron continuos sus cambios de funcionamiento y de planteamiento. En menos de un año desde su aparición el servidor Apache pasó a ser el servidor número uno en Internet.

El servidor más utilizado

Uno de los grandes argumentos que utilizan los usuarios de Apache es que es el servidor mas utilizado en la web a nivel mundial. Según un estudio realizado por la consultoría de Internet Netcraft, Apache es usado en un 40% mas que su mas inmediato competidor actualmente (Internet Information Server de Microsoft).

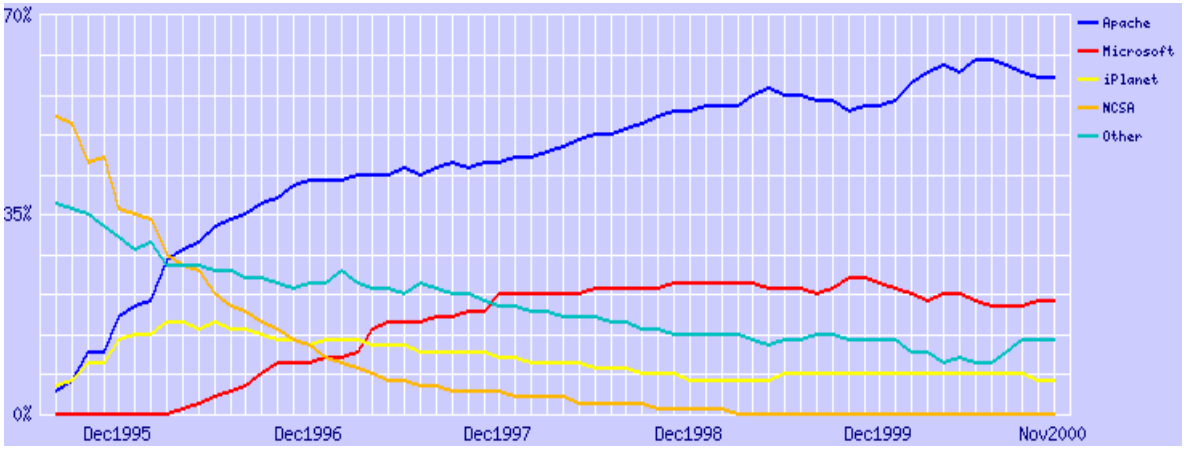


Figura 1 Cuota de mercado de los servidores web desde Agosto de 1995 hasta Noviembre de 2000

Como verá el lector en el cuadro anterior, el aumento de utilización de Apache es espectacular, al contrario que sus competidores, que o bien se mantienen, o bajan de manera notoria. Todo esto ocurre por una serie de circunstancias que trataremos de explicar de aquí en adelante.

Descripción de Apache

Existen una serie de características que convierten a Apache en uno de los servidores web más utilizados, como son el tener el código fuente abierto, mantener una evolución rápida y continuada de versiones, poder ser utilizado por desarrolladores de cualquier plataforma, y además, es gratuito.

Apache es un servidor web multiplataforma, que permite indexación de directorios, uso de sobrenombres con las carpetas, informes configurables sobre errores http, ejecución de programas CGI y que además admite la última versión del protocolo http/1.1.

Es un servidor que se basa en una configuración de un sistema de ficheros, pues no dispone de interfaz gráfico alguno. El lector conocerá más adelante que inicialmente este sistema de ficheros a pasado de tres a un fichero, lo único que necesitaremos será un editor de textos (vi, edit, notepad, etc...)

Una característica importante a señalar es que Apache permite trabajar con servidores virtuales tanto con direcciones IP así como con nombres virtuales. También se podría convertir nuestro servidor en un servidor Proxy.

En todo momento, a través de un explorador web, se podría conocer el estado de nuestro servidor, pues tiene registros configurables para guardar dicho estado, así como poder registrar las acciones de los usuarios.

Además de CGI, Apache puede trabajar con otros lenguajes de respuesta del servidor como Perl y Java (servlets) siempre y cuando se añadan los módulos necesarios en el fichero de configuración.

Arquitectura

Este servidor web, se compone de un núcleo, creado desde el código de Apache, y una serie de módulos adicionales que son los que van a definir la manera de trabajar del mismo. Muchos de estos módulos ya vienen compilados en la instalación predeterminada, pudiendo añadir más funcionalidad a nuestro servidor, o bien eliminando aquellos que no se van a utilizar. Por ejemplo, si se quiere trabajar con servlets se debería añadir el módulo mod_jserv .

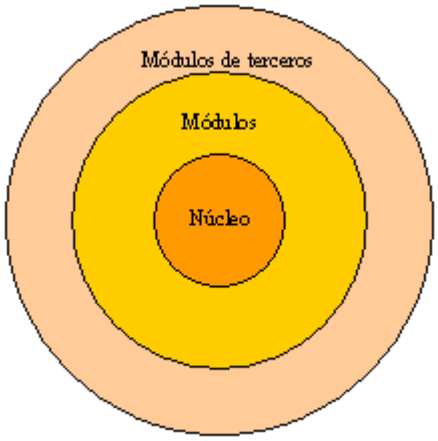


Figura 2. Arquitectura interna de Apache

Esta estructura da mucha flexibilidad a la hora de programar un sitio web, pues se puede llegar a tener un servidor a medida según las necesidades que se tengan.

El interfaz gráfico de Apache

Actualmente no existe por parte del Grupo Apache un interfaz gráfico para trabajar con este servidor, pero el lector debe saber que existen en proyecto algunos desarrollos de terceros que ya están funcionando y que se podrían obtener también a través de la web. No se descarta que próximamente se disponga de este interfaz de una manera oficial por parte del Grupo, aunque por la evolución seguida

por sus desarrolladores, no es de momento lo que más les preocupe. La Figura 3 muestra un ejemplo de estos programas que facilitarían la labor a cualquiera de los administradores del servidor Apache.

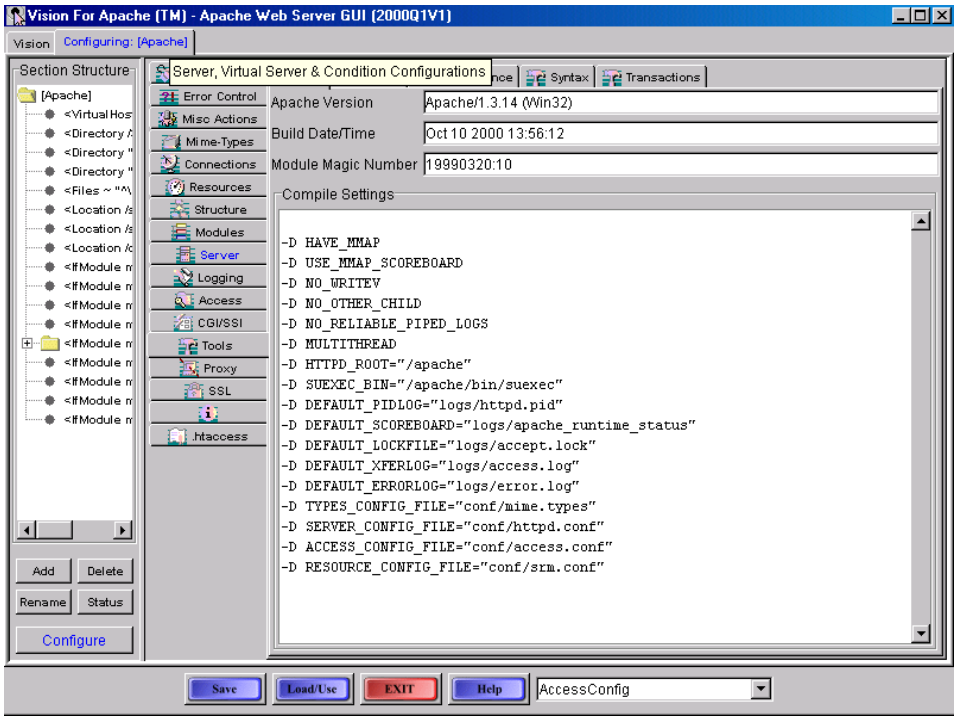


Figura 3. Página de configuración de Vision for Apache

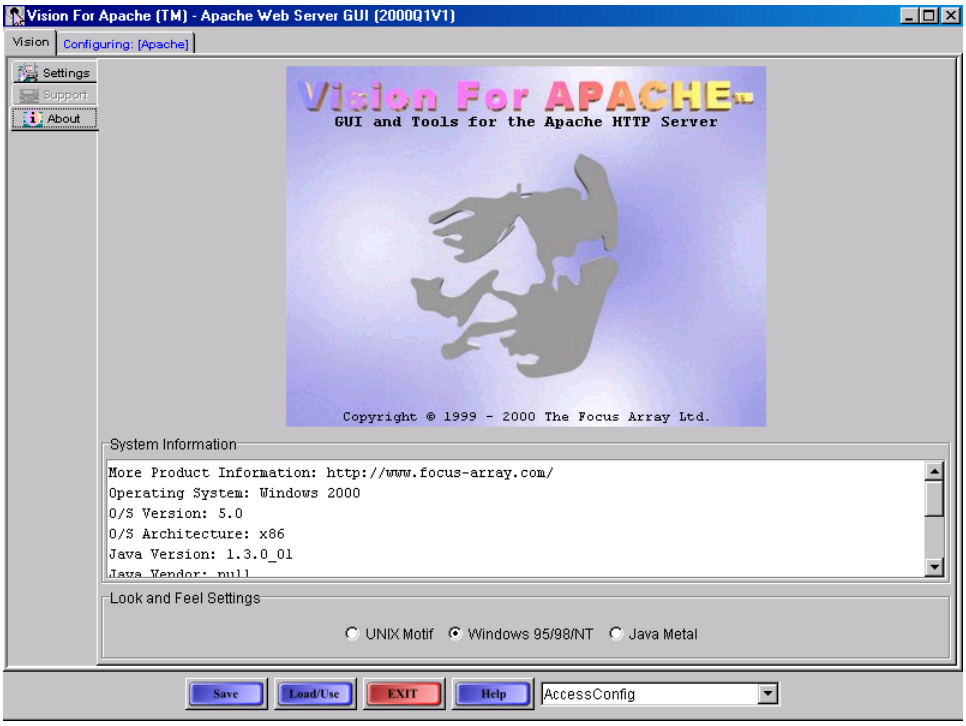


Figura 4. Página Acerca de Vision for Apache

2

Instalación del servidor

Cuando el lector se decida por fin a utilizar Apache, lo primero con lo que debe enfrentarse es el obtener el producto para instalarlo. Ya se sabe que una de las ventajas de utilizar este servidor es que es gratuito, por lo tanto no habrá que preocuparse de encontrar un distribuidor para obtenerlo. Otra de las ventajas con las que el lector se encontrará será que el Grupo Apache dispone de un lugar común donde, además de servir como centro de trabajo entre los distintos miembros del grupo, se podrá seleccionar el código fuente, o bien, los archivos binarios de Apache necesarios para cada sistema operativo en el que se quiera instalar. De hecho, lo más recomendable para conseguir los fuentes es siempre hacerlo desde este lugar, teniendo así garantizado que el código está totalmente revisado, y no utilizar otros medios de descarga.

Requisitos del sistema

Es posible que el lector no tenga problemas a la hora de instalar el software de Apache, pues los equipos informáticos han evolucionado más rápidamente que las necesidades que se precisan. A pesar de ello, se deben conocer algunos aspectos antes de comenzar una instalación.

El hardware siempre será el necesario para que nuestro sistema operativo funcione. No va a tener la misma necesidad un equipo funcionando con Windows 95 (98,ME), otro con Windows 2000, que los que estén funcionando con Linux o con Unix. Cuando instalamos Apache, son necesarios tener libres en el equipo de 6 a 10 MB. Un equipo con 8MB de RAM será mas que suficiente, pues ya se ha dicho que el servidor no consume muchos recursos. De todas maneras, si que se debe tener en cuenta que los archivos de registro que genera Apache, consumen unos 80 bytes por entrada, es decir, una web con unas 10.000 entradas diarias haría que ciertos archivos crecieran, en dicho intervalo de tiempo, 8MB.

Descarga de Apache

Ya hemos comentado que el servidor Apache es gratuito. El software lo encontraremos en www.apache.org . Siempre deberemos buscar los fuentes para nuestro sistema operativo.

Figura 5. Apache Group

En la Figura 5 observará el lector el enlace a la pagina de Apache Server. Se pulsa y obtenemos la página web que vemos en la Figura 6. En página nos encontraremos, además de los enlaces necesarios para descargar el software, información de ultima hora del Grupo, la documentación del servidor, notas sobre la licencia de uso, etc.

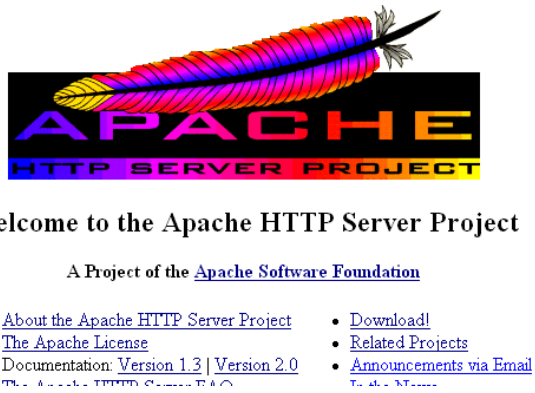


Figura 6. Apache Server

Se debe pulsar Download para descargar la versión que el lector quiera utilizar.

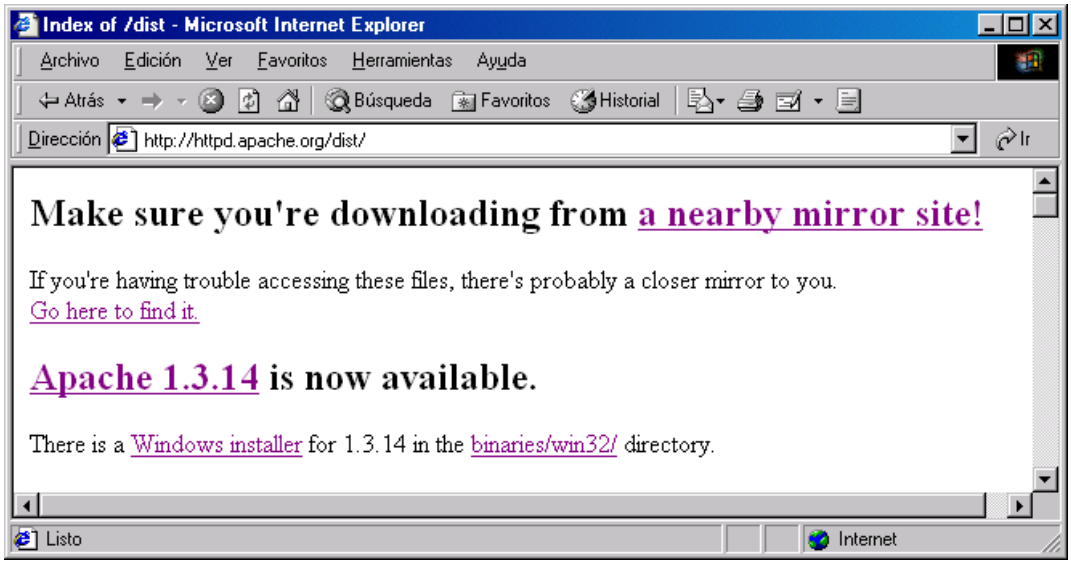


Figura 7. Pagina de descarga

Pulsando Windows instaler obtenemos la ventana de la Figura 8.

Figura 8. Descarga de apache_1_3_14_win32_r2.exe

Como verá el lector, se trata de la versión 1.3.14 de Apache. Ya se ha comentado que este servidor esta siendo continuamente revisado.

Una vez descargado el fichero, se ejecutará, apareciendo un programa de instalación típico de Windows. Las siguientes figuras muestran la secuencia de instalación del producto.

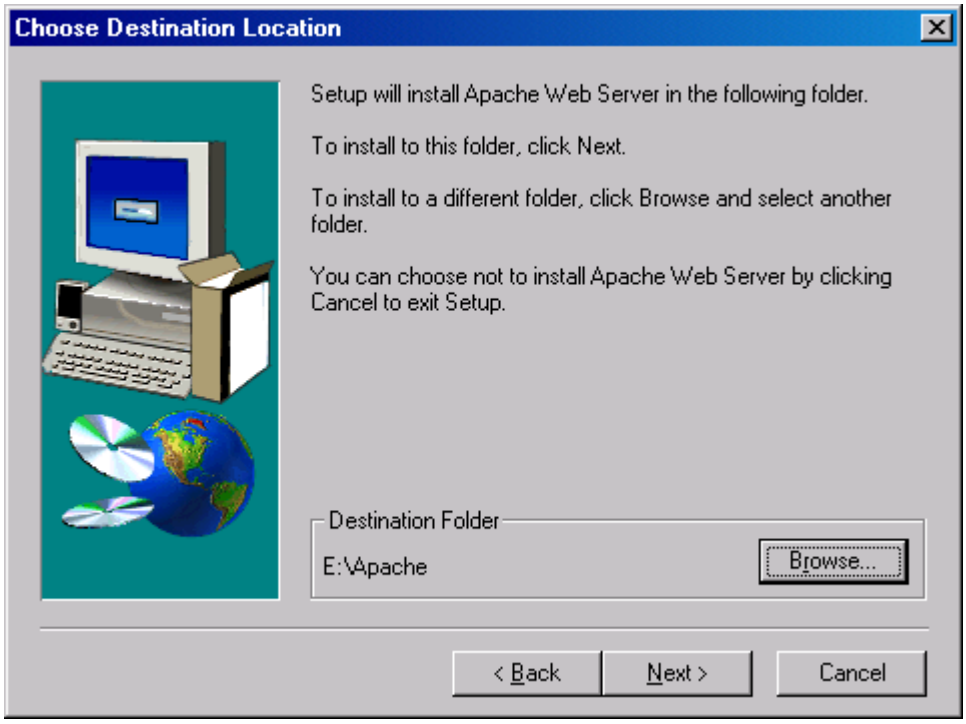


Figura 9. Cambio de directorio de instalación

Recomiendo el cambio de directorio de instalación a \apache pues por defecto aparece \Archivos de programa\Apache Group\Apache que puede llegar a ser un poco menos manejable para utilizarlo que el anterior. Este directorio será a partir de ahora %APACHE_HOME% como directorio de referencia del servidor.

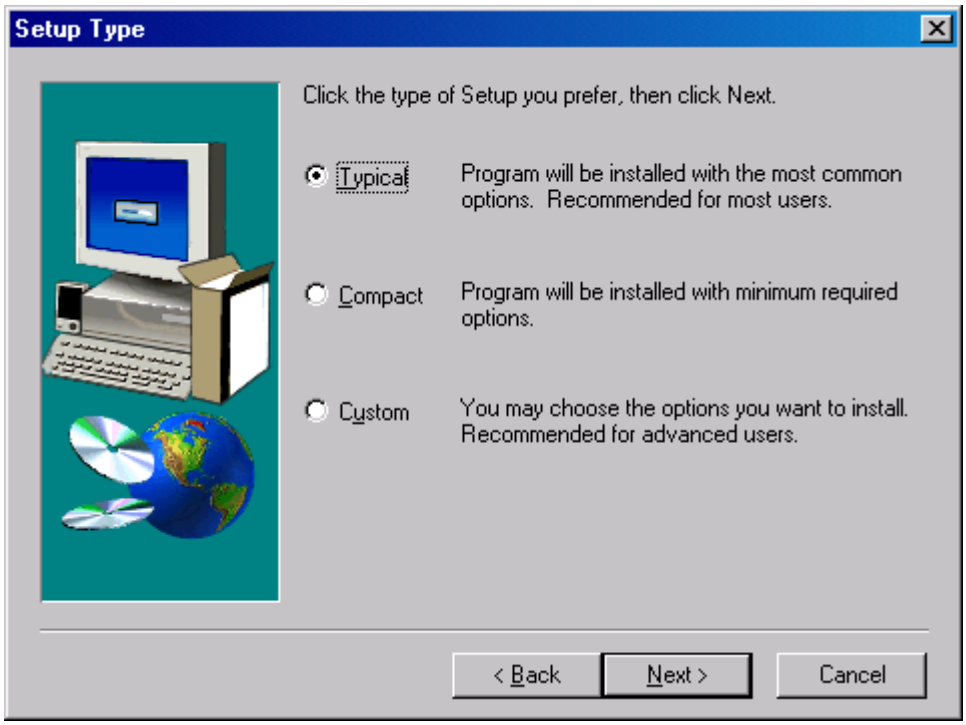


Figura 10. Tipo de instalación

Elegimos la instalación típica (Figura 10)

Cuando se termine la instalación se debe probar si el servidor se ha instalado correctamente. Para ello pulsamos Menú Inicio->Programas->Apache Web Server->Management-> Start Apache. Aparecerá entonces la ventana de la Figura 11.

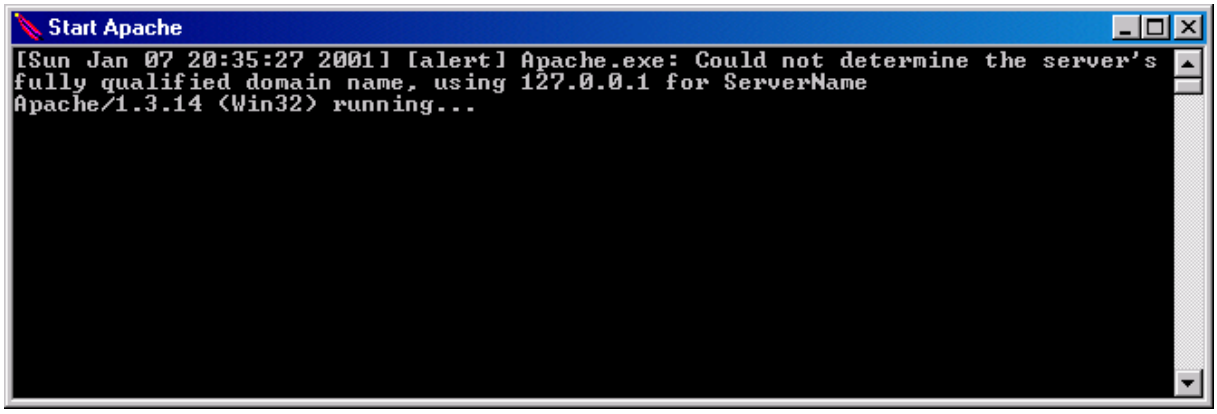


Figura 11. Apache en servicio

Como verá el lector, la ejecución aporta un mensaje de error que indica que hay una clausula llamada ServerName que toma un valor por defecto. El lector corregirá mas adelente este aviso. Para comprobar definitivamanete que todo va bien, escribamos en la barra de direcciones del navegador http://localhost. Si no aparece la pagina de la Figura 12, es que hay algun problema en la instalación o en la configuración, pero el lector debe saber que con estos minimos pasos ya se puede tener en funcionamiento el servidor Apache.

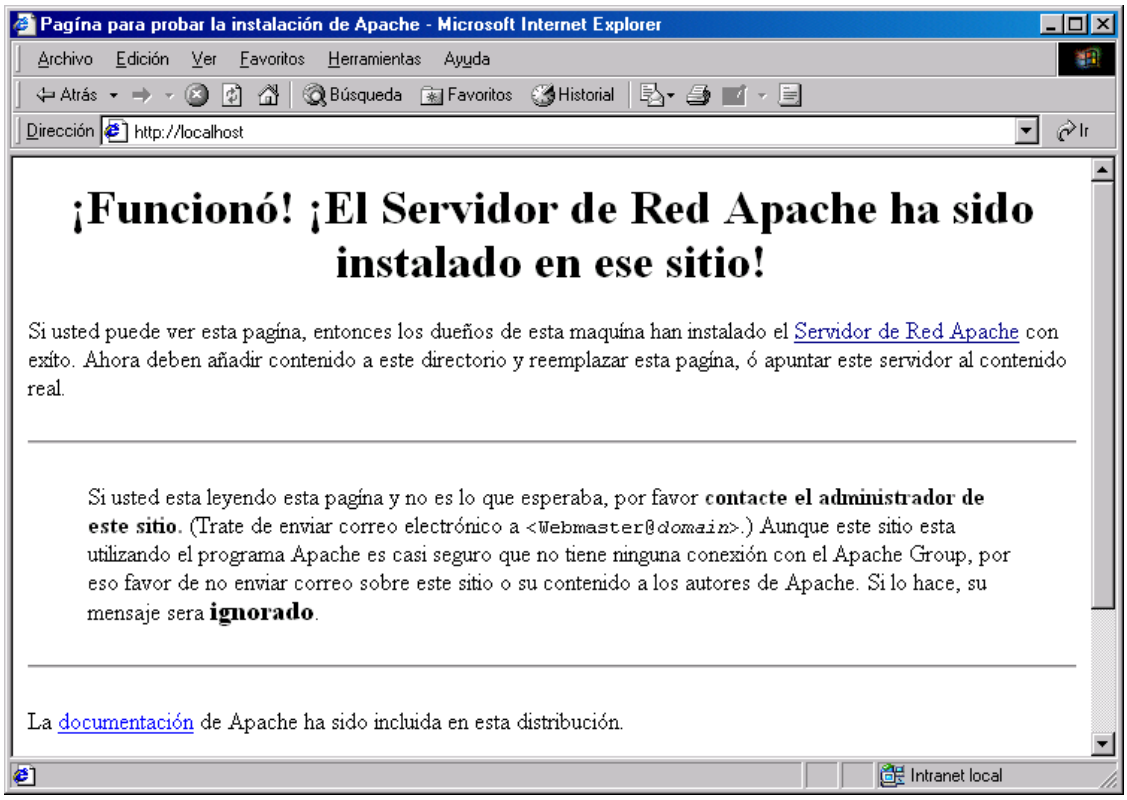


Figura 12. Pagina html por defecto del Servidor Apache

Compilación del código fuente de Apache

No siempre es posible descargar los archivos de Apache en formato binario para todos los sistemas operativos. En el caso anterior lo hemos podido hacer porque el Grupo Apache aporta la instalación de su servidor para el sistema operativo Windows, pero si lo queremos para por ejemplo otro sistema operativo, por ejemplo Linux, deberemos descargar primero el código fuente y después compilarlo. Debemos descargar el archivo apache_1.3.14.tar.gz, que trae los archivos fuentes de todos los sistemas operativos

Por ejemplo, para un sistema Linux eligiremos los mismos fuentes que para un sistema Unix. Para descomprimir este fichero en windows lo podemos hacer, por ejemplo con Winzip. En un sistema Linux lo hariamos con los ejecutables tar, gnunzip o gzip, escribiendo :

>tar xvzf apache_1.3.14.tar.gz

o incluso :

>gzip –d apache_1.3.14.tar.gz

>tar xvf apache_1.3.14.tar

Este fichero se descomprime en un directorio que se llama apache_1.3.14, donde se va a guardar toda la estructura de directorios de Apache. Esta estructura coincide con la que habíamos hecho anteriormente para Windows, por lo que recomiendo para aquellos lectores que quieran compilar el código fuente con Microsoft Visual C++, copien todo el contenido de este directorio a %APACHE_HOME%. La carpeta \apache_1.3.14\src\os contiene otras carpetas con código fuente para varios sistemas operativos.

Si seguimos con Linux, el lector debera configurar un fichero, para posteriormente ejecutar Configure para obtener un fichero Makefile para compilar Apache . Dicho archivo de configuracion se llama Configuration (también exixte un fichero de plantilla de este llamado Configuration.tmpl) y su contenido se encuentra en el Código fuente 1.

```
# Config file for the Apache httpd.
# Configuration.tmpl is the template for Configuration. Configuration should
# be edited to select the modules to be included as well as various flags
# for Makefile.
# The template should only be changed when a new system or module is added,
# or an existing one modified. This will also most likely require some minor
# changes to Configure to recognize those changes.
# There are 5 types of lines here:
# '#' comments, distinguished by having a '#' as the first non-blank character
#
# Makefile options, such as CC=gcc, etc...
#
# Rules, distinguished by having "Rule" at the front. These are used to
# control Configure's behavior as far as how to create Makefile.
#
# Module selection lines, distinguished by having 'AddModule' at the front.
# These list the configured modules, in priority order (highest priority
# last).  They're down at the bottom.
#
# Optional module selection lines, distinguished by having '%Module'
# at the front.  These specify a module that is to be compiled in (but
# not enabled).  The AddModule directive can be used to enable such a
# module.  By default no such modules are defined.
```

```
# Makefile configuration
#
# These are added to the general flags determined by Configure.
# Edit these to work around Configure if needed. The EXTRA_* family
# will be added to the regular Makefile flags. For example, if you
# want to compile with -Wall, then add that to EXTRA_CFLAGS. These
# will be added to whatever flags Configure determines as appropriate
# and needed for your platform.
#
# You can also set the compiler (CC) and optimization (OPTIM) used here as
# well.  Settings here have priority; If not set, Configure will attempt to
# guess the C compiler, looking for gcc first, then cc.
#
# Optimization note:
# Be careful when adding optimization flags (like -O3 or -O6) on the OPTIM
# entry, especially when using some GCC variants. Experience showed that using
# these for compiling Apache is risky. If you don't want to see Apache dumping
# core regularly then at most use -O or -O2.
#
# The EXTRA_DEPS can be used to add extra Makefile dependencies to external
# files (for instance third-party libraries) for the httpd target. The effect
# is that httpd is relinked when those files are changed.
#
EXTRA_CFLAGS=
EXTRA_LDFLAGS=
EXTRA_LIBS=
EXTRA_INCLUDES=
EXTRA_DEPS=

CC=gcc
#CPP=
#OPTIM=
#RANLIB=

# Name of the installed Apache HTTP webserver.
#
#TARGET=

# Dynamic Shared Object (DSO) support
#
# There is experimental support for compiling the Apache core and
# the Apache modules into dynamic shared object (DSO) files for
# maximum runtime flexibility.
#
# The Configure script currently has only limited built-in
# knowledge on how to compile these DSO files because this is
# heavily platform-dependent. The current state of supported and
# explicitly unsupported platforms can be found in the file
# "htdocs/manual/dso.html", under "Supported Platforms".
#
# For other platforms where you want to use the DSO mechanism you
# first have to make sure it supports the pragmatic dlopen()
# system call and then you have to provide the appropriate
# compiler and linker flags below to create the DSO files on your
# particular platform.
#
# The placement of the Apache core into a DSO file is triggered
# by the SHARED_CORE rule below while support for building
# individual Apache Modules as DSO files and loading them under
# runtime without recompilation is triggered by 'SharedModule'
# commands. To be able to use the latter one first enable the
# module mod_so (see corresponding 'AddModule' command below).
# Then enable the DSO feature for particular modules individually
# by replacing their 'AddModule' command with 'SharedModule' and
# change the filename extension from '.o' to '.so'.
#
# Sometimes the DSO files need to be linked against other shared
```

```
# libraries to explicitly resolve symbols from them when the
# httpd program not already contains references to them. For
# instance when buidling mod_auth_db as a DSO you need to link
# the DSO against the libdb explicity because the Apache kernel
# has no references for this library. But the problem is that
# this "chaining" is not supported on all platforms. Although one
# usually can link a DSO against another DSO without linker
# complains the linkage is not really done on these platforms.
# So, when you receive "unresolved symbol" errors under runtime
# when using the LoadModule directive for a particular module try
# to enable the SHARED_CHAIN rule below.
#CFLAGS_SHLIB=
#LD_SHLIB=
#LDFLAGS_SHLIB=
#LDFLAGS_SHLIB_EXPORT=

Rule SHARED_CORE=default
Rule SHARED_CHAIN=default

# Rules configuration
#
# These are used to let Configure know that we want certain
# functions. The format is: Rule RULE=value
#
# At present, only the following RULES are known: WANTHSREGEX, SOCKS4,
# SOCKS5, IRIXNIS, IRIXN32, PARANOID, and DEV_RANDOM.
#
# For all Rules except DEV_RANDOM, if set to "yes", then Configure knows
# we want that capability and does what is required to add it in. If set
# to "default" then Configure makes a "best guess"; if set to anything
# else, or not present, then nothing is done.
#
# SOCKS4:
#  If SOCKS4 is set to 'yes', be sure that you add the socks library
#  location to EXTRA_LIBS, otherwise Configure will assume
#  "-L/usr/local/lib -lsocks"
#
# SOCKS5:
#  If SOCKS5 is set to 'yes', be sure that you add the socks5 library
#  location to EXTRA_LIBS, otherwise Configure will assume
#  "-L/usr/local/lib -lsocks5"
#
# IRIXNIS:
#  Only takes effect if Configure determines that you are running
#  SGI IRIX.  If you are using a (ancient) 4.x version of IRIX, you
#  need this if you are using NIS and Apache needs access to it for
#  things like mod_userdir.  This is not required on 5.x and later
#  and you should not enable it on such systems.
#
# IRIXN32:
#  If you are running a version of IRIX and Configure detects
#  n32 libraries, it will use those instead of the o32 ones.
#
# PARANOID:
#  New with version 1.3, during Configure modules can run
#  pre-programmed shell commands in the same environment that
#  Configure runs in. This allows modules to control how Configure
#  works. Normally, Configure will simply note that a module
#  is performing this function. If PARANOID is set to yes, it will
#  actually print-out the code that the modules execute
#
# EXPAT:
#  Include James Clark's Expat package into Apache, for use by the
#  modules. The "default" is to include it if the lib/expat-lite/
#  directory is present. This rule will always be interpreted as "no"
#  if the directory is not present.
#
```

```
Rule SOCKS4=no
Rule SOCKS5=no
Rule IRIXNIS=no
Rule IRIXN32=yes
Rule PARANOID=no
Rule EXPAT=default

# DEV_RANDOM:
#  Note: this rule is only used when compiling mod_auth_digest.
#  mod_auth_digest requires a cryptographically strong random seed for its
#  random number generator. It knows two ways of getting this: 1) from
#  a file or device (such as "/dev/random"), or 2) from the truerand
#  library. If this rule is set to 'default' then Configure will choose
#  to use /dev/random if it exists, else /dev/urandom if it exists,
#  else the truerand library. To override this behaviour set DEV_RANDOM
#  either to 'truerand' (to use the library) or to a device or file
#  (e.g. '/dev/urandom'). If the truerand library is selected, Configure
#  will assume "-L/usr/local/lib -lrand".
Rule DEV_RANDOM=default

# The following rules should be set automatically by Configure. However, if
# they are not set by Configure (because we don't know the correct value for
# your platform), or are set incorrectly, you may override them here.
# If you have to do this, please let us know what you set and what your
# platform is, by filling out a problem report form at the Apache web site:
# <http://bugs.apache.org/>.  If your browser is forms-incapable, you
# can get the information to us by sending mail to apache-bugs@apache.org.
#
# WANTHSREGEX:
#  Apache requires a POSIX regex implementation. Henry Spencer's
#  excellent regex package is included with Apache and can be used
#  if desired. If your OS has a decent regex, you can elect to
#  not use this one by setting WANTHSREGEX to 'no' or commenting
#  out the Rule. The "default" action is "yes" unless overruled
#  by OS specifics
Rule WANTHSREGEX=default

# Module configuration
#
# Modules are listed in reverse priority order --- the ones that come
# later can override the behavior of those that come earlier.  This
# can have visible effects; for instance, if UserDir followed Alias,
# you couldn't alias out a particular user's home directory.
# The configuration below is what we consider a decent default
# configuration.  If you want the functionality provided by a particular
# module, remove the "#" sign at the beginning of the line. But remember,
# the more modules you compile into the server, the larger the executable
# is and the more memory it will take, so if you are unlikely to use the
# functionality of a particular module you might wish to leave it out.
## mod_mmap_static is an experimental module, you almost certainly
## don't need it.  It can make some webservers faster.  No further
## documentation is provided here because you'd be foolish
## to use mod_mmap_static without reading the full documentation.
# AddModule modules/experimental/mod_mmap_static.o
## mod_vhost_alias provides support for mass virtual hosting
## by dynamically changing the document root and CGI directory
## based on the host header or local IP address of the request.
## See "../htdocs/manual/vhosts/mass.html".
# AddModule modules/standard/mod_vhost_alias.o

##
## Config manipulation modules
##
## mod_env sets up additional or restricted environment variables to be
## passed to CGI/SSI scripts.  It is listed first (lowest priority) since
## it does not do per-request stuff.
```

```
AddModule modules/standard/mod_env.o

##
## Request logging modules
##

AddModule modules/standard/mod_log_config.o
## Optional modules for NCSA user-agent/referer logging compatibility
## We recommend, however, that you just use the configurable access_log.
# AddModule modules/standard/mod_log_agent.o
# AddModule modules/standard/mod_log_referer.o

##
## Type checking modules
##
## mod_mime_magic determines the type of a file by examining a few bytes
## of it and testing against a database of filetype signatures.  It is
## based on the unix file(1) command.
## mod_mime maps filename extensions to content types, encodings, and
## "magic" type handlers (the latter is obsoleted by mod_actions, and
## don't confuse it with the previous module).
## mod_negotiation allows content selection based on the Accept* headers.
# AddModule modules/standard/mod_mime_magic.o
AddModule modules/standard/mod_mime.o
AddModule modules/standard/mod_negotiation.o

##
## Content delivery modules
##
## The status module allows the server to display current details about
## how well it is performing and what it is doing.  Consider also enabling
## the 'ExtendedStatus On' directive to allow full status information.
## Please note that doing so can result in a palpable performance hit.
AddModule modules/standard/mod_status.o
## The Info module displays configuration information for the server and
## all included modules. It's very useful for debugging.
# AddModule modules/standard/mod_info.o
## mod_include translates server-side include (SSI) statements in text files.
## mod_autoindex handles requests for directories which have no index file
## mod_dir handles requests on directories and directory index files.
## mod_cgi handles CGI scripts.
AddModule modules/standard/mod_include.o
AddModule modules/standard/mod_autoindex.o
AddModule modules/standard/mod_dir.o
AddModule modules/standard/mod_cgi.o

## The asis module implements ".asis" file types, which allow the embedding
## of HTTP headers at the beginning of the document.  mod_imap handles internal
## imagemaps (no more cgi-bin/imagemap/!).  mod_actions is used to specify
## CGI scripts which act as "handlers" for particular files, for example to
## automatically convert every GIF to another file type.
AddModule modules/standard/mod_asis.o
AddModule modules/standard/mod_imap.o
AddModule modules/standard/mod_actions.o

##
## URL translation modules.
##

## The Speling module attempts to correct misspellings of URLs that
## users might have entered, namely by checking capitalizations
## or by allowing up to one misspelling (character insertion / omission /
## transposition/typo). This catches the majority of misspelled requests.
## If it finds a match, a "spelling corrected" redirection is returned.
# AddModule modules/standard/mod_speling.o
## The UserDir module for selecting resource directories by user name
## and a common prefix, e.g., /~<user> , /usr/web/<user> , etc.
```

```
AddModule modules/standard/mod_userdir.o
## The Alias module provides simple URL translation and redirection.
AddModule modules/standard/mod_alias.o
## The URL rewriting module allows for powerful URI-to-URI and
## URI-to-filename mapping using a regular expression based
## rule-controlled rewriting engine.
# AddModule modules/standard/mod_rewrite.o

##
## Access control and authentication modules.
##
AddModule modules/standard/mod_access.o
AddModule modules/standard/mod_auth.o

## The anon_auth module allows for anonymous-FTP-style username/
## password authentication.
# AddModule modules/standard/mod_auth_anon.o
## db_auth and dbm_auth work with Berkeley DB files - make sure there
## is support for DBM files on your system.  You may need to grab the GNU
## "gdbm" package if not and possibly adjust EXTRA_LIBS. (This may be
## done by Configure at a later date)
# AddModule modules/standard/mod_auth_dbm.o
# AddModule modules/standard/mod_auth_db.o

## "digest" implements HTTP Digest Authentication rather than the less
## secure Basic Auth used by the other modules.  This is the old version.
# AddModule modules/standard/mod_digest.o
## "auth_digest" implements HTTP/1.1 Digest Authentication (RFC 2617)
## rather than the less secure Basic Auth used by the other modules.
## This is an updated version of mod_digest, but it is not as well tested
## and is therefore marked experimental.  Use either the one above, or
## this one below, but not both digest modules.
## Note: if you add this module in then you might also need the
## truerand library (available for example from
## ftp://research.att.com/dist/mab/librand.shar) - see the Rule
## DEV_RANDOM above for more info.
##
## Must be added above (run later than) the proxy module because the
## WWW-Authenticate and Proxy-Authenticate headers are parsed in the
## post-read-request phase and it needs to know if this is a proxy request.
# AddModule modules/experimental/mod_auth_digest.o
## Optional Proxy
##
## The proxy module enables the server to act as a proxy for outside
## http and ftp services. It's not as complete as it could be yet.
## NOTE: You do not want this module UNLESS you are running a proxy;
##       it is not needed for normal (origin server) operation.
# AddModule modules/proxy/libproxy.a
## Optional response header manipulation modules.
##
## cern_meta mimics the behavior of the CERN web server with regards to
## metainformation files.
# AddModule modules/standard/mod_cern_meta.o
## The expires module can apply Expires: headers to resources,
## as a function of access time or modification time.
# AddModule modules/standard/mod_expires.o
## The headers module can set arbitrary HTTP response headers,
## as configured in server, vhost, access.conf or .htaccess configs
# AddModule modules/standard/mod_headers.o
## Miscellaneous modules
##
## mod_usertrack is the new name for mod_cookies.  This module
## uses Netscape cookies to automatically construct and log
## click-trails from Netscape cookies, or compatible clients who
## aren't coming in via proxy.
##
## You do not need this, or any other module to allow your site
```

```
## to use Cookies.  This module is for user tracking only
# AddModule modules/standard/mod_usertrack.o
## The example module, which demonstrates the use of the API.  See
## the file modules/example/README for details.  This module should
## only be used for testing—DO NOT ENABLE IT on a production server.
# AddModule modules/example/mod_example.o
## mod_unique_id generates unique identifiers for each hit, which are
## available in the environment variable UNIQUE_ID.  It may not work on all
## systems, hence it is not included by default.
# AddModule modules/standard/mod_unique_id.o
## mod_so lets you add modules to Apache without recompiling.
## This is an experimental feature at this stage and only supported
## on a subset of the platforms we generally support.
## Don't change this entry to a 'SharedModule' variant (Bootstrapping!)
# AddModule modules/standard/mod_so.o
## mod_setenvif lets you set environment variables based on the HTTP header
## fields in the request; this is useful for conditional HTML, for example.
## Since it is also used to detect buggy browsers for workarounds, it
## should be the last (highest priority) module.
AddModule modules/standard/mod_setenvif.o
```

Código fuente 1. Fichero Configuration

Debemos indicar que la línea CC indica cual va a ser el compilador a utilizar. Existen también algunos parámetros externos (EXTRA_FLAGS) para indicar algún código externo al compilador. Para darle funcionalidad a Configure disponemos de las reglas. Algunas de las mas importantes son:

Rule STATUS=yes (status_module)

Rule SOCKS4=no (TCP/IP)

Rule WANTHSREGEX=default (paquete de expresión normal)

Rule IRIXNIS=no (procesadores IRIS o NIS)

El lector ya conoce que Apache se puede configurar con ciertos módulos que van a añadir mas funcionalidad al servidor. Para poder añadir un módulo disponemos de la instrucción AddModule . Para añadir un módulo lo haremos como sigue:

AddModule modules/standard/mod_cgi.o

Con esta línea se añadirá el modulo mod_cgi que se encuentra en el directorio %ApachRoot%/ src/modules/estándar. Ya se encuentran activados los módulos para una instalación típica. Si el lector quisiera desactivar alguno, lo haría mediante el símbolo #.

El siguiente paso que se debe realizar consiste en la ejecución del fichero Configure, escribiendo la siguiente línea:

>./Configure

Esta ejecución crea el fichero Makefile. Una vez obtenido este fichero, escribiremos en la línea de comandos:

>make

Este comando usará el archivo Makefile que se creó con Configure obteniendo un ejecutable llamado httpd, que será el que utilizaremos. Escribiremos entonces:

>./httpd –v

Esto le mostrará la versión de Apache, la cual deberá coincidir con la que el lector haya descargado. Copie el ejecutable al directorio %APACHE_HOME%/bin, y los archivos de configuración a %APACHE_HOME%/conf, y no permita que nadie que no tenga un perfil de administrador acceda a estos directorios.

Compilación para Windows

La manera mas fácil de compilar Apache para Windows es hacerlo mediante Microsoft Visual C++ de Visual Studio. El lector no debe olvidar que la versión en formato binario se encuentra disponible en la web.

Una vez que hayamos realizado la descompresión del archivo apache_1.3.14.tar.gz, por ejemplo con Winzip, obtendremos el directorio apache_1.3.14, Si anteriormente hemos instalado la versión binaria, copiamos el contenido de este directorio en el que habiamos tomado como %APACHE_HOME%, quedando la estructura tla y como aparece en la Figura 13

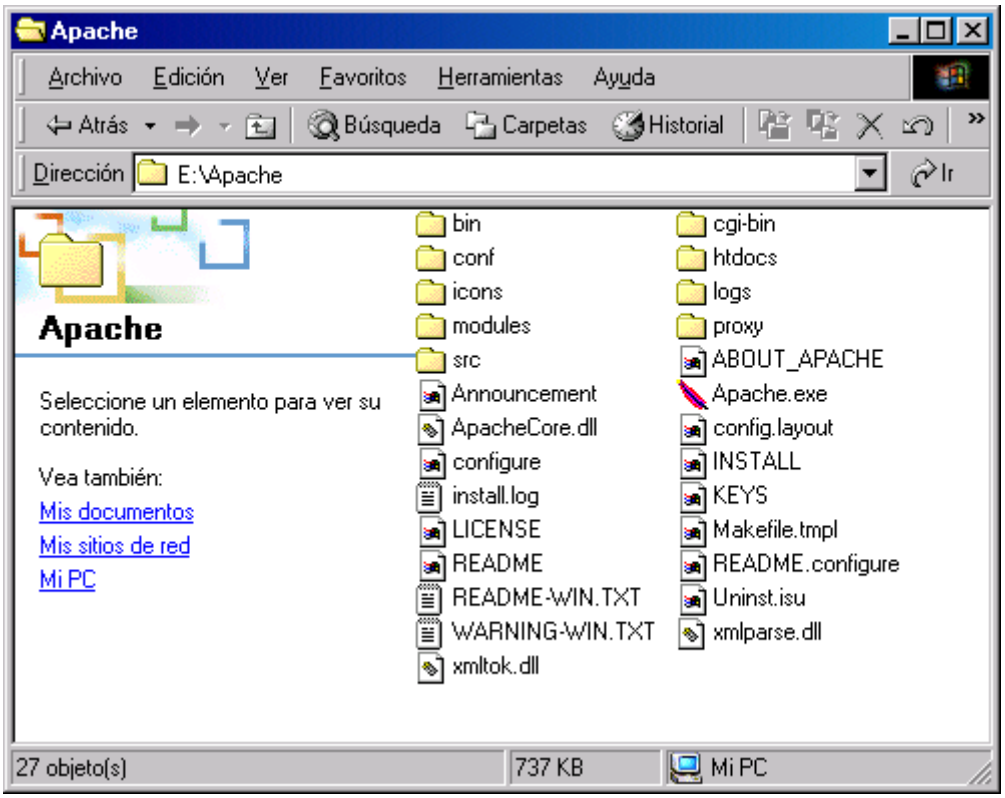


Figura 13. %APACHE_HOME%

Para compilar Con Visual C++ , deberemos tener ejecutado el archivo `C:\Archivos de programa\Microsoft Visual Studio\VC98\Bin\VCVARS32.BAT` que configurará las variables de entorno para esta herramienta. Existe un archivo, en la carpeta %APACHE_HOME%\src, llamado makefile.win (Código fuente 2), que será el archivo que utilizaremos para la compilación del código fuente. Seguidamente escribimos en la línea de comandos:

```
%APACHE_HOME%\src>nmake /f Makefile.win INSTDIR=%APACHE_HOME%
installr
```

```
# Makefile for Windows NT and Windows 95/98/2000
# Targets are:
#   _apacher   - build Apache in Release mode
#   _apached   - build Apache in Debug mode
#   installr   - build and install a Release build
#   installd   - build and install a Debug build
#   clean      - remove (most) generated files
#   _cleanr    - remove (most) files generated by a Release build
#   _cleand    - remove (most) files generated by a Debug build
#
# The default installation directory is \Apache. This can be changed
# with the INSTDIR macro, for example:
#
#   nmake /f Makefile.nt INSTDIR="d:\Program Files\Apache" installr
#
# Note: this does NOT change the compiled in default "server root"
!IF "$(INSTDIR)" == ""
INSTDIR=\Apache
!MESSAGE Using default install directory \Apache
!ENDIF

!IF "$(MAKE)" == "NMAKE"
# Microsoft NMake options
MAKEOPT=-nologo
!ELSEIF "$(MAKE)" == "make"
# Borland make options
# Borland build of Apache is highly experimental and unsupported.
MAKEOPT=-s -N
!ELSE
!MESSAGE Warning: unrecognized build command "$(MAKE)"
!ENDIF

default:        _apacher
_apacher:
$(MAKE) $(MAKEOPT) -f Makefile.win SHORT=R LONG=Release _build
_apached:
$(MAKE) $(MAKEOPT) -f Makefile.win SHORT=D LONG=Debug   _build
installr:
$(MAKE) $(MAKEOPT) -f Makefile.win SHORT=R LONG=Release _build _install
installd:
$(MAKE) $(MAKEOPT) -f Makefile.win SHORT=D LONG=Debug   _build _install
_cleanr:
$(MAKE) $(MAKEOPT) -f Makefile.win SHORT=R LONG=Release CTARGET=CLEAN _build
_cleand:
      $(MAKE) $(MAKEOPT) -f Makefile.win SHORT=D LONG=Debug   CTARGET=CLEAN _build

clean: _cleanr _cleand
      $(MAKE) $(MAKEOPT) -f Makefile.win SHORT=R LONG=Release CTARGET=CLEAN
_installdll

installdll:
$(MAKE) $(MAKEOPT) -f Makefile.win SHORT=R LONG=Release _installdll

_build:
#     echo LONG $(LONG) SHORT $(SHORT) x
      cd os\win32
       $(MAKE) $(MAKEOPT) -f ApacheOS.mak CFG="ApacheOS - Win32 $(LONG)" RECURSE=0
$(CTARGET)
      cd ..\..
      cd regex
       $(MAKE) $(MAKEOPT) -f regex.mak CFG="regex - Win32 $(LONG)" RECURSE=0
$(CTARGET)
      cd ..
```

```
	cd ap
	 $(MAKE) $(MAKEOPT) -f ap.mak CFG="ap - Win32 $(LONG)" RECURSE=0 $(CTARGET)
	cd ..
	cd support
	 $(MAKE) $(MAKEOPT) -f htpasswd.mak CFG="htpasswd - Win32 $(LONG)" RECURSE=0
$(CTARGET)
	 $(MAKE) $(MAKEOPT) -f htdigest.mak CFG="htdigest - Win32 $(LONG)" RECURSE=0
$(CTARGET)
	 $(MAKE) $(MAKEOPT) -f logresolve.mak CFG="logresolve - Win32 $(LONG)"
RECURSE=0 $(CTARGET)
	 $(MAKE) $(MAKEOPT) -f rotatelogs.mak CFG="rotatelogs - Win32 $(LONG)"
RECURSE=0 $(CTARGET)
	cd ..
	cd lib/expat-lite
	 $(MAKE) $(MAKEOPT) -f xmltok.mak CFG="xmltok - Win32 $(LONG)" RECURSE=0
$(CTARGET)
	 $(MAKE) $(MAKEOPT) -f xmlparse.mak CFG="xmlparse - Win32 $(LONG)" RECURSE=0
$(CTARGET)
	cd ../..
	cd lib/sdbm
	 $(MAKE) $(MAKEOPT) -f sdbm.mak CFG="sdbm - Win32 $(LONG)" RECURSE=0
$(CTARGET)
	cd ../..
	cd main
	 $(MAKE) $(MAKEOPT) -f gen_uri_delims.mak CFG="gen_uri_delims - Win32
$(LONG)" RECURSE=0 $(CTARGET)
	 $(MAKE) $(MAKEOPT) -f gen_test_char.mak CFG="gen_test_char - Win32 $(LONG)"
RECURSE=0 $(CTARGET)
	cd ..
•	del Core$(SHORT)\buildmark.obj
$(MAKE) $(MAKEOPT) -f ApacheCore.mak CFG="ApacheCore - Win32 $(LONG)" RECURSE=0
$(CTARGET)
$(MAKE) $(MAKEOPT) -f Apache.mak CFG="Apache - Win32 $(LONG)" RECURSE=0 $(CTARGET)
cd os\win32
set CFG=ApacheModuleStatus - Win32 $(LONG)
$(MAKE) $(MAKEOPT) -f ApacheModuleStatus.mak CFG="ApacheModuleStatus - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleInfo.mak CFG="ApacheModuleInfo - Win32 $(LONG)"
RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleAuthAnon.mak CFG="ApacheModuleAuthAnon - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleDigest.mak CFG="ApacheModuleDigest - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleCERNMeta.mak CFG="ApacheModuleCERNMeta - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleExpires.mak CFG="ApacheModuleExpires - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleHeaders.mak CFG="ApacheModuleHeaders - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleSpeling.mak CFG="ApacheModuleSpeling - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleUserTrack.mak CFG="ApacheModuleUserTrack - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleRewrite.mak CFG="ApacheModuleRewrite - Win32
$(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleAuthDigest.mak CFG="ApacheModuleAuthDigest -
Win32 $(LONG)" RECURSE=0 $(CTARGET)
$(MAKE) $(MAKEOPT) -f ApacheModuleAuthDBM.mak CFG="ApacheModuleAuthDBM - Win32
$(LONG)" RECURSE=0 $(CTARGET)
cd ..\..
cd modules\proxy
$(MAKE) $(MAKEOPT) -f ApacheModuleProxy.mak CFG="ApacheModuleProxy - Win32 $(LONG)"
RECURSE=0 $(CTARGET)
cd ..\..
_install:
•	mkdir $(INSTDIR)
•	mkdir $(INSTDIR)\modules
```

```
•      mkdir $(INSTDIR)\logs
•      mkdir $(INSTDIR)\conf
•      mkdir $(INSTDIR)\bin copy Apache$(SHORT)\Apache.exe $(INSTDIR) copy
Core$(SHORT)\ApacheCore.dll $(INSTDIR)
copy lib\expat-lite\$(LONG)\xmltok.dll $(INSTDIR)
copy lib\expat-lite\$(LONG)\xmlparse.dll $(INSTDIR)
copy os\win32\ApacheModuleStatus$(SHORT)\ApacheModuleStatus.dll $(INSTDIR)\modules
copy os\win32\ApacheModuleInfo$(SHORT)\ApacheModuleInfo.dll $(INSTDIR)\modules
copy os\win32\ApacheModuleAuthAnon$(SHORT)\ApacheModuleAuthAnon.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleDigest$(SHORT)\ApacheModuleDigest.dll $(INSTDIR)\modules
copy os\win32\ApacheModuleCERNMeta$(SHORT)\ApacheModuleCERNMeta.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleExpires$(SHORT)\ApacheModuleExpires.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleHeaders$(SHORT)\ApacheModuleHeaders.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleRewrite$(SHORT)\ApacheModuleRewrite.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleSpeling$(SHORT)\ApacheModuleSpeling.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleUserTrack$(SHORT)\ApacheModuleUserTrack.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleAuthDigest$(SHORT)\ApacheModuleAuthDigest.dll
$(INSTDIR)\modules
copy os\win32\ApacheModuleAuthDBM$(SHORT)\ApacheModuleAuthDBM.dll
$(INSTDIR)\modules
copy modules\proxy\$(LONG)\ApacheModuleProxy.dll $(INSTDIR)\modules
copy support\$(LONG)\htpasswd.exe $(INSTDIR)\bin
copy support\$(LONG)\htdigest.exe $(INSTDIR)\bin
copy support\$(LONG)\logresolve.exe $(INSTDIR)\bin
copy support\$(LONG)\rotatelogs.exe $(INSTDIR)\bin
copy support\dbmmanage $(INSTDIR)\bin\dbmmanage.pl
_installdll:
cd os\win32\installer\installdll
$(MAKE) $(MAKEOPT) -f install.mak CFG="install - Win32 $(LONG)" RECURSE=0
$(CTARGET)
cd ..\..\..
```

Código fuente 2. Makafile.win

El fichero de compilación está preparado para copiar el fichero ejecutable y los ficheros de configuración a su ubicación estándar. Tambien es posible editar todo el código fuente utilizando toda la funcionalidad gráfica que aporta Visual C++. Para ello deberemos abrir un fichero de proyecto llamado Apache.dsw, obteniendo el entorno de la Figura 14.

De todas formas , con compilar el proyecto no será suficiente. El lector deberá seguir unos pasos que le voy a indicar.

Lo primero es abrir los proyectos, uno a uno, con Visual C++ con el siguiente orden:

1. os\win32\ApacheOS.dsp
2. regex\regex.dsp
3. ap\ap.dsp
4. lib\expat-lite\xmltok.dsp
5. lib\expat-lite\xmlparse.dsp (requiere xmltok)

6. main\gen_uri_delims.dsp
7. main\gen_test_char.dsp
8. ApacheCore.dsp (requiere todos los anteriores)
9. Apache.dsp (requiere ApacheCore)

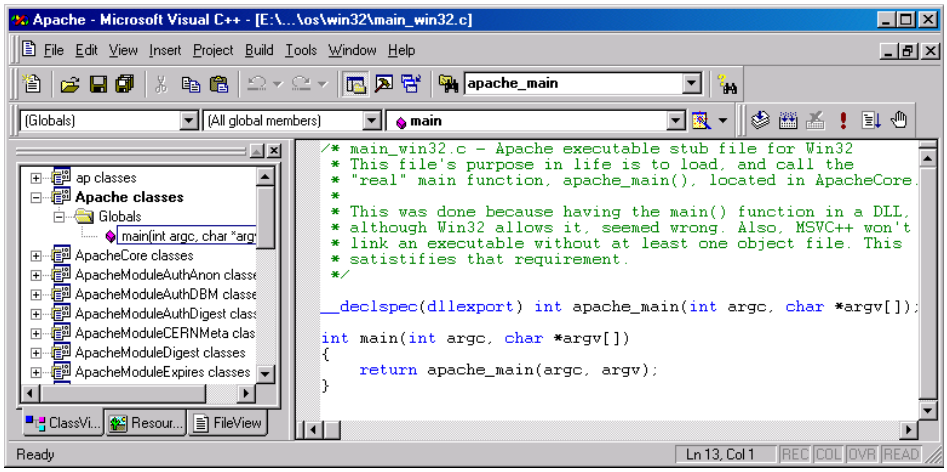


Figura 14. Proyecto de Apache con Visual C++

Cuando un proyecto requiere uno anterior debemos indicarlo en el menu Project->Settings->Link, en el apartado Object/library modules. La Figura 15 muestra como en el proyecto lib\expat-lite \xmlparse.dsp indicamos la librería que se requiere.

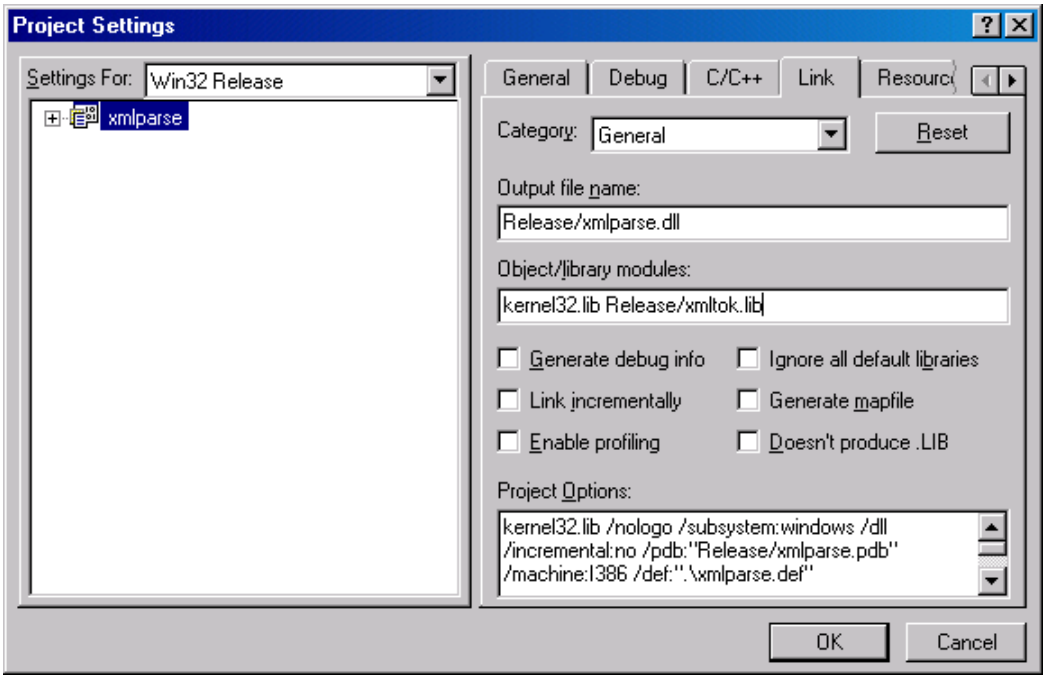


Figura 15 Opciones de Link

No debemos olvidar que debemos compilar en modo Release. Una vez compilado (Figura 16) debemos abrir un nuevo proyecto que nos ayudará a configurar nuestra instalación.

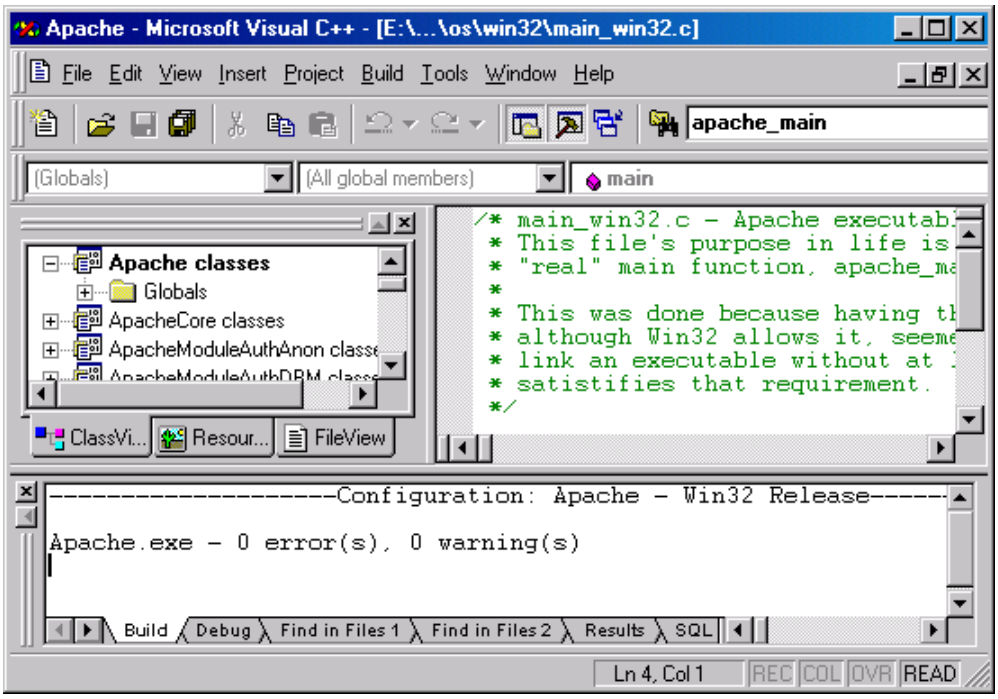


Figura 16. Fin de compilación de Apache

El proyecto que terminará por configurar la estrutcutra que debe poseer el servidor nos la creara el proyecto InstallBin.dsp. Se encargará de copiar los archivos generados (Apache.exe y DLL's) en los directorios correspondientes.

Si ponemos ejecutamos Apache obtendremos la ventana de la Figura 17.

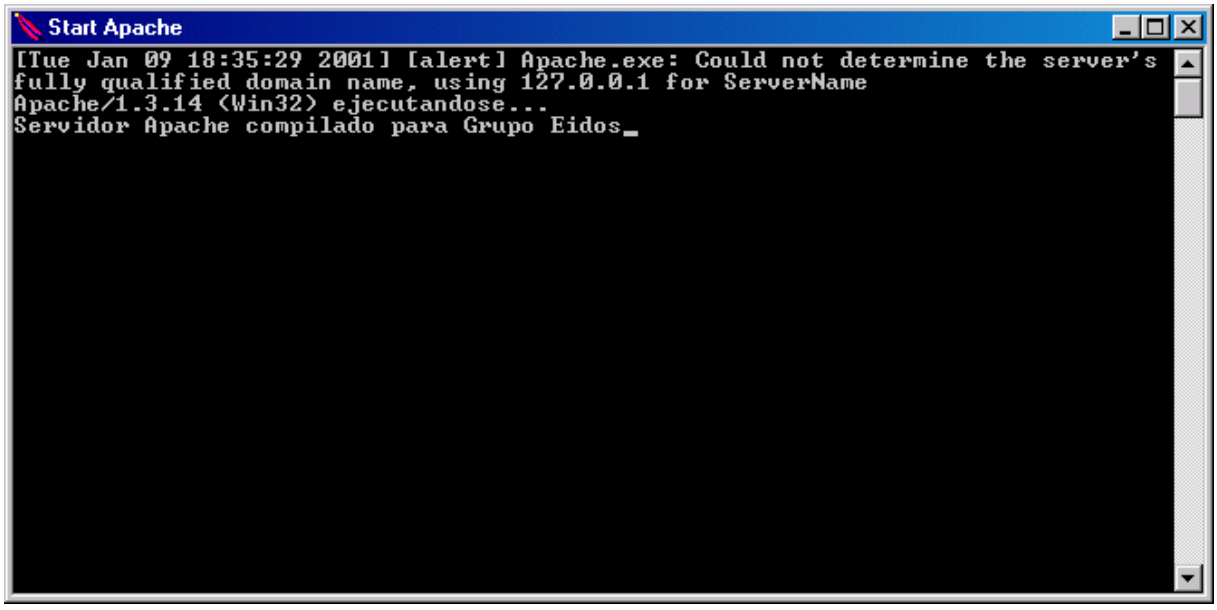


Figura 17. Apache compilado ejecutándose.

Solamente queda configurar como deseamos que funcione nuestro servidor.

Módulos de Apache

Durante la explicación del fichero de configuración, el lector habrá observado que hacíamos referencia a unas directrices en función de si estaban activos o no ciertos archivos. Esas directrices que aumentaban considerablemente la funcionalidad de Apache, pues se utilizaban como parámetros de dichos archivos. Estos son los módulos, que son partes de código fuente que se compilan con Apache para aumentar su funcionalidad. Muchos se incluyen en la compilación estandar.

mod_access

El módulo mod_access se utiliza en el control de acceso al servidor.

mod_actions

Se utiliza para ejecutar código CGI o una solicitud http

mod_alias

Se utiliza para convertir el sistema donde esta instalado el servidor en parte del servidor.

mod_asis

Este módulo se utiliza para enviar un documento del servidor sin cabeceras http. Resulta útil para redireccionar a un cliente sin necesidad de utilizar un scrip.

mod_auth

Se utiliza para confirmar la autenticación, mediante el nombre de los usuarios y sus contraseñas.

mod_auth_anon

Este módulo permite acceder de manera anónima a partes del servidor. Todos los usuarios pueden utilizar un identificador llamado “anonymus” para acceder a ciertas partes del sistema.

mod_auth_db

Cuando es necesario automatizar el proceso de autenticación, y no se pueden utilizar archivos DBM, pero si se pueden utilizar otros archivos, como por ejemplo los Berkeles DB, es preciso utilizar el módulo mod_auth_db

mod_auth_dbm

Como en el anterior caso, con los archivos de texto .htpasswd y .htaccess en procesos de alta velocidad, se puede ver afectado el rendimiento del servidor. Por lo tanto es necesario automatizar el proceso de autenticación, utilizando archivos DBM, es decir, archivos con contenido clave=valor, que mantienen una tabla indexada con todas las claves del archivo. Pueden utilizar también archivos GDBM, NDBM,SDMM y Berkeley DB.

mod_auth_external

Este módulo se utiliza cuando nos vemos en la necesidad de utilizar un sistema de autenticación externo.

mod_autoindex

Cuando un cliente hace una petición en un directorio, el servidor busca en el mismo un archivo con algún nombre de los indicados en la directiva DirectoryIndex. Esta es la función del módulo mod_autoindex.

mod_cern_meta

Este módulo se encarga de la metainformación. Tiene que ver con las directrices MetaFiles, MetaDir y MetaSuffix.

mod_cgi

Cuando compilamos Apache con este módulo, lo hacemos para poder usar programas CGI, así como su configuración.

mod_digest

Se utiliza para poder utilizar la autenticación diges.

mod_dir

Con este módulo, tnenemos la posibilidad de no incluir al final de una solicitud la barra inclinad /. Es decir, cuando escribimos www.eidos.es/algoritmodigital, el servidor lo convertirá a www.eidos.es/algoritmodigital/, donde buscará un nombre de archivo indicado en la directriz Director ;Index.

mod_env

Permite el envío de variables de entorno a los script CGI o SSI.

mod_expires

Este módulo se utiliza para indicar al cliente la cantidad de tiempo que dispone antes de que su solicitud pierda su validez, utilizando para ello las cabeceras http Expires.

mod_headers

Nos permite manipular las cabeceras http mediante la directriz Header.

mod_imap

Con este módulo, Apache permite la utilización de mapas de imágenes en los programas CGI.

mod_include

Se utiliza para los documentos SSI con la directriz inlude, que permite insertar el texto de un fichero dentro de otro.

mod_info

Con este módulo podemos visualizar a través de la red la información sobre la configuración y el estado del servidor.

mod_log_agent

Apache dispone de módulo para registrar la información del agente de usuario en un archivo independiente.

mod_log_config

Este módulo permite el registro de señales de peticiones recibidas por el servidor mod.

mod_log_referer

Permite señalar documentos por parte de aquellos que han sido accedidos desde el servidor.

mod_mime

Se utiliza para entregarle a los clientes metainformación sobre los documentos.

mod_negotiation

Apache se hace cargo mediante éste módulo de la negociación de contenido. Esta consiste en la selección de una versión de documento que corresponda a la mejor de las posibilidades del cliente. Existen dos tipos de mecanismos: mapas de tipos y búsquedas multiviews.

mod_setenvif

Este módulo se utiliza para crear variables de entorno personalizadas para ayudar a optimizar ciertas decisiones.

mod_spelling

Permite controlar las solicitudes URL mal escritas. Admite errores de una letra procurando localizar el documento pedido.

mod_unique_id

Módulo que se asegura que cada petición es única. Para ello utiliza la variable de entorno UNIQUE-ID.

3

Configuración y ejecución

Una vez que tenemos instalado el servidor Apache, el lector habrá observado que seguimos teniendo un aviso de que algo no funciona (figura nº 17). Esto es así porque la ejecución del fichero Apache.exe (httpd en Linux y Unix), toma unos valores por defecto para su funcionamiento. Piense el lector que, por ejemplo, cuando hemos ejecutado desde el navegador http://localhost , la página de saludo que aparece, no la hemos construido nosotros, y que desde luego , debe tener una ubicacón en el disco duro que aun no conocemos. Para esta configuración existen varios ficheros que se encuentran en el directorio %APACHE_HOME%\conf, siendo el mas importante el fichero httpd.conf. Alguno de estos ficheros se mantienen por compatibilidad con versiones anteriores, como por ejemplo srm.conf y access.conf Código fuente 3.

```
# This is the default file for the ResourceConfig directive in httpd.conf.
# It is processed after httpd.conf but before access.conf.
#
# To avoid confusion, it is recommended that you put all of your
# Apache server directives into the httpd.conf file and leave this
# one essentially empty.
```

Código fuente 3. srm.conf y access.conf

El contenido de estos ficheros se gestionará íntegramente en httpd.conf

El fichero de configuración httpd.conf

Este es el fichero principal de Apache, y se encuentra ubicado en el directorio %APACHE_HOME% \conf. Se utiliza para indicarle al servidor como debe ejecutarse, así como que otros programas debe ejecutar. La mejor manera de entender este fichero es dividirlo en varias partes par que el lector vaya acompañándome en su estudio.

```
# Based upon the NCSA server configuration files originally by Rob McCool.
#
# This is the main Apache server configuration file.  It contains the
# configuration directives that give the server its instructions.
# See <URL:http://www.apache.org/docs/> for detailed information about
# the directives.
#
# Do NOT simply read the instructions in here without understanding
# what they do.  They're here only as hints or reminders.  If you are unsure
# consult the online docs. You have been warned.
#
# After this file is processed, the server will look for and process
# E:/Apache/conf/srm.conf and then E:/Apache/conf/access.conf
# unless you have overridden these with ResourceConfig and/or
# AccessConfig directives here.
#
# The configuration directives are grouped into three basic sections:
#  1. Directives that control the operation of the Apache server process as a
#     whole (the 'global environment').
#  2. Directives that define the parameters of the 'main' or 'default' server,
#     which responds to requests that aren't handled by a virtual host.
#     These directives also provide default values for the settings
#     of all virtual hosts.
#  3. Settings for virtual hosts, which allow Web requests to be sent to
#     different IP addresses or hostnames and have them handled by the
#     same Apache server process.
#
# Configuration and logfile names: If the filenames you specify for many
# of the server's control files begin with "/" (or "drive:/" for Win32), the
# server will use that explicit path.  If the filenames do *not* begin
# with "/", the value of ServerRoot is prepended -- so "logs/foo.log"
# with ServerRoot set to "/usr/local/apache" will be interpreted by the
# server as "/usr/local/apache/logs/foo.log".
#
# NOTE: Where filenames are specified, you must use forward slashes
# instead of backslashes (e.g., "c:/apache" instead of "c:\apache").
# If a drive letter is omitted, the drive on which Apache.exe is located
# will be used by default.  It is recommended that you always supply
# an explicit drive letter in absolute paths, however, to avoid
# confusion.
#

### Section 1: Global Environment
#
# The directives in this section affect the overall operation of Apache,
# such as the number of concurrent requests it can handle or where it
# can find its configuration files.
#

#
# ServerType is either inetd, or standalone.  Inetd mode is only supported on
# Unix platforms.
ServerType standalone
```

Código fuente 4. Inicio de httpd.conf

Como verá el lector nos encontramos dentro de la configuración de parámetros de entorno globales.La primera instrucción que nos encontramos en el archivo de configuración es ServerType. Sirve para indicar como se ejecutará el servidor, pues habrá un proceso que este escuchando a través de un puerto. Esta directriz puede tener dos opciones

- Standalone
- Inet

La primera opción es la predeterminada, y la única disponible funcionando con Windows. El segundo valor indica que las peticiones pasan por inet (subproceso) antes de llegar al servidor. Con la primera opción, el servidor las atiende directamente.

```
# 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.
#
ServerRoot "E:/Apache"
```

Código fuente 5. httpd.conf (continuación)

La siguiente directriz que nos encontramos es ServerRoot. Esta indica el directorio principal de Apache, dentro del cual van a encontrarse todos aquellos ficheros de configuración, registro, ejecutables,etc, necesarios para su funcionamiento. En el encontraremos toda la información relativa al servidor.

```
# PidFile: The file in which the server should record its process
# identification number when it starts.
#
PidFile logs/httpd.pid
```

Código fuente 6. httpd.conf (continuación)

El fichero httpd.pid será el encargado de almacenar el identificador del proceso principal del servidor Apache cuando este se inicia. La directriz PidFile será la encargada de indicar cual es la ruta donde se encuentra este archivo.

```
# ScoreBoardFile: File used to store internal server process information.
# Not all architectures require this.  But if yours does (you'll know because
# this file will be  created when you run Apache) then you *must* ensure that
# no two invocations of Apache share the same scoreboard file.
#
ScoreBoardFile logs/apache_runtime_status
```

Código fuente 7. httpd.conf (continuación)

Existe un archivo donde se guardará la información relacionada con los procesos internos del servidor. Esta directriz indicará donde se encuentra dicho fichero.Por ejemplo, en Windows no es necesaria, aunque si lo es en otras plataformas.

```
# In the standard configuration, the server will process httpd.conf (this
# file, specified by the -f command line option), srm.conf, and access.conf
# in that order.  The latter two files are now distributed empty, as it is
# recommended that all directives be kept in a single file for simplicity.
# The commented-out values below are the built-in defaults.  You can have the
# server ignore these files altogether by using "/dev/null" (for Unix) or
# "nul" (for Win32) for the arguments to the directives.
#
#ResourceConfig conf/srm.conf
#AccessConfig conf/access.conf
```

Código fuente 8. httpd.conf (continuación)

En el proceso de instalación estándar los archivos srm.conf y access.conf se instalan vacíos (Código fuente 3). Si por alguna razón hubiera que activarlos, se harían mediante las directrices ResourceConfig y AccessConfig respectivamente, las cuales indican al ubicación de dichos archivos.

```
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 300
```

Código fuente 9. httpd.conf (continuación)

La directriz Timeout indicará el tiempo tras el cual se cierra la conexión, cuando una petición, que lo hace en forma de paquetes, no terminan de llegar.

```
# KeepAlive: Whether or not to allow persistent connections (more than
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On

#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests 100

#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 15
```

Código fuente 10. httpd.conf (continuación)

Las directivas KeepAlive se refieren a la forma en que una conexión TCP/IP se va a realizar. Cuando KeepAlive se encuentra a On, significa que el servidor admite conexiones persistentes, es decir, en lugar de recibir a traves de una conexión TCP/IP y enviar por otra, utiliza la misma para varias transacciones. MaxKeepAliveRequests indica el numero de peticiones permitidas por conexión y KeepAliveTimeout indica el tiempo que el servidor esperará una peticion antes de cerrar la conexión.

```
# Apache on Win32 always creates one child process to handle requests.  If it
# dies, another child process is created automatically.  Within the child
# process multiple threads handle incoming requests.  The next two
# directives control the behaviour of the threads and processes.
#

#
# MaxRequestsPerChild: the number of requests each child process is
# allowed to process before the child dies.  The child will exit so
# as to avoid problems after prolonged use when Apache (and maybe the
# libraries it uses) leak memory or other resources.  On most systems, this
# isn't really needed, but a few (such as Solaris) do have notable leaks
# in the libraries.  For Win32, set this value to zero (unlimited)
# unless advised otherwise.
#
# NOTE: This value does not include keepalive requests after the initial
#       request per connection. For example, if a child process handles
#       an initial request and 10 subsequent "keptalive" requests, it
#       would only count as 1 request towards this limit.
#
MaxRequestsPerChild 0
```

Código fuente 11. httpd.conf (continuación)

Una petición al servidor puede abrir un subproceso, el cual, a su vez, se puede hacer cargo de muchas peticiones. El número de peticiones que un subproceso puede atender es indicado por la directiva MaxRequestsPerChild. Cuando el valor es 0, significa que el subproceso no expirará nunca.

```
# Number of concurrent threads (i.e., requests) the server will allow.
# Set this value according to the responsiveness of the server (more
# requests active at once means they're all handled more slowly) and
# the amount of system resources you'll allow the server to consume.
#
ThreadsPerChild 50
```

Código fuente 12. httpd.conf (continuación)

Si tenemos en cuenta que el sistema operativo Windows es un sistema multitarea , entenderemos que se ejecuten a la vez, varios subprocesos (Threads) .La directiva ThreadsPerChild indica el número de hebras que pueden trabajar a la vez. El servidor funcionará mejor cuanto mas alto sea el valor.

```
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, in addition to the default. See also the <VirtualHost>
# directive.
#
#Listen 3000
#Listen 12.34.56.78:80
```

Código fuente 13. httpd.conf (continuación)

Con la directiva Listen vamos a enlazar nuestro servidor con una dirección IP, o con una combinación de dirección IP y un puerto o solo a un puerto, teniendo así la opción de utilizar otras opciones que no sean las que se marcan en la directiva Port.

```
# BindAddress: You can support virtual hosts with this option. This directive
# is used to tell the server which IP address to listen to. It can either
# contain "*", an IP address, or a fully qualified Internet domain name.
# See also the <VirtualHost> and Listen directives.
#
#BindAddress *
```

Código fuente 14. httpd.conf (continuación)

La directiva BindAddress indica a través de que dirección IP de la máquina se pueden escuchar las conexiones que tengan a través de ella. Si ponemos * podrá escuchar por cualquiera de ellas. Es importante cuando se utilizan servidores virtuales en Unix.

```
# WARNING: This is an advanced option that may render your server inoperable!
# Do not use these directives without expert guidance.
#
#ClearModuleList
#AddModule mod_so.c mod_mime.c mod_access.c mod_auth.c mod_negotiation.c
#AddModule mod_include.c mod_autoindex.c mod_dir.c mod_cgi.c mod_userdir.c
#AddModule mod_alias.c mod_env.c mod_log_config.c mod_asis.c mod_imap.c
#AddModule mod_actions.c mod_setenvif.c mod_isapi.c
```

Código fuente 15. httpd.conf (continuación)

La directiva AddModule se utiliza para Activar módulos precompilados que esté desactivado. El lector conocerá mas adelante la funcionlidad de estos módulos.

```
# Dynamic Shared Object (DSO) Support
#
# To be able to use the functionality of a module which was built as a DSO you
# have to place corresponding `LoadModule' lines at this location so the
# directives contained in it are actually available _before_ they are used.
# Please read the file README.DSO in the Apache 1.3 distribution for more
# details about the DSO mechanism and run `apache -l' for the list of already
# built-in (statically linked and thus always available) modules in your Apache
# binary.
#
# Note: The order in which modules are loaded is important.  Don't change
# the order below without expert advice.
#
#LoadModule anon_auth_module modules/ApacheModuleAuthAnon.dll
#LoadModule dbm_auth_module modules/ApacheModuleAuthDBM.dll
#LoadModule digest_auth_module modules/ApacheModuleAuthDigest.dll
#LoadModule cern_meta_module modules/ApacheModuleCERNMeta.dll
#LoadModule digest_module modules/ApacheModuleDigest.dll
#LoadModule expires_module modules/ApacheModuleExpires.dll
#LoadModule headers_module modules/ApacheModuleHeaders.dll
#LoadModule proxy_module modules/ApacheModuleProxy.dll
#LoadModule rewrite_module modules/ApacheModuleRewrite.dll
#LoadModule speling_module modules/ApacheModuleSpeling.dll
#LoadModule info_module modules/ApacheModuleInfo.dll
#LoadModule status_module modules/ApacheModuleStatus.dll
#LoadModule usertrack_module modules/ApacheModuleUserTrack.dll
```

Código fuente 16. httpd.conf (continuación)

Con la directiva LoadModule se podrá cargar en Apache aquellos archivos binarios, compilados posteriormente, pudiendo ser incluso ficheros de terceros. Un ejemplo de ello sería Apache-Tomcat, mediante el cual podremos ejecutar paginas JSP y servlets hechos en Java.

```
# ExtendedStatus controls whether Apache will generate "full" status
# information (ExtendedStatus On) or just basic information (ExtendedStatus
# Off) when the "server-status" handler is called. The default is Off.
#
#ExtendedStatus On
```

Código fuente 17. httpd.conf (continuación)

La sentencia ExtendedStatus, cuando esté en On generará toda la información posible, pero cuando está en off, solamente la basica. Esta directriz tiene mucho que ver con el modulo ApacheModule Status.dll

```
### Section 2: 'Main' server configuration
#
# The directives in this section set up the values used by the 'main'
# server, which responds to any requests that aren't handled by a
# <VirtualHost> definition.  These values also provide defaults for
# any <VirtualHost> containers you may define later in the file.
#
# All of these directives may appear inside <VirtualHost> containers,
# in which case these default settings will be overridden for the
# virtual host being defined.
#

#
# Port: The port to which the standalone server listens.  Certain firewall
# products must be configured before Apache can listen to a specific port.
# Other running httpd servers will also interfere with this port.  Disable
# all firewall, security, and other services if you encounter problems.
# To help diagnose problems use the Windows NT command NETSTAT -a
#
Port 80
```

Código fuente 18. httpd.conf (continuación)

Puerto en que el servidor estará escuchando. Por defecto es el 80. Para acceder a los puertos menores a 1024 hay que ser administrador Si no se tienen estos privilegios solo se puede ejecutar el servidor en los puertos 1024 a 327697. Si se especifica un puerto distinto del 80 hay que incluirlo en todas las URLs dirigidas al servidor. Por ejemplo, http://www.eidos.es:8800/pagina.html. Si Listen o BindAddress se utilizan para especificar un numero de puerto, Port no tendrá validez.

```
# ServerAdmin: Your address, where problems with the server should be
# e-mailed.  This address appears on some server-generated pages, such
# as error documents.
#
ServerAdmin adm@eidos.es
```

Código fuente 19. httpd.conf (continuación)

Cuando ocurre un error, el servidor saca una página con la dirección que tenga la directriz ServerAdmin, y todos los mensajes de error que se hayan producido.

```
# ServerName allows you to set a host name which is sent back to clients for
# your server if it's different than the one the program would get (i.e., use
# "www" instead of the host's real name).
#
# Note: You cannot just invent host names and hope they work. The name you
# define here must be a valid DNS name for your host. If you don't understand
# this, ask your network administrator.
# If your host doesn't have a registered DNS name, enter its IP address here.
# You will have to access it by its address (e.g., http://123.45.67.89/)
# anyway, and this will make redirections work in a sensible way.
#
# 127.0.0.1 is the TCP/IP local loop-back address, often named localhost. Your
# machine always knows itself by this address. If you use Apache strictly for
# local testing and development, you may use 127.0.0.1 as the server name.
#
#ServerName new.host.name
```

Código fuente 20. httpd.conf (continuación)

Recordará el lector que cuando arrancamos el servidor por primera vez, aparecía un mensaje de alerta avisándonos de la no existencia de un nombre en ServerName y que asumía la que tenía como localhost. Pues bien, es con esta instrucción con la que le vamos a dar un nombre a nuestro servidor Apache. El lector verá mas adelante, cuando tratemos los servidores virtuales, que con esta instrucción les vamos a asignar también su nombre.

```
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "E:/Apache/htdocs"
```

Código fuente 21. httpd.conf (continuación)

Con la instrucción DocumentRoot vamos a asignar cual será el path base de nuestro servidor, a partir del cual colgarán todos los ficheros y subdirectorios necesarios para la web del servidor.

```
# Each directory to which Apache has access, can be configured with respect
# to which services and features are allowed and/or disabled in that
# directory (and its subdirectories).
#
# First, we configure the "default" to be a very restrictive set of
# permissions.
#
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

#
# Note that from this point forward you must specifically allow
# particular features to be enabled - so if something's not working as
# you might expect, make sure that you have specifically enabled it
# below.
```

```
# This should be changed to whatever you set DocumentRoot to.
#
<Directory "E:/Apache/htdocs">

#
# This may also be "None", "All", or any combination of "Indexes",
# "Includes", "FollowSymLinks", "ExecCGI", or "MultiViews".
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
    Options Indexes FollowSymLinks MultiViews

#
# This controls which options the .htaccess files in directories can
# override. Can also be "All", or any combination of "Options", "FileInfo",
# "AuthConfig", and "Limit"
#
    AllowOverride None

#
# Controls who can get stuff from this server.
#
    Order allow,deny
    Allow from all
</Directory>
<Directory "E:/Applets Java">

#
# This may also be "None", "All", or any combination of "Indexes",
# "Includes", "FollowSymLinks", "ExecCGI", or "MultiViews".
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
    Options Indexes FollowSymLinks MultiViews

#
# This controls which options the .htaccess files in directories can
# override. Can also be "All", or any combination of "Options", "FileInfo",
# "AuthConfig", and "Limit"
#
    AllowOverride None

#
# Controls who can get stuff from this server.
#
    Order allow,deny
    Allow from all
</Directory>
```

Código fuente 22. httpd.conf (continuación)

La directriz Directory se utiliza para para agrupar varias directrices que afectan a uno o varios directorios del sistema operativo, así como a sus subdirectorios. Lo que conseguimos con esta directriz es crear directorios protegidos. Dentro de cada agrupación, se pueden insertar otras opciones que afecten a dicho directorio. Como verá el lector en el Código fuente 22, dentro de cada directriz de agrupación existen varias instrucciones que afectaran la directorio indicado.

La primera que nos encontraremos será la directriz Options, que como propiedades mas importanles tiene las siguientes:

- Indexes indica que si hay un indice, señalado en la instrucción DirectoryIndex, se muestra si no se crea uno nuevo. El ejemplo de la Figura 18 muestra un indice creado automáticamente al no encontrar ningún fichero con la extensión indicada en DirectoryIndex.

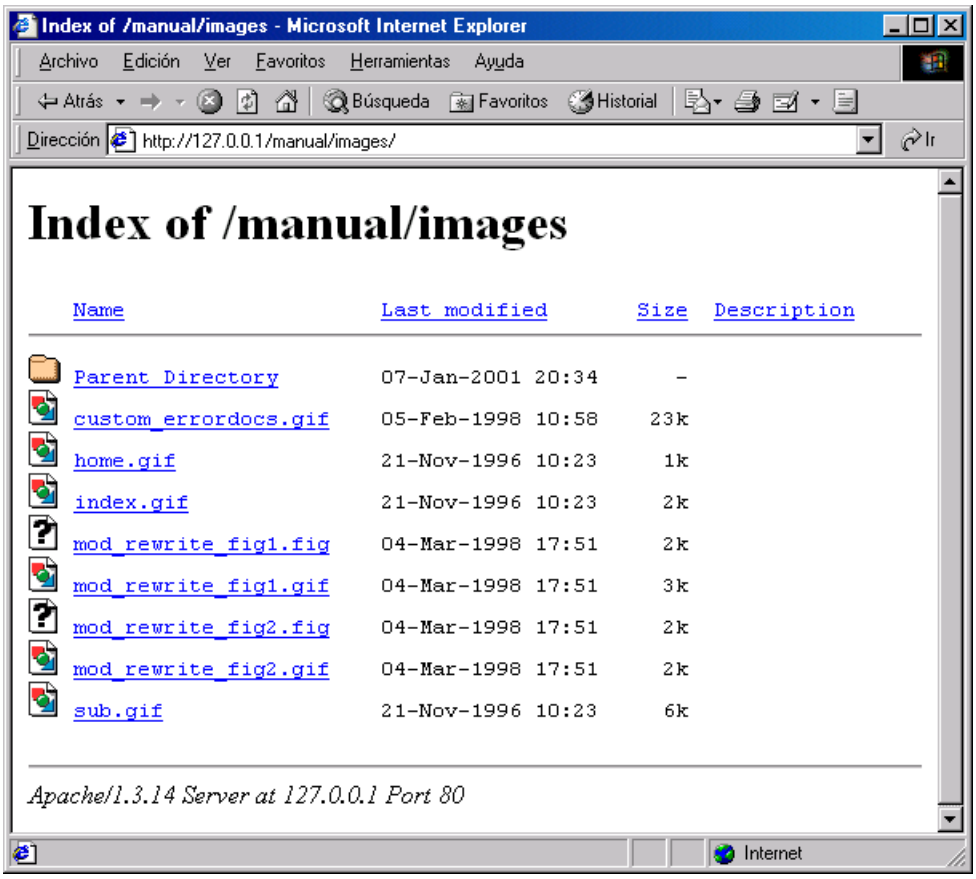


Figura 18. Indice creado automáticamente

- FollowSymlinks se utiliza para indicar que es posible la utilización de directorios virtuales. El servidor seguirá los vínculos simbolicos que apunten a los directorios de sistema, sin cambiar el path indicado en la directriz Directory.

- MultiViews permite que el contenido de un documento pueda ser negociado, basándose en el lenguaje que se emplee.

Otra directriz que nos encontraremos en Directory es AllowOverride, que controla el ambito de aplicación de las restricciones de acceso, puesto que la configuración por defecto puede colisionar con la configuración de cada directorio protegido. Cuando se el pasa el parámetro none, se indica que no se puede sobrescribir ninguna opción.

Cuando leamos dentro de la directriz Directory `Order allow,deny Allow from all` estamos indicando al directorio que se permite acceso a todo. Al escribir Order allow, deny, se está indicando que las directivas allow se leen antes que las deny. Si escribiésemos Order deny, allow seria lo contrario. Al escribir Allow from all, tiene prioridad la concesión de permiso.

```
# UserDir: The name of the directory which is appended onto a user's home
# directory if a ~user request is received.
```

```
#
# Under Win32, we do not currently try to determine the home directory of
# a Windows login, so a format such as that below needs to be used.  See
# the UserDir documentation for details.
#
<IfModule mod_userdir.c>
    UserDir "E:/Apache/users/"
</IfModule>
#
# Control access to UserDir directories.  The following is an example
# for a site where these directories are restricted to read-only.
#
#<Directory "E:/Apache/users">
#    AllowOverride FileInfo AuthConfig Limit
#    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
#    <Limit GET POST OPTIONS PROPFIND>
#        Order allow,deny
#        Allow from all
#    </Limit>
#    <LimitExcept GET POST OPTIONS PROPFIND>
#        Order deny,allow
#        Deny from all
#    </LimitExcept>
#</Directory>
```

Código fuente 23. httpd.conf (continuación)

La directriz UserDir se utiliza para indicar un directorio que se ha de considerar como DocumentRoot cuando existen varios usuarios, y queramos que cada uno tenga su sitio web. La opción Limit, que aparece en el Código fuente 23, se utiliza para encapsular métodos de acceso a HTTP.

```
# DirectoryIndex: Name of the file or files to use as a pre-written HTML
# directory index.  Separate multiple entries with spaces.
#
<IfModule mod_dir.c>
    DirectoryIndex index.html
</IfModule>
```

Código fuente 24. httpd.conf (continuación)

La directiva DirectoryIndex define una lista de recursos a buscar cuando el cliente realiza una petición en el directorio, y este no ha escrito nada después del slash final en una URL. Usualmente, será el nombre de un fichero, y usará el primero que encuentre. Si no existiese ningun recurso, si la opción indexes está activa, el servidor generará su propia lista. El recurso tambien pedria ser una url relativa.

```
# AccessFileName: The name of the file to look for in each directory
# for access control information.
#
AccessFileName .htaccess
#
# The following lines prevent .htaccess files from being viewed by
# Web clients.  Since .htaccess files often contain authorization
# information, access is disallowed for security reasons.  Comment
# these lines out if you want Web visitors to see the contents of
# .htaccess files.  If you change the AccessFileName directive above,
# be sure to make the corresponding changes here.
#
```

```
# Also, folks tend to use names such as .htpasswd for password
# files, so this will protect those as well.
#
<Files ~ "^\.ht">
    Order allow,deny
    Deny from all
</Files>
```

Código fuente 25. httpd.conf (continuación)

Mientras que se retorna un documento al cliente, el servidor busca el primer fichero de control de acceso existente en AccessFileName, para determinar si el acceso es correcto.

La directiva Files permite una gestión de control de acceso fichero por fichero. Debe incluir un archivo o un comodín relativo a una cadena de caracteres. Las secciones internas son tratadas en el orden en el que aparecen en el fichero de configuración.

```
# CacheNegotiatedDocs: By default, Apache sends "Pragma: no-cache" with each
# document that was negotiated on the basis of content. This asks proxy
# servers not to cache the document. Uncommenting the following line disables
# this behavior, and proxies will be allowed to cache the documents.
#CacheNegotiatedDocs
```

Código fuente 26. httpd.conf (continuación)

Si esta activo, el servidor permite a los documentos resultantes de un proceso de negociación de contenido, ser cacheados por un servidor proxy.

```
# UseCanonicalName:  (new for 1.3)  With this setting turned on, whenever
# Apache needs to construct a self-referencing URL (a URL that refers back
# to the server the response is coming from) it will use ServerName and
# Port to form a "canonical" name.  With this setting off, Apache will
# use the hostname:port that the client supplied, when possible.  This
# also affects SERVER_NAME and SERVER_PORT in CGI scripts.
UseCanonicalName On
```

Código fuente 27. httpd.conf (continuación)

Cuando la directriz UseCanonicalName esta activa, si se necesita construirse una referencia URL, usará ServerName más Port, formando un nombre comprensible.

```
# TypesConfig describes where the mime.types file (or equivalent) is
# to be found.
#
<IfModule mod_mime.c>
    TypesConfig conf/mime.types
</IfModule>
```

Código fuente 28. httpd.conf (continuación)

La directriz TypesConfig define el fichero de configuración de tipos mime.

```
# DefaultType is the default MIME type the server will use for a document
# if it cannot otherwise determine one, such as from filename extensions.
# If your server contains mostly text or HTML documents, "text/plain" is
# a good value.  If most of your content is binary, such as applications
# or images, you may want to use "application/octet-stream" instead to
# keep browsers from trying to display binary files as though they are
# text.
DefaultType text/plain
```

Código fuente 29. httpd.conf (continuación)

Se puede llegar a requerir de un servidor un documento en donde el tipo mime no pueda ser determinado por las tablas de MIME. Es entonces cuando se utilizará el tipo por defecto indicado en la directriz DefaultType.

```
# The mod_mime_magic module allows the server to use various hints from the
# contents of the file itself to determine its type.  The MIMEMagicFile
# directive tells the module where the hint definitions are located.
# mod_mime_magic is not part of the default server (you have to add
# it yourself with a LoadModule [see the DSO paragraph in the 'Global
# Environment' section], or recompile the server and include mod_mime_magic
# as part of the configuration), so it's enclosed in an <IfModule> container.
# This means that the MIMEMagicFile directive will only be processed if the
# module is part of the server.
<IfModule mod_mime_magic.c>
    MIMEMagicFile conf/magic
</IfModule>
```

Código fuente 30. httpd.conf (continuación)

La directiva MimeMagicFile define el nombre del fichero magic, necesario para el módulo mod_mime_magic, utilizado para determinar el contenido de un documento leyendo los primeros octetos del mismo.

```
# HostnameLookups: Log the names of clients or just their IP addresses
# e.g., www.apache.org (on) or 204.62.129.132 (off).
# The default is off because it'd be overall better for the net if people
# had to knowingly turn this feature on, since enabling it means that
# each client request will result in AT LEAST one lookup request to the
# nameserver.
#
HostnameLookups Off
```

Código fuente 31. httpd.conf (continuación)

La directiva HostnameLookups autoriza la resolución de nombre DNS en cada petición. Cuando esta activa Apache guarda el nombre del servidor del cliente en una variable de entorno (REMOTE_ HOST).

```
# ErrorLog: The location of the error log file.
# If you do not specify an ErrorLog directive within a <VirtualHost>
```

```
# container, error messages relating to that virtual host will be
# logged here.  If you *do* define an error logfile for a <VirtualHost>
# container, that host's errors will be logged there and not here.
#
ErrorLog logs/error.log

#
# LogLevel: Control the number of messages logged to the error.log.
# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
#
LogLevel warn

#
# The following directives define some format nicknames for use with
# a CustomLog directive (see below).
#
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

#
# The location and format of the access logfile (Common Logfile Format).
# If you do not define any access logfiles within a <VirtualHost>
# container, they will be logged here.  Contrariwise, if you *do*
# define per-<VirtualHost> access logfiles, transactions will be
# logged therein and *not* in this file.
#
CustomLog logs/access.log common

#
# If you would like to have agent and referer logfiles, uncomment the
# following directives.
#
#CustomLog logs/referer.log referer
#CustomLog logs/agent.log agent
#
# If you prefer a single logfile with access, agent, and referer information
# (Combined Logfile Format) you can use the following directive.
#CustomLog logs/access.log combined
```

Código fuente 32. httpd.conf (continuación)

Los errores en el servidor van a ser tratados por varias directrices.La directiva ErrorLog define el nombre del fichero en el cual el servidor señalará los errores que se ocasionen. LogLevel nos indicará el modo como se escribirán los errores en ErrorLog, mientras que la directiva LogFormat se encargará de definir el formato con el que se almacenarán los errores. CustomLog , por el contrario permite dar un formato personalizado que deberes indicar como argumento.

```
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (error documents, FTP directory listings,
# mod_status and mod_info output etc., but not CGI generated documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of:  On | Off | EMail
ServerSignature On
```

Código fuente 33. httpd.conf (continuación)

Cuando la directiva ServerSignature está activada, el servidor permite un pie de línea en aquellos documentos generados por el servidor (mensajes de error, registros de mod_proxy, etc)

```
# Apache parses all CGI scripts for the shebang line by default.
# This comment line, the first line of the script, consists of the symbols
# pound (#) and exclamation (!) followed by the path of the program that
# can execute this specific script.  For a perl script, with perl.exe in
# the C:\Program Files\Perl directory, the shebang line should be:

   #!c:/program files/perl/perl

# Note you _must_not_ indent the actual shebang line, and it must be the
# first line of the file.  Of course, CGI processing must be enabled by
# the appropriate ScriptAlias or Options ExecCGI directives for the files
# or directory in question.
#
# However, Apache on Windows allows either the Unix behavior above, or can
# use the Registry to match files by extention.  The command to execute
# a file of this type is retrieved from the registry by the same method as
# the Windows Explorer would use to handle double-clicking on a file.
# These script actions can be configured from the Windows Explorer View menu,
# 'Folder Options', and reviewing the 'File Types' tab.  Clicking the Edit
# button allows you to modify the Actions, of which Apache 1.3 attempts to
# perform the 'Open' Action, and failing that it will try the shebang line.
# This behavior is subject to change in Apache release 2.0.
#
# Each mechanism has it's own specific security weaknesses, from the means
# to run a program you didn't intend the website owner to invoke, and the
# best method is a matter of great debate.
#
# To enable the this Windows specific behavior (and therefore -disable- the
# equivilant Unix behavior), uncomment the following directive:
#
#ScriptInterpreterSource registry
#
# The directive above can be placed in individual <Directory> blocks or the
# .htaccess file, with either the 'registry' (Windows behavior) or 'script'
# (Unix behavior) option, and will override this server default option.
#

#
# Aliases: Add here as many aliases as you need (with no limit). The format is
# Alias fakename realname
#
<IfModule mod_alias.c>

    #
    # Note that if you include a trailing / on fakename then the server will
    # require it to be present in the URL.  So "/icons" isn't aliased in this
    # example, only "/icons/"..
    #
    Alias /icons/ "E:/Apache/icons/"

    <Directory "E:/Apache/icons">
        Options Indexes MultiViews
        AllowOverride None
        Order allow,deny
        Allow from all
    </Directory>

    #
    # ScriptAlias: This controls which directories contain server scripts.
    # ScriptAliases are essentially the same as Aliases, except that
    # documents in the realname directory are treated as applications and
```

```
    # run by the server when requested rather than as documents sent to the client.
    # The same rules about trailing "/" apply to ScriptAlias directives as to
    # Alias.
    #
    ScriptAlias /cgi-bin/ "E:/Apache/cgi-bin/"

    #
    # "E:/Apache/cgi-bin" should be changed to whatever your ScriptAliased
    # CGI directory exists, if you have that configured.
    #
    <Directory "E:/Apache/cgi-bin">
        AllowOverride None
        Options None
        Order allow,deny
        Allow from all
    </Directory>

</IfModule>
# End of aliases.
```

Código fuente 34. httpd.conf (continuación)

La directiva Alias permite transformar cualquier ruta de ficheros en otros emplazamientos del sistema tema, a partir del path marcado por DocumentRoot.La directriz ScriptAlias solamente permite asignar un alias (URL-path) de un path, no pudiendo convertirlo, por ejemplo en un nombre de fichero. Como habra comprobado el lector, estas dos directrices solo actuarán si se carga el módulo mod_alias.c

```
# Redirect allows you to tell clients about documents which used to exist in
# your server's namespace, but do not anymore. This allows you to tell the
# clients where to look for the relocated document.
# Format: Redirect old-URI new-URL
#

#
# Directives controlling the display of server-generated directory listings.
#
<IfModule mod_autoindex.c>

    #
    # FancyIndexing is whether you want fancy directory indexing or standard
    #
    IndexOptions FancyIndexing

    #
    # AddIcon* directives tell the server which icon to show for different
    # files or filename extensions.  These are only displayed for
    # FancyIndexed directories.
    #
    AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip

    AddIconByType (TXT,/icons/text.gif) text/*
    AddIconByType (IMG,/icons/image2.gif) image/*
    AddIconByType (SND,/icons/sound2.gif) audio/*
    AddIconByType (VID,/icons/movie.gif) video/*

    AddIcon /icons/binary.gif .bin .exe
    AddIcon /icons/binhex.gif .hqx
    AddIcon /icons/tar.gif .tar
    AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
    AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
    AddIcon /icons/a.gif .ps .ai .eps
    AddIcon /icons/layout.gif .html .shtml .htm .pdf
```

```
    AddIcon /icons/text.gif .txt
    AddIcon /icons/c.gif .c
    AddIcon /icons/p.gif .pl .py
    AddIcon /icons/f.gif .for
    AddIcon /icons/dvi.gif .dvi
    AddIcon /icons/uuencoded.gif .uu
    AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
    AddIcon /icons/tex.gif .tex
    AddIcon /icons/bomb.gif core

    AddIcon /icons/back.gif ..
    AddIcon /icons/hand.right.gif README
    AddIcon /icons/folder.gif ^^DIRECTORY^^
    AddIcon /icons/blank.gif ^^BLANKICON^^

    #
    # DefaultIcon is which icon to show for files which do not have an icon
    # explicitly set.
    #
    DefaultIcon /icons/unknown.gif

    #
    # AddDescription allows you to place a short description after a file in
    # server-generated indexes.  These are only displayed for FancyIndexed
    # directories.
    # Format: AddDescription "description" filename
    #
    #AddDescription "GZIP compressed document" .gz
    #AddDescription "tar archive" .tar
    #AddDescription "GZIP compressed tar archive" .tgz
    AddDescription "Archivo de imagen" .gif
    #
    # ReadmeName is the name of the README file the server will look for by
    # default, and append to directory listings.
    #
    # HeaderName is the name of a file which should be prepended to
    # directory indexes.
    #
    # If MultiViews are amongst the Options in effect, the server will
    # first look for name.html and include it if found.  If name.html
    # doesn't exist, the server will then look for name.txt and include
    # it as plaintext if found.
    #
    ReadmeName README
    HeaderName HEADER

    #
    # IndexIgnore is a set of filenames which directory indexing should ignore
    # and not include in the listing.  Shell-style wildcarding is permitted.
    #
    IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t

</IfModule>
# End of indexing directives.
```

Código fuente 35. httpd.conf (continuación)

Las directivas AddIcon* definen los iconos que se asociarán a cada fichero en función de su extensión, cuando se cree automáticamente índice de ficheros si es que no existe ninguno de los indicados en DirectoryIndex. En la figura 19, el lector podrá comprobar que según sea la extensión de los ficheros , así se le asociará un icono grafico, al igual que los directorios, que también tienen el suyo asociado. También podemos indicar algun tipo de comentario para cada fichero. El lector puede comprobar en la Figura 19 la diferencia que existe con la Figura 18, después de haber incluido el fichero de configuración la línea:

```
AddDescription "Archivo de imagen" .gif
```

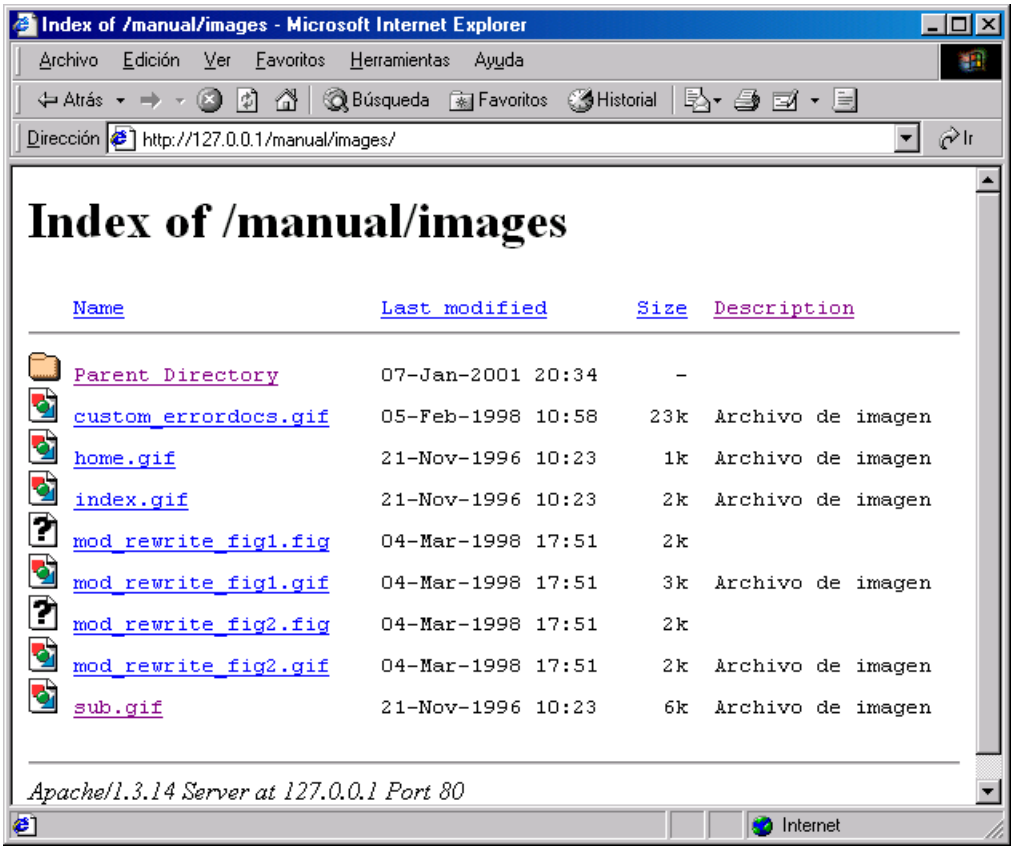


Figura 19. Indice creado automáticamente con comentarios

Cuando usamos para crear el índice las directivas ReadmeName y HeaderName, le estamos indicando a la pagina como debe ser el pie de pagina y la cabecera, respectivamente. Si encuentra un fichero llamado readme.html en el directorio mostrado se añadirá al final del índice, pero si además existiera otro fichero llamado header.html, pasado como parámetro a HeaderName, su contenido aparecería en la cabecera del índice.

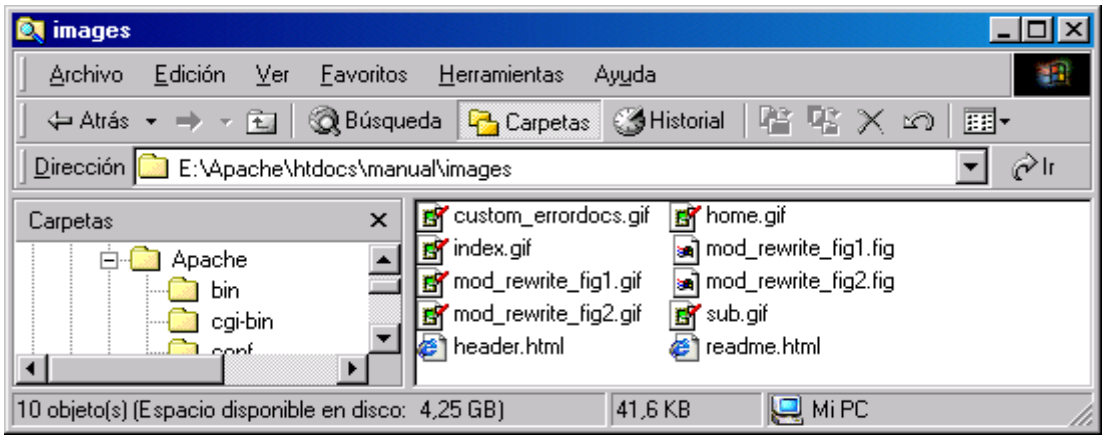


Figura 20. Contenido de la carpeta mostrada en la figura 22 en el sistema operativo Windows

Como vera el lector, los ficheros con extenxión html no aparecen en el contenido del directorio del índice de la Figura 21, pero si en la carpeta del sistema operativo en la Figura 20.

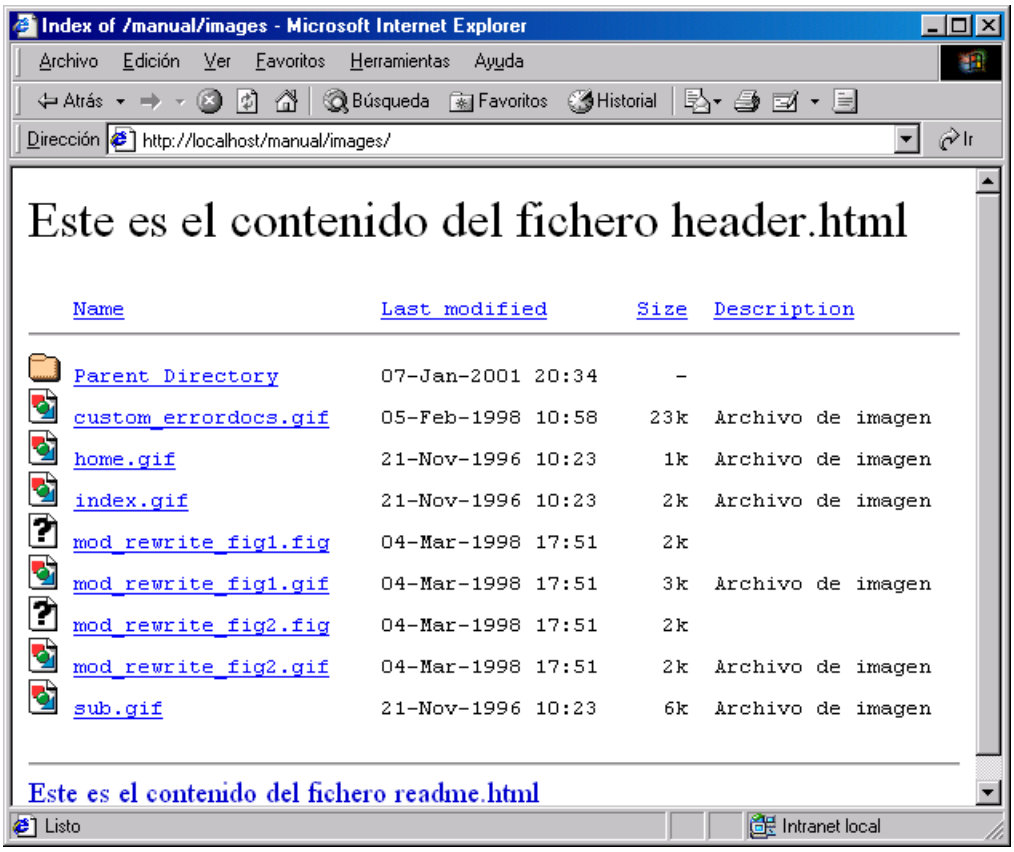


Figura 21. ReadmeName y HeaderName

La directiva AddEncoding define como una extensión o lista de extensiones deben finalizar para ser considerados como una codificación especifica de tipo MIME. Tal y como se muestra la definición en el Código fuente 36, se considerará que un archivo con extensión gz o tgz, se corresponde con x-gzip .

```
<IfModule mod_mime.c>

    #
    # AddEncoding allows you to have certain browsers (Mosaic/X 2.1+) uncompress
    # information on the fly. Note: Not all browsers support this.
    # Despite the name similarity, the following Add* directives have nothing
    # to do with the FancyIndexing customization directives above.
    #
    AddEncoding x-compress Z
    AddEncoding x-gzip gz tgz
```

Código fuente 36. httpd.conf (continuación)

```
# AddLanguage allows you to specify the language of a document. You can
    # then use content negotiation to give a browser a file in a language
    # it can understand.
    #
    # Note 1: The suffix does not have to be the same as the language
    # keyword --- those with documents in Polish (whose net-standard
    # language code is pl) may wish to use "AddLanguage pl .po" to
    # avoid the ambiguity with the common suffix for perl scripts.
    #
```

```
    # Note 2: The example entries below illustrate that in quite
    # some cases the two character 'Language' abbriviation is not
    # identical to the two character 'Country' code for its country,
    # E.g. 'Danmark/dk' versus 'Danish/da'.
    #
    # Note 3: In the case of 'ltz' we violate the RFC by using a three char
    # specifier. But there is 'work in progress' to fix this and get
    # the reference data for rfc1766 cleaned up.
    #
    # Danish (da) - Dutch (nl) - English (en) - Estonian (ee)
    # French (fr) - German (de) - Greek-Modern (el)
    # Italian (it) - Korean (kr) - Norwegian (no)
    # Portugese (pt) - Luxembourgeois* (ltz)
    # Spanish (es) - Swedish (sv) - Catalan (ca) - Czech(cz)
    # Polish (pl) - Brazilian Portuguese (pt-br) - Japanese (ja)
    # Russian (ru)
    #
    AddLanguage da .dk
    AddLanguage nl .nl
    AddLanguage en .en
    AddLanguage et .ee
    AddLanguage fr .fr
    AddLanguage de .de
    AddLanguage el .el
    AddLanguage he .he
    AddCharset ISO-8859-8 .iso8859-8
    AddLanguage it .it
    AddLanguage ja .ja
    AddCharset ISO-2022-JP .jis
    AddLanguage kr .kr
    AddCharset ISO-2022-KR .iso-kr
    AddLanguage no .no
    AddLanguage pl .po
    AddCharset ISO-8859-2 .iso-pl
    AddLanguage pt .pt
    AddLanguage pt-br .pt-br
    AddLanguage ltz .lu
    AddLanguage ca .ca
    AddLanguage es .es
    AddLanguage sv .se
    AddLanguage cz .cz
    AddLanguage ru .ru
    AddLanguage tw .tw
    AddCharset Big5         .Big5    .big5
    AddCharset WINDOWS-1251 .cp-1251
    AddCharset CP866        .cp866
    AddCharset ISO-8859-5   .iso-ru
    AddCharset KOI8-R       .koi8-r
    AddCharset UCS-2        .ucs2
    AddCharset UCS-4        .ucs4
    AddCharset UTF-8        .utf8
```

Código fuente 37. httpd.conf (continuación)

La directiva AddLanguage convierte las extensiones que se indiquen en un lenguaje MIME determinado. Por ejemplo, en el Código fuente 37 aparece la directiva `AddLanguage es .es` que está indicando que todos los archivos con extensión .es seran abiertos para el idioma es. Por lo tanto, cuando un servidor encuentre el mismo archivo, pero con distintas extensiones, mostrará aquel que corresponda con la configuración del país. La directiva AddCharset se utiliza para informar a la petición cliente sobre el carácter que debe poner en el código del documento.

```
# LanguagePriority allows you to give precedence to some languages
```

```
    # in case of a tie during content negotiation.
    #
    # Just list the languages in decreasing order of preference. We have
    # more or less alphabetized them here. You probably want to change this.
    #
    <IfModule mod_negotiation.c>
        LanguagePriority en da nl et fr de el it ja kr no pl pt pt-br ru ltz ca es
sv tw
    </IfModule>
```

Código fuente 38. httpd.conf (continuación)

La directiva LanguagePriority indica qué preferencia de idioma utilizará el servidor cuando el cliente no ha indicado ninguna preferencia. El orden de la lista de valores se indica de forma decreciente.

```
    # AddType allows you to tweak mime.types without actually editing it, or to
    # make certain files to be certain types.
    #
    # For example, the PHP 3.x module (not part of the Apache distribution - see
    # http://www.php.net) will typically use:
    #
    #AddType application/x-httpd-php3 .php3
    #AddType application/x-httpd-php3-source .phps
    #
    # And for PHP 4.x, use:
    #
    #AddType application/x-httpd-php .php
    #AddType application/x-httpd-php-source .phps

    AddType application/x-tar .tgz

    #
    # AddHandler allows you to map certain file extensions to "handlers",
    # actions unrelated to filetype. These can be either built into the server
    # or added with the Action command (see below)
    #
    # If you want to use server side includes, or CGI outside
    # ScriptAliased directories, uncomment the following lines.
    #
    # To use CGI scripts:
    #
    #AddHandler cgi-script .cgi

    #
    # To use server-parsed HTML files
    #
    #AddType text/html .shtml
    #AddHandler server-parsed .shtml

    #
    # Uncomment the following line to enable Apache's send-asis HTTP file
    # feature
    #
    #AddHandler send-as-is asis

    #
    # If you wish to use server-parsed imagemap files, use
    #
    #AddHandler imap-file map

    #
    # To enable type maps, you might want to use
    #
```

```
    #AddHandler type-map var

</IfModule>
# End of document types.
```

Código fuente 39. httpd.conf (continuación)

La directiva AddHandler define un controlador para una o mas tensiones. Por ejemplo, cuando escribimos `AddHandler server-parsed .shtml` estamos indicando al servidor que si encuentra una pagina con extensión .shtml, debe ser procesada por el controlador server-parsed del módulo mod_include. La directiva AddType convierte una lista de extensiones en tipo-MIME.

```
# MetaDir: specifies the name of the directory in which Apache can find
# meta information files. These files contain additional HTTP headers
# to include when sending the document
#
#MetaDir .web

#
# MetaSuffix: specifies the file name suffix for the file containing the
# meta information.
#
#MetaSuffix .meta
```

Código fuente 40. httpd.conf (continuación)

La directiva MetaDir especifica el nombre del directorio que se utiliza para guardar metainformación. Esta se almacena en un archivo que aparece en la cabecera de la respuesta http. La extensión de dicho archivo viene indicada por la directiva MetaSuffix.

```
# Customizable error response (Apache style)
#  these come in three flavors
#
#    1) plain text
#ErrorDocument 500 "The server made a boo boo.
#  n.b.  the single leading (") marks it as text, it does not get output
#
#    2) local redirects
#ErrorDocument 404 /missing.html
#  to redirect to local URL /missing.html
#ErrorDocument 404 /cgi-bin/missing_handler.pl
#  N.B.: You can redirect to a script or a document using server-side-includes.
#
#    3) external redirects
#ErrorDocument 402 http://some.other_server.com/subscription_info.html
#  N.B.: Many of the environment variables associated with the original
#  request will *not* be available to such a script.

#
# Customize behaviour based on the browser
#
<IfModule mod_setenvif.c>

    #
    # The following directives modify normal HTTP response behavior.
    # The first directive disables keepalive for Netscape 2.x and browsers that
    # spoof it. There are known problems with these browser implementations.
```

```
    # The second directive is for Microsoft Internet Explorer 4.0b2
    # which has a broken HTTP/1.1 implementation and does not properly
    # support keepalive when it is used on 301 or 302 (redirect) responses.
    #
    BrowserMatch "Mozilla/2" nokeepalive
    BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0

    #
    # The following directive disables HTTP/1.1 responses to browsers which
    # are in violation of the HTTP/1.0 spec by not being able to grok a
    # basic 1.1 response.
    #
    BrowserMatch "RealPlayer 4\.0" force-response-1.0
    BrowserMatch "Java/1\.0" force-response-1.0
    BrowserMatch "JDK/1\.0" force-response-1.0

</IfModule>
# End of browser customization directives
```

Código fuente 41 httpd.conf (continuación)

La directiva BrowserMatch define la variable de entorno sobre la base del contenido del campo User-Agent en la cabecera HTTP. Por ejemplo, como verá el lector en el Código fuente 40, cuando encuentra en la cabecera la palabra JDK/1\.0, force-response tomará el valor 1, al no haberle indicado ningún otro valor.

```
# Allow server status reports, with the URL of http://servername/server-status
# Change the ".your_domain.com" to match your domain to enable.
#
#<Location /server-status>
#    SetHandler server-status
#    Order deny,allow
#    Deny from all
#    Allow from .your_domain.com
#</Location>

# Allow remote server configuration reports, with the URL of
# http://servername/server-info (requires that mod_info.c be loaded).
# Change the ".your_domain.com" to match your domain to enable.

#<Location /server-info>
#    SetHandler server-info
#    Order deny,allow
#    Deny from all
#    Allow from .your_domain.com
#</Location>

# There have been reports of people trying to abuse an old bug from pre-1.1
# days.  This bug involved a CGI script distributed as a part of Apache.
# By uncommenting these lines you can redirect these attacks to a logging
# script on phf.apache.org.  Or, you can record them yourself, using the script
# support/phf_abuse_log.cgi.

#<Location /cgi-bin/phf*>
#    Deny from all
#    ErrorDocument 403 http://phf.apache.org/phf_abuse_log.cgi
#</Location>
```

Código fuente 42. httpd.conf (continuación)

Con la directiva Location se podrá controlar el acceso a través de una URL. Es similar a la directiva Directory, por lo tanto necesita terminar con </Location>

```
# Proxy Server directives. Uncomment the following lines to
# enable the proxy server:
#
#<IfModule mod_proxy.c>
#    ProxyRequests On

#    <Directory proxy:*>
#        Order deny,allow
#        Deny from all
#        Allow from .your_domain.com
#    </Directory>

    #
    # Enable/disable the handling of HTTP/1.1 "Via:" headers.
    # ("Full" adds the server version; "Block" removes all outgoing Via: headers)
    # Set to one of: Off | On | Full | Block
    #
#    ProxyVia On

    #
    # To enable the cache as well, edit and uncomment the following lines:
    # (no cacheing without CacheRoot)
    #
#    CacheRoot "E:/Apache/proxy"
#    CacheSize 5
#    CacheGcInterval 4
#    CacheMaxExpire 24
#    CacheLastModifiedFactor 0.1
#    CacheDefaultExpire 1
#    NoCache a_domain.com another_domain.edu joes.garage_sale.com

#</IfModule>
# End of proxy directives.
```

Código fuente 43. httpd.conf (continuación)

Si quisiéramos utilizar un servidor proxy con Apache lo deberíamos hacer utilizando el modulo mod_proxy, y quitando los comentarios en el Código fuente 43. Las directrices mas importantes para utilizar el servidor como proxy son ProxyRequests, ProxyVia. La primera permite activar el servicio cache, mientras que la segunda permite, si está activa, cambiar el titulo en la cabecera HTTP en la petición con el servidor proxy.

```
### Section 3: Virtual Hosts
#
# VirtualHost: If you want to maintain multiple domains/hostnames on your
# machine you can setup VirtualHost containers for them. Most configurations
# use only name-based virtual hosts so the server doesn't need to worry about
# IP addresses. This is indicated by the asterisks in the directives below.
#
# Please see the documentation at <URL:http://www.apache.org/docs/vhosts/>
# for further details before you try to setup virtual hosts.
#
# You may use the command line option '-S' to verify your virtual host
# configuration.

#
```

```
# Use name-based virtual hosting.
#
#NameVirtualHost *

#
# VirtualHost example:
# Almost any Apache directive may go into a VirtualHost container.
# The first VirtualHost section is used for requests without a known
# server name.
#
#<VirtualHost *>
#    ServerAdmin webmaster@dummy-host.example.com
#    DocumentRoot /www/docs/dummy-host.example.com
#    ServerName dummy-host.example.com
#    ErrorLog logs/dummy-host.example.com-error_log
#    CustomLog logs/dummy-host.example.com-access_log common
#</VirtualHost>
```

Código fuente 44. httpd.conf (Final)

Con la directiva VirtualHost, Apache nos va a permitir utilizar servidores virtuales en una misma máquina, pudiendo asignar otras directivas, dentro de esta, que den características particulares a dicho servidor.

Una vez que se haya configurado el fichero http.conf , ya estamos en condiciones de poner en marcha nuestro servidor web. Recuerde el lector que en Unix y en Linux, lo haremos con el fichero binario httpd y en Windows lo haremos con el ejecutable apache.exe. Todos los parámetros que vayamos a indicar se utilizan indistintamente en todos los sistemas operativos. Nosotros utilizaremos la ejecución en Windows para ver la puesta en marcha del servidor, teniendo en cuenta siempre todo lo que hemos configurado anteriormente.

Ejecución de Apache

Como habrá visto el lector, la tarea de configuración del servidor es minuciosa, aunque por suerte ya nos viene hecha. La tarea que vamos a comenzar ahora, también la tenemos resuelta, pero es conveniente que conozcamos que está pasando cuando realizamos la ejecución de un programa de Apache.

Ya vimos que toda nuestra tarea se concentra en que en la primera ejecución en el navegador veamos la pantalla que nos aparece en la Figura 12, pero para ello debemos disponer de Apache como un servicio más, o como un programa en ejecución (como un demonio en Unix).

Las opciones siguientes son las que utilizaremos en la línea de comandos para utilizar el servidor Apache:

`-d` *serverroot*: Podemos indicar valor inicial ServerRoot en la linea de comandos, que posteriormente sera sustituido por la orden ServerRoot del archivo de configuración. El valor por defecto es /usr/local/apache en Unix, /apache en Windows y /os2httpd en OS/2.

`-D` *name*: Define un nombre para usar con la directiva IfDefine. Esta opción puede usarse para habilitar cierta funcionalidad opcional en el archivo de configuración.

`-f` *config*: Ejecuta las directivas de configuración del fichero *config* en el arranque. Si *config* no comienza por /, entonces se considera el acceso relativo a la posición dadad por ServerRoot. El valor por defecto es `conf/httpd.conf`.

`-C "directive"`: Procesa la directiva indicada antes de leer el archivo de configuración.

`-c "directive"`: Procesa la directiva indicada antes de leer el archivo de configuración.

`-X`: Ejecuta en modo simple para depuración. No usar cuando el servidor se pone en producción.

`-v`: Imprime la versión de compilación y su fecha.

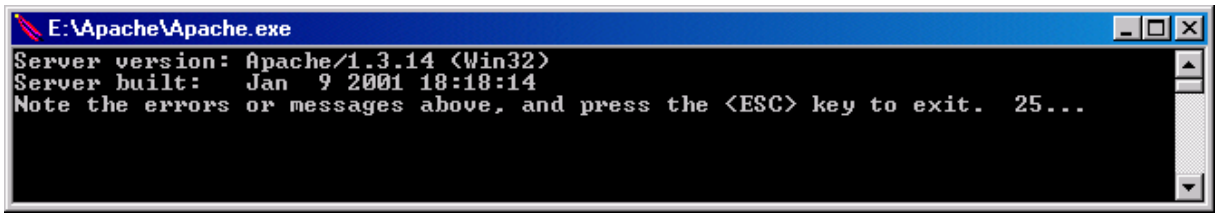


Figura 22. Apache.exe -v

`-V`: Imprime la versión de compilación y su fecha, así como una lista de parámetros que han sido utilizados en la compilación.

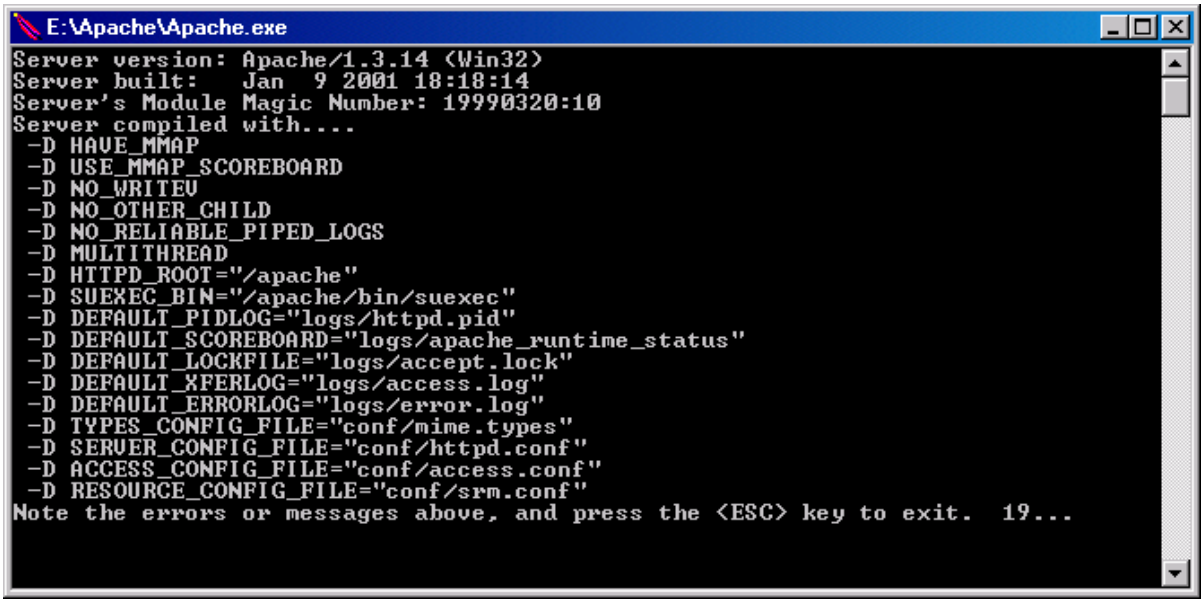


Figura 23. Apache.exe –V

`-L`: Devuelve una lista de directivas de configuración, junto con los argumentos que se les han pasado.

`-l`: Devuelve una lista de todos los módulos compilados en esta versión del servidor

`-h`: Devuelve una lista de opciones posible para ser utilizadas con el ejecutable del servidor. Es similar a la opción -?, veamoslo en la Figura 24.

`-S`: Muestra los valores de cómo ha sido analizado el archivo de configuración (solo muestra valores de errores que pudieran surgir)

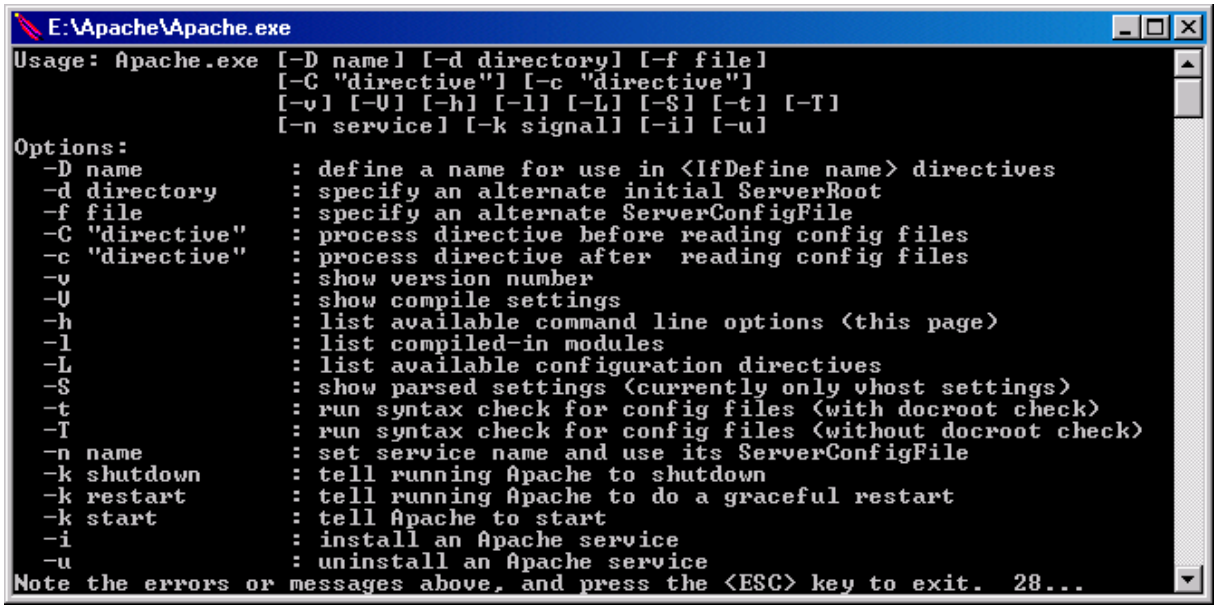


Figura 24. Apache.exe –h

-t: Comprueba la sintaxis del archivo de configuración. Si hubiera errores muestra una lista con los mismos. Si no los hay termina con un estado de salida cero. Verifica si todas las entradas de DocumentRoot y de los directorios son válidas.

-T: Comprueba la sintaxis del archivo de configuración. Si hubiera errores muestra una lista con los mismos. Si no los hay termina con un estado de salida cero. Esta opción no verifica ni DocumentRoot ni los directorios.

-k *option*: Se usa para indicar la servidor su se cierra o se abre mediante shutdown o restart, respectivamente. Solamente se utiliza con Windows.

-?: Devuelve una lista de opciones posible para ser utilizadas con el ejecutable del servidor. Es similar a la opción -h

Start Apache

Para iniciar la puesta en marcha del servidor utilizaremos la siguiente instrucción:

```
%APACHE_HOME%\Apache.exe -d %APACHE_HOME% -s -k start
```

automáticamente nos aparece la ventana de la Figura 17.

Stop Apache

La parada del servidor Apache se realizará inmediatamente después de ejecutar la siguiente instrucción:

```
%APACHE_HOME%\Apache.exe -d %APACHE_HOME% -s -k shutdown
```

cerrándose de esta manera el proceso o demonio de Apache.

Restart Apache

Par a reiniciar el servidor se utiliza la siguiente instrucción:

```
%APACHE_HOME%\Apache.exe -d %APACHE_HOME% -s -k restart
```

Instalación de apache como un servicio de Windows NT (Windows 2000)

Una de las posibilidades que nos encontramos cuando trabajamos con Windows es el poder instalar Apache como si de un servicio del sistema operativo se tratase, teniendo la posibilidad de controlar su puesta en marcha o parada a través del administrador de servicios, pudiendo conseguir con ello que el servidor se ponga en servicio a la vez que lo hace la máquina sobre la que está instalado el sistema operativo.

Una vez instalado y configurado Apache, simplemente tenemos que ejecutar la siguiente instrucción:

```
%APACHE_HOME%\Apache.exe -d %APACHE_HOME% -I
```

Es entonces cuando lo podemos administrar desde los servicios de Windows, tal y como muestran la Figura 25y la Figura 26

Si quisiéramos desinstalar el servicio lo único que bebemos hacer es ejecutar la siguiente línea:

```
E:\Apache\Apache.exe -d "E:\Apache" -u
```

Si queremos comprobar que el servicio de Apache está corriendo, simplemente lo podemos comprobar con el sistema operativo, en el administrador de tareas.

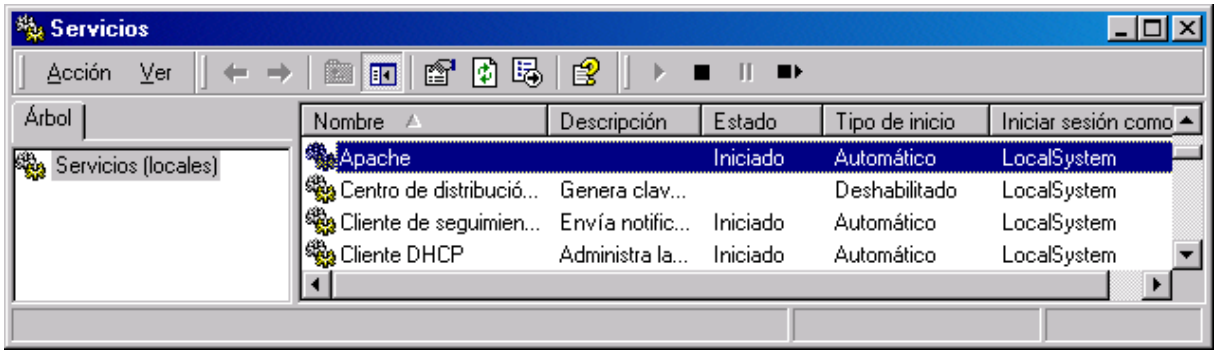


Figura 25. Apache como un servicio más de Windows

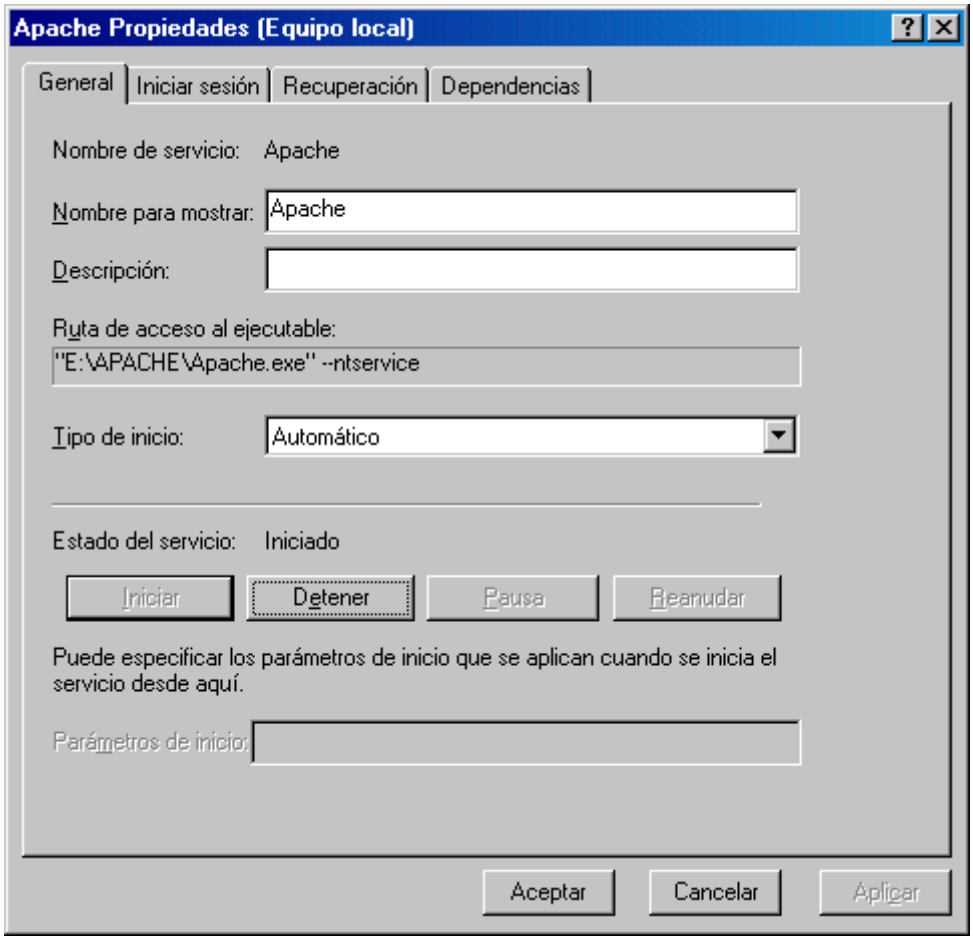


Figura 26. Configuración del servicio de Apache

Administración del servidor

Una vez que el lector haya instalado y configurado a su gusto el servidor, observará que con pocos parámetros que manipule es más que suficiente para que nuestro servidor Apache funcione.

Pero no siempre va a ser así de sencillo. Siempre que utilizamos un servidor web nos hacemos las siguientes preguntas: ¿Podría tener instalado un sitio virtual?.¿Sería posible controlar el acceso a ciertos directorios mediante claves de acceso?. ¿Es posible visualizar la actividad del servidor?.

Todas estas preguntas tienen una única respuesta. En Apache sí es posible realizar todos estos trabajos de administración.

Sitios virtuales

El término sitio virtual se refiere a la técnica que consiste en mantener más de un servidor sobre una misma máquina. Por ejemplo, es preferible para las empresas poseer un servidor web con varios dominios accesibles desde Internet, siempre administrados desde una misma máquina.

Apache es uno de los primeros servidores que soportan sitios virtuales basados en direcciones IP. Existen dos tipos de sitios virtuales, unos basados en la dirección IP.

Sitios virtuales basados sitios virtuales basados en una direcciones ip

Como su propio nombre indica, los sitios virtuales basados en una dirección IP, deben disponer de una dirección IP diferente para cada sitio virtual. Esta dirección IP puede ser obtenida si la máquina dispone de varias conexiones físicas de red, o utilizando interfaces virtuales.

Para configurar un sitio virtual, lo primero que debemos tener es una dirección IP para cada sitio virtual que vamos a mostrar. Partamos de un servidor cuyo ServerName es Minerva, cuya dirección IP es 172.26.0.2. Añadamos posteriormente otra dirección IP que será la compuesta por 172.26.0.3

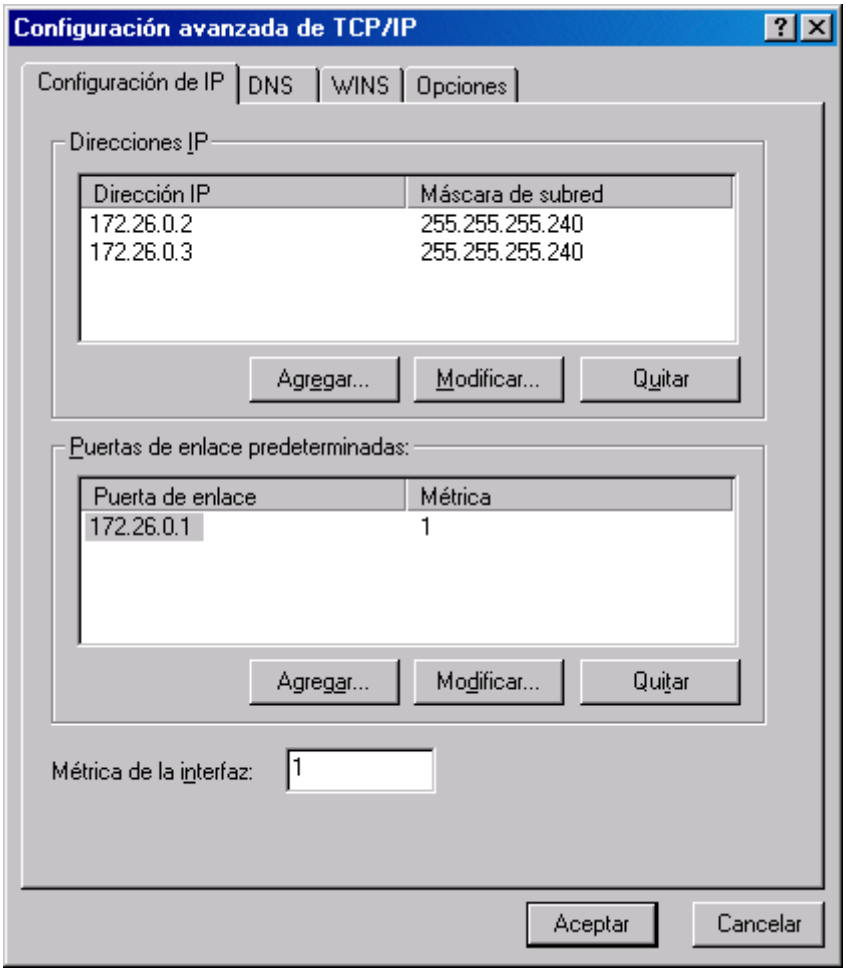


Figura 27. Configuración del servicio de Apache

Lo primero que debemos hacer en nuestro fichero de configuración de Apache es crear una directriz de sitio virtual para indicar al servidor de su existencia.

```
# VirtualHost example:
# Almost any Apache directive may go into a VirtualHost container.
# The first VirtualHost section is used for requests without a known
# server name.
#
#<VirtualHost *>
#    ServerAdmin webmaster@dummy-host.example.com
```

```
#    DocumentRoot /www/docs/dummy-host.example.com
#    ServerName dummy-host.example.com
#    ErrorLog logs/dummy-host.example.com-error_log
#    CustomLog logs/dummy-host.example.com-access_log common
#</VirtualHost>

<VirtualHost 172.26.0.3>
    #ServerAdmin minerva
    DocumentRoot "E:/Apache/htdocs"
    ServerName atenea
    #ErrorLog logs/dummy-host.example.com-error_log
    #CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

Código fuente 45. Sitio virtual

Una vez que hemos configurado esta directriz, que como verá el lector dispone de otras directrices que dan características al sitio virtual para que funcionen de manera independiente, nos disponemos a su ejecución. Desde el mismo servidor web, estamos llamando a nuestro sitio minerva, y de igual manera, llamaremos al sitio 172.26.0.3

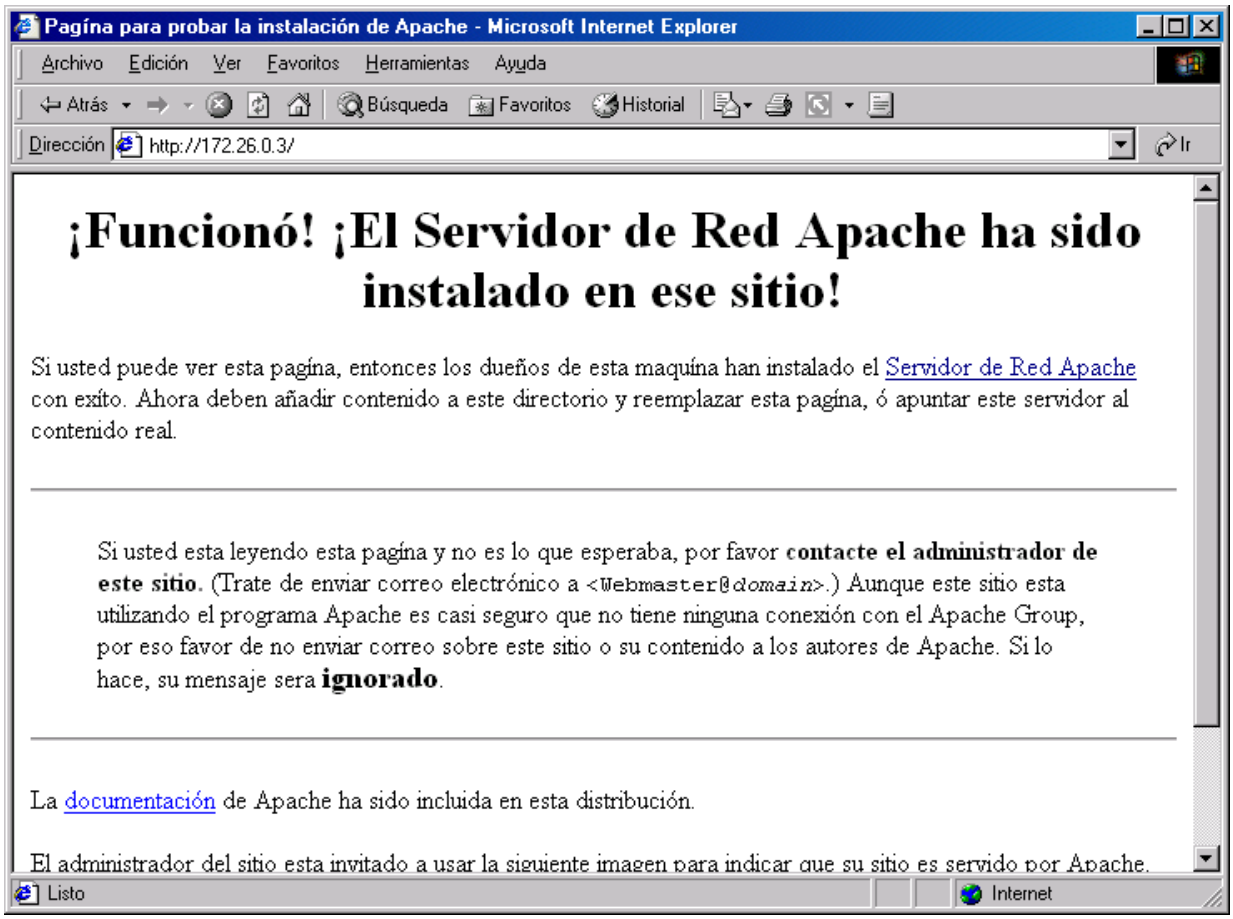


Figura 28. Ejecución sitio virtual 172.26.0.3

Si queremos darle un nombre a esta dirección IP, debemos configurar una DNS en la que indicaremos un nombre para esa dirección, por ejemplo atenea.

```
# Éste es un ejemplo de archivo HOSTS usado por Microsoft TCP/IP para Windows.
#
# Este archivo contiene las asignaciones de las direcciones IP a los nombres de
# host. Cada entrada debe permanecer en una línea individual. La dirección IP
# debe ponerse en la primera columna, seguida del nombre de host correspondiente.
# La dirección IP y el nombre de host deben separarse con al menos un espacio.
#
#
# También pueden insertarse comentarios (como éste) en líneas individuales
# o a continuación del nombre de equipo indicándolos con el símbolo "#"
#
# Por ejemplo:
#
#      102.54.94.97     rhino.acme.com          # servidor origen
#       38.25.63.10     x.acme.com              # host cliente x

127.0.0.1       localhost
172.26.0.3      atenea
```

Código fuente 46. Fichero host

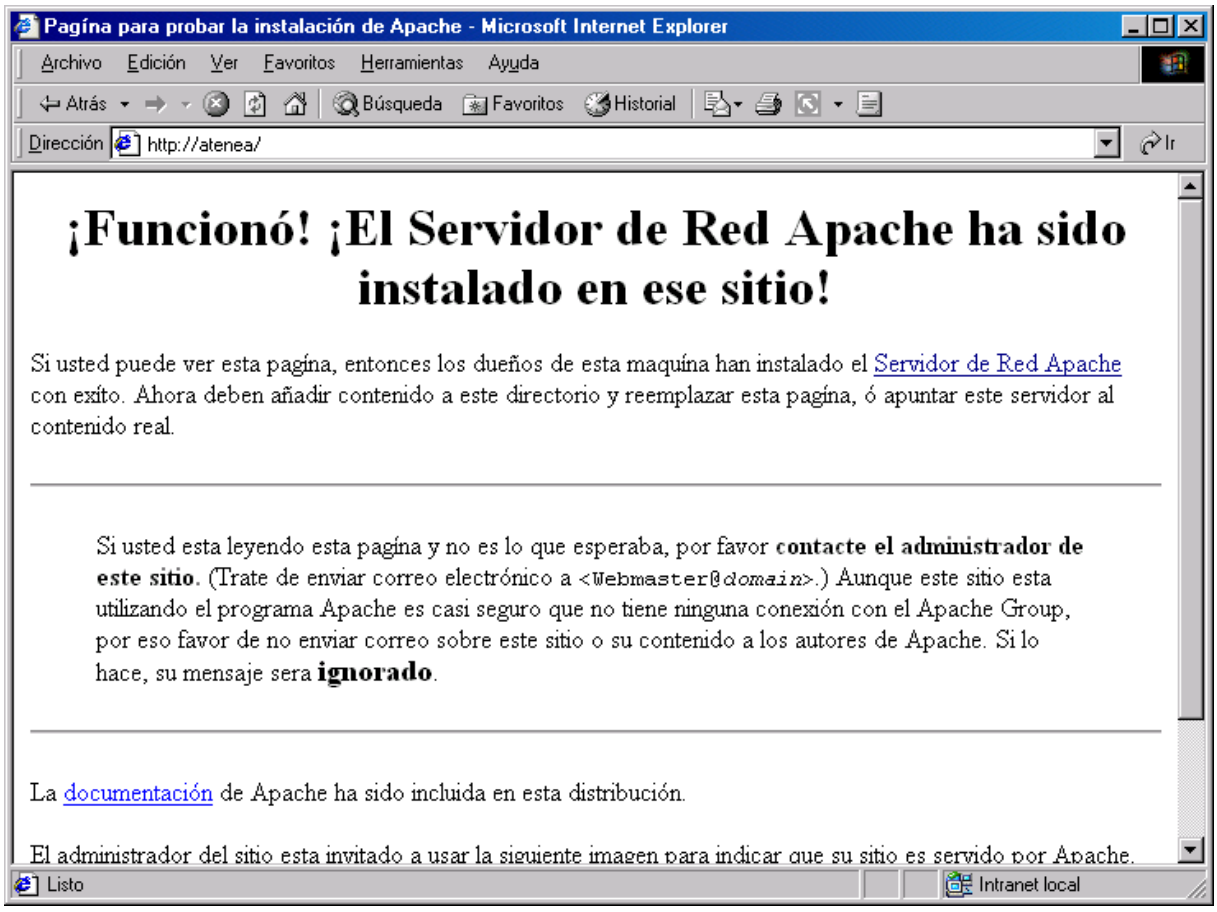


Figura 29. Ejecución sitio virtual atenea (172.26.0.3)

Sitios virtuales basados en un nombre ip

Aunque los sitios virtuales basados en una dirección IP funcionan muy bien, no es la solución más elegante, pues a veces es difícil obtener varias direcciones IP para una misma máquina. El protocolo

HTTP/1.1 define un método de identificación del servidor que puede determinar que nombre de dirección ha sido indicado. Por lo tanto, podemos utilizar la misma dirección IP para varios nombres de servidores. Lo único que debemos es disponer de varios nombres asociados a una dirección IP.

```
# Éste es un ejemplo de archivo HOSTS usado por Microsoft TCP/IP para Windows.
#
# Este archivo contiene las asignaciones de las direcciones IP a los nombres de
# host. Cada entrada debe permanecer en una línea individual. La dirección IP
# debe ponerse en la primera columna, seguida del nombre de host correspondiente.
# La dirección IP y el nombre de host deben separarse con al menos un espacio.
#
#
# También pueden insertarse comentarios (como éste) en líneas individuales
# o a continuación del nombre de equipo indicándolos con el símbolo "#"
#
# Por ejemplo:
#
#      102.54.94.97     rhino.acme.com          # servidor origen
#       38.25.63.10     x.acme.com              # host cliente x

127.0.0.1       localhost
172.26.0.3      java
172.26.0.3      Confidencial
```

Código fuente 47. Fichero host con varios nombres y la misma dirección IP

Vamos a configurar dos sitios virtuales que van a tener dos nombres, pero la misma dirección IP.

```
NameVirtualHost 172.26.0.3
<VirtualHost 172.26.0.3>
    #ServerAdmin minerva
    DocumentRoot "E:/TutorialJava"
    ServerName java
    #ErrorLog logs/dummy-host.example.com-error_log
    #CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost><VirtualHost 172.26.0.3>
    #ServerAdmin minerva
    DocumentRoot "E:\Virtual2
    DirectoryIndex index.html index.htm
    ServerName confidencial
    #ErrorLog logs/dummy-host.example.com-error_log
    #CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

Código fuente 48. Sitios virtuales con varios nombres y la misma dirección IP

Si ejecutamos directamente la dirección IP 172.26.0.3, nos aparecerá la primera que tengamos configurada en nuestro archivo de configuración. Si escribimos en la barra de direcciones de nuestro navegador localhost, tenemos nuestra página de siempre (Figura 30)

Si utilizamos el nombre de la segunda, atenea, nos aparece un conocido tutorial que se encuentra en el directorio que hemos indicado en la directriz DocumentRoot de nuestro sitio virtual (Figura 31)

Cuando escribamos el tercer nombre, que es confidencial, aparece una nueva página de otro sitio que tenemos configurado (Figura 32)

Figura 30. Ejecución de localhost

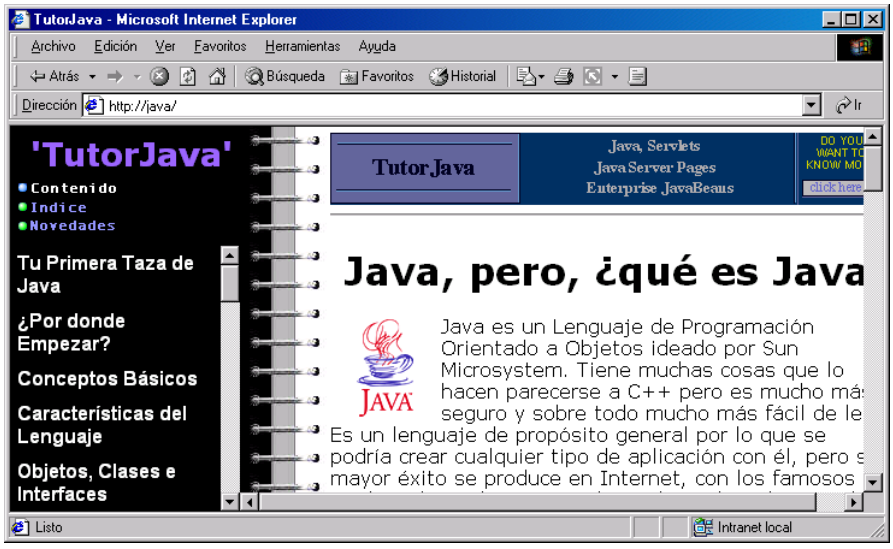


Figura 31. Ejecución del sitio virtual java

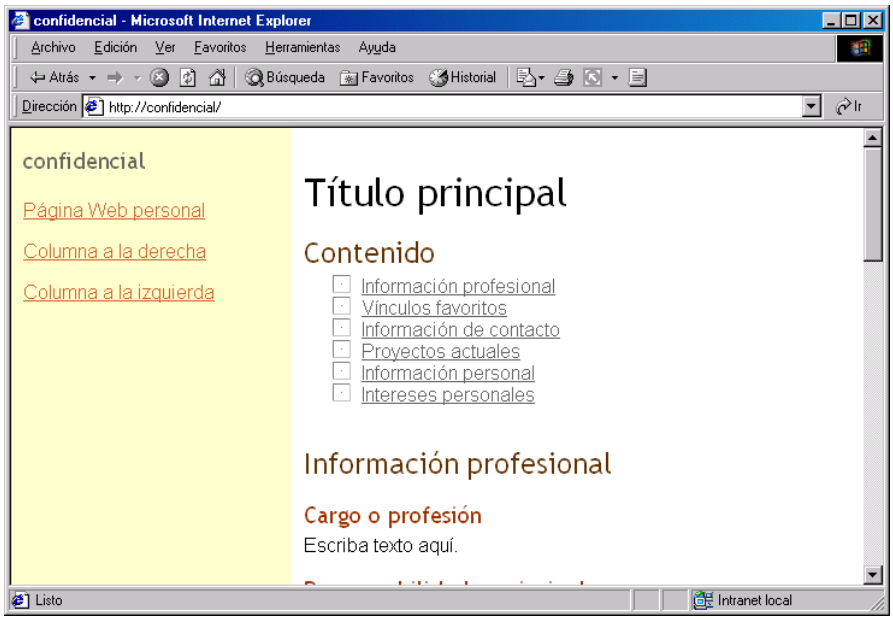


Figura 32. Ejecución del sitio virtual confidencial(172.26.0.3)

El lector debe observar como las llamadas las direcciones java y confidencial, pertenecen a la misma dirección IP, pero siendo dos sitios virtuales distintos.

Autenticación básica

Cuando un cliente se registra en una red, la sesión se mantiene en el servidor hasta que el cliente abandona la red. El protocolo HTTP no es permanente, por lo que las sesiones no se mantienen de modo indefinido entre el explorador web y un servidor web como Apache.

Para realizar una autenticación con Apache nosotros vamos a realizarla mediante una autenticación de HTTP básico. Cuando un explorador Web solicita un URL que está protegido por una autenticación HTTP, el servidor Web le devuelve una cabecera de estado 401, y otra de respuesta para la autenticación. La cabecera contiene el sistema de autenticación que se está utilizando y el nombre del dominio.

El módulo que interviene en éste proceso se compila por defecto, y es mod_auth. Nosotros debemos activar este módulo en el archivo de configuración.

```
LoadModule anon_auth_module modules/ApacheModuleAuthAnon.dll
```

Código fuente 49. mod_auth

AuthUserFile

Esta directriz nos va ha indicar el nombre del fichero que va a tener el nombre de los usuarios y sus contraseñas. Este archivo lo debemos crear con una utilidad llamada htpasswd.

AuthGroupFile

Esta directriz especifica el archivo de texto que se utiliza como lista de los grupos de usuarios.

AuthAuthoritative

Se utiliza cuando existe más de un sistema de autenticación para un mismo directorio. Si falla uno, se consigue que el primer par nombre/contraseña pase al otro. Para configurar que cuando se solicite el acceso a un directorio, este nos pida el nombre de usuario y su contraseña, lo que debemos hacer es configurar dicho directorio como restringido.

```
<Directory "E:\Virtual2">
    Options Indexes FollowSymLinks MultiViews
    AllowOverride All
    #AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

Código fuente 50. Directorio restringido

Una vez que dicho directorio ya esta restringido, deberemos crearnos un archivo de usuario, en el que se guardarán el nombre y contraseña.

Para crear este archivo ejecutamos desde la línea de comandos del sistema operativo el siguiente ejecutable:

```
htpasswd.exe  -c <nombre y ruta del fichero><usuario>
```

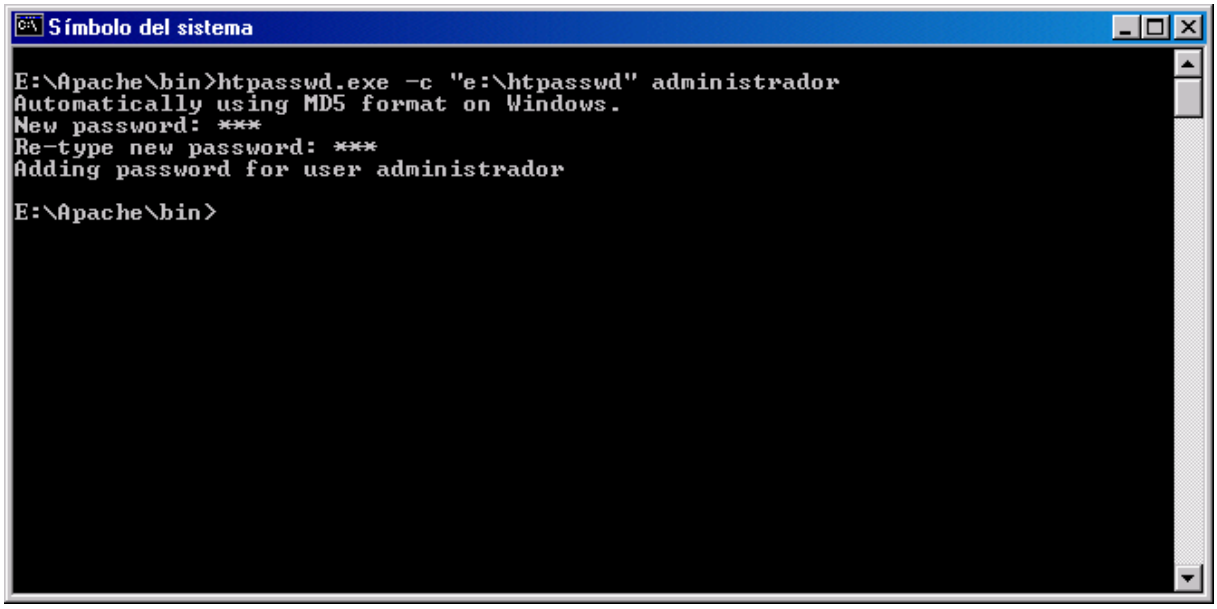


Figura 33 Ejecución htpasswd

Hemos llamado al archivo como htpasswd. Este fichero puede editar normalmente como fichero de texto que es. Si el lector comprueba la Figura 34, observará que la contraseña está encriptada.

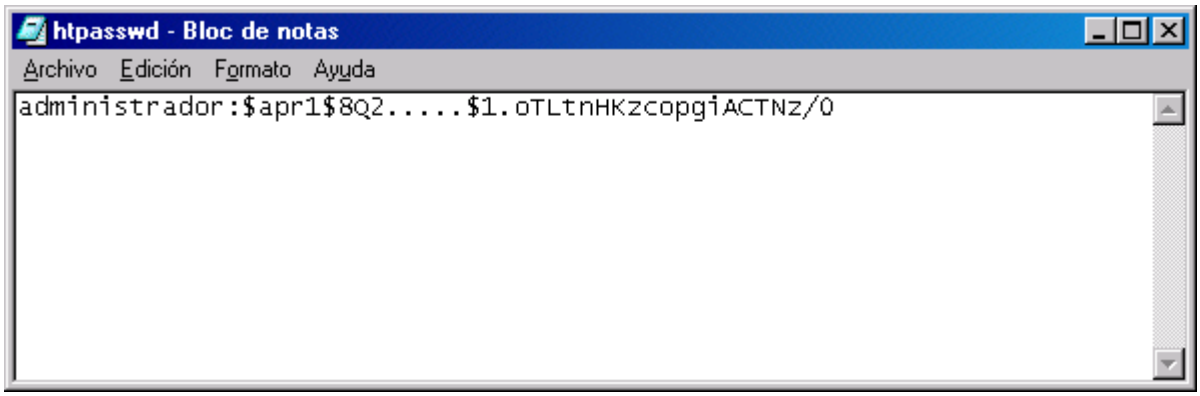


Figura 34. Fichero de usuario

Cuando configuramos Apache, mediante la directriz AccessFileName, estamos indicando cual es el fichero de control de acceso que debe existir en un directorio al que queremos restringir su acceso. Nosotros hemos lo hemos llamado htaccess.

Debemos utilizar un editor de texto para introducir las líneas que aparecen en el Código fuente 51.

```
AuthName "Acceso Privado"
AuthType Basic
AuthUserFile "e:/htpasswd"
require group grupo
```

Código fuente 51. Fichero de acceso

En la directiva AuthName indicamos un nombre de esquema de autorización a un directorio. Esta directiva que en realidad es una etiqueta, debe ir acompañada de las directiva AuthType y require, indicando también mediante la directiva AuthUserFile y AuthGroupFile, donde se encuentran los ficheros de contraseñas. La directiva AuthType se utiliza para seleccionar el tipo de identificación para un directorio. Los valores que puede utilizar son Basic y Digest. La directiva require indica que usuario o grupo están autorizados a acceder a un directorio.

Una vez que los archivos htaccess y htpasswd están creados, es muy importante que a estos ficheros solamente pueda acceder Apache. Para ello deberemos utilizar la directiva File, tal y como muestra el Código fuente 52.

```
<Files ~ "^\htpasswd">
    Order allow,deny
    Deny from all
</Files>
<Files ~ "^\htaccess">
    Order allow,deny
    Deny from all
</Files>
```

Código fuente 52. Restriccion de acceso a los ficheros htpasswd y htaccess

De esta manera nadie podrá acceder al contenido de estos ficheros. El fichero htaccess debe encontrarse en el directorio que queremos proteger.

Una vez que lo tenemos todo configurado, intentamos acceder al directorio protegido desde nuestro navegador. Si funciona correctamente, deberá aparecer una caja de diálogo, como la de la Figura 35, solicitándonos un nombre de usuario y una contraseña.

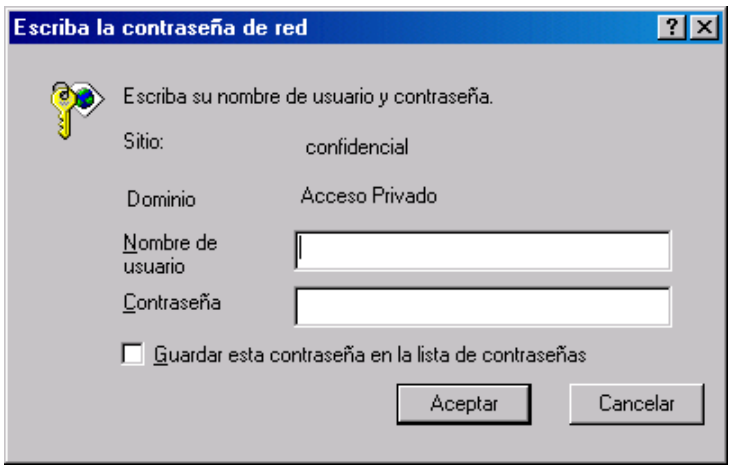


Figura 35 Control de acceso

Si el navegador Apache permite durante tres intentos el poder teclear correctamente el nombre de usuario y la contraseña. Si esto no fuera así, aparecería una página indicándonos que no podemos acceder al directorio solicitado.

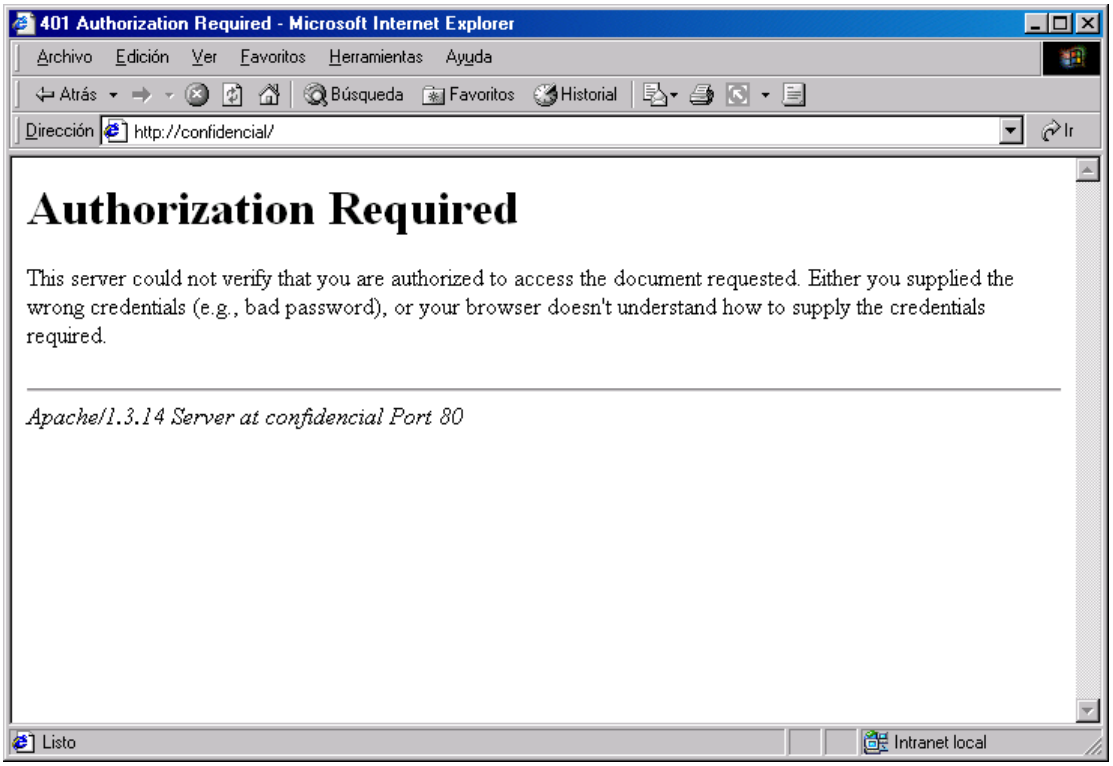


Figura 36. Permiso denegado.

Con el usuario y la contraseña correcta, podemos entrar en nuestro directorio.

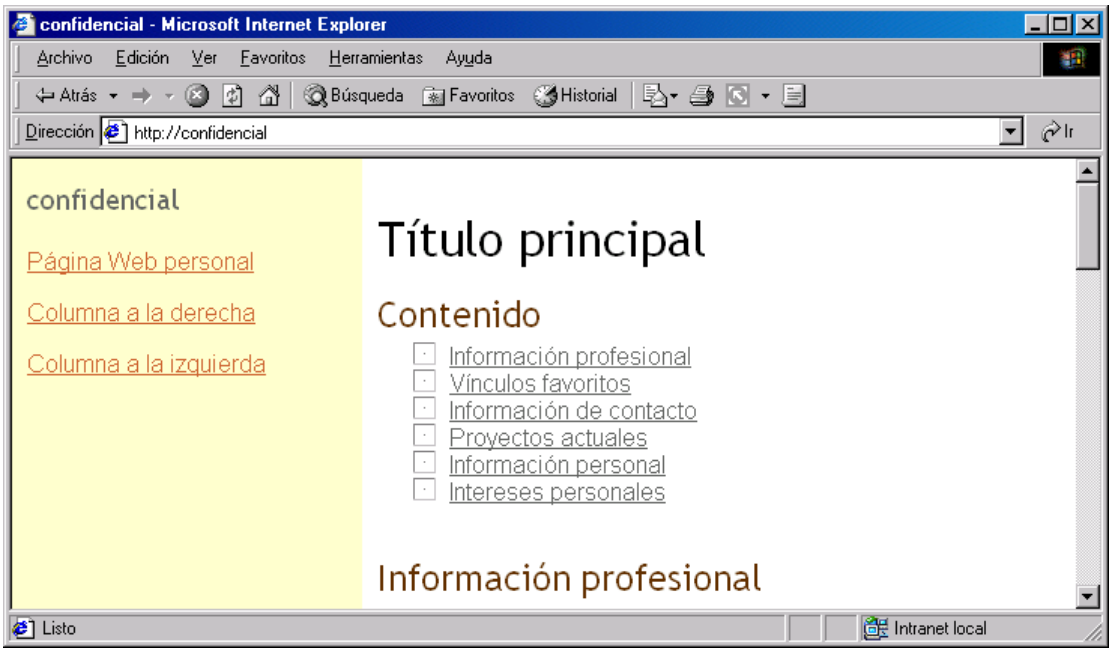


Figura 37. Permiso concedido.

Para permitir el acceso a un grupo, bastará con crear un nuevo archivo llamado htgroup, y posteriormente, mediante un editor de texto, escribir el nombre de un grupo, indicando el nombre de sus componentes, los cuales ya estarán dados de alta en el fichero de contraseñas.

```
grupo: fjegea administrador
```

Código fuente 53. Fichero htgroup.

```
fjegea:$apr1$SW3.....$kX/fhBEKJgSlsdU2.k6Yl.
administrador:$apr1$4X3.....$x1cpx1jc1pWCUTBQyaAYX1
```

Código fuente 54. Fichero htpasswd con dos usuarios

Lo único que tenemos que hacer es añadir en el fichero de acceso la directriz AuthGroupFile con la ruta y el nombre del fichero creado para el grupo, y modificar la directriz require.

```
AuthName "Acceso Privado"
AuthType Basic
AuthUserFile "e:/htpasswd"
AuthGroupFile "e:/htgroup"
require group grupo
```

Código fuente 55. Fichero htaccess con acceso a grupo

Visualizar información de configuración

Apache permite visualizar dos tipos de información

- Información sobre la configuración del servidor.
- Estado del servidor.

La información sobre la configuración es estática. Para poder disponer de esta información deberemos tener disponible el módulo mod_info, para lo cual deberemos tenerlo activo en el fichero de configuración.

```
LoadModule info_module modules/ApacheModuleInfo.dll
```

Código fuente 56. mod_info

Una vez que hemos activado este módulo, tenemos que añadir también al fichero las líneas del Figura 37.

```
<Location /server-info>
```

```
    SetHandler server-info
    Order deny,allow
    Allow from 172.26.0.2/255.255.255.240

</Location>
```

Código fuente 57. Directorio web server-info

Con todo configurado, ejecutamos desde nuestro navegador:

<nombre dominio>/server-info

Aparecerá entonces toda la información reflejada en el Código fuente 58

```
Apache Server Information
Server Settings, mod_jserv.c, mod_status.c, mod_info.c, mod_auth_anon.c,
mod_isapi.c, mod_setenvif.c, mod_actions.c, mod_imap.c, mod_asis.c,
mod_log_config.c, mod_env.c, mod_alias.c, mod_userdir.c, mod_cgi.c, mod_dir.c,
mod_autoindex.c, mod_include.c, mod_negotiation.c, mod_auth.c, mod_access.c,
mod_mime.c, mod_so.c, http_core.c
------------------------------------------------------------------------------
Server Version: Apache/1.3.14 (Win32) tomcat/1.0
Server Built: Jan 9 2001 18:18:14
API Version: 19990320:10
Run Mode: standalone
User/Group: #-1(1)/1
Hostname/port: minerva:80
Daemons: start: 5    min idle: 5    max idle: 10    max: 1024
Max Requests: per child: 0    keep alive: on    max per connection: 100
Threads: per child: 50
Excess requests: per child: 0
Timeouts: connection: 300    keep-alive: 15
Server Root: e:/apache
Config File: e:/apache/conf/httpd.conf
PID File: logs/httpd.pid
Scoreboard File: logs/apache_runtime_status

------------------------------------------------------------------------------

Module Name: mod_jserv.c
Content handlers: jserv-servlet , jserv-status , jserv-action
Configuration Phase Participation: Child Init, Create Server Config, Merge Server
Configs
Request Phase Participation: Translate Path, Check Type
Module Directives:
ApJServManual - Whether Apache JServ is running in manual or automatic mode.
ApJServProperties - The full pathname of jserv.properties file.
ApJServDefaultProtocol - The default protocol used for connecting to Apache JServ.
ApJServDefaultHost - The default host running Apache JServ.
ApJServDefaultPort - The default port on which Apache JServ is running on.
ApJServMount - Where Apache JServ servlets will be mounted under Apache.
ApJServMountCopy - Whether inherits base host mount points or not.
ApJServLogFile - Apache JServ log file relative to Apache root directory.
ApJServLogLevel - Apache JServ log verbosity.
ApJServSecretKey - Apache JServ secret key file relative to Apache root directory.
ApJServProtocolParameter - Apache JServ protocol-dependant property.
ApJServAction - Apache JServ action mapping extension to servlets.
ApJServBalance - Apache JServ load-balancing server set.
ApJServHost - Apache JServ host definition.
ApJServRoute - Apache JServ host routing identifier.
ApJServShmFile - The full pathname of shared memory file.
```

```
ApJServRetryAttempts - Apache JServ: retry attempts (1s appart) before returning
server error
ApJServVMTimeout - Apache JServ: the amount of time given for the JVM to start or
stop
ApJServVMInterval - Apache JServ: the interval between 2 polls of the JVM
ApJServEnvVar - Apache JServ: protocol ajpv12 : env var to send to the server
Current Configuration:

--------------------------------------------------------------------------------

Module Name: mod_status.c
Content handlers: application/x-httpd-status , server-status
Configuration Phase Participation: none
Request Phase Participation: none
Module Directives:
ExtendedStatus - "On" to enable extended status information, "Off" to disable
Current Configuration:
conf/httpd.conf
ExtendedStatus On

--------------------------------------------------------------------------------

Module Name: mod_info.c
Content handlers: server-info
Configuration Phase Participation: Create Server Config, Merge Server Configs
Request Phase Participation: none
Module Directives:
AddModuleInfo - a module name and additional information on that module
Current Configuration:

--------------------------------------------------------------------------------

Module Name: mod_auth_anon.c
Content handlers: none
Configuration Phase Participation: Create Directory Config
Request Phase Participation: Verify User ID, Verify User Access
Module Directives:
Anonymous - a space-separated list of user IDs
Anonymous_MustGiveEmail - Limited to 'on' or 'off'
Anonymous_NoUserId - Limited to 'on' or 'off'
Anonymous_VerifyEmail - Limited to 'on' or 'off'
Anonymous_LogEmail - Limited to 'on' or 'off'
Anonymous_Authoritative - Limited to 'on' or 'off'
Current Configuration:

--------------------------------------------------------------------------------

Module Name: mod_isapi.c
Content handlers: isapi-isa
Configuration Phase Participation: none
Request Phase Participation: none
Module Directives:
ISAPIReadAheadBuffer - Maximum bytes to initially pass to the ISAPI handler
ISAPILogNotSupported - Log requests not supported by the ISAPI server
ISAPIAppendLogToErrors - Send all Append Log requests to the error log
ISAPIAppendLogToQuery - Append Log requests are concatinated to the query args
Current Configuration:

--------------------------------------------------------------------------------

Module Name: mod_setenvif.c
Content handlers: none
Configuration Phase Participation: Create Directory Config, Merge Directory
Configs, Create Server Config, Merge Server Configs
Request Phase Participation: Post-Read Request, Header Parse
Module Directives:
SetEnvIf - A header-name, regex and a list of variables.
```

```
SetEnvIfNoCase - a header-name, regex and a list of variables.
BrowserMatch - A browser regex and a list of variables.
BrowserMatchNoCase - A browser regex and a list of variables.
Current Configuration:
conf/httpd.conf
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0

--------------------------------------------------------------------------------

Module Name: mod_actions.c
Content handlers: */*
Configuration Phase Participation: Create Directory Config, Merge Directory Configs
Request Phase Participation: none
Module Directives:
Action - a media type followed by a script name
Script - a method followed by a script name
Current Configuration:

--------------------------------------------------------------------------------

Module Name: mod_imap.c
Content handlers: application/x-httpd-imap , imap-file
Configuration Phase Participation: Create Directory Config, Merge Directory Configs
Request Phase Participation: none
Module Directives:
ImapMenu - the type of menu generated: none, formatted, semiformatted, unformatted
ImapDefault - the action taken if no match: error, nocontent, referer, menu, URL
ImapBase - the base for all URL's: map, referer, URL (or start of)
Current Configuration:

--------------------------------------------------------------------------------

Module Name: mod_asis.c
Content handlers: httpd/send-as-is , send-as-is
Configuration Phase Participation: none
Request Phase Participation: none
Module Directives: none

--------------------------------------------------------------------------------

Module Name: mod_log_config.c
Content handlers: none
Configuration Phase Participation: Create Server Config, Merge Server Configs
Request Phase Participation: Logging
Module Directives:
CustomLog - a file name, a custom log format string or format name, and an optional
"env=" clause (see docs)
TransferLog - the filename of the access log
LogFormat - a log format string (see docs) and an optional format name
CookieLog - the filename of the cookie log
Current Configuration:
conf/httpd.conf
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
CustomLog logs/access.log common

--------------------------------------------------------------------------------

Module Name: mod_env.c
Content handlers: none
Configuration Phase Participation: Create Directory Config, Merge Directory Configs
```

```
Request Phase Participation: Fixups
Module Directives:
PassEnv - a list of environment variables to pass to CGI.
SetEnv - an environment variable name and a value to pass to CGI.
UnsetEnv - a list of variables to remove from the CGI environment.
Current Configuration:

-------------------------------------------------------------------------------

Module Name: mod_alias.c
Content handlers: none
Configuration Phase Participation: Create Directory Config, Merge Directory
Configs, Create Server Config, Merge Server Configs
Request Phase Participation: Translate Path, Fixups
Module Directives:
Alias - a fakename and a realname
ScriptAlias - a fakename and a realname
Redirect - an optional status, then document to be redirected and destination URL
AliasMatch - a regular expression and a filename
ScriptAliasMatch - a regular expression and a filename
RedirectMatch - an optional status, then a regular expression and destination URL
RedirectTemp - a document to be redirected, then the destination URL
RedirectPermanent - a document to be redirected, then the destination URL
Current Configuration:
conf/httpd.conf
Alias /icons/ "E:/Apache/icons/"
ScriptAlias /cgi-bin/ "E:/Apache/cgi-bin/"

-------------------------------------------------------------------------------

Module Name: mod_userdir.c
Content handlers: none
Configuration Phase Participation: Create Server Config
Request Phase Participation: Translate Path
Module Directives:
UserDir - the public subdirectory in users' home directories, or 'disabled', or
'disabled username username...', or 'enabled username username...'
Current Configuration:
conf/httpd.conf
UserDir "E:/Apache/users/"

-------------------------------------------------------------------------------

Module Name: mod_cgi.c
Content handlers: application/x-httpd-cgi , cgi-script
Configuration Phase Participation: Create Server Config, Merge Server Configs
Request Phase Participation: none
Module Directives:
ScriptLog - the name of a log for script debugging info
ScriptLogLength - the maximum length (in bytes) of the script debug log
ScriptLogBuffer - the maximum size (in bytes) to record of a POST request
Current Configuration:

-------------------------------------------------------------------------------

Module Name: mod_dir.c
Content handlers: httpd/unix-directory
Configuration Phase Participation: Create Directory Config, Merge Directory Configs
Request Phase Participation: none
Module Directives:
DirectoryIndex - a list of file names
Current Configuration:
conf/httpd.conf
DirectoryIndex index.html
DirectoryIndex index.html index.htm

-------------------------------------------------------------------------------
```

```
Module Name: mod_autoindex.c
Content handlers: httpd/unix-directory
Configuration Phase Participation: Create Directory Config, Merge Directory Configs
Request Phase Participation: none
Module Directives:
AddIcon - an icon URL followed by one or more filenames
AddIconByType - an icon URL followed by one or more MIME types
AddIconByEncoding - an icon URL followed by one or more content encodings
AddAlt - alternate descriptive text followed by one or more filenames
AddAltByType - alternate descriptive text followed by one or more MIME types
AddAltByEncoding - alternate descriptive text followed by one or more content
encodings
IndexOptions - one or more index options
IndexOrderDefault - {Ascending,Descending} {Name,Size,Description,Date}
IndexIgnore - one or more file extensions
AddDescription - Descriptive text followed by one or more filenames
HeaderName - a filename
ReadmeName - a filename
FancyIndexing - Limited to 'on' or 'off' (superseded by IndexOptions FancyIndexing)
DefaultIcon - an icon URL
Current Configuration:

---------------------------------------------------------------------------

Module Name: mod_include.c
Content handlers: text/x-server-parsed-html , text/x-server-parsed-html3 , server-
parsed , text/html
Configuration Phase Participation: Create Directory Config
Request Phase Participation: none
Module Directives:
XBitHack - Off, On, or Full
Current Configuration:

---------------------------------------------------------------------------

Module Name: mod_negotiation.c
Content handlers: application/x-type-map , type-map
Configuration Phase Participation: Create Directory Config, Merge Directory Configs
Request Phase Participation: Check Type, Fixups
Module Directives:
CacheNegotiatedDocs - no arguments (either present or absent)
LanguagePriority - space-delimited list of MIME language abbreviations
Current Configuration:
conf/httpd.conf
LanguagePriority en da nl et fr de el it ja kr no pl pt pt-br ru ltz ca es sv tw

---------------------------------------------------------------------------

Module Name: mod_auth.c
Content handlers: none
Configuration Phase Participation: Create Directory Config
Request Phase Participation: Verify User ID, Verify User Access
Module Directives:
AuthUserFile - text file containing user IDs and passwords
AuthGroupFile - text file containing group names and member user IDs
AuthAuthoritative - Set to 'off' to allow access control to be passed along to
lower modules if the UserID is not known to this module
Current Configuration:

---------------------------------------------------------------------------

Module Name: mod_access.c
Content handlers: none
Configuration Phase Participation: Create Directory Config
Request Phase Participation: Check Access
Module Directives:
```

```
order - 'allow,deny', 'deny,allow', or 'mutual-failure'
allow - 'from' followed by hostnames or IP-address wildcards
deny - 'from' followed by hostnames or IP-address wildcards
Current Configuration:
conf/httpd.conf
<Directory "E:/Apache/htdocs">
  Order allow,deny
  Allow from all
</Directory>
<Directory "E:\Virtual2">
  Order allow,deny
  Allow from all
</Directory>
<Files ~ "^\.ht">
  Order allow,deny
  Deny from all
</Files>
<Files ~ "^\htpasswd">
  Order allow,deny
  Deny from all
</Files>
<Files ~ "^\htaccess">
  Order allow,deny
  Deny from all
</Files>
<Directory "E:/Apache/icons">
  Order allow,deny
  Allow from all
</Directory>
<Directory "E:/Apache/cgi-bin">
  Order allow,deny
  Allow from all
</Directory>
<Location /server-status>
  Order deny,allow
  deny from all
  Allow from 172.26.0.2/255.255.255.240
</Location>
<Location /server-info>
  Order deny,allow
  Allow from 172.26.0.2/255.255.255.240
</Location>

--------------------------------------------------------------------------------

Module Name: mod_mime.c
Content handlers: none
Configuration Phase Participation: Create Directory Config, Merge Directory Configs
Request Phase Participation: Check Type
Module Directives:
AddType - a mime type followed by one or more file extensions
AddEncoding - an encoding (e.g., gzip), followed by one or more file extensions
AddCharset - a charset (e.g., iso-2022-jp), followed by one or more file extensions
AddLanguage - a language (e.g., fr), followed by one or more file extensions
AddHandler - a handler name followed by one or more file extensions
ForceType - a media type
RemoveHandler - one or more file extensions
RemoveEncoding - one or more file extensions
RemoveType - one or more file extensions
SetHandler - a handler name
TypesConfig - the MIME types config file
DefaultLanguage - language to use for documents with no other language file
extension
Current Configuration:
conf/httpd.conf
TypesConfig conf/mime.types
AddEncoding x-compress Z
```

```
AddEncoding x-gzip gz tgz
AddLanguage da .dk
AddLanguage nl .nl
AddLanguage en .en
AddLanguage et .ee
AddLanguage fr .fr
AddLanguage de .de
AddLanguage el .el
AddLanguage he .he
AddCharset ISO-8859-8 .iso8859-8
AddLanguage it .it
AddLanguage ja .ja
AddCharset ISO-2022-JP .jis
AddLanguage kr .kr
AddCharset ISO-2022-KR .iso-kr
AddLanguage no .no
AddLanguage pl .po
AddCharset ISO-8859-2 .iso-pl
AddLanguage pt .pt
AddLanguage pt-br .pt-br
AddLanguage ltz .lu
AddLanguage ca .ca
AddLanguage es .es
AddLanguage sv .se
AddLanguage cz .cz
AddLanguage ru .ru
AddLanguage tw .tw
AddCharset Big5 .Big5 .big5
AddCharset WINDOWS-1251 .cp-1251
AddCharset CP866 .cp866
AddCharset ISO-8859-5 .iso-ru
AddCharset KOI8-R .koi8-r
AddCharset UCS-2 .ucs2
AddCharset UCS-4 .ucs4
AddCharset UTF-8 .utf8
AddType application/x-tar .tgz
<Location /server-status>
  SetHandler server-status
</Location>
<Location /server-info>
  SetHandler server-info
</Location>

-------------------------------------------------------------------------------

Module Name: mod_so.c
Content handlers: none
Configuration Phase Participation: Create Server Config
Request Phase Participation: none
Module Directives:
LoadModule - a module name and the name of a shared object file to load it from
LoadFile - shared object file or library to load into the server at runtime
Current Configuration:
conf/httpd.conf
LoadModule anon_auth_module modules/ApacheModuleAuthAnon.dll
LoadModule info_module modules/ApacheModuleInfo.dll
LoadModule status_module modules/ApacheModuleStatus.dll

-------------------------------------------------------------------------------

Module Name: http_core.c
Content handlers: */* , default-handler
Configuration Phase Participation: Create Directory Config, Merge Directory
Configs, Create Server Config, Merge Server Configs
Request Phase Participation: Translate Path, Check Access, Check Type
Module Directives:
```

```
<Directory - Container for directives affecting resources located in the specified
directories
</Directory> - Marks end of
<Location - Container for directives affecting resources accessed through the
specified URL paths
</Location> - Marks end of
<VirtualHost - Container to map directives to a particular virtual host, takes one
or more host addresses
</VirtualHost> - Marks end of
<Files - Container for directives affecting files matching specified patterns
</Files> - Marks end of
<Limit - Container for authentication directives when accessed using specified HTTP
methods
</Limit> - Marks end of
<LimitExcept - Container for authentication directives to be applied when any HTTP
method other than those specified is used to access the resource
</LimitExcept> - Marks end of
<IfModule - Container for directives based on existance of specified modules
</IfModule> - Marks end of
<IfDefine - Container for directives based on existance of command line defines
</IfDefine> - Marks end of
<DirectoryMatch - Container for directives affecting resources located in the
specified directories
</DirectoryMatch> - Marks end of
<LocationMatch - Container for directives affecting resources accessed through the
specified URL paths
</LocationMatch> - Marks end of
<FilesMatch - Container for directives affecting files matching specified patterns
</FilesMatch> - Marks end of
AuthType - An HTTP authorization type (e.g., "Basic")
AuthName - The authentication realm (e.g. "Members Only")
Require - Selects which authenticated users or groups may access a protected space
Satisfy - access policy if both allow and require used ('all' or 'any')
AddDefaultCharset - The name of the default charset to add to any Content-Type
without one or 'Off' to disable
AccessFileName - Name(s) of per-directory config files (default: .htaccess)
DocumentRoot - Root directory of the document tree
ErrorDocument - Change responses for HTTP errors
AllowOverride - Controls what groups of directives can be configured by per-
directory config files
Options - Set a number of attributes for a given directory
DefaultType - the default MIME type for untypable files
ServerType - 'inetd' or 'standalone'
Port - A TCP port number
HostnameLookups - "on" to enable, "off" to disable reverse DNS lookups, or "double"
to enable double-reverse DNS lookups
User - Effective user id for this server
Group - Effective group id for this server
ServerAdmin - The email address of the server administrator
ServerName - The hostname of the server
ServerSignature - En-/disable server signature (on|off|email)
ServerRoot - Common directory of server-related files (logs, confs, etc.)
ErrorLog - The filename of the error log
PidFile - A file for logging the server process ID
ScoreBoardFile - A file for Apache to maintain runtime process management
information
LockFile - The lockfile used when Apache needs to lock the accept() call
AccessConfig - The filename of the access config file
ResourceConfig - The filename of the resource config file
ServerAlias - A name or names alternately used to access the server
ServerPath - The pathname the server can be reached at
Timeout - Timeout duration (sec)
KeepAliveTimeout - Keep-Alive timeout duration (sec)
MaxKeepAliveRequests - Maximum number of Keep-Alive requests per connection, or 0
for infinite
KeepAlive - Whether persistent connections should be On or Off
IdentityCheck - Enable identd (RFC 1413) user lookups - SLOW
```

```
ContentDigest - whether or not to send a Content-MD5 header with each request
UseCanonicalName - How to work out the ServerName : Port when constructing URLs
StartServers - Number of child processes launched at server startup
MinSpareServers - Minimum number of idle children, to handle request spikes
MaxSpareServers - Maximum number of idle children
MaxServers - Deprecated equivalent to MaxSpareServers
ServersSafetyLimit - Deprecated equivalent to MaxClients
MaxClients - Maximum number of children alive at the same time
MaxRequestsPerChild - Maximum number of requests a particular child serves before
dying.
RLimitCPU - Soft/hard limits for max CPU usage in seconds
RLimitMEM - Soft/hard limits for max memory usage per process
RLimitNPROC - soft/hard limits for max number of processes per uid
BindAddress - '*', a numeric IP address, or the name of a host with a unique IP
address
Listen - A port number or a numeric IP address and a port number
SendBufferSize - Send buffer size in bytes
AddModule - The name of a module
ClearModuleList -
ThreadsPerChild - Number of threads a child creates
ExcessRequestsPerChild - Maximum number of requests a particular child serves after
it is ready to die.
ListenBacklog - Maximum length of the queue of pending connections, as used by
listen(2)
CoreDumpDirectory - The location of the directory Apache changes to before dumping
core
Include - Name of the config file to be included
LogLevel - Level of verbosity in error logging
NameVirtualHost - A numeric IP address:port, or the name of a host
ScriptInterpreterSource - Where to find interpreter to run Win32 scripts (Registry
or script shebang line)
ServerTokens - Determine tokens displayed in the Server: header - Min(imal), OS or
Full
LimitRequestLine - Limit on maximum size of an HTTP request line
LimitRequestFieldsize - Limit on maximum size of an HTTP request header field
LimitRequestFields - Limit (0 = unlimited) on max number of header fields in a
request message
LimitRequestBody - Limit (in bytes) on maximum size of request message body
Current Configuration:
conf/httpd.conf
ServerType standalone
ServerRoot "E:/Apache"
PidFile logs/httpd.pid
ScoreBoardFile logs/apache_runtime_status
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
MaxRequestsPerChild 0
ThreadsPerChild 50
Port 80
ServerAdmin fjegea@eidos.es
ServerName minerva
DocumentRoot "E:/Apache/htdocs"
<Directory />
  Options FollowSymLinks
  AllowOverride None
</Directory>
<Directory "E:/Apache/htdocs">
  Options Indexes FollowSymLinks MultiViews
  AllowOverride None
</Directory>
<Directory "E:\Virtual2">
  Options Indexes FollowSymLinks MultiViews
  AllowOverride All
</Directory>
<IfModule mod_userdir.c>
```

```
</IfModule>
<IfModule mod_dir.c>
</IfModule>
AccessFileName htaccess
UseCanonicalName On
<IfModule mod_mime.c>
</IfModule>
DefaultType text/plain
<IfModule mod_mime_magic.c>
</IfModule>
HostnameLookups Off
ErrorLog logs/error.log
LogLevel warn
ServerSignature On
<IfModule mod_alias.c>
<Directory "E:/Apache/icons">
  Options Indexes MultiViews
  AllowOverride None
</Directory>
<Directory "E:/Apache/cgi-bin">
  AllowOverride None
  Options None
</Directory>
</IfModule>
<IfModule mod_mime.c>
<IfModule mod_negotiation.c>
</IfModule>
</IfModule>
<IfModule mod_setenvif.c>
</IfModule>
NameVirtualHost 172.26.0.3
<VirtualHost 172.26.0.3>
DocumentRoot "E:/Apache/htdocs"
ServerName atenea
</VirtualHost>
<VirtualHost 172.26.0.3>
DocumentRoot "E:\Virtual2
ServerName confidencial
</VirtualHost>
include e:/jakarta-tomcat/conf/tomcat-apache.conf

-------------------------------------------------------------------------------

Apache/1.3.14 Server at minerva Port 80
```

Código fuente 58. Salida de server-info

Con server-info obtenemos toda la información del servidor, así como la de sus módulos.

Páginas de estado

Con las páginas de estado vamos a poder visualizar desde la red el estado del servidor. Para ello debemos activar el módulo mod_status.

```
LoadModule status_module modules/ApacheModuleStatus.dll
```

Código fuente 59. mod_status

La información que podemos visualizar es la siguiente:

- Hora del sistema del servidor.
- Hora del último servicio del servidor.
- Tiempo transcurrido desde que se encendió el servidor.
- Número total de accesos desde la puesta en marcha.
- Bytes totales transmitidos.
- Número de hilos de ejecución dedicados a las peticiones.
- Número total de hilos de ejecución.
- Estado de cada ejecución.
- Promedio de peticiones por segundo.
- Porcentaje de CPV utilizado por cada proceso, y el total utilizado por Apache.

Para configurar server-status deberemos añadir el texto al Código fuente 60 del fichero de configuración.

```
<Location /server-status>
    SetHandler server-status
    Order deny,allow
       deny from all
    Allow from 172.26.0.2/255.255.255.240

</Location>
```

Código fuente 60. Directorio server-status

También deberemos activar la directriz ExtendedStatus

```
# ExtendedStatus controls whether Apache will generate "full" status
# information (ExtendedStatus On) or just basic information (ExtendedStatus
# Off) when the "server-status" handler is called. The default is Off.
#
ExtendedStatus On
```

Código fuente 61. Directiva ExtendedStatus

Una vez configurado todo lo anterior simplemente escribiremos en nuestro navegador:

`<nombre dominio>/server-status`

Aparecerá el texto del Código fuente 62.

```
Apache Server Status for minerva
Server Version: Apache/1.3.14 (Win32) tomcat/1.0
Server Built: Jan 9 2001 18:18:14

-------------------------------------------------------------------------------
Current Time: Thursday, 08-Feb-2001 23:09:04 Hora estándar romance
Restart Time: Thursday, 08-Feb-2001 21:55:40 Hora estándar romance
Parent Server Generation: 1
Server uptime: 1 hour 13 minutes 24 seconds
Total accesses: 50 - Total Traffic: 52 kB
.0114 requests/sec - 12 B/second - 1064 B/request
1 requests currently being processed, 49 idle servers
___________W_____________________________________.............
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................
.....................................................................

Scoreboard Key:
"_" Waiting for Connection, "S" Starting up, "R" Reading Request,
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,
"L" Logging, "G" Gracefully finishing, "." Open slot with no current process


Srv PID Acc M  SS Req Conn Child Slot Client VHost Request
0-0 0 0/8/8 _  4067 0 0.0 0.00 0.00  172.26.0.2 (unavailable) GET
/examples/images/code.gif HTTP/1.1
1-0 0 0/1/1 _  4067 0 0.0 0.00 0.00  172.26.0.2 (unavailable) GET
/examples/images/return.gif HTTP/1.1
2-0 0 0/2/2 _  1947 0 0.0 0.00 0.00  172.26.0.2 (unavailable) GET /apache_pb.gif
HTTP/1.1
3-0 0 0/1/1 _  1935 0 0.0 0.000 0.000  172.26.0.3 (unavailable) GET / HTTP/1.1
4-0 0 0/3/3 _  1807 0 0.0 0.00 0.00  172.26.0.3 (unavailable) GET / HTTP/1.1
5-0 0 0/5/5 _  1701 0 0.0 0.00 0.00  172.26.0.3 (unavailable) GET / HTTP/1.1
6-0 0 0/3/3 _  1674 0 0.0 0.00 0.00  172.26.0.3 (unavailable) GET / HTTP/1.1
7-0 0 0/2/2 _  1655 0 0.0 0.00 0.00  172.26.0.2 (unavailable) GET /apache_pb.gif
HTTP/1.1
8-0 0 0/14/14 _  1634 0 0.0 0.00 0.00  172.26.0.3 (unavailable) GET
/P%C3%A1gina%20Web%20personal_archivos/image001.gif HTTP/1.
9-0 0 0/8/8 _  1638 0 0.0 0.00 0.00  172.26.0.3 (unavailable) GET
/P%C3%A1gina%20Web%20personal_archivos/image001.gif HTTP/1.
10-0 0 0/2/2 _  1608 0 0.0 0.000 0.000  172.26.0.3 (unavailable) GET
/P%C3%A1gina%20Web%20personal_archivos/image001.gif HTTP/1.
11-0 0 0/1/1 _  886 0 0.0 0.04 0.04  172.26.0.2 (unavailable) GET /server-info
HTTP/1.1
12-0 0 0/0/0 W  0 0 0.0 0.00 0.00  172.26.0.2 (unavailable) GET /server-status
HTTP/1.1



-------------------------------------------------------------------------------
Srv Child Server number - generation
PID OS process ID
Acc Number of accesses this connection / this child / this slot
M Mode of operation
```

```
SS Seconds since beginning of most recent request
Req Milliseconds required to process most recent request
Conn Kilobytes transferred this connection
Child Megabytes transferred this child
Slot Total megabytes transferred this slot

--------------------------------------------------------------------------------

Apache/1.3.14 Server at minerva Port 80
```

Código fuente 62. Salida de server-status

El lector comprenderá que la salida en su máqina para server-status como para server-info será distinta a la que aparezca en estos ejemplos de código.

5

Jakarta Tomcat

Introducción

En este capítulo vamos a comentar la instalación y configuración del servidor Web Jakarta Tomcat. Se puede utilizar Jakarta Tomcat 3.2 como servidor Web independiente y como contenedor de servlets y de páginas JSP, es decir Jakarta Tomcat nos ha ofrecido la implementación de la especificación Java servlet 2.2 y de la especificación JavaServer Pages 1.1.

Existen otros servidores Web y motores que implementan las especificaciones servlet 2.2 y JSP 1.1, aunque el más extendido, que es reconocido como la implementación oficial, es el servidor Web Jakarta Tomcat 3.1.

Instalación de Jakarta Tomcat

El servidor Web Jakarta Tomcat se puede utilizar como un servidor Web de prueba para ejecutar servlets y de páginas JSP (JavaServer Pages), es decir, además de implementar la especificación Java servlet 2.2 implementa la especificación JavaServer Pages 1.1. Jakarta Tomcat se usa como un añadido para el servidor Apache, es decir, permite que el servidor Web Apache pueda ejecutar servlets y páginas JSP. Esta última utilización es la que se le debería dar a Jakarta Tomcat en un entorno de producción real.

Jakarta Tomcat está preparado para poder ejecutarse sin problemas en una máquina en la que ya se encuentre instalado un servidor Web, ya que por defecto el servidor Web Jakarta Tomcat utiliza el puerto 8080 para el protocolo HTTP, en lugar del estándar, es decir, el puerto 80.

Jakarta Tomcat es un servidor Web gratuito ofrecido por Apache, y que podemos obtener en la siguiente dirección http://jakarta.apache.org.

Jakarta Tomcat se ofrece en forma binaria, es decir, compilado y listo para utilizar, y también se ofrece el código fuente para que sea el desarrollador quien lo compile. Por sencillez vamos a utilizar la primera opción. Lo que obtenemos del sitio Web de Jakarta Tomcat es un fichero comprimido en formato ZIP (jakarta-tomcat.zip), que debemos descomprimir en nuestra máquina.

Al descomprimir el fichero, se nos generará una estructura de directorios que contiene los distintos elementos del servidor Web Jakarta Tomcat, en el apartado correspondiente detallaremos esta estructura de directorios y que funcionalidad tiene cada uno de ellos. El directorio en el que se encuentra toda la estructura de Jakarta Tomcat se llama jakarta-tomcat.

La configuración del servidor Web Jakarta Tomcat no resulta nada intuitiva ni fácil de realizar, al igual que ocurre con el servidor Web Apache. En este apartado vamos a comentar únicamente la configuración mínima necesaria que se necesaria para poder ejecutar Tomcat.

Dos ficheros que se adjuntan junto Jakarta Tomcat, y a los que hay que prestar un atención especial son: STARTUP.BAT y SHUTDOWN.BAT. Al ejecutar el primero de ellos iniciaremos la ejecución del servidor Web Tomcat, y al ejecutar el segundo se finalizará la ejecución del servidor Jakarta Tomcat. Estos dos ficheros se encuentran el directorio jakarta-tomcat\bin.

Para poder ejecutar el servidor Web Jakarta Tomcat debemos modificar el fichero de inicio del mismo, es decir, el fichero STARTUP.BAT. Esta modificación consiste en añadir una línea que especifique un parámetro que indicará la localización de la implementación de la plataforma Java 2, es decir, la localización del Java 2 SDK (Software Development Kit), que se supone ya tenemos instalado.

En el fichero STARTUP.BAT se debe especificar la variable JAVA_HOME después de los comentarios que parecen en el fichero (varias líneas comenzando por rem). Tanto si hemos instalado la plataforma J2EE o la plataforma J2SE, deberemos indicar el directorio de instalación del Java 2 SDK Estándar Edition, que por defecto es c:\jdk1.3\, de todas formas el lector puede comprobar fácilmente el directorio raíz en el que ha instalado el Java 2 SDKSE. Por lo tanto la línea que debemos añadir en STARTUP.BAT es:

```
SET JAVA_HOME=c:\jdk1.3\
```

Un ejemplo de fichero STARTUP.BAT completo se ofrece en el Código fuente 63.

```
@echo off
rem $Id: startup.bat,v 1.7 2000/03/31 19:40:02 craigmcc Exp $
rem Startup batch file for tomcat servner.

rem This batch file written and tested under Windows NT
rem Improvements to this file are welcome

SET JAVA_HOME=C:\jdk1.3\

if not "%TOMCAT_HOME%" == "" goto start

SET TOMCAT_HOME=.
if exist %TOMCAT_HOME%\bin\tomcat.bat goto start

SET TOMCAT_HOME=..
if exist %TOMCAT_HOME%\bin\tomcat.bat goto start

SET TOMCAT_HOME=
echo Unable to determine the value of TOMCAT_HOME.
```

```
goto eof

:start
call %TOMCAT_HOME%\bin\tomcat start %1 %2 %3 %4 %5 %6 %7 %8 %9

:eof
```

Código fuente 63

Una vez realizado este cambio ya estamos preparados para iniciar la ejecución de Jakarta Tomcat. Para iniciar la ejecución de Jakarta Tomcat simplemente debemos realizar doble clic sobre el fichero STARTUP.BAT, y aparecerá una ventana muy similar a la de la Figura 38.

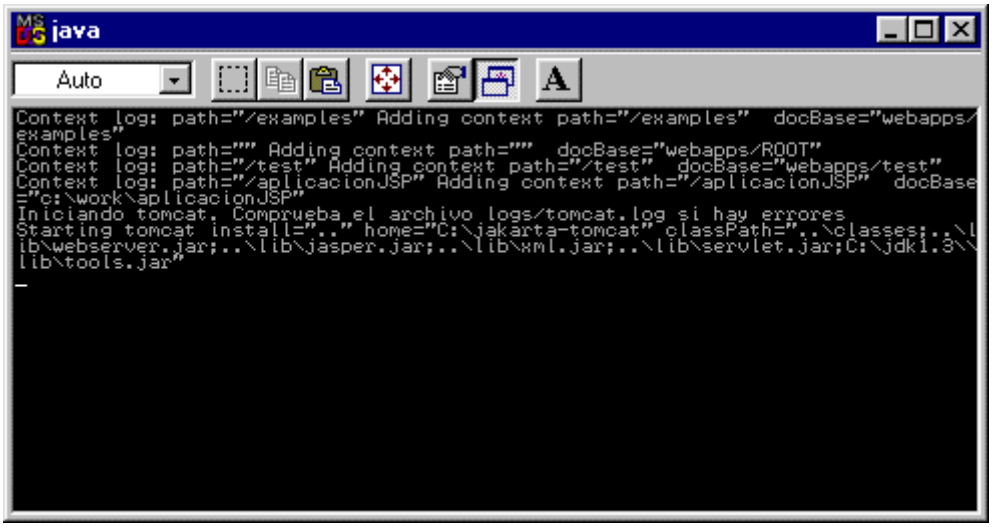


Figura 38. Inicio de Jakarta Tomcat

Para verificar que está todo correcto y que Jakarta Tomcat puede ejecutar servlets y páginas JSP sin ningún problema vamos a realizar una sencilla comprobación.

Debemos ejecutar un navegador Web, ya sea Internet Explorer o Netscape Communicator, y debemos especificar la siguiente URL http://localhost:8080/examples.

Esta URL se corresponde con los ejemplos de servlets y páginas JSP, que se ofrecen en la instalación de Jakarta Tomcat, el aspecto de esta página se puede comprobar en la Figura 39.

Para comprobar el correcto funcionamiento de un servlet en el servidor Tomcat pulsaremos sobre el enlace servlets.

Este enlace nos lleva a la página de ejemplos de servlets, en la que al pulsar cada uno Así por ejemplo si pulsamos sobre el primer ejemplo, llamado Hello World, ejecutaremos el servlet HelloWorld Example que nos mostrará la página de la Figura 40.

Si deseamos detener Tomcat no tenemos nada más que ejecutar el fichero SHUTDOWN.BAT. Al ejecutar este fichero se detendrá el servidor Web Jakarta Tomcat y se cerrará la ventana del intérprete de comandos que se había abierto al iniciarse su ejecución.

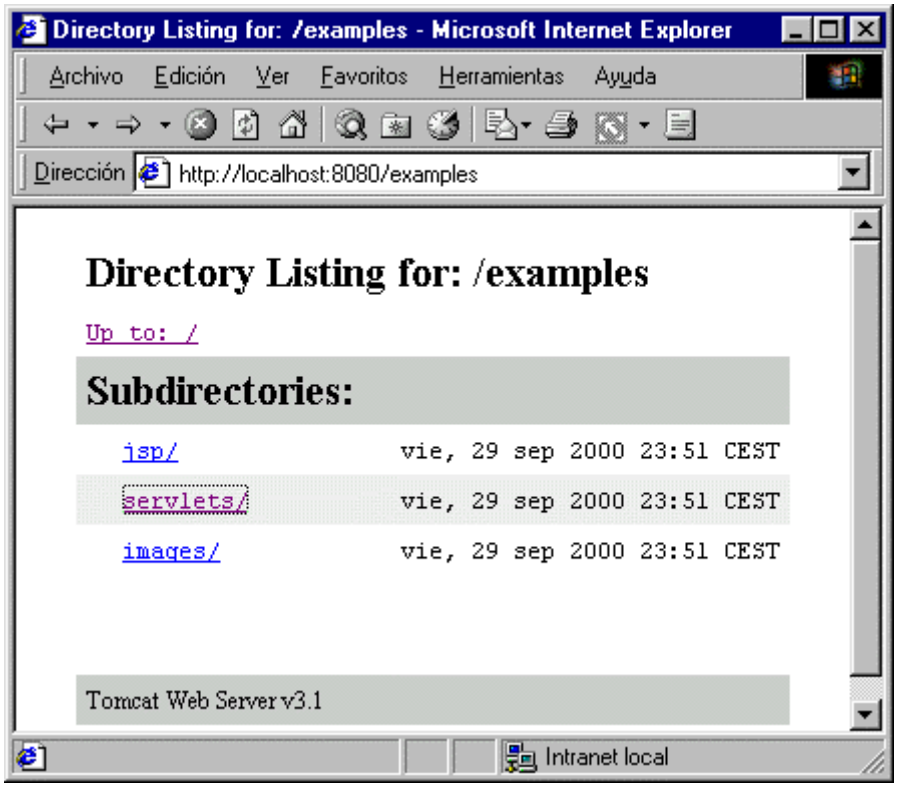


Figura 39. Ejemplos disponibles en Tomcat

Figura 40. Ejecución de un servlet de ejemplo

Estructura de directorios de Jakarta Tomcat

Al instalar Jakarta Tomcat, se crea un directorio llamado jakarta-tomcat como directorio raíz de la aplicación, dentro de este directorio encontramos diversos subdirectorios que son los que pasamos a comentar a continuación:

- BIN: contiene los scripts de arranque y parada del servidor Web, es decir, los ficheros por lotes STARTUP.BAT y SHUTDOWN.BAT respectivamente.

- CONF: contiene varios ficheros de configuración del servidor, los principales los SERVER.XML y WEB.XML. El primero de estos ficheros es el fichero de configuración principal del servidor, y el segundo de ellos establece los valores por defecto de las distintas aplicaciones Web del servidor. La estructura de estos dos directorios la comentaremos más adelante.

- DOC: contiene la documentación de Jakarta Tomcat.

- LIB: contiene varios ficheros JAR de Tomcat. Se ofrece un fichero JAR (Java Archive), llamado SERVLET.JAR que contiene los paquetes de la especificación Java servlet 2.2 y de la especificación JavaServer Pages 1.1, y también los ficheros JASPER.JAR y WEBSERVER.JAR, que contienen las clases que forman parte del propio contenedor de servlets y páginas JSP.

- LOGS: en este directorio se encuentran los ficheros de registro de Tomcat.

- SRC: contiene los ficheros fuente de la especificación Java servlet y de la especificación JavaServer Pages.

- WEBAPPS: contiene las aplicaciones Web de ejemplo que se distribuyen con Tomcat.

- WORK: es le directorio de trabajo de Tomcat, es dónde tiene lugar la traducción de las páginas JSP a los servlets correspondientes. Existirá un subdirectorio del directorio WORK por cada aplicación Web, así si nuestra aplicación Web se llama ejemplos, su directorio de trabajo será c:\jakarta-tomcat\work\localhost_8080%2Fejemplos. El directorio de trabajo de la aplicación encontraremos un fichero .JAVA y otro fichero .CLASS que contienen en su nombre la cadena paginaJSP, es decir, el nombre de la página JSP a la que corresponden y a partir de la que se han generado. En este caso han generado los ficheros _0002fpaginaJSP_0002ejsppaginaJSP_jsp_0.java y _0002fpaginaJSP_0002ejsppaginaJSP.class. Si abrimos el fichero fuente del servlet, podemos ver el código Java que ha generado el contenedor de página JSP para crear el servlet equivalente a la página.

Ficheros de configuración

Como ya hemos comentado dentro del directorio CONF se encuentran los dos ficheros de configuración de Tomcat, que son SERVER.XML y WEB.XML.

Fichero SERVER.XML

El fichero SERVER.XML es el fichero general de configuración del servidor Web, y una de las funciones principales es la de definir las aplicaciones Web del servidor.

Este fichero de configuración está definido en formato XML (Extensive Markup Language), no vamos a entrar a comentar el lenguaje XML, sino que simplemente vamos a comentar y describir las etiquetas que nos permiten definir una nueva aplicación en el servidor Tomcat.

Antes de seguir comentando como definir una aplicación en Tomcat en el fichero SERVER.XML, vamos a mostrar en el Código Fuente 64 el contenido del fichero SERVER.XML que viene por defecto junto con la instalación del servidor Jakarta Tomcat.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<Server>
    <!-- Debug low-level events in XmlMapper startup -->
    <xmlmapper:debug level="0" />

    <!-- This is quite flexible; we can either have a log file per
         module in Tomcat (example: ContextManager) or we can have
         one for Servlets and one for Jasper, or we can just have
        one tomcat.log for both Servlet and Jasper.

        If you omit "path" there, then stderr should be used.

        verbosityLevel values can be:
           FATAL
           ERROR
           WARNING
             INFORMATION
             DEBUG
         -->

    <Logger name="tc_log"
            path="logs/tomcat.log"
            customOutput="yes" />

    <Logger name="servlet_log"
            path="logs/servlet.log"
            customOutput="yes" />

    <Logger name="JASPER_LOG"
           path="logs/jasper.log"
            verbosityLevel = "INFORMATION" />

    <!-- Add "home" attribute if you want tomcat to be based on a different
directory
         "home" is used to create work and to read webapps, but not for libs or
CLASSPATH.
         Note that TOMCAT_HOME is where tomcat is installed, while ContextManager
home is the
         base directory for contexts, webapps/ and work/
      -->
    <ContextManager debug="0" workDir="work" >
        <!-- ContextInterceptor className="org.apache.tomcat.context.LogEvents" / -
->
        <ContextInterceptor className="org.apache.tomcat.context.AutoSetup" />
        <ContextInterceptor className="org.apache.tomcat.context.DefaultCMSetter"
/>
        <ContextInterceptor
className="org.apache.tomcat.context.WorkDirInterceptor" />
        <ContextInterceptor className="org.apache.tomcat.context.WebXmlReader" />
        <ContextInterceptor
className="org.apache.tomcat.context.LoadOnStartupInterceptor" />
        <!-- Request processing -->
        <RequestInterceptor className="org.apache.tomcat.request.SimpleMapper"
debug="0" />
        <RequestInterceptor
className="org.apache.tomcat.request.SessionInterceptor" />
        <RequestInterceptor className="org.apache.tomcat.request.SecurityCheck" />
        <RequestInterceptor className="org.apache.tomcat.request.FixHeaders" />

        <Connector className="org.apache.tomcat.service.SimpleTcpConnector">
            <Parameter name="handler"
value="org.apache.tomcat.service.http.HttpConnectionHandler"/>
            <Parameter name="port" value="8080"/>
        </Connector>
```

```
        <Connector className="org.apache.tomcat.service.SimpleTcpConnector">
            <Parameter name="handler"
value="org.apache.tomcat.service.connector.Ajp12ConnectionHandler"/>
            <Parameter name="port" value="8007"/>
        </Connector>

        <!-- example - how to override AutoSetup actions -->
        <Context path="/examples" docBase="webapps/examples" debug="0"
reloadable="true" >
        </Context>
        <!-- example - how to override AutoSetup actions -->
        <Context path="" docBase="webapps/ROOT" debug="0" reloadable="true" >
        </Context>

        <Context path="/test" docBase="webapps/test" debug="0" reloadable="true" >
        </Context>

    </ContextManager>
</Server>
```

Código Fuente 64. Contenido del fichero SERVER.XML

No debemos asustarnos con tanto código XML, por ahora sólo nos vamos a fijar en las etiquetas <Context> que aparecen al final del fichero SERVER.XML. Para definir una aplicación dentro del fichero SERVER.XML debemos utilizar la etiqueta <Context>, existirán tantas etiquetas <Context> como aplicaciones definidas en el servidor Tomcat. Debido a esto, las aplicaciones Web en el entorno de Tomcat también se pueden denominar contextos.

El aspecto de la etiqueta <Context> lo podemos ver en el Código Fuente 65 para la aplicación que se ofrece de ejemplo (examples) con el servidor Jakarta Tomcat.

```
<Context path="/examples" docBase="webapps/examples" debug="0" reloadable="true" >
```

Código Fuente 65

Vamos a comentar las propiedades principales que posee la etiqueta <Context>:

- path: la propiedad path indica el nombre de directorio que va a recibir la aplicación en su URL correspondiente, así por ejemplo, para acceder a la aplicación examples debemos escribir la siguiente URL en el navegador Web: http://localhost:8080/examples.

- docBase: la propiedad docBase de la etiqueta <Context> indica la localización física de los ficheros que forman parte de la aplicación. En este caso los ficheros que forman la aplicación se identifican con un camino relativo, que es un subdirectorio del directorio de instalación de Tomcat y que es dónde por defecto se encuentran todas alas aplicaciones Web de Tomcat, este directorio se llama c:\jakarta-tomcat\webapps, por lo tanto la ruta física completa del directorio físico de la aplicación examples es c:\jakarta-tomcat\webapps\examples.

- debug: mediante debug indicamos el nivel de detalle, de cero a nueve, de los diferentes mensajes de depuración que se registren, cero es el menor nivel de detalle y nueve es el que ofrece mayor información.

- reloadable: la propiedad reloadable de la etiqueta <Context> puede tomar un valor booleano, que indicará si el servidor Tomcat detecta automáticamente las modificaciones que se realicen en los servlets de la aplicación. Es muy recomendable dar el valor de verdadero (true) a esta

propiedad en servidores de desarrollo, ya que de esta forma podremos realizar modificaciones sobre los servlets sin tener que reiniciar el servidor de nuevo. De todas formas esta operación es costosa y como tal requiere un tiempo extra en la ejecución del servidor.

De esta forma, si queremos crear una aplicación nueva llamada ejemplos, cuyos ficheros se encuentran en la ruta física c:\work\ejemplos, y además queremos utilizar el menor detalle en la depuración y que se recarguen los servlets de forma automática al modificase, añadiremos la siguiente etiqueta <Context> (Código Fuente 66) en el fichero SERVER.XML

```
<Context path="/ejemplos" docBase="c:\work\ejemplos" debug="0" reloadable="true" >
</Context>
```

Código Fuente 66

Como ya hemos visto en el código del fichero SERVER.XML, además de la etiqueta <Context>, existen otra serie de etiquetas XML que nos van a permitir configurar nuestro servidor Web.

- `<Server>`: es la etiqueta de mayor jerarquía y representa al servidor Tomcat, normalmente no debemos realizar ningún tipo de configuración ni tarea con este elemento.

- `<Logger>`: este elemento define un objeto de registro (logger object). Cada objeto de registro va a poseer un nombre y una ruta en la que se situará el fichero de registro en el que se irá almacenando la información correspondiente. Existen tres objetos de registro, una para los servlets, otro para las páginas JSP y otro para el propio Tomcat. En el Código Fuente 67 se pueden ver los objetos de registro que se definen por defecto.

```
<Logger name="tc_log"
         path="logs/tomcat.log"
         customOutput="yes" />

 <Logger name="servlet_log"
         path="logs/servlet.log"
         customOutput="yes" />

 <Logger name="JASPER_LOG"
       path="logs/jasper.log"
         verbosityLevel = "INFORMATION" />
```

Código Fuente 67

- `<ContextManager>`: esta etiqueta define la estructura para un conjunto de otras etiquetas, y permite especificar el directorio de trabajo de Tomcat, así como el nivel empleado en los mensajes de depuración.

- `<ContextInterceptor>` y `<RequestInterceptor>`: estas etiquetas denominadas de interceptores, tiene la función de atender a distintos eventos del ContextManager, como puede ser el inicio y finalización del servidor, cuyo encargado es el ContextInterceptos, y las peticiones que se realizan, que son seguidas por el RequestInterceptor. Normalmente no debemos realizar ningún tipo de tarea con estas etiquetas.

- `<Conector>`: esta etiqueta representa una conexión con el usuario, en esta etiqueta podemos especificar el puerto TCP/IP en el que se va a ejecutar el servidor, por defecto es el puerto 8080.

Normalmente la etiqueta que más se suele utilizare del fichero SERVER.XML es la etiqueta <Context>. Pasemos ahora al siguiente fichero de configuración, el fichero WEB.XML.

Fichero WEB.XML

Existe un fichero WEB.XML dentro del directorio CONF que define una serie de valores por defecto para todas las aplicaciones Web del servidor, pero cada aplicación WEB, en su directorio WEB-INF, puede tener su propio fichero WEB.XML, y así poder sobrescribir estos valores por defecto o añadir nuevos aspectos a la configuración de la aplicación Web correspondiente.

El fichero de configuración WEB.XML posee las siguientes funciones:

- Ofrecer una descripción de la aplicación Web.
- Realizar una correspondencia de nombre de servlets con URLs.
- Establecer la configuración de la sesión.
- Mapeo de librerías de etiquetas.
- Informar de los tipos MIME soportados.
- Establecer la página por defecto de la aplicación Web.
- Definir los parámetros de inicialización de un servlet.

De momento vamos a comentar como definir los parámetros de inicialización de un servlet, pero antes de esto tenemos que ver como asignar un nombre a un servlet, ya que los parámetros de inicialización se le asignan a un servlet a través de su nombre.

Primero debemos asociar un nombre con el fichero de la clase del servlet, una vez hecho esto asociamos el nombre del servlet con el parámetro de inicialización. Todas las etiquetas XML que vamos a utilizar se encuentran englobadas por una etiqueta general llamada <web-app> y que contiene toda la configuración para la aplicación Web en la que se encuentra el fichero WEB.XML. La estructura inicial de este fichero es la que se muestra en el Código Fuente 68.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
    "http://java.sun.com/j2ee/dtds/web-app_2.2.dtd">

<web-app>

</web-app>
```

Código Fuente 68

Dentro de una etiqueta <servlet> indicaremos la correspondencia entre el nombre que se le va a dar al servlet y el fichero que contiene la clase del mismo y los parámetros de inicialización que se van a utilizar en el servlet, además de otros aspectos de configuración del servlet. Es precisamente dentro de esta etiqueta dónde podremos indicar si el servlet se debe instanciar cuando se inicie la ejecución del servidor Tomcat.

Mediante la etiqueta <servlet-name> indicamos el nombre que se le va a asignar al servlet. El valor que se indique en esta etiqueta puede ser utilizado en la URL que invoca al servlet, así por ejemplo si al servlet MuestraMensaje le asignamos el nombre mensaje, podremos invocarlo desde el navegador con la URL http://localhost:8080/ejemplos/servlet/mensaje. Como se puede comprobar debemos seguir utilizando el directorio servlet.

Para indicar el servlet al que vamos a asignar el nombre especificado en la etiqueta <servlet-name>, utilizaremos la etiqueta <servlet-class>. En esta etiqueta se indica el nombre de la clase del servlet, así si tomamos el ejemplo anterior, deberíamos especificar la clase MuestraMensaje.

Tanto la etiqueta <servlet-name> como <servlet-class> son etiquetas obligatorias que forman parte de la etiqueta <servlet>.

De esta forma ya tendríamos un servlet determinado identificado mediante un nombre. En este momento el fichero WEB.XML tiene el aspecto del mostrado en el Código Fuente 69.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
    "http://java.sun.com/j2ee/dtds/web-app_2.2.dtd">

<web-app>
  <servlet>
      <servlet-name>
         mensaje
      </servlet-name>
      <servlet-class>
         MuestraMensaje
      </servlet-class>
  </servlet>

</web-app>
```

Código Fuente 69

Para asignar ahora los parámetros de inicialización al servlet hacemos uso de otras etiquetas llamadas <init-param>, <paran-name> y <param-value>. La etiqueta <init-param> indica que se va a definir un parámetro para el servlet actual, es decir, en servlet que se encuentra definido entre las etiquetas <servlet></servlet>.

La etiqueta <init-param> posee varios subelementos que comentamos a continuación:

- `<param-name>`: etiqueta en la que indicamos el nombre del parámetro de inicialización del servlet. El nombre del parámetro es case-sensitive, es decir, se distingue entre minúsculas y mayúsculas, esta norma la podemos aplicar como general a todos los elementos identificativos de los ficheros XML de configuración del servidor Tomcat y sus aplicaciones.

- `<param-value>`: etiqueta en la que se indica el valor del parámetro. Estos valores siempre se van a recuperar como objeto String, el servlet será encargado de convertir el valor del parámetro al tipo correspondiente.

- `<description>`: permite especificar una descripción del parámetro, esta etiqueta se utiliza a nivel de documentación.

De esta forma si queremos inicializar el servlet asignándole al parámetro repeticiones el valor de 4 y al parámetro mensaje el valor de la cadena “Hola, que tal”, el fichero de configuración de la aplicación Web, tendrá el aspecto del Código Fuente 70.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE web-app
    PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.2//EN"
    "http://java.sun.com/j2ee/dtds/web-app_2.2.dtd">

<web-app>
  <servlet>
       <servlet-name>
          mensaje
       </servlet-name>
       <servlet-class>
          MuestraMensaje
       </servlet-class>
       <init-param>
         <param-name>repeticiones</param-name>
         <param-value>4</param-value>
         <description>Número de veces que se repite el mensaje</description>
       </init-param>
       <init-param>
         <param-name>mensaje</param-name>
         <param-value>Hola, que tal</param-value>
         <description>Texto del mensaje</description>
       </init-param>
  </servlet>

</web-app>
```

Código Fuente 70

Dentro de la etiqueta <servlet> también podemos indicar que el servlet se cree, y ejecute el método init(), al iniciar la ejecución del servidor Tomcat. Para ello utilizaremos la etiqueta <load-on-startup>, si a este etiqueta no se le facilita ningún valor, el servidor instanciará y cargará el servlet cuando le sea posible al iniciarse. Pero si deseamos que se instancie el servlet atendiendo a una prioridad con respecto al resto de los servlets de la aplicación Web, especificaremos un entero como valor de la etiqueta. Cuanto mayor sea este número menor será la prioridad de creación del servlet, de esta forma primero se crearán los servlets con prioridad 1, luego los de prioridad 2, y así con el resto.

En siguiente aspecto que vamos a comentar del fichero WEB.XML es la forma en la que nos permite la correspondencia de los servlets con patrones de URLs.

La etiqueta <servlet-mapping> nos permite asignar a un servlet, identificado por su nombre asignado en la subetiqueta <servlet-name> de la etiqueta <servlet>, una URL determinada. La etiqueta <servlet-mapping> contiene dos subetiquetas <servlet-name> y <url-pattern>, que pasamos a comentar a continuación:

- `<servlet-name>`: en esta etiqueta indicamos el nombre del servlet al que queremos asociar con una URL determinada, o bien con un patrón de URL, como veremos más adelante.

- `<url-pattern>`: en esta etiqueta especificamos el patrón de la URL que queremos asignar al servlet. El contenido del cuerpo de esta etiqueta puede ser una cadena sencilla que indique una ruta más o menos compleja de la URL que se va a asociar con el servlet correspondiente, o bien una cadena con caracteres especiales que sirvan como un patrón.

Así por ejemplo si queremos invocar al servlet MuestraMensaje utilizando la URL http://localhost:8080/ejemplos/miServlet, deberemos añadir el Código Fuente 71 en el fichero WEB.XML de la aplicación ejemplos, justamente después del cierre de la etiqueta <servlet> que habíamos utilizado para definir el nombre del servlet y sus parámetros de inicialización.

```
<servlet-mapping>
      <servlet-name>mensaje</servlet-name>
      <url-pattern>/miServlet</url-pattern>
</servlet-mapping>
```

Código Fuente 71

Como se puede comprobar ya no se utiliza el directorio servlet en ningún lugar de la URL que utilizamos para invocar el servlet.

En el ejemplo anterior la etiqueta <url-pattern> contenía en su cuerpo únicamente una cadena que identificaba un directorio a partir de la URL de la aplicación actual, pero esto puede ser más complejo teniendo la posibilidad de haber especificado una estructura de directorios más compleja, como podría haber sido miServlet/primeros/uno. De esta forma el servlet MuestraMensaje podría haber sido invocado mediante la URL http://localhost:8080/ejemplos/miServlet/primeros/uno.

También es posible el uso de asteriscos (*) como caracteres especiales. El asterisco se puede utilizar como un comodín en la parte final de la URL que se va a asignar al servlet, así por ejemplo si en la etiqueta <url-pattern>, utilizamos la cadena miServlet/mensajes/*, todas las URLs que comiencen de la forma http://localhost:8080/ejemplos/miServlet/mensajes, invocarán el servlet MuestraMensaje. De esta forma si utilizamos la URL http://localhost:8080/ejemplos/miServlet/mensajes/la/898989/8898, invocaremos al ya conocido servlet MuestraMensaje.

Otro uso avanzado que podemos realizar de la etiqueta <url-pattern> es la de asignar los ficheros con una extensión determinada a un servlet determinado. Para ello en el cuerpo de la etiqueta <url-pattern> utilizaremos una cadena compuesta de un asterisco, un punto y el nombre de la extensión de los ficheros que queremos que se correspondan con la invocación del servlet. Así por ejemplo, si queremos que al utilizar una URL con un fichero que posea la extensión .ser se invoque al servlet MuestraMensaje deberemos añadir el Código Fuente 72 al fichero WEB.XML.

```
<servlet-mapping>
     <servlet-name>mensaje</servlet-name>
     <url-pattern>*.ser</url-pattern>
</servlet-mapping>
```

Código Fuente 72

Este es el mecanismo utilizado para invocar las JavaServer Pages (JSP), todos los ficheros con extensión .jsp se mapean o se corresponden con el servlet que realiza las funciones de compilador de páginas JSP.

Otra funcionalidad del fichero de configuración WEB.XML es la de indicar las librerías de etieutas personalizadas que se puede utilizar en la aplicación Web actual. Para ellos disponemos de la etiqueta <taglib>.

La etiqueta <taglib> dentro del fichero WEB.XML presenta dos subelementos que nos permiten registrar las librerías de etiquetas personalizadas necesarias en una aplicación Web. Los subelementos que la etiqueta <taglib> presenta son los comentados a continuación:

- `<taglib-uri>`: esta etiqueta contiene una URL que va a identificar a la librería de etiquetas, no se trata de una URL real, sino la que se va a utilizar en el atributo de la directiva taglib para identificar una librería determinada.

- `<taglib-location>`: este etiqueta indica la ruta real dónde se localiza el fichero TLD.

En el Código Fuente 73 se muestra un fragmento del fichero WEB.XML de una aplicación Web que va a utilizar una librería de etiquetas.

```
<taglib>
 <taglib-uri>
      libreriaEtiquetas.tld
 </taglib-uri>
 <taglib-location>
     /WEB-INF/tlds/libreriaEtiquetas.tld
 </taglib-location>
</taglib>
```

Código Fuente 73

Los ficheros TLD se suelen situar en el directorio WEB-INF/TLDS de la aplicación Web, de todas formas, en el siguiente apartado comentaremos la estructura de directorios que suelen presentar las aplicaciones Web dentro de Jakarta Tomcat.

Dentro del fichero WEB.XML podemos mapear extensiones MIME al tipo de fichero correspondiente, en el fichero WEB.XML que se encuentra en el directorio CONF, y que define la configuración por defecto de todas las aplicaciones Web del servidor, se encuentran mapeadas una serie de extensiones, las más comunes que podemos encontrar en el entorno de Internet, a sus tipos de ficheros correspondientes.

En el Código Fuente 74 se puede ver un fragmento del fichero WEB.XML en el que se realiza la correspondencia de la extensión mov con los tipos de fichero quicktime, los tipos MIME definidos son los que va a saber interpretar de manera satisfactoria el servidor Web.

```
<mime-mapping>
  <extension>
     mov
  </extension>
  <mime-type>
     video/quicktime
  </mime-type>
```

```
</mime-mapping>
```

Código Fuente 74

Como se deduce del código anterior, existe una etiqueta principal llamada <mime-mapping> que permite realizar la correspondencia del tipo de fichero con la extensión. La extensión se indica con la subetiqueta <extension>, y el tipo MIME que se corresponde con la extensión se especifica mediante la subetiqueta <mime-type>.

Otro aspecto que nos permite configurar para cada aplicación Web el fichero WEB.XML es la página de inicio de inicio o página por defecto de la aplicación. Esta página será la que se cargará si no especificamos ninguna página concreta de la aplicación Web en la URL, es decir, si utilizamos una dirección del tipo http://localhost:8080/ejemplos/ se mostrará la página o documento por defecto que hayamos indicado en el fichero WEB.XML.

Para indicar la página o páginas por defecto de la aplicación Web utilizaremos la etiqueta <welcome-file-list>. Esta etiqueta permite definir varias páginas por defecto, si no encuentra una de ellas buscará la siguiente el servidor, para definir cada una de las páginas se utiliza la subetiqueta <welcome-file>, que contendrá el nombre del fichero en cuestión.

En el Código Fuente 75 se ofrece un fragmento del fichero WEB.XML en el que se define la lista de páginas por defecto de las aplicaciones del servidor.

```
<welcome-file-list>
   <welcome-file>
        index.jsp
    </welcome-file>
   <welcome-file>
        index.html
    </welcome-file>
   <welcome-file>
        index.htm
    </welcome-file>
</welcome-file-list>
```

Código Fuente 75

En el siguiente apartado vamos a comentar la estructura de directorios que presenta una aplicación Web en el servidor Jakarta Tomcat.

Estructura de directorios de una aplicación Web

Una aplicación Web en Tomcat, quedaba definida a partir del directorio indicado en la etiqueta <Context> del fichero SERVER.XML del servidor.

A lo largo del texto hemos ido comentando los distintos directorios que puede presentar una aplicación Web en el servidor Jakarta Tomcat y la finalidad de cada uno de ellos, ahora en este apartado vamos a realizar un resumen de todos ellos.

Vamos a suponer que tenemos definida una aplicación Web, según el Código Fuente 76, llamada ejemplos, y a partir de ella vamos a comentar la estructura de directorios que presenta.

```
<Context path="/ejemplos" docBase="c:\work\ejemplos" debug="0" reloadable="true" >
</Context>
```

Código Fuente 76

En el directorio raíz de la aplicación Web, es decir, el directorio C.\WORK\EJEMPLOS, podemos incluir todos lo ficheros HTML, JSP, imágenes, etc., de los que consta la aplicación Web, así como distintos subdirectorios de contenidos que pueden contener más páginas JSP o HTML.

A partir del directorio raíz existe un subdirectorio llamado WEB-INF que contendrá el fichero de configuración de la aplicación Web, es decir, el fichero WEB.XML.

Dentro del directorio WEB-INF existe un subdirectorio llamado CLASSES, que contendrá todas las clases de los servlets y componentes JavaBeans de la aplicación Web actual. El directorio CLASSES pude contener múltiples subdirectorios para organizar las distintas clases, ya sean servlets o Beans, en paquetes.

También dentro del directorio WEB-INF encontremos el subdirectorio TLDS en el que debemos situar los ficheros descriptores de las librerías de etiquetas personalizadas, vistos en el capítulo anterior. Además podemos encontrar también dentro del directorio WEB-INF el subdirectorio LIB, que contendrá ficheros JAR con diversas clases utilizadas en la aplicación Web.

Arranque de Jakarta-Tomcat desde Apache

Una vez arrancado Jakarta-Tomcat,(podemos ejecutar jk_nt_service.exe para que se instale como un servicio mas de Windows 2000), se crea automáticamente un fichero llamado tomcat-apache.conf, con la configuración necesaria para que funcione con Apache.

```
LoadModule jserv_module modules/ApacheModuleJServ.dll
ApJServManual on
ApJServDefaultProtocol ajpv12
ApJServSecretKey DISABLED
ApJServMountCopy on
ApJServLogLevel notice
ApJServDefaultPort 8007
AddType text/jsp .jsp
AddHandler jserv-servlet .jsp

Alias /examples "E:/jakarta-tomcat/webapps/examples"
<Directory "E:/jakarta-tomcat/webapps/examples">
    Options Indexes FollowSymLinks
</Directory>
ApJServMount /examples/servlet /examples
<Location "/examples/WEB-INF/">
    AllowOverride None
    deny from all
</Location>
<Directory "E:/jakarta-tomcat/webapps/examples/WEB-INF/">
    AllowOverride None
    deny from all
</Directory>
<Location "/examples/META-INF/">
    AllowOverride None
    deny from all
```

```
</Location>
<Directory "E:/jakarta-tomcat/webapps/examples/META-INF/">
    AllowOverride None
    deny from all
</Directory>
Alias /admin "E:/jakarta-tomcat/webapps/admin"
<Directory "E:/jakarta-tomcat/webapps/admin">
    Options Indexes FollowSymLinks
</Directory>
ApJServMount /admin/servlet /admin
<Location "/admin/WEB-INF/">
    AllowOverride None
    deny from all
</Location>
<Directory "E:/jakarta-tomcat/webapps/admin/WEB-INF/">
    AllowOverride None
    deny from all
</Directory>
<Location "/admin/META-INF/">
    AllowOverride None
    deny from all
</Location>
<Directory "E:/jakarta-tomcat/webapps/admin/META-INF/">
    AllowOverride None
    deny from all
</Directory>

Alias /webdev "E:/jakarta-tomcat/webapps/webdev"
<Directory "E:/jakarta-tomcat/webapps/webdev">
    Options Indexes FollowSymLinks
</Directory>
ApJServMount /webdev/servlet /webdev
<Location "/webdev/WEB-INF/">
    AllowOverride None
    deny from all
</Location>
<Directory "E:/jakarta-tomcat/webapps/webdev/WEB-INF/">
    AllowOverride None
    deny from all
</Directory>
<Location "/webdev/META-INF/">
    AllowOverride None
    deny from all
</Location>
<Directory "E:/jakarta-tomcat/webapps/webdev/META-INF/">
    AllowOverride None
    deny from all
</Directory>

ApJServMount /servlet /ROOT
Alias /test "E:/jakarta-tomcat/webapps/test"
<Directory "E:/jakarta-tomcat/webapps/test">
    Options Indexes FollowSymLinks
</Directory>
ApJServMount /test/servlet /test
<Location "/test/WEB-INF/">
    AllowOverride None
    deny from all
</Location>
<Directory "E:/jakarta-tomcat/webapps/test/WEB-INF/">
    AllowOverride None
    deny from all
</Directory>
<Location "/test/META-INF/">
    AllowOverride None
    deny from all
</Location>
```

```
<Directory "E:/jakarta-tomcat/webapps/test/META-INF/">
    AllowOverride None
    deny from all
</Directory>
```

Código fuente 77. Fichero tomcat-apache.conf (generado automáticamente)

Este es el fichero que nos va a servir de conexión entr Jakarta-Tomcat y Apache. Pero una vez mas, nuestro servidor necesita un módulo para poder utilizar este complemento. Este modulo es ApacheModuleJServ.dll, el cual deberá encontrarse dentro del directorio module de APACHE_HOME.

Necesitamos el archivo de configuración de Apache para poner en marcha esta unión. Insertaremos la línea del Código fuente 78 al final de nuestro fichero. Debemos indicar la ruta completa donde se encuentra el archivo.

```
include e:/jakarta-tomcat/conf/tomcat-apache.conf
```

Código fuente 78. http.conf de Apache

Siempre con Jakarta-Tomcat funcionando, pondremos en marcha nuestro servidor Apache, el cual tomará la configuración del fichero tomcat-apache.conf. Una vez en servicio podemos llamar a nuestro directorio examples, pero siendo ahora un directorio que cuelga de nuestro localhost.

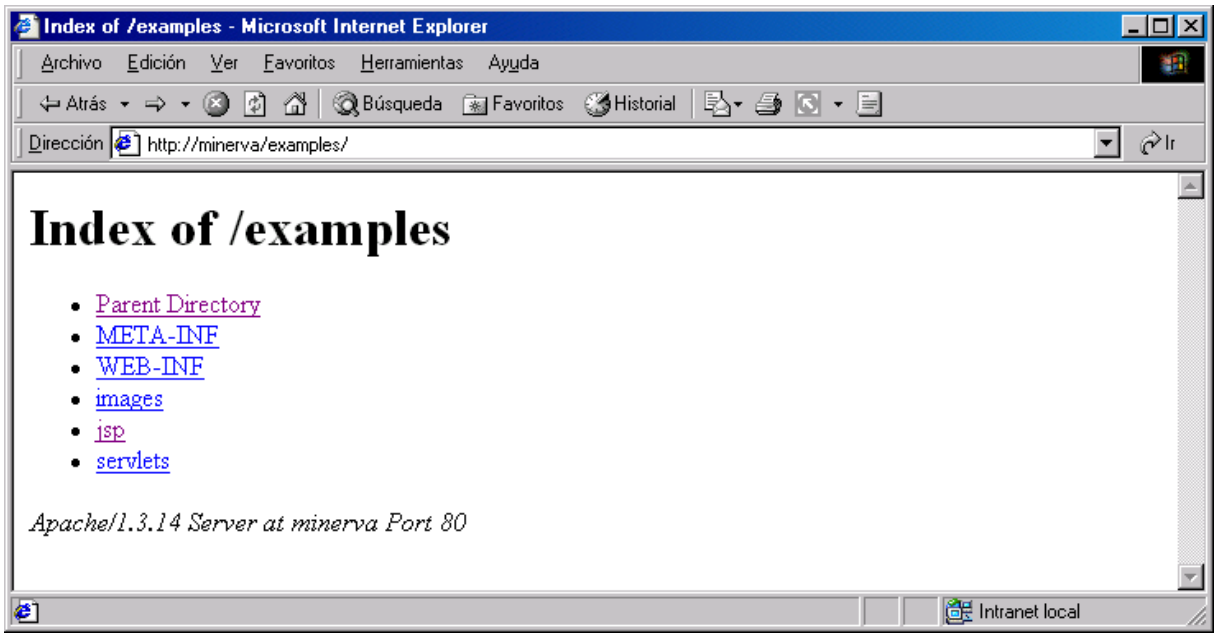


Figura 41. examples como directorio de localhost