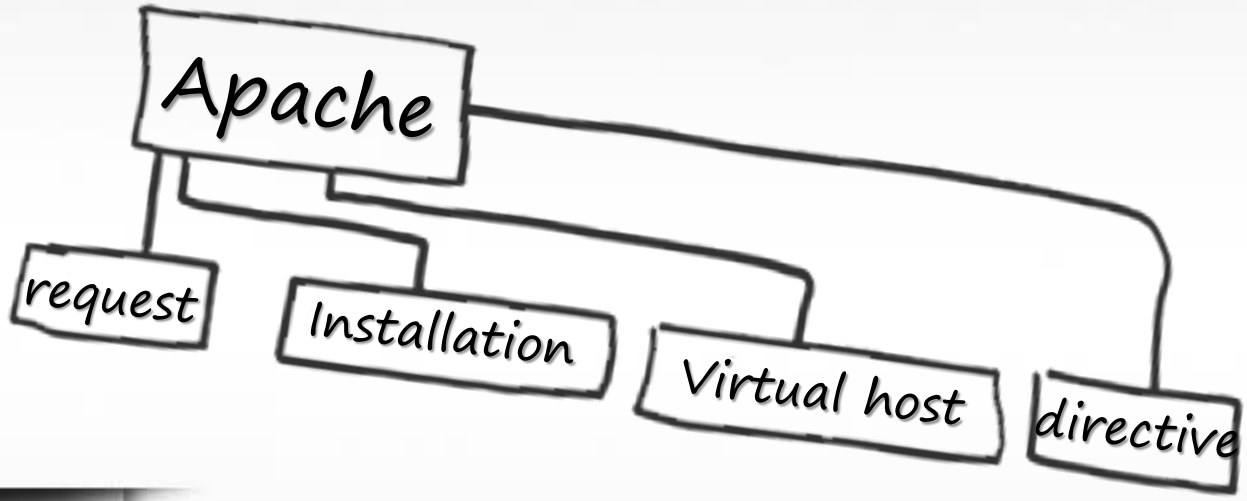
NOW

Apache Web Server



OPEN SOURCE DEPARTMENT

Information Technology Institute

Course Materials

You can access the course materials via this link

http://goo.gl/LwyPUe

Contents

- Clients, servers and URLs
- Anatomy of request and response
- HTTP status codes
- Web servers
- Configuration files
- Directives
- Modules
- Virtual hosts
- .htacess

Clients, Servers, and URLs

- Addresses on the Web are expressed with URLs - Uniform Resource Locators - which specify
  - a protocol (e.g. http),
  - a servername (e.g. www.apache.org),
  - a URL-path (e.g. /docs/current/getting-started.html), and
  - possibly a query string (e.g. ?arg=value) used to pass additional arguments to the server

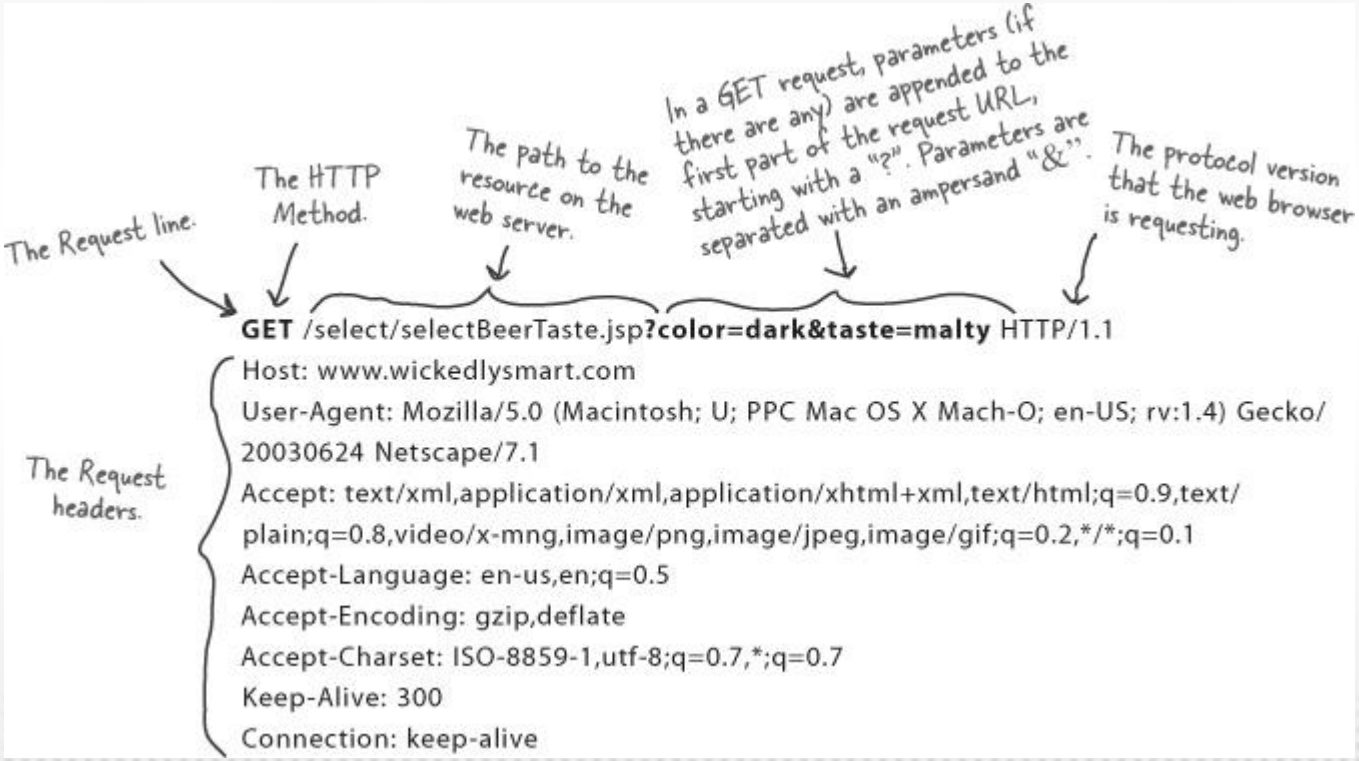
Clients, Servers, and URLs

- A client (e.g., a web browser) connects to a server (e.g., your Apache HTTP Server), with the specified protocol, and makes a **request** for a resource using the URL-path.
- The URL-path may represent any number of things on the server. It may be a file (like getting-started.html) or some kind of program file (like index.php).

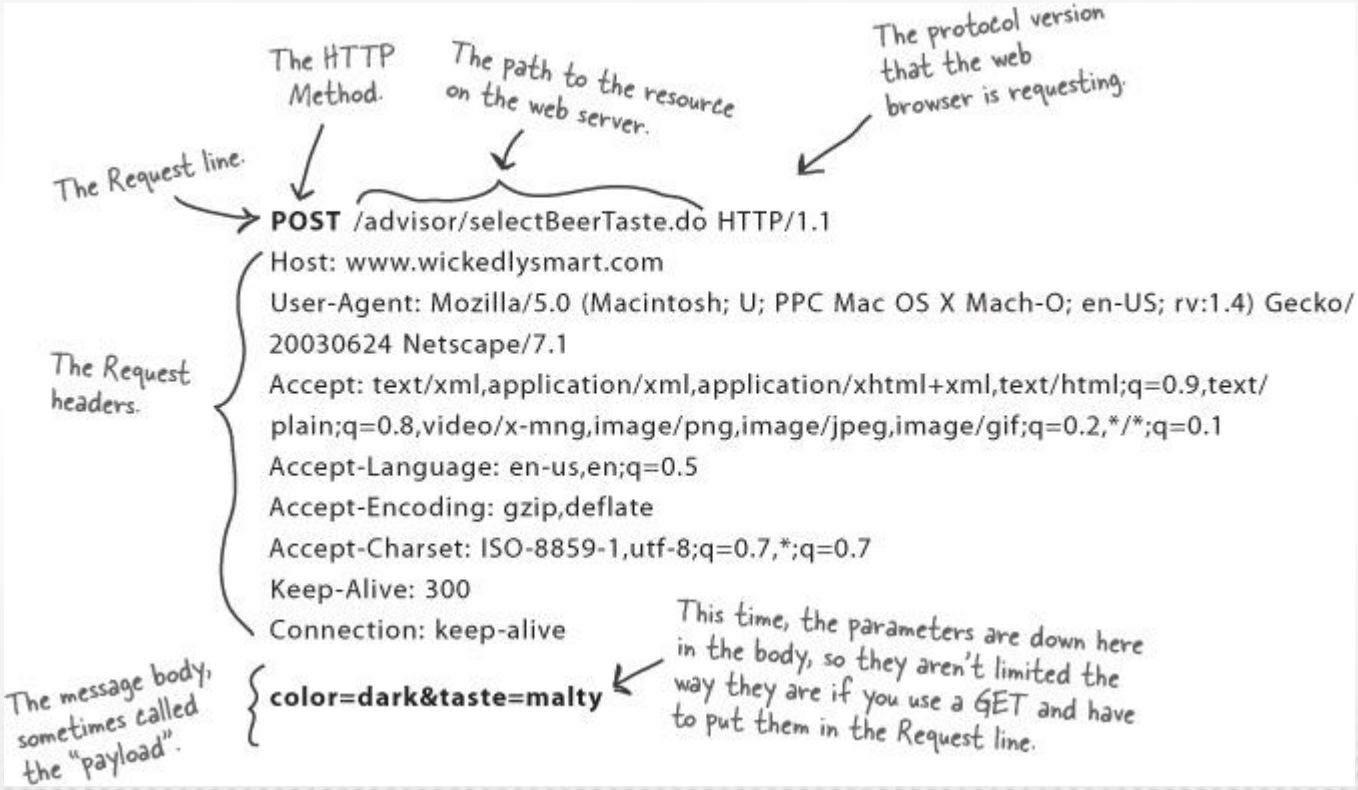
Clients, Servers, and URLs

- The server will send a **response** consisting of
  - a status code and,
  - optionally, a response body.
- The status code indicates whether the request was successful, and, if not, what kind of error condition there was. This tells the client what it should do with the response.

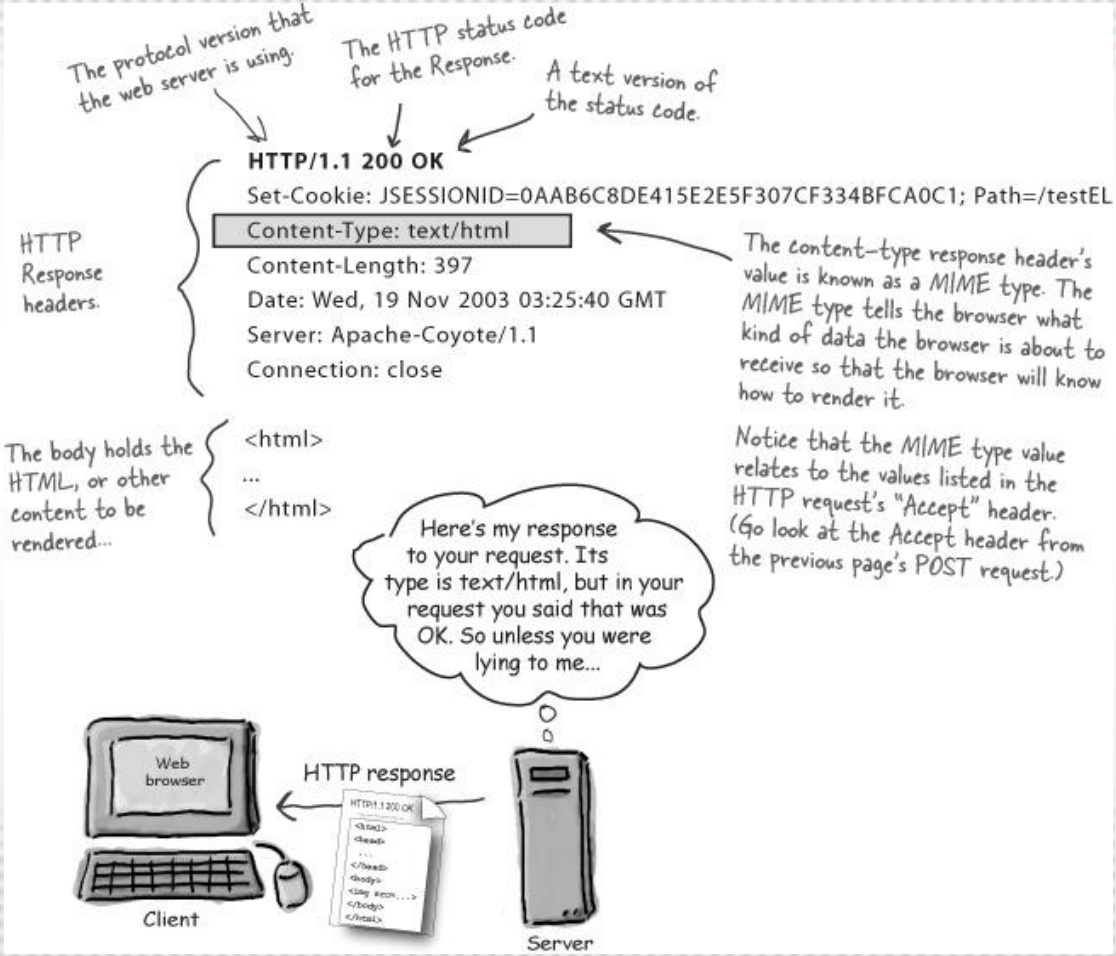
Anatomy of GET Request



Anatomy of POST Request



Anatomy of Response



HTTP status codes

- 1xx Informational
- 2xx Success
- 3xx Redirection
- 4xx Client Error
- 5xx Server Error

HTTP status codes Examples

- 200 OK
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found
- 500 Internal Server Error

See full list at:

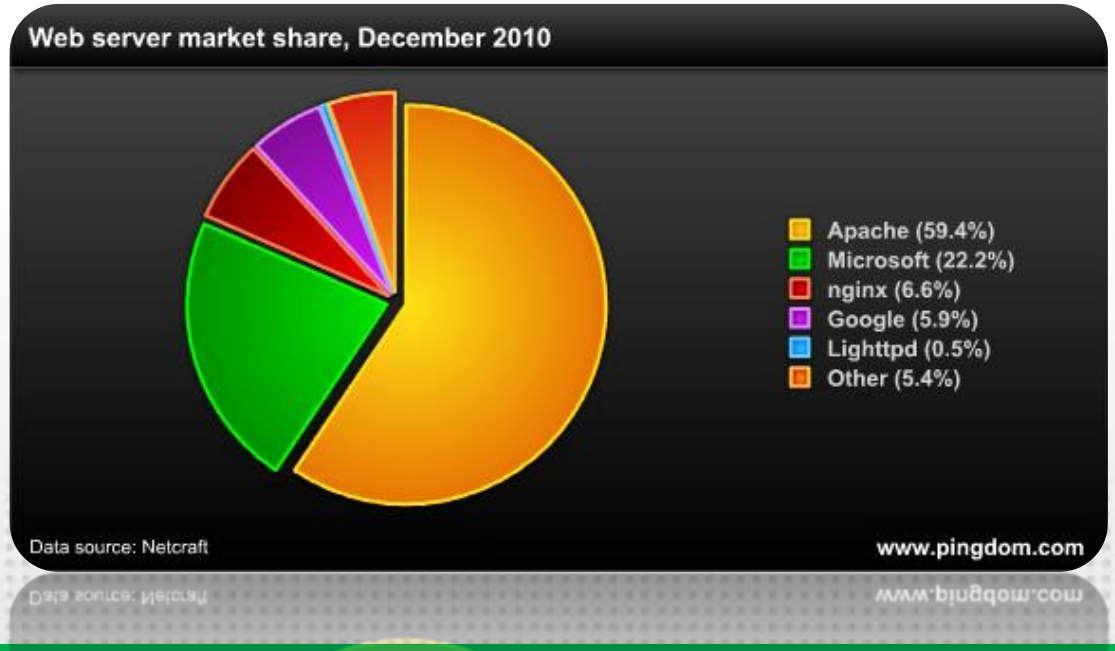
http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

So , What is a Web Server?

- This is a computer that's sole purpose is to distribute information that is hosted within its hard drives.
- Depending on the information, it is accessed and distributed differently.
- A detailed example would be standard Web Pages that are accessed via the Internet protocol HTTP on port 80 and distributed back in the same fashion. be stored on a web server.

Web Servers

- Any computer can be turned into a Web server by installing server software and connecting the machine to the Internet.
- There are many server software applications.



Apache Vs IIS

- Apache is free while IIS is packaged with Windows.
- IIS only runs on Windows while Apache can run on almost any OS including UNIX, Apple's OS X, and on most Linux Distributions.
- IIS has a dedicated staff to answer most problems while support for Apache comes from the community itself.
- IIS is optimized for Windows because they are from the same company.
- The Windows OS is prone to security risks.

Configuration Files

- The Apache HTTP Server is configured via simple text files.
- The default configuration file is usually called `httpd.conf`.
- The configuration is frequently broken into multiple smaller files, for ease of management. These files are loaded via the `Include` directive.

Directives

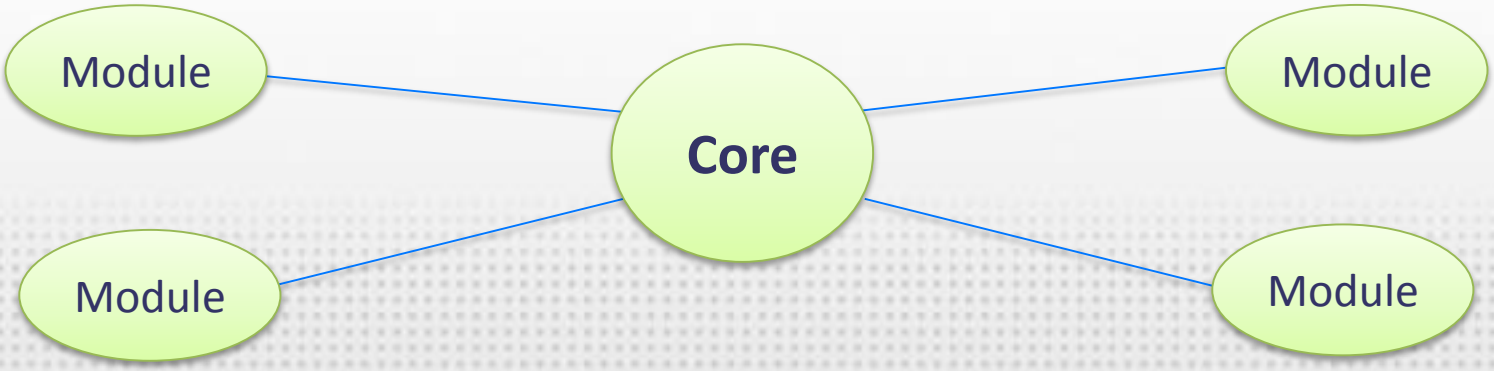
- The server is configured by placing configuration directives in configuration files.
- A directive is a keyword followed by one or more arguments that set its value.
- In addition to the main configuration files, certain directives may go in `.htaccess` files located in the content directories. `.htaccess` files are primarily for people who do not have access to the main server configuration file(s)

Where should I put directive?

- If it is a global setting, it should appear in the configuration file, outside of any `<Directory>`, `<Location>`, `<VirtualHost>`, or other section.
- If it is to apply only to a particular directory, then it should go inside a `<Directory>` section referring to that directory, and so on.

Apache Structure

- Apache has always accommodated a wide variety of environments through its modular design.
- This design allows the web-master to choose which features will be included in the server by selecting which modules to load either at compile-time or at run-time.



Installation

- Ubuntu

```
$ Sudo apt-get update
$ Sudo apt-get install tasksel
$ Sudo tasksel install lamp-
server
```

- CentOS

```
# yum install httpd
```

Main Configuration file

- Ubuntu

`/etc/apache2/apache2.conf`

- CentOS

`/etc/httpd/conf/httpd.conf`

Public Web files & log files

- Ubuntu

```
/var/www/html
/var/log/apache2
```

- CentOS

```
/var/www/html
/var/log/httpd
```

Starting/Stopping Server

- Ubuntu

```
$ Sudo /etc/init.d/apache2 start
$ Sudo Service apache2 stop
```

- CentOS

```
# /etc/init.d/httpd restart
# Service httpd reload
```

Main configuration file


```
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
#
# NOTE!  If you intend to place this on an NFS (or otherwise network)
# mounted filesystem then please read the LockFile documentation
# (available at <URL:http://httpd.apache.org/docs/2.2/mod/mpm_common.html#lockfi
le>);
# you will save yourself a lot of trouble.
#
# Do NOT add a slash at the end of the directory path.
#
ServerRoot "/etc/httpd"

#
# PidFile: The file in which the server should record its process
# identification number when it starts.
#
PidFile run/httpd.pid
#
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 120
-- INSERT --
```

Main configuration file

- `ServerRoot` directive

The path to the server's configuration

- `PidFile` directive

The process identification number for the httpd registered at starting the server

- `ServerName` directive

This is where you declare the name of your website

- `DocumentRoot` directive

This is where your web documents (html files, images etc) should be located.

Terms used

Description:	A brief description of the purpose of the directive.
Syntax:	This indicates the format of the directive as it would appear in a configuration file
Status:	This indicates how tightly bound into the Apache Web server the directive isPossible values for this attribute are: Core If a directive is listed as having "Core" status, that means it is part of the innermost portions of the Apache Web server, and is always available. MPM: A directive labeled as having "MPM" status is provided by a Multi-Processing Module.
Module:	This quite simply lists the name of the source module which defines the directive.

Terms used

Context:	This indicates where in the server's configuration files the directive is legal. **Server config:** This means that the directive may be used in the server configuration files (e.g., httpd.conf), but not within any <VirtualHost> or <Directory> containers. It is not allowed in .htaccess files at all. **Virtual host :**This context means that the directive may appear inside <VirtualHost> containers in the server configuration files. **Directory :**A directive marked as being valid in this context may be used inside <Directory>, <Location>, <Files>, and <Proxy> containers in the server configuration files, subject to the restrictions outlined in Configuration Sections. **.htaccess:** If a directive is valid in this context, it means that it can appear inside per-directory .htaccess files. It may not be processed, though depending upon the overrides currently active.

AllowOverride

Description:	Types of directives that are allowed in .htaccess files
Syntax:	AllowOverride All|None|directive-type [directive-type] ...
Default:	AllowOverride None (2.3.9 and later), AllowOverride All (2.3.8 and earlier)
Context:	directory
Status:	Core
Module:	core

When the server finds an .htaccess file (as specified by AccessFileName) it needs to know which directives declared in that file can override earlier configuration directives.

KeepAlive

Description:	Enables HTTP persistent connections
Syntax:	KeepAlive On|Off
Default:	KeepAlive On
Context:	server config, virtual host
Status:	Core
Module:	core

The Keep-Alive extension to HTTP/1.0 and the persistent connection feature of HTTP/1.1 provide long-lived HTTP sessions which allow multiple requests to be sent over the same TCP connection. In some cases this has been shown to result in an almost 50% speedup in latency times for HTML documents with many images. To enable Keep-Alive connections, set KeepAlive On.

MaxKeepAliveRequests

Description:	Number of requests allowed on a persistent connection
Syntax:	MaxKeepAliveRequests number
Default:	MaxKeepAliveRequests 100
Context:	server config, virtual host
Status:	Core
Module:	core

The MaxKeepAliveRequests directive limits the number of requests allowed per connection when KeepAlive is on. If it is set to 0, unlimited requests will be allowed. We recommend that this setting be kept to a high value for maximum server performance.

ErrorLog

Description:	Location where the server will log errors
Syntax:	ErrorLog file-path|syslog[:facility]
Default:	ErrorLog logs/error_log (Unix) ErrorLog logs/error.log (Windows and OS/2)
Context:	server config, virtual host
Status:	Core
Module:	core

The ErrorLog directive sets the name of the file to which the server will log any errors it encounters. If the file-path is not absolute then it is assumed to be relative to the ServerRoot

Directory

Description:	Enclose a group of directives that apply only to the named file-system directory, sub-directories, and their contents.
Syntax:	<Directory directory-path> ... </Directory>
Context:	server config, virtual host
Status:	Core
Module:	core

<Directory> and </Directory> are used to enclose a group of directives that will apply only to the named directory, sub-directories of that directory, and the files within the respective directories. Any directive that is allowed in a directory context may be used.

DirectoryMatch

Description:	Enclose directives that apply to the contents of file-system directories matching a regular expression.
Syntax:	<DirectoryMatch regex> ... </DirectoryMatch>
Context:	server config, virtual host
Status:	Core
Module:	core

<DirectoryMatch> and </DirectoryMatch> are used to enclose a group of directives which will apply only to the named directory (and the files within), the same as <Directory>. However, it takes as an argument a regular expression

Files

Description:	Contains directives that apply to matched filenames
Syntax:	<Files filename> ... </Files>
Context:	server config, virtual host, directory, .htaccess
Override:	All
Status:	Core

The <Files> directive limits the scope of the enclosed directives by filename. It is comparable to the <Directory> and <Location> directives. It should be matched with a </Files> directive. The directives given within this section will be applied to any object with a basename (last component of filename) matching the specified filename

FilesMatch

Description:	Contains directives that apply to regular-expression matched filenames
Syntax:	<FilesMatch regex> ... </FilesMatch>
Context:	server config, virtual host, directory, .htaccess
Override:	All
Status:	Core

The <FilesMatch> directive limits the scope of the enclosed directives by filename, just as the <Files> directive does. However, it accepts a regular expression.

Allow

Description:	Controls which hosts can access an area of the server
Syntax:	Allow from all|host|env=[!]env-variable [host|env=[!]env-variable] ...
Context:	directory, .htaccess
Override:	Limit
Status:	Extension

The Allow directive affects which hosts can access an area of the server. Access can be controlled by hostname, IP address, IP address range, or by other characteristics of the client request captured in environment variables.

Deny

Description:	Controls which hosts are denied access to the server
Syntax:	Deny from all|host|env=[!]env-variable [host|env=[!]env-variable] ...
Context:	directory, .htaccess
Override:	Limit
Status:	Extension

This directive allows access to the server to be restricted based on hostname, IP address, or environment variables. The arguments for the Deny directive are identical to the arguments for the Allow directive.

CacheEnable

Description:	Enable caching of specified URLs using a specified storage manager
Syntax:	CacheEnable cache_type [url-string]
Context:	server config, virtual host, directory, .htaccess
Status:	Extension
Module:	mod_cache

The CacheEnable directive instructs mod_cache to cache urls at or below url-string. The cache storage manager is specified with the cache_type argument. The CacheEnable directive can alternatively be placed inside either <Location> or <LocationMatch> sections to indicate the content is cacheable.

Modules

- Modules are extensions that enhance the basic functionality of the Web server
- There are two types of modules:
  - **Built-in modules**, which are compiled into Apache and will load with the server any time it is started. Their functionality cannot be removed without recompiling the package. These modules are also known as static.
  - **Loadable modules**, which can be loaded on and off as required. These are the shared modules

Modules

- You can always list the modules currently used by the server. The command below will display only the modules compiled into Apache.

```
$ httpd -l
```

- To list all modules, both static and shared

```
$ httpd  -m
```

- To enable module

```
$ Sudo a2enmod rewrite
```

IfModule

Description:	Encloses directives that are processed conditional on the presence or absence of a specific module
Syntax:	<IfModule [!]module-file|module-identifier> ... </IfModule>
Context:	server config, virtual host, directory, .htaccess
Override:	All
Status:	Core

The <IfModule test>...</IfModule> section is used to mark directives that are conditional on the presence of a specific module. The directives within an <IfModule> section are only processed if the test is true. If test is false, everything between the start and end markers is ignored.

Prefork Module

- This Multi-Processing Module (MPM) implements a non-threaded, pre-forking web server. Each server process may answer incoming requests, and a parent process manages the size of the server pool. It is appropriate for sites that need to avoid threading for compatibility with non-thread-safe libraries. It is also the best MPM for isolating each request, so that a problem with a single request will not affect any other.

Prefork Module


```
<IfModule mpm_prefork_module>
    StartServers              5
    MinSpareServers           5
    MaxSpareServers          10
    MaxClients              150
</IfModule>
```

Worker Module

- This Multi-Processing Module (MPM) implements a hybrid multi-process multi-threaded server. By using threads to serve requests, it is able to serve a large number of requests with fewer system resources than a process-based server. However, it retains much of the stability of a process-based server by keeping multiple processes available, each with many threads.

Worker Module


```
<IfModule mpm_worker_module>
    StartServers          2
    MinSpareThreads      25
    MaxSpareThreads      75
    MaxClients          150
</IfModule>
```

LoadModule

Description:	Links in the object file or library, and adds to the list of active modules
Syntax:	LoadModule module filename
Context:	server config
Status:	Extension
Module:	mod_so

The LoadModule directive links in the object file or library filename and adds the module structure named module to the list of active modules. Module is the name of the external variable of type module in the file, and is listed as the Module Identifier in the module documentation.

Virtual Hosts

- Virtual Hosts is an important, powerful feature that allows you to run several websites from a single computer.

```
<VirtualHost os.iti.gov.eg:80>
DocumentRoot /var/www/html/os-iti
ServerName os.iti.gov.eg
# other directives
</VirtualHost>
```

.htaccess

- .htaccess stands for *hypertext access*. This is the default name of the Apache directory-level configuration file.
- One of the most common uses is to require user authentication in order to serve certain web pages.

.htaccess for authentication


```
#.htaccess content
AuthType Basic
AuthName “Restricted web page”
AuthUserFile “/var/.htpasswd
require valid-user
```

- To create .htpasswd use this command

```
htpasswd -c .htpasswd username
```