SERVER-SIDE WEB PROGRAMMING UNIT1: SERVER CODE GENERATION

CFGS - DAW 2

Index

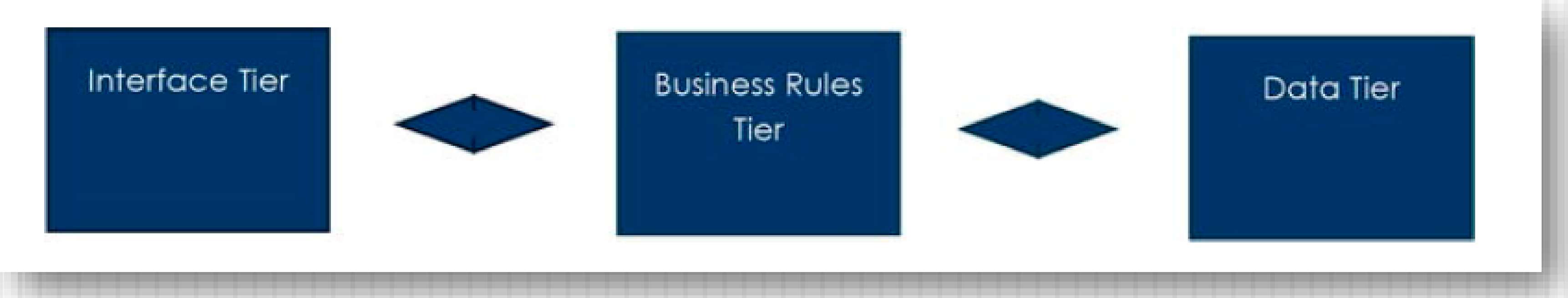
- Client-server programming models
- Web page generation
- Server side Programming languages
- Integration with Web servers
- Programming tools

1. Client-server programming models

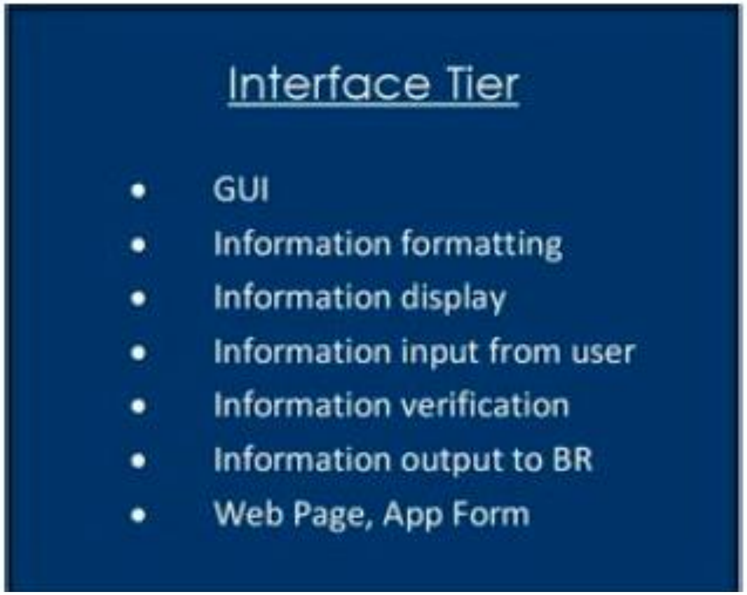
- **World Wide Web:**
  - Set of interconnected resources that make up the current human knowledge.
  - SW+HW:
    - Physical components
    - Communication protocols
    - DNS…

1. Client-server programming models

- A search engine can display a web page **interface** in a browser on the user 's PC:
  - Once the user enters the search request, the information can be transmitted across the Internet to a remote server (we don't know where), which executes a search application (we don't know what program language was used to create it).
  - The **application** then searches a database (we don't know what **DBMS** or where it is located) for the requested information.
  - The results are sent back to the application, which in turn, sends the information back to the browser (via the web server) on the user 's PC.

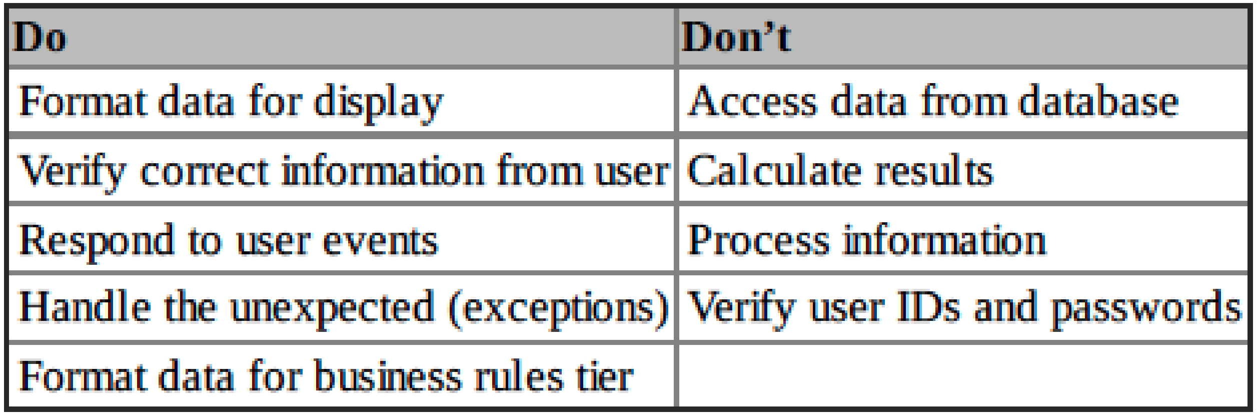


1. Client-server programming models



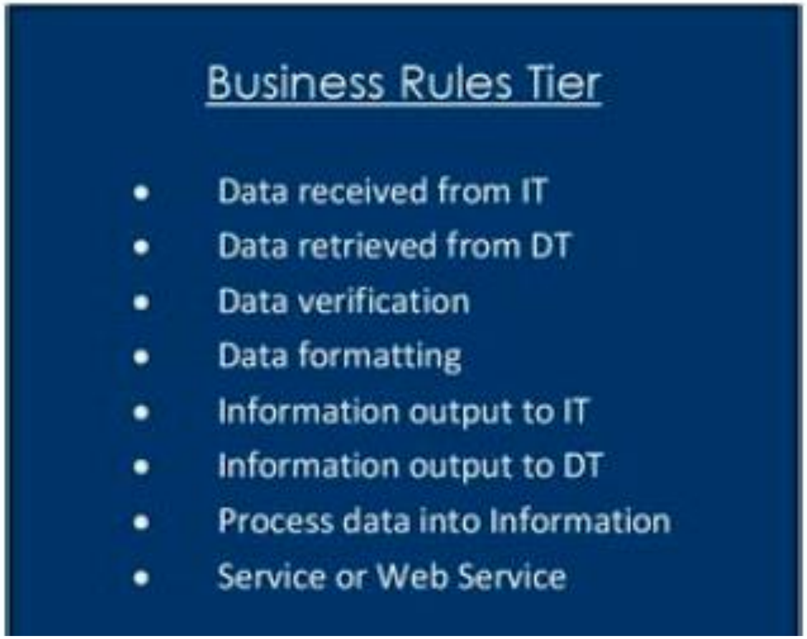
Interface Tier

- GUI
- Information formatting
- Information display
- Information input from user
- Information verification
- Information output to BR
- Web Page, App Form



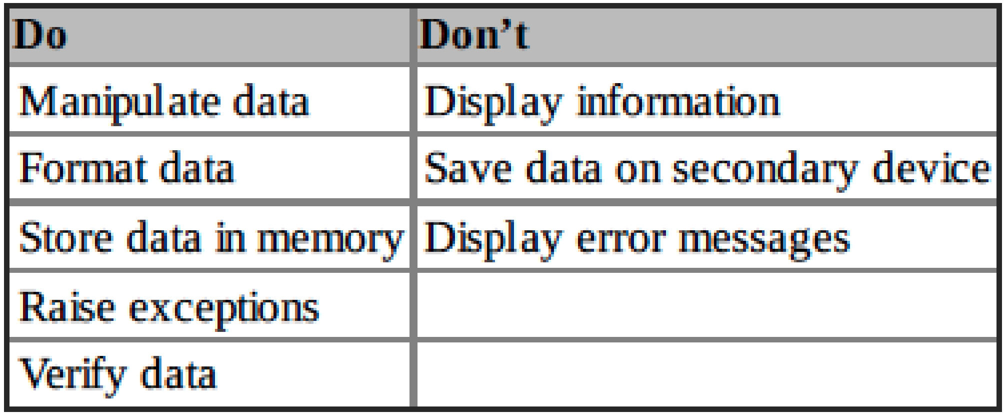
Do	Don’t
Format data for display	Access data from database
Verify correct information from user	Calculate results
Respond to user events	Process information
Handle the unexpected (exceptions)	Verify user IDs and passwords
Format data for business rules tier	

1. Client-server programming models



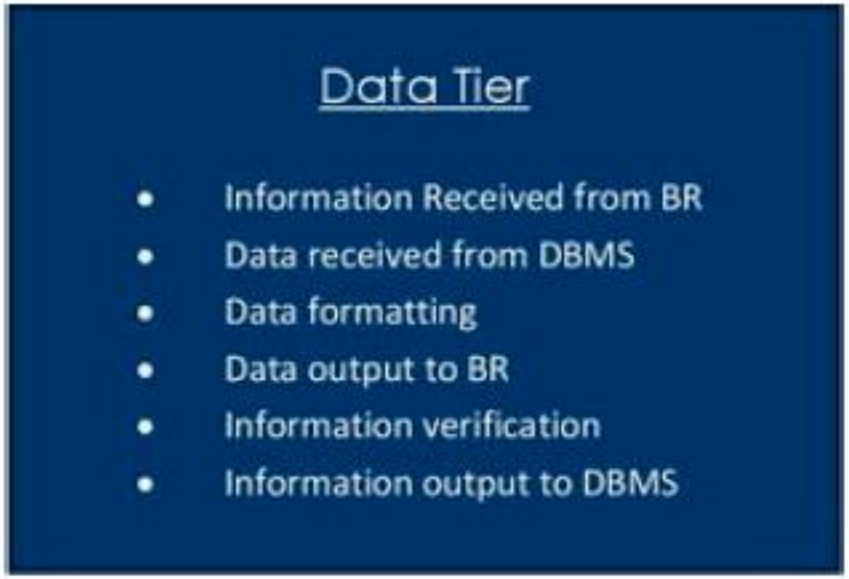
Business Rules Tier

- Data received from IT
- Data retrieved from DT
- Data verification
- Data formatting
- Information output to IT
- Information output to DT
- Process data into Information
- Service or Web Service



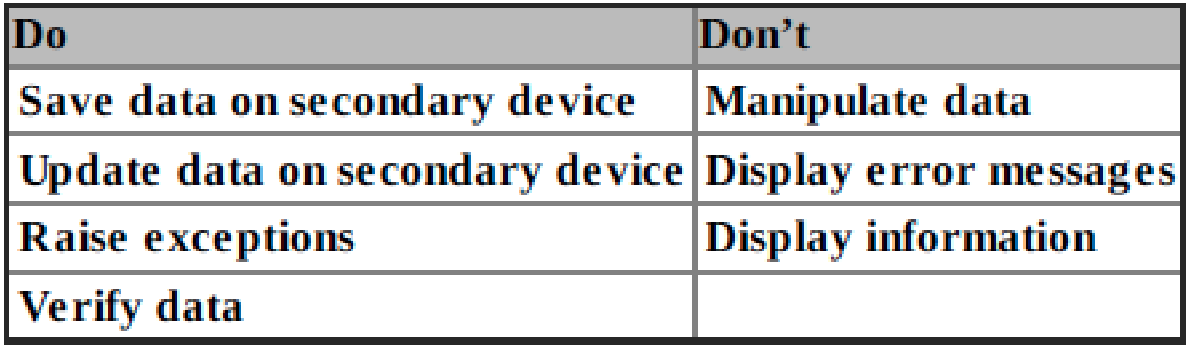
Do	Don’t
Manipulate data	Display information
Format data	Save data on secondary device
Store data in memory	Display error messages
Raise exceptions	
Verify data	

1. Client-server programming models



Data Tier

- Information Received from BR
- Data received from DBMS
- Data formatting
- Data output to BR
- Information verification
- Information output to DBMS



Do	Don’t
Save data on secondary device	Manipulate data
Update data on secondary device	Display error messages
Raise exceptions	Display information
Verify data	

1. Client-server programming models

- One advantage of breaking code into different tiers is the ability to reuse tiers with multiple applications.
  - For example, our search engine could use the same business rules tier and data tier for multiple devices while using a different interface (PC app or smart phone app).
- Distinct tiers can also be updated without affecting other tiers.

1. Client-server programming models

- Different programming model classifications:
  1. Depending on the size of the components:
     - Fat client (Thin server)
     - Fat server (Thin client)
  2. Depending on the division of functionalities between client and server:
     - 2 layers architecture: User + (Logic+Data)
     - 3 layers architecture

1. Client-server programming models

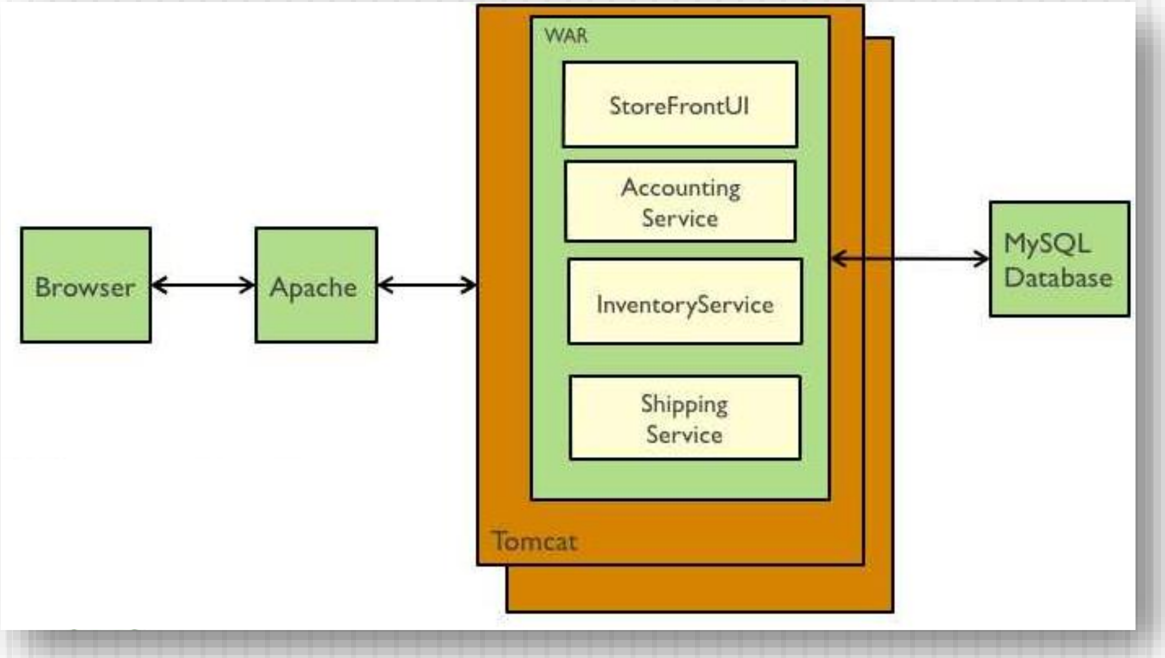
- **Monolithic vs Microservice Application Architectures:**

Monolithic vs microservices

8

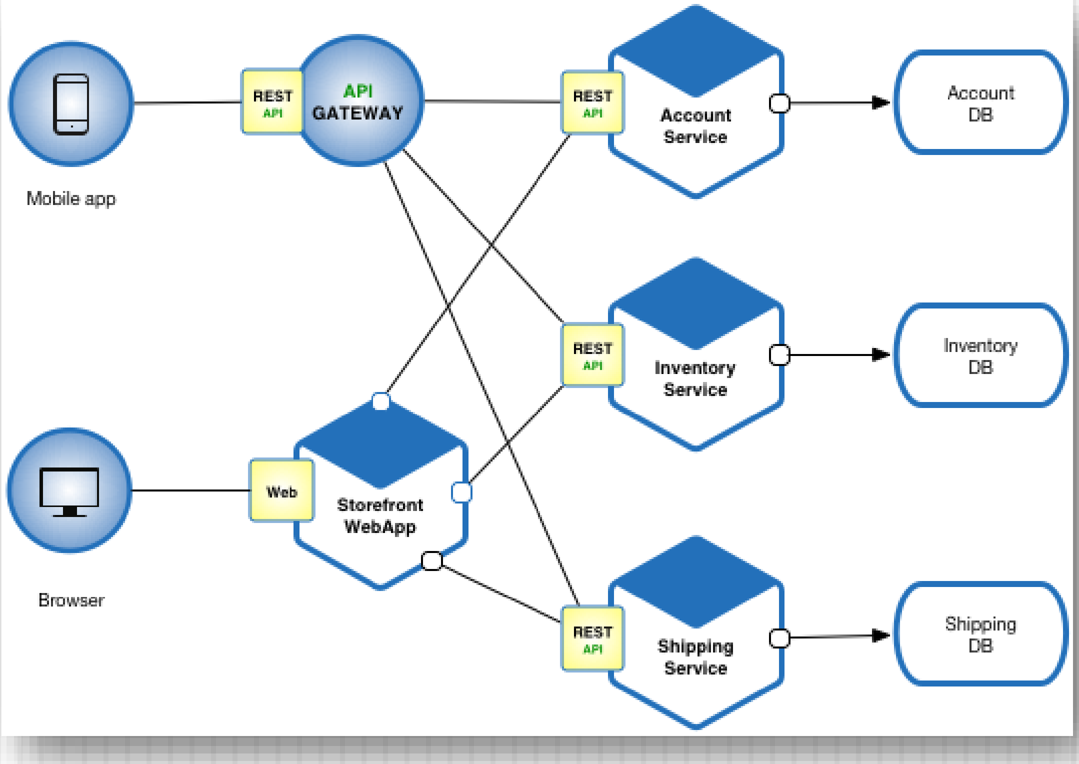
1. Client-server programming models

Monolithic architecture
Traditional server-side systems (PHP, JSP)
The entire system's function is based on a single application (deployed)
Faster to develop
Harder mantainance: domino effect



1. Client-server programming models

Microservices architecture
Moder server-side systems (Java Spring boot, Node.JS)
Each microservice can be developed, tested individually (and even deployed)
Ability to scale up
More straightforward to test microservices
No domino effect
Building an app with microservices can be a time-consuming process



2. Web page generation

1. Static web applications
2. Dynamic web applications
3. Interactive web applications

Do you know the difference between those 3?

3. Server side Programming languages

- Definition: code that runs on the server, using languages supported by the server.
- Scripting:
  - A high-level programming language that is interpreted by another program at runtime rather than compiled. Embedded (or not) within HTML, commonly are used to add functionality to a Web page.
- Web FWs:
  - PHP→Symfony, CakePHP, Laravel
  - Python→Django
  - JS→Angular (¿?) + Node.JS
  - C#→ASP.NET
  - Java→Spring / Spring boot

3. Server side Programming languages

A.1.1. Find information about those 5: Choose one of the languages above and describe its evolution, history, indicating if it has suffered any influence by other languages. Apart from that find interesting information about the Web FWs related. Try finding known web applications that use that language/FW.

4. Integration with Web servers

- On many occasions, the ability to respond to a request from a client depends on the capabilities that has the web server and modules or extensions you have installed.

How does a web server understand a formal request from a client?

4. Integration with Web servers

- Every Web server has an IP address and possibly a domain name.
- For example, if you enter the **URL** *http://www.mysite.org/index.html* in your browser, this sends a request to the Web server whose domain name is *mysite*. The server then fetches the page named *index.html* and sends it to your browser.

4. Integration with Web servers

- Two request methods (and subtypes):

1. **GET**: Requests data from a specified resource.

/test/demo_form.asp?name1=value1&name2=value2

2. **POST**: Submits data to be processed to a specified resource.

POST /test/demo_form.asp HTTP/1.1

Host: w3schools.com

name1=value1&name2=value2

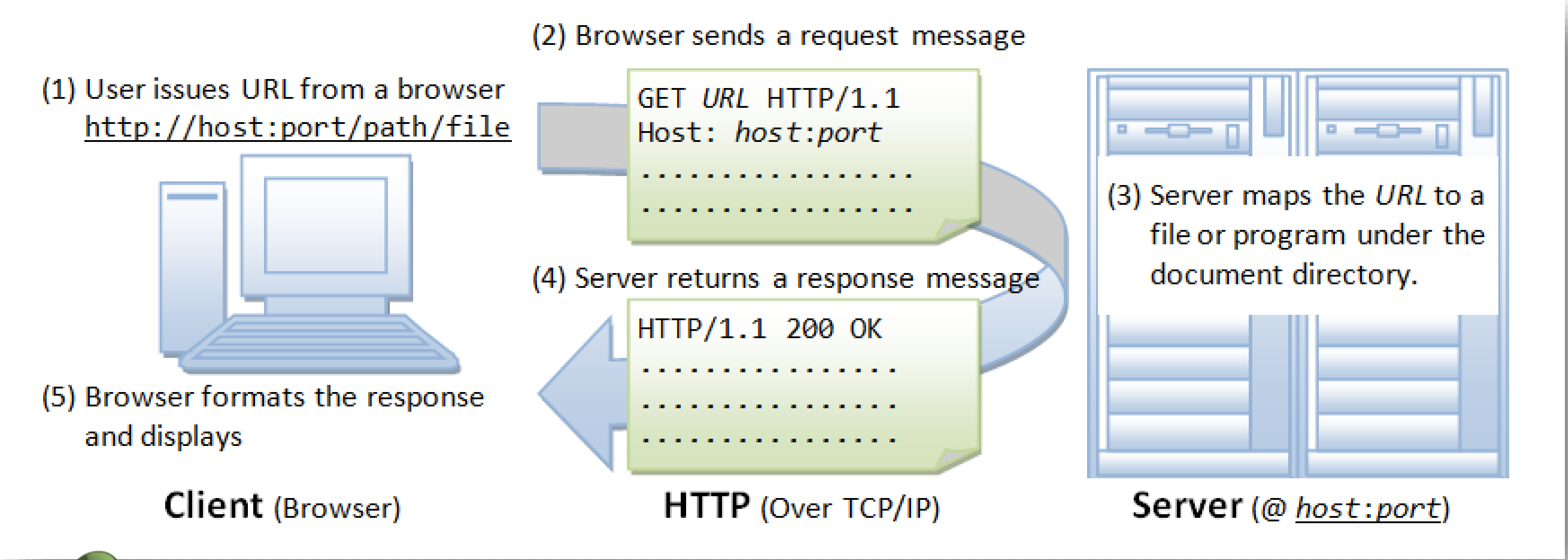
Do you know/remenber anything about pretty urls? Find an example of a pretty-url and a non-pretty on the internet.

4. Integration with Web servers

- Web server:
  - Program which main task is to serve data (as HTML documents encoded or JSON).
  - The exchange of data between a client and a web server is via a particular protocol, usually HTTP.

4. Integration with Web servers

Client-server communication sequence:



(1) User issues URL from a browser
http://host:port/path/file

(2) Browser sends a request message

```
GET URL HTTP/1.1
Host: host:port
.................
.................
```

(3) Server maps the *URL* to a file or program under the document directory.

(4) Server returns a response message

```
HTTP/1.1 200 OK
................
................
................
```

(5) Browser formats the response and displays

- Where would DNS take part?
- Which is the most common port used for Web Servers?

4. Integration with Web servers

- Whenever you enter a URL in the address box of the browser, the browser translates the URL into a request message according to the specified protocol; and sends the request message to the server.
- When this request message reaches the server, the server can take either one of these actions:
  1. The server interprets the request received, maps the request into a file under the server's document directory, and <u>returns the file</u> requested to the client.
  2. The server interprets the request received, maps the request into a program kept in the server, <u>executes the program</u>, and returns the output of the program to the client.
  3. The request <u>cannot be satisfied</u>, the server returns an error message.

4. Integration with Web servers

- Examples of Web servers:

1. *Apache Server: HTTP server designed to be used across multiple platforms and operating systems.*
2. *Ngnix:* It is a HTTP server which has gained market share in recent years (from <1% in 2007 to almost 10% in late 2011).
3. *Microsoft IIS.*
4. *LiteSpeed Web Server:* It is a web server with high performance and high scalability.
5. *Lighttp*: specialized in environments that require quick responses.
6. *GlassFish (before Sun Java System Web Server):* This is a web server for high performance, massively scalable and secure, that offers dynamic and static content.

4. Integration with Web servers

Check some statitics: W3Techs

5. Programming tools

- The development process does not have to be performed on the same computer on which the application will be finally deployed.
- Usually there are more than one environment.

5. Programming tools

- Tools:
  - *Web browsers*: Important to take into account, version and type.
  - *Document editors*: Arachnophilia, Notepad++, 🙂 UltraEdit 🙁
  - *IDEs*: Eclipse, Intellij, Dreamweaver, Aptana, Sublime,Visual Studio Code, Atom
  - *Image processing tools*
  - *Creating and managing DB tools*