Java EE Introduction

Sandeep Kulange, SunBeam

PowerPoint

What is Java EE?

- Java EE / JEE is also called as Enterprise Java / Advanced Java.
- Java EE is a specification defined by Sun/Oracle. It is super set of Java SE.
- It is a collection of interfaces and abstract classes.
- Anybody can implement Java EE specification but generally web server and application server vendors implement it.
- Following are the Java EE Specifications:
    - Servlet, JSP(Java Server Pages)
    - WebSocket, JSF(Java Server Faces), EJB(Enterprise Java Bean),JPA(Java Persistence API), JTA(Java Transaction API), JMS(Java Message Service)

Web Server

- Dedicated server software which processes incoming network requests over the HTTP protocol.
- It is responsible for taking care of presentation logic. In other words, it is designed to serve HTTP content.
- Web Server = Web Container + Extra Services
- Web Container = Servlet container + JSP Container
- Extra Services = Security, Connection pooling, JNDI etc
- Example of web server:
  - 1. Tomcat(Apache Foundation)
  - 2. Mongoose(Cesanta Software)

Application Server

- Dedicated server software which processes incoming network requests over the HTTP, RMI/RPC etc. protocol.
- It is responsible for taking care of presentation logic as well as business logic.
- App server = Web Container + EJB Container + Extra services
- Extra Services:
    1. Connection Pooling
    2. Object Pooling
    3. Transaction Support
    4. Messaging Services etc.

Application Servers

1. TomEE(Apache Foundation). TomEE = Tomcat + Java EE.
2. Geronimo(Apache Foundation)
3. GlassFish(Oracle)
4. WebLogic(Oracle)
5. WebSphere(IBM)
6. WildFly-Formerly JBoss(Red Hat)
7. Enhydra(Lutris Technologies)

Why Java EE?

1. It supports different types of clients:

2. It gives us JEE server independence:

3. Ready made implementation of primary services:

    - For example:

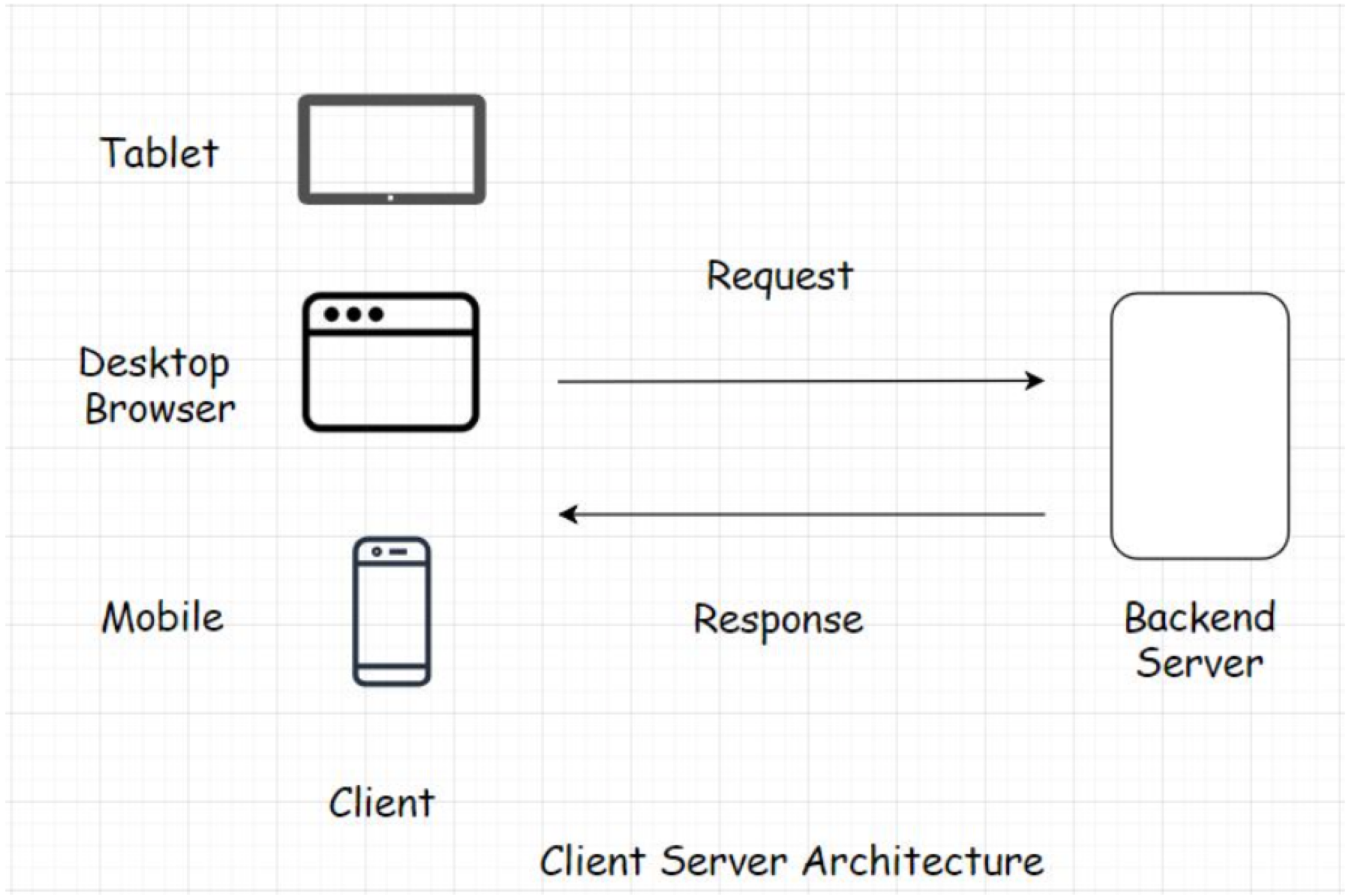
        - security

        - connection pooling

        - email

Java EE developer doesn’t have to worry about primary services. He/she can concentrate on actual business logic.

Client Server Architecture

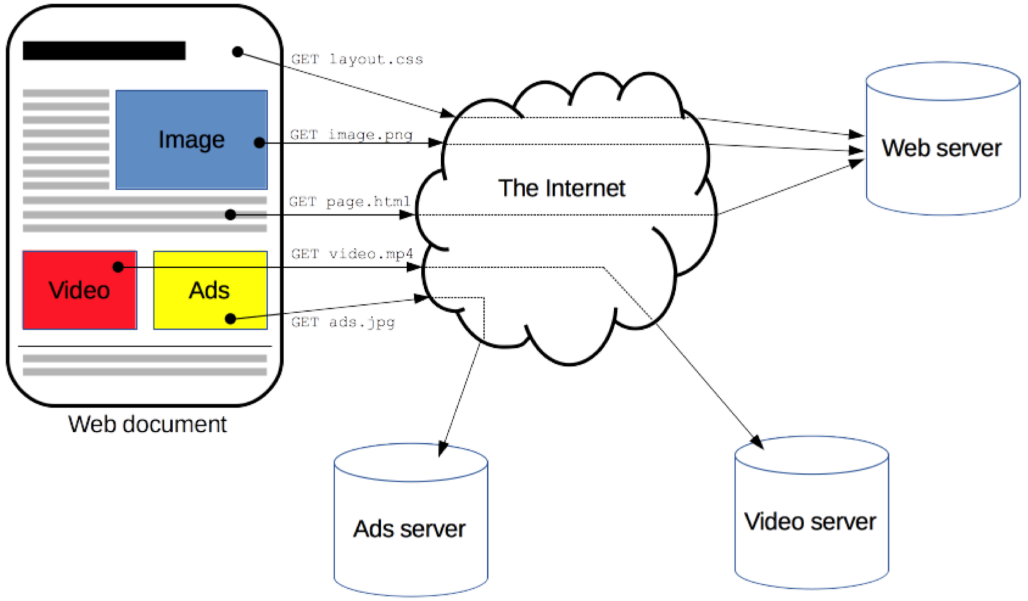


Web Basics

- An application that we install/deploy on server is called **web application.**
- A program which consumes services is called **client.**
- Types of client:
    1. Thin client
    2. Thick client
    3. Smart client
- A program which provide services to its client is called **server.**
- Types of server:
    1. Web Server
    2. Application Server
- A machine on which we install client program is called **client machine** and a machine on which we install server program is called **server machine.**

Hypertext Transfer Protocol [HTTP]

- It is an application layer protocol that is sent over TCP.
- It is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser.
- **HTTP** is a protocol which allows the fetching of resources, such as HTML documents.
- Clients and servers communicate by exchanging individual messages. The messages sent by the client, usually a Web browser, are called ***requests*** and the messages sent by the server as an answer are called ***responses***.

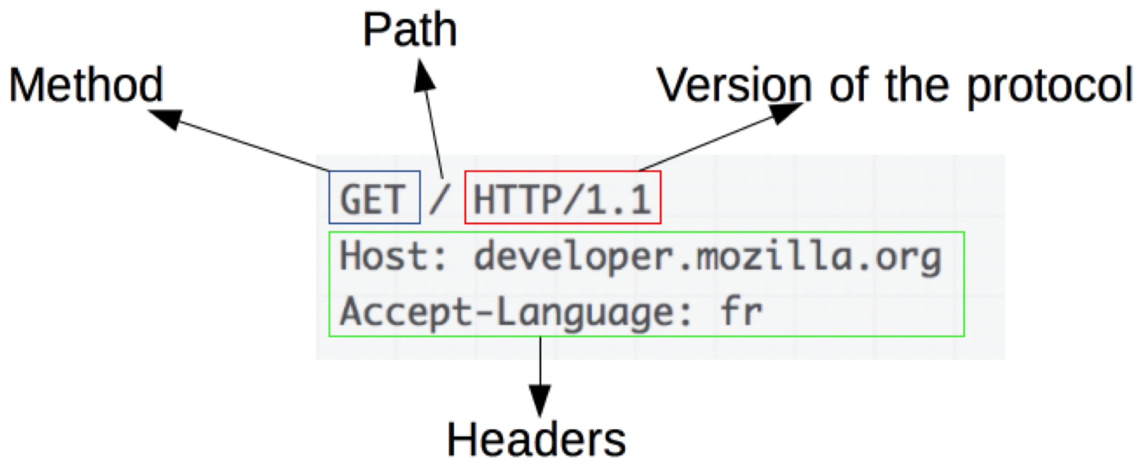


Basic Aspects of HTTP

- HTTP is **simple**
    - HTTP is generally designed to be simple and human readable.
    - HTTP messages can be read and understood by humans, providing easier testing for developers, and reduced complexity for newcomers.
- HTTP is **stateless**, but not **sessionless**
    - HTTP is stateless: there is no link between two requests being successively carried out on the same connection.
    - The core of HTTP itself is stateless, HTTP cookies allow the use of stateful sessions.

HTTP Request

- Using HTTP protocol client sends request to the server is called HTTP request.
- Requests consists of the following elements:
    1. HTTP method
    2. The path of resource to be fetch
    3. The version of HTTP protocol
    4. Optional headers that convey additional information for the servers.
    5. A body, for some methods like POST.



```
GET / HTTP/1.1
Host: developer.mozilla.org
Accept-Language: fr
```

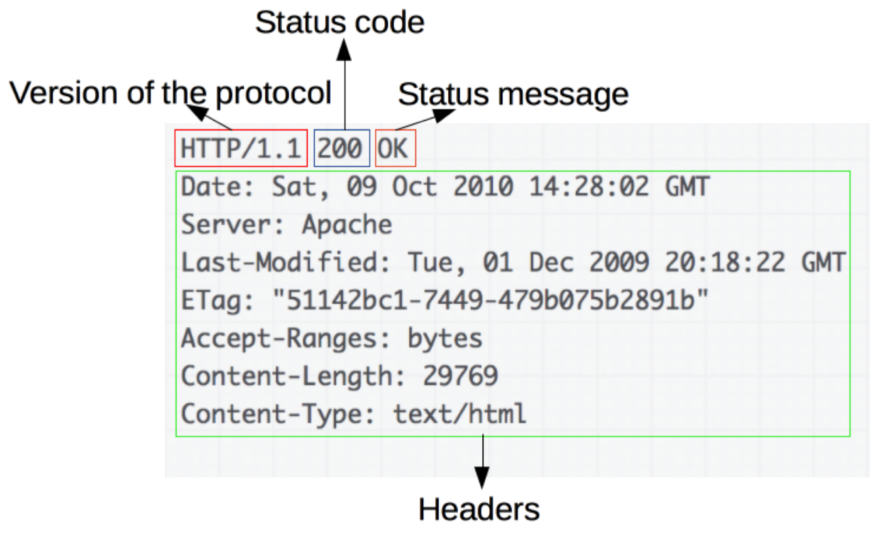
HTTP Request Methods

1. **GET**
   - Used to retrieve information from the given server using given URI.
2. **POST**
   - Used to send data the server. For example, customer information.
3. **PUT**
4. **DELETE**
5. **CONNECT**
6. **HEAD**
7. **TRACE**
8. **OPTIONS**
9. **PATCH**

Reference: https://developer.mozilla.org/en-US/docs/Web

HTTP Response

- Using HTTP protocol server sends response to the client is called HTTP response.
- Responses consist of the following elements:
    1. The version of the HTTP protocol they follow.
    2. A status code, indicating if the request was successful, or not, and why.
    3. A status message, a non-authoritative short description of the status code.
    4. HTTP headers, like those for requests.
    5. Optionally, a body containing the fetched resource.



```
HTTP/1.1 200 OK
Date: Sat, 09 Oct 2010 14:28:02 GMT
Server: Apache
Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
ETag: "51142bc1-7449-479b075b2891b"
Accept-Ranges: bytes
Content-Length: 29769
Content-Type: text/html
```

HTTP Status Code

HTTP response status codes indicate whether a specific HTTP request has been successfully completed. Responses are grouped in five classes:

1. Informational responses (100–199),
2. Successful responses (200–299),
3. Redirects (300–399),
4. Client errors (400–499),
5. and Server errors (500–599).

Reference : https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Content-Type / MIME Type

- **Multipurpose Internet Mail Extensions or MIME type** is a standard that indicates the nature and format of document.
- The simplest MIME type consists of a ***type* and a *subtype***; these are each strings which, when concatenated with a slash (/) between them, comprise a MIME type. **No whitespace is allowed in a MIME type.**
- **Important MIME types for Web developers**
  1. text/plain - default for textual files
  2. text/html - All HTML content should be served with this type.
  3. text/css - CSS files used to style a Web page **must** be sent with text/css.
  4. text/javascript - JavaScript files should always be served using it.
  5. application/octet-stream - This is the default for binary files.
- Reference : https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types

HTTP Flow

- When a client wants to communicate with a server, it performs the following steps:

    1. Open a TCP connection

        - The TCP connection is used to send a request, or several, and receive an answer. The client may open a new connection, reuse an existing connection, or open several TCP connections to the servers.

    2. Send an HTTP message

    3. Read the response sent by the server

    4. Close or reuse the connection for further requests.

HTTP Flow

- When a client wants to communicate with a server, it performs the following steps:

  1. Open a TCP connection

     - The TCP connection is used to send a request, or several, and receive an answer. The client may open a new connection, reuse an existing connection, or open several TCP connections to the servers.

  2. Send an HTTP message

  3. Read the response sent by the server

  4. Close or reuse the connection for further requests.

Thank you