INFO 5100
Application Engineering and Development

Week 8

Agenda

- Java Dependency Management Tool
- J2EE
- RESTful API

Dependency Management

- Referring/Using 3rd party packages are unavoidable in Software Development
- Out-of-data or latest issue
- Dependencies for Dependencies

- All programming languages have tools to perform dependency management

Dependency Management

- Maven: https://maven.apache.org/
- Uses a XML file called pom.xml to manage the configuration - https://maven.apache.org/pom.html
- Supported by all major IDEs
- Dependencies are downloaded and installed at ~/.m2 folder

pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>edu.neu.app</groupId>
    <artifactId>user-management</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>war</packaging>

    <dependencies>
        <dependency>
            <groupId>com.sun.jersey</groupId>
            <artifactId>jersey-server</artifactId>
            <version>1.8</version>
        </dependency>
    </dependencies>
</project>
```

pom.xml

```xml
.. continued
    <build>
        <finalName>user-management</finalName>
        <plugins>
            <plugin>
                <artifactId>maven-compiler-plugin</artifactId>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

J2EE

- Java Enterprise Edition
- APIs to facilitate development of enterprise applications
  - Servlet - how to manage HTTP requests, etc
  - JAX-RS
  - Java Persistence API
  - Java Transaction API
  - etc

Servlet HelloWorld

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
// Extend HttpServlet class
public class HelloWorld extends HttpServlet {
    private String message;
    public void init() throws ServletException {
        message = "Hello World";
    }
    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<h1>" + message + "</h1>");
    }
    public void destroy() {}
}
```

web.xml

```xml
<?xml version = "1.0" encoding = "UTF-8"?>
<web-app xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
        xmlns = "http://java.sun.com/xml/ns/javaee"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
  http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
        id = "WebApp_ID" version = "3.0">
  <display-name>User Management</display-name>
  <servlet>
      <servlet-name>HelloWorld</servlet-name>
      <servlet-class>HelloWorld</servlet-class>
  </servlet>

  <servlet-mapping>
      <servlet-name>HelloWorld</servlet-name>
      <url-pattern>/HelloWorld</url-pattern>
  </servlet-mapping>
</web-app>
```

Tomcat

- Tomcat: http://tomcat.apache.org/
- Tomcat is an Application Container that supports J2EE Servlet API
- Alternatives
  - GlassFish
  - Jetty

Tomcat

| bin	Includes binaries to run the server
lib	Library dependencies for tomcat
work	Working directory with apps
conf	Configuration files for tomcat
logs	Operational logs for tomcat
webapps	Compiled/packaged web app

RESTful API

- Systems/Services have been living in silos
- Integration was hard because of different coding/API styles
- API could be hard to use, for example, running at lower level of network
- API could be hard to use, for example, integrate with SDKs

- RESTful API, a concept of representing/sharing data across through network using HTTP protocol
- https://restfulapi.net/
- Not a technology, but a convention, and there are many implementations

OSI (Open Source Interconnection) 7 Layer Model

| Layer	Application/Example		Central Device/ Protocols		DOD4 Model
Application (7) Serves as the window for users and application processes to access the network services.	**End User layer** Program that opens what was sent or creates what is to be sent Resource sharing · Remote file access · Remote printer access · Directory services · Network management		**User Applications** SMTP	**G** **A** **T** **E** **W** **A** **Y** Can be used on all layers	Process
Presentation (6) Formats the data to be presented to the Application layer. It can be viewed as the "Translator" for the network.	**Syntax layer** encrypt & decrypt (if needed) Character code translation · Data conversion · Data compression · Data encryption · **Character Set Translation**		JPEG/ASCII EBDIC/TIFF/GIF PICT		
Session (5) Allows session establishment between processes running on different stations.	**Synch & send to ports** (logical ports) Session establishment, maintenance and termination · Session support - perform security, name recognition, logging, etc.		**Logical Ports** RPC/SQL/NFS NetBIOS names		
Transport (4) Ensures that messages are delivered error-free, in sequence, and with no losses or duplications.	**TCP** Host to Host, Flow Control Message segmentation · Message acknowledgement · Message traffic control · Session multiplexing	P A C K E T F I L T E R I N G	TCP/SPX/UDP		Host to Host
Network (3) Controls the operations of the subnet, deciding which physical path the data takes.	**Packets** ("letter", contains IP address) Routing · Subnet traffic control · Frame fragmentation · Logical-physical address mapping · Subnet usage accounting		**Routers** IP/IPX/ICMP		Internet
Data Link (2) Provides error-free transfer of data frames from one node to another over the Physical layer.	**Frames** ("envelopes", contains MAC address) [NIC card —— Switch —— NIC card] (end to end) Establishes & terminates the logical link between nodes · Frame traffic control · Frame sequencing · Frame acknowledgment · Frame delimiting · Frame error checking · Media access control		**Switch Bridge WAP** PPP/SLIP	Land Based Layers	Network
Physical (1) Concerned with the transmission and reception of the unstructured raw bit stream over the physical medium.	**Physical structure** Cables, hubs, etc. Data Encoding · Physical medium attachment · Transmission technique - Baseband or Broadband · Physical medium transmission Bits & Volts		Hub		

OSI 7 Layer Model
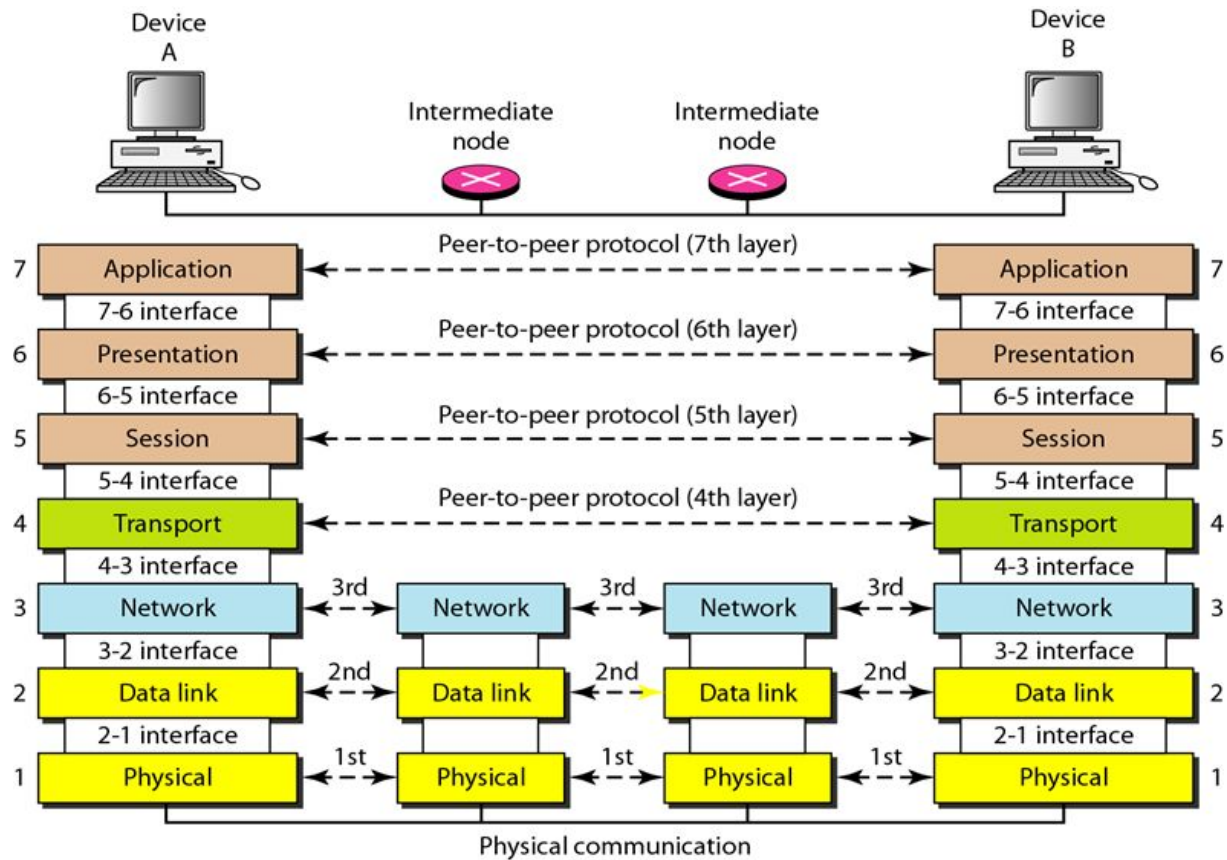
OSI 7 Layer Model

HTTP Protocol

- Layer 7 protocol
- https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

HTTP Request
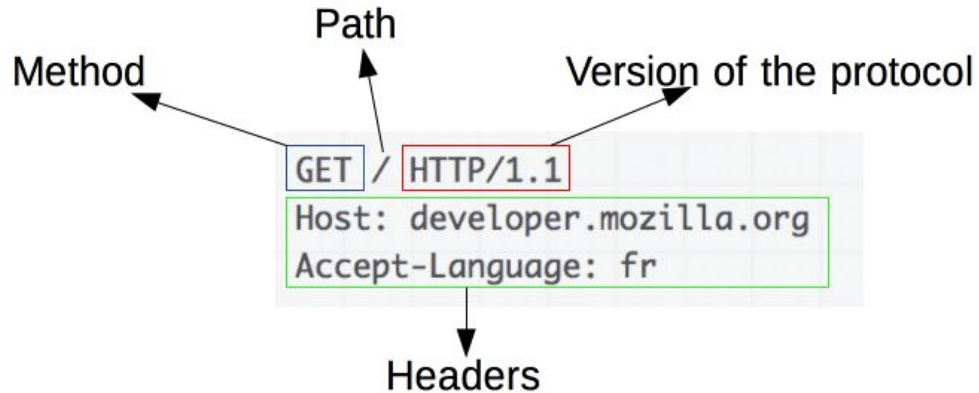
HTTP Response
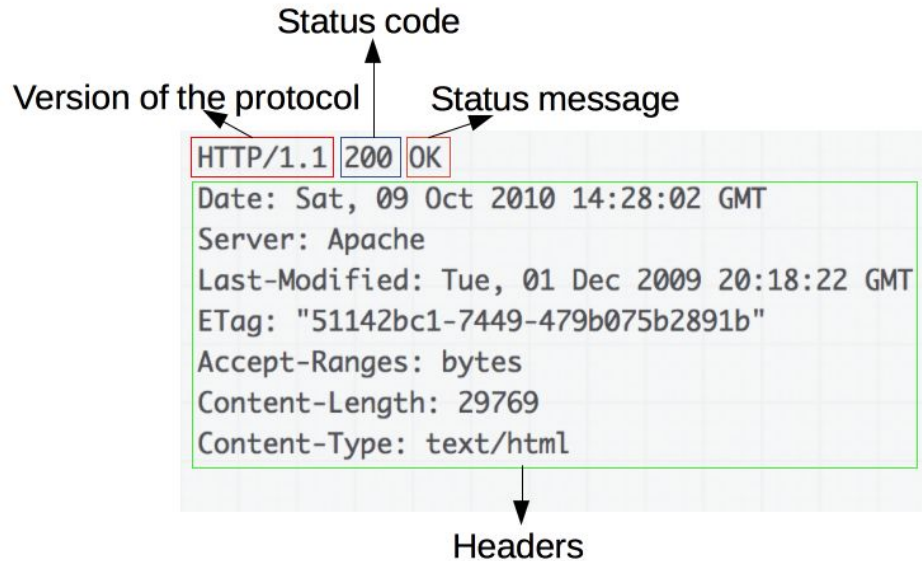
RESTful API

- A semantics combining HTTP Method/Resource URL/Response

GET /user-management/users	List out all users
POST /user-management/users with body	Create new users
GET /user-management/users/1	Get detail information for user with ID 1
DEL /user-management/users/1	Delete user with ID 1
GET /user-management/users?gender=f	List users based on condition

JAX-RS

- Standard API defined in J2EE
- Multiple implementations
  - Jersey
  - JBoss