Java 8 New Features

* Functional Interface – An Interface with only one abstract method.
* Lambda – Anonymous function/method. One-line/single line method.
* Streams – Improvements in collection API. Used to handle group of objects.
* Method Reference. (::) – Static /Instance & Constructor

Marker Interface – An Interface with no method at all. (Serializable)

In Java 8 and above versions, adding static and default concrete methods in interface is allowed.

@FunctionalInterface

Java.util.stream

Java.util.function – All the interface defined in this package are all Functional Interface

Predicate, Consumer, Collectors, Arrays, forEach()

Micro-Service

SOA – Service Oriented Architecture

MSA – MicroService Architecture

Monolith Application VS Micro-Service Application

Design Principles used in Micro-Service Implementation.

Adv & Dis-adv of Micro-Service.

STS – <https://spring.io/tools> - (EclipseIDE based STS/VS Code/ Theia)

STS – Spring Tools Suite

“java -jar sts4.jar” -- This command should be executed from the location where your sts4.jar is present.

SOA – Service Oriented Architecture

Service – Web Service (Any Service with the help of internet)

Web Service – Device to Device Communication using Internet (http/https)

http = HyperText Transfer Protocol

Protocol = Set of Rules & Regulations

SMTP = Simple Mail Transfer Protocol

FTP = File Transfer Protocol

Hyper Text = It has additional properties than normal text. It can link to other documents/ different location in the same document/url/email address.

HTML = HyperText Markup Language – Tag based Language.

Web Service – Web API (Another name of Web Service)

* URL – Uniform Resource Locator
* URI – Uniform Resource Identifier

URL = <http://www.google.com/index.html> = Default communication port number for http is 80

Client & Server Technology = Port is also associated.

Web Server – Tomcat - 8080

Database Server – MySQL – 3306, oracle = 8080/1521, postgres = 5432,

Angular – 4200

React – 3000

URI or End points

<https://www.google.com/api/v1/employees> - the output will be in XML/JSON format

http – it has many methods

XML – eXtensible Markup Language – It’s both case & space sensitive.

JSON – JavaScript Object Notation

Employee

Id (int)

Name(string)

Email (String)

XML/JSON – is used to represent data in platform/architecture/programming language independently.

XML representation

Employees.xml

<employees>

<employee>

<id>100</id>

<name>ABC</name>

<email>abc@gmail.com</email>

</employee>

<employee>

<id>101</id>

<name>XYZ</name>

<email>xyz@gmail.com</email>

</employee>

<employee>

<id>102</id>

<name>LMN</name>

<email>lmn@gmail.com</email>

</employee>

</employees>

Employees.json

Employees= [

{

“id”:100,

“name” : “ABC”,

“email”:”abc@gmail.com”

},

{

“id”:101,

“name” : “XYZ”,

“email”:”xyz@gmail.com”

},

{

“id”:102,

“name” : “LMN”,

“email”:”lmn@gmail.com”

}

];

Client & Server Technology = Request Object & Response Object

Google.com/search.jsp?q=microservice&type=ppt

String question = Request.getParameter(“q”);

http get method = default http method – It’s faster, append the URL with client side data. Data transferring will be visible to everyone. So sending sensitive data (username & password, OTP, SSN, PAN, UPI id) is not recommended in this method.

http post method = Data will be added to the Request Body

<form action=”search.jsp” method=”GET/POST”>

</form>

|  |  |  |  |
| --- | --- | --- | --- |
| Sl No | Http Method | Location of Data | Usage |
| 1 | Get | Data will be added to the URL and will be visible to all | For sending normal/ non-confidential data from client to server. It’s mainly used for reading some data from server |
| 2 | Post | Data will be added to the Request Object Body (Data will not be visible to anyone) | For sending Confidential and secured data. It’s mainly used for writing/adding some data in the server side. |
| 3 | Put | Data will be added to URL & Request Body | For modifying existing items in server side |
| 4 | Delete | Data will be added to URL or Request Body | For deleting existing item in the server side |

Type of Web Services

1. SOAP based Web Service
2. REST based Web Service

SOAP – Simple Object Access Protocol

By default soap gives us XML response.

SOAP uses WSDL – Web Service Definition/Description Language (XML) [ axis ]

REST – Representational State Transfer – It re-use the http methods.

API Development / Web Service Development / SOA = REST based web service (XML/JSON Response)

Web Service = Device to Device Communication using http

ABC Bank = Debit card

1. ABC bank ATM
2. XYZ bank ATM

Web Service can be developed using any Programming Lang.

Easiest way of creating REST ful web service is – Using Spring Boot & Java.

Steps in Creating Web Service

1. Open <https://start.spring.io> (Spring Initializr URL) or Open STS (File🡪 New🡪 Spring Starter Project)
2. Added all the details (Project type[maven], package type[jar], lang[Java], version[1.8], sb\_version[2.7.5], dependency[Spring Web]
3. Run as 🡪 Spring Boot App (modified application.properties file [server.port=8085]
4. Opened browser to access localhost:8085 – Got WhilteLabel Error page

Web Service -- Calling a Method written in any programming lang with the help of URI.

@RequestMapping – is a generic annotation – It can be used for any http method. Default is get method.

@GetMapping

@PostMapping

@PutMapping

@DeleteMapping

Product

**private** **int** id;

**private** String name;

**private** String unitPrice;

**private** **int** stock;

Order

Id (int)

CartId(FK)

List<Products> (ArrayList of Products)

TotalAmount (float)

DeliveryAddress (String)

PaymentMethod (String)

User

Id

Username

Password

Email

Mobile

Role

Id

Name

Cart

Id

List<Product>

Spring Boot Application

Modularity - Package Structure

<base\_package>

<base\_package>.entity/model/bean

<base\_package>.controller

<base\_package>.repo

<base\_package>.service

<base\_package>.exception

<base\_package>.util

<base\_package>.configure

Entity Bean 🡪 Entity Controller 🡪 Service 🡪 Repo (DAO – Data Access Object)

Data Read operation [From DB🡪 Application 🡪 Web Browser (JSON)]

Data Update Operation (Write/Insert, Update & Delete) [Web Browser(UI) [JSON &/ ID] 🡪 Application 🡪 Database]

UI – User Interface

Entity Bean 🡪 Repository (DAO) – findAll(), findById(), save(), delete() 🡪Database

Endpoints (Controller) 🡪 Service 🡪 repo.findAll()