# **Brillio Wave-2 JavaFSD Learning outcome (07/09/2020) FN**

# **1. Introduction to Webservice**

# **2. Web Service Architecture**

# **3. Web Services VS Web Applications**

# **4. Types of Web Services**

# **5. What is REST architecture?**

# **6. ResponseEntity and ResponseBody**

# **Maven Project**

# 

# **com.sr.model**

# **Employee.java**

# **com.sr.config**

# **ApplicationContextConfig.java**

# **WebAppInitializer.java**

# **com.sr.respository**

# **EmployeeDao.java ----->Interface**

# **EmployeeDaoImpl.java**

# **com.sr.service**

# **EmployeeService.java ---Interface**

# **EmployeeServiceImpl.java**

# **com.sr.controller**

# **EmployeeController.java**

# **userrequest -------> Applicationcontextconfig+WebAppIntilizer--->controller-- ---->service------>repository----> Database**

# **What is Web Service**

A Web Service is can be defined by following ways:

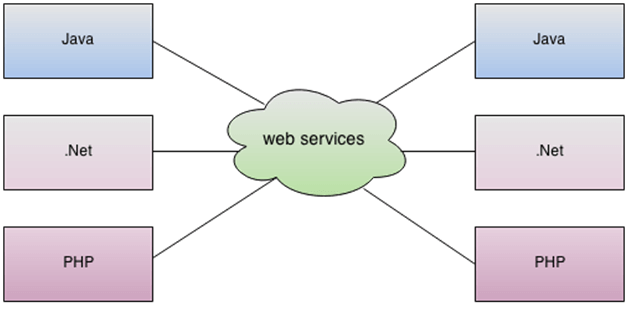
It is a client-server application or application component for communication.

The method of communication between two devices over the network.

It is a software system for the interoperable machine to machine communication.

It is a collection of standards or protocols for exchanging information between two devices or application.

# **Web Service Architecture**



**Differences between Web Services and Web Applications**

|  |  |  |
| --- | --- | --- |
| # | Web Service | Web Application |
| 1. | A web service doesn't have a user interface | A Web Application has a user interface or GUI |
| 2. | Web services are meant for other applications to be interacted with over internet | Websites are meant for use by humans |
| 3. | Web services are platform independent as they use open protocols | Websites are cross platform as they require tweaking to operate on different browsers, operating systems etc. |
| 4. | Web services are accessed by HTTP methods - GET, POST, PUT, DELETE etc | Websites are accessed by using their GUI components - buttons, text boxes, forms etc. |

## **Types of Web Services**

There are mainly two types of web services.

1. SOAP web services.
2. RESTful web services.

## **Web Service Features**

### XML-Based

### Loosely Coupled

### Coarse-Grained

### Ability to be Synchronous or Asynchronous

### Supports Remote Procedure Calls (RPCs)

### Supports Document Exchange

**Web Service Components**

There are three major web service components.

1. SOAP
2. WSDL
3. UDDI

## SOAP

SOAP is an acronym for Simple Object Access Protocol.

SOAP is a XML-based potocol for accessing web services.

SOAP is a W3C recommendation for communication between applications.

SOAP is XML based, so it is platform independent and language independent. In other words, it can be used with Java, .Net or PHP language on any platform.

## WSDL

WSDL is an acronym for Web Services Description Language.

WSDL is a xml document containing information about web services such as method name, method parameter and how to access it.

WSDL is a part of UDDI. It acts as a interface between web service applications.

WSDL is pronounced as wiz-dull.

## UDDI

UDDI is an acronym for Universal Description, Discovery and Integration.

UDDI is a XML based framework for describing, discovering and integrating web services.

UDDI is a directory of web service interfaces described by WSDL, containing information about web services.

## **What is REST architecture?**

REST stands for REpresentational State Transfer.

REST is web standards based architecture and uses HTTP Protocol.

It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods.

REST was first introduced by Roy Fielding in 2000.

In REST architecture, a REST Server simply provides access to resources and REST client accesses and modifies the resources. Here each resource is identified by URIs/ global IDs. REST uses various representation to represent a resource like text, JSON, XML. JSON is the most popular one.

### **HTTP methods**

Following four HTTP methods are commonly used in REST based architecture.

**GET** − Provides a read only access to a resource.

**POST** − Used to create a new resource.

**DELETE** − Used to remove a resource.

**PUT** − Used to update a existing resource or create a new resource.

## **ResponseEntity**

## represents the whole HTTP response: status code, headers, and body.

## we can use it to fully configure the HTTP response.

## we have to return it from the endpoint; Spring takes care of the rest.

## **ResponseBody**

## puts the return value into the body of the response,