9 : Web Services

IT 4206 – Enterprise Application Development

Level II - Semester 4

Overview

- Identify how to describe web services concepts and types.

Intended Learning Outcomes

- At the end of this lesson, you will be able to describe web services concepts and types.

List of sub topics

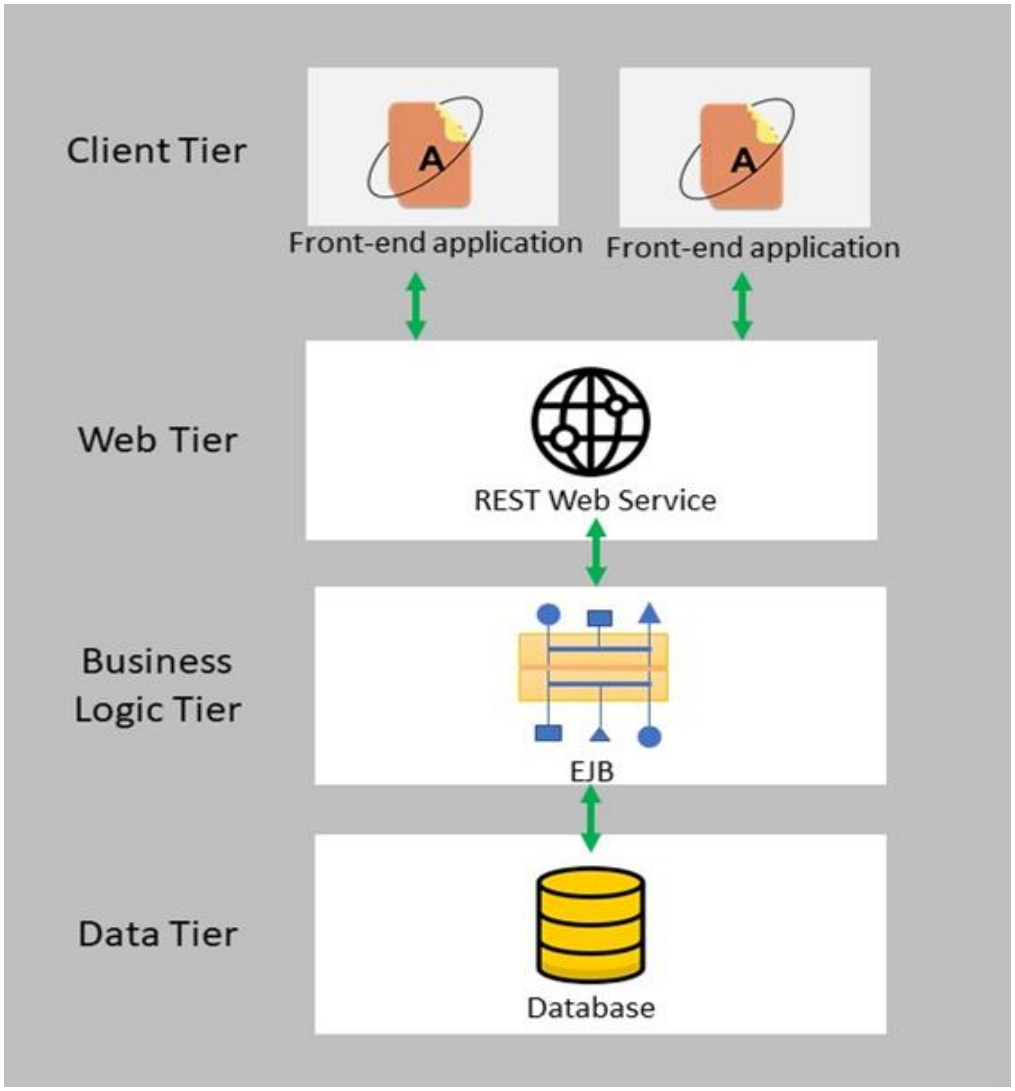
1.1 Web Services

1.2 RESTful web services

1.1 Web Services

- Web services are client and server applications that communicate over the World Wide Web's (WWW) HyperText Transfer Protocol (HTTP).
- Web services expose standardized communication for interoperability between application components over HTTP.
- Web services can be combined in a loosely coupled way to achieve complex operations.

Web Service Application Architecture



Types of Web Services

- There are two implementations of web services that mainly discussed:
  - JAX-RS RESTful Web Services
  - JAX-WS Web Services
- Both of these implementations provide the same benefits of using web services, such as loose coupling and standardized protocols.
- JAX-WS and JAX-RS differ in a number of important ways that must be carefully considered before deciding which approach to use.

1.2 Restful web services

- RESTful web services are built to work best on the Web.
- Representational State Transfer (REST) is an architectural style that specifies constraints, such as the uniform interface, performance, scalability, and modifiability, that enable services to work best on the Web.

- The following principles encourage RESTful applications to be simple, lightweight, and fast:
  - **Resource identification through URI:** A RESTful web service exposes a set of resources that identify the targets of the interaction with its clients.
  - **Uniform interface:** Resources are manipulated using a fixed set of four create, read, update, delete operations: PUT, GET, POST, and DELETE.
  - **Self-descriptive messages:** Resources are decoupled from their representation so that their content can be accessed in a variety of formats, such as HTML, XML, plain text, PDF, JPEG, JSON, and others.
  - **Stateful interactions through hyperlinks:** Every interaction with a resource is stateless; that is, request messages are self-contained.

Creating Restful web services with JAX-RS

- JAX-RS is the Java API used to create RESTful web services.
- A JAX-RS RESTful web service consists of one or more classes utilizing the JAX-RS annotations to create a web service.

Annotations

- Developers decorate Java programming language class files with JAX-RS annotations to define resources and the actions that can be performed on those resources.
- JAX-RS annotations are runtime annotations; therefore, runtime reflection will generate the helper classes and artifacts for the resource.
- Following slide shows annotations of JAX-RS.

Annotations

- @Path
- @GET
- @POST
- @PUT
- @DELETE
- @HEAD
- @PathParam
- @QueryParam
- @Consumes
- @Produces
- @Provider

Example of a root resource class that uses JAX-RS annotations:

```
import javax.ws.rs.GET;
import javax.ws.rs.Produces;
import javax.ws.rs.Path;

@Path("/helloworld")
public class HelloWorldResource {

@GET
@Produces("text/plain")
        public String getClichedMessage() {
                return "Hello World";
         }
}
```

The @Path Annotation and URI Path Templates

- The @Path annotation identifies the URI path template to which the resource responds and is specified at the class or method level of a resource.
- The @Path annotation's value is a partial URI path template relative to the base URI of the server on which the resource is deployed, the context root of the application, and the URL pattern to which the JAX-RS runtime responds.

- URI path templates are URIs with variables embedded within the URI syntax.
- These variables are substituted at runtime in order for a resource to respond to a request based on the substituted URI.

Example

- @Path("/users/{username}")
- In this kind of example, a user is prompted to type his or her name, and then a JAX-RS web service configured to respond to requests to this URI path template responds.
- For example, if the user types the user name “Kasun,” the web service responds to the following URL:
  - http://example.com/users/Kasun

Customizing Requests and responses

- One of the strengths of JAX-RS is the ability to customize both the MIME type of the request and response.
- The @Produces or @Consumes annotations can be modified to enforce XML as the response and request type, respectively.

- The annotation @Produces defines the MIME media type for the response returned by the service.
- The annotation @Consumes defines the MIME media type for the request required by the service.

HTTP Methods

- The HTTP protocol defines several methods through which the protocol enacts different actions.
- The client is responsible for specifying the type of request in addition to the path of the request.
- The following is a list of the methods:
  - GET: The GET method retrieves data.
  - POST: The POST method creates a new entity.
  - DELETE: The DELETE method removes an entity.
  - PUT: The PUT method updates an entity.

Injecting parameters from the URI

- In many instances, clients using RESTful web services need to request specific information from a service.
- This is accomplished either by providing parameters in the URI, either as a path parameter or a query parameter.

- To use a path parameter on a JAX-RS method, annotate it with the @PathParam annotation. This annotation is typically used when the client is requesting a specific resource, such as requesting a user's data.

```
@GET
 @Path("{id}")
 public Person getPerson(@PathParam("id") Long id) {
 return entityManager.find(Person.class, id);
 }
```

- To use a query parameter on a JAX-RS method, annotate it with the @QueryParam annotation.
- This annotation is typically used in searches and when filtering data, such as filtering users by their email preferences.