Programación de Aplicaciones Telemáticas

TEMA 6: ESTANDARES JAKARTA EE

AGENDA

- Jakarta EE
- Jakarta Contexts and Dependency Injection
- Bean Validation
- Jakarta Servlet
- Jakarta RESTful Web Services
- Jakarta Persistence
- Jakarta Messaging
- Referencias

JAKARTA EE

JAKARTA™ EE

JAKARTA EE

Jakarta EE, formerly Java Platform, Enterprise Edition (Java EE) and Java 2 Platform, Enterprise Edition (J2EE) is a set of specifications, for enterprise features such as distributed computing and web services. Jakarta EE applications are run on reference runtimes, that can be microservices or application servers, which handle transactions, security, scalability, concurrency and management of the components it is deploying.

JAKARTA EE

SPECIFICATIONS

- Web specifications
- Web service specifications
- Enterprise specifications
- Other specifications

JAKARTA EE

SPECIFICATIONS

Web specifications:

- **Jakarta Servlet:** defines how to manage HTTP requests, in a synchronous or asynchronous way. It is low level and other Jakarta EE specifications rely on it;
- **Jakarta WebSocket:** API specification that defines a set of APIs to service WebSocket connections;

JAKARTA EE

SPECIFICATIONS

Web specifications:

- **Jakarta Server Faces:** a technology for constructing user interfaces out of components;
- **Jakarta Expression Language (EL)** is a simple language originally designed to satisfy the specific needs of web application developers.

JAKARTA EE

WEB SERVICE SPECIFICATIONS

- **Jakarta RESTful Web** Services provides support in creating web services according to the Representational State Transfer (REST) architectural pattern;
- **Jakarta JSON Processing** is a set of specifications to manage information encoded in JSON format;
- **Jakarta JSON Binding** provides specifications to convert JSON information into or from Java classes;

JAKARTA EE

WEB SERVICE SPECIFICATIONS

- **Jakarta XML Binding** allows mapping XML into Java objects;
- **Jakarta XML Web Services** can be used to create SOAP web services.

JAKARTA EE

ENTERPRISE SPECIFICATIONS

JAKARTA EE

ENTERPRISE SPECIFICATIONS

- **Jakarta Persistence** (**JPA**) are specifications about object-relational mapping between relation database tables and Java classes.
- **Jakarta Transactions** (**JTA**) contains the interfaces and annotations to interact with the transaction support offered by Jakarta EE.
- **Jakarta Messaging** (**JMS**) provides a common way for Java programs to create, send, receive and read an enterprise messaging system's messages.

JAKARTA EE

OTHER SPECIFICATIONS

- **Validation:** This package contains the annotations and interfaces for the declarative validation support offered by the Bean Validation API.
- **Jakarta Batch** provides the means for batch processing in applications to run long running background tasks that possibly involve a large volume of data and which may need to be periodically executed.

CDI, JAKARTA CONTEXTS AND DEPENDENCY INJECTION

CDI (Contexts and Dependency Injection) is a standard dependency injection framework included in Java EE 6 and higher. It allows us to manage the lifecycle of stateful components via domain-specific lifecycle contexts and inject components (services) into client objects in a type-safe way.

BEAN VALIDATION (JSR 303)

JSR-303 standardizes validation constraint declaration and metadata for the Java platform. Using this API, you annotate domain model properties with declarative validation constraints and the runtime enforces them.

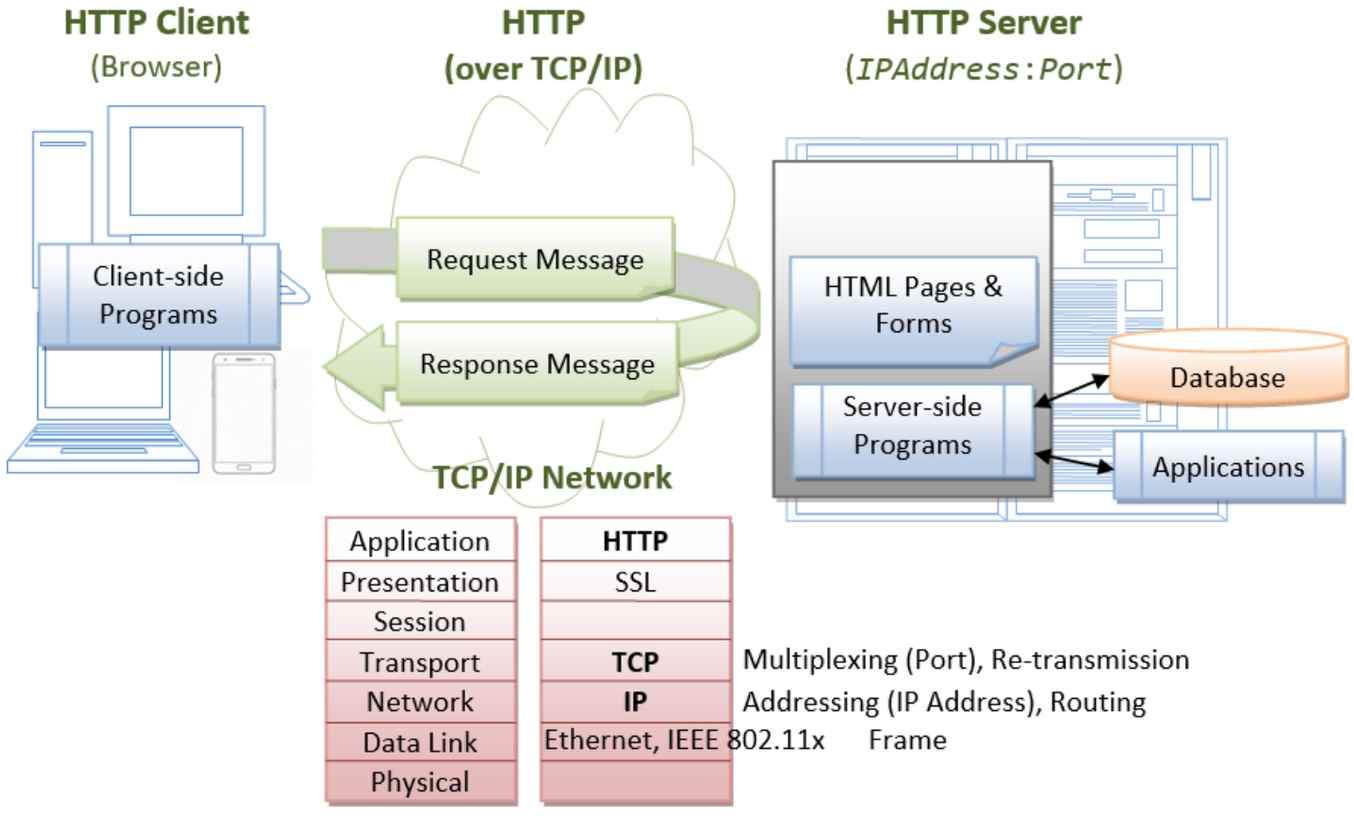
BEAN VALIDATION (JSR 303)

```
public class Car {

        @NotNull
        private String manufacturer;

        @NotNull
        @Size(min = 2, max = 14)
        private String licensePlate;

        @Min(2)
        private int seatCount;
}
```

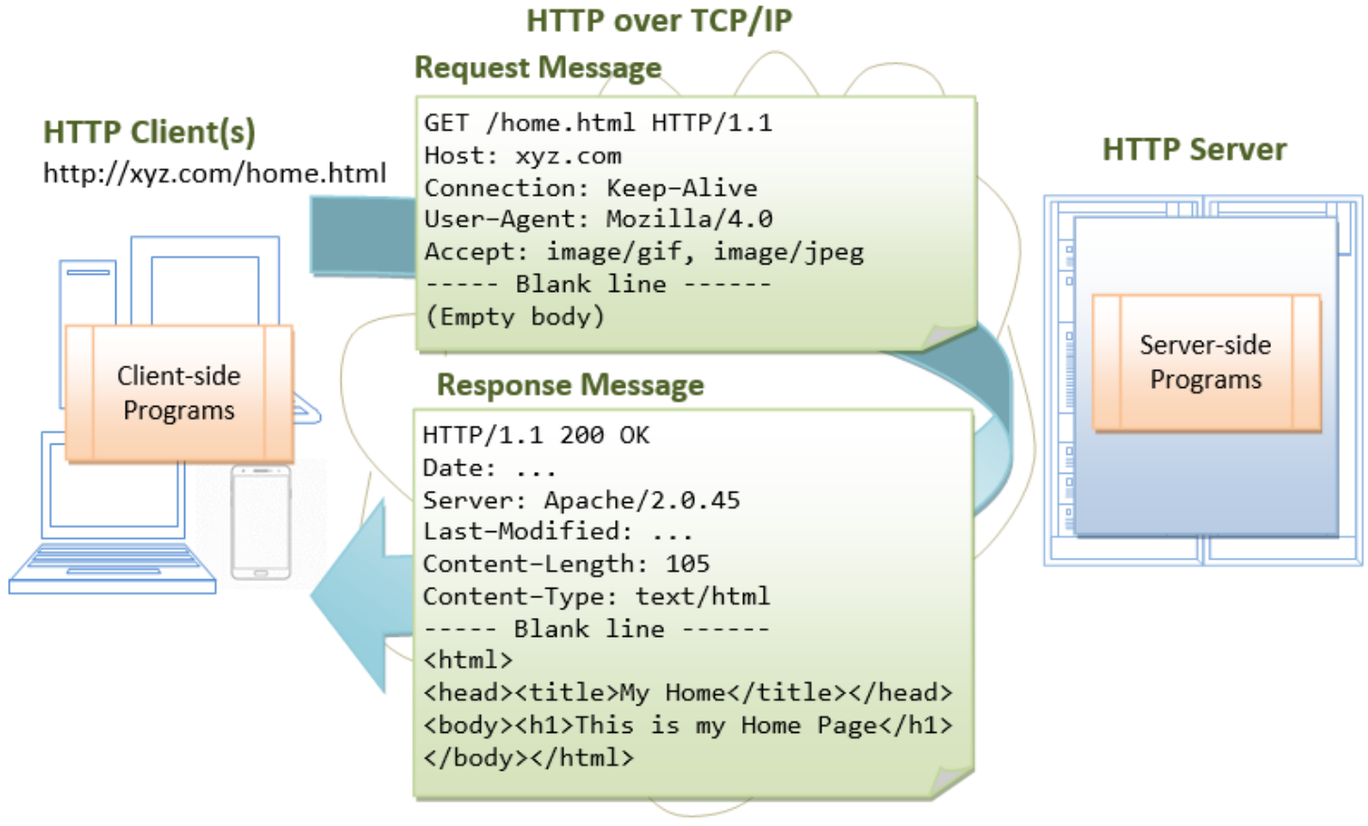
JAKARTA SERVLET

A Jakarta Servlet (formerly Java Servlet) is a Java software component that extends the capabilities of a server. Although servlets can respond to many types of requests, they most commonly implement web containers for hosting web applications on web servers and thus qualify as a server-side servlet web API.

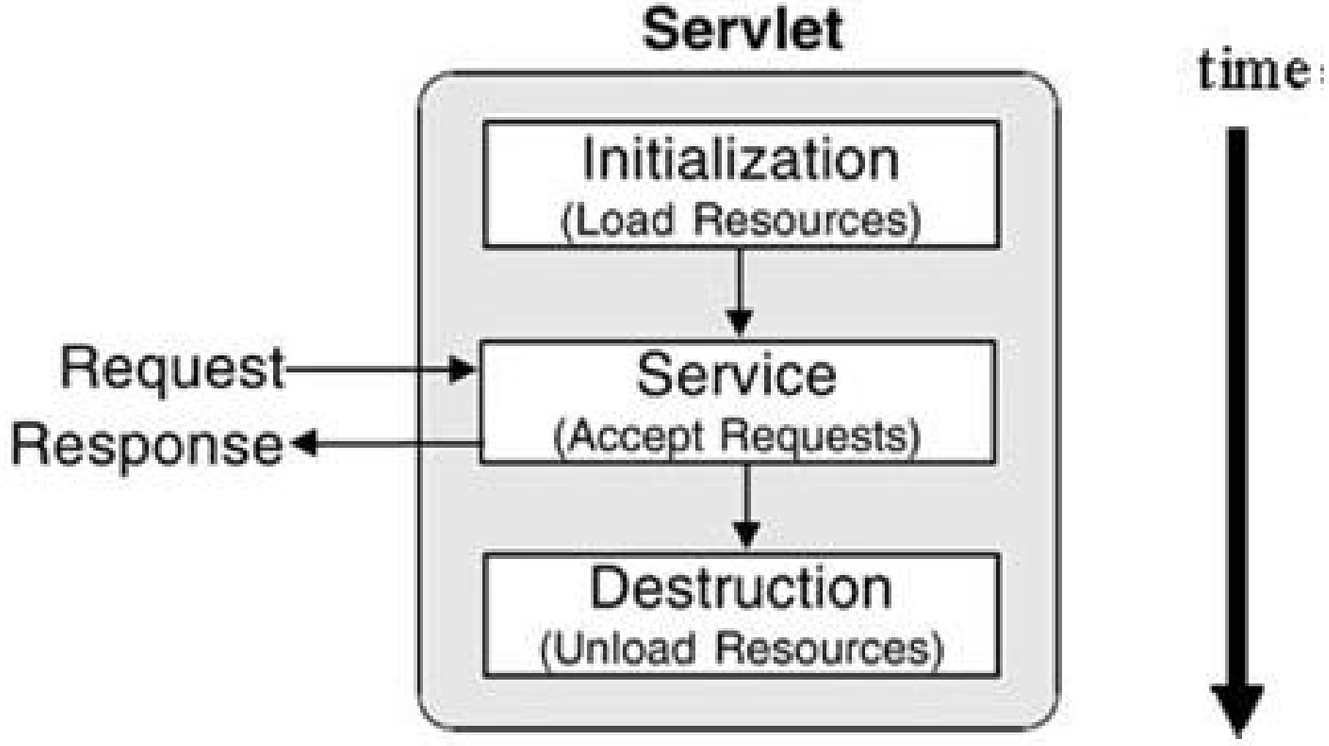
INTRODUCCIÓN



INTRODUCCIÓN

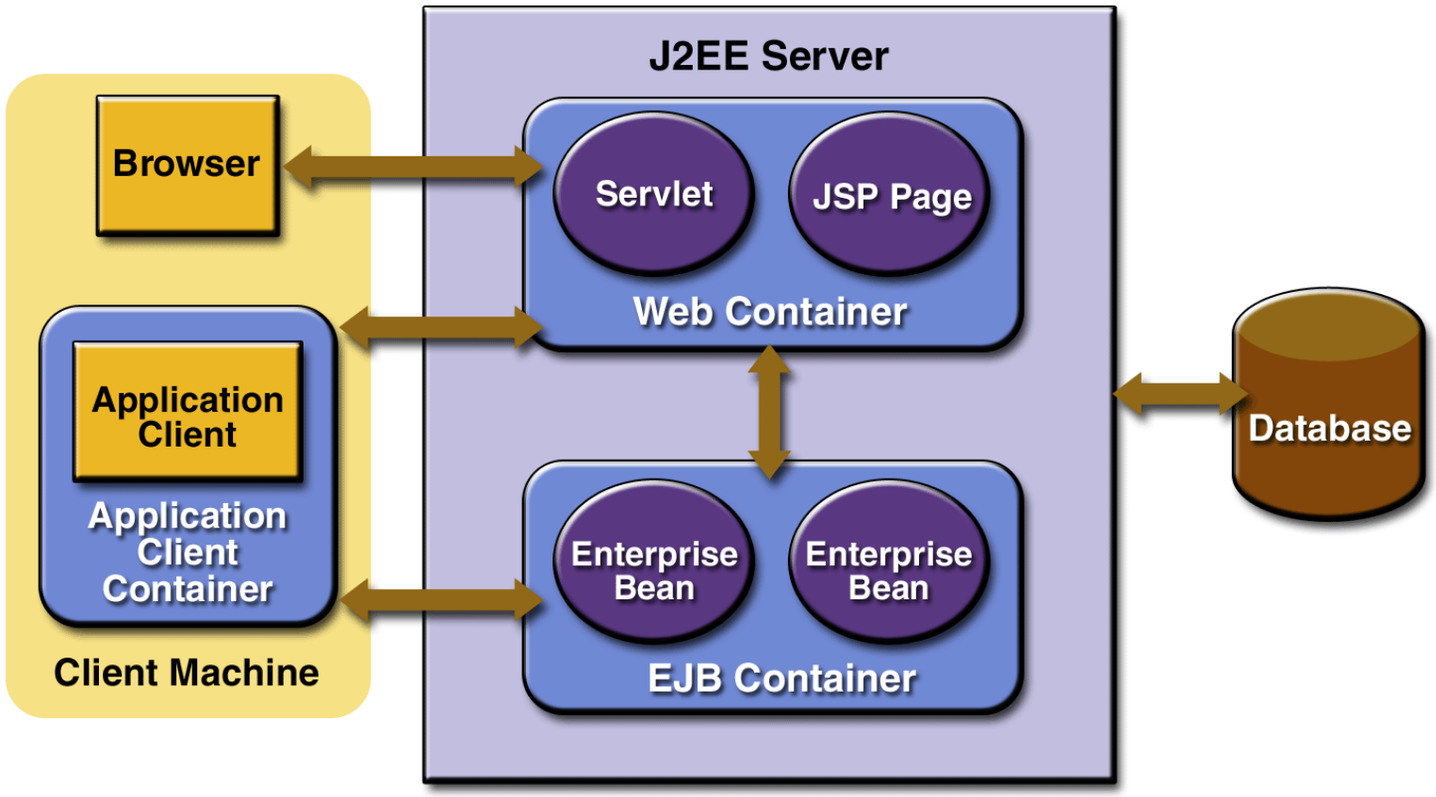


INTRODUCCIÓN



```java
public class HelloWorld extends HttpServlet {

        public void init() throws ServletException {}
        public void doGet(HttpServletRequest request, HttpServ
                throws ServletException, IOException {

                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                out.println("Hello World");
        }
        public void destroy() {}
}
```

CONTENEDORES DE APPLICACIONES



CONTENEDORES DE APPLICACIONES

.war files

JAX-RS, JAKARTA RESTFUL WEB SERVICES (JSR-370)

Jakarta RESTful Web Services, (JAX-RS; formerly Java API for RESTful Web Services) is a Jakarta EE API specification that provides support in creating web services according to the Representational State Transfer (REST) architectural pattern.

JAX-RS, JAKARTA RESTFUL WEB SERVICES (JSR-370)

JAX-RS provides some annotations to aid in mapping a resource class (a POJO) as a web resource.

- @Path specifies the relative path for a resource class or method.
- @GET, @PUT, @POST, @DELETE and @HEAD specify the HTTP request type of a resource.
- @Produces specifies the response MIME type.
- @Consumes specifies the accepted request Internet media types.

JAX-RS, JAKARTA RESTFUL WEB SERVICES (JSR-370)

JAX-RS, JAKARTA RESTFUL WEB SERVICES (JSR-370)

```
@Path("/employees")
public class EmployeeResource {

    @Autowired
    private EmployeeRepository employeeRepository;

    @GET
    @Path("/{id}")
    @Produces({ MediaType.APPLICATION_JSON, MediaType.APPLICAT
    public Employee getEmployee(@PathParam("id") int id) {
        return employeeRepository.getEmployee(id);
    }
}
```

JPA, JAKARTA PERSISTENCE

Jakarta Persistence (JPA; formerly Java Persistence API) is a Jakarta EE application programming interface specification that describes the management of relational data in enterprise Java applications.

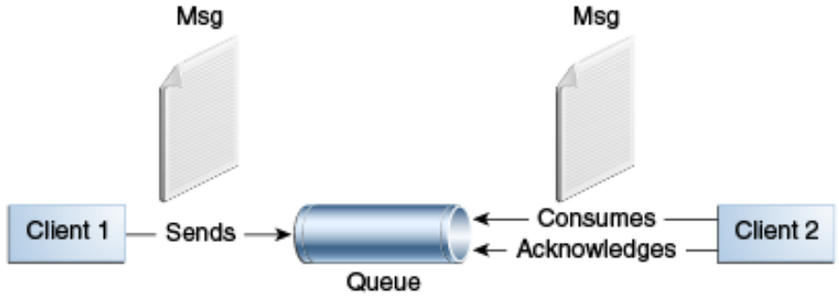
JPA, JAKARTA PERSISTENCE

```
public interface SimpleUserRepository extends CrudRepository<

        User findByTheUsersName(String username);

        Optional<user> findByUsername(Optional<string> usernam

        List<user> findByLastname(String lastname);

        @Query("select u from User u where u.firstname = :firs
        List<user> findByFirstname(String firstname);
}
                                                    </user></user>
```

JMS, JAKARTA MESSAGING (JSR-343)

The Java Message Service (JMS) API is a Java Message Oriented Middleware API for sending messages between two or more clients. It is a programming model to handle the producer-consumer messaging problem.

JMS, JAKARTA MESSAGING (JSR-343)



REFERENCIAS