Interceptors:

Embeded EJB container

Java+xml+annotations

AOP(aspect orientated programming)

New features:

-annotations

-callback methods and listeners

-interceptors

-DI(dependency injection)

Business logic-> SessionBean

Workflow logic -> MessageDrivenBean

JPA entity wrapped within SessionBean, no entity bean

Session beans-stateless (SLSB)

-stateful SFSB

-singleton (3.1+)

MDB->SLSB

-no client orientated

Entity bean ->BMP

SessionBean @Stateless

@Stateful

@Singleton

MessageDriver @MessageDriven

JPA Entities @Entity

@EntityManager

EJB Access Mode:

-default mode: local access(not over the network, only container)-> other EJB only in the same context

-remote ->same context and outside context

-web service ->SB and MDB – web services

->manages wsdl/soap mappings

|  |  |
| --- | --- |
| Bean Type | Annotation |
| SB | @Local,@Remote,@LocalBean(EJB 3.1),@WebService |
| MD | @WebService |
| JPA | N/A |

EJB Instance Management

In container-managed,container manages:

-transaction

-roles

-security

-Persistence and entities (persistence context)

In bean-managed, bean manages:

-transaction

-persistence

Manually management

Transactions @TransactionManagement(Container, Bean)

Security @RunAs()

@RolesAllowed

EJB access->no direct client access

Container instantiates EJBObject(proxy) jndi lookup

DI lookup EJB instance

Lifecycle phase:

-does not exist-> no bean instance in memory

-post construct/ not ready ->bean instance exists but not ready for interaction

-ready->exists,initialized and ready for interactions

-pre-removal->about to be removed

EJB Development

Each EJB described by metadata. Business interface ->interface

1.Write client

2.Write EJB

3.Define config

4Compile

5.Package

6.Generate client access classes

7Deploy

8.Test

Session bean cardinality:

|  |  |  |  |
| --- | --- | --- | --- |
|  | @Stateless | @Stateful | @Singleton |
| Remote Client Access | X | X | X |
| Local Client Access | X | X | X |
| Concurent client access |  |  | X |
| Unique per client |  | X |  |
| Client based instanced | Pooled | 1:1 | Many:1 |

Session Bean:

-must have public no argument constructor

-no finalize method

Local JNDI Name ->Mapped name (ptr Queue JNDI Name)

@EJB=@Inject

@EJB or @Resource (Same)

Session Façade could be implemented as @Local or @Remote

Interface for Façade, Bean A, Bean B

Java ME(Micro Edition) -> internet of things

Java EE extends Java SE(standard edition)

Enterprise App –multi tiered, scalable, reliable, secure

Client tier: ME, EE, FX browser

Middle tier: Java EE (webtier, business tier, interoperability tier)

EIS tier: Servers, Resources, Mainframes

@=metadates

https=combination between http and ssl

RMI/IIOP:-remote method invocation

* Internet inter ORB Protocol

Business tier: transactions, persistence, validation, batch

Web tier: servlets, web pages, web sockets, expression language

Common Services: injection, interception, security, concurrency

Java SE: DB access, directory, XML, remote, management

Bean Validation

@-> bean validation with annotations

In bean se foloseste intotdeauna local name(jndi name). In Active MQ remote jndi name.

@interfaceName

Constraint aggregation

Validated by={Classimpl.class}

Message()

Groups()

Payload();

!ValidationMessages.properties

CDI=context and dependency injection @Resource=@Inject

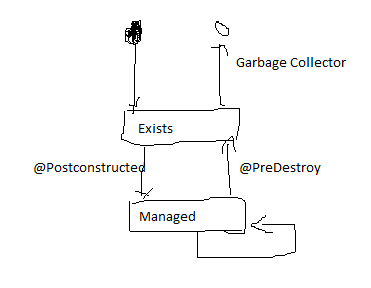
JSF=java server faces

Disable validation in web.xml

*Javax.faces.validator.DISABLE\_DEFAULT\_BEAN\_VALIDATOR*

IoC= container takes control of services (container where beans are executed).

Lifecycle of CDI Bean

Method invocation

Callback annotations: @Postconstructed, @PreDestroy not static or final (methods)

@Target ({TYPE,METHOD,PARAMETER,FIELD})

@Retention(RUNTIME)

ElementType.TYPE

@Vetoed = not visible by CDI (mascheaza fata de CDI cand trebuie facuta injectia)

@Alternatives = depends on environment(version, test, production)

@Observes ->synchronous ->

Expression Language #{expression{

@Named->bind to expression language

@ApplicationScoped-> one per application

@SessionScoped->one per session

@RequestScoped->not have to be serializable

@ConvesationScoped, @Dependent