EJB components are designed to encapsulate business logic and to protect the application developers from having to worry about many system level issues including Transactions, Security, Concurrency, Communication, Resource management, Persistency, Error handling and operating environment independency.

An EJB is just a collection of Java classes and xml file bundled into single unit.

An EJB runs in an EJB be container. The EJB containers runs within an application server and tracks responsibility for system level issues.

difference between javabeans and Enterprise Java beans

1. Java beans are reusable components that follow a naming convention that can be used by development tools
2. Regular javabeans run in a JVM but EJB must be run under the control of EJB container.
3. Java beans are used to customize existing object, but EJB’s are used to standardize the way in which business logic is written.
4. Java Bean and not distributed but EJB’s are distributed.
5. EJB’s live and run in the server and the server does Virtually everything to manage transaction, security, persistency and even the life and death of your objects.

EJB container

An EJB container is environment in which EJB execute. Its primary role is to serve as a buffer between the EJB and outside world.

Client’s don't directly connect to the EJB beans. The container is responsible for providing the beans with services.

these services are as follows:

1. support for transactions
2. support for management of multiple instances
3. support of persistency
4. support for security

EJB architecture

An EJB system is logically a 3 tier system. The 3 tiers are as follows:

1. the client
2. the EJB service
3. the database

this structure is a logical architecture because the three-tier don't necessarily have to reside on three different machines.

* The EJB container and the database might reside on the same machine
* The client and the EJB service might reside on the machine
* if you combine the two cases all three tier can reside on a single machine.

Types of EJB

EJB are of 3 types

1. session Beans
   1. Stateless Session Bean
   2. stateful session beans
2. JPA or Java persistence API
3. message driven Bean

session Bean - a session Bean typically represent a process. when you think of session think of verb. a session Bean is scoped to a single client and performs work on behalf of the client and is created in response to a request from client and is dedicated to the client.

The client communicates with the session Bean by invoking its methods.

The session Bean is not guaranteed to recover from a server failure.

The container will not allow concurrent access to single Session Bean. if client A is using the Bean Client B we will get an exception if it calls the method on that Bean.

Session beans perform business task without having a persistent storage mechanism such as database and can use the share data.

Session are not persistent. when the client terminated, session Bean appears to terminate and is no longer associated with the client.

Session Bean can be stateless or stateful

A Stateful bean can remember conversational state between methods called while a stateless Bean will not remember anything about a client between method invocation.

stateless Bean simply forget about the clients once the methods call completes.

Steps for Creating a Session bean

1. create a remote interface and keep the prototype of all methods in this interface which you want to invoke remotely and throw remote exception from each method.
2. code the Bean class which will implement the above interface.
3. create an XML file names ejb-jar.xml and keep it blank<ejb-jar></ejb-jar>
4. put this XML file into the META-INF folder.
5. Put this Bean class, interface and XML file into a JAR file.
6. deploy the JAR file into the container

stateless session - Stateless Session Bean performs a task for a particular client unlike a stateful session Bean, Stateless Session Bean do not maintain state.

Stateless Session Bean maintains the state only for the duration of a method invocation.

when the method is finished the state is no longer retained. all instances of a stateless bean are equivalent, allowing the EJB container to assign an instance to any client request.

some important points on Stateless Session Bean

1. Stateless Session Bean are never returned to secondary storage
2. For applications that require large number of clients, skeletons beans offer better scalability.
3. the system overhead of creating an instance is less for Stateless Session Bean then for a stateful session Bean.
4. Stateless Session Bean do not have instance variable to store information
5. with the Stateless Session Bean that container has the opportunity for Optimization by pooling beans. since they don't have state, there is no effect on the Beans from any clients call, so no one bean can be used for multiple clients
6. stateless Bean creation all removal is not tied to the client