

## EJB 3.0 – Stateless Session Bean

*Giulio Rossetti*

### Introduzione:

Gli Enterprise Java Beans 3.0 sono la diretta evoluzione della versione 2.1 nota per le tonnellate di file XML da compilare per riuscire a eseguire un corretto deploy e per la politica di naming delle entità deployate non sempre di immediata comprensione. EJB3, per la gioia dello sviluppatore, elimina tutta la logica di descrizione degli oggetti (rappresentata dai file XML) della precedente versione lasciando il compito di effettuare tali configurazioni alla JVM e all'application server tramite l'uso di annotations, scelta che rappresenta un chiaro punto di forza della nuova versione rilasciata dalla Sun dalla versione 1.5 di Java. Altro aspetto saliente è la modifica introdotta al sistema di lookup dei bean da parte del client, non sono infatti più necessarie operazioni di Narrowing, il reperimento delle classi avviene sullo stile utilizzato con successo e semplicità nel caso di RMI.

### Procediamo con ordine:

la specifica EJB3 definisce tutti i suoi componenti (siano essi Entity o Session Bean) come POJO (ovvero Plain Old Java Object) il che significa che essi sono istanziabili come normali classi Java oltre che tramite il loro uso remoto. Un EJB implementa due interfacce (una remota e una locale) definite dallo sviluppatore, interfacce standard del linguaggio Java arricchite dalla segnalazione tramite annotations del loro particolare stato. Un esempio possono essere:

```
//interfaccia locale
@Local
public class MySessionLocale{ }
```

```
//interfaccia remota
@Remote
public class MySessionRemote{
    public String saluta();
}
```

In questo tutorial prenderemo in considerazione esclusivamente i così detti Stateless Session Bean poiché trattandosi di una guida di base lo scopo è quello di mostrare l'uso immediato per applicazioni di realizzazione immediata (possiamo in questo caso considerare gli EJB coinvolti al pari di WebServices o di Remote Procedure Call sul modello fornito da RMI).

Come specificato dal nome gli Stateless Session Bean non tengono traccia, a seguito della invocazione di un loro metodo, dello stato dell'oggetto sul server; per svolgere un'attività simile sarà invece necessario l'impiego di Statefull Session Bean. Considerando le interfacce definite precedentemente un EJB che le implementa può essere il seguente:

```
//ejb3
@Stateless(name="mysession")
public class MySession implements MySessionRemote,MySessionLocal{
    public MySession(){ }
    public String saluta(){
        return "Ciao";
    }
}
```

le cose da notare sono due:

- Ogni classe che rispetti le specifiche Java Bean può essere usata come EJB3 (costruttore vuoto e proprietà accedute da metodi getXXX e setXXX)
- @Stateless indica che si tratta di un Stateless Session Bean (e che come tale deve essere trattato durante il deploy su application server) mentre (name="mysession") specifica il Naming da usare per il lookup

Le differenze rispetto alla versione 2.1 sono notevoli, adesso non è più necessario preoccuparsi del caricamento del bean implementando i metodi appositi né dichiararne l'esistenza all'application server tramite file di configurazione: le annotation vengono infatti utilizzate al pari di quelli che sono gli "include" in C, esse vengono sostituite dal

compilatore con tutto ciò che è necessario per definire l'entità EJB che deve essere creata a runtime.

### Un semplice client:

Vediamo adesso come effettuare un test dell'EJB così creato:

```
//classe di test
public class Test{
    public static void main(String [] args) throw Exception{
        //spiegheremo in seguito come ottenere un context corretto
        Context context ctx = new InitialContext();
        //eseguo il lookup
        MyRemote session = (MyRemote) ctx.lookup("mysession/remote");
        //chiamo il metodo
        System.out.println(session.saluta());
    }
}
```

Come si può notare con poche righe di codice siamo in grado di definire un test per il nostro EJB3.

Analizziamo meglio i due punti focali di questa classe:

- *Context context ctx = new InitialContext();*  
con questa riga si richiede un Context su cui operare il successivo lookup, il Context può essere definito in più modi (tramite un file di configurazione o come vedremo tramite Properties)
- *MyRemote session = (MyRemote) ctx.lookup("mysession/remote");*  
qui avviene il reperimento dell'EJB dall'application server. Notiamo in primo luogo che è stato utilizzato un cast all'interfaccia MyRemote solo ed esclusivamente poiché il metodo implementato era solo in essa definito. Con la stringa "mysession/remote" andiamo a specificare il nome che abbiamo assegnato al bean (nell'annotation @Stateless) e il tipo di interfaccia di interesse (remote o local). Se l'EJB dovesse essere deployato all'interno di un archivio ear (e solamente in quel caso) la stringa diverrebbe "nome\_archivio\_ear/nome\_ejb/tipo\_interfaccia".