Una risorsa è un oggetto del programma che offre connettività a sistemi quali Server Database e sistemi di messaging.

I componenti Java EE possono accedere a una grande varietà di risorse, incluse databases, mail sessions, Java Message Service Objects e URL’s.

La piattaforma Java EE7 offre meccanismi che permettono di accedere a tutte queste risorse in un modo simile tra loro.

In questo capitolo esamineremo i vari tipi di risorse e spiegheremo come crearle.

## 3.1 Resources and JNDI Naming

In un’applicazione commerciale di componenti hanno la necessità di accedere a altri componenti e risorse, quali ad esempio i database.

Nella piattaforma Java EE il servizio di naming JNDI permette ai componenti di individuare altri componenti e risorse.

Una risorsa è un oggetto che offre connessioni a sistemi quali appunto database e sistemi di messaggistica.

Ogni risorsa è identificata da un nome univoco chiamato JNDI name.

Nel server GlassFish si possono usare sia al Administration Console che il comando *asadmin* per creare le risorse.

L’applicazione poi utilizza le annotazioni per iniettarle.

Se un’applicazione utilizza l’iniezione sulle risorse la JNDI API è invocata dal server di GlassFish e non dall’applicazione anche se è possibile per un’applicazione individuare le risorse facendo chiamate dirette all’API.

Una risorsa e il suo nome JNDI sono legate dal servizio di naming and directory.

L’iniezione delle risorse viene effettuata tramite l’annotazione @Resource.

Si può utilizzare un descrittore (deployment descriptor) per effettuare l’override del mapping che è stato specificato in un’annotazione, in questo modo si può modificare un’applicazione tramite il repackagin invece di ricompilare sia il codice sorgent che il repackaging.

Comunque per la maggior parte delle applicazioni il descrittore non è necessario.

## 3.2 Oggetti DataSource e Connection Pool

Per archiviare, organizzare e richiamare dati la maggior parte delle applicazioni usa un database relazionale.

I componenti Java EE7 possono accedere ai database relazionali tramite l’API JDBC.

In JDBC l’accesso al database si ottiene tramite oggetti DataSource.

Un DataSource ha un set di proprietà che identificano e descrivono la fonte dei dati reali che rappresentano.

Queste proprietà includono informazioni quali la posizione del server del database, il suo nome, il protocollo di rete da usare per comunicarci, etc.

In un server GlassFish una fonte di dati è detta risorsa JDBC.

Le applicazioni accedono alla fonte dei dati tramite una connessione e un oggetto DataSource rappresenta una “fabbrica” di connessioni per la fonte dati che il DataSource rappresenta.

In un’implementazione base di un oggetto DataSource il metodo *getConnection* ritorna un’oggetto connection che è la connessione fisica alla fonte dati.

Un oggetto DataSource può essere registrato con il servizio di naming JNDI, se fatto l’applicazione può utilizzare l’API per accedere all’oggetto DataSource che può a sua volta essere utilizzato per connettere la fonte dati che rappresenta.

Gli oggetti DataSource che implementino il connection pooling producono anche una connessione a una particolare fonte dati che la classe DataSourcerappresenta.

L’oggetto connection che il metodo getConnection ritorna è un gestionale per un oggetto PooledConnection più che una connessione fisica.

L’applicazione utilizza l’oggetto connection nello stesso modo in cui utilizza la connessione.

Il ConnectionPooling non ha effetto sul codice dell’applicazione eccetto che una pooled connection, come ogni connessione, deve sempre essere chiusa esplicitamente.

Quando l’applicazione chiude una connessione pooled la connessione torna in un pool di connessioni riutilizzabili e la prossima volta che il metodo getConnection viene chiamato una di queste connessioni disponibili verrà ritornata.

Dato che il connection pooling evita di dover creare una nuova connessione fisica ogni volta che ce n’è la necessità l’applicazione girerà più velocemente.

Le applicazioni che usano la Persistence API specificano l’oggetto DataSource utilizzato nell’elemento

<jta-data-source>jdbc/MyOrderDB</jta-data-source>

Questa solitamente è l’unica referenza a un oggetto JDBC in una persistence unit.

## 3.3 Creating Resources Aministratively

Prima di sviluppare o far girare molte applicazioni potrebbe essere utile creare risorse.

Un’applicazione può includere un file *glassfish-resources.xml*  che può essere usato per definire tali risorse per quella o altre applicazioni.

Si può usare anche il comando *asadmin* specificando come argomento un file nominato *glassfish-resources.xml*  per creare le risorse come amministratore.

*asadmin add-resources glassfish-resources.xml*

Il file *glassfish-resources.xml* può essere creato in ogni applicazione a mano o utilizzando NetBeans IDE.