Questo capitolo offre una panoramica sulle iniezioni in Java EE e descrive i due meccanismi per utilizzarle forniti dalla piattaforma: **resource injection** e **dependency injection**.

Java EE offre un meccanismo per le dipendenze che permette agli oggetti di ottenere la referenza alle risorse o altre dipendenze senza doverle instanziare direttamente ma dichiarandole con l’utilizzo delle annotations.

Il container poi mette a disposizione le istanze richieste a runtime.

L’iniezione semplifica il codice e lo divide dall’implementazione delle dipendenze.

## 4.1 Resource Injections

Le resource injections permettono di iniettare ogni risorsa disponibile nel namespace JNDI in ogni oggetto gestito dal container come servlet, enterpreise beans o managed beans.

Usualmente si utilizza un’interfaccia per referenziare l’istanza iniettata e dividere il codice dall’implementazione delle risorse.

Ad esempio il codice seguente inietta un oggetto DataSource che offre connessioni per il Java DB di default tramite GlassFish Server.

*public class MyServlet extends HttpServlet {*

*@Resource(name="java:comp/DefaultDataSource")*

*private javax.sql.DataSource dsc;*

*...*

*}*

Oltre a utilizzare le iniezioni sulle proprietà dell’esempio precedente possiamo iniettare le risorse sul metodo:

*public class MyServlet extends HttpServlet {*

*private javax.sql.DataSource dsc;*

*...*

*@Resource(name="java:comp/DefaultDataSource")*

*public void setDsc(java.sql.DataSource ds) {*

*dsc = ds;*

*}*

*}*

Per usare le iniezioni sul metodo setter deve essere conforme alle convenzioni JavaBeans: il nome del metodo deve iniziare con set, avere void come valore di ritorno e avere solo un parametro.

L’annotazione @Resource è contenuta nel package *javax.annotation*.

Le resource injections non sono typesafe perché prende in considerazione il nome e non il tipo: il tipo della risorsa non è conosciuto in fase di compilazione quindi potrebbero occorrere errori a runtime se il tipo dell’oggetto e quello della referenza non coincidessero.

## 4.2 Dependency Injections

Le Dependency Injection permettono di trasformare delle classi Java normali in managed objects e di iniettarle in ogni altro managed object.

Usando le dependency injection il codice può dichiarare dipendenze su ogni managed object.

Il container offre in automatico le istanze di queste dipendenze agli injection points in runtime e gestisce il ciclo di vita di queste istanze in autonomia.

Le Dependency Injection in Java EE definiscono il campo di validità (scope) e con questo determinano il ciclo di vita degli oggetti che il container instanzia e inietta.

Possiamo definire object managed, che poi andremo a iniettare, assegnando un campo di validità a una normale classe:

*@javax.enterprise.context.RequestScoped*

*public class CurrencyConverter { ... }*

Per iniettare un managed bean si usa l’annotazione *javax.inject.Inject*:

*public class MyServlet extends HttpServlet {*

*@Inject CurrencyConverter cc;*

*...*

*}*

A differenza delle Resource Injection le Dependency Injection sono typesafe perché prendono in considerazione il tipo.

Per dividere il codice dall’implementazione dei managed bean possiamo referenziare le istanze iniettate usando un’interfaccia in cui implementarli.

## 4.3 Differenze principali tra Resource Injection e Dependency Injection

