CHAPTER 1

Introducing Seam

The first thing to understand about Seam is that it's a framework. Actually, it's a framework that sits on top of another framework (Java EE), and that framework sits on top of another one (Java). But don't get distracted by that just yet.

That word "framework" is a broad one, adopting many meanings depending on how it's used (and who is using it). In this case, I mean "framework" in a typical software technology sense: Seam knits together a set of APIs and services into an environment that makes it easy (or easier) to write Java EE web applications.

A framework typically "makes it easier" to do something by simplifying common tasks and providing built-in utilities that you'd otherwise have to write yourself. Seam is no different. Seam is based on Java EE, so it satisfies its framework duties in two fundamental ways:

- *Seam **simplifies** Java EE*: Seam provides a number of shortcuts and simplifications to the standard Java EE framework, making it even easier to effectively use Java EE web and business components.

- *Seam **extends** Java EE*: Seam integrates a number of new concepts and tools into the Java EE framework. These extensions bring new functionality within the Java EE framework.

You'll get familiar with Seam in this chapter by briefly examining each of these aspects. In the rest of this chapter, I'll list for you the various services and utilities that Seam provides. In the chapters that follow, you'll see these services in action directly, applied in application development cases.

Seam Simplifies Java EE

The standard Java EE environment consists of the Java Standard Edition (Java SE) with all of its APIs (JDBC for database access, JAXP for XML processing, etc.) supporting all of the enterprise-level capabilities of Java EE (JSF/JSP/servlets for web components, JAX-WS for web services, etc.). Your application components are then built directly on top of this overall framework, as depicted in Figure 1-1.

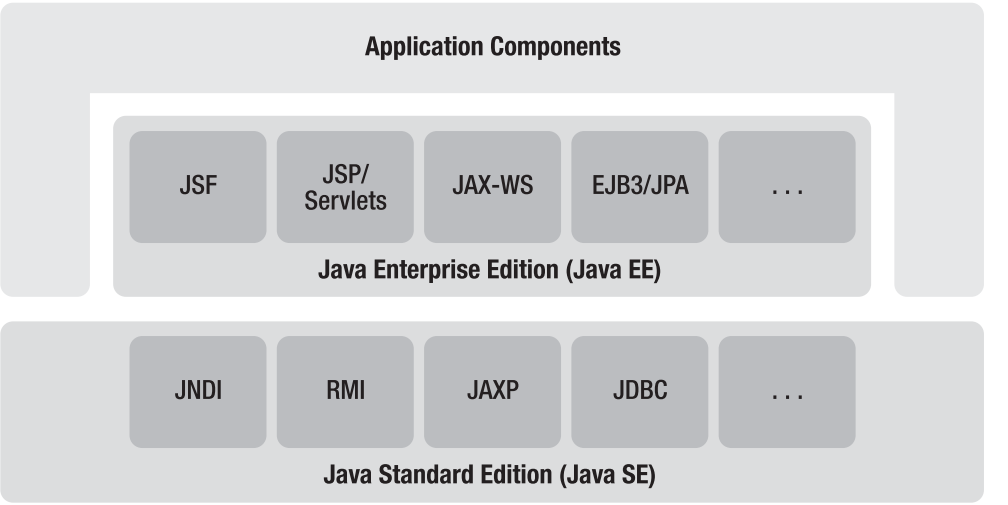

Figure 1-1. *Standard Java EE framework*

In addition to the APIs and component types depicted in Figure 1-1, Java EE also provides the deployment services, runtime security, and other services you need to create effective applications. And Java EE provides a number of improvements over its predecessor framework, J2EE, for example:

- Java 5.0 annotations are integrated liberally throughout the APIs in Java EE, giving you the option of using either externalized XML deployment data or embedded code annotations.
- The JavaServer Faces (JSF) 1.2, Java API for XML-based Web Services (JAX-WS) 2.0, and Enterprise JavaBeans (EJB) 3.0 APIs offer easier programming models than their J2EE predecessors, allowing you to implement most web, web service, and business components using simple JavaBeans.
- EJB 3.0 eliminates the need for many of the interfaces and other artifacts required in earlier versions of EJB, in most situations.

Even with the improvements delivered with Java EE, the JBoss Seam team saw room for simplifying things even further. Figure 1-2 depicts the Seam framework layered between your application code and the Java EE framework.