**Using scripts**

Scripts are text files, typically with an extension of \*.groovy, that can be executed from the command shell like this:

> groovy myfile.groovy

**Note:**

This is very different from Java. In Groovy, you’re executing the source code! An ordinary Java class is generated for you and executed behind the scenes. But from a user’s perspective, it looks like you’re executing plain Groovy source code.

Book b = new Book("The art of War")

println(b.getTitle())

println(getTitleBackwards(b))

String getTitleBackwards(book){

return book.getTitle().reverse()

}

Note how you’re able to invoke the method getTitleBackwards before it’s declared. Behind this observation is a fundamental difference between Groovy and scripting languages such as Ruby. A Groovy script is fully constructed—that is, parsed, compiled, and generated—*before execution*.

Another important observation is that you can use Book objects without explicitly

compiling the Book class! The only prerequisite for using the Book class is that

Book.groovy must reside on the classpath.

The Groovy runtime system will find the file, compile it transparently into a class, and yield a new Book object. Groovy combines the ease of scripting with the merits of object orientation.