25.11 Using an External Editor

Section 25.3.10 demonstrated **JShell Edit Pad** for editing code snippets. This tool provides only simple editing functionality. Many programmers prefer to use more powerful text editors. Using JShell's `/set editor` command, you can specify your preferred text editor. For example, we have a text editor named EditPlus, located on our Windows system at
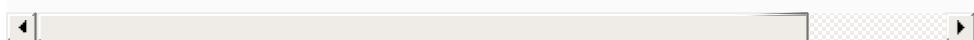
```
C:\Program Files\EditPlus\editplus.exe
```

The JShell command

```
jshell> /set editor C:\Program Files\EditPlus\editplu
|  Editor set to: C:\Program Files\EditPlus\editplus.

jshell>
```

sets `EditPlus` as the snippet editor for the current JShell session. The `/set editor` command's argument is *operating-system specific*. For example, on Ubuntu Linux, you can use the built-in `gedit` text editor with the command

```
/set editor gedit
```

◀ | ▶

and on macOS,[8] you can use the built-in `TextEdit` application with the command

[8.] On macOS, the `-wait` option is required so that JShell does not simply open the external editor, then return immediately to the next `jshell>` prompt.

```
/set editor -wait open -a TextEdit
```

◀ | ▶

Editing Snippets with a Custom Text Editor

When you're using a custom editor, each time you save snippet edits JShell immediately re-evaluates any snippets that have changed and shows their results (but not the snippets themselves) in the JShell output. The following shows a new JShell session in which we set a custom editor, then performed JShell interactions—we explain momentarily the two lines of output that follow the `/edit` command:

```
jshell> /set editor C:\Program Files\EditPlus\editplu
|  Editor set to: C:\Program Files\EditPlus\editplus.

jshell> int x = 10
x ==> 10

jshell> int y = 10
y ==> 20

jshell> /edit
```
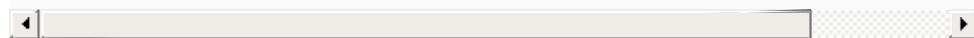
```
y ==> 20
10 + 20 = 30
jshell> /list

    1 : int x = 10;
    3 : int y = 20;
    4 : System.out.print(x + " + " + y + " = " + (x +

jshell>
```

First we declared the int variables x and y, then we launched the external editor to edit our snippets. Initially, the editor shows the snippets that declare x and y (Fig. 25.4).
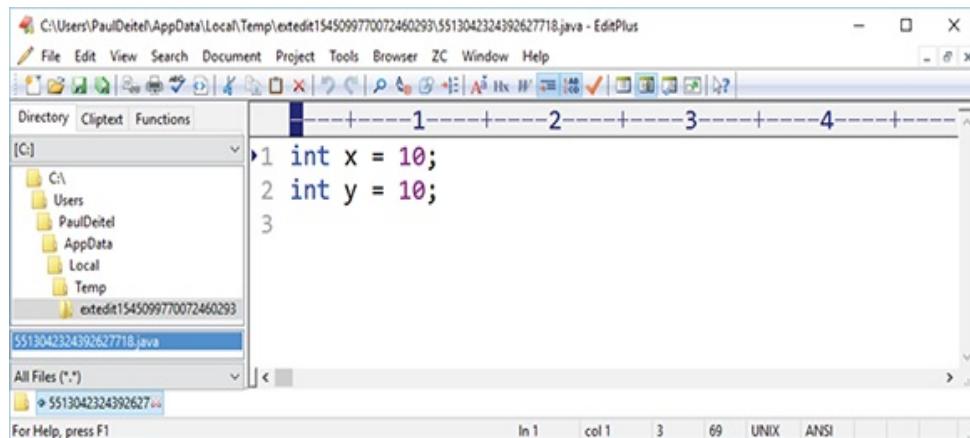

Fig. 25.4

External editor showing code snippets to edit.

Next, we edited y's declaration, giving it the new value 20, then we added a new snippet to display both values and their sum (Fig. 25.5).
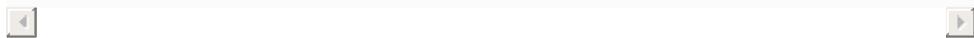
When we saved the edits in our text editor, JShell replaced y's original declaration with the updated one and showed

```
y ==> 20
```

to indicate that y's value changed. Then, JShell executed the new `System.out.print` snippet and showed its results

```
10 + 20 = 30
```

Finally, when we closed the external editor and pressed *Enter* in the command window, JShell displayed the next `jshell>` prompt.
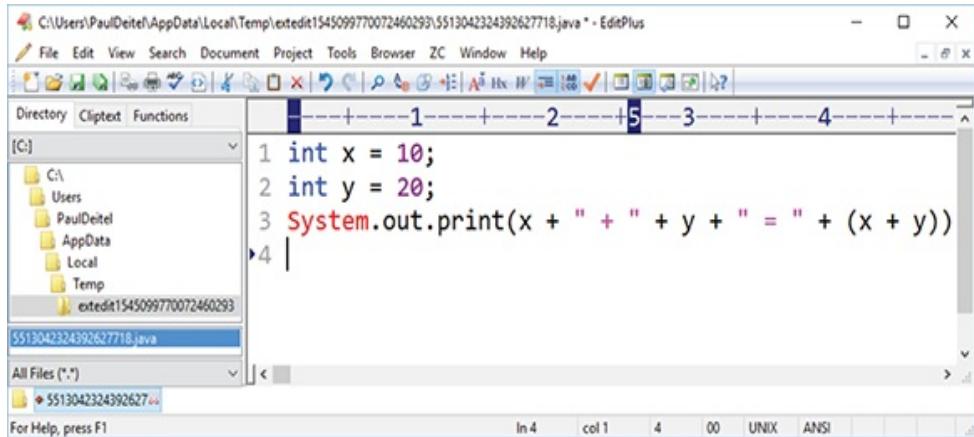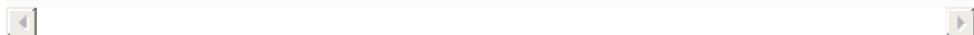

Fig. 25.5

External editor showing code snippets to edit.

Retaining the Editor Setting

You can retain your editor setting for future JShell sessions as follows:

```
/set editor -retain commandToLaunchYourEditor
```

Restoring the JShell Edit Pad As the Default Editor

If you do not retain your custom editor, subsequent JShell sessions will use **JShell Edit Pad**. If you do retain the custom editor, you can restore **JShell Edit Pad** as the default with

```
/set editor -retain -default
```