Transcript: First Program in jGRASP

>> So now let's look at our first program with jGRASP. If you will start up jGRASP, assuming you've got it installed, and then we're gonna do a number of activities. We're gonna open a new file, enter a program incrementally. This program is going to print out War Eagle three times.

We're going to save the program, compile it, run it. We'll do some things like setting a breakpoint and running a Debug mode. And we will also generate control structure diagrams, documentation, turn on and off, turn on and off-line numbers, and so on. So, looking at jGRASP, when you first open it, the desktop is organized into three primary panes.

The large window is where we'll edit our programs. Below that is the message window, it includes Compiler Messages, Messages from jGRASP. Your Run Input and Output, and Interactions, today, we'll see Compile Messages and Run I/O. On the left, we have a pane that's set to the Browse tab, that will be the default.

And this is where, this is the directory or files... of files that jGRASP was actually looking at. And up at the top, there's a little folder with an arrow on it, you can click that to go up. And then I'm looking now at where our notes are, the introductory set of notes, you can double click on that to go back down, and so on.

We have other tabs here, Find, Debug and Workbench, and we'll look at those a little bit later. So let's begin by opening a new file. You can do that with File, New, Java, and now there we see we've got an empty file at the bottom. You can also do a Ctrl+N for new, or Cmd+N, that will give you the Open File dialog, and create one as well.

I'm gonna go ahead and close this last one, and you close a file by clicking the X on the upper right, or you can do a Ctrl+W. OK, let's enter our first program, and so this is going to be a public class. And let's call this WarEagle -- all one word but notice we're using CamelCase.

And let's, let's call it WarEagle1, since we already have a WarEagle. And we have an open brace, and then skip a few lines, close brace. This is actually a syntactically correct class, and so it will compile, we need to save it. If you hit Save, it brings up the dialog, and notice the name of the file it's gonna use matches the class name, that's a requirement.

Transcript: First Program in jGRASP

So we're gonna say WarEagle1 and WarEagle1.java, that's the name of the file. So I save it, and now let's compile it. That's the green plus up here, when you when you make additions to a program, you need to recompile it. So think of addition and plus there, so I'm gonna click that button and notice it compiled OK.

And if I click the little run, it says, there's no main method. So, we need to add that main method we talked about earlier. So I'm going to tab over and say, "public static void main". And open parentheses, and it's got a parameter, and it's a string array.

Called args. And then another open brace. Skip a line or two, and close brace, and we have our main program, and we should be able to compile that. And it compiled with no problems. Now if we left out the open brace there and tried to compile it, we would have a compilation error.

It says here, expected a semicolon, that's not quite right there. But it was expecting something, so sometimes the compiler errors aren't really clear. But look what to see what mistake was likely made. So now, we can run this. And notice, it ran, but it didn't really do anything, that's because we haven't given it any statements in here, this is our algorithm.

So I'm going to key in our first print statement. And that's System.out.println. Open, and then we're going to print a string literal, War Eagle, and close the string literal with another quotes. String literals are enclosed in quotes. And then close parenthesis, semicolon. And let's see if that compiled, which it should do.

All right, now let's run it. And it printed War Eagle down below. We wanna print WarEagle three times, so I'm gonna hover here on the left margin, and I'm going to copy that line. Just insert it twice more and. Add an exclamation point or two, just to make the lines look different.

And this is exciting stuff. So, now let's let's compile. Actually, if we just click Run, you'll see down here that the compiled messages will pop to the top and then run I/O. Because jGRASP detects that the class file is out of date and recompiles. And then it runs, and there's our War Eagle three times.

Transcript: First Program in jGRASP

OK, so let's, let's explore some of the buttons on the toolbar. This is the toolbar up near the top, we've got a file, an Open File button, and then this is the Save button. The Browse button, the little file with the B on it, is a very useful button.

And what it does, wherever the file in focus is, it changes the Browse tab to that location. So, suppose I'm way up here, in our class notes, and I want to know where this file I'm working on is. I can just click the, that B button, and it takes me there, and the file is underlined.

When you're working on a file, you want that directory over here and you want to see the file you're working on underlined. That'll tell you it's really where you think it is. Next, we've got the Print button, and then we've got Generate Control Structure Diagram, that's this diagram here.

You can fold the program up and unfold it in layers, or unfold it all at once. This next button removes the CSD. I'm going to, under the View Menu, turn on Auto Generate. So, each time I load a program it'll automatically generate the CSD each time I compile.

It will regenerate it, in case I've added statements, it corrects the diagram, and so on. After we get the if statements and loops, it'll be a richer diagram and it'll have a good bit more meaning and become more useful. Next, we've got line numbers, you can toggle those on and off and this is a global setting.

So, all of your Windows will have line numbers if you turn them on. And, next is Documentation, this generates Documentation. We don't have any java.comments in this program, but let's open War Eagle from our examples. It's our same program, but we've added a java.comment before the class and then one before main.

And we'll wanna add java.comments for all of our classes and methods. And notice that it begins with a /**, each of these, and then ends with a */. Now, with these, we can actually generate documentation for this program, so let's do that. And here we see the, this is Class WarEagle, and it says here, WarEagle, my first program.

Transcript: First Program in jGRASP

And down here for main, it says Prints War Eagle 3 times. And if we click on main, it takes us down and here we see the details for main. Prints War Eagle 3 times, and it has a parameter, args although it's not used at this time. And then, here's our -- we've already done Compile, Run, here's the Debug button.

This button actually launches the debugger. And to use that, you just set a break point where you wanna stop, I'm gonna hover in the left margin. Set a break point, and then if I run it in Debug mode, it runs down to there and stops. And this highlighted line is the one that it's about to execute.

And so over in the Debug tab that's come to the top now, I'm gonna click the Step button. And you can see down below, we just printed War Eagle, and there it is again. Each time I step, it goes to a statement, and the last step actually ends the program.

One other button we wanna look at right now is the Checkstyle button. If you've got Checkstyle installed, you should see this little check up here. And what that does, that checks to see that your program is formatted correctly, and that you have these two java.comments. If I Run that, notice it says Starting audit, Audit done.

If I remove a java.comment, I'm just gonna take that star away, and this becomes a regular multi-line comment, but not a java.comment. And I Run Checkstyle, and see that I've got a missing java.comment, let's go back and put that asterisk back. Now when we run, we're OK. And if you do things like put spaces between a method, and an open parenthesis.

Or the argument, or maybe after, or before the semicolon. It hits you for those two things, too, but it says, it says parenthesis is preceded by whitespace. And so, pretty easy to correct that, and this one. And we're back in business. OK, I hope this brief introduction to our first program has been useful.

I think it's enough to get you going, so we'll stop here.