Third Java Tick
Alan Blackwell – February 2004

This tick comes from chapter 5 of the BlueJ book, exercises 5.2 – 5.44 (the **`tech-support1`** project).

Make your own copy of the project directory and open it in BlueJ. In the following exercises, your goal is to enhance the operation of the **`tech-support1`** system until it has the functionality of the **`tech-support-complete`** system that was demonstrated in one of the lectures. There is a copy of the complete solution in the PWF BlueJ directory. You can refer to this if you get into trouble, but you should try to do all of the exercises on this sheet without referring to the solution.

Investigate the Java API documentation for the **`String`** class, using the BlueJ Help/Java Class Libraries menu item. Follow exercises 5.2-5.6, answering questions about i) the **`startsWith`** method, ii) a method that tests whether a string ends with a given suffix, iii) a method that returns the number of characters in the string, and iv) the **`trim`** method.

Start your enhancements by changing the **`tech-support1`** system so that it is tolerant of leading spaces around the word "bye", and then testing this change (5.7 & 5.8). Improve the code of the **`SupportSystem`** class using the **`toLowerCase`** method, so that case in the input is ignored (5.9). Change your implementation to use the **`equals`** method instead of **`startsWith`** (5.10 & 5.11).

Investigate the library class **`Random`**, by studying the Java library documentation, and writing a new class called **`RandomTester`** to simulate the throwing of dice (5.12 - 5.16). Enhance **`tech-support1`** so that the **`getResponse`** method of the **`Responder`** class can select different responses randomly from an **`ArrayList`** or **`LinkedList`** of possible **`String`** responses (5.17 – 5.20).

Investigate the **`Map`** and **`HashMap`** classes in the API documentation (5.21 - 5.27). Note that you will find it hard to understand everything, because the documentation for these classes is not very good. Enhance **`tech-support1`** so that the responses generated are selected to be relevant to the user's query (5.28)

Investigate the HashSet and StringTokenizer classes (5.29 – 5.35) Use what you have learned to further enhance **`tech-support1`** as described in the exercises 5.36 – 5.40.

Use BlueJ's Generate Documentation function to generate documentation for your project. Examine the documentation. Is it accurate? Is it complete? Which parts are useful, which are not? Are there any errors? Complete the documentation using additional javadoc tags (called "key symbols" in the BlueJ book) (5.41 – 5.44). Note that the URL describing the Sun javadoc tool has changed since the BlueJ book was published, to:
`http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/javadoc.html`

For your tick assessment, print out the complete source code, and the javadoc documentation, for the **`Responder`** *class. You can print the documentation directly from a web browser. Unfortunately the "Print…" menu item in the BlueJ source code editor does not work under the PWF Linux java installation. Instead, you can print the source file for the* **`Responder`** *class with the following commands:*

```
cd tech-support1
lpr Responder.java
```

I also recommend completing exercises 5.45 – 5.55 (the **`balls`** project), to gain a better understanding of graphics, animation, class variables and constants. These exercises need not be submitted to the tickers.