More Sophisticated Behaviour

Technical Support System V1.0

Produced by:

Dr. Siobhán Drohan
Ms. Mairéad Meagher
Ms. Siobhán Roche

SE TU
Ollscoil Teicneolaíochta an Oirdheiscirt
South East Technological University

Department of Computing and Mathematics
http://www.setu.ie/

Lectures and Labs

- The Tech Support System lectures and labs are based on examples in Chapter 5 of:
  - Objects First with Java - A Practical Introduction using BlueJ, © David J. Barnes, Michael Kölling

CD-R
Objects First with
Java™
A Practical Introduction Using BlueJ
5th EDITION
vid Barnes | Michael Kölling

Topic List

1. Recap of Library Classes (**Java's API**).
2. **Interface Vs Implementation**.
3. Technical Support System **V1**:
   - Three Classes:
     1. **InputReader** class
     2. **Responder** class
     3. **SupportSystem** class

The Java class library (**API**)

- **API**: *Application Programmers' Interface*

- Thousands of **classes**.
- Tens of thousands of **methods**.
- Many useful classes that make life much easier.
- A competent Java programmer **must be able to work with the libraries**.
- Documentation of the Java libraries is in HTML format (generated using **javadoc** comments).
- Readable in a web browser.

Using **library classes**

- Classes from the library must be imported
  - using an ***import*** statement

```
import javax.swing.*;
```

```
import javax.swing.JOptionPane;
```

  - exception are classes from *java.lang*

- They can then be used like classes from the current project.

Working with the **library**

You should:

- know some important packages/classes by name.
- know how to find out about other classes.

Remember:

- We only need to know the **interface**,
  not the **implementation**.

- **API** contains the **interface description**
  for all library classes.

Topic List

1. Recap of Library Classes (**Java's API**).
2. **Interface Vs Implementation**.
3. Technical Support System **V1**:
   - Overview of the System
   - Three Classes:
     1. **InputReader** class
     2. **Responder** class
     3. **SupportSystem** class

Interface vs implementation

The documentation includes:

- Class name;
- Class description;
- List of constructors and methods
- Return values and parameters for constructors and methods
- Description of the purpose of each constructor and method

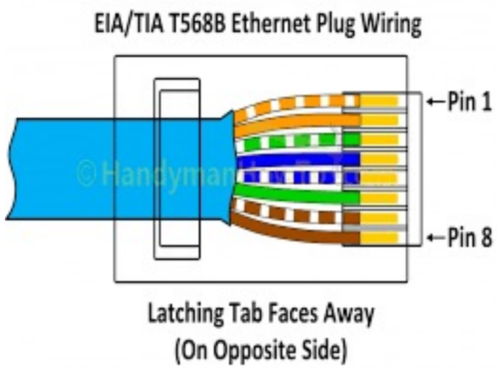
➡ **the *interface* of the class**

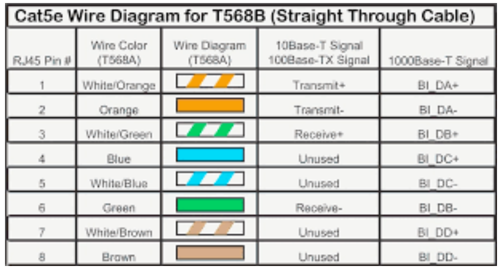
Interface vs **implementation**

*The documentation **does not** include*

- private fields
  - (most fields are private)
- private methods
- the bodies
  - (source code) for each method

➡ **the *implementation* of the class**





Cat5e Wire Diagram for T568B (Straight Through Cable)

RJ45 Pin #	Wire Color (T568A)	Wire Diagram (T568A)	10Base-T Signal 100Base-TX Signal	1000Base-T Signal
1	White/Orange	▨	Transmit+	BI_DA+
2	Orange	🟧	Transmit-	BI_DA-
3	White/Green	▨	Receive+	BI_DB+
4	Blue	🟦	Unused	BI_DC+
5	White/Blue	▨	Unused	BI_DC-
6	Green	🟩	Receive-	BI_DB-
7	White/Brown	▨	Unused	BI_DD+
8	Brown	🟫	Unused	BI_DD-

Topic List

1. Recap of Library Classes (**Java’s API**).
2. **Interface Vs Implementation**.
3. Technical Support System **V1**:
   - – Overview of the System
   - – Three Classes:
     1. **InputReader** class
     2. **Responder** class
     3. **SupportSystem** class

Technical Support System **V1**

01

Console based system.

02

Textual dialog system

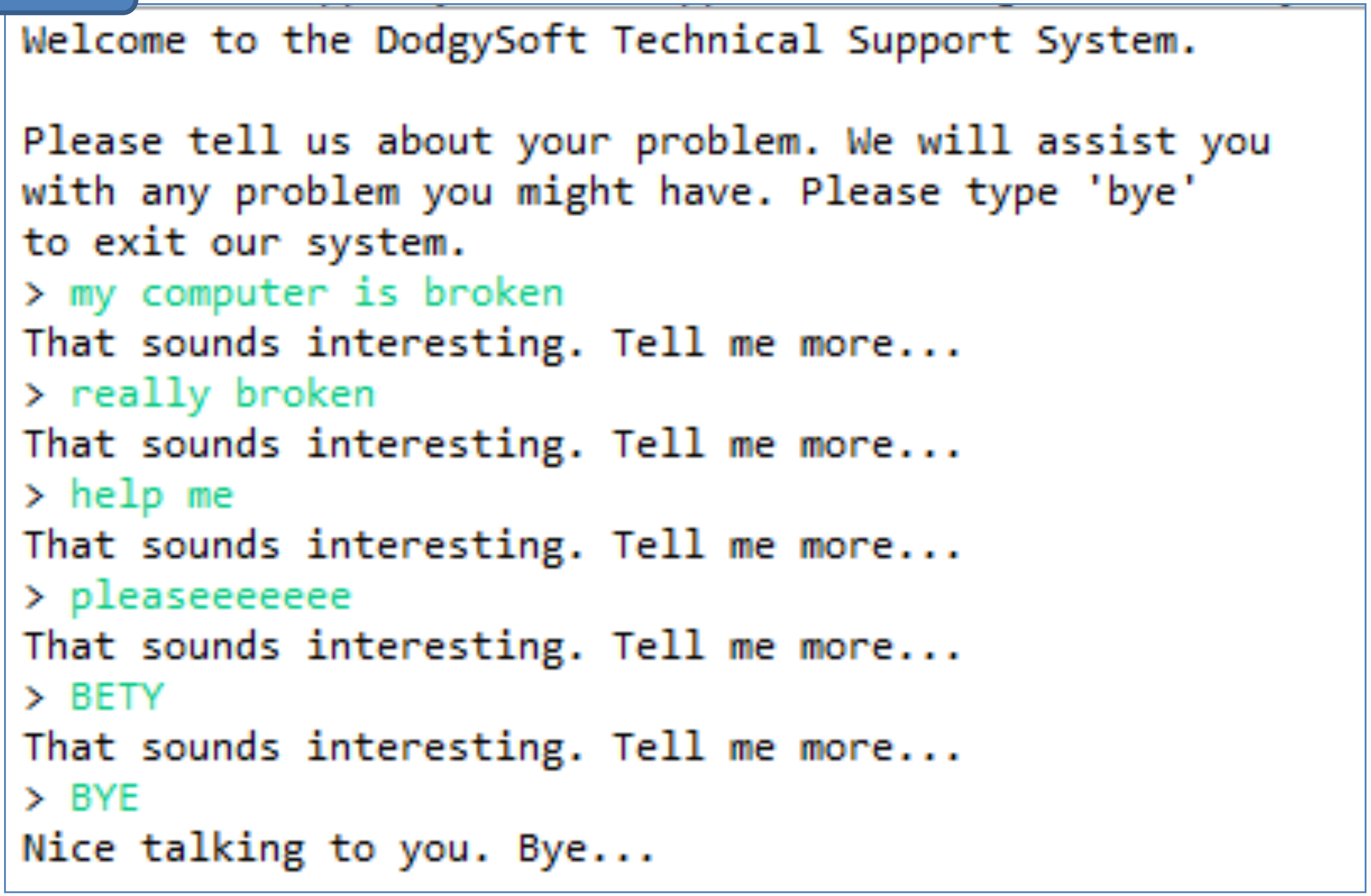
- you enter text on the console and the system will provide a response.

03

System always responds with the same String:

- *"That sounds interesting. Tell me more…"*

Technical Support System V1

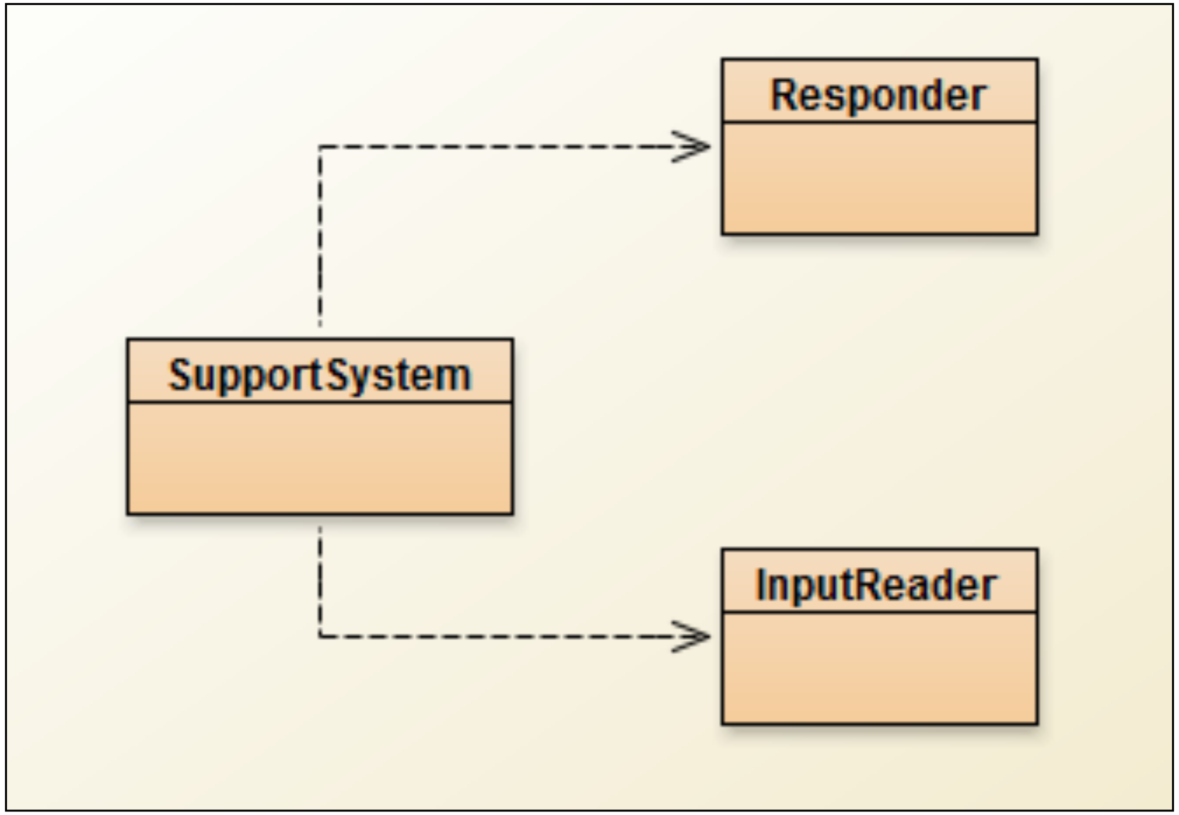


```
Welcome to the DodgySoft Technical Support System.

Please tell us about your problem. We will assist you
with any problem you might have. Please type 'bye'
to exit our system.
> my computer is broken
That sounds interesting. Tell me more...
> really broken
That sounds interesting. Tell me more...
> help me
That sounds interesting. Tell me more...
> pleaseeeeeee
That sounds interesting. Tell me more...
> BETY
That sounds interesting. Tell me more...
> BYE
Nice talking to you. Bye...
```

Topic List

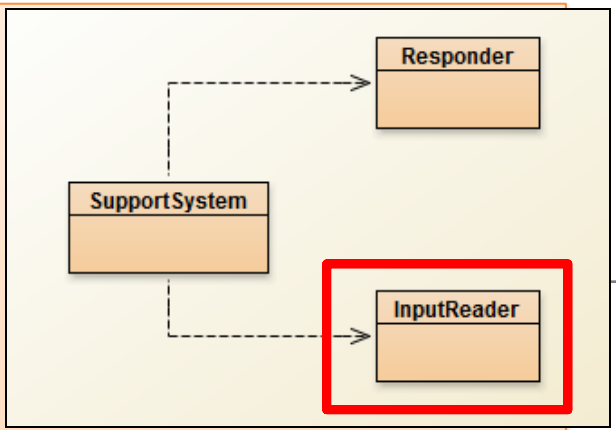
1. Recap of Library Classes (**Java's API**).
2. **Interface Vs Implementation**.
3. Technical Support System **V1**:
   – Overview of the System
   – Three Classes:
     1. **InputReader** class
     2. **Responder** class
     3. **SupportSystem** class

Class Diagram



Topic List

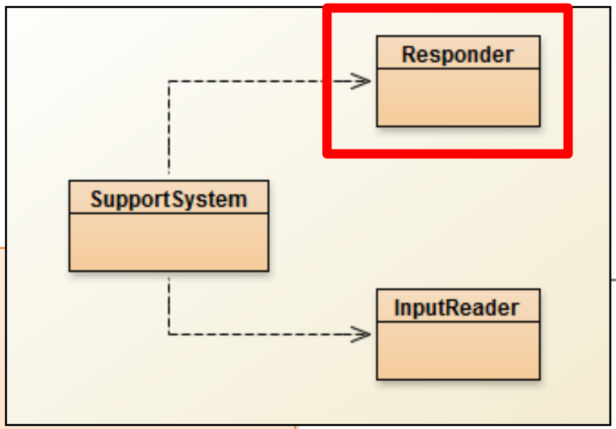
1. Recap of Library Classes (**Java's API**).
2. **Interface Vs Implementation**.
3. Technical Support System **V1**:
   - Overview of the System
   - Three Classes:
     1. **InputReader** class
     2. **Responder** class
     3. **SupportSystem** class



```
import java.util.Scanner;

public class InputReader{

    Scanner input;

    public InputReader(){
        input = new Scanner(System.in);
    }

    /**
     * Read a line of text from the console and return it as a String.
     *
     * @return  A String typed by the user.
     */
    public String getInput() {
        System.out.print("> ");              // print prompt
        String inputLine = input.nextLine().trim().toLowerCase();
        return inputLine;
    }
}
```

Topic List

1. Recap of Library Classes (**Java's API**).
2. **Interface Vs Implementation**.
3. Technical Support System **V1**:
   - Overview of the System
   - Three Classes:
     1. **InputReader** class
     2. **Responder** class
     3. **SupportSystem** class



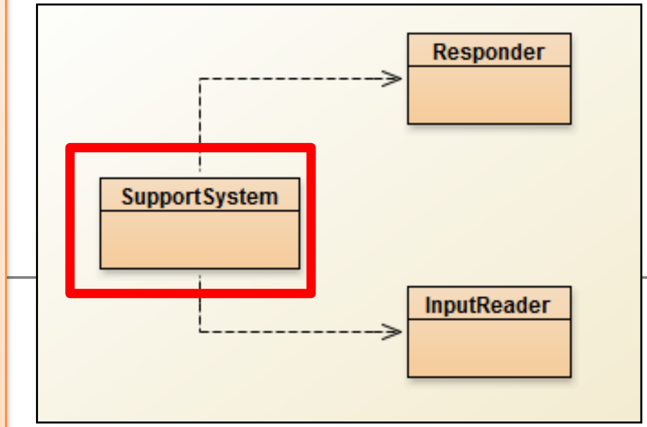
```
public class Responder{

  /**
   * Construct a Responder - nothing to do
   */
   public Responder(){
   }

   /**
   * Generate a response.
   * @return   A string that should be displayed as the response
   */
   public String generateResponse(){
       return "That sounds interesting. Tell me more...";
   }
}
```

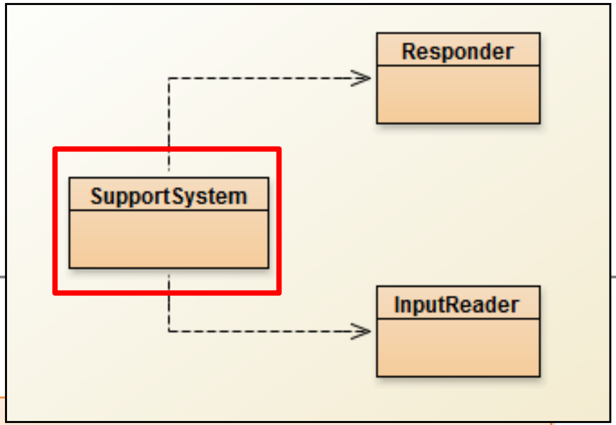
Topic List

1. Recap of Library Classes (**Java's API**).
2. **Interface Vs Implementation**.
3. Technical Support System **V1**:
   - Overview of the System
   - Three Classes:
     1. **InputReader** class
     2. **Responder** class
     3. **SupportSystem** class

```
public class SupportSystem{
    private InputReader reader;
    private Responder responder;

    public SupportSystem() {
        reader = new InputReader();
        responder = new Responder();
    }

    public static void main(String[] argvs){
        SupportSystem app = new SupportSystem();
        app.start();
    }

    public void start(){
         printWelcome();
         String input = reader.getInput();
         while(! input.startsWith("bye")) {
             String response = responder.generateResponse();
             System.out.println(response);
             input = reader.getInput();
         }
         printGoodbye();
    }
```

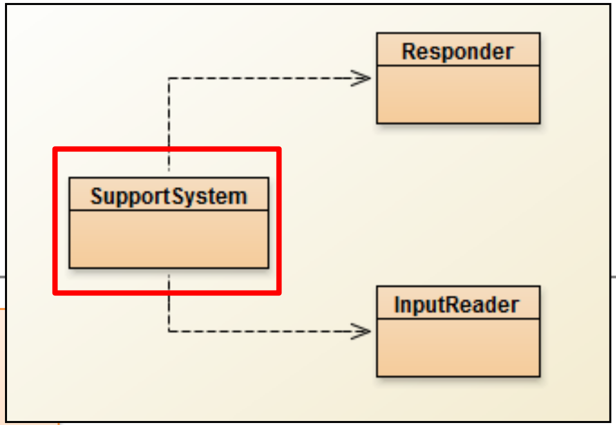
More on next slide >>





```
private void printWelcome(){
    System.out.println("Welcome to the DodgySoft Technical Support System.");
    System.out.println();
    System.out.println("Please tell us about your problem. We will assist you");
    System.out.println("with any problem you might have. Please type 'bye'");
    System.out.println("to exit our system.");
  }

  private void printGoodbye(){
    System.out.println("Nice talking to you. Bye...");
  }

}
```

Main loop structure



```
public void start(){
    printWelcome();
    String input = reader.getInput();
    while(! input.startsWith("bye")) {
        String response = responder.generateResponse();
        System.out.println(response);
        input = reader.getInput();
    }
    printGoodbye();
}
```

```
Get input
while(input does not start with “bye”){
    do something (i.e. print response)
    Get some new input
}
```

Any Questions?