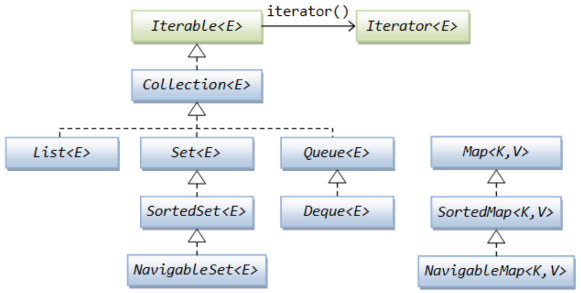
More Sophisticated Behaviour

Technical Support System V3.0

Produced by: Dr. Siobhán Drohan
Mr. Colm Dunphy
Mr. Diarmuid O'Connor
Dr. Frank Walsh

Java Collections Framework:



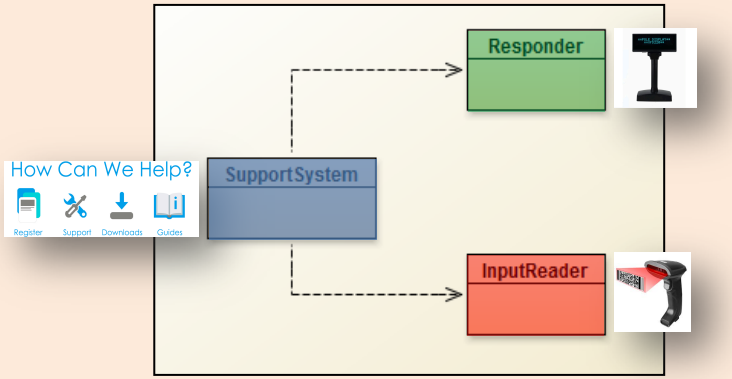
Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

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

Topic List

1. Recap: Technical Support System **V2**

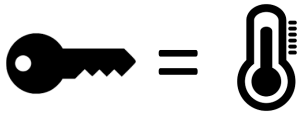
2. Technical Support System **V3**
   - Overview
     - 3 classes:
       - **Responder**
       - **InputReader**
       - **SupportSystem**



3. Class Development
   - Responder class
     - Generating a related response
     - ArrayList
     - Map and **HashMap**
   - InputReader class
     - Tokenizing Strings
     - Set and **HashSet**
   - Responder class
     - Finishing the class
   - SupportSystem class
     - A small change.

Tokenizing Strings

- We have a HashMap
  - containing a series of words with appropriate responses.

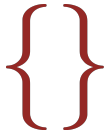


- Now we need to **search** the String of words the user entered on the console
  - to see if they typed in any of the words stored in the HashMap.

- We need to "split" the String of words entered by the user
  - into individual words
  - and store them in a collection
    - **Tokenizing Strings**.

- We need a new data structure to store these words just once

A **Set** stores **unique** values



Set {}

- A **Set** is a collection
  - that stores each individual element at most once
    - (i.e. unique elements).

- It does not maintain any specific order.

- The coding for **Set** is very similar to ArrayList coding.

Using **Sets**

```
import java.util.HashSet;
import java.util.Iterator;
...
HashSet<String> mySet = new HashSet<String>();

mySet.add("one");
mySet.add("two");
mySet.add("three");

Iterator<String> it = mySet.iterator();
while(it.hasNext()) {

    // call it.next() to get the next object
    // do something with that object

}
```

Compare this code to **ArrayList** code!

What is the **Difference** between **Set** and **List**?

List (e.g. ArrayList):

- keeps all elements entered in the desired **order**,
- provides access to elements by **index**
- can contain the **same element multiple times**.

Set (e.g. HashSet):

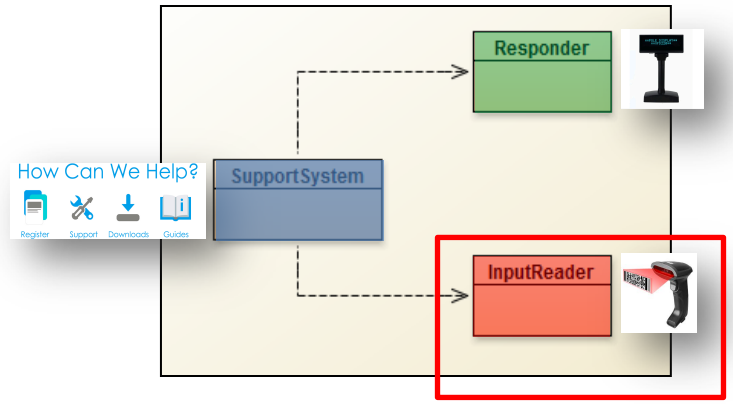
- **No specific order**
- ensures each element is in the set **at most once**
  - (entering an element a second time has no effect).

Returning to Tokenizing Strings

InputReader class

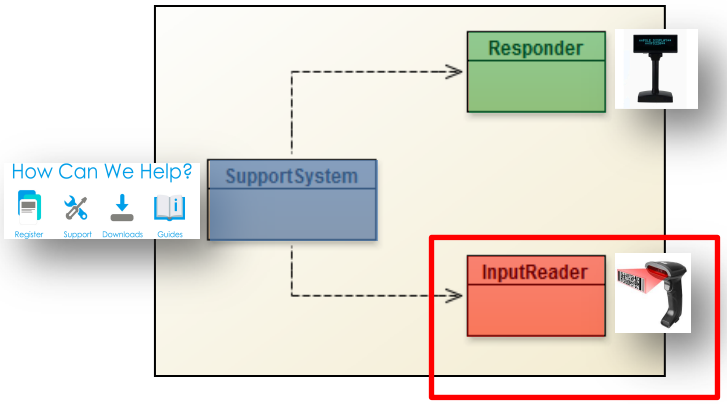
V2 Code

```
// V2 Code
import java.util.Scanner;

public class InputReader{

   Scanner input;

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

   /**
    * Read a line of text from standard input (the text terminal),
    * 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;
   }
}
```



In **V3** we modify the InputReader class to split out the input (stored in **inputLine**) into a primitive array of Strings...

```java
// V3 Code
import java.util.Scanner;

public class InputReader{

    Scanner input;

    public InputReader(){
      input = new Scanner(System.in);
    }
    public HashSet<String> getInput()
    {
       System.out.print("> ");              // print prompt
       String inputLine = input.nextLine().trim().toLowerCase();

       String[] wordArray = inputLine.split(" ");  // split at spaces

       // add words from array into hashset
       HashSet<String> words = new HashSet<String>();

       for (String word : wordArray) {
          words.add(word);
       }
       return words;
    }
}
```



V3 changes in InputReader class

1) Split up the **inputLine** object at spaces, storing each word in a **wordArray** of String[]

2) Declare & initialise **words** as a HashSet of String

3) For each **word** in the **wordArray**, add that **word** to the **words HashSet**

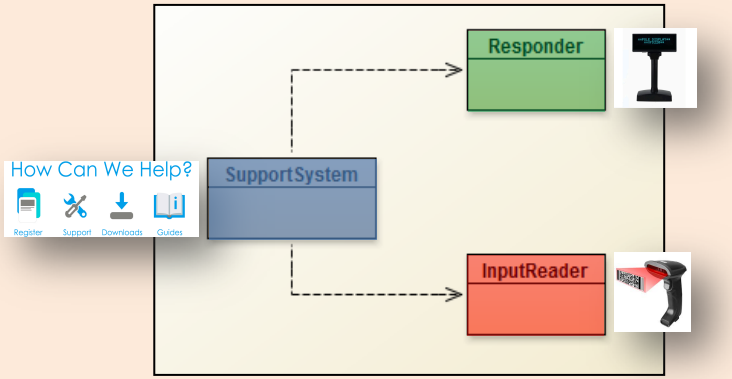
4) Return the HashSet of **words**

Topic List

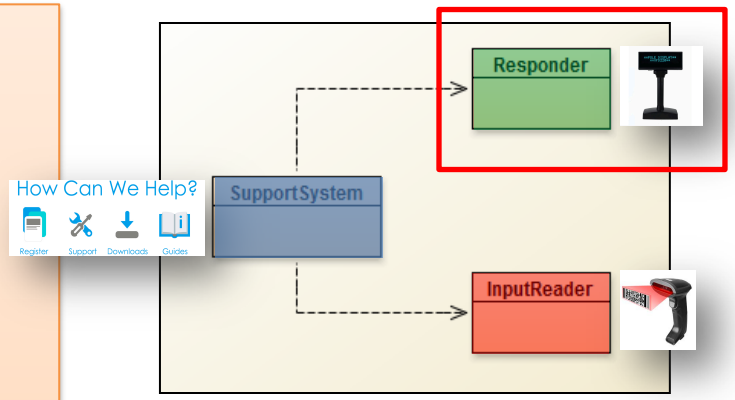
1. Recap: Technical Support System **V2**

2. Technical Support System **V3**
   - Overview
     - 3 classes:
       - **Responder**
       - **InputReader**
       - **SupportSystem**

3. Class Development
   - Responder class
     - Generating a related response
     - ArrayList
     - Map and **HashMap**
   - InputReader class
     - Tokenizing Strings
     - Set and **HashSet**
   - Responder class
     - Finishing the class
   - SupportSystem class
     - A small change.



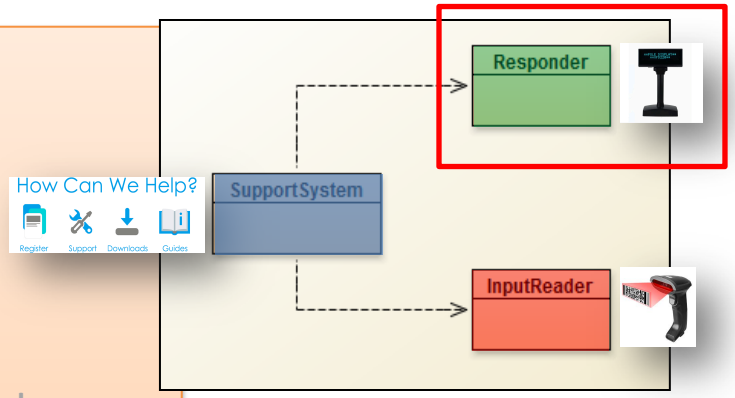
```
import java.util.HashMap;
import java.util.HashSet;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.Random;

public class Responder
{
    // Used to map key words to responses.
    private HashMap<String, String> responseMap;

    // Default responses to use if we don't recognise a word.
    private ArrayList<String> defaultResponses;

    private Random randomGenerator;

   public Responder()
    {
        responseMap = new HashMap<String, String>();
        fillResponseMap();
        defaultResponses = new ArrayList<String>();
        fillDefaultResponses();
        randomGenerator = new Random();
    }
```



V3.0 Responder Class

MORE changes (in red)
to handle a HashSet of Strings
passed into the
generateResponse() method.

```
public String generateResponse (HashSet<String> words)
  {
     Iterator<String> it = words.iterator();          // initialise an iterator called it

     while (it.hasNext()) {

        String word = it.next();                      // store the next key in the string word
        String response = responseMap.get(word); // Lookup the key in the Map
        if(response != null) {
           return response;                           // if found return the value of the key, else…
        }
     }
     // If we get here, none of the words from the input line were recognized.
     // In this case we pick one of our default responses (what we say when
     // we cannot think of anything else to say...)
     return pickDefaultResponse();
  }
```



V3.0 Responder Class

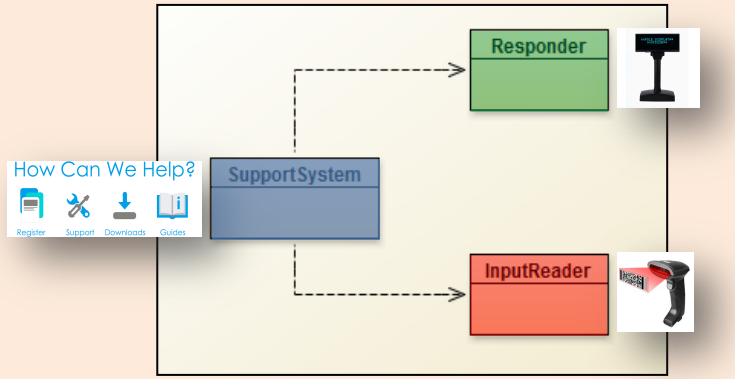
MORE changes (in red)
to handle a HashSet of Strings
passed into the
generateResponse() method.

Topic List

1. Recap: Technical Support System **V2**

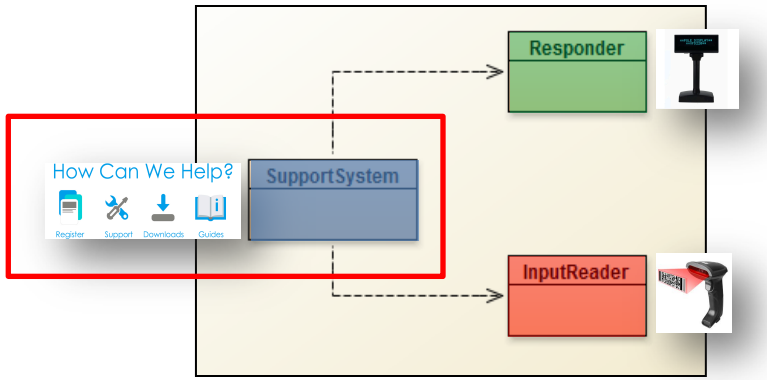
2. Technical Support System **V3**
   - Overview
     - 3 classes:
       - **Responder**
       - **InputReader**
       - **SupportSystem**

3. Class Development
   - Responder class
     - Generating a related response
     - ArrayList
     - Map and **HashMap**
   - InputReader class
     - Tokenizing Strings
     - Set and **HashSet**
   - Responder class
     - Finishing the class
   - SupportSystem class
     - A small change.



V2 Code

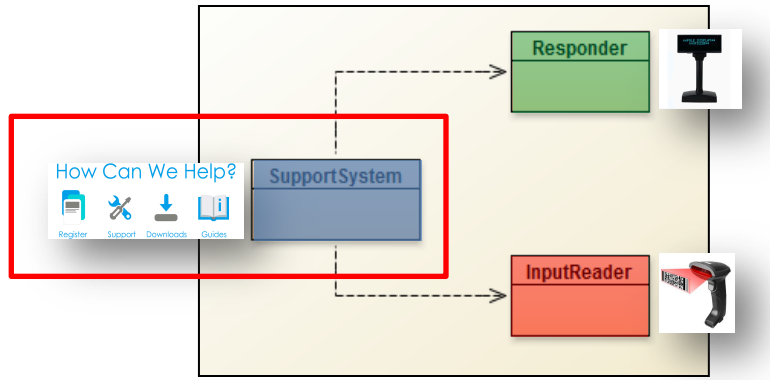
```
// V2 code
public class SupportSystem
{
    private InputReader reader;
    private Responder responder;

    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();
    }
```



In **V3**,
we change SupportSystem class,
mainly in the start() method…

V3 Code

```
import java.util.HashSet;
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.startSupport();
    }

public void startSupport(){
        printWelcome();
        HashSet<String> input = reader.getInput();
        while(!input.contains("bye")) {
                String response = responder.generateResponse(input);
                System.out.println(response);
                input = reader.getInput();
        }
        printGoodbye();
    }
```



V3

Uses a **HashSet of Strings** called **input** which is passed to **generateResponse()**

startSupport() replaces **start()**

Any Questions?