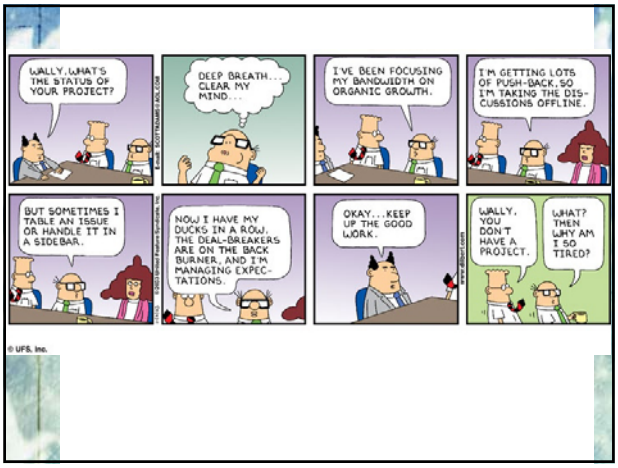
Grouping Objects 1

Introduction to Collections



Style

- variable names *(should be meaningful)*
- comments *(tell us information that is not obvious from the code)*
- layout *(indentation should reflect logical structure)*

`http://www.bluej.org/objects-first/styleguide.html`

Linked from the "Resources" on the module web page.

Main concepts to be covered

- Collections (especially: ArrayList)

Requirement to Group Objects

- Many applications involve *collections* of objects:
  - Personal organizers
  - Library catalogs
  - Student-record system
- The number of items stored varies:
  - Items get added
  - Items get deleted

Example: a personal notebook

- Notes may be stored.
- Individual notes can be viewed.
- There is no limit to the number of notes.
- It can report how many notes are stored.
- Explore the *notebook1* project.

Class Libraries

- Libraries of useful classes.
- We don't have to write everything from scratch.
- Java calls its libraries, *packages*.
- *Collections* of objects is a recurring requirement.
  - The **`java.util`** package contains classes for doing this.


```
import java.util.ArrayList;

/**
 * ...
 */
public class Notebook
{
    // Storage for an arbitrary number of notes.
    private ArrayList<String> notes;

    /**
     * Perform any initialization required for the
     * notebook.
     */
    public Notebook()
    {
        notes = new ArrayList<String>();
    }

    ...
}
```

A class within the **`java.util`** package

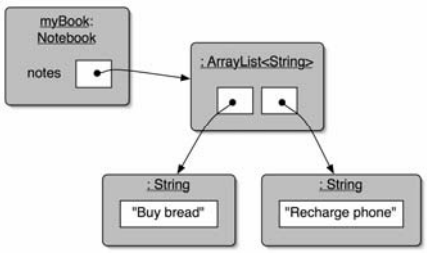
Type argument, supplied to **`Arraylist`**

No arguments needed for this **`Arraylist<String>`** constructor

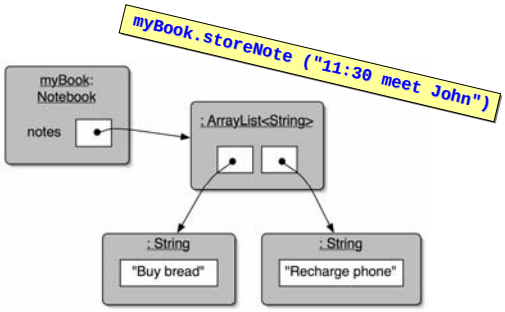
Collections

- We specify:
  - the type of collection: **`ArrayList`**
  - the type of objects it will contain: **`<String>`**
- We say: *"ArrayList of String"*

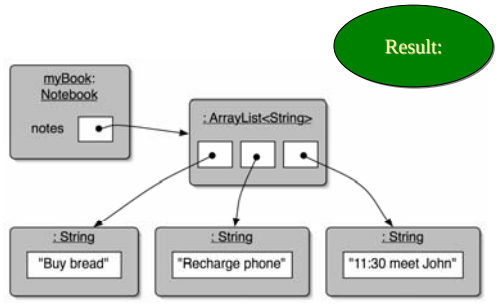
Object Structures with Collections



Adding a Third Note



Adding a Third Note



Features of the Collection

- It increases its capacity as necessary.
- It keeps a private count … which we can get using the **`size()`** accessor.
- It keeps the objects in the order they are added.
- Details of how all this is done are hidden:
  - Does that matter? Does not knowing how prevent us from using it?

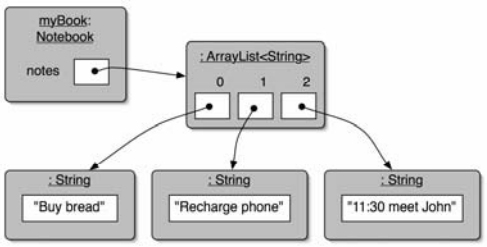
Using the Collection

```
public class Notebook
{
    private ArrayList<String> notes;
    ...

    public void storeNote(String note)
    {
        notes.add(note);
    }

    public int numberOfNotes()
    {
        return notes.size();
    }

    ...
}
```

Adding a new note (*delegation*)

Returning the number of notes (*delegation*)

Index Numbering



Retrieving an Object

```
public void showNote (int noteNumber)
{
    if (noteNumber < 0) {
        // This is not a valid note number, do nothing.
    }
    else if (noteNumber < numberOfNotes()) {
        // This is a valid note number, print it.
        System.out.println (notes.get(noteNumber));
    }
    else {
        // This is not a valid note number, do nothing.
    }
}
```

Index validity checks

Retrieve the note

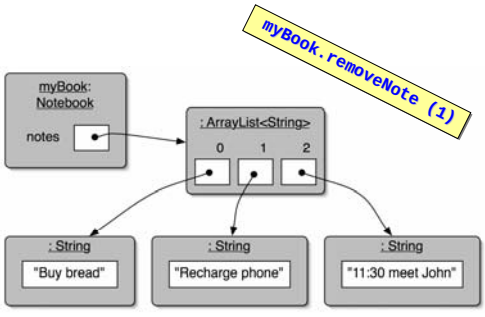
Removing an Object

```
public void removeNote (int noteNumber)
{
    if (noteNumber < 0) {
        // This is not a valid note number, do nothing.
    }
    else if (noteNumber < numberOfNotes()) {
        // This is a valid note number, print it.
        notes.remove(noteNumber));
    }
    else {
        // This is not a valid note number, do nothing.
    }
}
```

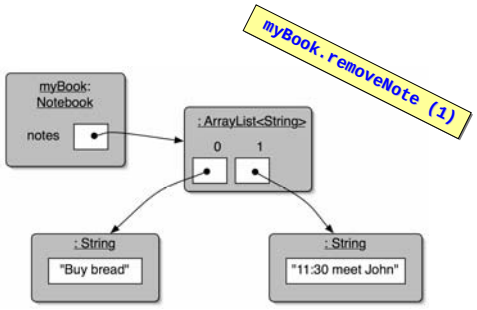
Index validity checks

Remove the note

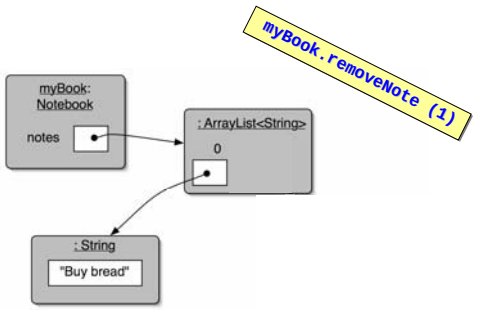
Index Numbering



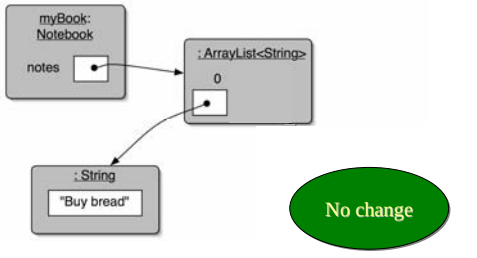
Removal affects Numbering



Removal affects Numbering



Removal affects Numbering



Generic Classes

- Collection classes are examples of *parameterised* or *generic* classes (types).
- `ArrayList` implements list functionality:
  - it has methods `add`, `get`, `size`, etc.
- The *<type>* parameter says what kind of data we want the list to hold:
  - `ArrayList<Person>`
  - `ArrayList<TicketMachine>`
  - etc.

Collection Classes – Review

- *Collections* allow an arbitrary number of objects to be stored.
- Java's class libraries contain tried-and-tested *collection classes*.
- Java's class libraries are called *packages*.
- We have used the `ArrayList` class from the `java.util` package.

Collection Classes – Review

- Items may be added and removed.
- Each item has an index.
- Index values may change if items are removed (or further items added).
- The main `ArrayList` methods are `add`, `get`, `remove` and `size`.
- `ArrayList` is a *parameterized* or *generic* type.