CS12420 Design then Code with BlueJ 2011-2012

Small Group project – CS12420

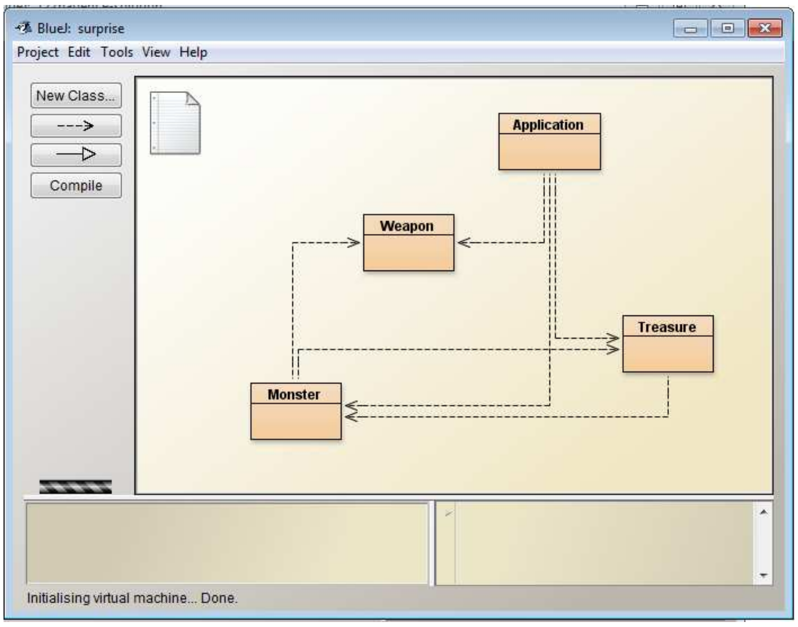
This is due on Friday March 9th by 6pm on Blackboard
and is worth 20% of your final mark for the module.

This assignment asks you to build a GUI that allows the user to create a class diagram suitable for a beginning java program, and then from it auto-generate simple Java code.

This code could then be loaded into BlueJ as a basis on which to code a program. The GUI should be built directly using Swing, you can use an IDE to help do your GUI for other assignments.

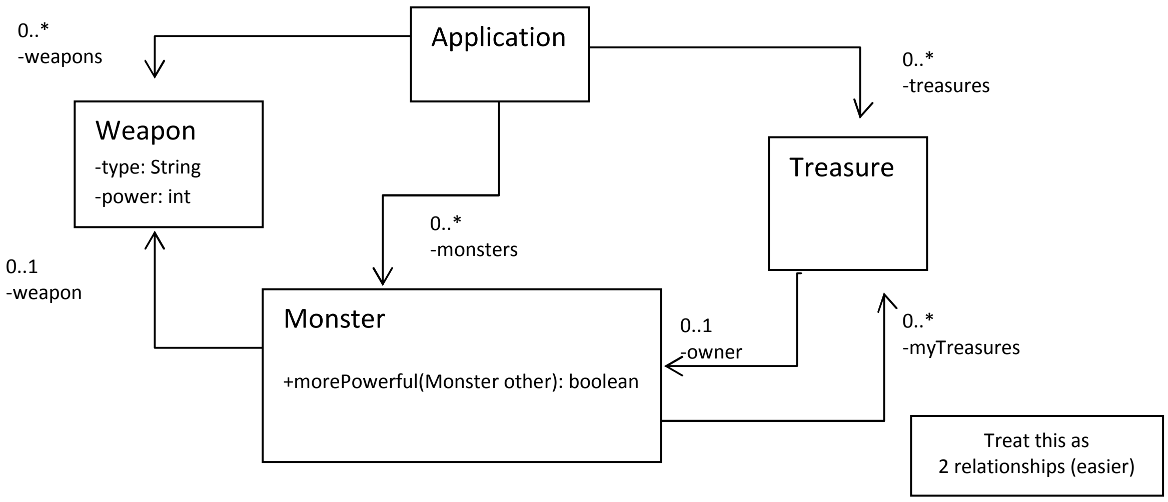
Basic Requirements

Those of you who took CS122 used a simple IDE called BlueJ. Here is BlueJ displaying some classes and links between them:



The trouble is that from the point of view of someone teaching this to beginners, the 'uses' arrows that are shown do not encode the depth of information needed.

I would like students to be able to produce a design that looks like this:



And then automatically generate the appropriate 4 .java files with linking variables in them. These files could then be uploaded to BlueJ using the 'Open non-BlueJ project' option. My hope is that this will help students write reasonable code while encouraging them to design first. (I may hope in vain).

The above would generate skeleton classes such as:

Monster.java

```
import java.util.*;

public class Monster{

 private ArrayList<Treasure> treasures;
 private Weapon weapon;

 public Weapon getWeapon(){
 }

 public setWeapon(Weapon w) {
 }

 public addTreasure(Treasure t) {
 }

 public boolean morePowerfulThan(Monster other){
 }

}
```

Weapon.java

```
import java.util.*;

public class Weapon{
  private String type;
  private int power;




}
```

With code that:

- translates '-' to private and '+' to public as shown,
- brings across any textual information (no need to check its validity since that can happen in BlueJ),
- implements the 0..* links as ArrayLists (I have doubts about this but it is simplest) and the 0..1 links as linking variables with add, sets and gets as appropriate.

This assignment asks you to produce a piece of software that does that.

- In other words there should be **at least** a menu, one Panel that holds buttons and another Panel where the design is drawn.
- The buttons should:
    - o allow the user to create a new class and place on the screen (easiest way to do that is to click on a position on the screen, or even type in the coordinates)
    - o allow the user to create new relationships between classes (the simplest way to do this is probably to left click on one class and right click on the other)
    - o allow the user to generate source code
    - o allow the user to clear the design
- There must be a way of entering the text into your 'classes' like I have done above with Weapon – the simplest is to pop up a dialog box:
```
  String text = JOptionPane.showInputDialog(null,
         "Enter text here", "Enter text here",OptionPane.QUESTION_MESSAGE);
```
  Likewise, clicking on one end of a relationship should pop up a way of entering the cardinality which should then be displayed on the screen appropriately
- One menu could allow you to quit, and to save designs and to load them; and another menu could duplicate the button behaviour.

The above are the functional requirements – as a non-functional requirement:

- You should make your final product fairly BlueJ-ish looking.

There is a lot of room for creativity here.

- You could have menu options that:
  - Print
  - Undo …..
- You could also:
  - Enter the text directly.
  - Select classes and move them and resize them and …….

Deliverables

1. One person in your group needs to create a single ZIP file and submit it on Blackboard by 6pm on Friday 9th March 2012. Use the Blackboard submission form for *Assignment 1 – Design then BlueJ* in the Assignments section of the CS12420 module.
   The ZIP file should include the following:

   - **A single supporting document, as a PDF of no more than 12 sides that includes the following. See [1] for suggestions on writing good documentation.**
     - Preliminaries - **all** your names, and **very clear** instructions on how I can run your program **from the command line.**
     - Analysis - problem definition (including use-case diagram),
     - Design - including class diagrams and also class descriptions, and a discussion of the algorithms that allow each use case to be implemented.
     - Testing - testing strategy, testing table and testing results (including screen shots).

   - **The Assignment Feedback Form [2] (modify it for group handin)**
     - On the feedback form give your group a realistic mark out of 100 for your work and explain. Say what you found hard or easy, what was omitted and why, how you worked together and what you learned from that. This is an evaluation for all of you and the group product and should include a short paragraph from each of you on your role.
     - **Without evaluation the project will not be marked.**
     - **If you wish to say something privately about the group process email ltt@aber.ac.uk directly**.

   - **Also submit all your Java code files as a ZIP file (a .zip file, NOT a .rar or anything else)**

     TEST THIS! We always run all programs so if yours doesn't work you are shooting yourself in the foot for want of 20 minutes work.

   Please upload well before the deadline, since uploads sometimes fail, servers may become unavailable and uploads may take longer than anticipated.
   Note that submissions that are made after 6pm will be marked as late. If you submit after 6pm, you must complete a Late Assignment Form and hand it to the Computer Science Reception. Your late submission will be dealt with according to the department rules specified in section 5.6 of the Student Handbook.

2. In your tutorial in the week of March 19th you will do a short (10 minutes maximum) presentation of your project. This should include a couple of slides that illustrate your approach and your design and a demonstration of the project working.

If all your accounts are locked then you will not be able to submit via Blackboard. In this case, you must email your document to ltt@aber.ac.uk before the deadline. If you do not have any access to email other than your University account you must burn a CD containing all of the materials and hand it in to Computer Science Reception and obtain a receipt before the deadline. If you must submit on CD, you will need to submit before the reception closes at 4.30pm.

Assessment

Marking will be in line with the usual assessment criteria for project work, as described at [4].

• Quality of code including variable names, indentation and Javadoc commenting	5%
• Degree and correctness of the basic implementation	40%
• Design and analysis	15%
• Testing	15%
• Presentation (in tutorial week of March 19th)	5%
• WOW marks!	20%

This is worth 20% of your final mark for the module.

Plagiarism: Cases of plagiarism in student assignments are not taken lightly by the department, and the consequences of plagiarism by students can be severe. Please ensure that you understand what constitutes plagiarism, read the plagiarism notice from the Student Handbook [3].

You do not need to submit a paper declaration of originality form if you are submitting via Blackboard. However, you need to read the equivalent statement on the Blackboard submission page. By submitting to Blackboard you are confirming acceptance of the statement that this is your own work. If your account is locked and you have to submit via CD, you must also complete the statement of originality on the Anonymous Marking Sheet.

Help

Obviously you will have to start cracking on this assignment very soon. There are really two parts:

- The GUI
  Part of your group can get started building a prototype that simply prints messages when the buttons are pressed. That way every group should have a GUI.
- The Model
  How are you going to store the information about a class? About a relationship? You need to discuss this as a group, but then one part of your group can write the code for storing those things. You can test this code separately from the GUI.

Once you have these basics done, you should approach the overall problem. I would probably go about it in this order:

- New class
- Getting text into classes
- Generating the .java files for a basic class (version 1)
- New relationship (just the lines)
- Cardinalities for relationships
- Generating the .java files so they model the relationship (version 2)
- Saving and loading and clearing
- WOW

The advisors will be able to help you but you have to think carefully about the problem first. I am also happy to help.

Make sure that everyone helps with the coding. It is crucial that you all get practice of coding or you won't learn anything!

References

[1] L. Thomas, “Suggestions for Good Documentation” [Online] Available at: *http://www.aber.ac.uk/~dcswww/Dept/Teaching/CourseNotes/2011-2012/CS12230/Lassign/Suggestions%20for%20good%20documentation.pdf (Accessed: 10th February 2012).*

[2] Computer Science Department, “Resources” [Online] Available at: http://www.aber.ac.uk/~dcswww/intranet/staff-students-internal/teaching/resources.php (Accessed 10th February 2012).

[3] Computer Science Department, “Undergraduate Student Handbook 2011-2012” (1 August 2011) Version 11.0 [Online] Available at: http://www.aber.ac.uk/~dcswww/Dept/Teaching/Handbook/ (Accessed: 10th February 2012)

[4] Computer Science Department, “Appendix AA. Assessment Criteria for Development” (11 May 2002) Version 1.2 [Online] Available at: http://www.aber.ac.uk/~dcswww/Dept/Teaching/Handbook/AppendixAA.pdf (Accessed: 10th February 2012).