

Appendix

Teaching Portfolio

I have included a sample of course materials, preparations, and evaluations from my instruction of Object-Oriented Programming with Java.

- Syllabus
- Example lesson plan
- A sample programming assignment (#4)
- Final Exam
- A sample of my evaluation that was returned to students in group 2 for the final project
- A sample of unedited student evaluations (written) of the course

CISC370: Object-Oriented Programming with Java



Summer 2008

Instructor: James Atlas

Email: atlas at cis dot udel dot edu

Office: 047 Memorial Hall

Office Hours: Tuesday, Thursday 3:30 - 4:30 p.m. in 047 Memorial Hall

Appointments: if you can't make office hours, email me for an appointment.

Teaching Assistant: TBD

Email: TBD

Office: 047 Memorial Hall

Office Hours: TBD

Meeting Times: TR 5:00 - 7:00 p.m. (QDH 024)

Course Web Page: <http://www.cis.udel.edu/~atlas/cisc370>

Course Manager: We will be using the **UD MyCourses website** (<http://www.udel.edu/mycourses>) to manage submissions and grades.

Project Number:

Description

By the end of this course, students should be able to implement data structures and algorithms discussed in previous courses using Java, leverage the rich Application Programming Interface (API) of Java to solve business-scale problems, and contribute to a software development team using common tools and techniques. Roughly half of the course is devoted to the first objective. The second half of the course introduces students to large-scale development tools, practices, and Java APIs.

Optional Textbooks

- **Recommended:** *Head First Java, 2nd Edition*. Sierra & Bates; O'Reilly Media, Inc. 2005. ISBN 0596009208
- **Not recommended:** *Java How to Program, Seventh Edition*. Deitel & Deitel; Prentice Hall 2007. ISBN 0132222205

Course Topics

We will be covering several core areas of object oriented programming in Java as well as some tools and concepts useful for large scale development.

Java language-specific:

- Java basics
- Syntax, primitive types, control flow, naming conventions
- Basic OOP syntax, object-oriented design
- OOP in Java: Inheritance, Interfaces, Polymorphism
- Advanced classes: inner classes and anonymous classes
- Packages
- Exceptions
- Javadoc
- Java Input/Output
- Serialization
- Java Collections framework
- The Swing toolkit
- Network programming
- Java Database Connectivity (JDBC)
- Multithreaded programming
- Reflection API

Tools and development concepts:

- Software design patterns
- Development Environments (Eclipse)
- Ant
- Subversion
- JUnit
- Mantis

And last but not least... teamwork! Yes, your second project will involve working with other people.

Grading

Scale:

Number	100-93	93-90	90-87	87-83	83-80	80-77	77-73	73-70	70-67	67-63	63-60	<60
Letter	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F

Weights:

	Percent of grade
Programming Assignments	40
Weekly Quizzes	10
2 Projects	30 (15 each)
Final	20
Total	100

Programming Assignments

Submitted work must be handed in to the instructor at the *beginning* of class and submitted to UD's MyCourses. Assignments that are late are assessed a **10% per day late penalty; after seven days they will not be accepted**. No special provisions for the weekend. This policy is necessary because late assignments are burdensome for the TA in terms of separate handling and grading time. Grading will be based primarily on the written copy; a 10% penalty will be enforced if a written copy is not submitted.

See the web site for more information about **submission standards** for assignments and **coding style**.

If you have a disability or circumstance that requires special accommodation, please contact me by email (atlas at cis dot udel dot edu) during the first week of class.

Weekly quizzes

At the beginning of every Tuesday lecture, we will have a 10-minute quiz on the last week's material. Review your notes and understand the assignments, and you should have no trouble with the quizzes.

Projects

You will have two substantial coding projects in this class. The two programming projects are intended for the student to study aspects of Java in depth. The first programming project will be assigned on June 24th and due on July 17th. Each student will work individually and have the same task for this project. The second project will be a team project, due August 12th. We will evaluate your projects

based on a scripted copy of your project in action. You will also do a demo of the second project for the class. During the demo, you should provide details about your design decisions. The first programming project will be accepted up to one week late with a penalty of 25%. The second programming project will not be accepted late.

Final Exam

The final exam will be a cumulative test of your knowledge and understanding of Java. You will not need to write any code for the exam, but you will be expected to recognize concepts represented in code. A section of the exam will also cover Java software development applications and technologies taught in this course.

Academic Honesty

I expect you to observe the highest ethical standards, avoiding even the perception of ethical compromise. You are expected to do your own work unless explicitly instructed otherwise. This includes homework assignments, programming projects, quizzes, and examinations. All violations of academic honesty will be handled according to University policy.

Policy for Grading Concerns

The instructor and TA will endeavor to get all assignments back to students within ten days of the submission date. If this date is not met, please bring it to the attention of the instructor.

All students have the right to know how their grades are calculated, and if any student believes a mistake has been made, it is up to the student to contact the grader to discuss it within ONE WEEK of the return of the assignment. Contact the TA first for homework and projects. If you are not satisfied after discussing the grade with the TA, then you may bring it to the instructor. Bring exams directly to the instructor.

The grade percentages are on this syllabus. Please use them to calculate estimates of your semester grade. This class typically has little or no curve.

Sections of this syllabus and site were borrowed from Robert Duvall, Terry Harvey, Jeffrey Six, Keith Trnka, and Sara Sprenkle.

Course: CISC370

Lesson plan: 07/10/2008

Goals

- Knowledge of core software concepts and APIs used to handle network programming
- Knowledge of web programming models used in HTTP communication
- Students should be able to implement basic client-server architectures.

Objectives

- Students will create a network connection during the mini-lab exercise and communicate over LAN with other students.
- Simple listening server will echo successful connection of student to class network on overhead.

Prerequisites

- Threading
- Stream processing

Materials

- Computer connected to network

Level

- Knowledge – networking programming APIs
- Application – use API to create simple client-server
- Analysis – performance discussion for multi-threading, comparison of connection and connectionless services

Lesson Procedure

1. Introduction – Distributed Programming models, network components
2. Discussion – connectionless vs connection oriented models
3. Example – simple echo server
4. Discussion – concurrent connections/multi-threading
5. Activity(Mini-lab) – simple classroom multi-cast server echo
6. Introduction – Web Programming model
7. Discussion – server side processing languages/models
8. Activity(Mini-lab) – Read/post content to university website

Follow up Lessons/Activities

- Assignment 4: Multi-threaded Web Server

Assignment 4 (100 Pts): Multi-threaded Web Server

Due: Thursday, July 24

The goal of this assignment is to build a functional HTTP/1.0 server. This assignment will teach you the basics of distributed programming, client/server structures, and issues in building high performance servers.

At a high level, a web server listens for connections on a socket (bound to a specific port on a host machine). Clients connect to this socket and use a simple text-based protocol to retrieve files from the server. For example, you might try the following command from a UNIX machine:

```
% telnet www.cis.udel.edu 80
GET / HTTP/1.0\n
\n
```

(type two carriage returns after the "GET" command). This will return to you (on the command line) the html representing the "front page" of the UD computer science web page.

You can start from the basic web server code we looked at in class: **WebServerExample.java**

One of the key things to keep in mind in building your web server is that the server is translating relative filenames (such as index.html) to absolute filenames in a local filesystem. For example, you might decide to keep all the files for your server in

```
~student/cisc370/server/files/, which we call the root. When your server gets a request for /index.html, it will prepend the root to the specified file and determine if the file exists. If the file does not exist, a file not found error is returned. (Look up what the official HTTP error return codes are.) Otherwise, an HTTP OK message is returned along with the contents of a file.
```

You should also note that web servers typically translate `GET /` to `GET /index.html`. That is, index.html or index.htm is assumed to be the filename if no explicit filename is present for any directory.

When you type a URL into a web browser, it will retrieve the contents of the file. If the file is of type text/html, it will parse the html for embedded links (such as images) and then make separate connections to the web server to retrieve the embedded files. If a web page contains 4 images, a total of five separate connections will be made to the web server to retrieve the html and the four image files. Note that the previous discussion assumes the HTTP/1.0 protocol, which is what you will be supporting in this assignment.

For this assignment, you will need to support enough of the HTTP protocol to allow an existing web browser (Firefox, Netscape or IE) to connect to your web server and retrieve these types of documents with the appropriate MIME content type:

```
text/html (.html, .htm)
text/plain (.txt)
image/jpg (.jpg)
image/gif (.gif)
image/png (.png)
```

Everything else can be retrieved as application/octet-stream. You can determine the type of the file based on its extension.

At a high level, your web server will be structured something like the following:

```
Forever loop:
  Listen for connections
  Accept new connection from incoming client
  Parse HTTP/1.0 request
  Ensure well-formed request (return error otherwise)
  Determine if target file exists (return error otherwise)
  Transmit contents of file to client (by performing reads on the
file and writes on the socket)
  Close the connection
```

Handling Threads

In class, we discussed some of the potential performance issues you should consider when designing a multithreaded web server. You should implement the "basic" thread-handling mechanisms we discussed--initializing a pool of threads at startup and restricting the number of connections waiting to enter the application. You can use the `java.util.concurrent.ThreadPoolExecutor` for this.

Server Status Page

You will need to create a special web page that can display the status of your web server. This can be done with a static page location reference such as: `http://yourserver/status`. The status of your server should include the total number of connections made, the total bytes transmitted (i.e. keep a counter that does sums the `File.length()` of each retrieved file), and the current number of connections.

Testing

- Extra Credit (20 points): You may want to build a multithreaded client test program that can "fling" requests at your server to test your server's synchronization.

If you have any questions about submission, ask *early!*

Grading (100 pts)

- Correctness (does the application work and provide all required functionality: threading, synchronization, file types, status page): 70 pts
 - Organization (design, separation of concerns/functionality, inheritance relationships): 10 pts
 - Testing: 10 pts
 - Documentation (completeness, correctness, explanation for design choices): 10 pts
-

CISC 370 Final

Name: _____

08/14/2008

- Answer all questions, and only answer the question. No answer should require more than 4-5 full sentences. Please be concise.
- If you need more space, you can continue your answer on the back of a page. If you do move to the back, please clearly mark what question you're answering.
- If something is unclear in a question, then ask.
- I will post your final exam grade on udel.edu/MyCourses when I have completed the grading.

Part A (30) + Part B (40) + Part C (30) = Total (100)

A. Very Short Answer (30 pts)

A1. (3 pts) What is the keyword used to include use of other Java classes in your program?

A2. (3 pts) What value is returned by: `return 1 / (double) 2;`

A3. (3 pts) Can you change the character values of a String? If so, state how to do it.

A4. (3 pts) Write a valid constructor signature for a Name class with two String properties, first and last.

A5. (3 pts) What is the term for a class that is defined within another class?

A6. (3 pts) Define encapsulation.

A7. (3 pts) Name 2 Java language constructs that promote polymorphic behavior.

A8. (3 pts) Can two object references refer to the same location in memory? If so, how could you copy the object in memory to provide a different but similar valued object for each reference?

A9. (3 pts) Does Java support multiple inheritance? If so, show an example of how to declare a President class as inheriting from both an ElectedPolitician and a MilitaryCommander. If not, show an alternate way to express the dual roles of a President.

A10. (3 pts) What is the main difference between a checked and an unchecked exception?

B. Short Answer (40 pts)

B1. (10 pts) Name two methods that every object in Java has. How does Java guarantee that every class has these methods?

B2. (10 pts) What is the benefit of having generic `Collections` that can contain different types of objects? What is the benefit of declaring a type for the objects within a `Collection`, i.e. `List<String>`?

B3. (10 pts) Name one problem or hazard for multi-threaded programming. Describe how the Java keyword synchronized can be used to solve this problem. You may provide pseudo-code if it simplifies your explanation.

B4. (10 pts) What software pattern do the Java I/O classes implement? Describe the process of how to send a Serializable Object over the network using only Java stream I/O classes. You can assume you already have a connected Socket object.

C. Understanding Code (30 pts)

Consider the following superclass declaration used for this 3-part problem.

```
/**
 * A Shape represents a generic geometric shape that fits within
 * a rectangle.
 */
public abstract class Shape {
    private double myWidth, myHeight; // dimensions

    protected Shape(double width, double height) {
        myWidth = width;
        myHeight = height;
    }
    // returns specific dimension asked for
    public double getWidth() {
        return myWidth;
    }
    public double getHeight() {
        return myHeight;
    }
}
```

Part 1 (6 points)

Briefly answer the following questions regarding the class above.

C1. What does it mean for the class to be abstract?

C2. This class has no default constructor. What is the purpose of the provided class constructor?

C3. Is the `Shape` class mutable or immutable? Defend your choice.

Part 2 (8 points)

The classes `Circle`, `Square`, and `EquilateralTriangle` below all extend `Shape`.

```
public class Circle extends Shape {
    public Circle (double diameter){
        super(diameter, diameter);
    }
}

public class Square extends Shape {
    public Square (double sideLength){
        super(sideLength, sideLength);
    }
}

public class EquilateralTriangle extends Shape {
    public EquilateralTriangle (double sideLength){
        super(Math.sqrt(3.0d/4.0d) * sideLength, sideLength);
    }
}
```

You have a `List` that contains `Squares`, `Circles`, and `EquilateralTriangles`. You want to sort the `List` by area so that the smallest objects are first in the list.

C4. Design a polymorphic strategy for sorting by area, and identify which existing Java classes or interfaces you would use. Describe the behavior of any additional methods or classes you would add (you do not have to implement the code).

Part 3 (16 points)

We now want to display our `Shape` objects in a GUI. The following classes `X`, `Y`, and `Z` represent the model, view, and controller in the MVC pattern. (You do not need to fill in the blanks)

```
public class X implements _____ {
    private Y _y;
    private Z _z;
    ...
    public void actionPerformed(ActionEvent e) {
        if (((JButton)e.getSource()).getText().equals("Add Circle")) {
            _y.add(new Circle(1.0d));
        }
    }
}

public class Y {
    private List<Shape> _shapes;
    ...
    public void add(Shape shape) {
        _shapes.add(shape);
    }
}

public class Z extends _____ {
    private Y _y;

    public Z(Y y) {
        super();
        _y = y;
    }

    protected void paintComponent(Graphics g) {
        for (Shape s : _y.getShapes()) {
            // draw each shape
            ... #2
        }
    }
}
```

C5. Label next to the classes the model, the view, and the controller.

C6. Describe how you could use a Factory pattern to draw each shape.

C7. Describe how you could use polymorphism in the `Shape` object to handle the drawing.

C8. Compare the two approaches to drawing. Does one approach fit better with the MVC pattern? Explain your choice.

CISC370: Evaluation for Project 2, Group 2

Ray - 207/200

Anshu - 207/200

Josh - 212/200

Implementation/correctness - 140/140 points

Individual (Anshu):

The database component works well and has all the necessary integration with the website and server/RMI portion. It also seems that you implemented a lot of the server-side handling of game states (specifically monitoring client participation with pings, etc.) and RMI calls.

Individual (Ray):

The AI works and plays competitively (I know the general strategies are not yours and that you were interested in expanding the capabilities for >2 cards). The server-side Game model also works and handles multiple AI/human players. You did the right thing to design the server structures, provide a basic implementation, and allow Anshu to take over as needed. The time constraints definitely played a part in this - but the bottom line is that you correctly implemented the necessary functionality.

Individual (Josh):

The 3d interface works very well. I had a chance to run the Test3D without the server connection and the implementation was able to deal out all 52 cards very quickly, handle the different position rotation/translations easily, and looked good. And as with the others you contributed to the working server/RMI processing.

Team:

Finished project has many features and is fairly robust. I was particularly impressed that you were able to get the multi-player functionality completed and working with multiple clients+AI.

High level of contribution from all participants. It is apparent from the comments in the code, the repository revision histories, and the group evaluation that you were able to integrate the individual contributions and work as a team on the "glue" code. This is a very important skill to learn as often code is not cleanly separated between contributors and you will need to coordinate over these overlapping details.

Design - 37/40 points

Encapsulation:

Good overall. Classes are separated by responsibility and tier. Some data is shared between components that shouldn't necessarily need the information (e.g. entire BJ_Player objects sent to all clients).

Extendability/Polymorphism:

Sufficient. Some of the pieces of code are independent of the BlackJack game and could be re-used for other client/server game playing (most of the DB layer, 3d card classes, login/ping behavior). The AI and game representation/rules handling is very specific to BJ and would be difficult to use directly in another project, but that is to be expected.

There is some messy code in the GameState2 class where Thread.sleep() is used to "pace" the server with the client. There is certainly a better way to do this, but it works for this project. Given the time restrictions for the project this choice is more than understandable.

Analysis - 20/20 points

Team:

Based on group evaluation feedback and conversations I have had with group members, the project was a success. Although some difficulties were encountered, it seems that the group was able to work together to solve the problems and achieved high throughput.

Individual:

No particular notes here.

Summary -

This was a phenomenal project and quite an accomplishment for a 2-3 week time frame. I am rewarding the effort of the team with 10 points extra credit for this assignment because you went well beyond the requirements. I also individually am awarding 5 points extra credit to Josh for the amount of effort applied to the 3d GUI based on group evaluation feedback and repository evidence.

CISC370: Selected student answers to course evaluation questions

Student answers (unedited) are in monospace typeset.

Please comment on the course material. Did the course meet your expectations?

1 It actually exceeded my expectations in terms of how much detail was given to the different parts of the Java API and how they worked together. I learned much more than I expected to.

2 I was very happy with the topics covered and the assignments. Unlike some other CISC370 classes which my friends have taken, our topics/assignments seemed to be more applicable to real world situations and I feel that I learned some very useful information.

Please explain how your instructor could improve in helping students learn the materials covered in class.

Class lectures were very well organized and power point were used effectively, but as with most programming courses, most of the learning is done while doing assignments. I am not sure what else could have been done from a lecturing standpoint but the online resources (i.e. code base, power point slides) were invaluable when completing assignments and were appreciated.

Please comment on any additional areas where the course and materials worked well or could be improved.

Again, power point slides were a very useful resource when completing assignments. The instructor's knowledge of Java more than made up for little to no instruction from a text book--this also allowed the class to cover a variety of relevant topics and allow for higher level programming assignments. Text books tend to set a pace that may not agree with the instructor or students and for an upper level course such as this, the instructor's knowledge is much more valuable.

General comments:

Lastly, let me just say that you did a wonderful job teaching the course and succeeded in making even the more difficult aspects of the language (some of those API's are wild) quite learnable and enjoyable and not many in the Com Sci department do. I enjoyed your course and I think I got a lot out of it (the powerpoint slides will serve me well as future reference points). Thanks.