CSE 687

Object Oriented Design

Nadeem Ghani
nghani@syr.edu
Office: CST 4-232
Office Hours: Friday 10 AM - 12 PM

TAs
Rui Zuo: rzuo02@syr.edu

CSE 687

Preconditions

- How much java do you know? 1-3
- How many people on github? Have your own repo?
- How many people expect to work in a tech role?

CSE 687

What are we going to do?

Some soft stuff

- UML (Booch et. al.)

Mostly technical focus

- The Object-Oriented Thought Process, Weisfeld
- Head First: Object-Oriented Analysis and Design, McLaughlin et. al.
- Head First: Design Patterns, Freeman et. al.

Java specifics

- Java Language Specification (JLS)
- Effective Java, Bloch
- JDK source code

CSE 687

What are we going to do?

Design Patterns

- Gamma et. al

Architectural Patterns

Open-Source examples

CSE 687

Grading

- Class Participation (10%)
- Homework (50%)
  - (and individual project)
  - deliberately underspecified; make decisions; comment.
- In-class quizzes (20%)
  - will require reading code
  - allowed to run code (also look at docs, google etc)
- Final Exam (20%)
  - a longer, more intense quiz

should search be case insensitive?

CSE 687

Class Participation: Extra Credit -> HW

- If I make a mistake in lecture..
  - (BTW: I’m going to make 2 ‘mistakes’ today)
  - first person to interrupt and point out mistake: +1
-

CSE 687

Academic Integrity

The rules:

- Write your own code
- Don’t show anyone your code
- If I get two identical submissions, both people are equally responsible!
- What is the effect of a semester-long suspension on student visa?

CSE 687

Today’s agenda

- Very basic introduction to java
-

CSE 687

Java Design Goals

- Familiar
  - keep as much commonality with C++ as possible
- Simple
  - remove “the unnecessary complexities of C++...”
- Object Oriented
-
- Architecture Neutral and Portable
  - compile once - run anywhere

CSE 687

JVM

- Programmer writes *.java files
- Java Compiler (javac) compiles code into bytecode
    - bytecode != native code
    - bytecode == architecture neutral, intermediate format
    - *.java -> javac -> *.class
    - *.java human readable
    - *.class machine readable
- Java Virtual Machine (jvm) interprets bytecode
-
- Keeps benefits of compile-time error checking, but also provides portability

CSE 687

JVM

- FirstDemo.java
- ls -l
- javac FirstDemo.java
- java -cp . FirstDemo
- javap -c -classpath . FirstDemo
- https://en.wikipedia.org/wiki/List_of_Java_bytecode_instructions

CSE 687

Objects

- Everything in Java is either a primitive or a reference.
- There are only two data types in Java, primitive and reference.

- Every ref type in Java is (descended from) an Object.
- User (programmer) can’t add another primitive type.
- Define a ref type using the ***class*** keyword
  - cookie cutter
- Create an instance of a ref type using ***new*** keyword
  - cookie

CSE 687

What is an Object?

- Object with main()
- System.out.println “Hello World”

CSE 687

What is an Object?

- Object with main()
    - calls another static method
        - System.out.println returned value

CSE 687

What is an Object?

Instantiate object, invoke its method, and print its return value.

CSE 687

Java Basics

```
public class Hello {
     public static void main(String[] args) {
          System.out.println(“hello world”);
     }
}
```

- one public class per file:
  - Hello.java must have a `public class Hello{}`
  - World.java must have a `public class World{}`

CSE 687

Java Basics

```
public class Hello {
     public static void main(String[] args) {
          System.out.println(“hello world”);
     }
}
```

- java [classpath] Hello

CSE 687

Java Basics

```
package edu.syr.demo;

public class Hello {
    public static void main(String[] args) {
        System.out.println("hello world");
    }
}
```

- package statement constrains the location of a .java file
  - Hello.java must be in edu/syr/demo
    - relative path?! relative to what?

CSE 687

Java Basics

```
public class FirstDemo {

    public static void main(String[] args) throws InterruptedException {
        while (true) {
            System.out.println("hello world");
            Thread.sleep(60000);
        }
    }
}
```

- Every application/process must start from a main method
- main thread (of execution)
    - (jvm threads)
- kill -3 [pid]

CSE 687

Java Basics

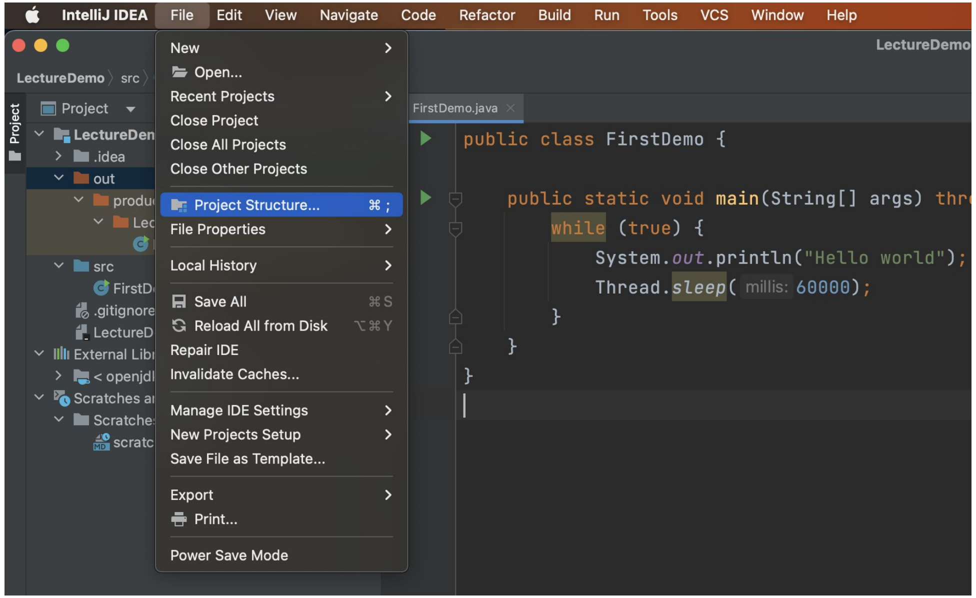
```
public class Hello {
    public static void main(String[] args) {
        System.out.println(“hello world”);
    }
}
```

- java -jar [name].jar
    - manifest points to which class’s main method should be run

CSE 687

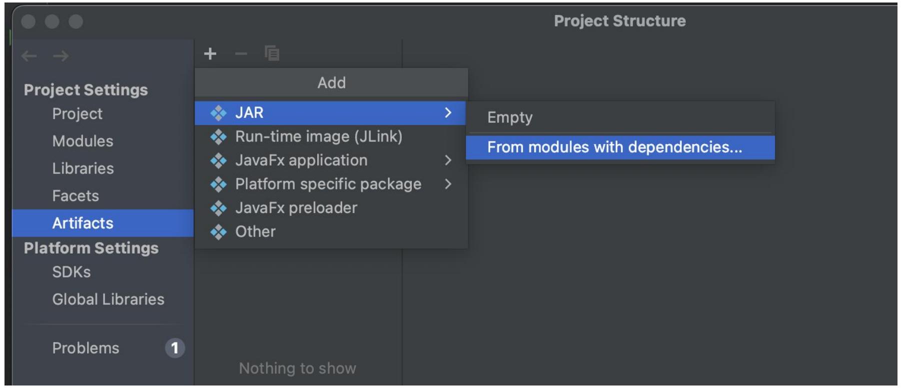
JVM

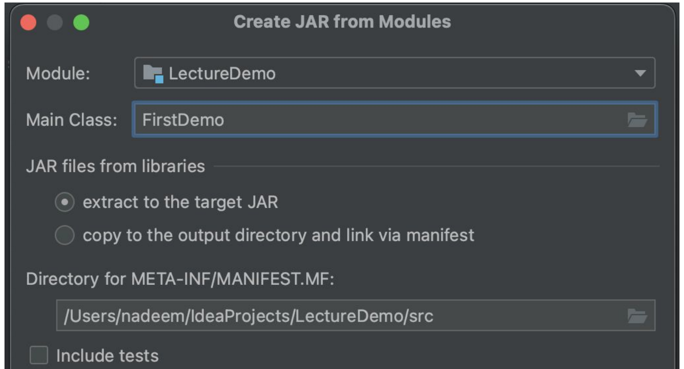
- Demo java -jar



```
public class FirstDemo {

    public static void main(String[] args) thr
        while (true) {
            System.out.println("Hello world");
            Thread.sleep( millis: 60000);
        }
    }
}
```




Create JAR from Modules

Module: LectureDemo

Main Class: FirstDemo

JAR files from libraries

- ● extract to the target JAR
- ○ copy to the output directory and link via manifest

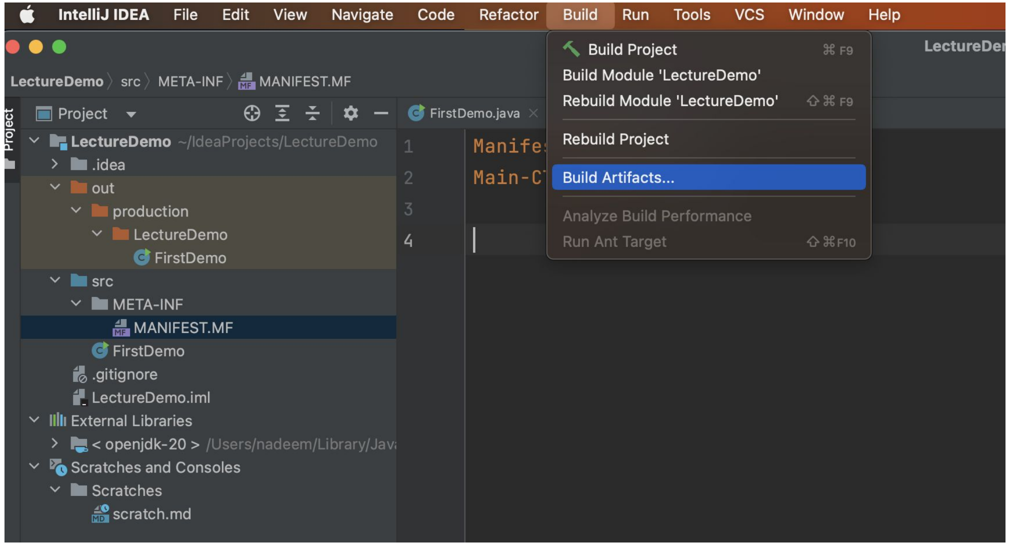
Directory for META-INF/MANIFEST.MF: /Users/nadeem/IdeaProjects/LectureDemo/src

☐ Include tests

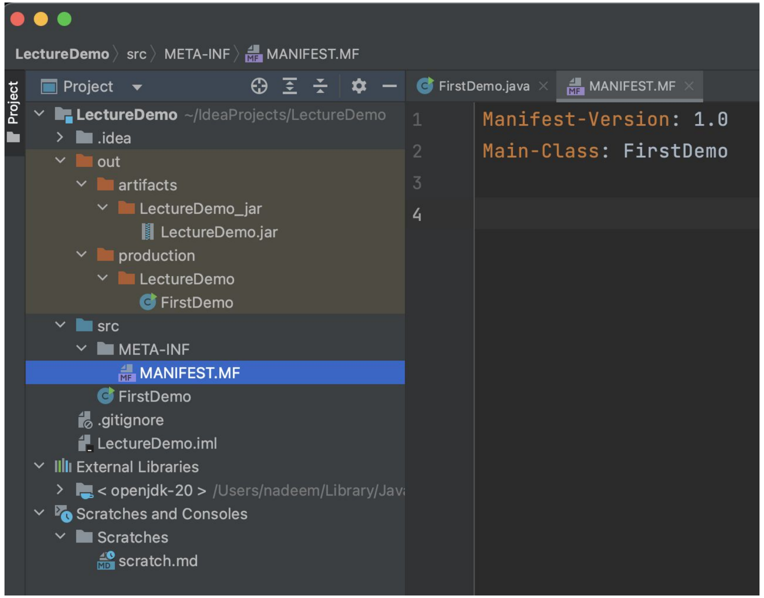
CSE 687

JVM

- Demo java -jar


IntelliJ IDEA File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
LectureDemo 〉 src 〉 META-INF 〉 MANIFEST.MF
Project
FirstDemo.java
LectureDemo ~/IdeaProjects/LectureDemo
.idea
out
production
LectureDemo
FirstDemo
src
META-INF
MANIFEST.MF
FirstDemo
.gitignore
LectureDemo.iml
External Libraries
< openjdk-20 > /Users/nadeem/Library/Java
Scratches and Consoles
Scratches
scratch.md
Build Project ⌘ F9
Build Module 'LectureDemo'
Rebuild Module 'LectureDemo' ⇧⌘ F9
Rebuild Project
Build Artifacts...
Analyze Build Performance
Run Ant Target ⇧⌘F10
LectureDe

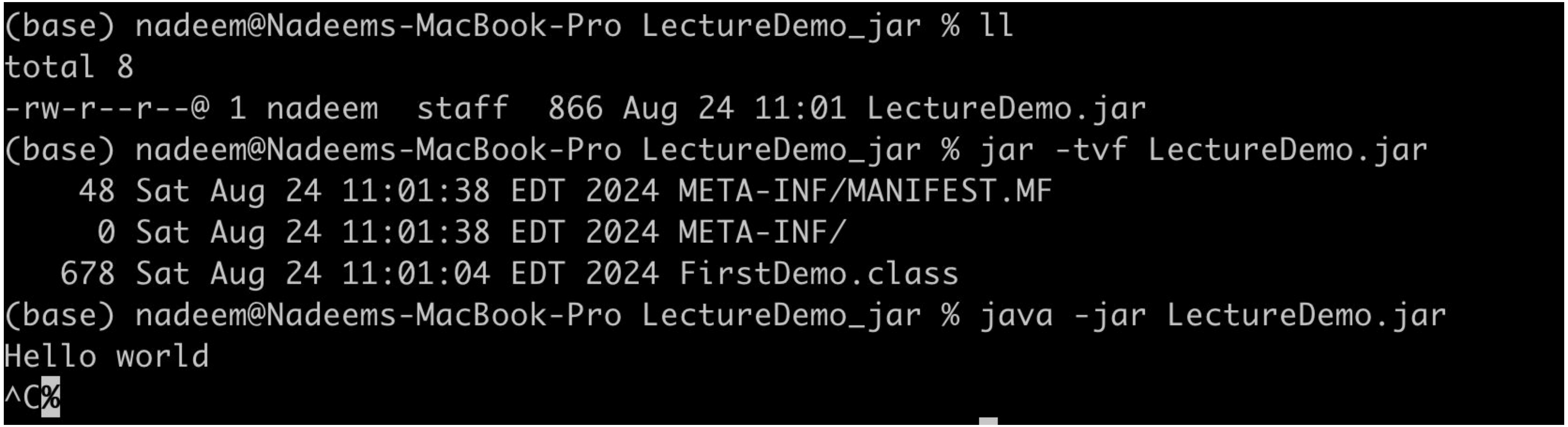
```
Manife
Main-C
```



```
Manifest-Version: 1.0
Main-Class: FirstDemo
```

CSE 687
JVM

- Demo java -jar



```
(base) nadeem@Nadeems-MacBook-Pro LectureDemo_jar % ll
total 8
-rw-r--r--@ 1 nadeem  staff  866 Aug 24 11:01 LectureDemo.jar
(base) nadeem@Nadeems-MacBook-Pro LectureDemo_jar % jar -tvf LectureDemo.jar
    48 Sat Aug 24 11:01:38 EDT 2024 META-INF/MANIFEST.MF
     0 Sat Aug 24 11:01:38 EDT 2024 META-INF/
   678 Sat Aug 24 11:01:04 EDT 2024 FirstDemo.class
(base) nadeem@Nadeems-MacBook-Pro LectureDemo_jar % java -jar LectureDemo.jar
Hello world
^C%
```