Intermediate Java – Part 2

Yotam Avivi

Oct-Nov 2017

Rethink Possible®

StringBuilder* – a Mutable String (for better performance)

```
StringBuilder sb = new StringBuilder("Hannah");
sb.length();   //6
sb.capacity(); //22 (16 characters initial capacity + string size)
sb.charAt(1);                          // a
sb.setCharAt(1,'i');                   //Hinnah
sb.setLength(2);                       //Hi
sb.append("l").append("l");            //Hill
sb.insert(0, "Big ");                  //Big Hill
sb.replace(3, 11, "");                 //Big
sb.reverse();                          //giB
```

*StringBuffer is a synchronized version of StringBuilder

Loops(1) – While Loop

```
while (condition){
        //do something
}
```

```
int x=0;
while (x < 10){  //What is the value of x when the loop ends ?
        x++;
}
```

Increases x by 1

Loops(2) – For Loops (with break)

```
String[] words = {“w1”, “w2”, “w3” .. “wN”};
for (int i = 0; i < words.length; i++) {
  if (“bingo”.equals(words[i]) {
      break;
  }
  //Do some stuff
}
```

Breaks the loop immediately

Loops(3) – For Loops (with Continue)

```
String[] words = {“w1”, “w2”, “w3” .. “wN”};
for (int i = 0; i < words.length; i++) {
  if (“bingo”.equals(words[i]) {
      continue;
  }
  //Do some stuff
}
```

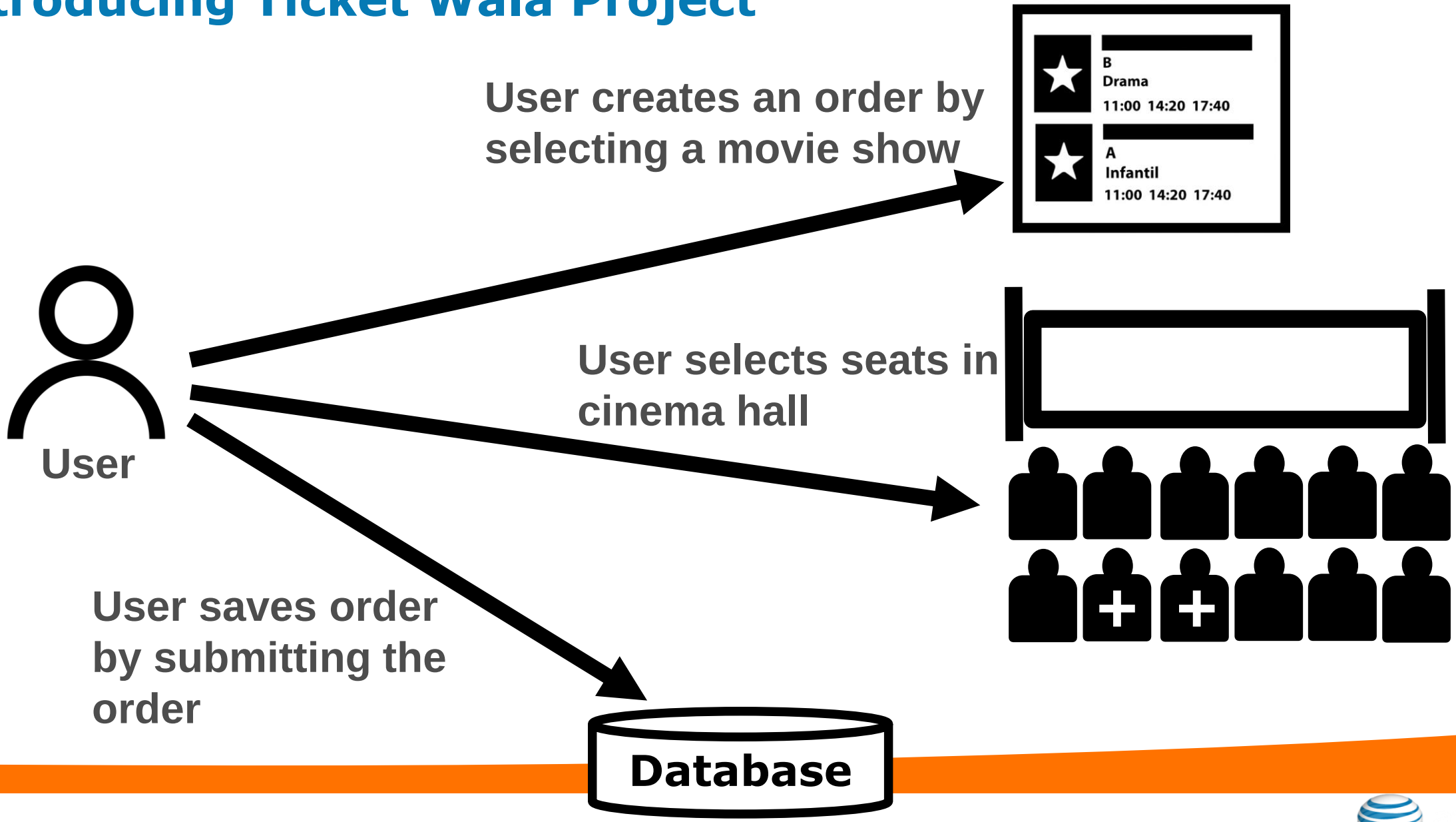
Skips to next iteration of the loop

Loops(4) – For Each Loop

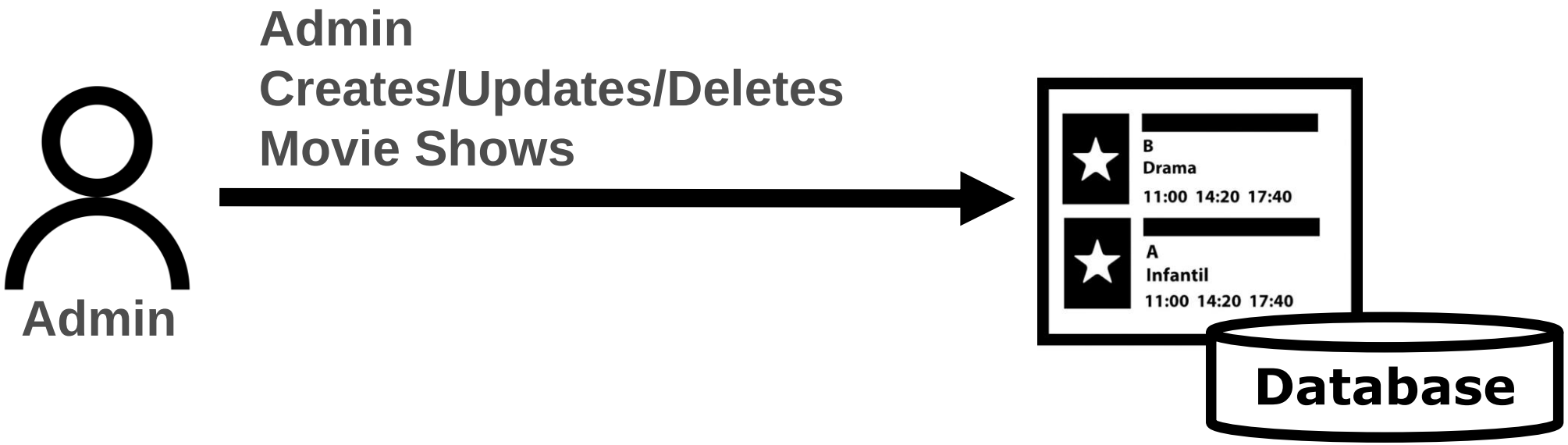
```
String[] words = {“w1”, “w2”, “w3” .. “wN”};
for (String s : words) {
   if (“bingo”.equals(s) {
       break;
   }
   //Do some stuff
}
```

A loop on each item in words array (no index is being used)

Introducing Ticket Wala Project



Introducing Ticket Wala Project (Cont.)



Exercise 4 – Building CLI for our Project

A user command is in the following structure:

<command-name> <arg1> <arg2> … <argn>

For example:

create-rectangle 5 6 ➔ **command-name**=create-rectangle **arg1**=5 **arg2**=6

add-user john doe ➔ **command-name**=add-user **arg1**=john **arg2**=doe

Write a program that reads commands from the user in a loop and displays the following text:

Command Name: <command-name>

Arg1: <arg1>

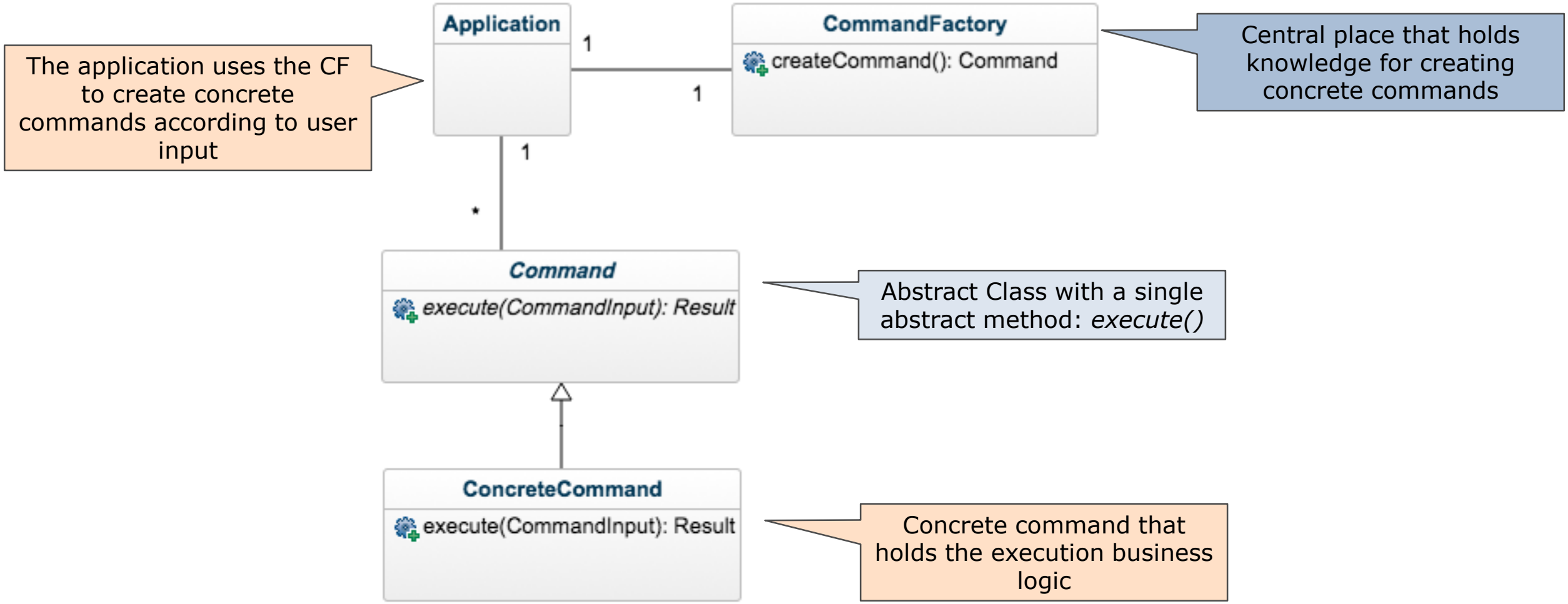
Arg2: <arg2> …

ArgN: <argn>

Exit Criteria: User types: "bye"*

* Use the Object's equals() method

Improve Implementation with Design Patterns – Factory/Command Patterns



Exercise 5 – Building CLI for out Project (Cont.)

Improve Exercise 4 design by implementing the Factory and Command Design Patterns.

Create a single command: echo (EchoCommand) that echoes back the user input.

For example:

> echo Hello World

Hello World

* Put new classes under package: com.ticketwala.command.api/impl

Exercise 5 – Example for the Main Method

```
System.out.println("Enter your command:\n");

CommandFactory commandFactory = new CommandFactory();

while (!"bye".equals(userCommandLine)) {
    try {

        userCommandLine = scanner.nextLine();

        //Use Command Factory that will create a suitable command according to user input
        Command command = commandFactory.createCommand(userCommandLine);

        //Execute command and display result
        Result result = command.execute();
        System.out.println(result.getMessage());

    } catch (Exception e) {
        System.err.println("Unexpected Command Line Error! " + e.getMessage());
    }
}
```

Arrays

Even though this is an array of primitives, in Java, an array is actually an object

```
int[] arrayOfIntegers = new int[10];


for (int i=0; i < arrayOfInteger.length; i++) {
    arrayOfIntegers[i] = i;
}
```

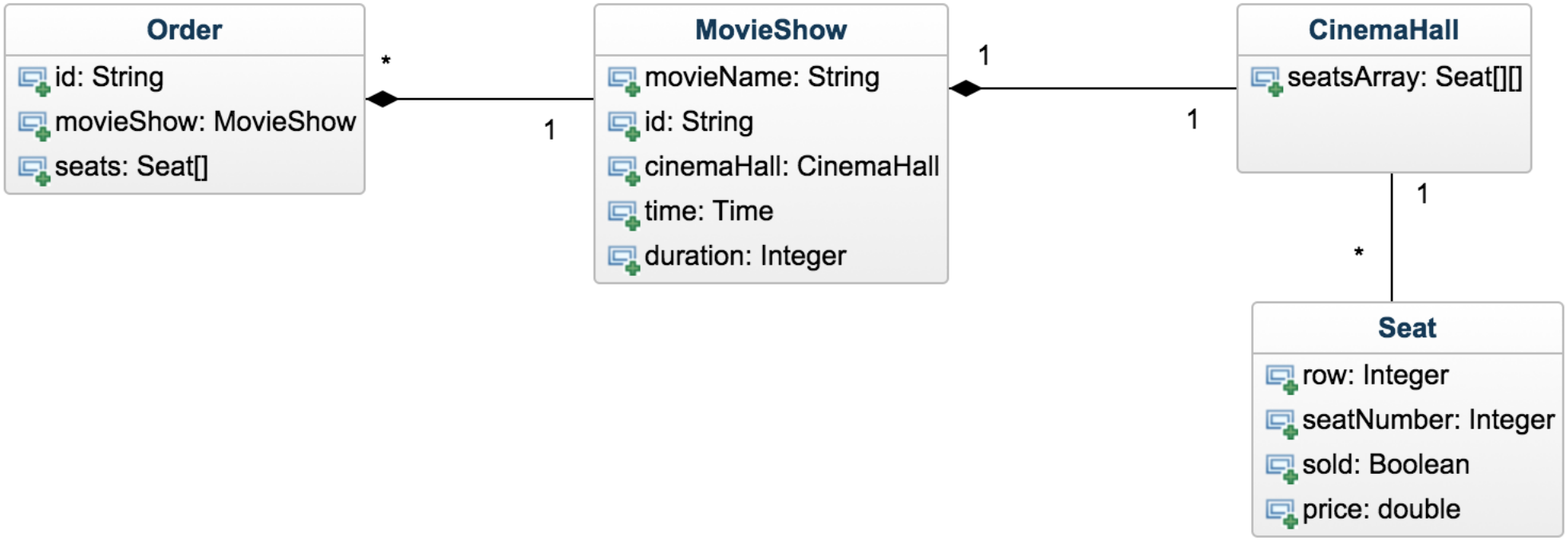
2D Arrays

```
int[][] matrixOfIntegers = new int[10][10];


for (int i=0; i < matrixOfIntegers.length; i++) {
  for (int j=0; j < matrixOfIntegers[i].length; j++) {
    matrixOfIntegers[i][j] = -1;  //How Many Iterations?
  }
}
```

A Nested Loop

Application Model Entities Class Diagram



Exercise 6 – Create Following Model Entities

Create the model entities according to the previous class diagram:

1. Seat – A Single Seat in a Cinema Hall
2. CinemaHall – Reflects the available and ordered seats in a certain Movie Show
   Override the toString() method that will be used to print a grid (2d array) representing the Hall itself (available seat is marked with '*' while a taken seat is marked with 'X'
3. MovieShow – Represents a single screening of a movie