Java and C# in Depth

Carlo A. Furia, Marco Piccioni, Bertrand Meyer

Exercise Session – Week 2

Organizational Part

- Send your assignments to alexey.kolesnichenko@inf.ethz.ch
- Start teaming up! Teams are of 3 people
- Enroll to piazza, if you haven't: piazza.com/ethz.ch/spring2014/252028400l/home

Agenda

- Quizzes
- More quizzes
- And even more quizzes …

Quiz 1. What will be printed?

- Integers

```java
public class Program{
    public static void main(String[] args) {
        Integer i1 = 123;
        Integer i2 = 123;
        Integer i3 = 128;
        Integer i4 = 128;
        System.out.println(i1 == i2);
        System.out.println(i3 == i4);
    }
}
```

```
true
false
```

```java
public class Program {
    static void Main(string[] args) {
        Object i1 = 123;
        Object i2 = 123;
        Object i3 = 128;
        Object i4 = 128;
        Console.WriteLine(i1 == i2);
        Console.WriteLine(i3 == i4);
    }
}
```

```
false
false
```

- (Java) Integer and int are different types
  - Autoboxing and unboxing
  - JVM creates a new object, sometime reuses an old one.
- (C#) no caching of boxed objects

Quiz 2. What will be printed?

- Floating point numbers

```
public class Program{
    public static void main(String[] args) {
        double d1=4.0;
        double d2=3.1;
        System.out.println(d1 – d2);
    }
}
```

0.8999999999999999

```
public class Program {
    static void Main(string[] args) {
        double d1 = 4.0;
        double d2 = 3.1;
        Console.WriteLine(d1 - d2);
    }
}
```

0.9

  - Why the imprecision?
    - Nature of floating point numbers (IEEE 754)
  - Why the precision?
    - Don’t get fooled! C# rounds float/double values to certain significant digits for base-10 string representation (what you see is not what it is)
    - Using **BigDecimal** in Java and **decimal** in C# solves these problems. Both classes work on base-10 representations, and are therefore slow.

Standard Numeric Format Strings (C#)

```
float sum = 0;
for (int i = 1; i < 11; i++){
    sum += 0.1f;
    Console.Write(String.Format("{0,-10}", sum.ToString()));
    Console.Write(", ");
    Console.WriteLine(sum.ToString("g9"));
}
```

```
0.1       , 0.100000001
0.2       , 0.200000003
0.3       , 0.300000012
0.4       , 0.400000006
0.5       , 0.5
0.6       , 0.600000024
0.7       , 0.700000048
0.8000001 , 0.800000072
0.9000001 , 0.900000095
1         , 1.00000012
```

Quiz 3: What does it do?

- Does the following Java code test oddity correctly?

```
public static boolean isOdd(int i) {
   return i % 2 == 1;
}
```

```
No
```

- What do you get from the following C# code?

```
public int Divide(int operand1, int operand2){
   return operand1 / operand2;
}
public int Test(){
   return Divide(4, 0);
}
```

```
DivideByZeroException
```

- What if we change int's to double's?

```
Compilation error (Java)
Double.PositiveInfinity or Double.NegativeInfinity (C#)
```

Quiz 4. What will be printed?

- Strings

```
public class ImmutableStrings{
   public static void main(String[] args){
      String a = "string";
      String b = "string";
      String c = "str";
      c += "ing";
      System.out.println(a.equals(b));
      System.out.println(a == b);
      System.out.println(a == c);
   }
}
```

```
true
true
false
```

```
class ImmutableStrings {
    static void Main(string[] args) {
        string a = "string";
        string b = "string";
        string c = "str";
        c += "ing";
        Console.WriteLine(a.Equals(b));
        Console.WriteLine(a == b);
        Console.WriteLine(a == c);
    }
}
```

```
true
true
true
```

- Strings are immutable objects in both languages
- Operator overloading in C#
  - Reference equality

```
Console.WriteLine(Object.ReferenceEquals(s1, s2));
```

Performance of concatenation versus StringBuilder (C#, in seconds)		
# of appends	+	StringBuilder
10	0.000	0.00
100	0.000	0.00
1,000	0.000	0.00
2,500	0.000	0.00
5,000	0.020	0.00
7,500	0.050	0.00
10,000	0.090	0.00
15,000	0.250	0.00
25,000	1.052	0.00
35,000	2.373	0.00
50,000	5.699	0.00
65,000	10.625	0.00
75,000	14.831	0.01
85,000	19.418	0.01
100,000	27.159	0.01
150,000	65.374	0.01
250,000	209.221	0.02
350,000	441.615	0.02
500,000	910.129	0.04
650,000	1521.708	0.06
750,000	1999.305	0.06
850,000	2576.575	0.06
1,000,000	3562.933	0.07

Quiz 5. Can we do this?

- Switch

```
int i1;
i1 = 2;
// can we do this?
switch (i1) {
    case 1: print(1);
    case 2: print(2);
    case 3: print(3);
}
```

```
double d1;
...
// this?
switch (d1) {
    case 1.0: ...
    case 2.0: ...
    case 3.0: ...
}
```

```
String s1;
...
// or this?
switch (s1) {
    case "1": ...
    case "2": ...
    case "3": ...
}
```

  - Rules about “fall-through”
  - Data types
    - (Java) **char**, **byte**, **short**, **int**, their corresponding reference types, **String**, or **enum**
    - (C#) Numeric, **String**, or enumeration data types
      - Refer to the language specification for more information

Quiz 6. What do these programs do? (Java)

- Type

```
public class Type1 {
    public static void main(String[] args) {
        String s = null;
        System.out.println(s instanceof String);
    }
}
```

Print “false”

```
public class Type2 {
    public static void main(String[] args) {
        System.out.println(new Type2() instanceof String);
    }
}
```

Compilation error

```
public class Type3 {
    public static void main(String args[]) {
        Type3 t3 = (Type3) new Object();
    }
}
```

Runtime exception

Quiz 7. What will be printed? (Java)

```
01 public class Test{
02     int x = 5;
03
04     public static void main(String[] args) {
05         final Test f1 = new Test();
07         Test f3 = testSwitch (f1);
08         System.out.println((f1 == f3) + " " + (f1.x == f3.x) + " " + f1.x + " "
09                            + " " + f3.x);
10     }
11
12     static Test testSwitch(final Test x) {
13         final Test z = x;
14         z.x = 6;
15         return z;
16     }
17 }
```

```
true true 6 6
```

- References f1, z, and f3 refer to the same instance.
  - final assures that a reference variable cannot be referred to a different object.
  - But final doesn't keep the object's state from changing.

Quiz 8. Type var and dynamic in C#

- Example

```
var myData = 365;
```

- True or False?
  - The `var` keyword can be used to define local variables in a method or property scope, return values, parameters, or field data of a custom type; False
  - Implicitly typed variables could be declared and initialized at different locations; False
  - Given the above declaration, assignment `myData = false;` will change the type of `myData` from `int` to `bool`. False
- What if we change *`var`* to *`dynamic`*?
- When to use?
  - var: Mostly, for results from LINQ queries
  - dynamic: scripting, interfacing with other languages, etc.

Quiz 9. Nullable data types in C#

- A nullable type can represent all the values of its underlying type, plus the value null

```
int? nullableInt = 10;
nullableInt = null;
```

- True or False?
  - Nullable types could be based on reference types too, e.g.

False

```
String? nullableStr = “A”;
```

  - The ? suffix notation is a shorthand for creating an instance of the generic System.Nullable<T> structure type.

True

```
if (nullableInt.HasValue)
    Console.WriteLine("Value of 'nullableInt' is: {0}",
          nullableInt.Value);
```

```
aInt = nullableInt.HasValue ? nullableInt.Value : 100;
// Could be written as:
aInt = nullableInt ?? 100;
```

Quiz 10. What does this program do? (C#)

- enum

```
using System;
class Program{
   private static int[] resource = new int[] {0, 1, 2};
   public enum Size{Small, Medium, Large}

   public static void Method1(Size theSize){
      Console.WriteLine(theSize);
      Console.WriteLine("Resource: {0}",
          resource[(int)theSize]);
   }

   static void Main(string[] args){
      Method1(Size.Small);
      Method1(Size.Large);
      Method1((Size) 1);
      Method1((Size) 3);
   }
}
```

IndexOutOfRangeException

```
if(System.Enum.IsDefined(
       typeof(Size), theSize)){
```

```
}
else{
    throw new Exception("…");
}
```

- Lacks type-safety
- Problem with serialization

Questions?