Java and C# in Depth

Carlo A. Furia, Marco Piccioni, Bertrand Meyer

Exercise Session – Week 6
Quiz 1: Does it compile? (Java)

```java
public class MyException extends Exception {
    // Checked exception

    // ... 
    public MyException(String message) { super(message); }
}

public class A {
    // doSomething throws a checked exception, but does not declare it
    public void doSomething() throws MyException {
        if (badStuffHappened) {
            throw new MyException("Help!");
        } else {
            // ...
        }
    }
}
```
Quiz 1 (cont.): Does it compile? (Java)

```java
public class A {
    public void doSomething() throws MyException {
        ...
    }
}

public class B {
    public void doMore() {
        A a;
        ...
        a.doSomething();
    }
}
```

doMore calls a method, which throws a checked exception, but doesn’t itself declare or catch it
private String readDataFromUrl(String url) throws MalformedURLException {
    // throw MalformedURLException if url is bad.
    // read data and return it.
}

private long parseLong(String data) throws NumberFormatException {
    // convert data to long.
    // throw NumberFormatException if data is not a valid long.
}

public long readNumberFromUrl(String url) throws MalformedURLException, NumberFormatException {
    String data = readDataFromUrl(url);
    long number = parseLong(data);
    return number;
}

How to avoid the accumulation of exception declarations up in the call hierarchy?


Quiz 2: solutions

Solution 1: Wrapping
public void readNumberFromUrl(String url) throws ApplicationException{
    try {
        String data = readDataFromUrl(url);
        long number = parseLong(data);
    } catch (MalformedURLException e) {
        throw new ApplicationException(e);
    } catch (NumberFormatException e) {
        throw new ApplicationException(e);
    }
}

Solution 2: Inheritance
public long readNumberFromUrl(String url) throws Exception {
    ...
Quiz 3: What will happen? (Java)

InputSteam input = null;
try {
    input = new FileInputStream("myFile.txt");
    //do something with the stream
} catch (IOException e) {
    throw new WrappedException(e);
} finally {
    try {
        input.close();
    } catch (IOException e) {
        throw new WrappedException(e);
    }
}

Suppose "myFile.txt" does not exist. Which exception will the client get?

NullPointerException is thrown inside the finally block and propagated to the client!

input is null

NullPointerException is thrown inside the finally block and propagated to the client!
Quiz 4: Does it compile? (Java)

List<Integer> li = new ArrayList<Integer>();
List<Number> ln = li;

List<String>[] lsa = new List<String>[10];
List<?>[] lsb = new List<?>[3]

public class DecimalString
    implements Comparable<Number>,
            Comparable<String> { ... }

Generics are not covariant!

Cannot create arrays of generic types

This is fine

Comparable<Number> and
Comparable<String> are the same
interface!
Quiz 4: Does it compile? (C#)

List<int> li = new List<int>();
List<object> lo = li;  // Generics are not covariant!

List<string>[] lsa = new List<string>[10];  // This is fine

public class DecimalString : 
    Comparable<int>, Comparable<string> { ... }  // This is fine

public class Comparable2<U, V> : 
    Comparable<U>, Comparable<v> { ... }  // U and V might be the same type
Quiz 5: ArrayList (Java and C#)

Does it work?
```java
class ArrayList<V> {
    private V[] storage;
    public ArrayList() {
        storage = new V[DEFAULT_SIZE];
    }
}
```

How to make it work in Java?
```java
class ArrayList<V> {
    private V[] storage;
    public ArrayList() {
        storage = (V[]) new Object[DEFAULT_SIZE];
    }
}
```

Java: cannot create an array of V
C#: it’s ok

It’s a cast, but it works
public T Convert<T>(Object o) where T: class
{
    if (o.GetType() == typeof(T)) {
        return (T) o;
    } else {
        return null;
    }
}

null might not be a valid value of T
Quiz 7: What is printed? (Java)

List<Integer> l1 = new ArrayList<Integer>();
List<String> l2 = new ArrayList<String>();

Class<?> c1 = l1.getClass();
Class<?> c2 = l2.getClass();

System.out.println(c1.equals(c2));

True
Both c1 and c2 are “ArrayList”
Quiz 7: What is printed (C#)

List<int> l1 = new List<int>(10);
List<string> l2 = new List<string>(10);

Type t1 = l1.GetType();
Type t2 = l2.GetType();

Console.WriteLine(t1.Equals(t2));

False
Questions?