Exception Handling in Java

CSCI 2300

Have you run into this?

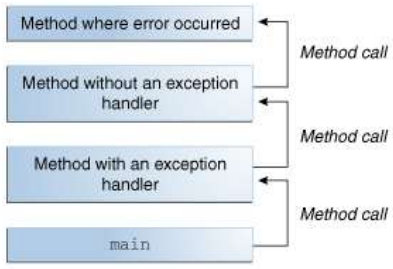
```
Exception in thread "main" java.util.InputMismatchException
at java.util.Scanner.throwFor(Scanner.java:864)
    at java.util.Scanner.next(Scanner.java:1485)
    at java.util.Scanner.nextInt(Scanner.java:2117)
    at java.util.Scanner.nextInt(Scanner.java:2076)
    at BoardPosition.select(BoardPosition.java:12)
    at TestBoardPosition.main(TestBoardPosition.java:5)
```

What is an Exception in Java?

An *exception* is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.

Method where error occurred creates an Exception object

Runtime system attempts to find something to handle it ("catch" it) in the call stack



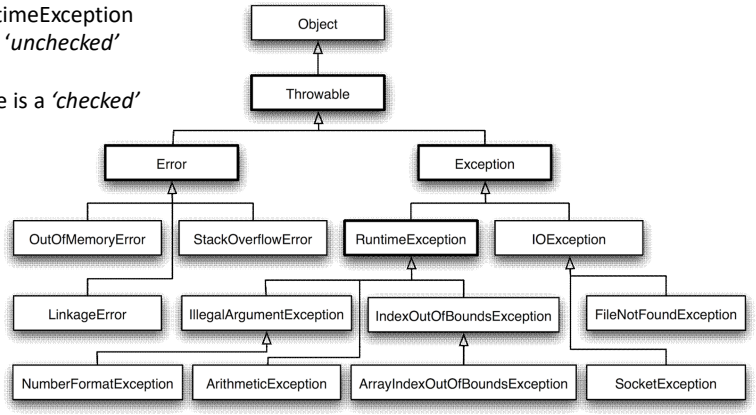
The call stack.

Catching Exceptions

```
try{
    // code that can throw an exception
}
catch (ExceptionType name){
   // handle this exception

}
catch (ExceptionType name){

   // handle this exception

}
```

Java Exception Class Hierarchy

- Error and RuntimeException subclasses are '*unchecked*' *exceptions*
- Everything else is a *'checked'* exception



Live Code Example

BoardPosition::select() uses java.util.Scanner class to read in two integers

What if the user enters non-integer values?

- An exception gets thrown

Should we handle it?

Where can we handle it?

How do we know that Scanner::nextInt() can throw an exception?

Your code should handle all exceptions that are subclasses of Exception class

Clean Exit on Exception

When a method exits 'abnormally', there may be some clean-up that doesn't get done

Example:

- You opened a file for writing and ran into an exception
- You want to close the file before the method exits

Solution: use 'try-finally' block

```
try{
   // regular code here
}
finally{
   // code that executes no matter what
}
```

Throwing Exceptions

When throwing 'checked' exception, specify with method name

Example:

```
public void someMethod() throws FileNotFoundException{
   //some code
   throw new FileNotFoundException();
}
```

Don't have to specify 'unchecked' exceptions

Can throw multiple exceptions:

```
public void someMethod() throws FileNotFoundException,
IOException
```

Defining your own Exception classes

Do you need an exception not defined in Java?

Would it help your users?

Extend **`Exception`** class or any subclass of **`Exception`**

```
public class InvalidInputException extends Exception{
   // define new methods
   // overwrite  methods of Exception class
}
```

Exceptions Practice

Do a 'git pull' on your repositories

You will see 'exceptions' directory

Navigate to that directory and examine the code

Modify the constructor of the Card class to throw an exception

Modify TestCard class to catch the exception