Exceptions
Lecture 13

Waterford Institute of Technology

June 8, 2014

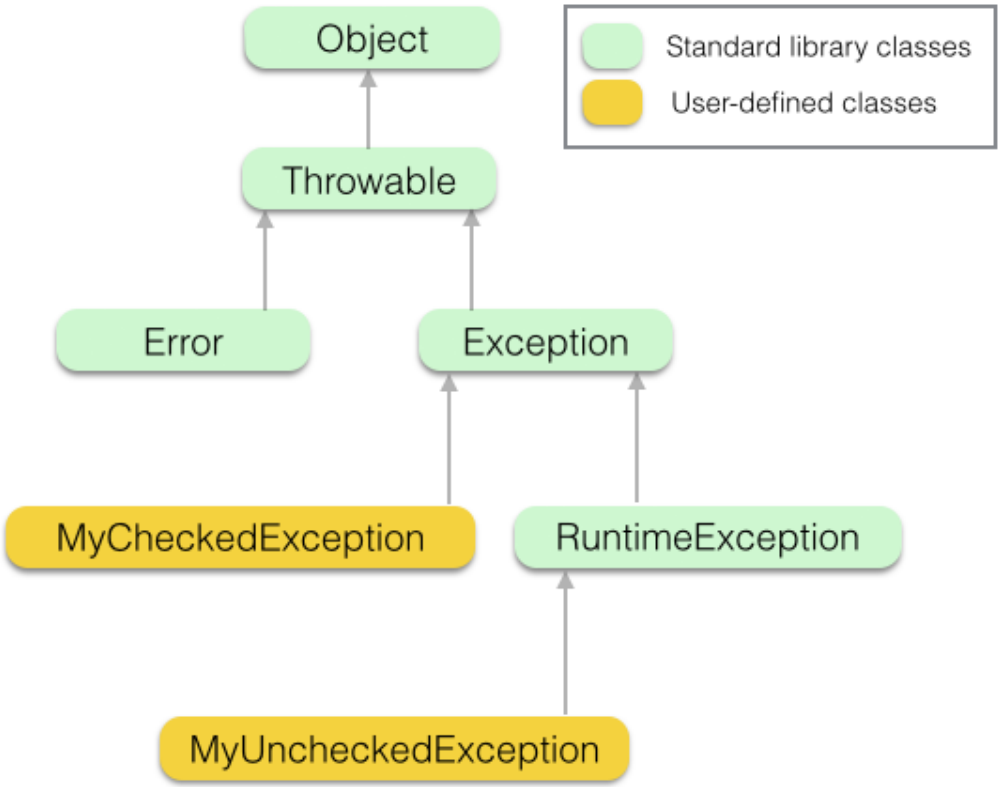
John Fitzgerald

Exceptions

Definition

Exception a disruptive event during program execution

- Managed in Java with Exception classes
- Derived from Throwable
- Checked
- Unchecked



Exceptions

Unchecked example

NullPointerException

- Library class runtime exception
- Attempt access through **null** reference
- No *check* necessary

```
Map<String, Book> books;

public void triggerNullPointerException()
{
  books.get(0);
}
```

```
Exception in thread "main" java.lang.NullPointerException
        at exceptions.ExceptionDemo.triggerNullPointerException(ExceptionDemo.java:11)
        at exceptions.ExceptionTest.main(ExceptionTest.java:10)
```

Exceptions

Checked example

BookException

- User-defined class
- Attempt access map using **null** key
- Must provide infrastructure to *check*

```
public Book getDetailsChecked(String key) throws BookException
{
  if(key == null)
  {
    throw new BookException("\nChecked exception demo: key is null");
  }
  return books.get(key);
}
```

```
exceptions.BookException:

Checked exception demo: key is null
        at exceptions.ExceptionDemo.getDetailsChecked(ExceptionDemo.java:26)
        at exceptions.ExceptionTest.main(ExceptionTest.java:17)
```

Exceptions

Check infrastructure

throws throw

- Signature method in which checked exception thrown

```
public Book getDetailsChecked(String key) throws BookException
```

- **throw** exception if key null

```
throw new BookException("\nChecked exception demo: key is null");
```

```
public Book getDetailsChecked(String key) throws BookException
{
  if(key == null)
  {
    throw new BookException("\nChecked exception demo: key is null");
  }
  return books.get(key);
}
```

Exceptions

Check infrastructure

try catch

- Wrap method invocation in *try* block

  try {demo.getDetailsChecked(null); }

- Wrap action in event of exception in *catch* block

  catch (BookException e) {e.printStackTrace(); }

```
public static void main(String[] args)
{
  try
  {
    demo.getDetailsChecked(null);
  }
  catch (BookException e)
  {
    e.printStackTrace();
  }
}
```

Exceptions

Check infrastructure

finally

- Where present, executes on exit from *try*
  - Additionally to exception handling,
  - Helps avoid resource leaks because
  - Block executed even when exception thrown

```
Out out = new Out();
// write to stdout
try{
  out.println("Test 1");
}
finally
{
  if(out != null)
  {
    out.close();
  }
}
```

Exceptions

User-defined BookException class

Convention is to end name with *Exception*
May optionally override *Throwable* methods such as:

- getLocalizedMessage
- getMessage
- printStackTrace

```
public class BookException extends Exception
{
  String message;
  BookException(String message) {
    this.message = message;
  }
  public String getLocalizedMessage() {
    return "\n" + message;
  }
}
```

Referenced Material

1. The Java Tutorials: Exceptions

`http://docs.oracle.com/javase/tutorial/essential/exceptions/index.html`

[Accessed 2014-06-07]