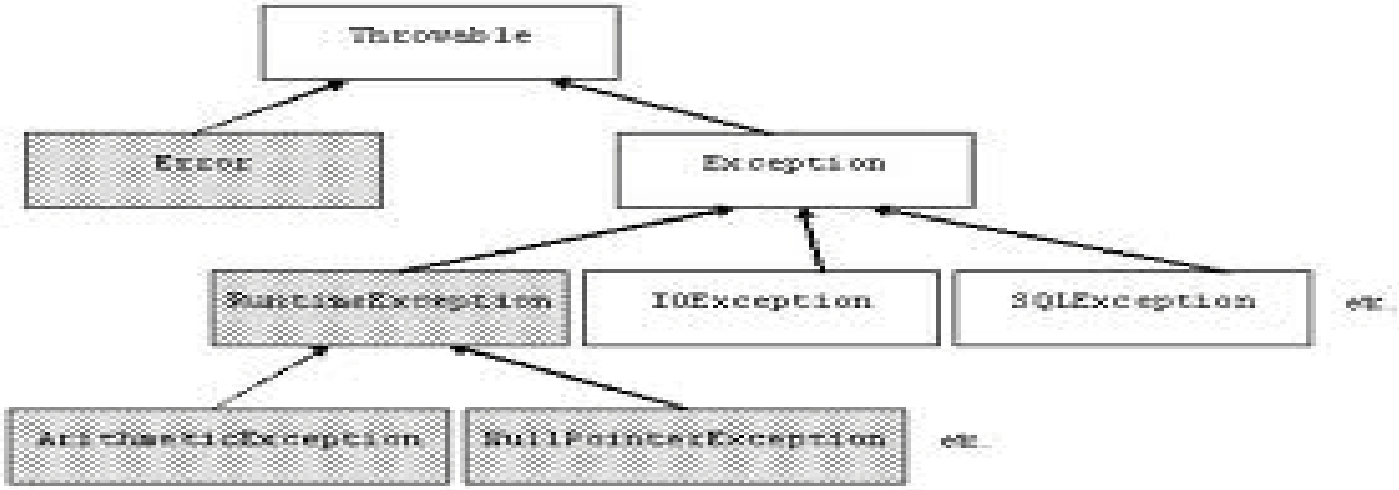
# Exception Handling

- In java there is a exception handling mechanism, according to which if a ‘known’ method gets failed at runtime at certain condition, then we are ready to handle that exception when that condition arrives.
- Hence at a “pre-known failure condition” the program doesn’t fails.
- Based on:

The exception handling mechanism is based on the rule that “if we call a risky method, then the caller must acknowledge that we know it is a risky method”

# Exception hierarchy

- All exceptions are subclasses of class Exception.
- Exception class is subclass of Throwable class.
- Exception class have two subclasses IOException and Runtime Exception.
- Runtime exception are those exception which are not checked by compiler and are thrown at the runtim



# Creating our own exception class

You can create your own exceptions in Java. Keep the following points in mind when writing your own exception classes:

- All exceptions must be a child of Throwable.
- If you want to write a checked exception that is automatically enforced by the Handle or Declare Rule (??), you need to extend the Exception class.
- If you want to write a runtime exception, you need to extend the RuntimeException class.

**public class myException extends Exception**

**{}**

# Declaring a risky method

- A method is said to be ‘risky’, when it throws an exception. Hence for declaring a risky method:

1. It must ‘throw’ a exception object.
2. It must say ‘throws’ during its declaration.

```
public void go() throws myException
{int x = 0;
if(x==0)
{
throw new myException();
}}
```

# Calling a risky method

- Whenever and wherever a risky method is called, we should acknowledge the compiler that we know!!!, it is an risky method, otherwise the code will not compile.
- This is done by try/catch block.

# Try/catch block and it's flow

- **public static void main(String[] args)**
- {
- PrivateConst p = **new PrivateConst();**
- **try**
- {p.go();

- }**catch (myException ex)**
- {
- System.*out.println("exception is handled");*
- }
- }
- Flow of Try/catch block

Whenever a risky method is called, flow goes to the 'try-block', if the method succeeded, then catch block never runs, otherwise the try block never runs and only catch runs

# Finally keyword

- Finally keyword is used to run a block of code irrespective of try/catch block.
- Whether the try runs or catch runs, Finally block always runs.
- **public static void main(String[] args)**
- {
- PrivateConst p = **new PrivateConst();**
- **try**
- {p.go();

- }**catch (myException ex)**
- {
- System.*out.println("exception is handled");*
- }
- **finally**
- {
- System.*out.println("always runs");*
- }

# Multiple Exceptions

# Exception Ducking and Duckers

# Rules for Exception