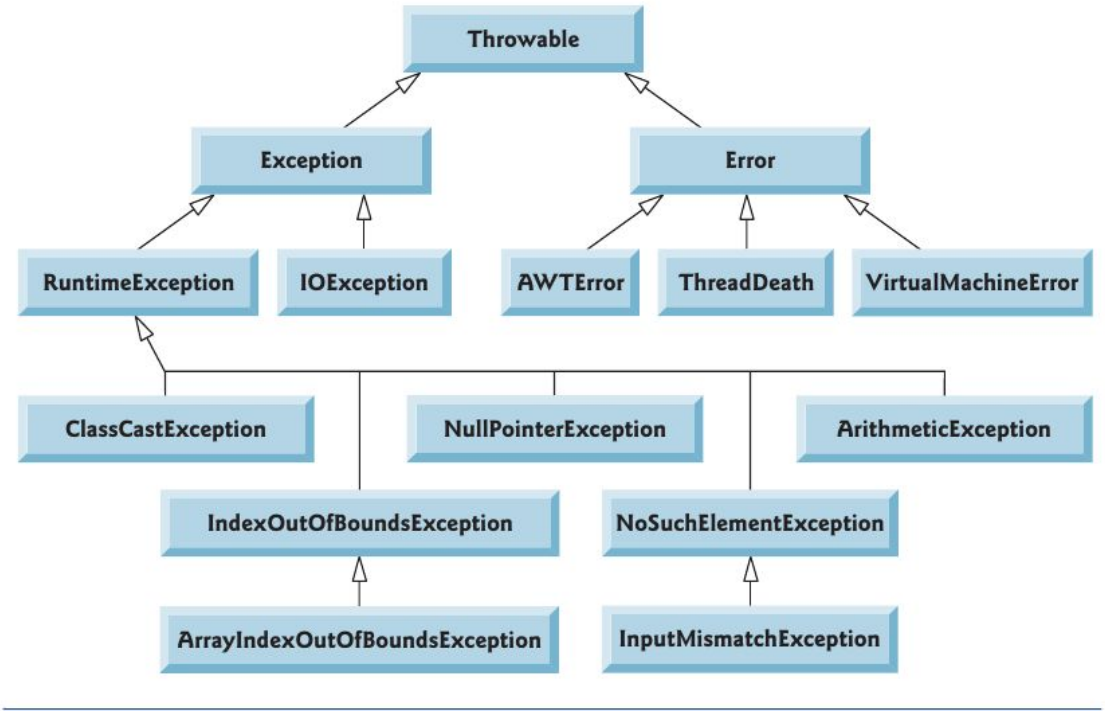
Exception Hierarchies



Exception Handling Stack Propagation with Method Overriding

Superclass method does not declare an exception, subclass overridden method cannot declare the checked exception but it can declare unchecked exception.

Example1: Will generate **Compile time error** as unchecked exceptions cannot be thrown by the subclass.

```
import java.io.IOException;
class ParentClass{
	public void testMethod(){
		System.out.println("In ParentClass......");
	}
}
class SubClass extends ParentClass{
	public void testMethod() throws IOException{ // overridden method
		//Exception IOException is not compatible with throws clause in
ParentClass.testMethod()
		System.out.println("In SubClass......");
```

```
	}
}
class ExceptionHierachy {
	public static void main(String []args) throws IOException{
		SubClass obj = new SubClass();
		obj.testMethod();
	}
}
```

Example2: Subclass can generate runtime exception when superclass has not defined any exception.

```
import java.io.IOException;
class ParentClass{
	public void testMethod(){
		System.out.println("In ParentClass......");
	}
}
class SubClass extends ParentClass{
	public void testMethod() throws RuntimeException{ // overridden method
		System.out.println("In SubClass......");
	}
}
class ExceptionHierachy {
	public static void main(String []args){
		SubClass obj = new SubClass();
		obj.testMethod();
	}
}
```

Superclass method declares an exception, subclass overridden method can declare same, subclass exception or no exception but cannot declare parent exception.

Example1: Subclass method declares no exception.

```
import java.io.IOException;
class ParentClass{
	public void testMethod() throws IOException{
		System.out.println("In ParentClass......");
	}
}
class SubClass extends ParentClass{
	public void testMethod(){ // overridden method
		System.out.println("In SubClass......");
	}
```

```
}
class ExceptionHierachy {
	public static void main(String []args){
		SubClass obj = new SubClass();
		obj.testMethod();
	}
}
```

Example2: Subclass method declares same exception as ParentClass method.

```
import java.io.IOException;
class ParentClass{
	public void testMethod() throws IOException{
		System.out.println("In ParentClass......");
	}
}
class SubClass extends ParentClass{
	public void testMethod() throws IOException{ // overridden method
		System.out.println("In SubClass......");
	}
}
class ExceptionHierachy {
	public static void main(String []args) throws IOException{
		SubClass obj = new SubClass();
		obj.testMethod();
	}
}
```

Example3: Subclass method declares super exception as compared ParentClass method will generate compile time error.

```
import java.io.IOException;
class ParentClass{
	public void testMethod() throws IOException{
		System.out.println("In ParentClass......");
	}
}
class SubClass extends ParentClass{
	public void testMethod() throws Exception{ // overridden method
		//Exception Exception is not compatible with throws clause in
ParentClass.testMethod
		System.out.println("In SubClass......");
```

```
	}
}
class ExceptionHierachy {
	public static void main(String []args) throws Exception{
		SubClass obj = new SubClass();
		obj.testMethod();
	}
}
```

Example4: Subclass method declares subclass exception as compared ParentClass method.

```
import java.io.IOException;
import java.io.FileNotFoundException;
class ParentClass{
	public void testMethod() throws IOException{
		System.out.println("In ParentClass......");
	}
}
class SubClass extends ParentClass{
	public void testMethod() throws FileNotFoundException{ // overridden
method
		System.out.println("In SubClass......");
	}
}
class ExceptionHierachy {
	public static void main(String []args) throws Exception{
		SubClass obj = new SubClass();
		obj.testMethod();
	}
}
```