Content Menu ▼

ExceptionHandling with MethodOverriding in Java

There are many rules if we talk about methodoverriding with exception handling. The Rules are as follows:

- **If the superclass method does not declare an exception**
  - If the superclass method does not declare an exception, subclass overridden method cannot declare the checked exception but it can declare unchecked exception.
- **If the superclass method declares an exception**
  - If the superclass method declares an exception, subclass overridden method can declare same, subclass exception or no exception but cannot declare parent exception.

If the superclass method does not declare an exception

1) Rule: If the superclass method does not declare an exception, subclass overridden method cannot declare the checked exception.

```
import java.io.*;
class Parent{
  void msg(){System.out.println("parent");}
}

class TestExceptionChild extends Parent{
  void msg()throws IOException{
    System.out.println("TestExceptionChild");
  }
  public static void main(String args[]){
   Parent p=new TestExceptionChild();
   p.msg();
  }
}
```

Test it Now

```
Output:Compile Time Error
```

2) Rule: If the superclass method does not declare an exception, subclass overridden method cannot declare the checked exception but can declare unchecked exception.

```
import java.io.*;
class Parent{
  void msg(){System.out.println("parent");}
}

class TestExceptionChild1 extends Parent{
  void msg()throws ArithmeticException{
    System.out.println("child");
  }
  public static void main(String args[]){
   Parent p=new TestExceptionChild1();
   p.msg();
  }
}
```

Test it Now

```
Output:child
```

If the superclass method declares an exception

1) Rule: If the superclass method declares an exception, subclass overridden method can declare same, subclass exception or no exception but cannot declare parent exception.

Example in case subclass overridden method declares parent exception

```
import java.io.*;
class Parent{
  void msg()throws ArithmeticException{System.out.println("parent");}
}
```

```
class TestExceptionChild2 extends Parent{
  void msg()throws Exception{System.out.println("child");}

  public static void main(String args[]){
   Parent p=new TestExceptionChild2();
   try{
   p.msg();
   }catch(Exception e){}
  }
}
```

Test it Now

```
Output:Compile Time Error
```

Example in case subclass overridden method declares same exception

```
import java.io.*;
class Parent{
  void msg()throws Exception{System.out.println("parent");}
}

class TestExceptionChild3 extends Parent{
  void msg()throws Exception{System.out.println("child");}

  public static void main(String args[]){
   Parent p=new TestExceptionChild3();
   try{
   p.msg();
   }catch(Exception e){}
  }
}
```

Test it Now

```
Output:child
```

Example in case subclass overridden method declares subclass exception

```
import java.io.*;
class Parent{
  void msg()throws Exception{System.out.println("parent");}
}

class TestExceptionChild4 extends Parent{
  void msg()throws ArithmeticException{System.out.println("child");}

  public static void main(String args[]){
   Parent p=new TestExceptionChild4();
   try{
   p.msg();
   }catch(Exception e){}
  }
}
```

Test it Now

```
Output:child
```

Example in case subclass overridden method declares no exception

```
import java.io.*;
class Parent{
  void msg()throws Exception{System.out.println("parent");}
}

class TestExceptionChild5 extends Parent{
  void msg(){System.out.println("child");}

  public static void main(String args[]){
   Parent p=new TestExceptionChild5();
   try{
   p.msg();
   }catch(Exception e){}
  }
}
```

Test it Now

```
Output:child
```