自定义异常

程序员在程序中主动抛出异常时，一般不会抛出一个 JDK 中已经有的异常，因为这样基本上没有太大的意义。所以 throw 的异常一般是程序员自定义的异常类型。

当 JDK 中的异常类型不能满足程序的需要时，就可以使用自定义异常了。自定义异常主要用在框架中，Java 在学到后期，我们会学习很多框架的课程，这些框架都有很多自己定义的异常类型。

如果我们自己也要定义一个异常类型，有这样几种方式。

1、继承 Throwable 类

2、继承 Exception 类

3、RuntimeException 类。

一般会选择继承 Exception 和 RuntimeException，如果不要求调用者一定要处理抛出的异常，就继承 RuntimeException。其余的就继承 Exception 就可以了。

继承 Exception 或 RuntimeException 之后，我们其实就已经完成了一个自定义的异常。但要让我们的自定义异常跟普通的异常相同的创建方式，需要几个构造方法，并继承父类的实现。Throwable 中一共定义了以下四个构造方法。

语法

```
//构造方法 1
public MyException() {
    super();
}
//构造方法 2
public MyException(String message) {
    super(message);
}
//构造方法 3
public MyException(String message, Throwable cause) {
    super(message, cause);
}
//构造方法 4
public MyException(Throwable cause) {
```

```
        super(cause);
    }
```

Exception 中的构造方法与 Throwable 相同，我们的自定义异常应该也有这样四个构造方法，这样，在创建我们的自定义异常对象时，所使用的方式就跟使用其他异常是一样的。前两个构造方法比较简单，第三个构造方法表示根据指定的原因构建新的异常对象，但是更换了异常的信息，第四个构造方法表示根据指定的原因构建新的异常对象，使用已有的异常信息，即 cause 中的异常信息，通过 getCause()方法可以获取。printStackTrace()也可以将其输出。

下面通过示例学习如何编写自定义异常，代码如示例 1 所示。

示例 1：

```
/**
 * 自定义异常
* @author 北大青鸟
 */

public class MyException extends Exception {
    public MyException() {
        super();
    }
    public MyException(String message) {
        super(message);
    }
    public static void main(String[] args) throws MyException {
        throw new MyException("我的自定义异常对象");
    }
}
```

运行示例 1 效果如图 1 所示。

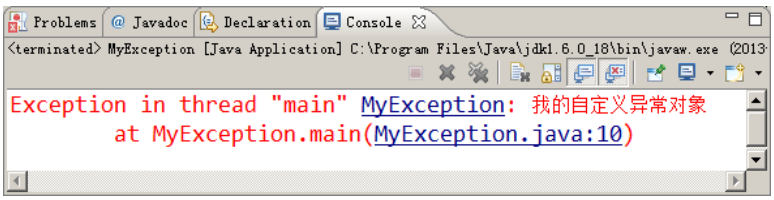

图 1 示例 1 运行效果

下面再通过一个实例具体看一下，自定义异常在实际应用中是如何应用的，代码如示例 2 所示。

示例 1：

```
/**
 * 自定义异常
 * @author 北大青鸟
 */
```

```
class GenderException extends Exception {
    public GenderException(String message){
        super(message);
    }
}
/**
 * 人类
 * @author 北大青鸟
 */
class Person {
    private String sex;
    public String getSex(){
        return sex;
    }
    public void setSex(String sex) throws GenderException {
        if ("男".equals(sex) || "女".equals(sex))
            this.sex = sex;
        else {
            throw new GenderException("性别必须是\"男\"或者\"女\"! ");
        }
    }
}
/**
 *测试类:
 * @author 北大青鸟
 */
public class Test {
    public static void main(String[] args) {
        Person person = new Person();
        try {
            person.setSex("Male");
        } catch (GenderException e) {
            e.printStackTrace();
        }
    }
}
```

运行示例 2 效果如图 2 所示。

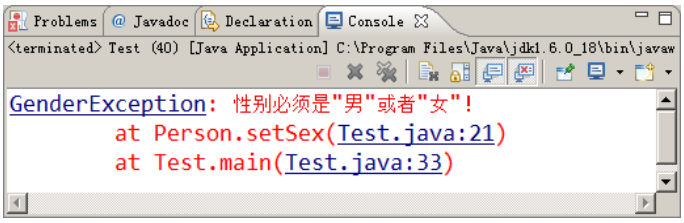

图 2 示例 2 运行效果