前置条件验证(Preconditions)

Guava提供有许多形形色色的预检查工具。我们强烈而深情的建议以静态的方式导入它们。
每个预检查的方法都有三种不同的重载形式：

- 无其他的附带参数。如果检查不通过，将抛出无错误提示信息的任何异常。
- 附带有一个`Object`的参数。如果检查不通过，将抛出以`object.toString()`为错误提示的异常。
- 附带有一个`String`类型的参数，还有任意数量的`Object`参数。这有点类似与printf方法，但为了兼容GWT(Google Web Toolkit)以及考虑到效率，它只允许使用`s%`表示符，形如：

```
checkArgument(i >= 0, "Argument was %s but expected nonnegative",
i);
checkArgument(i < j, "Expected i < j, but %s > %s", i, j);
```

Preconditions的方法大致如下：

方法签名(不包含附带参数)	方法描述	失败时抛出的异常
`checkArgument(boolean)`	检查参数中的boolean值是否为true.用于校验方法的参数	`IllegalArgumentException`
`checkNotNull(T)`	检查给定的值不是null，将直接返回该值。因此你可以内联的使用`checkNotNull(value)`	`NullPointerException`
`checkState(boolean)`	检查对象的某些状态值,但不依赖这个参数.例如一个`Iterator`对象可以在调用`remove`之前调用此方法来检查是否调用过它的`next`方法	`IllegalStateException`
`checkElementIndex(int index, int size)`	检查index在指定的list,string或array中是否是一个有效的元素下标,元素的下标应该是在0到size-1之间,你可以不用传入	`IndexOutOfBoundsException`

	list,string或array,只需要传入他们的size即可，此方法将返回`index`的值	
`checkPositionIndex(int index, int size)`	检查index在指定的list,string或array中是否是一个有效的元素下标,元素的下标应该是在0到size-1之间,你可以不用传入list,string或array,只需要传入他们的size即可，此方法将返回`index`的值	`IndexOutOfBoundsException`
`checkPositionIndexes(int start, int end, int size)`	检查[start,end)是否是指定的list,string,array的一个子范围,如果检查不通过,将抛出该方法自有的错误信息	`IndexOutOfBoundsException`

我们把这些方法与Apache的Commons工具作过一些对比后，还是觉得我们的这些工具要略胜一筹，其原因是：

- 静态导入这些方法后，Guava的方法名达意清晰，状物明了。如`checkNotNull`方法， 对所作之事，欲抛之异常，表达的直接清楚。
- `checkNotNull`方法在校验参数后会返回原来的参数值，因此允许写成简单的单行构造形式，如`this.field = checkNotNull(field)`
- 附带简易，变参风格的异常信息输出。

我们建议你在使用时把这些条件验证方法都分开使用，这样可以帮助你在debugging时快速定位到失败的地方。此外，你亦应该提供有用的错误信息，当检查出错时，会更容易的找出相应的原因。