**创建和引发异常（C# 编程指南）**

异常用于指示在运行程序时发生了错误。此时将创建一个描述错误的异常对象，然后使用 [throw](https://msdn.microsoft.com/zh-cn/library/1ah5wsex.aspx) 关键字“引发”该对象。然后运行时搜索最兼容的异常处理程序。

当存在下列一种或多种情况时，程序员应引发异常：

* 方法无法完成其中定义的功能。

例如，如果方法的参数具有无效值：

static void CopyObject(SampleClass original)

{

if (original == null)

{

throw new System.ArgumentException("Parameter cannot be null", "original");

}

}

* 根据对象的状态，对某个对象进行不适当的调用。

一个示例可能尝试对只读文件执行写操作。在对象状态不允许某项操作的情况下，引发 [InvalidOperationException](https://msdn.microsoft.com/zh-cn/library/system.invalidoperationexception.aspx) 的一个实例或基于此类的派生类的对象。以下为引发 [InvalidOperationException](https://msdn.microsoft.com/zh-cn/library/system.invalidoperationexception.aspx) 对象的方法的示例：

class ProgramLog

{

System.IO.FileStream logFile = null;

void OpenLog(System.IO.FileInfo fileName, System.IO.FileMode mode) {}

void WriteLog()

{

if (!this.logFile.CanWrite)

{

throw new System.InvalidOperationException("Logfile cannot be read-only");

}

// Else write data to the log and return.

}

}

* 方法的参数导致了异常。

在此情况下，应捕获原始异常并创建一个 [ArgumentException](https://msdn.microsoft.com/zh-cn/library/system.argumentexception.aspx) 实例。原始异常应作为 [InnerException](https://msdn.microsoft.com/zh-cn/library/system.exception.innerexception.aspx) 参数传递给 [ArgumentException](https://msdn.microsoft.com/zh-cn/library/system.argumentexception.aspx) 的构造函数：

static int GetValueFromArray(int[] array, int index)

{

try

{

return array[index];

}

catch (System.IndexOutOfRangeException ex)

{

System.ArgumentException argEx = new System.ArgumentException("Index is out of range", "index", ex);

throw argEx;

}

}

异常包含一个名为 [StackTrace](https://msdn.microsoft.com/zh-cn/library/system.exception.stacktrace.aspx) 的属性。此字符串包含当前调用堆栈上的方法的名称，以及为每个方法引发异常的位置（文件名和行号）。 [StackTrace](https://msdn.microsoft.com/zh-cn/library/system.exception.stacktrace.aspx) 对象由公共语言运行时 (CLR) 从 **throw** 语句点开始自动创建，因此必须从堆栈跟踪的开始点引发异常。

所有异常都包含一个名为 [Message](https://msdn.microsoft.com/zh-cn/library/system.exception.message.aspx) 的属性。应该设置此字符串来解释发生异常的原因。注意，不应将安全敏感信息放在消息文本中。除 [Message](https://msdn.microsoft.com/zh-cn/library/system.exception.message.aspx) 之外，[ArgumentException](https://msdn.microsoft.com/zh-cn/library/system.argumentexception.aspx) 还包含一个名为 [ParamName](https://msdn.microsoft.com/zh-cn/library/system.argumentexception.paramname.aspx) 的属性，应将该属性设置为导致引发异常的参数的名称。对于属性设置器，[ParamName](https://msdn.microsoft.com/zh-cn/library/system.argumentexception.paramname.aspx) 应设置为 value。

公共的受保护方法应在其无法完成预期功能时引发异常。引发的异常类应该是符合错误条件的最确切的可用异常。这些异常应编写为类功能的一部分，派生类或对原始类的更新应保留相同的行为，以实现向后兼容性。

**引发异常时要避免的情况**

下表确定了在引发异常时要避免的做法：

* 不应使用异常来更改正常执行过程中的程序流程。异常只能用于报告和处理错误条件。
* 只能引发异常，而不能作为返回值或参数返回异常。
* 不要从自己的源代码中有意引发 [System.Exception](https://msdn.microsoft.com/zh-cn/library/system.exception.aspx)、[System.SystemException](https://msdn.microsoft.com/zh-cn/library/system.systemexception.aspx)、[System.NullReferenceException](https://msdn.microsoft.com/zh-cn/library/system.nullreferenceexception.aspx) 或 [System.IndexOutOfRangeException](https://msdn.microsoft.com/zh-cn/library/system.indexoutofrangeexception.aspx)。
* 不要创建可在调试模式下引发但不会在发布模式下引发的异常。若要在开发阶段确定运行时错误，请改用调试断言。

**定义异常类**

程序可以引发 [System](https://msdn.microsoft.com/zh-cn/library/system.aspx) 命名空间中的预定义异常类（前面注明的情况除外），或通过从 [Exception](https://msdn.microsoft.com/zh-cn/library/system.exception.aspx) 派生来创建它们自己的异常类。派生类至少应定义四个构造函数：一个是默认构造函数，一个用来设置消息属性，一个用来设置 [Message](https://msdn.microsoft.com/zh-cn/library/system.exception.message.aspx) 属性和 [InnerException](https://msdn.microsoft.com/zh-cn/library/system.exception.innerexception.aspx) 属性。第四个构造函数用于序列化异常。新异常类应该可序列化。例如：

[Serializable()]

public class InvalidDepartmentException : System.Exception

{

public InvalidDepartmentException() : base() { }

public InvalidDepartmentException(string message) : base(message) { }

public InvalidDepartmentException(string message, System.Exception inner) : base(message, inner) { }

// A constructor is needed for serialization when an

// exception propagates from a remoting server to the client.

protected InvalidDepartmentException(System.Runtime.Serialization.SerializationInfo info,

System.Runtime.Serialization.StreamingContext context) { }

}

仅当新属性提供的数据有助于解决异常时，才应将其添加到异常类。如果向派生的异常类添加了新属性，则应重写 ToString() 以返回添加的信息。