常量是在编译时已知并在程序的生存期内不发生更改的不可变值。常量使用 [const](https://msdn.microsoft.com/zh-cn/library/e6w8fe1b.aspx) 修饰符进行声明。只有 C# 内置类型（[System.Object](https://msdn.microsoft.com/zh-cn/library/system.object.aspx) 除外）可以声明为 **const**。有关内置类型的列表，请参见[内置类型表（C# 参考）](https://msdn.microsoft.com/zh-cn/library/ya5y69ds.aspx)。用户定义的类型（包括类、结构和数组）不能为 **const**。请使用 [readonly](https://msdn.microsoft.com/zh-cn/library/acdd6hb7.aspx) 修饰符创建在运行时初始化一次即不可再更改的类、结构或数组。

C# 不支持 **const** 方法、属性或事件。

可以使用枚举类型为整数内置类型（例如 **int**、**uint**、**long** 等等）定义命名常量。有关更多信息，请参见 [enum（C# 参考）](https://msdn.microsoft.com/zh-cn/library/sbbt4032.aspx)。

## 常量必须在声明时初始化。例如：

class Calendar1

{

public const int months = 12;

}

在此示例中，常量 months 始终为 12，不可更改，即使是该类自身也不能更改它。实际上，当编译器遇到 C# 源代码（例如 months）中的常量修饰符时，将直接把文本值替换到它生成的中间语言 (IL) 代码中。因为在运行时没有与常量关联的变量地址，所以 **const** 字段不能通过引用传递，并且不能在表达式中作为左值出现。

| **https://wizardforcel.gitbooks.io/msdn-csharp/content/img/note.jpg 注意** |
| --- |
| 当引用在其他代码如 DLL 中定义的常量值时应十分谨慎。如果新版本的 DLL 为常量定义了新的值，程序仍将保留旧的文本值，直到针对新版本重新编译程序。 |

## 可以同时声明多个相同类型的常量，例如：

class Calendar2

{

const int months = 12, weeks = 52, days = 365;

}

## 如果不会造成循环引用，用于初始化一个常量的表达式可以引用另一个常量。例如：

class Calendar3

{

const int months = 12;

const int weeks = 52;

const int days = 365;

const double daysPerWeek = (double) days / (double) weeks;

const double daysPerMonth = (double) days / (double) months;

}

## 常量可标记为 [public](https://msdn.microsoft.com/zh-cn/library/yzh058ae.aspx)、[private](https://msdn.microsoft.com/zh-cn/library/st6sy9xe.aspx)、[protected](https://msdn.microsoft.com/zh-cn/library/bcd5672a.aspx)、[internal](https://msdn.microsoft.com/zh-cn/library/7c5ka91b.aspx) 或 protected\*\*internal\*\*。

这些访问修饰符定义类的用户访问该常量的方式。有关更多信息，请参见 [访问修饰符（C# 编程指南）](https://msdn.microsoft.com/zh-cn/library/ms173121.aspx)。

因为常量值对该类型的所有实例是相同的，所以常量被当作 [static](https://msdn.microsoft.com/zh-cn/library/98f28cdx.aspx) 字段一样访问。不使用 **static** 关键字声明常量。

## 未包含在定义常量的类中的表达式必须使用类名、一个句点和常量名来访问该常量。例如：

int birthstones = Calendar.months;