Atitit. c# 语法新特性 c#2.0 3.0 4.0 4.5 5.0 6.0 attilax总结

[1. 版本历史 1](#_Toc16233)

[2. C#9 2](#_Toc19892)

[2.1. Dictionary Literals# 2](#_Toc18923)

[3. C# 8 2](#_Toc10997)

[3.1. Records and Pattern-based With-Expression# 2](#_Toc26108)

[3.1.1. Switch expressions Switch 表达式 3](#_Toc15589)

[3.2. C# 1.0-纯粹的面向对象 3](#_Toc30943)

[3.3. C# 2.0-泛型编程新概念 3](#_Toc14530)

[3.4. C# 2.0的另一个突出的特性就是匿名方法 3](#_Toc15316)

[3.5. C#3.0 linq 3](#_Toc20446)

[3.6. C# 4.0动态编程 dynamic 3](#_Toc19376)

[3.7. C# 4.5 异步编程 async和await 3](#_Toc16732)

[3.8. C# 5.0 更方便的一步编程 3](#_Toc12835)

[3.9. C# 6.0 中的新特性 4](#_Toc730)

[3.10. C# 6特征 (VS 2015) 4](#_Toc7740)

[3.11. C# 7 特征 (Visual Studio 2017) 5](#_Toc32628)

[3.12. C# 7.1 特征 (Visual Studio 2017 version 15.3) 5](#_Toc28132)

[4. 参考资料 7](#_Toc10172)

# 版本历史

| **语言版本** | **发布时间** | **.NET Framework要求** | **Visual Studio版本** |
| --- | --- | --- | --- |
| C# 1.0 | 2002.1 | .NET Framework 1.0 | Visual Studio .NET 2002 |
| C# 1.1\1.2 | 2003.4 | .NET Framework 1.1 | Visual Studio .NET 2003 |
| C# 2.0 | 2005.11 | .NET Framework 2.0 | Visual Studio 2005 |
| C# 3.0 | 2007.11 | .NET Framework 2.0\3.0\3.5 | Visual Studio 2008 |
| C# 4.0 | 2010.4 | .NET Framework 4.0 | Visual Studio 2010 |
| C# 5.0 | 2012.8 | .NET Framework 4.5 | Visual Studio 2012\2013 |
| C# 6.0 | 2015.7 | .NET Framework 4.6 | Visual Studio 2015 |
| C# 7.0 | 2017.3 | .NET Framework 4.6.2 | Visual Studio 2017 |
| C# 7.1 | 2017.6 | .NET Framework | Visual Studio 2017 v15.3预览版 |
| C# 8.0 | 待发布 |  |  |

# C#9

## Dictionary Literals#

引入更简单的语法来创建初始化的Dictionary <TKey，TValue>对象，而无需指定Dictionary类型名称或类型参数。使用用于数组类型推断的现有规则推断字典的类型参数。

Copy

// C# 1..8

var x = new Dictionary <string,int> () { { "foo", 4 }, { "bar", 5 }};

// C# 9

var x = ["foo":4, "bar": 5];

该特性使C＃中的字典工作更简单，并删除冗余代码。此外，值得一提的是，在F＃和Swift等其他编程语言中也使用了类似的字典语法

Records

# C# 8

## **Records and Pattern-based With-Expression[#](https://www.cnblogs.com/CoderAyu/p/11000386.html" \l "idx_1)**

这个功能我等待了很长时间,Records是一种轻量级的不可变类型,它可以是方法,属性,运算符等,它允许我们进行结构的比较, 此外,默认情况下,Records属性是只读的。

Records可以是值类型或引用类型。

### **Switch expressions Switch 表达式**

带有模式的 switch 语句在 C# 7.0 中非常强大，但编写起来很麻烦。switch 表达式是一个“轻量级”版本，其中所有情况都是表达式：

var area = figure switch

{

Line \_ => 0,

Rectangle r => r.Width \* r.Height,

Circle c => c.Radius \* 2.0 \* Math.PI,

\_ => throw new UnknownFigureException(figure)

};

## C# 1.0-纯粹的面向对象

## C# 2.0-泛型编程新概念

## C# 2.0的另一个突出的特性就是匿名方法

## C#3.0 linq

## C# 4.0动态编程 dynamic

## C# 4.5 异步编程 [async和await](http://blog.csdn.net/attilax/article/details/42014327" \l "t7)

## C# 5.0 更方便的一步编程

#### ****C# 5特性 (VS 2012)****

* Asynchronous methods：异步方法
* Caller info attributes：调用方信息特性，调用时访问调用者的信息

## C# 6.0 中的新特性

## ****C# 6特征 (VS 2015)****

* Compiler-as-a-service (Roslyn)
* Import of static type members into namespace：支持仅导入类中的静态成员
* Exception filters：异常过滤器
* Await in catch/finally blocks：支持在catch/finally语句块使用await语句
* Auto property initializers：自动属性初始化
* Default values for getter-only properties：设置只读属性的默认值
* Expression-bodied members：支持以表达式为主体的成员方法和只读属性
* Null propagator (null-conditional operator, succinct null checking)：Null条件操作符
* String interpolation：字符串插值，产生特定格式字符串的新方法
* nameof operator：nameof操作符，返回方法、属性、变量的名称
* Dictionary initializer：字典初始化

## ****C# 7 特征 (Visual Studio 2017)****

* Out variables：out变量直接声明，例如可以out in parameter
* Pattern matching：模式匹配，根据对象类型或者其它属性实现方法派发
* Tuples：元组
* Deconstruction：元组解析
* Discards：没有命名的变量，只是占位，后面代码不需要使用其值
* Local Functions：局部函数
* Binary Literals：二进制字面量
* Digit Separators：数字分隔符
* Ref returns and locals：引用返回值和局部变量
* Generalized async return types：async中使用泛型返回类型
* More expression-bodied members：允许构造器、解析器、属性可以使用表达式作为body
* Throw expressions：Throw可以在表达式中使用

## ****C# 7.1 特征 (Visual Studio 2017 version 15.3)****

* Async main：在main方法用async方式
* Default expressions：引入新的字面值default
* Reference assemblies：
* Inferred tuple element names：
* Pattern-matching with generics：

作者:: 绰号:老哇的爪子 （ 全名：：Attilax Akbar Al Rapanui 阿提拉克斯 阿克巴 阿尔 拉帕努伊 ） 汉字名：艾龙，  EMAIL:1466519819@qq.com

转载请注明来源： http://blog.csdn.net/attilax

我们可以对这些新特性一个一个的进行讨论，而首先要列出 C# 6.0 中这些特性的一个清单

自动的属性初始化器 Auto Property Initializer

主构造器 Primary Consturctor

字典初始化器 Dictionary Initializer

声明表达式 Declaration Expression

静态的Using Static Using

catch 块中的 await

异常过滤器 Exception Filter

用于检查NULL值的条件访问操作符

# 参考资料

Atitit.c# .net 3.5 4.0 4.5 5.0 6.0各个版本新特性战略规划总结 - attilax的专栏 - 博客频道 - CSDN.NET.htm

C# 6.0 的新特性 - 技术翻译 - 开源中国社区.htm

C# 5新特性详解之一——异步编程-CSDN.NET.htm

C# 语言历史版本特性（C# 1.0到C# 7.1汇总更新） - CSDN博客.mhtml