

在C#中，Dictionary的主要用途是提供快速的基于键的元素查找。Dictionary的结构一般是这样的：Dictionary<key, value>，它包含在System.Collections.Generic命名空间中。在使用Dictionary前，你必须对它的键类型和值类型进行声明。

想要更轻松有效地学习C#教程，建议您学习[C#微课](#)

要使用Dictionary集合，需要导入C#泛型命名空间
System.Collections.Generic（程序集：mscorlib）

Dictionary的描述

- 1、从一组键（Key）到一组值（Value）的映射，每一个添加项都是由一个值及其相关连的键组成
- 2、任何键都必须是唯一的
- 3、键不能为空引用null（VB中的Nothing），若值为引用类型，则可以为空值
- 4、Key和Value可以是任何类型（string, int, custom class 等）

Dictionary常用用法：以 key 的类型为 int，value的类型为string 为例

1、创建及初始化

```
Dictionary<int, string>myDictionary=new Dictionary<int, string>();
```

2、添加元素

```
myDictionary.Add(1, "C#");  
myDictionary.Add(2, "C++");  
myDictionary.Add(3, "ASP.NET");  
myDictionary.Add(4, "MVC");
```

3、通过Key查找元素

```
if(myDictionary.ContainsKey(1))  
{  
    Console.WriteLine("Key: {0}, Value: {1}", "1", myDictionary[1]);  
}
```

4、通过KeyValuePair遍历元素

```
foreach(KeyValuePair<int, string>kvp in myDictionary)  
... {  
    Console.WriteLine("Key = {0}, Value = {1}", kvp.Key, kvp.Value);  
}
```

5、仅遍历键 Keys 属性

```
Dictionary<int, string>.KeyCollection keyCol=myDictionary.Keys;
foreach(intkeyinkeyCol)
... {
Console.WriteLine("Key = {0}", key);
}
```

6、仅遍历值 Value属性

```
Dictionary<int, string>.ValueCollection valueCol=myDictionary.Values;
foreach(stringvalueinvalueCol)
... {
Console.WriteLine("Value = {0}", value);
}
```

7、通过Remove方法移除指定的键值

```
myDictionary.Remove(1);
if(myDictionary.ContainsKey(1))
... {
    Console.WriteLine("Key:{0},Value:{1}", "1", myDictionary[1]);
}
else
{
Console.WriteLine("不存在 Key : 1");
}
```

其它常见属性和方法的说明：

Comparer	获取用于确定字典中的键是否相等的 IEqualityComparer
Count	获取包含在 Dictionary中的键/值对的数
Item	获取或设置与指定的键相关联的值
Keys	获取包含 Dictionary中的键的集合
Values	获取包含 Dictionary中的值的集合
Add	将指定的键和值添加到字典中
Clear	从 Dictionary中移除所有的键和值
ContainsKey	确定 Dictionary是否包含指定的键
ContainsValue	确定 Dictionary是否包含特定值

GetEnumerator	返回循环访问 Dictionary的枚举数
GetType	获取当前实例的 Type （从 Object 继承）
Remove	从 Dictionary中移除所指定的键的值
ToString	返回表示当前 Object的 String （从 Object 继承）
TryGetValue ji	获取与指定的键相关联的值