**装箱和取消装箱（C# 编程指南）**

装箱是将[值类型](https://msdn.microsoft.com/zh-cn/library/s1ax56ch.aspx)转换为 **object** 类型或由此值类型实现的任何接口类型的过程。当 CLR 对值类型进行装箱时，会将该值包装到 System.Object 内部，再将后者存储在托管堆上。取消装箱将从对象中提取值类型。装箱是隐式的；取消装箱是显式的。装箱和取消装箱的概念是类型系统 C# 统一视图的基础，其中任一类型的值都被视为一个对象。

在下面的示例中，将整型变量 i 进行了装箱并分配给对象 o。

int i = 123;

// The following line boxes i.

object o = i;

然后，可以将对象 o 取消装箱并分配给整型变量 i：

o = 123;

i = (int)o; // unboxing

以下示例演示如何在 C# 中使用装箱。

// String.Concat example.

// String.Concat has many versions. Rest the mouse pointer on

// Concat in the following statement to verify that the version

// that is used here takes three object arguments. Both 42 and

// true must be boxed.

Console.WriteLine(String.Concat("Answer", 42, true));

// List example.

// Create a list of objects to hold a heterogeneous collection

// of elements.

List<object> mixedList = new List<object>();

// Add a string element to the list.

mixedList.Add("First Group:");

// Add some integers to the list.

for (int j = 1; j < 5; j++)

{

// Rest the mouse pointer over j to verify that you are adding

// an int to a list of objects. Each element j is boxed when

// you add j to mixedList.

mixedList.Add(j);

}

// Add another string and more integers.

mixedList.Add("Second Group:");

for (int j = 5; j < 10; j++)

{

mixedList.Add(j);

}

// Display the elements in the list. Declare the loop variable by

// using var, so that the compiler assigns its type.

foreach (var item in mixedList)

{

// Rest the mouse pointer over item to verify that the elements

// of mixedList are objects.

Console.WriteLine(item);

}

// The following loop sums the squares of the first group of boxed

// integers in mixedList. The list elements are objects, and cannot

// be multiplied or added to the sum until they are unboxed. The

// unboxing must be done explicitly.

var sum = 0;

for (var j = 1; j < 5; j++)

{

// The following statement causes a compiler error: Operator

// '\*' cannot be applied to operands of type 'object' and

// 'object'.

//sum += mixedList[j] \* mixedList[j]);

// After the list elements are unboxed, the computation does

// not cause a compiler error.

sum += (int)mixedList[j] \* (int)mixedList[j];

}

// The sum displayed is 30, the sum of 1 + 4 + 9 + 16.

Console.WriteLine("Sum: " + sum);

// Output:

// Answer42True

// First Group:

// 1

// 2

// 3

// 4

// Second Group:

// 5

// 6

// 7

// 8

// 9

// Sum: 30

**性能**

相对于简单的赋值而言，装箱和取消装箱过程需要进行大量的计算。对值类型进行装箱时，必须分配并构造一个新对象。取消装箱所需的强制转换也需要进行大量的计算，只是程度较轻。有关更多信息，请参见[性能](https://msdn.microsoft.com/zh-cn/library/ms173196.aspx)。

**装箱**

装箱用于在垃圾回收堆中存储值类型。装箱是[值类型](https://msdn.microsoft.com/zh-cn/library/s1ax56ch.aspx)到 **object** 类型或到此值类型所实现的任何接口类型的隐式转换。对值类型装箱会在堆中分配一个对象实例，并将该值复制到新的对象中。

请看以下值类型变量的声明：

int i = 123;

以下语句对变量 i 隐式应用了装箱操作：

// Boxing copies the value of i into object o.

object o = i;

此语句的结果是在堆栈上创建对象引用 o，而在堆上则引用 **int** 类型的值。该值是赋给变量 i 的值类型值的一个副本。下图说明了两个变量 i 和 o 之间的差异。

装箱转换

还可以像下面的示例一样执行显式装箱，但显式装箱从来不是必需的：

int i = 123;

object o = (object)i; // explicit boxing

**说明**

此示例使用装箱将整型变量 i 转换为对象 o。这样一来，存储在变量 i 中的值就从 123 更改为 456。该示例表明原始值类型和装箱的对象使用不同的内存位置，因此能够存储不同的值。

class TestBoxing

{

static void Main()

{

int i = 123;

// Boxing copies the value of i into object o.

object o = i;

// Change the value of i.

i = 456;

// The change in i does not effect the value stored in o.

System.Console.WriteLine("The value-type value = {0}", i);

System.Console.WriteLine("The object-type value = {0}", o);

}

}

/\* Output:

The value-type value = 456

The object-type value = 123

\*/

**取消装箱**

取消装箱是从 **object** 类型到[值类型](https://msdn.microsoft.com/zh-cn/library/s1ax56ch.aspx)或从接口类型到实现该接口的值类型的显式转换。取消装箱操作包括：

* 检查对象实例，以确保它是给定值类型的装箱值。
* 将该值从实例复制到值类型变量中。

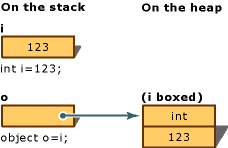
下面的语句演示装箱和取消装箱两种操作：

int i = 123; // a value type

object o = i; // boxing

int j = (int)o; // unboxing

下图演示上述语句的结果。



取消装箱转换

要在运行时成功取消装箱值类型，被取消装箱的项必须是对一个对象的引用，该对象是先前通过装箱该值类型的实例创建的。尝试取消装箱 **null** 会导致 [NullReferenceException](https://msdn.microsoft.com/zh-cn/library/system.nullreferenceexception.aspx)。尝试取消装箱对不兼容值类型的引用会导致 [InvalidCastException](https://msdn.microsoft.com/zh-cn/library/system.invalidcastexception.aspx)。

下面的示例演示无效的取消装箱及引发的 **InvalidCastException**。使用 **try** 和 **catch**，在发生错误时显示错误信息。

class TestUnboxing

{

static void Main()

{

int i = 123;

object o = i; // implicit boxing

try

{

int j = (short)o; // attempt to unbox

System.Console.WriteLine("Unboxing OK.");

}

catch (System.InvalidCastException e)

{

System.Console.WriteLine("{0} Error: Incorrect unboxing.", e.Message);

}

}

}

此程序输出：

Specified cast is not valid.Error: Incorrect unboxing.

如果将下列语句：

int j = (short) o;

更改为：

int j = (int) o;

将执行转换，并将得到以下输出：

Unboxing OK.