**在查询表达式中处理 Null 值（C# 编程指南）**

此示例演示如何处理源集合中可能的 null 值。诸如 [IEnumerable<T>](https://msdn.microsoft.com/zh-cn/library/9eekhta0.aspx) 等对象集合可能包含值为 [null](https://msdn.microsoft.com/zh-cn/library/edakx9da.aspx) 的元素。如果源集合为 null 或包含值为 null 的元素，并且查询未处理 null 值，当您执行查询时将会引发 [NullReferenceException](https://msdn.microsoft.com/zh-cn/library/system.nullreferenceexception.aspx)。

您可以采用防御方式进行编码以避免 null 引用异常，如下面的示例中所示：

var query1 =

from c in categories

where c != null

join p in products on c.ID equals

(p == null ? null : p.CategoryID)

select new { Category = c.Name, Name = p.Name };

在前面的示例中，**where** 子句筛选出类别序列中的所有 null 元素。此技术与 join 子句中的 null 检查无关。此示例中包含 null 的条件表达式将发挥作用，因为 Products.CategoryID 的类型为 **int?**（**Nullable<int>** 的简写形式）。

在 join 子句中，只要其中一个比较键是可以为 null 的类型，您就可以在查询表达式中将另一个比较键强制转换成可以为 null 的类型。在下面的示例中，假定 EmployeeID 是一个列，其中包含类型为 **int?** 的值：

void TestMethod(Northwind db)

{

var query =

from o in db.Orders

join e in db.Employees

on o.EmployeeID equals (int?)e.EmployeeID

select new { o.OrderID, e.FirstName };

}