VisualBasic构建线程安全的Singleton PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/252/2021_2022_VisualBasi_c97_252679.htm 大抵而言，Singleton 模式应该是设计模式中相当常用的一种了。它能够节省宝贵的 CPU 或内存资源，避免不必要的创建对象开销。然而，在多线程应用中，对于那些非线程安全的数据库组件，传统的 Singleton 将容易造成不稳定。如果使用 Synclock 进行同步，性能损耗可能将更大，特别在并发访问高的 ASP.net 应用中。 前十几天和网友聊天的时候，他把 ThreadwiseSingleton 发给了我，我随后改写成泛型类。事实上，它基于 Threadslot 构建，因此达到了线程隔离的效果。您需要传入一个 Func(Of TResult) 完成初始化的工作。如果您的类中含有非托管资源，并无法预见可能的错误时，请使用一个 Try...Finally... 包裹将要运行的程序。同时感到抱歉的是，拖延了很久很久才发出来。Imports System.Threading 一个线程隔离的 Singleton。 Public Class ThreadwiseSingletonClass ThreadwiseSingleton(Of T As IDisposable)Private Shared _Factory As Func(Of T) 获取构建此实例的工厂类。 Public Shared Property Factory()Property Factory() As Func(Of T) Get Return _Factory End Get Set(ByVal value As Func(Of T)) _Factory = value End SetEnd Property 获得线程中的唯一实例。 Public Shared ReadOnly Property Instance()Property Instance() As T Get Dim threadSlot As LocalDataStoreSlot = Thread.GetNamedDataSlot(GetType(T).ToString) Dim threadSlotObj As Object = Thread.GetData(threadSlot) If threadSlotObj Is Nothing Then Create singleton instance Dim ins As

T = Factory.Invoke Thread.SetData(threadSlot, ins) Return ins Else Return DirectCast(threadSlotObj, T) End If End GetEnd Property 私有的构造函数。 Private Sub New()Sub New()End Sub 释放此 Singleton 实例使用的资源。请不要直接调用 Instance.Dispose() 。 Public Shared Sub Dispose()Sub Dispose() Instance.Dispose() 放空槽 Dim threadSlot As LocalDataStoreSlot = Thread.GetNamedDataSlot(GetType(T).ToString)Thread.SetData(threadSlot, Nothing) End SubEnd Class 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com