# 视图

## 创建自定义视图引擎

一般情况下直接使用MVC框架自带的内建视图引擎即可，但如果想知道视图引擎是如何工作的，就需要从建立一个自定义视图引擎开始了。通过之前的学习我们都知道了内建视图引擎包括Razor和ASPX两种，ASPX是针对旧版本MVC程序的，他主要是维护旧版本MVC应用程序，保持系统的兼容性而保留的Web Form视图引擎；Razor是在MVC3引入的，它的语法更加简洁。

现在我们就先从自定义视图引擎开始，了解一下视图引擎的工作机制。视图引擎的接口是IViewEngine，其结构如下：

命名空间：System.Web.Mvc

方法：

1. FindPartialView

参数[类型]：

* controllerContext[ControllerContext]
* partialViewName[string]
* useCache[bool]

返回值：ViewEngineResult

1. FindView

参数[类型]：

* controllerContext[ControllerContext]
* viewName[string]
* masterName[string]
* useCache[bool]

返回值：ViewEngineResult

1. ReleaseView

参数[类型]：

* controllerContext[ControllerContext]
* view[IView]

返回值：ViewEngineResult

前两个方法（FindPartialView、FindView）接收的参数是描述请求的：处理该请求的控制器、视图名及布局。当框架对ViewResult进行处理时，会调用这两个方法。最后一个方法（ReleaseView）在视图不再需要时被调用，其功能就是要释放视图所占用的资源。

注：MVC框架对视图引擎的支持是由ControllerActionInvoker（控制器动作调用器）类实现的，这是IActionInvoker接口的内建实现。如果已经直接通过IActionInvoker或IControllerFactory接口实现了自己的动作调用器或控制器工厂，将无法自动地访问视图引擎特性。

当请求一个视图时，ViewEngineResult类使试图引擎能够对MVC框架作出响应。当视图引擎能够对请求提供视图时，将通过如下构造函数创建一个ViewEngineResult：

public ViewEngineResult(IView view, IViewEngine viewEngine)

当视图不能对请求提供视图时，则使用如下构造函数：

public ViewEngineResult(IEnumerable<string> searchedLocations)

该重载版本的构造函数是通过参数的视图位置的集合进行枚举查找并创建ViewEngineResult的，如果找不到视图，则该枚举的信息会显示给用户。

视图引擎系统的最后一个构造块是IVew接口：

namespace System.Web.Mvc

{

public interface IView

{

void Render(ViewContext viewContext, TextWriter writer);

}

}

该接口中定义的Render方法的ViewContext类型参数传递了客户端请求的信息，以及动作方法的输出。TextWriter类型参数则用于将输出写给客户端。

在了解了视图引擎的构造组成后，就来创建一个简单的视图引擎做一下深入的研究，我们的视图引擎简单到何种地步呢，我们只让其返回一个视图，该视图将渲染关于请求的信息，以及动作方法产生的视图数据。这样一来既能演示视图引擎的操作方式，也不会陷入解析视图模板的困境。

### 创建示例项目

项目模板：Empty

项目名称：Views

控制器：Home

Home控制器代码如下：

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace Views.Controllers

{

public class HomeController : Controller

{

//

// GET: /Home/

public ActionResult Index()

{

ViewData["Message"] = "Hello, World";

ViewData["Time"] = DateTime.Now.ToShortTimeString();

return View("DebugData");

}

public ActionResult List()

{

return View();

}

}

}

### 实现自定义的IView

### 实现自定义的IViewEngine

### 注册自定义视图引擎

### 测试自定义视图引擎