Контроллеры

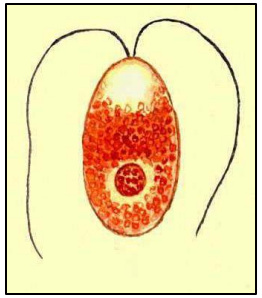
ASP.NET MVC 4.0

2013

Цель

- Ближе познакомиться с возможностями контроллеров.
- Добавить контроллеры для функций администратора в приложение ArtMusem.

Интерфейс IController



```
public interface IController
{
    void Execute(RequestContext requestContext);
}
```

Простейший контроллер

```
public class ProtoController: IController
{
    public void Execute(System.Web.Routing.RequestContext requestContext)
    {
        requestContext.HttpContext.Response.Write("Hello, world!");
    }
}
```

Контроллером является любой класс, который:
- реализует интерфейс IController
- не является обобщенным

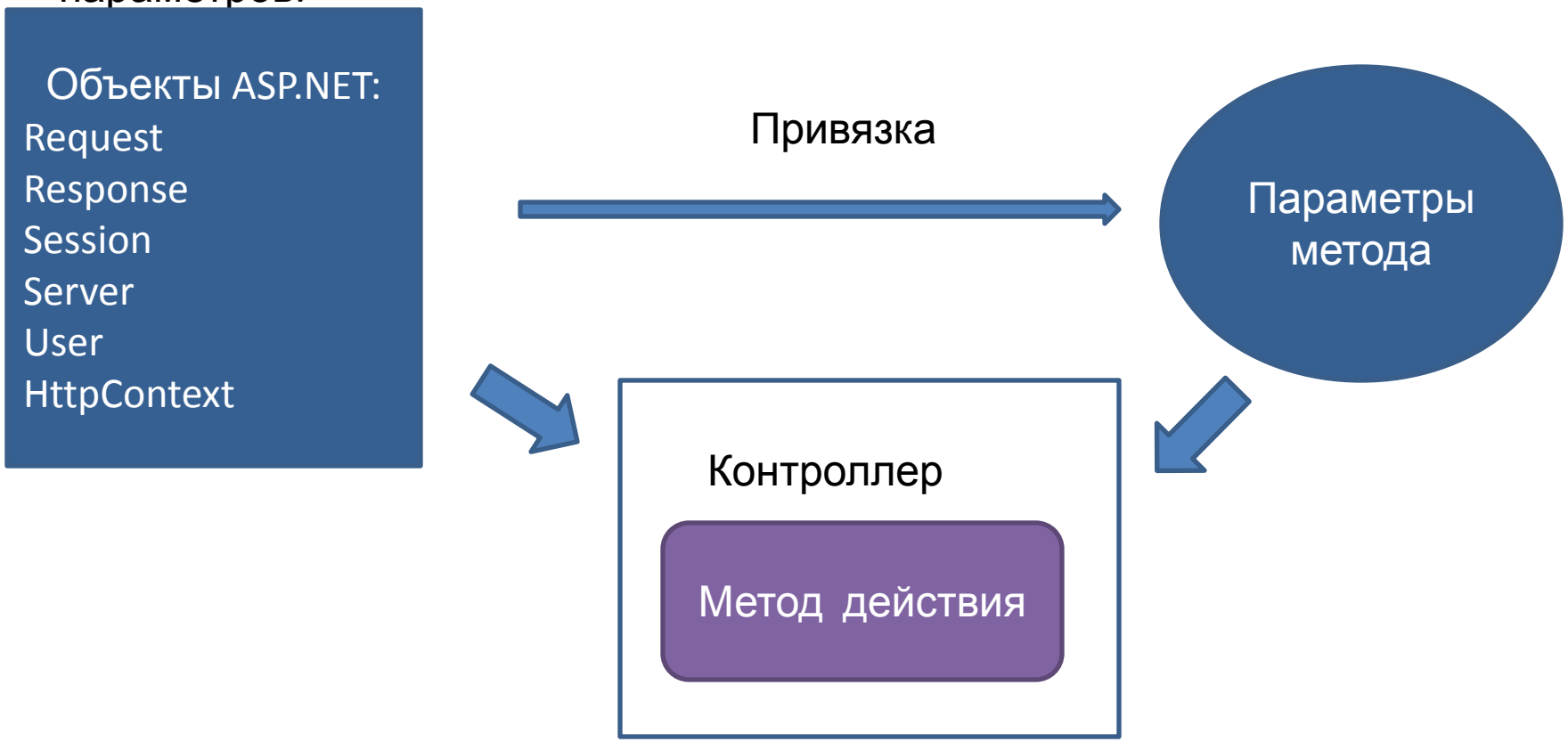
Состав контроллера

- **Методы действий** – открытые методы контроллера, которые вызываются по web-запросам.
- **Результаты действий** – объекты-описатели действий. Сами действия будут совершены позднее.
- **Фильтры** – специальные обработчики событий, которые предшествуют или следуют за действиями.

```
[OutputCache(Duration=600, VaryByParam="*")]  // фильтр
public class DemoContriller: Controller
{
    public ViewResult ShowGreeting()          // метод действия
    {
        ViewBag.Greeting = "Hello, world!";
        return View();                        // возвращает результат действия
    }
}
```

Вход контроллера

Методы-действия получают входные данные из контекста и из своих параметров.



Методы действий не могут иметь параметры out и ref.

Объект Request

```
public ActionResult Index()
{
    if (Request.Files.Count > 0)
    {
        Stream stream = Request.Files["111.png"].InputStream;
    }
    string command = Request.RequestType; // GET, POST
    string url = Request.Url.AbsoluteUri; // http://localhost:52262/
    string agent = Request.UserAgent;
              // Mozilla/5.0 (Windows NT 6.1; WOW64)
              // AppleWebKit/537.22 (KHTML, like Gecko)
              // Chrome/25.0.1364.172
              // Safari/537.22
}
```

Объект Response

```
public ActionResult Index()
{
    Response.Charset = "utf-8";
    Response.Cache.SetCacheability(HttpCacheability.Public);
    Response.Cache.SetExpires(DateTime.Now.AddSeconds(5));
    Response.Output.WriteLine("Текущее время: " + DateTime.Now);
    Response.StatusCode = 404;
    return null;
}
```

Посмотреть заголовки в браузере Chrome: F12 / Network / Headers.

Объект Session

```
Session["key"] = "Любые данные";
```

Объект Server

```
public ActionResult Index()
{
    string path = Server.MapPath("~/");     // "D:\\111\\MvcController\\MvcController\\"
    int timeout = Server.ScriptTimeout;     // 110
    Server.ClearError();
    Exception lastError = Server.GetLastError();       // null
    string encodedStr = Server.HtmlEncode("<script>"); // &lt;script&gt;
    string encodedUrl = Server.UrlEncode("host?name=Вася&id=5");
              // host%3fname%3d%d0%92%d0%b0%d1%81%d1%8f%26id%3d5

    return null;
}
```

Привязка моделей

Модель:

```
public class MyModel
{
    public string Name { set; get; }
}
```

Автоматическая привязка

```
public string Index(MyModel model)
{
     return model.Name;
}
```

Ручная привязка

```
public string Index()
{
    MyModel model = new MyModel();
    this.UpdateModel<MyModel>(model);
    return model.Name;
}
```

Если для ссылочных параметров не находится значений, им присваивается null. Если то же происходит для значимых параметров, выбрасывается исключение.

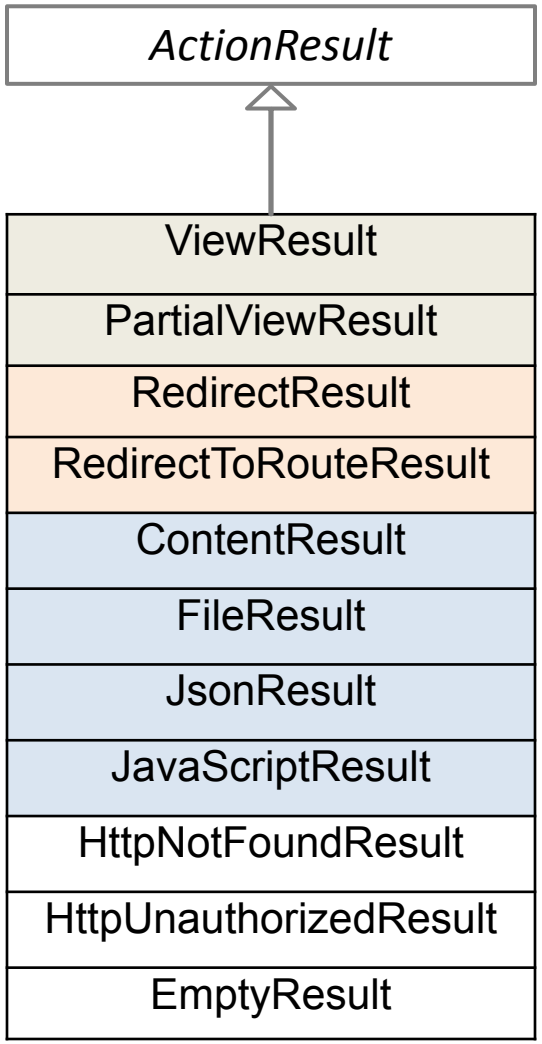
Чтобы избежать исключений, параметрам, которые рискуют не получить значения, следует задавать значения по умолчанию.

Разновидности вывода

Метод действия возвращает объект описания ответа.

Различают три типа ответа:

1. HTML разметка.
2. Перенаправление на другой URL или метод действия.
3. Текстовые или двоичные данные в потоке ответа.



Согласно правилам ООП, из методов-действий нужно возвращать максимально конкретный тип, т.е. ViewResult, а не ActionResult.

Производные классы от ActionResult

Тип действия	Содержание действия	Метод класса Controller
ViewResult	рендерит шаблон представления	return View()
PartialViewResult	то же для частичного представления	return PartialView()
RedirectResult	ответ 302: перенаправление на метод действия	return RedirectToAction() return RedirectToRoute()
RedirectToRouteResult	ответ 302: перенаправление на url	return Redirect()
ContentResult	текстовые данные	return Content()
FileResult	двоичные данные	return File()
JsonResult	текст в нотации json	return Json()
JavaScriptResult	код javascript	return JavaScript()
HttpNotFoundResult	ответ 404	return HttpNotFoundResult()
HttpUnauthorizedResult	ответ 401, запускающий механизм аутент.	return HttpUnauthorizedResult()
EmptyResult	нет действия	return new EmptyResult();

Перенаправление

Метод RedirectResult приводит к ответу 302 и выдаче браузером нового запроса GET с измененным URL. Напрямую вызывать метод контроллера не следует.

```
return RedirectToAction("Index");

return RedirectToAction("Index", "Product", new{color="red", size="xxl"});

return RedirectToRoute("Default", new{color="red", size="xxl"});
```

Перенаправление к другому URL

```
return Redirect("https://www.google.com.ua/");

return Redirect("~/Content/themes/base.html");
```

Для краткосрочного хранения данных между двумя запросами существует коллекция **TempData**. Она работает подобно сессии, но принудительно очищается после нового запроса.

Возврат текстовых данных

Чтобы вернуть любые текстовые данные, потребуется задать:

1. Сами данные в виде экземпляра String.
2. Заголовок content-type. Эти значения нужно брать из перечислителя System.Net.Mime.MediaTypeNames.
3. Кодировку текста. Если кодировка не указана, будет сделана та, что потребует браузер.

```
public ContentResult ActionMethod()
{
     // do something here ...
     return Content( content, contentType, contentEncoding);
     // return "Hello";
     //return 345;
}
```

Из метода действия можно вернуть простую строку. Она будет автоматически преобразована в вызов метода Content() с одним параметром.

Из метода действия можно вернуть любой объект. Будет вызван его метод ToString() и далее все, что сказано о строке.

Возврат JSON и javascript

Чтобы передать данные в формате JSON, нужно вызвать метод Json(), передав ему любой объект. Объект будет преобразован в строку в формате JSON.

Возврат команд JavaScript

Обычно требуется при обработке Ajax запросов.

```
return JavaScript("alert('abcd')");
```

JavaScriptResult – это тот же ContentResult, но с уже установленным заголовком content-type

Возврат файлов и двоичных данных

FileResult – это абстрактный базовый класс для всех результатов действий, связанных с двоичными данными. Последние могут быть:

1) файлом на сервере,
2) массивом байтов,
3) открытым потоком system.IO.Styream.

Пример – отправка клиенту массива байтов:

```
byte[] data = …
return File(data, "application/pdf", "Report.pdf");
```

Контроллер сам определит, какого типа результат создавать, в зависимости от первого параметра метода File().

Поиск шаблона представления

```
return View("Index", (object)"Home");

return View("Index");

return View();
```

~/Views/Home/Index.cshtml

~/Views/Shared/Index.cshtml

Если *контроллер* – имя контроллера, а *шаблон* – имя шаблона, то поиск представления будет производиться по путям:

~/Views/контроллер/шаблон.cshtml

~/Views/Shared/шаблон.cshtml

```
return
    View("~/Vies/Abc/Def.cshtml");
```

~/Vies/Abc/Def.cshtml

При указании полного пути представление может находиться где угодно, но расширение файла должно быть cshtml.

Использование фильтров

Фильтры – это атрибуты контроллеров и их методов, которые позволяют вставить дополнительную функциональность в процесс обработки запроса. Вставить можно:

a) перед и после запуска методов,
b) перед и после выполнения результатов действий,
c) в случае необработанного исключения.

Самостоятельно

Добавить контроллер AdminHalls в приложение ArtMuseum.

- Контроллер должен поддерживать все операции с залами музея: добавить, удалить, изменить, получить список залов. Поскольку операции стандартные, они реализованы в шаблонах студии.

- Чтобы воспользоваться этой реализацией, нужно создавать контроллер не вручную, а при помощи меню Add / Controller.

- В диалоговом окне указать шаблон кода “MVC controller with read/write actions and views using EF”, а также классы модели и контекста данных EF.

- В ответ будет создан не только контроллер, но и все нужные представления.