모델, 뷰, 컨트롤러

MVC 패턴

Controller: Client의 Http 요청에 대한 로직 담당

Model: 비즈니스 로직

View: User Interface

컨트롤러 클래스

- 컨트롤러 클래스를 상속 받은 사용자 컨트롤러 클래스는 HTTP 요청, 입력, 응용프로그램 로직 수행
- 실제 내용을 구현하는 메서드는 public 접근 제한자를 가진 액션(Action) 메서드

액션 메서드

- 컨트롤러 안에서 뷰를 호출하거나 데이터를 전달하는 메서드
- Startup.cs에 명시된 라우터 규칙에 맞게 요청받은 URL를 통해 해당 컨트롤러와 액션 메서드에 접근
- Example: http://localhost:55305/Home/About (Home 컨트롤러의 About 액션 메서드)
- ViewData 혹은 ViewBag 등의 개체를 통해 View 페이지에 값 전달

ViewData

- Key-Value Dictionary Collection
- ViewBag 보다 빠름
- MVC 1.0에 도입
- NET Framework 3.5 이상에서 작동

```
public IActionResult About()
{
    ViewData["Message"] = "Your application description page.";

    return View();
}
```

```
<h3>@ViewData["Message"]</h3>
```

```
Shoes shoes = new Shoes { maker = "Nike", size = 255 };
ViewData["shoes"] = shoes;
```

```
@{
    var ShoesData = ViewData["shoes"] as Shoes;
}
<h2>@ShoesData.size</h2>
```

ViewBag

- dynamic object Type
- ViewData 보다 느림
- MVC 3.0에 도입
- .NET Framework 4.0 이상에서 작동

```
Shoes shoes = new Shoes { maker = "Nike", size = 255 };
ViewBag.shoes2 = shoes;
```

```
<h2>@ViewBag.shoes2.size</h2>
```

컨트롤러와 액션 메서드

```
public class HomeController : Controller
{
    public void Index() {
    }

    public IActionResult About() {
        return View();
    }

    public string StringAction() {
        return "string을 반환하는 액션 메서드";
    }

    public DateTime DateTimeAction() {
        return DateTime.Now;
    }

    public IActionResult DefaultAction() {
        return View();
    }
}
```

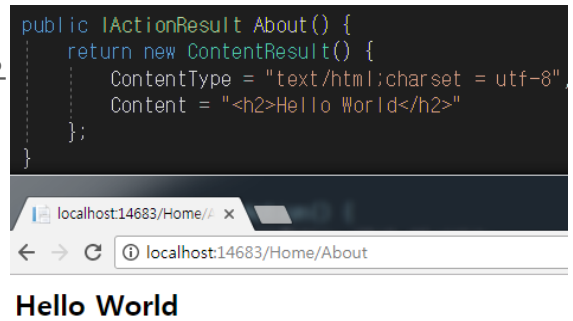
액션 반환값

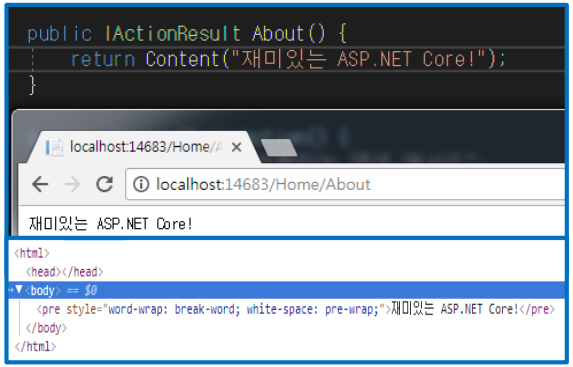
1. **return View();**

- 액션 메서드에 해당하는 뷰 페이지 실행(액션 메서드 이름 = 뷰 페이지 이름)

1. **return RedirectToAction(URL);**

- 특정 액션메서드로 이동 => RedirectToAction(“Index”, “Home”);
- HomeController의 Index 액션 메서드로 이동
- 컨트롤러 명시 하지 않을 경우





1. **return ContentResult();**

View

HomeController의 Index 액션 메서드를 호출 할 경우

1. /Views/Home/Index.cshtml 페이지 검색
2. /Views/Shared/Index.cshtml 페이지 검색

레이저 표현식

- 기존 웹 폼에서 사용하던 <% %> 스타일을 웹 폼 뷰 엔진
- @를 사용하여 서버측 C# 코드를 출력
- @@ or @(“@”)는 @ 기호 자체를 출력
- <a href=”mailto:me@dul.me”> 는 이메일 주소로 인식

```
public class HomeController : Controller
{
    public IActionResult Index() {
        return View();
    }
}
```

```
@{
    var sum = 155;
    var msg = "안녕하세요";
    var file = @"C:₩asp.net₩";
    var quote = @"안녕하세요. ""asp.net core";
 }
<h2>@sum</h2>
<pre>
    이메일: aspnet@dotet.com, a@(“@”)a.com
    트위터: @@aspnet
</pre>
```

155

이메일: aspnet@dotet.com, a@a.com
트위터: @aspnet

View - Html.Raw()

```
public IActionResult About() {
    ViewBag.strHtml = "안녕하세요?<br/> 잘부탁드립니다.";
    return View();
}
```

```
<h2>@ViewBag.strHtml</h2>
<h2>@Html.Raw(ViewBag.strHtml)</h2>
```

안녕하세요?
 잘부탁드립니다.

안녕하세요?
잘부탁드립니다.

_Layout.cshtml

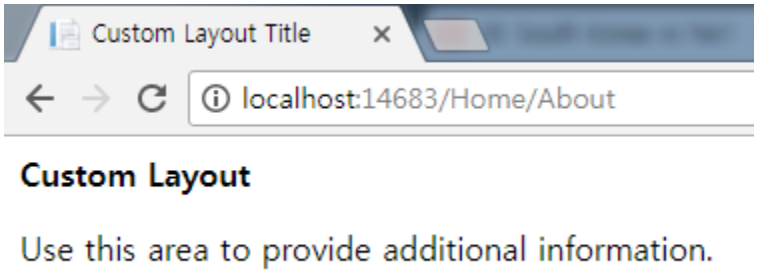
- ASP.NET 4.6 웹 폼의 마스터 페이지와 동일
- /Views/Shared 폴더에 존재

```
<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>
</head>
<body>
    <div>
        <span><strong>Custom Layout</strong></span>
    </div>
    <div>
        @RenderBody()
    </div>
</body>
</html>
```

```
@{
    Layout = "_CustomLayout";
    ViewBag.Title = "Custom Layout Title";
}

<p>Use this area to provide additional information.</p>
```



RenderSection(이름, 필수 여부)

```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>@ViewBag.Title</title>
    @RenderSection("alertFunc", false)
</head>
<body>
    <div>
        <span><strong>Custom Layout</strong></span>
    </div>
    <div>
        @RenderBody()
    </div>
</body>
</html>
```

```
@{
    Layout = "_CustomLayout";
    ViewBag.Title = "Custom Layout Title";
}

@section alertFunc{
    <script>alert(1);</script>
}

<p>Use this area to provide additional information.</p>
```

```
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Custom Layout Title</title>

    <script>alert(1);</script>

</head>
<body>
    <div>
        <span><strong>Custom Layout</strong></span>
    </div>
    <div>


<p>Use this area to provide additional information.</p>


    </div>
```

_ViewStart, _ViewImports.cshtml

_ViewStart.cshtml

- 각각의 .cshtml 페이지에 Layout 속성값을 지정 하는 대신 _ViewStart.cshtml에 한번만 지정하여 모든 페이지에 동일한 Layout 적용

_ViewImports.cshtml

- 각각의 .cshtml 페이지에 네임스페이스를 지정하는 대신 _ViewImports.cshtml에 한번만 지정하여 모든 페이지에 정의한 네임스페이스 적용

Partial View

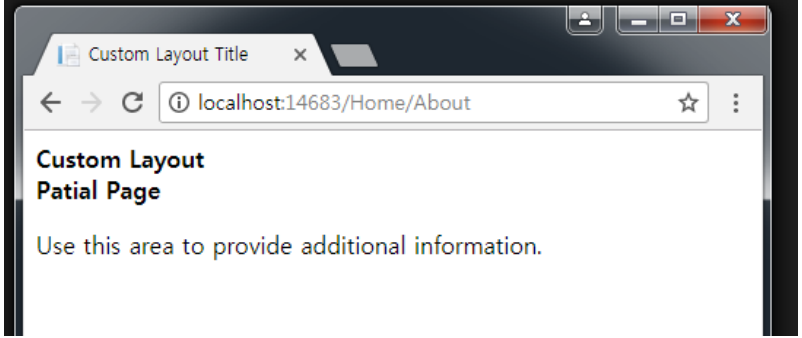
_PartialPage.cshtml

```
1   <span><strong>Patial Page</strong></span>
```

```
@{
    Layout = "_CustomLayout";
    ViewBag.Title = "Custom Layout Title";
}

@Html.Partial("_PartialPage")

<p>Use this area to provide additional information.</p>
```



Model

- DTO(Data Transfer Object)
- Entity
- Business Object
- Domain Model
- View Model
- Presentation Model

```
public class Shoes
{
    [StringLength(30)]
    public string maker { get; set; }

    [Range(200, 300)]
    public int size { get; set; }

    [Required]
    [DataType(DataType.EmailAddress)]
    [Display(Name = "Email address")]
    public string emailAddress { get; set; }
}
```

Attribute

- DataType (Password, CreditCard, Currency, EmailAddress, Url 등)
- Display (Label에 표시할 Text)

TempData

- 값을 저장 후 TempData 값을 요청하면 값이 사용되자마자 사라지는 형태의 데이터를 보관 (Ex. 에러메시지)
- TempData는 Session에 저장하기 때문에 Microsoft.AspNetCore.Session 및 Microsoft.Framework.Caching.Memory에 대한 참조 필요
- Startup.cs 파일의 ConfigureService 메서드에 service.AddMomoryCache(), service.AddSession() 메서드 추가
- Startup.cs 파일의 Configure 메서드에 app.UseSession() 미들웨어 추가

Environment Helper

```
<environment names="Development">
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.css" />
    <link rel="stylesheet" href="~/css/site.css" />
</environment>
<environment names="Staging,Production">
    <link rel="stylesheet" href="https://ajax.aspnetcdn.com/ajax/bootstrap/3.3.6/css/bootstrap.min.css"
          asp-fallback-href="~/lib/bootstrap/dist/css/bootstrap.min.css"
          asp-fallback-test-class="sr-only" asp-fallback-test-property="position" asp-fallback-test-value="absolute" />
    <link rel="stylesheet" href="~/css/site.min.css" asp-append-version="true" />
</environment>
```

- ASPNETCORE_ENVIRONMENT를 참조하여 운영중인 애플리케이션 환경을 가져옴
- 대소문자 구분 안함
- Properties/launchSettings.json 파일에 저장