ASP.NET MVC

C#

ASP.NET MVC

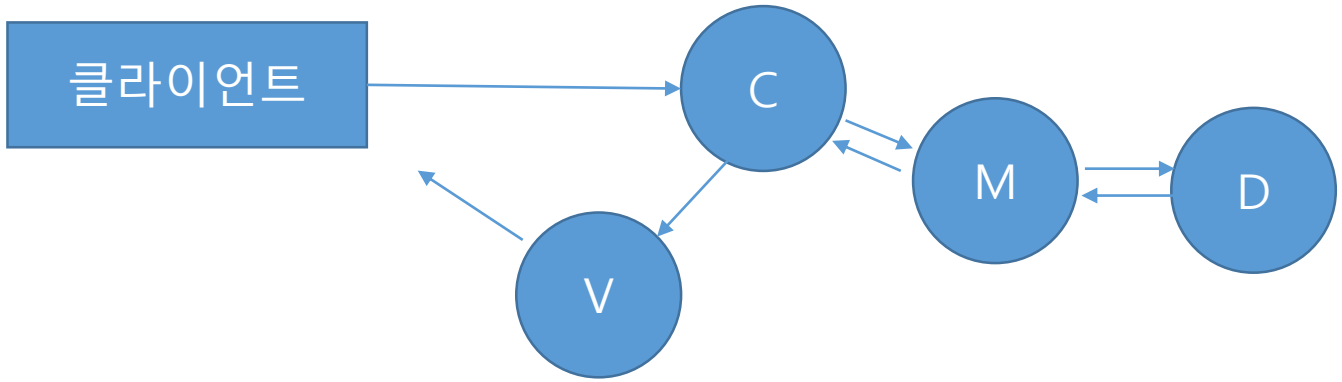
MVC패턴의 이해

MVC란 Model, View, Controller를 뜻하는 용어로 개발 형태의 일종

Model: 데이타 엑세스, 비지니스 로직, 유틸러티 클래스 등을 갖는다.
데이터베이스와의 관계를 담당.
클라이언트의 요청에서 필요한 자료를 데이터베이스로부터 추출하거나,
수정하여 Controller로 전달.

View : 사용자한테 보여지는 UI 화면.
데이터를 html 태그와 결합하여 표현. Controller에서 어떤 View 컴포넌트를 보여줄지 결정.

Controller: 클라이언트의 요청을 받고, 적절한 Model에 지시를 내리며,
Model에서 전달된 데이터를 적절한 View에 전달



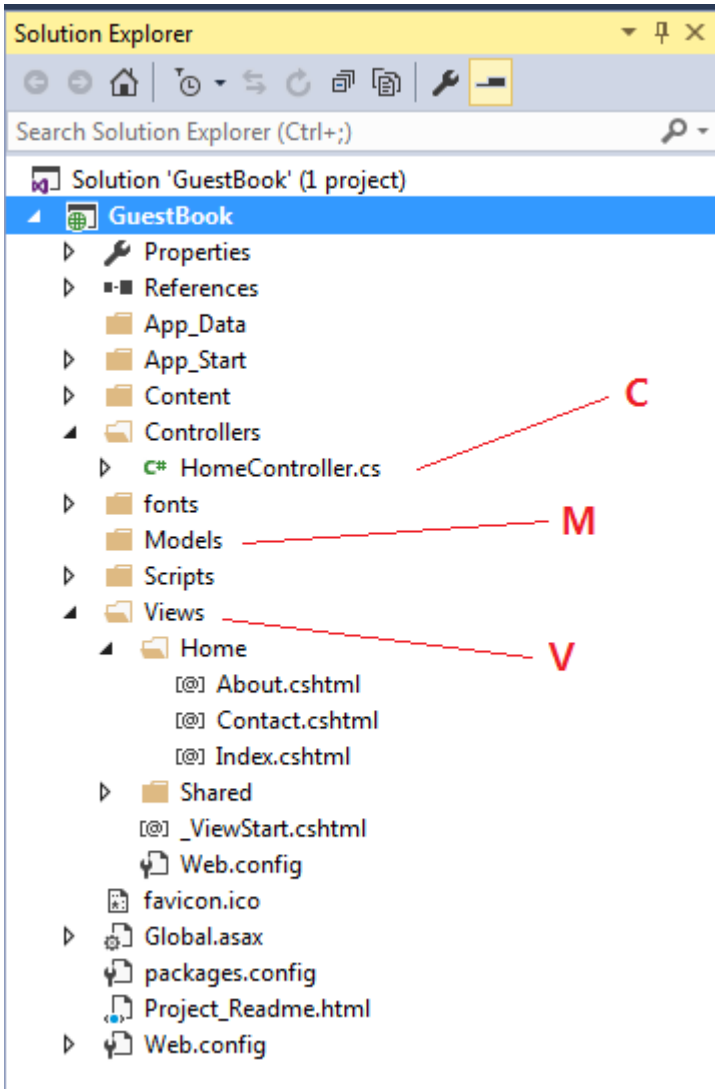
ASP.NET MVC패턴

Webform 보다 더 가볍고 모던한 패턴이다.

기본으로 생성되는 폴더 Models, Views, Controllers 는 이름 고정
(변경 불가)

Views 폴더 – Controller 이름의 서브폴더 생성하고 view 파일을 생성

/[Controller]/[ActionName]/[Parameters]



MVC Controller

사용자의 웹 Request 를 받아들이고 결과를 보내는 역할

웹 Request의 파라미터나 POST 데이터를 사용해서
=> 적절한 Model에서 데이터를 가져오거나 저장 하고
=> 결과 데이터를 View에 전달하여 rendering 된 HTML을 생성해서 보낸다.

Model 및 View와 긴밀하게 상호작용하면서 전체 흐름을 제어

Controller 안의 각 액션메소드는 하나의 웹 페이지에 해당 /컨트롤러명/메서드명
⇒ Action메소드는 Request를 받고, 결과물인 ActionResult 객체를 리턴
⇒ Controller.View()는 MVC Framework에서 지정한 /views/컨트롤러명/메서드명.cshtml 을 랜더링한 결과인 ViewResult 객체를 리턴
⇒ HTML, 파일, 문자열, JSON, javascript 등을 리턴하거나 Redirect 할 수 있다.

View로 데이터를 전달하기 위해서는 ViewBag(dynamic), ViewData(딕셔너리), Model 객체를 넘겨준다.
⇒ Controller.View(모델객체)
=> view 페이지 : @model 지정, Model.속성명
⇒ ViewBag.Title =“회원가입”
=> view 페이지 : @ViewBag.Title
⇒ ViewDara[“ProductName”] = “빼빼로”
=> view 페이지 : @ViewDara[“ProductName”]

MVC Model

비즈니스 로직이나 데이터 액세스를 위한 클래스로 구성 (.cs 파일)

일반적으로 DAC으로 구성되었던 클래스의 메소드와 동일

MVC View

View 파일을 랜더링하여 HTML 파일을 만들어 리턴한다.

HTML, CSS, MVC Helper 함수(View Engine)들을 제공한다.

가장 많이 사용되는 View Engine은 Razor 엔진으로, HTML과 Razor 문법을 함께 사용한다.

⇒ MVC4 이하에서는 VS 템플릿에서 Razor 뷰 엔진과 WebForm 뷰 엔진을 기본적으로 제공하였으나, MVC5에서는 Razor 뷰만 기본적으로 지원

⇒ Controller의 액센메서드에서 뷰파일을 추가 생성한다.

View Layout

⇒ 마스터페이지와 같은 기본 마스터 템플릿을 계층적으로 구성할 수 있다.
⇒ _Layout.cshtml 로 /views/Shared 폴더에 위치

Partial View

⇒ View의 특정 부분을 별도의 부분 view 로 만들어 사용 (웹폼의 .ascx와 같은 개념)

Razor

웹페이지에서 서버 기반의 코드를 끼워 넣기 위한 마크업 구문

Razor 마크업(@) + C# 코드 + Html 태그

Razor 구문이 작성된 파일은 .cshtml 파일 확장자

C# 사용을 위해서 @{ } 블록을 사용 (html 코드블럭의 주석은 @* ~~ *@)

조건문, 반복문 사용 가능
=> 처음 시작하는 부분에만 @를 붙이고, C# 처럼 조건문을 사용하면서 html을 사용할 수 있다.

```
aspx 페이지 :   <h1>a:<%=a %>, b:<%=b %>, c:<%=c %></h1>
Razor  :        <h1>a:@a, b:@b, c:@c</h1>

aspx 페이지 :    <% foreach(var p in products) { %>
                        <li><%=p.name%>(<%=p.price%>)</li>
                 <% } %>

Razor :          @foreach(var p in products) {
                        <li>@p.name(@p.price)</li>
                 }
```


Visual C#