



19

# ASP.NET MVC

모던 웹을 위한 Javascript jQuery 입문

# 19.1 Ajax 개요

## ❖ ASP.NET MVC란?

- 마이크로소프트에서 만든 서버 개발 프레임워크
- C#이나 베이직을 사용해 개발 가능

## ❖ Ajax란?

- 특정 언어나 프레임워크가 아니라 프로그램을 구현하는 방식

그림 19-1 Ajax예시: 네이버 SE



## ❖ 검색 사이트에서 자동 완성 기능

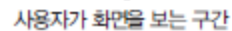
- Ajax의 가장 대표적인 예
  - 기존 웹 페이지는 새로운 데이터를 보여주려면 항상 페이지 전환 필요

## ❖ 포털 사이트에 로그인할 때

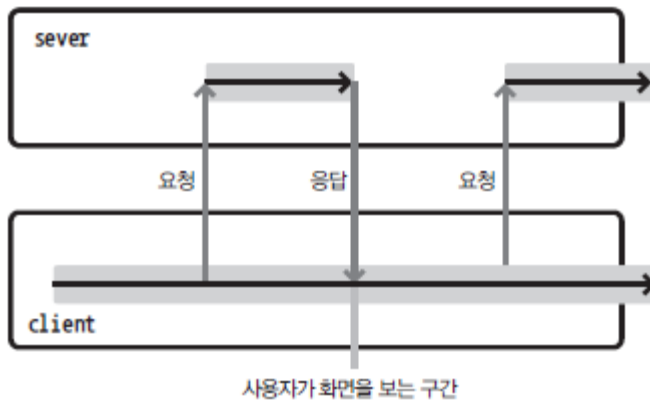
- 아이디와 비밀번호 입력
- '로그인' 버튼을 누르는 순간 데이터가 서버로 전송
- 서버에서는 아이디와 비밀번호가 일치하는지 확인
  - 일치하면 로그인 된 페이지 출력
  - 일치하지 않으면 비밀번호 재입력을 요구하는 페이지 출력
- 모바일은 페이지가 전환되는 1~2초 동안 흰 페이지 출력



사용자가 화면을 볼 수 없는 구간



사용자가 화면을 보는 구간



## ❖ Ajax 이용한 예인 Facebook

- 글 입력 시 보고 있는 현재 페이지에 글 입력
- 친구가 글을 쓰면 현재 페이지에 글 추가

그림 19-4 페이스북



## ❖ JSON이란?

- JavaScript Object Notation
- Ajax를 사용할 때는 대부분 JSON 사용
  - 클라이언트가 서버로부터 데이터 전달받음
- 이름 그대로 자바스크립트에서 쓰이는 객체 표현 방법
- 객체, 배열, 문자열, 숫자, 불리언, null만 들어갈 수 있음
- 문자열은 무조건 큰따옴표 사용

```
{
  "이름": "윤인성",
  "취미": [ "기타", "개발", "데굴데굴" ];
  "전화번호": [{
    "형태": "집전화",
    "번호": "070-0000-0000"
  }, {
    "형태": "핸드폰",
    "번호": "010-0000-0000"
  }
]
}
```

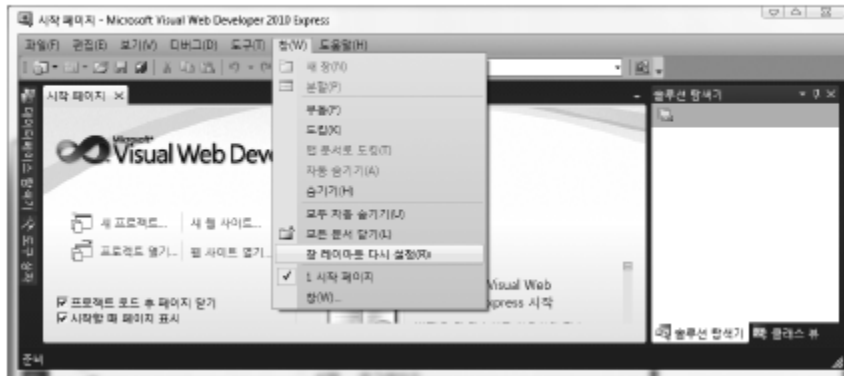


# 19.3 창 레이아웃 다시 설정

## ❖ Visual Web Developer 2010 Express의 레이아웃

- 변경되었다면 속도를 위해 초기화
- [창]→[창 레이아웃 다시 설정]
  - 처음 설치했을 때의 상태로 돌아감

그림 19-5 창 레이아웃 다시 설정



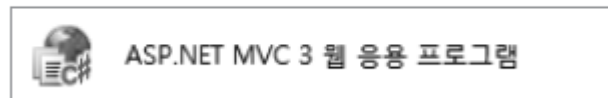
## ❖ 프로젝트 생성

- Visual Web Developer 2010 Express에서 [파일]→[새 프로젝트] 선택
- ASP.NET MVC 3 웹 응용 프로그램 선택 – 템플릿 이용

그림 19-6 새 프로젝트



그림 19-7 새 프로젝트 대화상자





## ❖ 프로젝트 생성 (2)

- 프로젝트 이름을 HelloMVC로 입력
- 뷰 엔진 선택
  - 예제에서는 주로 웹 서비스용으로 쓰게 됨
  - Razor의 형태가 간단하므로 Razor 사용
  - 오른쪽 솔루션 탐색기로 프로젝트 내의 파일 살펴볼 수 있음

그림 19-8 새 ASP.NET MVC 3 프로젝트 대화상자

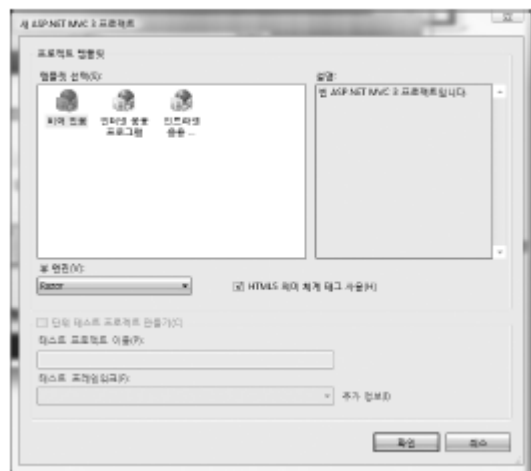
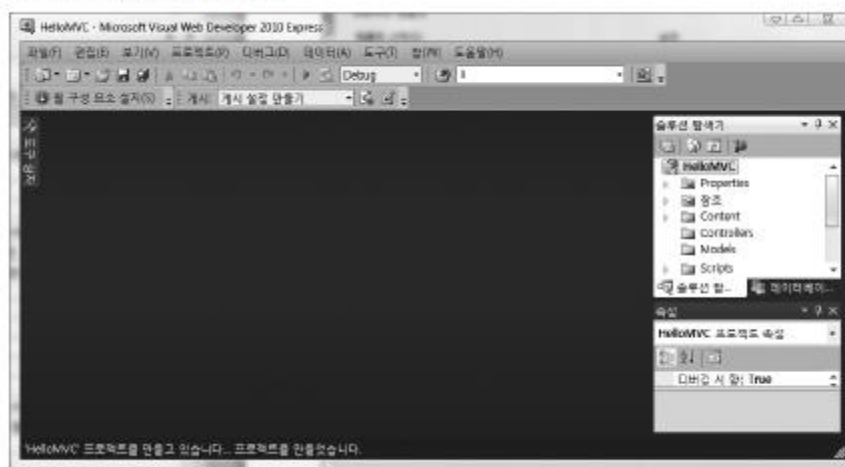


그림 19-9 생성된 프로젝트



# 19.5 컨트롤러와 뷰 추가

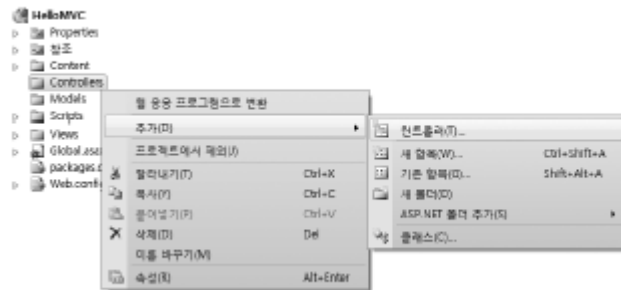
## ❖ ASP.NET MVC

- 웹 개발 방식 중 이름 그대로 MVCModel-View-Controller 패턴 따름
- MVC 패턴을 이루는 세 가지 요소 모델, 뷰, 컨트롤러 필요

## ❖ 컨트롤러 작성

- Controllers 폴더를 마우스 오른쪽 버튼으로 클릭해 컨트롤러 추가

그림 19-10 컨트롤러 추가



## ❖ 컨트롤러 작성 코드

코드 19-2 HomeController.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace HelloMVC.Controllers
{
    public class HomeController : Controller
    {
        // GET: /Home/
        public ActionResult Index()
        {
            return View();
        }
    }
}
```



## ❖ 메서드에 뷰 추가

그림 19-11 뷰 추가

```
public class HomeController : Controller
{
    // GET: /Home/
    public ActionResult Index()
    {
        return View();
    }
}
```

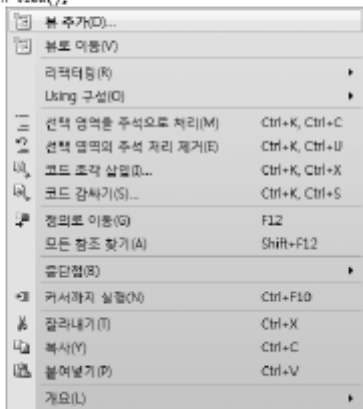
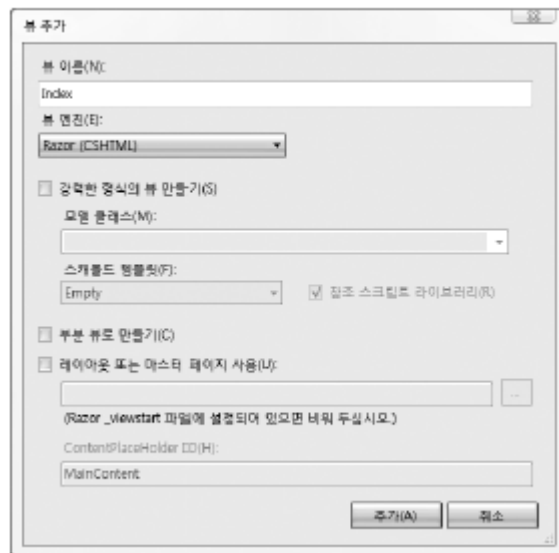


그림 19-12 뷰 추가 대화상자



## ❖ Index.cshtml 생성

코드 19-3 Index.cshtml

```
@{ Layout = null; }  
<!DOCTYPE html>  
<html>  
<head>  
    <title>Index</title>  
</head>  
<body>  
    <div>  
  
    </div>  
</body>  
</html>
```

그림 19-13 추가된 뷰



# 19.6 프로젝트 실행과 종료

## ❖ 프로젝트 실행과 종료

- 그림 19-14의 실행 버튼 클릭하거나 F5 누르면 프로젝트 실행
- 웹 브라우저가 자동으로 `http://localhost:Port/`에 접속
  - 포트는 프로젝트를 실행할 때마다 변경

그림 19-14 프로젝트 실행

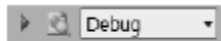


그림 19-15 실행된 프로젝트

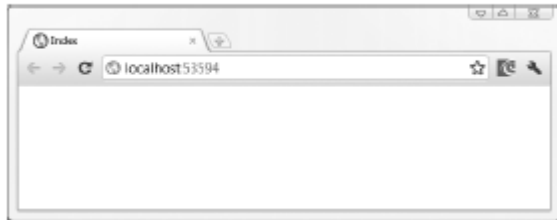


그림 19-16 프로젝트 종료



## ❖ URL 접속

- 프로젝트를 실행시 `http://localhost:Port/`로 접속
  - 현재 ASP.NET MVC의 기본 설정이 자동으로 <http://localhost:Port/Home/Index> 에 접속하게 설정
- 액션
  - HomeController 클래스에 Index() 메서드 있음
    - **Home 아래에 Index 페이지 존재**
  - Index() 메서드처럼 URL을 생성하는 메서드.

코드 19-4 HomeController.cs

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

그림 19-17 Views 폴더



## ❖ 액션 추가

- MyFirstAction에 접속하려면 ?
  - <http://localhost:PortNumber/Home/MyFirstAction> - 뷰가 없어 에러 발생

코드 19-5 HomeController.cs : 액션 MyFirstAction 추가

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

    public ActionResult MyFirstAction()
    {
        return View();
    }
}
```

그림 19-18 MyFirstAction 접속 결과

'/' 응용 프로그램에 서버 오류가 있습니다.

'MyFirstAction' 뷰 또는 레이아웃 마스터가 없거나 뷰 엔진이 검색한 위치를 지원하지 않습니다. 검색한 위치:  
~/Views/Home/MyFirstAction.aspx  
~/Views/Home/MyFirstAction.ascx  
~/Views/Shared/MyFirstAction.aspx  
~/Views/Shared/MyFirstAction.ascx  
~/Views/Home/MyFirstAction.cshtml  
~/Views/Home/MyFirstAction.vbhtml  
~/Views/Shared/MyFirstAction.cshtml  
~/Views/Shared/MyFirstAction.vbhtml





## ❖ C#의 “클래스”

- 기본적으로 이름 공간 namespace 안에 존재
- 예제 코드 19-6]
  - HomeController 클래스는 이름 공간 HelloMVC.Controllers 안에 존재

코드 19-6 HomeController.cs

```
namespace HelloMVC.Controllers
{
    public class HomeController : Controller
    {
        // GET: /Home/
        public ActionResult Index()
        {
            return View();
        }
    }
}
```



## ❖ 클래스 내부의 “메서드”

### ■ C# 메서드의 형태

```
접근제한자 리턴값의자료형 메서드이름(매개 변수)
{
    return 리턴값;
}
```

- 리턴값의 자료형 지정
- 자바스크립트와 접근 제한자 밖에 차이가 없음

코드 19-7 HomeController.cs: 메서드

```
public ActionResult Index()
{
    return View();
}
```



## ❖ 변수 선언과 초기화

- C#도 var 키워드를 사용해 변수 생성
  - C#은 문자열을 만들 때 반드시 큰따옴표 사용해야 !
- new 키워드를 사용해 객체(인스턴스) 생성

코드 19-8 HomeController.cs: 변수 선언과 초기화

```
public ActionResult Index()
{
    var numberVariable = 273;
    var stringVariable = "String";
    var dateTimeVariable = new DateTime();

    return View();
}
```



## ❖ 웹 서비스

- 클라이언트에서 요청하는 정보 제공 서비스
- Ajax 사용 위한 조건
- 예제코드
  - 간단한 HTML 태그 제공

코드 19-9 HomeController.cs: 액션 MyFirstStringAction 생성

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

    public ActionResult MyFirstStringAction()
    {
        return Content("<h1>Hello ASP.NET MVC</h1>");
    }
}
```

그림 19-19 액션 MyFirstStringAction

**Hello ASP.NET MVC**



## ❖ 복잡한 JSON 파일을 제공하는 웹 서비스

- 컨트롤러 HomeController에 액션 MyFirstJsonAction 추가

코드 19-10 HomeController.cs: 액션 MyFirstJsonAction 생성

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

    public ActionResult MyFirstStringAction()
    {
        return Content("<h1>Hello ASP.NET MVC</h1>");
    }

    public ActionResult MyFirstJsonAction()
    {
        return Json();
    }
}
```



## ❖ 복잡한 JSON 파일을 제공하는 웹 서비스 (2)

- Json() 메서드는 일반적인 C# 객체를 손쉽게 JSON 형태로 변환
- Json() 메서드의 두 번째 매개 변수
  - GET 방식으로도 액션 MyFirstJsonAction에 접근할 수 있게
  - Json을 리턴하는 액션은 보안상 POST 방식으로만 접근 가능
    - 예제에서는 직접 웹 페이지상에서 확인 하므로 GET 방식
  - 프로젝트를 실행
    - **http://localhost:Port/Home/MyFirstJsonAction에 접속**

코드 19-11 HomeController.cs: Json() 메서드의 사용

```
public ActionResult MyFirstJsonAction()
{
    var willReturn = new { name="윤인성", gender="남자", part="세컨드기타" };
    return Json(willReturn, JsonRequestBehavior.AllowGet);
}
```

```
{"name":"윤인성","gender":"남자","part":"세컨드기타"}
```



## ❖ XML 웹 서비스 만들기

코드 19-12 HomeController.cs: 액션 MyFirstXMLAction 생성

```
public ActionResult MyFirstXMLAction()
{
    var willReturn = "";
    return Content(willReturn, "text/xml");
}
```



## ❖ XML 웹 서비스 만들기 (2)

- Content() 메서드 사용
  - Content() 메서드의 두 번째 매개 변수
    - 제공하는 파일의 형태를 문자열로 입력
    - Ex) 액션 MyFirstXMLAction은 XML 제공
      - » ‘text/xml’ 입력

코드 19-13 HomeController.cs: XML 문자열 생성

```
public ActionResult MyFirstXMLAction()
{
    var willReturn = "";
    willReturn += "<people>";
    willReturn += "    <person>";
    willReturn += "        <name>윤인성</name>";
    willReturn += "        <gender>남자</gender>";
    willReturn += "        <part>기타</part>";
    willReturn += "    </person>";
    willReturn += "    <person>";
    willReturn += "        <name>연하진</name>";
    willReturn += "        <gender>여자</gender>";
    willReturn += "        <part>기타</part>";
    willReturn += "    </person>";
    willReturn += "</people>";

    return Content(willReturn, "text/xml");
}
```





## ❖ XML 웹 서비스 만들기 (3)

### ■ 프로젝트 실행

- <http://localhost:Port/Home/MyFirstXMLAction> 인터넷 익스플로러 실행

그림 19-20 인터넷 익스플로러의 실행 결과

```
<?xml version="1.0"?>
- <people>
  - <person>
    <name>윤인성</name>
    <gender>남자</gender>
    <part>기타</part>
  </person>
  - <person>
    <name>연하진</name>
    <gender>여자</gender>
    <part>기타</part>
  </person>
</people>
```



## ❖ 요청 매개 변수 처리

- 지금까지 사용했던 URL 입력해 처리
- 예제
  - 액션 ActionWithData 생성
  - 요청 매개 변수를 사용해 간단한 HTML 태그를 만들어 제공

코드 19-14 HomeController.cs: 액션 ActionWithData 생성

```
public ActionResult ActionWithData()
{
    return Content("");
}
```

코드 19-15 HomeController.cs: 요청 매개 변수 처리 (1)

```
public ActionResult ActionWithData()
{
    var name = Request["name"];
    var age = Request["age"];

    return Content("<h1>" + name + ":" + age + "</h1>");
}
```



## ❖ 요청 매개 변수 처리

### ■ 실행결과

- <http://localhost:Port/Home/ActionWithData?name=RintianTta&age=21>  
접속

그림 19-21 실행 결과

RintianTta:21

- 전달하는 매개 변수와 이름이 같은 변수를 매개 변수로 입력
  - C#은 메서드의 매개 변수 위치에 반드시 자료형 지정
  - 예제 코드 19-16]
    - » 매개 변수 name과 age를 string 자료형으로 지정

코드 19-16 HomeController.cs: 요청 매개 변수 처리 (2)

```
public ActionResult ActionWithData(string name, string age)
{
    return Content("<h1>" + name + ":" + age + "</h1>");
}
```





# Thank You !

모던 웹을 위한 Javascript jQuery 입문