■ 통합개발환경(IDE)의 설치 및 설정

1. 개발툴의 종류

1) 메모장

- 장점 : 가장 가벼움, 새로운 프로그램(Application)을 설치할 필요가 없음
- 단점 : 오타가 나기 쉽고, 모든 코드를 외워서 코딩해야 함

2. 텍스트 에디터 : EditPlus, UltraEdit 등

- 장점 : 꽤 가벼움, 유용한 기능들이 존재함
- 단점 : 개발시 불편한 점이 있음(코드 자동 완성 기능 등이 없음)

3. 웹에디터(WYSIWYG 가능 툴) : DreamWeaver, 나모웹에디터 등

- 장점 : 마우스만으로 코딩이 가능함, 쉽고 편리한 코딩이 가능함
- 단점 : 유료, 불필요한 코드가 생길 가능성이 높음
  편리해 실제 코딩에는 익숙하지 않게 됨

4. IDE(통합환경 개발툴) : Eclipse, Brackets, Atom, Sublime text 등

- 장점 : 자동완성 기능과 디버깅이 가능함(일부), 그 외 기능들
- 단점 : 복잡해 보임
  프로그램 자체가 무거움

WYSIWYG란?
What You See Is What You Get "보는 대로 얻는다" 문서 편집 과정에서 화면에 포맷된 낱말, 문장이 출력물과 동일하게 나오는 방식을 말한다.
IDE란?
코딩, 디버그, 컴파일, 배포 등 프로그램 개발에 관련된 모든 작업을 하나의 프로그램 안에서 처리하는 환경을 제공하는 소프트웨어이다.

■ WebStorm의 설치 및 설정

1) WebStorm 다운로드

그림) https://www.jetbrains.com/webstorm/

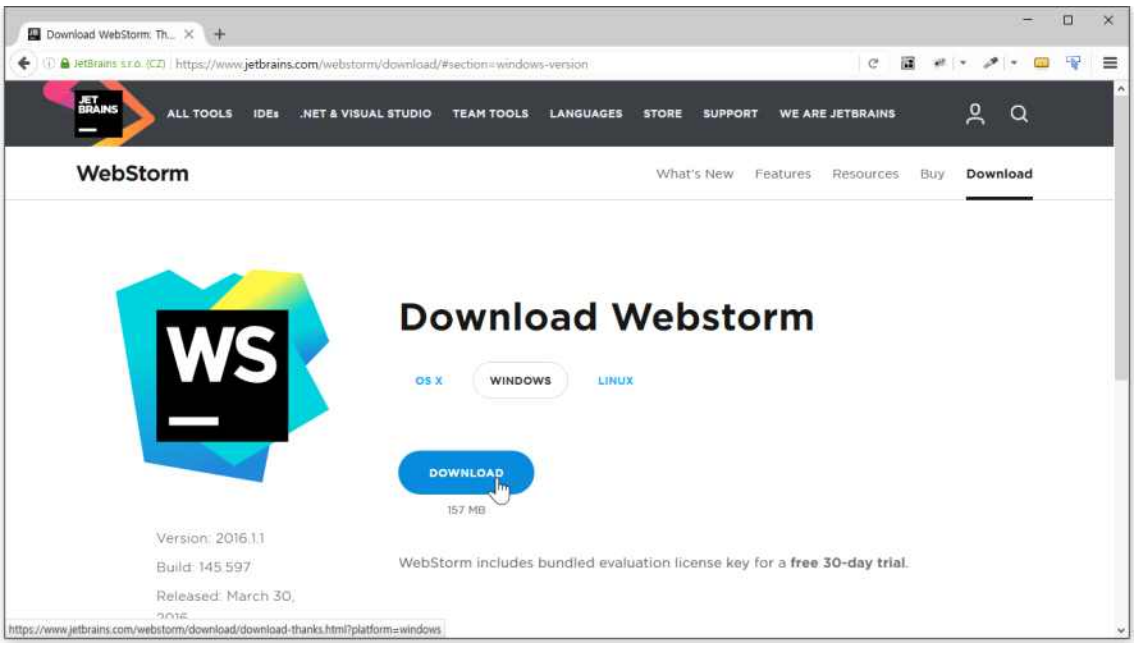
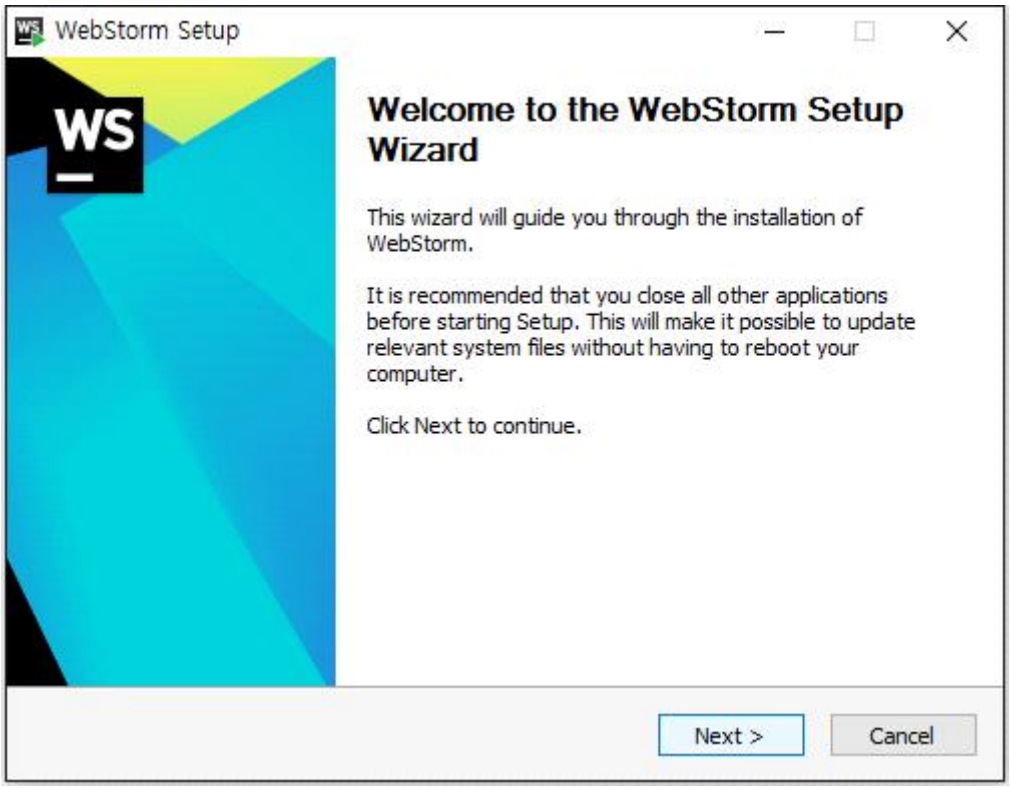
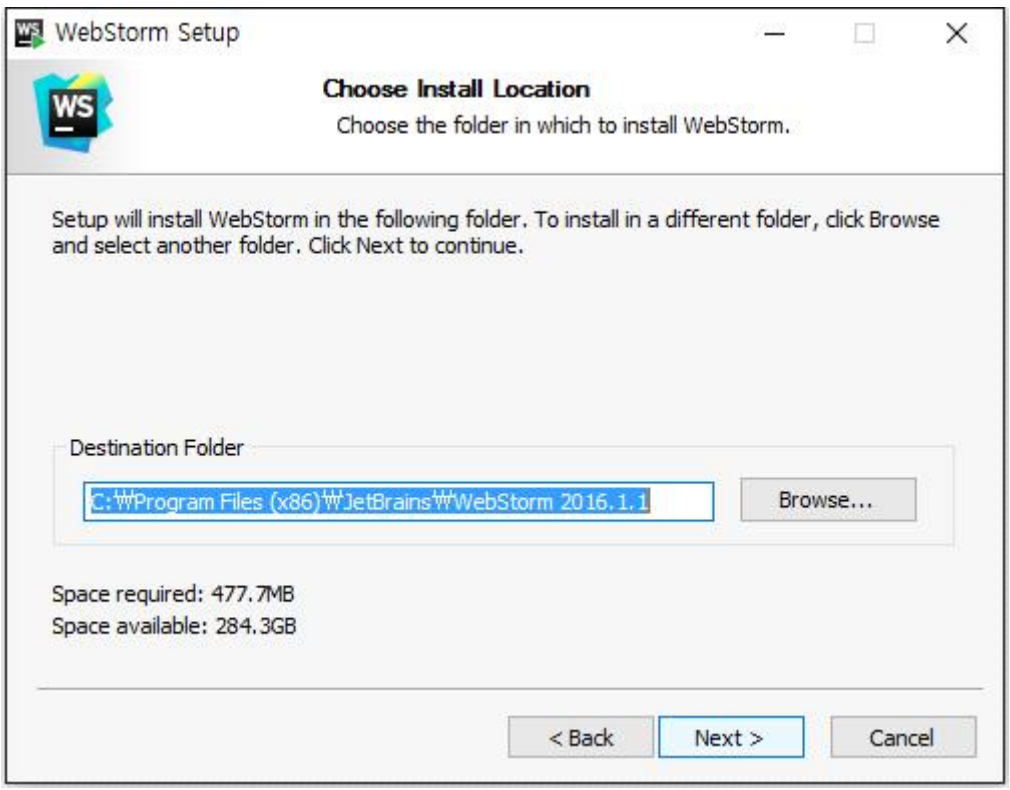
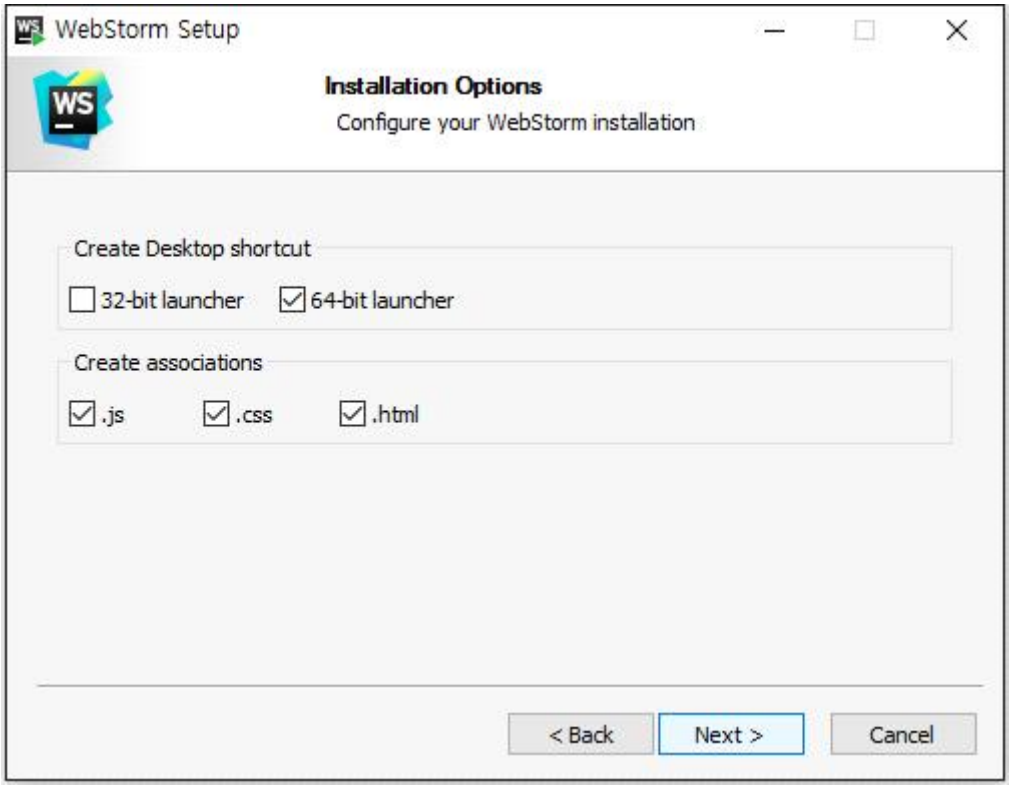


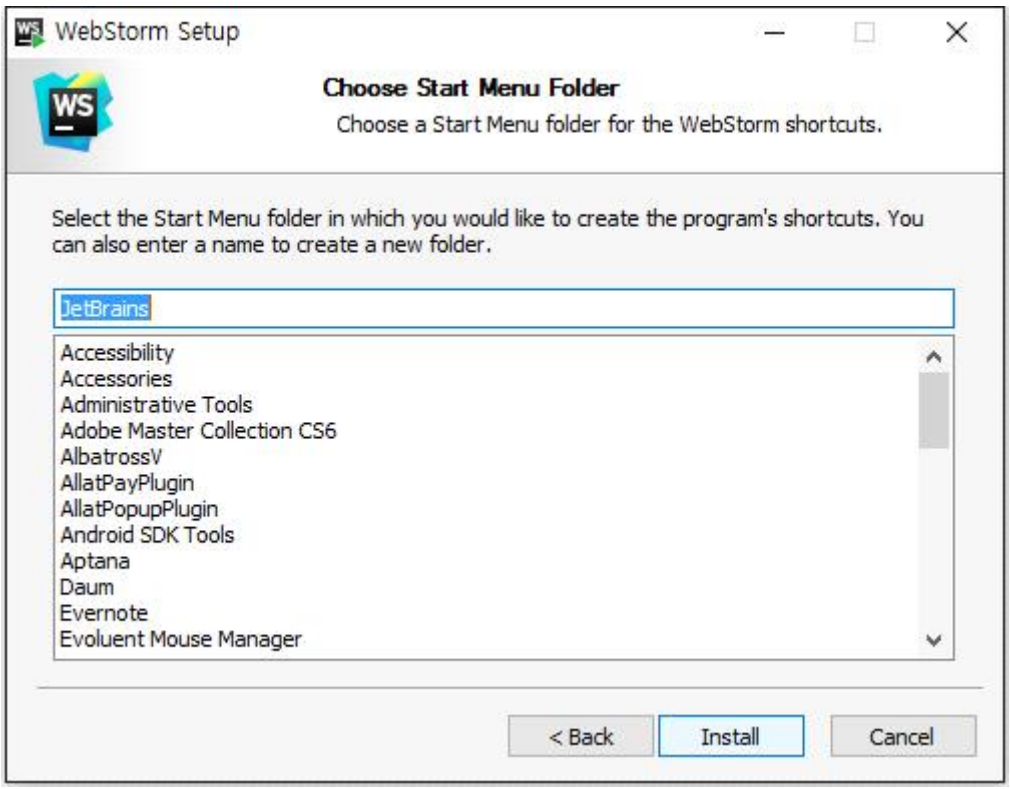
그림) https://www.jetbrains.com/webstorm/download/download-thanks.html?platform=windows

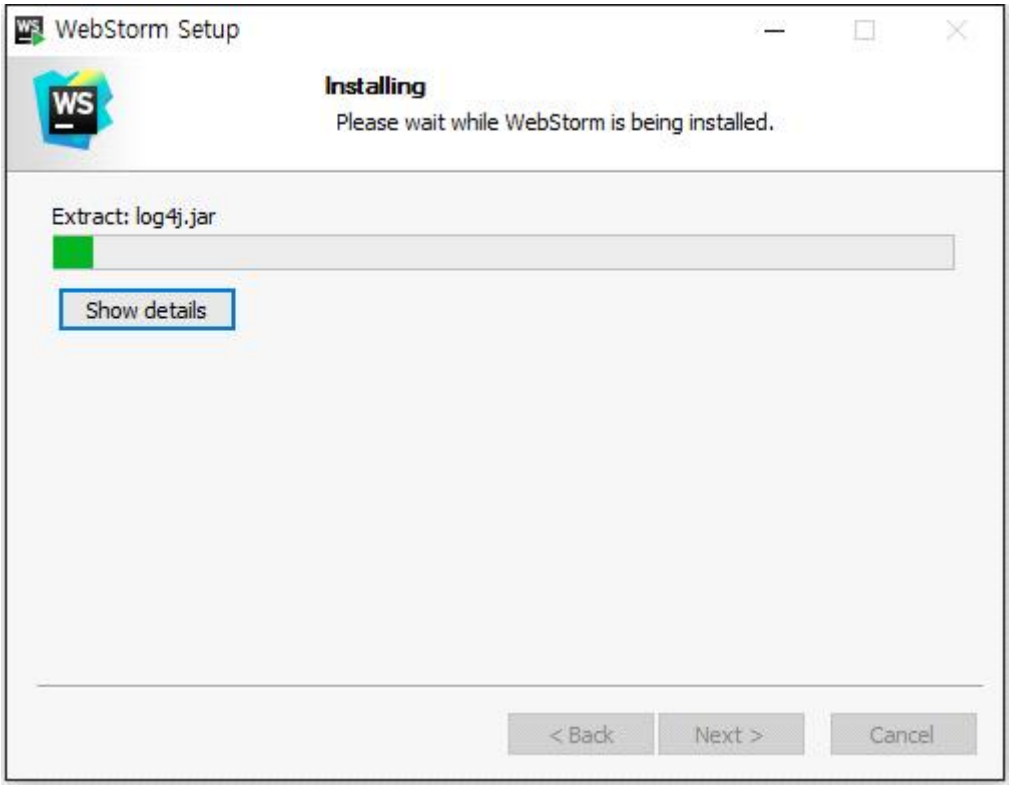
3) 설치

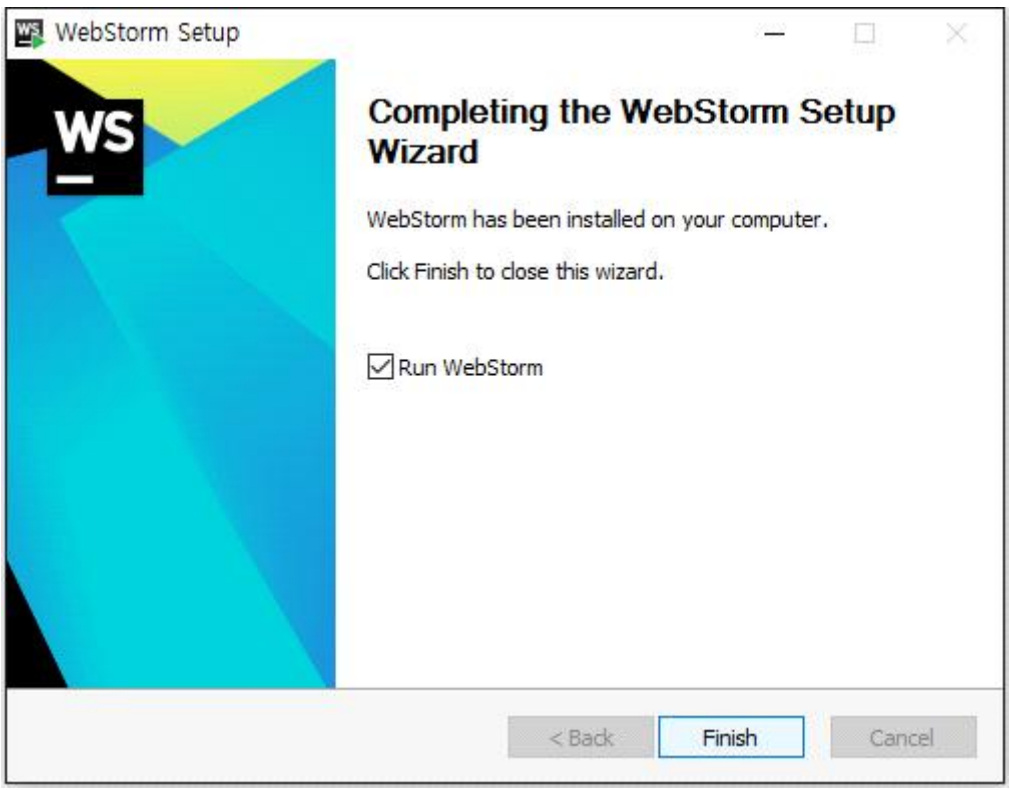

WebStorm Setup
Welcome to the WebStorm Setup Wizard
This wizard will guide you through the installation of WebStorm.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Next >
Cancel


WebStorm Setup
Choose Install Location
Choose the folder in which to install WebStorm.
Setup will install WebStorm in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
C:₩Program Files (x86)₩JetBrains₩WebStorm 2016.1.1
Browse...
Space required: 477.7MB
Space available: 284.3GB
< Back
Next >
Cancel

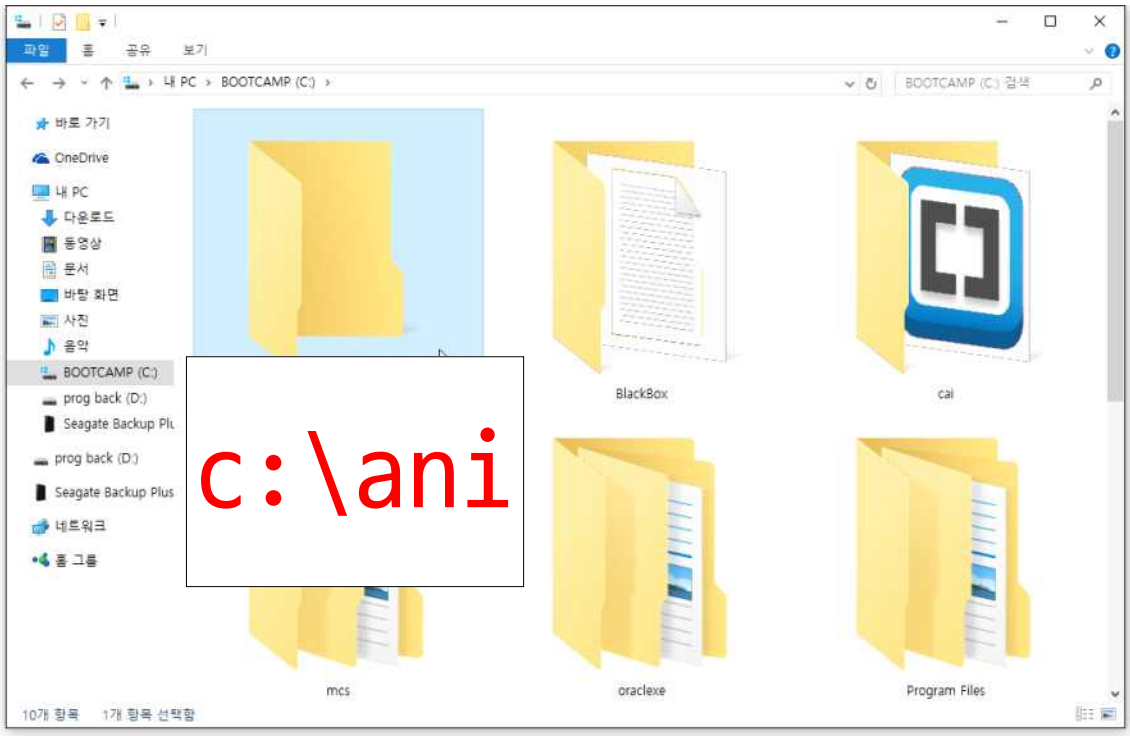

WebStorm Setup
Installation Options
Configure your WebStorm installation
Create Desktop shortcut
32-bit launcher
64-bit launcher
Create associations
.js
.css
.html
< Back
Next >
Cancel


WebStorm Setup
Choose Start Menu Folder
Choose a Start Menu folder for the WebStorm shortcuts.
Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.
JetBrains
Accessibility
Accessories
Administrative Tools
Adobe Master Collection CS6
AlbatrossV
AllatPayPlugin
AllatPopupPlugin
Android SDK Tools
Aptana
Daum
Evernote
Evoluent Mouse Manager
< Back
Install
Cancel

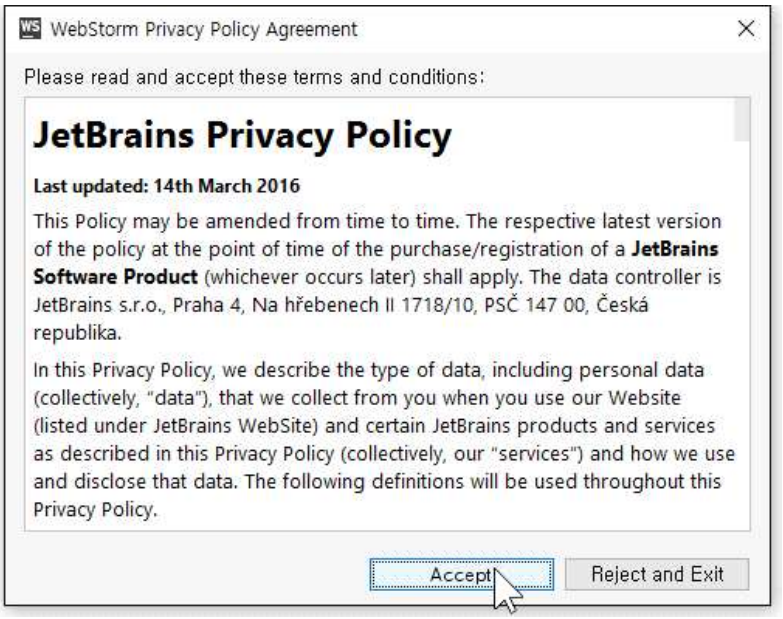

WebStorm Setup
Installing
Please wait while WebStorm is being installed.
Extract: log4j.jar
Show details
< Back
Next >
Cancel

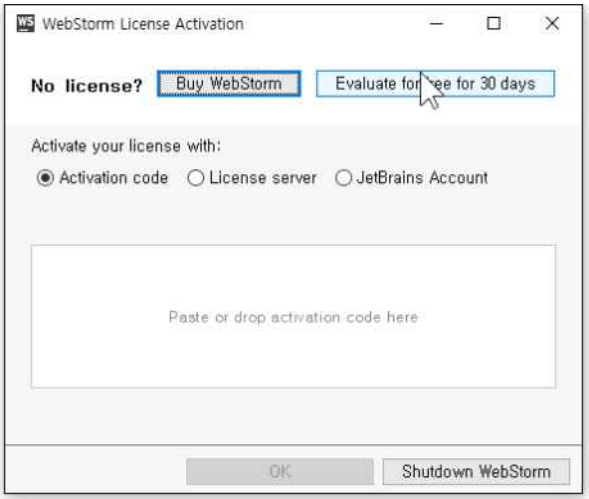

WebStorm Setup
Completing the WebStorm Setup Wizard
WebStorm has been installed on your computer.
Click Finish to close this wizard.
Run WebStorm
< Back
Finish
Cancel

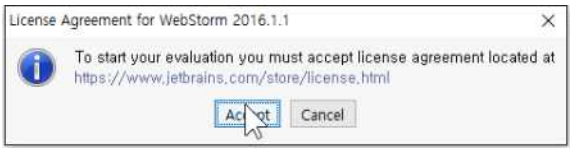
4) 수업소스폴더 생성

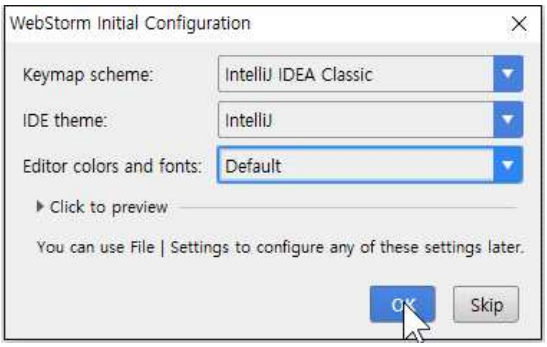


5) 실행

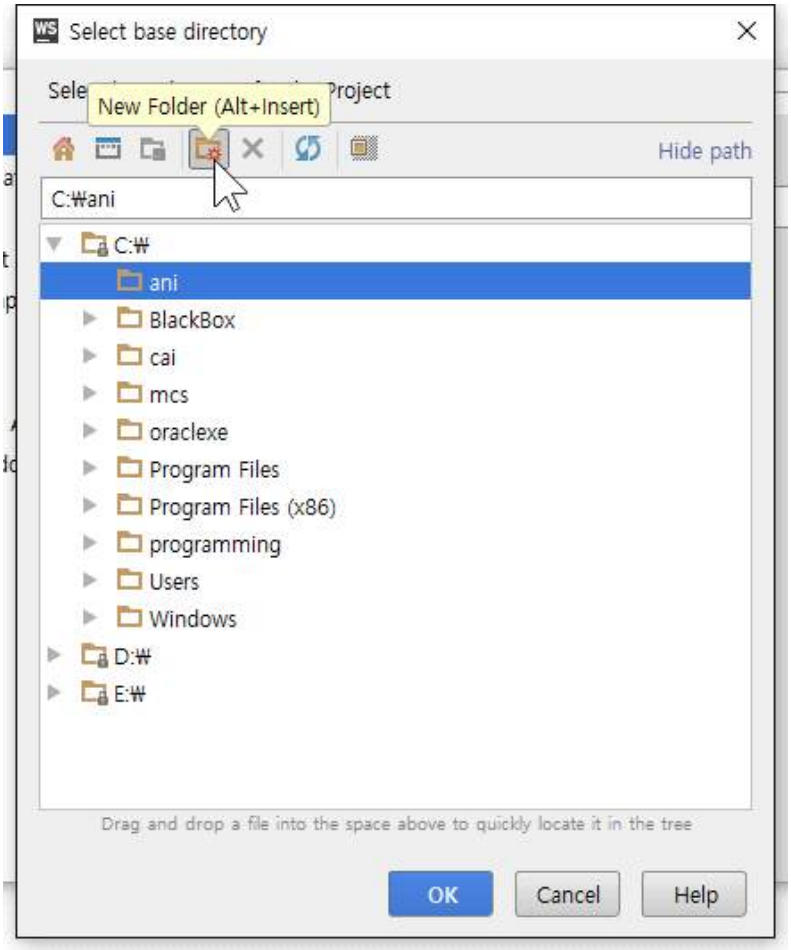



WebStorm License Activation
No license?
Buy WebStorm
Evaluate for free for 30 days
Activate your license with:
Activation code
License server
JetBrains Account
Paste or drop activation code here
OK
Shutdown WebStorm

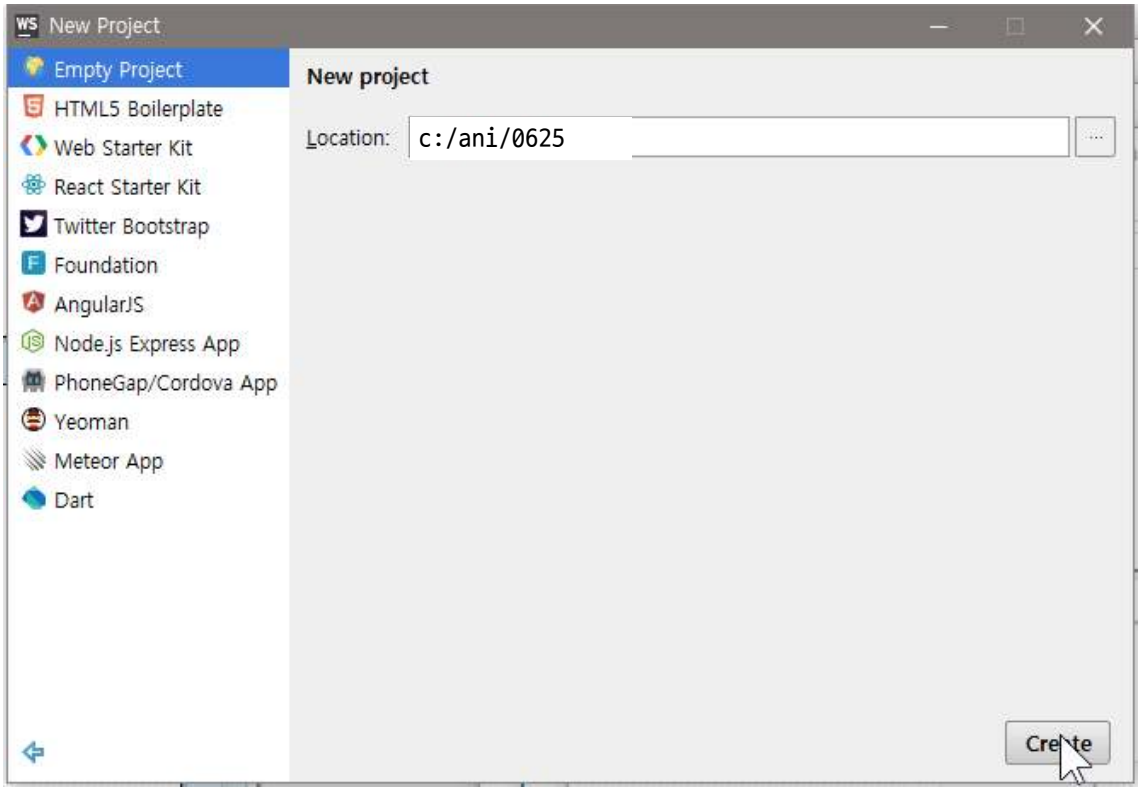

License Agreement for WebStorm 2016.1.1
To start your evaluation you must accept license agreement located at
https://www.jetbrains.com/store/license.html
Accept
Cancel

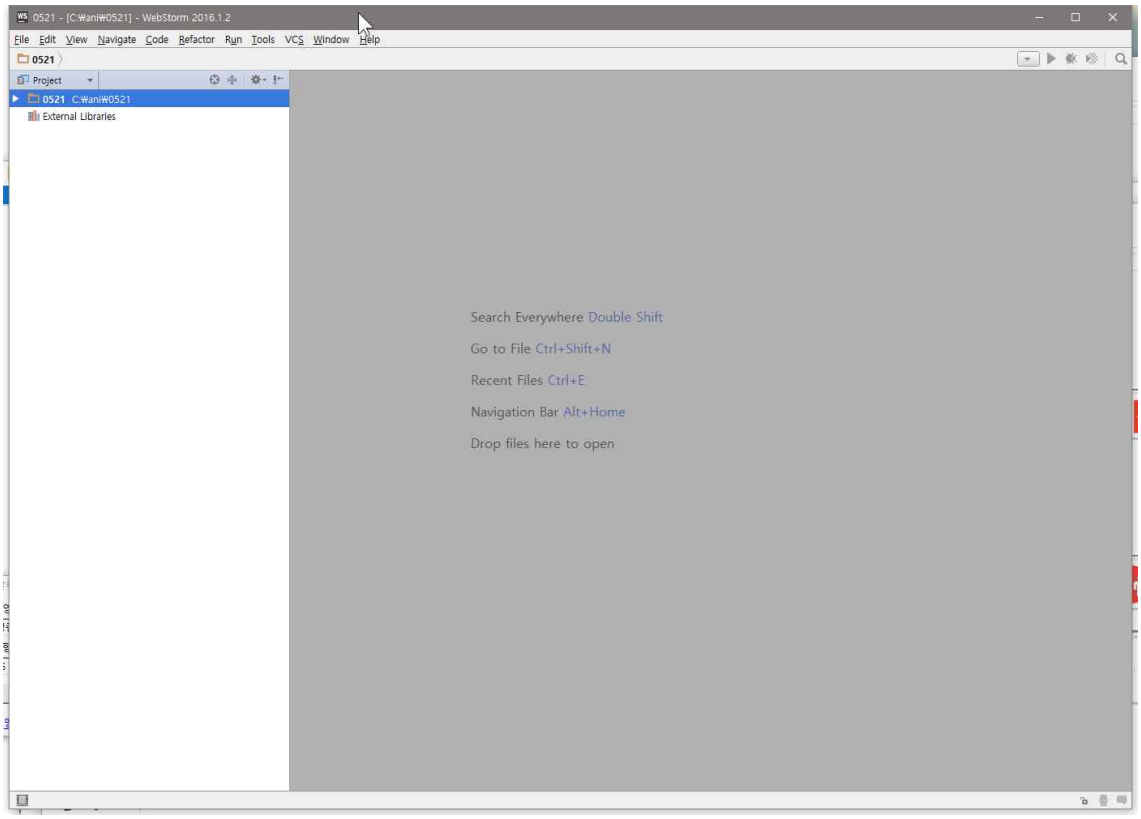

WebStorm Initial Configuration
Keymap scheme:
IntelliJ IDEA Classic
IDE theme:
IntelliJ
Editor colors and fonts:
Default
Click to preview
You can use File | Settings to configure any of these settings later.
OK
Skip


Welcome to WebStorm
WebStorm
Version 2016.1.1
Create New Project
Open
Check out from Version Control
Register
Configure
Get Help


Select base directory
New Folder (Alt+Insert)
Hide path
C:₩ani
C:₩
ani
BlackBox
cai
mcs
oraclexe
Program Files
Program Files (x86)
programming
Users
Windows
D:₩
E:₩
Drag and drop a file into the space above to quickly locate it in the tree
OK
Cancel
Help

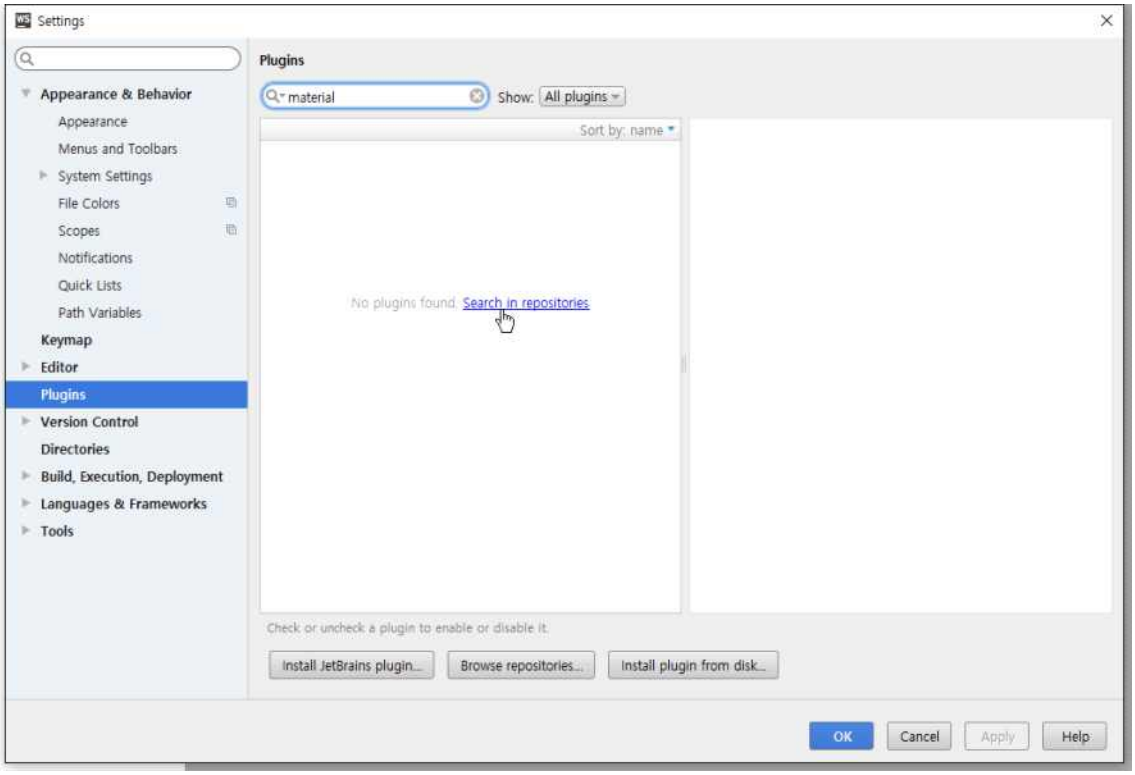

New Folder
Enter a new folder name:
0625
OK
Cancel

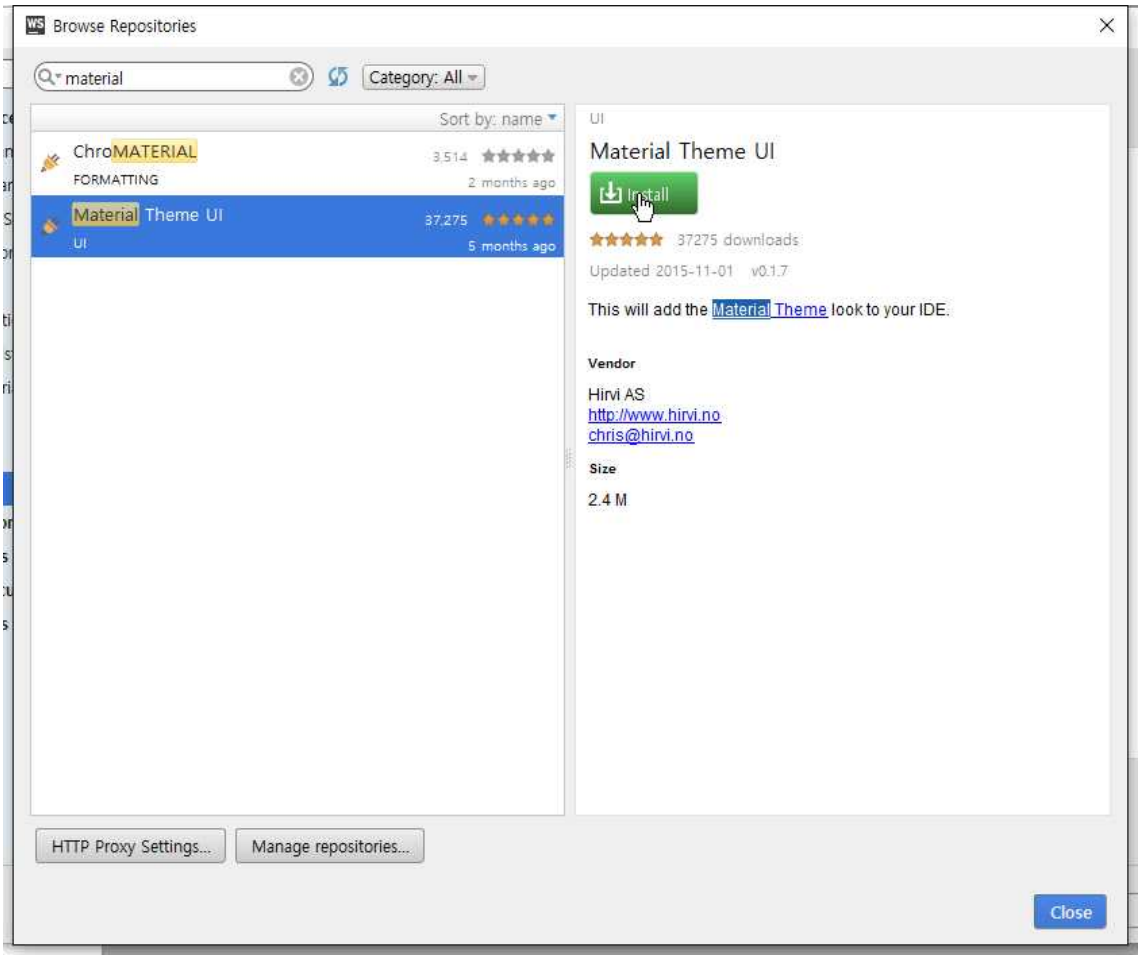

New Project
Empty Project
HTML5 Boilerplate
Web Starter Kit
React Starter Kit
Twitter Bootstrap
Foundation
AngularJS
Node.js Express App
PhoneGap/Cordova App
Yeoman
Meteor App
Dart
New project
Location:
c:/ani/0625
Create

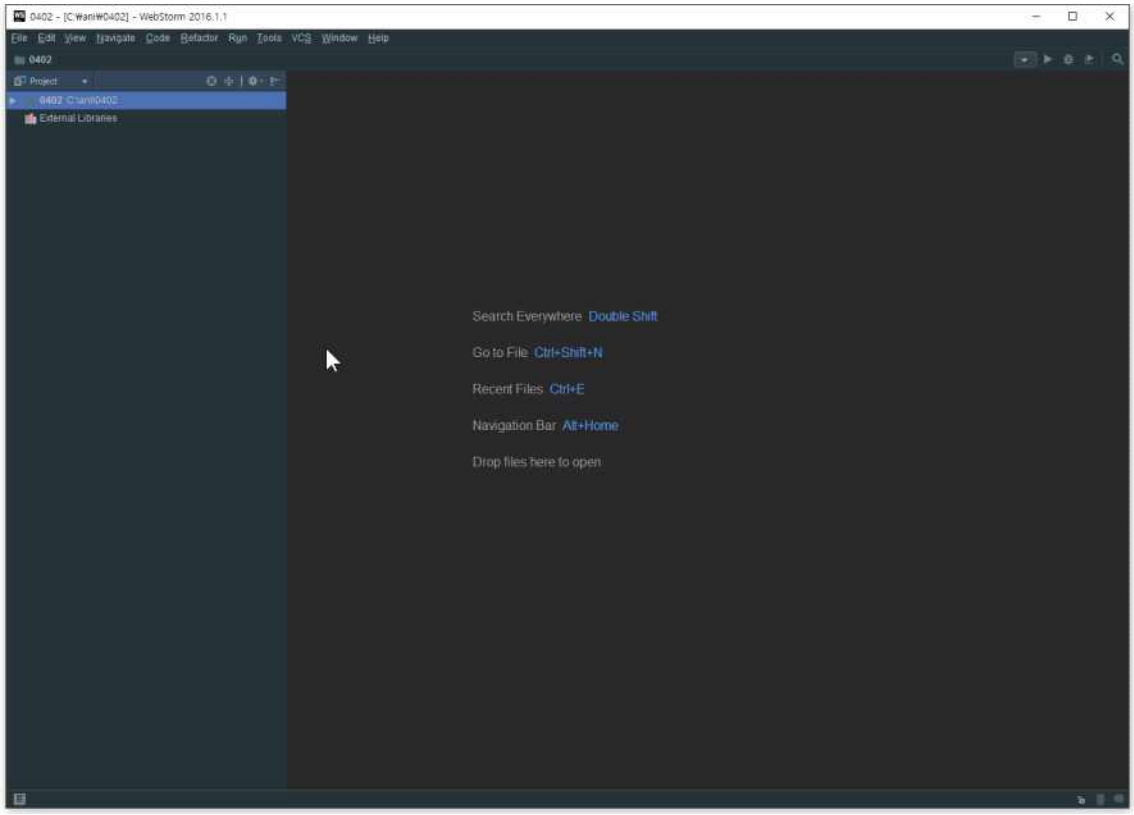

File Edit View Navigate Code Refactor Run Tools VCS Window Help
0521
Project
0521 C:\ani\0521
External Libraries
Search Everywhere Double Shift
Go to File Ctrl+Shift+N
Recent Files Ctrl+E
Navigation Bar Alt+Home
Drop files here to open

5) 설정

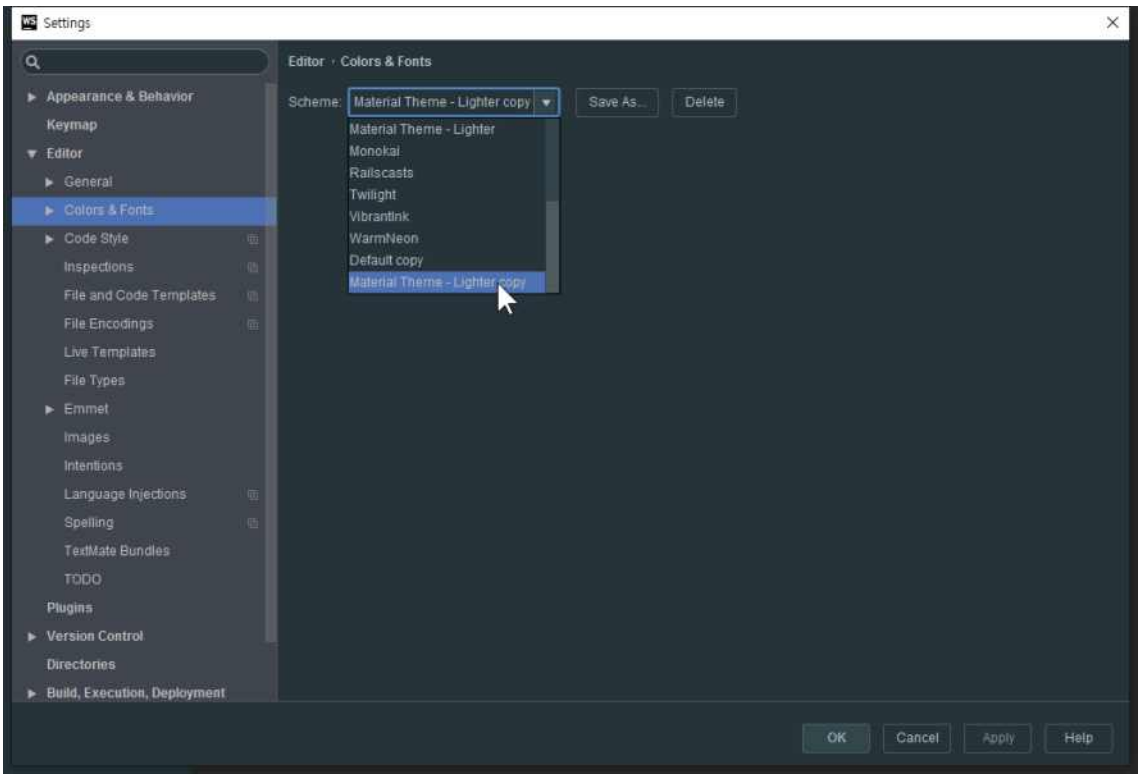
- 테마 변경


Settings
Plugins
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
File Colors
Scopes
Notifications
Quick Lists
Path Variables
Keymap
Editor
Plugins
Version Control
Directories
Build, Execution, Deployment
Languages & Frameworks
Tools
material
Show: All plugins
Sort by: name
No plugins found. Search in repositories
Check or uncheck a plugin to enable or disable it.
Install JetBrains plugin...
Browse repositories...
Install plugin from disk...
OK
Cancel
Apply
Help


Browse Repositories
material
Category: All
Sort by: name
ChroMATERIAL
FORMATTING
3,514
2 months ago
Material Theme UI
UI
37,275
5 months ago
UI
Material Theme UI
Install
37275 downloads
Updated 2015-11-01 v0.1.7
This will add the Material Theme look to your IDE.
Vendor
Hirvi AS
http://www.hirvi.no
chris@hirvi.no
Size
2.4 M
HTTP Proxy Settings...
Manage repositories...
Close



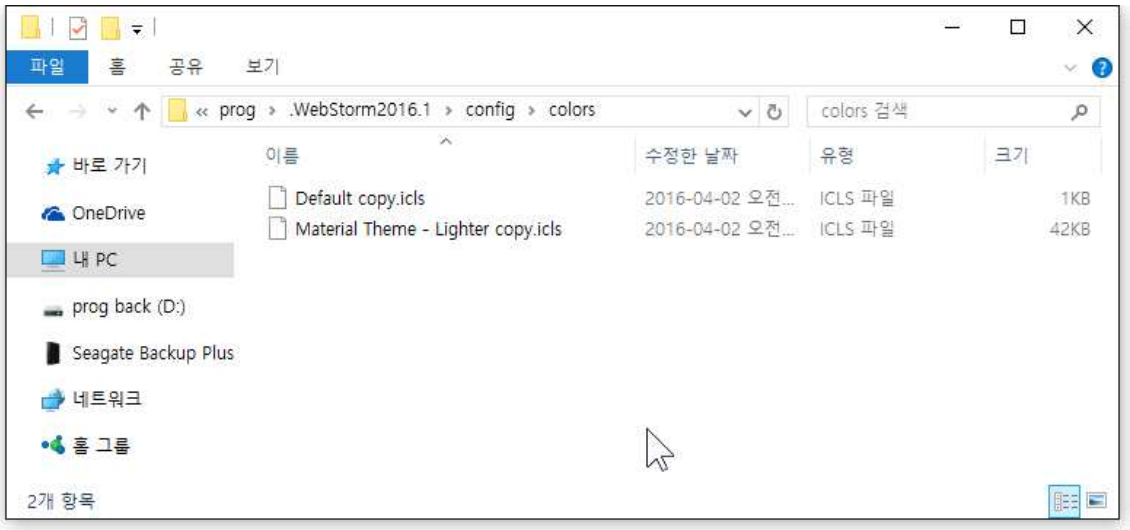
- 코드 스타일 변경



- 제가 Material Theme - Lighter를 수정한 스타일을 적용하려면

c:\사용자\현재사용자\.WebStorm2016.1\config\color폴더 생성후

Material Theme - Lighter copy.icls 파일을 넣음

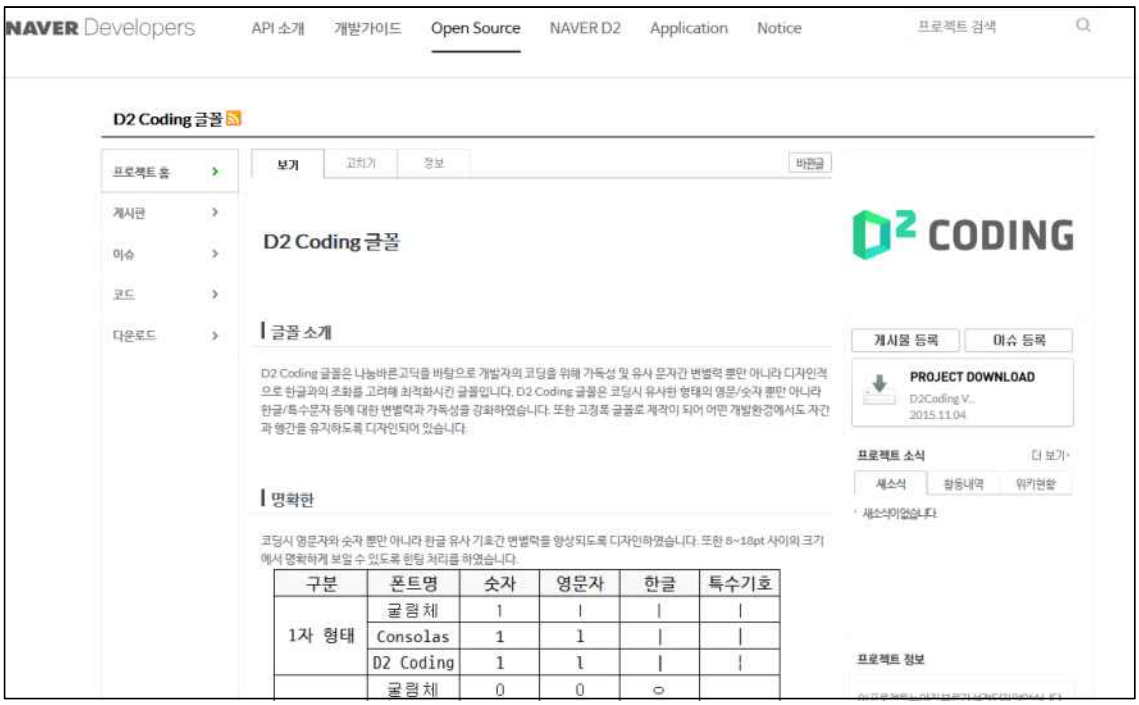


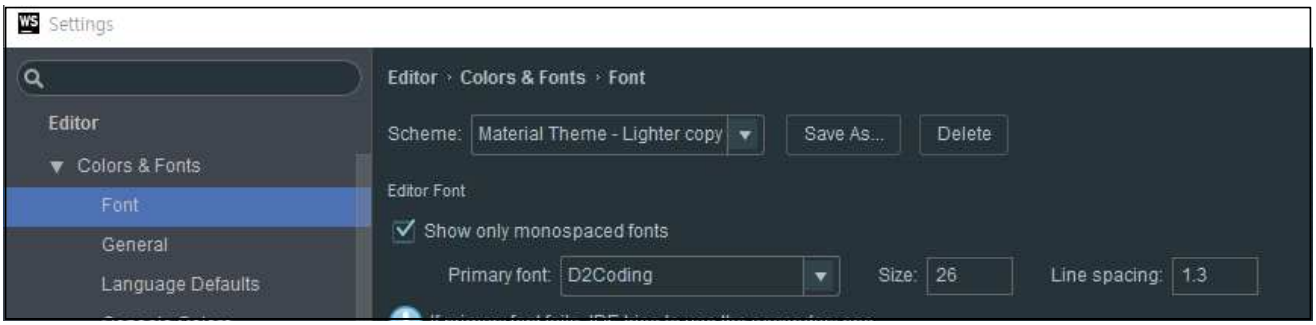
- 코딩폰트 변경

구분	폰트명	숫자	영문자	한글	특수기호
1자 형태	굴림체	1	l	|	|
	Consolas	1	l	|	|
	D2 Coding	1	l	|	¦
0자 형태	굴림체	0	0	ㅇ	
	Consolas	0	0	o	
	D2 Coding	0	0	o	
코드 기호	굴림체	( ){ }[ ]"'`":;,.,.--++==__/			
	Consolas	( ){ }[ ]"'`":;,.,.--++==__/			
	D2 Coding	( ){ }[ ]"'`":;,.,.--++==__/			

http://dev.naver.com/projects/d2coding

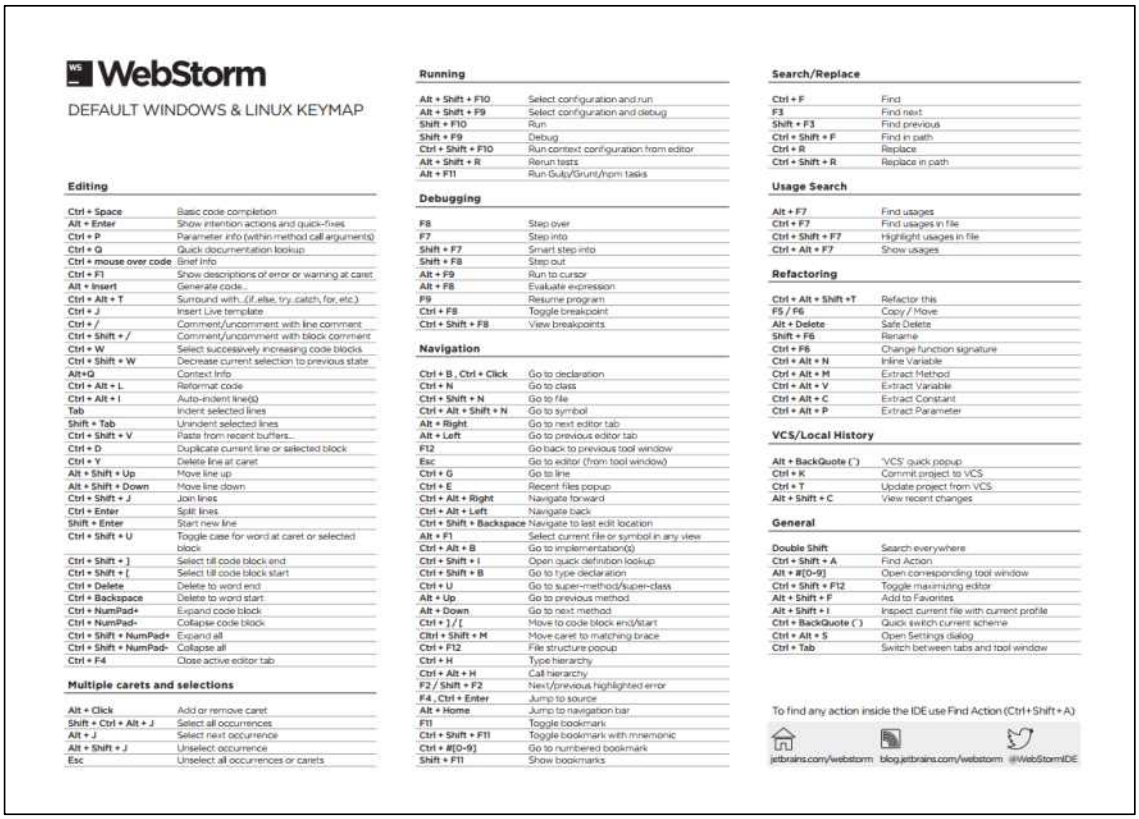
- D2 Coding폰트





6) 주요 단축키

기능	단축키
코드 자동완성	ctrl + space
프로젝트 뷰 토글	alt + 1
프로젝트 뷰 숨기기	shift + Esc
브라우저 실행	ctrl + shift + F10
한 줄 복사	ctrl + D
한 줄 삭제	ctrl + Y
줄 옮기기	alt + shift + 방향키
코딩 포맷팅	ctrl + alt + L



WebStorm

DEFAULT WINDOWS & LINUX KEYMAP

Editing

Ctrl + Space	Basic code completion
Alt + Enter	Show intention actions and quick-fixes
Ctrl + P	Parameter info (within method call arguments)
Ctrl + Q	Quick documentation lookup
Ctrl + mouse over code	Brief info
Ctrl + F1	Show descriptions of error or warning at caret
Alt + Insert	Generate code...
Ctrl + Alt + T	Surround with...(if..else, try..catch, for, etc.)
Ctrl + J	Insert Live template
Ctrl + /	Comment/uncomment with line comment
Ctrl + Shift + /	Comment/uncomment with block comment
Ctrl + W	Select successively increasing code blocks
Ctrl + Shift + W	Decrease current selection to previous state
Alt+Q	Context Info
Ctrl + Alt + L	Reformat code
Ctrl + Alt + I	Auto-indent line(s)
Tab	Indent selected lines
Shift + Tab	Unindent selected lines
Ctrl + Shift + V	Paste from recent buffers...
Ctrl + D	Duplicate current line or selected block
Ctrl + Y	Delete line at caret
Alt + Shift + Up	Move line up
Alt + Shift + Down	Move line down
Ctrl + Shift + J	Join lines
Ctrl + Enter	Split lines
Shift + Enter	Start new line
Ctrl + Shift + U	Toggle case for word at caret or selected block
Ctrl + Shift +]	Select till code block end
Ctrl + Shift + [	Select till code block start
Ctrl + Delete	Delete to word end
Ctrl + Backspace	Delete to word start
Ctrl + NumPad+	Expand code block
Ctrl + NumPad-	Collapse code block
Ctrl + Shift + NumPad+	Expand all
Ctrl + Shift + NumPad-	Collapse all
Ctrl + F4	Close active editor tab

Multiple carets and selections

Alt + Click	Add or remove caret
Shift + Ctrl + Alt + J	Select all occurrences
Alt + J	Select next occurrence
Alt + Shift + J	Unselect occurrence
Esc	Unselect all occurrences or carets

Running

Alt + Shift + F10	Select configuration and run
Alt + Shift + F9	Select configuration and debug
Shift + F10	Run
Shift + F9	Debug
Ctrl + Shift + F10	Run context configuration from editor
Alt + Shift + R	Rerun tests
Alt + F11	Run Gulp/Grunt/npm tasks

Debugging

F8	Step over
F7	Step into
Shift + F7	Smart step into
Shift + F8	Step out
Alt + F9	Run to cursor
Alt + F8	Evaluate expression
F9	Resume program
Ctrl + F8	Toggle breakpoint
Ctrl + Shift + F8	View breakpoints

Navigation

Ctrl + B, Ctrl + Click	Go to declaration
Ctrl + N	Go to class
Ctrl + Shift + N	Go to file
Ctrl + Alt + Shift + N	Go to symbol
Alt + Right	Go to next editor tab
Alt + Left	Go to previous editor tab
F12	Go back to previous tool window
Esc	Go to editor (from tool window)
Ctrl + G	Go to line
Ctrl + E	Recent files popup
Ctrl + Alt + Right	Navigate forward
Ctrl + Alt + Left	Navigate back
Ctrl + Shift + Backspace	Navigate to last edit location
Alt + F1	Select current file or symbol in any view
Ctrl + Alt + B	Go to implementation(s)
Ctrl + Shift + I	Open quick definition lookup
Ctrl + Shift + B	Go to type declaration
Ctrl + U	Go to super-method/super-class
Alt + Up	Go to previous method
Alt + Down	Go to next method
Ctrl +] / [	Move to code block end/start
Ctrl + Shift + M	Move caret to matching brace
Ctrl + F12	File structure popup
Ctrl + H	Type hierarchy
Ctrl + Alt + H	Call hierarchy
F2 / Shift + F2	Next/previous highlighted error
F4, Ctrl + Enter	Jump to source
Alt + Home	Jump to navigation bar
F11	Toggle bookmark
Ctrl + Shift + F11	Toggle bookmark with mnemonic
Ctrl + #[0-9]	Go to numbered bookmark
Shift + F11	Show bookmarks

Search/Replace

Ctrl + F	Find
F3	Find next
Shift + F3	Find previous
Ctrl + Shift + F	Find in path
Ctrl + R	Replace
Ctrl + Shift + R	Replace in path

Usage Search

Alt + F7	Find usages
Ctrl + F7	Find usages in file
Ctrl + Shift + F7	Highlight usages in file
Ctrl + Alt + F7	Show usages

Refactoring

Ctrl + Alt + Shift +T	Refactor this
F5 / F6	Copy / Move
Alt + Delete	Safe Delete
Shift + F6	Rename
Ctrl + F6	Change function signature
Ctrl + Alt + N	Inline Variable
Ctrl + Alt + M	Extract Method
Ctrl + Alt + V	Extract Variable
Ctrl + Alt + C	Extract Constant
Ctrl + Alt + P	Extract Parameter

VCS/Local History

Alt + BackQuote (`)	'VCS' quick popup
Ctrl + K	Commit project to VCS
Ctrl + T	Update project from VCS
Alt + Shift + C	View recent changes

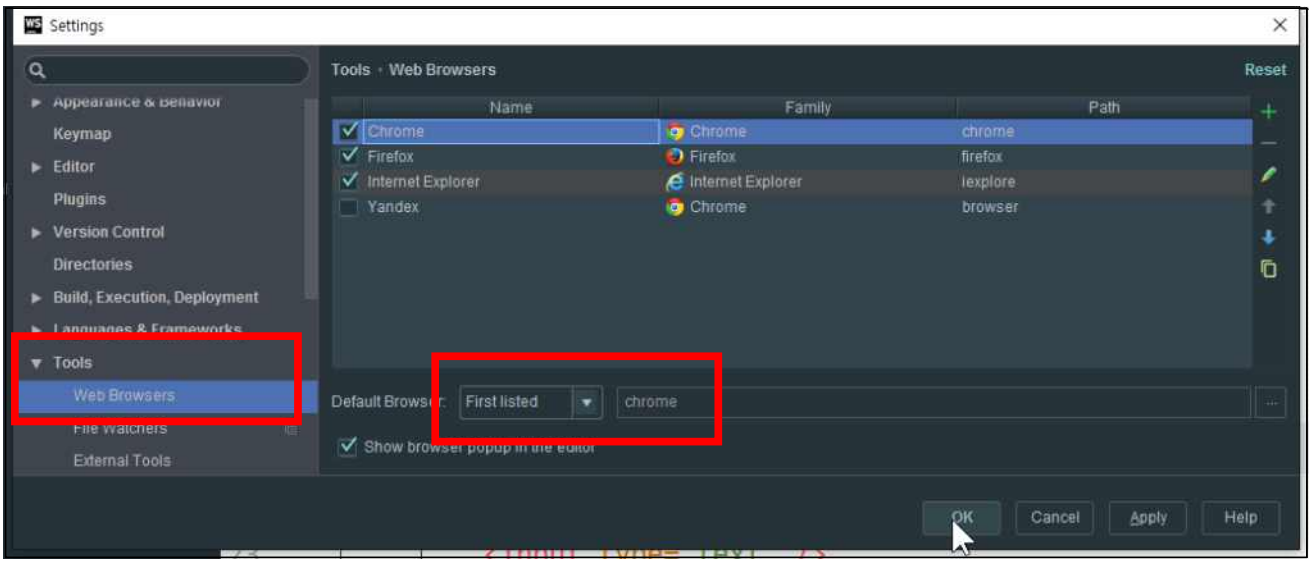
General

Double Shift	Search everywhere
Ctrl + Shift + A	Find Action
Alt + #[0-9]	Open corresponding tool window
Ctrl + Shift + F12	Toggle maximizing editor
Alt + Shift + F	Add to Favorites
Alt + Shift + I	Inspect current file with current profile
Ctrl + BackQuote (`)	Quick switch current scheme
Ctrl + Alt + S	Open Settings dialog
Ctrl + Tab	Switch between tabs and tool window

To find any action inside the IDE use Find Action (Ctrl+Shift+A)

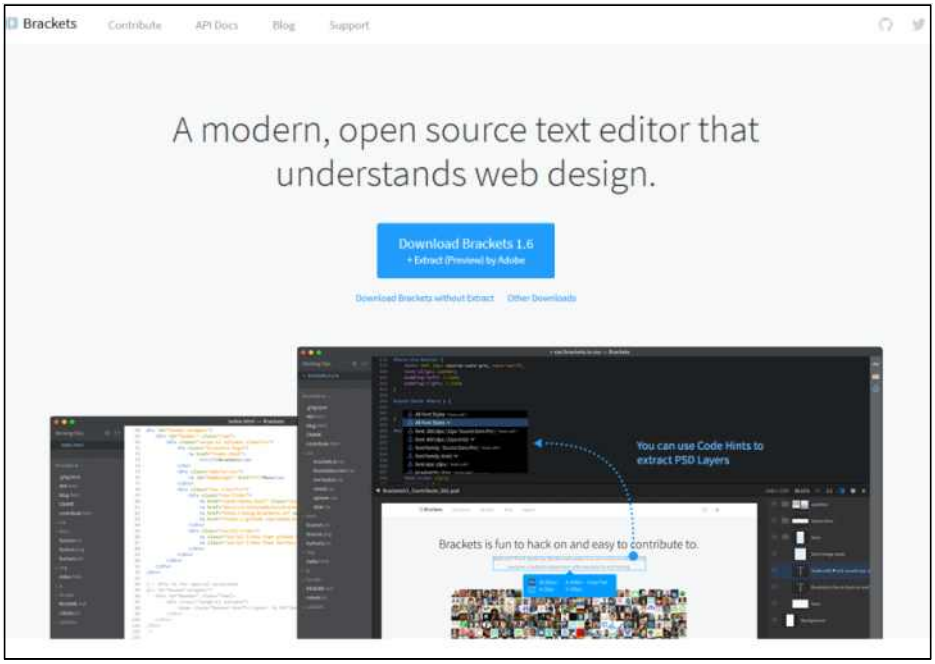
jetbrains.com/webstorm blog.jetbrains.com/webstorm @WebStormIDE

7) 기본 브라우저 변경

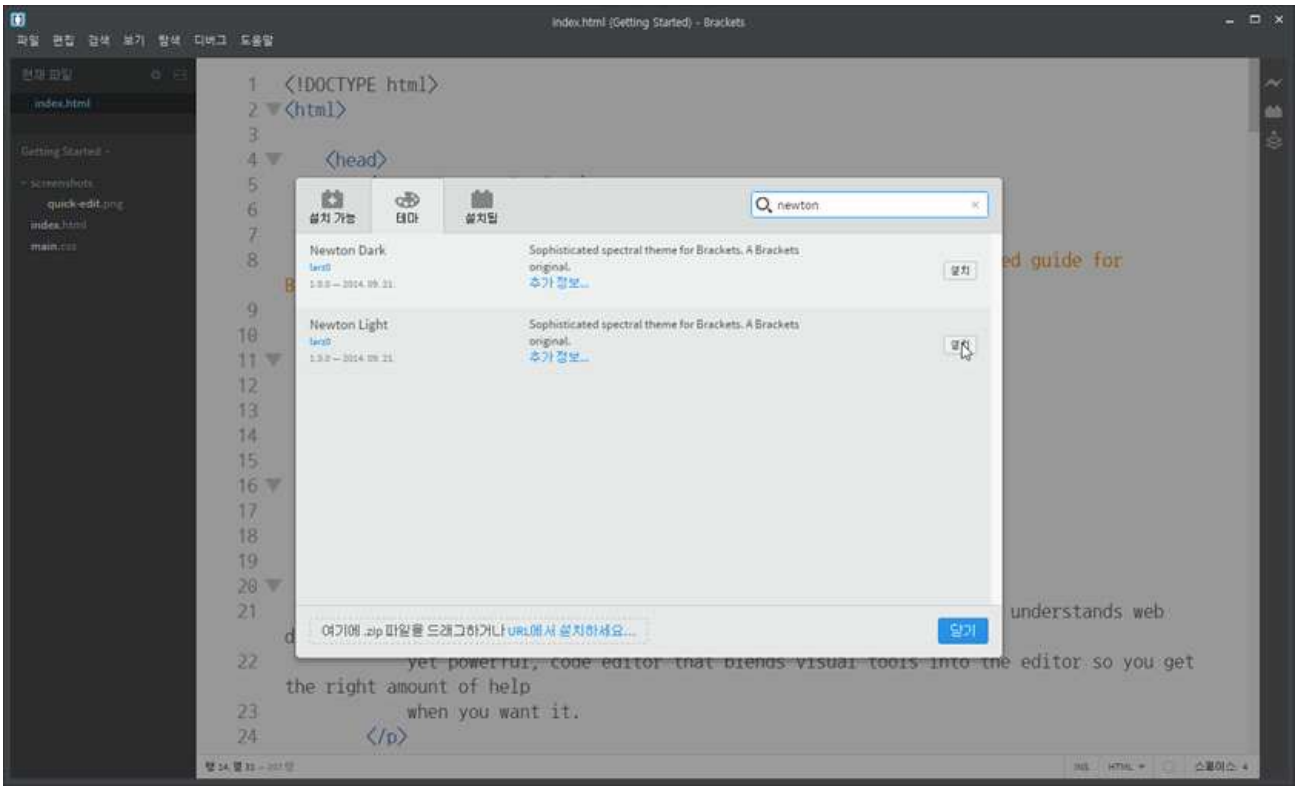


■ brackets 설치

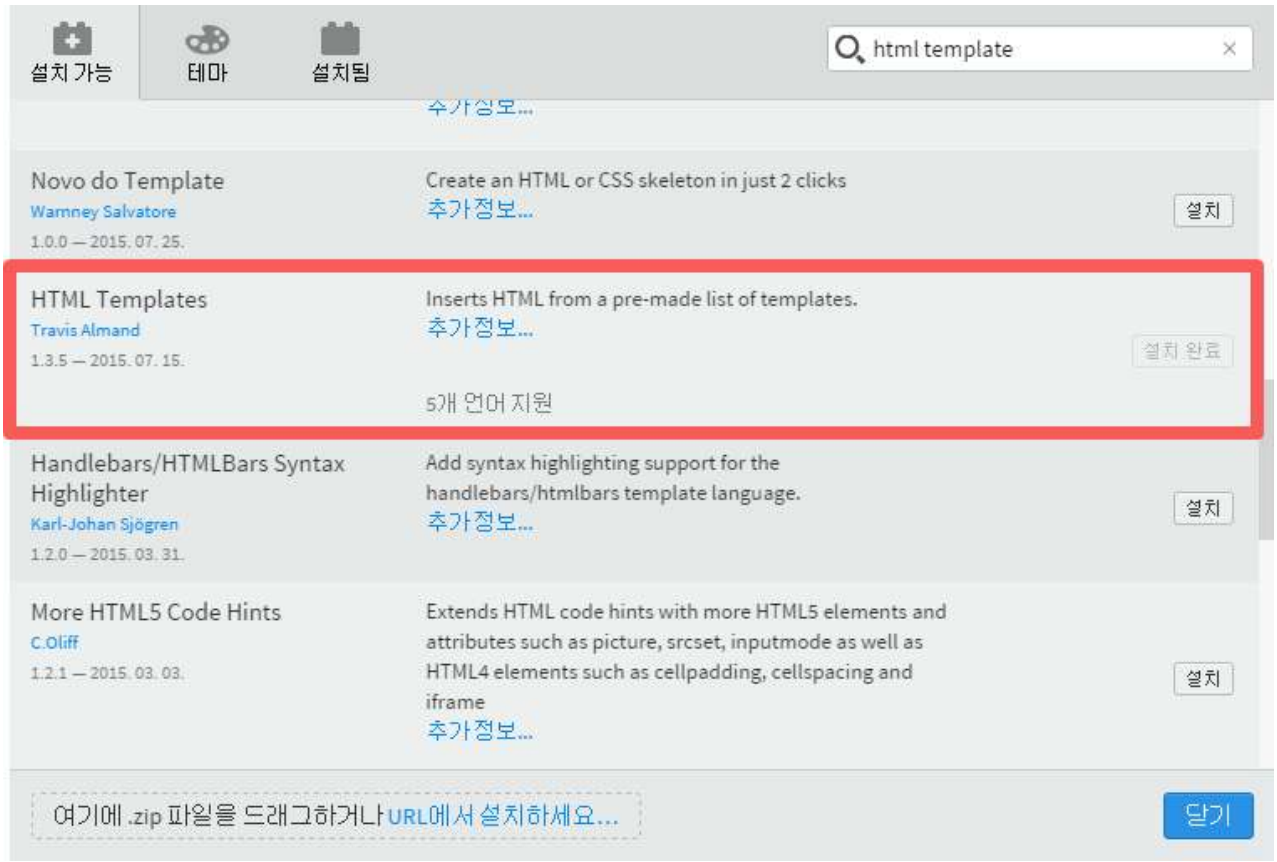
1. brakets.io



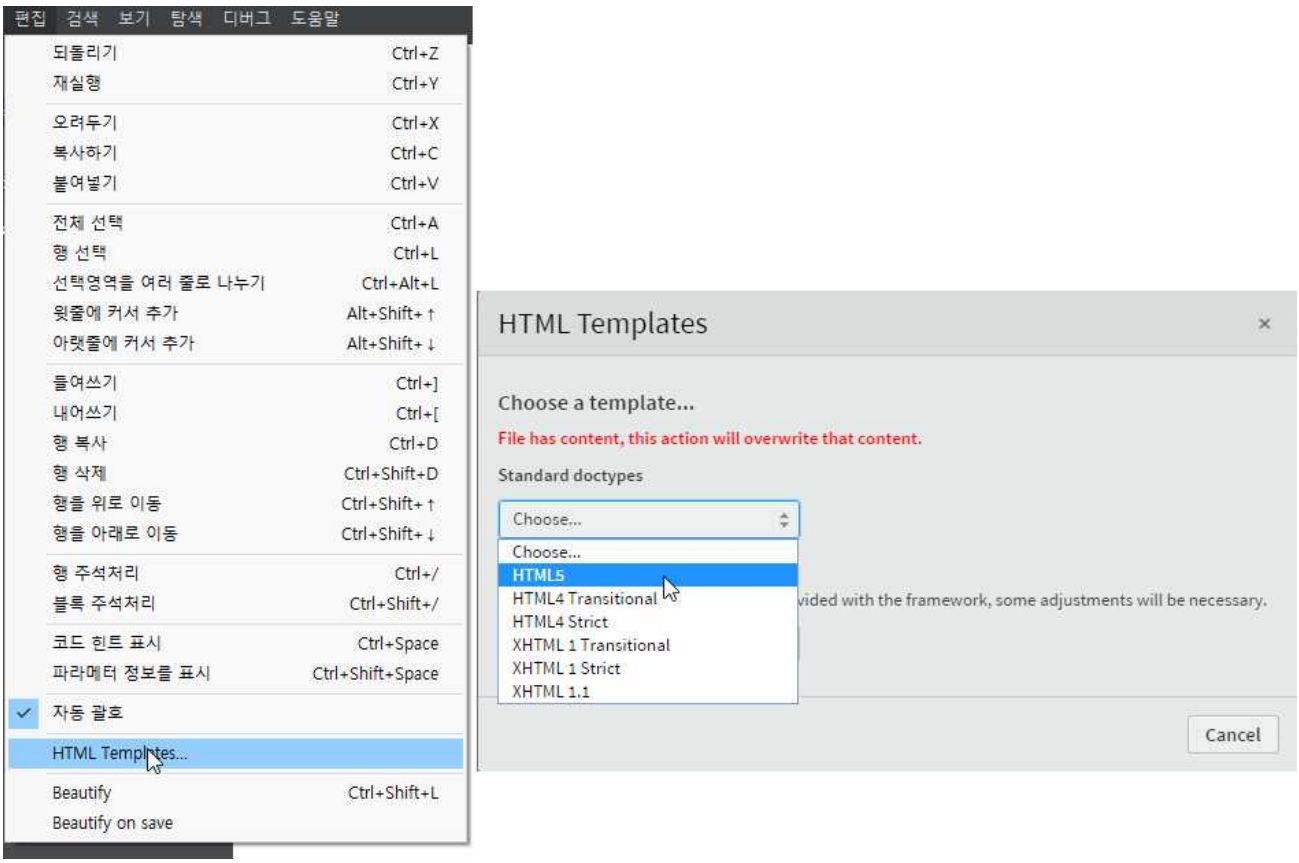
1) 테마 설정



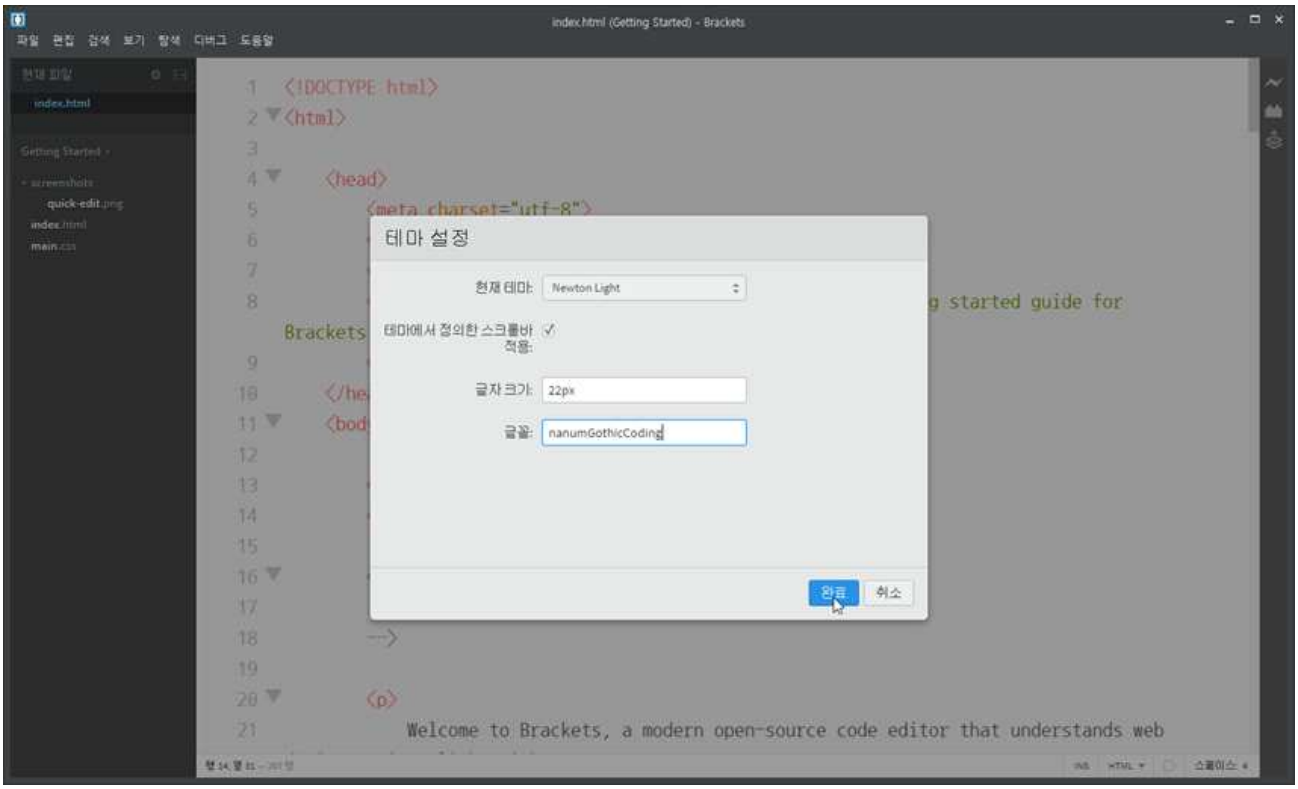
2) 템플릿 설정 확장프로그램



3) 템플릿의 사용



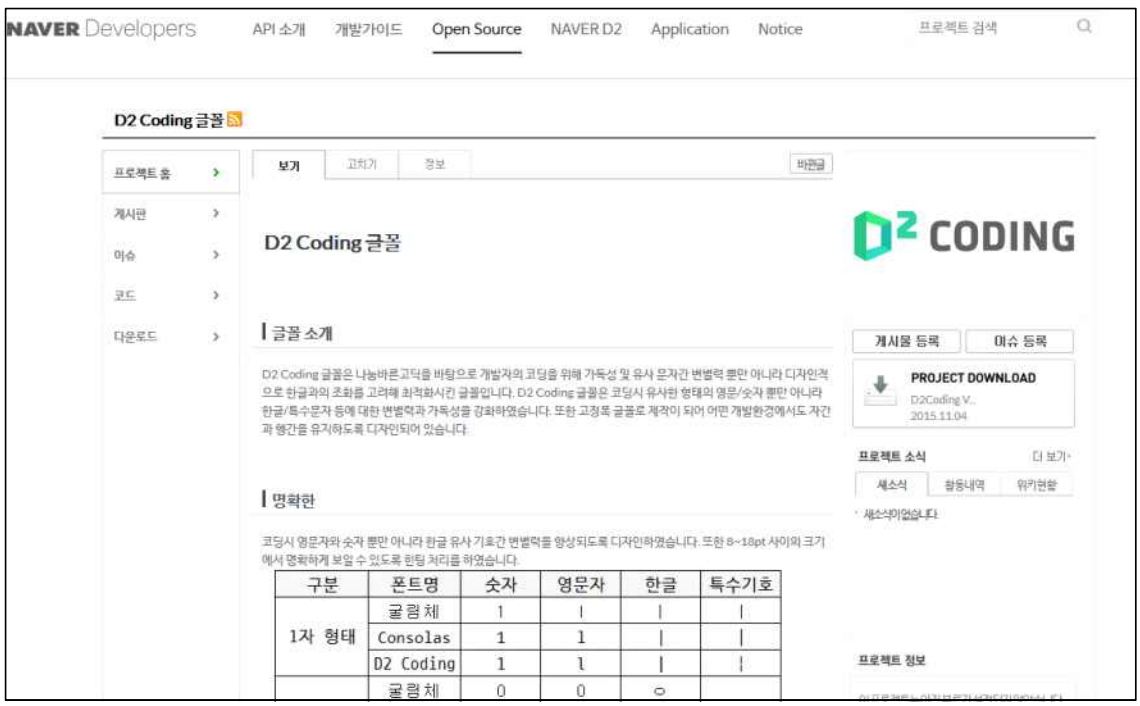
4) 테마 및 폰트 변경



5) 코딩폰트

- http://dev.naver.com/projects/d2coding

- D2 Coding폰트



■ 대화형 단일페이지 애플리케이션이란?

1. ajax를 이용하여 리로딩없이 데이터와 화면이 갱신되는 웹

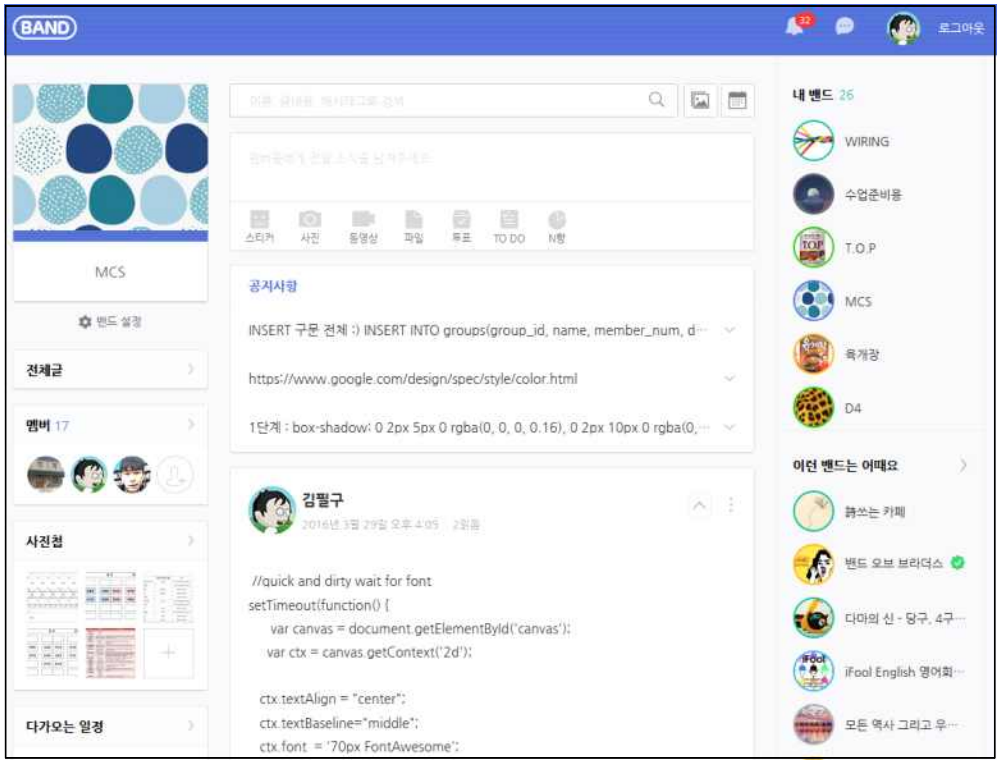


그림) 단일페이지애플리케이션으로 제작된 밴드(band.us)

- 다양한 단일페이지 애플리케이션을 볼 수 있는 사이트

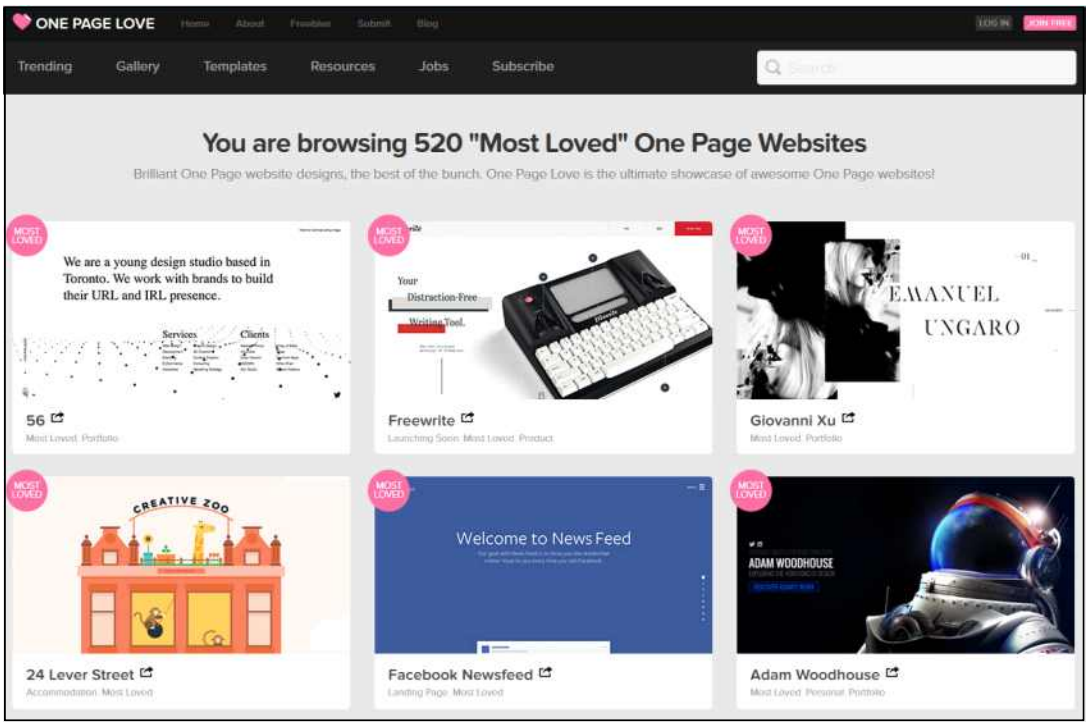


그림) https://onepagelove.com/

2) 첫 번째 요청에 HTML이 응답하고 그 이후에는 json으로 응답

- 백엔드 : 마이크로서버(서버는 json만 응답)

- 프론트엔드 : 프론트(클라이언트)에서 주소와 페이지의 변경해야 함

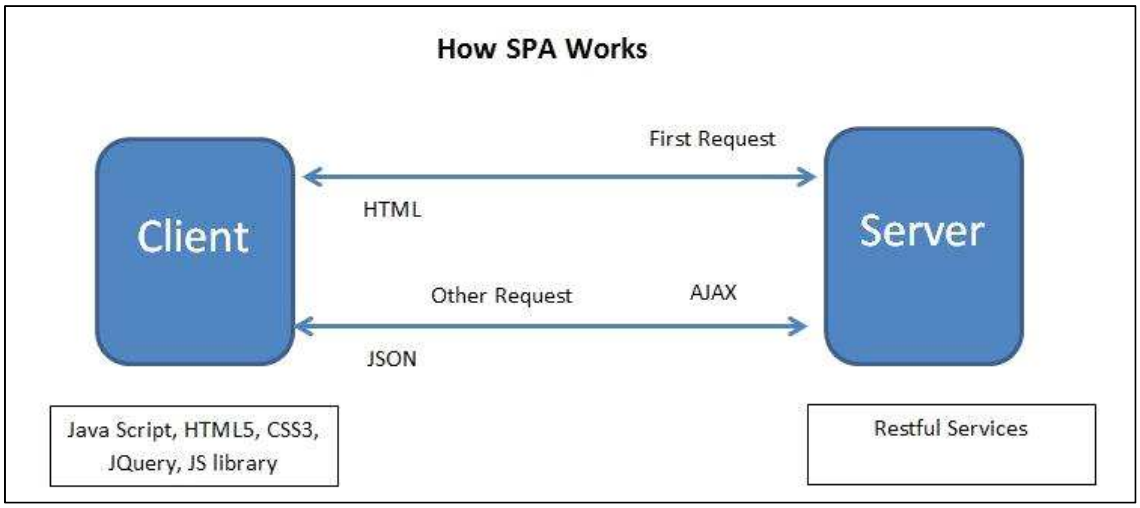


그림) SPA의 작동원리

3. 단일 페이지 애플리케이션을 여러명의 사람이 함께 제작할 경우 개발 및 유지보수가 어려움

- 한 사람이 맡을 분량을 나누기가 어려움

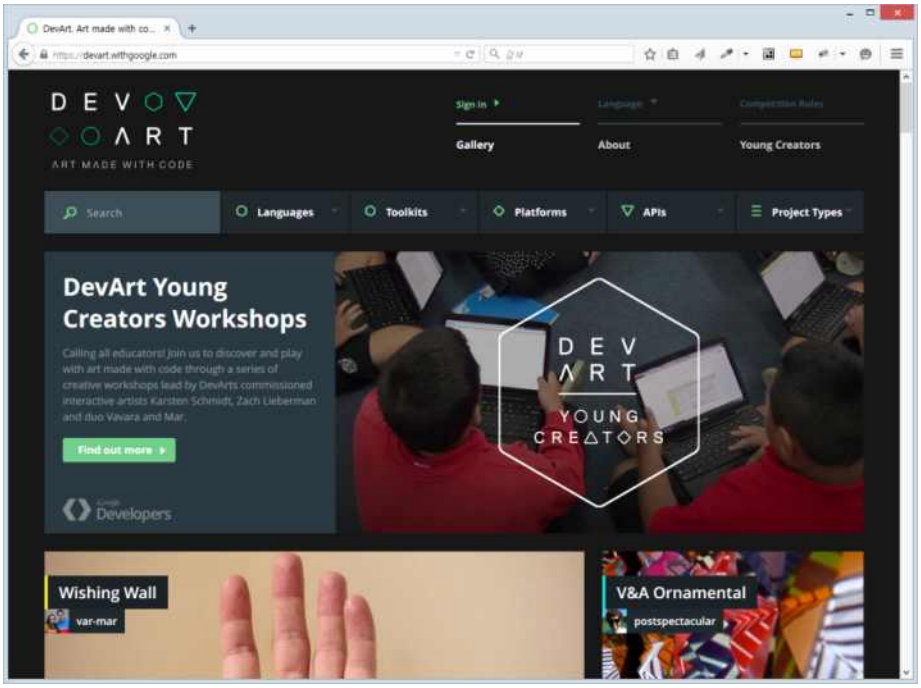
- 나눠져 있지 않으면 유지보수가 어려움

4. ClientMVC를 도입하여 프론트엔드에서 주소 / 페이지 / 데이터의 변경을 모두 담당함

■ Client MVC 왜 쓰나요?

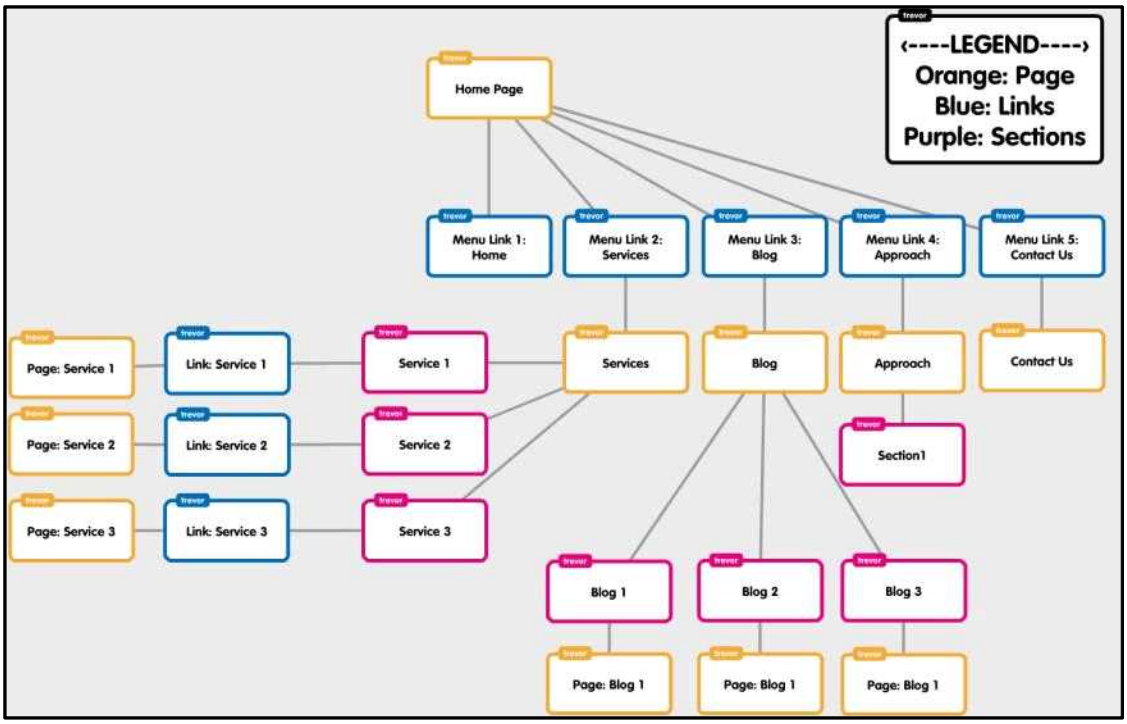
1) SPA때문에 씁니다.

2) SPA는?(Single Page Application) : 단일페이지 애플리케이션



(https://devart.withgoogle.com/)

3) 기존의 웹은 다수의 페이지 기반



4) 여러 사람이 개발할 경우 페이지별로 나눠서 개발이 가능

5) 유지보수역시 필요한 페이지만 유지보수 가능

6) 하지만 SPA이라면?

7) 쉽게 얘기하면 100개의 페이지코드(HTML,CSS,Javascript코드)가 한 페이지에 들어가 있다고 생각하면 됨)

8) 유지보수 및 개발이 쉽지 않음

9) 여러사람일 경우 한 페이지를 어떻게 나눠서 개발 및 유지보수를 할 수 있을지?

10) 구현은 싱글페이지이지만 개발 및 유지보수는 나눠서 할 수 있게

11) 페이지를 나눔

12) 하지만 페이지만 나누면 안되고 MVC로 분리하여 개발 및 유지보수를 해야 편함

13) MVC로 분리하지 않고 개발해본 사람만이 이해할 수 있는 부분!

14) 결론 : MVC로 분리하지 않고 단일페이지 개발은 힘들다.

15) 그런데 MVC는 서버측 MVC가 있고, 클라이언트 MVC가 있음

16) 우리가 배우는 Angular.js는 클라이언트 MVC(정확히는 MVW)임

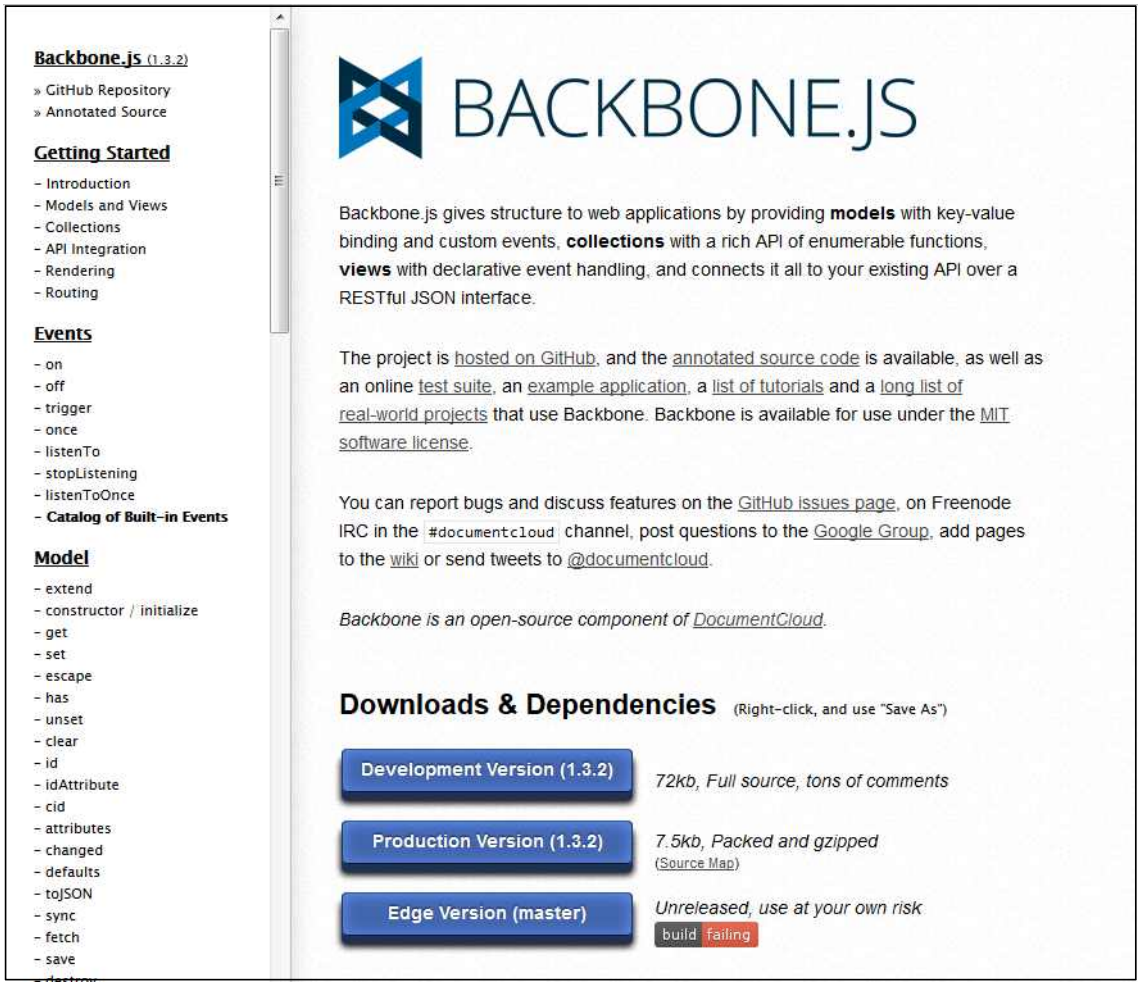
17) 그렇다면 서버는? (마이크로서버 구현)

18) 마이크로 서버란? : 서버는 페이지가 한 페이지만 존재하고, 나머지는 모두 JSON즉, 데이터만 주고받는 서버

19) 앱도 마이크로서버를 이용함

■ 다양한 Client MVC 프레임워크

1. backbone.js



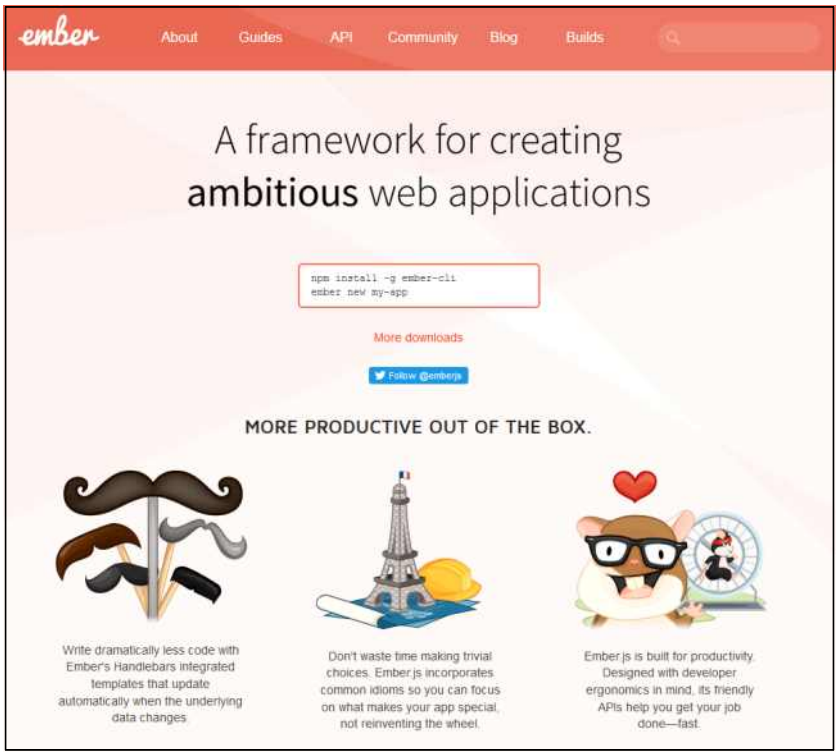
- 단일페이지 애플리케이션 초창기에 많이 사용
- 위에서 설명한 '밴드'도 backbone으로 만들어져 있음
- jquery / underscore에 의존적
- Model객체가 순수 자바스크립트가 아님(유틸리티 메서드가 많음)

2. angular.js

- 현재 가장 많이 사용하는 프레임워크

- 구글이 만들어서 구글의 다양한 웹애플리케이션에서 사용

3. ember.js

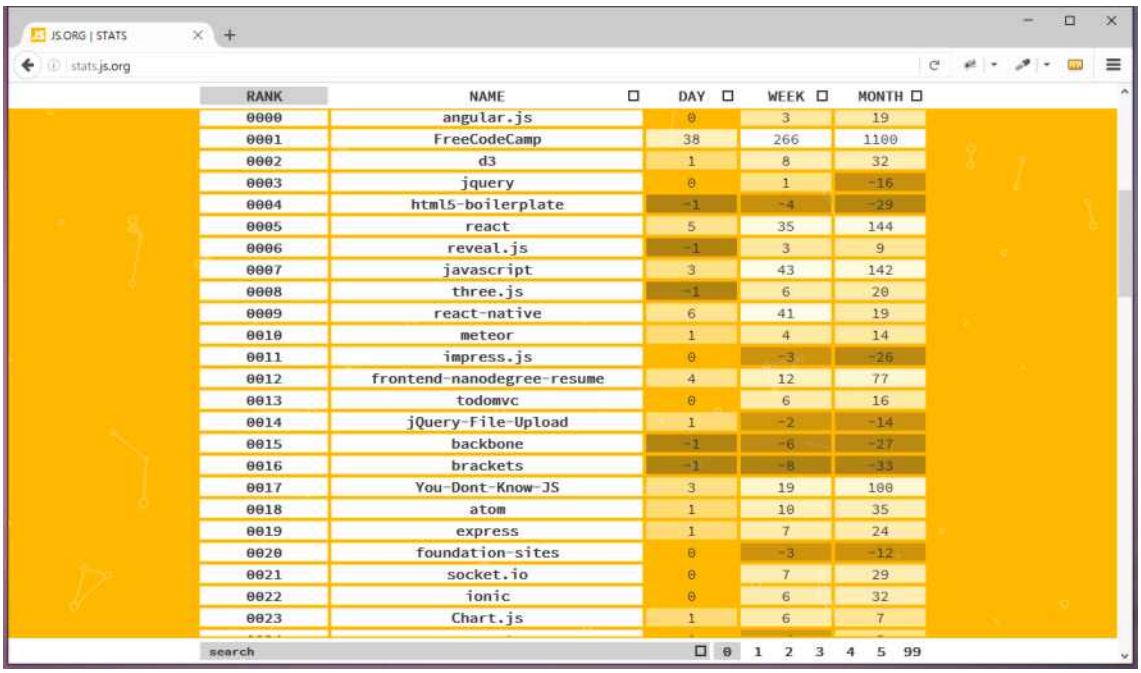


4. 그 외에도 다양한 프레임워크가 있음

■ angular.js

0) stats.js.org

- github에서 가장 '핫'한 프레임워크(?)의 순위를 나타내는 사이트

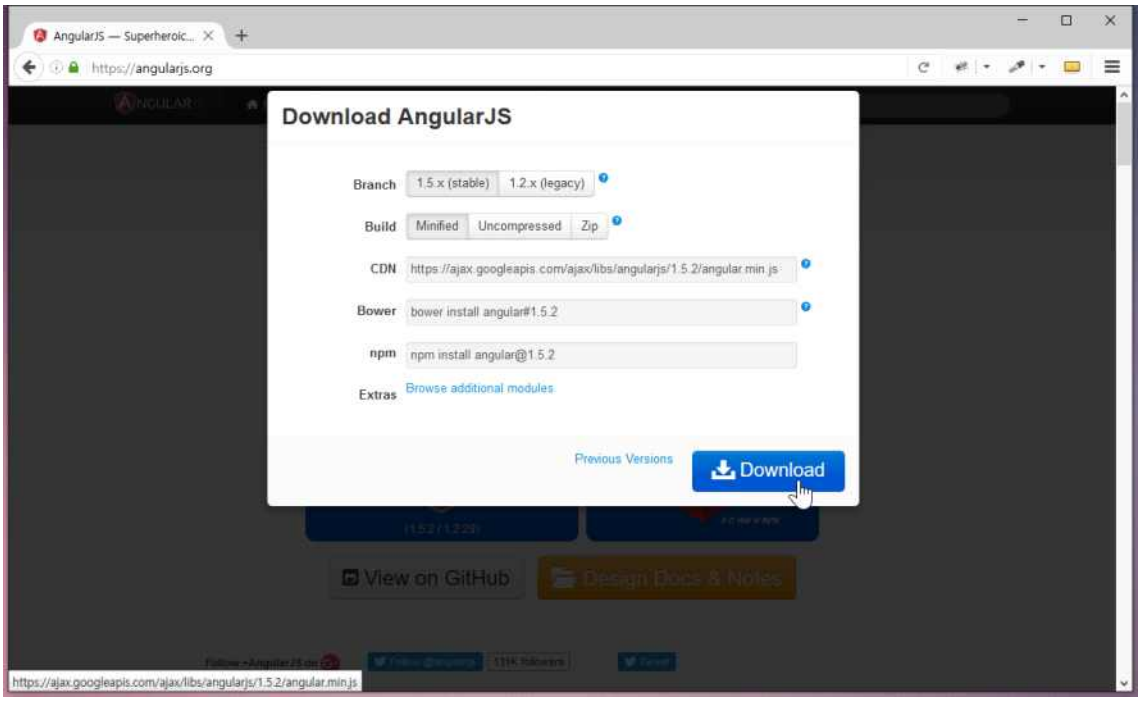


RANK	NAME	DAY	WEEK	MONTH
0000	angular.js	0	3	19
0001	FreeCodeCamp	38	266	1100
0002	d3	1	8	32
0003	jquery	0	1	-16
0004	html5-boilerplate	-1	-4	-29
0005	react	5	35	144
0006	reveal.js	-1	3	9
0007	javascript	3	43	142
0008	three.js	-1	6	20
0009	react-native	6	41	19
0010	meteor	1	4	14
0011	impress.js	0	-3	-26
0012	frontend-nanodegree-resume	4	12	77
0013	todomvc	0	6	16
0014	jQuery-File-Upload	1	-2	-14
0015	backbone	-1	-6	-27
0016	brackets	-1	-8	-33
0017	You-Dont-Know-JS	3	19	100
0018	atom	1	10	35
0019	express	1	7	24
0020	foundation-sites	0	-3	-12
0021	socket.io	0	7	29
0022	ionic	0	6	32
0023	Chart.js	1	6	7

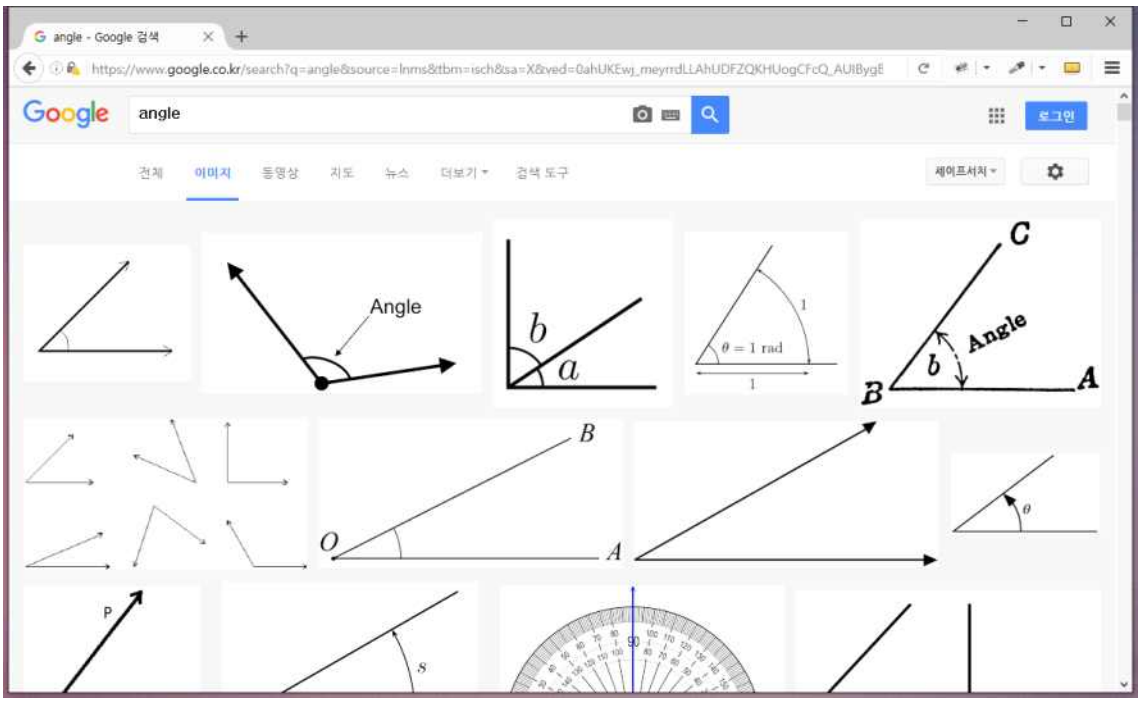
- https://www.madewithangular.com/#/

1) angularjs.org

2) angularjs의 다운로드



3) angular란? 각진이라는 뜻으로 각도기 즉, '<' 이 태그를 의미함



4) 'ng-' 지시어를 잘 알아야 함

5) ng-app 속성을 부여하면 그 태그안에서 angular.js가 실행됨

왜 ng인가?
홈페이지의 fag를 보면
'Because HTML has Angular brackets and "ng" sounds like "Angular".'라고 쓰여있음.
한글로 해석하면 '<'태그의 발음이 ng와 비슷하다고 함

4) 웹표준을 위하여 'data-'를 사용할 수도 있음

■ angular.js의 특징

1) 양방향 데이터 바인딩

- 양방향 바인딩의 장점

- 양방향 바인딩이 아닌 경우 일일이 페이지 내부를 모두 변경해야 함

2) MVC 구조(MVW : Model View Whatever)

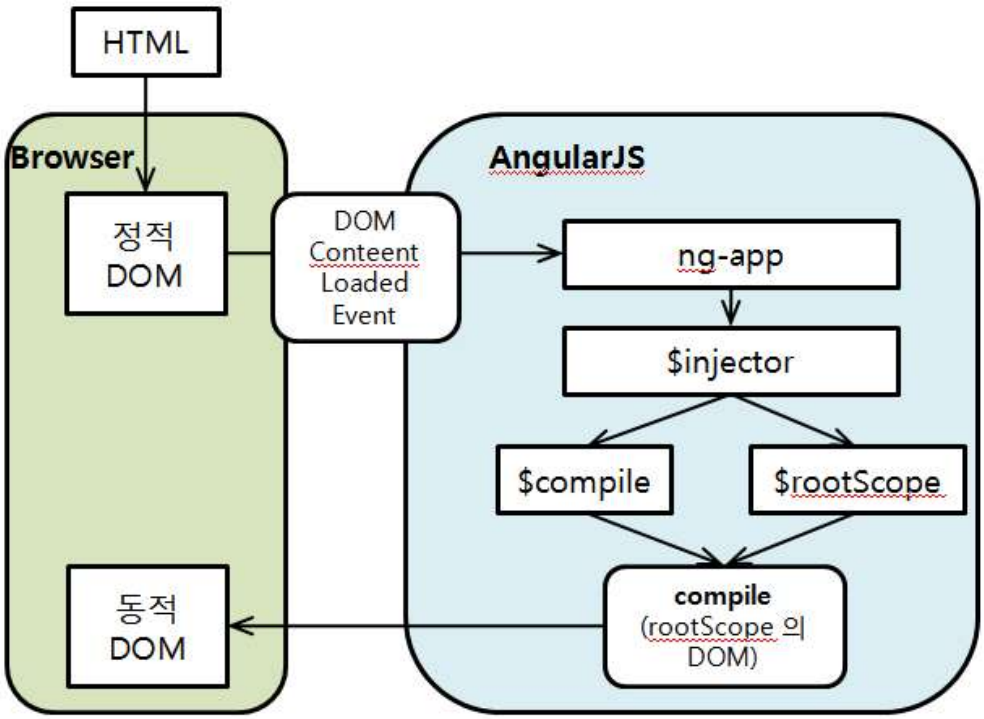
3) 지시자(directive)를 이용한 HTML 확장

4) 의존관계 주입(DI)

5) 단일페이지 웹 애플리케이션을 위한 라우터

6) 선언형 프로그래밍(jquery같이 명령형 프로그래밍과 개념이 다름)

■ angular.js의 작동방식



1) 브라우저가 html을 로드(DOM을 파싱)

2) Angular.js를 로드

3) DOM Content Loaded Event를 기다림

4) DOM이 모두 로드되면 ng-app 지시자를 찾음

5) ng-app에서는 dependency injection 을 위해 사용되는 $injector를 생성

6) injector 지시어는 어플리케이션의 모델을 위한 컨텍스트가 되는 루트 스코프 생성

7) 최종적으로 ng-app을 기준으로 하위DOM을 컴파일하고 rootScope와 링크

■ jquery와의 비교

1) input에 있는 글자를 h1요소에 출력하는 예제

```
<div id="txtBox">
	<input type="text" />
	<h1>안녕하세요?</h1>
</div>
<script src="js/zepto.js"></script>
<script>

	$("input").keyup(function() {
		var txt = $(this).val();

		$("h1").text(txt);
	});

</script>
```

■ angular.js의 시작

```
<!DOCTYPE html>
<html lang="ko" data-ng-app>
 <head>
  <meta charset="UTF-8">
  <title>Angular.js 예제</title>
 </head>
 <body>
 </body>
</html>
```

1) data-ng-app속성이 있어야 angular.js가 작동

2) 책이나 예제소스에 있는 ng-app과 같음(웹표준때문 data-ng-app을 추천)

'ng-' 지시어(directive)의 이름 표기법

예) ng-model 을,

ng-model,ng:model,ng_model,x-ng-model,x-ng:model,x_ng:model,data-ng-model,data-ng:model,data-ng_model 등으로 전부 사용할 수 있음

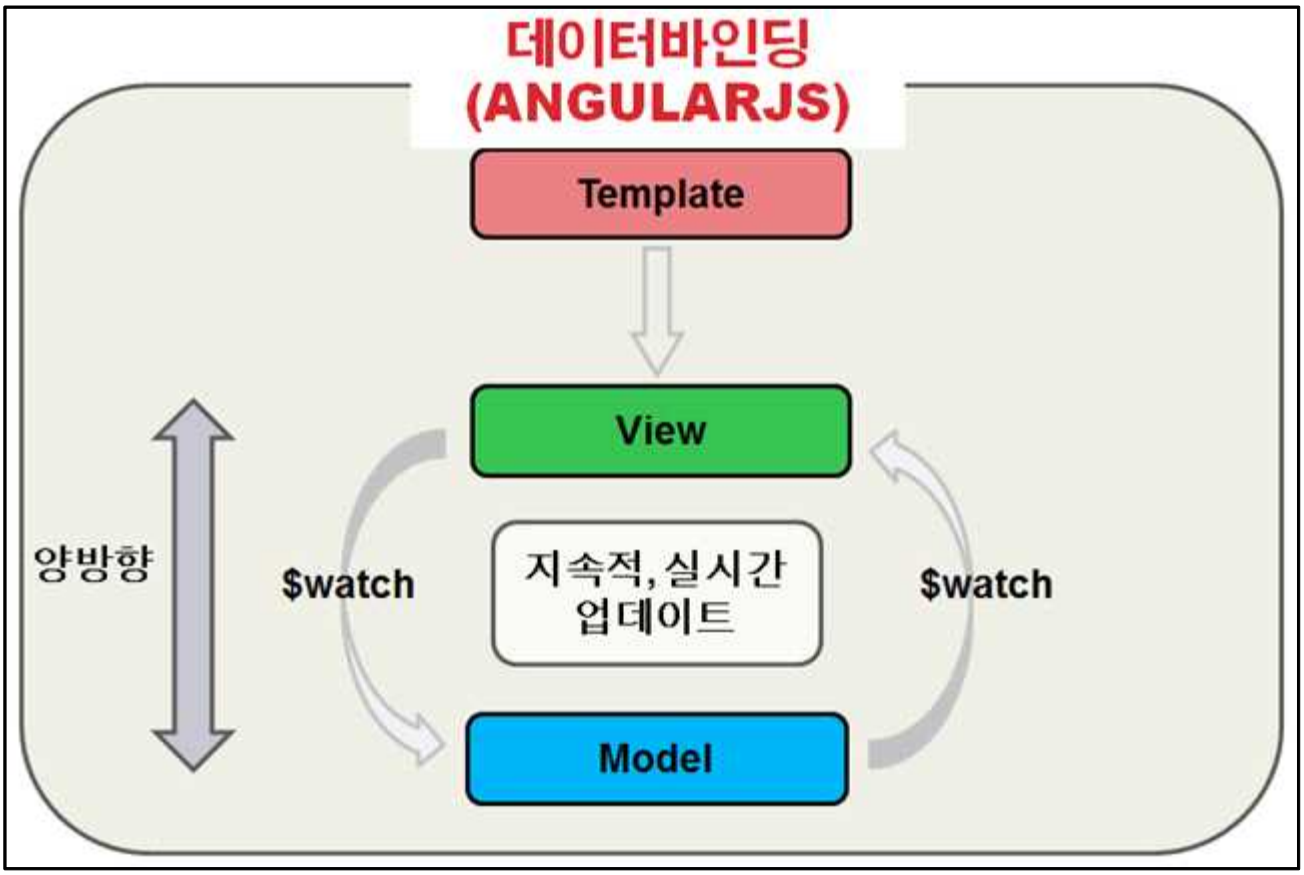
추천 :data-ng-model (HTML5표준)

■ angular.js의 완전기초

1) 'ng-' 지시어(directive)를 많이 알아야 함

2) {{ }}(인터폴레이션) : 모델과 바인딩 (ng-bind와 같음)

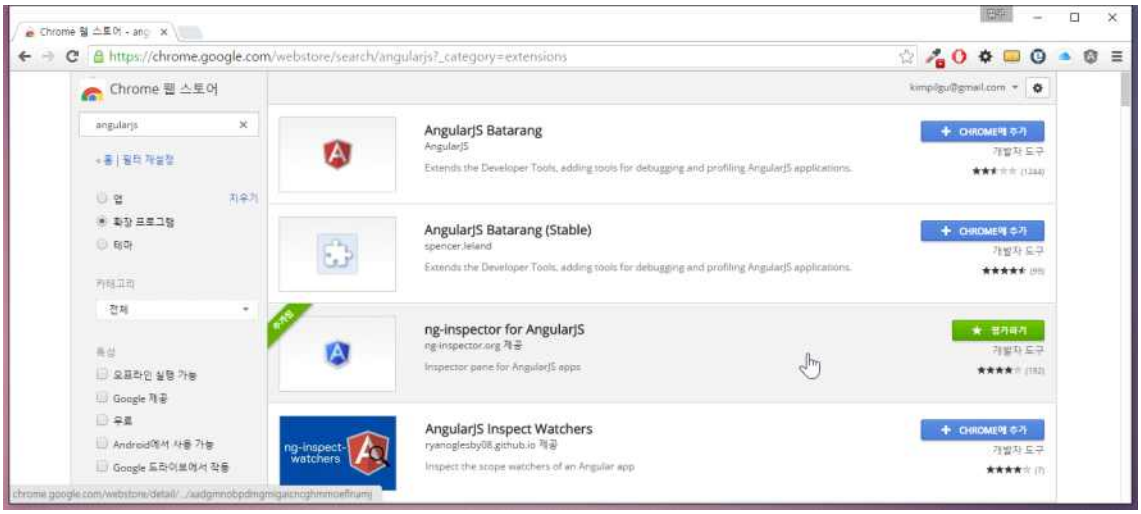
3) 데이터바인딩 : 모델과 양방향 바인딩



4) ng-controller : 컨트롤러 함수의 설정

5) $scope : 스코프는 모델의 적용범위를 나타냄

- 크롬 확장프로그램으로 스코프 알아보기(ng-inspector)



- 스코프등 모델을 한 눈에 볼 수 있음

■ Hello Angular예제

1) jquery로

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>jquery 복습</title>
<link rel="stylesheet" href="css/kakao.font.css"/>
<style>
	body {
		font-family: 'Kakao', sans-serif;


	}
    h1,h2 {
	 font-weight: 200;
    }
	input {
		font-family: 'Kakao', sans-serif;
		font-size:22px;
	}
</style>
</head>
<body>
	<div id="txtBox">
		<input type="text" />
		<h1>안녕하세요?</h1>
	 <h2>여기도</h2>
	</div>
	<script src="js/jquery.js"></script>
	<script>
```

```
            $("input").keyup(function() {
           var txt = $(this).val();

            $("h1").text(txt);
            $("h2").text(txt);

        });

      </script>
</body>
</html>
```

2) angualr를 이용한 예제

```
<!DOCTYPE html>
<html ng-app="helloApp">
<head>
<meta charset="UTF-8">
<title>Hello Angular.js</title>
<link rel="stylesheet" href="css/kakao.font.css"/>
<style>
      body {
            font-family: 'Kakao', sans-serif;

      }
   h1,h2 {
       font-weight: 200;
   }
      input {
            font-family: 'Kakao', sans-serif;
```

```
            font-size:22px;
      }
</style>
</head>
<body ng-controller="helloController">
      <div>
            <input type="text" ng-model="txt"/>
            <h1 ng-bind="txt"></h1>
       <h2>{{txt}}</h2>
      </div>

   <!-- angular library -->
      <script src="js/angular.min.js"></script>
      <script>

       //이건 문법!~
       var app = angular.module("helloApp",[]);

       //이건 문법!~
       app.controller("helloController",["$scope",function($scope){

           $scope.txt = "후후후";

           //$scope은 모델!
           //alert("zzz");
       }]);



      // @RequestMapping("xxx.html")
      // public void dsadsf(Model model) {
      //     model.addAttribute("txt","후후후");
```

```
      // }


      </script>
 </body>
 </html>
```

■ 명령형 / 선언형 프로그래밍

- 명령형 프로그래밍은 작동하는 모든 코드를 일일히 작성

- 선언형 프로그래밍의 경우 선언만 해두면 프레임워크가 알아서 작동

- 저수준(low-level) 컨트롤이 필요없음

- 명령형 예제(jquery를 이용)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>명령형 프로그래밍</title>
    <style>

        img {
            width:300px;
            box-shadow: 0 16px 28px 0 rgba(0, 0, 0, 0.22), 0 25px 55px 0 rgba(0,
0, 0, 0.21);
        }

        #toggleImg {
            transition: .4s ease;
            transform:perspective(500px) rotateY(0deg) scale(1);
        }
        #toggleImg.mini {
            transform:perspective(500px) rotateY(720deg) scale(.5);
        }
```

```
        #img1 {
            display: none;
        }
    </style>
</head>
<body>
    <h1>명령형 프로그래밍은?</h1>
    <div id="box">
        <p>
            <input type="checkbox" id="toggle">
            <label for="toggle">숨기기</label>
        </p>
        <img src="poster/p6.png" width="300" id="toggleImg"/>
        <img src="poster/p2.png" width="300" id="img1"/>
        <img src="poster/p4.png" width="300" id="img2"/>
    </div>
<script src="js/jquery.js"></script>
<script>
    $("#toggle").change(function () {
       var flag = $(this).prop("checked");
        if(flag) {
            $("#toggleImg").addClass("mini");
            $(this).next().text("보이기");
            $("#img1").show();
            $("#img2").hide();
        }else {
            $("#toggleImg").removeClass("mini");
            $(this).next().text("숨기기");
            $("#img2").show();
            $("#img1").hide();
        }
    });

</script>
</body>
</html>
```

- 선언형 프로그래밍

```
<!DOCTYPE html>
<html lang="ko" ng-app="declarativeApp">
<head>
    <meta charset="UTF-8">
    <title>선언형 프로그래밍</title>
    <style>

        img {
            width:300px;
            box-shadow: 0 16px 28px 0 rgba(0, 0, 0, 0.22), 0 25px 55px 0 rgba(0,
0, 0, 0.21);
        }

        #toggleImg {
            transition: .4s ease;
            transform:perspective(500px) rotateY(0deg) scale(1);
        }
        #toggleImg.mini {
            transform:perspective(500px) rotateY(720deg) scale(.5);
        }

    </style>
</head>
<body>
<h1>선언형 프로그래밍은?</h1>
<div id="box" ng-controller="ctrl">
    <p>
        <input type="checkbox" id="toggle" ng-model="toggle">
        <label for="toggle" ng-bind="toggle?'보이기':'숨기기'"></label>
    </p>
    <img          src="poster/p6.png"          width="300"          id="toggleImg"
ng-class="{mini:toggle}"/>
    <img src="poster/p2.png" width="300" id="img1" ng-show="toggle"/>
    <img src="poster/p4.png" width="300" id="img2" ng-hide="toggle"/>
</div>
<script src="js/angular.min.js"></script>
<script>
```

```
    var app = angular.module("declarativeApp",[]);

    app.controller("ctrl",["$scope",function ($scope) {

    }]);

</script>
</body>
</html>
```

■ 많이 사용하는 ng-repeat 사용해보기

1) jquery 이용예제

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>영화내용 출력페이지</title>
<style>
    ul {
        list-style: none;
        padding: 0;
        margin: 0;
    }
    li {
        float: left;
        width:100px;
        padding:20px;
        margin:10px;
        box-shadow: 0 12px 15px 0 rgba(0, 0, 0, 0.24), 0 17px 50px 0 rgba(0, 0,
0, 0.19);
    }
```

```
    h3 {
        margin: 0;
        text-align: center;
        font-size:13px;
    }
</style>
</head>
<body>
       <h1>영화목록<em>?</em>개</h1>
       <button id="deleteBtn">삭제</button>
       <ul>

       </ul>
    <!-- underscore template -->
    <script type="text/template" id="movieTmp">
    <% $(movies).each(function() {%>
    <li>
        <img src="poster/<%=this.poster%>" width="100"/>
        <h3><%=this.name%></h3>
    </li>
    <%}) %>
    </script>
<script src="js/jquery.js"></script>
<script src="js/underscore-min.js"></script>
<script>

var movies = [
{poster:"p1.png",director:"바박                                나자피",name:"런던해즈폴른
",grade:106,gradeNum:12},
{poster:"p2.png",director:"잭스나이더",name:"배트맨대슈퍼맨
",grade:37,gradeNum:4},
```

```
{poster:"p3.png",director:"조정래",name:"귀향",grade:51,gradeNum:6},
{poster:"p4.png",director:"모홍진",name:"널기다리며",grade:68,gradeNum:11},
{poster:"p5.png",director:"이준익",name:"동주",grade:29,gradeNum:4},
{poster:"p6.png",director:"바이론 하워드",name:"주토피아",grade:39,gradeNum:4}
];
    var tmp = _.template($("#movieTmp").html());

    var $em = $("em");

    $("ul").append(tmp({"movies":movies}));

    $em.text(movies.length);

    $("#deleteBtn").click(function() {

        $("li").eq(0).remove();
        $em.text($("li").size());

    });

</script>
</body>
</html>
```

- $.each()함수를 이용하여 json만큼 반복하여 jquery요소객체 생성후 서로 append 하여 DOM을 생성

2) angular.js를 이용한 예제

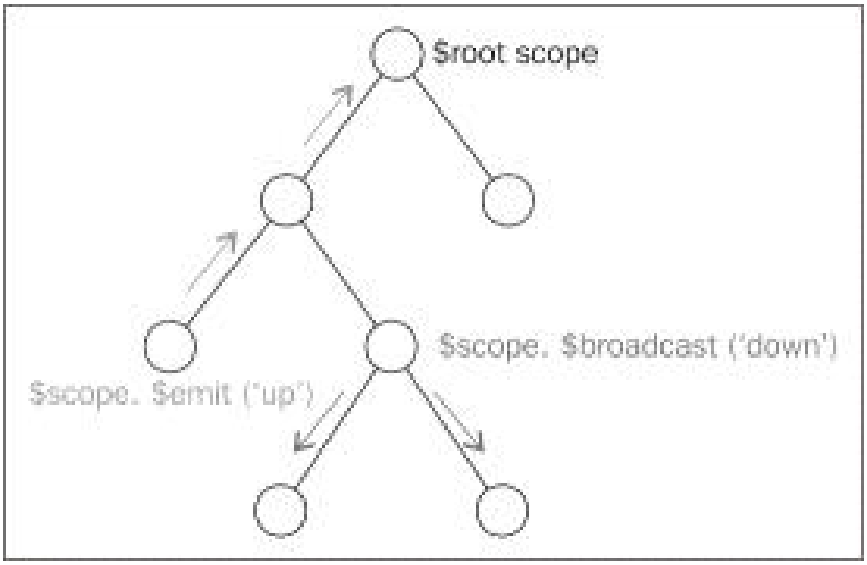
```
<!DOCTYPE html>
<html ng-app="repeatApp">

<head>
    <meta charset="UTF-8">
    <title>영화내용 출력페이지</title>
    <link rel="stylesheet" href="css/kakao.font.css" />
    <style>
        body {
            font: 200 20px "Kakao", sans-serif;
        }

        ul {
            list-style: none;
            padding: 0;
            margin: 0;
        }

        li {
            float: left;
            width: 100px;
            padding: 20px;
            margin: 10px;
            box-shadow: 0 12px 15px 0 rgba(0, 0, 0, 0.24), 0 17px 50px 0 rgba(0,
0, 0, 0.19);
        }

        h3 {
            margin: 0;
            text-align: center;
            font-size: 13px;
```

```
        }
    </style>
</head>

<body ng-controller="repeatController">
    <h1>영화목록<em>{{movies.length}}</em>개</h1>
    <button ng-click="deleteMovie(movies)">삭제</button>
    <ul>
        <!-- angular는 html을 그자체로 템플릿으로 이용 -->
        <!--
        <c:forEach item="${movies}" var="movie">
        <li>
            <img src="poster/${movie.poster}" width="100" />
            <h3>{movie.name}</h3>
        </li>
        </c:forEach>
        -->
        <li ng-repeat="movie in movies">
            <img src="poster/{{movie.poster}}" width="100" />
            <h3>{{movie.name}}</h3>
        </li>
    </ul>
    <script src="js/angular.min.js"></script>
    <script>
        var app = angular.module("repeatApp", []);

        app.controller("repeatController", ["$scope", function ($scope) {

            //모델인 movies가 필요하기 때문에 인자로 넣어줬습니다.
            $scope.deleteMovie = function (movies) {
                movies.shift();
```

```
            };

            $scope.movies = [
{poster:"p1.png",director:"바박                              나자피",name:"런던해즈폴른
",grade:106,gradeNum:12},
{poster:"p2.png",director:"잭스나이더",name:"배트맨대슈퍼맨
",grade:37,gradeNum:4},
{poster:"p3.png",director:"조정래",name:"귀향",grade:51,gradeNum:6},
{poster:"p4.png",director:"모홍진",name:"널기다리며",grade:68,gradeNum:11},
{poster:"p5.png",director:"이준익",name:"동주",grade:29,gradeNum:4},
{poster:"p6.png",director:"바이론 하워드",name:"주토피아",grade:39,gradeNum:4}
];

}]);
    </script>
</body>

</html>
```

- 템플릿 형식으로 간단하게 데이터 바인딩이 가능함

■ scope의 범위와 계층구조



1) 부모 $rootScope객체가 있고 자식 $scope가 tree구조로 생김

■ 스코프내의 함수를 이용하는 방법

HTML 마크업 영역

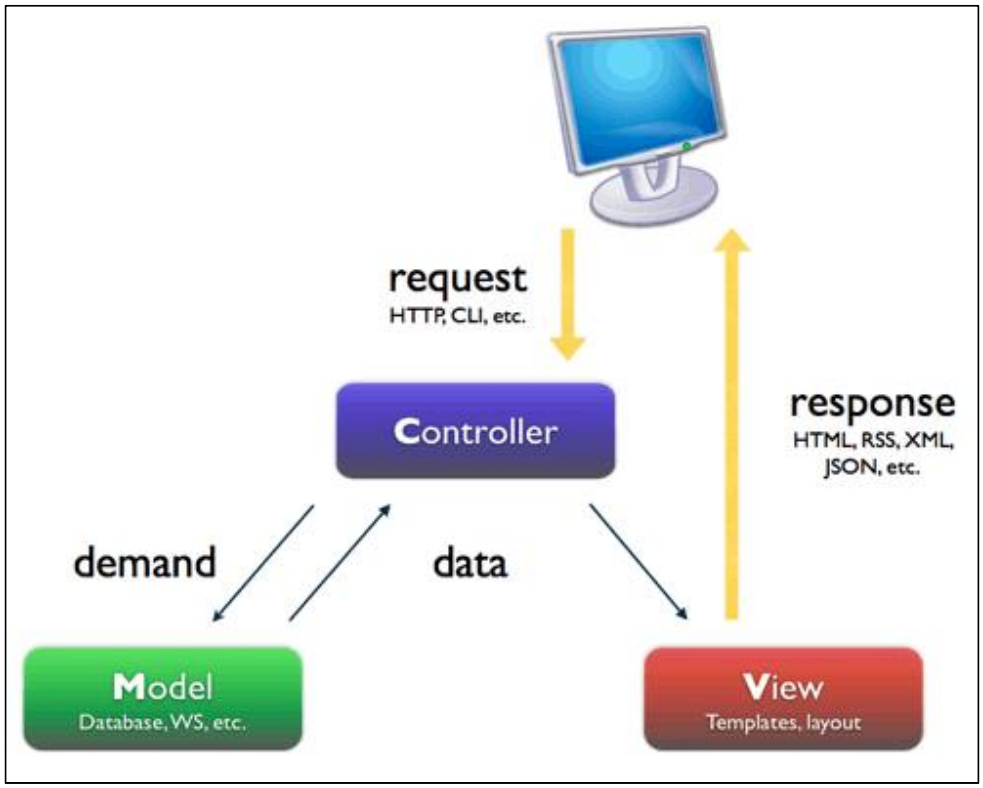
```
<ul id="movieList" data-ng-controller="moviesController">
	<li data-ng-repeat="movie in movies">
		<a href="movie.jsp">
			<img src="poster/{{movie.poster}}"
			width="230" height="336" />
			<div >
				<h3>{{movie.name}}</h3>
				<div class="box_grade">
					<strong class="txt_grade"
style="width:{{getStarWidth(movie)}}px">{{getAvgGrade(movie)}}</strong>
				</div>
			</div>
		</a>
	</li>
</ul><!-- //movieList -->
```

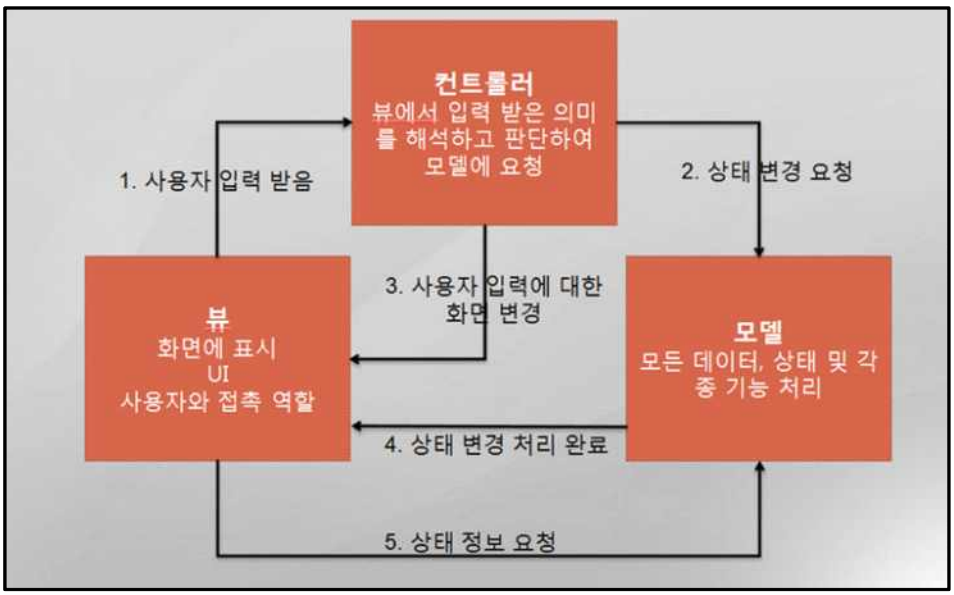
script영역

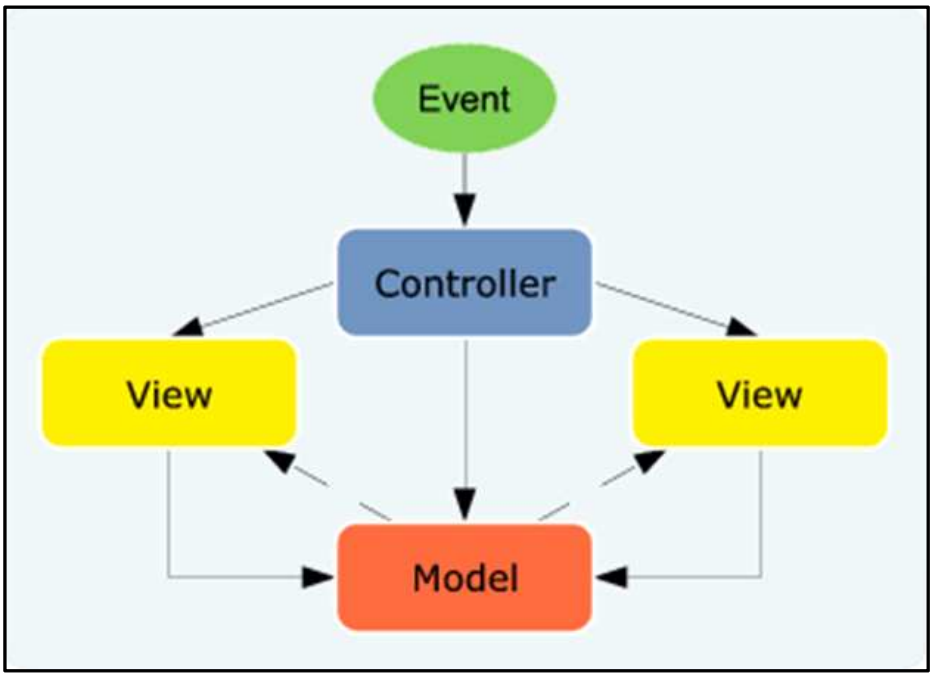
```
function moviesController($scope) {
```

```
	$scope.movies = [
		{poster:"Cinderella.jpg",director:"케네스    브레너",name:"신데렐라
",grade:13,gradeNum:2},
		{poster:"CityLights.jpg",director:"찰리채플린",name:"시티라이트
",grade:37,gradeNum:4},
		{poster:"EleanorRigby.jpg",director:"네드  벤슨",name:"엘리노어  릭
비",grade:51,gradeNum:6},
		{poster:"Fast_Furious7.jpg",director:"제임스 완",name:"분노의 질주
7",grade:68,gradeNum:11},
		{poster:"Kingsman.jpg",director:"매튜                본",name:"킹스맨
",grade:0,gradeNum:0},
	];

	$scope.getAvgGrade=function(movie) {

		if(movie.gradeNum!=0) {
			return Math.floor(movie.grade/movie.gradeNum*100)/100;
		}
		return movie.gradeNum;
	}

	$scope.getStarWidth=function(movie) {
		return this.getAvgGrade(movie)*12;
	}
}
```

■ MVC의 이해


request
HTTP, CLI, etc.
Controller
response
HTML, RSS, XML,
JSON, etc.
demand
data
Model
Database, WS, etc.
View
Templates, layout


컨트롤러
뷰에서 입력 받은 의미를 해석하고 판단하여 모델에 요청
1. 사용자 입력 받음
2. 상태 변경 요청
3. 사용자 입력에 대한 화면 변경
뷰
화면에 표시
UI
사용자와 접촉 역할
모델
모든 데이터, 상태 및 각종 기능 처리
4. 상태 변경 처리 완료
5. 상태 정보 요청



1. Model(모델) : 데이터 그 자체

2. View(뷰) : 유저인터페이스

3. Controller(컨트롤러) : 뷰와 모델의 연결(로직을 수행)

4. 라우터 : 주소에 따라 컨트롤러와 view를 선택

■ Model

1) 데이터 자체

2) backbone.js의 Model과 다름

3) 프레임워크 종속성이냐? 아니면 편리함이냐?

■ View

1) 유저 인터페이스

2) 모델의 정보가 출력되는 곳

3) 유저로부터 이벤트를 받는 곳

4) 선언적 프로그래밍을 해야 함

■ Controller

1) 이벤트의 감지 및 연결

2) 뷰의 이동(라우터 이용)

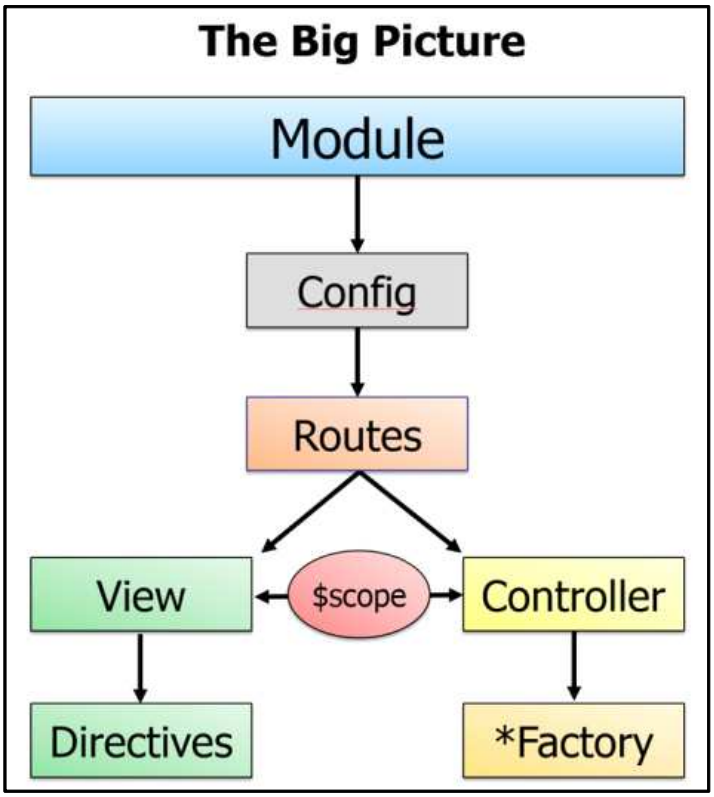
3) 서버와의 연결($http 이용)

4) 모델의 설정 및 초기화

■ 모듈의 이용

1) 전역상태로 모두 사용하면 유지보수나 코딩을 이해하는데 많은 어려움이 있음

2) 하나의 페이지에 모듈을 여러개로 나눠서 개발하면 편리함



```
<body data-ng-app="myApp">
	<div data-ng-controller="boxController">
		<h1>{{title}}</h1>

	</div>
	<script type="text/javascript" src="js/angular.min.js"></script>
	<script type="text/javascript">

		//{{}} <- 인터폴레이션

		var module = angular.module("myApp",[]);//[]필요한 모듈의 의존성 입
력

		module.controller("boxController",function($scope){
			$scope.title="모듈을 이용합니다.";
		});
	</script>
</body>
```

■ bind 지시어

지시어	설 명
ng-model	Model객체 생성(input,textarea,select)
ng-bind	모델을 바인딩({{}})와 같음
ng-show	true면 안보임 / false면 보임
ng-hide	true면 숨기고 / false면 안보임
ng-disabled	true면 disable / false면 반대
ng-readonly	true면 readonly / false면 반대
ng-checked	true면 checked / false면 반대
ng-multiple	true면 multiple속성 / false면 반대

```
<!DOCTYPE html>
<html lang="ko" data-ng-app>
<head>
	<meta charset="utf-8">
	<title>bind</title>
</head>
<body data-ng-init="txt='a'">
	<h1 data-ng-bind="txt">Test</h1>
	<input data-ng-model="txt" />
	<input type="checkbox" data-ng-model="chk"/>
	<h2  data-ng-show="chk">안녕하세요?</h2>
	<h2 data-ng-hide="chk">반갑습니다?</h2>
	<input data-ng-disabled="chk" value="못씀"/>
	<input data-ng-readonly="chk" value="읽기만 함"/>
	<script src="js/angular.min.js"></script>
</body>
</html>
```

■ filter의 사용

1) 데이터의 출력에 다양한 필터를 활용하거나 직접 filter를 만들 수 있음

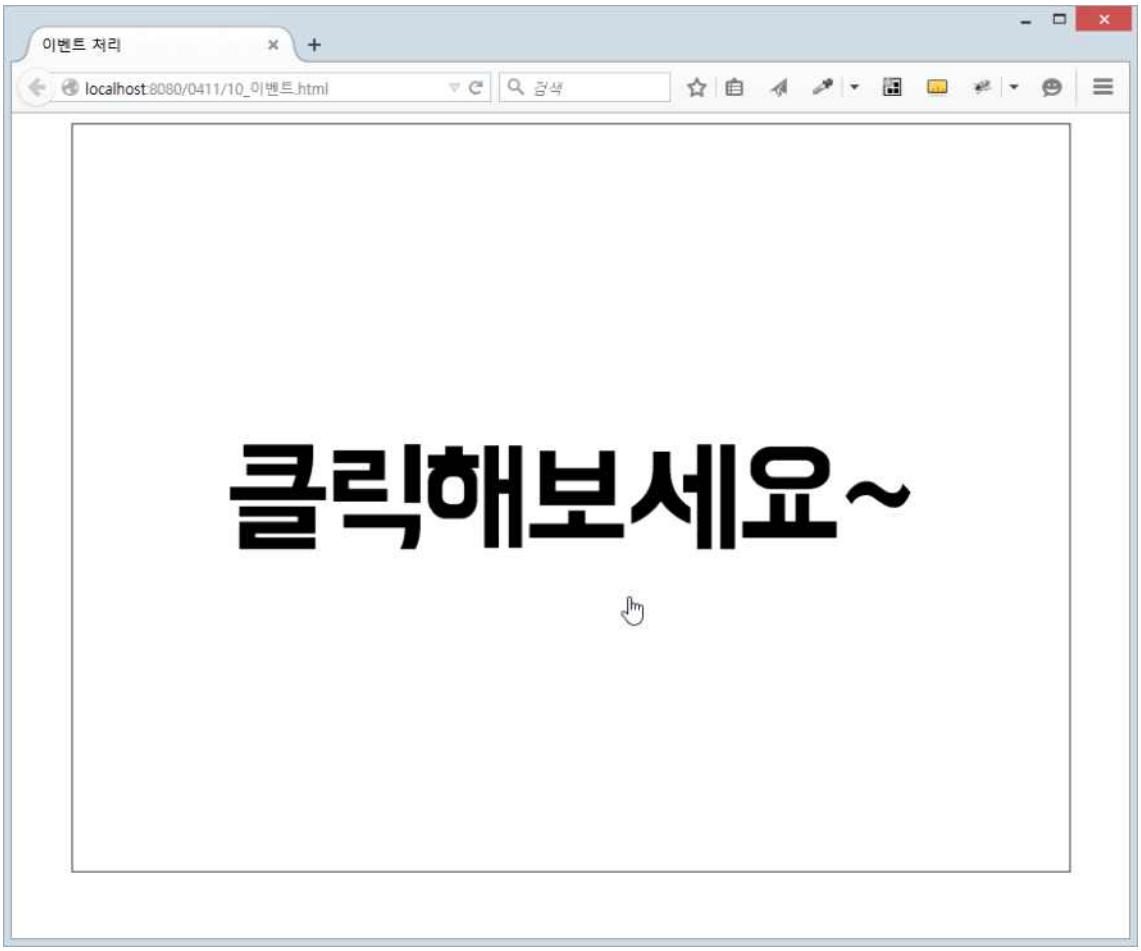
필터명	설 명
currency	통화기호와 숫자로 표현
date	날짜를 특정형식으로 표현함
number	소수점 자리수로 숫자를 표현
lowercase	소문자로 처리
uppercase	대문자로 처리
json	이 필터를 이용하여 json 디버깅이 편함

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>필터의 사용</title>
</head>
<body data-ng-app data-ng-controller="movieController">
<h1>영화검색</h1>
<p>
<label>이름으로 검색 : <input type="text" data-ng-model="search" /></label>
</p>
<ul >
<li data-ng-repeat="movie in movies ¦ filter:search">제목: {{movie.name}} 평점 :
{{getAvgGrade(movie)¦number:2}}</li>
</ul>
<p>
<h2>여자는 {{femaleNum}}명입니다.</h2>
<ul>
<li data-ng-repeat="person in people">{{person.name}}</li>
</ul>
<script type="text/javascript" src="js/angular.min.js"></script>
<script type="text/javascript">

function movieController($scope,$filter) {

       $scope.movies = [
                                          {poster:"Cinderella.jpg",director:"케네
스 브레너",name:"신데렐라",grade:13,gradeNum:2},
                                          {poster:"CityLights.jpg",director:"찰리
채플린",name:"시티라이트",grade:37,gradeNum:4},
                                          {poster:"EleanorRigby.jpg",director:"네
```

```
드 벤슨",name:"엘리노어 릭비",grade:51,gradeNum:6},
                                        {poster:"Fast_Furious7.jpg",director:"제
임스 완",name:"분노의 질주7",grade:68,gradeNum:11},
                                        {poster:"Kingsman.jpg",director:"매튜 본
",name:"킹스맨",grade:0,gradeNum:0},
                                    ];

      $scope.getAvgGrade=function(movie) {

            if(movie.gradeNum!=0) {
                  return movie.grade/movie.gradeNum;
            }
            return movie.gradeNum;
      }



      $scope.people= [
                              {name:"김연아",gender:"여자"},
                              {name:"류현진",gender:"남자"},
                              {name:"박지성",gender:"남자"},
                              {name:"수지",gender:"여자"}
                              ];

      $scope.femaleNum      =      $filter('filter')($scope.people,{gender:"여자
"}).length;

}

</script>
</body>
</html>
```

■ 이벤트 지시어

지시어	설 명
ng-change	change이벤트
ng-click	click이벤트
ng-dbclick	더블클릭이벤트
ng-mousedown	마우스누름
ng-submit	form에서 submit
ng-keydown	키 누름
그 외	우리가 아는 이벤트에 ng-붙이기




```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
```

```
<title>이벤트 처리</title>
<link rel="stylesheet"
href="http://fonts.googleapis.com/earlyaccess/hanna.css" />
<style>
	#clickBox {
		font-family: 'Hanna', serif;
		font-size:100px;
		width:800px;
		height:600px;
		text-align:center;
		line-height:600px;
		border:1px solid #333;
		margin:auto;
		cursor: pointer;
	}
</style>
</head>
<body data-ng-app="eventApp" data-ng-controller="eventController" >
	<div id="clickBox" data-ng-click="clickBox($event)">
		<span data-ng-bind="msg"></span>
	</div>
	<script type="text/javascript" src="js/angular.min.js"></script>
	<script type="text/javascript">
		var module = angular.module("eventApp",[]);

		module.controller("eventController",function($scope){

			$scope.msg="클릭해보세요~";

			$scope.clickBox = function($event) {
				$scope.msg = "x:" + $event.clientX+"/
```

```
y:"+$event.clientY;
                    }


            });


      </script>
</body>
</html>
```

■ 자바스크립트 배열 공부하기

1) 배열이 가진 메서드

메서드명	설 명
push(데이터)	배열에 새 요소를 추가하고 배열의 새 길이를 반환합니다.
pop()	배열의 마지막 요소를 제거하여 반환합니다.
unshift(데이터)	배열 시작에 새 요소를 삽입합니다.
shift()	배열에서 첫 번째 요소를 제거하여 반환합니다.
splice(index,갯수)	index부터 갯수만큼 제거

2) 예제

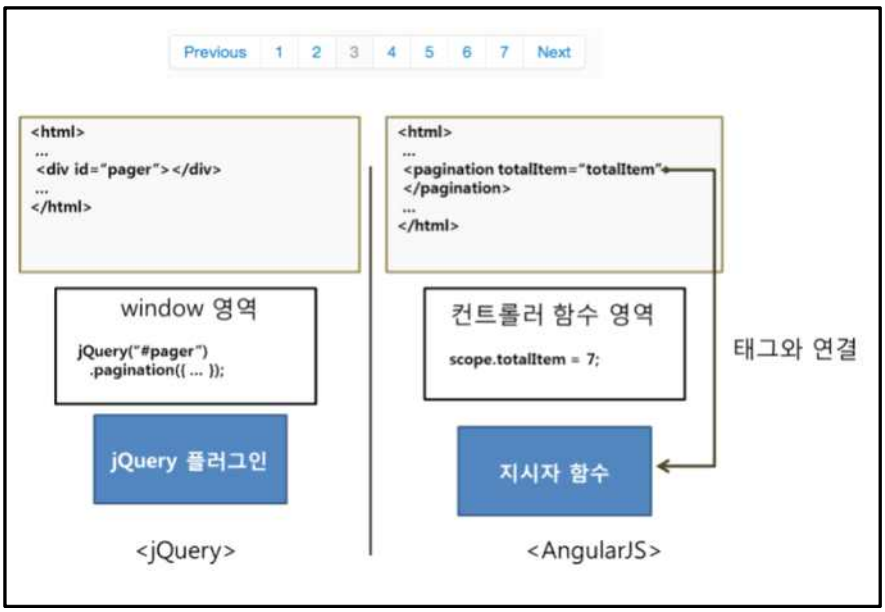
```
<!DOCTYPE html>
<html lang="ko">
<head>
<meta charset="utf-8">
<title>array</title>
</head>
<body>
<script>
      //var arr = new Array("김연아","손흥민","류현진");
      //alert(arr.length);
      var arr = ["김연아","손흥민","류현진"];
```

```
        arr.push("박지성");
        //alert(arr);//김연아,손흥민,류현진,박지성
        arr.unshift("박태환");
        //alert(arr);//박태환,김연아,손흥민,류현진,박지성
        arr.splice(2,1);//2번째index의 손흥민 제거
        //alert(arr);//박태환,김연아,류현진,박지성
        arr.pop();//박태환,김연아,류현진
        arr.shift();//김연아,류현진
        arr.shift();//류현진
        alert(arr);
</script>
</body>
</html>
```

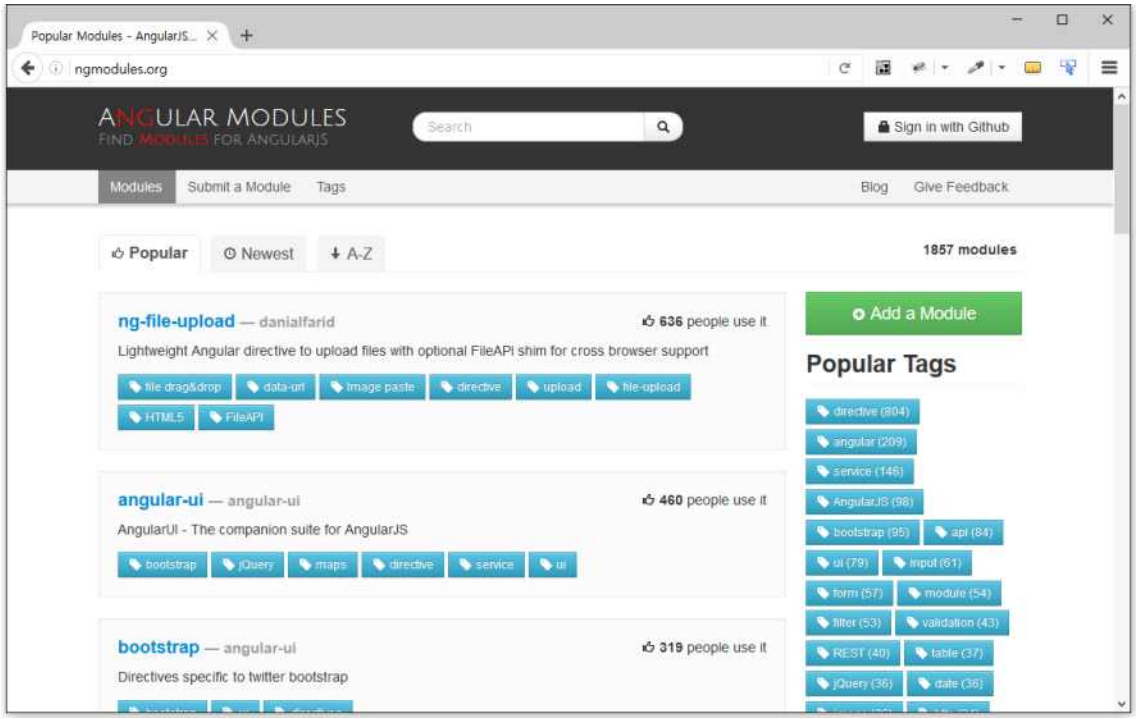
■ directive 라이브러리의 이용

1) datepicker같은 HTML이 지원하지 않는 기능을 확장하는 방식을 지시자로 제공

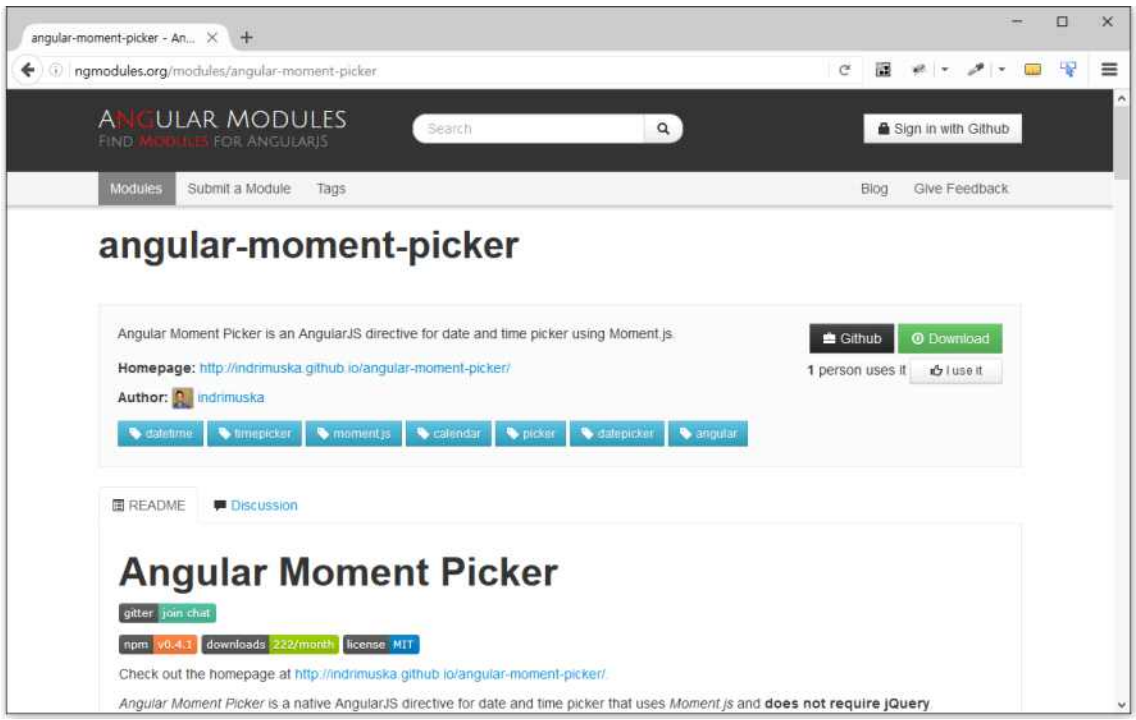
- jquery의 플러그인과 angular.js의 directive의 차이



- http://ngmodules.org/



- datepicker예제



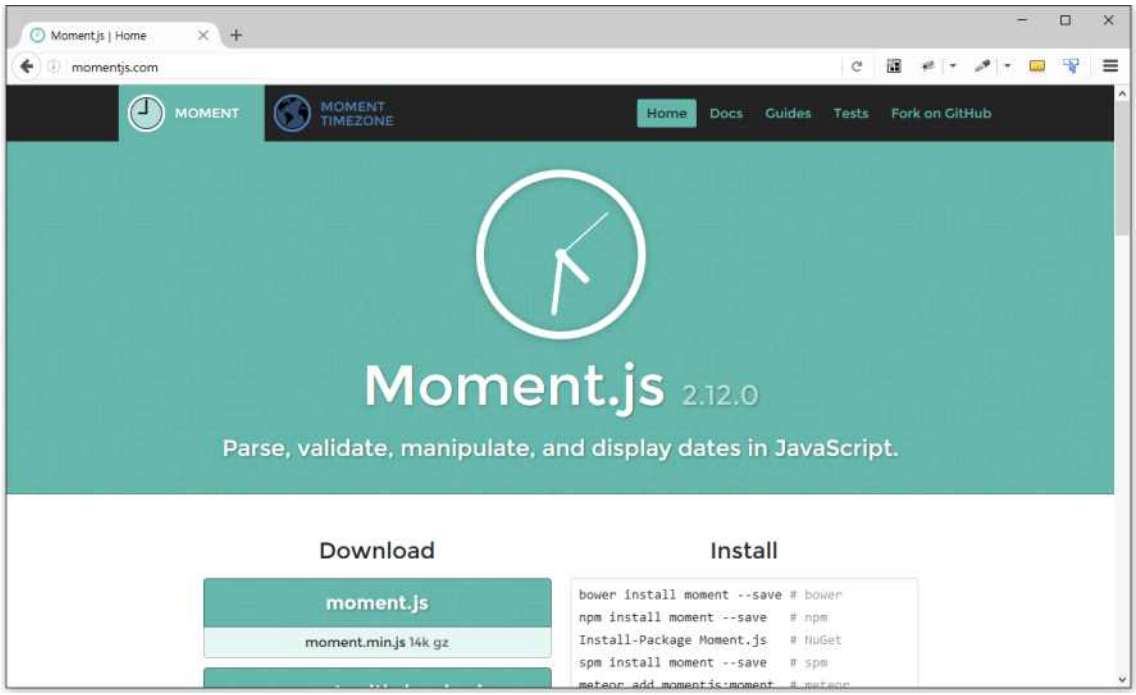
- moment.js

1) 날짜관련 javascript 내장 객체

```
<script type="text/javascript">
	var now = new Date();
	var year = now.getFullYear();
```

```
        var month = now.getMonth();
        var date = now.getDate();
        var hour = now.getHours();
        var min = now.getMinutes();
        var second = now.getSeconds();


        var old = new Date(2014, 1, 23);
        alert(old);
        var time = old.getTime();
        alert(time);
        alert(now - old);
</script>
```

2) moment.js




```
<!DOCTYPE html>
<html lang="ko">
<head>
<meta charset="utf-8">
<title>Date_moment</title>
<style>
```

```
	#time {
		margin:70px;
		text-align:center;
		font-size:60px;
		font-weight:bold;
	}
</style>
</head>
<body>
<div id="time">2015-4-18 12:30:54</div>
<script src="js/moment-with-locales.min.js"></script>
<script>

	//moment 한글
	moment.locale("ko");

	//특정시간을 moment객체로 만드는 방법
	//alert(moment().format('llll'));

	//UNIX타임(세컨드) : 1970년 1월 1일 0시 0분 0초부터 초

	var start = moment().unix();//초단위

	console.log("초단위" + start);

	var startMill = moment().valueOf();

	console.log("밀리초단위 : " + startMill);

	setInterval(function(){
```

```
        //처음에 스타트한 시간
        //moment(밀리세컨드)

        //var startMoment = moment(startMill);

        var startMoment = moment(start,"X");

        //현재시간까지 몇XXX 지났는지 확인
        //console.log(startMoment.fromNow());

        //현재 시간
        var now = moment();

        console.log(now.diff(startMoment,"seconds"));



        //var year = now.year();
        //var month = now.month();

        var fullTime =
        now.format("YYYY년 MM월 DD일 HH시mm분ss초.SSS");
        $("#time").text(fullTime);

//      var now = new Date();
//  var year = now.getFullYear();
//  var month = now.getMonth();
//      var date = now.getDate();
//      var hour = now.getHours();
//      var min = now.getMinutes();
//      var second = now.getSeconds();
```

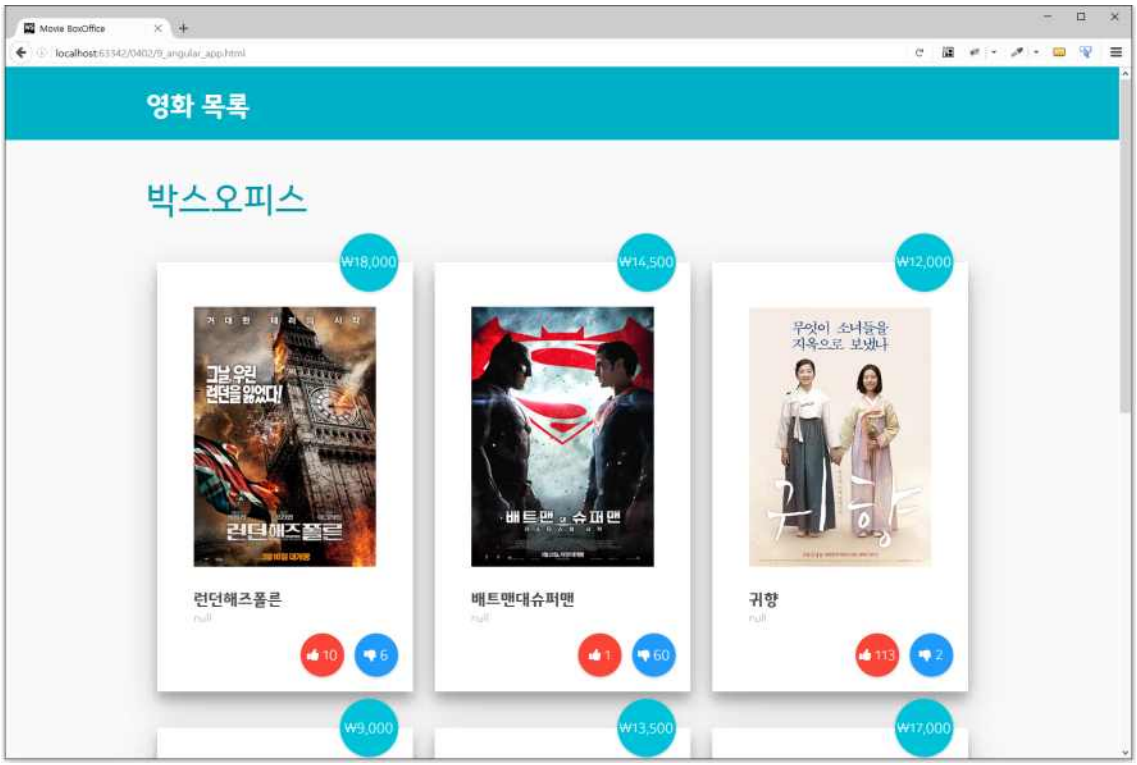
```
      },1);


</script>
</body>
</html>
```

- moment-datepicker예제코드

```
<!doctype html>
<html ng-app="pickerApp">
<head>
    <meta charset="UTF-8">
    <title>날짜 선택</title>
    <link rel="stylesheet"
          href="css/angular-moment-picker.css"/>
</head>
<body>
   <!-- 커스텀 디렉티브 -->
    <input class="form-control"
       ng-model="ctrl.input"
       format="YYYY-MM-DD"
       ng-model-options="{ updateOn: 'blur' }"
       placeholder="날짜를 선택하세요."
       moment-picker="ctrl.input">

    <script src="js/angular.min.js"></script>
    <script src="js/moment-with-locales.js"></script>
    <script src="js/angular-moment-picker.js"></script>
    <script>
var app = angular.module("pickerApp",["moment-picker"]);

    //설정
      app.config(["momentPickerProvider",
                  function(mpp){
           mpp.options({
```

```
                locale:"ko",
                minView:'month',
                maxView:'month',
                startView:'month'
            });

                  }]);


    </script>
</body>
</html>
```

■ 영화리스트 예제

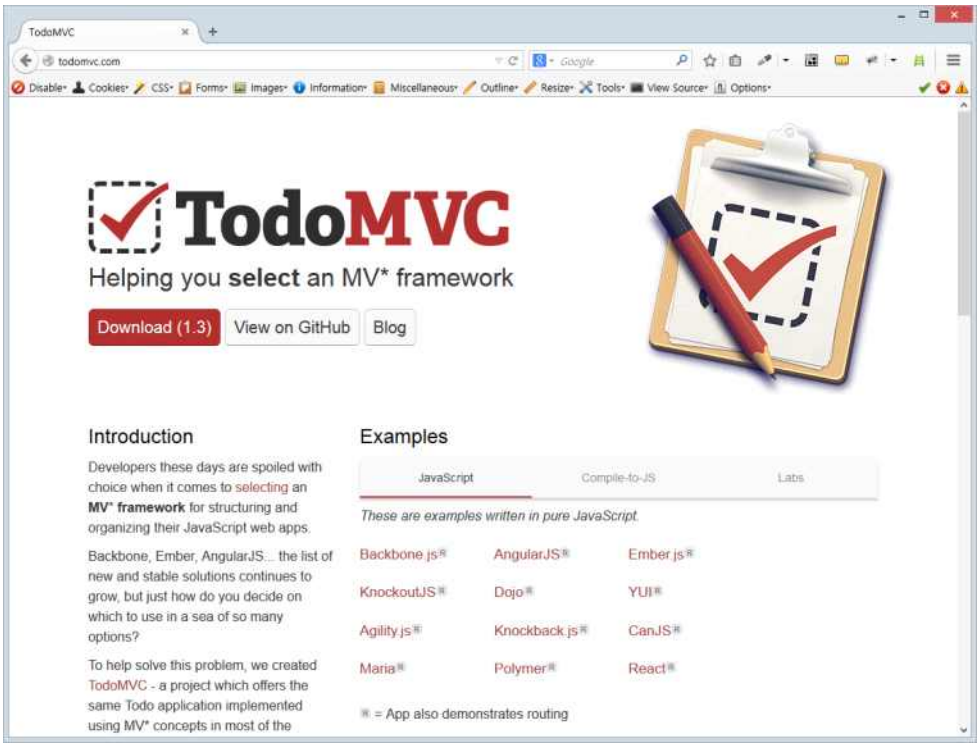



```
<!doctype html>
<html ng-app="movieApp">
  <head>
      <meta charset="UTF-8" />
      <title>Movie BoxOffice</title>
    <link href="css/bootstrap.min.css"
    rel="stylesheet" />
    <link href="css/main.css" rel="stylesheet" />
    <link rel="stylesheet" href="css/kakao.font.css" />
    <link rel="stylesheet" href="css/font-awesome.min.css" />
  </head>
  <body>
    <div class="header">
      <div class="container">
        <h1>영화 목록</h1>
      </div>
    </div>
    <div class="main" ng-controller="mainCtrl">
      <div class="container">
        <h2>제목</h2>
  <ul>
```

```
      <li class="col-md-4" ng-repeat="movie in movies">
        <div class="thumbnail">
          <img src="poster/{{movie.poster}}"/>
          <p class="price" ng-bind="movie.price|currency:undefined:0"></p>
          <p class="title" ng-bind="movie.title"></p>
          <p class="date">{{movie.release|date:"yyyy년 M월 d일"}}</p>
          <div class="rating">
            <p     class="likes"     ng-click="upLikes(movie)"><i     class="fa
fa-thumbs-up"></i> {{movie.likes}}</p>
            <p class="dislikes"
               ng-click="upDislikes(movies,$index)"><i                 class="fa
fa-thumbs-down"></i> {{movie.dislikes}}</p>
          </div>
        </div>
      </li>
  </ul>
      </div>
    </div><!-- movieController -->
    <div class="footer">
      <div class="container">
        <h2>&copy; 2016 ani</h2>
      </div>
    </div>
<script src="js/angular.min.js"></script>
<script src="js/angular-locale_ko.js"></script>
<script>


    // 생성자 함수
    function Movie(poster,title,price,release,likes,dislikes){
        this.poster = poster;
        this.title = title;
        this.price = price;
        this.release = release;
        this.likes = likes;
        this.dislikes = dislikes;
    }

var movies = [
```

```
new Movie("p1.png","런던해즈폴른",18000,new Date("2016-3-3"),10,6),
new Movie("p2.png","배트맨대슈퍼맨",14500,new Date("2016-3-24"),1,60),
new Movie("p3.png","귀향",12000,new Date("2016-2-15"),113,2),
new Movie("p4.png","널기다리며",9000,new Date("2016-3-16"),23,8),
new Movie("p5.png","동주",13500,new Date("2016-2-12"),7,6),
new Movie("p6.png","주토피아",17000,new Date("2016-2-14"),111,0)
];



    var app = angular.module("movieApp",[]);

    app.controller("mainCtrl",["$scope",function($scope){
        //모델로 등록
        $scope.movies = movies;

        $scope.upLikes = function(movie) {

            movie.likes++;
            //alert("dfasdfasdfa");

        }//upLikes function() end

        $scope.upDislikes = function(movies,$index) {

            movies[$index].dislikes++;

        }//upDislikes function() end

    }]);



</script>
  </body>
</html>
```

■ angular Todo 예제



http://todomvc.com/

1) todo목록 애플리케이션 제작

```
<!DOCTYPE html>
<html lang="ko">
 <head>
  <meta charset="UTF-8">
  <title>Todo App만들기</title>
 </head>
 <body>
       <div>
             <h1>제목</h1>
             <p>전체 할일 <em>2</em>개 / 남은 할일은 <strong>1</strong>개</p>
             <ul>
                   <li  class="to-do-list"><label><input  type="checkbox"  />독서
</label></li>
```

```
                    <li  class="to-do-list"><label><input  type="checkbox"  />공부
</label></li>
            </ul>
            <form>
                  <fieldset>
                        <legend>할일 입력폼</legend>
                        <div>
                              <input id="txt" placeholder="새로운 할일 입력" />
                              <button>추가</button>
                        </div>
                  </fieldset>
            </form>
      </div>
 </body>
</html>
```

2) data-ng-app 속성추가

```
<html lang="ko" data-ng-app>
```

3) controller 만들기

```
<div data-ng-controller="controller">
```

```
function controller($scope) {
      //alert("test");
}
```

4) 타이틀 동적으로 부여

```
<h1>{{title}}</h1>
```

```
function controller($scope) {
	$scope.title = '필구의 Todo!';
}
```

5) todo리스트를 json으로 만들기

```
$scope.todoList = [
	{ end : true, title : "독서"},
	{ end : false, title : "달팽이 구워먹기"},
	{ end : false, title : "프로젝트 완성"}
	];
}
```

6) 리스트 출력

● ng-repeat은 for문과 같음(jstl의 forEach와 같음)
● ng-model은 해당요소에 모델을 바인딩시키는 지시어
(ng-model은 input,textarea,select요소에만 가능)
● {{}}는 표현식

```
<li data-ng-repeat="todo in todoList">
	<label><input type="checkbox"
data-ng-model="todo.end"/>{{todo.title}}</label>
</li>
```

7) 새로운 할일 입력

● ng-ng-submit은 submit이벤트리스너
● $event객체와 / newDo MODEL을 DI(의존성 주입)

```
<form data-ng-submit="addTodo($event,newDo)">
<fieldset>
	<legend>할일 입력폼</legend>
```

```
	<div>
		<input id="txt" placeholder="새로운 할일 입력" data-ng-model="newDo"
/>
		<button>추가</button>
	</div>
</fieldset>
</form>
```

● $event.preventDefault()는 submit막기

● newDo 내용이 없다면 아무일도 안하기

● 내용이 있다면 todoList에 할일 객체 바인딩

● $scope.newDo를 비우기

● input에 focus()주기

```
$scope.addTodo=function($event,newDo) {
	$event.preventDefault();
	if(newDo.length==0) return false;
	$scope.todoList.push({end:false,title:newDo});
	$scope.newDo = ""; //newDo="";와는 완전히 다름
	angular.element("#txt").focus();
};
```

8) 전체 할일 갯수와 남은 할일 갯수 구하기

● data-ng-bind를 이용해 todoList 모델의 length 바인딩

● 남은 할일은 처리를 해야 하므로 remainDo()함수와 바인딩

```
<p>
전체 할일 <em data-ng-bind="todoList.length"></em>개
/ 남은 할일은 <strong data-ng-bind="remainDo(todoList)"></strong>개
</p>
```

● todoList를 DI

● angular.forEach메서드를 이용해 for each 수행

● 각 todo모델의 end속성이 false인 갯수를 리턴

```
$scope.remainDo = function(todoList) {
	var cnt = 0;
	angular.forEach(todoList, function(todo, index) {
		if(!todo.end) {
			cnt++;
		}
	});
	return cnt;
};
```

9) 완료된 할일 목록 비우기

● clearEnd(todoList) 이벤트리스너 붙이기

```
<button data-ng-click="clearEnd(todoList)">완료없애기</button>
```

● 기존의 todoList를 받고, 새로운 배열을 만들어 완료되지 않은 todo모델만 push

```
$scope.clearEnd=function(todoList) {
    var oldList = todoList;
    $scope.todoList = [];
	angular.forEach(oldList, function(value, key) {
		if(!value.end) {
			$scope.todoList.push(value);
		}
	});
};
```

10) 할일 개별삭제 버튼 만들기

● delete($index) 에서 $index는 요소의 index

```
<li data-ng-repeat="todo in todoList">
	<label><input type="checkbox" data-ng-model="todo.end"/>
	{{todo.title}}
	</label>
```

```
      <button data-ng-click="delete($index)">삭제</button>
</li>
```

todoList모델에서 해당 모델 삭제

(양방향 바인딩이 되어 있기 때문에 자동으로 삭제됨)

```
$scope.delete = function($index) {
      $scope.todoList.splice($index,1);
};
```

■ 전체 소스

```
<!doctype html>
<html ng-app="todoApp">
<head>
    <meta charset="UTF-8">
    <title>할일관리프로그램</title>
    <link rel="stylesheet" href="css/kakao.font.css"/>
    <link rel="stylesheet"
          href="css/angular-switcher.css"/>
    <style>
        body {
            font:200 20px "Kakao",sans-serif;
        }
        #wrap {
            width:800px;
            margin:auto;
            box-shadow: 0 12px 15px 0 rgba(0, 0, 0, 0.24), 0 17px 50px 0 rgba(0,
0, 0, 0.19);
            padding:20px;
        }
        input,button {
            padding: 5px;
            font:200 20px "Kakao",sans-serif;
```

```
        }
        button {
            background:#9E9E9E;
            border:none;
            cursor: pointer;
            padding:8px 12px;
            color:#fff;
            transition: .2s ease;
        }
        button:hover {
            background: #424242;
            box-shadow: 0 12px 15px 0 rgba(0, 0, 0, 0.24), 0 17px 50px 0 rgba(0,
0, 0, 0.19);
        }
        /* bar height */
        .styled .switcher-line:before { height: 20px; }
        /* bar backgrounds */
        .styled .switcher-line:before { background: #e35144; }
        .styled .active .switcher-line:before { background: deepskyblue; }
        ul {
            padding:0;
        }
        li {
            color:crimson;
            list-style: none;
        }
        li.end {
            color:deepskyblue;
        }

        li.ng-animate {
```

```
            /*애니메이션될 요소의 기본 css */
            transition: .5s ease;
        }
        li.ng-enter {
            transform: translate(-2000px,0);
        }
        li.ng-enter.ng-enter-active {
            transform: translate(0,0);
        }
        li.ng-leave {
            transform: translate(0,0);
        }
        li.ng-leave.ng-leave-active {
            transform: translate(-2000px,0);
        }

    </style>
</head>
<body ng-controller="todoCtrl">
<div id="wrap">
<h1>{{user}}님의 할일목록</h1>
<p>
    전체할일 : <em ng-bind="todos.length"></em>개
    /
    미완료 : <span>{{checkIncomplete(todos)}}</span>개
    <button ng-click="removeAllComplete(todos)">완료된 할일제거</button>
</p>
<form ng-submit="writeTodo(todos,title)">
<fieldset>
    <legend>할일입력폼</legend>
    <input type="text" ng-model="title"
```

```
           placeholder="할 일입력"/>
    <button>입력</button>
</fieldset>
</form>
<h2>할일목록</h2>
<ul>
<li ng-class="{end:todo.end}"
    ng-repeat="todo in todos">
<switcher class="styled"
          true-label=""
          false-label=""
          ng-model="todo.end">
</switcher>
   {{todo.title}} <button ng-click="removeTodo(todos,$index)">삭제</button>
</li>
</ul>
</div>
<script src="js/angular.min.js"></script>
<script src="js/angular-switcher.js"></script>
<script src="js/angular-animate.min.js"></script>
<script>
var app =
    angular.module("todoApp",["switcher","ngAnimate"]);

    function Todo(title,end) {
        this.title =title;
        this.end = end||false;
    }
```

```
    //alert(new Todo("UI시나리오작성").end);

app.controller("todoCtrl",["$scope",function($scope){

     $scope.user="김필구";

    $scope.removeAllComplete = function(todos) {

//        _.each(todos,function(todo,index)) {
//
//       })

//        $.each(todos,function(index) {
//           this
//        });

       //미완료된 할일만 넣는 임시배열
        var tmpArr = [];

        angular.forEach(todos,function(todo,index) {

            //완료된 todo일 경우
            if(!todo.end) {
                tmpArr.push(todo);
            }//if end
        });//forEach() end

        $scope.todos = tmpArr;

    }//removeAllComplete end
```

```
    $scope.removeTodo = function(todos,$index){

        todos.splice($index,1);

    }//removeTodo end

    $scope.writeTodo = function(todos,title) {

        todos.push(new Todo(title));

        $scope.title = "";

    }//writeTodo end

    $scope.checkIncomplete = function(todos) {

        //alert("ddd");
        var length = 0;

        for(var i = 0 ; i < todos.length;i++) {

            //todo의 end가 false일때 미완료이기 때문에
            if(!todos[i].end) {
                length++;
            }//if end

        }//for end
        return length;
    }//checkIncomplete() end

    $scope.todos = [
```

```
      new Todo("점심먹기"),
      new Todo("수업",true),
      new Todo("독서",true),
      new Todo("친구와 약속"),
      new Todo("은행가기")
    ];

}]);
</script>
</body>
</html>
```

■ route 구현

1) 웹 초창기에는 url은 단 하나의 물리적인 리소스에 접근하는 데 사용되었음

2) 동적으로 렌더링되는 (Ajax를 이용하여) 페이지는 불가능함

3) ajax를 이용하여 동적으로 렌더링되는 페이지는 F5를 눌러 새로고침하면 지금 하고 있던 작업이 사라지거나 초기화됨

4) 브라우저의 뒤로가기, 앞으로가기, URL 복사 기능이 가능하게 하기 위해서 hashbang 방식의 URL이 사용되었음

- 기존 페이지

```
example.com/movie/list : 영화 목록을 보여주는 URL
example.com/movie/34 : 34번의 영화를 보여주는 URL
```

- hashbang 방식

```
example.com/#/movie/list : 영화 목록을 보여주는 URL
example.com/#/movie/34 : 34번의 영화를 보여주는 URL
```

#이 들어가면 주소창의 URL이 변경되도 페이지를 다시 로딩하지 않음

(페이지 내부 링크이기 때문에)

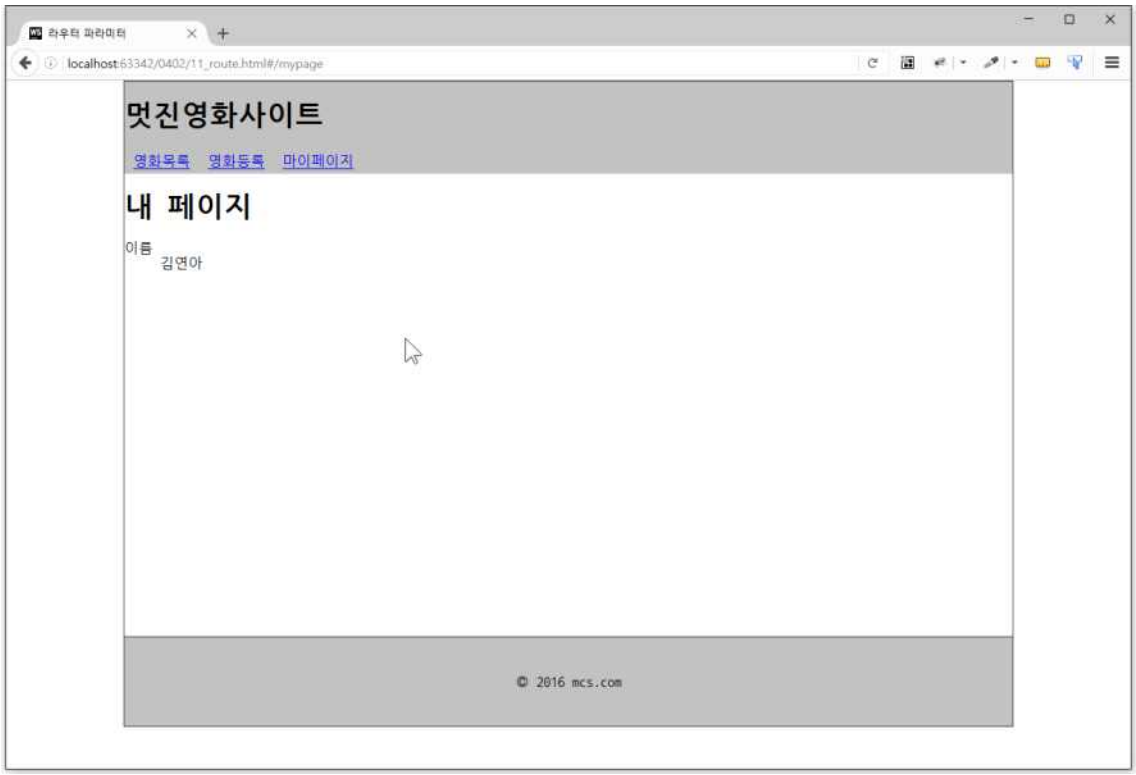
5) angular.js는 이를 활용하기 위해 $route 서비스가 존재함

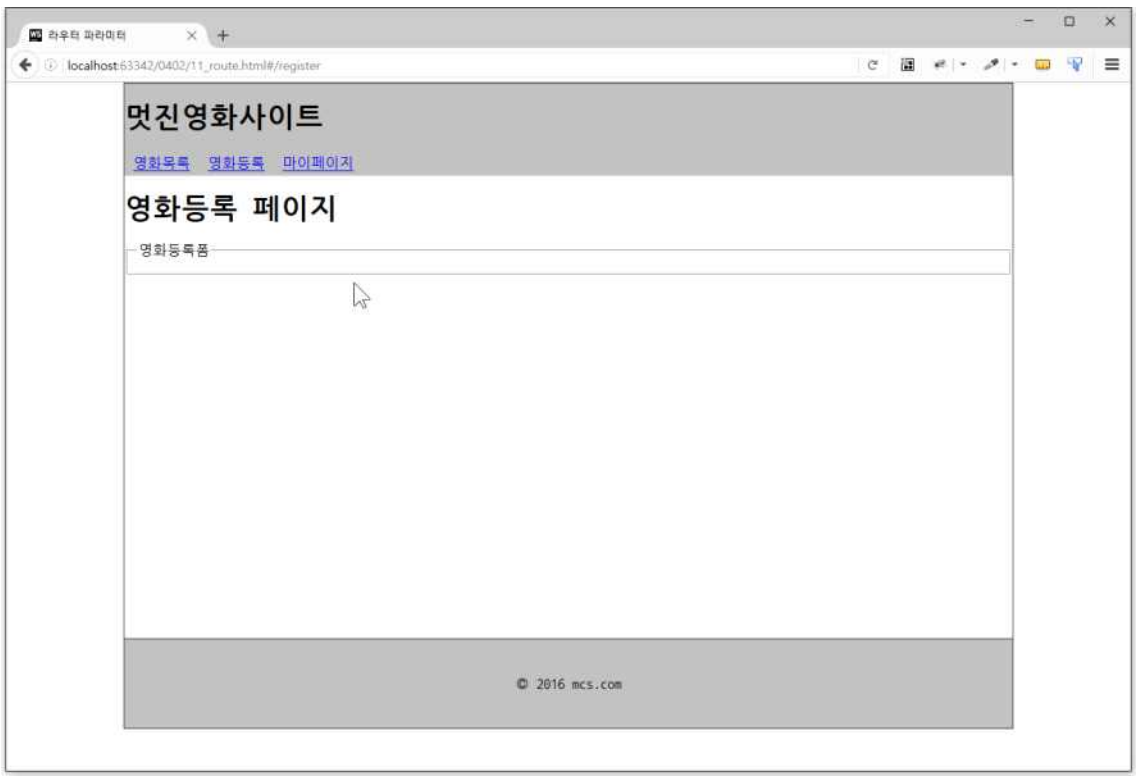
6) angular-route.min.js 추가

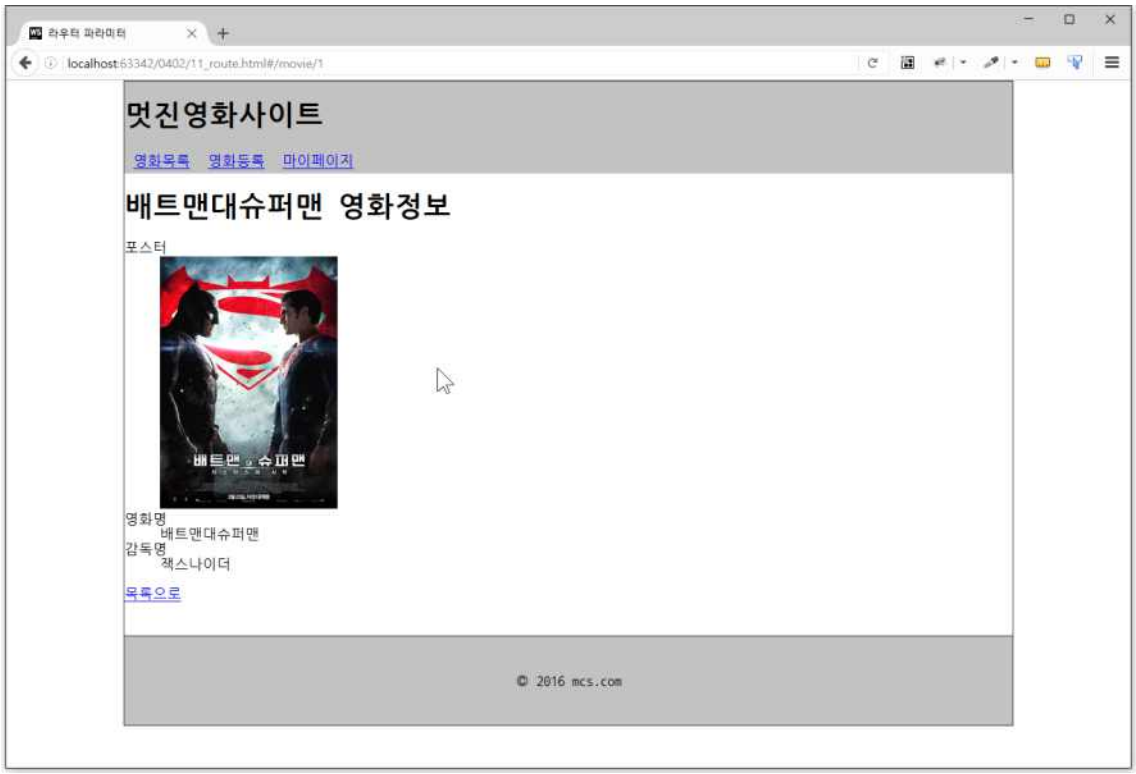
7) ngRoute 모듈에서 제공하는 서비스와 지시자

내용	설 명
지시자	ngView
서비스	$route, $routeParam

■ 예제








```
<!DOCTYPE html>
<html data-ng-app="routeApp">
<head>
<meta charset="UTF-8">
<title>라우터 파라미터</title>
```

```
<style>
	body {
	margin:0;}
	#wrap {
	width:1000px;margin:auto;border:1px solid #333;}

	#header {
	width:1000px;
	background:#BDBDBD;
	border-top:1px solid #aaa;
	}

	#header ul {
		margin:0;
		padding:0;
		list-style: none;
		overflow: hidden;
	}

	#header li {
		float:left;
		padding:5px 10px;
	}
	#content {
		min-height:500px;
	}
	#footer {
	height:100px;
	line-height:100px;
	text-align:center;
	 background: #BDBDBD;
```

```
	border-top:1px solid #333}


</style>
</head>
<body>
	<div id="wrap">
		<div id="header">
			<h1>멋진영화사이트</h1>
			<ul>
				<li><a href="#list">영화목록</a>
				<li><a href="#register">영화등록</a>
				<li><a href="#mypage">마이페이지</a>
			</ul>
		</div>
		<div id="content">
			<!-- 변경될 view -->
			<ng-view></ng-view>
		</div>
		<div id="footer">&copy; 2016 mcs.com</div>
	</div>
<script src="js/angular.min.js"></script>
<script src="js/angular-locale_ko.js"></script>
<script src="js/angular-route.min.js"></script>
<script>

var app = angular.module('routeApp', ['ngRoute']);

app.config(function($routeProvider){

	$routeProvider.when("/list",{
		templateUrl:"view/list.js",
```

```
            controller:"listController"
      });
   $routeProvider.when("/register",{
            templateUrl:"view/register.js",
            controller:"registerController"
      });
   $routeProvider.when("/mypage",{
            templateUrl:"view/mypage.js",
            controller:"mypageController"
      });

      $routeProvider.when("/movie/:movieId",{
            templateUrl:"view/movie.js",
            controller:"movieController"
      });

      //기본 메인페이지
      $routeProvider.otherwise({redirectTo:'/list'});



});

//service, factory는 객체주입

//value : 순수한 데이터 주입
app.value("$movies",[
{no:0,"poster":"p1.png","director":"바박            나자피","name":"런던해즈폴른
","grade":106,"gradeNum":12},
{no:1,"poster":"p2.png","director":"잭스나이더","name":"배트맨대슈퍼맨
","grade":106,"gradeNum":12},
{no:2,"poster":"p3.png","director":"조정래","name":"귀향
```

```
","grade":106,"gradeNum":12},
{no:3,"poster":"p4.png","director":"모홍진","name":"널기다리며
","grade":106,"gradeNum":12},
{no:4,"poster":"p5.png","director":"이준익","name":"동주
","grade":106,"gradeNum":12},
{no:5,"poster":"p6.png","director":"바이론                    하워드","name":"주토피아
","grade":106,"gradeNum":12}
]);

app.controller('listController',
       ['$scope','$movies', function($scope,$movies){
              $scope.title = "핫 무비";
              $scope.movies = $movies;



}]);
    app.controller('registerController',
       ['$scope', function($scope){
              $scope.title = "핫 무비";
              $scope.movies = $movies;




}]);
    app.controller('mypageController',
       ['$scope', function($scope){



}]);

app.controller('movieController',
```

```
	['$scope','$movies', '$routeParams',
	function($scope,$movies,$routeParams){
	// alert("test");
		//movie/1
		//movie/2 .....
		//등 파라미터로 넘어온 정보를
		//받아서 처리하기 위해서
		// $routeParams객체가 있습니다.

		//alert($routeParams.movieId);

	$scope.movie = $movies[$routeParams.movieId];

}]);

</script>
</body>
</html>
```

list.view

```
<h1>{{title}}목록</h1>
<ul>
	<li ng-repeat="movie in movies">
	<a href="#movie/{{movie.no}}">{{movie.name}}</a>
	</li>
</ul>
```

movie.view

```
<h1>{{movie.name}} 영화정보</h1>
<dl>
	<dt>포스터</dt>
	<dd>
	<img src="poster/{{movie.poster}}" width="200" />
	</dd>
	<dt>
```

```
            영화명
        </dt>
        <dd>
            {{movie.name}}
        </dd>
        <dt>감독명</dt>
        <dd>{{movie.director}}</dd>


</dl>
<a href="#list">목록으로</a>
```

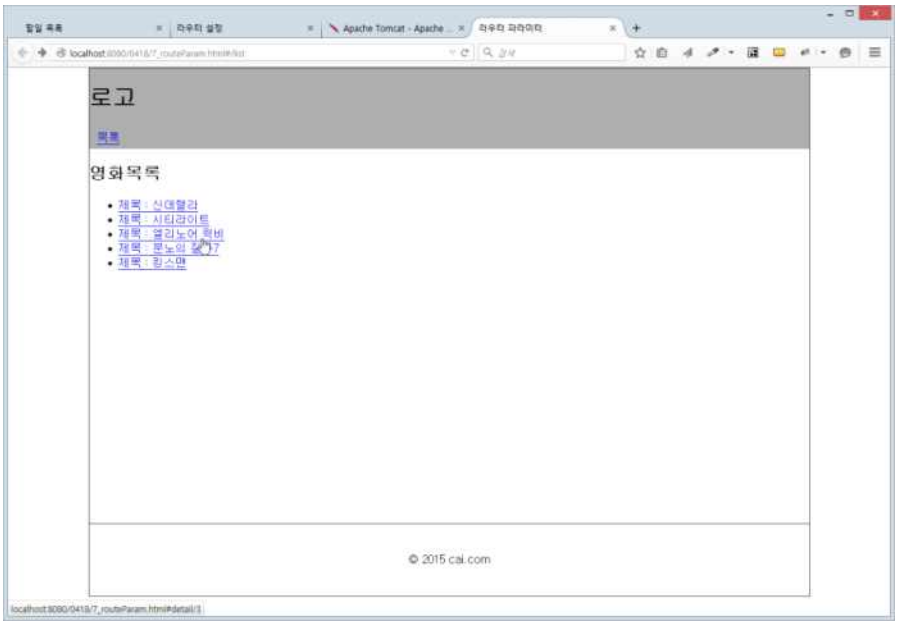
register.view

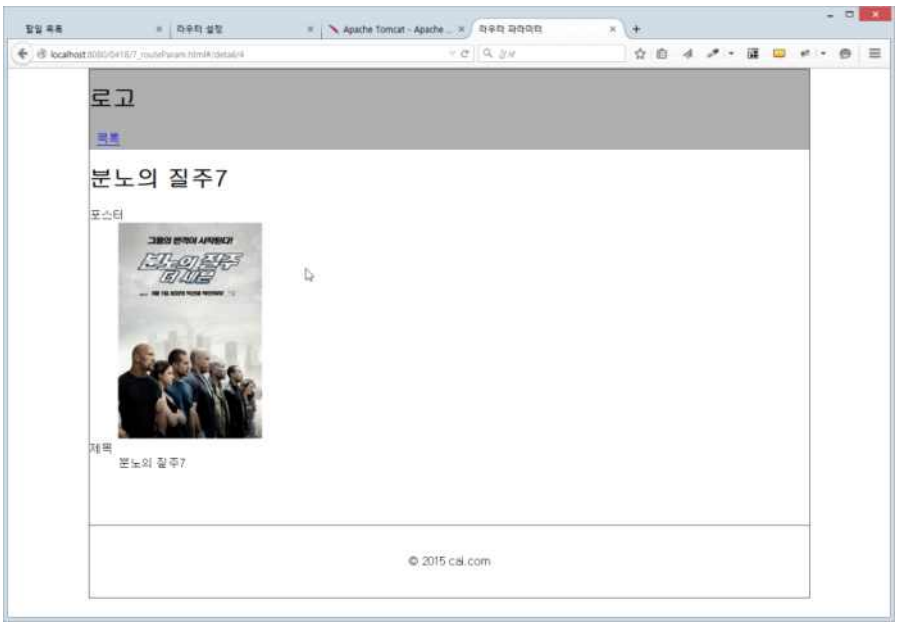
```
<h1>영화등록 페이지</h1>
<form>
    <fieldset>
        <legend>영화등록폼</legend>

    </fieldset>
</form>
```

mypage.view

```
<h1>내 페이지</h1>
<dl>
    <dt>이름</dt>
    <dd>김연아</dd>
</dl>
```

■ $routeParams의 이용





1) 넘어가는 파라미터를 얻어와서 처리할 수 있음

2) url에 :이름으로 설정 가능

```
'/detail/:movieId' : 설정
사용 : $routeParams.movieId
```