Angular 공부해보자

일단 초기 설정? generator 이용해보자.

김종엽

1. Angular 설치부터 하자

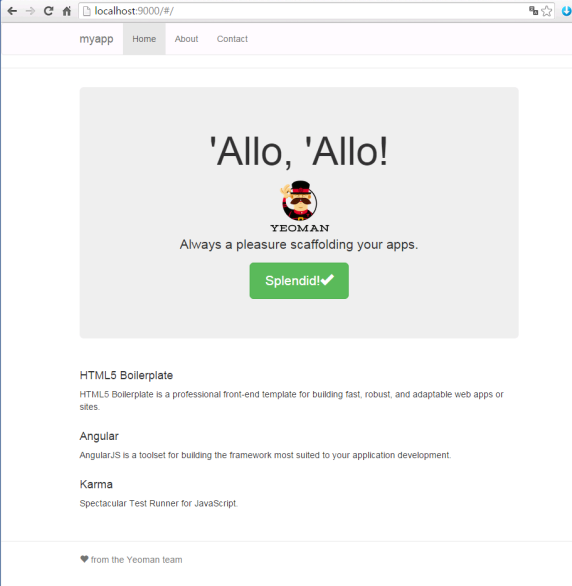
* 일단 Node.js, npm, git 필수 설치
* yeoman툴셋 설치하자 (yo, grunt-cli, bower, generator-angular, generator-karma 다설치
* npm install -g grunt-cli bower yo generator-karma generator-angular
* 버전 관리도 확실히 하자
* yo --version && bower --version && grunt --version

1. Angular workspace만들고 앱이름 만들고 실행해보자

* 난 test1 폴더에 만들거다 app 이름도 test1으로 하자
* yo angular test1
* grunt사용, compass안씀, bootstrap은 추가.
* 그담 체크하는데서 angular-router 제외하고 다 뺐다. 이것만 써보자!!
* grunt serve
* 실행했는데 실패한다!!!!

1. bower 설치가 안됐단다!!!!

* bower install 명령 이용해서 설치… 안된다!!
* git이 설치 안되어있단다… 어째서지? 아하 경로!!!
* 웃긴건 git 설치했지만 경로 설정 안되어있다…
* 컴퓨터 속성 > 고급 시스템 설정 > 환경 변수 > 시스템 변수 > path 편집 > C:\Program Files\Git\bin;C:\Program Files\Git\cmd 추가 (후우… 멀다 멀어)
* 이번엔 다시 bower install 설치 된다!!
* 다시 grunt serve, 기본적으로 localhost:9000 이다. 참고!!
* 성공!! 아래 화면이 나오면 성공이다!! > 기본 틀을 다 완성시켜준다!!



Angular 개념도 잡아보자

기본 코드 및 개념부터 잡자

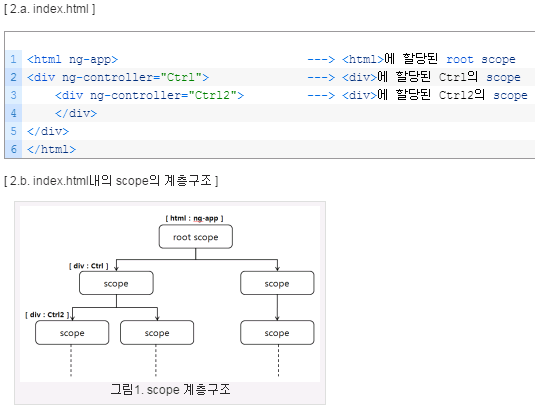
1. 배경 지식

* Must know: HTML/CSS, Javascript
* Not so important: jQuery, Ruby on Rails, Python, PHP, etc, Databases

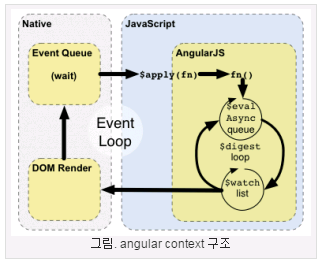
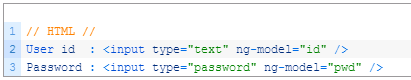
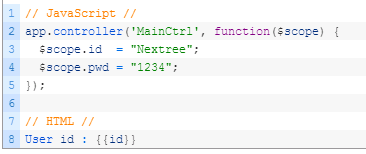
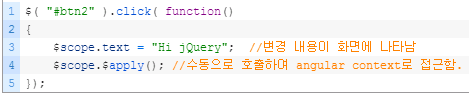
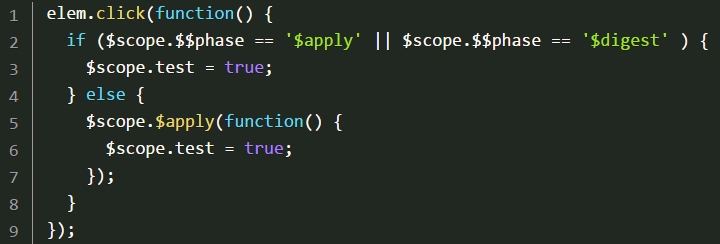
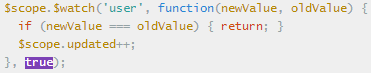
1. Angular 란?? 먼저 개념을 잡고 가자

* Angular는 MVC패턴 사용에 따른다.
* MVC란 Model, View, Controller를 완전히 분리하기 때문에 가시성이 높고 편리
* Module: 모듈은 Angular에서 최상위 레벨 컴포넌트, 게이트웨이 역할, 코드 및 컴포넌트를 정리하는데 도움을 준다.
* 모듈 추가는 뭐 가장 초기코드… var app = angular.module(‘myapp’, []);
* 여기에 컨트롤러를 추가하고, 핸들러와 데이터가 생기고, 그럼 scope가 등장하는데…

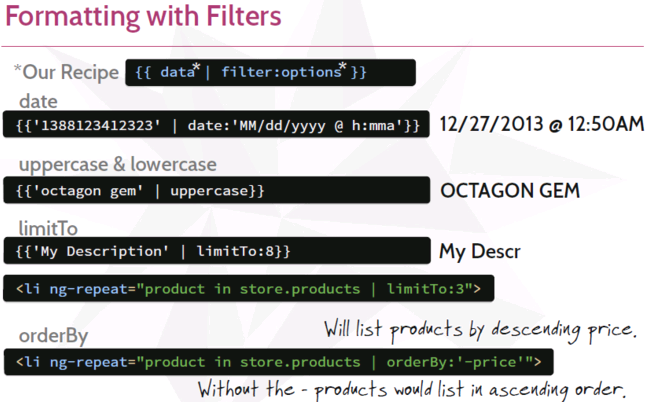
1. Scope 가 무엇이냐!

* Scope는 컨트롤러나 디렉티브의 유효범위내의 저장공간이라 할 수 있다. 이해 쉽게?
* View는 scope를 통해 컨트롤러 내부에 정의된 모델(데이터)나 핸들러 함수에 접근 가능.
* Angular에서 scope는 최상위 root scope와 여러 개의 child scope로 구성. 이게 기본.
* 즉 scope는 계층 구조라는 건데, 가장 최상위 root scope는 ng-app 디렉티브가 정의된 Dom에 할당.
* 즉 웹페이지 로드 후 Angular가 로드 될 때 root scope가 생성.
* 이후, root scope를 기준으로 컨트롤러나 디렉티브 내의 scope가 계층 구조를 이루어 생성되고 각 scope들은 해당 DOM에 할당.
* 
* DOM구조와 마찬가지로 scope도 동일한 계층구조를 가진다!
* 여기서 제일 중요한 것이!! scope 내의 모델에 접근하는 방식도 계층적 구조라는 것
* 즉! 뷰에서 모델에 접근 시 부모 scope를 거슬러 올라가는 것이 가능!!
* 아래 예제는 자식 scope에서 부모 scope의 데이터 접근이 가능한 것을 보여주는 예제
* 
* scope는 모델(데이터)을 찾을 때, 자신의 scope에서 먼저 찾고 없으면 부모 scope를 거슬러 올라가 최종적으로 root scope까지 거슬러 올라간다!!
* 이는 유용하지만, 구조 특성상 의도치 않은 값을 출력하게 되는 경우에 유의
* 이런 경우를 막기 위해 isolate scope를 지원!! 이는 일반 scope와 달리 오직 자신의 scope 내에서만 모델과 핸들러를 찾음, 이를 이용하여 자신만의 독립적인 사용자정의 디렉티브를 만드는데 활용

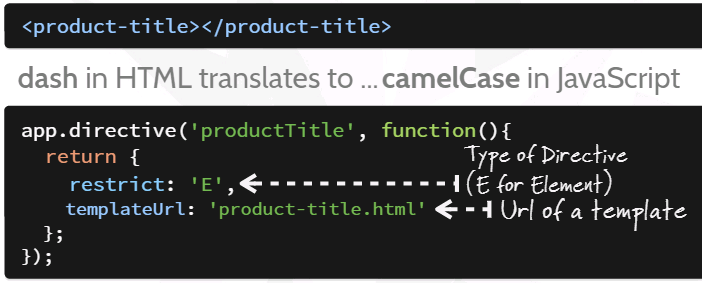
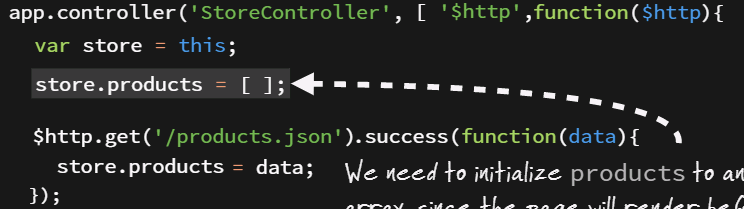
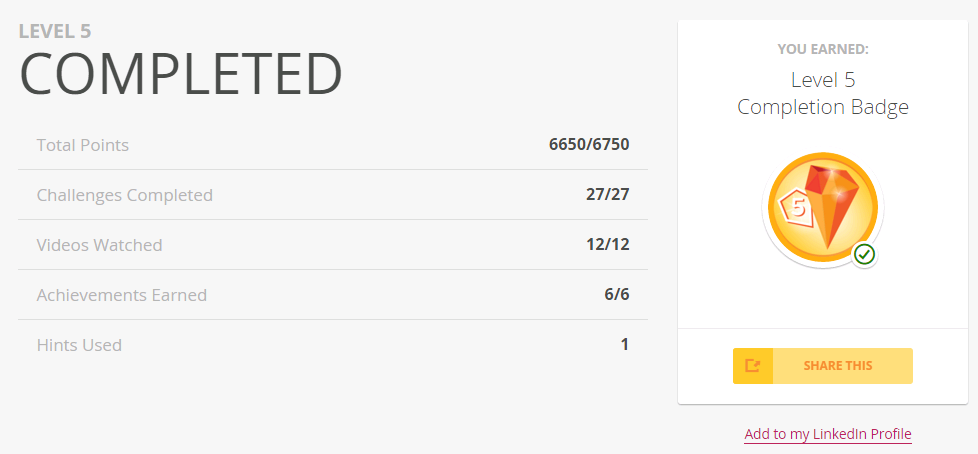
1. Angular의 강력한 데이터 바인딩!! 이게 무엇인지 알아보자

* Angular는 scope에 할당된 모델이 변경될 경우 view에 변경된 모델이 즉시 업데이트, 이의 반대 상황도 즉시 scope의 모델이 변경!! 이를 양방향 데이터 바인딩이라 하는데, 이것이 개발자에게 편의성을 제공하는 강력한 기능!!
* 이런 양방향 데이터 바인딩을 사용하기 위해 간단한 표현식 {{expression}}을 사용함으로 해결하는데 이를 좀 더 파보자.
* 일반적으로 브라우저는 사용자의 이벤트에 따라 동작, 브라우저는 사용자의 이벤트를 기다리고 이벤트 발생시 callback에서 해당 이벤트를 처리한다. Angular는 이런 일반적인 events-loop를 확장하여 “angular context”를 생성. 이를 아래처럼 보여줄 수 있다.
* 
* angular context 내에는 $digest loop가 존재. 이는 $watch list를 돌며 $watch를 통해 모델의 변경을 체크하고 변경되었을 경우 view를 업데이트. 모델의 변경을 감시하는 감시자의 역할을 하는 $watch는 UI와 모델이 바인딩 될 경우 생성되어 $watch list에 추가.
* 
* 위는 2개의 ng-model로 인해 바인딩 됐으므로 2개의 $watch 생성.
* 
* 위는 데이터는 2개지만, 1개의 {{expression}}을 사용했으므로 1개의 $watch 생성.
* 이렇게 생성된 $watch는 $watch list에 담기게 된다.
* 그럼 데이터바인딩이 어떻게 이루어지는지 더 파보자. 브라우저는 이벤트를 기다리다, 이벤트가 발생하면 이벤트는 angular context내에 접근. (이때 $apply()를 통해 접근하는데 이게 엄청 중요하므로 꼭 기억해두자!!)
* 이후 $digest loop를 호출하여 $watch list내의 각 $watch에게 변경이 일어났는지 확인을 요청. dirty-checking으로 인해 $watch가 변경되었다면 다시 $digest loop를 호출하여 모든 $watch의 변경을 다시 요청. 무한루프에 빠질 우려가 있으므로 루프는 최대 10번만 수행하게 되며 루프를 모두 수행한 후에는 변경이 일어난 $watch에 해당하는 scope객체를 view에 업데이트 하게 된다. 아따 길다.
* 여기서 주목해야 할 것은 이벤트가 $apply()로 angular context에 접근한다는 점.
* ng-click 등 Angular에서 지원하는 이벤트 리스너를 사용할 경우에는 내부적으로 $apply()를 호출하기 때문에 angular context로 접근이 가능하며 $digest loop를 수행하여 모델의 변경을 감지할 수 있다. 그럼 이게 왜!!! 중요하냐!!!!!
* 순수 Angular를 사용하여 개발하면 문제야 없다. 하지만… jQuery 쓸 때도 있잖아… Angular 이벤트 리스너를 사용하지 말고 jQuery의 이벤트 핸들러를 사용한다면!? 당연히 $apply()는 호출되지 않고!!! angular context 내부로도 접근 못하지… 즉 $watch를 통한 변경을 감지하지 못하고, 모델이 변경되어도 view에는 업데이트 안 된다는 것… ㅠㅠ
* 자 그럼 이를 어떻게 해결하나? 직접 $apply()를 호출하면 간단!
* 
* 위와 같이 수행하면 간단한데… 이는 에러를 캐치 못해!! 그러니 아래처럼!!
* 
* 위와 같이 function 인자를 받는 $apply(fn)을 사용하도록 하자. 내부적으로 try/catch를 하고 있기 때문에 exception 처리가 가능하다.
* $apply()는 3rd-party Library를 이용해 데이터바인딩을 구현하기 위한 함수로 순수 Angular내에서 $apply()를 사용하면 동작하지 않는다.
* 그런데 이 $apply()를 마구마구 사용하면… 충돌이 발생한다!!! 여기 또 함정이 ㅠㅠ.. 컨트롤러에서 이미 $apply()가 동작 중인데 다시 $apply()를 사용하면 충돌이 일어난다.
* 
* 이는 위와 같이 해결하면 된다.
* 자 그럼 Angular 내부에서 사용되는 방법은 없을까?? 라이브러리 없이… 있다!! $watch를 만들면 된다.
* $apply()와 동일한 메커니즘을 가졌고 Angular 내부에서 사용되며 특정 모델의 변경에 대한 함수를 등록할 수 있다. 오오!!
* $watch의 첫 번째 인자는 string, function 가능, 여러 개 모델 감시는 ;로 구분. 두 번째 인자는 첫 번째 인자의 값이 변경되면 수행될 핸들러.
* 
* <input> 값을 변경할 때마다 updated 값이 1씩 증가하여 {{updated}}에 값이 반영. 즉, DOM이 업데이트 된다. view에서 값이 변경되어 controller가 수행될 때 $watch를 발견하게 되고 바로 $watch가 수행되는 원리. 하지만 이는 미완성.
* controller가 수행되면서 무조건 $watch가 수행되는 것을 막기 위한 방법으로 두 번째 인자 function의 파라미터 값으로 new, old값을 받아 비교하자. 최초 수행 노노하자.
* 
* 위와 같이 수정하면 해결 가능. 주의할 점은 한 페이지에 $digest loop를 돌면서 체크하는 $watch가 겁나게 많이 생성되는 경우라면 성능상에 이슈가 있겠지만… 일반적인 경우 dirty-checking하는 $digest loop는 상당히 빠르다.

1. 이번엔 Filter 에 대해 알아보자!!

* 필터란 특정 컨트롤러나 데이터 타입에 국한되지 않고, 어플리케이션 전역에서 적용해야 하는 공통 데이터 변형을 정의하는데 사용. 필터는 데이터가 scope에서 디렉티브로 전달되는 과정에서 데이터를 변경하지만, 소스 데이터를 수정하지 않으므로 view에서 보여주고 싶은 대로 자유롭게 데이터를 변경할 수 있다.
* 필터는 view에서 보여주려는 어플리케이션 데이터에 공통으로 적용할 수 있는 변형 로직을 담는다.
* 필터는 디렉티브에서 데이터를 처리하고 뷰에서 데이터를 보여주기 전에 포맷한다.
* 
* <div ng-repeat="business in businessList" | **filter:{suburb: 'Brisbane City'}**> 이건 Brisbane City인 것만 추출하는 필터!! 뭐 이런 사용법도 있다.!!

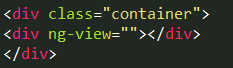
1. Shaping up with Agular.js 예제를 해보자 (여기엔 이슈를 적겠다.)

* ng-include는 개인용 컴퓨터에서 사용할 수 없다. ng-view도 마찬가지인데 이는 ajax 호출을 사용해서 템플릿 파일을 제공한다. 가상서버 없이 로컬로 실행할 경우 파일 요청을 보낼 수 없다. 해결 방법으로는 MAC 사용시 아파치 경로의 폴더에서 작업해야 한다. 오류 내용은 다음과 같다. XMLHttpRequest cannot load ng-include
* directive를 생성하여 진행해봤다. 이번에도 위와 비슷한 오류가 발생. 역시 서버를 이용해야 해결된다. 오류 내용은 다음과 같다. XMLHttpRequest cannot load
* element directive 사용법은 아래와 같다. 주의할 점!!! directive로 사용하는 모든 html파일 name 사이에 ‘-‘이 들어가야 한다. product-title.html 처럼… 이거 때문에 삽질 좀 했다.
* 
* 다음은 attribute directive 사용법이다.
* 
* 뭐 사용법은 위와 같이 두 가지다. html 파일을 생성해서 안에 내용을 추가해야 한다.
* directive 안에서 controller를 사용하고 싶을 땐 어떻게 할까? element directive 방식으로 해보자.
* 
* controller 명이 없어진다. controllerAs 넣을 수 있다.
* controller와 directive를 모듈 별로 나눠보자. 먼저 directive만의 모듈을 만든다.
* 
* 그 다음 controller 모듈에 directive 모듈을 넣어준다. 그럼 끝.
* 
* 이렇게 controller와 directive를 나눠서 관리할 수 있다.
* 자 이제 초기 데이터를 controller에 박아두는… 초기 과정을 빼버리고 http 서비스를 이용하여 데이터를 받아오자.
* 
* 위와 같이 http 서비스를 이용할 수 있다. http 말고 다른 것도 이용하고 싶다?
* 
* 간단하다. 그럼 데이터는 어떻게 가져오나?
* 
* json 파일을 다이렉트로 가져올 수 있다.
* 
* 먼저 데이터 초기화를 하고, 데이터를 가져오는데 성공할 경우에 feedback을 줘서 입력.
* 모두 완성한 경우 아래와 같은 결과를 볼 수 있다.
* 
* 부끄럽지만… 슬라이드 및 동영상을 안 보고 직행하다가 답을 한 번 봤다.. 굴욕이다 ㅠㅠ
* 이것으로 Angular 연습을 마친다.
* generator로 초기 설정을 만든 것을 이용하여 뭔가 만들어보자. 고고!!

Angular generator 분석해보자

맨 위에서 만든걸 이어서 해보자!

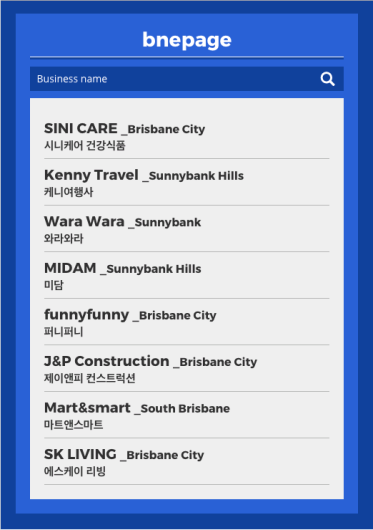
1. 프로젝트 구조 먼저 살펴보자

* 많은 파일들과 폴더들이 생겼는데… 주로 작업하는 폴더는 app 폴더다!!
* index.html: 메인 레이아웃을 담당. 일반적인 프로젝트들과 동일하게 스크립트 등을 이곳에 박아 넣는다.
* scripts 폴더: 당연히 스크립트 파일들이 들어가고, controller와 directive를 넣어주자.
* style 폴더: 말 할 것도 없다. Yeoman은 sass를 쓴다.
* view 폴더: template를 담자.
*  ng-view가 뭐냐? 완전 중요한 부분!! Angular directive 중 하나인데, 현재 라우터에 해당하는 template를 불러온다. index.html에서 저 부분만 계속 바뀜
*  이 부분이 app.js의 모듈 만드는 부분과 router 정의 및 설정이다.
* 그럼 Angular routing이란 뭘까? 페이지 이동에도 불구하고 페이지 refresh가 없고, url에 따라 다른 페이지 template를 일정한 영역(ng-view)에 불러온다. javascipt, ajax call과는 비교도 할 수 없게 쉽고 간편하다!! 와우!!
* Angular route를 사용하려면 ngRoute를 사용할 것이다 라고 알려주자. module에 추가.
* templateUrl이나 controller나 controllerAs는 위에서 다 봤던 거고, when에 페이지 url을, 그 외의 경로는 otherwise가 잡는다. redirecTo: ‘/’는 home으로 날라간다.
* 근데… 일일이 저걸 다 추가해주자고? 난 그렇게 못해!! Yeoman을 믿자. console에 다음 명령어를 입력하자. yo angular:route onemorepage 이렇게 한 줄이면 끝!!

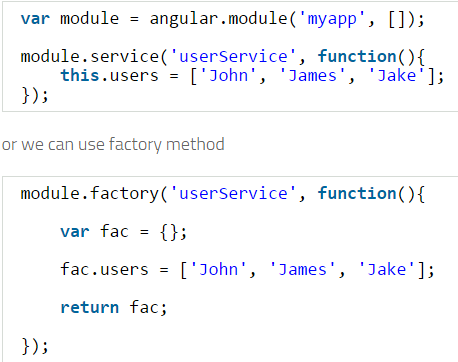
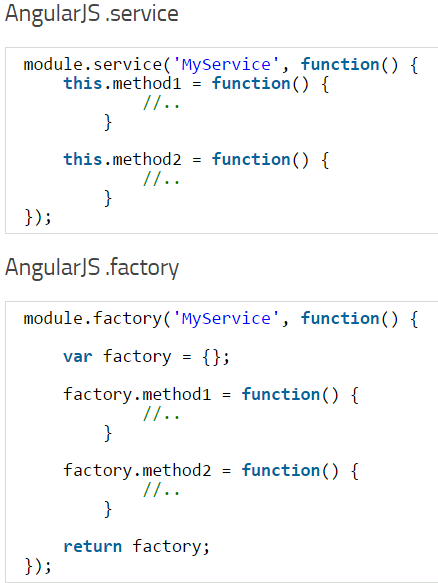
Angular issue!!

Angular에 대해 공부하면서 생긴 일들

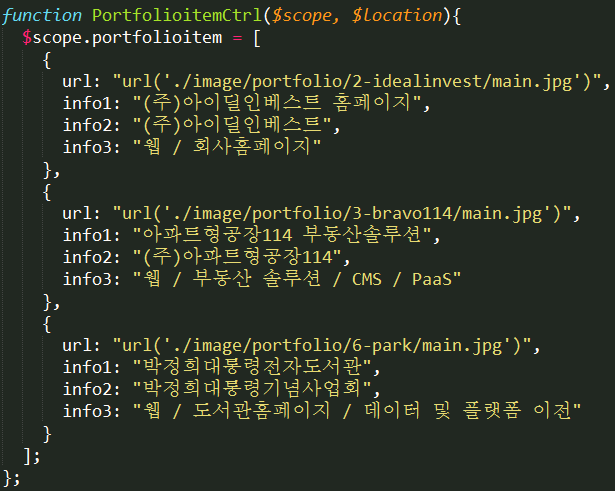
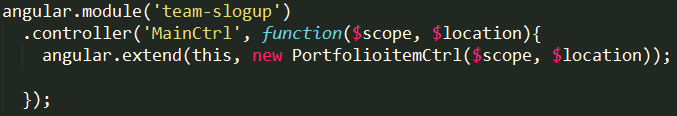
1. $http 서비스에 대해 제대로 알아보자.

* Shaping은… 서버 구동을 안 하고서는 하지 못했기 때문에 뭔가 불안해.. 내 꺼가 아닌 느낌이라 여기서!! http를 파보자.
* 여기서 어느 친절한 분이 예제를 만들어줬다. 이걸 이용해보자.
* git clone <https://github.com/joo486/bnepage.git>
* npm install
* bower install
* grunt serve 차례로 하나씩 입력하면 실행화면이 떠야한다… 근데 안된다? ㅠ
*  디자인은 아주 심플하다. 연습이니까! 연습으로 외부 데이터인 jason 데이터를 가지고 와서 저장하고 그 저장한 데이터를 리스트에 뿌리는 연습을 하자.
* bnepage 타이틀과 서치박스는 두고 하얀 박스 정보들만 바꾸자. ng-view를 이용!
* 그보다 안된다고!! 어째서 ㅠㅠ compass는 뭐야 해결할까… 굳이 이걸 위해…?
* 문제는 저 프로젝트는 compass를 추가했기 때문이었다… 하지만 앞으로의 공부에 굳이 필요치 않아 보인다. 스킵하자.
* 어쨌든 새로 app을 만들어 소스만 가져왔다. 역시 스타일은 깨진다. 하지만 정상적으로 $http 서비스가 작동하는 것을 확인했다. (json 파일을 불러왔다!! 성공!)
* 자 그럼 이제 서비스가 뭔지 궁금하다. 뭘까?

1. Angular service

* $http도 Angular 서비스 중에 하나다. 그럼 custom service를 만들 순 없을까? 파보자.
* 주로 business logic을 서비스에 정의한다. 로직들을 controller에 다 박으면 안되나? controller의 주 기능은 로직 담당이 아니고, $scope를 이용해서 model과 view를 이어주는 역할을 한다. 게다가 controller는 해당 페이지에만 영향을 주기 때문에 페이지가 넘어갈 때마다 $scope에 있는 데이터들은 사용하지 못한다. $rootscope에 저장하는 방법이 있겠지만… 역시 좋은 방법이 아니다!!
* service는 데이터들을 저장하고 controller간 커뮤니케이션을 도와준다. DI(dependency injection)만 해주면 controller, directive, filter 등 어디에서나 사용 가능!!
* 그럼 커스텀 서비스를 만들어보자. 주로 쓰는 방법은 service와 factory다. 사용법은 아래와 같다. 같은 기능을 하지만 방식의 차이가 보인다. factory는 새로운 object를 생성해서 return한다.
* 
* function을 사용할 경우는 아래와 같다.
* 

1. Angular controller 상속

* controller를 상속해서 사용해보자.
* 여기저기서 많이 사용될 item을 가지고 있는 controller가 있고, 이를 불러오는 controller가 하나가 아닐 때, controller를 상속해서 사용해야 한다. 이때 html DOM을 이용한 계층구조가 아니므로 상속하는 명령을 넣어줘야 한다.
* 먼저 item은 아래와 같이 생성한다.
* 
* $scope 뿐만 아니라 $location도 같이 넣어줘야 한다. $scope처럼 상속받는 객체에서 파라미터로 전달해 주기 위해 필요하다.
* 상속받는 controller는 다음과 같이 작성한다.
* 
* extend function 한 줄이면 끝!

1. ng-repeat 의 element에 접근?

* ng-repeat은 항상 새로운 scope를 생성한다.
* current element는 이 새로운 scope에서 사용 가능하다. 방법은 directive를 사용하여 접근하는 방식이 좋다. 아래와 같이 사용한다.
* 

1. Angular input radio type

* radio input을 사용할 때, ng-model을 통일시켜주면 하나씩 체크가 가능하다. 그럼 만약 ng-repeat으로 생성된 radio input은 어떻게 관리할까?
* ng-model=”$parent.value”로 해결할 수 있다. value는 상위 $scope value의 값을 넣어주자.