■ 복습

1) 1_복습.html 페이지 작성

2) angular.min.js 가져다 놓기

3) 한글 로컬 'angular-locale_ko.js' 가져다 놓기

4) data-ng-app 정의

5) module 설정

6) controller 등록

7) currency / date 필터 사용


```
<!doctype html>
<html ng-app="exApp">
<head>
    <meta charset="UTF-8">
    <title>angular복습</title>
</head>
<body ng-controller="exCtrl">
<h1>스마트폰 목록</h1>
<ul>
```

```
    <li ng-repeat="phone in phones">
        이름 : {{phone.name}} /
        가격 : {{phone.price|currency:undefined:0}} /
        출시일 : {{phone.release|date:"yyyy년 M월 d일"}}
    </li>
</ul>
<script src="js/angular.min.js"></script>
<script src="js/angular-locale_ko.js"></script>
<script>
    var app = angular.module("exApp",[]);
    //생성자 함수
    function Phone(name,price,release) {
        this.name = name;
        this.price = price;
        this.release = release;
    }
    app.controller("exCtrl",["$scope",function($scope){
       // alert("test");
        $scope.phones = [
            new Phone("갤럭시S7",760000,new Date("2016-03-07")),
            new Phone("LG G5",980000,new Date("2016-04-07")),
            new Phone("아이폰5SE",564000,new Date("2016-04-12"))
        ];
       // alert($scope.phones[0].release);
    }]);
</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>
```

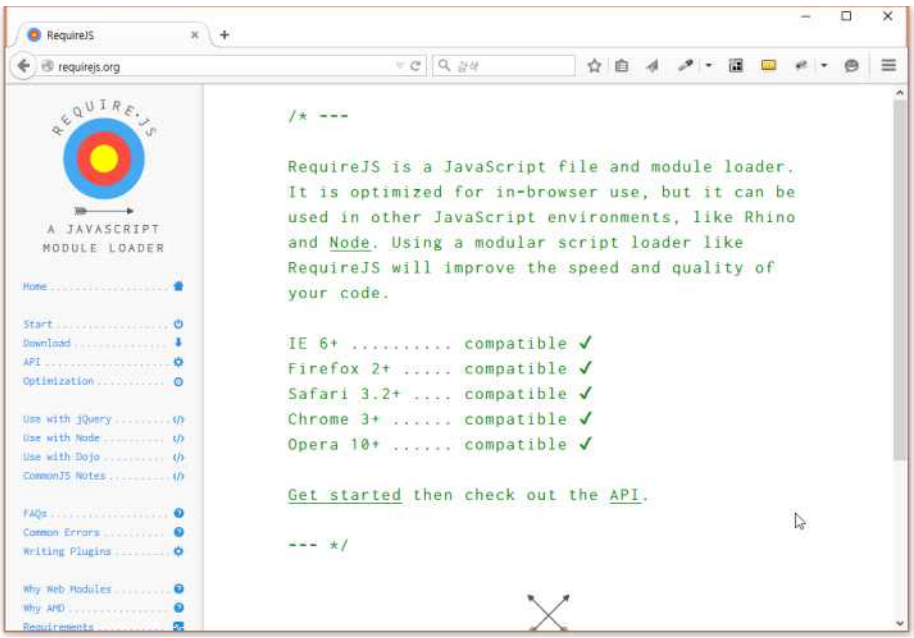
■ $rootScope를 이용한 컨트롤러간 객체 / 데이터 공유

1) 컨트롤러간에 필요한 일을 수행하는 객체를 factory나 service로 등록 가능

2) service나 factory는 기본적으로 필요한 것(설정부분에서 설정)

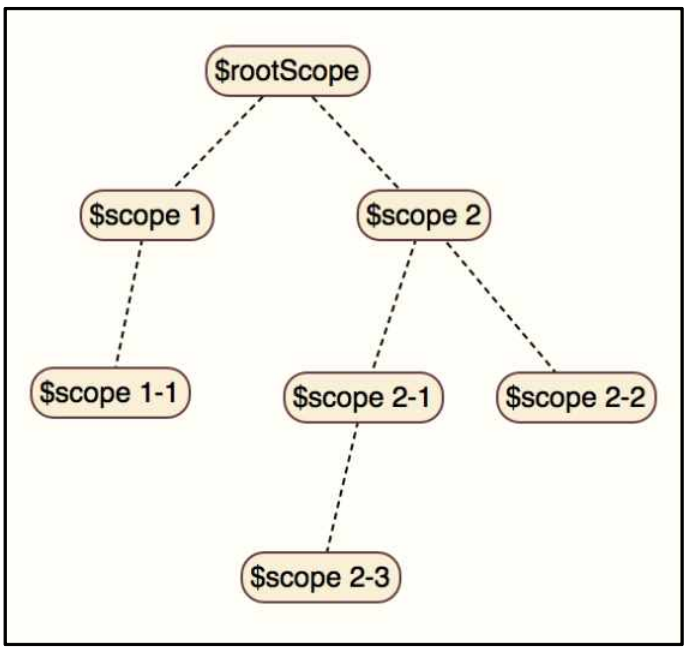
3) service, factory등의 의존성에 따른 로딩은 requirejs를 이용

AMD(Asynchronous Module Definition)



http://helloworld.naver.com/helloworld/textyle/591319

4) 동적으로 필요한 데이터나 객체를 공유하기 위해서는 $rootScope를 이용



```
<!DOCTYPE html>
<html lang="ko">
<head>
<meta charset="utf-8">
<title>active</title>
</head>
<body ng-app="activeApp">
<h1>컨트롤러간 동적으로 데이터 공유</h1>
<ul>
        <li><a href="#view1">1번view</a></li>
        <li><a href="#view2">2번view</a></li>
</ul>
<ng-view></ng-view>
<script src="js/angular.min.js"></script>
<script src="js/angular-route.min.js"></script>
<script>

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

        app.config(function($routeProvider){

                $routeProvider.when("/view1",{templateUrl:"template/view1.js",
                controller:"view1Controller"});

                $routeProvider.when("/view2",{templateUrl:"template/view2.js",
                controller:"view2Controller"});

        });

        app.controller("view1Controller",['$scope','$rootScope',function($scope,$rootScope){

                $rootScope.movie = "쥬라기월드";

                alert("동적으로 movie입력");

        }]);

        app.controller("view2Controller",['$scope','$rootScope',function($scope,$rootScope){

                if($rootScope.movie) {
                        alert($rootScope.movie);
                }else {
                        alert("아직~");
                }

        }]);
```

</script> </body> </html>
view1.js
view1입니다
view2.js
view2입니다

■ Animation의 구현

1) ng-repeat / ng-view 등의 디렉티브에서 변경시 애니메이션 작동

디렉티브	지원 애니메이션
ngRepeat	enter, leave, and move
ngView	enter and leave
ngInclude	enter and leave
ngSwitch	enter and leave
ngIf	enter and leave
ngClass	enter and leave
ngShow / ngHide	enter and leave

2) angular-animate.js가 필요함

3) ngAnimate모듈을 불러오기

4) .ng-enter / .ng-enter-active ¦ .ng-leave / .ng-leave-active

5) 예제

```
<!DOCTYPE html>
<html lang="ko" ng-app="animationApp">
<head>
    <meta charset="UTF-8">
    <title>애니메이션예제</title>
    <link rel="stylesheet" href="css/kakao.font.css"/>
    <style>
        input {
            font:100 20px 'Kakao',sans-serif;
            padding: 10px;
        }
        li {
            padding: 20px;
            width:300px;
            background:#fff;
            box-shadow: 0 12px 15px 0 rgba(0, 0, 0, 0.24), 0 17px 50px 0 rgba(0, 0, 0, 0.19);
            list-style: none;
            font:100 30px 'Kakao',sans-serif;
            cursor:pointer;
        }

        /*
         .ng-animate 클래스가 애니메이션되는 요소에
         들어오는 애니메이션 시작
         .ng-enter
         들어오는 애니메이션의 끝
         .ng-enter-active
         나가는 애니메이션의 시작
         .ng-leave
         나가는 애니메이션의 끝
         .ng-leave-active

        */

        li.ng-animate {
            transition:.6s cubic-bezier(.8,.14,.42,1.51);
            backface-visibility: hidden;
        }
        li.ng-enter {
            opacity:0.5;
            transform:perspective(400px) rotateX(540deg) scale(.5);
            /*
            transform:translateX(-2000px);
            */
        }
```

```
        li.ng-enter-active {
            opacity: 1;
            transform:perspective(400px) rotateX(0deg)  scale(1);
            /*
            transform:translateX(0);
            */
        }
        li.ng-leave {
            transform:translateX(0) rotate(0);
        }
        li.ng-leave-active {
            transform:translateX(2000px) rotate(540deg);
        }

    </style>
</head>
<body ng-controller="aniCtrl">
<input ng-model="name"
        ng-keypress="$event.which==13 && addPerson(people,name)"
       type="text" placeholder="이름입력" autofocus/>
<ul>
  <li  ng-click="remove(people,$index)"
       ng-repeat="x in people">{{x.name}}</li>
</ul>
<script src="js/angular.min.js"></script>
<!-- angular animation 디렉티브를 import-->
<script src="js/angular-animate.min.js"></script>
<script>

    //애니메이션을 적용하려면
    //1) angular-animate.min.js를 불러옵니다.
    //2) ngAnimate 디렉티브를 우리 모듈에 주입합니다.
    //적용되는 범위
    //view에서 route되거나, model에 추가, 삭제될때

    //CSS3의 transition, transform을 이용
    //css에서 처리

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

    app.controller("aniCtrl",["$scope",function($scope){
        //alert("zzzzz");

        //지우는 함수
        $scope.remove = function(people,$index) {
            people.splice($index,1);
        }//remove() end
```

```
        //사람배열
        $scope.people = [];
        //people에 한 명의 person을 추가하는 함수
        $scope.addPerson = function(people,name) {

            people.push({name:name});
            $scope.name = "";
            //alert(people.length);

        }//addPerson() end

    }]);

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

■ Custom Directive 만들기

1) 디렉티브의 종류

종류	설명
A	속성의 디렉티브
E	요소의 디렉티브
C	클래스이름의 디렉티브
M	코멘트와 매칭되는 디렉티브

2) 문법

```
.directive(이름,[의존성주입받을요소명...,function(변수...){
  return 옵션객체
}]);
```

- 이름에는 낙타표기법 (예 : ngAnimate / krInput 등)
- 실제 사용시에는 'ng-animate' / 'kr-input'등 ' -' 사용
- 재사용성을 위해서 모듈을 만들고 디렉티브를 생성

3) 옵션객체의 주요옵션들

옵션	설명
priority	여러 디렉티브중 compile/ link 부르는 순서(높으면 먼저) 기본값 : 0
restrict	디렉티브의 종류 : A,E,C,M
template	뷰 템플릿(글자로 입력)
templateUrl	뷰 템플릿이 될 문서의 경로
scope	스코프에 대한 설정 / '@' , '=', '<', '&'
replace	디렉티브가 없어지고 대치될지 결정(true / false)
compile	디렉티브가 DOM에 추가할때 호출되는 함수
link	compile이 정의되지 않았을때 호출되는 함수
transclusion	디렉티브의 원본을 포함시킬지 설정

4) 예제

movie.card.js

```
angular.module("movieCard",[]).directive('movieCard',[function() {
    return {
        priority:500,
        restrict:"E",
        replace:true,
        transclude:false,
        templateUrl:"js/directive/movieCard.html"
    };
}]);
```

movieCard.html

```
<div class="thumbnail">
    <img src="poster/{{movie.poster}}"/>
    <p class="title" ng-bind="movie.name"></p>
    <p class="price">{{movie.price|currency:undefined:0}}</p>
    <p class="date">{{movie.release|date:"yyyy년 M월 d일"}}</p>
</div><!-- // thumbnail -->
```

4_directive.html

```
<!DOCTYPE html>
<html lang="ko" ng-app="directiveApp">
<head>
    <meta charset="UTF-8"/>
    <title>디렉티브 공부</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">
    <div class="container" ng-controller="directiveCtrl">
        <h2 ng-bind="type"></h2>
        <ul>
            <li class="col-md-4" ng-repeat="movie in movies">
            <movie-card></movie-card>
            </li>
        </ul>
    </div>
</div>
<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 src="js/directive/movie.card.js"></script>
<script>
    var app = angular.module("directiveApp", ["movieCard"]);

    //객체를 만들때 같은 형태(속성의 이름이 같은)로
    //생성하기 위해서 '생성자 함수'를 선언하여 사용
    function Movie(name, price, release, poster) {
        this.name = name;
        this.price = price;
        this.release = release;
        this.poster = poster;
    };

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

        $scope.type = "4월 첫째주 박스오피스";

        $scope.movies = [
            new Movie("주토피아", 15000, new Date("2016-02-17"),"p6.png"),
            new Movie("슈퍼맨대배트맨", 14000, new Date("2016-03-25"),"p2.png"),
            new Movie("런던해즈폴른", 19000, new Date("2016-01-04"),"p1.png"),
```

```
            new Movie("귀향", 21000, new Date("2015-12-16"),"p3.png"),
            new Movie("널기다리며", 9000, new Date("2016-03-30"), "p4.png"),
            new Movie("동주", 18000, new Date("2016-03-02"),"p5.png")
        ];
    }]);
</script>
</body>
</html>
```

5) 예제2

```
<!DOCTYPE html>
<html lang="ko" ng-app="filterApp">
<head>
    <meta charset="UTF-8">
    <title>filter예제</title>
</head>
<body ng-controller="filterCtrl">
<h1>영화 검색</h1>
<!--
<label>검색 : <input type="text" ng-model="name" kr-input
                    placeholder="이름검색"/>
-->
<kr-input ng-model="q" placeholder="이름검색"></kr-input>
</label>
<ul>
    <li ng-repeat="movie in movies|filter:{name:q}">
        이름 :{{movie.name}} /
        감독 : {{movie.director}} /
        총점 : {{movie.grade}}
    </li>
</ul>
<script src="js/angular.min.js"></script>
<script src="js/directive/angular-ko.input.js"></script>
<script>
    var app = angular.module("filterApp",[]);

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

        $scope.test=function() {
            alert("dfsdf");
        }

        $scope.price = 5678890;

        $scope.now = new Date();//날짜형
```

```
        $scope.movies = [
            {"poster":"p1.png","director":"바박            나자피            ","name":"런던해즈폴른
","grade":106,"gradeNum":12},
            {"poster":"p2.png","director":"잭스나이더                  런던","name":"배트맨대슈퍼맨
","grade":106,"gradeNum":12},
            {"poster":"p3.png","director":"조정래 런던","name":"귀향","grade":106,"gradeNum":12},
            {"poster":"p4.png","director":"모홍진","name":"널기다리며","grade":106,"gradeNum":12},
            {"poster":"p5.png","director":"이준익","name":"동주","grade":106,"gradeNum":12},
            {"poster":"p6.png","director":"바이론                          하워드","name":"주토피아
","grade":106,"gradeNum":12}
        ];1

    }]);

    app.directive('krInput', [ function() {
        return {
            priority : 2,
            restrict : 'E',
            scope:{
                ngModel:"="
            },
            transclude:true,
            template:"<input type='text'>",
            replace:true,
            compile : function(element) {
                element.on('compositionstart', function(e) {
                    e.stopImmediatePropagation();
                });
            },
        };
    } ]);
</script>
</body>
</html>
```

6) 예제3

```
<!DOCTYPE html>
<html lang="ko" ng-app="canvasApp">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <style>
        .box {
            border:1px solid #424242;
```

```
            position:absolute;
            left:0;
            top:0;
        }
    </style>
</head>
<body ng-controller="ctrl">
    <box-canvas width="500" height="500" data="data"></box-canvas>
    <script src="js/angular.min.js"></script>
<script>
    var app = angular.module("canvasApp",[]);

    app.controller("ctrl",['$scope',function($scope) {
        $scope.data={width:100,height:100,color:"red",x:20,y:50};
    }]);

    app.directive("boxCanvas",['$document',function($document) {
        return {
          restrict:"E",
            replace:true,
            scope:{
                data:"="
            },
            template:"<div class='box'><canvas></canvas></div>",
            link:function(scope,el,attr) {

                var startX = 0, startY = 0, x = 0, y = 0;



                el.css({width:attr.width+"px",height:attr.height+"px",cursor:"pointer"});

                var canvas = el.children();
               // canvas.css({backgroundColor:"blue"})

                canvas[0].width=attr.width;
                canvas[0].height=attr.height;

                var ctx = canvas[0].getContext("2d");

                var data = scope.data;

                ctx.fillRect(data.x,data.y,data.width,data.height);

                console.log(attr.width);

                el.on('mousedown', function(event) {
                    // Prevent default dragging of selected content
```

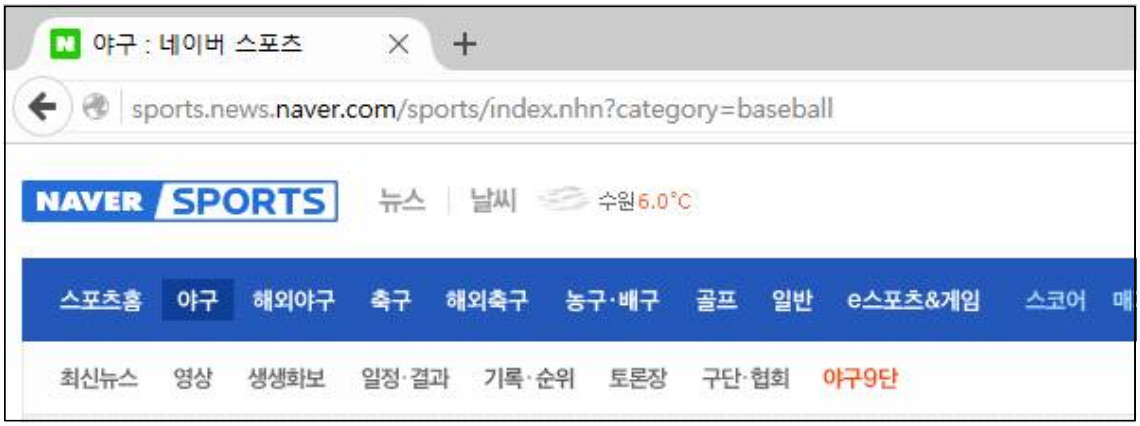
```
                    event.preventDefault();
                    startX = event.pageX - x;
                    startY = event.pageY - y;
                    $document.on('mousemove', mousemove);
                    $document.on('mouseup', mouseup);
                });

                function mousemove(event) {
                    y = event.pageY - startY;
                    x = event.pageX - startX;
                    el.css({
                        top: y + 'px',
                        left:  x + 'px'
                    });
                }

                function mouseup() {
                    $document.off('mousemove', mousemove);
                    $document.off('mouseup', mouseup);
                }

            }
        };
    }]);

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

■ RESTful Web Service란?

1) Representational safe transfer의 줄임말

(https://ko.wikipedia.org/wiki/REST)

2) 하나의 URI는 하나의 고유한 리소스를 대표할 수 있도록 uri를 설계해야 한다는 개념



- 변경 : http://sports.news.naver.com/baseball

3) 리소스 / 메서드 / 메세지로 분류

- 여행지, 카페, 유저, 드라마 : 리소스

- 여행지를 등록하다 / 불러오다 / 수정하다 / 삭제하다 : 메서드

- 여행지 12번의 이름을 '용두암'으로 변경하다 : 메세지

4) 리소스는 uri로 (http://localhost/cafes)

5) 메서드는 HTTP프로토콜의 메서드로

메서드 이름	의미
GET	얻어오기(SELECT)
POST	입력(INSERT)
PUT	수정(UPDATE)
DELETE	삭제(DELETE)

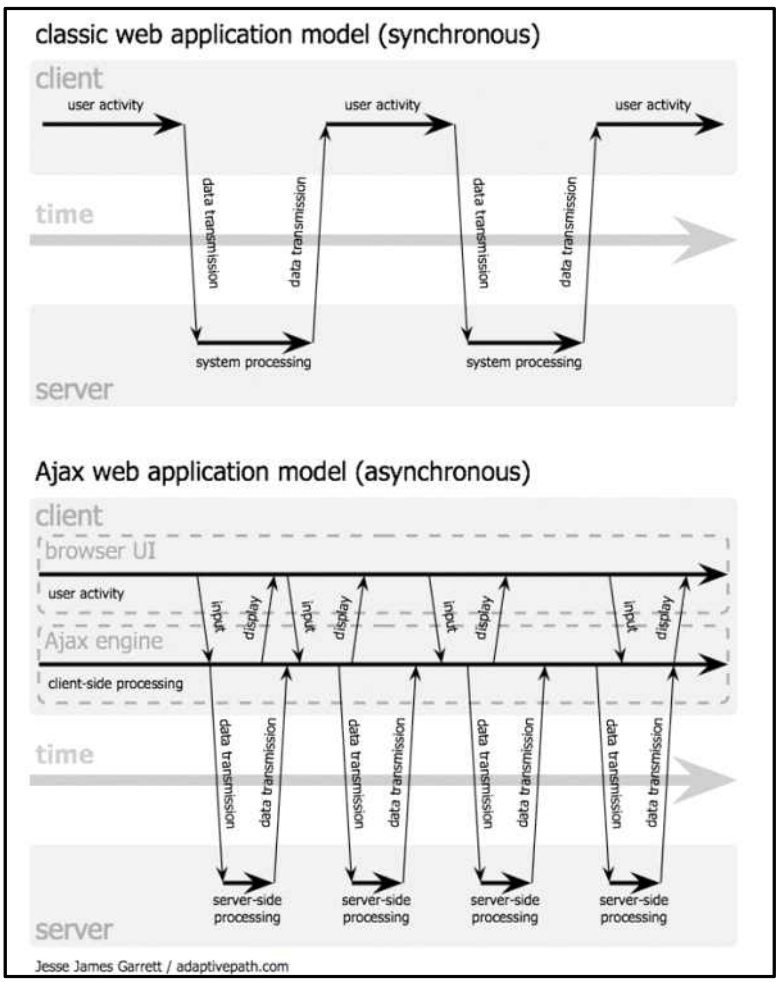
6) 메세지는 JSON으로

7) 예

url	메서드	설명
http://localhost/cafes	GET	카페리스트 페이지
http://localhost/cafes	POST	카페 입력
http://localhost/cafes	PUT	없음
http://localhost/cafes	DELETE	없음
http://localhost/cafes/12	GET	12번 카페 상세페이지
http://localhost/cafes/12	POST	없음
http://localhost/cafes/12	PUT	수정
http://localhost/cafes/12	DELETE	삭제

■ Ajax(Asycronous Javascript and XML)

1) 자바스크립트를 이용한 비동기 통신



2) 전통적인 웹 접근 방법

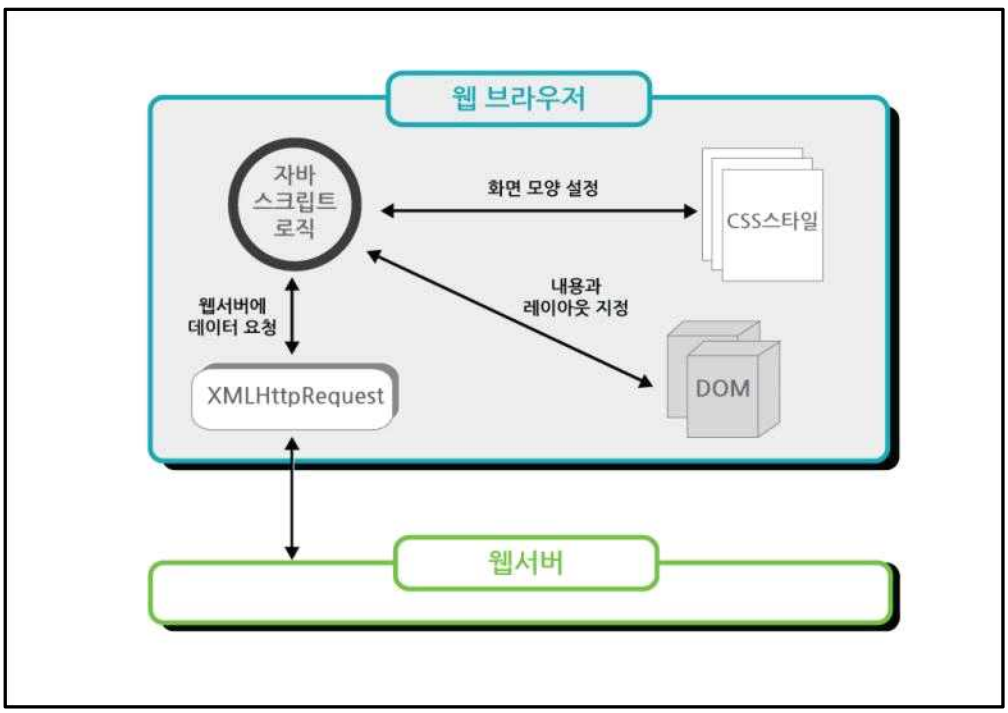
- 요청 / 응답 방식

- 요청에 대한 응답이 서버로부터 돌아오지 않으면 다른 일 수행 불가
- 응답이 돌아오면 다음 페이지로 리로딩 됨

3) Ajax를 이용한 웹 접근 방법

- 요청 / 응답 방식 동일
- 응답은 자바스크립트의 XMLHttpRequest객체가 받음
- 사용자는 하나의 페이지에서 다른일을 계속 수행할 수 있음
- 응답이 오면 자바스크립트의 함수를 통해 일을 수행
- 하나의 페이지에서 여러 가지 데이터를 서버로부터 실시간으로 갱신 가능
- 페이지 리로딩없이 계속 서버와 통신 가능

4) Ajax는 하나의 기술을 의미하는 것이 아님



5) Ajax의 장점

- 서버 처리를 기다리지 않고, 비동기 요청이 가능

- 수신하는 데이터의 양을 줄일 수 있음

- 서버의 부하를 줄이고, 성능좋은 클라이언트를 사용함

시간에 따른 Ajax와 기존방식의 누적전송량

6) Ajax의 단점

- Ajax를 쓸 수 없는 브라우저에 대한 문제가 있음

(사용가능한 브라우저들간의 비호환성 문제(Cross-Browser))

- Http클라이언트의 기능이 한정되어 있음

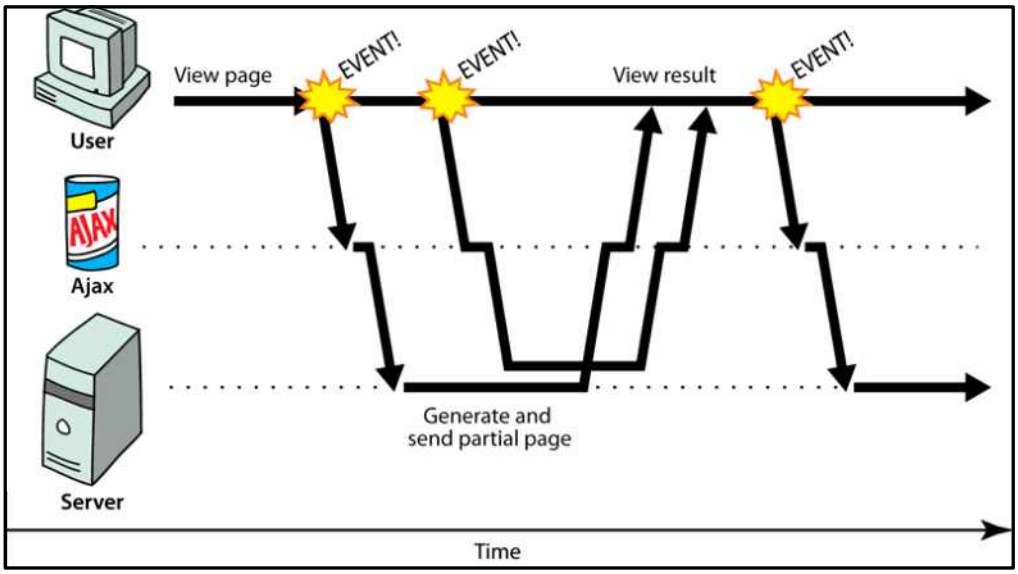
(구글이 크롬브라우저를 만든 이유 : 브라우저 경쟁을 통한 자바스크립트엔진 성능 향상)

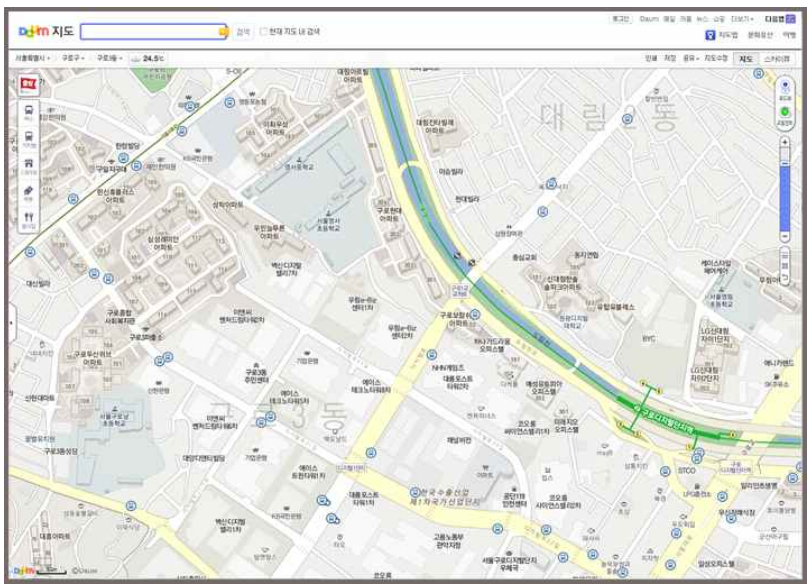
- 페이지 이동없는 통신으로 인한 보안상의 문제

- 클라이언스크립트로 작성되므로 Debugging이 용이하지 않음

■ $resource / $http 서비스 이용

1) ajax처리





2) $http는 promise객체 생성됨

3) 주요 메서드

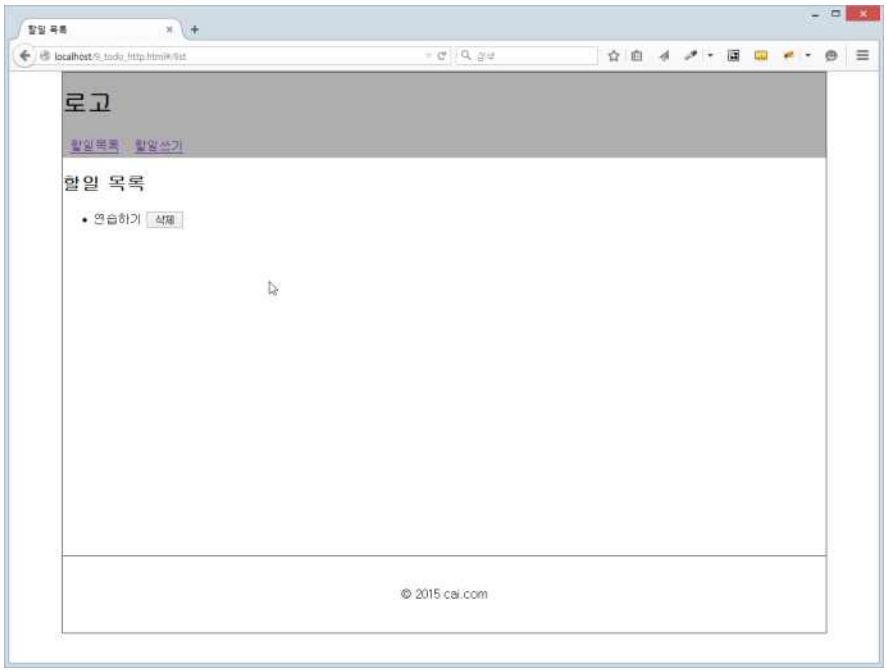
메서드	설명
$http()	기본적인 함수
$http.get()	get방식의 메서드
$http.post()	post방식의 메서드
$http.delete()	delete방식의 메서드
$http.put()	put방식의 메서드

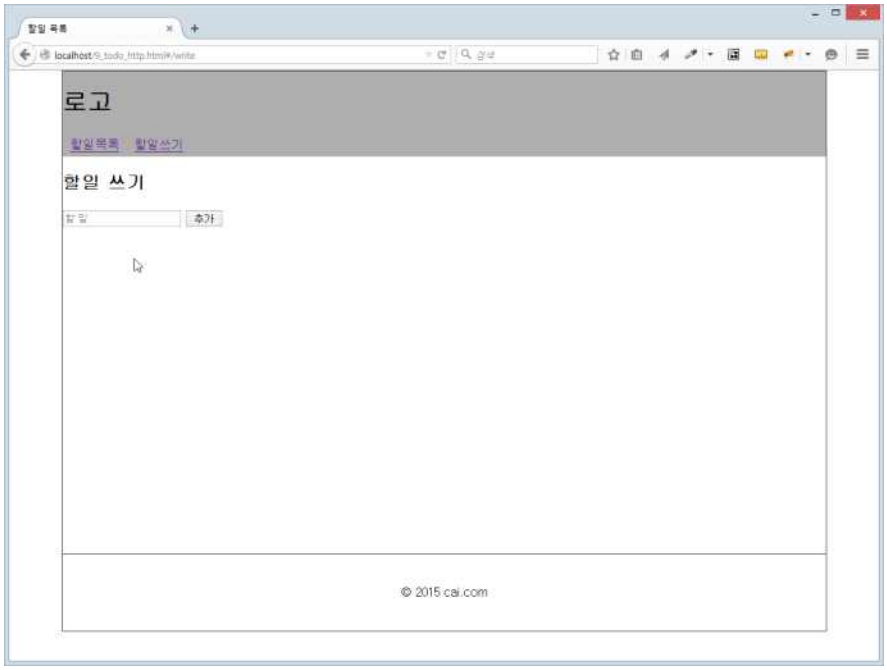
4) $http보다 REST에 최적화된 $resource 이용

5) $resource의 주요 메서드

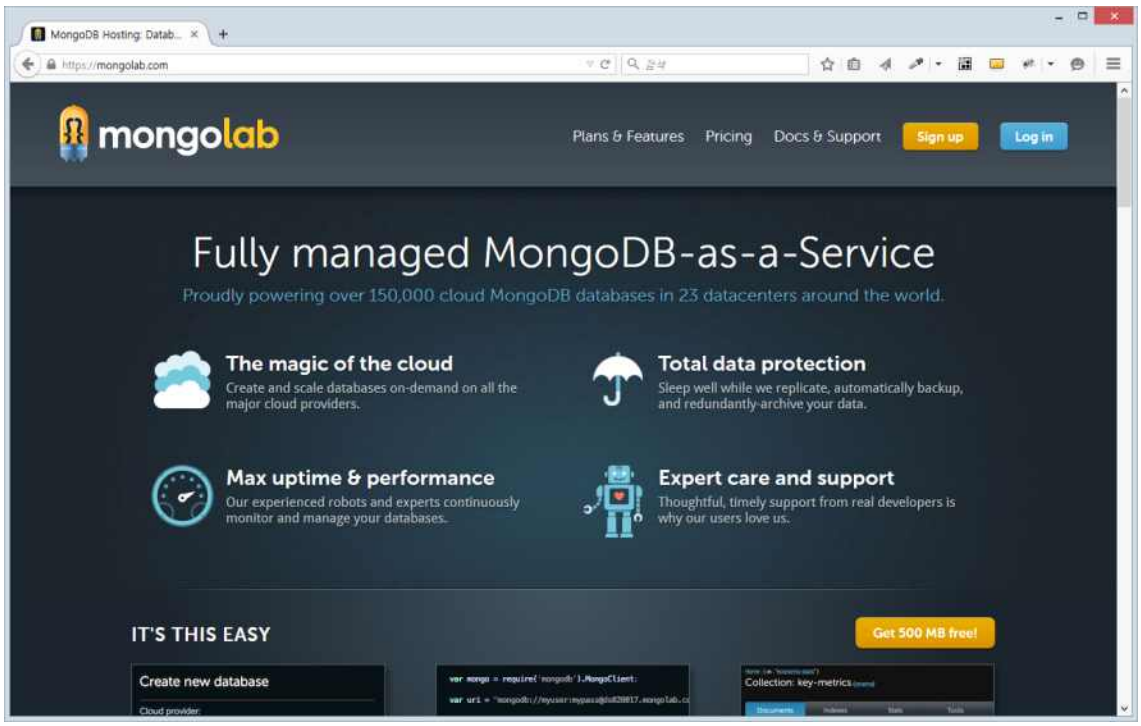
메서드	설명
save	method:PUT (새로 입력)
get	method:GET (한개 가져오기)
query	method:GET(배열로 가져오기)
remove	method:DELETE(삭제)
delete	method:DELETE(삭제)

■ 실제 서버와 연동되는 할일목록 만들기

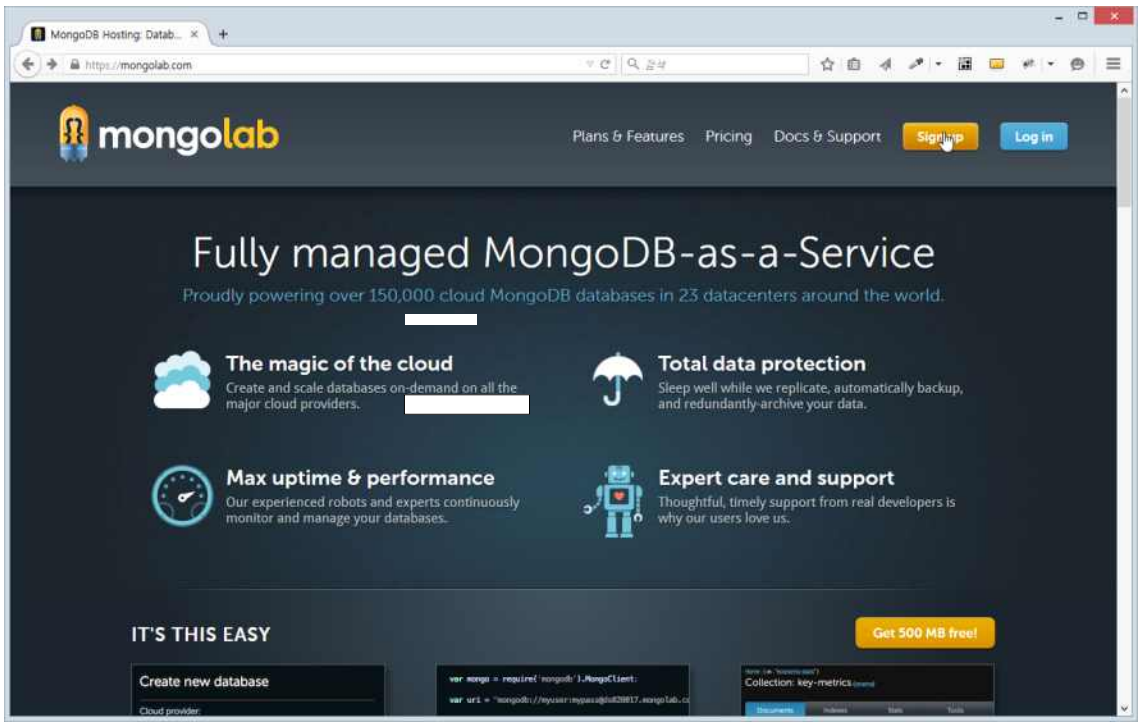




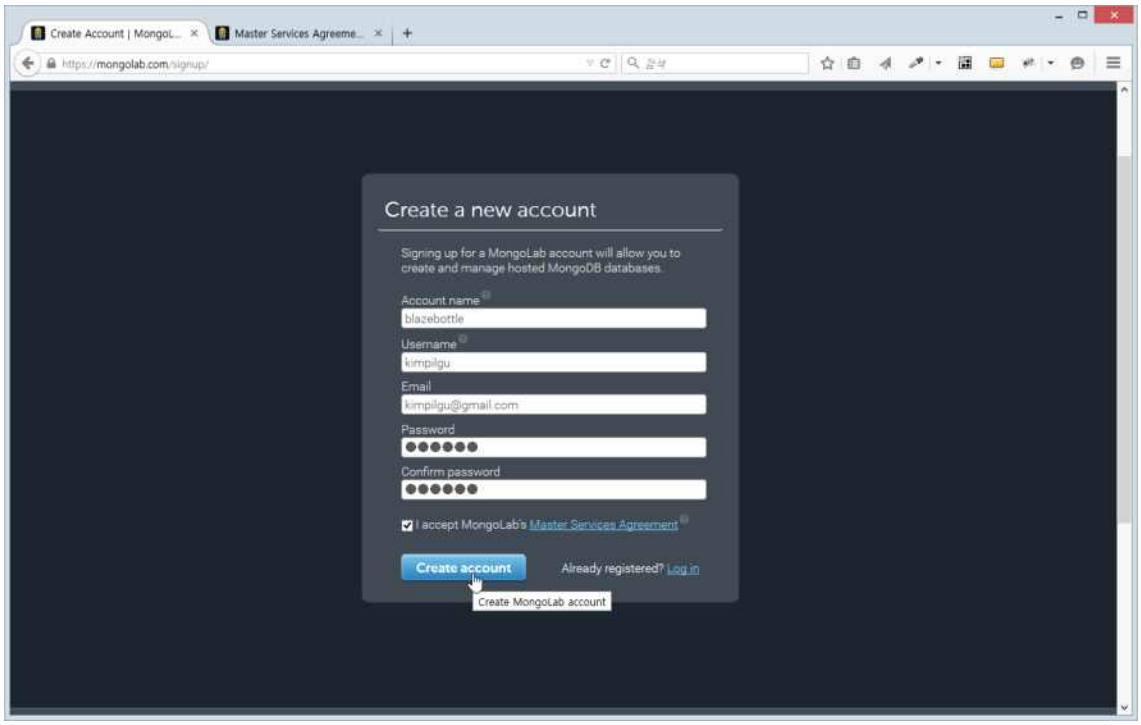
1) REST가 가능한 DB를 위해 mongolab.com을 사용



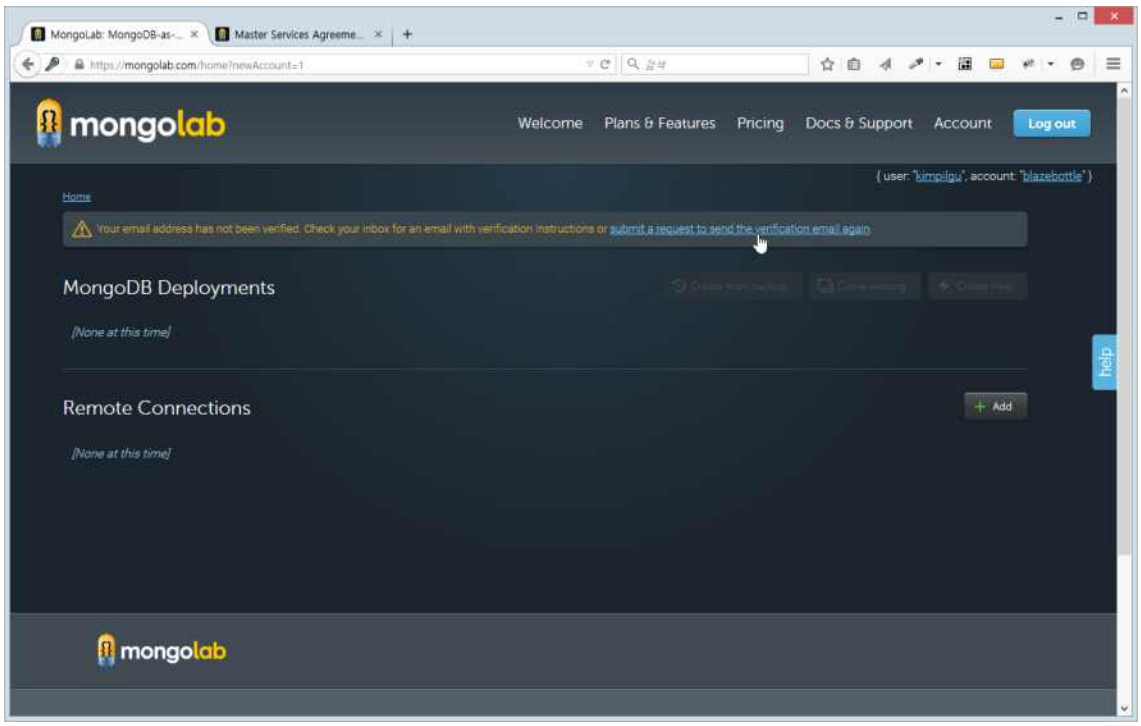
2) sing up(회원가입)



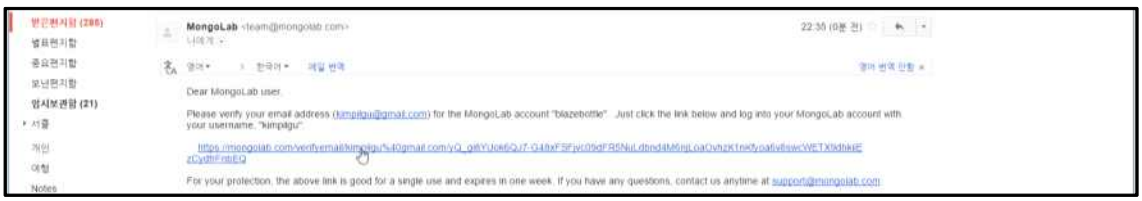
3) 새로운 계정 생성



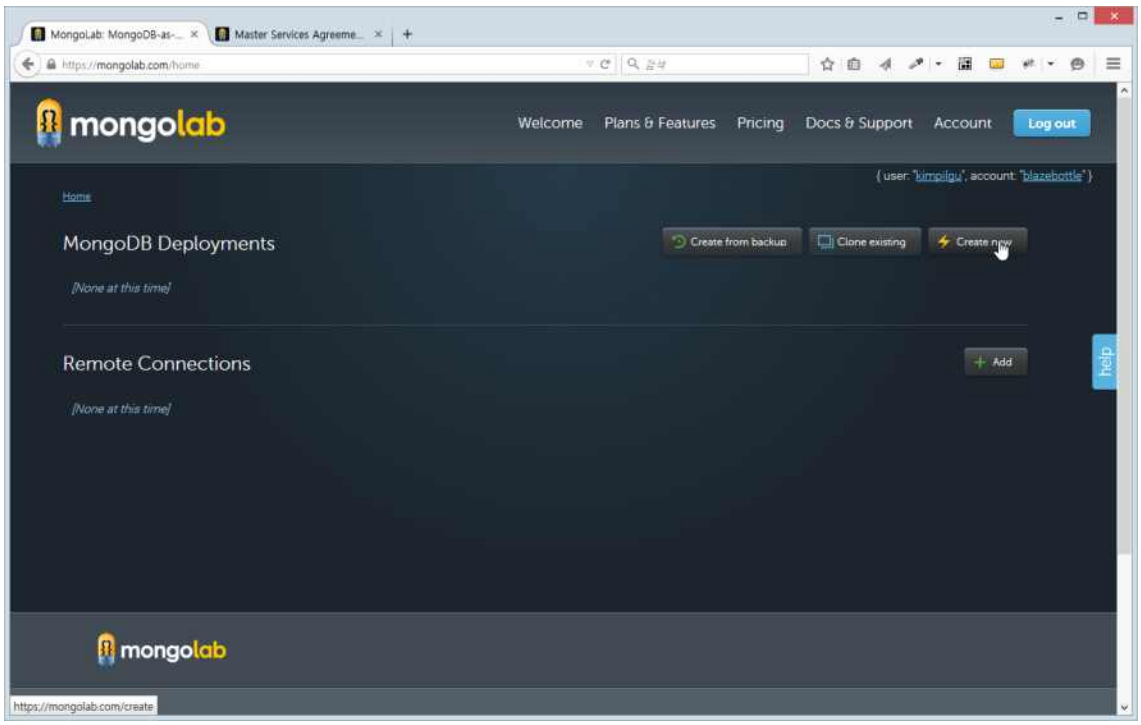
4) 로그인



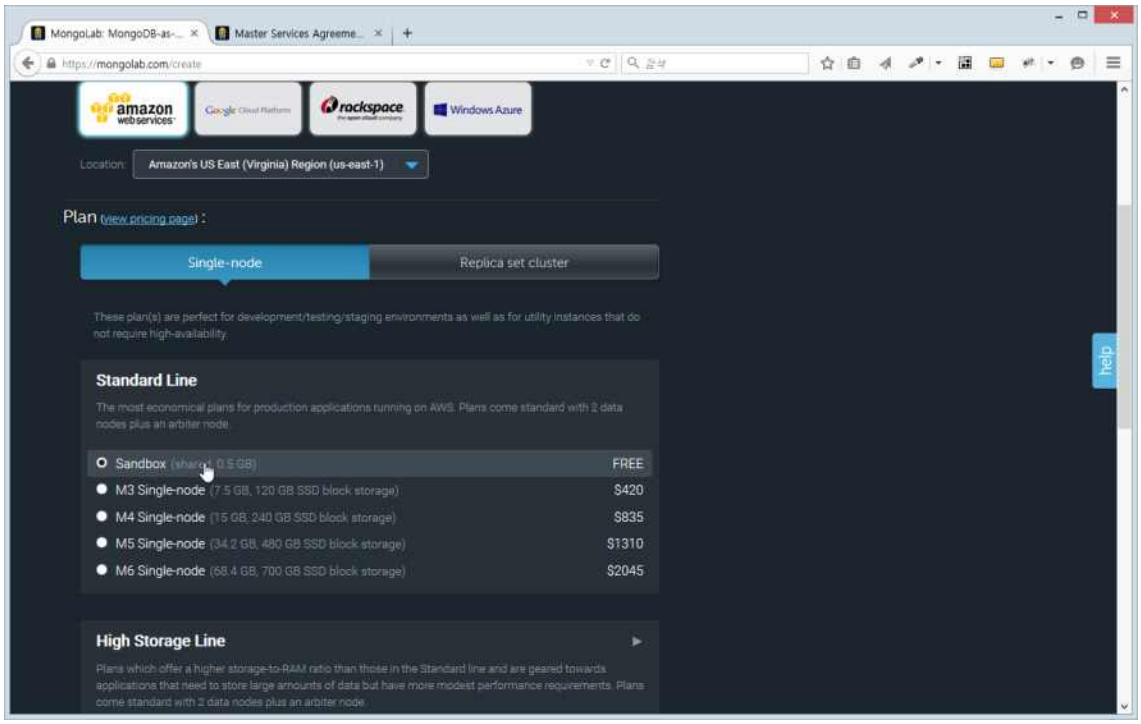
5) 자신의 이메일에서 링크 클릭해야 정상적인 사용이 가능함



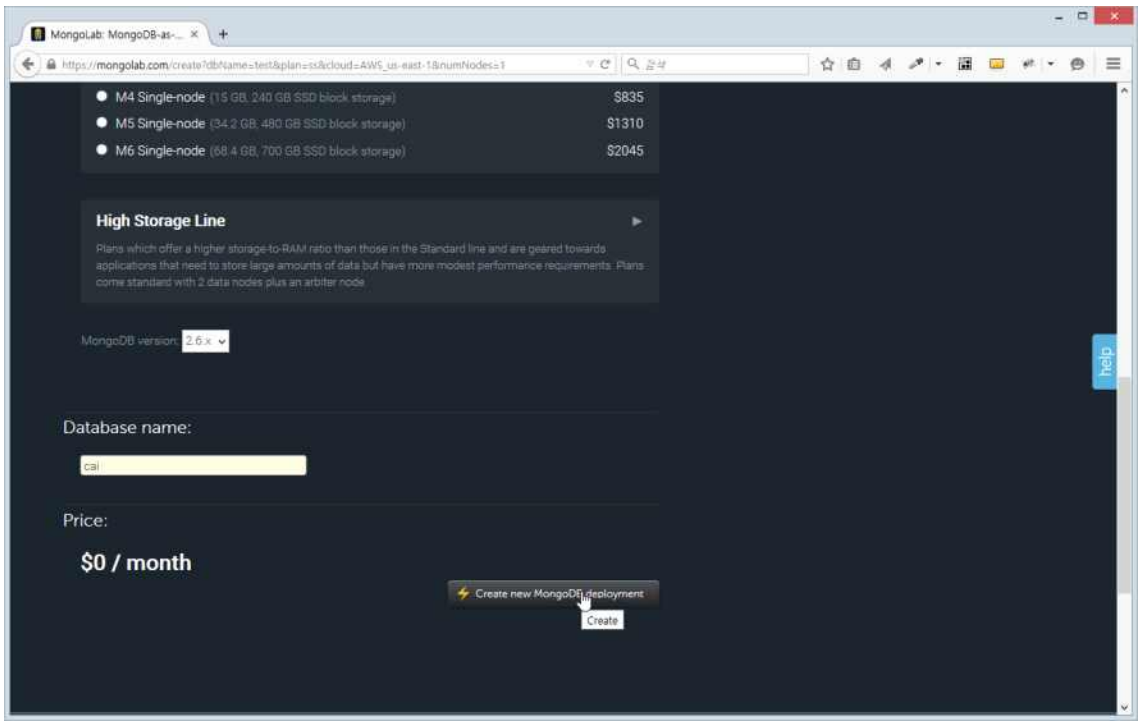
6) DB 생성



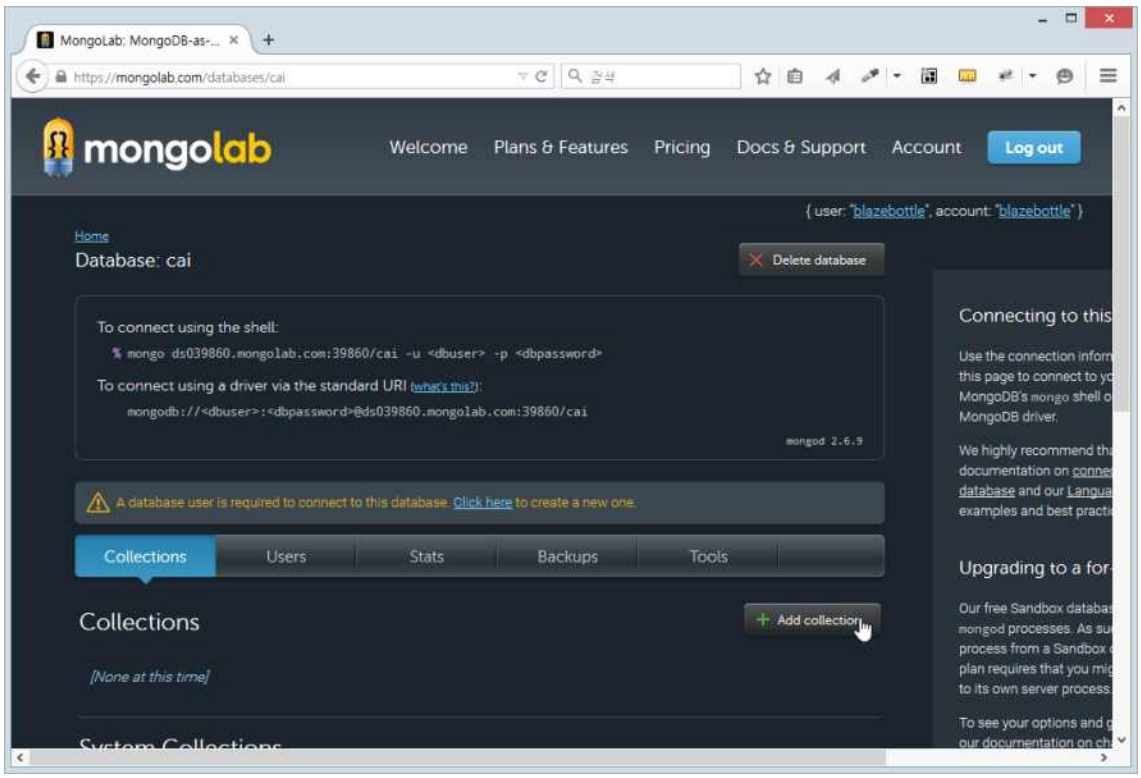
7) single-side / Sandbox로 하면 무료



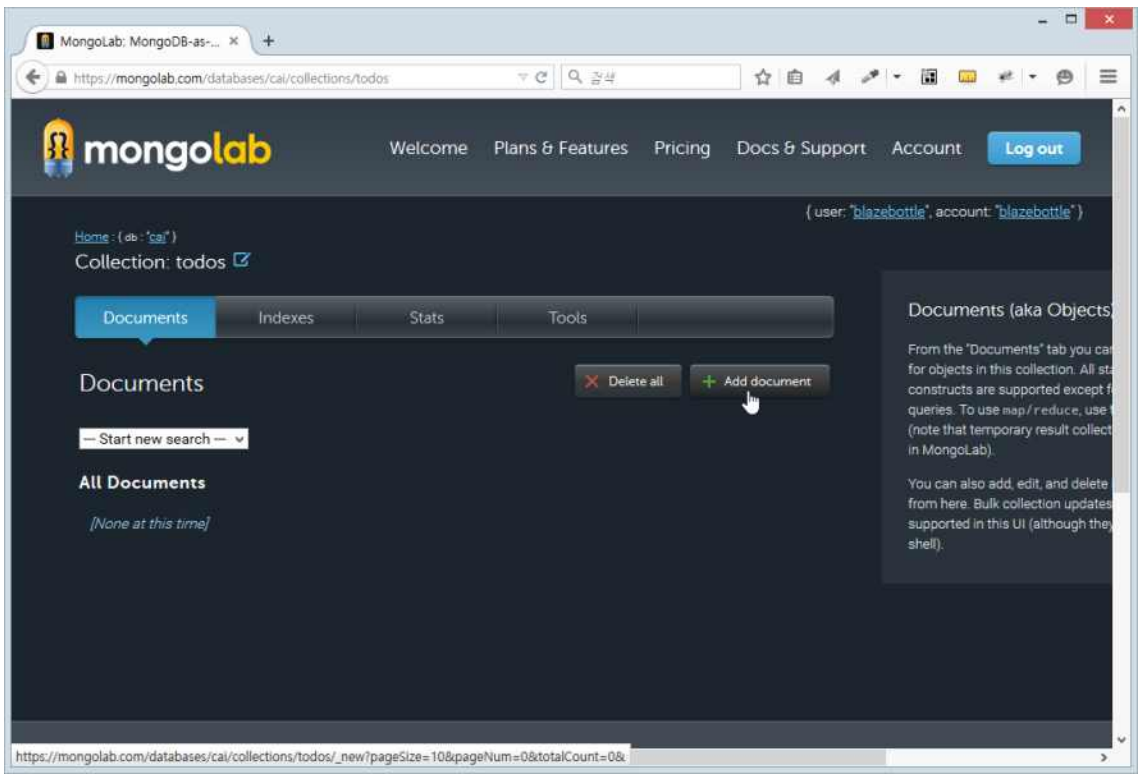
8) 이름을 'cai'로(상관없음)



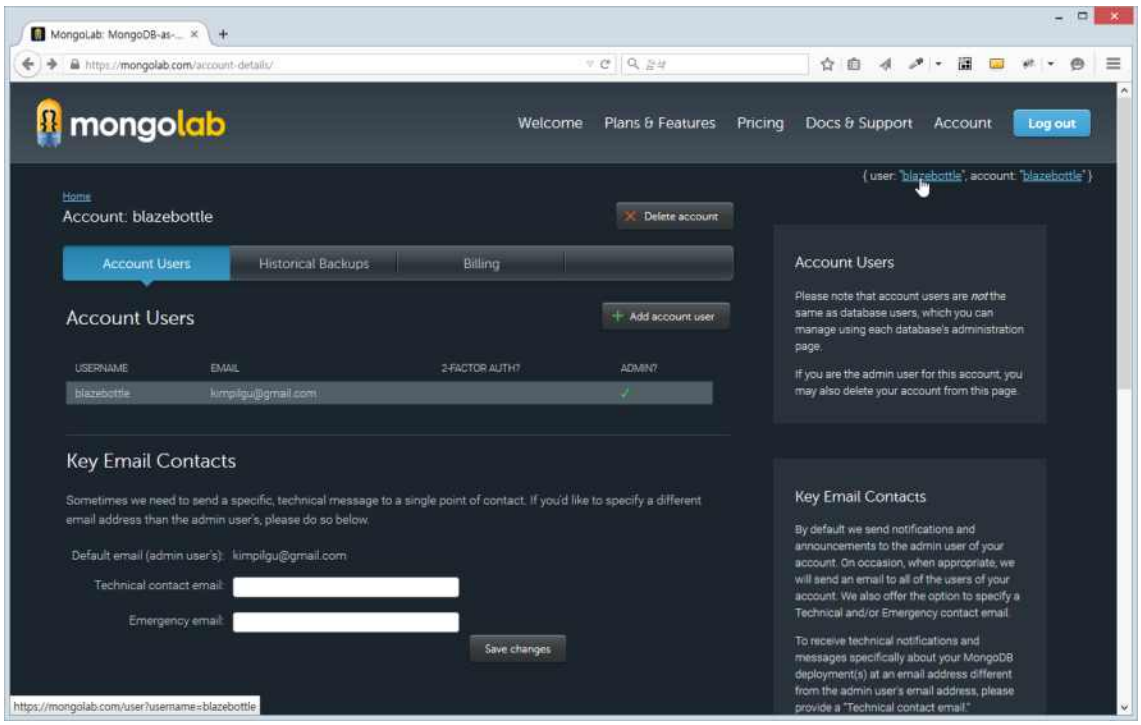
9) todos를 컬랙션을 생성



10) Document 생성



11) 실제 사용하기 위해 apikey를 알아보기



12) 주소와 파라미터

- https://api.mongolab.com/api/1/databases/cai/collections/todos/:id
- apikey : 키 입력
- id : @_id.$oid (mongolab의 아이디)


```
<!DOCTYPE html>
<html data-ng-app="todoApp">
<head>
<meta charset="UTF-8">
<title>할일 목록</title>
<style>
body {
margin:0;}
#wrap {
width:1000px;margin:auto;border:1px solid #333;}

#header {
width:1000px;
background:#aaa;
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;
border-top:1px solid #333}

</style>
```

```
</head>
<body>
<div id="wrap">
<div id="header">
<h1>로고</h1>
<ul>
        <li><a href="#list">할일목록</a>
        <li><a href="#write">할일쓰기</a>
</ul>
</div>
<div id="content">
<ng-view></ng-view>
</div>
<div id="footer">&copy; 2015 cai.com</div>
</div>
<script type="text/javascript" src="js/angular.min.js"></script>
<script type="text/javascript" src="js/angular-route.min.js"></script>
<script type="text/javascript" src="js/angular-locale_ko.js"></script>
<script type="text/javascript" src="js/angular-resource.min.js"></script>
<script type="text/javascript">
var todoApp = angular.module('todoApp', ['ngRoute','ngResource']);
todoApp.config(function ($routeProvider) {
$routeProvider
.when('/list', {templateUrl: 'template/list.js', controller: 'listController'})
.when('/write', {templateUrl: 'template/write.js', controller: 'writeController'})
.otherwise({redirectTo: '/list'});
});
todoApp.service("Todos",function($resource) {

var Todos = $resource("https://api.mongolab.com/api/1/databases/cai/collections/todos/:id",
{apiKey:"mEnc5U1gz4CKrRrgzKUUkIo2nLSbPNyO",id:'@_id.$oid'},
{delete: { method: 'DELETE'},
'update':{method:'PUT'}});

return Todos;
});
todoApp.controller("listController",['$scope',"Todos","$location",function($scope,Todos,
$location){
getList();
function getList() {
$scope.todos = Todos.query(function() {
});
};
$scope.removeTodo = function(todo) {

  todo.$delete(function(d) {
 // alert(JSON.stringify(d));
```

```
  getList();
});
};
$scope.updateTodo = function(todo) {

todo.$update(function(todo) {

        getList();
});

};

}]);

todoApp.controller("writeController",['$scope',"Todos","$location",function($scope,Todos,$location
){

$scope.addTodo=function(todo,$event) {

        //브라우저가 가진 기본기능 막기
        $event.preventDefault();
        $scope.addTodo=function(title,$event) {

        //브라우저가 가진 기본기능 막기
        $event.preventDefault();

        var todos = new Todos();

        todos.title = title;
        todos.end = false;

        todos.$save(function(d) {

                console.log(JSON.stringify(d));

                $location.path('/list');
        });

    title = "";

};
};

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

list.js

```
<h2>할일 목록</h2>
<ul>
        <li data-ng-repeat="todo in todos">
                <input type="checkbox" ng-model="todo.end" ng-click="updateTodo(todo)"/>
                {{todo.title}}
                <button data-ng-click="removeTodo(todo)">삭제</button>
        </li>
</ul>
```

write.js

```
<h2>할일 쓰기</h2>
<form method="post" data-ng-submit="addTodo(title,$event)">
<input type="text" name="todo" data-ng-model="title" placeholder="할 일"/>
<button>추가</button>
</form>
```