# MODERN SPRING WEB APPLICATIONS

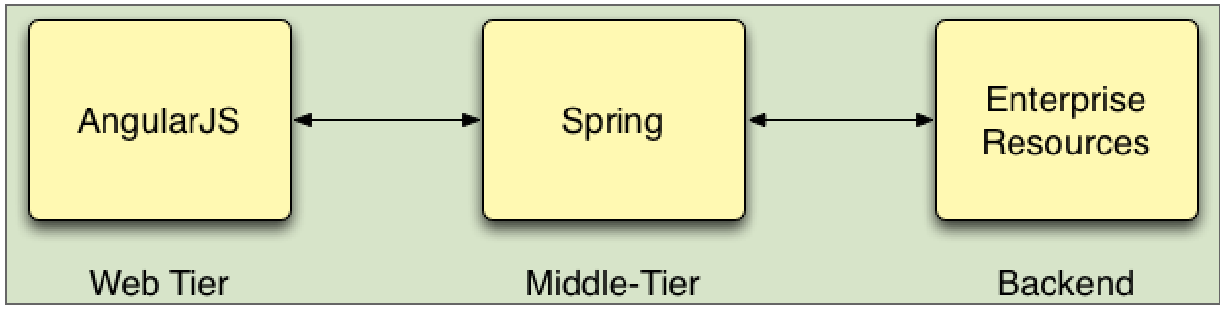
Why it is time to care…

## TRUE 3-TIER CLIENT/SERVER

Javascript on the front-end

Spring as a smart middleware engine

Whatever we need on the backend



## PRESENTERS

Ken Rimple - Chariot Solutions (JS and Spring MVC)

David Turanski - VMware (Spring / WebSockets / Spring Data / d3 push client)

## WHY SHOULD WE CARE ABOUT SPA?

This is where UI innovation is happening

## PROJECT LOCATION

All code available on GitHub at **github.com/krimple/quizzo-ete**.

## JAVASCRIPT RENAISSANCE

Tons of new tools, APIs

See our JS Panel and talks for examples

AngularJS talk is in this room next session!

## JAVASCRIPT'S GOOD PARTS

Turns out it's a functional programming language

And we've been using it WRONG!

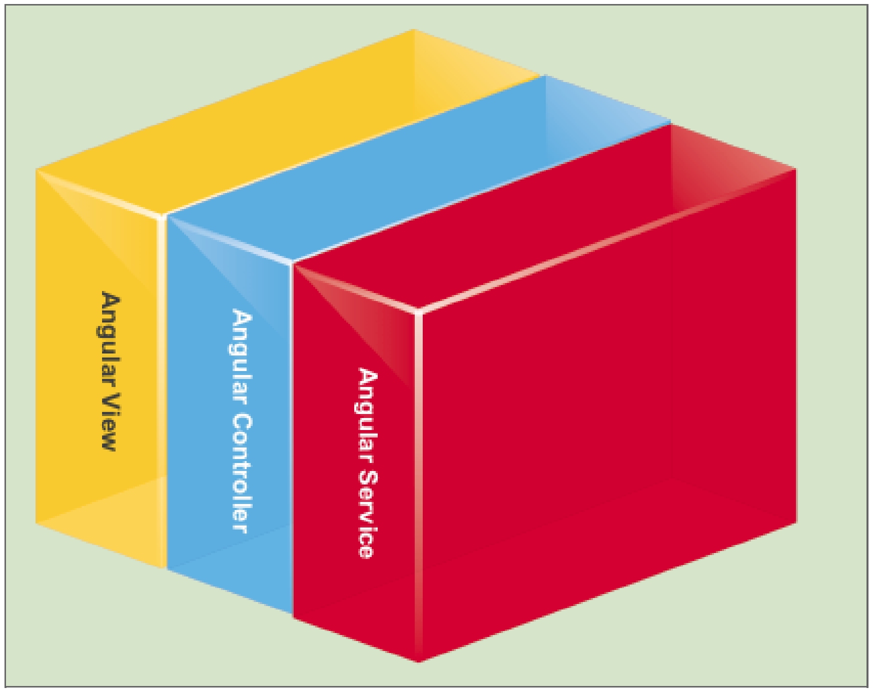
# FIRST, A TRIVIAL EXAMPLE

## THE GAME... - A "QUIZZO"-LIKE APPLICATION

*Let's play...*
LINK

## GAME ARCHITECTURE…

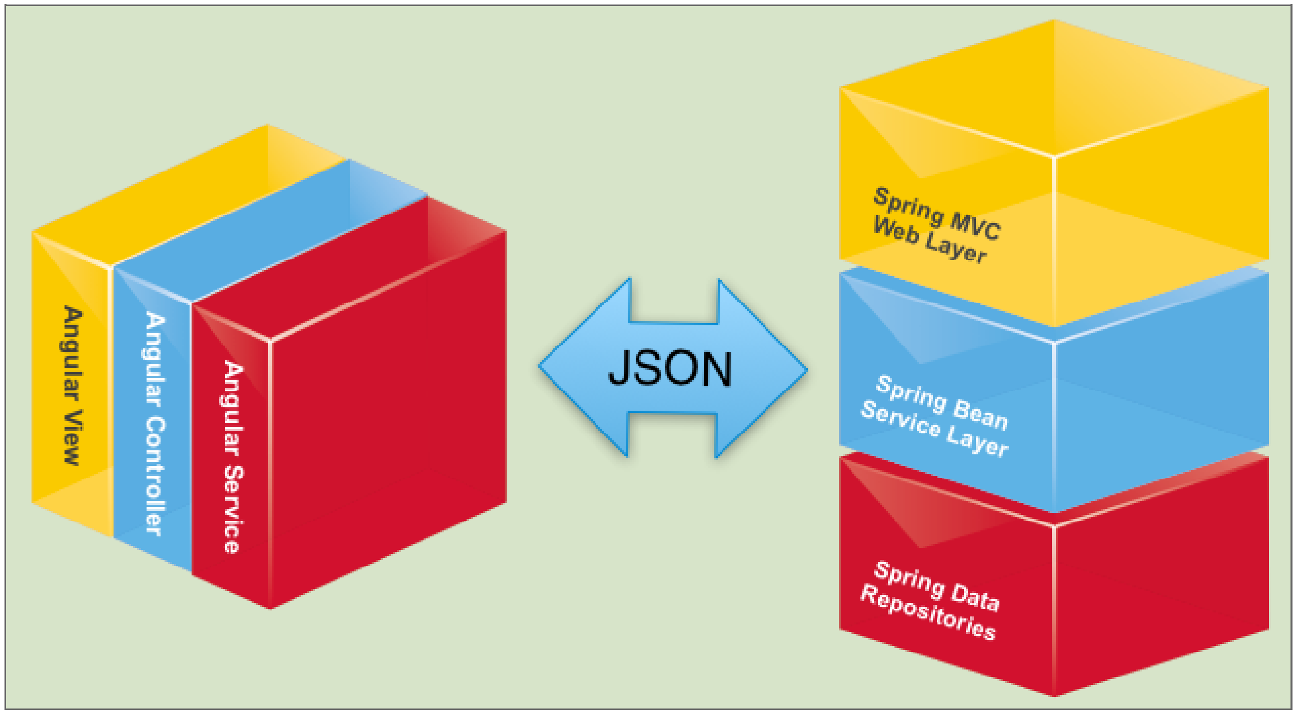
## ANGULARJS



## KEY COMPONENTS

- Router - Routes requests in browser appropriately
- Controller - Sets up shared data between view and page and handles events
- Service - Provides stateful access to application resources, external data
- Directive - Provides HTML componentry
- Scope - shares data between view and model - is a ViewModel

## PUT IT ALL TOGETHER...



## ANGULARJS BENEFITS

BI-DIRECTIONAL DATA-BINDING

MODEL/VIEW/CONTROLLER (OK, MVVM)

QUITE COMPLETE WITHOUT BEING TOO OPINIONATED

## ANGULAR MODULES

```
angular.module('quizzoApp', ['ui.bootstrap']).
  config(['$routeProvider', '$httpProvider',
    function ($routeProvider, $httpProvider) {
    $routeProvider.
      when('/register', {
        templateUrl: 'views/assign_player.html',
        controller: 'RegisterCtrl'
      }).

      when('/join_game/:gameId', {
        templateUrl: 'views/joining_game.html',
        controller: 'JoinGameCtrl'
      }).
      ...
  }]);
```

## CONTROLLERS

```
angular.module('quizzoApp').controller('RegisterCtrl',
   function ($scope, $location, registerPlayerSvc) {

    $scope.$on('GoodNick', function (event, values) {
      $scope.player = registerPlayerSvc.getPlayer();
      $location.path('/show_games');
    });

    $rootScope.join_game = function (nickName) {
      registerPlayerSvc.createNickName(nickName);
    };
});
```

## SERVICES

```
angular.module('quizzoApp').factory('registerPlayerServic
e',
 function (serverPrefix, $location, $rootScope, $http) {
   var implementation = {};

   implementation.createPending = false;
   implementation.currentPlayer = '';

   implementation.createNickName = function (nickName) {
     ...
   };

   implementation.getPlayer = function () {
     return this.currentPlayer;
   };

   return implementation;
```

## CALLING A WEB SERVICE

```
implementation.createNickName = function (nickName) {
  var that = this;
  this.createPending = true;
  $http.defaults.withCredentials = true;
  $http.post(serverPrefix + 'player/register/' + nickName
).
    success(function (data, status, headers, config) {
      if (status === 201) {
        $rootScope.badNick = false;
        that.currentPlayer = nickName;
        $rootScope.playerAndGameInformation
        $rootScope.$broadcast('GoodNick');
      } else if (status === 204) {
        ...
        $rootScope.$broadcast('BadNick');
      }
    });
```

## TEMPLATES

```
<form>
<label for="nickname"><b>Nickname</b></label>

<input type="text"
   ng-model="nickName"
   ng-change="clear_nick_bad()" />

<span class="error"
    ng-show="showJoinError">{{joinError}}</span>

<button ng-click="join_game(nickName)">
  Join Quizzo
</button>

</form>
```

## TEMPLATES

```
<div ng-repeat='game in gamesAvailable'>
  <a ng-href='#/join_game/{{game.gameId}}'>Play!</a>
  :
  <p class='lead'>{{game.title}}</p></div>
</div>
```

## SPRING MVC CONTROLLER

```
@Controller
@RequestMapping("/player")
public class PlayerController {

  @RequestMapping(method = RequestMethod.POST,
                   value="register/{nickName}")
   public @ResponseBody ResponseEntity
    registerUserByNickName(HttpSession session,
        @PathVariable String nickName) {
       ...
    }
   ...
}
```

## HANDLING SUCCESS

```
PlayerGameSession playerGameSession =
  getOrCreatePlayerGameSession(session);

playerGameSession.setPlayerId(player.getName());

responseEntity = new ResponseEntity(player, HttpStatus.CR
EATED);
return responseEntity;
```

## HANDLING FAILURE

```
try {
    player = playerService.registerPlayer(nickName);
} catch (PlayerAlreadyExistsException p) {
    responseEntity = new ResponseEntity<Player>(
        HttpStatus.NO_CONTENT);
    return responseEntity;
}
```

## QUIZZO DESIGN CHOICES

- Game state stored server-side
- Credentials and score established by client but kept on server
- User-specific state (what question, etc) were cached on client

## PAYING ATTENTION TO SERVER STATE

```
angular.module('quizzoApp',
    ['ui.bootstrap', 'angular-underscore']).
  config(['$routeProvider', '$httpProvider',
    function ($routeProvider, $httpProvider) {
      ...
      $httpProvider.defaults.withCredentials = true;
  }]);
```

## DEBUGGING DEMO

## IT COULD BE ANY JS FRAMEWORK

See todomvc.com - Addy Osmani

## SPRING MVC TIPS

- Use `ResponseEntity` for returning conditional values
- Try being RESTful where possible (POST/GET/PUT/DELETE for CRUD)
- Use Jackson JSON for easy JSON serialization
- Validate on both tiers... Ugly but necessary
- Don't send too much - GETs are cheap, sort/filter on server for large sets

## SPRING DATA

AN OVERVIEW

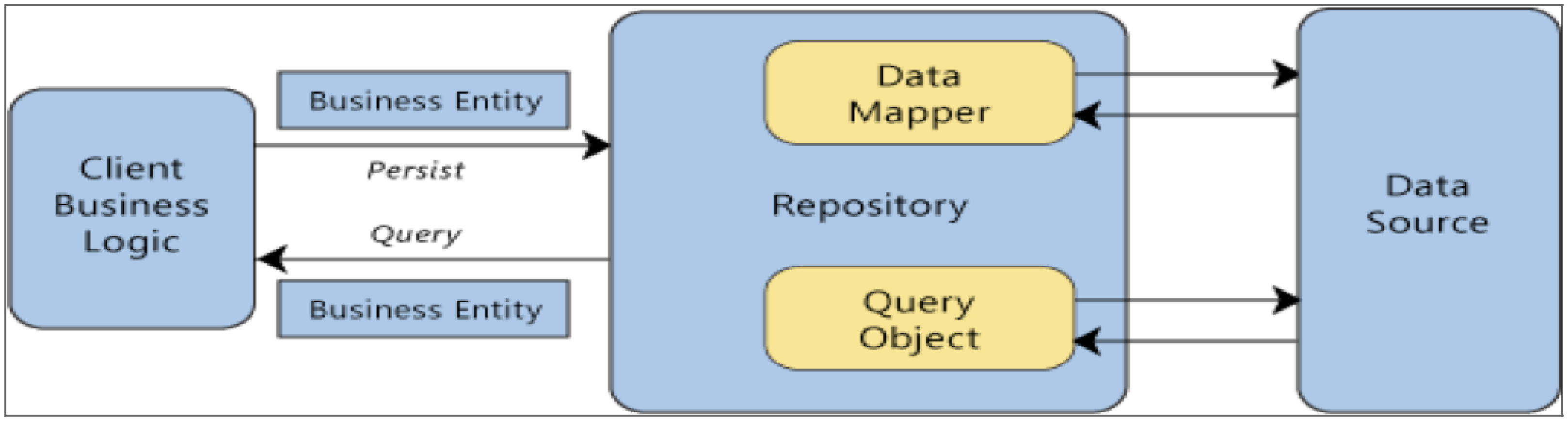
## JPA Query DSL JDBC and...

*“ ... provide a familiar and consistent Spring based programming model while retaining store-specific features and capabilities ”*

## REPOSITORY

*“Mediates between the domain and data mapping layers using a collection-like interface for accessing domain objects.”*

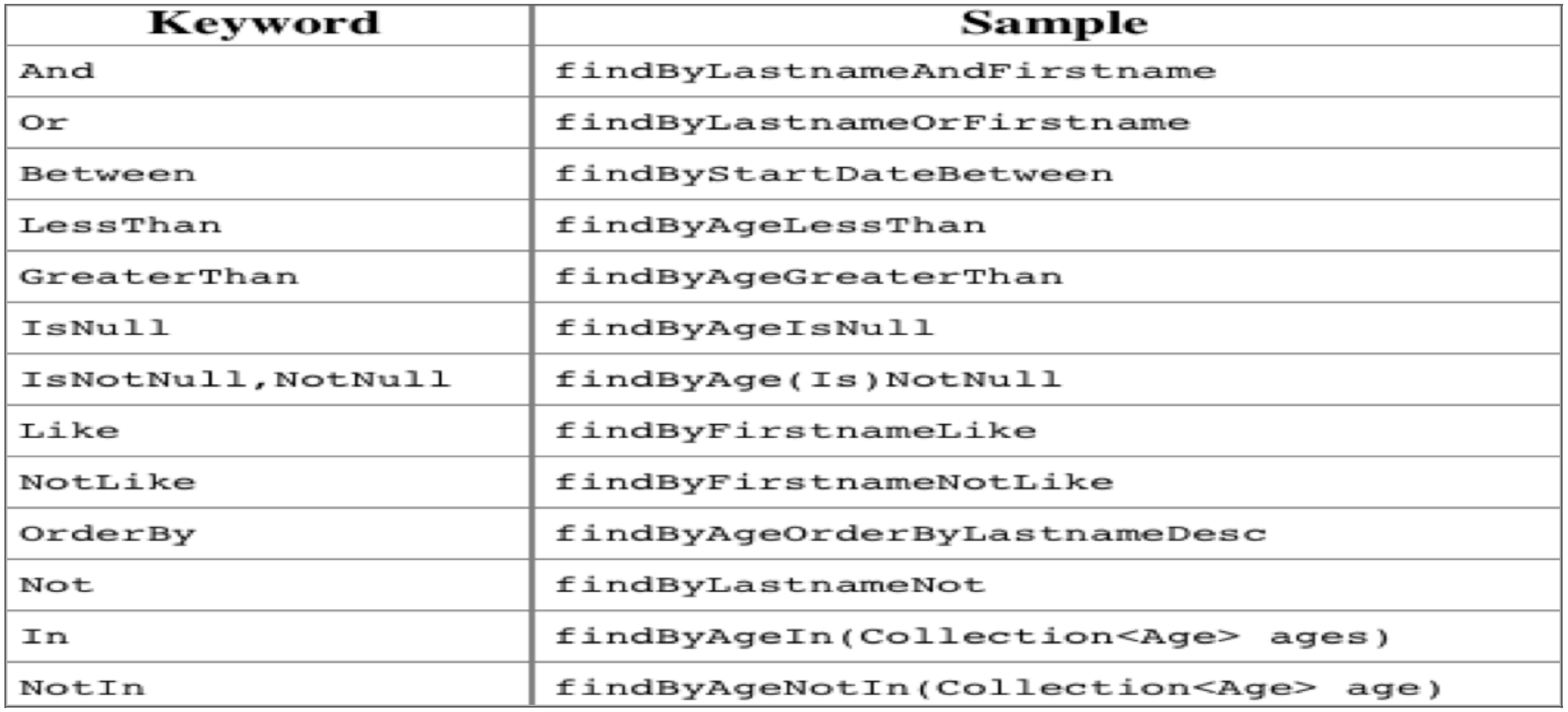
**Martin Fowler**



Spring Data CRUD Repository - You get this out of the box.

```
public interface CrudRepository<T, ID extends Serializabl
e> extends Repository<T, ID> {
  <S extends T> S save(S entity);
  <S extends T> Iterable<S> save(Iterable<S> entities);
  T findOne(ID id);
  boolean exists(ID id);
  Iterable<T> findAll();
  Iterable<T> findAll(Iterable<ID> ids);
  long count();
  void delete(ID id);
  void delete(T entity);
  void delete(Iterable<? extends T> entities);
  void deleteAll();
}
```

Query Methods



| Keyword | Sample |
|---|---|
| And | findByLastnameAndFirstname |
| Or | findByLastnameOrFirstname |
| Between | findByStartDateBetween |
| LessThan | findByAgeLessThan |
| GreaterThan | findByAgeGreaterThan |
| IsNull | findByAgeIsNull |
| IsNotNull,NotNull | findByAge(Is)NotNull |
| Like | findByFirstnameLike |
| NotLike | findByFirstnameNotLike |
| OrderBy | findByAgeOrderByLastnameDesc |
| Not | findByLastnameNot |
| In | findByAgeIn(Collection<Age> ages) |
| NotIn | findByAgeNotIn(Collection<Age> age) |

Also: StartsWith, EndsWith, Contains, After, Before

## PlayerAnswerRepository

```
public interface PlayerAnswerRepository
  extends MongoRepository<PlayerAnswer,BigInteger>,
          PlayerAnswerRepositoryCustom {
  public List<PlayerAnswer> findByQuizId(String quizId);

  public List<PlayerAnswer> findByGameId(String gameId);

  public List<PlayerAnswer> findByGameIdAndPlayerId(
               String gameId, String playerId);
...
```

PlayerAnswerRepository (cont)

```
    public PlayerAnswer
      findByGameIdAndPlayerIdAndQuestionNumber(
                String gameId, String playerId,
                int questionNumber);

    public List<PlayerAnswer>
      findByGameIdAndQuestionNumber(
                String gameId, int questionNumber);
  }
```

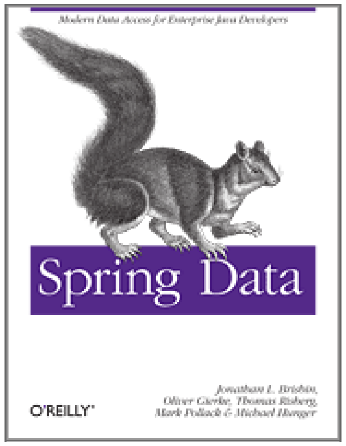
## MAPPING

Spring Data provids store-specific annotations for domain classes

QUESTIONS?

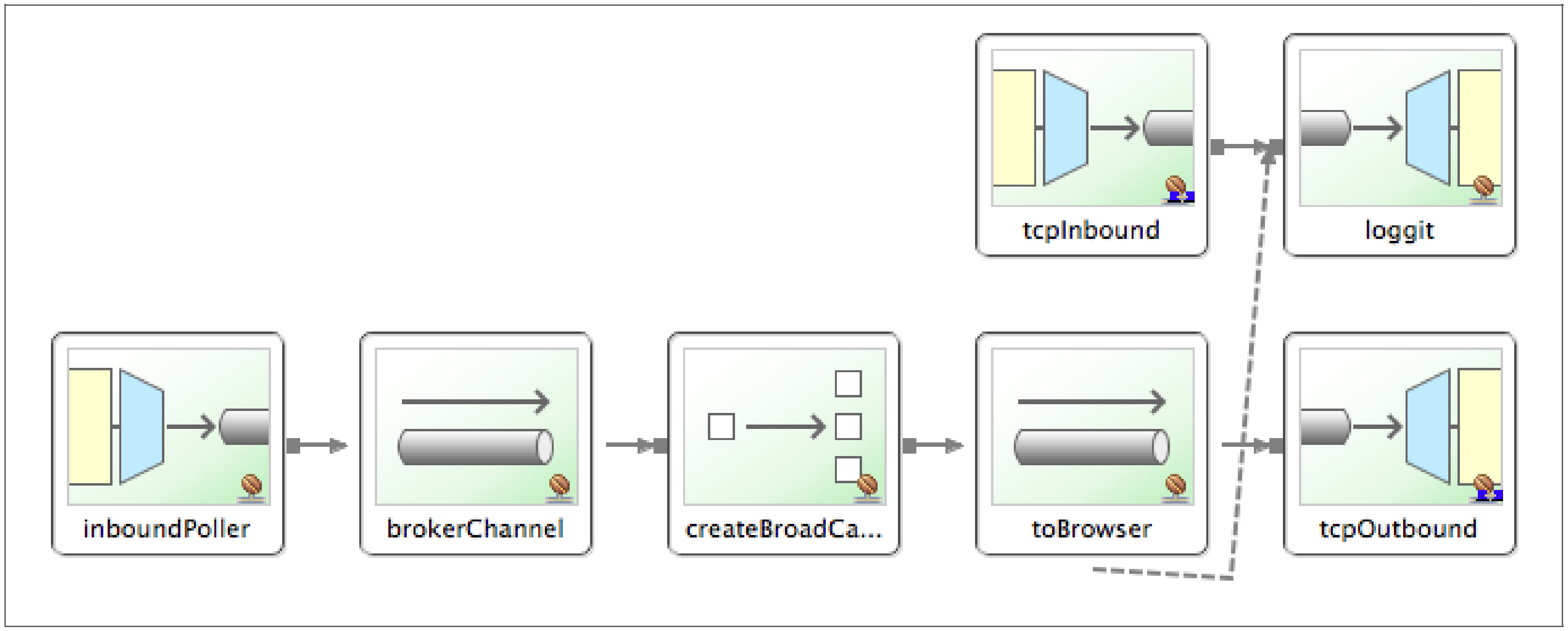
More Information

| | |
|---|---|
| Spring Data Project Page | **http://www.springsource.org/spring-data** |


Modern Data Access for Enterprise Java Developers
Spring Data
O'REILLY
Jonathan L. Brisbin, Oliver Gierke, Thomas Risberg, Mark Pollack & Michael Hunger

| | |
|---|---|
| Source Code | **http://github.com/SpringSource** |

## Spring Integration Flow for WebSockets


tcpInbound
loggit
inboundPoller
brokerChannel
createBroadCa...
toBrowser
tcpOutbound