ORACLE®

JavaOne™

ORACLE®

JAX-RS and AngularJS

Tools for even better experience

Martin Mares
Software developer
Oracle
September 30, 2014

Safe Harbor Statement

Program Agenda

1. Decide if this BoF is for You
2. JAX-RS and AngularJS
3. Day by day TOI (documentation)
4. Do not change your API (DTO support)
5. Few more notes

About this presentation

Primary audience, presentation style and content

- Software development team leader
  - **Developer** who is responsible for project development phase and final design dessign
  - Not project **manager**
- We will browse the code
  - Sources, console, browser
  - (I have a lot of slides, too)
- Series of introduction level description of various tools
  - Starting with JAX-RS and AgularJS – why it well fits together
  - Demonstration of other tools – JAVA TOOLS

About this presentation

Motivation – few lines of philosophy

- Java and JavaScript
  - Same name and so different styles
    - Static - dynamic type language
    - Refactor every day – never refactor
  - Must cooperate: Accept differences and define gray zone
- Java and JavaScript developers
  - Do you have 2in1 in your team? Good for you but it is probably not true
  - Find same patterns in gray zone
  - Leverage from unique style

Demo application

- JavaTwo conference ☺
- GIT: https://github.com/martinjmares/javatwo-jaxrs-angular-tools-demo.git
- Each tag represents one demo step
  - **It is not incremental**

Basics: JAX-RS and AngularJS

JAX-RS

https://jcp.org/en/jsr/detail?id=339

- @Path
  - @ApplicationPath, resource @Path, method @Path
  - {} path template
- @GET, @POST, @PUT, @DELETE, @HEAD
- IoC
  - @PathParam, @QueryParam, @FormParam, @HeaderParam, @CookieParam, @BeanParam, @MatrixParam
  - Entity
  - CDI

JAX-RS

https://jax-rs-spec.java.net/nonav/2.0/apidocs/

- @Consumes, @Produces
- @Provider
  - MessageBodyReader, MessageBodyWriter
  - Response.ResponseBuilder
- Return resource
  - Cool combination with @MatrixParam

AngularJS

https://angularjs.org/

- Imports (modules)
  - angular.js – ng: Core
  - angular-route.js - ngRoute: Navigation
  - angular-resource.js - ngResource: REST
- Bootstrap
  - ng-app – first **directive**
  - angular.module – first create module for your application
- View - template
  - Directive and expressions (https://docs.angularjs.org/guide/expression)

AngularJS

https://docs.angularjs.org/api

- Controller
  - Bind using directive
  - Bind using routing (navigation)
- Model in Scopes
  - Hierarchical structure – each controller has one
- Injection
- Module assembly

AngularJS

Modules

- ngRoute
  - $routeProvider
  - $routeParams
  - $location
- ngResource
- Module.factory

Day by day TOI

Simple documentation

WADL

- Jersey by default WADL
  - XSD? – No JSON schema!
  - Still helpful
  - No additional information (comments, javadoc)
- Jersey doclet
  - org.glassfish.jersey.wadl.doclet.ResourceDoclet
- Jersey maven-wadl-plugin
  - Still in contrib of Jersey 1

WADL tools

- XSLT for documentation
  - https://github.com/mnot/wadl_stylesheets.git
  - https://github.com/ipcsystems/wadl-stylesheet.git
- SWADL
  - https://github.com/ehearty/Swadl.git

Swagger

https://helloreverb.com/developers/swagger

- Large ecosystem
  - Build / runtime generation
  - Framework integration
- No doclet – driven by own annotations
  - @Api, @ApiOperation, @ApiParam, @ApiResponse
  - @ApiModel, @ApiModelProperty
- Generates own description – swagger.json
  - Can be viewed by swagger-ui

Swagger - generation

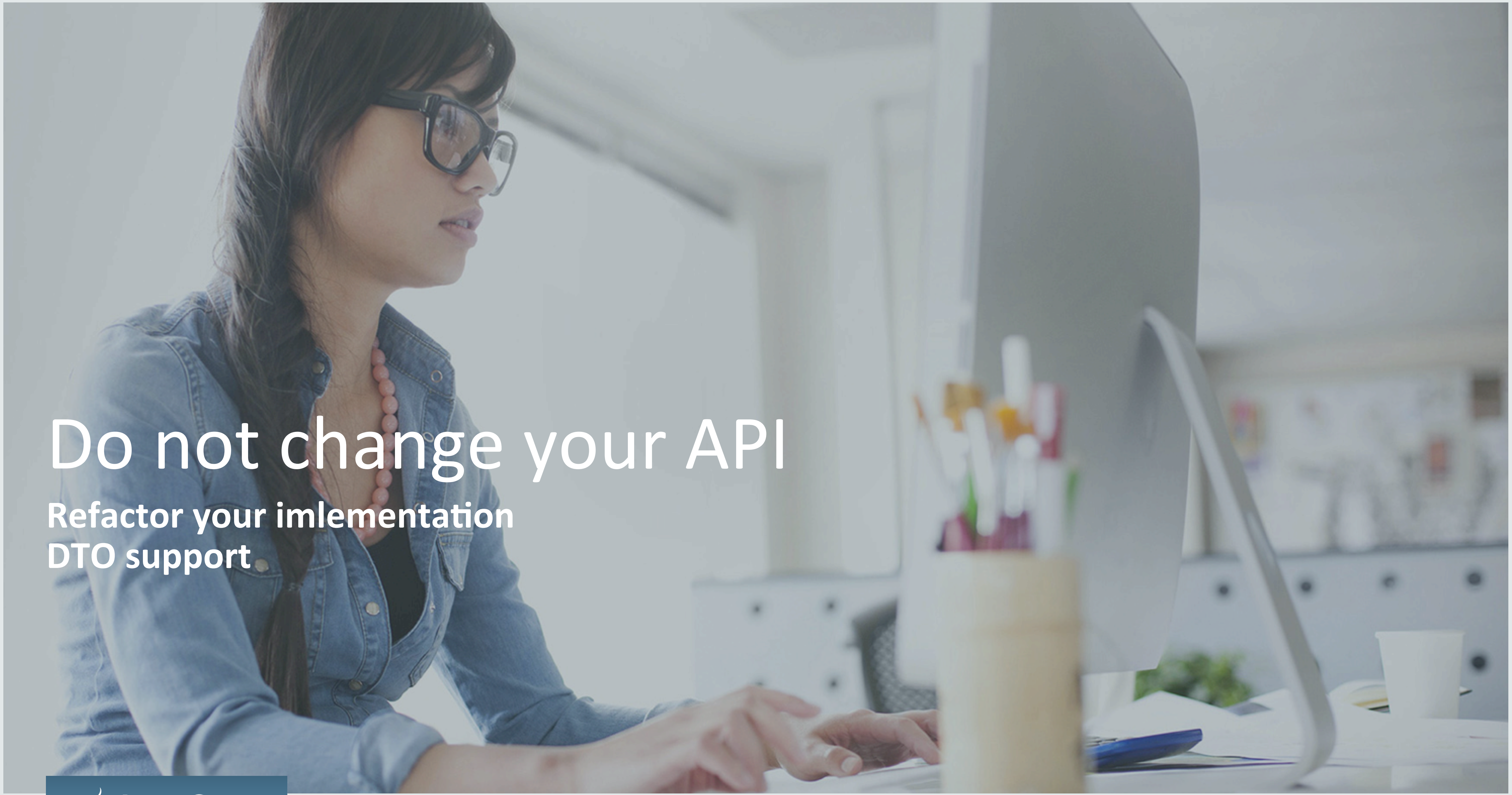
- Build time (https://github.com/kongchen/swagger-maven-plugin.git)
  - swagger-maven-plugin
  - Documentatio & aplication :: swaggerUIDocBasePath & basePath
    - Must support CORS
  - Supports mustache templates (https://github.com/kongchen/api-doc-template.git)
- JAX-RS support - swagger-jaxrs_2.10
  - Maven dependence
  - JAX-RS application must register it
  - Web.xml defines basic configuration
  - https://github.com/wordnik/swagger-core/wiki/Java-JAXRS-Quickstart

JavaOne™ ORACLE

Enunciate

http://enunciate.codehaus.org/

- Fast and easy
  - Low control
  - Jersey 1
  - Supports a lot of frameworks
- Easy
  - One dependence: enunciate-rt
  - One plugin: maven-enunciate-plugin



Do not change your API

Refactor your imlementation
DTO support

Dozer

http://dozer.sourceforge.net/

- Pros
  - Very known
  - Feature rich (mostly in convertors)
  - GUI for eclipse
- Cons
  - Slow
  - Primary XML base configuration
    - Refactoring issue, edit issue (who use Eclipse? ☺)
    - Annotation support is limited and configuration API is undocumented
    - **Very different thinking that is my nature**

Orika

http://orika-mapper.github.io/orika-docs/index.html

- Pros
  - Somehow similar Dozer
  - Generates pipelines – fast
- Cons
  - Only accessible properties

- GeDA (http://www.genericdtoassembler.org/)
  - Generates pipelines same as Orika – but faster
  - Support for enterprise technologies (Spring, OSGI)

Entity filtering

Just another jersey trick

- Annotate field and it will be filtered
  - Add dependency on jersey-entity-filtering
  - Register EntityFilteringFeature.class in your rest application
  - Create own annotation and annotate it with @EntityFiltering
  - Annotate fields which should be filtered out by default
  - Annotate your resource method which will not filter annotated fields

- Check also security and dynamic entity filtering

The end for today

Well, almost

Few more notes

- Check the future
  - JAX-RS 2.1: https://jcp.org/en/jsr/detail?id=370
  - MVC 1.0: https://jcp.org/en/jsr/detail?id=371
- Check Jasmine (http://jasmine.github.io/)
  - Test if your JSONs still fits to frontend application
- Do not force javascript developers to Maven or Gradle
  - Best is to separate backend and frontend
  - If not possible still use node for frontend support (and just execute it)
    - Npm, Grunt, Bower, Yeoman

WHERE IS DEMO

https://github.com/martinjmares/javatwo-jaxrs-angular-tools-demo.git

JavaOne™

ORACLE®

ORACLE®