Josh Long (龙之春)
@starbuxman
joshlong.com
josh@joshlong.com
slideshare.net/joshlong
github.com/joshlong
speakerdeck.com/joshlong

BUILDING REST SERVICES WITH

Spring

github.com/joshlong/**the-spring-rest-stack**

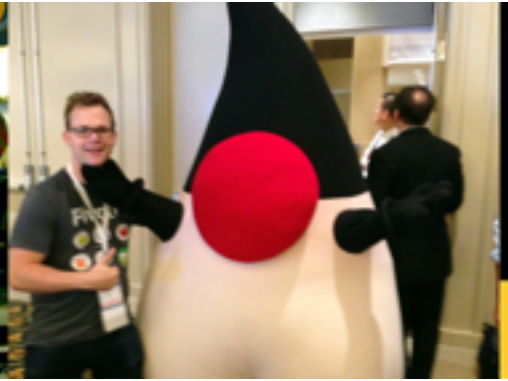
About Josh Long (龙之春)

Spring Developer Advocate, Pivotal

@starbuxman
josh@joshlong.com
slideshare.net/joshlong
github.com/joshlong
speakerdeck.com/joshlong

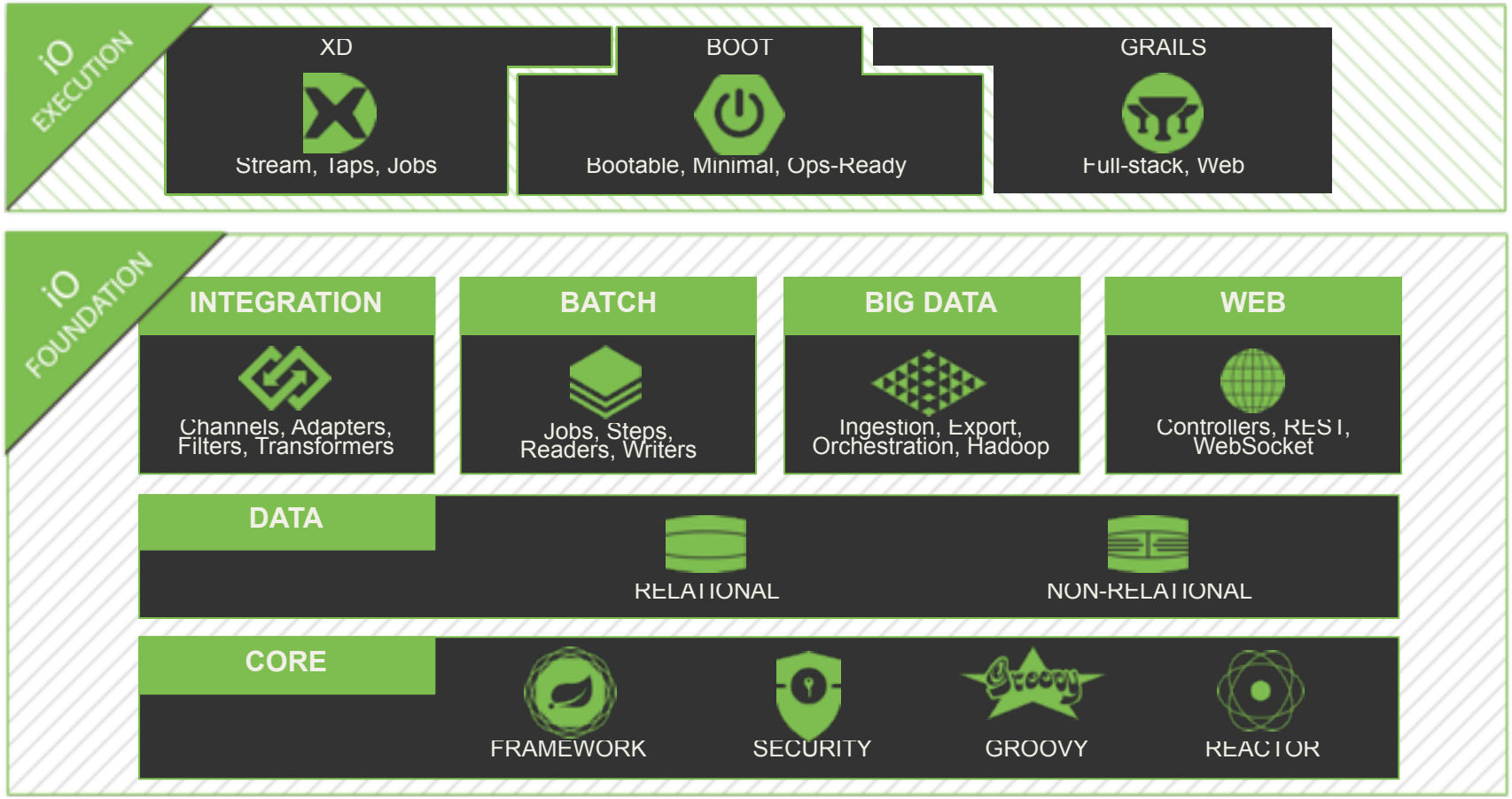
Jean Claude van Damme!

Java mascot Duke



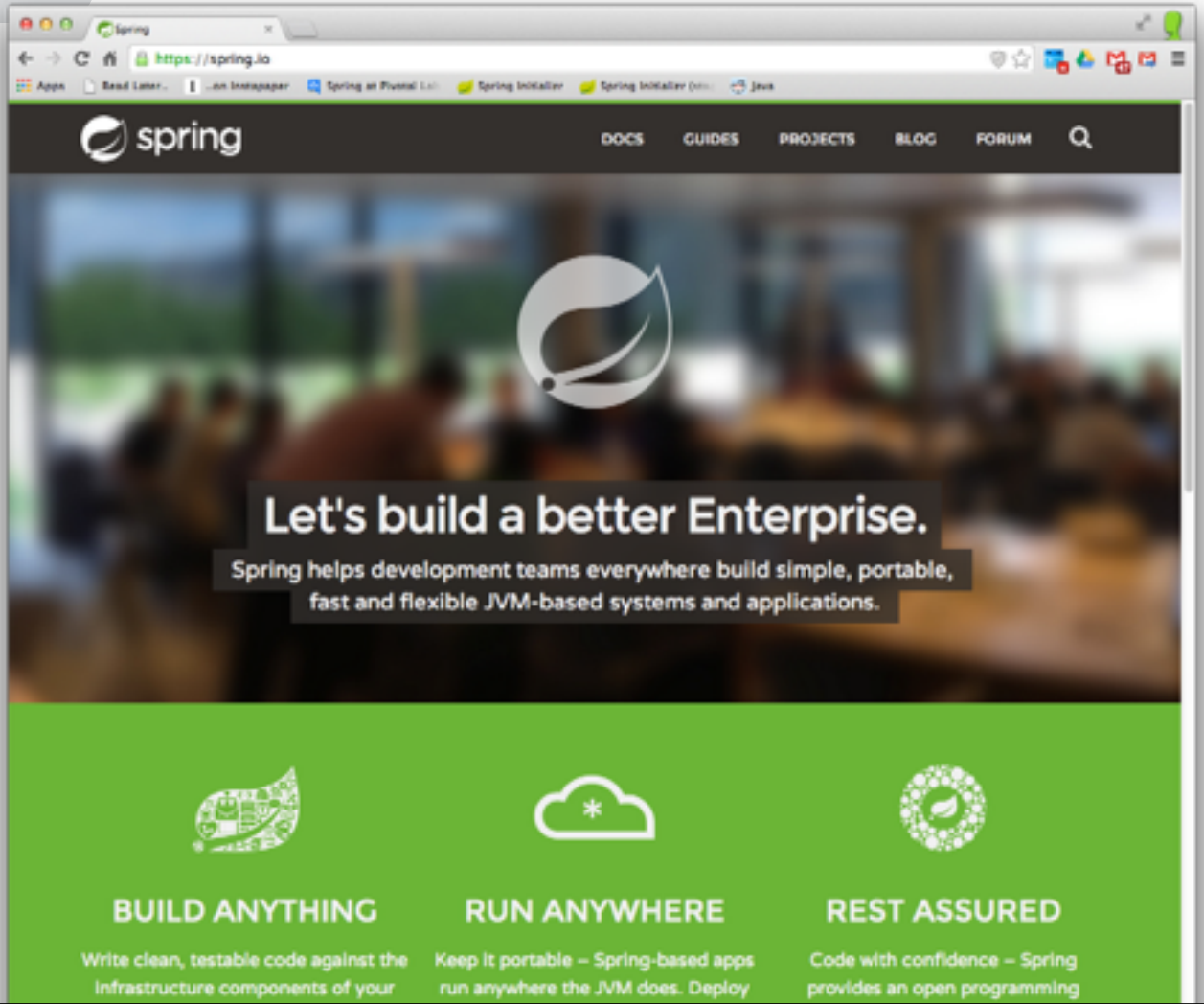
some thing's I've authored...

Starting with Spring


iO EXECUTION
XD
Stream, Taps, Jobs
BOOT
Bootable, Minimal, Ops-Ready
GRAILS
Full-stack, Web
iO FOUNDATION
INTEGRATION
Channels, Adapters, Filters, Transformers
BATCH
Jobs, Steps, Readers, Writers
BIG DATA
Ingestion, Export, Orchestration, Hadoop
WEB
Controllers, REST, WebSocket
DATA
RELATIONAL
NON-RELATIONAL
CORE
FRAMEWORK
SECURITY
GROOVY
REACTOR

A NEW HOME FOR SPRING

GITHUB.COM/JOSHLONG/**THE-SPRING-REST-STACK**



A NEW HOME FOR SPRING

GITHUB.COM/JOSHLONG/**THE-SPRING-REST-STACK**

https://spring.io/guides

spring

DOCS | GUIDES | PROJECTS | BLOG | FORUM

Guides

Whatever you're building, these guides are designed to get you productive as quickly as possible – using the latest Spring project releases and techniques as recommended by the Spring team.

Have a suggestion for a new guide? Let us know at @springcentral.

Find a guide...

Getting Started Guides

Designed to be completed in 15-30 minutes, these guides provide quick, hands-on instructions for building the "Hello World" of any development task with Spring. In most cases, the only prerequisites are a JDK and a text editor.

Building a RESTful Web Service
Learn how to create a RESTful web service with Spring.

Scheduling Tasks
Learn how to schedule tasks with Spring.

Detecting a Device
Learn how to use Spring to detect the type of device that is accessing your web site.

Consuming a RESTful Web Service
Learn how to retrieve web page data with Spring's RestTemplate.

Building Java Projects with Gradle
Learn how to build a Java project with Gradle.

Building Java Projects with Maven
Learn how to build a Java project with Maven.

Accessing Relational Data using JDBC with Spring
Learn how to access relational data with Spring.

Consuming a RESTful Web Service with Spring for Android
Learn how to retrieve web page data using Spring for Android's RestTemplate.

Uploading Files
Learn how to build a Spring application that accepts multi-part file uploads.

Authenticating a User with LDAP

Registering an Application with

Building Android Projects with Maven

websockets : supports JSR 356, native APIs

Async `RestTemplate`
based on NIO 2 HTTP client in JDK.

Java SE 8 and Java EE 7 extends support to emerging platforms

@Conditional provides the ability to conditionally create a bean

```
@Conditional (NasdaqIsUpCondition.class)
@Bean
 Mongo extraMongoNode(){

  // ...

}
```

And, best of all, **@Conditional** powers **Spring Boot**!

single point of focus, production-ready, easy to customize

Installation:

> Java 1.6 or better

> Maven 3.0 or better

> optionally install `spring` **CLI** (or `gvm` or `brew`)

Demonstration

Take Spring Boot CLI for
a spin around the block

Demonstration

Take Spring Boot around the track.

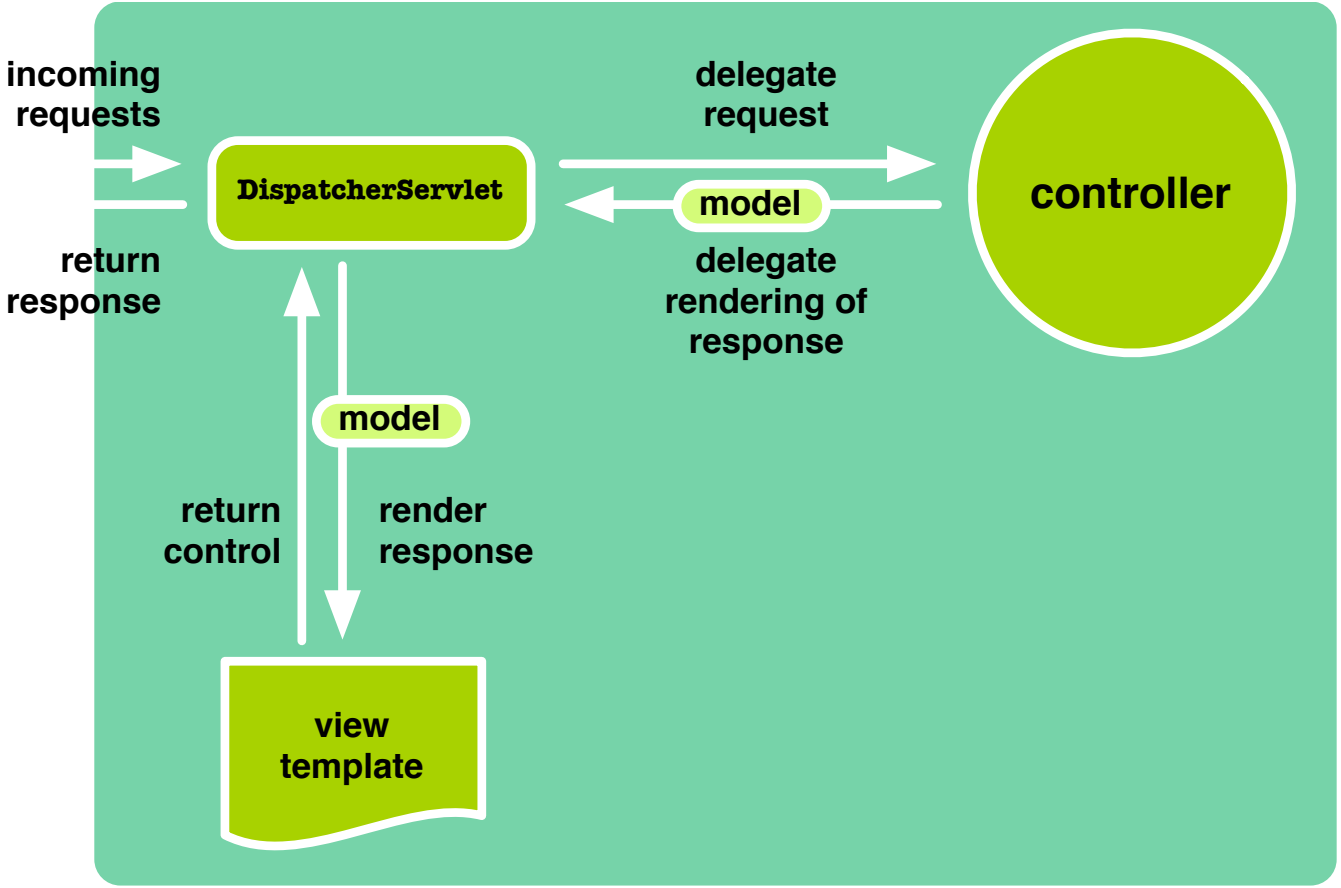
Testing

Demonstration

how to write unit tests with Spring

Spring MVC

stop me if you've heard this one before...



web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
            http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
  <distributable/>
  <listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
  </listener>
  <context-param>
    <param-name>contextInitializerClasses</param-name>
    <param-value>my.ApplicationContextInitializer</param-value>
  </context-param>
  <context-param>
    <param-name>contextClass</param-name>
    <param-value>org.springframework.web.context.support.AnnotationConfigWebApplicationContext
    </param-value>
  </context-param>
  <servlet>
    <servlet-name>appServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
      <param-name>contextConfigLocation</param-name>
      <param-value></param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>appServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>
</web-app>
```

WebApplicationInitializer ~= Java web.xml

```
public class SampleWebApplicationInitializer implements WebApplicationInitializer {

    public void onStartup(ServletContext sc) throws ServletException {

            AnnotationConfigWebApplicationContext ac =  new AnnotationConfigWebApplicationContext();
            ac.setServletContext(sc);
            ac.scan( “a.package.full.of.services”, “a.package.full.of.controllers” );

            sc.addServlet("spring", new DispatcherServlet(ac));

            // register filters, other servlets, etc., to get Spring and Spring Boot working

    }
}
```

or, just fill out the form...

```
public class SimplerDispatcherServletInitializer
      extends AbstractAnnotationConfigDispatcherServletInitializer {

   @Override
   protected Class<?>[] getRootConfigClasses() {
      return new Class<?>[]{ ServiceConfiguration.class };
   }

   @Override
   protected Class<?>[] getServletConfigClasses() {
      return new Class<?>[]{ WebMvcConfiguration.class  };
   }

   @Override
   protected String[] getServletMappings() {
      return new String[]{"/*"};
   }
}
```

or, **just use Spring Boot** and never worry about it

```
@ComponentScan
@EnableAutoConfiguration
public class Application extends SpringBootServletInitializer {

   private static Class< Application> applicationClass = Application.class;

   public static void main(String[] args) {
      SpringApplication.run(applicationClass);
   }

   @Override
   protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
      return application.sources(applicationClass);
   }
}
```

other niceties Spring's web support provides:

HttpRequestHandlers supports remoting technologies : Caucho, HTTP Invoker, etc.

DelegatingFilterProxy javax.filter.Filter that delegates to a Spring-managed bean

HandlerInterceptor wraps requests to **HttpRequestHandlers**

ServletWrappingController lets you force requests to a servlet through the Spring Handler chain

WebApplicationContextUtils look up the current **ApplicationContext** given a **ServletContext**

HiddenHttpMethodFilter routes HTTP requests to the appropriate endpoint

REST Essentials


meanwhile, in the enterprise,
somebody is using **SOAP**
because it's **"SIMPLE"**

REST is an architectural constraint based on HTTP 1.1, and created as part of Roy Fielding's doctoral dissertation in 2000.

It embraces HTTP.

It's a style, *not* a standard

http://en.wikipedia.org/wiki/Representational_state_transfer

REST has no hard and fast rules.

REST is an architectural **style**, not a standard.

REST uses **`Headers`** to describe requests & responses

REST embraces HTTP verbs. (DRY)

GET requests retrieve information.

GET can have side-effects (but it's unexpected)

GET can be conditional, or partial:

If-Modified-Since, Range

GET /users/21

DELETE requests that a resource be removed, though the deletion doesn't have to be immediate.

DELETE /users/21

POST requests that the resource *do something* with the enclosed entity

POST can be used to **create** or **update**.

POST /users
{ “firstName”: “Juergen” }

PUT requests that the entity be stored at a URI

PUT can be used to **create** or **update**.

```
PUT /users/21
{ "firstName": "Juergen" }
```

status codes convey the result of the server's attempt to satisfy the request.

Categories:

- **1xx**: informational
- **2xx**: success
- **3xx**: redirection
- **4xx**: client error
- **5xx**: server error

200 OK - Everything worked

201 Created - Returns a **Location** header for new resource

202 Accepted - server has accepted the request, but it is not yet complete. Status URI optionally conveyed in **Location** header

400 Bad Request - Malformed Syntax. Retry with change.

401 Unauthorized - authentication is required

403 Forbidden - server has understood, but refuses request

404 Not Found - server can't find a resource for URI

406 Incompatible - incompatible **Accept** headers specified

409 Conflict - resource conflicts with client request

Clients and services must agree on a representation media type through **content negotiation.**

Client specifies what it wants through **`Accept`** header

Server specifies what it produces through **`Content-Type`** header


If no match is made,
the client will receive a
406 Not Acceptable

Spring MVC supports multiple types of content negotiation through its **`ContentNegotiationStrategy`**:

- e.g., **`Accept`** header, URL extension, request parameters, or a fixed type

Advanced REST Client

Advanced Rest Client App

chrome-extension://hgmloofddffdnphfgcellkdfbfbjeloo/RestClient.html

Apps | Read Later | ...on Instapaper | Blog Admin | H2 Console | SpringSourceDev | @SpringCentral | Java | SpringSource on Fa | Other Bookmarks

Advanced Rest Client

Request
Socket
Projects
Saved
History
Settings
About

Rate this application
Donate

[Unnamed] Save Open

http://localhost:8080/users/5/

GET POST PUT PATCH DELETE HEAD OPTIONS Other

Raw Form Headers

Accept: application/json

Clear Send

Status 200 OK Loading time: 39 ms

Request headers
Accept: application/json
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/32.0.1700.107 Safari/537.36
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8,fr;q=0.6
Cookie: JSESSIONID=72902CE01012EF0BEF679487926ADA91

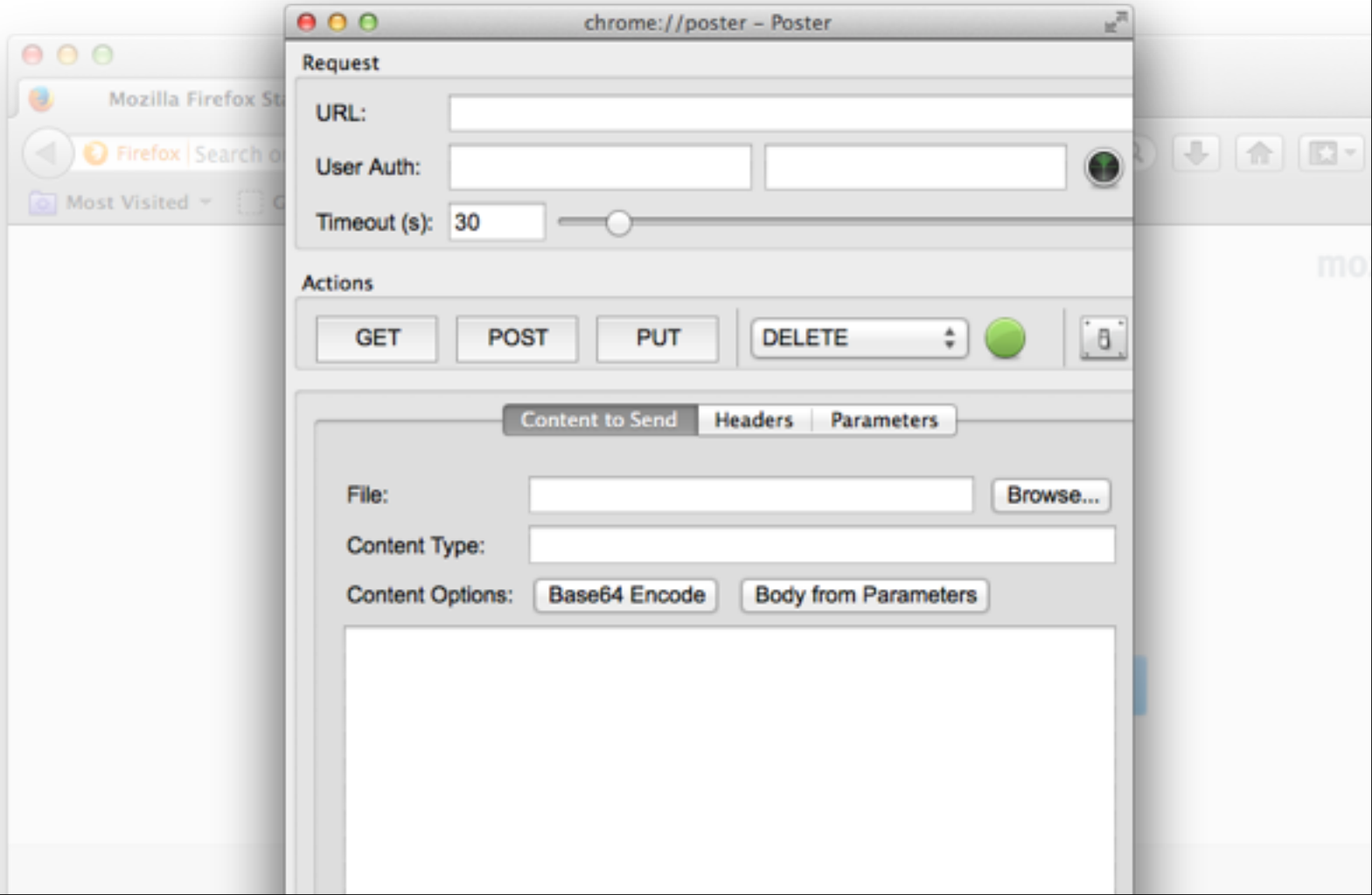
Response headers
Server: Apache-Coyote/1.1
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Thu, 20 Feb 2014 09:14:47 GMT

Raw JSON Response

Copy to clipboard Save as file


```
{
  id: 5
  firstName: "Josh"
  profilePhotoMediaType: "image/png"
  lastName: "Long"
  username: "joshlong"
  password: null
  profilePhotoImported: true
  enabled: true
  signupDate: 1378213431000
  _links: {
    self: {
      href: "http://localhost:8080/users/5"
    }
    customers: {
      href: "http://localhost:8080/users/5/customers"
    }
```

Poster


chrome://poster – Poster
Request
URL:
User Auth:
Timeout (s):
30
Actions
GET
POST
PUT
DELETE
Content to Send
Headers
Parameters
File:
Browse...
Content Type:
Content Options:
Base64 Encode
Body from Parameters
Mozilla Firefox St
Firefox
Search o
Most Visited

curl

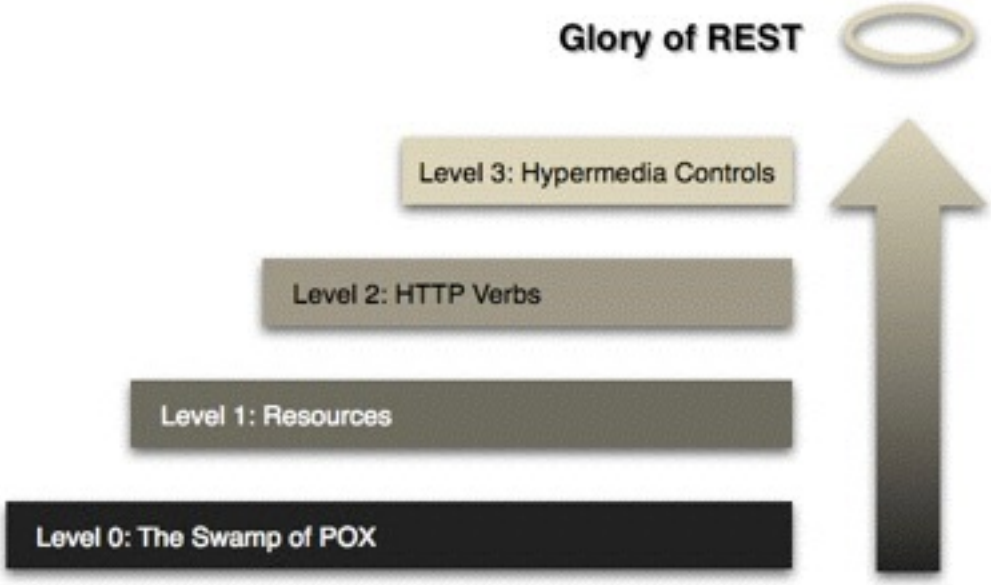
```
➜ ~ curl -X POST -u android-crm:123456 http://localhost:8080/oauth/token \
  -H "Accept: application/json" \
  -d "password=......"

{"access_token":"426481ea-c3eb-45a0-8b2d-d1f9cfae0fcc","token_type":"bearer","expire

➜ ~
```

Towards Hypermedia

The Richardson Maturity Model is a way to grade your API according to the REST constraints with 4 levels of increasing compliance



http://martinfowler.com/articles/richardsonMaturityModel.html

The Richardson Maturity Model

Level 0: swamp of POX

Uses **HTTP** mainly as a tunnel through one URI
e.g., **SOAP**, **XML-RPC**

Usually features on HTTP verb (**`POST`**)

http://martinfowler.com/articles/richardsonMaturityModel.html

The Richardson Maturity Model

Level 1: resources

Multiple URIs to distinguish related nouns

e.g., **`/articles/1`**, **`/articles/2`**, vs. just **`/articles`**

http://martinfowler.com/articles/richardsonMaturityModel.html

The Richardson Maturity Model

Level 2: HTTP verbs

leverage transport-native properties to enhance service
e.g., **HTTP GET** *and* **PUT** *and* **DELETE** *and* **POST**

Uses idiomatic HTTP controls like status codes, headers

http://martinfowler.com/articles/richardsonMaturityModel.html

Demonstration

Our first @RestController

The Richardson Maturity Model

Level 3: Hypermedia Controls (aka, **HATEOAS**)

No *a priori* knowledge of service required
Navigation options are provided by service and *hypermedia* controls

Promotes longevity through a *uniform interface*

http://martinfowler.com/articles/richardsonMaturityModel.html

Links provide possible navigations from a given resource

Links are dynamic, based on resource state.

```
<link href="http://...:8080/users/232/customers"
      rel= "customers"/>

{ href: "http://...:8080/users/232/customers",
  rel: "customers" }
```

Demonstration

Working with Hypermedia and
Spring HATEOAS

Spring Data REST simplifies the generic data-centric **@Controller**s

Builds on top of Spring Data Repository support:

```
@RestResource (path = "users", rel = "users")
public interface UserRepository extends PagingAndSortingRepository<User, Long> {
    User findByUsername(@Param ("username") String username);
```

Spring Data REST simplifies the generic data-centric **@Controller**s

Builds on top of Spring Data Repository support:

```
@RestResource (path = "users", rel = "users")
public interface UserRepository extends PagingAndSortingRepository<User, Long> {
    User findByUsername(@Param ("username") String username);
        select u from User where u.username = ?
```

Spring Data REST simplifies the generic data-centric **@Controller**s

Builds on top of Spring Data Repository support:

```
@RestResource (path = "users", rel = "users")
public interface UserRepository extends PagingAndSortingRepository<User, Long> {

    List<User> findUsersByFirstNameOrLastNameOrUsername(
        @Param ("firstName") String firstName,
        @Param ("lastName") String lastName,
        @Param ("username") String username);
}
```

Spring Data REST simplifies the generic data-centric **@Controller**s

Builds on top of Spring Data Repository support:

```
@RestResource (path = "users", rel = "users")
public interface UserRepository extends PagingAndSortingRepository<User, Long> {

    List<User> findUsersByFirstNameOrLastNameOrUsername(
        @Param ("firstName") String firstName,
        @Param ("lastName") String lastName,
        @Param ("username") String username);
}
```

```
select u from User u
where u.username = ?
or u.firstName = ?
or u.lastName = ?
```

Testing REST

Demonstration

Testing web services with
Spring MVC Test framework

Error Handling

Developers learn to use an API through errors

Extreme programming and Test-Driven development embrace this truth

Errors introduce transparency

Status codes map to errors

pick a meaningful subset of the 70+ status codes

200 - OK
201 - Created
304 - Created - Not Modified
400 - Bad Request
401 - Unauthorized
403 - Forbidden
404 - Not Found
500 - Internal Server Error

```
public enum HttpStatus {

    // 1xx Informational

    /**
     * {@code 100 Continue}.
     * @see <a href="http://tools.ietf.org/html/rfc26
     */
    CONTINUE(100, "Continue"),
    /**
     * {@code 101 Switching Protocols}.
     * @see <a href="http://tools.ietf.org/html/rfc26
     */
    SWITCHING_PROTOCOLS(101, "Switching Protocols"),
    /**
     * {@code 102 Processing}.
     * @see <a href="http://tools.ietf.org/html/rfc25
     */
    PROCESSING(102, "Processing"),
    /**
     * {@code 103 Checkpoint}.
     * @see <a href="http://code.google.com/p/gears/w
     * resumable POST/PUT HTTP requests in HTTP/1.0</
     */
    CHECKPOINT(103, "Checkpoint"),
```

https://blog.apigee.com/detail/restful_api_design_what_about_errors

Send meaningful errors along with status codes

```
{
  "message": "authentication failed",
  "errors": [
    {
      "resource": "Issue",
      "field": "title",
      "code": "missing_field"
    }
  ]
}
```

```
{
  "type": "authentication",
  "message": “the username and
password provided are invalid” ,
  "status": “401”
}
```

https://blog.apigee.com/detail/restful_api_design_what_about_errors

application/vnd.error+json & application/vnd.error+xml

```
{
  "logref": 42,
  "message": "Validation failed",
  "_links": {
    "help": {
      "href": "http://.../", "title": "Error Information"
    },
    "describes": {
      "href": "http://.../", "title": "Error Description"
    }
  }
}
```

https://github.com/blongden/vnd.error

Demonstration

Handling errors with `vnd.errors` and `@ControllerAdvice`

Demonstration

Using @ControllerAdvice

API Versioning

Build a version into your API

API versions can be dealt with one of two ways:

through API URIs: `https://api.foo.com/v1`

through media types: `application/vnd.company.urapp-v3+json`

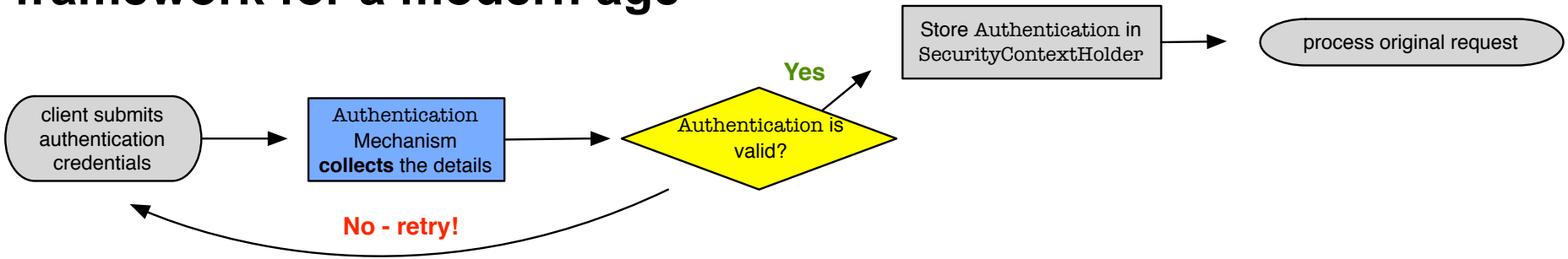
Security

Security is hard. Don't reinvent the wheel!

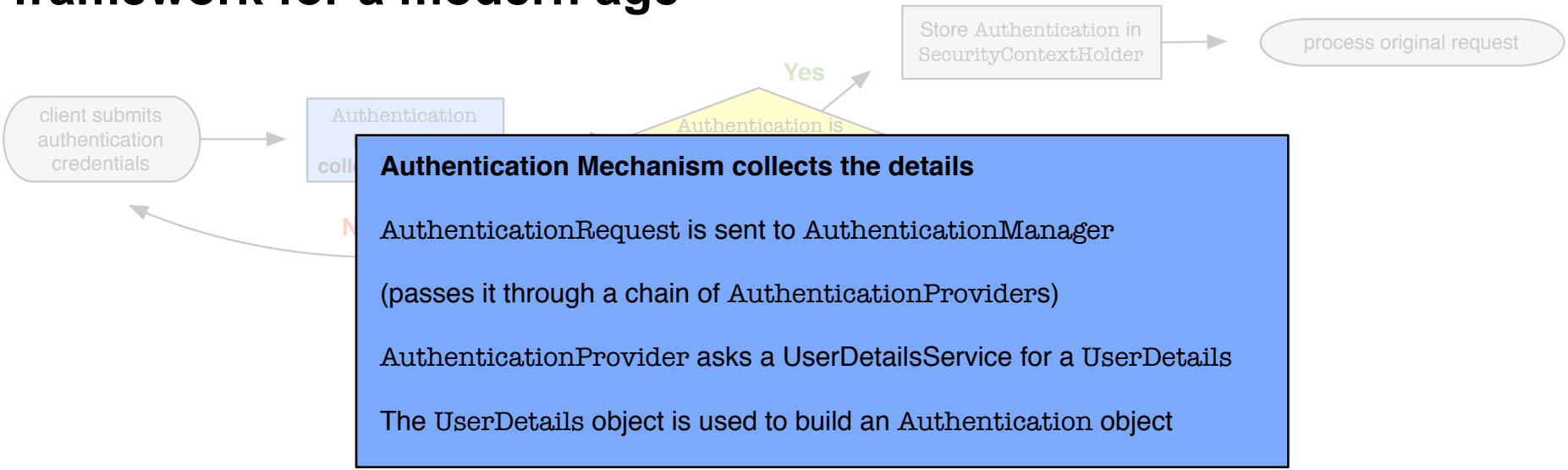
Things to worry about when developing web applications? **EVERYTHING**

(cross-site scripting, session fixation, identification, authorization, and authentication, encryption, and SO much more.)

Spring Security is a modern security framework for a modern age



Spring Security is a modern security framework for a modern age



Authentication Mechanism collects the details

AuthenticationRequest is sent to AuthenticationManager

(passes it through a chain of AuthenticationProviders)

AuthenticationProvider asks a UserDetailsService for a UserDetails

The UserDetails object is used to build an Authentication object

Demonstration

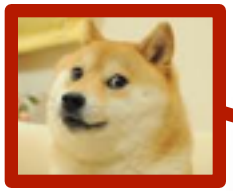
adding a Spring Security sign in form to a regular application

Usernames and Passwords

If you can **trust the client** to keep a secret like a password, then it can send the password using:

...**HTTP Basic** - passwords are sent plaintext!
... **HTTP Digest** - hashed passwords, but still plaintext.

SSL/TLS encryption helps prevent man-in-the-middle attacks



so trust!

wow

So, **SSL/TLS** is...**?**

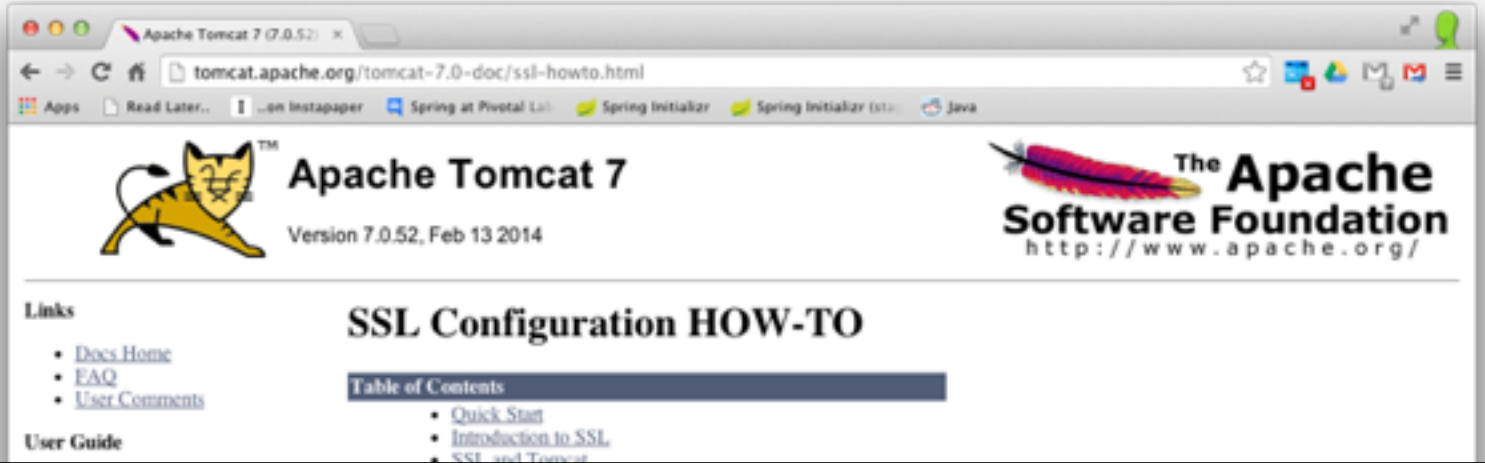
an implementation of **public key cryptography:**

public key cryptography only works because we all agree to trust well known root CAs

SSL/TLS is used routinely to verify the identify of servers.

Normally, the client confirms the server, but the server rarely requires the client to transmit a certificate.

It's easy enough to setup **SSL/TLS** on your web server.



Demonstration

Setting up SSL/TLS with embedded Apache Tomcat 7 and Spring Boot

SSL/TLS can be used to identify the client to the server, through *mutual authentication.*

browser/client must send their certificate, as well.

```
@Override
protected void configure(HttpSecurity http)
   throws Exception {
   http
      .authorizeRequests()
         .anyRequest().authenticated()
         .and()
      .x509();
}
```

```
@Configuration
@EnableWebMvcSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

  @Autowired
  public void configureGlobal(AuthenticationManagerBuilder auth)
      throws Exception {
    auth.
      inMemoryAuthentication()
        .withUser("mia").password("password").roles("USER").and()
        .withUser("mario").password("password").roles("USER","ADMIN");
  }

  @Override
  protected void configure(HttpSecurity http) throws Exception {
    http
      .authorizeRequests()
        .anyRequest().authenticated()
        .and()
      .x509();
  }
}
```

Demonstration

X509 Java configuration demo

Tim Bray says: Passwords don’t scale

Too easy to compromise.

Updating all your clients whenever you change your password would be a nightmare!

programmableweb

Hot APIs » Twitter YouTube Facebook Google Maps Flickr LinkedIn More » Latest news Today i

Home API News API Directory Mashups Community How-to

Keeping you up to date with APIs, mashups and the Web as platform. Learn more »

Find APIs, mashups, code and developers Search

Popular searches: photo google flash mapping enterprise sms

9082 APIs

7051 Mashups

New APIs

- Zapier Status

Mashup of the Day

New Mashups

- Next DC Metro

Most people just want their own clients to be able to talk securely to their own services.

x-auth offers one way of achieving this based on tokens

Demonstration

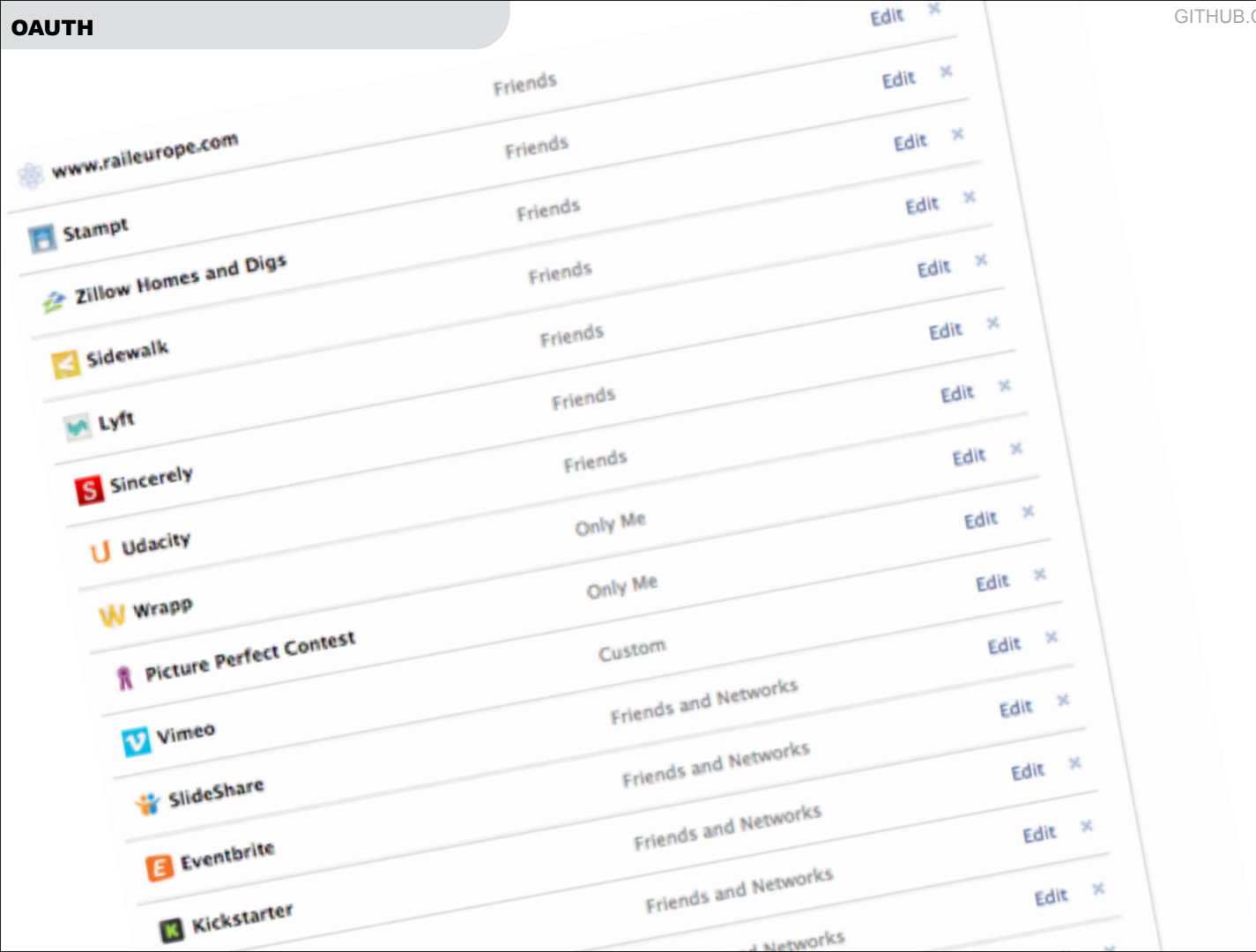
A custom x-auth example

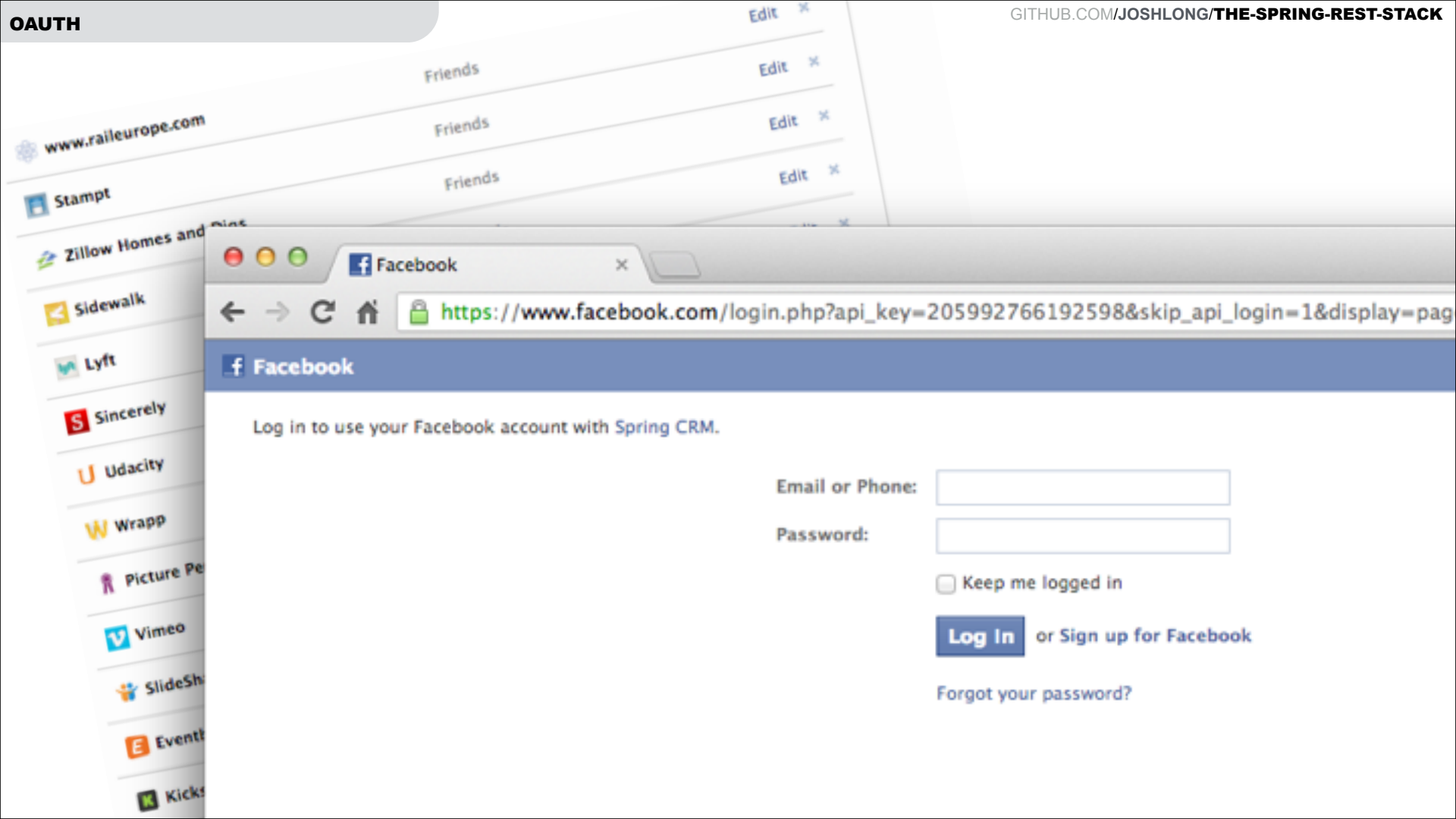
OAuth is a way for one (automated) process to securely identify itself to another

Assumes a user context:

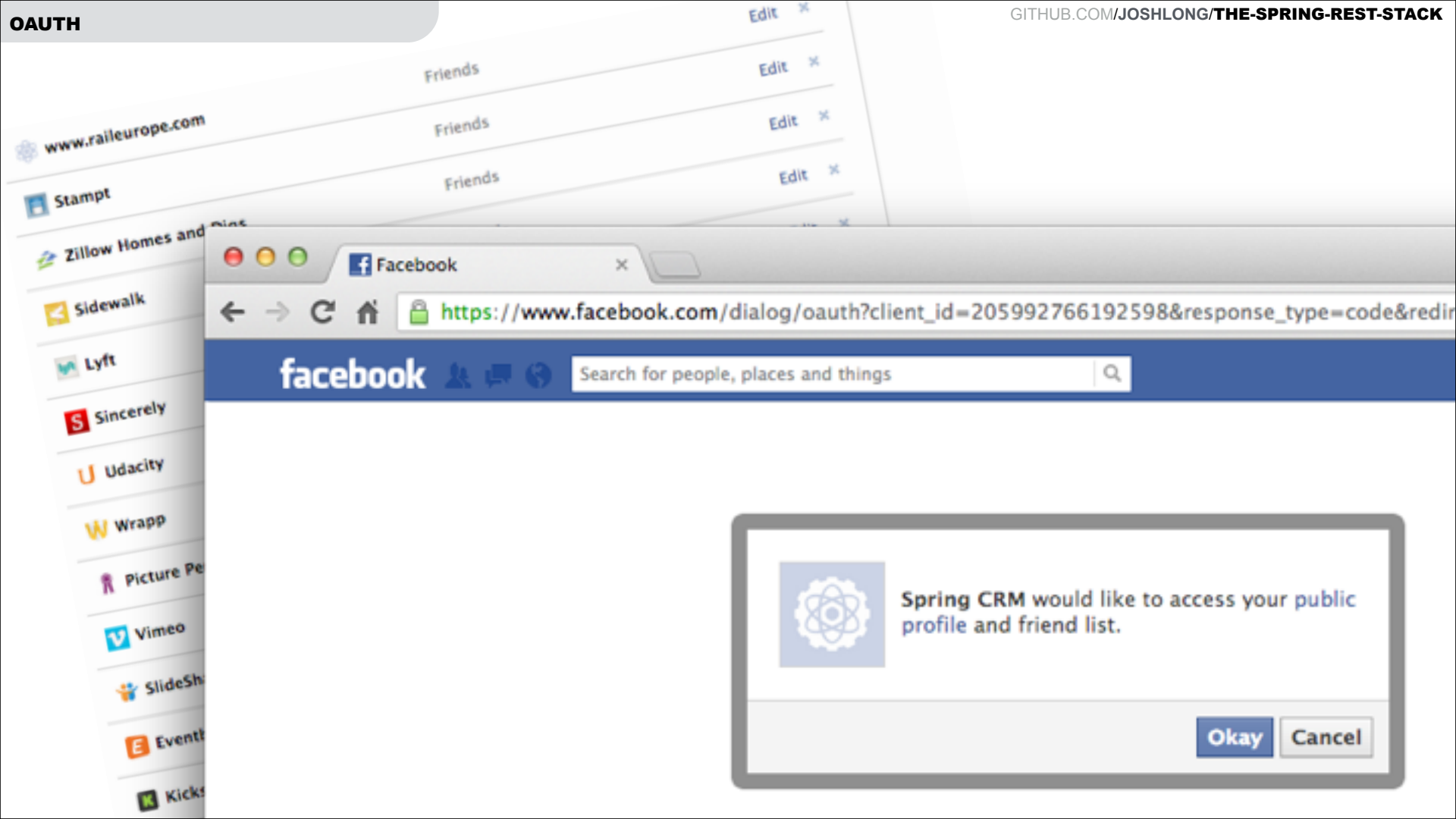
> "I authorize `$CLIENTX` to act on `$USER_Y`'s behalf"

OAuth is a way of authorizing a client with particular access (*scopes*)


Edit
www.raileurope.com
Friends
Edit
Stampt
Friends
Edit
Zillow Homes and Digs
Friends
Edit
Sidewalk
Friends
Edit
Lyft
Friends
Edit
Sincerely
Friends
Edit
Udacity
Only Me
Edit
Wrapp
Only Me
Edit
Picture Perfect Contest
Custom
Edit
Vimeo
Friends and Networks
Edit
SlideShare
Friends and Networks
Edit
Eventbrite
Friends and Networks
Edit
Kickstarter
Friends and Networks
Edit


Edit
Friends
Edit
www.raileurope.com
Friends
Edit
Stampt
Friends
Edit
Zillow Homes and
Sidewalk
Lyft
Sincerely
Udacity
Wrapp
Picture Pe
Vimeo
SlideSh
Eventb
Kicks
Facebook
https://www.facebook.com/login.php?api_key=205992766192598&skip_api_login=1&display=pag
Facebook
Log in to use your Facebook account with Spring CRM.
Email or Phone:
Password:
Keep me logged in
Log In
or Sign up for Facebook
Forgot your password?

OAUTH


Edit
Friends
Edit
www.raileurope.com
Friends
Edit
Stampt
Friends
Edit
Zillow Homes and
Sidewalk
Lyft
Sincerely
Udacity
Wrapp
Picture Pe
Vimeo
SlideSh
Eventb
Kicks
Facebook
https://www.facebook.com/dialog/oauth?client_id=205992766192598&response_type=code&redir
facebook
Search for people, places and things
Spring CRM would like to access your public profile and friend list.
Okay
Cancel

Demonstration

Spring Security OAuth in the `oauth` module

Demonstration

Writing a unit test for an OAuth service using the Spring MVC test framework

The Connected Web of APIs

*** source:** visual.ly/60-seconds-social-media

A Connected World in 00:60 seconds

facebook **700k** messages sent

Pinterest **1090** visitors

foursquare **2000** checkins

twitter **175k** tweets

LinkedIn **7610** searches

YouTube **2MM** videos viewed

flickr **3125** photos uploaded

StumbleUpon **7630** messages sent

Spring Social provides an authentication and authorization client for OAuth (1.0, 1.0a, 2.0)

Provides type-safe API bindings for various services

SPRING SOCIAL BINDINGS

GitHub

Tripit

Organize your travel

in

twitter

facebook

Api Providers - spring-pro

GitHub, Inc. [US] https://github.com/spring-projects/spring-social/wiki/api-providers

Apps | Read Later... | ...on Instapaper | Blog Admin | H2 Console | SpringSourceDev on | @SpringCentral | Java | SpringSource on Fac | Pivotal Command C | Other Bookmarks

This repository | Search or type a command | Explore | Gist | Blog | Help | joshlong

PUBLIC spring-projects / spring-social

Unwatch 92 | Star 339 | Fork 154

Home | Pages | History | New Page

Api Providers

Edit Page | Page History | Clone URL

The web is full of APIs that Spring Social can provide connection support and API bindings for. What follows is a list of known APIs, what authorization scheme they use, and (if available) a link to an existing Spring Social extension for that API.

If there is an API you're interested in that does not yet have a Spring Social extension associated with it, consider it an opportunity to contribute to the Spring Social community by creating one. Be sure to let us know where your project resides and give us a heads up on the Spring Social Forum.

Provider	Authorization	Spring Social Project
500px	OAuth 1.0a	https://github.com/sachin-handiekar/spring-social-500px
AllPlayers.com	OAuth 1.0	
App.net	OAuth 2	https://github.com/arikg/spring-social-appdotnet
Basecamp/37 Signals	OAuth 2	
Bebo	OAuth-like	

Demonstration

Using Spring Social in an Application

Demonstration

Building Your own Spring Social binding

Deployment

Micro Services ...

Promote single responsibility principle *

Promote loosely coupled, focused services.
(**SOLID** at the architecture level)

Don't like it? Throw it away!

* In object-oriented programming, the **single responsibility principle** states that every class should have a single responsibility, and that responsibility should be entirely encapsulated by the class. All its services should be narrowly aligned with that responsibility.

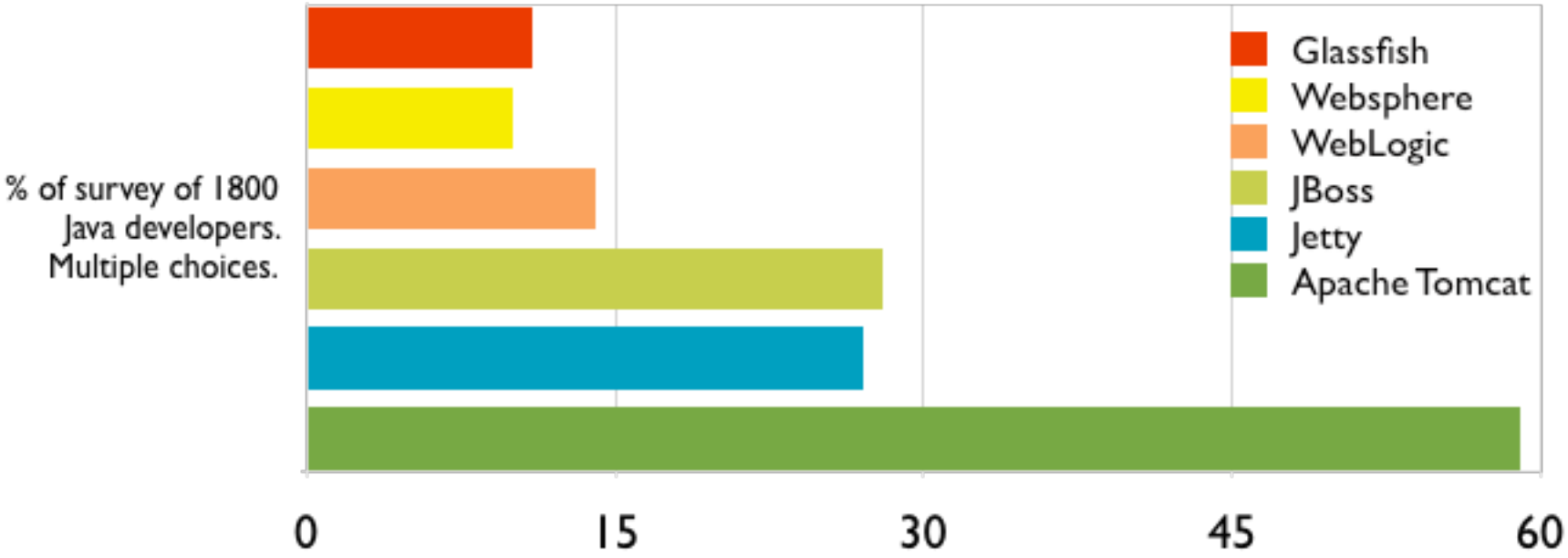
http://en.wikipedia.org/wiki/Single_responsibility_principle

Spring Boot supports Apache Tomcat 7 by default.

Easy to switch to Jetty, or Tomcat 8

Demonstration

Switching embedded web servers



Source: ZeroTurnaround Developer Productivity Report 2012

Demonstration

From fat `.jar` to `.war`

CLOUD

```
BUILD SUCCESSFUL

Total time: 9.366 secs
➜ spring-music git:(master) ✗ cf push
Using manifest file manifest.yml

Creating spring-music... OK

Creating route j1-spring-music-466b.cfapps.io... OK
Binding j1-spring-music-466b.cfapps.io to spring-music... OK
Uploading spring-music... OK
Preparing to start spring-music... OK
-----> Downloaded app package (19M)
-----> Downloading OpenJDK 1.7.0_21 JRE from http://download.pivotal.io.s3.amazonaws.com/openjdk/lucid/
x86_64/openjdk-1.7.0_21.tar.gz (10.8s)
       Expanding JRE to .java (1.0s)
-----> Downloading Auto Reconfiguration 0.7.1 from http://download.pivotal.io.s3.amazonaws.com/auto-rec
onfiguration/auto-reconfiguration-0.7.1.jar (1.2s)
       Modifying /WEB-INF/web.xml for Auto Reconfiguration
-----> Downloading Tomcat 7.0.42 from http://download.pivotal.io.s3.amazonaws.com/tomcat/tomcat-7.0.42.
tar.gz (3.1s)
       Expanding Tomcat to .tomcat (0.1s)
       Downloading Buildpack Tomcat Support 1.1.1 from http://download.pivotal.io.s3.amazonaws.com/tomc
at-buildpack-support/tomcat-buildpack-support-1.1.1.jar (0.0s)
-----> Uploading droplet (55M)
Checking status of app 'spring-music'......
  0 of 1 instances running (1 starting)
  0 of 1 instances running (1 starting)
  1 of 1 instances running (1 running)
Push successful! App 'spring-music' available at http://j1-spring-music-466b.cfapps.io
```

Demonstration

To the cloud!

Spring Boot is production-ready, by default

Comes out of the box with smart monitoring and management tools, the CrashD server, etc.

Demonstration

production ready REST services with Boot

Spring IO Guides
http://spring.io/guides

Roy Fielding's Dissertation introduces REST
http://www.ics.uci.edu/~fielding/pubs/dissertation/evaluation.htm#sec_6_1%7C

The Spring REST Shell
http://github.com/jbrisbin/rest-shell

Spring **Security, Security OAuth**, **Spring Data REST**, **HATEOAS**, **Social**
http://github.com/spring-projects

Spring **MVC Test Framework**
http://docs.spring.io/spring/docs/4.0.x/spring-framework-reference/html/testing.html

Oliver Gierke's talk on Hypermedia from Øredev
@ http://vimeo.com/53214577

Lez Hazelwood's talk on designing a beautiful JSON+REST API

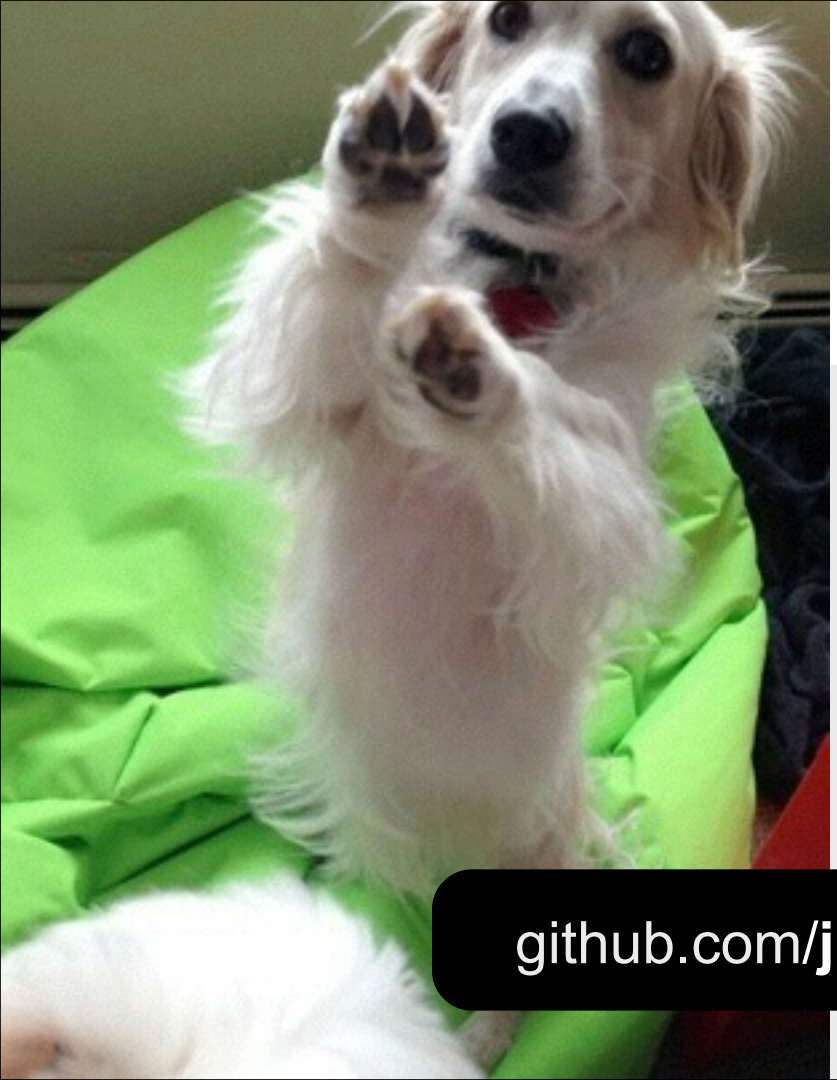
Ben Hale's talk on REST API design with Spring from SpringOne2GX 2012
@ http://www.youtube.com/watch?v=wylViAqNiRA

My links:

github.com/**joshlong/the-spring-rest-stack**

slideshare.net/**joshlong/rest-apis-with-spring**

@starbuxman



@starbuxman
josh@joshlong.com
slideshare.net/joshlong
github.com/joshlong
speakerdeck.com/joshlong

spring

github.com/**joshlong**/**the-spring-rest-stack**