CHARIOT SOLUTIONS

Spring Social

Messaging Friends & Influencing People

ETE 2011

Apr 27-28, 2011

CHARIOT SOLUTIONS

Who Am I?

- Instructor/Mentor at
  chariotsolutions.com/education

CHARIOT SOLUTIONS

Chariot Education Services
Public & Private Training
One-On-One Mentoring
Lunch and Learns

- Active Tweeter for Open Source Tech Topics
  twitter.com/gdickens

twitter

- Certified Instructor for springsource
  A division of vmware

- DZone Most Valuable Blogger
  dzone.com/page/mvbs – dzone.com/user/284679

  Technophile Blog - technophile.gordondickens.com

DZone

40% discount
"springroo40"

CHARIOT SOLUTIONS

Agenda

- Spring & REST
- Spring Social
- Security with OAuth
- Spring Mobile
- Spring Android
- Spring iPhone
- Spring Greenhouse

CHARIOT SOLUTIONS

Who is using REST?

Meetup

LinkedIn

amazon web services

Google

salesforce

facebook

brightkite
people. places. friends.

myspace.com
a place for friends

Gowalla

flickr

NETFLIX

github
SOCIAL CODING

TripIt

plaxo

twitter

Yammer

friendfeed

GUINNESS
EST 1759

CHARIOT SOLUTIONS

Spring REST

- Added in Spring 3.0
- `RestTemplate`
  - Client side
- Spring MVC
  - Server side
  - <mvc:annotation-driven/>
  - Annotations
  - Content Negotiation
  - Web page support for PUT & DELETE
    - `HiddenHttpMethodFilter`

CHARIOT SOLUTIONS

<mvc:annotation-driven/>

- Validation
  - JSR-303
- Formatting
  - Annotated Field Formatters
- Conversion
  - JSON
  - XML
  - ATOM/RSS
  - JodaTime
  - Custom Conversion (MyBean1 to/from MyBean2)

CHARIOT SOLUTIONS

Spring REST Annotations

- `@RequestMapping`
  - Method for URL Path
- `@PathVariable`
  - Var in URL path "/{myVar}"
- `@RequestParam`
  - Parameter "/myurl/?myVar=abc"
- `@RequestBody`
  - Unmarshal into bean
- `@ResponseBody`
  - Marshal into bean

- `@RequestHeader`
  - Value in Req Header
- `@ResponseStatus`
  - Http Codes on success
- `@ExceptionHandler`
  - Methods by exception
- `@ModelAttribute`
  - Returning result into model

CHARIOT SOLUTIONS

REST Controller Sample


```
@Controller
public class TwitterTimelineController {
    ...

@RequestMapping(value="/twitter/timeline/{timelineType}", method=RequestMethod.GET)
public String showTimeline(@PathVariable("timelineType") String timelineType, Model
    model) {

  TwitterApi twitterApi = getTwitterApi();
  if (timelineType.equals("Home")) {
    model.addAttribute("timeline", twitterApi.timelineOperations().getHomeTimeline());
  } else if(timelineType.equals("Public")) {
    model.addAttribute("timeline", twitterApi.timelineOperations().getPublicTimeline());
    ...
  }
  model.addAttribute("timelineName", timelineType);
  return "twitter/timeline";
}
```

CHARIOT SOLUTIONS

Content Negotiating

- **`ContentNegotiatingViewResolver`**
  - Client requests format of data
  - "Accept" header
  - "format" parameter
  - URL extension
  - Java Activation Framework

```
<bean class="o.s.w.s.v.ContentNegotiatingViewResolver">
  <property name="mediaTypes">
    <map>
      <entry key="json" value="application/json" />
      <entry key="xml" value="application/xml" />
      <entry key="htm" value="text/html" />
    </map>
  </property>
  <property name="defaultContentType" value="text/html" />
</bean>
```

CHARIOT SOLUTIONS

REST & Social Media

- Search Twitter with `RestTemplate`

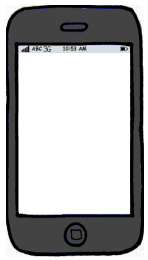
```
RestTemplate restTemplate=
   new RestTemplate();

String search= "#phillyete";

String results = restTemplate.getForObject
   ( "http://search.twitter.com/search.json?
   q={search}",String.class, search);
```

Note: All the cool kids are using REST

CHARIOT SOLUTIONS

Social & Mobile Projects

- *Spring Social*
- *Spring Mobile*
- *Spring Android*
- *Spring Greenhouse*



springsource
A division of vmware

OAUTH

twitter

Linked in

facebook

Tripit

CHARIOT SOLUTIONS

What is Spring Social?

- Allow our apps to interact with

twitter

TripIt

facebook

Gowalla

Linked in

github
SOCIAL CODING

CHARIOT SOLUTIONS

Templates to the Rescue

- Template for each Social Media Provider
  - `TwitterTemplate`, `LinkedInTemplate`, etc.
- Provider Specific
  - Authentication Support
  - API methods
- For other providers:
  - Roll your own
  - `RestTemplate` is our friend

CHARIOT SOLUTIONS

Simple Twitter


```
TwitterTemplate twitterTemplate =
   new TwitterTemplate();

List<String> friends =
   twitterTemplate.getFriends("chariotsolution");

for (String friend : friends) {
   logger.debug("Friend: {}", friend);
}
```

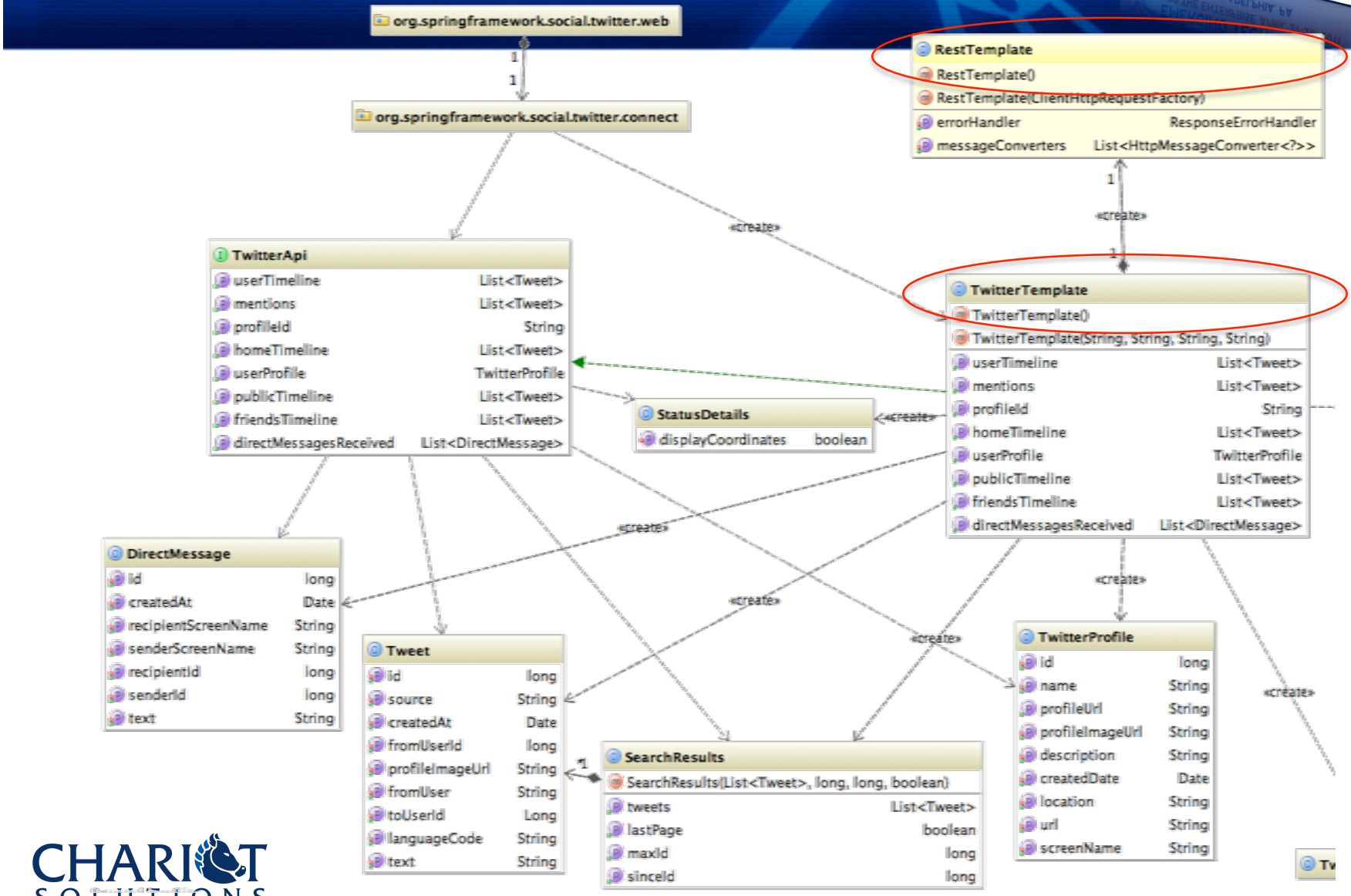
CHARIOT SOLUTIONS

What is TwitterTemplate?

- One of the Provider Specific Templates
- A convenience class
- Implements `TwitterApi`
- Wraps RESTful calls to Twitter
  - Using Spring 3 `RestTemplate`
- Authentication

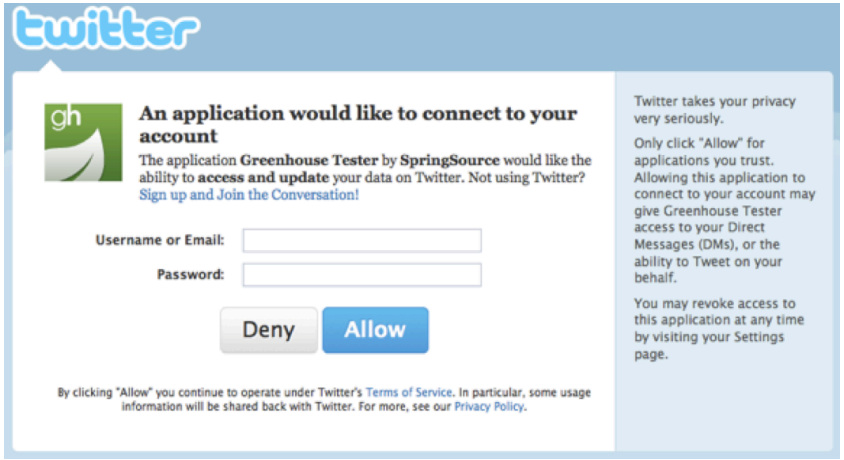
CHARIOT SOLUTIONS

Twitter API


org.springframework.social.twitter.web
org.springframework.social.twitter.connect
RestTemplate
RestTemplate()
RestTemplate(ClientHttpRequestFactory)
errorHandler ResponseErrorHandler
messageConverters List<HttpMessageConverter<?>>
«create»
TwitterApi
userTimeline List<Tweet>
mentions List<Tweet>
profileId String
homeTimeline List<Tweet>
userProfile TwitterProfile
publicTimeline List<Tweet>
friendsTimeline List<Tweet>
directMessagesReceived List<DirectMessage>
TwitterTemplate
TwitterTemplate()
TwitterTemplate(String, String, String, String)
StatusDetails
displayCoordinates boolean
DirectMessage
id long
createdAt Date
recipientScreenName String
senderScreenName String
recipientId long
senderId long
text String
Tweet
id long
source String
createdAt Date
fromUserId long
profileImageUrl String
fromUser String
toUserId Long
languageCode String
text String
SearchResults
SearchResults(List<Tweet>, long, long, boolean)
tweets List<Tweet>
lastPage boolean
maxId long
sinceId long
TwitterProfile
id long
name String
profileUrl String
profileImageUrl String
description String
createdDate Date
location String
url String
screenName String

Template API Calls

- Most calls require authentication
  - Some trivial calls do not need auth
- Authentication
  - Provider side
  - Secured with OAuth



CHARIOT SOLUTIONS

Authenticating

OAUTH

"An open protocol to allow secure API authorization in a simple and standard method from desktop to web applications."

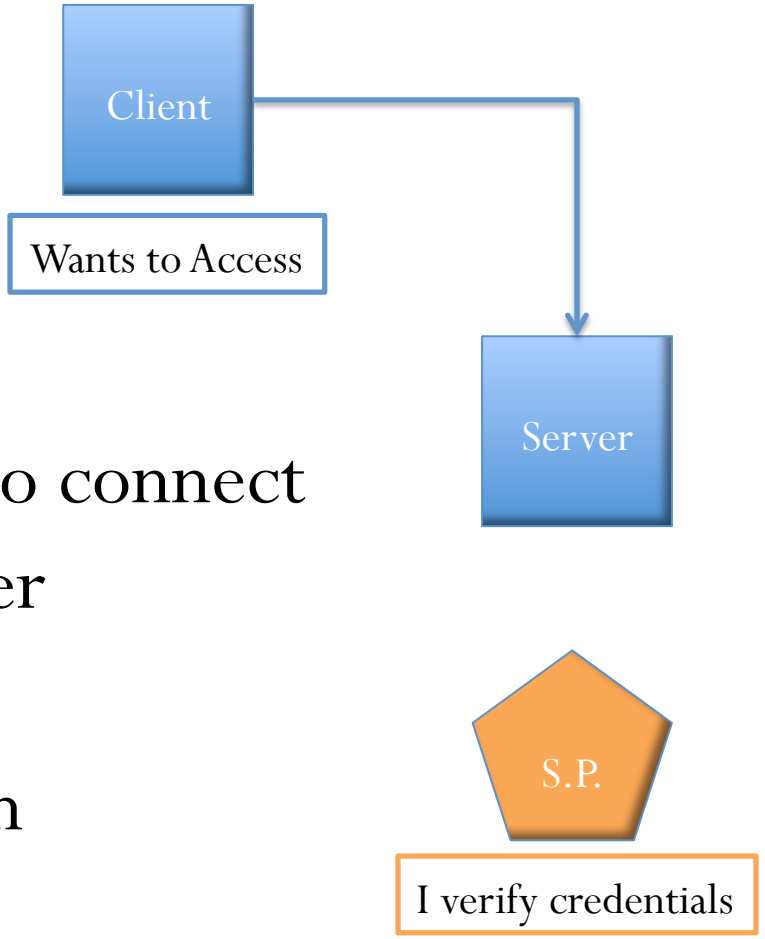
http://oauth.net

code.google.com/p/oauth/

CHARIOT SOLUTIONS

OAuth Participants

- Client
  - Our app
  - Wants to access server
- Server
  - With whom we want to connect
  - May not be the provider
- Service Provider
  - Provides authentication



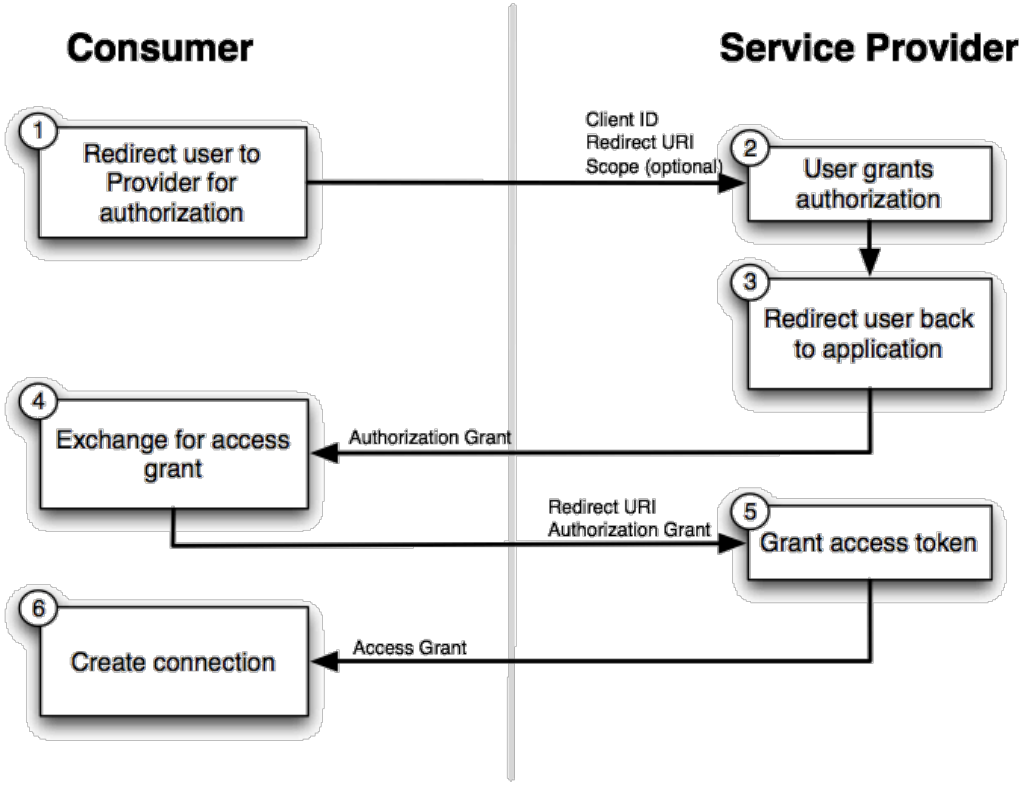
CHARIOT SOLUTIONS

OAuth Flavors

- 1.0a
  - More complex
  - Tripit twitter LinkedIn
- 2.0
  - github SOCIAL CODING facebook Gowalla

CHARIOT SOLUTIONS

OAuth Safety Dance



1. App asks for a **request token**
   - OAuth 1.0a - w callback URL

2. Provider issues a **request token**

3. App redirects user to provider's auth page
   - passes **request token**
   - OAuth 1.0a – w callback URL

4. Provider prompts to authorize

5. Provider redirects user back to app
   - OAuth 1.0a – w **verifier code**

6. App exchanges **request token** for **access token**
   - OAuth 1.0a - also w **verifier code**

7. Provider issues **access token**

8. App uses the **access token**
   - App now obtains a ref to the Service API
   - interacts with provider on behalf of the user

CHARIOT SOLUTIONS

Twitter Auth Steps

1. `ApiKey` – Redirect to provider auth page
2. *User authenticates themselves*
3. *User authorizes our app to act on their behalf*
4. *User redirected to our site w/* `RequestToken` & `VerificationCode`
5. `RequestToken` & `VerificationCode`, get `AccessToken`
6. Interact w/ provider using `ApiKey` & `AccessToken`

CHARIOT SOLUTIONS

Template & OAuth Summary

- Provider specific template
- OAuth support within template

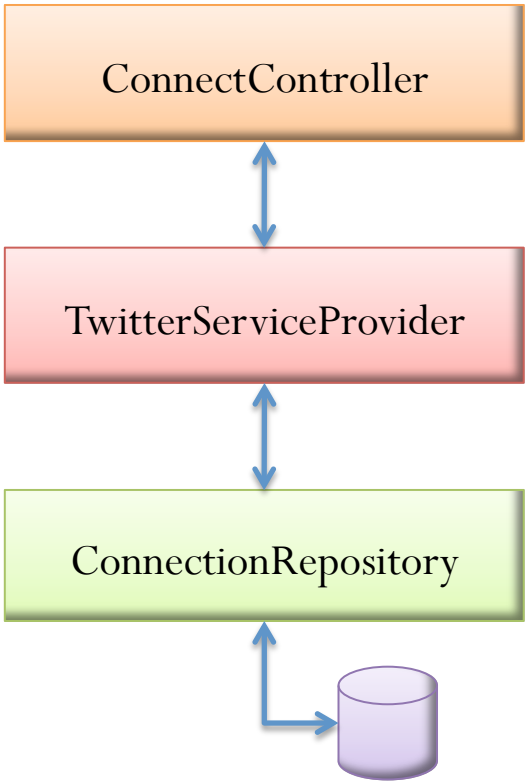
- How do we manage:
  - provider Login flow?
  - credential storage?

We need CPR!

CHARIOT SOLUTIONS

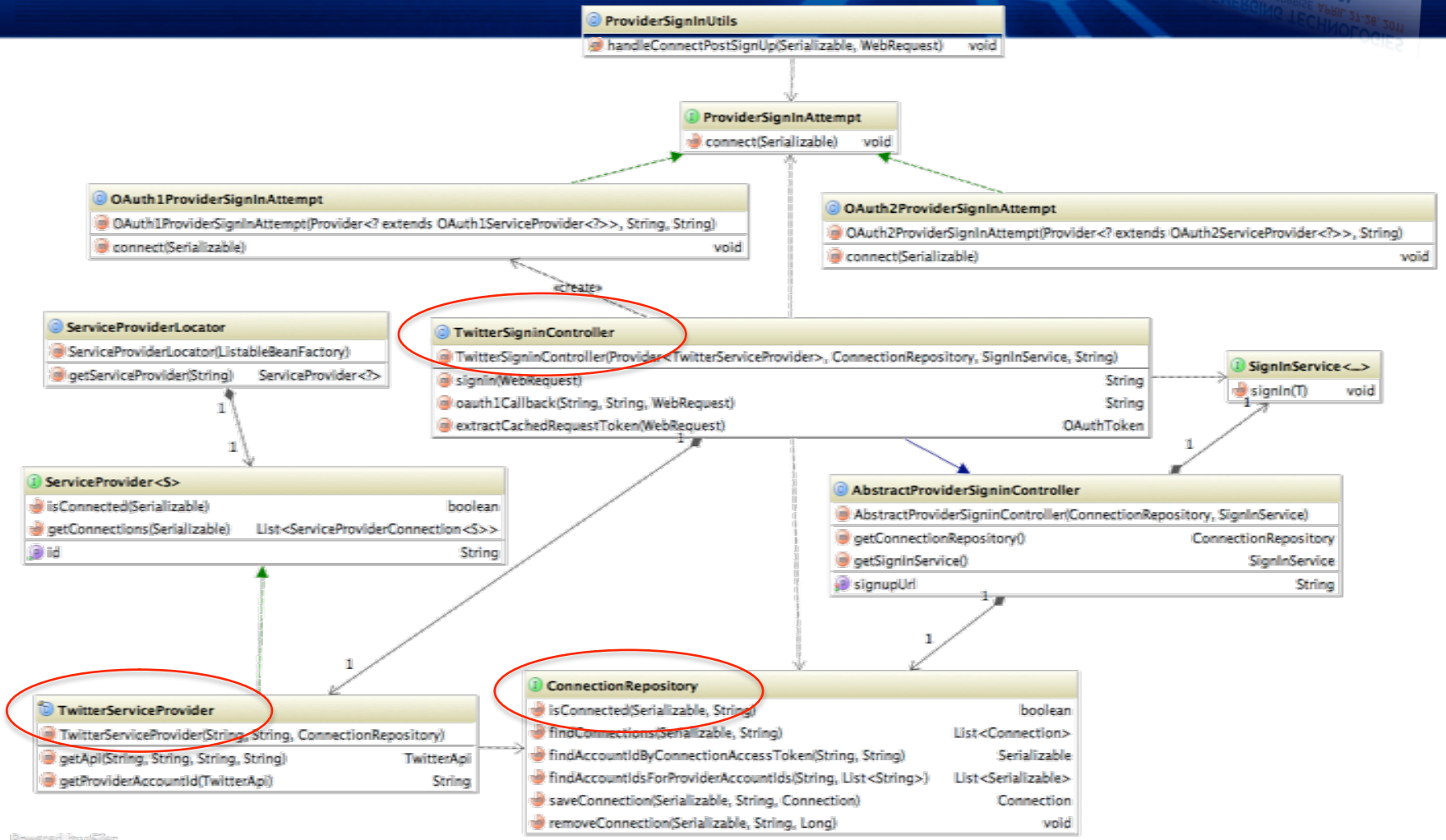
Spring CPR

- Connect **Controller**
  - Connection to Service Provider
- Service **Provider**
  - `Twitter`, `LinkedIn`, etc.
- Connection **Repository**
  - Storage for Connection Details



CHARIOT SOLUTIONS

Twitter CPR Signin



CHARIOT SOLUTIONS

CPR – Controller


```
<bean class="o.sf.social.web.connect.ConnectController">
    <constructor-arg value="${application.url}"/>
</bean>
```

- Base Connection Controller
  - Handles authentication process
  - Locates connection from repository
  - Connection Factory for Authentication
  - Interceptors *(custom, optional)*
    - Tweet After Connect
    - Post to Wall After Connect
- *application.url*
  - Base secure URL for our app
    - *http://localhost:8080/MySocialApp*
  - Callback URL passed to the service providers

GET /connect/{provider ID}
Display connection status

POST /connect/{provider ID}
Initiate connection flow

GET /connect/{provider ID}
?oauth_token={request token}
&oauth_verifier={verifier}
Handle callback

DELETE /connect/{provider ID}
Disconnect from provider

CHARIOT SOLUTIONS

CPR - Provider


```
<bean class="o.sf.social.twitter.connect.TwitterServiceProvider">
    <constructor-arg value="${twitter.appId}" />
    <constructor-arg value="${twitter.appSecret}" />
    <constructor-arg ref="connectionRepository" />
</bean>
```

- *twitter.appId*
  - ApiKey assigned to app by provider
- *twitter.appSecret*
  - VerifierCode returned by provider

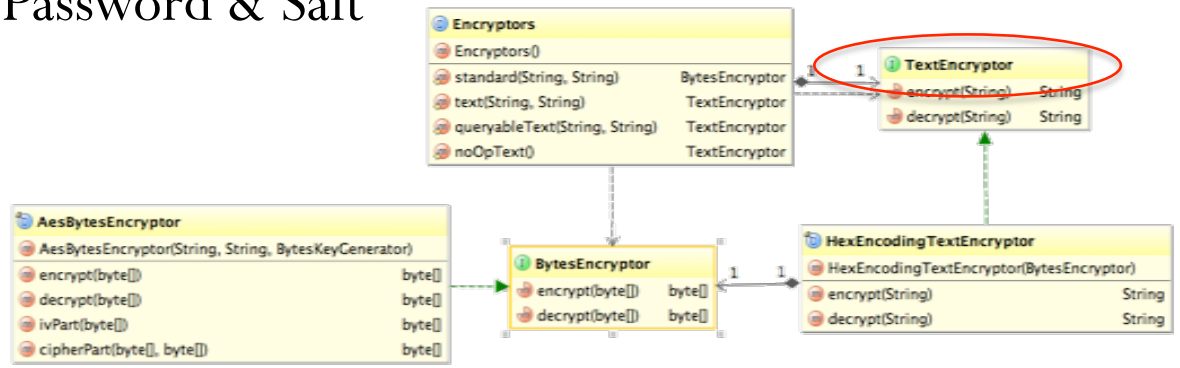
CHARIOT SOLUTIONS

CPR - Repository


```
<bean id="connectionRepository"
   class="o.sf.social.connect.jdbc.JdbcConnectionRepository">
    <constructor-arg ref="dataSource" />
    <constructor-arg ref="textEncryptor" />
</bean>
```

```
create table Connection (
  id identity,
  accountId varchar not null,
  providerId varchar not null,
  accessToken varchar not null,
  secret varchar,
  refreshToken varchar,
  providerAccountId varchar,
  primary key (id));
```

- `textEncryptor`
  - Spring Security 3.1 – crypto lib
  - interface for symmetric encryption of text strings
  - Unencrypted, Password & Salt



CHARIOT SOLUTIONS

Greenhouse: RI app suite

- Web App
  - Group Event Management
  - Environment-specific Beans and Profiles
  - Password Encoding &Data Encryption
- Social Media
  - Integration
  - Authentication
- Mobile Apps
  - Android
  - iPhone

CHARIOT SOLUTIONS

Greenhouse: Social & Mobile

- Spring **Social**
  - App Framework
  - Service Provider Framework
    - Twitter, Facebook, LinkedIn & TripIt
  - Sign Up, Sign In & Password Reset
  - Member Invite
  - Badge System
- Spring **Mobile**
  - iPhone Client
  - Android Client
  - Mobile web version for other smartphone platforms

CHARIOT SOLUTIONS

Spring Mobile

- Server Side
- Spring MVC Extension
- Key Facets:
  1. Mobile Device Detection
  2. Site Preference Management
  3. Site Switching

CHARIOT SOLUTIONS

Mobile Detection in MVC

- `DeviceResolver`
  - Inspects inbound HttpRequest
  - "User-Agent" header
  - Default Analysis: "Mobile device client?"

- More sophisticated resolvers identify
  - screen size
  - manufacturer
  - model
  - preferred markup

- **`DeviceResolverHandlerInterceptor`**
  - `LiteDeviceResolver` (*default*)
    - Based on Wordpress Lite Algorithm
    - Part of Wordpress Mobile Pack (*wpmp*)
  - Plug in another for specific features
    - `WurflDeviceResolver` (*Java API*)

CHARIOT SOLUTIONS

MVC Configuration

- Handler Interceptor

```
<mvc:interceptors>
  <!- Resolve device on pre-handle -->
  <bean class="o.sf.mobile.device.DeviceResolverHandlerInterceptor"/>
</mvc:interceptors>
```

- Enable `Device` for Controller Methods

```
<mvc:annotation-driven>
  <mvc:argument-resolvers>
    <bean class="o.sf.mobile.device.DeviceWebArgumentResolver"/>
  </mvc:argument-resolvers>
</mvc:annotation-driven>
```

- Inject `Device`

```
@Controller
public class MyController {
    …
  @RequestMapping("/")
  public void sayHi(Device device) {
    if (device.isMobile()) logger.info("Hello mobile user!");
```

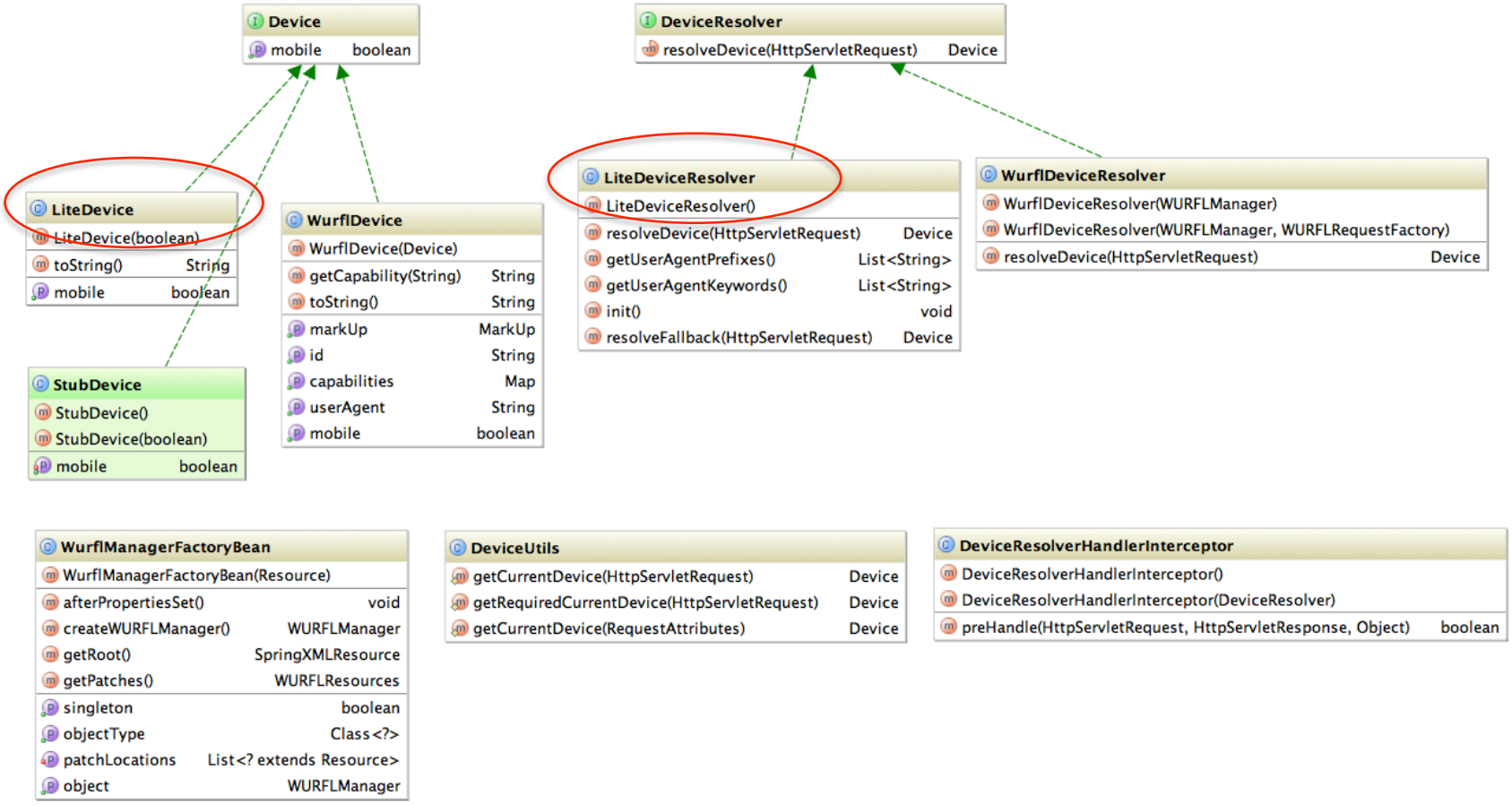
CHARIOT SOLUTIONS

Greenhouse CPR

- **`AnnotatedControllerConfig`**
  - `DeviceResolverHandlerInterceptor`
    - MVC interceptor resolves the `Device` originating the web request
  - `DeviceHandlerMethodArgumentResolver`
    - Uses `DeviceHandler` static class
    - Extracts current device from web request

CHARIOT SOLUTIONS

DeviceResolver API



CHARIOT SOLUTIONS

Preference Management

- App supporting multiple sites
  - "`mobile`"
  - "`normal`"
- Allow user to switch
- `SitePreferenceHandler` interface
- `SitePreferenceWebArgumentResolver`
- `CookieSitePreferenceRepository`

CHARIOT SOLUTIONS

MVC Pref Config

- Enable SitePreference support

```
<mvc:annotation-driven>
  <mvc:argument-resolvers>
    <bean class="o.sf.mobile.device.DeviceWebArgumentResolver" />
    <bean class="o.sf.mobile.device.site.SitePreferenceWebArgumentResolver" />
  </mvc:argument-resolvers>
</mvc:annotation-driven>
```

- Inject into Controller Methods

```
@Controller
public class MyController {
  @RequestMapping("/")
  public String home(SitePreference sitePreference, Model model) {
    if (sitePreference == SitePreference.MOBILE) { …
```

CHARIOT SOLUTIONS

Site Switching

- Hosting mobile site different location

- `SiteSwitcherHandlerInterceptor`
- mDot

  `m.${serverName}`
- dotMobi

  `${serverName}.mobi`

CHARIOT SOLUTIONS

Spring Android

- Client tools
- Http Components
- `RestTemplate`
- Object to JSON Marshaling
- Object to XML Marshaling
- RSS and Atom Support
- Greenhouse app in Android market
  - https://market.android.com/details?id=com.springsource.greenhouse

CHARIOT SOLUTIONS

Spring iPhone

- Greenhouse for iPhone client
- Download & Install the iOS SDK
- Project source code from the git
  - git clone git://git.springsource.org/greenhouse/iphone.git
- Open the Greenhouse.xcodeproj in Xcode
- Expects Greenhouse web app running locally
- Greenhouse App in iTunes
  - http://itunes.apple.com/us/app/greenhouse/id395862873?mt=8

CHARIOT SOLUTIONS

Spring Social Showcase

- Social MVC Application
- CPR configuration
- Authentication
- Interceptors

twitter

Tripit

facebook

CHARIOT SOLUTIONS

Summary

- Spring Social
  - App collaboration with SM Sites
  - Security through OAuth
- Spring REST
  - Significant for Social & Mobile
  - RestTemplate Rules!
- Spring Mobile
  - MVC Support for Mobile Devices
- Spring Android
  - Client Tools
  - based on `RestTemplate`

- Spring Greenhouse
  - Social and Mobile Demo
- Spring Social Showcase
  - Social Demo

CHARIOT SOLUTIONS

Q & F

- Questions?

- Follow Up
  - Learn REST! (*InfoQ*)
  - Spring Social
  - Spring Mobile
  - Spring Android
  - Spring Social Showcase (*git*)
  - Spring Greenhouse
- Contact me
  - http://technophile.gordondickens.com (*blog*)
  - http://chariotsolutions.com/education (*Spring & REST courses*)
  - gdickens@chariotsolutions.com
  - twitter.com/gdickens (*tech tweets*)

CHARIOT SOLUTIONS