JavaOne℠

Creating Server-Side and Mobile Mashups with OpenSocial's Java™ Client Libraries

Chris Schalk

Ryan Boyd

Google

JavaOne 2009

Agenda

- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - OAuth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

Agenda

- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - OAuth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

What is OpenSocial?

"OpenSocial defines a common set of APIs based on Open Standards for building social applications across multiple websites"

What is OpenSocial?

Before OpenSocial…

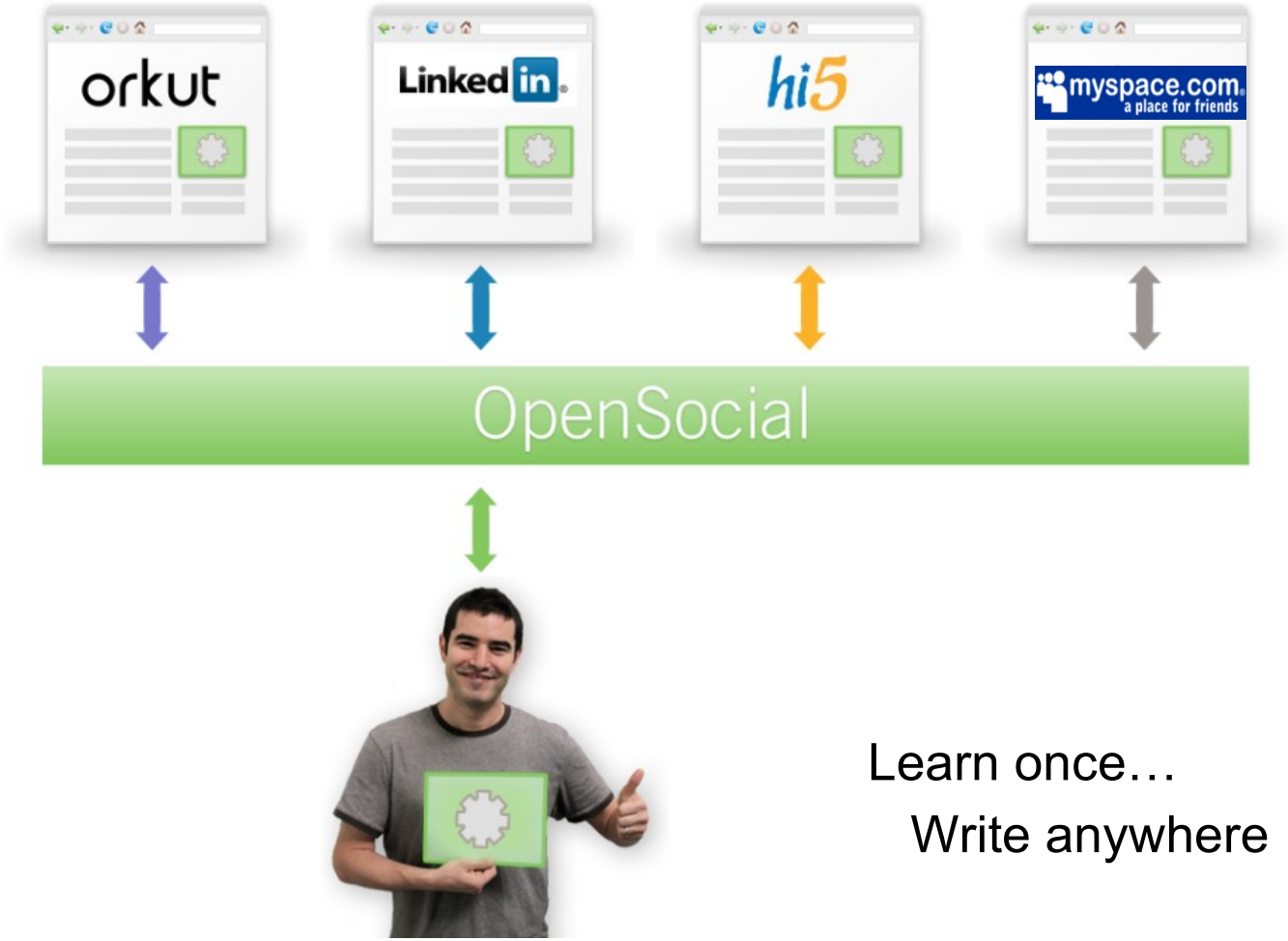
What is OpenSocial?

API

People/Friends
Activities
Shared Data

Standards Based!

What is OpenSocial?



Learn once…

Write anywhere

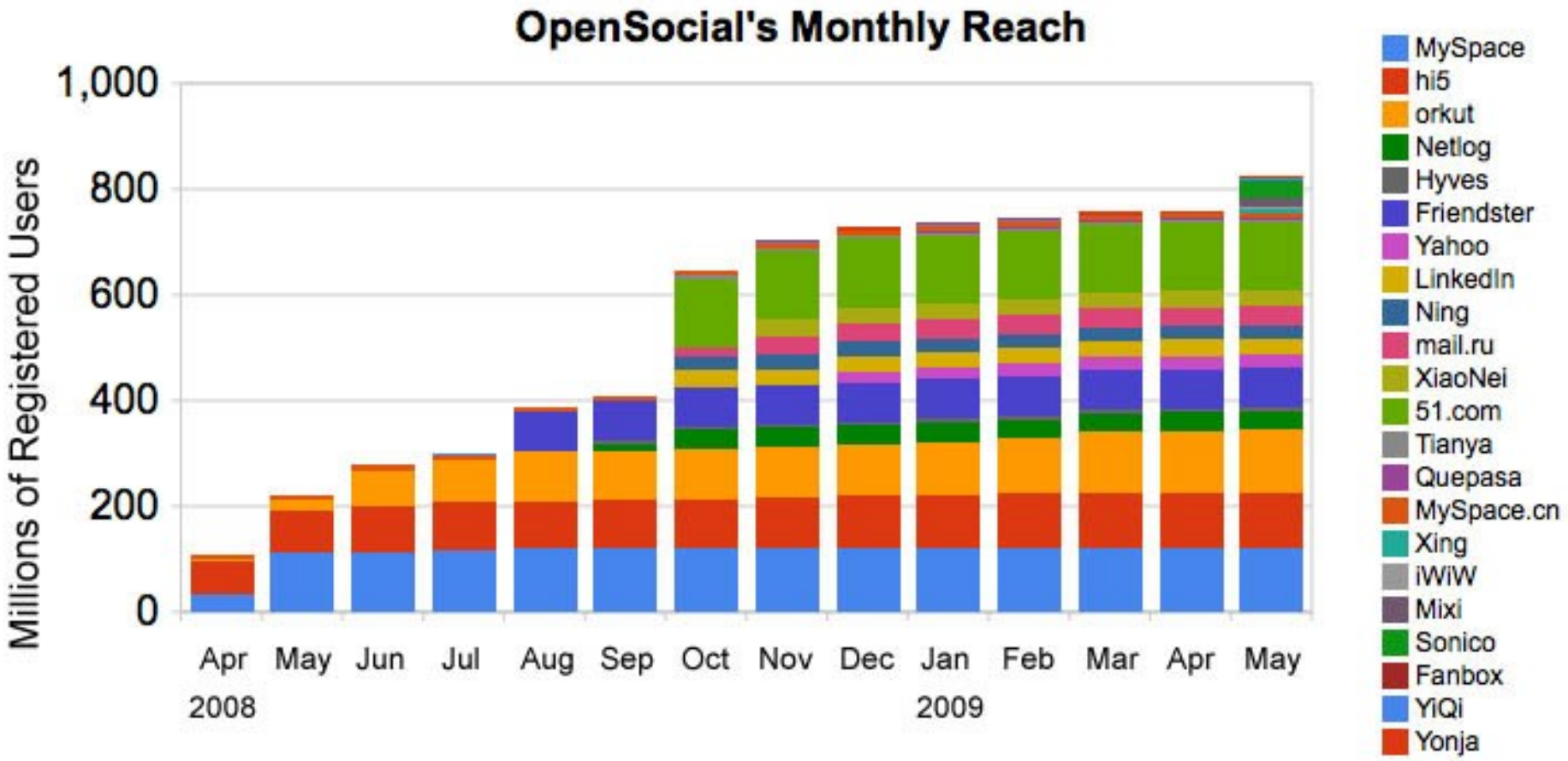
Who owns OpenSocial?

Google ? No!

OpenSocial is managed under the auspices of the "OpenSocial Foundation" - http://www.opensocial.org

Who's Using it?

And many more…



OpenSocial now has over 800 Million users

Demonstration

- A simple OpenSocial gadget demonstration

Agenda

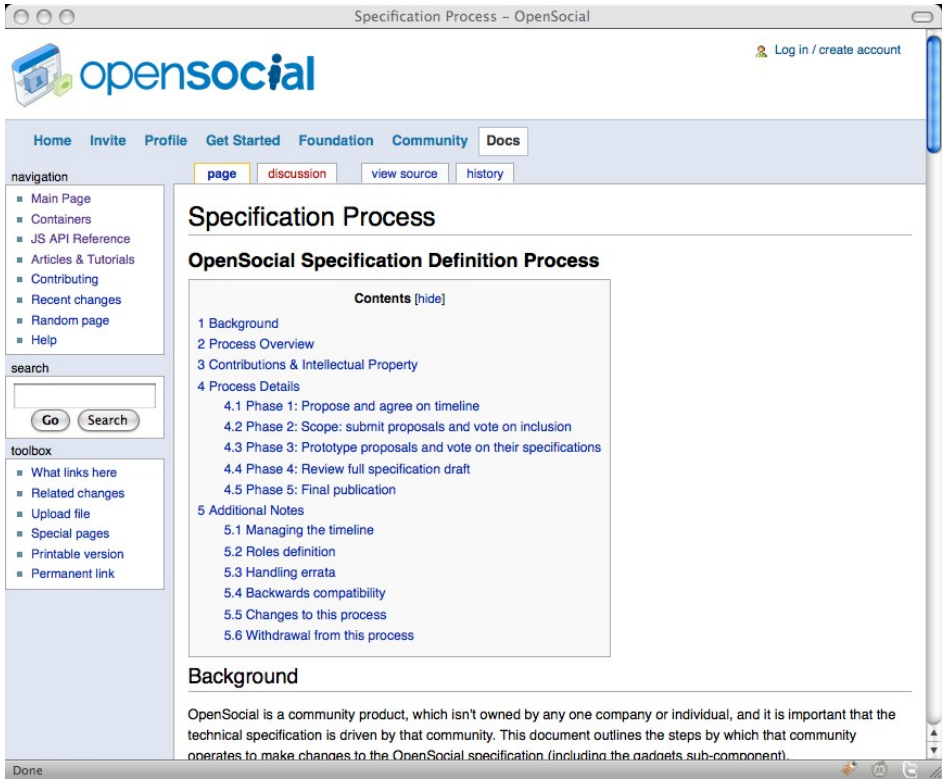
- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - OAuth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

The Evolution of the OpenSocial REST Specification

- Before v. 0.8, the only OpenSocial API was a JavaScript API
  - Only worked within gadgets
- Initial OpenSocial API did not have a REST Specification.
  - No server-to-server communication
  - No options for Desktop or Mobile integration

Lack of REST support meant no server-side integration

- Through OpenSocial's open specification process, the community responded and created the OpenSocial REST specification



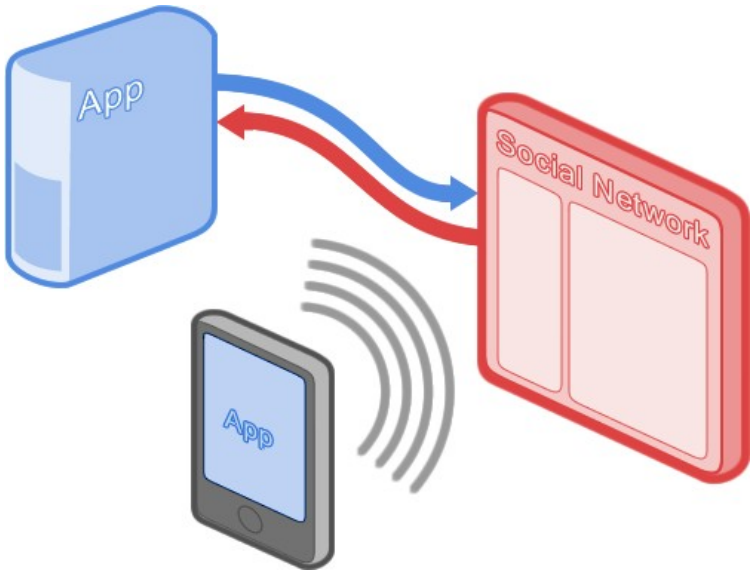
OpenSocial REST support rolls out

- Version 0.8
  - Server responded to REST requests in either XML or JSON
  - AtomPub operations supported
- Version 0.8.1
  - RESTful and RPC protocols established
    - RPC protocol responds to HTTP POSTs and allows for batching of requests

OpenSocial REST Support

Opens new development models

- Server-to-server communication
- Multiple client types
  - Server based UIs (JSP/JSF)
  - Mobile devices



RESTful Protocol

- Resources are URLs.

Example - People:

- All people connected to the given user:

```
/people/{guid}/@all
```

- All friends of the given user:

```
/people/{guid}/@friends
```

- Profile of the given user:

```
/people/{guid}/@self
```

- Profile of the authenticated user:

```
/people/@me/@self
```

- Supported Person fields:

```
/people/@supportedFields
```

RPC Protocol

- Http POST instead of GET

```
format={format}
```

  - Request extra fields

```
fields={-join|,|field}.
```

  - Filtering:

```
filterBy={fieldname}
filterOp={operation}filterValue={val
ue}
updatedSince={xsdDateTime}
networkDistance={networkDistance}
```

  - Paging:

```
count={count}
sortBy={fieldname}
sortOrder={order}
startIndex={startIndex}
```

Demonstration

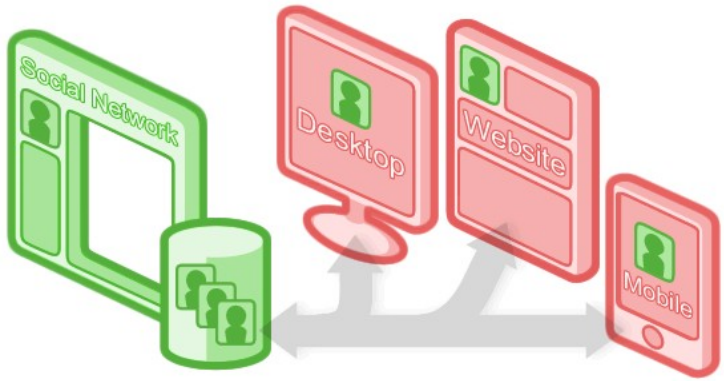
- Experimenting with the the RESTful protocol with local Apache Shindig and a browser

Agenda

- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - OAuth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

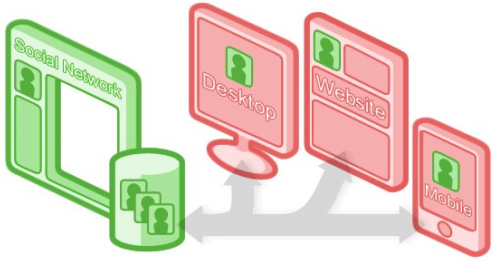
OpenSocial Client Libraries

A set of client libraries for that enable direct communication to an OpenSocial server.



- Client libraries exist for PHP, Ruby, Python and Java
- Supports both REST and RPC protocols
- Documentation Wiki pages
- Sample applications provided

OpenSocial Client Libraries

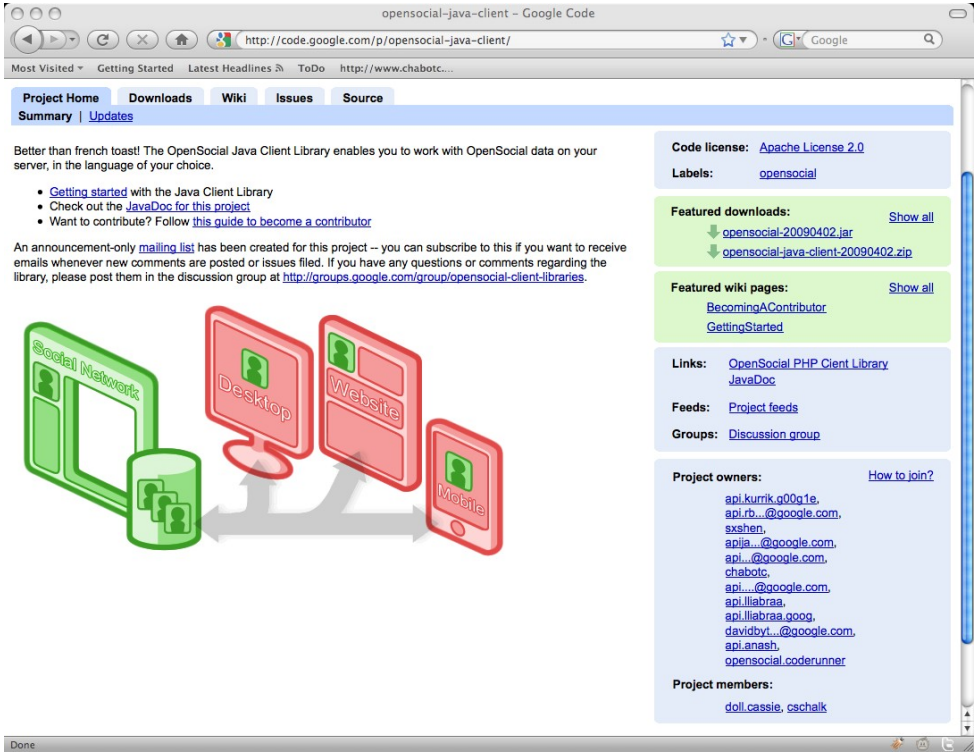


Hosted on Google code projects

http://code.google.com/p/opensocial-php-client
http://code.google.com/p/opensocial-ruby-client
http://code.google.com/p/opensocial-python-client
http://code.google.com/p/opensocial-java-client

Demonstration

- Getting familiar with the OpenSocial Java client library



Agenda

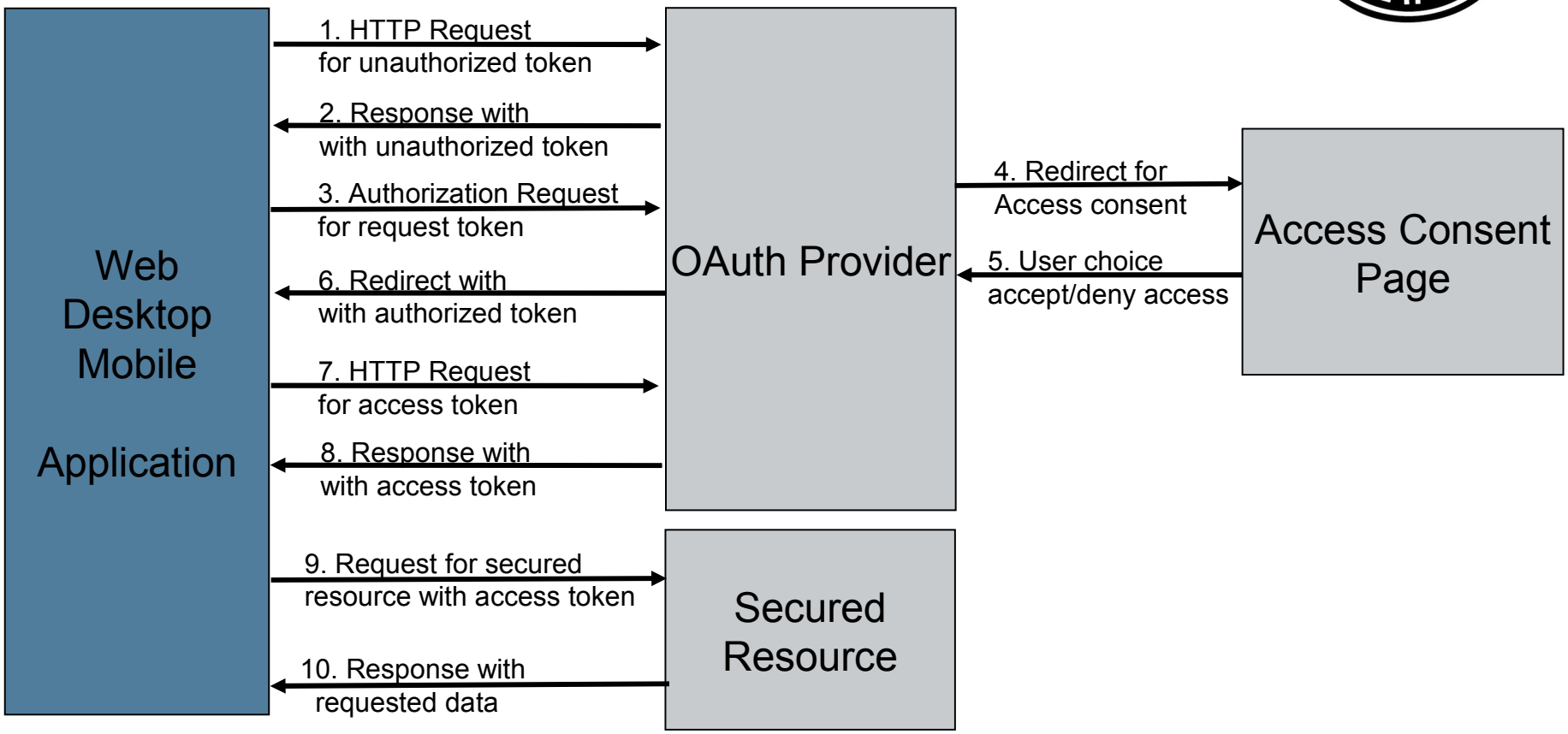
- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - OAuth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

What about security?

- OAuth to the rescue!
- “An open protocol to allow secure API authorization in a simple and standard method from desktop and web applications” - Oauth.net
- The OpenSocial Client Libraries offer built-in support for OAuth

The OAuth Authentication Process

Also known as “3-legged OAuth”



2-legged vs. 3-legged OAuth

- A “2-legged” OAuth transaction is actually a simplified version of a typical OAuth transaction which is referred to as the 3-legged OAuth “dance”.
  - The 2-legged process is simpler because it doesn’t force the end user through an access consent page.
  - Is optimal for server-to-server communication which can be outside of the context of a user.
- For a secured transaction with a user context, 3-legged OAuth would typically be employed.
  - A mobile or desktop application would typically use 3-legged OAuth.
- The OpenSocial client libraries support both authorization types.
  - Not all OpenSocial containers support both authorization types however.

Agenda

- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - OAuth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

Fetching Data from an OpenSocial container

Using “2-legged OAuth”

```
OpenSocialClient c = new OpenSocialClient(OpenSocialProvider.valueOf("ORKUT_SANDBOX"));

c.setProperty(OpenSocialClient.Property.CONSUMER_SECRET, "uynAeXiWTisflWX99KU1D2q5");
c.setProperty(OpenSocialClient.Property.CONSUMER_KEY, "orkut.com:623061448914");
c.setProperty(OpenSocialClient.Property.VIEWER_ID, "03067092798963641994");

try {
  // Retrieve the friends of the specified user using the OpenSocialClient
  Collection<OpenSocialPerson> friends = c.fetchFriends("03067092798963641994");

  for (OpenSocialPerson friend : friends) {
    System.out.println("- " + friend.getDisplayName());
  }
} catch (org.opensocial.client.OpenSocialRequestException e) {
  System.out.println("OpenSocialRequestException thrown: " + e.getMessage());
  e.printStackTrace();
}
```

Demonstration

- A 2-legged OAuth transaction using the Java client library.
  - Requesting profile information and friends list from orkut.com

Demonstration

- A 3-legged OAuth transaction example.
  - Doing the “dance” with MySpace.com
  - Using JSPs to show passing user to authorization page

Agenda

- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - Oauth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

Building Server-Side Mashups

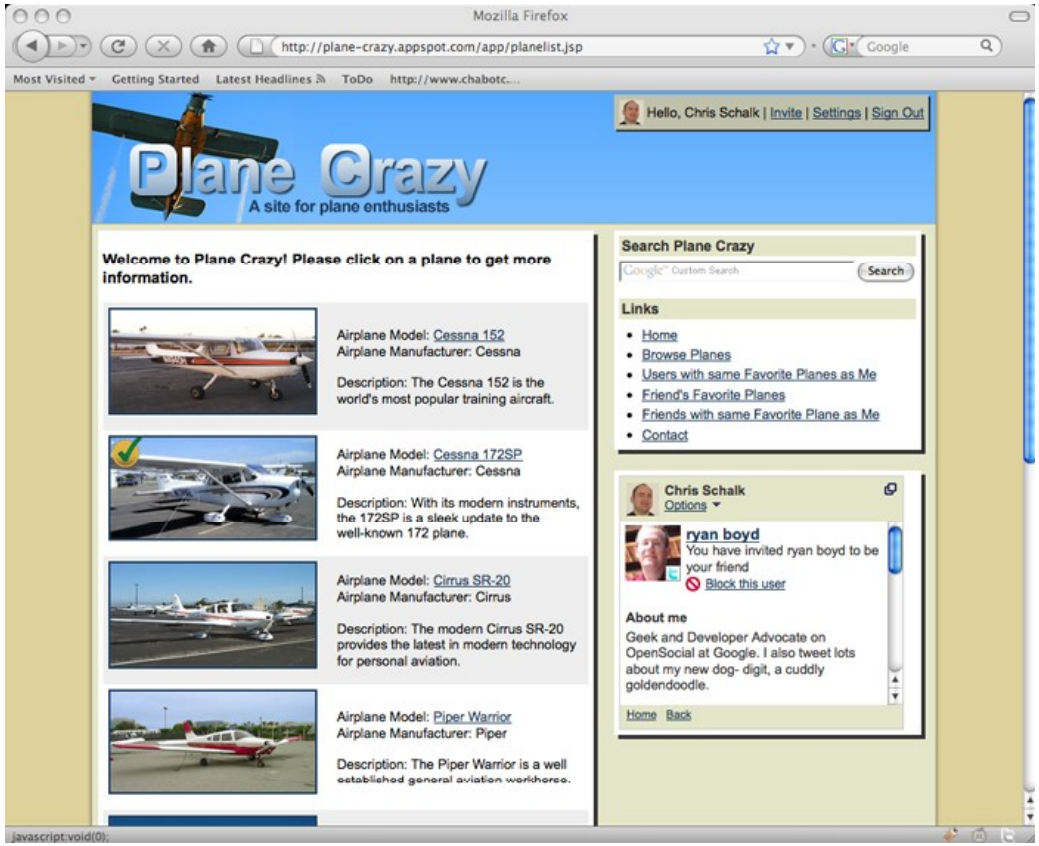
with the OpenSocial client libraries

- Any server-side Java Web application can easily integrate with an OpenSocial server

- An example:
  - A aviation themed website with social features enabled via the client libraries
    - App uses Java App Engine
    - Connects to Google Friend Connect which is an OpenSocial container that supports 2-legged OAuth.

Demonstration

- Examining the “Plane Crazy” demo site.



Agenda

- OpenSocial Background/Overview
- OpenSocial REST support
- Introducing the REST client libraries
- OpenSocial Security - Oauth
- Using the OpenSocial Java client library
- Building Server-Side Mashups
- Building Mobile Mashups
- Summary

Building Mobile Mashups

with the OpenSocial client libraries

- Any native mobile phone applications that can support OAuth and the REST protocol can communicate with an OpenSocial container

- For Java, this is easy.
  - The OpenSocial Java client library comes with a ready-to-run Android sample application which demonstrates extracting contact info from Plaxo using 3-legged OAuth.

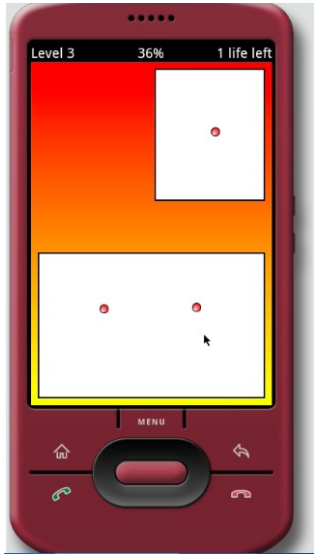
Demonstration

- A simple 3-legged OAuth demonstration from a mobile phone.
  - Accessing Plaxo contacts info from an Android phone.

Building Mobile Mashups

a more advanced Mobile example

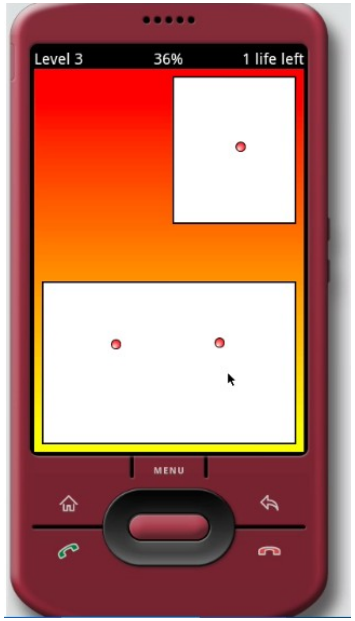
- How an existing mobile game was made *social* with OpenSocial and the REST client libraries



+

Demonstration

- The Social divide and conquer game!



Summary

- OpenSocial supports innovative server-side and mobile integrations via its support of RESTful and RPC protocols
- The OpenSocial Java client libraries take all the heavy lifting out of communicating via these protocols.
- Any mobile, desktop or server-side Java developer can easily integrate OpenSocial and social features into their applications!

Thank You

Chris Schalk
Ryan Boyd

Google Developer Advocates
JavaOne 2009