Sun Glassfish SocialSite Developer's Guide

Alpha

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–6356

081217@21808

Contents

CHAPTER 1

Overview

The goal of social applications is to build online communities of people who share interests and activities, or who are interested in exploring the interests and activities of others. The Sun Glassfish SocialSite software adds social features to existing web applications by providing a server to manage the social graph, as well as a library of social widgets that can be added to an existing web page.

About Social Graphs

With a typical social application such as MySpace, individual users can upload a picture of themselves, create their profile, and designate other users as "friends". Such social applications also offer the ability to create groups that share common interests or affiliations, and track individual/group activities.

Social applications use a social graph that is a global mapping of all users and how they are related such as:

- Personal profile information including contact information, personal tastes, body type, work history, or education
- Relationships between people
  - Friends
  - Business contacts
  - Membership in groups
- Activities for each user
  - Changes to the status or properties
  - New content created by other users
  - Message notifications from other others

Social applications may require a dedicated container (or server) the to manage the various profiles, friends, groups, and activities in the social graph. The OpenSocial (`http://code.google.com/apis/opensocial/docs/0.7/spec.html#concepts`) specification allows social applications to manage social graphs with a set of common application programming interfaces (API) that expose methods to information about people, their friends, and their data. The OpenSocial specifications usually are used to develop social applications from inception to deployment.

In contrast, the SocialSite software is an end-to-end solution that adds social networking features to existing web applications by providing a server to manage the social graph, as well as a library of social widgets that can be added to an existing web page. Since the SocialSite software already complies with the OpenSocial specifications, there is no need to implement the REST API (`http://code.google.com/apis/opensocial/docs/dataapis.html`) and the JavaScript API (`http://code.google.com/apis/gadgets/docs/dev_guide.html`) from scratch to create a social networking application. However, the SocialSite software does extend the OpenSocial REST API and JavaScript API specifications to provide a complete solution.

People

The OpenSocial specification defines the Person object to store information specific to a user. Part of this information is stored in the user's profile (such as hobbies and contact information). The other important user information is the set of connections with other users in the social graph and this is tracked as relationships.

Note – The user ID must be alphanumeric (A-Za-z0-9) and must uniquely identify the user in a container. This standardization is intended to allow for prefixing user IDs with a domain name and separator to create globally unique IDs. Refer to "Authentication Delegation" on page 32for more information on how the SocialSite software extends the OpenSocial specification to provide unique user identities.

There are two Person objects that can be requested directly from an OpenSocial container: the VIEWER and the OWNER. The VIEWER accesses the user profile of another user (who would the OWNER). Likewise, a user can act as both VIEWER and OWNER when the user's own profile is updated.

The standard OpenSocial specifications do not support creating and updating user profile data. Likewise, the standard OpenSocial specifications do not give you any control over who can see which parts of your profile or activities. The SocialSite software provides extensions to manage user profiles ("User Profile Data Extensions" on page 28) and limit access to user profiles ("Profile Privacy Settings Extensions" on page 28).

Relationships

There are two sets of relationships that are defined in OpenSocial: VIEWER_FRIENDS and OWNER_FRIENDS. In the case of accessing another user's profile, a VIEWER_FRIENDS request will return the set of users that are friends with you, while a OWNER_FRIENDS request will return the set of users that are friends with your coworker. If you're viewing your own profile, VIEWER_FRIENDS and OWNER_FRIENDS will be the same set of users. The OpenSocial specification makes no assumptions about the relationship between VIEWER and OWNER.

Note that the OpenSocial specification does not distinguish personal relationships. The VIEWER and OWNER could be friends, but if you're looking at a stranger's profile, there's no relationship between the VIEWER and the OWNER. Here are types of personal relationships that may be supported in social applications:

- A one-way relationship from one person to another person that is not reciprocated.
- A two-way relationship from one person to another that is reciprocated.
- A mutual relationship where two parties have agreed on some shared item of information about the relationship (work for the same manager).

The standard OpenSocial specifications do not support the process of requesting and negotiating some form of social relationship with another user. The SocialSite software provides extensions that manage personal relationships (see "People and Friends Extensions" on page 28).

In addition, the standard OpenSocial specifications do not support any group creation, management or invitation capabilities. The SocialSite software provides extensions to create and manage groups (see "Group Management Extensions" on page 28).

Activities

The OpenSocial specification has the notion of activity streams, or collections of actions that a user has taken in the context of a given container. These activities can include interaction with the container itself, (such as updating a profile) or interaction with an OpenSocial application. Activities can be separated into data and presentation so that multiple activities can combined summaries or bundles that let users know what their friends are up to without having to wade through a flood of messages.

The SocialSite software provides extensions that allow message notifications for individual and group activities (see "Messaging" on page 29).

CHAPTER 2

SocialSite Widgets

As an end-to-end solution that adds social networking features to existing host applications, the Sun Glassfish SocialSite software provides a library of social widgets that can be added to a web page. A SocialSite widget is an OpenSocial-compliant gadget that calls one or more SocialSite-specific extensions.

This chapter provides the following information for each SocialSite widget:

- what the widget does
- displayed information
- actions
- parameters

All Groups Widget

The All Groups widget shows a listing of all groups in the social graph. The All Groups widget uses the SocialSite REST API to obtain the listing.

Displayed Information

List of groups

Actions

Link to a group's homepage

Parameters

`profileID`	Person whose groups are to be displayed
`height`	Height setting
`width`	Width setting

Dashboard Widget

The Dashboard widget provides a common page to access information about users, groups, activities, and message notifications.

Activities Tab

Displayed Information

Filtered list of activities

Actions

List your activities
List friend's activities
Go to previous page
Go to next page

People Tab

Displayed Information

List of friends
Filter search for all SocialSite users
Filter search for all SocialSite users that are considered friends
Filter search for all SocialSite users belonging to a specified group

Actions

Specify user has no relationship
Specify user has a contact relationship
Specify user has a friend relationship

Send e-mail to a user
Invite a user to join a group
Go to previous page
Go to next page

Groups Tab

Displayed Information

List of your groups

Actions

Search for SocialSite groups
Filter search by all SocialSite groups
Filter search by SocialSite groups that you are a member
Create group
Go to previous page
Go to next page

Messages Tab

Displayed Information

List of notification messages

Actions

Display received messages
Display sent messages
Accept request
Ignore request
Go to previous page
Go to next page

Parameters

`height`	Height setting

`width`	Width setting

Face Widget

The Face widget displays a photo of the user.

Displayed Information

Photo of user

Actions

Save
Cancel

Parameters

`height`	Height setting
`width`	Width setting

Group Activities Widget

The Group Activities widget shows the most recent activities of a selected group.

Information Displayed

Icon associated with subject
Link to a Person/Group Profile of subject
Link to Person/Group Profile of object
Time of activity

Actions

Display activities.

Parameters

`groupID`	View activities of this group
`height`	Height setting
`width`	Width setting

Group Members Widget

The Group Members widget displays the members of a group.

Displayed Information

List of group members

Actions

Show which group
Send e-mail

Parameters

`handle`	Group ID
`height`	Height setting
`width`	Width setting

Group Profile Widget

The Group Profile widget displays the profile information of a group. The Group Profile widget uses the SocialSite REST API to obtain the group properties. The Group Profile widget uses the SocialSite REST API to send the Apply to Join request.

Displayed Information

Group logo
Group name
Group description
Group properties (as determined by viewer's membership status)

Actions

Apply-to-join group
Leave group if member and leaving group is allowed
Mail group
Invite user to join group

Parameters

`handle`	Group ID
`height`	Height setting
`width`	Width setting

Owner's Activities

The Owner's Activities widget shows the most recent activities of a user. The Owner Activities widget retrieves user activities using the OpenSocial REST API (in JSON format).

Information Displayed

Activity content
Icon associated with subject
Time of activity

Actions

Display activities

Parameters

`profileID`	View activities of this user
`height`	Height setting
`width`	Width setting

Owner's Friends Widget

The Owner's Friends widget displays the friends of the owner.

Displayed Information

List of user's friends

Actions

Invite friend to join a group
Send e-mail to a friend

Parameters

`profileID`	View activities of this user
`height`	Height setting
`width`	Width setting

Owner's Groups Widget

The Owner's Groups widget displays a listing of the groups that the owner is a member.

Displayed Information

List of groups

Actions

Link to a group's homepage (visibility determined by the owner's membership)
Invite user to join group
Next/previous links if listing is more than one page

Parameters

`profileID`	View activities of this user
`height`	Height setting
`width`	Width setting

People Widget

The People widget browses all people in the social graph. The People widget uses the OpenSocial REST API to fetch the friend list.

Displayed Information

Name of user
Next page/previous page buttons if more than one page of results

Actions

Display profile (per user)
Send e-mail message (per user)
Add As Friend (per user)

Parameters

`height`	Height setting
`width`	Width setting

Profile Editor Widget

The Profile Editor widget edits the profile of a user. The Displayed Information section displays the default values.

Displayed Information

Tab	Attribute	Field
Identification	Nickname	
	Display Name	
	Tags	
	Name	First Name
		Last Name
		Middle Name
		Subtitle
		Unstructured
	Web Sites	
Contact	Primary E-mail	
	Additional E-mail Addresses	
	Instant Messenger Accounts	
Extended	Current Location	Street Address
		Extended Address
		City
		State
		ZIP code
		Country
		Type
	Additional Phone Numbers	
	Addresses	

Tab	Attribute	Field
Personal	About Me	
	Gender	
	Activities	
	Interests	
	Music	
	Movies	
	Books	
	TV	
	Humor	
	Food	
	Fashion	
	Cars	
	Sports	
	Heroes	
	Quotes	
	Religion	
	Political Views	
	Happiest When	
	Scared Of	
	Song	URL Link
		Title
		Type
	Video	URL Link
		Title
		Type

Tab	Attribute	Field
More Personal	Age	
	Birthday	
	Relationship Status	
	Children	
	Drinker	
	Smoker	
	Romance Looking For	
	Sexual Orientation	
	Living Arrangement	
	Turn Ons	
	Turn Offs	
	Body Type	Build
		Eye Color
		Hair Color
		Height
		Weight
Experience	Interests	
	Experiences	
Education	Schools	
Privacy		

Actions

Save changes to profile settings
Cancel changes to profile settings

Parameters

`profileID`	Profile ID to be displayed
`height`	Height setting

`width`	Width setting

Profile Viewer Widget

The Profile Viewer widget displays the profile of a user.

Displayed Information

Public profile summary

Actions

Edit

Parameters

`profileID`	Profile ID to be displayed
`height`	Height setting
`width`	Width setting

Status Widget

The Status widget allows the status of a user to be updated.

Displayed Information

Text box for entering current status and an Update Status button

Actions

Update status

Parameters

`height`	Height setting
`width`	Width setting

Viewer and Friends Activities Widget

The Viewer and Friends Activities widget shows the most recent activities of a user and associated friends. The Viewer and Friends Activities widget retrieves user and friends activities using the OpenSocial REST API (in JSON format). The Viewer and Friends Activities widget retrieves group activities using the SocialSite REST API (in JSON format.)

Information Displayed

Icon associated with subject
Link to a Person/Group Profile of subject (per activity)
Link to Person/Group Profile of object (per activity)
Time of activity

Actions

Display activities.

Parameters

`profileID`	View activities of this person
`groupID`	View activities of this group
`height`	Height setting
`width`	Width setting

Viewer's Friends Widget

The Viewer's Friends widget displays the friends of the viewer.

Displayed Information

List of viewer's friends

Actions

Invite to join a group
Send e-mail

Parameters

`profileID`	View whose friends to display
`height`	Height setting
`width`	Width setting

Viewer's Groups Widget

The Viewer's Groups widget displays a listing of the groups that the viewer is a member.

Displayed Information

List of groups

Actions

Link to a group's homepage (visibility determined by the owner's membership)
Invite user to join group
Next Page/Previous Page links if listing is more than one page

Parameters

`profileID`	Person whose groups to display
`height`	Height setting
`width`	Width setting

CHAPTER 3

SocialSite Extensions

This chapter describes the compliance of the Sun Glassfish Socialsite software with the OpenSocial application programming interface (API) specifications, as well as SocialSite-specific extensions.

OpenSocial Compliance

The SocialSite software is an OpenSocial container that can run any OpenSocial application built using the associated specifications:

- JavaScript API (http://code.google.com/apis/opensocial/docs/0.7/reference/)
- Gadgets API (http://code.google.com/apis/gadgets/docs/spec.html)

The SocialSite software has added extensions that have been defined according to the OpenSocial specifications so that any extra person, activity or other object fields is defied in such a way that allow OpenSocial applications to discover these fields. These extensions are specific to the SocialSite container:

- People and friendships
- User profile data
- User profile privacy
- Group management
- Gadget management
- Messaging
- Search

A SocialSite widget is an OpenSocial-compliant gadget that calls one or more SocialSite-specific extensions.

People and Friends Extensions

The standard OpenSocial specifications do not support the process of requesting and negotiating some form of social relationship with another user.

SocialSite Widget	Actions
Dashboard>People	■ Specify user has no relationship ■ Specify user has a contact relationship ■ Specify user has friend relationship ■ Invite user to join a group

User Profile Data Extensions

The standard OpenSocial specifications do not support creating and updating profile data.

SocialSite Widget	Actions
Profile Editor "Profile Editor Widget" on page 19	■ Save changes to profile settings ■ Cancel changes to profile settings

Profile Privacy Settings Extensions

The standard OpenSocial specifications do not give you any control over who can see which parts of your profile or activities.

SocialSite Widget	Actions
Dashboard>People"People Tab" on page 12	■ Specify user has no relationship ■ Specify user has a contact relationship ■ Specify user has a friend relationship

Group Management Extensions

The standard OpenSocial specifications do not support any group creation, management or invitation capabilities.

SocialSite Widget	Actions
Dashboard>Groups“Groups Tab” on page 13	■ Create group
Dashboard>People“People Tab” on page 12	■ Invite a user to join a group

Messaging

Messaging is included in OpenSocial, but the SocialSite provides extensions to manage the processing of messages.

SocialSite Widget	Actions
Dashboard>Messages“Messages Tab” on page 13	■ Display received messages ■ Display sent messages ■ Accept request ■ Ignore request

Search

The OpenSocial specifications do not support searching the social graph. The SocialSite Widgets must provide the ability to search people, and groups.

SocialSite Widget	Actions
Dashboard>People	■ Search for SocialSite users ■ Filter search for all SocialSite users ■ Filter search for all SocialSite users that are considered friends ■ Filter search for all SocialSite users belonging to a specified group
Dashboard> Groups	■ Search for SocialSite groups ■ Filter search by SocialSite groups that you are a member

CHAPTER 4

SocialSite Host Requirements

The Sun Glassfish SocialSite software is an end-to-end solution that adds social features to existing web applications by providing a server to manage the social graph, as well as a library of social widgets that can be added to an existing web page. There Enterprise Social Server implements the OpenSocial REST API (`http://code.google.com/apis/opensocial/docs/dataapis.html`)and the OpenSocial JavaScript API (`http://code.google.com/apis/gadgets/docs/dev_guide.html`) so that existing web applications do not have to be rewritten from scratch to create a social application.

As implemented, the SocialSite software has the following prerequisites in order for a host application to be socially enabled:

- the host application is capable of rendering dynamic web pages
- the host application must have the authority to verify user identities

Once the prerequisites have been satisfied, a host application can be socially enabled by meeting the following requirements:

- create a page for the context delegation service
- add SocialSite widgets.
- create a page to return user context.

Dynamic Web Pages

The SocialSite software is intended to work with dynamic web pages. Dynamic pages can change content on a web page in response to different contexts or conditions (such as by user role):

- The SocialSite server uses a server-side scripting language (such JSP or PHP) to change the sequence of web pages or web content presented to the browser. The SocialSite server follows the OpenSocial REST API specification to manage:
  - user and friend profile data

- activities
- persistent data

The SocialSite widgets follow the OpenSocial JavaScript API specification to manage the presentation within a web page by changing the interface behaviors in response to mouse or keyboard actions or at specified timing events.

Authentication Delegation

The OpenSocial specification requires that every user ID must be alphanumeric (A-Za-z0-9) and must uniquely identify the user in a container. This standardization is intended to allow for prefixing IDs with a domain name and separator to create globally unique IDs.

The SocialSite server satisfies this requirement by delegating the authority to verify user context to the host application. This means that the SocialSite server does not perform user authentication when there is a request to access the social graph from a widget installed on a user's web page. A host application is responsible for the following:

- It must have some form of user authentication (such as the getRemoteUser method in the code example) so that a request from the SocialSite server to verify a user identity will cause the host application to return a valid user ID. Typically, user authentication consists of a log in page where user names and passwords are verified and stored.
- It must include a file (such as socialsite_context.jsp in the code example) that implements an authentication delegation service for the SocialSite server. The SocialSite server sends a request for this file to your host application using the same cookies as the client would. This is how the host application can assert the identity of the current user to the SocialSite server.

Refer to the article on the SocialSite security model (http://wikis.glassfish.org/socialsite/Wiki.jsp?page=SecurityModel) for more information.

Creating the Authentication Delegation Service

This is an example of what a web application may include as a context delegation service page (socialsite_context.jsp). Although a JSP file is used here, the concepts should directly translate to other page-generation languages (such as PHP):

```
<context>
 <viewer>
  <user name="<%=request.getRemoteUser()%>" />
 </viewer>
 <owner>
  <user name="<%=request.getRemoteUser()%>" />
 </owner>
</context>
```

The sample code returns an XML representation of the identity context for the current user's session. Note that the "context delegation service" is not required to be included within the host application, In this example, it is convenient to include the context delegation service within the host application to leverage a local HttpSession context.

Adding the SocialSite Widgets to the Host Application

Here is an sample of what a host application may include in its source code to add a SocialSite widget (profile.jsp):

```
01: <html>
02:  <head>
03:   <script type="text/javascript" src="http://socialsite.example.com/js/consumer.jsp"></script>
04:   <script type="text/javascript">
05:    socialsite.setContext({
06:      ’attributes’: {
07:        ’ownerId’: ’${ownerProfile.userId}’
08:      },
09:      ’delegate’: {
10:        ’method’: ’GET’,
11:        ’url’: ’&lt;%=request.getContextPath()%>/socialsite_context.jsp’,
12:        ’headers’: {
13:          ’cookie’: document.cookie
14:        }
15:      }
16:    });
17:   <script>
18:  <head>
19:  <body>
20:   <script type="text/javascript">socialsite.addGadget({’spec’:’http://example.com/gadget.xml’});</script>
21:   <script type="text/javascript">socialsite.addGadgets({’collection’:’PROFILE’});</script>
22:  </body>
23: </html
```

Line 03	The host application pulls in the necessary JavaScript code from the SocialSite software.
Lines 04-17	The host application directs the SocialSite software how to determine user context for the widgets. In the above example, the "delegate" entry tells the Social Server that it should makes its own HTTP GET request to another server-side URL ("socialsite_context.jsp" on the consumer site) to look up context settings.

Lines 20 and 21	The host application adds widgets to a page. This can either be done by explicitly referencing a widget's URL (as shown in line 20) or by referencing an identifier that should key into a collection of gadget instances that is known to the SocialSite software (as shown on line 21).

Creating a Page to Return User Context

The socialsite_context.xml file needs to be to the classpath of the SocialSite Enterprise software to specify the location of the context delegation service in order to assert a user ID. Here is a sample file:

```
<rules>
?
    <rule>
?
    <sources>
<indirect>*</indirect>
</sources>
?
    <attributes>
<accept>*</accept>
<reject>viewerId</reject>
</attributes>
</rule>
?
    <rule>
?
    <sources>
?
    <direct>
http://socialsite.example.com/socialsite_context.jsp
</direct>
</sources>
?
    <attributes>
<accept>*</accept>
</attributes>
</rule>
</rules>
```

Note that this file contains the URL of the context delegator service that you created. You can add this file to your classpath by copying it to your glassfish/domains/domain1/lib/classes/ directory, which should already contain a socialsite.properties file that was added during installation. Restart the GlassFish server to include the changes.

Index