SPARKANGELS APIS

API DESCRIPTION

Document Info:

Author(s)	PhM
Type	User API Description
Distribution	Externe
Number	T-1101

Revision History:

Revision	*Date*	*Comments*
1.0	20/11/11	Initilal version
2.0	3/11/12	Added Widgets, Accounts

Table of Contents

1. INTRODUCTION

1.1. OVERVIEW

This document presents some of the APIs exposed by SparkAngels.

These APIs are mainly provided by two servers:

- The PAN Server, responsible for the Presence And Notification services
- The RSS Web Application, responsible for Account management

The APIs exposed, for use by third parties with Sparkom permission, provide the following functionality:

Server Status

- Probe the communication with the server

Widget Group

- Acquisition of the presence status of an Access Point (identified by its **WidgetID**)
- Acquisition of the presence status of multiple Access Points at once
- Acquisition of the Accounting Model of an Access Point
- Acquisition of Extended Code-Based Widget information

SparkAngels Launch API

- Session startup request options

Pro (Accounting) Code management

- Query a Code status
- Create a Code
- Update a Code

Other APIs are internally used by Sparkom (in Web or Web Service form), and may be made available if a need is identified, notably in the areas of :

- *Personal Network Group* : management and acquisition by an User of Persnal Network information
- *Corporate Accounts Management* : Create/Update/Delete accounts within a Corporate environment

- *Corporation Management* : Querying Corporation Configuration data
- *Session Reporting*: Reports of Lists of Sessions (paginated)
- *Mini-CRM* : Acquisition, Record and Update of mini-CRM data (user identification and description, session notes...)
- *Notations* : Helper and Helpee side Notations (as enabled for the Widget).
- *InMail* : Sending and Retrieving Internal Mail and Mailboxes

1.2. Security Model

For all the APIs that require Authentication, the Login parameters must be supplied in the request, as indicated. For this reason, these APIs should only be invoked in HTTPS mode.

For specific deployments, a Security Token-based model may be provided if needed.

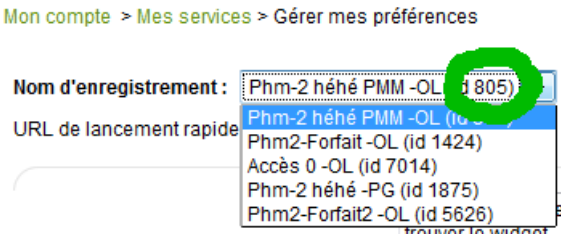
1.3. Character Encoding

All the APIs use the UTF-8 Character encoding scheme.

1.4. Identifiers used in the APIs

The **WidgetID** identifies uniquely a Service **Access Point** (also named "**Preference**" in SparkAngels' site). Several individual accounts can be associated with one Access Point [service providers]: they will all be notified when an user requests a session with this Access Point, and their individual presence status will be used for computing the presence status of the Access Point.

In SparkAngels' website, the **WidgetID** is indicated as the ID of the Record Name of an access point:



The **LoginAccountID** identifies uniquely an individual Account.

2. Server Status

2.1. Probe the server status

URL: **http://www.spark-angels.com/panserver2/probe**

Functionality: Allows to check the connectivity and status of a Server.

Returns: A JSON Object with the following fields:

message (string): “OK” or an error String.
retCode (int): 0 or an error code.

3. Widget Group API

3.1. Acquisition of the Presence Status of a Widget

URL: **http://www.spark-angels.com/panserver2/getpresencedynamic?widgetId=widgetId**

Functionality: Returns the Presence Status of an Access Point (Widget), computed from the Presence Status of the associated individual accounts.

Returns: A JSON Object with the following fields:

message (string): “OK” or an error String.
imageUrl (string): A pseudo-URL ending with one of the following strings: “**available.png**”, “**busy.png**”, “**notLogged.png**”, “**futureLogged**”, “**serviceIndisponible.png**”.

If the message value is “OK”, and when the imageUrl field contains “futureLogged”, the **nextOnlineDeltaTime** (integer) field indicates when the Access Point will become available (number of seconds from now).

3.2. Acquisition of the Presence Status of several Widgets

URL: **www.spark-angels.com/panserver2/querymultiplepresence?wl=widgetId;widgetId**

Functionality: Returns the Presence Status of a series of Access Points. The requested Widgets are listed in the 'wl' query parameter, and separated with the '**;**' character.

Returns: A JSON Object with the following fields:

message (string): “OK” or an error String.
retCode (int): 0 or an error code
nb_item (int): The number of Widget presences returned.

And for each item, named **item_i** (object):

widgetid (int): WidgetID of the widget

presence (int): 1=NOT Logged, 2=Available, 3= Busy, 4=Future Logged

nodt (int): In Future Logged case, the number of seconds from Now when the widget will be available.

3.3. Acquisition of the Accounting Model of a Widget

URL: **www.spark-angels.com/panserver2/gethelperpricechoices?widgetId=widgetId**

Functionality: Returns the Accounting information for the selected Widget. This information indicate whether the Widget requires an Authentication (by Spark-Angels Login Account or Code) or not, and its pricing model. Note that Billing information can be ignored in Corporate environments.

Returns: A JSON Object with the following fields:

message (string): "OK" or an error String.

retCode (int): 0 or an error code

widgetType (String):

"provar": No Authentication required. The session will be billed to the Helper account.

"prope": Authentication by Code required. Session billed to the Helper account.

"propi": Authentication by SparkAngels Account required. Session billed to the Helpee account.

"ifree": No Authentication required. Session will be billed as a "Free" session (subject to monthly limits and extended by paid Traffic Packs)

"ppvcn": No Authentication required. No Billing, only accounting (Reserved for prepaid cases)

"ppecn": Authentication by Code required. No Billing, only accounting (Reserved for prepaid cases)

"ppicn": Authentication by SparkAngels Account required. No Billing, only accounting (Reserved for prepaid cases)

selectedPrice (string): The princing value according to the pricing model.

jnlpurlforsapi (string): Base URL used for constructing the SparkAngels launch URL (takes into account the Specific Dynamic Application associated with the widget, if any).

3.4. Acquisition of Extended Code-Based Widget information

URL: **www.spark-angels.com/panserver2/CheckPropeCode?wid=WidgetID**
URL: **.../CheckPropeCode?wid=WidgetID&pecode=Code**
URL: **.../CheckPropeCode?tolaid=LoginAccountId**
URL: **.../CheckPropeCode?wid=WidgetID&members=1**
URL: **.../CheckPropeCode?wid=WidgetID&members=1&pecode=Code**

Functionality: Returns extended information about an Access Point that is configured in "Code-based mode" (a.k.a. "Requires Authentication by Code", or "prope"), including its Presence Status. Also allows to check whether a given Code is valid, and query the available time on this code. Also Allows to obtain the individual Presence status of accounts associated with this Wigdet.

Note: the Non-Code related functionality (widget and account presence, list of members) will work on all types of widgets.

Returns: A JSON Object with the following fields:

message (string): "OK" or an error String.
retCode (int): 0 or an error code

*Only the **wid=WidgetID** parameter is given:*
presence (int): Presence Status of the Widget (1=Not_Logger, 2=Available, 3=Busy, 4=FutureLogged)
nodt (int): In Future Logged case, the number of seconds from Now when the widget will be available

*Only the **tolaid=LoginAccountID** parameter is given:*
Same behaviour as with the WidgetID, except that the Presence considered is the Presence of the specified individual Account, not the aggregated Widget Presence.

*Both the **wid=WidgetID** and **pecode=Code** parameters are given:*
presence (int): Presence Status of the Widget (1=Not_Logger, 2=Available, 3=Busy, 4=FutureLogged)
nodt (int): In Future Logged case, the number of seconds from Now when the widget will be available
minutes (int): The number of Minutes available for this Code.
obcode (string): Obfuscated Code for use when Launching SparkAngels with pre-identification

*The **wid=WidgetID** and **members=1** parameters are given:*
presences (array): Array of the Presence Status of the Accounts associated with this Widget.
For each Account :
laid (int): The LoginAccountID that identifies the account
pres (int): Presence Status of the Account (1=Not_Logger, 2=Available, 3=Busy, 4=FutureLogged)
nodt (int): In Future Logged case, the number of seconds from Now when the widget will be available

*The **wid=WidgetID** and **members=1** and **pecode=Code** parameters are given:*
***presences** (array): as above*
***minutes** (int) and **obcode** (string): as above*

4. Pro (Accounting) Code management

4.1. Acquisition of the informations related with a Code

URL: **www.spark-angels.com/SHAPIserver/get_pcid_data/CODE/**
?helperLogin=LOGIN&helperPwd=widgetId

Functionality: Returns the informations related to a given Code.

Returns: A JSON Object with the following fields:

message (string): "OK" or an error String.
retCode (int): 0 or an error code
helperNotes (string): Notes on this Code.
helperCredits (int): Number of Credit Units on this Code.

A Credit Unit is a 5 minutes session duration credit.

4.2. Acquisition of the informations related with an Helper

URL: www.spark-angels.com/SHAPIserver/get_all_pcid_data/
?helperLogin=LOGIN&helperPwd=widgetId

Functionality: Returns all the CODE informations related to a given Helper.

Returns: A JSON Object with the following fields:

message (string): “OK” or an error String.
retCode (int): 0 or an error code
numProClientIds (int): Number of Codes returned

For each Code, a JSON object named **proclient.<i>** (i from 0 to numProClientIds-1):
proClientID (string): the Code
helperCredits (int): the number of Credit Units
helperNotes (string): the Notes on this Code.

4.3. Creation of a Code

URL: www.spark-angels.com/SHAPIserver/create_pcid/
?helperLogin=LOGIN&helperPwd=widgetId&proClientId=CODE

Functionality: Creates an Empty Code.

4.4. Deletion of a Code

URL: www.spark-angels.com/SHAPIserver/delete_pcid/
?helperLogin=LOGIN&helperPwd=widgetId&proClientId=CODE

Functionality: Deletes an existing Code.

4.5. Update of a Code

URL: www.spark-angels.com/SHAPIserver/set_pcid_data/
?helperLogin=LOGIN&helperPwd=widgetId&proClientId=CODE
&addedHelperCredits=<NUM>&helperNotes=<Optional_Notes>

Functionality: Updates the number of Credits (and Optionally the Notes on the Code) associated with an existing Code.

5. SparkAngels Launch API

5.1. Overview of SAPI-2 parameters

The SparkAngels can be started with a number of parameters, regrouped in the “SAPI-2” interface.

The Launch URL is constructed as follows:

```
http://dnld0.sparkom.com/sparkangels/sangels/sparkangels.jnlp?sapi2=<SAPI2-VALUE>
```

The SAPI2-VALUE is composed of the following fields, separated by ',' (comma). Each field is composed of a letter (field type) concatenated with the field value. Text values are URIComponent-encoded.

The Application part can be replaced with the URL of a specific Customized Application – This URL can be returned, if not known already, as the `jnlpurlforsapi` value in a ***gethelperpricechoices*** request.

Field	*Value*
a	**RSS2**
v	**2**
c	**Command:** i[dentify], r[ecord], s[tart], q[uerysapi], **w[idgetsapi],** b[uytrafficpack], **c[widgetsapi+followup Webchat]**
w	**WidgetID – Launch a session towards this Widget, in 'w' or 'c' command modes**
h	**HelperLoginID – Target a specific Helper behind a WidgetID in 'w' or 'c' command modes**
s	**Webchat SessionID when the Session is started from WebChat, in 'c' command mode**
l	**Login of the Helpee, URI-Encoded – Pre-Identification of the Helpee, in 'w' or 'c' modes**
p	**Password of the Helpee, URI-Encoded – Pre-Identification of the Helpee**
f	**Obfuscated Account – Pre-Identification of the Helpee (replaces login/password)**
e	**Obfuscated Widget Code, Pre-Identification of the Helpee in Code Mode** (see CheckPropeCode)
d	**DistributorID – Follow-up id. of the Distributor through which the session was initiated**

Example:

```
sapi2=aRSS2,v2,cW,w1234,lphilippe+martinou%40sparkom.com,pPASS
sapi2=aRSS2,v2,cW,w1234,e12345
```

The URI-Encoding of text values can be realized in JavaScript using the `encodeURIComponent` method, or in Java using `java.net.URLEncoder.encode(s,"UTF-8");`

The Charset is "UTF-8", as in all SparkAngels API methods.

5.2. Direct Connection mode

This is a mode of operation where two SparkAngels are launched independently, and connect themselves using a common “cross-reference code”. The SAPI parameters associated with this mode of operation are described in the document **“SparkAngels_SAPI_v5.0x.pdf”**.

5.3. SESSION STARTUP IN WIDGET MODE

To initiate a relationship towards the Access Point identified by a **WidgetID**, the methods described below can be used: either navigate to an URL (web context) or start a process (desktop application context).

In both cases, the SparkAngels application will start locally (and be downloaded if needed) and initiate the session towards the specified Access Point. The Available helpers associated with this Access Point will be notified, and the session will be established with the first one who accepts the request.

Numerous variants and options can be configured for an Access Point, to control for example:

- whether an authentication is requested (user account or private access code)
- what services are started initially (Screen Sharing, Chat...)
- whether a personalized application is to be launched
- whether a description of the request is asked before the session
- and more...

Typically, the basic startup command will be:
`sapi2=aRSS2,v2,cW,w<WIDGETID>`

and the following parameters can be added:

`,h<TargetLoginAccountID>` to specify the Individual Account behind the Widget.

`,s<WebChatSessionID>`, to link this Session with a webchat session initiated earlier.

And the helpee can be identified either by a Code (for a Widget in Code mode):

`,e<CODE>` to supply the Code and immediately start the relation, in Code mode.

Or by an Account:

`, l<LOGIN>,p<PASSWORD>  or  f<LOGIN/PASSWORD>` to supply the Login/Password of an Account