Profile Management - IDS

Overview

Profile Management (IDS) is a cloud-hosted database that consolidates user data acquired from various sources – e.g., social and traditional registration, site activity, multiple domains, or mobile apps. It enables full control over your user database, allowing you to easily search, view, edit, update, delete, target, and export user segments. You can access the users' data using Web tools as well as API methods.
Gigya maintains database performance and security, and even offers an automated way to stay in compliance with social networks so you can be confident that your data will always be accessible, safe, and updated to meet social network policies.

Major Features

- **Social Sync** - The user's social networks profile data, which is available after Social Login/Registration is automatically imported and stored in Profile Management and available for searching. In addition, whenever your Facebook users update their profiles on Facebook, the changes will sync automatically to Profile Management. Gigya automatically subscribes to the following Facebook fields: *birthday, books, education, email, first_name, hometown, last_name, likes, link, locale, location, movies, music, name, relationship_status, religion, verified, timezone* and *work*. Please note that Gigya complies with the social networks' platform policies.
- **Dynamic Schema** - The storage is built with a dynamic schema that can seamlessly process massive amounts of user data in an optimized way. Having a dynamic-schema means that you may store any custom data you have with no constraints on its structure, you don't have to know in advance how your custom data is going to look and it doesn't have to look the same for all objects. There is no need to go through schema creation or modification when the data structure changes.
- **Fully indexed** - The storage immediately indexes your data so you can search your entire user base using any combination of user attributes. Its SQL-like interface simplifies queries, providing a flexible way to retrieve user segments at scale. You don't need to plan your queries or create the appropriate index in advance.
- **Security and Performance** - Gigya's platform is Safe Harbor Certified, ISO27001 compliant, regularly scanned by PCI-approved scanning vendors, proven in high-scale enterprise environments, and holds an industry-leading uptime. Gigya also owns and operates its own hardware, with real-time data backup.
- **Social Compliance** - Gigya automatically manages compliance with Facebook's data management requirements.
- **Data Import** - You may import legacy data from an existing database, as well as data collected via traditional user workflows such as registration and newsletter sign-up.
- **Data Export** - Gigya can deliver regular extractions of new and changed data, tailored to your requirements.

Watch an Instructional Video

If you have an SAP logon, you can watch an instructional video about Identity Management here.

Web Access

Profile Management (IDS) supports full control over your user database using web-based tools:

- Identity Access - A site admin tool for accessing and editing user data. As a site administrator, you can view users' profiles, social data as well as site custom data. You can also search users, edit and delete user's information.
- Identity Query Tool - An intuitive and dynamic tool for searching and retrieving users' profile data, enabling people without DB knowledge to create a search query using simple drop-down menus.

API Access

Profile Management (IDS) provides an API for accessing the user data stored by Gigya. The user data stored includes Gigya's predefined fields, social graph, and a site's specific custom fields. All this data can be accessed using a single interface that includes the following API methods:

- ids.search - Allows querying Profile Management using simple SQL-like queries. Read more below.
- ids.getAccountInfo - Retrieves a user account.

Note: The ***regToken*** parameters of this method is not supported (it is only supported when implementing the Registration-as-a-Service p ackage).

- ids.setAccountInfo - Stores data in a specified user account.
  Note: Only the following parameters of this method may be used: ***UID***, ***profile***, ***data***, ***format***, ***callback***, ***httpStatusCodes*** (the other parameters are supported only when implementing the Registration-as-a-Service package).
- ids.deleteAccount - Deletes a user account.
- ids.getSchema - Retrieves the current schema of the Profile Management.
- ids.setSchema - Allows specifying a schema for specified fields. The schema defines a set of properties for certain static data fields.

Note: You may use one of our Server Side SDKs to implement the API usage. Alternatively, you may use direct REST API calls if there is no SDK available for your preferred language. For security reasons, most of the API is available only for server to server calls. From client SDKs, you only may access the account of the currently logged-in user (ids.getAccountInfo, ids.setAccountInfo).

Querying

Gigya provides an SQL-like query language for searching and querying Profile Management. The ids.search method receives a '*query*' parameter that accepts an SQL-like query string. For details of the query syntax, see the ids.search documentation.

Notes:

- Always prefer accessing user accounts directly using their UIDs (with ids.getAccountInfo) rather than using search, whenever possible. Accessing user accounts directly using their UIDs is faster and optimal for run-time.
- After data is published to the storage, building the indexes is done asynchronously for performance reasons, and may take up to one second to complete.

The User Account

Users' accounts are automatically created in Profile Management (IDS) after Social Login/Registration. For example, when a user uses Gigya's Lo gin plugin to log into your site: if the user is new, Gigya will create a new account for them in Profile Management and populate it with their social data. If the user is a returning user, Gigya will refresh the social data in their account.
To delete a user's account, please use the socialize.deleteAccount method.

The accounts in Profile Management have a certain structure. On the highest level a user's account includes:

- A set of account meta-data attributes (detailed below).
- Profile object - The user's profile information (name, age, etc.), a consolidation of social and site information.
- Data object - This is the place for you to insert any user-related custom data that isn't part of the Profile object. The structure of the Data object is open, flexible and dynamic. Use accounts.setSchema to define static fields in the object.
- Identities - An array of objects that represent the user's social identities. Each Identity object contains imported data from the corresponding social network.

Account Structure

Field	Type	Description
UID	string	The unique user ID. This user ID should be used for login verification as described under Validate the UID Signature in the Social Login Process.
loginProvider	string	The name of the provider that the user used in order to log in.
iRank	integer	Influencer rank of the user. The iRank is a number between 0-99, which denotes the percentile location of the user in comparison to all other site users as a site influencer. For example, if a user's iRank equals 60, this means that 60% of the site users influence less than this user, or in other words, this user is in the top 40% of site influencers. The iRank is calculated based on the amount of exposure this user provides the site. The calculation is done for the past several months, where recent activities receive higher ranks. The iRank is per site (per API key); the same user may have different ranks for different domains. The iRank calculation uses the following parameters: • The number of **friends** this user has in all the networks to which they are connected through this site. • The number of times this user **shared** something in this site (per month). • The number of **click backs** that were made as a result of this user's shares.

lastLoginTimestamp	integer	The time of the last login of the user in Unix time format, i.e., seconds since Jan.1st, 1970.
lastLogin	string	The time of the last login of the user in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
oldestDataUpdatedTimestamp	integer	The time when the oldest data of the object was refreshed in Unix time format, i.e., seconds since Jan. 1st, 1970.
oldestDataUpdated	string	The time when the oldest data of the object was refreshed in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
lastUpdatedTimestamp	integer	The time when the last update of the object occurred (either full or partial update) in Unix time format, i.e., seconds since Jan. 1st, 1970.
lastUpdated	string	The time when the last update of the object occurred (either full or partial update) in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
createdTimestamp	integer	The time the account was created in Unix time format, i.e., seconds since Jan. 1st, 1970. For Registration as-a Service accounts this is the time accounts.register was called, for Social Login accounts this is the first time socialize.login or socialize.notifyLogin were called.
created	string	The time the account was created in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
socialProviders	string	A comma-separated list of the names of the providers to which the user is connected/logged in.
profile	Profile object	The user's profile information as described in the Profile object.
data	JSON object	Custom data. Any data that you want to store regarding the user that isn't part of the Profile object.
identities	array	An array of Identity objects, each object represents a user's social identity. Each Identity object contains imported data from a social network that the user has connected to.

A field that does not contain data will not appear in the response.

A Sample Account in a JSON Object Representation:

```
{
  "UID": "e862a450214c46b3973ff3c8368d1c7e",
  "loginProvider": "facebook",
  "socialProviders": "site,facebook",
  "profile": {
    "email": "Joe@hotmail.com",
    "firstName": "Joe",
    "lastName": "Smith",
    "age" : "31",
    "gender" : "m",
    "country" : "US"
  },
  "data": {
    "newsletter": "true",
    "forums" : "news,entertainment"
  },
  "identities": [
    {
      "provider": "site",
      "providerUID": "e862a450214c46b3973ff3c8368d1c7e",
      "isLoginIdentity": false,
      "allowsLogin": false,
```

```
      "isExpiredSession": false,
      "lastUpdated": "2012-08-08T08:07:59.133Z",
      "lastUpdatedTimestamp": 1344413279133,
      "oldestDataUpdated": "2012-08-08T08:07:59.133Z",
      "oldestDataUpdatedTimestamp": 1344413279133
    },
    {
      "provider": "facebook",
      "providerUID": "khgfkhg83wd897y",
      "isLoginIdentity": true,
      "gender": "m",
      "email": "Joe@hotmail.com",
      "firstName": "Joe",
      "lastName": "Smith",
      "age" : "31",
      "country" : "US"
      "allowsLogin": true,
      "isExpiredSession": false,
      "lastUpdated": "2012-09-08T08:07:59.133Z",
      "lastUpdatedTimestamp": 1344413274523,
      "oldestDataUpdated": "2012-09-08T08:07:59.133Z",
      "oldestDataUpdatedTimestamp": 1344413276453
    }
  ],
  "created": "2012-08-08T08:07:59.128Z",
  "createdTimestamp": 1344413279128,
  "lastLogin": "2012-08-08T08:09:17Z",
  "lastLoginTimestamp": 1344413357000,
  "lastUpdated": "2012-09-08T08:07:59.133Z",
  "lastUpdatedTimestamp": 1344413279133,
```

```
  "oldestDataUpdated": "2012-08-08T08:07:59.133Z",
  "oldestDataUpdatedTimestamp": 1344413279133
}
```