Solution home Elements Implementing Elements

HR Data Import

Modified on: Thu, 2 Feb, 2017 at 10:24 AM

This article begins by defining the HR data which the system requires and explains how this data is then processed in *Elements*. After reviewing, read the article ***Elements* User Groups** (https://support.symplectic.co.uk/support/solutions/articles/6000126324), which discusses how the users imported via the HR feed are organised into Groups.

Glossary

FEED refers to all the data fed to *Elements* via the import-user REST service.

USERS refers to all the HR data in the *Elements* user table.

HR Data Import

Elements comes equipped with a set of REST services which allow human resources (HR) data to be imported into the system.

The HR data required by the system allows users to access the *Elements* website, receive emails from the system and to be classified appropriately so that the system interacts with them in a correct manner.

There are two distinct steps involved in keeping the list of users of *Elements* up to date:

1. First the user feed needs to be populated via a REST service (the authority and schedule for this action lies outside the system).
2. Then the user feed needs to be processed within *Elements* to update the system's user table (the authority and schedule for this action lies within the system).

Details of the REST service which is used to load the HR data into the system are available in the ***Elements* API forum** (http://support.symplectic.co.uk/support/solutions/folders/6000177986). More information is in the **Managing Users** (http://support.symplectic.co.uk/support/solutions/articles/6000049842-managing-users) and **Managing User Groups** (http://support.symplectic.co.uk/support/solutions/articles/6000049839-managing-user-groups)articles.

Data requirements

This table defines the data that is expected by the import-users REST service. The data requirements for each field are discussed further in the following sections.

Field name	Type	Length (characters)	Mandatory	Example
Title	nvarchar	50	N	Prof
Initials	nvarchar	50	N	I
Firstname	nvarchar	100	N	Isaac
Lastname	nvarchar	500	Y	Newton
KnownAs	nvarchar	100	N	Isaac
Suffix	nvarchar	50	N	FRS
Email	varchar	320	Y	i.newton@org.ac.uk
AuthenticatingAuthority	nvarchar	50	Y	ORG
Username	nvarchar	32	Y	inewt
Proprietary_ID	nvarchar	100	Y	0123456789

Field name	Type	Length (characters)	Mandatory	Example
PrimaryGroupDescriptor	nvarchar	100	N (but recommended)	Physical Sciences
Position	nvarchar		N	Professor
Department	nvarchar		N	Physics
Generic01–Generic50	nvarchar	Unlimited	N	REF Eligible
IsAcademic	bit	1	Y	1
IsCurrent	bit	1	N (default=1)	1
LoginAllowed	bit	1	N (default=1)	1
ArriveDate	datetime		N	2001-01-10
LeaveDate	datetime		N	2007-06-15

Note: When uploading values in a comma-delimited format, if you have data where a comma is part of the data (and not a delimiter), enclose that data in quotes.
*Example: Dr.,AM,Alan,Turing,,**"OBE, FRS"**,aturing@lit.ac.uk,LITAuth,aturing,4455667788,Faculty of Computer Science*

Active users

We define active users as those that have "IsCurrent" set to true.

Proprietary_ID

Proprietary_ID is the organisation-wide unique identifier for each user and is essential when feeding HR data to Symplectic *Elements*.
Once processing of the FEED is initialised within Symplectic *Elements*, USERS will be matched to the FEED via Proprietary_ID.
All rows already in the database having a Proprietary_ ID that does not exist in the FEED will have both "IsCurrent" and "LoginAllowed" set to false, i.e. they will be made inactive and will not be able to log into the system.
All rows in the FEED which have a Proprietary_ID that does not appear in the user table will be imported into Publications as new users, i.e. added to USERS.
All other Proprietary_IDs that are matched across the two tables will be used to update the data in USERS.

Initials

It is important that the Initials field includes all initials for the users, not just their middle initial. So, for example, Charles R Darwin would have 'CR' in the initials feed, not just 'R'.
The system will use the initials as entered for shortened versions of users' names and – importantly – in their initial search terms.
Where a middle initial is needed within the interface, the system will disregard the first character in this field and use the remaining characters in place of the middle initial.

***Note:** the treatment of initials detailed above may have some bearing on how you supply the initials field to users with 'compound', punctuated first names, e.g. Jean-Claude Duvall. To get the correct initials fed to the online data sources for this user, the best way of supplying the initials field is therefore "J-C", so that "J-C Duvall" is used in searches. This will mean, however, that occasionally within the system interface the user will be referred to as "Jean-Claude -C Duvall".*

Authenticating Authority

The field AuthenticatingAuthority enables an organisation to use more than one authentication system for their users.

For example, an organisation's faculties may each have their own authentication systems. Each user must be assigned an AuthenticatingAuthority string, a label of choice for which there should be a corresponding configuration within Symplectic *Elements*, which will instruct the system how to authenticate the user (i.e. which system to contact and which protocol to use).
For organisations where there exists a single authentication system for all users this label becomes irrelevant. However, a non-NULL value must still be assigned and consistently used.
The process of configuring an authentication system is explained in **Manage Authentication of Users (https://support.symplectic.co.uk/support/solutions/articles/6000049892)**.

Primary Group Descriptor

This is a string that allows Symplectic *Elements* to allocate each user to a Primary Group.
The special nature of Primary Groups lies in the ability to configure various components of Symplectic *Elements* to act differently for different Primary groups.
A good example of this is the 'help' page, which contains text that can be customised from within the system: you may wish people in different Primary Groups to see different people listed as a support contact.
Each user of the system therefore has to explicitly belong to one and only one Primary group, the membership of which is determined by the Primary Group Descriptor field.
Note that NULL is permitted in this field, in which case the user's Primary Group will be the top-level 'Organisation' group. This is also true for any users where their Primary Group Descriptor does not match any of the Primary Groups configured within *Elements*.
Due to the complexity involved in configuring various settings across large numbers of Primary Groups, it is recommended to limit their number to somewhere on the order of 5-10. However, this is only a guideline, and can vary depending on the institution's size and structure.
A more complete discussion on the importance and fundamental nature of Primary Groups in *Elements* is included within the User Groups section at the end of the document.

Generic fields and auto groups

Fifty Generic fields may be used to provide additional data relevant to the individual, such as codes or eligibility information for Assessment or research groups, which may be useful for upload to *Elements*.
The first ten of these fields are considered public and can thus be viewed by other users of the system and available to any user of the *Elements* API. Conversely, the remainder can only be viewed by users with appropriate administrative rights.
Generic fields can be used for creating Auto Groups on which a more complete discussion can be found in the User Groups section at the end of this document.
Not all fields need be used, although the structure and nature of data supplied in these fields must be consistent across the whole range of users.
Examples of data your institution might want to include in these generic fields are:

- Country-wide Identifier
- Faculty or School - if not assigned at the Primary Group Level
- Assessment Eligibility, Review Group or other codes to create Auto Groups
- Research group
- Research institute
- Job title
- Honorary/Visiting status
- Institution arrived from
- Institution departed to

IsAcademic

This is a Boolean field, used when a user is newly imported into the system to determine whether or not default search settings should be created for them, and whether online databases will be enabled for that user by default. It is also used within the reports section of the system to filter users supplied in reports.

Constraints

Note that with the exception of Feed_ID, the REST service will allow import of empty or NULL mandatory data. However, all rows which are missing any mandatory data will be dropped during the processing of the FEED, in other words those rows will not make it to USERS. Within USERS, the following constraints apply:

1. **Proprietary_ID is non-NULL and unique for all non-local users.** Local users are allowed to have a NULL Proprietary_ID, as they are not updated via the FEED.
2. **The pair (Username, Authenticating Authority) is unique for all active users.** Otherwise, Symplectic *Elements* would not be able to determine which user is logging into the system.

Local users

Apart from loading users via the API from a HR feed, users can be maintained locally in Symplectic *Elements*.
Local users can be created within the system and users imported originally via the API can be subsequently marked as local.
Once local, the particular user details are not updated by the automated HR feed processing, but have to be maintained manually by a System Administrator.
Local users (created manually or switched from being non-local in the past) can be switched to become non-local. From then on their data will be maintained by the automated HR feed.

Feed processing logic

The steps below outline the schedule used by Symplectic *Elements* when processing the FEED. They show where problems might occur when data are missing from the FEED, and what the system will do with records missing particular fields, with duplicate values or with other problems.

FEED cleanup stage:

1. All fields that contain empty strings are set to SQL NULL.
2. If KnownAs is non-NULL and is equal to FirstName, it is set to NULL.
3. All rows with Proprietary_ID = NULL are discarded.
4. All rows with Username = NULL are discarded.
5. All rows with AuthenticatingAuthority = NULL are discarded.
6. All rows with Email = NULL are discarded.
7. All rows with LastName = NULL are discarded.
8. All rows that have a non-unique (Username, AuthenticatingAuthority) pair are discarded.
9. All rows that have a non-unique Proprietary_ ID are discarded. At this point, the FEED contains only rows with non-NULL and unique Proprietary_IDs. USERS contain non-NULL and unique Proprietary_IDs by design (for all non-local users).
10. All rows in FEED which have a Proprietary_ID equal to any Proprietary_ID belonging to a user marked as local in USERS are discarded. This is to prevent any updates to a local user happening from outside the system.
11. All rows in FEED which have a (Username, Authenticating Authority) pair equal to any such pair belonging to a user marked as local and active in USERS are discarded.

This is to prevent an import of a user from outside the system whose login credentials would clash with an existing, active local user.

FEED checking stage:

1. We count the number of remaining rows in the FEED (after all the processing applied in the above steps is completed) whose "IsCurrent" and "LoginAllowed" settings would make the user active. We call this number "feed active".
2. We calculate the number of non-local active rows in USERS, and call this number "users active".
3. We count the number of rows obtained by joining the active FEED rows to the non-local active USERS rows on the Proprietary_ID. We call this number "overlap active".
4. The value ("feed-active" – "overlap active") is calculated; it denotes the number of users that will be reactivated or created if the feed processing is finished. The value ("users active" – "overlap active") is calculated, which denotes the number of users that will be deactivated if the feed processing is finished. If the sum of the two numbers ("feed-active" + "users active" – 2*"overlap active") exceeds a "cutoff" value (a setting manipulated by system administrators in the Symplectic *Elements* administration front end), further processing is aborted and logged.

The system can be configured to send an email to nominated administrators to inform them that the feed processing has aborted.
At this point we have checked that processing the FEED (thereby modifying the USERS) will not result in altering the number of active users by more than the cutoff value.
This cutoff is implemented to ensure that potentially large errors in the FEED do not propagate into Symplectic *Elements*. For example, the FEED may simply have been delivered with all Proprietary_IDs missing or all Usernames the same. This would result in the FEED effectively containing 0 rows after steps 3 or 8 are applied to it, and would result in the deactivation of all USERS.

FEED processing stage:

1. All non-local, active rows in USERS, whose Proprietary_ID does not appear in FEED are marked as inactive.
2. All non-local rows in USERS whose Proprietary_ID appears in FEED are checked for any change in data.
3. Each row in USERS for which a change is detected is updated with all data from its corresponding row in FEED.
4. All rows in FEED whose Proprietary_ID does not appear in USERS are inserted into USERS (with all corresponding data).