Entity Registry Updates April 2013 / Bundle 29

Introduction

The April bundle introduces many enhancements to the Entity Registry. This document details those enhancements. These changes were generally done to better support Campus Mobile and the Rules Engine, however given the common nature of the entity framework many of these changes are beneficial to all areas touched by the entity registry.

This document makes the assumption that you are already familiar with the Entity Registry as it is currently documented in the PeopleBooks and is designed merely to highlight new features.

Key Enhancements

- Entity Profiles – Allows the creation of subsets of data within an entity tree. Restricting access to child entities and properties for the rules engine, or trimming unnecessary data for web services.
- Common Attribute Framework Integration – Attributes can be accessed as properties on an entity. This allows attributes to be treated more like fields on the record.

UI Changes/Enhancements

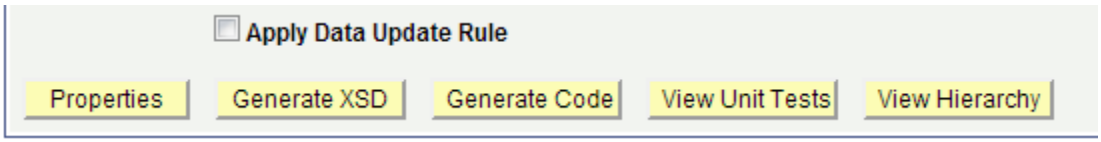
We have made several changes to the Entity Registry component, mostly to support new functionality.

Entity Registry

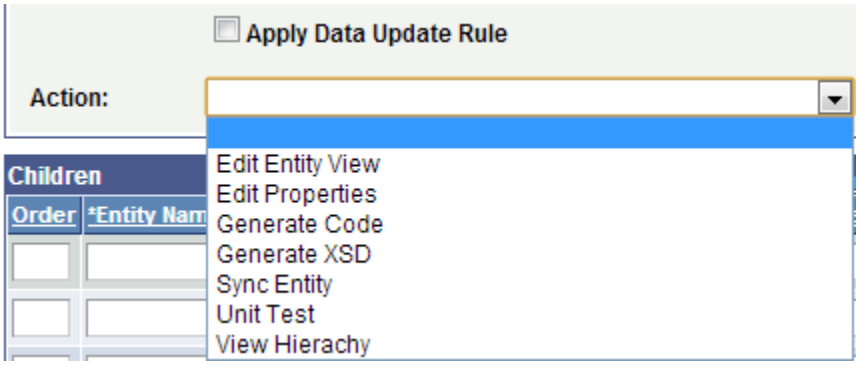
Action Drop Down

The old action button bar has been replaced with an action drop down. It supports the same functionality as the old buttons with a couple new inclusions.

- Entity View –Described in detail in the section on Entity Profiles, however entity views are accessed on the entity.
- Entity Sync – The Property sync process has been moved from the property page here. Since Common Attribute Framework properties can add children it needs to be called from the main page.



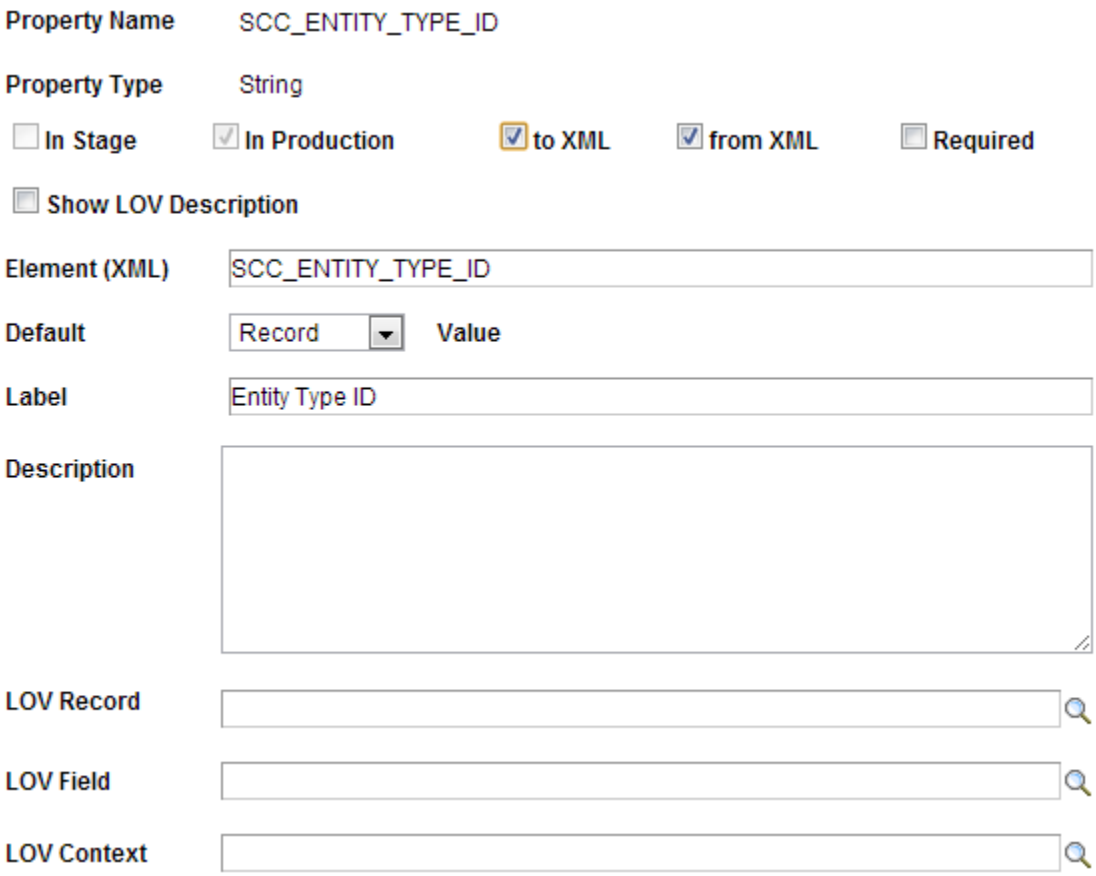
Old Buttons



New Drop Down

List of Values

We have added support for List of Value UID, allowing for a single key to reference a List of Values Setup rather than requiring record/field/context approach. For Properties already using the 3-key approach it will continue to work. All new LOV overrides require the UID approach, as it greatly simplifies setup. For fields that are using prompting based on the default underlying record/field prompting will continue to work as normal.



Old Property Details with 3-field LOV

Property Name SCC_ENTITY_TYPE_ID

Property Type String

☐ In Stage ☑ In Production ☐ Attribute ☐ List

☑ to XML ☑ from XML ☐ Required

☐ Show LOV Description

Element (XML) SCC_ENTITY_TYPE_ID

Default Record Value

Label Entity Type ID

Description

LOV Unique ID

New Property Details with LOV Unique ID

Properties Page

"to XML" and "from XML" have been replaced with "Viewable" and "Editable" respectively. With the addition of JSON and the rules engine, these attributes do not just apply to XML, but JSON and Rules Engine access as well.

Property Details — Personalize | Find | First 1-17 of 17 Last

Order	Property Name	Property Type	Field Name	In Production	to XML	from XML	Required	LOV	Details		
1	SCC_ENTITY_ID	String	SCC_ENTITY_ID	☑	☑	☑	☐	☐	Details	+	−

Old Grid

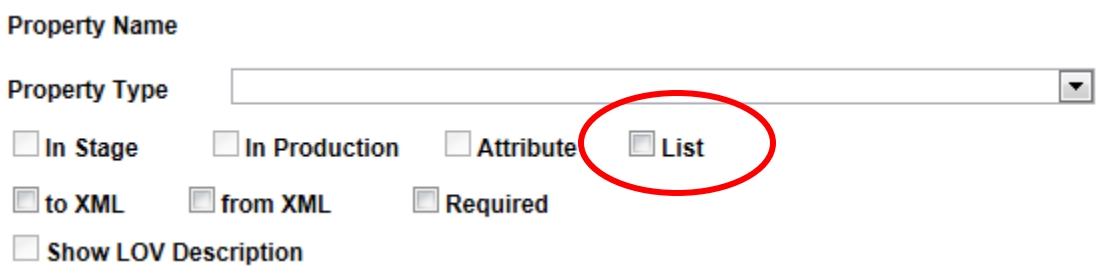
Property Details — Personalize | Find | First 1-17 of 17 Last

Order	Property Name	Property Type	Field Name	In Production	Attribute	Viewable	Editable	Required	LOV	Details		
1	SCC_ENTITY_ID	String	SCC_ENTITY_ID	☑	☐	☑	☑	☐	☐	Details	+	−

New Grid

List Properties

We now have support for treating properties as lists. This allows us to support the concept of repeatable attributes. Non-field based properties can also be marked as lists, in which case they are treated as arrays in the entity application class.



Entity Property Sync

There have been several Changes that required updating the Entity Sync Process.

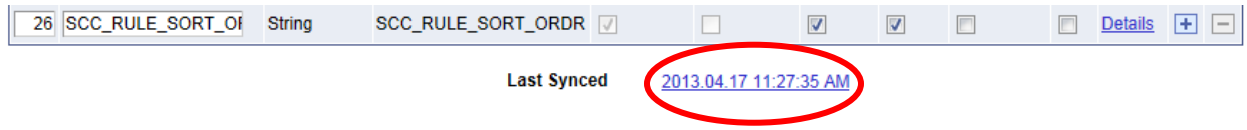
- The Number of entities has grown significantly.
- We now also sync entity views.
- We have increased the amount of caching.

UI Changes

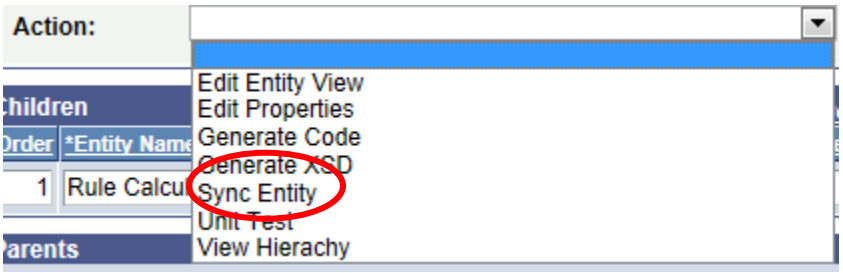
Entity Registry

Entity Registry setup component updates:

- Added persistent log view. You can see all sync based property changes from a link in the entity. property page.



- The reset button has been removed (it removed all properties and rebuilt from the record and could therefore unintentionally remove delivered properties).
- Sync has been moved to the action drop down, since it affects views and Common Attribute Framework integration could require it to add entity children.



Property Sync

Entity Property Sync

Press the "Sync All Entity Properties" to make sure all the properties on every entity get updated to properly reflect the record objects. The updates you manually have made to the properties will not be reset by this process.

Sync Log

Entity Name	Last Synced	Log
Disability	04/17/13 11:26:36AM	Log

Entity Sync Status Complete

Sync All Entity Properties

Entity Sync Datetime 04/17/13 11:28:47AM

Wipe Entity Cache

- Separated out Property Sync to 2 steps:
  - Sync Button now starts a Process Scheduler that does the sync operation. Once the request is in place it is replaced with a Refresh Button, to check the status, until the process is complete.
  - Wipe Cache allows manual wiping of the Entity Registry Meta Data cache. This was previously done as part of the sync process, but now that the sync is performed on a process scheduler, the cache wipe needs to be performed separately on the appserver.
- It is recommended to run a sync and wipe in these events:
  - Setting up a new database. This will make sure all your properties match the fields, and all your views match their underlying entity.
  - Importing data via dat/dms. Since dat/dms bypasses all processing logic this will fill in any holes.
  - If changes are made to records that have entities built on top of them via appdesigner. The sync process is the only way to update the entities to match the records. This can also be done on the entity registry component in a one-off manner.

Sync Process

The sync process keeps all the entity data up to date.

Here are the key features the sync process is responsible for:

- Updates properties to match underlying records.
- Keeps entity views up to date with changes to Entity Properties and Children.
- Keeps Entities properties/children synced with the Common Attribute Framework.

Cache Wipe

The Cache wipe wipes the entity registry cache on the appserver. This wipe should be performed anytime a sync is performed, once the sync process has completed successfully.

This cache uses rowset caches to store entity registry meta-data. This avoids unnecessary trips to the databases, bringing significant increases to performance for all entity registry operations.

Entity Profiles

Entity Profiles is a new feature that allows for entities to be grouped and subsets of entity trees to be created.

Main Features

- Categorize Entities for searching. The actual revised entity search is planned for a future bundle.
- Reduce data load for web services, and restrict access for updates.
- Restrict access for rules created using the rules engine.

Entity Views

Overview

Entity Views allow the creation of subsets of entity meta-data.

Entity Views can only reduce access from what is specified at the base entity. So if a property is set to viewable but not editable, it cannot be set to editable from a view, but viewable can be turned on or off.

So the options are:

Entity Properties

- Editable – Editable can be turned on or off on a per property/view basis.
- Viewable – Viewability can be set on a per property/view basis.
- LOV – List of Values can be turned on or off on a per view basis. However it requires that LOV is available and turned on for that property. The LOV setup for the property cannot otherwise be change on the view.

Entity Children

- Hide – Entity Children can be marked hidden, in which case those entities, that are a direct child of the current entity, will not be shown. This also means any entity that is a child of that entity will not be shown as well.
    - Note: Setting this value merely hides the entity child; it does not prevent that child from being filled as part of the initial fill process. This avoids certain complications in the rules engine.

Setup

Entity Name Entity Registry

Changes to properties and children will not be reflected in Entity Views until a save is performed.

Personalize | Find | | First 1 of 1 Last

Entity View Name	Default	Edit		
Complete	☑	Edit	+	−

Entity View Setup

Entity View ID	SCC_EVIEW_20130424102654
Entity Name	Entity Registry
***Entity View Name**	
Entity Description	

☐ **Default new items to disabled**

Entity Properties

Property Name	Viewable	Editable	LOV
SCC_ENTITY_ID	☑	☑	☐
SCC_STATUS	☑	☑	☐
SCC_ENTITY_NAME	☑	☑	☐
SCC_ENTITY_TYPE_ID	☑	☑	☐
SCC_ENTITY_DESCR	☑	☑	☐
SCC_APPCLASSNAME	☑	☑	☐
SCC_RECNAME	☑	☑	☐
SCC_STG_RECNAME	☑	☑	☐
SCC_ELEMENT_NAME	☑	☑	☐
SCC_APPLY_UPD_RULE	☑	☑	☐
SCC_ENTITY_INST_ID	☐	☑	☐
SCC_ENTITY_MSG	☑	☑	☐
SCC_ENT_LOCK	☑	☑	☐

Entity Children

	Entity Name	Hide
1	Entity Property	☐
2	Entity Child	☐
3	Entity Unit Test	☐

OK Cancel

Creating a new Entity View

Every entity has a “complete” view. The complete view always matches what has been set on the entity property and children. This view cannot be edited and is maintained during the sync process.

A single view, per entity, is always marked default. The default view is used if no profile is selected, or if that entity does not belong to the current profile. Default also specifies the view assigned when a new entity is added to a profile. The default is used by all entity based services built prior to profiles existing, such as EWS and AAWS.

Every view has the option to "Default new items to disabled". Normally any new property would be added to all views allowing access. Checking this box prevents any new properties/children from being editable/viewable/LOV without explicitly specifying it in the View definition.

Sync

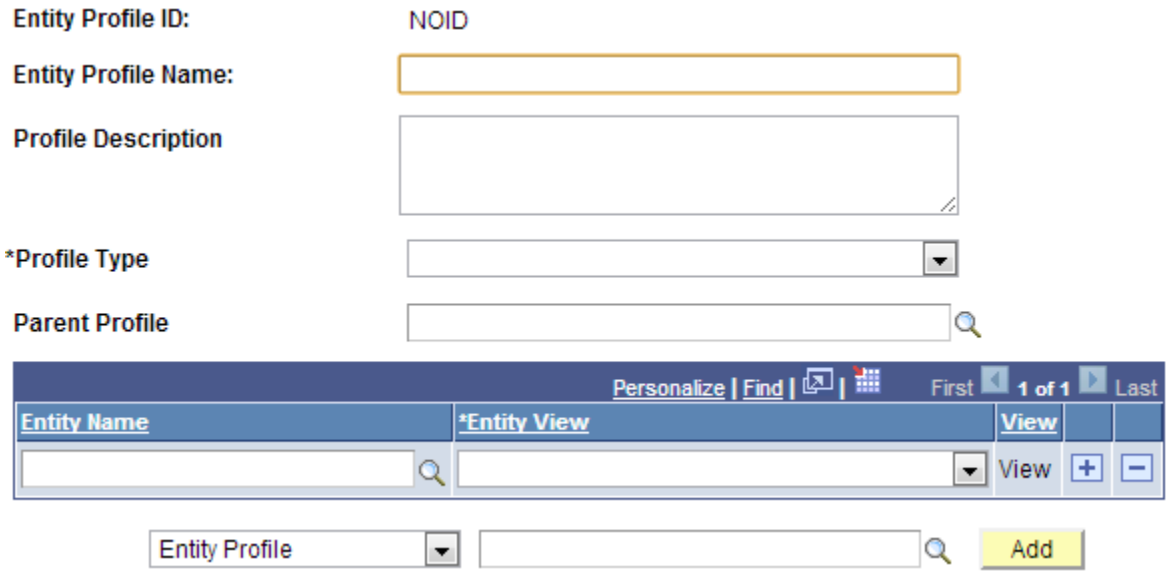
The sync process maintains the entity view matching the entity properties and children. No data is currently logged for entity view syncs.

Entity Profiles

Overview

Entity Profiles allow entities and views to be combined to create groupings of entities and views.

Setup



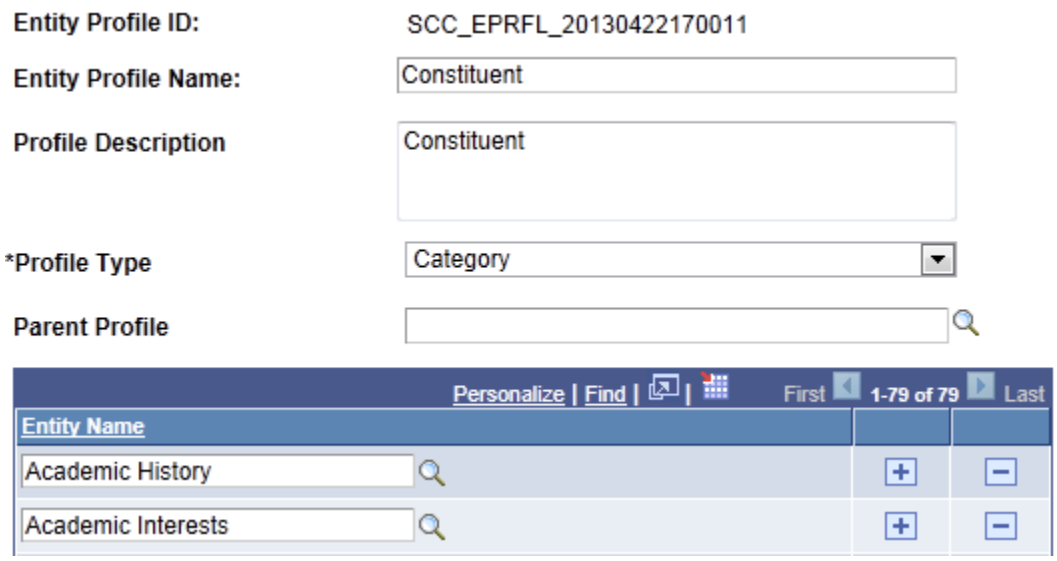
- Entity Profile ID – Auto Generated Unique ID for the profile.
- Entity Profile Name – The descriptive name of this profile.
- Entity Profile Description – The Description for this profile.
- Profile Type – One of 4 profile types, "Category", "Data Set Category", "Rules Engine", "Web Service". Each of those has a different purpose and may change available options. The details of each type are specified in the next section.
- Parent Profile – Each Profile may have a parent. A Child inherits entities, and their settings, from the parent profile.

    - Only one level of inheritance is allowed, so a profile can be a parent or a child, but not both. This was done to avoid the potential for confusion and maintenance difficulty.
    - Additional entities may be added on children, and entity settings may be overridden.
- Entity Name – The Name of the entity associated with this profile, from the list of entities defined in the registry.
- Entity View – The Entity View to use for this entity in the context of this profile.
- View – Shows details of access based on the entity view.
- Lower Drop Down – Allows selection of Entity Profile or Entity Tree, or adding all entities in a given tree, based on selecting the parent entity.
    - Entity Profile - This allows the addition of all entities in another profile. Since this is a one time copy it can be done as often as you want.
    - Entity Tree – Adds entities based on a parent entity being specified tracing down the entire tree and adding data. This is a one time add, so it does not do maintenance.

Types

Category

Category will be used to make searching through entities easier. Currently there are over 300 entities, making it more difficult than it used to be to find the actual entities that you want - being able to search for entities by category will help to alleviate that. Currently Category is not supported in search - that will be rectified soon.



For category there is no entity view, since it is not used to specify display a view is unnecessary.

Data Set Category

Entity Profile ID: SCC_EPRFL_20130306044322

Entity Profile Name: System Data Set

Profile Description System Data Set

***Profile Type** Data Set Catagory

Personalize | Find | First 1-2 of 2 Last

Entity Name		
AIR Courses	+	−
Student DataSet	+	−

Data set category is similar to category, except it is restricted to entities of type “rules engine data set”. Data Set Categories are created here but entities can be added in the data set setup component in the rules engine data set component.

Rules Engine

The Rules Engine profile allows restricting what properties and entity children are available based on the entity view. The available base entities for a rule category are also set here. Any Entity not specified as a base entity is not selectable as such when generating a rule in a rule category tied to this entity profile. Any entity not specified in here will still be accessible from the Rules Engine, using the default profile, if it is accessible from the entity tree.

Entity Profile ID: SCC_EPRFL_20130329103345

Entity Profile Name: System Profile AIR & APT

Profile Description System Profile AIR & APT

***Profile Type** Rules Engine

Parent Profile

Personalize | Find | First 1-10 of 10 Last

Entity Name	*Entity View	Base Entity	View		
AIR Course	Complete	☑	View	+	−
AIR Course List	Complete	☑	View	+	−

Web Service

Web Service entity profiles are designed for use with Web Services. They allow simple restriction of each entity to a particular view. Any entity not specified here can still be accessed using its default profile. If you want to remove an entity - remove it from the list of children on the parent view.

Common Attribute Framework Integration

Attributes from the common attribute framework are treated as entity properties. By adding the properties and syncing the entity the new attributes should show up as properties.

- Attribute types map to equivalent entity property types.

Attribute Type	Property Type
Short Text, Text, Long Text, Yes/No, LOV	String
Number	Number
Date	Date
Time	Time

- Repeatable maps to the list entity property list attribute.
- If the Attribute Type is a list of values the Property LOV UID is set.

- For each attribute record there is a child entity created. It is automatically added as a child. It is maintained and never should be accessed directly. This entity will never show up in XML/JSON and will not be accessible in the rules engine.

Under the Hood Changes/Enhancements

Entity Component Adapter

The entity component adapter allows entities to bind to the component row sets. By binding to the component, entity validation and presave logic can be run on live component data. This allows us to better consolidate common logic and make sure that components, services, APIS, etc. all follow the same rules for the data. The Entity Component Adapter also serves to allow rules to be run based on component data.

Performance Enhancements

There have been performance enhancements made across the entity registry - most notably in the area of Fill and LOV.

GenericParentEntity

Generic Parent Entity is an appclass that acts as a parent to any entity. It allows for an easy way to retrieve multiple entities of the same type. For example if you want to retrieve multiple constituents, you can use GenericParentEntity to easily get that.

Sample Code:

```
Local SCC_COMMON:ENTITY:GenericParentEntity &objGenericParentEntity;

/*Pass in the entityID you need a generic parent for in the constructor i.e. Constituent*/

    &objGenericParentEntity = create SCC_COMMON:ENTITY:GenericParentEntity(<entityID>);

/*this will fill based on a where clause on the underlying record, not as elegent but very flexible*/

    &objGenericParentEntity.fillRowsetFromWhere(&strWhereClause);

/*Another option is to set key values and use fill By Keys*/

&objGenericParentEntity.setProperty(<propname>,<value>);

&objGenericParentEntity.fillFromKeys(true);

/*Either way you can access the entity list via */

Array of Ientity &entitylist = &objGenericParentEntity.PrimaryChildEntity.childEntities;
```

/*The option also exists to call some of the usual entity functions, that will act on all the entities under the generic parent */

&objGenericParentEntity.toXMLNode(<XML Node>);

&objGenericParentEntity.save(<Message Log>);

JSON

Support for pushing data to JSON is in place just by running method toJSON() which returns a string, on any entity. There is some preliminary support for parsing JSON, but full support should be available in an upcoming bundle.

Multi-Field Sorting

Sorting entities has been enhanced to support sorting by multiple values. Sorting can be accessed on the childEntity using

```
method sortEntitiesByMultipleProps(&QSPropList As array of SCC_COMMON:UTIL:EntityQuickSortProp);
```

And EntityQuickSortProp

```
class EntityQuickSortProp

  /**The Name of the entity to sort by*/

  property string PropertyName;

  /**The sort order to use, A for Ascending, D for Descending*/

  property string SortOrder;

end-class;
```

The properties in the list take priority based on their order. Since this based on quicksort any sort will re-organize and invalidate any previous sorting.

Property List

Since properties can now be lists this can affect how those properties set as a list are handled in some situations.

- GetProperty – Returns an array. If no values have been set yet it returns an empty array of the proper data type.
- SetProperty – When adding/removing a value the entire list needs to be set. So if you need to add a new item to the property list, retrieve the list first using getproperty and set it using setProperty.

New Entity Types

There have been several new entity types added this release, most are product specific, dealing with Program Enrollment or Activity Management. These two are more generic so we are discussing them here.

Rules Engine Data Set

The Data Set Entity Type supports entities that are not record based. These entities are purely meant to act as data structures. This version of data sets is used by the rules engine, and controlled via a custom, simplified interface. For more information on Rules Engine Data Sets see the PeopleBooks chapter "Working with the Rules Engine".

Read Only

Read Only entities are essentially basic entities that have save and delete disabled. They make it easy to create entities on top of records for rules engine access. This makes it easy to add new entities accessible from the rules engine without introducing back door data updates. If at some point proper validation and/or presave logic is added the entity can be switched to Basic Entity and save and delete would be enabled.