Efecte

Efecte® Integration Suite

Efecte Query Language Description
Version 1.0
Confidential

TABLE OF CONTENTS

1 Introduction

The purpose of this document is to describe the Efecte Query Language (EQL) that is meant to be used for querying data from Efecte though the Web API.

2 Efecte Query Language Description

Efecte Query Language (EQL) is an SQL-like structured language for querying Efecte through the Web API. It enables performing powerful, flexible searches against the Efecte data model.
In order to use EQL effectively, one needs to know some basic concepts about the underlying data model. The elements of the data model most commonly used with EQL are "entity", "entitydata", "template" and "folder".
Below are simple examples to illustrate the use of EQL. The tables provide reference information about the most important classes and properties of the Efecte data model.

Table 1: Searchable classes

Class	Alias
com.efecte.datamodel.Entity	entity
com.efecte.datamodel.Folder	folder
com.efecte.datamodel.Template	template

Table 2: Properties of class entity (com.efecte.datamodel.Entity)

Property	Description
id	
template	reference to class template (com.efecte.datamodel.Template)
templateId	
folder	reference to class folder (com.efecte.datamodel.Folder)
name	
hidden	
deleted	
metaData	a group of objects of type com.efecte.datamode.Metadata
stringData	a group of objects of type com.bitmount.equipment.BSSStringEntityData
integerData	a group
floatData	a group
dateData	a group
referenceData	a group
* targetName	
* target	target data card

externalReferenceData	a group
* targetName	
* targetLocation	
textData	a group
staticStringData	a group
staticIntegerData	a group
staticFloatData	a group
staticDateData	a group

2.1 Properties of EntityData

The **EntityData** interface has several implementations, such as stringData, integerData, floatData, and so on. Attribute's data type determines which one is used. The most common property of any EntityData instance is "value".
EntityData instances of type referenceData and externalReferenceData have properties related to the reference target. These are "targetName" for both types, "target" for referenceData and "targetLocation" for externalReferenceData.
Static EntityData objects (staticStringData, staticIntegerData, staticFloatData and staticDateData) are special cases. Using their values in select and where clauses is a bit more complicated. For instance, consider the condition:

where entity.staticStringData.value.stringValue = 'foo'

The following table lists a few useful EntityData properties. Note that these are referred to as, for instance, "stringData.groupAttribute.code".

Table 3: Properties of class EntityData

Property
value
groupAttribute.id
groupAttribute.code
groupAttribute.attribute.name

2.2 Properties of class template

Table 4: Properties of class Template (com.efecte.datamodel.Template)

Property	Description
id	
name	

code	
systemCode	
protected	
referenceCascade	
primaryGroupAttributeID	
description	
templateGroups	
metadata	a group of objects of type com.efecte.datamode.Metadata
referringAttributes	a group of objects of type com.bitmount.equipment.BSSAttribute
permissions	a group of objects of type BSSTemplatePermission

2.3 Properties of class folder

Table 5: Properties of class Folder (com.efecte.datamodel.Folder)

Property	Description
id	
name	
code	
parentID	
parentType	
idCode	
newTemplatesAllowed	
metadata	a group of objects of type com.efecte.datamode.Metadata
allowedTemplates	a group of objects of type com.bitmount.equipment.BSSTemplate
roles	a group of objects of type com.bitmount.boas.AdminRole
permissions	a group of objects of type BSSEntityGroupPermission

2.4 EQL examples

2.4.1 Example: Search All Data Cards of the Whole Efecte System

Search all data cards, i.e. entities:

```
select entity from entity
```

2.4.2 Example: Search Specific Data from a Data Card

Search the name of the data card, name of the template and the name of the folder for entities that contain a string-attribute whose value is ‘falco’:

```
SELECT name, template.name, folder.name FROM entity WHERE stringData.value = 'falco' AND stringData.groupAttribute.id = 1234 AND template.id = 5678
```

2.4.3 Example 3)

Same as example 2, but the attribute is defined by using code:

```
SELECT name, template.name, folder.name FROM entity WHERE stringData.value = 'falco' AND stringData.groupAttribute.code = 'computer_name' AND template.id = 5678
```

2.4.4 Example 4)

Search all data cards of a single module:

```
SELECT entity FROM entity WHERE template.systemCode = 'beq'
```

or (demonstrates in sub search):

```
SELECT entity FROM entity WHERE templateId IN (SELECT id FROM template WHERE systemCode = 'beq')
```

or (demonstrates exists sub search):

```
SELECT entity FROM entity WHERE EXISTS (SELECT id FROM template WHERE id = entity.templateId AND template.systemCode = 'beq')
```