Retrieving Persistent State Metadata with the PersistentStateControl Adapter

The **Persistent State Control Adapter** gives access to persistent state metadata directly from within a service (self context). The same data can be retrieved using the **xUML Runtime API**.

If you want to retrieve metadata of persistent state of the very same service, always use the Persistent State Control Adapter.
If you want to retrieve data from other services, use the xUML Runtime API.

What a <<PersistentStateControl>> action does can be controlled via tagged value **action**. Currently the following actions are supported:

- listOwners
- getOwnerName
- listClasses
- getClassCounters
- getClassMetadata
- queryObjects
- deleteObject

Example File (Builder project Advanced Modeling/PState):

	<your example path>\Advanced Modeling\PState\uml\pstatePurchaseOrder.xml

On this Page:

Related Pages:

- xUML Runtime API
- Persistent State Ownership

Listing all Persistent State Owners

In Load Balancing context, when e.g. running multiple Bridges, you can setup persistent state services to share persistent state objects. The persistent state objects are distinguished by an owner and owner id reflecting the actual service that owns these objects.
Prerequisite is that these services share the same persistent state database, see Load Balanced Persistent State for more details.

listOwners lists all owners that are maintaining persistent state objects of the current service.
For more information on how to manage ownership of persistent state objects, refer to Persistent State Ownership.

Parameters

Name	Type	Direction	Mandatory	Description
owners	Array of **Owner**	out	✓	The adapter returns an array of owner details.

Getting the Name of an Owner

getOwnerName returns the name of the current owner (self).
For more information on how to manage ownership of persistent state objects, refer to Persistent State Ownership.

Parameters

Name	Type	Direction	Mandatory	Description
ownerName	**String**	out	✓	The adapter returns the name of the current owner as a **String**.

Listing all Available Persistent State Classes

listClasses returns an array list of all available persistent state classes of the current service (self). By specifying **includeObjectCount** = true, you can get the actual object count per class.

Parameters

Name	Type	Direction	Mandatory	Description	Allowed Values	
includeObjectCount	**Boolean**	in		Specify whether to include an object count per class to the response.	true	Include an object count per class.
					false	Do not include an object count per class (default).
classes	Array of **ClassInfo**	out	✓	The adapter returns a list of classes as an Array of **ClassInfo**.		

Getting Object Counters per Class

getClassCounters returns an array list of all available persistent state classes of the current service (self) and their actual counters. Refer to type ClassCounters for more details on which counters are available.

Parameters

Name	Type	Direction	Mandatory	Description
counters	Array of **ClassCounters**	out	✓	The adapter returns the object counters per class as an **Array**. Refer to type ClassCounters for more details on the counters.

Getting the Persistent State Class Metadata

getClassMetadata returns the metadata of a given class. The action returns an array list of all attributes and their types.

Parameters

Name	Type	Direction	Mandatory	Description
class	**String**	in	✓	Specify the name of the class to get metadata of. You can provide the value dynamically or via tagged value **class** on the adapter action.
classMetadata	**ClassMetadata**	out	✓	The adapter returns a **ClassMetadata** object, listing all attributes and their metadata, and the primary and search keys. The array of primary keys is sorted in definition order.

Querying Persistent State Objects of a Given Class

With action **queryObjects** you can query the persistent state database for objects of a given class. Queries can use simple query conditions and complex query conditions (and/or).

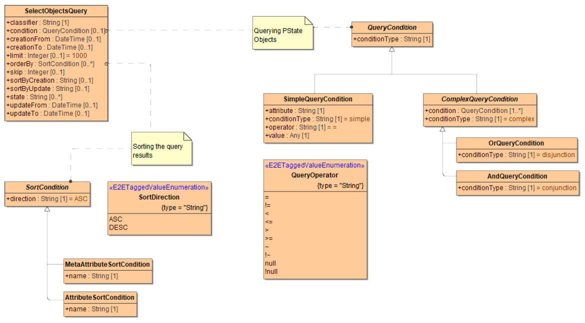
Using **queryObjects**, the persistent state database can be search by the **search keys** that are defined on the persistent state class.

Queries are steered by parameter **selectQuery** that, on the one hand, specifies global search data like searching by object dates and search order, and, on the other hand, can hold complex search queries.

Parameters

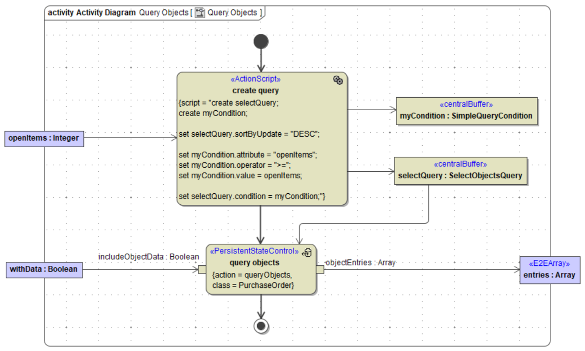
Name	Type	Direction	Mandatory	Description	Allowed Values	
selectQuery	**SelectObjectsQuery**	in	✓	Provide a search query.	.	
includeObjectData	**Boolean**	in		Specify whether to include object data of the matching objects to the response.	true	Return object data of the found objects.
					false	Only return the objects metadata (default).
objectEntries	Array of **ObjectEntry**	out	✓	The adapter returns an array of objects and some basic object metadata per object.		

Attribute **condition** of type **SelectObjectsQuery** holds the custom search query itself. Attribute **orderBy** holds sorting specifications.


SelectObjectsQuery
+classifier : String [1]
+condition : QueryCondition [0..1]
+creationFrom : DateTime [0..1]
+creationTo : DateTime [0..1]
+limit : Integer [0..1] = 1000
+orderBy : SortCondition [0..*]
+skip : Integer [0..1]
+sortByCreation : String [0..1]
+sortByUpdate : String [0..1]
+state : String [0..*]
+updateFrom : DateTime [0..1]
+updateTo : DateTime [0..1]
Querying PState Objects
QueryCondition
+conditionType : String [1]
SimpleQueryCondition
+attribute : String [1]
+conditionType : String [1] = simple
+operator : String [1] = =
+value : Any [1]
ComplexQueryCondition
+condition : QueryCondition [1..*]
+conditionType : String [1] = complex
OrQueryCondition
+conditionType : String [1] = disjunction
AndQueryCondition
+conditionType : String [1] = conjunction
«E2ETaggedValueEnumeration»
QueryOperator
{type = "String"}
=
!=
<
<=
>
>=
~
!~
null
!null
Sorting the query results
SortCondition
+direction : String [1] = ASC
«E2ETaggedValueEnumeration»
SortDirection
{type = "String"}
ASC
DESC
MetaAttributeSortCondition
+name : String [1]
AttributeSortCondition
+name : String [1]

Building a Simple Query

By using only one condition of type **SimpleQueryCondition** you can build a simple query. Find below a the activity diagram of a simple query that returns all purchase orders with 2 or more open items.



```
{script = "create selectQuery;
create myCondition;

set selectQuery.sortByUpdate = "DESC";

set myCondition.attribute = "openItems";
set myCondition.operator = ">=";
set myCondition.value = openItems;

set selectQuery.condition = myCondition;"}
```

```
{action = queryObjects,
class = PurchaseOrder}
```

selectQuery.conditions holds the query condition. It consists of

- name of the search key attribute to use for comparison
- an operator for the comparison
- a value to compare against

Valid operators are:

Operator	Description
=	Equal.
!=	Not equal.
<	Less than.
<=	Less or equal.
>	Greater than.

>=	Greater or equal.
~	Like (SQL).
!~	Not like (SQL).
null	Null.
!null	Not null.

All operators will be translated to SQL operators, so the relational operators will not work as expected if any of the operands is NULL. The Runtime will throw error `PSADSM/46` in this case.

Exception: The `=` and `!=` operators will map to `IS NULL` and `IS NOT NULL` in this case.

Querying does only work on persistent state attributes that have been marked as <<SearchKey>>. Also, if you apply this stereotype to a persistent state attribute later on, all previous persistent state objects are disregarded if searching with this key.

Building a Complex Query

Using type **ComplexQueryCondition**, you can build a complex query of multiple simple queries. They can be joined together by a **disjunction** (or) or a **conjunction** (and).

Assuming you have `myCondition1` and `myCondition2` of type **SimpleQueryCondition**, you can join them to an **and** query with an `andQuery` of type **AndQueryCondition** like:

```
create selectQuery;
create myCondition1;
create myCondition2;
create andQuery;

set selectQuery.sortByUpdate = "DESC";

set myCondition1.attribute = "openItems";
set myCondition1.operator = ">=";
set myCondition1.value = openItems;

set myCondition2.attribute = "customerID";
set myCondition2.operator = "=";
set myCondition2.value = "4711";

append myCondition1 to andQuery.condition;
append myCondition2 to andQuery.condition;

set selectQuery.condition = andQuery;
```

In this case, one of the to conditions could also be a complex condition instead of a simple one. Like that you can build very complex combinations of **and** and **or** queries.

Sorting the Query Results

You can sort the results that are returned by persistent state attributes and/or by persistent state meta data (creation timestamp and update timestamp).

Sorting	Attribute(s) of SelectObjectsQuery	Description
Only by persistent state meta data	• **sortByCreation** • **sortByUpdate**	Provide a sorting direction with the **sortByCreation** and **sortByUpdate** attributes of **SelectObjectsQuery** (ASC, DESC).

By persistent state attributes	• **orderBy**	Create an array of objects of type **AttributeSortCondition** and provide it with the **orderBy** attribute of **SelectObjects Query**. **sortByCreation** and **sortByUpdate** will be completely ignored in this case.
By a mixture of persistent state attributes and meta data	• **orderBy**	The array **orderBy** can hold objects of **AttributeSortCondition** and **MetaAttributeSortCondition** as they both derive from **SortCondition** (see class diagram above). Provide the meta attribute to sort by with an object of type **MetaAttributeSortCondition**. **sortByCreation** and **sortByUpdate** will be completely ignored in this case.

Deleting Persistent State Objects

deleteObject deletes the object identified by **class** and **objectId**.

Deleting objects directly via **deleteObject** is not best practice and can lead to odd side effects. Best practice is to model this in the state machine.

Parameters

Name	Type	Direction	Mandatory	Description	Example
class	**String**	in	✓	Specify the name of the class to delete objects from.	`PurchaseOrder`
objectId	**String**	in	✓	Specify the id of the object to delete.	`000100058a79cb967f6e00000079`

Parameter Types

Class	Attribute	Type	Description
AndQueryCondition	conditionType	**String**	Type of the cor
	condition	Array of **QueryCondition**	List of simple q
AttributeSortConditon	name	**String**	Name of the pe
	direction	**String**	Sort direction a
ClassAttributeMetadata	name	**String**	Name of the pe
	type	**String**	Type of the per • `String (` • UML clas `Item`
ClassCounters	name	**String**	Name of the pe
	count	**Integer**	Object count.
	stalledCount	**Integer**	Count of object
	states	Array of **StateCounters**	List of states in
ClassInfo	name	**String**	Name of the pe

	count	**Integer**	Object count.
ClassMetadata	name	**String**	Name of the pe
	attributes	Array of **ClassAttributeMetadata**	List of class att
	primaryKeys	Array of **String**	List of attribute The array of
	searchKeys	Array of **String**	List of attribute
MetaAttributeSortCondition	name	**String**	Name of the pe
	direction	**String**	Sort direction a
ObjectEntry	id	**String**	Unique identifie
	name	**String**	Name of the pe
	creation	**DateTime**	Creation date c
	lastUpdate	**DateTime**	Date object ha
	states	Array of **String**	List of states th
	object	**Any**	Copy of the pe **object** contain Persistent Stat
OrQueryCondition	conditionType	**String**	Type of the cor
	condition	Array of **QueryCondition**	List of simple q
Owner	id	**String**	Owner id. For more inforr sistent State O
	compositeName	**String**	Composite nan
	host	**String**	Name of the hc
	lastStartup	**DateTime**	Last recorded s
	lastShutdown	**DateTime**	Last recorded s
	ownedObjects	**Integer**	Count of owne
	isSelf	**Boolean**	True, if the cur
QueryCondition	Parent abstract class of **SimpleQueryCondition**, **AndQueryCondition**, or **OrQueryCondition**.		
SimpleQueryCondition	conditionType	**String**	Type of the cor
	attribute	**String**	Name of the se
	operator	**String**	Operator for th All operators work as exp `SM/46` in thi **Exception:** case.
	value	**Any**	Value to comp
SelectObjectsQuery	classifier	**String**	Name of the pe

	creationFrom	**DateTime**	Creation date f
	creationTo	**DateTime**	Creation date t
	sortByCreation	**String**	Sort by creatio • If **sortBy** unspecifie • If both, **sc** creation s • If you pro
	updateFrom	**DateTime**	Update date fro
	updateTo	**DateTime**	Update date to
	sortByUpdate	**String**	Sort by update • If **sortBy** unspecifie • If both, **sc** creation s • If you pro
	limit	**Integer**	Limit the count
	state	Array of **String**	List of states. A persistent sta (disjunction).
	condition	**QueryCondition**	Query conditio Given Class).
	skip	**Integer**	Skip the numbe implement pag When least t • •
	orderBy	Array of **SortCondition**	Provide sort co translated into If **ord** comp and **u**
SortCondition	direction	**String**	Sort direction.
StateCounters	name	**String**	Name of the st
	count	**Integer**	Count of object
	stalledCount	**Integer**	Count of stalle