

# 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

## 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 <a href="#">Owner</a>	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	<b>String</b>	out	✓	The adapter returns the name of the current owner as a <b>String</b> .

## Listing all Available Persistent State Classes

### On this Page:


- [Listing all Persistent State Owners](#)
  - [Parameters](#)
- [Getting the Name of an Owner](#)
  - [Parameters](#)
- [Listing all Available Persistent State Classes](#)
  - [Parameters](#)
- [Getting Object Counters per Class](#)
  - [Parameters](#)
- [Getting the Persistent State Class Metadata](#)
  - [Parameters](#)
- [Querying Persistent State Objects of a Given Class](#)
  - [Parameters](#)
  - [Building a Simple Query](#)
  - [Building a Complex Query](#)
  - [Sorting the Query Results](#)
- [Deleting Persistent State Objects](#)
  - [Parameters](#)
- [Parameter Types](#)

### Related Pages:

- [xUML Runtime API](#)
- [Persistent State Ownership](#)

**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 <a href="#">ClassInfo</a>	out		The adapter returns a list of classes as an Array of <b>ClassInfo</b> .		

## 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 <a href="#">ClassCounters</a>	out	✓	The adapter returns the object counters per class as an <b>Array</b> . Refer to type <a href="#">ClassCounters</a> 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 <b>class</b> on the adapter action.
classMetadata	<a href="#">ClassMetadata</a>	out	✓	The adapter returns a <b>ClassMetadata</b> object, listing all attributes and their metadata, and the primary and search keys. <div>The array of primary keys is sorted in definition order.</div>

## 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).

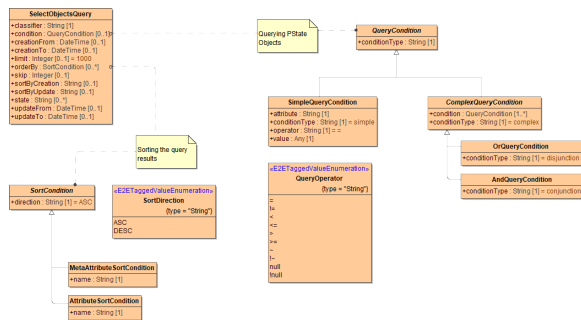
Using **queryObjects**, the persistent state database can be search by the **search keys** that are defined on the persistent state class.

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

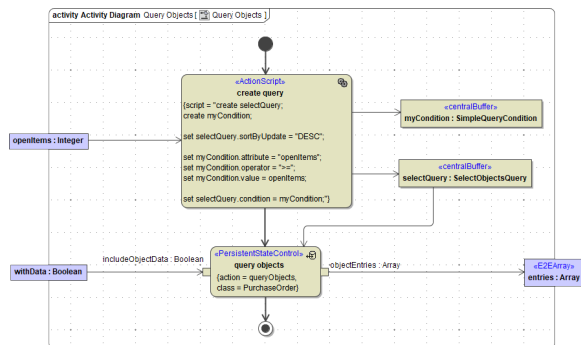
Name	Type	Direction	Mandatory	Description	Allowed Values
selectQuery	<a href="#">SelectObjectsQuery</a>	in	✓	Provide a search query.	.
includeObjectData	Boolean	in		Specify whether to include object data of the matching objects to the response.	<div>true</div> <div>Return object data of the found objects.</div> <div>false</div> <div>Only return the objects metadata (default).</div>
objectEntries	Array of <a href="#">ObjectEntry</a>	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.



## 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.



**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.

<b>&gt;=</b>	Greater or equal.
<b>~</b>	Like (SQL).
<b>!~</b>	Not like (SQL).
<b>null</b>	Null.
<b>!null</b>	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 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 <a href="#">SelectObjectsQuery</a>	Description
<b>Only by persistent state meta data</b>	<ul style="list-style-type: none"> <li><b>sortByCreation</b></li> <li><b>sortByUpdate</b></li> </ul>	Provide a sorting direction with the <b>sortByCreation</b> and <b>sortByUpdate</b> attributes of <a href="#">SelectObjectsQuery</a> (ASC, DESC).

By persistent state attributes	<ul style="list-style-type: none"> <li><b>orderBy</b></li> </ul>	<p>Create an array of objects of type <a href="#">AttributeSortCondition</a> and provide it with the <b>orderBy</b> attribute of <a href="#">SelectObjectsQuery</a>.</p> <div>  <b>sortByCreation</b> and <b>sortByUpdate</b> will be completely ignored in this case.         </div>
By a mixture of persistent state attributes and meta data	<ul style="list-style-type: none"> <li><b>orderBy</b></li> </ul>	<p>The array <b>orderBy</b> can hold objects of <a href="#">AttributeSortCondition</a> and <a href="#">MetaAttributeSortCondition</a> as they both derive from <b>SortCondition</b> (see <a href="#">class diagram</a> above). Provide the meta attribute to sort by with an object of type <b>MetaAttributeSortCondition</b>.</p> <div>  <b>sortByCreation</b> and <b>sortByUpdate</b> will be completely ignored in this case.         </div>

## 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	<b>String</b>	in	✓	Specify the name of the class to delete objects from.	PurchaseOrder
objectId	<b>String</b>	in	✓	Specify the id of the object to delete.	000100058a79cb967f6e00000079

## Parameter Types

Class	Attribute	Type	Description
<b>AndQueryCondition</b>	conditionType	<b>String</b>	Type of the cor
	condition	Array of <a href="#">QueryCondition</a>	List of simple q
<b>AttributeSortConditon</b>	name	<b>String</b>	Name of the pe
	direction	<b>String</b>	Sort direction a
<b>ClassAttributeMetadata</b>	name	<b>String</b>	Name of the pe
	type	<b>String</b>	Type of the per <ul style="list-style-type: none"> <li><b>String</b> (</li> <li>UML clas</li> <li>Item</li> </ul>
<b>ClassCounters</b>	name	<b>String</b>	Name of the pe
	count	<b>Integer</b>	Object count.
	stalledCount	<b>Integer</b>	Count of object
	states	Array of <a href="#">StateCounters</a>	List of states in
<b>ClassInfo</b>	name	<b>String</b>	Name of the pe

	count	Integer	Object count.
ClassMetadata	name	String	Name of the pe
	attributes	Array of <a href="#">ClassAttributeMetadata</a>	List of class att
	primaryKeys	Array of <b>String</b>	List of attribute <div> The array of </div>
	searchKeys	Array of <b>String</b>	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 <b>String</b>	List of states th
	object	Any	Copy of the pe <b>object</b> contains Persistent Stat
OrQueryCondition	conditionType	String	Type of the cor
	condition	Array of <a href="#">QueryCondition</a>	List of simple q
Owner	id	String	Owner id. For more inforr <a href="#">sistent State O</a>
	compositeName	String	Composite nan
	host	String	Name of the hc
	lastStartup	DateTime	Last recorded s
	lastShutdown	DateTime	Last recorded s
	ownedObjects	Integer	Count of owner
	isSelf	Boolean	True, if the curi
QueryCondition	Parent abstract class of <a href="#">SimpleQueryCondition</a> , <a href="#">AndQueryCondition</a> , or <a href="#">OrQueryCondition</a> .		
SimpleQueryCondition	conditionType	String	Type of the cor
	attribute	String	Name of the se
	operator	String	Operator for th <div> All operators work as exp SM/ 46 in thi  <b>Exception:</b> case. </div>
	value	Any	Value to compa
SelectObjectsQuery	classifier	String	Name of the pe

	creationFrom	DateTime	Creation date f
	creationTo	DateTime	Creation date t
	sortByCreation	String	Sort by creation <ul style="list-style-type: none"><li>• If <b>sortBy</b> unspecified</li><li>• If both, <b>sortByCreation</b></li><li>• If you pro</li></ul>
	updateFrom	DateTime	Update date fr
	updateTo	DateTime	Update date to
	sortByUpdate	String	Sort by update <ul style="list-style-type: none"><li>• If <b>sortBy</b> unspecified</li><li>• If both, <b>sortByCreation</b></li><li>• If you pro</li></ul>
	limit	Integer	Limit the count
	state	Array of String	List of states.  A persistent state (disjunction).
	condition	QueryCondition	Query condition (Given Class).
	skip	Integer	Skip the number of items to implement pagination. <div> When least 1 item is skipped, the first item is the first item in the list. • •</div>
SortCondition	direction	String	Sort direction.
			<div> If orderBy is specified and update is not, the sort order is ascending.</div>
StateCounters	name	String	Name of the state counter.
	count	Integer	Count of objects in the state.
	stalledCount	Integer	Count of stalled objects.