Freeze-Evictor

Freeze::Evictor

Overview

local interface Evictor extends Ice::ServantLocator

An automatic Ice object persistence manager, based on the evictor pattern. The evictor is a servant locator implementation that stores the persistent state of its objects in a database. Any number of objects can be registered with an evictor, but only a configurable number of servants are active at a time. These active servants reside in a queue; the least recently used servant in the queue is the first to be evicted when a new servant is activated.

Derived Classes and Interfaces

- Freeze::BackgroundSaveEvictor
- Freeze::TransactionalEvictor

See Also

- Freeze::ServantInitializer

Operation Index

setSize — Set the size of the evictor's servant queue.
getSize — Get the size of the evictor's servant queue.
add — Add a servant to this evictor.
addFacet — Like add, but with a facet.
remove — Permanently destroy an Ice object.
removeFacet — Like remove, but with a facet.
hasObject — Returns true if the given identity is managed by the evictor with the default facet.
hasFacet — Like hasObject, but with a facet.
getIterator — Get an iterator for the identities managed by the evictor.

Operations

void setSize(int sz)

Set the size of the evictor's servant queue. This is the maximum number of servants the evictor keeps active. Requests to set the queue size to a value smaller than zero are ignored.

Parameters

`sz` — The size of the servant queue. If the evictor currently holds more than `sz` servants in its queue, it evicts enough servants to match the new size. Note that this operation can block if the new queue size is smaller than the current number of servants that are servicing requests. In this case, the operation waits until a sufficient number of servants complete their requests.

Exceptions

Freeze::EvictorDeactivatedException — Raised if a the evictor has been deactivated.

See Also

- getSize

int getSize()

Get the size of the evictor's servant queue.

Return Value

The size of the servant queue.

Exceptions

Freeze::EvictorDeactivatedException — Raised if a the evictor has been deactivated.

See Also

- setSize

Object* add(Object servant, Ice::Identity id)

Add a servant to this evictor. The state of the servant passed to this operation will be saved in the evictor's persistent store.

Parameters

`servant` — The servant to add.
`id` — The identity of the Ice object that is implemented by the servant.

Return Value

A proxy that matches the given identity and this evictor's object adapter.

Exceptions

Ice::AlreadyRegisteredException — Raised if the evictor already has an object with this identity.
Freeze::DatabaseException — Raised if a database failure occurred.
Freeze::EvictorDeactivatedException — Raised if the evictor has been deactivated.

See Also

- addFacet
- remove
- removeFacet

Object* addFacet(Object servant, Ice::Identity id, string facet)

Like add, but with a facet. Calling `add(servant, id)` is equivalent to calling addFacet with an empty facet.

Parameters

`servant` — The servant to add.
`id` — The identity of the Ice object that is implemented by the servant.
`facet` — The facet. An empty facet means the default facet.

Return Value

A proxy that matches the given identity and this evictor's object adapter.

Exceptions

Ice::AlreadyRegisteredException — Raised if the evictor already has an object with this identity.
Freeze::DatabaseException — Raised if a database failure occurred.
Freeze::EvictorDeactivatedException — Raised if the evictor has been deactivated.

See Also

- add
- remove
- removeFacet

Object remove(Ice::Identity id)

Permanently destroy an Ice object.

Parameters

`id` — The identity of the Ice object.

Return Value

The removed servant.

Exceptions

Ice::NotRegisteredException — Raised if this identity was not registered with the evictor.
Freeze::DatabaseException — Raised if a database failure occurred.
Freeze::EvictorDeactivatedException — Raised if the evictor has been deactivated.

See Also

- add
- removeFacet

Object removeFacet(Ice::Identity id, string facet)

Like remove, but with a facet. Calling `remove(id)` is equivalent to calling removeFacet with an empty facet.

Parameters

`id` — The identity of the Ice object.
`facet` — The facet. An empty facet means the default facet.

Return Value

The removed servant.

Exceptions

Ice::NotRegisteredException — Raised if this identity was not registered with the evictor.
Freeze::DatabaseException — Raised if a database failure occurred.
Freeze::EvictorDeactivatedException — Raised if the evictor has been deactivated.

See Also

- remove
- addFacet

bool hasObject(Ice::Identity id)

Returns true if the given identity is managed by the evictor with the default facet.

Return Value

true if the identity is managed by the evictor, false otherwise.

Exceptions

Freeze::DatabaseException — Raised if a database failure occurred.
Freeze::EvictorDeactivatedException — Raised if a the evictor has been deactivated.

bool hasFacet(Ice::Identity id, string facet)

Like hasObject, but with a facet. Calling `hasObject(id)` is equivalent to calling hasFacet with an empty facet.

Return Value

true if the identity is managed by the evictor for the given facet, false otherwise.

Exceptions

Freeze::DatabaseException — Raised if a database failure occurred.
Freeze::EvictorDeactivatedException — Raised if a the evictor has been deactivated.

Freeze::EvictorIterator getIterator(string facet, int batchSize)

Get an iterator for the identities managed by the evictor.

Parameters

`facet` — The facet. An empty facet means the default facet.
`batchSize` — Internally, the Iterator retrieves the identities in batches of size batchSize. Selecting a small batchSize can have an adverse effect on performance.

Return Value

A new iterator.

Exceptions

Freeze::EvictorDeactivatedException — Raised if a the evictor has been deactivated.