IceGrid-AdminSession

IceGrid::AdminSession

Overview

interface AdminSession extends Glacier2::Session

Used by administrative clients to view, update, and receive observer updates from the IceGrid registry. Admin sessions are created either with the IceGrid::Registry object or the registry admin Glacier2::SessionManager object.

See Also

- IceGrid::Registry
- Glacier2::SessionManager

Operation Index

keepAlive — Keep the session alive.
getAdmin — Get the admin interface.
getAdminCallbackTemplate — Get a "template" proxy for admin callback objects.
setObservers — Set the observer proxies that receive notifications when the state of the registry or nodes changes.
setObserversByIdentity — Set the observer identities that receive notifications the state of the registry or nodes changes.
startUpdate — Acquires an exclusive lock to start updating the registry applications.
finishUpdate — Finish updating the registry and release the exclusive lock.
getReplicaName — Get the name of the registry replica hosting this session.
openServerLog — Open the given server log file for reading.
openServerStdErr — Open the given server stderr file for reading.
openServerStdOut — Open the given server stdout file for reading.
openNodeStdErr — Open the given node stderr file for reading.
openNodeStdOut — Open the given node stdout file for reading.
openRegistryStdErr — Open the given registry stderr file for reading.
openRegistryStdOut — Open the given registry stdout file for reading.

Operations

void keepAlive()

Keep the session alive. Clients should call this operation regularly to prevent the server from reaping the session.

See Also

- IceGrid::Registry::getSessionTimeout

["nonmutating"] IceGrid::Admin* getAdmin()

Get the admin interface. The admin object returned by this operation can only be accessed by the session.

Return Value

The admin interface proxy.

Object* getAdminCallbackTemplate()

Get a "template" proxy for admin callback objects. An Admin client uses this proxy to set the category of its callback objects, and the published endpoints of the object adapter hosting the admin callback objects.

Return Value

A template proxy. The returned proxy is null when the Admin session was established using Glacier2.

void setObservers(IceGrid::RegistryObserver* registryObs, IceGrid::NodeObserver* nodeObs, IceGrid::ApplicationObserver* appObs, IceGrid::AdapterObserver* adptObs, IceGrid::ObjectObserver* objObs) throws IceGrid::ObserverAlreadyRegisteredException

Set the observer proxies that receive notifications when the state of the registry or nodes changes.

Parameters

`registryObs` — The registry observer.
`nodeObs` — The node observer.
`appObs` — The application observer.
`adptObs` — The adapter observer.
`objObs` — The object observer.

Exceptions

IceGrid::ObserverAlreadyRegisteredException — Raised if an observer is already registered with this registry.

void setObserversByIdentity(Ice::Identity registryObs, Ice::Identity nodeObs, Ice::Identity appObs, Ice::Identity adptObs, Ice::Identity objObs) throws IceGrid::ObserverAlreadyRegisteredException

Set the observer identities that receive notifications the state of the registry or nodes changes. This operation should be used by clients that are using a bidirectional connection to communicate with the session.

Parameters

`registryObs` — The registry observer identity.
`nodeObs` — The node observer identity.
`appObs` — The application observer.
`adptObs` — The adapter observer.
`objObs` — The object observer.

Exceptions

IceGrid::ObserverAlreadyRegisteredException — Raised if an observer is already registered with this registry.

int startUpdate() throws IceGrid::AccessDeniedException

Acquires an exclusive lock to start updating the registry applications.

Return Value

The current serial.

Exceptions

IceGrid::AccessDeniedException — Raised if the exclusive lock can't be acquired. This might happen if the lock is currently acquired by another session.

void finishUpdate() throws IceGrid::AccessDeniedException

Finish updating the registry and release the exclusive lock.

Exceptions

IceGrid::AccessDeniedException — Raised if the session doesn't hold the exclusive lock.

string getReplicaName()

Get the name of the registry replica hosting this session.

Return Value

The replica name of the registry.

IceGrid::FileIterator* openServerLog(string id, string path, int count) throws IceGrid::FileNotAvailableException, IceGrid::ServerNotExistException, IceGrid::NodeUnreachableException, IceGrid::DeploymentException

Open the given server log file for reading. The file can be read with the returned file iterator.

Parameters

`id` — The server id.
`path` — The path of the log file. A log file can be opened only if it's declared in the server or service deployment descriptor.
`count` — Specifies where to start reading the file. If negative, the file is read from the begining. If 0 or positive, the file is read from the last `count` lines.

Return Value

An iterator to read the file.

Exceptions

IceGrid::FileNotAvailableException — Raised if the file can't be read.
IceGrid::ServerNotExistException — Raised if the server doesn't exist.
IceGrid::NodeUnreachableException — Raised if the node could not be reached.
IceGrid::DeploymentException — Raised if the server couldn't be deployed on the node.

IceGrid::FileIterator* openServerStdErr(string id, int count) throws IceGrid::FileNotAvailableException, IceGrid::ServerNotExistException, IceGrid::NodeUnreachableException, IceGrid::DeploymentException

Open the given server stderr file for reading. The file can be read with the returned file iterator.

Parameters

`id` — The server id.
`count` — Specifies where to start reading the file. If negative, the file is read from the begining. If 0 or positive, the file is read from the last `count` lines.

Return Value

An iterator to read the file.

Exceptions

IceGrid::FileNotAvailableException — Raised if the file can't be read.
IceGrid::ServerNotExistException — Raised if the server doesn't exist.
IceGrid::NodeUnreachableException — Raised if the node could not be reached.
IceGrid::DeploymentException — Raised if the server couldn't be deployed on the node.

IceGrid::FileIterator* openServerStdOut(string id, int count) throws IceGrid::FileNotAvailableException, IceGrid::ServerNotExistException, IceGrid::NodeUnreachableException, IceGrid::DeploymentException

Open the given server stdout file for reading. The file can be read with the returned file iterator.

Parameters

`id` — The server id.
`count` — Specifies where to start reading the file. If negative, the file is read from the begining. If 0 or positive, the file is read from the last `count` lines.

Return Value

An iterator to read the file.

Exceptions

IceGrid::FileNotAvailableException — Raised if the file can't be read.
IceGrid::ServerNotExistException — Raised if the server doesn't exist.
IceGrid::NodeUnreachableException — Raised if the node could not be reached.
IceGrid::DeploymentException — Raised if the server couldn't be deployed on the node.

IceGrid::FileIterator* openNodeStdErr(string name, int count) throws IceGrid::FileNotAvailableException, IceGrid::NodeNotExistException, IceGrid::NodeUnreachableException

Open the given node stderr file for reading. The file can be read with the returned file iterator.

Parameters

`name` — The node name.
`count` — Specifies where to start reading the file. If negative, the file is read from the begining. If 0 or positive, the file is read from the last `count` lines.

Return Value

An iterator to read the file.

Exceptions

IceGrid::FileNotAvailableException — Raised if the file can't be read.
IceGrid::NodeNotExistException — Raised if the node doesn't exist.
IceGrid::NodeUnreachableException — Raised if the node could not be reached.

IceGrid::FileIterator* openNodeStdOut(string name, int count) throws IceGrid::FileNotAvailableException, IceGrid::NodeNotExistException, IceGrid::NodeUnreachableException

Open the given node stdout file for reading. The file can be read with the returned file iterator.

Parameters

`name` — The node name.
`count` — Specifies where to start reading the file. If negative, the file is read from the begining. If 0 or positive, the file is read from the last `count` lines.

Return Value

An iterator to read the file.

Exceptions

IceGrid::FileNotAvailableException — Raised if the file can't be read.
IceGrid::NodeNotExistException — Raised if the node doesn't exist.
IceGrid::NodeUnreachableException — Raised if the node could not be reached.

IceGrid::FileIterator* openRegistryStdErr(string name, int count) throws IceGrid::FileNotAvailableException, IceGrid::RegistryNotExistException, IceGrid::RegistryUnreachableException

Open the given registry stderr file for reading. The file can be read with the returned file iterator.

Parameters

`name` — The registry name.
`count` — Specifies where to start reading the file. If negative, the file is read from the begining. If 0 or positive, the file is read from the last `count` lines.

Return Value

An iterator to read the file.

Exceptions

IceGrid::FileNotAvailableException — Raised if the file can't be read.
IceGrid::RegistryNotExistException — Raised if the registry doesn't exist.
IceGrid::RegistryUnreachableException — Raised if the registry could not be reached.

IceGrid::FileIterator* openRegistryStdOut(string name, int count) throws IceGrid::FileNotAvailableException, IceGrid::RegistryNotExistException, IceGrid::RegistryUnreachableException

Open the given registry stdout file for reading. The file can be read with the returned file iterator.

Parameters

`name` — The registry name.
`count` — Specifies where to start reading the file. If negative, the file is read from the begining. If 0 or positive, the file is read from the last `count` lines.

Return Value

An iterator to read the file.

Exceptions

IceGrid::FileNotAvailableException — Raised if the file can't be read.
IceGrid::RegistryNotExistException — Raised if the registry doesn't exist.
IceGrid::RegistryUnreachableException — Raised if the registry could not be reached.
