APS RabbitMQ Message Service Provider

User Guide

Version: 1.0.0

Author: Tommy Svensson

Table of Contents

₁ APS RabbitMQ Message Service Provider

This service provides an implementation of APSMessageService using RabbitMQ.

₁.₁ APSMessageService API

Javadoc

public *interface* **APSClusterService**   [se.natusoft.osgi.aps.api.net.messaging.service] {

This service defines a synchronized cluster.

void clusterUpdated(String key, APSBox value)

Receives an updated value.

Parameters

   key - *The key of the updated value.*

   value - *The actual value.*

void update(String key, APSBox value)

Updates a keyed value to the cluster.

Parameters

   key - *This uniquely specifies what value this is. How it is used is upp tp the actual cluster using it.*

   value - *The modified value to update.*

void addUpdateListener(UpdateListener updateListener)

Adds an update listener.

Parameters

   updateListener - *The update listener to add.*

void removeUpdateListener(UpdateListener updateListener)

Removes an update listener.

Parameters

   updateListener - *The listener to remove.*

APSBox getNamedObject(String name)

Gets named cluster-wide object. If it does not exist it will be created.

Parameters

> *name* - *The name of the cluster object to get.*

List<APSBox> getNamedList(String name)

Gets a cluster-wide named list. If it does not exist it will be created.

Parameters

> *name* - *The name of the list to get.*

Map<String, APSBox> getNamedMap(String name)

Gets a cluster-wide named Map. If it does not exist, it will be created.

Do note that this is mostly a convenience. Implementations can (and most will) use the name as a prefix to the map key and then call getNamedObject(expandedKey).

Parameters

> *name* - *The name of the map to get.*

void withLock(String nameToLock, Future future)

Locks a specific entry in the cluster or all entries starting with the specified 'nameToLock'. When a lock is acquired the future is executed and when done the lock is released.

Any other client who tries to update the locked name(s) during the lock period without acquiring their own lock will get an exception.

Parameters

> *nameToLock* - *The name of the value(s) to lock.*

> *future* - *A Future to execute when the lock is acquired.*

List<UpdateListener> listeners = Collections.synchronizedList(new LinkedList<UpdateListener>())

The listeners.

protected void updateListeners(String key, APSBox value)

Updates all listeners.

Parameters

> *key* - *The key of the update.*

> *value* - *The value of the update.*

protected List<UpdateListener> getListeners()

Returns the listeners.

}

---

public *interface* **APSMessageListener**   [se.natusoft.osgi.aps.api.net.messaging.service] {

Listener for APSMessage.

void messageReceived(byte[] message)

This is called when a message is received.

Parameters

message - *The received message.*

}

---

public *interface* **APSMessageService**   [se.natusoft.osgi.aps.api.net.messaging.service] {

This defines a simple message service. Can be implemented by using a message bus like RabbitMQ, Active MQ, etc or just a simple tcpip server or whatever.

Since the actual members are outside of this service API, it doesn't really know who they are and doesn't care, all members are defined by configuration.

void addMessageListener(APSMessageListener listener)

Adds a listener for types.

Parameters

listener - *The listener to add.*

void removeMessageListener(APSMessageListener listener)

Removes a messaging listener.

Parameters

listener - *The listener to remove.*

void sendMessage(byte[] message) throws APSMessagingException

Sends a message.

Parameters

message - *The message to send.*

Throws

APSMessagingException - *on failure.*

public *static abstract class* **AbstractMessageServiceProvider** implements APSMessageService [se.natusoft.osgi.aps.api.net.messaging.service] {

Provides an abstract implementation of the APSMessageService interface.

protected void sendToListeners(byte[] message)

Sends a message to the registered listeners.

Parameters

message - *The message to send.*

protected List<APSMessageListener> getMessageListeners()

Returns the message listeners.

}