



# CHAPTER 12

## Policy Management Functions

---

This chapter provides information about the following policy management functions:

- [asyncExecutePolicy](#), page 12-1
- [createPolicy](#), page 12-3
- [deletePolicy](#), page 12-4
- [enumerateDeviceFilterAttribute](#), page 12-4
- [enumerateSKUFilterAttribute](#), page 12-5
- [listAllPolicies](#), page 12-6
- [listFilteredDevices](#), page 12-6
- [listFilteredSKUs](#), page 12-7
- [readPolicy](#), page 12-7
- [writePolicy](#), page 12-8

### asyncExecutePolicy

#### Synopsis

```
String asyncExecutePolicy(UserToken token, String policy_id, IDStatusListener listener);
```

#### Description

This function executes the given policy by using the policy filters to select devices and stock-keeping units (SKUs) and obtain and deploy licenses for those devices.

This function is nonblocking and returns a request ID to the caller immediately. When calling this function, a client program provides a listener object that implements StatusListener interface. When the function is completed, the onStatus() method in the listener object is invoked.

**Input Parameters**

Parameter	Type	Value	Description
		—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
policy_id	String, mandatory	ID is a string containing up to 64 ASCII characters in the range from x21 to x7A	ID of the Policy object.
listener	IDStatusListener, mandatory	—	The listener object.

**Return**

the PAK ID + "-::-" + SKU name + "-::-" + UDI, and the error code and error message of the operation.

**Error and Exception**

messages. More than one error code and message may be contained in the IDStatus object. Each ID in the status object represents a device ID that participates in the execution of the policy.

The following example shows the error code and message in the onStatus() method:

```
public void onStatus(IDStatus status) {

    // The general error code of the operation.
    int err_code = status.getErrorCode();

    // The general error message of the operation.
    String err_msg = status.getErrorMessage();

    // A list of status for each individual element in the
    // bulk operation.
    IDStatusItem[] items = status.getIDStatusItems()

    // Iterate through the list to get individual status.
    for (int i = 0; i < items.length(); i++) {

        // Get the individual object ID returned by the
        // operation.
        String id = items[i].getID();

        // Get the individual error code corresponding to
        // the object ID.
        int item_err_code = items[i].getErrorCode();

        // Get the individual error message corresponding to
        // the object ID.
        String item_err_msg = items[i].getErrorMessage();
    }
}
```

```

    }
}

```

# createPolicy

## Synopsis

```

PolicyStatus createPolicy(UserToken token, String policy_name, SKUFilter sku_filter,
DeviceFilter dev_filter);

```

			<p>Name of the Policy object.</p> <p>The maximum length of policy name allowed depends on the length of the user name for the user who created the policy. The total length of user name and policy name together must not exceed 63 characters.</p>
sku_filter	SKUFilter, optional	—	<p>The input SKUFilter object specifies the criteria for searching the devices that match the criteria. If the filter is set to null, filtering is not performed.</p>
dev_filter	DeviceFilter, optional	—	<p>The input DeviceFilter object specifies the criteria for searching the devices that match the criteria. If the filter is set to null, filtering is not performed.</p>

```

public class Status implements Serializable {
    private int m_error_code;
    private String m_error_message;
}

public class PolicyStatusItem extends Status implements Serializable {
    private Policy m_policy;

    public Policy getPolicy() {
        return m_policy;
    }
}

```

# deletePolicy

## Synopsis

## Description

## Input Parameters

Parameter	Type	Value	Description

## Return

## Error and Exception

# enumerateDeviceFilterAttribute

## Synopsis

## Description

**Input Parameters**

Parameter	Type	Value	Description
			<ul style="list-style-type: none"> <li>Policy.DeviceAttribute.MODEL</li> <li>Policy.DeviceAttribute.GROUP</li> </ul>

**Return****Error and Exception**

# enumerateSKUFilterAttribute

**Synopsis****Description****Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
attrib	Policy.SKU Attribute, mandatory	—	The attribute name can be Policy.SKUAttribute.FEATURE_NAME.

**Return****Error and Exception**

This function returns a list of all policies that can be accessed by the user, including both the private policies owned by this user and all public policies.

token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
-------	-------------------------	---	--

This function returns an array of string representing the IDs of the policies.

When an error occurs, this function returns null.

This function generates a list of devices obtained by running the given device filter.

token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
dev_filter	DeviceFilter, mandatory	—	The input DeviceFilter object specifies the criteria for searching the list of devices.

This function returns an array of string representing the IDs of the devices.

---


**Input Parameters**


**Return**

**Error and Exception**

# writePolicy

**Synopsis**

```
Status writePolicy(UserToken token, Policy policy);
```
