Table of Contents

About the Java CLI SDK ................................................................. 4
   Audience................................................................................... 4
v-GO PM Java CLI SDK .................................................................. 5
   Installing the v-GO PM CLI.................................................. 5
Java CLI as an SDK ................................................................. 6
   Sample code .......................................................................... 6
   Additional Notes................................................................. 8
Class Definitions ..................................................................... 9
   CLIOperationParser class ............................................. 9
   OperationParser class ................................................. 10
   Operation Class .............................................................. 11
   OperationKeys Interface .................................................. 12
   ExtSearchKeys Interface .................................................. 13
About the Java CLI SDK

The Java Command-line Interface (CLI) Software Development Kit (SDK) is provided with Oracle Enterprise Single Sign-On Provisioning Gateway (ESSO-PG). The SDK exposes several interfaces, a class factory, and supporting types for communicating with the ESSO-PG Web Service. These programming APIs live inside the class library `pmcli.jar`. This is the same library that is the main executable for the Java CLI and is reused for the SDK.

This guide describes how to use the interfaces exposed by the Java CLI in your own applications.

Audience

This guide is intended for experienced Java application programmers responsible for the development of an organization’s provisioning solutions.

<table>
<thead>
<tr>
<th>Acronym or Abbreviation</th>
<th>Full Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSO Agent</td>
<td>ESSO-LM Agent</td>
</tr>
<tr>
<td>SSO Administrative Console</td>
<td>ESSO-LM Administrative Console</td>
</tr>
<tr>
<td>ESSO-LM</td>
<td>Oracle Enterprise Single Sign-On Logon Manager</td>
</tr>
<tr>
<td>ESSO-AM</td>
<td>Oracle Enterprise Single Sign-On Authentication Manager</td>
</tr>
<tr>
<td>ESSO-KM</td>
<td>Oracle Enterprise Single Sign-On Kiosk Manager</td>
</tr>
<tr>
<td>ESSO-PG</td>
<td>Oracle Enterprise Single Sign-On Provisioning Gateway</td>
</tr>
<tr>
<td>ESSO-PR</td>
<td>Oracle Enterprise Single Sign-On Password Reset</td>
</tr>
<tr>
<td>SSO</td>
<td>ESSO-LM</td>
</tr>
<tr>
<td>FTU</td>
<td>First Time Use</td>
</tr>
<tr>
<td>SSO Agent</td>
<td>ESSO-LM Agent</td>
</tr>
</tbody>
</table>
Welcome to the ESSO-PG Java CLI SDK Guide

The Java Command-line Interface (CLI) Software Development Kit (SDK) is provided with ESSO-PG. The SDK exposes several interfaces, a class factory, and supporting types for communicating with the ESSO-PG Web Service. These programming APIs live inside the class library pmcli.jar. This is the same library that is the main executable for the Java CLI and is reused for the SDK.

This document describes how to use the interfaces exposed by the Java CLI in your own applications.

Installing the v-GO PM CLI

The ESSO-PG CLI must be installed prior to performing the steps in the document. Please refer to the ESSO-PG Installation and Setup Guide for information on installing the ESSO-PG CLI.

The Java CLI is located under <home directory>\v-GO PM\Client\Java<version>.
Java CLI as an SDK

To use the Java CLI as an SDK, follow these steps:

1. Add pmcli.jar and supporting libraries to the CLASSPATH.
2. Import the provisioning classes into your application.
3. Create an instance of the ProvisioningConnection class.
4. Create an instance of the CLIOperationParser class.
5. Define the operation and operation parameters using a StringMap.
6. Create an instance of the Operation using the object instance created in step 4.
7. Set execution time (otherwise it defaults to “Now”).
8. Send Operation instance (step 6) to the Web service using the ProvisioningConnection (step 3) instance.
9. Retrieve success and results of operation.

Sample code

This sample code shows a simple program that implements each of these steps:

```java
// Import these classes into your application
import com.passlogix.vgo.pm.cli.*;
import com.passlogix.vgo.pm.operations.*;

// My routine for calling the web service
void CallWebService(/* Parameters */)
{
    // Arguments to ProvisioningConnection are defined as:
    // URL: the webservice URL
    // strAgent: user-defined name for the client agent
    // strUsername: the username to connect as
    // strPassword: the password to connect as
    ProvisioningConnection conn = new ProvisioningConnection(
        strURL, strAgent, strUsername, strPassword);
```
try {

    // Begin execution of instruction
    CLIOperationParser opParser = CLIOperationParser.newInstance();
    Operation.StringMap options = new Operation.StringMap();

    // Use OperationKeys class for most options
    // Use ExtSearchKeys class for ExtSearch operation
    options[OperationKeys.USERID] = "davidh";
    options[OperationKeys.APPLICATION] = "Visual SourceSafe";
    // And so forth...

    // strOper can be equal to any operation defined in
    // CLIOperationParser.
    Operation oper = opParser.parse(strOper, options);
    // Set the execution time of instruction.
    // If not set, defaults to Now
    oper.setExecTime(dtExec);
    conn.sendInstruction(oper);

    // Get results
    if (!oper.getSuccess()) {
        String strMsg = String.format("The command failed: id=%s, msg=%s", oper.getCommandID(), oper.getError());
        return;
    }
}
// Retrieve command ID and result attributes. For some
// commands, one or both of these is empty. See the .NET
// CLI/SDK documentation for more info on the command ID
// and format of result attributes.
String strID = oper.getCommandID();
CollectionsMap map = oper.getResultAttributes();
}

catch (Exception ex)
{
    // print exception
}
}

**Additional Notes**

Refer to the *ESSO-PG.Net CLI SDK Guide* to see all the available options for each operation. The available operations are defined as static members of the `CLIOperationParser` class. All of the available options and parameters for the supported operations are defined in the `OperationKeys` and `ExtSearchKeys` sections of this document.
Class Definitions

The following class definitions show the important constants and methods needed to programmatically send a request to the ESSO-PG Web Service.

**CLIOperationParser class**

This class inherits from `OperationParser`. An instance of itself can be created by calling `newInstance()`. When an instance exists, it can be used to create `Operation` objects representing the specific request to be executed on the server:

```java
// All supported operations defined as constant strings
static public final String ADD_CREDENTIAL        = "add_credential";
static public final String MODIFY_CREDENTIAL     = "modify_credential";
static public final String DELETE_CREDENTIAL     = "delete_credential";
static public final String DELETE_USER           = "delete_user";
static public final String STATUS                = "status";
static public final String CANCEL                = "cancel";
static public final String CHECK_SERVER          = "check_server";
static public final String GET_SETTINGS          = "get_settings";
static public final String GET_SCHEMA            = "get_schema";
static public final String SET_SETTINGS          = "set_settings";
static public final String EXT_SEARCH            = "ext_search";

// creates a new instance of this parser
static public CLIOperationParser newInstance();

// Prints the results to an output stream of choice
public void printResults(PrintStream out, Operation oper);
```
OperationParser class

This class is the base class for CLIOperationParser. It defines methods for supporting additional operations and creating Operation objects:

```java
// Add a new provisioning operation and its support class. The
// supporting class must be derived from the abstract Operation class.
// This method is intended for internal use.
public void addOperation(String strOper, Class<? extends Operation> c)
```

```java
// Create an instance of the Operation object for the given
// provisioning instruction. This instruction follows the same format
// as that passed in the command line.
public Operation parse(String strInstr)
    throws InstantiationException, IllegalAccessException
```

```java
// Create an instance of the Operation object based on the operation
// name.
public Operation parseNoOpt(String strOper)
    throws InstantiationException, IllegalAccessException
```

```java
// Create an instance of the Operation object based on the operation
// name and its parameters (specified as a map of key/value pairs).
public Operation parse(String strOper, Operation.StringMap options)
    throws InstantiationException, IllegalAccessException
```
**Operation Class**

The Operation Class is the base class for all Operations supported by the Java CLI. This class is responsible for constructing the correct message to send to the Web service and for retrieving and storing the response. The following methods can be used to query the results:

```java
// Get the raw xml response from the server
public String getResponse()

// Was this operation executed successfully?
public boolean getSuccess()

// Get the GUID associated with this operation after it is executed.
// This can be an empty string if no GUID is associated with the
// operation.
public String getCommandID()

// Get any error message if getSuccess returns false.
public String getError()

// Set the execution time of this operation on the server. If not set,
// the Operation will execute immediately. Otherwise the Operation
// will not execute until the given time.
public void setExecTime(Date dtExec)

// Get the result attributes array if the operation was successful.
// An empty CollectionsMap cab be returned if there are no results
// other than success to return. The format of CollectionsMap is a
// name/value pair map of lists or other maps. The exact format of
// which depends on the operation executed. More info can be found in
```
// the .NET CLI/SDK documentation.

public CollectionsMap getResultAttributes()

// Execute the operation. You generally should not call this method
// directly. Instead call ProvisioningConnection.sendInstruction(...)
// passing the Operation object to it.

public String send(ProvisioningConnection conn)
    throws PMCLIException, RemoteException

**OperationKeys Interface**

The **OperationKeys** interface defines all the possible parameters an **Operation** can accept. The parameters are specified as keys to the StringMap followed by their value. The exact subset of keys an **Operation** supports is described in the *ESSO-PG.Net CLI SDK Guide*:

```java
public interface OperationKeys
{
    static public final String USERID  = "sso_userid";
    static public final String APPLICATION    = "sso_application";
    static public final String DESCRIPTION    = "sso_description";
    static public final String APP_USERID     = "sso_app_userid";
    static public final String PASSWORD       = "sso_password";
    static public final String OTHER1         = "sso_other1";
    static public final String OTHER2         = "sso_other2";
    static public final String GUID           = "command_id";
    static public final String NAME           = "name";
    static public final String VALUE          = "value";
}
```
**ExtSearchKeys Interface**

The ExtSearchKeys interface defines the parameters supported for the `ExtSearch` operation. The `OperationKeys` interface does not apply for this operation. Acceptable parameters must come from this list:

```java
public interface ExtSearchKeys {
    // Supported keys for ExtSearch
    static public final String OPTION_CATALOG = "catalog";
    static public final String OPTION_USERID = "userId";
    static public final String OPTION_APPLICATION = "applicationName";
    static public final String OPTION_EVENTTYPE = "eventType";
    static public final String OPTION_STARTDATE = "startDate";
    static public final String OPTION_ENDDATE = "endDate";
    static public final String OPTION_LOGON = "logon";
    static public final String OPTION_SHOWLOGONS = "returnLogons";
    static public final String OPTION_SHOWPENDING = "returnInstructions";
    static public final String OPTION_UIDMATCH = "uidMatch";

    // Possible values for OPTION_UIDMATCH key
    static public final String MATCH_EQUAL = "equal";
    static public final String MATCH_SUBSTRING = "substring";

    // Possible values for OPTION_CATALOG key
    static public final String CATALOG_APPS = "Applications";
    static public final String CATALOG_EVENTLOG = "EventLog";
    static public final String CATALOG_USERS = "Users";
}
```