

# Example API and Toolkit README

- [Introduction](#)
- [Requirements](#)
- [Getting Started](#)
  - [Building the Toolkit](#)
- [Examples](#)
- [Accessing Additional API's](#)

## Introduction

The purpose of the unifiedconfig example toolkit is to provide an easy means of programmatically interacting with the unifiedconfig REST web service in Java. The examples provided include:

- A simple client that handles authentication, making requests, and receiving responses
- Classes to handle interacting with the Agent, Skill Group, and Attribute APIs
- Examples of creating, associating, and deleting Agents, Skill Groups, and Attributes

## Requirements

- Java 1.6.0\_30 or future 1.6 update
- Apache Maven 3.0.3
- Access to a 9.0(4) or greater CCE Data Server or External HDS

## Getting Started

The unifiedconfig example toolkit is downloadable as a zip file from Cisco's public website. Contained in the zip is a maven project and example source code for interacting with the REST service.

## Building the Toolkit

To build the Toolkit, simply use the maven install command from the base level of the project:

```
"mvn clean install -DskipTests"
```

To run the functional tests, pass in the IP address or hostname of the Data Server or External HDS, username (fully qualified with domain name), and password.

**Note: Execution of the functional tests create and delete objects on the server as part of the test**

```
"mvn clean install -PTESTS -Dtestbed=10.86.135.193 -Dusername=administrator@test.com -Dpassword=abc123"
```

## Examples

### Agent Skill Group Demo (PCCE Only)

The AgentSkillGroupDemo class can be found in the examples package.

**To launch the program from the command line** (passing in the real IP, username, and password for a PCCE Data Server), **type:**

```
mvn exec:java -Dexec.mainClass="com.cisco.ccbu.cce.unifiedconfig.toolkit.examples.AgentSkillGroupDemo" -Dexec.args="10.86.135.193 administrator@test.com abc123="
```

**The demo performs the following operations:**

- 1) Creates three Agents and one Skill Group
- 2) Performs a GET operation on an Agent URL that does not exist and prints out the error

- 3) Updates one Agent to include the Skill Group
- 4) Performs two list functions: One with search parameters and one without search parameters
- 5) Deletes the four objects.

### **Agent Attribute Demo (PCCE and UCCE)**

The AgentAttributeDemo class can be found in the examples package (If running under UCCE at least one agent must exist before launching the test).

**To launch the program from the command line** (passing in the real IP, username, and password for a PCCE Data Server or External AW HDS), type:

```
mvn exec:java -Dexec.mainClass="com.cisco.ccbu.cce.unifiedconfig.toolkit.examples.AgentAttributeDemo" -Dexec.args="10.86.135.193 administrator@test.com abc123="
```

**The demo performs the following operations:**

- 1) Creates one Agent (or looks up one Agent if UCCE)
- 2) Creates one Attribute
- 3) Performs an update operation on the Agent to associate it with the Attribute and give it a value
- 4) Perform an update operation on the Agent to remove the association and re-associate to any previously existing attributes
- 5) Deletes created objects: Agent and Attribute for PCCE, just Attribute for UCCE.

## **Accessing Additional API's**

Additional APIs can easily be built using this same methodology by following the PCCE API Developers Guide.

For each new API, follow these steps:

- 1) Create a class that extends BaseApiBean
- 2) Add the annotations @Path("<path suffix>") @XmlElement(name = "<bean name>") where path suffix is the lowercase suffix to the unified config URL and bean name is the camel case root XML element for the API.
- 3) Add in the attributes with get and set methods for API. Use the ReferenceBean class for all references (see AgentDeskSettings and SkillGroups in Agent for examples)
- 4) Create a class that extends the BaseApiListBean and implement the methods getItems and setItems methods, and add the @Path and @XmlElement annotations on the class and the @XmlElementWrapper and @XmlElement annotations on the getItems method.