



AXL Introduction

A practical approach using Python

Johannes Krohn
Technical Marketing Engineer
15 May 2019

AXL Introduction

AXL API Introduction

- The **Administrative XML Web Service (AXL)** is an XML/SOAP based interface that provides a mechanism for inserting, retrieving, updating and removing data from the Unified Communication configuration database.
- <https://developer.cisco.com/site/axl/>
- Thick AXL – API defines specific objects that can be created, removed, queried, or updated
- Thin AXL – Provides a mechanism to perform direct SQL queries / updates

Administrative XML Configuration API

- Read/Modify UCM Configuration Database
- Methods for All Database Objects
 - list*
 - add*
 - update*
 - get*
 - remove*
- Thin AXL methods:
 - ExecuteSQLupdate
 - ExecuteSQLquery
- Service port: <https://<server>:8443/axl/>
- Authentication:
 - Member of **AXL API Access** Group
 - Create group with custom permissions or use **Standard AXL Read Only API Access** Role

XML

- eXtensible Markup Language
- Opening / Closing tag for elements
- Not validated against schema
- Example:

```
<person>
    <lastname>Robbins</lastname>
    <givenname>Chuck</givenname>
</person>
```

- <http://www.w3.org/TR/2006/REC-xml-20060816/>

XML – Schema Definition

- Names, hierarchy, meaning of elements and attributes defined by Schema
- XML schema defined by W3C
- Primer: <http://www.w3.org/TR/xmlschema-0/>
- Example:
 - ```
<xsd:element name="person" type="PersonType"/>
<xsd:complexType name="PersonType">
 <xsd:sequence>
 <xsd:element name="lastname" type="xsd:string"/>
 <xsd:element name="givenname" type="xsd:string"/>
 </xsd:sequence>
</xsd:complexType>
```

# SOAP

- The specification formerly known as **Simple Object Access Protocol**
- W3C specification: <http://www.w3.org/TR/soap/>
- Exchange of structured and typed information based on XML
- SOAP specification defines
  - SOAP message format
  - How to send and receive messages
  - Data encoding

# SOAP Message Structure

## SOAP Message

Envelope (**SOAP-ENV:Envelope**)

Header (**SOAP-ENV:Header**) (optional)

Body (**SOAP-ENV:Body**) (required)

Fault (**SOAP-ENV:Fault**) (optional)

# Web Services Definition Language (WSDL)

- W3C: <http://www.w3.org/TR/wsdl20/>
- XML-based format (grammar) to describe web services
- Defines four pieces of data:
  - Publicly available methods; interface description, formats
  - Data type information for requests and responses
  - Binding; which transport protocol
  - Address information – where to find the service

# AXL Request Schema – addPhone

AXLAPI.wsdl

```
<operation name="addPhone">
 <soap:operation soapAction="CUCM:DB ver=11.5 addPhone" style="document"/>
 <input>
 <soap:body use="literal"/>
 </input>
 <output>
 <soap:body use="literal"/>
 </output>
 <fault name="fault">
 <soap:fault name="fault" use="literal"/>
 </fault>
</operation>

<operation name="addPhone">
 <input message="s0:addPhoneIn"/>
 <output message="s0:addPhoneOut"/>
 <fault name="fault" message="s0:AXLError"/>
</operation>
```

# AXL Request Schema - addPhone

AXLAPI.wsdl

```
<message name="addPhoneIn">
 <part element="xsd1:addPhone" name="axlParams"/>
</message>
<message name="addPhoneOut">
 <part element="xsd1:addPhoneResponse" name="axlParams"/>
</message>
```

# AXL Request Schema - addPhone

AXLSOAP.xsd

```
<xsd:element name="addPhone" type="axlapi:AddPhoneReq"/>
<xsd:element name="addPhoneResponse" type="axlapi:StandardResponse"/>

<xsd:complexType name="AddPhoneReq">
 <xsd:complexContent>
 <xsd:extension base="axlapi:APIRequest">
 <xsd:sequence>
 <xsd:element name="phone" type="axlapi:XPhone"/>
 </xsd:sequence>
 </xsd:extension>
 </xsd:complexContent>
</xsd:complexType>
```

# AXL Request Schema - addPhone

## AXLSOAP.xsd

```
<xsd:complexType name="XPhone">
 <xsd:sequence minOccurs="0">
 <xsd:element maxOccurs="1" minOccurs="1" name="name" nillable="false" type="axlapi:UniqueString128">
 <xsd:annotation>
 <xsd:documentation>The device name, using only URL-friendly characters</xsd:documentation>
 </xsd:annotation>
 </xsd:element>
 <xsd:element maxOccurs="1" minOccurs="0" name="description" nillable="false" type="axlapi:String128">
 <xsd:annotation>
 <xsd:documentation>Optional description of the device</xsd:documentation>
 </xsd:annotation>
 </xsd:element>
 <xsd:element maxOccurs="1" minOccurs="1" name="product" nillable="false" type="axlapi:XProduct">
 <xsd:annotation>
 <xsd:documentation>Product ID string. read-only except when creating a device.</xsd:documentation>
 </xsd:annotation>
 </xsd:element>
 <xsd:element maxOccurs="1" minOccurs="1" name="class" nillable="false" type="axlapi:XClass">
 <xsd:annotation>
 <xsd:documentation>Class ID string. Class information is read-only except when creating a device.</xsd:documentation>
 </xsd:annotation>
 </xsd:element>
```

# AXL Versioning

- Abstracts developer from DB schema changes
  - Thick AXL methods only, Thin AXL (direct SQL) does not offer backward compatibility
- Maintains Release minus 2 backward compatibility
  - Developers writing to Unified CM 12.0(1) will not have to make changes until 15.0(1)
  - Oldest supported schema is always the default
- Developer specifies desired schema version via the **SOAPAction** header
  - E.g., SOAPAction: **CUCM:DB ver=12.0**

# Performance

- Dynamic throttling
  - Single request limited to <8MB data
  - Concurrent request limited to <16MB
- Up to 1500 writes per minute
- Intelligently accepts or rejects requests
  - Row fetch returned when data limits are exceeded
- Always enabled

# Cisco AXL Toolkit

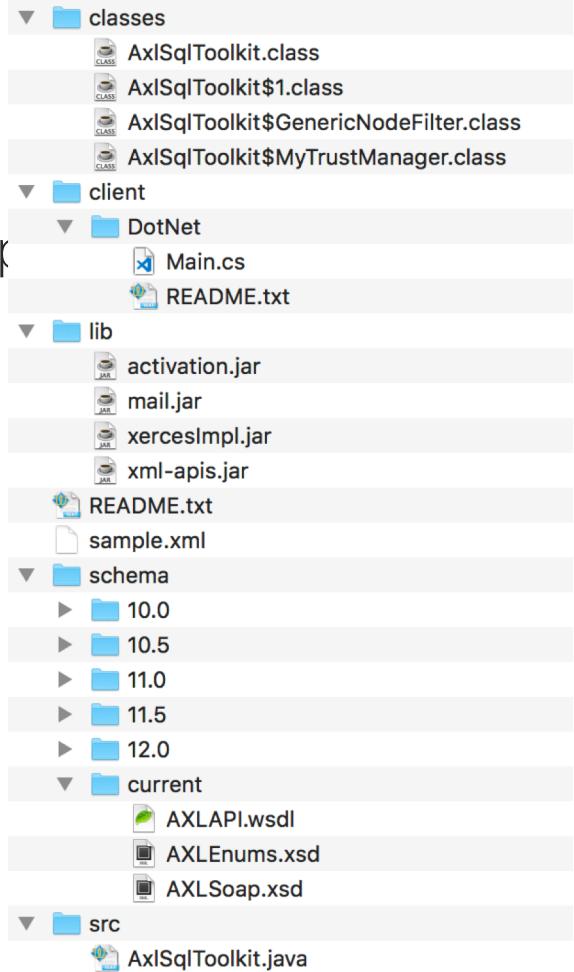
- Download from **Unified CM Administration > Application > Plugins**

The screenshot shows the Cisco Unified CM Administration interface. The top navigation bar includes links for Cisco Unified CM Administration, Navigation, Go, axl, Search Documentation, About, and Logout. Below the navigation is a main menu with options like System, Call Routing, Media Resources, Advanced Features, Device, Application, User Management, Bulk Administration, and Help. The current page is titled "Find and List Plugins". A status message indicates "13 records found". The main content area displays a table of plugins. The table has columns for "Plugin Name" (sorted by name), "Description", and "Actions". One row is visible for the "Cisco AXL Toolkit", which is described as enabling developers to create, read, update, and delete provisioning objects. The "Description" column contains a long SHA12 hash. The bottom right of the table shows a "Rows per Page" dropdown set to 50.

Plugin Name	Description
<a href="#">Download Cisco AXL Toolkit</a>	Cisco Administrative XML (AXL) Toolkit enables Developers to create applications that create, read, update and delete provisioning objects on the Cisco Unified Communications Manager Publisher. The zip file contains Java-based libraries that use SOAP over HTTP/HTTPS to send and receive AXL requests and responses. Install this toolkit on Developer workstations where AXL applications will be developed. SHA12(/usr/local/thirdparty/jakarta-tomcat/webapps/plugins/axlsq toolkit.zip)= 1d:26:46:64:7a:9a:35:06:d2:cd:ca:40:90:96:d0:cd:da:37:e8:45:73:37:13:e5:ae:17:05:ae:b2:b0:3c:28:e4:22:7b:a7:71:f7:c9:e:c5:09:43:a3:e6:03:6b:15:c6:c2:99:0a:eb:2d:8f:5e:15:d7:48:e8:2a:d5:c8:1c

# Cisco AXL Toolkit

- Schema folder contains AXL API schema for supported releases
  - AXLAPI.wsdl – WSDL file
  - AXLEnums.xsd – Enum type definitions
  - AXLSOap.xsd – Type definitions
- Sample Java classes and .NET code



# Documentation

- AXL Schema Reference
  - <https://developer.cisco.com/docs/axl-schema-reference/>
- AXL Developer Guide
  - <https://developer.cisco.com/docs/axl/#12-0-axl-developer-guide>
- UCM Data Dictionary
  - <https://developer.cisco.com/docs/axl/#12-0-cucm-data-dictionary>

# Unified CM Serviceability APIs

<https://developer.cisco.com/site/sxml/>

- **Real-Time Information** (RisPort) – Provides the current connection status of phones, devices, and applications connected to Cisco Unified Communications Manager (Unified CM).  
<https://<server>:8443/realtimeservice2/services/RISService70?wsdl>
- **Performance Monitoring** (PerfMon) – Provides real-time event feeds to monitor the status and health of Cisco Unified CM.  
<https://<server>:8443/perfmonservice2/services/PerfmonService?wsdl>

- **CDRonDemand** – SOAP/HTTPS interface to query the Unified CM Call Detail Records (CDR) Repository.

<https://<server>:8443/realtimeservice2/services/CDRonDemandService?wsdl>

- **Log Collection** – Retrieval of trace files and logs

<https://<server>:8443/logcollectionservice2/services/LogCollectionPortTypeService?wsdl>

- **Service Control** – Activate / Deactivate / Start / Stop Services

<https://<server>:8443/controlcenterservice2/services/ControlCenterServices?wsdl>

# Troubleshooting

# Quick Functionality Check

- Go to the AXL API URL via a web browser
- For instance, enter <https://cm1:8443/axl/> in the address text box
- When prompted for user name and password, use the standard administrator login, or use the configured AXL user
- Look for a plain page that states the AXL listener is working and accepting requests, but only communicates via POST

## **Cisco CallManager: AXL Web Service**

The AXL Web Service is working and accepting requests. Use HTTP POST to send a request.

- This verifies functionality and user access

# Enable AXL Traces

- Detailed AXL traces can be enabled in Cisco Unified Serviceability settings

The screenshot shows the 'Trace Configuration' page of the Cisco Unified Serviceability interface. The top navigation bar includes links for Alarm, Trace, Tools, Snmp, CallHome, and Help. Below the navigation is a toolbar with Save and Set Default buttons. The main area is titled 'Trace Configuration' and contains several sections:

- Status:** Ready
- Select Server, Service Group and Service:** Server\* ucm-pub.dcloud.cisco.com--CUCM Voice/Video, Service Group\* Database and Admin Services, Service\* Cisco AXL Web Service (Active). An unchecked checkbox 'Apply to All Nodes' is also present.
- Trace On:** A checked checkbox.
- Trace Filter Settings:** A dropdown menu set to 'Debug' with a 'Go' button, and a checked checkbox 'Enable All Trace'.
- Trace Output Settings:** Fields for 'Maximum No. of Files\*' (10) and 'Maximum File Size (MB)\*' (1).

At the bottom are Save and Set Default buttons, and a note: **i**\* - indicates required item.

# Analyze AXL logs

- Use Real-Time Monitoring Tool to access AXL log
- log contains incoming AXL requests and outgoing responses

# Analyze AXL logs

```
2019-04-18 03:15:46,157 DEBUG [http-bio-443-exec-5] filters.TimingFilter - Received request 1555496324288 from administrator at IP 10.16.66.88
2019-04-18 03:15:46,158 DEBUG [http-bio-443-exec-5] wrappers.RequestHeaderWrapper - Inside Request Header Wrapper
2019-04-18 03:15:46,158 DEBUG [http-bio-443-exec-5] filters.AuthenticationFilter - Operation:get api:getCCMVersion
2019-04-18 03:15:46,158 DEBUG [http-bio-443-exec-5] filters.ThrottlingFilter - Successfully set the value of counter: 4 value: 0
2019-04-18 03:15:46,158 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - Checking request version [10.0]
2019-04-18 03:15:46,158 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - Checking if requested api [getCCMVersion] the implementedHandlers list
2019-04-18 03:15:46,158 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - [getCCMVersion] is not in the implementedHandlers list
2019-04-18 03:15:46,159 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - Checking if version is 8.x
2019-04-18 03:15:46,159 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - function to check if the version is 8.x
2019-04-18 03:15:46,160 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - version is not 8.x
2019-04-18 03:15:46,160 INFO [http-bio-443-exec-5] servletRouters.AXLAlpha - Executing api: getCCMVersion in axis
2019-04-18 03:15:46,160 DEBUG [http-bio-443-exec-5] wrappers.RequestNamespaceWrapper - Inside Request Wrapper
2019-04-18 03:15:46,160 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - AXL REQUEST :
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns="http://www.cisco.com/AXL/API/10.0">
 <soapenv:Header/>
 <soapenv:Body>
 <ns:getCCMVersion>
 </ns:getCCMVersion>
 </soapenv:Body>
</soapenv:Envelope>
2019-04-18 03:15:46,160 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - Request processed by AXIS
2019-04-18 03:15:46,162 DEBUG [http-bio-443-exec-5] axlapiservice.Handler - dbConnector Initialization in handler.java
2019-04-18 03:15:46,162 DEBUG [http-bio-443-exec-5] axlapiservice.Axl - Connection given to current thread
2019-04-18 03:15:46,163 DEBUG [http-bio-443-exec-5] axlapiservice.GetCCMVersionHandler - select cv.version, pn.nodeid from componentversion as cv, processnode as pn where cv.softwarecomponent = 'cm-ver' and pn.pkid = cv.fkprocessnode order by nodeid
2019-04-18 03:15:46,171 DEBUG [http-bio-443-exec-5] axlapiservice.Axl - Connection closed and hashmap entry removed in AXL.java closing connection
2019-04-18 03:15:46,174 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - <?xml version='1.0' encoding='UTF-8'?><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Body><ns:getCCMVersionResponse xmlns:ns="http://www.cisco.com/AXL/API/10.0"><return><componentVersion><version>11.5.1.12900(21)</version></componentVersion></return></ns:getCCMVersionResponse></soapenv:Body></soapenv:Envelope>
2019-04-18 03:15:46,174 DEBUG [http-bio-443-exec-5] servletRouters.AXLAlpha - Finished processing request
```

# Database Access

# Database Dictionary

- Configuration in Communications Manager is stored in relational database
- Database Dictionary documents all existing tables in Communications Manager Database
  - field types
  - database constraints
  - relations
- Common Table Relationships
- Schema changes in recent releases

# Table relations (1)

- pkid is the primary key ID. It is always of type GUID.
- Fields that begin with the letters "fk" represent foreign keys into another table. The name of the field following the "fk" prefix up to but not including an underscore character is the name of the related table. The field in related table is always pkid. and is a GUID.
- Examples in table device:  
`device.fkenduser → enduser.pkid`
  - `device.fkenduser_mobility → enduser.pkid`
  - `device.fkcallingsearchspace → callingsearchspace.pkid`
  - `device.fkcallingsearchspace_aar → callingsearchspace.pkid`

## Table relations (2)

- Fields that begin with the letters "ik" represent internal keys into the same table.
- Example in table device:
  - device.ikdevice\_primaryphone → device.pkid

# Table relations (3)

- Fields that begin with a "tk" represent an enumerated type. This field is related to a table whose name begins with "Type" and ends with the name of the field following the prefix up to but not including an underscore character. The field in the related table is always "enum" and is an integer.
- Examples in table device
  - tkclass      **tkclass** → **type**class.enum
  - tkdeviceprotocol → typedeviceprotocol.enum
  - tkmodel → typemodel.enum
  - tkproduct → typeproduct.enum

# SQL

- language for retrieval and management of data stored in relational database management systems
- originally called SEQUEL (structured english query language)
- standardized by ISO/IEC

# SQL Statements on the CLI

- SQL statements can be executed on the CLI using „run sql“
- “&” can't be used on the CLI
- Can be used to test SQL statements to be used in scripts
- Example:

```
• admin:run sql select enum,name from typemodel where tksubclass=1
• enum name
• ===== =====
• 20 SCCP Phone
• 134 Remote Destination Profile
• 30027 Analog Phone
• 30028 ISDN BRI Phone
• 2 Cisco 12 SP+
• 3 Cisco 12 SP
• ...
```

# Example: dialplan

- All DNs and patterns are stored in table numplan

dnorpattern	fkroutepartition	tkpatternusage
Sample Line Template with TAG usage examples	NULL	11
1111	a5a6c703-1191-c371-9563-7822f04f5d3f	2
AutoReg_ULT	a5a6c703-1191-c371-9563-7822f04f5d3f	11
2000	a5a6c703-1191-c371-9563-7822f04f5d3f	5
\+!	f4d2a38f-490f-0f76-f740-b1a5b7d170d1	5
\+!#	f4d2a38f-490f-0f76-f740-b1a5b7d170d1	5
\+1[2-9]XX[2-9]XXXXXX	7463df44-6b77-fa70-7576-a3bbd8c6084c	5
\+14085554006	a5a6c703-1191-c371-9563-7822f04f5d3f	2
\+14085554019	a5a6c703-1191-c371-9563-7822f04f5d3f	2
911	05a1e0ee-b7f8-b688-942c-1d9bedb872d9	5
\+19195551055	a5a6c703-1191-c371-9563-7822f04f5d3f	2

- Which partition? What type of pattern?
- Let's look in tables routepartition and typepatternusage

# Example: dialplan

```
admin:run sql select dnorpattern, routepartition.name, typepatternusage.name from
numplan,routepartition,typepatternusage where fkroutpartition=routepartition.pkid and
tkpatternusage=typepatternusage.enum
```

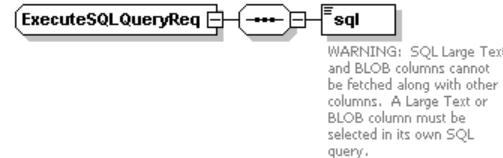
dnorpattern	name	name
\+!	PSTNInternational	Route
\+!#	PSTNInternational	Route
\+1[2-9]XX[2-9]XXXXXX	USPSTNNational	Route
1111	DN	Device
AutoReg_ULT	DN	Device template
2000	DN	Route
\+14085554006	DN	Device
\+14085554019	DN	Device
\+19195551055	DN	Device

- Assignment of DNs to devices is in table devicenumplanmap

# Database access via AXL (Thin AXL)

- AXL provides methods to execute SQL queries and updates:
- executeSQLQuery (SELECT)
- executeSQLUpdate (INSERT, UPDATE, DELETE)
- both methods take a SQL command as argument

# executeSQLQuery



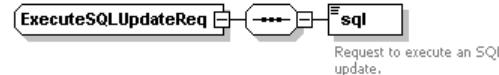
- Result is a sequence of rows
- each row has a number of sub-elements, one per column of the resulting table

```
<SOAP-ENV:Envelope ...>
<SOAP-ENV:Header/>
<SOAP-ENV:Body><axl:executeSQLQueryResponse ...>
<return>
 <row>
 <pkid>8555d448-5818-8494-e16a-de099e9a403c</pkid>
 <realm>jkrohn</realm>
 <userid>jkrohn</userid>
 <passwordreverse>...</passwordreverse>
 </row>
</return>
</axl:executeSQLQueryResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

- sequence can be empty!

- <SOAP-ENV:Body><axl:executeSQLQueryResponse ...>
 <return/>
 </axl:executeSQLQueryResponse>
 </SOAP-ENV:Body>

# executeSQLUpdate



- Writing to the database can destroy database integrity and thus compromise core functionality!
- Very limited to no integrity checks!
- delete means deleted ☺
- result is an element indicating the number of rows updated

```
<SOAP-ENV:Envelope ...>
<SOAP-ENV:Header/>
 <SOAP-ENV:Body>
 <axl:executeSQLUpdateResponse ...>
 <return>
 <rowsUpdated>1</rowsUpdated>
 </return>
 </axl:executeSQLUpdateResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

# Tools

# SoapUI, <https://www.soapui.org/>



- API test automation framework
- Free (open source) version availability
- Great tool to inspect the AXL WSDL, create, and execute AXL requests

A screenshot of the SoapUI 5.5.0 application window. The top menu bar includes Empty, SOAP, REST, Import, Save All, Forum, Trial, Preferences, Proxy, and Endpoint Explorer. The Endpoint Explorer tab is selected. The main area shows a "Request 1" panel with the URL https://198.18.133.3:8443/axl/. The "Raw" tab displays the XML of the AXL request:

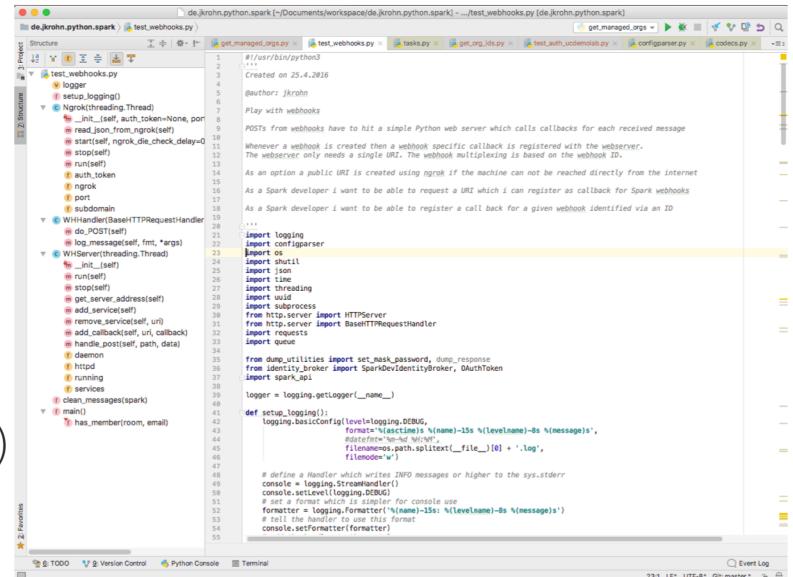
```
POST https://198.18.133.3:8443/axl/ HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
SOAPAction: "CUCM:DB ver=10.0 getCCMVersion"
Content-Length: 250
Host: 198.18.133.3:8443
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
Authorization: Basic YWRtaW5pc3RyYXRvcjkQ2xvdW
```

The "XML" tab displays the XML of the response:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Body>
<ns:getCCMVersionResponse xmlns:ns="http://www.cisco.com/AXL/API/10.0">
<return>
<version>11.5.1.12900(21)</version>
</componentVersion>
</componentVersion>
</return>
</ns:getCCMVersionResponse>
</soapenv:Body>
</soapenv:Envelope>
```

# IDE - Integrated Development Environment

- Helps to develop and test your application
- Features
  - GUI
  - Editor
  - Build automation
  - Syntax highlighting
  - Debugger
  - Integration w/ revision control system (e.g. Git)



The screenshot shows a Java-based IDE interface with the following details:

- Project Structure:** A tree view showing a project named "de.jkrohn.python.spark" containing files like "test\_webhooks.py", "logger", "Nginx(reading Thread)", "WHDHandler(BaseHTTPRequestHandler)", and "main".
- Code Editor:** The main window displays the content of "test\_webhooks.py". The code is a Python script that imports various modules and sets up a web server using Nginx and BaseHTTPServer. It includes comments explaining the purpose of different parts of the code, such as handling POST requests and registering callbacks for webhooks.
- Toolbars and Status Bar:** Standard IDE toolbars are visible at the top, and the status bar at the bottom shows "23:1 LF2 UTF-8! Git: master" and "Event Log".

# Syntax Highlighting

- What Do you prefer?
- This?

```
def get_attachments():

 def assert_folder(p_state, base_path, room_id, room_folder):
 """ make sure that the folder is created for the room
 """
 if not os.path.exists(base_path):
 # base directory needs to be created
 logging.debug('Base directory %s does not exist' % base_path)
 os.mkdir(base_path)

 full_path = os.path.join(base_path, room_folder)

 if room_id not in p_state:
 p_state[room_id] = {}
 room_state = p_state[room_id]

 if 'folder' not in room_state:
 logging.debug('No previous folder for room %s' % room_folder)
 # the folder for this room hasn't been created before
 i = 0
 base_folder = room_folder
 while True:
 full_path = os.path.join(base_path, room_folder)
 try:
 os.mkdir(full_path)
 logging.debug('Created folder %s' % full_path)
 except _FileExistsError:
 break
```

# Syntax Highlighting

- What Do you prefer?
- Or this?

```
def get_attachments():

 def assert_folder(p_state, base_path, room_id, room_folder):
 """ make sure that the folder is created for the room
 """
 if not os.path.lexists(base_path):
 # base directory needs to be created
 logging.debug('Base directory %s does not exist' % base_path)
 os.mkdir(base_path)

 full_path = os.path.join(base_path, room_folder)

 if room_id not in p_state:
 p_state[room_id] = {}
 room_state = p_state[room_id]

 if 'folder' not in room_state:
 logging.debug('No previous folder for room %s' % room_folder)
 # the folder for this room hasn't been created before
 i = 0
 base_folder = room_folder
 while True:
 full_path = os.path.join(base_path, room_folder)
 try:
 os.mkdir(full_path)
 logging.debug('Created folder %s' % full_path)
 except FileExistsError:
```

# Live Debugger

- Live Debugger allows to
  - Set breakpoints
  - Check variables
  - Evaluate expressions

→ Essential for effective SW development

```
workspace - Debug - de.jkrohn.python.spark/get_attachments.py - Eclipse
```

Variables

Name	Value
base_path	str: /Users/jkrohn/Documents/Spark.Attachments
p_state	dict: {Y2tY29zcGFyazovL3VzL1JPT00vMjYzhNjA1ZjkyZD0kMWU2LThNWUUYjdYcyMzA2: str: CoS_for_BYO-PSTN}
room_id	str: Y2tY29zcGFyazovL3VzL1JPT00vNzA3ZDhNzAMGEzMCoMWU3L7gsYzMODkzYjhjZi
room.state	dict: {lastActivity: 2017-06-15T14:02:40.508Z, folder: CoS_for_BYO-PSTN, messages: []}

Breakpoints

Expressions

Console

```
get_attachments.py [debug] [/usr/local/bin/python3]
```

# IDEs for Python

- IDLE (Standard IDE)
- PyCharm
- PyDev in Eclipse
- PythonAnywhere
- Cloud9



# Github

- Git repository hosting service
- Offers
  - Revision control
  - Source code management
- THE place to share your code



# Python

# Python – The Language

- Friendly and easy to learn, pleasant
- Readable
- Open
- Free
- Runs everywhere
- Flexible
- Fast
- Powerful; Modules for everything! (<https://pypi.python.org/pypi>)



# Python Characteristics

- Multi-purpose; not only “scripting”
  - GUI
  - Web development
  - Apps
  - ..
- Interpreted language .. actually compiled byte-code is executed
- Object Oriented
- Strongly typed
- Widely used: <https://www.python.org/about/success/>



# Python Releases

- History
  - 1989: created by Guido Van Rossum
  - 1994: Python 1.0 released
  - ..
  - 2000: Python 2.0 released – latest 2.x release is 2.7; released 2010
  - ..
  - 2008: Python 3.0 released – broke backward compatibility
  - ..
  - 2015: Python 3.5 released
  - 2016: Python 3.6 released
  - 2018: Python 3.7 released
- New feature development only in 3.x
- Recommendation: **start with latest Python release (3.7)**



# Learning Python

- Beginner's Guide: <https://wiki.python.org/moin/BeginnersGuide>
- The Hitchhiker's Guide to Python:  
<http://python-guide-pt-br.readthedocs.io/en/latest/intro/learning/>
- Online Courses
  - <https://www.edx.org/course/subject/computer-science/python>
  - <https://www.coursera.org/specializations/python>
  - <https://www.codecademy.com/learn/python>
- Great Book: “Learning Python, 5<sup>th</sup> Edition”, Mark Lutz; PDF available online
- Start with fun stuff (Sudoku?, ..)
- **Code, Play, have fun!**

# Executing Python Code

- Interactive: simply start python from the CLI

```
~ jkrohn$ python3
Python 3.5.0 (v3.5.0:374f501f4567, Sep 12 2015, 11:00:19)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello world!')
Hello world!
```

- Run python file from the CLI

```
~ jkrohn$ python hello_world.py
Hello World!
```

- Demos: Interactive Jupyter Notebooks

- “The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text”

# Demo – Python Basics

```
>>>
>>> s = "hello"
>>> s
'hello'
>>> s = 1
>>> s
1
>>> dict = {'name':'Bob', 'age':35, 'sex':'male'}
>>> dict
{'age': 35, 'sex': 'male', 'name': 'Bob'}
>>> dict['age']
35
>>> import json
>>> print(json.dumps(dict, indent=4))
{
 "age": 35,
 "sex": "male",
 "name": "Bob"
}
>>> █
```

# Docker

# Docker

- “Open source container software platform that packages applications in containers”
- The live Notebooks run in a Docker container
- Docker needs to be installed on the local machine.
- Installation:
  - Mac: <https://www.docker.com/docker-mac>
  - Windows: <https://www.docker.com/docker-windows>
- Free Community Edition is sufficient to run the Notebook container



# Live Jupyter Demo Notebooks

- You can build your own Docker image with a Jupyter server and the session Notebooks: see Readme on GitHub
- .. or use the prepared image (easiest option, single command)  
`docker run -it --rm --name axl -p 8888:8888 jeokrohn/axl_workshop`
- Point your browser to <http://localhost:8888> to access the notebooks
- The password to access the server is: ‘axl’
- If port 8888 on your local machine is not available then use different port mappings (for example -p 8889:8888 to use local port 8889)
- Take a look at file `start.sh` on GitHub for an example start script

# Starting a Docker Container

- Prepared Scripts in GitHub repository:

- start.sh

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
jeokrohn/axl_workshop
```

- start\_mount\_local.sh:

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
--mount type=bind,src="$(pwd)",dst=/home/jovyan \
jeokrohn/axl_workshop
```

# Starting a Docker Container

- Prepared Scripts in GitHub repository:

- start.sh

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
jeokrohn/axl_workshop
```

- Run image
- Interactive
- Remove container on exit
- Set name of container
- Map local port 8890 to port 8888 of container

- start\_mount\_local.sh:

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
--mount type=bind,src="$(pwd)",dst=/home/jovyan \
jeokrohn/axl_workshop
```

# Starting a Docker Container

- Prepared Scripts in GitHub repository:

- start.sh

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
 jeokrohn/axl_workshop
```

Name of image to run

- start\_mount\_local.sh:

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
 --mount type=bind,src="$(pwd)",dst=/home/jovyan \
 jeokrohn/axl_workshop
```

# Starting a Docker Container

- Prepared Scripts in GitHub repository:

- start.sh

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
 jeokrohn/axl_workshop
```

- start\_mount\_local.sh:

```
#!/usr/bin/env bash
docker run -it --rm --name axl -p 8888:8888 \
 --mount type=bind,src="$(pwd)",dst=/home/jovyan \
 jeokrohn/axl_workshop
```

Map Docker host path  
to path in Docker container  
\$(pwd): inserts the current path; might need  
to replace with static path definition on other  
platforms

Time to Play!

# Time to play

- All lab content is available at:  
[https://github.com/jeokrohn/axl\\_workshop](https://github.com/jeokrohn/axl_workshop)
- Start by downloading “Lab guide.pdf” and following the steps described there



