ADEPTIA

## Adeptia Suite

# Raising SNMP Trap

## Using Process Flow

343 West Erie, Suite 440
Chicago, IL 60654, USA
Phone: (312) 229-1727 x111
Fax: (312) 229-1736

## Document Information

*Raising SNMP Trap using Process Flow*
Dec 2016

## Adeptia Support Information

For support queries, please contact us at support@adeptia.com.
Adeptia website URL: www.adeptia.com

## Copyright

## Trademarks

## Confidentiality

This document is the confidential and proprietary information of Adeptia. The information set forth herein represents the confidential and proprietary information of Adeptia. Such information shall only be used for the express purpose authorized by Adeptia and shall not be published, communicated, disclosed or divulged to any person, firm, corporation or legal entity, directly or indirectly, or to any third person without the prior written consent of Adeptia.

## Disclaimer

Adeptia, Inc. provides this publication "as is" without warranty of any kind, either express or implied. In no event shall Adeptia be liable for any loss of profits, loss of business, loss of use or data, interruption of business, indirect, special, punitive, incidental, or consequential damages of any kind.

This publication is subject to replacement by a later edition. To determine if a later edition exists, contact www.adeptia.com.

## Objective

This document describes how to send SNMP Trap from a Process Flow in Adeptia Suite.

## Before you start

- Extract **SNMP.zip**.
- Place **joesnmp-0.3.4.jar** in **<AdeptiaInstallFolder/AdeptiaServer/ServerKernel/ext** folder.
- Place **org** folder in **<AdeptiaInstallFolder/AdeptiaServer/ServerKernel/customClasses** folder.
- Restart Kernel and WebRunner.

## Raising SNMP Trap using Process flow

- SNMP Trap Sender works as an activity in the Process Flow.
- Currently, SNMP Trap Sender is not available as an inbuilt service in Adeptia Suite. You need to use custom plugin service to create the activity.
- In custom plugin service, write Java code to send SNMP Trap.
- In some cases, a database record is read using Database source. When the record is available as an XML, it can be set as a variable in Process Flow context using Context Upload service. This service takes XML record as an input, columns are read in custom plugin, and sent as SNMP Trap.

Following code segment can be used in the scripted service.

```
import org.apache.log4j.LogManager;
import org.apache.log4j.ext.JoeSNMPTrapSender;
import org.apache.log4j.ext.SNMPTrapAppender;
String ManagementHostIP="192.168.1.159";
String EnterpriseOID="1.3.6.1.4.1.13078";
String ApplicationTrapOID="1.3.6.1.4.1.13078.1.2.4";
SNMPTrapAppender app = new SNMPTrapAppender();
app.setManagementHost(ManagementHostIP);
app.setEnterpriseOID(EnterpriseOID);
app.setApplicationTrapOID(ApplicationTrapOID);
app.setForwardStackTraceWithTrap("true");
JoeSNMPTrapSender trapSender = new JoeSNMPTrapSender();
trapSender.initialize(app);
String column1 = (String)context.get("Name");
String column2 = (String)context.get("Age");
String column3 = (String)context.get("Address");
String FieldDelimiter =",";

String Message= column1+ FieldDelimiter + column2+ FieldDelimiter+ column3;

trapSender.addTrapMessageVariable(app.getApplicationTrapOID(),Message);

trapSender.sendTrap();
```

## Code segment

Import required java classes. You can import more classes, if required.

```
import org.apache.log4j.LogManager;
import org.apache.log4j.ext.JoeSNMPTrapSender;
import org.apache.log4j.ext.SNMPTrapAppender;
```

Define SNMP Appender configuration. You can give these parameters specific to your environment.

```
String ManagementHostIP="192.168.1.159";
```

IP address of the computer on which SNMP Manager is running

```
String EnterpriseOID="1.3.6.1.4.1.13078";
```

Unique ID for Enterprise

```
String ApplicationTrapOID="1.3.6.1.4.1.13078.1.2.4";
```

Unique ID for Trap Sender

Configure SNMP Appender, no need to modify this code segment.

```
SNMPTrapAppender app = new SNMPTrapAppender();
app.setManagementHost(ManagementHostIP);
app.setEnterpriseOID(EnterpriseOID);
app.setApplicationTrapOID(ApplicationTrapOID);
app.setForwardStackTraceWithTrap("true");
```

Create SNMP Trap Sender object, no need to modify this code segment.

```
JoeSNMPTrapSender trapSender = new JoeSNMPTrapSender();
```

Initialize SNMP Trap Sender object with SNMP Appender, no need to modify this code segment.

```
trapSender.initialize(app);
```

Retrieve data that is to be sent. It is assumed that all data variables have been set in Process Flow context. Read the variables that you set in the context.

Name, Age, and Address have been retrieved in the following example:

```
String column1 = (String)context.get("Name");
String column2 = (String)context.get("Age");
String column3 = (String)context.get("Address");
String FieldDelimiter =",";
String Message= column1+ FieldDelimiter + column2+ FieldDelimiter+ column3;
```

Add message that is to be sent through SNMP, no need to modify this segment.

```
trapSender.addTrapMessageVariable(app.getApplicationTrapOID(),Message);
```

Send SNMP Trap, no need to modify this segment.

```
trapSender.sendTrap();
```

Write the above code in custom plugin service, save it, and use it in the Process Flow.
In case, you want to send each column as a separate variable, then don’t make delimited message out of these columns as we did above. Send them as a separate column in the following way.

```
trapSender.addTrapMessageVariable(app.getApplicationTrapOID(),column1);

trapSender.addTrapMessageVariable(app.getApplicationTrapOID(),column2);

trapSender.addTrapMessageVariable(app.getApplicationTrapOID(),column3);
```