EventManager Service User's Guide

Version 1.2.7

Last updated: March 10, 2006

Overview

The EventManager service allows you to register with the Event service in order to execute STAF Commands. When an Event is generated, the Event Service (which normally sends messages to machines registered for specified events) will execute a STAF Command via the EventManager.

Note that EventManager registration information is persistent data. This means that if you register with the EventManager service, if you shutdown STAF and restart it (even if you reboot the machine), the prior registration information will still be active. When STAF starts, it reads in the previous EventManager registration information, and again registers each STAF command with the Event service.

Installation and Configuration

1. Install Java 1.2 or later.

2. Install STAF Version 2.6.0 or later (but less than Version 3.0.0) by following the installation instructions in the STAF documentation.

3. If you want to use an Event service that is already registered on this or another machine, proceed to the next step. Otherwise, install and configure the Event service to be used by the EventManager service on this machine or another machine, following the instructions in the Event Service User's Guide.

   Note: The version the Event service must be 1.10 or later (but less than 3.0.0).

4. Install the EventManager service by downloading the EventManagerV127.zip/tar file from Get Services for STAF V2 to a local directory (e.g. C:\STAF\services or /usr/local/staf/services) and unzip/untar file file.

5. Configure the EventManager service by adding the following statement to your STAF configuration file:

   ```
   SERVICE <Name> LIBRARY JSTAF EXECUTE <EventManager Jar File Name>
                  [OPTION <Name[=Value]>]...
                  [PARMS <"> [EVENTSERVICEMACHINE <EventServiceMachine>]
                             [EVENTSERVICENAME <EventServiceName>]
                         <">]
   ```

   where:

   - `<Name>` is the name by which the EventManager service will be known on this machine.

   - `<EventMananger Jar File Name>` is the fully qualified name of the STAFEventManager.jar file. On Windows

systems, this might be C:\STAF\services\STAFEventManager.jar. On Unix systems, this might be /usr/local/staf/services/STAFEventManager.jar

- `OPTION` specifes a configuration option that will be passed on to the JSTAF Java service proxy library. This is typically used by service proxy libraries to further control the interface to the actual service implementation. You may specify multiple `OPTIONs` for a given service. See the STAF User's Guide for more information on options for the JSTAF Java service proxy library.

- `<EventServiceMachine>` is the name of the Event service machine. It defaults to the EventManager service machine if not specified. This option will resolve STAF variables.

- `<EventServiceName>` is the name that the Event service is registered as on the Event service machine. It defaults to "Event" (not case-sensitive) if not specified. This option will resolve STAF variables.

Notes:

1. If the Event service machine is NOT the same as the EventManager service machine, then you must specify the `EVENTSERVICEMACHINE` parameter.

2. If "Event" is NOT the name used when registering the Event service, then you must specify the `EVENTSERVICENAME` parameter.

3. If the Event service machine is the same as the EventManager service machine, then the configuration statement for the Event service must be specified in the STAF configuration file BEFORE the EventManager service configuration statement.

Examples:

```
service EventManager LIBRARY JSTAF EXECUTE c:/staf/services/STAFEventManager.jar

service EventManager LIBRARY JSTAF EXECUTE /usr/local/staf/services/STAFEventManager.jar

service EventManager LIBRARY JSTAF EXECUTE {STAF/Config/STAFRoot}/services/STAFEventManager.jar \
                     PARMS "EVENTSERVICEMACHINE machA.austin.ibm.com"

service EventManager LIBRARY JSTAF EXECUTE {STAF/Config/STAFRoot}/services/STAFEventManager.jar \
                     PARMS "EVENTSERVICEMACHINE machA.austin.ibm.com EVENTSERVICENAME Event2"
```

Request Syntax

The Event Manager service provides the following requests:

- REGISTER - Registers a STAF command to be executed when a specified event is generated.
- UNREGISTER - Unregisters a STAF command for event execution
- LIST - Lists registered STAF commands
- HELP - Displays a list of requests for the EventManager service and how to use them.

REGISTER

REGISTER registers a STAF command to be executed when the specified event is generated.

Syntax

```
REGISTER   MACHINE <machine> | PYTHONMACHINE <machine>
           SERVICE <service> | PYTHONSERVICE <service>
           REQUEST <request> | PYTHONREQUEST <request>
           TYPE <eventType> [SUBTYPE <eventSubType>]
           [PREPARE <script>]
```

`MACHINE` specifies the name of the machine where the command will be executed. This option will resolve STAF variables. When the specified Event Type/Subtype is generated, the value of `MACHINE` will not be evaluated as a python string.

`PYTHONMACHINE` specifies the name of the machine where the command will be executed. This option will not resolve STAF variables. When the specified Event Type/Subtype is generated, the value of `MACHINE` will be evaluated as a python string.

`SERVICE` specifies the name of the service to be executed. This option will resolve STAF variables. When the specified Event Type/Subtype is generated, the value of `SERVICE` will not be evaluated as a python string.

`PYTHONSERVICE` specifies the name of the service to be executed. This option will not resolve STAF variables. When the specified Event Type/Subtype is generated, the value of `SERVICE` will be evaluated as a python string.

`REQUEST` specifies the request to be executed. This option will resolve STAF variables. When the specified Event Type/Subtype is generated, the value of `REQUEST` will not be evaluated as a python string.

`PYTHONREQUEST` specifies the request to be executed. This option will not resolve STAF variables. When the specified Event Type/Subtype is generated, the value of `RQUEST` will be evaluated as a python string.

`TYPE` specifies the Event Type for which this command will be registered. `TYPE` is not case sensitive. This option will resolve STAF variables.

`SUBTYPE` specifies the Event SubType for which this command will be registered. `SUBTYPE` is not case sensitive. This option will resolve STAF variables. If SUBTYPE is not specified, the request will be executed when any Event with the specified TYPE is generated.

`PREPARE` specifies Python code which will be executed when the registered event is generated. This code will be executed prior to the `PYTHONMACHINE`, `PYTHONSERVICE`, and `PYTHONREQUEST` options being evaluated as python strings. This option will not resolve STAF variables. If the Python code sets the variable STAFEventManagerSubmit to any string other than 'true', then the request will not be submitted.

Security

This request requires at least trust level 5.

Results

Upon successful return, the result buffer contains the EventManager ID.

Event Generation

When an event is generated with a matching type/subtype, the following Python variables will be available when the PREPARE, PYTHONMACHINE, PYTHONSERVICE, and PYTHONREQUEST options are evaluated as python strings:

- eventservice
- eventid
- generatingmachine
- generatingprocess
- generatinghandle
- eventtimestamp
- eventtype
- eventsubtype

In addition, each PROPERTY option name=value pair for the generated event will be set as Python variables.

Also, a Python dictionary named "eventinfo" will contain all of the above name/value pairs.

Examples

Goal: Execute the command "start command notepad" whenever an Event of Type a and Subtype b is generated.

```
eventmanager register machine local service process request "start command notepad" type a subtype b
```

Goal: Execute the command "start command xxx" whenever an Event of Type a and Subtype b is generated (where xxx is a python variable called mycmd, which is set in the Event service GENERATE request).

```
eventmanager register machine local service process pythonrequest "'start command %s' % (mycmd)" type
a subtype b

event generate type a subtype b property mycmd=notepad

event generate type a subtype b property mycmd=regedit
```

UNREGISTER

UNREGISTER unregisters a STAF command with the Event service.

Syntax

```
UNREGISTER ID <registrationID>
```

ID specifies the EventManager ID which is to be unregistered.

Security

This request requires at least trust level 3 if the UNREGISTER request has been submitted from the same machine that submitted the REGISTER request. A trust level of 4 or higher is required if the UNREGISTER request has been submitted from a different machine.

LIST

LIST lists information about the commands registered with the EventManager service.

Syntax

```
LIST [MACHINE <machine>] [TYPE <eventType>]
```

`MACHINE` specifies the machine for which EventManager requests should be listed.
`TYPE` specifies the Type for which EventManager requests should be listed.

Results

Upon successful return, this request will return the matching information for any registered EventManager requests, in the format:
eventManagerID;machineName;machineScriptFlag;serviceName;serviceScriptFlag;request;requestScriptFlag;type;subtype;prepare;

```
3;local;true;process;true;program;true;e;f;;
2;local;false;process;false;start command regedit;true;x;y;a=14;
1;local;true;process;true;start command notepad;true;a;b;;
```

VERSION

VERSION displays the EventManager service version.

Syntax

```
VERSION
```

Results

The result is the version number of the EventManager service.

HELP

HELP displays the request options and how to use them.

Syntax

```
HELP
```

Results

The result buffer contains the Help messages for the request options for the EventManager service.

EventManager Service Logging

The EventManager service maintains a machine log where it writes an entry when the following occurs:

- A REGISTER request is received
- An event has been generated that matches the type/subtype previously specified in a REGISTER request. The log entry will include the STAF request number for the submitted STAF command.
- The service receives notification that a submitted STAF command has completed. The log entry will include the STAF request number, as well as the RC and result from the STAF command.
- An UNREGISTER request is received

The logname for the EventManager service is the name under which the service is registered.

Here is an example of what the EventManager service log looks like:

```
20040201-16:35:32|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Register ID=1
originMachine=78ntmm3.austin.ibm.com machine=78ntmm3 pythonMachine=false
service=event pythonService=false request=version pythonRequest=false prepare= type=a
subtype=b
20040201-16:35:50|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Register ID=2
originMachine=78ntmm3.austin.ibm.com machine=78ntmm3 pythonMachine=false
service=eventpythonService=false request=blah pythonRequest=false prepare=x = 'hello'
type=a subtype=b
20040201-16:36:38|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Submitted request #112
ID=2 type=a subtype=b prepare=x = 'hello' eventservice=STAF/SERVICE/Event/
eventid=295764 generatingmachine=78ntmm3.austin.ibm.com generatingprocess=STAF/Client
generatinghandle=22 eventtimestamp=20040201-16:36:37 properties={} machine=78ntmm3
service=event request=blah
20040201-16:36:38|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Submitted request #116
ID=1 type=a subtype=b prepare= eventservice=STAF/SERVICE/Event/ eventid=295764
generatingmachine=78ntmm3.austin.ibm.com generatingprocess=STAF/Client
generatinghandle=22 eventtimestamp=20040201-16:36:37 properties={} machine=78ntmm3
service=event request=version
20040201-16:36:38|78ntmm3|9|STAF/SERVICE/eventmanager|Error|Request #112: RC=7,
Result=blah, Request submitted: machine=78ntmm3 service=event request=blah
20040201-16:36:38|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Request #116: RC=0,
Result=1.3.0, Request submitted: machine=78ntmm3 service=event request=version
20040201-16:37:35|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Submitted request #138
ID=2 type=a subtype=b prepare=x = 'hello' eventservice=STAF/SERVICE/Event/
eventid=295765 generatingmachine=78ntmm3.austin.ibm.com generatingprocess=STAF/Client
generatinghandle=25 eventtimestamp=20040201-16:37:35 properties={middle=michael,
last=bender, first=david} machine=78ntmm3 service=event request=blah
20040201-16:37:35|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Submitted request #142
ID=1 type=a subtype=b prepare= eventservice=STAF/SERVICE/Event/ eventid=295765
generatingmachine=78ntmm3.austin.ibm.com generatingprocess=STAF/Client
generatinghandle=25 eventtimestamp=20040201-16:37:35 properties={middle=michael,
last=bender, first=david} machine=78ntmm3 service=event request=version
20040201-16:37:35|78ntmm3|9|STAF/SERVICE/eventmanager|Error|Request #138: RC=7,
Result=blah, Request submitted: machine=78ntmm3 service=event request=blah
20040201-16:37:35|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Request #142: RC=0,
Result=1.3.0, Request submitted: machine=78ntmm3 service=event request=version
20040201-16:38:01|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Unregister ID=1
```

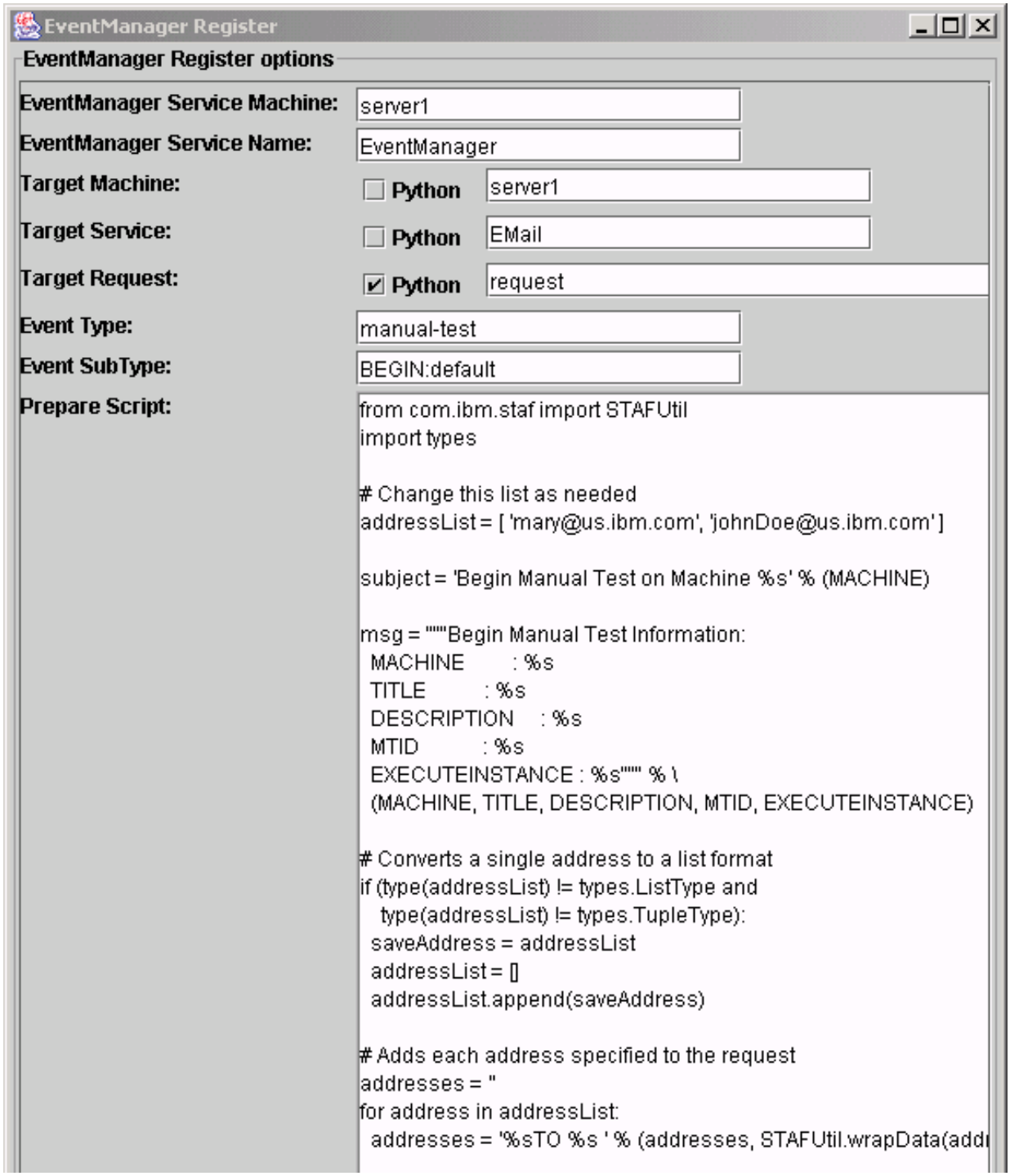
```
20040201-16:38:03|78ntmm3|9|STAF/SERVICE/eventmanager|Info|Unregister ID=2
```

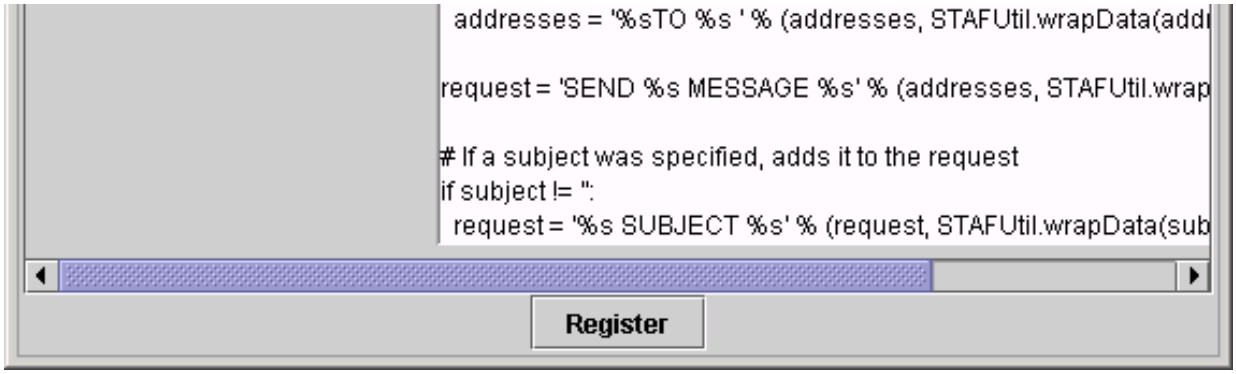
EventManager Registration GUI

The EventManager service provides a GUI to simplify the EventManager Registration. To use this GUI, after installing and configuring the EventManager service, from a command prompt enter the following command (note that to run this command, the STAFEventManager.jar file must be in your CLASSPATH):

```
java com.ibm.staf.service.eventmanager.EventManagerRegister
```

A GUI will be displayed which allows you to enter the EventManager Registration options.





The "EventManager Service Machine" default is the local machine. The "EventManager Service Name" default is "EventManager".

Fill in the required values for "Target Machine", "Target Service", and "Target Request". Optionally, you may specify Python code in the "Prepare Script" field.

Fill in the "Event Type" and "Event Subtype".

This example shows the options you would specify to set up EventManager to send an email (via the STAF Email service) when an event is generated. It also shows how you can reference the PROPERTY values that were specified when the event was generated (it uses them in the Prepare script).

Appendix A: Licenses and Acknowledgements

Jython

Jython is an implementation of the high-level, dynamic, object-oriented language Python written in 100% Pure Java, and seamlessly integrated with the Java platform. It thus allows you to run Python on any Java platform.

Acknowledgement

This product includes software developed by the Jython Developers (http://www.jython.org/).

Licence

```
Jython Software License
=======================

Copyright (c) 2000, Jython Developers
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions
are met:

 - Redistributions of source code must retain the above copyright
   notice, this list of conditions and the following disclaimer.
```

```
 - Redistributions in binary form must reproduce the above copyright
   notice, this list of conditions and the following disclaimer in
   the documentation and/or other materials provided with the distribution.

 - Neither the name of the Jython Developers nor the names of
   its contributors may be used to endorse or promote products
   derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```