system.util.sendMessage

This function is used in **Python Scripting.**

Description

This function sends a message to clients running under the Gateway, or to a project within the Gateway itself. To handle received messages, you must set up event script message handlers within a project. These message handlers run Jython code when a message is received. You can add message handlers under the "Message" section of the client/Gateway event script configuration dialogs.

Note that messages cannot be received within a Designer. However, messages can be sent within the Designer in a script (assuming that read/write comm is enabled).

Client Permission Restrictions

This scripting function has no Client Permission restrictions.

Syntax

system.util.sendMessage(project, messageHandler, payload, scope, clientSessionId, user, hasRole, hostName, remoteServers)

- Parameters

  String project - The name of the project containing the message handler.

  String messageHandler - The name of the message handler that will fire upon receiving a message.

  PyDictionary payload - Optional. A PyDictionary which will get passed to the message handler. Use "payload" in the message handler to access dictionary variables.

  String scope - Optional. Limits the scope of the message delivery to "C" (clients), "G" (Gateway), or "CG" for clients and the Gateway. Defaults to "C" if the user name, role or host name parameters are set, and to "CG" if none of these parameters are set.

  String clientSessionId - Optional. Limits the message delivery to a client with the specified session ID.

  String user - Optional. Limits the message delivery to clients where the specified user has logged in.

  String hasRole - Optional. Limits the message delivery to any client where the logged in user has the specified user role.

  String hostName - Optional. Limits the message delivery to the client that has the specified network host name.

  List remoteServers (since 7.8.2) - Optional. A list of Strings representing Gateway Server names. The message will be delivered to each server in the list. Upon delivery, the message is distributed to the local Gateway and clients as per the other parameters.

- Returns

  List - A List of Strings containing information about each system that was selected for delivery, where each List item is comma-delimited.

- Scope

  All

Code Examples

Code Snippet

```
# Simple message to both Client and Gateway handlers
project="X"
# It's important that both Gateway and Client versions of this message handler have been
created
messageHandler="myMessageHandler"
scope="CG"
myDict = {'first': "Hello", 'second': "World"}
results=system.util.sendMessage(project,messageHandler,myDict,scope)

# Assuming that there is one local client running project X, the results List will contain
these Strings:
type=Gateway,project=X,messageHandler=testHandler,filterParams={hostName=, clientSessionId=,
scope=CG, user=, hasRole=},sendStatus=SENT

type=Client,sessionId=65F7A472,clientAddress=127.0.0.1,clientHostName=127.0.0.1,project=X,
messageHandler=testHandler,filterParams={hostName=, clientSessionId=, scope=CG, user=,
hasRole=},sendStatus=SENT
```

Code Snippet

```
# Message to client handlers only where a specified user is logged in)
system.util.sendMessage(project="X",messageHandler="myMessageHandler",scope="C",user="Bob")
```

Code Snippet

```
# Message to remote servers over the Gateway Network (since 7.8.2)
servers = ["agent-8088", "agent-9000"]
system.util.sendMessage(project="X",messageHandler="myMessageHandler",remoteServers=servers)
```