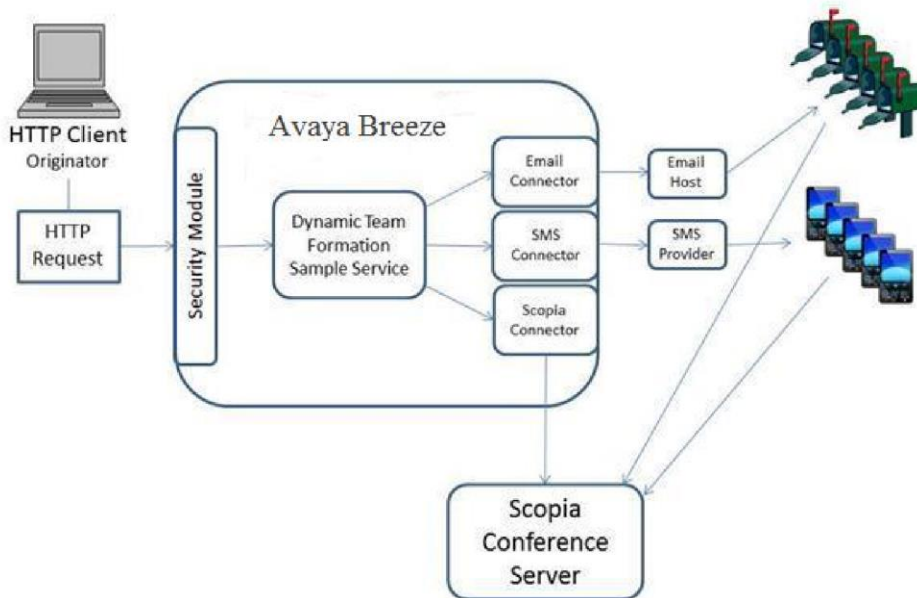


Dynamic Team Formation Sample Service

Introduction

The Dynamic Team Formation sample service (DTFSS) is a web service invoked by an HTTP client request that creates a Scopia video conferences and sends conference invite notification messages to recipients via email and SMS. The DTFSS is not intended to be added to a service profile like the Hello World and White List call intercepting sample services. This is true for all HTTP only invoked services.

Dynamic Team Formation Sample Service Diagram



Overview

The DTFSS is invoked when an HTTP request directed to the DTFSS is received by the Avaya Breeze® platform on the Security Module interface. After verifying the data in the HTTP request, the DTFSS retrieves its previously configured Service Global attributes from the Avaya Aura® System Manager and sends the conference notification messages to the intended recipients utilizing the Email and/or SMS Avaya Breeze APIs. Email and SMS listeners are then used to process responses as the DTFSS receives them.

Postman collection and environment files to be used with a web browser are included with the DTFSS to simplify and simulate generation of appropriate HTTP requests directed at the DTFSS.

Concepts Demonstrated

- Reception of inbound HTTP requests destined for the service
- Reading data from the service's Service Global attributes
- Creating a conference on Scopia and utilizing the returned conference URL
- Using the Email and SMS Avaya Breeze APIs to generate message requests for lists of recipients

Detailed description

ATTRIBUTES

The DTFSS defines multiple attributes at the Service Global level in the Avaya Aura® System Manager UI.

The screenshot shows the Avaya Aura System Manager UI. The left sidebar contains a navigation menu with the following items: Engagement Development Platform, Server Administration, Cluster Administration, Service Management, Configuration (selected), Service Profiles, Attributes (selected), Logging, Avaya Aura® Media Server, Event Catalog, HTTP Security, Implicit User Profiles, JDBC Providers, JDBC Sources, Service Ports, and System Tools. The main content area is titled 'Attributes Configuration' and includes a 'Commit' button and a 'Cancel' button. Below the title, there is a text box explaining that when a service is first installed, the factory default value is picked by the service writer, and it can be overridden using the Service Globals tab. The 'Service Globals' tab is selected, and the 'Service' dropdown is set to 'DynamicTeamFormationService'. A table titled '6 Items' shows the following attributes:

Name	Override Default	Effective Value	Description
Conference Maximum Duration (Hours)	<input type="checkbox"/>	2	String used to se
Email 'From' Address	<input type="checkbox"/>	DynamicTeamFormation@domain.com	String used in th
Email 'Reply-To' Address	<input type="checkbox"/>	DynamicTeamFormation@domain.com	String used for r
HTTP Pattern	<input type="checkbox"/>	https://myEDP.example.com/services/DynamicTeamFormationService/DynamicTeamFormationServlet	HTTP pattern use
SMS 'From' Number	<input type="checkbox"/>	5555555	String used in th
Supplier Id	<input type="checkbox"/>	10000000	Avaya provided

Conference announcement attribute:

- Conference Maximum Duration (Hours) – When the DTFSS creates a Scopia conference, this is the number of hours that the conference will be valid.

Email attributes:

- Email 'From' Address – Email recipients will see this value as the 'From' address.
- Email 'ReplyTo' Address – Email recipients will see this value as the 'Reply-To' address.

HTTP attribute:

- HTTP Pattern – Informational-only attribute that shows the URL used to invoke the DTFSS service.

SMS attribute:

- SMS 'From' Number – SMS recipients will see this attribute as the 'From' number.

Other attribute:

- Supplier Id – Read-only value of Avaya's Supplier ID.

DTFSS attributes are declared in the `properties.xml` file

`dynamicteamformation-svar/src/main/resources/properties.xml`. The following snippet of the `properties.xml` file defines the `emailFrom` attribute:

```
<attribute name="emailFrom">
  <displayName>Email 'From' Address</displayName>
  <helpInfo>String used in the 'From' header of the conference notification email.</helpInfo>

  <global>false</global>
  <validation name="anyString">
    <type>STRING</type>
  </validation>
  <admin_visible>true</admin_visible>
  <factory>
    <value>DynamicTeamFormation@domain.com</value>
    <user_changeable>true</user_changeable>
  </factory>
</attribute>
```

JAVA SOURCE

`com.avaya.zephyr.services.dynamicteamformation`

The Java source of the entry point for the DTFSS is in the main HTTP servlet,

`DynamicTeamFormationServlet`, contained in `dynamicteamformation-war` in the main package `com.avaya.zephyr.services.dynamicteamformation`.

The `DynamicTeamFormationServlet` is responsible for obtaining the service data for DTFSS attributes; invoking the conference scheduler, email sender, and SMS sender; and communicating the success or failure of the queuing of these invocations in the HTTP response. The rest of the source is contained in the `conference`, `email`, and `sms` sub-packages. The `conference` package contains a conference scheduler while the `email` and `sms` sub-packages contain sender and listener classes.

`com.avaya.zephyr.services.dynamicteamformation.conference`

The `com.avaya.zephyr.services.dynamicteamformation.confernece` package contains the conference scheduler `DynamicTeamFormationConferenceScheduler`.

The `DynamicTeamFormationConferenceScheduler` does not verify any key-value pairs sent in the HTTP request to DTFSS, but does pass the key-value pairs, `conferenceSubject` and `conferenceDescription`, on to the Scopia connector API. The DTFSS sets the duration of the conference to the value obtained from the DTFSS service global attributes 'Conference Duration'. The `DynamicTeamFormationConferenceScheduler` schedules the conference.

com.avaya.zephyr.services.dynamicteamformation.email

The `com.avaya.zephyr.services.dynamicteamformation.email` package contains the email sender `DynamicTeamFormationEmailSender`, and email listener `DynamicTeamFormationEmailListener`.

The `DynamicTeamFormationEmailSender` verifies that the required key-value pairs, `emailTo`, `emailSubject`, and `emailBody`, sent in the HTTP request contain data needed to send an email to a list of recipients. All fields must be present and contain data in the HTTP request, or an error string will be returned for email requests in the HTTP response. Each `emailTo` field may contain one, or a comma delimited list of email addresses. The DTFSS adds the 'From' and 'ReplyTo' email addresses to the email request if they exist in the DTFSS service global attributes. If the 'From' address does not exist, the email request is passed to the email connector in hopes that a 'From' email address is administered in its service global attribute. The DTFSS creates one email request for each HTTP request it receives containing all the email contacts that it sends to the email connector.

The `DynamicTeamFormationEmailListener` only logs statements and events when receiving responses about the email from the email connector.

com.avaya.zephyr.services.dynamicteamformation.sms

The `com.avaya.zephyr.services.dynamicteamformation.sms` package contains the SMS sender `DynamicTeamFormationSmsSender`, and SMS listener `DynamicTeamFormationSmsListener`.

The `DynamicTeamFormationSmsSender` verifies that the required key-value pairs, `smsTo`, and `smsBody`, sent in the HTTP request contain data needed to send an email to a list of recipients. Both fields must be present and contain data in the HTTP request, or an error string will be returned for SMS requests in the HTTP response. Each `smsTo` field may contain one, or a comma delimited list of SMS numbers. The DTFSS creates one SMS request for each HTTP request it receives containing all the SMS contacts that it sends to the SMS connector.

The `DynamicTeamFormationSmsListener` only logs statements and events when receiving responses about the SMS from the SMS connector.

Installation and Configuration

DTFSS

The DTFSS is included in the SDK in the SDK zip file at `/samples/DynamicTeamFormation`. Change directory to where the DTFSS resides and compile it (`mvn clean install`). Then load and install the svar on System Manager and administer the Service Global attributes.

POSTMAN (HTTP REQUEST GENERATOR)

In order to demonstrate an incoming HTTP request to Avaya Breeze, we have chosen to include Postman Collection and Postman Environment setting files in the DTFSS (`/samples/DynamicTeamFormation/DynamicTeamFormationCollection`). We suggest installing Google Chrome and then installing the 'Postman – REST Client' from the Chrome Web Store. It is free and fairly simple to use once you get the hang of it. Follow the Postman video tutorial in the Web Store and install and configure the Postman Collection and Postman Environment.

The purpose of the Postman Environment is to create default HTTP request URL encoded parameters (default domains, host addresses, etc.) that can be substituted in the Postman Collection. You do not have to use the Postman Environment, but it makes it easier for generating and testing large numbers of voice announcement, email, and SMS recipients. In the Postman Environment, set the `host` field to the IP address of the Security Module on Avaya Breeze you will be sending your HTTP request to. Do not enter the FQDN or IP address of the Network Management interface of Avaya Breeze! The `servicename` and `servicepath` should be left to their defaults. Change the `toemaildomain` field to its desired value.

In order to use the Postman Environment defaults (i.e. `host`, `servicename`, etc.) in the Postman Collection use double braces surrounding the field name. (i.e.

`{{host}}/services/{{servicename}}/{{servicepath}}`, `email@{{toemaildomain}}`, etc.).

When editing/viewing the Postman Collection, please use the “x-www-form-urlencoded” button to view the data sent in the HTTP request. Leave the line containing the URL to

`{{host}}/services/{{servicename}}/{{servicepath}}`. These are the values used from the Postman Environment. The Postman Collection is defaulted to schedule a conference and send an email to one recipient and send an SMS to one recipient. In order to send email, or SMS to multiple contacts you can either add a new row of data with `emailTo`, or `smsTo` in the `Key` field; or append multiple recipients to the existing `emailTo`, or `smsTo` key-value pairs separating them with a comma.

Example: `“emailTo/email1@domain.com”`
 `“emailTo/email2@domain.com”`
 OR
 `“emailTo/email1@domain.com, email2@domain.com”`

If you do not wish to send an email, delete all the key-value pairs that begin with `email`. The same applies for SMS.

Testing the service

Assuming three email boxes (email1@emaildomain.com, email2@emaildomain.com, and email3@emaildomain.com), and one SMS enabled phone (789):

- Configure Postman to generate an HTTP request for the above recipients.

The image shows two screenshots from the Postman application. The top screenshot is the 'Manage environments' dialog box. It has a title bar 'Manage environments' with a close button. Below the title is a section 'DynamicTeamFormation'. It contains a table with the following entries:

Key	Value
host	10.129.144.33
servicename	DynamicTeamFormationService
servicepath	DynamicTeamFormationServlet
toemaildomain	example.com

At the bottom of the dialog are 'Submit' and 'Back' buttons.

The bottom screenshot is the Postman interface. On the left is a sidebar with 'POSTMAN' logo, 'History', and 'Collections' tabs. Below 'Collections' is a collection named 'DynamicTeamFormation' with a 'POST' request named 'DynamicTeamFormation'. The main area shows the configuration for this request. At the top, there are tabs for 'Normal', 'Basic Auth', 'Digest Auth', and 'OAuth 1.0'. The 'DynamicTeamFormationEnvironment' is selected. Below the tabs is the request configuration for 'DynamicTeamFormation'. The URL is set to `{{host}}/services/{{servicename}}/{{servicepath}}` and the method is 'POST'. There are buttons for 'URL params' and 'Headers (0)'. Below the URL bar are tabs for 'form-data', 'x-www-form-urlencoded', and 'raw'. The 'form-data' tab is selected. It shows a list of key-value pairs for the request body:

emailTo	email1@{{toemaildomain}}
emailTo	email2@{{toemaildomain}}, email3@{{toemaildomain}}
emailSubject	DTF Email Subject
emailBody	DTF Email Body
conferenceSubject	DTF Conference Subject
conferenceDescription	DTF Conference Description
smsTo	5555555
smsBody	DTF SMS Body

At the bottom of the form data list are 'Send', 'Save', 'Preview', and 'Add to collection' buttons. On the far right of the interface is a 'Reset' button.

- Send the HTTP request from Postman.
- In Postman, verify that the HTTP response from Avaya Breeze resembles the following:
 - o Info: Email queued to be sent to email1@emaildomain.com
 - o Info: Email queued to be sent to email2@emaildomain.com
 - o Info: Email queued to be sent to email3@emaildomain.com
 - o Info: SMS queued to be sent to 789
- Verify the conference was created.
- Verify the three emails were received.
- Verify the SMS message was received.

Troubleshooting

PROBLEM: Receive “503 Service Temporarily Unavailable” HTTP response in Postman when sending a DTFSS request.

ACTION: Verify the Avaya Breeze is in the “Accepting” System State on Avaya Aura® System Manager. This is found on the `Elements->Avaya Breeze->Server Administration` page.

PROBLEM: Receive “404 Not Found” HTTP response in Postman when sending a DTFSS request.

ACTION: Verify that the service is installed on the Avaya Breeze. If the DTFSS has recently been installed, you may need to wait a few minutes before the service can be invoked.