aculabcloud
A true cloud telecoms platform

Writing a conference service

How to create a conference service with Aculab Cloud

Writing a conference service with Aculab Cloud

Aculab Cloud provides a simple, easy to learn API for rapid development of telephony applications.

Using the features provided by Aculab Cloud, you can rapidly develop a conferencing solution that could be used to provide a feature-rich conferencing service, with features such as:

- Conference announcements
- Conference recording (for playback after the event)
- Multiple, country specific access numbers

In the following code example, we show how easy it is to create a hosted conference solution using Aculab Cloud's RESTful API. The conference service has the following features:

- Access to ad-hoc conference rooms via unique conference codes
- Leader PIN control for starting a conference

Before you start

Working with Aculab Cloud

The structure of this application

For this code walkthrough, we will use the PHP wrapper for Aculab Cloud's REST API. The code used for this walkthrough can be downloaded from the Aculab Cloud website.

Aculab Cloud applications written using the RESTful API consist of one or more web pages, hosted on your web server. Each web page describes a list of actions Aculab Cloud should perform. Aculab Cloud makes requests for these pages, when asked to run your telephony application.

An action could be something such as playing a message, or requesting user input, or adding a caller to a conference.

The code for this solution is provided in a single file, Simple_conference.php. When making requests for this web page, a 'page' URL parameter should also be passed. The value of this parameter determines the actions Aculab Cloud is asked to perform.

http://your_web_server/Simple_conference.php?page=GetConferenceCode

Once Aculab Cloud has completed the first list of actions provided, it sends another request to the web server for another list of actions.

The page Aculab Cloud requests may differ, depending on the result of an action – for example, pressing different keys during a RunMenu action will result in different pages being requested.

Before you start

Conference service web page flow

Web page flow

Below is a diagram that shows the list of possible values for the ‘page’ parameter, and the order in which they will be requested by Aculab Cloud.

Subsequent sections of this document describe the construction of the web page, and walk through the actions performed when each ‘page’ is requested.

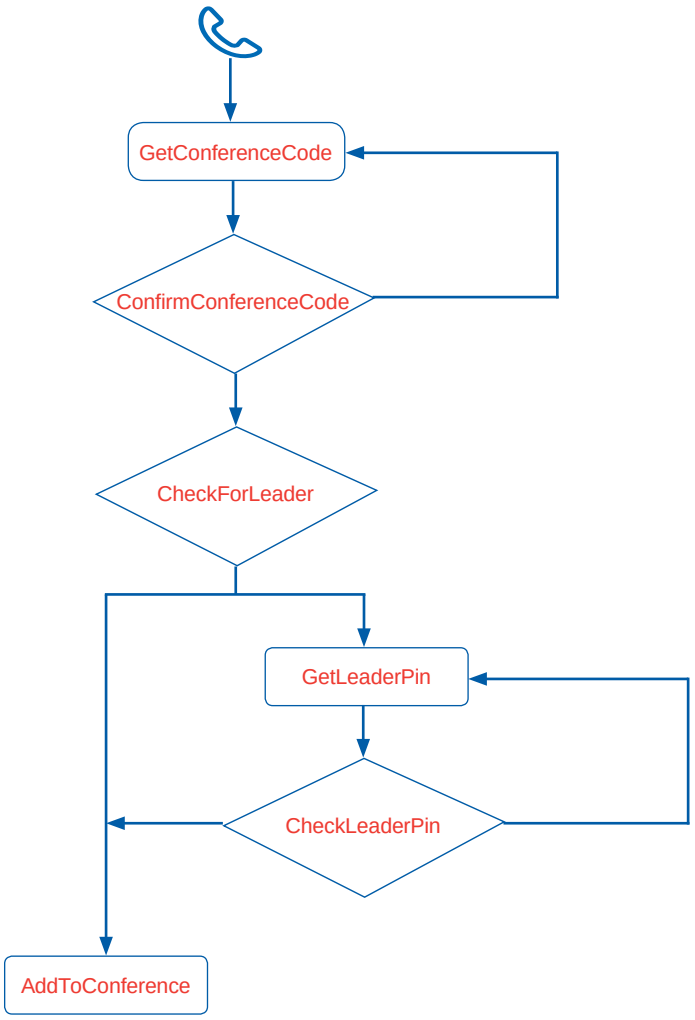

Figure 1: Conference service web page flow

The structure of the web page

Understanding the stucture of the web page

Understanding the structure of the web page

As mentioned earlier, the code for this conference application is all within one web page, and a ‘page’ parameter is used to determine which section of code on the website is executed when a request for the page is made.

The structure of the web page looks like this:

```
[Common Code]

if(isset($_GET['page']))
{

        if ($_GET['page'] == 'first_parameter')
        {

                [first_parameter] specific code


        }
        else if ($_GET['page'] == 'second_parameter')

        {

                [second_parameter] specific code

        }
}

[Common Code]
```

Figure 2: The stucture of the web page

First, we will show you the structure of the common code blocks, which are executed on each page request.

In subsequent sections, we will go through the parameter specific code blocks, which are executed depending on the value of the ‘page’ URL parameter.

A full code listing can be found in the appendix.

Common code

The structure of the common code

The structure of the common code

The structure of the common code is shown below.

```
<?php
declare(encoding='UTF-8');
spl_autoload_register();

header("Content-Type: application/json; charset=UTF-8");
// set headers to prevent the page being cached by intermediaries
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Sat, 26 Jul 1997 05:00:00 GMT"); // Date in the past


use \Aculab\TelephonyRestAPI\InstanceInfo;
use \Aculab\TelephonyRestAPI\Actions;
use \Aculab\TelephonyRestAPI\Play;
use \Aculab\TelephonyRestAPI\GetNumber;
use \Aculab\TelephonyRestAPI\ConferenceParticipantConfiguration;
use \Aculab\TelephonyRestAPI\ConnectToConference;
use \Aculab\TelephonyRestAPI\RunMenu;
use \Aculab\TelephonyRestAPI\MessageList;
use \Aculab\TelephonyRestAPI\Redirect;
use \Aculab\TelephonyRestAPI\HangUp;

$info = InstanceInfo::getInstanceInfo();
$my_actions = new Actions();
$token = null;
$token_as_json = array();

##########################################################################
# key value pairs used for storing conference code and leader PIN
#
# NOTE: In a real system this information would be stored in a database
##########################################################################
$CONF_CODE_ARRAY = array(
                            "123456" => array('leader_pin'=>"11111"),
                             "654321" => array('leader_pin'=>"22222"),
                            );

 if ($info != null)
{
        // Get token and decode to JSON if provided
        $token = $info->getToken();
        if($token != null)
        {
                $token_as_json = json_decode($token, true);
        }
        else
        {
                // We hit this the first time we call in
                $token_as_json["leader_pin_retries"] = 0;
        }


        // Retrieve call info and action results
        $callInfo = $info->getThisCallInfo();
        $result = $info->getActionResult();
}
```

Figure 3: Common code

Writing a conference service

How to create a conference service with Aculab Cloud

The first thing to be done is set the character encoding to Unicode, which allows for international characters in any text. The line spl_autoload_register() is used to auto-load any classes defined within the included Aculab classes.

```
declare(encoding='UTF-8');
spl_autoload_register();
```

Next, set the following HTTP headers so the page requests will not be cached by intermediate servers.

```
header("Content-Type: application/json; charset=UTF-8");
// set headers to prevent the page being cached by intermediaries
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Sat, 26 Jul 1997 05:00:00 GMT"); // Date in the past
```

Next, import the Aculab classes you are going to use within your web page.

```
use \Aculab\TelephonyRestAPI\InstanceInfo;
use \Aculab\TelephonyRestAPI\Actions;
use \Aculab\TelephonyRestAPI\Play;
use \Aculab\TelephonyRestAPI\GetNumber;
use \Aculab\TelephonyRestAPI\ConferenceParticipantConfiguration;
use \Aculab\TelephonyRestAPI\ConnectToConference;
use \Aculab\TelephonyRestAPI\RunMenu;
use \Aculab\TelephonyRestAPI\MessageList;
use \Aculab\TelephonyRestAPI\Redirect;
use \Aculab\TelephonyRestAPI\HangUp;
```

Each request from Aculab Cloud includes an instance info object. This Instance info object contains the following useful information:

Information about the call (callinfo)
: Such as called and calling numbers associated with this call.

Results of actions that have occurred (actionResult)
: For example, if you requested the user to enter some digits, what those digits were.

User-defined token (token)
: A string which is sent back and forth between Aculab Cloud and your web server, in which you can pass your own application specific information between web pages.

To enable the website to access this useful information, retrieve the instance info from the request received from Aculab Cloud.

```
$info = InstanceInfo::getInstanceInfo();
```

Writing a conference service

How to create a conference service with Aculab Cloud

Then pull out the callinfo, actionResult and user-defined token, for use later.

```
// Get token and decode to JSON if provided
$token = $info->getToken();
if($token != null)
{
        $token_as_json = json_decode($token, true);
}
else
{
        // We hit this the first time we call in
        $token_as_json["leader_pin_retries"] = 0;
}


// Retrieve call info and action results
$callInfo = $info->getThisCallInfo();
$result = $info->getActionResult();
```

Finally, before calling the unique code determined by the value of the 'page' parameter, create a new Action array, to which you add the actions you wish Aculab Cloud to perform.

```
$my_actions = new Actions();
```

After calling the 'page' parameter specific code, encode the user-defined token as a JSON string and associate it with your actions list.

```
$token = json_encode( $token_as_json);
```

Finally, print the actions to standard output. This causes an HTTP response to be created, and sends the list of actions to Aculab Cloud.

```
// Send actions
print $my_actions;
```

Writing a conference service

How to create a conference service with Aculab Cloud

GetConferenceCode

Page Structure

The first 'page' parameter passed with the request from Aculab Cloud is GetConferenceCode. This code asks the caller to enter their conference code, and requires an input via a telephone keypad.

```
// Ask caller to enter conference code
$prompt = Play::sayText( "Please enter your conference code followed by the pound or hash sign" );
$getConferenceCode = new GetNumber($prompt);
$getConferenceCode->setEndDigit('#');
$getConferenceCode->setValidDigits('1234567890');
$getConferenceCode->setNextPage('Simple_Conference.php?page=ConfirmConferenceCode');
$my_actions->add($getConferenceCode);
```

Figure 4: GetConferenceCode

Page Components

In this code segment, the GetNumber action is sent to Aculab Cloud to ask the user to enter the code of the conference they wish to join. The user is asked to press the hash ('#') key once they have entered all the digits of the conference code.

```
// Ask caller to enter conference code
$prompt = Play::sayText( "Please enter your conference code followed by the pound or hash
sign" );
$getConferenceCode = new GetNumber($prompt);
$getConferenceCode->setEndDigit('#');
$getConferenceCode->setValidDigits('1234567890');
```

The digits entered by the caller are sent to your web server by Aculab Cloud, and the page to which they are sent is used to confirm the caller has entered a valid conference code.

```
$getConferenceCode->setNextPage('Simple_Conference.php?page=ConfirmConferenceCode');
$my_actions->add($getConferenceCode);
```

Writing a conference service

How to create a conference service with Aculab Cloud

ConfirmConferenceCode

Page Structure

In this code segment, the conference code previously entered by the caller is checked. If correct, the next page asks the caller whether they are the conference leader. If incorrect, the next page asks again for their conference code.

```
// Retrieve conference code from action result
$conference_code = $result->getEnteredNumber();


// Check whether the conference code is valid
if ( isValidConferenceCode($conference_code) )
{
        // Store the conference code in the user-defined token
        $token_as_json["conference_code"] = $conference_code;
        // Mark caller as NOT leader in user-defined token
        $token_as_json["isLeader"] = false;
        // Now check whether the caller is the conference leader
        $my_actions->add(new Redirect( 'Simple_Conference.php?page=CheckForLeader') );
}
else
{
        // Invalid conference code.  Please try again
        $my_actions->add( Play::sayText("<say-as interpret-
        as='digits'>".$conference_code."</say-as> Is not a valid conference code.") );
        $my_actions->add(new Redirect( 'Simple_Conference.php?page=GetConferenceCode') );
}
```

Figure 5: ConfirmConferenceCode

Page Components

Aculab Cloud sends the digits entered by the caller to this page, and they can be retrieved from the action result.

```
// Retrieve conference code from action result
$conference_code = $result->getEnteredNumber();
```

The conference code is checked against a list of valid codes. In this application, the codes are stored in a global array. However in a real life application, the conference codes would more likely be stored in a database.

```
// Check whether the conference code is valid
if ( isValidConferenceCode($conference_code) )
```

Writing a conference service

How to create a conference service with Aculab Cloud

If a valid conference code is entered, store it in the caller's user-defined token.

```
// Store the conference code in the user-defined token
$token_as_json["conference_code"] = $conference_code;
```

Next ask the caller if they are the conference leader, and if yes, ask them for their leader PIN..

```
// Mark caller as NOT leader in user-defined token
$token_as_json["isLeader"] = false;
// Now check whether the caller is the conference leader
$my_actions->add(new Redirect( 'Simple_Conference.php?page=CheckForLeader') );
```

If the conference code entered is invalid, tell the caller, and ask them to re-enter the conference code.

```
// Invalid conference code.  Please try again
$my_actions->add( Play::sayText("<say-as interpret-
as='digits'>".$conference_code."</say-as> Is not a valid conference code."));
$my_actions->add(new Redirect( 'Simple_Conference.php?page=GetConferenceCode'))
```

Writing a conference service

How to create a conference service with Aculab Cloud

CheckforLeader

Page Structure

In this code segment, the caller is asked to press star ('*') if they are the leader. If they are the leader, a leader PIN is requested, otherwise the caller is added to the requested conference.

```
// Ask the caller whether they are the conference leader
$mp = Play::sayText( "If you are the leader, press star now.");
$menu = new RunMenu($mp);
$menu->setHelpDigit('#');
$menu->addMenuOption('*', 'Simple_Conference.php?page=GetLeaderPin');
$menu->setOnDigitTimeoutMessages(new MessageList());
$menu->setOnInvalidDigitMessages(new MessageList());
$my_actions->add($menu);


// We fall here on timeout or invalid key press
$my_actions->add(new Redirect( 'Simple_Conference.php?page=AddToConference') );
```

Figure 6: CheckforLeader

Page Components

A RunMenu action is used to ask the caller whether they are the leader of the conference. If the caller pressed star ('*') to confirm they are the leader, the page to ask for the leader PIN will be requested.

```
// Ask the caller whether they are the conference leader
$mp = Play::sayText( "If you are the leader, press star now.");
$menu = new RunMenu($mp);
$menu->setHelpDigit('#');
$menu->addMenuOption('*', 'Simple_Conference.php?page=GetLeaderPin');
```

A MessageList is used to define prompts which will be played when no digits are pressed by the caller, or when a caller presses a digit not associated with a menu option.

Set the Digit timeout message list and invalid digit message list to an empty MessageList so no prompts are played when a digit timeout occurs, or an invalid digit is pressed.

```
$menu->setOnDigitTimeoutMessages(new MessageList());
$menu->setOnInvalidDigitMessages(new MessageList());
$my_actions->add($menu);
```

So when a caller presses any digit other than star ('*'), or presses no digit, they will be added to the conference as a participant.

```
// We fall here on timeout or invalid key press
$my_actions->add(new Redirect( 'Simple_Conference.php?page=AddToConference') );
```

Writing a conference service

How to create a conference service with Aculab Cloud

GetLeaderPin

Page Structure

The code section is called if the caller pressed star ('*') to confirm they wish to be the leader of the conference.

```
// Get leader PIN
$prompt = Play::sayText( "Please enter your leader pin followed by the pound or hash sign" );
$getLeaderPin = new GetNumber($prompt);
$getLeaderPin->setEndDigit('#');
$getLeaderPin->setValidDigits('1234567890');
$getLeaderPin->setNextPage('Simple_Conference.php?page=CheckLeaderPin');
$my_actions->add($getLeaderPin);
```

Figure 7: GetLeaderPin

Page Components

In this code segment, the GetNumber action is sent to Aculab Cloud to ask the user to enter the leader PIN for the conference they wish to join. The user is asked to press the hash ('#') key once they have entered all the digits of the leader PIN.

```
$getLeaderPin->setNextPage('Simple_Conference.php?page=CheckLeaderPin');
$my_actions->add($getLeaderPin);
```

The digits entered by the caller are sent to your web server by Aculab Cloud, and the page to which they are sent is used to confirm the caller has entered the correct leader PIN.

Writing a conference service

How to create a conference service with Aculab Cloud

CheckLeaderPin

Page Structure

In this code section, the leader PIN is checked to see whether it is valid. The caller gets three attempts to enter the PIN.

If the caller has entered an invalid leader PIN, the leader_pin_retries variable is incremented, and this value is stored in the user-defined token. The leader_pin_retries count will then be checked each time the leader PIN is validated, and the caller disconnected if they fail to enter a valid leader PIN after three attempts.

```
$MAX_PIN_RETRIES = 3;


// Retrieve the leader PIN from the action result
$leader_pin = $result->getEnteredNumber();


// Check to see whether the leader PIN is valid
if( isValidLeaderPin( $token_as_json['conference_code'], $leader_pin ) )
{
        // Mark the caller as the leader in the user-defined token
        $token_as_json["isLeader"] = true;
        // Now add the caller to the conference
        $my_actions->add( new Redirect( 'Simple_Conference.php?page=AddToConference' ) );
}
else
{
        // Retrieve and increment leader PIN retries
        $retries = $token_as_json["leader_pin_retries"];
        $retries = $retries + 1;
        $token_as_json["leader_pin_retries"] = $retries;
        if( $retries < $MAX_PIN_RETRIES )
        {
                if($retries < $MAX_PIN_RETRIES - 1 )
                {
                        $my_actions->add(Play::sayText("That leader pin is not valid.  Please try
                        again."));
                }
                else
                {
                        $my_actions->add(Play::sayText("That leader pin is not valid.  This is your last
                        attempt."));
                }
                        $my_actions->add( new Redirect('Simple_Conference.php?page=GetLeaderPin'));
                }
                else
                {
                        $my_actions->add(Play::sayText("That leader pin is not valid.  You have had
                        ".$MAX_PIN_RETRIES." attempts.  Please call in again.  Good bye.<break/>"));
                        $my_actions->add(new Hangup());
                }
}
```

Figure 7: CheckLeaderPin

Writing a conference service

How to create a conference service with Aculab Cloud

Page Components

Aculab Cloud sends the digits entered by the caller to this page, and they can be retrieved from the action result.

```
// Retrieve the leader PIN from the action result
$leader_pin = $result->getEnteredNumber();
```

The leader PIN is checked against the valid PIN for the conference code. In this application, the PINs are stored in a global array. However in a real life application, the PINs would probably be stored in a database.

```
// Check to see whether the leader PIN is valid
if( isValidLeaderPin( $token_as_json['conference_code'], $leader_pin ) )
```

If a valid leader PIN is entered, the caller is marked as the conference leader, and the page to add the caller to the conference is requested.

```
        // Mark the caller as the leader in the user-defined token
        $token_as_json["isLeader"] = true;
        // Now add the caller to the conference
        $my_actions->add( new Redirect( 'Simple_Conference.php?page=AddToConference' ) );
```

If an invalid leader PIN is entered, increment the 'leader_pin_retries' variable to reflect the number of failed PIN attempts.

```
        // Retrieve and increment leader PIN retries
        $retries = $token_as_json["leader_pin_retries"];
        $retries = $retries + 1;
        $token_as_json["leader_pin_retries"] = $retries;
```

If the caller has entered an invalid leader PIN less than three times, tell them the PIN was invalid, and allow them further attempts.

```
        if($retries < $MAX_PIN_RETRIES - 1 )
        {
                $my_actions->add(Play::sayText("That leader pin is not valid.  Please try
                again."));
        }
```

Writing a conference service

How to create a conference service with Aculab Cloud

Before the final attempt, let them know this is their final attempt at entering their leader PIN.

```
else
{
        $my_actions->add(Play::sayText("That leader pin is not valid.  This is your last
        attempt."));
}

$my_actions->add( new Redirect('Simple_Conference.php?page=GetLeaderPin'));
```

After three attempts at entering the leader PIN, tell the caller they have to call in again, and disconnect them from the conference service.

```
$my_actions->add(Play::sayText("That leader pin is not valid.  You have had
".$MAX_PIN_RETRIES." attempts.  Please call in again.  Good bye.<break/>"));
$my_actions->add(new Hangup());
```

Writing a conference service

How to create a conference service with Aculab Cloud

AddToConference

Page Structure

In this section of code, the caller is added to the required conference. If the caller is a participant, they will hear music-on-hold, until the leader has joined the conference.

```
// Get conference code from the user-defined token
$conference_code = $token_as_json["conference_code"];


// Tell the caller they are being added to the conference
$prompt = Play::sayText( "Adding you to conference <say-as interpret-
as='digits'>$conference_code</say-as>" );
$my_actions->add($prompt);


if( $token_as_json["isLeader"] )
{
        // Tell the caller they are the leader
        $my_actions->add(Play::sayText("As the leader"));
}
else
{
        // Tell the caller to wait for the leader to join
        $my_actions->add(Play::sayText("The conference will start when the leader joins"));
}


// Prepare conference room settings
$participant_config = new ConferenceParticipantConfiguration();
$participant_config->setTalker(true);
$participant_config->setBeepOnEntry(true);
$participant_config->setMuteDigit('#');
$participant_config->setUnmuteDigit('*');
$participant_config->setExitDigit('9');
$participant_config->setWhileStoppedMedia(Play::playFile('holdmusic.wav'));


if( $token_as_json["isLeader"] )
{
        $participant_config->setStartOnEntry(true);
        $participant_config->setDestroyOnExit(true);
}
else
{
        $participant_config->setStartOnEntry(false);
        $participant_config->setDestroyOnExit(false);
}


// Add the caller to the conference
$ctc = new ConnectToConference($conference_code, $participant_config);
$ctc->setSecondsAnswerTimeout(30);
$my_actions->add($ctc);
```

Figure 8: AddToConference

Writing a conference service

How to create a conference service with Aculab Cloud

Page Components

First, retrieve the conference code that you had stored in the user-defined token.

```
// Get conference code from the user-defined token
$conference_code = $token_as_json["conference_code"];
```

Tell the caller they are about to added to the conference.

```
// Tell the caller they are being added to the conference
$prompt = Play::sayText( "Adding you to conference <say-as interpret-
as='digits'>$conference_code</say-as>" );
$my_actions->add($prompt);
```

If they are the leader, tell them they are the leader, and if the are a participant, tell them the conference will not start until the leader has joined the conference.

```
if( $token_as_json["isLeader"] )
{
        // Tell the caller they are the leader
        $my_actions->add(Play::sayText("As the leader"));
}
else
{
        // Tell the caller to wait for the leader to join
        $my_actions->add(Play::sayText("The conference will start when the leader joins"));
}
```

Each caller added into the conference has settings, which define how they will be able to interact in the conference. These settings are defined in a caller's ConferenceParticipantConfiguration. These settings will be configured differently, depending on whether the caller is a participant or a conference leader.

All callers to the conference will be able to talk, so set each caller as a 'talker'.

Also, configure the participant settings so that a tone is played when the caller is initially added to the conference, as well as setting the keys, the user can press to mute/unmute their line, or leave the conference.

A message or music can be looped for the participants to hear while they are waiting for the leader to join the conference, and for the leader to hear while he is the only caller in the conference. In this example we are using a music file stored on your cloud account.

Writing a conference service

How to create a conference service with Aculab Cloud

```
// Prepare conference room settings
$participant_config = new ConferenceParticipantConfiguration();
$participant_config->setTalker(true);
$participant_config->setBeepOnEntry(true);
$participant_config->setMuteDigit('#');
$participant_config->setUnmuteDigit('*');
$participant_config->setExitDigit('9');
$participant_config->setWhileStoppedMedia(Play::playFile('holdmusic.wav'));
```

If the caller is the leader, specify that the conference should be started when they enter the conference, and finish when they leave the conference.

```
$participant_config->setStartOnEntry(true);
$participant_config->setDestroyOnExit(true);
```

If the caller is a participant then they need to wait for the leader of the conference to start and when they leave the conference will continue.

```
$participant_config->setStartOnEntry(false);
$participant_config->setDestroyOnExit(false);
```

Finally connect the caller to the conference.

```
// Add the caller to the conference
$ctc = new ConnectToConference($conference_code, $participant_config);
$ctc->setSecondsAnswerTimeout(30);
$my_actions->add($ctc);
```

That completes the walkthrough of all the code required to build a conference service. A full code listing for this conference application can be found in the appendix.

In this application note we have focused on PHP. However, Aculab Cloud offers a number of API wrappers for its REST API.

If you have any questions, you can email Aculab's support team support@aculab.com

Appendix

Full Code Listing

```
<?php
declare(encoding='UTF-8');
spl_autoload_register();

header("Content-Type: application/json; charset=UTF-8");
// set headers to prevent the page being cached by intermediaries
header("Cache-Control: no-cache, must-revalidate"); // HTTP/1.1
header("Expires: Sat, 26 Jul 1997 05:00:00 GMT"); // Date in the past


use \Aculab\TelephonyRestAPI\InstanceInfo;
use \Aculab\TelephonyRestAPI\Actions;
use \Aculab\TelephonyRestAPI\Play;
use \Aculab\TelephonyRestAPI\GetNumber;
use \Aculab\TelephonyRestAPI\ConferenceParticipantConfiguration;
use \Aculab\TelephonyRestAPI\ConnectToConference;
use \Aculab\TelephonyRestAPI\RunMenu;
use \Aculab\TelephonyRestAPI\MessageList;
use \Aculab\TelephonyRestAPI\Redirect;
use \Aculab\TelephonyRestAPI\HangUp;

$info = InstanceInfo::getInstanceInfo();
$my_actions = new Actions();
$token = null;
$token_as_json = array();

##########################################################################
# key value pairs used for storing conference code and leader PIN
#
# NOTE: In a real system this information would be stored in a database
##########################################################################
$CONF_CODE_ARRAY = array(
                            "123456" => array('leader_pin'=>"11111"),
                             "654321" => array('leader_pin'=>"22222"),
                            );

 if ($info != null)
{
        // Get token and decode to JSON if provided
        $token = $info->getToken();
        if($token != null)
        {
                $token_as_json = json_decode($token, true);
        }
        else
        {
                // We hit this the first time we call in
                $token_as_json["leader_pin_retries"] = 0;
        }


        // Retrieve call info and action results
        $callInfo = $info->getThisCallInfo();
        $result = $info->getActionResult();
}
if(isset($_GET['page']))
{
        error_log("Page is ".$_GET['page']."\ntoken\n".json_encode($token_as_json,
        JSON_PRETTY_PRINT) );
```

Appendix

Full Code Listing

```
if ($_GET['page'] == 'GetConferenceCode')
{


        // Ask caller to enter conference code
        $prompt = Play::sayText( "Please enter your conference code followed by the pound or
        hash sign" );
        $getConferenceCode = new GetNumber($prompt);
        $getConferenceCode->setEndDigit('#');
        $getConferenceCode->setValidDigits('1234567890');
        $getConferenceCode->setNextPage('Simple_Conference.php?page=ConfirmConferenceCode');
        $my_actions->add($getConferenceCode);
}
else if ($_GET['page'] == 'ConfirmConferenceCode')
{
        // Retrieve conference code from action result
        $conference_code = $result->getEnteredNumber();


        // Check whether the conference code is valid
        if ( isValidConferenceCode($conference_code) )
        {

                // Store the conference code in the user-defined token
                $token_as_json["conference_code"] = $conference_code;
                // Mark caller as NOT leader in user-defined token
                $token_as_json["isLeader"] = false;
                // Now check whether the caller is the conference leader
                $my_actions->add(new Redirect( 'Simple_Conference.php?page=CheckForLeader') );
        }
        else
        {
                // Invalid conference code.  Please try again
                $my_actions->add( Play::sayText("
                <say-as interpret-as='digits'>".$conference_code."</say-as> Is not a valid conference
                code.") );
                $my_actions->add(new Redirect( 'Simple_Conference.php?page=GetConferenceCode') );
        }
}
else if ($_GET['page'] == 'CheckForLeader')
{
        // Ask the caller whether they are the conference leader
        $mp = Play::sayText( "If you are the leader, press star now.");
        $menu = new RunMenu($mp);
        $menu->setHelpDigit('#');
        $menu->addMenuOption('*', 'Simple_Conference.php?page=GetLeaderPin');
        $menu->setOnDigitTimeoutMessages(new MessageList());
        $menu->setOnInvalidDigitMessages(new MessageList());
        $my_actions->add($menu);


        // We fall here on timeout or invalid key press
        $my_actions->add(new Redirect( 'Simple_Conference.php?page=AddToConference') );
}
else if ($_GET['page'] == 'GetLeaderPin')
{
        // Get leader PIN
        $prompt = Play::sayText( "Please enter your leader pin followed by the pound or hash sign" );
        $getLeaderPin = new GetNumber($prompt);
        $getLeaderPin->setEndDigit('#');
        $getLeaderPin->setValidDigits('1234567890');
        $getLeaderPin->setNextPage('Simple_Conference.php?page=CheckLeaderPin');
        $my_actions->add($getLeaderPin);
}
else if ($_GET['page'] == 'CheckLeaderPin')
{
```

Appendix

Full Code Listing

```
$MAX_PIN_RETRIES = 3;


// Retrieve the leader PIN from the action result
$leader_pin = $result->getEnteredNumber();


// Check to see whether the leader PIN is valid
if( isValidLeaderPin( $token_as_json['conference_code'], $leader_pin ) )
{


        // Mark the caller as the leader in the user-defined token
        $token_as_json["isLeader"] = true;
        // Now add the caller to the conference
        $my_actions->add( new Redirect( 'Simple_Conference.php?page=AddToConference' ) );
}
else
{


        // Retrieve and increment leader PIN retries
        $retries = $token_as_json["leader_pin_retries"];
        $retries = $retries + 1;
        $token_as_json["leader_pin_retries"] = $retries;
        if( $retries < $MAX_PIN_RETRIES )
        {
                if($retries < $MAX_PIN_RETRIES - 1 )
                {
                        $my_actions->add(Play::sayText("That leader pin is not valid.  Please try
                        again."));
                }
                else
                {
                        $my_actions->add(Play::sayText("That leader pin is not valid.  This is your last
                        attempt."));
                }
                        $my_actions->add( new Redirect('Simple_Conference.php?page=GetLeaderPin'));
                }
                else
                {
                        $my_actions->add(Play::sayText("That leader pin is not valid.  You have had
                        ".$MAX_PIN_RETRIES." attempts.  Please call in again.  Good bye.<break/>"));
                        $my_actions->add(new Hangup());
                }
        }
}
else if ($_GET['page'] == 'AddTo Conference')
{
        // Get conference code from the user-defined token
        $conference_code = $token_as_json["conference_code"];


        // Tell the caller they are being added to the conference
        $prompt = Play::sayText( "Adding you to conference <say-as interpret-
        as='digits'>$conference_code</say-as>" );
        $my_actions->add($prompt);


        if( $token_as_json["isLeader"] )
        {
                // Tell the caller they are the leader
                $my_actions->add(Play::sayText("As the leader"));
        }
```

Appendix

Full Code Listing

```
        else
        {
                // Tell the caller to wait for the leader to join
                $my_actions->add(Play::sayText("The conference will start when the leader joins"));
        }


        // Prepare conference room settings
        $participant_config = new ConferenceParticipantConfiguration();
        $participant_config->setTalker(true);
        $participant_config->setBeepOnEntry(true);
        $participant_config->setMuteDigit('#');
        $participant_config->setUnmuteDigit('*');
        $participant_config->setExitDigit('9');
        $participant_config->setWhileStoppedMedia(Play::playFile('holdmusic.wav'));


        if( $token_as_json["isLeader"] )
        {
                $participant_config->setStartOnEntry(true);
                $participant_config->setDestroyOnExit(true);
        }
        else
        {
                $participant_config->setStartOnEntry(false);
                $participant_config->setDestroyOnExit(false);
        }


        // Add the caller to the conference
        $ctc = new ConnectToConference($conference_code, $participant_config);
        $ctc->setSecondsAnswerTimeout(30);
        $my_actions->add($ctc);


        }
        else if ($_GET['page'] == 'FinalPage')
        {
                // Nothing to clear up on end of call
        }
        else if ($_GET['page'] == 'ErrorPage')
        {
                // Record the error string in webserver logging
                error_log("Error: ".$info->getErrorResult()->getResult());


                // Tell the user an error occurred
                $my_actions->add(Play::sayText("An error has occurred, please dial in again.<break />"));
        }
}
else
{
        // Return Error
}
// Convert JSON token to string
$token = json_encode( $token_as_json );
$my_actions->setToken( $token );


// Send actions
print $my_actions;
```

Appendix

Full Code Listing

```
/*
 * Lookup to see whether conference code is valid
 */
function isValidConferenceCode($conference_code)
{
        global $CONF_CODE_ARRAY;
        return array_key_exists($conference_code, $CONF_CODE_ARRAY);
}


/*
 * Lookup to see whether leader pin is valid
 */
function isValidLeaderPin( $conference_code, $leader_pin )
{
        global $CONF_CODE_ARRAY;
        return $CONF_CODE_ARRAY[$conference_code]['leader_pin'] == $leader_pin;
}
```

Aculab Cloud

Three decades of innovation - in the cloud

About Aculab

Aculab provides deployment proven telephony products to the global communications market

Whether you need telephony resources on a board, on a host server processor or from a cloud-based platform, Aculab ensures that you have the choice. We are an innovative, market leading company that places product quality and support right at the top of our agenda. With over 35 years of experience in helping to drive our customers' success, our technology is used to deliver multimodal voice, data and fax solutions for use within IP, PSTN and mobile networks – with performance levels that are second to none.

For more information

To learn more about Aculab Cloud and Aculab's extensive telephony solutions visit:

www.aculab.com

aculab.com

AMERICAS
+1 (781) 352 3550 | sales@aculab.com

EMEAA
+44 (0) 1908 273802 | sales@aculab.com

Leverage the heritage of Aculab when you move to the cloud

Moving your application development environment to a cloud infrastructure is a big step. Despite the clear benefits of cloud migration, it's natural for developers of hardware-based solutions to be concerned about the risks of moving their technology IP – and the years of investment and knowledge that has gone into creating it – to a new cloud development platform. Most of the big names in cloud communications are relatively new entrants to the communications market; some are working with open source technologies and, as the market consolidates, it is likely that many will not be in business in just a few years' time. So how do you know that a cloud platform can deliver the same level of reliability and performance that you've come to expect from a hardware deployment, and that it will be around for decades?

Three decades of innovation — the next chapter

Aculab Cloud deploys Aculab's industry benchmark technology and has been built organically out of more than 35 years' worth of experience in the communications enablement market. Put simply, it's the result of more than three decades of experience and innovation.

Aculab Cloud developers can be assured that the technology that powers Aculab Cloud has been used to enable tens of thousands of mission-critical applications across the world. Aculab Cloud features robust, field-proven protocols that have been developed and honed in conjunction with thousands of developers and deployed across hundreds of networks.

It's the only cloud communications platform that delivers the expertise, experience and reliability that you get from working with a proven communications enabler.

Leverage our heritage when you move to the cloud.