SIP SoftPhone for Salesforce

Salesforce integration with the Mizu WebPhone

About

In this tutorial we will go through the steps in adding standard VOIP/SIP phone call functionality to Salesforce platform using Mizutech Webphone.

WebPhone is a modern cross-platform browser SIP softphone from MizuTech, offering integration capabilities with any third-party software, including Salesforce.

Salesforce is a popular cloud based CRM solution from Salesforce, offering integration capabilities with other systems.

Integrating the Webphone you will add VoIP capabilities to Salesforce such as voice calls, video calls, conference, chat, presence, call recording and the capability to:

- interconnect with any VoIP service provider or use your own IP-PBX
- make cheap outbound calls to landline/mobile
- make free calls to other webphones, softphones or IP phones (including agent to agent calls)
- accept incoming voice calls, video calls and text messaging

In this document we are going to demonstrate how easy it is to add telephony capability to Salesforce platform, by integrating Webphone. The Webphone is an all-in-one VoIP client module which can be used as-is (as a ready to use softphone or click to call solution) or as a JavaScript library (to implement any custom VoIP client or add VoIP call capabilities to existing applications).Here we are going to use the Webphone as a Javascript library to integrate it with Salesforce using Open CTI (Computer-Telephony Integration) toolkit.

Using Salesforce CTI's API and the WebPhone API we can "interconnect" them to provide a full featured, cross-browser compatible and flexible solution for initiating/receiving calls or sending/receiving chat messages.

Using the Web phone involves the following steps:

1. Copy the Webphone files to webserver
2. Add the Webphone to salesforce using the Salesforce Open CTI capability/API (discussed below)
3. Configure the Web VoIP library in the webphone_api.js (set your SIP server address as the “serveraddress” parameter)
4. Add some code to Salesforce’s preferred Visualforce (but you can also use plain JavaScript) and define a Call Center (discussed below)

Once the Webphone is started from Salesforce:

1. The web sip client will connect (register) automatically to the configured VoIP server with the provided SIP username/password
2. At this point the web sip client is ready to accept incoming calls. You can use the onCallStateChange callback to catch the incoming calls (and display it to the agent)
3. To make outbound calls, you just have to use the call() API. Example: call(+441234567);
4. Optionally use other webphone features such as sendchat(), dtmf() or any others after your needs

For the sake of simplicity we are just going to add click to dial functionality to Salesforce, but this can easily be extended to support other features, like receiving calls or engaging conversations using chat.

If you are completely new to Salesforce platform, I would suggest having a look to this getting started tutorial first.

If you are completely new to Mizu WebPhone then have a look at the webphone quick start guide and check the documentation.

Getting down to business, let's tackle the problem in two parts:

1. Setting up a Salesforce Help Desk and creating a Salesforce console app to use Open CTI
2. Create an Open CTI SoftPhone and integrate the Webphone

Salesforce setup

In this step we will setup a Salesforce Help Desk and will create a Salesforce console app to use Open CTI.

Prerequisites:

1. Check the Salesforce Open CTI documentation if you don't have any experience yet
2. If you don't already have a Salesforce account, then begin by creating a free developer account here. After the account is ready, sign in with it.

First we will create a Salesforce console app with basic functionality so that agents can select it from the Force.com app menu, located in the upper right of every Salesforce page.

1. From Setup, enter Apps in the Quick Find box, then select **Apps** and click **Next**
2. Select Console and click **Next**.
3. Type a label for the app (this will be the app's name in the Force.com app menu) and click **Next**.
4. Select the items to include in the navigation tab, which is a tab where agents can select objects and choose lists. You might add Cases and Accounts here which are often used by Agents. Click **Next** once ready.
5. Choose how items display in the console when they're not selected from a primary tab or subtabs and then click **Next**

6. Check the Visible box next to any user profiles that will access the console. For example, if you have a user profile for agents, make the console visible to that profile.
7. Click **Save**.

We've created a console app and configured its tabs, and later we'll give agents access to your console so that they find it on any page in Salesforce. As an administrator, you can create more such apps if needed later.

Now you need to assign users to your console to allow access to it. We will assign the Service Cloud User license to agents so that they can start using your console.

1. From Setup, enter Users in the Quick Find box, and then select **Users**.
2. Click **Edit** next to an agent's name.
3. Select the "Service Cloud User" checkbox option and click **Save**.

Add more users in the same way and don't forget to add access also for yourself to be able to access the console.

Add the SIP WebPhone to Salesforce

In this step we will create an Open CTI SoftPhone and integrate the Webphone int Salesforce.

We will need to setup a VoIP call channel or computer-telephony integration (CTI) as named in Salesforce, so that your customers can contact an agent from a phone. More exactly we will add a Softphone call-control tool to the footer of the Salesforce console so that agents will be able to initiate or answer phone calls and update customer details in the CRM while speaking with customers.
To keep this guide short, for now we integrate the web sip phone basic click to call capabilities only to let the agents to initiate SIP calls by clicking on the phone numbers and screen pop a contact record, however this working example will be trivial to extend with any features you might need.

We are going to host the webphone on web site at "https://www.domain.com/webphone/". You will need to replace the domain.com with your website's domain and path where you are hosting the webphone.

First we have to configure a VoIP account in the webphone webphone_api.js file. This consists of setting the following parameters at the beginning of the webphone_api.js file in parameters section (replace the words in maroon accordingly).

```
var parameters = {
        webphonebasedir: ' https://www.domain.com/webphone/', // the location of the webphone,
                                // where it is hosted . This is mandatory for the webphone in order
                                // to know, where to load its contents from.
        serveraddress: 'VOIP_SERVER_ADDRESS', // your VoIP server IP address, domain name
        username: 'SIPACCOUNT_USERNAME', // preconfigured SIP account username
        password: 'SIPACCOUNT_PASSWORD', // preconfigured SIP account password
```

```
        loglevel: 5,
        //you can add any extra parameters here after your needs
    };
```

Webphone parameters (serveraddress, username, password) can be also set dynamically using the webphone_api.setparameter (param, value) API function. For more details about the webphone Javascript API, please check the "***JavaScript API***" section in the webphone documentation.

Now we are going to write some code in Visualforce.
(However, if you wish, you can write code for Open CTI in other languages, and host pages related to Open CTI on servers that have little to do with Salesforce).
Make sure to have the Development Mode enabled so that you can create Visualforce pages.

1. To create a Visualforce page, from Setup, enter Visualforce Pages in the Quick Find box, then select **Visualforce Pages**, and then click **New**.
2. Type a label and name for the page, such as softphone.
3. Copy/paste the code below into your page. This code determines the webphone functions and you can easily extend it later if required

```
<apex:page showHeader="false">
<apex:includeScript value="/support/api/29.0/interaction.js"/>
// include the webphone
<apex:includeScript value=" https://www.domain.com/webphone/webphone_api.js"/>

<style>
   body {background-color:white;}
</style>
<script type="text/javascript">
  // Variable that keeps track of click-to-dial state. If true, click-to-dial is enabled, false otherwise.
   var isClickToDialEnabled = false;
   // Callback used with enableClickToDial API method
   var enableClickToDialCallback = function (response) {
      isClickToDialEnabled = true;
   };

   // Callback used with disableClickToDial API method
   var disableClickToDialCallback = function (response) {
      isClickToDialEnabled = false;
   };

   //Toggle the click-to-dial feature.
    function toggleClickToDial() {
      if (isClickToDialEnabled) {
         // This function allows phone elements to be clickable on a Salesforce page.
```

```
            sforce.interaction.cti.disableClickToDial(disableClickToDialCallback);
        } else {
            // Enable click-to-dial.
            sforce.interaction.cti.enableClickToDial(enableClickToDialCallback);
        }
    }

    // Callback for screenPop API method.
    var screenPopCallback = function (response) {
        if (response.result) {
            alert('Screen pop was set successfully.');
        } else {
            alert('Screen pop failed.' + result.error);
        }
    };

//screenPop to a contact in your organization. Replace contact id with a valid one
    function screenPop() {
        var objectIdRelUrl = '/003D000000PS4iL'; // Replace the Id with a valid one from your organization.
        sforce.interaction.screenPop(objectIdRelUrl, true, screenPopCallback);
    }
    // Callback for onClickToDial API method.
    var onClickListener = function (response) {
        if (response.result) {
            console.log(User clicked on a phone number. The data returned as JSON format is : " +
response.result);
            //make sure to extract the target number from the JSON data when passing to the call() function
            webphone_api.call(response.result)
        }
    };

    // Registers a callback function that will execute when a user clicks on a phone link.
    sforce.interaction.cti.onClickToDial(onClickListener);
</script>
    <button onclick='toggleClickToDial();'>Toggle Click-to-Dial</button>
    <button onclick='screenPop();'>Screen Pop</button>
</apex:page>
```

4. Click **Save**.

We are ready with a basic phone channel now, but it contains only limited capabilities allowing the agents to make phone numbers clickable in Salesforce, and click a button to screen pop a contact record.

More details about Open CTI and its methods can be found here.

Create a Call Center

In the previous step, you created a webphone, but you need to also hook it up to a call center definition file to be more usable. A call center definition file just tells Salesforce what your call system can do. The file is a few lines of code in an XML file. Let's create a call center definition file named callcenter.xml and saved it to your desktop. Here's what it looks like:

```
<callCenter>
   <section sortOrder="0" name="reqGeneralInfo" label="General Information">
    <item sortOrder="0" name="reqInternalName" label="InternalName">DemoOpenCTI</item>
    <item sortOrder="1" name="reqDisplayName" label="Display Name">Demo Call Center Open
CTI</item>
    <item sortOrder="2" name="reqAdapterUrl" label="CTI Adapter URL">/apex/softphone</item>
    <item sortOrder="3" name="reqUseApi" label="Use CTI API">true</item>
    <item sortOrder="4" name="reqSoftphoneHeight" label="Softphone Height">300</item>
    <item sortOrder="5" name="reqSoftphoneWidth" label="Softphone Width">500</item>
   </section>
</callCenter>
```

Import the call center definition file into Salesforce so that our SoftPhone can "talk" with Salesforce:

1. From Setup, enter Call Centers in the Quick Find box, then select **Call Centers**. If an Introducing Salesforce CRM Call Center page displays, click **Continue**.
2. Click **Import** then click **Choose File** to navigate to the call center definition file on your desktop.
3. Click **Open** to enter the path in the Call Center Definition File field and then click **Import**.

Before your service team can access a Web SIP Softphone, we must assign the agents to the call center we created.

1. From Setup, enter Call Centers in the Quick Find box, then select **Call Centers**.
2. Click the name of the call center we created.
3. In the Call Center Users related list, click **Manage Call Center Users** and then click on **Add More Users**.
4. Specify search criteria to find agents who should be assigned to the call center. Since you're an administrator who wants to use a SoftPhone, let's enter the criteria of Profile Contains Administrator. Click **Find**.
5. Check the box next to your name, and click **Add to Call Center**.

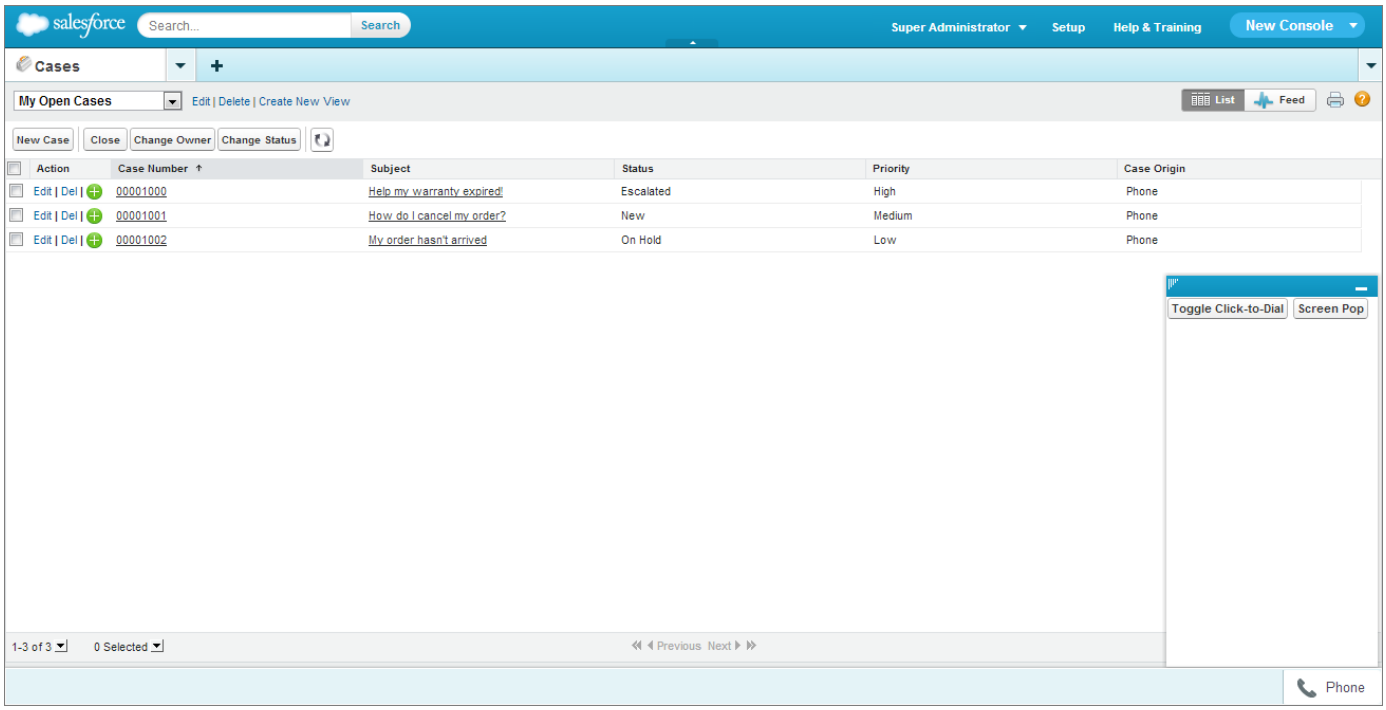
Now, you and your service team agents can use a Web Softphone from the Salesforce console we created previously.

Run the WebPhone on Salesforce

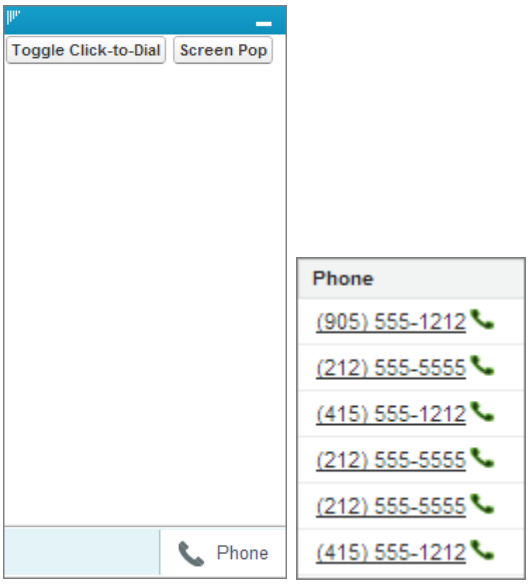
In this step we will check how WebPhone works in the Help Desk

So far we've created a webphone, defined a call center, and assigned your service team to the call center. In this step, we'll use the Web Phone to make calls by clicking phone numbers in the console we created in part one:

1. Go to the console we created by clicking **New Console** from the Force.com app menu.
2. Click **Phone** in the footer of the console.



3. Click **Toggle Click-to-Dial** and go to any record or list view with phone numbers. Notice that the phone numbers are links that you can click to place a call.



There you have it—we've set up the WebPhone integrated with your Salesforce CRM console. Now, your service team can make and receive regular VoIP calls directly from Salesforce while interacting with the customers details, considerably boosting your agent's productivity.

Next Steps

Following this tutorial you have accomplished the most important functionality of the WebPhone integration: the agents now can initiate and receive standard SIP calls from Salesforce.
You can easily extend the above example and add more features to your Salesforce Softphone if needed.

Here are a few possibilities:

- Pass the serveraddress/username/password dynamically (if these have to be changed by account). For example you can use the SIP username/password assigned to the logged-in Salesforce agent. Use the setparameter() webphone API to pass these at run-time or generate the username/password parameters dynamically from the CTI
- Setup callbacks for the webphone state changes and display it accordingly for the agents (onStart, onRegistered, onCallStateChange webphone callbacks)
- Create a better phone user interface (or use one of the templates provided with the mizu webphone such as the softphone or click to call user interface)
- Add chat support (you just need to call the webphone sendchat() api to send messages and setup an onChat() callback to be able to catch and display somewhere the incoming messages)
- Add more call control features such as hold/mute/transfer/conference (just call the webphone API accordingly for this)

- Add more integration by calling from webphone state changes to Salesforce (for example you can save call details after each call from the onCdr() webphone callback)

Consult the followings for more help:

Salesforce homepage
Mizutech WebPhone homepage
WebPhone documentation
Salesforce CTI
Salesforce CTI HelpDoc
CTI Methods
Salesforce Service Cloud Workbook