

Outbound SMS Dialer

Duration: 15 minutes

Objectives

In this lab, you'll create a basic dialer that will send multiple SMS to a list of Mobile phone numbers read from a CSV file.

Pre-requisites

This lab assumes you have some basic knowledge of JavaScript, and you have experience creating Tropo applications. Alternatively, you could have gone through the Tropo introductory lab: “Create a Voice Machine”.

You'll need a Tropo account that has been enabled for Outbound SMS.

If you're attending a Cisco event, reach to your instructor to get activated. If you're running this lab offline, [reach to Tropo support by email](#) and ask for the procedure to activate your Tropo account.

How to setup your own computer

This lab can be completed from any platform with an HTML5-compatible browser.

Step 1 – Upload the CSV file

In this section, we'll upload a CSV file to your private Tropo file manager.

On your local machine, create a new file named “numbers.csv” in the following format. Adapt the contents with other lab participants.

```
name,number
Phil,+15555551212
Enrico,+394253345321
Steve,+33678007834
```

From the [Tropo dashboard](#), log into your account.

Navigate to the **MY FILES** tab. This is the Tropo file manager, which you can use to host your resource files for free and access them directly from Tropo when you need to edit them.

Note that you can upload files using the Tropo file manager or you can use an FTP client. If you use an FTP client, set the host name to `ftp.tropo.com`, and use your Tropo dashboard credentials to login.

Click on the folder named **www**, and click on the folder icon labeled **Create New Directory**. Name this new directory 'data'.

Open your new directory folder and click **Upload file**.




Select the “numbers.csv” file you just created.




MY APPS MY FILES LOGS BILLING ACCOUNT

My Files

Manage your scripts and associates files, such as audio files and logs, from here.

Current Path: ~ / [www](#) / [data](#)

File	Size	Date	Action
 numbers.csv	75 bytes	11/22/2016 11:38:00 PM	 

 [Upload file](#)  [Create new file](#)  [Create new directory](#)

Step 2 – Parse the CSV file

In this section, you will create a new Tropo application whose Javascript code will parse the CSV file we just uploaded.

From the Tropo dashboard, navigate to the **MY APPS** tab and select **Create application**.

Name your new App 'CSV Dialer'. Select **Scripting API** under **Type of Application**.

Open the script editor by clicking the pencil icon titled **New script**. Paste the code below, name the file dialer.js, and click **Save**.

```
function loadFile(url) {  
    var line;  
    var returnFile = "";  
    connection = new java.net.URL(url).openConnection();  
    connection.setDoOutput(false);  
    connection.setDoInput(true);  
    connection.setInstanceFollowRedirects(false);  
    connection.setRequestMethod("GET");  
    connection.setRequestProperty("Content-Type", "text/plain");  
    connection.setRequestProperty("charset", "utf-8");  
    connection.connect();  
  
    var dis = new java.io.DataInputStream(connection.getInputStream());  
    while (dis.available() != 0) {  
        line = dis.readLine();  
        returnFile += line + "#end";  
    }  
    return returnFile;  
}  
  
function csvJSON(csv) {  
    var lines=csv.split("#end");  
    var returnJSON = [];  
    var headers=lines[0].split(",");  
    for(var i=1;i<lines.length;i++){  
        var obj = {};
```

```
    var currentline=lines[i].split(",");  
    for(var j=0;j<headers.length;j++){  
        obj[headers[j]] = currentline[j];  
    }  
    returnJSON.push(obj);  
}  
return returnJSON;  
}
```

In order to load the data from our CSV file and convert it to JavaScript Object Notation (JSON), we are leveraging 2 functions:

- *loadFile(), loads the file at the url passed into it. This file uses Java to load the file and can be used with Tropo-hosted apps, as long as the file you're loading is hosted in your Tropo account.*
- *csvJSON() is based on the csv-to-json.js script from TechSlides.com, which can be found here: <https://gist.github.com/iwek/7154578#file-csv-to-json-js>. This script does not work for data that contains commas inside of strings and may break in other cases as well. For production uses, you may need a more robust CSV to JSON conversion script, such as Papa Parse (<http://papaparse.com/>).*


Click **Create App** and proceed to next section where we will add code to send SMS.

Step 3 - Making the calls

Reach to the “Text Script” section of your application, and click **Edit script**.

Text Script:

/www/dialer.js

 [Edit script](#)

 [Select my files](#)

SAVE SETTINGS

DELETE APPLICATION

Append the block of code below to load your CSV file into a JSON array. **Make sure to Replace the 5554321 numbers before the “/www/”** in the file path with your unique Tropo account ID, located in the upper right of the Tropo dashboard; right after your user name.

```
var csvFile = loadFile("http://hosting.tropo.com/5554321/www/data/numbers.csv");
var numbersToDial = csvJSON(csvFile);
```

To complete our code, we now need to actually call to the phone numbers listed in our CSV file.

Write a “for loop” that will iterate through each of the people in the JSON object. Inside that “for loop”, we’ll create a variable to represent the properties of the current person, initiate a new call on a SMS channel, personalize a message to be sent to the callee, and hang up to proceed to the next callee.

```
for (var i = 0; i<numbersToDial.length-1; i++){
  var callee = numbersToDial[i];
  call(callee.number, { network: "SMS" });
  say("Hi, " + callee.name);
  hangup();
}
```

*Note that for a very small number of calls, using the **call** method inside a for loop is an acceptable way to call multiple people. In a production application, you should use the Tropo REST API to initiate each of the calls individually.*

Click **Save**. The code editor closes.

To complete this section, we will now attach a phone number to our application. Tropo will use that number to actually deliver the SMS to each recipient.


Reach to the “Numbers” section of your application.

Select a phone number that is SMS-enabled and click **Add**.

If you are living anywhere else than US or Canada, pick a number in Canada, as Canada is Tropo’s gateway to issue international calls. In other words, it is the place from which you should get the best reach.

Numbers

Phone numbers associated with your application. You can delete existing numbers or add new ones.

	Number	Country	Region	Cost (monthly)	Connection Support
	+1 306-988-8875	Canada	Regina	\$1	Voice + Text
Add new number:	<input type="text" value="Canada"/>	<input type="text" value="1902 CHARLOTTETOWN (\$1)"/>	<input type="button" value="ADD"/>		

Step 4 –Invoke your script

The last step of this lab is to invoke our script.

The difficulty here is that we need a hook, a way to tell Tropo to create a new session in which our application will load and execute its JavaScript code. For that purpose, Tropo defines two API keys that let you invoke the Voice and SMS scripts respectively.

These API keys are located at the bottom of the application settings.

API Keys

Your API keys can be managed here.

voice	43764c4149767773444b51574a6a50454e78794c5864414664585a43544972565363534575	copy to clipboard	regenerate	launch	see token URL
messaging	5741414945584f6756625a4b737a734b7564576c754467464552486472504c6c70786f50616 <small>https://api.tropo.com/1.0/sessions?action=create&token=5741414945584f6756625a4b737a734b7564576c754467464552486472504c6c70786f50616</small>	copy to clipboard	regenerate	launch	hide token url

Click the **launch** link corresponding the messaging API key. This action will fire the script we edited earlier. *Note that behind the scene, a GET request is sent against the Tropo API endpoint: <https://api.trom.com/1.0/sessions>. To reveal that URL, click on “see token URL”.*

If you're working with a Tropo account authorized for Outbound SMS, and the CSV file has been correctly uploaded, your script should start texting each of the phone numbers in order!

If you encountered any trouble, copy paste the complete script provided in next section.

Clean up instructions

In order to preserve Tropo Development Platform resources, we will remove some resources you reserved through the lab.

Reach to your Tropo application, and delete the phone numbers attached to your application by clicking the Red sign on the left of the phone numbers.



The complete code

Your finished script should look like this:

If you paste this script, make sure to replace the Tropo account number below (5554321) with your own Tropo account.

```
var csvFile = loadFile("http://hosting.tropo.com/5554321/www/data/numbers.csv");
var numbersToDial = csvJSON(csvFile);

for (var i = 0; i<numbersToDial.length-1; i++){
    var callee = numbersToDial[i];

    call(callee.number, { network: "SMS" });
    say("Hi, " + callee.name);
    hangup();
}

//file loading function.
function loadFile(url){
    var line;
    var returnFile = "";
    connection = new java.net.URL(url).openConnection();
    connection.setDoOutput(false);
    connection.setDoInput(true);
    connection.setInstanceFollowRedirects(false);
    connection.setRequestMethod("GET");
    connection.setRequestProperty("Content-Type", "text/plain");
    connection.setRequestProperty("charset", "utf-8");
    connection.connect();

    var dis = new java.io.DataInputStream(connection.getInputStream());
    while (dis.available() != 0) {
        line = dis.readLine();
        returnFile += line + "#end";
    }
}
```



```
    return returnFile;
}

function csvJSON(csv) {
    var lines=csv.split("#end");
    var returnJSON = [];
    var headers=lines[0].split(",");
    for(var i=1;i<lines.length;i++){
        var obj = {};
        var currentline=lines[i].split(",");
        for(var j=0;j<headers.length;j++){
            obj[headers[j]] = currentline[j];
        }
        returnJSON.push(obj);
    }
    return returnJSON;
}
```