Create Twiml App:

1) Login with your Twilio credentials
2) Move to Programmable Voice
3) Click on TwiML
4) Click on TwiML app
5) Click on Plus icon (+)
6) Fill the data and click on create
7) Give friendly name, Request URL give as (“https://”+ your runtime domain+”client-voice”)
8) Now Twiml app created

Create Capability Token URL:

1) Click on Runtime
2) Click on Overview
3) Click on Functions
4) Click on Plus icon (+)
5) Select Twilio Client QuickStart
6) Paste your Twiml app sid in the TWIML_APP_SID textfield
7) Caller_ID is your Twilio number
8) Click on create
9) Now Twilio Client QuickStart (Capability Token) and Twilio Client QuickStart (voice calls) created
10) To get the capability token URL, click on Twilio Client QuickStart (CapabilityToken) and copy the path URL

*****IMPORTANT*********

Replace the following code in Runtime Functions:

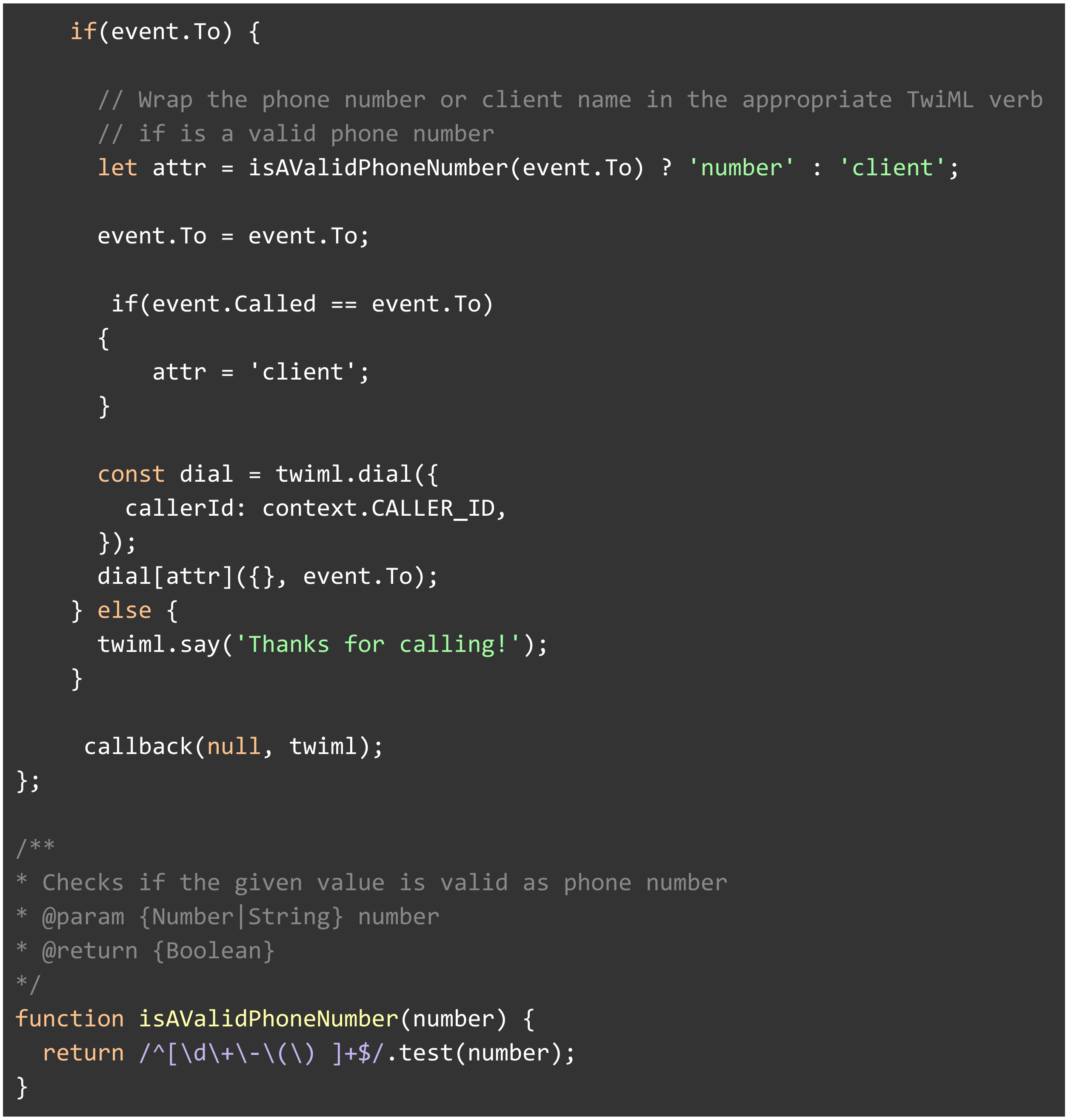
1) Click on Runtime
2) Click on Functions in Runtime Overview
3) Click on Twilio Client Quickstart (Voice calls)
4) Go to configuration- code

Replace the following code

```
exports.handler = function(context, event, callback) {
    let twiml = new Twilio.twiml.VoiceResponse();

    console.log("Event = "+event);

```



```
    if(event.To) {

      // Wrap the phone number or client name in the appropriate TwiML verb
      // if is a valid phone number
      let attr = isAValidPhoneNumber(event.To) ? 'number' : 'client';

      event.To = event.To;

       if(event.Called == event.To)
      {
          attr = 'client';
      }

      const dial = twiml.dial({
        callerId: context.CALLER_ID,
      });
      dial[attr]({}, event.To);
    } else {
      twiml.say('Thanks for calling!');
    }

     callback(null, twiml);
};

/**
* Checks if the given value is valid as phone number
* @param {Number|String} number
* @return {Boolean}
*/
function isAValidPhoneNumber(number) {
  return /^[\d\+\-\(\) ]+$/.test(number);
}
```

In the same way, replace the following code in Twilio Client Quick Start (Capability token):

```
/*
 *  Twilio Client Quickstart Template
 *
 *  This Template shows you how to create capability tokens for Twilio Client. Please note, this is for prototyping purposes
 *  only. You will want to validate the identity of clients requesting Capability Tokens in most production applications and set
 *  the identity when minting the Token.
 *
 *  Each function has a publicly accessible URL which a malicious actor could use to obtain tokens for your app and abuse them.
 *  See the section on capability tokens below to learn how to generate capability tokens in your
 *  own C#, Java, Node.js, PHP, Python, or Ruby application.
 *
 *  More reading
 *  - Capability Tokens (https://www.twilio.com/docs/voice/client/capability-tokens)
 *  - In-depth tutorial on Capability Tokens (https://www.twilio.com/docs/voice/client/tutorials/capability-tokens)
 */

exports.handler = function(context, event, callback) {

  let response = new Twilio.Response();

  // Add CORS Headers
  let headers = {
    "Access-Control-Allow-Origin": "*",
    "Access-Control-Allow-Methods": "GET",
    "Content-Type": "application/json"
  };

  // Set headers in response
  response.setHeaders(headers);

  response.setStatusCode(200);
  let ClientCapability = require('twilio').jwt.ClientCapability;

  const identity = event.To;

  const capability = new ClientCapability({
    accountSid: context.ACCOUNT_SID,
    authToken: context.AUTH_TOKEN,
  });

  capability.addScope(new ClientCapability.IncomingClientScope(identity));
  capability.addScope(new ClientCapability.OutgoingClientScope({
    applicationSid: context.TWIML_APP_SID,
    clientName: identity,
  }));


  // Include identity and token in a JSON response
  response.setBody({
    'identity': identity,
    'token': capability.toJwt(),
    'context':context,
    'event':event
  });

  callback(null, response);
};
```

//

Twilio Plugin Configuration in Appup:

1) PLUGIN SRC NAME : AppupTwilio
2) PLUGIN SRC: https://appup-cdn.s3.amazonaws.com/static/twilio.js
3) HOLD URL :

Suppose Your Hold URL as like below :

https://pbxplus.500apps.com/pbxplus/holdcall/ACd15e7f28f2fdf6df554b3e96f80ca1e7/f35611bb765a3acd8012cddaff81f0ec/CAfd0619412a0d6b1e4c3ab5f13190fd1f.

Then give the following URL in plugin config as

```
https://pbxplus.500apps.com/pbxplus/holdcall/{{AccountSID}}/{{AuthToken
}}/{{callsid}}
```

4) CALL TRANSFER URL:

Suppose your call transfer URL as like below:

https://pbxplus.500apps.com/pbxplus/transfercall/ACd15e7f28f2fdf6df554b3e96f80ca1e7/f35611bb765a3acd8012cddaff81f0ec/CAfd0619412a0d6b1e4c3ab5f13190fd1f/+918074444729

Then give the URL in plugin config as:

```
https://pbxplus.500apps.com/pbxplus/transfercall/{{AccountSID}}/{{AuthToken
}}/{{callsid}}/{{transfernumber}}
```

The keys should be in the following words

1) AccountSID
2) AuthToken
3) callsid
4) transfernumber

5) CALL CONFERENCE URL:

This is under development.

Send your Account SID, AuthToken and CapabilityTokenURL as below

```
window. AppupPluginsManager.exec("AppupTwilio","setTwilioPrefs",
{
"capabilityTokenURL":"https://lion-rail-2261.twil.io/capability-token
",
"AccountSID":"AC3ac5d09ed209d8771f048de25115a80a",
"AuthToken":"9678a806b66b0ce6534c7e0eda9fd07a",
"ConfigureNumber":"+12727702019"
});
```

Things to Remember:

1) While sending Configure Number (Your device number) to setTwilioPrefs (above functionality), send in E.164 format. Refer the following link

   https://www.twilio.com/docs/glossary/what-e164

2) Your number must configure with TWIML APP, following are the steps to configure TWIML APP

   i) Click on Phone Numbers
   ii) Click on number
   iii) Move to Voice or voice & Fax (if fax configurable only)
   iv) Select Twiml app
   v) Select your app