The goal of this library is to turn the browser in a softphone, using the correct protocol depending on the browser used by the client. On supported browsers, the goal is to allow calls using WebRTC, and to fall back on the RTMP protocol if WebRTC is not supported by the browser.

Initialization of the Library

First of all you need to include the JavaScript in the browser. In order to do so, you need to add the following snippet in your HTML.

```
<script src="lib/kazoo.js" type="text/javascript"></script>
```

Once this is done the library will be loaded at the same time that this HTML page is accessed. It will create a global variable named kazoo that you can access from anywhere in your JavaScript code.

Now that the library is loaded, we need to initialize it by calling kazoo.init(params). The expected parameters are the following:

- onLoaded: Javascript callback called once the library has been loaded
- onFlashMissing(container): Javascript callback called when trying to use RTMP without Flash available. The div initially supposed to contain the Flash element is provided as a parameter.
- flashContainer: OPTIONAL, id of a div that will contain the Flash Embed element
- forceRTMP: OPTIONAL, if set to true, will force the library to use the RTMP protocol.

```
var paramsInit = {
    forceRTMP: false,
    flashContainer: 'flash_div',
    onLoaded: function() {
        document.getElementById('loginForm').style.display = 'block';
    },
    onFlashMissing: function(container) {
        container.innerHTML = 'This requires the Adobe Flash Player. ';
        container.className = 'flash-missing';
    }
};

kazoo.init(paramsInit);
```

Example 1: Initializing the Library

Now that the library has been initialized, we can use it to register to some SIP credentials and we'll then be able to place and receive calls!

In order to register to some SIP credentials, we need to use the kazoo.register function, the expected parameters are the following:

- ***wsUrl***: Web Socket Server URL (eg: ws://10.26.0.41:8080)
- ***rtmpUrl***: RTMP Server URL (eg: rtmp://10.26.0.41/sip)
- ***realm***: SIP Realm (Realm linked to your Kazoo account, eg: d218ds.sip.2600hz.com)
- ***privateIdentity***: Full SIP Address (eg: sip:user_31dsajsjds@ d218ds.sip.2600hz.com)
- ***publicIdentity***: SIP Username (eg: user_31dsajsjds)
- ***password***: SIP Password (eg: 23bf1f9wwdslw2)
- ***onIncoming***: Javascript callback called once an incoming call has been detected. It takes one argument, a call object that has 2 methods: accept, and reject (which will either accept or reject the call) and one attribute: callerName (which will give you the callerId for this incoming call).
- ***onConnected***: OPTIONAL, Javascript function called by the library once the user is successfully registered
- ***onAccepted***: OPTIONAL, JavaScript function called by the library once a call originated by the browser has been picked up
- ***onHangup***: OPTIONAL, Javascript callback called once a call has been terminated.
- ***onCancel***: Javascript callback called whenever a call gets cancelled/rejected. The cancel details (such as SIP code, message and event) are provided as an object parameter when available.
- ***onTransfer***: Javascript callback called once a call has been transferred
- ***onNotified***: Javascript callback called whenever a notification is sent. The notification details are provided as an object parameter. See below for the list of currently existing notifications.
- ***onError***: Javascript callback called whenever an error is returned. The error details are provided as an object parameter. See below for the list of currently existing errors.
- ***reconnectMaxAttempts***: OPTIONAL, Integer value representing the maximum number of attempts to auto-reconnect after getting disconnected unexpectedly. **0** will disable the auto-reconnect feature, while a negative value will set it to attempt reconnecting indefinitely (until it finally connects). The default value is **-1**. *Note: getting your registration overridden will not attempt to automatically reconnect*.
- ***reconnectDelay***: OPTIONAL, Integer value representing the delay in milliseconds before attempting to auto-reconnect after a disconnection or a failed re-connect attempt. The minimum value is 1000. The default value is 5000.

Here is an example of a call of the register function:

```
function onIncoming(call) {
      confirm(call.callerName + ' is calling you! Pick up the call?') ?
call.accept() : call.reject();
}

function onHangup() {
      document.getElementById('divCalling').style.display = "none";
}

var registerParams = {
      forceRTMP: true,
      wsUrl: 'ws://10.26.0.41:8080',
      rtmpUrl: 'rtmp://10.26.0.41/sip',
      realm: document.getElementById('realm').value,
      privateIdentity: document.getElementById('privateIdentity').value,
      publicIdentity: document.getElementById('publicIdentity').value,
      password: document.getElementById('password').value,
      onAccepted: function() { /*****/ },
      onConnected: function() { /*****/ },
      onHangup: onHangup,
      onIncoming: onIncoming,
      onCancel: function(cancelDetails) { /*****/ },
      onTransfer: function() { /*****/ },
      onNotified: function(notification) { /*****/ },
      onError: function(error) { /*****/ },
      reconnectMaxAttempts: 3,
      reconnectDelay: 10000
};
kazoo.register(registerParams);
```

Example 2: Registering to a SIP account

In Example 2, we registered to a SIP account (with credentials inputted in a HTML form). If it registered properly, you'll now able to place and receive calls from your browser!

Below is the complete list of notifications and errors that you can handle in the onNotified and onError callbacks. The notification and error objects follow the same structure, including a string ***key***, a string ***message*** explaining the error/notification, and a ***source*** object containing the source of the error/notification. Note that the source object structure will vary depending on the protocol used.

"key"	"message"	Occurence	Available on	
			WebRTC	RTMP
server_not_reachable	Could not reach the server.	***Error*** returned when trying to register a device to an unreachable server.		X
unauthorized	Invalid credentials.	***Error*** returned when trying to register a device with invalid credentials.	X	X
disconnected	You have been disconnected.	***Error*** returned when the server unexpectedly disconnects you (e.g. the server shuts down).	X	X
overriding_registration	You have overridden an existing registration for this device.	***Notification*** sent when registering a device that is already currently registered.	X	X
replaced_registration	You have been disconnected: someone else has registered this device.	***Notification*** sent when your currently registered device is being registered somewhere	X	X

		else.		
voicemail_notification	You have one or more voicemail message(s).	***Notification*** sent when the server detects one or more messages on your device's voicemail box.	X	X
connectivity_notification	You have lost your connection to the network. OR You have recovered your connection to the network.	***Notification*** sent when the network connectivity changes. This notification also has a *status* that may contain the values 'online' or 'offline'.	X	X
reconnecting_notification	Attempting to reconnect...	***Notification*** sent when the kazoo.reconnect() function is called, either manually or automatically from the auto-reconnect feature.	X	X
unknown_notification	You have received a SIP notification.	***Notification*** sent when the server sends a SIP notification that isn't recognized in any of the above categories.	X	X

Now that we looked at how to get started, we'll look at the different methods of the kazoo object.

Methods

The kazoo object has 7 accessible methods:

- ***init***: Method to initialize the Library, see Example 1
- ***register***: Method to register using some SIP credentials, see Example 2
- ***connect***: takes a SIP URL as the only parameter, and will try to call the following address (eg: kazoo.connect('sip:2222@dd21d.sip.2600hz.com'))
- ***hangup***: no arguments needed, will hangup the current call. (eg: kazoo.hangup());
- ***sendDTMF***: takes a character from this list 0,1,2,3,4,5,6,7,8,9,0,*,#, as a parameter and send it as a DTMF to the platform (eg: kazoo.sendDTMF('2'));
- ***logout***: Will unregister the browser from the platform (eg: kazoo.logout());
- ***transfer***: takes a SIP URL as the only parameter, and will try to call the following address (eg: kazoo.transfer('sip:2222@sdsdd.sip.2600hz.com')).
- ***muteMicrophone***: takes a boolean as a parameter (*true* to mute, *false* to unmute), as well as two callbacks for success and error (e.g. kazoo.muteMicrophone(true, success, error);).]
- ***reconnect***: Attempts to re-register the device with the same parameters originally provided to the *register* function and sends a *reconnecting_notification*. This function gets called by the auto-reconnect feature.
- ***startAutoReconnect***: triggers the auto-reconnect using the *reconnectMaxAttempts* and *reconnectDelay* values provided in the *register* parameters. You should not have to call this method as it will be automatically called when getting disconnected (except when the registration is getting overridden).
- ***stopAutoReconnect***: stops the currently ongoing auto-reconnect. Note that the auto-reconnect automatically stops by itself as soon as the connection is successful.
- ***monitorConnectivity***: takes a Boolean as a parameter, which is true by default. This function allows to enable (true) or disable (false) the monitoring of your network connectivity. It is enabled by default, and disabling it will prevent the library from knowing when your connectivity has been loss, hence impairing the auto-reconnect feature. This is especially used for recovering from sleep/hibernate mode.

Using this, you should now be able to create some cool applications using softphones in your browser!

Contact

If you have any question or remark about the library or its documentation, feel free to contact me directly at jr@2600hz.com.