

Contents

- [1 Introduction](#)
- [2 Requirements](#)
 - ◆ [2.1 Launching the daemon](#)
- [3 Examples](#)
 - ◆ [3.1 Perl ESL::IVR Syntax](#)
- [4 Troubleshooting](#)
- [5 See Also](#)

Introduction

The FreeSWITCH IVR daemon ("ivrd") is a handy abstraction layer that sits on top of the [ESL](#). The basic idea is that the ivrd will allow the user to have a STDIN/STDOUT interface for simple call control. This makes it even easier to use the powerful event socket for handling calls.

Requirements

There are a few prerequisites that you will need:

- Executable script that can read/write to STDIN/STDOUT
- An unblocked TCP/UDP port where the ivrd can listen for socket connections
- If using [ESL::IVR](#) module then ESL's [perlmod](#) needs to be built
- At least one dialplan extension that sets the [ivr_path](#) variable and calls the [socket](#) application

Launching the daemon

The daemon must be left running in it's own process. We recommend using GNU [screen](#) and leaving the ivrd running in the foreground on one of the screens.

Launch the ivrd and specify the host and port:

```
/usr/local/freeswitch/bin/fs_ivrd -h 127.0.0.1 -p 9090
```

The daemon will be listening for socket connections on port 9090. (Use whichever port suits you.) The daemon can listen for connections on the localhost or on a different host. Be sure that you have good network connectivity between the host and the daemon if they are not on the same machine.

Examples

Using the ivrd launched above (localhost, port 9090), here is a simple dialplan entry:

```
<extension name="ivr-example1">
  <condition field="destination_number" expression="^(995\d)$">
    <action application="log" data="INFO Let's do some ivrd, shall we?"/>
    <action application="set" data="ivr_path=/usr/local/freeswitch/scripts/ivr-hello_world.pl"/>
    <action application="socket" data="127.0.0.1:9090 full"/>
  </condition>
</extension>
```

Note how we set the channel variable **ivr_path** before calling the **socket** app. The **ivr_path** variable must contain the name of an *executable* script. Below is an example of such a script:

```
#!/usr/bin/perl
#
# ivrd-hello_world.pl
#
# Launched from FreeSWITCH using socket app and ${ivr_path} variable
#
use strict;
use warnings;

# Use lib is easy, but you can also install your ESL.pm and esl/perl/ESL/* into site_perl
use lib '/usr/src/freeswitch/libs/esl/perl';
use ESL::IVR

$| = 1;          # Turn off buffering
select STDERR; # We'll take this stream, thanks
print "Starting ivrd-hello_world.pl...\n\n";

## Set the root dir for sound files
my $sound_root = '/usr/local/freeswitch/sounds/en/us/callie';

## Create the connection object which is basically an IVR
my $con = new ESL::IVR;

## Get this channel's uuid and dialed number
my $uuid = $con->{_uuid};
my $dest = $con->getVar('destination_number');

## Answer call, sleep for half second to let media get up and running
$con->execute('answer', '', $uuid);
$con->execute('sleep', '500');

## Play a simple greeting
$con->playback('ivr/ivr-welcome_to_freeswitch.wav');

## Tell the call what he/she dialed
$con->execute('sleep', '500');
$con->execute('say', "en number iterated $dest");

## Politely disconnect
$con->execute('sleep', '1000');
$con->playback('ivr/ivr-thank_you.wav');
$con->execute('sleep', '500');
$con->playback('voicemail/vm-goodbye.wav');
```

Perl ESL::IVR Syntax

The \$con object above has very simple syntax:

Get or Set a channel variable:

```
my $script_var = $con->getVar('var_name');
$con->setVar('var_name', 'var_value_to_set');
```

Execute an API:

```
my $result = $con->api('api_name', 'api args here');
```

Execute a dialplan application:

```
$con->execute('app_name', 'app args here');
```

Playback shortcut:

```
$con->playback('/path/to/sound/file.wav');
```

PlayAndGetDigits shortcut:

```
my $digits = $con->read('min', 'max', 'tries',
                       'timeout', 'termination_digits',
                       '/path/to/file.wav',
                       '/path/to/invalid_msg_file.wav',
                       'chan_var', 'regex');
```

Read shortcut:

```
my $digits = $con->read('min', 'max', 'chan_var',
                       '/path/to/file.wav',
                       'timeout', 'termination_digits');
```

Troubleshooting

If you are experiencing issues, like "socket errors" on the FS CLI then check a few things:

- Confirm that the port number of the daemon and the socket app are the same
- Test your extension using **nc** as described [here](#)
- Make sure that your script is executable
- Run your script from the command line and confirm that there are no syntax errors

See Also

- [Event Socket Outbound](#)
- [PHP Example](#)