FD Tone Notify

FD Tone Notify is Dispatch Tone Notification Software for Fire Departments and Enthusiasts

- Simultaneously detect tones for multiple departments or stations.
- Send Pushbullet, SMS, or email notifications. Integrate with Custom Webhooks
- Detect any number of tones from 1 to 5+ (no hard limit) with custom set tolerances and match thresholds. High accuracy and granularity to suit various applications
- Cross Platform and tested on the Raspberry Pi 3. The simplicity and low cost of the Raspberry Pi 3 allows new installations to be deployed quickly and cheaply.
- Web based remote monitoring interface with the ability to live stream audio.

🏳🏳

Getting Started

1. Setup: Installing Necessary Software
2. Getting Started, Usage, and Examples to determine the tones for your department and setup Pushbullet notifications

🏳🏳

Setup: Installing Necessary Software

Before using FD Tone Notify certain software needs to be installed so FD Tone Notify can access your audio input device(s).

Windows

1. Install SoX - Sound eXchange. Follow these setup instructions.
2. Once you have installed SoX following the steps above confirm, verify that sox has been added to your path. Open a new command prompt (cmd.exe) and type sox. If everything is working information on using sox will be printed to the console. If the path is not setup correctly an error "sox is not recognized" will display. If this error persists try restarting the computer and checking that the PATH was setup correctly.

Linux and Raspberry Pi

⚠️ **The Raspberry Pi does not support audio input by default. A USB sound card is required. This USB Sound Card has been tested and works well.**

1. Install ALSA

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install alsa-base alsa-utils
```

2.Copy `config/asound.conf` to `/etc/asound.conf` . This configures a `dsnoop` audio input that is a copy of hardware device `hw:1,0` . > ⚠️ **When setting the `inputDevice` in the config file it is critical to change the value to `dsnoop` on Linux/Raspberry Pi. If the default value of `hw:1,0` is not changed recording will fail and post recording notifications will not be sent**

Getting Started, Usage, and Examples

FD Tone Notify is a command line application.

🐧 **If you have never worked with a command line application before don't be scared.**FD Tone Notify is easy to setup and use and there a plenty of examples below.

Quick Start

To get started right away download FD Tone Notify and simply run `./fd-tone-notify` or doubleclick on `fd-tone-notify.exe` if you are using Windows.

This will create a `config/` directory that will store the configuration files. When run for the first time FD Tone Notify will automatically create a sample configuration file and blank `secrets.template.json`. For more information see the Configuration and Environment Variables and Secrets sections below.

How do I know what to enter for the tones?

If you don't know the tones for the department/station you want to monitor run FD Tone Notify with the `--tone-detector` option.

```
fd-tone-notify.exe --tone-detector --web-server
```

In this mode FD Tone Notify **WILL NOT** send notifications. Instead it will detect any multi-tones and print the results to console. For example:

```
crit : ALL TONE DETECTOR MUTLI-TONE DETECTED: 919Hz, 2940Hz
```

This indicates that a two-tone dispatch was detected with tones 919Hz, 2940Hz. Enter `tones: [919, 2940]` to detect this department.

ℹ The `--tone-detector` option cannot detect single tones. It will only print results for multi-tone matches.

💡 You can run FD Tone Notify in `--tone-detector` mode and normal "notification" mode at the same time. This is recommended when you are building out and fine tuning your configuration.

Running FD Tone Notify

Running FD Tone Notiy is simple. Just run

```
fd-tone-notify.exe --web-server --port 3000
```

This will start FD Tone Notify with the webserver option. The monitoring & Audio Streaming interface will be available on http://localhost:3000. FD Tone Notify will print a list of all of the configured detectors during the startup process. This is a perfect opportunity to verify your configuration.

Notification Basics

The easiest way to get started with notifications is by setting up Pushbullet. Create a new account (or signin). Once logged in Create A New Channel. The channel tag (unique) is what is needed in the notifications config as `channelTag`. The channel name can be anything.

Once the channel is created see Environment Variables and Secrets for instructions on creating and saving your Pushbullet API Key.

Download the Pushbullet iOs/Android application and Login. You will now receive push notifications from FD Tone Notify. Other users can subscribe to channel tag and also receive notifications.

Configuration

FD Tone Notify is configured via configuration files in the `config/` directory and environment variables.

⚠ *Advanced Feature* It is possible to run multiple instances of FD Tone Notify simultaneously using different config files by specifying an instance name. The *only* use case for this is to monitor multiple audio inputs. Otherwise list multiple `detectors` in the config file and run a single instance. See the `--instance-name` argument.

Config Files

When run for the first time FD Tone Notify will automatically create a `config/default.json` configuration file that can be modified.

💡 If you ever want to start with a new `default.json` config file rename your `config/` directory to `config_backup/` and a fresh default `config/` directory will be generated when you restart FD Tone Notify.

Config File Sections

Audio - Required

- `inputDevice` : The hardware device FD Tone Notify should use to listen for tones and recording. On Windows `hw:1,0` will always be the "default" input device selected in Windows. On Linux and Raspberry Pi this should be set to `dsnoop` as configured above in the Setup Linux and Raspberry Pi Section.
- `sampleRate` : The sample rate in Hz for the selected input device. 44100 is a common default but your system may be different.
- `frequencyScaleFactor` : 🚧 Should stay 1. If you need to adjust this then change your sampleRate instead
- `recordingScaleFactor` : The default value is 2. If your recording is not working or sounds distorted try changing this value to `1` . **Most users will NOT need to change the default value of 2**
- `silenceAmplitude` : This is the RMS (Root Mean Squad) amplitude threshold for detecting silence. Any RMS value under the threshold is considered silence. Default/Recommended: `0.05`
- `channels` : Unless a stereo audio input is being used keep the default value of `1` .

📝 Most users will not need to modify the `audio` defaults

Detection - Required

- `minRecordingLengthSec` : Recordings will be a minimum of this many seconds. Use a value that works for you. Keep in mind the larger this value, the longer it will take for Post Recording notifications to be sent.
- `defaultMatchThreshold` : This is the default number of samples needed for a frequency match to be completed. For example, a tone of 1000Hz with a `matchThreshold` of 6 requires that 6 samples match 1000Hz within the `tolerancePercent` to be considered a match. Adjust this lower to increase sensitivity (tones matched faster). Adjust this value higher to decrease sensitivity (tones matched slower). ⚠ A value too low may result in false positives while a value too high may result in legitimate tones being missed. **Recommended Value: Lowest possible value where no false positives are identified.**
- `defaultTolerancePercent` : The default plus/minus tolerance applied to a frequency to determine a match. Default is `0.05` or 5%. For example, if a tone of 1000Hz is specified with a `tolerancePercent` of `0.05` any detected sample with a frequency between 950Hz and 1050Hz is considered a match counting towards the `matchThreshold` for that tone.

📝 `defaultMatchThreshold` can be overridden at the `detector` level using `matchThreshold` . `defaultTolerancePercent` can be overridden at the `detector` level using `tolerancePercent`

Detectors - Array of values in Detection Section

The `detectors` property is the most important part of the configuration as it defines what tones to listen for and what notifications to send. Each entry in the array (or list) must match the format below. Use the bare minimum "Second Test Fire Department" in the `config/default.json`
as a starting point. There can be as many detectors in a single configuration file as needed.

ℹ Use different detectors to detect different departments or stations that have unique tone outs.

- `name` : User defined name of station, department, etc for the corresponding tones

- `tones` : An array of 1 to many tones in Hertz separated by commas. For example, if a department uses a two-tone system using tones 850Hz and 525Hz enter `[850, 525]` . If the system only uses a single 1000Hz tone enter `[1000]` . If the system is a 5 tone system utilizing tones 1000Hz, 750Hz, 1200Hz, 1500Hz, and 500Hz specify `[1000, 750, 1200m 1500, 500]` . ⚠ Speiifying a single tone is possible but can lead to false positives. For a single tone configuration it is recommended to use a `tolerancePercent` of `0.01` (1%) and `matchThreshold` greater than or equal to `35` .
- `matchThreshold` (Optional): If specified overrides the `defaultMatchThreshold` . If excluded the `defaultMatchThreshold` is used.
- `tolerancePercent` (Optional): If specified overrides the `defaultTolerancePercent` . If excluded the `defaultTolerancePercent` is used.

› Notifications

Notification settings are specified for each detector individually. The `notifications` property is made up of two sections that use the same format: `preRecording` and `postRecording` . `preRecording` notifications are sent as soon as the detector identifies a full match of all tones. `postRecording` notifications are sent after recording is completed and have access to the recorded file.

› Pushbullet

Pushbullet.com allows users to install the Pushbullet Android/iOs app and receive push notifications from FD Tone Notify. This is a *very fast* notification method. Pushbullet notifications are typically delivered within seconds making Pushbullet ideal for `preRecording` notifications.

- `title` : Any string that serves as the title for the push notification. Typically a department/station name
- `channelTag` : The channel tag to send the push notification to. **IMPORTANT Must provide a Pushbullet API key that owns the specified channel.**
- `body` : Body to include in the push notification. Can use `%d` to dynamically insert the date & time.

When Pushbullet notifications are specified in `postRecording` the push notification will include the recorded dispatch audio.

› Webhooks

Webhooks can be used to integrate FD Tone Notify into other software applications. Webhooks specified in the `preRecording` section will POST JSON to the `address` specified with the following format.

```
{
    timestamp: 1614996124315,
    tones: [1000, 2000],
    matchAverages: [1001, 2001],
    filename: "1614996124315.wav",
    recordingRelPath: "./1614996124315.wav",
    detectorName: "Webhook Test Detector",
    custom: {}
}
```

Webhooks specified in `postRecording` section will POST to the address with multi-part-form-data including the recorded file. The form data will also include all of the information above.

ⓘ For webhooks that require authentication FD Tone Notify supports populating headers with environment variables using the following format `CUSTOM_ENV_VAR_` followed by the name of the variable. For example, `CUSTOM_ENV_VAR_AUTH_HEADER` to subsitute the value for the variable `AUTH_HEADER` . If the specified variable is not set `null` is used.

› External Commands

External commands can be used to run any executable on the local machine. In the case of the Raspberry Pi an external command could be configured to utilize the hardware GPIO pins to interact with the environment such as activating station alerting systems.

- `command` : The full command to run. The following values can be used to add arguments. `[timestamp]` , `[detectorName]` , `[description]` , `[tones]` , `[matchAverages]` , `[recordingRelPath]` amd `[custom]` . Example:

```
    node ./rel/path/main.js [timestamp] \"[detectorName]\" \"[description]\" [tones] [matchAverages]
[recordingRelPath] [custom]
```

- `description` : A description that will be used while logging
- `custom` (Optional): This value will be stringified to JSON and passed as the `[custom]` argument

Emails

Emails can be sent both for `preRecording` and `postRecording` notifications. Emails sent `postRecording` include the dispatch recording as an attachment. Email notifications can also be used to send text message notifications via carrier SMS gateways. For more information on SMS Gateways see Wikipedia

i Due to the nature of email, notifications may take several seconds to be delivered depending on the email provider/SMS Gateway.

i In order to be able to send email notifications it is necessary to configure several email secrets. The `email` section in the config file must also be complete. See [Email](#Email - Required for Email Notifications)

- 'to': A comma separated list of emails. Example: `5555555555@mms.att.net,test@example.com,person@example.com`
- `bcc` : A comma separated list of emails to bcc. See example above
- `subject` : The subject line of the email notification
- `text` : The text body of the email. HTML not supported. Can use `%d` to insert date

i Multiple emails notification objects can be listed. Use this to send notifications with different subjects. Otherwise list recipients in the `to` or `bcc` fields.

i To get started quickly check out SendGrid. Setup is free and easy. The free plan should be sufficient for most.

Example Full Notifications Configuration

```
{
             "webhooks": [
               {
                 "address": "http://localhost:8500/endpoint",
                 "headers": {"custom-header": "value", "from-env-var": "CUSTOM_ENV_VAR_AUTH_HEADER", "note":
"from-env-var set to value of 'AUTH_HEADER' environment var or null if the variable is not set"},
                 "custom": {
                   "anyObject": true
                 }
               },
               {
                 "address": "http://localhost:8500/other",
               }
             ],
             "externalCommands": [
               {
                 "command": "node ./rel/path/main.js [timestamp] \"[detectorName]\" \"[description]\" [tones]
[matchAverages] [recordingRelPath] [custom]",
                 "description": "fancy description for the logs",
                 "custom": {
                   "anyObject": true,
                   "notes": "This object will be stringified to JSON and passed as a command line argument"
                 }
               }
             ],
             "emails": [
               {
                 "to": "test@example.com,user@domain.com",
                 "bcc": "person@example.com,name@domain.com",
                 "subject": "Test Fire Department Tone Received",
                 "text": "Tone Received %d"
               },
               {
                 "to": "other@example.com,people@domain.com",
                 "bcc": "john.doe@example.com,jane.doe@domain.com",
                 "subject": "Test Fire Department Tone Received",
                 "text": "Tone Received %d"
               }
             ]
           }
```

Coralogix - Optional

Configure this section to use Coralogix for remote logging.

Email - Required for Email Notifications

This section is required to enable the sending of Email notifications.

- `from` : The email address to send email from. For Sendgrid this email *must* be configured & validated in the account.
- `host` : SMTP host for sending email. For Sendgrid use `smtp.sendgrid.net` .
- `port` : The SMTP port for sending email notifications. `465` for Sendgrid secure
- `secure` : Please don't set this to false. Make sure to use the port corresponding to your secure setting.

i It is also necessary to configure Email secrets. See Environment Variables and Secrets

Command Line Options

```
Usage: fd-tone-notify [options]

Options:
  --tone-detector        Secondary functionality: Instead of reading the config file and sending notifications when
specific tones are detected this option activates multi tone detector mode. In this mode the
                         frequency spectrum from 300Hz to 4000Hz is monitored. When a multi tone is detected the
result is logged to the console. Use this mode to determine the frequencies to monitor and
                         enter the results in the "tones" parameter for the corresponding department.
  --debug                Overrides FD_LOG_LEVEL environment var forcing the log level to debug
  --silly                Overrides FD_LOG_LEVEL environment var forcing the log level to silly
  --instance-name        Overrides NODE_APP_INSTANCE environment allowing different config files for different
instances running on the same machine. Example: "--instance-name my-fd" will load config files
                         default-my-fd.json and local-my-fd.json
  --web-server           Starts the webserver. The webserver provides remote monitoring capability and ability to
listen to live audio
  --port <port>          Overrides FD_PORT environment var setting the port for the web server. Has no effect
without --web-server option. Default port 3000
  --secrets-file <path>  Path to secrets file. By default secrets will be loaded from config/secrets.json. Use this
option to specify a different path
  --force-secrets-file   Using this option forces all secrets to be read from the secrets file (Either the default
or the path specified by --secrets-path). Values from environment variables will be
                         disregarded. If the file cannot be loaded or parsed the application will exit with code 99
indicating an invalid secrets configuration.
  -h, --help             display help for command
```

Environment Variables and Secrets

Secrets can either be loaded from environment variables or a static secrets file. By default the secrets file is located at `./config/secrets.json` . For most users adding values to the `secrets.json` is the easiest method. The same secret names are used as environment and in the `secrets.json` . ℹ Remember to rename `secrets.template.json` to `secrets.json`

- `FD_PUSHBULLET_API_KEY` : The API key for Pushbullet. This API key must be the owner/creator of the channel tags used to send notifications. After setting up a Pushbullet account use the Create Access Token button on the Account Page to get an API Key
- `FD_CORALOGIX_PRIVATE_KEY` : If using Coralogix enter the Private Key here
- `FD_SMTP_USERNAME` : The SMTP username. For Sendgrid this will always be `apikey`
- `FD_SMTP_PASSWORD` : The SMTP password. For Sendgrid this an API Key. Sendgrid API keys can be created here.

ℹ Enviornment variables override valus set by secrets file. To change this behavior use the `--force-secrets-file` option.

ℹ An alternative secrets file can be specified at runtime using the `--secrets-file <path>` option.

💡 To help debug the secrets that are loaded use the `--silly` option. When starting FD Tone Notify the source of each secret will be logged to the console.