
Newfies-Dialer Documentation

Release 1.0.0

Arezqui Belaid

November 21, 2011

CONTENTS

1	Getting Started	3
1.1	Introduction	3
2	Installation	7
2.1	Installation overview	7
2.2	Broker Installation	9
2.3	Celery Installation	11
3	User Guide	13
3.1	Overview	13
3.2	How to use Newfies-Dialer	13
3.3	Admin Panel	20
3.4	Customer Panel	30
4	Configuration and Defaults	39
4.1	Sample Configuration	39
4.2	Celery Configuration	42
5	FreeSwitch Installation and Configuration	45
5.1	Freeswitch Installation and configuration	45
5.2	Plivo Installation and configuration	45
5.3	Freeswitch Trunk configuration	46
6	Developer doc	49
6.1	Prerequisites	49
6.2	Coding Style & Structure	49
6.3	Objects Description	50
6.4	Database Design	56
6.5	Newfies-Dialer Views	56
6.6	Newfies-Dialer Admin Views	61
6.7	Newfies Tasks	64
6.8	Newfies Signals	64
6.9	Test Case Descriptions	64
7	API Reference	69
7.1	CampaignResource	69
7.2	CampaignDeleteCascadeResource	72
7.3	PhonebookResource	73
7.4	BulkContactResource	75
7.5	CampaignSubscriberResource	75

7.6	CallrequestResource	76
7.7	CdrResource	78
7.8	AnswercallResource	79
7.9	HangupcallResource	79
8	Contributing	81
8.1	Community Code of Conduct	81
8.2	Reporting a Bug	82
8.3	Coding Style	83
9	Frequently Asked Questions	85
9.1	General	85
9.2	Misconceptions	86
10	Troubleshooting	87
10.1	Where to find help	87
10.2	Where to find the log files	87
10.3	How to run a quick test call	88
10.4	Run in debug mode	88
10.5	Celerymon	88
10.6	How to discard all pending tasks	89
10.7	Checking Plivo is running	89
10.8	Checking Freeswitch	89
10.9	Step By Step Checklist	89
11	Resources	91
11.1	Getting Help	91
11.2	Bug tracker	91
11.3	Wiki	91
11.4	Contributing	91
11.5	License	92
12	Indices and tables	93
	Python Module Index	95
	Index	97

Release 1.0

Date November 21, 2011

Contents:

GETTING STARTED

Contents:

1.1 Introduction

Web <http://www.newfies-dialer.org/>

Download <http://www.newfies-dialer.org/download/>

Source <https://github.com/Star2Billing/newfies-dialer/>

Keywords dialer, voip, freeswitch, django, asynchronous, rabbitmq, redis, python, distributed

– Newfies is an open source VoIP Dialer based on distributed message passing. It has been built to support cloud based servers and can also work on standalone servers. It uses [Freeswitch](#) (VoIP Server) to outbound calls, but support for other VoIP Servers such as [Asterisk](#) could be easily added in the future. The platform is focused on real-time operations and task call distributions to clustered brokers and workers.

Newfies-Dialer is written in Python, using the [Django](#) Framework. It also operates with message brokers such as [RabbitMQ](#), [Redis](#) but support for Beanstalk, MongoDB, CouchDB and DBMS is also available.

Newfies-Dialer provides an extensive set of APIs to easily integrate with third-party applications.

Using very simple steps, Newfies-Dialer will help you create campaigns, add phonebooks, contacts, build audio messages and create complex telephony applications. Once your campaigns are ready to start, your messages will be dispatched and delivered.

- [Overview](#)
- [Utility](#)
- [Features](#)
- [Architecture](#)
- [Latest documentation](#)

1.1.1 Overview

Newfies-Dialer can be installed and used by anyone who has a need for mass outbound calling, voice broadcasting or providing outbound IVR. Some of the potential uses for Newfies-Dialer are listed below.

The system may be installed and used by either companies who wish to make calls on their own behalf, or by SaaS (Software as a Service) companies that want to provide bulk dialling facilities to their own customers.

1.1.2 Utility

Newfies-Dialer is loaded up with a list of telephone numbers that can be dialled sequentially at very high rates of calling depending on carrier capacity and hardware, potentially delivering many millions of calls per day.

When the called party answers the call, Newfies-Dialer passes the call to a telephony application that is custom designed to provide the desired behaviour.

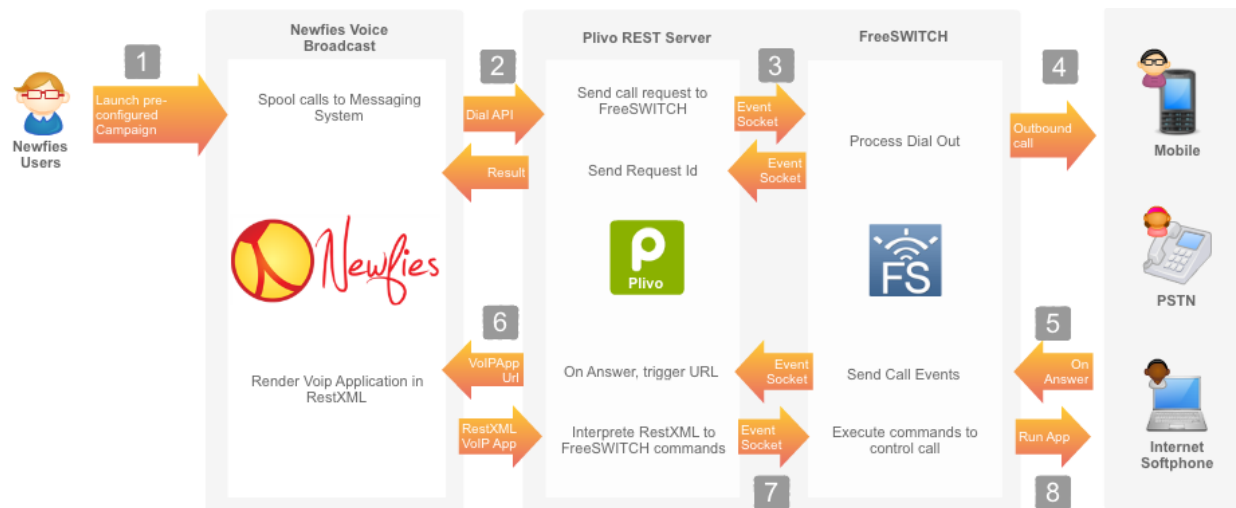
Below are examples of some of the uses for Newfies-Dialer

- **Telecasting:** Broadcast marketing or informational messages to customers and clients.
- **Telemarketing:** Broadcast a marketing message to potential customers, and give them the option to be put through to a call-centre via an IVR (Interactive Voice Response) Menu.
- **Phone Polling, Surveys and Voting:** Ring large numbers of people and present IVR options for either polling their opinions, interactive surveys, or taking their vote and record the results.
- **Debt Control:** Customers can be automatically reminded at intervals that they owe money, and an IVR menu presented to talk to the finance department or passed to a credit card capture IVR to pay over the phone.
- **Appointment reminders:** Doctors, Dentists, and other organisations that make appointments for their clients can integrate Newfies-Dialer into their appointment systems to pass a message reminding them of an upcoming appointment.
- **Dissemination of information via phone:** Newfies-Dialer was originally designed to call large numbers of people and disseminate medical and health advice via the ubiquitous cellphone in 3rd world countries where often, literacy levels are low. On a local scale, it can be used to disseminate information about forthcoming community events.
- **Mass Emergency broadcast:** Where there is a necessity to warn large numbers of people in a short space of time, such as weather warnings.
- **Voice Conferencing:** Attendees for a voice conference or podcast can be dialled up from a central location, and be connected in an audio conference room.
- **Subscription Reminders and Renewals:** Where a company sells an annual subscription for a product or service, Newfies-Dialer can be configured to dial the customer, remind them that the subscription is due, and optionally pass the call into a call centre or into a credit card payment IVR.

1.1.3 Features

Telephony PBX	Based on leading open source Freeswitch, Asterisk
Distributed	Runs on one or more machines. Supports broker <i>clustering</i> and <i>HA</i> when used in combination with <i>RabbitMQ</i> . You can set up new workers without central configuration (e.g. use your grandma's laptop to help if the queue is temporarily congested).
Concurrency	Throttle Concurrent Calls
Scheduling	Supports recurring tasks like cron, or specifying an exact date or countdown for when the task should be executed. Can re-try to the non connected numbers at a later time
IVR support	Accommodates multiple IVR scripts with options to connect the user to some other IVR/phone number on pressing a key
Web Interface	Newfies can be managed via a Web interface. Realtime web-based reports for call details and current calls. You can query status and results via URLs, enabling the ability to poll task status using Ajax.
Error Emails	Can be configured to send emails to the administrator if a tasks fails.
Import Contact	Import contact details from a .csv file

1.1.4 Architecture



- User selects contacts, phonebooks and campaigns, then chooses a voice application to use. The campaign is then launched
- Newfies-Dialer spools the outbound calls to FreeSWITCH via Plivo.
- Plivo sends the dial request to FreeSWITCH using the event socket.
- FreeSWITCH dials the contact via the configured telephony gateways.
- Contact picks up the call, and the answer event is received in FreeSWITCH and passed back to Plivo.
- Newfies-Dialer is notified that the call is answered, then renders & relays RestXML to Plivo.
- Plivo interprets RestXML and sends the application call-flow commands to FreeSWITCH.
- The voice application is delivered to the contact by FreeSWITCH.

1.1.5 Latest documentation

The [latest documentation](#) with user guides, tutorials and API reference is hosted at “Readthedocs”.

INSTALLATION

Contents:

2.1 Installation overview

2.1.1 Install requirements

A Requirements file gives you a way to create an environment where you can put all optional dependencies which are needed for your Project/Application.

To get started with Newfies-Dialer you must have the following installed:

- python >= 2.4 (programming language)
- Apache / http server with WSGI modules
- Django Framework >= 1.3 (Python based Web framework)
- Celery >= 2.2 (Asynchronous task queue/job queue based on distributed message passing)
- MySQL-python >= 1.2.3 (MySQL for python language)
- Werkzeug >= 0.6.2 (Collection of various utilities for WSGI applications)
- amqpplib >= 0.6.1 (Client library for AMQP)
- anyjson >= 0.3 (Loads the fastest JSON module)
- django-celery >= 2.2.4 (Celery integration for Django)
- django-extensions >= 0.6 (Collection of global custom management extensions for Django)
- django-jsonfield >= 0.6 (Reusable django field that can use inside models)
- django-pagination >= 1.0.7 (Utilities for creating robust pagination tools throughout a django application)
- django-picklefield >= 0.1.9 (Implementation of a pickled object field)
- django-threaded-multihost >= 1.4-0 (Provides multi-host utilities to Django projects)
- django-urlauth = 0.1.1 (Allows to build links with authentication effect)
- django-uuidfield >= 0.2 (Provides a UUIDField for your Django models)
- django-reusableapps >= 0.1.1 (Python module to enable Django to load reusable, pluggable and egg-based applications)
- docutils >= 0.7 (Text processing system for processing plaintext documentation into useful formats)

- importlib >= 1.0.2 (Implementation of the *import* statement)
- kombu >= 1.0.2 (An AMQP - Advanced Message Queuing Protocol messaging framework for Python)
- pyparsing >= 1.5.5 (A general parsing module for Python)
- python-dateutil >= 1.5 (Extensions to the standard datetime module)
- redis >= 2.2.2 (Redis Python Client)
- simplejson >= 2.1.3 (Simple, fast, complete, correct and extensible JSON)
- uuid >= 1.30 (UUID object and generation functions)
- wsgiref >= 0.1.2 (Validation support for WSGI)
- switch2bill-common (Common library that are reused by Star2Billing)
- simu-prefix-country (Provide Prefix and Country information)
- django-tastypie (Creating delicious APIs for Django)
- BeautifulSoup >= 3.2.0 (HTML parser optimized for screen-scraping)
- Pygments >= 1.4 (A generic syntax highlighter)
- django-admin-tools (Collection of tools for the django administration)
- python-memcached >= 1.47 (Python based API for communicating with the memcached distributed memory object cache daemon)
- django-memcache-status >= 1.0.1 (Displays statistics about memcached instances)
- django-notification >= 0.1.3 (User notification management for the Django web framework)
- identicon (identicon python implementation)
- django-sentry >= 1.8.6.2 (Real-time logging / error tracing for Django)
- django-qstats >= 0.3.1 (A django microframework that eases the generation of aggregate data for queriesets)

Use PIP to install all the requirements,:

```
$ pip install -r requirements.txt
```

2.1.2 Installation Script

You can install Newfies-Dialer manually or using the shell script provided.

To install Newfies-Dialer using the script,:

```
$ chmod +x install/install-newfies.sh
```

```
$ ./install/install-newfies.sh
```

```
$ chmod +x install/install-celery.sh
```

```
$ ./install/install-celery.sh
```

2.1.3 Running a Newfies-Dialer

Inside Newfies-Dialer directory you should run:

```
$ mkdir database
$ python manage.py syncdb
$ python manage.py collectstatic
$ python manage.py runserver
```

`syncdb` will create a database named `test.db` in `database` folder of the Newfies-Dialer directory. We have configured Newfies-Dialer to do this, but you can change this simply by modifying `settings.py` where `DATABASES` dictionary is constructed. You can find more information about this in the Django documentation.

`collectstatic` will fetch all necessary media files and put them into `static` folder defined in the `settings` module.

`runserver` runs an embedded webserver to test your site. By default it will run on <http://localhost:8000>. This is configurable and more information can be found on `runserver` in Django documentation.

2.1.4 Caching System

When a User requests a page, the Web server makes calculations for business logic and to create the page that your visitor sees. It creates a processing overhead higher than a standard read-a-file-off-the-filesystem server arrangement.

This is where caching comes in.

To cache something is to save the result of an expensive calculation so that you don't have to perform the calculation next time.

```
$ mkdir /usr/share/django_cache
```

2.2 Broker Installation

This document describes the installation of two different Brokers. One is Redis and second is Rabbitmq. You can install either to work with Newfies-Dialer.

2.2.1 Redis

Download Source

Download : [redis-server_2.0.0~rc2-1_amd64.deb](#).

To install Redis-Server

```
$ sudo dpkg -i redis-server_2.0.0~rc2-1_amd64.deb
```

or you can use `apt-get`

```
$ apt-get install redis-server
```

Running Server

```
$ redis-server
```

2.2.2 Rabbitmq

RabbitMQ is a complex and sophisticated product. If you don't need this level of robustness, then you might want to take a look at Redis - it installs easily, runs relatively lean, and can be monitored and maintained without a lot of fuss.

See [Installing RabbitMQ](#) over at RabbitMQ's website.

Note: If you're getting *nodedown* errors after installing and using **rabbitmqctl** then this blog post can help you identify the source of the problem:

<http://somic.org/2009/02/19/on-rabbitmqctl-and-badrpcnodedown/>

Download Source

<http://www.rabbitmq.com/server.html>

Debian APT repository

To make use of the RabbitMQ APT repository,

1. Add the following line to your `/etc/apt/sources.list`

```
deb http://www.rabbitmq.com/debian/ testing main
```

Note: The word **testing** in the above line refers to the state of the release of RabbitMQ, not any particular Debian distribution. You can use it with Debian stable, testing or unstable, as well as with Ubuntu. In the future there will be a stable release of RabbitMQ in the repository.

2. (optional) To avoid warnings about unsigned packages, add RabbitMQ's public key to your trusted key list using `apt-key(8)`

```
$ wget http://www.rabbitmq.com/rabbitmq-signing-key-public.asc
```

```
$ sudo apt-key add rabbitmq-signing-key-public.asc
```

3. Run `apt-get update`.
 4. Install packages as usual; for instance,
- ```
$ sudo apt-get install rabbitmq-server
```

## Setting up RabbitMQ

To use celery we need to create a RabbitMQ user, a virtual host and allow that user access to that virtual host:

```
$ rabbitmqctl add_user myuser mypassword
$ rabbitmqctl add_vhost myvhost
$ rabbitmqctl set_permissions -p myvhost myuser ".*" ".*" ".*"
```

See the [RabbitMQ Admin Guide](#) for more information about [access control](#).

## Starting/Stopping the RabbitMQ server

To start the server:

```
$ sudo rabbitmq-server
```

you can also run it in the background by adding the *-detached* option (note: only one dash):

```
$ sudo rabbitmq-server -detached
```

Never use **kill** to stop the RabbitMQ server, but rather use the **rabbitmqctl** command:

```
$ sudo rabbitmqctl stop
```

When the server is running, you can continue reading [Setting up RabbitMQ](#).

## 2.3 Celery Installation

### 2.3.1 Celery

Celery is an asynchronous task queue/job queue based on distributed message passing. It is focused on real-time operation, but supports scheduling as well.

You can install Celery either via the Python Package Index (PyPI) or from source.

#### To install using pip

```
$ pip install Celery
```

#### To install using easy\_install

```
$ easy_install Celery
```

#### Downloading and installing from source

To Download the latest version [click here](#).

You can install it by doing the following:

```
$ tar xvfz celery-0.0.0.tar.gz
$ cd celery-0.0.0
$ python setup.py build
```

```
$ python setup.py install # as root
```

### **Using the development version**

You can clone the repository by doing the following:

```
$ git clone git://github.com/ask/celery.git
```



# USER GUIDE

Contents:

## 3.1 Overview

Newfies-Dialer is an open source VoIP Dialer based on distributed message passing. It has been built to support cloud servers and also works on standalone servers. It uses Freeswitch (VoIP Server) to outbound calls, but support for other VoIP Servers such as Asterisk could be easily added in the future. The platform is focused on real-time operations and task call distributions to clustered brokers and workers.

Newfies-Dialer is a computerised system that automatically dials a group of telephone numbers for connection to assigned campaigns.

### Features:

- \* Restful-API based to easily integrate the platform dialer with third-party applications
- \* Web-based administrative/customer interfaces
- \* Lower operating costs
- \* Calls are made through Internet VoIP. No need for telephony hardware
- \* Call reports and Statistics

## 3.2 How to use Newfies-Dialer

### 3.2.1 Freeswitch Set-Up

Configure trunks and gateways in Freeswitch by creating an XML file in `/usr/local/freeswitch/conf/sip_profiles/external/` and give it an identifiable name, e.g. `call-labs.xml`, and place the following lines in the file, edited to suit your provider:

```
<include>
<gateway name="ip address or hostname of carrier">
<!--// account username *required* //-->
<param name="username" value="your username provided by carrier"/>
<!--// auth realm: *optional* same as gateway name, if blank //-->
<!--<param name="realm" value="asterlink.com"/>-->
<!--// username to use in from: *optional* same as username, if blank //-->
<param name="from-user" value="your username provided by carrier"/>
<!--// domain to use in from: *optional* same as realm, if blank //-->
<!--param name="from-domain" value=""/-->
<!--// account password *required* //-->
```

```
<param name="password" value="your password supplied by carrier"/>
<!--/// extension for inbound calls: *optional* same as username, if blank ///-->
<!--<param name="extension" value="cluecon"/>-->
<!--/// proxy host: *optional* same as realm, if blank ///-->
<!--<param name="proxy" value="asterlink.com"/>-->
<!--/// send register to this proxy: *optional* same as proxy, if blank ///-->
<!--<param name="register-proxy" value="mysbc.com"/>-->
<!--/// expire in seconds: *optional* 3600, if blank ///-->
<!--<param name="expire-seconds" value="60"/>-->
<!--/// do not register ///-->
<param name="register" value="true"/>
<!-- which transport to use for register -->
<!--<param name="register-transport" value="udp"/>-->
<!--How many seconds before a retry when a failure or timeout occurs -->
<!--<param name="retry-seconds" value="30"/>-->
<!--Use the callerid of an inbound call in the from field on outbound calls via this gateway -->
<!--<param name="caller-id-in-from" value="false"/>-->
<!--extra sip params to send in the contact-->
<!--<param name="contact-params" value="tport=tcp"/>-->
<!--send an options ping every x seconds, failure will unregister and/or mark it down-->
<!--<param name="ping" value="25"/>-->
</gateway>
</include>
```

Then in the Freeswitch CLI (fs\_cli) “sofia profile external restart reloadxml”. When the command is complete, check the gateway has registered with the command “sofia status”.

### 3.2.2 Create Gateway

Having created the gateway in Freeswitch, Newfies-Dialer has to be told that it can use it. In admin, add a new dialer gateway, e.g. sofia/gateway/myprovider/ (The final / is important) where “myprovider” is the gateway name setting used in above xml file in Freeswitch.

Only the fields in bold are compulsory.

### 3.2.3 Dialer Settings

Dialer settings are mapped with system users who are going to create campaigns & contacts. If dialer settings are not mapped to users, notifications & emails are sent to super admin user.

To create restrictions (like the Max. no of campaign, Max no of contacts etc.) for system User, Click on Add dialer settings. Add numeric values for the limit.

To apply the dialer settings limit on a system user, click on Customers or Admins in admin UI, select the user to update, & apply the settings from the dialer settings list.

### 3.2.4 Create Voice Application

A number of voice applications are provided with Newfies-Dialer. Click Add VoIP App give the voice application a name, select the type of application from the dropdown, select the gateway to use if the call is to be redirected, and provide the data to be used, e.g. in the case of “Speak” this would be the words to convert to text to speech.

### 3.2.5 Create call list

To create a call list, click on Add in Phonebook list, add name of phonebook & its description. Click on Contacts and add phone numbers in the contact list. You can also import your call list from csv files, via clicking on Import contact.

[Update Phonebook](#)

**Phonebook**

**Name:**

**Description:**

[Phonebook Notes](#)

**Contact Number:**  ☒ Equals ☐ Begins with ☐ Contains ☐ Ends with

**Contact Name:**

**Phonebook:**

**Status:** ☐ Inactive ☐ Active ☒ All

**Contacts**

<input type="checkbox"/>	Id	Phonebook	Contact	Last Name	First Name	Status	Date	Action
<input type="checkbox"/>	90	Default_Phonebook	640234000	Belaid	Arezqui	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	91	Default_Phonebook	640234001	Fourth	John	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	92	Default_Phonebook	640234002	J. Lowrey	Bill	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	93	Default_Phonebook	640234003	G. Cartwright	Ebony	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	94	Default_Phonebook	640234004	K. Conley	Stephen	ACTIVE	2011-06-10 09:27:28	
<input type="checkbox"/>	95	Default_Phonebook	640234005	Burke	Linda	ACTIVE	2011-06-10 09:27:28	
<input type="checkbox"/>	96	Default_Phonebook	640234006	Mildred	Peacock	ACTIVE	2011-06-10 09:27:28	
<input type="checkbox"/>	97	Default_Phonebook	640234007	Parker	Donald	ACTIVE	2011-06-10 09:27:28	

10 Page 1 of 2 Displaying 1 to 10 of 20 items

Sample File						
contact	last name	first name	email-id	description	status	additional_vars
1234	Belaid	Arezqui	areski@gmail.com	test subscriber	1	test
5678	Fourth	John	john@gmail.com	test subscriber	0	test

SUBSCRIBER STATUS = ('1'-ACTIVE) ('0'-INACTIVE)

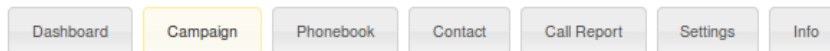
**Import Contact**

**Phonebook:** Default\_Phonebook   
Select Phonebook

**Upload CSV File :**     
Browse CSV file

### 3.2.6 Create campaign











To create a campaign, click on Add in campaign list, add details for the campaign. Important: Add the campaign's start and end dates with times & week-day exceptions. Select the gateway through which calls will be routed & the phonebook(s) linked with the contacts.





Campaigns

Add



Delete Selected


	Id	Name	Start date	End date	Aleg-gateway	Answer url	Contacts	Action
<input type="checkbox"/>	1	Default_Campaign	2011-04-08 18:31:11	2011-04-08 18:31:12	Default_Gateway		10	    
<input type="checkbox"/>	5	my campaign	2011-06-10 07:18:48	2011-06-17 07:18:48	Default_Gateway		10	    

10



Page 1 of 1





Displaying 1 to 2 of 2 items

Update Campaign

**Campaign**

**Name:**

**Description:**   
Short description of the Campaign

**CallerID:**   
CallerID used to call the A-Leg

**Status:**

**A-Leg Gateway:**   
Select gateway to use for this campaign

**VoIP Application:**   
Select VoIP Application to use with this campaign

**Extra Parameters:**   
Additional application parameters.

**Phonebook:**   
Hold down "Control", or "Command" on a Mac, to select more than one.

**Campaign settings**

**Frequency:**   
Define the frequency, speed of the campaign. This is the number of calls per minute.

**Call Max Duration:**   
Define the call's duration maximum. (Value in seconds 1800 = 30 minutes)

**Max Retries:**   
Define the max retry allowed per user.

**Time between Retries:**   
Define the time to wait between retries in seconds

**Timeout on Call:**   
Define the amount of second to timeout on calls

**Campaign schedule**

**Start:**   
Date Format: YYYY-mm-DD HH:MM:SS

**Daily start time:**   
Time Format: HH:MM:SS

**Finish:**   
Date Format: YYYY-mm-DD HH:MM:SS

**Daily stop time:**   
Time Format: HH:MM:SS

**Week Days:** ☒ Monday ☒ Tuesday ☒ Wednesday ☒ Thursday ☒ Friday ☒ Saturday ☒ Sunday

## 3.2.7 Dialer Settings

Dialer settings has to be mapped with system users who are going to create campaigns & contacts. If dialer setting is not mapped with system users, notification & mail has been sent to Super admin user.

To create restrictions (like the Max. no of campaign, Max no of contacts etc.) for system User, Click on Add in dialer settings list of admin side. Add numeric values for the limit.

To apply the dialer settings limit on a system user, click on Customers or Admins in admin UI, select the user to update, & apply the settings from the dialer settings list.

### Select Dialer Setting to change

Select Dialer Setting to change

Add Dialer Setting +

Q

Search

Action:

Go

0 of 1 selected

<input type="checkbox"/>	Name	Max frequency	Call Max Duration	Max Retries	Timeout on Call	Max number campaign	Max number subscriber campaign	Blacklist	Whitelist	Updated date
<input type="checkbox"/>	Default_Dialer_Setting	100	3	3	45	2	3	*	12345*	Apr# 29, 2011, 8:43 a.m.

1 Dialer Setting

### Change Dialer Setting

[History](#)

**Name:**   
Define a name for this set of settings

**Max frequency:**   
Define the Max frequency, speed of the campaign. This is the number of calls per minute.

**Call Max Duration:**   
Define the max retry allowed.

**Max Retries:**   
Define the max retry allowed per user.

**Timeout on Call:**   
Define the maximum amount of second to timeout on calls

**Max number campaign:**   
Max Number of campaign

**Max number subscriber campaign:**   
Max Number of subscriber per campaign

**Blacklist:**Define Regular Expression of phonenumber you want to forbid. This is used to blacklist phonenumber to be called. Example "[2-4][1]\*" will forbid all the phonenumber starting by 2,3 or 4 and followed by 1

**Whitelist:**

## 3.2.8 Reach to contacts/subscribers

A call-request will spool a call directly from the platform using a dialer gateway and update the call-request status after receiving a response from the gateway.

## Change Call Request

History

Standard options	
Uniqueid:	<input type="text" value="2342jdst-00123"/>
Campaign:	<input type="text" value=""/> Select Campaign
Callback time:	Date: <input type="text" value="2011-05-01"/> Today Time: <input type="text" value="11:22:33"/> Now
Status:	<input type="text" value="SUCCESS"/>
Exten:	<input type="text" value="1231321"/>
Context:	<input type="text" value="mycontext"/>
Application:	<input type="text"/>
Timeout:	<input type="text" value="30000"/>
Callerid:	<input type="text" value="650784355"/>
Advanced options <small>Hide</small>	
Call Request Type:	<input type="text" value="ORIGINAL"/>
A-Leg Gateway:	<input type="text" value=""/> Select Gateway to use to reach the subscriber
VoIP Application:	<input type="text" value=""/> Select VoIP Application to use with this campaign
VoIP App Data:	<input type="text"/> Define the additional data to pass to the application
Campaign Subscriber:	<input type="text"/> Campaign Subscriber related to this callrequest

### 3.2.9 VoIP Call Report

As per the status of a call-request, it will be stored in the VoIP call records. This gives information of all the calls & call statistics made with the call-request and also you can search for records on the basis of date range. You can export the VoIP call report into a csv file.

From:   
Date Format: YYYY-MM-DD

To:   
Date Format: YYYY-MM-DD

Disposition :

Call Records									
Start date	Call ID	Leg	Caller ID	Phone No.	Gateway	Duration	Bill Sec	Disposition	Hangup C
2011-11-15 23:20:40	123456789	A-Leg	435435		Default_Gateway	3	0		
2011-11-16 07:15:59	123456789	A-Leg	123123		Default_Gateway	3	0	ANSWER	

10 Page 1 of 1 Displaying 1 to 2 of 2 items

Call Detail Report Summary				
Date	Duration	Graphic	Calls	ACT
Total	00:00		0	00:00

Export CSV file



### 3.2.10 Settings

The settings page provides a number of functions:-

Account - Change the detail of the account. Password - Change the password. Notifications - Display system notifications. Limitation - Displays the parameters of the dialer settings. Authorized - Facitly to check that a number is not blacklisted.

## 3.3 Admin Panel

<http://localhost:8000/admin/>

This interface provides user (ACL) management, full control of all Campaigns, Phonebooks, Subscribers, Gateways and configuration of the Audio Application.

- Screenshot with Features



### 3.3.1 Screenshot with Features

#### Dashboard

Dashboard page for the admin interface after successful login with superuser credentials

The screenshot displays the Newfies-Dialer Admin Dashboard. At the top, there is a navigation bar with the following links: **NEWFIES DASHBOARD**, **BOOKMARKS**, **APPLICATIONS**, **ADMINISTRATION**, and **CUSTOMER PANEL**. Below the navigation bar, the dashboard is organized into several panels:

- Administration**: Contains a sub-panel **Auth** with links for **Admins**, **Customers**, and **Groups**, each with **Add** and **Change** options.
- Settings**: Contains a sub-panel **Dialer\_Settings** with a **Dialer Setting** link, also with **Add** and **Change** options.
- Task Manager**: Contains a sub-panel **Djcelery** with links for **Crontabs**, **Intervals**, **Periodic tasks**, **Tasks**, and **Workers**, each with **Add** and **Change** options.
- Voip Dialer**: Contains a sub-panel **Dialer\_Campaign** with links for **Campaign Subscribers**, **Campaigns**, **Contacts**, and **Phonebooks**, each with **Add** and **Change** options. Below this is a sub-panel **Dialer\_Cdr** with links for **Call Requests** and **VoIP Call**, and a sub-panel **Dialer\_Gateway** with a **Dialer Gateways** link, all with **Add** and **Change** options.
- Voip Server**: Contains a sub-panel **Voip\_App** with a **VoIP Applications** link, and a sub-panel **Voip\_Server** with links for **VoIP Server Groups** and **VoIP Servers**, each with **Add** and **Change** options.
- Quick links**: Contains links for **Go to Newfies**, **Go to FreedomFone**, **Change password**, and **Log out**.
- DialCode**: Contains a sub-panel **Prefix\_Country** with links for **Countries** and **Prefixes**, each with **Add** and **Change** options.
- Recent Actions**: A list of recent actions, including **Campaign Subscriber 19**, **Contact 6407444581**, **Contact 607451232**, **Contact 3467346457**, and **Contact 650784355 (donkiki)**, all dated **April 25, 2011**.
- memcached status**: A bar at the bottom right showing **memcached: 127.0.0.1:11211 (1) - 0% load**.

#### Phonebook

The phonebook list will be displayed from the following URL. You can add a new phonebook by clicking **Add phonebook** and adding the name of the phonebook and its description, Also from the phonebook list, click on the phonebook that you want to update.

**URL:**

- [http://localhost:8000/admin/dialer\\_campaign/phonebook/](http://localhost:8000/admin/dialer_campaign/phonebook/)

**Select Phonebook to change** Add Phonebook +

Action:  Go 0 of 1 selected

ID	Name	Description	User	Date	Contacts
<input type="checkbox"/> 1	Default_Phonebook	This is default phone book	areski	April 8, 2011, 7:55 a.m.	3

1 Phonebook

Filter

By user:  All

By Date:  Any date

To Add/Update phonebook for a user

URL:

- [http://localhost:8000/admin/dialer\\_campaign/phonebook/add/](http://localhost:8000/admin/dialer_campaign/phonebook/add/)
- [http://localhost:8000/admin/dialer\\_campaign/phonebook/1/](http://localhost:8000/admin/dialer_campaign/phonebook/1/)

**Change Phonebook** History

Name:  Default\_Phonebook

Description:  This is default phone book

Short description about the Phonebook

User:  areski

## Contact

The contact list will be displayed from the following URL and you can add a new contact by clicking Add contact & adding the contact details (i.e. phone number, name, description about contact, contact status) to one phonebook from the phonebook list.

If the contact is active and the linked phonebook is also attached to a running campaign, then the contact will be added into campaign subscribers.

From the contact list, click on the contact that you want to update.

URL:

- [http://localhost:8000/admin/dialer\\_campaign/contact/](http://localhost:8000/admin/dialer_campaign/contact/)

**Select Contact to change** Add Contact + Import Contacts +

Action:  Go 0 of 3 selected

ID	Phonebook	Contact	Name	Status	Date
<input type="checkbox"/> 1	Default_Phonebook	650784355	Belaid Avezqui	ACTIVE	April 29, 2011, 8:44 a.m.
<input type="checkbox"/> 2	Default_Phonebook	650153589	Fourth John	INACTIVE	April 29, 2011, 8:44 a.m.
<input type="checkbox"/> 3	Default_Phonebook	650483459	Belaid Marta	INACTIVE	April 29, 2011, 8:44 a.m.

3 Contacts

Filter

By Date:  Any date




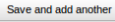
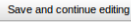

To Add/Update a contact

URL:

- [http://localhost:8000/admin/dialer\\_campaign/contact/add/](http://localhost:8000/admin/dialer_campaign/contact/add/)
- [http://localhost:8000/admin/dialer\\_campaign/contact/1/](http://localhost:8000/admin/dialer_campaign/contact/1/)

**Change Contact**

History

<b>Phonebook:</b>	Default_Phonebook 
<small>Select Phonebook</small>	
<b>Contact:</b>	650153589
<b>Name:</b>	Fourth John
<b>Description:</b>	<div></div>
<small>Additional information about the contact</small>	
<b>Status:</b>	INACTIVE 
<b>Additional vars:</b>	amount=10
<div>  Delete         <div>    </div> </div>	

To import bulk contacts into a phonebook, click on **Import contacts**. where you can upload the contacts via a CSV file in to one phonebook.

**URL:**


- [http://localhost:8000/admin/dialer\\_campaign/contact/import\\_contact/](http://localhost:8000/admin/dialer_campaign/contact/import_contact/)

## Import Contact

Sample File

contact	last name	first name	email-id	description	status	additional_vars
1234	Belaid	Arezqui	areski@gmail.com	test subscriber	1	test
5678	Fourth	John	john@gmail.com	test subscriber	0	test

CONTACT STATUS = ('1'-'ACTIVE') ('0'-'INACTIVE')

<b>Phonebook:</b>	Default_Phonebook 
<small>Select Phonebook</small>	
<b>Upload CSV File :</b>	<div></div> <div>Browse...</div>
<small>Browse CSV file</small>	
<div> <div>Submit</div> <div>Reset</div> </div>	

## Campaign

The campaign list will be displayed from the following URL. You can add a new campaign by clicking **Add campaign**. While adding a campaign, it is important to add campaign's start and end dates with time & week-day exceptions. Also select the gateway through which calls will be routed & the phonebook(s) linked with contacts.

From the campaign list, click on the campaign that you want to update.

**URL:**

- [http://localhost:8000/admin/dialer\\_campaign/campaign/](http://localhost:8000/admin/dialer_campaign/campaign/)

### Select Campaign to change

Action: <div><div></div></div> <div>Go</div>		0 of 1 selected													
<input type="checkbox"/>	ID	Name	User	Starting	Expiring	Frequency	Call Max Duration	Max Retries	A-Leg Gateway	VoIP Application	VoIP App Data	Status	Action	Contact	Campaign Subscriber
<input type="checkbox"/>	1	Default_Campaign	areski	April 8, 2011, 6:31 p.m.	April 8, 2011, 6:31 p.m.	10	3	3	Default_Gateway	Default_VoIP_App		START	Pause   Stop	3	<a href="#">Details</a>
1 Campaign															

To Add/Update Campaign for user

**URL:**

- [http://localhost:8000/admin/dialer\\_campaign/campaign/add/](http://localhost:8000/admin/dialer_campaign/campaign/add/)
- [http://localhost:8000/admin/dialer\\_campaign/campaign/1/](http://localhost:8000/admin/dialer_campaign/campaign/1/)

### Change Campaign

History

Standard options

**Name:**

**Description:**

Short description of the Campaign

**User:**

**Status:**

**Starting:** Date:  Today |  Now |

**Expiring:** Date:  Today |  Now |

**A-Leg Gateway:**  

Select Gateway to use to reach the contact

**VoIP Application:**  

Select VoIP Application to use with this campaign

**VoIP App Data:** 

Define the additional data to pass to the application

**Phonebook:** 

Available phonebook

Chosen phonebook

Select your choice(s) and click +

## Campaign Subscriber

The Campaign Subscriber list will be displayed from the following URL. You can add a new campaign subscriber by clicking Add Campaign Subscriber. Also from the campaign subscriber list, click on the subscriber to update.

While creating a contact, if its linked phonebook is also attached to a running campaign, then the contact will be added into the campaign subscriber.

**URL:**

- [http://localhost:8000/admin/dialer\\_campaign/campaignsubscriber/](http://localhost:8000/admin/dialer_campaign/campaignsubscriber/)

## Select Campaign Subscriber to change

Action:  Go 0 of 1 selected

ID	Contact	Campaign	Callrequest	Last attempt	Count attempt	Contact	Contact name	Status	Date
1	650784355 (Belaid Avezqui)	Default_Campaign	(None)	(None)	0	650784355	Belaid Avezqui	PENDING	April 29, 2011, 8:44 a.m.

1 Campaign Subscriber

Filter

By campaign:

By Date:

By last attempt:

To Add/Update Campaign Subscriber

URL:

- [http://localhost:8000/admin/dialer\\_campaign/campaignsubscriber/add/](http://localhost:8000/admin/dialer_campaign/campaignsubscriber/add/)
- [http://localhost:8000/admin/dialer\\_campaign/campaignsubscriber/1/](http://localhost:8000/admin/dialer_campaign/campaignsubscriber/1/)

## Change Campaign Subscriber

History

Contact:  650784355 (Belaid Avezqui)

Campaign:  Default\_Campaign

Callrequest:

Last attempt: Date:  Today  Now

Count attempt:  0

Contact:  650784355

Status:  PENDING

## Dialer Settings

The dialer settings list will be displayed from the following URL. The Dialer settings list is applied to a system User. You can add a new setting by clicking Add Dialer Settings and add numeric values for the limit. Also from the dialer settings list, click on the setting to update.

URL:

- [http://localhost:8000/admin/dialer\\_settings/dialersetting/](http://localhost:8000/admin/dialer_settings/dialersetting/)

## Select Dialer Setting to change

Add Dialer Setting +

Search

Action:  Go 0 of 1 selected

Name	Max frequency	Call Max Duration	Max Retries	Timeout on Call	Max number campaign	Max number subscriber campaign	Blacklist	Whitelist	Updated date
Default_Dialer_Setting	100	3	3	45	2	3	*	12345*	April 29, 2011, 8:43 a.m.

1 Dialer Setting

To Add/Update dialer settings for a Newfies-Dialer user

URL:

- [http://localhost:8000/admin/dialer\\_settings/dialersetting/add/](http://localhost:8000/admin/dialer_settings/dialersetting/add/)
- [http://localhost:8000/admin/dialer\\_settings/dialersetting/1/](http://localhost:8000/admin/dialer_settings/dialersetting/1/)

**Change Dialer Setting**History

<b>Name:</b>	<input type="text" value="Default_Dialer_Setting"/>
Define a name for this set of settings	
<b>Max frequency:</b>	<input type="text" value="100"/>
Define the Max frequency, speed of the campaign. This is the number of calls per minute.	
<b>Call Max Duration:</b>	<input type="text" value="3"/>
Define the max retry allowed.	
<b>Max Retries:</b>	<input type="text" value="3"/>
Define the max retry allowed per user.	
<b>Timeout on Call:</b>	<input type="text" value="45"/>
Define the maximum amount of second to timeout on calls	
<b>Max number campaign:</b>	<input type="text" value="2"/>
Max Number of campaign	
<b>Max number subscriber campaign:</b>	<input type="text" value="3"/>
Max Number of subscriber per campaign	
<b>Blacklist:</b>	<div><div>*</div><div></div></div>
Define Regular Expression of phonenumber you want to forbid. This is used to blacklist phonenumber to be called. Example "[2-4][1]*" will forbid all the phonenumber starting by 2,3 or 4 and followed by 1	
<b>Whitelist:</b>	<div><div>12345*</div><div></div></div>

To apply dialer settings limit to a User, click on Customers or Admins, select the user to be updated & apply settings from the dialer settings list.

### URL:

- <http://localhost:8000/admin/auth/staff/1/>

**User Profile**

**User Profile: UserProfile object**Delete

<b>Accountcode:</b>	<input type="text"/>
<b>Dialer Setting:</b>	<input type="text" value="[1] Default_Dialer_Setting"/> +
<b>Gateway:</b>	<input type="text" value="Default_Gateway"/> +
Hold down "Control", or "Command" on a Mac, to select more than one.	
<b>Server Group:</b>	<input type="text" value="Default_Group"/> +
Hold down "Control", or "Command" on a Mac, to select more than one.	

✖ DeleteSave and add anotherSave and continue editingSave

## Dialer Gateway

The Dialer Gateway list will be displayed from the following URL. You can add a new gateway by clicking Add Dialer Gateway and adding the details (e.g. gateway name, hostname, protocol etc.). Also from the gateway list, click on the gateway that you want to update.

### URL:

- [http://localhost:8000/admin/dialer\\_gateway/gateway/](http://localhost:8000/admin/dialer_gateway/gateway/)

## Select Dialer Gateway to change

Action:  Go 0 of 1 selected

ID	Name	Protocol	Hostname	Addprefix	Removeprefix	Secondused	Count call
<input type="checkbox"/> 1	Default_Gateway	SIP	localhost			(None)	(None)

1 Dialer Gateway

Add Dialer Gateway +

Filter

By protocol: All

By hostname: All

To Add/Update a dialer gateway

## URL:

- [http://localhost:8000/admin/dialer\\_gateway/gateway/add/](http://localhost:8000/admin/dialer_gateway/gateway/add/)
- [http://localhost:8000/admin/dialer\\_gateway/gateway/1/](http://localhost:8000/admin/dialer_gateway/gateway/1/)

## Change Dialer Gateway

History

Standard options

Name:   
Enter Gateway Name

Description:   
Short description about Provider

Protocol:

Hostname:

Gateway Status:

Advanced options [Hide](#)

Addprefix:

Removeprefix:

Failover:  +  
Select Gateway

Addparameter:

Maximum call:

[Delete](#) [Save and add another](#) [Save and continue editing](#) [Save](#)

## VoIP Application

The VoIP application list will be displayed from the following URL. You can add a new application by clicking Add VoIP Application. Also from the application list, click on the application to update.

## URL:

- [http://localhost:8000/admin/voip\\_app/voipapp/](http://localhost:8000/admin/voip_app/voipapp/)

## Select VoIP Application to change

Action:  Go 0 of 1 selected

ID	Name	Description	Type	Date
<input type="checkbox"/> 1	Default_VoIP_App		REDIRECT	April 8, 2011, 8 a.m.

1 VoIP Application

Add VoIP Application +

Filter

By Date: Any date

To Add/Update a VoIP application

## URL:

- [http://localhost:8000/admin/voip\\_app/voipapp/add/](http://localhost:8000/admin/voip_app/voipapp/add/)

- [http://localhost:8000/admin/voip\\_app/voipapp/1/](http://localhost:8000/admin/voip_app/voipapp/1/)

**Change VoIP Application**

History

**Name:**

Description:   
Description about the Voip Application

Type:

Gateway:   
Gateway used if we redirect the call

[✖ Delete](#) [Save and add another](#) [Save and continue editing](#) [Save](#)

## Call Request

The call request list will be displayed from the following URL. You can add a new call request by clicking Add Call Request. Also from the call request list, click on the request to update.

**URL:**

- [http://localhost:8000/admin/dialer\\_cdr/callrequest/](http://localhost:8000/admin/dialer_cdr/callrequest/)

**Select Call Request to change**

Add Call Request +

<

To Add/Update a Call Request

**URL:**

- [http://localhost:8000/admin/dialer\\_cdr/callrequest/add/](http://localhost:8000/admin/dialer_cdr/callrequest/add/)
- [http://localhost:8000/admin/dialer\\_cdr/callrequest/1/](http://localhost:8000/admin/dialer_cdr/callrequest/1/)



## Change Call Request

History

Standard options	
Uniqueid:	2342jdst-00123
Campaign:	<input type="text"/> Select Campaign
Callback time:	Date: 2011-05-01 Today Time: 11:22:33 Now
Status:	SUCCESS
Exten:	1231321
Context:	mycontext
Application:	<input type="text"/>
Timeout:	30000
Callerid:	650784355
Advanced options	
Call Request Type:	ORIGINAL
A-Leg Gateway:	 Select Gateway to use to reach the subscriber
VoIP Application:	 Select VoIP Application to use with this campaign
VoIP App Data:	<input type="text"/> Define the additional data to pass to the application
Campaign Subscriber:	<input type="text"/> Campaign Subscriber related to this callrequest

## VoIP Call Report

A VoIP Call list will be displayed from following URL. You **can not** add new call reports.

## URL:

- [http://localhost:8000/admin/dialer\\_cdr/voipcall/](http://localhost:8000/admin/dialer_cdr/voipcall/)

## Call Report

Search Option

From:   
Date Format: YYYY-MM-DD.

To:   
Date Format: YYYY-MM-DD.

Disposition:

Action: Go 0 of 2 selected

<input type="checkbox"/>	User	Used gateway	Callid	CallerID	Phone number	Starting date	Sessiontime	Disposition	Destination
<input type="checkbox"/>	areski	Default_Gateway	SIP-1234567890	Areski	9427164510	June 24, 2011, 12:11 a.m.	90	ANSWER	91
<input type="checkbox"/>	areski	Default_Gateway	SIP-1234567890	Areski	9427164510	June 24, 2011, 12:10 a.m.	120	ANSWER	91

2 VoIP Call

Call Detail Report Summary				
Date	Duration	Graphic	Calls	ACT
2011-06-24	03:30	<div style="width: 100%;"></div>	2	01:45
Total	03:30		2	01:45

Export CSV file



## 3.4 Customer Panel

User Interface :

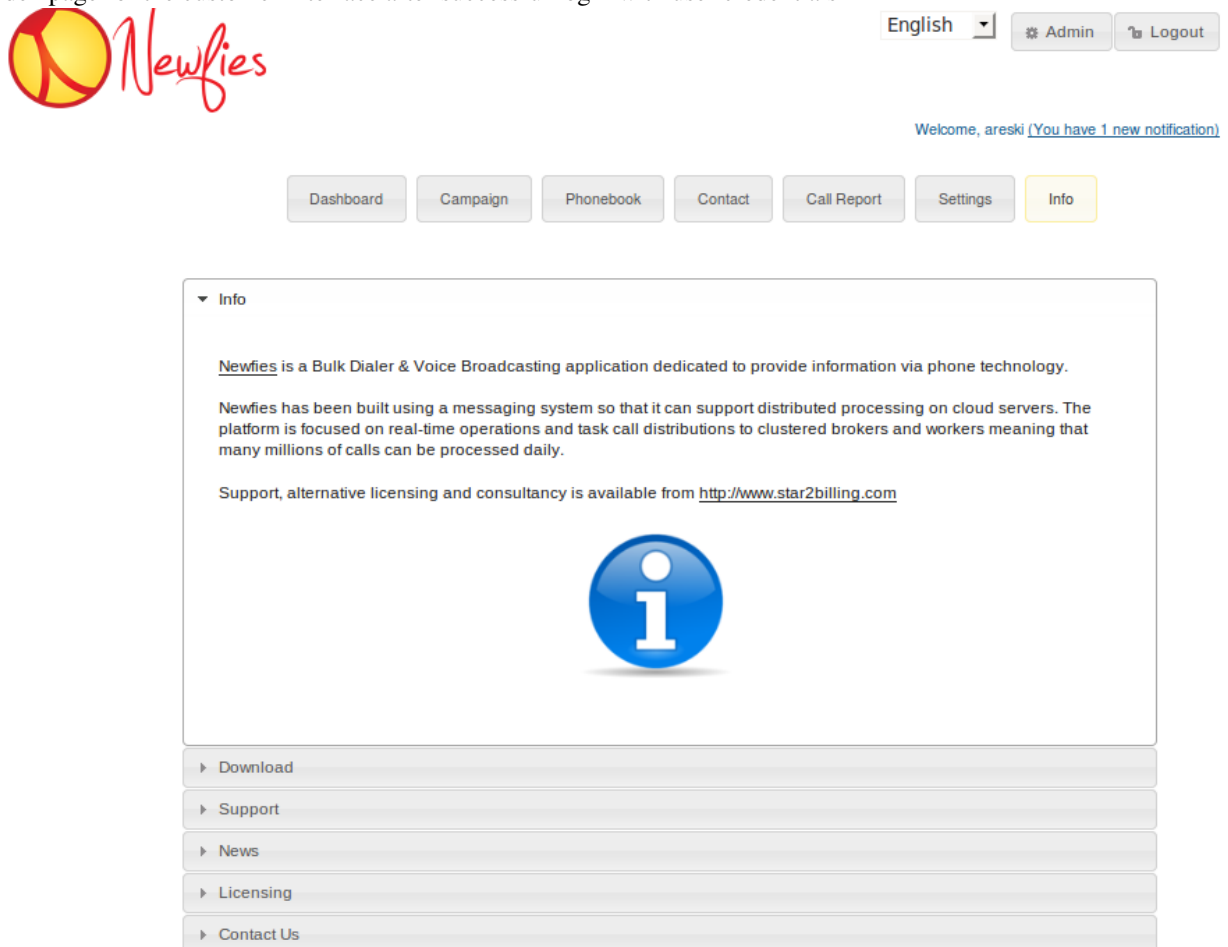
<http://localhost:8000/> This application provides a user interface for restricted management of the User's Campaigns, Phonebooks and Subscribers. It also provides detailed reporting of calls and message delivery.

- Screenshot with Features

### 3.4.1 Screenshot with Features

#### Index

Index page for the customer interface after successful login with user credentials



## Phonebook

The phonebook list will be displayed from the following URL. You can add a new phonebook by clicking Add phonebook and add the name of a phonebook and its description. Also from the phonebook list, click on the phonebook to update.

### URL:

- [http://localhost:8000/dialer\\_campaign/phonebook/](http://localhost:8000/dialer_campaign/phonebook/)

Dashboard
Campaign
Phonebook
Contact
Call Report
Settings
Info

▲

Phonebooks

Add
 Delete Selected

<input type="checkbox"/>	Id	Name	Description	Date	Contacts	Action
<input type="checkbox"/>	1	Default_Phonebook	This is default phone book	2011-05-24 06:57:48	10	
<input type="checkbox"/>	3	myphonebook		2011-06-13 06:10:46	10	

10 ▼

Page 1 of 1

Displaying 1 to 2 of 2 items

To Add/Update a Phonebook for a logged in user

### URL:

- [http://localhost:8000/dialer\\_campaign/phonebook/add/](http://localhost:8000/dialer_campaign/phonebook/add/)
- [http://localhost:8000/dialer\\_campaign/phonebook/1/](http://localhost:8000/dialer_campaign/phonebook/1/)

Update Phonebook

Phonebook

Name:

Default\_Phonebook

Description:

This is default phone book

Phonebook Notes

Update

Clear

Delete

### Contact

The contact list will be displayed from following the URL. You can add a new contact by clicking `Add contact` & adding the contact details (i.e. phone number, name, description about contact, contact status) under the logged in user's phonebook from the phonebook list. On the contact list, click on the contact to update.

#### URL:

- [http://localhost:8000/dialer\\_campaign/contact/](http://localhost:8000/dialer_campaign/contact/)

Dashboard
Campaign
Phonebook
Contact
Call Report
Settings
Info

Contact Number: 
☒ Equals
☐ Begins with
☐ Contains
☐ Ends with

Contact Name:

Phonebook:

Status:
☐ Inactive
☐ Active
☒ All

Submit
Retset

Add
Import
Delete Selected

<input type="checkbox"/>	Id	Phonebook	Contact	Last Name	First Name	Status	Date	Action
<input type="checkbox"/>	90	Default_Phonebook	640234000	Belaid	Arezqui	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	91	Default_Phonebook	640234001	Fourth	John	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	92	Default_Phonebook	640234002	J. Lowrey	Bill	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	93	Default_Phonebook	640234003	G. Cartwright	Ebony	ACTIVE	2011-06-10 09:27:27	
<input type="checkbox"/>	94	Default_Phonebook	640234004	K. Conley	Stephen	ACTIVE	2011-06-10 09:27:28	
<input type="checkbox"/>	95	Default_Phonebook	640234005	Burke	Linda	ACTIVE	2011-06-10 09:27:28	
<input type="checkbox"/>	96	Default_Phonebook	640234006	Mildred	Peacock	ACTIVE	2011-06-10 09:27:28	
<input type="checkbox"/>	97	Default_Phonebook	640234007	Parker	Donald	ACTIVE	2011-06-10 09:27:28	

10
Page 1 of 2
Displaying 1 to 10 of 20 items

To Add/Update a contact in a phonebook

URL:

- [http://localhost:8000/dialer\\_campaign/contact/add/](http://localhost:8000/dialer_campaign/contact/add/)
- [http://localhost:8000/dialer\\_campaign/contact/1/](http://localhost:8000/dialer_campaign/contact/1/)

Update Contact

**Contact**

**Phonebook:**
Default\_Phonebook
Select Phonebook

**Last name:**
Belaid

**First name:**
Arezqui

**Email:**
areski@gmail.com

**Country:**
-----

**City:**

**Description:**
test subscriber
Contact Notes

**Status:**
ACTIVE

**Additional parameters:**

Update
Clear
Delete

To import bulk contacts into a phonebook, click on **Import**. where you can upload contacts via a CSV file under a logged in user's phonebook.

#### URL:

- [http://localhost:8000/dialer\\_campaign/contact/import/](http://localhost:8000/dialer_campaign/contact/import/)

Sample File						
contact	last name	first name	email-id	description	status	additional_vars
1234	Belaid	Arezqui	areski@gmail.com	test subscriber	1	test
5678	Fourth	John	john@gmail.com	test subscriber	0	test

SUBSCRIBER STATUS = ('1'-ACTIVE) ('0'-INACTIVE)

**Import Contact**

**Phonebook:**
Default\_Phonebook
Select Phonebook

**Upload CSV File :**
Browse...
Browse CSV file

Submit
Reset

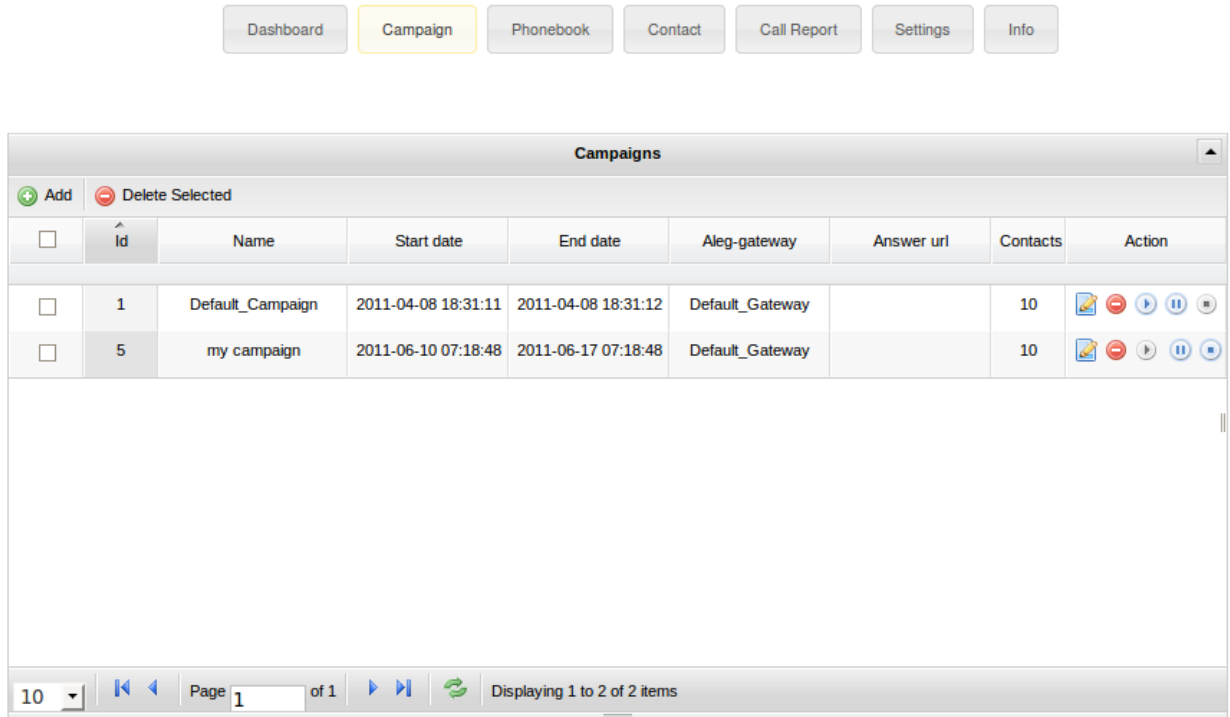
## Campaign

The campaign list will be displayed from the following URL. You can add a new campaign for the logged in user by clicking **Add campaign**. When adding a campaign, it is important to add the campaign's start and end dates with

time & week-day exceptions. Select the gateway through which calls will be routed & phonebook(s) that are linked with contacts from the campaign list, click on campaign to update.

**URL:**

- [http://localhost:8000/dialer\\_campaign/campaign/](http://localhost:8000/dialer_campaign/campaign/)



### To Add/Update a Campaign for a logged in user

**URL:**

- [http://localhost:8000/dialer\\_campaign/campaign/add/](http://localhost:8000/dialer_campaign/campaign/add/)
- [http://localhost:8000/dialer\\_campaign/campaign/1/](http://localhost:8000/dialer_campaign/campaign/1/)

Update Campaign

Campaign

Name: Default\_Campaign

Description:   
Short description of the Campaign

CallerID: 123987  
CallerID used to call the A-Leg

Status: PAUSE ▾

A-Leg Gateway: Default\_Gateway ▾  
Select gateway to use for this campaign

VoIP Application: Default\_VoIP\_App ▾  
Select VoIP Application to use with this campaign

Extra Parameters: 2000  
Additional application parameters.

Phonebook: Default\_Phonebook ▾  
Hold down "Control", or "Command" on a Mac, to select more than one.

Campaign settings

Frequency: 20  
Define the frequency, speed of the campaign. This is the number of calls per minute.

Call Max Duration: 50  
Define the call's duration maximum. (Value in seconds 1800 = 30 minutes)

Max Retries: 3  
Define the max retry allowed per user.

Time between Retries: 3000  
Define the time to wait between retries in seconds

Timeout on Call: 60  
Define the amount of second to timeout on calls

Campaign schedule

Start: 2011-06-28 06:46:08  
Date Format: YYYY-mm-DD HH:MM:SS

Daily start time: 00:00:00  
Time Format: HH:MM:SS

Finish: 2011-07-05 06:46:08  
Date Format: YYYY-mm-DD HH:MM:SS

Daily stop time: 23:59:59  
Time Format: HH:MM:SS

Week Days: ☒ Monday ☒ Tuesday ☒ Wednesday ☒ Thursday ☒ Friday ☒ Saturday ☒ Sunday

Update Clear Delete

## Dashboard

Dashboard gives the information anbout campaign & its related call records







# CONFIGURATION AND DEFAULTS

Contents:

## 4.1 Sample Configuration

This is a sample configuration to get you started. It should contain all you need to create a basic set-up.

### 4.1.1 The Configuration Module

Some of the more important parts of the configuration module for the Newfies, `settings.py`, are explained below:

```
import os.path
APPLICATION_DIR = os.path.dirname(globals()['__file__'])
```

`APPLICATION_DIR` now contains the full path of your project folder and can be used elsewhere in the `settings.py` module so that your project may be moved around the system without you having to worry about changing any troublesome hard-coded paths.

```
DEBUG = True
```

turns on debug mode allowing the browser user to see project settings and temporary variables.

```
ADMINS = (('xyz', 'xyz@abc.com'))
```

sends all errors from the production server to the admin's email address.

```
DATABASE_ENGINE = 'mysql'
DATABASE_NAME = 'db-name'
DATABASE_USER = 'user'
DATABASE_PASSWORD = 'password'
DATABASE_HOST = 'mysql-host'
DATABASE_PORT = ''
```

sets up the options required for Django to connect to your database.

```
MEDIA_ROOT = os.path.join(APPLICATION_DIR, 'static')
```

tells Django where to find your media files such as images that the HTML templates might use.

```
ROOT_URLCONF = 'urls'
```

tells Django to start finding URL matches at in the `urls.py` module in the `newfies` project folder.

```
TEMPLATE_DIRS = (os.path.join(APPLICATION_DIR, 'templates'),)
```

tells Django where to find your HTML template files.

```
INSTALLED_APPS = (
 'django.contrib.auth',
 'django.contrib.contenttypes',
 'django.contrib.sessions',
 'django.contrib.sites',
 'django.contrib.admin',
 ...
 'dialer_gateway',
 'dialer_campaign',
 'dialer_cdr',
 'dialer_settings',
 'user_profile',
 'voip_server',
 'voip_app',
 ...
)
```

tells Django which applications (custom and external) to use in your project. The custom applications, `dialer_gateway`, `dialer_campaign` etc. are stored in the project folder along with these custom applications.

### 4.1.2 The URLs modules

The defined URL patterns for the CPI Pilot project are divided into URL patterns specific to the project and URL patterns specific to the applications. For more information on how the pattern matching syntax work or how to write your own url patterns please consult Django's [URL Dispatcher](#) documentation.

#### Project specific URL patterns

The URL patterns specific to the project are applied in the `urls.py` file that is stored in the project directory `newfies`. The code segments that add these URL patterns aren't lengthy and are shown below:

```
urlpatterns = patterns('',
 # redirect
 (r'^$', 'django.views.generic.simple.redirect_to', {'url': '/dialer_campaign/'}),
 (r'^admin/', include(admin.site.urls)),
 (r'^api/dialer_campaign/', include('dialer_campaign.api.urls')),
 (r'^dialer_campaign/', include('dialer_campaign.urls')),
 (r'^static/(?P<path>.*)$', 'django.views.static.serve',
 {'document_root': settings.STATIC_ROOT}),
)
```

#### Application specific URL patterns

The URL patterns specific to the `dialer_campaign` application are applied in the `/dialer_campaign/urls.py` file in the `dialer_campaign` application folder. The code segment that adds these URL patterns isn't lengthy either and is shown below:

```
urlpatterns = patterns('dialer_campaign.views',
 (r'^phonebook/$', 'phonebook_list'),
 (r'^phonebook/add/$', 'phonebook_add'),
```

```
(r'^phonebook/(.+)/$', 'phonebook_change'),
)
```

### 4.1.3 The Views module

The functions defined in `views.py` represent the logic behind the webpages. The view functions (called through the URL matching) decide which data structures need to be constructed and sent through to the HTML templates. To do this, each view function uses Django's object relational model (ORM) to query the database picking out what is needed for any particular page.

```
@login_required
def phonebook_add(request):
 """
 Add new Phonebook
 """
 form = PhonebookForm()
 if request.method == 'POST':
 form = PhonebookForm(request.POST)
 if form.is_valid():
 obj = form.save(commit=False)
 obj.user = User.objects.get(username=request.user)
 obj.save()
 request.session["msg"] = _('%s' is added successfully.' %\
 request.POST['name'])
 return HttpResponseRedirect('/dialer_campaign/phonebook/')
 template = 'dialer_campaign/phonebook/change.html'
 data = {
 'form': form,
 'action': 'add',
 }
 return render_to_response(template, data,
 context_instance=RequestContext(request))
```

### 4.1.4 The Admin Module

The classes defined in `admin.py` tell Django what attributes are visible and modifiable from the admin site.

Code for naming convention (e.g. Voip -> VoIP) (in `admin.py`)

**Example:**

```
def get_urls(self):
 urls = super(VoipAppAdmin, self).get_urls()
 my_urls = patterns('',
 (r'^add/$', self.admin_site.admin_view(self.add_view)),
)
 return my_urls + urls

def add_view(self, request, extra_context=None):
 ctx = {
 'app_label': _('VoIP'),
 'title': _('Add VoIP'),
 }
 return super(VoipAppAdmin, self)\
 .add_view(request, extra_context=ctx)
```

## 4.2 Celery Configuration

### 4.2.1 After installing Broker (Redis or Rabbitmq)

#### 1. Redis Settings

This is a configuration example for Redis.

```
Redis Settings
CARROT_BACKEND = "ghettoq.taproot.Redis"

BROKER_HOST = "localhost" # Maps to redis host.
BROKER_PORT = 6379 # Maps to redis port.
BROKER_VHOST = "0" # Maps to database number.

CELERY_RESULT_BACKEND = "redis"
REDIS_HOST = "localhost"
REDIS_PORT = 6379
REDIS_DB = 0
#REDIS_CONNECT_RETRY = True
```

#### 2. Rabbitmq Settings

This is a configuration example for Rabbitmq.

```
BROKER_HOST = "localhost"
BROKER_PORT = 5672
BROKER_USER = "root"
BROKER_PASSWORD = "root"
BROKER_VHOST = "localhost"

CELERY_RESULT_BACKEND = "amqp"
```

### 4.2.2 Launch celery/celerybeat in debug mode

If you don't want to run celeryd and celerybeat as a daemon then

To run celeryd

```
$ python manage.py celeryd -E -l debug
```

To run celerybeat

```
$ python manage.py celerybeat --schedule=/var/run/celerybeat-schedule
```

To run both

```
$ python manage.py celeryd -E -B -l debug
```

### 4.2.3 Running celeryd/celerybeat as a daemon (Debian/Ubuntu)

To configure celeryd you will need to tell it where to change directory to, when it starts in order to find your celeryconfig.

```
$ cd install/celery-init/etc/default/
```

1. Open celeryd in text editor & change the following variables

Configuration file: /etc/default/celeryd

Init script: celeryd.

Usage : /etc/init.d/celeryd {start|stop|force-reload|restart|try-restart|status}:

```
Where to chdir at start
CELERYD_CHDIR="/path/to/newfies/"

Path to celeryd
CELERYD="/path/to/newfies/manage.py celeryd"

Extra arguments to celeryd
CELERYD_OPTS="--time-limit=300"

Name of the celery config module.
CELERY_CONFIG_MODULE="celeryconfig"

Extra Available options
%n will be replaced with the nodename.
Full path to the PID file. Default is /var/run/celeryd.pid.
CELERYD_PID_FILE="/var/run/celery/%n.pid"

Full path to the celeryd log file. Default is /var/log/celeryd.log
CELERYD_LOG_FILE="/var/log/celery/%n.log"

User/Group to run celeryd as. Default is current user.
Workers should run as an unprivileged user.
CELERYD_USER="celery"
CELERYD_GROUP="celery"
```

2. Open celeryd (for periodic task) in text editor & add the following variables

Configuration file: /etc/default/celerybeat or /etc/default/celeryd

Init script: celerybeat

Usage: /etc/init.d/celerybeat {start|stop|force-reload|restart|try-restart|status}:

```
Path to celerybeat
CELERYBEAT="/path/to/newfies/manage.py celerybeat"

Extra arguments to celerybeat
CELERYBEAT_OPTS="--schedule=/var/run/celerybeat-schedule"
```

3. Copy the configuration file & init scripts to /etc dir:

```
$ cp etc/default/celeryd /etc/default/

$ cp etc/init.d/celeryd /etc/init.d/

$ cp etc/init.d/celerybeat /etc/init.d/
```

4. Run/Start or Stop celery as a daemon:

```
$ /etc/init.d/celeryd start or stop

$ /etc/init.d/celerybeat start or stop
```

### 4.2.4 Troubleshooting

If you can't get the celeryd as a daemon to work, you should try running them in verbose mode:

```
$ sh -x /etc/init.d/celeryd start
```

```
$ sh -x /etc/init.d/celerybeat start
```



# FREESWITCH INSTALLATION AND CONFIGURATION

Contents:

## 5.1 Freeswitch Installation and configuration

Freeswitch is the telephony engine used by Newfies-Dialer to make calls, as well as broadcast voice applications

Newfies-Dialer communicates with Freeswitch through the Event-Socket. Communication is made via the Communications Framework [Plivo](#). Our Freeswitch dependencies are the same as Plivo, therefore the following modules will need to be installed

```
mod_curl, asr_tts, mod_flite, asr_tts, mod_shout, mod_dingaling, mod_shell_stream, mod_xml_cdr
```

In order to retrieve CDR Status from the outbound calls, you will have to configure `xml_cdr.conf.xml` and point it to the Newfies API to store CDR, which is by default : [http://hostname\\_newfies/api/dialer\\_cdr/store\\_cdr/](http://hostname_newfies/api/dialer_cdr/store_cdr/)

A script for Freeswitch Installation which will install Freeswitch with the required modules and configure it for you is available.

Download and run the Freeswitch installation script.

Once logged in as root, execute the following command:

```
wget https://raw.githubusercontent.com/Star2Billing/newfies-dialer/master/install/install-freeswitch.sh
```

The above command download the installation script. We can then execute the script with the following command:

```
bash install-freeswitch.sh
```

This will download and install Freeswitch with the modules appropriate for Newfies-Dialer. The installation will take some time, but does not require your interaction once started.

## 5.2 Plivo Installation and configuration

When [Freeswitch](#) is installed, the next task is to install [Plivo](#). Plivo is an open source communications framework to rapidly deploy voice based applications used in conjunction with Newfies-Dialer.

Run the following commands:

```
wget https://raw.githubusercontent.com/plivo/plivo/master/scripts/plivo_install_beta.sh
```

then:

```
bash plivo_install_beta.sh /usr/share/plivo
```

This will download and install Plivo and all its dependencies. We need to have Plivo start on boot, so run the following command to make it automatically start.

```
ln -s /usr/share/plivo/bin/plivo /etc/rc2.d/S99plivo
```

Please note that the Plivo script makes alterations to the Freeswitch dial-plan, so it should not be blindly run on an existing working Freeswitch installation, as it will change your current configuration. If you wish to install Plivo on an existing version of Freeswitch, use the script as a guide, or edit it to suit your requirements.

## 5.3 Freeswitch Trunk configuration

In order for Newfies-Dialer to make outbound calls to its subscribers, you will need a SIP trunk. The Freeswitch wiki can provide more information on configuring trunks. However creating a trunk simply for Newfies-Dialer is straightforward.

Trunks or gateways, as they are known in Freeswitch, are configured using XML syntax, so using your favourite text editor, while logged in as root “`sudo su -`” create an XML file in `/usr/local/freeswitch/conf/sip_profiles/external/` and give it an identifiable name, e.g. `call-labs.xml`, and place the following lines in the file:

```
<include>
<gateway name="ip address or hostname of carrier">
<!--/// account username *required* ///-->
<param name="username" value="your username provided by carrier"/>
<!--/// auth realm: *optional* same as gateway name, if blank ///-->
<!--<param name="realm" value="asterlink.com"/>-->
<!--/// username to use in from: *optional* same as username, if blank ///-->
<param name="from-user" value="your username provided by carrier"/>
<!--/// domain to use in from: *optional* same as realm, if blank ///-->
<!--<param name="from-domain" value="" />-->
<!--/// account password *required* ///-->
<param name="password" value="your password supplied by carrier"/>
<!--/// extension for inbound calls: *optional* same as username, if blank ///-->
<!--<param name="extension" value="cluecon"/>-->
<!--/// proxy host: *optional* same as realm, if blank ///-->
<!--<param name="proxy" value="asterlink.com"/>-->
<!--/// send register to this proxy: *optional* same as proxy, if blank ///-->
<!--<param name="register-proxy" value="mysbc.com"/>-->
<!--/// expire in seconds: *optional* 3600, if blank ///-->
<!--<param name="expire-seconds" value="60"/>-->
<!--/// do not register ///-->
<param name="register" value="true"/>
<!-- which transport to use for register -->
<!--<param name="register-transport" value="udp"/>-->
<!--How many seconds before a retry when a failure or timeout occurs -->
<!--<param name="retry-seconds" value="30"/>-->
<!--Use the callerid of an inbound call in the from field on outbound calls via this gateway -->
<!--<param name="caller-id-in-from" value="false"/>-->
<!--extra sip params to send in the contact-->
<!--<param name="contact-params" value="tport=tcp"/>-->
<!--send an options ping every x seconds, failure will unregister and/or mark it down-->
<!--<param name="ping" value="25"/>-->
```

```
</gateway>
</include>
```

The uncommented lines are almost certainly required by your carrier and Freeswitch, the remaining parameters can be uncommented and used, if required by your carrier. The XML syntax for comments are denoted by lines that begin “<!-- “ and end in “--> “.

Finally we need to load the new configuration, and check the trunk is registered. Enter the Freeswitch CLI (Command Line Interface) from the console:

```
/usr/local/freeswitch/bin/fs_cli
```

You should now see the Freeswitch CLI, so now reload the Freeswitch configuration with the following command: (tip; Tab auto-completes):

```
sofia profile external restart reloadxml
```

When complete, check the trunk has registered with the command:

```
sofia status
```

Against the name of the trunk you configured in the XML file, you should see REGED (registered) at the end of the line. Take a note of the trunk name, we are going to need it for telling Newfies-Dialer that it can use this trunk.

To exit the Freeswitch CLI, do CTRL D, or /exit

Freeswitch configuration is now complete.



# DEVELOPER DOC

Contents:

## 6.1 Prerequisites

To fully understand this project, developers will need to have a advanced knowledge of:

- Django : <http://www.djangoproject.com/>
- Celery : <http://celeryproject.org/>
- Python : <http://www.python.org/>
- Freeswitch : <http://freeswitch.org/>
- Freeswitch : Event\_Sockets ([wiki.freeswitch.org/wiki/Mod\\_event\\_socket](http://wiki.freeswitch.org/wiki/Mod_event_socket))

## 6.2 Coding Style & Structure

### 6.2.1 Style

Coding follows the [PEP 8 Style Guide for Python Code](#).

### 6.2.2 Structure

The newfies directory:

```
|-- custom_admin_tools - The code for admin dashboard/menu
|-- dialer_campaign - The code for dialer campaign
| |-- api
| |-- fixtures
|-- dialer_cdr - This defines the call request & its information
| |-- api
| |-- fixtures
|-- dialer_gateway - This defines the trunk to deliver the VoIP Calls
| |-- fixtures
|-- dialer_settings - This defines sets of settings to apply on user
|-- voip_app - This defines application that are defined on the platform
| |-- fixtures
|-- static
```

```
| |-- newfies
| | |-- css
| | |-- js
| | |-- images
|-- user_profile - The code for user profile to extend auth model of Django
|-- resources - This area is used to hold media files
'-- templates - This area is used to override templates
 |-- admin
 | |-- dialer_campaign
 | |-- dialer_cdr
 |-- admin_tools
 |-- registration
 |-- memcache_status
 '-- frontend
```

## 6.3 Objects Description

### 6.3.1 Phonebook

**class** dialer\_campaign.models.**Phonebook** (\*args, \*\*kwargs)

This defines the Phonebook

**Attributes:**

- name - phonebook name.
- description - description about the phonebook.

**Relationships:**

- user - Foreign key relationship to the User model. Each phonebook is assigned to a User

**Name of DB table:** dialer\_phonebook

**phonebook\_contacts** ()

This will return a count of the contacts in the phonebook

### 6.3.2 Contact

**class** dialer\_campaign.models.**Contact** (\*args, \*\*kwargs)

This defines the Contact

**Attributes:**

- contact - Contact no
- last\_name - Contact's last name
- first\_name - Contact's first name
- email - Contact's e-mail address
- city - city name
- description - description about a Contact
- status - contact status
- additional\_vars - Additional variables

**Relationships:**

- **phonebook** - Foreign key relationship to the Phonebook model. Each contact mapped with a phonebook
- **country** - Foreign key relationship to the Country model. Each contact mapped with a country

**Name of DB table:** dialer\_contact

**contact\_name()**

Return Contact Name

### 6.3.3 CampaignManager

**class** dialer\_campaign.models.**CampaignManager**

Campaign Manager

**get\_expired\_campaign()**

Return all the campaigns which are expired or going to expire based on the expiry date but status is not 'END'

**get\_running\_campaign()**

Return all the active campaigns which will be running based on the expiry date, the daily start/stop time and days of the week

### 6.3.4 Campaign

**class** dialer\_campaign.models.**Campaign**(\*args, \*\*kwargs)

This defines the Campaign

**Attributes:**

- **campaign\_code** - Autogenerate campaign code to identify the campaign
- **name** - Campaign name
- **description** - Description about the Campaign
- **status** - Campaign status
- **callerid** - Caller ID
- **startingdate** - Starting date of the Campaign
- **expirationdate** - Expiry date of the Campaign
- **daily\_start\_time** - Start time
- **daily\_stop\_time** - End time
- **week\_day\_setting** (monday, tuesday, wednesday, thursday, friday, saturday, sunday)
- **frequency** - Frequency, speed of the campaign. number of calls/min
- **callmaxduration** - Max retry allowed per user
- **maxretry** - Max retry allowed per user
- **intervalretry** - Time to wait between retries in seconds
- **calltimeout** - Number of seconds to timeout on calls
- **aleg\_gateway** - Gateway to use to reach the contact
- **extra\_data** - Additional data to pass to the application

**Relationships:**

- voipapp - Foreign key relationship to the VoipApp model. VoIP Application to use with this campaign
- phonebook - Many-To-Many relationship to the Phonebook model.
- user - Foreign key relationship to the a User model. Each campaign assigned to a User

**Name of DB table:** dialer\_campaign

**campaignsubscriber\_detail ()**

This will link to campaign subscribers who are associated with the campaign

**count\_contact\_of\_phonebook (status=None)**

Count the no. of Contacts in a phonebook

**get\_active\_callmaxduration ()**

Get the active call max duration

**get\_active\_contact ()**

Get all the active Contacts from the phonebook

**get\_active\_contact\_no\_subscriber ()**

List of active contacts that do not exist in Campaign Subscriber

**get\_active\_max\_frequency ()**

Get the active max frequency

**get\_pending\_subscriber (limit=1000)**

Get all the pending subscribers from the campaign

**is\_authorized\_contact (str\_contact)**

Check if a contact is authorized

**progress\_bar ()**

Progress bar generated based on no of contacts

**update\_campaign\_status ()**

Update the campaign's status

For example, If campaign is active, you can change status to 'Pause' or 'Stop'

### 6.3.5 CampaignSubscriber

**class** dialer\_campaign.models.**CampaignSubscriber** (\*args, \*\*kwargs)

This defines the Contact imported to a Campaign

**Attributes:**

- last\_attempt -
- count\_attempt -
- duplicate\_contact -
- status -

**Relationships:**

- contact - Foreign key relationship to the Contact model.
- campaign - Foreign key relationship to the Campaign model.

**Name of DB table:** dialer\_campaign\_subscriber



### 6.3.6 Callrequest

**class** dialer\_cdr.models.**Callrequest** (\*args, \*\*kwargs)

This defines the call request, the dialer will read any new request and attempt to deliver the call.

**Attributes:**

- request\_uuid - Unique id
- call\_time - Total call time
- call\_type - Call type
- status - Call request status
- callerid - Caller ID
- last\_attempt\_time -
- result -
- timeout -
- timelimit -
- extra\_dial\_string -
- phone\_number -
- parent\_callrequest -
- extra\_data -
- num\_attempt -
- hangup\_cause -

**Relationships:**

- user - Foreign key relationship to the User model. Each campaign assigned to a User
- voipapp - Foreign key relationship to the VoipApp model. VoIP Application to use with this campaign
- aleg\_gateway - Foreign key relationship to the Gateway model. Gateway to use to reach the subscriber
- campaign\_subscriber - Foreign key relationship to the CampaignSubscriber Model.
- campaign - Foreign key relationship to the Campaign model.

**Name of DB table:** dialer\_callrequest

### 6.3.7 VoIPCall

**class** dialer\_cdr.models.**VoIPCall** (\*args, \*\*kwargs)

This gives information of all the calls made with the carrier charges and revenue of each call.

**Attributes:**

- callid - callid of the phonecall
- callerid - CallerID used to call out
- phone\_number - Phone number contacted
- dialcode - Dialcode of the phonenumber
- starting\_date - Starting date of the call

- `duration` - Duration of the call
- `billsec` -
- `progresssec` -
- `answersec` -
- `waitsec` -
- `disposition` - Disposition of the call
- `hangup_cause` -
- `hangup_cause_q850` -

**Relationships:**

- `user` - Foreign key relationship to the User model.
- `used_gateway` - Foreign key relationship to the Gateway model.
- `callrequest` - Foreign key relationship to the Callrequest model.

**Name of DB table:** `dialer_cdr`

**`destination_name()`**  
Return Recipient dialcode

**`min_duration()`**  
Return duration in min & sec

### 6.3.8 Gateway

**class** `dialer_gateway.models.Gateway(*args, **kwargs)`

This defines the trunk to deliver the Voip Calls. Each of the Gateways are routes that support different protocols and sets of rules to alter the dialed number.

**Attributes:**

- `name` - Gateway name.
- `description` - Description about the Gateway.
- `addprefix` - Add prefix.
- `removeprefix` - Remove prefix.
- `gateways` - “user/user”, # Gateway string to try dialing separated by comma. First in the list will be tried first
- `gateway_codecs` - “PCMA,PCMU’,PCMA,PCMU”, # Codec string as needed by FS for each gateway separated by comma
- `gateway_timeouts` - “10,10”, # Seconds to timeout in string for each gateway separated by comma
- `gateway_retries` - “2,1”, # Retry String for Gateways separated by comma, on how many times each gateway should be retried
- `originate_dial_string` - `originate_dial_string`
- `secondused` -
- `failover` -
- `addparameter` -

- count\_call -
- count\_in\_use -
- maximum\_call -
- status - Gateway status

**Name of DB table:** dialer\_gateway

### 6.3.9 DialerSetting

**class** dialer\_settings.models.**DialerSetting** (\*args, \*\*kwargs)

This defines the settings to apply to a user

**Attributes:**

- name - Settings name.
- max\_frequency - Max frequency, speed of the campaign. This is the number of calls per minute.
- callmaxduration - Max retries allowed
- maxretry - Max retries allowed per user
- max\_calltimeout - Maximum number of seconds to timeout on calls
- max\_number\_campaign - Max Number of campaigns
- max\_number\_subscriber\_campaign - Max Number of subscribera
- blacklist - Used to blacklist phone numbers to be called
- whitelist - Used to whitelist phone numbers to be called

**Name of DB table:** dialer\_setting

### 6.3.10 UserProfile

**class** user\_profile.models.**UserProfile** (\*args, \*\*kwargs)

This defines extra features for the user

**Attributes:**

- accountcode - Account name.

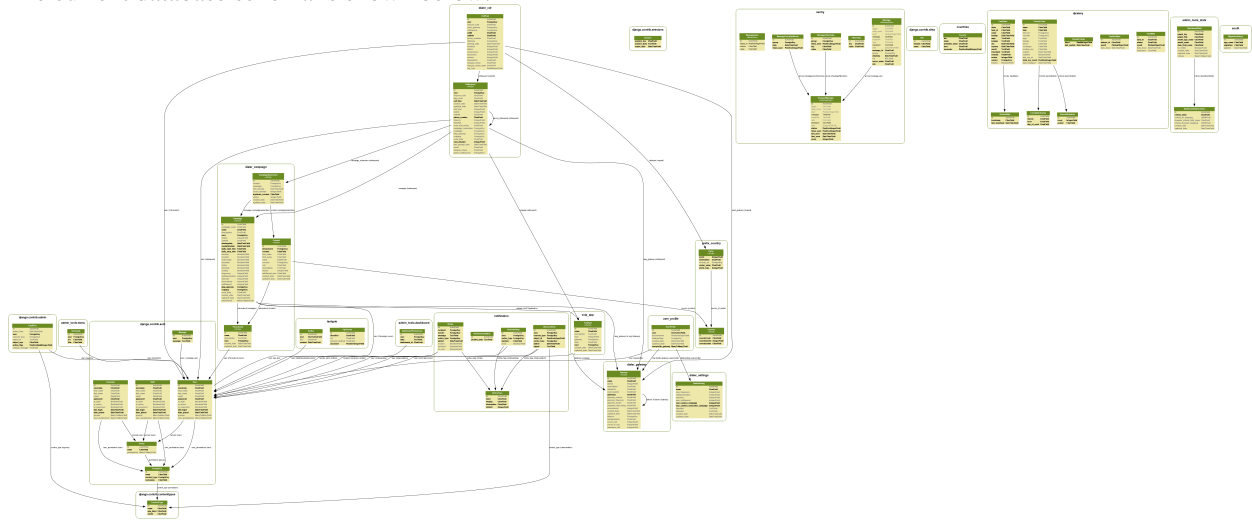
**Relationships:**

- user - Foreign key relationship to the User model.
- userprofile\_gateway - ManyToMany
- userprofile\_voipservergroup - ManyToMany
- dialersetting - Foreign key relationship to the DialerSetting model.

**Name of DB table:** user\_profile

## 6.4 Database Design

The current database schema is shown below:



## 6.5 Newfies-Dialer Views

### 6.5.1 index

`dialer_campaign.views.index(request)`

Index view of the Customer Interface

**Attributes:**

- `form` - LoginForm
- `template` - frontend/index.html

### 6.5.2 customer\_dashboard

`dialer_campaign.views.customer_dashboard(request, *args, **kwargs)`

Customer dashboard gives the following information

- No of Campaigns for logged in user
- Total phonebook contacts
- Total Campaigns contacts
- Amount of contact reached today
- Disposition of calls via pie chart
- Call records & Duration of calls are shown on graph by days/hours basis.

**Attributes:**

- `template` - frontend/dashboard.html
- `form` - DashboardForm

### 6.5.3 login\_view

`dialer_campaign.views.login_view(request)`

Check User credentials

**Attributes:**

- form - LoginForm
- template - frontend/index.html

**Logic Description:**

- Submitted user credentials need to be checked. If it is not valid then the system will redirect to the login page.
- If submitted user credentials are valid then system will redirect to the dashboard.

### 6.5.4 cust\_password\_reset

`dialer_campaign.views.cust_password_reset(request)`

Use `django.contrib.auth.views.password_reset` view method for forgotten password on the Customer UI

This method sends an e-mail to the user's email-id which is entered in `password_reset_form`

### 6.5.5 cust\_password\_reset\_done

`dialer_campaign.views.cust_password_reset_done(request)`

Use `django.contrib.auth.views.password_reset_done` view method for forgotten password on the Customer UI

This will show a message to the user who is seeking to reset their password.

### 6.5.6 cust\_password\_reset\_confirm

`dialer_campaign.views.cust_password_reset_confirm(request, uidb36=None, token=None)`

Use `django.contrib.auth.views.password_reset_confirm` view method for forgotten password on the Customer UI

This will allow a user to reset their password.

### 6.5.7 common\_send\_notification

`dialer_campaign.views.common_send_notification(request, status, recipient=None)`

User Notification (e.g. start | stop | pause | abort | contact/campaign limit) needs to be saved. It is a common function for the admin and customer UI's

**Attributes:**

- pk - primary key of the campaign record
- status - get label for notifications

**Logic Description:**

- This function is used by `update_campaign_status_admin()` & `update_campaign_status_cust()`

### 6.5.8 common\_campaign\_status

`dialer_campaign.views.common_campaign_status(pk, status)`

Campaign Status (e.g. start | stop | abort | pause) needs to be changed. It is a common function for the admin and customer UI's

**Attributes:**

- pk - primary key of the campaign record
- status - selected status for the campaign record

**Logic Description:**

- Selected Campaign's status needs to be changed. Changed status can be start, stop or pause.
- This function is used by `update_campaign_status_admin()` & `update_campaign_status_cust()`

### 6.5.9 phonebook\_list

`dialer_campaign.views.phonebook_list(request, *args, **kwargs)`

Phonebook list for the logged in user

**Attributes:**

- template - frontend/phonebook/list.html

**Logic Description:**

- List all phonebooks which belong to the logged in user.

### 6.5.10 phonebook\_grid

`dialer_campaign.views.phonebook_grid(request, *args, **kwargs)`

Phonebook list in json format for flexigrid.

**Model:** Phonebook

**Fields:** [id, name, description, updated\_date]

### 6.5.11 phonebook\_add

`dialer_campaign.views.phonebook_add(request, *args, **kwargs)`

Add new Phonebook for the logged in user

**Attributes:**

- form - PhonebookForm
- template - frontend/phonebook/change.html

**Logic Description:**

- Add a new phonebook which will belong to the logged in user via the phonebookForm & get redirected to the phonebook list

### 6.5.12 phonebook\_change

`dialer_campaign.views.phonebook_change(request, *args, **kwargs)`

Update/Delete Phonebook for the logged in user

**Attributes:**

- `object_id` - Selected phonebook object
- `form` - PhonebookForm
- `template` - `frontend/phonebook/change.html`

**Logic Description:**

- Update/delete selected phonebook from the phonebook list via PhonebookForm & get redirected to phonebook list

### 6.5.13 contact\_list

`dialer_campaign.views.contact_list(request, *args, **kwargs)`

Contact list for the logged in user

**Attributes:**

- `template` - `frontend/contact/list.html`
- `form` - ContactSearchForm

**Logic Description:**

- List all contacts from phonebooks belonging to the logged in user

### 6.5.14 contact\_grid

`dialer_campaign.views.contact_grid(request, *args, **kwargs)`

Contact list in json format for flexigrid

**Model:** Contact

**Fields:** [`id`, `phonebook__name`, `contact`, `last_name`, `first_name`, `description`, `status`, `additional_vars`, `updated_date`]

### 6.5.15 contact\_add

`dialer_campaign.views.contact_add(request, *args, **kwargs)`

Add a new contact into the selected phonebook for the logged in user

**Attributes:**

- `form` - ContactForm
- `template` - `frontend/contact/change.html`

**Logic Description:**

- Before adding a contact, check dialer setting limit if applicable to the user.
- Add new contact belonging to the logged in user via ContactForm & get redirected to the contact list

### 6.5.16 `contact_change`

`dialer_campaign.views.contact_change(request, *args, **kwargs)`

Update/Delete contact for the logged in user

**Attributes:**

- `object_id` - Selected contact object
- `form` - `ContactForm`
- `template` - `frontend/contact/change.html`

**Logic Description:**

- Update/delete selected contact from the contact list via `ContactForm` & get redirected to the contact list

### 6.5.17 `contact_import`

`dialer_campaign.views.contact_import(request, *args, **kwargs)`

Import CSV file of Contacts for the logged in user

**Attributes:**

- `form` - `Contact_fileImport`
- `template` - `frontend/contact/import_contact.html`

**Logic Description:**

- Before adding contacts, check dialer setting limit if applicable to the user.
- Add new contacts which will belong to the logged in user via csv file & get the result (upload success and failure statistics)

**Important variable:**

- `total_rows` - Total no. of records in the CSV file
- `retail_record_count` - No. of records imported from the CSV file

### 6.5.18 `campaign_list`

`dialer_campaign.views.campaign_list(request, *args, **kwargs)`

List all campaigns for the logged in user

**Attributes:**

- `template` - `frontend/campaign/list.html`

**Logic Description:**

- List all campaigns belonging to the logged in user

### 6.5.19 `campaign_grid`

`dialer_campaign.views.campaign_grid(request, *args, **kwargs)`

Campaign list in json format for flexigrid

**Model:** Campaign



**Fields:** [id, campaign\_code, name, startingdate, expirationdate, aleg\_gateway, aleg\_gateway\_\_name, status, voipapp\_\_name]

### 6.5.20 campaign\_add

`dialer_campaign.views.campaign_add(request, *args, **kwargs)`

Add a new campaign for the logged in user

**Attributes:**

- `form` - CampaignForm
- `template` - frontend/campaign/change.html

**Logic Description:**

- Before adding a campaign, check dialer setting limit if applicable to the user.
- Add the new campaign which will belong to the logged in user via CampaignForm & get redirected to campaign list

### 6.5.21 campaign\_change

`dialer_campaign.views.campaign_change(request, *args, **kwargs)`

Update/Delete campaign for the logged in user

**Attributes:**

- `object_id` - Selected campaign object
- `form` - CampaignForm
- `template` - frontend/campaign/change.html

**Logic Description:**

- Update/delete selected campaign from the campaign list via CampaignForm & get redirected to the campaign list

## 6.6 Newfies-Dialer Admin Views

### 6.6.1 CampaignAdmin

`class dialer_campaign.admin.CampaignAdmin(model, admin_site)`

Allows the administrator to view and modify certain attributes of a Campaign.

**add\_view** (*request, extra\_context=None*)

Override django add\_view method for checking the dialer setting limit

**Logic Description:**

- Before adding campaign, checked dialer setting limit if applicable to the user, if matched, the user will be redirected to the campaign list

**form**

alias of CampaignAdminForm

### 6.6.2 PhonebookAdmin

**class** dialer\_campaign.admin.**PhonebookAdmin** (*model, admin\_site*)  
Allows the administrator to view and modify certain attributes of a Phonebook.

### 6.6.3 ContactAdmin

**class** dialer\_campaign.admin.**ContactAdmin** (*model, admin\_site*)  
Allows the administrator to view and modify certain attributes of a Contact.

**add\_view** (*request, extra\_context=None*)  
Override django admin add\_view method for checking the dialer setting limit

**Logic Description:**

- Before adding a contact, check the dialer setting limit if applicable to the user. If matched, the user will be redirected to the contact list

**import\_contact** (*request*)  
Add custom method in django admin view to import CSV file of Contacts

**Attributes:**

- form - Contact\_fileImport
- template - admin/dialer\_campaign/contact/import\_contact.html

**Logic Description:**

- Before adding contact, check the dialer setting limit if applicable to the user.
- Add a new contact which will belong to the logged in user via csv file & get the result (Upload success & failure statistics)

**Important variable:**

- total\_rows - Total no. of records in the CSV file
- retail\_record\_count - No. of records which are imported from The CSV file

### 6.6.4 CampaignSubscriberAdmin

**class** dialer\_campaign.admin.**CampaignSubscriberAdmin** (*model, admin\_site*)  
Allows the administrator to view and modify certain attributes of a CampaignSubscriber.

### 6.6.5 CallrequestAdmin

**class** dialer\_cdr.admin.**CallrequestAdmin** (*model, admin\_site*)  
Allows the administrator to view and modify certain attributes of a Callrequest.

### 6.6.6 VoIPCallAdmin

**class** dialer\_cdr.admin.**VoIPCallAdmin** (*model, admin\_site*)  
Allows the administrator to view and modify certain attributes of a VoIPCall.

**changelist\_view** (*request*, *extra\_context=None*)

Override changelist\_view method of django-admin for search parameters

**Attributes:**

- **form** - VoipSearchForm
- **template** - admin/dialer\_cdr/voipcall/change\_list.html

**Logic Description:**

- VoIP report Record Listing with search option & Daily Call Report search Parameters: by date, by status and by billed.

**export\_voip\_report** (*request*)

Export a CSV file of VoIP call records

**Important variable:**

- **request.session['voipcall\_record\_qs']** - stores voipcall query set

**Exported fields:** [**user**, **callid**, **callerid**, **phone\_number**, **starting\_date**, **duration**, **disposition**, **used\_gateway**]

**has\_add\_permission** (*request*)

Remove add permission on VoIP Call Report model

**Logic Description:**

- Override django admin has\_add\_permission method to remove add permission on VoIP Call Report model

**used\_gateway\_link** (*obj*)

Used gateway link to edit gateway detail

**user\_link** (*obj*)

User link to user profile

## 6.6.7 GatewayAdmin

**class** dialer\_gateway.admin.**GatewayAdmin** (*model*, *admin\_site*)

Allows the administrator to view and modify certain attributes of a Gateway.

## 6.6.8 DialerSettingAdmin

**class** dialer\_settings.admin.**DialerSettingAdmin** (*model*, *admin\_site*)

Allows the administrator to view and modify certain attributes of a DialerSetting.

**add\_view** (*request*, *extra\_context=None*)

Add Dialer setting

**change\_view** (*request*, *object\_id*, *extra\_context=None*)

Edit dialer settings

**changelist\_view** (*request*, *extra\_context=None*)

Dialer setting list

## 6.7 Newfies Tasks

**class** `dialer_campaign.tasks.check_campaign_pendingcall`

This will execute the outbound calls in the campaign

**Attributes:**

- `campaign_id` - Campaign ID

**class** `dialer_campaign.tasks.campaign_running`

A periodic task that checks the campaign, create and tasks the calls

**Usage:**

`campaign_running.delay()`

**class** `dialer_campaign.tasks.collect_subscriber`

This task will collect all the subscribers

**Attributes:**

- `campaign_id` - Campaign ID

**class** `dialer_cdr.tasks.init_callrequest`

This task outbounds the call

**Attributes:**

- `callrequest_id` - Callrequest ID

**class** `dialer_cdr.tasks.callrequest_pending`

A periodic task that checks for pending calls

**Usage:**

`callrequest_pending.delay()`

## 6.8 Newfies Signals

### 6.8.1 `post_save_add_contact`

`dialer_campaign.models.post_save_add_contact` (*sender, \*\*kwargs*)

A `post_save` signal is sent by the `Contact` model instance whenever it is going to save.

**Logic Description:**

- When a new contact is added into `Contact` model, active the campaign list will be checked with the contact status.
- If the active campaign list count is more than one & the contact is active, the contact will be added into `CampaignSubscriber` model.

## 6.9 Test Case Descriptions

### 6.9.1 Requirement

**Run/Start Celery:**

```
$ /etc/init.d/celery start
```

or:

```
$ python manage.py celeryd -l info
```

#### Run/Start Redis:

```
$ /etc/init.d/redis-server start
```

## 6.9.2 How to run test

### 1. Run Full Test Suit:

```
$ python manage.py test --verbosity=2
```

### 2. Run NewfiesTastypieApiTestCase:

```
$ python manage.py test dialer_cdr.NewfiesTastypieApiTestCase --verbosity=2
```

### 3. Run NewfiesAdminInterfaceTestCase:

```
$ python manage.py test dialer_cdr.NewfiesAdminInterfaceTestCase --verbosity=2
```

### 4. Run NewfiesCustomerInterfaceTestCase:

```
$ python manage.py test dialer_cdr.NewfiesCustomerInterfaceTestCase --verbosity=2
```

## 6.9.3 Tastypie API Test Case

```
class dialer_cdr.tests.NewfiesTastypieApiTestCase (methodName='runTest')
```

```
 Test cases for Newfies-Dialer API.
```

```
 test_create_answercall ()
```

```
 Test Function to create a answercall
```

```
 test_create_bulk_contact ()
```

```
 Test Function to bulk create contacts
```

```
 test_create_callrequest ()
```

```
 Test Function to create a callrequest
```

```
 test_create_campaign ()
```

```
 Test Function to create a campaign
```

```
 test_create_campaign_subscriber ()
```

```
 Test Function to create a campaign subscriber
```

```
 test_create_cdr ()
```

```
 Test Function to create a CDR
```

```
 test_create_hangupcall ()
```

```
 Test Function to create a answercall
```

```
 test_create_phonebook ()
```

```
 Test Function to create a phonebook
```

```
 test_delete_campaign ()
```

```
 Test Function to delete a campaign
```

```
test_delete_cascade_campaign ()
 Test Function to cascade delete a campaign

test_read_callrequest ()
 Test Function to get all callrequests

test_read_campaign ()
 Test Function to get all campaigns

test_read_campaign_subscriber ()
 Test Function to get all campaign subscriber

test_read_phonebook ()
 Test Function to get all phonebooks

test_update_campaign ()
 Test Function to update a campaign

test_update_campaign_subscriber ()
 Test Function to update a campaign subscriber
```

### 6.9.4 Admin Interface Test Case

```
class dialer_cdr.tests.NewfiesAdminInterfaceTestCase (methodName='runTest')
 Test cases for Newfies Admin Interface.

 setUp ()
 To create admin user

 test_admin_index ()
 Test Function to check Admin index page

 test_admin_newfies ()
 Test Function to check Newfies Admin pages
```

### 6.9.5 Customer Interface Test Case

```
class dialer_cdr.tests.NewfiesCustomerInterfaceTestCase (methodName='runTest')
 Test cases for Newfies Customer Interface.

 test_campaign_view ()
 Test Function to check phonebook

 test_contact_view ()
 Test Function to check Contact

 test_dashboard ()
 Test Function to check customer dashboard

 test_index ()
 Test Function to check customer index page

 test_phonebook_view ()
 Test Function to check phonebook

 test_user_settings ()
 Test Function to check User settings

 test_voip_call_report ()
 Test Function to check VoIP call report
```

```
test_voipapp_view()
 Test Function to check voipapp
```

### 6.9.6 Customer Interface Forgot Test Case

```
class dialer_cdr.tests.NewfiesCustomerInterfaceForgotPassTestCase(methodName='runTest')
 Test cases for Newfies Customer Interface. for forgot password

 test_check_password_reset()
 Test Function to check password reset
```





# API REFERENCE

Contents:

## 7.1 CampaignResource

`class api.resources.CampaignResource (api_name=None)`

**Attributes:**

- `campaign_code` - Autogenerate campaign code
- `name` - Name of the Campaign
- `description` - Short description of the Campaign
- `callerid` - Caller ID
- `startingdate` - Start date. Epoch Time, ie 1301414368
- `expirationdate` - Expiry date. Epoch Time, ie 1301414368
- `daily_start_time` - Daily start time, default '00:00:00'
- `daily_stop_time` - Daily stop time, default '23:59:59'
- `monday` - Set to 1 if you want to run this day of the week, default '1'
- `tuesday` - Set to 1 if you want to run this day of the week, default '1'
- `wednesday` - Set to 1 if you want to run this day of the week , default '1'
- `thursday` - Set to 1 if you want to run this day of the week, default '1'
- `friday` - Set to 1 if you want to run this day of the week, default '1'
- `saturday` - Set to 1 if you want to run this day of the week, default '1'
- `sunday` - Set to 1 if you want to run this day of the week, default '1'

**Campaign Settings:**

- `frequency` - Defines the frequency, speed of the campaign. This is the number of calls per minute.
- `callmaxduration` - Maximum call duration.
- `maxretry` - Defines the max retries allowed per user.
- `intervalretry` - Defines the time to wait between retries in seconds
- `calltimeout` - Defines the number of seconds to timeout on calls

**Gateways:**

- aleg\_gateway - Defines the Gateway to use to call the subscriber
- voipapp - Defines the application to use when the call is established on the A-Leg
- extra\_data - Defines the additional data to pass to the application

**Validation:**

- CampaignValidation()

**Create:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data
```

**Response:**

```
HTTP/1.0 201 CREATED
Date: Fri, 23 Sep 2011 06:08:34 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Type: text/html; charset=utf-8
Location: http://localhost:8000/api/app/campaign/1/
Content-Language: en-us
```

**Read:****CURL Usage:**

```
curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/campaig
```

**Response:**

```
{
 "meta":{
 "limit":20,
 "next":null,
 "offset":0,
 "previous":null,
 "total_count":1
 },
 "objects":[
 {
 "callerid":"123987",
 "callmaxduration":1800,
 "calltimeout":45,
 "campaign_code":"XIUER",
 "created_date":"2011-06-15T00:49:16",
 "daily_start_time":"00:00:00",
 "daily_stop_time":"23:59:59",
 "description":"",
 "expirationdate":"2011-06-22T00:01:15",
 "extra_data":"",
 "frequency":10,
 "friday":true,
 "id":"1",
 "intervalretry":3,
 "maxretry":3,
 "monday":true,
 "name":"Default_Campaign",
```

```
 "resource_uri":"/api/app/campaign/1/",
 "saturday":true,
 "startingdate":"2011-06-15T00:01:15",
 "status":1,
 "sunday":true,
 "thursday":true,
 "tuesday":true,
 "updated_date":"2011-06-15T00:49:16",
 "wednesday":true
 }
]
}
```

**Update:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type: application/json" -X PUT --data
```

**Response:**

```
HTTP/1.0 204 NO CONTENT
Date: Fri, 23 Sep 2011 06:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Length: 0
Content-Type: text/html; charset=utf-8
Content-Language: en-us
```

**Delete:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type: application/json" -X DELETE htt
curl -u username:password --dump-header - -H "Content-Type: application/json" -X DELETE htt
```

**Response:**

```
HTTP/1.0 204 NO CONTENT
Date: Fri, 23 Sep 2011 06:48:03 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Length: 0
Content-Type: text/html; charset=utf-8
Content-Language: en-us
```

**Search:****CURL Usage:**

```
curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/campaig
```

**Response:**

```
{
 "meta":{
 "limit":20,
 "next":null,
 "offset":0,
 "previous":null,
```

```
 "total_count":1
 },
 "objects":[
 {
 "aleg_gateway":{

 "created_date":"2011-06-15T00:28:52",
 "description":"",
 "id":"1",
 "maximum_call":null,
 "name":"Default_Gateway",
 },
 "callerid":"1239876",
 "callmaxduration":50,
 "calltimeout":45,
 "campaign_code":"DJZVK",
 "created_date":"2011-10-13T02:06:22",
 "daily_start_time":"00:00:00",
 "daily_stop_time":"23:59:59",
 "description":"",
 "expirationdate":"2011-03-28T17:08:56",
 "extra_data":"2000",
 "frequency":20,
 "friday":true,
 "id":"16",
 "intervalretry":3000,
 "maxretry":3,
 "monday":true,
 "name":"mycampaign2",
 "resource_uri":"/api/v1/campaign/16/",
 "saturday":true,
 "startingdate":"2011-03-29T09:48:56",
 "status":2,
 "sunday":true,
 "thursday":true,
 "tuesday":true,
 "updated_date":"2011-10-13T02:06:22",
 "user":{
 "id":"1",
 "username":"areski"
 },
 "voipapp":{
 "id":"1",
 "name":"Default_VoIP_App",
 },
 "wednesday":true
 }
]
}
```

## 7.2 CampaignDeleteCascadeResource

**class** `api.resources.CampaignDeleteCascadeResource` (*api\_name=None*)

**Attributes:**

- `campaign_id` - Campaign ID

**CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type: application/json" -X DELETE http://
```

**Example Response:**

```
HTTP/1.0 204 NO CONTENT
Date: Wed, 18 May 2011 13:23:14 GMT
Server: WSGIServer/0.1 Python/2.6.2
Vary: Authorization
Content-Length: 0
Content-Type: text/plain
```

## 7.3 PhonebookResource

```
class api.resources.PhonebookResource (api_name=None)
```

**Attributes:**

- name - Name of the Phonebook
- description - Short description of the Campaign
- campaign\_id - Campaign ID

**Validation:**

- PhonebookValidation()

**Create:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data
```

**Response:**

```
HTTP/1.0 201 CREATED
Date: Fri, 23 Sep 2011 06:08:34 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Type: text/html; charset=utf-8
Location: http://localhost:8000/api/app/phonebook/1/
Content-Language: en-us
```

**Read:****CURL Usage:**

```
curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/phonebo
```

**Response:**

```
{
 "meta":{
 "limit":20,
 "next":null,
 "offset":0,
 "previous":null,
 "total_count":1
 },
 "objects":[
```

```
{
 "created_date": "2011-04-08T07:55:05",
 "description": "This is default phone book",
 "id": "1",
 "name": "Default_Phonebook",
 "resource_uri": "/api/v1/phonebook/1/",
 "updated_date": "2011-04-08T07:55:05",
 "user": {
 "first_name": "",
 "id": "1",
 "last_login": "2011-10-11T01:03:42",
 "last_name": "",
 "resource_uri": "/api/v1/user/1/",
 "username": "areski"
 }
}
```

**Update:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type: application/json" -X PUT --data
```

**Response:**

```
HTTP/1.0 204 NO CONTENT
Date: Fri, 23 Sep 2011 06:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Length: 0
Content-Type: text/html; charset=utf-8
Content-Language: en-us
```

**Delete:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type: application/json" -X DELETE htt
curl -u username:password --dump-header - -H "Content-Type: application/json" -X DELETE htt
```

**Response:**

```
HTTP/1.0 204 NO CONTENT
Date: Fri, 23 Sep 2011 06:48:03 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Length: 0
Content-Type: text/html; charset=utf-8
Content-Language: en-us
```

**Search:****CURL Usage:**

```
curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/phonebo
```

## 7.4 BulkContactResource

**class** `api.resources.BulkContactResource` (*api\_name=None*)

API to bulk create contacts

### Attributes

- `contact` - contact number of the Subscriber
- `phonebook_id` - the phonebook Id to which we want to add the contact

### Validation:

- `BulkContactValidation()`

### CURL Usage:

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data '{"p
```

### Response:

```
HTTP/1.0 201 CREATED
Date: Thu, 13 Oct 2011 11:42:44 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Type: text/html; charset=utf-8
Location: http://localhost:8000/api/v1/bulkcontact/None/
Content-Language: en-us
```

## 7.5 CampaignSubscriberResource

**class** `api.resources.CampaignSubscriberResource` (*api\_name=None*)

### Attributes Details:

- `contact` - contact number of the Subscriber
- `last_name` - last name of the Subscriber
- `first_name` - first name of the Subscriber
- `email` - email id of the Subscriber
- `description` - Short description of the Subscriber
- `additional_vars` - Additional settings for the Subscriber
- `phonebook_id` - the phonebook Id to which we want to add the Subscriber

### Validation:

- `CampaignSubscriberValidation()`

### Create:

#### CURL Usage:

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data
```

#### Response:

```
HTTP/1.0 204 NO CONTENT
Date: Wed, 18 May 2011 13:23:14 GMT
Server: WSGIServer/0.1 Python/2.6.2
Vary: Authorization
Content-Length: 0
Content-Type: text/plain
```

**Read:**

**CURL Usage:**

```
curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/campaig
```

or

```
curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/campaig
```

**Response:**

**Update:**

**CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type: application/json" -X PUT --data
```

**Response:**

```
HTTP/1.0 204 NO CONTENT
Date: Fri, 23 Sep 2011 06:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Length: 0
Content-Type: text/html; charset=utf-8
Content-Language: en-us
```

## 7.6 CallrequestResource

**class** `api.resources.CallrequestResource` (*api\_name=None*)

**Attributes:**

- `callrequest_id` - Callrequest Id
- `request_uuid` -
- `call_time` -
- `call_type` -
- `timeout` -
- `timelimit` -
- `status` -
- `campaign_subscriber` -
- `campaign` -
- `voipapp` -
- `callerid` -



- phone\_number -
- extra\_dial\_string -
- extra\_data -
- num\_attempt -
- last\_attempt\_time -
- result -
- hangup\_cause -
- last\_attempt\_time -

**Validation:**

- CallrequestValidation()

**Create:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data
```

**Response:**

```
HTTP/1.0 201 CREATED
Date: Fri, 23 Sep 2011 06:08:34 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Type: text/html; charset=utf-8
Location: http://localhost:8000/api/app/campaign/1/
Content-Language: en-us
```

**Read:****CURL Usage:**

```
curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/callreq

curl -u username:password -H 'Accept: application/json' http://localhost:8000/api/v1/callreq
```

**Response:**

```
{
 "meta":{
 "limit":20,
 "next":null,
 "offset":0,
 "previous":null,
 "total_count":1
 },
 "objects":[
 {
 "call_time":"2011-10-20T12:21:22",
 "call_type":1,
 "callerid":"650784355",
 "created_date":"2011-10-14T07:33:41",
 "extra_data":"",
 "extra_dial_string":"",
 "hangup_cause":"",
 "id":"1",
```

```
 "last_attempt_time":null,
 "num_attempt":0,
 "phone_number":"8792749823",
 "request_uuid":"2342jtdsf-00123",
 "resource_uri":"/api/v1/callrequest/1/",
 "result":"",
 "status":1,
 "timelimit":3600,
 "timeout":30000,
 "updated_date":"2011-10-14T07:33:41",
 "user":{
 "first_name":"",
 "id":"1",
 "last_login":"2011-10-11T01:03:42",
 "last_name":"",
 "resource_uri":"/api/v1/user/1/",
 "username":"areski"
 },
 "voipapp":{
 "created_date":"2011-04-08T08:00:09",
 "data":"",
 "description":"",
 "id":"1",
 "name":"Default_VoIP_App",
 "resource_uri":"/api/v1/voipapp/1/",
 "type":1,
 "updated_date":"2011-10-14T07:33:41"
 }
}
]
```

**Update:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type: application/json" -X PUT --data
```

**Response:**

```
HTTP/1.0 204 NO CONTENT
Date: Fri, 23 Sep 2011 06:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Length: 0
Content-Type: text/html; charset=utf-8
Content-Language: en-us
```

## 7.7 CdrResource

```
class api.resources.CdrResource (api_name=None)
```

**Attributes:**

- cdr - XML string assigned from the Telephony engine

**Validation:**

- CdrValidation()

**Create:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data
```

**Response:**

```
HTTP/1.0 201 CREATED
Date: Fri, 23 Sep 2011 06:08:34 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Type: text/html; charset=utf-8
Location: http://localhost:8000/api/v1/store_cdr/None/
Content-Language: en-us
```

## 7.8 AnswercallResource

```
class api.resources.AnswercallResource (api_name=None)
```

**Attributes:**

- RequestUUID - A unique identifier for the API request.

**Create:****CURL Usage:**

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data
```

**Response:**

```
HTTP/1.0 200 OK
Date: Tue, 01 Nov 2011 11:30:59 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Type: application/json
Content-Language: en-us

<?xml version="1.0" encoding="utf-8"?>
 <Response>
 <Dial timeLimit="3600" callerId="650784355">
 <Number gateways="user/,user" gatewayTimeouts="30000"></Number>
 </Dial>
 </Response>
```

## 7.9 HangupcallResource

```
class api.resources.HangupcallResource (api_name=None)
```

**Attributes:**

- RequestUUID - RequestUUID
- HangupCause - Hangup Cause

**Validation:**

- HangupcallValidation()

### Create:

#### CURL Usage:

```
curl -u username:password --dump-header - -H "Content-Type:application/json" -X POST --data
```

#### Response:

```
HTTP/1.0 200 OK
Date: Tue, 01 Nov 2011 12:04:35 GMT
Server: WSGIServer/0.1 Python/2.7.1+
Vary: Accept-Language, Cookie
Content-Type: application/json
Content-Language: en-us

<?xml version="1.0" encoding="utf-8"?>
 <Response>
 </Response>
```

# CONTRIBUTING

- [Community Code of Conduct](#)
- [Reporting a Bug](#)
- [Coding Style](#)

## 8.1 Community Code of Conduct

Members of our community need to work together effectively, and this code of conduct lays down the ground rules for our cooperation.

Please read the following documentation about how the Newfies-Dialer Project functions, coding styles expected for contributions, and the community standards we expect everyone to abide by.

The Code of Conduct is heavily based on the [Ubuntu Code of Conduct](#), [Celery Code of Conduct](#), and the [Pylons Code of Conduct](#).

### 8.1.1 Be considerate.

Your work will be used by other people, and you in turn will depend on the work of others. Any decision you take will affect users and colleagues, and we expect you to take those consequences into account when making decisions. Even if it's not obvious at the time, our contributions to Newfies-Dialer will impact the work of others. For example, changes to code, infrastructure, policy, documentation and translations during a release may negatively impact others work.

### 8.1.2 Be respectful.

The Newfies-Dialer community and its members treat one another with respect. Everyone can make a valuable contribution to Newfies-Dialer. We may not always agree, but disagreement is no excuse for poor behaviour and bad manners. We might all experience some frustration now and then, but we cannot allow that frustration to turn into a personal attack. It's important to remember that a community where people feel uncomfortable or threatened is not a productive one. We expect members of the Newfies-Dialer community to be respectful when dealing with other contributors as well as with people outside the Newfies-Dialer project and with users of Newfies-Dialer.

### 8.1.3 Be collaborative.

Collaboration is central to Newfies-Dialer and to the larger free software community. We should always be open to collaboration. Your work should be done transparently and patches from Newfies-Dialer should be given back to the community when they are made, not just when the distribution is released. If you wish to work on new code for existing upstream projects, at least keep those projects informed of your ideas and progress. It may not be possible to get consensus from upstream, or even from your colleagues about the correct implementation for an idea, so don't feel obliged to have that agreement before you begin, but at least keep the outside world informed of your work, and publish your work in a way that allows outsiders to test, discuss and contribute to your efforts.

### 8.1.4 When you disagree, consult others.

Disagreements, both political and technical, happen all the time and the Newfies-Dialer community is no exception. It is important that we resolve disagreements and differing views constructively and with the help of the community and community process. If you really want to go a different way, then we encourage you to make a derivative distribution or alternate set of packages that still build on the work we've done to utilise as common a core as possible.

### 8.1.5 When you are unsure, ask for help.

Nobody knows everything, and nobody is expected to be perfect. Asking questions avoids many problems down the road, and so questions are encouraged. Those who are asked questions should be responsive and helpful. However, when asking a question, care must be taken to do so in an appropriate forum.

### 8.1.6 Step down considerably.

Developers on every project come and go and Newfies-Dialer is no different. When you leave or disengage from the project, in whole or in part, we ask that you do so in a way that minimises disruption to the project. This means you should tell people you are leaving and take the proper steps to ensure that others can pick up where you leave off.

## 8.2 Reporting a Bug

Bugs can always be described to the [Mailing list](#), but the best way to report an issue and to ensure a timely response is to use the issue tracker.

1. Create a GitHub account.

You need to [create a GitHub account](#) to be able to create new issues and participate in the discussion.

2. Determine if your bug is really a bug.

You should not file a bug if you are requesting support. For that you can use the [Mailing list](#).

3. Make sure your bug hasn't already been reported.

Search through the appropriate Issue tracker. If a bug like yours was found, check if you have new information that could be reported to help the developers fix the bug.

4. Collect information about the bug.

To have the best chance of having a bug fixed, we need to be able to easily reproduce the conditions that caused it. Most of the time this information will be from a Python traceback message, though some bugs might be in design, spelling or other errors on the website/docs/code.

If the error is from a Python traceback, include it in the bug report.

We also need to know what platform you're running (Windows, OSX, Linux, etc), the version of your Python interpreter, the version of Newfies-Dialer and related packages that you were running when the bug occurred.

5. Submit the bug.

By default [GitHub](#) will email you to let you know when new comments have been made on your bug. In the event you've turned this feature off, you should check back on occasions to ensure you don't miss any questions a developer trying to fix the bug might ask.

## 8.2.1 Issue Trackers

Bugs for a package in the Newfies-Dialer ecosystem should be reported to the relevant issue tracker.

- Newfies-Dialer: <http://github.com/Star2Billing/newfies-dialer/issues/>
- Celery: <https://github.com/ask/celery/issues/>
- Freeswitch: <http://jira.freeswitch.org/secure/Dashboard.jspa>
- Plivo: <https://github.com/plivo/plivo/issues/>

If you are unsure of the origin of the bug you can ask the [Mailing list](#), or just use the Newfies-Dialer issue tracker.

## 8.3 Coding Style

You should probably be able to pick up the coding style from surrounding code, but it is a good idea to be aware of the following conventions.

- All Python code must follow the [PEP-8](#) guidelines.

`pep8.py` is a utility you can use to verify that your code is following the conventions.

- Docstrings must follow the [PEP-257](#) conventions, and use the following style.

Do this:

```
def method(self, arg):
 """Short description.

 More details.

 """
```

or:

```
def method(self, arg):
 """Short description."""
```

but not this:

```
def method(self, arg):
 """
 Short description.
 """
```

- Lines should not exceed 78 columns.
- Wildcard imports must not be used (*from xxx import \**).





# FREQUENTLY ASKED QUESTIONS

- General
- Misconceptions

## 9.1 General

### 9.1.1 What is Newfies-Dialer?

**Answer:** .

Newfies-Dialer is a voice broadcast application designed and built to automate the delivery of interactive phone calls to contacts, clients and the general public.

### 9.1.2 Why should I use Newfies-Dialer?

**Answer:** .

Below are some examples of some of the uses that Newfies-Dialer can be put to. There are more details and examples at <http://www.newfies-dialer.org/solutions/>

- Telecasting:

Broadcast marketing or informational messages to customers and clients.

- Telemarketing:

Broadcast a marketing message to potential customers, and give them the option to be put through

- Phone Polling, Surveys and Voting:

Ring large numbers of people and present IVR options for either polling their opinions, interact

- Debt Control:

Customers can be automatically reminded at intervals that they owe money, and an IVR menu presen

- Appointment reminders:

Doctors, Dentists, and other organisations that make appointments for their clients can integrat

- Dissemination of Information by Phone:

Newfies-Dialer was originally designed to call large numbers of people and disseminate medical a

- Mass Emergency Broadcasting:

Where there is a necessity to warn large numbers of people in a short space of time, such as wea

- Voice Conferencing:

Attendees for a voice conference or podcast can be dialled up from a central location, and be co

- Subscription Reminders and Renewals:

Where a company sells an annual subscription for a product or service, Newfies-Dialer can be con

### 9.1.3 What's the history behind Newfies-Dialer?

**Answer:** .

Newfies-Dialer is a bulk dialer application which was commissioned by a charity named Kubatana (<http://www.kubatana.net>) based in Zimbabwe, which sponsors the Freedomfone project (<http://www.freedomfone.org/>) dedicated to providing information via phone technology.

In less economically developed countries, Internet is often limited, but there is usually comprehensive mobile phone coverage. Freedomfone uses Newfies-Dialer to dial up people's phones and offer health information on Cholera, Malaria and so many other avoidable health issues in the third world, which may be alleviated by education. Newfies-Dialer was so named after the Newfoundland Dog nicknamed Newfies and used by sea rescue services around the world.

## 9.2 Misconceptions

### 9.2.1 Is Newfies-Dialer dependent on Celery?

**Answer:** Yes.

# TROUBLESHOOTING

- Where to find help
- Where to find the log files
- How to run a quick test call
- Run in debug mode
- Celerymon
- How to discard all pending tasks
- Checking Plivo is running
- Checking Freeswitch
- Step By Step Checklist

## 10.1 Where to find help

### 10.1.1 Documentation:

<http://www.newfies-dialer.org/documentation/>

### 10.1.2 Mailing list:

We have set up a mailing list at <http://groups.google.com/group/newfies-dialer>

### 10.1.3 Forum:

We have a forum at <http://forum.newfies-dialer.org/>

### 10.1.4 Support:

Star2Billing S.L. offers consultancy including installation, training and customisation

## 10.2 Where to find the log files

All the logs are centralized into one single directory `/var/log/newfies/`

**newfies-django-db.log** : This contains all the Database queries performed by the UI

**newfies-django.log** : All the logger events from Django

**err-apache-newfies.log** : Any apache errors pertaining to Newfies-Dialer

**celery-newfies-node1.log** : This contains celery activity

## 10.3 How to run a quick test call

Go on the admin panel and check if there is any call request that has been spooled.

- [http://your-ip:8008/admin/dialer\\_cdr/callrequest/](http://your-ip:8008/admin/dialer_cdr/callrequest/)

If there are no calls queued, this means that the campaign is not properly configured.

You should:

1. Check if the campaign is started that the “Start time”, “Finish Time” and server time are correct.
2. Make sure that you configured a Dialer Setting for the user running the campaign, although there will be a warning for this on the Customer UI : [http://your-ip:8008/admin/dialer\\_settings/dialersetting/](http://your-ip:8008/admin/dialer_settings/dialersetting/)

If there is an existing Call Request, check the status, and check the Celery log stored in /var/log/newfies

## 10.4 Run in debug mode

Make sure you stop the services first:

```
$ /etc/init.d/newfies-celeryd stop
```

Then run in debug mode:

```
$ workon newfies-dialer
$ cd /usr/share/newfies/
$ python manage.py celeryd -EB --loglevel=DEBUG
```

## 10.5 Celerymon

- <https://github.com/ask/celerymon>

Running the monitor :

**Start celery with the `-events` option on, so celery sends events for celerymon to capture::** `$ workon newfies-dialer $ cd /usr/share/newfies/ $ python manage.py celeryd -E`

Run the monitor server:

```
$ workon newfies-dialer
$ cd /usr/share/newfies/
$ python manage.py celerymon
```

However, in production you probably want to run the monitor in the background, as a daemon:

```
$ workon newfies-dialer
$ cd /usr/share/newfies/
$ python manage.py celerymon --detach
```

For a complete listing of the command line arguments available, with a short description, you can use the help command:

```
$ workon newfies-dialer
$ cd /usr/share/newfies/
$ python manage.py help celerymon
```

Now you can visit the webserver celerymon starts by going to: <http://localhost:8989>

## 10.6 How to discard all pending tasks

<http://docs.celeryproject.org/en/latest/faq.html?highlight=purge#how-do-i-discard-all-waiting-tasks>

## 10.7 Checking Plivo is running

At the command line, type:

```
$ ps aux | grep plivo
```

This should tell you that Plivo-rest, Plivo-Outbound and Plivo-cache are all running. If they are not, these services can be restarted with the following commands:

```
$ /etc/init.d/plivo stop
$ /etc/init.d/plivocache stop
$ /etc/init.d/plivo start
$ /etc/init.d/plivocache start
```

If there are still issues with Plivo, then check the logs for clues at **/usr/share/plivo/tmp/**

## 10.8 Checking Freeswitch

Entering the Freeswitch CLI should indicate whether it is running by typing `fs_cli` at the console. Once logged in, you can check the trunk registration by typing `sofia status` at the Freeswitch CLI. CTRL-D exits the Freeswitch CLI.

If the Freeswitch CLI cannot be launched, then the status of freeswitch can be checked with:

```
$ ps aux | grep freeswitch
or
$ /etc/init.d/freeswitch status
```

If Freeswitch is not running, then it can be started with

```
$ /etc/init.d/freeswitch start
```

## 10.9 Step By Step Checklist

The step by step checklist below should be used to validate that all components of the platform are running.

User interface :

- 1. Dialer Gateway matching a configured trunk is set up in the UI

- 2. Dialer Settings configured and attached to the appropriate user
- 3. Phonebook Created with contacts attached to the phonebook
- 4. Configured voice application
- 5. Campaign created, and started, with a phone book attached, and the campaign schedule current

Backend :

- 1. Celery Monitor Running
- 2. Plivo Running
- 3. Freeswitch running

If there are still problems, then raise a support question on the mailing-list <http://groups.google.com/group/newfies-dialer> or our forum, <http://forum.newfies-dialer.org/>, alternatively, contact [newfies-dialer@star2billing.com](mailto:newfies-dialer@star2billing.com) for commercial support.

# RESOURCES

- Getting Help
  - Mailing list
- Bug tracker
- Wiki
- Contributing
- License

## 11.1 Getting Help

### 11.1.1 Mailing list

For discussions about the usage, development, and future of Newfies-Dialer, please join the [Newfies-Dialer](#) mailing list.

## 11.2 Bug tracker

If you have any suggestions, bug reports or annoyances please report them to our issue tracker at <https://github.com/Star2Billing/newfies-dialer/issues/>

## 11.3 Wiki

<https://github.com/Star2Billing/newfies-dialer/wiki/>

## 11.4 Contributing

Development of *Newfies-Dialer* happens at Github: <https://github.com/Star2Billing/newfies-dialer>

You are highly encouraged to participate in the development of *Newfies-Dialer*. If you would prefer not to use Github, you are welcome to send us regular patches

Be sure to also read the [Contributing](#) section in the documentation.

## 11.5 License

This software is licensed under the *AGPL License*. See the `LICENSE` file in the top distribution directory for the full license text.



# INDICES AND TABLES

- *genindex*
- *modindex*
- *search*



# PYTHON MODULE INDEX

## a

`api.resources`, 73

## d

`dialer_campaign.admin`, 61

`dialer_campaign.models`, 64

`dialer_campaign.tasks`, 64

`dialer_campaign.views`, 56

`dialer_cdr.admin`, 62

`dialer_cdr.models`, 52

`dialer_cdr.tests`, 65

`dialer_gateway.admin`, 63

`dialer_gateway.models`, 54

`dialer_settings.admin`, 63

`dialer_settings.models`, 55

## u

`user_profile.models`, 55



# INDEX

## A

`add_view()` (dialer\_campaign.admin.CampaignAdmin method), 61  
`add_view()` (dialer\_campaign.admin.ContactAdmin method), 62  
`add_view()` (dialer\_settings.admin.DialerSettingAdmin method), 63  
`AnswercallResource` (class in `api.resources`), 79  
`api.resources` (module), 69, 72–76, 78, 79

## B

`BulkContactResource` (class in `api.resources`), 75

## C

`Callrequest` (class in `dialer_cdr.models`), 53  
`callrequest_pending` (class in `dialer_cdr.tasks`), 64  
`CallrequestAdmin` (class in `dialer_cdr.admin`), 62  
`CallrequestResource` (class in `api.resources`), 76  
`Campaign` (class in `dialer_campaign.models`), 51  
`campaign_add()` (in module `dialer_campaign.views`), 61  
`campaign_change()` (in module `dialer_campaign.views`), 61  
`campaign_grid()` (in module `dialer_campaign.views`), 60  
`campaign_list()` (in module `dialer_campaign.views`), 60  
`campaign_running` (class in `dialer_campaign.tasks`), 64  
`CampaignAdmin` (class in `dialer_campaign.admin`), 61  
`CampaignDeleteCascadeResource` (class in `api.resources`), 72  
`CampaignManager` (class in `dialer_campaign.models`), 51  
`CampaignResource` (class in `api.resources`), 69  
`CampaignSubscriber` (class in `dialer_campaign.models`), 52  
`campaignsubscriber_detail()` (`dialer_campaign.models.Campaign` method), 52  
`CampaignSubscriberAdmin` (class in `dialer_campaign.admin`), 62  
`CampaignSubscriberResource` (class in `api.resources`), 75  
`CdrResource` (class in `api.resources`), 78  
`change_view()` (`dialer_settings.admin.DialerSettingAdmin` method), 63

`changelist_view()` (`dialer_cdr.admin.VoIPCallAdmin` method), 62  
`changelist_view()` (`dialer_settings.admin.DialerSettingAdmin` method), 63  
`check_campaign_pendingcall` (class in `dialer_campaign.tasks`), 64  
`collect_subscriber` (class in `dialer_campaign.tasks`), 64  
`common_campaign_status()` (in module `dialer_campaign.views`), 58  
`common_send_notification()` (in module `dialer_campaign.views`), 57  
`Contact` (class in `dialer_campaign.models`), 50  
`contact_add()` (in module `dialer_campaign.views`), 59  
`contact_change()` (in module `dialer_campaign.views`), 60  
`contact_grid()` (in module `dialer_campaign.views`), 59  
`contact_import()` (in module `dialer_campaign.views`), 60  
`contact_list()` (in module `dialer_campaign.views`), 59  
`contact_name()` (`dialer_campaign.models.Contact` method), 51  
`ContactAdmin` (class in `dialer_campaign.admin`), 62  
`count_contact_of_phonebook()` (`dialer_campaign.models.Campaign` method), 52  
`cust_password_reset()` (in module `dialer_campaign.views`), 57  
`cust_password_reset_confirm()` (in module `dialer_campaign.views`), 57  
`cust_password_reset_done()` (in module `dialer_campaign.views`), 57  
`customer_dashboard()` (in module `dialer_campaign.views`), 56

## D

`destination_name()` (`dialer_cdr.models.VoIPCall` method), 54  
`dialer_campaign.admin` (module), 61  
`dialer_campaign.models` (module), 50, 64  
`dialer_campaign.tasks` (module), 64  
`dialer_campaign.views` (module), 56  
`dialer_cdr.admin` (module), 62  
`dialer_cdr.models` (module), 52  
`dialer_cdr.tests` (module), 65

dialer\_gateway.admin (module), 63

dialer\_gateway.models (module), 54

dialer\_settings.admin (module), 63

dialer\_settings.models (module), 55

DialerSetting (class in dialer\_settings.models), 55

DialerSettingAdmin (class in dialer\_settings.admin), 63

## E

export\_voip\_report() (dialer\_cdr.admin.VoIPCallAdmin method), 63

## F

form (dialer\_campaign.admin.CampaignAdmin attribute), 61

## G

Gateway (class in dialer\_gateway.models), 54

GatewayAdmin (class in dialer\_gateway.admin), 63

get\_active\_callmaxduration() (dialer\_campaign.models.Campaign method), 52

get\_active\_contact() (dialer\_campaign.models.Campaign method), 52

get\_active\_contact\_no\_subscriber() (dialer\_campaign.models.Campaign method), 52

get\_active\_max\_frequency() (dialer\_campaign.models.Campaign method), 52

get\_expired\_campaign() (dialer\_campaign.models.CampaignManager method), 51

get\_pending\_subscriber() (dialer\_campaign.models.Campaign method), 52

get\_running\_campaign() (dialer\_campaign.models.CampaignManager method), 51

## H

HangupcallResource (class in api.resources), 79

has\_add\_permission() (dialer\_cdr.admin.VoIPCallAdmin method), 63

## I

import\_contact() (dialer\_campaign.admin.ContactAdmin method), 62

index() (in module dialer\_campaign.views), 56

init\_callrequest (class in dialer\_cdr.tasks), 64

is\_authorized\_contact() (dialer\_campaign.models.Campaign method), 52

## L

login\_view() (in module dialer\_campaign.views), 57

## M

min\_duration() (dialer\_cdr.models.VoIPCall method), 54

## N

NewfiesAdminInterfaceTestCase (class in dialer\_cdr.tests), 66

NewfiesCustomerInterfaceForgotPassTestCase (class in dialer\_cdr.tests), 67

NewfiesCustomerInterfaceTestCase (class in dialer\_cdr.tests), 66

NewfiesTastypieApiTestCase (class in dialer\_cdr.tests), 65

## P

Phonebook (class in dialer\_campaign.models), 50

phonebook\_add() (in module dialer\_campaign.views), 58

phonebook\_change() (in module dialer\_campaign.views), 59

phonebook\_contacts() (dialer\_campaign.models.Phonebook method), 50

phonebook\_grid() (in module dialer\_campaign.views), 58

phonebook\_list() (in module dialer\_campaign.views), 58

PhonebookAdmin (class in dialer\_campaign.admin), 62

PhonebookResource (class in api.resources), 73

post\_save\_add\_contact() (in module dialer\_campaign.models), 64

progress\_bar() (dialer\_campaign.models.Campaign method), 52

## S

setUp() (dialer\_cdr.tests.NewfiesAdminInterfaceTestCase method), 66

## T

test\_admin\_index() (dialer\_cdr.tests.NewfiesAdminInterfaceTestCase method), 66

test\_admin\_newfies() (dialer\_cdr.tests.NewfiesAdminInterfaceTestCase method), 66

test\_campaign\_view() (dialer\_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66

test\_check\_password\_reset() (dialer\_cdr.tests.NewfiesCustomerInterfaceForgotPassTestCase method), 67

test\_contact\_view() (dialer\_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66

test_create_answercall() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	test_user_settings() aler_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66	(di-
test_create_bulk_contact() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	test_voip_call_report() aler_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66	(di-
test_create_callrequest() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	test_voipapp_view() aler_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66	(di-
test_create_campaign() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	<b>U</b>	
test_create_campaign_subscriber() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	update_campaign_status() aler_campaign.models.Campaign method), 52	(di-
test_create_cdr() (dialer_cdr.tests.NewfiesTastypieApiTestCase method), 65		used_gateway_link() (dialer_cdr.admin.VoIPCallAdmin method), 63	
test_create_hangupcall() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	user_link() (dialer_cdr.admin.VoIPCallAdmin method), 63	
test_create_phonebook() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	user_profile.models (module), 55	
test_dashboard() (dialer_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66		UserProfile (class in user_profile.models), 55	
test_delete_campaign() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	<b>V</b>	
test_delete_cascade_campaign() aler_cdr.tests.NewfiesTastypieApiTestCase method), 65	(di-	VoIPCall (class in dialer_cdr.models), 53	
test_index() (dialer_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66		VoIPCallAdmin (class in dialer_cdr.admin), 62	
test_phonebook_view() aler_cdr.tests.NewfiesCustomerInterfaceTestCase method), 66	(di-		
test_read_callrequest() aler_cdr.tests.NewfiesTastypieApiTestCase method), 66	(di-		
test_read_campaign() aler_cdr.tests.NewfiesTastypieApiTestCase method), 66	(di-		
test_read_campaign_subscriber() aler_cdr.tests.NewfiesTastypieApiTestCase method), 66	(di-		
test_read_phonebook() aler_cdr.tests.NewfiesTastypieApiTestCase method), 66	(di-		
test_update_campaign() aler_cdr.tests.NewfiesTastypieApiTestCase method), 66	(di-		
test_update_campaign_subscriber() aler_cdr.tests.NewfiesTastypieApiTestCase method), 66	(di-		