radioco Documentation

Release 4.0

Iago Veloso Abalo

Oct 28, 2018

Contents

CHAPTER 1

Overview

RadioCo is a broadcasting radio recording scheduling system. RadioCo has been intended to provide a solution for a wide range of broadcast projects, from community to public and commercial stations.

Here are a few of the key features:

- designed to work with any web browser
- drag and drop scheduling calendar interface
- live shows can be recorded and published automatically
- complete authentication system (user accounts, groups, permissions)
- . . . and much more

Note: This software is not for live streaming

CHAPTER 2

Release Notes

This document refers to version 4.0

CHAPTER 3

Table of contents

3.1 Installation Guide

The pages in this section of the documentation are designed to help you get started quickly and show how easy it is to work with RadioCo

The guides follow a logical progression and build on each other, so it's recommended to work through them in the order presented here.

3.1.1 Installing web application

To allow RadioCo to generate correct dates it's necessary to set the timezone variable:

1. Find your timezone in this list
2. Go to the config folder (radioco/configs/base) and create if not exists a local_settings.py in the same directory than settings.py
3. Add to the local settings the variable timezone, for example: TIME_ZONE = "Europe/Madrid"

Installing locally

This tutorial is written for Python 2.7 and Ubuntu 12.04 or later.

Ubuntu

The easiest way of installing the app is using Docker engine, follow the installation steps to install Docker.

Open a terminal and introduce the following commands:

```
sudo apt-get install git-core python-dev python-pip
```

Next, download the project and cd into it:

```
git clone https://github.com/iago1460/django-radio
cd django-radio
```

Execute the following commands to deploy the app in docker, this step take some time:

```
chmod +x ./run
./run start_dev
./run manage create_example_data
```

Warning: If you have faced the error "ValueError: unknown locale: UTF-8" on MacOS X, execute:

export LC_ALL=en_US.UTF-8 export LANG=en_US.UTF-8

Testing

Let's verify your installation

Now that the server's running, visit http://127.0.0.1:8000/

Also if you want to access the admin area the default credentials are *admin/1234*

Warning: Don't use this server in anything resembling a production environment.

Using RadioCo on production

The Internet is a hostile environment. Before deploying this project, you should take some time to review your settings, with security, performance, and operations in mind. Keep in mind this critical settings.

Locally

RadioCo provides a staging environment, safer than the previous one but still insecure, **use at your own risk**.

```
./run start
```

Now that the server's running, visit http://127.0.0.1:8000/

3.1.2 Installing recorder program

This tutorial is written for Python 2.7 and Ubuntu 12.04 or later.

Installing on Ubuntu

We'll get started by setting up our environment.

```
sudo apt-get install python-dev python-pip python-virtualenv git-core alsa-utils␣
↪vorbis-tools
```

Next, download the project and create the virual environment:

```
git clone https://github.com/iago1460/django-radio-recorder.git
cd django-radio-recorder
```

Create and activate a virtual env:

```
virtualenv venv
source venv/bin/activate
```

Install the requirements:

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

Using your favorite text editor, configure the `settings.ini` file

Launch the program

```
python main.py
```

3.1.3 Communication Setup

In your Admin interface go to **Podcast Configuration**, copy the **Recorder token** and put it into the Recorder Program settings:

```
token:8fdde6d703c05773084ea83e5ec2da62637666a0 #for example
```

Modify the **url** in your Recorder Program settings:

```
url:http://yourdomain:80/api/1/
```

3.2 Customize

Technical reference material.

RadioCo has a number of settings to configure its behaviour, please read this section carefully.

3.2.1 Application Setup

RadioCo can be adapted to your needs, you have three ways to do it. Editing the Global Settings section in the administration page using your browser or manually overriding the settings.py and overriding templates.

Global settings

Go to the admin section on your browser and edit the information available.

Global Configuration

In this section you can add information related to your site apart of the google analytics id.

Calendar Configuration

Settings related to the calendar, things like the first day of the week

Podcast Configuration

Settings related with the recorder, change here recording delays and get the Recorder token necessary for the recorder programme to work

Site

Change `example.com` with the fully qualified domain name associated with the website. Note that some RSS feed attributes will be incorrect if this value is not properly set up.

Templates

There is a empty folder called templates inside the radioco folder. You should override templates here, make sure to keep the relative path.

For example, to override the episode detail page copy the episode_detail.html file from radioco/apps/programmes/templates/programmes/episode_detail.html to radioco/templates/programmes/episode_detail.html

Settings.py

These settings are available in `settings.py`. Your settings should be in a `local_settings.py` file in the same directory as `settings.py`.

> **Warning:** Your changes on settings should be in `local_settings.py` to avoid conflicts when update, create that file if it's necessary. Be aware that this file is excluded from Git.

USERNAME_RADIOCO_RECORDER

This specifies who is the user of the recorder program:

```
USERNAME_RADIOCO_RECORDER = 'RadioCo_Recorder'
```

Note: It's a good idea change this value for security reasons.

LANGUAGE_CODE

A string representing the language code for this installation. It provides a fallback language in case the user's preferred language can't be determined or is not supported by the website. More info:

```
LANGUAGE_CODE = 'en'
```

PROGRAMME_LANGUAGES

New in version 1.1

Default: A tuple of the following three languages.

This specifies which languages are available for language selection in your programmes:

```
gettext_noop = lambda s: s

PROGRAMME_LANGUAGES = (
    ('es', gettext_noop('Spanish')),
    ('en', gettext_noop('English')),
    ('gl', gettext_noop('Galician')),
)
```

You can see the current list of translated languages by looking in django/conf/global_settings.py (or view the online source).

Disqus

New in version 2.0

Default: Disabled by default.

Add comments to your site with Disqus. Create your account and get your API key.:

```
DISQUS_ENABLE = True
DISQUS_API_KEY = 'YOUR_API_KEY'
DISQUS_WEBSITE_SHORTNAME = 'YOUR_SHORTNAME'
```

3.2.2 Recorder Program Setup

The settings are available in your settings.ini.

3.3 API

Api documentation.

3.3.1 Browsable API

Experimental in version 3.0

RadioCo has a Web browsable API, go to http://127.0.0.1:8000/api/2/ in your browser to explore it.

Optionally these endpoints support filtering and ordering in the majority of the exposed fields.

Programmes

Programmes can be filter using after and before parameters as well as Transmissions.

Example query to get all available programmes on New Year's Eve order by name:

```
http://127.0.0.1:8000/api/2/programmes?after=2016-12-31&before=2016-12-31&
↪ordering=name
```

Transmissions

Transmissions are always ordered by date, the after and before parameters are required.

Example query:

```
http://127.0.0.1:8000/api/2/transmissions?after=2016-12-19&before=2016-12-26
```

Also is possible to request the dates in a specific timezone:

```
http://127.0.0.1:8000/api/2/transmissions?timezone=Europe%2FMadrid&after=2016-12-19&
↪before=2016-12-26
```

Finally, there is a endpoint to get the current transmission:

```
http://127.0.0.1:8000/api/2/transmissions/now
```

3.3.2 Radiocom API

New in version 3.1

Radiocom is an Android application for Community Media Stations. RadioCo has the Radiocom API under the following urls:

```
http://127.0.0.1:8000/api/2/radiocom/programmes
http://127.0.0.1:8000/api/2/radiocom/radiostation
http://127.0.0.1:8000/api/2/radiocom/transmissions
http://127.0.0.1:8000/api/2/radiocom/transmissions/now
```

3.4 Development & community

RadioCo is an open-source project, and relies on its community of users to keep getting better.

You don't need to be an expert developer to make a valuable contribution - all you need is a little knowledge of the system, and a willingness to follow the contribution guidelines.

Remember that contributions to the documentation are highly prized, and key to the success of the project. Any time and effort you are willing to contribute is greatly appreciated!

3.4.1 Branch policy

- **master**: this is the current stable release, the version released on PyPI.
- **develop**: this branch always reflects a state with the latest delivered development changes for the next release.
- **feature branches**: these are used to develop new features.

3.4.2 Contributing Translation

For translators we have a Transifex account where you can translate the .po files and don't need to install git to be able to contribute. All changes there will be automatically sent to the project.

3.4.3 Contributing Documentation

Perhaps considered "boring" by hard-core coders, documentation is sometimes even more important than code! This is what brings fresh blood to a project, and serves as a reference for old timers. On top of this, documentation is the one area where less technical people can help most - you just need to write semi-decent English. People need to understand you. We don't care about style or correctness.

Documentation should be:

- written using valid Sphinx/restructuredText syntax (see below for specifics) and the file extension should be `.rst`
- written in English (we have standardised on British spellings)
- accessible - you should assume the reader to be moderately familiar with Python and Django, but not anything else. Link to documentation of libraries you use, for example, even if they are "obvious" to you

Merging documentation is pretty fast and painless.

Except for the tiniest of change, we recommend that you test them before submitting.

Documentation markup

Sections

We use Python documentation conventions for section marking:

- # with overline, for parts
- * with overline, for chapters
- =, for sections
- -, for subsections
- ^, for subsubsections
- ", for paragraphs

Inline markup

- **use backticks - ``settings.py`` - for:**
  - literals
  - filenames
  - names of fields and other items in the Admin interface:
- **use emphasis - *Home* around:**
  - the names of available options in the Admin
  - values in or of fields

References

Use absolute links to other documentation pages - `:doc:`/how_to/toolbar`` - rather than relative links - `:doc:`/../toolbar``. This makes it easier to run search-and-replaces when items are moved in the structure.

3.5 radioco

3.5.1 manage module

3.5.2 radioco package

Subpackages

radioco.apps package

Subpackages

radioco.apps.api package

Subpackages

radioco.apps.api.tests package

Submodules

radioco.apps.api.tests.test_api module

radioco.apps.api.tests.test_fullcalendar module

radioco.apps.api.tests.test_programmes module

radioco.apps.api.tests.test_radiocom_api module

radioco.apps.api.tests.test_recorder module

radioco.apps.api.tests.test_schedules module

Module contents

Submodules

radioco.apps.api.recorder_views module

radioco.apps.api.serializers module

radioco.apps.api.urls module

radioco.apps.api.views module

radioco.apps.api.viewsets module

Module contents

radioco.apps.global_settings package

Subpackages

It goes without saying that you should **backup your database** before embarking on any process that makes changes to your database.

3.6.1 1.1 release notes

What's new in 1.1

- Numerous updates to the documentation
- Updates to facilitate the starting-up (radioco_recorder_user and a empty shedule_board are self-created)
- Added custom languages to programmes (by default, all django languages)

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

Activate your virtualenv and do the following in your project main directory:

```
pip install django-radio==1.1
python manage.py migrate
```

3.6.2 1.1.1 release notes

What's new in 1.1.1

- Added CKEditor to programme's synopsis, episode's summary and user's biography
- Added default settings to the project
- Added customisable footer

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

Activate your virtualenv and do the following in your project main directory:

```
pip install django-radio==1.1.1
python manage.py collectstatic
python manage.py migrate
```

In your settings.py import default settings:

```
from radio.settings_base import *
```

3.6.3 1.2 release notes

What's new in 1.2

- Fixed project urls when RadioCo is deployed in a subdirectory

- Fixed default url of photos when STATIC_URL is not "/static/"
- Fixed errors 404 when users don't have a personal webpage.

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

Activate your virtualenv and do the following in your project main directory:

```
pip install django-radio==1.2
python manage.py migrate
```

3.6.4 2.1 release notes

What's new in 2.1

- New layout! The whole site has been redesigned
- Added image versions, RadioCo will adapt your images to fit in the new template
- Added comments support.
- Added analytics support.

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

You need to replace your current source with the content of https://github.com/iago1460/django-radio. To setup your settings please read the configuration section.

You should be able to keep your current database but make sure to create a backup before start.

```
pip install -r radio/configs/common/requirements.txt
python manage.py collectstatic
python manage.py migrate
```

3.6.5 3.0 release notes

What's new in 3.0

In this version RadioCo has almost been rewritten completely, internally the major improvements are:

- Use of time-zone-aware datetime objects (more info)
- Added good amount of tests
- Changed calendar to use API methods
- Using bower to download js and css dependencies
- Renamed internal folders and models
  - django-radio/radio –> django-radio/radioco

  - django-radio/radio/configs/common –> django-radio/radioco/configs/base
  - ScheduleBoard model has been renamed to Calendar
  - dashboard app has been merged into schedules

Special mention to @stefan-walluhn - Radio Corax who helped in this release, specifically adding Bower, Rest Framework API, tests and complex recurrence rules.

And the visible features are:

- Added download button to episodes (#38 Credit to @AlexCortinas)
- Removed unnecessary Scroll configuration setting (#13 Credit to @AlexCortinas)
- Improvements in title representation of episodes when not set (#17 Credit to @txenoo)
- Removed warnings when compiling (#32 Thanks to @txenoo)
- Added Radiocom endpoints (Thanks to @pablogrela)
- Added upload feature to CKEditor (#22)
- Added new calendar setting slotDuration
- Added a suite of helper tasks using invoke (Easy deploy to docker)
- Added public API to access to programmes, episodes, schedules and transmissions (#2)
- The schedule manager has received important updates (#20, #4)

Now managing schedules should be a lot more simple. From the manager is possible to create and edit programmes in addition to schedules.

The forced weekly recurrence has been removed in favor of complex recurrence rules, having a unique calendar should be enough.

The Calendar model doesn't have date constraints anymore and only one can be active at the same time. If you want to restrict a programme you should do it filling the start and end date fields inside itself.

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

Please **make sure of having a copy of your database** before even started to look to the next steps.

If you have custom code or templates I recommend putting them in a safe place, special mention to local_settings.py files, your are going to need to move them to the correct folder.

Since this migration introduces timezone support, it's necessary to set the timezone variable:

1. Find your timezone in this list
2. Go to the config folder and create if not exists a local_settings.py in the same directory than settings.py
3. Add to the local settings the variable timezone, for example: TIME_ZONE = "Europe/Madrid"

Warning: Using an incorrect timezone will cause incorrect dates, you should set/override that setting in your local_settings.py.

Warning: If you have a database backend different to PostgreSQL you must convert your data from local time to UTC. Fortunately for you a migration was written to fix episode issue_dates, other dates will suffer alterations.

Warning: We strongly recommend to use PostgreSQL as backend. Other database engines **will not be supported** in the future.

You have to install bower before running the following commands in the project root folder:

```
bower install
pip install -r radioco/configs/base/requirements.txt
python manage.py collectstatic
python manage.py migrate
```

After the migration process you will find that there is a new calendar set to active that contains all your schedules and your other calendars have been renamed with the prefix "Legacy - ". Feel free of delete these, they are not require anymore.

Note: Warning messages could appear during the migration process, it's usually safe to ignore them.

3.6.6 3.0.1 release notes

What's new in 3.0.1

- Fixing invoke commands.
- Simplified docker setup.

Take a look to our installation section.

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

You need to replace your current source with the content of https://github.com/iago1460/django-radio. To setup your settings please read the configuration section.

This update doesn't require any further steps.

3.6.7 3.1 release notes

What's new in 3.1

- Fixing staging docker port issue. (Thanks to @mmontes11 for report)
- Adding stable Radiocom API
- Solving ticket #43
- Upgrading libraries to last version

- Small fixes to invoke scripts
- Improving API (Outputting absolute urls and adding some fields)

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

You need to replace your current source with the content of https://github.com/iago1460/django-radio. To setup your settings please read the configuration section.

You should be able to keep your current database but make sure to create a backup before start.

```
pip install -r radioco/configs/base/requirements.txt
python manage.py collectstatic
python manage.py migrate  # check the following note in case of error
```

Note: When migrating from version 3.0 to version 3.1 the following error will appear during the migration process *django.db.utils.ProgrammingError: relation "recurrence_date" already exists* to solve this please run the following command:

```
python manage.py migrate --fake-initial
```

Or if you are using our docker setup:

```
inv docker.clean -e staging
inv docker.setup -e staging  # check the following note in case of error
```

Note: If a migration error appear during the setup please run:

```
inv docker.manage -e staging -c "migrate --fake-initial"
```

3.6.8 3.2 release notes

What's new in 3.2

- Solving ticket #5
- Improving rss (adding images and fixing itunes encoding)
- Added rss autodiscovery
- Documenting django.contrib.sites
- Improving ORM queries
- Adding cache to singleton models using memcached
- Improve performance in staging config using nginx cache, uwsgi and caching templates
- Updating Spanish and Galician translations
- Fixing progress bar in the main page

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

You need to replace your current source with the content of https://github.com/iago1460/django-radio. To setup your settings please read the configuration section.

You should be able to keep your current database but make sure to create a backup before start.

Warning: You must ensure your episodes and programmes only have one role per user. A database constraint has been put in place as part of this migration.

Warning: This migration depends on `settings.LANGUAGE_CODE`, it will convert the role field into a editable CharField using your preferred language. Note you can change this value temporally to run this migration. More info.

```
pip install -r radioco/configs/base/requirements.txt
python manage.py collectstatic
python manage.py migrate
```

Note: If you have a custom production environment please consider to apply our staging setup to improve performance.

Or if you are using our docker setup:

```
inv docker.clean -e staging
inv docker.setup -e staging
```

3.6.9 3.2.1 release notes

What's new in 3.2.1

- Making rss links protocol aware so they can be opened by podcast apps.

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

You need to replace your current source with the content of https://github.com/iago1460/django-radio. To setup your settings please read the configuration section.

This update doesn't require any further steps.

3.6.10 3.2.2 release notes

What's new in 3.2.2

- Fixing bug that made impossible deleting inline objects.
- Removing undocumented switch user functionality.
- Tweaking calendar css to improve readability.
- Filtering for active programmes in the home page.

Thanks to @beaulalume for reporting the previous issues.

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

You need to replace your current source with the content of https://github.com/iago1460/django-radio. To setup your settings please read the configuration section.

You should be able to keep your current database but make sure to create a backup before start.

```
python manage.py collectstatic
```

Or if you are using our docker setup:

```
inv docker.clean -e staging
inv docker.setup -e staging
```

3.6.11 4.0 release notes

What's new in 4.0

- Fix django-filebrowser (downgraded version) and ckeditor_uploader
- Simplifying config.
- Updating requirements.
- Upgrading to python3.6

How this affects you

If you're starting with a new installation, you don't need to worry about this. Don't even bother reading this section; it's for upgraders.

You need to replace your current source with the content of https://github.com/iago1460/django-radio. To setup your settings please read the configuration section.

You should be able to keep your current database but make sure to create a backup before start.

Running this project requires **python 3.6**

```
pip3 install -r requirements.txt
python3 manage.py collectstatic
```

Or if you are using our docker setup:

```
./run start
```

3.7 Indices and tables

- genindex
- modindex
- search