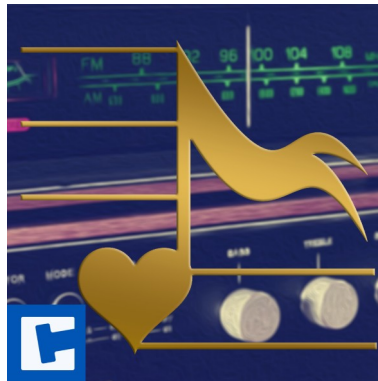


Radio PRO

Music matters



Documentation

Date: 15.03.2024

Version: 2024.1.2

© 2015-2024 **crosstales** LLC

<https://www.crosstales.com>

Table of Contents

1. Overview.....	4
2. Features.....	5
2.1. Radio stations.....	5
2.2. Flexible & expandable.....	5
2.3. Filter & order.....	5
2.4. OnRadio integration.....	6
2.5. Documentation & control.....	6
2.6. Compatibility.....	6
2.7. Integrations.....	7
3. Demonstration.....	8
3.1. PlayStations.....	8
3.2. StationList.....	9
3.3. PlayOwnStation.....	9
3.4. 3DAudio.....	10
3.5. Loudspeaker.....	10
4. Setup.....	11
4.1. Schema.....	11
4.2. Single radio-station (RadioPlayer).....	12
4.2.1. Parameter.....	13
4.3. Multiple radio-stations (RadioSet).....	14
4.4. SimplePlayer.....	15
4.5. Other components.....	16
4.5.1. StreamSaver.....	16
4.5.2. SurviveSceneSwitch.....	16
4.5.3. Loudspeaker.....	16
4.5.4. CrossFader.....	16
5. Providers.....	17
5.1. How-to create your own text resources.....	18
6. API.....	19
6.1. Players.....	19
6.1.1. Play.....	19
6.1.2. Stop.....	19
6.1.3. Restart.....	19
6.2. RadioManager.....	19
6.2.1. Next.....	19
6.2.2. Previous.....	20
6.2.3. NextStation.....	20
6.2.4. PreviousStation.....	20
6.2.5. StopAll.....	20
6.3. RadioProvider.....	20
6.3.1. Next.....	20
6.3.2. Previous.....	20
6.4. Callbacks.....	21
6.4.1. Playback start and end.....	21
6.4.2. Buffering start, end and progress.....	21
6.4.3. Audio start, end and time.....	21
6.4.4. Record change and time.....	22
6.4.5. Next record change and delay.....	22
6.4.6. Errors.....	22
6.4.7. Provider ready.....	23
6.4.8. Station change.....	23
6.4.9. Example.....	24

6.5. Complete API.....	24
7. Third-party support (PlayMaker etc.).....	25
8. Verify installation.....	25
9. Upgrade to new version.....	25
10. Important notes.....	25
10.1. Android.....	25
11. OnRadio.....	26
11.1. Step 1.....	26
11.2. Step 2.....	26
11.3. Step 3.....	26
12. Legal.....	27
12.1. USA.....	27
12.2. UK.....	27
12.3. Germany.....	28
12.4. France.....	28
12.5. Netherlands.....	28
12.6. South Africa.....	28
12.7. Canada.....	28
12.8. General note.....	28
13. Problems, improvements etc.....	28
14. Release notes.....	29
15. Credits.....	29
16. Contact and further information.....	30
17. Our other assets.....	31

Thank you for buying our asset "Radio PRO"!

If you have any questions about this asset, send us an email at radio@crosstales.com. Please don't forget to rate it or write a little review – it would be very much appreciated.

1. Overview

Have you ever wanted to implement **radio stations** but don't want (or can't) pay an horrendous amount of money?

Whenever you like to provide good **sound** from **famous artists** for your games or apps, tune in on one of the uncountable **Internet MP3** and **OGG radio stations** available for **free**.

Thanks to this asset, it is now possible for all Unity developers to listen to high quality sound **without additional charges**.

Alternatively **receive music** streams from your **own server** (e.g. Icecast, VLC Server etc.).

We also implemented a unique **sound visualizer** in the demo scene, which you can modify to your liking.

Our asset "Radio" **receives** all Internet MP3 and OGG radio stations. This is important to know for legal purposes, on which we will inform you later on.

2. Features

2.1. Radio stations

- **Thousands of internet radio stations**: test the demo with your favourite channels
- **Receive music** from your **own server** (e.g. Icecast, VLC server etc.)
- **Performance**: very low impact on performance
- **No limits**: does survive changing scenes! The music is not interrupted even during load operations.
- **Good start**: contains more than **1'500** high-quality **radio stations**
- **MP3 & OGG**: Works with any MP3 and OGG settings (e.g. bit rate 32-500kbit/s)
- Read the **lyrics** of the **current track**
- Open **Spotify** with the **current track**
- **Information** about the **current** and **upcoming track** (title, artist)
- Details like downloaded **data**, total **play time** and **requests**
- **History** of all **played tracks** per station
- **Save** the **songs** of a station as **WAV** files
- Tune into **multiple stations** at the same time (and blend between stations)

2.2. Flexible & expandable

- Support for **loading** and **saving** of **user-managed lists**
- **Configurable** via **Shoutcast ID**, **PLS**, **M3U**, **XSPF** and **text files** (external / local)
- **Easy adaptation** and **extension opportunities** for existing radio stations
- Pre-configured radio station providers for **resources**, **files** and **URLs**. Deliver the radios your way or implement your **own provider** (e.g. for XML, JSON).
- **Reads** and **saves** M3U, PLS and XSPF files

2.3. Filter & order

Filter and **order** (ascending/descending) the **radio stations** by:

- Name
- URL
- Format
- Station URL
- Bitrate
- Genres
- Rating
- City
- Country
- Language

2.4. OnRadio integration

Access the full [OnRadio](#) API of the following services:

- Playlist (matching songs/stations for a query)
- Reco2 (matching or similar songs for a given artist)
- Topsongs (top songs of a genre)
- Station
- DARStation (details of the current station)
- SongArt (icon of the current song)

Query the songs/stations by:

- Artist
- Title
- Callsign (Name)
- Genre
- City
- Country
- Language

2.5. Documentation & control

- **Test** all radio **stations** inside the **editor**
- Powerful [API](#) for **maximum control**
- Detailed **demo scenes**
- Comprehensive [documentation](#) and **support**
- Full **source code** (including libraries)

2.6. Compatibility

- Supports **most build platforms** (except WebGL and WSA)
- Works with **Windows, Mac** and **Linux editors**
- Compatible with **Unity 2019.4 – 2023**
- Supports **AR** and **VR**
- **C# delegates** and **Unity events**
- Works with [Online Check](#)
- [PlayMaker](#) actions!

2.7. Integrations

- [Audio Visualizer](#)
- [Complete Sound Suite](#)
- [PlayMaker](#)
- [Visualizer Studio](#)
- [Apollo Visualizer Kit](#)
- [Rhythm Visualizator](#)
- [Volumetric Audio](#)

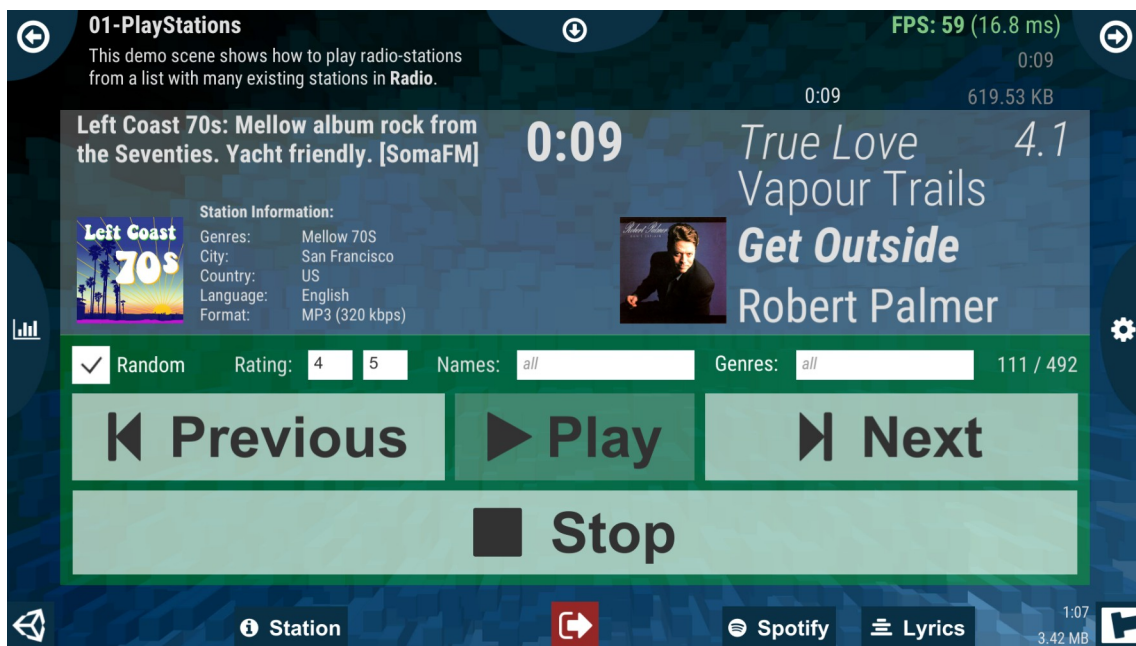
3. Demonstration

The asset comes with many demo scenes to show the main usage.

Note: The already existing ratings for the stations are based upon our personal taste, please feel free to adjust them to your liking.

3.1. PlayStations

Play radio stations with next and previous.



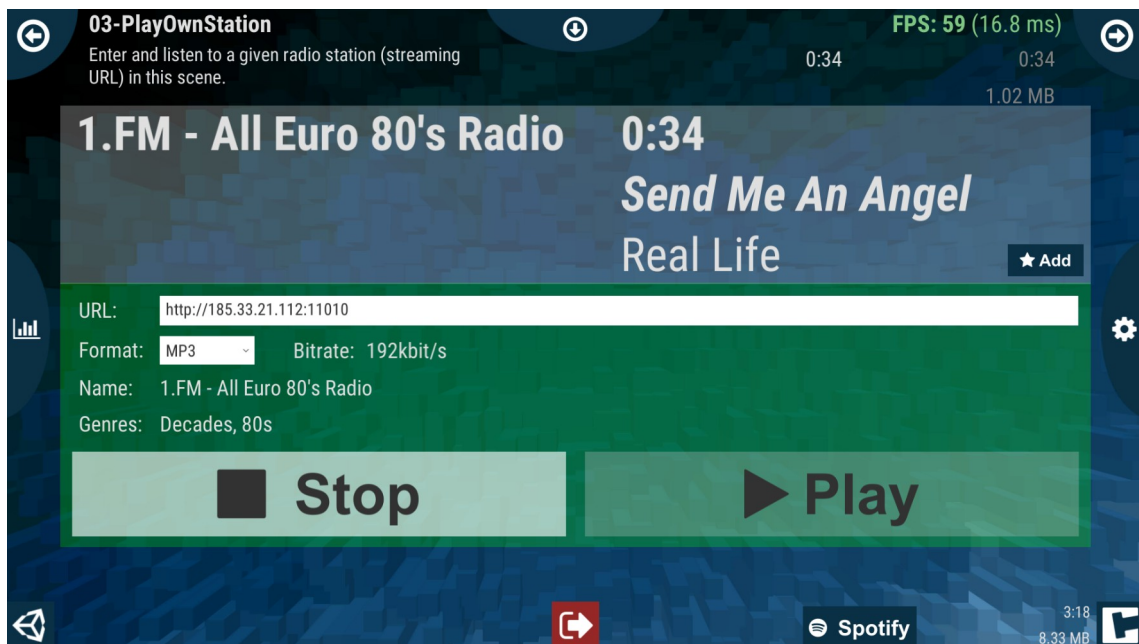
3.2. StationList

We included/linked over 500 existing radio-stations. Use it to order, rate and play stations.



3.3. PlayOwnStation

This demo scene shows how to play your own radio station.



3.4. 3DAudio

This scene demonstrates 3D positioned audio with 4 different RadioPlayers.

Needs the [Unity Standard Assets](#)-package.

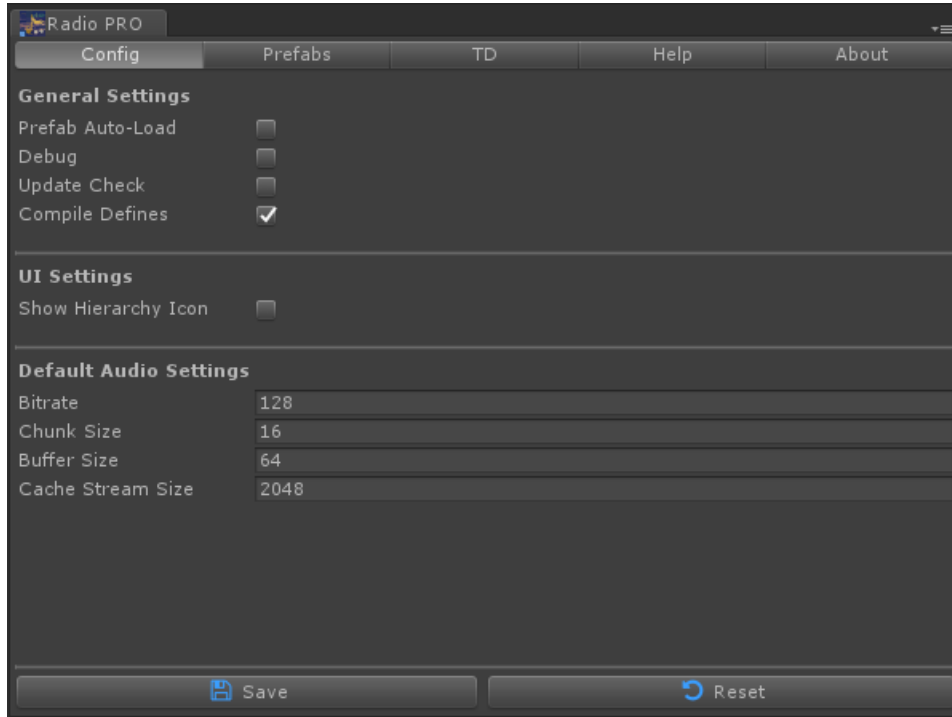
3.5. Loudspeaker

This scene demonstrates 3D positioned audio with one origin RadioPlayer and 4 Loudspeakers.

Needs the [Unity Standard Assets](#)-package.

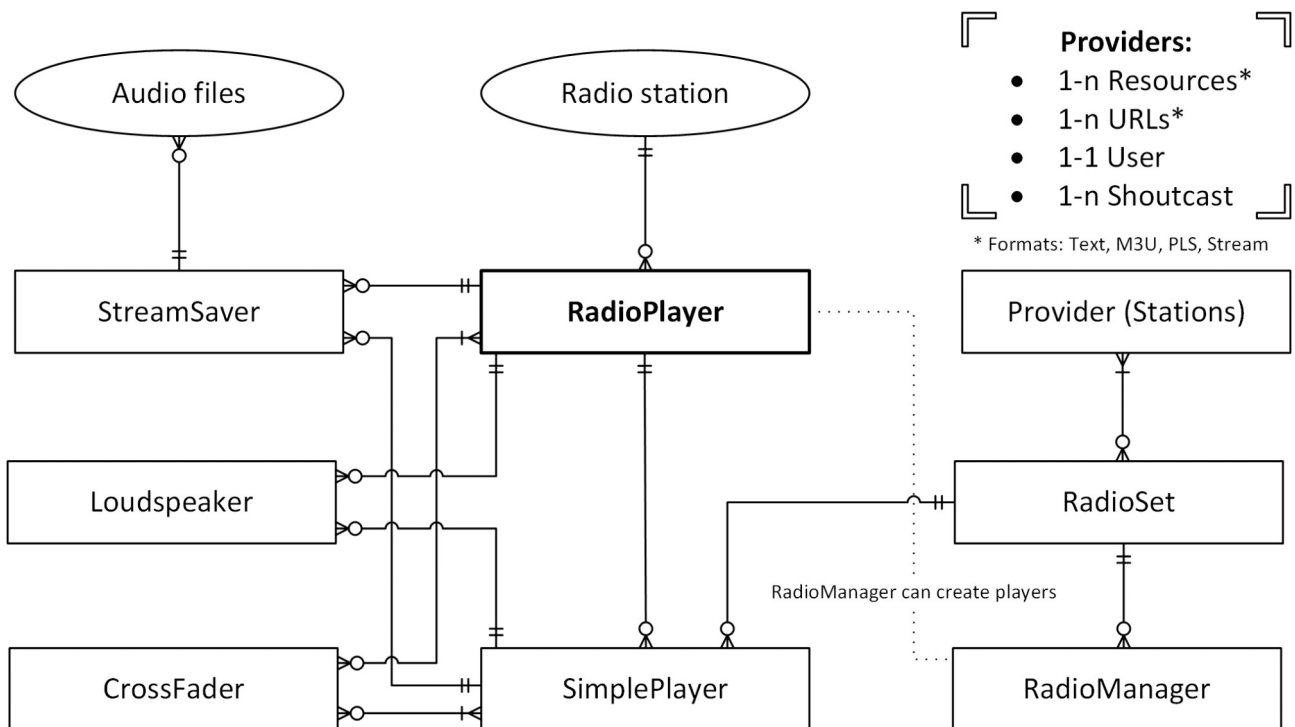
4. Setup

"Radio PRO" has global settings under "Edit\Preferences..." and under "Tools\Radio PRO\ Configuration...":



4.1. Schema

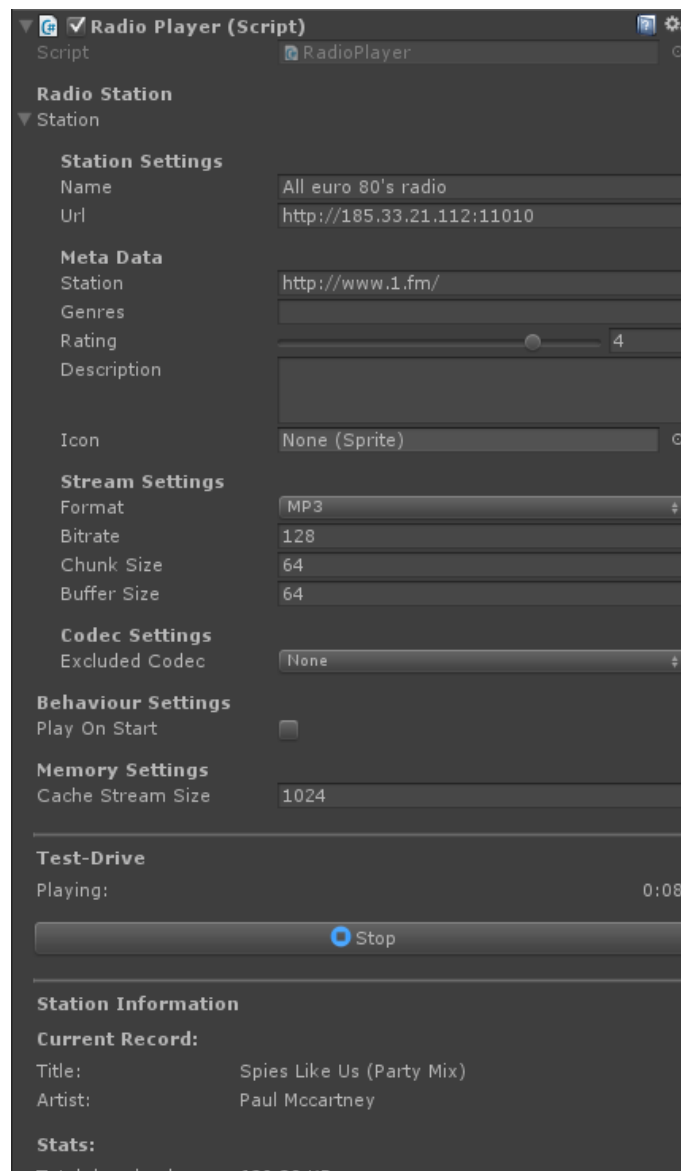
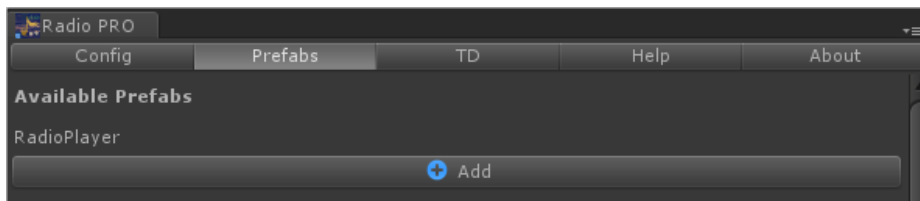
The following graphic explains the relationships between all relevant components:



4.2. Single radio-station (RadioPlayer)

There are four ways to use a single radio-station:

1. Add the prefab **RadioPlayer** from Assets/Plugins/crosstales/Radio/Resources/Prefabs to the scene
2. Or go to *Tools => Radio PRO=> Prefabs => RadioPlayer*
3. Right-click in the *hierarchy-window => Radio PRO => RadioPlayer*
4. Add it from the Prefabs-tab:



4.2.1. Parameter

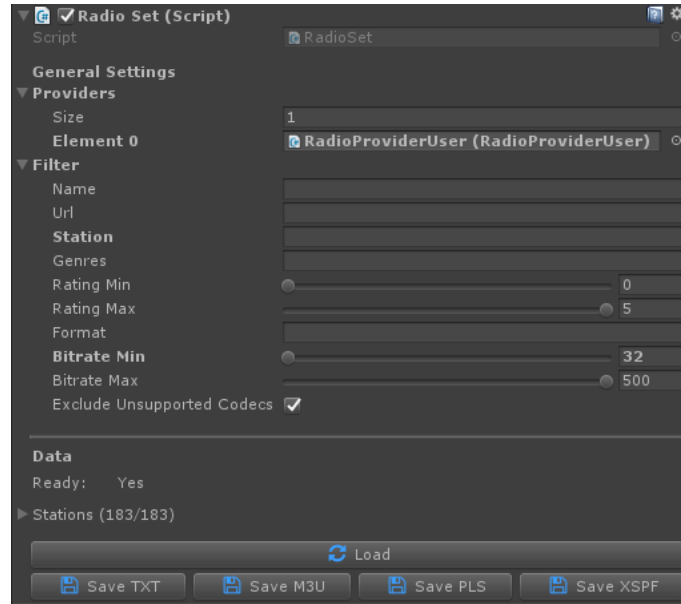
The following table will explain every parameter in detail:

Parameter	Description
Name	Name of the radio station.
Url	Streaming-URL of the station.
Station	Name of the station.
Genre	Genres of the radio.
Rating	The rating of the radio.
Description	Description for the radio station.
Icon	Icon for the radio station.
Format	Audio format of the stream (MP3 or OGG)
Bitrate	Bitrate in kbit/s.
Chunk Size	Size of the streaming-chunk in kilo-bytes.
Buffer Size	Size of the local buffer in kilo-bytes.
Play On Start	Play the radio on start on/off.
Cache Stream Size	Size of the cache stream in kilo-bytes KB (added to the buffer size).

4.3. Multiple radio-stations (RadioSet)

The "RadioSet" can be added in the same way as "RadioPlayer".

Standard-setup:



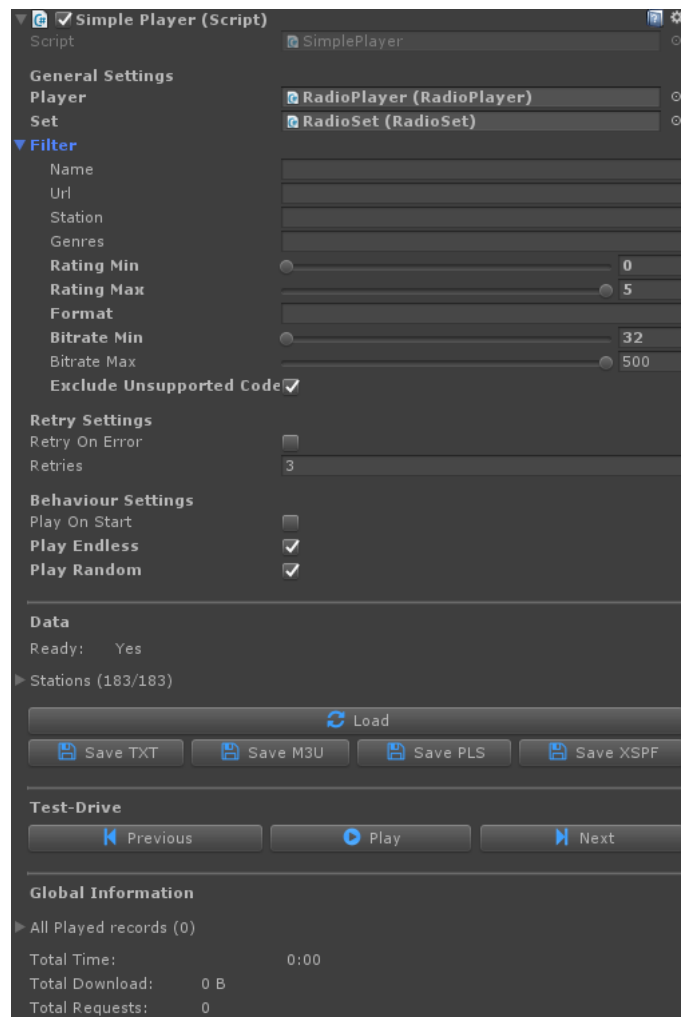
The important part is the "Providers". There you can add your own "Resource" containing all radios you want to provide with your game or application. You can easily implement your own **provider** (e.g. for XML, JSON) by sub-classing "RadioProvider.cs"!

You can modify the existing radio-stations as you like (edit, add or remove).

4.4. SimplePlayer

The "SimplePlayer" can be added in the same way as "RadioPlayer".

It connects a "RadioPlayer" and "RadioSet" and offers functions like "play next station" etc.



4.5. Other components

The other components can be added in the same way as "RadioPlayer"

4.5.1. StreamSaver

Allows to save songs from a station as WAV files.

4.5.2. SurviveSceneSwitch

Allows any Unity gameobject to survive a scene switch. This is especially useful to keep the music playing while loading a new scene.

4.5.3. Loudspeaker

This is useful to use the same player on multiple locations in the game.

4.5.4. CrossFader

Allows to fade between two players.

5. Providers

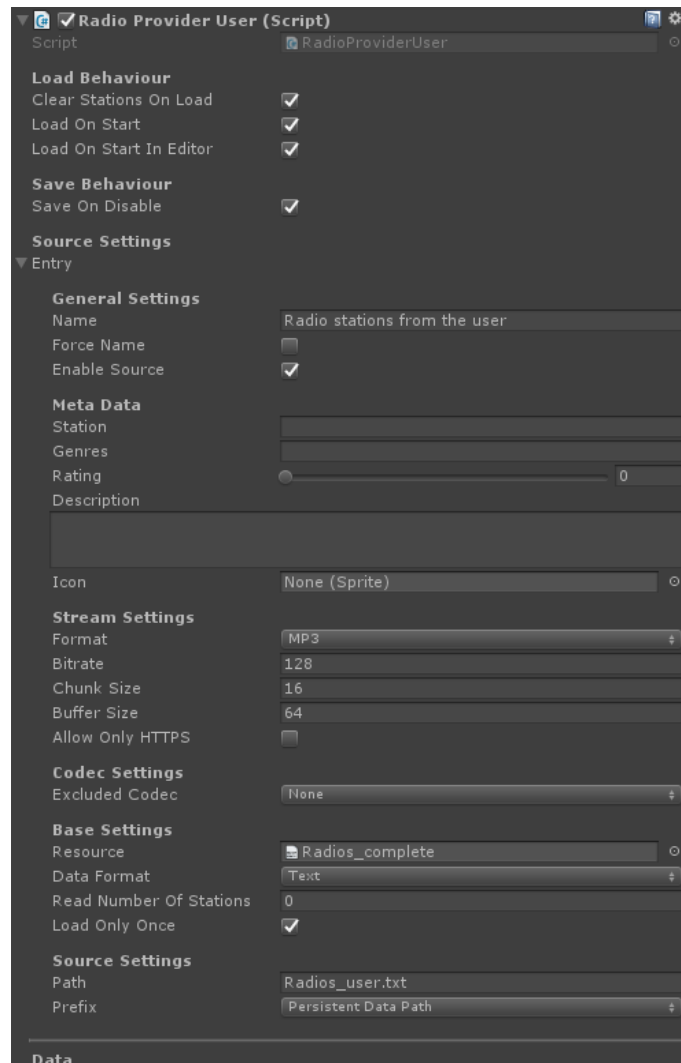
Providers are a collection of radio stations. The main benefit is the extensibility of this concept – we deliver different providers for:

1. Unity resources (RadioProviderResource)
2. Accessing files on the local machine (RadioProviderURL)
3. Accessing files on a web server (RadioProviderURL)
4. Accessing stations via Shoutcast-ID (RadioProviderShoutcast)
5. Load and save of user-based files (RadioProviderUser)

You can easily extend the base classes and build whatever you like (e.g. a provider to access data from XML or JSON).

The providers support text files, M3U, PLS and plain stream URLs.

Providers are used by the "RadioSet":



5.1. How-to create your own text resources

A resource must be a text file (typically with the extension "txt"). Every station is a new line inside this file and the format is like that ([] indicates **optional** parameters):

```
Name;Url;DataFormat;AudioFormat;[Station (optional);Genres (optional);Bitrate (in kbit/s, optional);Rating (0-5, optional);Description (optional);ExcludeCodec (optional);ChunkSize (in KB, optional);BufferSize (in KB, optional)]
```

A typical entry looks like that:

```
my radio;http://myradio.fm;stream;mp3;http://myradio.fm;hits;128;3  
#disabled;http://someradio.fm;stream;ogg;http://someradio.fm;pop;128;4
```

The file can contain *any number* of radio stations. The hash-sign (#) is used to comment lines.

6. API

The asset contains various classes and methods. The most important ones are explained here.

Make sure to **include** the **name space** in the relevant source files:

```
using Crosstales.Radio;
```

6.1. Players

These are important methods for the different players (**RadioPlayer** and **SimplePlayer**).

6.1.1. Play

```
void Play()
```

Plays the radio-station.

6.1.2. Stop

```
void stop()
```

Stops the playback of the radio-station.

6.1.3. Restart

```
void Restart()
```

Restarts the playback of the radio-station.

6.2. RadioManager

6.2.1. Next

```
RadioPlayer Next(bool random = false, bool stopAll = true, bool playImmediately = true)
```

Next (normal/random) radio from this manager.

6.2.2. Previous

```
RadioPlayer Previous(bool random = false, bool stopAll = true, bool  
playImmediately = true)
```

Previous (normal/random) radio from this manager.

6.2.3. NextStation

```
RadioStation NextStation(bool random = false)
```

Previous (normal/random) radio station from this manager.

6.2.4. PreviousStation

```
RadioStation PreviousStation(bool random = false)
```

Previous (normal/random) radio station from this manager.

6.2.5. StopAll

```
void stopAll()
```

Stops the playback of all radio-stations.

6.3. RadioProvider

6.3.1. Next

```
RadioStation Next(bool random = false)
```

Previous (normal/random) radio station from this manager.

6.3.2. Previous

```
RadioStation Previous(bool random = false)
```

Previous (normal/random) radio station from this manager.

6.4. Callbacks

There are various callbacks available. Subscribe them in the "Start"-method and unsubscribe in "OnDestroy".

6.4.1. Playback start and end

```
PlaybackStart(RadioStation station);
```

```
PlaybackStart OnPlaybackStart;
```

Triggered whenever the playback starts.

```
PlaybackEnd(RadioStation station);
```

```
PlaybackEnd OnPlaybackEnd;
```

Triggered whenever the playback ends.

6.4.2. Buffering start, end and progress

```
BufferingStart(RadioStation station);
```

```
BufferingStart OnBufferingStart;
```

Triggered whenever the buffering starts.

```
BufferingEnd(RadioStation station);
```

```
BufferingEnd OnBufferingEnd;
```

Triggered whenever the buffering ends.

```
BufferingProgressUpdate(RadioStation station, float progress);
```

```
BufferingProgressUpdate OnBufferingProgressUpdate;
```

Triggered whenever the buffering progress changes.

6.4.3. Audio start, end and time

```
AudioStart(RadioStation station);
```

```
AudioStart OnAudioStart;
```

Triggered whenever the audio starts.

```
AudioEnd(RadioStation station);
```

```
AudioEnd OnAudioEnd;
```

Triggered whenever the audio ends.

```
AudioPlayTimeUpdate(RadioStation station, float playtime);
```

```
AudioPlayTimeUpdate OnAudioPlayTimeUpdate;
```

Triggered whenever the audio playtime changes.

6.4.4. Record change and time

```
RecordChange(RadioStation station, RecordInfo newRecord)
```

```
RecordChange OnRecordChange;
```

Triggered whenever an audio record changes.

```
RecordPlayTimeUpdate(RadioStation station, RecordInfo record, float playtime);
```

```
RecordPlayTimeUpdate OnRecordPlayTimeUpdate;
```

Triggered whenever the audio record playtime changes.

6.4.5. Next record change and delay

```
NextRecordChange(RadioStation station, RecordInfo nextRecord)
```

```
NextRecordChange OnNextRecordChange;
```

Triggered whenever the next record information is available.

```
NextRecordDelayUpdate(RadioStation station, RecordInfo nextRecord, float delay)
```

```
NextRecordDelayUpdate OnNextRecordDelayUpdate;
```

Triggered whenever the next record delay time changes.

6.4.6. Errors

```
ErrorInfo(Model.RadioStation station, string info);
```

```
ErrorInfo OnErrorInfo;
```

Triggered whenever an error occurs.

6.4.7. Provider ready

Callback for **RadioSet**, **RadioManager** and **SimplePlayer**.

```
ProviderReady();
```

```
ProviderReady OnProviderReady;
```

Triggered whenever all providers are ready.

6.4.8. Station change

Callback for **SimplePlayer**.

```
StationChange(RadioStation newStation);
```

```
StationChange OnStationChange;
```

Triggered whenever an radio station changes.

6.4.9. Example

```
void OnEnable() {
    // Subscribe event listeners
    Radio.OnPlaybackStart += playBackStart;
    Radio.OnPlaybackEnd += playBackEnd;
    Radio.OnAudioPlayTimeUpdate += audioPlayTime;
    Radio.OnBufferingProgressUpdate += bufferingProgress;
}

void OnDisable() {
    // Unsubscribe event listeners
    Radio.OnPlaybackStart -= playBackStart;
    Radio.OnPlaybackEnd -= playBackEnd;
    Radio.OnAudioPlayTimeUpdate -= audioPlayTime;
    Radio.OnBufferingProgressUpdate -= bufferingProgress;
}

private void playBackStart(RadioStation station) {
    Debug.Log("Playback started");
}

private void playBackEnd(RadioStation station) {
    Debug.Log("Playback ended");
}

private void audioPlayTime(RadioStation station, float playtime) {
    Debug.Log("Playtime: " + Helper.FormatSecondsToHourMinSec(playtime));
}

private void bufferingProgress(RadioStation station, float progress) {
    Debug.Log("Buffer: " + progress.ToString("0%"));
}
```

6.5. Complete API

Please read the [Radio-api.pdf](#) for more details.

7. Third-party support (PlayMaker etc.)

"Radio PRO" supports various assets from other publishers. Please import the desired packages from the "Assets/Plugins/crosstales/Radio /3rd party"-folder.

8. Verify installation

Check if Radio is installed:

```
#if CT_RADIO
    Debug.Log("Radio installed: " + Util.Constants.ASSET_VERSION);
#else
    Debug.LogWarning("Radio NOT installed!");
#endif
```

9. Upgrade to new version

Follow this steps to upgrade the version of "Radio PRO":

1. Update "Radio PRO" to the latest version from the "Unity AssetStore"
2. Inside the project in Unity, go to menu "File" => "New Scene"
3. Delete the "Assets/Plugins/crosstales/Radio" folder from the Project-view
4. Import the latest version downloaded from the "Unity AssetStore"

10. Important notes

10.1. Android

Many radio stations still only support plain http://-URLs. Therefore, if HTTP is needed, enable cleartext traffic by using an AndroidManifest.xml file that includes an **android:usesCleartextTraffic="true"** application attribute.

If that's not an option, enable "Allow Only HTTPS" on the station providers.

11. OnRadio

OnRad.io has a massive radio station search engine that plugs into Radio PRO providing a wide array of audio to enrich any game play.

There's an infinite way to play music by artist, song title, genres or stations that provide players great audio satisfaction.

OnRad.io is an annual service that maintains a list of 100'000 radio stations that are constantly changing.

As an OnRad.io customer you can rely on our meticulously maintained station list and a unique search engine that allows for searching by song or artist as the songs are played on radio.

All customers receive a unique partner token then will allow for queries that can look up song or station data and get streaming URLs so those stations can be played inside any Unity app using Radio PRO.

There are 3 price tiers for the OnRad.io for Radio PRO based on the number of DAILY API queries desired. To enable OnRad.io service follow these steps:

11.1. Step 1

Purchase the desired OnRad.io tier from <https://dar.fm/upgrade.php#radiopro>

Pro Bronze: \$99.95/year - 10'000 daily API queries (special Radio PRO price \$79.95)

Pro Plus: \$139.95/year - 50'000 daily API queries

Pro Platinum: \$299.95/year - 250'000 daily API queries

11.2. Step 2

Email sales@dar.fm with your receipt number requesting a partner token for Radio PRO.

11.3. Step 3

Insert the partner token received in Step 2 into Radio PRO to enable full access to OnRad.io.

12. Legal

Since music copyright is a big issue, we decided to look up on eventual legal ramifications our "Radio" could cause.

According to the GT2b ([Gemeinsamer Tarif 2b 2014-2021](#): *Entschädigung für das Weitersenden von Radio- und Fernsehprogrammen und der darin enthaltenen Werke und Leistungen über IP-basierte Netze auf mobile Endgeräte oder auf PC-Bildschirme*) which was approved by the "Eidgenössische Schiedskommission für die Verwendung von Urheberrechten und verwandten Schutzrechten" in 2013, it is perfectly legal to **receive** Internet radio stations, as long as you **don't send** anything. Since our "Radio" asset functions only as a **receiver**, it is legal to use it (at least here in Switzerland).

The various internet Radio stations pay copyright bills, they are the ones sending the music. Again, since "Radio" does **NOT SEND** any music but just **RECEIVES** it (including commercials of the various stations), there are no legal ramifications. It's just like listening to Radio stations the "old school" way.

If you are unsure about the legal practices in your country, check up with the music licensing body of your country.

Again: the legal status discussed above is from **Switzerland**, we don't know every country's laws regarding this issue.

The following links to music licensing bodies may be of help:

12.1. USA

www.loudcity.net

www.swcast.com

www.bmi.com

www.ascap.com

www.soundexchange.com

12.2. UK

www.ppluk.com

www.mcps-prs-alliance.co.uk

12.3. Germany

www.gema.de

www.gvl.de

12.4. France

www.sacem.fr

12.5. Netherlands

www.bumastemra.nl

12.6. South Africa

www.samro.org.za

12.7. Canada

www.cmrra.ca

12.8. General note

We included the various stations in the demo scenes mainly to show that it works technically and also for fun (we like music and it was interesting to find all these different radio stations).

If you plan on including stations in a game that you release on a commercial basis, we **strongly recommend** that you **contact** the **stations** you want to use yourself.

13. Problems, improvements etc.

If you encounter any problems with this asset, just [send us an email](#) with a problem description and the invoice number and we will try to solve it.

We will try and make a version for all platforms as well, please bear with us.

14. Release notes

See "VERSIONS.txt" under "Assets/Plugins/crosstales/Radio/Documentation" or online:
<https://crosstales.com/media/data/assets/radio/VERSIONS.txt>

15. Credits

"Radio PRO" uses modified versions of:

- [NAudio 1.7.2](#)
- [NLayer 1.1.0](#)
- [NVorbis 0.8.4](#)
- UDE

The included radio-stations are mainly provided by:

- [1.fm](#)
- [SomaFM.com](#)
- [Energy](#)

The search is provided by [Spotify](#).

The lyrics are provided by [AZLyrics](#).

The querying and art APIs are provided by [OnRadio](#).

We aren't affiliated to those companies.

The icons are based on [Font Awesome](#).

16. Contact and further information

crosstales LLC

Schanzeneggstrasse 1

CH-8002 Zürich

Homepage: <https://www.crosstales.com/en/portfolio/radio/>

Email: radio@crosstales.com

AssetStore: <https://assetstore.unity.com/lists/crosstales-42213>

Forum: <https://forum.unity3d.com/threads/radio-mp3-and-ogg-streaming-solution.334604/>

Documentation: <https://www.crosstales.com/media/data/assets/radio/Radio-doc.pdf>

API: <https://www.crosstales.com/en/assets/radio/api>









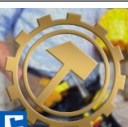
Windows-Demo: <https://drive.google.com/file/d/1uvfqDe2dWVGiVjaBqP2mwUvXgZUFIPQP/view?usp=sharing>


Mac-Demo: <https://drive.google.com/file/d/1v0-KA2Xik0cat35destgOn8qwoe3xyKS/view?usp=sharing>

Linux-Demo: <https://drive.google.com/file/d/1v5VMKx1VrobPtxzRmWuL14yGryVreyi/view?usp=sharing>

Android-Demo: https://drive.google.com/file/d/1vA4cTErI7N33djUpw9zeHBso4wR_0qvX/view?usp=sharing

17. Our other assets

 <p>3D Skybox</p>	<p>Those beautiful packages contain professional 8k, HDR, stereoscopic 360° real-world skyboxes for your projects.</p>
 <p>Bad Word Filter</p>	<p>The "Bad Word Filter" (aka profanity or obscenity filter) is exactly what the title suggests: a tool to filter swearwords and other "bad sentences".</p>
 <p>DJ</p>	<p>DJ is a player for external music-files. It allows a user to play his own sound inside any Unity-app. It can also read ID3-tags.</p>
 <p>File Browser</p>	<p>File Browser is a wrapper for native file dialogs on Windows, macOS, Linux and UWP (WSA).</p>
 <p>Online Check</p>	<p>You need a reliable solution to check for Internet availability? Here it is!</p>
 <p>RT-Voice</p>	<p>RT-Voice uses the computer's (already implemented) TTS (text-to-speech) voices to turn the written lines into speech and dialogue at run-time! Therefore, all text in your game/app can be spoken out loud to the player.</p>
 <p>True Random</p>	<p>True Random can generate "true random" numbers for you and your application. The randomness comes from atmospheric noise, which for many purposes is better than the pseudo-random number algorithms typically used in computer programs.</p>
 <p>Turbo Backup</p>	<p>Turbo Backup is the fastest and safest way to backup your Unity project. It only stores the difference between the last backup, this makes it incredible fast.</p>
 <p>Turbo Builder</p>	<p>Turbo Builder creates builds for multiple platforms in one click. It works together with Turbo Switch to offer an incredible fast build pipeline.</p>

 <p>Turbo Switch</p>	<p>Turbo Switch is a Unity editor extension to reduce the time for assets to import during platform switches. We measured speed improvements up to 100x faster than the built-in switch in Unity.</p>
--	---