

Radio PRO

Music matters



Documentation

crosstales LLC

Date: 19. October 2017

Version: 2.8.5

Table of Contents

1. Overview.....	4
2. Features.....	5
2.1. Radio stations.....	5
2.2. Flexible & expandable.....	5
2.3. Documentation & control.....	5
2.4. Compatibility.....	5
2.5. Supported third-party assets.....	6
3. Demonstration.....	7
3.1. ComplexUI.....	7
3.2. PlayStations.....	8
3.3. 3DAudio.....	8
3.4. Loudspeaker.....	8
4. Setup.....	9
4.1. Single radio-station (RadioPlayer).....	9
4.1.1. Parameter.....	10
4.2. Multiple radio-stations (RadioManager).....	11
4.3. SimplePlayer.....	12
4.4. Other components.....	12
4.4.1. InternetCheck.....	12
4.4.2. SurviveSceneSwitch.....	12
4.4.3. Proxy.....	12
5. Providers.....	13
5.1. How-to create your own text resources.....	14
6. API.....	15
6.1. RadioPlayer.....	15
6.1.1. Play.....	15
6.1.2. Stop.....	15
6.1.3. Restart.....	15
6.2. RadioManager.....	15
6.2.1. Next.....	15
6.2.2. Previous.....	16
6.2.3. NextStation.....	16
6.2.4. PreviousStation.....	16
6.2.5. StopAll.....	16
6.3. RadioProvider.....	16
6.3.1. Next.....	16
6.3.2. Previous.....	16
6.4. SimplePlayer.....	17
6.5. Callbacks.....	17
6.5.1. Playback start and end.....	17
6.5.2. Buffering start, end and progress.....	17
6.5.3. Audio start, end and time.....	17
6.5.4. Record change and time.....	18
6.5.5. Next record change and delay.....	18
6.5.6. Errors.....	18
6.5.7. Provider ready.....	19
6.5.8. Station change.....	19
6.5.9. Example.....	20
6.6. Complete API.....	20
7. Third-party support (PlayMaker etc.).....	21
8. Upgrade to new version.....	21

9. Use the source files.....	21
10. Legal.....	22
10.1. USA.....	22
10.2. UK.....	22
10.3. Germany.....	23
10.4. France.....	23
10.5. Netherlands.....	23
10.6. South Africa.....	23
10.7. Canada.....	23
10.8. General note.....	23
11. Problems, improvements etc.....	23
12. Release notes.....	24
13. Credits.....	24
14. Contact and further information.....	25
15. Our other assets.....	26

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 a load operation if necessary!
- **Good start:** Contains more than 200 high-quality radio stations!
- **MP3 & OGG:** Works with any MP3 and OGG settings (e.g. bit rate 32-500kbit/s)
- 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** 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)!

2.3. Documentation & control

- **Test** radio stations inside the editor!
- Powerful [API](#) for **maximum control**!
- **Proxy manager** for **Internet connections**!
- **Internet availability tester** included!
- Detailed **demo scenes**!
- Comprehensive [documentation](#) and **support**!
- Full **C# source code**

2.4. Compatibility

- Supports **all build platforms** (except Web & WSA)
- Works with **Windows**, **Mac** and **Linux editors**!
- Compatible with **Unity 5.1 – Unity 2017**
- Supports **AR** and **VR**!
- [PlayMaker](#) actions!

2.5. Supported third-party assets

- [Audio Visualizer](#)
- [Complete Sound Suite](#)
- [PlayMaker](#)
- [Visualizer Studio](#)

3. Demonstration

The asset comes with many demo scenes to show the main usage. All demo scenes also contain the **unique visualization tool** we made just for this asset.

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

3.1. ComplexUI

We included/linked over 200 existing radio-stations. Here you can order and play your stations.

01-ComplexUI FPS: 45 (22.3 ms)

This scene shows a list with many existing radio-stations. Here you can order manage and play your stations. Changes are saved to a local editable user file.

Name	Station	Bitrate	Genre	Rating		Stations: 207
all	all	32	500	all	0 5	
1 Hits 50'S	hits50s.com	128	50s	3	<input type="range"/>	stopped ▶ Play
1 Hits 60'S	hits70s.com	128	60s	3	<input type="range"/>	stopped ▶ Play
1 Hits 70'S	hits70s.com	128	70s	4	<input type="range"/>	Sun Of Jamaica - The Goombay Dance Band 0:05 ■ Stop
1 Hits 80'S	hits80s.com	128	80s	4	<input type="range"/>	stopped ▶ Play
1 Hits 90'S	hits90s.com	128	90s	4	<input type="range"/>	stopped ▶ Play
100 Djay	100radios.net	128	electro, hits, dubstep, dance	4	<input type="range"/>	stopped ▶ Play
100 Hit Radio	100hitradio.net	128	charts	4	<input type="range"/>	stopped ▶ Play
100 Urban	100radios.net	128	hits, highpop, r'n'b, rap, urban	4	<input type="range"/>	stopped ▶ Play
1000 Classical Hits	1000classicalhits.playtheradio.com	128	baroque, chamber, classic, opera, symphony	4	<input type="range"/>	stopped ▶ Play
1000 Hits Classical Music	1000hitsclassical.com	128	baroque, chamber, classic, symphony	3	<input type="range"/>	stopped ▶ Play
101 Smooth Jazz	101smoothjazz.com	128	jazz, easy listening	4	<input type="range"/>	stopped ▶ Play

3.2. PlayStations

This demo scene shows how to play a random radio-station.



3.3. 3DAudio

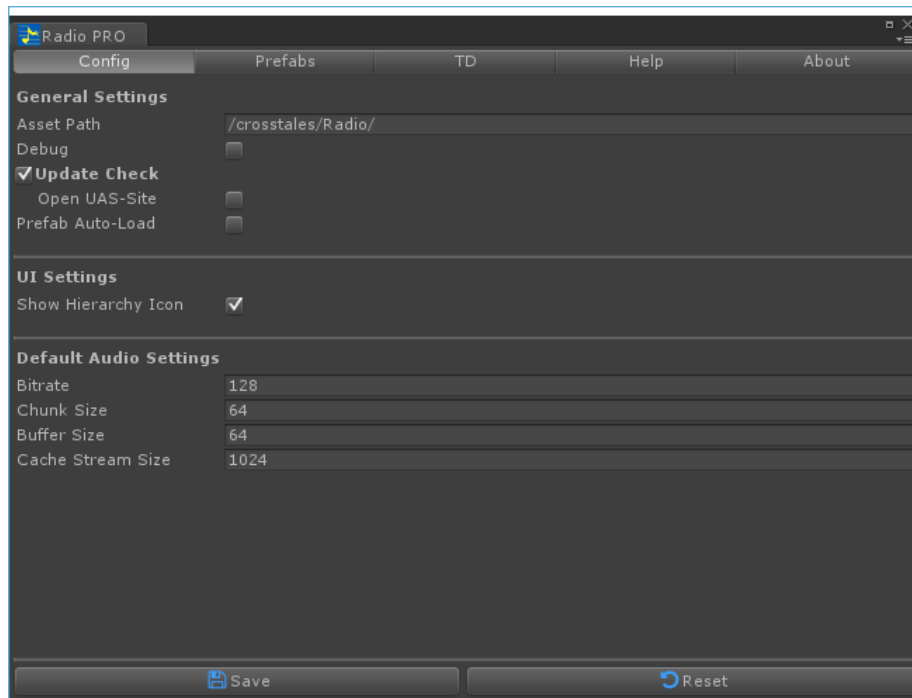
This scene demonstrates 3D positioned audio with 4 different RadioPlayers.

3.4. Loudspeaker

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

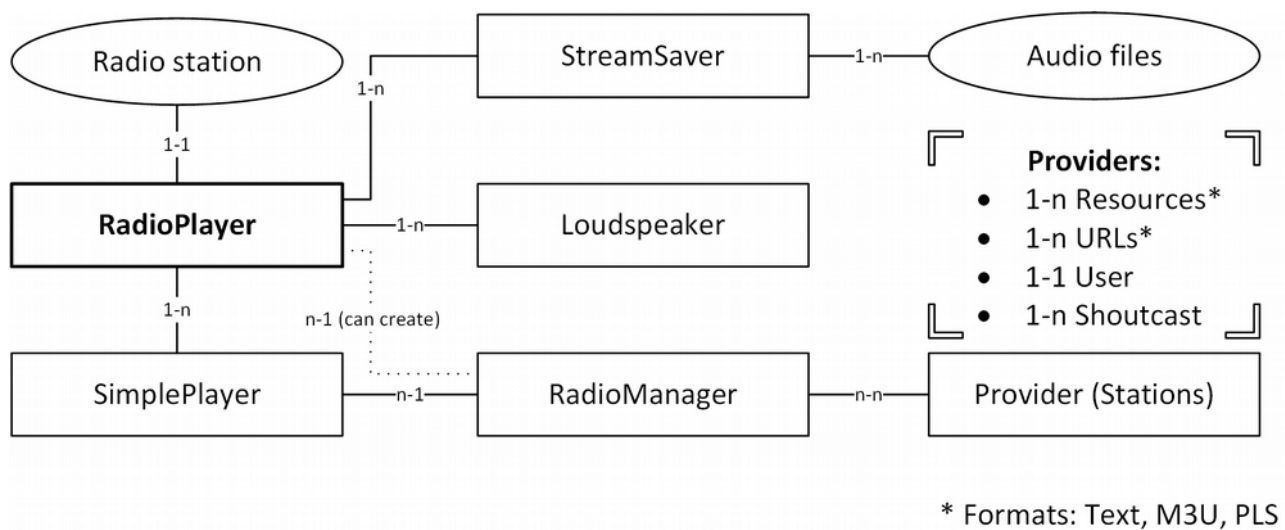
4. Setup

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



4.1. Schema

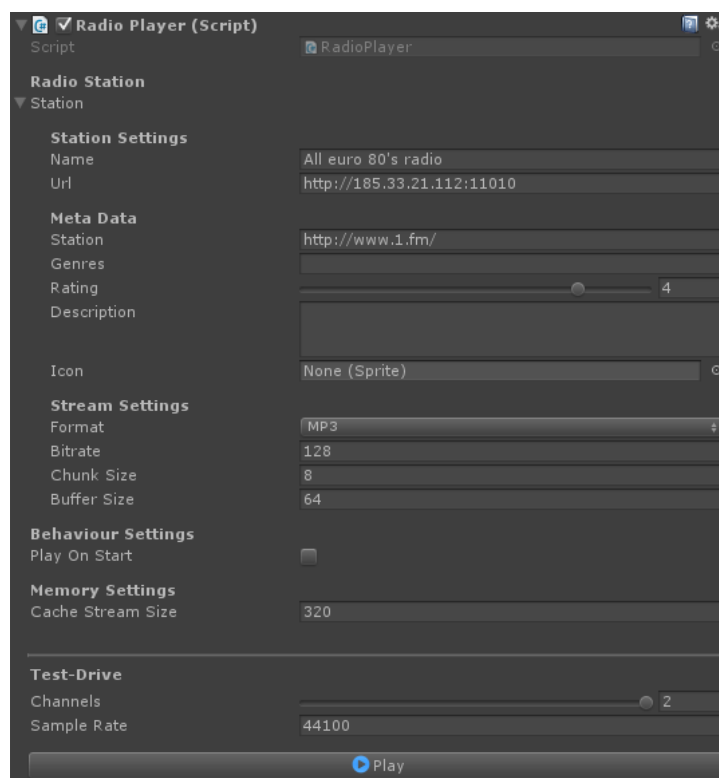
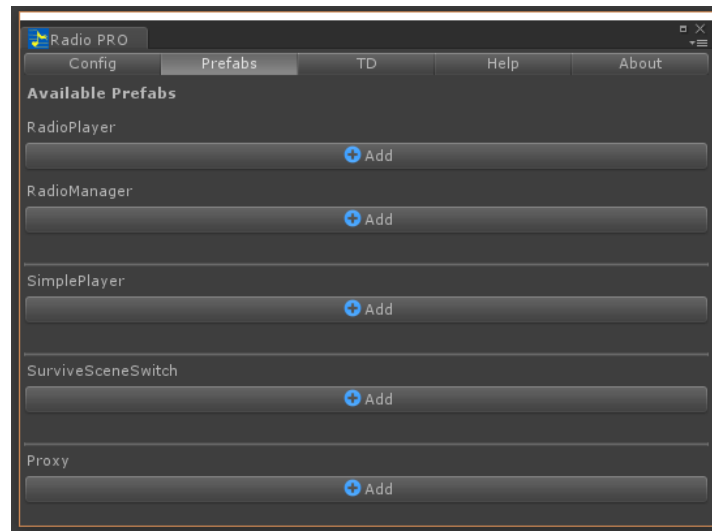
The following graphic explains the relationships between all relevant components:



4.2. Single radio-station (RadioPlayer)

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

1. Add the prefab **RadioPlayer** from Assets/crosstales/Radio/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

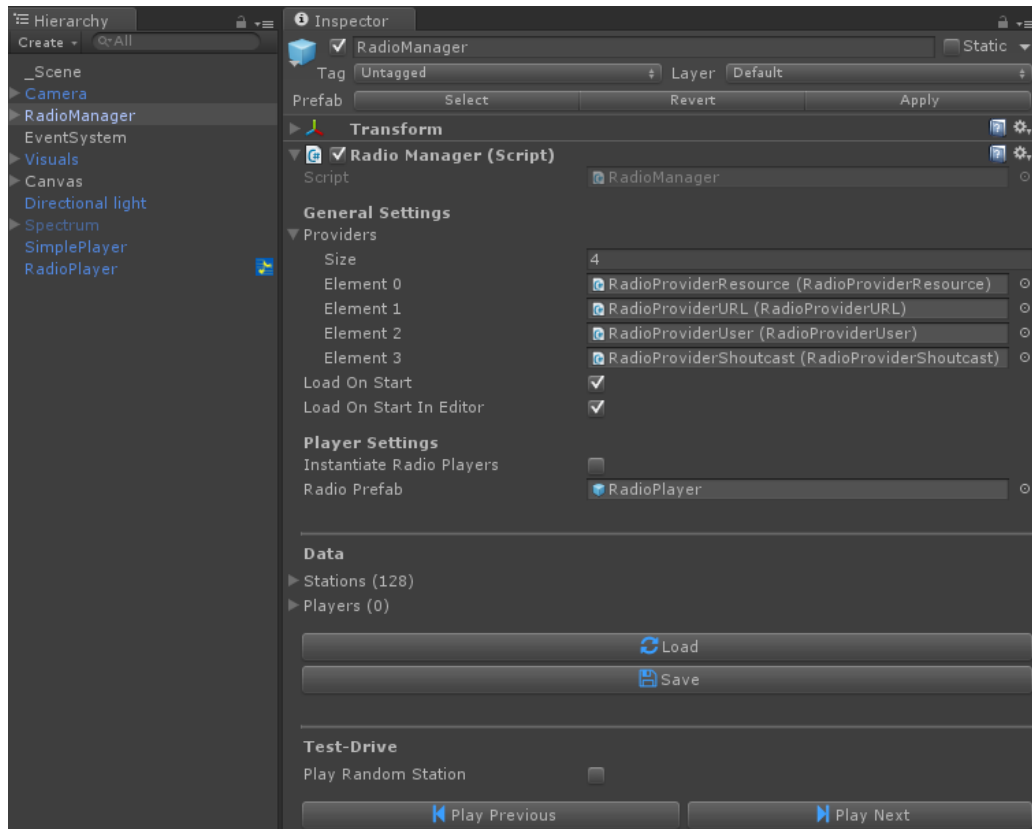
The following table will explain every parameter in detail:

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	Your 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 (RadioManager)

The “RadioManager” can be added in the same way as “RadioPlayer”.

Standard-setup:



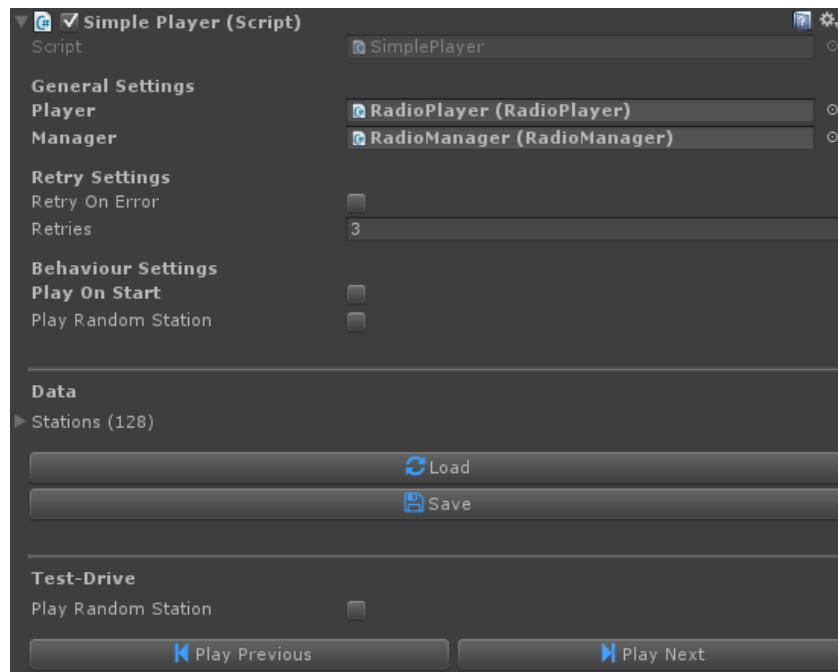
The important part is the “Providers”. There you can add your own “Resource” containing all radios you want to provide with your game or application. You can easily implement your own **provider** (e.g. for XML, JSON) by sub-classing “RadioProvider.cs”!

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 „RadioManager“ 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. InternetCheck

Checks the Internet availability.

4.5.4. Proxy

Handles HTTP/HTTPS Internet connections via proxy server.

5. Providers

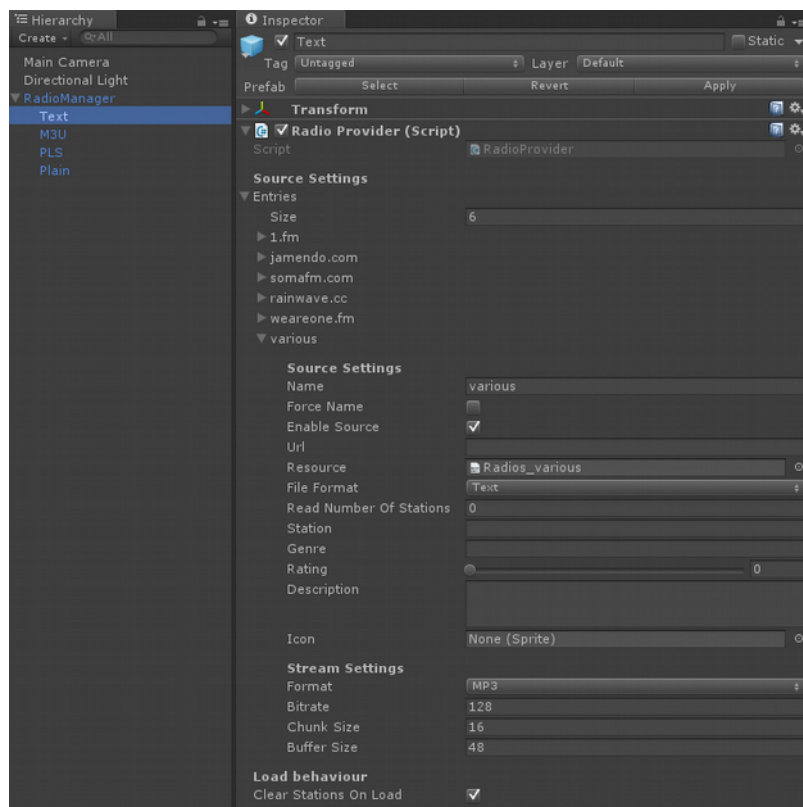
Providers are a collection of radio stations. The main benefit is the extensibility of this concept – we deliver different providers for:

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 “RadioManager”:



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 methods and the most important are explained here.
Make sure to **include** the **name space** in your relevant source files:

```
using Crosstales.Radio;
```

6.1. RadioPlayer

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. SimplePlayer

This class combines a given **RadioPlayer** and a **RadioManager** to a single simple player with all the necessary functions defined in both classes.

6.5. Callbacks

There are various callbacks available. Subscribe them in the "Start"-method and unsubscribe in "OnDestroy" (**RadioPlayer** and **SimplePlayer**).

6.5.1. Playback start and end

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

```
PlaybackStart OnPlaybackStart;
```

Triggered whenever the playback starts.

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

```
PlaybackEnd OnPlaybackEnd;
```

Triggered whenever the playback ends.

6.5.2. 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.

6.5.3. 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.

6.5.4. 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.

6.5.5. 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.

6.5.6. Errors

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

```
ErrorInfo OnErrorInfo;
```

Triggered whenever an error occurs.

6.5.7. Provider ready

Callback for **RadioManager** and **SimplePlayer**.

```
ProviderReady();
```

```
ProviderReady OnProviderReady;
```

Triggered whenever all providers are ready.

6.5.8. Station change

Callback for **SimplePlayer**.

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

```
StationChange OnStationChange;
```

Triggered whenever an radio station changes.

6.5.9. Example

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

void OnDestroy() {
    // 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%"));
}
```

6.6. 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 „3rd party“-folder.

8. Upgrade to new version

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

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

9. Use the source files

1. Create an empty scene inside your project
2. Delete the "Assets/crosstailes/Radio/Plugins/Radio.dll"
3. Import the "Assets/crosstailes/Radio/Sources.unitypackage"

10. 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 2014-2017](#): *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. Plus you can't save or repeat any of the music, 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:

10.1. USA

www.loudcity.net

www.swcast.com

www.bmi.com

www.ascap.com

www.soundexchange.com

10.2. UK

www.ppluk.com

www.mcps-prs-alliance.co.uk

10.3. Germany

www.gema.de

www.gvl.de

10.4. France

www.sacem.fr

10.5. Netherlands

www.bumastemra.nl

10.6. South Africa

www.samro.org.za

10.7. Canada

www.cmrra.ca

10.8. 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.

11.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.

12. Release notes

See "VERSIONS.txt" under "Assets/crosstales/Radio/Documentation".

13. Credits

"Radio PRO" uses modified versions of:

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

The included radio-stations are mainly provided by:

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

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

We aren't affiliated to this companies.

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

14. Contact and further information

crosstales LLC

Weberstrasse 21

CH-8004 Zürich

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

Email: radio@crosstales.com

AssetStore: <https://goo.gl/qwtXyb>

Forum: <http://goo.gl/HxgngH>

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

API: <http://goo.gl/G0hu6n>








Windows-Demo: https://www.crosstales.com/media/data/assets/radio/downloads/Radio_demo.zip

Mac-Demo: https://www.crosstales.com/media/data/assets/radio/downloads/Radio_demo_mac.zip

Linux-Demo: https://www.crosstales.com/media/data/assets/radio/downloads/Radio_demo_linux.zip

Android-Demo: <https://www.crosstales.com/media/radio/Radio.apk>

15. Our other assets

 <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>Online Check</p>	<p>You need a reliable solution to check for Internet availability? Here it is!</p>
 <p>RSockpol</p>	<p>Reliable Socket Policy Server which acts as replacement for Unitys own „sockpol.exe“.</p>
 <p>RTVoice</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>TPS</p>	<p>Turbo Platform 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>
 <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>