Edge FMV Reference Guide

by XGASOFT

1. Introduction 3
2. Buy Now 3
3. Download PDF 3
4. Changelog 3
5. Reference Guide 4
5.1. edgefmv_load 6
5.2. edgefmv_play 7
5.3. edgefmv_unload 8
5.4. edgefmv_pause 8
5.5. edgefmv_skip 8
5.6. edgefmv_seek 8
5.7. edgefmv_set_volume 9
5.8. edgefmv_get_time 9
5.9. edgefmv_draw_progress 9
6. MakeFMV 10
7. End-User License Agreement ("EULA") 13

1. Welcome to Edge FMV

Thank you for choosing Edge FMV, the first native GML video solution for GameMaker Studio. **Edge Engine** is a fully cross-platform, modular framework built to augment Game Maker Studio with pre-made code and assets that serve as the foundation for a wide variety of game genres. All Edge Engine modules feature creative, human-readable code with helpful notations throughout, making them both powerful and easy to use.

Edge FMV is an *experimental* solution for achieving video playback in GameMaker Studio without any external libraries or extensions. This makes it very small, affordable, and multiplatform, and best of all, its custom container format does not require any special licensing or royalty to use in commercial projects.

In this guide you will learn how to convert standard video formats to FMV and how to load and play them from two unique storage methods, as well as learn the advantages and disadvantages of each. You will also learn to manipulate FMVs using the available playback controls and even how to display basic UI elements such as a progress bar.

2. Buy Now(https://marketplace.yoyogames.com/assets/2947/gml-video-player-edge-engine)

3. Download PDF(https://docs.xgasoft.com/wp-content/uploads/sites/2/edge-fmv-reference-guide-8.pdf)

4. What's New

v1.1.4

• Updated MakeFMV to v1.3.0.5, adding advanced options to set video framerate, dimensions, and image quality (see documentation)

v1.1.3

• Fixed a crash when using the `all` keyword with `edgefmv_unload`

• Updated documentation to a new format

v1.1.2

• Updated MakeFMV to v1.3.0 with minor improvements to the GUI, as well as functional improvements including even better support for nonstandard paths, fix for command window not showing up on conversion, and fix for some antivirus software generating a false positive when running MakeFMV.

• Added `edgefmv_get_time` script for reading the playback position of FMVs

v1.1.0

• Updated MakeFMV to v1.2.0 with GUI frontend, custom output paths, and better support for nonstandard paths (e.g. not on C drive, paths with spaces, etc.)

v1.0.2

• Fixed looped videos drawing a blank frame between replays

v1.0.1

• Added ability to loop FMVs

v1.0

• Initial release

5. Overview

Warning: Edge FMV is an *experimental* video player, and likely always will be. It is intended for advanced users only, and may not perform as expected due to limitations within GameMaker Studio.

Introduction: What's an FMV?

'FMV' stands for **F**ull **M**otion **V**ideo, and historically has been used as an extension for game videos since the early 90s. The name itself does not connote any specific video format, and in fact more often than not implies a non-standard or proprietary format created specifically for use in games.

Despite being experimental, Edge FMV is designed to be highly flexible and easy to use. Playing your own videos with Edge FMV comes in two stages: **conversion** and **operation**. While the second stage is the same no matter

what, the first will vary slightly based on the target platform and project needs.

Although Edge FMV-compatible video files can be made with a variety of tools, it is strongly recommended to use the included **MakeFMV** application to convert standard video format files into files readable by Edge FMV. Edge FMV **cannot** read standard video filetypes directly (such as MP4, AVI, or WMV, to name only a few). Converting with MakeFMV will ensure the best possible balance of image quality, file size, and game performance, and will also simplify the conversion process for quick and easy importing.

Utilization: What's a Preload?

Edge FMV can read converted files through two methods of storage: **preload**, and **precache.** Each method has advantages and disadvantages, and it is up to each developer to decide which is best for their projects.

The **preload** method stores videos as a single contained file much like traditional video formats, making it clean and easy to work with. Preloading will also yield the highest performance within the GameMaker Studio IDE itself, as only one file is backed up when the project is saved or tested. As such it is recommended to use this method when *developing* games, however it may not be best for exporting to the final product. Preload files must be cached to system storage before usage, which will temporarily halt the game until the cache is fully created—a process which can take several seconds to complete, depending on the speed of the system storage medium. Furthermore, this method requires twice the actual storage space per video, as both the cache and its parent file occupy the same amount of space. It is also not guaranteed that this cache will be unloaded later, as various factors may prevent GameMaker Studio from deleting the cache if it is deemed still in-use. Preload files are also not compatible with all platforms, in which case attempting to load one will silently fail.

That's where the **precache** method comes in. As its name implies, a precache is essentially a preload file which is already in system cache form. Rather than exist as a single file, a precache is an entire folder containing individual video frame and audio files. The precache method is compatible with almost all platforms and introduces virtually no delay when initializing playback, plus it requires no additional space on the system storage to operate. The downside to a precache is simply its cumbersome number of files. A video's entire precache folder must be imported to GameMaker Studio for playback, which can take several minutes to complete depending on the length of the video. This folder is also backed up upon saving or testing the project, adding significantly to the length of both operations. Therefore, the precache method is well suited for *finished* projects instead.

Implementation: What can it do?

Additionally, Edge FMV features several built-in variables that can assist in general **operation** of converted videos. While it is only necessary to run the `edgefmv_load` and `edgefmv_play` scripts to achieve video playback, making use of these variables can take Edge FMV functionality beyond simple playback for a variety of useful effects.

First, the `global.fmv_complete` variable stores the status of the video as either **true** (the video is finished playing) or **false** (the video is not finished playing). Testing this variable can be used as a means of executing code when the video is complete. For example:

```
if global.fmv_complete = true {

   room_goto_next();

}
```


…will automatically go to the next room of the game when the video is finished playing.

It may also be desirable to allow users to pause videos during playback. While this is accomplished with the `edgefmv_pause` *script* and not a variable, the `global.fmv_pause` variable itself can be used to execute code when a video is paused. For example:

```
if global.fmv_pause = true {

   draw_set_alpha(0.5);

   draw_rectangle_color(0, 0, room_width, room_height, c_black, c_black, c_black,
c_black, false);

   draw_set_alpha(1);

}
```

…will draw an overlay to fade the video while paused.

Furthermore, to aid in stylization of elements such as a pause screen, Edge FMV includes two variables retaining the dimensions of the current video: `global.fmv_frame_width` and `global.fmv_frame_height`. These variables are scaled, meaning they contain the dimensions of the video as it is *drawn*, not necessarily as it is stored on-disk. See information on the `edgefmv_play` script for more on video scaling.

And these are but a few of the variables that are accessible to developers. Advanced users are encouraged to study the `edgefmv_load` script and use the variables it initializes to their advantage. For all that Edge FMV has to offer under regular usage, refer to the rest of this reference guide.

5.1. edgefmv_load(fname)

```
edgefmv_load(working_directory + “videosmyvideo.fmv”);
```

The first script that must be run for FMV playback. As the name implies, `edgefmv_load` will cache preload files and set a number of essential variables for both preload and precache videos. It will also return the integer 1 or the number of files cached to indicate that the load operation was successful. If the load fails, 0 will be returned instead.

The script requires only one parameter, the **fname**, or *filename* of the FMV to be loaded. This should be input as a string (enclosed in quotes), including the file’s extension and any necessary path information. In this case, the root directory is the ‘Included Files’ section of the GameMaker Studio project. For example, the path “foldermyfile.fmv” would actually refer to “C:UsersUSERDocumentsGameMakerProjectsPROJECT.gmxdatafilesfoldermyfile.fmv”. Do not include a slash at the beginning of the path. If no sub-folder is used, no slashes are necessary.

This script must be run in the Create event or elsewhere that will only trigger it once. It does not, however, have to be run in the same object or even the same room as the `edgefmv_play` script. This is especially useful for preload files, as preloading can occur during a loading screen elsewhere in the game prior to actually playing the video.

Note that regardless of whether a preload or precache is used, the input fname should include both the filename and extension. In the case of a precache, the filename should match that of the folder, and the extension should match that of the frame files’ extension (typically ‘.fmv’). Files converted with MakeFMV will use the same fname regardless of whether they are stored in preload or precache mode.

If both a precache and preload version of the same file are present, the precache will be loaded preferentially.

> Note: Once a preload file has been cached, it does not matter where it is loaded from again—the same cache will be used regardless of the original file's location provided the cache still exists after the file has been moved.
>
> Note 2: Changes to the folder structure in the 'Included Files' section will require re-importing files to the new folder structure. This is a limitation of GameMaker Studio.
>
> Note 3: While only the most recent loaded video can be played, multiple videos can be preloaded and then played in succession by running `edgefmv_load` for each video again when its turn comes. If a cache already exists it will not be recreated, therefore there is no performance penalty to the second instance of `edgefmv_load`.

5.2. edgefmv_play(framerate, a/v offset, x, y, xscale, yscale, rot, color, alpha, hidemouse, loop)

```
edgefmv_play(30, 0, 0, 0, 1, 1, 0, c_white, 1, true, false);
```

Plays the most recent video loaded with the `edgefmv_load` script with a variety of playback options and transforms.

Of greatest importance is the **framerate** parameter, which sets the FPS at which the loaded video is intended to be played. Fractional values are acceptable, therefore framerates such as `23.976` and `29.97` are valid inputs here. All videos converted with MakeFMV will have a framerate of `30`.

The **a/v offset** parameter can be used to adjust the tracking between the video and audio being played, to fix incorrect lip-syncing for example. This value is in number of frames, and can be both positive and negative to offset the audio either forwards or backwards as necessary. Generally, no offset should be necessary, in which case this parameter can simply be set to `0`.

The **x** and **y** parameters specify where the video should be drawn, with the top-left corner being the video's origin point. The **xscale** and **yscale** parameters set the size of the drawn video as well as the actual values stored in the built-in `global.fmv_frame_width` and `global.fmv_frame_height` variables. A value of `1` equals 100% size. Negative values can be used to mirror the video on either axis.

The **rot** parameter sets the video's rotation in degrees, where `0` is straight up and `180` is directly upside-down. Although videos are positioned by the top-left corner, applying rotation will rotate them about the center.

The **color** parameter sets a blending color to add a tint to the video, where `c_white` is neutral. The input color can be anything from a built-in GameMaker Studio variable to a color made with `make_color_rgb` or even Edge Engine's own `make_color_hex`.

The **alpha** parameter is a fractional value that sets the video's transparency, where `0` is completely transparent and `1` is completely opaque.

Next, the **hidemouse** parameter is a true/false value that enables or disables hiding the mouse cursor during video playback on desktop platforms where a mouse cursor is used. The cursor will be restored when the video is completed or skipped.

Lastly, the **loop** parameter is another true/false value that enables or disables endlessly repeating video playback until the video is skipped.

This script must be run in the Draw or Draw GUI event.

> Note: While only the most recent loaded video can be played, multiple videos can be preloaded and then played in succession by running `edgefmv_load` for each video again when its turn comes. If a cache already exists it will not be recreated, therefore there is no performance penalty to the second instance of `edgefmv_load`.

5.3. edgefmv_unload(fname)

```
edgefmv_unload(working_directory + “videosmyvideo.fmv”);
```

Resets all FMV playback variables to null values and attempts to delete the relevant preload cache, if any. See `edgefmv_load` for usage of the **fname** parameter.

Note that while only one video can be loaded at any given time, the same does not necessarily apply to unloading. When videos are preloaded, their cache remains on the system storage until it is unloaded manually. In this way multiple preload files can be queued for successive playback without pausing to cache additional videos in between. However, this will also constitute data buildup on the system storage, which is undesirable in large quantities.

To unload specific videos, supply that video’s filename with any necessary path information. Otherwise to unload precache videos or to unload all preload videos simultaneously, the keyword `all` can be supplied in place of a filename here. For example, `edgefmv_unload(all)`. In general, this is the preferred way to use `edgefmv_unload`.

Like `edgefmv_load`, this script should be executed in an event where it is only triggered once.

5.4. edgefmv_pause();

```
edgefmv_pause();
```

Pauses and unpauses the current video, if any is playing.

No parameters are necessary; simply run the script in the desired input pressed event, where it will only be triggered once at a time.

5.5. edgefmv_skip()

```
edgefmv_skip();
```

Skips the current video, if any is playing.

No parameters are necessary; simply run the script in the desired input pressed event, where it will only be triggered once.

5.6. edgefmv_seek(time)

```
edgefmv_seek(10.1);
```

Sets the playback position of the current video, if any is playing.

The **time** parameter is a value in seconds. Fractional values are acceptable for seeking to a precise time in the current video. As with many other Edge FMV scripts, it is important to only run `edgefmv_seek` where it will be triggered once.

> Tip: Try setting the time value to a coordinate like `mouse_x` in a button event like Global Mouse Left. With a little additional math, you can create a full scrubbing system just like a real media player!

5.7. edgefmv_set_volume(volume)

```
edgefmv_set_volume(0.5);
```

Sets the volume of the current video, if any is playing.

The **volume** parameter is a fractional value where `0` is silent and `1` is max volume. Changes in volume will transition smoothly over a period of 100 milliseconds. This script should ideally be run in an event where it will only be triggered once.

5.8. edgefmv_get_time()

```
edgefmv_get_time();
```

Returns the playback time of the current video in seconds, if any is playing. Otherwise will return 0.

No parameters are necessary.

5.9. edgefmv_draw_progress(x1, y1, x2, y2, col1, col2, col3, col4, outline)

```
edgefmv_draw_progress(50, 50, 500, 100, c_white, c_white, c_gray, c_gray, false);
```

Draws a colored rectangle as a progress bar indicating the current video's playing time, if any is playing.

The **x1** and **y1** parameters set the position to draw the top-left corner of the progress bar, and the **x2** and **y2** parameters set the position to draw the bottom-right corner. Progress will be displayed between these two values as a percentage, where 100% equals the distance between x1 and x2.

The next four parameters set what color to draw the progress bar in, where **col1** refers to the top-left corner of the progress bar, **col2** the top-right corner, **col3** the bottom-right corner, and **col4** the bottom-left corner. These colors will be blended into a gradient.

Lastly, the **outline** parameter is a true/false value that enables or disables drawing the progress bar as an outline rather than a solid.

This script must be run in the Draw or Draw GUI event.

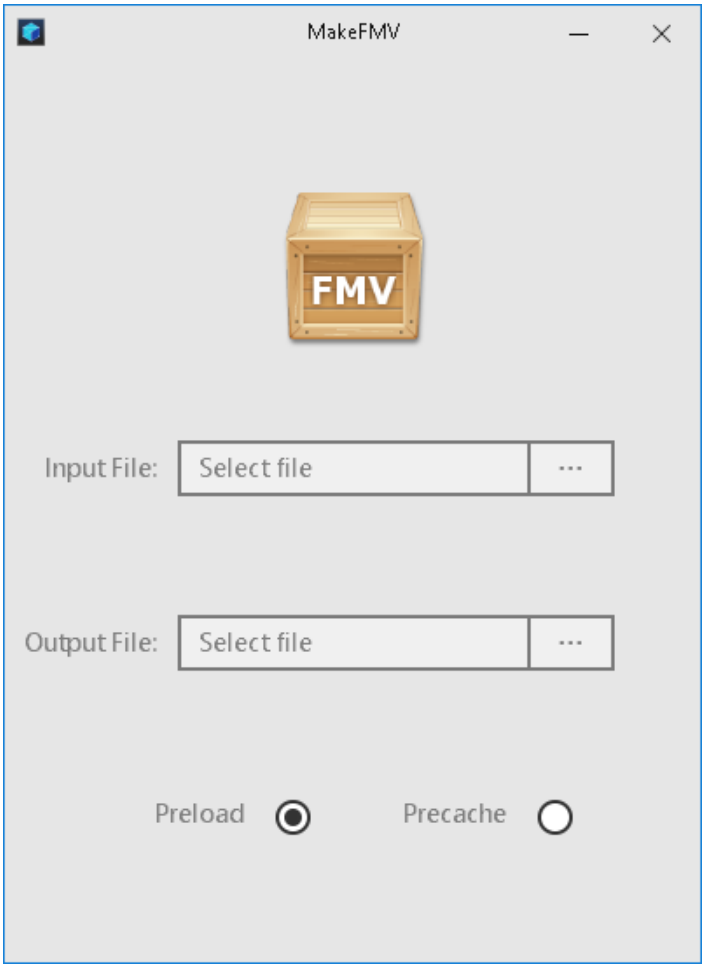
> Tip: Try running the `draw_rectangle_color` script before `edgefmv_draw_progress` to create a solid background for the progress bar to appear over!

6. MakeFMV

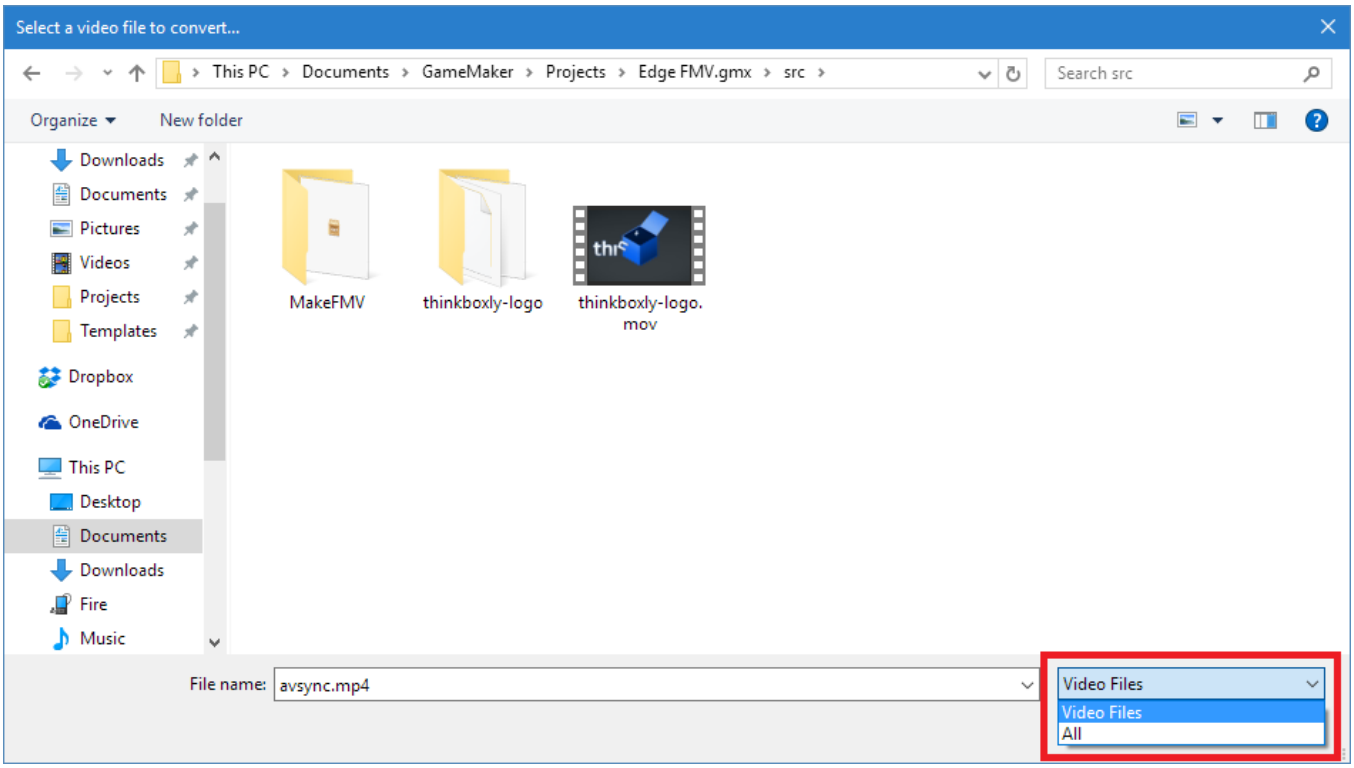
As Edge FMV is a native GML video player, it relies on a unique format for storing videos. To simplify the process of creating Edge FMV-compatible videos with the proper technical standards and right balance of quality and output file size, Edge FMV also comes with its own conversion utility: **MakeFMV**.

While MakeFMV is packaged with Edge FMV, MakeFMV itself is not commercial software and can be downloaded free of charge. Download it here.

As of version 1.1.0, MakeFMV has a GUI frontend which is very simple and easy to use.



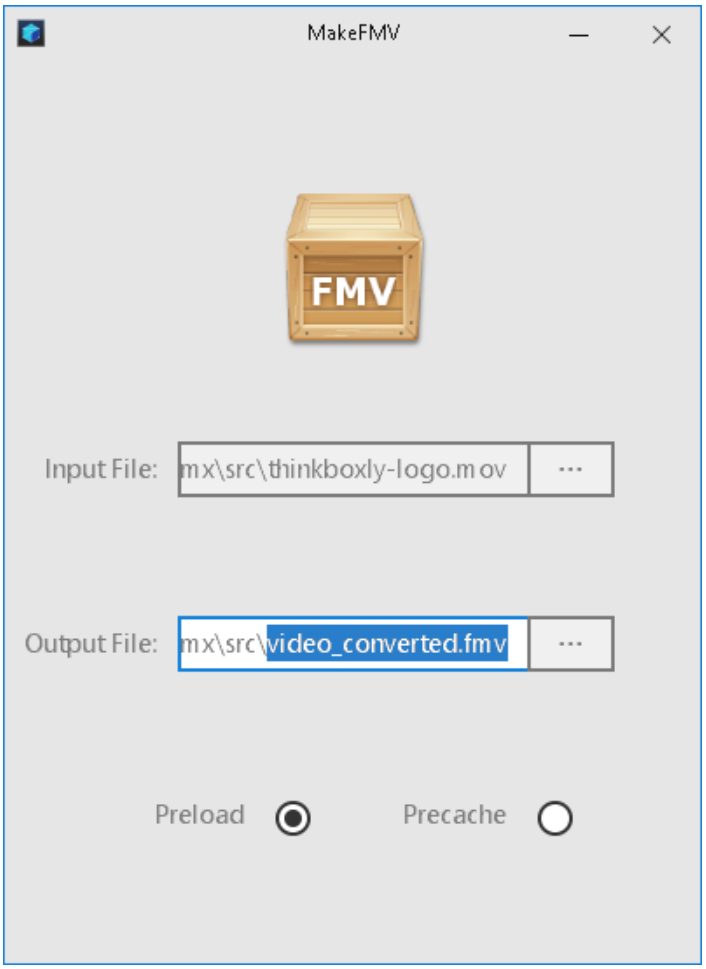
First, select the video file you wish to convert by clicking the '…' button to the right of the 'Input File' box. A familiar dialog window will appear allowing you to select your video from anywhere on your PC. By default, the list of files will be filtered by popular video file formats, but if your video is not listed you can always disable the filter by setting 'Video Files' to 'All' in the bottom-right corner of the window.



It is also possible to fill out the ‘Input File’ path manually. If you do so, remember **not** to put quotes around the path!

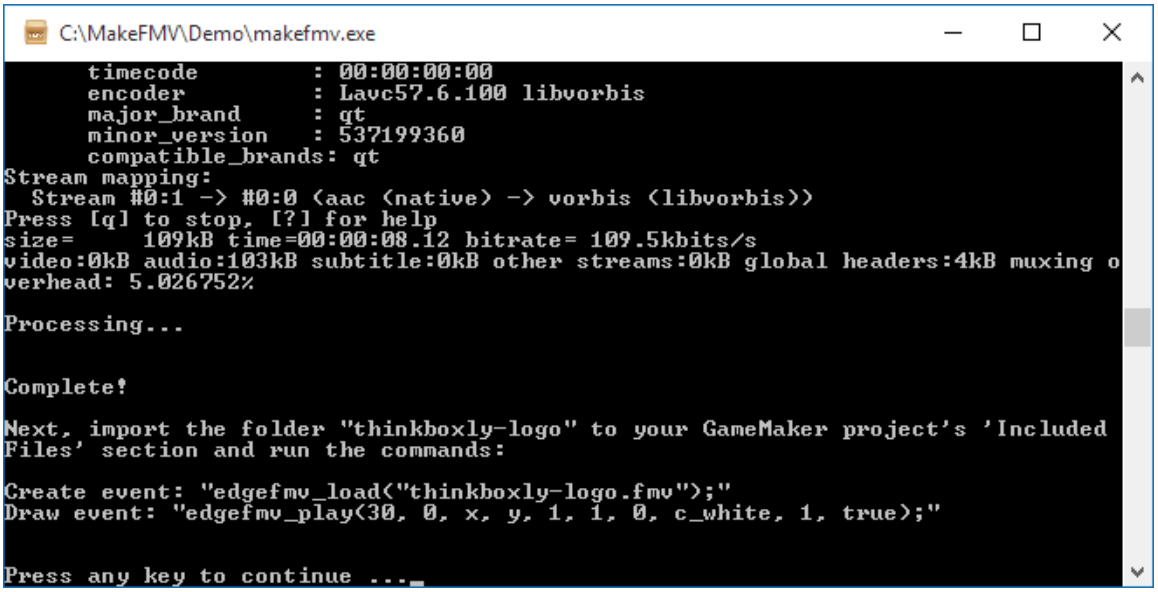
Next, repeat the same process with the ‘Output File’ path. Unlike with the input file, this time you will not be able to specify an output format—the .fmv format is required.

The output file does not have to be in the same directory as the input file. Create your .fmv anywhere you like!



Finally, select whether to convert the input video as a ‘Preload’ or a ‘Precache’ by checking the appropriate radio button at the bottom of the window. Remember, a **preload** FMV is a self-contained file that is cached at runtime, while a **precache** is a folder that requires no extra processing to be used. **For mobile devices, precache is required!**

Once you’ve made all your selections, click the big ‘MakeFMV’ logo at the top of the window and start converting! A terminal window will appear so you can keep an eye on the whole process.



When the conversion is complete, the MakeFMV terminal will provide the example Edge FMV commands needed to play the output file in your GameMaker Studio project. Simply drag the output file or folder to your GameMaker Studio project's 'Included Files' section and play it with the `edgefmv_load` and `edgefmv_play` scripts.

And that's it! Your MakeFMV-converted video is now ready to be played with Edge FMV.

Advanced Operation

As of version 1.3.0.5, MakeFMV features a hidden 'advanced' mode which can be activated at any time by pressing the 'A' button while the MakeFMV GUI is visible. This will prompt for custom settings for frame width and height, framerate, and image quality. Pressing the 'A' button again and disabling advanced mode will reset these options to their defaults.

This new mode is provided in the spirit of Edge FMV as an experimental video player. Custom conversion settings are **not recommended**, as changes to the default values may result in poor video performance. Use at your own risk!

7. End-User License Agreement ("EULA")

LAST UPDATED: 12/16/2019

We know that reading EULAs isn't very exciting, but this is important. Please take your time to review and ensure you understand the terms of this document before proceeding to use XGASOFT products in your own work.

If you have any questions or concerns about the terms outlined in this document, please feel free to contact us at contact@xgasoft.com or by visiting our Contact & Support page.

LICENSE AGREEMENT

This License Agreement (the "Agreement") is entered into by and between XGASOFT (the "Licensor"), and you (the "Licensee"). This agreement is legally binding, and becomes effective when you purchase and/or download a free product from XGASOFT or authorized third-party distributors. If you do not agree to the terms of this Agreement, do not purchase, download, or otherwise use XGASOFT products.

In order to accept this Agreement, you must be at least eighteen (18) years of age or whatever age is of legal majority in your country. Otherwise, you must obtain your parent's or legal guardian's approval and acceptance of this Agreement in your stead. XGASOFT accepts no liability for your failure to meet this requirement.

XGASOFT delivers content through authorized third-party distributors, each of which may require its own separate End-User License Agreement ("EULA"). XGASOFT accepts no liability for the terms of any third-party agreements, nor for your failure to meet them.

STANDARD LIFETIME LICENSE

This is a license, not a sale. XGASOFT retains ownership of all content (including but not limited to any copyright, trademarks, brand names, logos, software, images, animations, graphics, video, audio, music, text, and tutorials) comprising digital products and services offered by XGASOFT (the "Property"). All rights not expressly granted are reserved by XGASOFT.

Subject to your acceptance of the terms of this Agreement, XGASOFT grants you a worldwide, revocable, non-exclusive, non-transferable, and **perpetual** license to download, embed, and modify for your own purposes XGASOFT Property solely for incorporation with electronic applications and other interactive media, including both commercial and non-commercial works, wherever substantial value has been added by you.

Any source code included as part of XGASOFT Property must be compiled prior to redistribution as an incorporated work, whether for commercial or non-commercial purposes.

PATREON LIMITED LICENSE

When you register as a recurring financial supporter of XGASOFT through Patreon (Patreon, Inc.), XGASOFT may provide free access to XGASOFT Property as a reward, subject to the terms of each contribution tier. This is a privilege, not a right.

XGASOFT retains ownership of all content (including but not limited to any copyright, trademarks, brand names, logos, software, images, animations, graphics, video, audio, music, text, and tutorials) comprising digital products and services offered by XGASOFT (the “Property”). All rights not expressly granted are reserved by XGASOFT.

Subject to your acceptance of the terms of this Agreement, XGASOFT grants you a worldwide, revocable, non-exclusive, non-transferable, and **temporary** license to download, embed, and modify for your own purposes XGASOFT Property solely for incorporation with electronic applications and other interactive media, including both commercial and non-commercial works, wherever substantial value has been added by you.

Any source code included as part of XGASOFT Property must be compiled prior to redistribution as an incorporated work, whether for commercial or non-commercial purposes.

This license shall remain effective for the duration of your subscription to XGASOFT through Patreon. In the event that you cancel or reduce your contribution to a lower tier not qualifying for free access to XGASOFT Property, this license will be considered revoked and void for any and all public commercial and non-commercial activities. In order to continue using XGASOFT Property publicly, you must purchase a standard lifetime license.

This limitation shall not be applied retroactively, so that any existing, complete, and publicly available commercial and non-commercial properties using XGASOFT Property will not be considered in violation of this agreement. Furthermore, this limitation shall not apply in the event that XGASOFT suspends, revokes, or disables the contribution of financial support to XGASOFT through Patreon. In such case as contributions are limited or prohibited by XGASOFT (and not the Licensee), the terms of the Standard Lifetime License shall apply to any and all XGASOFT Property granted as rewards for recurring financial support prior to the date of suspension.

SINGLE-USER

This Agreement grants one (1) user an applicable license to use XGASOFT Property on unlimited devices. This license may not be transferred, shared with, or sold to other users.

However, you, the Licensee, may use XGASOFT Property along with a team or company of collaborators wherever substantial value has been added by you.

This limitation does not extend a license to other users. For any works unrelated to you, collaborators must purchase separate licenses.

MODIFICATIONS

In accordance with the terms of this Agreement, you may freely modify, or alter the functionality of XGASOFT Property exclusively for your own use.

Modifying the Property will not terminate your license, however XGASOFT cannot guarantee the quality and functionality of modified versions of the Property, nor its compatibility with other products.

XGASOFT accepts no liability for any loss or damage incurred by the modified Property, and reserves the right to refuse technical support for the modified Property.

Modifications made to XGASOFT Property in no way represent a change of ownership of the Property.

You may not reverse-engineer XGASOFT Property for the purpose of commercial exploitation which may be in competition with XGASOFT.

MUTABILITY

License fees are determined for each product and service on a case-by-case basis, and XGASOFT reserves the right to change fees on the Property with or without prior notice.

XGASOFT reserves the right to modify, suspend, or terminate this Agreement, the Property, or any service to which it connects with or without prior notice and without liability to you, the Licensee.

LIABILITY

By using XGASOFT Property, you agree to indemnify and hold harmless XGASOFT, its employees, and agents from and against any and all claims (including third party claims), demands, actions, lawsuits, expenses (including attorney’s fees) and damages (including indirect or consequential loss) resulting in any way from your use or reliance on XGASOFT Property, any breach of terms of this Agreement, or any other act of your own.

This limitation will survive and apply even in the event of termination of this Agreement.

GOVERNING LAW AND JURISDICTION

This Agreement shall be governed by and interpreted according to the laws of the United States of America and the State of Kansas.

If any provision of this Agreement is held to be unenforceable or invalid, such provision will be changed and interpreted to accomplish the objectives of such provision to the greatest extent possible under applicable law, and the remaining provisions will continue in full force and effect.

CONCLUSION

This document contains the whole agreement between XGASOFT and you, the Licensee, relating to the Property

and licenses thereof and supersedes all prior Agreements, arrangements and understandings between both parties regarding XGASOFT Property and licenses.

