Edge Display Scaler Reference Guide

by XGASOFT

1. Welcome to Edge Display Scaler

Thank you for choosing Display Scaler, the first published member of the Edge Engine family. **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 Display Scaler (or Edge DS) is a lightweight scaling solution to run your Game Maker Studio projects in any resolution, at any aspect ratio and respond intelligently to changes in resolution in real time. In today's market of mobile and multiplatform games a good scaling solution is a must. Regardless of what platform you're building on, Edge Display Scaler has you covered…and all it takes to get started is a single object.

In this guide you will learn about the multiple scaling methods Edge Display Scaler offers and how to utilize them in your projects to achieve the desired results. You will also learn concepts such as relative scaling and positioning—core programming techniques required to take a scaled viewport and apply the same scaling to your own sprites and objects. Remember, no amount of scaling will automatically make a project *look good* when scaled—that part is up to you.

2. Buy Now(https://marketplace.yoyogames.com/assets/1257/display-scaler-edge-engine)

3. Download PDF(https://docs.xgasoft.com/wp-content/uploads/sites/2/edge-display-scaler-reference-guide-7.pdf)

4. What's New

v1.7.0

• Improved performance, especially when downscaling

• Improved compatibility with HTML5

• Added fit and proportion scaling to `edgeds_draw_background` for parity with `edgeds_draw_sprite`

v1.6.7

• Updated documentation to new format

v1.6.6

• Fixed a bug causing backgrounds to not scale properly across rooms

v1.6.5

• Improved GUI layer scaling

• Added aspect ratio window resizing for `edgeds_set_scale`

• Improved aspect ratio window resizing for `edgeds_set_screenres`

• Improved performance

v1.6.1

• Fixed a regression from previous versions causing performance to drop as a result of DPI calculations

v1.6.0

• Fixed a bug causing inaccurate scaling on HTML5 platforms when using `edgeds_set_width` or `edgeds_set_height`

v1.5.9

• GUI layer is now automatically scaled along with the application layer

v1.5.8

• Spring cleaning—removed non-view scaling as this feature was deemed unuseful and unnecessary

• Removed mobile hi-res workaround, as >2K screens are now supported by GameMaker natively (thanks, YoYo!)

• Added aspect ratio window resizing for `edgeds_set_width`, `edgeds_set_height`, and `edgeds_set_screenres`

• Improved, more accurate handling of DPI-based scaling

• Improved, more accurate scale linking. No longer experimental!

• Minor additional improvements and optimizations

v1.5.3

• Added auto mode to view zooming, allowing to minimize the differences in viewport pixel dimensions between resolutions

• Fixed issues in asset scaling scripts causing inaccurate scaling while following an object with the master view

v1.5.1

• Added optional view zooming to all `edgeds_set_*` scripts

v1.5.0

• Added optional minimum resolutions to applicable `edgeds_set_*` scripts

• Added DPI adjustment warning to built-in stats

• Syntactical adjustments for consistency (see 'Migrating Versions')

• Merged `edgeds_get_prevscale` with other scripts; `edgeds_get_prevscale` no longer necessary

• Fixed an issue preventing scaling from carrying over across rooms

• Additional fixes and improvements

v1.4.1

• Minor bug fixes

v1.4.0

• Complete overhaul! Majority of code re-edited, re-written, simplified, and/or improved

• All-new scripts for scaling backgrounds, sprites, and views

• Can now scale any view, or disable view scaling entirely (see documentation)

• Added optional DPI scaling to all `edgeds_set_*` scripts

• Added DPI override setting to force a given DPI on any display

v1.2.8

• Improved support for >2k displays on devices not supporting >2k texture pages

• Improved DPI handling in DPI scaling mode

• Performance improvements

• Minor bug fixes and general improvements

v1.2.6

• Added support for >2k displays on Android and iOS devices (other platforms already supported)

• Added optional DPI mode to `edgeds_set_scale`

• Added `edgeds_scale_width` and `edgeds_scale_height` scripts for single-axis scaling

• General improvements

v1.2.3

• Fixed an issue causing crashes when the game window is minimized

v1.2.2

• Updated `edgeds_view_fit` script to utilize two additional views as letterboxes

v1.2.1

• Added `edgeds_set_screenres` script

v1.2.0

• New naming structure for Edge Display Scaler scripts, for uniformity across Edge Engine modules

• Included ‘Legacy Scripts.zip’ to ease transition from previous versions

• HTML5 users no longer need to manually edit a script to enable scaling on HTML5 platforms

• Added `edgeds_background_fill` script for scaling backgrounds to fill the view

• Added `edgeds_view_fit` script for scaling additional views to fit within view 0, the ‘master view’

• General improvements

v1.1.2

• Fixed an issue with HTML5 scaling

v1.1.1

• Added syntax guides

v1.1

• Rebranded as Edge Engine Display Scaler

• Completely rewritten documentation

• New tools: `image_scale_abs` and optional scale linking in `move_link` (experimental; disabled by default)

• General improvements

v1.0

• Initial release as Simple Display Scaler

5. Migrating to New Versions of Edge Display Scaler

As Edge Display Scaler matures and develops, old features sometimes become incompatible with the latest updates. This does not mean projects built on older versions of Edge DS cannot be upgraded, but it does mean the process is not automatic and will require some effort on the part of the developer.

Note that the latest version of Edge DS should already be installed in your project before following this guide. To install Edge DS on top of older versions, remove Edge Display Scaler from your project’s ‘Extensions’ folder and delete the entire ‘Edge Display Scaler’ folder from your ‘Scripts’ folder. Also delete any Edge DS demo objects from your ‘Objects’ folder, leaving only your own custom objects and code behind. To update these custom project assets, refer to the migration guide below.

• **Scaling scripts will now automatically scale the GUI layer along with the application layer.**

This means it is no longer necessary to run the `display_set_gui_maximize()` command manually, and any existing instances of this function should be removed when using Edge Display Scaler.

• **edgeds_init_scale has been reduced to a single argument for view scaling.**

As non-view scaling was determined to be an unnecessary feature it has been removed and as such it is no longer possible to enable or disable view scaling—it is enabled by default. As a result, all instances of `edgeds_init_scale` must be updated to include only one argument setting the view to scale. See documentation for further details on the updated syntax.

• **`edgeds_set_scale`, `edgeds_set_screenres`, `edgeds_set_width`, and `edgeds_set_height` have all been given optional DPI scaling support with optional DPI override, minimum resolution, and zoom support.**

All existing instances of these scripts must be updated to either enable or disable DPI scaling and set either auto or manual DPI, as well as set the minimum scaling resolution and zoom multiplier. See documentation for further details on the updated syntax.

• **`move_link` now features both movement and scale linking which can be toggled independently.**

This means there are now three parameters to the script and existing instances will need to be updated to include these new values. See documentation for further details on the updated syntax.

6. Overview

Using Edge Display Scaler comes in two stages: 1) Scaling the game itself, and 2) programming your objects to *respond* to that scaling by adjusting themselves accordingly ("responsive design"). Thankfully, all you need to get started is the first step, and with Edge Display Scaler, it's practically done for you.

First, open your project's `Global Game Settings` page, navigate to the tab for your desired platform, and under the `Graphics` settings set the `Scaling` option to `Full Scale` or `Stretch`. On platforms that support it, you should also ensure the `Interpolate Colors Between Pixels` box is checked.

Next, create or open the first room of your game project and switch to the `Views` tab. Ensure the `Enable Use of Views` box is checked, and set the parameters for the view of your choice according to the desired base resolution for your game. Whatever *port* width/height are used will act as your base resolution—the point at which the game is operating at a 1:1 scale.

> Note: Whichever view you use must be reserved in every room as the 'master' scaling view. Edge Display Scaler will enable this view automatically if it is not already activated, but keep in mind that it is not available for normal use. To use views for your own purposes, only use the other seven available views instead.

> Note 2: Not sure what the difference between 'view' and 'port' is? It's easy: think of the *view* as game pixels and the *port* as screen pixels. The two don't have to be equal, but in the end, 'port' always rules. No matter how much or little of the game view is drawn, it will always be stretched to fit the port. Likewise, if your game is not running in fullscreen, the game window will always be the size of the largest active port.

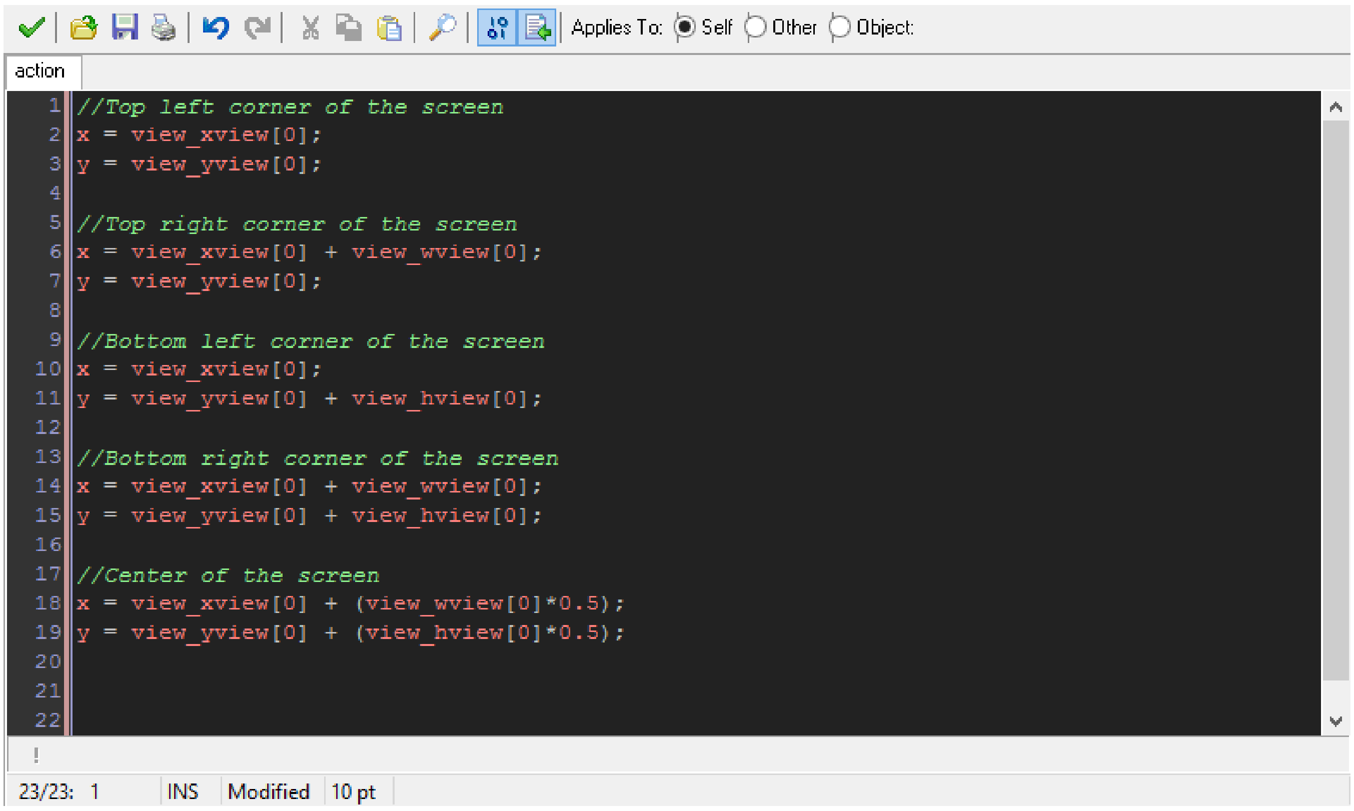
Finally, drop the included `obj_scaler` object into the room. It can go anywhere you like and is not visible by default. This object is also persistent, meaning you only need to include it once at the beginning of your game to achieve perfect full-display scaling entirely throughout.

> Note: Enabling visibility of the obj_scaler object will draw a variety of debug statistics in the top left corner of the screen, from FPS to display size to DPI to orientation!

And that's it! The most important step is now done, but now's when the real work begins. Neither this nor any other guide can tell you precisely how to use the power of Edge Display Scaler. That depends on you and your game. What this guide can do, however, is educate you on the tools at your disposal so you can go about using them effectively for yourself.

The first of these tools is actually not a part of Edge Display Scaler at all, nevertheless it is fundamental in understanding responsive design. That tool is **procedural positioning**. Normally, to position an object you simply drag and drop it into a room editor, but to use display scaling effectively you must learn new, programmatic positioning methods. Some objects, such as menu or GUI elements, must remain on-screen at all times, even if the size and shape of that screen changes either in real-time or just from device to device. To accomplish this, objects must be positioned relative to the view with a little code.

Take a look at the following example:



```
//Top left corner of the screen
x = view_xview[0];
y = view_yview[0];

//Top right corner of the screen
x = view_xview[0] + view_wview[0];
y = view_yview[0];

//Bottom left corner of the screen
x = view_xview[0];
y = view_yview[0] + view_hview[0];

//Bottom right corner of the screen
x = view_xview[0] + view_wview[0];
y = view_yview[0] + view_hview[0];

//Center of the screen
x = view_xview[0] + (view_wview[0]*0.5);
y = view_yview[0] + (view_hview[0]*0.5);
```

Assuming View 0 is set as your master scaling view, `view_wview[0]` and `view_hview[0]` will always refer to the full width and height in pixels of the current display—be it a window, a fullscreen application, or running in a browser with HTML5. Use these and other measurements to position your objects relatively, rather than absolutely. Think of your coordinates as *percentages* of the game window, rather than absolute pixels.

With this basic setup your project will scale itself according to the default settings for Edge Display Scaler, however you will most likely wish to customize them to best suit your specific project. To learn how, refer to the functions in the rest of this guide.

6.1. edgeds_init_scale(view)

```
edgeds_init_scale(0);
```

Run once at the creation of `obj_scaler`. Sets the base resolution for the game and defines all the main variables needed for display scaling. The **view** parameter specifies which view (0-7) to scale and retrieve the base resolution from. Note that base resolution is retrieved from the view's *port* as set in the room settings of the room containing the scaler object.

Note that `edgeds_init_scale` initializes a number of global variables that can be referenced elsewhere for various display scaling purposes. These variables are:

- `global.display_view` (refers to the master scaling view)
- `global.display_wres` (the base horizontal resolution for the game)
- `global.display_hres` (the base vertical resolution for the game)
- `global.display_xprevscale` (refers to the previous scale of the display, relative to the base resolution)
- `global.display_yprevscale` (refers to the previous scale of the display, relative to the base resolution)
- `global.display_xscale` (refers to the current scale of the display, relative to the base resolution)
- `global.display_yscale` (refers to the current scale of the display, relative to the base resolution)
- `global.display_orientation` (refers to whether the game is running in portrait or landscape)
- `global.display_zoom` (refers to the current zoom multiplier of the display, where 1 = 100%)
- `global.display_dpi` (refers to the actual DPI or PPI of the display)
- `global.display_dpi_current` (refers to the virtual/override DPI or PPI of the display)

Use these variables in your own code in any object to make the most of what Edge Display Scaler offers.

6.2. edgeds_set_scale(dpi mode, dpi override, minwidth, minheight, zoom)

```
edgeds_set_scale(false, 0, 640, 360, 1);
```

Calculates the current display scale and orientation and re-scales the game to match.

As of version 1.2.6, `edgeds_set_scale` also comes with the option to enable a DPI (or PPI) scaling mode that will attempt to adjust the scale of the application based on the *physical* size of the display, not just the actual number of pixels. This is especially important for mobile devices, such as high-end Android smartphones, which can have super high-density displays in very small sizes. While DPI scaling is very useful, be aware that it is also exponentially more complex to work with, and relying on it will require entirely pixel-agnostic programming to position elements where you want. Think of DPI mode as scaling in *percentages* and standard mode as scaling in *pixels*. To enable DPI mode, set the **dpi mode** parameter to 'true'. Otherwise set it to 'false' for standard scaling (recommended).

As of version 1.4.0, Edge Display Scaler also permits DPI override, which will force the display to scale to the DPI value specified in the **dpi override** parameter. This feature should be used carefully, and is generally recommended to leave disabled by setting the **dpi override** parameter to 0.

The **minwidth** and **minheight** parameters set a minimum resolution limit in pixels to prevent the game from scaling below a desirable size. For no minimum, set both of these values to 0.

As of version 1.5.1 scaled views can also be zoomed with the **zoom** parameter. This value is a multiplier where a value of 1 is default, or 100% of the view dimensions plus any DPI adjustments that may be enabled. Larger values equals greater zoom factor. A value of 0 or -1 will enable auto zoom, which will attempt to compensate for differences in resolution so that the view always displays approximately the same in-game area. Note that this area will inevitably still vary somewhat based on aspect ratio.

This script should **not** be run simultaneously with any other `edgeds_set_*` script, and ideally **should** be run in the Step event.

6.3. edgeds_set_width(dpi mode, dpi override, width, minheight, zoom)

```
edgeds_set_width(false, 0, 1920, 360, 1);
```

Functions as an alternative to `edgeds_set_scale`.

This script will force an absolute pixel **width** while scaling height dynamically to fill the display by aspect ratio. This can be handy as an easier and more predictable alternative to other scaling methods, though sometimes more limited in its scope.

As usual, the **minheight** parameter still sets a minimum vertical resolution limit in pixels to prevent the game from scaling below a desirable size. For no minimum, set this value to `0`.

This script should **not** be run simultaneously with any other `edgeds_set_* script`, and ideally **should** be run in the Step event.

As of version 1.4.0, this script also features optional DPI mode and DPI override settings.

As of version 1.5.1 this script also features optional view zooming. See documentation for `edgeds_set_scale` for usage details.

6.4. edgeds_set_height(dpi mode, dpi override, minwidth, height, zoom)

```
edgeds_set_height(false, 0, 640, 1080, 1);
```

Functions as an alternative to `edgeds_set_scale`.

This script will force an absolute pixel **height** while scaling width dynamically to fill the display by aspect ratio. This can be handy as an easier and more predictable alternative to other scaling methods, though sometimes more limited in its scope.

As usual, the **minwidth** parameter still sets a minimum horizontal resolution limit in pixels to prevent the game from scaling below a desirable size. For no minimum, set this value to `0`.

This script should **not** be run simultaneously with any other `edgeds_set_*` script, and ideally **should** be run in the Step event.

As of version 1.4.0, this script also features optional DPI mode and DPI override settings.

As of version 1.5.1 this script also features optional view zooming. See `edgeds_set_scale` for usage details.

6.5. edgeds_set_screenres(dpi mode, dpi override, width, height, zoom)

```
edgeds_set_screenres(false, 0, 1920, 1080, 1);
```

Functions as an alternative to `edgeds_set_scale`.

Rather than give the user free reign to scale the display, `edgeds_set_screenres` will force the display to conform to the specified **width** and **height** dimensions.

This script should **not** be run simultaneously with any other `edgeds_set_*` script, and ideally **should not** be run in the Step event, but rather triggered just once (e.g. in the Create event).

As of version 1.4.0, this script also features optional DPI mode and DPI override settings.

As of version 1.5.1 this script also features optional view zooming. See `edgeds_set_scale` for usage details.

6.6. edgeds_draw_background(background, x, y, index, foreground, tile, scale mode)

```
edgeds_draw_background(my_bg, view_xview[0], view_yview[0], 0, false, false, 0);
```

Assigns the background asset specified in the **background** parameter to the background slot specified in the **index** parameter and scales it to fill the 100% of the display with one of three scaling modes.

As of version 1.7.0, five background **scaling modes** are available in Edge Display Scaler: x and y full scaling (`0`), x-only scaling (1), y-only scaling (2), fit scaling (3), and proportion scaling (4). Mode 0 scales a background to always fill the screen without stretching, with modes 1 and 2 offering single-axis versions of the same style of scaling. Mode 3 only scales a background downward as necessary to fit small displays while never exceeding 100% scale on larger ones, and Mode 4 scales proportionately to the display whether larger or smaller than the base resolution (without guaranteeing seamless fullscreen).

Because the background is intended to fill the display, it is recommended to set the **x** and **y** parameters set to `0` or `view_xview[0]` and `view_yview[0]`, where `[0]` refers to the desired scaling view, however the background may still be positioned freely regardless of scaling.

By setting the **foreground** parameter to `true` the background can appear on top of game assets rather than behind, otherwise `false` will display the background normally.

Finally, by setting the **tile** parameter to `true` the background will repeat in both directions, which can be useful when using manual x and y offsets. Otherwise, `false` will draw the background only once.

This script can be run in virtually any event.

6.7. edgeds_draw_sprite(sprite, index, x, y, rot, color, alpha, scale mode, notme)

```
edgeds_draw_sprite(my_spr, image_index, view_xview[0], view_yview[0], 0, c_white, 1,
4, true);
```

Draws the sprite asset specified in the **sprite** parameter with the scaling mode specified in the **scale mode** parameter.

As of version 1.4.0, five sprite **scaling modes** are available in Edge Display Scaler: x and y full scaling (`0`), x-only scaling (1), y-only scaling (2), fit scaling (3), and proportion scaling (4). Mode 0 scales a sprite essentially as a fullscreen background, with modes 1 and 2 offering single-axis versions of the same style of scaling. Mode 3 only scales a sprite downward as necessary to fit small displays while never exceeding 100% scale on larger ones, and Mode 4 scales proportionately to the display whether larger or smaller than the base resolution.

It is also possible to draw sprites with a degree of customization. The **index** parameter specifies the frame of the sprite to draw (use `-1` or `image_index` for the current frame in an animation), and the **x** and **y** parameters specify where to draw the given sprite. The **rot** parameter sets the sprite's rotation in degrees, ranging from 0-360. The **color** parameter allows for blending the sprite with different colors, such as `c_blue` for blue, or `c_white` for default. The **alpha** parameter sets the transparency of the drawn sprite as a fractional range from 0-1.

Lastly, the **notme** parameter sets whether to apply the sprite settings to the running object's assigned sprite. If this value is set to `false`, the sprite specified in this script will be assigned to the running object and any scaling performed on the drawn sprite will be applied directly to the object as well. Conversely, by setting this parameter to `true` only the drawn sprite will be affected (standard behavior as compared to Game Maker Studio's `draw_sprite` scripts).

6.8. edgeds_draw_view(view, x, y, rot, color, alpha, scale mode)

```
edgeds_draw_view(1, view_xview[0], view_yview[0], 0, c_white, 1, 0);
```

Draws the contents of the view specified in the **view** index parameter as a surface with optional scaling applied.

The **scale mode** here is either `0` for fullscreen aspect-ratio scaling or any positive number for absolute scaling. For example, a value of 2 will draw the view at twice the original size. This parameter can also be set to `-1` for no scaling at all, which can be useful simply to take advantage of the script's other features.

The **x** and **y** parameters set where the view will be drawn, but keep in mind that when the **scale mode** is set to `0` this value will be overridden and the view will be centered first before applying any additional x and y values. Therefore, when full scaling is enabled, **x** and **y** should both be set to `0` or `view_xview[0]` and `view_yview[0]`. The **rot** parameter sets the drawn view's rotation in degrees, ranging from 0-360. The **color** parameter allows for blending the drawn view with different colors, such as `c_blue` for blue, or `c_white` for default. The **alpha** parameter sets the transparency of the drawn view as a fractional range from 0-1.

Note that drawing views this way is only intended for scaling small views *up*, **not** scaling large views *down*, as the latter will result in missing visual data. By scaling small views up, low-resolution games can run on high-resolution displays with different aspect ratios without pixel distortion. However, the practical usefulness of this functionality is limited, and `edgeds_draw_view` primarily exists as a quick and dirty way to convert unscaled games to scaled. For regular usage it is strongly recommended not to rely on this script.

6.9. move_link(child object, pos link, scale link)

```
move_link(obj_child, true, true);
```

'Links' the running object as a virtual parent to the specified **child object**.

Contrary to Game Maker Studio's built-in 'parent' system, `move_link` allows for child objects to move and scale independently and merely follow the parent *relatively* whenever the parent is moved or scaled. This can be very useful when scaling objects that are designed to interact with each other, such as a virtual chess board with its various playing pieces on top. In situations like this, the 'parent' object can be scaled with a script such as `edgeds_draw_sprite`, and with `move_link` all 'child' objects will follow suit even without running a scaling script of their own.

Can apply to all instances of an object or just one. For example, `move_link(obj_child, true, true);` would link *all* instances of the object `obj_child` to the running object, while `move_link(1234, true, true);` would link only the *instance* with the ID ‘1234’.

As of version 1.4.0, Edge Display Scaler also features scale linking, which will match the scale of child objects to the parent object in addition to position. Both position and scale linking can be enabled or disabled individually. To enable or disable position and scale linking, set the **pos link** and **scale link** parameters to `true` or `false`, respectively.

> Note: Objects linked with this script should **not** also be linked with Game Maker Studio’s own parenting system! Only one or the other can be used, otherwise conflicts will occur and `move_link` will be unable to scale child objects as intended.

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

LAST UPDATED: 02/14/2019

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.

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.