RageTools™ - User Guide

Revision 1.1.7

Disclaimer: RageTools, all its components and related code are registered trademarks of *Breno Azevedo da Silva* and *Freakow Interactive*, hereby called *Freakow*.

License Agreement: By installing this Unity component pack you automatically agree to the conditions of the Unity Asset Store's official License Agreement.

Suggestions and Support: Feel free to direct us any questions and suggestions about the product, all related communication should be carried out through e-mail, to contact@freakow.com. Support request e-mails should have the subject "RageTools Support". Also check back on our site (www.freakow.com) for news, you will also find general information and other RageTools users in our Unity forum thread.

Follow us - to know first about the latest updates and announcements:

Twitter: *@freakowgames, @brenoazevedo*

Facebook: *https://www.facebook.com/FreakowGames*

1. Intro

1.1. What is *RageTools*?

1.1.1. RageTools is a set of components aimed to bring Adobe Flash™-like functionality, and more, to the Unity™ platform. That includes import of vector-art (like those from Adobe Illustrator™ and InkScape), controller creation, GUI-alignment options and adaptive-resolution anti-aliasing.

1.2. What are RageTools requirements?

1.2.1. Unity 3.5.3 or above (older versions not supported)

1.2.1.1. RageTools was developed on the Windows version of the Unity Editor but has been tested and worked fine on the Mac OSX. Please report if you find any problems.

1.2.2. *RageSpline latest version*, available for purchase at the Unity Asset Store. One of the components, CanvasAlign, doesn't require RageSpline to work.

1.3. Why does RageTools require RageSpline?

1.3.1. RageSpline is responsible for all lower-level vector rendering functionality of RageTools, including converting the curves/cubic bezier-spline data to polygons on the fly. Not only it wouldn't be wise trying to re-invent the wheel, it also allowed us more time to invest on the high level. After version 1.1 we've acquired the RageSpline product from its original creator, Juha Kiili. This gives us the power to make all products work and develop harmoniously, providing the best possible results to our end-users.

2. Install.

2.1. Ok, enough of all this PR-like talk, let's cut to the chase. To install RageTools:

2.1.1. Make sure you have both the RageSpline and RageTools packages available and downloaded from the Asset Store.

2.1.2. Import the RageSpline package into your Unity project (*Assets -> Import Package -> Custom Package)*

2.1.3. In the same way, import the RageTools package into your Unity project.

2.2. After installation, you can find the components in the *RageTools/Code* project folder. You can also add the components to any selected Game Object by selecting the desired option in the auto-created *Components/RageTools* menu.

2.3. There are tutorial scenes in the RageTools/Demo/Tutorial folder, to get you up to speed ASAP. Trying them in their numbered sequence is recommended.

3. Components Description

3.1. ***RageGroup*** – This is the "heart" of RageTools, most components require it to work, either in the same game object or somewhere in its parent hierarchy. Its usability is straightforward; just add it to the desired parent game object and the "member list" containing all its children RageSplines will be automatically created. In some circumstances you will need or want to manually update the list, to do that simply hit the big clickable "Update" button. The root or parent Game Object can be either empty (generally the case) with one or many children RageSpline-enabled game objects, or have a RageSpline component attached to it as well. For some RageTools components that require it, RageGroup is auto-added to the Game Object. RageGroup also serves and centralizes the refresh of other associated RageTools components: Edgetune, RageConstraint, RageText (Pro) and RageMagnet (Pro), besides its own "Tweak" action.

3.1.1. Visible – Sets all grouped spline's Mesh Renderer component to either enabled or disabled.

3.1.2. Auto – Enables the "Auto Refresh" setting on all grouped splines.

3.1.3. Reset – Resets the shapes anti-aliasing and vertex density (RageSpline settings) to their default values, set when the "Update" button was clicked. Especially useful after using EdgeTune, to reset the original look of the shapes.

3.1.4. Update – Creates or Updates a RageGroup list. Auto-executed the first time you add the component to a GameObject.

3.1.5. Step – RageGroup employs a sophisticated "progressive update" method that provides huge performance gains, sufficient to make it very usable on mobile devices. The *Step Count* field defines how many splines will have their polygons refreshed per update. The default value of 8 is a good compromise between speed and quality, lower values like 1 or 2 are so processor-efficient that they barely affect performance, yet the refresh will "lag behind" some shapes. This may cause visible artifacts, especially when many shapes are contained in the RageGroup and there are very fast, drastic size or rotation changes. For the best possible refresh quality and to replicate the original behavior of updating all splines at once, simply set this value to zero.

3.1.5.1. Due to the very slow refresh rate of the Unity Editor, and the need for frequently refreshing groups affected by remote controllers, we have restricted the progressive updater to play or build modes. When in editor and out of play mode, the entire group always refreshes at once.

3.1.6. Draft – When it's enabled and the RageGroup takes a refresh call, doesn't re-triangulate the mesh or recalculates the physics. That provides a good redraw speed boost in most cases, but tends to produce artifacts – like outlines getting thinner or disappearing in some parts or distorted shapes. It may prevent holes or "rips" from showing up in the shape when deforming it though. It's up to the developer to test it and see if it works for his needs.

3.1.7. Tweak – Check this on to enable real-time, bulk-setting options for all RageSplines in the list. While *Tweak* is on and Multiple (see below) is set to off, any change to the Group settings automatically updates the Member List.

3.1.7.1. AntiAlias – The anti-aliasing value to be set to all member RageSplines.

3.1.7.2. Density – The Vertex Density value to be set for the whole group. Vertex Density is the number of generated polygon points divided by the number of control points. In practice, the higher it is, the smoother and less faceted the curved segments (lines between vector points) of your RageSplines will look, at the cost of more triangles being generating and potentially lowering performance – so use it with discretion.

3.1.8. Multiply – When *Multiply* is set to on (the default), *AntiAlias*, *Density* (if Tweak enabled) and Opacity values become multipliers to each member's default values. Besides, the Member List default values stop being updated dynamically to prevent invalid results. After you hit *Update*, the multiplier values are automatically re-set to 1 for consistency.

3.1.8.1. **Important:** After you're done adjusting your shapes with *Group Tweak* and *Multiple* enabled, click the *Update* button to commit the result.

3.1.8.2. If you enable "Tweak" and there's an active Edgetune component on the same Game Object, RageGroup automatically disables Edgetune (ie. turns off its "Live" toggle) and warns the user about it.

3.1.9. Opacity – When the Group is Visible, you can adjust its opacity using this control. Zero is fully transparent, One is fully opaque. Notice that, although a fully transparent group looks the same as a non-visible group, the second is faster to compute.

3.1.10. Pin UVs – Enables the "Pin UVs" setting on all grouped splines. Very useful to simulate distortion effects on image-mapped shapes.

3.1.11. Optimize – Enables the "Optimize" setting on all grouped splines.

3.1.11.1. Angle – Adjusts the 'Optimize Angle' value on all grouped splines.

3.1.12. ***(new)*** FlipX – Horizontal-mirror toggle for the group. It requires that the pivot of the group is centered (tip: apply Pivotools) and that no local rotation is applied to the group's object (tip: nest it under a controller, rotate that instead).

3.1.13. Styles– Expanding the *Styles* foldout, you can access the powerful group styling settings in RageGroup. With them you can use standard RageSpline styles to quickly change visual attributes and assign physics settings to a number of RageSplines. To create a RageSpline style check any RageSpline's Inspector editor, enter a name in the 'New Style' field and click 'create'. The new style asset will show up at the root of your Project, you can use the filter (next to the magnifying glass icon) to find it quickly. After you drop or select the first style to the style field, the filtering and application settings show up.

3.1.13.1. **Attention:** Applying a Style to a group removes any existing RageSplineStyles applied to the grouped RageSplines.

3.1.13.2. Name: The styling entry's style name shows here. You can have multiple entries with different filters.

3.1.13.3. Filter: Text string used to filter the spline names that will receive this style look or physics settings. The filter text has to be in the RageSpline's game object name, regardless if at the start, middle or end of it. For instance, "_custom" will select both "_customArt" and "cap_custom".

3.1.13.4.  Delete button: Click it to exclude that styling entry.

3.1.13.5. 

New Style: Drop a style here to add it to the styling list

3.1.13.6. Apply Styles button: Click it to apply all styles to the filtered splines

3.1.13.7. Apply Physics: Click it to apply all style physics to the filtered splines

3.1.14. Members – List showing all RageSplines in this RageGroup, or "No Members" if the list is currently empty. It sports the *Name* of the Game Object it's attached to, its default anti-aliasing (*Def AA*) and default density (*Def Dns*). Read "default" as "the values set when the RageGroup was last updated".

3.1.14.1. **Attention:** the default values, shown in the "Member List", are NOT necessarily the current values on each RageSpline. That's why it's a read-only list - it doesn't fit to change the actual RageSpline values individually with those fields. These default values are used for a multitude of RageTools operations and it's very useful to control when they're updated, as you'll learn.

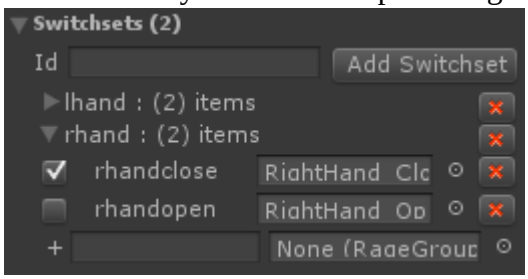
3.1.15. Ignore Groups – Many times, while setting up group hierarchies (groups within groups) you need to make sure a parent group ignores some splines already affected by a child group. That way you can, for instance, tweak the density or anti-aliasing settings of an entire character except for its head, which you want to fine-tune using its specific group (child of the 'main' group). Another very common usage is to isolate the influence of a parent RageMagnet (Pro-only component), so that it doesn't influence a child magnet and distort your shapes in unwanted ways.

3.1.15.1. Delete button: Click it to exclude that styling entry.

3.1.15.2. 

New Ignored Group: Drop a RageGroup here to add it to the Ignore list

3.1.16. Switchsets – This RageGroup block allow you to set sub-groups of mutually-exclusive shapes. Imagine multiple eye shapes or hand poses. Simply add a RageGroup to each of those shapes and assign them to the Switchset item fields. From there on you can switch among them using simple animation clip events, passing strings formatted like these examples:



- "mouth.smile" – Switches the "mouth" switchset to the "smile" item.
- "lhandopen,rhandopen" – Switches any of the group's sets which contains the "lhandopen" item and another one to the "rhandopen" item.
- "lhand.lhandopen,rhand.rhandopen" – Switches the "lhand" switchset (specifically) to the "lhandopen" item and the "rhand" set to the "rhandopen" item.

3.1.16.1. Id – Name of the next switchset to be added. Fill this in first.

3.1.16.2. Add Switchset – Creates a new switchset entry, named as above.

3.1.16.3. Switchset Entry – Expand the entry foldout to add items

3.1.16.3.1. Switch Toggle – Switches the set to this Item. Useful for tests.

3.1.16.3.2. New Switchset Item Id – Name of the next Item to be added.

3.1.16.3.3. Switchset Item Group – Drag and drop a RageGroup to create a new Switchset Item, named as the field above.

3.1.17. *Context Menu Actions* – Accessible by Right-clicking the component name

3.1.17.1. UV Fit – Auto-fits all contained splines fill texturing to the total boundary of the grouped shapes. Usually you'll draw a boundary rectangle, bigger than its sibling shapes, and use this option to automatically fit a certain texture size to all the grouped shapes at once.

3.1.17.2. Cleanup Names – When it finds duplicate names in its structure, the SVG file format automatically appends an underscore followed by a number to those names – for instance, arm_1, arm_2, etc. Not only that's not a restriction within Unity, it will ruin name-matching algorithms like the "Copy UV" macro. Firing up 'Cleanup Names' automatically remove these suffixes for you.

3.1.17.2.1. **Attention:** For this reason, we recommended using hyphens when naming shapes in your illustration application or in Unity.

3.1.17.3. Refresh Layers – When you manually tweak multiple RageLayer values on overlapping shapes, use this option to quickly refresh all RageLayer components attached to member RageSplines.

3.2. ***Rage Layer*** – ***Attention:*** *Due to the new 3D Mode (read RageSpline online docs at ragespline.com/docs-userguide),Rage Layer has become obsolete and has only been left for backwards compatibility reasons. Its usage in new projects is discouraged as the component is a prime candidate for removal in future releases.*

3.2.1. RageSplines shaders are originally designed to be used with an orthographic camera aligned to one of the base axis (X, Y or Z). If you use a perpsective camera or simply rotate an ortho camera out of one of the flat views (front, side, etc), you'll most probably find overlaid shapes changing their draw order inadvertently. To overcome this issue, you assign the RageLayer component manually in your overlapping shapes, and set incremental Z Order values to them, back to front. This way the camera can be rotated at will and the shapes will keep properly ordered. RageLayer can be automatically set while importing SVG art.

3.2.2. Z-Order – Sets the Z-draw order of the shape. The higher this value, the "closest" to the camera the shape will be rendered. Use positive integers only. Check Unity's docs on *renderer.material.renderQueue* for further info.

3.2.3. Refresh – If you change this value manually you'll need to click the Refresh button on the component. Notice you can refresh all grouped RageSplines at once using RageGroup's context menu option.

3.3. ***Rage Svg In*** – This is the SVG importer component. To use it first place your SVG files anywhere in your project folder (there are many samples at the

"SvgFiles" demo folder). After adding the component to a game object (Component > RageTools > Rage SvgIn), simply drag the SVG file object from the Project list to the *Svg File* field, and click Import. You can also import SVGs straight from web URLs, read more below. SVG-In is compatible with a versatile yet limited subset of the SVG specification, to learn about what's supported or not check the "SVG Compatibility" section.

- ***Attention:*** *Svg-In requires that a camera tagged as "MainCamera" exists on the scene. That's the default on any new scene.*

3.3.1. On Start – Imports the SVG File once the scene is played. Can reduce the file size of your deployed project, since this way your scene doesn't store the larger mesh file data.

- *Depending on the SVG complexity and the processing power of the target platform, this option might sensibly slow down the project's loading time.*

3.3.2. URL – When enabled retrieves the SVG file from a web URL. Can be used in tandem with "On Start" to import SVG files directly from the web during load. Please read: Unity docs for information on how to properly set up your web site's access.

- *Unity's webplayer and PC/Mac builds don't allow direct access to the local hard drive. You may test local files inside the editor in play mode, but for production make sure to upload your SVG file to a properly set web server.*

- *If you don't use any forward slashes (/) in the URL path – eg.: test.svg –, SVG-In assumes you want to load the file directly from the runtime folder, as informed by the Application.dataPath Unity command. This changes depending on the platform, if your SVG doesn't show up try verifying the actual path that SVG-In is trying to load from, by entering Debug mode in the inspector and enabling the "Show Debug Message" option.*

3.3.3. Import – Performs the import operation.

3.3.4. Undo Import – Removes all previously imported game objects.

3.3.5. File – Drag the SVG File object here. Check the instructions above. When the URL toggle is enabled, shows the URL address field instead.

3.3.5.1. Multiple Files Import – If instead of filling in the File field with an SVG file you assign a project folder (containing multiple SVGs), all SVGs will be imported sequentially, with all shapes nested under a game object with the same name of the file.

3.3.6. Density Min – Minimum Vertex Density. You can optionally set a starting Minimum and Maximum Vertex Density values. If these fields have different

values you automatically engage the "adaptive density" import mode, which considers the size of each shape relative to the main camera view to decide how much the vertex density of each shape should be.

3.3.7. Density Max – Maximum Vertex Density. Check the previous item.

3.3.8. Settings

3.3.8.1. AA Width – Anti-aliasing width to be applied to all imported shapes. ***Attention:*** *The Anti-aliasing might be visually inconsistent if your art has grouped scaled shapes and your RageSplines have their Antialiasing "local" setting enabled. Check Pivotools for a way to flatten its look.*

3.3.8.2. Z Offset – Layered shapes are automatically offset in the Z-axis by this amount during import. If it's set to zero you might get overlay artifacts.

3.3.8.3. Merge Radius – Automatically merges nearby points during import. Although prior shape clean up in your illustration program is recommended, try playing with this setting if you're having shape-ripping, broken corners or other problems during import.

3.3.8.4. Midline Controls – Adds a central tangent (control point) to linear segments. It doesn't change the shapes in any form or fashion, simply makes them easier to tweak later. This settings is highly recommend if you're importing shapes with straight lines or sharp corners and want to deform or animate it afterwards, either manually or using RageMagnets (available with RageTools Pro).

3.3.8.5. Outline Behind Fill – Sets the RageSplines to render the outlines behind the fill, including in the perspective view. This is generally the preferred setting if you're importing a font to use with RageText (Pro only), but bear in mind that it is incompatible with RageSpline's *emboss* mode.

3.3.8.6. Create Holes – When enabled, automatically create holes on shapes according to the SVG file compound shapes definition. Please notice that too many nested compound shapes may break the importing routine.

3.3.8.7.Auto-Layering – Automatically adds and configures the new RageLayer component to each imported spline. ***Attention:*** *Obsolete. Read more.*

3.3.8.7.1. Layer Group – When disabled, every created spline has its own RageLayer Z-Order value. When enabled, it makes all Z-Order values the same in SVG-grouped shapes. With this option on and carefully grouping non-overlapping shapes in your illustration package, you can significantly reduce the number of draw calls in

the imported art. Check the included *peng-layers.svg* art for an example. ***Attention:*** *Obsolete. Read more.*

3.3.8.7.2. Layer Materials – When enabled, forces the creation of materials in the editor, instead of only on start (after clicking play). This allows proper previewing of the layering effect right after importing the SVG, but provokes error messages in the console – which can be safely ignored. ***Attention:*** *Obsolete. Read more.*

3.3.8.8. 3D Mode – When enabled, automatically sets all created RageSplines to 3D Mode. Read the RageSpline docs for more info.

3.3.8.9. Textures Alpha ***(new)*** – When enabled, automatically sets "alpha transparent" materials to imported images. Useful for alpha-enabled bitmap sprites.

3.4. ***Rage Edgetune*** – Makes your vector shapes adaptive-resolution –much better than simple resolution independence. Automatically sets the proper anti-alias (AA) width and segments subdivision (Density) for the shape, according to its size relative to the canvas area. In layman terms, it makes sure that whatever is the look of your vector art in the scene view, it will be the look in the output device. Yeah, even in the *New iPad* outrageous 'Retina' resolution. Edgetune takes as its basis the current Unity Scene View's height. A recommended workflow is to interactively fine tune the anti-aliasing and density of a RageGroup using the *Tweak* option, disable *Tweak* then click 'Initialize' on the Game Object's Edgetune.

3.4.1. Live – When toggled on, enables EdgeTune's functionality. It differs from simply disabling the script since some functions like *Initialize* still require the script to be running.

- *If you try to enable Edgetune ("Live" toggle) and its RageGroup component has* Group Tweak *enabled, RageGroup automatically disables Edgetune's* Live *and warns the user about it.*

3.4.2. Start Only – When enabled, Edgetune is only processed once, before the game starts. Perfect for non-scaling objects to just adapt its AA and density to a certain output resolution. Since this isn't updated in real-time, it has no performance impact on the RageSpline shapes, making it an ideal option for any fixed-resolution devices like phones and tablets. Please note that this option might increase loading times if used in many dense shapes simultaneously. To use it, simply initialize as usual and make sure both "Live" and "Start Only" are enabled.

3.4.3. Initialize – Finds and stores the reference (initial) object height and camera Z distance – if using a perspective camera. Clicking this button sets the *Default Res Height* field of the scene's RageCamera component with the current Game View's size. Usually you'll click *Initialize* right before enabling *Live*.

3.4.4. Auto Density – Automatically sets the vertex density (artistically-speaking the curve smoothness, technically-speaking the polygonal subdivision level) of your shape according to its visible size. It works as a Level-of-detail (LoD) functionality so that it can actually improve frame rates with smaller or more distant objects. Very useful to scale performance according to resolution.

3.4.4.1. Max – Auto-density can potentially generate a large amount of polygons when very large subdivision values get calculated by the Edgetune algorithm. To prevent sluggish performance when a shape becomes too large on screen, cap the max density to this value.

3.4.4.2. Guess – A good rule of thumb to find a proper Max Density value is to look for the highest *Def Dens* value in the the RageGroup's *Member List*, then enter double that value in the *Max* field. As usual and repetitive tasks deserve automation, we've added the 'Guess' button for you.

3.4.5. Settings

3.4.5.1. AA Factor – Multiplying factor on the component's anti-aliasing adjustment effect. Lower values will provide for crisper AA than default as the related shapes change in visible size. You generally won't want to change the default value of 1 (one), but it's useful if you want to play with Edgetune's anti-aliasing without changing RageGroup's settings.

3.4.5.2. Density Factor – Determines how "aggressive" the subdivision/LoD algorithm is applied to the affected shape(s). The value of 0.5 is good in most cases; higher values will lead to quicker subdivision as the object scales up, resulting in smoother yet more polygon-expensive shapes.

3.4.5.3. Perspective Blur – When using the new SVG-In Auto-Layering mode, or setting up the RageLayer component manually in your overlapping shapes, the camera can be rotated and the shapes will keep properly ordered. Nevertheless, when seen from an angled point of view, the regular anti-aliasing will be insufficient, thus jaggies will show up on the edges of the shapes. To correct that, Edgetune has the Perspective Blur setting, an anti-aliasing value added to its computation according to the object's rotation relative to the camera's. Three is a good default value, meaning the maximum anti-aliasing factor is 4 (1 being the base factor). Setting this to 0 disables the effect and speeds up processing.

3.4.5.4. Debug TextMesh – For debugging purposes, you can assign an external TextMesh component to this field. It then displays the final anti-aliasing multiplying factor being calculated by Edgetune, in real-time.

3.4.5.4.1. Density – When this option is checked, the Debug TextMesh displays the calculated density factor instead.

3.4.5.5. Camera – Shows the scene camera to be used by Edgetune. By clicking the dropdown box you can select another camera.

3.4.5.5.1. Update – Click this button to refresh the scene's camera list.

3.5. ***Rage Pivotools*** – Many times after you import or draw a vector shape you want to set its pivot point (rotation center) to a specific point – or to the geometric center of the object. That's what Pivotools is for, and differently from the built-in RageSpline feature it works on multiple shapes at once. To use it simply add it to the RageGroup root object and click Center.

Important: *Make sure the pivot/center button in Unity is set to "pivot"*

- The Pivotools component can be added to any game object with the Ctrl+Shift+V (Cmd+Shift+V on OSX) keyboard shortcut. ***(new)*** Use Alt+R (Opt+R on OSX) to remove it.

3.5.1. In Place – If this option is toggled on, the pivot centering operation is carried out while preserving the current "visual position" of the game object. Nevertheless, notice that this *does* change its transform position properties.

3.5.2. Preserve Child Pivots– When enabled, prevents Pivotools from changing the pivot positions of children splines.

3.5.3. Freeze Rotation & Scale – Resets the rotation (to 0,0,0) and scale (to 1,1,1) of all Game Objects in the hierarchy, *while keeping the points in the same position.* This is a very important operation to avoid having to deal with nested transformations inside Unity and also to keep your anti-aliasing settings uniform across the scene. To us it, simply click Freeze Rotation & Scale: all hierarchy's transformations reset, same look.

3.5.4. Center – Performs the pivot-positioning operation selected in *Mode*:

3.5.4.1. Geometric – Centers the RageGroup pivot to the average, geometric center of all member RageSplines ('freeze position' in 3D apps). With *In-Place* off, the RageGroup object will be centered in the world before carrying out the centering operation. Originally called "Default".

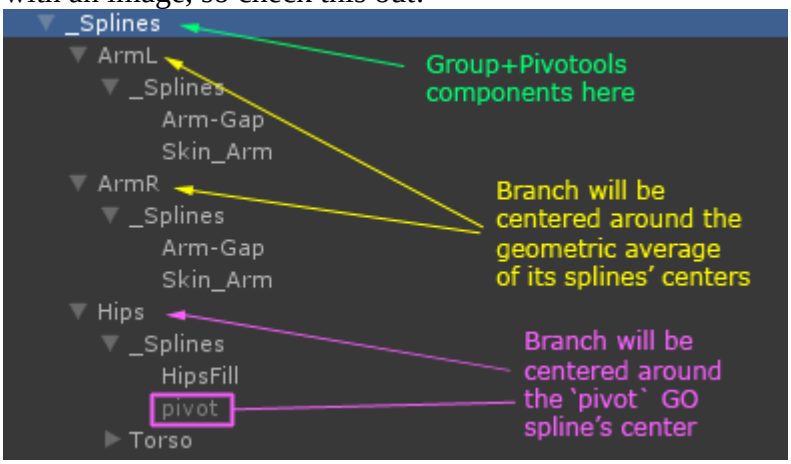
3.5.4.2. Reference – Moves the RageGroup's pivot point to a reference Game Object's transform position, when *In-Place* is on. When used with *In-Place* off, the grouped shapes' points are actually *offset* to "center around" the reference transform position, while the pivot point – and consequentially the object's transform position – remains static.

3.5.4.2.1. Ref. Object – Drag and drop the reference Game Object here. **Important:** this object must be out of the RageGroup's hierarchy.

3.5.4.3. Per-Item – This option works just like the default setting, but works independently on each spline item (listed on the RageGroup), finding each geometric center and centering relative to it. When *In-Place* is off,

each spline is centered at the RageGroup's center. This option doesn't affect the RageGroup or empty game objects in the hierarchy.

3.5.4.4. Per-Branch – Centers each hierarchy branch relative to its root element (ie. Game Objects immediately under this group). If there's a spline named "pivot" in the hierarchy, will use its geometric center to define the pivot of the entire branch hierarchy. Otherwise it just uses the average of all branch's splines geometric centers. Harder to explain in words than with an image, so check this out:



3.5.4.4.1. Delete Pivot References – Normally you'll create simple disc shapes in your illustration package, name them "pivot" and put them within the segment hierarchy (ie. Arm, foot, etc) for later usage with Pivotools Per-Branch option. After it carries out its operation, you usually won't want those temporary shapes. When the 'Delete Pivot References' toggle is on, it automatically deletes them for you.

3.5.4.5. Interactive – Allows you to place the group pivot directly in the scene view by moving a circle gizmo and hitting Enter on the keyboard.

3.5.5. ***Attention:*** To see the Pivot point within Unity you must have the centering setting (next to the Unity UI scale button) set to "pivot", not to "center".

3.6. ***Rage CanvasAlign*** – Very useful for GUI construction, CanvasAlign allows you to align any element to a corner, side or center of the display canvas (Game View), no matter how large or small its resolution. Possible elements are standard polygonal meshes, RageGroup, TextMesh, or a RageText (available with RageTools Pro). To offset your element's position relative to its "anchor", you can simply change its container area size or apply CanvasAlign to a parent GameObject and directly change the position of its child elements.

3.6.1. Container Area: To define its container area, CanvasAlign employs a standard Unity box collider. Collider boundaries can be easily and interactively

edited by holding Unity's default Shift shortcut (or enabling the "Edit Collider" button after Unity 4.6) and dragging the on-screen handles. Notice that since Unity 3.5 the box collider has to be enabled.

3.6.2. Hor – Horizontal Alignment setting. Can be Center, Left or Right. This setting can be overridden by RageText, if present in the same object.

3.6.3. Ver – Vertical Alignment setting. Can be Center, Top or Bottom. This setting can be overridden by RageText, if present in the same object.

3.6.4. Start Only – If enabled, makes CanvasAlign be only executed in the first frame. To see the alignment in editor time (before Play), this setting must be disabled. This option is ideal for mobile devices, since it's not executed every frame and there's never a resolution change.

3.6.5. Camera – Shows the scene camera to be used by CanvasAlign. By clicking the dropdown box you can select another camera with a RageCamera attached.

3.6.5.1.Update – Click this button to refresh the scene's camera list.

3.6.6. ***(new)*** Offset – Defines the padding area, offset from the anchor point defined by the alignment settings. Enable 'Start Only' temporarily to move the game object, then click the 'Rec' button to store it. Check this video: https://www.youtube.com/watch?v=HdKNxu7UOHM#t=40

3.6.7. ***(new)*** Rec – Records the current offset. Read the 'Offset' description above.

3.6.8. ***(new)*** Canvas Scaler – Assign a Canvas (Unity 4.6+ new UI system) component to align and scale the component according to its settings. Works with Screen and constant size modes. Offset is not currently 100% precise with this mode, consider leaving it at zero and using regular scene nesting instead.

3.7. ***Rage Constraint*** – Whenever you need a Game Object to "follow" another Game Object rotation, position or scale, use RageConstraint. RageConstraint is an excellent tool for building animation-focused setups commonly called "rigs", and just perfect to connect multiple RageIk chains or RageMagnet chains (Pro only) in separate hierarchies. RageConstraints are a necessity if you'll animate a RageSpline hierarchy with the Animation Window.

3.7.1. Visible – Toggles visibility of the controlled RageSpline or Group.

3.7.2. Follower – This is the Game Object that is going to follow the current one.

3.7.3. Refresh – Updates the connected components. Useful when you've added a RageSpline or Group to the Follower object after setting its reference.

3.7.4. Position – Enable this toggle to make the follower copy the transform position of the current object.

3.7.5. Rotation – Makes the follower copy the transform rotation of the current object.

3.7.6. Scale – Enable this toggle to make the follower copy the transform scale of the current object. Notice the copy is performed using local scale coordinates.

3.7.7. Local – *Position* and *Rotation* follow types use world coordinates by default. When *Local* is enabled, they use local coordinates instead. This toggle does not affect the *Scale* follow type, always performed in local coordinates.

3.8. ***Rage Handle*** – This simple component allow you to programmatically set the icon gizmo for its Game Object. The Unity editor has a standard way to do that, by simply selecting the red-green-blue box dropdown (top-left corner, right under the "Inspector" label) and picking any gizmo available in the scene. RageHandle has the advantage that you can switch those icons by setting a simple string in the component, disable its drawing programmatically and also disable all RageHandle gizmos at once. To do the global icon disabling, simply select the gizmos dropdown on the Scene view menu bar and unselect the checkbox next to "RageHandle".

3.8.1. Live – This toggle, when active, actually enables the handle icon drawing.

3.8.2. Gizmo File – Enter a string with the name of the gizmo file, extensions (.png, .jpg, etc) aren't needed. The gizmo file, to be found by Unity, must be placed in the Gizmos folder of your project's Assets folder. A number of icons are provided in the project folder.

3.9. ***RageCamera***– This component is included with RageSpline (since version 1.5, please check its documentation in ragespline.com/docs-userguide/) but has largely enhanced functionality with RageTools.

3.9.1. Initialize – When you click this button the Default Res Height (formerly called *Pixel Perfect Height*) fields will be automatically filled-in, with the current Game View's size.

3.9.2. Default Ortho Size – That's the ortho size where the current anti-aliasing and density values of your RageSplines look good to the artist/designer. Referred to by all Edgetune components in the scene, this field is only relevant if you use an orthographic camera.

3.9.3. Default Res Height – That's the Game View height, in pixels, with which the current anti-aliasing and density values of your RageSplines look good to the artist/designer. Referred to by all Edgetune components in the scene.

3.9.4. Update Threshold – Huge performance booster used mainly by Edgetune, this value defines how much a RageSpline transform has to change in size, percentually, to trigger a refresh. 0.2 means the shape scale has to change 20%, up or down, before refreshing.

3.9.5. Editor Mode Update – When enable, makes Edgetune components update passively while In edit mode. Useful when the gameobject scale is being changed indirectly while editing, usually through a controller.

4. Macros

4.1. Intro – Macros are functions or windows which automate repetitive or specific functions that would be otherwise tedious to perform manually. They can be accessed through the Unity component/RageTools/Macros menu, and in many cases require you to have a GameObject selected in the scene to work.

4.2. **RigidBody Multi-Setup:** Generally after adding colliders to your RageSpline shapes – both through RageSpline directly or by applying Physics styles in RageGroup – you'll want a quick way to add and setup multiple Rigidbodies at once. RB Multi-Setup to the rescue! Simply select its menu item, change its options (laid out just as the default Rigidbody inspector), make sure you have the desired Hierarchy's root Game Object selected and click "Process". If no filter is active (read below), the macro is gonna add Rigidbodies to all GameObjects in the hierarchy. When it finds an existing Rigidbody, updates its values to the macro's.

4.2.1. Filter Collider – will make the macro only process GameObjects that already have a collider.

4.2.2. Filter Name – will filter in only GameObjects that include this string in their names.

4.3. **RageGroup Hierarchy Update:** Works as if you had clicked "Update" on all RageGroup components found as a child of the selected object.

4.4. **RageGroup Apply Material:** Simply select a RageGroup, run this macro and assing the material field. After you click 'process' all grouped splines meshrenderer components will receive the material. Useful to quickly apply duplicate RageSplineTextured materials that take a specific texture.

4.5. **RageConstraint Create Controller:** To easily create a controller (Game Object that controls the movement, rotation or scale of another one), simply select the Game Object(s) to be controlled then run this macro. Created controllers will be parented to a '_Controllers' game object at the scene root. Specific settings (like constrain type) should be set per controller after the macro is executed. When multiple game objects are selected before running the macro, it will mirror the original hierarchy under the '_Controllers' root game object

4.6. **RageLayer Group Offset:** When you import various art pieces, or have multiple grouped shapes using the RageLayer component, you might find the need to re-arrange their overlapping or z-ordering. Use this Macro for that, simply select the shape you want to come on top of the other and set the offset to a sizable number, for example, 100. This way a layer z-order of 1 will become 101, 2 will become 102, and so on. You can also use negative numbers for that, just bear in mind that the z-order number is recommended to be always positive.

4.7. **Pivotools Clone UVs:** Usually when preparing a split-parts character you need to paint its segments, apply the texture to each part and painstakingly re-assemble the character (hierarchy, set pivot points, etc) inside Unity. *Clone Uvs* turn this into a one-step process. Simply assign the "exploded parts" object to the *Source* field, and the assembled object to the *Target* field, then click *Process* to offset the texture in the target object shapes to match the source object shapes texture coordinates. Groups can be auto-added to each object when needed (*Add Groups* checkbox), and removed after the operation (*Temporary* checkbox). *Name Cleanup* assures name matching will work, and "Auto Pivots" execute a pivotools-branch operation on both objects. You'll generally want all options checked. For detailed steps on the process, please check this two-parts tutorial video:

- Part 1: http://www.youtube.com/watch?v=6LizyRYRxg8
- Part 2: http://www.youtube.com/watch?v=2c7nWJI8Msw

4.8. **Transform Select Group:** Many times you have multiple splines in a single "group" game object, and you set your pivots and animation keyframes to this group. When you select shapes directly in the scene view, the select group macros will come in handy – especially their handy shortcuts: Alt+G to select the parent (group) of the current selection and Alt+Shift+G (Opt+G and Opt+Shift+G on OSX respectively) to select the root parent of the current selection.

5. SVG Compatibility

5.1.1. *RageTools SVG-In* was tested and proved compatible with a number of SVG shapes exported from both Adobe™ Illustrator™ and InkScape, leading players in the industry. Many of these files can be found in the *SvgFiles* project folder. Nevertheless, SVG has many features that aren't supported yet, like hollow shapes, masks and radial gradients, so *it's impossible to assure that your vector art will be imported flawlessly*. Full compliance takes time and effort both on the RageTools end and on the RageSpline's end, and exception cases will become lesser as the products develop. Follows the formal compatibility commands list - according to their default W3C specification - and some comments where appropriate:

5.1.1.1. g (Group)

5.1.1.1.1. Illustration programs' layers are regularly exported as groups.

5.1.1.2. transform – transformations (move, rotate, scale) are supported on groups and all graphical elements.

5.1.1.2.1. Transform matrix – Proportional, non-skewed transformations in matrix form, as commonly exported by Inkscape, are supported

5.1.1.3. linearGradient, radialGradient, stop – RadialGradient is converted to a linear gradient, the only type supported by RageSpline. Due to another RageSpline limitation, only two stops are supported as start and end gradient colors, additional gradient stops are ignored.

5.1.1.4. rect – fully supported, except for rounded corners

5.1.1.5. circle

5.1.1.6. line

5.1.1.7. polyline

5.1.1.8. polygon

5.1.1.9. ellipse

5.1.1.10. path

5.1.1.10.1. The following absolute and relative commands (upper or lowercase) are supported: M (move to), L (line to), H (horizontal

line to), V (vertical line to), C (cubic curve to), S (cubic smooth curve to) and Z (close shape).

5.1.1.11. Clipping Path and image import – Make sure the image is linked, not embedded, in your drawing application and the same image (PNG or JPG format) is placed anywhere in your project before import with SVG-In. There's a new 'ClipPath' demo folder within RageTools/Demo/SvgIn. These features only work if Build Settings / Platform is not set to WebPlayer. After import you can set the build platform to WebPlayer normally.

5.1.2. Not supported

5.1.2.1. Besides the exception cases noted above, SVG-In currently doesn't support:

5.1.2.1.1. Non-proportional or skewed Matrix Transformations – These operations aren't directly supported by Unity, and emulating them through nested transformations doesn't work reliably for multi-level hierarchies. To work around that, there's a way to "freeze" (ie. commit the points' screen positions) your shapes. In Inkscape, go to *File* > *Inkscape Preferences* and at the Transforms sections make sure 'Store Transformation' is set to 'Optimized'. Then just select your art and recursively ungroup everything by hitting Ctrl+Shift+G multiple times. Then, to make sure everything is flat and nice, select *Path* > *Object to Path* and save it as Flat SVG. If you use Illustrator, simply save your file as "SVG Basic" and it'll flatten all transforms.

5.1.2.1.2. Arcs

5.1.2.1.3. Vector-Vector Masks – Not supported by RageSpline. Try using Booleans to emulate its effect.

5.1.2.1.4. Use – Except for Gradient definitions

5.1.2.1.5. Pattern

5.1.2.1.6. Percentage Measures – Currently being parsed as absolute units.

5.1.2.1.7. Text – Will be supported in the future through RageText.

5.1.2.2. To circumvent most command limitations, try converting your shapes to editable shapes ("convert to curves" or "Object to Path") in your illustration package before export. If you use Inkscape, it's also good practice to select the 'plain SVG' file type in the save-as dialog box.

6. Flash Compatibility *(Deprecated in latest versions)*

6.1. RageTools 1.1.5 supports Unity's Adobe Flash build module. SVG-In doesn't support the Flash platform on build settings, make sure to switch it to PC/Mac before using it. You may switch back normally to Flash after import is done.

6.2. To build to Flash you'll first need to disable RageFarseer. Read section below.

7. Disabling RageFarseer *(Deprecated in latest versions)*

7.1. If you don't need or want Farseer Physics in your project, you can completely disable and/or remove it from your folder. To do it, simply select your scene's RageCamera and turn off the "Enable Farseer" button. This will move the Farseer folder to the project's root (above 'Assets') and switch the FarseerAdapter C# scripts. Turning it on again reverses the operation. After disabling it, feel free to completely delete the Farseer folder.

8. Conclusion

When I started working on RageTools, I was looking for ways to improve my own game-making toolset, with a bit of Flash-envy and all its resolution-independence glory, dreaming about being able to mix that with Unity's amazingly productive IDE and workflow; yet at the end of the day what I really wanted were better tools that could help me make better games with less effort, so that, as a game designer and developer, I could focus my strength on what really mattered. It didn't take me long to realize how advantageous these tools could be for every other developer – but coding for others is not as easy as coding for yourself. Things like UI readability, usability and flexibility, generally a second thought when you're your only user, suddenly becomes of utmost importance. Hopefully all the extra work and care does show. Developing RageTools in tandem with a real game (Pet Pong) has put the tool set under the "acid test" it needed, but now it's time for its real test: you! After months of working on this product, I truly hope you find it of good use for your own game, and I look forward to your feedback. Whatever makes your life easier will make mine easier too, so fire that feedback away! Last but not least, a heartfelt thank-you to my wife Márcia for her enormous patience with those insane late night working hours, to the community for the continuous "where's my SVG importer?" pushes and Juha Kiili for his "what the hell are you doing?" friendly yet even harder pushes. Huge thanks to all of you for the support, you're all responsible for RageTools existing.

- Breno Azevedo