Full Circle

INKSCAPE SERIES SPECIAL EDITION

THE INDEPENDENT MAGAZINE FOR THE UBUNTU LINUX COMMUNITY

INKSCAPE SERIES SPECIAL EDITION Vol 10

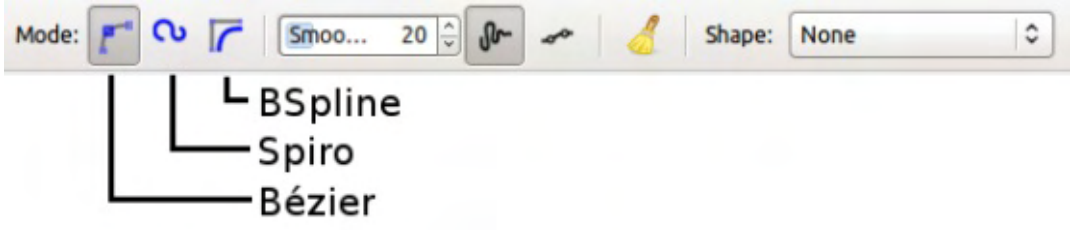
INKSCAPE

Volume Ten **Parts 65 - 72**

Inkscape - Part 65

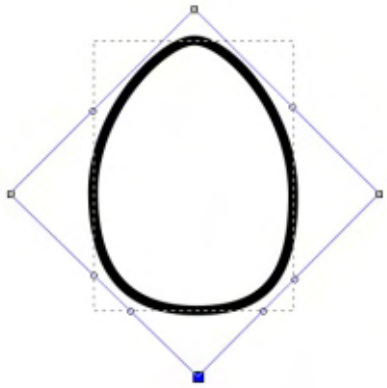
In parts 42 to 47 of this series, I described, in some detail, each of the "Live Path Effects" in Inkscape at the time, culminating with the new LPEs added in 0.91. Since then, the release of 0.92 has seen the addition of many new effects, so I'll spend the next few articles introducing each of them. I will assume, however, that you're already familiar with the idea of LPEs, as well as how to add and remove them. There are also a few user interface conventions that are common to many of the effects, which I'll also assume familiarity with. If necessary, you might want to re-read the earlier parts of the series to remind yourself of the details.

To begin with, we'll take a look at a few LPEs that are also exposed through other parts of the Inkscape interface – the "convenience" effects that are bound to toolbar buttons in the Pencil (Freehand) and Bézier tools. Both tools previously offered the ability to create either regular Bézier paths, or to use the "Spiro" algorithm to easily create smoothly curving shapes. To this roster has been added a third option: BSpline paths.

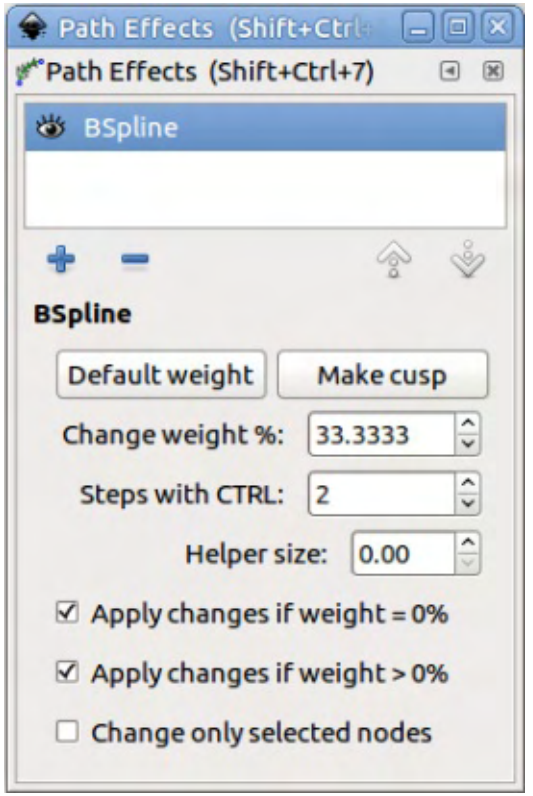


Like Spiro shapes, BSpline paths are also smooth, with seamless transitions from one segment to the next. The difference, to the end user, is that Spiro paths are made of circular arcs defined by points that lay on the path itself; when you use the Node tool to edit the path, you simply change the position of the ends of each segment. BSpline paths, on the other hand, are defined by points that "pull" the path in their direction, with handles to set the "weight" or strength of that pull. This allows for tight, asymmetric curves that aren't possible with Spiro paths.

Despite initial expectations, the handles that set the weight of each node can't simply be dragged around – you have to also hold down the Shift key, for reasons that presumably made sense to the developer. In this screenshot, you can see a circular path that's had the BSpline LPE applied in order to distort it into an egg shape. The general shape is set by the square node handles, but moving the circular "weight" handles has allowed the top to become more pointed, and the bottom more rounded (for reference, the equivalent shape using a Spiro path required twice as many nodes).



If you implicitly add the BSpline LPE by enabling that mode in the Pencil or Bézier tools, you have extremely limited control over the effect. In the Pencil tool, the value of the smoothing field will have an impact on the number of nodes that are placed – set this too high and you'll get very few nodes, with a BSpline that doesn't really reflect what you've drawn. In the Bézier tool, the only choice you have is to



either click to place a normal node, or Shift-click to place a cusp, or corner node. The latter has a weight of zero, allowing for sharp transitions in your otherwise smooth path. If, however, you explicitly add the BSpline LPE to a path – or if you open the Live Path Effect dialog for a path that's had it implicitly added – you'll get a few options in the UI to let you tweak your shape.

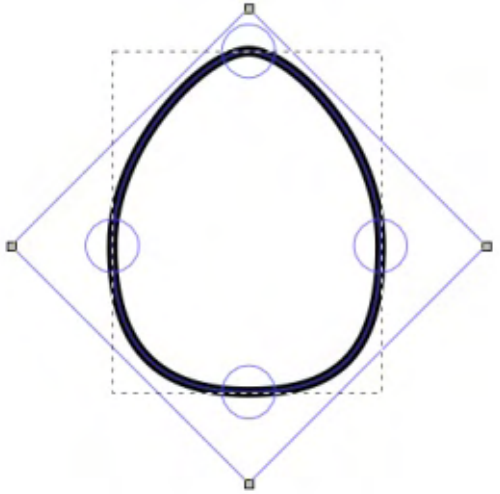
As is often the case with the Path Effects dialog, the labelling and arrangement of the controls could definitely benefit from some improvement. For a start, the first thing you need to interact with is the last section of the dialog – the three checkboxes at the bottom. These dictate which of your nodes will be affected by changes to the upper half, and failing to set these correctly can result in too many, or too few, of your nodes being changed.

If the first checkbox is selected, any changes will affect only nodes with a weight of 0%. A better way of saying this is that the changes will affect only cusp nodes. The second checkbox does the same but for non-cusp nodes. Checking both allows changes to all nodes; checking neither will stop your changes affecting anything at all. The third checkbox further limits changes to only those nodes that are selected. The effects are cumulative, so if the just the first and third boxes are ticked, your changes will affect only the selected cusp nodes, and will not affect any non-cusp nodes, even if they're also selected.

Moving to the top half, the Default Weight button sets the nodes to a weight of 33.333%. In other words, it positions the node handles a third of the way from one node to the next. Make Cusp, as you might imagine, turns the nodes into cusp nodes. It obviously makes sense to use this only if the second checkbox is also ticked.

Change Weight % lets you adjust the weight of all the targeted nodes. It should probably be labelled "Set Weight", as putting a value in here sets an absolute value, not a relative one. For example, putting 25 in will set the handles to a quarter of the distance between the node and its neighbours – it won't adjust their existing positions by 25%.

There is a convenience feature when changing the positions of the handles interactively on the canvas: if you hold down Ctrl rather than Shift, you can set the handles to pre-set positions. By default, these are at 0% (cusp node), 33%, 66% and 100%. Given that the two extremes are always possible, the "Steps with CTRL" option lets you define how many intermediate steps are available. Set this to 3, for example, to get steps at 25%, 50% and 75%. As for the "Helper size" control, your guess is as good as mine! As far as I can tell, it simply draws some circles on your path of the specified size. How these are meant to help you, I don't know.



In practical terms, the controls available through the LPE dialog are of questionable benefit. For artistic purposes, you're more likely to just adjust the node positions and weight handles on the canvas until your path is visually of the right shape. Personally, I find the BSpline option on the Pencil tool to be of little use, as it's tricky to get the smoothing to just the right balance between too little effect and too much. On the Bézier tool, however, it makes much more sense. You can click-click-click to define the shape of your object, and an outline of the BSpline path is shown interactively as you do so, letting you more clearly see what the result will look like when you finish. With 0.92, this interactivity also extends to the Spiro option on the Bézier tool, which makes this mode much easier to work with too.

The Pencil tool has acquired another LPE shortcut, which is potentially far more useful: Smoothing. When drawing with the Pencil tool, Inkscape will tend to create lots of nodes, faithfully reproducing every jitter and bump your hand makes as you move the mouse or stylus around. Smoothing attempts to compensate for this by averaging

out your movements to create a smoother – arguably more "vectorish" – path. By adjusting the amount of smoothing applied, you can find the right trade-off between faithful reproduction and oversimplification.

The trouble with this process in the past was that smoothing was applied only at the point of recording your movements. You would set the smoothing control and Inkscape would average your movements as you went along, completely replacing the original position data with its own calculated equivalents. If you set the smoothing control too high, there was no way to reduce it afterwards and regain some of your finer details.

Looking back to the toolbar image at the start of this article, you can see the smoothing slider after the BSpline button (note to the Inkscape developers: make this control a bit wider, and you could populate it with something more useful than "Smoo…"). Immediately after that is a new button that toggles between the previous smoothing method, and the new LPE-based approach. If the LPE Smoothing button is selected, an additional new button appears besides it, as shown in the screenshot, which looks like a pair of wonky spectacles.

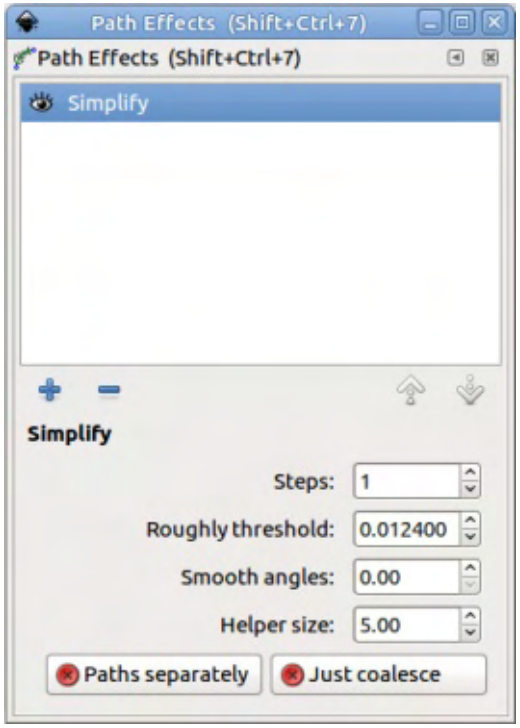
With the LPE Smoothing button toggled off, the Pencil tool will behave as it always has (see part 18 of this series for a little more detail). In this mode, you set a smoothing value, then draw something and see it converted into a simplified version of your original path. Select it and check the status bar to see how many nodes it has: high levels of smoothing can produce paths with drastically reduced numbers. With the path selected, switch back to the Pencil tool and try modifying the smoothing value. You should see no change to your path, or the number of nodes it has – all you're doing is changing the smoothness value that will be used for the next path you draw.

Now toggle the LPE button on, and repeat the exercise. As you draw, you'll see the original rough path, which then gets converted into a smoother version when you finish. Switch to the selection tool to check the number of nodes; again you'll see a low number, but this time it also says "path effect: Simplify", confirming that an LPE is in play. Here's the big difference though: return to the Pencil tool and change the smoothness value. Your path should move from rough to smooth and back in real-time as you drag the slider around. Did you set the value too high or low when drawing the path? Just change the value afterwards!

To achieve this feat, Inkscape actually stores the original path, as though you had set smoothness to its lowest value. It then automatically applies the Simplify LPE, which does the same job as the Path > Simplify option, except that it does so as a live (and reversible) process.

There's no doubt that the added flexibility of the Simplify LPE can be a huge advantage, so why would you ever want to turn it off? Well, as you might expect, such flexibility comes with a cost: in this case, it's the cost of processing the LPE every time the path is rendered or changed. For one or two paths it might not have much of an impact, but, as with filters, too many live effects can rapidly drag your PC to a crawl. There is, however, a compromise in the form of the wonky glasses button. Clicking this will "flatten" the path by replacing the original shape with the output from the Simplify LPE, then removing the effect completely. This fixes the smoothness so you can no longer alter it by adjusting the slider – but it also means that Inkscape no longer has to calculate it in real-time.

If you open the Live Path Effects dialog, you'll find that the Simplify LPE has a few controls to let you tweak the effect.



The Steps value sets the number of times that the Simplify algorithm should be applied – it's the live equivalent of selecting Path > Simplify multiple times in succession. The bewilderingly named 'Roughly Threshold' parameter is the value that gets changed when you move the smoothness slider in the Pencil tool's controls. This sets the strength of each pass of the algorithm and is what primarily defines the amount of simplification that is performed. Unfortunately, it seems to operate over a peculiar range of numbers: it is rather non-linear, with the most effect occurring at really rather small values. Fractions down to hundredths or thousandths of a unit aren't uncommon. This control would probably have been better as a slider, mirroring the smoothness control in the Pencil tool, and operating over a normalized range of values. As it stands, I recommend starting as low as you can (enter zero and Inkscape will set it to the smallest allowed value), then use your mouse-wheel over the control to adjust the value until you have the appearance you want.

The Smooth Angles control allows you to affect how the algorithm deals with cusp nodes. With this set at its maximum of 360, all nodes will be smoothed. By reducing this value, you can set a cut-off for the angle between the handles of a cusp node, above which the algorithm won't change the node. Consider a right-angle in your path: by setting this value to 80, the right-angled cusp (where the handles make an angle of 90°) won't be smoothed, and you'll have a nice sharp corner. Set it to 90 or more, however, and the right-angle will be smoothed, softening your shape. If you have tight angles in your path that you want to keep sharp, play with this parameter. It doesn't seem to work so well on acute angles, though.

Next is another Helper Size control. As with the one in the BSpline UI, this causes circles (and in this case also squares) to be drawn on the path – but I still have no idea how that's supposed to help you! The Just Coalesce button below it is also a mystery to me: it seems to have some effect on the shape of the final path, but the details of what it's doing are less than obvious.

That just leaves the Paths Separately button – for which I'm going to hazard a guess as to the function, based purely on the name. The Simplify LPE may be applied to a complex path – one made up of multiple sub-paths – or even to a group of path objects. Toggling this button on seems to apply the effect individually to each sub-path, or each path in a group. Leaving it off applies the effect to all of the paths or sub-paths as one. The distinction – and the effect it will generally have – is a subtle one, and probably not worth further consideration for most users.

This approach of exposing LPEs as buttons and options in other tools is, I think, a good one. It opens up these advanced capabilities to users who might otherwise be put off by the sometimes complex appearance of the Live Path Effects dialog, whilst still allowing advanced users access to the (sometimes bewildering) controls. But this article has barely touched on the LPEs added in 0.92, and the others aren't so easily accessed. Next month, we'll be taking a look at some of the other new effects, so if you do need a refresher on using LPEs, now's the time to start reading those back issues of FCM that are sitting on your hard drive.

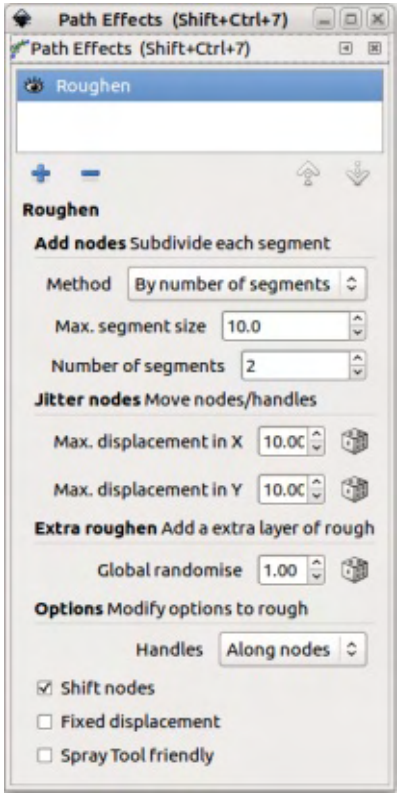
Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, Inked' and 'Elvie', which can all be found at http://www.peppertop.com/

Inkscape - Part 66

Last month's look at the Live Path Effects added in 0.92 included a section on the Simplify effect. This adds a "live" version of one of Inkscape's existing tools (Path > Simplify), whereas the older tool modifies the original path. The next LPE we'll look at fills a similar niche – it's another live version of an existing feature. It's the "Roughen" LPE.

The go-to tool for adding some randomness to the shape of a path is the Tweak tool, described in detail in parts 22 and 23. The latter article describes using the tool to modify the nodes of an existing path, including the ability to roughen the path by adding new nodes and slightly randomizing their positions. As with Path > Simplify, the procedure modifies the original path, so there's no way to subsequently tweak the settings to retrospectively alter the result. The Roughen LPE does the same job, but, being a Live Path Effect, allows you the flexibility to go back and change its parameters after the fact. One trade-off for this capability is that the effect applies to the whole of the path, whereas the Tweak tool is interactively "sprayed" onto the path, allowing you to confine its effects to a particular area, should you need to. But, if you need to work on an entire path, it's well worth further investigation.

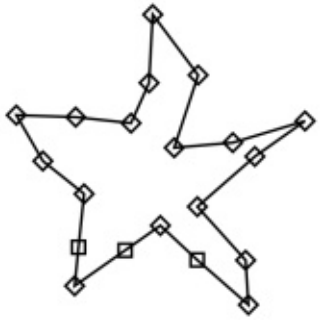
To demonstrate this LPE, I've created a simple five-pointed star using the Stars and Polygons tool,



then added the effect via the Path > Path Effects dialog. As you might expect, the dialog gains a number of parameters that can be adjusted to alter the result.

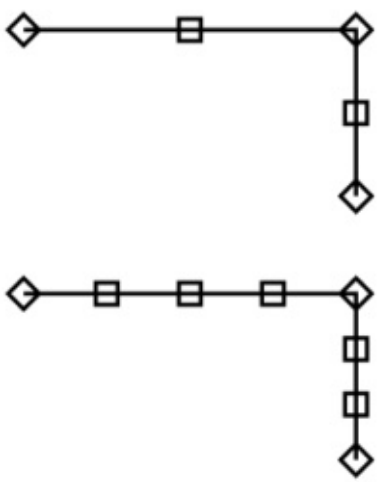
With the settings shown in the screenshot, my simple star was immediately distorted into something more random.

To get a better idea of what's happening, I'm going to add another new LPE to the chain: Show Handles. This draws representations of the path's nodes and handles, which can be invaluable when trying to gain an understanding of exactly how your input path has been changed by the applied LPEs (more on this later). The first thing to notice is that our five-pointed star (which, if converted to a path, would have had 10 nodes) now has 20 nodes:



This is due to the top section of the Roughen effect's controls which adds extra nodes. More specifically, it adds nodes to each segment of your path, spacing them evenly along it. The number of nodes added is either determined directly by choosing the "By number of segments" option and setting a value in the "Number of segments" field; or indirectly by choosing the "By max segment size" option, and filling a value in the "Max segment size" field. The difference is clearer when you have a path with different segment lengths; consider this example of a right-angled path where the horizontal arm is about twice as long as the

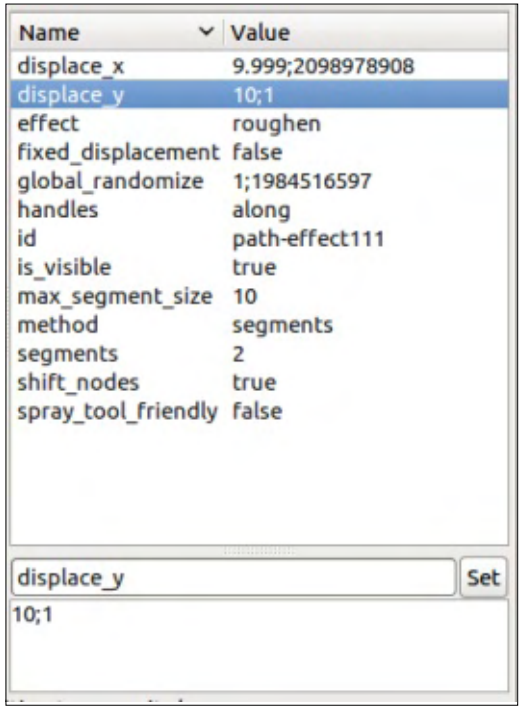
vertical one.



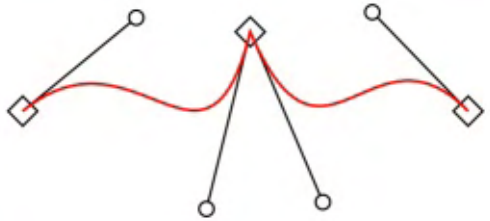
The top image shows the effect of using "By number of segments" to divide each path segment into two. The bottom picture uses "By max segment size", resulting in two new nodes on the vertical arm, but three new ones on the longer horizontal arm. Returning to our star, therefore, the default settings split each segment into two, doubling the number of nodes.

Once you've created more nodes to work with, it's time to jiggle their positions a bit. The "Jitter nodes" section lets you define the maximum amount that each node will be displaced – although the precise value for each is random. You can set different values for the X and Y directions, and the dice buttons will re-seed the random number generator, adjusting the node positions in either the X or Y directions accordingly. The "Extra roughen" section provides an additional displacement factor. This value acts as a multiplier – setting it to zero will cancel any displacement, regardless of how large the X and Y values are, whilst larger numbers increase the amount of displacement that takes place. This time, the die button re-randomizes the nodes' positions in both the X and Y directions at once.

It's worth noting that the random seed for each of these controls defaults to 1 when the LPE is first added to a path. If you have several similar paths that you wish to roughen differently, therefore, it's worth clicking each of these buttons a few times. The actual seed isn't visible in the UI, but can be found via the XML editor. The settings for LPEs are stored in the <defs> section near the top of the file, and if you've applied several to your page, you might need a little trial-and-error to find the relevant entry (tip: put an easy to spot value, such as 9.999 into one of the fields so that it stands out more in the XML dialog). Here you can see that I've clicked the randomize buttons for the X displacement and the global randomization, but left the highlighted Y displacement with its default value. The seed in each case is the part after the semicolon, which you can copy and paste if you wish to use the same non-default seed across multiple objects.

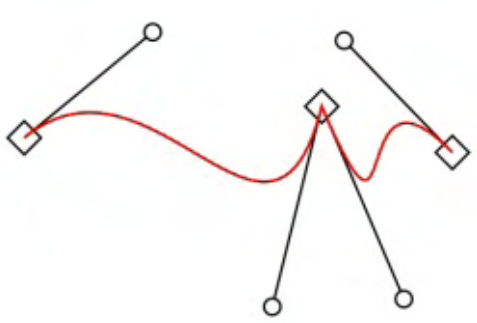


To roughen a path, inserting new nodes and randomizing their positions may be sufficient. But this effect also provides a few options about what to do with the node handles. Our pointed star, with its straight line segments, offers little of interest when it comes to node handles, so, to demonstrate the possibilities, I'll switch to a curved shape with a sharp transition in the middle. Once again, I've applied the Show Handles LPE, but I've put a duplicate of the original path on top (in red) for clarity. Here's how it looks before the Roughen LPE is applied.



To isolate the effects on the handles, without adding extra nodes to complicate matters, I've used the "By number of segments" mode, with that number reduced to 1. This effectively neuters the top part of the dialog, and no new nodes are added, though the existing ones will still be moved. The "Handles" pop-up towards the bottom of the dialog determines what will happen to the nodes' handles. With the default setting

of “Along nodes”, the handles simply move along [with the] nodes, maintaining their relative sizes and positions.



The “Rand” option randomises the position of the handles. Unfortunately, there’s no button to set a random seed, and no seed value appears in the XML file, so I guess you’re stuck with whatever random positions the LPE gives you.

“Retract”, as its name suggests, retracts the handles completely, converting your path segments into straight lines, whilst “Smooth” ensures that the handles on either side of each node form a straight line, with the result that the path transitions smoothly from one segment to the next, even if doing so drastically changes the shape of your path. Adding extra nodes using the top section can help to reduce the amount of distortion that takes place.

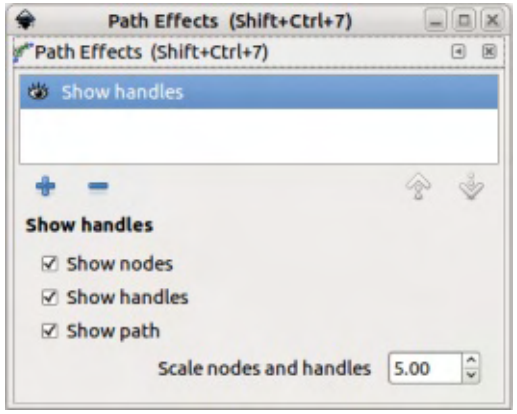
At the bottom of the dialog are three checkboxes, starting with “Shift Nodes”. With this unchecked, the nodes won’t move, regardless of the X, Y and Global settings. At first, this seems to rather defeat the point of the dialog, but it opens up a couple of possibilities that aren’t immediately obvious. With this unchecked, you can use the top section of this LPE as a means to subdivide your path into smaller sections without affecting its shape. This might be handy as a “pre-processing” step before sending the result to another effect. Another option is to uncheck this but to change the “Handles” pop-up, so that the nodes don’t move but the handles are still randomised, retracted or smoothed.

The second checkbox seems less useful to me. It fixes the amount of allowed displacement to 1/3 of the length of the line segment, regardless of the X, Y and Global values. Why this should be a particularly good idea escapes me. I can understand that there might sometimes be a benefit to having the amount of displacement related to the length of the path segment, but, in that case, I would prefer to have a control to set that factor, rather than have it hard-coded as 1/3 of the segment size.

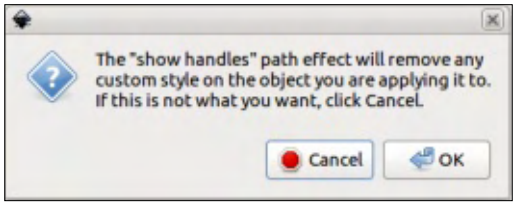
The last checkbox, “Spray Tool friendly”, is a mystery to me. The tooltip suggests it’s “for use with the spray tool in copy mode”, but my own experimentation of using the Spray tool on roughened shapes suggests that it has no obviously useful effect on the result. With this checked, some of my “copies” were slightly distorted compared to their peers, but not by enough to recommend this as a way of producing randomized copies.

So far, I’ve concentrated on using this effect to produce small changes to a path. In practice, an LPE called “Roughen” might be expected to have more dramatic effects in most situations. Returning to my original star shape, increasing the number of path segments, adjusting the X, Y and Global values, and randomizing the handles, produces arguably the output most people would expect from this effect.

Having briefly introduced the Show Handles LPE earlier on, I’ll finish this month by delving into a little more detail. The UI for this effect is so straightforward that it barely warrants a mention: the three checkboxes toggle the visibility of nodes, handles, and the path itself, whilst the spinbox lets you set the size of the rendered nodes and handles.



Compared with many LPEs, this UI is a model of simplicity. But what the UI doesn't reveal is that there is a major problem with this effect which you really need to be aware of before using it: it completely wipes out any fill or stroke styles you might have applied to the original path. If your final aim is to render the nodes and handles – as in the images in this article – that's perhaps not so much of a concern. But if you merely want to temporarily see what your chain of effects has done to the path, be aware that turning this LPE off, or even removing it entirely, won't reinstate your original style settings. To be fair, the first time you try to add this effect in each session you are presented with the chance to back-out:



If you press ahead, you'll find that your path is reduced to a thin, black stroke, with no fill. You can subsequently set a fill or change the stroke, but remember that the output from any LPE is itself a single path (albeit one with sub-paths, in this case), so you can apply only one set of styles to the entire output. In other words, you can't color the path differently to the handles or nodes – not without using multiple copies or clones of the path, at least.

What to do, then, if you do want to use this effect without altering the style of your original path? If you just want to view the results temporarily, perhaps duplicating the original and applying the LPE to the new copy would be sufficient. But if you want the results to hang around a little longer, and therefore stay in sync with any changes to the original, you will have to work on a clone. That might sound simple, but making clones work with LPEs isn't without its problems.

The obvious approach is just to clone the original (select it and press Alt-D). With the clone selected, opening the Live Path Effects dialog will show a message at the bottom saying "Click add button to convert clone" or similar. As soon as you click the "+" button in the dialog to add a new LPE, you'll find that a "Fill between many" effect is automatically added, and your clone's fill and stroke become unset. I won't go into the details of this effect now (but its appearance in this role has promoted it to the subject of next month's column), but suffice to say that it offers one way to link an existing path into a new LPE chain. You can go ahead and add other effects if you wish – including "Show handles" – but as soon as you try to move the clone to another location, you'll have problems. It tends to jump back to the position of the original path and although there are ways to persuade it to sit elsewhere, the slightest nudge will send it scurrying back to its parent again. I note in a related bug report that the main developer of LPEs has recently committed some improvements to this effect into the Inkscape trunk, so hopefully we'll see this addressed in the next release.

In the meantime, you can use the "Clone original path" LPE that was described in Part 47. In short, the steps you need to perform are:

- Select your original path and copy it to the clipboard.
- Create a sacrificial path to attach the new LPE to. A simple two-node line will do.
- Add the "Clone original path" LPE to the sacrificial path.
- Click the "Link to path" button in the LPE dialog (the first of the two buttons in the effect's UI).
- You can freely drag this clone wherever you need to on the page.
- Add the "Show handles" LPE to the chain (or, indeed, other LPEs if you wish).

Part 47 also describes a shortcut, using Edit > Clone > Clone Original Path (LPE), but that now also applies the "Fill between many" effect, so, until the issues with it are ironed out, it's probably best to stick with the steps above.

Next month, I'll continue looking at the new LPEs in 0.92, including a closer look at the "Fill between many" effect, and what it can more usefully be used for.

Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, Inked' and 'Elvie', which can all be found at http://www.peppertop.com/

Inkscape - Part 67

In researching last month's article, I found that Inkscape's default behaviour when adding an LPE to a clone, or when using the Edit > Clone > Clone Original Path (LPE) option, has changed from adding the "Clone original path" effect, to using the "Fill between many" effect instead. On the surface, this brings an obvious regression: it's no longer possible to move the clone independently of the original object. So why was this change made? In order to explain that, you have to understand what this LPE does in normal day-to-day use.

Something that is often misunderstood about LPEs (though hopefully not by readers of this column, as I've pointed it out previously) is that the output from an LPE is just a single path. It might be a complex path, containing sub-paths, but it's still a single path from the perspective of styling. It can have only a single stroke color and style, and a single fill. When you use something like the Power Stroke LPE (see part 47), or the new Taper Stroke LPE (see below), it doesn't somehow imbue Inkscape with the magical ability to create variable width strokes (a feature that the underlying SVG 1.x format doesn't allow). Rather, it creates a filled shape that happens to follow your original path, and gives the appearance of being a variable width stroke. But in taking this approach, the ability to fill the original shape is lost – any fill you apply now applies to the stroke-like path that is output from the LPE.

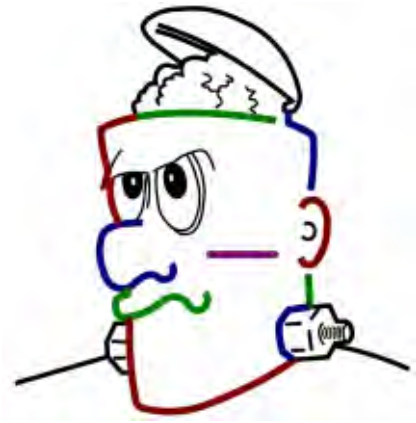
This is a particular problem for things like cartoons and comics, where an artistically varying stroke is a necessity, but you also want to fill areas with color. So the Clone Original Path LPE (also covered in part 47) was born, which at last allowed you to apply one LPE to your stroke, but then also clone the original shape via a sacrificial path with the LPE applied which could be independently filled. But it works with only one path at a time. And cartoon characters are often made of more than one – sometimes with artistically placed gaps in the outlines. Cue the return of "Frankie", a character from my "Monsters, Inked" comics drawn by my co-creator, Vincent Mealing. This time, I'm presenting him having already manually traced the hand-drawn lines, and with some artistic gaps inserted for demonstrative purposes:

Trying to fill a shape like this is a problem. It can be done manually by creating a separate object for the fill and adjusting its shape to suit. The Bucket Tool can help with this, but those gaps need careful plugging with temporary objects first. In either case, a tweak to the outline shape wouldn't be automatically reflected in the fill. For that, we need a clone of some sort, but that interferes with our ability to add LPEs to vary the stroke width. The Clone Original Path LPE can't help us either, as the shape we want to fill (just the face, for now) is clearly made up of more than one path. And this is precisely where the Fill Between Many effect comes to the fore.

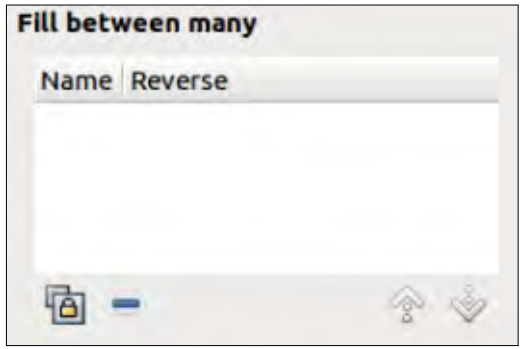
Fill Between Many allows you to collect together several paths into a single LPE. The output is the result of creating a path that joins all of the constituent paths together into a single shape, by implicitly linking the end of each one to the next in the list, and the end of the last path to the start of the first. A trivial example should get the idea across:

On the left are three paths – straight in this case for clarity, though curved works just as well. On the right is the result of creating a sacrificial path, adding the LPE to it, then adding each of the three paths. The result of the LPE was filled with orange, and had its stroke removed. It also automatically updates to match any changes that are made to the shapes of the original paths. Hopefully it's pretty clear that this new shape is the result of drawing a path that follows each of its component parts, and joins their ends together with straight lines.

Now that you understand the basic idea behind this effect, let's go through a more detailed example by coloring Frankie. For clarity, I'll start by changing the color and thickness of each path we'll be adding to the LPE:



You may have noticed the purple scar on his face. Don't worry, he hasn't been in a fight. That line is the sacrificial path that the LPE will be added to, and it'll disappear in just a moment. Selecting that path, and adding the Fill Between Many effect, results in this trivial UI:



Now the hard work begins. We have to copy each of the source paths to the clipboard in order to add a link to it in this dialog using the button at the bottom left. Common sense would suggest that you could select all the paths and add them as a single operation, but, unfortunately, common sense would be wrong in this case. Instead, you have to add them one-by-one. To make matters worse, each time you select a path to copy, the sacrificial path becomes de-selected, so the UI above vanishes. You then need to re-select the sacrificial path before you can add the copied link. It's not difficult, but it is time consuming. So, to add our first path (the top left of the face), here are the steps:

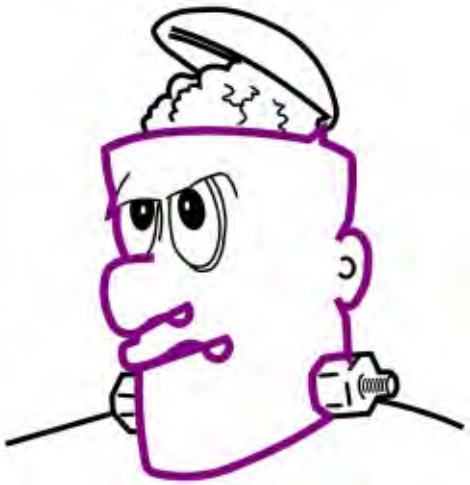
1) Create a sacrificial path, if you haven't already done so, and add the LPE to it.
2) Select the first path to add and copy it to the clipboard.
3) Re-select the sacrificial path.
4) Click the button in the bottom left of the LPE dialog to add the link.

Do all that correctly and you'll end up with something like this.

There are a few important things to note at this stage. First you can see that the sacrificial path has disappeared, replaced with the output of the LPE. This takes the same shape as the attached path, but with the ends connected. It also adopts the style of the sacrificial path. If your first path is straight, it can be easy to lose the LPE output on top of it, so I strongly recommend using a contrasting style for your sacrificial path for this stage of the process. You'll also notice that the path is listed in the LPE dialog, as "path918" in this instance. That's fine for shapes with only a handful of paths, but you can quickly lose track of which is which as the complexity grows. This name is taken from the path's label, which can be set via the Object Properties dialog on the object's context menu. As will become clear, you may need to know which path is which, so setting labels can make life a lot easier – albeit at the expense of more work initially. Be aware that the list in the LPE dialog doesn't update dynamically, so you have to set up the labels first. Repeating steps 2-4 for a few more paths gets us to this stage:

Things were going quite well, with the new path taking on the outline of Frankie's head, right up until the point where I added the ear. When this LPE connects between the ends of paths it doesn't join to the closest end, but rather from the end of one path to the start of the next. In this case, the ear path had been created "in reverse", with the start at the bottom and the end at the top, resulting in the twist in the output that you can see in the image. Fortunately, there are a couple of easy solutions to this issue: you can simply reverse the direction of the source path using Path > Reverse or, better still in most cases, you leave the source path alone and reverse it within the LPE by checking the Reverse box in the effect dialog for the appropriate path. Now you can see why it's useful to have proper names for your paths.

Whilst we're on the subject of editing the output path in this way, it's worth noting that the dialog also has buttons for removing a selected path from the list (multi-selection is not allowed), and for moving the selected path up or down, thus changing the order in which the paths are joined to generate the output. Continuing to add the remaining paths, and reversing a few along the way, produces this result:



As you can see, the result isn't perfect. The fact that the ends of the nose and mouth extend inside the outline results in a shape with crossed lines, and reversing these paths only makes things worse. Breaking the nose and mouth paths into smaller pieces would have allowed me to produce the desired result but, once filled, this shape only actually leaves a small gap at the end of the mouth that can be visually patched up with another path, which I consider to be an acceptable compromise. So now I can fill the resultant path, lose its outline, send it to the back, and add in a small patch to cover the hole. And revert the original paths back to thinner black lines as well.

It's worth pausing at this point to really appreciate what we've just achieved. Although the end result appears to be similar to that produced by manually drawing an outline, or filling some gaps then using the Bucket Fill tool, there is one substantial difference: this shape is live. A change to Frankie's nose, or the curve of his ear, is accompanied by an instant change in the fill shape. If we need to make more radical changes, it's possible to add, remove and reverse paths accordingly. And ultimately that is the reason why this effect is now used by default when adding an LPE to a clone.

Unlike the Clone Path effect, the Fill Between Many LPEs is more flexible, but still serves a similar purpose if it is used on a single path. Unfortunately, as discussed last time, the inability to move the resultant paths does make it useless for some applications, so there's definitely still a place for Clone Path in your toolbox.

To finish Frankie in style, we can apply yet other LPEs to the original lines: Power Stroke is a classic option for cartoon-style pieces that lets us change the stroke width arbitrarily, but, most of the time, just a little tapering in and out of the width is all that's required. For that, the new Taper Stroke effect is a far simpler option.

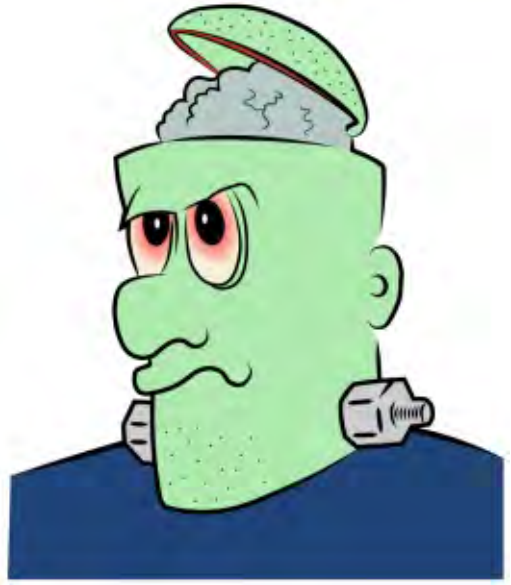
Rather than fiddling around with additional node types on the canvas (the approach taken by the Power Stroke effect), this LPE simply lets you set the stroke

width for the non-tapered part of the line, and two offsets representing the distance from each end that the stroke reaches that width. Set an offset to zero and that end gets a normal square cap with no tapering; otherwise larger numbers generally mean longer tapers.

Now there are a couple of caveats here: firstly the taper can only progress as far as the first cusp node; secondly, I have no idea what sort of units are used for the offsets. In practice I tend to just roll my mouse wheel over each field and watch the path on the canvas to set a value that gives the artistic result I'm looking for. These values can also be set by switching to the Node tool, and moving the on-canvas handles, if you prefer to work that way.

The Taper Smoothing control effectively sets the shape of both tapers, but can give some odd results at extreme values. I tend to stick to about 0.5, which gives a reasonably linear taper that suits the comic style I'm working with. The Join Type and Mitre Limit controls have the same effect as those in the Fill and Stroke dialog, except that the LPE also offers an "Extrapolated" join type which can be better for some particularly tight angles.

I'll be honest, some of these new LPEs are still a little buggy at times. I found that to be the case particularly when trying to combine the Fill Between Many and Taper Stroke effects in order to complete my Frankie trace. In the end, I had to resort to some manually drawn paths, or "fixing" some of the LPEs by using Path > Object to Path, in order to get the results I expected. Still, even with those compromises, this take on Frankie is one of the best yet.

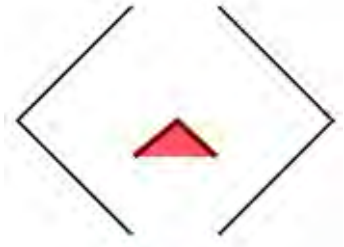


Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, Inked' and 'Elvie', which can all be found at http://www.peppertop.com/

Inkscape - Part 68

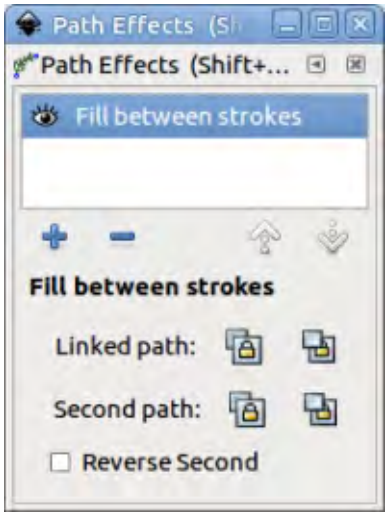
Last time I looked at the "Fill Between Many" live path effect, so it makes sense to start this instalment with the closely related "Fill Between Strokes" LPE. I'll race through this one quickly as, quite honestly, I can see very little benefit in using this effect over its more capable sibling.

Whereas "Fill Between Many" allows you to create a new shape that connects numerous individual paths, "Fill Between Strokes" does the same job, but is limited to a single pair of paths. Once again we also need a sacrificial path to which the effect will be applied (and which will be the source of the style for the resultant object).

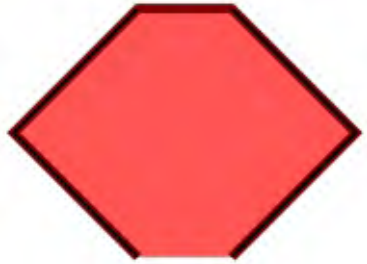


To demonstrate, my test image consists of a couple of black paths, plus a thick red path with a light red fill to use as the sacrificial path. I could have used a simple straight two-node path for the latter, but, by giving it a bit more shape, it's obvious that there's a fill applied.

The first step is, of course, to select the sacrificial path and apply the LPE. This results in a simple UI in the Live Path Effects dialog.



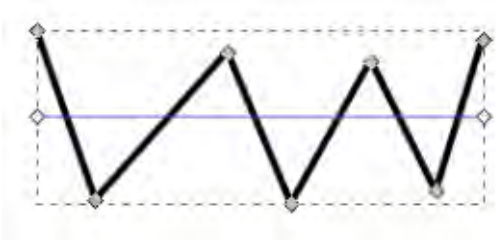
Now we have merely to add the two paths we wish to use. As usual when linking a path in an LPE this is a little contrived; you first have to select the path and copy it to the clipboard, but that de-selects the sacrificial path and you lose the UI in the dialog. Re-select the sacrificial path, then use the first button on the "Linked Path" line to "paste" the link to the path into the effect. Do the same for the second path as well. If necessary, you can reverse the second path using the checkbox, but you can't reverse the first path other than by changing it directly in your drawing. With those steps performed, the result looks like this.



This is almost identical to using "Fill Between Many" with only two paths, except that the resultant path isn't closed – you can see that the thick red stroke doesn't continue along the bottom of the shape. Given that this appears to be the only real difference between the two effects, my general recommendation is to just stick with the more versatile "Fill Between Many".

Having largely dismissed one not-so-useful effect, let's take a look at a couple of others. The first is "Transform By 2 Points". I'm sure most readers are familiar with the "pinch to zoom" effect on smartphones and tablets – and the extended version that includes rotation and panning which is often used in mapping applications. This LPE essentially brings that same functionality to bear on Inkscape paths. On adding this effect to a path, you'll be presented with a complex looking interface:

This is another effect where the UI seems to be upside down – all the interesting things are in the bottom section. The buttons at the bottom act as a set of toggles, enabling and disabling different types of transformation. Begin by activating the "Elastic" and "From original width" buttons. If you now switch to the Node tool (F2) you should see that a thin blue line with diamond handles has appeared on your path.

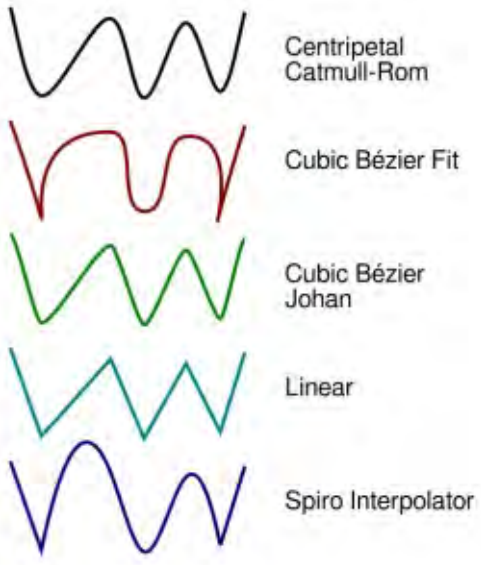


Drag either of the handles around on the canvas and you'll see your path dynamically stretch and rotate in a very intuitive manner. Leave the "From original width" button enabled but try toggling some of the others on and off to see what effect they each have. It should be clear that this is a very fast and natural way to transform a path so that it fits in a specific gap, or connects visually to other elements in your drawing.

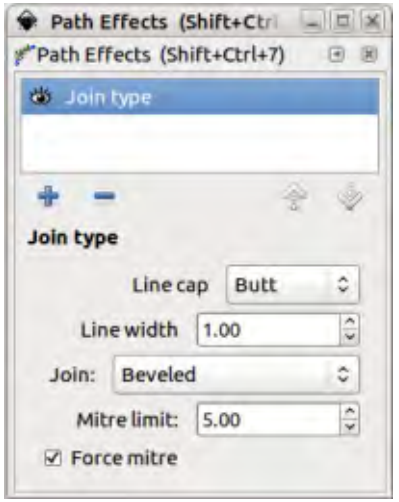
If you turn off the "From original width" toggle you can be even more specific about the path placement. Now the "First Knot" and "Last Knot" fields come into play, by letting you specify nodes in your path that should be used as the start and end of the transformation line. If you need to scale an arbitrary path so that its ends lie at specific points, this will do the trick nicely.

I quite like this effect, but I find it disappointing because it's just that – an effect. It would be so much better to see a transform option like this make its way into the roster of top-level tools in Inkscape. That would allow you to use the same sort of intuitive control whether you're transforming paths, shapes, groups or even text.

Next in the roster of effects is "Interpolate Points". This simply redraws your path using one of five pre-defined interpolations between the nodes. The UI is so minimal that I'm not even going to bother with a screenshot – it's just a pop-up menu to let you select the type of interpolation you want to use. There's no way to change the Bézier handles or otherwise adjust the curves, and the selected interpolation is applied to all the path segments; there's no way to have different values applied to each individual segment or even sub-path. This effect is probably of most use as part of a chain with other LPEs, when you want to change the output of a previous step from straight lines to curves, or vice-versa – but otherwise there's rarely any good reason to use this effect in preference to modifying your original path. Here's an example of a simple path rendered using each of the five options:



The last effect I'll be covering this month is "Join Type". This lets you set the type of join used between path segments, in much the same manner as the Stroke Style tab of the Fill & Stroke dialog – except that it does have a couple of new tricks up its sleeve, and a big limitation to be aware of. On adding this effect to a path, you'll be presented with this interface:

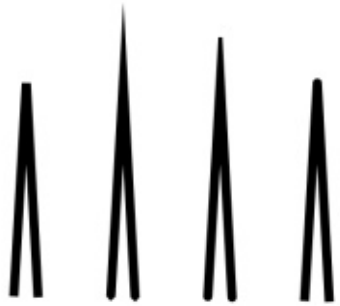


The Line Cap pop-up offers the usual options of Butt, Rounded and Square line ends. But it adds a new option as well: Peak. This gives the line ends a slight point, but there are no further options to set the size or angle of the shape. It's certainly no replacement for the Taper Stroke effect. The Line Width control, on the other hand,

offers no surprises. It just alters the width of the stroke.

The Join pop-up again offers the usual options from the Fill & Stroke dialog (Rounded, Beveled, Mitre), but throws in four types of "Extrapolated Arc" options and "Mitre Clip". More on those shortly. The Mitre option has the usual Mitre Limit control to set the limit at which really tight corners are rendered as beveled rather than a long, thin mitre – but this effect also offers a Force Mitre checkbox to force all the corners to render as mitres, regardless of the Mitre Limit. This is a nice addition that avoids you trying to work out just how large a limit you need to set if you want all your corners pointed.

With the Mitre Clip join type, and Force Mitre turned off, the Mitre Limit control takes on a new purpose. Rather than switching the join to beveled, the mitre is drawn - but cut off, as though truncated by a clipping path, at the length specified in this widget. When dealing with very sharp angles, it provides something of a halfway house between the excessive length of a full mitre and the stubby corner of a bevel. The example below shows a simple path rendered using Beveled, Mitre, Mitre Clip and Rounded joins, with Butt, Peak, Rounded and Square line caps.



As for those Extrapolated Arc options – they do for curved lines what Mitre does for straight ones. The example below shows a path made up of a pair of curved lines with a sharp join between them. The first version has a simple Beveled join, the second extends the join linearly with a Mitre, losing the style of the curve in the process. The third version, however, shows the dramatic result of using an Extrapolated Arc.

The four types of extrapolated arc vary slightly in their rendering in most cases, but the differences can be more pronounced with some paths than others. It's easiest to just try all of them and see which suits your particular image the best.

As with the Interpolate Points LPE, this effect is all-or-nothing. There's no way to apply different join types to individual nodes, beyond the effects of the mitre limit. So if you want an extrapolated arc in the middle of a series of rounded joins, for example, you'll have to manually break the path into separate objects and apply the effect to each of them separately.

Now for the elephant in the room – and it's the usual pachyderm that we've encountered numerous times when looking at path effects. In order to remain compatible with SVG, the Inkscape developers haven't just added their own line cap and join types to the file format. Instead, as with the Power Stroke and Taper Stroke effects, the results you see here are produced by rendering the path as a filled object. So if you want to use the beautiful extrapolated arcs (or the less impressive peak cap) on an object with a fill, you'll have to use one of the usual roster of workarounds that I've covered in the past – whether that's manually maintaining a second copy of the object to hold the fill, or using the Fill Between Many LPE to create a clone of your shape for filling.

This month, I've looked at what I consider to be some of the less useful new effects. They all have some interesting capabilities, but they also largely overlap with the functionality of other effects or tools. That's not to dismiss them out of hand, as they do have the ability to fill specific niche use-cases, but I doubt they'll ever be counted amongst the more commonly used tools in your Inkscape toolbox.

Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, Inked' and 'Elvie', which can all be found at http://www.peppertop.com/

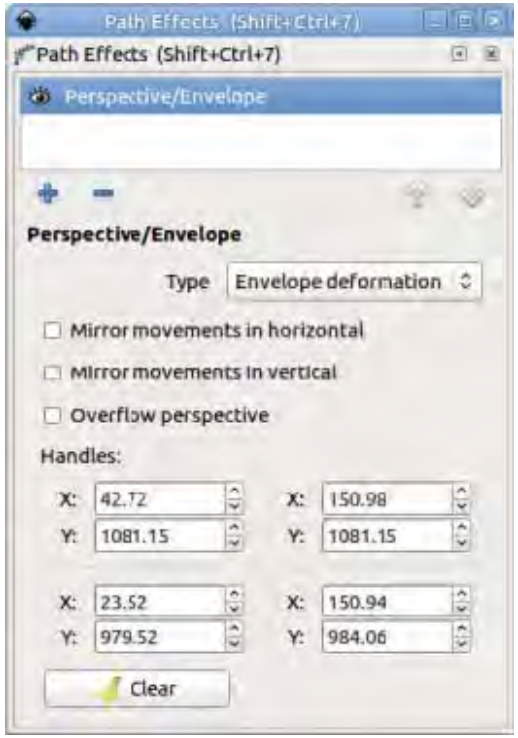
Inkscape - Part 69

This time, we're going to move on to the final batch of new LPEs in 0.92,starting with one that's both simple to use and potentially very useful: Perspective/Envelope.

One significant limitation of SVG,from an artistic point of view, is that it allows for only affine transformations. These are transformations that preserve straig ht and parallel lines, lim itin g I nkscape to rotation, scaling and skewing. Non-affine transformations allow the source image to be more radically changed, introducing curves or d istortions that let parallel lines converge. They would be a great addition to SVG – especially to allow text to be distorted into log os whilst retaining accessibility – but unfortunately it seems to be an enhancement that's of little interest to the SVG Working Group.

The Perspective/Envelope LPE provides a couple of non-affine transformations, though there's still no accessibility g ain to be had as text must be converted to paths fi rst. I n short, they allow you to distort a path (or group of paths) to follow the shape of a bounding q u ad rilateralChanging the positions of the four corners of the bounding shape causes the path to be distorted accordingly.

The effect has a simple UI: the spinboxes at the bottom display the coordinates of the four handles and can largely be ignored as it's easier to just move the handles on the canvas using the N ode tool (F2). The two "mirror" checkboxes are self-explanatory – though there's no point checking both at once as that will limit you to affine scaling that could be done with Inkscape's usual transformation tools. The Overflow Perspective checkbox allows the algorithm that calculates perspective transformations to go beyond its usual bounds, allowing for some outlandish distortions that probably have little use in the real world.



The "Type" popup lets you choose whether to use Envelope or Perspective deformation. You can freely switch between them to see which gives the best result for your particular image but, broadly speaking, Envelope is a simple distortion of your shape to fit the ou tline, whereas Perspective allows more complex distortions to provide a sense of depth. You can see the difference in this example – note the more constant thickness of the letters from left to right on the Envelope distortion (top) compared with the Perspective version.

For even more wild distortions, the "Lattice Deformation 2" LPE lets you distort your path (or group of paths) by adjusting the position of 25 nodes in a 5×5 grid. The UI for this is self explanatory – but don't be tempted to expand the "Show Points" section unless you've got a very tall monitor! I t presents a long list of fields

holding the coordinates for all 25 nodes and, unfortunately, the LPE dialog doesn't resize back down again when you close the list, leaving you wrestling with a dialog that's too big for many screens. Here's an example of the kind of transformation that is trivial to perform with this effect:

The next couple of effects allow you to create via an LPE the sort of symmetry tricks that would have previously required manually

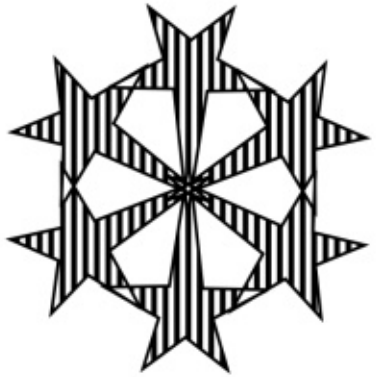
setting up some mirrored or rotated clones – as described way back in part 29 of this series. Starting with the Mirror Symmetry effect, once again we have a simple UI that can produce some rather complex results.

This effect simply creates multiple copies of the input path, rotated around a common center. The "Number of copies" field defines the number of copies that are created (the orig inal is lost in the process), whilst the "Starting" and "Rotation angle" fields are used to define the angle at which the first copy is placed, and the angle between each copy, respectively. The rotation angle is ignored if the "360° Copies" box is checked – in which case the copies are spaced evenly over a whole ci rcle.I n this imag eyou can see two stylised sun images – the former has the box checked to create a whole sun, whilst the latter uses a smaller rotation angle and different starting angle to produce a sunrise or sunset shape.

The rotational center for the effect can be manually set using the corresponding fields or, more pra ctica lly,by dragging the handle that appears in Node editing mode. There is also a second handle for adjusting the starting angle – though oddly there's no third one for setting the rotation ang le.The "Fuse paths" checkbox determines whether overlapping paths in the output are kept as separate sub-paths, or fused together into a composite shape. The difference can be seen in the following example, where the shapes in the left hand image are kept separate, whilst those on the right are fused:

If you do fuse the shapes, don't be surprised to see the output somewhat truncated. Try moving the rotation start using the on-canvas handle to gradually expose more of your original shape. The mesmerising effect as you dynamically do this is similar to a kaleidoscope – indeed the original name for this effect was Kaleidoscope!

I t's worth noting that, whether you fuse the shapes or not, the result of the LPE is (as always) a sing le path. The effect of this is clearer when a fill pattern is used on the original path – the entire output path takes on the fill pattern as one, it is not rotated with the individual shapes. I f you want the fill to rotate with each copy you'll need to use real duplicates or clones rather than an LPE.



The second of the symmetry effects, Mirror Symmetry, does a similar trick to Rotate Copies, but using reflective symmetry rather than rotational. The UI also has a few similarities:

Whereas rotation takes place around a single point, reflection req u ires a line; the pop-up offers several different options, inclu d ing reflection about the horizontal or vertical center line of the page, but by far the most useful option is (at least in my installation) labelled "Free from reflection line" - where I think "free from" is a typo for "free form". I n this mode, a mirror line appears on the page which can be moved using on-canvas handles via the Node tool. The two end handles are used to scale and rotate the line, the center one to move it without changing its orientation. If you choose the "X from middle knot" option in the pop-u,p the line becomes vertical and can be dragged via the center handle to set its distance from the orig inal path; there's also a corresponding "Y" option for a horizontal mirror line.

The "Discard original path?" option is pretty self-d escriptive – it removes the original path leaving only the reflection. I f the orig inal and its reflection overlap, the "Fuse paths" checkbox behaves the same as the corresponding one in the Rotate Copies effect: by d efinition this implies that the original object straddles the mirror li n e; the "Opposite fuse" option lets you switch which side of the line is kept and which is discarded when the fuse takes place. There's no option to keep both sides, which seems like something of an oversig ht.

Although the Mirror Symmetry effect allows only a single mirror line to be used, you can, of course, add multiple copies of the LPE, each with different settings, in order to mirror an object in multiple directions – such as in both the X and Y directions to create four shapes from the orig inal one.

We're now down to the last three LPEs that have been introduced with 0.92, and, quite frankly, they don't warrant their own article, so I 'll fin ish with a whistle-stop tour of "Bounding Box", "Ellipse by 5 Points" and "Attach Path".

What can I say about the Bounding Box LPE, other than "I couldn't get it to work"? I t's supposed to let you add the effect to a sacrificial path, then link another path to it – at which point the sacrificial path is replaced by a rectangle that surrounds the linked path. That much seems to work – at least for simple examples – but the new rectangle is also meant to follow any transformations that you perform on the linked path, and in my testing this simply did not happen. The release notes suggest that this effect could be used to add a background color to a PNG export, bu t I th in k I 'll stick to drawing a rectangle of my own and putting it at the bottom of the z-order.

The Ellipse By 5 Points effect has no UI whatsoever. All it d oes is replace your input path with an ellipse that passes through the first five points of your path. I t has no effect if your path has less than five points – or they're arranged in a way that can't form an ellipse – and it simply ignores any beyond the first five. I 'm su re it will be useful to someone, but if you just want an ellipse in your drawing and don't have a mathematical requirement for it to pass through five specific points, it's more efficient to simply draw one with the Ellipse tool and then size and rotate it as necessary.

The final LPE, Attach Path, lets you join one path to another by creating an additional line segment from the start of the path with the LPE to an arbitrary point on a linked path. You can also do the same from the end of the path, allowing a single path to potentially be attached to two others, should you wish. The position of the join on the linked path is set via the relevant position spinbox; a value of 0.00 will attach to the start of the linked path, whilst 1 .00 will attach to the end of the first line segment, 2.00 to the

end of the second, and so on. You can use fractional values as well, so that 2.50 would connect your line to a point halfway along the third segment of the linked path. These positions can also be set via on-canvas handles with the Node tool a ctive.

The new line segments can be curved at either end via on-canvas handles or the relevant Angle and Distance field s in the U I To make the lines straig ht, set the distances to zero (it would be nice if a future release of Inkscape added shortcut buttons to do this for you).

This effect provides a workaround to another missing feature from SVG – the ability to have a path with a branch in it. By attaching paths together with this LPE,the connections will be maintained even if the original or linked paths are transformed or ed ited ,making this a potentially useful effect for diagrams such as family trees.

Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, I nked 'a n d ' Elvie\which can all be found at http://www.peppertop.com/_ _

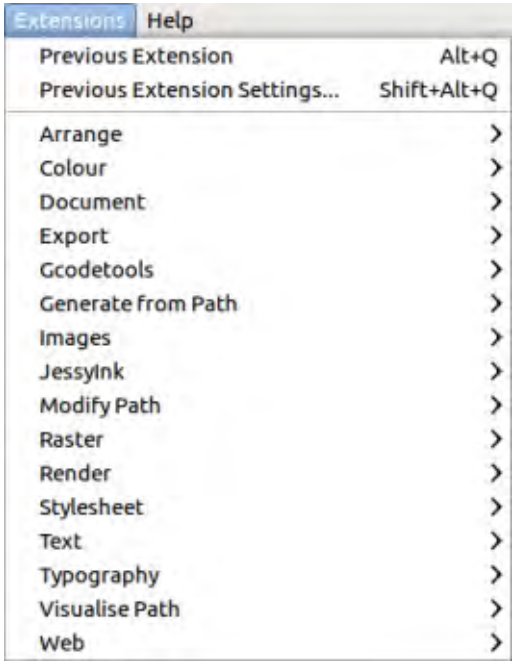
Inkscape - Part 70

With 69 previous instalments of this series behind us, it's clear that Inkscape has a lot of features and functionality, despite being slightly shackled by the limitations of the SVG file format. But there are some tasks which don't warrant inclusion in the main Inkscape application. To support these, the developers added a simple extension mechanism to Inkscape, which allows it to hand the document off to an external program for further manipulation, and receive the modified document back in response.

The program that performs the modification could be anything from a small shell script to a fully blown, compiled C++ application. In practice, most take the form of simple Python scripts. Because the program just needs to manipulate a text document, just about any language could be used, but the output must still be a well-formed SVG file, so using a language with good XML libraries certainly makes the job of writing an extension easier.

Because extensions are external programs, they can be shipped independently of Inkscape. If a manufacturer wants to create an extension to convert documents into the right format for their plotter or vinyl cutter, they can do so. With the right programming skills, users can even create their own extensions. But before considering third-party extensions, it's worth looking at the ones that ship with Inkscape by default.

Extensions can be found, unsurprisingly, under the Extensions menu. Here you'll initially find a couple of shortcuts – the first to re-run the last used extension with the same settings you used previously, and the second to open the settings dialog for the last used extension, if it has one (otherwise it also just re-runs the extension). The rest of the menu is taken up with sub-menus, each of which holds menu items to launch the extensions themselves – and, in some cases, further sub-menus before you get to the actual launchers.



If you take a couple of minutes to scroll through the available extensions, you'll realise that there are a lot of them. Over 150 of them on my default installation of 0.92! That should keep this column full for the next few years, so let's press on with the first one…

…or maybe not. I'm not so cruel as to go through each and every extension in detail, but will pick out a few examples in order to show the common UI features that you'll find. Since extensions are just normal programs, they can accept parameters, in the same way that running a tool on the command-line often requires additional arguments. The exact arguments that are needed are defined in a configuration file for the extension (this also includes other details, such as which sub-menu to put the launcher in). This file defines not only the names of any additional arguments, but also the type of value the argument expects. This allows Inkscape to generate a simple dialog, ensuring that the right type of UI widget is used for each parameter. You can tell in advance if an extension will prompt for additional parameters by looking at its name in the menu: as is the convention in computer programs, entries ending in an ellipsis ("…") will display a dialog, while those without will have an immediate effect.

As an example of an extension with no UI, let's consider the Color > Brighter extension. As you might expect this makes the selected objects brighter, which it does by altering their fill and stroke colors

– albeit by only small amounts at a time. Here's a before-and-after image, having applied this extension many, many times to the object on the right:

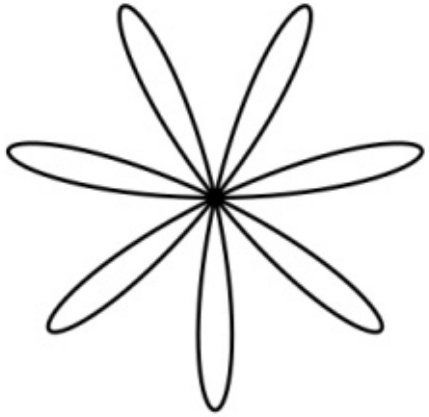
Because this extension has no UI, its effect takes place immediately, so you might think that using the "Previous Extension" menu entry, or, better still, its keyboard shortcut (ALT-Q) would be a fast way to incrementally increase the brightness of an object. Unfortunately using an extension de-selects everything in your drawing, so repeatedly calling the same extension also involves an intermediate step of re-selecting the object you wish to operate on. In many cases, including the Brighter extension, failing to select an object results in the effect being applied to every element in the drawing. If you forget this, and just hammer ALT-Q a few times you'll quickly find that everything gets brighter, not just the element you had selected when you first ran the extension.

A better approach to achieve a similar effect is to use the Color > HSL Adjust… extension. As the ellipsis indicates, this extension displays a user interface, so you can adjust the amount of brightening you wish to apply before the extension actually runs. Furthermore, recent versions of Inkscape add a "Live Preview" checkbox to the extension dialog, allowing you to see the effect of your changes before they're finally applied.

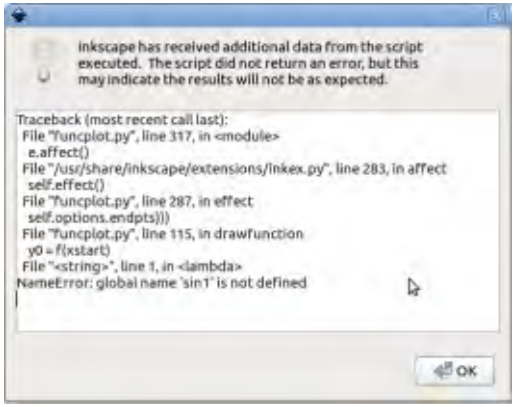
On the surface it might seem like the "Live Preview" option is a bit of a no-brainer. Why would you ever not want that checked? But consider that each extension is a separate program that needs to be launched, receive a copy of the entire Inkscape document, process it, return the entire document back to Inkscape, and then close. And this process will happen for every little change you make to the settings in the dialog. With a large document or a complex extension it can take several seconds, or even minutes, to preview the changes. Un-ticking the checkbox lets you change several settings at once without this overhead. If you already know the values you need to enter, or just want to preview after making a number of changes, being able to turn off this checkbox is vital.

This dialog also shows a few other items worth noting. Inkscape has displayed the boolean parameters as checkboxes, and the numeric parameters as the GTK3 style hybrid spinbox and slider that is used elsewhere in the program. Furthermore, the sliders have different ranges – the Hue input runs from -360 to +360, whereas the Saturation and Lightness run from -100 to +100. By using the right field types, and limiting the possible values, well written extensions can ensure that users are protected from entering nonsense that the extension will then have to deal with.

But Inkscape's selection of UI widgets is limited to a few basic types, and doesn't even allow the developer to specify a template or regular expression to validate free-text fields. For most extensions, this isn't a problem, but some do have specific requirements for the data that you enter into their fields. For example, Render > Function Plotter… lets you draw mathematical curves by entering a function into a text field. Typing "sin(x*7)" and setting the checkbox to use polar coordinates, for example, renders this seven-lobed flower.

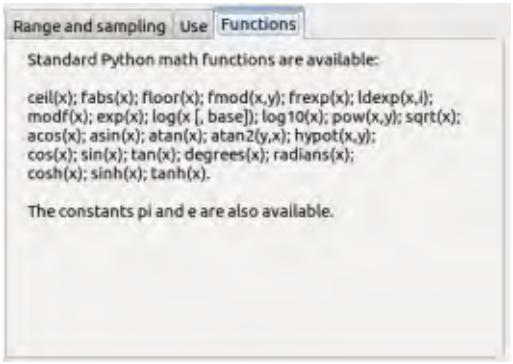


Type an invalid function into the input field, however, and you're likely to see something like this instead:



The problem is that Inkscape has no way to validate the input, and the extension has decided to send the whole Python error back to Inkscape, rather than a more friendly "The formula you entered is not valid". When you are working with a free-text field that requires specifically formatted content, it's worth disabling the live preview until you've finished editing your data.

One other thing to look out for is a Help tab, or similar, which will often contain further information about what sort of values are valid in the fields. The Function Plotter, for example, has a "Functions" tab which details the Python mathematical functions that can be used in the text field:



Whilst experimenting with different settings in an extension's dialog, you'll probably discover that you can't zoom or pan the canvas, or change the selected objects. This is particularly frustrating when using something like the Function Plotter, as some combinations of parameters can lead to shapes being drawn that are too large or small for the current zoom level. The cause of this restriction is actually the preview mode – simply un-tick the Live Preview checkbox and you'll be able to make changes on the canvas, before ticking it again to restore the preview.

Once you've finished playing with the Function Plotter, it's worth exploring some of the other extensions in the Render sub-menu. Given the ubiquity of barcode scanning software on phones now, the Render > Barcode > QR Code… extension could be a useful tool when designing a poster, flyer or other promotional material.

This menu also includes extensions to draw calendars, grids (including logarithmic and polar grids, which Inkscape can't do natively), printer's registration and color marks, and charts (though you're probably better off using a spreadsheet or dedicated graphing program for anything other than the simplest of charts). On the more frivolous side of things, there are also extensions to create spirograph-style images, simple fractal trees, and fake 3D objects.

The last extension to look at this month is one that you'll find either absolutely invaluable, or you'll probably never need to use at all. It's also one of the extensions with a less than descriptive name: Render > Hershey Text. What this extension does is to render some text using a so-called Hershey font. At first glance, this might not look terribly different from normal Inkscape text in a similar font:

This is normal text
This is Hershey Text

Zoom in, however, and things start to look a little different.

Switch them both to a minimal stroke, and no fill, and the

difference really becomes apparent:

As you can see, the normal text has a clean and simple outline with curves where needed. Conversely the Hershey text is made up of straight lines that sometimes overlap awkwardly – and just what is that little square doing in the dot on the i? The answer is that Hershey text is intended for use with pen plotters, laser engravers, and similar devices.

Suppose you're preparing a file for use with a laser engraver, but you want some solid text. Such devices really care only about outlines, not fills, so using a normal font would just result in an outline version of your text. If you wanted to fill it, you would need to draw a hatch pattern inside the text, resulting in the laser going over adjacent parts of your design repeatedly. At best, this might cost you more, as the job will take a lot longer. At worst you might find that these parts of the design become excessively scorched by keeping the beam in the same area for so long.

Hershey text, on the other hand, is not designed to be used with a fill. Rather the letters should be left as outlines, with the thickness of the beam or pen being used to provide any "fill" as the outline is drawn. With that knowledge, the little square makes a lot more sense.

If you print your designs using only an inkjet, laser printer, or even a professional printing press, you'll be fine using normal text. But if you decide to use one of the increasing number of laser cutting and engraving bureaus that accept Inkscape files, or if you purchase one of the hobbyist pen plotters that work with the program, this little extension could save you a lot of time, money or ink.

Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, Inked' and 'Elvie', which can all be found at http://www.peppertop.com/

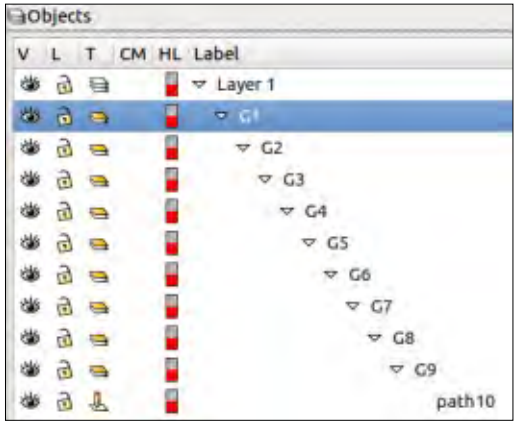
Inkscape - Part 71

Continuing from last month's introduction to extensions in Inkscape, this time I want to point out a few little utility extensions that can make some seemingly minor tasks a whole lot easier to achieve than would be possible with Inkscape's native tools.

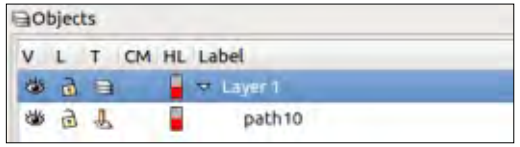
Starting at the top of the menu, we find Arrange > Deep Ungroup. As its name suggests, it ungroups any grouped objects it finds, with the "deep" part indicating that it then continues down, ungrouping any groups that were nested inside those groups, then any groups within groups within groups – and so on, until every group in your document has been expanded back to its constituent objects. It can be particularly useful when importing SVG files from other programs, some of which nest groups to such an extent that it becomes difficult to edit the content with Inkscape.

Most of the time, the default options in this extension are fine. With nothing selected on the canvas, it will ungroup everything across all layers, no matter how deep (well… up to 65535 levels deep, which may as well be infinite in SVG terms). Be warned, however, that this will even remove any layers you have (since layers are just groups with extra attributes). You can limit its effect by selecting specific groups to operate on before running the extension.

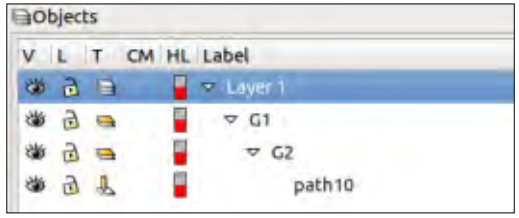
Whilst selecting specific groups lets you restrict the "breadth" of the changes, changing the values in the dialog lets you adjust the depth of nesting that will be affected. To demonstrate, I've put a single object inside a group, which is inside a group, which is inside a group… to 10 levels of nesting. The structure in Inkscape can be seen using the Object > Objects dialog (see part 63 of this series):



Using the default values, with nothing selected, breaks my path out of its deeply nested jail, and throws away the layer to boot. Better to select the topmost group, which breaks the path out whilst still leaving the layer intact:



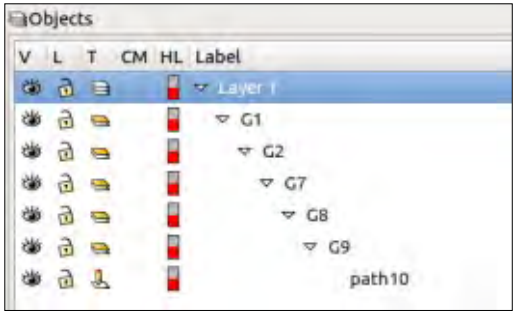
Change the Starting Depth parameter to 2, however, and the first two levels of grouping are left intact:



With the starting depth back at 0, but the Stopping Depth at 5, I get this (note that it's actually removed 6 levels of grouping as this figure appears to use "programmer's numbering" where the count actually starts at zero):

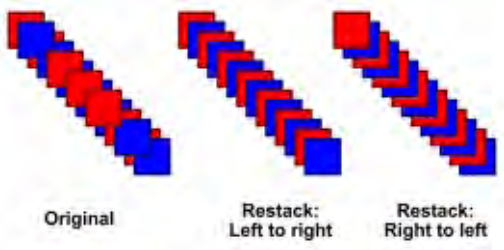
Exactly the same result can be achieved by setting the Stopping Depth back to 65535, and setting the Depth to Keep to 3. The difference is whether you want to start at the outermost group and count the levels down, or start at the innermost group and count up. I advise setting only one of these fields, leaving the other at 65535 (Stopping Depth) or 0 (Depth to Keep), as it's not clear how they interact with each other. You can, however, combine one of these fields with the Starting Depth to keep the top few groups and the deepest groups, whilst chopping

out all the ones in the middle. This can be useful when a file contains a lot of redundant nesting and you want to simplify it down without losing too much structure. For example, with the Starting Depth at 2 and the Depth to Keep at 3, my file ends up like this:



Having removed all the groups in your document, you may now be faced with a whole load of individual objects that aren't necessarily arranged the way you want them to be. Their position on the canvas should still be the same as it was when they were grouped, but their position in the z-stack could be all over the place. Much of the time this won't matter, but, when objects overlap, or you need to use them in Boolean operations, the z-order can matter immensely. That's where the Arrange > Restack extension comes into its own.

In normal operation, this extension changes the z-index on each element based on its coordinates. With the settings shown here, for example, the object whose top left corner (the Object Reference Point) is furthest to the left will be moved to the bottom of the stack, with each subsequent object from left to right being placed on top, until the object whose top left corner is furthest to the right is placed on the top. Changing the Restack Direction popup lets you change that left to right ordering so that the stacking runs from right to left, top to bottom, or bottom to top. If none of those suit, you can use the Custom tab to choose an angle that works with your design. For restacking objects that are more circularly arranged, there are even options for Radial Outward and Radial Inward. In short, this extension lets you trivially achieve this:



If your objects are already stacked in a sensible order, there are really only two things you might want to do with them: reverse the order, or randomise it. Both of these operations are also available in this extension, by switching to the "Based on Z-Order" tab.

The Modify Path or Visualise Path submenus seems like good places to look for useful utilities, but so useful are the tools in them that many have been re-implemented as Live Path Effects in recent releases, and it's usually best to use the LPE versions. Don't forget you can use 'Path' > 'Object to Path' to "set" the results if you don't want them to be "live". For example, you can find both Envelope and Perspective extensions in the Modify Path submenu, each of which requires a source path to distort, and a four-node guide path to distort into. But they have different ideas about the order of the nodes in the guide path, and whilst Envelope will happily distort a group of paths, Perspective won't, requiring you to ungroup and union the paths into a single object first. Far better to use the Perspective/Envelope LPE which avoids all these problems, and has the advantage of being able to interactively adjust the guide path.

Although Scribus makes for a far better desktop publishing program, Inskape is sometimes put to use for producing flyers or leaflets – and maybe even the occasional newsletter (although the lack of multi-page support would seem to limit its usefulness in that regard). It can also be a handy tool for mocking up a website layout. One thing that unites all these tasks is the need to lay out sections of text, either as real content itself, or as a placeholder to indicate where the real content will go. The Text submenu offers a few extensions that may help.

For placeholder text, it's hard to

beat the classic "lorem ipsum" prose – a passage of nonsensical Latin that has traditionally been used for this purpose. You could go online to one of the many lorem ipsum generators, then copy and paste the text into Inkscape, or you could just use the Text > Lorem Ipsum extension. A word of caution though: this extension produces flowed text, which makes it great for mocking up a website, but no good for actually putting online. Use the 'Text' > 'Convert to Text' menu option to fix it as normal, non-flowed text (see part 10 for more details).

Perhaps the most useful utilities in this submenu are those for dealing with small amounts of text that are already in a document. If you have several pieces of text that you want to use in another program, rather than editing each of them to copy the content to the clipboard one-by-one, the Text > Extract extension offers a means to pull out any text it finds in the page or selection, then present it to you concatenated into a single block of ASCII text in a dialog. From there you can easily copy and paste the entire text as one.

Conversely you might want to join several smaller pieces of text into a single object within Inkscape. This can be the case when importing a document from another program, as sometimes lines of a paragraph are stored as separate text elements in the SVG file. It seems to be a particular issue with PDF files. In this case, select all the text that should be joined and use Text > Merge. In almost all cases the standard settings are fine, and it will result in your text being duplicated as a single block at the top left corner of the document for you to then adjust and position as you see fit.

The counterpoint to Merge is 'Text' > 'Split Text'. This allows you to break a single text object into several separate objects, splitting by line break, word break, or even into individual characters. The styling and positioning of the text will often be lost in the process, though. One use I've had for this in the past is to split a too-long speech bubble in a cartoon into separate lines; they can then be rearranged to fit over two or three individual bubbles, and Text > Merge used to turn each group of lines back into a single text object.

As well as not being a great DTP program, Inkscape is also a pretty poor choice for dealing with raster images (also known as bitmap images, though that term should not be confused with the image format of the same name that is common in the Windows world). Still, people persist in doing so, so I'll finish this month with a quick look at the extensions that might help if you really want to go down that route.

As you may recall from part 15 of this series, when adding a raster image to Inkscape you have the choice to either embed it (in which case the raw bitmap data is included within your SVG file), or link to it (in which case the SVG contains the URL or path of the file). Embedding makes your SVG file more portable, at the expense of file size. Linking also has the advantage that edits made outside of Inkscape are automatically reflected in your document. One useful approach, therefore, is to link by default, but embed the final version of the image if you have to send your file to someone else. The Images > Embed Images extension will handle this for you.

On the other hand, if you have a file with embedded images, the Images > Extract Image extension will save them to your hard drive and automatically replace the copy in your document with a link to the newly saved file. Note that it works with only one image at a time, and doesn't complain if you have more than one selected, preferring to extract only the first one. If you subsequently need to move the saved files, you'll see the following error image in Inkscape, in place of the missing file:

If you do see this, just right-click on it and select "Image Properties" in the context menu, then edit the path to match the new location of the file. In recent versions of Inkscape, you'll also find that Embed Image and Extract Image have made their way to this context menu. It doesn't matter whether you use those options or the extensions, the end result is the same.

When dealing with raster images, you might be tempted by some of the options in the Raster submenu. My advice is to steer clear of these. They generally apply filters to your raster image (embedding it in the process, if it's a linked image) – but these are not the editable SVG filters that you would find in Inkscape's Filters menu. Instead, they are bitmap filters whose effect is to destructively change the pixels in your raster image. In that respect they're no different to the result you would get if you just edited the image in an external program such as The GIMP, except that the range of filters available in a true raster editor vastly outweighs the paltry selection exposed as Inkscape extensions. Better to copy your original image to a new name, link it into Inkscape, then play around with the filters in a dedicated raster graphics program.

Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, Inked' and 'Elvie', which can all be found at http://www.peppertop.com/

Inkscape - Part 72

This month, I'm going to look at an "application" which actually manifests itself as a collection of Inkscape extensions: JessyInk. This is a way to turn Inkscape into an editor for presentations (think PowerPoint and similar) that can then be viewed with a web browser. To achieve this, JessyInk modifies your document; it adds JavaScript code – to allow keyboard and mouse navigation and to implement some basic transition effects. A good example of what JessyInk is capable of can be seen in the "JessyInk 1.5.5 Showcase" presentation, which can be found on Canonical's Launchpad site:
https://launchpad.net/jessyink/+download

You can click the link on the page to view this with any modern SVG-capable web browser, but a better idea is to right-click on the link and save it to your local machine first. One recommended approach to creating a JessyInk presentation is to simply load this showcase into Inkscape and replace the contents with your own, in which case having a local copy of the file is a necessary first step. Note that, in my testing at least, I had to download the file directly from the Launchpad page: loading it into the browser first and then saving the running document always resulted in a file that failed to reload afterwards.

Rather than modifying the showcase document, I'm going to demonstrate how to create a JessyInk presentation from scratch. I recommend having both Inkscape and a web browser open throughout. As you make changes to the Inkscape file, you can save them (CTRL-S), and then immediately reload the file in the browser (F5) to check the results. So with your browser at the ready, and a blank document in Inkscape, let's begin...

A blank Inkscape file knows nothing about JessyInk and its extensive JavaScript functions, so the first step is to initialise it by running the Extensions > JessyInk > Install/update… extension (for brevity I'll refer to extensions just by their name from now on; they're all in the Extensions > JessyInk menu). There's nothing to do here, other than to click the Apply button, and then close the dialog once it's finished.

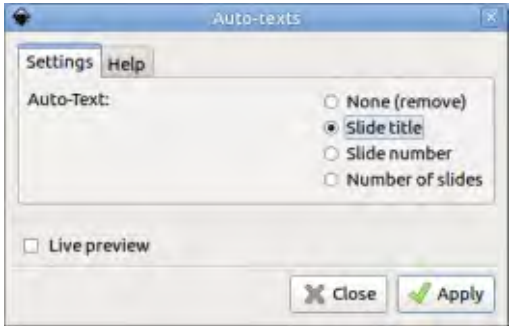
For consistency in presentations, it's common to create a master slide, containing elements that will be present in all the slides, such as a particular background or company logo. In JessyInk terms, one particular layer of your drawing can serve as a master slide, so we'll make it blindingly obvious by opening the Layers dialog and renaming the existing layer to "MASTER". Then it's worth opening the Document Properties dialog and setting the units to "px" and the page size to suit the screen you'll be presenting on (1024x768 in my case). This isn't essential, as the browser will scale your SVG to fit the screen, but does at least give you some idea of how the final presentation will appear. Note, however, that although the browser will scale the document, the aspect ratio will be preserved, so presenting on a screen of a different ratio will result in blank borders.

Finally, add common page elements to the document: I've drawn a rectangle with a gradient fill as a background, then added some solid rectangles with placeholder text for the slide title, and "nn of NN". The latter has been implemented as three separate text objects, for reasons which will become clear shortly. Here's how our master page looks:

Our master slide may look correct, but, at the moment, JessyInk doesn't know that it should be treated differently to any other slide. Run the "Master slide…" extension and supply the name of your layer ("MASTER" in my case) before clicking Apply and

then closing the dialog when it's finished.

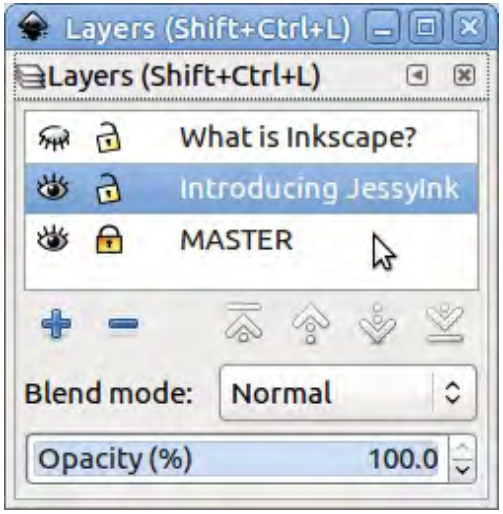
Before moving on from the master slide, let's deal with the placeholder text we've added. Select the "SLIDE TITLE" text element and run the "Auto-texts…" extension. This is used to define text elements on the page that will be dynamically replaced when the slideshow runs. For this element, you should choose the "Slide title" radio button and click Apply. Now, with the dialog still open, select the "nn" text and change the radio button to "Slide number" before applying. Finally select the "NN" text and change the radio button to "Number of slides", then apply once more. You



can now close the dialog.

We've finished with the master slide for now, so it's probably a good idea to lock the layer to prevent accidental changes when we start adding the real content to our presentation.

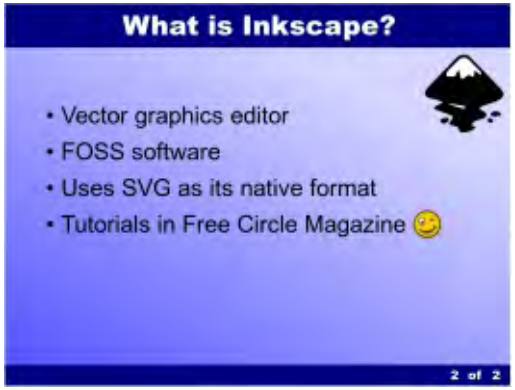
Now it's time to add some real slides. Because Inkscape (and SVG) has no inherent ability to handle multi-page documents, we can't just have each slide on a new page and switch between them using tabs or thumbnails, as you might in other presentation programs. Instead, we have to fake the effect of multiple documents by creating each slide on its own layer. This approach works, but requires a little discipline to work with: at any time you should only have the master slide and one other slide visible. When you want to switch between slides, you must remember to hide the old slide, otherwise you could easily be confused into thinking that some



of the content of the old slide is actually part of the new one. You also need to make sure you select the new layer, otherwise your modifications will apply to the wrong slide. To illustrate this, let's create a couple of layers for our first two slides, and name them based on the content we intend to put in them.

Perhaps counter-intuitively, the order of the slides runs from bottom to top, so the first slide is the one called "Introducing JessyInk", whilst the second is named "What is Inkscape?". Note that the master slide is locked, and only the master slide and the first slide are visible. The first slide is currently selected, so it's time to add some content. This can be any content that would normally go into an Inkscape document, including text, vector graphics and bitmaps. Remember, however, that it will eventually be rendered by a web browser, so the final display will be limited by the browser's SVG engine: don't use flowed text, and be aware that fonts may be different between your Inkscape machine and the final viewer. If you must use particular fonts then you should probably convert them to paths – but remember to keep a copy of the un-converted presentation, otherwise you won't be able to edit the text in future.

With some content in both slides, it's time to test the presentation in a browser. Load the file from disk (press CTRL-O in the browser to bring up a file selector), and optionally press F11 to put the browser into full-screen mode (it's the same key to return to normal afterwards). You should see that the placeholder text in the title, and in the page count at the bottom, have been replaced



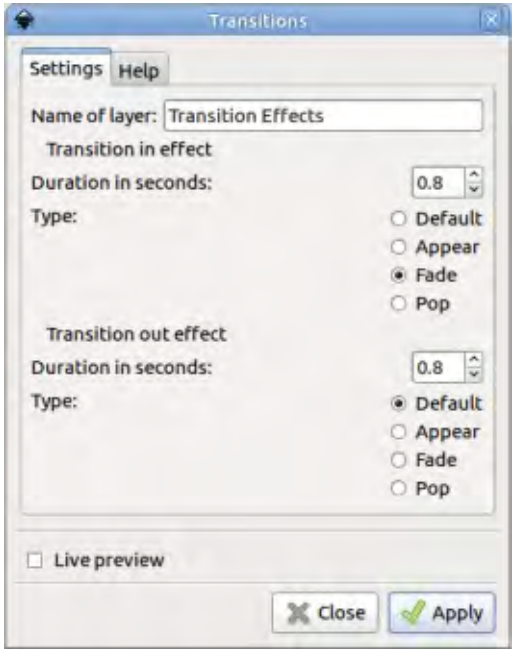
with real content.

To move through your presentation, click the mouse button or use the LEFT and RIGHT cursor keys. HOME and END will take you to the start and end of your presentation, respectively – although with only two slides, they're somewhat redundant at

contents ^

this point. Let's build up our presentation a bit more with another slide, describing JessyInk's transition effects.

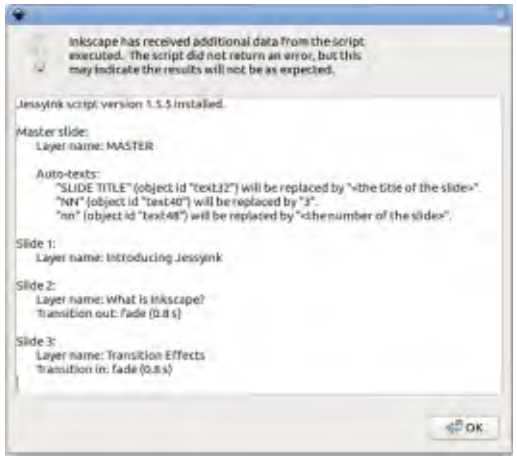
The ways in which a slide can transition in or out are limited in JessyInk, but that's probably a good thing: if you're relying on fancy effects to keep people interested then you need to rethink your presentation! To set a transition on a slide, it's handy to first double-click on the layer in the Layers dialog, then copy the layer name to the clipboard. You'll need to paste this into the Transitions… dialog to identify



which slide you're adjusting.

You can then choose how the slide should appear and disappear: Appear results in one slide immediately replacing the next with no animation. Fade causes the slide to fade in. Pop scales the slide from a small initial rendering up to its full size. Fade and Pop work best when the previous slide uses one of them as a transition-out effect, otherwise they can be quite jarring. You can also adjust the duration of the effect, although I've found the default value of 0.8s to be fine in most cases. You may be wondering about the Default transition type. This removes the in or out transition, effectively setting it back to the default behaviour – which seems to be identical to the Appear option.

After editing a few slides it can be easy to lose track of the transition state of each of them. The Summary… extension is useful here: on clicking apply it produces a dialog with a summary of the presentation, including the transition types and times you've set. The dialog always open a little small, but you can resize it rather than reading the summary line-by-line in a tiny textbox.



As well as slide transitions, similar animations can be applied to individual elements of your slide, using the Effects… extension. In this case the transition is applied to each selected element, and an "Order" field is used to determine what sequence the effects are applied in. The elements are transitioned starting with Order 1 and working upwards. Multiple elements with the same Order will be transitioned at the same time. The None (default) option is used to stop them transitioning, such that they are always present on the slide.

By now it's probably time to test your presentation once more. Simply save the Inkscape file, then press F5 in the browser to reload the file. There's no need to quit and re-launch either application, making it easy to quickly iterate your design to refine the fine details of your presentation.

If JessyInk was limited to simple slideshows with a few transitions it would be of little benefit over using LibreOffice Impress. But it also offers the ability to create "zoom and pan" presentations – originally made popular by a website called Prezi (prezi.com) with a fresh take on presentations that are more dynamic than the linear PowerPoint shows of old. Prezi's editing software originally required the use of Adobe Flash, though they now have an HTML5 offering. There's also an Open Source program called Sozi that performs the same trick, if you want to try another alternative to JessyInk.

A common theme in these types of presentation is that a single slide is used to give a big picture overview of a topic, then the viewing program zooms, rotates and pans the viewport to "dive in" to more detailed information. For our purposes I'm going to create another slide with three further "views" within it. I've marked out

each view (including the initial view of the whole slide) with a rectangle in Inkscape – and given them bright green strokes to ensure they stand out. They are hidden in the final presentation, so you can use any color or stroke style you want to help you keep track of the individual views.

I've kept the same aspect ratio for each of the rectangles as I've used for the presentation as a whole. That way I can ensure that my views are properly sized and positioned relative to the content. The one around the "3" has also been turned to demonstrate that rotation is allowed, as well as zooming and panning. With the views marked out, we just have to select each one in turn and use the View… extension to define the order in which they are visited during the presentation, starting at 0 for the initial state of the slide.

Just zooming in on three numbers isn't terribly useful, but by combining the zoom order with some object animation effects, you can make parts of the slide fade in and out as the viewer is panned around. Here's the final version of this slide, looking a little more cluttered:

In practice, however, the "1" fades out during step 1 of the slide order, and the "Zoom" text fades in at the same time. Similar rules have been applied to the other areas, and the red outline is used as a final view. The result is that the slide shows 1, 2 and 3 when it's first displayed, then zooms and pans to the red circle (showing the word "Zoom"), pans to the green one (showing "Pan"), and pans and rotates to the third one (showing "Rotate"). Finally the presentation zooms back out to the overview, which now shows the words instead of the numbers.

Now that we have a finished presentation, I'd like to mention a few more features of JessyInk. Pressing "i" during a presentation will bring up an index page showing all your slides, letting you easily jump back or skip sections using the mouse or keyboard. Pressing "d" will switch to drawing mode, with which you can annotate a slide on-the-fly. By pressing particular keys, you can even change the color and size of the pen. Run the Keybindings… extension to view or change all the various keyboard shortcuts that JessyInk offers.

To conclude, JessyInk is really a remarkable example of what an Inkscape extension can achieve with a little lateral thinking and a lot of JavaScript! The smoothness of the resultant presentations, however, is highly dependent on your browser's performance – something to bear in mind before you stand before a room full of people to give a talk. Nevertheless, if you're more comfortable in Inkscape than LibreOffice, it could prove to be an invaluable tool to know about.

Mark uses Inkscape to create three webcomics, 'The Greys', 'Monsters, Inked' and 'Elvie', which can all be found at http://www.peppertop.com/