Full Circle

THE INDEPENDENT MAGAZINE FOR THE UBUNTU LINUX COMMUNITY

INKSCAPE SERIES SPECIAL EDITION

INKSCAPE SERIES
SPECIAL EDITION

INKSCAPE

Volume Two Parts 8-14

Full Circle

THE INDEPENDENT MAGAZINE FOR THE UBUNTU LINUX COMMUNITY

About Full Circle

Full Circle is a free, independent, magazine dedicated to the Ubuntu family of Linux operating systems. Each month, it contains helpful how-to articles and reader-submitted stories.

Full Circle also features a companion podcast, the Full Circle Podcast which covers the magazine, along with other news of interest.

Welcome to another 'single-topic special'

Continuing our **Inkscape** series by Mark Crutch, all you budding artists can work through the features of this immensely capable vector graphics application in this compilation of Inkscape series **Parts 8-14,** from issues #68 through 74.

Please bear in mind the original publication date; current versions of hardware and software may differ from those illustrated, so check your hardware and software versions before attempting to emulate the tutorials in these special editions. You may have later versions of software installed or available in your distributions' repositories.

Enjoy!

Find Us

Website:
http://www.fullcirclemagazine.org/

Forums:
http://ubuntuforums.org/forumdisplay.php?f=270

IRC: #fullcirclemagazine on chat.freenode.net

Editorial Team

Editor: Ronnie Tucker
(aka: RonnieTucker)
ronnie@fullcirclemagazine.org

Webmaster: Rob Kerfia
(aka: admin / linuxgeekery-
admin@fullcirclemagazine.org

Editing & Proofreading
Mike Kennedy, David Haas,
Gord Campbell, Robert Orsino

Our thanks go to Canonical and the many translation teams around the world.

Written by Mark Crutch

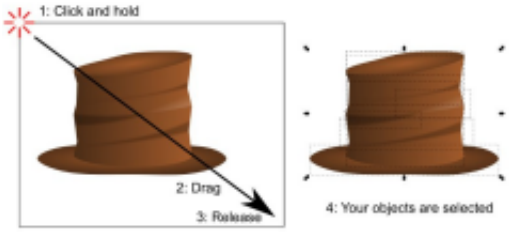
Inkscape - Part 8

If you've been following this series from the start, you should now be quite comfortable with ellipses, rectangles and stars. You should be able to construct some quite complex paths using the Boolean operations, then manipulate them with the Node tool. You can give your objects colours, gradients and patterns, as well as apply markers, thickness and different end caps to their strokes. In short, you've got enough of a toolkit that you can create quite complex drawings, should you wish. In this instalment we will look at ways to manage that complexity.

As the number of objects in a drawing increases, it becomes less and less likely that manipulating them one at a time will be acceptable. If you want to scale or rotate our snowman's hat, for example, you will quickly become frustrated if you have to first manipulate the brim, then the side, then the top – and that's without considering the shadows and highlights we added last time. A far better approach is to select all of the parts and perform your operations on them simultaneously.

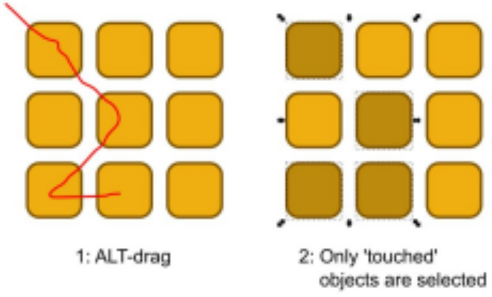
Last time you learnt how to select two objects at once by holding SHIFT as you click on the second one. If you continue to hold SHIFT while clicking on more objects, they will each be added to the selection. Holding it while clicking on an already selected object will remove it from the selection. This approach is useful when you want to select just a few objects, or if they're widely spaced with other objects in between.

In the case of our hat, however, there's a much better way. The objects are clustered closely together which makes it easy to simply drag a selection box around them, as follows: click and hold on a blank area of the canvas, then drag the mouse diagonally away from the starting point. You'll see a rectangle – the selection box – which follows the mouse pointer. When you release the button, anything inside the rectangle will be selected.



Unfortunately, there's not always a blank area of the canvas nearby. In the case of our hat, if it's in situ on the snowman, then you'll probably have the background rectangle in the way. If you try to drag a selection box by starting on another object you'll just end up moving that object instead. The answer is to hold SHIFT as you start dragging your mouse, which will prevent Inkscape from selecting the object you started on. Remember, SHIFT-CLICK will add to your selection (or remove from it), but SHIFT-DRAG will prevent the clicked object being added.

One limitation with a selection box is that it will select only objects that are entirely enclosed within the box. This can be a problem if you're zoomed right in, or your objects are too close together and you want to select only some of them. In these cases, you can use Inkscape's 'touch select' mode: just hold down the ALT key as you click and drag your mouse around, drawing a red line as you go. When you release the mouse button any objects that were touched by the red line will be selected. If you want to add to the existing selection, hold down SHIFT as well.



On many Linux systems you'll quickly discover a small issue if you try to use ALT-drag for the touch selection mode: often window managers use ALT-drag as a way to move the window around without having to drag the title bar, which prevents ALT-drag from working in Inkscape. There are three ways to

contents ^

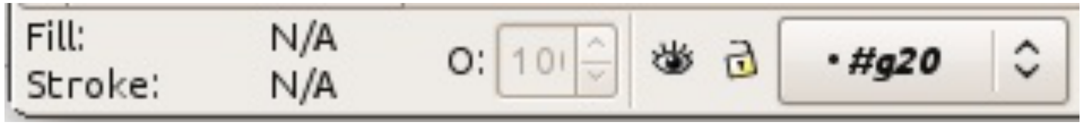
deal with this limitation:

- Disable the ALT-drag option in your window manager's preferences. This isn't always easy to do, and will usually have the side-effect of disabling it for all windows from all applications, not just Inkscape.
- Hold down the SUPER key (that's what it's nominally called in the Linux world, but you probably know it better as the WINDOWS key) in addition to ALT or SHIFT-ALT.
- Start dragging or SHIFT-dragging as though you are dragging a selection box, then press ALT to switch to touch select mode during the process. You can press and release ALT as often as you like during this process – it's only when the mouse button is released that Inkscape will check it to determine whether to use the selection box or touch select mode.

With several objects selected, it's time to combine them into a 'group' using the toolbox icon, the CTRL-G shortcut, or selecting the Object > Group menu item. Now you can move, scale, skew and rotate all the objects as though they were a single item. This makes it easy to combine the separate shapes that make up a single entity, such as our snowman's hat. You can even create a group that contains only one object, but usually grouping is used to make it easier to work with several objects that form a single part of your image.

Often you will want to modify an object that is in your group. One approach is to select the group, then use Object > Ungroup (CTRL-SHIFT-G) or the Ungroup toolbox button (shown left) in order to break the group apart into its constituent parts. After modifying your object you can re-group them again. It's usually beneficial to edit an object without the rigmarole of ungrouping and then re-grouping afterwards, so Inkscape lets you 'enter' a group in order to work with its contents directly. The fastest way to enter a group is simply to double-click on it, though there is an option at the bottom of the right-click context menu for 'Enter Group g#20'. Inkscape assigns a unique identifier to a group, so this menu entry will be slightly different for each of them.

Once you have entered a group, you are free to edit its contents individually. The status bar will show you when you're inside a group by temporarily placing its ID into the layers pop-up, to the right of the fill, stroke and opacity settings (shown above).

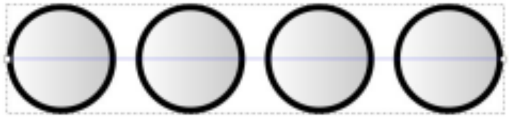


Fill: N/A
Stroke: N/A
O: 10
#g20

Having entered a group, any objects that you paste from the clipboard will be added to the contents of that group, as will any new objects you create. There are various ways to exit a group, but the most common is simply to select an object that's outside the group. You can also double-click on a blank area of the canvas if you don't want to select something else, or select 'Go To Parent' from the context menu.

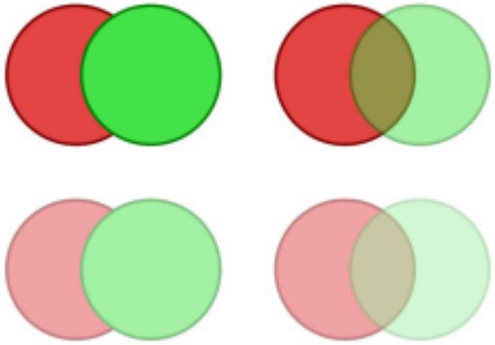
In addition to moving and transforming a group, you can also adjust its fill and stroke settings. Setting the fill or stroke will apply those values to every object in the group, as will a change to the stroke width. Unfortunately, none of the other line style attributes can be set like this, so if you want all the objects in your group to have a dashed stroke you'll need to enter the group and set each one individually.

Setting a flat color for a group's fill or stroke does exactly what you might expect. But try setting a gradient or a pattern to the whole group and you might be a little surprised by the results. Suppose you want to apply a gradient from white to black across a number of objects: you might be tempted to group them and then apply the gradient to the group. This is the actual result you would get:



As you can see, although the gradient stops cover the whole width of the group, Inkscape has actually just used a small portion of it – the amount covered by the first object – and then repeated that small amount for every other object in the group. So instead of white to black across four objects, we get white to grey across one object, which is then used for the other three as well. The same problem applies for patterns: each object has a copy of the first object's pattern applied, rather than there being one pattern that covers the whole group. There is an answer to both these problems in the form of clipping paths, but that's a more complex subject for another day.

The bottom section of the Fill and Stroke dialog does have an effect at the group level. If you set the opacity or blur for a group, it applies to the whole group as a single object. This is in addition to any opacity or blur that has been applied to individual objects, which lets you create complex combinations of effects. In this image, the top row shows two groups, one with no opacity on its objects, and one with the green circle set to 50% opacity. The bottom row shows what happens when you then also apply an opacity of 50% to the group itself.



Rather than the opacity being applied to each individual object – as a fill color would be – the circles retain their individual opacities and then the group opacity is applied to the whole. The same rules apply for blurs: the individual objects are blurred first, then the group-level blur is applied to the whole collection.

It may seem counter-intuitive to have fill and stroke affect groups in one way, while blur and opacity affect them in a different way. In practice, it's something you get used to very quickly, and the artistic benefits of having multiple levels of opacity and blur easily outweigh any short-term confusion.

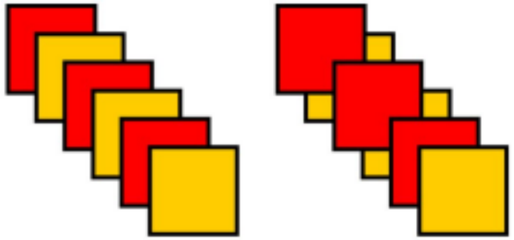
Groups are invaluable for gathering related objects together into a single easily-managed entity – such as the case of the snowman's hat at the start of this article. You could also create another group for the snowman's head, and a third for his body and arms. Moving him around on your drawing is now a lot simpler as there are only three objects to select and move, rather than the dozens you had to contend with previously. But we can make things simpler still by creating a group that is made up of our three existing groups. Just select all three and create a group in the same way that we did earlier in the article.

With a single group containing all the parts of our snowman, it's trivial to move him around. If you need to adjust the position or angle of his hat, just enter the group and you can interact with the three groups inside it. Select one of those and you can enter it again to get down to the individual objects. Inkscape lets you nest your groups as deeply as you like.

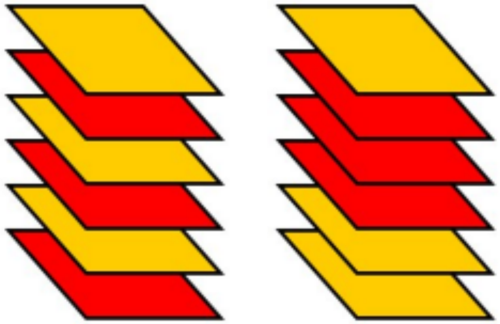
To exit a deeply nested group, you can use the same techniques as for a single group. To step back out of your groups one level at a time, you can double-click on a blank area of the canvas, or use the context menu's Go To Parent option. To jump directly to a particular level, you only need to click on another object or group that's at that level. Alternatively, you can use the layers pop-up on the status bar to jump straight to any ancestor group, or even right back up to the page level.

Nested groups follow the same rules as any other groups when it comes to fill and stroke colors, opacity and blur. Set a fill color on a nested group and all the objects, no matter how deeply nested, will be set to that color, but opacity and blur is applied to each object and group separately.

Although groups are an invaluable tool for managing complex drawings, they're not without their limitations. In particular, a group occupies a single 'slot' in the z-order stack – so you can't interleave the objects from one group with those from another. In this image, I've drawn a series of interleaved squares on the left. The right-hand image shows the result of combining just the red squares into a group: the entire group takes up just a single slot in the z-order, occupying the same slot as the topmost object in the group.



If you imagine these objects as a stack of paper sheets, it's a bit more obvious what's happened. The following image shows the interleaved sheets stacked on top of each other, followed by the rearrangement of the z-order once the red sheets are grouped together.



If you enter a group, you can rearrange the z-order of the objects, but only relative to one another. The group as a whole will always just take up a single slot. In this case, it means that the three red sheets are indivisible – you can't move one of the orange sheets between them without either ungrouping the red sheets, or moving the orange sheet inside the group as well. This limitation will prevent you from creating some groups that might otherwise be useful – a rope that snakes around both the front and back of another object couldn't be grouped as a single object – but in many other cases groups will simplify your use of Inkscape significantly.

Mark has been using Linux since 1994, and uses Inkscape to create two webcomics, 'The Greys' and 'Monsters, Inked' which can both be found at: http://www.peppertop.com/

CODEWORD

Every number in the grid is 'code' for a letter of the alphabet. Thus the number '2' may correspond to the letter 'L', for instance. All - except the difficult codeword puzzles - come with a few letters to start you off

11	12	10	10	8	10	7	6	■	7	20	6	14
■	10	■	9	■	17	■	4	■	10	■	17	■
26	17	8	14	10	7	■	22	14	6	10	8	26
■	3	■	24	■	8	10	14	■	10	■	3	■
2	10	17	8	■	24	■	12	22	15	13	10	21
■	15	■	■	■	5	■	■	■	14	■	■	■
14	6	10	14	6	10	■	4	22	6	14	10	21
■	■	■	10	■	■	■	10	■	■	■	15	■
22	4	25	17	19	14	■	21	■	1	10	14	6
■	20	■	19	■	24	21	21	■	10	■	11	■
17	19	23	20	22	14	■	8	22	4	25	22	21
■	18	■	25	■	10	■	10	■	4	■	15	■
20	6	10	6	■	4	17	7	14	16	7	10	21

1	2	3	4	5	6	7	8	9	10	11	12	13
14	15	16	17	18	19	20	21	22	23	24	25	26

ABCDEFGHIJKLMNOPQRSTUVWXYZ

Solutions are on the second last page.

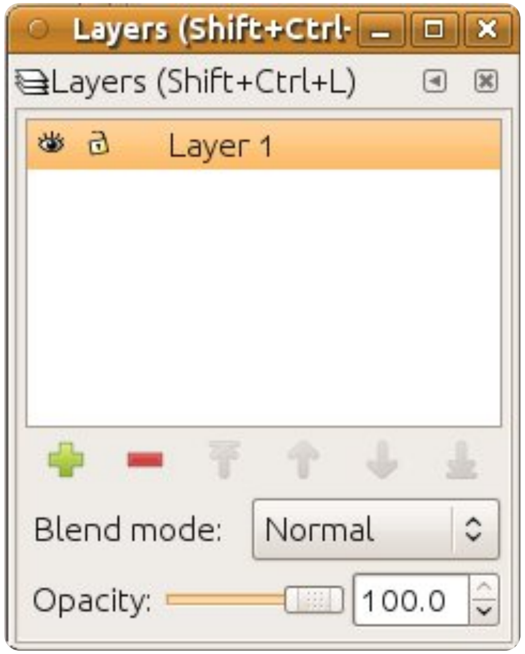
Puzzles are copyright, and kindly provided by,
The Puzzle Club - www.thepuzzleclub.com

contents ^

Written by Mark Crutch

Inkscape - Part 9

Having covered the use of groups in Inkscape, we're now going to move onto 'layers' – which are just like groups, but with a different interface for manipulating them. There's a good reason for the similarities between groups and layers: the SVG format has no concept of layers, so Inkscape actually implements each layer as a group with some extra bits of custom data.

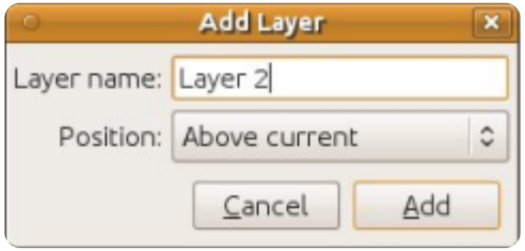


Blend mode: Normal

Opacity: 100.0

There are three parts of the Inkscape interface that are used for managing layers: the Layer menu, a dedicated Layers dialog, and some quick access tools in the status bar at the bottom of the screen. Let's start with the Layers dialog which can be opened via the Layer > Layers... menu entry, by pressing CTRL-SHIFT-L, or by using the View Layers button on the toolbox:

The layers dialog is fairly sparse, consisting of a list of layers at the top, and a few buttons and other widgets at the bottom. For a new file, only a single layer will be present, usually named “Layer 1.” Create a few objects in your drawing, and they will become part of that layer. Now try clicking on the little eye icon to the left of “Layer 1,” and you'll notice that your objects vanish. Click it again and they reappear. Click on the lock icon and you'll be prevented from making any changes to the objects in your layer. A second click will unlock the layer.



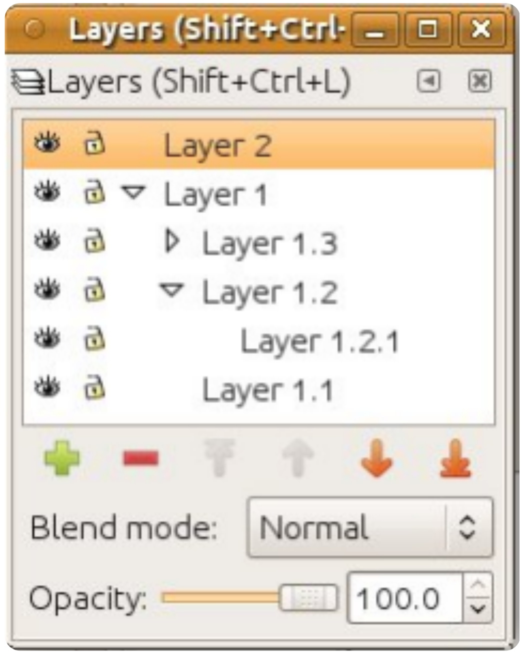
Layer name: Layer 2

Position: Above current

There are several ways to create a new layer, but the most obvious is simply to click the green + button at the bottom of the Layers dialog. You'll be prompted to give the layer a name, and pick a position for it. For now, let's call our new layer “Layer 2” and position it above the current layer.

The Layers dialog should appear much as you would expect, with two entries: “Layer 2” is at the top of the list and “Layer 1” is below it. The order of layers in the dialog represents their z-index within the drawing such that layers lower in the list will appear behind layers that are higher in the list. Check that “Layer 2” is selected and draw some more objects, ensuring that they overlap the previous objects from “Layer 1” and are in a suitably contrasting color. You should be able to see that your new objects are always drawn on top of the old objects. Using the arrow icons at the bottom of the Layers dialog, you can re-order your layers, which in turn alters the z-index of the objects in your drawing. You may recall that the contents of groups can't be interleaved, with each group occupying a single “slot” in the z-index. The same applies to layers: you can move them relative to one another, but their contents can't mix.

When creating a new layer you can use the Position pop-up to choose whether the new layer should be above or below the current layer. There is also a third option, which is to make it a sublayer of the current layer. Sublayers appear below their parent layer in the dialog, slightly indented. The parent gains a small triangle icon that can be used to show or hide the list of sublayers. In the same way that groups can be nested down to deep levels, so it's possible to add sublayers to sublayers – although going more than two or three levels deep is likely to confuse more than it helps. This image shows our “Layer 1” with the addition of three sublayers, two of which also have sublayers of their own, one of which is collapsed using the triangle button:

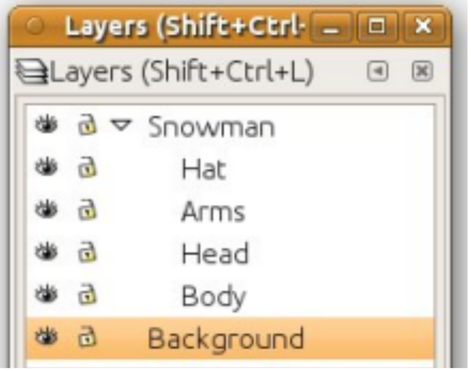


Blend mode: Normal

Opacity: 100.0

The order of sublayers within the dialog follows the same rules as for normal layers: the lower down the dialog a sublayer is, the lower down the z-index its contents will be in the drawing. From this, it follows that a sublayer's content will always appear below any content in its parent layer. Although this makes logical sense, it's often not what you want artistically, so I find that the best approach to using sublayers is simply to keep the parent layer empty of objects and construct an additional sublayer below the others to hold any background content.

The biggest advantage of sublayers is as a means to quickly hide several parts of an image at once. Considering our snowman from previous instalments, you could easily construct a parent layer to contain the snowman as a whole, with several sub-layers for each part of his body. Now you have the ability to hide or lock individual parts of him by using the icons next to each sublayer, or to quickly hide or lock the whole character using the icons by the parent layer.

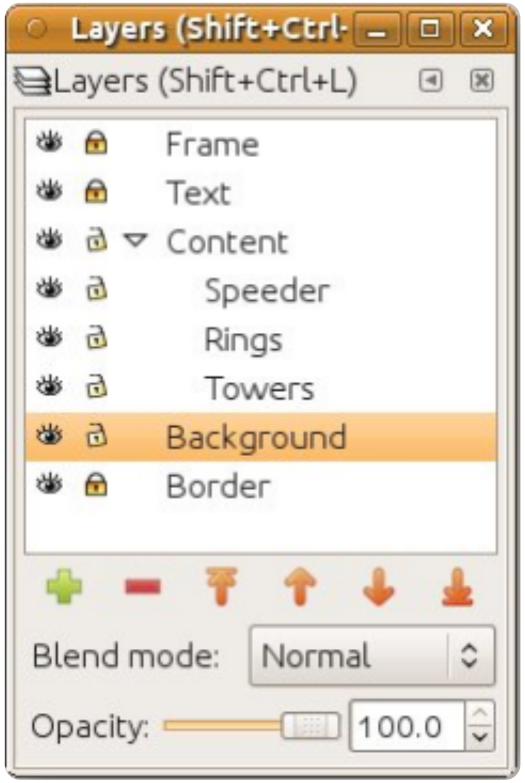


You may recall that the previous instalment suggested grouping the various parts of the snowman together to create a single object that you can easily move around. So which is better, grouping or layers? As is so often the case, the answer is “it depends”. For something like the snowman I would tend to use groups: the ability to move the whole character as one is usually more useful than the ability to easily hide it. For the background of the image – the snow, sky and stars – I would probably use a layer: you're unlikely to move the background much, but you may want to hide it when working on the snowman, or lock it to prevent it accidentally being modified once you're happy with it.

Let's look at a real-world example (shown below). This is a comic strip I drew in 2010, together with the corresponding Layers dialog. You can download the Inkscape source file from the URL at the end of the article.

At last the Hogwart's Quidditch team faced some serious competition...
http://www.peppertop.com
Some rights reserved
cc BY NC SA

Working from the top layer downwards, we first have a Frame layer which is locked. This is a thick black border that sits as the topmost layer in most of my comics and hides the ends of lines that extend outside the main image, meaning that I don't have to be quite so careful when drawing objects near the edge of the scene. Next is the text layer, which holds both the caption for this comic, and the box it sits in. That layer is also locked to prevent me accidentally moving it when editing other parts of the comic.



Below that is the Content layer. In many of my comics this is used as a normal layer in its own right, with all the main characters and props being placed directly into it. In this case, I wanted to be able to hide various parts of the drawing as I was working on it, so the Content layer is just a holder for various sub-layers. Inside each of those sub-layers the objects are grouped, such that the Towers layer contains three groups, one for each layer. That allowed me to move each tower individually, whilst still giving me the option to hide them all at once.

The Background layer holds everything else that's visible within the comic image, including the arena walls and hilly horizon. The Border layer, locked once again, holds a common border that I use with my comics which contains license information, the URL of my website and a drop shadow.

Those last two layers are particularly interesting because they both contain heavily blurred objects. Blurs are implemented as SVG filters, which will be covered in a future article, but at this stage it's useful to know that filters can be mathematically intensive, and can significantly slow Inkscape's redraw speed. One of the best ways to avoid this slowdown is to hide the layer that contains the filtered objects. Inkscape doesn't have to draw them, so it doesn't do the mathematical operations required, and it can render your image much more quickly.

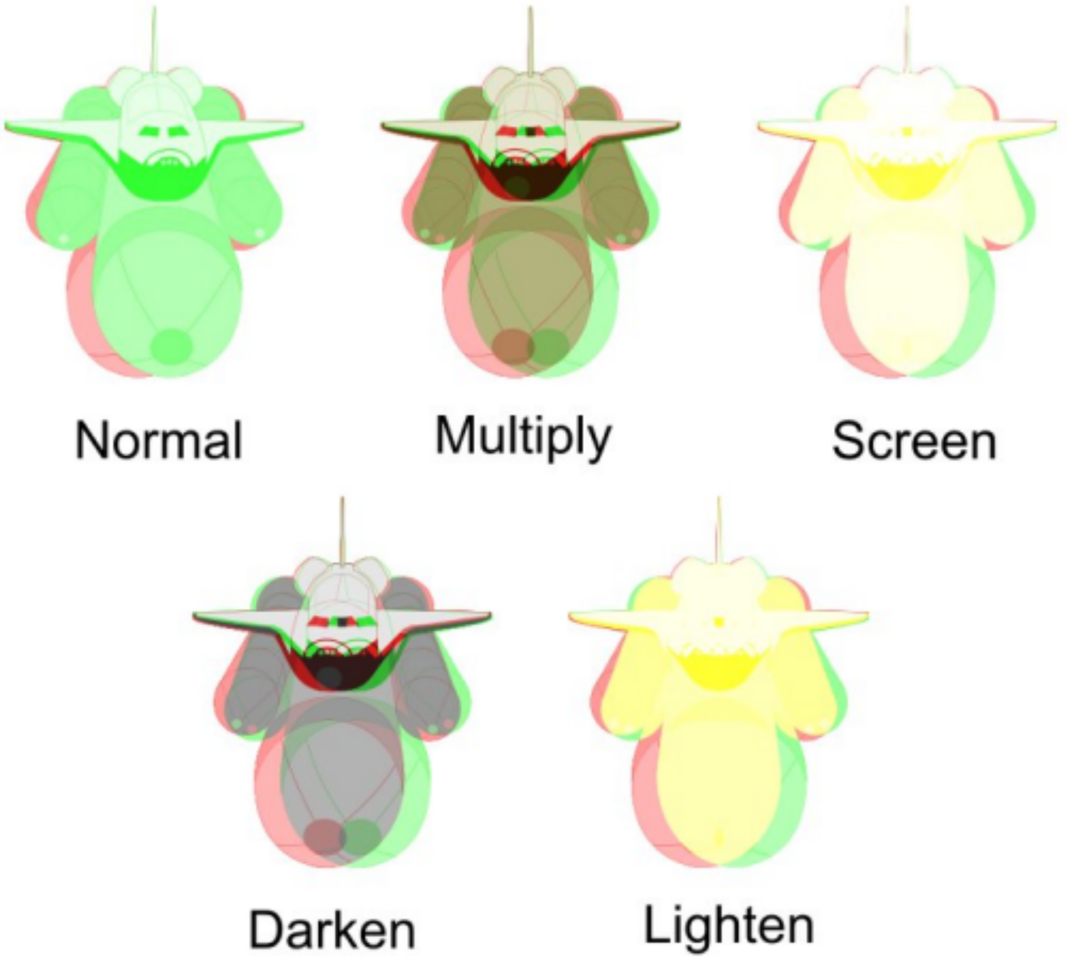
When drawing a complex image, you may find that you need to move objects between layers. This can't be done from the layers dialog, but is instead handled from the Layer menu or a keyboard shortcut. Select the objects you want to move, and use Layer > Move Selection to Layer Above (SHIFT+PageUp) or Layer > Move Selection to Layer Below (SHIFT+PageDown).

Another option you'll see in the Layer menu is Duplicate Current Layer. This not only duplicates the layer itself, but also all the objects within that layer, including any sub-layers and their contents. Because the duplicated objects appear directly on top of the originals, it's not always obvious that copies have been created, so be a little careful when using this option. You can also duplicate layers directly inside the Layers dialog by using the context menu on the layer name. Although the context menu offers up a “Rename Layer...” option, it's usually easier to just click on a selected layer name which turns it into an editable field.

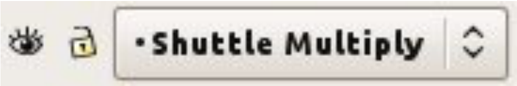
One odd omission from the context menu is the option for deleting a layer. This is available via the Layer > Delete Current Layer menu entry, or by the red minus button in the dialog. Be aware that this will delete the layer and everything within it, including sub-layers, without any further prompting. If you do inadvertently delete a layer then all is not lost, as Edit > Undo (CTRL-Z) will restore it.

Let's finish our tour of the Layers dialog with the controls at the bottom. The Opacity slider works in the same manner as the one in the Fill and Stroke dialog, but applies to all the objects in the selected layer. It's yet another of Inkscape's many ways of making objects invisible.

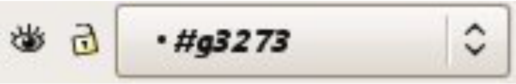
The Blend Mode pop-up defines how the selected layer is drawn with respect to the layers below it. Much like the Blur slider in the Fill and Stroke dialog, it's actually a shortcut for adding a filter. The blend modes on offer are Normal, Multiply, Screen, Darken, and Lighten – although there's rarely much reason to use anything other than Normal. I know of one artist who puts all the shadows in his images into a single layer which is set to Multiply, and I've also used the Multiple mode to produce an anaglyph 3D image of the space shuttle. In this image I've used that anaglyph space shuttle to demonstrate the differences between the blend modes, but the actual effect produced is highly dependent on the colours you start out with.



There's one aspect of layer management left to mention: the pop-up menu on the status bar. This allows you to quickly switch between layers and to hide, show, lock, or unlock the current layer using the buttons to the left of the pop-up.



The previous instalment introduced this pop-up in the context of managing groups. This is a side-effect of the fact that layers are just groups with a bit of extra metadata. In this case, the name in the button will actually be the internal ID of the group, and you can use the menu to exit a group by switching to a parent group, or straight to a different layer entirely. This can be particularly useful if you're in a deeply nested collection of groups as it provides a mechanism to jump back through several steps at once.



Beware, however, that using the Hide or Lock buttons when you've got a group selected can be dangerous, resulting in hidden groups that you can't easily reveal or locked groups that you can't readily unlock. For this reason I tend to only use the Hide and Lock buttons in the Layers dialog, or at least double-check that I'm not inside a group before using them from the status bar.

LINKS

The comics used in this tutorial, plus their Inkscape source files, can be downloaded from: http://www.peppertop.com/fc/

Mark has been using Linux since 1994, and uses Inkscape to create two webcomics, 'The Greys' and 'Monsters, Inked' which can both be found at: http://www.peppertop.com/

HOW-TO

Written by Mark Crutch

Inkscape - Part 10

The previous instalment included a comic strip which was made in Inkscape using paths, ellipses and rectangles with flat fills and gradients – all elements that have been covered in this series so far. But it also included one other type of object which is an essential element of many images: text.

Creating text objects in Inkscape isn't difficult, but does come with a few caveats that can easily trip up beginners. Some of these are common to all vector graphics programs, but the first issue you're likely to face is peculiar to Inkscape and involves a brief history lesson…

Inkscape's native file format is SVG, an open format specified by the World Wide Web Consortium (W3C). By using an open format, Inkscape creates files that can be viewed and edited, at least to some extent, in a wide range of applications. This is a huge benefit to the user, whose files aren't locked-in to being used just within Inkscape, but has the disadvantage that the Inkscape developers have little direct control over what makes it into the specification, and what doesn't.

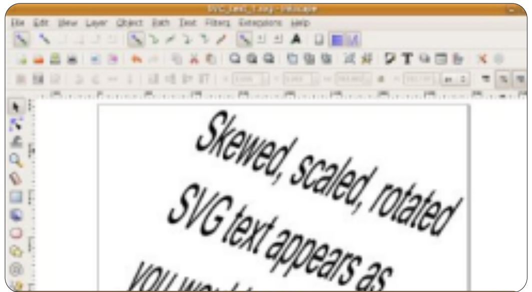
The first version of the SVG spec was released in 2001, followed by version 1.1 in 2003. After that, several years were spent working on version 1.2, which was to include many additions and improvements – including additions to support text which will wrap and reflow to fill its container. The Inkscape developers spent quite some time implementing support for this "flowed text" format, fully anticipating its official release in the new SVG standard.

Then SVG 1.2 withered and died. It never became a standard and to this day – 8 years later! – SVG 1.1 is still the latest official version of the SVG specification. This left Inkscape with the ability to create objects that are compatible with only an aborted spec, but, as this facility had made it into a release version of the software, it would break compatibility with users' files if the code was simply excised. The Inkscape developers took the pragmatic decision to leave the Flowed Text feature in place, even though using it will create files that other applications will not fully understand.

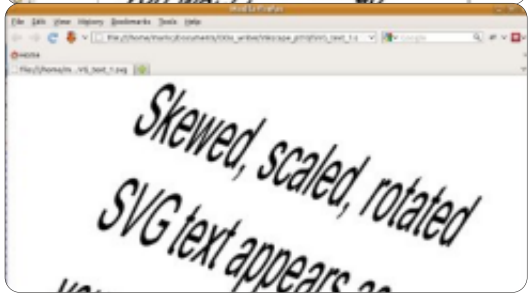
As a result of this historical issue, Inkscape can create text in two different forms: the SVG 1.1 type, which doesn't automatically flow into its container (which I'll be referring to as SVG Text), and the SVG 1.2 type which does flow (Flowed Text), but which doesn't conform to the SVG specification. The problem arises because it's far too easy to accidentally create Flowed Text, especially if you have previous experience with other graphics applications.

Let's get practical and actually create some text. First select the Text tool by clicking its icon in the tool palette, or by pressing "T" or F8.

Now, simply click inside the drawing window and start typing. If you can't see anything, check that your color and opacity settings make sense via the status bar. Congratulations, you've just created some SVG Text. If you switch back to the Select tool using the tool palette or F1 key, you can move, scale, skew and rotate your text object in the same way as any other SVG element. Because this type of text object conforms to SVG 1.1, it can be displayed or edited by various other applications. As you can see from this image, even skewing and rotating the text object in Inkscape (below) isn't enough to prevent it displaying in Firefox (bottom):

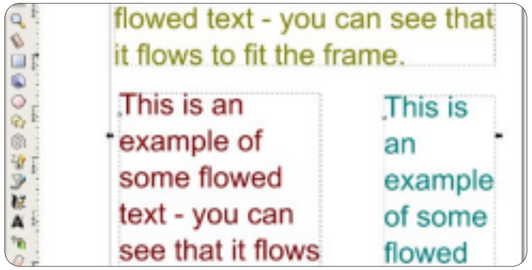


Skewed, scaled, rotated
SVG text appears as



Skewed, scaled, rotated
SVG text appears as

Some other graphics applications require you to drag a rectangle on the canvas to contain your text. This is especially common in desktop publishing programs such as Scribus, where almost everything is defined by drawing a frame to contain it. You can do this in Inkscape as well – just select the Text tool then click and drag a rectangle onto the canvas before typing. You've now created a Flowed Text object. With the Text tool still active, and the Flowed Text object selected, you should see a small square handle at the bottom-left of the text frame. By moving this handle you can change the size and shape of your frame, and the text will re-flow automatically. The following image shows the same Flowed Text object duplicated a couple of times in Inkscape. The copies have had their frame sizes changed, and you can easily see that the text has moved around, and, in the case of the bottom-right frame, it has been automatically truncated:



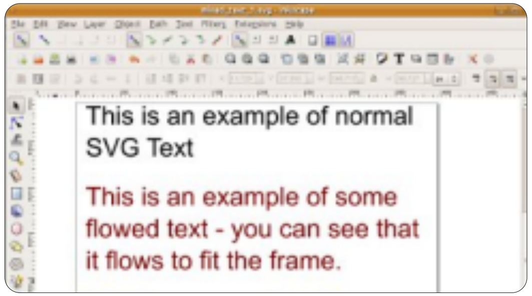
flowed text - you can see that
it flows to fit the frame.

This is an
example of
some flowed
text - you can
see that it flows

This is
an
example
of some
flowed

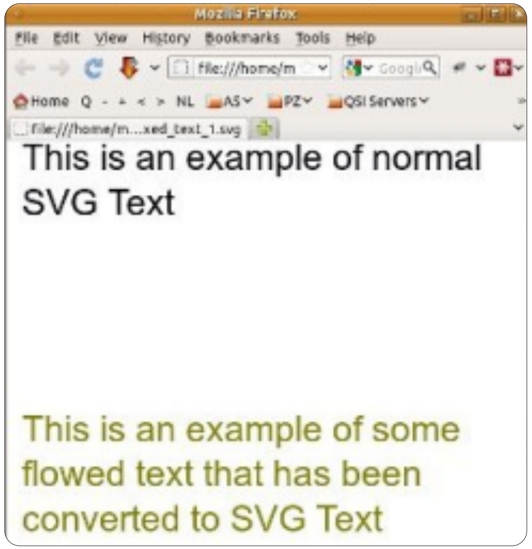
If we load this SVG file into Firefox, the result is a blank page. Firefox ignores the Flowed Text completely, and the same applies to almost every other application. Remember, the difference between creating SVG Text and Flowed Text is as simple as whether you just click, or click-drag. If you want to use your SVG files in other applications, you should almost always just click when creating your text objects. If you're in any doubt, select your text object and check the status bar, which will describe the object as either “Text” or “Flowed text”.

Despite the tone of the previous few paragraphs, there are sometimes valid reasons why you would want to use Flowed Text. If you don't want to use your Inkscape files in another application, then the presence of non-standard SVG code won't affect you. Even if you do want to use your SVG files elsewhere, it can sometimes be easier to create Flowed Text during the drawing stage, and then convert it to SVG text using the Text > Convert to Text menu entry just before you save the final version of your file. Loading the file into Firefox gives exactly the result you would expect:



This is an example of normal
SVG Text

This is an example of some
flowed text - you can see that
it flows to fit the frame.

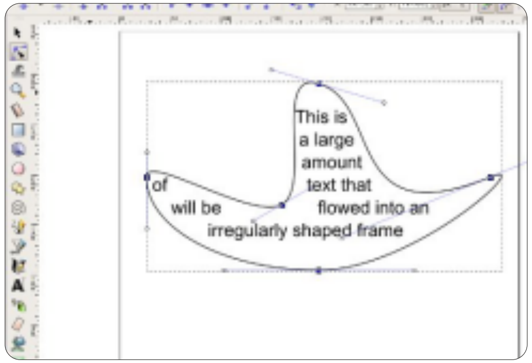


Mozilla Firefox
File Edit View History Bookmarks Tools Help
Home NL AS PZ QSI Servers
file:///home/m...xed_text_1.svg

This is an example of normal
SVG Text

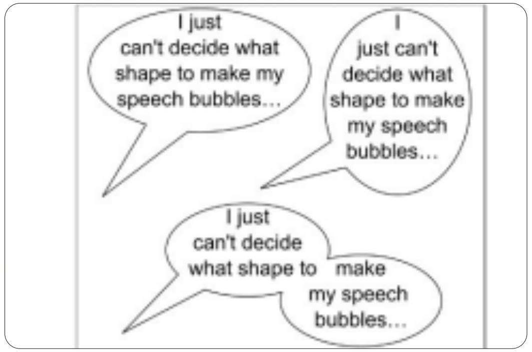
This is an example of some
flowed text that has been
converted to SVG Text

The real advantage of Flowed Text in Inkscape is that it can flow into shapes other than simple rectangles. First you will need a shape for the text to flow into: this can be any of the simple Inkscape primitives such as rectangles, ellipses and stars, or it can be a path element which allows you to create complex shapes using all the Boolean operations and node editing tools that have been described in previous articles. It must be a single object though, so can't be a group. You will also need some text, but it doesn't matter whether you create SVG Text or Flowed Text at this stage. Select both your shape and the text, and then use the Text > Flow into Frame menu to perform the magic. Note that the status bar now describes your text object as “Linked flowed text”, and that you can modify your shape as much as you like, with the text re-flowing to fit:



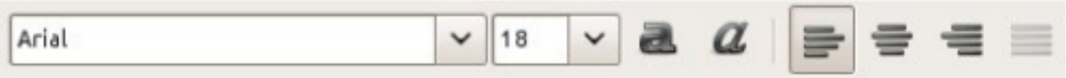
This is
a large
amount
of text that
will be flowed into an
irregularly shaped frame

As with normal Flowed Text, this will not be understood by other SVG applications. You can still use Text > Convert to Text, although with very complex paths you may find that the text shifts around a lot during the conversion. Nevertheless, for labels and speech bubbles, the ability to change your container shapes and have the text re-flow to suit, can be a real time saver.


I just can't decide what shape to make my speech bubbles...
I just can't decide what shape to make my speech bubbles...
I just can't decide what shape to make my speech bubbles...

Now that you know how to create basic text objects in their various forms, it's time to exert a little more control over the style of your words. Most commonly, you'll want to choose a suitable font, set its size, and perhaps change the justification. All of these options are available from the Tool Control Bar, and, although they can be changed at any time, it's often easier to set them before you click (or click-drag) to place your text cursor to avoid problems with the focus being in the wrong place when you start to type.

The drop down menu to the left of the toolbar lets you select a font. Inkscape can be a little fussy about its fonts, so you may find that some fonts on your system aren't available, especially those that haven't been created by a professional type foundry. There's also an occasional glitch that can occur when you first open this menu: Inkscape shows a shortened version of it, with just a few fonts listed. If that happens, simply click away from the menu to close it, and then re-open it. Finally, you may find that some fonts simply refuse to stay selected when picked from this menu. I commonly have this problem with “Arial Black”, which Inkscape immediately replaces with a bold version “Arial”. In this case, you can use the “Text and Font” dialog from the Text menu to select the stubborn typeface, which will be used when you click the Apply button.

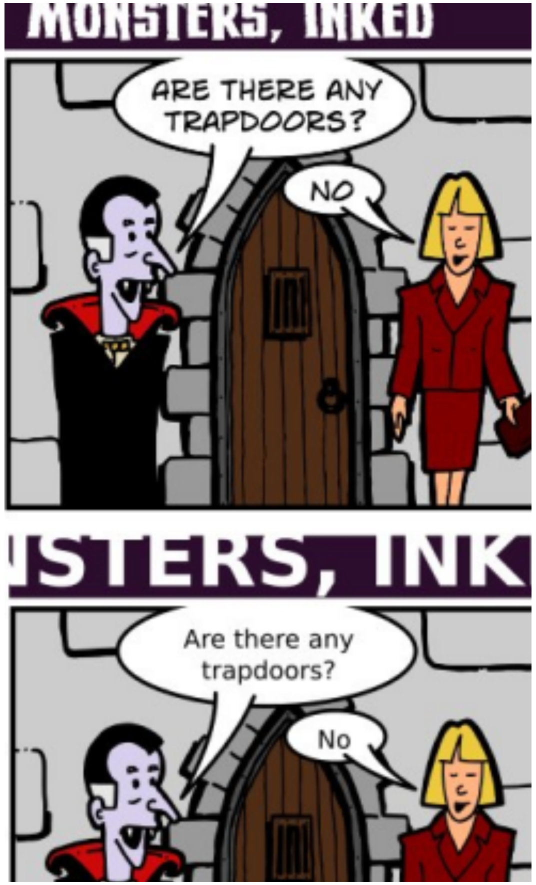


The size drop-down lets you pick from a few predefined sizes, all in units of “SVG pixels”. You can also type directly into this box to specify a different size, but there's no way to use any other units. The following two buttons are toggles, used to switch to bold or italic versions of the selected font – or a bold-italic version if both are active.

Finally, there are four buttons for setting the text's justification. Left, centre and right justification can be used for any piece of text, but full justification (where the program tries to line up both the start and end of every line) is available only for flowed text. You can convert fully justified flowed text into SVG text – which does a surprisingly good job of maintaining the justification, but it plays havoc with Inkscape's on-page text editing if you need to subsequently change the content.

Having created your text objects in Inkscape, you may want to transfer the SVG file to another machine, or post it online. In doing so, there's a good chance that you'll run into a problem with missing fonts. Text in SVG files is stored as a string of characters, together with some style information which contains the font name. If an identically named font isn't present on the destination computer, the software used to display the file will substitute it with an alternative, often with dramatic effects on the appearance of your image. The image right shows a couple of panels from one of my comic strips, first as it should appear using a couple of commercial comic fonts, and then how it appears on an Ubuntu box that doesn't have those fonts installed.

There are four possible solutions to this problem:
- Ensure that the required fonts are present also on the destination machine. This may not be possible if it's not your computer, or if the font licence prohibits it.



- Don't use the SVG format to transfer files to other systems. For my comics, I always export a PNG version of the file to ensure that my readers see a pixel-perfect representation of the image. Exporting to other formats will be the subject of part 12 of this series.
- Only use commonly available fonts. This is not a perfect solution, but may be viable in some circumstances. My “Greys” comics, for which I make the SVG files available to download, use the ubiquitous “Arial” font for this reason. Even with that precaution, they often display poorly in a web browser, and although this approach makes it easier to open the files in Inkscape, I still produce PNG versions for the benefit of casual readers.
- Convert your text into paths before saving the file.

That last solution is as simple as selecting the text then the Path > Object to Path menu entry. In Inkscape 0.48, your text will be replaced with some identical looking paths, one for each letter, grouped together. Earlier versions simply produced a single composite path that contained all the letters. Once converted to paths, the text is just another collection of shapes in your SVG file and no longer require the fonts to be installed.

This might sound like an ideal solution, but does come with a drawback: your “text” is no longer a text object, so can no longer be edited using the text Tool in Inkscape. If you decide to use this approach I recommend performing the conversion to paths as late as possible. You should also make a copy of the text object before you do so, in case you need to edit it again at a later date. The text object should then be put onto a hidden layer, dropped behind some other object, made transparent, or secreted away in some other manner so that it doesn't interfere with the rendering of the file on the destination machine.

Mark has been using Linux since 1994, and uses Inkscape to create two webcomics, 'The Greys' and 'Monsters, Inked' which can both be found at: http://www.peppertop.com/

Full Circle Side-Pod Episode Thirteen: Thats How it Feels To Be Wrong

Full Circle Side-Pod Episode Thirteen: That's How it Feels To Be Wrong

In this episode, Ubuntu Phone and TV.

Full Circle Side-Pod Episode Thirteen: Thats How it Feels To Be Wrong [1:21:56] Hide Player | Play in Popup | Download

Full Circle Side-Pod Episode Thirteen: Thats How it Feels To Be Wrong [1:21:56] Hide Player | Play in Popup | Download

File Sizes:

- OGG 45.5Mb
- mp3 52.1Mb

Running Time: 1 hour 21 mins 56 sec

Feeds for both MP3 and OGG:

RSS feed, MP3: http://fullcirclemagazine.org/category/podcast/feed

RSS feed, OGG: http://fullcirclemagazine.org/category/podcast/feed/atom

The podcast is in MP3 and OGG formats. You can either play the podcast in-browser if you have Flash and/or Java, or you can download the podcast with the link underneath the player. Show notes after the jump. (more...)

Continue Reading » An article by Robin Catling 8 Comments

Full Circle Podcast Episode 33, With Added Horsepower!!

Full Circle Podcast Episode 33, With Added Horsepower!!

In this episode we are joined by Freaky Clown, we talk about the Ubuntu Phone release, Linux peripherals and some events.

Full Circle Podcast Episode #33: With Added Horsepower!! in MP3 [56:56] Hide Player | Play in Popup | Download

Full Circle Podcast Episode #33: With Added Horsepower!! in OGG [56:56] Hide Player | Play in Popup | Download

File Sizes:

- OGG 31.5Mb
- MP3 54.9Mb

Running Time: 59 minutes 22 seconds

Feeds for both MP3 and OGG:

RSS feed, MP3: http://fullcirclemagazine.org/category/podcast/feed

RSS feed, OGG: http://fullcirclemagazine.org/category/podcast/feed/atom

The podcast is in MP3 and OGG formats. You can either play the podcast in-browser if you have Flash and/or Java, or you can download the podcast with the link underneath the player. Show notes after the jump.

(more...)

Continue Reading » An article by Oliver Clark 2 Comments

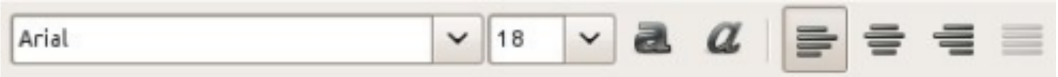
contents ^

Written by Mark Crutch

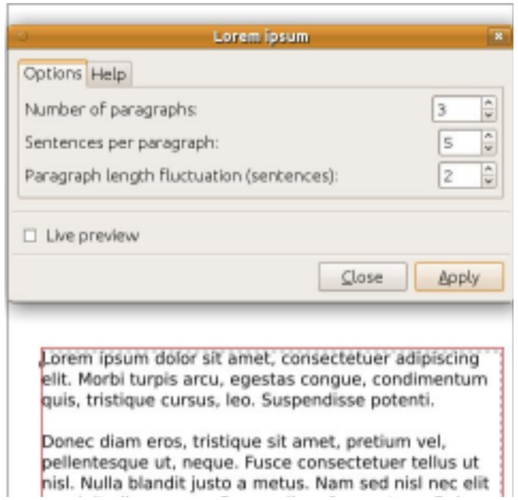
Inkscape - Part 11

In this instalment, we'll be continuing our investigation of Inkscape's text tools. Previously, you learned how to create SVG Text (compliant with the SVG 1.1 spec, and supported in many other applications), and a couple of ways to create Flowed Text (not compatible with any official SVG spec, so practically restricted to use within Inkscape). Whichever sort of text you use, when the Text tool is selected, you are presented with the same Tool Control Bar. Last time, we looked at the left half of this toolbar, where you can select a font, size, style and justification.

We used these controls to choose the settings for an entire block of text at a time, but Inkscape also lets you apply most of them to individual words or characters within a text object. The obvious use for this is to emphasise particular words by making them bold, italic, or bold and italic, but you can also change the font, size and color of parts of your text, should you need to.



To begin with, you need to have some text to modify. If you're stuck for ideas as to what to type, why not try the Lorem Ipsum extension, which will generate paragraphs of the classic nonsense Latin text that typesetters traditionally use as a placeholder. Simply select the Extensions > Text > Lorem Ipsum menu item, pick some values for the three fields, and click Apply. A Flowed Text object will be created on a new layer, with its flow box set to the size of the page. You may want to change the size of the flow box by double-clicking on the text and then moving the small diamond handle at the bottom right, or you could just make it flow into a new object using the Text > Flow into Frame menu that we looked at last time.



Number of paragraphs: 3

Sentences per paragraph: 5

Paragraph length fluctuation (sentences): 2

☐ Live preview

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Morbi turpis arcu, egestas congue, condimentum quis, tristique cursus, leo. Suspendisse potenti.

Donec diam eros, tristique sit amet, pretium vel, pellentesque ut, neque. Fusce consectetuer tellus ut nisl. Nulla blandit justo a metus. Nam sed nisl nec elit

With the Text tool selected, click in the flowed text at the point that you would like to place the text editing cursor (from now on, I'll use the term caret to differentiate it from the mouse cursor or cursor keys on the keyboard). Alternatively, if the Select tool is active, you can just double click in the flowed text to both position the caret and switch to the Text tool in one fell swoop.

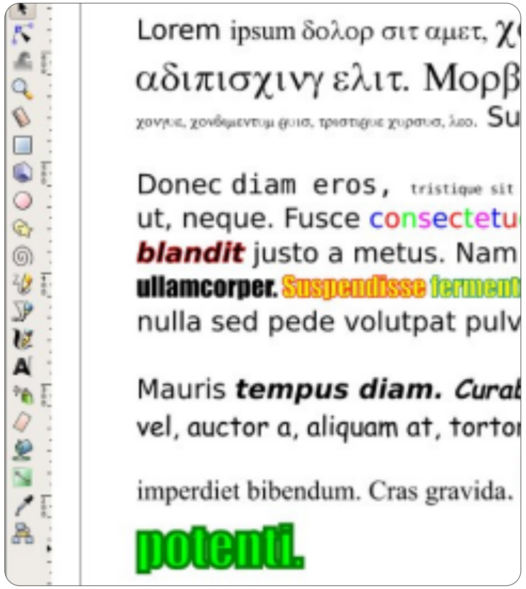
With the caret happily flashing in the middle of your text, you should be able to move it around using the cursor keys, just as you would in a word processor. Pressing the Home and End keys will jump the caret to the start or end of the current line, respectively, and holding down the Shift key whilst performing any of these movements will select the appropriate section of text. The mouse isn't without its uses either: click to immediately position the caret, or click and drag to select a contiguous section of the text. Double-click to select a word, triple-click to select a whole line.

With a portion of the text selected, it's time to play with the style. Start by setting the fill to a different color, or perhaps adding a stroke. You can set the stroke-width and join-type using the Fill and Stroke dialog, but adding markers will have no effect. Other parts of the dialog affect the text in different ways: you can set an alpha level on the fill or stroke to give it some transparency, but changing the opacity setting has no effect. If you try to use blur, gradients or patterns, you'll find the whole text object is affected, rather than just the selected section. Setting a dash style on the stroke will also affect the whole text object, though you won't notice it on any words that don't

contents ^

have a stroke applied.

On the text control bar, you can change the font for the selection, alter its size, or use the bold and italic buttons. The justification buttons work for only the whole text object, not individual selections. This does, however, mean that, if you want to left-justify one paragraph and right-justify the next, you'll have to split them into separate text objects. Nevertheless, by playing with fonts, fills, strokes and more, you can easily create some truly awful text designs.



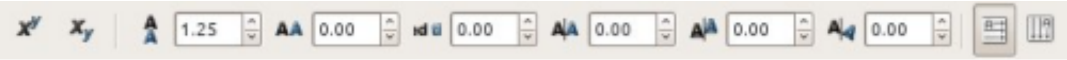
Lorem ipsum δολορ σιτ αμετ, χ
αδιπισχινγ ελιτ. Μορβ
χονγυε, χονδιμεντυμ φυισ, τριστιχυε χυρσυσ, λεο. Su

Donec diam eros, tristique sit
ut, neque. Fusce consectetu
blandit justo a metus. Nam
ullamcorper. **Suspendisse ferment**
nulla sed pede volutpat pulv

Mauris ***tempus diam.*** *Curab*
vel, auctor a, aliquam at, tortor

imperdiet bibendum. Cras gravida.
potenti.

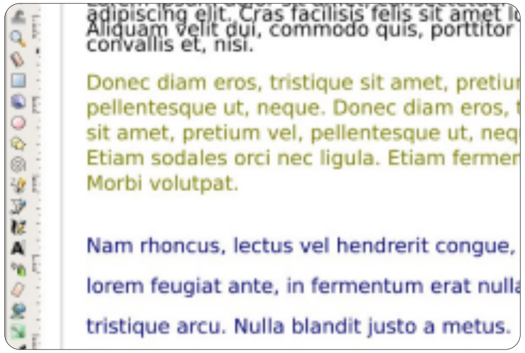
Now that you've got the hang of positioning the caret and selecting parts of the text, it's time to investigate the less frequently used icons and controls that remain on the rest of the bar.



The first of these you may recognise as Superscript and Subscript. Although you can apply them to an entire text object, they work best on a selection of just a few characters at a time. They have the effect of reducing the font size for the selection, and adjusting the text's baseline up or down. The size can subsequently be modified using the toolbar, but adjusting the position isn't so straightforward, so although these buttons are useful for simple super- and sub-scripts such as chemical formulae, they're not very useful if you want to finely position your text.

Fine positioning is precisely what the next six controls are all about. The first three work on both SVG Text and Flowed Text, whereas the last three are disabled for Flowed Text objects. The former all deal with the general spacing of your text, whereas the latter allow fine control over individual characters.

The first of the spacing controls affects the spacing between lines in a paragraph of text. The value in here is multiplied by the font size in order to produce the final spacing. You can reduce this as low as 0, in which case all the lines will be on top of each other, but it doesn't allow negative values so you can't use it as a way to make your paragraphs run from bottom to top. Typically it's set to 1.25 for normal paragraphs, though you may wish to adjust it for a looser or tighter design. This image shows three paragraphs of our Lorem Ipsum text, set to 0.75, 1.25 and 2.0 respectively.



adipiscing elit. Cras facilisis felis sit amet l
Aliquam velit dui, commodo quis, porttitor
convallis et, nisi.

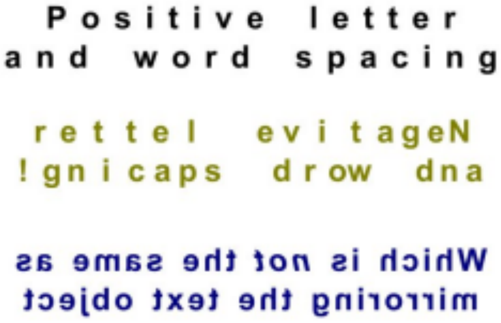
Donec diam eros, tristique sit amet, pretiu
pellentesque ut, neque. Donec diam eros, t
sit amet, pretium vel, pellentesque ut, neq
Etiam sodales orci nec ligula. Etiam fermen
Morbi volutpat.

Nam rhoncus, lectus vel hendrerit congue,

lorem feugiat ante, in fermentum erat nulla

tristique arcu. Nulla blandit justo a metus.

The next two controls are used to set the standard spacing between individual letters, and the spacing between words. The tooltips claim that both these values are in pixels, but, in my experience, typing a value directly into these – even if your document is set to use pixels as the default units – results in the value being converted to something different. In practice, it's not too great a problem as it's rare to need specific values in these fields. More usually you will adjust them up and down to make your text a little tighter or looser. These fields will allow you to enter negative values, if you really do want your text to run backwards!

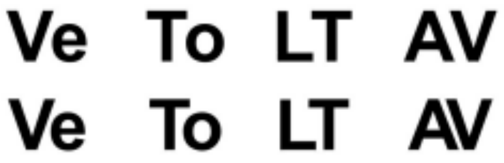


Positive letter and word spacing

rettel evitageN
!gnicaps drow dna

Which is not the same as
mirroring the text object

The next control is used to adjust horizontal kerning on SVG Text. Kerning is the term used to describe the spacing between two individual characters. By adjusting the kerning, you can arrange for characters to slot together a little more neatly, giving a more pleasing look to text, with fewer blank areas that can form visual "rivers" of white on a page. Using this field is as simple as placing the caret between the pair of letters that you wish to kern, and then entering a value to adjust the spacing. Negative values are most commonly used, to encourage the second character to tuck-in to the white space within the shape of the first, but you can also use a positive value to force a pair of characters further apart. This image shows the effect on a few pairs of characters with no kerning, and then with a negative value.



After the Horizontal Kerning control, there's a similar field for adjusting the vertical position of your characters. If you simply position the caret, then it shifts the text from that point to the end of the line up or down. Alternatively you can select specific characters or words to limit its effects – although, if your selection spans more than one line, the behaviour can be a little unexpected. Negative values in this field will move your text upwards, positive numbers will move it downwards. Combined with changing the font size, this can give you more fine-grained control than you get from using the Superscript and Subscript buttons.

The last of this group of controls lets you rotate individual characters, with the value being a figure in degrees. Positive numbers rotate clockwise, negative numbers counter-clockwise. Placing the caret will cause it to rotate just the following character. Selecting some text will rotate the individual characters of the selection, not the whole selection as one. It's not possible to select the rotation centre, but using the Horizontal Kerning and Vertical Shift controls can allow you to compensate for this if you need to.

In practice, the Horizontal Kerning, Vertical Shift and Character Rotation controls are often best adjusted using keyboard shortcuts. Holding ALT while using the cursor keys will adjust the kerning and vertical shift, whilst ALT-] and ALT-[can be used for character rotation. The Text > Remove Manual Kerns menu entry will remove all the Horizontal Kerning, Vertical Shift and Character Rotation adjustments for the whole text object, but won't remove any changes to the line, letter or word spacing controls.

The final buttons on the toolbar let you switch between creating horizontal and vertical text. In the latter case, the individual characters are the right way up, but the words run vertically down the page – as opposed to simply rotating the text object, in which case the characters are also rotated.

There's one big elephant in the room which can't go unmentioned when discussing text and SVG files: SVG fonts. The SVG specification includes a font format where the individual glyphs are defined using standard SVG objects. In theory, this should allow fonts to be created that contain colour and animations, and that can be dynamically changed by using standard Javascript code in a web browser – all while still presenting understandable text content to search engines.

Although Inkscape contains a dedicated interface for creating SVG fonts, via the Text > SVG Font Editor menu, there are a couple of reasons why it's probably not worth using. The first is that the Firefox developers have specifically rejected the idea of supporting SVG fonts, due to their lack of some layout and internationalisation features that are available in other font formats. Their concerns are certainly valid for a general purpose font format, but I think that misses some of the advantages that SVG fonts can offer when used in an SVG image, and which no other format can.

An even bigger reason not to use SVG fonts, ironically, is Inkscape itself. Although it has an interface to help create them, it has no mechanism to actually use them once they've been created. The Font Editor, therefore, is useful only if you're creating SVG fonts as an interim step towards generating a TrueType or Postscript font using an application such as FontForge.

Mark has been using Linux since 1994, and uses Inkscape to create two webcomics, 'The Greys' and 'Monsters, Inked' which can both be found at:
http://www.peppertop.com/

HOW-TO

Written by Mark Crutch

Inkscape - Part 12

Creating images, diagrams and drawings in Inkscape is all very well, but, at some point, you are likely to need them in a format other than Inkscape's own variation of SVG. Inkscape has many import and export formats, with the exact selection depending on various external applications as well. In this article I will introduce the most common and most useful formats, generally found in the file format popup of the File > Save As... dialog. Let's start with a very common file type amongst Inkscape users: SVG.

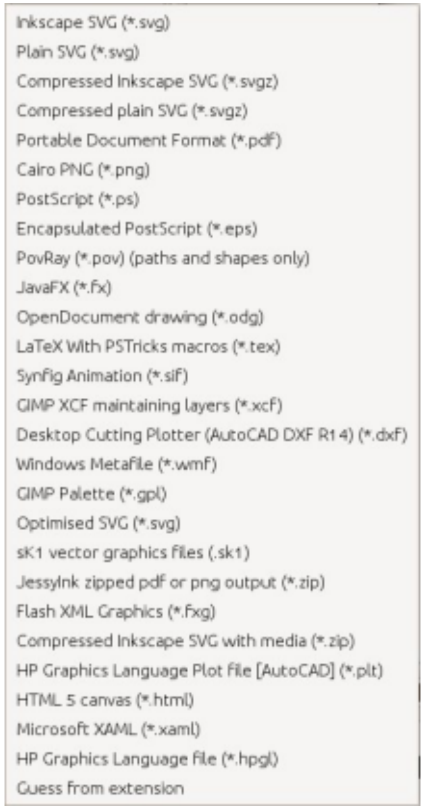
You may not have given much thought to SVG as an export format, other than knowing that it's Inkscape's default file type. Yet Inkscape actually offers six variations of SVG in its Save As... dialog, each making different trade-offs between file size and content. The first, simply referred to as “Inkscape SVG”, is the standard Inkscape format, and is the one you should probably use to store the master copies of your Inkscape drawings. It preserves all the Inkscape-specific data, which is great for use as a master format, but does mean that the file size is large, and it's saving a lot of information that most other applications won't understand.

If the size of the file is your main concern, but you still want to preserve the Inkscape-specific data, then you should use “Compressed Inkscape SVG”. This has an svgz file extension, and is the same as a standard Inkscape SVG file, but compressed using the Gzip algorithm. This can typically reduce the file size by fifty percent or more. Compressed files are more prone to data loss if the file becomes corrupted, and sometimes won't display in applications that are otherwise happy to render SVG files. Some web browsers won't render them when loaded as local files, despite being happy to accept them when issued from a web server.

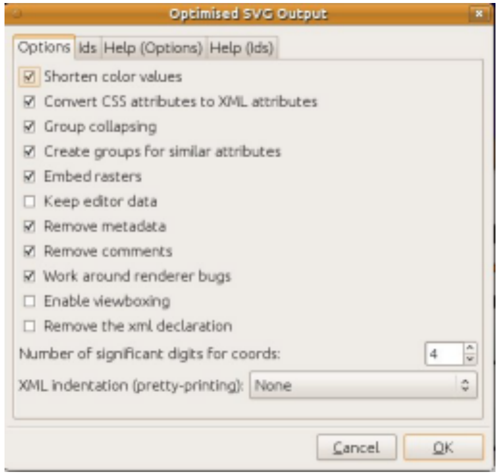
Most other applications won't understand the Inkscape-specific data in an SVG file, so you can also save a version with this stripped out. This is the “Plain SVG” option, and its Gzipped counterpart, “Compressed Plain SVG”. These will save you a few bytes and produce a purer version of the SVG that can be easier to work with if you subsequently have to edit the file by hand, or if you want to use it on a site like Wikipedia—where simple SVG files are favoured over application-specific versions. Although this might appear to be an ideal format for serving on the web, it does remove <script> elements, which limits its appeal for some web content.

If you really want to remove every redundant byte from your files, then the “Optimised SVG” option (below right) is the one to go for. This runs the output through a Python script called “scour” which is also available as a stand-alone application. It presents a dialog to let you fine-tune the optimisations it will perform, and can take a while to run if the file is complex.

Getting the best out of Scour relies on some knowledge about the structure of SVG files. There's



Inkscape SVG (*.svg)
Plain SVG (*.svg)
Compressed Inkscape SVG (*.svgz)
Compressed plain SVG (*.svgz)
Portable Document Format (*.pdf)
Cairo PNG (*.png)
PostScript (*.ps)
Encapsulated PostScript (*.eps)
PovRay (*.pov) (paths and shapes only)
JavaFX (*.fx)
OpenDocument drawing (*.odg)
LaTeX With PSTricks macros (*.tex)
Synfig Animation (*.sif)
GIMP XCF maintaining layers (*.xcf)
Desktop Cutting Plotter (AutoCAD DXF R14) (*.dxf)
Windows Metafile (*.wmf)
GIMP Palette (*.gpl)
Optimised SVG (*.svg)
sK1 vector graphics files (.sk1)
JessyInk zipped pdf or png output (*.zip)
Flash XML Graphics (*.fxg)
Compressed Inkscape SVG with media (*.zip)
HP Graphics Language Plot file [AutoCAD] (*.plt)
HTML 5 canvas (*.html)
Microsoft XAML (*.xaml)
HP Graphics Language file (*.hpgl)
Guess from extension



☑ Shorten color values
☑ Convert CSS attributes to XML attributes
☑ Group collapsing
☑ Create groups for similar attributes
☑ Embed rasters
☐ Keep editor data
☑ Remove metadata
☑ Remove comments
☑ Work around renderer bugs
☐ Enable viewboxing
☐ Remove the xml declaration
Number of significant digits for coords: 4
XML indentation (pretty-printing): None

contents ^

no compressed version of the “Optimised SVG” format available from within Inkscape's save dialog, but you can manually Gzip the resultant SVG file for the same effect.

The final SVG format is “Compressed Inkscape SVG with media”. This actually creates a zip file (not Gzip) which contains an Inkscape SVG file, plus copies of any linked media. The linked media are typically bitmap images that have been added to a drawing, but not embedded. Adding bitmap graphics is a subject we'll be covering later in this series. This format is useful for transferring an Inkscape drawing, and all its linked media, to another machine, but, ironically, it can't be opened by a copy of Inkscape on the receiving end. Instead, the zip file has to be decompressed, and only then can the included SVG file be opened.

As a comparison between the sizes of these SVG variants, I saved a copy of the snowman drawing from part nine of this series in each format. I also did the same for one of my longer comic strips. The “Compressed Optimised SVG” was manually compressed using “gzip -9 filename.svg” to give the best compression, and the filename extension then changed from “svg.gz” to “svgz”. Finally I also exported both files to PNG format to show the difference (shown above) in size between bitmap and vector graphics.

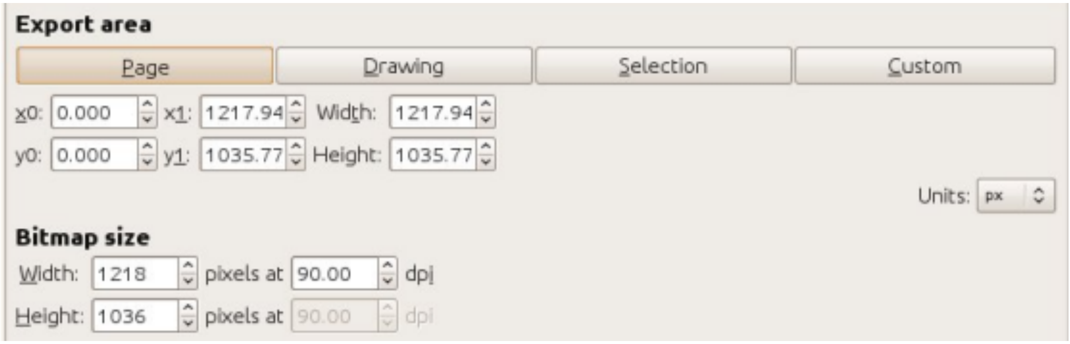
Format	Snowman	Comic
Inkscape SVG	172KB	849KB
Plain SVG	154KB	769KB
Optimised SVG	133KB	504KB
Compressed Inkscape SVG	93KB	274KB
Compressed plain SVG	90KB	261KB
Compressed Optimised SVG	87KB	209KB
PNG exported at 90dpi	211KB	1400KB

As you can see, the differences become more significant as the complexity and size of the image increase; however, we're still talking about relatively small savings in these days of multi-terabyte hard drives. Saving plain, optimised or compressed files is generally only worth doing if you have a specific reason or requirement – such as wanting to hand edit the files, or use them on a site like Wikipedia. If you've got plenty of space on your web server, it's not even worth compressing your files for online use: instead ensure that your web server software is configured to gzip the data on the fly. In my own case, I use compressed Inkscape format for the files that can be downloaded from my website – with over 200 comics available to download, it helps keep storage costs down – but I use the uncompressed Inkscape format when storing files locally.

After SVG, the most common export format is probably PNG. This is a bitmap format that can be viewed by almost all web browsers and graphics programs. It's the only standard bitmap format that Inkscape can export to, so if you want to convert your image to a JPEG, TIFF, Windows BMP file or any other type of bitmap graphic your first step will be to create a PNG and then convert it in another application.

A very common—and understandable—mistake that new Inkscape users make is to use the “Cairo PNG” option from the File > Save As... dialog. Unfortunately, this is almost never the right way to create PNGs, as it doesn't support transparency or filters. Instead you should use the File > Export Bitmap... menu entry, which will open the PNG export dialog.

The four buttons at the top of the dialog are handy shortcuts for choosing what part of your image you would like to export: the whole page, a rectangle large enough to enclose the whole drawing – which could be larger or smaller than the page, a rectangle large enough to enclose all the objects you currently have selected, or a



Export area

x0: 0.000 x1: 1217.94 Width: 1217.94
y0: 0.000 y1: 1035.77 Height: 1035.77
Units: px

Bitmap size

Width: 1218 pixels at 90.00 dpi
Height: 1036 pixels at 90.00 dpi

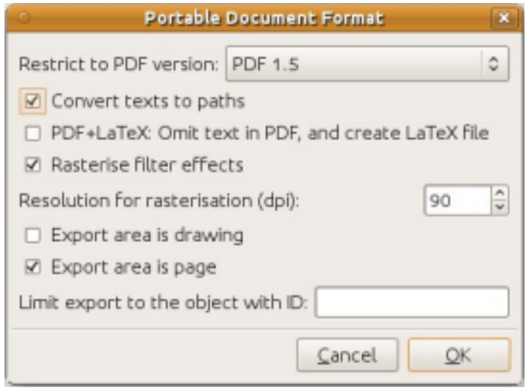
custom rectangle whose size is set by the x0, x1, y0 and y1 coordinates. You can also specify a custom rectangle using x0, y0, Width and Height, in which case the x1 and y1 figures will be updated automatically.

Usually everything that's visible on screen and which lies within the specified rectangle is exported. If you want only the selected object or group exported, without any background elements, you can check the “Hide all except selected” box at the bottom of the dialog. If you have more than one object or group selected, you can use the “Batch export” checkbox to save each of them to a separate file.

The “Bitmap Size” part of the dialog allows you to set the Width and Height of the PNG file that will be created. Alternatively, you can set the “dots per inch” or “dpi”, which will also change the Width and Height fields. Increasing the dpi will produce a file that is larger, with more pixels; reducing the dpi will result in a smaller file with fewer pixels. 90dpi is usually good for web graphics, but you might want to use 300dpi for a file that is going to be printed. If you want it printed twice as large, use 600dpi instead, or 150dpi for half the size. The rule here is the same as with a digital camera: more pixels equals more detail, but a bigger file size.

The last part of this dialog, the “Filename” section, is a little deceptive. You can type a path and name, although whatever name you use, Inkscape will always produce a PNG file. Alternatively, you can click on the “Browse...” button to bring up a file selector. The catch with this is that the file selector has a “Save” button which doesn't actually save the file. Instead it simply puts the selected path and filename into the “Filename” field, but the image isn't actually created on disk until you click the “Export” button. Exporting is usually quite fast, but can take a while for large dpi values, or if the image contains complex filters.

Moving back to the “Save As...” file selector, the “Portable Document Format”, “Postscript” and “Encapsulated Postscript” options are all variations on a theme. These are most useful when creating files for a print bureau, as they often expect to receive these “industry standard” formats. They all present a very similar looking dialog in which to set a few options.


Portable Document Format

Restrict to PDF version: PDF 1.5

☑ Convert texts to paths

☐ PDF+LaTeX: Omit text in PDF, and create LaTeX file

☑ Rasterise filter effects

Resolution for rasterisation (dpi): 90

☐ Export area is drawing

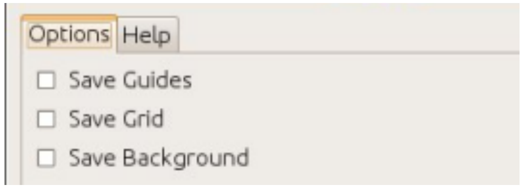
☑ Export area is page

Limit export to the object with ID:

The most important options are “Convert texts to paths” and “Rasterise filter effects”. The former will automatically convert your text objects into paths within the exported file, which prevents problems caused by using fonts on your machine that aren't present on the target machine. None of these formats support the filter effects – including the simple blur – that SVG offers, so if you don't choose to “rasterise” these, the corresponding objects will simply be omitted from the file. Rasterising consists of creating a bitmap version of the filtered content, so, much like the Export Bitmap dialog, there's a setting for the dpi. Once again a high value in here will produce more detail in the output file. If you're creating a PDF to view on a computer screen, then 90dpi may be sufficient, but, if you want a high quality print, or you expect your readers to zoom in, you should probably set this higher.

An alternative way to create a PDF for a print bureau is to import your SVG file into Scribus and use that to generate the PDF file. Unfortunately, Scribus doesn't support all of Inkscape's SVG features, so this approach may not work for more complex drawings. My experience has been that it's often easiest to simply export a high resolution PNG file and let the print bureau deal with getting it into a suitable format for their systems. The downside of this approach, other than the large file you'll need to create, is that your vectors are rasterized prematurely, so you probably won't get the smoothest possible output. For small items, that may not matter, but, if your plan is to create posters or billboards, it can make all the difference. Inkscape's PNG files are all in RGB format, and some particularly fussy bureaus may complain that they want CMYK files. Even if they're happy to work with PNG files, do make sure you get a proof from them first to check that the colors are what you expect.

There are a variety of other export formats available in the Save As... file selector, serving various niches. Each has its own limitations on what Inkscape and SVG features can be successfully represented. For more details about many of these formats read the Exporting Files section of Tavmjong Bah's excellent Inkscape manual (link at the end of the article). Given the target audience of Full Circle Magazine, though, there's one more format that should be discussed: “GIMP XCF maintaining layers”.



Options Help

☐ Save Guides

☐ Save Grid

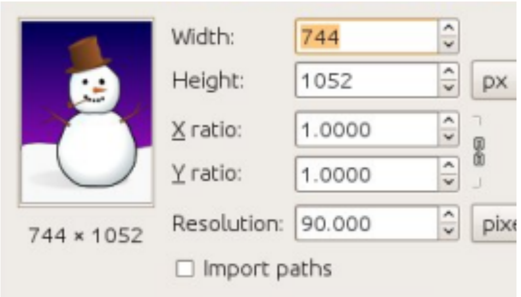
☐ Save Background

This export format is available only if GIMP can be found in the system path. For most Linux systems, this will be done automatically if you install the application using your package manager, or if it's installed as part of the default installation. Once it's available to you, selecting it will present a dialog with a few options for the export.

The “Save Guides” and “Save Grid” options will include any Inkscape guides, and the first rectangular grid, as their GIMP counterparts. We haven't talked about guides and grids yet in this series, but, if you've stumbled across them on your own, these options might be useful. In practice, the guides may be useful, but the grid will appear much denser in The GIMP than in the original Inkscape file, and can even be so dense as to obscure the image entirely! The “Save Background” option is also a problem as it applies the document background color (set in File > Document Properties... within Inkscape) to every single layer, rather than just creating a single background layer. In practice, therefore, I would suggest leaving at least the latter two options unchecked unless you have a particular reason to do otherwise.

Loading the resultant XCF file into The GIMP will produce exactly what you expect – a bitmap representation of your Inkscape file with each of the Inkscape layers present in a corresponding GIMP layer... almost. Unfortunately, any sub-layers are automatically rendered into the parent layer, rather than remaining as separate layers in The GIMP. If you want to keep your sub-layers separate, you'll need to promote them to top-level layers before you export the file. The images created by this export option are fixed at 90dpi – if you want a different sized image then you have to scale the objects within Inkscape first.

An alternative to creating an XCF file from Inkscape is to instead load an SVG file directly into The GIMP. This will not preserve any layers, flattening the image down to a single layer instead. Some more advanced SVG features, or Inkscape-specific additions, may not render correctly. Loading an SVG file into The GIMP does, however, let you set the size of the rendered bitmap



Width: 744

Height: 1052 px

X ratio: 1.0000

Y ratio: 1.0000

Resolution: 90.000 pixe

744 × 1052

☐ Import paths

The “Import paths” option will create a GIMP path for each object in your SVG file. This can be useful if you wish to convert one or more paths into a selection in order to limit the scope of your GIMP edits. Generally, it's worth checking this box – you can simply ignore the paths if you don't need them, but, having them available can make some editing tasks a lot easier. The “Merge imported paths” option is less useful. It creates a path for each object in the SVG file, then merges them all into a single path. Given that you can combine multiple paths into a single selection within The GIMP, leaving them separate still allows you to create a single unified path if you need to.

LINKS:

Scour: https://launchpad.net/scour

“Exporting Files” in the Inkscape manual: http://tavmjong.free.fr/INKSCAPE/MANUAL/html/File-Export.html

Mark has been using Linux since 1994, and uses Inkscape to create two webcomics, 'The Greys' and 'Monsters, Inked' which can both be found at: http://www.peppertop.com/

Inkscape - Part 13

Written by Mark Crutch

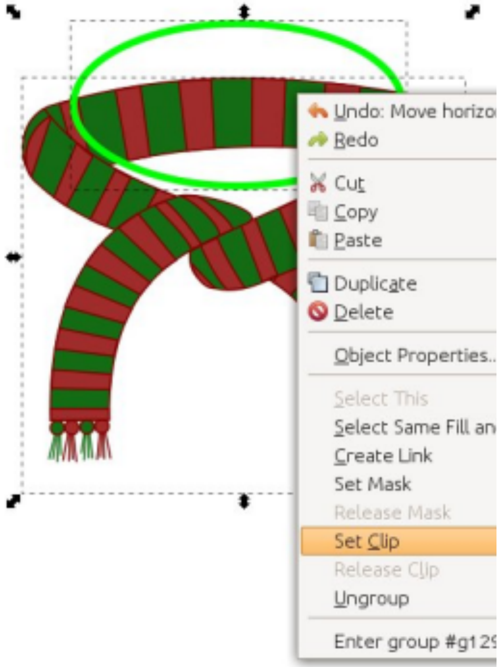
One thing you've probably noticed about Inkscape throughout the course of this series is that it operates with a stack of objects placed one above another—the "z-order." You can introduce some overall structure by putting objects into groups and layers, but it's simply not possible to create an element that passes both over and under another object.

Because we're mostly interested in creating an artistic result, the solution is simply to put the object on top, and then remove the part that would be hidden behind our second object. One way to remove the hidden part is to edit the paths that make up our problem object, perhaps using Boolean operations to cut out sections. But what if the object is complex, such as a group of many individual elements that would each have to be cut independently? A better option in this case would be to tell Inkscape that it should draw only certain areas, and leave the others transparent for the background object to show through. This is done by creating a path and applying it as a "clip path." Any part of the object inside the path will be visible, whereas the parts that are outside it are not drawn at all.

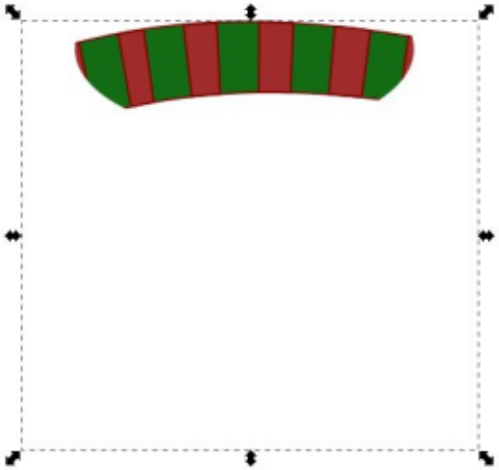
As an example let's resurrect the snowman from earlier in this series, and give him a scarf. We could do this simply by drawing the front part of the scarf onto the snowman, but, in this contrived case, our scarf already has a back section that we need to clip out of the final image.

Despite its name, a clip path doesn't have to be a path. It can be any closed shape—a rectangle, oval, star or polygon. It can also be a complex path which contains multiple sub-paths (more on that later). Whatever shape you use, however, it must be a single object rather than a group or a selection of several elements.

Defining a clip path is simple: just draw the path or shape you want, ensuring that it sits higher in the z-order than the object or group you want to clip. Now select both the clip path and the object you wish to clip, and use the "Set Clip" option on the right-click context menu. The Object > Clip > Set menu item has the same effect. Here, I've created an oval on top of the scarf. I usually draw my clipping paths with a bright green stroke and no fill—so that they're easy to spot against the item I'm clipping. The color is irrelevant, as is the thickness of the stroke: it's purely the shape of the path that is used to define the clip.

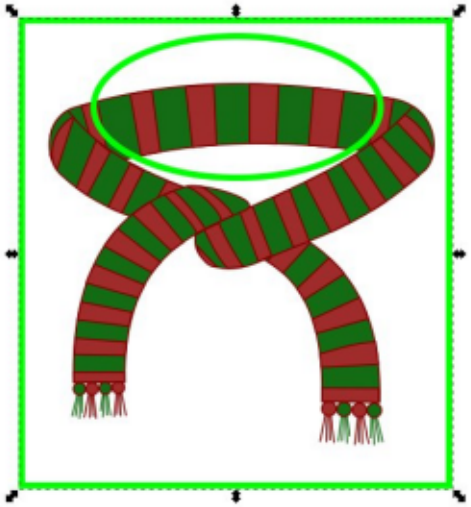


Unfortunately the result of this clip is the opposite of the effect we're looking for. Instead of removing the back of the scarf, we're left with nothing but the back of the scarf!

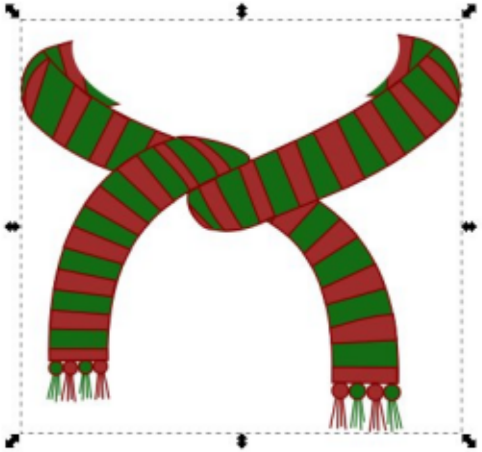


contents ^

Remember, clipping always leaves the content inside the clip path visible, and hides everything outside it. There's no way to invert this behaviour to hide the content of the path, so, instead you have to work around it by creating a more complex path that does the job for you. This is an ideal opportunity to use the Boolean operations that were introduced in Part 7 of this series. In this case, you just draw a rectangle that's large enough to encompass the whole of the scarf then, ensuring the oval is on top, use Path > Difference to "subtract" the oval from the rectangle.



At first glance this may appear to simply be a rectangle with an oval on top of it, but, in practice, this is now a single complex path. "Complex" simply means that it's made up of separate sub-paths. The rectangle is one sub-path, and the oval is a second sub-path, but the whole object is still considered to be a single path and can therefore be used for clipping. Once again, only the area inside the path will remain, but in this case "inside" consists of the space between the oval and the rectangle; "outside" is anything beyond the rectangle, or within the oval. Try setting a fill if the difference between inside and outside isn't clear. Clipping with this complex path gets us much closer to our desired result.



Remember, this is a purely visual effect. The original scarf remains the same, it's just that parts of it are not being drawn. You can undo the effect at any time by selecting a clipped object then choosing Release Clip from the context menu, or Object > Clip > Release from the main menu bar. This will return the object to its normal appearance, and also make the clipping path visible again.

If we add the scarf to the snowman you can see that we're close to the effect we're looking for, but there are still parts of the scarf that are being drawn in front of the character's head.

The problem is that the clip path we've used was just an approximation, and needs to be tweaked to more closely match the section that we want to hide. In earlier versions of Inkscape, the only way to do this was to release the clip, adjust the path, then re-clip. Even with the latest version, that's still the only option if your clip "path" is actually a rectangle, star, oval or other primitive object. Thankfully, our Boolean operation has turned our combination of an oval and a rectangle into a fully fledged path, which can be edited without releasing the clip since the 0.48 version of Inkscape.

To switch to editing mode, you have to select the clipped object (as usual keep an eye on the status bar where a clipped object will be described as such), then switch to the node tool—using the icon on the tool palette or by pressing "n" or F2. You should now see your clipping path rendered as a single pixel green path, regardless of the color or thickness of the original path. The green color in this mode is the reason I choose bright green when drawing my path in the first place—it helps to reinforce the mental link between green paths and clipping. If you don't see the green path, make sure the "Show clipping path(s) of selected object(s)" button (in the tool control palette) is active.

From this point you can just edit

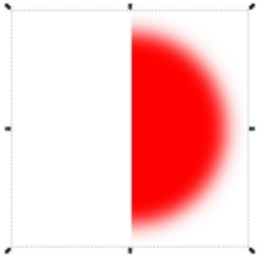
contents ^

the path using the node tools that were described in Part 6. You may find it easier to reduce the opacity of your clipped object so that you can see how your clipping path relates to the objects behind. The changes to the clipping path will be applied live, so you can see the result as you work. Once you've finished editing, return your object's opacity to 100% if you need to, and change to the selection tool to hide the green clipping path and see your results. Remember that the spacebar can be used as a convenient shortcut to switch to the selection tool and then back to the node tool during editing, if you find that the nodes are obscuring your view.

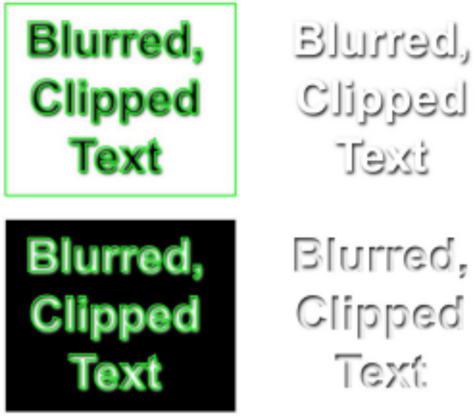
With a little time spent on node editing, you should be able to adjust the clipping path to follow the head of the snowman and make the scarf seem to disappear behind it.

Sometimes you will be able to get away with a simple rectangle or oval as a clipping path, but usually it's worth pressing CTRL-SHIFT-C (or using Path > Object to Path) before setting the clip, simply for the advantage of being able to edit the path without having to un-set then re-set the clip.

Because clipping always results in a hard edge, it can be used to create some effects that would otherwise be very difficult to produce. Wherever you have to stop a blur from spilling over an edge, clipping can provide a solution. Consider something like a semicircle which should only be blurred on the curved edge and cut cleanly on the straight edge. Without clipping this would be a very difficult image to create.



The ability to cut out parts of a blurred object can lead to some particularly interesting effects when used with text. For each of these examples, I've created two copies of a text object. One of them is blurred, whilst the other has been converted to a path and used to clip the blurred version. Different arrangements of clip path, blur, text, background and color choices can produce a variety of results.



Clipping can also be used to work around the limited selection of gradient types. Inkscape follows the SVG standard in supporting only linear and radial gradients, but other types can often be emulated through careful use of blurred objects and clip paths. In this example, a few blurred segments and a donut-shaped clipping path makes an acceptable substitute for a conical gradient when drawing a CD.

Although clipping has many uses, and is a technique that's well worth getting to grips with, sometimes you need a little more subtlety than the hard edges it creates. Next time, we'll look at masking—a related technique that lets you fade objects out gradually, rather than just stopping them dead at the edge of a path.

Mark has been using Linux since 1994, and uses Inkscape to create two webcomics, 'The Greys' and 'Monsters, Inked' which can both be found at: http://www.peppertop.com/

HOW-TO

Written by Mark Crutch

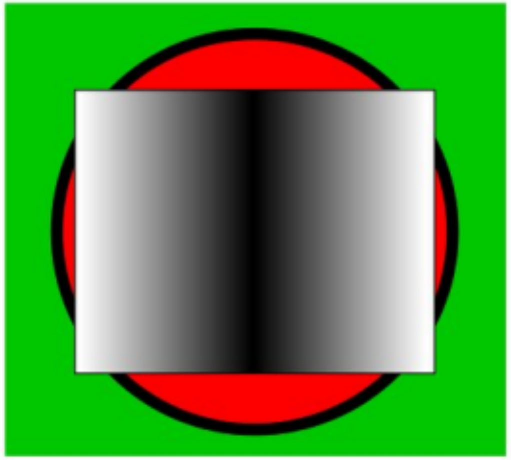
Inkscape - Part 14

Transparency – or its counterpart, opacity – is such a fundamental concept in Inkscape that it was one of the earliest topics covered in this series. Back in Part 3 you were introduced to the Opacity spin-box on the status bar, which gives you the ability to set a single transparency level for a whole object, or even a group of objects.

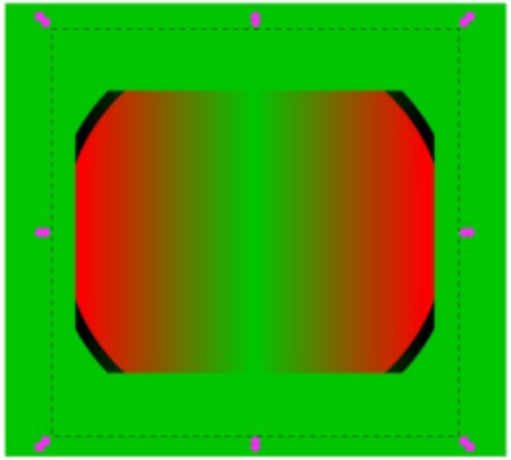
If you want a single object to have varying opacity – such as completely transparent at one end, and opaque at the other – you can use a gradient for the fill and stroke. But what about doing the same for a group of objects? You could set suitable gradients on each individual item in the group separately to get the desired effect, but if your group contains lots of objects, then that approach becomes tedious very quickly.

The solution to this problem is masking. This is a technique that uses a grayscale image as a means of specifying the opacity of another object or group. Any white sections in the image cause that part of the masked object to appear opaque. Black sections cause that part to appear transparent. And grays in-between result in varying levels of opacity.

Let's look at an example. In this image (below) I've placed a solid red circle on top of a solid green background. The green is there mainly to show the effect more clearly. On top of the circle is a rectangle with a white-black-white gradient. Using this rectangle as a mask for the circle gives the result of a clipped circle that fades from opaque to transparent and back to opaque.



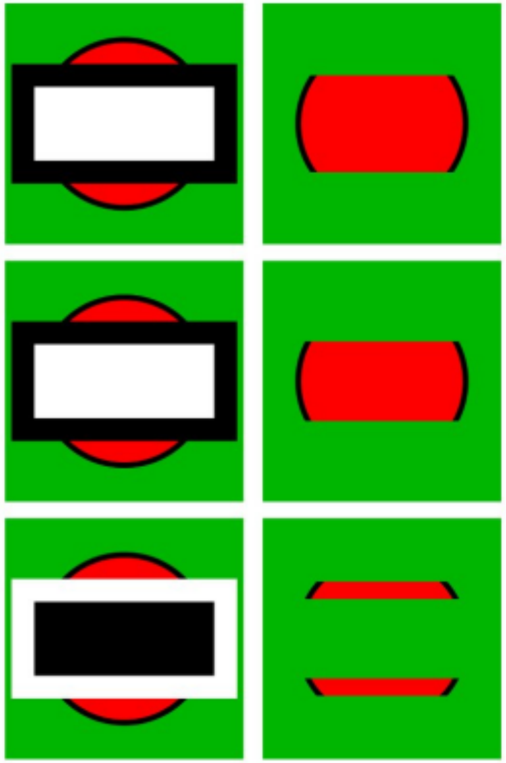
Note that anything outside the mask is clipped. In that respect, masking could even be used as an alternative to the clip paths that were introduced in the previous part of this tutorial. Draw your mask entirely in white – or in solid shades of white and black – and any parts of the masked object which lie outside the mask, or which are colored black, will be clipped. Usually I recommend using a clip path in preference to a mask if you simply want a hard clipped edge, but, as we will see later in this article, there are times when it's easier to use a mask.



You may recall that the fill and stroke of a clip path are irrelevant to the final effect on the clipped object. With masks, however, the opposite is true – the color of each individual pixel is taken into account when calculating its effect on the masked object, so fills and strokes have significance.

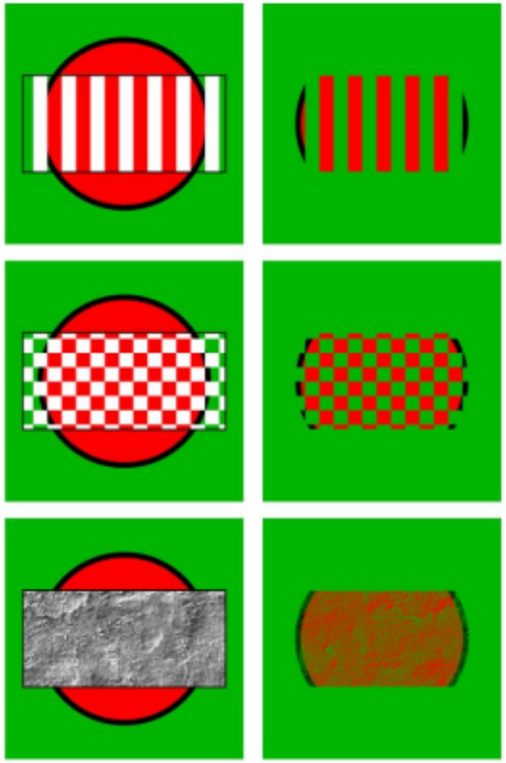
Did I say pixel? In a vector format? In practice, there are few cases where an SVG file is actually used as a purely vector format. If you want to display an SVG file on a computer screen, or print it to an inkjet or laser printer, it is ultimately converted to pixels. Masking takes place during this final output step, so although the mask may be made up of vector objects, and the item being masked is also a vector, the final result is really generated only at the point that those vectors are converted to individual pixels for display or printing.

Using a thick-stroked rectangle and strictly black or white pixels, it's easy to see the difference between a clip path and the use of a mask to simulate clipping. In the top pair of images, I've used the rectangle as a clip path, and you can see that the result is aligned with the center line of the stroke. The second and third pairs of images show the rectangle being used as a mask, with only white parts remaining and black parts becoming transparent. It's especially clear in the last pair that the stroke has an effect.



By now it should be clear that clipping and masking are similar operations, so it's not really surprising that Inkscape exposes them with the same user interface. Just as with clipping, the masking object must be on top of the object to be masked, in terms of the z-index. You then select both items and choose Set Mask from the context menu, or Object > Mask > Set from the main menu. Releasing a masked object is as simple as selecting it (it will be described as “Masked” in the status bar) then choosing Release Mask from the context menu, or Object > Mask > Release from the main menu. Just as with clip paths, it is possible to edit masks using the node tool without releasing them first, but generally there's little benefit to doing so as you can edit only the outline shape of the path, not its color.

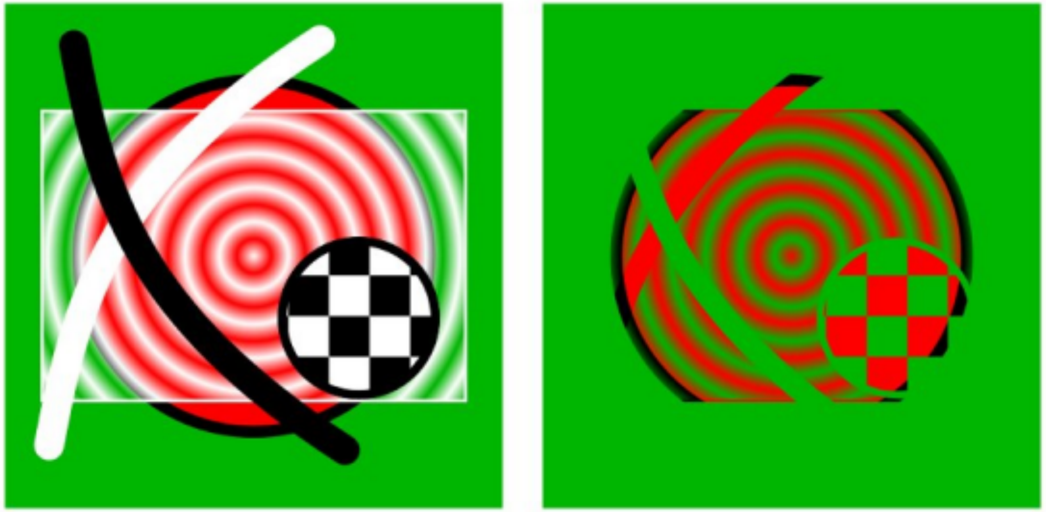
As you've seen, using a grayscale gradient as a mask gives you some control over the opacity of the masked object. You may also recall that Inkscape offers a selection of black and white patterns in its Fill & Stroke dialog, plus a few grayscale bitmap fill patterns. These may previously have seemed rather limiting, but their monochrome nature makes them perfect for use as a mask.



The checkerboard pattern is a good example of a situation where it's easier to use a mask than a clip path. This same effect could be achieved by clipping with a complex path, but it's probably not worth the extra effort. Looking closely at the checkerboard, you'll notice that the fill isn't made up of black and white squares, but rather white and transparent. Using white and transparent can sometimes make it easier to see what parts of an object you are masking: white areas will remain visible, and transparent areas will be clipped. A translucent white area will have the same effect as a shade of gray. It may seem counter-intuitive to cover the parts of the object that you want to keep, and expose the parts that will be hidden, but that's the way that masking has been defined in the SVG specification, so that's the way it works in Inkscape.

Because clip paths are defined by a boundary and a concept of what's inside and outside that boundary, it's possible to use only a single path or object to define a clipping shape. Masks, on the other hand, are concerned with only the color of pixels, so it doesn't really matter whether your mask is made up of a single object or a whole load of objects inside a group. This allows you to create complex arrangements of colors and patterns which would be impossible to reproduce using a single path.

This example uses a group made up of a few random objects as a mask. You can see that the repeating radial gradient of the rectangle, made up of a white-to-transparent transition, results in a similar pattern being visible from the masked red circle. But by adding extra black and white features to the mask group, you can “override” the pattern with explicit sections of transparency and opacity. One thing to note is the place where the black line crosses the white one: because the black line is on top, the pixels at that position are black, so that part of the masked object is transparent. It doesn't matter what's going on within the group, all that matters is the color of the final pixels. This may not be the best piece of abstract art in the world, but it would have been a lot harder to draw without using masks.



Although you can use masks to create complex arrangements of clipping and transparency, probably the most common use is simply to add “feathering” to the edges of a group so that it fades into the background. In that case all you need for your mask design is a strongly blurred white ellipse, as demonstrated in this wholly unseasonal greeting card design.

Throughout this article I've referred to masks as being grayscale images. That's not strictly true – a mask can be any combination of colors that you want. However, anything other than white, black or gray will be converted into a grayscale image before it's used as a mask. This can make it difficult to predict the exact effect of a color on the resultant opacity, so I recommend using pure grayscales when drawing your masks in almost all cases.

Mark has been using Linux since 1994, and uses Inkscape to create two webcomics, 'The Greys' and 'Monsters, Inked' which can both be found at:
http://www.peppertop.com/

SEASON'S GREETINGS