A short route to tgif pictures

Daan Reuhman*

November 20, 2002

1 What's the problem?

For starters, let's quote the `tgif` manual page:

> `Tgif` is an interactive drawing tool that allows the user to draw and manipulate objects in the X Window system.

Although after a while the program is reasonably easy (*sic*) to operate, beginners are typically somewhat puzzled by the seemingly unusual user interface. With the problems for the uninitiated in mind, this document aims to prevent unnecessary frustration and tries to speed up the learning process.

The "getting to know `tgif`" period is roughly divided into three parts. First, preliminary action is required to get the program to start without complaints. Second, we must get acquainted with the important concepts and definitions in the `tgif` world. Third, and prick up your ears, we shall draw a real picture. A more detailed breakdown of the learning time yields the following:

- getting the X Windows system to run;
- defining the so-called `meta` key;
- setting handy default values;
- starting `tgif`;
- getting acquainted with the windows;
- learning about the modes and objects;
- creating objects;

*With many thanks to William Chia-Wei Cheng for writing the software, and Jaap Haalboom for installing and making `tgif` work on the Apollos.

- typing text;
- selecting objects;
- moving objects;
- copying objects;
- deleting objects;
- grouping;
- save a picture; and
- stopping `tgif`.

2 Before starting `tgif`

`Tgif` is fully based on the X Windows graphics system. Therefore a form of X graphics server must be running on the local station. The question "how to get X running?" is clearly beyond the scope of this document, but it can certainly be answered by your friendly computer supporter.

Many of `tgif`s interactive commands can be (and are) given from the keyboard as opposed to via mouse-based menu selection only. A command is a combination a so-called modifier key and some other key. The two modifier keys are:

- `ctrl` key, shown as `^` in `tgif` documentation; and
- `meta` key, confusingly shown as `#` in `tgif` documentation.

These keys will be referred to as `ctrl` and `meta`, respectively.

Most keyboards have a standard `ctrl` key, but the `meta` key requires setup work. For Apollo workstations the `F0` function key will be defined as the `meta` key. First, create (or edit) the file `$HOME/.xmodmaprc` to read:

```
clear Mod1
add Mod1 = F10
```

Now, make the key definition effective by running `xmodmap`:

```
$ xmodmap $HOME/.xmodmaprc
```

Text in `tgif` can be drawn in many font sizes. The largest size, 34 points, is not by default present. Therefore, we have to use `xset` to extend the preferred font path:

```
$ xset fp+ /usr/lib/X11/fonts/iso_8859.1/100dpi/
```

Like other X programs, `tgif` has a range of options for which default values can be set. The following settings appear useful and may go into the file `$HOME/.Xdefaults`:

```
Tgif*Geometry: 1100x850+0+0
Tgif*InitialFont: Helvetica
Tgif*GridSystem: Metric
Tgif*InitialGrid: -1
Tgif*WhereToPrint: EPS
```

To explain a bit: the geometry gives a us a wide `tgif` screen, the initial font is made to be `helvetica`, the grid is set in centimeters with an initial grid size of 1 mm, and printing is to be done into a format suitable for LaTeX inclusion.

That ends the preparations: X Windows is running on the workstation, a definition for the `meta` key has been made and the desired values have been put into `$HOME/.Xdefaults`.

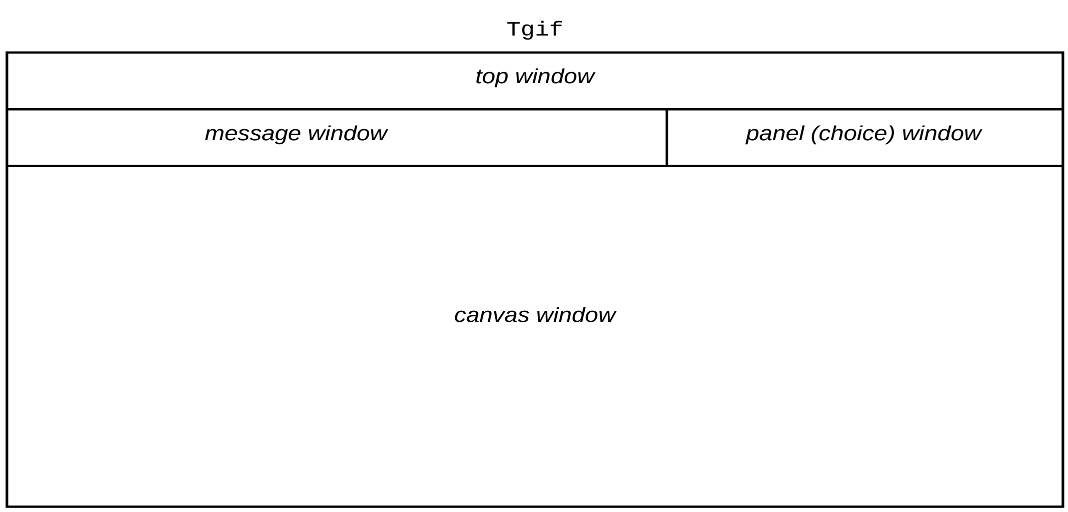
Well now, it is high time for the real thing:

```
$ tgif hello
```

Because the file `hello.obj` does not yet exist, an empty picture shows up on the screen.

3 Basic `tgif` definitions

After starting a `tgif` session, you will soon be confronted with the following screen (well, schematically it looks like this):



The *top window* shows the input file name: `hello.obj`.

The *message window*, under the top window, displays messages.

The *panel (choice) window* displays the current choice (by means of an intuitively recognizable icon) for all sorts of settable items in `tgif`, such as the type of object currently being drawn, the current font, the line width, and many more. Clicking with the left or right mouse button on an icon changes the choice.

The *canvas window* is the actual drawing area.

3.1 Modes

Before drawing anything in the canvas window, it is important to realize that `tgif` has a *select* mode and a *drawing* mode. After the program has started, it will be in select mode, as can be seen from the first icon in the choice panel. When the icon shows an arrow pointing into the upper-left direction,  , select mode is enabled.

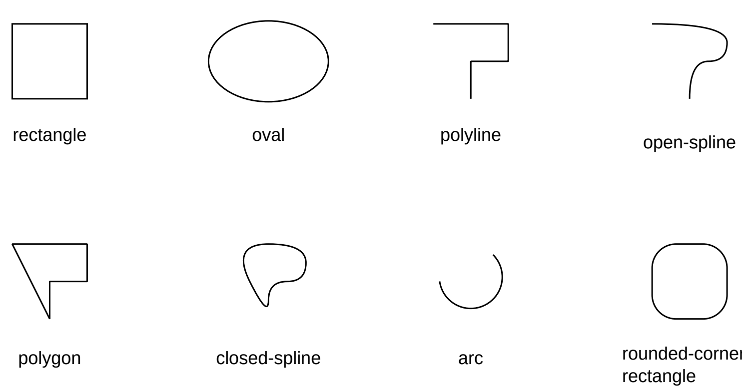
Try to remember:

> Holding down the shift key and clicking the right mouse button will change the drawing mode to select.
>
> The other way around: holding down the right mouse button to get choice menu, moving to the desired object symbol and release the right mouse button.

Switching between drawing mode and select mode is likely to be one of the most frequently performed actions when using `tgif`.

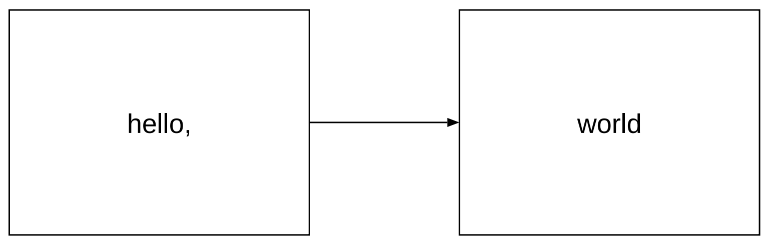
3.2 Objects

Everything drawn in `tgif`, including text, is called an object. Supported objects are:



Objects can be moved, duplicated, deleted, rotated, and flipped.

The next part of the document is devoted to the day-to-day details of operating `tgif`. To set an objective, the picture we shall construct is shown below:



Should things go wrong during the excercise, it may be difficult to recover partially from an erroneous state. Selective un-doing of actions is possible, by typing:

```
meta u
```

In other words: hold down the `meta` key and type the letter `u`. Because we will be masters of `tgif` only after the completion of the excercise, the following sequence of commands is recommended when things seem to be in a hopeless state:

```
ctrl a
ctrl x
```

In human language: select the entire picture drawn so far, and delete it.

Then, start again, if you still feel like it.

4 And now for real

Interacting with `tgif` is most conveniently done in a combination of mouse button clicking and keyboard commands. For each drawing operation described in the remainder of the document a recipe will be given formulated in all the required "button-click and key-stroke" actions.

4.1 Drawing a rectangle

Put the cursor somewhere in the canvas window so that the drawing of a rectangle can begin (for precise manoeuvring at grid points, look at the rulers showing the exact horizontal and vertical position of the cursor):

hold right mouse button to get choice menu

move to rectangle symbol on menu

release right mouse button

—`tgif` is in rectangle drawing mode now—

position cursor at grid point (1, 1)

hold and move left mouse button to (5, 4)

release left mouse button

And there is the first rectangle in `tgif`.

Note that in the current mode only rectangles can be drawn.

4.2 Typing text

Some identifying text must into the picture. As mentioned earlier, text (T) is one of the object types available. If we haven't done anything after drawing the rectangle, we can proceed as follows:

hold right mouse button to get choice menu

move to symbol on menu

release right mouse button

position cursor at grid point (2, 5)

click left mouse button

type `hello,`

And for the other string:

click left mouse button at (8, 5)

type `world`

again click left mouse button

4.3 Drawing an arrow

The screen is now filled with a rectangle and two text objects (in `tgif` speak). Let's now add an arrow. Arrows are a special case of the general polyline indicated on the choice menu by . Here is how to draw the arrow:

hold right mouse button to get choice menu

move to polyline symbol on menu

release right mouse button

position cursor at grid point (4, 5)

hold and move left mouse button to (6, 5)

release left mouse button

click right mouse button

Real polyline objects are created when repeatedly releasing and moving the left mouse button. The drawing stops only after clicking the right mouse button. However, clicking the right button does keep us in polyline drawing mode. To go back into select mode, shown as on the menu, do:

hold right mouse button to get choice menu

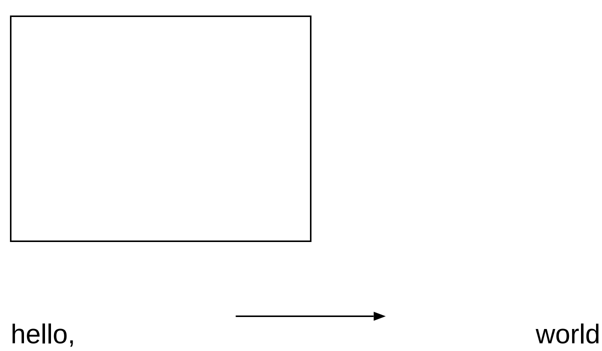
move to the select symbol on menu

release mouse button

click left mouse button

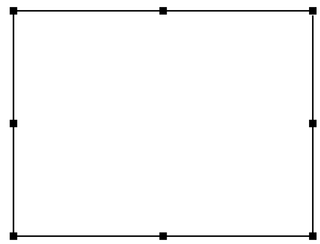
All this might seem a little bit tricky: the arrow symbol on the choice menu indicates select mode and has nothing to do with drawing nicely looking arrows.

Concluding the drawing part leaves us with an interesting picture, containing the following objects:



4.4 Selecting and moving objects

From now no objects will be drawn explicitly. Instead, the rest of the picture will be realized by copying, moving, aligning and grouping of existing objects. Therefore, it is vital to know how to select objects on the screen in order to manipulate them: *select an object by moving the cursor on it and click the left mouse button.* Selecting additional objects is done by holding down the shift key and clicking the left mouse on an object. Selected objects are highlighted by a bounding box of small rectangles, often referred to as handles:



Moving objects is performed by using the left mouse button:

select object by clicking left mouse button

hold left mouse button on *border* of object

move left mouse button, thereby moving the object

Stretching (making bigger or smaller) of existing objects is performed in much the same way:

select object by clicking left mouse button

hold left mouse button on one of the *handle*s

move left mouse button, thereby stretching the object

When multiple objects have been selected, moving and stretching applies to all of them at the same time.

4.5 Copying the rectangle

We assume that, after drawing the arc, the select mode was set again. To get a second rectangle, we shall copy the one already on the screen:

click left mouse button on rectangle border

type `ctrl d`

hold left mouse button on rectangle border

move second rectangle to grid point (1, 6)

release left mouse button

4.6 Moving the text

The strings of text both have to be moved (but not yet exactly aligned!) into a rectangle:

click left mouse button on `hello,`

hold left mouse button and move `hello,` into rectangle

Perform the same actions for the text `world` and the other rectangle.

4.7 Aligning text and rectangles

Of course, the text strings must be aligned with the centers of the rectangles. To achieve this, move the cursor to the choice window where two icons indicate the current alignment settings. The defaults are:

- L — left horizontal alignment; and
- T — top vertical alignment.

while we need:

- C — center horizontal alignment; and
- M — middle vertical alignment.

Changing the setting goes like:

> move cursor to L icon
> click left mouse button until C appears
> move cursor to T icon
> click left mouse button until M appears

Subsequently the centers of text and rectangles can be aligned with the following command sequence:

> click left mouse button on `hello,`
> hold shift and click left mouse button on left rectangle border
> type `ctrl l`

and, likewise:

> click left mouse button on `world`
> hold shift and click left mouse button on right rectangle border
> type `ctrl l`

4.8 Grouping text and rectangles

Each rectangle can now be "grouped" with its text into one object, so that future selection operations apply to the two objects as a single unit. The selection of objects is like in the alignment action discussed before, only the grouping command is different:

> click left mouse button on `hello,`
>
> hold shift and click left mouse button on left rectangle border
>
> type `ctrl g`

And perform the same actions for `world`.

4.9 Aligning arc and rectangles

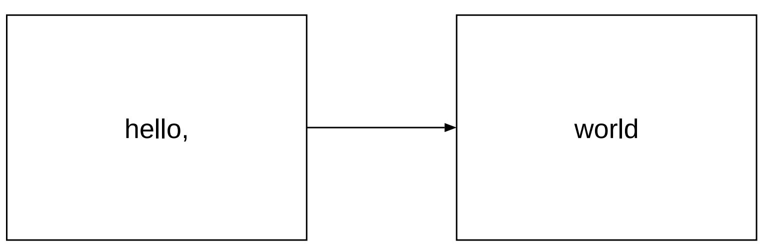
After grouping only three selectable objects remain, which have to be aligned vertically and abutted horizontally. The necessary commands are most conveniently issued from the keyboard:

> type `ctrl a`
>
> type `meta +`

for selecting and aligninmg objects vertically, and:

> type `meta _` (underscore)

for the abutment. And then we see the picture as we had intended it to be:



4.10 Saving the picture

Until now the picture has not yet been saved on disk. Typing:

> `ctrl s`

writes the file `hello.obj` containing a save copy of your picture.

A PostScript file must be created for the inclusion of the picture in (e.g. LaTeX) documents. The command:

```
ctrl p
```

creates the file `hello.eps` with a `%% BoundingBox` statement containing the exact height and width of the drawn picture.

> move cursor to LaTeX icon in choice window
> click left mouse button until appears
> type `ctrl p`

thereby writing the file `hello.ps`.

4.11 Terminate `tgif`

And so we have come to the end of the `tgif` introduction. A last:

> ctrl q

stops the program leaving us free to design other beautiful pictures.