Tmdoc LaTeX Template

Detlef Groth

November 9, 2020

Contents

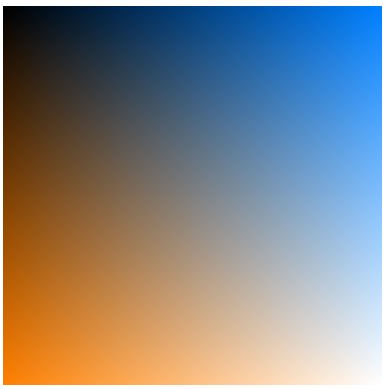
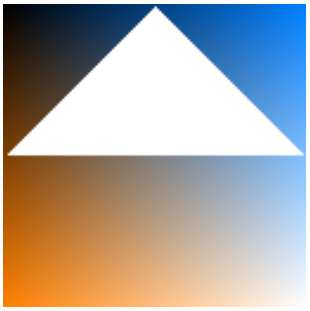
1 Color codes

Some notes about the color code:

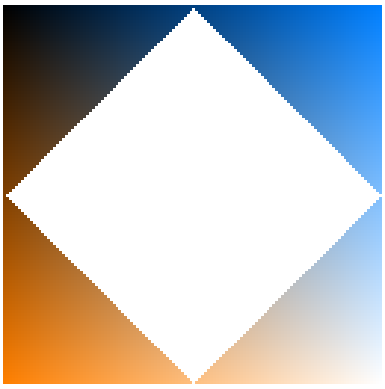
- red foreground: code written into the LaTeX document but not evaluated
- grey background: code evaluated by Tcl if *echo=true* (default)
- blue background: Tcl output, the last statement of the code block is return automatically.
- red background: error messages of the Tcl interpreter is shown

2 Tcl code blocks

Code block start with triple codes at the line start:

```
```{tcl}
set x 1
```
```

Output:

```
set x 1
```

```
==> 1
```

Showing the code can be hidden using *echo=false* as option for the code chunk:

```
```{tcl echo=false}
set x 1
```
```

```
==> 1
```

The output can be as well hidden using the option *results=hide*.

```
```{tcl echo=false,results=hide}
set x 3
```
```

Nothing will be displayed, the value of x is changed invisible. Alternatively you can return at the end of the code block an empty string.

```
```{tcl}
set x 4
return {}
```
```

```
set x 4
return {}
```

3 Tcl code blocks with puts

Puts works as well:

```
```{tcl echo=false}
set x 1
foreach i [list 1 2 3] {
   puts {Hello World!}
}
set x {} ;# supress return
```
```

```
Hello World!
Hello World!
Hello World!
```

4 Tcl inline codes

In standard text outside of code blocks as well short Tcl statements can be executed using single backticks: like this `` `tcl code` ``.

Here two examples in the following lines:

Tcl/Tk: `` `tcl set tcl_patchLevel` `` produces ==> Tcl/Tk: 8.6.8

and:

x is `` `tcl set x 5` `` produces ==> x is 5 ...

5 Images with Tcl

Images can be generated with Tcl in various ways:

- with the canvas widget together with canvas::snap library from tklib
- with tcl.gd
- with other libraries (see for more at the end of this document)

Here an example were we create an image using a basic Tk installation which was programmed by Maurice Bredelet, code at `https://wiki.tcl-lang.org/page/Generating+a+color+image`:

```
```{tcl}
package require Tk
set size 128
... see below
img put $pixels
img write test.png -format png
```
```

```
package require Tk
set size 128
# create image
image create photo img -width $size -height $size
# compute colors
set pixels {}
set k [expr {128.0 / $size}]
for {set i 0} {$i < $size} {incr i} {
  set row {}
  for {set j 0} {$j < $size} {incr j} {
      set R [expr {int($i * 2 * $k)}]
      set G [expr {int(($i + $j) * $k)}]
      set B [expr {int($j * 2 * $k)}]
      lappend row [format #%02x%02x%02x $R $G $B]
   }
  lappend pixels $row
}
img put $pixels
img write test.png -format png
```

You can then use the LaTeX command *includegraphics* to place the figure.



To avoid this manual writing of the file to the file system with a self selected name, we can give a code chunk option *fig=true* together with a label. That is close to the approach R's well known Sweave and Knitr packages are using. To get this working, we have to create procedure with the name *figure* before. Within this procedure we could use for instance the Tcl package *canvas::snap* to select items from a canvas and save them to the file system. This *figure* procedure must have as first argument a filename argument which is the filename wherein the file is saved. The extension png is automatically appended. To overwrite the png extension you have to use the chunk option *ext*, for instance *ext=pdf* will generate a filename with a pdf extension. It does not matter where this *figure* procedure is declared it just must be there if you use the option *fig=true*: You can as well redefine this procedure at any later tpoint to switch to an other graphical object or library.

```
```{tcl label=testimg,fig=true,fig.width=4cm}
proc figure {filename args} {
    img write $filename -format png
}
# set transparency
set s2 [expr {$size / 2}]
... see below
    }
}
```
```

```
proc figure {filename args} {
    img write $filename -format png
}
# set transparency
set s2 [expr {$size / 2}]
for {set i 0} {$i < $s2} {incr i} {
    set i2 [expr {$s2 - $i}]
    for {set j 0} {$j < $size} {incr j} {
        if {$j > $i2 && $j < $size - $i2} {
            img transparency set $j $i 1
        }
    }
}
```



If you use as well the option *include=false* the automatic placement is avoided and you can again use the *includegraphics* command of LaTeX to place the figure as you like. The filename you have to use in the LaTeX code consists of the basename of the LateX file and the label of the code chunk. So the filename for the following chunk is *tmdoc-template-testimg2.png*

```
```{tcl label=testimg3,fig=true,include=false}
for {} {$i < $size} {incr i} {
    set i2 [expr {$i - $s2}]
...
}
```
```

```
for { } {$i < $size} {incr i} {
    set i2 [expr {$i - $s2}]
    for {set j 0} {$j < $size} {incr j} {
        if {$j > $i2 && $j < $size - $i2} {
            img transparency set $j $i 1
        }
    }
}
```



6 Summary

- triple ticks code block ```` ```tcl option=value ... code ... ``` ````
- single ticks with normal text `` `tcl code` ``

Chode chunk options:

- *label=labelname* - label for chunk, defaults to *chunk-nn* where nn is the automatically generated chunk number, default chunk-nn
- *echo=true/false* - should the Tcl code from the code block be shown, default: true
- *results=show/hide* - should the return value or puts output be shown or hidden, default: show
- *fig=true/false* - should procedure figure called at the end of the block, default: false

- *fig.width=12cm* - width for figure if *include=true*, default: 12cm
- *include=true/false* - should the figure generated be included, default: true
- *ext=png/pdf* - file extension of the generated image file if *fig=true*, default: png

Some Tcl packages to create figures are:

- canvas::snap from Tklib `https://core.tcl-lang.org/tklib/home`
- tcl.gd `https://github.com/flightaware/tcl.gd`
- TclMagic `http://tclmagick.sourceforge.net/`
- pdf4tcl `http://pdf4tcl.sourceforge.net/`
- tclcairo `https://wiki.tcl-lang.org/page/tclcairo`

This document was created on Mon Nov 09 08:00:30 CET 2020 using Tcl/Tk 8.6.8 with package *tmdoc::tmdoc* 0.4 on operating system 5.8.17-100.fc31.x86_64.

To create process the PDF document document you should do the following coommands:

```
mkdir build
cd build
tclsh ../../tmdoc.tcl ../tmdoc-template.Tnw \
    --outfile tmdoc-template.tex
pdflatex tmdoc-template.tex
pdflatex tmdoc-template.tex
```