7. Videomixer

The videomixer element makes it possible to mix different videostreams together. Here is a CLI example:

```
$ gst-launch-0.10 filesrc location=tvlogo.png ! pngdec ! alphacolor ! ffmpegcolorspace ! videobox border-alpha=0 alpha=0.5 top=-20 lef
        videomixer name=mix ! ffmpegcolorspace ! autovideosink videotestsrc ! video/x-raw-yuv, width=320, height=240 ! mix.
```

You have to make a PNG image (100x100 px) to be able to run it. With the videobox element you can move the image around and add more alpha channels.

NEW COMPETITION! I can't figure out how to make a transparent alpha channel for the PNG in the pipeline above so if anyone makes it work the prize from example 3.2 will be given to this winner instead as the other one passed. TIA

UPDATE! An anonymous dude solved the problem with an alphacolor element, many thanks man.

In the next example we take the now working Mpeg2-Player from example 3.2 and pump it with the stuff shown above.

Example 7.1

```
#!/usr/bin/env python

import sys, os
import pygtk, gtk, gobject
import pygst
pygst.require("0.10")
import gst

class GTK_Main:

        def __init__(self):
                window = gtk.Window(gtk.WINDOW_TOPLEVEL)
                window.set_title("Mpeg2-Player")
                window.set_default_size(500, 400)
                window.connect("destroy", gtk.main_quit, "WM destroy")
                vbox = gtk.VBox()
                window.add(vbox)
                hbox = gtk.HBox()
                vbox.pack_start(hbox, False)
                self.entry = gtk.Entry()
                hbox.add(self.entry)
                self.button = gtk.Button("Start")
                hbox.pack_start(self.button, False)
                self.button.connect("clicked", self.start_stop)
                self.movie_window = gtk.DrawingArea()
                vbox.add(self.movie_window)
                window.show_all()

                self.player = gst.Pipeline("player")
                source = gst.element_factory_make("filesrc", "file-source")
                demuxer = gst.element_factory_make("mpegdemux", "demuxer")
                demuxer.connect("pad-added", self.demuxer_callback)
                self.video_decoder = gst.element_factory_make("mpeg2dec", "video-decoder")
                png_decoder = gst.element_factory_make("pngdec", "png-decoder")
                png_source = gst.element_factory_make("filesrc", "png-source")
                png_source.set_property("location", "tvlogo.png")
                mixer = gst.element_factory_make("videomixer", "mixer")
                self.audio_decoder = gst.element_factory_make("mad", "audio-decoder")
                audioconv = gst.element_factory_make("audioconvert", "converter")
                audiosink = gst.element_factory_make("autoaudiosink", "audio-output")
                videosink = gst.element_factory_make("autovideosink", "video-output")
                self.queuea = gst.element_factory_make("queue", "queuea")
                self.queuev = gst.element_factory_make("queue", "queuev")
                ffmpeg1 = gst.element_factory_make("ffmpegcolorspace", "ffmpeg1")
                ffmpeg2 = gst.element_factory_make("ffmpegcolorspace", "ffmpeg2")
                ffmpeg3 = gst.element_factory_make("ffmpegcolorspace", "ffmpeg3")
                videobox = gst.element_factory_make("videobox", "videobox")
                alphacolor = gst.element_factory_make("alphacolor", "alphacolor")

                self.player.add(source, demuxer, self.video_decoder, png_decoder, png_source, mixer,
                        self.audio_decoder, audioconv, audiosink, videosink, self.queuea, self.queuev,
                        ffmpeg1, ffmpeg2, ffmpeg3, videobox, alphacolor)
                gst.element_link_many(source, demuxer)
                gst.element_link_many(self.queuev, self.video_decoder, ffmpeg1, mixer, ffmpeg2, videosink)
                gst.element_link_many(png_source, png_decoder, alphacolor, ffmpeg3, videobox, mixer)
                gst.element_link_many(self.queuea, self.audio_decoder, audioconv, audiosink)

                bus = self.player.get_bus()
                bus.add_signal_watch()
                bus.enable_sync_message_emission()
                bus.connect("message", self.on_message)
                bus.connect("sync-message::element", self.on_sync_message)

                videobox.set_property("border-alpha", 0)
                videobox.set_property("alpha", 0.5)
                videobox.set_property("left", -10)
                videobox.set_property("top", -10)

        def start_stop(self, w):
                if self.button.get_label() == "Start":
```

```
71                      filepath = self.entry.get_text()
72                      if os.path.isfile(filepath):
73                              self.button.set_label("Stop")
74                              self.player.get_by_name("file-source").set_property("location", filepath)
75                              self.player.set_state(gst.STATE_PLAYING)
76              else:
77                      self.player.set_state(gst.STATE_NULL)
78                      self.button.set_label("Start")
79
80      def on_message(self, bus, message):
81              t = message.type
82              if t == gst.MESSAGE_EOS:
83                      self.player.set_state(gst.STATE_NULL)
84                      self.button.set_label("Start")
85              elif t == gst.MESSAGE_ERROR:
86                      err, debug = message.parse_error()
87                      print "Error: %s" % err, debug
88                      self.player.set_state(gst.STATE_NULL)
89                      self.button.set_label("Start")
90
91      def on_sync_message(self, bus, message):
92              if message.structure is None:
93                      return
94              message_name = message.structure.get_name()
95              if message_name == "prepare-xwindow-id":
96                      imagesink = message.src
97                      imagesink.set_property("force-aspect-ratio", True)
98                      imagesink.set_xwindow_id(self.movie_window.window.xid)
99
100     def demuxer_callback(self, demuxer, pad):
101             if pad.get_property("template").name_template == "video_%02d":
102                     queuev_pad = self.queuev.get_pad("sink")
103                     pad.link(queuev_pad)
104             elif pad.get_property("template").name_template == "audio_%02d":
105                     queuea_pad = self.queuea.get_pad("sink")
106                     pad.link(queuea_pad)
107
108 GTK_Main()
109 gtk.gdk.threads_init()
110 gtk.main()
```

Prev
6. Capabilities
Home
Next
8. Webcam Viewer