GStreamer
Desktop Multimedia Framework

Introduction to GStreamer & KDE

By Christian F.K. Schaller and Scott Wheeler

Who are we

- Christian F.K. Schaller
  - Long time Linux user and advocate. Most recently involved in GStreamer and librsvg project. Just left his job at Oracle to start working for free software company Fluendo doing GStreamer and GStreamer based products full time
- Scott Wheeler
  - Linux specialist who has been employed in the SAP LinuxLab in Walldorf, Germany since 2002. He has been active in several areas of KDE development in the last several years, most notably multimedia. He is the author of JuK, FlashKard, KSig, TagLib and assorted hacks throughout the rest of KDE

What is GStreamer?

- A pipeline based multimedia framework
- Provides an abstraction for audio and video playback, recording, streaming, editing and metadata
- Plugin based with around 80 plugins distributed presently covering a wide variety of input and output mechanisms, codec support and effects

What is GStreamer <u>not</u>?

- GStreamer is not a sound server and does not include or have a preferred sound server
- GStreamer is not a playback only solution (like MPlayer, Xine, etc.)
- GStreamer does not require any libraries from either of the two major Linux desktops; integration is done through plugins

History of GStreamer

- Original design based on research project at Portland University
- First code on Sourceforge January 2000
- The goal was to make a pipeline based streaming media framework
- Focus mostly on transcoding, server backend processing and embedded uses

Recent developments

- Higher focus on desktop issues as more and more desktop oriented developers have gotten involved
- Addition of related functions like abstraction of sound system/card mixer
- Desktop friendly error handling introduced
- Mac OS X and Windows ports
- A lot of work and focus on clearing as many legal issues as possible out of the way

Technical details

- C based core – using glib for object orientation
- Language bindings for Ruby, Python, Perl, Mono, KDE style C++ and Guile
- All of the core using LGPL license
- Small core rest is handled using plugins

What is a pipeline?

- A pipeline is composed of *elements*
- A pipeline starts with a source – this can be a file, a KIO slave, or a physical device
- A number of elements are connected to process this data – decoders, effects, mixing, etc.
- The output is sent to a sink – a sink can be a file, a sound server, a display or a device

Quick & easy development

- Focus on making API as easy to use as possible for application developers
- Pipelines can be defined using XML
- Easy testing and prototyping with command line tool gst-launch
  - gst-launch filesrc location=example.ogg ! oggdemux name=mux { mux. ! queue ! vorbisdec ! audioconvert ! osssink } { mux. ! theoradec ! ffcolorspace ! xvimagesink }ffcolorspace ! xvimagesink }

What GStreamer offers KDE

- A ready complete multimedia framework
- Large and active development community behind it
- Cross-platform support to match Qt's cross platform abilities (Linux/Unix/Mac OSX/Windows)
- LGPL licensing
- A shared implementation between GNOME and KDE reducing redundant work being done and increasing development speed of both desktops

Current status

- Doing 0.8.x release series – API/ABI stable release series. All stable GStreamer series are parallel installable
- Current focus on improving playback to support at least as many 'weird/broken' files as Xine and MPlayer
- Improving cross-platform capabilities
- Having 'best-in-class' support for free formats

The competition

- We feel GStreamer has a unique position currently
- Potential competitors are either:
  - Too narrow in their functionality
  - Unacceptably licensed
  - Too immature
  - Lacking mindshare

GStreamer in the real world

- Already the shipping framework of GNOME with many applications using it – Totem, Rhytmbox, Sound Juicer, Marlin, SoundScrape, GNOME Mixer, GNOME Sound Recorder etc.
- Commercial support and development through Fluendo (http://www.fluendo.com)
- First stage projects underway at major companies

KDE GStreamer usage

- Already two usable applications with GStreamer support – JuK and AmaroK
- KDE style bindings (currently just 0.6 API)
- Experimental integration work with existing KDE multimedia applications

Open Questions - KDE integration

- Official media framework for KDE 4.x?
- Porting/replacing of official KDE bundled media applications to GStreamer
- Moving KDE mixers to Gstreamer cross platform and cross sound framework mixing interface

- (Oh, and this conference currently streamed using Streaming server made using GStreamer framework by Fluendo)

KDE Bindings

- Current bindings provide a low-level wrapper for the GStreamer API and a high level binding for *simple* playback
- C++ / Qt / QObject based
- GStreamer already contains in its plugins distribution integration for an aRts *sink* and a KIO *source*.

Simple Example of KDE::GST::Play

- Simple example instantiates a player
- Connects the streamEnd() and timeTick() slots so that actions may be inserted at those points
- Sets the source location
- Starts playback

```
KDE::GSTPlay::Play *player = new Play(Play::PIPE_AUDIO_BUFFER_THREADED, this, "Play");

connect(player, SIGNAL(streamEnd()), this, SLOT(finished()));

connect(player, SIGNAL(timeTick(long long)), this, SLOT(tick(long long)));

player->setLocation(?/home/user/foo.ogg?);

player->setState(Element::STATE_PLAYING);
```

Questions?

- Any questions for Christian or Scott?
- More information to be found on www.gstreamer.net