

HTTP Live Streaming (HLS)

Sanchayan Maity

Who

- ▶ Who am I?

Who

- ▶ Who am I?
 - ▶ Embedded Systems background

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ [asymptotic](#)

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ [asymptotic](#)
 - ▶ Open source consulting firm based out of Bangalore and Toronto

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ **asymptotic**
 - ▶ Open source consulting firm based out of Bangalore and Toronto
 - ▶ Work on low level systems software centred around multimedia

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ **asymptotic**
 - ▶ Open source consulting firm based out of Bangalore and Toronto
 - ▶ Work on low level systems software centred around multimedia
 - ▶ GStreamer, PipeWire, PulseAudio

Who

- ▶ Who am I?
 - ▶ Embedded Systems background
 - ▶ Prefer C, Haskell and Rust
 - ▶ Organize and speak at Rust and Haskell meet-ups in Bangalore
- ▶ Work?
 - ▶ Software Engineer @ **asymptotic**
 - ▶ Open source consulting firm based out of Bangalore and Toronto
 - ▶ Work on low level systems software centred around multimedia
 - ▶ GStreamer, PipeWire, PulseAudio
 - ▶ Language Polyglots

Open source contributions

- ▶ GStreamer

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire
- ▶ PulseAudio

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire
- ▶ PulseAudio
- ▶ Linux

Open source contributions

- ▶ GStreamer
- ▶ gst-plugins-rs
- ▶ PipeWire
- ▶ PulseAudio
- ▶ Linux
- ▶ u-boot

Agenda

- ▶ Whirlwind tour of GStreamer

Agenda

- ▶ Whirlwind tour of GStreamer
- ▶ What's HTTP Live Streaming (HLS)

Agenda

- ▶ Whirlwind tour of GStreamer
- ▶ What's HTTP Live Streaming (HLS)
- ▶ HLS implementation

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)
 - ▶ GstLAL (gravitational wave data analysis)

GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)
 - ▶ GstLAL (gravitational wave data analysis)
 - ▶ Rygel (DLNA streaming server and renderer)

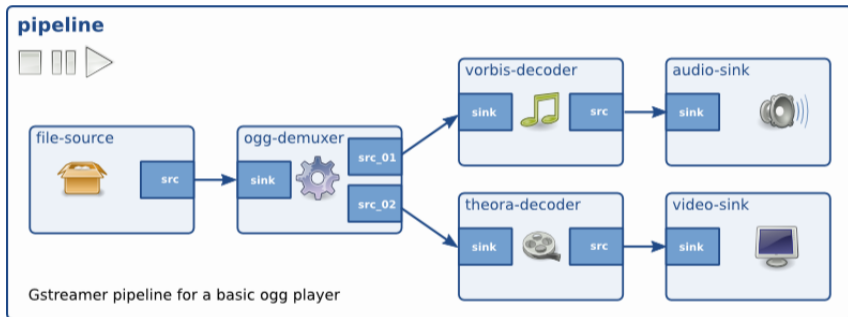
GStreamer

- ▶ Multiplatform Pipeline based multimedia framework
- ▶ Bindings for various languages
- ▶ Allows building complex media processing workflows
- ▶ Some applications
 - ▶ PiTiVi (Video Editor)
 - ▶ amaroK, Banshee, Clementine (audio players)
 - ▶ Empathy (VOIP and video conferencing)
 - ▶ GstLAL (gravitational wave data analysis)
 - ▶ Rygel (DLNA streaming server and renderer)
 - ▶ Totem (movie player for the GNOME desktop)

Simple pipeline

```
gst-launch-1.0 videotestsrc ! autovideosink  
gst-launch-1.0 audiotestsrc ! autoaudiosink
```

Media pipeline¹



¹Dynamic Pipelines

Hls

- ▶ HTTP-based adaptive bit-rate streaming communications protocol

Hls

- ▶ HTTP-based adaptive bit-rate streaming communications protocol
- ▶ Developed by Apple and released in 2009

Hls

- ▶ HTTP-based adaptive bit-rate streaming communications protocol
- ▶ Developed by Apple and released in 2009
- ▶ Standardised in [RFC 8216](#)

Hls

- ▶ HTTP-based adaptive bit-rate streaming communications protocol
- ▶ Developed by Apple and released in 2009
- ▶ Standardised in [RFC 8216](#)
- ▶ HTTP traffic, unlike UDP-based protocols such as RTP

Hls

- ▶ HTTP-based adaptive bit-rate streaming communications protocol
- ▶ Developed by Apple and released in 2009
- ▶ Standardised in [RFC 8216](#)
- ▶ HTTP traffic, unlike UDP-based protocols such as RTP
- ▶ Content can be offered from conventional HTTP servers

Hls

- ▶ HTTP-based adaptive bit-rate streaming communications protocol
- ▶ Developed by Apple and released in 2009
- ▶ Standardised in [RFC 8216](#)
- ▶ HTTP traffic, unlike UDP-based protocols such as RTP
- ▶ Content can be offered from conventional HTTP servers
- ▶ Delivered over widely available HTTP-based content delivery networks

Playlist²

```
#EXTM3U
#EXT-X-TARGETDURATION:10
#EXT-X-VERSION:3
#EXTINF:9.009,
http://media.example.com/first.ts
#EXTINF:9.009,
http://media.example.com/second.ts
#EXTINF:3.003,
http://media.example.com/third.ts
#EXT-X-ENDLIST
```

²RFC 8216

Master playlist³

```
#EXTM3U
#EXT-X-STREAM-INF:BANDWIDTH=1280000,AVERAGE-BANDWIDTH=1000000
http://example.com/low.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=2560000,AVERAGE-BANDWIDTH=2000000
http://example.com/mid.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=7680000,AVERAGE-BANDWIDTH=6000000
http://example.com/hi.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=65000,CODECS="mp4a.40.5"
http://example.com/audio-only.m3u8
```

³RFC 8216

Master playlist⁴

```
#EXTM3U
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="aac",NAME="English", \
  DEFAULT=YES,AUTOSELECT=YES,LANGUAGE="en", \
  URI="main/english-audio.m3u8"
#EXT-X-MEDIA:TYPE=AUDIO,GROUP-ID="aac",NAME="Deutsch", \
  DEFAULT=NO,AUTOSELECT=YES,LANGUAGE="de", \
  URI="main/german-audio.m3u8"
#EXT-X-STREAM-INF:BANDWIDTH=1280000,CODECS="...",AUDIO="aac"
low/video-only.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=2560000,CODECS="...",AUDIO="aac"
mid/video-only.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=7680000,CODECS="...",AUDIO="aac"
hi/video-only.m3u8
#EXT-X-STREAM-INF:BANDWIDTH=65000,CODECS="mp4a.40.5",AUDIO="aac"
main/english-audio.m3u8
```

⁴RFC 8216

Implementation

- ▶ New GStreamer plugin

Implementation

- ▶ New GStreamer plugin
- ▶ Written in Rust

Implementation

- ▶ New GStreamer plugin
- ▶ Written in Rust
- ▶ Uses `m3u8-rs`

Implementation

- ▶ New GStreamer plugin
- ▶ Written in Rust
- ▶ Uses `m3u8-rs`
- ▶ Open MR:
https://gitlab.freedesktop.org/gstreamer/gst-plugins-rs/-/merge_requests/1515

Demo

Using

▶ [videojs](#)



Demo

Using

- ▶ `videojs`
 - ▶ `python3 -m http.server`
-

Demo

Using

- ▶ `videojs`
- ▶ `python3 -m http.server`
- ▶ Sample GStreamer Rust Code⁵

⁵`HLS sample app`

References

- ▶ RFC 8216

References

- ▶ RFC 8216
- ▶ HLS in depth

References

- ▶ [RFC 8216](#)
- ▶ [HLS in depth](#)
- ▶ [HTTP Live Streaming - A Practical Guide](#)

References

- ▶ [RFC 8216](#)
- ▶ [HLS in depth](#)
- ▶ [HTTP Live Streaming - A Practical Guide](#)
- ▶ [HTTP Live Streaming - Wikipedia](#)

Questions

- ▶ Reach out on

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net
 - ▶ Mastodon: sanchayanmaity.com

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net
 - ▶ Mastodon: [sanchayanmaity.com](https://mastodon.social/@sanchayanmaity)
 - ▶ Telegram: <https://t.me/SanchayanMaity>

Questions

- ▶ Reach out on
 - ▶ Email: sanchayan@sanchayanmaity.net
 - ▶ Mastodon: [sanchayanmaity.com](https://mastodon.social/@sanchayanmaity)
 - ▶ Telegram: <https://t.me/SanchayanMaity>
 - ▶ Blog: sanchayanmaity.net