常见的多媒体框架

多媒体框架是一种软件框架，它在电脑上处理媒体数据并通过网络进行传播。一个良好的多媒体框架提供了直观的API和模块化的架构，这使得它易于添加对新的音频、视频和容器格式以及传输协议的支持。多媒体框架不仅可以支持媒体播放器和音视频编辑器等程序，还可以用于编译视频会议程序、媒体转换器及其他多媒体工具。

多媒体框架通常包括以下几个层级：

- 编解码层：负责对音视频数据进行编码和解码。
- 封装层：处理音视频数据的压缩和封装。
- 协议层：管理网络数据的传输和通信协议。
- 应用层：整合整个流程，并进行统筹管理。

常见的多媒体框架包括：FFmpeg、GStreamer、DirectShow(微软)、AVFoundation(苹果)、OpenMax等。

1.FFmpeg

FFmpeg是一个非常强大的开源多媒体处理框架，它提供了一系列用于处理音频、视频和多媒体流的工具和库。它也是最流行且应用最广泛的框架！

官方网址：https://ffmpeg.org/

FFmpeg 的主要特点和功能：

1. **编解码器支持**: FFmpeg 支持众多音视频编解码器，包括常见的 H.264、H.265、AAC、MP3 等，也支持一些不常见的编解码器。
2. **格式支持**: 它支持多种多媒体格式的解析和封装，包括 AVI、MP4、MKV、FLV、MOV 等。
3. **转码和处理**: FFmpeg 可以进行音视频的转码、裁剪、拼接、水印添加等处理操作，使其在不同格式、分辨率和编码方式之间进行转换。
4. **流媒体处理**: 它支持从摄像头、文件或网络流等源接收多媒体流，并能进行实时处理和转发，用于流媒体直播和视频会议等场景。
5. **滤镜和特效**: FFmpeg 提供了丰富的滤镜和特效，可以实现图像处理、色彩调整、模糊、锐化等效果。
6. **音频处理**: 它能够进行音频的分割、合并、音量调整、混音等操作。
7. **跨平台性**: FFmpeg 是跨平台的，可以在 Windows、MacOS、Linux 等操作系统上运行。
8. **开源和免费**: FFmpeg 是完全开源的，可以免费使用，并且具有活跃的社区支持和持续的更新和改进。

1.1 安装FFmpeg

在 Ubuntu 上安装 FFmpeg 可以通过包管理器 `apt` 来完成。以下是在 Ubuntu 上安装 FFmpeg 的步骤：

1. **更新软件包列表**：

```
sudo apt update
```

1. **安装 FFmpeg**：

```
sudo apt install ffmpeg -y
```

安装完成后，你可以通过以下命令验证是否成功安装了 FFmpeg：

```
ffmpeg -version
```

这将显示安装的 FFmpeg 版本信息。

如需安装FFmpeg拓展开发包可执行如下所示命令：

> sudo apt install libavcodec-dev -y
> sudo apt install libavformat-dev -y
> sudo apt install libavcodec-extra -y

FFmpeg开发文档：https://ffmpeg.org/ffmpeg.html

1.2 使用命令行执行ffmpeg

转换视频格式：

```
ffmpeg -i input.mp4 output.avi
```

这个命令将输入的 MP4 文件转换为 AVI 格式。

合并视频和音频：

```
ffmpeg -i video.mp4 -i audio.mp3 -c:v copy -c:a aac -strict experimental
output.mp4
```

这个命令将一个视频文件和一个音频文件合并为一个 MP4 文件，视频流不变，音频流重新编码为 AAC 格式。

调整视频大小：

```
ffmpeg -i input.mp4 -vf scale=1280:720 output.mp4
```

这个命令将输入的 MP4 文件调整为 1280x720 分辨率的输出。

改变视频帧率：

```
ffmpeg -i input.mp4 -r 24 output.mp4
```

这个命令将输入的 MP4 文件的帧率改为 24 帧每秒。

提取视频中的帧：

```
ffmpeg -i input.mp4 -vf "select=eq(n\,100)" -vsync vfr output.png
```

这个命令将从输入的 MP4 文件中提取第 100 帧，并将其保存为 PNG 图像文件。

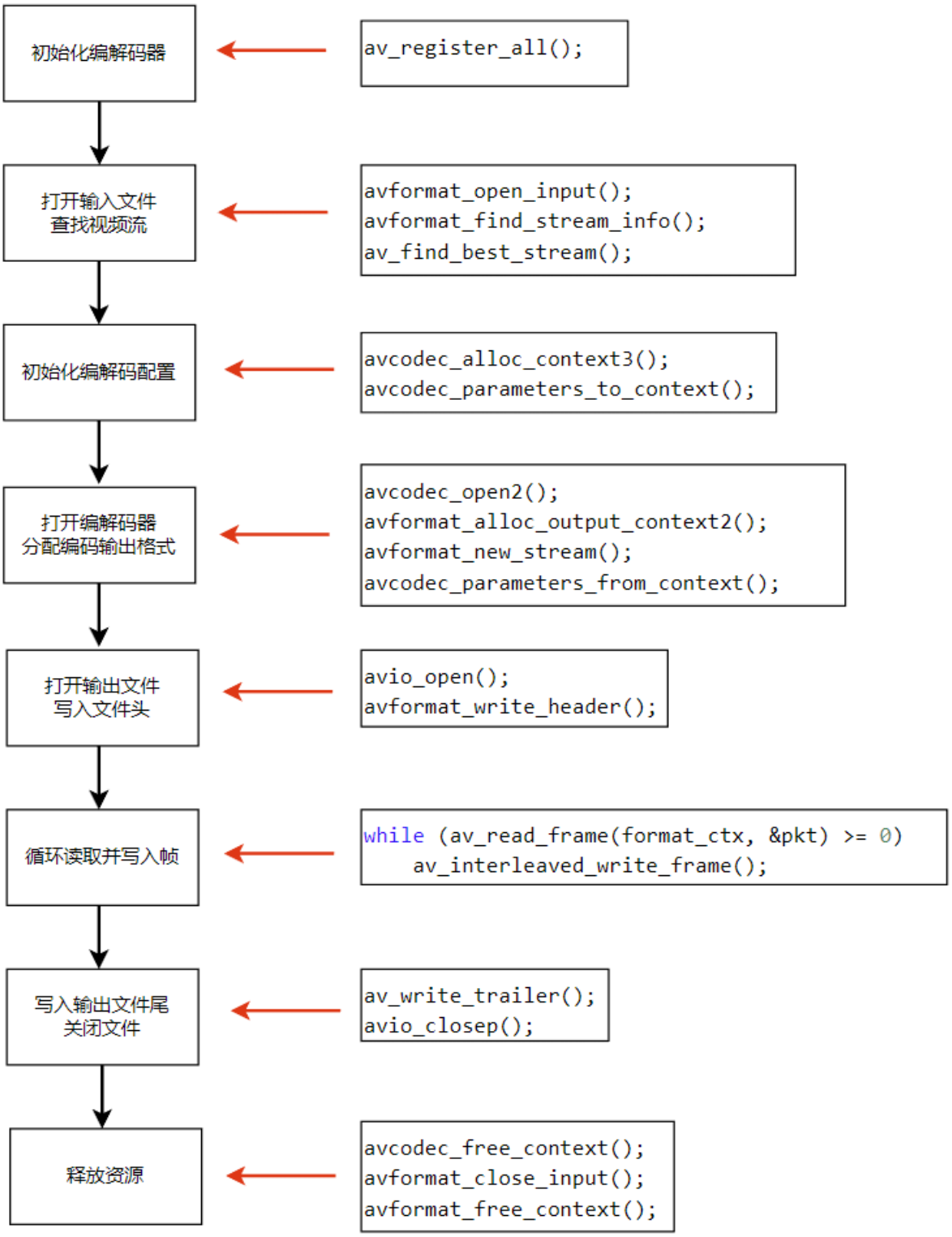
添加水印：

```
ffmpeg -i input.mp4 -i test.jpg -filter_complex "overlay=10:10" output.mp4
```

这个命令将一个水印图像叠加到输入的 MP4 文件的左上角。

1.3 代码实现视频格式转换

C++代码实现MP4视频格式转换AVI，程序主要内容如下图所示：



编译运行程序后，可以看到当前文件夹下从MP4转换后的AVI文件.

```
$ ./convert input.mp4 output.avi
$ ls
convert  input.mp4  main.cpp  main.o  Makefile  output.avi
```

使用ffplay播放视频

```
$ ffplay output.avi
```

1.4 FFmpeg核心库介绍

FFmpeg库	介绍
libavutil	功能：提供了一系列用于简化程序编写的工具函数和数据结构，是其他库的基础。它包含了数学、数据结构、内存管理、日志系统等方面的功能。 作用：为其他 FFmpeg 库提供了公共的辅助功能和数据结构，如内存管理、数据类型转换、日志记录等。
libavcodec	功能：实现了音频和视频的编码和解码功能，支持众多的音视频编解码器。 作用：允许将各种不同格式的音频和视频进行解码（读取）、编码（写入）操作，实现音视频的转码、提取、压缩和解压等功能。
libavformat	功能：用于封装和解封装音频和视频流，支持多种音视频格式的读写。 作用：实现了对音频和视频容器格式（如MP4、AVI、FLV、MKV等）的解析和生成，使得 FFmpeg 能够读取和写入不同格式的音视频文件。
libavdevice	功能：提供了对音视频设备的访问接口，如摄像头、麦克风等。 作用：允许直接从音视频设备中捕获数据或向设备写入数据，用于实时采集、录制和播放音视频。
libavfilter	功能：实现了音视频过滤器框架，用于对音视频流进行各种效果、滤镜和处理操作。 作用：允许在音视频流中应用各种滤镜、效果和处理算法，如调整音量、添加水印、去噪等，以实现音视频的处理和增强。
libavresample	功能：提供了音频重采样功能，用于在不同的采样率和声道数之间进行转换。 作用：允许在不同的音频采样格式之间进行转换和重采样，以适应不同设备和要求。
libswscale	功能：实现了视频像素格式转换和缩放功能，用于处理视频帧的像素格式和尺寸。 作用：允许在不同的视频像素格式之间进行转换，以及对视频帧进行缩放、裁剪和处理，用于实现视频的格式转换和缩放。
libswresample	功能：提供了音频采样格式转换和重采样功能，类似于 libavresample，但更专注于音频领域。 作用：允许在不同的音频采样格式和采样率之间进行转换和重采样，以适应不同的音频设备和要求。
libpostproc	功能：提供了视频后期处理功能，包括一些视频效果的处理算法。 作用：允许对视频进行后期处理，如去除隔行、去噪等，以提升视频质量和效果。

2.GStreamer

GStreamer 是一个功能强大的开源多媒体框架，用于创建、处理和播放音频和视频流。官方网址：[GStreamer: open source multimedia framework]

1. **模块化架构**: GStreamer 的设计是基于模块化的架构，可以根据需要添加或移除各种插件和元件，从而实现灵活的功能扩展和定制。
2. **跨平台性**: GStreamer 可以在多种操作系统上运行，包括 Linux、Windows、macOS 等。
3. **丰富的插件支持**: GStreamer 提供了大量的插件，用于处理各种多媒体格式、编解码器、滤镜、特效等，可以满足各种多媒体处理需求。
4. **流式处理**: GStreamer 支持流式处理，可以处理实时音视频流，适用于流媒体直播、视频会议等场景。
5. **音视频编解码支持**: 它支持多种常见的音视频编解码器，包括 H.264、H.265、AAC、MP3 等，也支持一些不常见的编解码器。
6. **容器格式支持**: GStreamer 支持多种多媒体容器格式的解析和封装，包括 AVI、MP4、MKV、FLV、MOV 等。
7. **图形界面和命令行工具**: GStreamer 提供了图形界面和命令行工具，用于配置和管理多媒体处理流程。
8. **多语言支持**: GStreamer 可以通过各种语言的绑定进行使用，包括 C、C++、Python、Java 等。

GStreamer的核心是基于管道（Pipeline）的概念，这意味着你可以将多个处理步骤（称为元素或Element）连接起来，以实现复杂的媒体处理任务。

2.1 安装GStreamer

1. **更新软件包列表**:

   在终端中执行以下命令，确保系统的软件包列表是最新的：

   ```
   sudo apt update
   ```

2. **安装 GStreamer**:

   安装 GStreamer 的基本运行时库和插件。你可以根据需要选择不同的插件包，如 `gstreamer1.0-plugins-base`、`gstreamer1.0-plugins-good`、`gstreamer1.0-plugins-bad` 和 `gstreamer1.0-plugins-ugly`。

   ```
   sudo apt-get install libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev
   libgstreamer-plugins-bad1.0-dev gstreamer1.0-plugins-base gstreamer1.0-
   plugins-good gstreamer1.0-plugins-bad gstreamer1.0-plugins-ugly gstreamer1.0-
   libav gstreamer1.0-tools gstreamer1.0-x gstreamer1.0-alsa gstreamer1.0-gl
   gstreamer1.0-gtk3 gstreamer1.0-qt5 gstreamer1.0-pulseaudio -y
   ```

   这些插件包括了各种常见的音视频编解码器、文件格式支持以及其他的功能扩展。

3. **安装额外的 GStreamer 插件**（可选）：

   如果你需要更多特定功能的插件，你可以根据需要安装额外的插件包。例如，如果你需要使用 GStreamer 的 Python 绑定，可以安装 `gir1.2-gst-plugins-base-1.0` 包。

   ```
   sudo apt install gir1.2-gst-plugins-base-1.0
   ```

4. **验证安装**：

   安装完成后，你可以通过运行以下命令来验证 GStreamer 是否正确安装：

   ```
   gst-inspect-1.0 --version
   ```

   这将显示 GStreamer 版本信息。

GStreamer开发文档：https://gstreamer.freedesktop.org/documentation

2.2 使用命令行执行

转换视频格式：

```
gst-launch-1.0 -e filesrc location=input.mp4 ! qtdemux name=demux \
    demux.video_0 ! queue ! decodebin ! videoconvert ! \
    avenc_mpeg4 ! matroskamux name=mux \
    demux.audio_0 ! queue ! decodebin ! audioconvert ! \
    audioresample ! audioconvert ! mux. \
    mux. ! filesink location=output.mkv
```

这个命令将输入的 MP4 文件转换为 MKV格式。

合并视频和音频：

```
gst-launch-1.0 -e filesrc location=video.mp4 ! qtdemux name=demux \
demux.video_0 ! queue ! h264parse ! mux. \
filesrc location=audio.mp3 ! decodebin ! audioconvert ! voaacenc ! aacparse ! 
mux. \
qtmux name=mux ! filesink location=output.mp4
```

这个命令将一个视频文件和一个音频文件合并为一个 MP4 文件，视频流不变，音频流重新编码为 AAC 格式。

调整视频大小：

```
gst-launch-1.0 filesrc location=input.mp4 ! decodebin ! videoscale ! video/x-
raw,width=1280,height=720 ! videoconvert ! x264enc ! mp4mux ! filesink 
location=output.mp4
```

这个命令将输入的 MP4 文件调整为 1280x720 分辨率的输出。

改变视频帧率：

```
gst-launch-1.0 filesrc location=input.mp4 ! decodebin ! videorate ! video/x-
raw,framerate=24/1 ! videoconvert ! x264enc ! mp4mux ! filesink 
location=output.mp4
```

这个命令将输入的 MP4 文件的帧率改为 24 帧每秒。

提取视频中的帧：

```
gst-launch-1.0 filesrc location=input.mp4 ! qtdemux ! decodebin ! videorate !
video/x-raw,framerate=1/1 ! videoconvert ! pngenc ! filesink location=output.png
```

这个命令将从输入的 MP4 文件中提取视频中的图像，并保存为 PNG 图像文件。

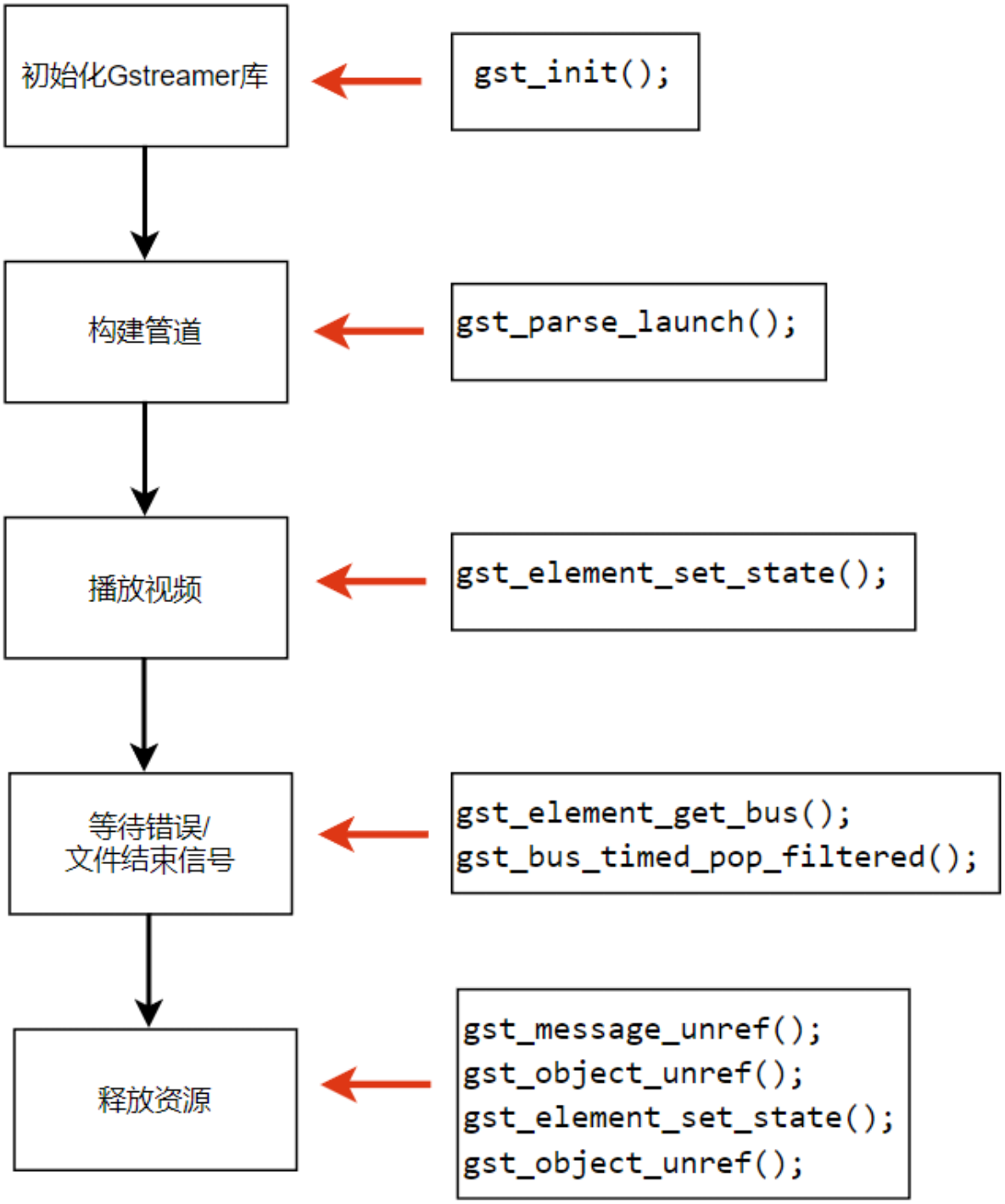
视频叠加文字:

```
gst-launch-1.0 filesrc location=input.mp4 ! decodebin ! videoconvert !
textoverlay text="Hello, World!" valignment=top halignment=left ! x264enc !
mp4mux ! filesink location=output.mp4
```

这个命令将从输入的 MP4 视频上叠加文字并将结果保存到输出文件中。

2.3 第一个GStreamer 应用程序

C代码实现MP4文件的播放，程序主要内容如下图所示:

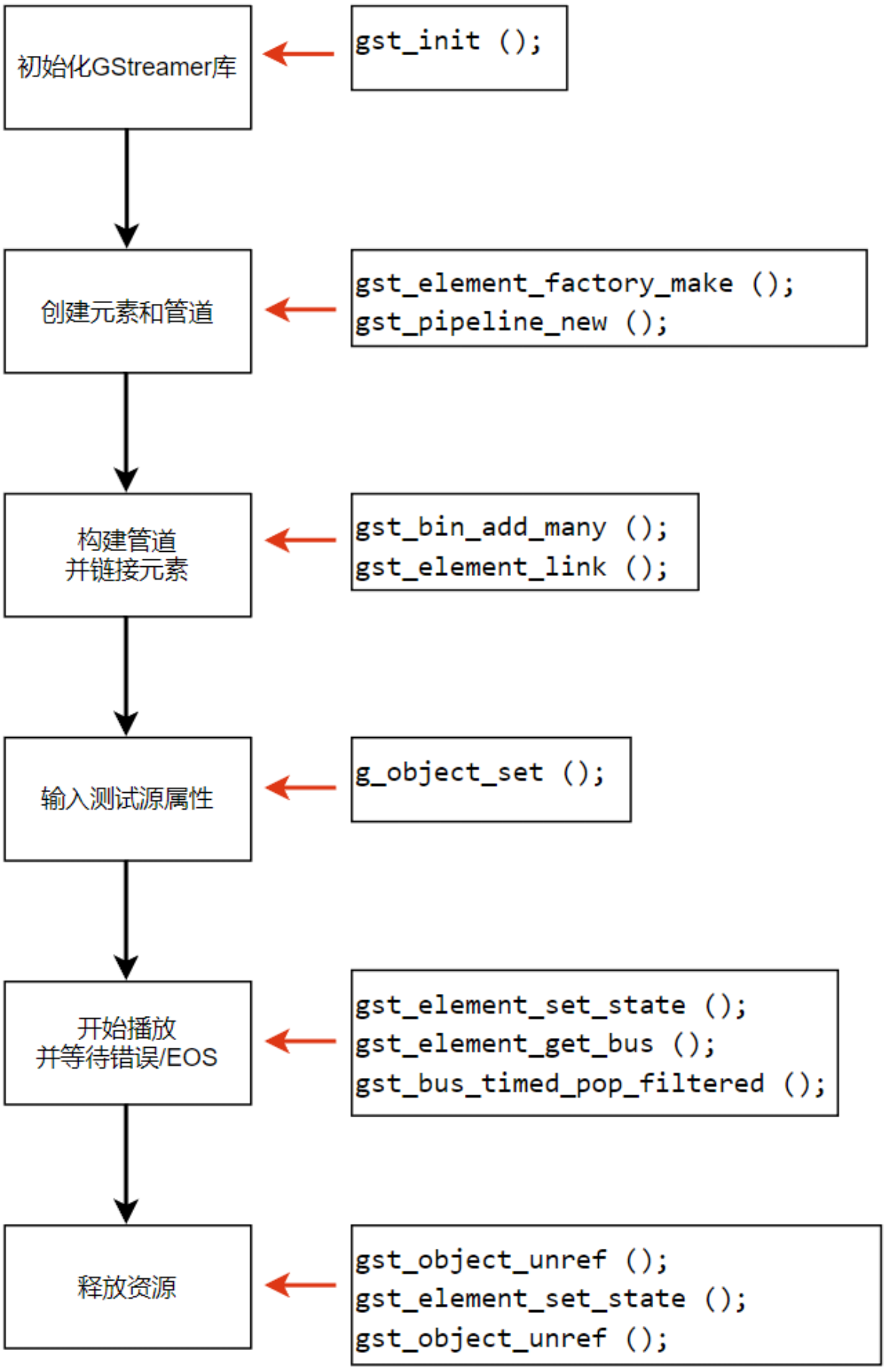


编译运行程序后，可以看到程序会读取指定路径下的文件并弹出一个窗口播放视频：

```
$ ls
basic-tutorial input.mp4 main.c
$ ./basic-tutoria
```

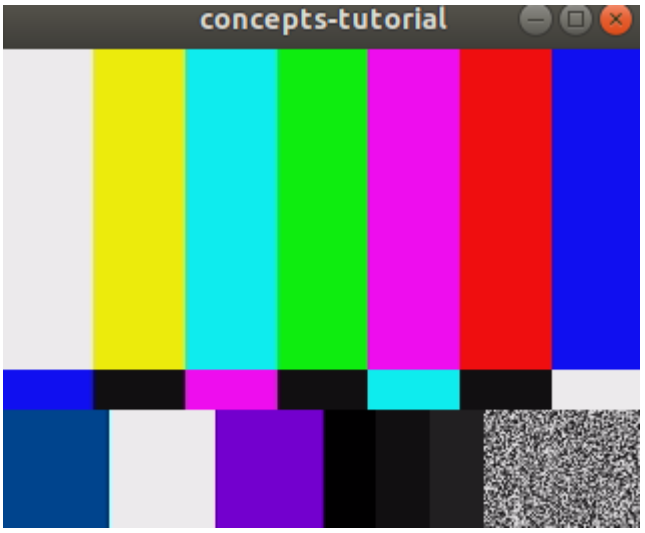
2.4 创建并连接管道

C代码实现创建元素并将元素相互连接，程序主要内容如下图所示：



编译运行程序后，可以看到程序会读取指定路径下的文件并弹出一个窗口播放默认视频：

```
$ ls
concepts-tutorial  main.c
$ ./concepts-tutorial
```



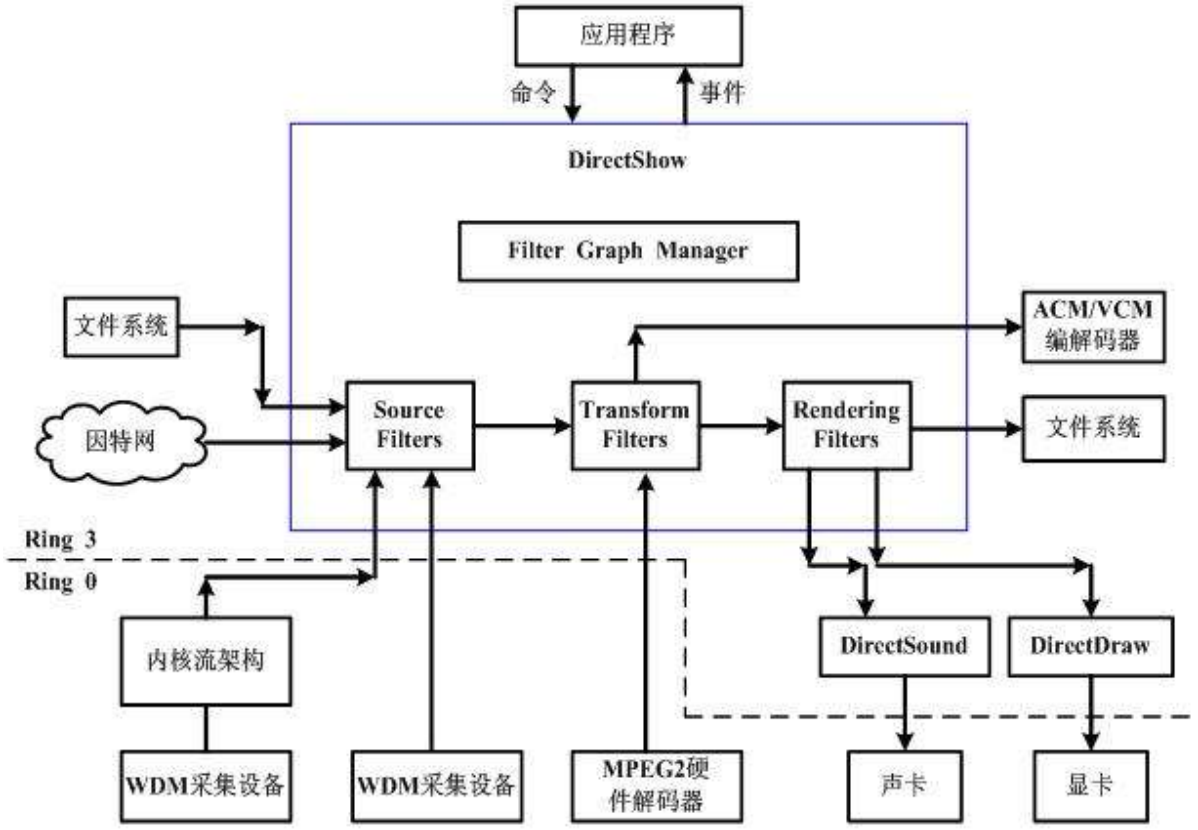
3.DirectShow

Microsoft® DirectShow® 是 Microsoft Windows® 平台上流媒体的体系结构。 DirectShow 提供多媒体流的高质量捕获和播放。 它支持多种格式，包括高级系统格式 (ASF) 、电影专家组 (MPEG) 、Audio-Video交错 (AVI) 、MPEG 音频层 3 (MP3) 和 WAV 声音文件。 它支持基于 Windows 驱动程序模型 (WDM) 或视频从数字和模拟设备捕获。 它会自动检测并使用视频和音频加速硬件（如果可用），但也支持没有加速硬件的系统。

3.1 架构

按照功能来分，Filter 大致分为三类：Source Filters、Transform Filters 和 Rendering Filters。

- Source Filters 主要负责取得数据，数据源可以是文件、因特网、或者计算机里的采集卡、数字摄像机等，然后将数据往下传输；
- Transform Fitlers 主要负责数据的格式转换、传输；
- Rendering Filtes 主要负责数据的最终去向，我们可以将数据送给声卡、显卡进行多媒体的演示，也可以输出到文件进行存储。



在 DirectShow 系统上，我们看到的，即是我们的应用程序（Application）。应用程序要按照一定的意图建立起相应的 Filter Graph，然后通过 Filter Graph Manager 来控制整个的数据处理过程。DirectShow 能在 Filter Graph 运行的时候接收到各种事件，并通过消息的方式发送到我们的应用程序。这样，就实现了应用程序与 DirectShow 系统之间的交互。

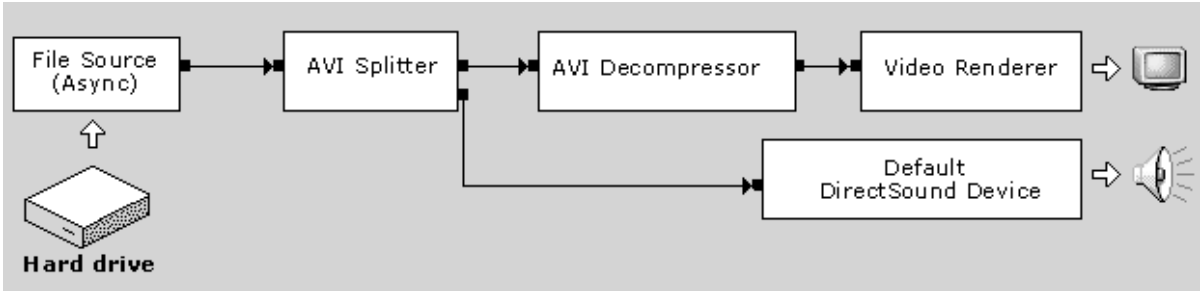
现对于面向 Windows 7 或更高版本的 DirectX，是集成在Windows SDK里，感兴趣的同学可自行尝试安装与学习。

Windows SDK下载地址：[Windows SDK和模拟器存档 | Microsoft Developer]

DirectShow开发文档：[DirectShow - Win32 apps | Microsoft Learn]

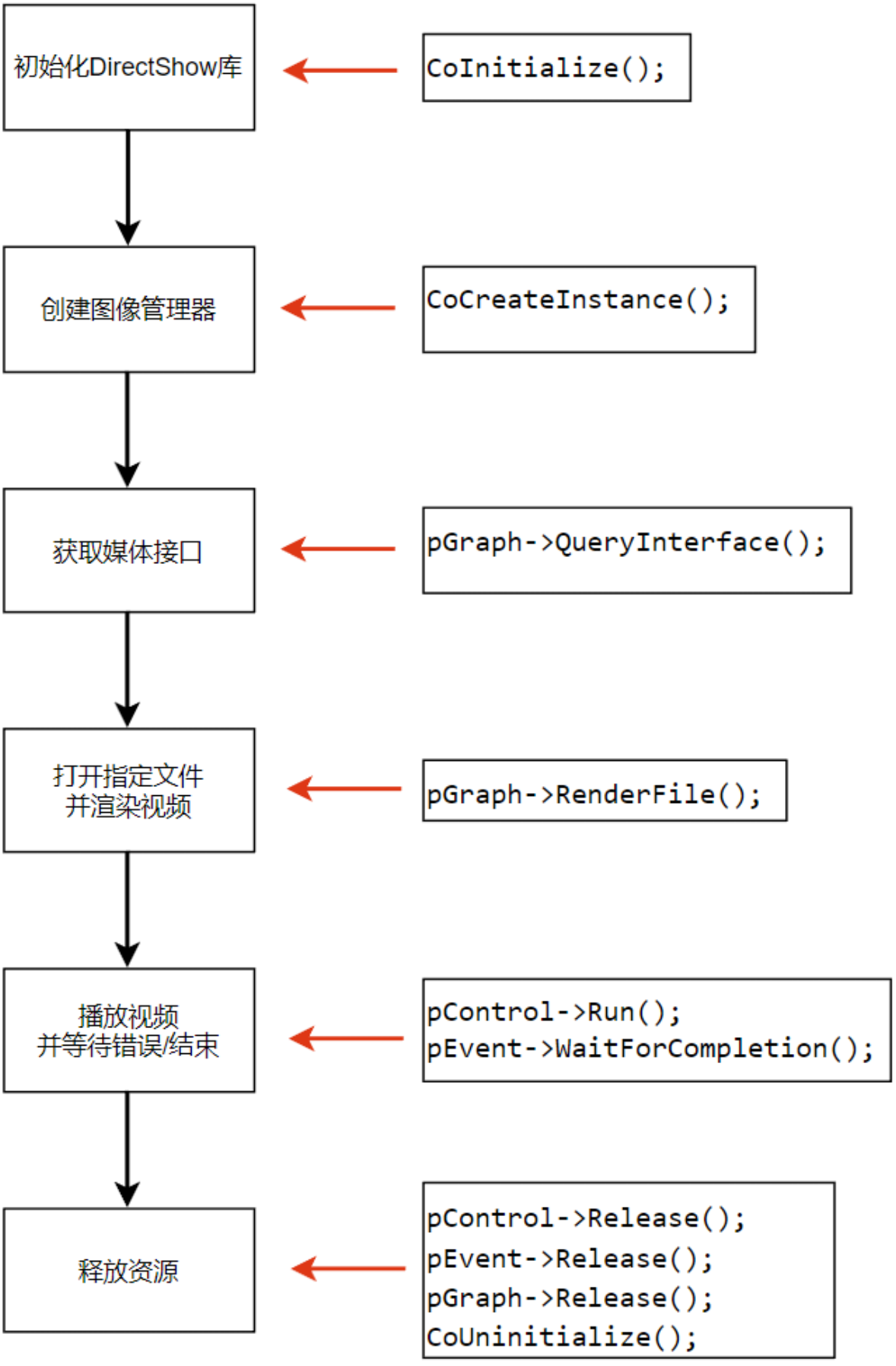
3.2 分析AVI视频播放程序

在 DirectShow 中，应用程序通过将筛选器链连接在一起来执行任何任务，以便一个筛选器的输出成为另一个筛选器的输入。 一组连接的筛选器称为 *筛选器图*。 例如，下图显示了用于播放 AVI 文件的筛选器图。



代码地址：[如何播放文件 - Win32 apps | Microsoft Learn]

代码实现AVI文件的播放，程序主要内容如下图所示：



上面的程序演示了DirectShow的基本用法，包括初始化COM库（DirectShow 库）、创建过滤器图形、播放视频文件以及资源的释放。

4.AVFoundation

AVFoundation 是苹果开发的一个全功能框架，用于在iOS、macOS、watchOS和tvOS上处理基于时间的音视频媒体。使用AVFoundation，您可以轻松播放、创建和编辑QuickTime电影和MPEG-4文件，播放HLS流，并在您的应用中构建强大的媒体功能。

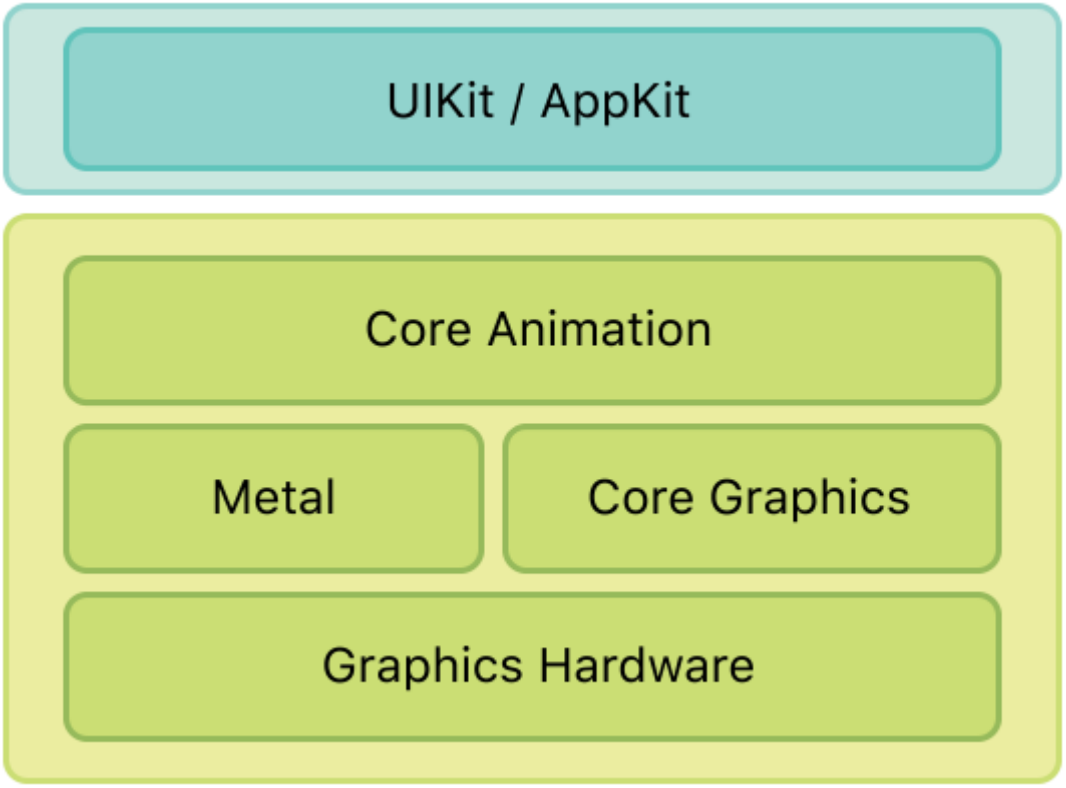
AVFoundation

它是Objective-C/Swift接口的一部分，允许开发者在详细级别上处理基于时间的音视频数据。例如，您可以使用AVFoundation来检查、创建、编辑或重新编码媒体文件。它还允许从设备获取输入流，并在实时捕捉和回放过程中操作视频。

AVFoundation 官方网址：[AVFoundation Overview - Apple Developer]

4.1 核心框架

Core Animation 是 iOS 和 OS X 上提供的图形渲染和动画基础架构，可用于对 App 的视图和其他视觉元素进行动画处理。使用 Core Animation，绘制动画每一帧所需的大部分工作都为您完成。您所要做的就是配置一些动画参数（例如起点和终点）并告诉 Core Animation 开始。Core Animation 将完成剩下的工作，将大部分实际绘图工作交给板载图形硬件以加速渲染。这种自动图形加速可实现高帧率和流畅的动画，而不会增加 CPU 负担并降低应用程序速度。

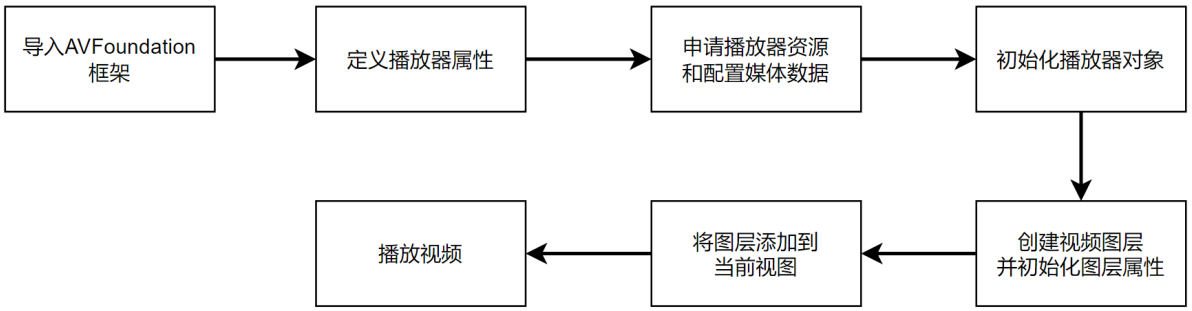


如果您使用AVFoundation进行开发，可以从Mac App Store下载并安装最新版本的Xcode，感兴趣的同学可自行尝试安装与学习

4.2 分析MOV视频播放程序

代码地址：[AVFoundationSimplePlayer-iOS：使用 AVFoundation 播放媒体 (apple.com)]

代码实现MOV文件的播放，程序主要内容如下图所示：



AVFoundation框架通过创建和配置`AVPlayer`（播放器）及其关联的`AVPlayerLayer`（视频图层）来播放MOV文件，实现了媒体资源的加载、管理和用户界面展示的整合。

5.OpenMax

OpenMAX 是一个由 Khronos Group 制定的开放标准，旨在提供一个统一的接口，用于访问多媒体处理功能，如音频、视频和图形。它可以用于各种嵌入式系统，包括移动设备、数字电视和汽车娱乐系统等。

请关注下一章节的内容。。。