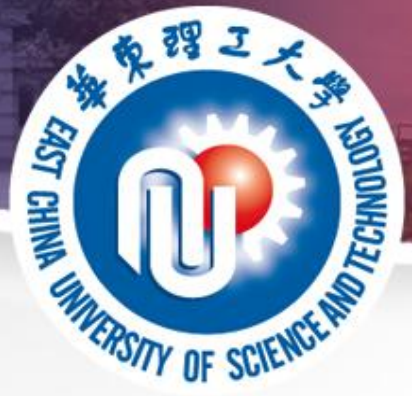



# 第13章 嵌入式GUI及应用程序设计

华东理工大学计算机系

罗 飞

Content

# 嵌入式GUI设计概述

- 嵌入式GUI简介
- 嵌入式GUI设计需求
- 嵌入式GUI设计原则
- 主流嵌入式GUI简介

# 嵌入式GUI简介

- GUI是Graphical User Interface的简称，即图形用户界面
- 一种结合计算机科学、美学、心理学、行为学，及各商业领域需求分析的人机系统工程，强调人—机—环境三者作为一个系统进行总体设计
- 嵌入式GUI就是嵌入式产品的屏幕视觉体验和互动操作部分

# 嵌入式GUI特点

- GUI是当今计算机发展的重大成就之一，他方便了非专业用户，避免记忆大量的命令，取而代之的是可以通过窗口、菜单、按键等方式来方便地进行操作
- GUI的出现是PC应用的一个分水岭
- 嵌入式GUI具有以下基本特点：轻型、占用资源少、高性能、高可靠性、便于移植、可配置。

# 嵌入式GUI分类

- 与操作系统结合的GUI
- 外挂GUI平台
- 简单GUI

# 嵌入式GUI设计内容

- 一般针对特定的硬件设备或环境，嵌入式系统都会设计不同的用户图形界面系统GUI，从而达到与硬件完美接合的效果。
  - 硬件设计，通过LCD控制器把LCD显示器和开发系统连接起来
  - 驱动程序设计，驱动硬件，嵌入式GUI系统，为上层程序设计提供图形函数库
  - 用户界面程序设计，提供图形化程序设计

# 嵌入式GUI设计需求

- 提供桌面和窗口管理功能
- 提供各种窗口组件
- 提供各种图形操作
- 支持基本的输入输出设备
- 提供资源管理的功能

# 嵌入式GUI设计原则

- 可移植性好
- 较高的稳定性和可靠性
- 系统开销较少
- 可裁剪性和配置性好

# 主流嵌入式GUI简介（1）

- Qt/E：许多基于Qt的X Window程序可以非常方便地移植Qt/Embedded版本上
- MiniGUI：提供一个轻量级的图形用户界面支持系统，比较适合工控领域的应用
- Microwindows：提供了类似X Windows的客户端/服务器体系结构，并提供了相对完善的图形功能，包括一些高级的功能
- Tiny-X：在小容量内存的环境下运行，非常适合用作嵌入式linux的GUI系统，因此TinyX作为Xfree86的子集，性能和稳定性都不容怀疑

# 主流嵌入式GUI简介（2）

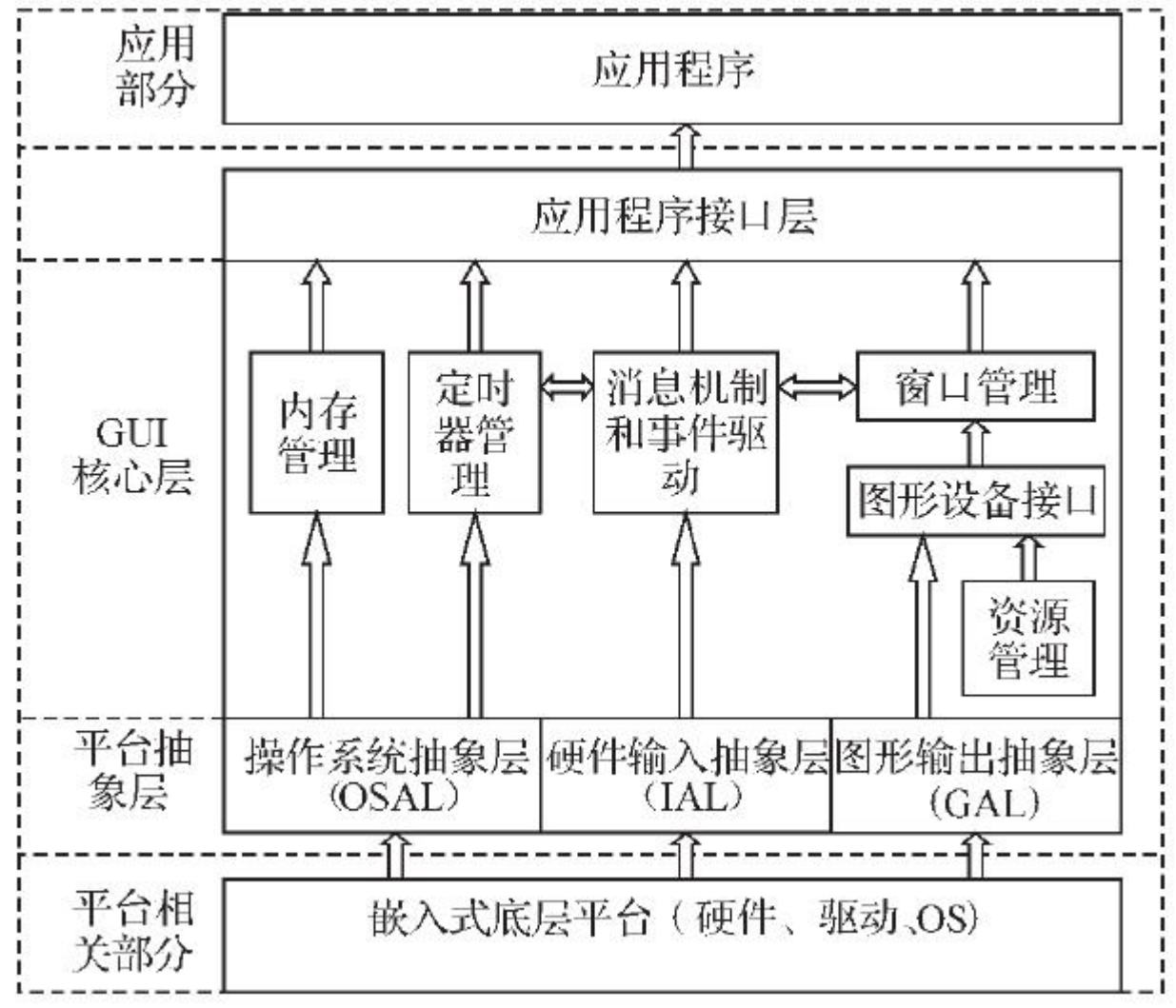
- Palm：完全为Palm产品设计和研发，一度普占据了90%的PDA市场的份额
- WinCE：Microsoft针对嵌入式产品的一套模块化设计的操作系统
- Android：Google主导的开放手机联盟开发开源移动操作系统，来支撑其嵌入式平台
- iOS：苹果公司开发的手持设备操作系统，也是以Darwin为基础的
- 鸿蒙

Content

# 嵌入式GUI体系结构设计-分层设计

- 嵌入式GUI体系结构设计
- 抽象层
- 核心层
- 应用程序接口层

# 嵌入式GUI体系结构设计



# 抽象层

- 操作系统抽象层：提供与具体操作系统相关的统一接口，与操作系统交互
- 硬件输入抽象层：为触摸屏、键盘等设备的输入提供一组统一接口，实现硬件输入
- 图形输出抽象层：与显示设备交互，把GUI所需要显示的数据发送给硬件设备

# 核心层（1）

- 图形设备接口：完成点、线、多边形、填充多边形等基本操作，这些操作与硬件无关，在内存中完成绘制
- 资源管理：GUI使用的图片和字体库等资源的管理
- 窗口管理：包含：主窗口、子窗口和控件三级窗口管理

# 核心层（2）

- 消息管理：包含监视键盘、触摸屏的事件，转换成系统消息后，发给相应的应用程序，再根据事件做相应的处理动作
- 内存管理：申请一块连续的共享内存, 用链表把此块内存管理起来,当遇到内存申请时首先在共享内存中申请, 当共享内存不够时再从全局内存中申请

# 应用程序接口

- 供各种GUI对象( 窗口、控件) 的应用编程接口以及图形绘制接口

Content

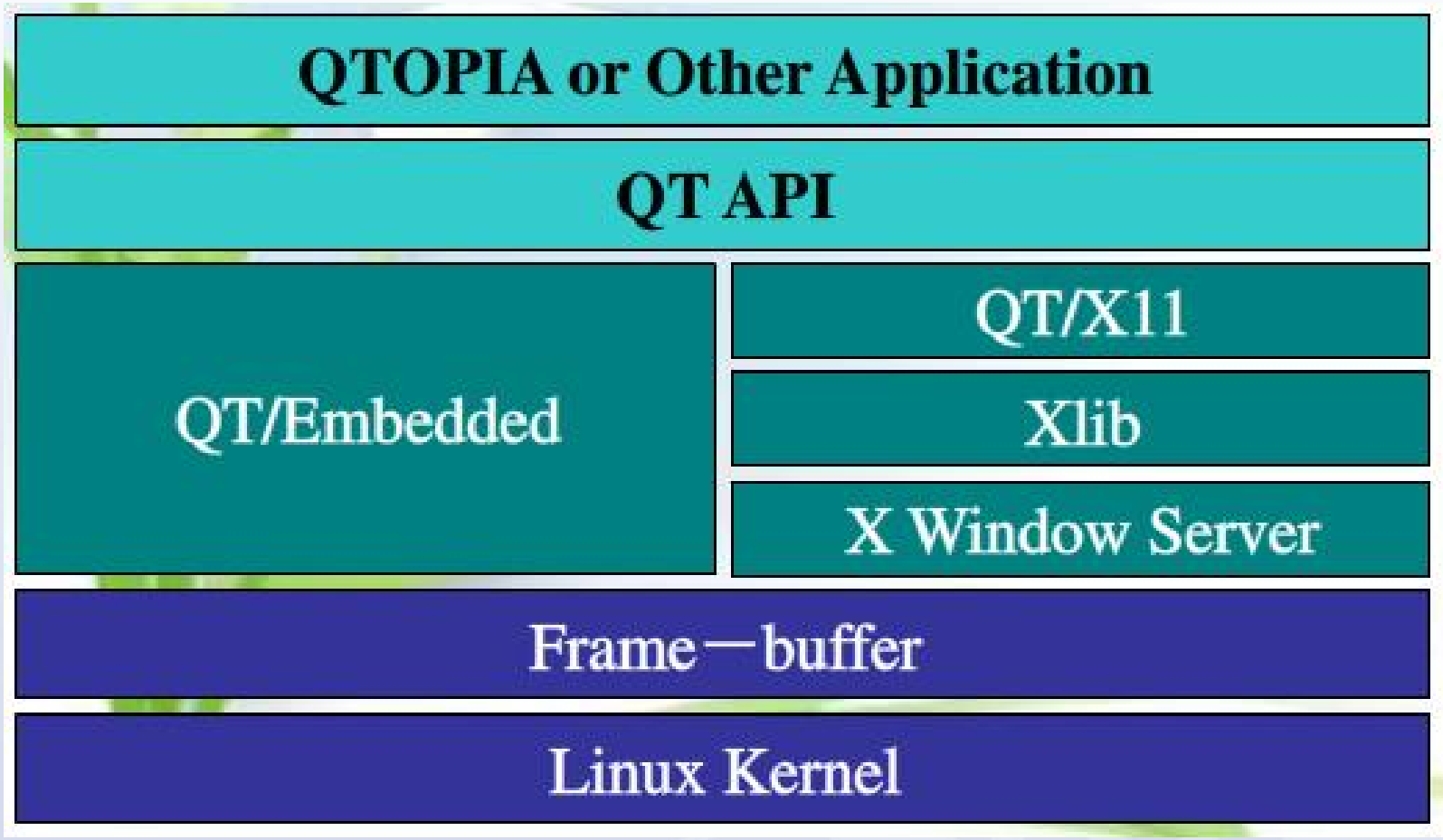
# 主流嵌入式GUI系统介绍

- Qt/E
- MiniGUI
- Android
- 鸿蒙

# Qt/E介绍（1）

- Qt/E是著名的Qt库开发商TrollTech发布的面向嵌入式系统的Qt版本
- Qt是KDE等项目使用的GUI支持库，所以有许多基于Qt的X Window程序可以非常方便地移植Qt/E上
- Qt/E是一个C++函数库，声称可以裁剪到最少630K，但它还是对硬件有较高的要求
- Qt/E是一个多平台的C++图形用户界面应用程序框架，采用面向对象的编程方法

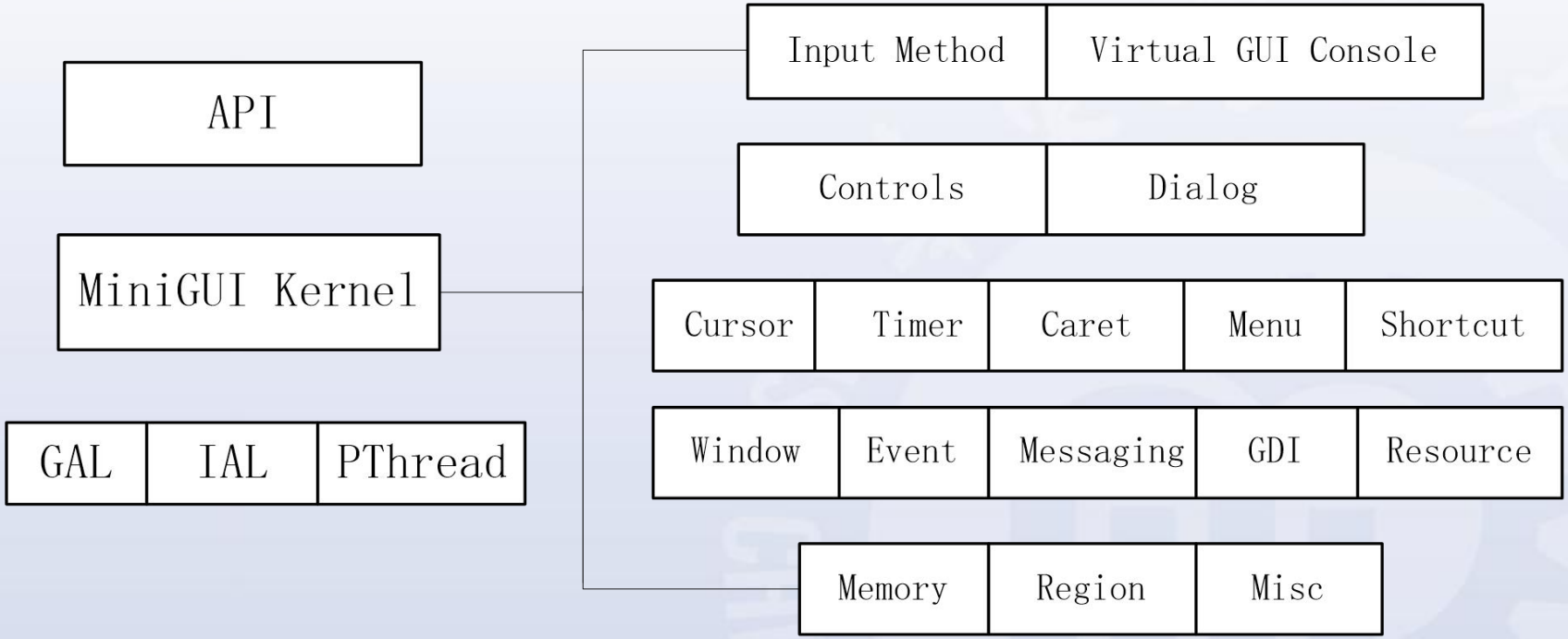
# Qt/E介绍（2）


QTOPIA or Other Application
QT API
QT/Embedded
QT/X11
Xlib
X Window Server
Frame－buffer
Linux Kernel

# MiniGUI介绍（1）

- MiniGUI由魏永明主持和开发的自由软件项目，现由北京飞漫软件技术有限公司负责维护并开展后续开发
- 该项目自1998年底开始到现在，已经非常成熟和稳定，并且得到广泛应用
- MiniGUI能够支持Linux/uCLinux、eCos、uC/OS-II、Vxworks、PSoS、ThreadX等多种操作系统，并且可以工作在Intel X86、ARM系列、PowerPC、MIPS、M68K等多种硬件平台

# MiniGUI介绍（2）


API
MiniGUI Kernel
GAL
IAL
PThread
Input Method
Virtual GUI Console
Controls
Dialog
Cursor
Timer
Caret
Menu
Shortcut
Window
Event
Messaging
GDI
Resource
Memory
Region
Misc

# MiniGUI安装

- MiniGUI v1.6.10主要有源代码包、资源包、游戏等演示程序构成
  - qvfb-1.1.tar.gz：由Qt提供的虚拟FrameBuffer的X11
  - libpng_src.tgz：支持PNG展现的库的源代码包
  - jpegsrc.v6b.tar.gz：支持JPEG的源代码包
  - games-1.6.10.tar.gz：运行在MiniGUI上的小游戏的安装包
  - samples-1.6.10.tar.gz：基于MiniGUI的例程
  - minigui-res-1.6.10.tar.gz：MiniGUI的基本资源包
  - mg-samples-1.6.10.tar.gz：示例程序
  - mde-1.6.10.tar.gz：演示程序包
  - libminigui-1.6.10.tar.gz：核心源代码包

# MiniGUI配置

- MiniGUI有两种基本的运行环境
  - 运行在模拟FrameBuffer的X11程序——QVFB（基于X Window）上
  - 直接运行在Linux内核提供的FrameBuffer驱动程序上
- 大部分基于Linux等操作系统的嵌入式设备都支持FrameBuffer的驱动程序，可以直接在宿主机与嵌入式系统上运行相同的程序代码
- 由于FrameBuffer的复杂性，建议使用QVFB作为MiniGUI的运行环境；可绕过其复杂的安装与配置过程，适合初学者与普通用户使用

# 基于MiniGUI的应用程序设计

- 编程环境介绍
- 编程框架介绍
- 基础编程
- 对话框和控件编程
- 图形编程

# 编程环境介绍

- 由于MiniGUI完全由C语言编写
  - 交叉编译工具链

- 在Linux环境下，利用GCC实现相关的编译工作，再由QVFB完成在Linux平台上直接模拟FrameBuffer驱动环境对开发的应用程序进行运行与调试

# 编程框架介绍

- MiniGUI采用了基于线程的体系结构
  - 较为完备的消息传递与多窗口处理机制
- MiniGUI采用了传统的Client/Sever消息管理模式。
- 事件线程获取来自于IAL层（Input Abstract Layer，输入抽象层）的事件，并传递给桌面线程
- 计时线程触发定时器事件
- MiniGUI将图形处理与输入处理过程封装为抽象接口

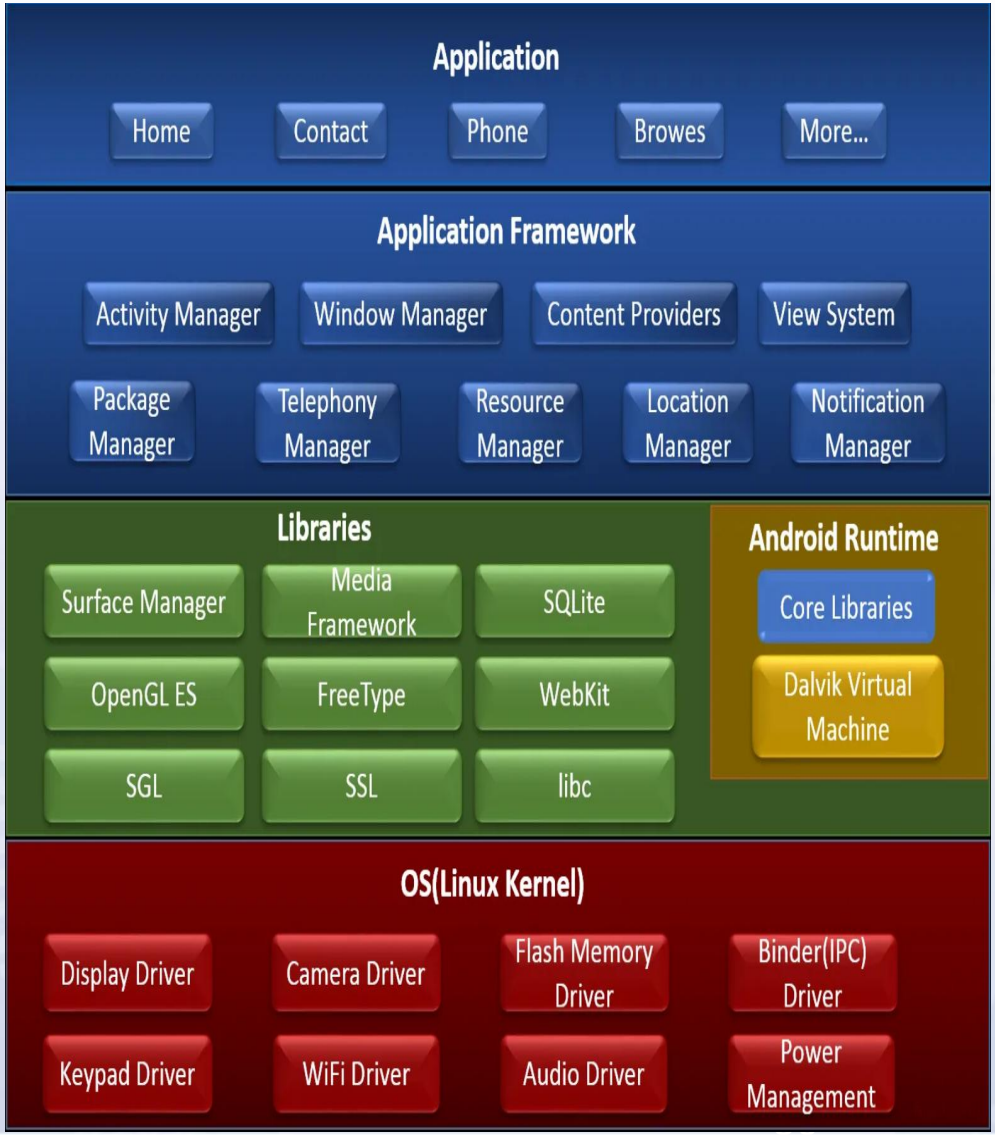
# Android应用程序设计

**应用程序：**
应用层是使用Java语言进行开发的一些应用程序，程序的运行也在应用层上。

**应用程序框架：**
主要是 Google发布的一些操作支持的类库（API框架），开发人员可以使用这些类库方便地进行程序开发。

**系统运行库与Android运行环境：**
通过一些C/C++库来为Android提供主要的特性支持。Android运行环境（ART）使得每一个android程序可以运行在独立的进程中，并且拥有自己的Dalvik虚拟机实例。

**Linux内核：**
Android系统是基于Linux内核层的，这一层为android设备的各种硬件提供了底层的强大驱动。



- 一个Android应用程序通常是由 4 个组件构成
  - **活动（Activity）**
  - **意图（Intent）**
  - **服务（Service）**
  - **内容提供器（Content Provider）**

# Activity

- Activity（活动）是Android应用程序中最基本的组成单位
- Activity主要负责创建显示窗口
  - 一个Activity对象通常就代表了一个单独的屏幕
- Activity是用户唯一可以看得到的组件，用来与用户进行交互的
- Activity是通过一个Activity栈来进行管理

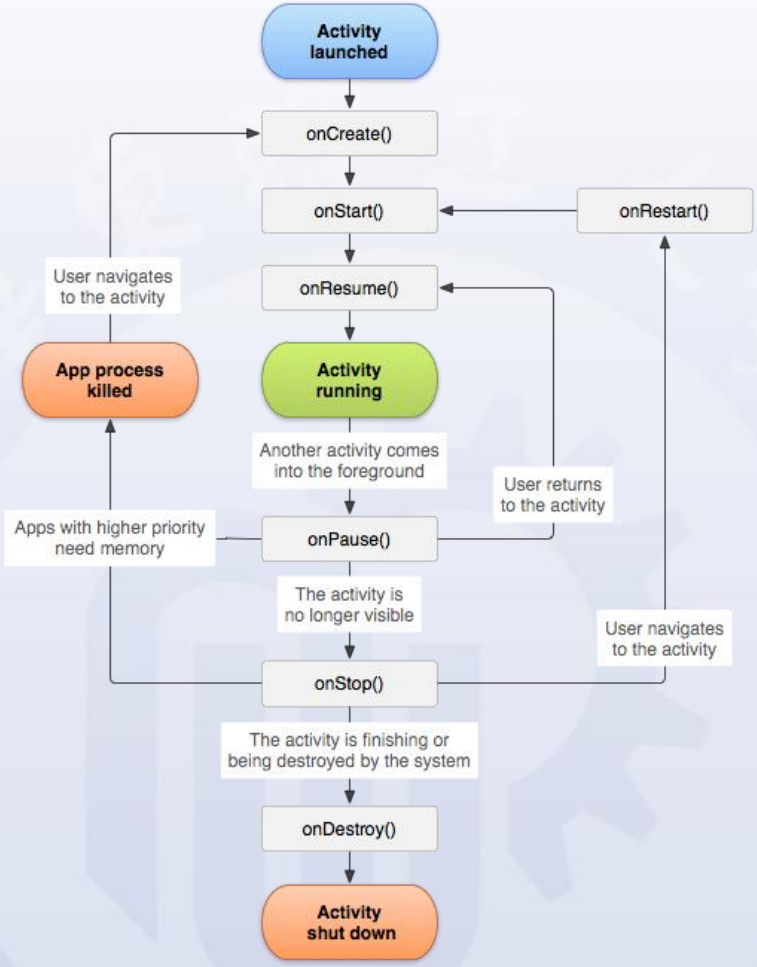
```
import android.app.Activity;
import android.os.Bundle;
public class MyActivity extends Activity {
    ......
}
```

# Activity生命周期

当用户浏览、退出和返回到应用时，应用中的Activity实例会在其生命周期的不同状态间转换。

为了在Activity生命周期的各个阶段之间导航转换，Activity类提供六个核心回调：

- onCreate()
- onResume()
- onStop()
- onStart()
- onPause()
- onDestroy()



# Intent概述

- Android的核心组件
  - 利用消息实现应用程序间的交互机制
  - 这种消息描述了应用中一次操作的动作、数据以及附加数据
  - 系统通过该Intent的描述负责找到对应的组件，并将Intent传递给调用的组件，完成组件的调用。

- Intent由动作、数据、分类、类型、组件和扩展信息等内容组成
  - 每个组成都由相应的属性进行表示，并提供设置和获取相应属性的方法

# Service简介

- 适用于开发无界面、长时间运行的应用功能
-
- 特点
  - 没有用户界面
  - 比Activity 的优先级高，不会轻易被Android系统终止
  - 即使Service被系统终止，在系统资源恢复后Service也将自动恢复运行状态
  - 用于进程间通信（Inter Process Communication，IPC），解决两个不同Android应用程序进程之间的调用和通讯问题

# 数据存储

- 1、SharedPreferences
- 2、文件
- 3、SQLite数据库
- 4、ContentProvider

# 编程环境

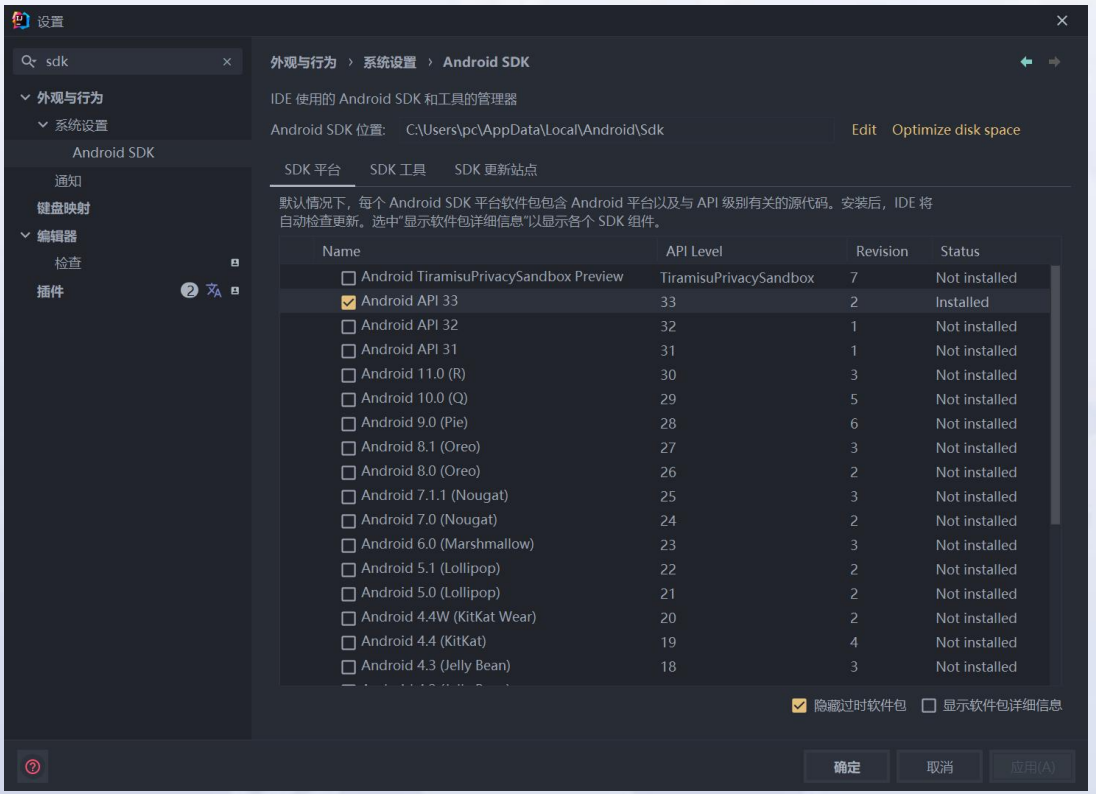
- Eclipse开发环境 + ADT插件
  - Java语言 + AndroidSDK
- Android Sdudio
- ....

# Android应用开发环境的搭建—idea编辑器

1.安装jdk

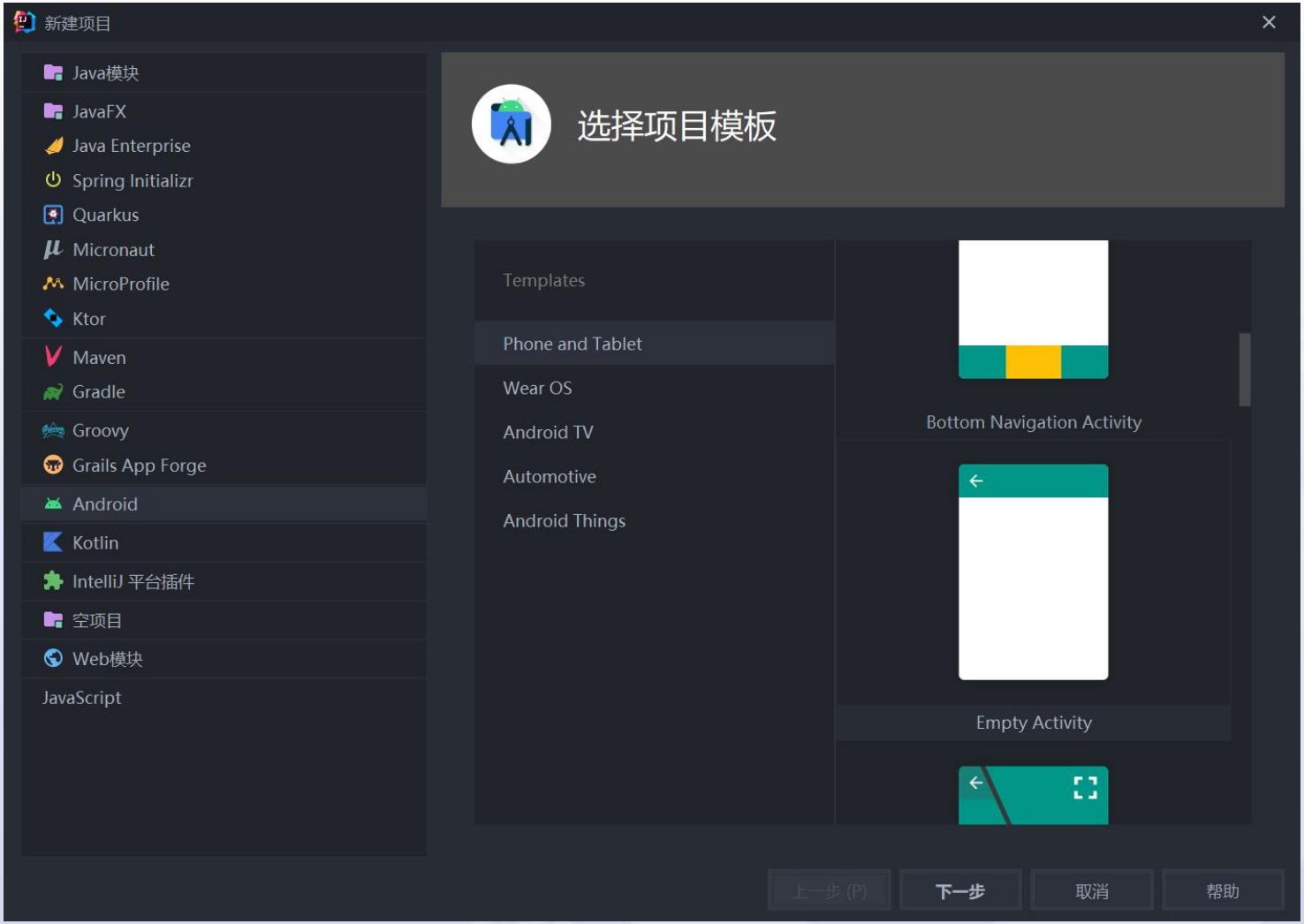
```
C:\Users\pc>java -version
java version "1.8.0_201"
Java(TM) SE Runtime Environment (build 1.8.0_201-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
```

2.安装sdk，并在idea中配置



# Android应用开发环境的搭建—idea编辑器

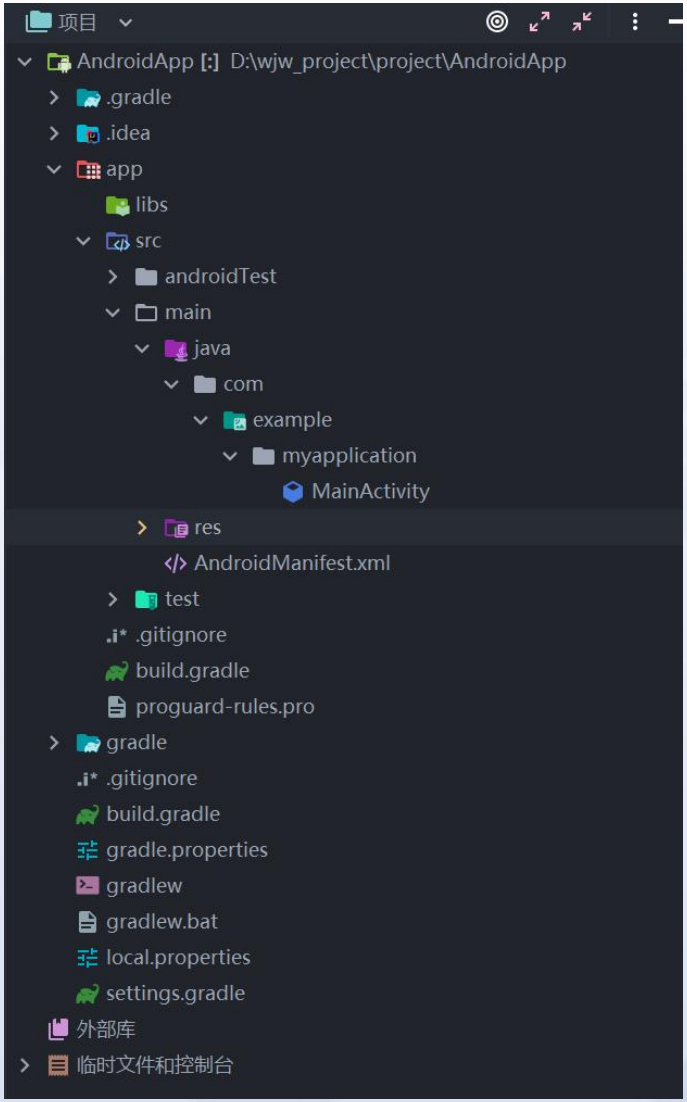
3.新建Android项目



# Android应用开发环境的搭建—idea编辑器

**目录结构：**

- libs：库文件
- main.java:存放各种类，实现功能
- main.res:存放资源，包括图片，布局，字符串等
- build.gradle:全局构建脚本
- .properties:配置文件



# Android应用程序架构

- src/ java原代码存放目录
- gen/ 自动生成目录
- res/ 资源(Resource)目录
- AndroidManifest.xml 功能清单文件
- default.properties 项目环境信息

## 开发一个电话拨号器程序

# 电话拨号器开发（1）

因为应用要使用手机的电话服务，所以要在清单文件AndroidManifest.xml中添加电话服务权限:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="cn.itcast.action"
    android:versionCode="1"
    android:versionName="1.0">
    略....
  <uses-sdk android:minSdkVersion="6" />
  <uses-permission android:name="android.permission.CALL_PHONE"/>
</manifest>
```

# 电话拨号器开发（2）

界面布局:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >
    <TextView
    android:layout_width="fill_parent" android:layout_height="wrap_content"
    android:text="@string/inputmobile"/>

    <EditText android:layout_width="fill_parent" android:layout_height="wrap_content"
    android:id="@+id/mobile"/>

    <Button android:layout_width="wrap_content" android:layout_height="wrap_content"
    android:text="@string/button"
    android:id="@+id/button"/>
</LinearLayout>
```

LinearLayout (线性布局)、AbsoluteLayout(绝对布局)、RelativeLayout(相对布局)、TableLayout(表格布局)、FrameLayout(帧布局)

# 电话拨号器开发（3）

Activity:

```
public class DialerAction extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button button = (Button)findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener(){
        public void onClick(View v) {
                EditText editText = (EditText)findViewById(R.id.mobile);
                Intent intent = new Intent(Intent.ACTION_CALL, Uri.parse("tel:"+
editText.getText()));
                DialerAction.this.startActivity(intent);
        }
        });
    }
}
```

# 鸿蒙应用开发-历史背景

**数字化的时代背景**

**中国面临“卡脖子”挑战**

**大数据和云计算**

**IoT和5G**

5G物联网时代对操作系统的新要求

**AI兴起**

**全球信息安全受到挑战**

网络安全威胁呈现多元化、
复杂化、高频发

# 系统架构


应用层
系统应用
桌面
控制栏
设置
电话
……
扩展应用/三方应用
框架层
系统基本能力子系统集
UI框架
用户程序框架
Ability框架
多模输入子系统
图形子系统
安全子系统
AI子系统
基础软件服务子系统集
事件通知子系统
电话子系统
多媒体子系统
DFX子系统
MSDP&DV子系统
增强软件服务子系统集
智慧屏专有业务子系统
穿戴专有业务子系统
IoT专有业务子系统
硬件服务子系统集
位置服务子系统
生物特征识别子系统
穿戴专有硬件服务子系统
IoT专有硬件服务子系统
系统服务层
分布式任务调度
分布式数据管理
分布式软总线
方舟多语言运行时子系统
公共基础库子系统
内核层
KAL（内核抽象层）
内核子系统
Linux Kernel
LiteOS
.....
驱动子系统
HDF（硬件驱动框架）

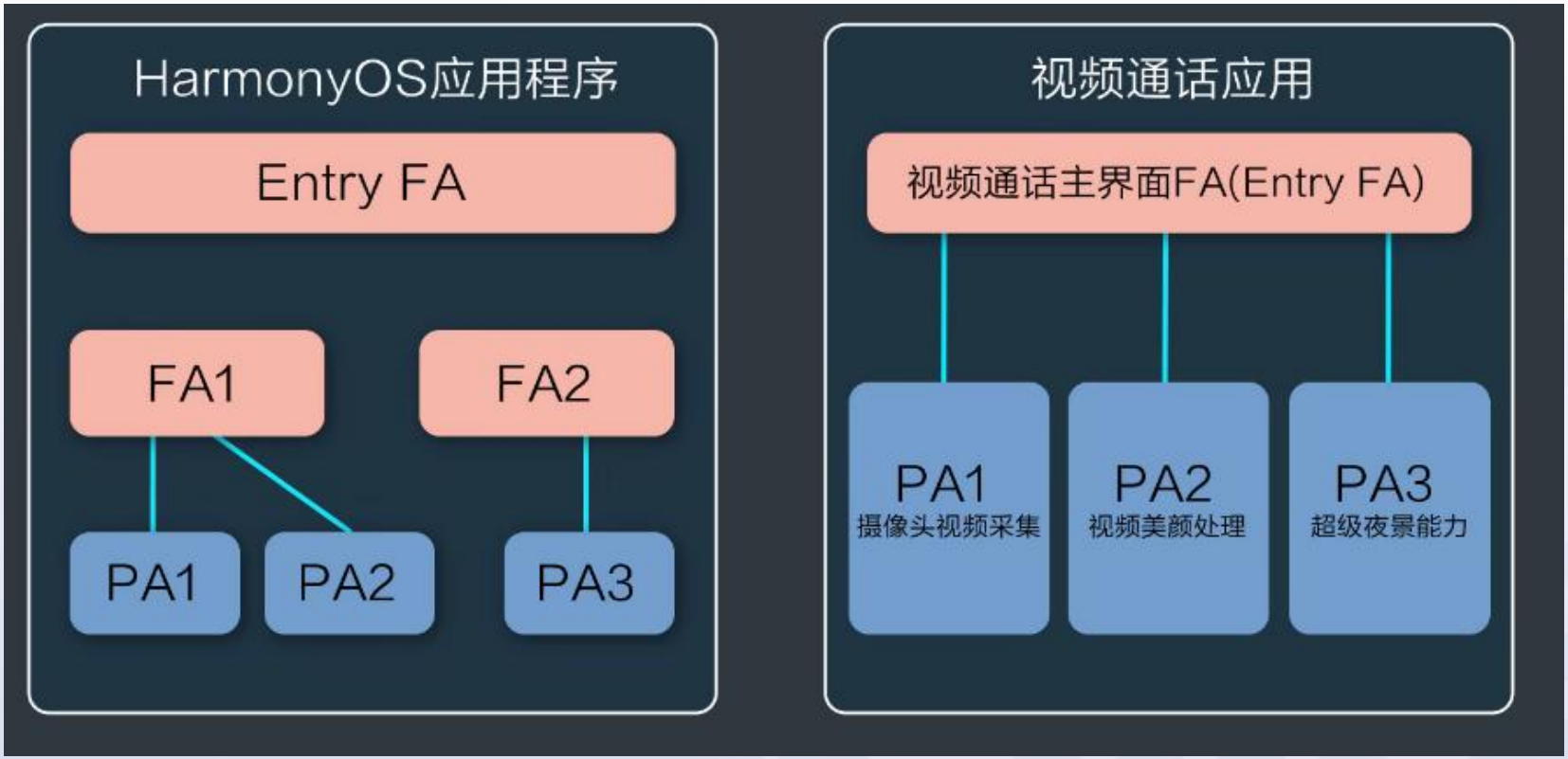
# 应用开发

- 准备开发环境
- ABILITY开发
- UI开发
- 业务功能开发
- 应用调试及发布

# Ability

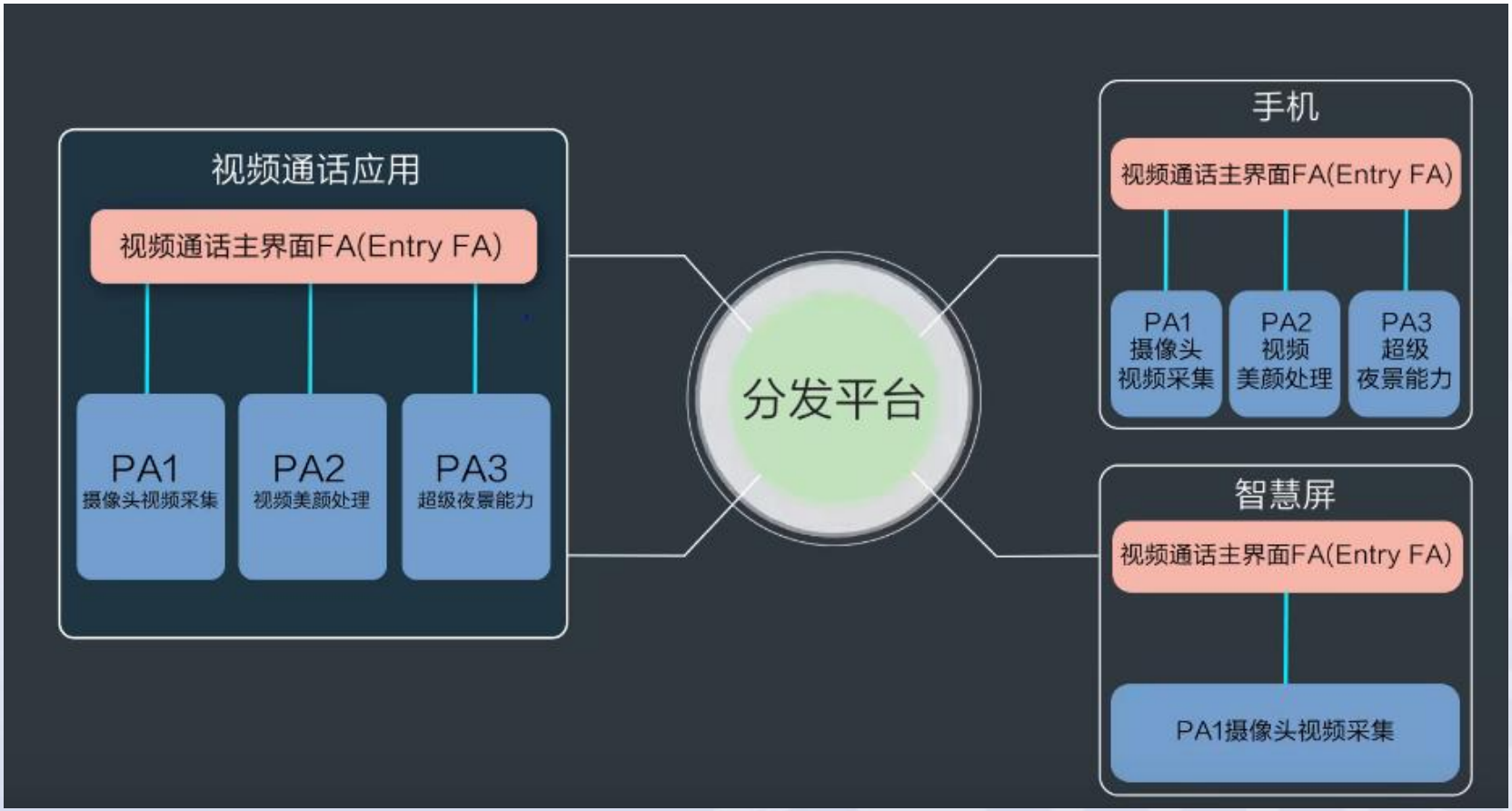
- Ability 是应用所具备能力的抽象，也是应用程序的重要组成部分
- 一个应用可以具备多种能力（即可以包含多个 Ability）
  - HarmonyOS 支持应用以 Ability 为单位进行部署
- Ability 可以分为 FA（Feature Ability）和 PA（Particle Ability）两种类型

# Ability




FA是UI界面，PA则是提供了其他Ability调用自定义的服务，如图中的视频采集等功能

# 分布式




利用一个分发平台分别针对不同的设备 部署不同的PA

如手机就部署了美颜和夜景功能，智慧屏则只有一个PA1

# 技术特性

- 硬件互助，资源共享
  - 分布式设备虚拟化
  - 分布式数据管理
  - 分布式任务调度
- 一次开发，多端部署
- 统一OS，弹性部署


# THANKS!