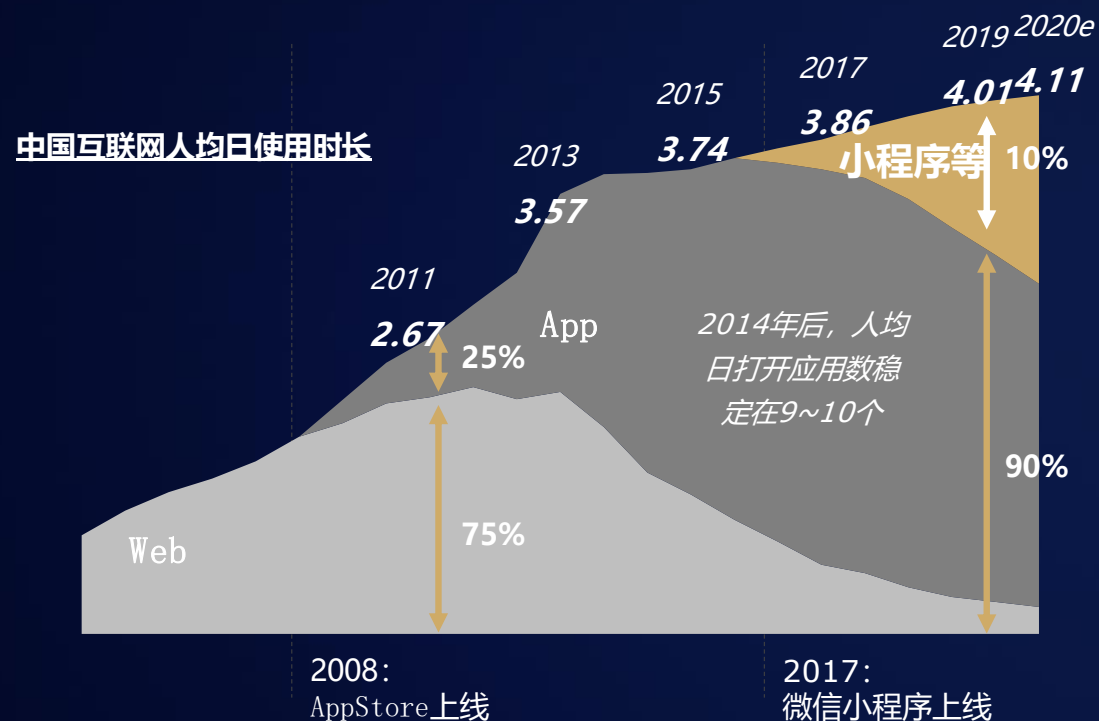


# HarmonyOS元服务的设计与开发解析

# 趋势：适应万物互联时代的元服务新赛道已经展现

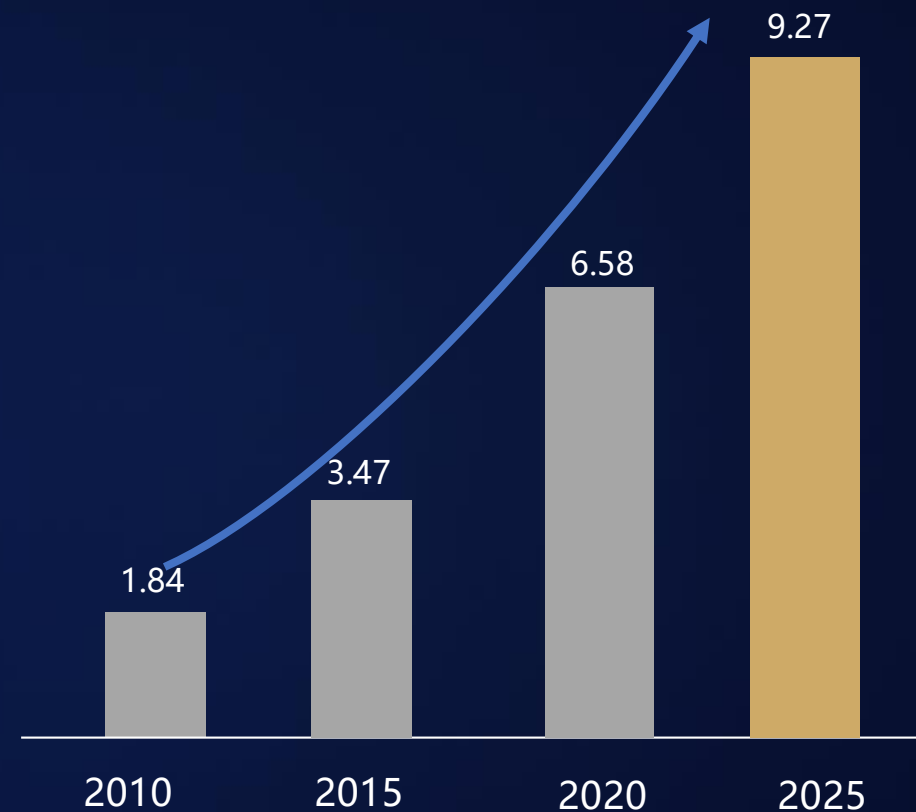


## 元服务的应用场景快速增加



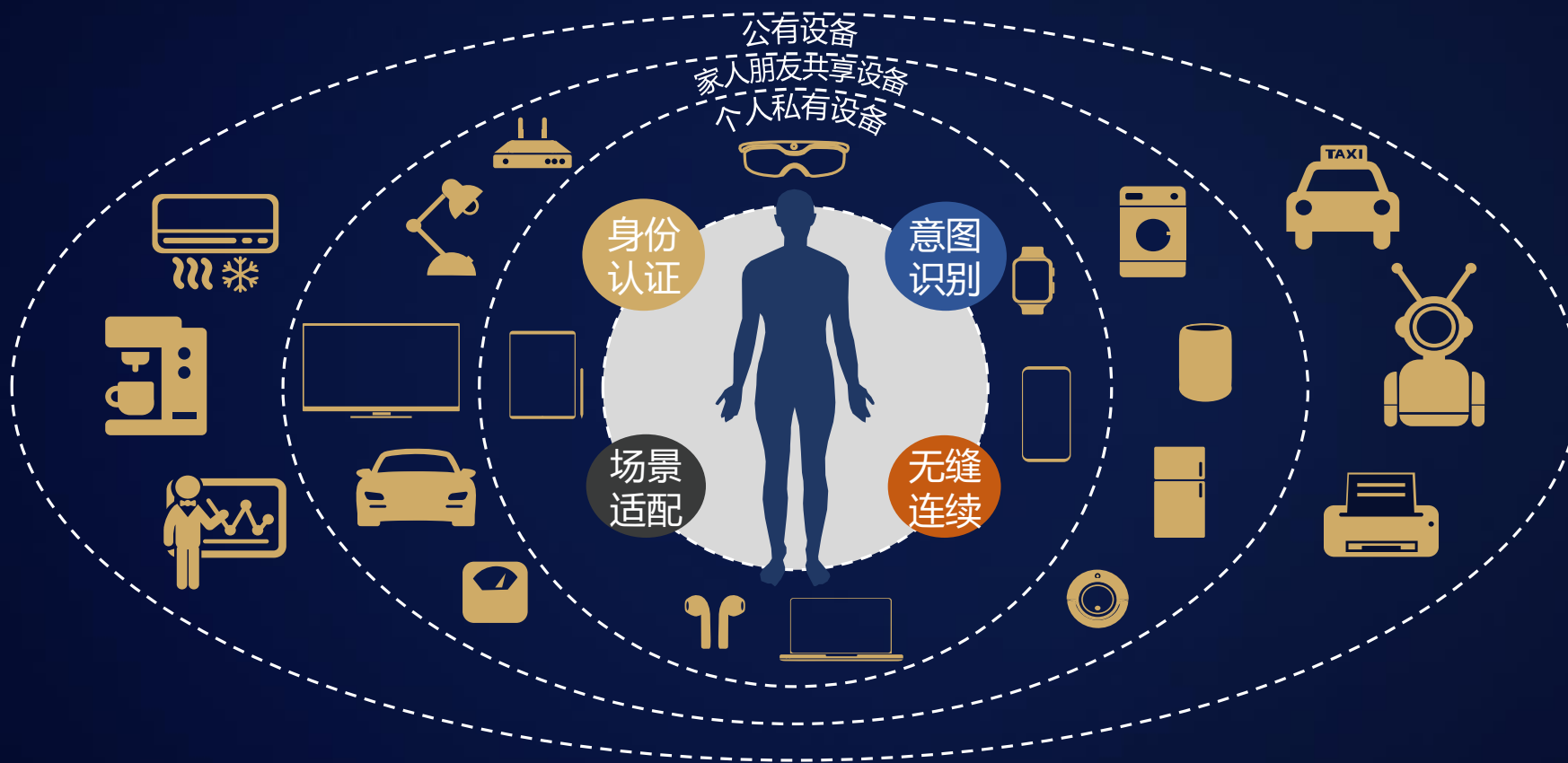
数据来源: CNNIC、eMarketer、阿拉丁

## 人均持有设备统计和预测 (单位: 台)



来源: <https://www.statista.com/statistics/678739/forecast-on-connected-devices-per-person/>

挑战：设备和场景的多样性，增加了应用开发的复杂性



# 案例分享：优酷× HarmonyOS，使能大小屏联动看剧

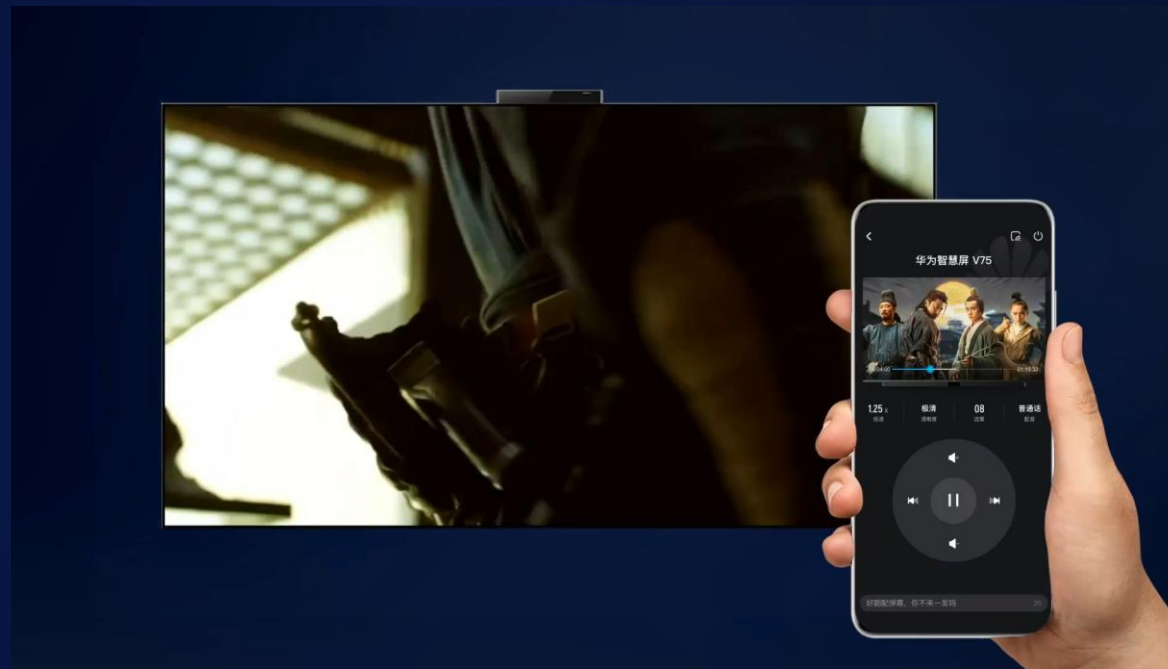


大小屏**相互通信**

实时**触发**联动看剧

大屏看剧情**主线**

小屏看精彩**看点**



# 设计原则：轻量直达的单设备体验 + 无缝一体的多设备体验

## 用户旅程



# 目录

- 关键技术解析
- 单设备场景设计与开发
- 多设备场景设计与开发

# 元服务关键特征：可分、可合、可流转

关键技术

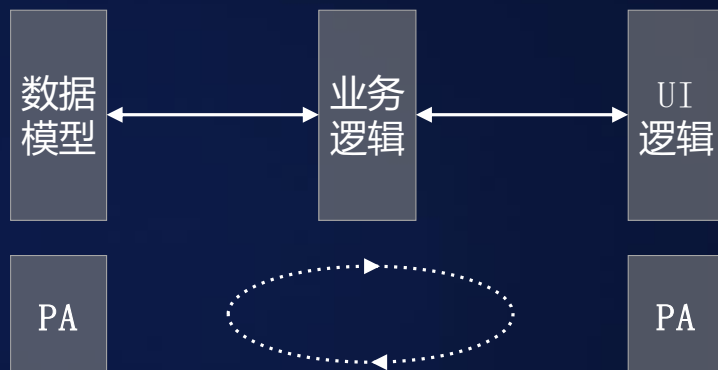
单设备场景

多设备场景



- FA: FA有UI界面, 提供与用户交互的能力
- PA: 无UI界面, 提供后台运行任务的能力以及统一的数据访问抽象; 仅对系统服务有依赖, PA之间不存在依赖

## 关键特征



- 跨设备迁移、跨设备调用、跨设备控制
- 数据模型、业务逻辑、UI逻辑两两分离
- 模块解耦、灵活组装

# HarmonyOS元服务开发框架

关键技术

单设备场景

多设备场景





# 目录

- 关键技术解析
- 单设备场景设计与开发
- 多设备场景设计与开发

免安装FA 1秒内打开，美味一步到位

关键技术

单设备场景

多设备场景



# 服务卡片：

## 快捷高效获取应用内重要信息和高频功能

### 重要信息直接展示

无需打开便可获取应用内重要信息展示和动态变化

### 高频功能一步直达

支持应用内高频功能的快捷入口直达

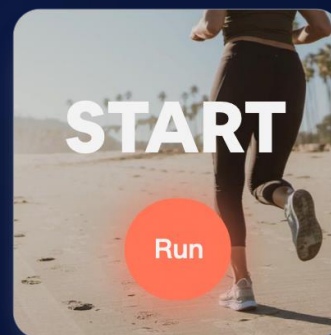
### 免安装打开约束

建议每个免安装的服务HAP包大小 $\leq 5\text{MB}$

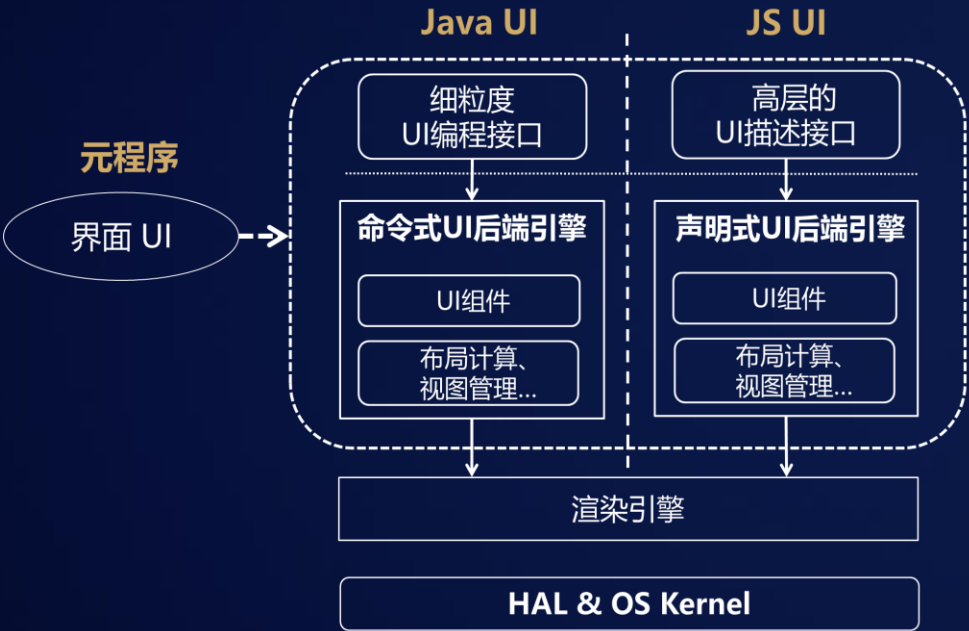
关键技术

单设备场景

多设备场景



支持主流的语言生态 – Java & JS (JavaScript)



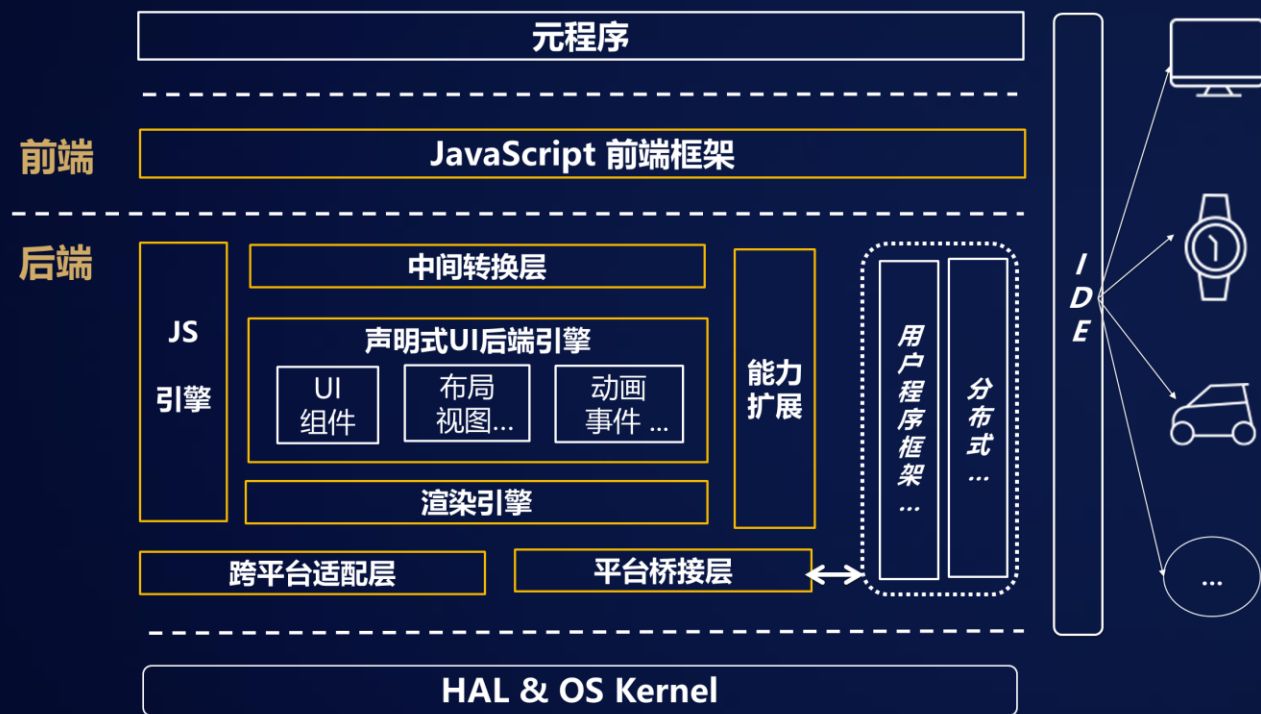
	Java UI	JS UI
语言生态	Java	JS
接口方式	命令式; 细粒度的UI编程接口	声明式; 相对高层的UI描述
执行方式	开发者处理, 基于API驱动的UI变更	框架层处理, 基于数据驱动的UI自动变更

# JS UI总体架构

关键技术

单设备场景

多设备场景



易开发->类Web前端;

高性能->声明式UI后端引擎;

一致性->跨平台渲染;

PC实时预览能力

# JS UI应用示例

关键技术

单设备场景

多设备场景

```
<div class="musicPlayBG">
  <div class="musicPlayerInfo">
    ...
  </div>
  <div class="musicPlayerControl">
    <div class="playProgressDiv">
      <progress percent="{{progress}}"/></progress>
    </div>
    <div class="playControlBtnDiv">
      ...
      <button icon="{{playImg}}" onclick="musicPlay()"></button>
    </div>
  </div>
</div>
```

模板

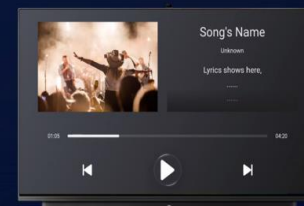
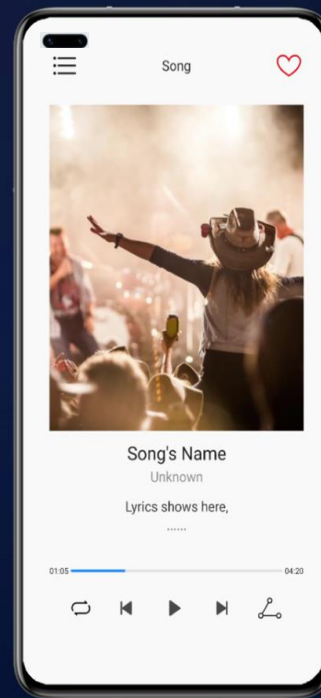
- ☐ 数据绑定
- ☐ 事件绑定
- ☐ 扩展能力
- ☐ 多设备UI自适应

```
@media screen and (device-type : tv) {
  .musicPlayerInfo {
    /* flex布局支持 */
    flex-direction: row;
    flex-grow: 1;
  }
  .musicPlayerControl {
    flex-direction: column;
  }
}
/* MediaQuery支持多设备UI自适应 */
@media screen and (device-type: watch) {
  ...
}
```

样式

```
export default {
  data: {
    playImg = "common/playImg.png"
    progress = 0; // 数据绑定
  },
  async musicPlay() { // 事件绑定
    ...
    // Ability调用
    var result =
    await FeatureAbility.callAbility(action);
    var ret = JSON.parse(result);
    if (ret.code == 0) {
      progress = ret.progress;
    }
  }
}
```

业务逻辑

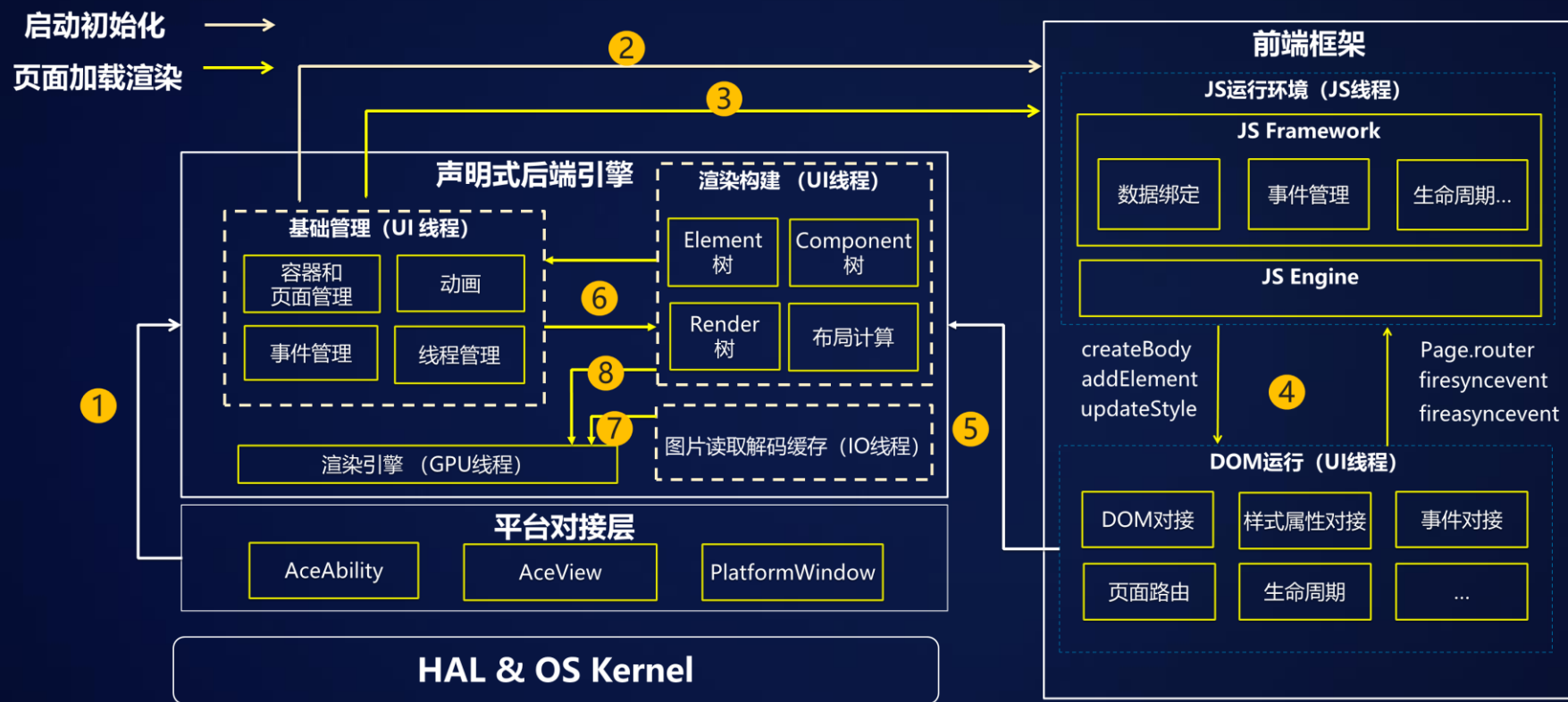


# JS UI运行流程

关键技术

单设备场景

多设备场景



测试部典型应用	评测维度	典型场景	友商方案	ACEJS
无干扰	性能体验	冷启动时延 (ms)	534	525
	内存开销	冷启动内存 (MB)	43	40
视频播放	性能体验	视频播放帧率 (fps)	58	59.8
	内存开销	视频播放内存 (MB)	200	80
	功耗开销	视频播放功耗 (mA)	790	232
复杂动效	性能体验	复杂动效帧率 (fps)	59	59.8
	内存开销	复杂动效内存 (fps)	220	120
	功耗开销	复杂动效功耗 (fps)	438	438



# 目录

- 关键技术解析
- 单设备场景设计开发
- 多设备场景设计开发

多设备不能快速协同，就如驾驶时没有后视镜

关键技术

单设备场景

多设备场景



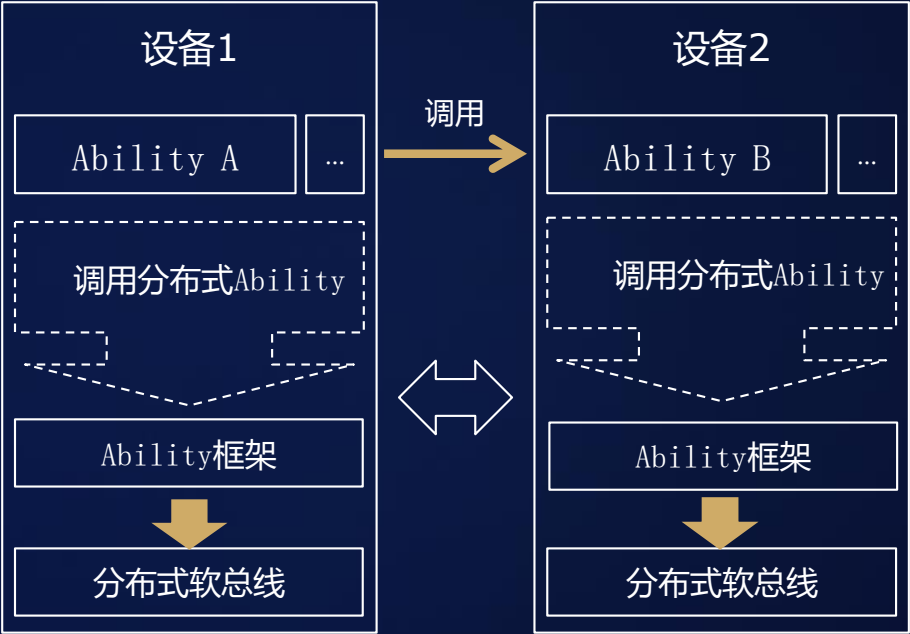
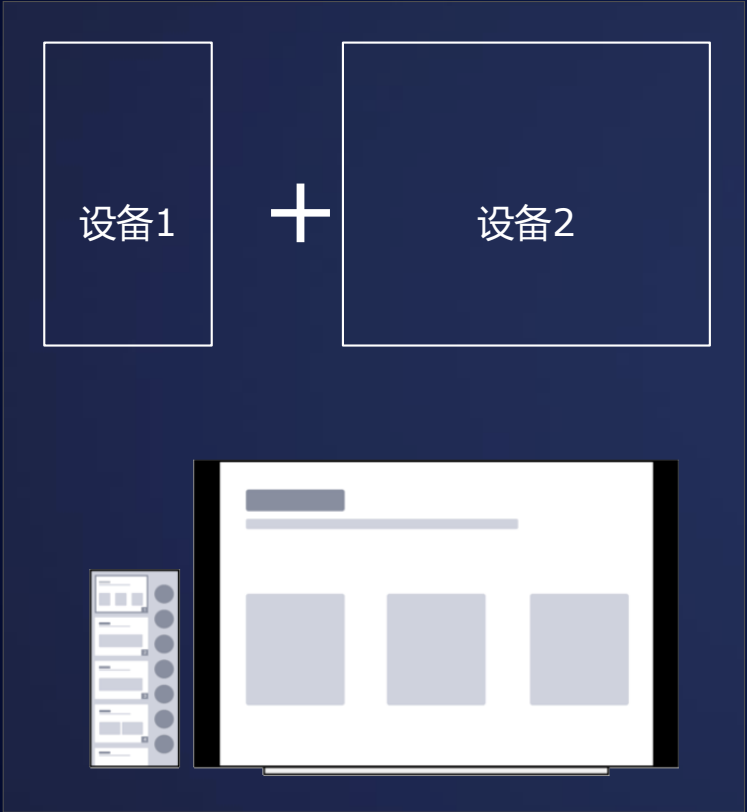
# 分布式协同体验设计

关键技术

单设备场景

多设备场景

协同设计



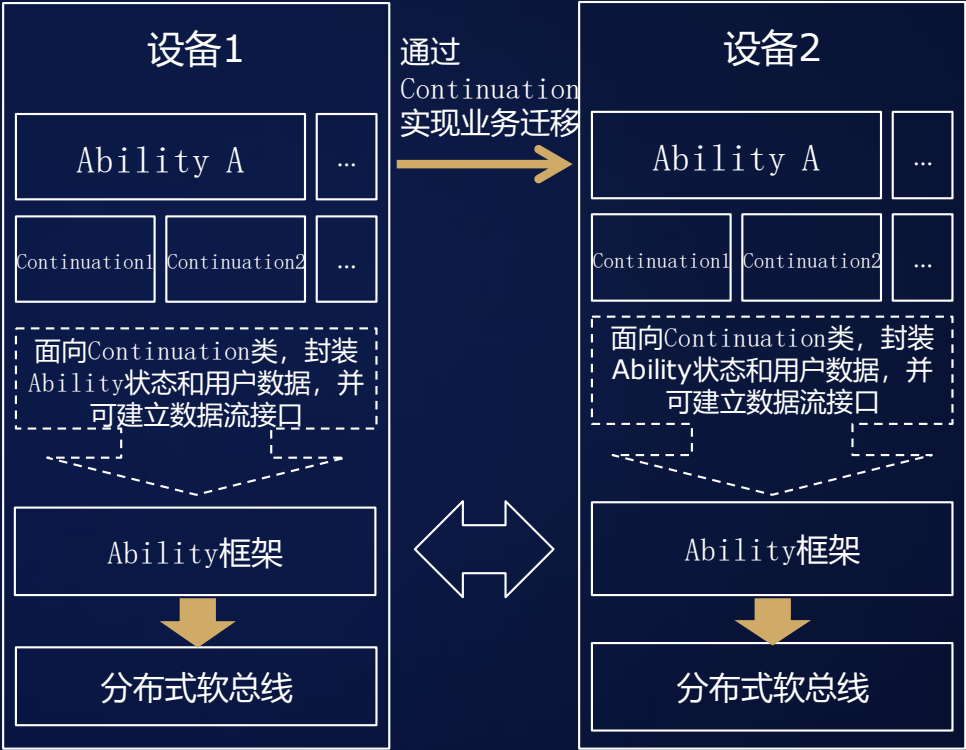
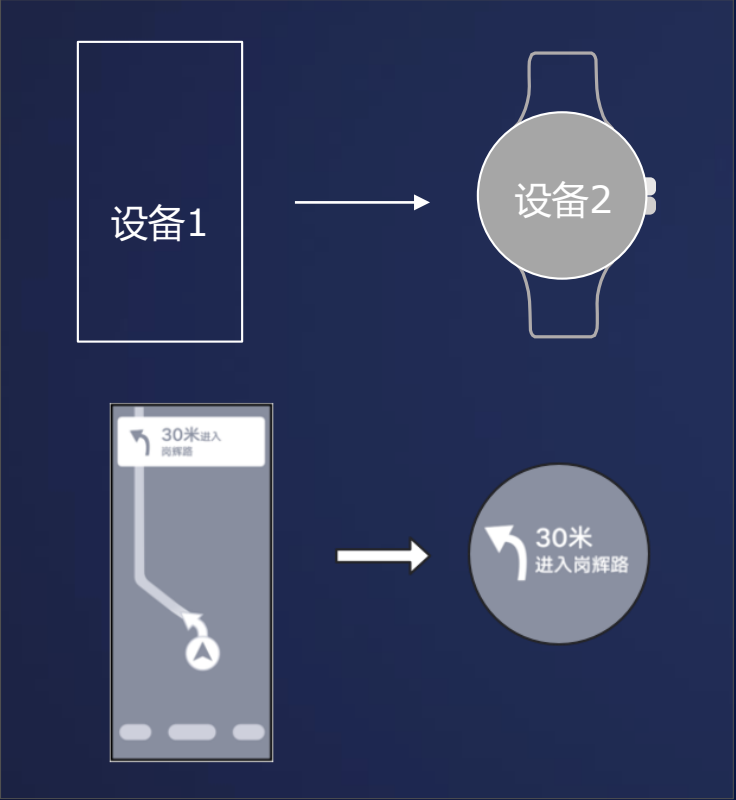
# 分布式连续性体验设计

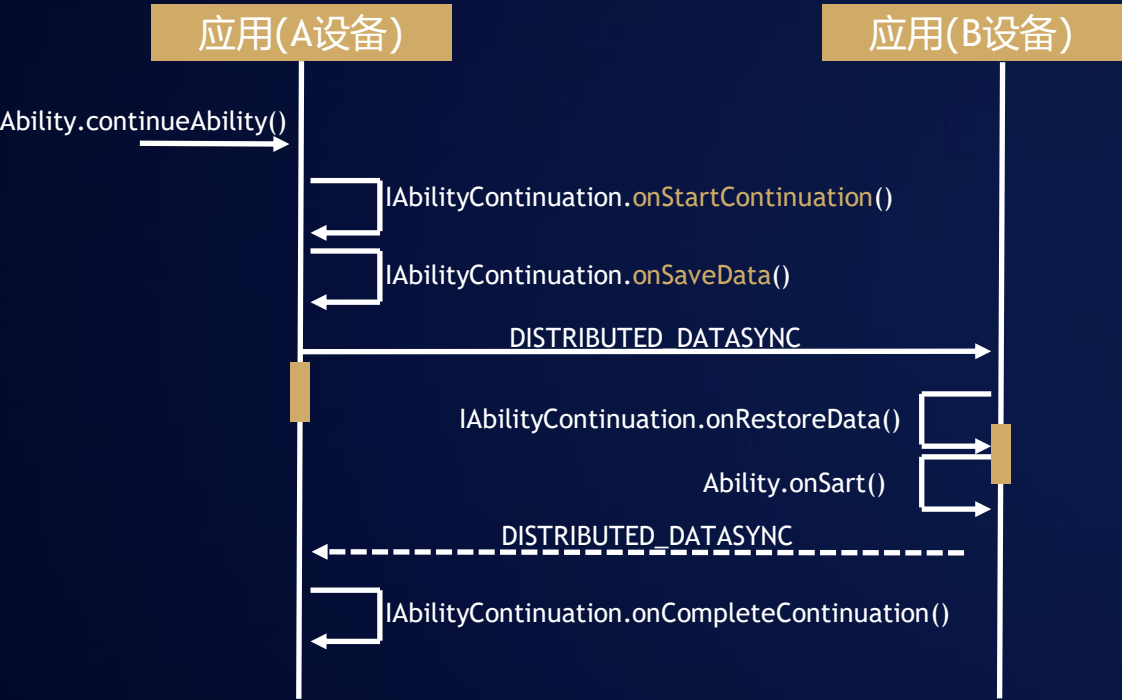
关键技术

单设备场景

多设备场景

连续性设计



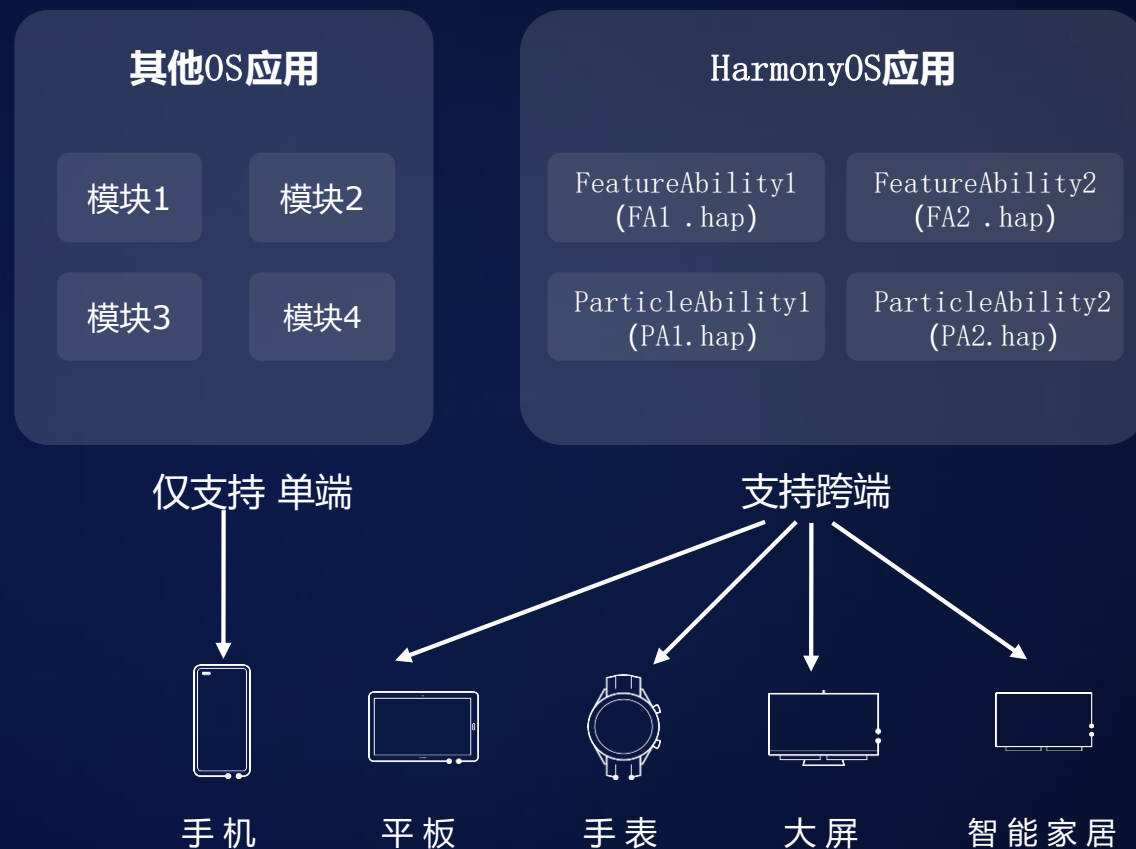


## Ability implements IAbilityContinuation

接口名称	说明
<code>onStartContinuation()</code>	Page请求迁移后，系统首先回调此方法，开发者可以在此回调中决策当前是否可以执行迁移。
<code>onSaveData()</code>	Page请求迁移后，系统首先回调此方法，开发者可以在此回调中决策当前是否可以执行迁移。
<code>onRestoreData()</code>	源侧设备上Page完成保存数据后，系统在目标侧设备上回调此方法，开发者在此回调中接受用于恢复Page状态的数据。注意，在目标侧设备上的Page会重新启动其生命周期，无论其启动模式如何配置。且系统回调此方法的时机在 <code>onStart()</code> 之前。
<code>onCompleteContinuation()</code>	目标侧设备上恢复数据一旦完成，系统就会在源侧设备上回调Page的此方法，以便通知应用迁移流程已结束。开发者可以在此检查迁移结果是否成功，并在此处理迁移结束的动作，例如，应用可以在迁移完成后终止自身生命周期。

## 元服务 实现跨端部署的基础单元

- 关键技术：可分、可合、可流转
- 单设备体验：轻量直达、性能高效
- 多设备体验：无缝一体、服务随行



# THANK YOU



欢迎访问HarmonyOS开发者官网



欢迎关注HarmonyOS开发者微信公众号