China · Shenzhen

无线动态化框架 LuaView&性能优化

陈松涛（野松）

天猫－无线基础框架与创新－资深无线开发工程师

ArchSummit 全球架构师峰会 2017

Alibaba Group 阿里巴巴集团

主办方 Geekbang. InfoQ

CNUTCon 2017
全球运维技术大会

上海·光大会展中心大酒店 | 2017.9.10–11

智能时代的新运维

大数据运维
DevOps
安全
SRE
Kubernetes
Serverless
游戏运维
AIOps
智能化运维
基础架构
监控
互联网金融

主办方 Geekbang. 极客邦科技 InfoQ

StuQueue
斯达克学院

实践驱动的IT教育

斯达克学院(StuQ)， 极客邦旗下实践驱动的IT教育平台。通过线下和线上多种形式的综合学习解决方案 ， 帮助IT从业者和研发团队提升技能水平。

人工智能 | 大数据 | 前端开发 | 后端开发 | 架构设计

移动开发 | 运维设计 | 产品测试 | 产品经理 | 技术管理

10大职业技术领域课程

http://www.stuq.org

China · Shenzhen

SPEAKER

INTRODUCE

陈松涛 (野松)

天猫－无线基础框架与创新
资深无线开发工程师

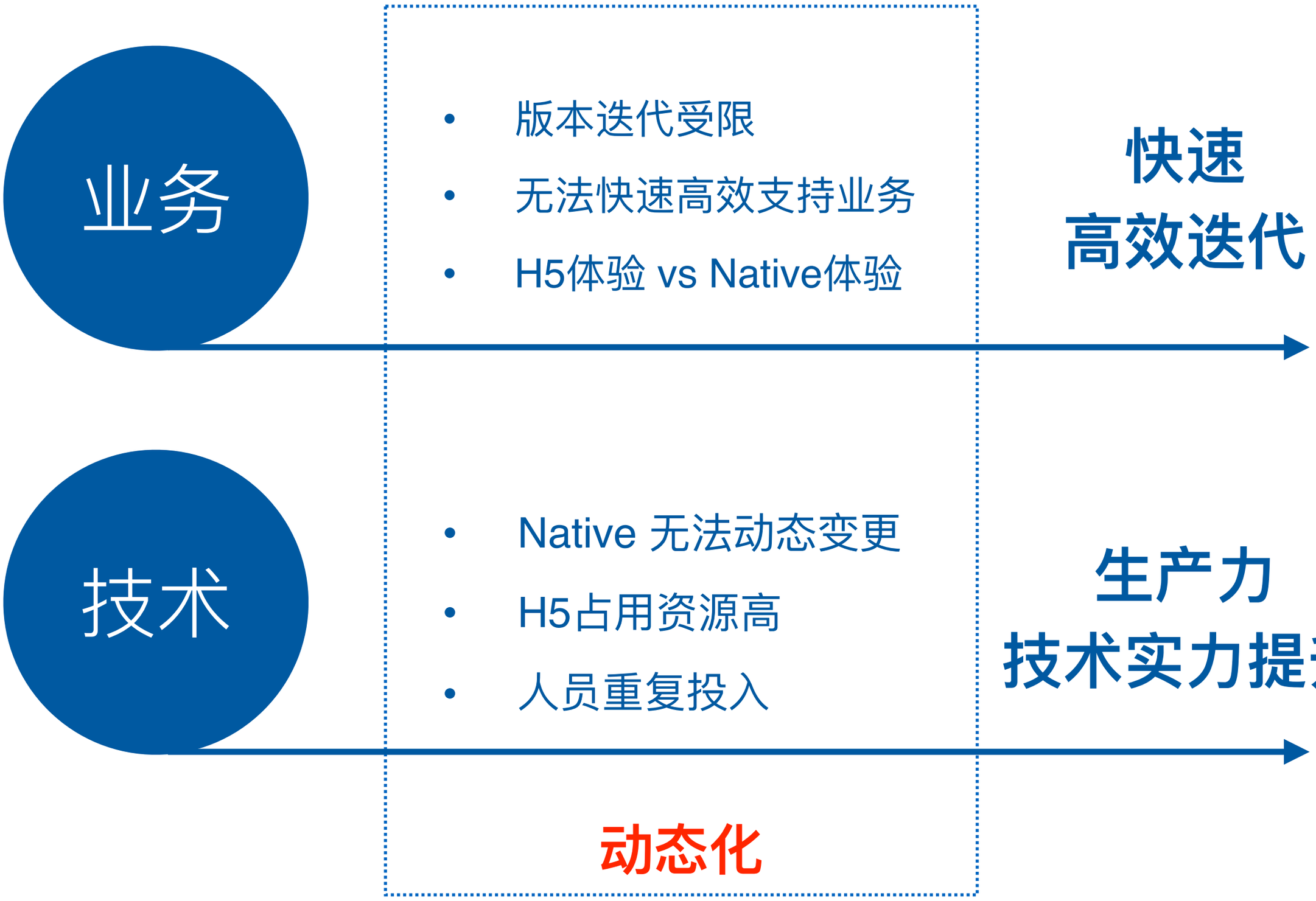
- 浙江大学硕士，2014年加入阿里巴巴聚划算
- 先后负责过聚划算Android客户端开发、聚划算无线数据产品开发，聚划算无线基础框架开发等工作
- 产出过无线数据展示框架AData，无线埋点框架JTrack，以及无线动态化框架LuaView
- 开源动态化框架 LuaView 的Android端作者

SPEAKER

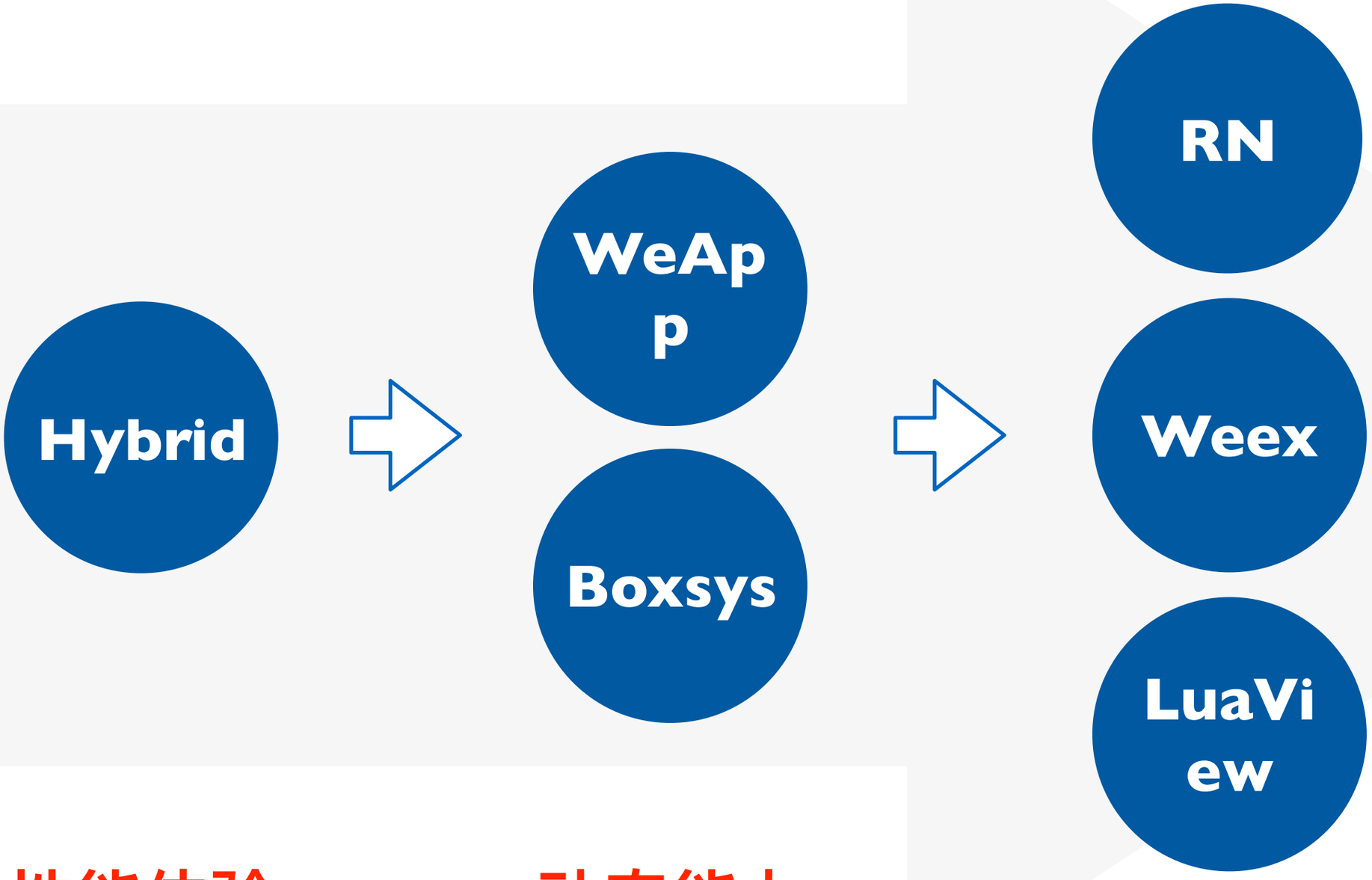
ArchSummit 2017`ShenZhen

TABLE OF
CONTENTS 大纲

移动开发面临的痛点


业务
• 版本迭代受限
• 无法快速高效支持业务
• H5体验 vs Native体验
快速
高效迭代
技术
• Native 无法动态变更
• H5占用资源高
• 人员重复投入
生产力
技术实力提升
动态化

动态化方案的发展

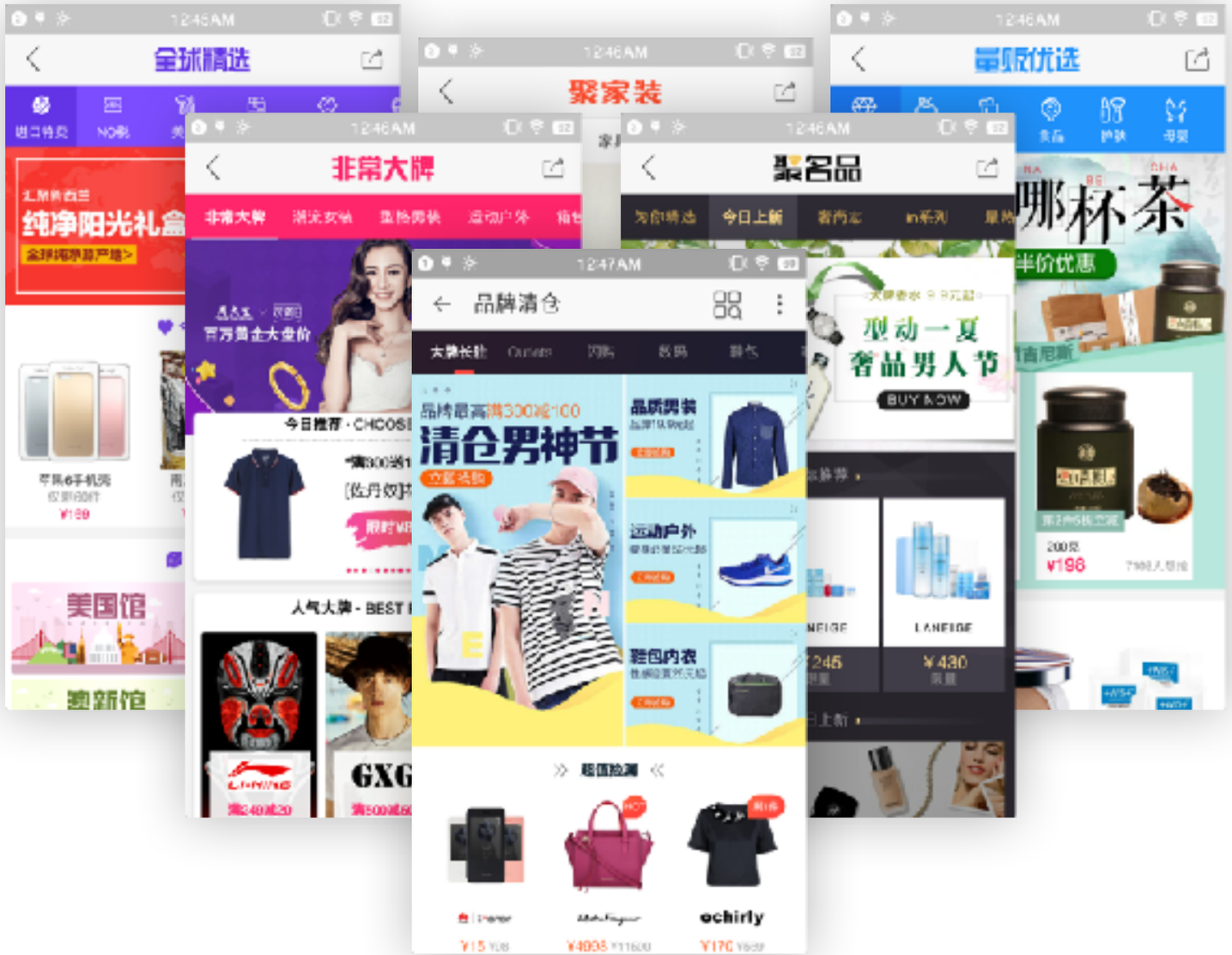


性能体验　　动态能力

LuaView的由来

H5 性能 **Boxsys** 场景 **LuaView**

LuaView在阿里的应用 – （淘宝、天猫、聚划算）亿级用户



日常/大促

活动

创新玩法

动态化方案比较

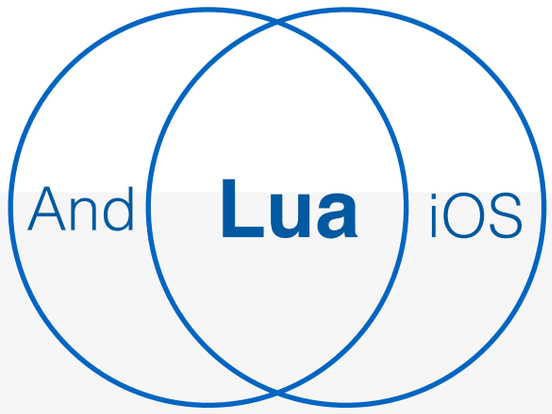
方案	支持平台	开发效率 调试工具	SDK大小 性能	功能丰富程度	动态能力 可扩展性	社区
Hybrid	★★★	★★	★	★	★★	★
Boxsys	★	★	★★★	★	★	-
React Native	★★★	★★	★★	★★★	★★	★★
LuaView	★★	★	★★★	★★	★★☆	★
			500k		**Page、View UI、Kit、Component**	

TABLE OF
CONTENTS 大纲

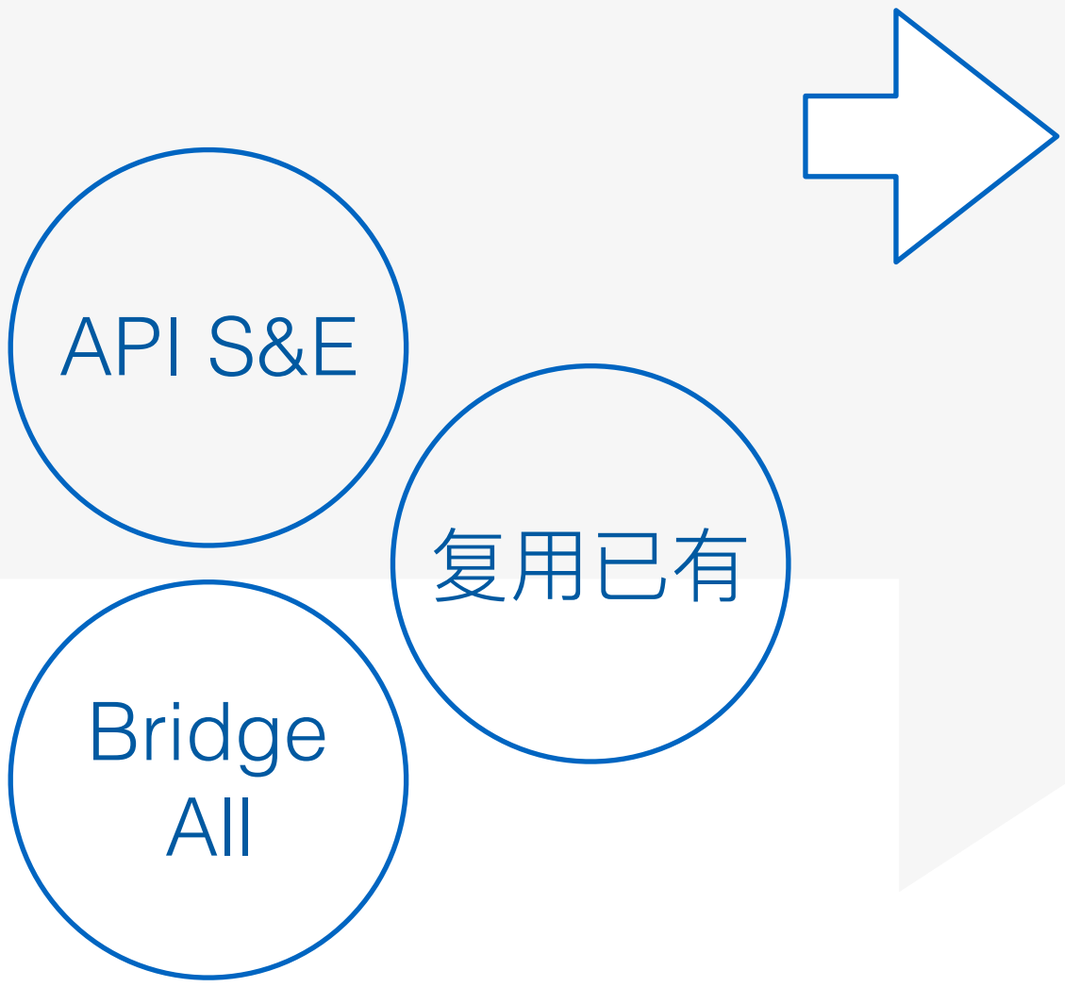
- LuaView 的前世今生
- LuaView 的设计思想&技术架构
- LuaView 的性能优化
- LuaView 的未来

LuaView的设计理念&核心能力

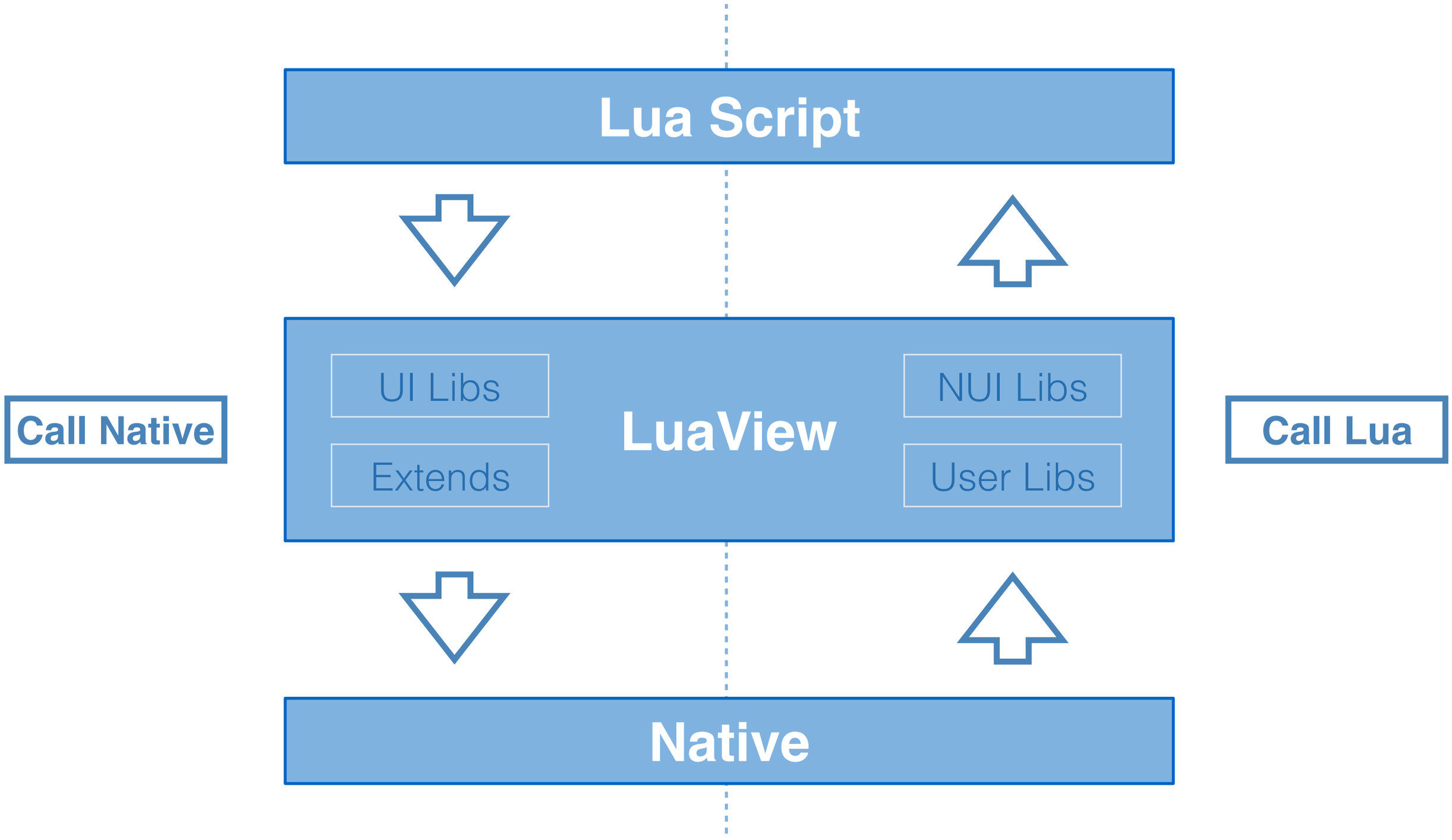
1. WORB



2. Simple & Easy



LuaView是如何工作的 - I



LuaView是如何工作的 - II

userdata <-> native

Call Native

```
local btn = Button()
btn:text(“Test Button”)
…
```

→ function
return userdata

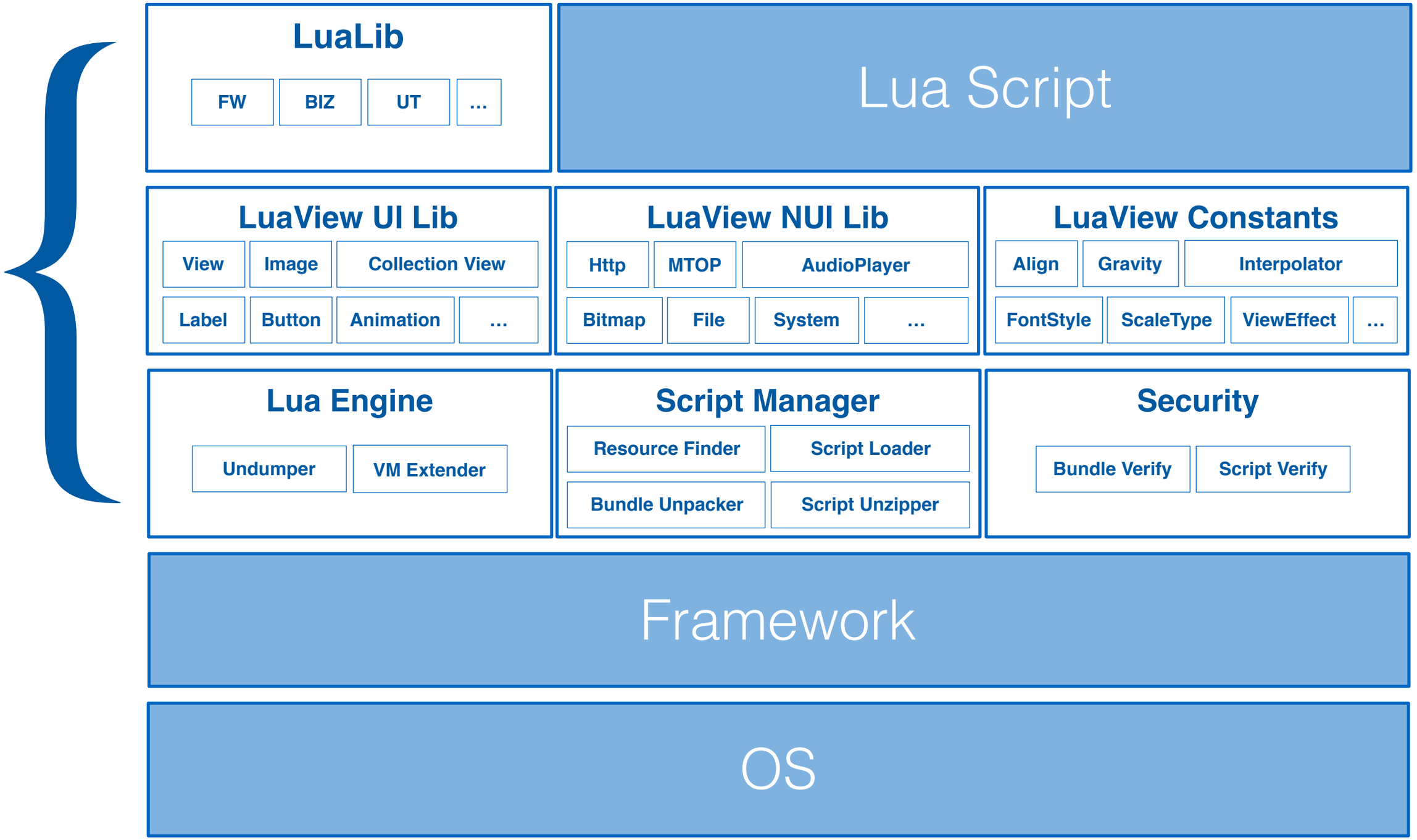
→ call userdata's function
(call native function)

Call Lua

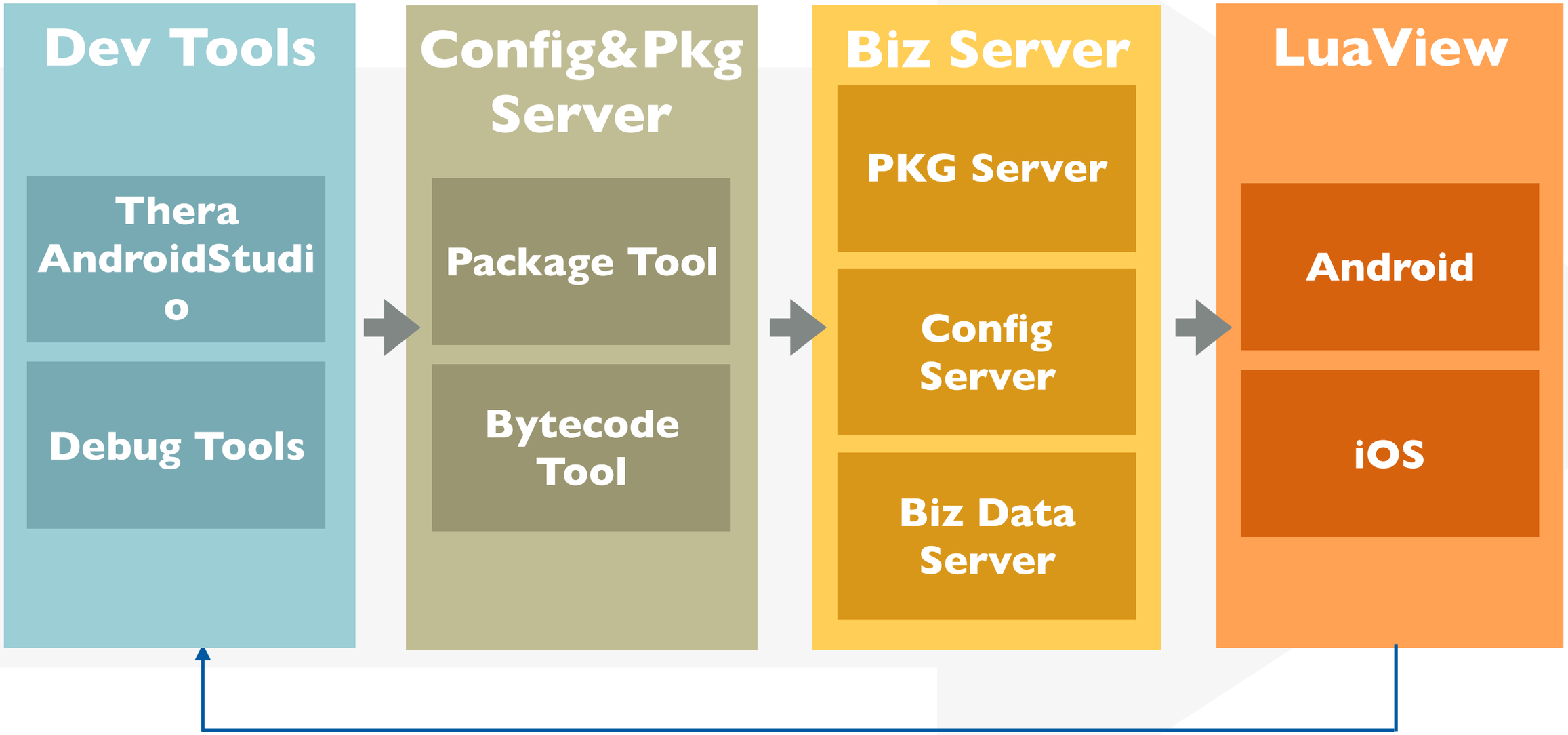
```
btn:onClick(function()
    …
end)
```

→ save lua function
&
call lua function

LuaViewSDK的架构



LuaView开发流程&工具集



LuaView SDK 相关数据

SDK 大小

And：500k

iOS：650k

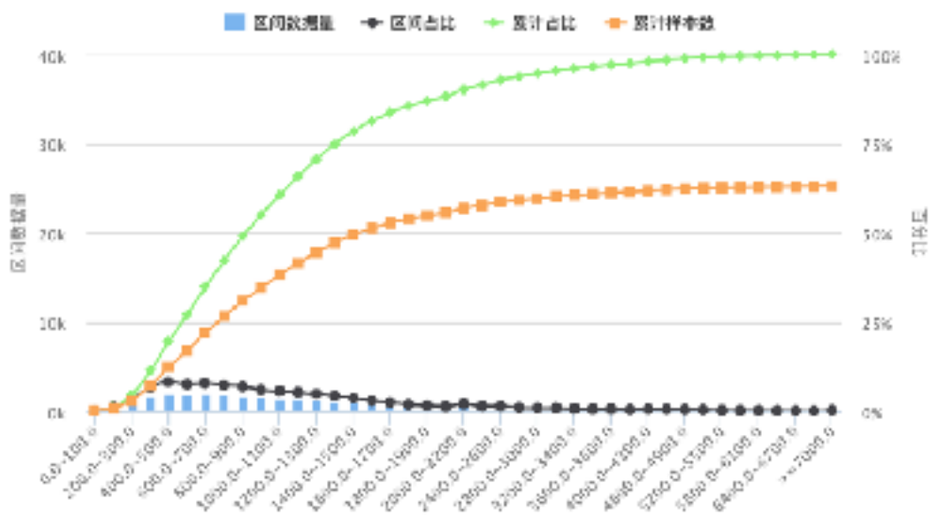
耗时

完整渲染（10000）

And首次: 平均 1.2s，90% 2s，60% 1s

And二次: 平均 1s，90% 1.7s，68% 1s

iOS: 平均 0.65s，90% 1.5s



帧率

52+、48+

稳定性

总体成功率 > 0.9995

SDK错误率 < 0.0005

数据提升

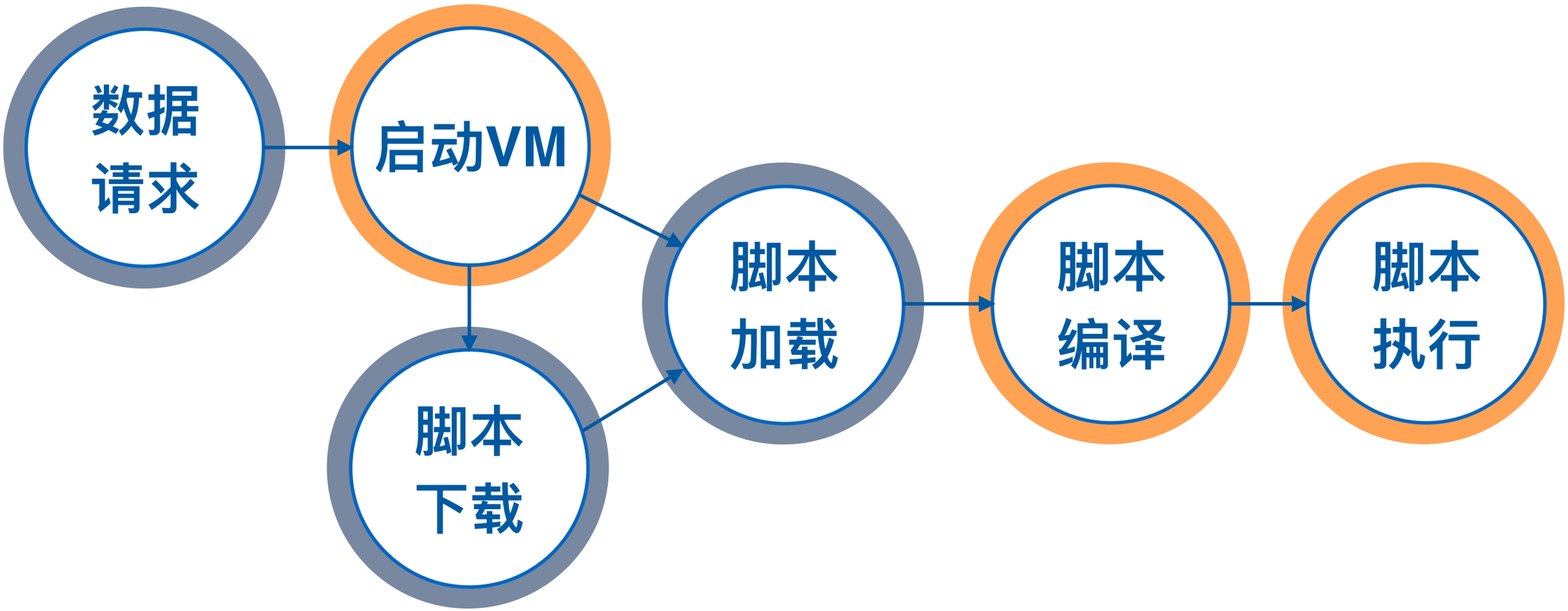
点击率提升10%

点击次数提升1.3次/人

TABLE OF
CONTENTS 大纲

- LuaView 的前世今生
- LuaView 的设计思想&技术架构
- LuaView 的性能优化
- LuaView 的未来

LuaView 页面加载流程



VM无关：不依赖Lua VM模块

VM相关：依赖Lua VM模块

VM无关的优化

数据请求

优化网络请求
接口依赖

- HTTP2、ACCS
- 先加载上次、后请求刷新

脚本下载

优化网络请求
减少包大小
预下载

- HTTP2、ACCS
- Simplify、混淆、合并
- 提前后台下载脚本

脚本编译

Bytecode预编译
Prototype缓存

- 编译阶段放到服务端
- 缓存中间结果

VM相关的优化

启动VM

- 异步VM
- LazyLoad
- Metatable缓存
  - VM创建过程和脚本加载并行
  - Lib库延迟加载
  - 多VM共用Metatable

脚本加载

- 减少包大小
- 代码合并
- 包缓存
  - 混淆、合并、资源压缩
  - 脚本内存缓存
  - 文件格式修改、减少IO

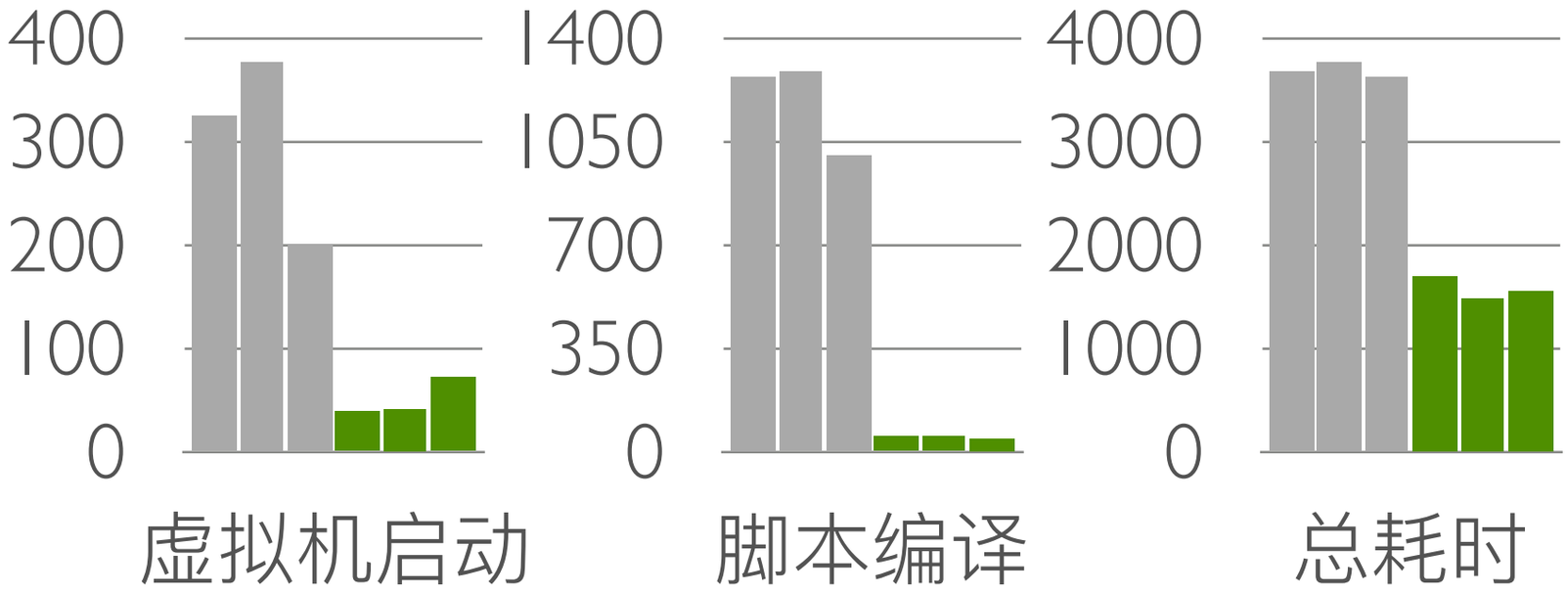
脚本执行

- VM替换
- 代码优化
- simplify
  - 使用LuaJIT
  - 优化布局代码，降低层级

VM相关的优化

15000行代码

阶段	day1-before	day2-before	day3-before	day1-after	day2-after	day3-after
数据请求	634.84	629.49	556.16	537.06	457.49	443.15
虚拟机启动	323.37	377.68	200.21	38.72	40.18	70.91
脚本下载&加载	1008.95	1029.03	1473.49	726.88	675.64	707.02
脚本编译	1263.75	1282.28	1005.42	51.81	54.90	44.47
脚本执行	447.64	449.44	380.18	334.26	259.12	291.35
总耗时	3678.55	3767.92	3615.46	1688.73	1487.33	1556.9



- VM启动性能提升
  - 300%+（3倍+）
- 下载加载性能提升
  - 138%+（1.3倍+）
- 编译耗时性能提升
  - 2000%+（20倍+）
- 脚本执行性能提升
  - 130%+（1.3倍+）
- 总耗时性能提升
  - 200%+（2倍+）

LuaJIT VS LuaJ

痛点：性能提升的瓶颈

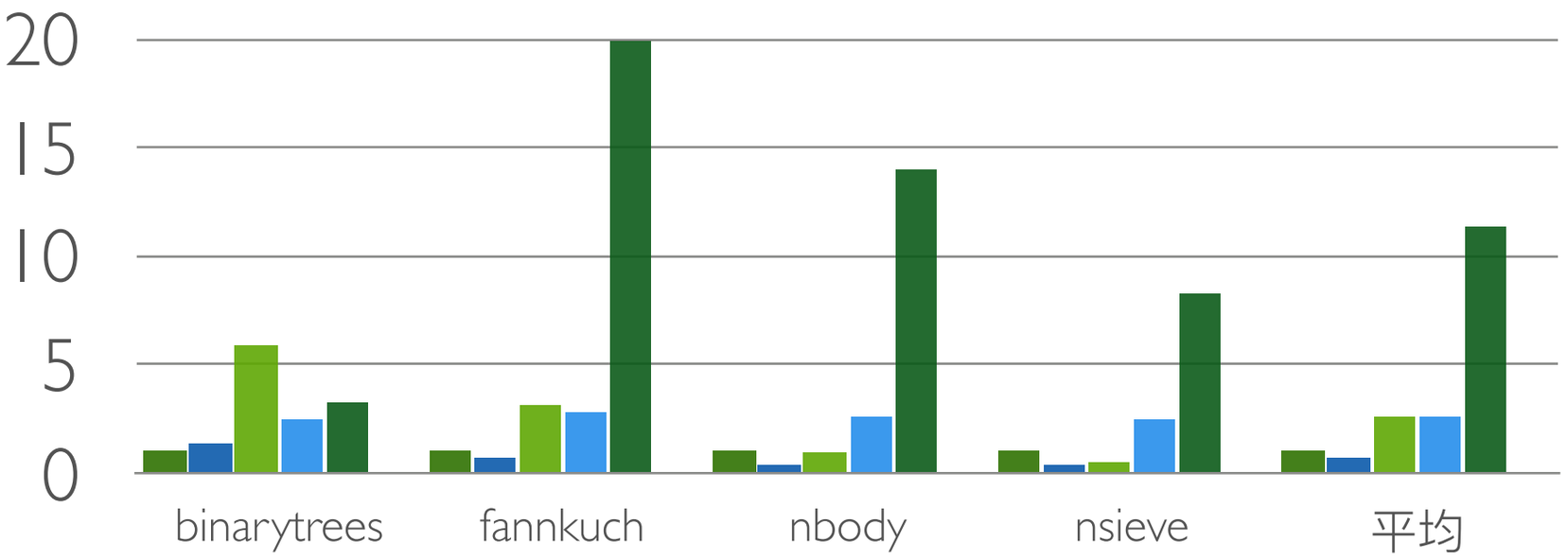
- 虚拟机启动
- 脚本编译
- 脚本执行

LuaJ、LuaJIT典型算法数据比较

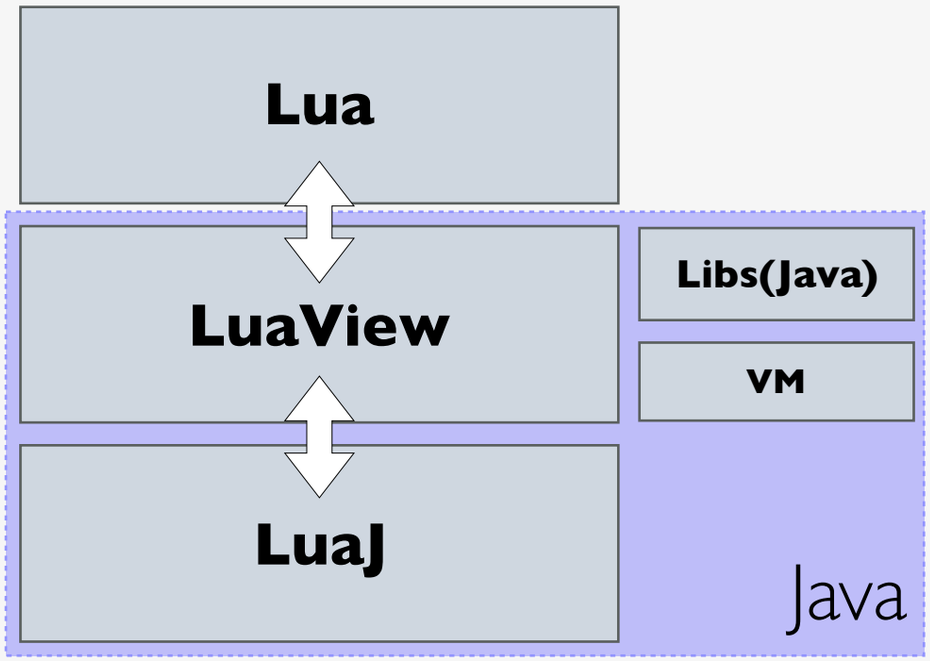
算法	Lua 5.1.4	LuaJ 3.0 (intercept)	LuaJ 3.0	LuaJIT 2.0.0 (intercept)	LuaJIT 2.0.0
binarytrees (15)	1	1.37	5.92	2.50	3.20
fannkuch (10)	1	0.69	3.16	2.77	19.89
nbody (1E+06)	1	0.41	0.91	2.58	13.97
nsieve (9)	1	0.36	0.49	2.47	8.27
平均值	1	0.71	2.62	2.58	11.33

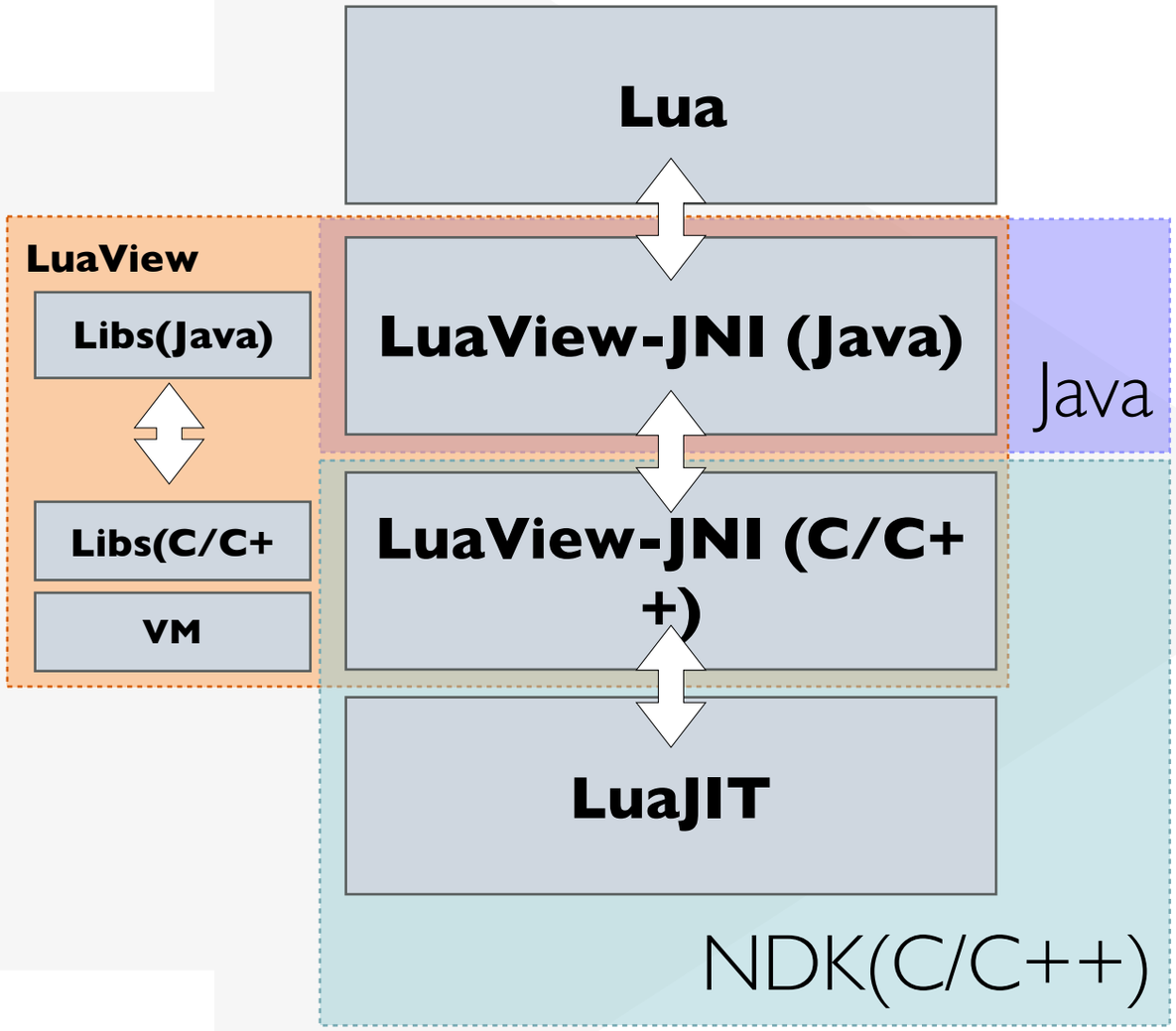
方案：性能对比LuaJIT2.0 & LuaJ3.0

- 解释执行模式下提升约3.63倍
- 编译执行模式下提升约4.32倍



架构变迁－LuaJ to LuaJIT+JNI





Java

C/C++/Java

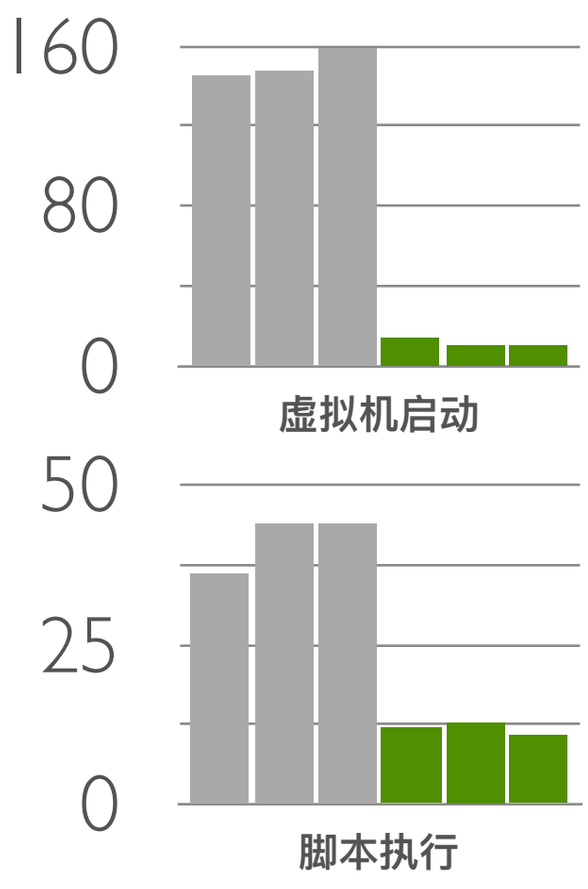
LuaJIT迁移碰到的那些坑 & JNI Tips

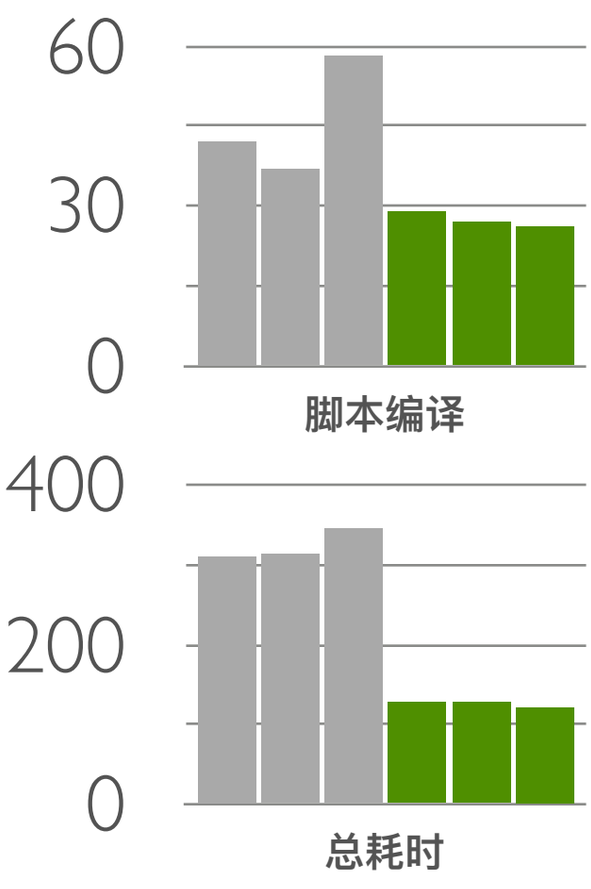
- **local reference table overflow (max=512)**
  - 方法体内Local Reference使用&释放
  - 递归调用Local Reference及时释放
- **global reference table overflow (max=51200)**
  - NewGlobalReference 替换成 jlong，在Java层通过Map索引对象
- **Lua GC**
  - __gc 及时清理Java对象
  - luaL_unref() 释放lua对象
- **Tips**
  - FindClass、GetMethodId耗时，做缓存
  - 尽量避免Java/C/C++环境切换

优化前后数据对比

首次启动页面耗时

阶段	LuaJ-case1	LuaJ-case2	LuaJ-case3	LuaJIT-case1	LuaJIT-case2	LuaJIT-case3
虚拟机启动	145	147	159	14	10	10
脚本编译	42	37	58	29	27	26
脚本执行	36	44	44	12	13	11
总耗时	310	314	344	127	127	120





- VM启动

  提升1000%+（10倍+）

- 编译

  提升135%+（1.35倍+）

- 执行

  提升300%+（3倍+）

- 总耗时

  提升240%+（2.4倍+）

TABLE OF
CONTENTS 大纲

未来发展

SDK: 高性能、小巧、扩展性

生态: 开源共建、社区

场景: 产品、低端设备、IOT等

附录

GitHub: https://github.com/alibaba/LuaViewSDK

WiKi: https://alibaba.github.io/LuaViewSDK

Demo:

Android

iOS

68697053
songtao.cst@alibaba-inc.com

LuaView QQ交流群

我的个人QQ

阿里技术官方微信

THANKS!