使用C++开发的App RingCentral

Glip: Team Chat

欢迎前往App Store or Google Play
下载体验

React从入门到精通

掌握当下最热门的前端利器

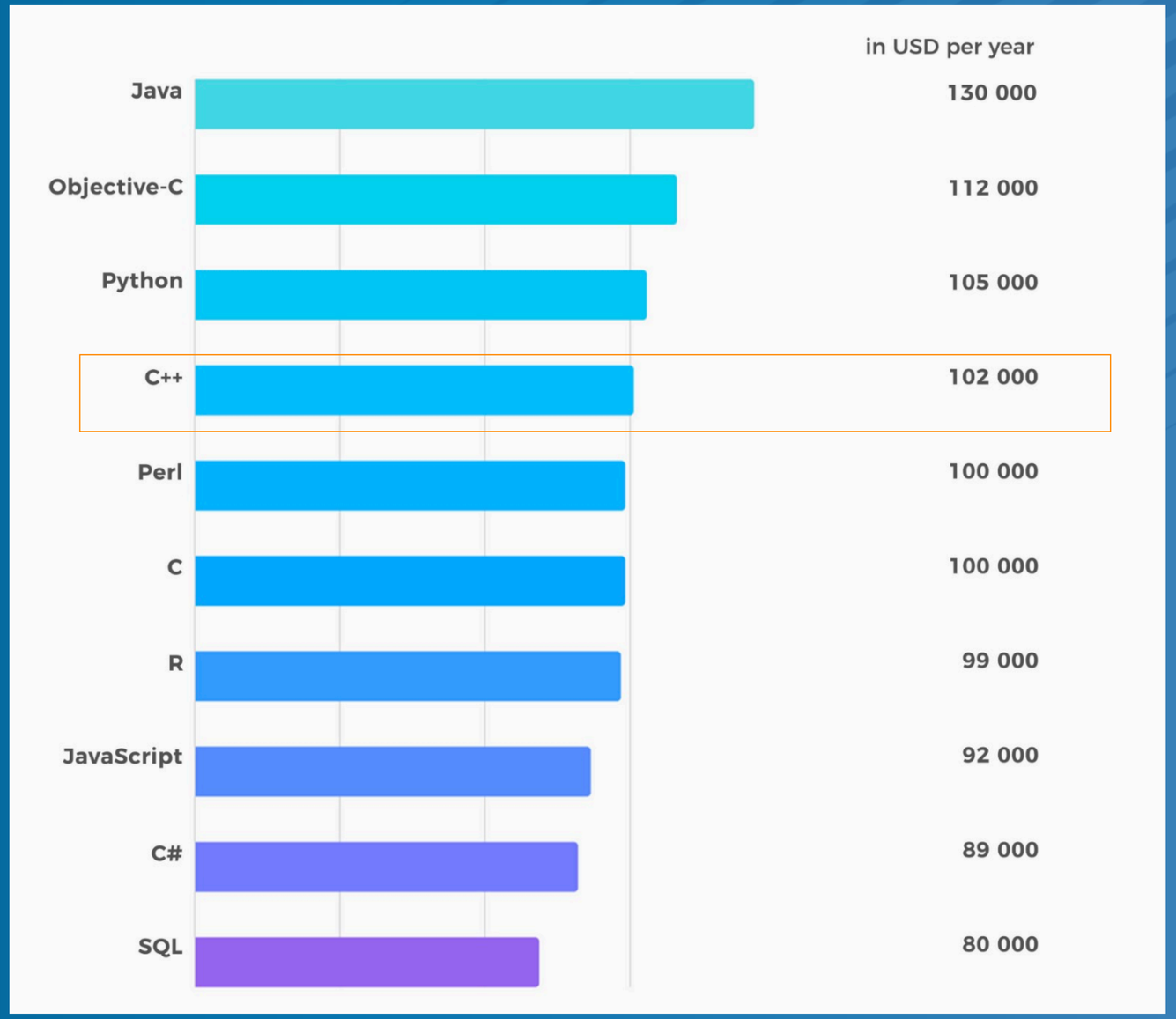
王沛
eBay 资深技术专家

你将获得

1. 全面学习 React 常用技术栈
2. 深入理解 React 设计模式
3. 常见场景下的编程实战指南
4. 掌握用 React 开发大型项目的能力

立即扫码，免费试看

移动开发薪资排行榜


in USD per year
Java
130 000
Objective-C
112 000
Python
105 000
C++
102 000
Perl
100 000
C
100 000
R
99 000
JavaScript
92 000
C#
89 000
SQL
80 000

RingCentral

基于djinni的跨平台App开发的实践分享

蒋伟 Jacob Jiang

RingCentral Collaboration Mobile Manager

Leading the World | Winning the Cloud

Unified Communication is New Future

- Leader in UCaaS from Silicon Valley
- RingCentral R&D center in Xiamen

A Gartner Magic Quadrant Leader for UCaaS 2017, 2016 & 2015

2017 STEVIE GOLD WINNER — International Business Awards

Forbes — 100 Most Trustworthy Companies in America 2016 & 2017

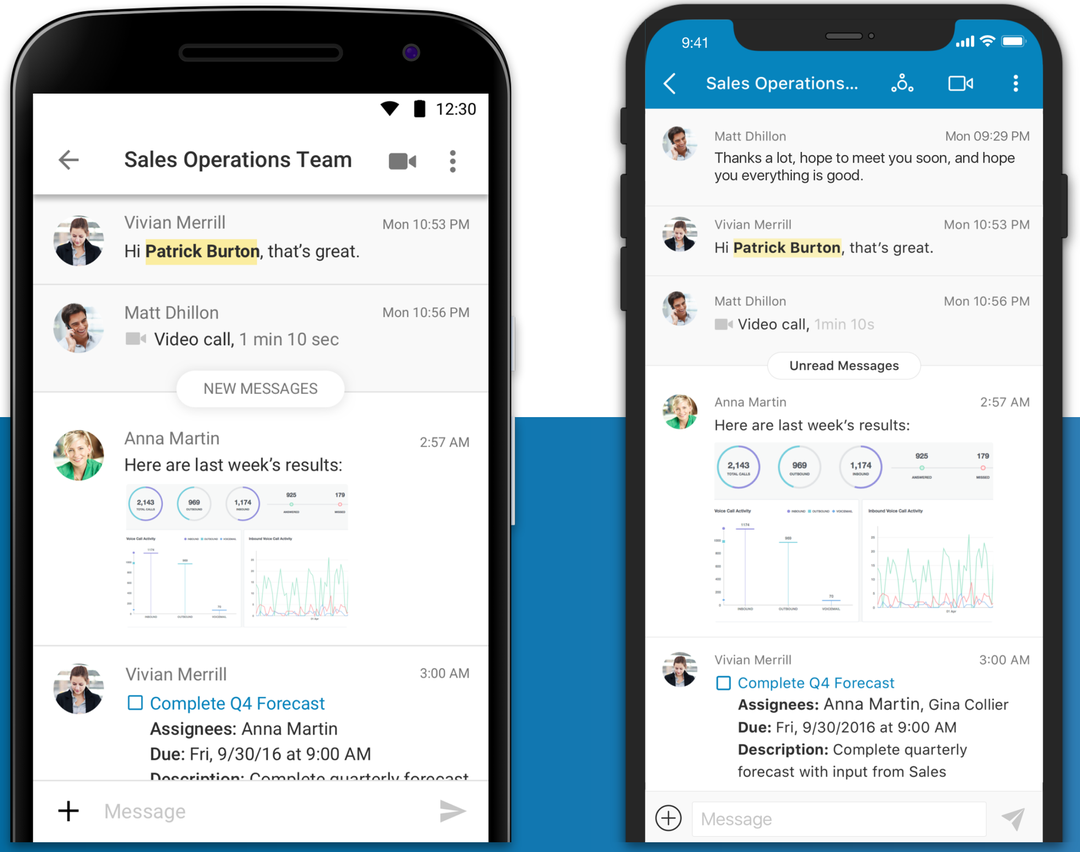
synergy research group — Top Cloud Communications Provider

NYSE — Listed on the NYSE

RingCentral®

RingCentral

Glip 团队协作平台



RingCentral

Glip Mobile产品需求

1. 团队一站式协作沟通平台，无缝集成email
2. PBX业务多，Cloud电话随时可用，在线传真
3. 视频会议、文件分享、评论
4. 高可用性、稳定

Glip Mobile产品需求协作平台

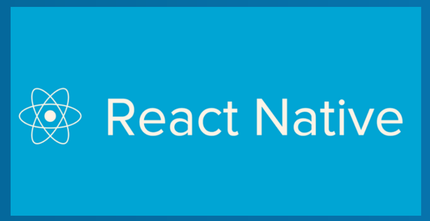
- Native方式各写一遍？
- 是否需要用跨平台技术？
- 完全自定义的跨平台还是部分跨平台？

原生界面 + 不想写2遍代码

- 首先决定是否需要用跨平台技术？ 是
- 完全自定义的跨平台还是部分跨平台？ 部分
- 前端界面跨平台还是底层逻辑跨平台？ 底层

跨平台方案非常多


Hybrid
Apps Development

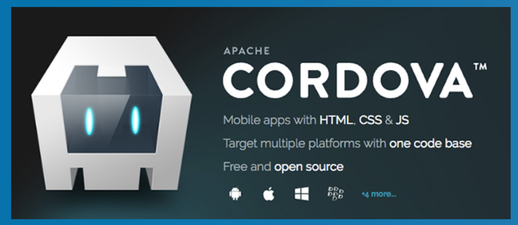

React Native


Flutter


WEEX


小程序


PWA


APACHE
CORDOVA™
Mobile apps with HTML, CSS & JS
Target multiple platforms with one code base
Free and open source
+4 more...

Djinni介绍

接口定义语言 IDL

自动生成桥接代码

- Java Proxy class, JNI Marshaling, Java <-> C++ (via JNI)
- Obj-C Interface, Obj-C <-> C++ (via Obj-C++)
- C++ abstract base class

开发实现C++ implementation

demo

```
item_list = record {
    items: list<string>;
}

sort_order = enum {
    ascending;
    descending;
    random;
}

sort_items = interface +c {
    # For the iOS / Android demo
    sort(order: sort_order, items: item_list);
    static create_with_listener(listener: textbox_listener): sort_items;

    # For the localhost / command-line demo
    static run_sort(items: item_list): item_list;
}

textbox_listener = interface +j +o {
    update(items: item_list);
}
```

RingCentral

Generated Code iOS & Android

```objc
// AUTOGENERATED FILE - DO NOT MODIFY!
// This file generated by Djinni from example.djinni

#import "TXSItemList.h"
#import "TXSSortOrder.h"
#import <Foundation/Foundation.h>
@class TXSSortItems;
@protocol TXSTextboxListener;


@interface TXSSortItems : NSObject

/** For the iOS / Android demo */
- (void)sort:(TXSSortOrder)order
       items:(nonnull TXSItemList *)items;

+ (nullable TXSSortItems *)createWithListener:(nullable id<TXSTextboxListener>)liste

/** For the localhost / command-line demo */
+ (nonnull TXSItemList *)runSort:(nonnull TXSItemList *)items;

@end
```

```java
/*package*/ abstract class SortItems {
    /** For the iOS / Android demo */
    public abstract void sort(@Nonnull SortOrder order, @Nonnull ItemList items);

    @CheckForNull
    public static native SortItems createWithListener(@CheckForNull TextboxListener listener);

    /** For the localhost / command-line demo */
    @Nonnull
    public static native ItemList runSort(@Nonnull ItemList items);

    private static final class CppProxy extends SortItems
    {
        private final long nativeRef;
        private final AtomicBoolean destroyed = new AtomicBoolean(false);

        private CppProxy(long nativeRef)
        {
            if (nativeRef == 0) throw new RuntimeException("nativeRef is zero");
            this.nativeRef = nativeRef;
        }

        private native void nativeDestroy(long nativeRef);
        public void destroy()
        {
            boolean destroyed = this.destroyed.getAndSet(true);
            if (!destroyed) nativeDestroy(this.nativeRef);
        }
        protected void finalize() throws java.lang.Throwable
        {
            destroy();
            super.finalize();
        }

        @Override
        public void sort(SortOrder order, ItemList items)
        {
            assert !this.destroyed.get() : "trying to use a destroyed object";
            native_sort(this.nativeRef, order, items);
        }
        private native void native_sort(long _nativeRef, SortOrder order, ItemList items);
    }
}
```

Generated Code C++

```cpp
// AUTOGENERATED FILE - DO NOT MODIFY!
// This file generated by Djinni from example.djinni

#pragma once

#include <memory>

namespace textsort {

class TextboxListener;
enum class sort_order;
struct ItemList;

class SortItems {
public:
    virtual ~SortItems() {}

    /** For the iOS / Android demo */
    virtual void sort(sort_order order, const ItemList & items) = 0;

    static std::shared_ptr<SortItems> create_with_listener(const std::shared_ptr<TextboxListener> & listener);

    /** For the localhost / command-line demo */
    static ItemList run_sort(const ItemList & items);
};

}  // namespace textsort
```

RingCentral

Handwrite Code C++

sort_items_impl.cpp

```
void SortItemsImpl::sort(sort_order order, const ItemList & items) {
    auto lines = items.items;
    switch (order) {
        case sort_order::ASCENDING: {
            std::sort(lines.begin(), lines.end(), std::less<std::string>());
            break;
        }
        case sort_order::DESCENDING: {
            std::sort(lines.begin(), lines.end(), std::greater<std::string>());
            break;
        }
        case sort_order::RANDOM: {
            std::shuffle(lines.begin(), lines.end(), std::default_random_engine{});
            break;
        }
    }

    // Pass result to client interface
    this->m_listener->update(ItemList(lines));
}

ItemList SortItems::run_sort(const ItemList & items) {
    auto lines = items.items;
    std::sort(lines.begin(), lines.end(), std::less<std::string>());
    return ItemList(lines);
}

}
```

Generated Code iOS & Android - Callback

TXSTextboxListenerImpl.h

```objc
#import "TXSTextboxListener.h"
#import <Foundation/Foundation.h>

@interface TXSTextboxListenerImpl : NSObject <TXSTextboxListener>

- (id)initWithUITextView:(UITextView *)textView;

@end
```

TextboxListener.java

```java
// AUTOGENERATED FILE - DO NOT MODIFY!
// This file generated by Djinni from example.djinni

package com.dropbox.textsort;

import javax.annotation.CheckForNull;
import javax.annotation.Nonnull;

/*package*/ abstract class TextboxListener {
    public abstract void update(@Nonnull ItemList items);
}
```

Generated Code C++ - Callback

textbox_listener.hpp

```cpp
// AUTOGENERATED FILE - DO NOT MODIFY!
// This file generated by Djinni from example.djinni

#pragma once

namespace textsort {

struct ItemList;

class TextboxListener {
public:
    virtual ~TextboxListener() {}

    virtual void update(const ItemList & items) = 0;
};

}  // namespace textsort
```

Handwrite Code iOS & Android - Callback

TextboxListenerImpl.java

```java
package com.dropbox.textsort;

import android.widget.EditText;

import java.util.ArrayList;

public class TextboxListenerImpl extends TextboxListener {

    private EditText mTextArea;

    public TextboxListenerImpl(EditText textArea) {
        this.mTextArea = textArea;
    }

    @Override
    public void update(ItemList items) {
        ArrayList<String> list = items.getItems();
        StringBuilder builder = new StringBuilder();
        for (String str : list) {
            builder.append(str);
            builder.append("\n");
        }
        mTextArea.setText(builder);
    }
}
```

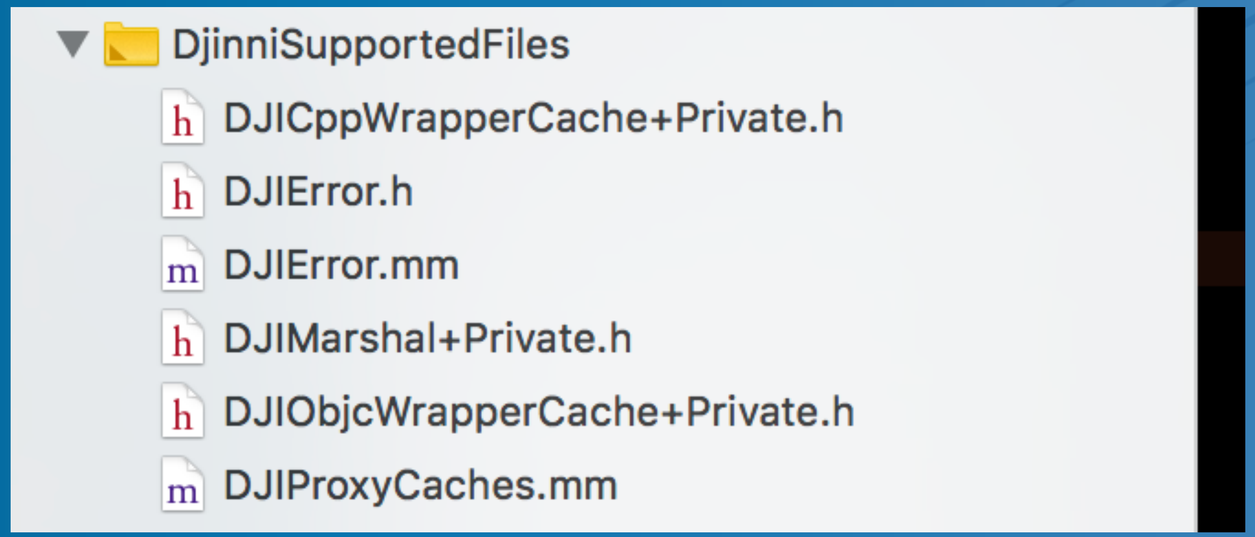
TXSTextboxListenerDebugableImpl.swift 〉 No Selection

```swift
import UIKit

final class TXSTextboxListenerDebugableImpl : NSObject, TXSTextboxListener {
    private var textView: UITextView

    @available(*, unavailable)
    override init() {
        fatalError("Unsupported")
    }

    @objc(initWithUITextView:)
    init(textView: UITextView) {
        self.textView = textView
    }

    func update(_ items: TXSItemList) {
        let string = items.items.joined(separator: "\n")
        print("TXSTextboxListenerDebugableImpl -> update \n\(string)")
        textView.text = string
    }
}
```

RingCentral

编译前提

1. 添加新增的文件、代码至工程
2. 添加djinni依赖库
3. Compile

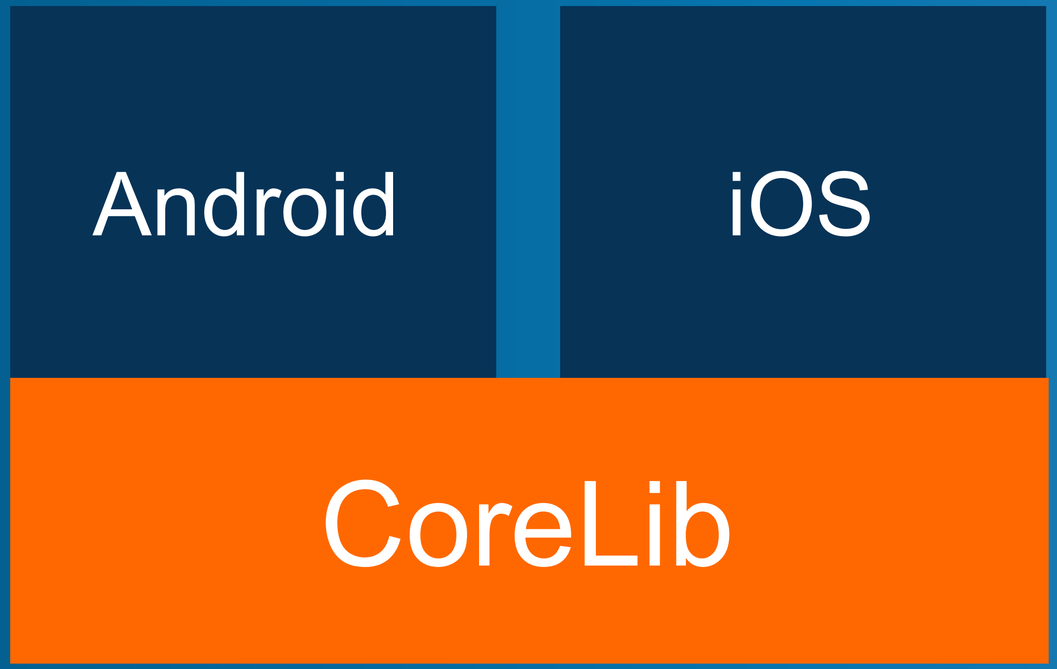


Djinni支持的主要数据类型

- Bool
- Fixed-Width Integers: i8, i16, i32, i64
- Double-Precision Floating-Point: f64
- String
- Binary
- list<T>, set<T>, map<K,V>
- optional<T>
- Date

Djinni编译工具

- iOS & OS X: LLVM 9.0 (Xcode 9.0)
- Android: Clang 3.8, NDK r14b (Android Studio)
- C++ 11

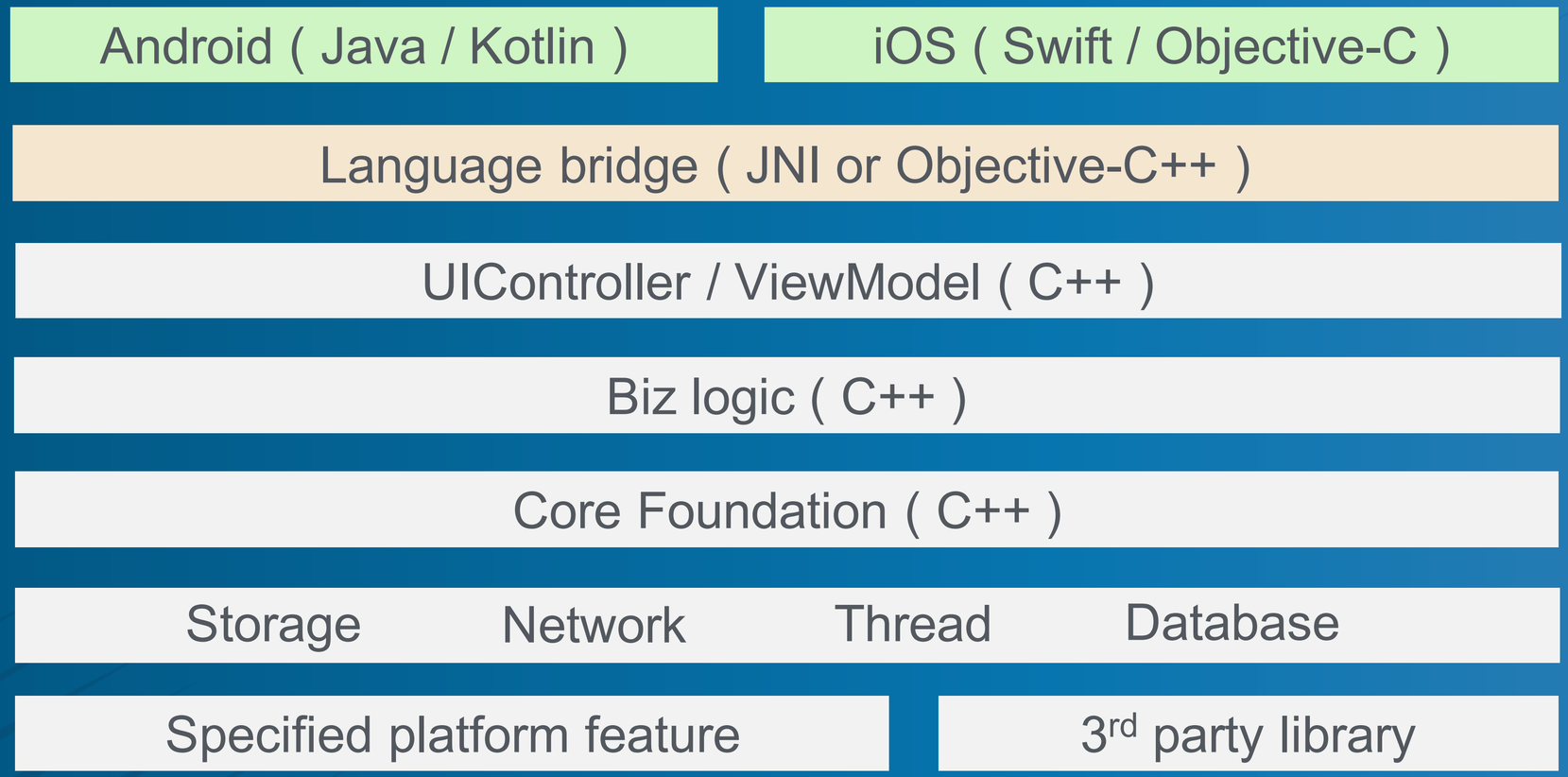

Android
iOS
CoreLib

RingCentral

Glip Mobile的架构

- UI - platform specific (Java or Obj-C)
- Language bridge(JNI or Obj-C++)
- Business logic (C++)

Native UI + 业务逻辑复用



C++ Libraries

坐享大量优质稳定的库

Websockets, openSSL, libcurl, sqlite, WebRTC, phoneparser, …

好处

- 功能跨平台一致（bug也是）
- 高性能


Http Client
Log
File Storage
Thread
Platform Specific library
Network State
Application State
Device Contact
Device Calendar
• • •

RingCentral

Http

- iOS AFNetworking
- Android HttpURLConnection

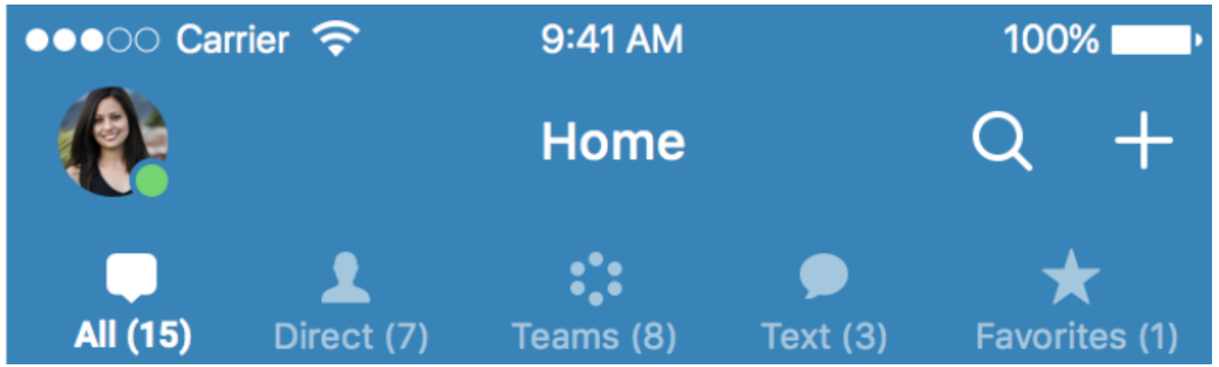
Log

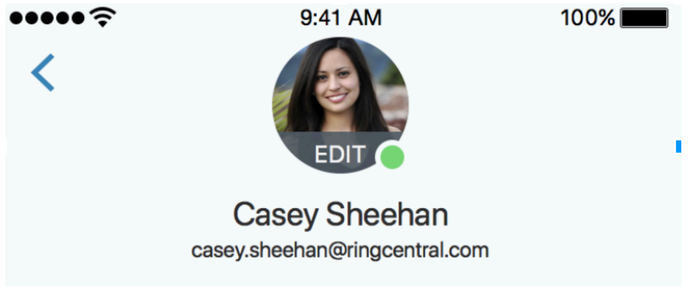
- iOS NSLog
- Android Logcat
- Log4CPP write log file

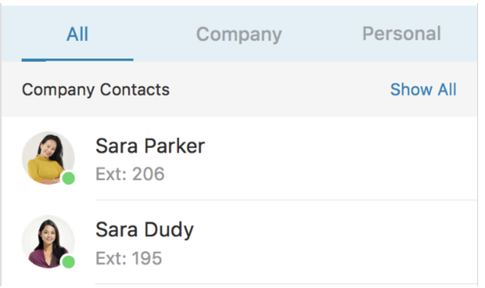
Thread

- iOS NSOperation
- Android ThreadPoolExecutor

Presence 例子


Carrier
9:41 AM
100%
Home
All (15)
Direct (7)
Teams (8)
Text (3)
Favorites (1)


9:41 AM
100%
EDIT
Casey Sheehan
casey.sheehan@ringcentral.com


All
Company
Personal
Company Contacts
Show All
Sara Parker
Ext: 206
Sara Dudy
Ext: 195

无需定制逻辑

presence_ui_controller.djinni

```
i_presence_delegate = interface +j +o {
    onPresenceChanged(presenceState: ePrensenceState);
}

i_presence_ui_controller = interface +c {
    setDelegate(delegate: i_presence_delegate);
    subscribe(personId: i64);
    unsubscribe();
    getPresence(personId: i64): ePrensenceState;
}
```

RingCentral

工作流

- 公共逻辑优先定义层间接口
- Git引用submodule关系，native/corelib同时开发
- Android直接使用编译库，直接收益

RingCentral

收益

- 整体开发效率提升了2倍
- 部门间沟通效率提升
- 代码质量更稳定
- 模块化、编译速度快

RingCentral

代码量节约8W行

- Swift/Obj-C 11W
- Kotlin/Java 10W
- C++ 4W 业务代码，头文件8W

RingCentral

A Gartner UCaaS Magic Quadrant Leader. Again.

RingCentral positioned furthest for completeness of vision

RingCentral Named #1 UCaaS Scorecard Leader 2018

Learn more ›

IHS Markit
2018
UCAAS
SCORECARD
LEADER

平台差异性

- Bundle resource path
- Network (background session)
- Thread (Thread is different on iOS /Android)
- App background states

编译时间优化

大量的djinni头文件、*.mm文件

反复引用，编译时间太长

4分钟->10秒

Glip 〉 GlipCore.xcodeproj 〉 OCDjinniInterface 〉 gen 〉 oc_unity_build.mm 〉 No S

```
1  #import "GlipAvUtils+Private.mm"
2  #import "GlipAvatar+Private.mm"
3  #import "GlipAvatar.mm"
4  #import "GlipBackgroundModeBuilder+Private.mm"
5  #import "GlipButtonStateInfo+Private.mm"
6  #import "GlipButtonStateInfo.mm"
7  #import "GlipCallControlStateInfo+Private.mm"
8  #import "GlipCallControlStateInfo.mm"
9  #import "GlipChangedAttributeValue+Private.mm"
10 #import "GlipChangedAttributeValue.mm"
11 #import "GlipCommonErrorCode+Private.mm"
12 #import "GlipCommonErrorCode.mm"
13 #import "GlipContactSection+Private.mm"
14 #import "GlipContactSection.mm"
15 #import "GlipCoreUtil+Private.mm"
16 #import "GlipCrashProducer+Private.mm"
17 #import "GlipDialingPlanCountryModel+Private.mm"
18 #import "GlipDownloadUtil+Private.mm"
19 #import "GlipECloudContactEditStatus+Private.mm"
20 #import "GlipECloudContactEditStatus.mm"
21 #import "GlipECloudContactFieldSize+Private.mm"
22 #import "GlipECloudContactFieldSize.mm"
23 #import "GlipEventDataModel+Private.mm"
24 #import "GlipGlipIdType+Private.mm"
25 #import "GlipGlipIdType.mm"
26 #import "GlipIAVUiController+Private.mm"
27 #import "GlipIAccount+Private.mm"
28 #import "GlipIAccountSettingCallback+Private.mm"
29 #import "GlipIAccountSettingUiControllerDelegate+Private.mm"
30 #import "GlipIAccountSettingsUiController+Private.mm"
```

性能优化

- 启动库太庞大，启动时间占用比例大
- 动态库转静态库处理
- C++宏定义静态

Glip 〉 GlipCore.xcodeproj 〉 GlipCore 〉 constant 〉 constant.cpp 〉 No Selection

```
//

#include "./constant.h"

namespace glip_mobile {

    using namespace std;

    //Global configuration key
    const string kGlipAppExternalId = "com.glip.mobile";
    const string kGlipDeviceToken                    = "deviceToken";
    const string kGlipVoipDeviceToken  = "voip_device_token";
    const string kGlipIsSandboxAPNSEnvironment       = "is_sandbox_APNS_env";
    const string kGlipUUID = "uuid";
    const string kGlipVoipUUID = "voip_uuid";
    const string kRcEndpointId = "endpoint_id";
    const string kGlipSipProvisionDeviceId = "sip_provision_device_id";
    const string kGlipAppVersion = "app_version";
    const string kMthorAppVersionKey = "mthor_app_version";
    const int64_t kMthorAppVersionValue = 1;
    const string kGlipIsRcMeetingInstalled = "is_rc_meeting_installed";
    const string kGlipIsRcMobileInstalled = "is_rc_mobile_installed";
    const string kGlipForceUpgradeVersion = "GlipForceUpgradeVersion";


    const string kGlipConfigJsonFileNameValue  = "glipconfig.json";
    const string kGlipUpgradeToMthorFile = "upgrade_to_mthor_config.json";
```

短板

- 语言层面没有block支持，需要写大量delegate
- Share_ptr强引用，导致delegate与ViewController互为强引用
- 类文件生成较多
- Djinni类无法支持继承

RingCentral

回顾

- 为什么选择使用C++跨平台
- djinni工具的使用
- 使用djinni的产品架构
- 优缺点

RingCentral

总结

跨平台C++移动App开发效率高、速度快、不牺牲原生界面体验

枯燥无味的接口编写工作被djinni取代

一次开发 多平台复用

极客邦企业培训与咨询

帮助企业与技术人成长

精品课程

Excellent Course

- ☑《互联网大规模分布式架构设计与实践》
- ☑《基于大数据的企业运营与精准营销》
- ☑《大数据和人工智能在金融领域的应用》
- ☑《区块链应用与开发技术高级培训》
- ☑《通往卓越管理的阶梯》

扫码关注官方微信服务号
了解更多课程详细信息

Geekbang
极客邦科技

RingCentral

THANK YOU

厦门铃盛软件有限公司