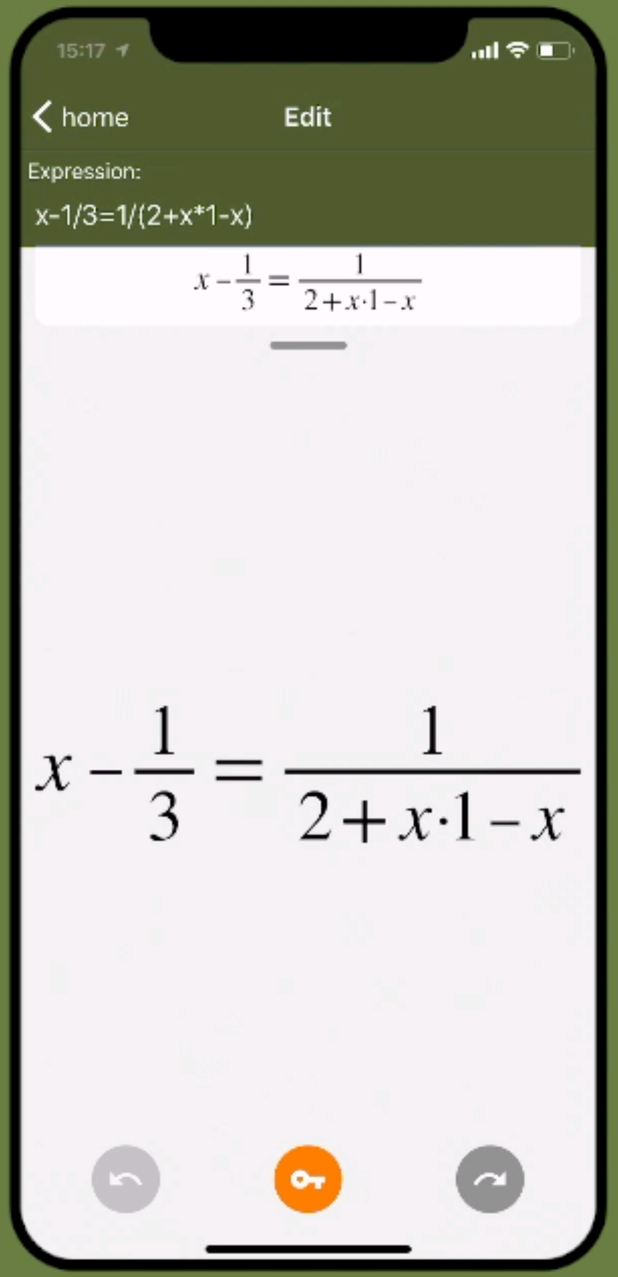
Lars Melchior

Expanding The Boundaries Of React-Native With C++

Going from a cross-platform graphical C++ app to a full-scale mobile application

Maphi

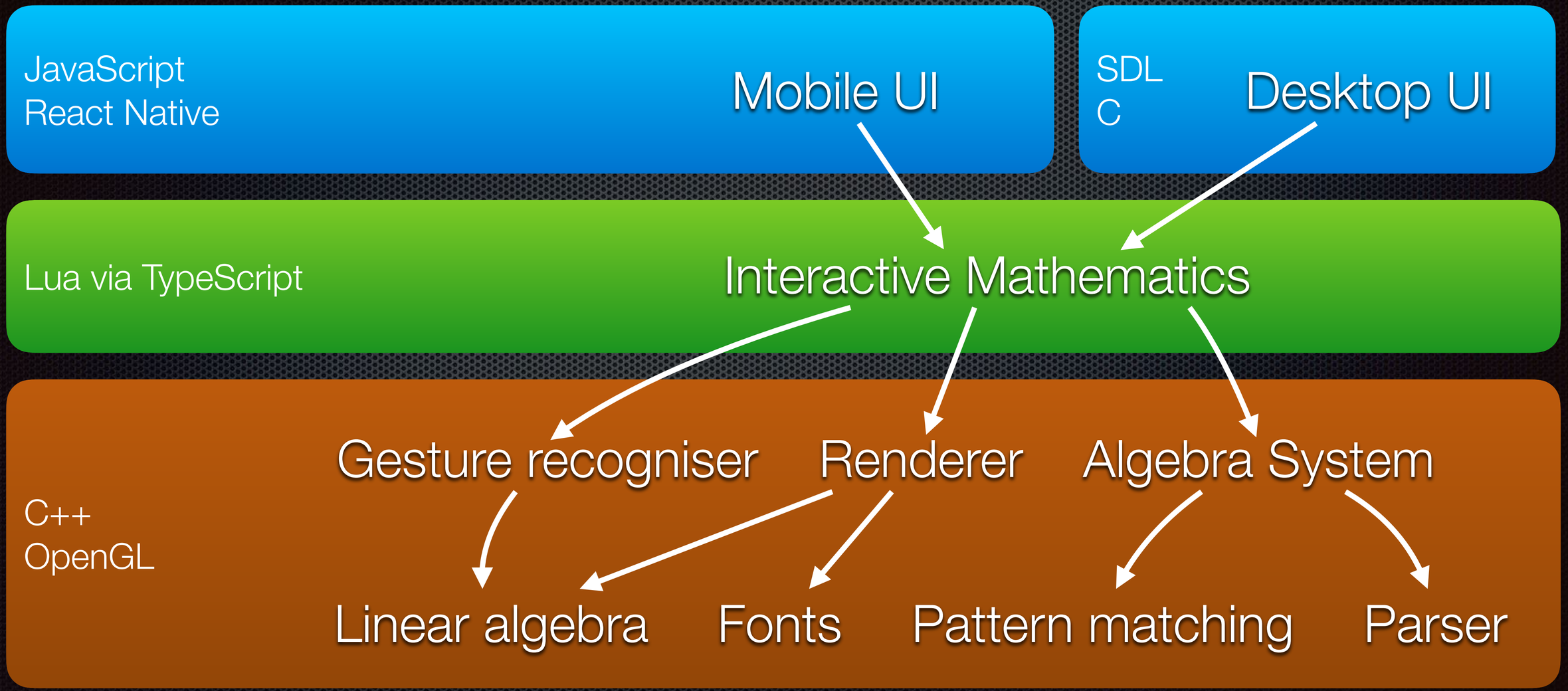

15:17
home
Edit
Expression:
x-1/3=1/(2+x*1-x)
$x-\frac{1}{3}=\frac{1}{2+x\cdot 1-x}$
$x-\frac{1}{3}=\frac{1}{2+x\cdot 1-x}$

Maphi

	Fast	Platform agnostic	Native UI	Rapid development	Long-Term
Cocoa Touch Swift / Obj C	✓	x	✓	o	✓
Android SDK Java / Kotlin	✓	x	✓	o	✓
Ionic JavaScript	x	o	x	✓	✓
React Native JavaScript	x	o	✓	✓	x
Flutter Dart	✓	o	x	✓	x
Bare Metal C++ / OpenGL	✓	✓	x	x	✓

	Fast	Platform agnostic	Native UI	Rapid development	Long-Term
Cocoa Touch Swift / Obj C	✓	x	✓	o	✓
Android SDK Java / Kotlin	✓	x	✓	o	✓
Ionic JavaScript	x	o	x	✓	✓
React Native JavaScript	x	o	✓	✓	x
Flutter Dart	✓	o	x	✓	x
Bare Metal C++ / OpenGL	✓	✓	x	x	✓

Project Structure



C++ Dependency Management

- Install (apt-get, dnf, brew, …)
  - Not reproducible
  - not cross-platform
- Add source code explicitly
  - Difficult to update and maintain
- Git submodules
  - Breaks transitive dependencies
- Package managers (Conan, vcpkg, Hunter, etc)
  - Dependencies must be packaged / registered
  - Version freezing adds extra complexity
- CMake
  - Verbose

Dependency Management

- JavaScript
  - npm / yarn

package.json

```
{
  "main": "node_modules/expo/AppEntry.js",
  "scripts": {
    "start": "expo start",
    "android": "expo start --android",
    "ios": "expo start --ios",
    "web": "expo start --web",
    "eject": "expo eject",
    "test": "jest"
  },
  "dependencies": {
    "expo": "^35.0.0",
    "jest": "^24.9.0",
    "react": "16.8.3",
    "react-dom": "16.8.3",
    "react-native": "https://github.com/expo/react-native/archive/s
    "react-native-web": "^0.11.7"
  },
  "devDependencies": {
    "babel-preset-expo": "^7.0.0"
  },
  "private": true
}
```

Dependency Management

- JavaScript
  - npm / yarn
- C++
  - CMake
    - find_package
      - FindFoo.cmake?
      - conan / vcpkg?
      - Version?
      - Cross-compiling?

CMakeLists.txt

```
cmake_minimum_required(VERSION 2.6)
project(Foo)

# Library sources



# find package Foo
find_package(Foo REQUIRED)



# add binary
add_executable(bar bar.cpp)

# add library
target_link_libraries(bar Foo)

# install
install(
  TARGETS bar
  RUNTIME DESTINATION bin
  COMPONENT "${INSTALL_BIN_DIR}"
)
```

Dependency Management

- JavaScript
  - npm / yarn
- C++
  - CMake
    - CPM

https://github.com/TheLartians/CPM.cmake

CMakeLists.txt

```
cmake_minimum_required(VERSION 2.6)
project(Foo)

# Library sources
include(cmake/CPM.cmake)

CPMAddPackage(
  NAME Foo
  GITHUB_REPOSITORY TheLartians/Foo
  VERSION 1.1
)

# add binary
add_executable(bar bar.cpp)

# add library
target_link_libraries(bar Foo)

# install
install(
  TARGETS bar
  RUNTIME DESTINATION bin
  COMPONENT "${INSTALL_BIN_DIR}"
)
```

Platform-specific considerations

- iOS
  - Resource path determined by OS [[NSBundle mainBundle] resourcePath]
  - Certain C++17 features won’t compile for old iOS versions

```
main.cpp:27:5: Call to unavailable function 'visit': introduced in iOS 12
```

```
main.cpp:27:5: Call to unavailable function ‘get': introduced in iOS 12
```

Platform-specific considerations

- iOS
  - Resource path determined by OS [[NSBundle mainBundle] resourcePath]
  - Certain C++17 features won’t compile for old iOS versions
- Android
  - OpenGL can loose context anytime
  - Java garbage collection runs on dedicated thread
  - No file handle to resources

Handling resources

- Let build system handle resources
- Write a platform-dependent resource manager

```
class Resource {
public:
  virtual std::unique_ptr<std::istream> stream() const = 0;
  virtual ~Resource() {}
};

std::shared_ptr<const Resource> loadResource(std::string name);
```

DEMO

DEMO

CPM

Glue

TypeScript

Widgets

React Native

JavaScript

Layout engine
(C++)

Platform UI
(Java)
(Objective C)

JSON

Hooking up C++ with iOS


```
auto getCLIArguments() {
  std::vector<std::string> arguments;
  for (NSString *string in [[NSProcessInfo processInfo] arguments]) {
    arguments.emplace_back([string UTF8String]);
  }
  return arguments;
}
```

Hooking up C++ with Android


```
@Namespace("maphi")
@Name(“Any")
public static class GlueAny extends Pointer {
  ...
  @Name(“get<std::shared_ptr<glue::Map>>")
  public native @ByVal GlueMap asGlueMap();
  ...
}
```

Hooking up with React Native

Hooking up with React Native

Create new views

iOS

```
- (UIView *) view {
  return [MaphiView new];
}
```

Android

```
override fun createViewInstance(reactContext: ThemedReactContext): MaphiView {
    return MaphiView()
}
```

Hooking up with React Native

Passing properties

iOS

```
RCT_EXPORT_VIEW_PROPERTY(expression, NSString *)
```

```
- (void) setInitialExpression: (NSString *)expression {
  updateProperty("expression", std::string([expression UTF8String]));
}
```

Android

```
@ReactProp(name = "expression")
fun setExpression(view: MaphiView, expression: String) {
  view.setProperty("expression", expression)
}
```

Hooking up with React Native

Passing callbacks

iOS

```
@property (nonatomic, copy) RCTBubblingEventBlock onExpressionChanged;
```

```
RCT_EXPORT_VIEW_PROPERTY(onPresentQuiz, RCTBubblingEventBlock)
```

Android

```
override fun getExportedCustomBubblingEventTypeConstants(): MutableMap<String, Any>? {
  return MapBuilder.builder<String, Any>()
    .put(
      "onExpressionChanged",
      MapBuilder.of(
        "phasedRegistrationNames",
        MapBuilder.of("bubbled", “onExpressionChanged”)))
...
```

DEMO

Summary and Outlook

App Core

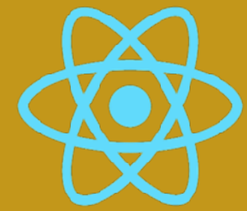
C++

Platform Support

Objective C

Java

React Native



JavaScript

Summary and Outlook

App Core

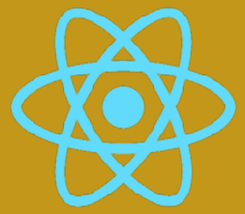
C++

Platform Support

Objective C

Java

React Native



JavaScript

React Native Core

C++

Objective C

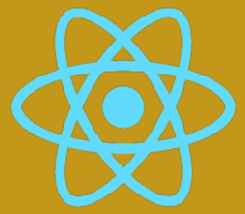
Java

Summary and Outlook

App Core

C++

React Native



JavaScript

React Native Core

C++
Objective C
Java

Outlook

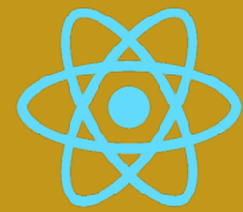
App Core

C++

React Native Core

C++
Objective C
Java

React Native



JavaScript

JavaScript Interface (JSI)

```
auto SampleCxxModule::getMethods() -> std::vector<Method> {
  return {
    Method(
      “addIfPositiveAsPromise",
      [](dynamic args, Callback cb, Callback cbError) {
        auto a = jsArgAsDouble(args, 0);
        auto b = jsArgAsDouble(args, 1);
        if (a < 0 || b < 0) {
          cbError({"Negative number!"});
        } else {
          cb({a + b});
        }
      }
    )
  };
}
```

TurboModules

Coming Soon !

Q&A

github.com/TheLartians

linkedin.com/in/lars-melchior

@LarsMelchior_

www.maphi.app