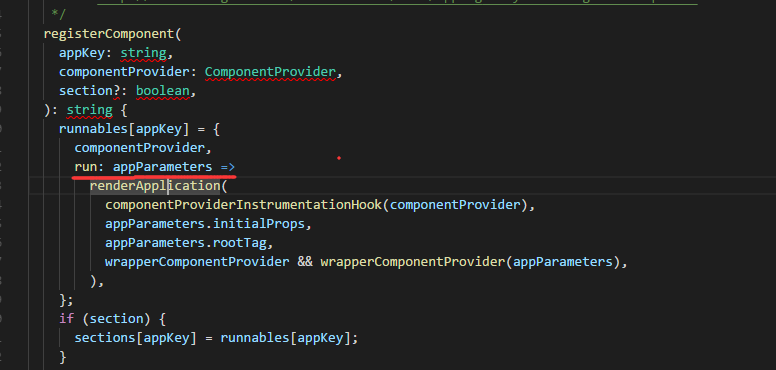
通过AppRegistry注册组件

AppRegistry.registerComponent('Test', () => Test);

内部会生成一个runnable对象，包含一个run函数，组件渲染时会调用该函数



run函数调用react-native\Libraries\ReactNative\renderApplication.js的renderApplication

然后调用react-native\Libraries\Renderer\ReactNativeRenderer-dev.js中的ReactNative.render(renderable, rootTag)

ReactNative.render(renderable, rootTag)

getReactNativeHost().getReactInstanceManager()->createReactInstanceManager

mReactRootView.startReactApplication触发创建ReactApplicationContext，

**if** (!mReactInstanceManager.hasStartedCreatingInitialContext()) {  
 mReactInstanceManager.createReactContextInBackground();  
}

com.facebook.react.XReactInstanceManagerImpl#createReactContext

初始化js，注册module

两端各持有一份相同的config，config中有一些已经注册的模块，两端的通信就是通过传输这样的“A”，“B”或者”C”来实现的。这个config对应到RN的代码是NativeModuleRegistry和JavaScriptModuleRegistry。

崩溃地址

com.facebook.react.bridge.CatalystInstanceImpl#CatalystInstanceImpl构造函数

调用

com.facebook.react.bridge.CatalystInstanceImpl#initializeBridge

到c++层

CatalystInstanceImpl::initializeBridge

instance\_->initializeBridge

Instance::initializeBridge-》jsQueue->runOnQueueSync

JMessageQueueThread::runOnQueueSync

到java层

com.facebook.react.bridge.queue.MessageQueueThreadImpl#runOnQueue

然后抛到

mqt\_jsjava线程，执行

com.facebook.react.bridge.queue.MessageQueueThreadHandler#dispatchMessage