本文出自《React Native学习笔记》系列文章。

在做React Native开发时，少不了的需要对React Native程序进行调试。调试程序是每一位开发者的基本功，高效的调试不仅能提高开发效率，也能降低Bug率。本文将向大家分享React Native程序调试的一些技巧和心得。

Developer Menu

Developer Menu是React Native给开发者定制的一个开发者菜单，来帮助开发者调试React Native应用。

> 提示：生产环境release (production) 下Developer Menu是不可用的。

如何开启Developer Menu

在模拟器上开启Developer Menu

Android模拟器：

可以通过 `Command⌘ + M` 快捷键来快速打开Developer Menu。也可以通过模拟器上的菜单键来打开。

> 心得：高版本的模拟器通常没有菜单键的，不过Nexus S上是有菜单键的，如果想使用菜单键，可以创建一个Nexus S的模拟器。

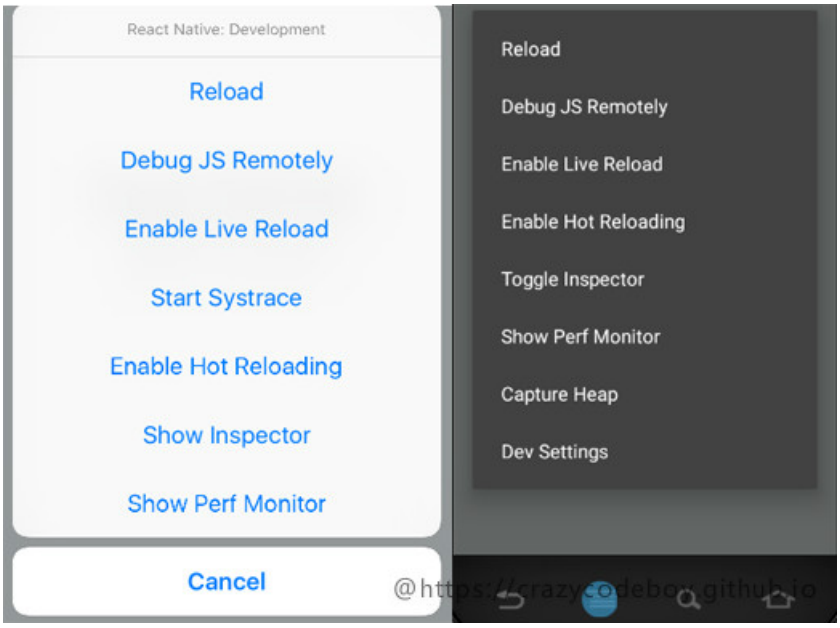
iOS模拟器：

可以通过 `Command⌘ + D` 快捷键来快速打开Developer Menu。

在真机上开启Developer Menu：

在真机上你可以通过摇动手机来开启Developer Menu。

预览图



Reloading JavaScript

在只是修改了js代码的情况下，如果要预览修改结果，你不需要重新编译你的应用。在这种情况下，你只需要告诉React Native重新加载js即可。

> 提示：如果你修改了native 代码或修改了Images.xcassets、res/drawable中的文件，重新加载js是不行的，这时你需要重新编译你的项目了。

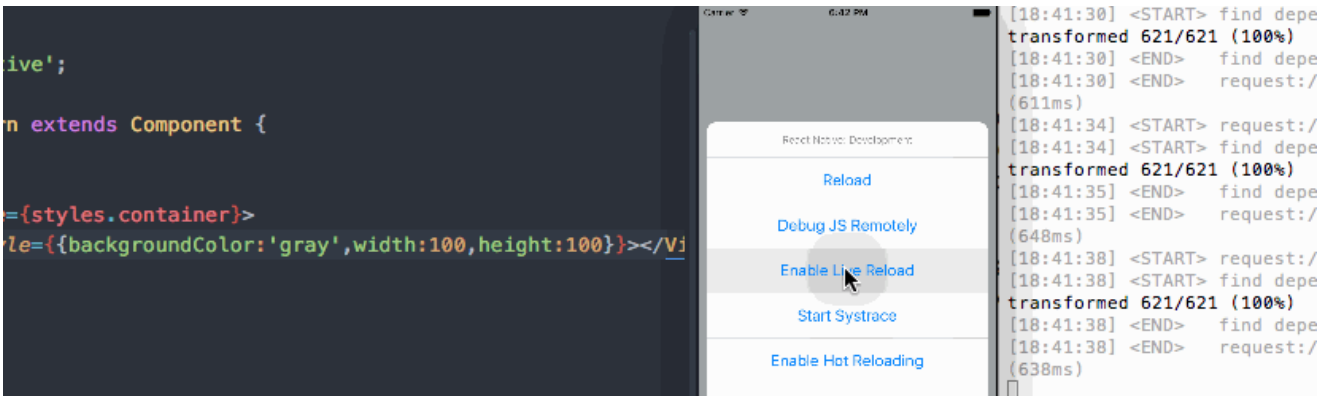
Reload js

Reload js即将你项目中js代码部分重新生成bundle，然后传输给模拟器或手机。
在Developer Menu中有 `Reload` 选项，单击 `Reload` 让React Native重新加载js。对于iOS模拟器你也可以通过 `Command⌘ + R` 快捷键来加载js，对于Android模拟器可以通过双击 `r` 键来加载js。

> 提示：如果 `Command⌘ + R` 无法使你的iOS模拟器加载js，则可以通过选中Hardware menu中Keyboard选项下的 "Connect Hardware Keyboard" 。

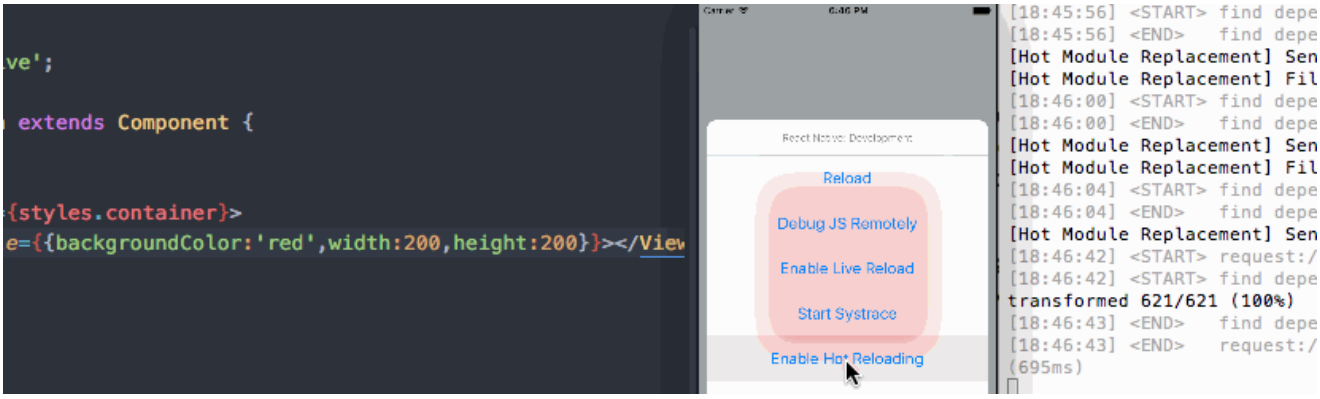
小技巧：Automatic reloading

Enable Live Reload



React Native旨在为开发者带来一个更好的开发体验。如果你觉得上文的加载js代码方式太low了或者不够方便，那么有没有一种更简便加载js代码的方式呢？
答案是肯定的。
在 Developer Menu中你会看到"Enable Live Reload" 选项，该选项提供了React Native动态加载的功能。当你的js代码发生变化后，React Native会自动生成bundle然后传输到模拟器或手机上，是不是觉得很方便。

Hot Reloading



另外，Developer Menu中还有一项需要特别介绍的，就是"Hot Reloading"热加载，如果说Enable Live Reload解放了你的双手的话，那么Hot Reloading不但解放了你的双手而且还解放了你的时间。 当你每次保存代码时Hot Reloading功能便会生成此次修改代码的增量包，然后传输到手机或模拟器上以实现热加载。相比 Enable Live Reload需要每次都返回到启动页面，Enable Live Reload则会在保持你的程序状态的情况下，就可以将最新的代码部署到设备上，听起来是不是很疯狂呢。

> 提示：当你做布局的时候启动Enable Live Reload功能你就可以实时的预览布局效果了，这可以和用AndroidStudio或AutoLayout做布局的实时预览相媲美。

Errors and Warnings

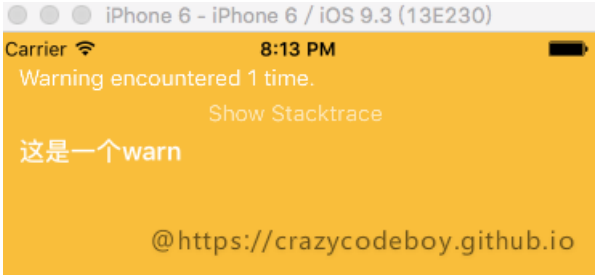
在development模式下，js部分的Errors 和 Warnings会直接打印在手机或模拟器屏幕上，以红屏和黄屏展示。

Errors

React Native程序运行时出现的Errors会被直接显示在屏幕上，以红色的背景显示，并会打印出错误信息。 你也可以通过 `console.error()` 来手动触发Errors。

Warnings

React Native程序运行时出现的Warnings也会被直接显示在屏幕上，以黄色的背景显示，并会打印出警告信息。 你也可以通过 `console.warn()` 来手动触发Warnings。 你也可以通过 `console.disableYellowBox = true` 来手动禁用Warnings的显示，或者通过 `console.ignoredYellowBox = ['Warning: ...'];` 来忽略相应的Warning。



> 提示：在生产环境release (production)下Errors和Warnings功能是不可用的。

Chrome Developer Tools

Chrome 开发工具

谷歌 Chrome 开发工具，是基于谷歌浏览器内含的一套网页制作和调试工具。开发者工具允许网页开发者深入浏览器和网页应用程序的内部。该工具可以有效地追踪布局问题，设置 JavaScript 断点并可深入理解代码的最优化策略。 Chrome 开发工具一共提供了8大组工具：

- Element 面板： 用于查看和编辑当前页面中的 HTML 和 CSS 元素。
- Network 面板：用于查看 HTTP 请求的详细信息，如请求头、响应头及返回内容等。
- Source 面板：用于查看和调试当前页面所加载的脚本的源文件。
- TimeLine 面板： 用于查看脚本的执行时间、页面元素渲染时间等信息。
- Profiles 面板：用于查看 CPU 执行时间与内存占用等信息。
- Resource 面板：用于查看当前页面所请求的资源文件，如 HTML，CSS 样式文件等。
- Audits 面板：用于优化前端页面，加速网页加载速度等。
- Console 面板：用于显示脚本中所输出的调试信息，或运行测试脚本等。

> 提示：对于调试React Native应用来说，Sources和Console是使用频率很高的两个工具。

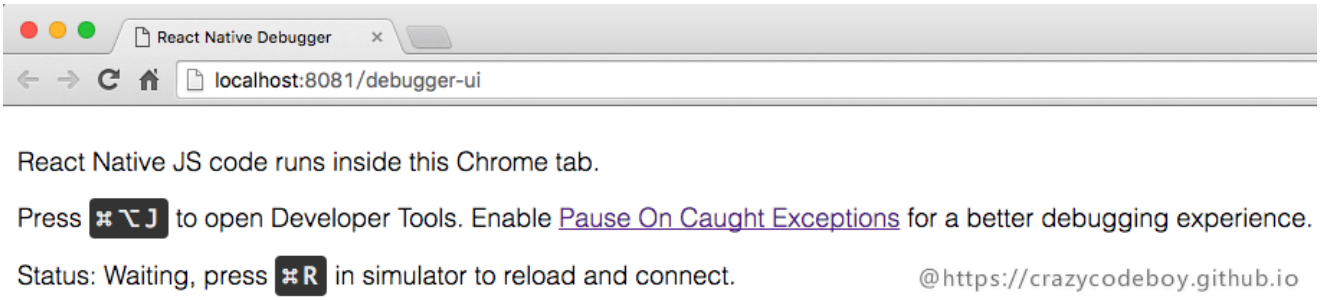
你可以像调试JavaScript代码一样来调试你的React Native程序。

如何通过 Chrome调试React Native程序

你可以通过以下步骤来调试你的React Native程序：

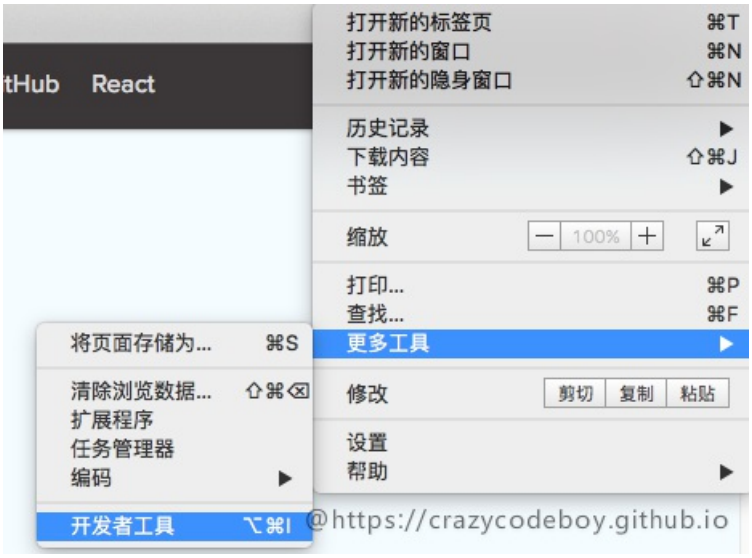
第一步：启动远程调试

在Developer Menu下单击"Debug JS Remotely" 启动JS远程调试功能。此时Chrome会被打开，同时会创建一个“http://localhost:8081/debugger-ui.” Tab页。

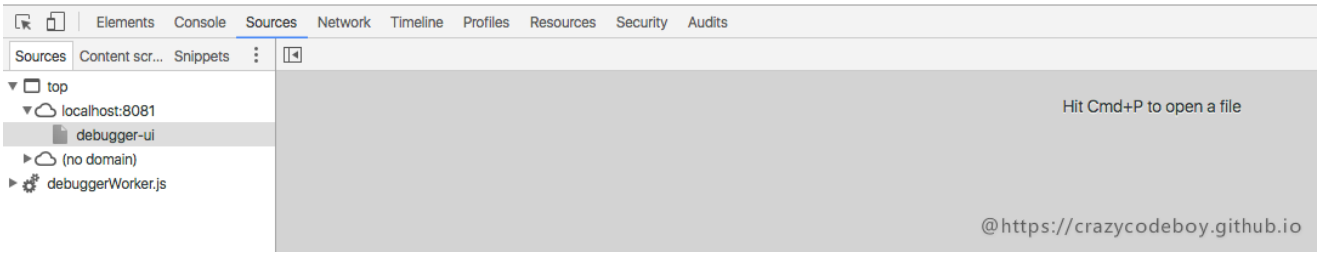


第二步：打开Chrome开发者工具

在该“http://localhost:8081/debugger-ui.”Tab页下打开开发者工具。打开Chrome菜单->选择更多工具->选择开发者工具。你也可以通过快捷键(Command⌘ + Option⌥ + I on Mac, Ctrl + Shift + I on Windows)打开开发者工具。



打开Chrome开发着工具之后你会看到如下界面：



真机调试

在iOS上

打开"RCTWebSocketExecutor.m "文件，将“localhost”改为你的电脑的ip，然后在Developer Menu下单击"Debug JS Remotely" 启动JS远程调试功能。

在Android上

方式一：
在Android5.0以上设备上，将手机通过usb连接到你的电脑，然后通过adb命令行工具运行如下命令来设置端口转发。

```
adb reverse tcp:8081 tcp:8081
```

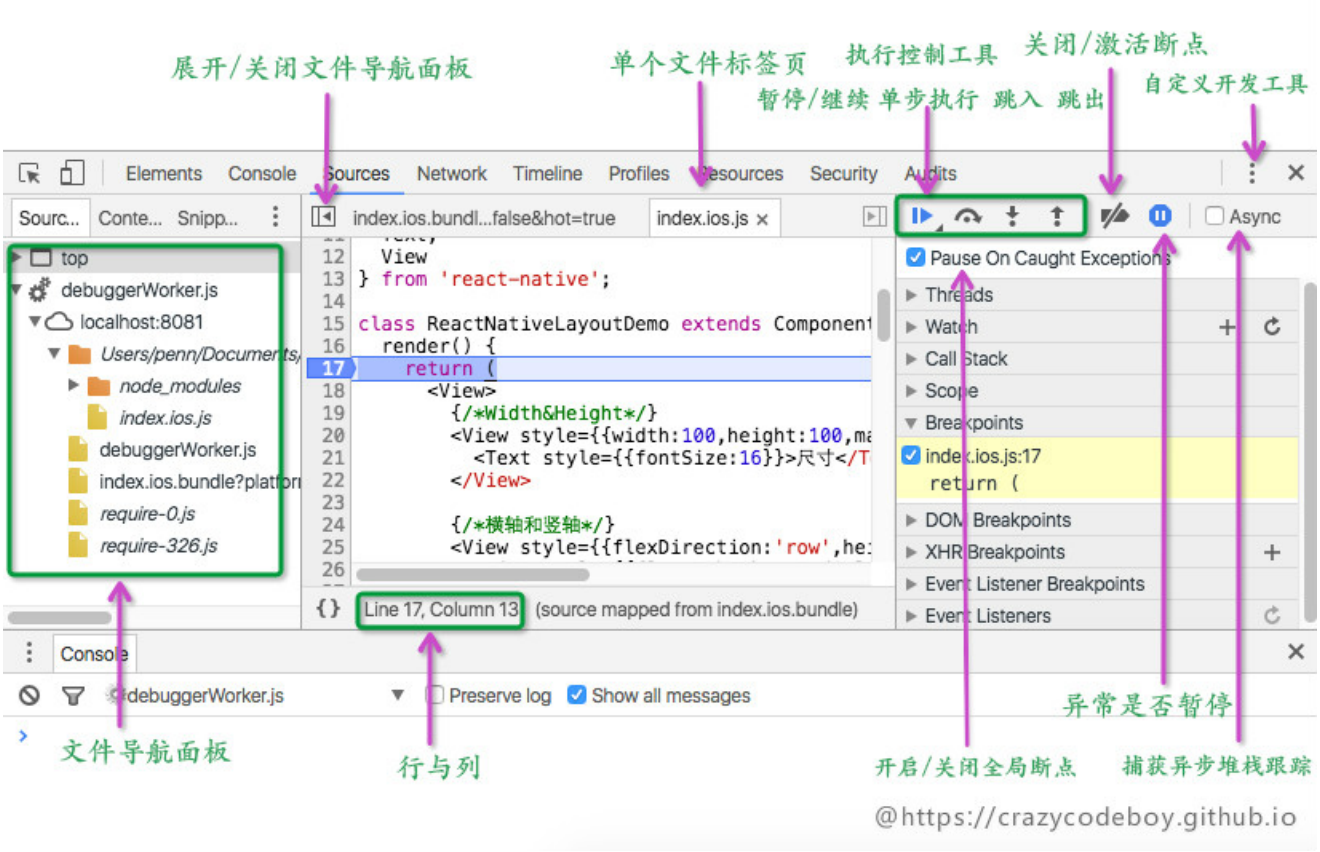
方式二：
你也可以通过在“Developer Menu”下的“Dev Settings”中设置你的电脑ip来进行调试。

> 心得：在使用真机调试时，你需要确保你的手机和电脑处在同一个网段内，即它们实在同一个路由器下。

小技巧：

巧用Sources面板

Sources 面板提供了调试 JavaScript 代码的功能。它提供了图形化的V8 调试器。



Sources 面板可以让你看到你所要检查的页面的所有脚本代码，并在面板选择栏下方提供了一组标准控件，提供了暂停，恢复，步进等功能。在窗口的最下方的按钮可以在遇到异常(exception)时强制暂停。源码显示在单独的标签页，通过点击 打开文件导航面板，导航栏中会显示所有已打开的脚本文件。

> 心得：Chrome开发着工具中的Sources面板几乎是我最常用的功能面板。通常只要是开发遇到了js报错或者其他代码问题，在审视一遍自己的代码而一无所获之后，我首先就会打开Sources进行js断点调试。

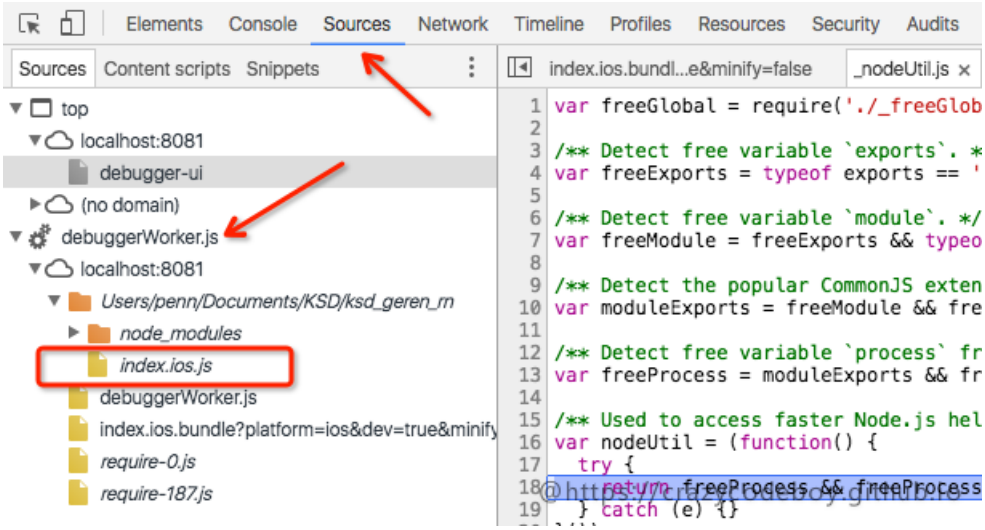
执行控工具

从上图可以看到“执行控工具”按钮在侧板顶部，让你可以按步执行代码，当你进行调试的时候这几个按钮非常有用：

- 继续(Continue): 继续执行代码直到遇到下一个断点。
- 单步执行(Step over): 步进代码以查看每一行代码对变量作出的操作，当代码调用另一个函数时不会进入这个函数，使你可以专注于当前的函数。
- 跳入(Step into): 与 Step over 类似，但是当代码调用函数时，调试器会进去这个函数并跳转到函数的第一行。
- 跳出(Step out): 当你进入一个函数后，你可以点击 Step out 执行函数余下的代码并跳出该函数。
- 断点切换(Toggle breakpoints): 控制断点的开启和关闭，同时保持断点完好。

查看js文件

如果你想在开发者工具上预览你的js文件，可以在打开Sources tab下的debuggerWorker.js选项卡，该选项卡下会显示当前调试项目的所有js文件。



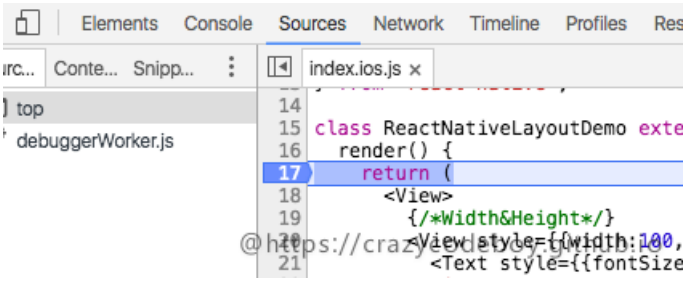
断点其实很简单

断点(Breakpoint) 是在脚本中设置好的暂停处。在DevTools中使用断点可以调试JavaScript代码，DOM更新和 network calls。

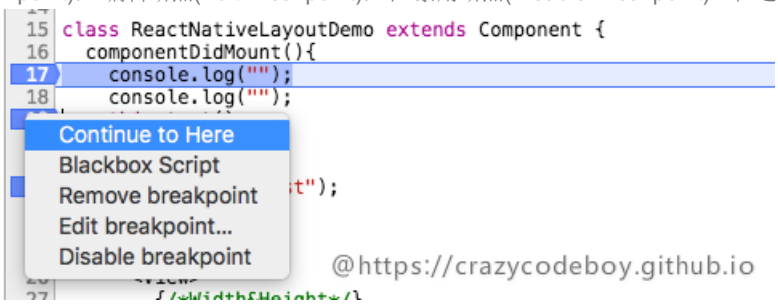
> 心得：你可以像使用Xcode/AndroidStudio调试Native应用一样，来使用Chrome开发者工具通过断点对程序进行调试。

添加和移除断点

在 Sources 面板的文件导航面板中打开一个JavaScript文件来调试，点击边栏(line gutter) 为当前行设置一个断点，已经设置的断点处会有一个蓝色的标签，单击蓝色标签，断点即被移除。



> 心得：右键点击蓝色标签会打开一个菜单，菜单包含以下选项：执行到此(Continue to Here)，黑盒脚本(Blackbox scripts)，移除断点(Remove Breakpoint)， 编辑断点(Edit Breakpoint)，和 禁用断点(Disable Breakpoint)。在这里你可以对断点进行更高级的定制化的操作。



高级操作

上文讲到右键点击蓝色标签会打开一个菜单，下面就介绍一下该菜单下的高级操作。

执行到此(Continue to Here)：

如果你想让程序立即跳到某一行时，这个功能会帮到你。如果在该行之前还有别的断点，程序会依次经过前面的断点。另外需要提出的是这个功能在任意一行代码的边栏(gutter line)前单击右键都会看到。

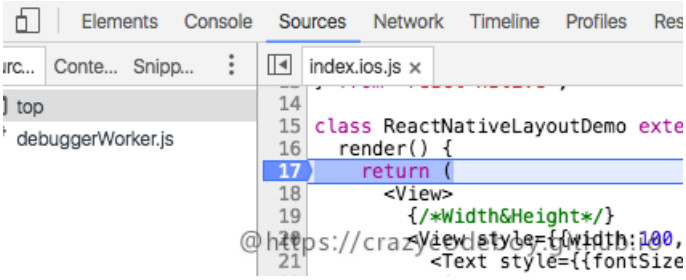
黑盒脚本(Blackbox scripts)：

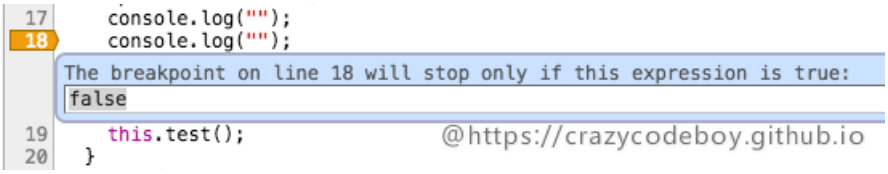
黑盒脚本会从你的调用堆栈中隐藏第三方代码。

编辑断点(Edit Breakpoint)：

通过该功能你可以创建一个条件断点，你也可以在边栏(gutter line) 右键并选择添加条件断点(Add Conditional Breakpoint) 。在输入

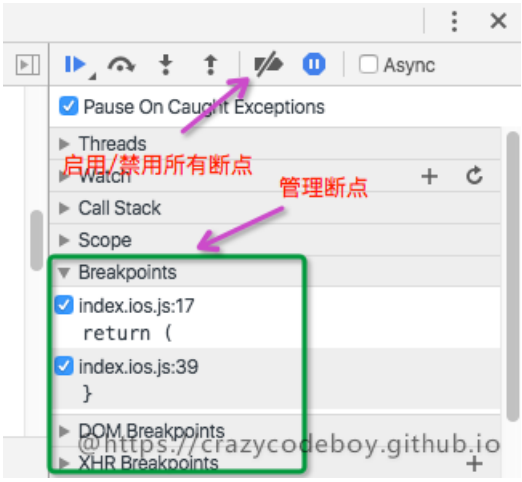
框中，输入一个可解析为真或假的表达式。仅当条件为真时，执行会在此暂停。



> 心得：如果你想让程序在某处从来都不要暂停，可以编辑一个条件永远为false的条件断点。另外，你也可以在该行代码的边栏(gutter line)前单击右键选择“Never pause here”即可，你会发现“Never pause here”其实就是在该行代码上设了一个永远为false的条件断点。
>
> 
>

管理你的断点

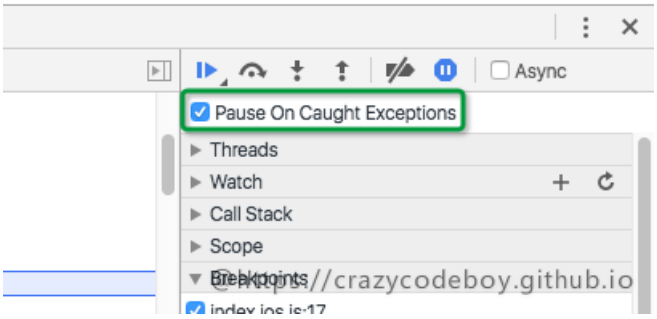
你可以通过Chrome开发者工具的右边面板来统一管理你的断点。



> 心得：你可以通过断点前的复选框来启用和禁用断点，也可以单击右键来进行更多的操作(如：移除断点，移除所有断点，启用禁用断点等)。

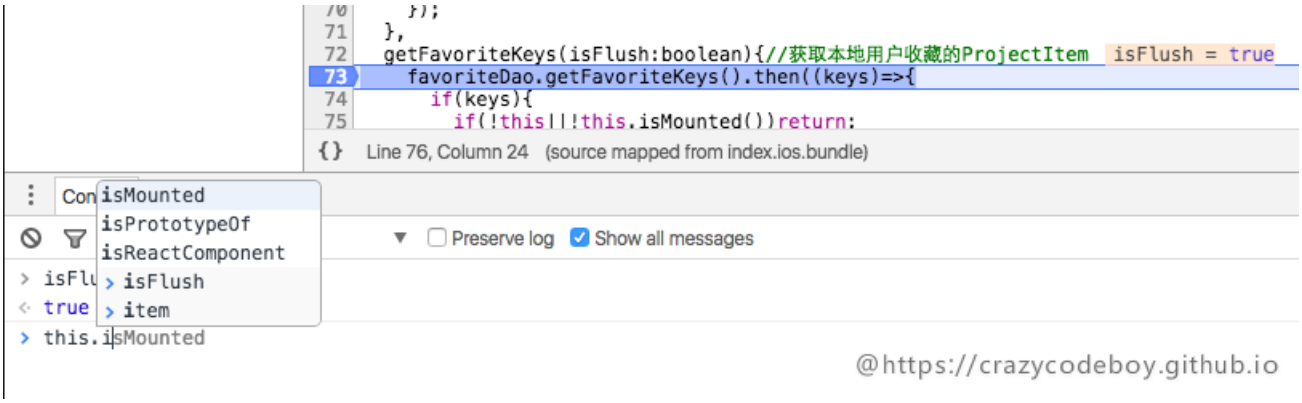
有一种断点叫全局断点

全局断点的作用是，当程序出现异常时，会在异常的地方暂停，这对快速定位异的常位置很方便。
做iOS开发的同学都知道在Xcode中可以设置全局断点，其实在Chrome 开发者工具中也同样有与之对应的功能，叫“Pause On Caught Exceptions”。如果勾选上此功能，则即使所发生运行时异常的代码在 try/catch 范围内，Chrome 开发者工具也能够在错误代码处停住。



不要忽略控制台

DevTools 控制台(Console) 可以让你在目前已暂停的状态下进行试验。按 Esc 键打开/关闭控制台。

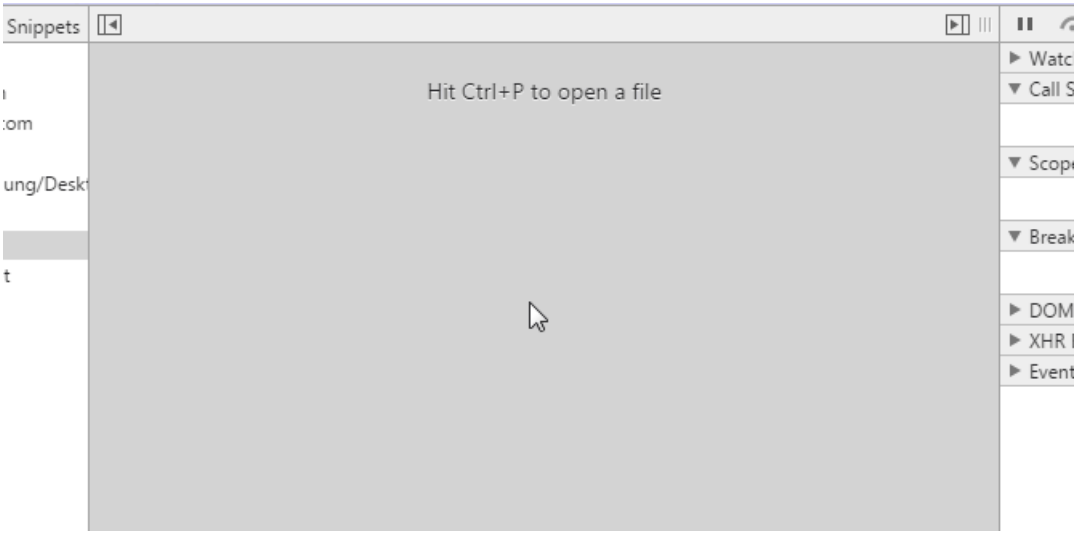


> 心得：你可以在控制台(Console)上打印变量，执行脚本等操作。在开发调试中非常有用。

Chrome Developer Tools 快捷键

> 快速切换文件

当DevTools被打开的时候，按Ctrl+P或O（cmd+p或o on mac）,就能快速搜寻和打开你项目的文件。



> 在当前文件中查找内容

在当前文件中查找内容可通过Ctrl(cmd) + F快捷键。

> 在所有文件中查找内容

如果你希望在在所有文件中查找内容怎么办呢？在页面已经加载的文件中搜寻一个特定的字符串，快捷键是Ctrl + Shift + F (Cmd + Opt + F)，这种搜寻方式还支持正则表达式。

> 快速跳转到指定行

在Sources标签中打开一个文件之后，按Ctrl + G，然后输入行号，DevTools就会允许你跳转到文件中的任意一行。

> 心得：如果你在“快速切换文件”输入框中输入“:”然后加行号也能跳转到指定行。

参考

chrome-devtools
CN-Chrome-DevTools
Debugging

About

本文出自《React Native学习笔记》系列文章。
了解更多，可以关注我的GitHub
@https://crazycodeboy.github.io/

推荐阅读

- React Native 学习笔记

- Reac Native布局详细指南
- React Native发布APP之签名打包APK
- React Native应用部署、热更新-CodePush最新集成总结