

查询缓存：当一个SQL进来时，如果开启查询缓存功能，MySQL会优先去查询缓存中检查是否有数据匹配，如果匹配上，就不会再去解析对应的SQL啦，但如果语句中有**用户自定义函数、存储函数、用户变量、临时表、mysql库中的系统表**时，都不会走缓存； 对于查询缓存来说，在**MySQL8.0已经去除**，官方回应的是在一定场景上，查询缓存会导致性能上的瓶颈。

解析器：对于一个SQL语句，MySql根据语法规则需要对其进行解析，并生成一个内部能识别的**解析树**；

优化器：负责对解析器得到的解析树进行优化，MySQL会根据内部算法找到一个MySQL认为最优的执行计划，后续就按照这个执行计划执行。所以后续我们分析的就是MySQL针对SQL语句选择出来的最优执行计划，结合业务，根据规则对SQL进行优化，从而让SQL语句在MySQL内部达到真正的最优。

执行器：得到执行计划之后，就会找到对应的存储引擎，根据执行计划给出的指令依次执行。

存储引擎：数据的存储和提取最后是靠存储引擎；MySQL内部实现可插拔式的存储引擎机制，不同的存储引擎执行不同的逻辑；

物理文件：数据存储的最终位置，即磁盘上；协同存储引擎对数据进行读写操作。

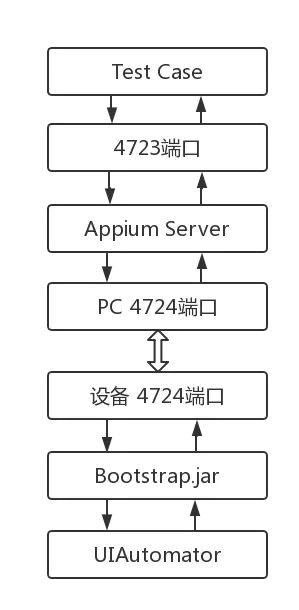
当一个SQL发送到MySQL执行时，需要经过内部优化器进行优化，而使用explain关键字可以模拟优化器执行SQL查询语句，从而知道MySQL是如何处理SQL的，即SQL的执行计划；根据explain提供的执行计划信息分析SQL语句，然后进行相关优化操作。

iOS 9.3 and above: Apple's XCUITest

iOS 9.3 and lower: Apple's UIAutomation

Android 4.3+: Google's UiAutomator/UiAutomator2

Windows: Microsoft's WinAppDriver



We meet requirement #2 by wrapping the vendor-provided frameworks in one API, the WebDriver API. WebDriver (aka "Selenium WebDriver") specifies a client-server protocol (known as the JSON Wire Protocol). Given this client-server architecture, a client written in any language can be used to send the appropriate HTTP requests to the server.

Appium is a server written in Node.js.

appiumserver 启动了一个监听端口例如4724, 同时向手机端adb push 一个bootstrap.jar的脚本，手机端通过该脚本同时监听端口4724

3.webdriver是基于http协议的，第一连接会建立一个session会话，并通过post发送一个json告知服务端相关测试信息。

Appium服务器。它是一个基于node.js的HTTP服务器。主要功能是接受从Appium客户端发起的链接，监听客户端发送来 命令，将命令发送到bootstrap.jar(IOS为bootstrap.js)执行，并将命令的结果通过HTTP应答反馈给Appium客户端。

AppiumBootstrap是一个jar包； Appium会将这个jar包打到device中； 然后使用adb shell uiautomator runtest AppiumBootstrap.jar -c io.appium.android.bootstrap.Bootstrap会使用一个SocketServer用于监听发送过来的命令； 得到命令之后，扔给Uiautomator执行；