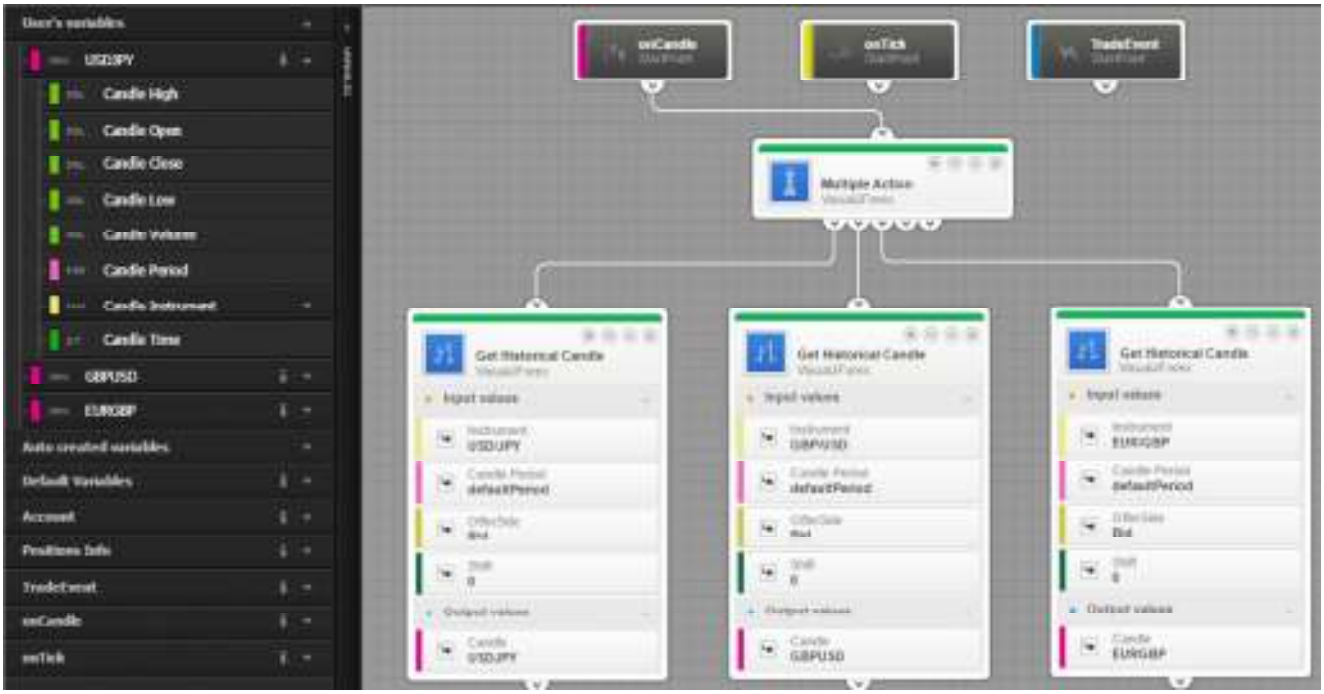
7.2. 如何使用多个交易产品

建立一个有关多个交易产品的策略是可行的，但是对其进行测试将比较困难因为它需要导出策略到 JForex 平台以此来同时运行在多个交易产品上。从技术上将，建立此类策略是基于“Get historical candle（获取历史蜡烛）”的帮助来给每个交易产品使用之前的蜡烛信息。

根据交易产品的需要使用尽可能多的“Get historical candle（获取历史蜡烛）”并删除默认变量然后从下拉菜单里选择所需的交易产品。相对于使用自动生成的输出值，在下面这个例子中使用专门的数据；这将简化每个交易产品的变量认证。



在上述那些模块之前添加一个期限过滤器是可行的。

在左侧截图里，当策略启动后，变量被放入工作区来调试它们的参数。如图所示，OHLC 价格是基于每个交易产品显示的并且可以用于之后进一步的条件。

之前获取蜡烛数据的方法被广泛用于当策略理念应用于交叉交易产品逻辑的时候；换言之，比如当条件 A 被 EURUSD 验证后交易 USDJPY。因此，要交易一个既定的交易产，必须在另一个交易产品上验证交易条件。对于那些在几个交易产品上实施同样逻辑的策略，它会更简单的创建一个额外的策略文件并且在视像 JForex 层面更改交易产品；亦或者在 JForex 里运行或测试策略并且选择所需要的交易产品。

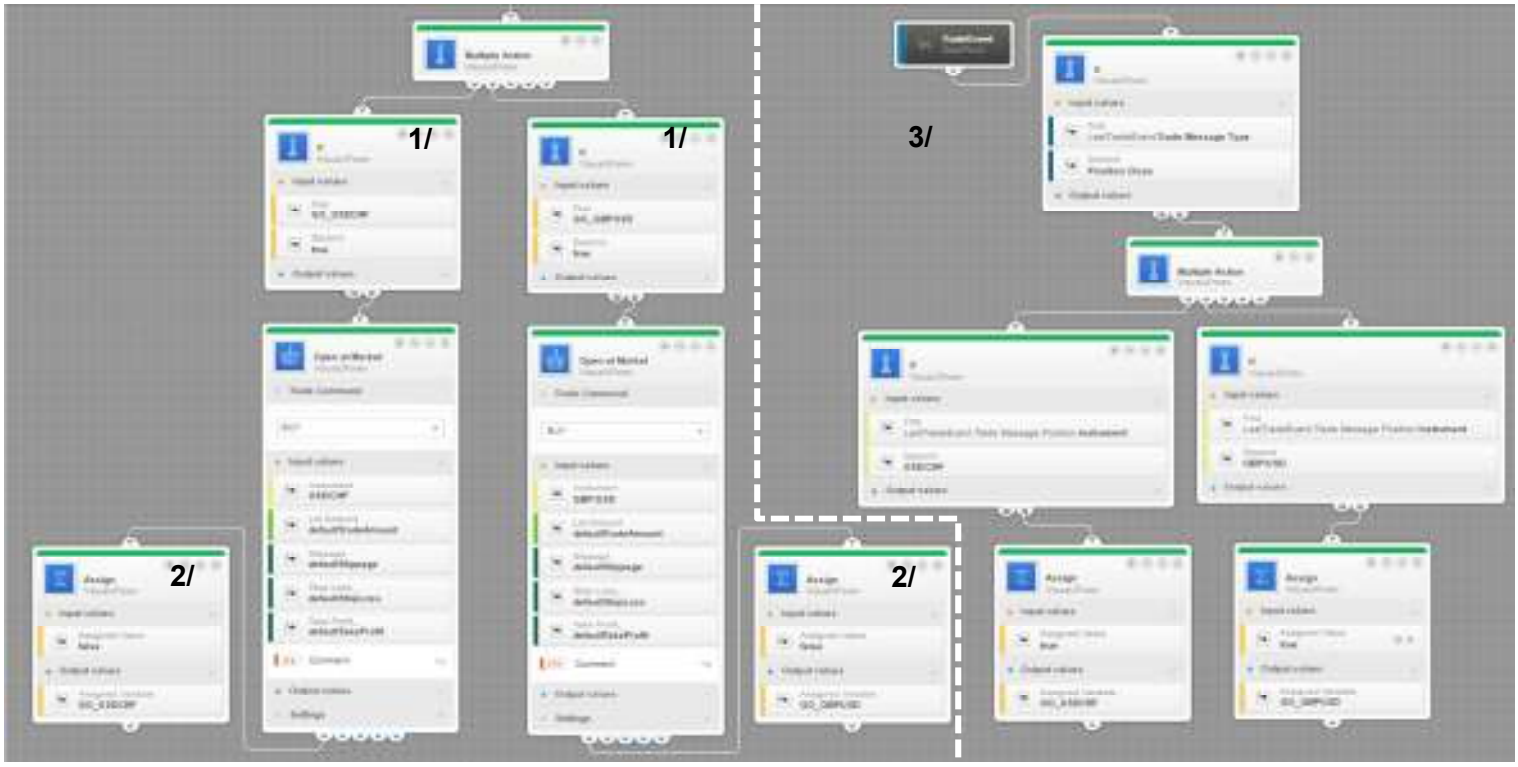
测试对个交易产品的策略无法在视像 JForex 里完成因为视像 JForex 实施一个单一的默认交易产品。此类策略只能在 JForex 里测试。

7.8. 如何处理逻辑型触发器

逻辑型触发器是由用户创建的，当条件 A 发生的时候获取一个既定值以及当条件 B 发生的时候转换到另一个数值。让我们来假设以下情形：
一个策略交易 2 个交易产品并且用户需要管理基于每个交易产品的已开仓位数量：要达成这个目的的 一种方法为创建 2 个可以触发/阻止建仓指令的变量。

***1/** 此类变量的起始值需要使该策略能够建立一个仓位；*
***2/** 一旦仓位被建立，变量会马上更改其数值来阻止下一阶段的仓位建立；*
***3/** 一旦仓位被关闭，同样的变量需要重置回最初的数值。*

该流程图基于 **2 个条件**实行了数值的变更，因此我们可以选择一个 Boolean 变量来获取两种值，True 或者 False。



在左侧: 变量 “GO_USDCHF” 和 “GO_GBPUSD” 在开始的时候拥有“True” 值; 在最一开始的流程里，第一个 IF 条件将被通过然后每种交易产品在“GO_USDCHF” 和 “GO_GBPUSD”被切换为“False”的时候会有一个仓位被建立。当策略执行下一个流程的时候，IF 条件(**1/**) 将阻止进一步仓位的建立。其结果为: 每个货币对的已开仓位的最大数量不会超过 1。

在右侧: 在该例子中， ‘Trade event’ （交易时间）做为起始点来获取 “Position Closed”（已平仓位）的信息， 策略将检查货币对并分配“True”值给触发器，由此，新的仓位可以在下一流程被建立。

当需求是基于 Start / Hold（开始/保持）逻辑的时候，实行逻辑型触发器是可以行的。用户需要精确的确定触发器什么时候会转换以及什么时候会重置为初始值。拥有 3 或 4 个序列的更为复杂的流程可以使用拥有“Double” 或者 “Integer”的变量。