IM472884

如何用 Inventor 实现高效设计自动化： Inventor iLogic + Inventor API

范娟娟
Autodesk

学习目标

- 学习 iLogic 组件和基本功能
- 了解 iLogic 使用技巧和窍门
- 了解 iLogic 进阶功能
- 了解 Inventor 设计自动化中常用的 API 接口
- 了解 iLogic 和 Inventor API 学习资源

说明

我们为什么需要设计自动化？原因不言而喻，我们想要加速设计流程，想要减少人力劳动的辛苦和成本。 越来越多的个人和企业开始认识到自动化的重要性，已经启动抑或已经深度使用并完成设计自动化，并从中体验到提高效率的乐趣和便利。

您听说过 Inventor iLogic 和 Inventor API 帮助设计人员减负并大幅提高设计效率的事例吗？或者您听说过 iLogic 和 API，但是从来没有深度学习和使用过，以及不知道什么时候使用呢？

在本课程中，您将了解到如何使用 Inventor iLogic 和 InventorAPI 来自动化一些工作流程，从而更高效地使用 Inventor。内容会包括 iLogic 基础和进阶介绍，使用小贴士和 Inventor API 的常用接口示例和查询。

讲师

范娟娟是 Autodesk Inventor 团队高级测试工程师。 2004 年四川大学毕业后加入 Autodesk，一直从事 Inventor API, ETO，Inventor iLogic 以及设计自动化相关产品例如 Inventor IO 的测试及设计自动化工作。

Jane.fan@autodesk.com

Inventor Customization

iLogic 基础

iLogic 是什么？

iLogic 是 Inventor 的内置附加项，可以帮助用户比较方便地用少量编码（规则）定义自己的工作流逻辑，从而实现设计驱动，设计自动化。

iLogic 历程

一路走来，iLogic 随着 Inventor 的版本升级也在不断的提升自身能力，对照下图来看看 iLogic 在历届 Inventor 中的历程：

- Inventor 2010 : 引入 iLogic
- Inventor 2011 : 界面增强和参数驱动
- Inventor 2012 : iLogic 表单
- Inventor 2018 : 代码安全性检查
- Inventor 2018.1 : 全局事件触发, 编码智能联想
- Inventor 2019 : iLogic 装配自动化
  - 日志输出
  - 代码调试
- Inventor 2021: iLogic 工程图自动化

iLogic 入口

我们可以通过 Ribbon 上的控件直达 iLogic 的各个操作。如果是添加/删除规则或者表单，也可以通过模型浏览器旁边的小加号来打开 iLogic 的浏览器进行操作。下图展示了 Ribbon 上的 iLogic 面板，以及 iLogic 的浏览器。

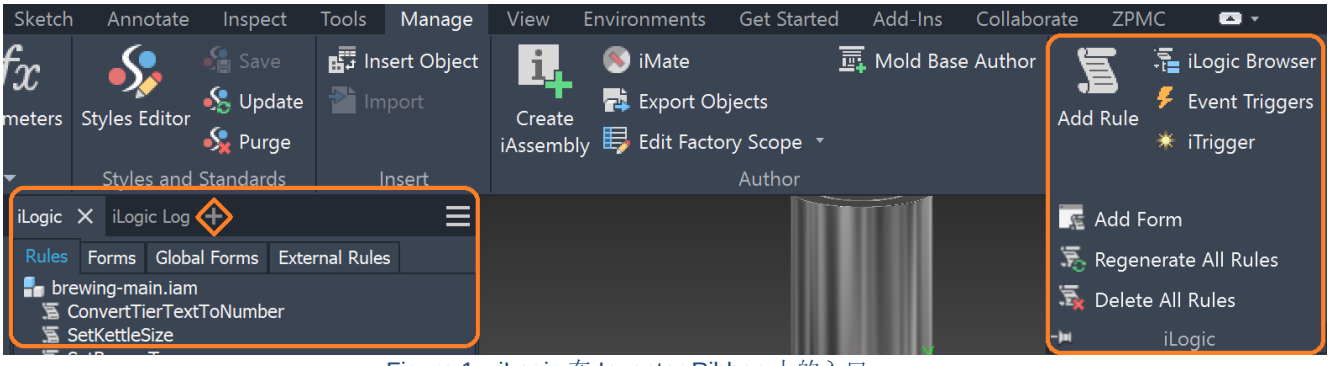

Figure 1：iLogic 在 Inventor Ribbon 上的入口

iLogic 组件

iLogic 提供了一下的各组件以及功能概括。我们接下来会简要介绍一下常用模块。

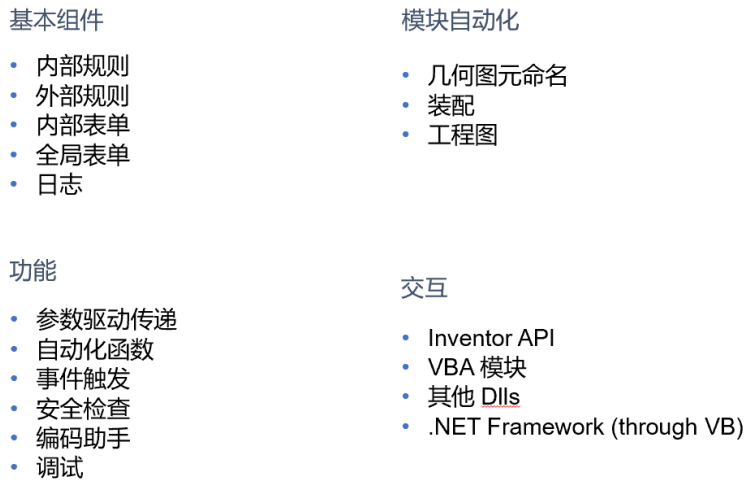

Figure 2：iLogic 提供的组件

iLogic 规则

iLogic 的核心就是逻辑，逻辑需要用规则来实现。使用 iLogic 需要写一些规则。写简单逻辑的规则很容易，比较复杂逻辑的规则会稍微困难一点。但是即使是简单的规则，也可以帮助我们自动化很多流程。

iLogic 规则其实是 VB.Net 代码。为了方便使用，在 iLogic 可以不用显示声明“Sub Main”，直接写自己的逻辑即可。iLogic 提供了代码编译器，提供代码片段方便使用和学习；iLogic 可以直接引用及修改 Inventor 参数；iLogic 规则还封装了部分 Inventor API and 一些 .Net 接口，比如 ThisDoc, Feature.IsActive， GoExcel 等等。在 iLogic 的编辑器里，我们可以使用 Inventor API，iLogic 接口， iLogic API 和.Net 的接口和函数。

iLogic 规则分为两种类型，内部规则和外部规则。内部规则存在文件内部（IPT，IAM 或者 IDW），只适用于当前文档。外部规则是独立的外部文件（iLogicVb，txt，vb），可以被重用以及驱动多个文件。

规则触发

写规则的目的是让它可以根据条件判断来运行，运行规则也可以被叫做触发规则。除了显示的手动运行规则之外，规则在多种情况下可以被自动触发。

对于内部规则来说，最为常用的触发方式是在规则中引用参数，参数修改会自动触发规则。当然还有其他的触发方式：

- 规则引用了参数，参数发生改变时
- 另一个规则在程序内部调用规则
- 事件触发器触发

对于外部规则来说， 通常使用以下的触发方式：

- 显示调用和运行
- 事件触发器触发

下面是一个演示规则通过参数引用触发的示例（示例 1）：

步骤：

1. 在任何有参数 d0 的文件中，添加一个 iLogic 规则。

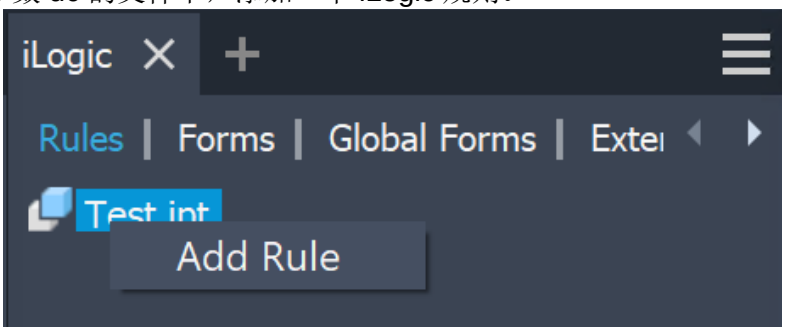

Figure 3：添加规则

2. 在规则中写入如下代码：

```
Dim temp = d0
If d0 <1 Then
MsgBox("d0 is too small, set it to 1")
d0 = 1
Else If d0 >5Then
MsgBox("d0 is too large, set it to 5")
d0 = 5
End If
InventorVb.DocumentUpdate(True)
```

Figure 4:iLogic 规则

3. 打开参数表，修改 d0 为 0.5，或者 7，查看结果.

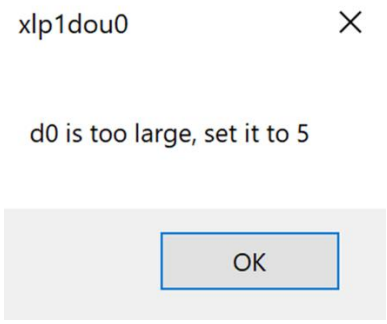

Figure 5：运行结果 1

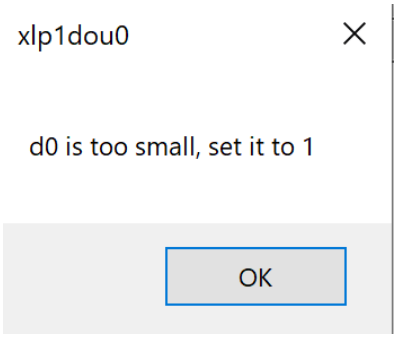

Figure 6：运行结果 2

iLogic 表单

iLogic 表单是用户自定义的弹出窗口，提供了简单便捷的方式来展示以及修改配置参数。表单支持配置参数，属性，规则按钮，可以拖拽构造表单的结构，甚至可以展示图片。

拿上面示例 1 来举例，我们可以把 d0 放在表单上，这样可以直接通过表单来修改 d0，不需要打开参数对话框。对于参数比较多的模型，这样的操作可以提供很大便利去找到需要配置的参数。参考图 7 和图 8.

除了简单设计，我们还可以设计表单的结构，通过标签页，不同的容器来容纳不同的控件，来设计相对复杂功能分块的表单，例如图 10 所示。

当然，表单也有内部表单和外部表单，和内外部规则的机制一样，内部表单存在于当前的文档中，只适用于当前文档。外部表单则是可以被公共使用。

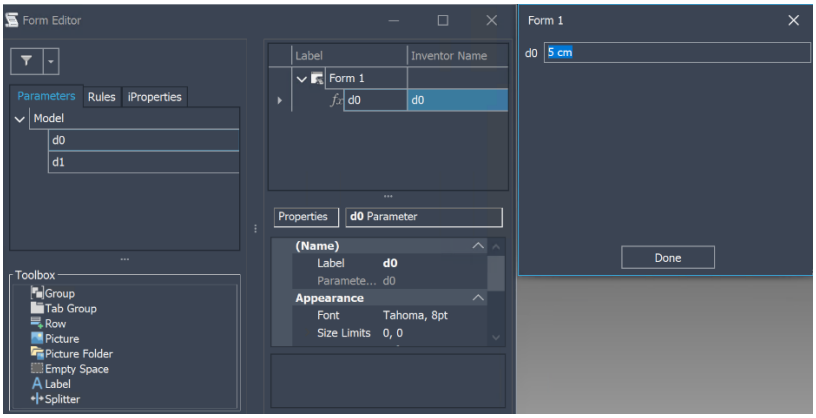

Figure 7:表单编辑器

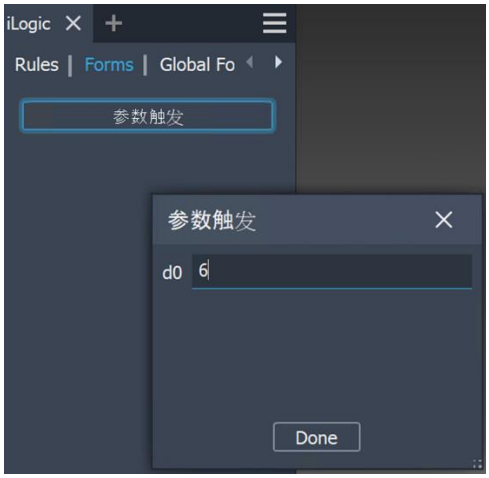

Figure 8：表单修改参数

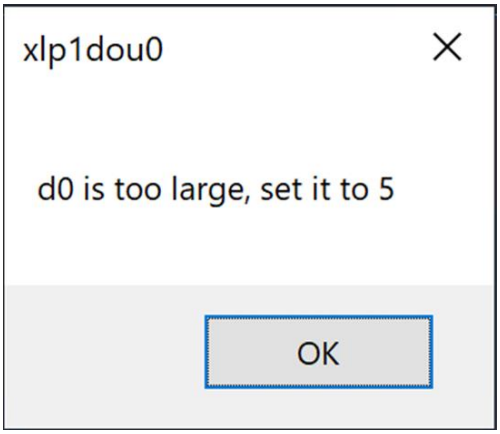

Figure 9：运行结果

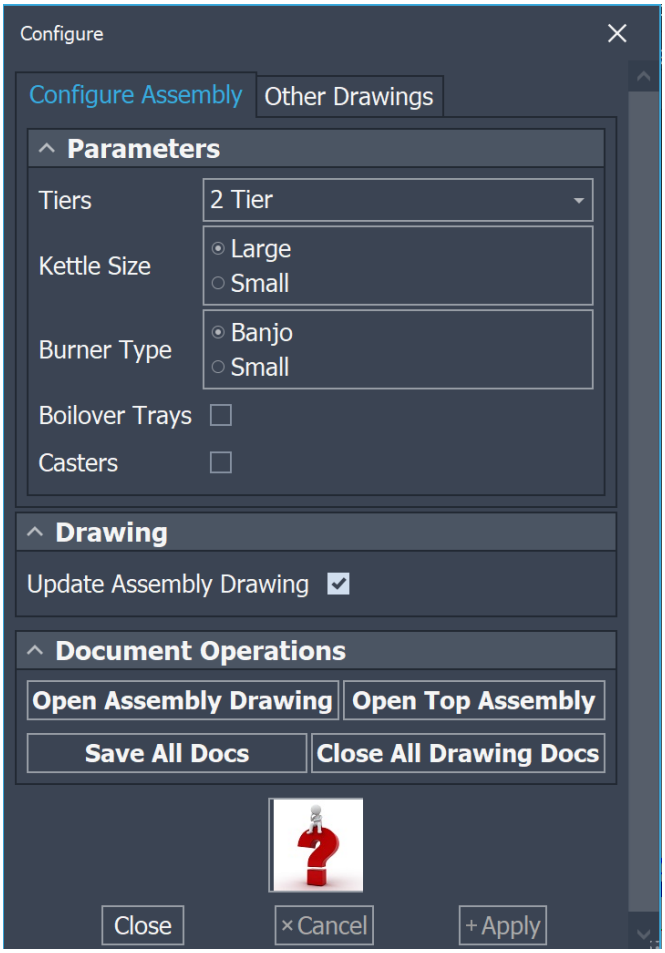

Figure 10：示例表单

iLogic 事件触发

iLogic 的事件是一系列的预定义的事件，可以根据事件触发来运行事件下的规则。我们知道规则有内部规则和外部规则， 在事件触发器中，同样的道理，内部规则只可放在当前文档的事件下被控制触发；而外部规则则可放在任何文档下，或者某一类型的文档下。

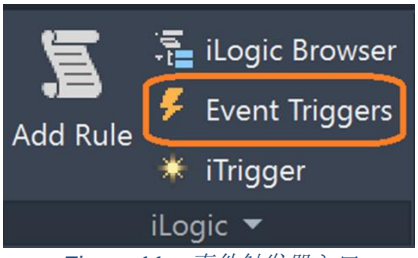

Figure 11：事件触发器入口

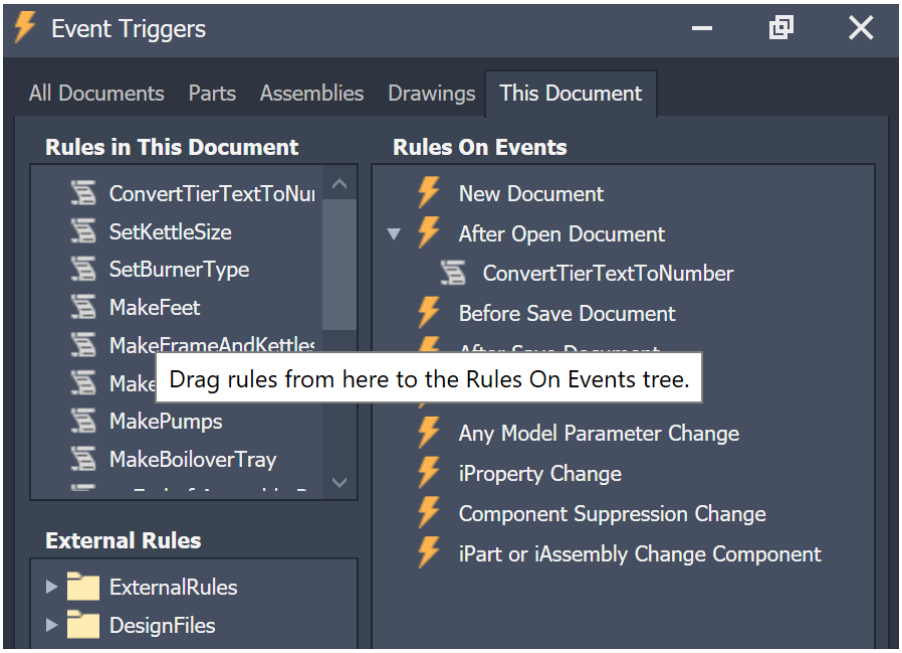

Figure 12：事件触发器

下面是一个事件触发的示例（示例 2），演示了如何通过把规则放入事件触发器中，每次文档保存都自动触发规则运行。

步骤：

1. 添加一个外部规则（ChangeProperty.iLogicVb），可以在任何地方：

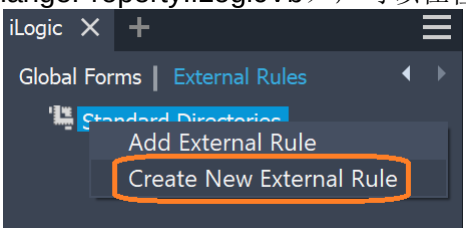

Figure 13：添加外部规则

2. 规则内写入以下代码：

```
iProperties.Value("Project", "Part Number") = "Autodesk: " + ThisDoc.FileName
iProperties.Value("Summary", "Company") = "Autodesk"
```

Figure 14：规则代码

3. 按照图 15 和 16 来设置外部规则文件夹路径。设回路径的目的是保存外部规则触发的设置，会自动生成一个 XML 文件在此路径下。

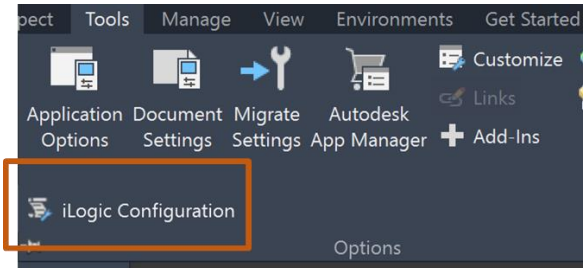

Figure 15：iLogic 高级配置入口

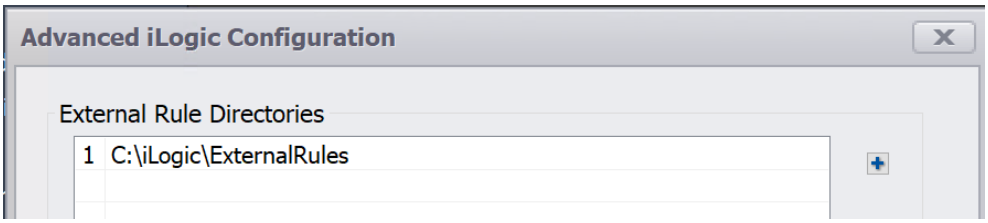

Figure 16：添加外部规则文件夹

4. 打开事件触发器，切换到第二个标签（Part/零件）下，找到外部规则 ChangeProperty，拖拽到“Before Save Document”事件下。

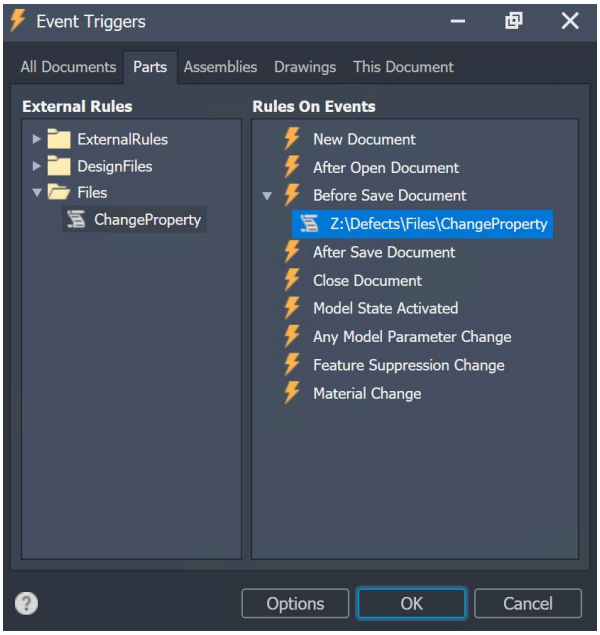

Figure 17：设置规则的事件触发

5. 打开已有的零件或者新建一个零件，保存文件，每次查看一下零件的属性是否修改。试试看？
6. 对于新建的还没有命名零件，想一想为什么结果不一样？再把规则拖拽到“After Save Document”试试？

iLogic 装配和工程图

iLogic 装配是 Inventor 2019 的新功能，iLogic 工程图是用了类似的机制在 Inventor 2021 中提供的功能。目的都是在 iLogic 里提供更多的接口，使 iLogic 不仅是一个修改已有的特征属性组件工程图的工具，还能实现配置化地动态添加子零部件，约束以及工程图标注。

iLogic 装配的机制：

根据条件判断自动删除无用的子零部件和约束.

基于参数、属性等不同的设置创建不同的装配。如果驱动参数或者属性发生变化，装配内容将基于规则进行更新或者重新生成。

自动命名模型图元（点、线、面），或使用已有名称

可使用 BegainManage 和 EndManage 的语句来进行控制，条件满足，运行代码添加组件和约束；条件不满足，则删除管理组内的组件和约束。如图 20 所示。

iLogic 装配可以自动获取当前的设计，自动生成代码。在部件中新建 iLogic 规则，在模型标签下，尝试右键点击任意组件节点，或者约束节点，选择不同的捕获设计来查看生成的代码。如图 18 和 19 所示。

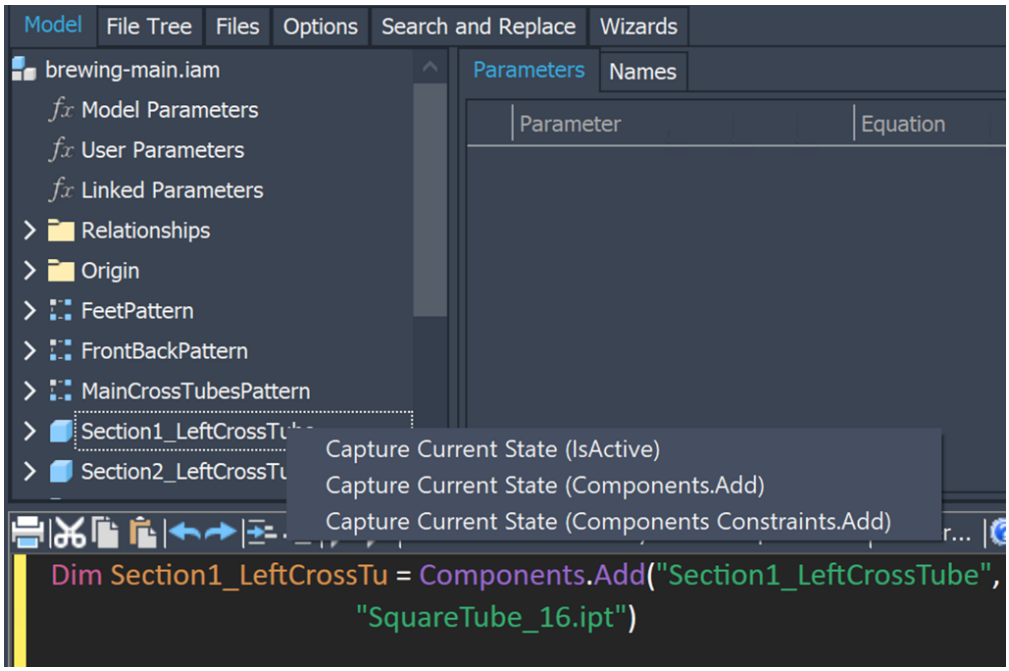

Figure 18：组件的捕获

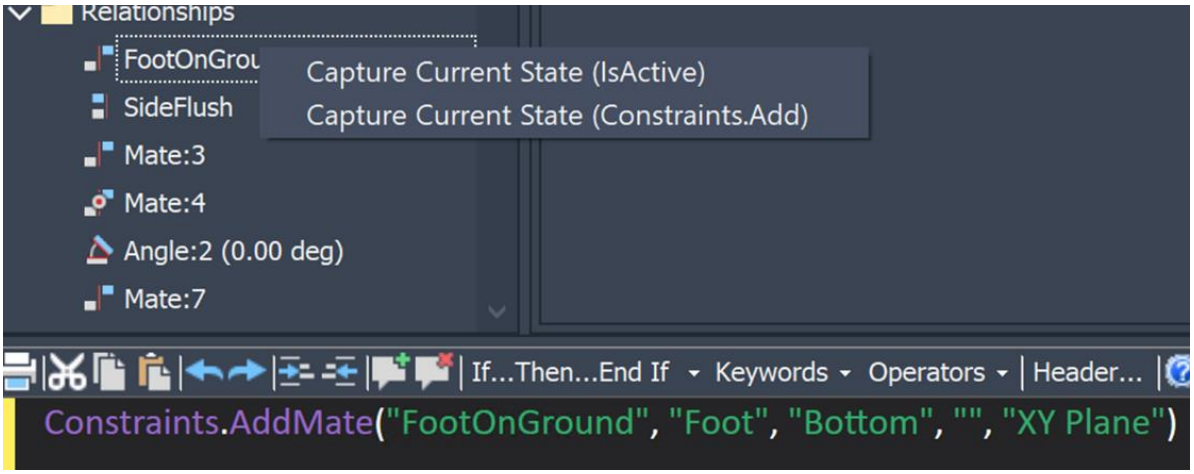

Figure 19：约束的捕获

获取代码后，尝试添加 BeginManage 和 EndManage，使用用户 bool 参数来控制组件和约束的生成与删除。

```
Dim addCompB = True
ThisAssembly.BeginManage("Group 1")
Dim componentA = Components.Add("occName", "a.ipt", position := Nothing, grounded := False, visible := True, appearance := Nothing)
If (addCompB)
   Dim componentB = Components.Add("occName", "b.ipt", position := Nothing, grounded := False, visible := True, appearance := Nothing)
End If
ThisAssembly.EndManage("Group 1")
```

Figure 20：管理组的示例

使用了 BeginManage 和 EndManage 控制之后，所有的组件约束会根据条件动态生成；条件不满足，组件和抑制会从当前的总装里面删除，没有组件和约束的抑制情况。

iLogic 工程图的机制：

根据条件判断添加或删除工程图标注.

如果驱动参数或者属性发生变化，工程图标注将基于规则进行更新或者重新生成。

自动命名模型图元（点、线、面），或使用已有名称

自动命名模型图元（点、线、面），或使用已有名称，

跟 iLogic 部件类似，iLogic 在工程图也有管理组的概念。BeginManage EndManage 管理组来进行控制约束的添加和删除。

iLogic 在工程图视图环境下在右键上下文菜单中提供了 iLogic 的入口，可以获取或者命名工程图中几何线，自动生成代码到剪贴板。然后新建或者打开 iLogic 规则，可以直接粘贴剪贴板中的代码到 iLogic 编辑器。

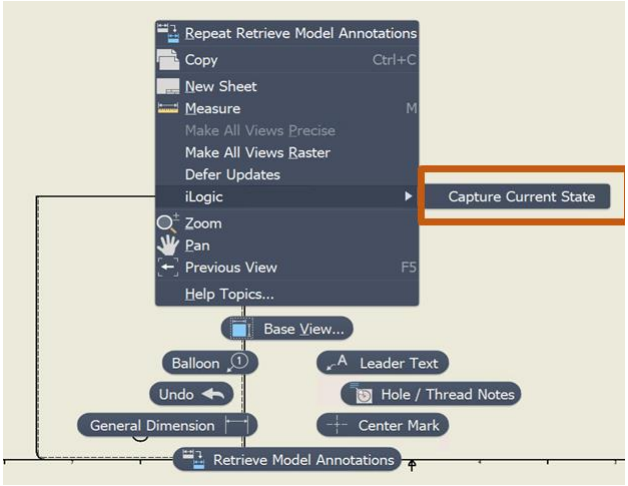

Figure 21：工程图iLogic 捕获入口

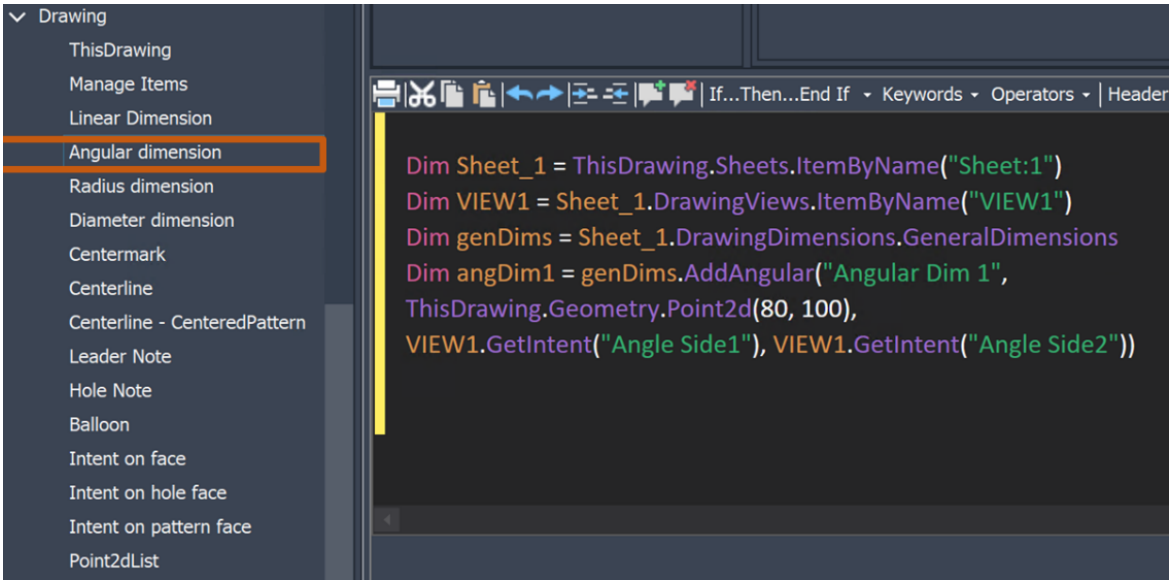

Figure 22：约束添加代码示例

```
ThisAssembly.BeginManage
If AddConstraint Then
   Constraints.AddByiMates("Mate1", "Part1:1", "iMate:1", "Part2:1", "iMate:1")
End If
ThisAssembly.EndManage
```

Figure 23：管理组示例

iLogic 进阶

iLogic API

在 iLogic 里面，我们可以直接通过以下方式获得 iLogic 的 API：

```
Dim iLogicAuto = iLogicVb.Automation
```

iLogic 的 API 提供了更多的方式来使用 iLogic。可以更随心地添加删除运行或者获取规则，可以修改 iLogic 的各种设置，比如设置 ExcelEngine，外部规则文件夹路径，或者是 LogLevel，还可以从工程图中获取模型参数。参考下图：

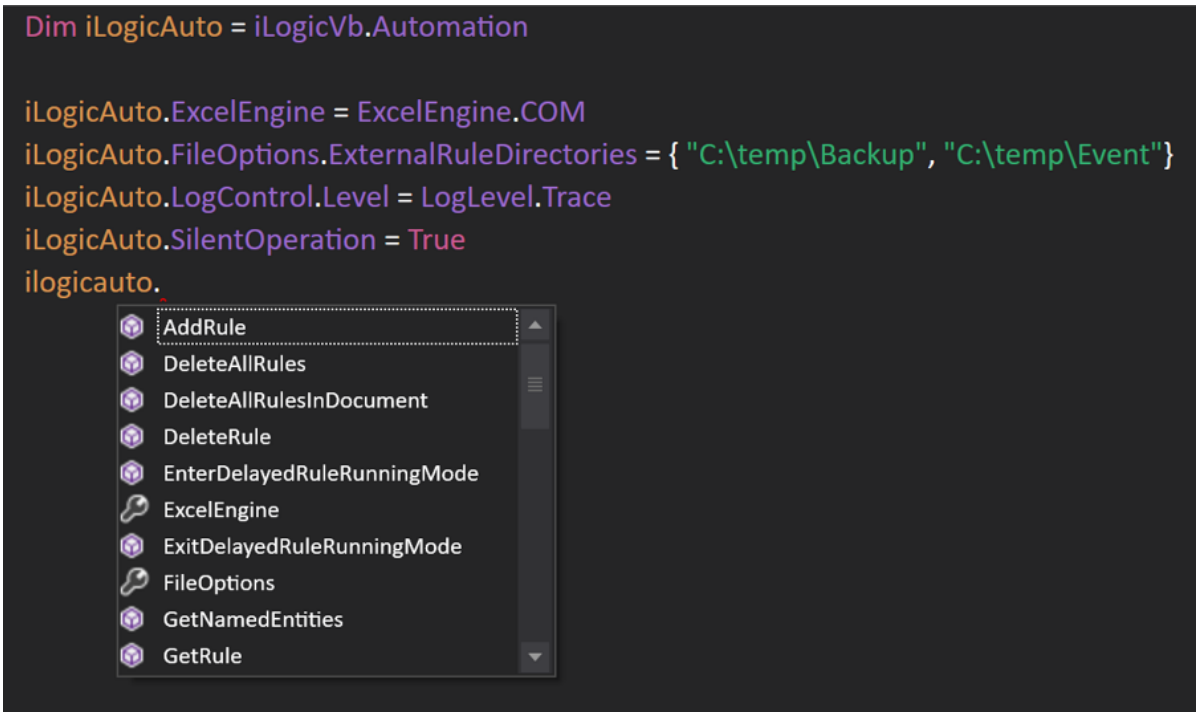

Figure 24：iLogicAPI 接口

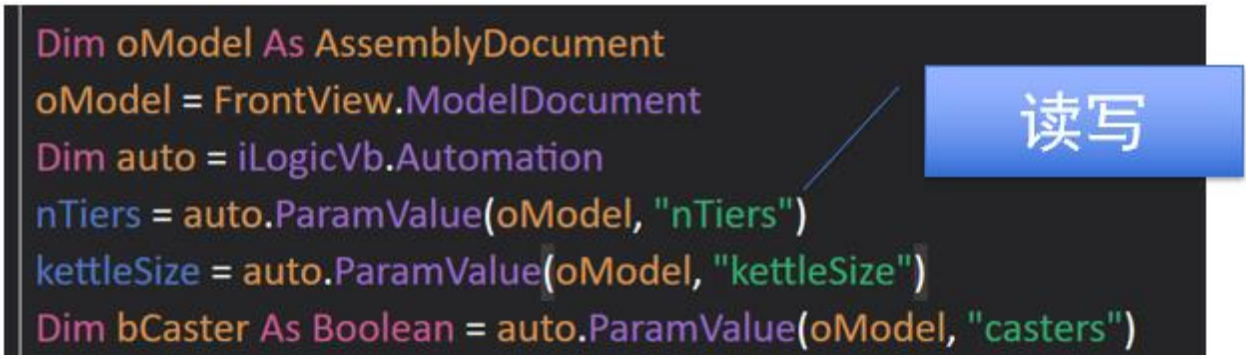

Figure 25：iLogicAPI 读工程图模型参数

在 iLogic 外面，甚至 Inventor 外面，我们同样可以调用 iLogic API。下面是一则在 Inventor VBA 环境调用 iLogic API 的示例：

```
Sub GetiLogic()
    Dim auto As Object
    Set auto = GetiLogicAutomation(ThisApplication)

    'ilogicvb.Automation.SilentOperation
    auto.SilentOperation = True
End Sub

Function GetiLogicAutomation(oApplication As InventorServer) As Object
    Set addIns = oApplication.ApplicationAddIns

    Dim addin As ApplicationAddIn
    Set addin = oApplication.ApplicationAddIns.ItemById("{3bdd8d79-2179-4b11-8a5a-257b1c0263ac}")

    addin.Activate
    Set GetiLogicAutomation = addin.Automation
End Function
```

Figure 26：代码示例

iLogic 代码调试

相较于其他的代码编辑器，比如 VBA，VS，传统的 iLogic 编辑器的不足之处在于调试。一般情况下，我们除了使用 MessageBox 来输出查看代码执行是否返回期望结果，并没有很好的实时查询对象以及变量返回值的调试工具。

所幸，iLogic 在 2021 提供了 Logger 接口以及 Log 输出面板，使查看执行结果更加便利:

```
Logger.Trace("This is a test")
ilogicvb.RunRule("Rule0")
```

Figure 27：演示代码

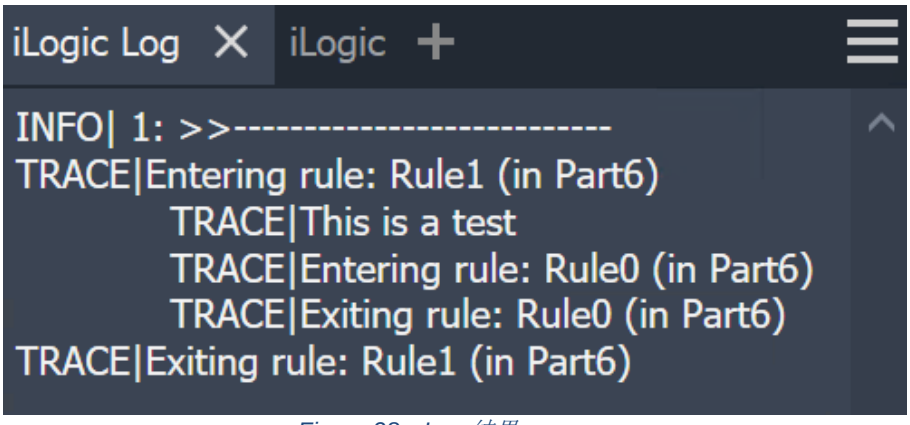

Figure 28：Log 结果

同时，也提供了结合 Visul Studio 进行实时调试的功能。调试功能依赖于 Visual Studio，所以需要有以下前提和步骤参考：

1. 安装 Microsoft Visual Studio 2017 或者 2019 (Community, Professional 或者 Enterprise edition），电脑上已有可忽略。
2. 用 inventor 打开含有 iLogic 规则的文件
3. 启动 Visual Studio
4. 在 VS 里，Debug->Attach to the Inventor process。
   此时， Visual Studio 已经依附于 Inventor 的进程。

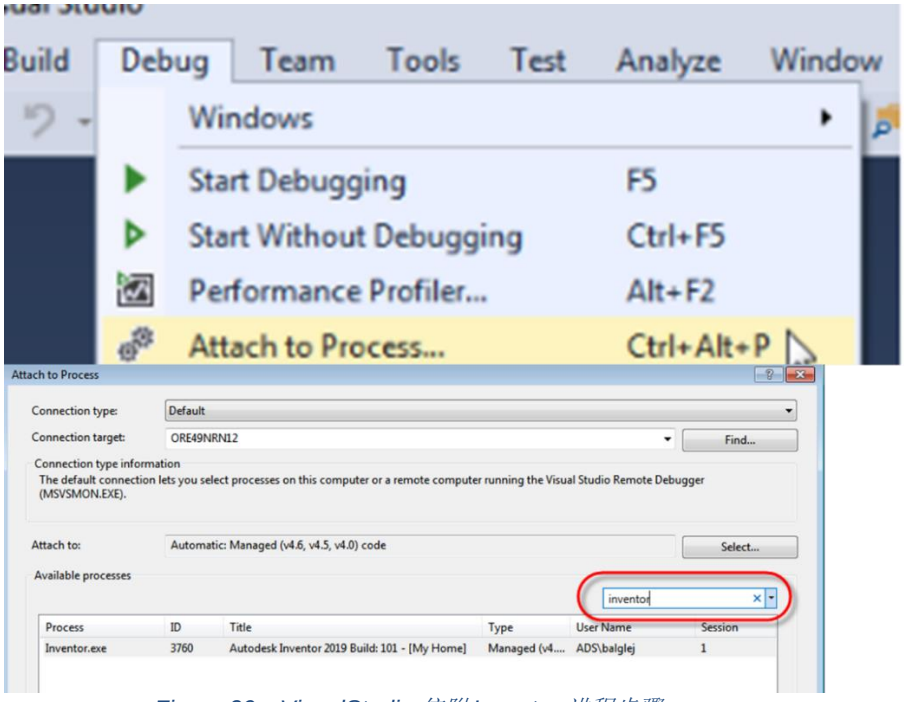

Figure 29：VisualStudio 依附 Inventor 进程步骤

5. 运行规则，如果规则出错，则会自动跳转到 VS，加载规则的临时文件，自动停止在错误行。
6. 如果规则不出错，又想单步调试，则需要手动打开自动生成的临时规则文件，设置断点，然后运行规则即可。

   默认情况下，临时规则文件保存在一下路径：
   "%Temp%\iLogic Rules“

iLogic + VBA

如果我们有一些现存的 VBA 代码，想要通过 iLogic 的触发方式让代码自动运行，则可以通过 iLogic 的 RunMacro 方法来实现。如下图，在 iLogic 中调用 InventorVb.RunMacro("projectName", "moduleName", "macroName")，填入对应的参数即可。

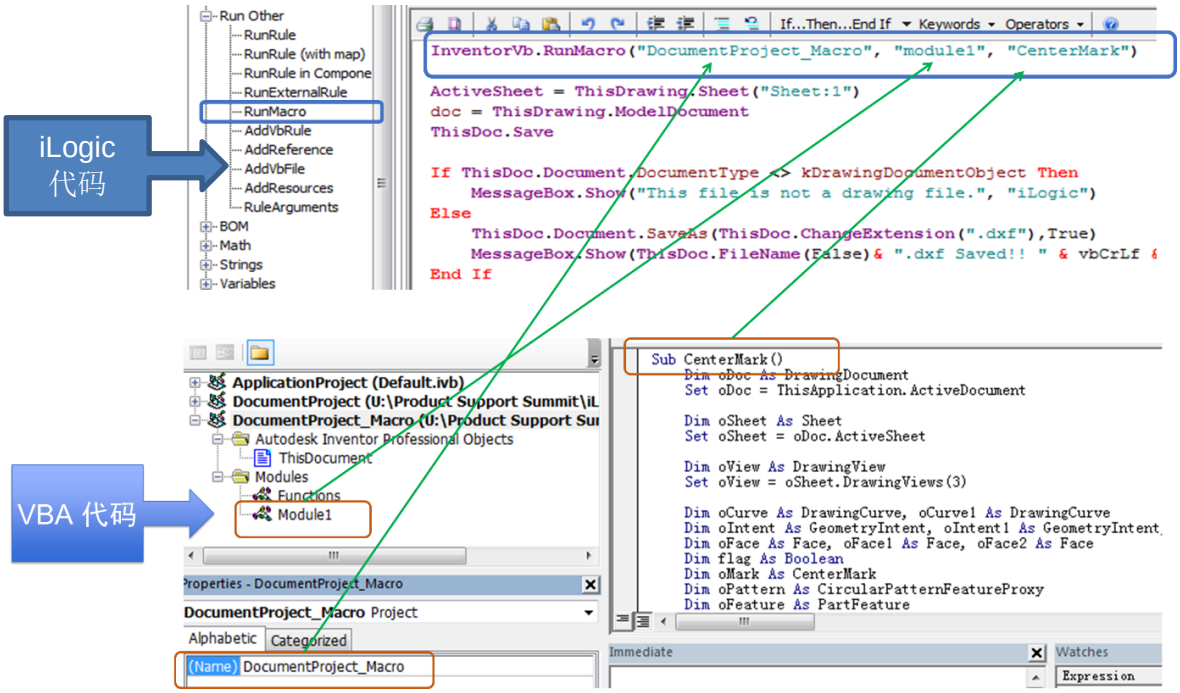

Figure 30：iLogic 中运行 VBA 程序参数对照

iLogic 小贴士和学习

iLogic 参数触发

如我们之前所说，参数引用可以自动触发 iLogic 规则。请正确理解参数引用哦，下图可以比较清楚的解释哪种情况下是引用：

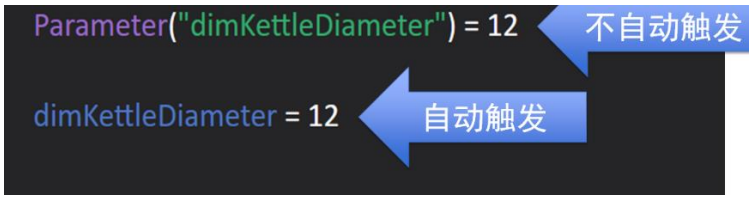

Figure 31：参数引用触发

在有些情况下，当前规则没有用到某些参数，但是又想在参数修改时被连带触发，则可以使用下面的语句，把参数以引用的形式写入规则：

Figure 32：参数引用触发 2

Excel 引擎

这个议题来自于用户的问题。如果在 iLogic 规则内使用 GoExcel 写 Excel 单元值，而且单元值写回 Excel 后，又驱动修改其他的单元值，则被驱动修改的单元值不能及时更新回 Inventor 参数列表。

出现这种情况的原因是 Inventor 2021 的新改动：默认设置 Excel 引擎为 LibXL。LibXL 提供了更好的性能，但是它却没有计算的能力。

遇到此问题的解决方案如下（任选其一）：

- 调用 iLogic API 代码内设置 Excel 引擎

```
Dim iLogicAuto = iLogicVb.Automation

iLogicAuto.ExcelEngine = ExcelEngine.COM
```

Figure 33：iLogicAPI 设置 Excel 引擎

- 从 iLogic 配置入口切换 Excel 选项为 COM 引擎。

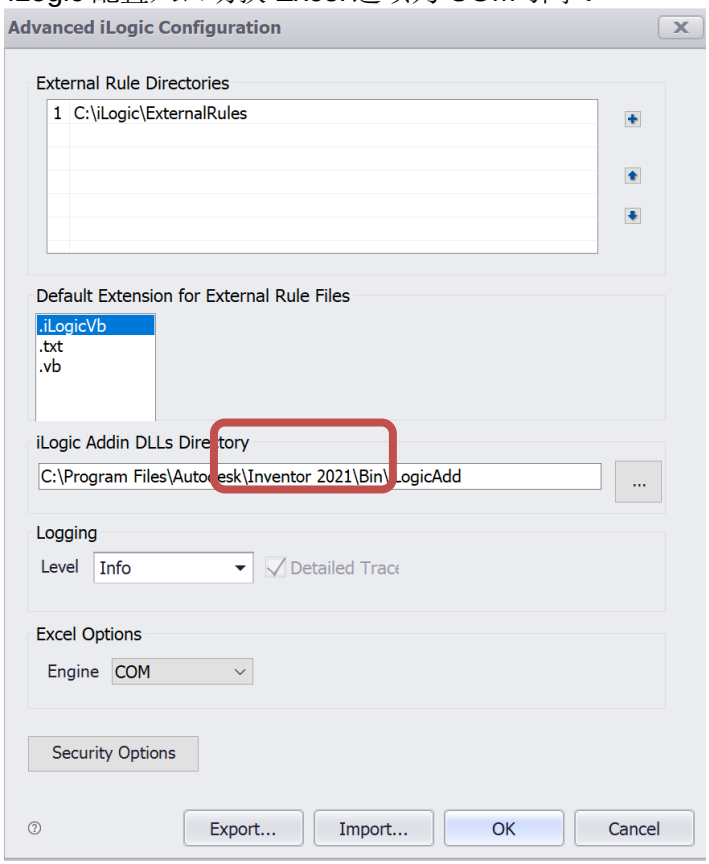

Figure 34：配置入口切换 Excel 引擎

外部规则遗失

此议题依然来自于客户问题。在某些情况下，用户会拿到一套方案，可能调用到的不重要的外部规则有丢失，然而这些外部规则又恰好在事件触发器中被监控。在这种情况下，用户在打开文件或者做某些操作时，遗失的外部规则被自动触发，则会出现错误从而弹出很多文件丢失的对话框。

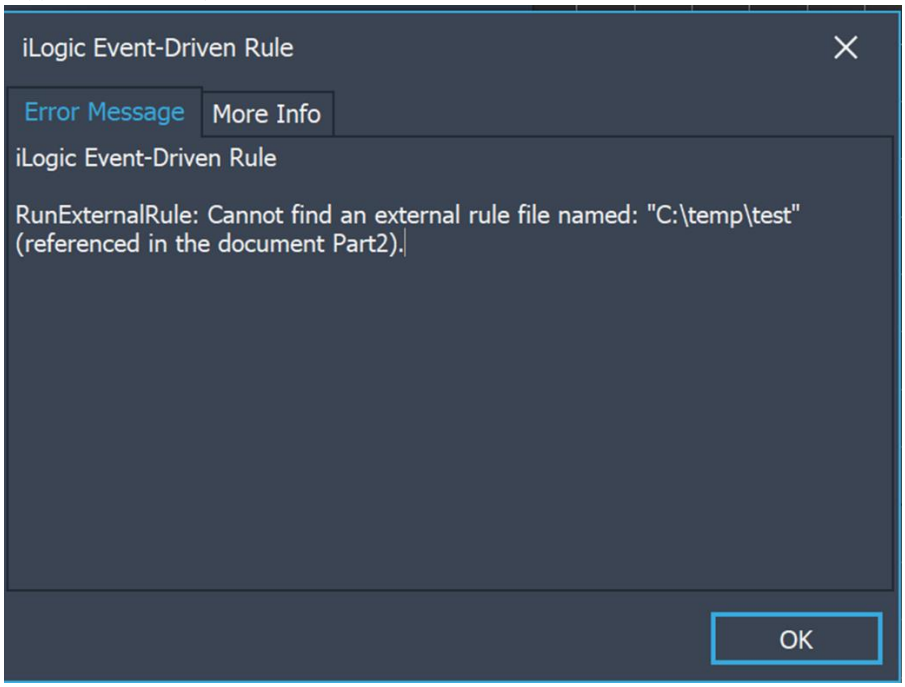

Figure 35：错误对话框

这儿提供一个小方法可以绕过去：在提示丢失的文件路径下建一个空的同名规则文件，以*.iLogicVb 或者.txt 均可。

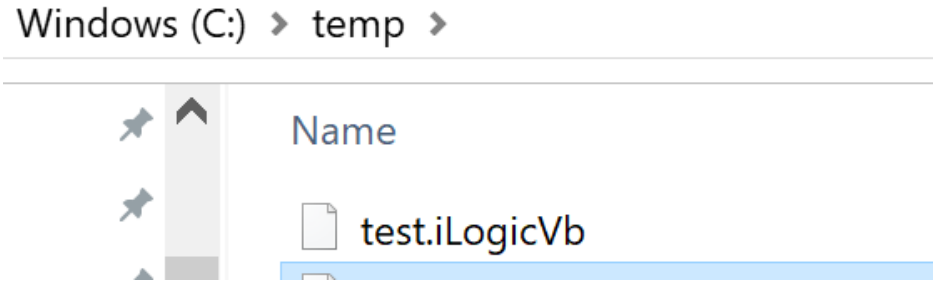

Figure 36：示例

函数注释和收缩

使用 iLogic 过程中，可能会需要写函数的注释，以及想要大段收缩一部分函数。这儿提供一个实用小功能：

- 直接在函数前输入'''，则函数的注释会自动生成，包涵有所有函数需要的输入参数列表，可以根据需要再添加其余的注释信息。
- 在规则代码中成对使用'[和]'，可达到收缩代码的功能。

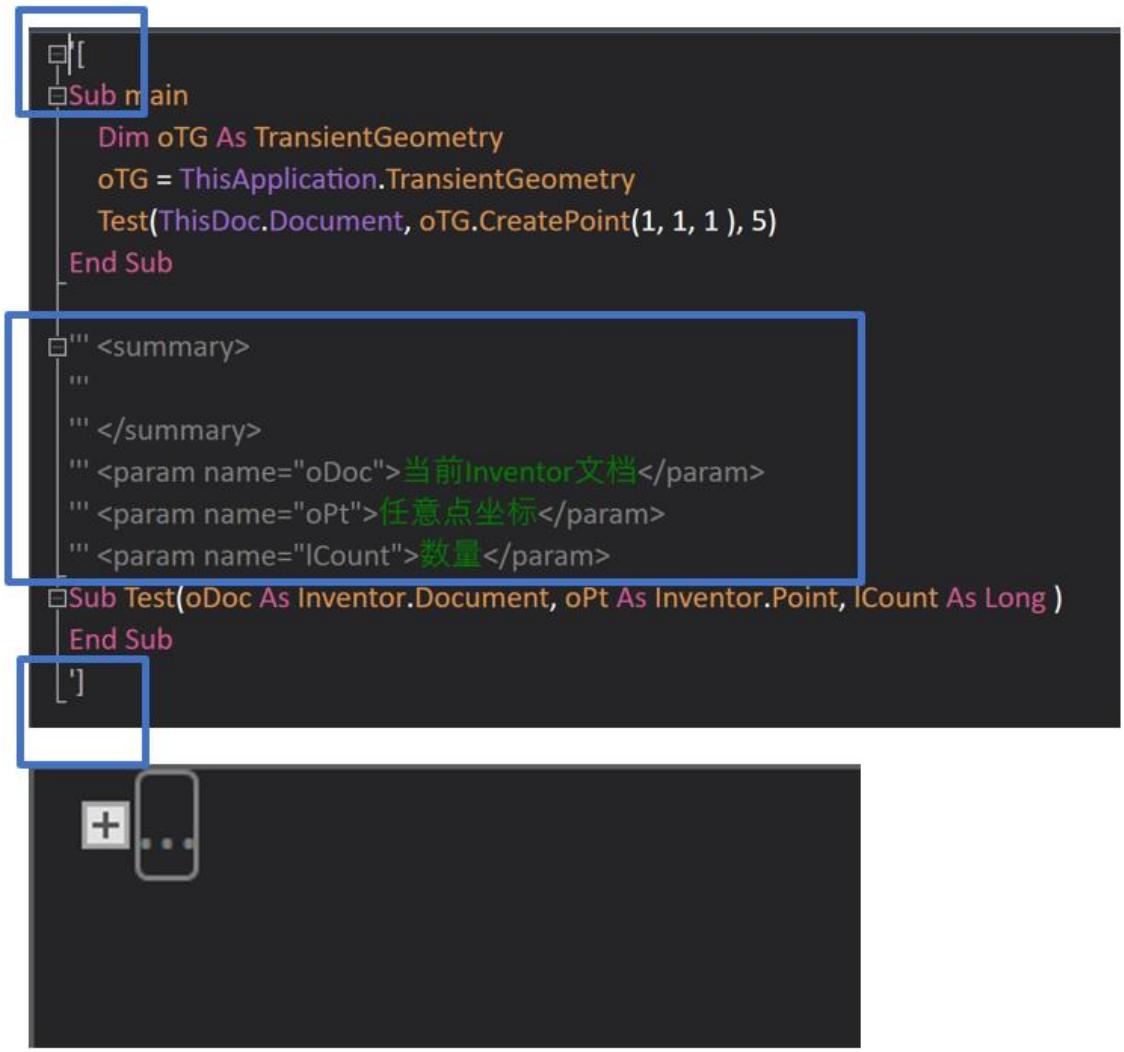

Figure 37：注释及收缩示例

工程图和模型

iLogic 中提供了通过工程图文件或者视图直接得到模型相关信息的代码段：

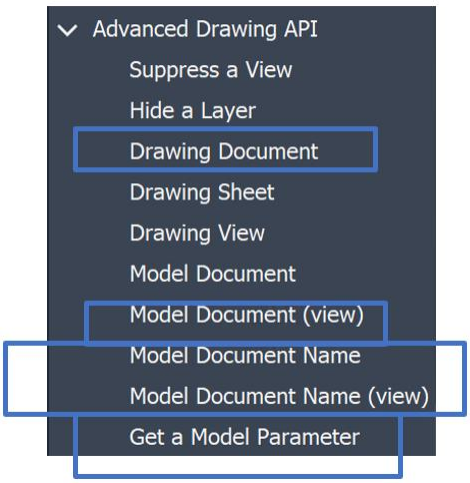

Figure 38：iLogic AdvancedDrawingAPI

双击即可获取代码，代码分享在这里供大家参考：

- **获取工程图中的模型**
  - doc = ActiveSheet.View("VIEW1").ModelDocument
  - doc = ThisDrawing.ModelDocument

- **工程图中读写模型参数**
  - Parameter(modelName, ParamName)
  - iLogicVb.Automation.ParamValue(Model, ParamName)

- **获取模型文件路径**
  - modelName = IO.Path.GetFileName(ThisDrawing.ModelDocument.FullFileName)
  - modelName = IO.Path.GetFileName(ActiveSheet.View("VIEW1").ModelDocument.FullFileName)

Figure 39：获取代码示例

iLogic + Inventor API

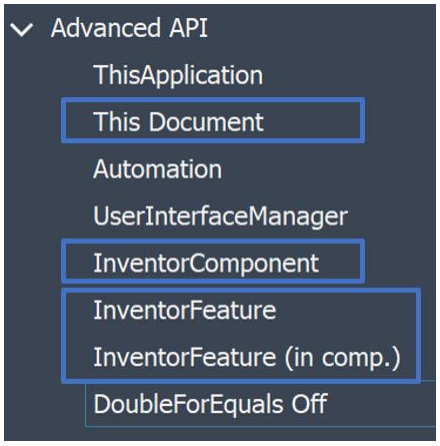

Figure 40：iLogic AdvancedAPI

同时 iLogic 也提供了直接获取 Inventor API 的接口，双击代码片段入口即可获得代码：：

- **获取Inventor API的Document对象**
  - Dim doc = ThisDoc.Document

- **获取InventorAPIComponentOccurrence对象**
  - Dim compOcc = Component.InventorComponent("PartA:1")

- **获取InventorAPI Feature对象**
  - Dim feat = Feature.InventorFeature("Extrusion1")
  - Dim feat = Feature.InventorFeature(componentOrDocName, "Extrusion1")

Figure 41：获取代码示例

学习资源

iLogic 最为便利的学习资源则是自身提供的代码片段，可以双节接口，获取代码到规则编辑器，稍微根据文档自身的情况加以修改即可使规则运行。除了下图左的系统接口代码片段，在用户自定义的标签页，iLogic 还提供了更多的功能性的代码片段，有些可以直接运行，有些则需要稍微改一下参数即可。推荐大家试一试。

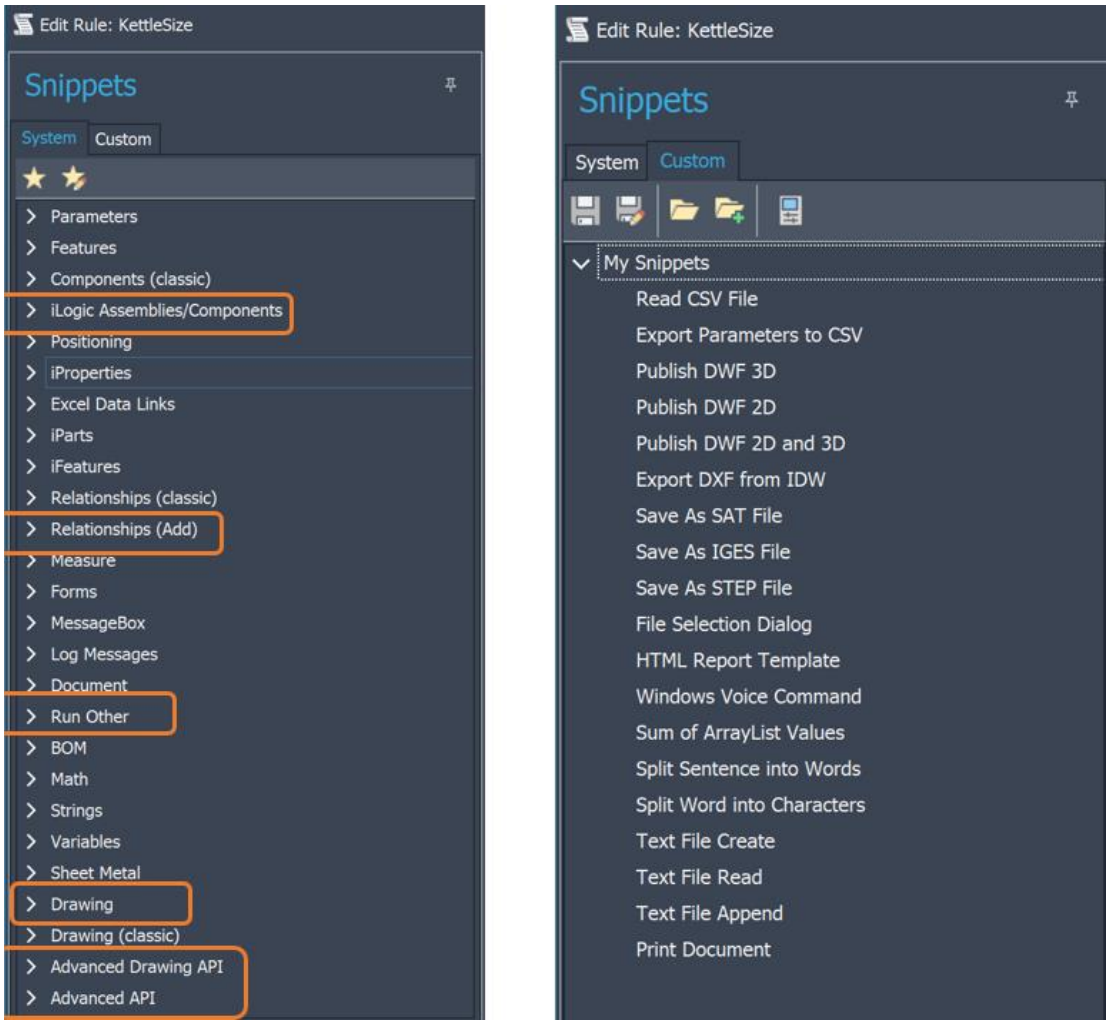

Figure 42：iLogic 代码片段接口

除了代码片段，不知您注意过没有，在 Inventor 的基础教程里面，有 iLogic 的两个手把手分步教程：

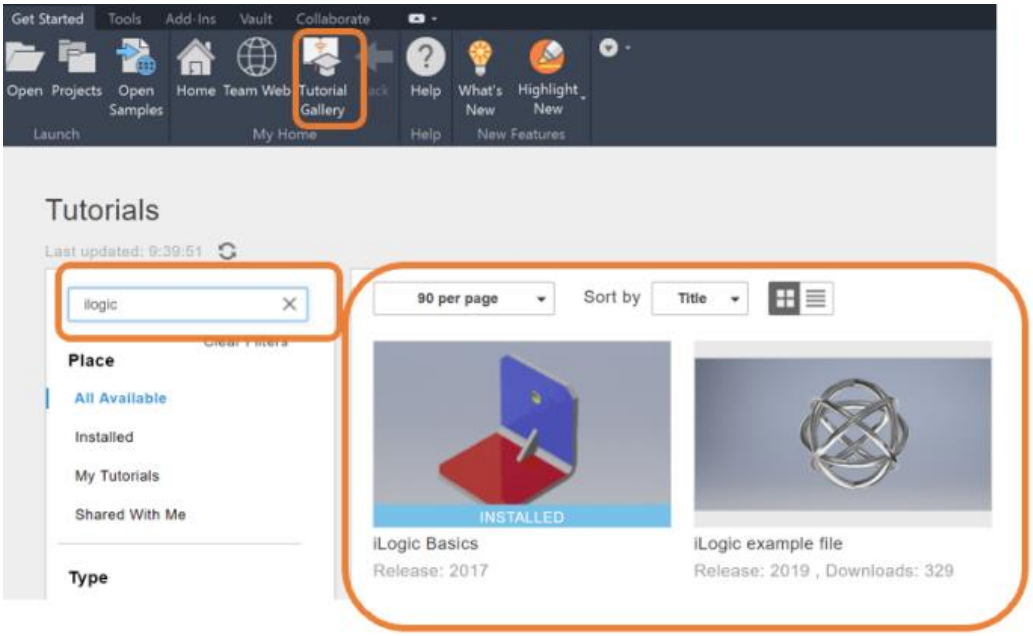

Figure 43：教程

学习的时候如果可以参考一些现有的示例拓展一下思维则会事半功倍。为了展示 iLogic 常用的功能，我们创建了下面的示例，希望可以有抛砖引玉的作用。

您可以在如下链接中找到它：

https://knowledge.autodesk.com/sites/default/files/file_downloads/BrewMain_Drawing.zip

视频展示：

https://myshare.autodesk.com/:v:/g/personal/jane_fan_autodesk_com/EdtPcaHBPzpBoEhQ7oKV-cIBRQg_GIQjbNkWJDsHUQGDtQ?e=nGKqBB

如果还要查询更多的 iLogic 以及 iLogicAPI 的接口信息，请查询帮助文档：

iLogic：

http://help.autodesk.com/view/INVNTOR/2021/CHS/?guid=GUID-AB9EE660-299E-408F-BBE1-AFE44C723F59

iLogic API：

http://help.autodesk.com/view/INVNTOR/2021/CHS/?guid=110f3019-404c-4fc4-8b5d-7a3143f129da

Inventor API 常用接口

Application

Application 指的是 Inventor 的 Application，在调用 Inventor API 时是绕不开的总结点。

```
Sub Application()
    Dim oApp As Inventor.Application
    Set oApp = ThisApplication

    Debug.Print oApp.SoftwareVersion.Major
    Debug.Print oApp.ActiveColorScheme.Name
    Debug.Print oApp.ActiveMaterialLibrary.DisplayName
    Debug.Print oApp.ActiveDocument.FullDocumentName
End Sub
```

Figure 44：Applicaton 示例

小贴士：

可以设置 ThisApplication.SilentOperation = True 来避免一些弹出对话框哦！

Document

这里说的 Document 是一个基础类，可以是任意的 Document 类型。以下 VBA 代码展示了拿到当前激活的文档，读写属性，以及获取模型树某节点全路径。

```
Sub Document()
    Dim oDoc As Document
'    Set oDoc = ThisApplication.Documents.Open(FullDocumentPath)
    Set oDoc = ThisApplication.ActiveDocument

    Debug.Print oDoc.FullFileName

    '1: Summary Information
    '2: Document Summary Information
    '3: Design Tracking Properties
    '4: User Defined Properties
    Debug.Print oDoc.PropertySets.Item("Design Tracking Properties").Item("Part Number").Value
    Debug.Print oDoc.PropertySets.Item("Summary Information").Item("Title").Value
    Debug.Print oDoc.PropertySets.Item("Document Summary Information").Item("Company").Value
    Debug.Print oDoc.PropertySets.Item("User Defined Properties").Item("Test").Value

    oDoc.PropertySets.Item("User Defined Properties").Item("Test").Value = "NewValue"
    Debug.Print oDoc.PropertySets.Item("User Defined Properties").Item("Test").Value

    '1: Favorites
    '2: Model
    '3: Representations
    Debug.Print oDoc.BrowserPanes.Item("Model").TopNode.FullPath
    Debug.Print oDoc.BrowserPanes.Item(2).TopNode.BrowserNodes.Item(1).FullPath
End Sub
```

Figure 45：Document 示例

我们来看看输出结果：

```
C:\iLogic\Models\BrewMain-Drawing\brewing-main.iam
brew-2c
Autodesk Inc.
UserDefinedValue
NewValue
brewing-main.iam
brewing-main.iam:Model States: Master
```

Figure 46：输出结果

SelectSet

如果不太确定要查询的对象是什么类型，以及此对象有什么属性，SelectSet 是一个好帮手。我们只需要以下几个步骤就可轻松查到对象类型以及对象的属性。

1. 在 VBA 代码里写入下图几行代码，设置断点在 End Sub。
2. 在 Inventor 图形窗口或者模型树，选择需要查询的对象或者节点。
3. 返回 VBA 环境，F5 运行代码。
4. 选中 oObj，添加到 Watches 窗口，我们可以获得对象的类型。
5. 展开对象，则可查询到对象的一系列属性。

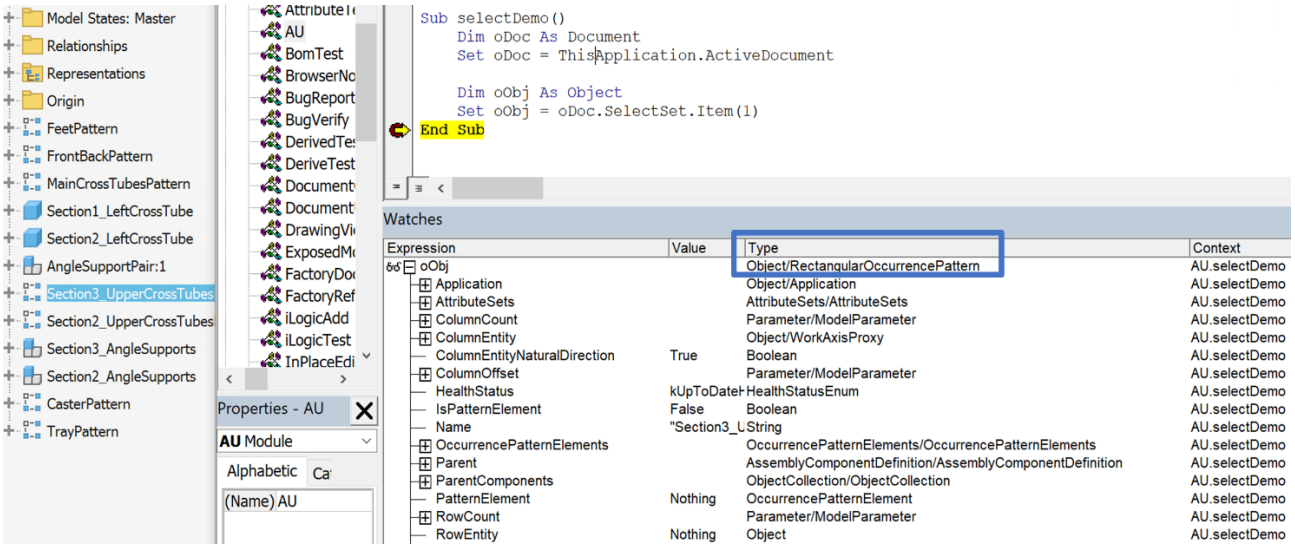

Figure 47：VBA SelectSet 示例

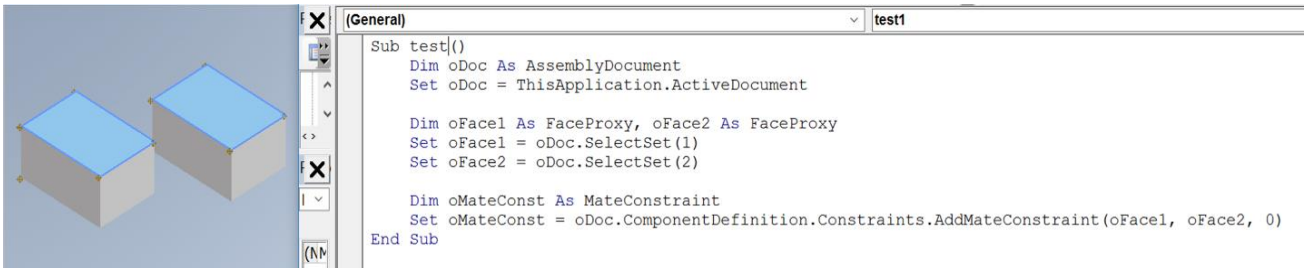

Figure 48：VBA SelectSet 示例 2

部件中的遍历

基于部件做设计自动化时，非常常见的需求就是要遍历部件所引用的所有的子零部件。下面一则示例可以提供一些思路。

示例中获取了当前的部件文档，遍历所有部件引用的文档，删除所有的用户自定义属性，同时给每个文档添加需要的属性。

大家有发现下面是 iLogic 代码，而不是 Inventor VBA 的代码吗？有什么区别呢？

```
Dim oDoc As Document
oDoc = ThisApplication.ActiveDocument

Dim osubDoc As Document
Dim oPropset As Inventor.PropertySet
Dim lCount As Long
For Each osubdoc In oDoc.AllReferencedDocuments
   oPropset = osubDoc.PropertySets("User Defined Properties")
   lCount = oPropset.Count
   If lCount > 0 Then
      For i = lCount To 1 Step -1
         oPropset.Item(i).Delete
      Next
   End If
   osubDoc.PropertySets.Item(1).Item("Author").Value = "Inventor"
   osubDoc.PropertySets(2).Item("Company").Value = "Autodesk Inc."
   osubDoc.PropertySets(3).Item("Designer").Value = "Autodesk Inc."
   osubDoc.Save2(True)
Next
```

Figure 49：部件中的遍历

批量添加参数

也许也会有一些需求，想要批量给部件所引用的零件添加某一名称或者类型的参数，下面也提供一则示例供大家参考：

```
Sub main()
    Dim oAssy = ThisDoc.Document
    Dim oDocDes As File
    Dim oDoc As Document

    For Each oDocDes In oAssy.File.AllReferencedFiles
        oDoc = ThisApplication.Documents.Open(oDocDes.FullFileName, True)
        If oDoc.DocumentType = DocumentTypeEnum.kPartDocumentObject Then
            If Not oDoc.componentdefinition.IsContentMember  Then
                'change length parameters
                PutParameter(odoc, "TestLength", 1, "length", "mm")
                'change text parameters
                PutParameter(odoc, "TestText", "This is a test", "text")
                'change bool parameters
                PutParameter(odoc, "TestBool", True, "bool")
                odoc.save
                odoc.close
            End If
        End If
    Next
End Sub

Sub PutParameter(oDoc As Document, sName As String, vValue As Object, sType As String, Optional sLengthUnits As String = "cm")
    Dim oParam As Inventor.Parameter
        oParam = FindParamUsingName(oDoc, sName)
        If oParam Is Nothing Then
            Select Case sType.ToLower()
                Case "length" : oParam = oDoc.ComponentDefinition.parameters.userparameters.addbyvalue(sName, vValue, sLengthUnits)
                Case "text" : oParam = oDoc.ComponentDefinition.parameters.userparameters.addbyvalue(sName, vValue, UnitsTypeEnum.kTextUnits)
                Case "bool" : oParam = oDoc.ComponentDefinition.parameters.userparameters.addbyvalue(sName, vValue, UnitsTypeEnum.kBooleanUnits)
            End Select
        End If

End Sub

Function FindParamUsingName(oDoc As Document, sName As String) As Inventor.Parameter
    Dim oParam As Inventor.Parameter
    Dim bFound = True
    For Each oParam In oDoc.componentdefinition.parameters
        If oParam.Name.ToLower.Equals(sName.ToLower())
            bFound = True
            Return oParam
        End If
    Next
    If Not bFound Then
        Return Nothing
    End If
End Function
```

Figure 50：批量添加参数示例

Inventor API 学习资源

想要深入使用 iLogic 进行高度设计自动化，始终也绕不开 Inventor API。与 iLogic 相比，Inventor API 需要更多的编程技能，当然也需要有一些指引，可以让我们从茫茫接口中找到自己需要的那个。我们一开始时总是不知道从何处入手，如何查询，怎么使用以及更快速地学习。现在我们来提供几个比较方便查找 Inventor API 相关信息的渠道，以及如何找到已有的代码来快速学习。

对象图

对象图就是一个名为 ObjectModel.pdf 的文件，它定义了 Inventor API 的对象的层次从属结构和对象的逻辑关系。对象图存在于 Inventor SDK 包内，所有安装 Inventor 的电脑，都可以安装对应路径下的 DeveloperTools.exe 来获得。

安装：

C:\Users\Public\Documents\Autodesk\Inventor 2021\SDK\developertools.msi

文件路径：

C:\Users\Public\Documents\Autodesk\Inventor 2021\SDK\DeveloperTools\Docs\InventorObjectModel.pdf

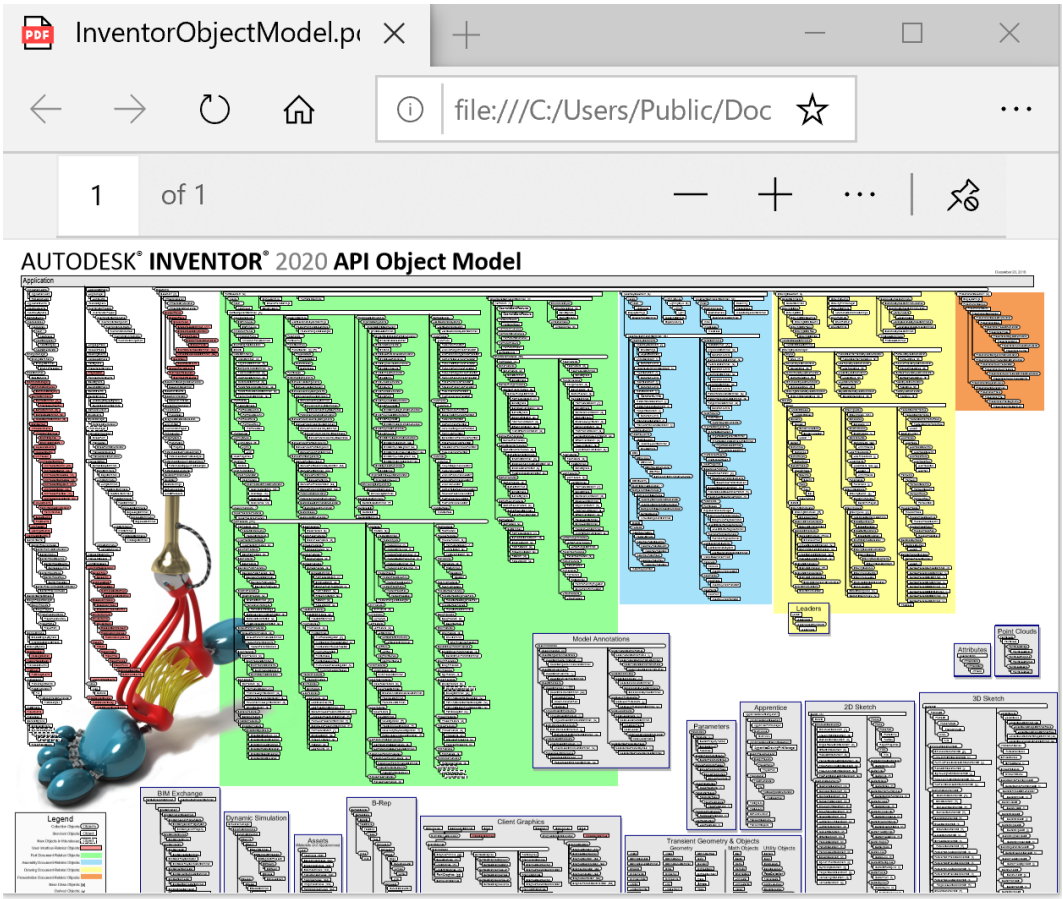

Figure 51：对象图

对象图太大了，从何处查起呢？别急，请先看左下角的导引，不同的颜色区域定义了不同的类型，是不是更方便找到自己要查找的对象了呢？

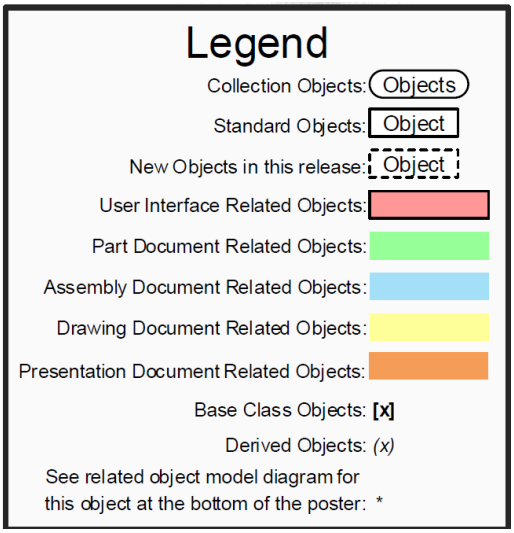

Figure 52：对象图中的图例

查询浏览器

如果知道或者已经查询到对象的类型，还想要查询更多对象的方法和属性，通过 VBA 环境的浏览器查询也是一个渠道哦！

而且，点击下方图中上方的小问号，可以直接链接到 Inventor API 的帮助文档中去。您或许能从帮助文档中找到自己需要的代码示例！

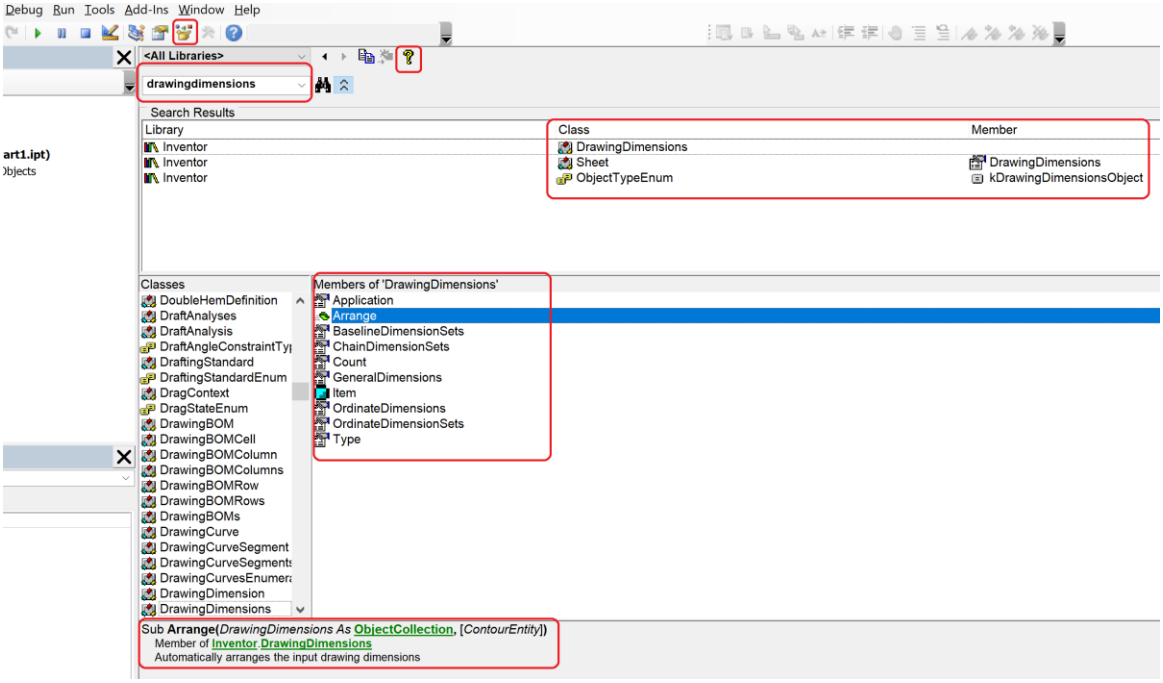

Figure 53：VBA 对象浏览器查询

查询帮助

刚才说到 Inventor API 的帮助文档，看这儿：
可以直接从 Inventor 中直接进入：

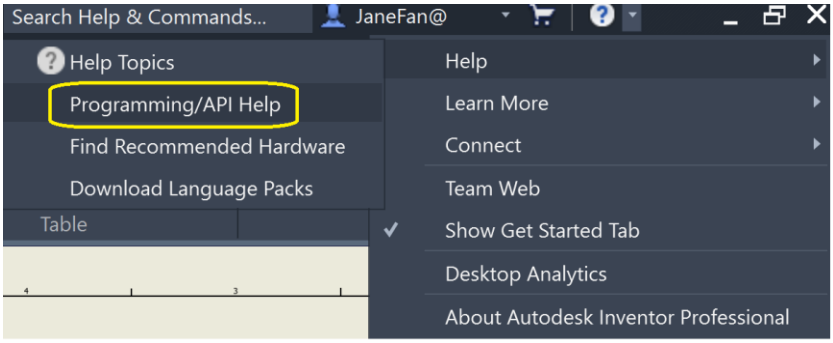

Figure 54：Inventor 帮助入口

打开如下图所示：

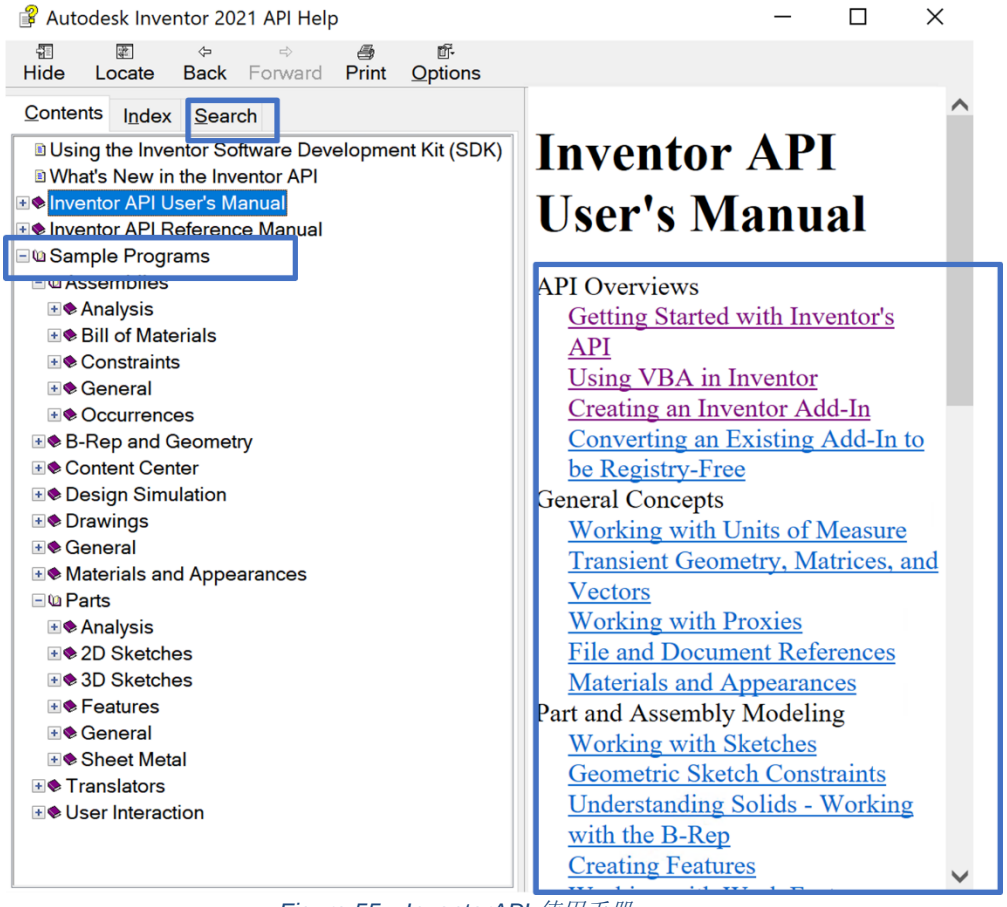

Figure 55：InventorAPI 使用手册

示例程序

刚才我们说到安装 DeveloperTools 可以获得 Inventor API 的对象图，除了对象图，其实还有更多的代码示例，如果您顺带安装了 UserTools.exe，则会获得更多的代码示例。

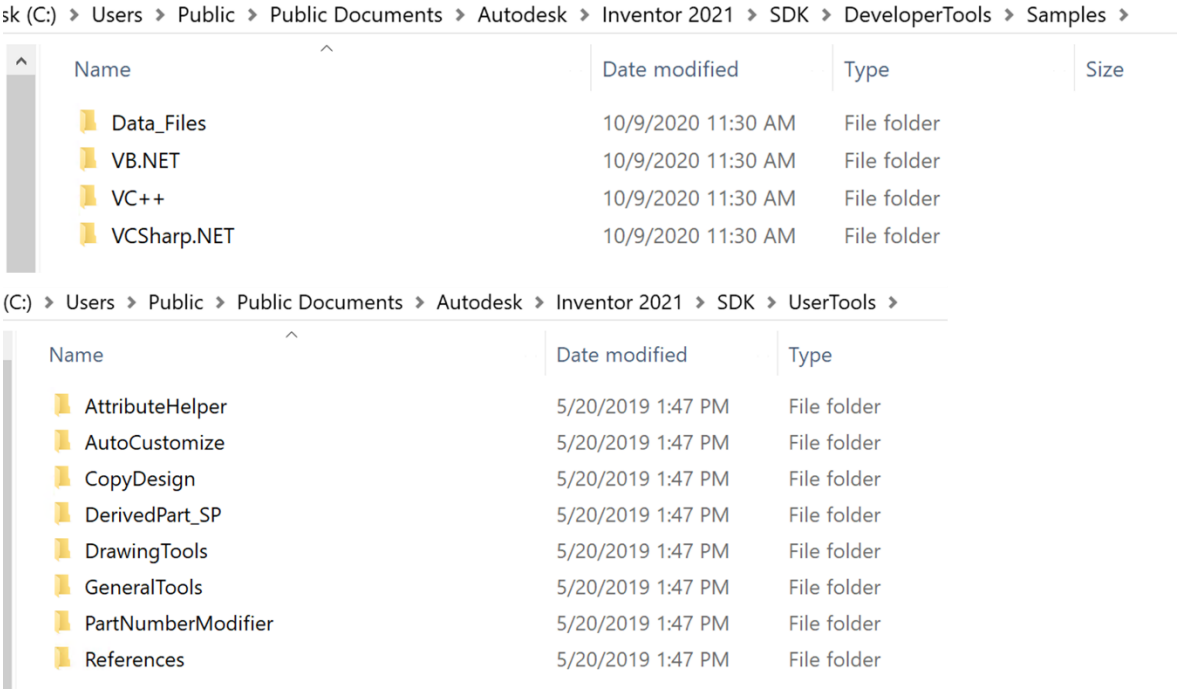

Figure 56：SDK 代码示例

交流

如果还有更多的问题想要咨询和交流，欢迎来到 Inventor 的论坛来探讨。
英文：
https://forums.autodesk.com/t5/inventor-customization/bd-p/120

中文：
https://forums.autodesk.com/t5/inventor-chan-pin-ji-shu-ying-yong-tao-lun-qu/bd-p/915

感谢！