HTML5：TC 3 HMI 的铠甲，还是软肋？

陈利君 2020.02.17

前言：TC3 HMI 是 Beckhoff 公司的组态软件，自 2017？年推出市场以来，以其功能先进、扩展性强为广大国内客户所熟知。它最大特点是基于 HTML5，使用浏览器打开指定网址就可以访问产线或者设备的组态画面。无疑这对最终用户很有吸引力，但对开发者却是个挑战：因为 TC3 HMI 涉及的 HTML 和 CSS、Java Script 和 Json 等知识对于自动化工程师而言太过陌生，毕竟大家不是做网页设计的。如果把 TC 3 HMI 比喻成屠龙宝刀，怎么舞得好另说，可怎么才舞得动？

TC 3 HMI 涉及的 HTML 技术真的那么高深吗？作为一个对 HTML 完全外行的自动化工程师，我花一周的时间亲历了从门外到门内的过程，结论是：除了 Java Script 算一种编程语言之外，HTML、CSS 和 JSON 都只是格式语言，用 LD 的工程师能学会 ST，用 ST 的工程师就能学会 Java Script，触类旁通而已，实在不必望而生畏。如果你跨过了这道坎，

下面是我整理的学习记录，一周时间，从零基础开始。相信你也可以，甚至更快！

目录：

HTML5：TC 3 HMI 的铠甲，还是软肋？
1 TC3 HMI 入门
1.1 资料下载和安装
1.2 学习 Quick Start
1.3 常用控件
1.4 常用功能
1.5 熟悉 TcHMI Package 提供的控件
1.5.1 用这些控件做画面
1.5.2 用 JavaScript 访问控件
2 控件导入和导出功能
2.1 创建和导出 TcHMI Package
2.2 导出现有项目中引用的 Package
2.3 导入 TcHMI Package
3 导入开源的代码
3.1 如何导入一个开源的控件
3.2 导入的开源.js 脚本
3.3 在.js 脚本中使用 PLC 变量
4 问题和猜想
4.1 Question
4.2 Idea
5 学习资料
5.1 JAVA Script 学习

5.1.1 资料
5.1.2 相对于 PLC 语言，JS 的特点
5.1.3 TC3 HMI Function 使用的 JavaScript
5.1.4 TC3 HMI 的控件 Event 中执行的嵌入式脚本
5.2 Json 学习
5.2.1 学习资料
5.2.2 什么是 JSON
5.2.3 Json 字符串的构成
5.2.4 从 Json 字符串读入到 Java Script 对象
5.2.5 Tc 3 HMI 的 Function 配套.json 文件
5.3 HTML 学习资料
5.3.1 HTML 是用来描述网页的一种语言
5.3.2 HTML 标签
5.3.3 Tc3 HMI 中的 HTML
5.4 CSS 学习
5.4.1 学习资料
5.4.2 什么是 CSS
5.4.3 TC3 HMI 中的 CSS 文件

1 TC3 HMI 入门

1.1 资料下载和安装

(1) **VS2017 社区版 vs_community__1354608913.1581581688.exe**

从微软官网下载，1.2M，安装时需要连网。

安装选项“通用 Windows 开发平台”和“ASP.net 和 Web 开发”，需要 18.67G 空间。网速 10Mb/s 以内，安装持续约一个小时，比以前安装 VS2015 和 VS2013 快得多。

第一次使用前需要注册一个微软帐号，激活该帐号以后就可以长期使用了。

安装最新版的 VS 和最新版的 TC3 是为了使用最新版的 TC3 HMI 的所有功能，比如自建 Framework 就需要 C#和 web 组件，只装 VS Shell 就不行。

(2) **TC31-Full-Setup.3.1.4024.4.zip**

从倍福官网下载，1012M

(3) **TE2000-HMI-Engineering.exe**

从倍福官网下载，398M

(4) **TF2000-HMI Server.exe**

从倍福官网下载，30M

(5) **TE2000_TC3_HMI_EN.pdf**

从倍福官网下载，47M

1.2 学习 Quick Start

TE2000_TC3_HMI_EN 2020.pdf

Version 1.6 2020-01-22

打印第 4 章：Quick Start，Page 27-41

一字不落，对照每个功能测试。这是对界面的熟悉过程，不可以跨越。

1.3 常用控件

(1) 历史曲线

(2) 报警

(3) 配方功能

(4) 用户管理

这部分内容，手册很清楚，网校上的资料和培训视频也齐全。

我没有一个个试，相信不会有难度。

1.4 常用功能

(1) 项目发布

手册第 5.11 Publishing，第 110 页。

写得很清楚，连上硬件应该没有什么变数。

(2) PLC 配置

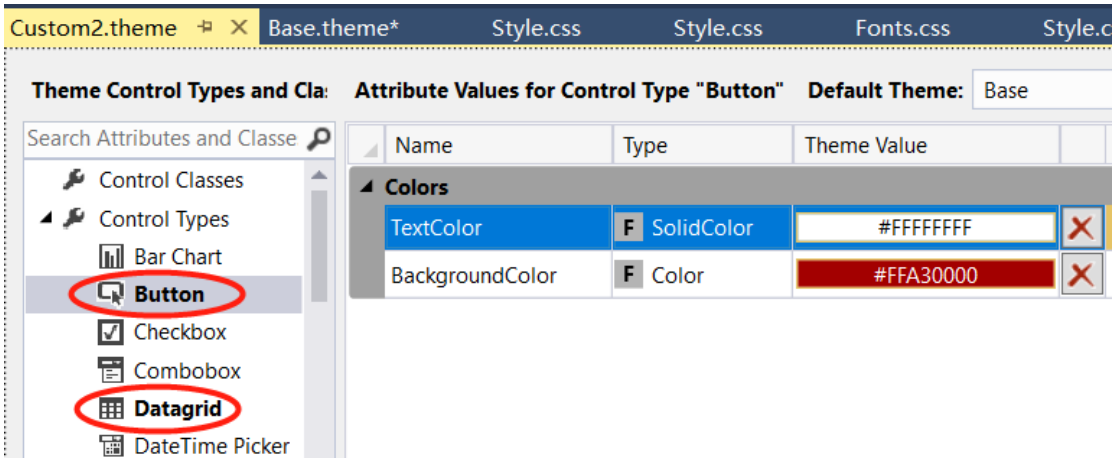
默认是 PLC1，本机的 851 端口。适用于 Server 和 PLC 在同一台控制器的情况。

如果一个 Server 要访问多个 PLC 的数据，才需要配置。

手册 13.1 ADS，第 600 页。

(3) 主题（Theme）

主题的使用比较简单，可以从按钮的 Event 事件去切换主题，也可以用 JavaScript 通过 API 函数去读取或者选择。至于每个主题要修改哪些控件的默认配置，可以手动在.theme 文件中配置。理论上每个控件的每个属性都可以修改，并且修改过的控件会以加粗显示：



高阶用户也可以在 JS 脚本中调用 API 函数“SetAttributeToThemeValue”去修改当前活动主题的某个属性值。

1.5 熟悉 TcHMI Package 提供的控件

1.5.1 用这些控件做画面

快速浏览第 6 章：Controls，Page 132-521。

这一章可以当字典查询，但不宜一项一项地照着测试。即使要查询的时候，也是打开一个 Sample 再对照手册来查询，然后修改 Sample，以加深对控件的理解。

6.2 Beckhoff 控件：工具箱中所有可以插入画面的单个可视控件的配置说明

6.3 System Control：对界面布局、指代、调用等后台运行复杂功能的控件，比如 Contrainer，Region，View，Grid 都是在 Quick Start 中使用过的。

6.5 Framework Controls，在这一节讲得很简略，但其实很常用。倍福的综合示例“TC3_HMI_Sample”中就引用了现成的 Framwork.

1.5.2 用 JavaScript 访问控件

手册第 15 章　Framework，

最重要的是 15.1.1 TcHmi，这里有倍福提供的所有组的 API 接口，就是在 JavaScript 中如何使用这些组件。实际上，在可视化界面做的任何设置，后台都会自动解释成 JavaScript 脚本，并且可以在 Code 视图中查看。只是为了兼容传统组态软件的用户习惯，保留了可视化的配置界面。

2 控件导入和导出功能

TC3 HMI 很讲究模块化编程，最终 HMI 项目的开发一定是基于很多自定义的控件。所以 HMI 开发人员的必备技能之一就是控件的单个或者批量导入和导出。

手册第 5 章 5.13 Package Management

2.1 创建和导出 TcHMI Package

(1) 新建 TwinCAT HMI 项目

选择 TwinCAT HMI | Extensibility | TwinCAT Framework Control(JS)

表示用 Java Script，也可以选其它两个使用 C#。

如果没有安装完整版的 Visual Studio，就会提示没有模版，建不了 Extensibility 项目。

(2) 导出 Package

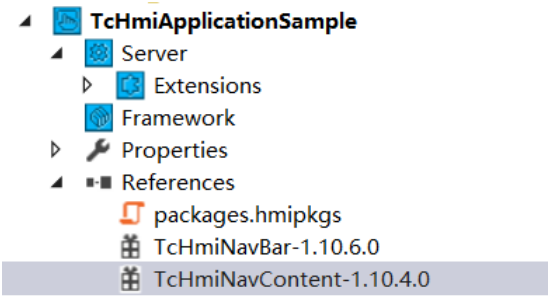
手册第 5 章 5.13.1 Creating a Package

导出的时候要选择路径。

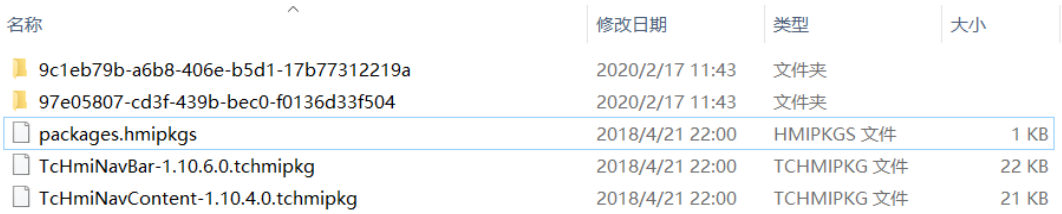
比如选择 Export to local repository，就会存到“C:\TwinCAT\Functions\TE2000-HMI-Engineering\References”

2.2 导出现有项目中引用的 Package

引用 Package 之前，这个 Package 的来源可能是自己做的，也可能是别人给的，也可能是看到某个项目里用到了一些你感兴趣的控件。比如：



如何从现有项目中提取 Package 呢？可以定位到该项目的.hmipkgs 路径下查看，比如：\TC3_HMI_Sample\TcHmiProject\TcHmiApplicationSample\.hmipkgs



名称	修改日期	类型	大小
9c1eb79b-a6b8-406e-b5d1-17b77312219a	2020/2/17 11:43	文件夹	
97e05807-cd3f-439b-bec0-f0136d33f504	2020/2/17 11:43	文件夹	
packages.hmipkgs	2018/4/21 22:00	HMIPKGS 文件	1 KB
TcHmiNavBar-1.10.6.0.tchmipkg	2018/4/21 22:00	TCHMIPKG 文件	22 KB
TcHmiNavContent-1.10.4.0.tchmipkg	2018/4/21 22:00	TCHMIPKG 文件	21 KB

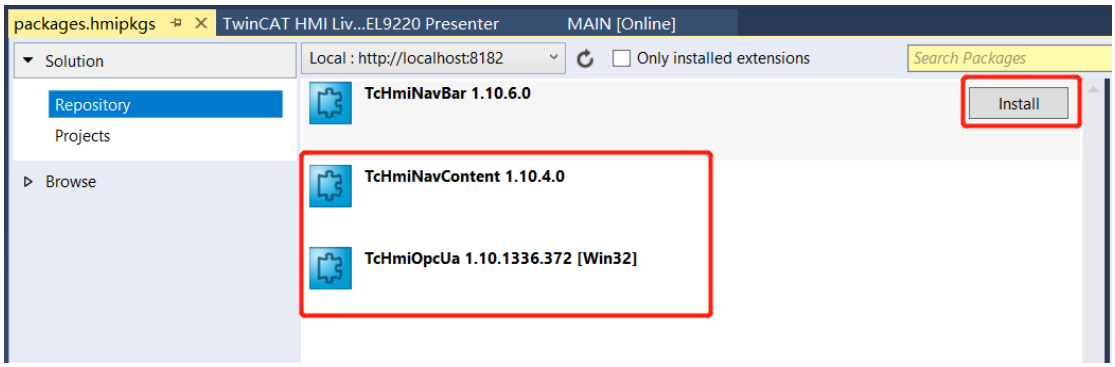
把.tchmipkg 文件复制到开发 PC 的以下路径：“C:\TwinCAT\Functions\TE2000-HMI-Engineering\References”，就可以在其它项目中安装使用这个 Package 了。

2.3 导入 TcHMI Package

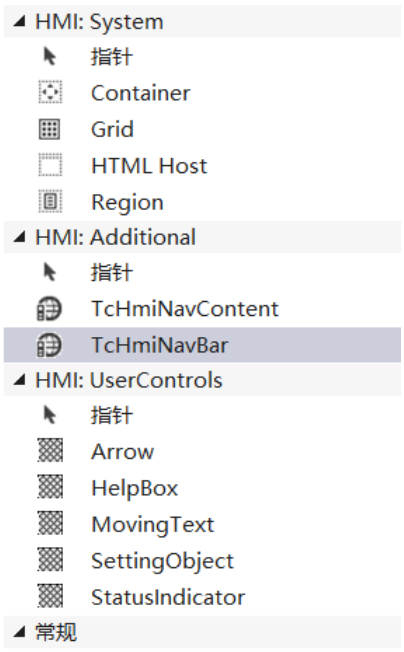
(1) 基本步骤

手册第 5 章 5.13.2 Install a Package

如果“C:\TwinCAT\Functions\TE2000-HMI-Engineering\References”下有自定义 Package，就可以在 HMI 项目的 Reference 右键菜单选择“Manage TwinCAT HMI Packages”进行安装：



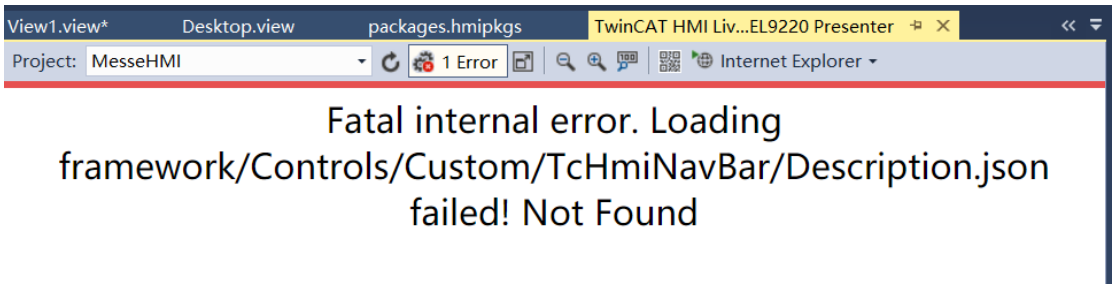
Install 要花的时间比较长，要耐心等待。等 Install 按钮变成“Uninstall”就表示安装成功。此时再打开 Toolbox，就发现工具箱中就多了 HMI:Additional：



(2) **遇到的问题及解决办法**

提示缺.json 文件

直接在新项目中插入 Reference 中添加的 TcHmiNavBar 时，提示报错：



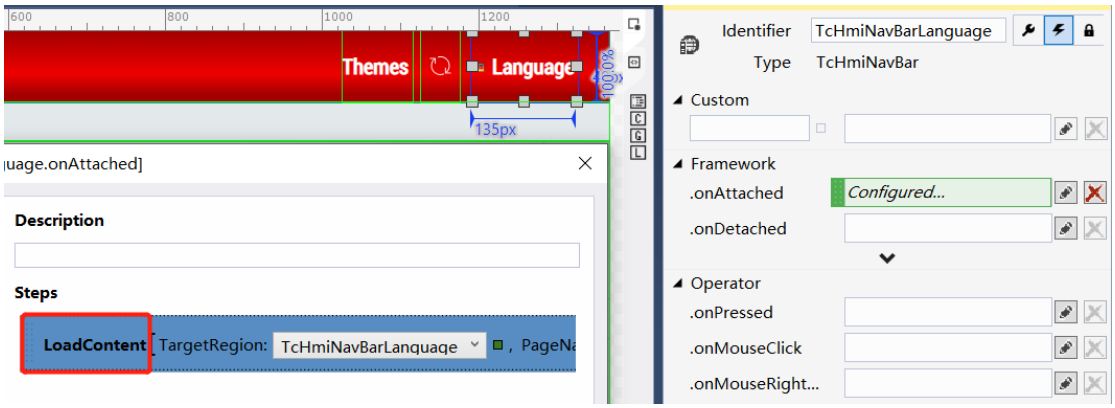
描述文件.json 找不到，但是在指定的项目下并没有这样一个 Description.json，实际上在注册后项目这个 Json 文件在 HMI 项目的以下路径：

\.hmipkgs\97e05807-cd3f-439b-bec0-f0136d33f504\Content\

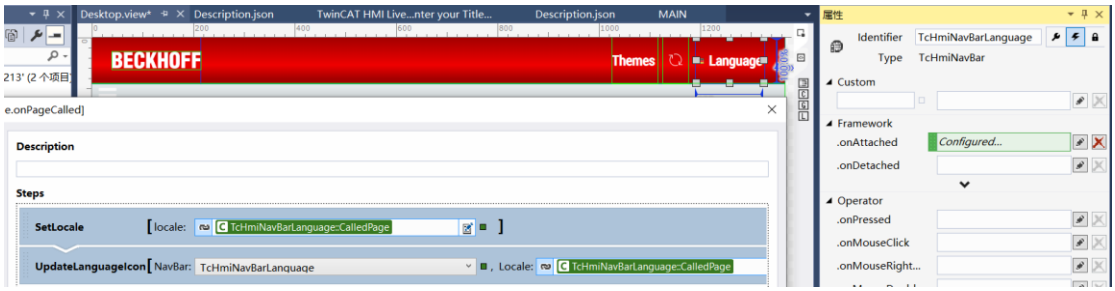
解决办法：关闭 VS 后重新打开，就可以装载了。

Event 配置中的非标 Function 是如何输入的？

这个问题在 Package 的引用无关，是通用的控件配置问题。即：调用 TcHmiNavBar 的时候，onAttached 的动作中，LoadContent 以及 SetLocale 等是怎么输入的？因为选是选择不出来，只能手动输入，但是手动输从哪里打开？



以及：



解决办法：从界面的 Code 中可以找到并编辑。可见辅助编辑的可视化界面不是万能的。

Desktop.view | Description.json | TwinCAT HMI Live...nter your Title... | Description.json

```
{
  "event": "TcHmiNavBarLanguage.onPageCalled",
  "actions": [
    {
      "objectType": "Function",
      "active": true,
      "fn": "SetLocale",
      "fnParams": [
        {
          "objectType": "Symbol",
          "symbolExpression": "%ctrl%TcHmiNavBarLanguage::CalledPage%/ctrl%"
        }
      ],
      "asyncWait": true
    },
    {
      "objectType": "Function",
      "active": true,
      "fn": "UpdateLanguageIcon",
      "fnParams": [
        {
          "objectType": "Symbol",
          "valueType": "tchmi:framework#/definitions/Control",
          "symbolExpression": "%ctrl%TcHmiNavBarLanguage%/ctrl%"
        },
        {
          "objectType": "Symbol",
          "symbolExpression": "%ctrl%TcHmiNavBarLanguage::CalledPage%/ctrl%"
        }
      ],
      "asyncWait": true
    }
  ]
},
```

自定义组件的说明书

自定义的组件是否配说明书或者功能视频？如果是正式发布或者授权的 Package，作者当然会配说明书，但如果本来就是免费共享，就需要使用者自行摸索了。

对于普通 TC HMI 用户，可能连这个 Framework 都没有包括。所以“收藏”“开源社区”“导入导出”这是必备技能，否则完全靠一个工程师自己做出一个机器的界面，是非常困难的。

3 导入开源的代码

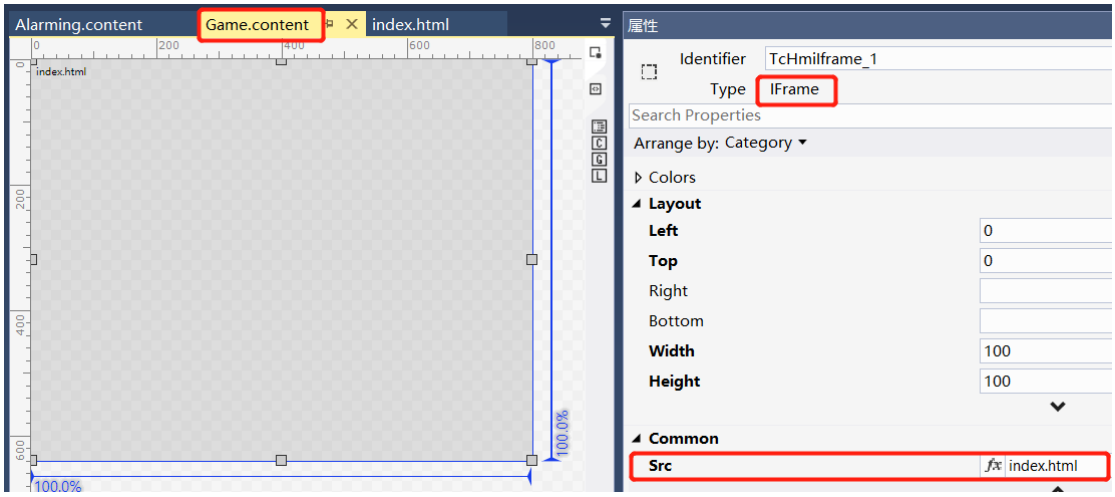
3.1 如何导入一个开源的控件

以“切西瓜”游戏为例

(1) 拷入相关文件

将它的 Html 和相关的图片 Logo 都拷入项目文件夹。

(2) 创建一个 Iframe，并指定 Source 为某个 html 文件



Region.Target —> Game.content —> Iframe.Src —> Index.html —> all.js

(3) 在 Index 文件中指定.js 文件

游戏的代码就在.js 里面，而相关的图片、声音、图标，都需要 Copy 到项目路径下。因为.js 文件会引用这些图片、声音和图标，如果没有这些文件或者放的路径不对，.js 文件就会执行不了。

index.html* TwinCAT HMI Live...nter your Title...

```
1   <!DOCTYPE html>
2   <html>
3   <head>...</head>
15  <body>
16
17      <p></p>
18      <div id="extra"></div>
19      <em> -- Fruit Ninja -- </em>
20      <em> The game is developed by the Baidu JS team, </em>
21      <em> we provide the source in git: https://github.com/ChineseDron/fruit-ninja </em>
22      <em> follow me on weibo http://weibo.com/baidujs </em>
23      <em> or learn more, to see http://tangram.baidu.com </em>
24      <canvas id="view" width="10" height="10"></canvas>
25
26      <div id="desc">
27          <!--<div style="text-align:center;clear:both;">
28              <script src="/gg_bd_ad_720x90.js" type="text/javascript"></script>
29              <script src="/follow.js" type="text/javascript"></script>
30          </div>-->
31          <div id="browser"></div>
32      </div>
33
34      <script src="scripts/all.js"></script>
35  </body>
36  </html>
```

(4) **运行效果**

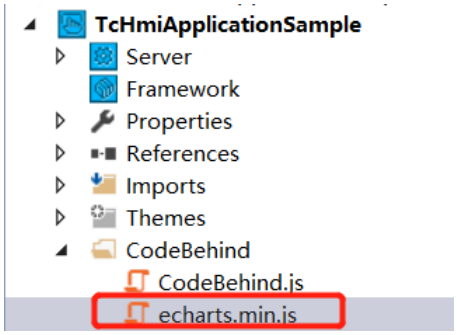
(5) **扩展：如何让游戏与 PLC 变量关联？**

在.js 文件中，利用函数读取 PLC 变量，并参与游戏进程。

这个方法可以导入任何网页版游戏，或者其它感兴趣的内容。

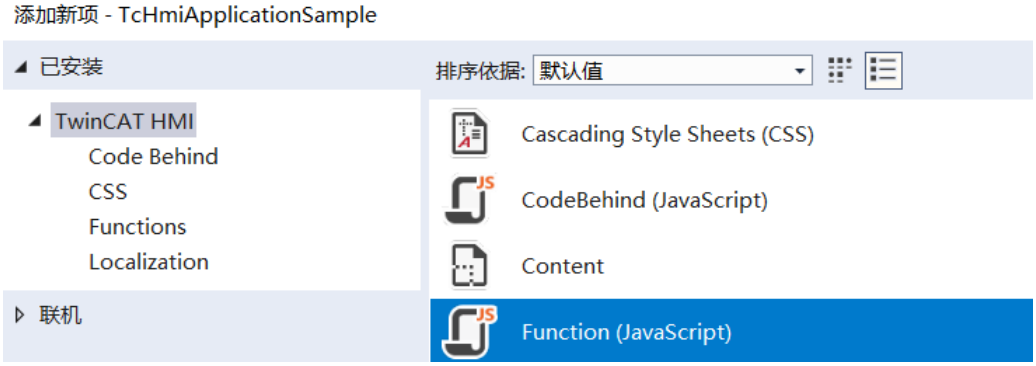
3.2 导入的开源.js 脚本

(1) **把核心代码的.js 文件复制到 CodeBehind，并添加现有项：**

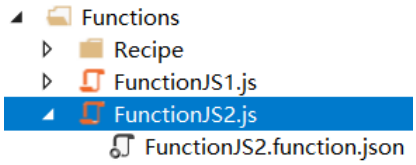


(2) **新建一个 JavaScript 的 Function**

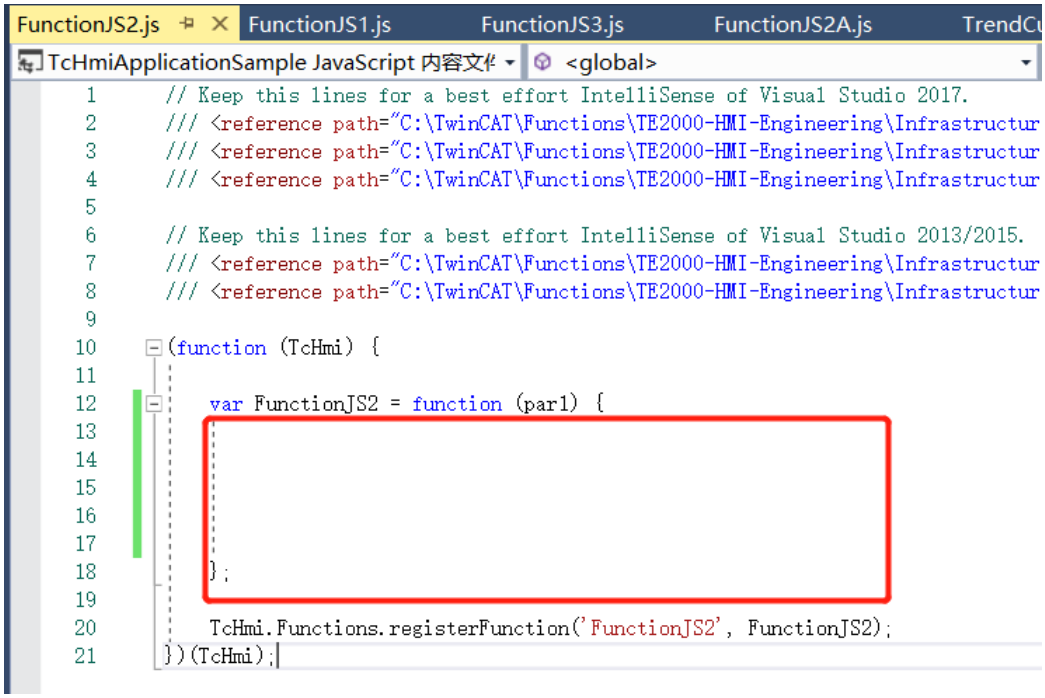
新建引用 echarts 的 Function



命名为 FunctionJS2，此时它的.json 文件也自动建立了：

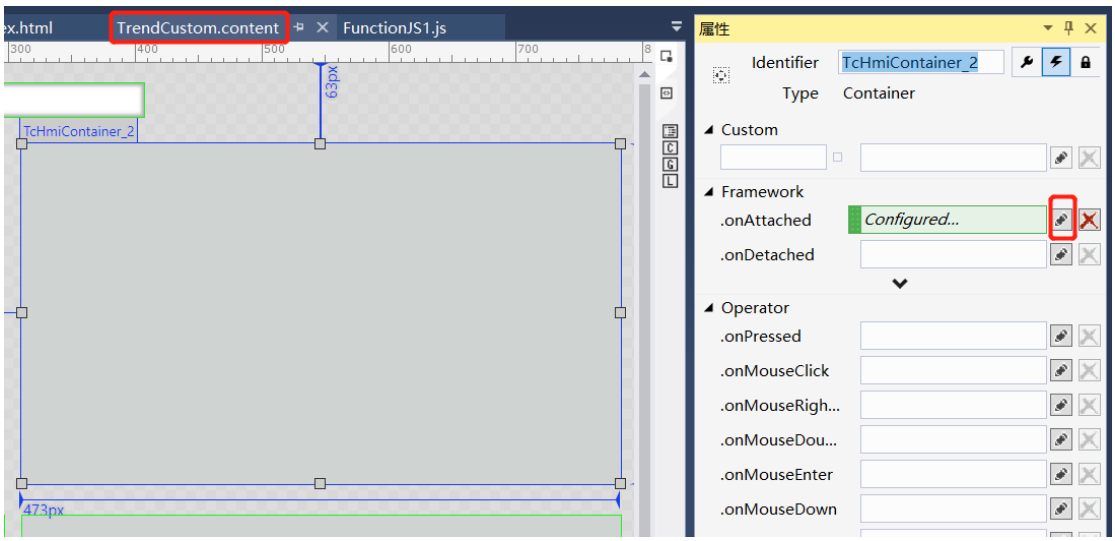


双击.js 文件，可以看到模板已经自动添加的代码：

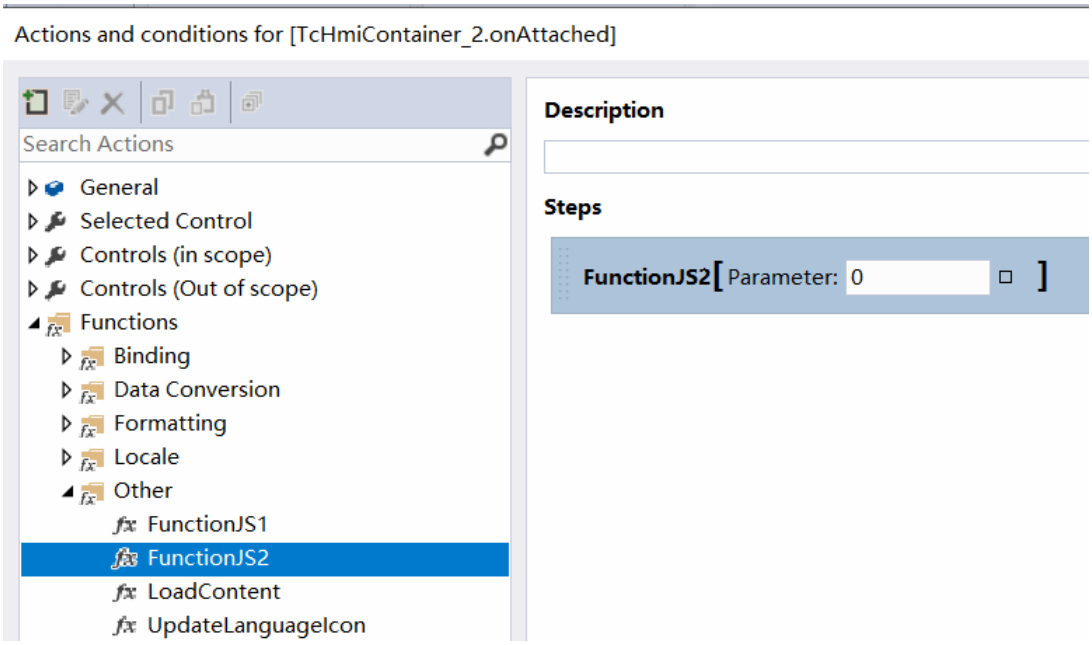


用户代码以后将添加到红线框的位置。

(3) 在 View 或者 Content 的画面上创建一个 Contrainer



并在它的 Framework 的.onAttached 事件中指定 Function：



(4) **把下载的 js 实现代码粘贴到 FunctionJS2 中**

```
(function (TcHmi) {

    var FunctionJS2 = function (par1) {
        var dom = document.getElementById("TcHmiContainer_2");
        var myChart = echarts.init(dom);
        var app = {};
        option = null;

        option = ...;

        if (option && typeof option === "object") {
            myChart.setOption(option, true);
        }

    };

    TcHmi.Functions.registerFunction('FunctionJS2', FunctionJS2);
})(TcHmi);
```

注意 dom 的用法，如果不声明一个 dom，引用的控件就无法初始化。TcHmiContainer_2 是准备放这个控件实例的容器名字。

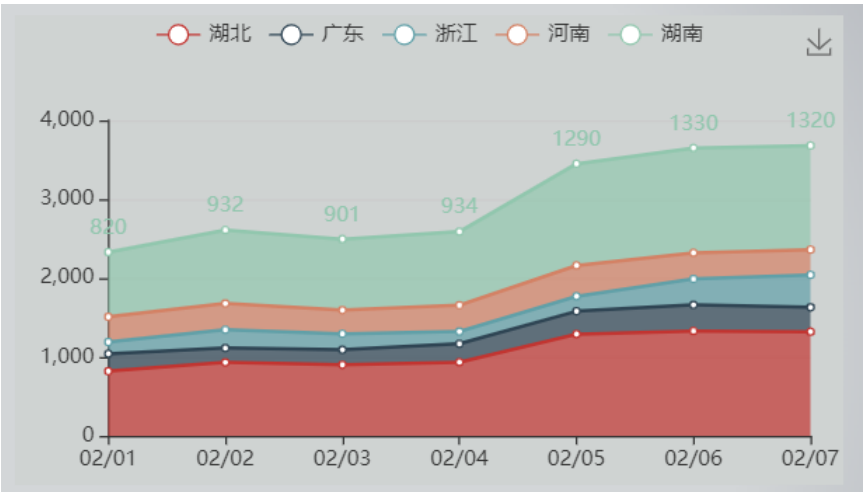
Option 中的代码隐藏了，里面是对控件的各个元素赋值。每个控件都会不一样，例中 echart 的 Option 是这样的：

```
option = {
        title: {
            text: ''
        },
        tooltip: {
```

```
        trigger: 'axis',
        axisPointer: {
            type: 'cross',
            label: {
                backgroundColor: '#6a7985'
            }
        }
    },
    legend: {
        data: ['湖北', '广东', '浙江', '河南', '湖南']
    },
    toolbox: {
        feature: {
            saveAsImage: {}
        }
    },
    grid: {
        left: '3%',
        right: '4%',
        bottom: '3%',
        containLabel: true
    },
    xAxis: [
        {
            type: 'category',
            boundaryGap: false,
            data: ['02/01', '02/02', '02/03', '02/04', '02/05', '02/06', '02/07']
        }
    ],
    yAxis: [
        {
            type: 'value'
        }
    ],
    series: [
        {
            name: '湖北',
            type: 'line',
            stack: '总量',
            areaStyle: { normal: {} },
```

```
                data: [820, 932, 901, 934, 1290, 1330, 1320]
            },
```

(5) **显示效果**



可以看到几条曲线显示的值，都是 Opiton 中定义的固定值

3.3 在.js 脚本中使用 PLC 变量

例如前面显示曲线的例子，对于自动化项目来说，这些数据通常都来自 PLC，曲线应该是根据 PLC 变量的值动态变化的。这个示例中，Series[]中的 Data 值，就不该是一个常量组成的数组，而应该是一个数组变量。假如要用在画面中显示本周新冠肺炎各省确诊病例的趋势图，就可以在上述示例中增加以下步骤：

(1) **在 PLC 中新建一个结构体“ST_SARS2”，**

```
TYPE ST_SARS2 :
STRUCT
    Hubei:        ARRAY[1..7] OF INT:=[100,20,30,35,38,40,41];
    Guangdong:    ARRAY[1..7] OF INT:=[10,100,30,35,38,40,41];
    Zhejiang:     ARRAY[1..7] OF INT:=[10,20,100,35,38,40,41];
    Sichuan:      ARRAY[1..7] OF INT:=[10,20,30,100,38,40,41];
    Beijing:      ARRAY[1..7] OF INT:=[10,20,30,35,100,40,41];
    Shanghai:     ARRAY[1..7] OF INT:=[10,20,30,35,38,100,41];
END_STRUCT
END_TYPE
```

并在 Main 中声明一个变量“arrayChina”：

```
arrayChina                    : ST_SARS2;
```

(2) **在 FunctionJS2.js 中用 SetInterval 刷新数据**

在 FunctionJS2.js 的脚本中，并用 SetInterval (function{ }, 200)每 200ms 用函数

Symol.ReadEx 读取一次数组 arrayChina 的值：

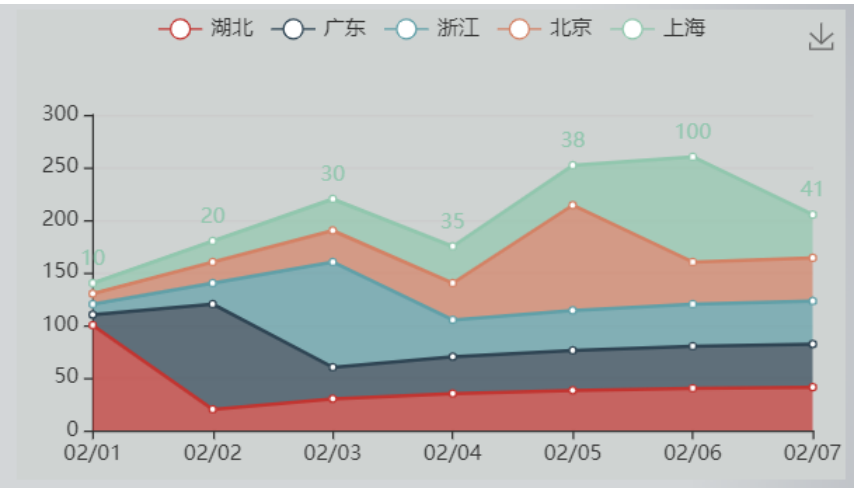
```
var myData = null;
var data = [];
setInterval(function () {
        TcHmi.Symbol.readEx2("%s%PLC1.MAIN.arrayChina%/s%", function (data) {
                if (data.error === TcHmi.Errors.NONE) {
                    myData = data.value;
                }
            },200);
```

(3) **同样在 SetInterval 中利用读取的数据刷新曲线**

读入的值 myData 可以用来刷新 Echart 的曲线，例如：

```
myChart.setOption({
    series: [{          name: '湖北',
                        type: 'line',
                        stack: '总量',
                        areaStyle: { normal: {} },
                        data: myData.Hubei}]
});
```

(4) **运行结果**



对照 PLC 变量 arrayChina 中的数据：

Hubei: ARRAY[1..7] OF INT:=[100,20,30,35,38,40,41];

Guangdong: ARRAY[1..7] OF INT:=[10,100,30,35,38,40,41];

Zhejiang: ARRAY[1..7] OF INT:=[10,20,100,35,38,40,41];

Sichuan: ARRAY[1..7] OF INT:=[10,20,30,100,38,40,41];

Beijing: ARRAY[1..7] OF INT:=[10,20,30,35,100,40,41];

Shanghai:ARRAY[1..7] OF INT:=[10,20,30,35,38,100,41];

可见图表中显示的正是结构体 arrayChina 中的数据

(5) **扩展：如果显示的数组大小变化**

如果希望显示不只是 7 天，而是 10 天的数据，应该怎么办呢？

修改 PLC 中的结构体中数组的大小

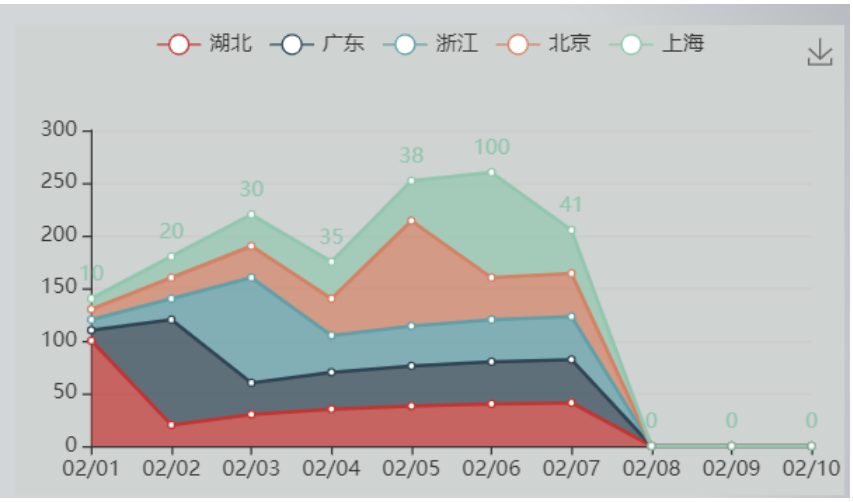
```
TYPE ST_SARS2 :
STRUCT
    Hubei:          ARRAY[1..10] OF INT:=[100,20,30,35,38,40,41];
    Guangdong:      ARRAY[1..10] OF INT:=[10,100,30,35,38,40,41];
    Zhejiang:       ARRAY[1..10] OF INT:=[10,20,100,35,38,40,41];
    Sichuan:        ARRAY[1..10] OF INT:=[10,20,30,100,38,40,41];
    Beijing:        ARRAY[1..10] OF INT:=[10,20,30,35,100,40,41];
    Shanghai:       ARRAY[1..10] OF INT:=[10,20,30,35,38,100,41];
    strDate:        ARRAY[1..10] OF STRING(8):=['02/01', '02/02', '02/03', '02/04', '02/05', '02/06',
'02/07', '02/08', '02/09', '02/10'];
END_STRUCT
END_TYPE
```

注意，为了不在.js 代码中由于数组大小的不确定而修改 X 轴的标记，特意在结构体中增加 strData 数组，以初始化将来在界面上显示的日期字符。

修改.js 文件中 X 轴的显示

```
    setInterval(function () {
        TcHmi.Symbol.readEx2("%s%PLC1.MAIN.arrayChina%/s%", function (data) {
            if (data.error === TcHmi.Errors.NONE) {
                myData = data.value;
            }
        });
        myChart.setOption({
            xAxis: [
                {
                    type: 'category',
                    boundaryGap: false,
                    data: myData.strDate
                }
            ],
            series: [
            ]
        });
    }, 200)
    if (option && typeof option === "object") {
        myChart.setOption(option, true);
    }
```

运行结果

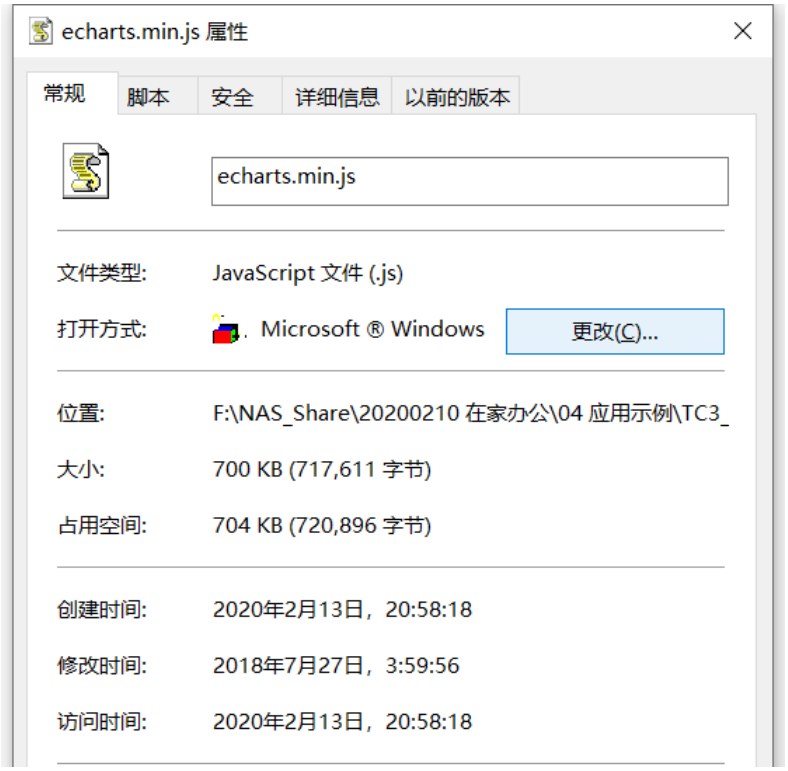


这样就不必为每次显示的数组大小而修改代码了。

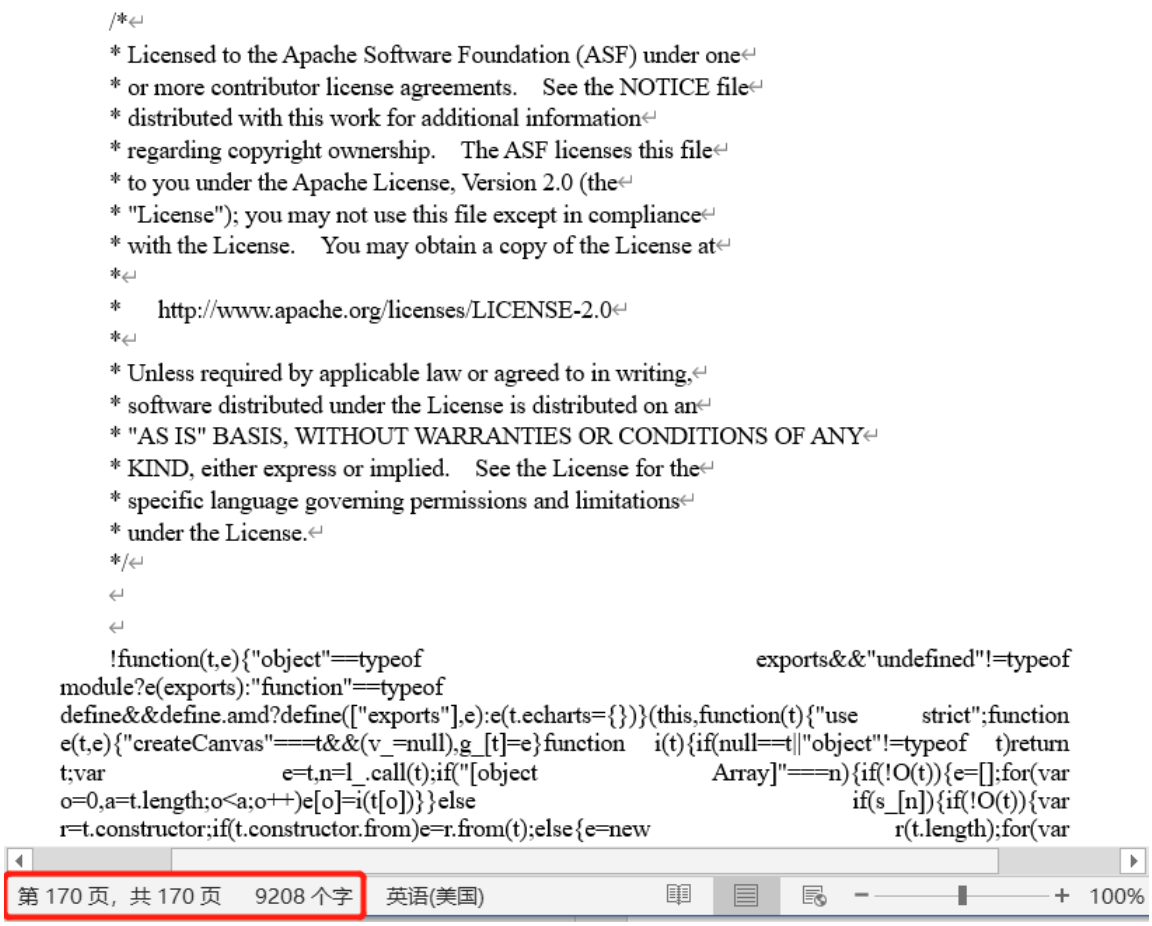
(6) 扩展：如果要修改曲线的显示方式

本例中这是几条曲线迭加的效果而不是独立显示的，这个效果是在控件的核心代码 echarts.min.js 中定义的，如果想修改曲线的显示方式而控件本身又没有开放相应的接口，就只能自己打开源代码去修改了。

有兴趣的话可以打开 echarts.min.js 来看看：



echarts.min.js 的代码放在 Word 中显示，需占用 170 页 A4 纸：



可见要写一个好用的控件相当不容易，好在互连网可以汇集众人的力量。而 TC3 HMI 开放了引用.js 文件的功能，就等于是让用户站在众人的肩膀上进行创作，而不是使用组态软件厂家有限的控件。当然，要使用无限的互联网资源，学习一些 JavaScript 的知识也是必不可少的。

(7) 示例代码

详见“JS 中引用 PLC 变量.zip”

4 问题和猜想

4.1 Question

(1) 如果开发 HMI 的人没有 PLC 项目源码

TC3 HMI 的开发环境中，如何通过 PLC 项目文件获取变量，比如 tpy 或者 tmc?

2020/02.14 的新品介绍“beckhoff_新产品_16x9_cn.pptx”显示：Tc3 HMI Version 1.12 支持 TMC 文件。

(2) 如果变量多，还是推荐组成结构体吗？

显然 TC3 HMI 有从 Server 的 PLC 读数结构体的能力，鉴于底层还是 ADS 通讯，所以其它高级语言与 PLC 做 ADS 通讯的经验/教训仍然适用。成千上万个离散的变量，放到哪

种通讯都是效率很低的。

4.2 Idea

(1) Code 界面 vs 可视化界面

熟悉 js 的工程师，完全可以在记事本而不是 TC3 HMI 中开发界面。可视化配置界面确实不是必须的，Code 视图中的内容才是。

传统的组态软件用户习惯在可视化界面操作，而熟悉 JavaScript 和 TcHMI 的用户理论上可以脱离可视化界面而直接在 Code 视图中编辑。实际上确实可以用记事本打开.view 文件，当然也可以修改。

现实的做法是，优先使用可视化的配置界面，仅当需要添加配置界面没有提供的功能，才自己进入 Code 界面进行脚本编辑。

(2) think more

.view 文件只包含 div 语句，不能冠以<html>和<body>执行，否则会执行报错。而尝试在.view 中加入<p>这种最常用的 HTML 标签，则运行时既不报错，也不显示。

要验证.view 的工作原理，可以在 Chrome 浏览器打开网页的时候，显示其 HTML 代码（默认快捷键 Ctrl+Shift+I）也许可以发现一些线索。不过这与 TC3 HMI 的正常使用没有关系了，纯属个人好奇。

总之，TC3 HMI 不允许在最底层执行 HTML 文件。HTML 文件只能以 Iframe.Source 的形式引用到 HMI 项目。

5 学习资料

5.1 JAVA Script 学习

5.1.1 资料

https://developer.mozilla.org/zh-CN/docs/Web/JavaScript

凡是在 TE2000 手册上搜索不到的 Method，都可以在这个 JAVA 学习网站上找到。

虽然这里有完整的教程和参考资料，但是 Tc HMI 的用户并不需要完整阅读，因为 TC HMI 并不要求从零搭一个网站，绝大部分做网页的功能已经在模板中自动实现了。在这个网站上只要查询一些基本的语法、函数就可以了。如果是简单的普及性的内容，也可以查看百度文库中的两个简易资料（见附件）：

JavaScript(课件).ppt

JAVA-SCRIPT 入门学习.pdf

5.1.2 相对于 PLC 语言，JS 的特点

(1) 关于句尾的“；”，推荐添加，但是也可以不加

(2) 注释：js 中注释用 /* */，而 HTML 中用 <!-- -->，

后者在 Tc3 HMI 的编辑器中会自动成对。 等字符也有智能输入功能，只要写一个< 就会弹出 p a picture 等供选择。

(3) 变量和函数声明都不需要指定类型

- Var I;
  - 无须申明类型

```
function count( int I ) {
    Var t = 0;
    I = I + t;
    Return I;
}
```

(4) 对象种类

其对象除了标准的数组、日期、字符串、数字等内置对象之外，还有浏览器对象和 ActiveX 对象以及自定义对象。比如 TcHmi 中定义的对象 Server、Symbol 等

(5) 执行条件在( )里面，而执行代码在{ }里面

IF 语法

条件在 If( ) 的括号里，没有 Then，满足条件要执行的代码在{ }中。表示判断的等于，用两个“＝”连起来，即“＝＝”。而一个“=”就表示赋值。例如：

◆ 条件判断 – if

```
• If ( 1 == i ) {
• } else if ( 2 == i ) {
• } else {
• }
```

Switch 语句

Switch 类似 PLC 的 ST 语言中的 CASE 语句，但是选择分支的变量用括号，而以 Case n 来分开不同的分支：

```
Switch( i ) {
        Case 1:
            Y = y +1;
            Break;
        Case 2:
            Y = y +2;
            Break;
            default:
           Y = y +1;
}
```

For 和 While

◆ 循环 – for

```
• For ( var I = 0 ; I < 10 ; I ++ ) {
    - ……
• }
```

◆ 循环 – while

```
- While ( 9 == i ) {
    • ……
- }
```

注意 For ( )的条件，要执行的代码用{ }而不是用 DO 来引导。

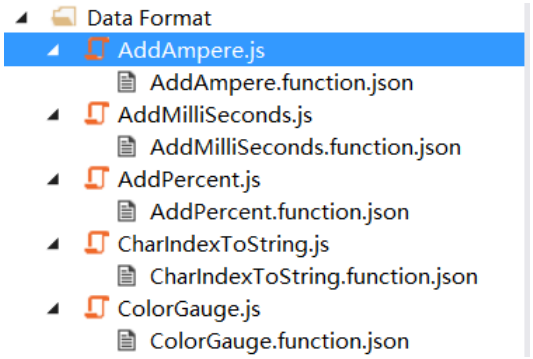
(6) 与 ST 写法不同的几个常用运算符

求余	%	指数运算	**
不等于	!=	逻辑与	&&
自累加	x++，等效于 X:=X+1	逻辑或	||
自递减	x--，等效于 X:=X-1	逻辑非	!

5.1.3 TC3 HMI Function 使用的 JavaScript

参考示例 02_TC_Example_EL9227.tnzip，其中只用 JavaScript 来创建 Funtion，用到的语法很少。

(1) 典形的 Function



JSON 文件

在可视化界面中编辑参数更加方便，虽然 TC3 HMI 也提供了它的代码。在代码或者可视化界面中的修改结果是等效的。

(2) Function 的 js 文件基本格式

```
(function (TcHmi) {

    var AddAmpere = function (current) {
        return current.toString().concat(' A');
    };

    TcHmi.Functions.registerFunction('AddAmpere', AddAmpere);
})(TcHmi);
```

这段代码中，只要定义好 Funtion 的名称 AddAmpere（新建对象时命名）后，除了黄色部分外，都是自动生成的。而参数 current 等，在自动生成的子文件 Name.function.json 中定义。并且增减参数后，Java Script 中的 function(　)中的参数也会自动更新。

```
(function (TcHmi) {

    var MySum = function (X,Y,Z) {
        Return (X+Y).toFixed(1).Concat('Good')
    };

    TcHmi.Functions.registerFunction('MySum', MySum);
    })(TcHmi);
```

(3) **固定组成部分**

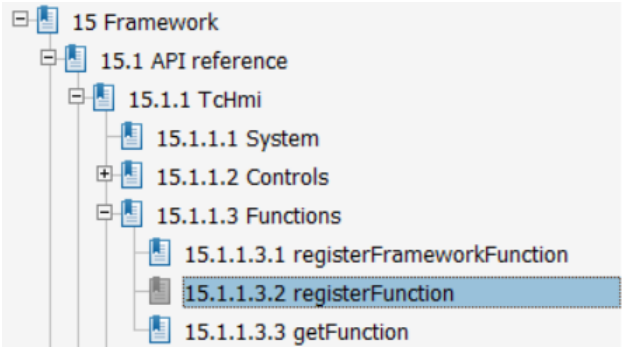
函数声明

```
(function (TcHmi) {
})(TcHmi);
```

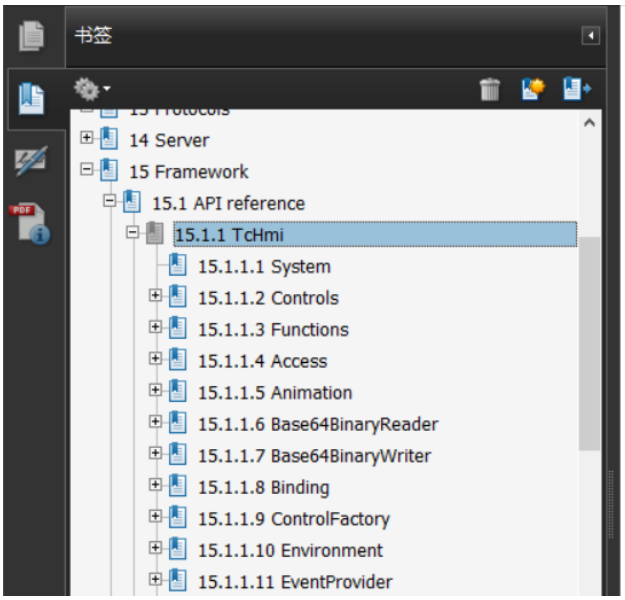
函数注册

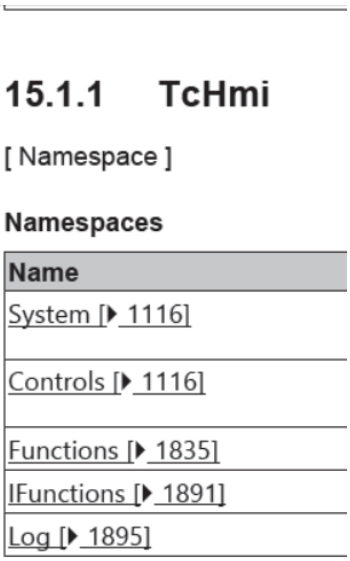
```
TcHmi.Functions.registerFunction('MySum', MySum);
```

其中，TcHmi 是个固定的指代，而.Functions 按理并行的还有很多子元素，但是在 JavaSritpt 编辑器中没有 Smart Coding 功能，所以只能有样学样。同理，.regesterFuntion 也是这样。在手册中找到了 TcHmi 下的内容，比如 `TcHmi.Functions.registerFunction`



在手册中查看 TcHmi，完整的内容列表如下：





15.1.1 TcHmi

[Namespace]

Namespaces

Name
System [▶ 1116]
Controls [▶ 1116]
Functions [▶ 1835]
IFunctions [▶ 1891]
Log [▶ 1895]

其中包括：

NameSpace：比如 Controls，Functions，iFunctions，Log

Classs:　比如 Server，Symbol，Theme，View，

Functions：

Enumerations：

Interfaces：比如 Color，FontStyle，

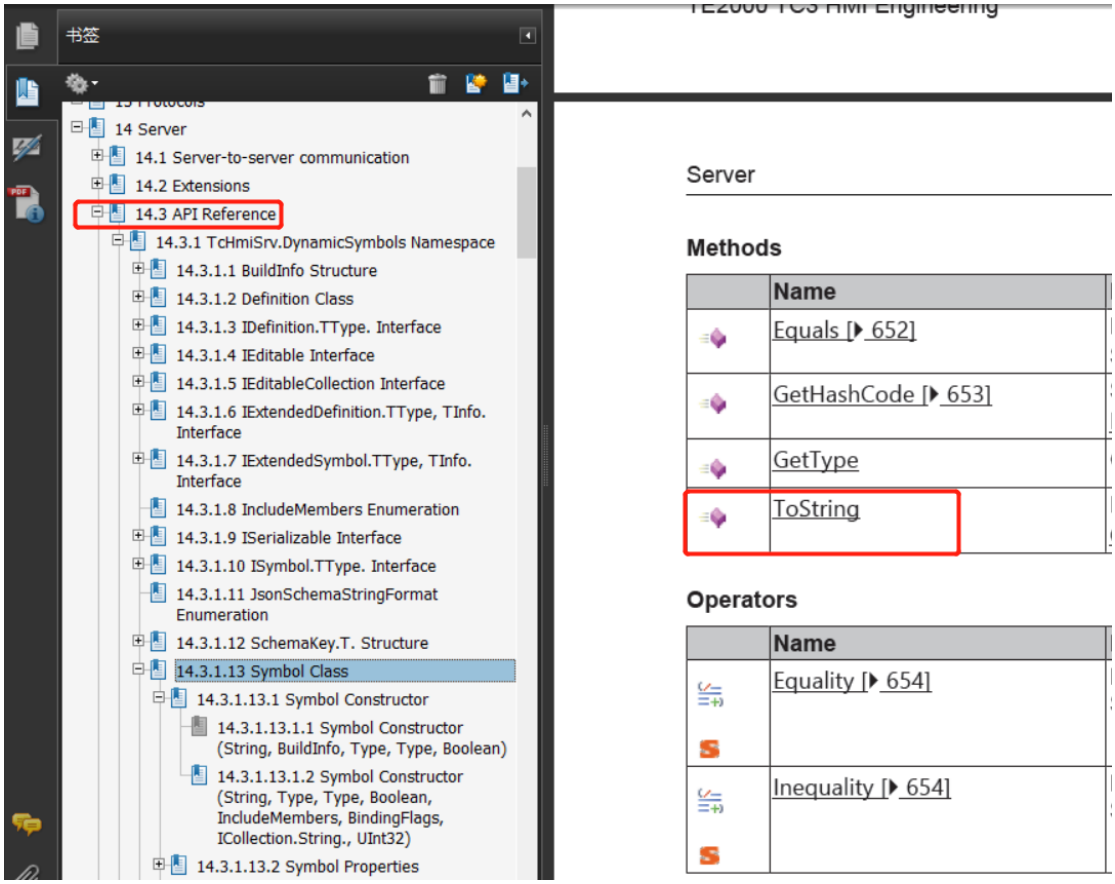
这些都是 HMI 开发才会用到的特别对象，可以用 JavaScript 来访问，而在写 PLC 程序的时候是完全不会涉及这些内容的。比如要表达一个颜色，PLC 里面用一个长整数就可以表达，而在 HMI 里要表达颜色就要用专门的 Interface：Color。

(4) 示例中用到的语法

ToString()和 Concat

AddAmpere	return current.toString().concat(' A');
AddMilliSeconds	return number.toString().concat(' ms');
AddPercent	return number.toString().concat(' %');

在许多 Class 的 Method 中，都包括了 ToString。在 TC HMI 手册中搜索可得，例如：



字符串的连接运算，不是 TcHmi 定义的，而是 JS 的语法，在“JAVA-SCRIPT 入门学习.pdf”第 7 页有写。另外，关于 toString()，在该书中也有提到。

ToFixed(1)

MilliVoltToVolt	return (voltage/10).toFixed(1).concat(' V');

ToFixed 这个函数在 TE2000 的手册中没有，在“JAVA-SCRIPT 入门学习.pdf”中也没有。在 JAVA 学习网站上找到 ToFixed 的用法描述：https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/Number/toFixed：

Switch……Case 和 Symbol.ReadEx

CharIndexToString	switch(charIndex) {
	case 1:
	return TcHmi.Symbol.readEx('%l%charStandard%/l%');
	case 2:
	return TcHmi.Symbol.readEx('%l%charTypeF%/l%');
	case 3:
	return TcHmi.Symbol.readEx('%l%charTypeT%/l%');
	case 4:
	return TcHmi.Symbol.readEx('%l%charManual%/l%');
	default:
	return 'Unknown';
	}

OpModeIndexToString	switch(index) {
	case 0:
	return TcHmi.Symbol.readEx('%l%opModeStandAlone%/l%');
	case 1:
	return TcHmi.Symbol.readEx('%l%opModeEtherCat%/l%');
	default:
	return 'Unknown';
	}

Switch 类似 PLC 的 ST 语言中的 CASE 语句，是 JS 的标准语法，不用解释。

Symbol.readEx 则可以从 TE2000 手册中找到这个方法：

15.1.1.18 Symbol

[Class]

Access to symbol information.

Constructors

Name	Description
constructor [▶ 2037]	Creates an instance of the symbol.

Functions

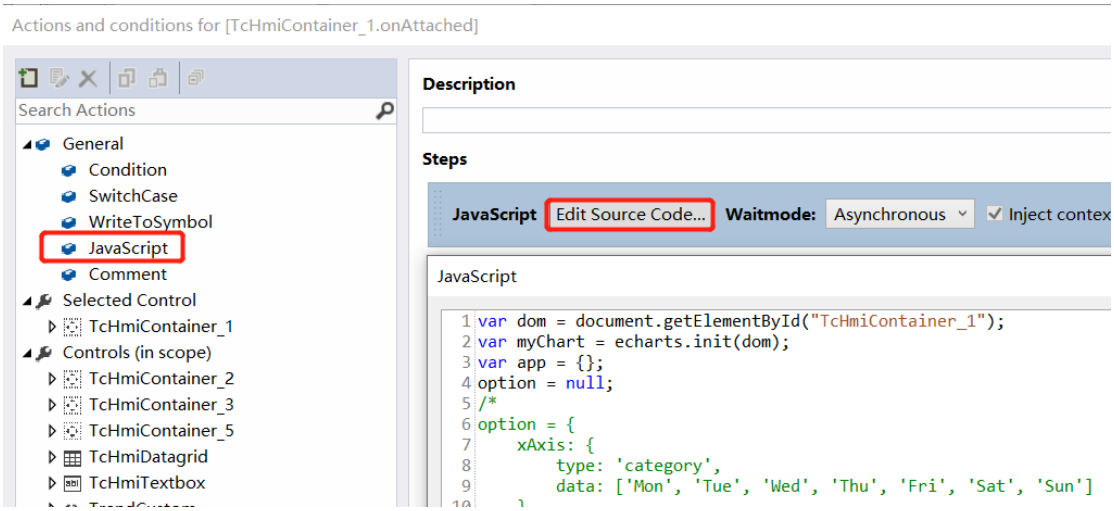
Name	Description
read [▶ 2037]	Reads the value of the symbol.
readEx [▶ 2038]	Reads the value of the symbol.
write [▶ 2039]	Writes the value of the symbol.
watch [▶ 2040]	Monitors the symbol for changes.
exists [▶ 2041]	Checks if the symbol exists.
resolveSchema [▶ 2041]	Resolves the schema of the symbol and returns it using the callback function.
getExpression [▶ 2042]	Returns the TcHmi.SymbolExpression object associated with the symbol.
destroy [▶ 2042]	Destroys the Object symbol.

Gauge 的 Range 和 Color（可选）

ColorGauge var ColorGauge = function (current,warningLvl)	return '[{ "color": { "color": "rgba(0, 151, 59, 1)" }, "start": 0.0, "end":' + warningLvl*current/100 + '}, { "color": { "color": "rgba(220, 131, 0, 1)" }, "start":' + warningLvl*current/100 + ', "end":' + 1.1*current + ' }, { "color": { "color": "rgba(230, 0, 0, 1)" }, "start": '+1.1*current+' , "end": '+current*1.5+' }]';

这个函数返回的类型比较特别，需要断句来看。实际上，它返回的是一个数组，共 3 个元素，每个元素又是一个结构体，包括 Color，Start 和 End。至于为什么要这样写，为什么时 Color 而不是 Colour，这应该不是英式英语和美式英语的差别，而是在 Gauge 这个控件里确定的参数名称。用（"color": "rgba(0, 151, 59, 1)"）表示一个颜色，是 TcHMI 基础框架的约定。

5.1.4 TC3 HMI 的控件 Event 中执行的嵌入式脚本



写在这里的 Code，实际上是位于控件所在的 Contend 或者 View 的代码中，把这部分代码 Copy 到一个独立的 Funtion 中，本质上没有区别。

5.2 Json 学习

5.2.1 学习资料

https://www.w3school.com.cn/json/index.asp

5.2.2 什么是 JSON

JSON(JavaScript Object Notation) 是一种表达对象的文本格式。根据百度百科：**JSON 是 JS 对象的字符串表示法，它使用文本表示一个 JS 对象的信息，本质是一个字符串。**

JSON 简单的语法格式和清晰的层次结构明显要比 XML 容易阅读，并且在数据交换方面，由于 JSON 所使用的字符要比 XML 少得多，可以大大节约传输数据所占用的带宽。所以 IoT 通讯中与云端交换数据也采用 json 格式。

5.2.3 Json 字符串的构成

要表达中国几个省份包括的城市，Json 字符串表达为：

```
{
    "name": "中国",
    "province": [
{   "name": "黑龙江",
            "cities": {"city": ["哈尔滨", "大庆"] }
        },

{   "name": "广东",
            "cities": {"city": ["广州", "深圳", "珠海"] }
        },

{   "name": "台湾",
            "cities": { "city": ["台北", "高雄"] }
        },

{   "name": "新疆",
            "cities": { "city": ["乌鲁木齐"] }
        }
]
}
```

整个字符串只包含几个关键字符

{ }	一个对象要用{ }括起来
[ ]	如果值是数组，所有元素要用[ ]括起来
:	键名和键值之间用：分隔
,	数组的各元素之间用，分隔
“ ”	键名和键值，都用“ ”括起来

5.2.4 从 Json 字符串读入到 Java Script 对象

```
var obj = {a: 'Hello', b: 'World'};          //这是一个对象，对应 PLC 就是结构体类型的变量，结构体包含两个元素 a 和 b，其值分别为字符串 Hello 和字符串 Word
var json = '{"a": "Hello", "b": "World"}';   //这是一个字符串，并且符合 json 规则
```

要实现从 JSON 字符串转换为 JS 对象，使用 JSON.parse() 方法：

```
var obj = JSON.parse('{"a": "Hello", "b": "World"}'); //结果是 {a: 'Hello', b: 'World'}
```

要实现从 JS 对象转换为 JSON 字符串，使用 JSON.stringify() 方法：

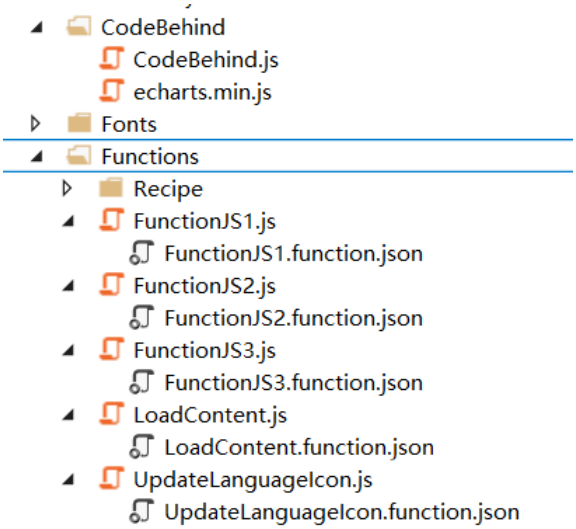
```
var json = JSON.stringify({a: 'Hello', b: 'World'}); //结果是 '{"a": "Hello", "b": "World"}'
```

可见在 json 字符串中，变量名，变量值，都用双引号括起来了。

5.2.5 Tc 3 HMI 的 Function 配套.json 文件

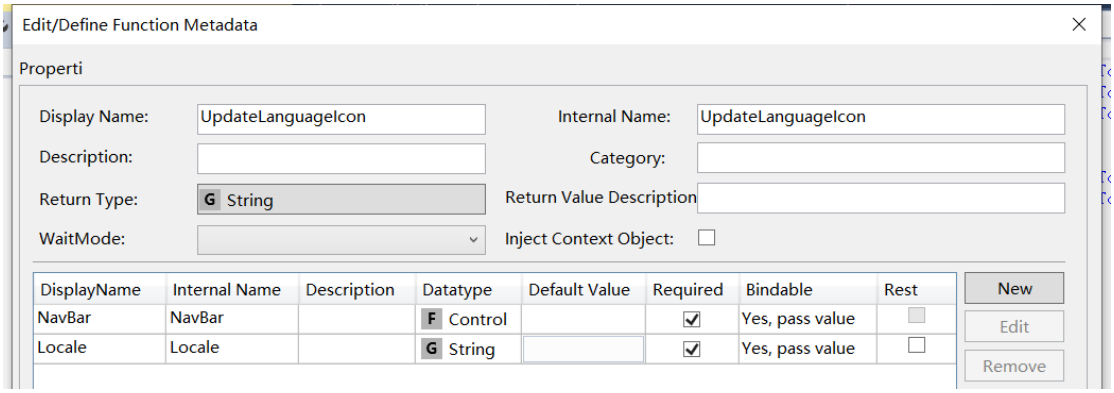
(1) TC 3 HMI 中的 Json 文件

TC HMI 中的 Json 文件是新建对象的时候自动生成的。比如创建一个 Funtion(JS)，系统就会自动创建同名的 Json 描述文件。但是 CodeBehind 的.js 文件就不会配套生成 Json 文件：



(2) TC HMI 提供比字符串更友好的可视化界面

如果用记事本打开 Function 的 json 文件，可以看到它包括一个总的描述，和若干个 Argument，每个 Argument 有相同的属性需要填写。了解 Json 格式当然有助于裸眼阅读 Json 字符串，但 Tc HMI 中提供更加友好的可视化界面：



用户甚至可以忽略 Json 格式，直接在这里编辑变量，后面就动自动生成相应的符合 Json 格式的字符串。

5.3 HTML 学习资料

5.3.1 HTML 是用来描述网页的一种语言

HTML 指的是超文本标记语言 (Hyper Text Markup Language)，它不是一种编程语言，而是一种标记语言 (markup language)，它有一整套标记标签 (markup tag)，HMTL 文件就使用这些标记标签来描述网页。

HTML 文档 = 网页

HTML 文档描述网页，它包含 HTML 标签和纯文本。Web 浏览器的作用是读取 HTML 文档，并以网页的形式显示出它们。浏览器不会显示 HTML 标签，而是使用标签来解释页面的内容。

用记事本在写几行 HTML 语句，另存为.html 文件，就可以用浏览器打开它演示网页效果，非常简单。

5.3.2 HTML 标签

HTML 标签是由尖括号包围的关键词，通常是成对出现的，比如 和 ，比如 <html>和</html>。带“/”的是结束标签，要放在后面。

常用标签

<html>……</html>

<head>……</head>

<body>……</body>

…… ：段落

<h1>……</h1>：一级标题，同理 h2 就是二级标题，最多到 h6

<a>……：链接

……</br>：换行

复杂一点的标签： …… ，这是布局，可以嵌套，TC3 HMI 中用得非常多。

当然还有许多其它标签，可以表达复杂的格式，但是 TC3 用户只要知道 html 的原理、简单用法、基本规则就可以了。如果碰到了之前不熟悉的标签，再到网上去查：

https://www.w3school.com.cn/html5/index.asp

HTML 和 HTML5

后者只是增加了一些新的标签类型，可以更好更简洁地表达一些流行的对象，比如更好的音乐、视频播放。TC3 HMI 支持 HTML5。

5.3.3 Tc3 HMI 中的 HTML

初学者不需要自己编辑 HTML 文件，网页都是 Tc3 HMI 开发环境根据用户选择的控件、设置的参数在后台自己生成的。

用户可以通过 Iframe 并指定其 Source（选项 Common | Src）链接到某个 HTML 文件来实现在 Tc3 HMI 的画面框架内的指定区域显示网页。例如 3.1 节中介绍的的导入一个网页版游戏。

高阶用户可以用这种方式实现自由的网页设计。

5.4 CSS 学习

5.4.1 学习资料

https://www.w3school.com.cn/cssref/index.asp

5.4.2 什么是 CSS

根据百度百科，CSS 指层叠样式表(英文全称：Cascading Style Sheets)。是一种用来表现 HTML 或 XML 等文件样式的计算机语言。

CSS 定义样式结构，如字体、颜色、位置等，用于描述网页上的信息格式化和实现的方式。CSS 样式可以直接存储于 HTML 网页或者单独一个带有文件扩展名.css 的“样式单”文件。

CSS 不仅可以静态地修饰网页，还可以配合各种脚本语言动态地对网页各元素进行格式化。修改 CSS 就可以更新与之相关的所有页面元素。所以，CSS 的规范是 HTML 规范的一个子集。即使不用 CSS，HTML 也能把需要表达的格式表达出来，只是有了 CSS 表达起来更方便，类似“宏”或者“子程序”的作用。

HTML 中的 CSS

```
<head>
<style type="text/css">
body {background-color: red}
p {margin-left: 20px}
</style>
</head>
```

独立的 CSS 文件

```
head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

具体的格式，就在 mystyle.css 中定义。

5.4.3 TC3 HMI 中的 CSS 文件

(1) 项目下的.css 文件

项目自有的.css 文件只有两个：

TcHmiProject1\bin\Themes\Base\ Style.css （TC3 Build 4022.4 项目有但低版本的没有）

TcHmiProject1\bin\Fonts\ Fonts.css （高低版本都有）

其余.css 文件都在“\TcHmiProject1\bin\Libraries\Controls\”的“\Beckhoff 和\System”下每个子文件夹表示一个控件，每个控件都包含一个 Style.css

示例中通常都没有修改 CSS 的格式，相关 CSS 的代码都是自动生成的。

(2) 模板自带.css 文件的内容

Style.css

这个模板自带的 Style.css 文件中只对字体进行了定义。如果用户需要更丰富的格式显示，应该在相应主题下去修改 Style.css 文件即可。

```
/**
    关于 Tc HMI 的字体包容性
    我们默认引用 Roboto Condensed（在文件夹 Fonts 中）作为网页字体，如果客户端的系统语言为于拉丁文、希腊文、越南文、西里尔等，将使用该字体恢复网页显示。
    对于 Roboto 不能还原的象形文字，我们则使用预装的备用语言。
    对于中文字符，各操作系统使用的恢复显示字体为：微软雅黑（Windows 操作系统），Hiragino Sans GB（苹果操作系统），'Noto Sans CJK SC/TC' (安卓操作系统)，'WenQuanYi Micro Hei' (Linux 操作系统)
    对于日文字符，各操作系统使用的恢复字体为：'Meiryo' (Windows 操作系统), 'Hiragino Kaku Gothic Pro' (苹果操作系统), 'Noto Sans CJK JP' (安卓操作系统)
    如果要使用其它字体，你可以把 Woff 文件添加到 TcHmi 项目并自行添加一个 @font-face 的 Section（节）
    谷歌的 noto 家族尝试覆盖所有应用，该功能可以免费使用，并且与 Roboto 匹配良好。
    请参考：https：//www.google.com//get///noto/and https：//fonts.google.com//earlyaccess

    字体的使用在当前活动主题的主 CSS 文件中定义，例如 Themes/Base/Style.css）。*/

html {
    font-family:RobotoCondensed,
        'Microsoft YaHei', '微软雅黑',
        'Hiragino Sans GB', '冬青黑体',
        'STXihei', '华文细黑',
        'WenQuanYi Micro Hei',
        'Meiryo', 'メイリオ',
        'Hiragino Kaku Gothic Pro', 'ヒラギノ角ゴ ProN',
        'Noto Sans CJK SC', 'Noto Sans CJK TC', 'Noto Sans CJK JP', 'Noto Sans CJK KR',
        sans-serif
        ;
    font-size: 12px;
    font-style: normal;
    font-weight: normal;
}
```

Fonts.css

```
@font-face {
    font-family: RobotoCondensed;
    src: url(Roboto-Condensed-webfont.woff);
}

/**
    关于 Tc HMI 的字体包容性
    我们默认引用 Roboto Condensed（在文件夹 Fonts 中）作为网页字体，如果客户端的系统语言为于拉丁文、希腊文、越南文、西里尔等，将使用该字体恢复网页显示。
    对于 Roboto 不能还原的象形文字，我们则使用预装的备用语言。
    对于中文字符，各操作系统使用的恢复显示字体为：微软雅黑（Windows 操作系统），Hiragino Sans GB（苹果操作系统），'Noto Sans CJK SC/TC' (安卓操作系统)，'WenQuanYi Micro Hei' (Linux 操作系统)
    对于日文字符，各操作系统使用的恢复字体为：'Meiryo' (Windows 操作系统), 'Hiragino Kaku Gothic Pro' (苹果操作系统), 'Noto Sans CJK JP' (安卓操作系统)
    如果要使用其它字体，你可以把 Woff 文件添加到 TcHmi 项目并自行添加一个 @font-face 的 Section（节）
    谷歌的 noto 家族尝试覆盖所有应用，该功能可以免费使用，并且与 Roboto 匹配良好。
    请参考：https：//www.google.com//get///noto/and https：//fonts.google.com//earlyaccess

    字体的使用在当前活动主题的主 CSS 文件中定义，例如 Themes/Base/Style.css）。*/
```