第七章 易控工程中的编程—用户程序

培训目的

1. 了解易控的用户程序的概念和特点
2. 掌握易控的用户程序的开发方法
3. 掌握易控用户程序的组织
4. 掌握在用户程序中灵活调用易控的对象和命令
5. 了解如何在易控的用户程序中调用使用用户已经开发完成的代码和第三方程序代码

第一节 概述

用户程序是易控提供的一种在工程中编写程序的功能。通过在工程中编写用户程序代码，可实现对工程中各种资源的灵活控制，改变工程的执行流程、使用第三方已经开发出来的程序代码和编写自己的特需功能。用户程序采用了开放的C#高级语言作为编程语言。用户程序可编译执行，运行速度快，功能强，而且可视化编程和智能感知等编程技术的采用极大简化了用户程序的编写难度。

一、组态软件的脚本程序

组态软件作为通用的监控系统二次开发平台，在各行各业都有广泛应用，因此最终用户的监控需求千变万化，他们需要组态软件含有强有力的工具或方法来实现自己的特殊需求，在组态软件中可以通过编写自己的程序来实现自己的特殊功能和流程控制，组态软件是通过提供“脚本”程序的功能来满足这一需求的。“脚本”是组态软件提供了一种简单的脚本语言，用户可以通过该语言编写脚本程序，来调用组态软件中的一些功能指令和进行流程控制。用户编写的脚本程序嵌入在组态软件之中，由组态软件来解释执行，这极大地增强了组态软件的灵活性，使组态软件能够去适应不同行业、不同用户的需求，让用户可以按照自己的意愿来编写自己的逻辑和控制流程。 可以说“脚本”是组态软件最重要的功能之一。

脚本程序所使用的编程语言，开始是组态软件厂商自己定义的语言，其语法类似 BASIC 语言或 C 语言，一般称为“类似 BASIC”语言或者“类似 C 语言”的脚本语言。由于这些脚本语言的封闭性和功能局限性，随着时间的推移，一些组态软件采用了通用的脚本语言作为自己的脚本程序语言，比如 VBA，JavaScript

等，这些通用脚本语言比自定义脚本语言提供了更多的功能和更灵活的程序控制。

随着信息技术的发展，用户对组态软件“脚本程序”的要求越来越高，不管是用自定义脚本语言还是通用脚本语言所编写的脚本程序，其能实现的功能也越来越显示出其局限性。主要体现在脚本语言的简易性，指令和功能的局限性，不能很好利用用户自己或第三方现成的代码，而且解释性的脚本程序的执行效率低下。正是为了解决这些问题，易控推出了自己的“用户程序”，作为“脚本程序”的换代升级。

二、易控的用户程序

在易控中，没有了“脚本程序”，取而代之的是“用户程序”。易控的“用户程序”是传统组态软件中“脚本程序”的扩充扩展。

易控直接采用了 Microsoft 最新专为.NET 平台开发而设计的高级语言 C#（CSharp）作为自己的“脚本”程序（用户程序）的编程语言，其功能较传统的自定义脚本语言有革命性的提升。C#语言的语法风格源自 C/C++家族，融合了 Visual Basic 的快速程序开发特性和 C/C++的强大功能，具有优雅、简单、安全、性能高、面向对象等诸多优点。易控的“脚本”程序因此功能十分强大。在用户程序中，可以直接使用 C#语言的丰富指令、.NET Framework 框架平台提供的数以万计的类库、易控提供的各种功能指令、易控工程中的变量、图形等对象。另外 C#本身是全开放的高级语言，可以方便地利用用户自己或者第三方已经编写好的程序代码，让这些功能代码无缝集成和嵌入到易控中运行，这极大丰富和扩充了组态软件的功能。

易控的“用户程序”和传统的“脚本程序”不同，不再是解释执行的，而是编译成了计算机可以直接执行的二进制代码。这样“用户程序”比“脚本程序”运行的效率更高。这已经超越了“脚本语言”的定义，这也是易控不再继续使用“脚本程序”概念的原因。实际上，用户可以在易控中编写自己真正的计算机程序。将组态软件的功能推向了一个前所未有的新高度。

易控的用户程序功能虽然强大，但掌握和使用却并不困难。用户可能在不知不觉中就已经开发出了功能强大的用户程序片段了。因为易控既考虑到了一些高级语言编程高手希望自由发挥和施展的需要，也考虑到了没有什么高级语言编程经验的工程技术人员的实际情况，让他们都能够得心应手地各取所需，完成自己的工程任务。

易控提供了可视化、图形化的编程模式，使得不需要记忆指令，只要按照简单的逻辑规则就可以开发用户程序，还提供了智能感知、自动代码填充、着色、关键字窗口、命令窗口、工程对象窗口、语法检查等各种手段极大简化了开发用户程序的难度，用户可以更多关注自己工程所要解决的问题，而不是编程语言本

身。

另外，易控还在工程中列出了用户最可能需要编写用户程序的地方，按照这些程序执行的触发条件进行了清晰的组织，例如当某一个变量改变的时候、当某一画面打开的时候等等。 简化了对用户程序的管理。

总之，用户程序功能虽然强大，但开发使用却并不困难。

三、 用户程序 VS 脚本程序

和传统的“脚本程序”相比，易控的“用户程序”具有明显的优势：

- 功能大幅增强：C# 高级语言能更灵活实现脚本语言所不能实现的功能
- 执行效率大幅提高：易控的用户程序是在工程运行前编译成可执行代码的，而脚本程序是工程运行时逐条解释执行的。
- 开放性大幅增强：C# 是完全开放的国际标准语言，用户无需专门学习一门语言。
- 集成能力大幅增强：用户程序中可以直接使用用户以前开发的.NET 模块（DLL），可以使用第三方软件供应商提供的成熟代码（DLL）。
- 用户程序是面向对象的，而脚本程序是面向过程的。

第二节 用户程序组织管理

用户程序是易控工程中不可缺或的重要功能。通过用户程序，能对工程的各个部分进行有效的衔接、组织和管理，使工程变得更加灵活和高效，能够实现复杂的动画效果，对数据管理、设备通信管理、逻辑控制、工程安全控制、工程语言切换等等的程序化管理，还可以直接使用易控之外的任何.NET 程序代码，极大增强了易控工程的实际功能。

在实际工程中，用户需要根据自己的工程要求选择在哪些地方使用用户程序，因此工程中的用户程序不是都编写在一起的一大堆代码，而是分散在工程不同部位的程序片断。用户决定编写哪些程序（片断），不需要哪些程序（片断），以及这些程序片断的内容。所有这些程序片断构成了工程的用户程序。如果不对这些程序片断进行有效的组织和管理，工程的开发将会十分混乱，因此需要对用户程序（片断）进行有条理地组织。

易控将工程中经常需要使用用户程序（片断）的地方，按照这些用户程序执行的触发条件分门别类罗列出来了，对用户程序进行了组织划分和管理，方便了用户开发工程的用户程序。

易控将工程中常用的用户程序（片断）分为以下一些类型：工程程序、画面

程序、变量改变程序、条件程序、热键、画面上图形对象的操作事件程序等。除画面上图形对象的操作事件程序外，其它类型的用户程序（片断）都组织在开发环境工程树的“用户程序”节点之下，如下图所示。

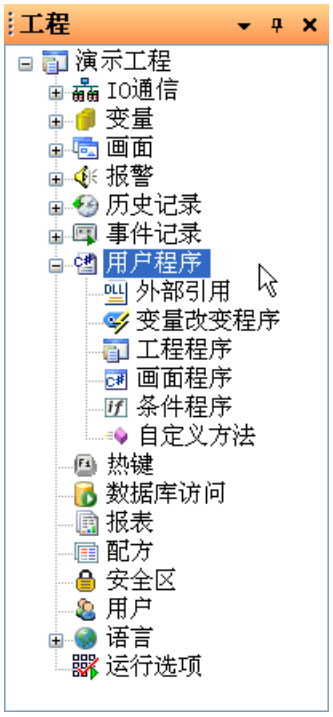

图 7.1 工程树中“用户程序”的组织

一、工程程序

是指在工程启动或者退出的时候，或者在工程运行期间需要周期性执行的程序代码。

二、画面程序

是指某一指定画面在打开或者关闭的时候，或者画面打开以后需要周期性执行的程序代码。

三、变量改变程序

是工程中的某一变量发生变化，就立即执行的程序代码。

四、条件程序

则具有更为广泛的涵义，表明工程中当满足某一条件的时候，需要执行的程序代码。条件是用户随意指定的，它是以工程中的变量为基础的逻辑表达式，其结果为真即是某一条件得到满足，否则为不满足。用户还可以指定程序代码是在条件满足的时刻执行一次，还是在不满足的时刻执行一次，还是在条件从满足到不满足，从不满足到满足时各执行一次，还是在条件满足后周期性执行，直到条件不满足或反之。

五、热键程序

热键即键盘的按键或按键组合，在工程运行的任何时候按下时都执行预先定义好的用户程序。

热键用于重新定义键盘的功能。用户可以指定在工程运行期间，某一键盘按键对应的用户程序、执行方式和所在的安全区。

六、画面上图形对象的操作事件程序

是指工程运行期间，用户使用鼠标点击或者拖动某一指定图形对象时，需要执行的程序代码。这一部分程序代码是和画面紧密相关的，没有列于易控开发环境的工程树下，而是在画面图形对象的“事件”属性窗口中配置的。

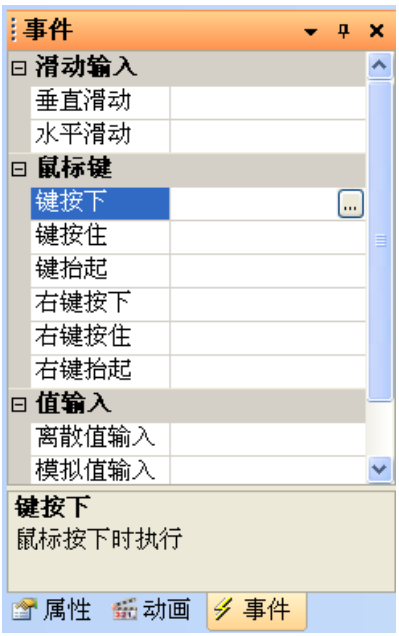

图 7.2 画面上图形对象的事件程序在选中对象的事件窗口中配置

通过将工程中的用户程序（片断）按照从工程角度上更易于理解的方式呈现和加以组织，方便了用户。

画面上的图形对象的操作事件程序是指工程运行期间，用户使用鼠标点击或者拖动某一指定图形对象时，需要执行的程序代码。这一部分程序代码是和画面紧密相关的，没有列于易控开发环境的工程树下，而是在画面图形对象的“事件”属性窗口中配置的。

第三节 用户程序开发

按照易控工程中对用户程序的分类和组织，编写各个用户程序片断的工作就是用户程序开发的过程。按照易控对用户程序的组织和管理，有两种方法开始编写用户程序代码。

对画面上图形对象的操作事件程序在画面开发期间进行配置，方法是在打开的画面中选中运行时发生操作事件的图形对象，在“事件”窗口中按照触发事件的类型分别进行配置。详细情况请参见“动画和事件”一章。

其它用户程序代码的开发方法都是类似的，从开发环境工程窗口下的“用户程序”处分别双击“工程程序”、“画面程序”、“变量改变程序”、“条件程序”或者位于“用户程序”之外的“热键”节点，会在工作区中打开相应的用户程序配置页面。这些用户程序配置页面的风格是类似的，以表格的方式来管理可能的多个用户程序片断。除“工程程序”的工作区配置页有固定的三行外，其它配置页中的用户程序片断个数（行数）是由用户自己建立的，默认为空。

用户程序配置页面的一行，对应一个用户程序片断。 每一个程序片断都具有“说明”、“程序”、“执行方式”、“时间间隔”等属性。其中“程序”一列是最重要的，即该程序（片断）的主体，点击该列栅格中的“…” 按钮，就弹出“用户程序编辑器”开始编写该程序片断的代码。 有关“用户程序编辑器”的内容，请参本章“用户程序编辑器”一节。

“执行方式”决定该用户程序是一次性执行还是多次周期性执行，若是一次性执行，执行的时刻，若是多次周期性执行，需要输入执行的间隔周期。（注：“工程程序”的三种执行方式在“名称”一栏（“启动时用户程序”、“运行期间用户程序”、“退出时用户程序”）中已经固定了，因此不需要“执行方式”外，其它的需要用户自己在“执行方式”列中选择）。

“时间间隔”一栏，默认为“N/A”即无效，当相应用户程序片断的执行方式选择为周期性执行时，在该栏输入时间间隔周期。如画面程序的“存在期间”程序、条件程序的“为真期间”程序，热键按住时的不断执行的程序等。

“热键”程序有“安全区”一列，即热键程序的执行有安全条件限制，只有

具有指定安全区权限的操作人员，才可以操作该热键。

可以在“说明”一栏中输入对该用户程序片断的描述信息，以方便工程的开发和管理。

下面简要说明各种用户程序的配置页面和配置要点，并配以图示，其使用方法比较简单，所以不作详细说明。

一、 工程程序

工程程序配置工作页如图 7.3 所示。

图 7.3 工程程序配置工作页

在“程序”栅格中编写用户程序代码。在“时间间隔（毫秒）”栅格中配置运行期间用户程序的时间间隔周期。“启动时用户程序”在工程启动执行时运行一次。“运行期间用户程序”在工程运行时，按照指定的时间间隔周期性执行。“退出时用户程序”在工程退出运行时运行一次。在“说明”一栏中输入对相应程序片断的说明文字，方便记忆和管理。

图 7.3 的例子中配置了工程启动时执行一次的用户程序。具体用户程序代码可通过选中“启动时用户程序”行的“程序”栅格再点击栅格中的“…” 按钮弹出的“用户程序代码编辑器”察看、输入和修改的。详见“用户程序编辑器”一节。

二、 画面程序

画面程序配置工作页如图 7.4 所示。

图 7.4 画面程序配置工作页

在“画面名称”栅格中选择一幅画面。在“程序”栅格中编写用户程序代码。“执行方式”的“打开时”是画面在第一次调入时执行一次的程序代码，“关闭时 ”是在画面关闭时执行一次的程序代码，“存在期间”是画面打开后关闭前按照指定间隔时间周期性执行的用户程序代码。在“说明”一栏中输入对相应程序片断的说明文字，方便记忆和管理。

图 7.4 的例子中配置了名为“总画面”的画面在打开时执行一次的用户程序，和该画面打开后每 10 秒钟执行一次的用户程序代码。具体用户程序代码可通过选中相应“程序”栅格再点击栅格中的“ ”按钮弹出的“用户程序代码编辑器”察看、输入和修改的。详见“用户程序编辑器”一节。

三、变量改变程序

变量改变程序配置工作页如图 7.5 所示。

图 7.5 变量改变程序配置工作页

在“变量名称”栅格中选择一个工程变量，其值发生变化时执行在“程序”栅格中配置的用户程序。在“说明”一栏中输入对相应程序片断的说明文字，方便记忆和管理。

图 7.5 的例子中，配置了当系统变量（SystemVariable）组中的日期（Day）变量改变时执行一次的用户程序，即当每天的凌晨即 Day 变量改变时，配置的用户程序会执行一次。具体用户程序代码可通过选中该变量所在行的“程序”栅格再点击栅格中的“…” 按钮弹出的“用户程序代码编辑器”察看、输入和修改的。详见“用户程序编辑器”一节。

四、条件程序

条件程序的配置工作页如图 7.6 所示。

图 7.6 条件程序配置工作页

在“条件表达式”栅格中输入一个条件，即由变量构成的逻辑表达式，在“程序”栅格中配置用户程序。在“执行方式”栅格中配置当条件满足以下五种情况之一时执行配置的用户程序。在“说明”一栏中输入对相应程序片断的说明文字，方便记忆和管理。

执行方式	含义
变真时	当条件变得满足的那一时刻执行一次所配用户程序
为真期间	当条件满足时，按照“时间间隔”栅格中的周期，反复执行所配用户程序
变假时	当条件不满足的那一时刻执行一次所配用户程序
为假期间	当条件不满足时，按照“时间间隔”栅格中的时间周期，反复执行所配用户程序
改变时	当条件变得满足或不满足的时刻都执行一次所配用户程序

图 7.6 的例子中配置了当变量 LightOn 为真，并且温度值大于 100 度时，每隔 5 秒钟执行一次的条件程序。具体用户程序代码可通过选中该条件行的“程序”栅格再点击栅格中的“…” 按钮弹出的“用户程序代码编辑器”察看、输入和修改的。详见“用户程序编辑器”一节。

五、 热键

热键是在工程运行时，在任何时候都有效的键盘按键。热键程序的配置工作页如图 7.7 所示。

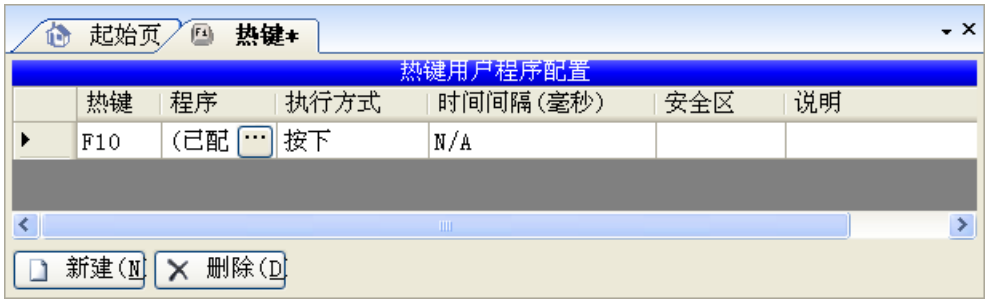

图 7.7　热键条件程序配置工作页

选中“热键”栅格，按下键盘按键，可以是组合键（注：有些特殊键不能作为热键使用）。易控将自动为你在“热键”栅格中输入代表该按键的字符。在“程序”栅格中配置用户程序。在“执行方式”栅格中选择热键程序执行的时机，如下所示：

执行方式	含义
按下	当热键按下的时刻执行一次所配用户程序
按住	当热键一直按住时，按照“时间间隔”栅格中的周期，反复执行所配用户程序
抬起	当热键按下后抬起的时刻执行一次所配用户程序

在“安全区”一栏中选择对该按键的安全控制，即只有事先分配有权限的用户按下该键时才有效，有关安全区的内容，请参考“工程安全”一章。

图 7.7 的例子中配置了当按下 F10 键时执行一次的用户程序。没有配置安全

区意味着任何操作员都可以按 F10 并执行配置好的用户程序。具体用户程序代码可通过选中 F10 键所在行的“程序”栅格再点击栅格中的“…” 按钮弹出的“用户程序代码编辑器”察看、输入和修改的。详见“用户程序编辑器”一节。

六、 用户程序代码

上面介绍了易控工程中常用的用户程序（片断）类型，但还没有涉及用户程序代码本身和编写过程，本小节将重点介绍易控用户程序的代码本身，下一小节将介绍这些代码的开发方法。

用户程序本体是用户需要执行的功能代码，是实现最终功能的重点。 和其它程序一样，用户程序数据、程序指令、执行某些特定功能的功能代码、程序执行流程的控制代码等构成。

易控的用户程序功能十分强大，在代码中可以直接使用易控工程中的变量等数据，直接调用易控提供的各种指令，C#语言的功能和指令，.NET Framework 框架平台数以万计的类库中的指令，以及第三方提供的类库中的指令。这样易控的用户程序可以实现十分复杂的功能。现分别作一些简单说明。

1. 访问易控系统所提供的指令和易控工程中的对象：

通过访问易控系统所提供的指令和易控工程中的对象，能对易控工程进行有效的控制。用户程序中可以使用的易控指令和可以访问的工程对象在用户程序编辑器的“命令”和“对象”窗口中罗列，请参考“用户程序编辑器”一节，通过使用易控的指令和工程对象，用户程序可以完成：

- 工程的退出、系统的关闭、机器重启、启动其它应用程序等；
- 操作人员的工程登录、退出，密码检查和修改、添加新的操作人员等；
- 画面的打开、关闭、打印等；
- 画面上的图形的属性改变，可实现特殊的动画效果；
- 操控工程中的变量，从而影响下位机和系统的运行；
- 操控设备的通信状况，如启停某些设备的通信；
- 切换工程的界面语言，以使得开发有多语言能力的工程以不同语言显示界面
- 响应报警、允许和禁止声音报警、修改报警的限值；
- 启动或停止历史数据的纪录；
- 报表的建立、打印和管理；
- 配方的操控；

- 外部数据库的操控；
- 其它；

下面是一些用户程序代码的示例，代码的作用在每一行代码的后面或者每一段代码片断的前面作了注释。（“//”号后面就是注释，这也是易控用户程序和 C# 语言所使用的注释方法）

```
// ================================================
// 注释：以下代码片断功能：
//       1）关闭工程的所有画面；
//       2）退出工程；
//       3）关闭计算机电源；
//================================================
Grp.CloseAll();         // 关闭所有画面
Project.Exit();         // 退出运行的工程
InSystem.PowerOff();    // 关闭计算机电源

// ================================================
// 注释：以下代码片断功能：
//       1）把支持英文的工程的显示界面切换为英文；
//       2）打开主画面；
//       3）打开设备变量组中的开关 1；
//       4）确认所有报警；
//       5）将一组变量预设值写入 PLC；
//       6）将一组生产过程的数据写入外部数据库
//       7）修改温度报警的高高报警设定值
//       8）停止记录历史数据
//================================================
ProjLanguage.SwitchLanguageTo("en-GB");  // 切换语言为英文
Grp.Open("主画面");        // 打开主画面
电机组.电机1 = true;       // 打开电机
Alarm.AckAllAlarm();     // 确认所有报警
Recipe.LoadRecipeValues("钢化玻璃", "送入阶段"); // 装载配方值
DbAccess.TagToCurrentRow("玻璃炉窑");      // 变量写入外部数据库
AlarmManager.窑炉.温度.HiHiLimit = 200;  // 修改报警限值
HistoryRecords.StopRecord();              // 停止记录历史数据
```

2. .NET Framework 框架类库的使用：

.NET Framework 框架类库包含数以万计的类，这些类功能丰富，是一个浩瀚的海洋，通过使用它们，和易控所提供的功能的结合，可以很完美实现大量高级的功能，无限扩展易控的能力。

下面一些代码片断，演示了其冰山之一角。

```
// ================================================
```

```
// 注释：以下代码片断功能：打开一个 Windows 选择文件对话框，
//       让用户选择一个声音文件，并记住将该声音文件的名字。
//=====================================================
System.Windows.Forms.OpenFileDialog dlg = new
    System.Windows.Forms.OpenFileDialog(); // 建立一个新对话框
dlg.CheckFileExists = true;                 // 检查文件的存在
dlg.Filter = "WAV files (*.wav)|*.wav";     // 只允许选择 .wav 文件
dlg.DefaultExt = ".wav";
if (dlg.ShowDialog() == DialogResult.OK)  // 显示对话框，选择文件
{
   功能演示.WaveFileName = dlg.FileName;    // 保存文件的名字
}
```

执行上述用户程序代码，弹出如下面图 7.8 所示的对话框。将用户选择的文件名称保存在工程的“功能演示”变量组中的一个字符变量 WaveFileName 中，在工程以后的其它地方就可以利用这个用户参与选择的文件。

由此可以看出，易控的用户程序和用户编写C#高级语言程序是几乎相同的，这样就在易控中嵌入了编写许多用户定制程序功能的能力。

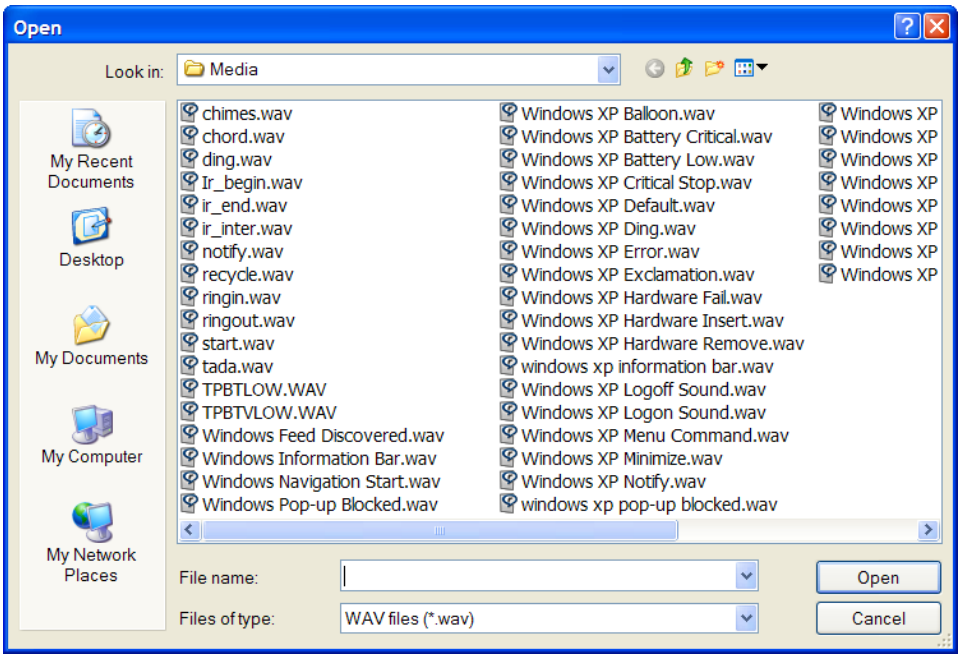

图 7.8 用户程序打开一个选择文件的对话框

3. 第三方类库的使用

.NET Framework 框架类库是微软提供的免费类库，用户也可以使用其它软件厂商所提供的各种各样的专业.NET 类库。

4. 用户自己编写的类库

和第三方类库一样，客户自己以前编写的一些符合.NET Framework 框架规范的类库（程序代码），也可以插入到易控的用户程序之中进行使用。

5. 过时动态库的使用

用户如果想利用一些以前编写的 Windows 动态库，也可以对它们进行一些.NET 包装从而在易控中可以使用。有关包装的内容是一些高级课题，请参考一些 C#的一些编程教材，在这里不再赘述。

七、C#用户程序编辑器

易控中所有用户程序代码的开发，包括画面上图形对象的操作事件的处理程序，都是在统一的“用户程序编辑器”中进行的。下面将介绍用户程序编辑器的内容。

用户程序编辑器是易控中开发编写用户程序代码的地方，凡是需要用户编写用户程序的地方，都会弹出这个统一的编辑器，在编辑器中显示、输入和修改代码。

用户程序编辑器是一个独立的窗口，包含自己的菜单、工具栏、状态栏、代码编辑区和其它子窗口。 这些子窗口包括：命令窗口、对象窗口、运算符窗口和关键字窗口，其目的是为了方便程序代码的开发。和易控开发环境一样，编辑器的子窗口停靠位置等都是可以调整的，默认的用户程序编辑器窗口外观如图 7.9 所示。

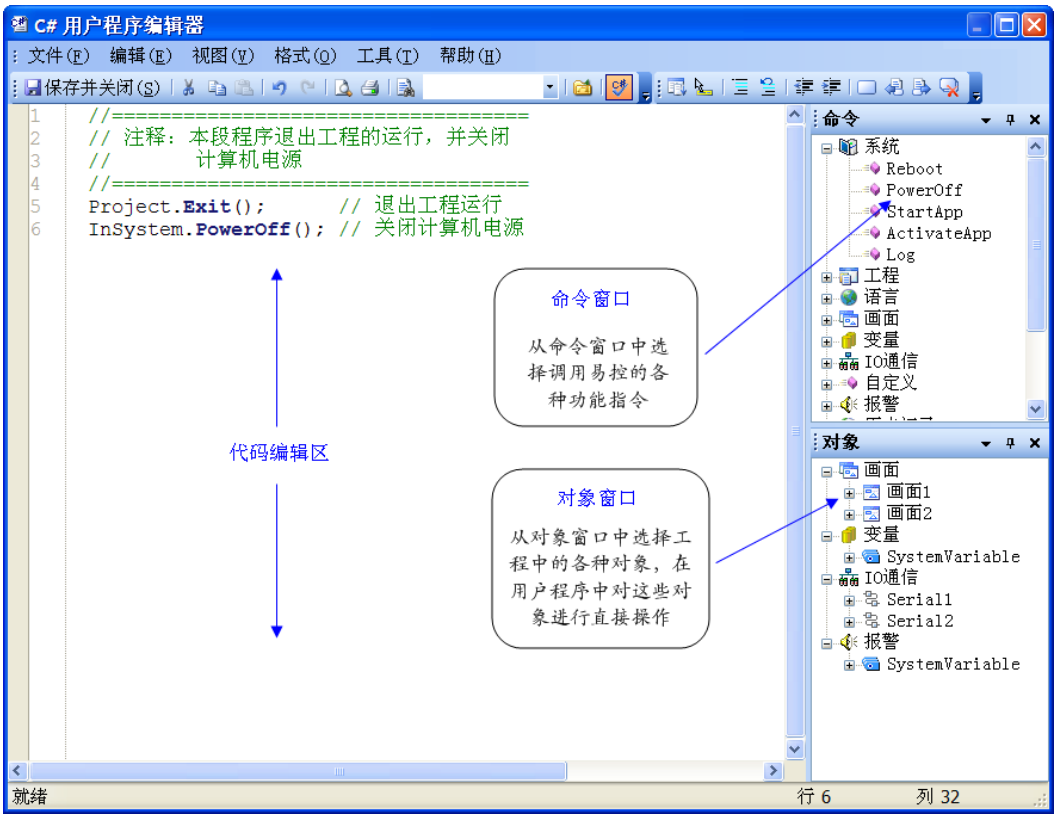

图 7.9 用户程序编辑器

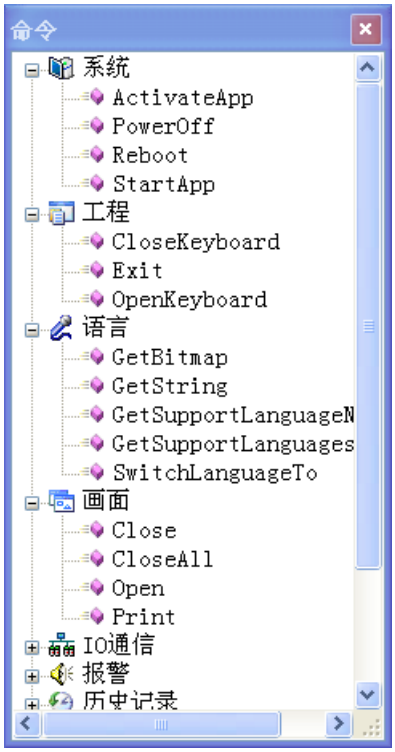

图 7.10 用户程序编辑器的命令窗口

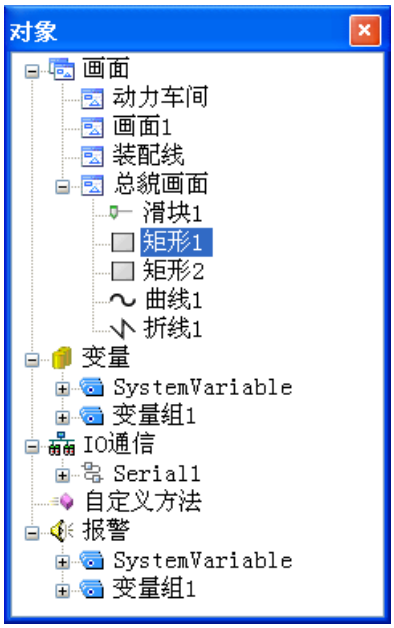

图 7.11　用户程序编辑器的对象窗口

用户程序编辑器中常用的有命令窗口和对象窗口。在图 7.9 中这两个窗口是展开的，其中的“运算符”和“关键字”窗口是折叠的。

命令窗口

在“命令”窗口里按照功能分类列出了可以在用户程序中直接调用的易控的各种功能指令，比如工程画面的打开和关闭，工程语言的切换，工程的退出、计算机的重启、关电等指令。可以使用的命令只和软件有关，和用户开发的工程无关，即易控的任何工程中命令窗口的内容是相同的。命令窗口的指令分类如下表所示：

指令分类	分类含义	分类对应的对象	指令使用举例
系统	和计算机系统操作有关的指令，如关机，重新启动等	InSystem	“InSystem.PowerOff();”指令将运行易控应用工程的计算机关机。
工程	和所运行的工程有关的指令，如退出工程等。	Project	“Project.Exit();”指令将退出运行的应用工程。计算机系统退出到操作系统状态。
语言	和工程应用所使用自	ProjLanguage	“ProjLanguage.SwitchLa

	然语言有关的指令，如切换工程的显示语言。		nguageTo("zh-CN");”指令将工程显示语言切换为健体中文；“ProjLanguage.SwitchLanguageTo("en-CA");”指令将工程的显示语言切换为加拿大英语。注：引号中的语言参数，不需要记忆，通过图形化编程的选择语言即可。
画面	和画面有关的指令，如打开和关闭画面等。	Grp	“Grp.Open("总貌画面");”指令打开和显示工程中的“总貌画面”。
通信	和设备通信有关的指令，如启停工程的对外通信。	DeviceServer	“DeviceServer.Stop();”指令可以暂停设备通信过程。
报警	和工程应用的报警处理有关的指令，如报警的确认等	Alarm	“Alarm.AckAllAlarm();”指令确认所有发生的报警
历史记录	和变量的历史记录有关的指令。	HistoryRecords	“HistoryRecords.StopRecord();”指令停止记录历史数据。
数据库访问	和操作外部数据库表格有关的指令	DbAccess	“DbAccess.CurrentRowToTag("数据库表 1");”指令将一个外部数据库表格中的当前记录中的各个字段的数据写入关联的工程变量中去，从而从外部数据库取得数据。
用户	和工程的安全控制有关的指令。如配置新的操作工，修改密码，登录等	User	“User.LogOn();”指令将提示用户输入名称和密码，确认正确后让其进入系统，否则拒绝。
配方	和配方有关的指令，如配方值的保存和加载等	Recipe	“Recipe.SaveRecipeValues("生产效率","高效");”指令能将应用（如生产

			线）工程中的一组与生产效率相关的变量的数值，保存到一个名为“生产效率”的配方中的“高效”配方值组。
报表	和报表有关的指令。如报表的定时打印等。	Report	“Report.SetupPage();”指令可以设置打印报表的纸张样式

注：由于易控的面向对象设计，任何指令都会依附一个对象来进行，因此在书写指令时，前面要写上负责执行该指令的对象。

每一个指令对象，如 InSystem，所支持的全部指令在这里并没有列出，在易控中使用时，有非常友好的提示，了解和掌握使用比较容易，也可以参阅软件的用户手册。

对象窗口

用户程序编辑器的对象窗口中按分类列出了可以在用户程序中直接使用的工程对象，这些对象是用户在进行工程组态时建立的，如画面上绘制的图形对象，定义的工程变量等，配置的通信设备，配置的报警信息、工程中添加的操作工等等。对象窗口的内容，除了分类是和易控软件本身相关外，具体的内容和开发的工程有关，随工程的不同而不同。对象窗口的分类如下：

对象分类	分类含义	分类对应的对象	使用举例
画面	包含工程中的所有画面，画面再包含绘制在其上的图形对象。	GrpManager	“GrpManager.总貌画面.矩形 1”是总貌画面上的名称为“矩形 1”的图形对象
变量	包含工程中的所有变量组。变量组再包含变量。	变量组名，如 SystemVariable 和其它用户自定义的变量组名称	“SystemVariable.Day”是工程中易控默认的系统变量组中的日期变量
IO 通信	包含工程中所有的通信通道，通道下面挂接通信设备。	DeviceIoManager	“DeviceIoManager.串口.FX”是连接在一个名为“串口”的串行通道下的一个名为”FX”的通信设备，如 PLC

报警	包含报警变量	AlarmManager	“AlarmManager.加热炉.压力”是工程中定义的一个报警变量，该变量是位于“加热炉”变量组中的“压力”变量

注：易控的开放性使用户有更大的灵活性和方便性，也可以实现复杂的功能，但全面开放工程中的对象有些时候可能对工程经验和编程经验不足的用户带来容易犯错误的机会，也因此只开放了部分工程对象。

运算符窗口和关键字窗口

运算符窗口包含了编写用户程序经常使用的运算符号，如算数运算的加、逻辑运算、索引和成员访问等等。关键字窗口包含了编写用户程序的各种流程控制关键字，如 if、else、foreach、switch、case、break 等等。

易控用户程序的编程语言选用的是微软的 C#高级语言，所以所有的运算符和关键字和开放的 C#相同。

用户程序的开发方法

用户程序代码编辑区是输入程序代码的地方。 完全手工输入代码编写程序对用户的要求是比较高的， 用户需要记住大量的关键字、运算符、功能调用指令、指令所需要的参数等，或者依赖厚厚的用户手册或在线帮助文档。这也是多数组态软件目前所使用的方法。 易控采用了图形化编程、智能感知、关键字着色、自动代码填充等一系列技术手段，让用户不需要记住大量的指令和使用参考手册就可以很容易地编写自己的用户程序（片段）。

- 双击命令窗口中的命令、对象窗口中的对象、关键字窗口中的关键字或者运算符窗口中运算符，将自动在编辑区中的当前光标处插入相应的代码。
- 需要输入指令时，在编辑器的指令窗口根据该指令的类型查找，需要引用工程中的对象时在对象窗口中根据对象的类型和名称查找。编程是可视化的。
- 双击需要参数的指令后，编辑器会弹出对话框让用户可视化地选择参数，确认后编辑器将图形化的选择结果变成程序代码插入到编辑区的代码中。参数的输入是图形化的。
- 在编辑区手动输入代码时，每输入一个字符，编辑器都会自动以下拉菜单的形式提示可用的代码，用上下箭头可以选择其中的代码，继续输入字符会逐渐缩小提示的可用代码范围，直至输入或找到完整的关键字或

者对象。这些可用提示是智能的，即智能感知。

- 在输入指令或对象时，输入名称时要从管理该对象的对象开始。 比如 GrpManager 对象是管理画面的对象，然后是具体的画面，再是画面上的图形对象。如引用“总貌画面”上的名为“矩形 1”的对象方法是输入“GrpManager.总貌画面.矩形 1”而不是直接使用“矩形 1”；关机指令的正确输入方法是“InSystem.PowerOff ()”而不是“PowerOff ()”。注：在易控中，并不需要记住“GrpManager”、“InSystem”等名称，在编辑器的对象或指令窗口中选择对象和指令后，系统会自动为你添加这些对象名称。
- 在对象的后面在输入“.”字符（句点字符）会弹出该对象支持的属性和方法（指令）列表，用上下箭头可以选择其中的属性或者方法（指令）。这种提示也是智能的。
- 代码的关键字、指令、参数、对象名称等在程序编辑区中是以不同颜色显示的，用户可以方便的进行检查，修改参数时非常方便，而不容易出错。

总之，易控采用的图形化、智能化等多种手段使用户不需要记忆大量的指令，极大降低了用户程序的编写难度，为初学者提供了极大的方便，为熟练用户提高了编写效率。下面列举的一些简单例子图示了简单的用户程序开发过程。

图 7.12 和图 7.13，演示了用户在不需要输入任何字符的情况下，编写了一条打开画面的用户程序。Grp.Open(“总貌画面”,“动力车间”,“装配线”)；该指令打开工程中的“总貌画面”、“动力车间”和“装配线”三幅画面。

图 7.12　第一步：双击指令窗口中画面分类下的“Open”

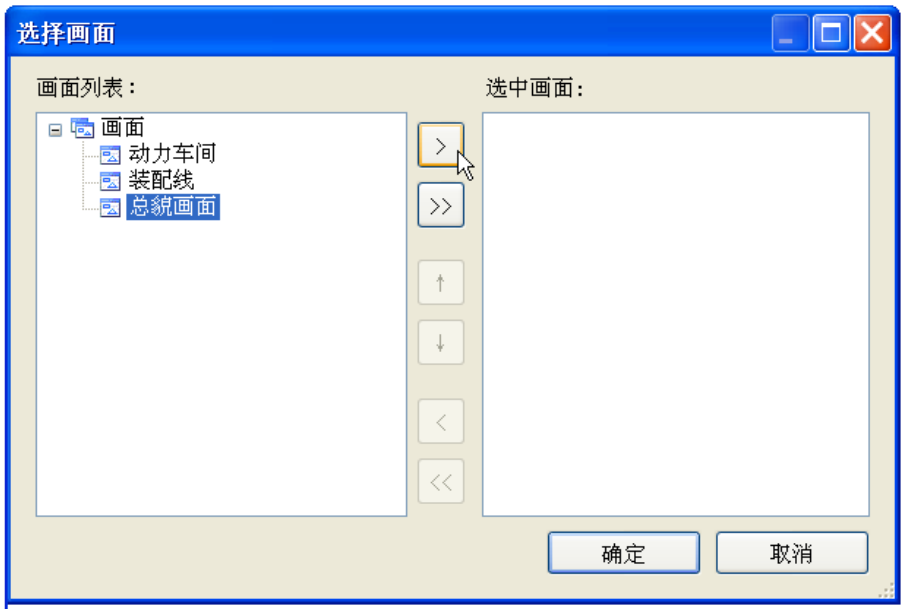

图 7.13 第二步：可视化选择要打开的画面

点击“确定”后，编辑区光标处自动添加如下代码：Grp.Open(“总貌画面”,“动力车间”,“装配线”)；

下面的图 7.14 演示了调用一个名为“总貌画面”的画面上的“矩形 1”的旋转角度的方法。

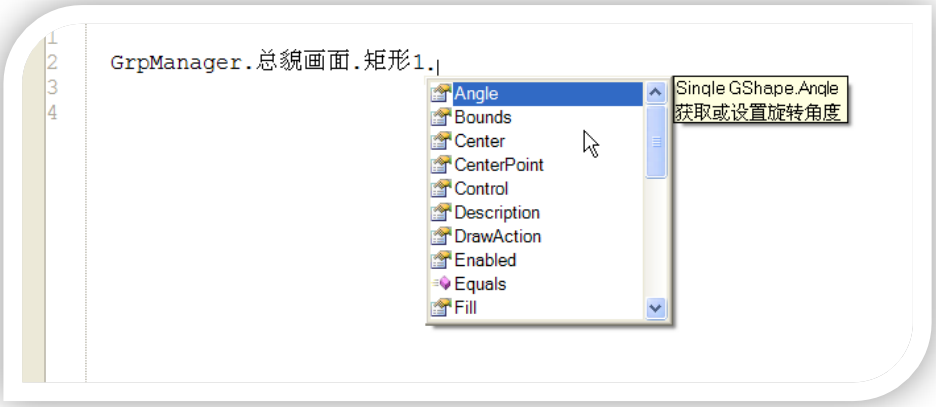

图 7.14 在对象的名称后输入“.”字符弹出对象可用的属性和方法

以下图示 7.15 到图示 7.20 演示了在输入 Grp.Open(“总貌画面”)；指令的过程中，智能感知所作的帮助性工作。

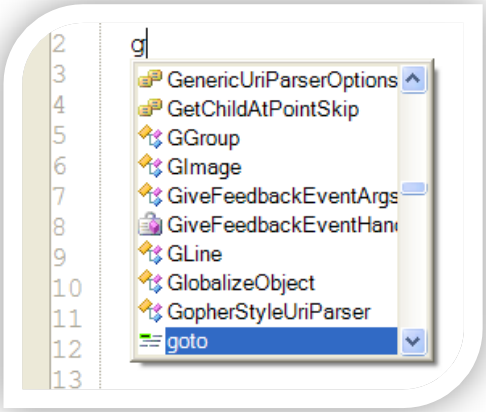

图 7.15 用户程序编辑器的智能感知：输入字符“g”

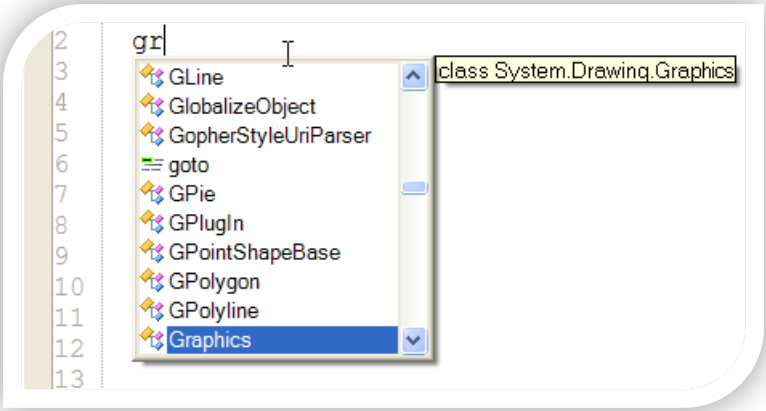

图 7.16 用户程序编辑器的智能感知：继续输入“r”

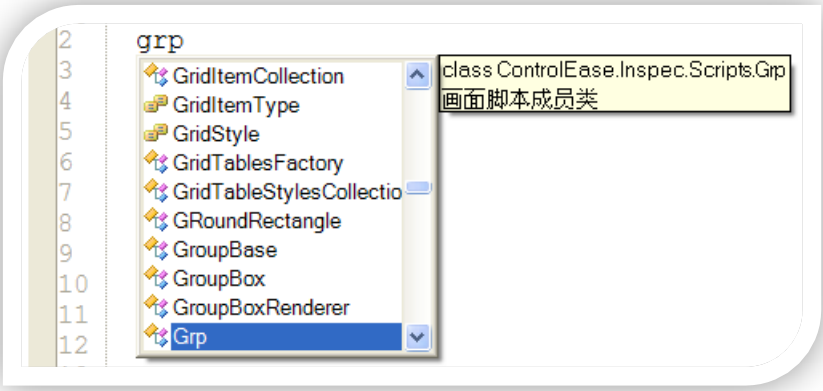

图 7.17 用户程序编辑器的智能感知：继续输入“p”

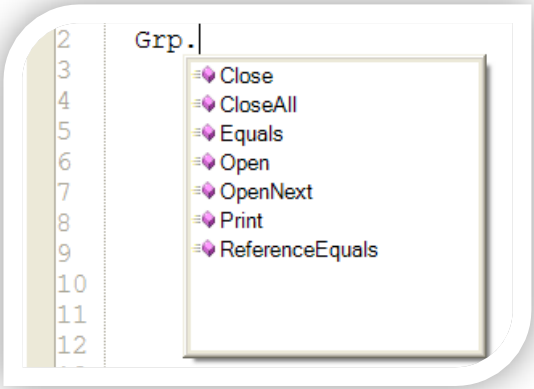

图 7.18 用户程序编辑器的智能感知：输入“.”

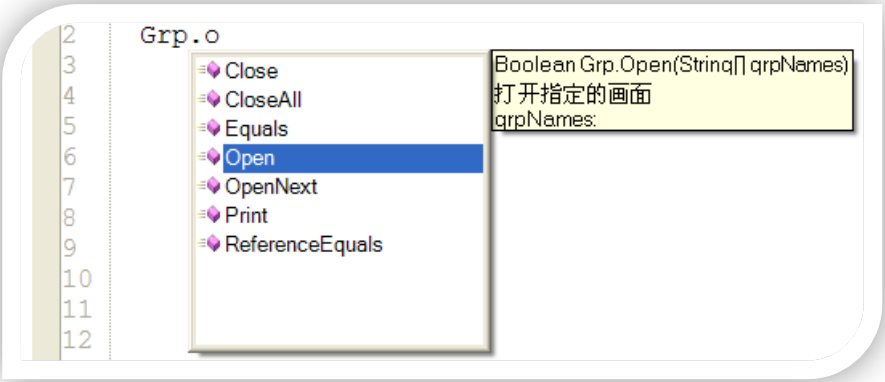

图 7.19 用户程序编辑器的智能感知示：继续输入“.”

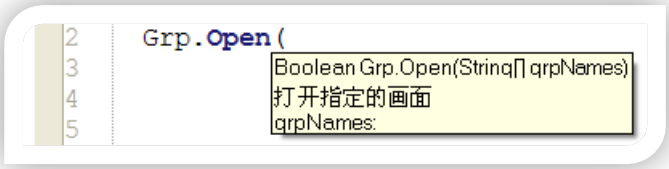

图 7.20 用户程序编辑器的智能感知：选择“Open”

第四节 自定义方法

在编写用户程序时，有时候需要到处使用重复的代码片断，如果到处拷贝粘贴，即容易出错也不便于代码的修改完善。因此需要把反复使用的代码封装到一起，定义为一个函数指令，供其它地方调用。在易控面向对象的语言中，这个函数被称为“方法”，用户自己定义的函数就是自定义方法。自定义方法是一段封装的程序片断，可以在工程的任何用户程序中调用。