Super Basic手册

现有VB6比较过时，加之VB.NET与VB6.0开发思路完全改变，故此开发Super Basic语言。该语言增加一些必要的语法特性与功能支持。对VB6.0作不完全支持。

1. 语言自述

本语言使用Basic语法，并对其进行修改与扩展，使得丰富Basic编程体验。

1. 符号表与关键字
2. 符号（具体优先级按颜色排列，同种优先级同种颜色，优先级从上到下按顺序排列）

|  |  |  |  |
| --- | --- | --- | --- |
| 符号 | 描述 | 说明 | 重载方式 |
| ID() | 双括号 | 函数调用 | Operator() |
| ID[] | 双中括号 | 数组下标访问 | Operator[] |
| ID.ID | 访问 | 对象成员访问 | 无法重载 |
| ID?.ID | 存在访问 | 如果存在访问则访问，否则返回Null | 无法重载 |
| Expr++/-- | 双加减号 | 变量自增/自减(后置) | Operator++/-- |
| Cast<T>(Expr) |  | 显式类型转换 | explicit |
| (T)ID |
| Typeof(ID) |  | 取得运行时最子类型的类型 | 无法重载 |
| !Expr | 感叹号 | 逻辑取反 | Operator! |
| Not Expr |
| ~Expr | 波浪号 | 按位取反 | Operator~ |
| BInv Expr |
| ++/--ID | 双加减号 | 变量自增运算（前置） | Static Operator++/-- |
| +/-Expr | 加减号 | 前置正负运算符号 | Operator+/- |
| \*Expr | 星号 | 地址提领操作 | 无法重载 |
| PtrVar(Expr) |
| &ID | And符号 | 取址操作 | 无法重载，可以delete |
| VarPtr(ID) |
| New T() |  | 创建实例并调用构造函数 | 通过继承重载 |
| Delete ID |  | 调用析构函数并释放类实例 |
| Expr\*//Expr | 乘除号 | 对两个数值进行Operator\*//调用 | Operator\*// |
| Expr % Expr | 取模（余） | 对两个数值 进行Operator%调用 | Operator% |
| Expr Mod Expr |
| Expr+/-Expr | 加减号 | 对两个数值进行Operator+/-调用 | Operator+/- |
| Expr<<Expr | 左移符号 | 对两个数值进行左移操作 | Operator<< |
| Expr SHL Expr |
| Expr>>Expr | 右移符号 | 对两个数值进行右移操作 | Operator>> |
| Expr SHR Expr |
| Expr<Expr | 小于 | 前者表达式判断是否小于后者 | Operator< |
| Expr<=Expr | 小于等于 | 前者表达式判断是否小于等于后者 | Operator<= |
| Expr>Expr | 大于 | 前者表达式判断是否大于等于后者 | Operator> |
| Expr>=Expr | 大于等于 | 前者表达式判断是否大于等于后者 | Operator>= |
| Expr==Expr | 逻辑等于 | 判断两表达式是否逻辑相等 | Operator== |
| Expr Eqv Expr |
| Expr!= Expr | 逻辑不等于 | 判断两表达式是否逻辑不等 | Operator!= |
| Expr NEqv Expr |
| Expr & Expr | BAnd符号 | 返回两者按位与值 | Operator& |
| Expr BAnd Expr |
| Expr ^ Expr | Xor符号 | 返回两者按位异或值 | Operator^ |
| Expr Xor Expr |
| Expr | Expr | Or符号 | 返回两者按位或值 | Operator| |
| Expr BOr Expr |
| Expr && Expr | And符号 | 返回逻辑与值 | Operator&& |
| Expr And Expr |
| Expr || Expr | Or符号 | 返回逻辑或值 | Operator|| |
| Expr Or Expr |
| Expr?Stmt:Stmt | :?判断符号 | 判断表达式，并执行对应语句 | 无法重载 |
| IIf(Expr,S,S) |
| ID=Expr | 等于 | 赋值 | Implicit T |
| += | 加等于 |  | Operator+= |
| -= | 减等于 |  | Operator-= |
| \*= | 乘等于 |  | Operator\*= |
| &= | 与等于 |  | Operator&= |
| ^= | 异或等于 |  | Operator^= |
| |= | 或等于 |  | Operator|= |
| <<= | 左移等于 |  | Operator<<= |
| >>= | 右移等于 |  | Operator>>= |
| Stmt:Stmt | 冒号 | 分割单行语句 | 无法重载 |
| Stmt,Stmt | 逗号 |

1. 关键字
2. 标色以及符号说明

关键字、系统自带的类型一律为蓝色。例如： Class、Type、Property、Function、Delegate、Long、Single。

类型、自定义类型、调用委托一律为绿色，在定义时简写为T。例如：MyType 。

标识符、以及其他语句一律为默认颜色（文档中为黑色）。例如：Dim MyVar As Long。

1. 类、过程

|  |  |  |
| --- | --- | --- |
| 名称 | 描述 | 示例 |
| Class T | 定义一个类 | Class MyClass  …Class Body…  End Class |
| Type T | 定义一个结构 | Type MyType  …Type Define…  End Type |
| Module T | 定义一个模块 | 模块可存放Class、Const、Type、Delegate  以及Static的Function、Dim  不可以存在非Static的Dim和Property |
| Interface T | 定义一个接口 | 接口是调用标准，对于Class，都可以自动生成一个接口而不必显式再实现一份接口 |
| Dim ID As T | 定义一个T类型变量 | Dim cls As MyClass  Dim arr As Long[]  Dim cls1 = New MyClass(), cls2 As MyClass, int1=20 As Long  改变  (VB6) Dim arr(1, 5) As Long  ( SB) Dim arr As Long[1][5] |
| Const ID As T | 定义一个T类型常数 | Const PI As Single = 3.1415926535 |
| Dynamic T  Dynamic T(IID, Namespace)  Dynamic T(I) | T通过接口I直接访问指定类成员(自动绑定到对应接口)  *如果不带参数定义Dynamic变量则必须在定义时初始化。* | Interface IMyClass  End Interface  Dynamic MyClass(IMyClass)  Dynamic MyClass2({123456-……}, MyTypelib) |
| Delegate T(PARAMS) As T | 定义一个调用委托  T省略或为Void时无返回值  *Delegate改写为Param时，不可含有返回值* | Delegate MyDelegate(cls As MyClass) As Long  Param MyParam(cls As MyClass) |
| Event T(PARAMS) | 继承自Param ID(PARAMS)的Event  *目前仅为保持VB6兼容性存在* | Event MyEvent(msg As Long) |
| Function ID(PARAMS) AS T | 定义一个函数  T省略或为Void时无返回值  *Function改写为Sub时，不可含有返回值* | Function MyFunc() As MyClass  …Function Body…  End Function  Function MyFunc(cls As MyClass) As Long  …Function Body…  End Function |
| Property ID As T | 定义一个成员属性  如果没有显示定义Get/Set/Let过程，则使用默认过程。  也可以在声明中单独定义Get/Set/Let，但是需要加上Static标志 | Property MyProp As Long    End Property  Property MyProp2 As Long  Get  Return 1  End Get  End Property  Static Property Let MyProp4(v As Long)  v = 1  End Property  Static Property Get MyProp3() As Long  MyProp3 = 1  End Property |
| Typedef T As T | 将T作为T的别称 | Typedef Boolean As Short  Typedef BoolAPI As Long |

1. 流控制语句

If语句：

|  |
| --- |
| If(…BooleanExpr…)Then  …Statements…  End If  If(…BooleanExpr…)Then …Statements…  If(…BooleanExpr…)Then …Statements…  Else If(…BooleanExpr…)Then …Statements… |

1. 语言细节
2. 面向对象以及类、接口

本程序提供基本的面向对象支持，比如类继承、接口继承等支持。

定义一个类

|  |
| --- |
| Class MyClass  End Class |

定义类后，该类会自动产生隐藏的接口 IMyClass，该接口可以通过Object的GetInterface方法获得。如果一个对象存储在Variant类型的变量中，则该对象必须实现一个类似于IDispatch的接口，才能获得动态调用的机能。如果对象不支持IDispatch接口，则可能无法跳过Interface直接访问类成员。如果想通过反射直接访问未含Interface的类的成员请使用Dynamic关键字定义类实例类型，请注意，使用该类型时，类必须由本语言实现，或显式的规定Interface。

|  |
| --- |
| #Import "MyCOMLib.tlb"  Module MyTestWithTypelib  Static Sub Main()  Dim MyClassID As CLSID = {123456-……}, MyClassIID As IID = {123457-……}  '假设MyClass继承自IUnknow，tlb导入MyCOMLib命名空间并提供IMyClass类接口  Dynamic MyClass(MyClassIID, MyCOMLib)  '显式规定Interface  Dim clsInstance As MyClass  clsInstance = COM.CreateInstanceByCLSID(MyClassID)  End Sub  End Module  '如果不使用Typelib，可以直接定义接口  Module MyTestWithMyInterface  [COMInterface IID={1234567-……}]  Interface MyInterface  Vitrual Function MySub() As HRESULT  [COMProduce RetName=Result]  Vitrual Function MyFunc(In param1 As Long, Out param2 As Long, Out Result As Long) As HRESULT  End Interface  Static Class Program  Dim MyClassID As CLSID = {123456-……}  Dynmic MyClass(MyInterface)  Dim clsInstance As MyClass  Dim clsInstance2 As MyInterface  clsInstance = COM.CreateInstanceByCLSID(MyClassID)  clsInstance2 = clsInstance  End Class  End Module |

1. 基本数据类型

|  |  |  |
| --- | --- | --- |
| 名称 | 大小 | 描述 |
| LongLong | 8 | Int64别名 |
| QWord | 8 | Int64别名 |
| Long | 4 | Int32别名 |
| ULong | 4 | UInt32别名 |
| Integer | 4 | Int32别名 |
| UInteger | 4 | UInt32别名 |
| Short | 2 | Int16别名 |
| UShort | 2 | UInt16别名 |
| Char | 2 | Int8别名，代表字符 |
| Byte | 1 | UInt8别名 |
| Bool | 1 | Bool8别名 |
| Boolean | 2 | Bool16别名 |
| Bool8 | 1 | 带符号1字节整数，0为False，其他为True |
| Bool16 | 2 | 带符号2字节整数，0为False，其他为True |
| Bool32 | 4 | 带符号4字节整数 |
| Int8 | 1 | 带符号1字节整数 |
| Int16 | 2 | 带符号2字节整数 |
| Int32 | 4 | 带符号4字节整数 |
| Int64 | 8 | 带符号8字节整数 |
| UInt8 | 1 | 无符号1字节整数 |
| UInt16 | 2 | 无符号2字节整数 |
| UInt32 | 4 | 无符号4字节整数 |
| UInt64 | 8 | 无符号8字节整数 |
| IntPtr | 4-8 | 平台相关带符号指针 |
| UIntPtr | 4-8 | 平台相关无符号指针 |
| Half | 2 | 半精度浮点数 |
| Single | 4 | 单精度浮点数 |
| Double | 8 | 双精度浮点数 |
|  |  |  |

1. 可调用对象（委托）以及
2. 其余