IEC中的面向对象编程(OOP)

内容概述

- IEC 61131-3 编程标准第三版
- 详细内容介绍
- OOP编程的应用

历史

- 1993年第一版
- 2003年第二版：错误修正和内容扩充
- 2011年第三版
  - 2008年着手编辑
  - 2009年7月第一份草案
  - 2010年发布

新的数据类型

- LINT, ULINT, LWORD
  - 64位

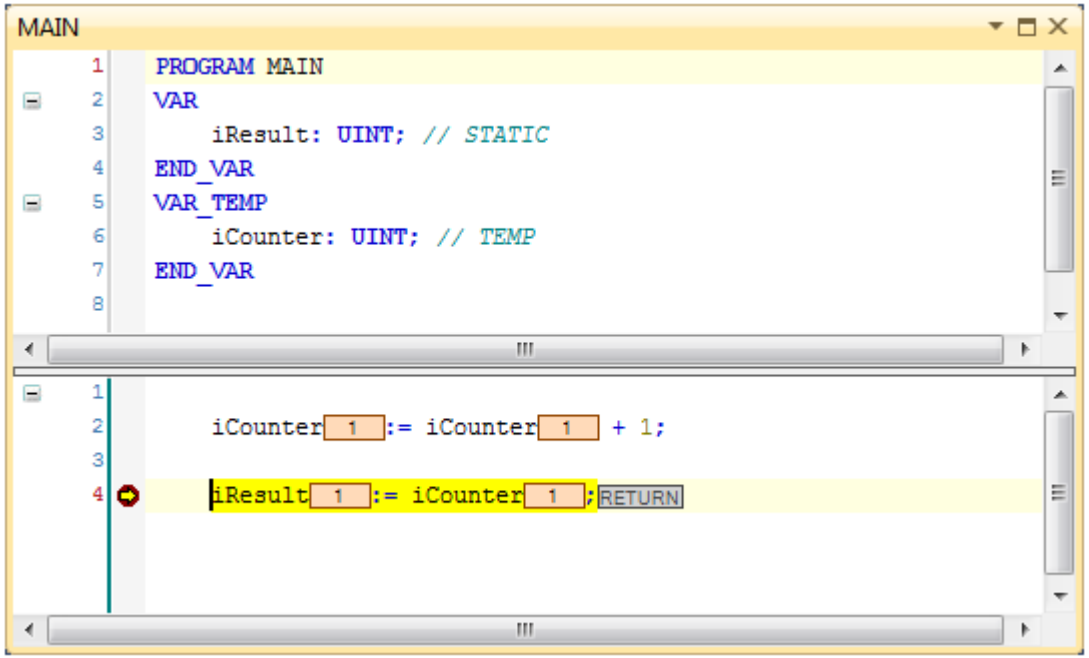
- LTIME, LDATE, LDATE_AND_TIME, LTIME_OF_DAY
  - tTime:= LTIME#12s34ms2us44ns
  - DT:= CONCAT_DATE_TOD(DATE, TOD);
  - DATE:= CONCAT_DATE(YEAR, MONTH, DAY);

- UNION
  - 多个变量共用同一段内存地址
- WSTRING
  - Unicode编码

```
sChinese: WSTRING:= "答复";
```

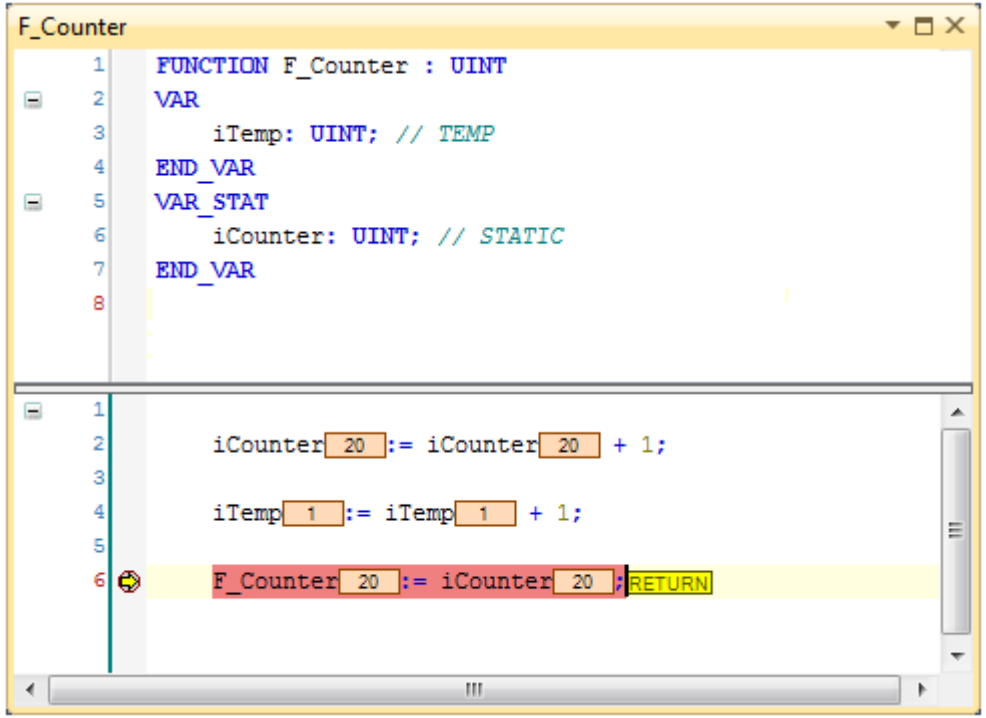
VAR_TEMP - 临时变量

- 临时变量在每次调用POU时都会初始化。
- 用在程序和功能块中。
- 在方法和函数中的内部变量都是临时变量!

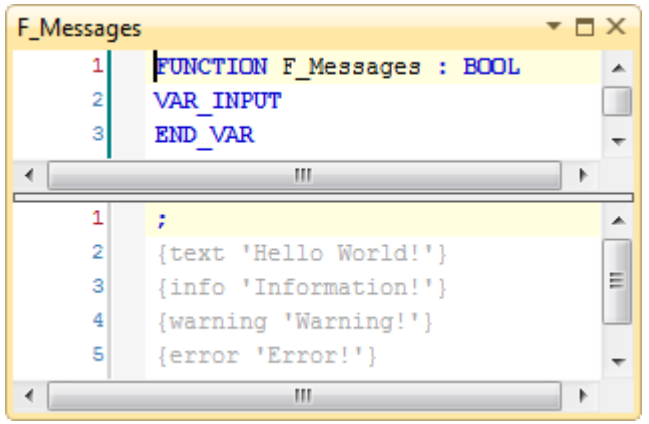


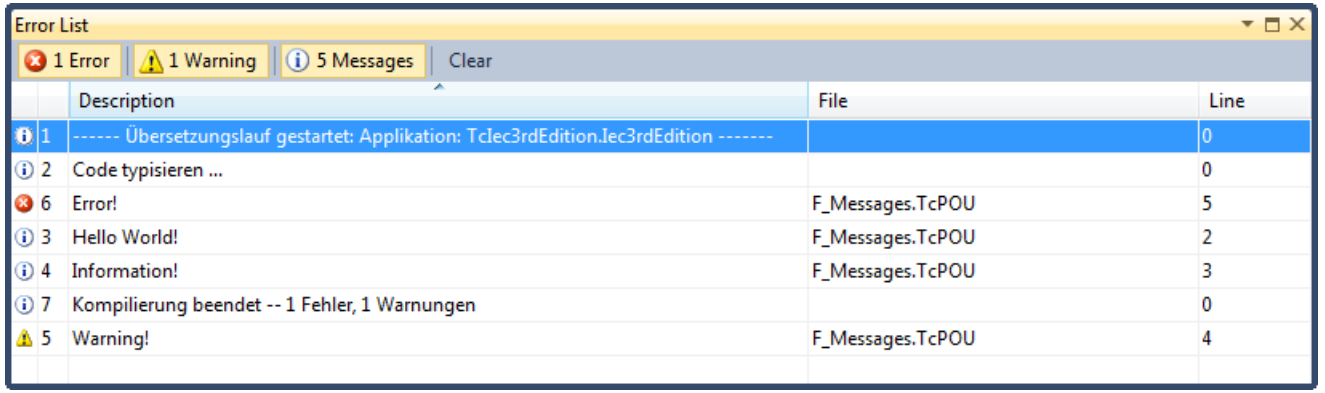
VAR_STAT - 静态变量

- 静态变量只在POU的第一次调用时被初始化。
- 静态变量声明赋值后，值不会丢失。
- 可用在方法和函数中。
- 在程序和功能块中声明的变量都是静态的!



- Messages
  - {text 'Hello World!'}
  - {info 'Information!'}
  - {warning 'Warning!'}
  - {error 'Error!'}





- {defined identifier}
- {undefined identifier}
- {IF expr}
- {ELSIF expr}
- {ELSE}
- {END_IF}

  - {IF defined(variable:iInput)}
  - {IF defined(pou:CheckBounds)}
  - {IF defined(type:ST_DataType)}
  - {IF hasattribute(variable:iInput, 'DoSomething')}
  - {IF hastype(variable:iInput, UINT)}

MAIN

```
1  PROGRAM MAIN
2  VAR
3  {IF defined(bA)}
4      bA: BOOL;
5  {END_IF}
6  END_VAR
```

```
1
2  {IF defined(variable:bA)}
3      {info 'Variable bA is defined!'}
4  {ELSE}
5      {error 'No variable bA found!'}
6  {END_IF}
7
8  bA:= TRUE;
9
```

Error List

3 Errors | 0 Warnings | 4 Messages | Clear

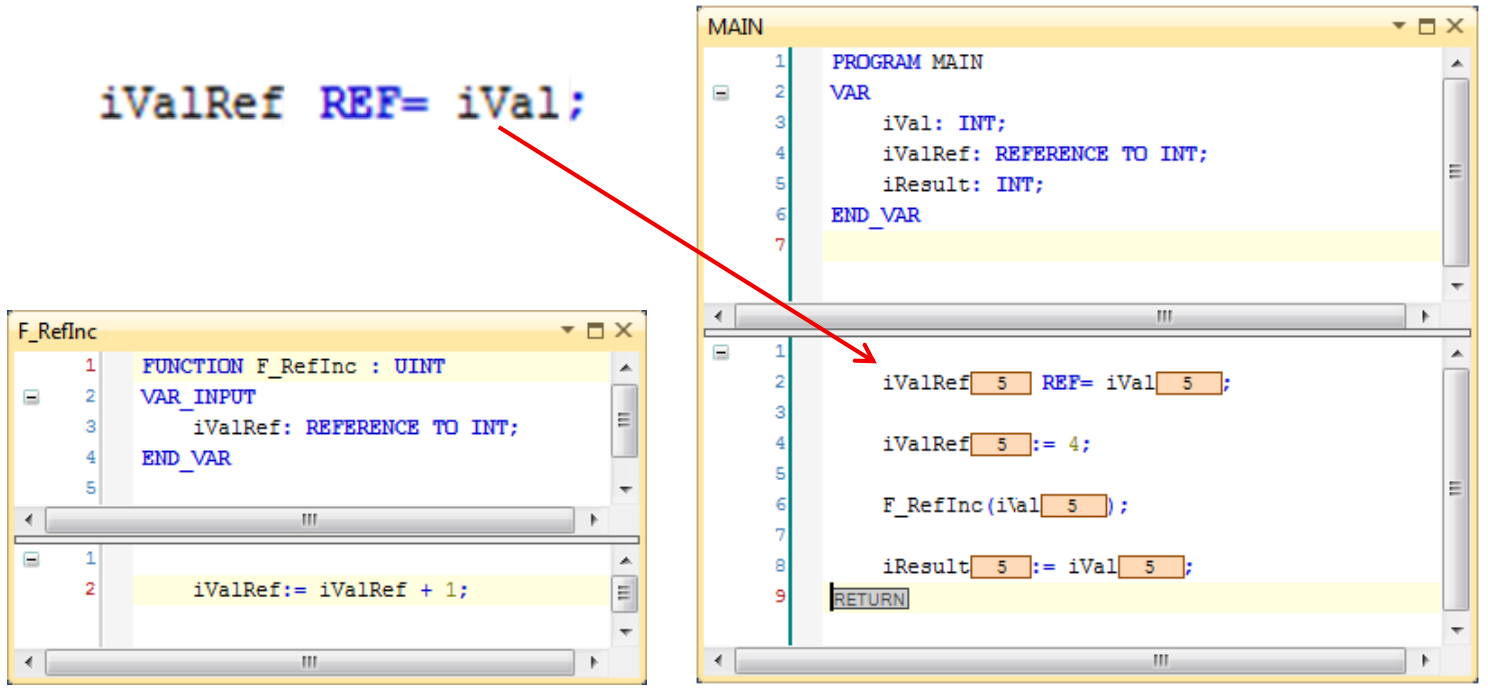
	Description	File	Line
5	'bA' ist kein gültiges Zuweisungsziel	MAIN.TcPOU	8
1	------ Übersetzungslauf gestartet: Applikation: TcIec3rdEdition.Iec3rdEdition -------		0
4	Bezeichner 'bA' nicht definiert	MAIN.TcPOU	8
2	Code typisieren ...		0
6	Kompilierung beendet -- 3 Fehler, 0 Warnungen		0
3	No variable bA found!	MAIN.TcPOU	5
7	Übersetzen beendet -- 3 Fehler, 0 Warnungen : kein Download möglich!		0

第三类：属性编译，需要我们添加属性

属性种类很多，具体都可以在帮助文档找到（搜索Pragmas）

如果想简单了解一下，可以看OOP的教学视频（8：09开始简单讲了一个hide隐藏属性）

- 引用类型的变量可以看做是一个对象的别名。
- 作用与指针型变量一样,可以代替指针型变量。



面向对象编程（OOP）

- TwinCAT 3是多功能的PLC开发平台（可以在线调试，修改等）
- 将面向对象的思想融入到PLC程序中（可以很简单地对已有项目进行升级改造）
- OOP并不是一种新的PLC编程语言，而是对IEC标准语言的扩充。
- OOP适用于所有IEC语言（不仅仅是ST）
- IEC标准组织已经将OOP融入IEC61131-3标准中。

➢ IEC61131-3第三版，2011年成为一个编程标准。

内容概述

- IEC 61131-3 编程标准第三版
- 详细内容介绍
- OOP编程的应用

关键词:

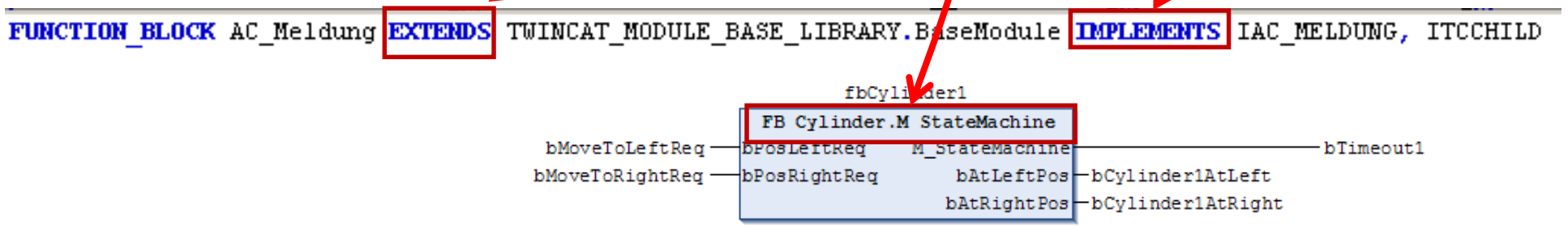
- METHOD：方法，带自定义变量的FB action。
- EXTENDS：继承，获得继承对象的方法，变量，属性等。继承的对象可以是FB或结构类型变量
- THIS：一种指针，引用当前功能块的方法、变量或属性。
- SUPER：一种指针，引用父类功能块的方法、变量或属性。
- INTERFACE：接口，一种复杂的结构类型，包含变量，方法，属性。（类似于没有代码的FB）
- IMPLEMENTS in the FB: 功能块引用一个指定的接口类型。
- PROPERTY: 属性，使用Set/Get方法来访问变量的一种方式。

方法的调用:

- *对象*.方法名 **(...)**

IEC 61131-3第三版中面向对象增加的概念:

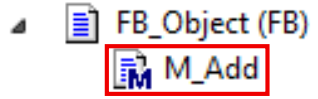
- 概念:
  - Classes
  - Interfaces
  - Methods
  - Inheritance
  - Attributes
  - Keywords such as THIS, SUPER

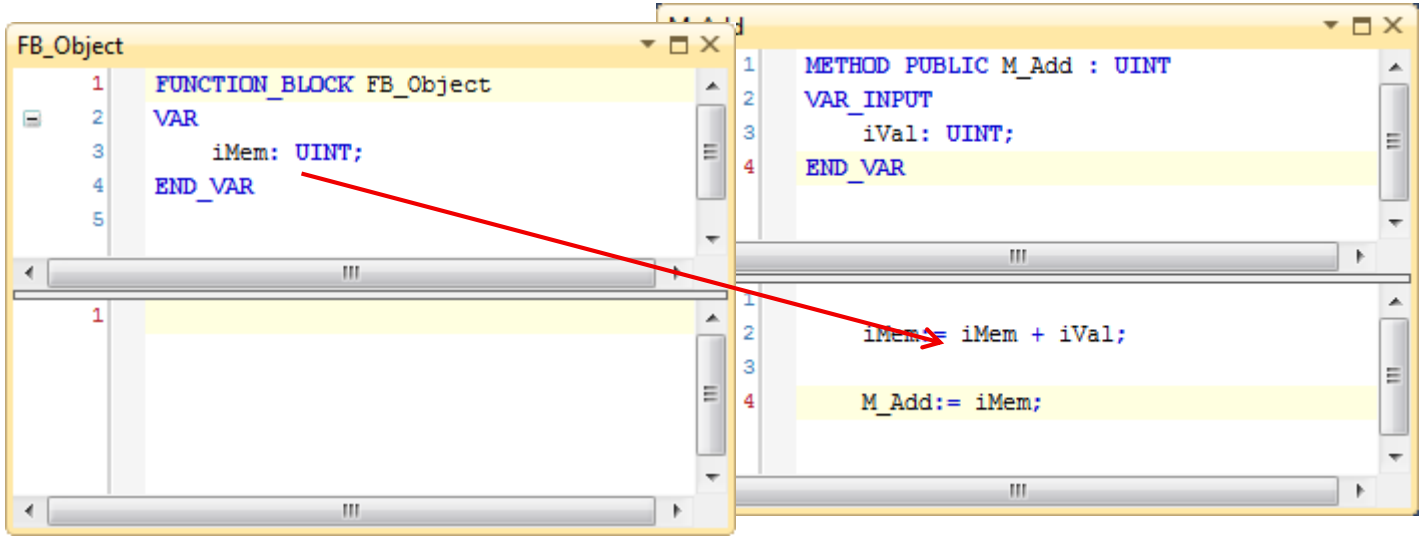


- 扩展的功能
  适用于所有的IEC语言
  与硬件无关
  可以不使用这些扩展的功能

方法 METHOD

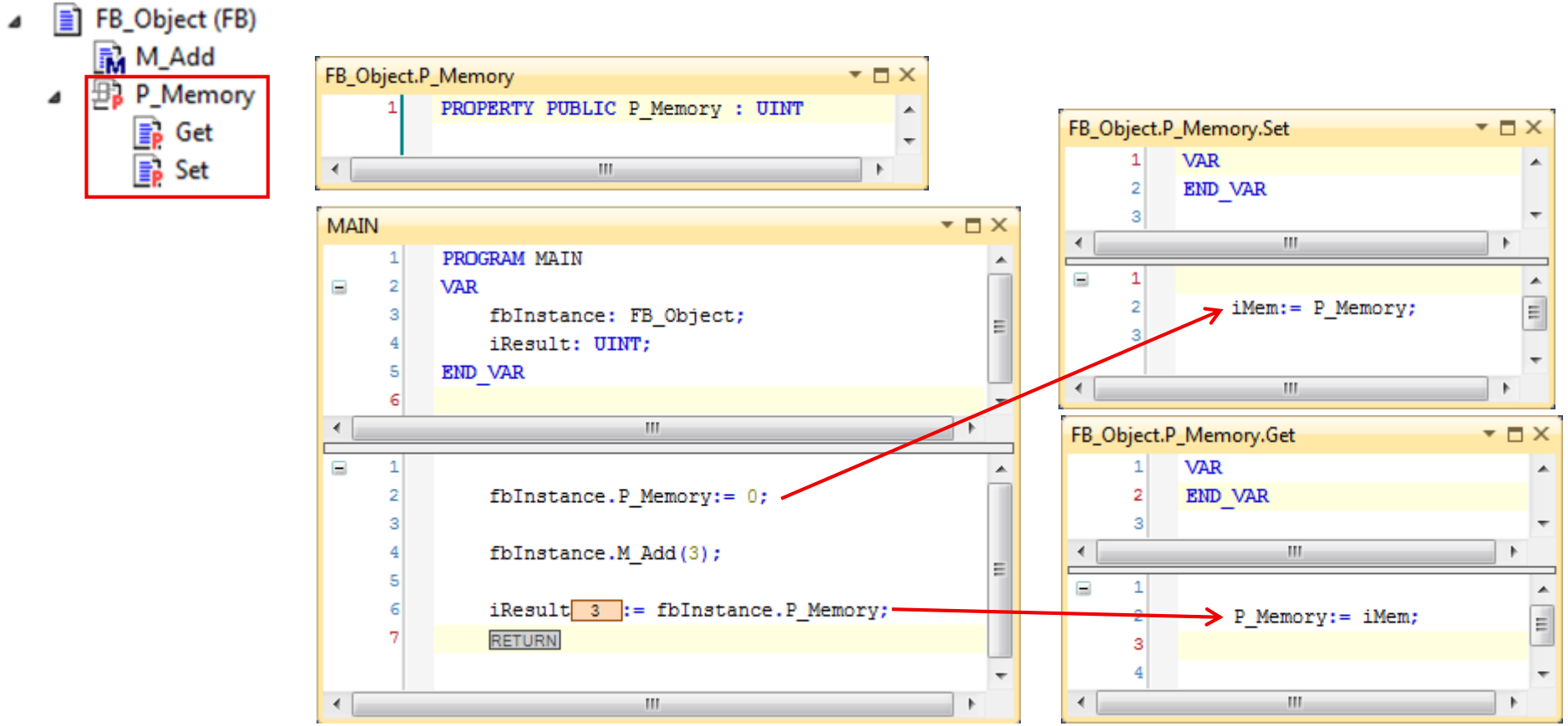
- 方法描述了一系列的动作指令,实现目的功能。
- 类似于函数，依附功能块。
- 将功能块的ACTION扩展以下内容。
  - VAR_INPUT, VAR_IN_OUT and VAR_OUTPUT
- 可以访问所属功能块中的变量。





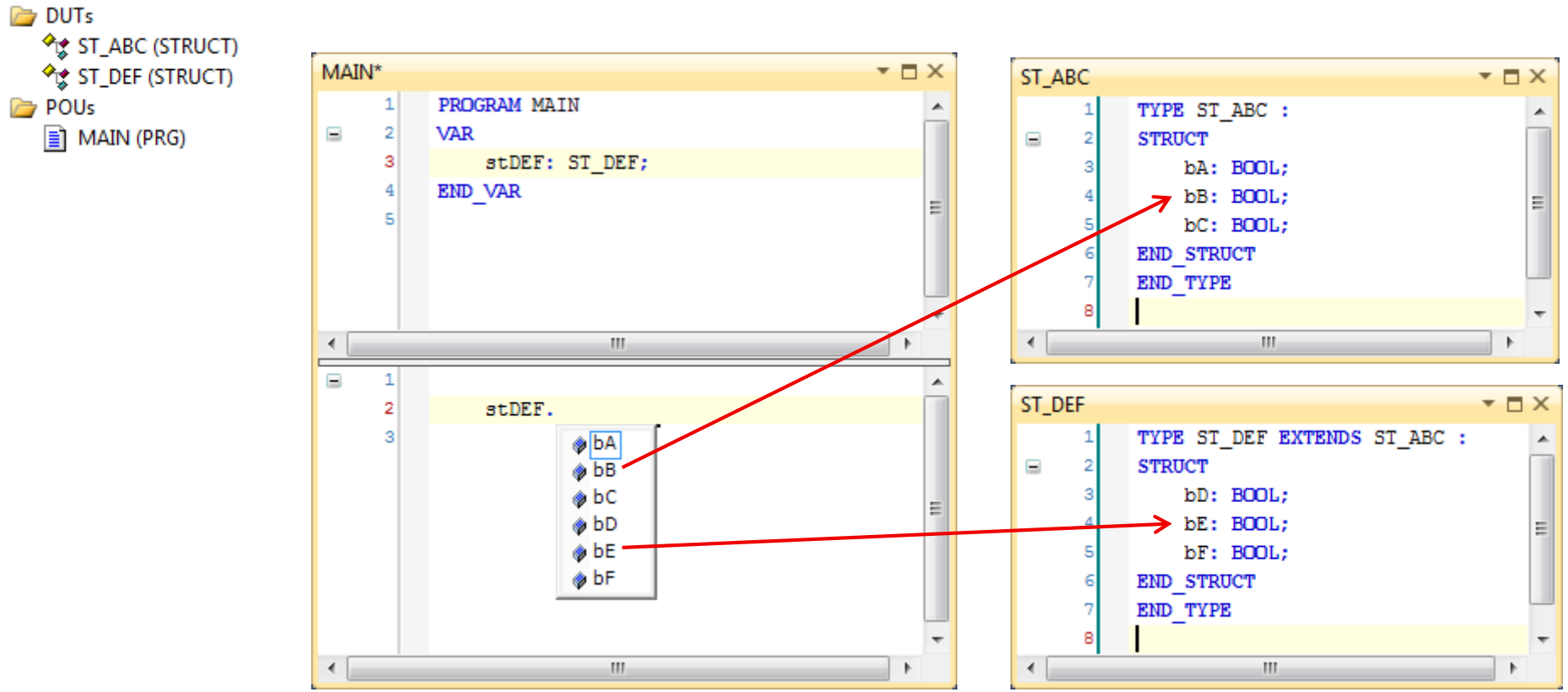
属性 PROPERTY

- 访问功能块内部变量的状态或值。
- 属性提供了Get和Set方法对目标进行读取和设置。



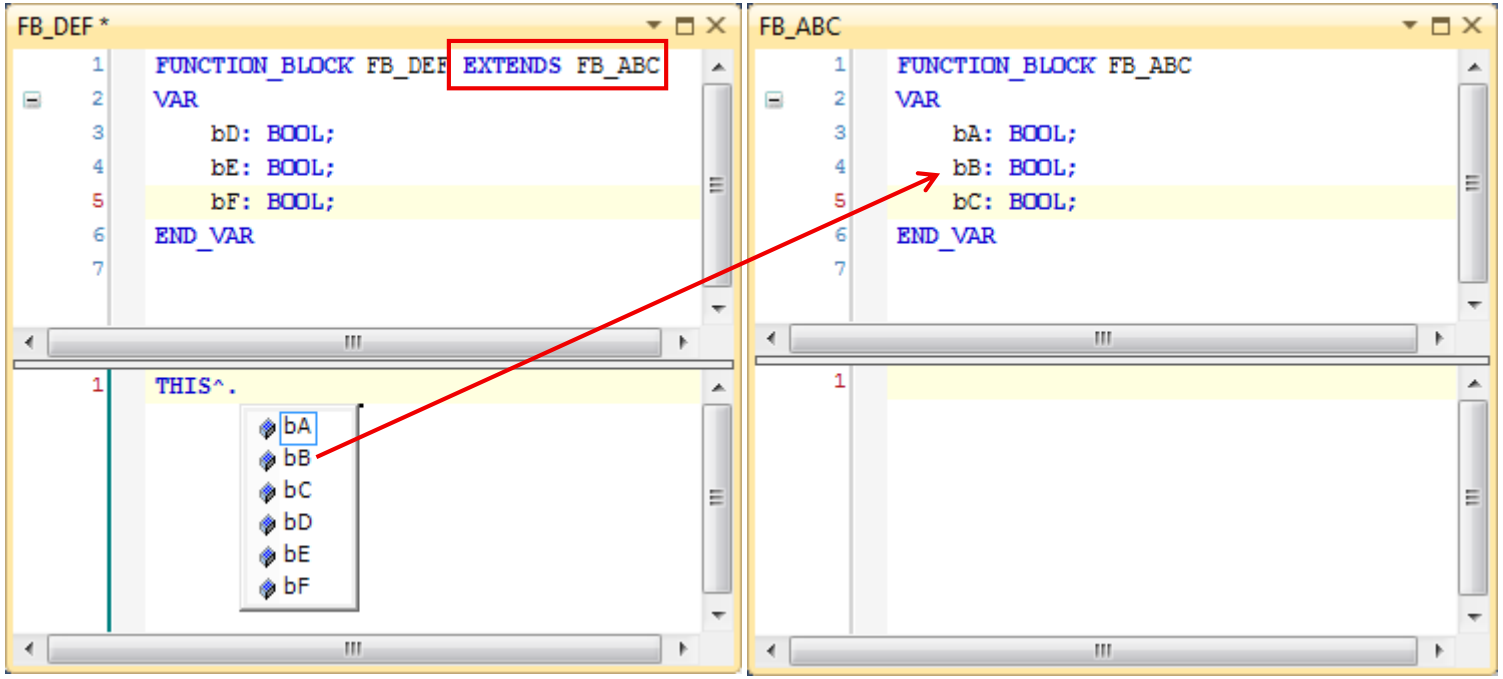
扩展 EXTENDS

- 结构体包含了扩展前的所有变量。



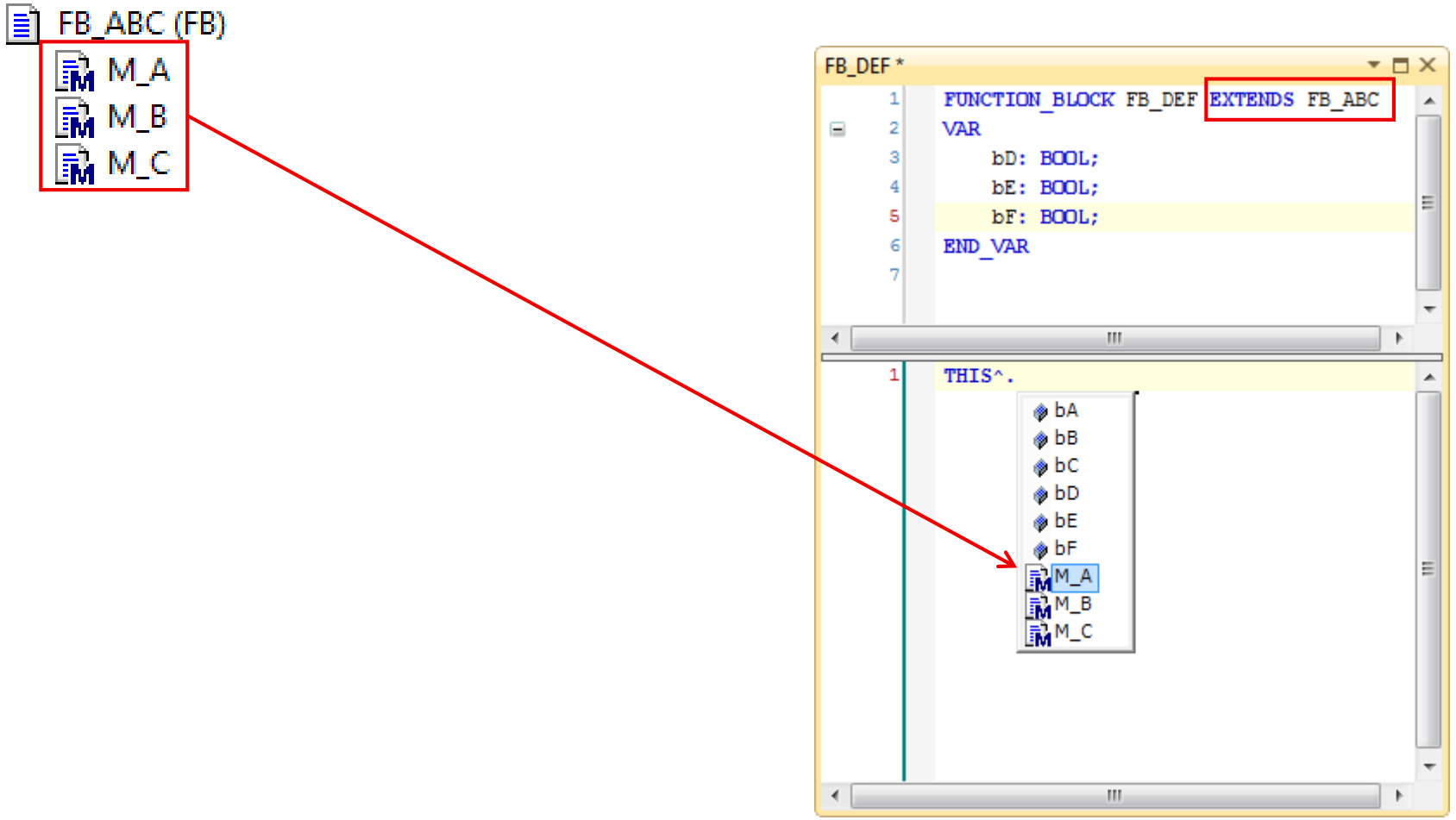
- 功能块包括了扩展前的所有变量。

FB_ABC (FB)



扩展 EXTENDS

- 功能块包括了扩展前的所有方法。



SUPER 和 THIS

- SUPER 一种指针，引用父类功能块的方法、变量或属性。
- THIS 一种指针，引用当前功能块的方法、变量或属性。

FB_1

```
1  FUNCTION_BLOCK FB_1 EXTENDS FB_Base
2  VAR_INPUT
3      bLocal: BOOL;
4  END_VAR
5  VAR_OUTPUT
6      iBase: INT;
7  END_VAR
8  VAR
9  END_VAR
```

```
1  IF bLocal THEN
2      //Aufruf der Methoden aus FB_1
3      THIS^.METH_DoIt();
4      THIS^.METH_DoAlso();
5      iBase:=iCnt;
6  ELSE
7      //Aufruf der Methoden aus FB_Base
8      SUPER^.METH_DoIt();
9      SUPER^.METH_DoAlso();
10     iBase:=SUPER^.iCnt;
11 END_IF
```

FB_Base.METH_DoAlso

```
1  METHOD PUBLIC METH_DoAlso : BOOL
2  VAR_INPUT
3  END_VAR
```

```
1  METH_DoAlso:=TRUE;
```

FB_Base.METH_DoIt

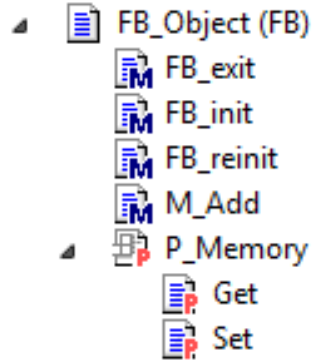
```
1  METHOD PUBLIC METH_DoIt : BOOL
2  VAR_INPUT
3  END_VAR
```

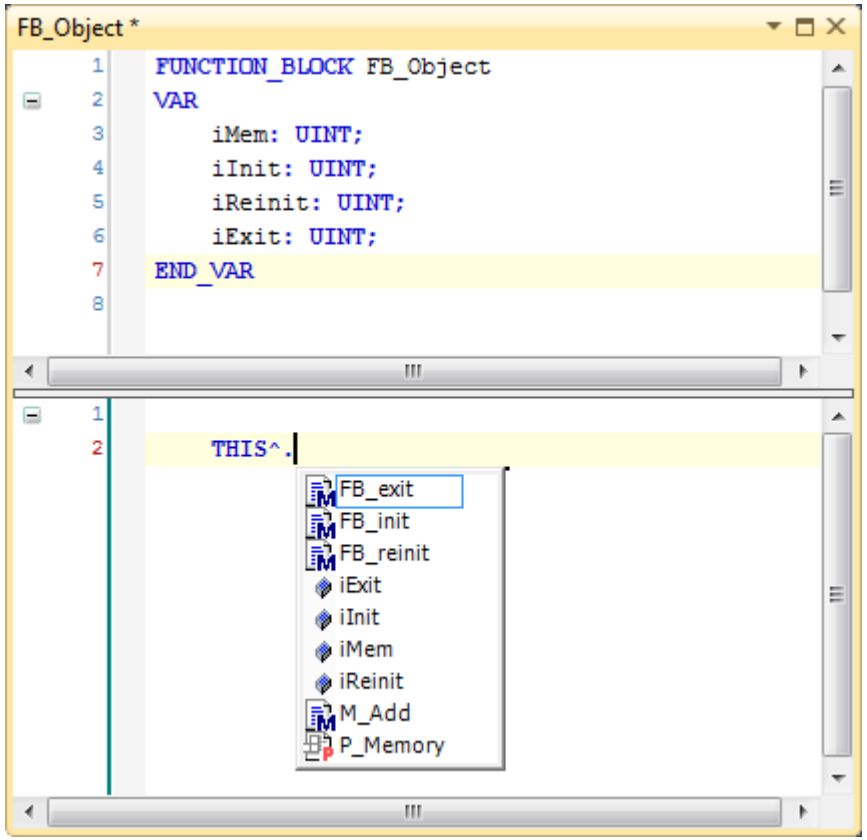
```
1  iCnt:=-1;
```

Textual

THIS

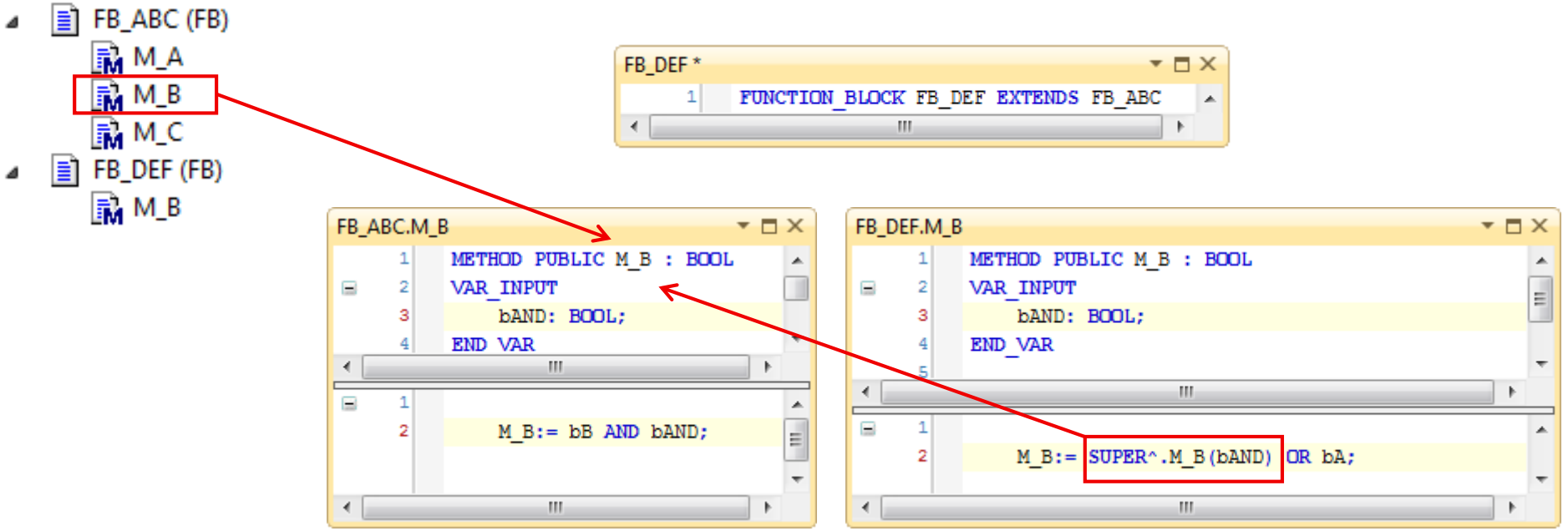
- 每个功能块都提供指针THIS。
- THIS指向当前功能块。



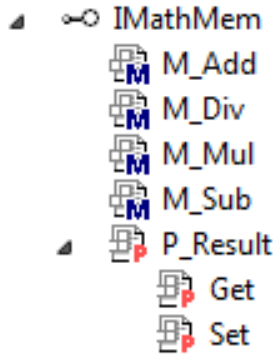


- 每一个功能块都提供了指针SUPER
- SUPER指向原有的功能块

- 用原有的方法覆盖现有的方法
  - 改变了现有功能块的行为

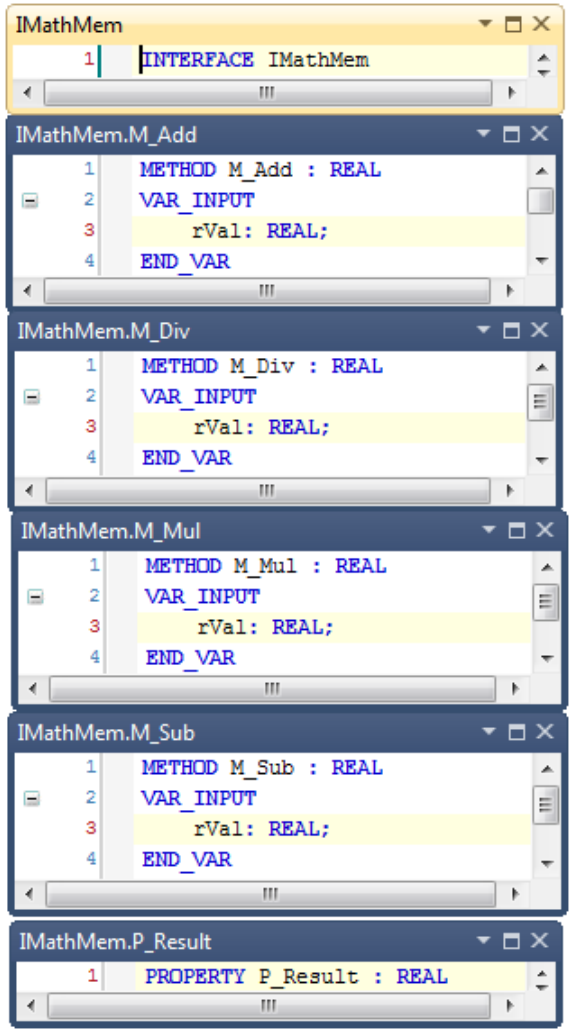


接口 INTERFACE



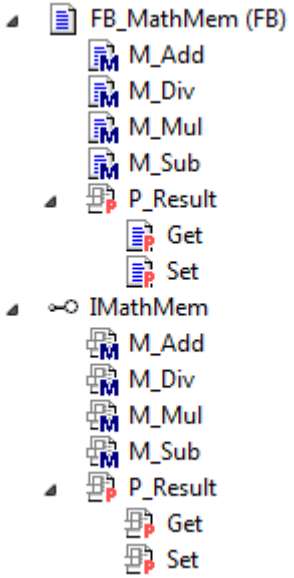
- 一种POU可以引用的复杂结构类型，包含变量、方法和属性。

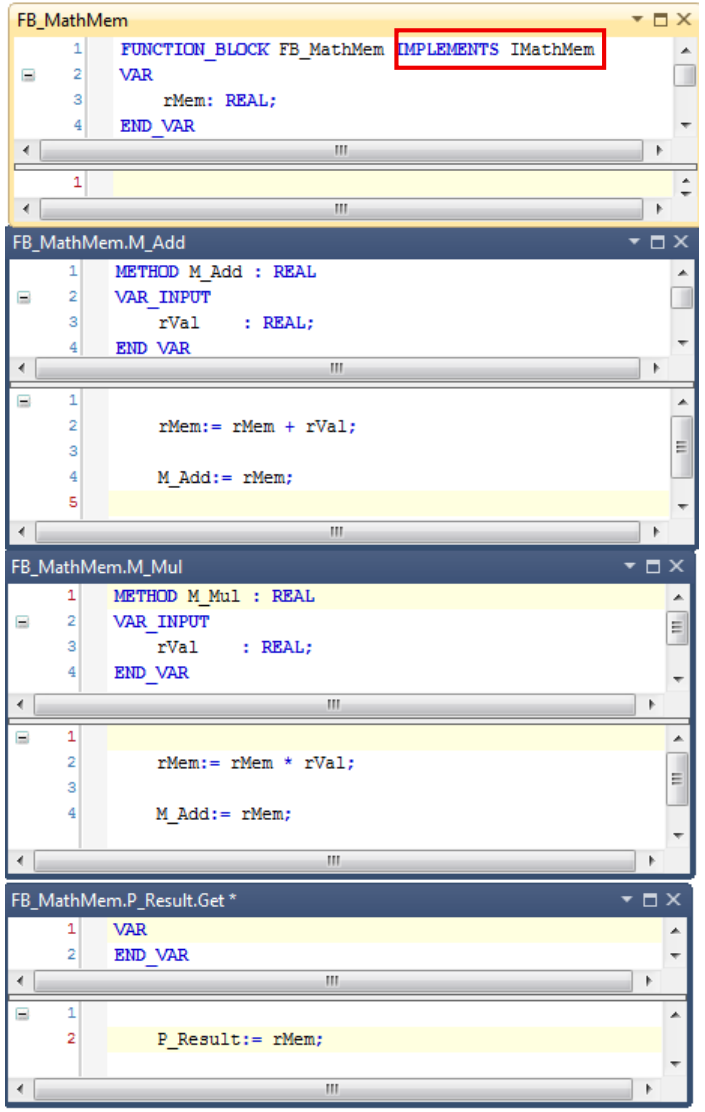
- 只是声明，没有实际代码
- 只有输入，输出和输入输出变量
- 没有内部变量。



实现 IMPLEMENTS

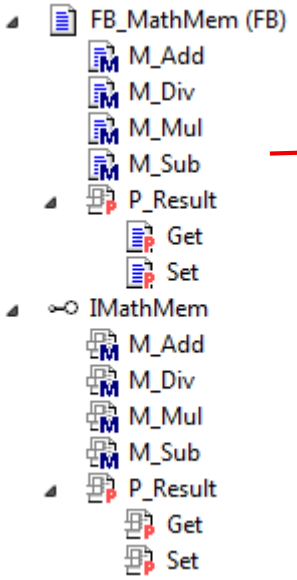
- 声明一个新的功能块去引用接口。

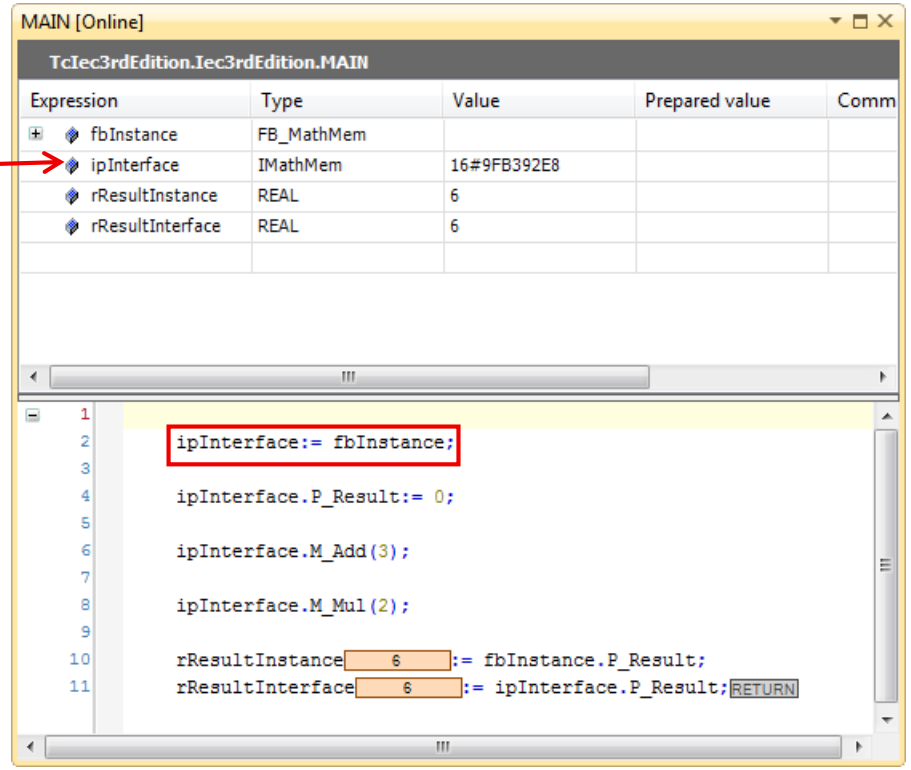




接口的用法

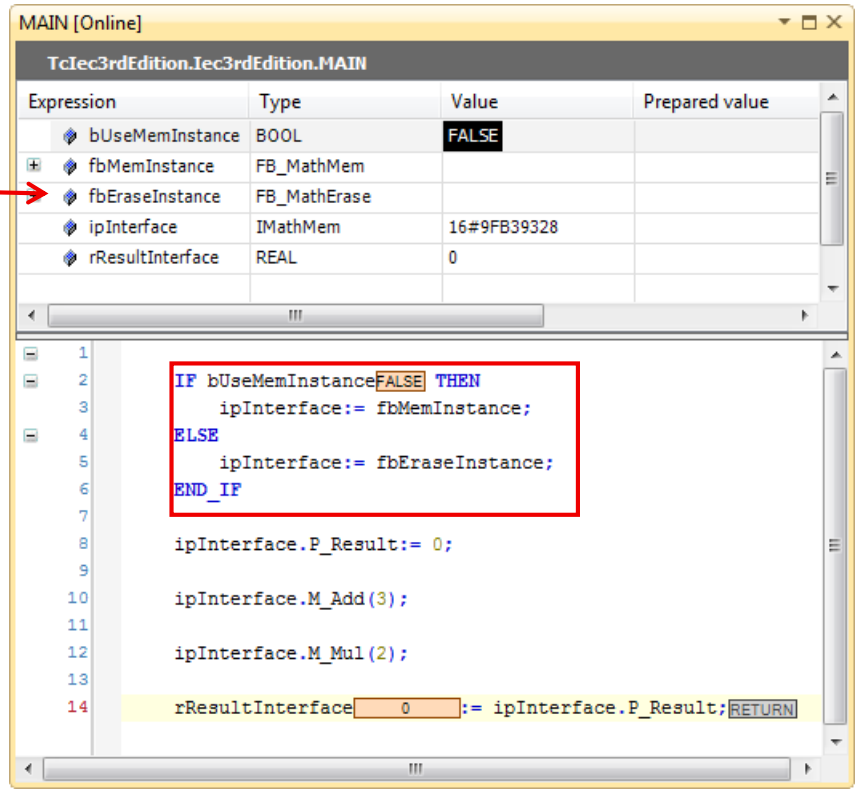
- 实例化功能块，引用该接口。
- 声明接口的变量。
- 将接口指向功能块。





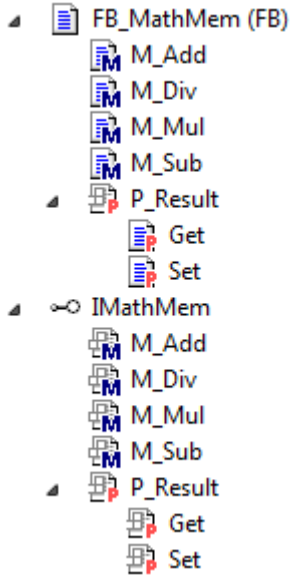
接口的优点

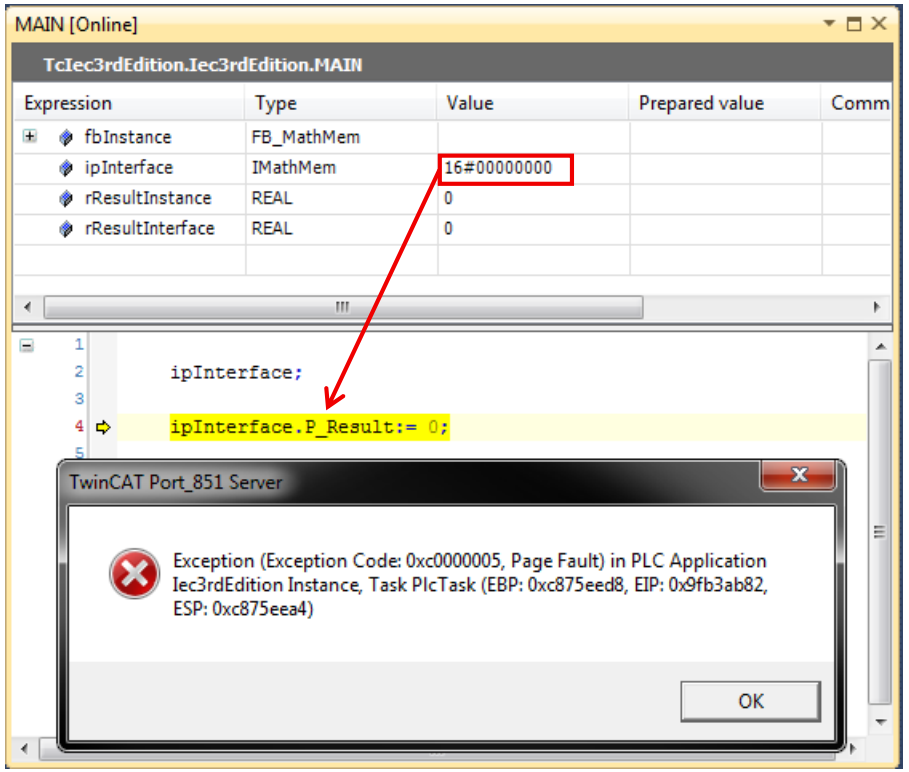
- 根据情况使用不同的FB。
  - 如果条件满足，那么就使用FB_MathMem。
  - 否则使用FB_MathErase。



- 功能块内部是具体的值。
- 接口存储功能块的地址。

➔ 接口是指针！



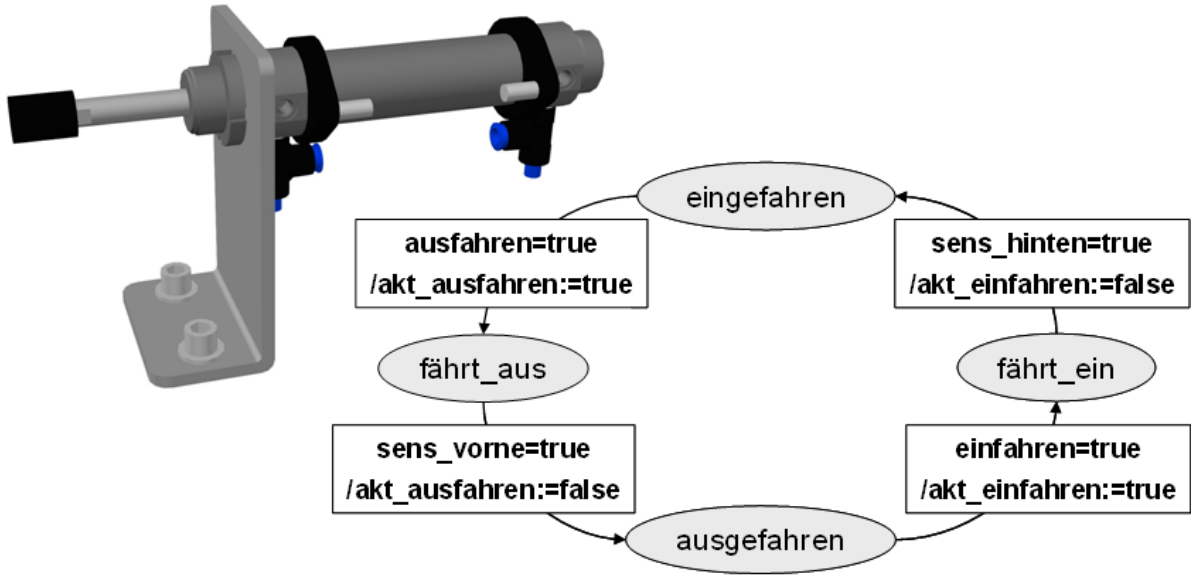


内容概述

- IEC 61131-3 第三版的面向对象编程
- 详细内容介绍
- OOP的示例应用

Sample：气缸

Example: 气缸

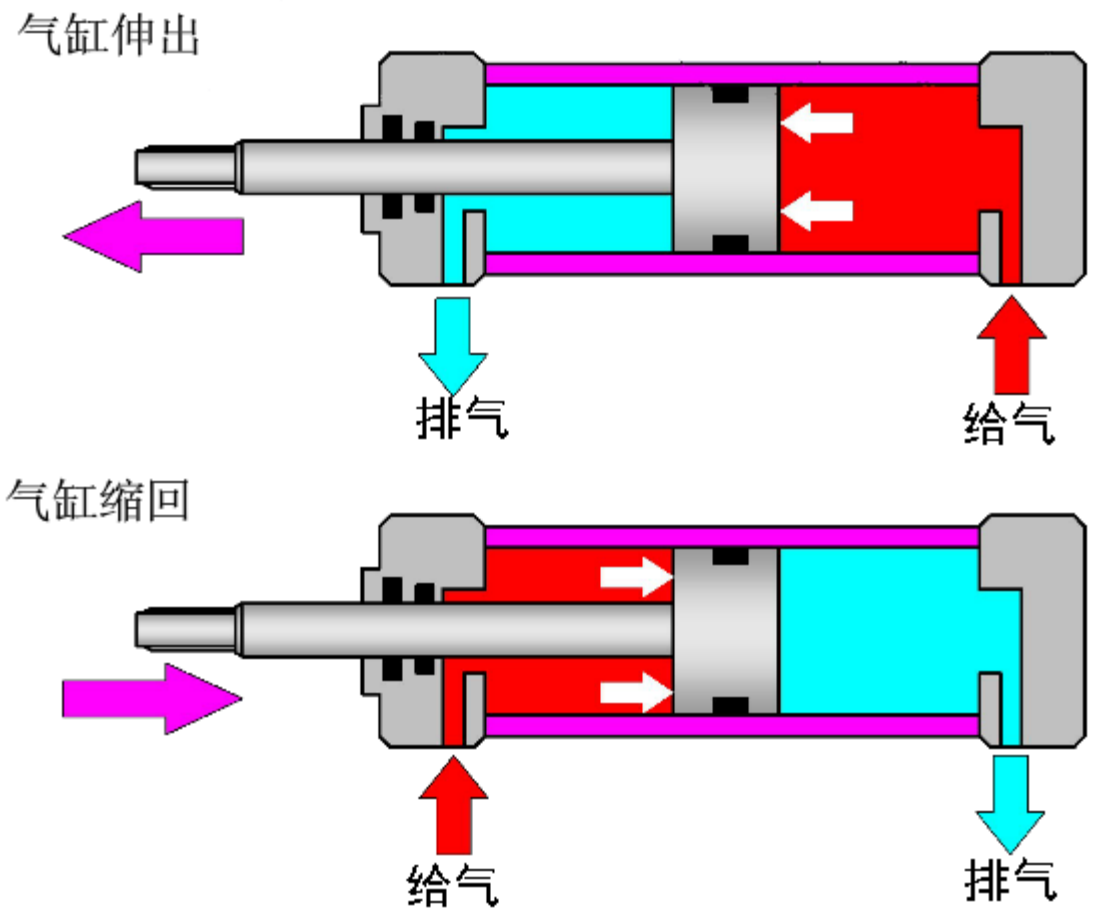


任务提示：

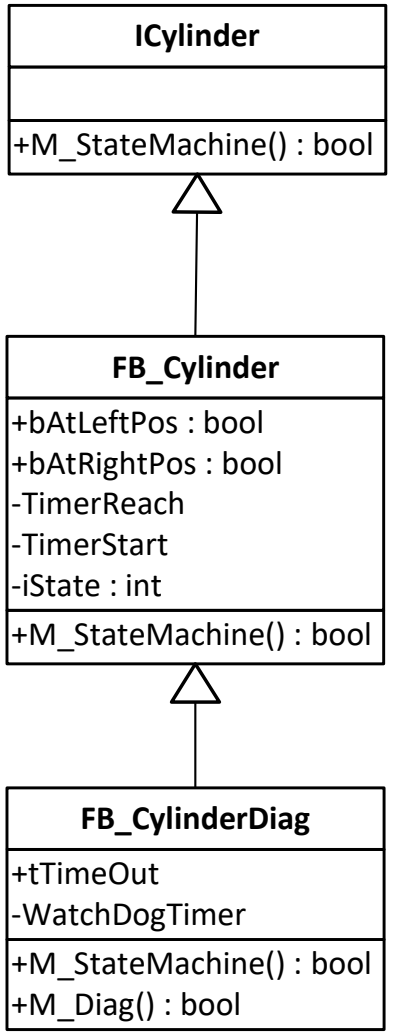
- 同一类型的气缸
- 一致的动作
- 有带诊断的气缸

Sample：气缸

- 气缸的工作原理

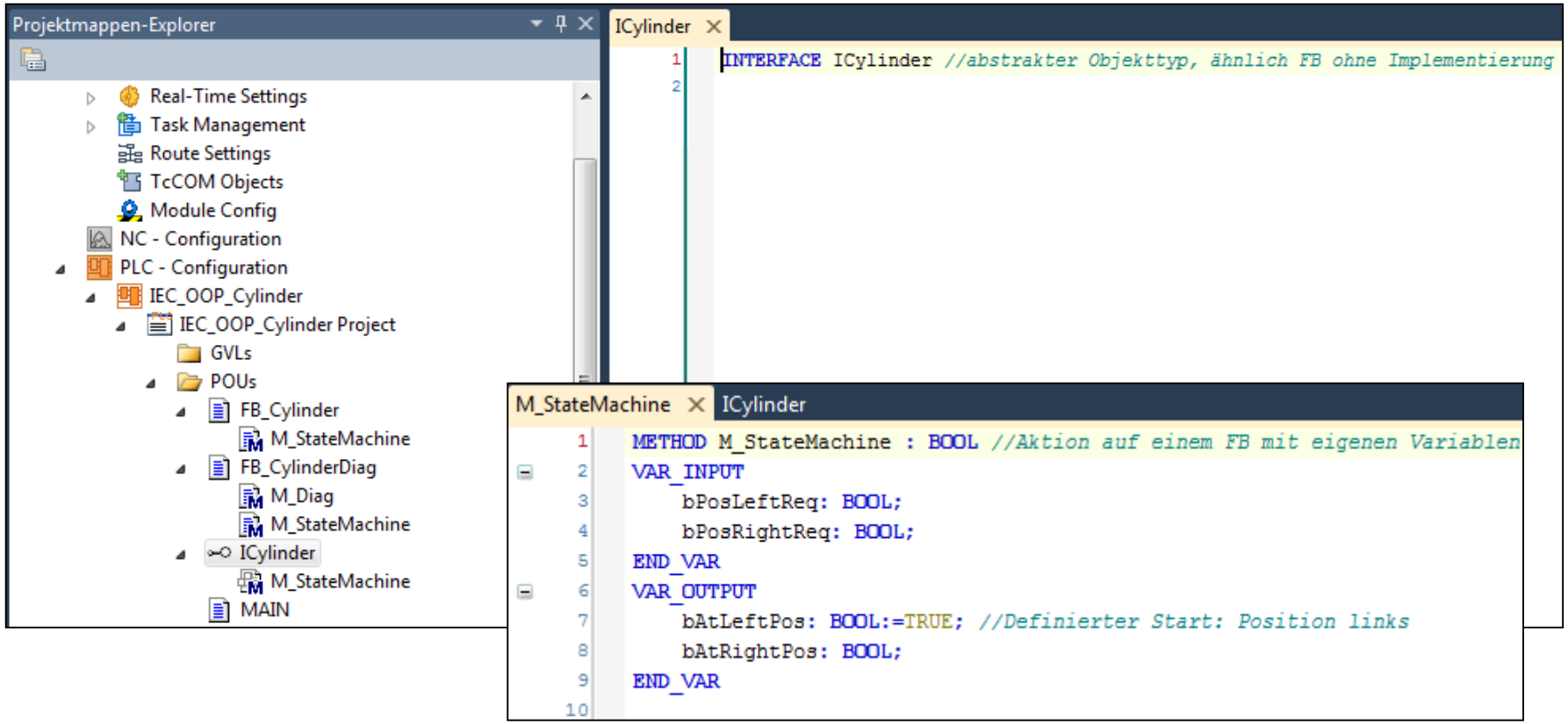


Sample : PLC | OOA 气缸



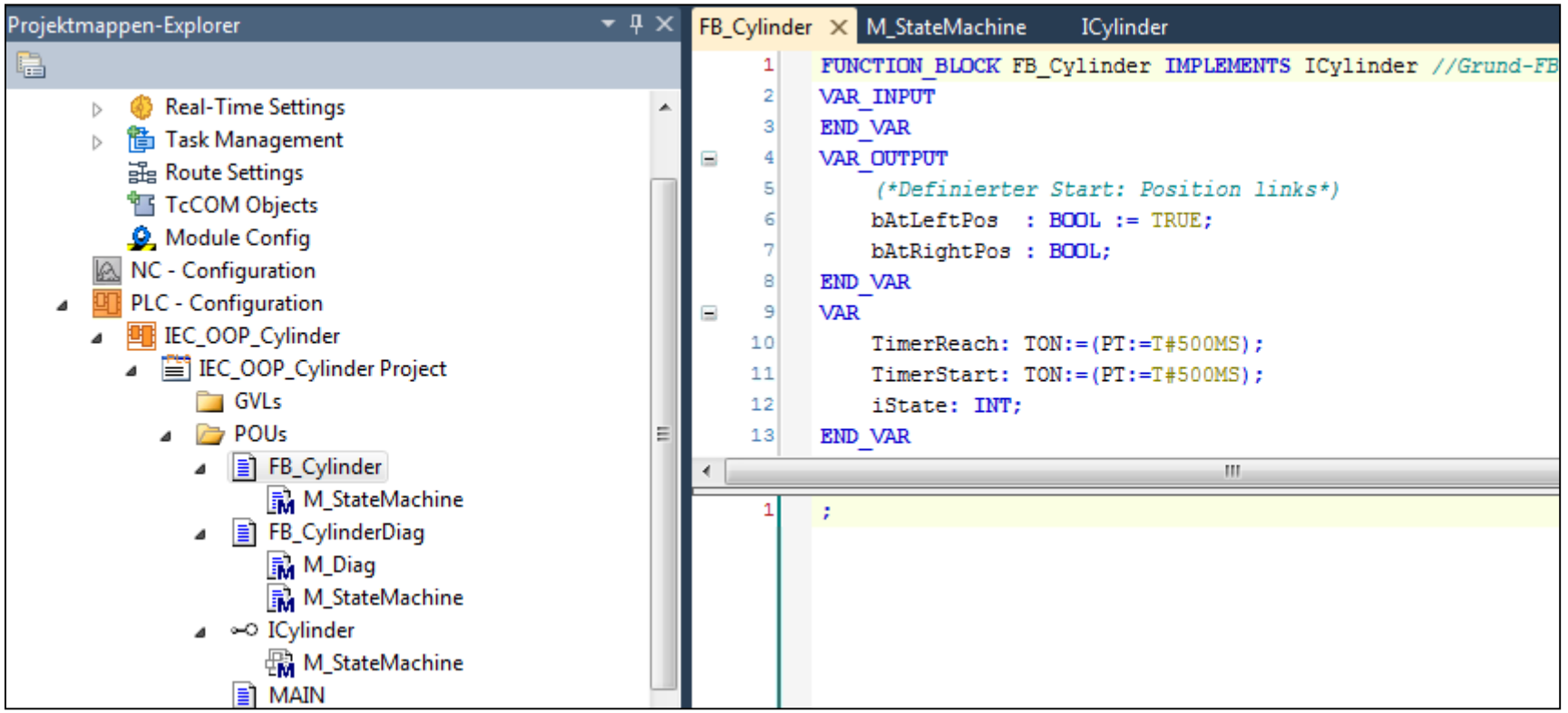
Sample：PLC | 气缸 | 接口

- 定义一个抽象的父类 (接口)：I 气缸 ICylinder
- 相关的方法只是被创建，相关的数据仅在实现时被定义。



Sample：PLC | 气缸 | 功能块

- 定义一个基本功能块FB_Cylinder来实现接口 I 气缸。
- 气缸的基本动作都在这个功能块中定义。



Sample：PLC | 气缸 | 方法

- 用于控制所有气缸的方法：

接口方法的实现

M_StateMachine | FB_Cylinder | M_StateMachine | ICylinder

```
METHOD M_StateMachine : BOOL (*Aktion auf einem FB mit eigenen Variablen, kein Quellcode*)
VAR_INPUT
    bPosLeftReq : BOOL;
    bPosRightReq    : BOOL;
END_VAR
VAR_OUTPUT
    (*Definierter Start: Position links*)
    bAtLeftPos  : BOOL := TRUE;
    bAtRightPos : BOOL;
END_VAR
```

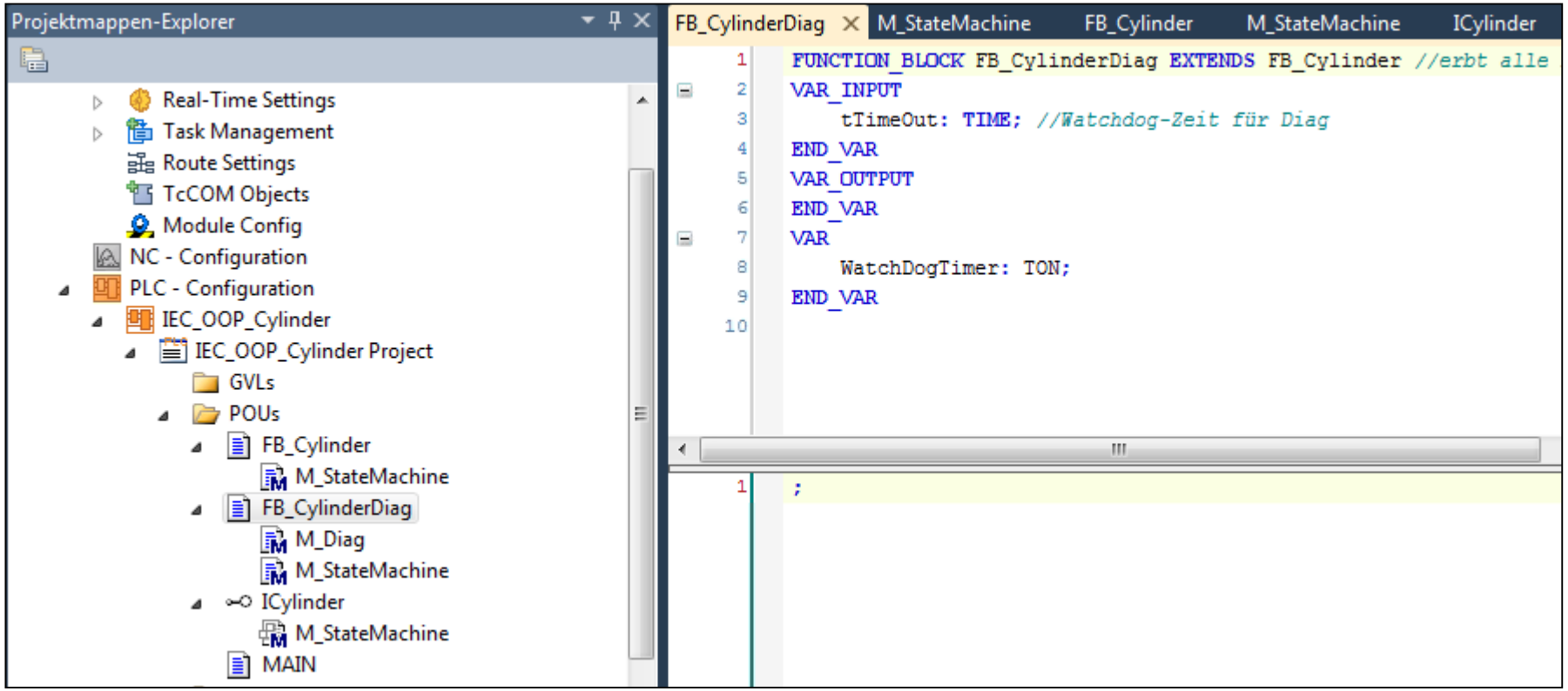
```
(*simple cylinder with feedback*)
CASE iState OF
    0: //at left pos
    bAtLeftPos:=TRUE;
    bAtRightPos:=FALSE;

    IF bPosRightReq THEN
        TimerStart(IN:=FALSE);
        TimerReach(IN:=FALSE);
        iState:=iState+1;
    END_IF

    1: //start motion to right side
    TimerStart(IN:=TRUE);
```

Sample : PLC | 气缸 | 继承

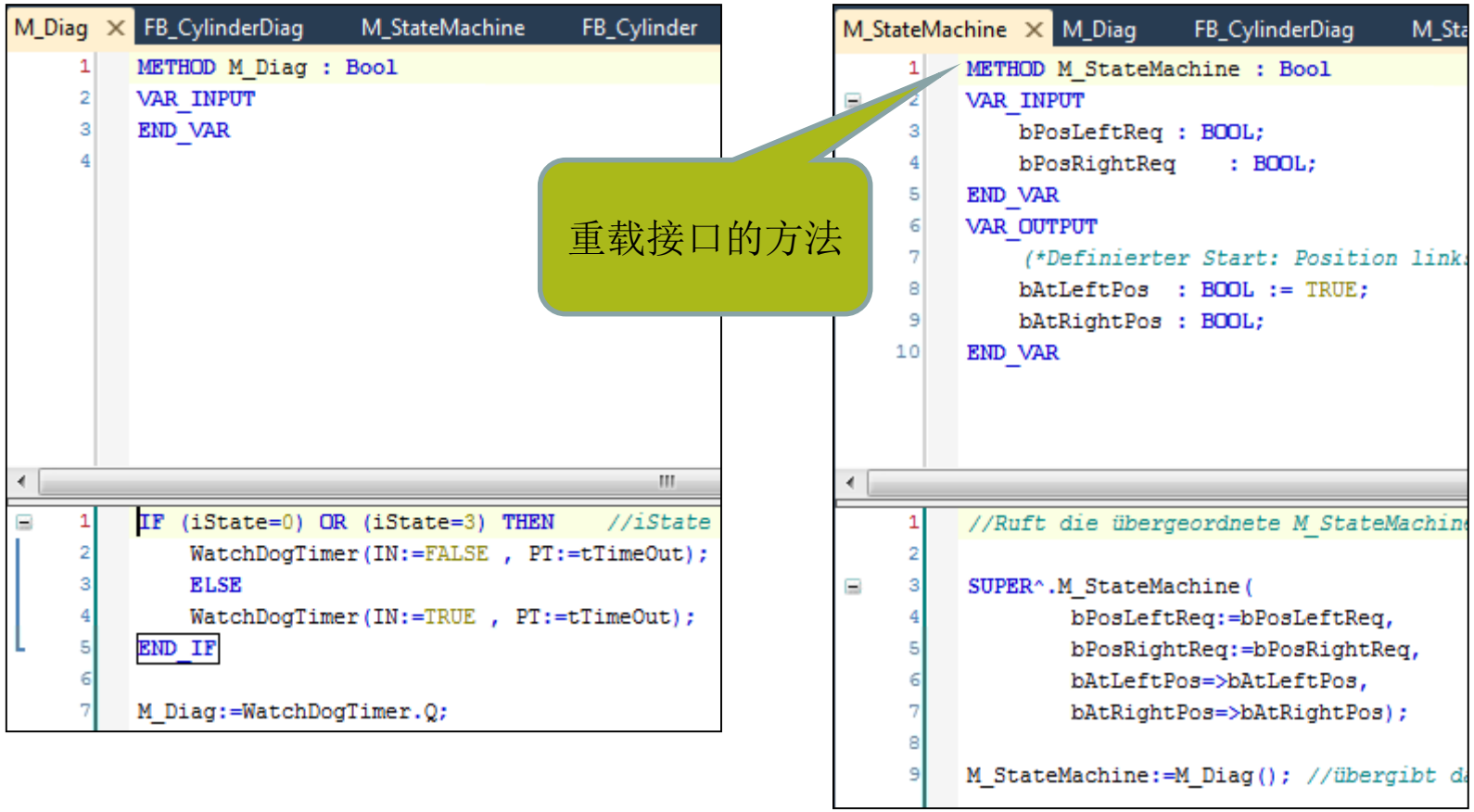
- 创建一个诊断功能块，它继承了基本型 (EXTENDS):



- 重载方法： M_StateMachine from FB_Cylinder

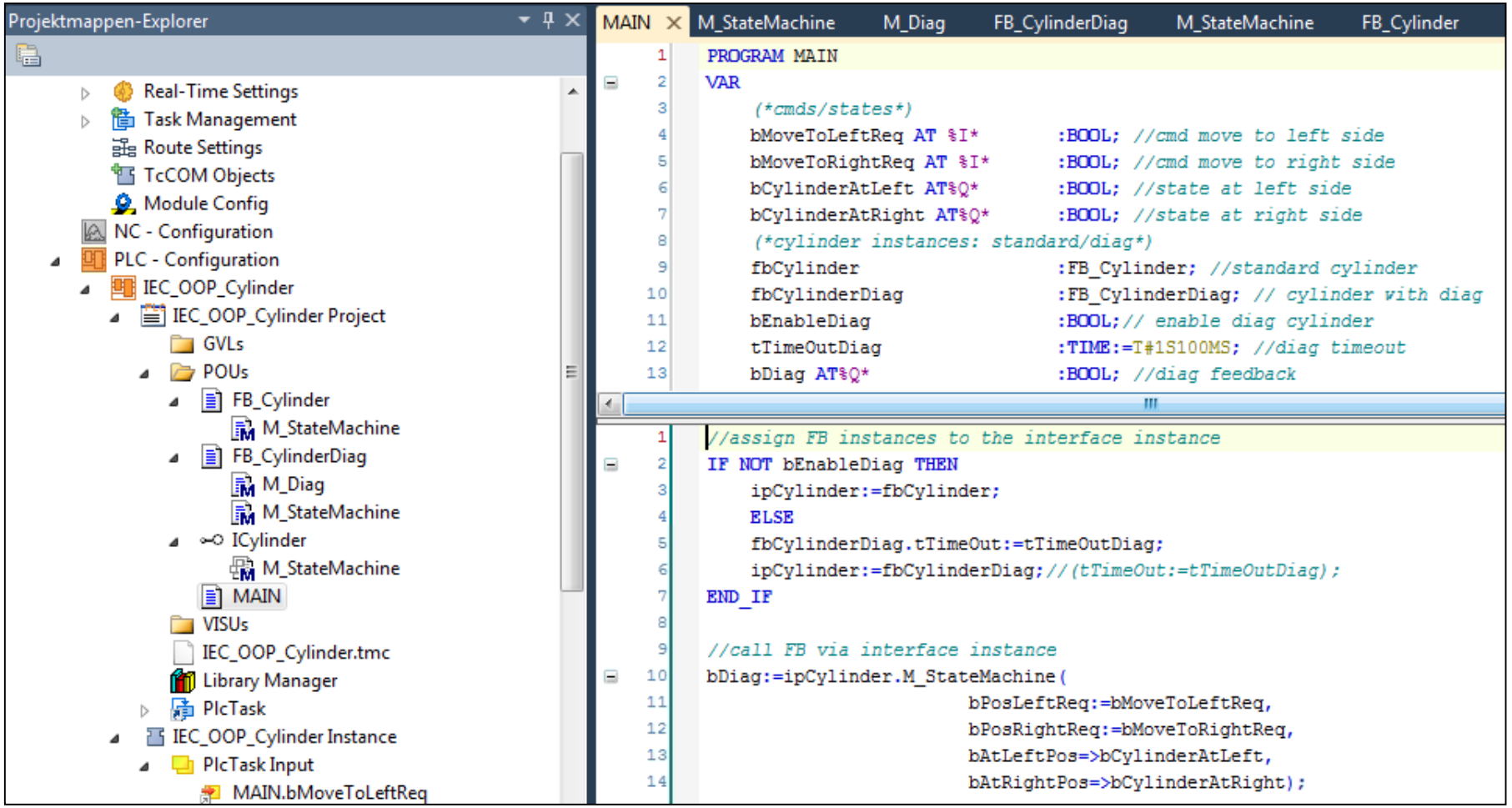
Sample：PLC | 气缸 | 功能块

- 通过特殊类型的诊断，来扩展被继承的方法：



Sample：PLC | 气缸 | 通过接口调用

- 调用功能块 - 类型独立 - 通过接口



语言比较

Language properties	2nd Edition IEC 61131-3	3rd Edition IEC 61131-3	C++	Java	C#
Multilingual capability	+	+	-	-	-
OOP/procedural mixed	-	+	+	-	-
Classes	~ (FB)	+	+	+	+
Methods	~ (Actions)	+	+	+	+
Interfaces	-	+	-	+	+
Partially abstract classes	-	-	+	+	+
Polymorphism	-	+	+/-	+	+
Reference semantics	-	+ (Interfaces)	-	+	+
Constructor/Destructor	-	+	+	+	+
Properties	-	+	-	-	+
Visibility	~ (Variables)	~ (Variablen)	+	+	+
Dyn. memory (‘new’)	-	- (in TC3)	+	+	+

联系我们

BECKHOFF

德国倍福自动化有限公司

上海总部
上海市汶水路299弄9号（市北智汇园）

电话：　021-6631 2666
传真：　021-6631 5696
E-Mail:　support@beckhoff.com.cn
Web:　www.beckhoff. com.cn
虚拟学院：https://tr.beckhoff.com.cn

扫一扫，关注倍福官方微信！