SIEMENS

SIMATIC HMI

WinCC V7.0 SP1
MDM - WinCC： 工具（智能工具、用户归档、界面）

系统手册

在线帮助的打印

11/2008

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ **危险**
表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ **警告**
表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ **小心**
带有警告三角，表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

小心
不带警告三角，表示如果不采取相应的小心措施，可能导致财产损失。

注意
表示如果不注意相应的提示，可能会出现不希望的结果或状态。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ **警告**
Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是西门子股份有限公司的注册商标。标签中的其他符号可能是一些其他商标，这是出于保护所有者权利的 目地由第三方使用而特别标示的。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
德国

℗ 08/2009

目录

智能工具

1

1 资源

1.1 概述

内容

智能工具是使用 WinCC 进行工作时有用的程序集合。

它包括下列程序和文件：

- 变量模拟器
- 变量导出/导入
- 动态向导编辑器
- 文档查看器
- WinCC 交叉索引助手
- OnlOff 和 OnlOn
- 通讯组态器
- WinCC 组态工具
- WinCC 归档组态工具

说明

智能工具是辅助工具。请记住，它们可能会影响 WinCC 的工作方式，例如影响其运行系统行为和需要的内存空间。

就用户友好性和功能性而言，将不必应用与 WinCC 基本软件相同的标准。

参见

WinCC 通讯组态器 (页 123)

变量导出/导入 (页 12)

变量模拟器 (页 21)

动态向导编辑器 (页 26)

WinCC 文档阅读器 (页 110)

WinCC 交叉索引助手 (页 113)

1.2 变量导出/导入

1.2.1 变量导出/导入

简述

程序将来自打开项目的所有连接、所有数据结构和所有变量导出到相应的 ASCII 文件。 然后可以将它们再导入第二个项目。 ASCII 格式允许数据在重新导入之前由电子表格程序进行处理。

1.2.2 安装导出/导入变量

简介

可以用两种不同的方法安装“变量导出/导入”：

只有在 WinCC 也已经被安装的情况下，此程序才有使用的意义。

要求

在 WinCC 多用户项目中，智能工具“变量导出/导入”不能用于不带自己的项目的客户机。

步骤

1. WinCC 安装期间，从“程序”对话框选择“完整的 WinCC V7.0”。
   WinCC 将与智能工具、WinCC 组态工具和 WinCC 归档组态工具一起安装。

“变量导出/导入”应用程序通过选择“SIMATIC”>“WinCC”>“工具”启动。

可选步骤

也可以从 WinCC DVD 安装“变量导出/导入”应用程序。

1. 切换到 WinCC DVD 目录“InstData\WinCC\setup\Products\SC_SMARTTOOLS”。
2. 双击 setup.exe。
3. 从“组件”对话框选择“VarExpImp”。
4. 单击“继续”。按照对话框中的指示进行操作。

1.2.3 操作

导出

1. 首先启动 WinCC 并打开想要从其中导出变量的项目。 启动“VAR_EXIM.EXE”。
2. 选择想要导出到其中的文件的路径和名称。 开头只需要不具有扩展名的文件名称。
3. 选择“导出”模式。
4. 按下“执行”。 确认消息框中的信息。
5. 一直等到状态栏中显示“结束导出/导入”。
6. 使用相应的按钮“tag”（变量）、“con”（连接）、“dex”（结构）和“diag”（记录册），可以查看创建的文件。

空组不导出。

下划线（_）为命名保留。 这就是文件名称中决不能包含下划线的原因。

导入

1. 首先启动 WinCC 并打开想要导入变量到其中的项目。
2. 将要导入连接到其中的所有通道驱动程序必须在项目中都可用。 如果需要，将缺少的驱动程序添加到项目中。
3. 启动“VAR_EXIM.EXE”。
4. 选择想要从中导入的文件的路径和名称。 开头只需要不具有扩展名的文件名称。 使用选择对话框时，单击三个导出文件中的一个。
5. 选择“导入”或“导入重写”模式。 在“导入重写”模式中，目标项目中已存在的变量使用相同名称的导入变量进行重写。 在“导入”模式中，一条消息将写到日志文件中，目标项目中的变量保持不变。

6. 按下“执行”。 确认消息框中的信息。
7. 一直等到状态栏中显示“结束导出/导入”。
8. 在 WinCC 变量管理器中查看创建的数据。

当 WinCC 运行系统启用时，两种模式的导入都不能进行。

下面各节描述变量导出/导入的技术细节。 但是，对于标准情况（目标计算机与用于导出的 WinCC 系统的系统组态相同）不需要这些信息。 只有在想要通过 ASCII 文件创建新变量或修改已存在的变量的情况下，才需要 WinCC 变量结构的知识。

1.2.4 变量导出/导入工具

“变量导出/导入”工具是以 WinCC API 为基础的独立的应用程序。 该工具可以用来将项目的所有 WinCC 变量导出到 ASCII 文件，将变量导入到第二个项目。 这样的话，将创建三个文件。

- [*名称*]_*cex.csv*，用于逻辑连接
- [*名称*]_*dex.csv*，用于结构描述
- [*名称*]_*vex.csv*，用于变量描述

这些文件中都生成一个标题，用来通知有关创建的数据。 在导入期间，这三个文件将被自动读入。

文件 [*名称*]_*cex.csv* 首先导入，因为只有相关连接已存在时才能创建变量。
然后是在 [*名称*]_*dex.csv* 文件中定义的数据结构。 这些都是在创建此类型的变量前必须知道的自定义数据类型。
此后，变量定义将从 [*名称*]_*vex.csv* 文件读入。

变量组不能独立于变量创建。 如果某个组不存在，将会与变量一起自动进行创建。 这就是导出时不生成任何组文件的原因。 如果在不包含任何变量的项目中定义了组，则不导出这些组。

创建变量时，定义变量在自动化系统中的物理位置的寻址与其它要素一样进行组态。 此地址取决于连接通道和所连接的自动化系统。 在通过 WinCC 项目管理器组态时，为用户提供了一个依赖于通道的输入画面。 编辑“变量导出/导入”的导入文件时，必须考虑这些特征。
在任何情况下，只有当第二个 WinCC 项目的组态与被导出数据的系统的组态一样时，被导出的变量才可以导入到此项目。 将数据导入到项目时，可能已经存在的变量必须为已知。 如果需要，可能必须手动改正那些地址。
另一方面，在某些情况下，可能从具有另一种通道组态的系统导入变量。 此类过程成功与否取决于通道和 AS！

但是，导出的结构类型不依赖于硬件。 所有通道或连接特定的参数（如连接、组名称或地址信息）将不予考虑。 只有在创建结构变量时才定义这些参数。

关于导出 szSpecific 中的地址字符串的注意要点

创建变量时，地址字符串由通道指定的软件检查。 这些通道 DLL 要求确定的语法，此语法不得根据特定国家/地区进行改变。 因此，导出时地址字符串将置于附加的“ ”内，在导入时这些插入成分将被删除。 这样可以保证工具（如 Excel）不会修改地址信息（列表分隔符问题）。

1.2.5 连接

只有当相应的通道驱动程序已经组态的情况下才能导入连接。 此外，文件 [*名称*]_*cex.csv* 中的参数必须适合于组态的通道驱动程序。

如果目标计算机使用的通道与产生导出数据的源系统不一样，必须先对导出的数据进行此修改。

确定所需要的连接数据的简单步骤：
组态目标计算机上的所有连接并开始导出过程。 这样就允许您从文件 [*名称*]_*cex.csv* 获取目标计算机的参数。

逻辑连接实例：

#Conname	单位	公共部分	特定部分	标记
VerbAS1	工业以太网	常规	""	0
VerbAS2	Profibus FMS	常规	""	0

第二行显示来自 Simatic S7 Protocol Suite 的工业以太网驱动程序的逻辑连接。
第三行包含 Profibus FMS 驱动程序的逻辑连接。

在 #Conname 项下给出逻辑连接的名称。 在导出文件中，可以在变量数据中找到此逻辑连接的名称。 此逻辑连接用于与自动化系统进行通讯以在过程中访问外部变量。

1.2.6 文件结构

文件 [*名称*]_*dex.csv* 的结构：

#结构名称	类型 ID	创建者 ID	项目路径	
Control_1	1001	2500	C:\Testdaten \Proj.mcp	
#Varname	C.Vartype	C.CreatorID	C.VarLength	等等
NewTag1	...	...	...	...
NewTag2	...	...	...	...
NewTag3	...	...	...	...
等等	...	...	...	...

#:	注释字符
#Structure Name：	下一行包含有指定结构参数的文件结构名称。 项目路径的表示只用于记录从哪个项目导出数据。 数据自动导入到当前打开的项目。
#Varname：	下行包含文件结构的元素，直到检测到新的 #Structure Name 行或直到不再有已定义的行。 一行包含定义一个变量所需的所有参数。

对 WinCC API 用户的注释：

第一行的列标题显示相应 API 调用文件结构中的参数名称。 然后数据可以在 Excel 表中得到清楚的解释。
如果名称由一个字母跟着一个点组成，这样做是为了方便 API 调用的分配。

实例：

C.nnnnnn	包含在 Common 子结构中
P.nnnnnn	包含在 Protocol 子结构中
L.nnnnnn	包含在 Limits 子结构中
S.nnnnnn	包含在 Scaling 子结构中

1.2.7 变量

变量和连接由常规部分和特殊部分组成。 特殊部分始终由通道 DLL 提供。 即使在组态期间有可能丢失该部分（在激活前必须对其进行定义），其特殊部分已丢失的所有对象在导入期间都将被忽略。 在导出期间，将用“*”替代丢失的部分。

导入自定义类型的变量。

预定义的 WinCC 变量将通过数据类型来识别，其值位于 1 和 18 之间。

用户自定义的结构类型将接收 TypeID（类型 ID）作为数据类型，在创建数据结构期间由数据管理器分配 TypeID。 此 TypeID 大于 1000。

结构类型是由其名称和数据类型来确定的。

数据结构的名称在其导出的计算机上与将要为其导入的计算机上是完全相同的。 但 TypeID 不必相同或将会是不同的。

为了创建一个结构类型的变量，必须给结构名称分配一个 TypeID。

实例：

导出结构类型到文件 [*名称*]_*dex.csv*。

#结构名称	类型 ID	创建者 ID	项目路径		
ExternStr1	1046	0	G:\Testdaten\...		
#Varname	C.Vartype	C.CreatorID	C.VarLength	C.VarProperty	...
Tag1	1	0	2	...	...
Tag2	2	0	1	...	...
Tag3	3	0	1	...	...
Tag4	4	0	2	...	...

导出变量到文件 [*名称*]_*vex.csv*。

#Varname	Conn	组	Spec	标记	CTyp
InstExStr1	VerbLp	GruLp	DB200,DBB10	0	1046
InstExStr1.Tag1	VerbLp	GruLp	DB200,D10.0	0	1
InstExStr1.Tag2	VerbLp	GruLp	DB200,DBB10	0	2

InstExStr1.Tag3	VerbLp	GruLp	DB200,DBB10	0	3
InstExStr1.Tag4	VerbLp	GruLp	DB200,DBB10	0	4

在文件 [*名称*]_*dex.csv* 中，定义了名称为 ExternStr1、TypID 为 1046 的结构类型。

在文件 [*名称*]_*vex.csv* 中，定义了名称为 InstExStr1、类型为 ExternStr1 的结构变量。 通过“CTyp”列中的数值（该列包含有该结构类型的 TypeID 1046），可对“ExternStr1”结构类型进行分配。

欲导入结构变量，结构类型也必须包含在文件 [*名称*]_*dex.csv* 中，且该类型的变量包含在文件 [*名称*]_*vex.csv* 中。

“变量导入/导出”将记录“名称/TypID”分配所确定的结构变量的数量，即使 TypID 在目标计算机上是不同的。

如果希望在导入期间导入类型为“自定义”的变量且不用读取数据结构（例如没有任何文件 [*名称*]_*dex.csv*），那么，该数据结构的目标计算机的 TypID 必须在 csv 文件中手动编辑。 在这种情况下，如上所述，TypID 将通过使用目标计算机的已导出文件 [*名称*]_*dex.csv* 来确定。

变量属性将以十进制数值的形式显示在导出文件 [*名称*]_*vex.csv* 的列 CPro 中。 有下列变量属性：

变量属性	十进制数值	十六进制数值
带有项目范围更新的内部变量	2	2
带有指定计算机更新的内部变量	8194	2002
外部变量	4	4

例如，如果带有指定计算机更新的内部变量将要从 WinCC 中导出，而带有项目范围更新的内部变量将要导入到 WinCC，那么，对于各个变量，列 CPro 中的变量属性值在导出文件中可能要从 8194 变为 2。 随后，保存所修改的导出文件，并导入到 WinCC 中。

1.2.8 附录

□□□

```
---------------

Varname: char szVarName[MAX_DM_VAR_NAME +1];

Conn: char szConnection[MAX_DM_CONNECTION_NAME +3];

Group: char szGroupName[MAX_DM_GROUP_NAME +1];

Spec: char szSpecific[MAX_DM_VAR_SPECIFIC +1];
```

```
Flag: DWORD dwFlags;
```

□□□□

```
Ctyp: DWORD dwVarType; // Variable type
CLen: DWORD dwVarLength; // Variable length
CPro: DWORD dwVarProperty; // Variable property internal/external
CFor: DWORD dwFormat; // Format convertion
```

□□□

```
P1 : BOOL bTopLimitErr; // error in top limit
P2 : BOOL bBottomLimitErr; // error in bottom limit
P3 : BOOL bTransformationErr; // transformation error
P4 : BOOL bWriteErr; // write error
P5 : BOOL bWriteErrApplication;
P6 : BOOL bWriteErrProcess;
```

□□□

```
LF1: double dTopLimit;
LF2: double dBottomLimit;
LF3: double dStartValue;
LF4: double dSubstituteValue;
```

□□□□

```
L1 : BOOL bTopLimit; // use substitute value on top limit
L2 : BOOL bBottomLimit; // use substitute value on bottom limit
L3 : BOOL bStartValue; // use substitute value on start
L4 : BOOL bConnectionErr; // use substitute value on error
connection
L5 : BOOL bTopLimitValid; // value top limit is valid
```

```
L6 : BOOL bBottomLimitValid; // value bottom limit is valid
L7 : BOOL bStartValueValid; // value start is valid
L8 : BOOL bSubstValueValid; // value substitute is valid
```

```
□ Ctyp □□
-------------------------
BIT 1
SBYTE 2
BYTE 3
SWORD 4
WORD 5
SDWORD 6
DWORD 7
FLOAT 8
DOUBLE 9
TEXT_8 10
TEXT_16 11
RAW 12
ARRAY 13
STRUCT 14
BITFIELD_8 15
BITFIELD_16 16
BITFIELD_32 17
TEXTREF 18
```

```
□□□□□□
--------------------------------
```

```
Conname: char szConnection[MAX_DM_CONNECTION_NAME +3];
Unit: char szUnitName [MAX_DM_UNIT_NAME+1];
Common: char szCommon [MAX_DM_CON_COMMON +1]
Specific: char szSpecific [MAX_DM_CON_SPECIFIC +1] ;
```

1.3 变量模拟器

1.3.1 变量模拟器

简述

变量模拟器用来模拟内部变量和过程变量。

变量模拟器的一个典型应用领域就是，例如，在不连接过程外围设备或连接了过程外围设备但过程没有运行的情况下，对组态进行检测。

在没有已连接的过程时，可能只模拟内部变量。

对于已连接的过程外围设备，过程变量的值将由**变量模拟器**直接提供。 这样可以使用户用原有的硬件对 HMI 系统进行功能测试。

变量值的刷新时间为 1 秒。 只有在功能激活或项目文件夹改变时，所作的修改才能生效。

最多可以组态 300 个变量。

变量模拟器的另一种可能应用是执行项目演示。
HMI 系统的展示经常没有系统连接。 在这些情况下，模拟器将控制内部变量。

变量模拟器的详细描述参见相应的在线帮助。

⚠ 警告

变量模拟器把过程值写入到所连接的自动化系统。 这意味着必须考虑所连接过程外围设备的可能响应。

1.3.2 使用变量模拟器

变量模拟器的一个典型应用领域就是，例如，在不连接过程外围设备或连接了过程外围设备但过程没有运行的情况下，对组态进行检测。

模拟没有过程外围设备时的过程变量

在没有已连接的过程时，可能只模拟内部变量。 为了对过程进行离线模拟，建议按下列步骤进行操作：

1. 通过复制项目文件夹来创建一份自己的项目备份并将其重命名（例如 xxx_sim）。 使用该备份副本作为测试对象。 并用此项目副本来打开 WinCC。
2. 使用“剪切”和“粘贴”功能将所要模拟的变量添加给内部变量。 **切勿**使用“复制”和“粘贴”，否则，WinCC 项目管理器将自动生成变量名称的扩展名，以确保变量名称在项目中是唯一的。对于已声明为内部变量的变量，其地址信息将因此而丢失。
3. 这样，就可以在变量模拟器的帮助下为这些变量提供数值。
4. 测试阶段结束后就可以继续执行原项目。

模拟具有已连接过程外围设备时的过程变量

对于已连接的过程外围设备，过程变量的值可能由变量模拟器直接提供。 这样可以使用户用原有的硬件对 HMI 系统进行功能测试，例如：

- 检查限制值等级、消息输出。
- 测试报警、警告、出错消息的连续性，并检查状态显示。
- 预置、读取和修改数字量和模拟量的输入和输出。
- 报警模拟。

1.3.3 模拟器的函数

简介

模拟器为组态器提供了六种不同的函数。 这些函数可以给已组态的对象提供实际值。

为了测试不同情况，模拟器提供 6 种函数。 每个变量都可以分配到这 6 种函数中的一种。

正弦

作为非线性周期性函数。

振荡

用于模拟参考变量的跳转。

随机数

“随机数”函数为用户提供随机产生的数值。

自增

向上计数器，达到最大值后又从最小值开始。

自减

向下计数器，达到最小值后又从最大值开始。

滚动条

允许用户设置固定值的滚动条。

1.3.4 安装模拟器

变量模拟器可以用两种不同的方法安装：

步骤

1. WinCC 安装期间，从“程序”对话框选择“完整的 WinCC V7.0”。
   WinCC 将与智能工具、WinCC 组态工具和 WinCC 归档组态工具一起安装。

通过选择“SIMATIC”>“WinCC”>“工具”启动变量模拟器。

可选步骤

也可以从 WinCC DVD 安装变量模拟器应用程序。

1. 切换到 WinCC DVD 目录“InstData\WinCC\setup\Products\SC_SMARTTOOLS”。
2. 双击 setup.exe。
3. 在“组件”对话框中选择条目“WinCC 变量模拟器”。
4. 单击“继续”。按照对话框中的指示进行操作。

启动模拟器

模拟器 Simulation.exe 可以通过 Windows 资源管理器启动，也可以将它输入 WinCC 项目管理器的启动列表，由此每当项目激活时它将自动启动。

要先在 WinCC 项目管理器中激活一个项目后模拟器才能正常运行。如果在项目启动列表中已将模拟器添加进去，那么将自动满足上述条件。

1.3.5 添加/删除变量

添加新变量

使用菜单命令“编辑/新建变量”，可以将变量添加到模拟中。 为此，将调用 WinCC 项目管理器的变量选择对话框，在那里可以选择激活项目中所需的变量。 如果要创建新变量，也可以在变量选择对话框中进行。 通过单击“确定”按钮确认选择，先前选择的变量即输入到模拟器的“属性...”标签中。 可以在此指定如何修改变量值。

为了模拟器最终将其接收，在添加下一个变量之前，必须点击变量标签控件。

组态的变量模拟可以保存在扩展名为“sim”的组态文件中。

删除变量

如果要从模拟器列表中删除变量，则必须通过单击菜单条目“编辑/删除变量”将其选中并删除。随后从要模拟的变量列表中删除所选择的变量，无需使用确认对话框。

1.3.6 函数的参数分配

可以为每个变量单独设置函数的参数。

正弦波

对于正弦函数，数值范围可以使用**振幅**参数设置。

数值范围的零点可以用**偏移量**定义。

周期用**振荡周期**参数设置（设置数值*周期时间）。

振荡

设定值参数用于定义瞬间反应后的保留值。

过冲参数指定当衰减设置为零时偏离设定值多少值。

振荡周期参数定义时间间隔。 到达时间间隔后，振动将再次开始。

随机数

参数**下限**和**上限**指定随机数的间隔。

自增

参数**起始值**和**停止值**指定向上计数器的间隔。

自减

参数**起始值**和**停止值**指定向下计数器的间隔。

滚动条

参数**起始值**和**停止值**定义滚动条的调整范围。

1.3.7 激活/取消激活变量

为了从离线组态平滑过渡到在线组态，变量可以通过自己的复选框单独激活和取消激活。

如果启用变量，其值由模拟器计算并传送到 WinCC 项目管理器。

如果复选框未激活，模拟不会向 WinCC 项目管理器传送任何值。

1.3.8 显示变量

在组态期间，通过在“变量”标签中显示下列信息使组态器便于使用变量控制。

- 当前 WinCC 项目
- 变量的名称
- 分配的函数
- 状态（激活/未激活）
- 当前值

选择变量名称后，其它参数即输入“属性”标签。

1.3.9 装载/保存模拟数据

可以保存模拟数据，使它们在重新启动模拟器时可用。 选择菜单条目“文件/保存”或“文件/另存为...”进行保存。

已保存的模拟组态可以使用菜单命令“文件/打开”装载。

启动模拟器时，自动装载与 WinCC 项目关联的上次使用的组态。

1.3.10 常见问题解答

DM-API 出错，DLL 没有找到

调用与 DLL 连接的模拟器时发生错误是由于在文件 AUTOEXE.BAT 中缺少路径指令。请检查此文件的路径指令中是否有下列条目：

SET PATH = ...;<WinCC 驱动器>:\<WinCC 目录>\bin

例如： SET PATH=C:\SIEMENS\WINCC\BIN;

1.4 动态向导编辑器

1.4.1 动态向导编辑器

简述

动态向导编辑器是一个用于创建自己的动态向导的工具。 动态向导使频繁重现的组态过程自动化。

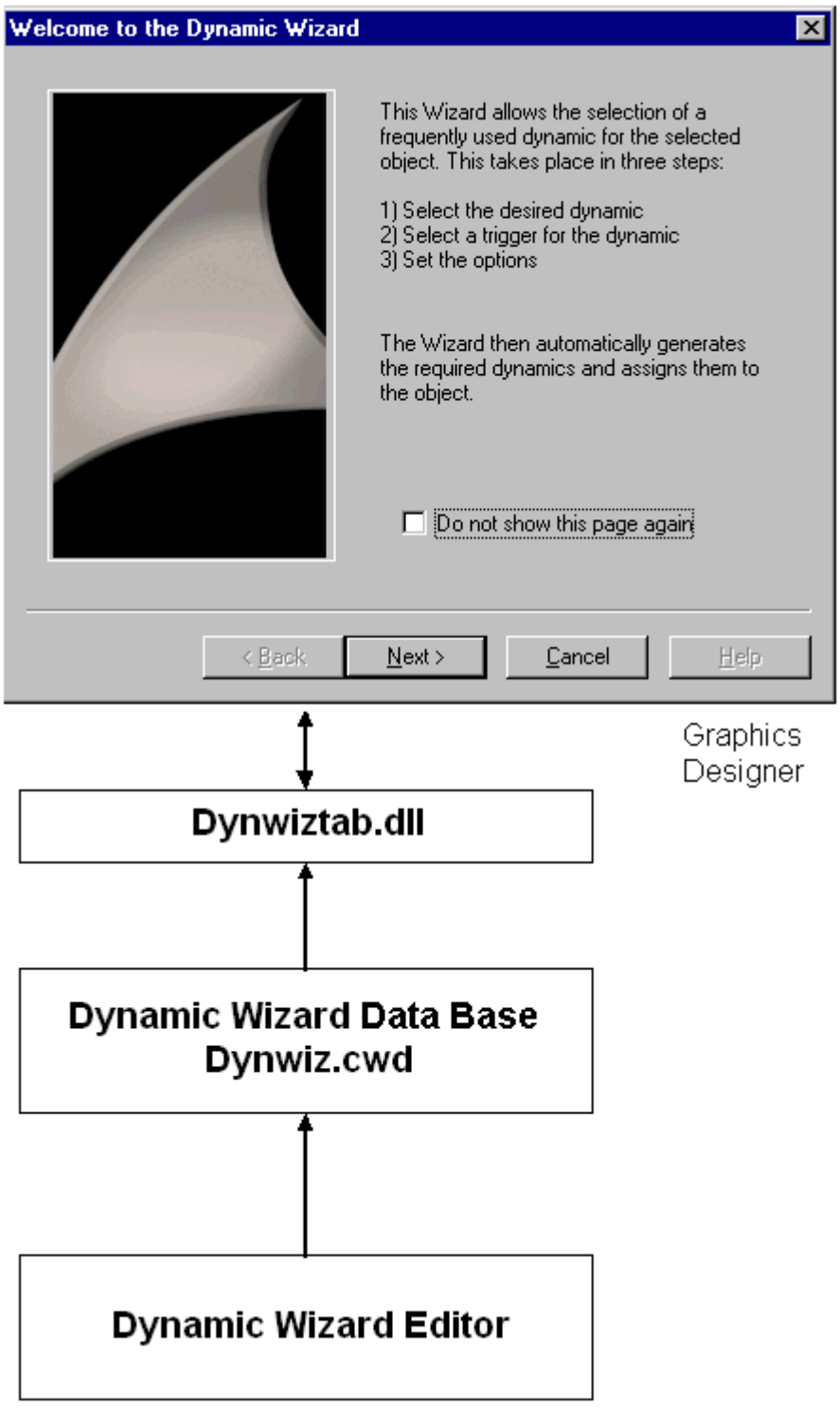
1.4.2 概述

简介

动态向导为图形编辑器带来了附加的功能。 它有助于用户频繁处理再次发生的组态顺序。这将减少组态工作和发生组态错误的风险。

动态向导包含许多动态向导函数。 它已经提供了大量动态向导函数。 并且这些函数可以用您自己创建的函数进一步扩充。

包括一个编辑器，用于创建自己的动态向导函数。 此编辑器是 dynwizedit.exe 程序。



所有的动态向导函数存储在硬盘的数据库（...\WinCC\wscript\Dynwiz.cwd）中。 动态向导具有一个选择和参数化动态向导函数的标准的显示和用户界面。 在选择了动态向导函数后，动态向导将装载它到存储器并且启动它。

动态向导和动态向导函数之间的接口

动态向导和动态向导函数之间的链接是通过动态向导函数中的系统接口完成的。 定义了该接口的结构。 该接口包含动态向导能计算的信息。
该接口的基本内容是：

参考过程函数

该过程函数是动态向导的目标主函数。 它包含动态向导函数为用户提供的“服务”，例如，在图形对象上创建一个动作。

选项列表定义过程函数所需要的参数。 选项列表也定义如何通过对话框的用户界面指定参数。

触发器列表定义了链接到生成的对象的触发器。 触发器列表也定义如何通过对话框的用户界面指定触发器。

参见

触发器列表 (页 43)

选项列表 (页 40)

1.4.3 安装动态向导编辑器

动态向导编辑器可以用两种不同的方法安装：

步骤

1. WinCC 安装期间，从“程序”对话框选择“完整的 WinCC V7.0”。
   WinCC 将与智能工具、WinCC 组态工具和 WinCC 归档组态工具一起安装。

通过选择“SIMATIC”>“WinCC”>“工具”启动动态向导编辑器。

可选步骤

也可以从 WinCC DVD 安装动态向导编辑器。

1. 切换到 WinCC DVD 目录“InstData\WinCC\setup\Products\SC_SMARTTOOLS”。
2. 双击 setup.exe。
3. 在“组件”对话框中选择“动态向导编辑器”。
4. 单击“下一步”。按照对话框中的指示进行操作。

1.4.4 结构

1.4.4.1 结构

动态向导编辑器包含下列元素：

菜单栏

菜单栏包含动态向导编辑器的功能。菜单栏总是可见的。

工具栏

工具栏需要时可使其可见，并可使用鼠标拖动到屏幕的任何地方。

编辑器窗口

当动态向导函数打开进行编辑时或创建新的动态向导函数时，编辑窗口才可见。每个函数都将在自己的编辑窗口中打开。可同时打开多个编辑窗口。

输出窗口

需要时可以使输出窗口可见。它显示“创建 CWD”、“读向导脚本”和“编译脚本”函数的结果。

状态栏

需要时可以使状态栏可见。它提供有关键盘设置的信息和编辑窗口中光标位置的信息。

动态向导

利用动态向导，可使用 C 动作使对象动态化。当执行向导时，预组态的 C 动作和触发器事件被定义，并被传送到对象属性中。

参见

输出窗口 (页 34)

编辑器窗口 (页 32)

工具栏 (页 30)

1.4.4.2 工具栏

简介

工具栏允许更快地执行动作。不必通过菜单进行多次选择直到选中所需要的功能。

图标

图标	描述
	创建新的动态向导函数。
	打开已存在的动态向导函数（*.wnf）。
	保存动态向导函数。
	剪切选择的文本并将它复制到剪贴板上。
	将所选文本复制到剪贴板。
	将剪贴板上的内容粘贴到光标所在的位置。
	打印当前编辑窗口的内容。
	在动态向导编辑器中显示附加信息。
	创建动态向导数据（CWD）。 该功能读取当前设置语言的所有现有向导脚本，并准备在动态向导中处理。 创建的文件可以在 WinCC 安装路径（Installation path \wscripts\dynwiz.cwd）下找到。
	读取向导脚本和使它们在动态向导中可用。

图标	描述
	设置组态向导脚本所使用的语言。 使 WinCC 中支持的语言在此可用，不受安装语言的约束。 改变向导语言不会影响整个系统和组态界面。
	改变对象。 动态向导，也可以在编辑器中用于测试目的，依赖于图形编辑器中对象的不同属性。 可以使用该功能切换到已存在画面中的已有对象，以在编辑器中可以测试新的或已存在的向导脚本。 新设置的对象调整动态向导以便只显示那些适合于该对象的向导脚本。
	显示所选语言的所有动态向导脚本。 另外，可从列表删除存在于该对话框中的向导脚本。
	打开帮助编辑器。
	编译脚本。

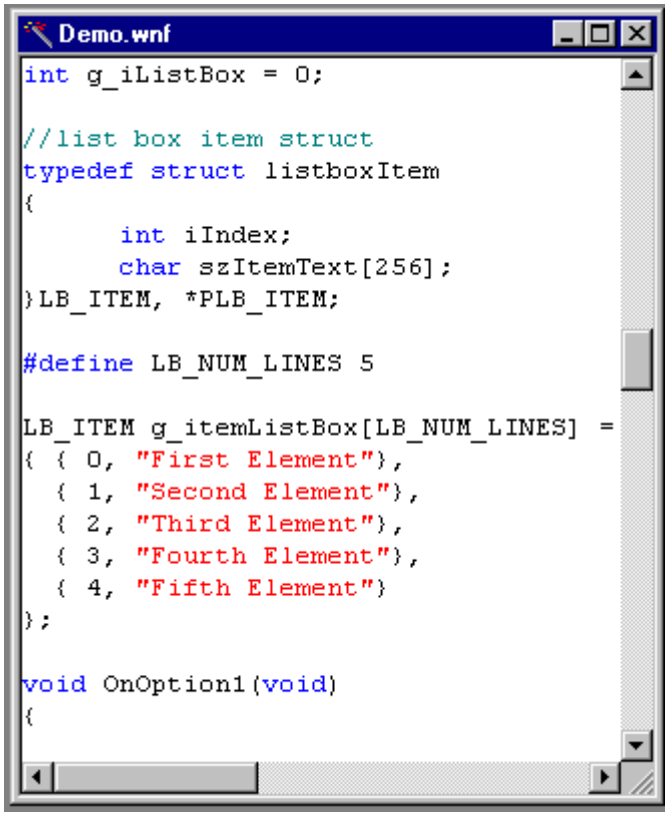
参见

帮助编辑器 (页 33)

1.4.4.3 编辑器窗口

简介

编辑器窗口用于创建和编辑动态向导函数。



颜色编码

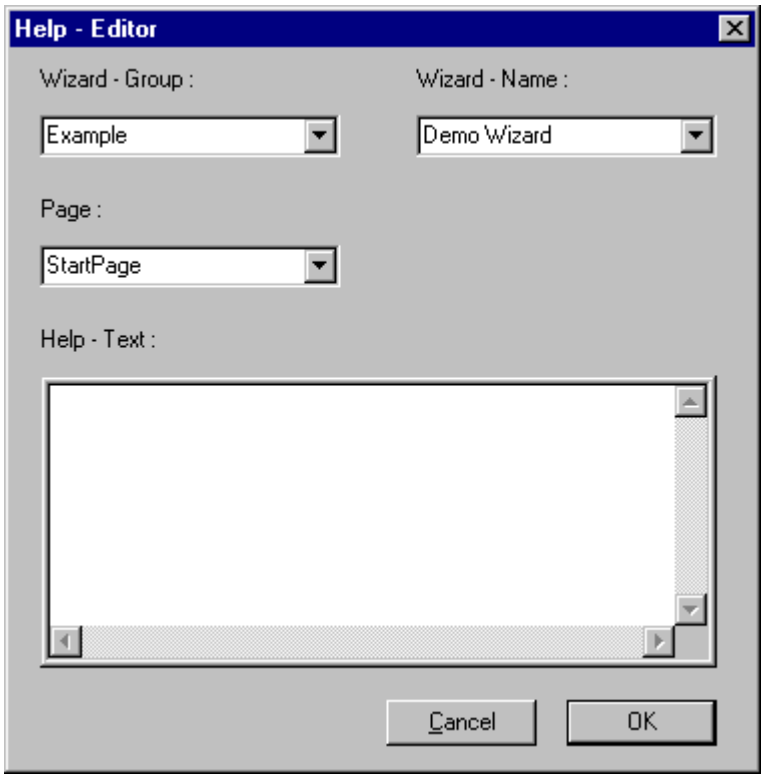
C 代码用以下颜色显示：

颜色	含义	实例
蓝色	关键字	#define、void
绿色	注释	// 这是注释
红色	字符串	“First Element”
黑色	其它 C 代码	OnOption1

1.4.4.4 帮助编辑器

简介

在该对话框中，可为每个通过向导脚本创建的页面输入帮助文本。 仅可以输入已创建动态向导的帮助文本。



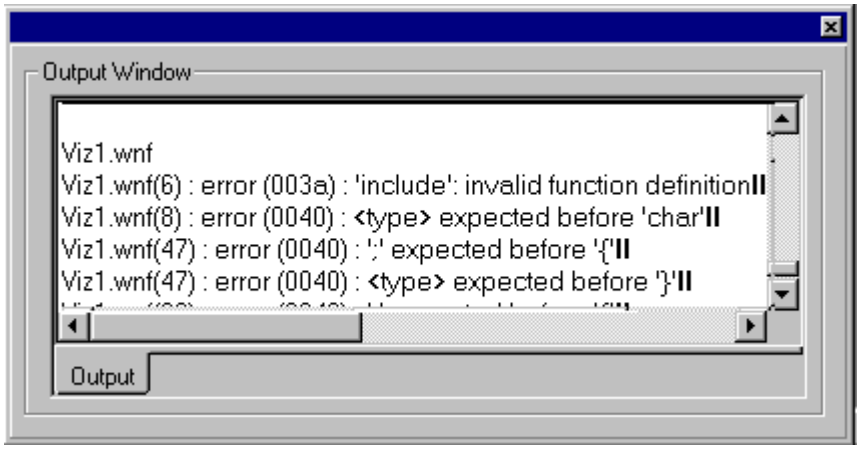
帮助编辑器的元素

元素	描述
向导组	该域用于指定包含动态向导的组（=tab）。
向导名称	该域用于选择为其创建帮助文本的动态向导。
页面	该域用于选择为其创建帮助文本的对话框页面。
帮助文本	在该域中输入帮助文本。

1.4.4.5 输出窗口

简介

输出窗口显示“创建 CWD”、“读向导脚本”和“编译脚本”函数的结果。



输出窗口有助于在脚本中发现错误。

如果脚本中存在错误，将显示以下消息：

...\WinCC\wscripts\wscripts.deu\DemoWiz1.wnf(6):error(003a): 'include': 无效的函数定义

	描述
...\WinCC\wscripts\wscripts.deu\	包含 wnf 文件的目录。
DemoWiz1.wnf(6)	发生错误的文件名和行号
error(003a): 'include': 无效的函数定义	错误号和错误的描述。

1.4.5 动态向导函数的结构

1.4.5.1 动态向导函数的结构

简介

动态向导函数必须具有某种指定的结构。 它对应于所需的组件。

1. 集成头文件和 DLL
2. 与语言相关的定义
3. 向导标记

4. 属性列表
5. 系统接口
6. 全局变量
7. 选项列表
8. 触发器列表
9. 参数分配的显示

参见

参数分配的显示 (页 45)

触发器列表 (页 43)

选项列表 (页 40)

全局变量 (页 40)

系统接口 (页 39)

属性列表 (页 38)

向导标记 (页 37)

与语言相关的定义 (页 36)

集成头文件和 DLL (页 36)

1.4.5.2 动态向导对话框

简介

每一个动态向导选项都有其指定的功能。 但是，由于预定义的功能结构，所有功能都具有相似的顺序和相似的对话框界面。 动态向导对话框由几个对话框页面组成。

- “欢迎使用动态向导”对话框
- “选择触发器”对话框
- “设置选项”对话框
- “完成！”对话框

1.4.5.3 集成头文件和 DLL

简介

头文件包含常数、数据类型、变量和函数的声明。

头文件用 #include 指令集成到函数。 要集成的最重要文件是 dynamic.h 文件，在其中，除了其它内容，还声明了用于设计动态向导界面的函数。

```
//*******************************************************
//**        Integration of Header-Files                **
//*******************************************************
            #include "dynamic.h"
```

DLL 文件（动态链接库）是可执行的例行程序，当程序需要它们执行时可将其装载。

为了能够使用 DLL 文件，它们用 #pragma 指令集成到函数。

```
//*************************************************
//**        Integration of Dlls                  **
//*************************************************
            #pragma code("pdlcsapi.dll")
            #include "pdlcsapi.h"
            #pragma code()
```

在动态向导编辑器中定义下列路径：

WinCC 头文件： ...\WinCC\aplib\

WinCC DLL： ...\WinCC\bin\

如果这些文件存储在另一个目录中，完整的路径需要在 `#include-` 和 `#pragma` 指令中指定。

1.4.5.4 与语言相关的定义

简介

动态向导标准函数以三种语言形式存在：德语、英语和法语。 当在 WinCC 项目管理器中切换语言时，也同时选择了动态向导函数的相应语言版本。

在路径中

..\WinCC\wscripts\wscripts.deu

..\WinCC\wscripts\wscripts.enu

..WinCC\wscripts\wscripts.fra

对于每个向导函数必须存在一个 WNF 文件。

在创建时，所有依赖于语言的定义应该排列在该部分中。 这便于创建其它语言版本。

```
//*************************************************************************//
//                  Language-Dependent Definitions                         //
//*************************************************************************//
//                          German                                         //
//-------------------------------------------------------------------------//
include  "defdeu.h"

char* DynWizGroupName     = "Beispiel";
char* DynWizDynamicName   = "Motor dynamisieren";
char* DynWizToDoOption1   = "Wählen Sie die gewünschte Strukturvariable:";
//-------------------------------------------------------------------------//
//                         Englisch                                        //
//-------------------------------------------------------------------------//
#include  "defenu.h"

char* DynWizGroupName     = "WinCC C-Course";
char* DynWizDynamicName   = "Make a Motor Dynamic";
char* DynWizToDoOption1   = "Select the desired Structure Tag:";
//-------------------------------------------------------------------------//
//                        French                                           //
//-------------------------------------------------------------------------//
#include  "deffra.h"

char* DynWizGroupName     = "Cours de C WinCC";
char* DynWizDynamicName   = "Dynamiser moteur";
char* DynWizToDoOption1   = "Sélectionnez la variable de structure:";
```

1.4.5.5 向导标记

简介

这些标记用于定义动态向导函数应用的组态类型。

```
WIZARD_FLAGS(WIZARD_FLAG_OCX | WIZARD_FLAG_ALL_PROJECT_TYPES)

BEGIN_PROPERTY_SCHEME
END_PROPERTY_SCHEME
```

标记

标记	
WIZARD_FLAG_OCX	用于所有的 OCX 文件
WIZARD_FLAG_ALL_PROJECT_TYPES	用于所有的项目
WIZARD_FLAG_SINGLEUSER_PROJECT	仅用于单用户项目
WIZARD_FLAG_MULTICLIENT_PROJECT	用于客户机项目
WIZARD_FLAG_MULTIUSER_PROJECT	仅用于没有项目数据的客户机

1.4.5.6 属性列表

简介

属性列表定义可以使用动态向导函数的对象类型。 这通过指定对象属性的列表来完成。 如果对象至少具有一个已列出的属性，则动态向导函数可用于该对象。

```
//************************************************
//** Objektauswahl mittels Objekteigenschaften  **
//************************************************

BEGIN_PROPERTY_SCHEME
 {"BackColor", VT_I4},
END_PROPERTY_SCHEME
```

在属性列表中的每个条目由两个参数组成：

- 属性的名称，例如英语版本中的 BackColor。
- WinCC 数据类型

如果使用的是空属性列表，则动态向导函数可应用于所有的对象类型。 无论如何，必须有一个属性列表，即使它是空的。

1.4.5.7 系统接口

简介

系统接口用于定义新的动态向导函数的属性。

```
BEGIN_DYNAMICS
      {
      DynWizGroupName,          // 1. Parameter
      DynWizDynamicName,        // 2. Parameter
      NULL,                     // 3. Parameter
      "logo16.bmp",             // 4. Parameter
      DynWizHelpText,           // 5. Parameter
      {                         // 6. Parameter
      // "OnOption1",
      // "OnOption2",
      NULL
      },
      "OnGenerate",             // 7. Parameter
      "OnShowGenerateInfo",     // 8. Parameter
      {                         // 9. Parameter
      // PREDEFINED_MACRO,
      // {DynWizTrigger1Text, OnTrigger1},
      {NULL,NULL}
      },
      },
END_DYNAMICS
```

参数描述

1. 第一个参数定义显示动态向导函数的标签。
2. 第二个参数定义在其下显示动态向导函数的名称。
3. 第三个参数始终为 NULL。
4. 第四个参数指定用于动态向导函数的图标的名称。
5. 第五个参数是一个更详细地描述动态向导函数功能的帮助文本。
6. 第六个参数是一个为各选项页面创建的功能的名称列表。 该列表将由 NULL 条目终止。 最多可创建五个选项页面。 关于此主题的其它信息请参见“选项列表”。
7. 第七个参数是单击“完成”按钮后调用的过程函数的名称。 该过程函数是动态向导的目标主函数。 它包含动态向导函数为用户提供的“服务”，例如，在图形对象上创建一个动作。
8. 第八个参数是概括选项页面中的设置并在用户单击”完成“按钮之前将其显示给用户的函数的名称。 关于此主题的其它信息请参见“显示参数分配”。
9. 第九个参数是要在触发器页面上显示的触发器的列表。 对于大多数一般应用程序，都具有完成触发器列表的宏。 关于此主题的其它信息请参见“触发器列表”。

1.4.5.8 全局变量

简介

对于要在选项页中设置的每个参数，必须定义一个全局变量。 这样可以确保设置的参数在所有已创建的函数中是已知的并且可以使用。

仅可能通过全局变量在系统函数之间传递数据。 当必须传送触发器和/或选项参数到过程函数时，这总是必需的。

```
//****************************************
//        Definition of Global Tags
//****************************************

char g_Demo_Typ ="Demo"
```

1.4.5.9 选项列表

简介

选项是参数，它对动态向导函数是必须的。 选项不需要触发器。

在系统接口的选项列表中定义了选项。 选项列表包括（对每一个选项）指定选项函数的名称，例如“OnOption1”。

```
BEGIN_DYNAMICS
{
DynWizGroupName,
DynWizDynamicName,
NULL,
"logo16.bmp",
DynWizHelpText,
//*******************************
//          Optionenliste
//*******************************
{
"OnOption1",
"OnOption2",
NULL
},
"OnGenerate",
"OnShowGenerateInfo",
{ // Triggerliste
{ NULL, NULL }
},
},
END_DYNAMICS
```

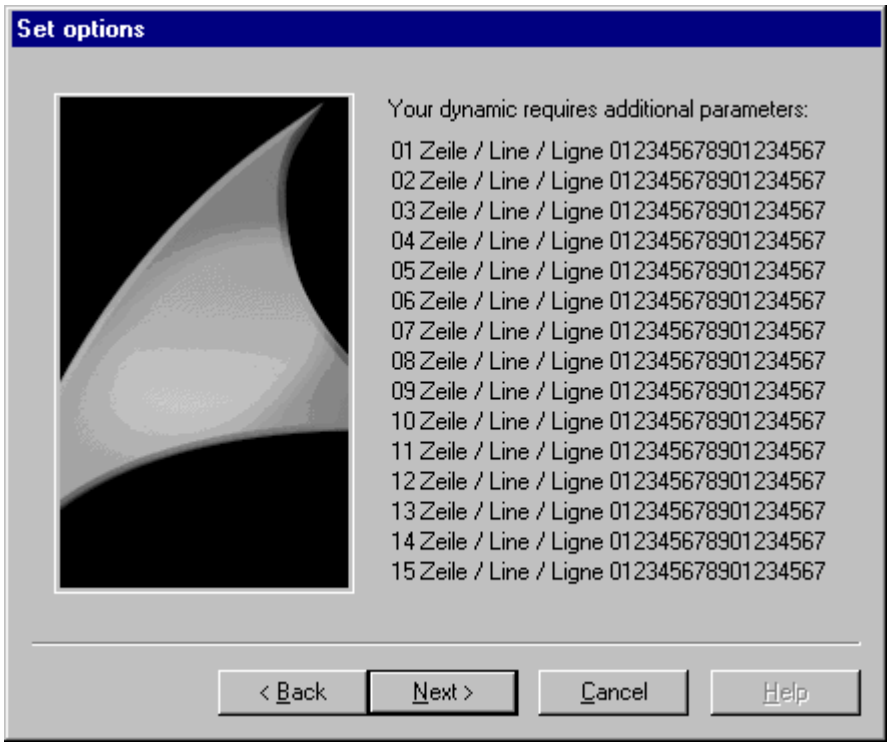
选项列表将由 NULL 指针终止。 列表中最多可以定义五个选项。

选项函数

动态向导根据它们在选项列表中的顺序一个接一个地调用选项函数。 对每一个选项函数都将显示“设置选项”对话框，在其中函数对其特定条目进行编程。

向导系统函数用于动态向导的编程。 有关此主题的更多信息，请参见“向导系统函数”。

在“设置选项”对话框中，存在一个在其中可以排列静态文本、输入域和其它输入 框的已定义区域。



在“设置选项”对话框中，用 1 到 15 行填满整个区域。

相关的选项函数显示如下：

```
//-----------------------------------------------------------------
//    Option-Funktion OnOption1
//-----------------------------------------------------------------

void OnOption1(void)
{
CreateStatic(0, 0,"01 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0, 15,"02 Zeile / Line / Ligne 012345678901234567");
CeateStatic(0, 30,"03 Zeile / Line / Ligne 012345678901234567");
CeateStatic(0, 45,"04 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0, 60,"05 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0, 75,"06 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0, 90,"07 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,105,"08 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,120,"09 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,135,"10 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,150,"11 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,165,"12 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,180,"13 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,195,"14 Zeile / Line / Ligne 012345678901234567");
CreateStatic(0,210,"15 Zeile / Line / Ligne 012345678901234567");
}
```

1.4.5.10 触发器列表

简介

触发器仅在应用于图形对象的动作连接时可用。

在系统接口的触发器列表中定义触发器。 触发器列表包含每个触发器的条目。

```
BEGIN_DYNAMICS
{
DynWizGroupName,
DynWizDynamicName,
NULL,
"logo16.bmp",
DynWizHelpText,
"OnOption1",
"OnOption2",
NULL
},
"OnGenerate",
"OnShowGenerateInfo",
{
//*********************************
//          Trigger  list
//*********************************
{ "Mouse click" , "OnTriggerMC" },
{ "Pressing left mouse key" , "OnTriggerLMDown" },
{ "Releasing left mouse key" , "OnTriggerLMUp" },
{ "Pressing right mouse key" , "OnTriggerRMDown" },
{ "Releasing right mouse key" , "OnTriggerRMUp" },
{ NULL, NULL }
      },
      },
END_DYNAMICS
```

每个条目由两个参数组成。 第一个参数是触发器的名称，它将显示在用户界面上，例如，按鼠标左键。 第二个参数提供相关触发函数的名称。

触发器列表将由 NULL 指针对终止。 可以在列表中至多定义五十个触发器。

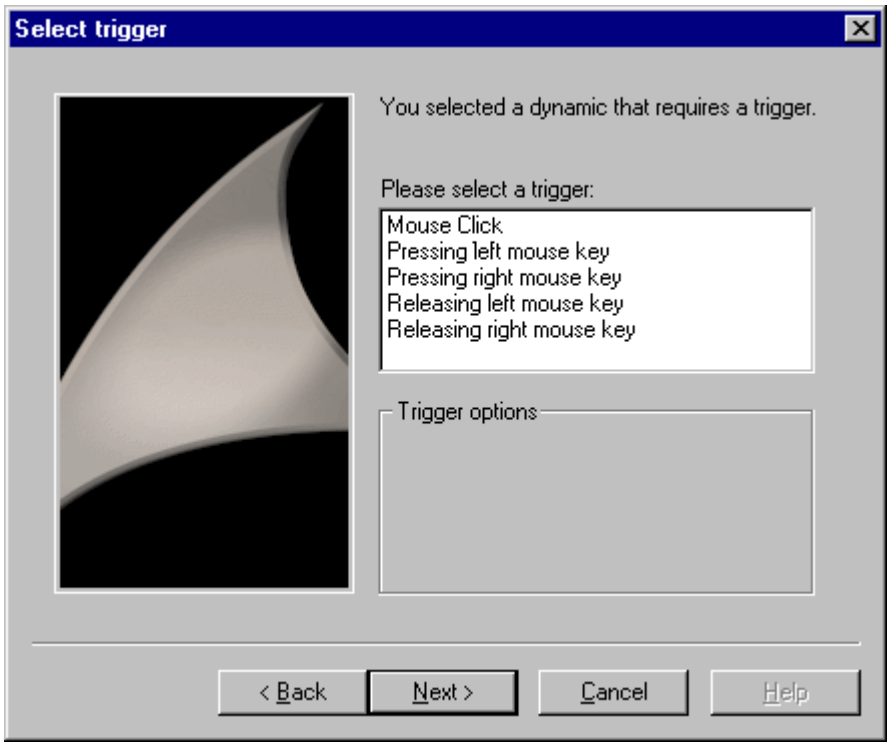
经常使用的触发器具有预定义的宏。

宏	
JCR_TRIGGERS	触发事件 DECLARE_JCR_TRIGGERS 鼠标单击、鼠标左键、鼠标右键
JCR_ZYCL_TRIGGERS	周期性触发 DECLARE_JCR_ZYKL_TRIGGERS 画面周期、窗口周期、根据改变、250 毫秒、500 毫秒、1 秒、2 秒、5 秒、 10 秒、1 分、5 分、用户周期 1、用户周期 2、用户周期 3、用户周期 4、用户周期 5
JCR_ACTION_TRIGGERS	动作触发 DECLARE_JCR_ACTION_TRIGGERS

```
...
...
BEGIN_PROPERTY_SCHEME
END_PROPERTY_SCHEME
BEGIN_DYNAMICS
      {
       "System Functions",
       "Exit WinCC Runtime",
       NULL,
       "logo16.bmp",
       "Exits WinCC Runtime and switches to \r\nthe DESIGN Mode.",
       { NULL, NULL, },
       "OnGenerate",
       "OnShowGenerateInfo",
       {
            JCR_TRIGGERS,
       },
END_DYNAMICS

DECLARE_JCR_TRIGGERS
...
...
```

从触发器列表生成“选择触发器”对话框。 所有的触发器名称显示在一个列表框中，用于选择。



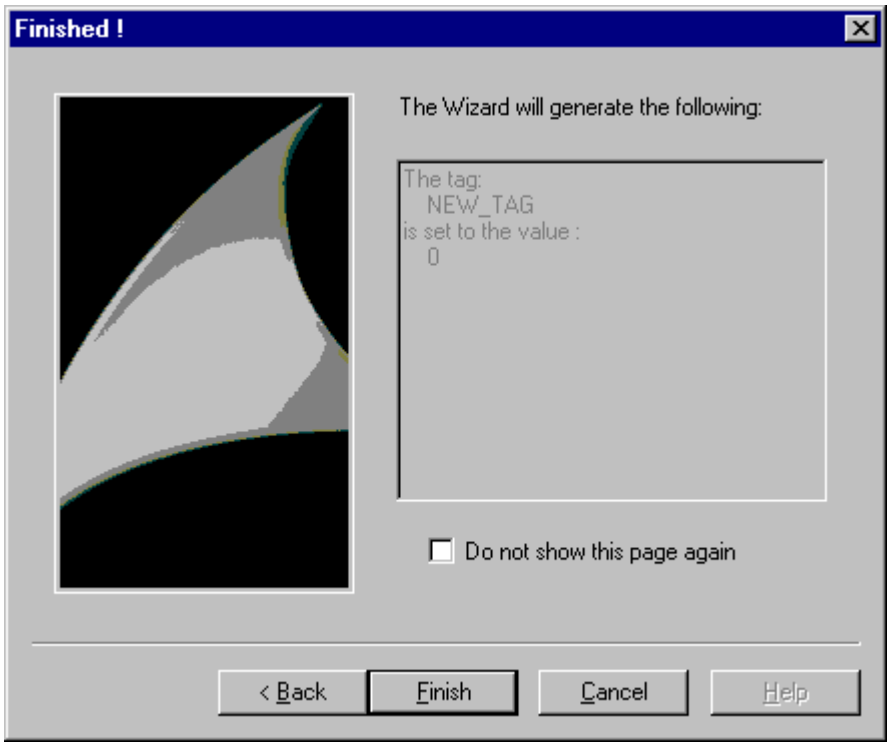
在选择了一个触发器后，动态向导将调用指定的触发函数。

1.4.5.11 参数分配的显示

简介

触发器和选项参数可以在“完成！”对话框中显示。 在该方式中，用户可以再次检查参数，如果需要，将其更正。

Windows SetWindowText 函数可用于在“完成！”页面的显示域中输出文本。显示域是 12 行高。



1.4.5.12 参数输入的向导函数

CreateStatic

简介

在“设置选项”对话框中，静态文本将显示在坐标轴 x、y 上。

语法

HWND CreateStatic (int x, int y, char* "Text")

参数

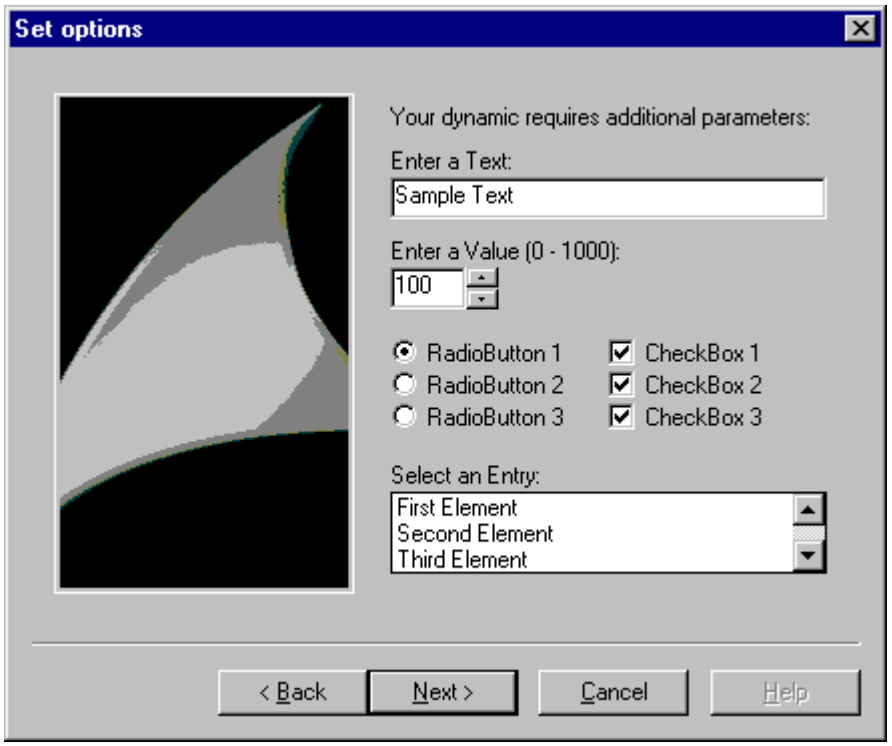
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
char* Text	显示输出的文本。

返回值

	返回值
HWND	返回对象的句柄。

实例

下列来自“Demo.wnf”文件的引用说明该函数的使用。




```
char* DynWizEditStatic = "Enter a text:";

...

..

void OnOption1(void)

{

static BOOL bFirst = TRUE;

HWND hWnd = NULL;

.....
```

```
if (bFirst == TRUE)
{
strcpy(g_szEdit,DynWizEdit);
bFirst = FALSE;
}
//静态文本
CreateStatic(0,5,DynWizEditStatic);
.....
......
}
```

CreateEdit

简介

在“设置选项”对话框中，输入域将显示在坐标轴 x、y 上。 在输入域中可以输入文本。

语法

HWND CreateEdit (int x, int y, char* pText)

参数

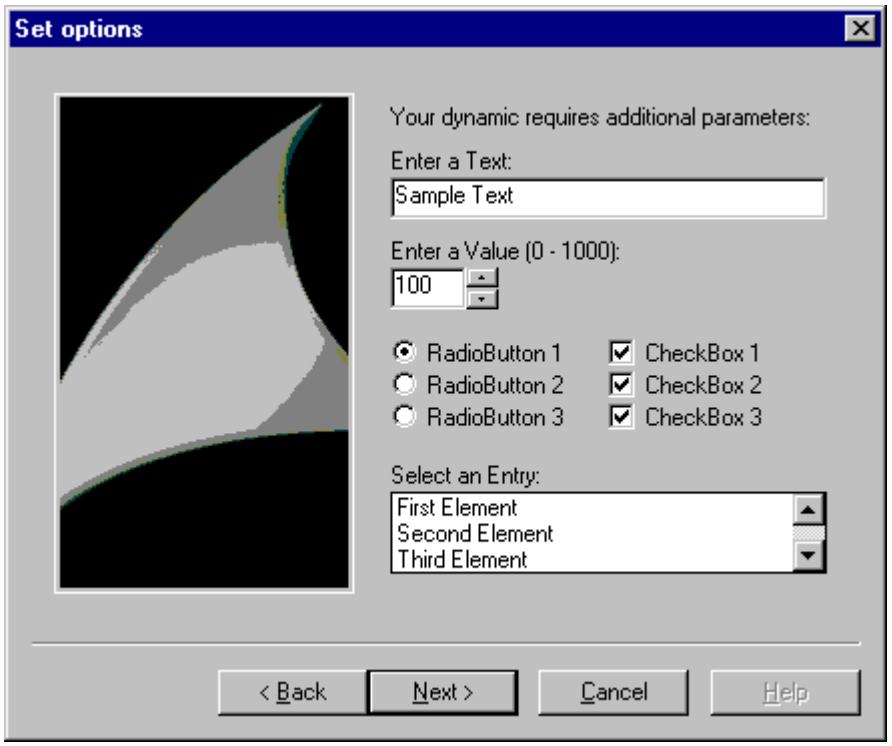
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
char* pText	指针指向一个输入缓冲区。 输入缓冲区可以具有一个预置的数值。 它将在输入域中显示。

返回值

	返回值
HWND	返回对象的句柄
pText	输入缓冲区包含已输入的文本。

实例

下列来自“Demo.wnf”文件的引用说明该函数的使用。
输入域将显示在“演示向导”的“设置选项”对话框中。




```
char* DynWizEditStatic = "Enter a text:";

char* DynWizEdit = "Sample text";

...

..

char g_szEdit[256];

void OnOption1(void)
```

```
{
static BOOL bFirst = TRUE;
HWND hWnd = NULL;
.....
if (bFirst == TRUE)
{
strcpy(g_szEdit,DynWizEdit);
bFirst = FALSE;
}
//输入域的静态文本
CreateStatic(0,5,DynWizEditStatic);
//输入域
hWnd = CreateEdit(0,20,g_szEdit)
GetWindowRect(GetParent(hWnd), &rect);
MoveWindow(hWnd,0,20,(rect.right-rect.left),21,TRUE);
.....
......
}
```

CreateSpinEdit

简介

在“设置选项”对话框中，一个带控件的输入域将显示在坐标轴 x、y 上。

该输入域用于在输入变量中输入整型数值。

语法

HWND CreateSpinEdit (int x, int y, int* pValue, int Min, int Max, int Base)

参数

参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
int* pValue	指针指向整型输入变量。 输入变量可以具有一个预置的默认值。
int Min	输入数值的下限
int Max	输入数值的上限
int Base	输入的数字格式： 10 = 十进制条目 16 = 十六进制条目

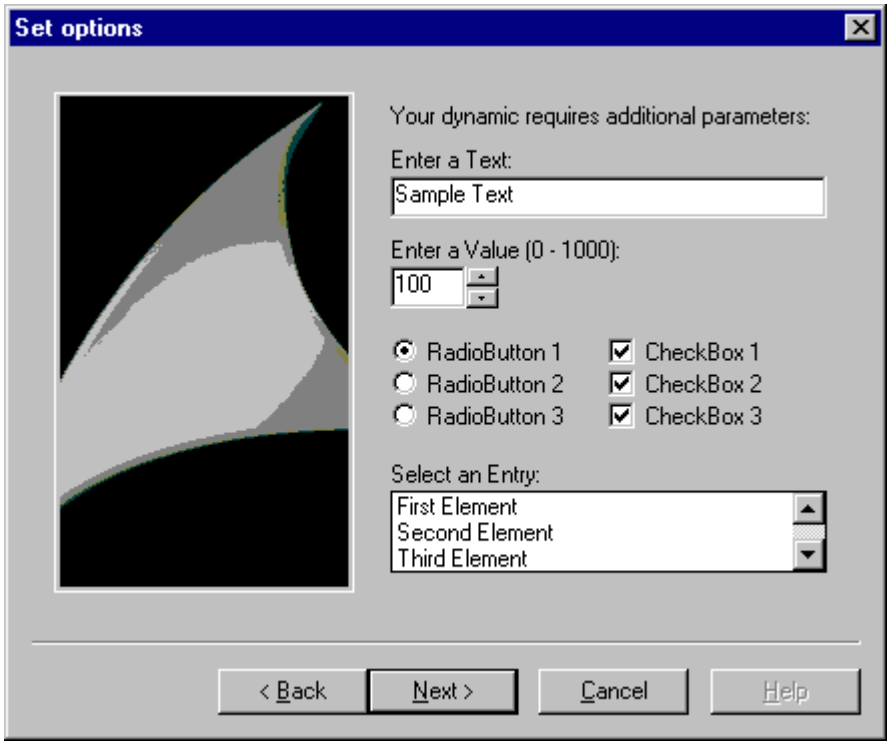
返回值

	返回值
HWND	返回对象的句柄。
pValue	输入变量包含已输入的值。

实例

下列来自“Demo.wnf”文件的引用说明该函数的使用。

带控件的输入域将显示在“演示向导”的“设置选项”对话框中。 可以在该域中设置一个 0 和 1000 之间的数值。




```
char* DynWizSpinStatic= "Enter a value (0 - 1000):";
char* DynWizEdit = "Sample text";
...
...
char g_szEdit[256];
void OnOption1(void)
{
static BOOL bFirst = TRUE;
HWND hWnd = NULL;
.....
if (bFirst == TRUE)
{
strcpy(g_szEdit,DynWizEdit);
```

```
bFirst = FALSE;
}
...
...
```

//带控件的输入域的静态文本

```
CreateStatic(0,50,DynWizSpinStatic);
```

...

//带控件的输入域

```
hWnd = CreateSpinEdit(0,65,&g_iSpinEdit,0,1000,10);
```

MoveWindow(hWnd,0,65,(rect.right-rect.left)/4,21,TRUE);

```
...
...
}
```

CreateListBox

简介

在“设置选项”对话框中，选择域将显示在坐标轴 x、y 上。 选择域允许列出几个条目。 单击鼠标，可以选择一个条目。

语法

HWND CreateListbox (int X, int Y, char* Headline, int NumLines, int* pSelect)

参数

参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
char* Headline	选择域的标题。

参数	描述
int NumLines	选择域中行的数量 必须指定下列内容： NumLines = 行数 + 1 (1 =< NumLines = “16)
int* pSelect	指针指向结果变量

返回值

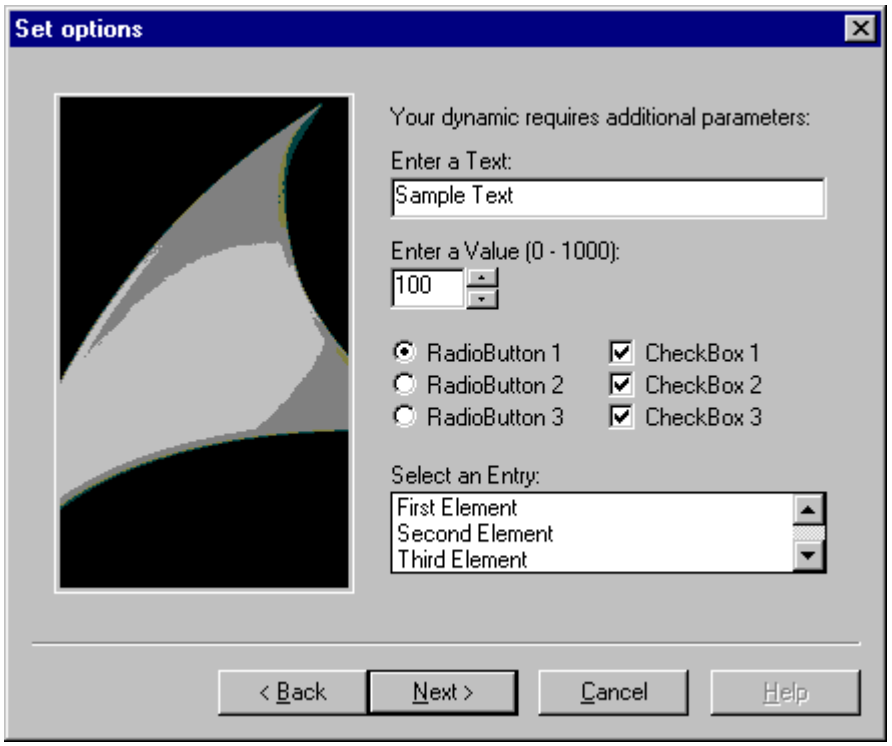
	返回值
HWND	返回对象的句柄
pSelect	所选条目数。 数目是在列表中的索引（从 0 开始）。

实例

下列来自“Demo.wnf”文件的引用说明了该函数的使用。 选择域将显示在“演示向导”的“设置选项”对话框中。 选择域的显示可达到三行。 由于条目多于三个，将显示滚动条。

说明

用“CreateListbox”函数，只能创建选择域本身。 必须用“SendMessage”函数输入行的内容。




```
char* DynWizListStatic= "Select an entry:";
...
int g_iListBox = 0;
//在选择域中的元素类型定义
typedef struct listboxItem
{
int iIndex;
char szItemText[256];
}LB_ITEM, *PLB_ITEM;

#define LB_NUM_LINES 5

LB_ITEM g_itemListBox[LB_NUM_LINES] =
{
{ 0, "First Element"},
```

```
{ 1, "Second Element"},
{ 2, "Third Element"},
{ 3, "Fourth Element"},
{ 4, "Fifth Element"}
};


void OnOption1(void)
{
static BOOL bFirst = TRUE;
HWND hWnd = NULL;
.....
if (bFirst == TRUE)
{
strcpy(g_szEdit,DynWizEdit);
bFirst = FALSE;
}
...
...
```

//选择域的静态文本

```
CreateStatic(0,162,DynWizListStatic);
...
```

//选择域

```
hWnd = CreateListbox(0,177,"Headline",LB_NUM_LINES,&g_iListBox);
```

MoveWindow(hWnd,0,177,(rect.right-rect.left),50,TRUE);

//用“CreateListbox”函数，只能创建框本身。 //必须用“SendMessage”函数输入行的内容。

```
for (i=0; i<LB_NUM_LINES; i++)
{
SendMessage(hWnd,LB_INSERTSTRING,(WPARAM)-1,
(LPARAM)g_itemListBox[i].szItemText);
```

```
}
}
```

CreateCheckBox

简介

在“设置选项”对话框中，一个复选框将显示在坐标轴 x、y 上。 该复选框用于启用选项。 多个复选框可用在单个对话框上。

语法

HWND CreateCheckBox (int x, int y, char* Text, BOOL* pSelect)

参数

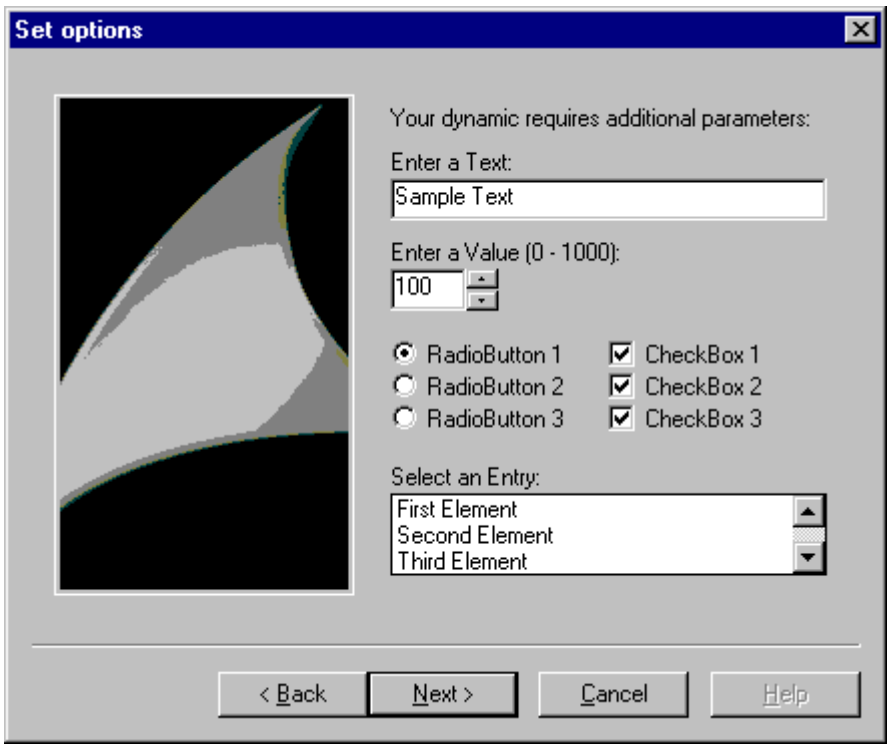
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
char* Text	文本在复选框右边显示。
BOOL* pSelect	指针指向结果变量。 变量名可以预置为默认值 (True/False)。

返回值

	返回值
HWND	返回对象的句柄
pSelect	激活状态 FALSE = 没有激活 TRUE = 激活

实例

下列来自“Demo.wnf”文件的引用说明了该函数的使用。 三个复选框将显示在“演示向导”的“设置选项”对话框中，每个复选框代表一个选项。 每个选项可以分别激活。




```
BOOL g_bCheck1 = TRUE;

BOOL g_bCheck2 = TRUE;

BOOL g_bCheck3 = TRUE;


void OnOption1(void)

{

static BOOL bFirst = TRUE;

HWND hWnd = NULL;

.....

if (bFirst == TRUE)

{

...
```

```
}
...
...
//复选框
iMid = (rect.right-rect.left)/2 ;

CreateCheckBox(iMid,100,"CheckBox 1",&g_bCheck1);
CreateCheckBox(iMid,116,"CheckBox 2",&g_bCheck2);
CreateCheckBox(iMid,132,"CheckBox 3",&g_bCheck3
}
```

CreateFrame

简介

矩形框将在“设置选项”对话框中显示。 框架的左上角由坐标轴 x、y 定义。 矩形框的右下角与选项对话框的右下角相同。

语法

HWND CreateFrame (int x, int y, char* Title)

参数

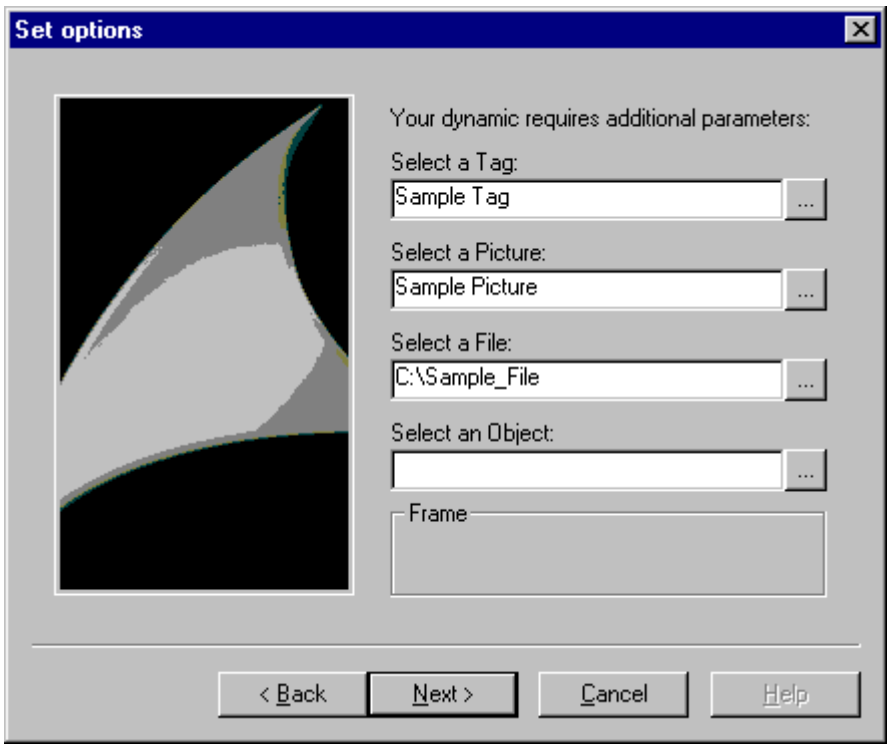
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 Y 坐标轴的数值。
char* Title	矩形上边缘标题

返回值

	返回值
HWND	返回对象的句柄

实例

下列来自“Demo.wnf”文件的引用说明该函数的使用。 带有标题“框架”的框架将显示在“演示向导”的“设置选项”对话框中。




```
void OnOption2(void)
{
//□□
CreateFrame(0,150,"Frame");
}
...
...
```

CreateRadioButton

简介

在“设置选项”对话框中，单选钮将显示在 x、y 坐标轴上。 单选钮可以用来启用选项。

在单个对话框中存在多个单选钮时，使用单选钮才有用。 一次只有一个单选钮可激活。

语法

HWND CreateRadioButton (int x, int y, char* Text, BOOL* pSelect)

参数

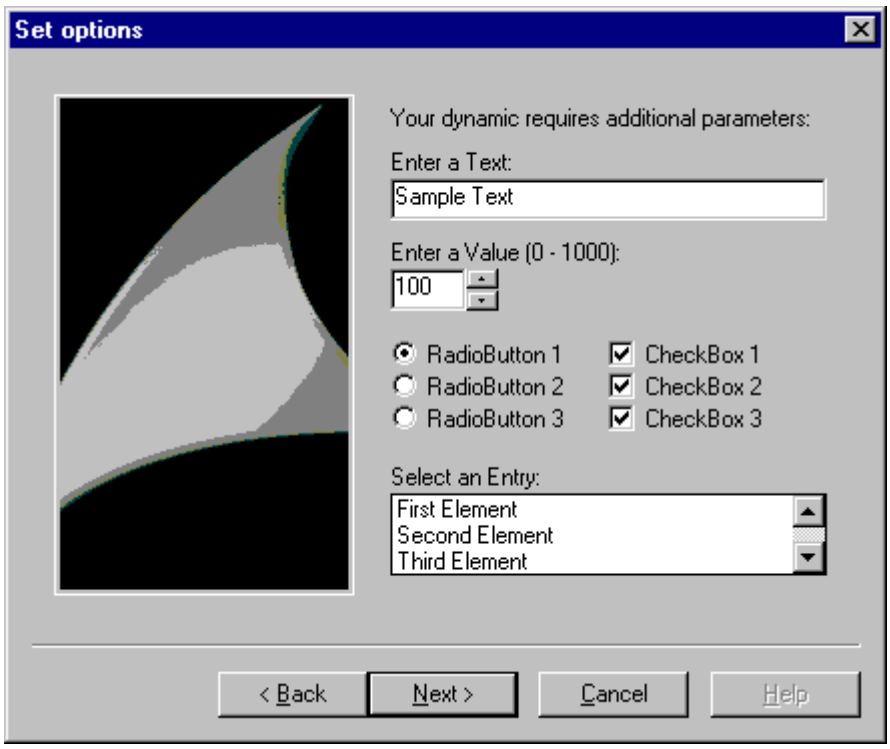
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 Y 坐标轴的数值。
char* Text	通过单选钮激活的选项的名称。 文本将显示在单选钮的右边。
BOOL* pSelect	指向结果变量的指针。应为结果变量预先指定一个默认值 (True/False)。

返回值

	返回值
HWND	返回对象的句柄
pSelect	激活状态： FALSE = 没有激活 TRUE = 激活

实例

下列来自“Demo.wnf”文件的引用说明了该函数的使用。 在演示向导的“设置选项”对话框中，将显示三个单选钮，它们各自代表一个选项。 只有一个选项可以激活。




```
BOOL g_bOption1 = TRUE;

BOOL g_bOption2 = FALSE;

BOOL g_bOption3 = FALSE;


void OnOption1(void)

{

static BOOL bFirst = TRUE;

HWND hWnd = NULL;

.....

if (bFirst == TRUE)

{

...
```

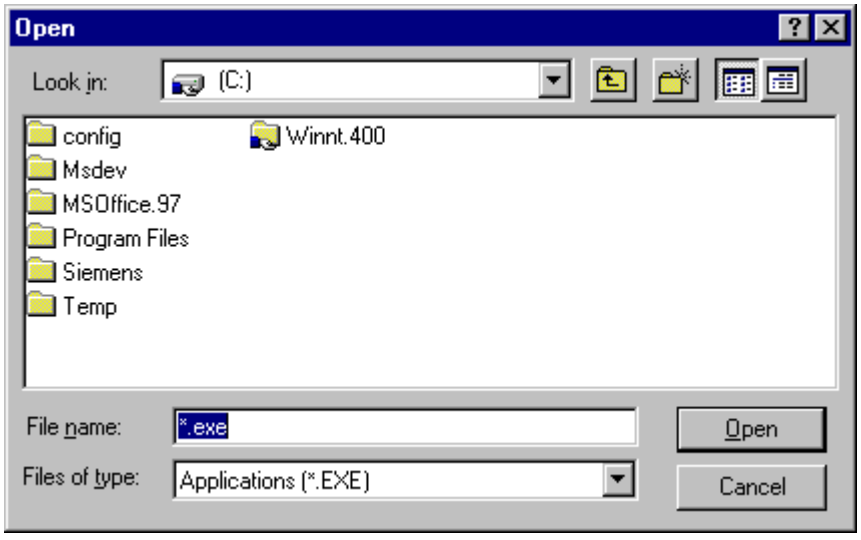
```
}
...
...
//单选钮
CreateRadioButton(0,100,"RadioButton 1",&g_bOption1);
CreateRadioButton(0,116,"RadioButton 2",&g_bOption2);
CreateRadioButton(0,132,"RadioButton 3",&g_bOption3);
}
```

CreateFileBrowser

简介

在“设置选项”对话框中，一个浏览按钮将显示在 x、y 坐标轴上。 在输入域中可以输入文件名。

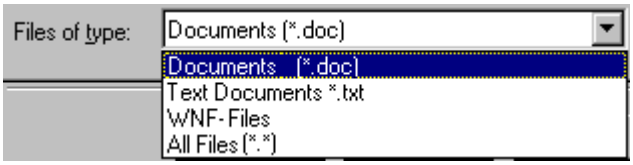
单击“浏览”按钮打开文件选择对话框。



语法

HWND CreateFileBrowser (int x, int y, DWORD Flags, char* Filter, char* Dateiname)

参数

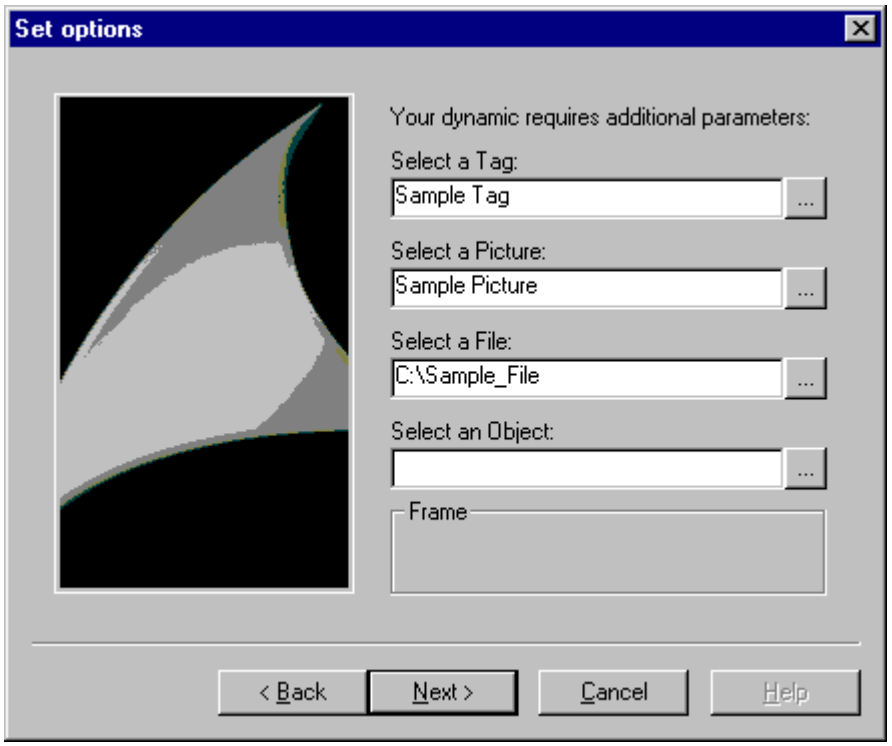
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
DWORD 标记	选择窗口的控制标记： FB_WITHPATH = 带路径的文件名 FB_SAVE_AS = 显示“另存为”对话框代替“打开”对话框。
char* Filter	文件选择对话框的列表框中显示的数据类型的过滤器。 通过指定扩展名，可以指定将显示在列表框中的数据类型。 过滤器由一对字符串组成。 第一个字符串是过滤器的名称。 第二个字符串是具有 *.typ 文件形式的过滤器函数，“typ”是文件的扩展名。 只有这些具有该扩展名的文件才显示在选择域中。 1. 第一个和第二个字符串之间用一个 | 隔开。 多个过滤器可以链接起来，用 | 隔开。 最后一个过滤器将用 || 限制。 实例： char* Filter1 = "Graphic pictures (*.PDL) | *.PDL||";  char* Filter2 = "Documents (*.doc) | *.doc|" "Text files *.txt | *.txt|" "WNF files | *.wnf|" "All files (*.*) | *.*||";  在过滤器函数的最后不允许有空格。
char* File name	文件名的输入缓冲区。 路径名称可以预置为默认值。 该标准值具有下列作用： 在输入域中路径名称将显示为默认。 如果单击“浏览”按钮，将在文件选择对话框中设置路径。 如果文件名有扩展名“*.typ”，则所有该类型的文件将显示在选择对话框的选择域中。

返回值

	返回值
HWND	返回对象的句柄
文件名	输入缓冲区包含文件名。

实例

下列来自“Demo.wnf”文件的引用说明了该函数的使用。 在演示向导的“设置选项”对话框中，将显示带有浏览按钮的输入域。 单击“浏览”按钮打开文件选择对话框。




```
char* DynWizFileBrowserStatic = "Select a file:";

char* DynWizFileBrowser = "C:\\Sample file";

char* DynWizFilter = "Text files (*.txt) | *.txt|"
  "All files (*.*) | *.*||";

...

char g_szFileBrowser[256];

...
```

```
void OnOption2(void)
{
static BOOL bFirst = TRUE;
HWND hWnd = NULL;
RECT rect;
...
if (bFirst == TRUE)
{
...
strcpy(g_szFileBrowser,DynWizFileBrowser);
First = FALSE;
}
...
...
//带浏览按钮的输入域的静态文本
CreateStatic(0,95,DynWizFileBrowserStatic);
//文件选择对话框
hWnd =
CreateFileBrowser(0,110,FB_WITHPATH,DynWizFilter,g_szFileBrowser)
;
MoveWindow(hWnd,0,110,(rect.right-rect.left),21,TRUE);
}
```

CreateVarBrowser / CreateVarBrowserEx

简介

在“设置选项”对话框中，将针对 x、y 坐标显示一个带有浏览按钮的输入域。 在输入域中可以输入变量名。 单击浏览按钮，将打开 WinCC 的变量选择对话框。 用“CreateVarBrowserEx”函数，可以将变量过滤器进一步参数化。 该过滤器可限制显示在变量选择对话框中的变量。 可以过滤数据类型、变量组、变量名和连接。

语法

HWND CreateVarBrowser (int x, int y, char* VarName)

HWND CreateVarBrowserEx (int x, int y, LPDM_VARFILTER VarFilter, char* VarName)

参数

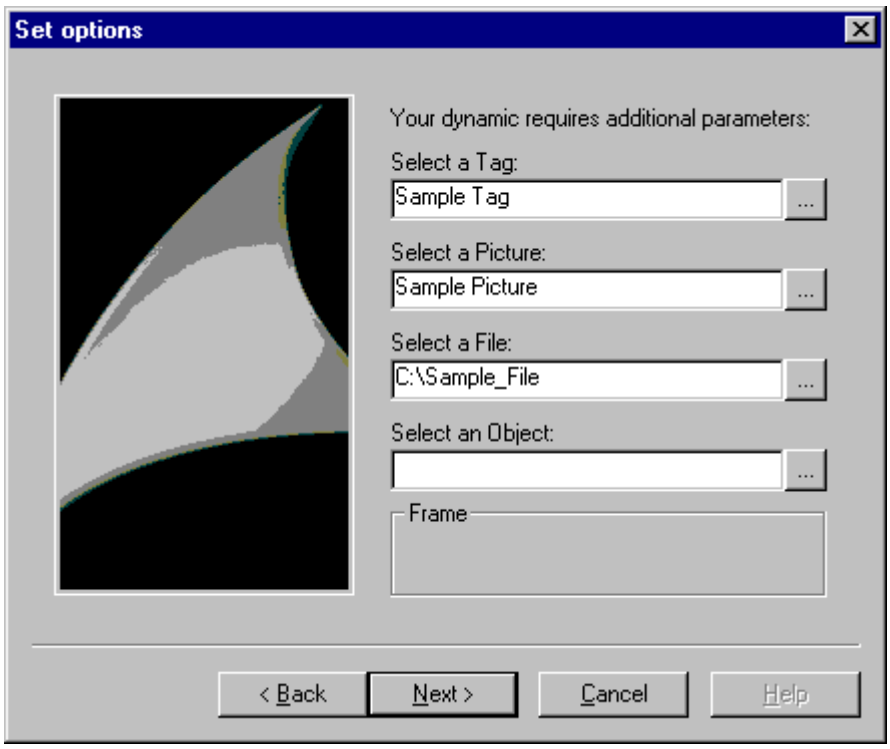
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
LPDM_VARFILTER VarFilter	变量过滤器的指针可选条目。 如果指定 NULL 指针，则不激活过滤器。 变量过滤器必须用 DM_VARFILTER 结构定义。 更多信息，请参见 WinCC ODK 文档。
char* VarName	包含变量的名称。 变量名可以具有预置的默认值。 总是显示该条目。

返回值

	返回值
HWND	返回对象的句柄
VarName	输入缓冲区包含变量名

实例

下列来自“Demo.wnf”文件的引用说明了该函数的使用。在演示向导的“设置选项”对话框中，将显示带有浏览按钮的输入域。单击浏览按钮，将打开 WinCC 的变量选择对话框。




```
char* DynWizVarBrowser = "Sample tag";
char* DynWizPicBrowserStatic = "Select a picture:";
...
char g_szVarBrowser[256];
...
void OnOption2(void)
{
static BOOL bFirst = TRUE;
HWND hWnd = NULL;
RECT rect;
...
if (bFirst == TRUE)
```

```
{
...
strcpy(g_szVarBrowser,DynWizVarBrowser);
First = FALSE;
}
...
...
//带浏览按钮的输入域的静态文本
CreateStatic(0,95,DynWizFileBrowserStatic);
//变量选择对话框
hWnd =
CreateFileBrowser(0,110,FB_WITHPATH,DynWizFilter,g_szFileBrowser)
;
GetWindowRect(GetParent(hWnd), &rect);
MoveWindow(hWnd,0,110,(rect.right-rect.left),21,TRUE);
}
```

CreatePackageBrowser/CreatePackageBrowserEx

简介

在“设置选项”对话框中，将针对 x、y 坐标显示一个带有浏览按钮的输入域。 在输入域中可以输入一个名称。 数据包浏览器由输入域右边缘的浏览按钮启用。 从数据包显示的数据类型，由标记或 ProgID 指定。

对于“CreatePackageBrowserEx”函数，可以通过 ProgID 代替标记。

语法

HWND CreatePackageBrowser (int x, int y, DWORD flags, char* Name)

HWND CreatePackageBrowserEx (int x, int y, char* ProgID, char* Name)

参数

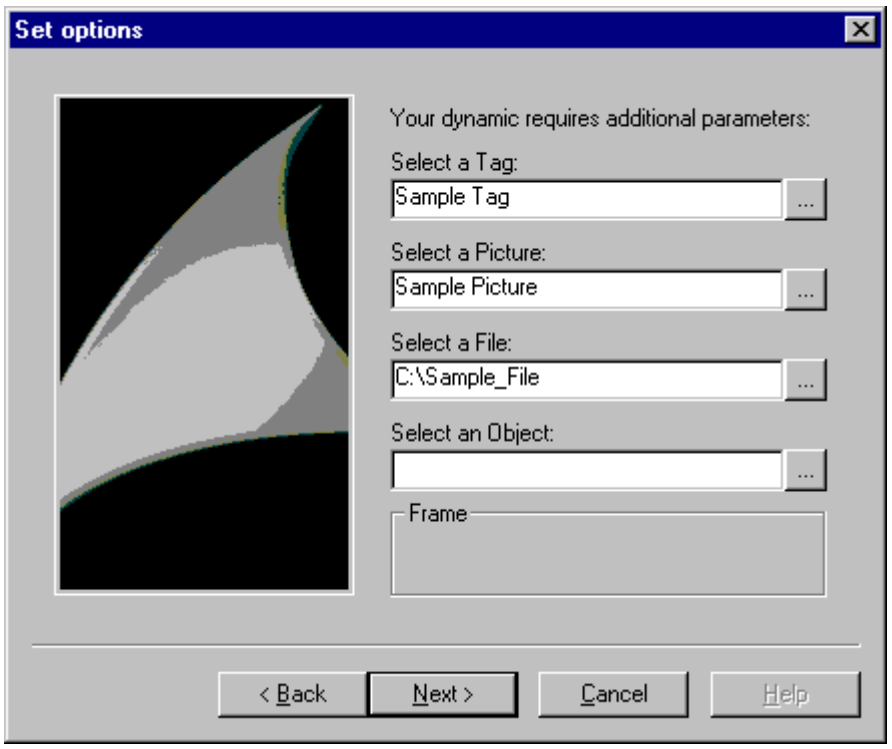
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
DWORD 标记	当前只能使用 PB_PICTURE。 它激活画面选择。
char* ProgID	创建选择的组件的程序 ID。 通过传送 “WinCC.CCFileASOStub.1”，将选中画面选择。
char* Name	包含名称。 名称可以具有预置的默认值。总是显示该条目。

返回值

	返回值
HWND	返回对象的句柄
名称	输入缓冲区包含名称

实例

下列来自“Demo.wnf”文件的引用说明了该函数的使用。在演示向导的“设置选项”对话框中，将显示带有浏览按钮的输入域。 单击“浏览”按钮打开画面选择对话框。




```
char* DynWizPicBrowserStatic = "Select a picture:";

char* DynWizPicBrowser = "Sample picture";


...

char g_szPicBrowser[256];

...

void OnOption2(void)

{

static BOOL bFirst = TRUE;

HWND hWnd = NULL;

RECT rect;

...
```

```
if (bFirst == TRUE)
{
...
&#9;strcpy(g_szPicBrowser,DynWizPicBrowser);

First = FALSE;
}
...
...
```

//带浏览按钮的输入域的静态文本

```
CreateStatic(0,50,DynWizPicBrowserStatic);
```

//画面选择对话框

```
hWnd = CreatePackageBrowser(0,65,PB_PICTURE,g_szPicBrowser);
MoveWindow(hWnd,0,65,(rect.right-rect.left),21,TRUE);
}
```

CreateObjectBrowser

简介

在“设置选项”对话框中，将针对 x、y 坐标显示一个带有浏览按钮的输入域。 对象或属性名称可以在输入域中输入。 单击浏览按钮打开选择对话框。 在选择对话框中，可以搜索并且选择对象或属性名称。

语法

HWND CreateObjectBrowser (int x, int y, char* Title, DWORD flags, char* ObjectName)

参数

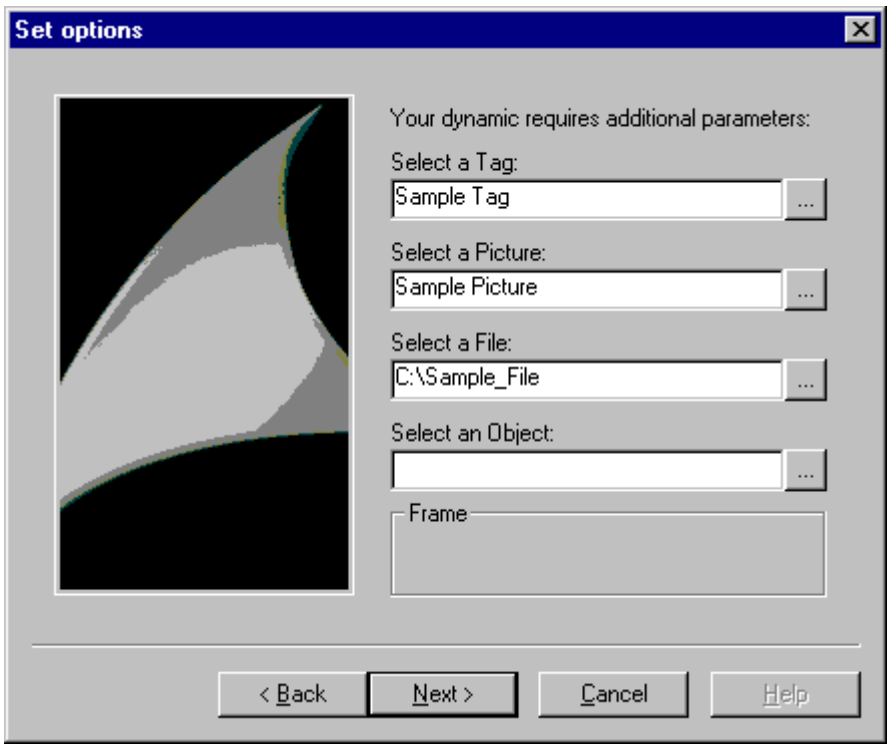
参数	描述
int x	显示 x 坐标轴的数值。
int y	显示 y 坐标轴的数值。
char* title	选择对话框的标题。
DWORD 标记	可以传递两种不同的标记： OB_OBJECTS 显示所有对象 OB_PROPERTIES 也提供属性选择。
char* ObjectName	对象或属性名称的输入缓冲区。 输入缓冲区可以设置为默认值。

返回值

	返回值
HWND	返回对象的句柄
ObjectName	输入缓冲区包含对象或属性名称。

实例

下列来自“Demo.wnf”文件的引用说明了该函数的使用。在演示向导的“设置选项”对话框中，将显示带有浏览按钮的输入域。 单击浏览按钮，将打开窗口对象选择对话框。




```
char* DynWizObjectBrowserStatic = "Select an object:";

char* DynWizObjectBrowser = "Object";

char* DynWizObject = "Window object selection";

;

...

char g_szObjectBrowser[256];

...

void OnOption2(void)

{

static BOOL bFirst = TRUE;

HWND hWnd = NULL;

RECT rect;
```

```
...
if (bFirst == TRUE)
{
...
strcpy(g_szObjectBrowser,DynWizObjectBrowser);
First = FALSE;
}
...
...
//带浏览按钮的输入域的静态文本
CreateStatic(0,50,&#9;CreateStatic(0,140,DynWizObjectBrowserStatic););
//窗口选择对话框
hWnd =
CreateObjectBrowser(0,155,DynWizObject,OB_OBJECTS,g_szObjectBrowser);
MoveWindow(hWnd,0,155,(rect.right-rect.left),21,TRUE);
}
```

1.4.5.13 生成动态的向导函数

GenerateBLOB

简介

GenerateBLOB (BLOB = Binary Large OBject) 函数用于创建一个动作，可将其附加到图形对象属性。动作由 3 部分组成。

开端：这是 C 函数的头部。

实例：
#include "apdefap.h"
void OnLButtonDown(char* lpszPictureName,char* lpszObjectName,char* lpszPropertyName, UINT nFlags, int x, int y)

{
开端依赖于用于启动动作的触发器（在上例中为单击鼠标左键）。

结尾：这是 C 函数的结尾，由符号“}”组成。

核心：这部分包含 C 函数的实际功能。 实例：ProgramExecute("notepad.exe");

此函数用于创建及编译动作的 C 代码。编译结果是生成 P 代码。该代码将由 WinCC 运行系统进行解释和处理。如果 C 代码不正确，将不会生成 P 代码。

该函数会创建一个 BLOB，在其中存储动作的各个部分（C 代码、P 代码、触发器...）。在向导函数结束前，必须再次删除 BLOB。有关删除 BLOB 函数的详细信息，请参见“DeleteBLOB”。

语法

AP_BLOB GenerateBLOB (char* Prolog, char* Epilog, char* Format, ...)

参数

参数	描述
char* Prologue	动作的开端，ASCII 字符串形式。
char* Epilogue	动作的结尾，ASCII 字符串形式。
char* Format	动作的核心，ASCII 字符串或格式字符串形式，符合标准函数“printf”。

说明

C 代码由 C 函数 sprintf 生成。参数以格式字符串的形式处理，即评估格式控制字符（例如，\ % " ）。如果这些字符即将传递到 C 代码中（例如，作为某个动作中 printf 调用的格式字符串），那么必须为它们提供一个 \ 符号。
实例：
\ → \ \
% → \%
" → \"

返回值

对于以下结构组件，该函数将返回 AP_BLOB 类型的结构变量：

结构组件	返回值
DWORD dwPCodeSize	生成的 P 代码的长度（字节）
LPVOID lpPCode	指向所生成的 P 代码的指针
int nErrors	编译器的错误数
int nWarnings	编译器的警告数

实例

下列来自“Execute Programm.wnf”文件的引用说明了该函数的使用。向导函数将生成用于启动另一应用程序的 C 脚本（在本例中为 notepad.exe）。

```
...
...
void OnGenerate(void)
{
PCMN_ERROR pError;
AP_BLOB *blob;
char code[500];
char sError[500];
..
Slash2DblSlash(g_Picture, strlen(g_Picture));
..
sprintf(code,"%sProgramExecute(\"%s\");",ifcode,g_Picture);
..
//开端
blob = GenerateBLOB("#include \"apdefap.h\"\r\n"
"void OnClick(char* lpszPictureName," "char*lpszObjectName,char*
lpszPropertyName,"
"UINT nFlags,int x, int y) {",
```

//结尾

```
"}",
```

//核心

```
code);

BEGIN_JCR_BLOBERRORS

SetAction(NULL, blob, g_Trigger);

END_JCR_BLOBERRORS

DeleteBLOB(blob);
}
```

生成的 C 脚本

```
#include "apdefap.h"
void OnLButtonDown(char* lpszPictureName,
char* lpszObjectName,
char* lpszPropertyName,
UINT nFlags, int x, int y)
{
ProgramExecute("notepad.exe");
}
```

DeleteBLOB

简介

GenerateBLOB 函数生成一个 BLOB。 在向导函数的末尾，必须删除 BLOB。 DeleteBLOB 函数用于删除 BLOB。

语法

void DeleteBLOB (AP_BLOB* blob)

参数

参数	描述
AP_BLOB* blob	指针指向“GenerateBLOB”函数的结果变量。

实例

DeleteBLOB(blob);

SetAction

简介

动作被附加至指定触发器处的选定图形对象。

如果触发器是一个事件，它将在调用参数时直接指定。

如果触发器是属性的动态化，则必须使用函数 AddVarTrigger 或 AddTimeTrigger 将其预先输入到 BLOB 中。

说明

如果动作未被附加至所选择的对象，而是另一个对象，则必须使用 API 函数 PDLCSSetAction。有关 PDLCSSetAction 函数的详细信息，请参见 WinCC ODK 手册。

语法

BOOL SetAction (char* Property, AP_BLOB* Blob, DWORD Trigger)

参数

参数	描述
char* Property	属性名称。总是使用属性的英文名称。 在事件触发的情况下，必须传递 NULL 指针。
AP_BLOB* Blob	指向“GenerateBLOB”函数结果变量的指针。
DWORD TriggerID	触发器的 ID： NOTDEFINED = 触发器已输入到 BLOB 中 MOUSECLICK = 鼠标单击 MOUSELBUTTONDOWN = 按下鼠标左键 MOUSELBUTTONUP = 释放鼠标左键 MOUSERBUTTONDOWN = 按下鼠标右键 MOUSERBUTTONUP = 释放鼠标右键 KEYBOARDDOWN = 按下键盘键 KEYBOARDUP = 释放键盘键 OBJECTCHANGE = 对象切换 PROPERTYCHANGE = 属性切换 PICTUREOPEN = 画面选择 PICTURECLOSE = 关闭画面

返回值

	返回值
BOOL 返回值	TRUE = 函数成功执行。 FALSE = 函数执行未成功。

实例

请参见 GenerateBLOB 函数中的实例。

参见

GenerateBLOB (页 75)

AddTimeTrigger

简介

该函数用“周期性触发器”类型的触发器补充动作。

语法

BOOL AddTimeTrigger (AP_BLOB* Blob, char* Name, DWORD TriggerType, DWORD GraphCycleType, DWORD CycleID)

参数

参数	描述
AP_BLOB* Blob	指针指向“GenerateBLOB”函数的结果变量。
char* Name	事件的名称。 它可以是任何 ASCII 字符串。 该名称将作为事件名称显示在动作窗口中。
DWORD TriggerType	循环触发的类型： 2 = 时间周期（标准周期） 4 = 图形对象周期

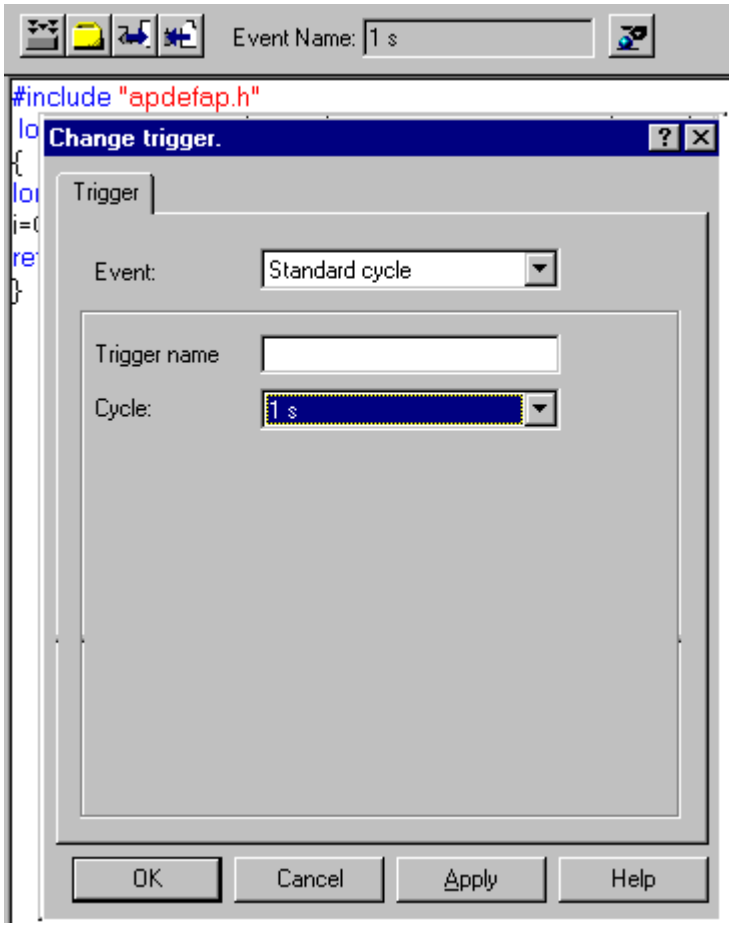
参数	描述
DWORD GraphCycleType	图形对象循环的类型： 2 = 窗口周期 1 = 画面周期
DWORD CycleID	触发周期： 0 = 根据变化 1 = 250 毫秒 2 = 500 毫秒 3 = 1 秒 4 = 2 秒 5 = 5 秒 6 = 10 秒 7 = 1 分钟 8 = 5 分钟 9 = 10 分钟 10 = 1 小时 11 = 用户周期 1 12 = 用户周期 2 13 = 用户周期 3 14 = 用户周期 4 15 = 用户周期 5

返回值

	返回值
BOOL	TRUE = 函数成功执行。 FALSE = 函数执行未成功。

实例

两个动作触发的时间间隔为 1 秒。




```
BOOL FctRet;

..

FctRet = AddTimeTrigger(blob, "1 sec", 2, 0, 3);
```

AddVarTrigger /AddVarTriggerEx

简介

该函数用“变量触发器”类型的触发器补充动作。

语法

BOOL AddVarTrigger (AP_BLOB* Blob, char* EventName, char* VarName)

BOOL AddVarTriggerEx (AP_BLOB* Blob, char* EventName, char* VarName, DWORD CycleID)

参数

参数	描述
AP_BLOB* Blob	指针指向“GenerateBLOB”函数的结果变量。
char* EventName	事件的名称。 它可以是任何 ASCII 字符串。 该名称将作为事件名称显示在动作窗口中。
char* VarName	WinCC 变量的名称，它触发事件或包含在触发中。
DWORD CycleID	触发周期： 0 = 根据变化 1 = 250 毫秒 2 = 500 毫秒 3 = 1 秒 4 = 2 秒 5 = 5 秒 6 = 10 秒 7 = 1 分钟 8 = 5 分钟 9 = 10 分钟 10 = 1 小时 11 = 用户周期 1 12 = 用户周期 2 13 = 用户周期 3 14 = 用户周期 4 15 = 用户周期 5 对于 AddVarTrigger 函数，数值 CycleID = 4 是预定义的。

返回值

	返回值
BOOL	TRUE = 函数成功执行。 FALSE = 函数执行未成功。

实例

在触发器类型“变量”中，已输入变量“StartTurbine1”作为触发器。 这些变量的值一发生变化，动作就启动。


```
BOOL FctRet


FctRet = AddVarTriggerEx(blob,"Turbine Start","StartTurbine1",0);
```

SetValidateFct

简介

验证函数的名称将被传递到动态向导中。利用验证函数，可以检查选项和触发器参数。如果测试结果为负值，可重新输入。

当在“选择选项”对话框或“设置触发”对话框中单击“继续”按钮时，将调用验证函数。如果测试结果为正值，将关闭对话框并显示下一页面。如果测试结果为负值，对话框将保持打开状态。只有在输入正确的参数后方可继续。

只要在动态向导中设置验证函数，便会立即生效。对后续选项页面仍然生效。如果没有任何验证函数生效或是其它验证函数生效，则必须设置哑元函数（包含正测试结果）或其它验证函数。

语法

BOOL SetValidateFct (LPCSTR FctName)

参数

参数	描述
LPCSTR FctName	验证函数的名称，ASCII 字符串形式。

返回值

	返回值
BOOL	测试的结果 TRUE = 正测试结果。 FALSE = 负测试结果。

实例

下列来自“Instanzobjekt.wnf”文件的引用说明了该函数的使用。

向导函数已通过验证函数进行了扩展。

```
...
```

```
...
//验证选项 1
BOOL ValidateOpt1(void)
{
//所选的属性
return (strcmp(g_NewInst, ""));
}

void OnOption1(void)
{
HWND hWnd;
RECT rect;
DM_VARFILTERdmFilter = {DM_VARFILTER_TYPE, 1, NULL, NULL, NULL,
NULL };

SetValidateFct("ValidateOpt1");
sprintf(g_NewInst, "");
..
}
```

EnumProperty/EnumPropertyEx

简介

EnumProperty 函数列出对象的属性。 用 EnumPorpertyEx 函数，可以为对象指定将要列出的属性。

语法

BOOL EnumProperty (char* FName, LPVOID pItem, DWORD dwFlags);

BOOL EnumPropertyEx (LPCTSTR Projectname, LPCTSTR Picturename, LPCTSTR Objectname, char* FName, LPVOID pItem, DWORd dwFlags);

参数

参数	描述
LPCTSTR Projectname	指针指向项目的名称，包含目录和文件扩展名。
LPCTSTR Picturename	指针指向应列出其对象的画面的名称。 区分大小写。
LPCTSTR 对象名称	指针指向对象的名称
char* FName	回叫函数的名称，为每个对象属性调用一次。
LPVOID pItem	指针指向应用程序指定的要传递给回叫函数的数据。 该指针不会经函数计算，且不能被回叫函数使用。
DWORD dwFlags	dwFlags 指定将要列出的属性类型。 当前，下列规范是可能的： **PropertyHasDynamic**（数值： 0x0001） 仅列举具有动态的对象属性。 **PropertyHasEvents**（数值： 0x0001） 仅列举具有事件的对象属性。 **PropertyIsDynamicable**（数值： 0x0001） 仅列举可成为动态的对象属性。

返回值

	返回值
BOOL	TRUE = 列出对象类型的对象属性 FALSE = 出错

实例

下列来自“Dynamic Property.wnf”文件的引用说明该函数的使用。

```
...
...
//□□□□
BOOL EnumFct(char *property, VARTYPE vt, LPVOID pItem)
{
```

```
sprintf(g_prop[SendMessage((HWND)pItem,LB_INSERTSTRING,
(WPARAM)-1,(LPARAM)property)],property);
return TRUE;
}

void OnOption1(void)
{
HWND hWnd,LBHwnd;
RECT rect;
static BOOL bFirst = TRUE;

if(bFirst)
{
...
}
...

CreateStatic(0, 10,"Properties of the current object :");
LBHwnd=CreateListbox(0, 30, g_Headline, 8, &g_indexProperty);
EnumProperty("EnumFct", LBHwnd, 3);
GetWindowRect(GetParent(LBHwnd), &rect);
...
```

1.4.5.14 向导 WinCC 函数

GetProjectName

简介

确定当前 WinCC 项目的路径。

语法

LPCSTR GetProjectName (void)

返回值

	返回值
LPCSTR	指针指向 MCP 文件的 ASCII 字符串

实例

LPCSTR Name;

Name = GetProjectName();

例如，函数提供下列结果： C:\Siemens\WinCC\WinCCProjects\Example.mcp

GetPictureName

描述

将确定当前画面（*.pdl）的名称。

语法

LPCSTR GetPictureName (void)

返回值

	返回值
LPCSTR	指针指向 PDL 文件的 ASCII 字符串

实例

LPCSTR Name;

Name = GetPictureName();

例如，函数提供下列结果： TurbineControl.PDL

GetDefaultWNFPath

描述

确定当前 WNF 目录的路径。

语法

LPCSTR GetDefaultWNFPath (void)

返回值

	返回值
LPCSTR	指针指向路径名称的 ASCII 字符串

实例

LPCSTR Name;

Name = GetDefaultWNFPath();

例如，函数提供下列结果： C:\Siemens\WinCC\wscripts\wscripts.deu\

GetObjectName

简介

确定在当前画面中所选图形对象的名称。

语法

LPCSTR GetObjectName (void)

返回值

	返回值
LPCSTR	指针指向对象名的 ASCII 字符串

实例

LPCSTR Name;

Name = GetObjectName();

例如，函数提供下列结果： Button1

InsertXRefSection

描述

该函数用于按照 Xref 表示法将某个区段插入到传递的源代码中，从而按照定义的方式来输入传递的变量和画面名称。

语法

BOOL InsertXRefSection (char * SourceCode, char* TagName[], int TagCount, char* PictName[], int PictCount)

参数

参数	描述
char *SourceCode	代码缓冲区，在其中插入 Xref 区段
char *TagName[]	NULL 或插入到 Xref 区段中的变量名称域。
int TagCount	域 DayName[] 中的变量名的数量
char *PictName[]	NULL 或插入到 Xref 区段中的画面名称域
int PictCount	域 PictName[] 中画面名称的数量

返回值

	返回值
BOOL	说明函数是否已成功执行的结果值。
char *TagName[]	用于定义在同一位置传递的变量的域
char *PictName[]	用于定义在同一位置传递的画面的域

实例

```
char* szPictureArray[1];
char szPictName[255];
char szSourceCode[1100];

strcpy(szPictName, "Newpdl.pdl");
szPictureArray[0] = szPictName;
strcpy(szSourceCode, "");
InsertXrefSection(szSourceCode,NULL,0,szPictureArray,1);
```

该函数返回下列结果：

```
szSourceCode:
// WINCC:TAGNAME_SECTION_START
//语法：#define TagNameInAction "DMTagName"
// next TagID :1
// WINCC:TAGNAME_SECTION_END
// WINCC:PICNAME_SECTION_START

//语法：#define PicNameInAction "PictureName"
// next PicID :1
#define PIC_0 " Newpdl.Pdl"
```

// WINCC:PICNAME_SECTION_END

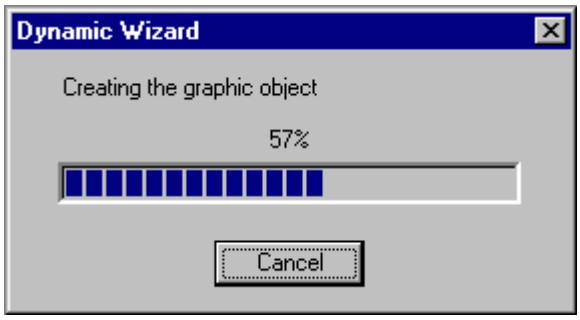
szPictureArray[0]:"PIC_0"

1.4.5.15 向导进程函数

向导进程函数

简介

进程函数用于显示在“进行框”中处理程序的进程（用 % 表示）。



当创建进程条（CreateProgressDlg）时，必须指定开始和结束值以及增量值。 开始值相应于进程 0%，结束值相应于进程 100%。 增量指定进程改变的步长。

通常开始值 = 0，增量 = 1。结束值是操作步骤数。

对于每个处理步骤，进程将增加一个增量（Progress_StepIt）或设置为一个定义的值（Progress_SetPos）。

在处理程序的结尾，必须清除进程显示（DestroyProgressDlg）。

在进程条（Progress_SetStatus）中可以显示一个文本，例如文本“创建图形对象”。 它也可以在处理过程中进行改变，以区分过程的不同部分。

在大多数情况下，以允许时间顺序线性进程显示的方式来分开此过程是不可能的。 然而，这实际上没有必要。 这样显示进程已足够。

参见

DestroyProgressDlg (页 97)

Progress_SetPos (页 97)

Progress_StepIt (页 96)

Progress_SetStatus (页 96)

CreateProgressDlg (页 95)

CreateProgressDlg

简介

在进程条中，处理程序的进程将在 0 到 100% 的范围内显示。

语法

PROGRESS_DLG CreateProgessDlg (int nLower, int nUpper, int nStepInc)

参数

参数	描述
int nLower	进程的开始值（对应为 0 %）
int nUpper	进程的结束值（对应为 100 %）
int nStepInc	进程的增量

返回值

	返回值
PROGRESS_DLG	对象的句柄

Progress_SetStatus

描述

在进程条中将输入作为标题的文本。

语法

void Progress_SetStatus (PROGRESS_DLG hDlg, char* ActionName)

参数

参数	描述
PROGRESS_DLG hDlg	对象的句柄
char* ActionName	标题的文本

Progress_StepIt

描述

处理程序的进程将通过一个步长增加。

语法

void Progress_StepIt (PROGRESS_DLG hDlg)

参数

参数	描述
PROGRESS_DLG hDlg	对象的句柄

Progress_SetPos

描述

在进程条中，进程将被设置为所定义的数值。 该值必须位于开始值和结束值之间。

语法

void Progress_SetPos (PROGRESS_DLG hDlg, int nPos)

参数

参数	描述
PROGRESS_DLG hDlg	对象的句柄
int nPos	进程值

DestroyProgressDlg

简介

将关闭进程条。

语法

void DestroyProgressDlg (PROGRESS_DLG hDlg)

参数

参数	描述
PROGRESS_DLG hDlg	对象的句柄

1.4.5.16 向导 Windows 函数

向导 Windows 函数

简介

下面显示 Windows 函数的简要描述，它可以或必须与向导系统函数（特别是 windows 函数用于输入参数时）偕同使用。

详细信息请参见 Microsoft Developers Studio /Win32 SDK 编程员的参考。

参见

MessageBox (页 103)

ShowWindow (页 102)

GetWindow (页 101)

SendMessage (页 101)

MoveWindow (页 100)

GetWindowRect (页 99)

GetParent (页 98)

GetParent

简介

给定一个窗口，可以确定其父窗口的句柄，例如，选项窗口的句柄。

语法

HWND GetParent (HWND hWnd)

参数

参数	描述
HWND hWnd	要为其确定父窗口的窗口句柄。

返回值

	返回值
HWND	父窗口的句柄 NULL = 不存在父窗口。

GetWindowRect

简介

确定窗口的大小和坐标，例如选项窗口的大小。

语法

BOOL GetWindowRect (HWND hWnd, LPRECT lpRect)

参数

参数	描述
HWND hWnd	窗口句柄
LPRECT lpRect	指针指向结构化的结果变量

返回值

	返回值
BOOL	TRUE = 函数成功执行。 FALSE = 函数执行未成功。
LPRECT lpRect	具有结构组件的 LPRECT 结构的结构化结果变量： LONG left： 左上角的 X 坐标 LONG top： 左上角的 Y 坐标 LONG right： 右下角的 X 坐标 LONG bottom： 右下角的 Y 坐标：

参见

将“Motor.wnf”脚本添加到数据库中 (页 108)

CreateEdit (页 48)

MoveWindow

简介

将改变窗口的位置和尺寸，例如，选项窗口中输入域的位置和大小。

语法

BOOL MoveWindow (HWND hWnd, int x, int y, int nWidth, int nHeight, BOOL bRepaint)

参数

参数	描述
HWND hWnd	窗口句柄
int x	左上角的 X 坐标
int y,	左上角的 Y 坐标
int nWidth	宽度
int nHeight	高度
BOOL bRepaint	TRUE = 窗口重新绘制。

返回值

	返回值
BOOL	TRUE = 函数成功执行。 FALSE = 函数执行未成功。

参见

为电机创建“动态化向导功能” (页 107)

SendMessage

简介

消息被发送到窗口。 函数用于填充选择域，例如。

语法

LRESULT SendMessage (HWND hWnd, UINT Msg, WPARAM wParam, LPARAM lParam)

参数

参数	描述
HWND hWnd	窗口句柄
UINT Msg,	消息类型： LB_INSERTSTRING = 在 ListBox 中插入文本
WPARAM wParam	1. 消息参数： -1 = 文本将附在最后。
LPARAM lParam	2. 消息参数： 指向文本的指针

返回值

	返回值
LRESULT	对象的句柄

GetWindow

简介

确定在与其它窗口（原先的窗口）有特殊关系的窗口的句柄。

语法

GetWindow (HWND hWnd, UINT uCmd)

参数

参数	描述
HWND hWnd	原始窗口的句柄
UINT uCmd	关系 GW_HWNDFIRST = 上窗口

返回值

	返回值
HWND	已找到窗口的句柄或 NULL

ShowWindow

简介

设置窗口的显示类型。

语法

ShowWindow (HWND hWnd, int nCmdShow)

参数

参数	描述
HWND hWnd	窗口句柄
int nCmdShow	窗口的显示状态 SW_HIDE = 不可见

返回值

	返回值
BOOL	TRUE = 窗口可见 FALSE = 窗口不可见

MessageBox

简介

当错误发生或有必要进行用户操作时，函数将向用户显示消息。

消息通过用户指定的文本、标题和按钮显示。

语法

int MessageBox (HWND hWnd, LPCTSTR lpText, LPCTSTR lpCaption, UINT uType)

参数

参数	描述
HWND hWnd	父窗口的句柄 NULL = 消息没有父窗口。
LPCTSTR lpText	消息文本
LPCTSTR lpCaption	标题文本
UINT uType	框类型 MB_OK = 带“确定”按钮的消息 MB_OKCANCEL = 带“确定”和“取消”按钮的消息

返回值

	返回值
int	被单击的按钮的标识符： IDOK = 单击“确定”按钮 IDCANCEL = 单击“取消”按钮

实例


```
int RetMsg;
RetMsg = MessageBox (NULL, "Error calling the API functions", "System error", MB_OK);
```

1.4.6 实例

1.4.6.1 实例

简介

在该描述的上下文中，将提供两个动态向导函数的实例：

- 演示向导
- 动态电机

参见

动态电机 (页 107)
演示向导 (页 104)

1.4.6.2 演示向导

演示向导

简介

在“Demo.wnf”文件中，创建名为“演示向导”的动态向导。 该动态向导显示用户可用于方便地输入数据的基本功能。 然而，演示向导不执行动作。

参见

将“Demo.wnf”脚本添加到数据库中 (页 106)

创建帮助文本 (页 105)

为演示向导创建动态向导功能 (页 105)

为演示向导创建动态向导功能

要求

必须打开一个 WinCC 项目。

步骤

1. 在 Windows 资源管理器中，将“Demo.wnf”文件从目录“Siemens\ WinCC\ documents\ german”复制到目录“Siemens\ WinCC\ wscript\ wscript.deu”。
2. 启动动态向导编辑器。
3. 在动态向导编辑器的“文件”菜单中，选择“打开”。 文件选择对话框打开。
4. 标记“Demo.wnf”文件。 单击“打开”。 “Demo.wnf”文件显示在编辑器窗口中。
5. 单击工具栏中的  图标，编译脚本。 结果显示在输出窗口中。

参见

将“Demo.wnf”脚本添加到数据库中 (页 106)

创建帮助文本

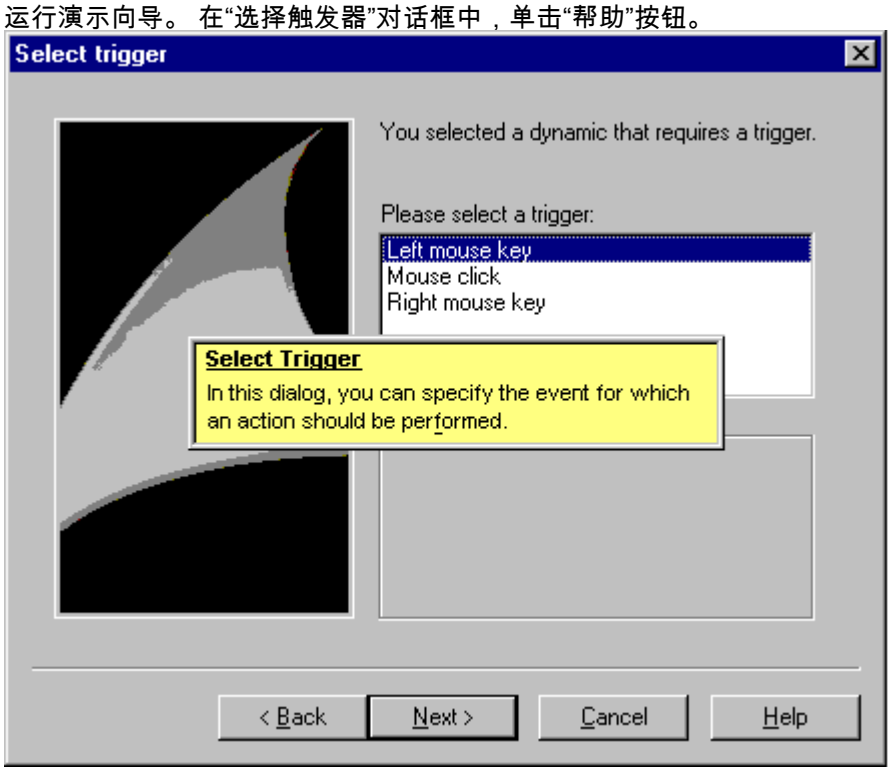
简介

在该部分，将创建“选择触发器”对话框的帮助。

步骤

1. 单击工具栏中的  图标。 将打开帮助编辑器。
2. 在域“向导 - 组”中，选择“实例”。
3. 在域“向导 - 名称”中，选择“演示向导”。
4. 在域“页面”中，选择“TriggerPage”。
5. 在“帮助 - 文本”域中输入下列文本： “选择触发器
   在此对话框中，可以指定执行行动的结果。”

6. 单击“确定”按钮关闭帮助编辑器。
7. 运行演示向导。 在“选择触发器”对话框中，单击“帮助”按钮。



将“Demo.wnf”脚本添加到数据库中

简介

为了在图形编辑器中使用动态向导函数“Demo.wnf”，它必须集成到动态向导的数据库。

为了完成此功能，有必要执行下列步骤，

1. 导入向导脚本
2. 创建 cwd 文件

步骤

1. 单击工具栏中的  图标。 文件选择对话框打开。
2. 选择“Demo.wnf”文件。 单击“打开”。
3. 单击工具栏中的  图标以创建新数据库。

4. 在动态向导编辑器的“视图”菜单中，选择“动态向导”。

5. 单击“实例”标签。 双击条目“演示向导”。

1.4.6.3 动态电机

动态电机

简介

在“Motor.wnf”脚本文件中，创建名为“Make Motor Dynamic”的动态向导。

说明

这是特别创建以使用户对象调用电机动态化，不能应用于任何其它种类的对象。

参见

使自定义对象“电机”动态化 (页 109)

将“Motor.wnf”脚本添加到数据库中 (页 108)

为电机创建“动态化向导功能” (页 107)

为电机创建“动态化向导功能”

要求

必须打开一个 WinCC 项目。

步骤

1. 打开在 Windows 资源管理器中目录“Siemens\ WinCC\ documents\ german”下的 Winzip 文件“Motor.zip”。
2. 解压缩“Motor.wnf”文件至目录“..\WinCC\wscripts\wscripts.deu”下。
3. 解压缩“Motor_dyn.pdl”文件至目录“..\WinCC\WinCCProjects\Name of the WinCCProject \GraCs”下。
4. 启动动态向导编辑器。
5. 在动态向导编辑器的“文件”菜单中，选择“打开”。 文件选择对话框打开。
6. 标记“Motor.wnf”文件。 单击“打开”。 “Motor.wnf”文件显示在编辑器窗口中。
7. 单击工具栏中的  图标，编译脚本。 结果显示在输出窗口中。

参见

将“Motor.wnf”脚本添加到数据库中 (页 108)

将“Motor.wnf”脚本添加到数据库中

简介

为了在图形编辑器中使用动态向导函数“Motor.wnf”，它必须集成到动态向导的数据库。

为了完成此功能，有必要执行下列步骤，

1. 导入向导脚本
2. 创建 cwd 文件

步骤

1. 单击工具栏中的  图标。 文件选择对话框打开。
2. 选择“Motor.wnf”文件。 单击“打开”。
3. 单击工具栏中的  图标以创建新数据库。

参见

使自定义对象“电机”动态化 (页 109)

使自定义对象“电机”动态化

简介

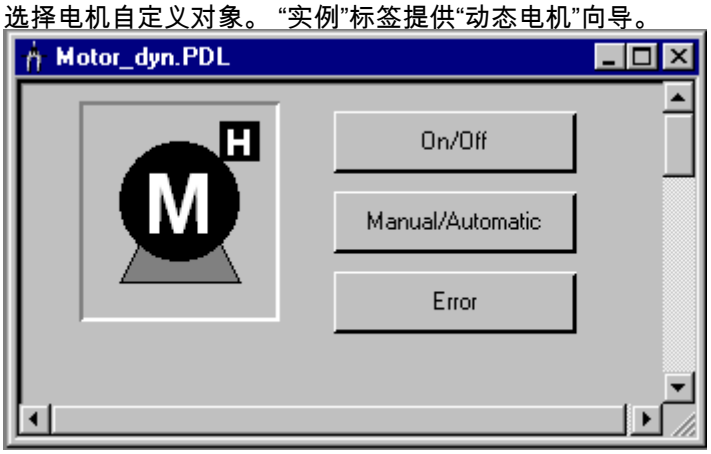
自定义对象“电机”将通过动态向导“电机动态化”链接到具有结构类型“MotorStruct”的 WinCC 结构变量中。 在此上下文中，各种 C 动作和变量连接在该对象上创建。 本向导不能用在其它对象类型上。

要求

- 创建“文本变量 8 位字符集”数据类型的内部变量“T08i_course_wiz_selected”。
- 创建名为“MotorStruct”的结构，并创建三个名称分别为“激活”、“手动”和“出错”的数据类型为“BIT”的内部元素。
- 创建一个数据类型为“MotorStruct”的名为“STR_Course_wiz1”的内部变量。

步骤

1. 打开图形编辑器。 选择菜单“文件”中的“打开”条目。 在文件选择对话框中，选择画面“Motor_dyn.pdl”。
2. 选择电机自定义对象。 “实例”标签提供“动态电机”向导。



3. 启动动态向导。 在对话框“欢迎使用动态向导”中，单击“继续”按钮。 “设置选项”对话框打开。
4. 在“设置选项”对话框，单击浏览按钮。 打开变量选择对话框。 选择“STR_Course_wiz1”作为结构变量。 单击“确定”按钮，关闭对话框。
5. 在“设置选项”对话框，单击继续按钮。 “完成！”对话框打开。 单击“确定”按钮，关闭对话框。
6. 保存该画面。 启动图形编辑器运行系统。
7. 按钮可以用来模拟所选择电机的变量值。

参见

创建结构和结构变量 (页 110)

创建结构和结构变量

简介

该部分将显示如何组态“MotorStruc”结构和“STR_Course_wiz1”结构变量。 结构和结构变量用于实例“电机动态化”中。

步骤

1. 从结构类型关联菜单中选择“新建结构类型”。 将显示结构属性对话框。
2. 重命名结构为“MotorStruc”。 单击“新建元素”，创建数据类型为 BIT 的内部变量“Active”。
3. 单击“新建元素”，创建数据类型为 BIT 的内部变量“Hand”。
4. 单击“新建元素”，创建数据类型为 BIT 的内部变量“Error”。 单击“确定”按钮，关闭对话框。
5. 在浏览框架中，单击变量管理器图标前的加号。 从内部变量关联菜单中选择“新建变量”。 创建数据类型为“MotorStruc”的 WinCC 变量“STR_Course_wiz1”。

1.5 文档阅读器

1.5.1 WinCC 文档阅读器

简述

WinCC 报表系统的打印作业可以另外传递到一个文件中。 对于较大量的数据，将为每一个报表页面创建一个文件。

WinCC 文档阅读器可以显示和打印这些文件。

1.5.2 安装 WinCC 文档阅读器

WinCC 文档阅读器可以用两种不同的方法安装：

步骤

1. WinCC 安装期间，从“程序”对话框选择“完整的 WinCC V7.0”。
   WinCC 将与智能工具、WinCC 组态工具和 WinCC 归档组态工具一起安装。

通过选择“SIMATIC”>“WinCC”>“工具”启动 WinCC 文档阅读器。

可选步骤

也可以从 WinCC DVD 安装 WinCC 文档阅读器。

1. 切换到 WinCC DVD 目录“InstData\WinCC\setup\Products\SC_SMARTTOOLS”。
2. 双击 setup.exe。
3. 在“组件”对话框中选择条目“WinCC 文档阅读器”。
4. 单击“继续”。按照对话框中的指示进行操作。

说明

如果 WinCC 项目被激活，只有该项目的“emf”文件可以进行查看和打印。如果 WinCC 未激活，那么可以用 WinCC 文档阅读器打开和打印所有的“emf”文件。

1.5.3 描述

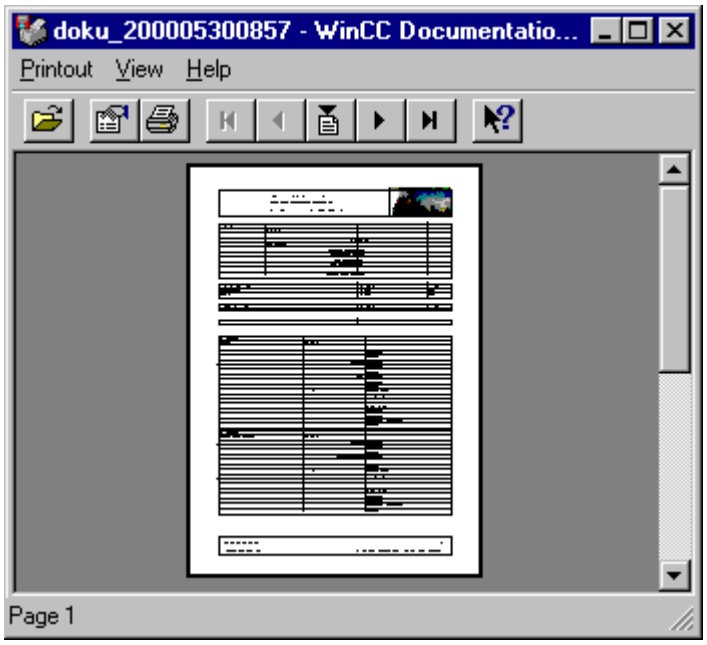
简介

打印作业可以重定向到文件中。对于较大量的数据，将为每一个报表页面创建一个文件。

WinCC 文档阅读器可以显示和打印这些文件。

说明

如果在启动 WinCC 文档阅读器时，一个 WinCC 项目已经被激活，那么只有该项目的“emf”文件可以显示和打印出来。
如果启动阅读器时 WinCC 已打开但未激活，那么可以用阅读器打开和打印所有的“emf”文件。
一旦禁用运行系统，则无论如何阅读器都会关闭。



WinCC 文档阅读器由三个区域组成。

屏幕的上边框包含菜单栏。菜单项在直接帮助中描述。

工具栏在菜单栏的正下方。频繁使用的功能，例如：前一页和下一页，载入为工具栏上的图标。单个图标的功能在直接帮助中描述。

窗口显示当前文档。通过单击可分两步扩大显示。

屏幕被显示当前操作信息的状态栏限制在底部。

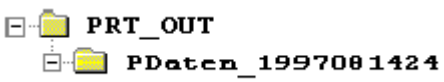
1.5.4 创建 .emf 文件

简介

打印作业可以另外传递到一个文件中。 对于较大量的数据，将为每一个报表页面创建一个文件。 打印输出被传递给一个和多个 .emf 文件。 文件名为 Page <nnnnnn>.emf 并存储在路径中，这里 <nnnnn> 代表连续的五位数字。

路径名称的组成如下： 项目路径（例如“C:\VFSWinCC\PRT\”）和 <storage> + <YYYYMMDDHHMM>（YYYY = 年，MM = 月，DD = 日，HH = 小时，MM = 分钟）。

如果在“Storage”域中输入“PDdata”，将在项目目录内为打印作业创建下列路径结构。



步骤

1. 在 WinCC 编辑器中，选择“文件”菜单中的“项目文档设置”命令。
2. 在“打印作业属性”对话框中，单击“打印机设置”标签。
3. 在“打印机设置”标签中，激活“文件（*.emf）”复选框。 如果不想同时输出到打印机，取消激活“打印机”复选框。
4. 在“Storage”域中，输入将存储此文件的路径名。 单击“确定”按钮，关闭对话框。
5. 选择菜单“文件”中的“打印项目文档”条目。 打印输出被传递给一个和多个 .emf 文件。 文件名为 Page <nnnnnn>.emf 并存储在路径中，这里 <nnnnn> 代表连续的五位数字。

1.6 WinCC 交叉索引助手

1.6.1 WinCC 交叉索引助手

简述

WinCC 交叉索引助手是一个在脚本中搜索画面名称和变量脚本并补充相关脚本的工具，以便使 WinCC 组件**交叉索引**查找画面名称和变量，并在交叉索引列表中列出它们。

1.6.2 安装交叉索引助手

WinCC 交叉索引助手有德语、英语和法语用户界面。

步骤

1. WinCC 安装期间，从“程序”对话框选择“完整的 WinCC V7.0”。
   WinCC 将与智能工具、WinCC 组态工具和 WinCC 归档组态工具一起安装。

通过选择“SIMATIC”>“WinCC”>“工具”启动 WinCC 交叉索引助手。

可选步骤

也可以从 WinCC DVD 安装 WinCC 交叉索引助手。

1. 切换到 WinCC DVD 目录“InstData\WinCC\setup\Products\SC_SMARTTOOLS”。
2. 双击 setup.exe。
3. 在“组件”对话框中选择“交叉索引助手”条目。
4. 单击“继续”。按照对话框中的指示进行操作。

1.6.3 常规

WinCC 能够创建交叉索引列表。 这样在创建这些列表时，函数调用中的变量能被正确识别，WinCC 由组态规则扩展，提供下列功能：

为搜索和替换在 C 动作中使用的变量和画面名称，必须如下编写脚本：

在脚本的开始处，必须分两段声明所有变量和画面名。 在段内，不必输入更多说明。

段的结构如下：

```
// WINCC:TAGNAME_SECTION_START
// syntax: #define TagNameInAction DMTagName
// next TagID : 1
#define ApcVarName1 "VarName1"
// WINCC:TAGNAME_SECTION_END

// WINCC:PICNAME_SECTION_START
// syntax: #define PicNameInAction PictureName
// next PicID : 1
#define ApcPictureName1 "PictureName1"
#define ApcPictureName2 "PictureName2"
#define ApcPictureName3 "PictureName3"
// WINCC:PICNAME_SECTION_END
```

启动读或写变量的标准功能必须通过定义的变量和画面进行。

```
GetTagDWord (ApcVarName1);
OpenPicture(ApcBildname1);
SetPictureName( ApcPictureName2, "PictureWindow1",ApcPictureName3);
```

如果不遵守组态规则，则不能创建交叉索引表，因为不能分辨脚本中变量和画面的引用。

借助于 WinCC 交叉索引助手，在脚本管理器中所知的所有函数调用由以上描述的格式替换。只有项目函数、画面和动作被转换。

WinCC 交叉索引助手的运行系统环境是 WinCC。 如果 WinCC 不运行或要转换的项目没有装载，则 WinCC 由 WinCC 交叉索引助手启动，或载入项目。

参见

已知的函数（脚本管理）(页 115)

1.6.4 已知的函数（脚本管理）

缺省情况下，下列函数由向导识别并且被转换：

将变量作为参数的函数：

GetTagBit()

GetTagByte()

GetTagChar()

GetTagDouble()

GetTagDWord()

GetTagFloat()

GetTagRaw()

GetTagSByte()

GetTagSDWord()

GetTagSWord()

GetTagWord()

SetTagBit()

SetTagByte()

SetTagChar()

SetTagDouble()

SetTagDWord()

SetTagFloat()

SetTagRaw()

SetTagSByte()

SetTagSDWord()

SetTagSWord()

SetTagWord()

GetTagBitWait()

GetTagByteWait()

GetTagCharWait()

GetTagDoubleWait()

GetTagDWordWait()

GetTagFloatWait()

GetTagRawWait()

GetTagSByteWait()

GetTagSDWordWait()

GetTagSWordWait()

GetTagWordWait()

SetTagBitWait()

SetTagByteWait()

SetTagCharWait()

SetTagDoubleWait()

SetTagDWordWait()

SetTagFloatWait()

SetTagRawWait()

SetTagSByteWait()

SetTagSDWordWait()

SetTagSWordWait()

SetTagWordWait()

GetTagBitState()

GetTagByteState()

GetTagCharState()

GetTagDoubleState()

GetTagDWordState()

GetTagFloatState()

GetTagRawState()

GetTagSByteState()

GetTagSDWordState()

GetTagSWordState()

GetTagWordState()

SetTagBitState()

SetTagByteState()

SetTagCharState()

SetTagDoubleState()

SetTagDWordState()

SetTagFloatState()

SetTagRawState()

SetTagSByteState()

SetTagSDWordState()

SetTagSWordState()

SetTagWordState()

GetTagBitStateWait()

GetTagByteStateWait()

GetTagCharStateWait()

GetTagDoubleStateWait()

GetTagDWordStateWait()

GetTagFloatStateWait()

GetTagRawStateWait()

GetTagSByteStateWait()

GetTagSDWordStateWait()

GetTagSWordStateWait()

GetTagWordStateWait()

SetTagBitStateWait()

SetTagByteStateWait()

SetTagCharStateWait()

SetTagDoubleStateWait()

SetTagDWordStateWait()

SetTagFloatStateWait()

SetTagRawStateWait()

SetTagSByteStateWait()

SetTagSDWordStateWait()

SetTagSWordStateWait()

SetTagWordStateWait()

将画面名称作为参数的函数：

SetPictureName()

GetPictureName()

GetVisible()

SetVisible()

GetLink()

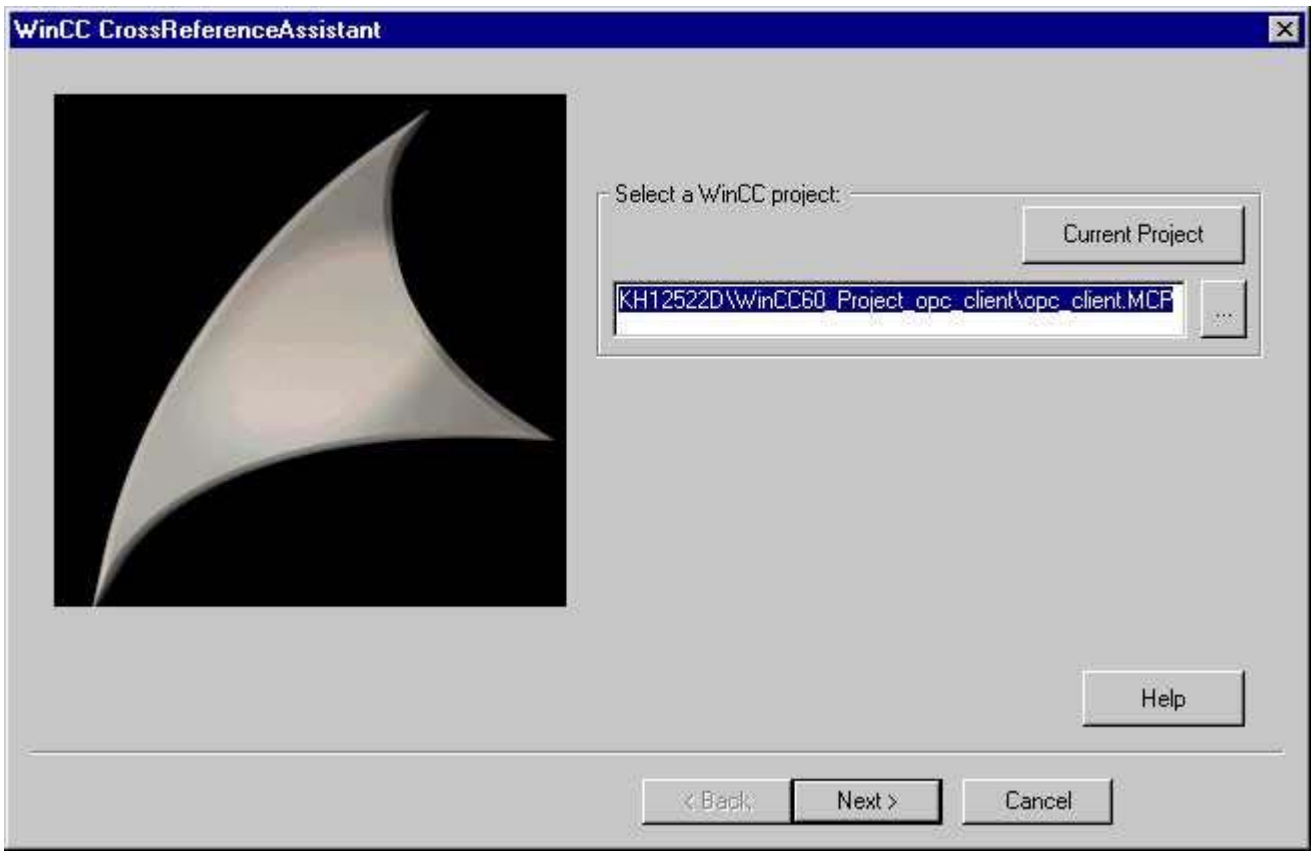
SetLink()

Set_Focus()

OpenPicture()

GetLinkedVariable()

1.6.5 项目选择



单击“**...**”打开 OpenFile 对话框，允许选择任何项目。 通过单击**当前项目**，WinCC 交叉索引助手设法导入和显示当前在 WinCC 中装载的项目。 如果 WinCC 不运行或没有项目装载，则它将启动或装载所需项目。

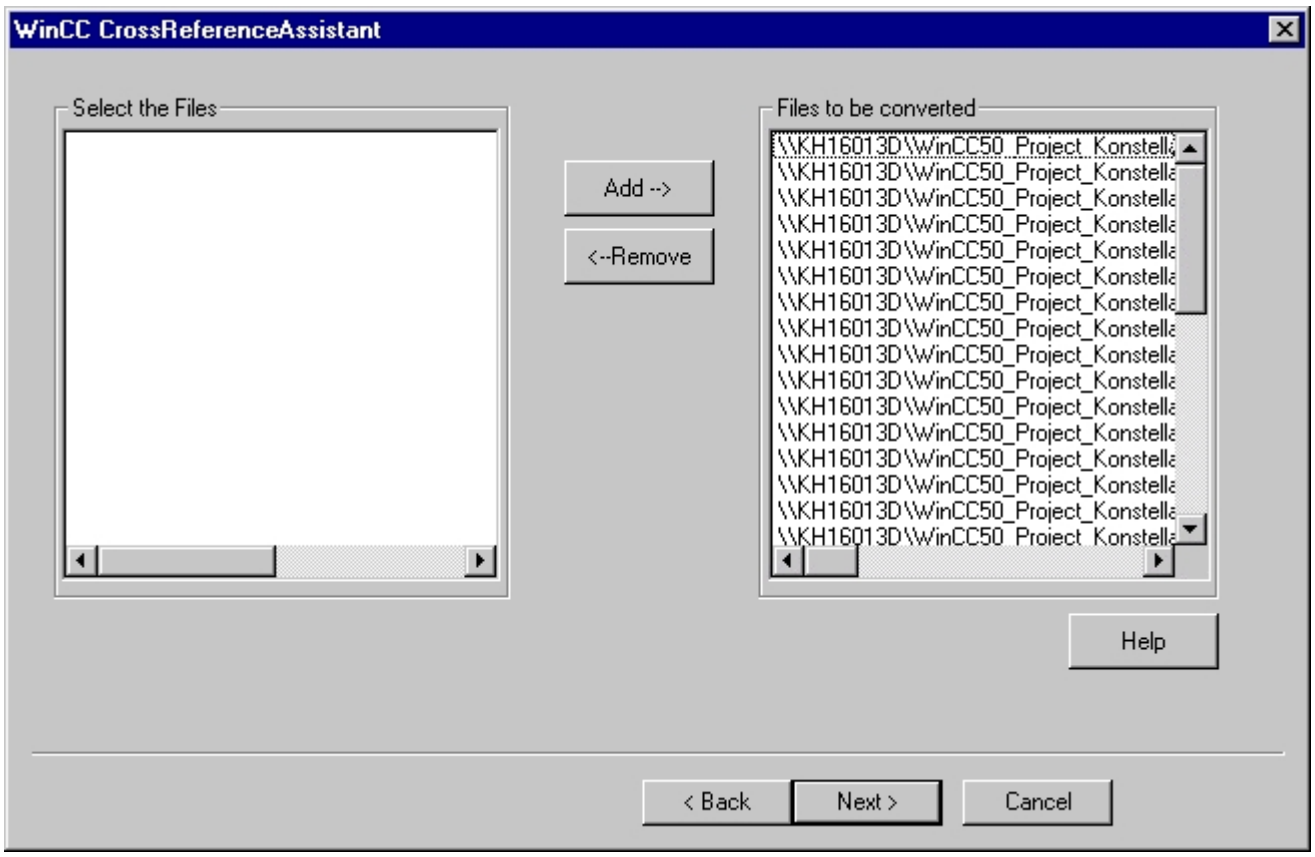
如果装载不同的项目，则它将关闭并装载所需项目。 此过程可能需要一些时间。

在所提示的“**选择 WinCC 项目**”输入行中输入文本后，可以单击按钮“**下一步 >**”。 然后检查指定的项目，以查看它是否是有效的 WinCC 项目。 如果项目不合法，在输入行上设置焦点，带有相关错误解释的消息窗口打开。

单击“**取消**”退出 WinCC 交叉索引助手。

1.6.6 文件选择

所有属于此项目的画面、项目函数和 C 动作都显示在对话框的右列表中。 在缺省设置中，所有属于项目的文件被转换。



用户可以决定排除对某些文件的转换，并可在以后将其添加。 通过在“**要转换的文件**”列表中（多项）选择相应的文件，然后单击“**<--删除**”按钮，将文件从转换列表中删除。

删除文件显示在左边列表中，并可以再次添加到转换中。 为此，列表必须在“**选择文件**”列表中选择。 在单击“**添加-->**”后这些文件将被添加到右边列表“**要转换的文件**”。

在选择文件后，单击“**下一步 >**”。 指定的文件即被读取和分析。

单击“**< 上一步**”返回项目选择。 单击“**取消**”退出 WinCC 交叉索引助手。

参见

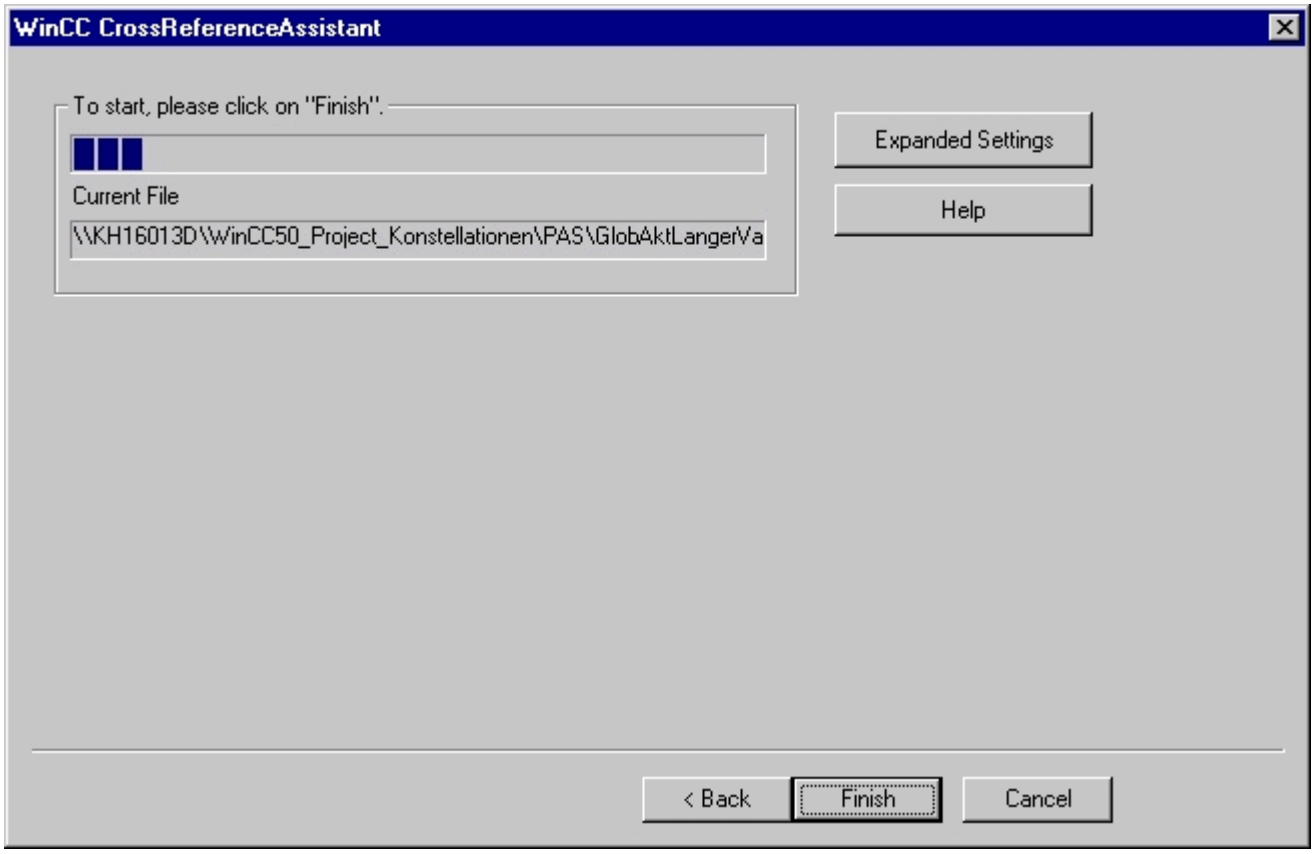
项目选择 (页 119)

1.6.7 转换

简介

在向导的最后一页，一方面可以进行“高级设置”（请参见“高级设置”），另一方面可以查看开始转换之后的进度和目前正在处理的文件。

描述



单击“< 上一步”后退到“文件选择”。 单击“取消”退出 WinCC 交叉索引助手。

要启动脚本转换，单击“完成”。 转换开始之后，不能后退（“< 上一步”）或单击“高级设置”。

转换期间，进度条显示转换完成百分比。还可以查看哪个文件当前正在被转换。

转换的执行方式如下：为需要画面或变量参数的函数调用检查脚本。如果在脚本中发现这样的函数，作为参数传递的字符串被定义替换（参见组态规则）。

脚本管理文件检查哪些函数需要画面或变量参数。这就是为什么所有这些函数必须在此文件中输入，并以此引入系统。脚本转换也可以用于扩展这些函数和同样需要画面和变量参数的项目函数和标准函数的列表（高级设置）。

在转换完成时，将显示有关画面中有多少函数、画面和脚本以及有多少变量已被转换的摘要信息。

如果出现错误，可以通过查看在转换期间创建的日志文件来查找更多关于出错原因的详细信息。此文件在项目目录中，名为 CCCrossReferenceAssistant.log。

参见

扩展设置 (页 122)

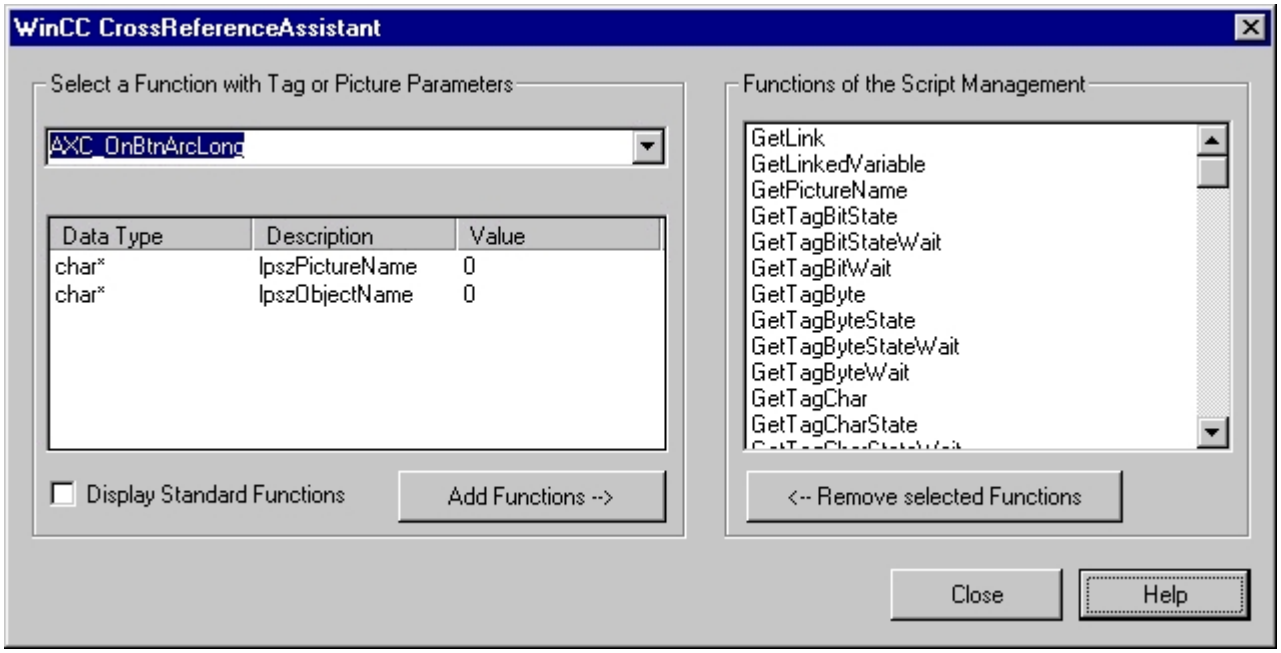
常规 (页 114)

文件选择 (页 120)

1.6.8 扩展设置

如果需要，可以激活自己创建的函数。

在“**选择有变量和画面参数的函数：**”显示列表、所有项目函数。 如果“**显示标准函数**”复选框激活，则 WinCC 中的缺省函数显示。



用户可以从函数列表选择变量或画面作为参数的函数。 此处选择的所有函数包括在项目指定的脚本管理文件中。

交叉索引助手只识别那些具有画面和变量参数的已被引入系统的函数。 所以需要变量参数的自定义函数的调用可以按照组态规则改变，必须在转换过程中包括在脚本管理中。

“**脚本管理函数**”对话框显示所有已在脚本管理中的函数。 显示此对话框时，标准和项目组态文件将被读取，并且两个文件的共同内容将被显示。

要命名需要变量或画面参数的函数，必须首先从“**选择有变量或画面参数的函数：**” 组合域中选择它。

然后使用参数列表，可以定义特殊参数代表变量还是画面。 通过单击“**...**”弹出式菜单打开用户可选择所选参数是变量还是画面。

对所有的参数都要重复此过程，以选择应用哪种标准。

""“**添加函数 -->**”确认输入并添加所选函数到对话框的右列表中。 如果操作有误，则可以撤消该操作，方法是在“**脚本管理函数**”列表中选择要删除的函数，然后单击“**<-- 删除所选函数**”将其从列表中删除。

通过单击“**关闭**”将组信息写入组态文件，并在转换时考虑改变的信息，然后对话框关闭。

参见

常规 (页 114)

1.7 WinCC 通讯组态器

WinCC 通讯组态器（CCComunicationConfigurator.exe）是可用简单的方式设置用于网络环境的 WinCC 通讯参数的一种工具。

若没有传输速率为 100 MBit/s 的以太网局域网，则应使用 WinCC 通讯组态器（Communication Configurator）。甚至在高负载所导致的连接偶尔不稳定情况时（比如未连接到数据服务器，I/O 域没有显示值），也推荐使用组态器。

WinCC 通讯用标准参数组态，所以它对于通讯出错反应非常灵敏，例如以便向用户快速报告发生的所有故障，还以便确保在客户机的情况下冗余服务器短暂的“错误结束”时间。

在具有低传送率或高网络/CPU 负载的网络上，WinCC 逻辑网络连接的稳定性受此出错敏感特性的影响，因为在设备状态监视的低水平机制下无法达到预期的反馈时间。

通讯组态器使通讯参数适应已存在的情况，以便保证出错灵敏性和连接稳定性之间具有最佳的协调。

说明

如果在这种方式下使用 WinCC 通讯组态器，在 WinCC 客户机和 WinCC 服务器上都必须这样做。
通讯组态器只修改 WinCC 通讯的设置，而不修改操作系统通讯链接的参数设置。

域/选项	描述
100 Mbps	适用于传输速率为 100 Mbps 的以太网局域网（缺省设置）。
10 Mbps	用于传输速率为 10 Mbps 的以太网局域网。
1 Mbps	用于传输速率为 1 Mbps 的网络。
0.1 Mbps	用于传输速率为 0.1 Mbps 的网络和通信链接。此设置适用于使用 ISDN（MultiLink）、ISDN 和调制解调器的链接。
“服务器连接客户机”复选框	通过服务器来检查与客户机的连接。
“缺省”按钮	将设定值设置为缺省值“以太网局域网（100 Mbps）”

说明

建议使用传输速率至少为 128 Kbps 的客户机。

1.8 WinCC 组态工具

1.8 资源

1.8.1 引言

简介

WinCC 组态工具提供简单、高性能的选件来组态 WinCC 中的批量数据。Microsoft Excel 用作用户界面。这允许用户在 Microsoft Excel 中创建 WinCC 项目，并利用 Microsoft Excel 所提供的有关操作的优点。

本章提供 WinCC 组态工具功能和操作该工具的概述。

概述

WinCC 组态工具提供简单、高性能的选件来组态 WinCC 中的批量数据。

Microsoft Excel 用作用户界面。因为其表格结构，它非常适合处理和表示 WinCC 数据。此外，它还提供了大量的编辑选项（包括自动填写等等）。另一个优点是有经验的用户可以通过创建 VBA 程序（宏）扩展编辑选项。

组态工具可以创建新的 WinCC 项目并且可以从一开始就使用 Excel 组态项目。此外，还可以读入现有的 WinCC 项目并在 Excel 中处理它们。本地组态数据用于此目的。只有没有其自身项目的客户机，才可以从远程读入。 必须在服务器的项目中输入组态工具正在其上运行的客户机的计算机名称。 必须已经分配了“远程组态”的权限。

组态在特殊类型的 Excel 工作薄（又称作 WinCC 项目文件夹）中执行。该工作薄包含各种类型的工作表，它们用于组态特定类型的 WinCC 对象。组态工具可用于组态来自数据管理器、报警记录、变量记录和文本库的数据。

说明

在组态工具中只能编辑 WinCC 中缺省包含的通道的链接或变量。

1.8.2 系统要求

简介

组态工具与 WinCC 和 Microsoft Excel 的系统要求相同。 但是，也可以选择不通过 WinCC 来使用组态工具。 在这种情况下，数据当然不能写入 WinCC。

系统要求

- Windows XP SP3/Windows Server 2003 (R2) SP2/Windows Vista SP1
- Microsoft Excel XP、Office 2003、Office 2007。必须安装 Visual Basic for Applications。
- Internet Explorer 6 或 7

说明

使用 Excel 时需要 Office 助手，以便输出组态工具的警告消息。

参见

组态工具中的数量结构实例 (页 237)

1.8.3 安装组态工具

WinCC 组态工具可以用两种不同的方法安装。

步骤

1. WinCC 安装期间，从“程序”对话框选择“完整的 WinCC V7.0”。
   WinCC 将与智能工具、WinCC 组态工具和 WinCC 归档组态工具一起安装。

通过选择“SIMATIC”>“WinCC”>“工具”启动 WinCC 组态工具。

可选步骤

也可以从 WinCC DVD 安装 WinCC 组态工具。

1. 切换到 WinCC DVD 目录“InstData\WinCC\setup\Products\ConfigurationTool”。
2. 双击 setup.exe。
3. 按照对话框中的指示进行操作。
   WinCC 组态工具即被安装。

1.8.4 接口

1.8.4.1 接口

简介

组态工具提供许多新功能。 还提供自己的工具栏和系统菜单条目。 新的菜单选项也添加到 Excel 菜单中。 在下列各章中详细说明上述菜单选项和功能。

参见

状态栏 (页 131)

工具栏 (页 126)

弹出式菜单 (页 130)

下拉菜单 (页 127)

1.8.4.2 工具栏

简介

组态工具的工具栏包含下列元素。

	创建项目文件夹 “新建项目文件夹”向导打开。 可通过向导创建新项目文件夹。
	切换语言 打开“选择语言”对话框。
	帮助 打开组态工具在线帮助。
	创建 WinCC 项目 “新建 WinCC 项目”向导打开。 向导可用于从现有的项目文件夹创建新 WinCC 项目。 项目文件夹链接到新 WinCC 项目。
	建立项目连接 仅在激活的项目文件夹已分配到 WinCC 项目并且 WinCC 项目未激活的情况下，该选项才可用。
	添加表格 “添加表格”对话框打开。 在该对话框中，可将工作表添加到项目文件夹。
	写入至 WinCC “写”对话框打开。 在该对话框中，可将项目文件夹中的所有数据写入到 WinCC。

1.8.4.3 下拉菜单

简介

下拉菜单中包含在各自的菜单标题下可用的选项。 组态工具还将单独的菜单项目添加到 Excel 菜单。

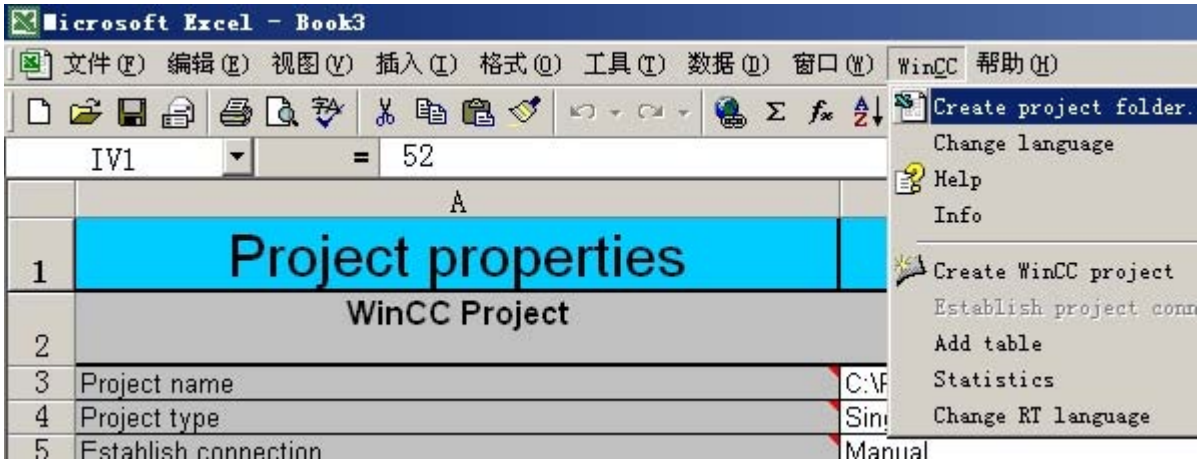
下拉菜单元素

下拉菜单提供了许多可选项，这些选项取决于连接状态和项目文件夹类型。

无项目文件夹处于激活状态

创建项目文件夹	“新建项目文件夹”向导打开。 可使用向导创建新项目文件夹。
切换语言	打开“选择语言”对话框。
帮助	打开组态工具在线帮助。

链接到 WinCC 项目的项目文件夹处于激活状态



创建 WinCC 项目	**“新建 WinCC 项目”向导打开。 可以使用该向导创建新的 WinCC 项目。项目文件夹链接到新 WinCC 项目。**
建立项目连接	在项目文件夹和关联的 WinCC 项目之间建立链接。
添加表格	“添加表格”对话框打开。 在该对话框中，可将工作表添加到项目文件夹。
改变 RT 语言	打开“改变 RT 语言”对话框。
写对象	根据激活工作表的类型，可能还会提供附加的菜单选项。 这些菜单选项主要用于写入可在激活工作表中组态的对象。

使用组态工具组态多种语言

使用 WinCC 组态工具可以更改 WinCC 界面语言以及运行系统语言，如同 WinCC 中一样：

- 要更改界面语言，请选择菜单选项“更改语言”。
- 要更改运行语言，请选择菜单选项“更改 RT 语言”。 只有项目文件夹连接到 WinCC 项目时，菜单项才可用。

例如，用户文本块和消息文本取决于运行语言。 切换运行语言时，文本以所选语言显示并可以扩展它们。

1.8.4.4 弹出式菜单

简介

W 组态工具向 Excel 的行弹出式菜单添加两个附加的菜单选项。 仅当在工作表中选择整行时，行弹出式菜单才可用。

WinCC - 写选择	仅当存在到关联 WinCC 项目的链接时，此菜单才可用。 “WinCC - 写选择”菜单选项用于将所有选择的对象写入到 WinCC。 这是唯一允许将单个对象写入到 WinCC 的选项。
WinCC - 删除选择	“WinCC - 删除选择”菜单选项用于删除所有项目文件夹和 WinCC 中选择的对象。 只有此菜单允许从 WinCC 删除单个对象。

1.8.4.5 状态栏

简介

组态工具中，Excel 的状态栏显示关于 WinCC 的信息。 如果项目在 WinCC 中打开，当前在 WinCC 中打开的 WinCC 项目的路径和名称显示在状态栏中。

1.8.5 操作 WinCC 组态工具

1.8.5.1 操作组态工具

简介

通常，Excel 提供的所有功能都可以不受限制地使用。 但是，排序和删除功能例外。它们由组态工具本身使用。

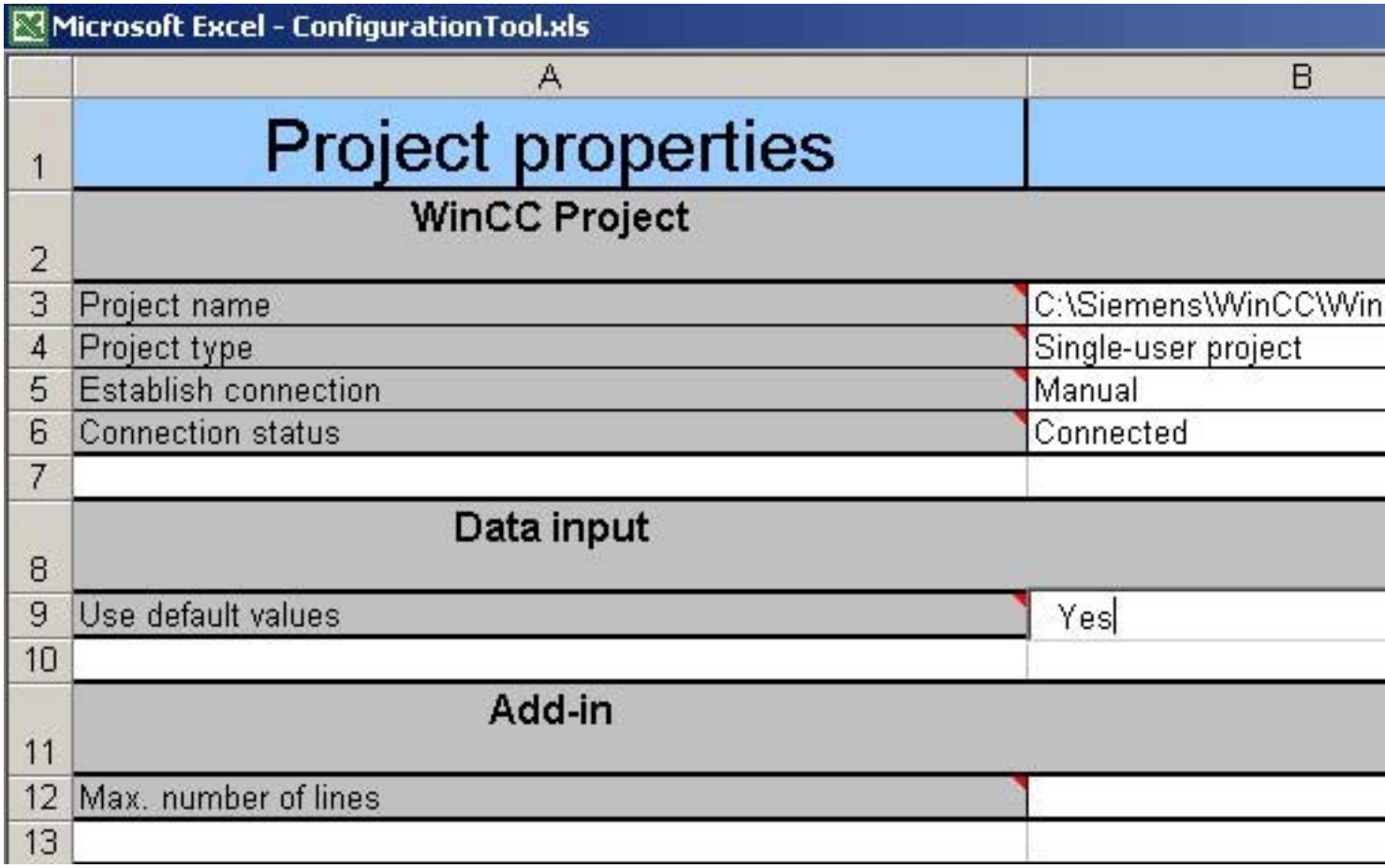
组态工具提供各种有用功能支持数据输入。

下拉列表框

通过双击表格中的各行，可访问下拉式列表框。 许多参数作为文本输入。 通过使用下拉式列表框输入参数，可确保输入的数值有效。

自动填写

自动填写是 Excel 标准功能，允许用户以一种非常有效的方式输入许多对象。 组态工具以两种方式支持自动填写操作。 在项目属性表中指定要使用的自动填写方法。



- 自动填写默认数值
  自动填写默认数值的优势在于：在各自的默认数值表中使用由组态工具自动提供的参数。
  只需要为变量表选择变量的名称和数据类型，其它参数从默认数值表中自动加载。 如果默认数值表中的数值是经过慎重选择的，那么此工具有助于节省大量时间。
- 不带默认数值的自动填写
  不带默认数值自动填写的优势在于：参数域不是由组态工具自动填写。 而是先检查单元格中的每个数值，如果发现是有效的就不重写。
  这一选项的实例应用是复制整个对象。

说明

有关自动填写工具的详细信息，请参考 Microsoft Excel 帮助。

唯一名称

组态工具检查对象名称是否唯一。 如果检测到已使用的名称，可以组态系统自动生成新的唯一的名称。 在这种情况下，连续的编号添加到原名称后。

然而如果要生成连续编号的对象名称，建议使用 Excel 为此用途设计的标准功能（输入第一个名称，例如 Tag_1，然后自动填写）。

输入提示

组态工具检查每个数据输入。 如果输入的数据无效，就自动更正。 组态工具执行的每一次改变都由输入提示进行注释。 这由 Office Assistant 显示。 输入提示无需确认而在下一次输入时自动消失。

说明

如果没有安装 Office Assistant，就不能使用输入提示。

1.8.5.2 创建新的项目文件夹

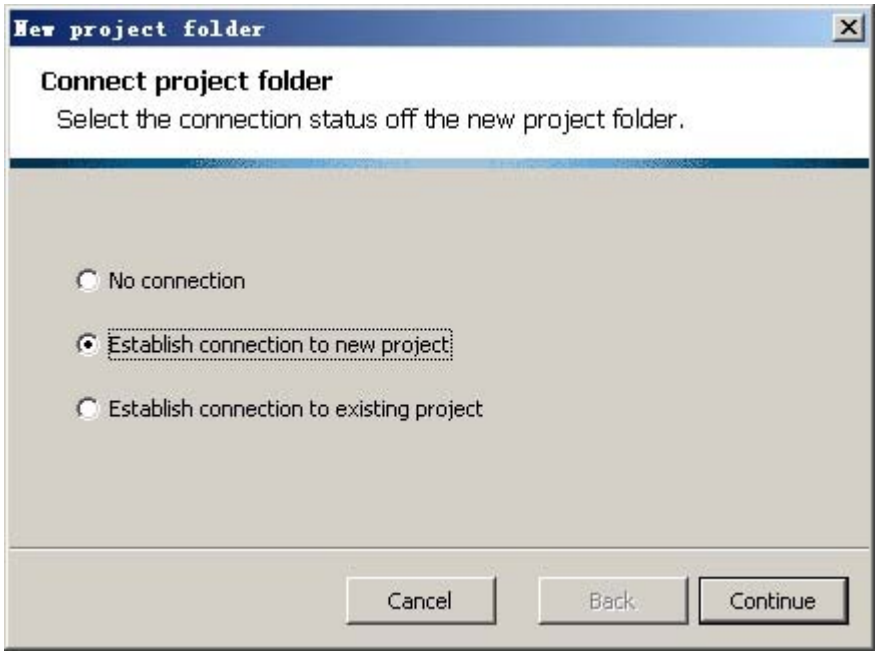
创建新的项目文件夹

简介

WinCC 项目的组态数据在项目文件夹中进行管理。 该文件夹包含各种类型的表格工作表，它们用于组态指定类型的 WinCC 对象。 使用工具栏或下拉菜单可以创建新的项目文件夹。

步骤

1. 单击工具栏中的  按钮。 “新建项目文件夹”向导打开。
2. 在向导的第一页上以下 3 个选项可用：
   无连接
   将创建不分配给任何 WinCC 项目的项目文件夹。
   建立至新项目的连接
   创建新的项目文件夹。 创建新的 WinCC 项目并将其分配给项目文件夹。
   建立至现有项目的连接
   创建新的项目文件夹。 文件夹分配给现有的 WinCC 项目。 读取 WinCC 项目数据。



3. 选择所需要的选件并按下“完成”按钮。 如果选择了选项“建立至新项目的连接”和“建立至已有项目的连接”，按“继续”按钮。
4. 此操作的步骤在下面的章节中描述。

参见

创建带有至现有 WinCC 项目的连接的项目文件夹 (页 135)

创建带有至新 WinCC 项目的连接的项目文件夹 (页 137)

创建不带连接的项目文件夹 (页 135)

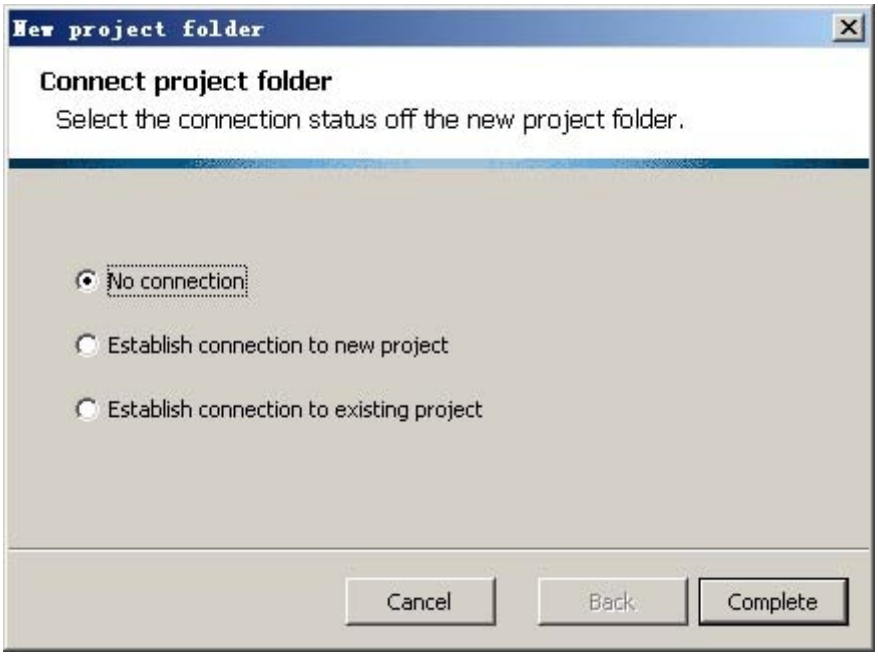
创建不带连接的项目文件夹

简介

组态工具可以创建不分配给 WinCC 项目的项目文件夹。 在不链接到 WinCC 项目的项目文件夹中，组态不受限制。 然而，不能向 WinCC 写数据。 为了将组态的数据写入 WinCC，必须创建新的 WinCC 项目。

步骤

1. 单击工具栏中的  按钮。 “新建项目文件夹”向导打开。
2. 选择选项“无连接”。 单击“完成”关闭向导。



参见

创建新 WinCC 项目 (页 139)

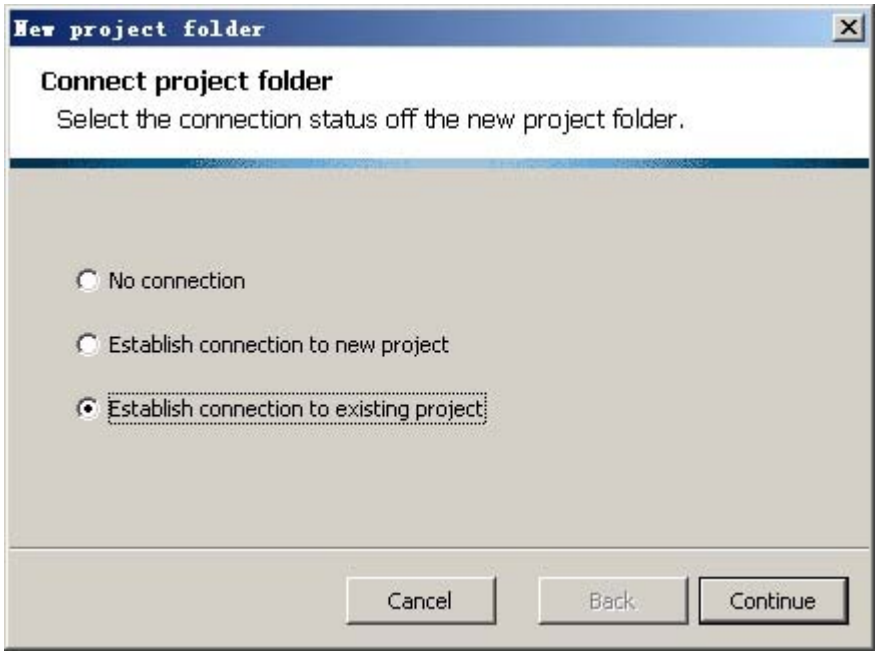
创建带有至现有 WinCC 项目的连接的项目文件夹

简介

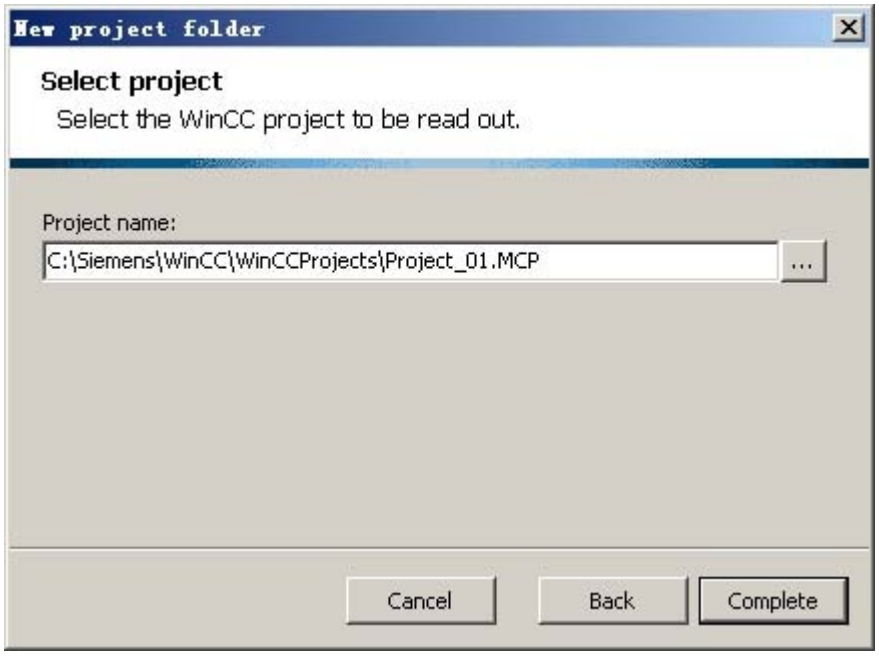
组态工具提供选项创建新的项目文件夹并将其分配给已经存在的 WinCC 项目。 将 WinCC 项目数据读入新的项目文件夹。

步骤

1. 单击工具栏中的  按钮。 “新建项目文件夹”向导打开。
2. 选择选项“建立至已有项目的连接”。 使用“继续”按钮打开向导的第二页。



3. 选择需要的 WinCC 项目。 单击“完成”关闭向导。



说明

读取大 WinCC 项目时，Microsoft Excel 会显示 OLE 消息框。 组态工具自动确认此框。 显示 OLE 消息框时，可能对性能有不良影响。

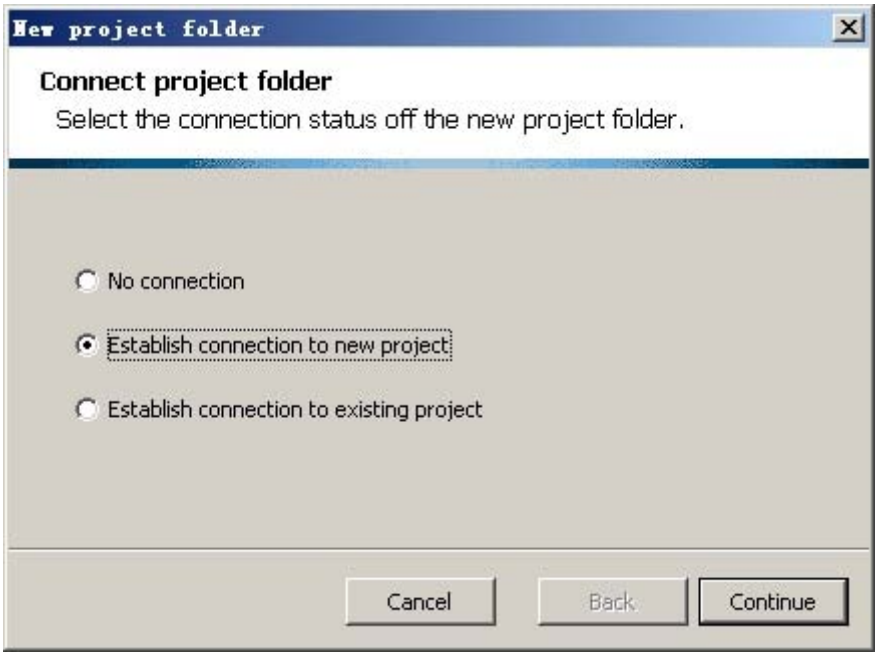
创建带有至新 WinCC 项目的连接的项目文件夹

简介

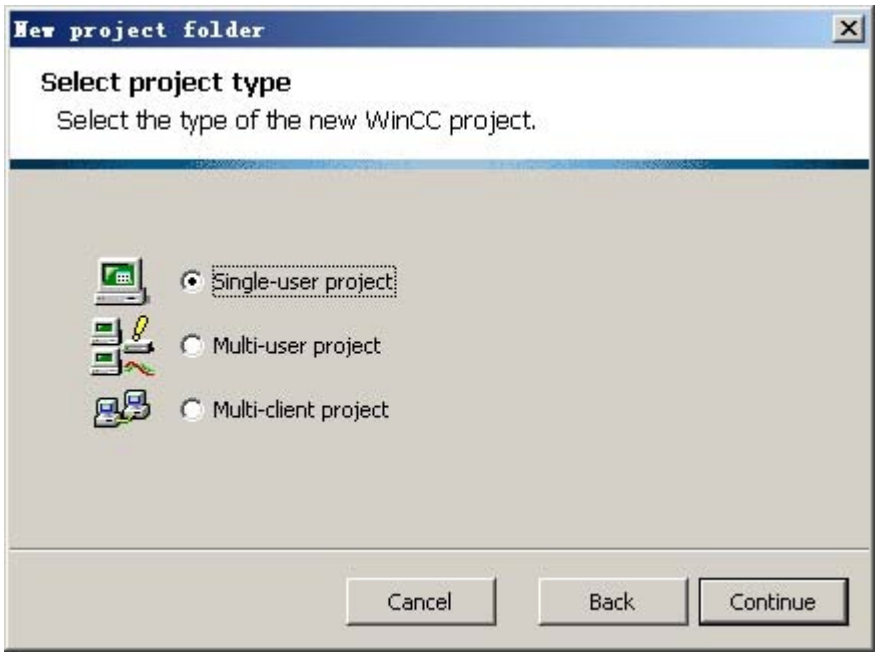
组态工具提供选项创建新的项目文件夹并将其分配给新的 WinCC 项目。 读取项目中已经提供的数据。

步骤

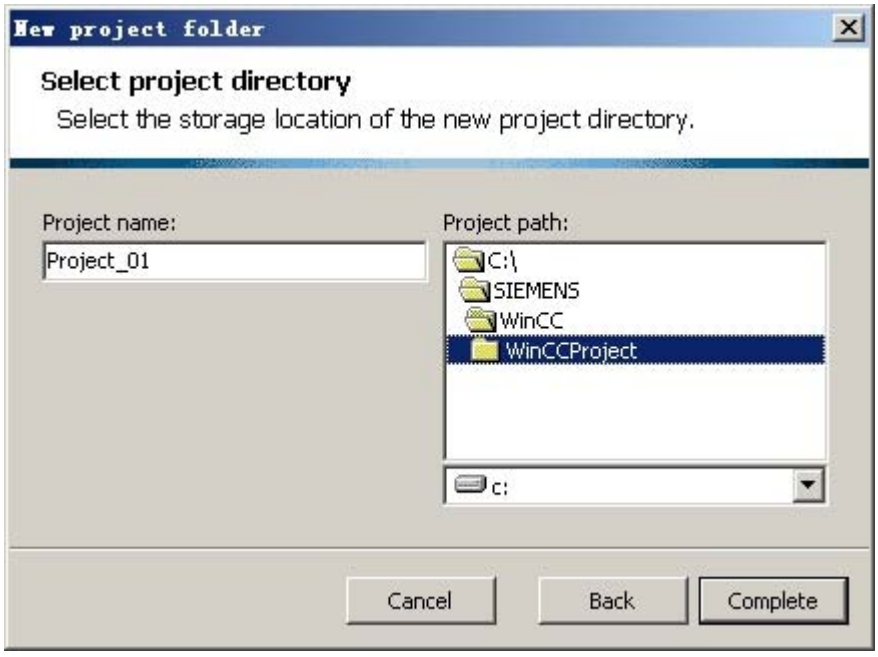
1. 单击工具栏中的  按钮。 “新建项目文件夹”向导打开。
2. 选择选项“建立至新项目的连接”。 使用“继续”按钮打开向导的第二页。



3. 选择新的 WinCC 项目类型。 使用“继续”按钮打开向导的第三页。



4. 在向导的第三页上，选择新 WinCC 项目的保存位置。 输入新 WinCC 项目的名称。 还指定要在其中创建项目文件夹的文件夹。 单击“完成”关闭向导。



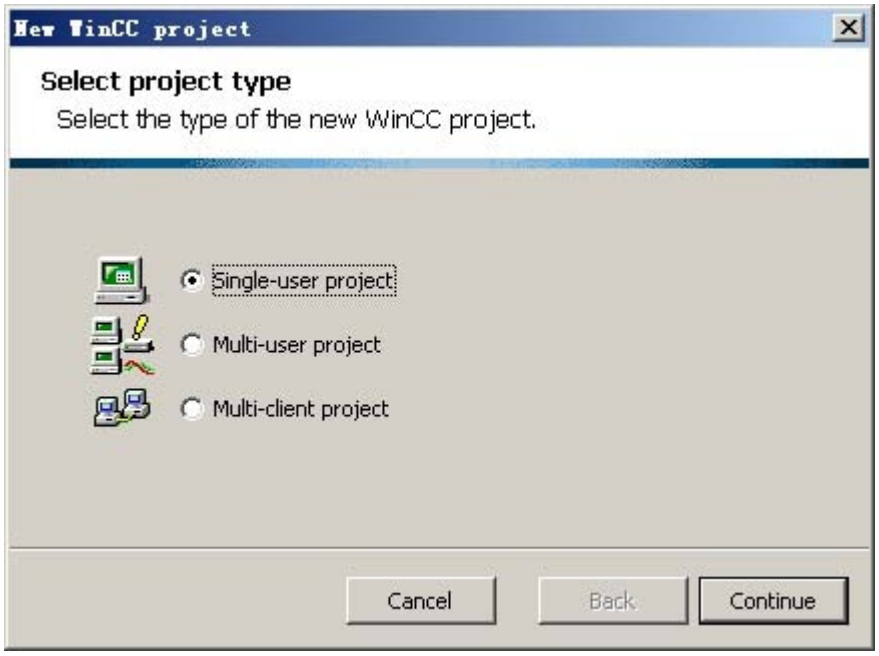
创建新 WinCC 项目

简介

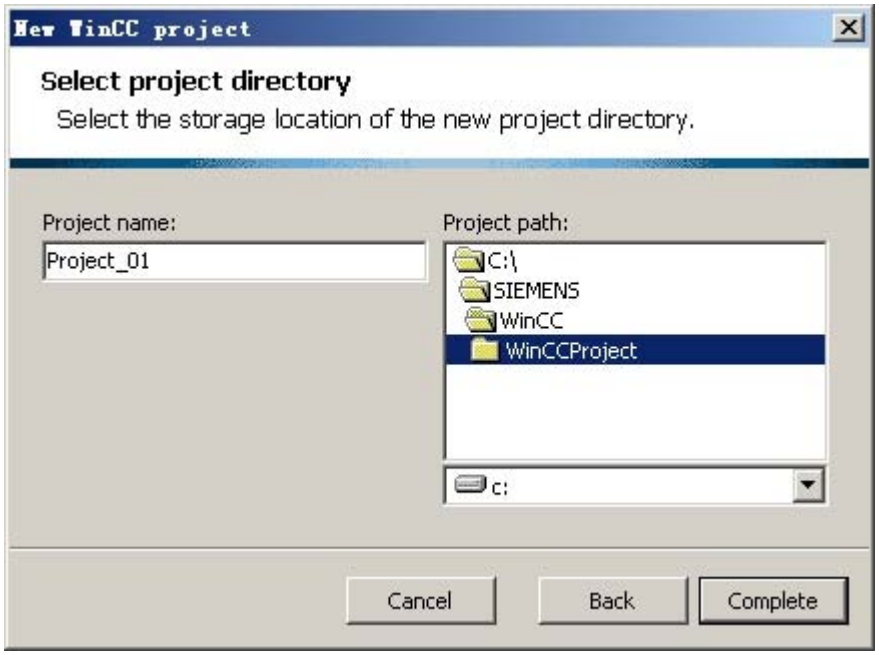
通过组态工具可以为现有的项目文件夹创建新 WinCC 项目。 WinCC 项目被分配给项目文件夹。

步骤

1. 单击工具栏中的  按钮。 “新建项目文件夹”向导打开。
2. 选择新的 WinCC 项目类型。 使用“继续”按钮打开向导的第二页。



3. 在向导的第二页上，选择新 WinCC 项目的保存位置。 必须输入新建 WinCC 项目的名称。 还指定要在其中创建项目文件夹的文件夹。 单击“完成”关闭向导。



1.8.5.3 工作表

工作表

简介

创建新的项目文件夹时，为每个所需的表单类型至少创建一个副本。 用户还可以添加附加表单。

工作表结构

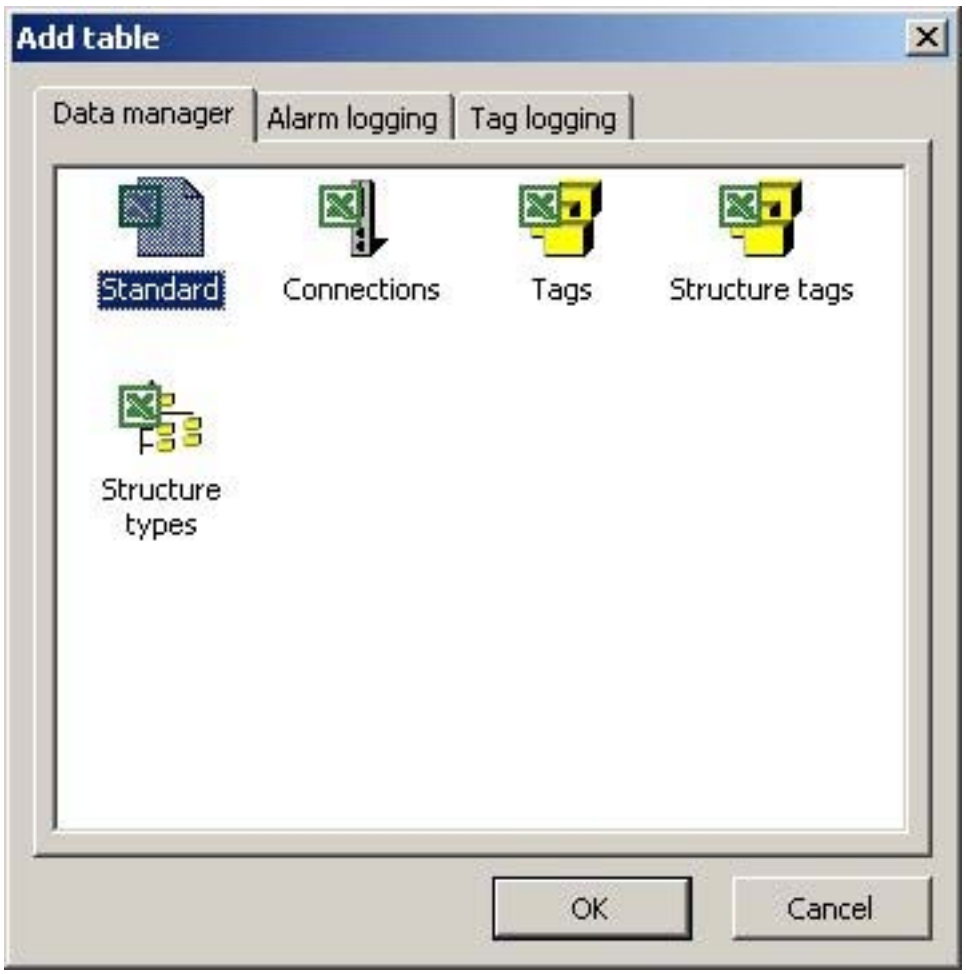
对于每个要组态的对象，在表格中都为其分配一行。 彩色行是标题，灰色单元格为参数标题。 参数标题下面的单元格组成数据区。 根据需要，数据区之外的单元格可以使用。

Microsoft Excel - ConfigurationTool.xls

	A	B	C	
1	Tags			
2	Name	Data type	Length	Fo
3	Tag_1	Binary tag	1	
4	Tag_2	8-bit value with sign	1	
5	Tag_3	8-bit value without sign	1	
6	Tag_4	Text tag, 8-bit font	0	
7	Tag_5	Floating point number 32-bit IEEE	4	
8	Tag_6	Raw data type	0	
9	Tag_7	Floating point number 64-bit IEEE	8	Dou
10	Tag_8	16-bit value with sign	2	Sho

创建新的表单

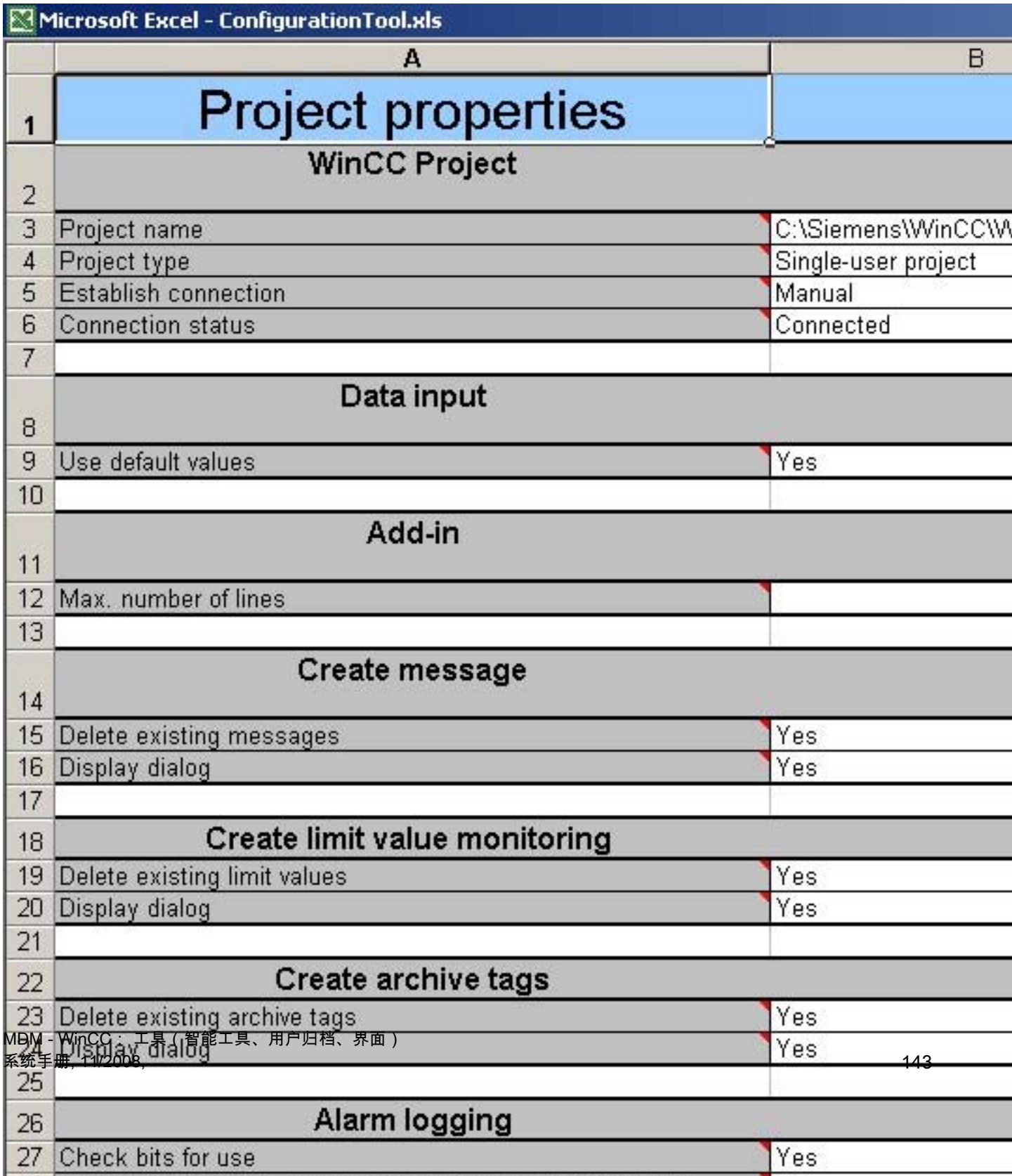
通过单击工具栏中相应的按钮或从下拉菜单中选择来打开“添加表格”对话框。 在该对话框中选择所需表格类型。 单击“确定”将表格添加到项目文件夹。



操作“项目属性”表单

简介

“项目属性”表单包含有关 WinCC 项目的信息。此外，可以在此定义影响整个项目文件夹的设置。


Microsoft Excel - ConfigurationTool.xls

	A	B
1	Project properties	
2	WinCC Project	
3	Project name	C:\Siemens\WinCC\W
4	Project type	Single-user project
5	Establish connection	Manual
6	Connection status	Connected
7		
8	Data input	
9	Use default values	Yes
10		
11	Add-in	
12	Max. number of lines	
13		
14	Create message	
15	Delete existing messages	Yes
16	Display dialog	Yes
17		
18	Create limit value monitoring	
19	Delete existing limit values	Yes
20	Display dialog	Yes
21		
22	Create archive tags	
23	Delete existing archive tags	Yes
24	Display dialog	Yes
25		
26	Alarm logging	
27	Check bits for use	Yes

WinCC 项目

项目名称	所连接 WinCC 项目的路径和项目文件。
项目类型	项目类型。
建立连接	可以在“手动”和“立即打开”之间进行选择。 如果选择“手动”，则在打开项目文件夹后，使用工具栏按钮或下拉菜单建立至关联 WinCC 项目的连接。 如果选择“立即打开”，则在打开项目文件夹时，至关联 WinCC 项目的连接会自动建立。
连接状态	指示关联的 WinCC 项目是否打开。只有建立了至项目的连接时，数据才能被写入 WinCC。

数据输入

使用默认值	可以定义是否使用默认值。 如果是这种情况，则在组态期间使用来自默认值表格的默认值。 如果不是这种情况，则不使用默认值。 使用默认值可显著提高性能。

添加项

最大行数	这用于定义行数,其后应在读出数据时创建新表。 限制行数可改善性能，因为 Excel 需要相当长的时间来从某些行创建新数据。

创建消息

删除现有的消息	在此处指定为所选变量生成变量表消息时是否删除已经存在的消息。
显示对话框	在此处定义是否要使用该对话框生成变量表消息。 如果不是，则使用“报警记录默认值”表单中定义的设置。

创建限制值监控

删除现有的限制值	在此处定义在从变量表生成限制值时是否删除所选变量的现有限制值。
显示对话框	在此处定义对话框是否应该用于从变量表中创建限制值。 如果不是，则使用“报警记录默认值”表单中定义的设置。

创建归档变量

删除现有的归档变量	在此处定义在从变量表中创建归档变量时是否删除所选变量的现有归档变量。
显示对话框	指定对话框是否用于从变量表中创建归档变量。 如果您禁用此对话框，则此对话框将打开一次。 在此对话框中输入过程值归档作为缺省归档。 勾选“保留设置”选项，以便将来不再打开此对话框。将使用在“变量记录默认值”表中定义的设置。

报警记录

检查使用的位	在此处定义 WinCC 组态工具是否检查报警记录中的变量位组合。
提示显示修改所有使用的状态文本的请求	此选项用于定义在对消息类别中的状态文本进行修改之后，是否显示提示请求修改所有完全相同的状态文本。
提示显示修改所有使用的消息文本的请求	此选项用于定义在对消息文本进行修改之后，是否显示提示请求修改所有完全相同的消息文本。
删除不使用的文本	此选项用于定义在从报警记录系统中删除对象时，不使用的文本是否从文本库中自动删除。
在删除单个消息时删除限制值	此选项用于定义在删除设置的单个消息时，限制值是否也要删除或者是否应该设置缺省消息号。

注释

显示注释	在此处定义注释是否显示在“项目属性”表单上。

数据管理器

数据管理器

简介

连接、变量、结构变量和结构类型可以在数据管理器表单上进行组态。 也可以在“数据管理器缺省值”表单中为变量定义缺省值。 在以下章节中说明在数据管理器中组态数据的步骤。

参见

操作“连接/组”表单 (页 150)

操作“结构变量”表单 (页 155)

操作“结构类型”表单 (页 158)

操作“变量”表单 (页 152)

操作“数据管理器默认值”表单 (页 146)

操作“数据管理器默认值”表单

简介

组态工具提供选项将默认值预分配给新创建对象的参数。这些默认值可以在“数据管理器默认值”表单上定义。

此处定义的设置也用于结构类型元素。

步骤

可为变量设置的每种数据类型都有一行可用。在此行中定义相关数据类型的设置。

Microsoft Excel - ConfigurationTool.xls

	A	B	
1	Default values	Data manager	
2	Tags		
3	Data type	Length	Fo
4	Binary tag	1	
5	8-bit value with sign	1	
6	8-bit value without sign	1	
7	16-bit value with sign	2	
8	16-bit value without sign	2	
9	32-bit value with sign	4	
10	32-bit value without sign	4	
11	Floating point number 32-bit IEEE	4	
12	Floating point number 64-bit IEEE	8	
13	Text tag, 8-bit font	0	
14	Text tag, 16-bit font	0	
15	Text reference	4	
16	Raw data type	0	

禁用默认值

默认值的使用可以在“项目属性”表单中禁用。如果禁用默认值，则检查新创建对象所有参数的有效性，并在必要时进行更正（例如空行）。然而，这样在创建对象时会降低性能。因此，默认情况下会激活默认值的使用。

表格结构

下列表格列出必须为“数据管理器默认值”表单中变量定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
数据类型	可用于变量的所有数据类型。该列是写保护的。
长度	WinCC 中以字节为单位的变量长度。对于大部分数据类型来说，不能编辑此数值。
 类型转换	变量的类型转换。只有在外部变量的情况下才能调整。 不是所有数据类型均分配类型转换功能。
 连接	变量的连接。
 组	变量的组。该参数是可选的。
地址	变量的地址。只有在外部变量的情况下才能调整。地址的结构取决于为连接设置的通讯驱动程序。有关地址的结构，请参见 WinCC 项目管理器的参数列中的变量。当前不检查地址的有效性。所有条目都视为有效。
 更新	变量的更新。此选项只可用于内部变量。可以在“用于整个项目”和“本地相关计算机”选项之间进行选择。
 线性标定是/否	在此处指定是否使用线性标定。只有对外部变量才能选择线性标定。不是所有的数据类型都支持线性标定。
线性标定过程从	过程中的标定范围（源标定）。
线性标定过程至	过程中的标定范围（源标定）。
线性标定变量从	变量的标定范围（标定的表达式）。
线性标定变量至	变量的标定范围（标定的表达式）。
上限	变量的上限值。
下限	变量的下限值。
起始值	变量的起始值。

列	简述
替换值	变量的替换值。只能为外部变量组态替换值。
 上限替换值	如果实际值高于上限值，则使用替换值。
 下限替换值	如果实际值低于下限值，则使用替换值。
 替换起始值	以替换值代替起始值。
 故障替换值	在连接故障的情况下使用替换值。
变量同步	一旦在其中一台冗余服务器上修改了一个内部变量，就会在伙伴计算机上比较该变量。

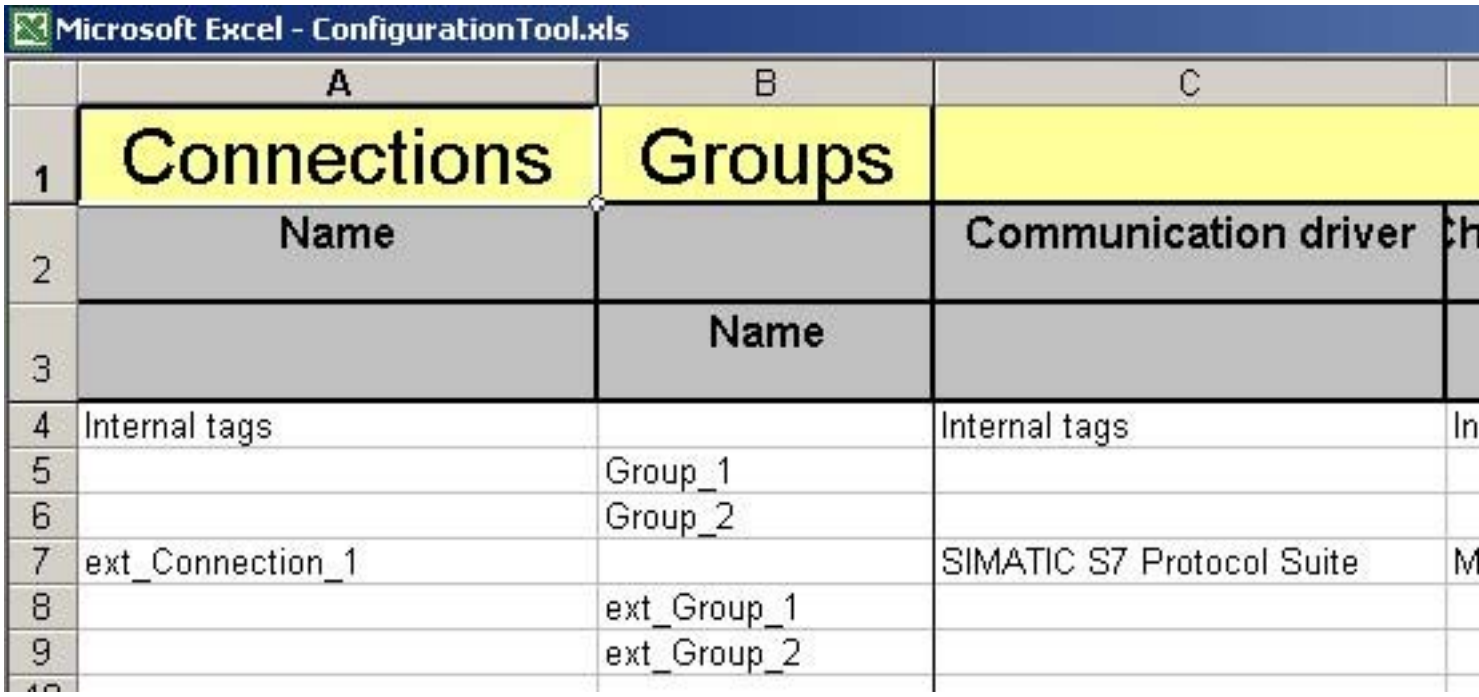
参见

操作“项目属性”表单 (页 143)

操作“连接/组”表单

简介

连接表单用于组态两个不同类型的 WinCC 对象。这将涉及到连接和分配给连接的组。连接和组的逻辑分配由表格中各个对象的位置来确定。



	A	B	C
1	Connections	Groups	
2	Name		Communication driver
3		Name	
4	Internal tags		Internal tags
5		Group_1	
6		Group_2	
7	ext_Connection_1		SIMATIC S7 Protocol Suite
8		ext_Group_1	
9		ext_Group_2	

步骤

连接

为了创建一个新连接，必须为连接分配一个名称。这样，连接就变为有效对象，并可写入 WinCC。可给连接参数分配默认值。默认值不能被修改。

组

只能将组添加到现有的连接上。 为了组态组，只能为组分配一个名称。

写入

始终将组与其关联的连接一起自动写入 WinCC。为了写入组，必须写入关联的连接。

删除

删除连接时，所有的从属组连同它一起被删除。此外，所有连接变量也被删除。“内部变量”连接不能被删除。删除组时，组中的所有变量也被删除。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

下列表格列出必须为“连接/组”表单中的连接设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
名称	连接的名称。名称必须唯一。
 通讯驱动程序	连接的通讯驱动程序。
 通道单元	连接的通道单元。
参数	连接的参数字符串。参数字符串的结构取决于所选的通讯驱动程序。当前不检查参数字符串的有效性。所有条目都视为有效。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

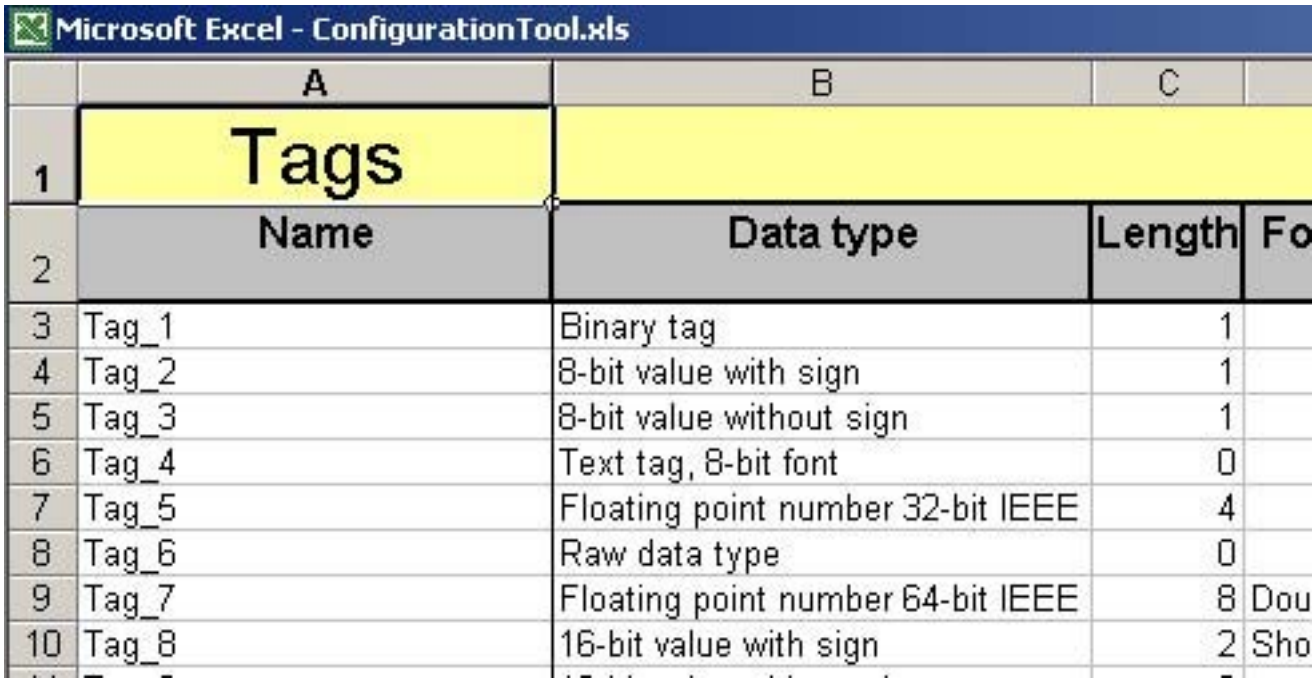
下列表格列出必须为“连接/组”表单中的组设置的所有参数。

列	简述
名称	组的名称。名称必须唯一。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

操作“变量”表单

简介

所需的变量可以在“变量”表单中组态。组态结构变量时，其结构实例元素在变量表中自动创建。


Microsoft Excel - ConfigurationTool.xls

	A	B	C	
1	Tags			
2	Name	Data type	Length	Fo
3	Tag_1	Binary tag	1	
4	Tag_2	8-bit value with sign	1	
5	Tag_3	8-bit value without sign	1	
6	Tag_4	Text tag, 8-bit font	0	
7	Tag_5	Floating point number 32-bit IEEE	4	
8	Tag_6	Raw data type	0	
9	Tag_7	Floating point number 64-bit IEEE	8	Dou
10	Tag_8	16-bit value with sign	2	Sho

步骤

为了创建一个新变量，必须为其分配一个名称。然而，仅分配名称不表示该变量是有效对象，或者可将其写入 WinCC。

如果没有指定变量名称，则不能编辑其它变量参数。在指定变量名之后，只能编辑数据类型。只有为变量分配数据类型之后它才成为有效对象。

如果已经激活了用默认值填写，那么给变量的其它归档参数分配默认值表格中定义的数值。

写入

一旦变量变成有效对象，就可以将它写入 WinCC。结构实例元素不能从变量表写入。它们与关联的结构变量一起自动写入。

删除

删除变量时，也删除为变量组态的所有限制值监视设置。对于其它对象来说，到该变量的交叉引用也被删除。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

特殊功能

组态工具允许组态单个消息、限制值监视和变量表的归档变量。变量表的行弹出式菜单提供许多选择选项。通过变量表的行弹出式菜单创建对象的过程在以下章节中进行了说明。

表格结构

下列表格列出必须为“变量”表单中的变量设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
名称	变量的名称。名称必须唯一。该名称可能与现有结构变量的名称不一致。
 数据类型	可用于变量的所有数据类型。
长度	WinCC 中以字节为单位的变量长度。对于大部分数据类型来说，不能编辑此数值。
 类型转换	变量的类型转换。只有在外部变量的情况下才能调整。不是所有的数据类型都分配格式改编功能。
 连接	变量的连接。
 组	变量的组。该参数是可选的。

列	简述
地址	变量的地址。只有在外部变量的情况下才能调整。地址的结构取决于为连接设置的通讯驱动程序。有关地址的结构，请参见 WinCC 项目管理器的参数列中的变量。当前不检查地址的有效性。所有条目都视为有效。
 更新	变量的更新。此选项只可用于内部变量。可以在“用于整个项目”和“本地相关计算机”选项之间进行选择。
 线性标定是/否	在此处指定是否使用线性标定。只有外部变量才能选择线性标定。不是所有的数据类型都支持线性标定。
线性标定过程从	过程中的标定范围（源定标）。
线性标定过程至	过程中的标定范围（源定标）。
线性标定变量从	变量的标定范围（标定的表达式）。
线性标定变量至	变量的标定范围（标定的表达式）。
上限	变量的上限值。
下限	变量的下限值。
起始值	变量的起始值。
替换值	变量的替换值。只能为外部变量组态替换值。
 上限替换值	如果实际值高于上限值，则使用替换值。
 下限替换值	如果实际值低于下限值，则使用替换值。
 替换起始值	以替换值代替起始值。
 故障替换值	在连接故障的情况下使用替换值。
变量同步	一旦在其中一台冗余服务器上修改了一个内部变量，就会在伙伴计算机上比较该变量。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配错误文本“OK”。

说明

如果使用映射器从 STEP7 在 WinCC 的变量管理中创建外部变量，也不能使用 WinCC 组态工具更改连接关联。

参见

从变量表创建归档变量 (页 220)

从变量表创建限制值监视 (页 227)

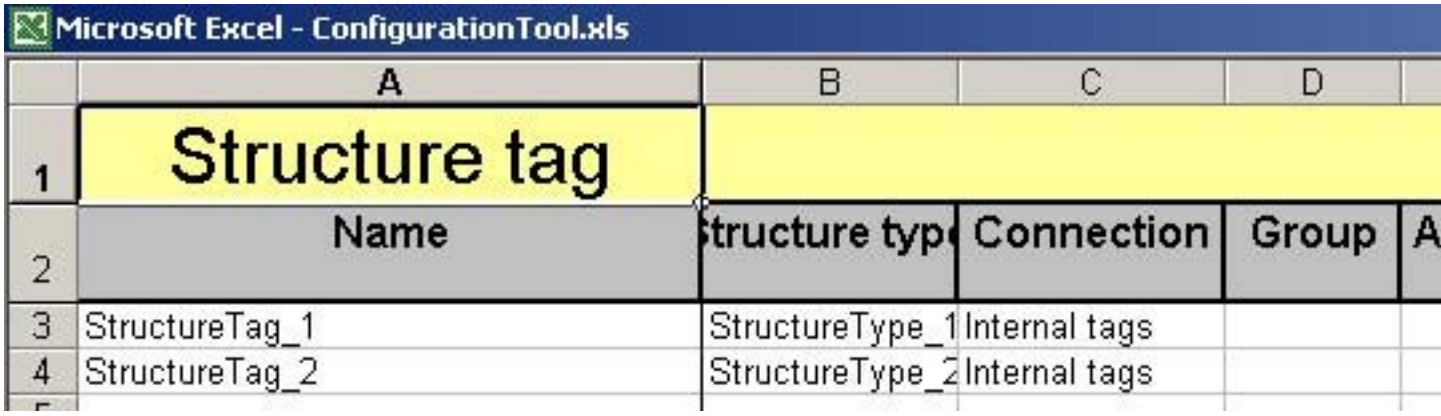
从变量表创建单个消息 (页 223)

Simatic S7 Protocol Suite 的地址串 (页 243)

操作“结构变量”表单

简介

所需的结构变量可以在“结构变量”表单中组态。 为此，在项目文件夹中必须至少已经组态了一种结构类型。



步骤

为了创建新的结构变量，必须为其分配一个名称。 然而，仅分配名称不表示该结构变量是有效对象，或者可将其写入 WinCC。

如果没有指定名称，也就不能编辑其它结构变量参数。 在指定变量名称之后，只能编辑数据类型。 结构变量在分配数据类型之后成为有效对象。

定义结构类型后，结构变量所需的结构实例元素自动在变量表上创建。 在对结构变量进行修改之后，其结构实例元素自动更新。 这也可以应用在结构类型本身的修改上。

当添加新结构元素或更改结构元素的数据类型时，需要输入此类型的所有结构变量的地址。

结构实例元素

创建结构变量时，其结构实例元素在变量表中自动创建。 结构实例元素与普通变量的不同之处在于其大多数参数是写保护的。 只能设置地址和更新，以及上限、下限、起始值和替换值。

说明

在以下 SIMATIC 通道中，结构实例元素的地址从结构变量的起始地址自动生成：

- SIMATIC S7 Protocol Suite
- SIMATIC S5 Ethernet Layer 4
- SIMATIC S5 PMC Ethernet
- SIMATIC TI Ethernet Layer 4
- SIMATIC S5 PMC Profibus
- SIMATIC S5 Serial 3964R
- SIMATIC S5 Programmers Port AS511

写

如果在尝试向 WinCC 写入结构变量时，设置的结构类型在 WinCC 中不存在或其参数在 WinCC 中未更新，将出现一提示询问是否应该也写入该结构类型。 如果对提示回答“否”，则所选择的结构变量不写入 WinCC。

结构实例元素也自动写入 WinCC。

删除

删除结构变量时，所有相关的结构实例元素也被删除。

说明

数据从项目文件夹和 WinCC 中永久删除。 数据的删除不能撤消。

表格结构

下列表格列出必须在“结构变量”表单上设置的所有参数。 带有下拉式列表框的参数用  图标标识。

列	简述
名称	结构变量的名称。 名称必须唯一。 该名称可能与现有变量的名称不一致。
 结构类型	结构变量的结构类型。
 连接	结构变量的连接。
 组	结构变量的组。 该参数是可选的。
地址	结构变量的地址。 只能为外部结构变量定义地址。 地址的结构取决于为连接设置的通讯驱动程序。 有关地址的结构，请参见 WinCC 项目管理器的参数列中的变量。 当前不检查地址的有效性。 所有条目都视为有效。
 更新	结构变量的更新。 只能为内部结构变量设置此参数。 可以在“用于整个项目”和“本地相关计算机”选项之间进行选择。
错误文本	写入 WinCC 时出现的错误简述。 无错误对象分配错误文本“OK”。

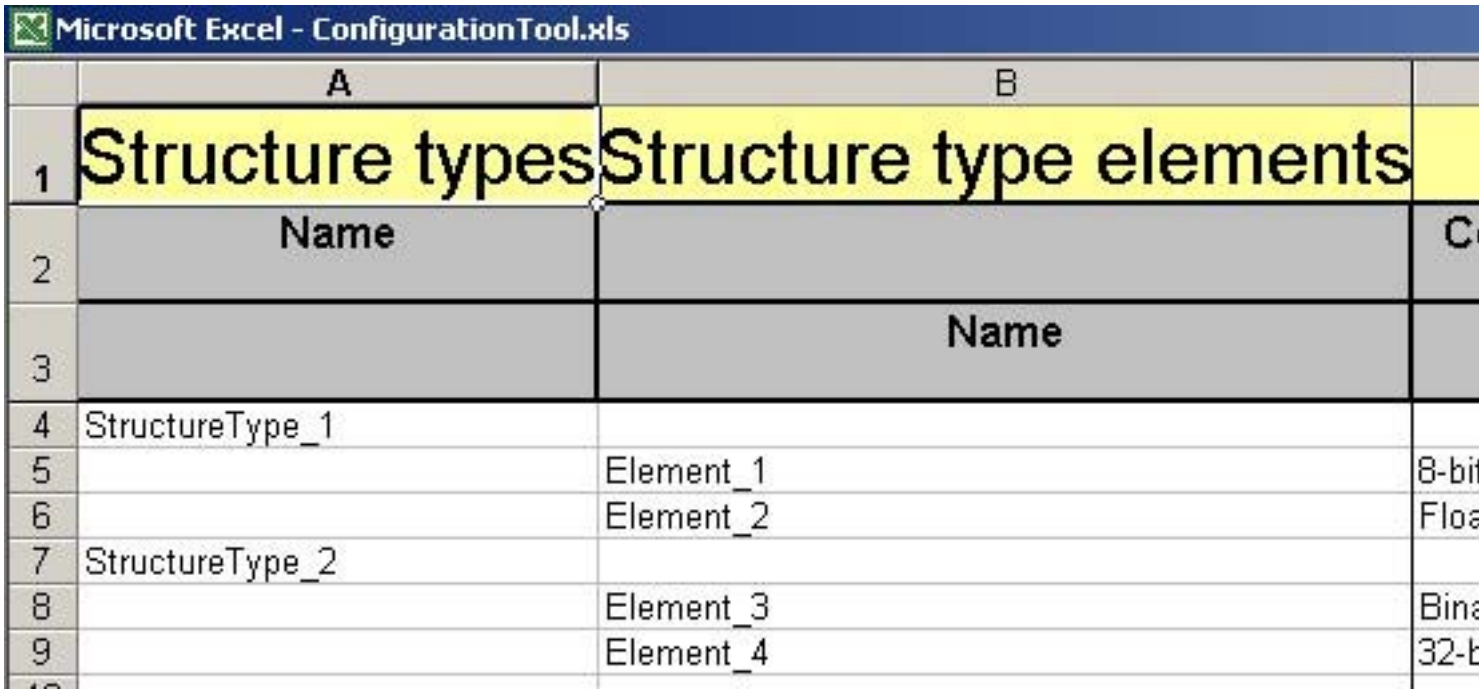
参见

Simatic S7 Protocol Suite 的地址串 (页 243)

操作“结构类型”表单

简介

结构类型表格用于组态两种不同类型的 WinCC 对象，即结构类型本身以及与其相关的结构类型元素。结构类型和结构类型元素之间的逻辑分配从表格中各个对象的位置来确定。


Microsoft Excel - ConfigurationTool.xls

	A	B	
1	Structure types	Structure type elements	
2	Name		C
3		Name	
4	StructureType_1		
5		Element_1	8-bi
6		Element_2	Floa
7	StructureType_2		
8		Element_3	Bina
9		Element_4	32-b

步骤

结构类型

为了创建新的结构类型，必须为其分配一个名称。 用这种方法，类型将成为有效对象并可以写入 WinCC。其它的结构类型参数是可选的。

修改结构类型

当修改已经用于插入一个或多个结构变量的结构类型时，需要在“结构变量”电子表格中再次输入地址。

结构类型元素

结构类型元素只能在结构类型已存在的前提下组态。 为了创建新的结构类型元素，必须为其分配一个名称。然而，只有名称不能说明该结构类型元素为有效的对象。 只有在为结构类型元素分配数据类型之后它才成为有效对象。

只要未指定任何结构类型元素名称，就不能编辑其它结构类型元素参数。指定结构类型元素名称后，只能编辑数据类型。 只有在为结构类型元素指定数据类型后它才成为有效对象。

如果已经激活了用缺省值填写，那么将为其它结构类型元素参数分配缺省值表格中为变量定义的数值。

编写

不能单个地将结构类型元素写入 WinCC。 必须始终将整个结构类型写入 WinCC。 它与选择了结构类型及其所有结构类型元素还是只选择结构类型无关。通常是整个结构类型及其所有元素写入 WinCC。

删除

如果删除结构类型，所有相关的结构类型元素、结构变量和结构实例元素也将一起被删除。如果删除结构类型元素，所有相关的结构实例元素也被删除。从 WinCC 中删除结构实例元素在下次写入结构类型时执行。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

下表列出在“结构类型/结构类型元素”表单中必须为结构类型定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
名称	结构类型的名称。
 通讯驱动程序	用于格式化的通道单元的通讯驱动程序。该参数是可选的。
 通道单元	应使用其格式的通道单元。这些参数只能在已选择了通讯驱动程序后才能设置。

下表列出在“结构类型/结构类型元素”表单中必须为结构类型元素定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
名称	结构类型元素的名称。该名称在结构类型中必须是唯一的。
 数据类型	变量的数据类型。
长度	过程中变量的长度以字节为单位。 对于大多数数据类型，该长度取决于格式改编且不能修改。
偏移量	结构实例元素相对于在结构变量中设置的地址（起始地址）的地址偏移量。
偏移位	设置结构变量起始地址的结构类型元素偏移位。
 格式改编	变量的格式改编。这只能为外部结构类型元素定义。不是所有的数据类型都分配格式改编功能。
 外部变量	指定结构类型元素是外部的还是内部的。
 线性标定是/否	此处指定是否使用线性标定。只有外部变量才能选择线性标定。不是所有的数据类型都支持线性标定。

列	简述
线性标定过程从	过程的定标范围（源定标）。
线性标定过程至	过程的定标范围（源定标）。
线性标定变量从	变量的定标范围（表达式定标）。
线性标定变量至	变量的定标范围（表达式定标）。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

报警记录

报警记录

简介

报警记录表单可用于组态消息块、消息类别、组消息、单个消息以及限制值监视。 此外，“报警记录缺省值”表单可以用于定义单个消息和限制值监视的缺省值。以下章节说明在报警记录中组态数据的步骤。

参见

操作“限制值监视”表单 (页 178)

操作“单个消息”表单 (页 175)

操作“组消息”表单 (页 173)

操作“消息类别/消息类型”表单 (页 169)

操作“消息块”表单 (页 167)

操作“报警记录默认值”表单 (页 161)

操作“报警记录默认值”表单

简介

组态工具提供选项将缺省值预分配给新创建对象的参数。这些缺省值可以在“报警记录缺省值”表单上定义。

步骤

报警记录可用于为单个消息、“生成单个消息”对话框、限制值监视和“生成限制值”对话框设置缺省值。

Microsoft Excel - Mappe2

	A	B
1	Default values	Alarm loggin
2	Single messages	
3	Number	Class
4	Number	Error
5		
6		
7	Generate messages	
8	Bit 0	Bit 1
9	Yes	No
10		
11		
12	Limit value monitoring	Limit value
13	Tag	
14		Limit value
15	Tag	
16		Limit value
17		
18	Generate limit values	
19	Limit value	Limit value indirect from
20	0	

取消激活缺省值

缺省值的使用可以在“项目属性”表单中取消激活。 如果取消激活缺省值，则检查新创建对象所有参数的有效性，并在必要时进行更正（例如空行）。然而，这样在创建对象时会降低性能。因此，缺省情况下缺省值的使用被激活。

表格结构

单个消息

下列表格列出必须为“报警记录缺省值”表单中单个消息定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
数字	该参数写保护。
 类别	单个消息的消息类别。
 类型	单个消息的消息类型
 组	单个消息的组消息。该参数是可选的。
优先级	单个消息的优先级。
 消息变量	单个消息的消息变量。只有某些数据类型可用作消息变量。该参数是可选的。
消息位	单个消息的消息位。如果设置消息变量，必须指定消息位。
 状态变量	单个消息的状态变量。只有某些数据类型可用作状态变量。该参数是可选的。
状态位	单个消息的状态位。如果设置状态变量，必须指定状态位。
 确认变量	单个消息的确认变量。只有某些数据类型可用作确认变量。该参数是可选的。

列	简述
确认位	单个消息的确认位。如果设置确认变量，必须指定确认位。
PLC 编号	单个消息的 PLC 编号
CPU 编号	单个消息的 CPU 编号
消息文本 1 至消息文本 10	单个消息的消息文本。仅当激活相应的消息块时，才能组态消息文本。
过程值变量 1 至过程值变量 10	单个消息的过程值变量。仅当激活相应的过程值块时，才能组态过程值变量。
信息文本	单个消息的信息文本。
 要求单独确认	定义是否必须单独确认单个消息。
 控制中央信号设备	定义单个消息是否要触发发送信号设备。
 将被归档	定义是否要归档单个消息。
 在负沿上创建	定义是否在负沿上触发单个消息。
 触发动作	定义是否要触发标准的 GMsgFunction。
报警回路功能	由单个消息触发功能的名称。
报警回路参数	报警回路功能的传送参数。
 FormatDll	修改进入的原始数据报文以适合 WinCC 报警记录系统的格式 DLL。

生成消息

下表列出必须在“生成消息”对话框的“报警记录缺省值”表单中定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
 0 位至 31 位	指出是否应该为该位生成消息。
原始数据量	指出应该为一个原始数据变量生成多少条消息。

限制值监视/限制值

下表列出必须为“报警记录缺省值”表单中的限制值监视定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
变量	该参数写保护。
用于所有限制值的一个消息	用于所有限制值的单个消息。
延迟时间	定义触发单个消息前，监视值分别超过上限值或低于下限值的时间段。
 延迟单位	延迟时间的单位。

下表列出必须为“报警记录缺省值”表单中的限制值定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
限制值	该参数写保护。
限制值间接来自变量	该参数写保护。
 限制	定义限制值是上限值还是下限值。

列	简述
消息号	要触发的单个消息。 如果为限制值监视设置参数“用于所有限制值的一个消息”，此处不能输入消息号。
 抑制质量代码不是“GOOD”的消息	定义当变量的质量代码不是“GOOD”时，是否抑制该变量的消息。
 滞后	定义滞后是绝对的还是相对的（用百分比表示）。
滞后值	滞后的值。0 表示不使用滞后。
 滞后生效自	定义滞后的生效时间。

生成限制值

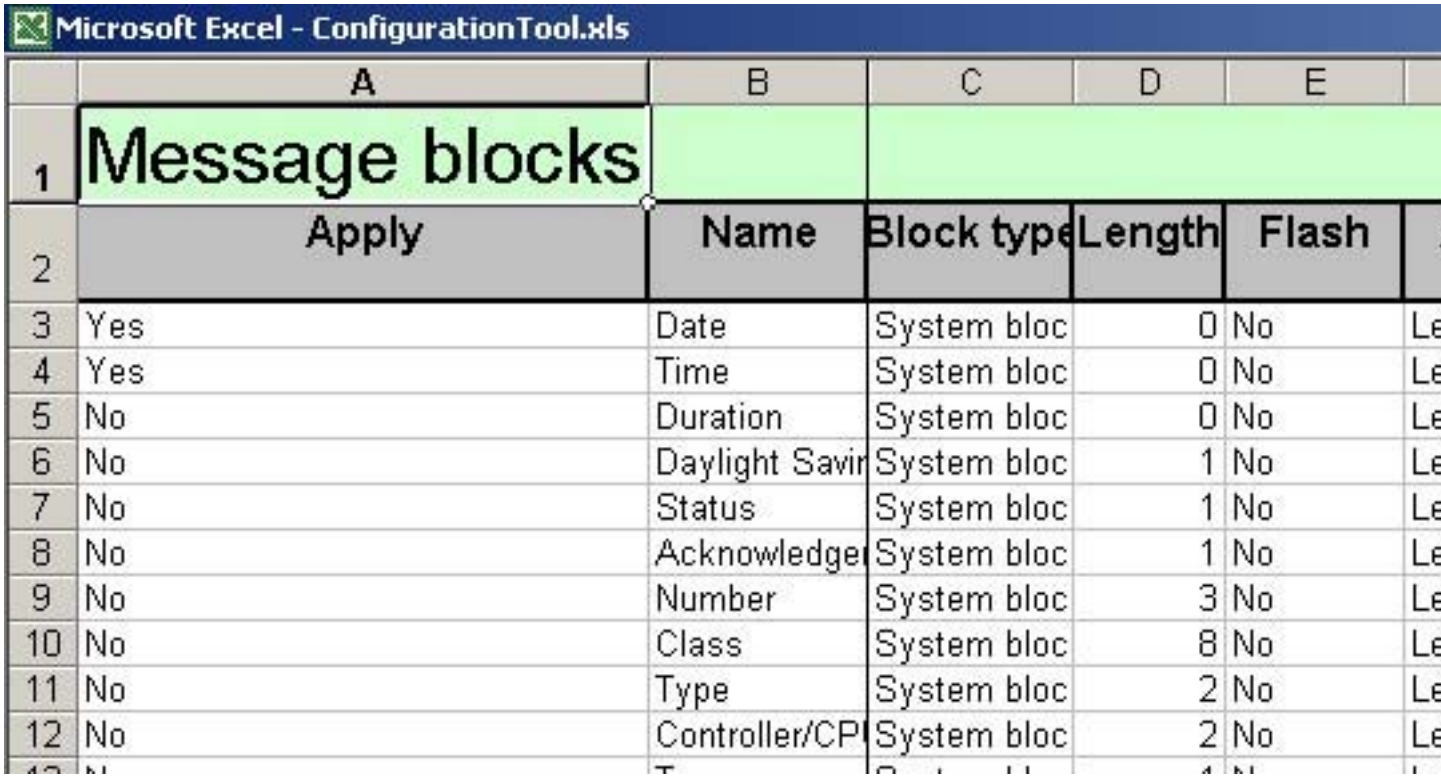
下表列出必须在“生成限制值”对话框的“报警记录缺省值”表单中定义的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
限制值	变量的限制值。
 限制值间接来自变量	限制值应从其中间接采用的变量。
限制	定义限制值是上限值还是下限值。

操作“消息块”表单

简介

“消息块”表单可用于激活或禁用系统块、用户文本块和过程值块。



	A	B	C	D	E
1	Message blocks				
2	Apply	Name	Block type	Length	Flash
3	Yes	Date	System bloc	0	No
4	Yes	Time	System bloc	0	No
5	No	Duration	System bloc	0	No
6	No	Daylight Savir	System bloc	1	No
7	No	Status	System bloc	1	No
8	No	Acknowledge	System bloc	1	No
9	No	Number	System bloc	3	No
10	No	Class	System bloc	8	No
11	No	Type	System bloc	2	No
12	No	Controller/CP	System bloc	2	No

步骤

为了激活消息块，在“使用”列中将参数设置为“是”。并非所有参数设置都可以在所有消息块中修改。如果禁用消息块，则为该消息块定义的单个消息表格上的所有参数设置都将被删除。

如果用户文本块或过程值块被激活，则新列将添加到单个消息表格上。消息块的名称作为标题输入。

不能添加其它消息块。

写入

由于所有可用的消息块都已存在，可以立即向 WinCC 写入消息块。

删除

消息块不能在组态工具中删除，只能禁用。

表格结构

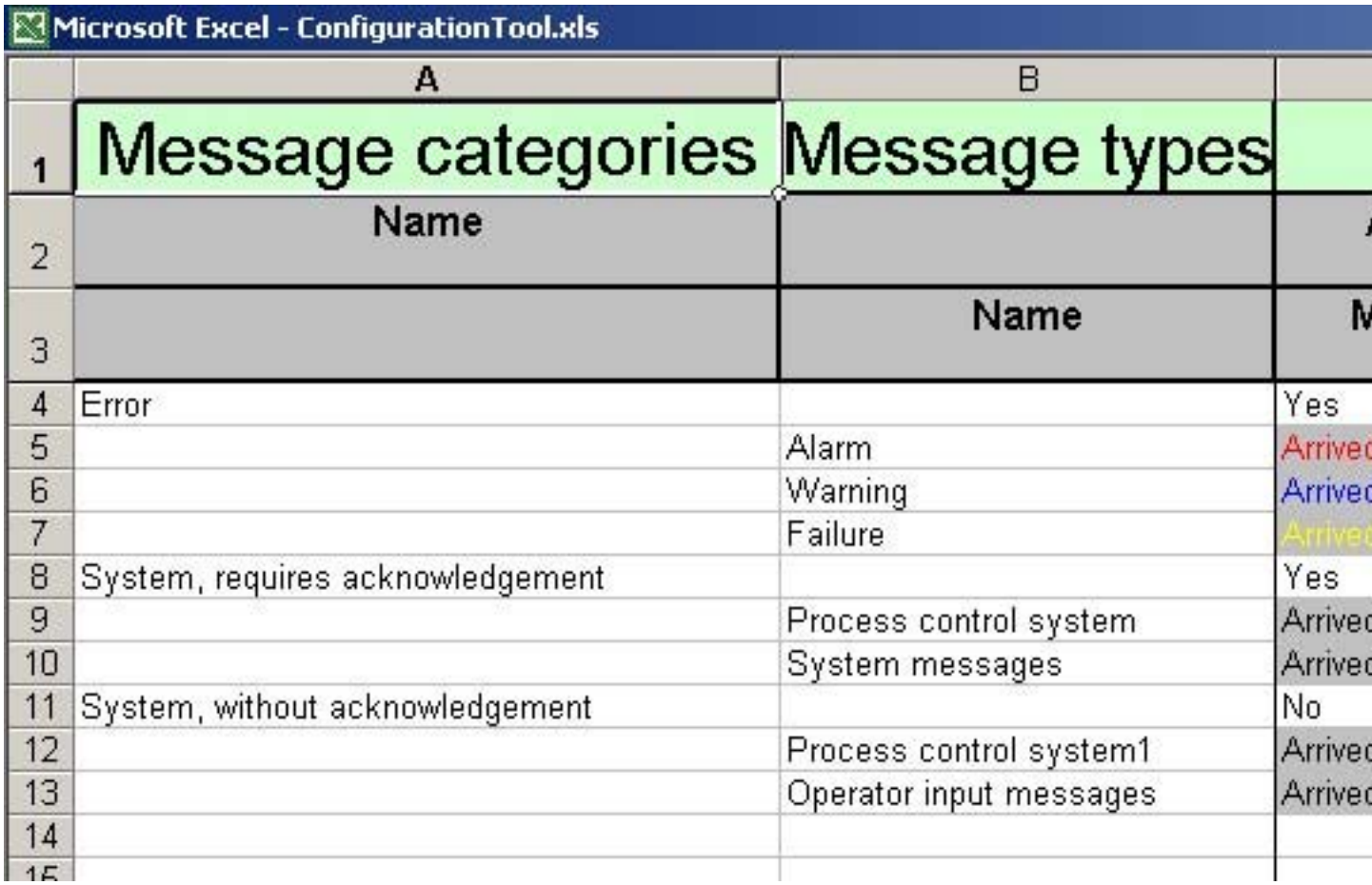
下表列出必须为消息块设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
 使用	定义是否使用消息块。
名称	消息块的名称。
块类型	消息块类型。该参数只用于显示信息并且是写保护的。
长度	消息块长度。不能为所有消息块设置该参数。
 闪烁	定义显示在消息窗口中的消息是否闪烁。
 对齐	定义显示在消息窗口中文本的对齐方式。
 格式 1	消息窗口中的表示方式。格式 1 由消息块决定并且只能针对某些消息块选用。
 格式 2	消息窗口中的表示方式。格式 2 由消息块决定并且只能针对某些消息块选用。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配错误文本“OK”。

操作“消息类别/消息类型”表单

简介

“消息类别/消息类型”表单用于组态两种不同类型的 WinCC 对象，即消息类别和从属于消息类别的消息类型。消息类别和消息类型的逻辑分配由表格中相应对象的位置确定。



	A	B	
1	Message categories	Message types	
2	Name		
3		Name	M
4	Error		Yes
5		Alarm	Arrived
6		Warning	Arrived
7		Failure	Arrived
8	System, requires acknowledgement		Yes
9		Process control system	Arrived
10		System messages	Arrived
11	System, without acknowledgement		No
12		Process control system1	Arrived
13		Operator input messages	Arrived
14			
15			

步骤

消息类别

为创建一个新消息类别，必须为其分配一个名称。为保留的参数分配缺省值。 输入名称后，消息类别定义为有效对象并可写入 WinCC。消息类别的名称必须唯一，并且不能与消息类型或组消息的名称相同。

最多可以创建 16 个用户定义的消息类别，每个类别包括 16 个消息类型。此外，已经提供了两个系统消息类别，各包括两个系统消息类型，但只能对其进行有限组态。

消息类型

要创建新的消息类型，必须已经有可用的消息类别。必须给新的消息类型一个名称。为保留的参数分配缺省值。消息类型的着色必须使用对单元格的标准 Excel 功能定义。

说明

在 WinCC 中为每个消息类别和消息类型自动生成的组消息参数的不同之处在于表单中已经提供这些参数。

编写

消息类型不能单个写入 WinCC。要在这里写入消息类型，必须将上级消息类别写入 WinCC。下级消息类型自动随消息类别一起写入。

删除

如果删除消息类别，则所有相关消息类型随着一起删除。系统消息类别不能删除。消息类型可以从表格中单个删除。要从 WinCC 中删除消息类型，相关消息类别必须写入 WinCC。

属于消息类型或消息类别的单个消息也会在删除操作中一起被删除。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

下表列出在“消息类别/消息类型”表单中必须为消息类别设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
名称	消息类别的名称。名称必须唯一。
 确认已到达	定义是否要确认进入的消息。

列	简述
 确认已离开	定义是否要确认离开的消息。
 闪烁	定义排队的消息是否闪烁。
 仅第一个值闪烁	定义是否仅该消息类别的第一个排队消息闪烁。
 不带“已离开”状态的消息	定义是否只归档状态为“已到达”的消息。
 确认中央信号设备	定义是否通过单独的确认按钮或通过单独确认对发送信号设备进行确认。
 中央发送信号变量	通过其触发发送信号设备的变量。
状态文本已到达	用于“已到达”状态的文本。该文本最长可以为 63 个字符。
状态文本已离开	用于“已离开”状态的文本。该文本最长可以为 63 个字符。
状态文本已确认	用于“已确认”状态的文本。该文本最长可以为 63 个字符。
状态文本已到达和已离开	用于“已到达和已离开”状态的文本。该文本最长可以为 63 个字符。
 组消息状态变量	消息类别的组消息状态变量。该参数是可选的。
组消息状态位	消息类别的组消息状态位。使用两个变量位。如果指定状态变量，也必须指定状态位。
 组消息锁定变量	消息类别的组消息锁定变量。该参数是可选的。
组消息块位	消息类别的组消息块位。 如果指定锁定变量，也必须指定锁定位。

列	简述
 组消息确认变量	消息类别的组消息确认变量。该参数是可选的。
组消息确认位	消息类别的组消息确认位。如果指定确认变量，也必须指定确认位。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

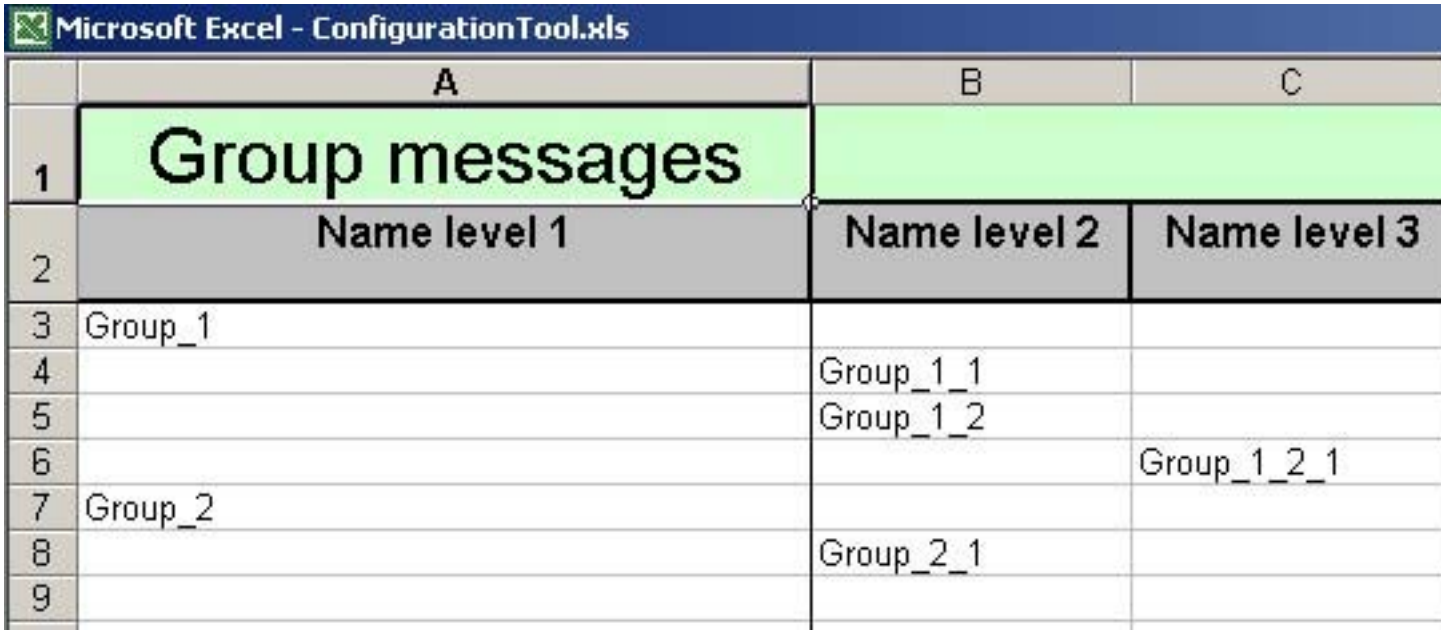
下表列出在“消息类别/消息类型”表单中必须为消息类型设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
名称	消息类型的名称。名称必须唯一。
消息已到达着色	状态为“消息已到达”的消息的文本颜色和背景色。
消息已离开着色	状态为“消息已离开”的消息的文本颜色和背景色。
消息已确认着色	状态为“消息已确认”的消息的文本颜色和背景色。
 组消息状态变量	消息类型的组消息状态变量。该参数是可选的。
组消息状态位	消息类型的组消息状态位。使用两个变量位。如果指定状态变量，也必须指定状态位。
 组消息块变量	消息类型的组消息块变量。该参数是可选的。
组消息块位	消息类型的组消息块位。 如果指定锁定变量，也必须指定锁定位。
 组消息确认变量	消息类型的组消息确认变量。该参数是可选的。
组消息确认位	消息类型的组消息确认位。如果指定确认变量，也必须指定确认位。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

操作“组消息”表单

简介

在“组消息”表单上，可以组态所有六个可能等级的组消息。



	A	B	C
1	Group messages		
2	Name level 1	Name level 2	Name level 3
3	Group_1		
4		Group_1_1	
5		Group_1_2	
6			Group_1_2_1
7	Group_2		
8		Group_2_1	
9			

步骤

为创建一个新组消息，必须为其分配一个名称。输入名称后，组消息就定义为有效对象并且可以将它写入 WinCC。组消息的名称必须是唯一的。

组消息是分层结构。也就是说，与第一级相关的组消息必须在组态第二级组消息之前先组态。为了组态第三级组消息，第二级组消息必须已经预先组态好，以此类推。

写入

只有来自第一级的组消息可以写入 WinCC。存在的所有子组自动与其一起写入 WinCC。

删除

可以删除所有级别的组消息。所有子组自动随其一起删除。

为其指定删除的组消息的单个消息将不被删除。只会删除该组消息的引用。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

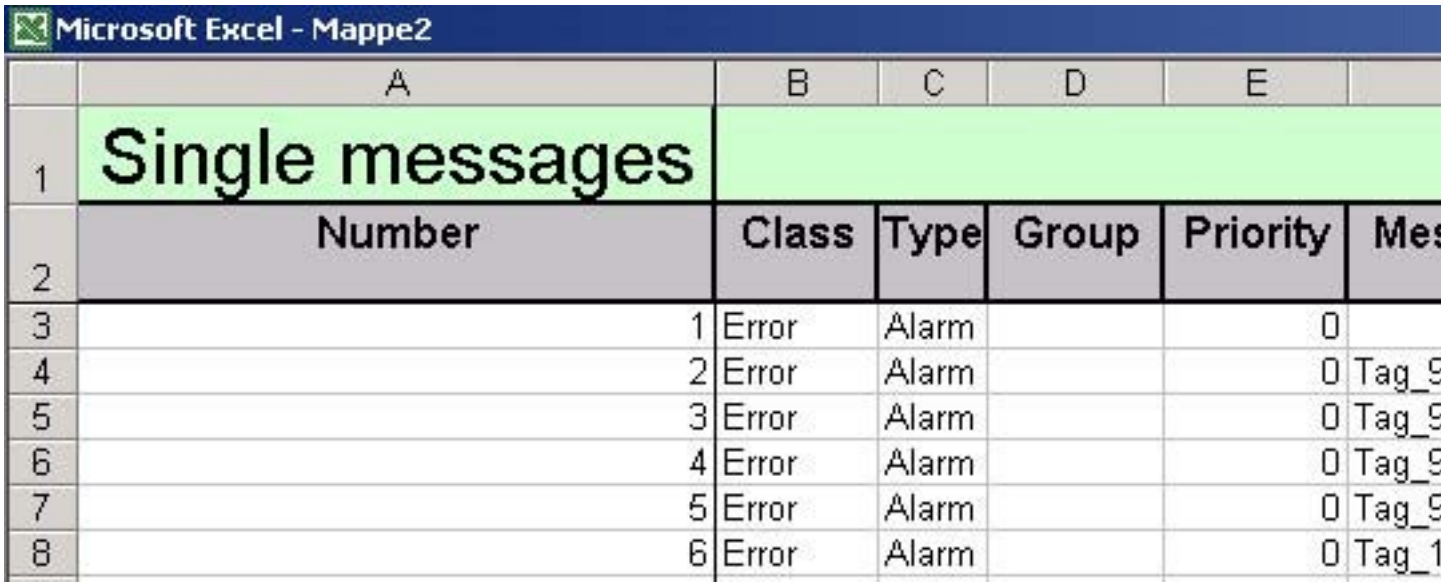
下表列出必须为组消息设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
第 1 级至第 6 级的名称	组消息的名称。
 状态变量	组消息的状态变量。该参数是可选的。
状态位	组消息的状态位。使用两个变量位。如果指定状态变量，也必须指定状态位。
 锁定变量	组消息的锁定变量。该参数是可选的。
锁定位	组消息的锁定位。如果指定锁定变量，也必须指定锁定位。
 确认变量	组消息的确认变量。该参数是可选的。
确认位	组消息的确认位。如果指定确认变量，也必须指定确认位。
隐藏变量	自定义组消息的隐藏变量用于自动隐藏属于该组消息的单个消息。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配错误文本“OK”。

操作“单个消息”表单

简介

所需的单个消息可以在“单个消息”表单上组态。根据已经激活或禁用的消息块数目，需要设置的参数数目也有所不同。单个消息也可以从变量表生成。



	A	B	C	D	E	
1	Single messages					
2	Number	Class	Type	Group	Priority	Mes
3	1	Error	Alarm		0	
4	2	Error	Alarm		0	Tag_9
5	3	Error	Alarm		0	Tag_9
6	4	Error	Alarm		0	Tag_9
7	5	Error	Alarm		0	Tag_9
8	6	Error	Alarm		0	Tag_1

步骤

为了创建单个消息，必须为其分配一个名称。如果已经激活了用默认值填写，那么将为单个消息的其它参数分配默认值表格中定义的数值。否则，为剩余的参数分配预先定义的数值。

输入数值后，单个消息就定义为有效对象并且可以将它写入 WinCC。

单个消息也可以从变量表生成。此操作的步骤在下面的章节中描述。

写入

一旦单个消息被定义为有效对象，就可以将它写入 WinCC。也可以将组态的消息文本写入 WinCC。注意一定要将单个消息表中使用的变量预先写入 WinCC。

删除

在删除单个消息时，任何不再需要的消息文本也会被删除。这点在项目属性表输入的设置中进行定义。限制值也根据项目属性表中的设置进行相应的处理（删除或设置缺省消息号）。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

下表列出必须为单个消息设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
编号	单个消息的编号
 类别	单个消息的消息类别。
 类型	单个消息的消息类型
 组	单个消息的组消息。该参数是可选的。
优先级	单个消息的优先级。
 消息变量	单个消息的消息变量。只有某些数据类型可用作消息变量。该参数是可选的。
消息位	单个消息的消息位。如果设置消息变量，必须指定消息位。
 状态变量	单个消息的状态变量。只有某些数据类型可用作状态变量。该参数是可选的。
状态位	单个消息的状态位。如果设置状态变量，必须指定状态位。
 确认变量	单个消息的确认变量。只有某些数据类型可用作确认变量。该参数是可选的。

列	简述
确认位	单个消息的确认位。如果设置确认变量，必须指定确认位。
PLC 号	单个消息的 PLC 号
CPU 编号	单个消息的 CPU 编号
消息文本 1 至消息文本 10	单个消息的消息文本。仅当激活相应的消息块时，才能组态消息文本。
过程值变量 1 至过程值变量 10	单个消息的过程值变量。仅当激活相应的过程值块时，才能组态过程值变量。
信息文本	单个消息的信息文本。
 仅限单个确认	定义是否必须单独确认单个消息。
 控制中央信令设备	定义单个消息是否要触发信号发送设备。
 将被归档	定义是否要归档单个消息。
 在负沿上创建	定义是否在负沿上触发单个消息。
 触发动作	定义是否要触发标准的 GMsgFunction。
报警回路函数	由单个消息触发的函数名称。
报警回路参数	传送参数到报警回路函数。
 格式 Dll	修改进入的原始数据报文以适合 WinCC 报警记录系统的格式 DLL。
隐藏掩码	指定通过隐藏掩码隐藏单个掩码的系统状态。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配错误文本“OK”。

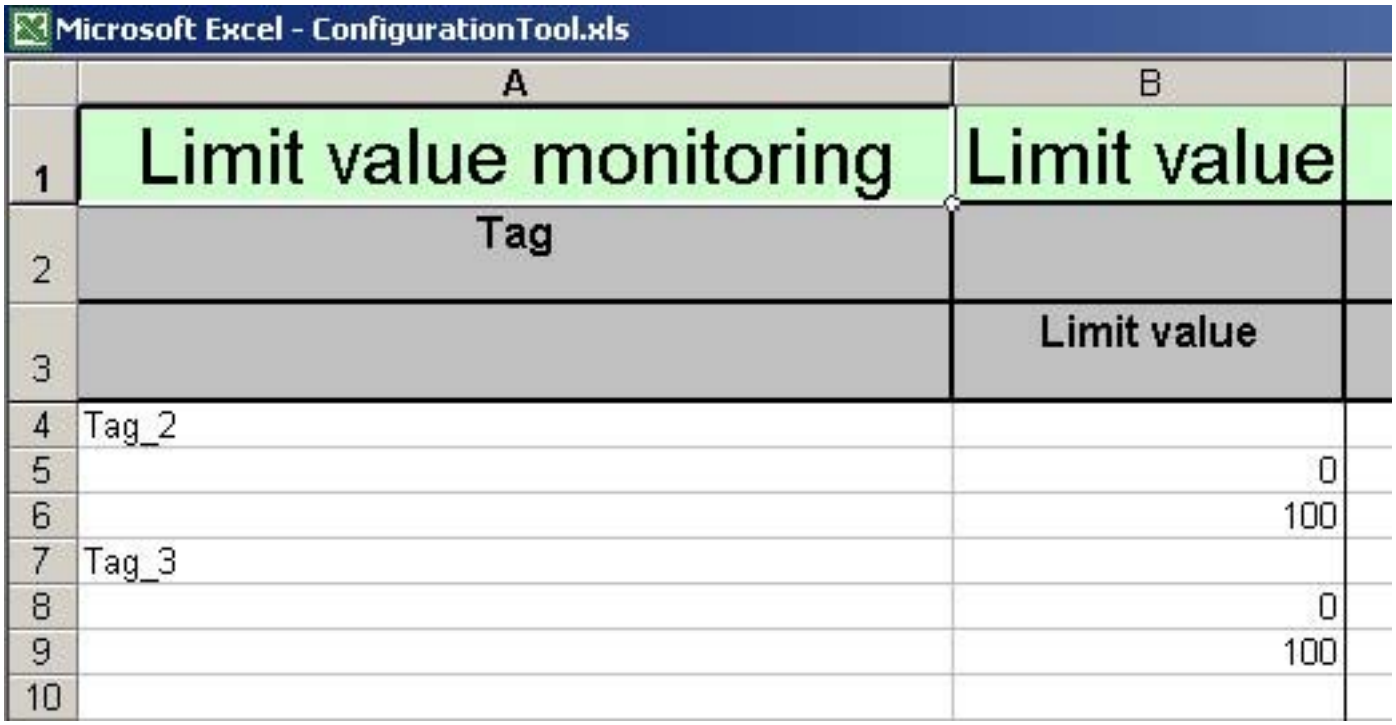
参见

从变量表创建单个消息 (页 223)

操作“限制值监视”表单

简介

“限制值监视”表单用于组态两个不同类型的 WinCC 对象，即限制值监视以及从属限制值。限制值监视和限制值之间的逻辑分配从表格中各个对象的位置来确定。



	A	B
1	Limit value monitoring	Limit value
2	Tag	
3		Limit value
4	Tag_2	
5		0
6		100
7	Tag_3	
8		0
9		100
10		

步骤

限制值监视

为了创建限制值监视，必须设置变量。 通过设置变量，限制值监视已定义为有效对象并且可以将它写入 WinCC。 为限制值监视参数分配缺省值。这些缺省值可以在“报警记录缺省值”表单上定义。

限制值

限制值只能添加到现有的限制值监视设置中。为了创建限制值，必须输入限制值或输入限制值将被其间接传送的变量。 通过这种方法，限制值定义为有效对象并且可以将它写入 WinCC。 为限制值参数分配缺省值。这些缺省值可以在“报警记录缺省值”表单上定义。

编写

限制值连同上级限制值监视一起自动写入。限制值不能单个地写入 WinCC。

说明

在 WinCC 中，需要通过选择“工具 - 加载项”激活报警记录编辑器中的限制值监视功能。

删除

删除限制值监视设置时，也删除了所有从属限制值。也可单个地删除限制值。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

使用变量表弹出式菜单，可以生成带有限制值的限制值监视。此操作的步骤在下面的章节中描述。

用于现有限制值监视的限制值可以通过使用“限制值监视/限制值”表格表单生成。为此，选择要向其添加限制值的限制值监视。在表单弹出式菜单中选择条目“WinCC - 创建限制值”。

显示“限制值监视”对话框。第 3 项以后的步骤在下面的章节中描述。

表格结构

下表列出必须在“限制值监视/限制值”表单中为限制值监视设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
 变量	要监视的变量。
用于所有限制值的一个消息	用于所有限制值的单个消息。如果输入了消息号，则不能为单个限制值输入其它单个消息。

列	简述
延迟时间	定义触发单个消息前，监视值分别超过上限值或低于下限值的时间段。数值 0 表示没有延迟。总计的延迟时间可能最多为 24 小时。
 延迟单位	延迟时间的单位。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

下表列出必须在“限制值监视/限制值”表单中为限制值设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
限制值	变量的限制值。 只有在未从变量间接传送限制值时才能对其进行设置。
 限制值间接来自变量	其限制值受监视的变量的限制值从其它变量间接传送。
 限制	定义限制值是上限值还是下限值。
消息号	要触发的单个消息。
 抑制质量代码不是“GOOD”的消息	定义当变量的质量代码不是“GOOD”时，是否抑制该变量的消息。
 滞后	定义滞后是绝对的还是相对的（用百分比表示）。
滞后值	滞后的值。0 表示不使用滞后。
 滞后生效自	定义滞后的生效时间。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

参见

从变量表创建限制值监视 (页 227)

变量记录

变量记录

简介

“变量记录”表单可用于组态时间、过程值归档和压缩归档。 此外，“变量记录缺省值”表单可以用于定义过程值归档和压缩归档的缺省值。以下章节说明在变量记录中组态数据的步骤。

参见

操作“压缩归档”表单 (页 194)

操作“过程值归档”表单 (页 190)

操作“时间”表单 (页 188)

操作“变量记录默认值”表单 (页 182)

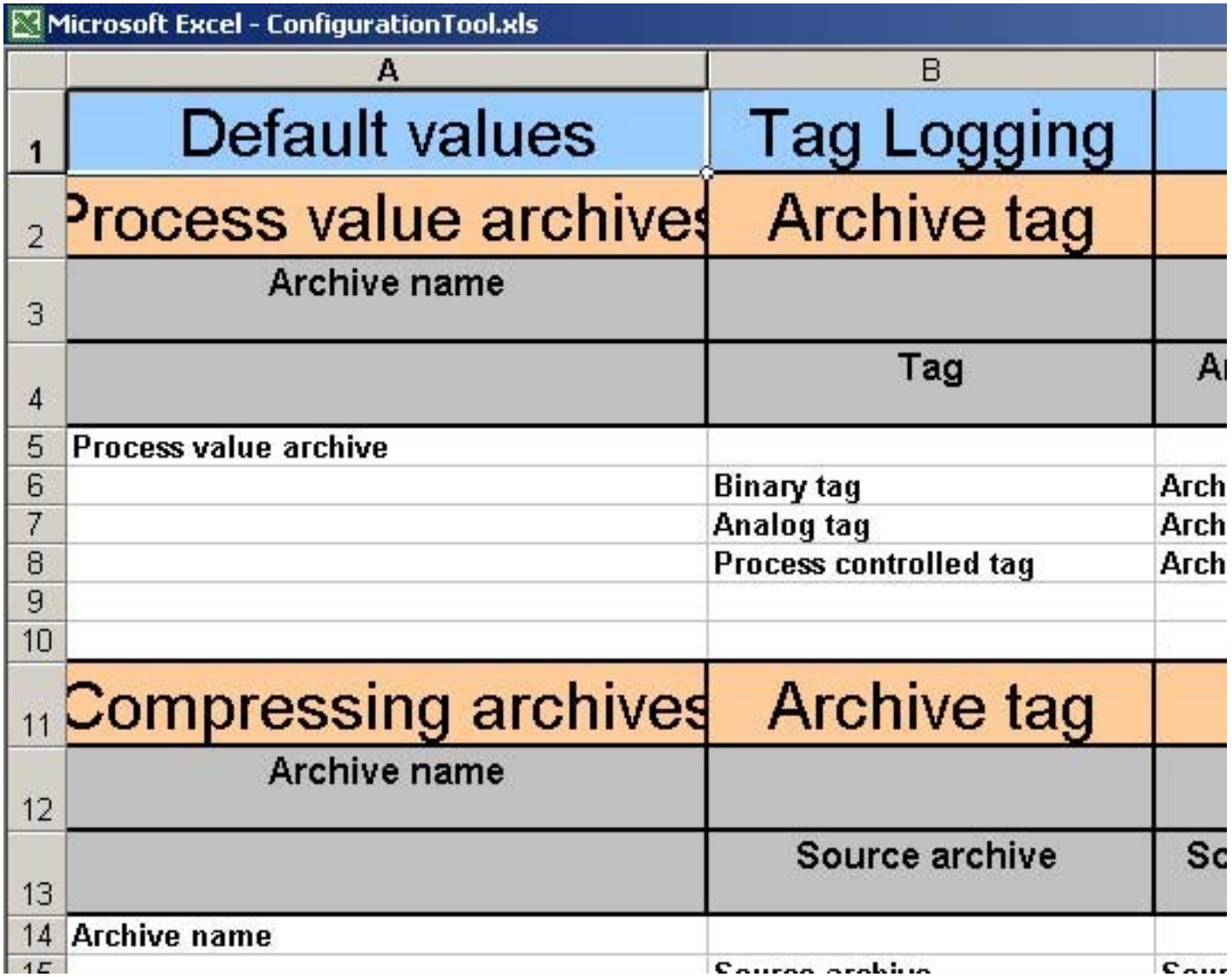
操作“变量记录默认值”表单

简介

组态工具提供选项将默认值预分配给新创建对象的参数。这些默认值可以在“变量记录默认值”表单上定义。

步骤

变量记录功能可用于定义过程值归档、过程值归档变量、压缩归档和压缩归档变量的默认值。



禁用默认值

默认值的使用可以在“项目属性”表单中禁用。如果禁用默认值，则检查新创建对象所有参数的有效性，并在必要时进行更正（例如空行）。然而，这样在创建对象时会降低性能。因此，默认情况下会激活默认值的使用。

表格结构

下表列出必须在表单中设置的参数。带有下拉式列表框的参数用  图标标识。

过程值归档

列	简述
归档名称	该参数为写保护。
 归档类型	该参数为写保护。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
启动/启用动作	启动或启用归档时调用的动作。
大小	数据记录中归档的大小。
 存储位置	只能为循环归档设置该参数。定义归档存储在主存储器上还是硬盘上。

过程值归档变量

列	简述
变量	该参数为写保护。
归档变量名称	该参数为写保护。
归档变量类型	该参数为写保护。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
 提供变量	定义变量数据是自动应用还是手动应用。

列	简述
 采集类型	数据采集的类型。
 采集周期	用于数据采集的周期。
 归档周期	归档和显示数据的周期使用的时基。归档周期得自时基和归档周期系数的乘积。它必须是采集周期的整数倍。
归档周期系数	归档周期系数。归档周期得自时基和归档周期系数的乘积。它必须是采集周期的整数倍。
同时写入变量	最后的归档值将被传送到的变量。
 编辑	所采集数据的处理。在周期性选择和周期性连续归档情况下，所有选择项都可使用。在非周期归档和每次更改后归档的情况下，只有当前值可用。
处理的动作	用其处理数据的动作。该参数只有在将处理设置成动作后才能设置。
单位	所输入时间值的单位。
长期关联	定义是否将归档变量写入到中央归档服务器 (WinCC CAS) 的归档之中。
 更改时归档	定义变量是否只在修改后进行归档。
 滞后	定义滞后是绝对的还是相对的（用百分比表示）。
滞后值	滞后的值。0 表示不使用滞后。
 归档条件	在二进制过程值归档变量的情况下触发。该参数对于模拟和过程控制变量都是写保护的。
 出错时保存	定义出错后归档最后值还是替换值。

列	简述
先行值数目	归档前要考虑在内的值的数量。该参数只有在周期性选择性归档情况下才能设置。
后续值数目	归档后要考虑在内的值的数量。该参数只有在周期性选择性归档情况下才能设置。
显示限制，下限	变量值范围的下限。0 表示无限制。
显示限制，上限	变量值范围的上限。0 表示无限制。
启动事件	触发归档的功能名称。
开始变量	二进制变量。
停止事件	停止归档的功能名称。
停止变量	二进制变量。
长期关联	可在此处定义是否要将中央归档服务器（WinCC CAS）上的归档变量写入归档。
段更改后归档	可以在此确定段更改时是否归档非周期性归档的变量。
 标准化 DLL	与变量关联的修改程序。它取决于 AS 制造商。

压缩归档

列	简述
归档名称	该参数为写保护。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
启动/启用动作	启动或启用归档时调用的动作。
 处理方法	处理源归档或压缩归档变量的方法。
 压缩周期	归档建立时的压缩时间周期。压缩周期必须至少为一分钟。

压缩归档变量

列	简述
源归档	该参数为写保护。
源变量	该参数为写保护。
归档变量名称	该参数为写保护。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
 编辑	在归档周期内已编译数值的处理方法。
单位	要归档的变量的单位。
长期关联	可在此处定义是否将归档变量写入到中央归档服务器（WinCC CAS）的归档之中。

操作“时间”表单

简介

归档所需时间周期可以在“时间”表单上组态。创建新项目文件夹或新 WinCC 项目时，缺省情况下提供 5 种时间。这个设置可以修改或删除。

Microsoft Excel - ConfigurationTool.xls

	A	B	C	D	
1	Times				
2	Name	Basis	Factor	Trigger at system start	Trigger
3	500 ms	500 ms	1	No	No
4	1 second	1 second	1	No	No
5	1 minute	1 minute	1	No	No
6	1 hour	1 hour	1	No	No
7	1 day	1 day	1	No	No
8					

步骤

要创建新的时间，必须为其指定一个名称。给保留的参数分配预定义的数值。输入名称后，时间就定义为有效对象并且可以将它写入 WinCC。

编写

一旦时间变成有效对象，就可以将它写入 WinCC。如果将时间写入 WinCC，所有修改的归档系统对象也将写入 WinCC。

删除

只能删除任何归档都不使用的那些时间。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

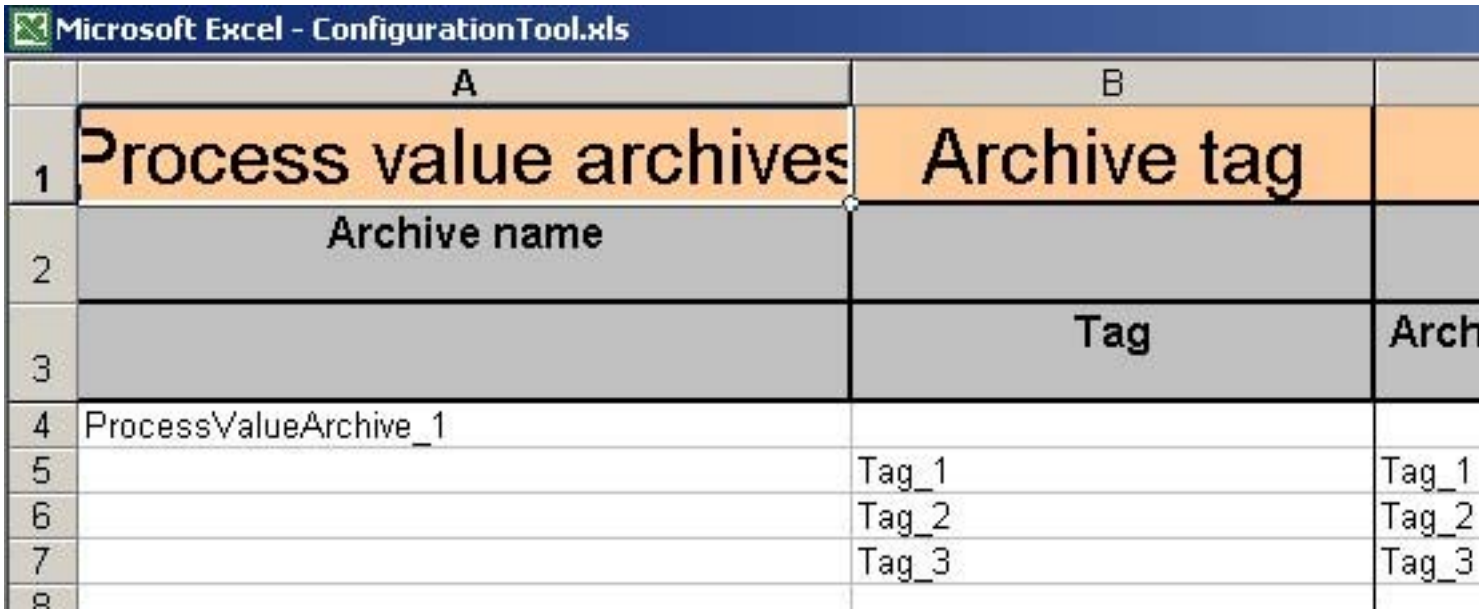
下表列出必须为时间设置的所有参数。带有下拉式列表框的参数用  图标标识。

列	简述
名称	时间的名称。
 时基	基准时间。时基与系数相乘得出周期时间。
系数	与时基相乘的系数。时基与系数相乘得出周期时间。
 系统启动时触发	定义周期是否要在系统启动时触发。
 在起始点处触发	定义周期是否要在起始点触发。
起始点月份	“起始点月份”只能在“在起始点处触发”选项设置为“是”且周期时间至少为一小时的时候设置。
起始点日	“起始点日”只能在“在起始点处触发”选项设置为“是”且周期时间至少为一小时的时候设置。
起始点小时	“起始点小时”只能在“在起始点处触发”选项设置为“是”且周期时间至少为一分钟的时候设置。
起始点分钟	“起始点分钟”只能在“在起始点处触发”选项设置为“是”时设置。
起始点秒	“起始点秒”只能在“在起始点处触发”选项设置为“是”时设置。
错误文本	写入 WinCC 时出现的错误简述。无错误对象分配有错误文本“OK”。

操作“过程值归档”表单

简介

过程值归档表格用于组态两种不同类型的 WinCC 对象，即过程值归档和与其关联的过程值归档变量。归档和变量之间的逻辑分配从表格中各个对象的位置来确定。



步骤

过程值归档

为了创建新的过程值归档，必须为其分配一个名称。一旦分配了名称，归档就被定义为有效对象，就可以将它写入 WinCC。如果已经激活了用默认值填写，则将为其它归档参数分配默认值表格中定义的数值。

过程值归档变量

为了创建新的过程值归档变量，在“变量”列中必须输入一个现有的变量。要在“变量”列中输入变量，可以使用变量选择对话框，而该对话框可以通过双击各单元格打开。一旦输入了变量，过程值归档变量就被定义为有效对象，就可以将它写入 WinCC。

如果已经激活了用默认值填写，那么将为过程值归档变量的其它归档参数分配默认值表格中定义的数值。

说明

过程值归档和过程值归档变量也可以使用变量表格组态。以下章节将说明在变量表中组态数据的步骤。

写入

如果将过程值归档或过程值归档变量写入 WinCC，则变量记录功能的所有已修改对象会自动写入 WinCC。

删除

如果删除过程值归档，所有子过程值归档变量也随着一起删除。此外，通过过程值归档或过程值归档变量设置的所有压缩归档变量也被删除。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

下表列出必须在表单中设置的参数。带有下拉式列表框的参数用  图标标识。

过程值归档

列	简述
归档名称	过程值归档的名称。名称必须唯一。
 归档类型	该参数为写保护。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
启动/启用动作	启动或启用归档时调用的动作。

列	简述
大小	数据记录中归档的大小。
 存储位置	只能为循环归档设置该参数。定义归档存储在主存储器上还是硬盘上。

过程值归档变量

列	简述
变量	要归档的变量。
归档变量名称	过程值归档变量的名称。名称必须唯一。
归档变量类型	该参数为写保护。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
 提供变量	定义变量数据是自动应用还是手动应用。
 采集类型	数据采集的类型。
 采集周期	用于数据采集的周期。
 归档周期	归档和显示数据的周期使用的时基。归档周期得自时基和归档周期系数的乘积。它必须是采集周期的整数倍。
归档周期系数	归档周期系数。归档周期得自时基和归档周期系数的乘积。它必须是采集周期的整数倍。
同时写入变量	最后的归档值将被传送到的变量。

列	简述
 编辑	所采集数据的处理。在周期性选择和周期性连续归档情况下，所有选择项都可使用。在非周期归档和每次更改后归档的情况下，只有当前值可用。
处理的动作	用其处理数据的动作。该参数只有在将处理设置成动作后才能设置。
单位	所输入时间值的单位。
 更改时归档	定义变量是否只在修改后进行归档。
 滞后	定义滞后是绝对的还是相对的（用百分比表示）。
滞后值	滞后的值。0 表示不使用滞后。
 归档条件	在二进制过程值归档变量的情况下触发。该参数对于模拟和过程控制变量都是写保护的。
 出错时保存	定义出错后归档最后值还是替换值。
先行值数目	归档前要考虑在内的值的数量。该参数只有在周期性选择性归档情况下才能设置。
后续值数目	归档后要考虑在内的值的数量。该参数只有在周期性选择性归档情况下才能设置。
显示限制，下限	变量值范围的下限。0 表示无限制。
显示限制，上限	变量值范围的上限。0 表示无限制。
启动事件	触发归档的功能名称。
开始变量	二进制变量。
停止事件	停止归档的功能名称。
停止变量	二进制变量。
长期关联	可在此处定义是否要将中央归档服务器（WinCC CAS）上的归档变量写入归档。
段更改后归档	可以在此确定段更改时是否归档非周期性归档的变量。

列	简述
错误文本	可以在此输入错误文本。
 标准化 DLL	与变量关联的修改程序。它取决于 AS 制造商。

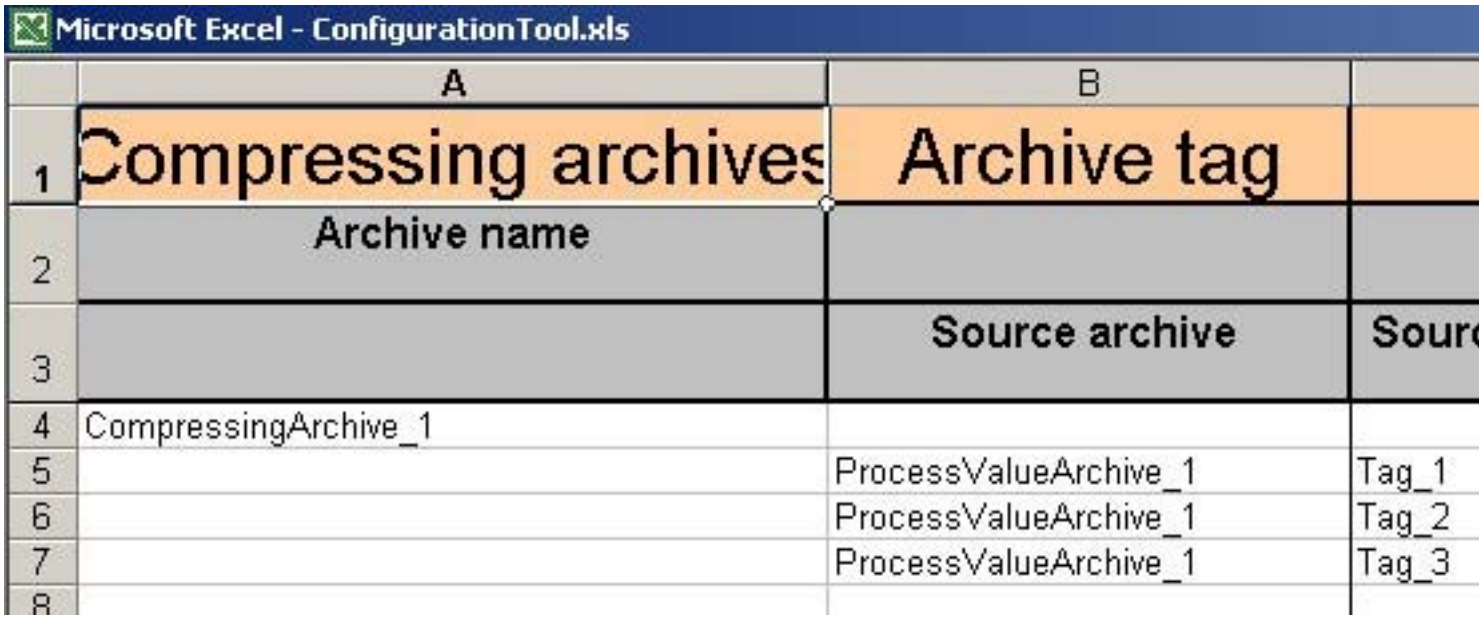
参见

从变量表创建归档变量 (页 220)

操作“压缩归档”表单

简介

压缩归档表格用于组态两种不同类型的 WinCC 对象，即压缩归档和与压缩归档关联的压缩归档变量。归档和变量之间的逻辑分配从表格中各个对象的位置来确定。



步骤

压缩归档

为了创建新的压缩归档，必须分配一个名称。一旦分配了名称，归档就被定义为有效对象，就可以将它写入 WinCC。

如果已经激活了用默认值填写，则将为其它归档参数分配默认值表格中定义的数值。

压缩归档变量

为了创建新的压缩归档变量，必须指定一个源归档。双击“源归档”列打开“归档变量选择”对话框。使用该对话框选择要压缩的过程值归档变量。

如果已经激活了用默认值填写，那么将为压缩归档变量的归档参数分配默认值表格中定义的数值。

写入

如果将压缩归档或压缩归档变量写入 WinCC，则变量记录功能的所有已修改对象也自动写入 WinCC。

删除

如果删除压缩归档，所有从属压缩归档变量也随着一起删除。

说明

数据从项目文件夹和 WinCC 中永久删除。数据的删除不能撤消。

表格结构

下表列出必须在表单中设置的参数。带有下拉式列表框的参数用  图标标识。

压缩归档

列	简述
归档名称	压缩归档的名称。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
启动/启用动作	启动或启用归档时调用的动作。

列	简述
 处理方法	处理源归档或压缩归档变量的方法。
 压缩周期	归档建立时的压缩时间周期。压缩周期必须至少为一分钟。

压缩归档变量

列	简述
源归档	包含源变量的源归档。
源变量	要压缩的过程值归档变量。
归档变量名称	压缩归档变量的名称。
注释	任意注释。
 在系统启动时进行归档	定义是否要在系统启动时开始归档。
 编辑	在归档周期内已编译数值的处理方法。
单位	要归档的变量的单位。
长期关联	可在此处定义是否将归档变量写入到中央归档服务器（WinCC CAS）的归档之中。
错误文本	可以在此输入错误文本。

文本库

文本库

简介

“文本库”可以用于编辑在运行系统中被各种模块使用的文本。

操作“文本”表单

简介

“文本”表单用于管理文本库。 默认情况下，系统提供了德语、英语、法语、意大利语和西班牙语。 您可以选择添加或删除文本和语言。

	A	B	C	D
1	Texts			
2	Text ID	German (Germany)	English (United States)	French (F
3	1	Störung	Error	Incident
4	2	System, quittierpflichtig	System, requires acknowledgem	Système, acc
5	3	System, ohne Quittierung	System, without acknowledgeme	Système, sar
6	4	Alarm	Alarm	Alarme
7	5	Warnung	Warning	Avertissemen
8	6	Fehler	Failure	Erreur
9	7	Leittechnik	Process control system	Système de c
10	8	Systemmeldungen	System messages	Alarmes syst
11	9	Bedienmeldungen	Operator input messages	Alarmes de c
12	10	+	+	+

步骤

在“文本”表单上，可编辑现有的文本和添加新的文本。

新文本可以用两种不同的方法来添加。 新文本可以通过分配新的文本标识符来创建。或者通过在任意文本列中进行输入创建新文本。 此种情况下，自动分配文本标识符，但可以更改。

消息文本可以在相应的表单上直接组态，这些表单在文本库中自动创建。

编写

输入文本后，就可以将它写入 WinCC。 如果将包含消息文本的表单写入 WinCC，则文本库也被写入 WinCC。

删除

不能删除消息块名称、消息类别名称和消息类型名称。 如果删除了消息文本或状态文本，它们也从使用的位置被删除。 可以删除消息系统中不使用的文本。

说明

数据从项目文件夹和 WinCC 中永久删除。 数据的删除不能撤消。

特殊功能

要将语言添加到文本库，从“WinCC”菜单中选择“添加 RT 语言”。 显示“选择语言”对话框。 所有可用的语言在其中列出。 选择要添加的语言，然后单击“确定”以关闭对话框。 所选语言添加到新列。

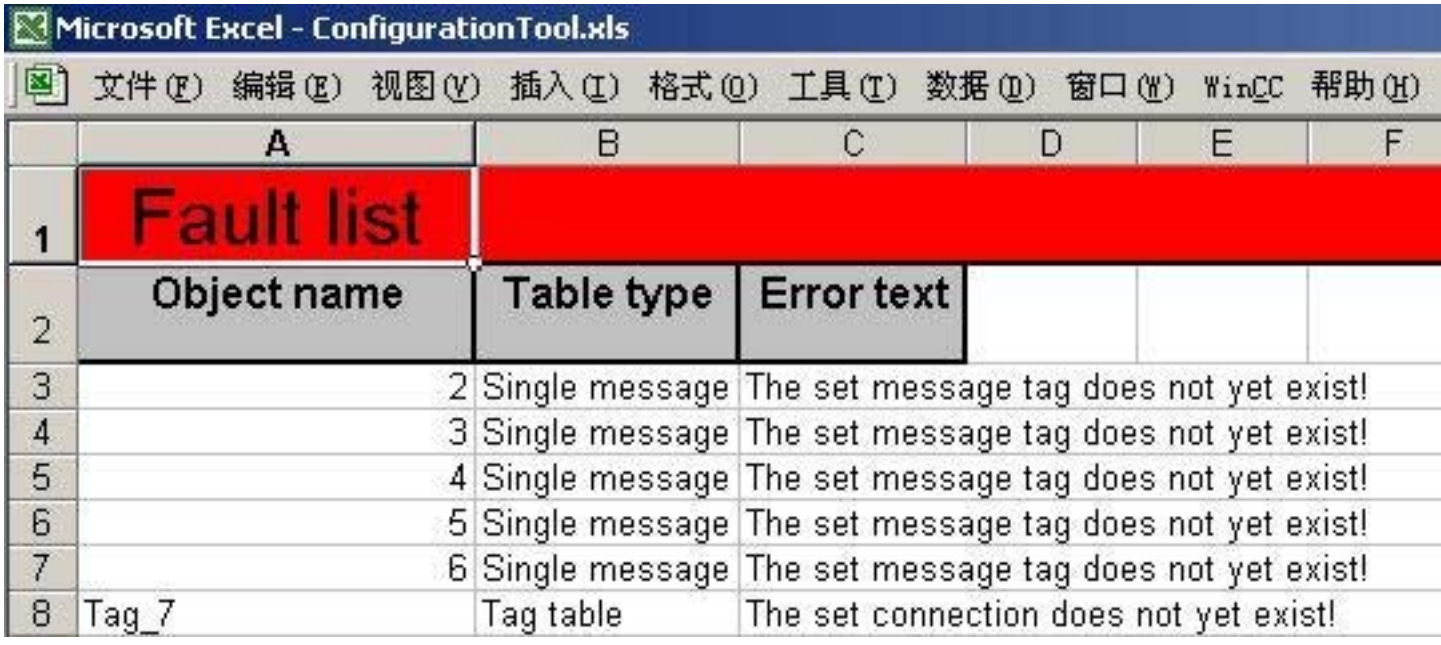
表格结构

列	简述
文本标识符	文本的标识符。 只要文本尚未写入 WinCC，就可以更改文本标识符。
语言	所有已组态文本的列表。 语言列的数目取决于已组态语言的数目。
错误文本	写入 WinCC 时出现的错误简述。 无错误对象分配有错误文本“OK”。

操作“写错误”表单

简介

该表单列出所有包含在项目文件夹中的错误对象。



步骤

写入对象时如果出现错误，则将错误列表添加到项目文件夹中。

错误列表中的第一列包含在其中出现错误的对象名称。 第二列包含表格类型，而非表格名称。 第三列包含出现错误的简述。

双击错误列表中的条目可直接访问错误对象。

表格结构

列	简述
对象名	在其中出现错误的对象的名称。
表格类型	在其中出现错误的表格的类型。
错误文本	错误的简述。

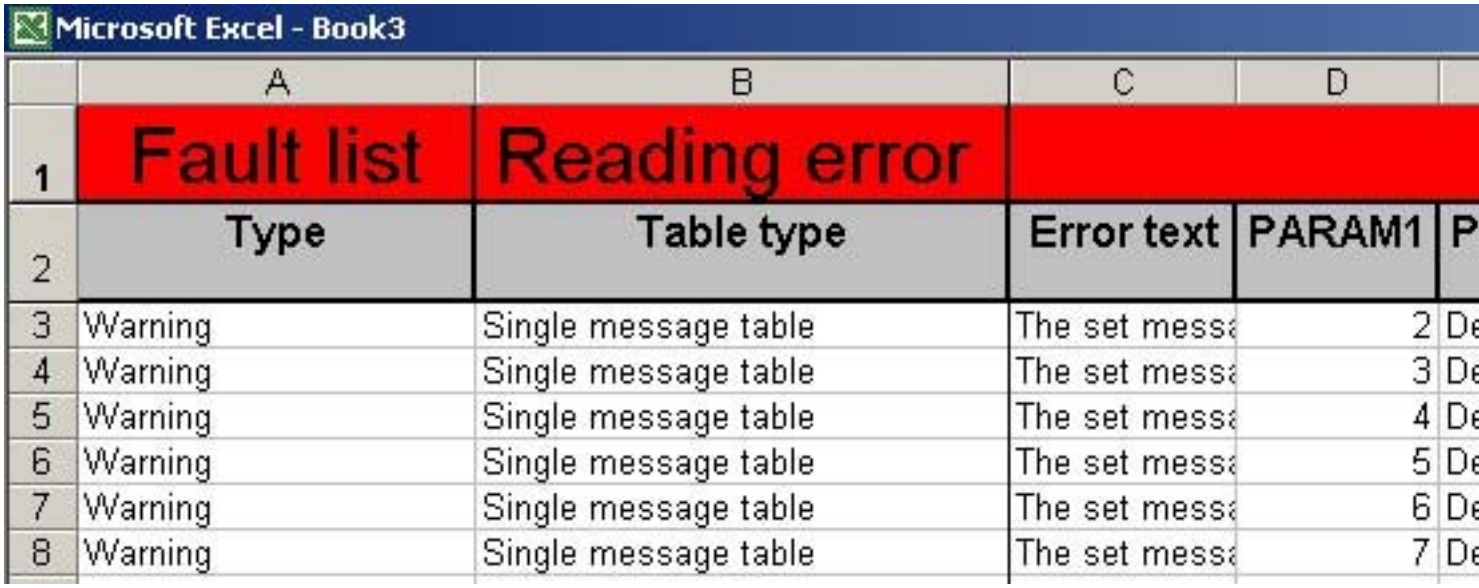
参见

错误列 (页 234)

操作“读错误”表单

简介

该表单列出在从 WinCC 中读取时检测到的所有错误对象。



步骤

读对象出现错误时，错误列表只添加到项目文件夹中。

错误列表的第一列指示错误的类型。 系统发出警告和错误消息。 对于警告的情况，该对象被输入到列表中并更正错误参数。 对于错误的情况，该对象不输入表格中，因为错误不能更正。

第二列包含表格类型，而非表格名称。 第三列包含出现错误的简述。

其它列包含在其中发生读取错误的对象数据。

表格结构

列	简述
类型	错误的类型。
表格类型	在其中出现错误的表格的类型。
错误文本	错误的简述。
参数 1 到参数 89	错误对象的所有参数。

1.8.5.4 对话框

对话框

简介

组态工具中对话框的概述。组态工具对话框在以下章节中进行更详细的描述。

参见

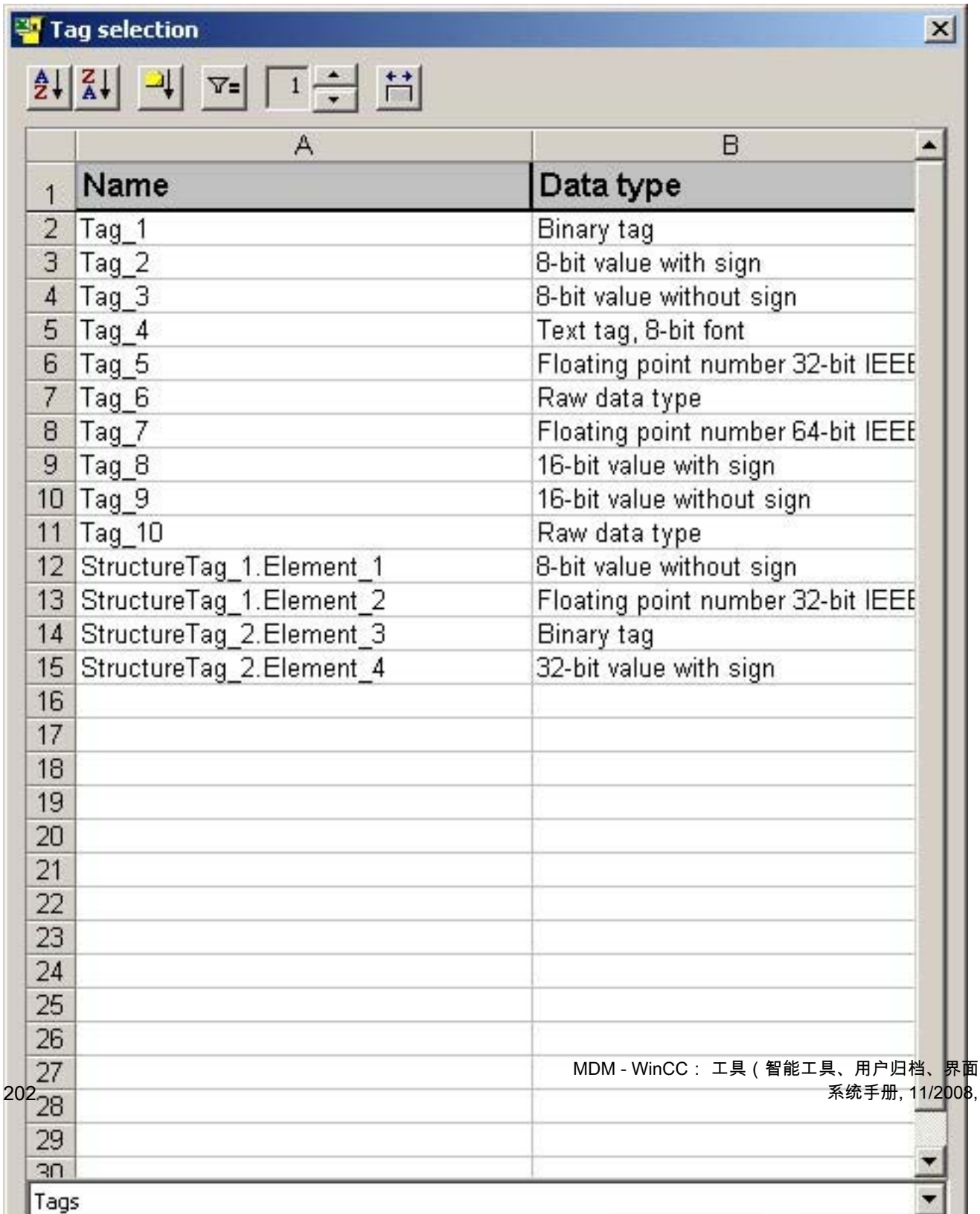
操作“归档变量”对话框 (页 205)

操作“变量”对话框 (页 202)

操作“变量”对话框

简介

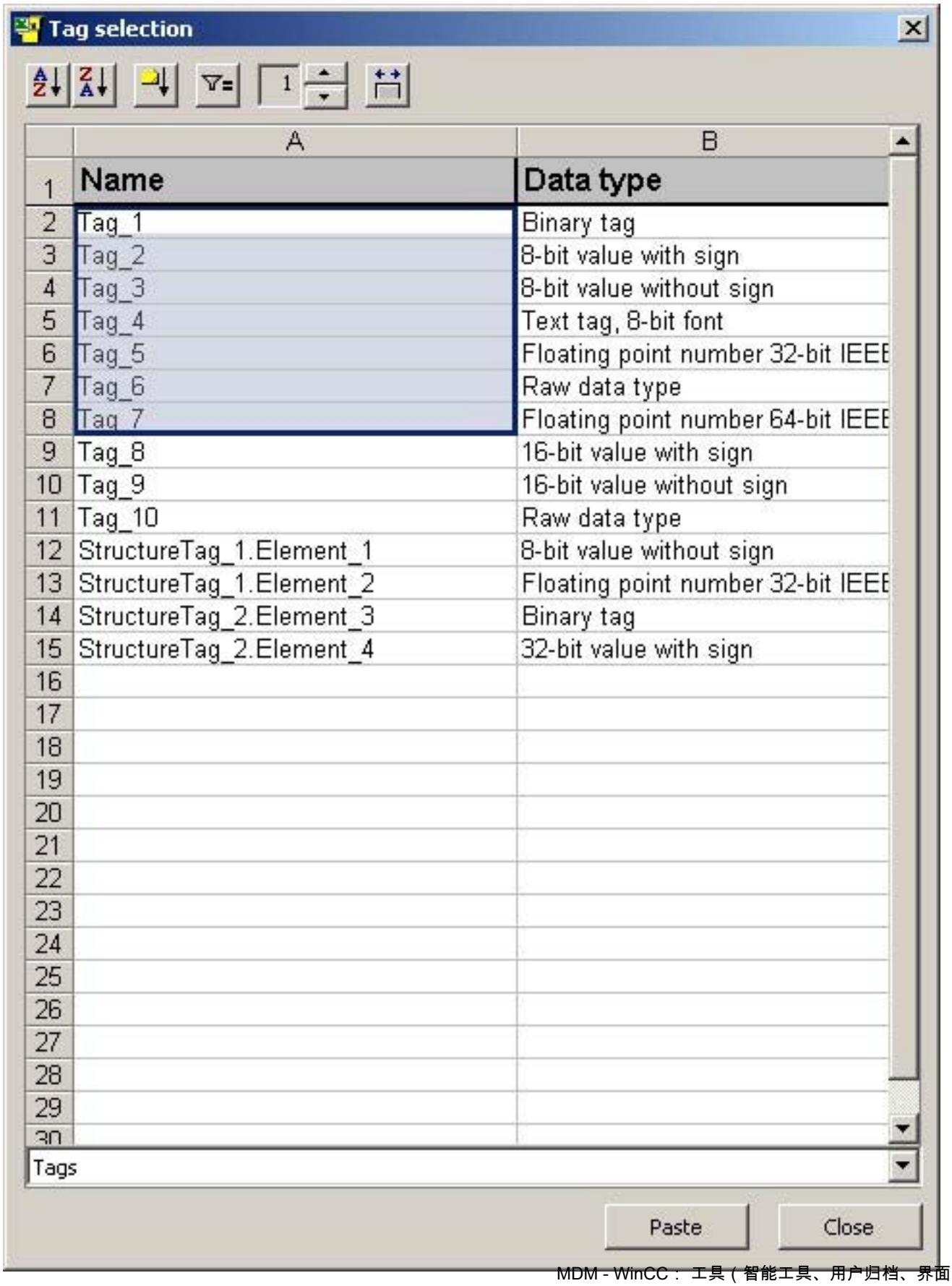
组态工具提供输入变量用的变量对话框。 建议总是使用变量对话框。 首先，这可确保变量存在，其次，在该变量对话框中进行一次操作就可以输入多个变量。 对话框的底部是一个下拉式列表框，从中可以选择要在其中插入变量的变量表。



操作归档变量对话框

通过双击组态工具中要在其中输入变量的列，打开“变量”对话框。 选择要插入的变量。 使用  自旋框定义变量插入的次数。 单击“粘贴”按钮插入变量。

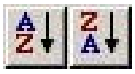
“变量”对话框也允许一次操作插入多个不同的变量。 要执行此操作，选择要插入的变量。


Tag selection
1
A
B
Name
Data type
Tag_1
Binary tag
Tag_2
8-bit value with sign
Tag_3
8-bit value without sign
Tag_4
Text tag, 8-bit font
Tag_5
Floating point number 32-bit IEEE
Tag_6
Raw data type
Tag_7
Floating point number 64-bit IEEE
Tag_8
16-bit value with sign
Tag_9
16-bit value without sign
Tag_10
Raw data type
StructureTag_1.Element_1
8-bit value without sign
StructureTag_1.Element_2
Floating point number 32-bit IEEE
StructureTag_2.Element_3
Binary tag
StructureTag_2.Element_4
32-bit value with sign
Tags
Paste
Close

重复使用“粘贴”按钮插入所选择的变量，插入次数如  自旋框中所定义。

“归档变量”对话框的按钮

- 按名称对变量排序

  单击  按钮按名称以字母顺序对变量排序（升序或降序）。

- 按数据类型对变量排序

  单击  按钮根据变量数据类型按降序对变量进行排序。

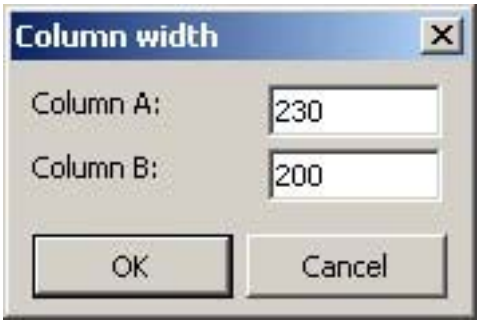
- 应用过滤器

  单击  按钮为“名称”和“数据类型”列定义过滤器。 使用添加的下拉列表框选择过滤器。

  要复位过滤器，再次单击  按钮。

- 设置对话框中的列宽

  单击  按钮访问“列宽”对话框。 在该对话框中，可修改两列的宽度。

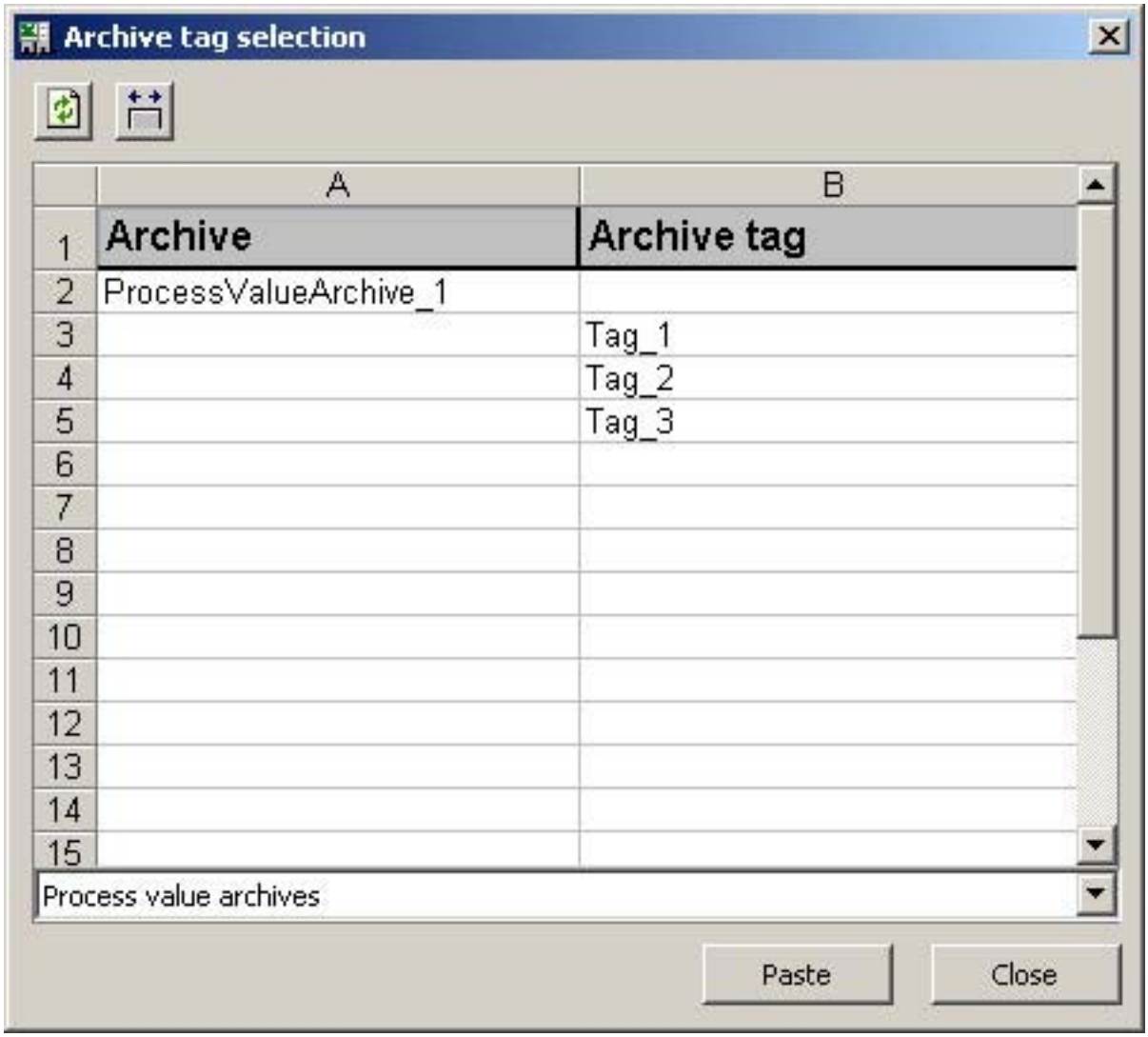


操作“归档变量”对话框

简介

组态工具提供“归档变量”对话框，可以用其在压缩归档表格中输入过程值归档变量。 建议总是使用“归档变量”对话框。 首先，这可确保归档变量存在，其次，在“归档”变量对话框中

进行一次操作就可以输入多个变量。 对话框的底部是一个下拉式列表框，从中可以选择要在其中插入变量的过程值表。



操作“变量”对话框

通过双击组态工具中要在其中输入变量的列，打开“归档变量”对话框。 选择要插入的变量。 单击“粘贴”按钮插入变量。

“归档变量”对话框也允许一次操作插入多个不同的变量。 要执行此操作，选择要插入的变量。

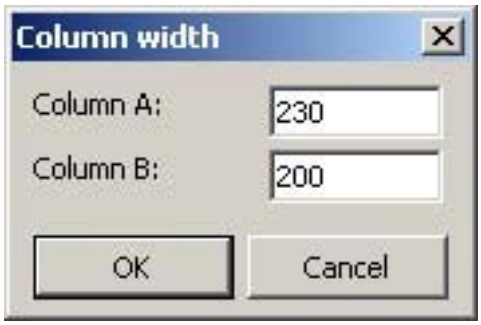
“归档变量”对话框的按钮

- 更新显示

  单击  按钮更新“归档变量”对话框的视图。

- 设置对话框中的列宽

  单击  按钮访问“列宽”对话框。 在该对话框中，可修改两列的宽度。



1.8.5.5 处理组态的数据

处理组态的数据

简介

通常，Excel 提供的所有功能都可以不受限制地使用。 然而，在处理组态工具中的组态数据时，必须将几点特殊情况考虑进去。 在以下章节中详细说明这几点。

参见

对数据排序 (页 215)

设置筛选 (页 215)

剪切数据 (页 214)

复制数据 (页 214)

从项目文件夹和 WinCC 中删除数据 (页 211)

将数据写入 WinCC (页 208)

将数据写入 WinCC

简介

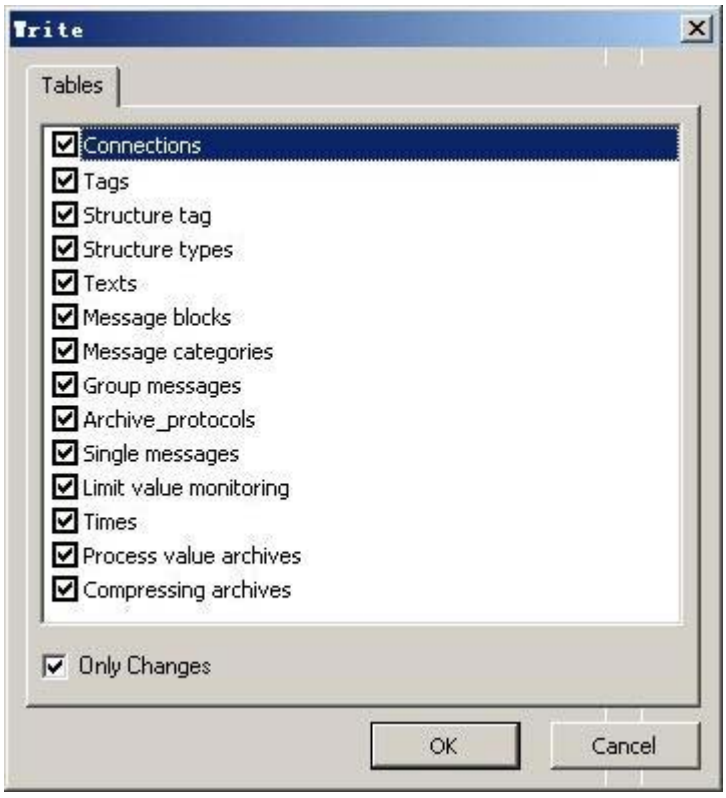
为了将数据写入 WinCC，必须存在对应于 WinCC 项目的连接。 以下选项可用于将数据写入 WinCC。

说明

如果在写数据后，WinCC 项目中的数据和项目文件夹中的数据不匹配，可能是因为 WinCC 项目管理器的更新出了问题。 要更新显示，关闭 WinCC 项目然后再打开。

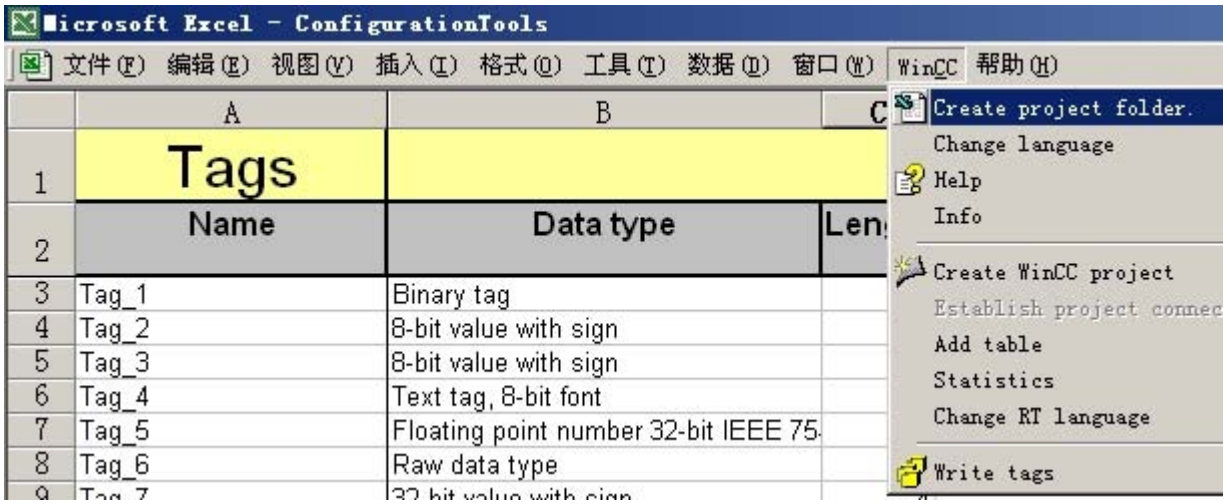
使用工具栏向 WinCC 中写入

1. 单击工具栏中的 按钮。 “写”对话框打开。
2. 使用对话框选择要写入 WinCC 的数据所在的表单。
3. 使用“仅更改”复选框指定只将更改的对象写入 WinCC。 如果未选中该复选框，则将所有对象写入 WinCC 中。
4. 单击“确定”按钮将工作表中的数据写入 WinCC。

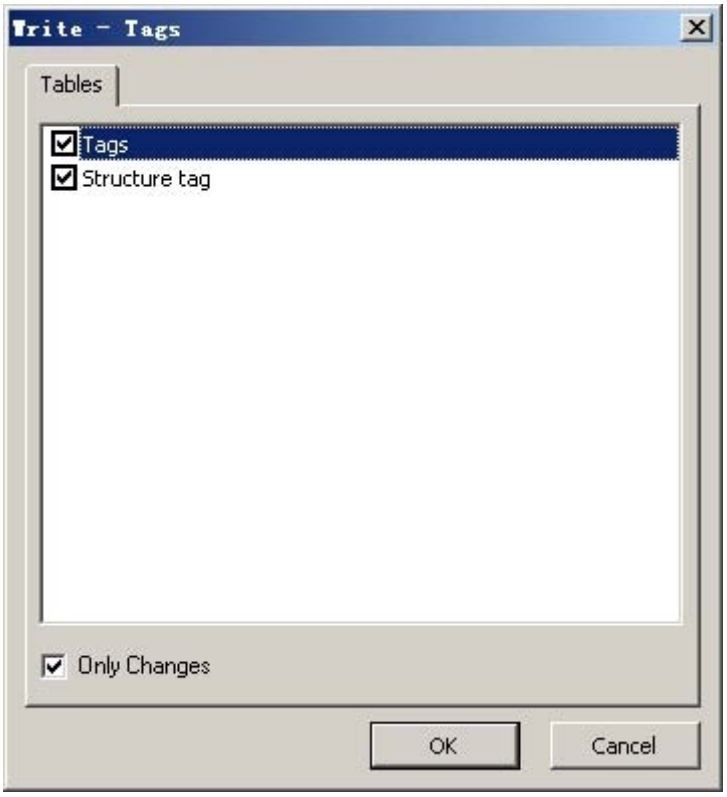


使用下拉菜单向 WinCC 中写入

1. 选择“WinCC”下拉菜单。
2. 根据项目文件夹中激活的表，用于写对象的菜单条目在下拉菜单中可以获得。 选择“写入变量”选项打开“写”对话框。



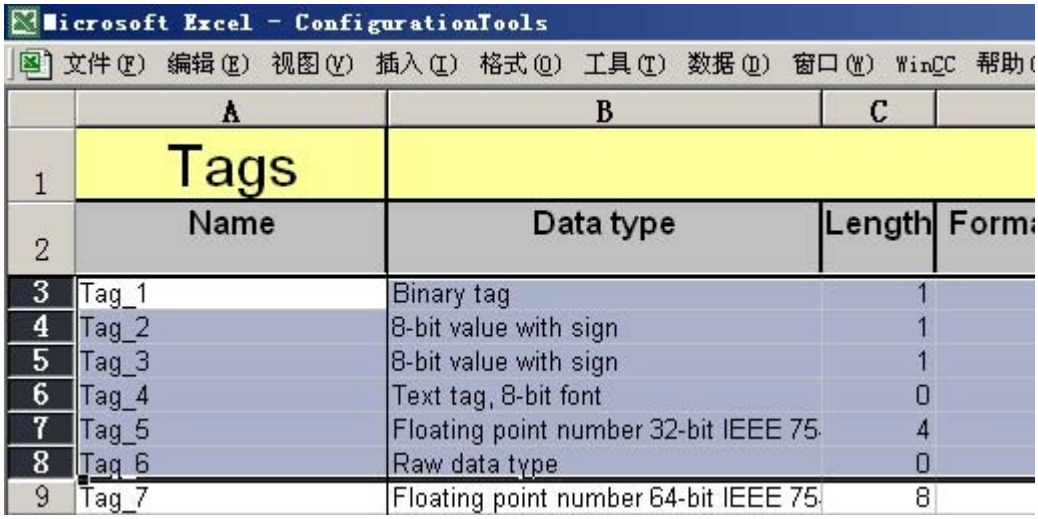
3. 使用对话框选择要写入 WinCC 的数据所在的表单。 请注意，只有与激活表格类型相同的表格才可用。



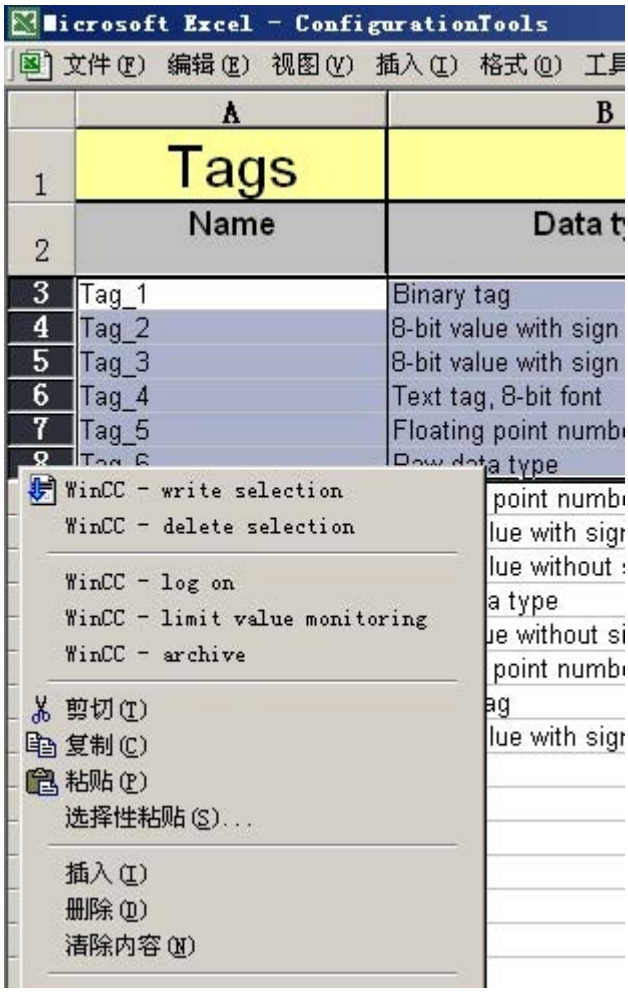
4. 以后步骤按“使用工具栏写入至 WinCC”中所述继续。

使用弹出式菜单向 WinCC 中写入

1. 选择要写入的对象。 必须选择整行。



2. 使用鼠标右键打开行弹出式菜单，然后选择“WinCC - 写选择”。 将所有选择的对象写入 WinCC。 通过使用弹出式菜单，只有修改后的对象才写入 WinCC。



从项目文件夹和 WinCC 中删除数据

简介

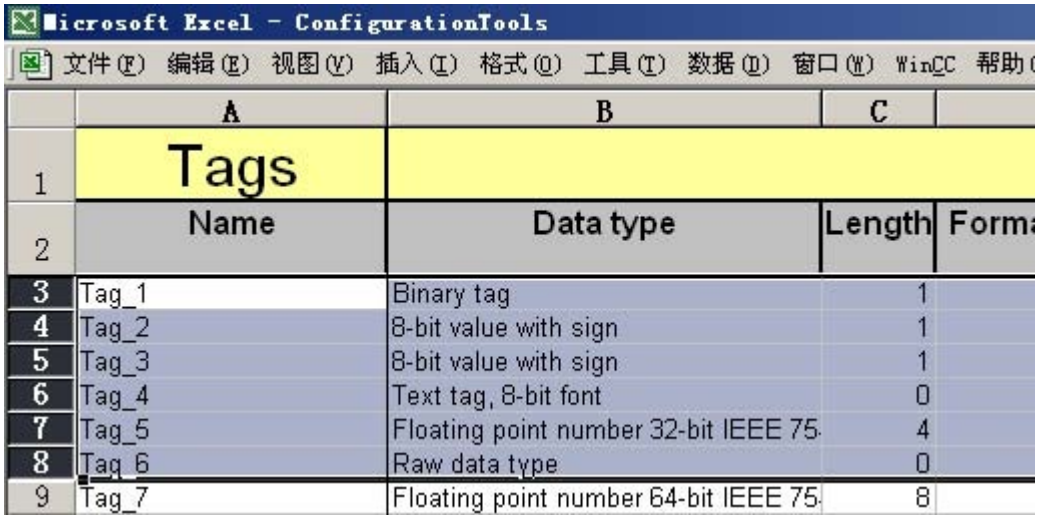
不能从 WinCC 对象的数据区中删除行、列或单元格。 要删除 WinCC 对象，使用弹出式菜单条目“WinCC - 删除选择”。 删除完整表单时，在其上面组态的所有 WinCC 对象也一起删除。 请注意并非所有的表单都可以删除。

说明

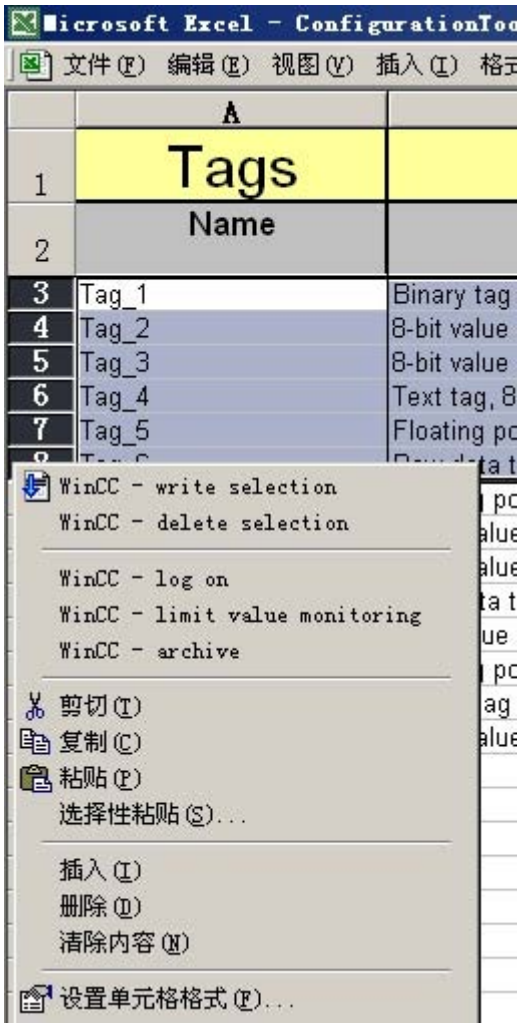
如果在写数据后，WinCC 项目中的数据和项目文件夹中的数据不匹配，可能是因为 WinCC 项目管理器的更新出了问题。 要更新显示，关闭 WinCC 项目然后再打开。

通过弹出式菜单进行删除

1. 选择要删除的对象。 必须选择整行。

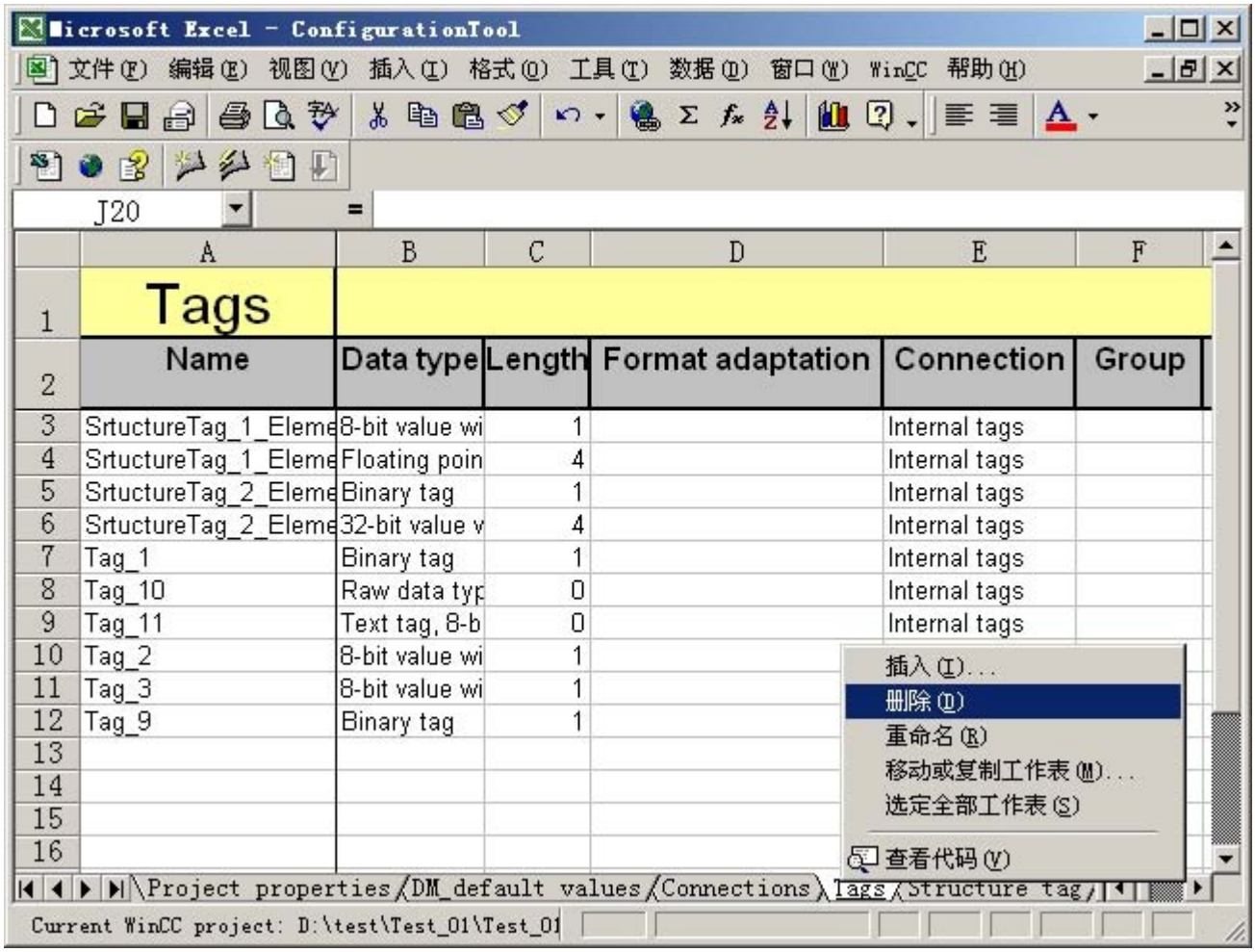


2. 使用鼠标右键打开行弹出式菜单，然后选择“WinCC - 删除选择”。 删除所有选择的对象。



删除表单

1. 使用鼠标右键打开与表单相关的弹出式菜单，然后选择“删除”。 该表单连同在其上面组态的数据一起删除。



不带至 WinCC 连接的删除

要删除对象，不必存在至分配的 WinCC 项目的连接。 下次建立至分配的 WinCC 项目的连接时，比较数据，从项目文件夹中删除的所有对象也从 WinCC 中删除。

带至 WinCC 连接的删除

从项目文件夹中删除的对象也从 WinCC 中删除（当它们存在于 WinCC 项目中时）。 请注意删除的数据不能恢复。

复制数据

简介

使用 Excel，数据可以通过弹出式菜单、下拉菜单、工具栏和组合键来复制。 数据可以在 WinCC 组态工具中无限制地进行复制和粘贴。 对插入的数据进行检查，并在必要时更正数据。 要复制整个 WinCC 对象，按以下所述进行。

复制 WinCC 对象

1. 访问“项目属性”表单并将“使用默认值”参数设置为“否”。
2. 复制对象的整行。
3. 将复制的对象粘贴到相同类型的表格中。 例如，变量只能粘贴到变量表中。
4. 组态工具检查粘贴的数据。 由于必须使用唯一的名称，Office Assistant 提示是否应生成唯一的名称。 使用“确定”进行确认。

说明

如果在复制 WinCC 对象前已经指定名称，则只能复制对象参数。

剪切数据

简介

在 Excel 中，可在任意位置剪切和粘贴任何数据。 然而，这在 WinCC 组态工具中不能实现。为了在 WinCC 组态工具中剪切数据，必须总是选择整个对象。 只能在相同类型的表单之间剪切和粘贴剪切的数据。 Excel 提供的剪切数据的所有选项都可使用。

剪切

1. 选择要剪切的对象。 请选择整行或所有对象参数。
2. 剪切数据。

粘贴

1. 选择剪切的数据应粘贴到的区域，或者选择作为插入数据起始点的初始单元格。 如果选择了一个区域，剪切大小和粘贴大小必须相同。
2. 粘贴数据。

对数据排序

简介

WinCC 组态工具可用于按列对项目文件夹中的数据进行排序。 这可通过两种方式进行。

通过双击排序

1. 要使用 WinCC 组态工具对数据按升序或降序排序，只要双击参数标题即可。 表单上的数据于是按该列进行排序。

通过菜单排序

1. 选择控制菜单条目“数据”中的“排序”子菜单。 显示“排序”对话框。
2. 在“排序”对话框中，可以指定最多三列要按升序或降序排序的列。 单击“确定”确认排序顺序。

设置筛选

简介

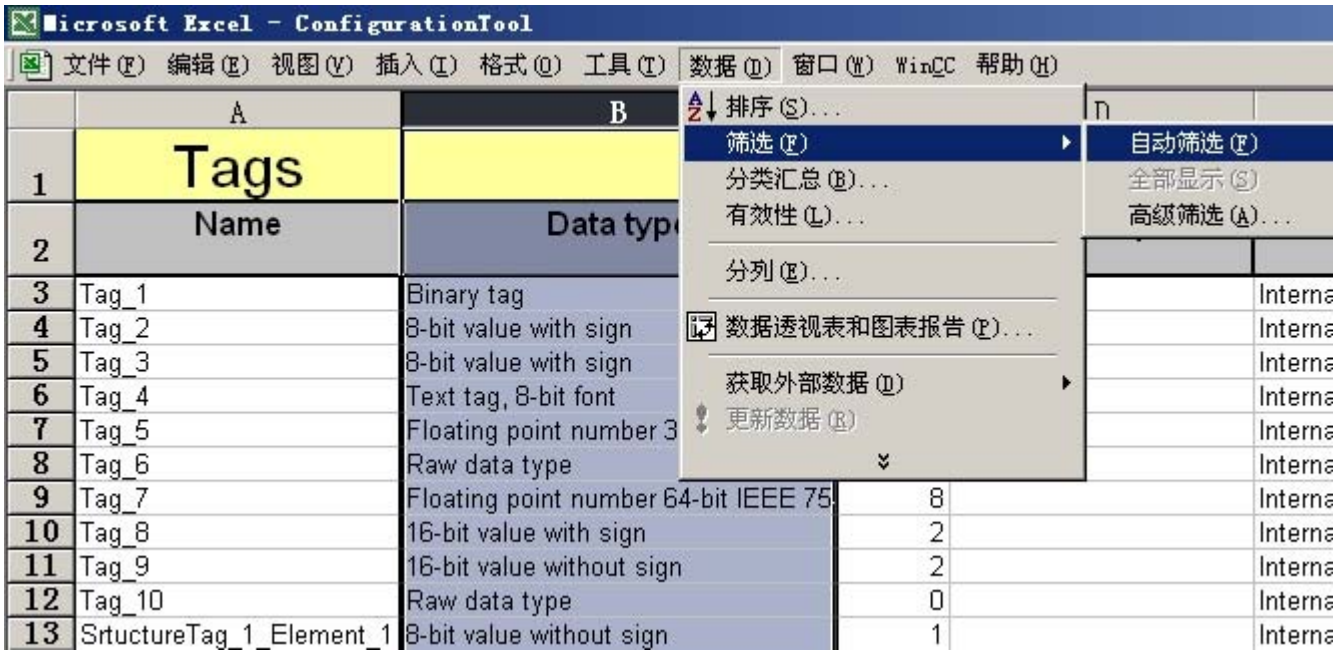
筛选是标准 Excel 功能。 筛选可用于以更简明的方式组织表格中的数据。 例如，使用变量表时，可以为“数据类型”列定义一个过滤器，以便只显示一个特殊的数据类型。

说明

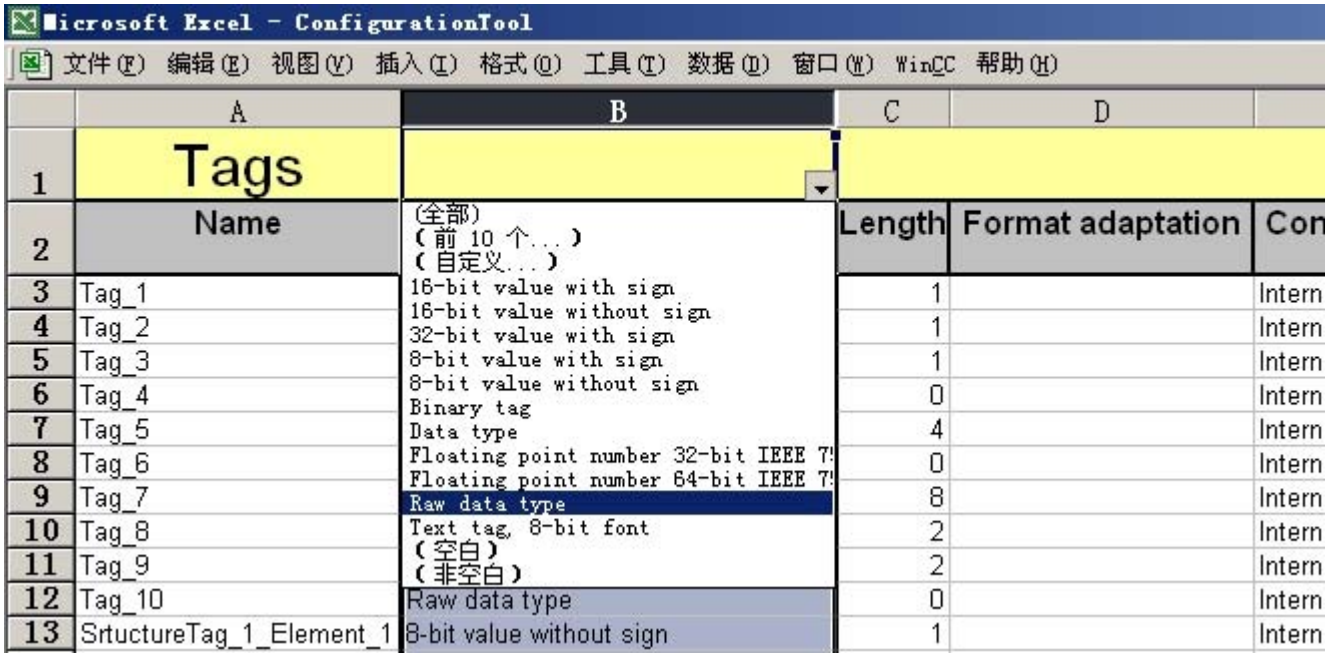
应用筛选后，如果尝试使用弹出式菜单向 WinCC 写数据或删除数据，只有显示的数据才会被写入或删除。

设置过滤器

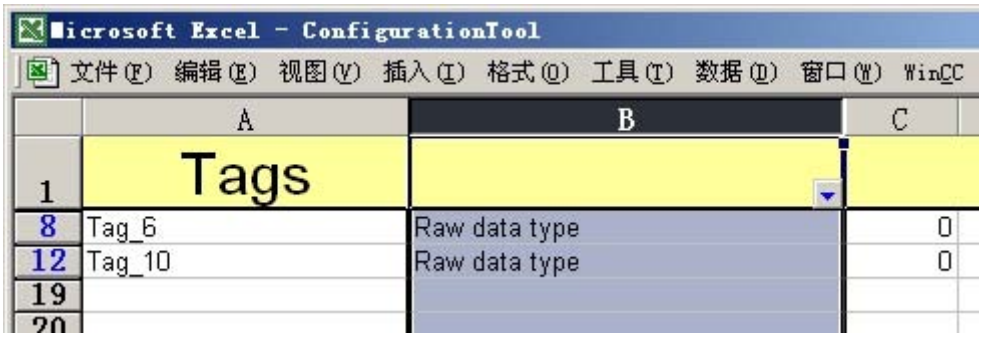
1. 突出显示“变量”表单中的整个“数据类型”列。
2. 选择菜单“数据”中的“过滤器”选项。 选择子选项“自动过滤器”。 为“数据类型”列定义一个过滤器。



3. 附加到“数据类型”列顶部单元格的下拉式列表框指示设置了筛选。 现在可以在下拉式列表框中选择要设置的筛选条件，如“原始数据类型”。

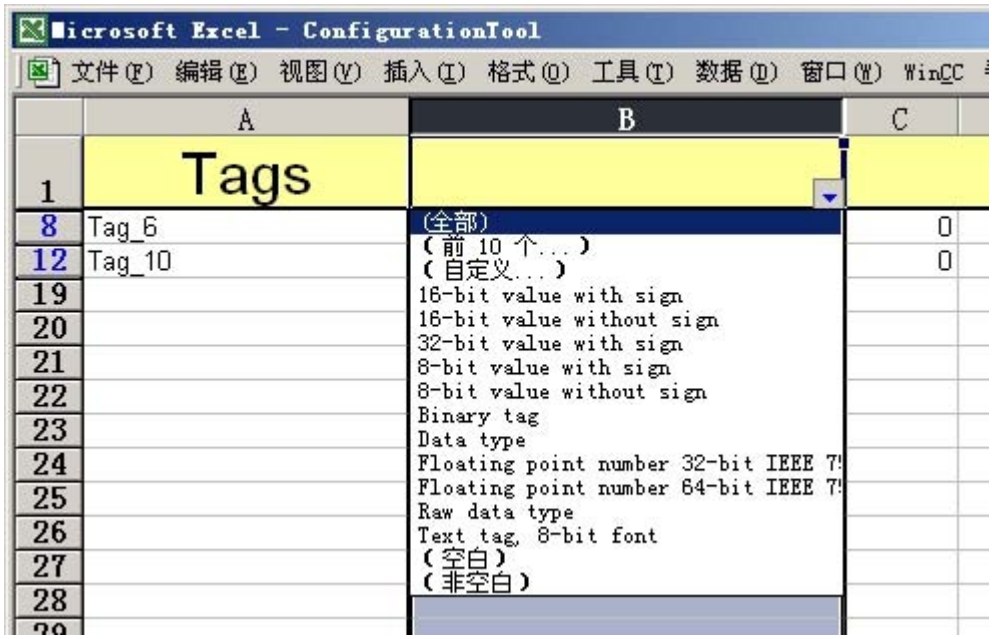


4. 只有数据类型符合“原始数据类型”的变量才显示在表格中。

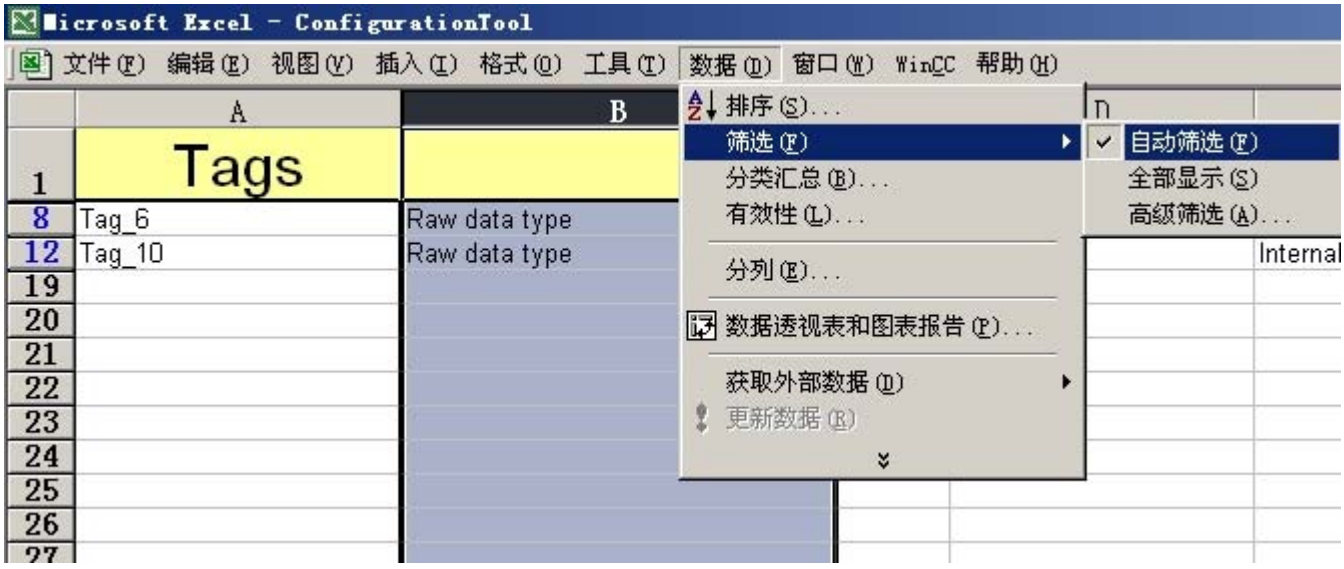


重新设定过滤器

1. 在筛选下拉式列表中，选择“（全部）”条目。 显示表单中的所有变量。



2. 要永久删除筛选，单击“数据”菜单。 选择“筛选”选项。 选择子选项“自动过滤器”。 筛选被删除。

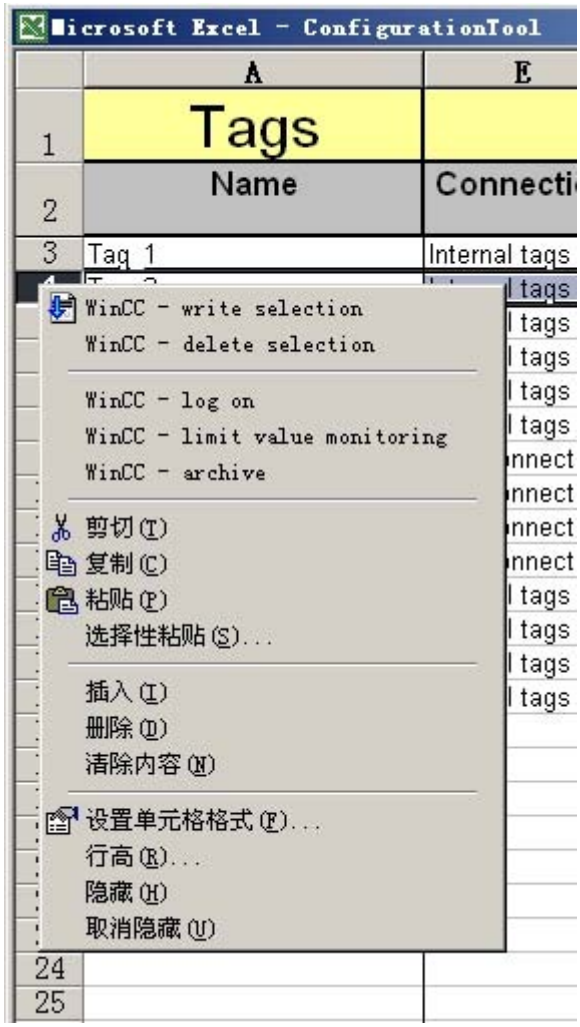


1.8.5.6 通过变量表的弹出式菜单来创建对象

通过变量表的弹出式菜单来创建对象

简介

WinCC 组态工具提供选项，可以通过使用变量表来创建对象。 这可通过变量表行弹出式菜单完成。 通过该菜单可很容易地创建单个消息、限制值监视过程和归档变量。 此操作的步骤在下面的章节中描述。



参见

从变量表创建归档变量 (页 220)

从变量表创建限制值监视 (页 227)

从变量表创建单个消息 (页 223)

从变量表创建归档变量

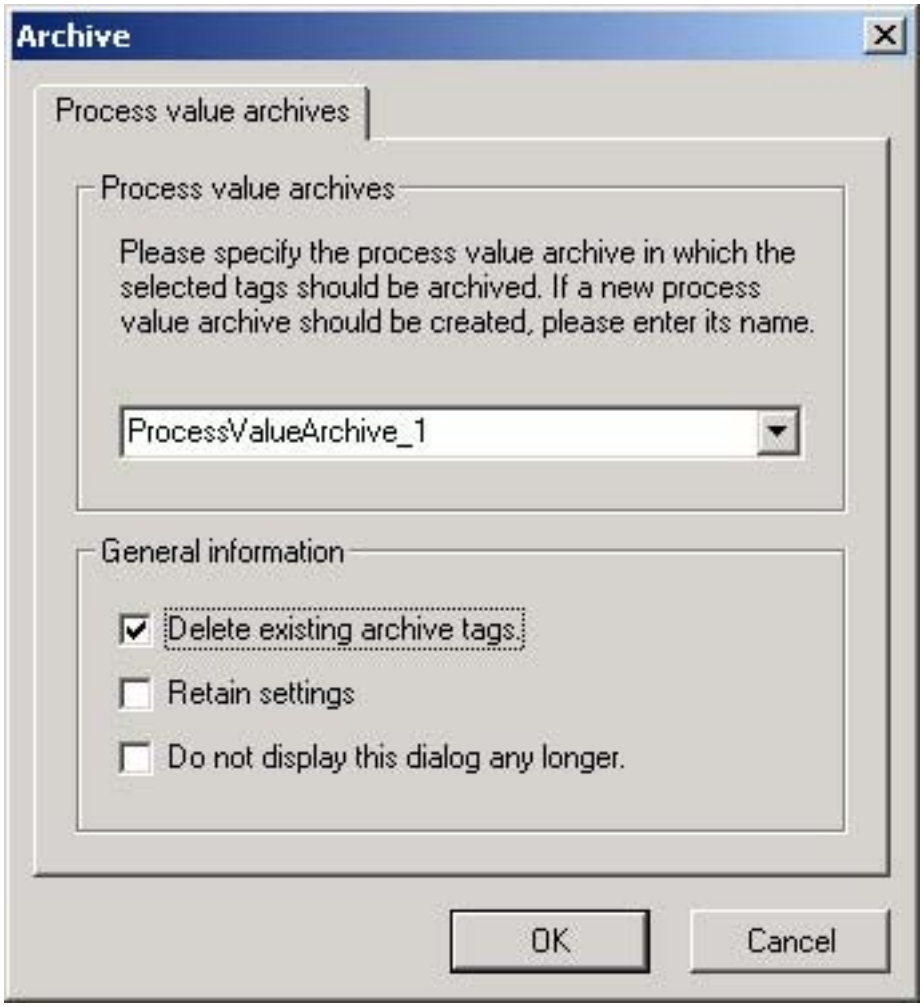
简介

WinCC 组态工具提供从变量表创建归档变量的选项。这可通过变量表弹出式菜单完成。

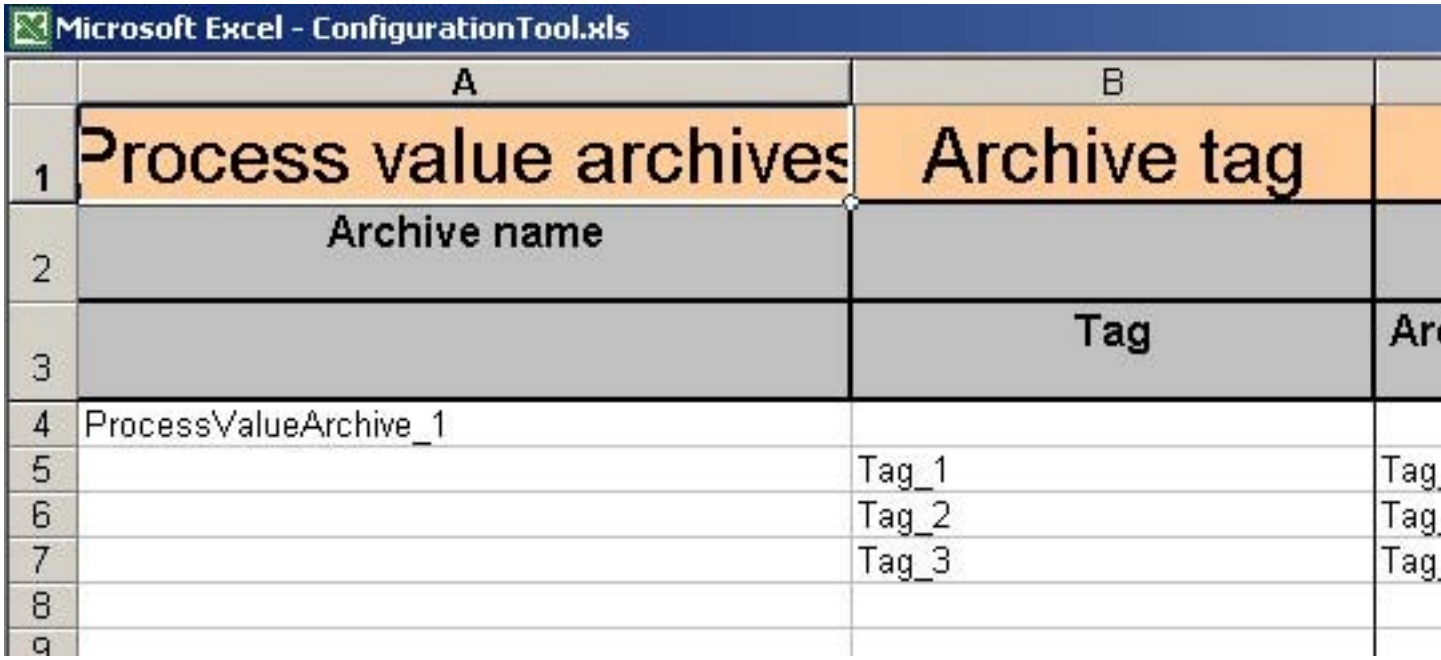
步骤

1. 选择要归档的所有变量。
2. 打开行弹出式菜单并选择“WinCC - 归档”条目。显示“归档”对话框。

3. 在“归档”对话框中，可使用下拉式列表框选择要用于归档的过程值归档。如果要创建新的过程值归档，在下拉式列表框中输入过程值归档的名称。单击“确定”按钮建立归档变量。



4. 将在过程值归档表格中创建相应的过程值归档变量。



从变量表创建单个消息

简介

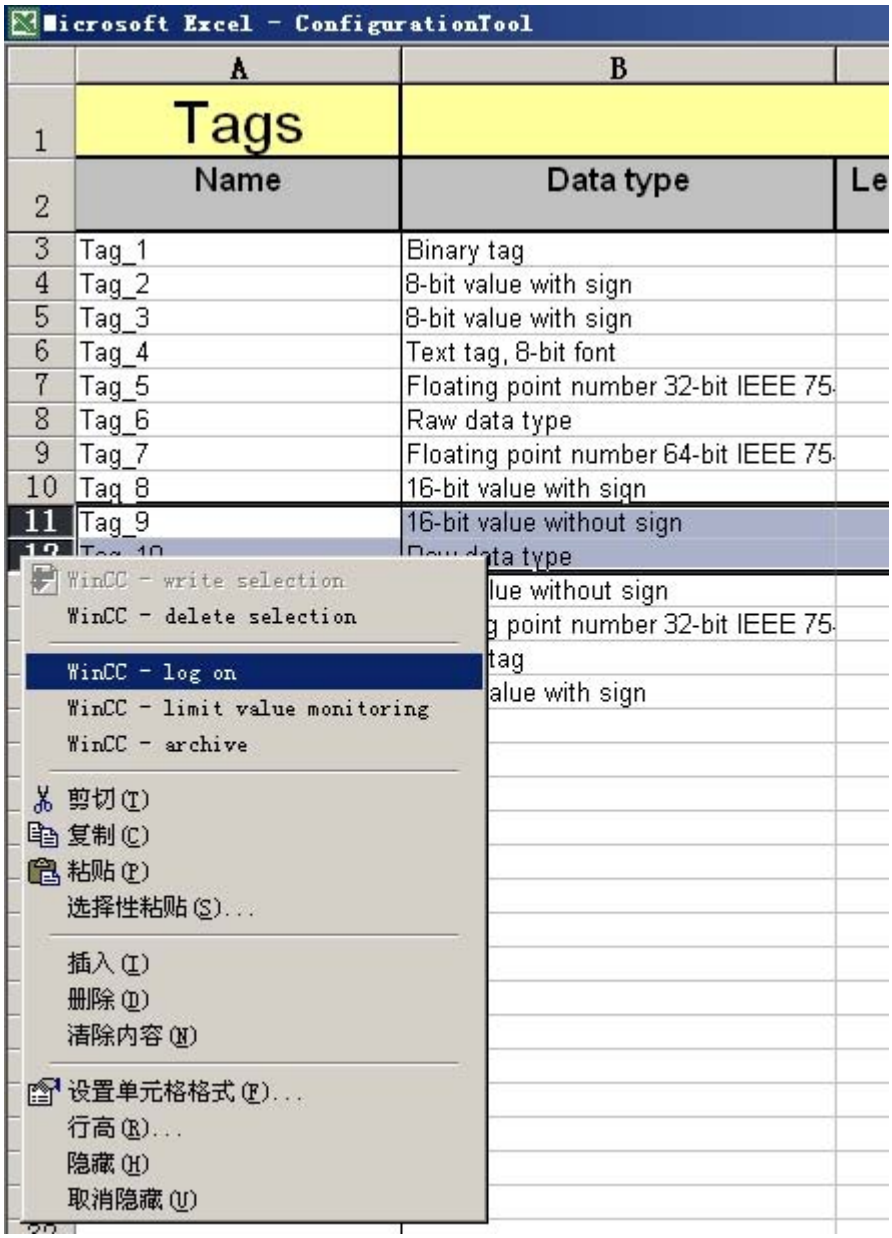
WinCC 组态工具提供从变量表创建单个消息的选项。这可通过变量表弹出式菜单完成。

说明

可以通过“报警记录默认值”表单调整对话框中的设置。

步骤

1. 选择要为其生成消息的所有变量。
2. 打开行弹出式菜单并选择“WinCC - 登录”。 显示“登录”对话框。

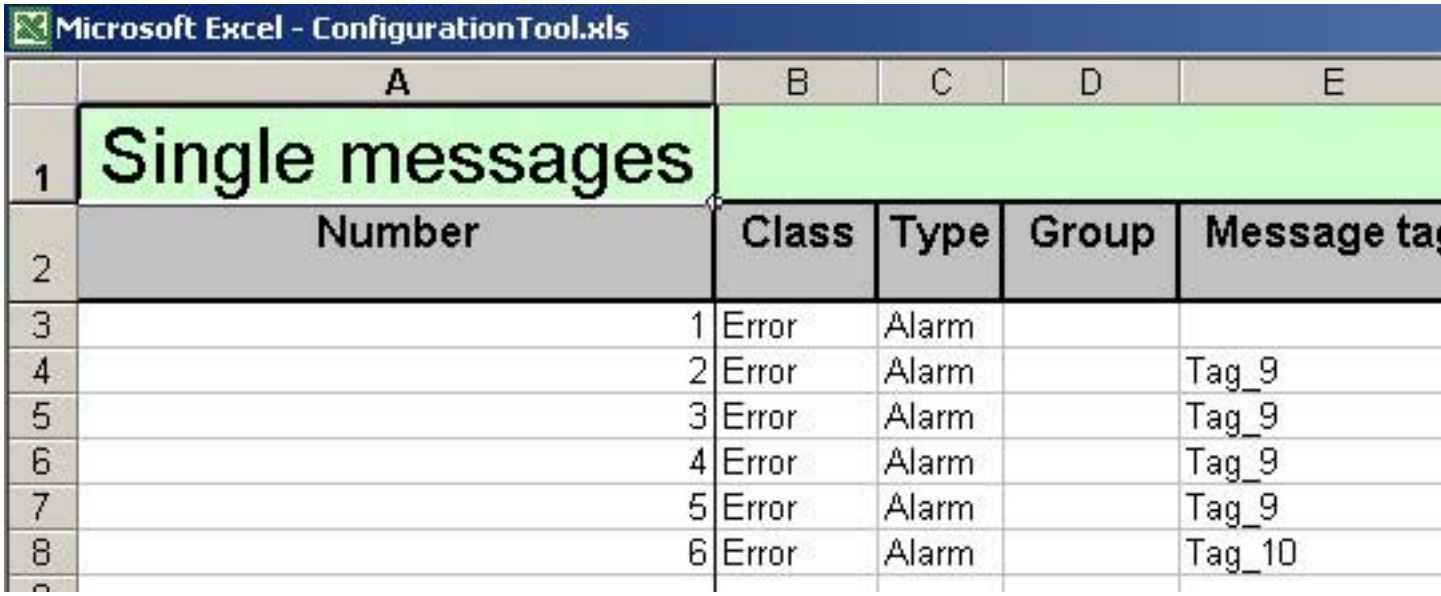


3. 使用“登录”对话框选择要生成单个消息的变量位。 如果要删除变量的可能已存在的所有消息并生成新的消息，选中“删除现有消息”复选框。 如果未选中该筛选框，将保留现有单个消息并且只生成不存在的消息。

4. 对于原始数据变量，输入要生成的单个消息的数目。

单击“确定”生成消息。

5. 相关的消息在单个消息表格中生成。


Microsoft Excel - ConfigurationTool.xls

	A	B	C	D	E
1	Single messages				
2	Number	Class	Type	Group	Message tag
3	1	Error	Alarm		
4	2	Error	Alarm		Tag_9
5	3	Error	Alarm		Tag_9
6	4	Error	Alarm		Tag_9
7	5	Error	Alarm		Tag_9
8	6	Error	Alarm		Tag_10

从变量表创建限制值监视

简介

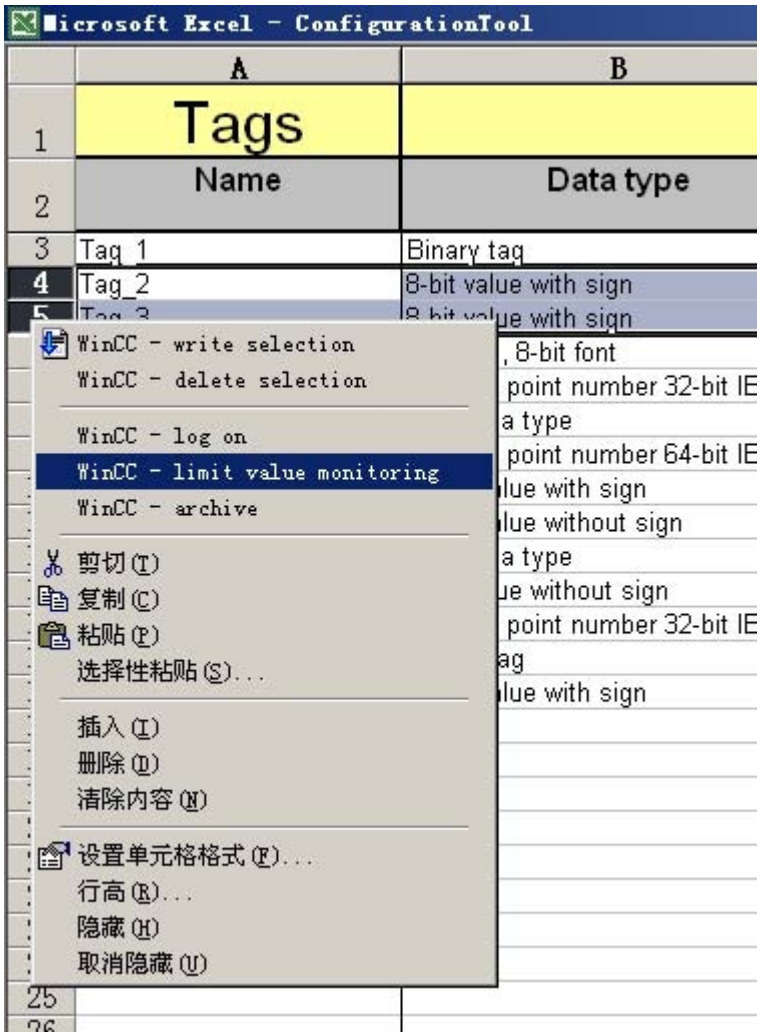
组态工具提供从变量表创建限制值监控的选项。这可通过变量表弹出式菜单完成。

说明

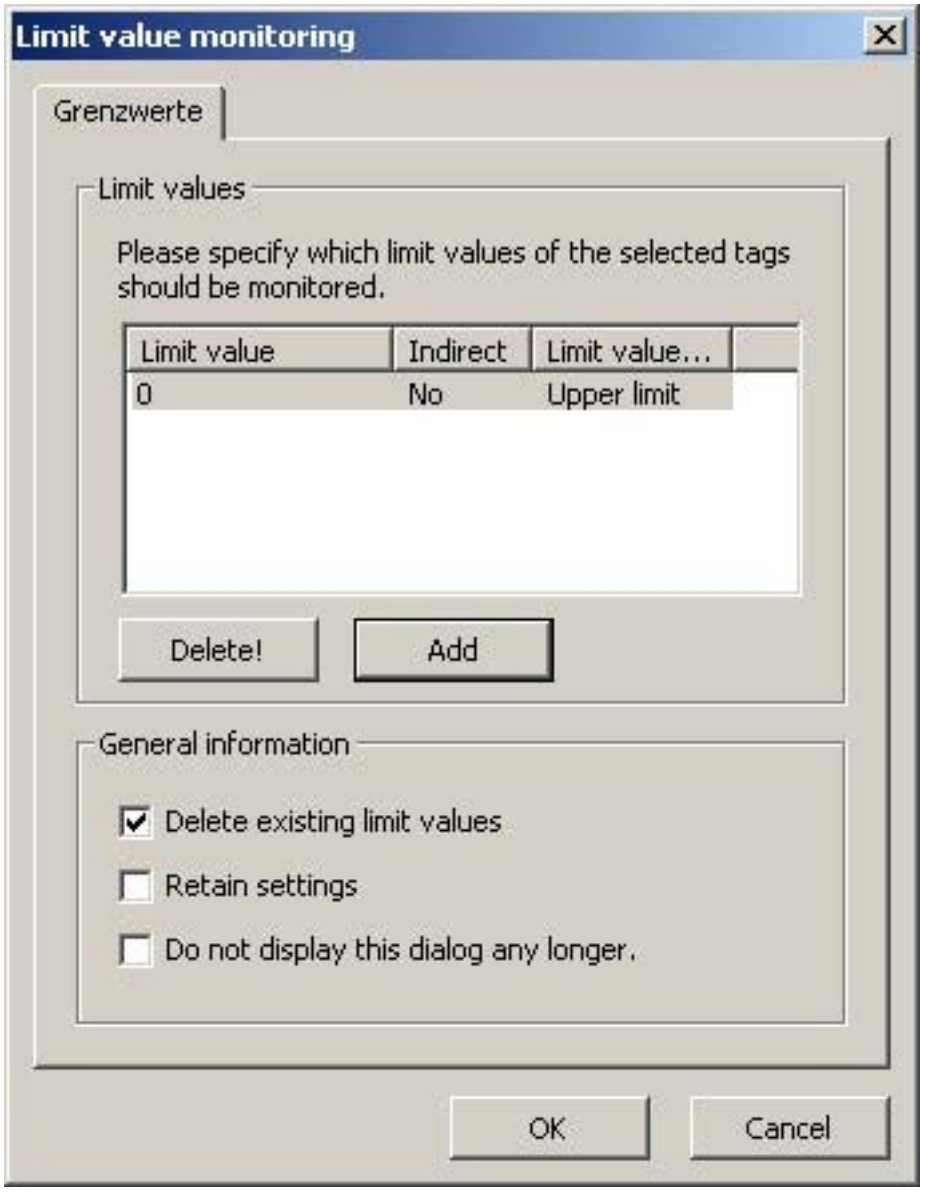
可以通过“报警记录默认值”表单调整对话框中的设置。

步骤

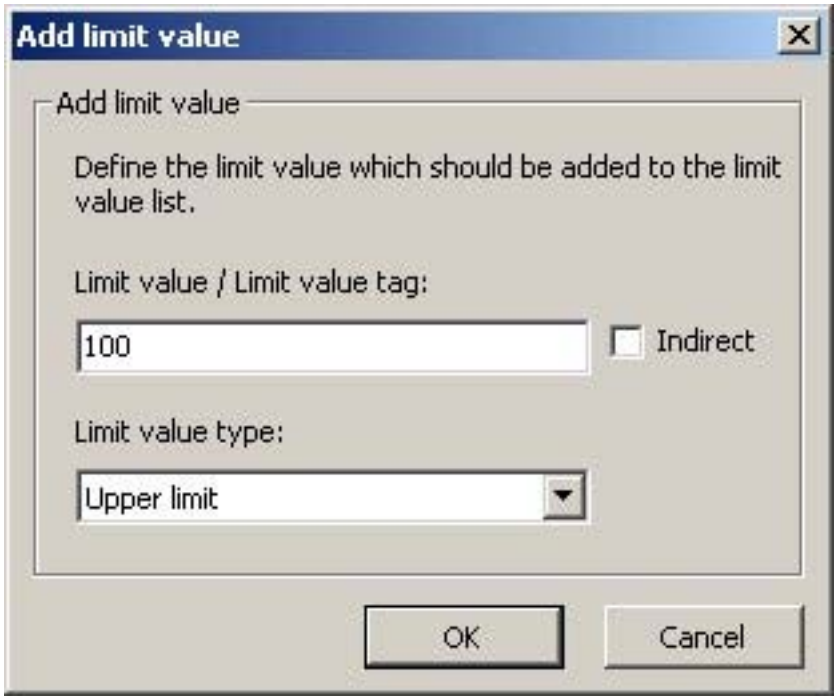
1. 选择要为其建立限制值监视的变量。
2. 打开变量表的行弹出式菜单并选择“WinCC - 限制值监视”。 显示“限制值监视”对话框。



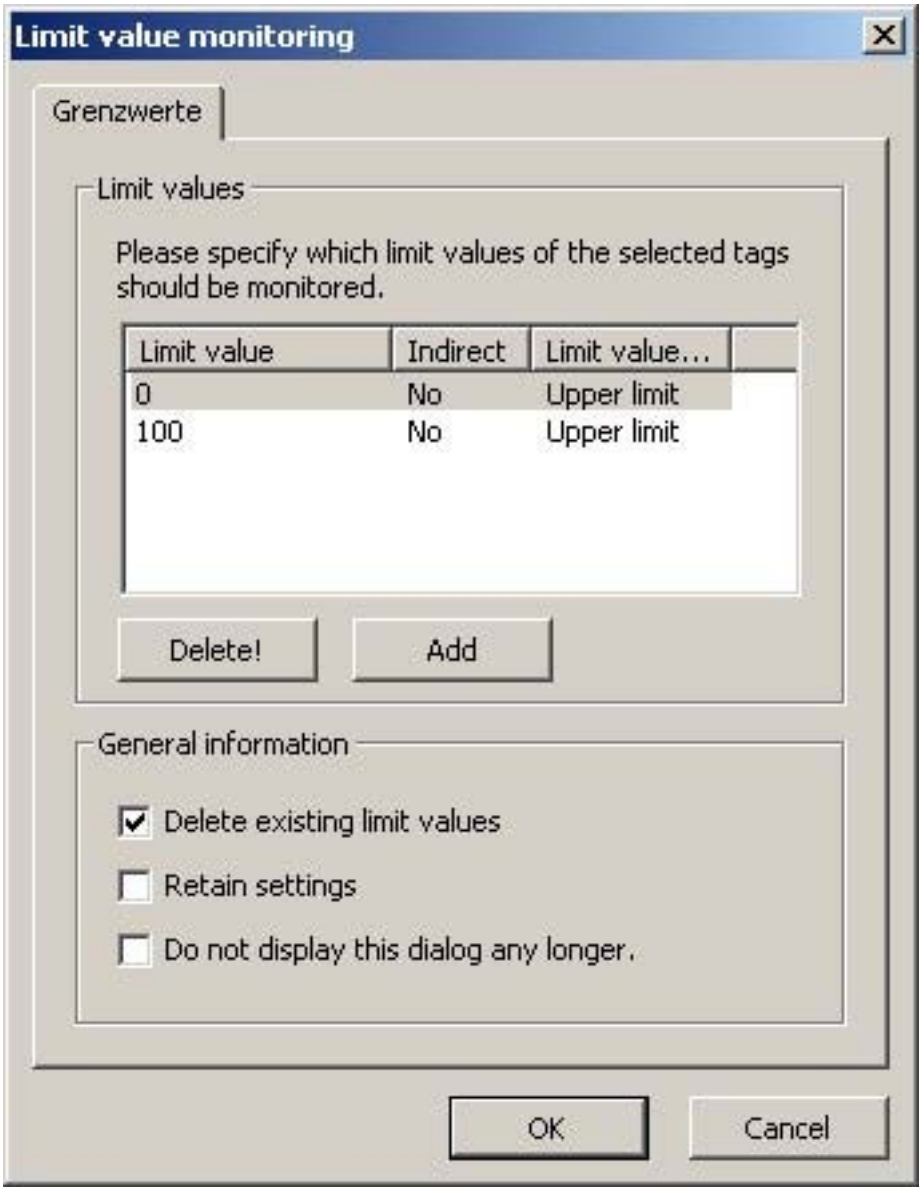
3. 默认值表格中存在的限制值已经输入到“限制值监视”对话框中。 单击“添加”按钮为变量添加更多限制值。 显示“添加限制值”对话框。



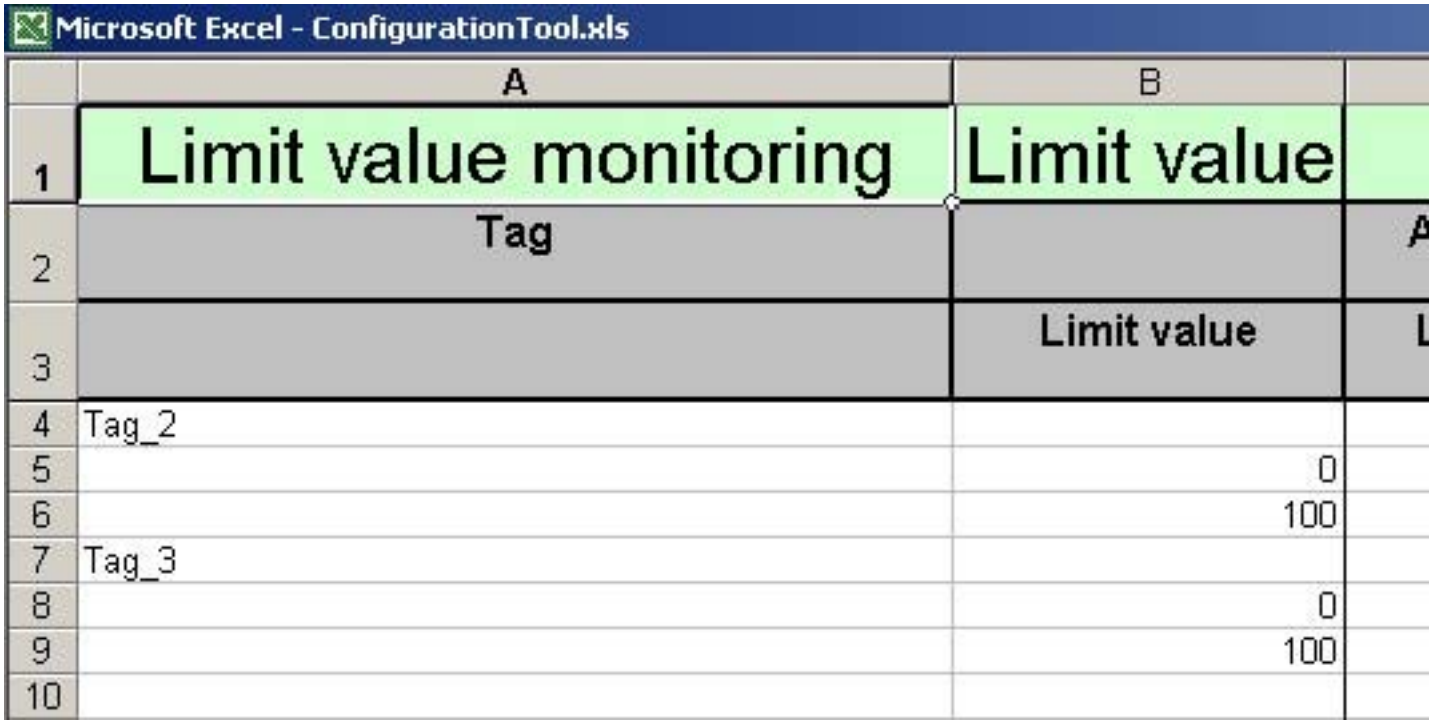
4. 限制值可以在“添加限制值”对话框中输入。 确保新限制值在由变量确定的实际限制范围内。 为了从变量中间接读取限制值，选择“间接”复选框并输入已存在变量的名称替代限制值。 使用“限制”下拉式列表定义限制值是上限还是下限。 单击“确定”按钮后，限制值在“限制值监控”对话框中接受。



5. 使用“删除已存在的限制值”复选框定义所选变量已存在的限制值是否删除。 单击“确定”按钮后创建限制值。



6. 变量的限制值在“限制值监控/限制”工作表上生成。


Microsoft Excel - ConfigurationTool.xls

	A	B
1	Limit value monitoring	Limit value
2	Tag	
3		Limit value
4	Tag_2	
5		0
6		100
7	Tag_3	
8		0
9		100
10		

1.8.5.7 改变数据存储位置

改变数据存储位置

简介

WinCC 组态工具可以改变数据存储位置。 例如，可以将项目文件夹传送到不同的计算机或改变 WinCC 项目的存储位置。

参见

传送 WinCC 项目 (页 232)

传送项目文件夹 (页 233)

传送 WinCC 项目

简介

可以毫无问题地将连接到项目文件夹的 WinCC 项目传送到其它存储位置。 它的存在就是为了建立一个新的项目连接。

步骤

1. 关闭 WinCC 项目并将它传送到所需目标位置。 打开连接到项目的项目文件夹。 使用下拉菜单或工具栏建立新的项目连接。
2. 显示“项目连接”的 Excel Office Assistant。 回答“是”以在项目文件夹中记录新的 WinCC 项目路径。
3. 显示“打开”对话框。 选择 WinCC 项目并关闭“打开”对话框。 项目连接就建立了。

传送项目文件夹

简介

组态工具使您可以将项目文件夹传送到其它计算机上。 这样就可以将项目文件夹移动到其它计算机上作进一步的处理。

步骤

1. 保存现有项目文件夹并将它传送给新的计算机。
2. 打开新计算机上的项目文件夹。 Excel Office Assistant 出现。 回答“否”。
3. 组态工具询问您是否要更新引用。 单击“是”以更新新计算机上项目文件夹的引用。

1.8.6 诊断

1.8.6.1 诊断

简介

组态工具提供 Office Assistant 和以诊断和错误定位为目的的错误列。 Office Assistant 在组态数据时提供支持。 使用组态工具执行的每次修改或更正都通过 Office Assistant 启动。

如果向 WinCC 写入时出错，错误就显示在错误列中。 此外，将生成“错误列表”表格。 该表格列出所有项目文件夹错误。

参见

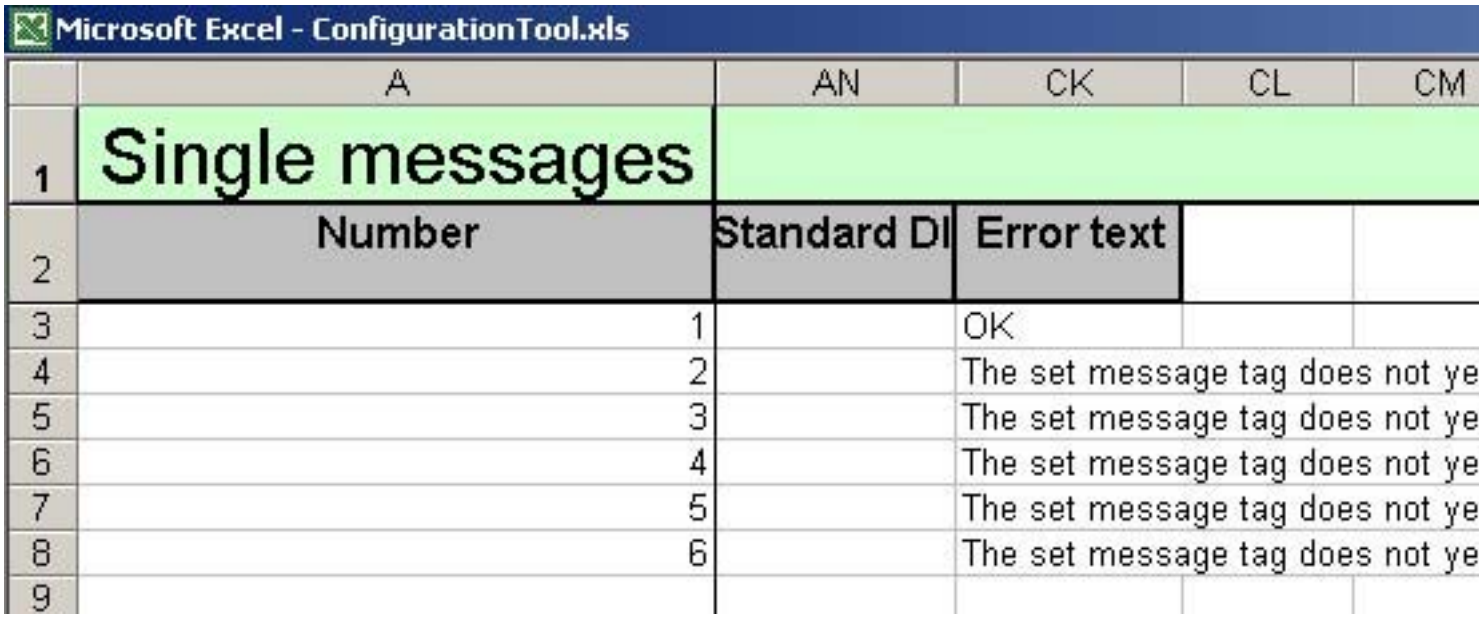
错误列 (页 234)

操作“读错误”表单 (页 200)

1.8.6.2 错误列

简介

可以进行数据组态的每个表单都提供有错误列。 出现错误时，相应的错误文本显示在该列中。 下表显示组态工具中提供的所有错误文本。



	A	AN	CK	CL	CM
1	Single messages				
2	Number	Standard DI	Error text		
3	1		OK		
4	2		The set message tag does not ye		
5	3		The set message tag does not ye		
6	4		The set message tag does not ye		
7	5		The set message tag does not ye		
8	6		The set message tag does not ye		
9					

错误列表

错误文本	原因	更正措施
确定	无错误	无错误
创建中的常规错误！	将整个数据块写入 WinCC 失败。	与 WinCC 仍保持连接吗？
创建对象时出错！	对象直接在 WinCC 中创建吗？	
修改期间的常规错误！	将整个数据块写入 WinCC 失败。	与 WinCC 仍保持连接吗？
修改对象时出错！	对象直接从 WinCC 中删除吗？	
设置的连接还不存在！	设置的连接尚未写入 WinCC。	将设置的连接写入 WinCC。

错误文本	原因	更正措施
消息块名称在文本库中尚未存在！	文本库尚未写入 WinCC。	将文本库写入 WinCC。
不能安装通讯驱动程序！	在系统上未安装设置的通讯驱动程序。	使用不同的通讯驱动程序或安装设置的通讯驱动程序。
设置的冲程变量尚未存在！	设置的冲程变量尚未写入 WinCC。	将设置的冲程变量写入 WinCC。
设置的组还不存在！	设置的组尚未写入 WinCC。	将设置的组写入 WinCC。
设置的结构类型尚未存在！	设置的结构类型尚未写入 WinCC。	将设置的结构类型写入 WinCC。
消息类别名称在文本库中尚未存在！	文本库尚未写入 WinCC。	将文本库写入 WinCC。
一个或多个组消息变量还不存在！	设置的组消息变量在 WinCC 中尚未存在。	将设置的组消息变量写入 WinCC。
消息类型名称在文本库中尚未存在！	文本库尚未写入 WinCC。	将文本库写入 WinCC。
设置的消息类别还不存在！	设置的消息类别在 WinCC 中还不存在。	将设置的消息类别写入 WinCC。
设置的消息类型还不存在！	设置的消息类型在 WinCC 中还不存在。	将设置的消息类型写入 WinCC。
一个或多个消息文本在文本库中还不存在！	文本库尚未写入 WinCC。	将文本库写入 WinCC。
一个或多个过程值块变量还不存在！	设置过程值块变量在 WinCC 中还不存在。	将设置的过程值块变量写入 WinCC。
创建信息文本时出错！	在 WinCC 中已经直接执行了修改。	将 WinCC 中不存在的所有数据写入 WinCC。 关闭当前项目文件夹。 创建带有至用户 WinCC 项目的连接的项目文件夹。
创建循环报警时出错！	在 WinCC 中已经直接执行了修改。	将 WinCC 中不存在的所有数据写入 WinCC。 关闭当前项目文件夹。 创建带有至用户 WinCC 项目的连接的项目文件夹。

错误文本	原因	更正措施
写入高等级对象时出错！	高等级对象不应写入 WinCC。	检查高级对象的错误文本。尝试更正错误并将其重新写入 WinCC。
创建过滤器时出错！	在 WinCC 中已经直接执行了修改。	将 WinCC 中不存在的所有数据写入 WinCC。 关闭当前项目文件夹。 创建带有至用户 WinCC 项目的连接的项目文件夹。
设置变量还不存在！	设置变量在 WinCC 中还不存在。	将设置变量写入 WinCC。
设置消息还不存在！	设置消息在 WinCC 中还不存在。	将设置消息写入 WinCC。
一个或多个状态文本在文本库中还不存在！	文本库尚未写入 WinCC。	将文本库写入 WinCC。
设置的消息变量还不存在！	设置的消息变量在 WinCC 中还不存在。	将设置的消息变量写入 WinCC。
设置的状态变量还不存在！	设置的状态变量在 WinCC 中还不存在。	将设置的状态变量写入 WinCC。
设置的确认变量还不存在！	设置的确认变量在 WinCC 中还不存在。	将设置的确认变量写入 WinCC。
设置的组消息还不存在！	设置的组消息在 WinCC 中还不存在。	将设置的组消息写入 WinCC。
设置的块变量还不存在！	设置的块变量在 WinCC 中还不存在。	将设置的块变量写入 WinCC。
设置的限制值变量还不存在！	设置的限制值变量在 WinCC 中尚未存在。	将设置的限制值变量写入 WinCC。

1.8.7 提示与技巧

1.8.7.1 提示与技巧

简介

以下章节包含一些提示和技巧，旨在帮助您使用 WinCC 组态工具。

参见

数据包 (页 247)

Simatic S7 Protocol Suite 的地址串 (页 243)

特殊字符 (页 243)

VBA 宏 (页 242)

地址生成 (页 240)

行限制 (页 239)

切换工作表 (页 238)

组态工具中的数量结构实例 (页 237)

1.8.7.2 组态工具中的数量结构实例

简介

例如，通过组态工具，可以在 WinCC 中创建大量变量、消息和归档变量。 然而，所使用的硬件和 Office 版本将影响性能。

以下实例基于具有合适值的典型组态。

数量结构 - 实例

组态

通过组态工具将 WinCC 项目读取到 Excel 中。

通过组态工具创建一个新的 WinCC 项目。 并将来自第一个项目的数据写入新的 WinCC 项目中。

数量结构

- 20,000 个内部变量
- 40,000 个过程变量
- 40,000 个归档变量
- 10,000 条消息
- 80,000 个文本

所使用的硬件

- Pentium 4，3.1 GHz
- 内存： 1024 MB
- 虚拟内存： 756 MB

所使用的软件

- Windows XP Professional SP1
- Internet Explorer 6.0 SP1
- Office 2003
- WinCC V6.0 SP3

性能

- 读入 Excel： 约为 30 分钟
- 写入 WinCC： 约为 3.5 小时

说明

该实例中的值取决于其它因素，例如系统组态和 WinCC 项目组态。

参见

系统要求 (页 125)

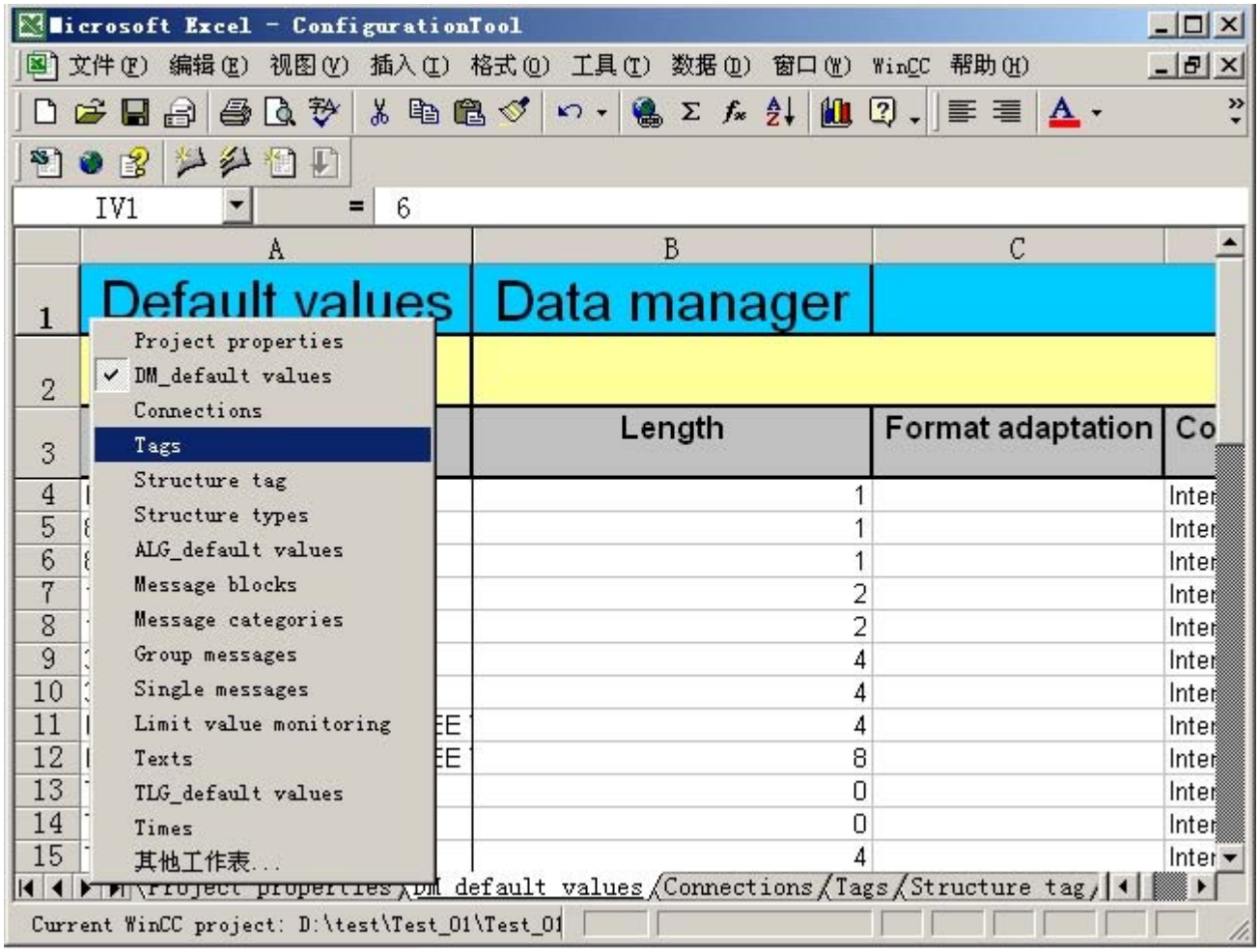
1.8.7.3 切换工作表

简介

对于一定数量的表单，有必要在表单之间进行切换。 Excel 为此提供了一个有用的工具。

步骤

使用可在窗口左下角访问的弹出式菜单在工作表之间切换。



1.8.7.4 行限制

简介

在 Excel 中组态期间，不应使用最大的可用行数，即 65536。如果 Excel 表单达到极限，性能会显著降低，因此应该限制所使用的行数。 组态工具提供将数据分布到多个工作表的选项。

组态工具中行限制的相关默认值为每个表单 16,000 行。 建议使用该行限制。

1.8.7.5 地址生成

简介

使用组态工具时，执行 Excel 提供的标准函数产生外部变量的地址。

下列实例说明如何生成带有“有符号 8 位数”数据类型的 SIMATIC S7 Protocol Suite 的地址字符串。

步骤

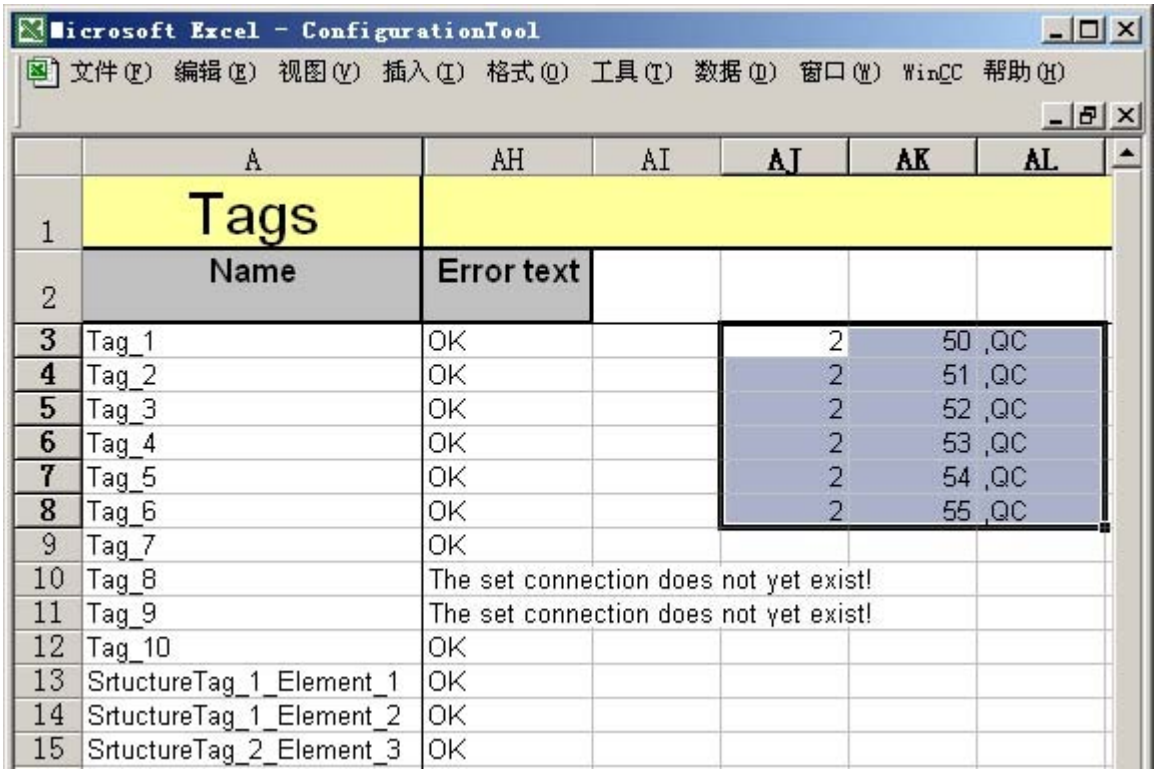
实例的地址串构成如下：

DB50、DBB0、QC

数据块编号 50 和字节数 0 是地址串的可编辑部分。 QC 字符串是可选项，用来定义质量代码。

1. 对于地址串中每个可修改的部分，在变量表的用户定义区域中定义一个列。 填充第一和第二行。

2. 突出显示头两个地址。 使用自动填充功能自动生成其余地址。



3. 在地址列中，输入下面显示的公式。 列标签 AJ3、AK3 和 AL3 取决于地址字符串组分在用户定义区域中的设置位置。

4. 单击第一个地址。 其余地址由自动填充功能自动产生。

1.8.7.6 VBA 宏

简介

由于组态工具中的数据在 Excel 文件夹中可用，所以其可通过 VBA 宏访问。 也就是说，例如，对象可由宏自动创建。

然而，VBA 宏仅应由富有经验的用户使用，因为使用组态工具的操作过程不同于普通 Excel 文件夹。

1.8.7.7 特殊字符

简介

Microsoft Excel 将以某些特殊字符开始，或其中包含特殊字符的文本解释为方程式。 使用组态工具时这可能引起问题。

这些特殊字符是“=”、“+”和“-”。 因此，这些字符不应用在对象名（组名称、变量名称、文本等）中。

1.8.7.8 Simatic S7 Protocol Suite 的地址串

简介

地址串不会在组态工具中自动产生。 地址串必须在组态工具中手动输入。 下面的表格列出 Simatic S7 ProtocolSuite 的所有地址串。

地址串

表格 1-1 二进制变量

数据区	寻址	地址串
位	位	DB1、D0.0
标记	位	M0.0
输入	位	E0.0
输出	位	A0.0

表格 1-2 无符号 8 位数/有符号 8 位数

数据区	寻址	地址串
DB	BYTE（字节）	DB1、DBB0
标记	BYTE（字节）	MB0
输入	BYTE（字节）	EB0
输出	BYTE（字节）	AB0

数据区	寻址	地址串
DB	字	DB1、DBW0
标记	字	MW0
输入	字	EW0
输出	字	AW0
计数器	字	Z0
定时器	字	T0
DB	双字	DB1、DD0
标记	双字	MD0
输入	双字	ED0
输出	双字	AD0

表格 1-3　　无符号 16 位数/有符号 16 位数

数据区	寻址	地址串
DB	BYTE（字节）	DB1、DBB0
标记	BYTE（字节）	MB0
输入	BYTE（字节）	EB0
输出	BYTE（字节）	AB0
DB	字	DB1、DBW0
标记	字	MW0
输入	字	EW0
输出	字	AW0
计数器	字	Z0
定时器	字	T0
DB	双字	DB1、DD0
标记	双字	MD0
输入	双字	ED0
输出	双字	AD0

表格 1-4　无符号 32 位数/有符号 32 位数

数据区	寻址	地址串
DB	BYTE（字节）	DB1、DBB0
标记	BYTE（字节）	MB0
输入	BYTE（字节）	EB0
输出	BYTE（字节）	AB0
DB	字	DB1、DBW0
标记	字	MW0
输入	字	EW0
输出	字	AW0
计数器	字	Z0
定时器	字	T0
DB	双字	DB1、DD0
标记	双字	MD0
输入	双字	ED0
输出	双字	AD0

表格 1-5　浮点数 32 位 IEEE 754/浮点数 64 位 IEEE 754

数据区	寻址	地址串
DB	BYTE（字节）	DB1、DBB0
标记	BYTE（字节）	MB0
输入	BYTE（字节）	EB0
输出	BYTE（字节）	AB0
DB	字	DB1、DBW0
标记	字	MW0
输入	字	EW0
输出	字	AW0
计数器	字	Z0
定时器	字	T0
DB	双字	DB1、DD0
标记	双字	MD0

数据区	寻址	地址串
输入	双字	ED0
输出	双字	AD0

表格 1-6　文本变量 8 位字符集/文本变量 16 位字符集

数据区	寻址	地址串
DB	BYTE（字节）	DB1、DBB0
标记	BYTE（字节）	MB0
输入	BYTE（字节）	EB0
输出	BYTE（字节）	AB0
DB	字	DB1、DBW0
标记	字	MW0
输入	字	EW0
输出	字	AW0
计数器	字	Z0
定时器	字	T0

表格 1-7　原始数据类型

数据区	寻址	地址串
DB	BYTE（字节）	RAW_BSEND（DB1、DBB0）
标记	BYTE（字节）	RAW_BSEND(MB0)
输入	BYTE（字节）	RAW_BSEND(EB0)
输出	BYTE（字节）	RAW_BSEND(AB0)
DB	字	RAW_BSEND(DB1,DBW0)
标记	字	RAW_BSEND(MW0)
输入	字	RAW_BSEND(EW0)
输出	字	RAW_BSEND(AW0)
DB	双字	RAW_BSEND(DB1、DD0)
标记	双字	RAW_BSEND(MD0)
输入	双字	RAW_BSEND(ED0)
输出	双字	RAW_BSEND(AD0)

1.8.7.9 数据包

简介

组态工具也可用于对从数据包装载的变量进行操作。 为此，在 WinCC 项目管理器中创建新 WinCC 项目（多用户项目），并装载所需要的数据包。 用组态工具读取此 WinCC 项目。数据包中提供的变量可以通过变量对话框进行选择。

创建新的 WinCC 项目

新 WinCC 项目可从使用的数据包中的项目文件夹创建。 使用的数据包，然而，必须在创建之后装载到新 WinCC 项目（多用户项目）中。

1.9 WinCC 归档组态工具

1.9 资源

1.9.1 引言

简介

WinCC 归档组态工具是 Excel 加载项。该工具使得用户可以简单、高效地组态批量数据，以用于归档 WinCC 过程值。

它允许处理变量记录编辑器不充分支持的数量结构。

使用 WinCC 归档组态工具

WinCC 归档组态工具提供了创建新归档的可能性。此外，可以读出已存在的归档并进行编辑。如果必须进行大量的修改，建议删除已存在的归档并重新创建一个完整的归档。此外，可以有选择地将一些小的修改重写到 WinCC 项目中。

归档在 Excel 工作表中进行组态。Excel 工作表由一个归档表和一个或多个归档变量表组成。归档表用于组态归档的常规属性。归档变量表用于组态归档变量及其属性。

WinCC 归档不能完全替代变量记录编辑器。定时器组态和归档组态（高速和低速变量记录）仍然必须在变量记录编辑器中完成。

输入期间不检查数据。然而，对整个归档的检查包括多个方面。这些方面包括参数设置的大体正确性、所用名称的唯一性以及所定义对象（变量、时间等）是否存在。

比较

	变量记录编辑器	WinCC 组态工具	WinCC 归档组态工具
用途	归档系统的标准组态接口	整个 WinCC 项目的易于使用的规划接口。	用于归档极大数量结构的高性能组态接口。
组态归档系统	归档系统完全可组态。	除了归档组态（高速和低速变量记录）之外，归档系统完全可组态。	可以分别组态一个归档。
数量结构建议	最多执行大约 100 - 200 个归档变量。	最多执行大约 10,000 - 30,000 个归档变量。	执行大约 10,000 个以上归档变量。
数据输入	通过属性对话框和表格界面进行数据输入，使用有所限制。	以表格形式进行数据输入，具有全面的支持（默认值、组合框、对话框和 Office 助手）。	以表格形式进行数据输入，具有有限的支持。
数据检查	输入期间检查数据。	输入期间检查数据。	写入 WinCC 项目之前先检查数据。也可以禁用各种检查。
可扩充性	无。	可以通过 VBA 宏进行扩充。	可以通过 VBA 宏进行扩充。

参见

快速入门 (页 250)

系统要求 (页 249)

1.9.2 系统要求

系统要求

- Windows XP SP3/Windows Server 2003 (R2) SP2/Windows Vista SP1
- Microsoft Excel XP、Office 2003、Office 2007。必须安装 Visual Basic for Applications。
- Internet Explorer 6 或 7

说明

使用 Excel 时需要 Office 助手，以便输出组态工具的警告消息。

不使用 WinCC

可以在不安装 WinCC 的情况下使用 WinCC 归档。 然而，在这种情况下，不能从 WinCC 读取数据或将数据写至 WinCC。 使用 Excel 的全部组态选项都保持可用。

1.9.3 安装归档组态工具

WinCC 归档组态工具可以通过两种不同的方法安装。

步骤

1. WinCC 安装期间，从“程序”对话框选择“完整的 WinCC V7.0”。
   WinCC 将与智能工具、WinCC 组态工具和 WinCC 归档组态工具一起安装。

选择“SIMATIC”>“WinCC”>“工具”启动 WinCC 归档组态工具。

可选步骤

也可以从 WinCC DVD 安装 WinCC 归档组态工具。

1. 切换到 WinCC DVD 目录“InstData\WinCC\setup\Products\WinCCArchive”。
2. 双击 setup.exe。
3. 按照对话框中的指示进行操作。
   WinCC 归档组态工具安装完成。

1.9.4 快速入门

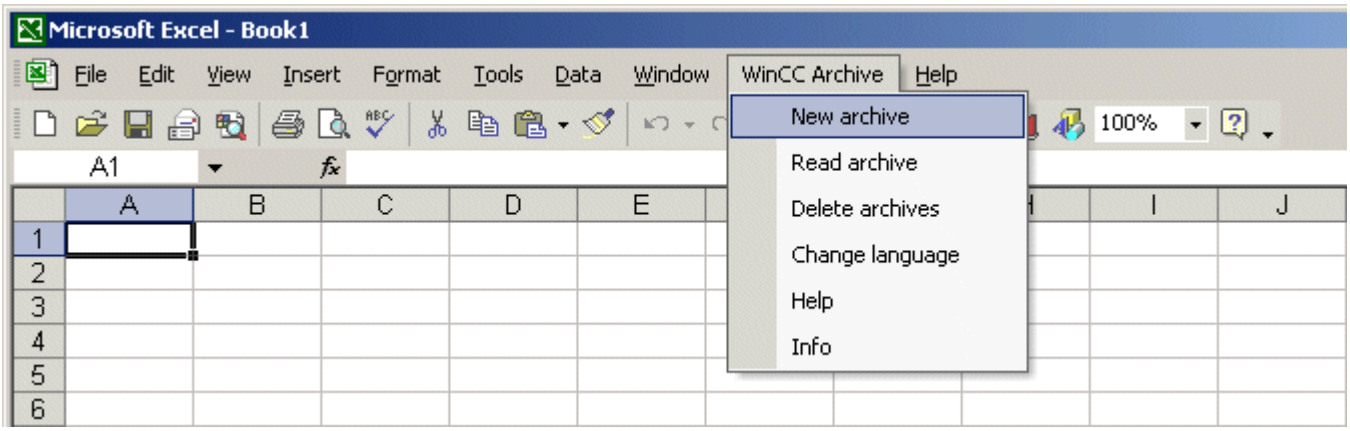
引言

本说明使得用户可以对如何使用 Excel 插件 WinCC 归档快速入门。 简要介绍了一些可用的最重要功能。

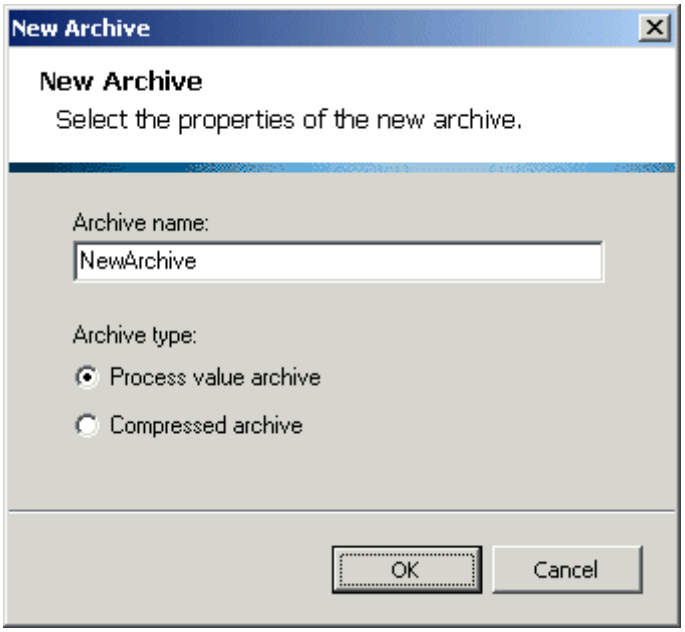
新建归档

步骤

1. 打开 Excel。 Excel 插件 WinCC 归档同时自动启动。 Excel 菜单包含“WinCC 归档”条目。 从该菜单条目中选择“新建归档”。

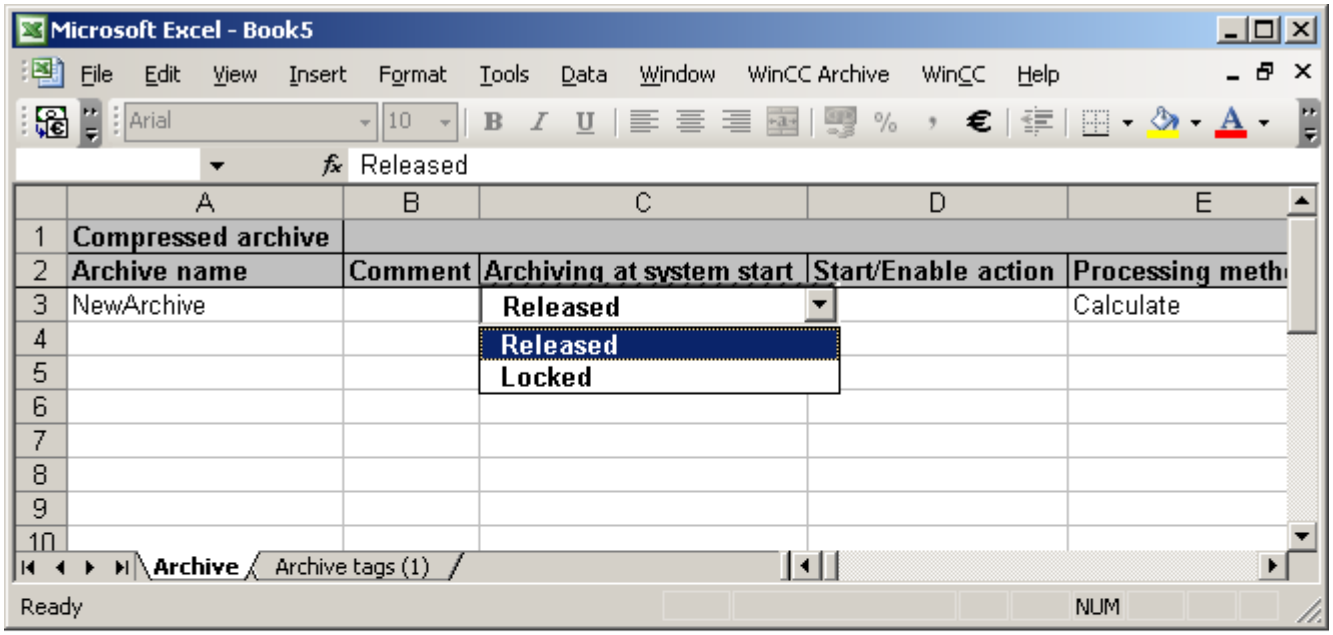


2. 对话框打开，在其中定义要创建的归档的基本属性。 为归档输入一个名称。 选择所需要的归档类型。 本说明介绍过程值归档的创建。 单击“确定”按钮。

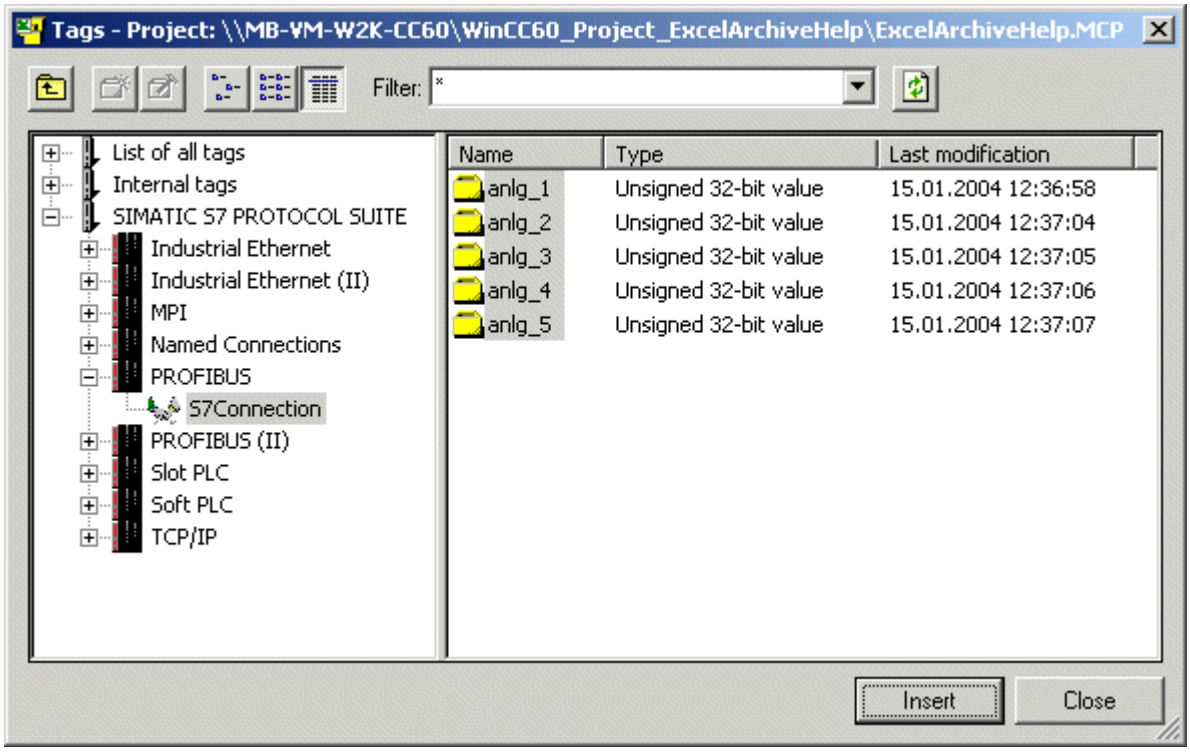


3. 创建了一个新的 Excel 表，用于组态新的归档。 新的 Excel 表包含下列表格列表：
   - “归档”： 定义归档属性的表格。
   - “归档变量 (1)”： 组态二进制和模拟归档变量的表格。
   - “归档变量 (2)”： 组态过程控制的归档变量的表格。

4. 在“归档”表格中定义归档属性。 可以根据需要编辑所有单元格。 如果有多个参数，可以使用组合框来简化输入。 例如，双击单元格来输入“系统启动时归档”参数。 显示包含该参数可能值的组合框。

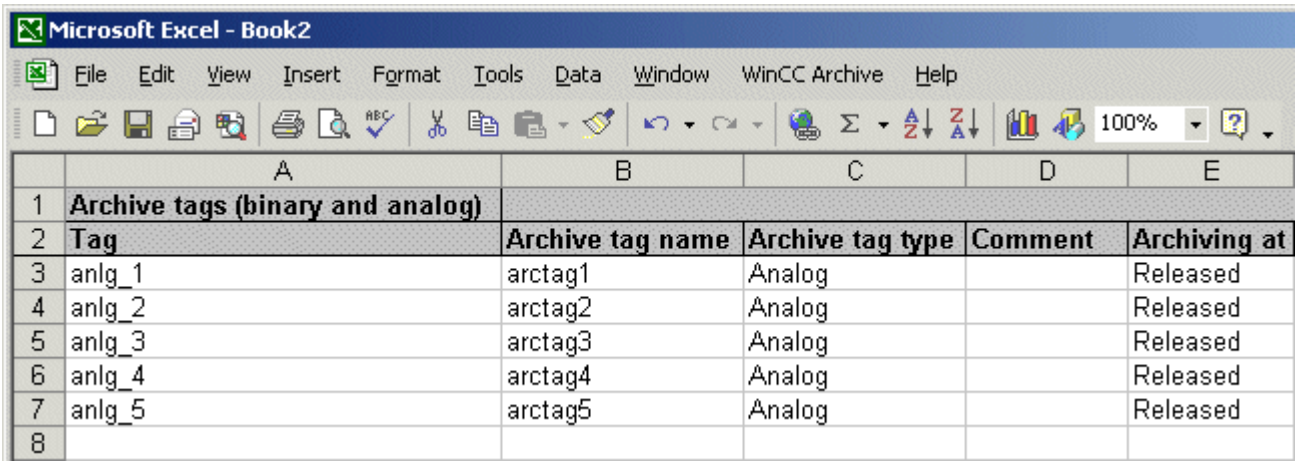


5. 切换至“归档变量 (1)”表格。 在该表格中最多可以组态 65634 个二进制和模拟归档变量。 可以根据需要编辑所有单元格。 如果有多个参数，可以使用组合框来简化输入。 如果 WinCC 项目已经打开，可以进一步打开一个对话框，在其中选择 WinCC 变量。 通过双击“变量”列的数据区来打开该对话框。



6. 用于选择 WinCC 变量的对话框包含多项选择功能。 当对话框打开时，仍然可以操作 Excel 表格。 从对话框中选择所需的变量。 在 Excel 表格中选择将插入所选变量名称的单元格。 单击对话框中的“粘贴”按钮。 对话框保持打开。 所选变量的名称被插入到 Excel 表格中。 单击“关闭”关闭对话框。

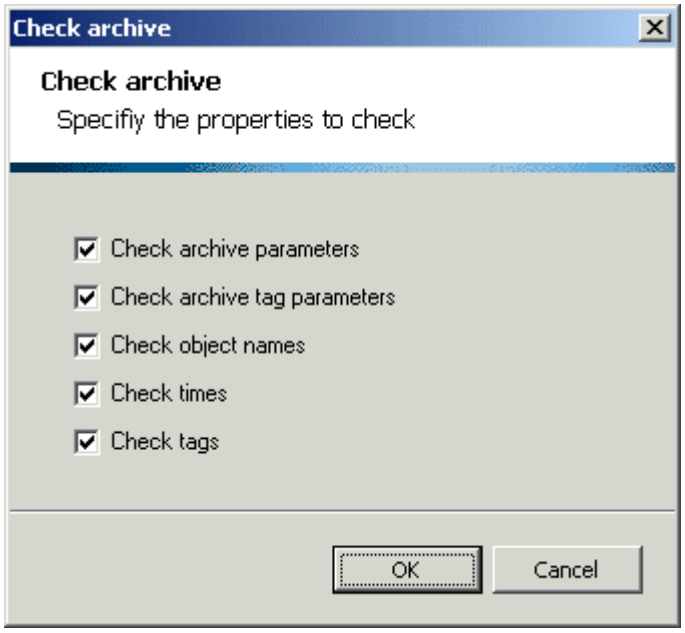
7. 在“归档变量名称”列中指定新归档变量的名称。 这些名称在归档中必须唯一。 在“归档变量类型”列中定义新归档变量的正确类型。 可使用组合框输入该参数。 定义新归档变量剩下的属性。



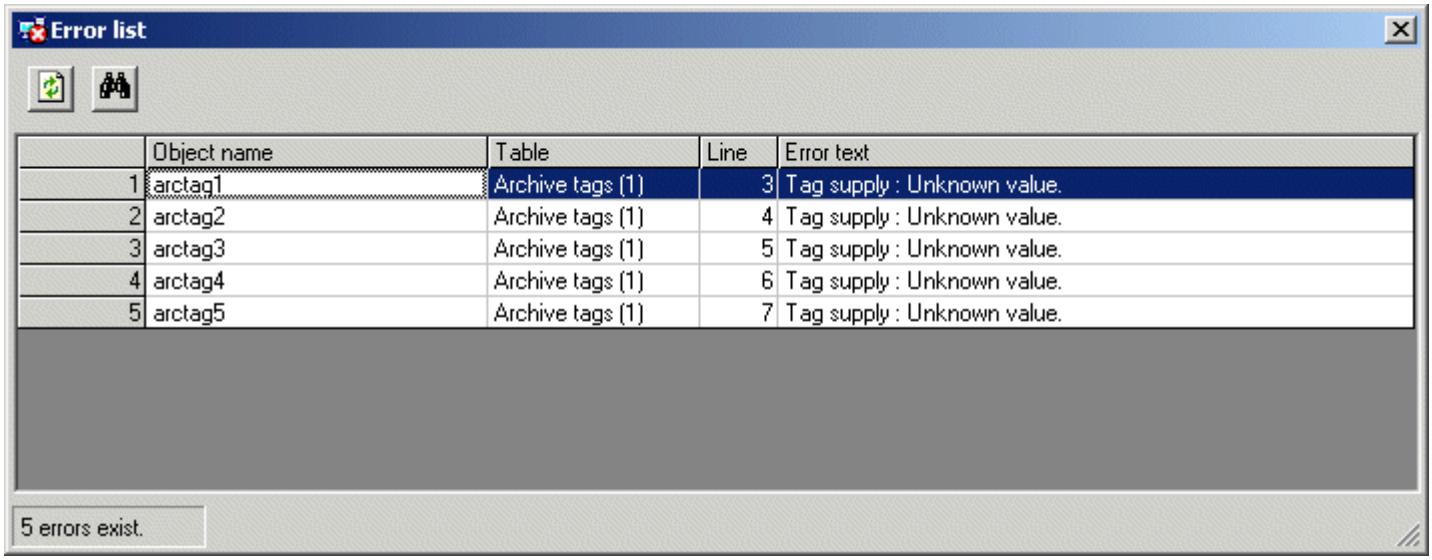
	A	B	C	D	E
1	**Archive tags (binary and analog)**				
2	**Tag**	**Archive tag name**	**Archive tag type**	**Comment**	**Archiving at**
3	anlg_1	arctag1	Analog		Released
4	anlg_2	arctag2	Analog		Released
5	anlg_3	arctag3	Analog		Released
6	anlg_4	arctag4	Analog		Released
7	anlg_5	arctag5	Analog		Released
8					

8. 在 Excel 表格中输入数据时不进行任何检查。 选择菜单条目“WinCC 归档”>“检查归档”。 打开一个对话框以检查整个归档。 可以限制要检查的归档属性。 激活所有相关的复选框并单击“确定”按钮。

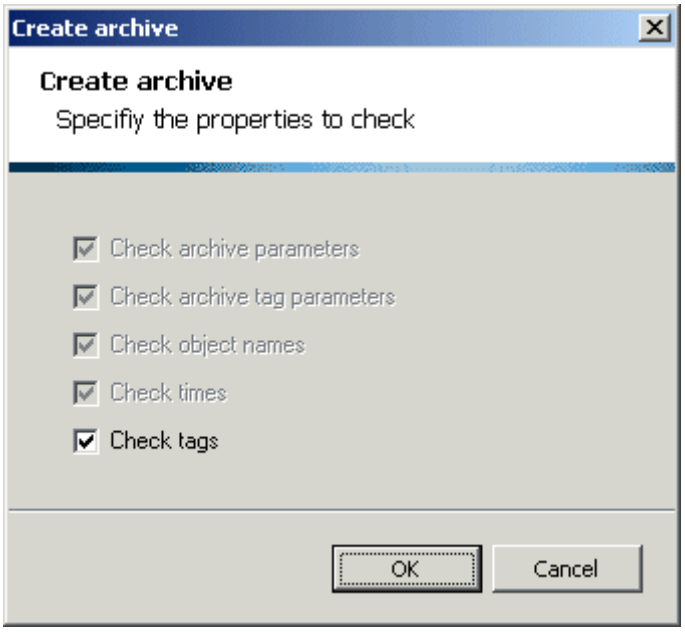
   注意： 如果尚未在变量记录编辑器中为 WinCC 项目进行任何组态，则必需的周期时间仍然不可用。 它们显示在变量记录编辑器中，但是仅在组态数据发生改变后才存储到数据库中。 要触发变量记录编辑器来存储周期时间，只需将周期时间“500 ms”重命名为“_500 ms”并保存项目。 当然，随后您还可以将周期时间名重置为原来的值。



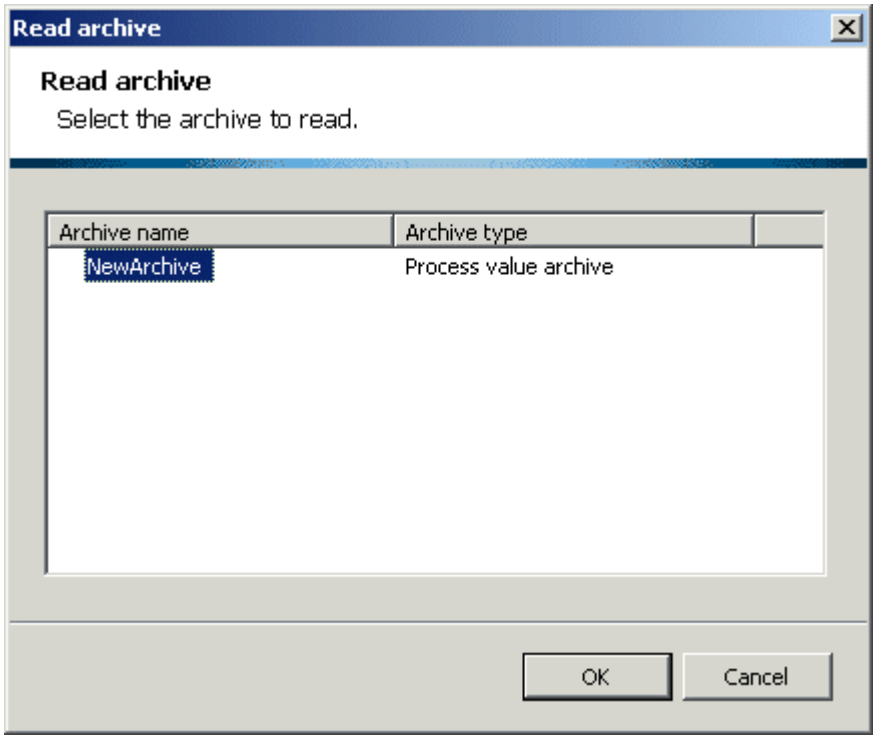
9. 显示测试结果。 如果发现错误，会提示用户是否应显示错误列表。 单击“是”来确认提示。 打开一个对话框，包含错误对象的列表和相应的错误原因。 通过双击列表中的一个条目，系统直接跳至 Excel 表格中的错误对象，错误被清除。 相应对象的“错误文本”列也包含错误原因。 错误原因具有下列结构：“列： 错误”。 清除所有发现的错误并再次执行测试。



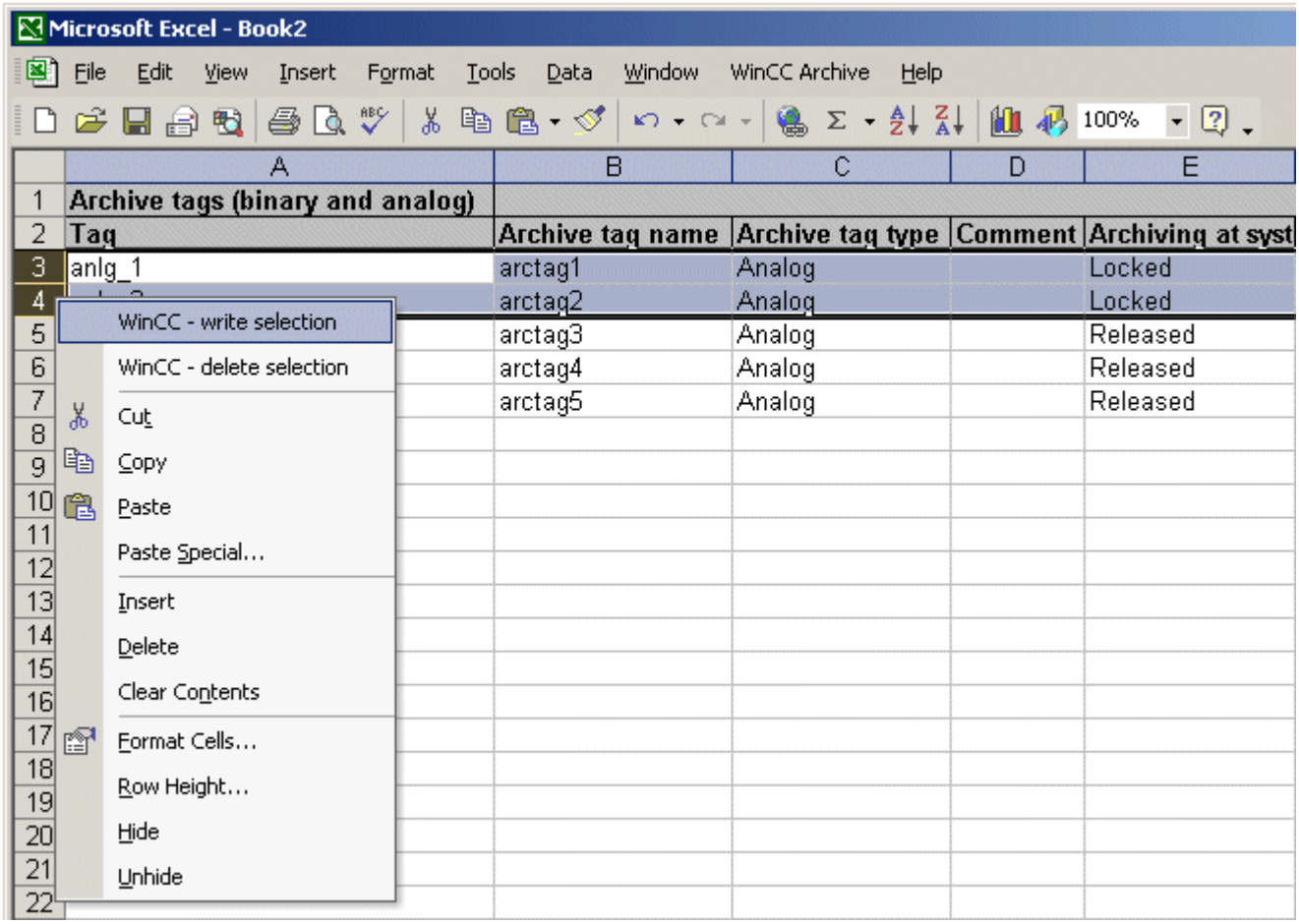
10. 选择菜单条目“WinCC 归档”->“创建归档”。 打开一个对话框以创建整个归档。 单击“确定”按钮。 如果已经存在与要创建归档相同名称的归档，必须将其删除。 因此，显示相应的提示。 通过单击“确定”来确认提示。 在 WinCC 项目中创建了归档。 关闭 Excel 表。



11. 无法在变量记录编辑器中打开新创建的归档。 编辑归档的唯一方法是将其读回 Excel 表格中。 选择菜单条目“WinCC 归档”->“读归档”。 打开一个对话框，其中包含 WinCC 项目中所有可用归档的列表。 选择要读取的归档并单击“确定”按钮。 创建了一个新的 Excel 表。 数据被从所选归档读入该表中。



12. 然后，可以修改归档的属性或单个归档变量的属性。 可以有选择地将所作的修改传送至 WinCC 项目。 选择已修改的相关对象。 通过单击鼠标右键来打开上下文菜单并选择菜单条目“WinCC - 写选择”。 写归档变量之前，打开一个对话框，在其中定义测试标准。 通过单击“确定”来确认对话框。



1.9.5 操作 WinCC 归档

1.9.5.1 操作 WinCC 归档

引言

本节提供操作 WinCC 归档的所有方面的详细说明。

参见

创建、修改和删除 (页 293)
检查归档数据 (页 288)
组态压缩归档 (页 280)
组态过程值归档 (页 264)
创建归档文件夹 (页 258)

1.9.5.2 创建归档文件夹

创建归档文件夹

引言

本节描述创建和操作归档文件夹的常规步骤。 归档文件夹与通过 WinCC 归档创建的 Excel 表有关。 归档文件夹用于组态归档和相关归档变量的属性。

参见

如何添加新表格 (页 262)
读取已存在的归档 (页 260)
创建新的归档文件夹 (页 258)

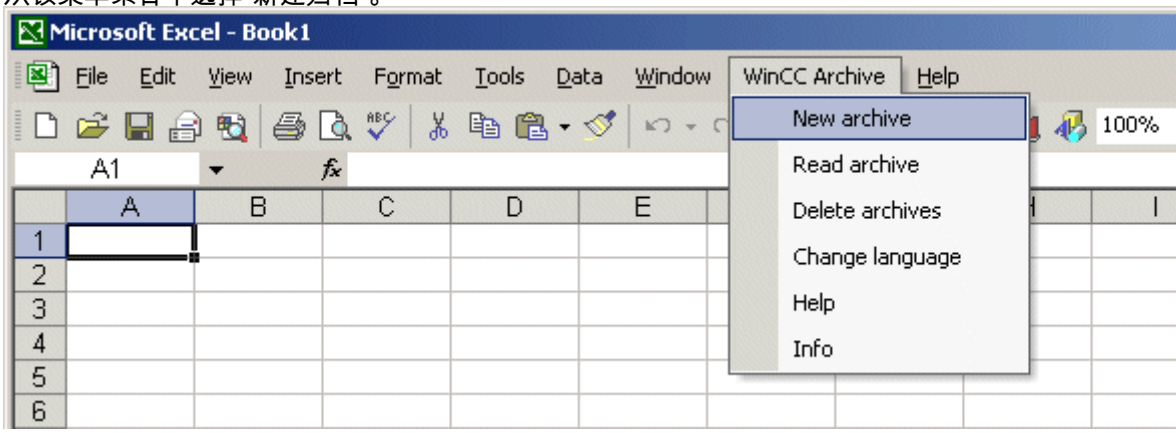
创建新的归档文件夹

引言

本说明阐述了创建新归档文件夹的步骤。

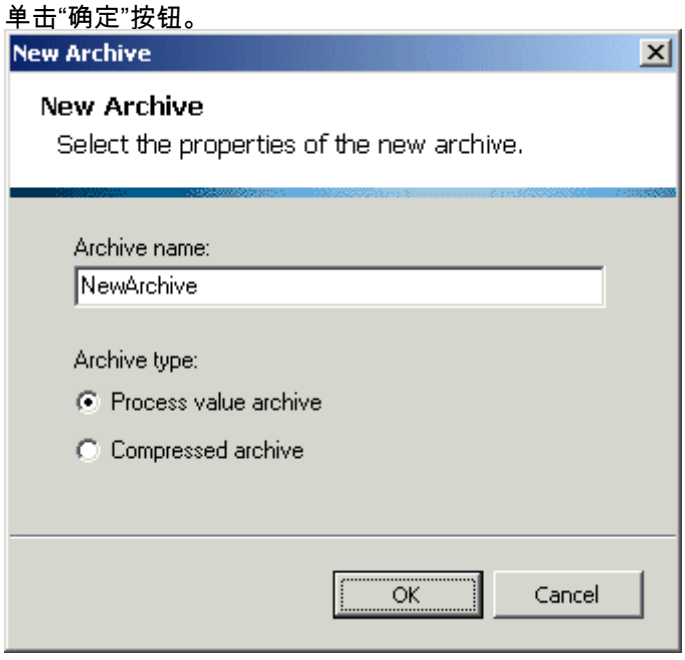
步骤

1. 打开 Excel。 Excel 插件 WinCC 归档同时自动启动。 Excel 菜单包含“WinCC 归档”条目。 从该菜单条目中选择“新建归档”。



2. 对话框打开，在其中定义要创建的归档的基本属性。 为归档输入一个名称。 选择所需要的归档类型。
使用 WinCC 归档可以组态两种不同类型的归档。 这就是过程值归档和压缩归档。创建的归档文件夹取决于所选的归档类型。 根据所包含表格的结构，过程值归档和压缩归档的归档文件夹有所不同。

3. 单击“确定”按钮。



4. 创建了一个新的 Excel 表，用于组态所需的归档类型。
默认状态下，归档文件夹包含一个名称为“归档”的表格。 在其中定义归档属性。 归档属性的定义位于表格数据区（第三行）的第一行。
归档文件夹至少包含一个名为“归档变量”的表格和一个连续编号。 这些表格用于组态与归档相关的归档变量。 表格的整个数据区（从第三行开始）可用于组态归档变量。 可以将更多归档变量表添加到归档文件夹。

说明

Microsoft Excel 将以某些特殊字符开头的文本或者包含某些特殊字符的文本解释为方程式。使用组态工具时这可能引起问题。

这些特殊字符是“=”、“+”和“-”。 因此，这些字符不应用作对象名称（组名称、变量名称、文本等）的一部分。

参见

如何添加新表格 (页 262)

读取已存在的归档 (页 260)

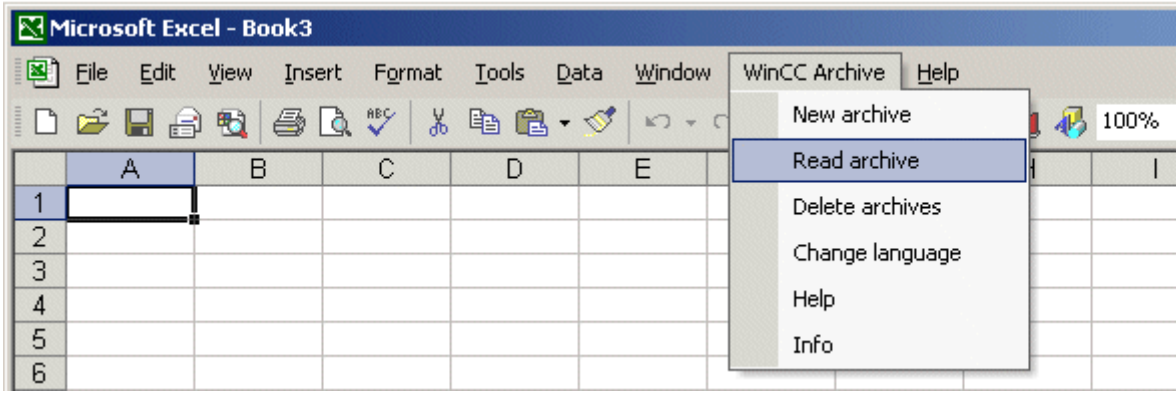
读取已存在的归档

引言

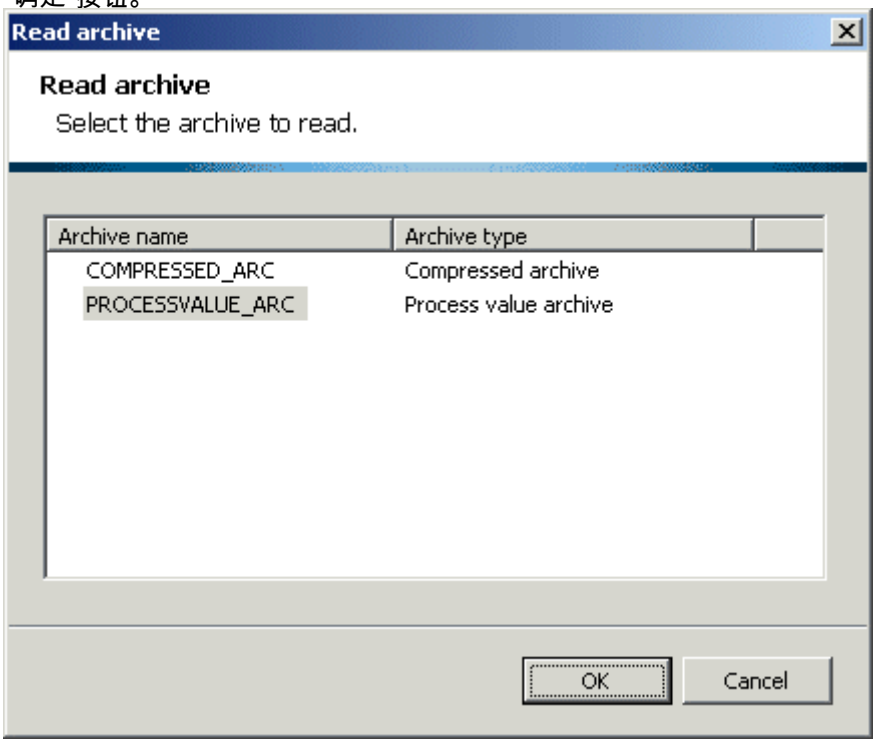
本说明阐述了从 WinCC 项目中读取归档的步骤。

步骤

1. 打开 Excel。 Excel 插件 WinCC 归档同时自动启动。 Excel 菜单包含“WinCC 归档”条目。 从该菜单条目中选择“读归档”。



2. 打开一个对话框，其中包含 WinCC 项目中所有可用归档的列表。 选择要读取的归档并单击“确定”按钮。



3. 创建一个新的归档文件夹。 数据被从所选归档读入该表中。 要创建的归档文件夹的类型取决于正在读取的归档的类型。 WinCC 归档可用于编辑过程值归档和压缩归档。
归档文件夹包含一个名称为“归档”的表格。 这包含归档的属性。 归档文件夹还至少包含一个名为“归档变量”的表格和一个连续编号。 这些表格包含与归档相关的归档变量。

参见

如何添加新表格 (页 262)
创建新的归档文件夹 (页 258)

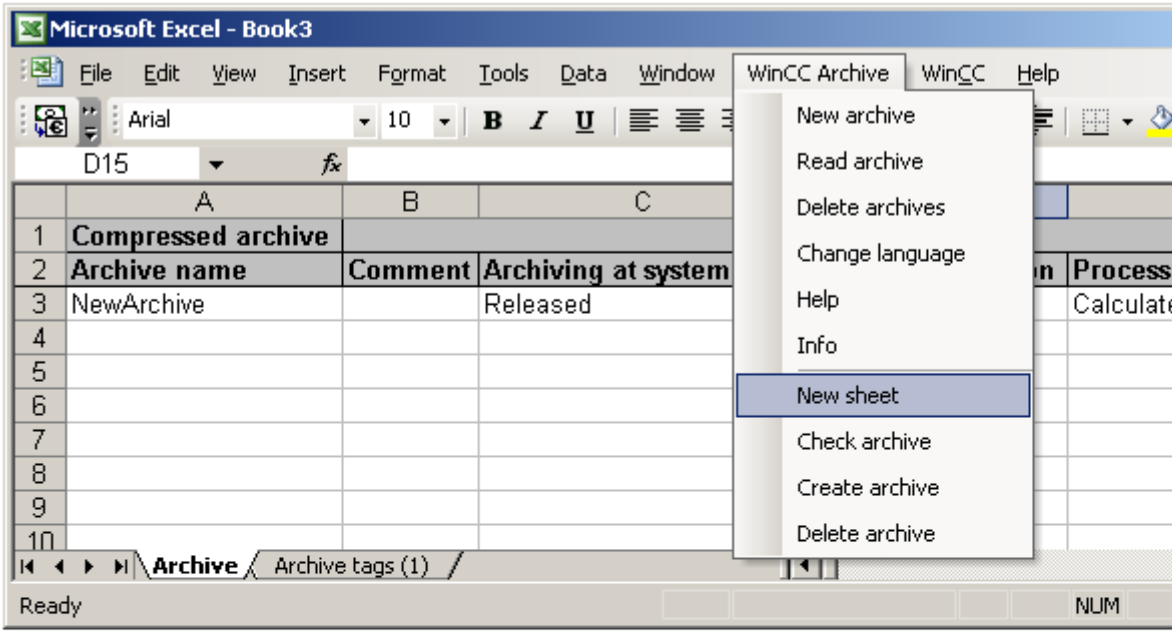
如何添加新表格

引言

本说明阐述了将新表格添加到归档文件夹的步骤。

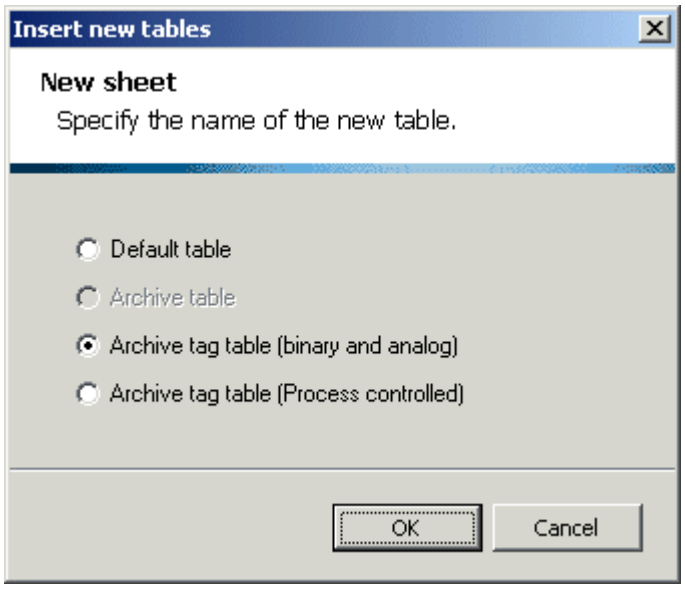
步骤

1. 打开要向其添加新表格的归档文件夹。 选择菜单条目“WinCC 归档”>“新表格”。



2. 打开一个对话框，可以在其中选择新表格的类型。 下列表格类型可用。
   - 标准表格： 标准 Excel 表格。
   - 归档表格： 定义归档属性的表格。 只有在删除已存在的归档表格后，才能选择这种类型的表格。
   - 归档变量表： 用于组态归档变量的表格。 如果是过程值归档，有两种不同类型的归档变量表可用于选择。 如果是压缩归档，只有一种类型可用于选择。

   选择所需要的类型并单击“确定”按钮。



3. 新表已添加至归档文件夹之中。 如有需要，以后可以为其重命名。

参见

读取已存在的归档 (页 260)

创建新的归档文件夹 (页 258)

1.9.5.3 组态过程值归档

组态过程值归档

引言

本节描述使用 WinCC 归档组态过程值归档的常规步骤。

参见

使用变量选择对话框 (页 279)

组态归档变量参数（过程控制）(页 275)

组态归档变量参数（二进制和模拟）(页 268)

组态归档参数 (页 265)

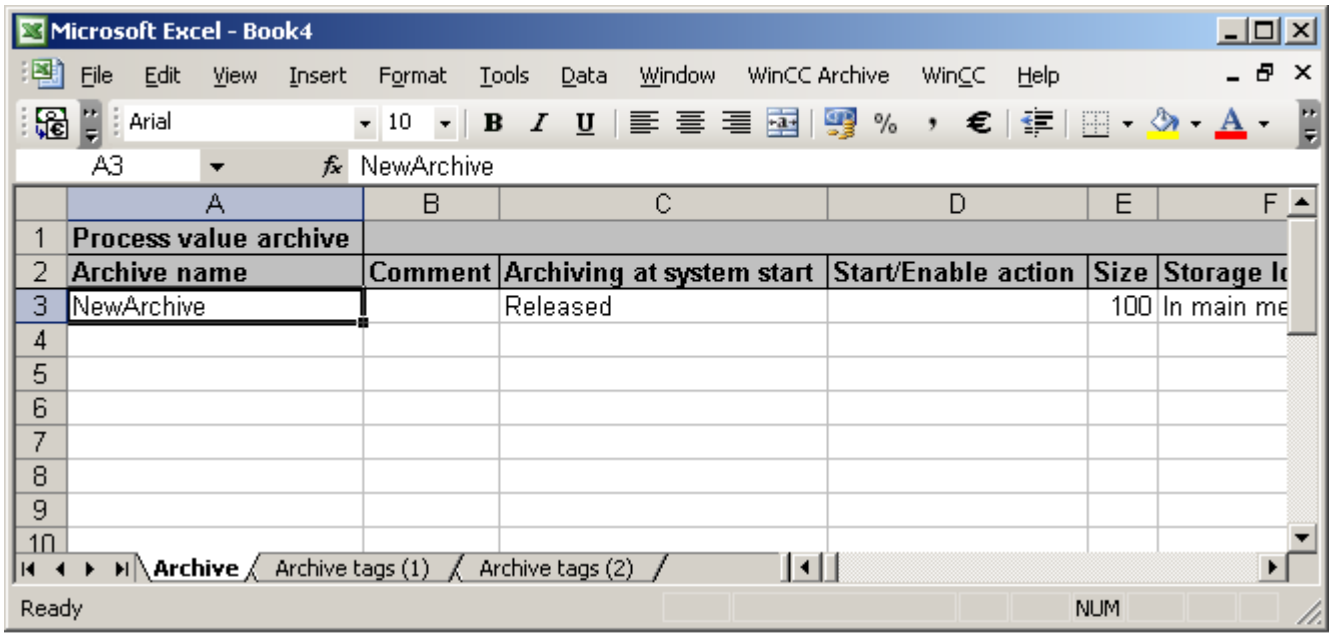
组态归档参数

简介

本节描述归档参数的组态。各个归档参数将单独说明。

步骤

1. 激活归档表格。其名称通常为“归档”。可以改变表格的名称。



2. 必须在归档表格数据区的第一行中定义归档参数。只有来自该行的内容才会被 WinCC 归档计算。

归档参数

下表列出了必须为过程值归档输入的所有参数。

	列	列表	参数	描述
1	A		归档名称	此处指定归档的名称。
2	B		注释	此处输入归档的注释。
3	C		在系统启动时进行归档	定义系统启动后应“启用”还是“禁用”归档。
4	D		启动/启用动作	指定开始归档时必须执行的操作。
5	E		大小	指定要存储在归档中的数据记录的数目。该参数只与主存储器中的归档相关。
6	F		存储位置	定义归档的存储位置。可在“硬盘”或“主存储器中”之间选择。

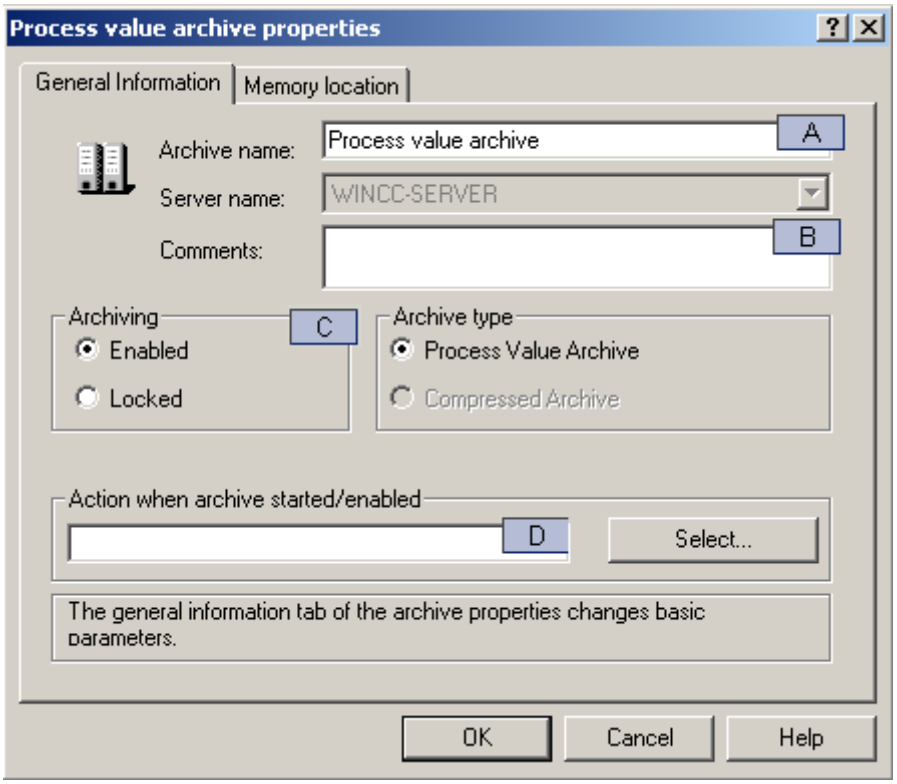
以下部分说明“列表”列中所使用符号的意义。

双击之后打开一个组合框。

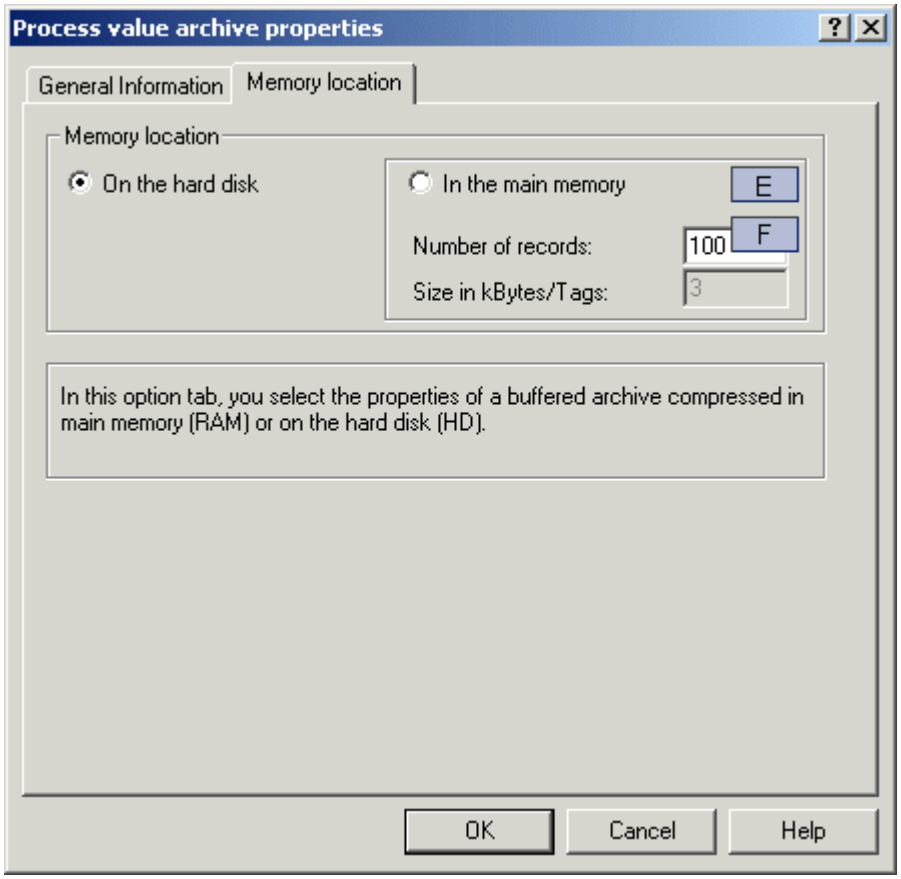
变量记录编辑器

下图说明使用变量记录编辑器进行组态和使用 WinCC 归档进行组态之间的关系。除了要在变量记录编辑器中设置的参数之外，还要指定 WinCC 归档中的各个列。

1. 下图显示了“过程值归档属性”对话框。“常规”选项卡已被选中。对于每个参数，将显示 WinCC 归档表格结构中的对应列。



2. 下图显示了“过程值归档属性”对话框。“存储位置”选项卡已被选中。对于每个参数，将显示 WinCC 归档表格结构中的对应列。



参见

组态归档变量参数（过程控制）(页 275)

组态归档变量参数（二进制和模拟）(页 268)

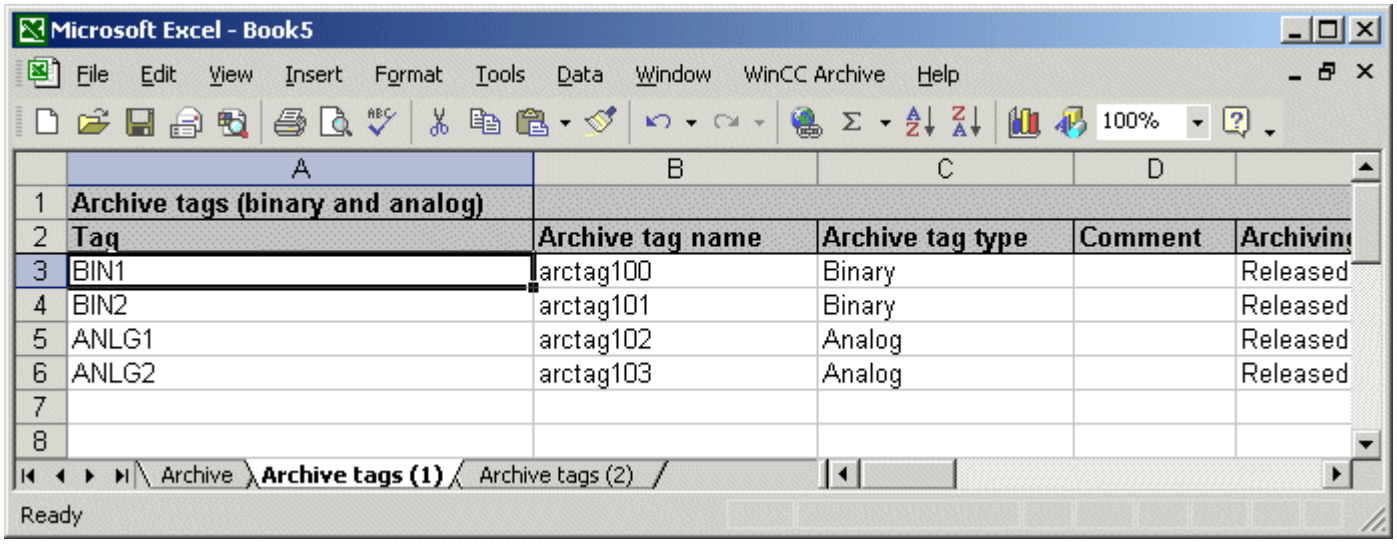
组态归档变量参数（二进制和模拟）

简介

本说明阐述二进制和模拟归档变量的组态。必须为二进制和模拟归档变量定义的各个参数将单独进行说明。

步骤

1. 激活用于组态二进制和模拟归档变量的表格。其名称通常为“归档变量 (1)”。可以改变表格的名称。因为二进制和模拟归档变量的参数十分相似，所以在相同的表格结构中组态。



2. 表格的整个数据区可用于组态归档变量。这样，在表格上最多可以组态 65534 个归档变量。如果这样不够，可以添加更多表格。

归档变量参数

下表列出了必须为二进制和模拟归档变量定义的所有参数。

	列	列表	参数	描述
1	A		变量	这用于输入要归档的变量的名称。
2	B		归档变量名称	这用于指定归档变量的名称。这在归档中必须唯一。
3	C		归档变量类型	这用于指定归档变量的类型。必须根据要保存的变量的类型进行定义，可以是“模拟”或“二进制”。
4	D		注释	这用于输入归档变量的注释。
5	E		在系统启动时进行归档	这用于定义系统启用后应“启用”还是“禁用”归档。
6	F		提供变量	定义提供变量的类型。选择“系统”或“手动输入”。
7	G		采集类型	这用于定义归档变量采集的类型。可以选择“非周期”、“周期连续”、“周期选择”或“根据变化”。

	列	列表	参数	描述
8	H		采集周期	这用于指定请求变量值应采用的周期。
9	I		归档周期	这用于指定归档变量值应采用的周期。
10	J		归档周期系数	这可用于指定归档周期的系数。对变量值归档应采用的周期得自“归档周期”和“归档周期系数”的乘积。
11	K		同时写入变量	可以指定应写入归档值的变量的名称。
12	L		编辑	这用于指定应如何处理编译后的变量值。可以选择“实际值”、“平均值”、“总和”、“最小值”、“最大值”和“动作”。
13	M		处理的动作	这可用于指定要用于处理变量值的动作。为此，启用“处理”列中的“动作”选项。
14	N		单位	这用于指定归档变量值的单位。
15	O		更改时归档	这用于指定只在变量值改变时进行归档。
16	P		滞后	这用于在已经激活“更改时归档”时定义滞后类型。可在“绝对值”和“百分比”之间进行选择。
17	Q		滞后值	这用于在已经激活“更改时归档”时定义滞后的数字值。
18	R		归档条件	这用于指定应何时归档二进制归档变量。可以设置为“每次信号改变”、“始终”、“信号从 0 变为 1”和“信号从 1 变为 0”。
19	S		出错时保存	这用于定义发生错误时应保存哪个值。在“最后值”和“替换值”之间进行选择。
20	T		先行值数目	这可用于指定归档开始前必须计算在内的数值个数。
21	U		后续值数目	这可用于指定归档时刻之后必须计算在内的数值个数。
22	V		显示限制，下限	这可用于指定归档变量数值范围的下限。
23	W		显示限制，上限	这可用于指定归档变量数值范围的上限。

	列	列表	参数	描述
24	X		启动事件	这可用于将一个函数指定为启动归档的动作。
25	Y		开始变量	这可用于将一个二进制变量指定为启动归档的动作。
26	Z		停止事件	这可用于将一个函数指定为停止归档的动作。
27	AA		停止变量	这可用于将一个二进制变量指定为停止归档的动作。
28	AB		长期关联	可在此处定义是否将归档变量写入到中央归档服务器（WinCC CAS）的归档之中。
29	AC		段更改后归档	可以在此确定段更改时是否归档非周期性归档的变量。
30	AD		状态	可以在此显示归档的状态。
31	AE		错误文本	可以在此定义错误文本。

以下部分说明“列表”列中所使用符号的意义。

双击之后打开一个组合框。

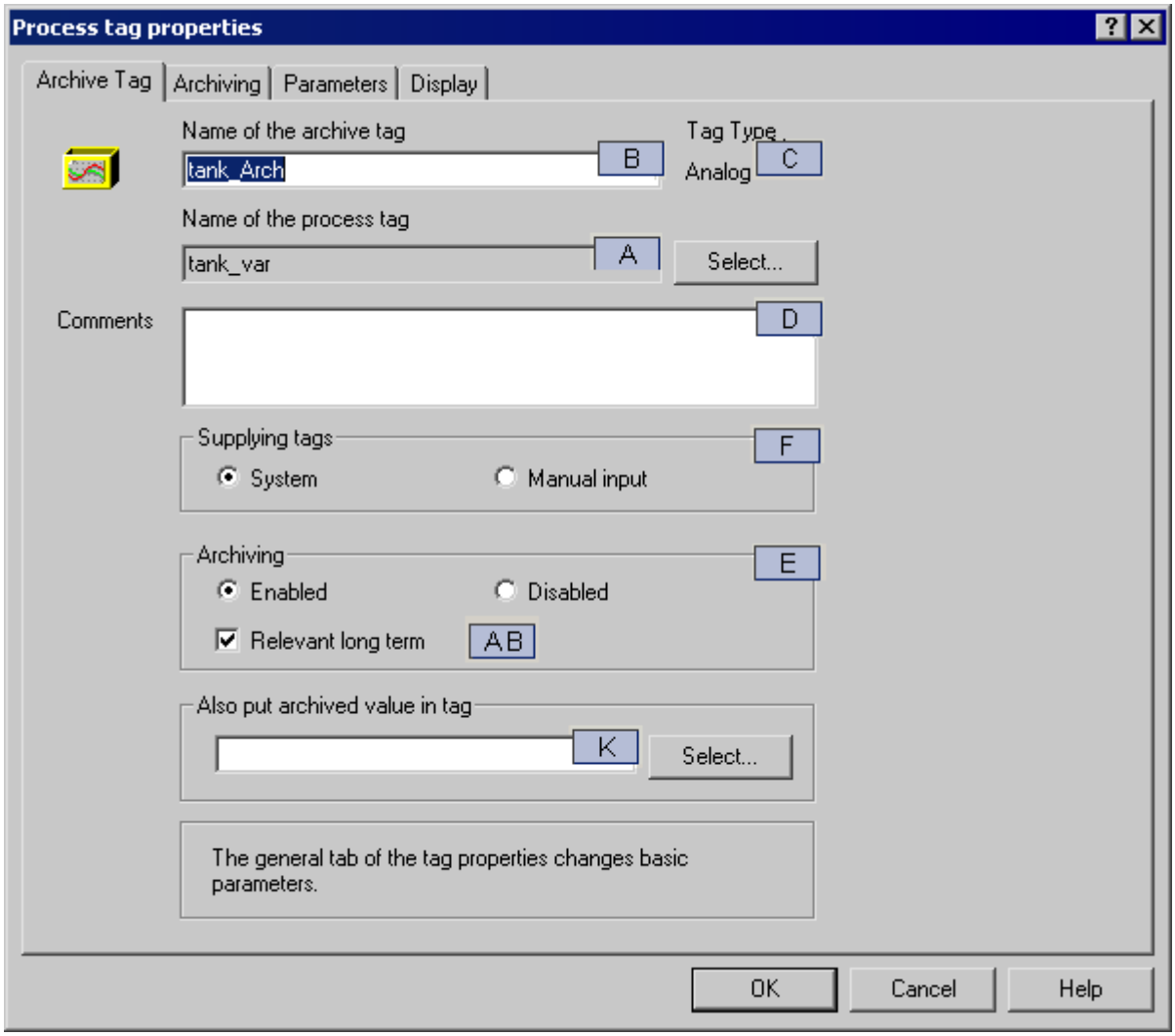
双击之后打开一个组合框。必须打开一个 WinCC 项目。

双击之后打开变量对话框。必须打开一个 WinCC 项目。

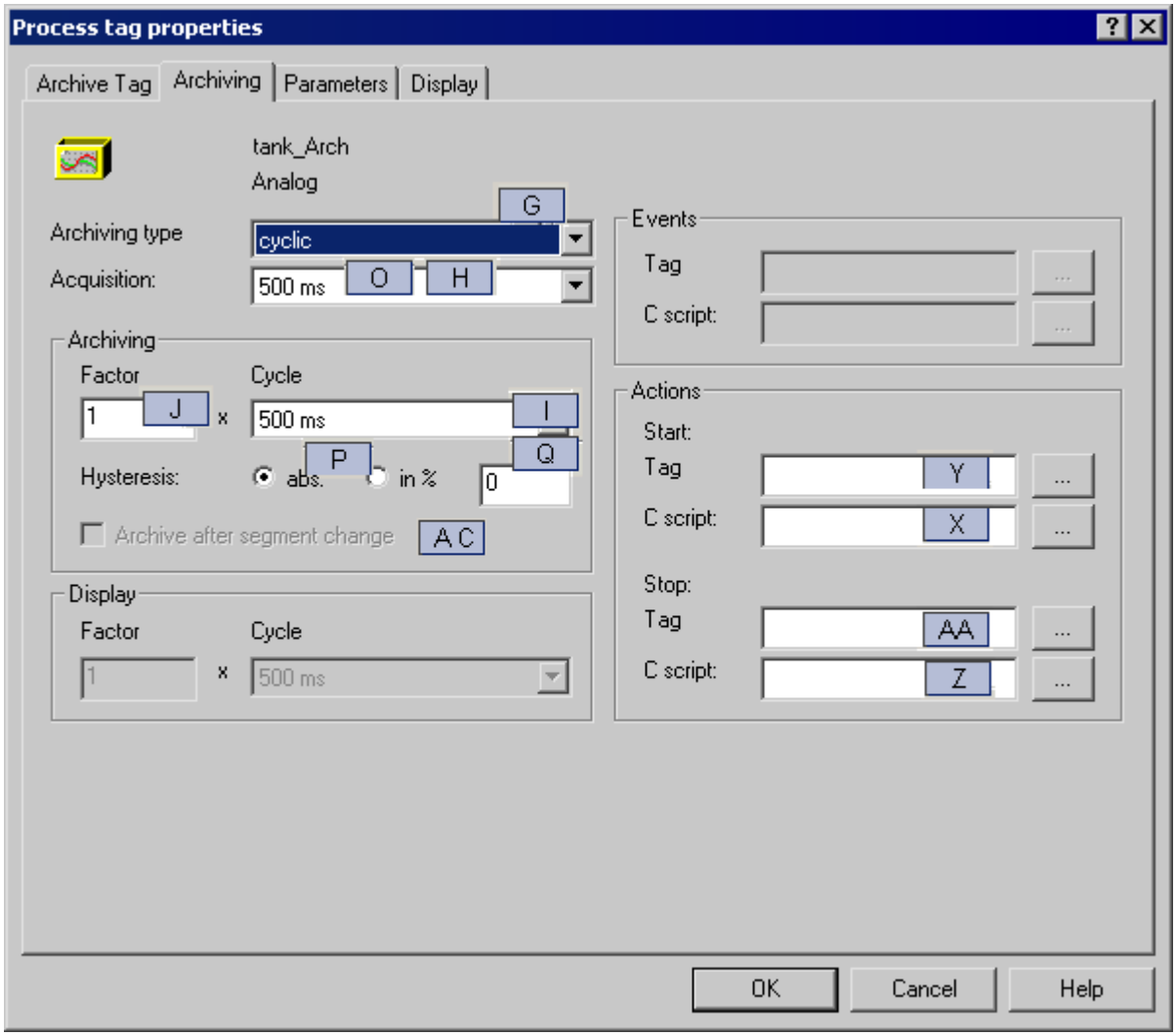
变量记录编辑器

下图说明使用变量记录编辑器进行组态和使用 WinCC 归档进行组态之间的关系。除了要在变量记录编辑器中设置的参数之外，还要指定 WinCC 归档中的各个列。

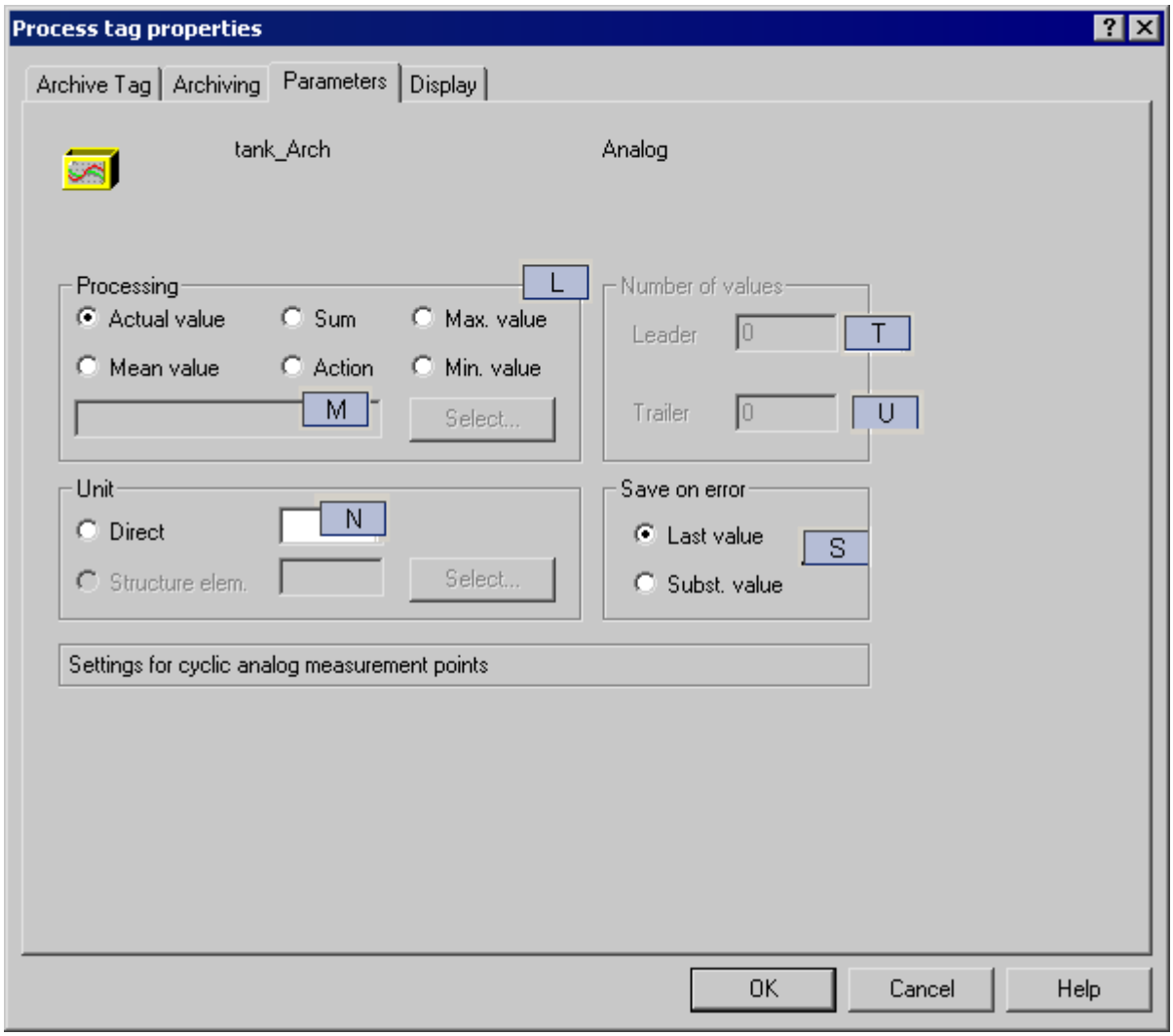
1. 下图显示了“过程变量属性”对话框。已经选择了“归档变量”选项卡。对于每个参数，将显示 WinCC 归档表格结构中的对应列。



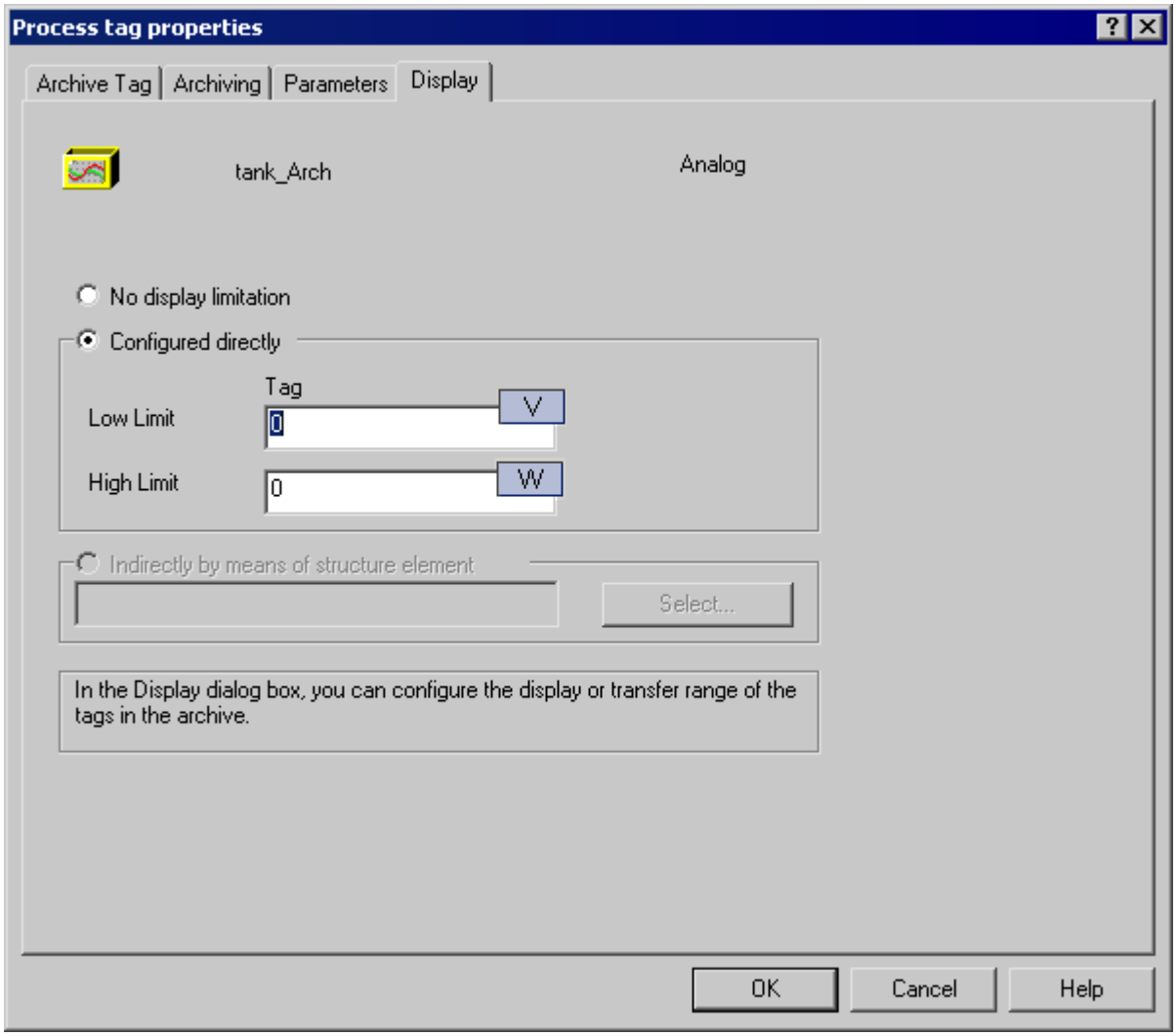
2. 下图显示了“归档”选项卡的参数。对于每个参数，将显示 WinCC 归档表格结构中的对应列。


Process tag properties
Archive Tag
Archiving
Parameters
Display
tank_Arch
Analog
G
Archiving type
cyclic
Acquisition:
500 ms
O
H
Archiving
Factor
Cycle
1
J
x
500 ms
I
Hysteresis:
P
abs.
in %
Q
0
Archive after segment change
A C
Display
Factor
Cycle
1
x
500 ms
Events
Tag
C script:
Actions
Start:
Tag
Y
C script:
X
Stop:
Tag
AA
C script:
Z
OK
Cancel
Help

3. 与模拟归档变量相比，二进制归档变量的“参数”选项卡的组织有所不同。关于模拟归档变量的组织已经有所叙述。在二进制归档变量中，“处理”、“单位”和“更改时归档”区域不可用。对于二进制归档变量，WinCC 归档表格结构中的相应列必须保留空白。另一方面，二进制归档变量中存在一个名称为“归档条件”的区域。该区域被分配给 WinCC 归档表格结构中的“R”列。如果是模拟归档变量，该列无用，必须保持空白。



4. 下图显示了“显示”选项卡的参数。对于每个参数，将显示 WinCC 归档表格结构中的对应列。“显示”选项卡对于二进制归档变量不可用。对于二进制归档变量，WinCC 归档表格结构中的相应列必须保留空白。



参见

组态归档参数 (页 265)

使用变量选择对话框 (页 279)

组态归档变量参数（过程控制）(页 275)

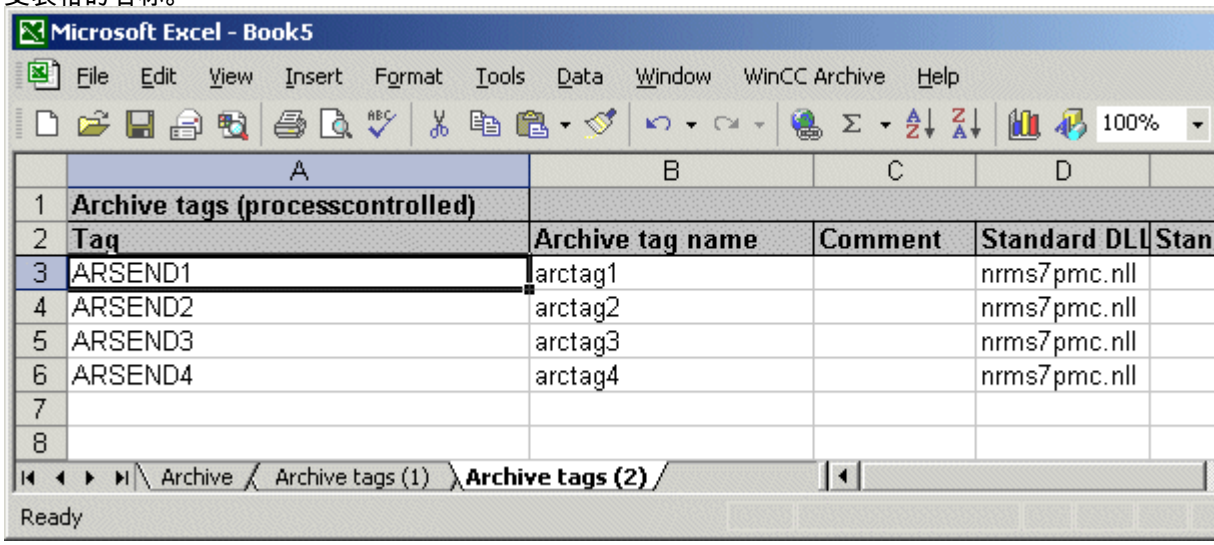
组态归档变量参数（过程控制）

引言

本说明阐述过程控制归档变量的组态。 必须为过程控制归档变量定义的各个参数将单独进行说明。

步骤

1. 激活用于组态过程控制归档变量的表格。 默认情况下，它具有名称“归档变量 (2)”。 可以改变表格的名称。



2. 表格的整个数据区可用于组态归档变量。 这样，在表格上最多可以组态 65534 个归档变量。 如果这样不够，可以添加更多表格。
组态过程控制归档变量时，必须指定要使用的格式化 DLL。 WinCC 归档只支持使用格式化 DLL“nrms7pmc.nll”组态过程控制归档变量。

归档变量参数

下表列出了必须为过程控制归档变量定义的所有参数。

	列	列表	参数	描述 1
1	A		Tag	这用于指定通过其执行归档的原始数据变量的名称。
2	B		归档变量名称	此处指定归档变量的名称。 该名称在归档中必须为唯一。
3	C		注释	这用于输入归档变量的注释。
4	D		格式 DLL	此处设置要使用的格式化 DLL。 只支持格式化 DLL“nrms7pmc.nll”。
5	E		格式化 DLL 参数 1	使用“nrms7pmc.nll”时： AR_ID
6	F		格式化 DLL 参数 2	使用“nrms7pmc.nll”时： AR_ID 子编号

以下部分说明“列表”列中所使用符号的意义。

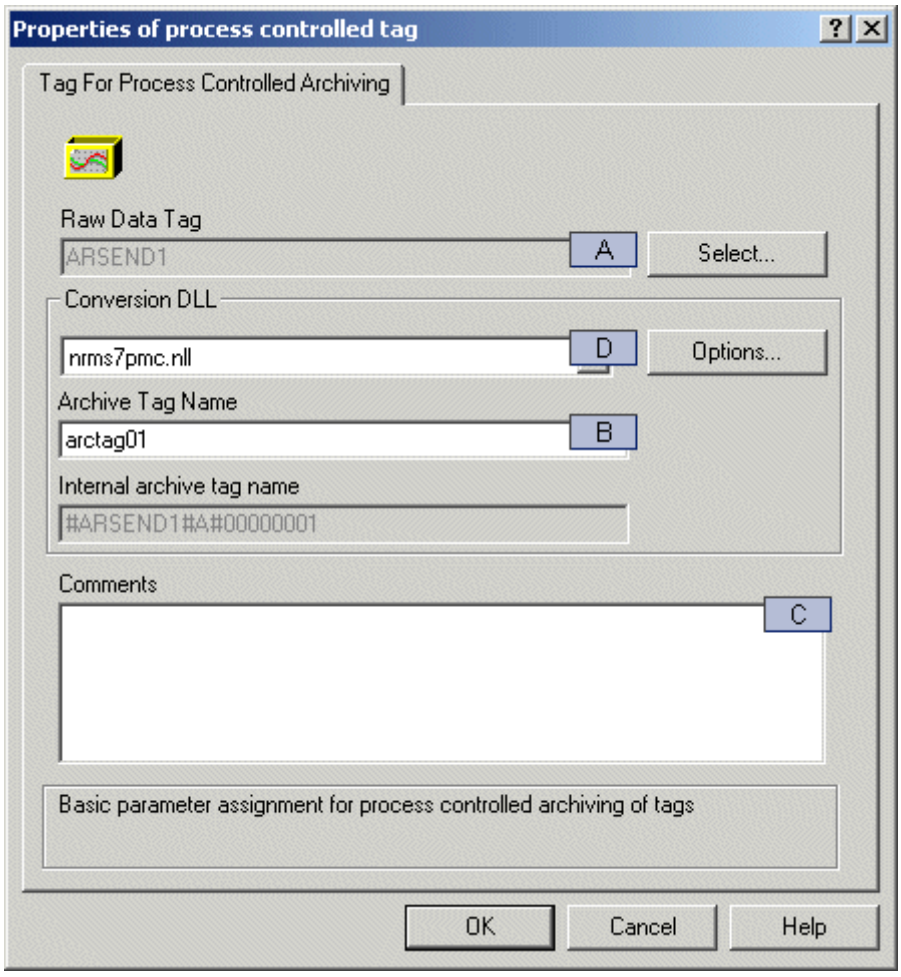
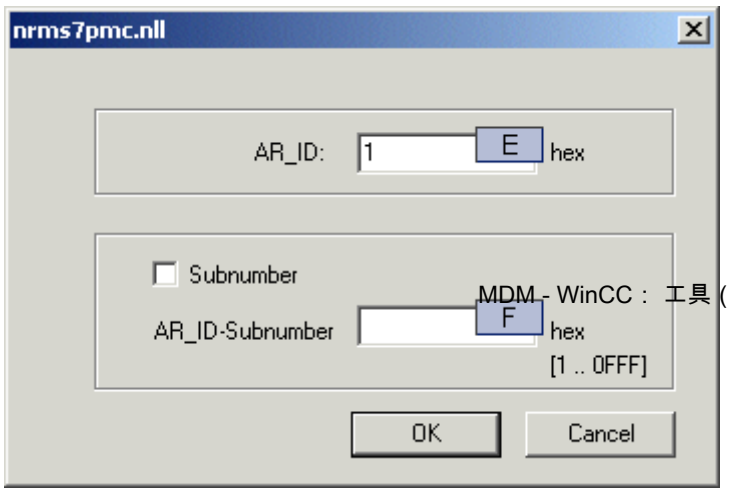
双击之后打开一个组合框。

双击之后打开变量对话框。 必须已经打开一个 WinCC 项目。

为过程控制归档变量提供有一个参数“归档变量名称”。 这在 WinCC 归档表格结构中可用。 此外，过程控制归档变量具有内部归档变量名称。 这不能由用户直接处理，因此未包含在 WinCC 归档表格结构中。 在传送至 WinCC 项目期间，自动分配内部归档变量名称。

变量记录编辑器

下图说明使用变量记录编辑器进行组态和使用 WinCC 归档进行组态之间的关系。 除了要在变量记录编辑器中设置的参数之外，还要指定 WinCC 归档中的各个列。

	标签
1	下图显示了“过程控制变量属性”对话框。 对于每个参数，显示 WinCC 归档表格结构中的相应列。 
2	下图显示“nrms7pmc.nll”对话框。 可以通过单击“过程控制变量属性”对话框中的“选项”按钮将其打开。 但是，可以这样做的条件是“nrms7pmc.nll”被定义为“格式化 DLL”（参考步骤 1）。 对于每个参数，显示 WinCC 归档表格结构中的相应列。 

参见

使用变量选择对话框 (页 279)

组态归档变量参数（二进制和模拟）(页 268)

组态归档参数 (页 265)

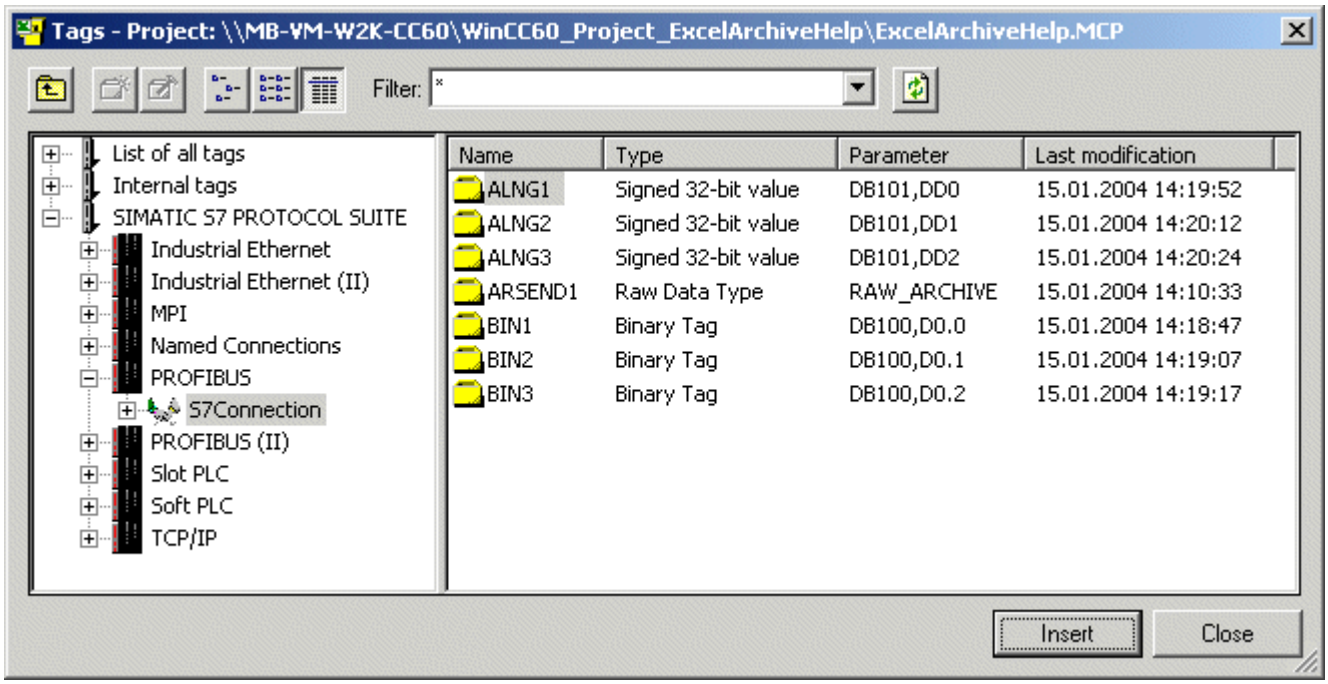
使用变量选择对话框

简介

组态归档变量时要输入的第一个参数是要归档的 WinCC 变量的名称。如同所有其它参数一样，可以在相应单元格中直接输入该参数。此外，WinCC Excel 还提供使用特殊对话框来选择变量的选项。它提供了当前打开的 WinCC 项目中的所有变量以供选择。如果没有打开任何 WinCC 项目，该对话框不可用。

步骤

1. 确保已打开一个 WinCC 项目。打开或创建一个归档文件夹来组态过程值归档。激活任一归档变量表。双击数据区中“变量”列的单元格。变量选择对话框打开。



2. 变量选择对话框不会禁止 Excel 的操作。也就是说，即使变量选择对话框打开，仍然可以继续归档文件夹中的工作。
在归档变量表中选择要插入 WinCC 变量名的单元格。在变量选择对话框中选择所需的 WinCC 变量。单击变量选择对话框中的“粘贴”按钮。所选的 WinCC 变量名被插入归档变量表中。变量选择对话框保持打开。
通过在选择列表中双击所需的变量，也可以粘贴单个 WinCC 变量。

3. 变量选择对话框支持多项选择。也就是说，通过同时按住 Ctrl 或 Shift 键可以一次选择多个 WinCC 变量。
   在归档变量表中选择要插入 WinCC 变量名的单元格。在变量选择对话框中选择所需的 WinCC 变量。单击变量选择对话框中的“粘贴”按钮。所选的 WinCC 变量名被粘贴到归档变量表中。变量选择对话框保持打开。
4. 可以通过输入过滤标准来限制显示在变量选择对话框中 WinCC 变量的数目。
   在“过滤”输入域中输入所需的过滤标准。例如，如果输入“B*”作为过滤标准，则只显示那些名称以字母 B 开头的变量。单击  按钮来应用过滤标准。 仅显示其名称与所选过滤标准相匹配的 WinCC 变量。
   输入“*”作为过滤标准即可再一次显示所有可用的 WinCC 变量。
5. 单击“关闭”按钮来终止使用变量选择对话框进行工作。

参见

组态归档变量参数（过程控制）(页 275)

组态归档变量参数（二进制和模拟）(页 268)

1.9.5.4 组态压缩归档

组态压缩归档

引言

本节描述使用 WinCC 归档组态压缩归档的常规步骤。

参见

组态归档变量参数 (页 284)

组态归档参数 (页 280)

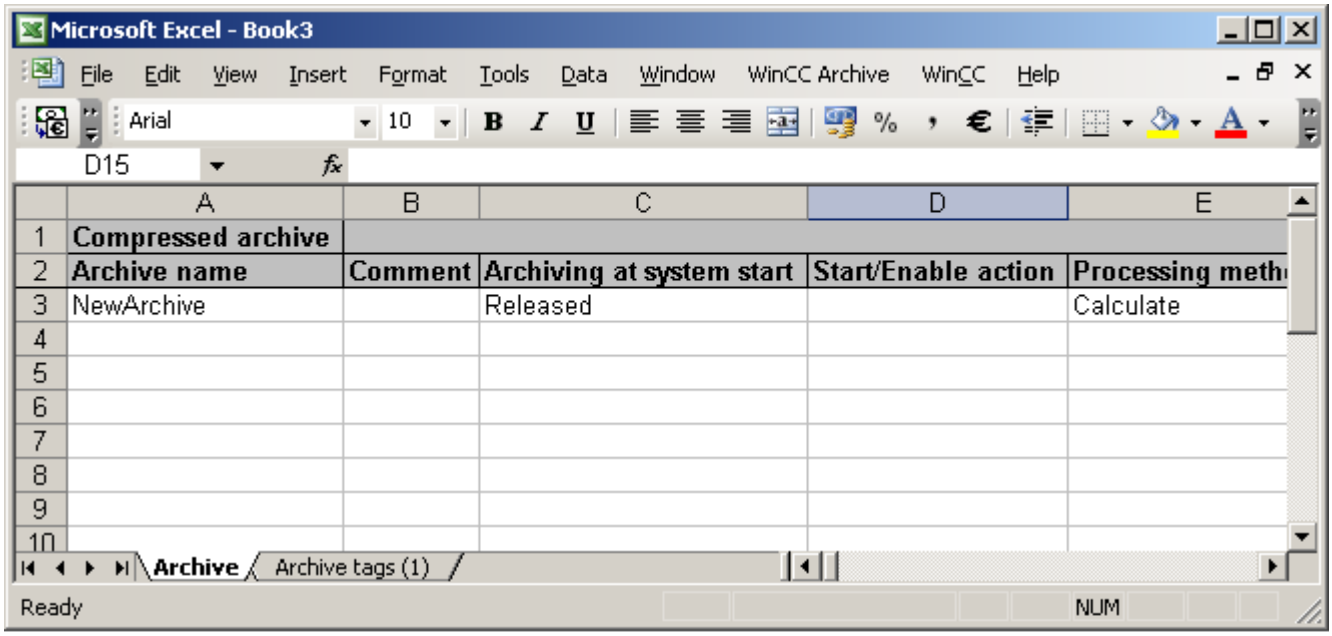
组态归档参数

简介

本节描述归档参数的组态。各个归档参数将单独说明。

步骤

1. 激活归档表格。其名称通常为“归档”。可以改变表格的名称。



2. 必须在归档表格数据区的第一行中定义归档参数。只有来自该行的内容才会被 WinCC 归档计算。

归档参数

下表列出了必须为过程值归档输入的所有参数。

	列	列表	参数	描述
1	A		归档名称	此处指定归档的名称。
2	B		注释	此处输入归档的注释。
3	C		在系统启动时进行归档	定义系统启动后应“启用”还是“禁用”归档。

	列	列表	参数	描述
4	D		启动/启用动作	指定开始归档时必须执行的操作。
5	E		处理方法	这用于定义应如何处理压缩归档。 可以选择“计算”、“计算并复制”、“计算并删除”和“计算、复制并删除”选项。
6	F		压缩周期	这用于指定将归档变量中的值压缩到压缩归档中的条目的周期。

以下部分说明“列表”列中所使用符号的意义。

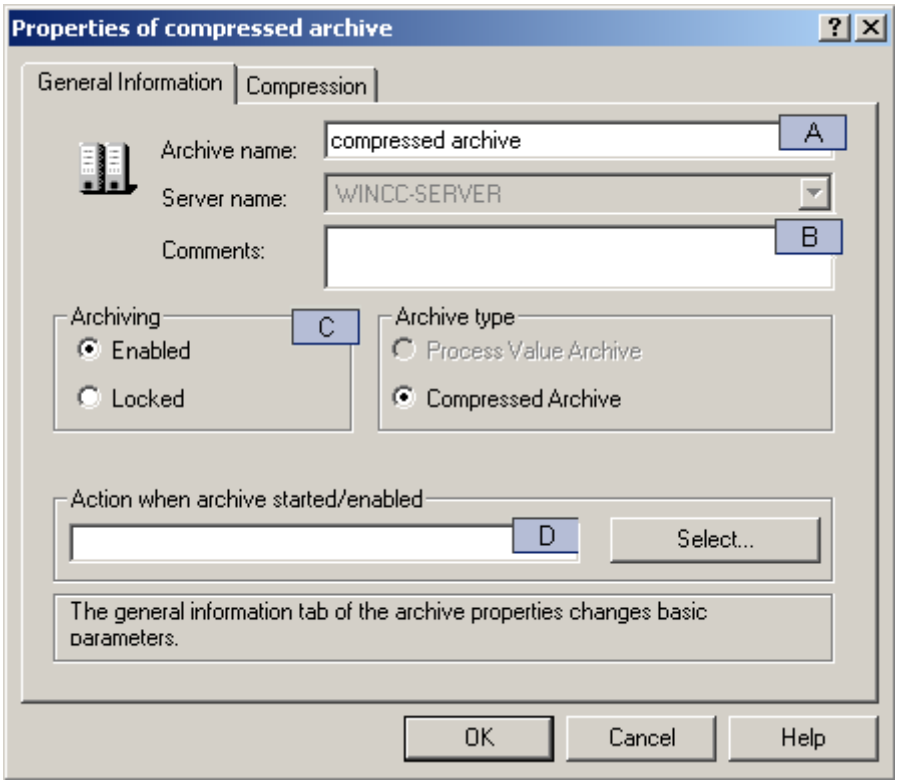
双击之后打开一个组合框。

双击之后打开一个组合框。必须打开一个 WinCC 项目。

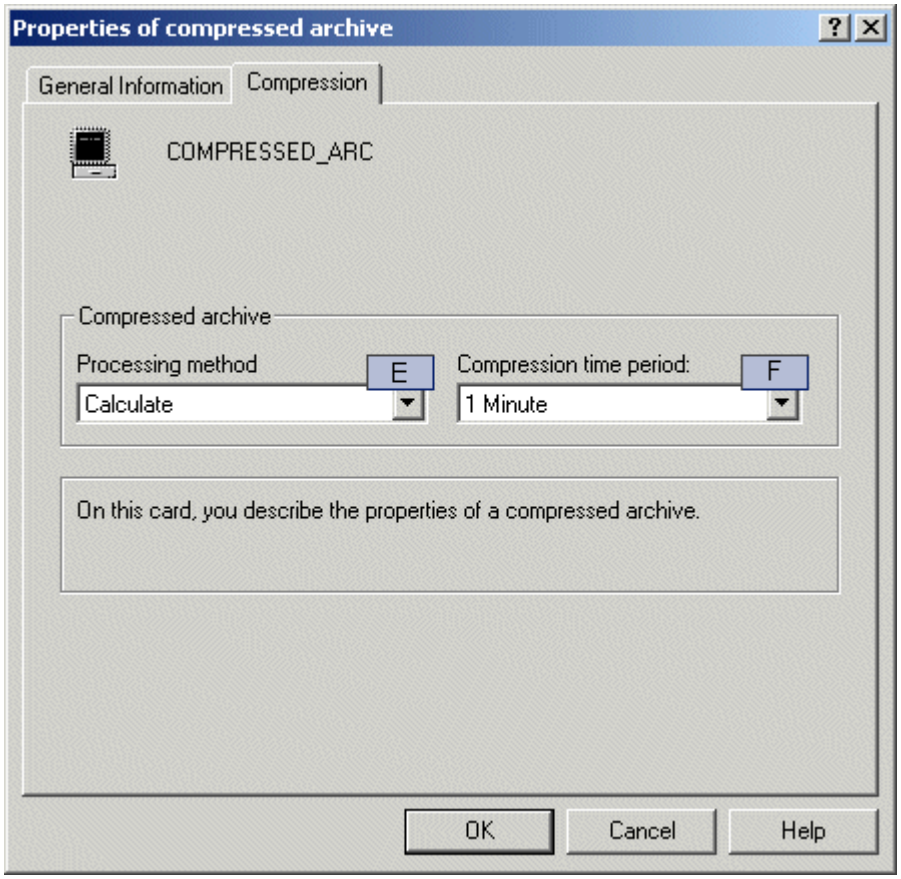
变量记录编辑器

下图说明使用变量记录编辑器进行组态和使用 WinCC 归档进行组态之间的关系。除了要在变量记录编辑器中设置的参数之外，还要指定 WinCC 归档中的各个列。

1. 下图显示了“压缩归档属性”对话框。“常规”选项卡已被选中。对于每个参数，将显示 WinCC 归档表格结构中的对应列。



2. 下图显示了“压缩归档属性”对话框。已经选择了“压缩”选项卡。对于每个参数，将显示 WinCC 归档表格结构中的对应列。



参见

组态归档变量参数 (页 284)

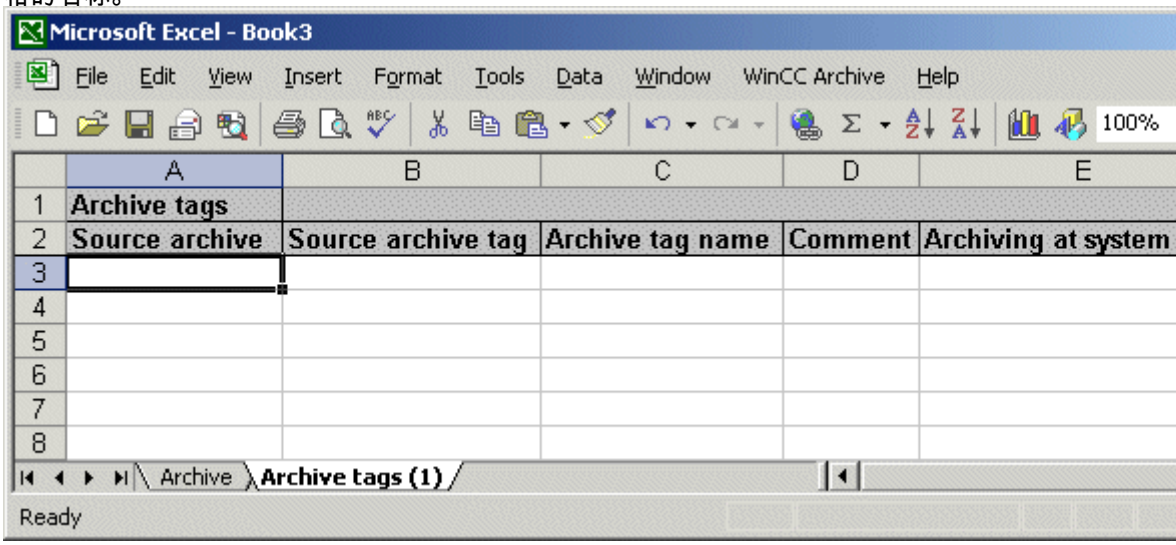
组态归档变量参数

引言

本说明阐述压缩归档变量的组态。 必须为压缩归档变量定义的各个参数将单独进行说明。

步骤

1. 激活用于组态压缩归档变量的表格。 默认情况下，它具有名称“归档变量 (1)”。 可以改变表格的名称。



2. 表格的整个数据区可用于组态归档变量。 这样，在表格上最多可以组态 65534 个归档变量。 如果这样不够，可以添加更多表格。

归档变量参数

下表列出了必须为压缩归档变量定义的所有参数。

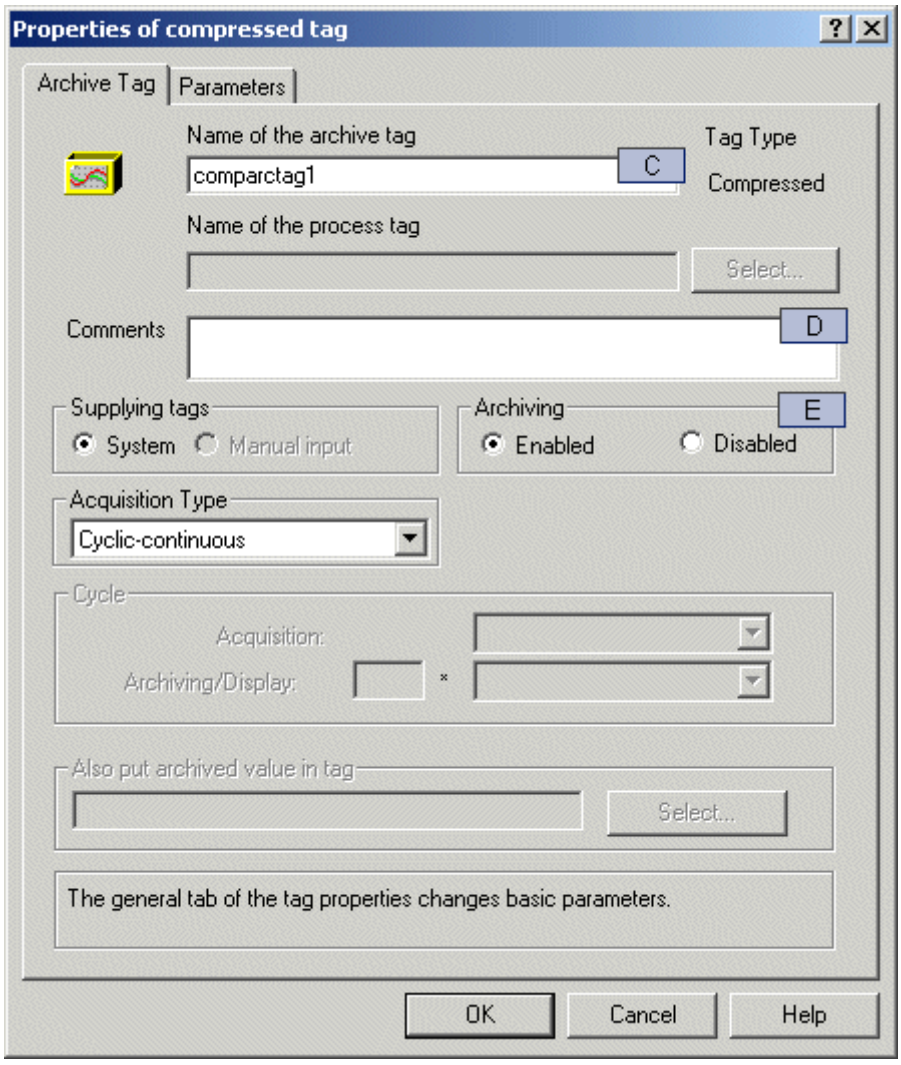
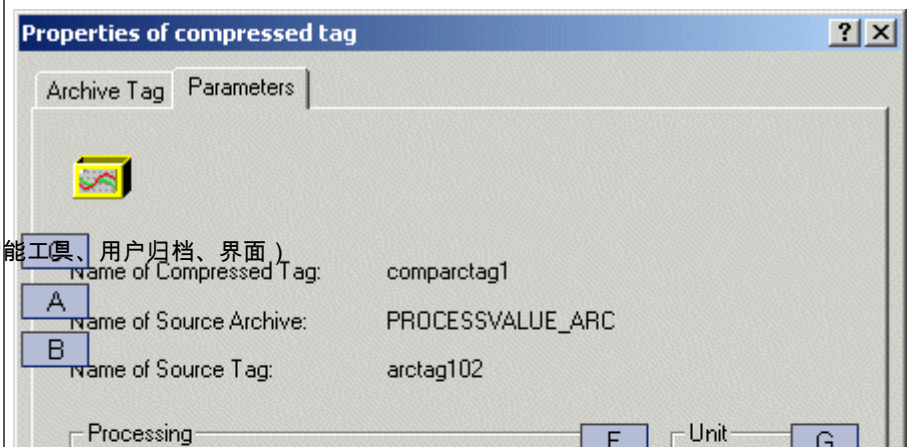
	列	列表	参数	含义
1	A		源归档	这用于指定包含要压缩的归档变量的归档的名称。
2	B		源归档变量	这用于指定要压缩的归档变量的名称。
3	C		归档变量名称	此处指定归档变量的名称。 该名称在归档中必须为唯一。
4	D		注释	这用于输入归档变量的注释。
5	E		在系统启动时进行归档	这用于定义系统启动后应“启用”还是“禁用”归档。
6	F		编辑	这用于定义应如何处理压缩归档变量值。 可以选择“平均值”、“总和”、“最小值”和“最大值”选项。
7	G		单位	这用于指定压缩归档变量值的单位。

以下部分说明“列表”列中所使用符号的意义。

双击之后打开一个组合框。

变量记录编辑器

下图说明使用变量记录编辑器进行组态和使用 WinCC 归档进行组态之间的关系。 除了要在变量记录编辑器中设置的参数之外，还要指定 WinCC 归档中的各个列。

	标签
1	下图显示了“压缩变量属性”对话框。 已经选择了“归档变量”标签。 对于每个参数，显示 WinCC 归档表格结构中的相应列。 
2	下图显示了“压缩变量属性”对话框。 已经选择了“参数”标签。 对于每个参数，显示 WinCC 归档表格结构中的相应列。 

参见

组态归档参数 (页 280)

1.9.5.5 检查归档数据

检查归档数据

引言

本节说明检查已组态数据的步骤。

组态期间，WinCC 归档不对数据执行任何测试。 结果是，用户可以使用所有选项来输入数据。 将归档传送至 WinCC 项目之前，必须确保组态正确。 为此，WinCC 归档提供了特殊的选项来检查归档数据。

参见

错误列表 (页 291)

使用错误对话框 (页 290)

检查归档数据 (页 288)

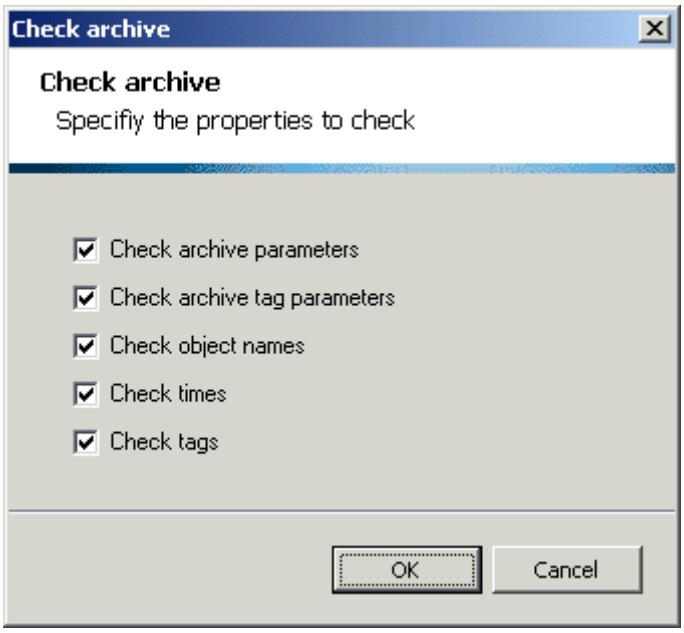
检查归档数据

引言

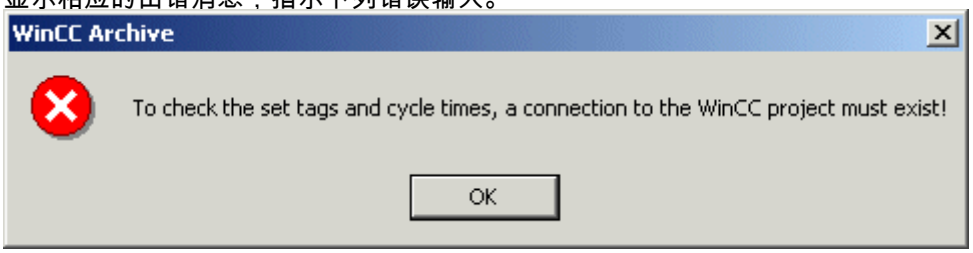
本说明阐述了 WinCC 归档提供的用于检查归档数据的选项。

步骤

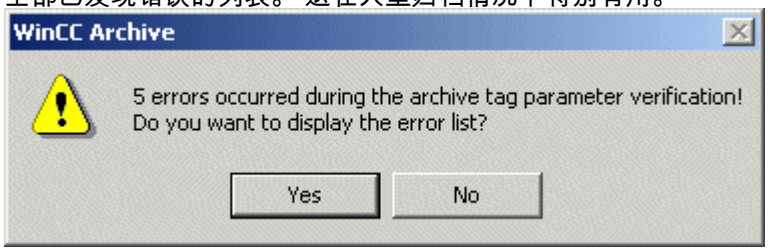
1. 打开要检查的归档文件夹。 选择菜单条目“WinCC 归档”>“检查归档”。 打开“检查归档”对话框。
“检查归档”对话框提供了对归档数据进行各种检查的选项。 可用的检查在下面进行了简要说明。
   - 检查归档参数： 检查归档表格中定义的设置。 此检查不包括归档变量表。
   - 检查归档变量参数： 检查归档变量表中定义的设置。 检查所有必要的参数是否已经设置且具有可以正确解释的值。
   - 检查对象名： 检查归档变量的名称在归档中是否唯一。 此外，检查名称是否符合归档变量名称约定。
   - 检查周期时间： 检查设置的周期时间在 WinCC 项目中是否可用。 还检查数值设置对于所需应用是否有效。
   - 检查变量： 检查设置的变量在 WinCC 项目中是否可用。 还检查数据类型对于所需应用是否有效。
   - 检查归档变量： 为压缩归档提供此检查，代替“检查变量”测试。 检查所有设置的归档变量在 WinCC 项目中是否可用。



2. 通过激活相应的复选框选择所需要的检查。 默认状态下，选择所有检查。 单击“确定”按钮。
“检查周期时间”和“检查变量”（或“检查归档变量”）检查只有在 WinCC 项目打开时才能执行。
显示相应的出错消息，指示下列错误输入。



3. 所选择的检查被连续执行。 只有在前一个检查已经正确完成时才能执行下一个检查。 每个没有检查到错误的对象在“错误文本”列中用“正确”标识。 每个错误的对象在“状态”列中用数值“1”标识。 检查到的错误的更详细说明显示在“错误文本”列中。
如果检查期间检测到错误，则通过相应的消息进行指示。 提供了一个选项，用来打开包含全部已发现错误的列表。 这在大型归档情况下特别有用。



4. 清除所有发现的错误并再次执行测试。 重复这些步骤，直至不再检测到错误。 在这种情况下，显示下列消息。

参见

错误列表 (页 291)

使用错误对话框 (页 290)

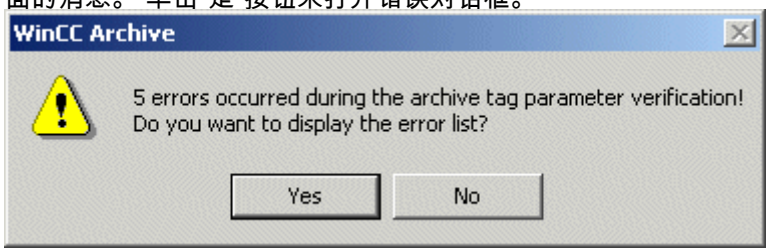
使用错误对话框

引言

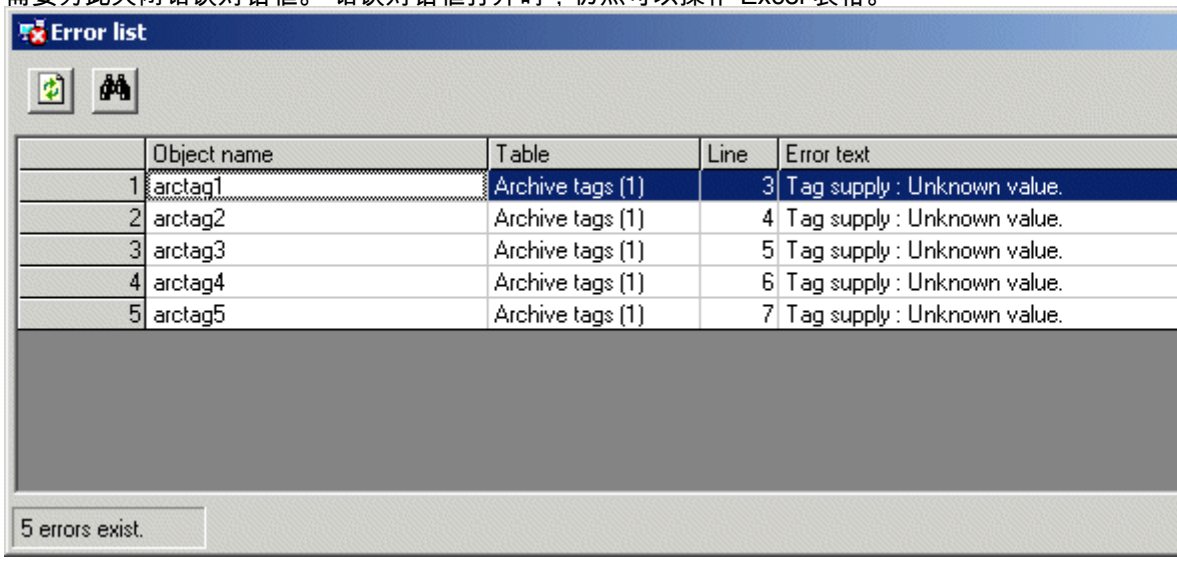
WinCC 归档以测试期间归档数据中所找到全部错误的总览形式为用户提供帮助功能。 帮助在错误对话框中提供。 本节描述错误对话框以及如何使用该对话框。

步骤

1. 检查完归档数据后，显示一条消息。 它提供关于测试结果的信息。 如果发现错误，显示下面的消息。 单击“是”按钮来打开错误对话框。



2. 打开错误对话框。 错误对话框包含所有错误对象的列表。 列表显示每个对象及其名称、相关表格的名称和相关的行号以及错误文本。
3. 双击列表中的对象直接跳到错误的对象。 对象在 Excel 表格中被标记，可以清除错误。 不需要为此关闭错误对话框。 错误对话框打开时，仍然可以操作 Excel 表格。



4. 如果已经关闭错误对话框，可以通过双击“状态”或“错误文本”列将其重新打开。 然而，这仅在归档文件夹包含错误时适用。

参见

错误列表 (页 291)

检查归档数据 (页 288)

错误列表

简介

本节包含检查归档数据时可能检测到的全部错误的列表。每个错误都有说明和关于如何清除的指示。

状态列

状态列可以包含下列内容。

	状态	描述
1		单元格为空。该行中的对象未经检查或检查成功完成。
3	1	单元格包含数值 1。该行中的对象已经检查过。检测到错误。

状态列可用于更快地定位错误对象。例如，可以在状态列中选择任何单元格。按下“End”按钮。然后按下向上或向下的光标键之一。选择跳到包含数值 1 的下一个单元格，也就是下一个错误对象。

错误文本列

错误文本列可以包含下列内容。

	错误文本	描述
1		单元格为空。该行中的对象未经检查。归档变量仅在变量被指定要进行归档时（或者是压缩归档中的归档变量）才进行检查。
2	OK	单元格包含文本“OK”。该行中的对象已经检查。执行的检查成功完成。
3	处理：未知的值。	单元格包含错误文本。该行中的对象已经检查。检测到错误。错误文本一方面包含含有错误源的列的名称，另一方面又包含产生的错误的更详细说明。

错误列表

	错误	描述
1	未知错误	产生了无法更精确定位的错误。
2	数据类型错误	该参数需要不同的数据类型。例如，输入了文本，但是需要数字值。
3	文本过长	指定文本的长度超出该参数允许的值。
4	值超出允许的限制	指定的值超出该参数允许的值范围。
5	周期时间比无效	采集周期和归档周期之比不正确。归档周期必须是采集周期的整数倍。
6	未指定任何周期时间	必须为该参数指定周期时间。

	错误	描述
7	未知的值	指定的文本不符合该参数可选择的任何选项。可使用组合框输入参数。
8	变量未找到	指定的变量在 WinCC 项目中未找到。
9	周期时间未找到	指定的周期时间在 WinCC 项目中未找到。
10	变量不是模拟量	必须为该参数指定模拟变量。这对于变量参数是必须的，例如，当归档变量类型定义为模拟量时。
11	变量不是二进制	必须为该参数指定二进制变量。这对于变量参数是必须的，例如，当归档变量类型定义为二进制时。
12	名称无效	对象的名称不符合所需的约定。 例如，它可能包含无效的特殊字符。
13	名称不是唯一	对象的名称不是唯一。它在归档内多次使用。
14	未指定任何名称	没有为对象分配任何名称。
15	无原始数据变量	必须为该参数指定原始数据变量。过程控制归档变量的变量参数就是这种情况。
16	组态已经在使用	该过程控制归档变量的组态已经在使用。然而，它在归档内必须唯一。从组态中产生一个内部归档变量名称。
17	源无效	指定的压缩归档变量的源不正确。例如，它完全丢失或部分丢失。
18	源不存在	为压缩归档变量指定的源在 WinCC 项目中未找到。不能将原来的归档指定为源归档。
19	时间值小于一分钟	必须指定大于或等于一分钟的时间作为压缩周期。

参见

使用错误对话框 (页 290)

检查归档数据 (页 288)

1.9.5.6 创建、修改和删除

创建、修改和删除

引言

本节说明创建、修改和删除整个归档以及单个归档变量的常规步骤。

参见

删除归档 (页 296)

删除单个归档变量 (页 299)

创建和修改单个归档变量 (页 298)

创建完整的归档 (页 294)

创建完整的归档

引言

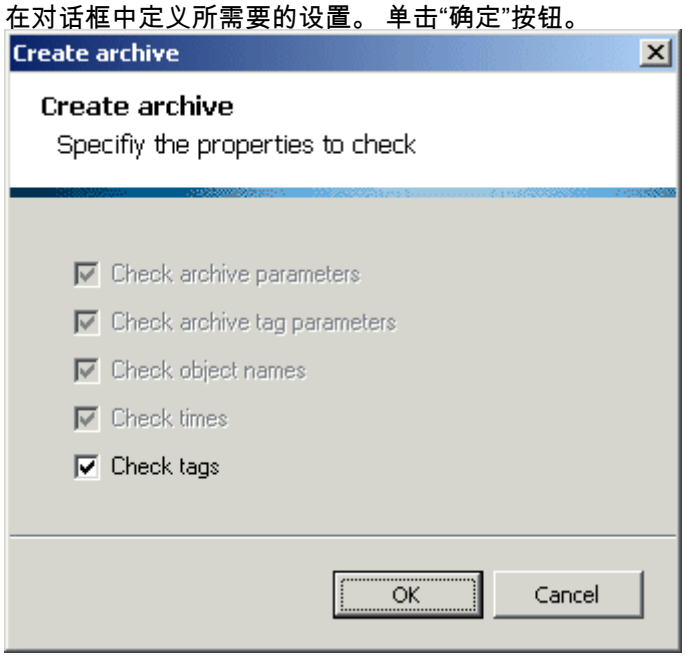
本说明阐述了创建要在归档文件夹中进行组态的归档的步骤。 使用所描述的选项只能创建完整的归档。 如果已经存在相应的归档，该归档将被删除。

检查归档数据

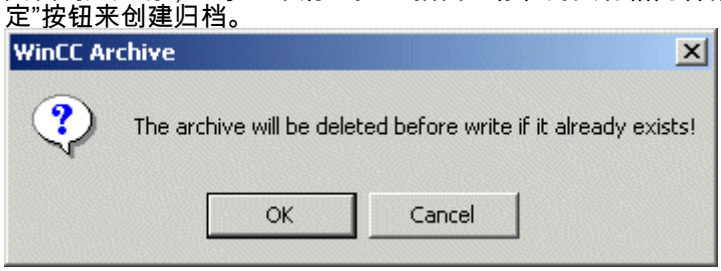
创建归档之前，组态的数据应该进行所有可用的测试。 原则上，在创建归档时也可以执行测试。 在这种情况中，创建所有无错误的对象，然而却不创建有错误的对象。 清除发现的错误后，可以再次创建完整的归档（所有先前创建的对象被预先删除）。 也可以有选择地创建有错误的对象。 相反，要在创建归档之前检查数据，所描述的其它两种方法耗时更长。

步骤

1. 打开要在其中创建归档的归档文件夹。 应该预先检查组态的数据。 选择菜单条目“WinCC 归档”->“创建归档”。 打开“创建归档”对话框。
“创建归档”对话框显示归档创建期间所执行的全部检查。 大部分测试始终执行，不能取消激活。 这涉及所有时间相对较少的检查。
检查变量或压缩归档中检查归档变量可以被取消激活。 如果是较大的 WinCC 项目，这些检查可能会十分耗时。 如果在创建归档之前进行检查，则在写入期间它就不再必要。 在这种情况下，它可以被取消激活。
2. 在对话框中定义所需要的设置。 单击“确定”按钮。



3. 归档创建之前，显示一条消息。 它指出已存在的具有相同名称的归档将被删除。 单击“确定”按钮来创建归档。



4. 归档被创建。 归档变量以最多 256 个对象的块写到 WinCC 项目。 创建过程完成时，显示一条消息。 它提供信息指示是否产生了错误。 如果错误已经产生，使用错误对话框来定位错误的对象。

参见

删除归档 (页 296)

删除单个归档变量 (页 299)

创建和修改单个归档变量 (页 298)

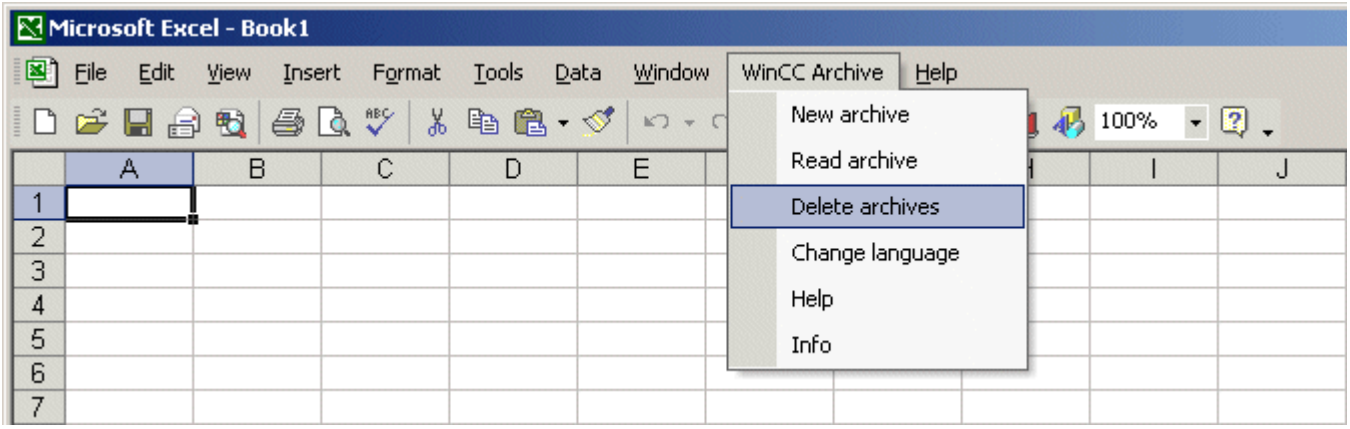
删除归档

简介

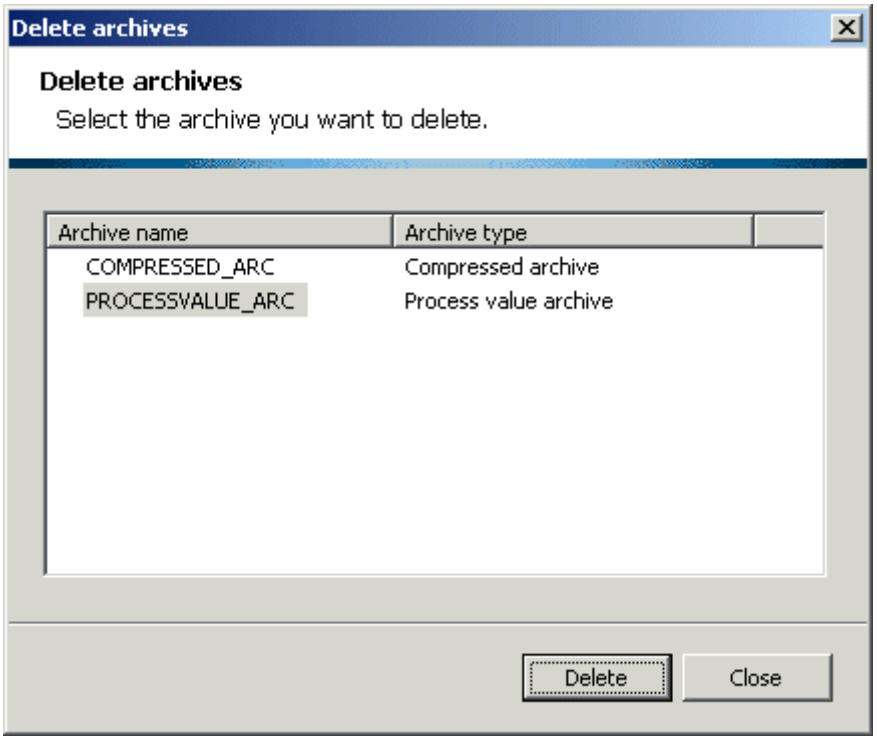
本说明阐述了从 WinCC 项目中删除任意归档的步骤。

步骤

1. 打开 Excel。Excel 加载项 WinCC 归档同时自动启动。Excel 菜单包含“WinCC 归档”条目。从该菜单条目中选择“删除归档”。



2. 打开一个对话框，其中包含 WinCC 项目中所有可用归档的列表。 选择要删除的归档并单击“删除”。将从 WinCC 项目中删除此归档。对话框保持打开。单击“关闭”关闭对话框。



参见

删除单个归档变量 (页 299)

创建和修改单个归档变量 (页 298)

创建完整的归档 (页 294)

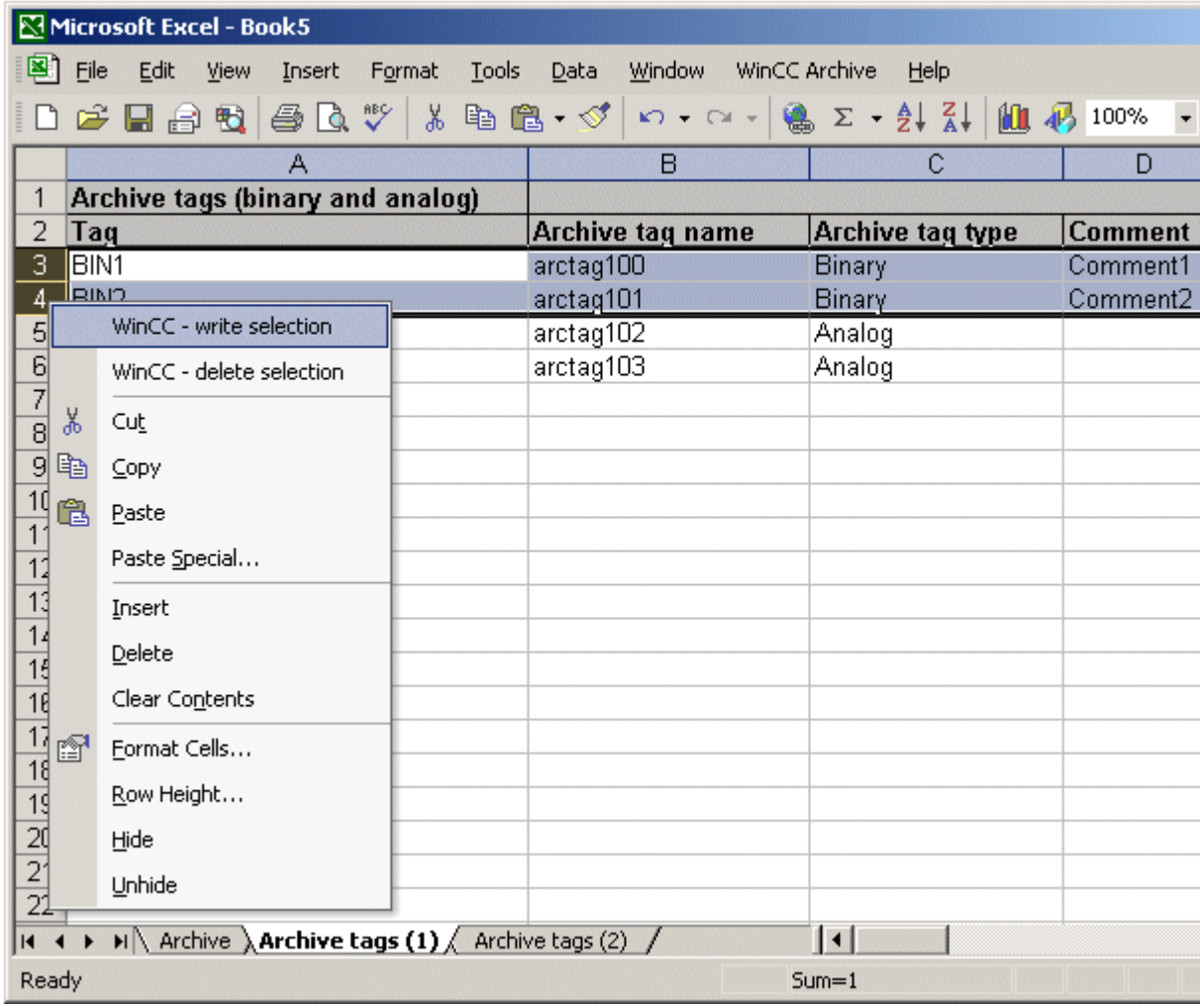
创建和修改单个归档变量

引言

本说明阐述了将单个归档变量添加到已存在归档的步骤和修改已存在归档变量的步骤。

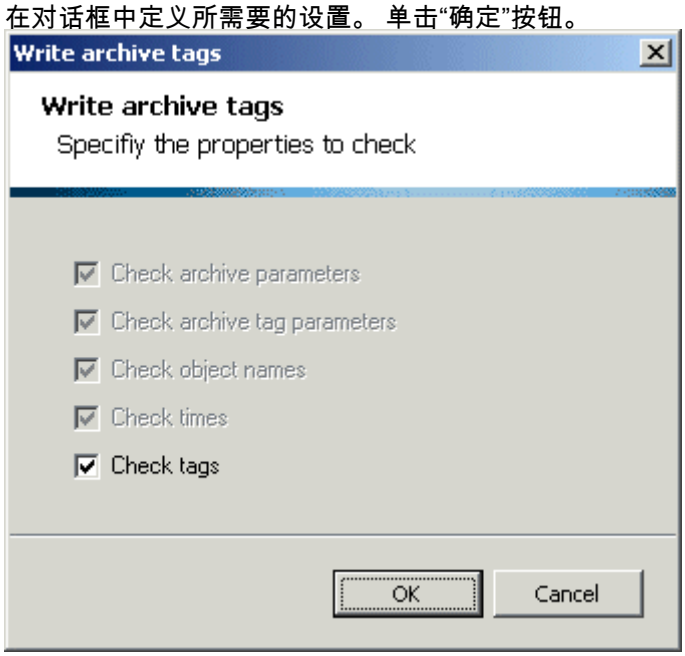
步骤

1. 打开包含要编辑的归档的归档文件夹。 也可以使用菜单条目“WinCC 归档 -> 读归档”从 WinCC 项目读入要编辑的归档。
2. 对归档进行必需的修改。 添加新的归档变量或修改已存在的归档变量。 选择新添加或修改的对象 打开弹出式菜单并选择“WinCC - 写选择”。



3. “写归档变量”对话框打开。 对话框显示写归档变量期间执行的所有检查。 大部分测试始终执行，不能取消激活。 这涉及所有时间相对较少的检查。
检查变量或压缩归档中检查归档变量可以被取消激活。 如果是较大的 WinCC 项目，这些检查可能会十分耗时。 如果预先执行了检查，则在写过程期间不需要再次执行。 在这种情况下，它可以被取消激活。

4. 在对话框中定义所需要的设置。 单击“确定”按钮。



5. 写入归档变量。 写过程完成时，显示一条消息。 它提供信息指示是否产生了错误。 如果错误已经产生，使用错误对话框来定位错误的对象。

参见

删除归档 (页 296)

删除单个归档变量 (页 299)

创建完整的归档 (页 294)

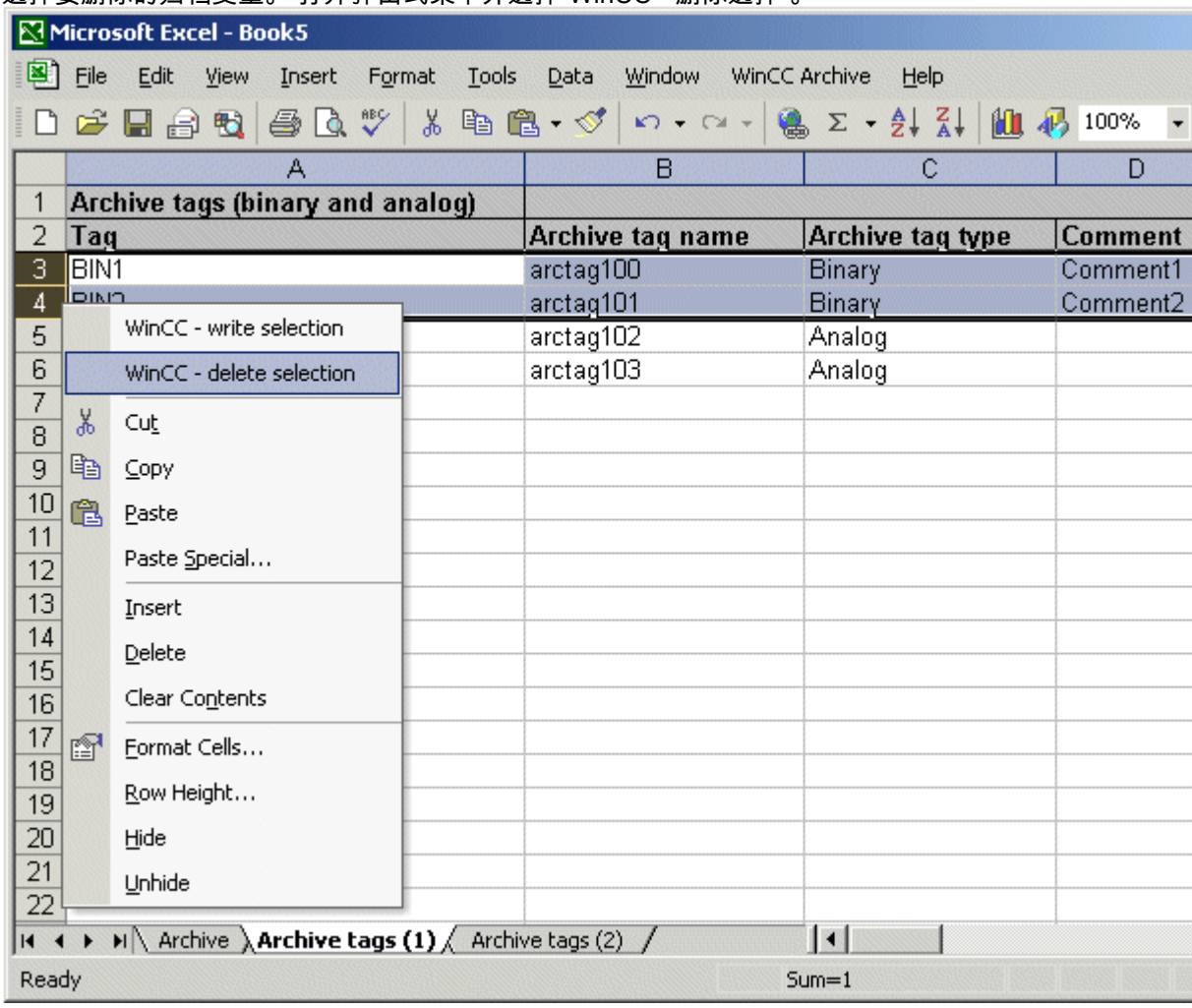
删除单个归档变量

引言

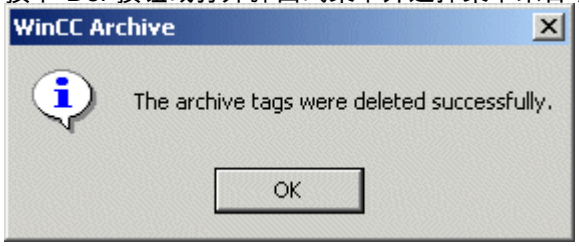
本说明阐述了从已存在的归档中删除单个归档变量的步骤。

步骤

1. 打开包含要编辑的归档的归档文件夹。 也可以使用菜单条目“WinCC 归档 -> 读归档”从 WinCC 项目读入要编辑的归档。
2. 选择要删除的归档变量。 打开弹出式菜单并选择“WinCC - 删除选择”。



3. 归档变量被删除。 删除过程完成时，显示一条消息。
4. 从 WinCC 项目中删除的归档变量没有从 Excel 表格删除。 要从 Excel 表格删除归档变量，按下“Del”按钮或打开弹出式菜单并选择菜单条目“删除内容”。



参见

创建和修改单个归档变量 (页 298)

删除归档 (页 296)

创建完整的归档 (页 294)

用户归档 2

2 资源

2.1 用户归档简介

在“用户归档”简介中，您将获得以下信息：

- 用户归档的应用领域
- 用户归档编辑器的组件
- 组态和运行系统
- 用户归档的功能范围

WinCC 用户归档编辑器可用来在服务器 PC 上连续保存来自工艺过程的数据。 在图形编辑器中，可以组态 WinCC 用户归档控件来以表格显示运行系统中用户归档的在线数据。

用户归档还可用于准备自动化系统（例如 S5、S7）的数据。 如果必要，数据可以配方或设定值的形式从控制器读出。

用户归档编辑器提供两种数据库表格：

- 用户归档： 用户归档是用户可在其中创建自己的数据域的数据库表格。 用户归档用于存储数据，并根据 SQL 数据库约定提供对这些数据的标准化访问。
- 视图： 视图接收来自用户归档的数据并用于数据的汇总，例如，以便获得有关产品组的概述。

对于用户归档的创建和编辑，有两种可能性：

- 使用可轻松地交互式组态用户归档的用户归档编辑器。
- 使用 WinCC 脚本语言中用于编辑用户归档的函数。

通过 WinCC 脚本语言的函数，还可在运行模式下执行多种操作。 在运行系统画面中，可以组态与自动化系统的过程映像直接相连的表格。

参见

用户归档的应用领域 (页 306)

2.2 “用户归档”编辑器

2.2.1 “用户归档”编辑器简介

2.2.1.1 “用户归档”编辑器

“用户归档”编辑通过其 Windows 用户界面提供了一种轻松创建和维护用户归档的方法。
“用户归档”编辑器的工作区分为三个区域：

① 导航窗口

② 数据窗口

③ 表格窗口

- 用于选择用户归档和视图的**导航窗口**。
- 用于显示和更改域的**数据窗口**。 在数据窗口中，将显示在导航窗口中选择的用户归档和视图的域。
- 用于显示和修改所选用户归档和视图的在线数据的**表格窗口**。 在编辑器的表格窗口中，可以实现到 PLC 过程映像的在线连接。

该编辑器的导航窗口和数据窗口具有类似于资源管理器的用户界面，可快速访问用户归档的所有元素。使用对话框和向导可轻松完成用户归档的创建和更改。

参见

用户归档的应用领域 (页 306)

2.2.1.2 WinCC 用户归档控件

可使用图形编辑器组态 WinCC 用户归档控件。 用户归档控件用于在运行期间显示和更改用户归档数据。

WinCC 用户归档控件的基本属性：

- 用户归档控件通过图标和组合键进行操作。
- 在用户归档控件中，可创建、更改或删除域的内容。
- 浏览功能还使对大型用户归档的访问变得很简单。
- 可以导入和导出用户归档。
- 选择标准和排序条件可用于确定显示内容。
- 通过与自动化系统的直接连接，可以在线读取和写入数据。

在组态过程中，用户归档控件连接到选定的用户归档或窗体，并且只能访问该用户归档或窗体。要进行访问，必须启用用户归档/窗体（访问保护）。 在用户管理器中，可以将特定授权分配给用户归档控件。

如果取消了该访问保护，则必须在图形编辑器中将用户归档控件重新连接到用户归档，以便用户归档控件能检测已取消的访问保护。

当打开用户归档控件的画面时，将查询归档或域的访问保护。必须通过对象属性（例如画面、I/O 域或按钮的属性）来对受保护归档的控制变量单独执行访问保护。

说明

在 WinCC V7 之前，用户归档的显示在用户归档表格元素中组态。

2.2.1.3 WinCC 脚本语言的标准函数

WinCC 脚本语言的函数分为：

- 用于组态用户归档的**组态函数**
- 用于组态运行系统操作中动作的**运行系统函数**

运行系统函数可通过在运行系统画面中的操作来激活，例如，鼠标单击指定按钮。 WinCC 脚本语言基于高级 C 语言，而数据库函数基于 SQL 标准语言。

参见

用户归档简介 (页 303)

2.2.1.4 用户归档的应用领域

功能范围

组态用户归档时，可以使用“用户归档”编辑器或 WinCC 脚本语言的函数来创建自己的数据库表格。

即使在组态过程中，也可以使用“用户归档”编辑器创建新的数据记录和编辑现有数据记录中的数据。

在运行系统中，可将 WinCC 用户归档控件的画面窗口中的用户归档（即数据库表格）显示为表格。 通过原始数据变量或 WinCC 变量，可实现与 AS 的连续数据交换。

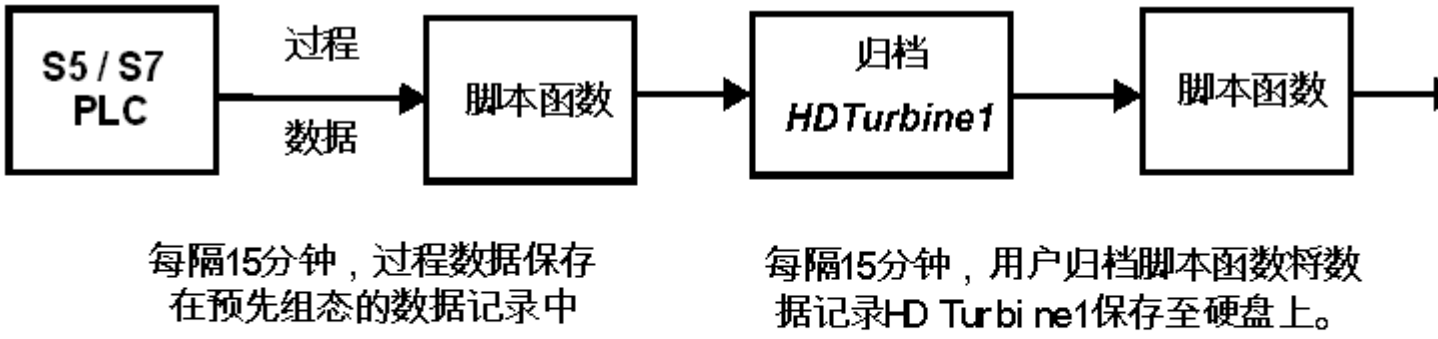
涡轮的可操作数据记录的实例

能源生产操作员设置用户归档“HDTurbine1”。该用户归档用于对高压涡轮进行运行状态监视。用户归档“HDTurbine1”具有下列数据域：

HDTurbine1
Index
Speed
Inlet pressure
Exit pressure
Steam temperature1
Steam temperature2
Oscillation frequency
Oscillation amplitude

Storage temperature1
Storage temperature2

在运行系统中，则可以按设置的间隔将涡轮的操作数据以用户归档数据记录的形式保存到 PC 大容量存储器（硬盘）上：



可以使用 WinCC 脚本语言函数分析用户归档的新近数据，或使用 WinCC 用户归档控件进行显示。

饮料生产商配方的实例

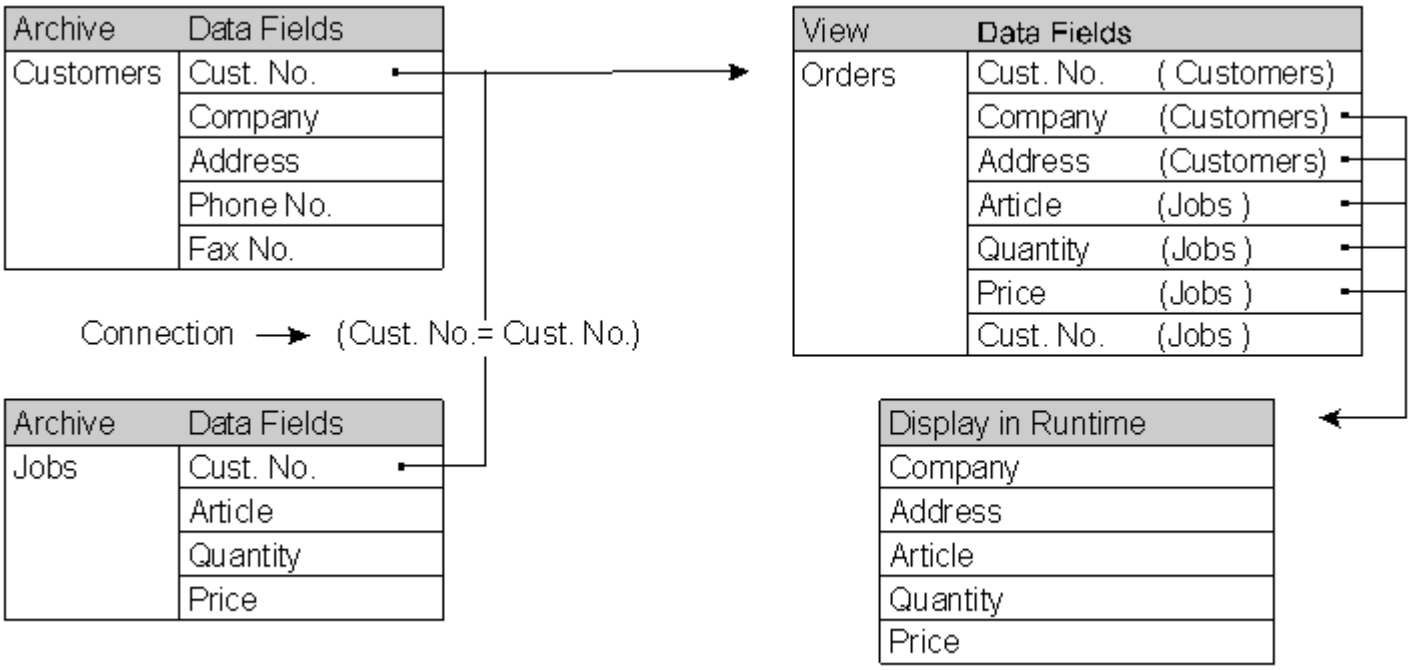
PLC 的数据流为配方的实例。在本实例中，生产可乐类饮料和橙汁的饮料生产商使用用户归档为 AS 准备其饮料成份的配方。

用户归档	数据域
Cola	Water
	Sugar
	Dyestuff7
	Phosphoric acid
	Caffeine

用户归档通过 WinCC 数据管理器中的原始数据或 WinCC 变量，使用由 WinCC 提供的与 AS 的数据接口。对于传送到 AS 或从 AS 传出的数据，WinCC 提供一组适当的 C 动作。

视图的应用方法

WinCC 为服务器的多个用户归档提供了附加性能标记“视图”。 视图能够汇总来自不同用户归档的数据域。因此，例如，可以通过不同用户归档的数据域用 SQL 语言创建逻辑操作，以便以视图形式显示运行系统中必需的关系。所使用的用户归档必须至少具有一个公共要素。



本实例中，用户创建订单的视图。用户从归档“Customers”和“Jobs”中获取订单的所需信息。客户号码是两个用户归档的公共要素，并作为视图的连接标记。用户只需要在运行系统中显示已连接用户归档的所需域。

说明

现有软件（已实现针对低于 4.02 版用户归档的 ODBC 数据库直接访问）无法访问 4.02 或更高版本的用户归档。

2.2.1.5 用户归档的功能范围

以下简短标题中介绍了用户归档的性能标记：

组态

- 可以以表格形式创建用户归档和视图，这样即可简单直接地将数据分配给用户归档或视图（按行和列分配）的域。
- 以表格形式在运行系统中在线显示
- 通过 I/O 域输入/输出数据（通过 C 动作/控制变量分配用户归档域）

记录

- 通过 WinCC 报表在表格视图中记录组态数据和运行系统数据
- 以 CSV 格式导出数据（通过外部程序进行处理，例如 Excel）

传送自/到 AS（S5、S7 等）

- 用户归档的完整数据记录（通过原始数据变量）
- 数据记录的单个数据域（通过 WinCC 变量）
- 通过 WinCC 提供的所有接口进行通讯

编辑可能性

- 在表格视图中
- 通过 I/O 域（利用 C 动作/控制变量）

操作

- 通过标准化图标在表格中操作
- 使用 C 动作

删除或创建新的数据记录

- 通过按钮在表格中创建数据记录
- 通过 C 动作创建和删除数据记录

控制变量

- 向导支持创建 WinCC 变量作为控制变量
- 使脚本和 AS 能快速访问用户归档
- C 动作的间接寻址

数量框架

在“用户归档”编辑器中，最多可组态 500 个归档和 500 个归档视图。每个归档最多可创建 500 个域。

归档

归档中的数据记录的最大数目受到限制，取决于已组态列的数目和归档中包含的数据记录。列和数据记录的乘积不能大于 320000。如果选择了“上个用户”和“上次访问”列，则列的数目还必须包括由系统创建的“ID”列。

实例：

归档中组态了 15 个单独的列，且选择了“上次访问”列。因此，包括“ID”列在内，一共组态了 17 列，从而最大可提供 320000 / 17 = 18823 条数据记录。

参见

用户归档的应用领域 (页 306)

2.2.2 “用户归档”编辑器

2.2.2.1 “用户归档”编辑器的结构

组态

可通过“用户归档”编辑器的菜单、工具栏，热键或直接单击鼠标来对其进行操作。

以下将介绍有关下列主题的信息：

- “用户归档”编辑器的菜单
- “用户归档”编辑器的工具栏

2.2.2.2 “用户归档”编辑器的菜单

“用户归档”编辑器的菜单

菜单操作

本节将描述菜单操作。此处不会描述与 Windows 标准相对应的功能。

“用户归档”编辑器包含下列菜单：

菜单	菜单命令	快捷键
项目	恢复	Ctrl + N
	保存	Ctrl + S
	导出...	
	导入...	
	检查...	
	退出	
编辑	剪切	Ctrl + X
	复制	Ctrl + C
	粘贴	Ctrl + V
	运行系统数据	Ctrl + R
	选项	Ctrl + O
视图	工具栏	
	状态栏	
	拆分	
	更新	F 5
运行系统数据	导入	
	导出	
帮助	帮助主题	
	日志文件...	
	关于...	

说明

“剪切、粘贴和复制”功能只能在数据窗口中使用。一次只能剪切、复制或粘贴一个用户归档、域或视图。只能在当时没有引用用户归档时（例如，在组态系统 (CS) 或运行系统中显示表格窗口时会引用用户归档）进行保存。

菜单命令“恢复”

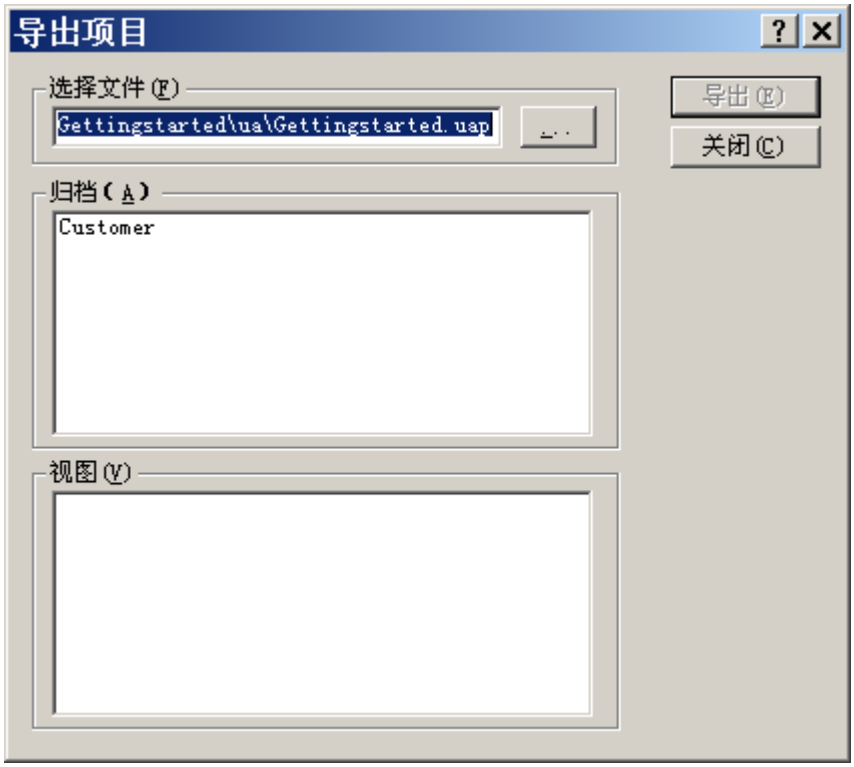
恢复

通过“恢复”，可在不关闭编辑器的情况下取消上次更改并恢复上次保存的状态。 此外，可使用此功能以接受在打开脚本编辑器或外部程序后执行并保存的更改。 编辑器不会自动完成此类外部更改。

菜单命令“导出”（菜单项目）

导出（菜单项目）

使用此菜单命令以导出已打开 WinCC 项目的用户归档和视图结构（CS 数据）。



单击菜单“项目”的导出命令时将打开一个对话框，可在其中选择在已打开 WinCC 项目中创建的用户归档和视图。 可以单选和多选。 在文件选择区域中，会自动设置打开项目的项目路径、项目名称的文件名和文件扩展名“uap”。 单击选择文件的按钮后将打开一个选择对话框，可在其中设置任意选择的存储位置。 设置存储位置后，将在单击按钮“导出”后执行导出。 导出选定的用户归档和视图后，关闭该对话框。

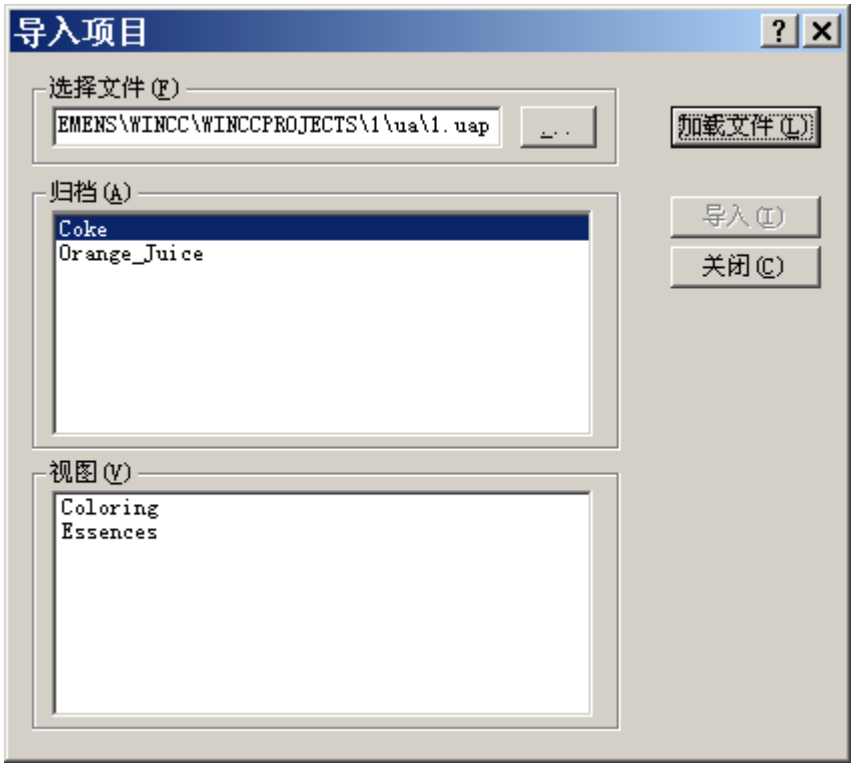
说明

要导出运行系统数据，必须在菜单“运行系统数据”中使用菜单命令“导出”。

菜单命令“导入”（菜单项目）

导入（菜单项目）

使用此菜单命令以导入已打开 WinCC 项目的用户归档和视图结构（CS 数据）。



要能够导入用户归档和视图，必须首先导出将要导入的项目中的各个用户归档和视图，以便创建 uap 文件。 要启动导入，单击菜单“项目”中的命令“导入”。 将打开一个对话框，用于选择需要导入的用户归档和视图。 可以单选和多选。 在文件选择区域中，会自动设置打开项目的项目路径和文件名，包括项目的名称和文件扩展名“uap”。 单击选择文件的按钮时

将打开一个选择对话框，可在此选择需要导入的文件。 选择文件后，将在单击按钮“导入”后执行导入。 导入选定的用户归档和视图后，关闭该对话框。

为保持结构数据的一致性，务必将视图和相关归档同时导出，并在导入时与该视图同时导入。 不得在导入过程中覆盖具有相同名称的现有归档。 如果要保留归档名称，必须在导入前先删除项目中存在的具有相同名称的归档。 必须首先保存现有运行系统数据，因为在删除归档时也将删除这些数据。

说明

要导出运行系统数据，必须在菜单“运行系统数据”中使用菜单命令“导入”。

菜单命令“检查”

检查

该菜单命令允许检查用户归档编辑器中引用的变量是否存在于 WinCC 项目管理器中。 如果检测到错误，将得到以下消息：“检查过程中未发现任何错误”。 如果发生错误，将得到以下错误消息。

将显示变量管理器中的用户归档、相关的域和丢失的变量。

说明

该功能不会检查结构变量。

菜单命令“运行系统数据”

运行系统数据

该菜单命令允许在表格窗口中编辑在线数据。 菜单中的检查将显示“运行系统数据”已激活。

菜单命令“选项”

选项

使用该菜单命令以设置如何完成用户归档和视图的创建。 单击此菜单后将显示以下对话框：

循环创建用户归档：

如果此选项为激活，则在自动输入用户归档和其域以后，将显示用于输入更多用户归档的对话框。

循环创建域：

如果此选项为激活，则在自动输入用户归档数据域以后，将显示用于输入更多数据域的对话框。

循环创建视图：

如果此选项为激活，则在自动输入视图和其列以后，将显示用于输入更多视图的对话框。

循环创建视图的列：

如果此选项为激活，则在自动输入列以后，将显示用于输入更多列的对话框。

菜单命令“拆分”

拆分

该菜单命令用于更改用户归档编辑器的三个分割窗口的大小。

菜单命令“导入”（“运行系统数据”菜单）

导入（菜单-运行系统数据）

该菜单命令用于将数据记录（运行系统数据）导入到所选的用户归档中。

在导入的文件中，没有关于数据类型和列数的信息。 因此，导入数据和目标归档的结构必须相等，或在事先导出数据的用户归档中完成导入。

在导出期间，将数据记录 ID 输入导出数据，以便在导入期间对导入的数据实现唯一分配。 如果在导入期间，WinCC 了解到用户归档中已存在将要导入的任一 ID，则将产生一个错误消息，并在日志文件“UALogFile.txt”中建立条目以指定相关的 ID。

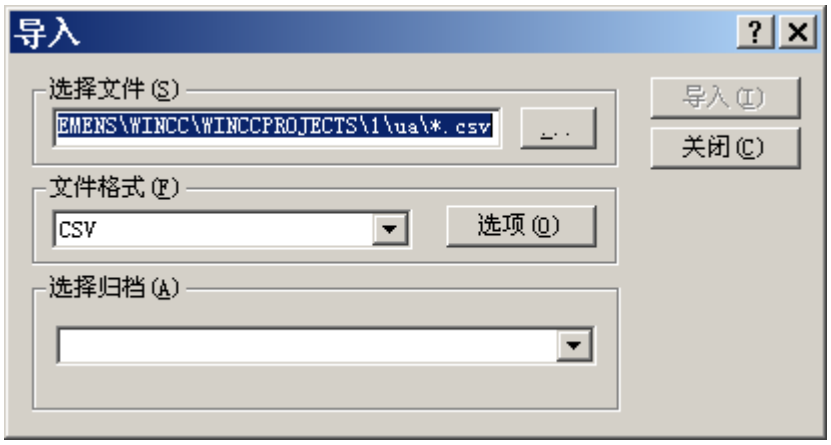
将具有新数据记录 ID 的数据作为新数据记录添加到用户归档。

说明

如果要导入的数据在 WinCC 外编辑并来自当前用户归档，且该数据将覆盖现有归档数据，则首先必须删除该归档的所有数据记录。 否则在导入期间将会由于相同的数据记录 ID 而出现错误消息。

要导入用户归档的结构和视图结构，必须使用“项目”菜单中的菜单命令“导入”。

如果启用了功能“运行系统数据”，则将禁用此菜单项目（菜单“编辑”）。



在域“文件选择”中，输入将要导入的用户归档的路径和文件信息。 按钮“...”"可在选择文件时提供支持。 将根据活动用户归档的项目路径中的文件夹“ua”来自动设置文件路径。

在“文件格式”域中，可以选择将要从中读取的用户归档的文件格式。 使用“选项”按钮以指定所需的分隔符标记。 缺省分隔符为分号“;”。

在“归档选择”域中，选择当前项目的任一用户归档作为目标归档。 选择后，将启用“导入”按钮。

使用按钮“导入”后，将执行导入。

说明

对于客户机-服务器项目，必须考虑以下内容： 如果服务器上有用户归档（例如在“c:\Projects\Test\UA”下），通过此指定路径则可启用它。 客户机通过网络驱动器（例如“I:\Test\UA”）来映射该激活。 此后，在客户机上用户归档的标准路径为“I:\Test\UA”。 但是，在服务器上符合此描述的目录不存在。 如果要将用户归档导入/导出客户机，则必须改变客户机上的标准路径，在本实例中改为“C:\Projects\Test\UA”。

菜单命令“导出”（“运行系统数据”菜单）

导出（菜单运行系统数据）

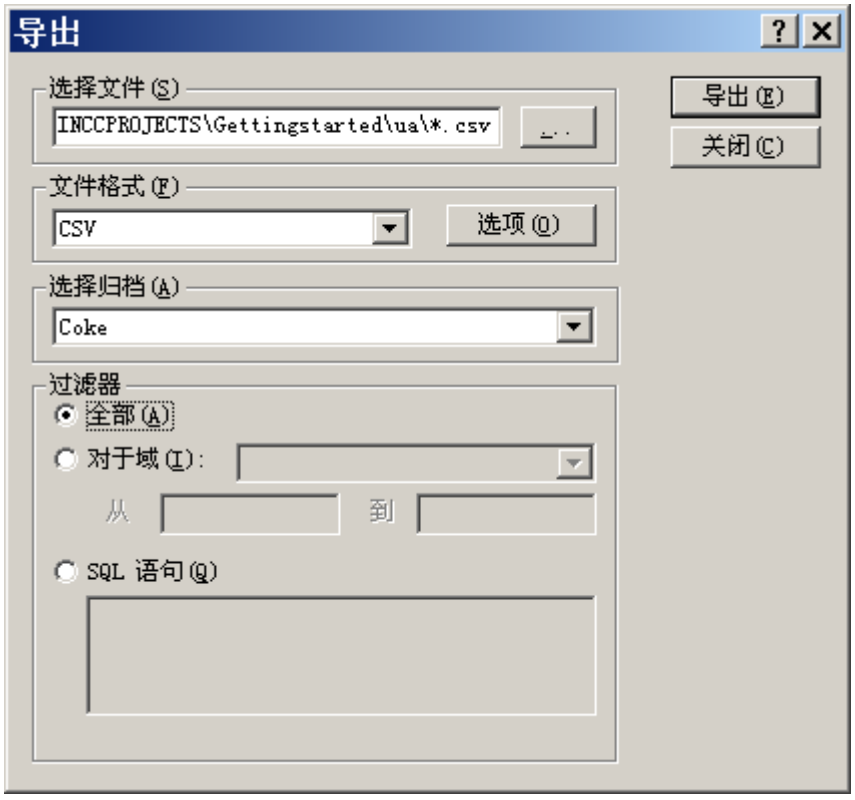
使用该菜单命令以导出所选用户归档的数据记录（运行系统数据）。 可以在其它用户程序（如 MS-Excel）中编辑导出的数据，然后再导入用户归档中。

说明

如果启用了功能“运行系统数据”，则将禁用此菜单项目（菜单“编辑”）。

要导出用户归档的结构和视图结构，必须使用“项目”菜单中的菜单命令“导出”。

在 WinCC V5.1 后，还将导出列标题，且不能更改。



在域“文件选择”中，输入将要导出的用户归档的路径和文件信息。 按钮“...”"可在选择文件时提供支持。 将根据活动用户归档的项目路径中的文件夹“ua”来自动设置文件路径。

在“文件格式”域中，可选择将要导出的用户归档的文件格式。 使用“选项”按钮以指定所需的分隔符标记。 缺省分隔符为分号“;”。

在“归档选择”域中，选择当前项目的任一用户归档作为目标归档。 选择后，将启用“导出”按钮。

在“过滤器”区域中，如有必要，可以指定用于导出用户归档的过滤器。 在“域的过滤器”域中指定过滤器将要采用的域。 在“从...到”域中，将显示将通过过滤器的数值范围。

启用选项按钮“SQL 语句”后，可在其下的输入域中输入 SQL 语言的过滤语句。 有关 SQL 语句的更多信息，请参阅附录。

使用按钮“导出”后，将执行导出。

在导出期间，将数据记录 ID 输入导出文件，以便在导入期间对导入的数据实现唯一分配。

说明

对于客户机-服务器项目，必须考虑以下内容： 如果服务器上有用户归档（例如在“c:\Projects\Test\UA”下），通过此指定路径则可启用它。 客户机通过网络驱动器（例如“I:\Test\UA”）来映射该激活。 此后，在客户机上用户归档的标准路径为“I:\Test\UA”。 但是，在服务器上符合此描述的目录不存在。 如果想要导出用户归档数据，则必须改变客户机上的标准路径，在本实例中改为“C:\Projects\Test\UA”。

2.2.2.3 “用户归档”编辑器的工具栏

工具栏

可在工具栏中直接单击鼠标来操作“用户归档”编辑器。下面介绍工具栏上的各个符号（按字母顺序排列）。

符号	描述
	浏览
	属性
	导出
	帮助
	导入

	删除
	新建
	运行系统数据
	保存
	恢复

浏览

“浏览”符号允许在运行期间快速浏览用户归档。

属性

使用“属性”符号可编辑用户归档或数据域的属性。使用鼠标右键单击用户归档或数据域还可以编辑用户归档或数据域的属性。

在运行系统中使用光标控制键在表格内编辑数据域时，只可在已激活“属性”符号的栏内移动，此处的域可以立即进行编辑。只有在选择数据域后，才可启用“属性”按钮。

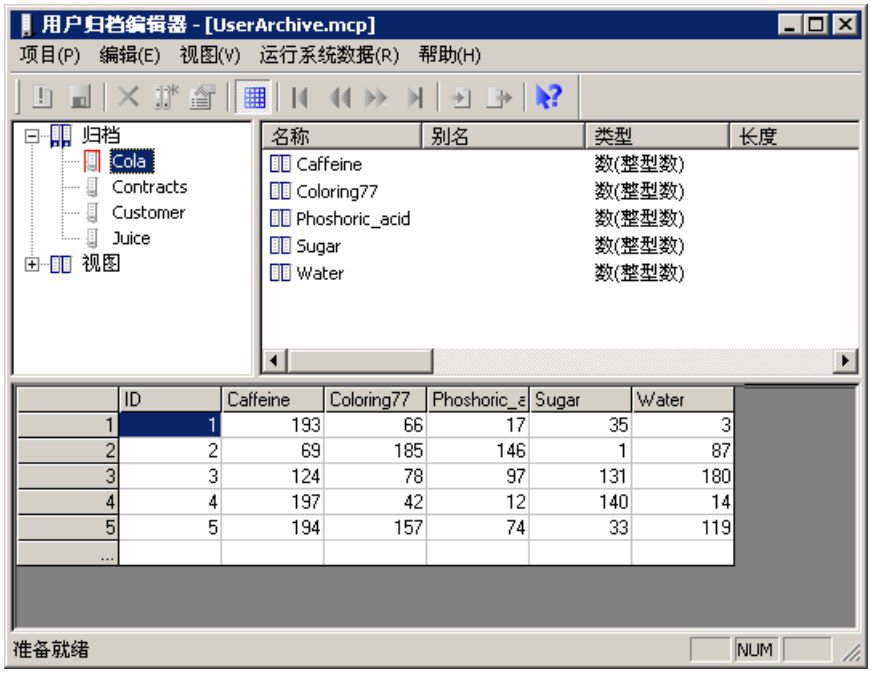
删除

使用“删除”符号可删除用户归档或数据域。使用鼠标右键单击用户归档或数据域还可以删除归档或数据域。此外，还可以单击用户归档或数据域，然后按“Delete”键。

新建

使用“新建”符号可创建新的用户归档或数据域。在任一顶部窗口中使用鼠标右键单击还可以创建新的用户归档。

2.2.2.4 “用户归档”编辑器的表格窗口



可以使用菜单项“编辑 - 运行系统数据”或相应的按钮来打开或关闭表格域。可以双击其中一个域来启用数据输入。这由文本光标来标记。 在运行系统中使用光标控制键在表格内编辑数据域时，只可在已激活“属性”符号的栏内移动，此处的域可以立即进行编辑。只有在选择数据域后，才可启用“属性”按钮。也可通过弹出式菜单在表格域中使用编辑功能。为了将数据记录复制到外部程序，可选择所需的表格行并使用组合键“Ctrl”+“c”将其复制到剪贴板。使用快捷键“Ctrl”+“v”可将复制的内容粘贴到外部程序中。 此方法无法将外部数据粘贴到“用户归档”编辑器的表格窗口中。

说明

如果在“用户归档”编辑器的表格域或用户归档控件表格中更改了一个或多个值，则必须在输入后退出该数据记录（通过单击其它表格单元格或行），以使该数值被数据库接受并在所有显示中更新。

2.2.3 组态

2.2.3.1 组态

第一步是组态新用户归档。可以使用向导进行组态，向导提供了轻松的、用户控制的方法。下列组态步骤是必需的：

组态用户归档

- 创建用户归档
- 设置用户归档域

组态视图

- 创建视图
- 设置数据域
- 设置关系

WinCC 用户归档控件的组态

- 创建用户归档控件

2.2.3.2 组态用户归档

创建新用户归档

创建新用户归档

从 WinCC 项目管理器启动用户归档编辑器。

- 要执行该操作，单击用户归档，然后在弹出式菜单中选择菜单项“打开”。 屏幕上将显示用户归档编辑器的命令界面。

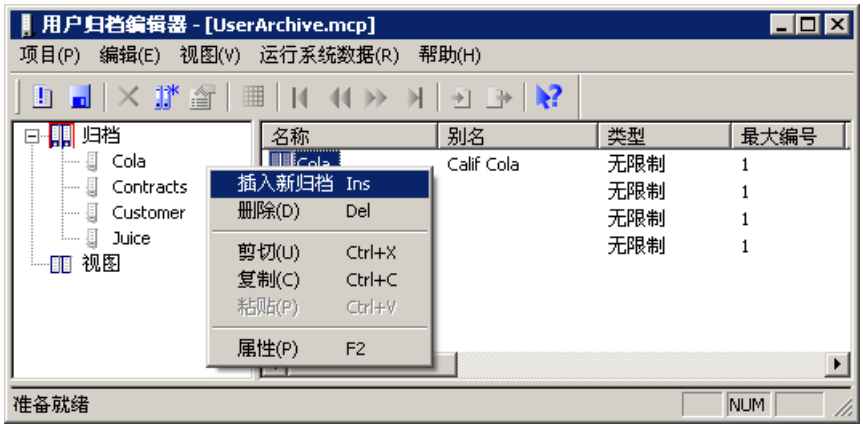
说明

每个用户归档可创建 500 个域。

缺省选项“Put archive in cycle”（循环归档）允许连续创建多个用户归档。 如果只需创建一个用户归档，可在“编辑 - 选项”菜单中禁用该选项。

按照以下步骤创建用户归档。

1. 在浏览窗口中使用鼠标左键单击“归档”。
2. 然后使用鼠标右键单击浏览或数据窗口。 将出现以下画面中显示的弹出式菜单。



3. 选择“新的归档”选项

将显示组态用户归档向导。

用户归档域的常规属性

指定用户归档域的常规属性

在“常规”对话框中，将确定要创建的用户归档域以及域类型。 对于域类型“字符串”，还可进一步指定域的长度。

- 在“域名”输入域中，可以输入首个用户归档域的名称；本实例中，输入的是首个配方成份“Water”（水）。
- 在别名域中，可以为该域指定第二个名称，以便对该域进行注释，或通过文本库改变运行系统中显示的语言。 该输入是可选的。
  – 有关语言切换的更多信息，请参阅在线帮助。

输入的名称用于将来表格视图域的分配。

在“类型”输入域中，可以指定下列任一种变量类型：

- 整型 有符号 32 位数
- 浮点 浮点数 32 位 IEEE 754
- 双精度 浮点数 64 位 IEEE 754
- 字符串 文本变量，8 位字符集
- 日期/时间 无特定数据类型可用

说明

对于“日期/时间”变量类型，日期和时间的输入格式取决于操作系统的设置。

参见

改变语言 (页 355)

通讯设置

通讯设置

在“通讯”对话框中，可以设置 PLC 和用户归档之间的连接类型：

在“类型”条目下，可以指定通讯类型：

- **无：** 不能进行通讯
- **通过原始数据变量：** 通过原始数据变量访问 AS。
- **通过 WinCC 变量：** 通过 WinCC 变量访问 AS

要通过原始数据变量获得链接，单击“通过原始数据变量”。 输入“PLCID”作为归档标识。 “PLCID”包含 8 个 ASCII 字符，在 WinCC 项目内部是唯一的。 它能标识相应的用户归档，并且是 AS 将过程画面数据返回给正确用户归档的先决条件。

如果选择了“通过原始数据变量”，可以单击“选择”并选择原始数据变量。

如果选择了通过 WinCC 变量进行通讯，则在用户归档域的属性对话框中分配变量。

说明

对于通过原始数据变量进行通讯，会将完整的数据记录连接到原始数据变量。 如果使用了 WinCC 变量，则会将变量连接到用户归档域。

对于通过原始数据变量进行用户归档通讯，PLCID 用作归档的唯一名称。 不能使用在所使用原始数据变量中组态的 R_ID，因为它只与 AS 进行的通讯有关。 此外，相同的原始数据变量可以提供给多个用户归档。

设置控制变量

设置控制变量

在“控制变量”标签上，将控制变量设置为 WinCC 变量形式，可用于访问用户归档域。

在该标签的四个输入域内设置 WinCC 变量，可用于访问数据记录 ID、作业代码、归档域和归档域值。

每个输入域旁都有一个“选择”按钮，可用于打开变量选择对话框。 此处可以显示和选择所有存在的 WinCC 变量。

使用“创建...”按钮可自动创建变量。 可以使用该按钮创建新的变量组“@UA[Archive name]”，并将所创建的变量以 @UA[Archive name]ID、@UA[Archive name]Job 等形式保存。

说明

为了检查该功能，必须始终通过向导设置或创建用户归档的全部四个控制变量。
不得更改控制变量的数据类型。
创建完用户归档后，建议使用向导创建和保存控制变量。

利用这四个控制变量，可以触发用户归档。 要触发用户归档，必须为“ID”和“Job”变量或“Job”、“Field”和“Value”变量提供相应的值。

有关用户归档 PLC 的更多信息，请参阅章节“控制变量的属性”中的控制变量。

例如，如果不想使用控制变量，可以关闭对话框而不输入任何内容。 可以在章节“使用控制变量的实例”中找到使用控制变量的实例。

参见

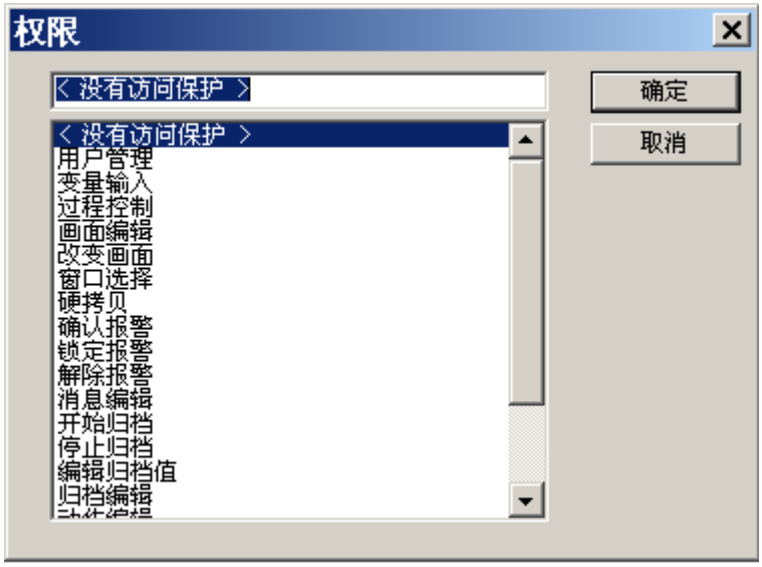
使用控制变量的实例 (页 339)

设置权限和标记

设置权限和标记

“权限和标记”对话框用于设置对用户归档的访问权限，以及单独列中上次访问/上个用户的输出设置。

显示当前设置的所有权限以便读写所有访问。 要更改这些设置，可以单击其中一个“选择”按钮。 然后会出现“权限”对话框，可以从中选择在用户管理器中设置的权限：

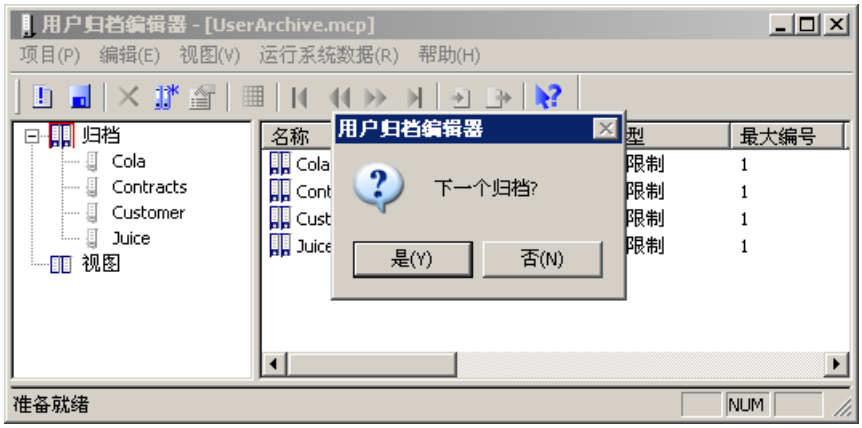


通过激活“域 - 上一次修改”选项来设置包含上次访问日期和时间的列。 使用“域 - 上个用户”选项可以设置列出了上次访问用户归档的用户的列。

1. 选择其中一个权限
2. 例如，选择“上个用户”域。
3. 使用“完成”键可以退出创建用户归档。

创建完用户归档后，会出现“是否添加域？”的询问对话框。 如果选择“是”确认，将打开用于创建用户归档域的“常规”对话框。 有关如何创建用户归档域的信息，请参阅章节“创建用户归档域”。

如果已在“编辑 - 选项”菜单中激活了“Create archives in cycle”（循环创建归档）选项，则在创建完域后会出现“下一个归档？”询问对话框。 如果单击“是”，将出现用于输入下一个用户归档的“常规”初始对话框。



最后，通过鼠标单击保存符号或通过激活“项目 - 保存”菜单来保存新用户归档。

说明

对用户归档所进行的更改只有通过“保存”才能存入数据库。 如果要通过“冗余”选项对齐用户归档，必须激活“上次访问”标记。

本实例中，“Cola”用户归档的属性如下：

用户 归档	**属性**
Cola	名称： Cola
	别名： Calif Cola
	类型： 无限制
	最多记录： 1
	通讯类型： 原始
	PLCID： S7112
	变量名： CalifVarGroup
	读权限： 0
	写权限： 0
	标记： U
	编号： 3
	上次访问： 03/05/98 12:54

参见

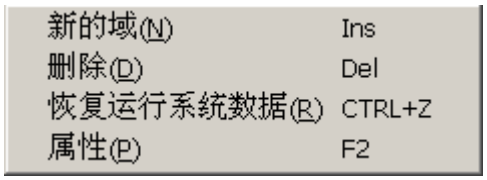
创建新用户归档域 (页 328)

创建新用户归档域

本节将介绍如何创建用户归档数据域。

缺省选项“Put archive in cycle”（循环归档）允许连续创建多个域。 如果只创建一个域，可在“编辑 - 选项”菜单中禁用该选项。

1. 在浏览窗口中扩展“归档”（单击“+”号）。 然后，新用户归档“Cola”将显示在浏览窗口中。
2. 在浏览窗口中使用鼠标左键单击“Cola”用户归档。 将显示以下弹出式菜单：



1. 单击菜单项“新的域”。

将显示“常规”对话框。

说明

在下列情况下，对用户归档域数据所进行的更改将会丢失：
如果对于已存在的数据，无法再满足新的一致性条件，例如“唯一”、“非空”等。
如果已经重命名域名。
新的数据类型无法再转换来自数据源的数据时。
有关更改用户归档的更多信息，请参见章节“更改用户归档的组态”。

参见

修改用户归档的组态 (页 343)

用户归档的常规属性

设置用户归档的常规属性

以下是“常规”对话框，可通过该对话框创建新用户归档。

例如输入“Cola”作为用户归档的名称。 在“别名”域中，可以为用户归档指定第二个名称（例如“Calif Cola”），以便对用户归档进行注释，或通过文本库改变运行系统中显示的语言。 该输入是可选的。

有关语言切换的更多信息，请参阅在线帮助。

如果将归档类型指定为“限制”，则可以在“记录”域中设置数据记录的最大数目。 类型“无限制”可创建不限制数据记录数目的用户归档。

说明

数据库语言 SQL 的关键字（或保留词）不可用作归档或域的名称。 更多详细信息，请参阅章节“按字母顺序排列的 SQL 关键字列表”。

创建数据记录时，将不会检查这些内容是否完整或正确。

参见

改变语言 (页 355)

按字母顺序排列的 SQL 关键字列表 (页 367)

设置用户归档域的值

设置值

在“值”对话框中，可以输入最小值、最大值和起始值。

最小值、最大值和起始值

此处可根据选择的数据类型来设置相应的值。

WinCC 变量

此处可设置保存用户归档域值的 WinCC 变量。 可以...

1. 在输入域中直接输入变量
2. 通过“选择”按钮交互式选择或创建变量
3. 通过“创建”按钮自动创建新的变量
4. 通过“编辑”按钮更改变量的属性

“权限和标记”的设置

设置权限和标记

在“权限和标记”对话框中，可以设置用户归档域的访问权限和属性。

权限

利用“选择”按钮，可以在此设置读写访问的权限。 用户管理器中设置了可能的权限。 按照章节“创建用户归档”中所描述的内容来设置访问权限。

标记

在“标记”区域中，可以为所选的数据域设置下列属性：

1. “域内须有值”：
   域中具有非零值。
2. “域内须有具唯一性的值”：
   域内须有具唯一性的值，即该列的值必须互不相同。
3. “索引应支持域”：
   如果可能，域支持索引值。 该索引可以提高搜索命令的性能。
4. 单击“完成”按钮退出数据域输入。
   现在将在“Cola”用户归档中创建新的数据域。
   如果已在“编辑 - 选项”菜单中激活了“Create fields in cycle”（循环创建域）选项，则在创建完域后会出现“下一个归档？”询问对话框。 如果单击“是”，将出现用于输入下一个域的“常规”初始对话框。
5. 保存新用户归档

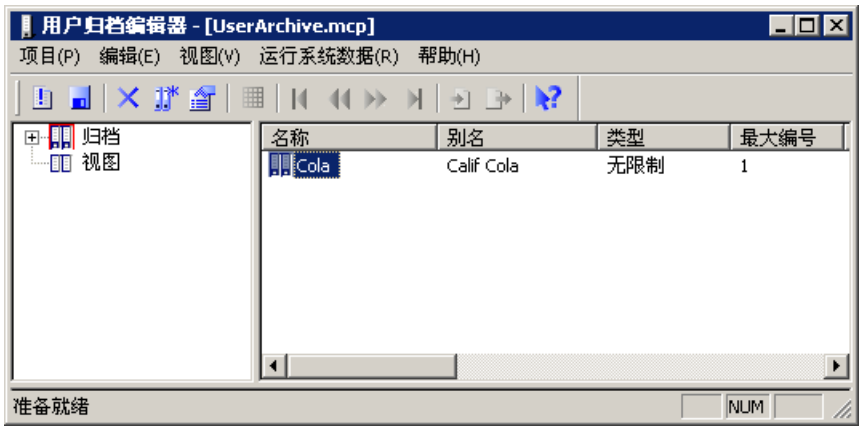
说明

对用户归档所进行的更改只有通过“保存”才能存入数据库。

用户归档的属性

要编辑窗体的属性...

- 在浏览窗口中使用鼠标右键单击“Cola”用户归档（首先扩展归档）。
- 在弹出式菜单中选择“属性”。



出现“归档属性”对话框，可以在其中更改属性。 “常规”、“通讯”、“标记”和“选择权限”标签已在“创建用户归档”章节进行了描述。 附加的“顺序”标签用于设置用户归档的顺序。

用户归档： “顺序”标签

“顺序”标签

“顺序”标签用于设置用户归档的顺序。

选择一个或多个用户归档，然后使用“上”和“下”键移动其位置。 使用“确定”按钮对输入进行确认。 最后，通过鼠标单击保存符号或通过激活“项目 - 保存”菜单来保存用户归档。 然后，用户归档的顺序会显示在用户归档编辑器的“编号”列中。

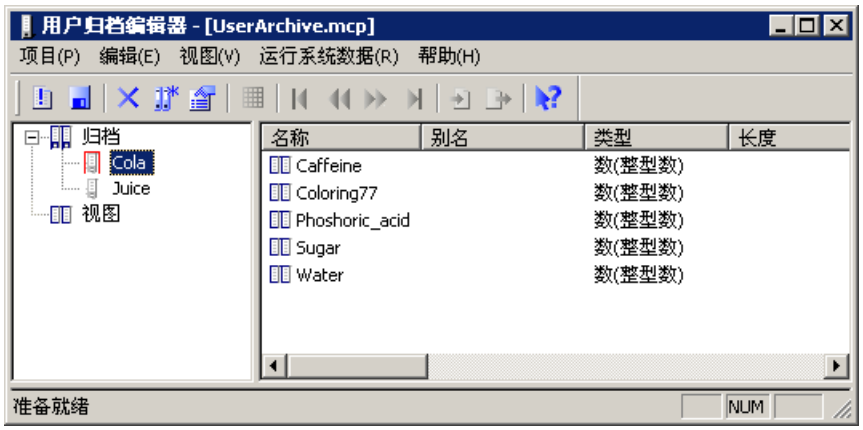
说明

对用户归档所进行的更改只有通过“保存”才能存入数据库。

用户归档域的属性

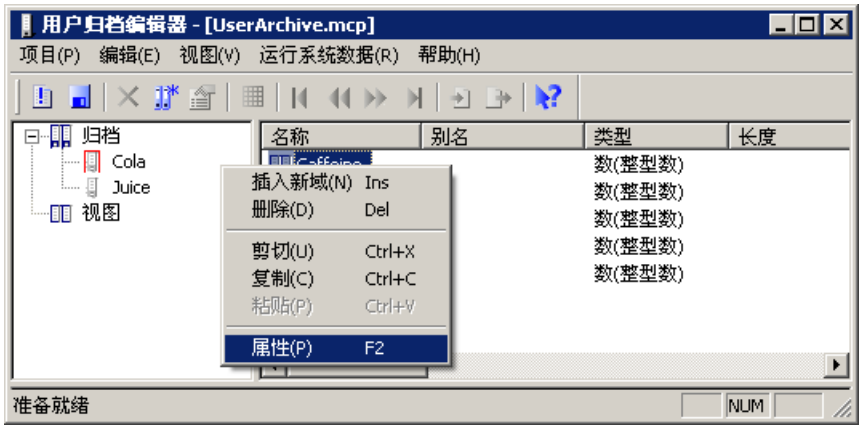
要编辑数据域的属性...

- 单击“Cola”用户归档的浏览窗口（首先扩展归档）。
- 在用户归档编辑器的数据窗口中，应显示“Cola”用户归档的数据域：



要编辑用户归档的数据域...

- 在用户归档编辑器的数据窗口中，单击域名“Water”
- 在弹出式菜单中单击“属性”按钮



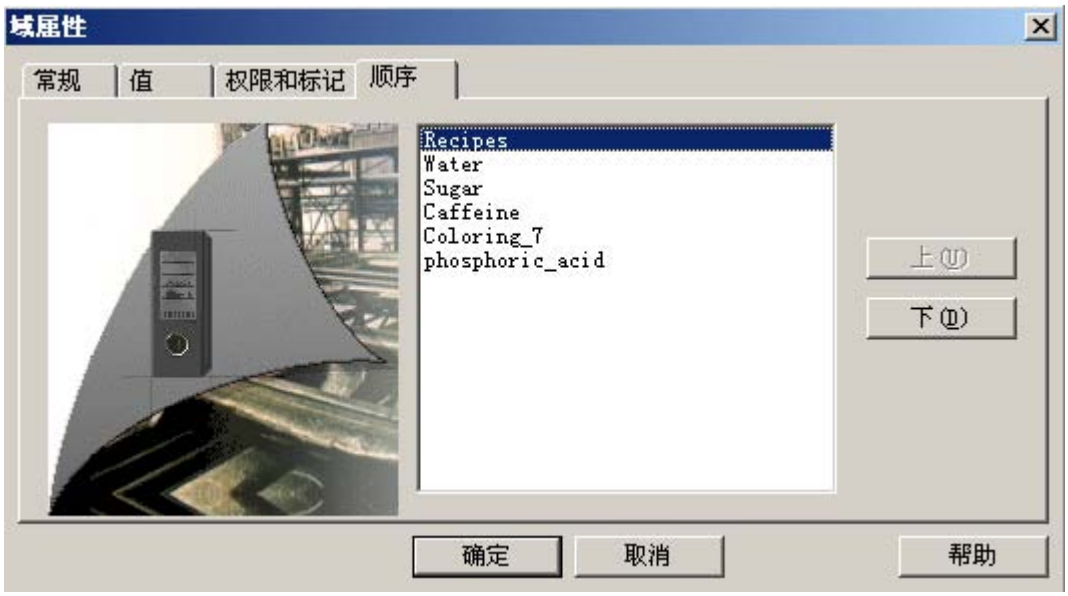
出现“域属性”对话框，可以在其中更改数据域的属性。

“常规”、“值”、“标记”和“选择权限”标签已在“定义用户归档域”章节进行了描述。 附加的“顺序”标签用于设置用户归档域的顺序。

用户归档域： “顺序”标签

“顺序”标签

“顺序”标签用于设置数据域的顺序。 此处设置的顺序会影响用户归档编辑器表格窗口和运行系统画面控件中数据的显示，并影响通过 WinCC 脚本语言函数进行访问的索引的分配。



选择一个或多个域，然后使用“上”和“下”键移动其位置。 使用“确定”按钮对输入进行确认。 最后，通过鼠标单击保存符号或通过激活“项目 - 保存”菜单来保存用户归档。 然后，用户归档域的顺序会显示在用户归档编辑器的“编号”列中。

本实例中，“Cola”用户归档包含下列属性：

用户归档	数据域	属性
Cola	Water（水）	名称： Water
		别名： Wasser_aus_Brunnen_4
		类型：整型
		长度：
		精度：
		最小值： 1000
		最大值： 1200
		起始值： 1100

		变量 n...：
		权限（读）： 0
		权限（写）： 0
		标记： NN
		P...： 3
		上次访问： 03/05/98 12:54
	Sugar（糖）	名称
		别名
		....
	Dye 7（色素 7）	名称
		别名
		....
	Caffeine（咖啡因）	名称
		别名
		....
	Phosphoric acid（磷酸）	名称
		别名
		....

最后保存用户归档。

说明

对用户归档所进行的更改只有通过“保存”才能存入数据库。

控制变量的属性

“变量属性”对话框

可以使用用户归档和用户归档域的属性对话框来编辑控制变量的属性。 单击工具栏上相应标签中的“编辑”按钮。 将显示“变量属性”对话框，可以根据需要控制和更改变量的属性。

控制变量的数据类型	
@UA_Cola_ID 的数据类型	有符号 32 位数
@UA_Cola_Job 的数据类型	有符号 32 位数
UA_Cola_Field 的数据类型	文本变量，8 位
@UA_Cola_Value 的数据类型	文本变量，8 位

说明

不得更改控制变量的数据类型。

利用这四个控制变量，可以触发用户归档。 要触发用户归档，必须为“ID”和“Job”变量或“Job”、“Field”和“Value”变量提供相应的值。

控制变量的功能	
ID	用户归档的 ID（对应于记录编号）
作业	可以进行三种作业： 读、写和删除： 读 = 6 写 = 7 删除 = 8 执行作业后，该控制变量将包含一个错误 ID： 无错 = 0 错误 = -1
数组	归档域
数值	归档域值

控制变量“ID”和“Job”的更多值组合

ID	Job = 6	Job = 7	Job = 8
-1	添加数据记录	-	删除带最低 ID 的数据记录

-6	读取带最低 ID 的数据记录	写入带最低 ID 的数据记录	删除带最低 ID 的数据记录
-9	读取带最高 ID 的数据记录	写入带最高 ID 的数据记录	删除带最高 ID 的数据记录

控制变量提供了两种访问用户归档的方法：

1. 通过输入控制变量“ID”和“Job”，可以写入或读取或删除数据记录中的目标值。
2. 可以使用控制变量“Field”和“Value”代替控制变量“ID”来搜索数据记录。 通过控制变量“Job”，可以写入或读取或删除通过该方法选择的数据记录。 例如，如果必须将数据记录从表格中删除然后将其再添加到表格末尾，则可以使用该数据选择类型。 “值”域必须唯一，否则将采用域中满足该条件值的第一条数据记录。

说明

为了检查该功能，必须始终通过向导设置或创建用户归档的全部四个控制变量。

创建完用户归档后，建议使用向导创建和保存控制变量。

有关如何提供控制变量的更多信息，请参阅使用控制变量的实例。

参见

使用控制变量的实例 (页 339)

使用控制变量的实例

使用控制变量的实例：

为了能够在实例中使用控制变量进行工作，必须执行以下步骤：

在“用户归档”编辑器中

1. 创建用户归档（本实例中为“Cola”用户归档）。利用向导创建用户归档时，请输入以下执行的设置。如果已经创建项目“Cola”，则可以通过用户归档的属性来检查设置，并可在必要时更改设置。

“Cola”用户归档的属性	
用户归档类型	“无限制”
通讯	通过 WinCC 变量
控制变量	创建
变量组	“@UA_Cola”
@UA_Cola_ID 的数据类型	有符号 32 位数

@UA_Cola_Job 的数据类型	有符号 32 位数
@UA_Cola_Field 的数据类型	文本变量，8 位
@UA_Cola_Value 的数据类型	文本变量，8 位

1. 在用户归档中创建数据域“Water”、“Sugar”、“Dye stuff 7”、“Caffeine”和“Phosphoric acid”（整型）。
2. 创建数据域“Recipes”（字符串类型）。

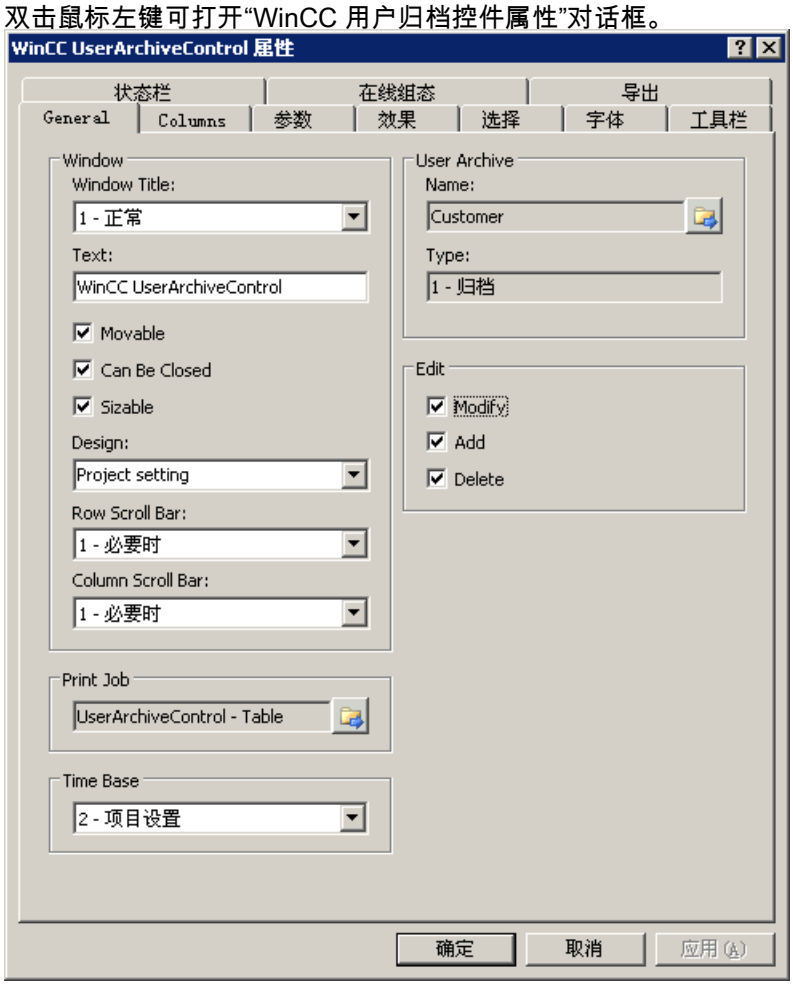
在图形编辑器中

1. 打开新画面并创建 WinCC 用户归档控件。

说明

在 WinCC V7 之前，用户归档的显示在用户归档表格元素中组态。

2. 双击鼠标左键可打开“WinCC 用户归档控件属性”对话框。



3. 更改下列设置：
   – 在“常规”选项卡的“用户归档”区域中选择用户归档“Cola”。
   – 在“编辑”域中激活访问类型“更改”、“添加”和“删除”。
   – 可以选定其它设置，而不作任何改变。 如果已创建用户归档控件，则可以使用属性对话框检查设置，并在必要时更改设置。
4. 为这四个控制变量中的每一个变量都创建一个 I/O 域，然后选择下列设置：

控制变量	数据格式	输出格式
@UA_Cola_ID	十进制	0999
@UA_Cola_Job	十进制	s9
@UA_Cola_Field	字符串	*
@UA_Cola_Value	字符串	*

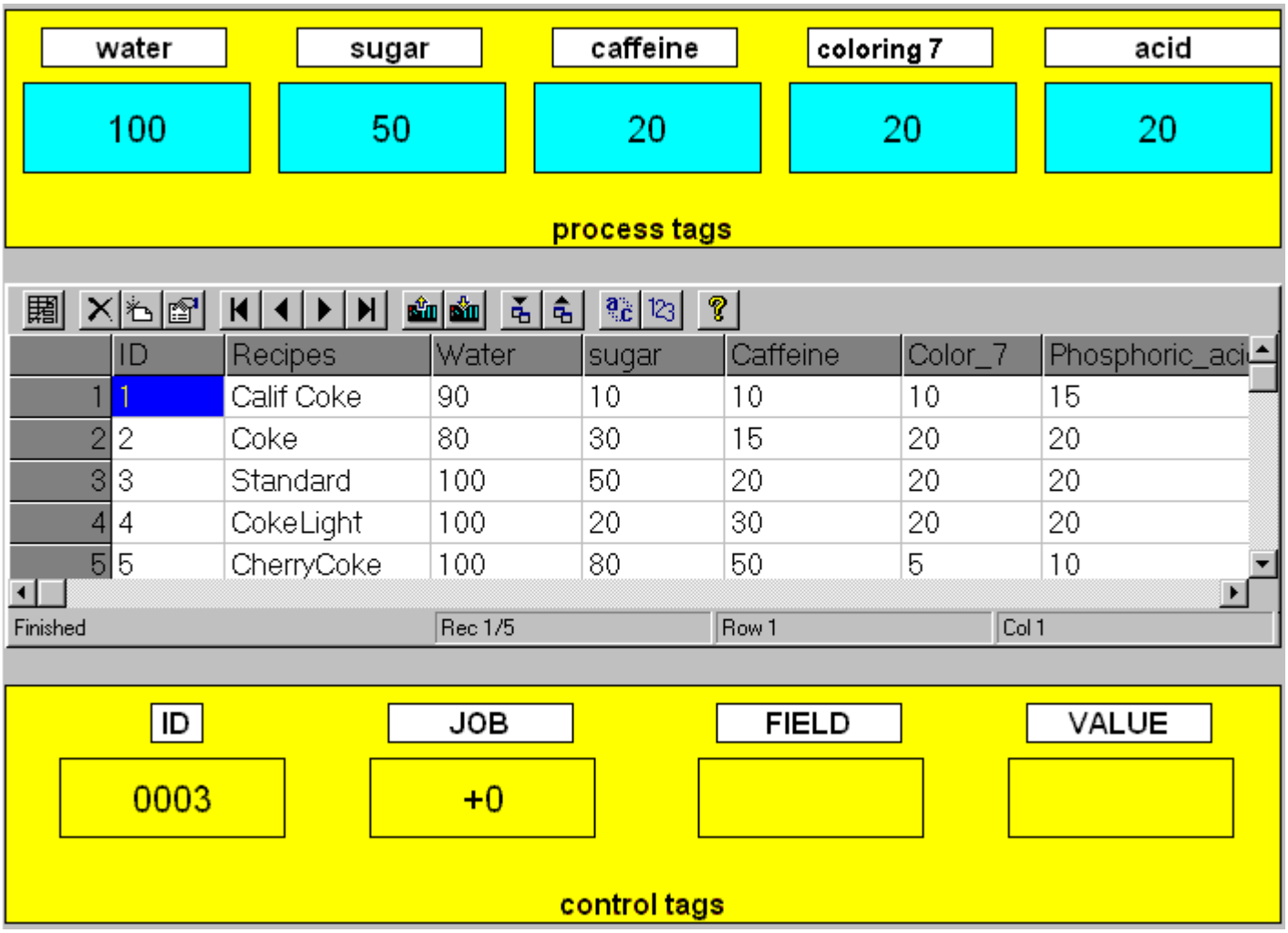
1. 为每一个变量选择对象属性“根据变化更新”。
2. 为已组态的每个数据域（Water、sugar 等）创建 I/O 域，并将其连接到相关变量（例如将“Water”的 I/O 域连接到过程变量“@UA_Cola_Water”）。为每一个变量选择对象属性“根据变化更新”。

说明

有关组态 I/O 域的更多信息，请参考图形编辑器的文档。

为已组态的每个 I/O 域创建用作标签的文本域，以便可在运行系统中分配各个域。保存输入的内容并激活 WinCC 运行系统。现在在表格窗口中输入五条数据记录。在 ID 为 2 的数

据记录的配方列中输入“Cola”，在第四条数据记录中输入“Cola Light”。 下图显示了使用 WinCC 用户归档表格元素的实例（WinCC V7 之前版本）：



执行可能的各个动作的步骤。

1. 使用 ID 选择数据记录并写入数据记录的值：

- 在 I/O 域“ID”中输入 ID“3”，并在 I/O 域“Job”中写入 7（写）。
- 现在，数据记录“3”的值便显示在过程变量的 I/O 域中。

- 如果操作成功，错误号“0”会显示在 I/O 域“Job”中。如果发生错误，则会显示错误号“-1”。
- 不需输入控制变量“Field”和“Value”。

说明

通过输入 ID“-1”和作业“6”，过程变量的当前内容将被读入表格。新的值将被添加到表格末尾；记录的 ID 也会随之增加。可以在“控制变量的属性”章节中找到控制变量“ID”和“Job”的其它值组合。

1. 使用 ID 选择数据记录并读取数据记录的值：

- 更改过程变量 I/O 域中的值，然后在“ID”域中输入“5”。在 I/O 域“Job”中，输入“6”（读）。
- 更改后的过程变量值将被写入数据记录“5”。该数据记录中之前所包含的值将被覆盖。
- 不需输入控制变量“Field”和“Value”。

1. 通过控制变量“Field”和“Value”选择数据记录：

- 在 I/O 域“Field”中输入单词“Recipe”，并在 I/O 域“Value”中写入“Cola Light”。在 I/O 域“Job”中，输入“7”（写）。
- 现在，已写入数据记录“Cola Light”，该数据记录的值显示于过程变量的 I/O 域中。
- 不需输入控制变量“ID”，因此必须将其设置为 0。

说明

连接到控制变量“Value”的域必须分配给“权限和标记”对话框中的“域内须有具唯一性的值”标记。否则将无法将数据记录唯一地分配给域内的值。

参见

在图形编辑器中创建 IO 域

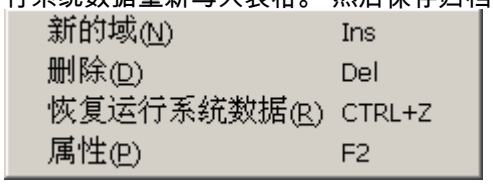
2.2.3.3 修改用户归档的组态

如果要更改或扩展现有的用户归档，则将丢失数据库表格中已存在的数据。 尤其是，如果更改了数据库表格的结构或域属性，则无法再满足数据库的一致性条件。

为避免数据丢失，建议使用下列步骤：

1. 禁用运行系统，在用户归档编辑器中打开用户归档，然后执行所需的更改。 只在完成所有的更改后再保存归档（无缓存保存）。
2. 保存后按下“编辑运行系统数据”按钮。 在打开的表格上，可以查看用户归档中的现有数据是否仍然可用。

3. 如果数据仍然存在，则可以使用该归档或执行进一步更改。 如果进行了更改，则在完成每次保存过程后检查数据是否仍然可用。
4. 如果变更后数据没有显示，首先撤消在归档中进行的所有变更而不进行保存。 现在选择用户归档编辑器中的归档，然后在弹出式菜单中选择“恢复 RT 数据”命令。 这样会将先前的运行系统数据重新写入表格。 然后保存归档并检查数据是否已重新写入归档。



说明

完成每次保存过程后，检查运行系统数据是否仍然可用。 如果已完成第一次更改保存步骤且已丢失数据，那么再次保存更改会导致数据丢失。

如果在更改后启动运行系统，然后检测到丢失数据，则仍然可以按上述方法通过“恢复 Rt 数据”命令将数据恢复到表格中。 即使已关闭用户归档编辑器或已退出 WinCC，数据仍然可以恢复。 这种关系的关键点是对每次更改的单独保存。

如果对用户归档进行了全面的更改，则建议导出发生数据丢失前已存在的数据。 通过自定义导出的数据表格可以将数据重新写入更改后的归档。

如果改变服务器上的用户归档组态，则不带其自身项目的客户机将要求更新服务器数据包，以便在运行系统中正确显示数据。 可以在服务器上手动生成更新，或通过使用隐含更新数据包进行。 对于隐含更新数据包，请在“隐含更新”弹出式菜单中，使用服务器的“服务器数据”编辑器来组态设置“当发生改变时，立即生成 WinCC CS 服务器数据”。

参见

菜单命令“导出”（“运行系统数据”菜单） (页 317)

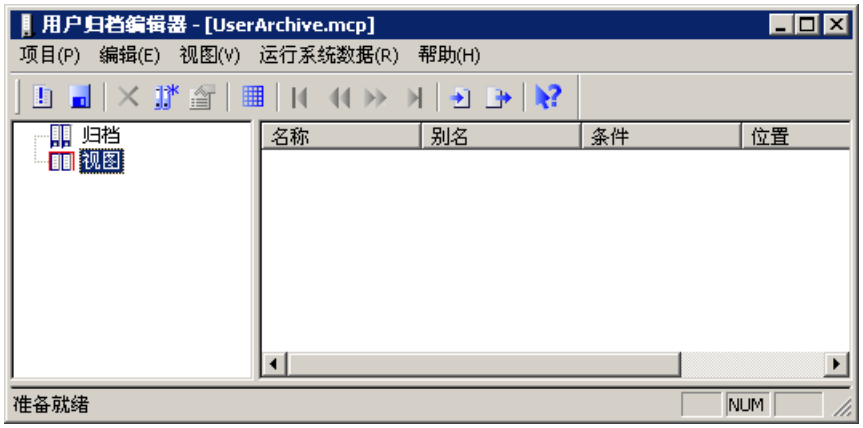
2.2.3.4 组态视图

创建新视图

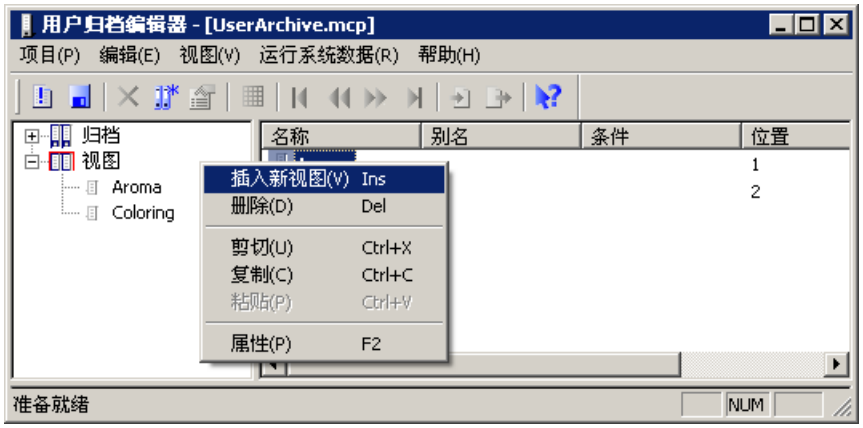
缺省选项“Put views in cycle”（循环视图）允许连续创建多个视图。 如果只创建一个视图，可在“编辑 - 选项”菜单中禁用该选项。

可按照以下步骤来创建新视图：

1. 在浏览窗口中使用鼠标左键单击“视图”。



1. 然后使用鼠标右键单击浏览或数据窗口。 将出现以下画面中显示的弹出式菜单。



1. 选择“新的视图”选项

现在已启动了向导来组态视图。 以下是“常规”对话框，可通过该对话框创建新视图。

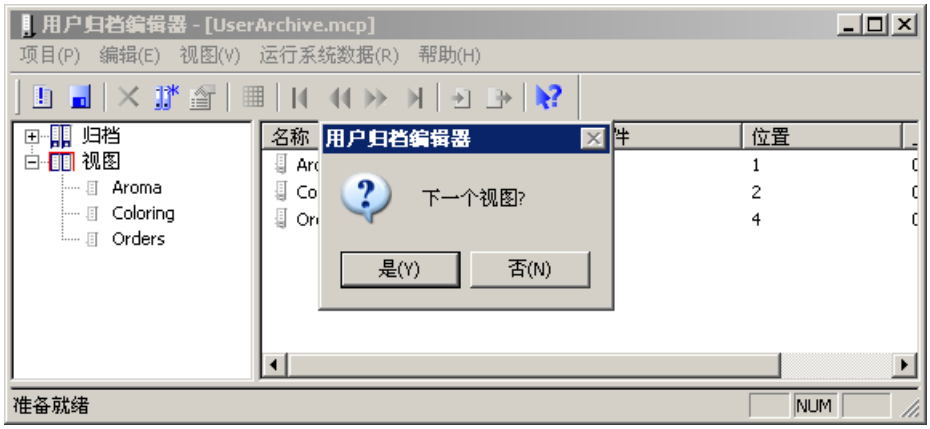
输入实例“Dyes”（色素）作为视图名称。 在“别名”域中，可以为视图指定第二个名称（例如“Dye stuffs in Cola and Orange Juice”（可乐和果汁中的色素））， 以便对用户归档进行注释，或通过文本库改变运行系统中显示的语言。 该输入是可选的。

有关语言切换的更多信息，请参阅在线帮助。

单击“完成”按钮退出创建视图。

创建完视图后，会出现“是否添加列？”的询问对话框。 如果选择“是”确认，将打开用于创建视图的“常规”对话框。 有关如何在视图中创建列的更多信息，请参阅章节“创建视图的列”。

如果已在“编辑 - 选项”菜单中激活了“Create forms in cycle”（循环创建视图）选项，则在创建完视图的列后会出现“下一个视图？”询问对话框。 如果单击“是”，将出现用于输入下一个视图的“常规”初始对话框。



完成后保存视图。

说明

对视图所进行的更改只有通过“保存”才能存入数据库。

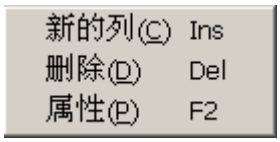
参见

改变语言 (页 355)

创建视图的新列

缺省选项“Put columns of a view in cycle”（循环视图的列）允许连续创建多个列。 如果只创建视图的一个列，可在“编辑 - 选项”菜单中禁用该选项。

1. 在浏览窗口中扩展“视图”。 新的视图（例如“Dyes”（色素））将在随后显示于浏览窗口中。
2. 在浏览窗口中使用鼠标左键单击“Dyes”（色素）视图。 将显示以下弹出式菜单：



1. 单击“新的列”按钮。

将显示“常规”对话框。

视图列的常规属性

视图列的常规属性

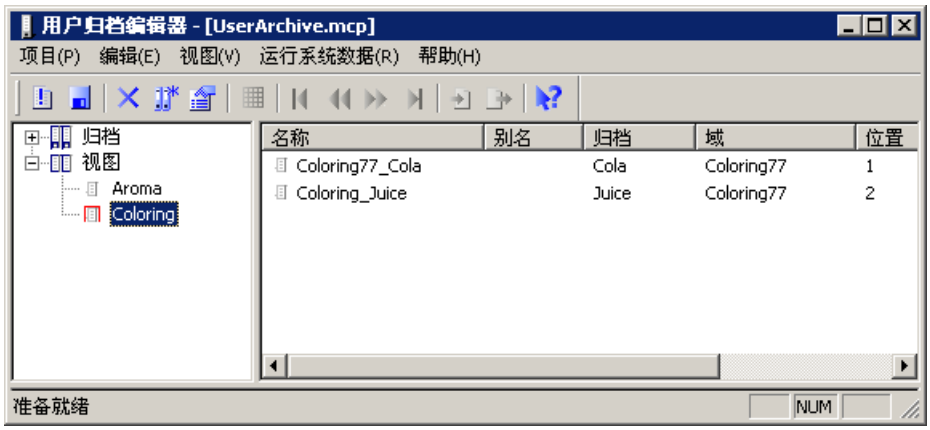
在“常规”对话框中，可以从用户归档选择域，声明所选的域为所创建视图的列，并可以对它们命名。

- 在“归档”选择对话框中，可以选择其中一个设置的用户归档。 例如，保持“Cola”设置而不作任何更改。
- 在“域”选择对话框中，选择 Cola 用户归档中的一个域。 例如，保持“Dye 7”设置而不作任何更改。 使用 TAB 键切换到下一个域，或者通过鼠标单击移动到其它条目。
- 如果单击“列名”域，将接受“域”域中的条目。 也可以自由选择列名，但该名称在视图内必须是唯一的。 例如，接受名称“Dyes”。

在别名域中，可以为该列指定第二个名称，以便对该列进行注释，或通过文本库改变运行系统中的语言。 该输入是可选的。

有关语言切换的更多信息，请参阅在线帮助。

单击“完成”后，便已完成对已组态数据域的设置：



例如在本实例中，通过从“Cola”（可乐）和“Juice”（果汁）用户归档连接数据域“Dye 7”（色素 7）和“Dye 16”（色素 16），饮料生产商创建了一个“Dyes”（色素）视图：

如果已在“编辑 - 选项”菜单中激活了“Create columns of a view in cycle”（循环创建视图的列）选项，则会出现“下一列？”询问对话框。 如果单击“是”，将出现用于输入下一个列的“常规”初始对话框。

最后保存视图的域。

说明

对视图所进行的更改只有通过“保存”才能存入数据库。

参见

改变语言 (页 355)

视图的属性

编辑视图的属性

1. 1. 在用户归档编辑器中，使用鼠标右键单击一个视图
2. 2. 在弹出式菜单中选择“属性”。

随后将显示“视图属性”对话框。

视图的常规属性

视图的常规属性

“常规”标签中将显示所选视图的属性。

可以在“视图名称”域中更改视图的名称，也可以在“别名”域中更改别名。 该对话框中还会显示上次更改的日期和时间。

设置视图的关系

设置视图的关系

在“关系”标签中，可以创建用于视图输出的多个用户归档之间的关系。 可以直接用 SQL 语言编写逻辑操作，或者通过给定的关系运算符交互式进行设置。 确保想要设置关系的用户归档域具有相同的变量类型。

关系

在“关系”域中，可以直接输入 SQL 语句。 有关 的更多信息，请参阅附录。

条件

可以在选择域中交互式输入条件。 要进行此操作，单击左侧和右侧“域”列表中的域，然后通过单击“OP”列表中相应的操作来设置关系。 单击“添加”后，条件会被接受并将随后显示于“关系”域中。

原理

在所选的用户归档中，所有显示关系的域彼此相互连接。 通过使用设置关系，可以过滤域内容，并使结果在运行系统中显示为视图。 还可以在运行系统中编辑视图数据，修改的数据将插入原有归档。

说明

连接的用户归档必须至少显示一个共同因子或关系。

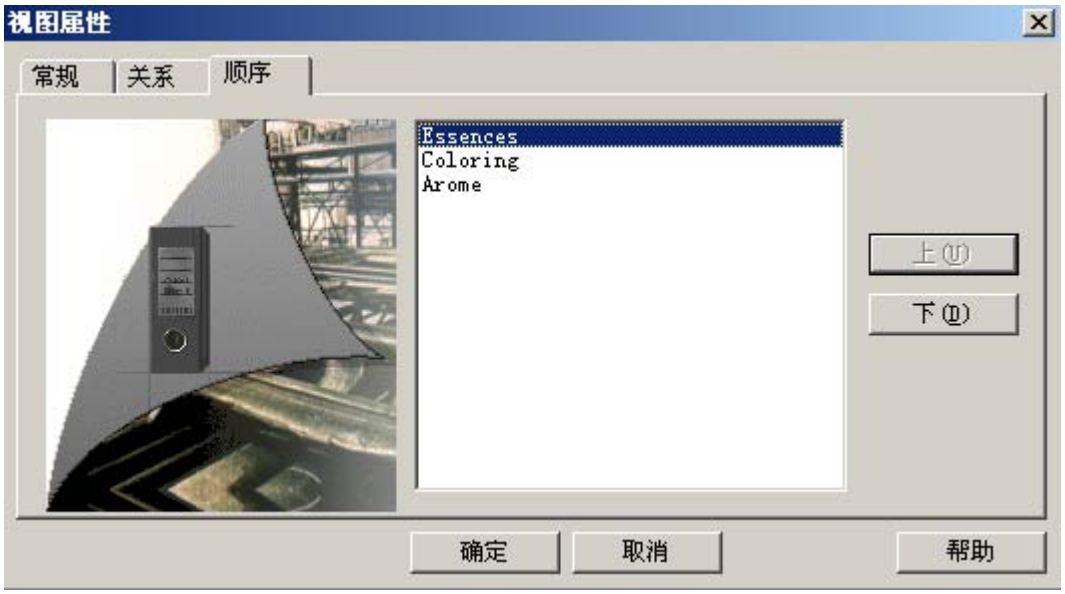
参见

改变语言 (页 355)

设置视图的顺序

设置视图的顺序

在“顺序”标签中，设置视图的顺序。



选择一个或多个视图，然后使用“上”和“下”键移动其位置。 使用“确定”按钮对输入进行确认。 最后，通过鼠标单击保存符号或通过激活“项目 - 保存”菜单来保存视图。 然后，视图的顺序会显示在用户归档编辑器的“编号”列中。

视图列的属性

按照以下步骤编辑视图列的属性：

1. 在用户归档编辑器中，使用鼠标右键单击视图的一个列
2. 在弹出式菜单中选择“属性”。 随后将显示“列属性”对话框：

“常规”标签与新的视图列结构中包含相同的域。 上次更改的日期和时间显示在“已修改”域中。

设置视图列的顺序

设置视图列的顺序

在“顺序”标签中，可以设置视图列的顺序。

选择一个或多个列，然后使用“上”和“下”键移动其位置。 使用“确定”按钮对输入进行确认。 最后，通过鼠标单击保存符号或通过激活“项目 - 保存”菜单来保存视图。 此处设置的顺序会对在用户归档编辑器表格窗口和运行系统画面控件中显示列时产生影响

组态提示

- AS 和用户归档之间的通讯限于每个用户归档一个连接。
- 建立与自动化系统之间的通讯时，PLCID 不得包含超过 8 个字符。
- 包含特殊字符或保留字的短语不可用作域名和表格名。更多详细信息，另请参阅章节“按字母顺序排列的 SQL 关键字列表”。
- 只有当时任何用户归档都没有冗余调整时，才能保存在运行系统中对组态进行的更改。
- 如果使用了 WinCC 冗余，则必须在将被调整的用户归档的两台服务器上使用相同的结构。因此对于这些用户归档的组态，它们的属性和域/记录结构必须完全相同。
- 对于通过原始数据变量进行的用户归档通讯，两台服务器上使用的原始数据变量名也必须完全相同。
- 如果启动“用户归档”编辑器后，工具栏的所有域都变为灰色（恢复除外），则项目路径中的文件“UAEditor.loc”可能已被删除。 这也适用于不再可操作的用户归档控件。
- 如果在启动运行系统时或将用户归档控件切换到运行系统视图时，出现错误消息“连接数据时出错！”，则表示用户归档控件没有连接到用户归档或视图。请检查输入的连接是否正确、组态是否已更改，或者所选的用户归档或视图是否仍然存在。

说明

在组态过程中，用户归档控件连接到选定的用户归档或窗体，并且只能访问该用户归档或窗体。要进行访问，必须启用用户归档/视图（访问保护）。在用户管理器中，可以将特定权限分配给控件。

如果取消了该访问保护，则必须在图形编辑器中将用户归档控件重新连接到用户归档，以便用户归档控件能检测已取消的访问保护。

当打开用户归档控件的画面时，将查询归档或域的访问保护。必须通过对象属性（例如画面、I/O 域或按钮的属性）来对受保护归档的控制变量单独执行访问保护。

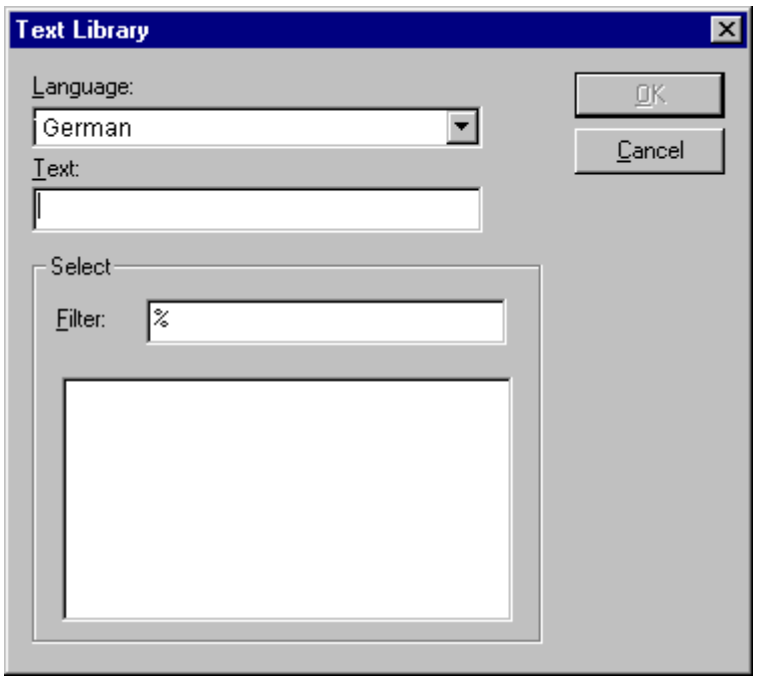
参见

冗余用户归档

按字母顺序排列的 SQL 关键字列表 (页 367)

改变语言

对于用户归档、用户归档域、视图或视图域，可以通过文本库来改变语言。 单击相应对话框中的“文本库...”按钮即可实现。 文本库对话框已打开



用于文本库文本的步骤

1. 首先在“Language”（语言）选择对话框中设置将要进行的组态的语言。
2. 如果已在文本库中创建了计划使用的语言，则可以通过鼠标单击将所有已存在的文本载入选择域。 这些内容显示于选择窗口中，可以选择所需的短语。
3. 选择后，所选择的短语将出现在“Text”（文本）域中。
4. 使用“OK”（确定）确认对话框。
5. 现在，在“别名”域中，该词语的位置号将由文本库给出。

如果运行系统中改变了语言，则将显示在文本库中选择的用于所选语言的短语。

用于新文本的步骤

1. 首先在“Language”（语言）选择对话框中设置将要进行的组态的语言。
2. 在“Text”（文本）域中输入想要改变语言的文本或短语。
3. 使用“OK”（确定）确认对话框。
4. 现在，在“别名”域中，该词语的位置号将根据文本库给出。

5. 打开“文本库”编辑器。 现在，可在此处将用户归档编辑器中输入的文本的翻译输入到所需语言的列中。

ID	英语(美国)	德语(德国)
4	+/-	+/-
5	*	*
6	System, requires acknowledg...	System, quittierpflichtig
7	System, without acknowledgm...	System, ohne Quittierung
8	Alarm	Alarm
9	Warning	Warnung
10	Failure	Fehler
11	Process control system	Leittechnik
12	System messages	Systemmeldungen
13	Operator input messages	Bedienmeldungen
14	Date	Datum
15	Time	Uhrzeit
16	Duration	Dauer
17	Daylight Saving / Standard Time	Sommer- / Winterzeit
18	Status	Zustand
19	Acknowledgment Status	Quittierstatus

6. 翻译完成后关闭文本库。

如果运行系统中改变了语言，则将显示在文本库中选择的用于所选语言的短语。

说明

对于客户机项目，必须在用户归档文本服务器和客户机的文本库中使用相同的文本 ID，否则客户机的运行系统中将显示错误的文本。

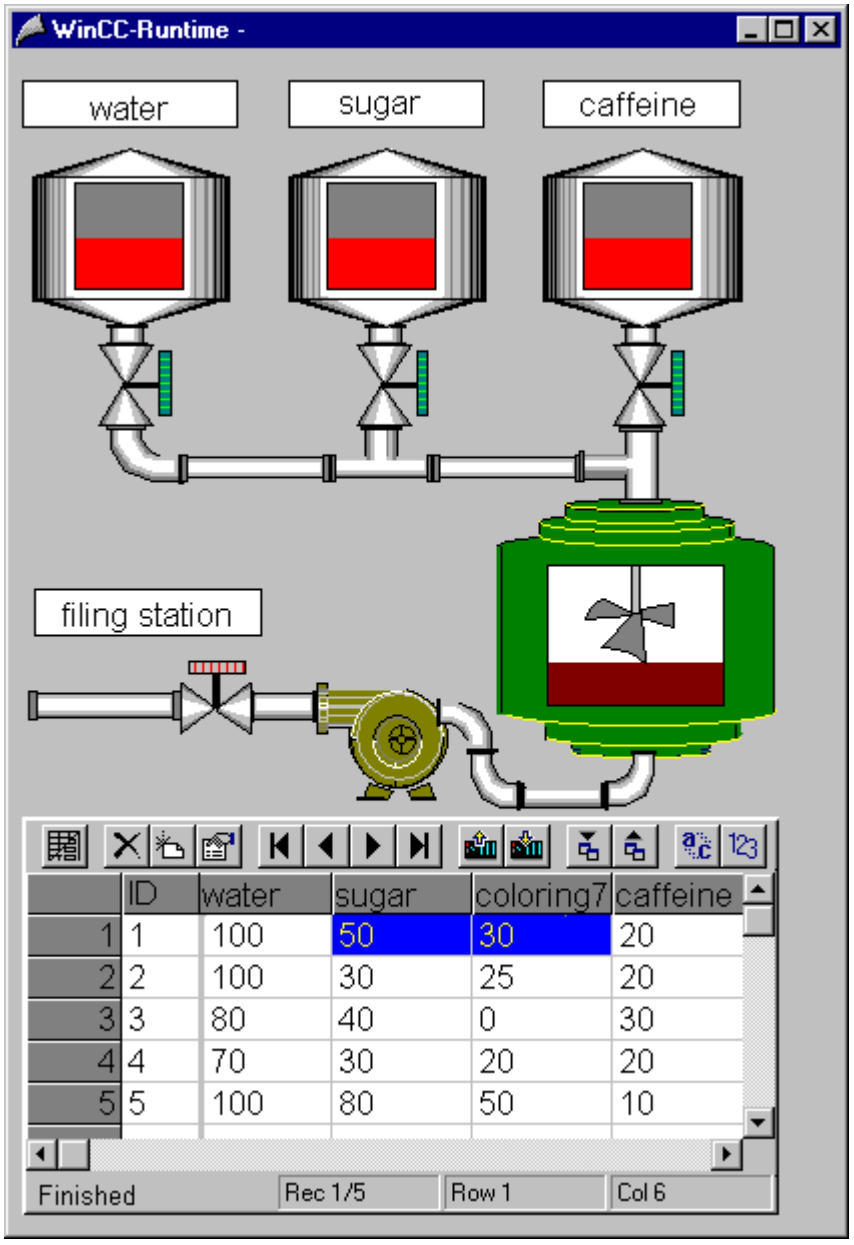
在“Filter”（过滤）域中，可以指定文本库文本的过滤属性，例如“a%”可以列出文本库中所有以字母 a 开头的短语。如果选择了新的过滤标准，必须再次单击选择窗口来更新文本选择。

2.2.3.5 用户归档实例

饮料生产的用户归档

本实例中，饮料生产商“Sun Drink”生产“Calif Cola”可乐饮料和“Sunny Juice”橙汁。 为保存饮料的成份配方，Sun Drink 使用了 WinCC 用户归档。如果灌注系统的存储容器空了，配

方数据将通过每个通讯通道从 WinCC 发送到 AS。随后 AS 可以利用配方数据再次灌注存储容器。 下图显示了使用 WinCC 用户归档表格元素的实例（WinCC V7 之前版本）：



按如下方法使用用户归档：

- **用户归档**：此处有一个可乐饮料的用户归档和一个橙汁的用户归档。
- **视图**：使用视图来同时显示两个归档的数据域；该实例中：色素产品组。

每个用户归档均由具有可编辑属性的数据域组成。本实例中，可乐饮料的成份位于数据域内。每个数据域都具有多个属性，如名称、别名、类型、长度、值等。用户归档编辑器中

数据域和属性以行和列显示。因此，我们叙述时用行代替数据域，用列代替属性。例如，“Cola”用户归档的结构如下：

Cola 用户归档	属性（列）						
数据域（行）	名称	别名	类型	长度	最小值	最大值	起始值
Water	Water	Brun.5	Int	2	1000	1500	1000
Sugar	Sugar	Zmela	Int	2	120	140	130
Dye stuff 7	Dyestuff7	FS1007	Int	2	6	8	6
Caffeine	Caffeine	Caffeine	Int	2	2	3	2
Phosphoric acid	Phosphoric acid	PhosAc	Int	2	170	190	170

2.2.4 使用 SIMATIC S5 / S7 进行数据交换

2.2.4.1 SIMATIC 消息框架接口

用户归档与 S5 和 S7 PLC 之间的数据交换可以通过原始数据变量或通过 WinCC 变量执行。可以使用除 AS511 编程接口外的所有 SIMATIC 接口。

使用原始数据变量与 WinCC 进行的数据交换可以通过以下命令完成：

- S7-400
- S5-PLC-115U 或更高版本

将说明下列主题：

- 通过 WinCC 变量与 S5 和 S7 进行数据交换
- 通过原始数据变量与 S5 和 S7 进行数据交换
- WinCC 和 S5/S7 之间的数据格式差异

2.2.4.2 通过 WinCC 变量与 S5 和 S7 进行数据交换

通过 WinCC 变量与 S5 和 S7 进行数据交换非常简单。 但必须注意，对于用户归档数据类型，只能使用特定的变量管理器数据类型。

当用户归档编辑器中使用的数据类型为整型、双精度和字符串时，必须在数据管理器的变量管理器中使用以下数据类型。 对于用户归档数据类型日期/时间，变量管理器中没有合适的数据类型。

用户归档编辑器中的选择	变量管理器/WinCC 变量
数字（整型）	有符号 32 位数
数字（浮点型）	浮点数 32 位 IEEE 754
数字（双精度）	浮点数 64 位 IEEE 754
字符串（字符串型）	文本变量，8 位字符集
日期/时间	没有合适的数据类型

2.2.4.3 通过原始数据变量与 S5 和 S7 进行数据交换

通过原始数据变量与 S5 和 S7 进行数据交换

以下将介绍通过 WinCC 原始数据变量，在用户归档和自动化系统之间进行的数据交换。 要执行此操作，在 AS 中使用 BSEND/BRCV 函数。 主动从 AS 发送原始数据变量。 消息框架包含发给 WinCC 用户归档的一条或多条请求。 这些请求可以是写请求和读请求。 作为对这些请求的答复，WinCC 将发送回请求的数据和处理确认。

说明

由于 AS 主动参与数据交换，必须在 AS 中直接启动 WinCC 用户期望的用户归档功能，例如从归档值读取/写入。 例如，该触发可以通过 AS 中用于触发用户归档相应功能的外部 WinCC 变量的值来完成。

进行数据交换时，作业或确认报头中使用的“Job type”参数不能用于触发 AS 的功能，因为它只在与用户归档的关系中才具有功能。

将介绍有关下列主题的信息：

- 发送作业/数据到 WinCC
- 将处理确认/数据发送到 SIMATIC S5 和 S7
- 通讯报头的结构

发送作业/数据到 WinCC

原始数据

将作业和数据从 SIMATIC S5 和 S7 PLC 发送到 WinCC 的原始数据变量的结构：

到 S5 / S7 的报文
报文报头报文报头
作业报头 1 作业报头 1
作业 1 的数据作业 1 的数据
可能的作业报头 2
可能的作业 2 的数据
作业 n

将处理确认/数据发送到 SIMATIC S5 / S7

原始数据变量

用于将处理确认和数据从 WinCC 发送到 SIMATIC S5 和 S7 PLC 的原始数据变量的结构：

要发送到 S5 和 S7 的原始数据变量
处理确认
确认报头
确认数据

通讯报头的结构

报文块的结构

独立消息框架块的结构（字节数分布）：

域函数	**注释**
以字节为单位的消息框架长度 LSB *)	域长度为 4 个字节
.	最大长度为 4091 个字节

.	.（bec. S5/ S7 传输）
以字节为单位的消息框架长度 MSB **)	.
传送类型	1 为从 WinCC，2 为从 PLC
保留	
消息框架中作业的数量 LSB *)	域长度为 2 个字节
消息框架中作业的数量 MSB**)	.
PLCID 1.字符	指定名称
.	以 ASCII 执行
.	域长度为 8 个
.	字节
.	.
.	.
.	.
PLCID 8.字符	.

*） LSB = 最低有效字节（最低值字节）

**) MSB = 最高有效字节（最高值字节）

作业报头

作业报头的结构

独立作业报头的结构（字节数分布）：

域函数	**注释**
以字节为单位的作业长度 LSB	域长度为 2 个字节
以字节为单位的作业长度 MSB	.
Job TypeJob Type	参见描述
保留	
域编号 LSB	域长度为 2 个字节
域编号 MSB	.
记录编号 LSB	域长度为 4 个字节
.	.
.	.

数据记录编号 MSB	.
选择标准 LSB	域编号（选择依据）
选择标准 MSB	（不适用于 0）域长度为 2 个字节

作业的数据

作业的数据

作业数据对应于数据记录（或指定地址的域）的内容。

说明

1. 文本框并非以 \0 终止!!!
2. 数字必须以 Intel 格式传送（首先传送 LSB，最后传送 MSB）。
3. 整型域的长度为 4 个字节，浮点域为 4 个字节，双域为 8 个字节。
4. 数据以域的长度移动，该长度在选择标准的值不等于 0 时被作为选择标准。
5. 如果要使用选择标准，则会将数据区的起始值用作选择标准域范围内的选择值。

确认报头

确认报头

独立确认报头的结构（字节数分布）：

域函数	**注释**
以字节为单位的消息框架长度 LSB	域长度为 4 个字节
.	.
.	.
以字节为单位的消息框架长度	.
传送类型	1 为从 WinCC，2 为从 PLC
保留	
Job TypeJob Type	参见描述
ErrorCodeErrorCode	参见描述
保留	
保留	

域编号 LSB	域长度为 2 个字节
域编号 MSB	.
记录编号 LSB	域长度为 4 个字节
.	.
.	.
数据记录编号 MSB	.
PLCID 1.字符	指定名称已完成
.	使用 ASCII
.	域长度为 8 个字节
.	.
.	.
.	.
.	.
PLCID 8.字符	.

确认的数据

确认包含数据记录或指定地址的域（读取请求中）或为空（编写作业、归档作业）。

作业类型和错误代码的描述

作业类型的描述

类型	描述
4	检查用户归档是否存在
5	删除用户归档中的所有记录
6	读取数据集
7	写入数据记录
8	删除记录
9	读取数据记录域
10	写入数据记录域

错误代码的描述

组	编号	描述	造成错误的可能原因
常规	0	函数已被执行	--
归档	2	数据不可用	- 不存在使用该 PLCID 配置的归档
数据记录	101	数据未经允许	- 记录布局不匹配，例如域的编号或类型 - 添加或更新记录失败，例如，因为已配置域的类型“有限”的归档，或最大值或最小值 - 过滤标准错误
数据记录	102	数据不可用	(仅适用于作业类型 6) - 无可用数据 - 过滤标准错误
数组	201	数据未经允许	(仅适用于作业类型 10) - 过滤标准错误，例如，因为域不可用或已为域配置最大值或最小值
数组	202	数据不可用	(仅适用于作业类型 9) - 过滤标准错误或未找到符合过滤标准的域
常规	254	功能不可用	--
常规	255	未定义的错误	--

2.2.4.4 WinCC 和 S5/S7 之间的数据格式差异

WinCC 中的数据格式与 SIMATIC-S5/S7 PLC 中的数据格式存在很大差异。必须考虑到这点以避免意外错误。

WinCC 中保留了 Intel 和 Microsoft 的数据类型，其中首先存储最低有效字节，最后存储最高有效字节。这种数据格式很常见，通常称为“Intel 格式”。可以用一个实例来说明“Intel 格式”：

Intel 格式

在 Intel 格式中，十进制数 300 的存储方式如下：

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
二进制	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0

十六进制	0	1	2	C

根据 Intel 格式，十进制数 300 与十六进制数 12C 相对应（1*256 + 2*16 + 12）。

SIMATIC 格式

SIMATIC 格式中，最低有效字节存储于最高有效位置上。在 SIMATIC 格式中，十进制位数 300 的存储方式如下：

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
二进制	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1

十六进制	2	C	0	1

根据 SIMATIC 格式，十进制数 300 与十六进制数 2C01 相对应。如果根据 Intel 格式错误地转换 2C01，将得到十进制的 11265，这是个严重偏差。

对 SIMATIC PLC 而言，存在可执行相应数据转换的可用功能块。在 S5/ S7 和 WinCC 之间进行数据传送前后，都将启动这些功能块。 可以通过 Internet 上的 Siemens 客户支持下载这些功能块。

http://support.automation.siemens.com/

输入下面的搜索表达式： "Funktionsbausteine ANSI_S5"

提供了压缩文件 ANSI_S5.EXE 供下载。ANSI_S5.EXE 包含功能块“IEEE:GP”。

PLC 或 CP（通讯处理器）的参考手册中对主动发送进行了描述。

2.2.5 附录

2.2.5.1 附录

附录中的主题

您可在本附录中查看以下主题：

- 用于指定用户归档的排序和过滤条件的 SQL 指令
- 按字母顺序排列的 SQL 关键字列表，这些关键字不可用作用户归档中的归档或域名称
- 典型应用程序，尤其是读写变量时的性能以及画面打开时间的性能
- 必须注意的实项

2.2.5.2 SQL 语言

SQL（结构化查询语言）是一种功能强大的通用数据库语言。在 WinCC 脚本语言的函数中，SQL 语言用于数据库作业。更多信息请参考专业参考资料。

对于某些标准函数以及用户归档编辑器中的某些函数而言，您必须以数据库语言 SQL 指定条件，以编辑数据记录的规范。下面将提供有关如何使用 SQL 指令的部分实例：

- FieldA > '1992-12-31 23:45:12.12'：该函数语句将选择“FieldA”列中的值大于输入值的所有数据记录。FieldA 的数据类型为 DB_TYP_TIME。
- FieldB like 'Cauldron%'：该函数语句用于选择“FieldB”列中含有例如“Cauldron1”、“Cauldron4”、“Cauldron12”等值的数据记录。FieldB 的数据类型为 DB_TYP_CHAR。
- FieldC > 100：通过这种形式的条件限制，将选择“FieldC”列中的值大于 100 的所有数据记录。FieldC 的数据类型为 DB_TYP_INTEGER。
- BETWEEN FieldC = 20 AND FieldC = 200：该函数语句将选择“FieldC”列中的值介于 20 和 200 之间的所有数据记录。FieldC 的数据类型为 DB_TYP_INTEGER。
- FieldD：该函数语句使数据记录按“FieldD”列排序。
- FieldE desc：该函数语句将根据“FieldD”列以反向字母顺序排序（降序）。

2.2.5.3 按字母顺序排列的 SQL 关键字列表

SQL 关键字

归档、视图和域的名称只能由字母、数字和下划线“_”组成，且最多为 25 个字符。 第一个字符必须为字母。

以下词语不可用作归档、视图和域的名称：

- “Archives”
- “View”
- “Field”
- “ViewCol”
- 所有的 SQL 关键字

数据库语言 SQL 的关键字（或保留词）不可用作用户归档中的归档、视图或域的名称。以下为 SQL 关键字：

SQL 语言中使用的关键字			
add	all	alter	and
any	as	asc	begin
between	binary	break	by
call	cascade	cast	char
char_convert	character	check	checkpoint
close	comment	commit	connect
constraint	continue	convert	create
cross	current	cursor	date
dba	dbspace	deallocate	dec
decimal	declare	default	delete
desc	distinct	do	double
drop	else	elseif	encrypted
end	endif	escape	exception
exec	execute	exists	fetch
first	float	for	foreign
from	full	goto	grant
group	having	holdlock	identified

SQL 语言中使用的关键字			
if	in	index	inner
inout	insert	instead	int
integer	into	is	isolation
join	key	left	like
lock	long	match	membership
message	mode	modify	named
natural	noholdlock	not	null
numeric	of	Off	on
open	option	Options	or
order	others	out	outer
passthrough	precision	prepare	primary
Print	privileges	proc	procedure
raiserror	readtext	Real	reference
references	release	remote	rename
resource	restrict	return	revoke
right	rollback	save	savepoint
schedule	select	set	share
smallint	some	sqlcode	sqlstate
start	stop	subtrans	subtransaction
synchronize	syntax_error	table	temporary
then	time	tinyint	to
tran	trigger	truncate	tsequal
union	unique	unknown	update
user	using	validate	values
varbinary	varchar	variable	varying
view	when	where	while
with	work	writetext	

2.2.5.4 数量框架

测试环境

下述测量结果均在以下测试环境中测得：

- 硬件： Pentium III 600 / 256MB
- 连接： S7 Protocol Suite，通道单元 MPI
- 项目环境：
  - 数据量： 三个用户归档，其中归档 1 具有 100 个域 3000 个数据记录，归档 2 具有 200 个域 1500 个数据记录，归档 3 具有 500 个域 500 个数据记录。
  - 运行系统中的测量结果，对第一行和最后一行进行读取或写入，触发工具栏按钮上方表格控件中的读/写令。
  - 编辑器关闭，未执行任何 C 函数，且未执行任何画面交换。

WinCC 变量和原始数据变量

测量过程中使用了 WinCC 变量。 如测量结果显示，访问时间会随用户归档的增大而增加。

对于较大的用户归档，建议使用原始数据变量。 原始数据变量以数据包形式传送数据，而且对于较大的用户归档也具有较快的访问速度。

说明

对每个用户归档，最多可创建 500 个域。

2.2.5.5 读写变量的性能

性能

此处所述的性能测量中，用户归档行为均在读写变量的过程中进行测量。

需要注意，性能取决于所采用的典型应用程序。

域的数目	数据记录的数目	写入变量的时间（以秒计）	从变量读取的时间（以秒计）
100	1	1	2
100	10	1	2-3
100	50	1	3-4

100	100	1	3-9
100	1000	1-2	>3（取决于连接）
200	1	2	3
200	10	2	4
200	50	1-2	>4
200	100	1-2	>4
200	1000	2-3	>4
500	1	3	4
500	10	3	7
500	50	3-4	约为 15
500	100	4	>15
500	500	4	>15

计算出的时间取决于相应表格的大小。

2.2.5.6 画面打开时间的性能

画面打开时间

在以下表格中，将显示一份画面打开时间的性能测量数据。 假设编辑器用户归档中的表格窗口未激活。

需要注意，画面打开时间取决于所采用的典型应用程序。

域的数目	数据记录的数目	画面打开时间（以秒计）
100	1	1
100	10	2
100	100	3
100	500	3
100	1000	3
100	2000	3
100	3000	3
200	1	1
200	10	2
200	100	4

200	500	4
200	1000	4
200	1500	>4
500	1	3
500	10	4
500	100	>4
500	500	>4

2.3 用户归档脚本

2.3.1 WinCC 脚本语言的标准函数

用户归档标准函数的描述分为以下几部分：

- 有关脚本编程的常规信息
- 用于编辑和显示用户归档的函数
- 标准函数的句柄
- 脚本的实用例子
- 用户归档标准函数的参考
- 可在 WinCC 用户归档在线帮助中找到对用户归档函数的详细描述。

WinCC 提供许多标准函数，使用户能以灵活的方式执行用户归档。

这些标准函数以统一的命名规则来标识。 用户归档的所有标准函数均以“ua”开始，例如“uaConnect”、“uaArchiveOpen”、“uaArchiveGetFields”等。用户归档运行系统函数始终以“uaArchive”开始。

这些函数分为组态函数和运行系统函数。 组态函数和运行系统函数需要句柄，它们由先前调用的“uaQueryConfiguration”、“uaConnect”和“uaOpen”函数返回。

说明

用户归档函数中的“ua”始终以小写字母形式书写。

在脚本内部，必须确保日期是最新数据。

如果脚本已打开了用户归档且已通过控制或用户归档编辑器添加或删除记录，则脚本不会获知此更改。 仅在再次查询后才将更改通知脚本。

参见

脚本实例 (页 377)

用于编辑和显示用户归档的函数 (页 372)

用于用户归档组态的句柄 (页 373)

WinCC 脚本语言标准函数的描述 (页 387)

2.3.2 动作组态

执行以下步骤以组态动作：

1. 启动图片编辑器并创建系统画面
2. 右键单击要对其添加动作（例如按钮）的对象。
3. 选择属性
4. 从“属性”或“事件”标签中选择元素，然后双击所需的动作（例如，要为“按左键”鼠标动作组态动作，请选择“事件/鼠标/按左键”）。 在后面的对话框中，可直接输入 C 代码然后进行编译。
5. 5. 单击“确认”以结束动作的组态。

2.3.3 用于编辑和显示用户归档的函数

使用标准函数进行组态

“uaQueryConfiguration”函数为组态函数提供句柄（UAHCONFIG）。 可通过该句柄调用“uaSetArchive”、“uaAddArchive”、“uaSetField”、“uaAddField”等组态函数。
“uaReleaseConfiguration”函数将终止组态。

建立与用户归档的连接。

为了在运行系统中进行访问，必须调用 uaConnect 标准函数以建立与用户归档组件的连接。uaConnect 生成 UAHCONNECT 句柄，可以通过该句柄来打开用户归档和视图。
“uaDisconnect”函数将终止与用户归档的连接。

打开运行系统函数

运行系统操作需要已组态的用户归档。 “uaQueryArchive”和“uaQueryArchiveByName”函数为运行系统函数提供了句柄。 通过“uaArchiveOpen”函数打开用户归档后，便可使用用户归档运行系统函数。

用于运行系统操作的函数

“uaArchiveNext”、“uaArchivePrevious”、“uaArchiveFirst”和“uaArchiveLast”函数可以移动指针。 通过“hArchive”句柄生成了用户归档数据记录的唯一分配。 该分配还允许间接寻址，只要画面对话框提出请求。

“uaArchiveUpdate”函数将临时数据记录存储到用户归档中，并覆盖指针当前指向的数据记录。 该数据记录必须事先由“uaArchiveNext”、“uaArchivePrevious”、“uaArchiveFirst”或“uaArchiveLast”函数读取。

终止与用户归档的连接

“uaArchiveClose”函数将关闭用户归档。 “uaReleaseArchive”函数终止与当前用户归档的连接，而“uaDisconnect”函数则终止与用户归档组件的连接。

说明

脚本中所建立的与用户归档的连接也必须再次在该脚本中终止。

用于建立连接的函数	用于终止连接的函数
UaQueryConfiguration	uaReleaseConfiguration
uaConnect	uaDisconnect
uaQueryArchive	uaReleaseArchive
uaQueryArchiveByName	uaReleaseArchive
uaArchiveOpen	uaArchiveClose

对于用户归档，存在两种 API 调用形式：

1. 通过用于脚本（全局脚本和动作编程）的前缀“ua”（小写字母）。

2. 通过用于在 WinCC 之外运行的程序的前缀“UA”（大写字母）。

如在动态向导中使用用户归档的调用，则这些调用必须以前缀“UA”（大写字母）开始。

2.3.4 用于用户归档组态的句柄

“UAHCONFIG”句柄

“uaQueryConfiguration”函数生成“UAHCONFIG”句柄，该句柄是进行用户归档组态的先决条件。 也就是说，必须首先调用“uaQueryConfiguration”函数才能获得“UAHCONFIG”句柄。

该句柄随后将使您能调用下列组态函数。 最后，要完成组态，必须调用“uaReleaseConfiguration”。

用于用户归档组态的句柄	
UaQueryConfiguration	---> UAHCONFIG 句柄
	请求者：
	uaAddArchive
	uaAddField
	uaGetArchive
	uaGetField
	uaGetNumArchives
	uaGetNumFields
	uaReleaseConfiguration
	uaRemoveAllArchives
	uaRemoveAllFields
	uaRemoveArchive
	uaRemoveField
	uaSetArchive
	uaSetField

2.3.5 用于运行系统函数的句柄

“UAHCONNECT”句柄

“uaConnect”用户归档函数生成“UAHCONNECT”句柄，该句柄需用于打开和关闭用户归档和视图。 也就是说，必须首先调用“uaConnect”函数才能获得“UAHCONNECT”句柄。 该句柄随后将使您能调用下列用于打开和关闭归档和视图的函数。 最后，要完成组态，必须调用“uaDisconnect”。

“uaQueryArchive”和“uaQueryArchiveByName”函数生成“UAHARCHIVE”句柄。 该句柄是“uaArchiveOpen”函数（打开用于运行系统操作的用户归档）的先决条件。 要终止连接，必须调用“uaRelease”和“uaArchiveClose”函数。

说明

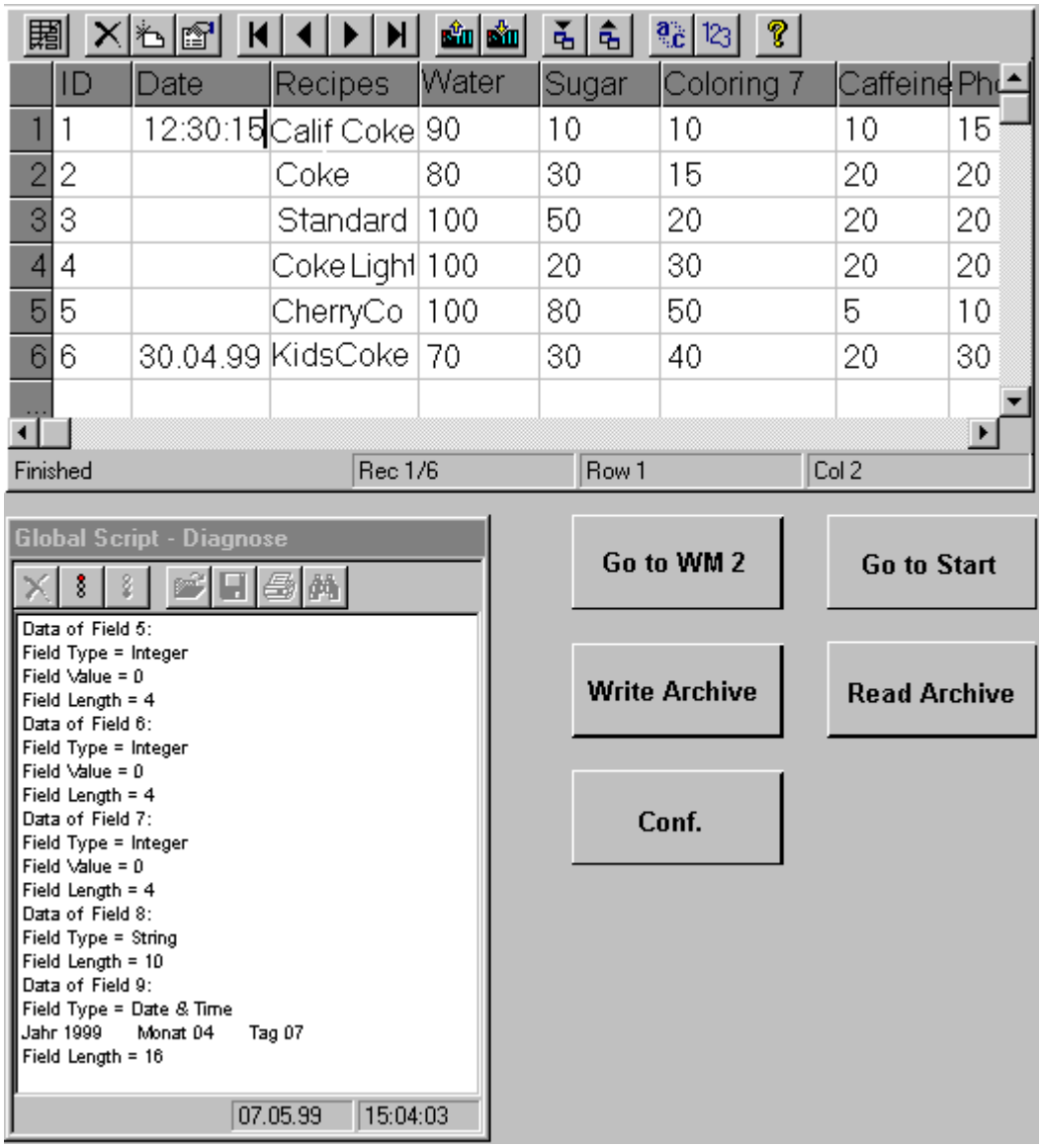
对用户归档进行排序和过滤

可以对用户归档使用“uaArchiveSetSort”和“uaArchiveSetFilter”函数，而无需使用“uaArchiveOpen”打开用户归档。

用于运行系统函数的句柄		
uaConnect	-> 句柄 UAHCONNECT	
	请求者...	
	uaDisconnect	
	uaQueryArchive	--> UAHARCHIVE 句柄
	uaQueryArchiveByName	--> UAHARCHIVE 句柄
		请求者...
		uaArchiveOpen
		需用于：
		uaArchiveClose
		uaArchiveDelete
		uaArchiveExport
		uaArchiveGetCount
		uaArchiveGetFieldLength
		uaArchiveGetFields
		uaArchiveGetFieldType
		uaArchiveGetFieldValueDate
		uaArchiveGetFieldValueDouble
		uaArchiveGetFieldValueFloat
		uaArchiveGetFieldValueLong
		uaArchiveGetFieldValueString
		uaArchiveGetFieldName
		uaArchiveGetFilter
		uaArchiveGetID
		uaArchiveGetName

用于运行系统函数的句柄		
		uaArchiveGetSort
		uaArchiveImport
		uaArchiveInsert
		uaArchiveMoveFirst
		uaArchiveMoveLast
		uaArchiveMoveNext
		uaArchiveMovePrevious
		uaArchiveReadTagValues
		uaArchiveReadTagValuesByName
		uaArchiveRequery
		uaArchiveSetFieldValueDate
		uaArchiveSetFieldValueDouble
		uaArchiveSetFieldValueFloat
		uaArchiveSetFieldValueLong
		uaArchiveSetFieldValueString
		uaArchiveSetFilter*
		uaArchiveSetSort*
		uaArchiveUpdate
		uaArchiveWriteTagValues
		uaArchiveWriteTagValuesByName
		uaReleaseArchive
*: 只可在“uaArchiveOpen”之前调用的函数。		

2.3.6 脚本实例



以下章节包含可对运行系统中用户归档进行读写的两个标准函数的实例。
“UAReadFromArchive”函数读取“Cola”用户归档，并在全局脚本诊断控件窗口中显示读取的数据。 “UAWriteToArchive”函数编写用户归档，并显示状态和消息。 创建诊断窗口的方法为：将 OLE 控件（从“对象选项板”->“智能对象”中选择）置于图形编辑器中，然后从“插入 OLE 控件（OCX）”对话框中选择“WinCC 全局脚本 - 诊断控件”。

在图形编辑器中，为项目创建新的画面。 在画面中，创建“读取归档”和“写入归档”按钮，并为它们分配下述标准函数。 其步骤如下：

1. 打开“对象选项板”选择窗口，然后选择“窗口对象”下的“按钮”域。
2. 将按钮置于图形编辑器的工作区中，然后按住鼠标按钮将其拖动至合适的大小。

3. 右键单击该新建按钮，然后从弹出式菜单中选择“属性”。 在“属性”标签中，可定义按钮标签（文本）和颜色。 例如，标签的名称可为“读取归档”和“写入归档”。
4. 在“事件”标签中，选择“鼠标”然后双击“鼠标动作”可将动作分配给鼠标。 将显示脚本编辑器。 输入下列“UAReadFromArchive”标准脚本函数：

```
#include "apdefap.h"

void UAReadFromArchive()
{
   UAHCONNECT    hConnect = 0;
   UAHARCHIVE    hArchive = 0;
   LONG          IndexArchive;
   LONG          FieldLength;
   LONG          FieldType,
   LONG          NumberOfFields;
   LONG          Index;
   long          IntValue;
   double        DoubleValue;
   char          ArchiveName[255], StringField[255];
   SYSTEMTIME    SysDate;

//*******    连接到组件用户归档 ****************************
   if (uaConnect( &hConnect ) == FALSE )
   {
   printf("uaConnect error: %d\n", uaGetLastError() );
   return;
   }
   if (hConnect == NULL)
   {
   printf("Handle UAHCONNECT equals 0\n" );
   return;
   }

```

```
//*******   连接到归档（通过归档名称） ****************************
   if (uaQueryArchiveByName( hConnect, "Cola", &hArchive ) == FALSE )
   {
   printf("uaQueryArchive Error: %d\n", uaGetLastError() );
   goto finish;
   }


//*******   打开归档 ***********************************************************
   if ( uaArchiveOpen( hArchive ) == FALSE )
   {
   printf("uaArchive Open Error\n" );
   goto finish;
   }

//*******   移至第一个记录设置 *************************************************
   if (uaArchiveMoveFirst(hArchive) == FALSE )
   {
   printf("uaArchiveMoveFirst Error = %u\n" );
   goto finish;
   }

//*******   获取域的数目 *******************************************
   NumberOfFields = uaArchiveGetFields( hArchive );
   printf("Number of Fields = %u\n", NumberOfFields );

//*******   读取并显示数据域 ***************************************
   for ( Index = 1; Index < NumberOfFields; Index++ )
   {
   printf("Data of Field %u: \n", Index  );
   FieldType = uaArchiveGetFieldType( hArchive, Index );
   switch ( FieldType )
```

```
{
case UA_FIELDTYPE_INTEGER :
printf("Field Type = Integer\n");
if ( uaArchiveGetFieldValueLong ( hArchive, Index, &IntValue ) == TRUE )
printf( "Field Value = %u\n", IntValue );
else
printf("Error callinguaArchiveGetFieldValueLong: %d\n", uaGetLastError() );

break;

case UA_FIELDTYPE_DOUBLE :
printf("Field Type = Double\n");
if (uaArchiveGetFieldValueDouble (
hArchive, Index, &DoubleValue ) == TRUE )
printf("Field Value = %g\n", DoubleValue );
else
printf("Error calling uaArchiveGetFieldValueDouble: %d\n", uaGetLastError() );

break;

case UA_FIELDTYPE_STRING :
printf("Field Type = String\n");
if (uaArchiveGetFieldValueString ( hArchive, Index, StringField, 20 ) ==     TRUE )
printf("Field Value = %s\n", StringField );
else
printf("Error callinguaArchiveGetFieldValueString: %d\n", uaGetLastError() );

break;

case UA_FIELDTYPE_DATETIME :
printf("Field Type = Date & Time\n");
if (uaArchiveGetFieldValueDate ( hArchive, Index, &SysDate ) == TRUE )
```

```
   printf("%d.%d.%d\n ",SysDate.wDay, SysDate.wMonth, SysDate.wYear );
   else
   printf("Error calling uaArchiveGetFieldValueLong: %d\n", uaGetLastError() );

   break;

   case -1 :
   default:
   printf("Error executing uaArchiveGetFieldType\n");
   }

//*******   读取并显示域长度 ************************************
   FieldLength = uaArchiveGetFieldLength( hArchive, Index );
   if ( FieldLength != -1 )
   printf("Field Length = %u\n", FieldLength );
   else
   printf("Error executing uaArchiveGetFieldLength\n");
   }

//*******   关闭所有句柄和连接 ***************************
finish:;
//*******   关闭归档 ************************************************
   if( NULL != hArchive )
   {
      if (uaArchiveClose ( hArchive ) == FALSE )
      {
      printf("error on closing archive\n" );
      }
   }
//******   释放与归档的连接 ************************************
   if( NULL != hArchive )
   {
```

```
    if (uaReleaseArchive ( hArchive ) == FALSE )
    {
    printf("error on releasing archive\n" );
    }
    hArchive = 0;
  }
//*******   断开与组件用户归档的连接 *************************
  if( NULL != hConnect )
  {
    if (uaDisconnect ( hConnect ) == FALSE )
    {
    printf("error on disconnection\n" );
    }
    hConnect = 0;
  }
}
```

创建第二个按钮用于描述用户归档。 按照创建第一个按钮的步骤进行。 选择标准函数“UAWriteToArchive”，然后输入以下脚本：

```

void UAWriteToArchive()
{
  UAHCONNECT      hConnect = 0;
  UAHARCHIVE      hArchive = 0;
  LONG            IndexArchive;
  LONG            FieldLength;
  LONG            FieldType,
  LONG            NumberOfFields;
  LONG            Index;
  long            IntValue;
  double          DoubleValue;
  char            StringField[255];
```

```
   SYSTEMTIME       SysDate;

//*******   连接到组件用户归档 **************************
   if (uaConnect( &hConnect ) == FALSE )
   {
   printf("uaConnect error: %d\n", uaGetLastError() );
   return;
   }
   if (hConnect == NULL)
   {
   printf("Handle UAHCONNECT equals NULL\n" );
   return;
   }

//*******   连接到归档（通过名称） ************************************
   if (uaQueryArchiveByName( hConnect, "Cola", &hArchive ) == FALSE )
   {
   printf("uaQueryArchive Error: %d\n", uaGetLastError() );
   goto finish;
   }

//*******   打开归档 ***************************************************
   if ( uaArchiveOpen( hArchive ) == FALSE )
   {
   printf("uaArchive Open Error\n" );
   goto finish;
   }

//*******   获取域的数目 ***********************************************
   NumberOfFields = uaArchiveGetFields( hArchive );
   printf("Number of Fields = %u\n", NumberOfFields );
```

```
//*******   读取最后一个数据集 ************************************************
   if (uaArchiveMoveLast( hArchive ) == TRUE )
   printf("Number of Fields = %u\n", NumberOfFields );
   else
   {
   printf("uaArchiveMoveLast Error: %d\n", uaGetLastError() );
   goto finish;
   }


//*******   写入数据域 ******************************************
   IntValue = 32;
   DoubleValue = 64;
   strcpy(StringField, "Text12" );
   GetSystemTime( &SysDate );

   for ( Index = 1; Index < NumberOfFields; Index++ )
   {
   printf("Data of Field %u: \n", Index  );

   FieldType = uaArchiveGetFieldType( hArchive, Index );

   switch ( FieldType )
   {
   case UA_FIELDTYPE_INTEGER :
   printf("Field Type = Integer\n");
   if (uaArchiveSetFieldValueLong ( hArchive, Index, IntValue ) == TRUE )
   printf( "Field Value = %u\n", IntValue );
   else
   printf("Error calling uaArchiveSetFieldValueLong: %d\n", uaGetLastError() );
```

```
   break;

   case UA_FIELDTYPE_DOUBLE :
   printf("Field Type = Double\n");if  ( uaArchiveSetFieldValueDouble (hArchive, Index, DoubleValue ) == TRUE )
   printf("Field Value = %g\n", DoubleValue );
   else
   printf("Error calling uaArchiveSetFieldValueDouble: %d\n", uaGetLastError() );

   break;

   case UA_FIELDTYPE_STRING :
   printf("Field Type = String\n");
   if (uaArchiveSetFieldValueString ( hArchive, Index, StringField ) == TRUE )
   printf("Field Value = %s\n", StringField );
   else
   printf("Error calling uaArchiveSetFieldValueString: %d\n", uaGetLastError() );

   break;

   case UA_FIELDTYPE_DATETIME :
   printf("Field Type = Date & Time\n");
   if (uaArchiveSetFieldValueDate ( hArchive, Index, &SysDate ) == TRUE )
   printf("%d.%d.%d\n ",SysDate.wDay, SysDate.wMonth, SysDate.wYear );
   else
   printf("Error calling uaArchiveGetFieldValueLong: %d\n", uaGetLastError() );

   break;

   case -1 :
   default:
   printf("Error executing uaArchiveGetFieldType\n");
```

```
    }

    FieldLength = uaArchiveGetFieldLength( hArchive, Index );
    if ( FieldLength != -1 )
    printf("Field Length = %u\n", FieldLength );
    else
    printf("Error executing uaArchiveGetFieldLength\n");
    }
// ******* 更新归档 *****************************************************
    if (uaArchiveUpdate(hArchive) == FALSE )
    {
    printf("uaArchiveUpdate Error:\n" );
    }

//*******   关闭所有句柄和连接 ***************************
finish:;
//*******   关闭归档 *************************************************
    if( NULL != hArchive )
    {
        if (uaArchiveClose ( hArchive ) == FALSE )
        {
        printf("error on closing archive\n" );
        }
    }
//******   释放与归档的连接 ***********************************
    if( NULL != hArchive )
    {
        if (uaReleaseArchive ( hArchive ) == FALSE )
        {
        printf("error on releasing archive\n" );
        }
        hArchive = 0;
```

```
    }
//*******    断开与组件用户归档的连接 *************************
    if( NULL != hConnect )
    {
        if (uaDisconnect ( hConnect ) == FALSE )
        {
        printf("error on disconnection\n" );
        }
        hConnect = 0;
    }
}
```

然后便可关闭对话框并启动运行系统。 这样便可在“全局脚本诊断窗口”中监视脚本的效果。

2.3.7 附录

2.3.7.1 WinCC 脚本语言标准函数的描述

WinCC 脚本语言

以下章节包含对 WinCC 脚本语言函数的详细描述。

这些函数分为四组：

- 用于用户归档的标准函数
- 用于组态用户归档的函数
- 常规运行系统函数
- 归档专用的运行系统函数

组中的函数按字母顺序排列。

在创建数据记录时，不检查完整性或正确性。 特别需要指出的是，域不得留空。

参见

WinCC 脚本语言的标准函数 (页 371)

2.3.7.2 用于用户归档的标准函数

uaGetLastError

描述：

WinCC 脚本语言函数返回一个 BOOL 值，其中的 TRUE 对应无错误处理。 如果返回 FALSE，则可通过“uaGetLastError()”和“uaGetLastHResult()”来读取最后一个函数的错误。

如果是在运行若干函数后调用“uaGetLastError()”，则“uaGetLastError()”将返回最后发生的错误。 为了确定发生错误的函数，在每次函数返回 FALSE 时都应调用“uaGetLastError()”和“uaGetLastHResult()”。

实例：

```
if ( uaArchiveGetFieldValueLong ( hArchive, Index, &IntValue ) == TRUE )
printf( "Field Value = %u\n", IntValue );
else
printf("Error calling uaArchiveGetFieldValueLong: %d / %08lx\n", uaGetLastError(), uaGetLastHResult());
```

对于未返回值（VOID）的函数，无论如何都将通过 uaGetLastError() 出现请求。

实例：

```
  uaArchiveGetFilter(hArchive, pszFilter, cMaxLen);
    INT nUAError = uaGetLastError (  );
    if ( UA_ERROR_SUCCESS != nUAError)
    {
        printf( "Filter = [%s]\n", pszFilter );
    }
    else
    {
        printf("Error calling uaArchiveGetFilter: %d, hr=0x%08lX \n", nUAError, uaGetLastHResult());
    }
```

```
INT uaGetLastError()
```

返回值：

执行最后一个函数的错误状态。 “uaGetLastError()”可返回以下错误：

UA_ERROR_SUCCESS

UA_ERROR_GENERIC

UA_ERROR_CONNECT_FAILED

UA_ERROR_OPEN_FAILED

UA_ERROR_CLOSE_FAILED

UA_ERROR_REQUERY_FAILED

UA_ERROR_MOVE_FAILED

UA_ERROR_INSERT_FAILED

UA_ERROR_UPDATE_FAILED

UA_ERROR_DELETE_FAILED

UA_ERROR_IMPORT_FAILED

UA_ERROR_EXPORT_FAILED

UA_ERROR_READ_FAILED

UA_ERROR_WRITE_FAILED

UA_ERROR_GET_FAILED

UA_ERROR_SET_FAILED

UA_ERROR_INVALID_NAME

UA_ERROR_INVALID_TYPE

UA_ERROR_INVALID_NUMRECS

UA_ERROR_INVALID_COMMTYPE

UA_ERROR_INVALID_LENGTH

UA_ERROR_INVALID_PRECISION

UA_ERROR_NULL_POINTER

UA_ERROR_INVALID_POINTER

UA_ERROR_INVALID_HANDLE

UA_ERROR_INVALID_INDEX

UA_ERROR_SERVER_UNKNOWN

这些错误常量以及用户归档例程的预定义均位于 CCUACAPI.H 中。

uaGetLastHResult

描述：

读取最后发生的 COM 错误。 该函数主要用于在 COM 执行过程中诊断不兼容性，或用于检测注册和通讯错误。
如果用户归档函数（例如 uaConnect）以“FALSE”显示错误，则除了使用 UAGetLastError 外，主要还需使用该函数。

```
LONG uaGetLastHResult()
```

返回值：

最后发生的 COM 错误

2.3.7.3 用于组态用户归档的函数

用于组态用户归档的函数

用于组态的函数

这些函数用于组态用户归档。

函数	描述
uaAddArchive	添加新的用户归档。
uaAddField	添加新的域
uaGetArchive	读取归档组态
uaGetField	读取域组态
uaGetNumArchives	查找已组态归档的数目
uaGetNumFields	查找域的数目
UaQueryConfiguration	建立与用户归档组态的连接
uaReleaseConfiguration	终止与组态的连接

uaRemoveAllArchives	删除所有归档
uaRemoveAllFields	删除所有域
uaRemoveArchive	删除归档
uaRemoveField	删除域
uaSetArchive	写入归档组态
uaSetField	设置域组态

uaAddArchive

描述：

添加新的用户归档。 这与使用用户归档编辑器组态新用户归档相对应。

```
LONG uaAddArchive (
UAHCONFIG hConfig,
UACONFIGARCHIVE* pArchive )
```

参数：

```
UAHCONFIG hConfig,
```

用户归档的组态句柄。 该句柄通过“uaQueryConfiguration”生成。

```
UACONFIGARCHIVE* pArchive
```

用于存储用户归档组态的缓冲区上的指针。

返回值：

新用户归档的索引。 错误对应于“-1”。

uaAddField

描述：

添加新的数据域。

```
LONG uaAddField (
UAHCONFIG hConfig,
```

```
long lArchive,
UACONFIGFIELD* pField )
```

参数：

```
UAHCONFIG hConfig,
```

用户归档的组态句柄。 该句柄通过“uaQueryConfiguration”生成。

```
long lArchive,
```

归档索引（0 至 (uaGetNumArchives()-1)）

```
UACONFIGFIELD* pArchive
```

域组态缓冲区的指针。

返回值：

新域的索引。 值“-1”表示发生了错误。

参见

用于编写“uaCONFIGFIELDA”句柄程序的结构 (页 400)

用于编写“uaCONFIGFIELDW”句柄程序的结构 (页 401)

uaGetArchive

描述：

读取用户归档组态。

```
BOOL uaGetArchive (
UAHCONFIG hConfig,
long lArchive,
UACONFIGARCHIVE* pArchive )
```

参数：

```
UAHCONFIG hConfig,
```

用户归档的组态句柄。 该句柄通过“uaQueryConfiguration”生成。

```
long lArchive,
```

归档索引 (0 至 (uaGetNumArchives()-1))

```
UACONFIGARCHIVE* pArchive
```

用于接收用户归档组态的缓冲区上的指针。

返回值：

TRUE： 对用户归档的访问成功

FALSE：错误

uaGetField

描述：

读取域组态。

```
BOOL uaGetField (
UAHCONFIG hConfig,
long lArchive,
long lField,
UACONFIGFIELD* pField )
```

参数：

```
UAHCONFIG hConfig,
```

用户归档的组态句柄。 该句柄通过“uaQueryConfiguration”生成。

```
long lArchive,
```

归档索引 (0 至 (uaGetNumArchives()-1))

```
long lField,
```

域编号，其中 lField = 0 表示指向第一个域的地址。

```
UACONFIGFIELD* pArchive
```

用于接收域组态的缓冲区上的指针。

返回值：

TRUE： 对用户归档的访问成功

FALSE：错误

uaGetNumArchives

描述：

读取当前已组态用户归档的数目。

```
LONG uaGetNumArchives (
UAHCONFIG hConfig )
```

参数：

```
UAHCONFIG hConfig
```

用户归档的组态句柄。 该句柄通过“uaQueryConfiguration”生成。

返回值：

当前已组态用户归档的数目。 如果发生错误，将返回 -1。

uaGetNumFields

描述：

提供已组态域的数目。 不包括“ID”、“上个用户”和“上次访问”域。 在组态调用中，索引将指示为 0 至“uaGetNumFields() -1”。

```
LONG uaGetNumFields (
UAHCONFIG hConfig,
long lArchive )
```

参数：

```
UAHCONFIG hConfig,
```

用户归档的组态句柄。 该句柄通过“uaQueryConfiguration”生成。

```
long lArchive,
```

归档索引（0 至 (uaGetNumArchives()-1)）

返回值：

已组态域的数目。 如果发生错误，将返回 -1。

UaQueryConfiguration

描述：

建立与用户归档（用于组态）的连接。

```
BOOL uaQueryConfiguration (
UAHCONFIG* phConfig )
```

参数：

```
UAHCONFIG* phConfig,
```

指向归档句柄的指针。

返回值：

TRUE： 对用户归档的访问成功

FALSE：错误

uaReleaseConfiguration

描述：

终止与用户归档（组态）的连接。

```
BOOL uaReleaseConfiguration (
UAHCONFIG hConfig,
BOOL bSave )
```

参数：

UAHCONFIG hConfig

用户归档的组态句柄。该句柄通过“uaQueryConfiguration”生成。

BOOL bSave

终止与用户归档的连接之前保存所作的组态更改。

TRUE = 保存更改，FALSE = 放弃更改

□□□□□□□□□□□□□□□□□“□□□□”□bSave = TRUE□□您可以通过请求 uaIsActive() 来检查运行系统是否为激活状态。

返回值：

TRUE：成功终止连接

FALSE：错误

uaRemoveAllArchives

描述：

删除视图中当前未使用的所有用户归档。

```
BOOL uaRemoveAllArchives
( UAHCONFIG hConfig )
```

参数：

UAHCONFIG hConfig

用户归档的组态句柄。该句柄通过“uaQueryConfiguration”生成。

返回值：

TRUE：成功删除

FALSE：错误

注释

执行此操作后，可通过“uaGetNumArchives()”查询是否所有归档都已删除。

uaRemoveAllFields

描述：

删除所有域。

```
BOOL uaRemoveAllFields (
UAHCONFIG hConfig,
long lArchive )
```

参数：

`UAHCONFIG hConfig,`

用户归档的组态句柄。该句柄通过“uaQueryConfiguration”生成。

`long lArchive,`

归档索引（0 至 (uaGetNumArchives()-1)）

返回值：

TRUE：成功删除域

FALSE：错误

uaRemoveArchive

描述：

uaRemoveArchive 删除已组态的整个用户归档。

```
BOOL uaRemoveArchive (
UAHCONFIG hConfig,
long lArchive )
```

参数：

`UAHCONFIG hConfig,`

用户归档的组态句柄。该句柄通过“uaQueryConfiguration”生成。

`long lArchive,`

归档索引（0 至 (uaGetNumArchives()-1)）

返回值：

TRUE：已成功删除用户归档

FALSE：错误

uaRemoveField

描述：

删除域。

```
BOOL uaRemoveField (
UAHCONFIG hConfig,
long lArchive,
long lField )
```

参数：

`UAHCONFIG hConfig,`

用户归档的组态句柄。该句柄通过“uaQueryConfiguration”生成。

`long lArchive,`

归档索引（0 至 (uaGetNumArchives()-1)）

`long lField,`

域编号，其中 lField = 0 表示指向第一个域的地址。

返回值：

TRUE：已成功删除域

FALSE：错误

uaSetArchive

描述：

定义用户归档的组态。

```
BOOL uaSetArchive (
UAHCONFIG hConfig,
long lArchive,
UACONFIGARCHIVE* pArchive
 )
```

参数：

`UAHCONFIG hConfig,`

用户归档的组态句柄。该句柄通过“uaQueryConfiguration”生成。

`long lArchive,`

归档索引（0 至 (uaGetNumArchives()-1)）

`UACONFIGARCHIVE* pArchive`

指向存有用户归档组态的 pArchive 缓冲区的指针。

返回值：

TRUE：对用户归档的访问成功

FALSE：错误

uaSetField

描述：

设置域组态。

```
BOOL uaSetField (
UAHCONFIG hConfig,
long lArchive,
```

```
long lField,
UACONFIGFIELD* pField )
```

参数：

```
UAHCONFIG hConfig,
```

用户归档的组态句柄。该句柄通过“uaQueryConfiguration”生成。

```
long lArchive,
```

归档索引（0 至 (uaGetNumArchives()-1)）

```
long lField,
```

域编号，其中 lField = 0 表示指向第一个域的地址。

```
UACONFIGFIELD* pField
```

域组态缓冲区的指针。

返回值：

TRUE：对用户归档的访问成功

FALSE：错误

用于编写“uaCONFIGFIELDA”句柄程序的结构

构造“uaCONFIGFIELDA”

```
typedef struct tagUACONFIGFIELDA
{
LONG lArchiveId; // 归档的唯一 ID
LONG lArchiveId; // 归档的唯一 ID
LONG lPosition; // 归档的位置（顺序）
CHAR szName[UA_MAXLEN_NAME+1]; // 归档名称最多可以有 20 个字符
CHAR szAlias[UA_MAXLEN_ALIAS+1]; // 别名最多可以有 50 个字符
LONG lType; //归档类型
LONG lLength; /* 数据域为字符串类型时的最大字符数；否则不使用此参数 */
```

```
LONG lPrecision; // 内部使用；无需填充
CHAR szMinValue[UA_MAXLEN_VALUE+1]; /* 数据域不是字符串或日期类型时的最小字符数；否则不使用此参数 */
CHAR szMaxValue[UA_MAXLEN_VALUE+1]; /* 数据域不是字符串或日期类型时的最大字符数；否则不使用此参数 */
CHAR szStartValue[UA_MAXLEN_VALUE+1]; // 起始值
CHAR szDMVarName[UA_MAXLEN_DMVARNAME+1]; /* 数据管理器中的变量（用于通过 WinCC 变量进行通讯的归档） */
DWORD dwReadRight; // 读访问权限
DWORD dwWriteRight; // 写访问权限
DWORD dwFlags; // 上次访问
} UACONFIGFIELDA;
```

参见

uaAddField (页 391)

用于编写“uaCONFIGFIELDW”句柄程序的结构

构造“uaCONFIGFIELDW”

```
typedef struct tagUACONFIGFIELDW
{
LONG lArchiveId; // 归档的唯一 ID
LONG lArchiveId; // 归档的唯一 ID
LONG lPosition; // 归档的位置（顺序）
WCHAR wszName[UA_MAXLEN_NAME+1]; // 归档名称最多可以有 20 个字符
WCHAR wszAlias[UA_MAXLEN_ALIAS+1]; // 别名最多可以有 50 个字符
LONG lType; //归档类型
LONG lLength; /* 数据域为字符串类型时的最大字符数；否则不使用此参数 */
LONG lPrecision; // 内部使用；无需填充
```

```
WCHAR wszMinValue[UA_MAXLEN_VALUE+1]; /* 数据域不是字符串或日期类型时的最小字符数；否则不使用此参数 */
WCHAR wszMaxValue[UA_MAXLEN_VALUE+1]; /* 数据域不是字符串或日期类型时的最大字符数；否则不使用此参数 */
WCHAR wszStartValue[UA_MAXLEN_VALUE+1]; // 起始值
WCHAR wszDMVarName[UA_MAXLEN_DMVARNAME+1]; /* 数据管理器中的变量（用于通过 WinCC 变量进行通讯的归档） */
DWORD dwReadRight; // 读访问权限
DWORD dwWriteRight; // 写访问权限
DWORD dwFlags; // 上次访问
} UACONFIGFIELD
```

参见

uaAddField (页 391)

用于编写“uaAddArchive”句柄程序的结构

构造“uaAddArchive”

```
typedef struct tagUAADDARCHIVE
{
LONG lArchiveId; // 归档的唯一 ID
LONG lPosition; // 归档的位置（顺序）
CHAR szName[UA_MAXLEN_NAME+1]; // 归档名称最多可以有 20 个字符
CHAR szAlias[UA_MAXLEN_ALIAS+1]; // 别名最多可以有 50 个字符
LONG lType;UA_ARCHIVETYPE_UNLIMITED // 归档类型不限
UA_ARCHIVETYPE_LIMITED // 归档类型有限制
LONG lNumRecs; // 数据集的最大数量
LONG lCommType;
UA_COMMTYPE_NONE // 无通讯
```

```
UA_COMMTYPE_RAW // 通过原始数据进行通讯
UA_COMMTYPE_DIRECT // 通过 DM 变量进行通讯
CHAR szPLCID[UA_MAXLEN_PLCID+1]; // 原始数据变量的 PLCID
CHAR szDMVarName[UA_MAXLEN_DMVARNAME+1]; // 原始数据变量的名称
CHAR szIDVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量 ID
CHAR szJobVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量作业
CHAR szFieldVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量域
CHAR szValueVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量值
DWORD dwReadRight; // 读访问权限
DWORD dwWriteRight; // 写访问权限
DWORD dwFlags; UA_ARCHIVEFLAG_ACCESS // 上次访问
UA_ARCHIVEFLAG_USER // 上个用户
} UAADDARCHIVE;
```

说明

LONG lArchiveId; //归档的唯一 ID

指示值：0 时，会自动使用唯一 ID 并通过“AddArchive”返回。然后，返回的 ID 被指定给“AddField”（当返回 ID 为 -1 时，不能创建归档）。

用于编写“uaCONFIGARCHIVEA”句柄程序的结构

构造“uaCONFIGARCHIVE”

```
typedef struct tagUACONFIGARCHIVE
{
LONG lArchiveId; // 归档的唯一 ID
LONG lPosition; // 归档的位置（顺序）
CHAR szName[UA_MAXLEN_NAME+1]; // 归档名称最多可以有 20 个字符
CHAR szAlias[UA_MAXLEN_ALIAS+1]; // 别名最多可以有 50 个字符
```

```
LONG lType;UA_ARCHIVETYPE_UNLIMITED // 归档类型不限
UA_ARCHIVETYPE_LIMITED // 归档类型有限制
LONG lNumRecs; // 数据集的最大数量
LONG lCommType;
UA_COMMTYPE_NONE  // 无通讯
UA_COMMTYPE_RAW // 通过原始数据进行通讯
UA_COMMTYPE_DIRECT //  通过 DM 变量进行通讯
CHAR szPLCID[UA_MAXLEN_PLCID+1]; // 原始数据变量的 PLCID
CHAR szDMVarName[UA_MAXLEN_DMVARNAME+1]; // 原始数据变量的名称
CHAR szIDVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量 ID
CHAR szJobVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量作业
CHAR szFieldVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量域
CHAR szValueVar[UA_MAXLEN_DMVARNAME+1]; // 控制变量值
DWORD dwReadRight; // 读访问权限
DWORD dwWriteRight; // 写访问权限
DWORD dwFlags; UA_ARCHIVEFLAG_ACCESS // 上次访问
UA_ARCHIVEFLAG_USER // 上个用户
} UACONFIGARCHIVE;
```

说明

LONG lArchiveId; //归档的唯一 ID

指示值：0 时，会自动使用唯一 ID 并通过“ConfigArchive”返回。然后，返回的 ID 被指定给“ConfigField”（当返回 ID 为 -1 时，不能创建归档）。

2.3.7.4 常规运行系统函数

常规运行系统函数

用于用户归档的函数

这些函数为运行系统操作打开和关闭用户归档和视图。

函数	描述
uaConnect	建立与用户归档的连接。该连接对运行系统中的所有用户归档均有效
uaDisconnect	如存在与用户归档（运行系统）的连接，将终止该连接。
uaGetLocalEvents	读取本地事件
uaIsActive	确定运行系统是否为活动状态
uaOpenArchives	确定打开的用户归档的数量
uaOpenViews	确定打开的视图的数量
uaQueryArchive	建立与用户归档的连接
uaQueryArchiveByName	与使用该归档名称的用户归档建立连接
uaReleaseArchive	终止与用户归档的连接
uaSetLocalEvents	设置本地事件
uaUsers	查找活动连接或活动用户的数量。

uaConnect

描述：

建立与用户归档的连接（运行系统）。

```
BOOL uaConnect (
 UAHCONNECT* phConnect )
```

参数：

```
UAHCONNECT* phConnect
```

新连接用户归档的句柄的指针

返回值：

TRUE：已成功连接用户归档

FALSE：错误

uaDisconnect

描述：

如存在与用户归档（运行系统）的连接，将终止该连接。

```
BOOL uaDisconnect (
 UAHCONNECT hConnect )
```

参数：

```
UAHCONNECT hConnect
```

已连接的用户归档的句柄（运行系统）。该句柄通过“uaConnect”生成。

返回值：

TRUE：成功断开与用户归档的连接

FALSE：错误

uaGetLocalEvents

描述：

读取本地事件

```
BOOL uaGetLocalEvents
( UAHCONNECT hConnect )
```

参数：

```
UAHCONNECT hConnect
```

用户归档的句柄。该句柄通过“uaConnect”生成。

返回值：

本地事件（bLocalEvents）

uaIsActive

描述：

检查运行系统是否为活动状态。

```
BOOL uaIsActive (
UAHCONNECT hConnect )
```

参数：

```
UAHCONNECT hConnect
```

用户归档的句柄。该句柄通过“uaConnect”生成。

返回值：

TRUE：运行系统中打开了一个或多个用户归档

FALSE：运行系统中没有用户归档

uaOpenArchives

描述：

查询运行系统中打开了多少个用户归档。

```
LONG uaOpenArchives (
UAHCONNECT hConnect )
```

参数：

```
UAHCONNECT hConnect
```

用户归档的句柄。该句柄通过“uaConnect”生成。

返回值：

当前打开的用户归档的数量。

uaOpenViews

描述：

查询运行系统中打开了多少个用户归档视图。

```
LONG uaOpenViews (
UAHCONNECT hConnect )
```

参数：

```
UAHCONNECT hConnect
```

视图的句柄。该句柄通过“uaConnect”生成。

返回值：

当前打开的视图的数量。

uaQueryArchive

描述：

建立与用户归档（用于运行系统操作）的连接。UaQueryArchive 创建句柄 UAHARCHIVE。

```
BOOL uaQueryArchive (
UAHCONNECT hConnect,
 LONG lArchive,
 UAHARCHIVE* phArchive )
```

参数：

```
UAHCONNECT hConnect
```

已连接的用户归档的句柄（运行系统）。该句柄通过“uaConnect”生成。

```
LONG lArchive
```

归档索引（0... uaGetNumArchives() -1）

```
UAHARCHIVE* phArchive
```

指向归档句柄的指针。

返回值：

TRUE：成功生成用户归档句柄

FALSE：错误

注释：

如在客户机项目中使用查看冗余服务器对的用户归档函数，则更改主机时用户归档连接不能自动切换至新主机。在这种情况下，所有用户归档调用将传送 LastError UA_ERROR_SERVER_UNKNOWN = 1004。结果，用户程序必须运行新的 uaQueryArchive() 或 uaQueryArchiveByName() 和 uaArchiveOpen()。

uaQueryArchiveByName

描述：

用归档名称建立与用户归档（用于运行系统操作）的连接。UaQueryArchiveByName 创建用户归档句柄 UAHARCHIVE。

```
BOOL uaQueryArchiveByName (
UAHCONNECT hConnect,
 LPCSTR pszName,
 UAHARCHIVE* phArchive )
```

参数：

```
UAHCONNECT hConnect
```

已连接的用户归档的句柄（运行系统）。该句柄通过“uaConnect”生成。

```
LPCSTR pszName
```

用户归档的名称。对于客户机项目，如使用的是缺省服务器以外的服务器，则可添加服务器前缀“::”作为与归档名称的分隔符。

```
UAHARCHIVE* phArchive
```

指向归档句柄的指针。

返回值：

TRUE：已成功生成用户归档句柄

FALSE：错误

注释：

如在客户机项目中使用查看冗余服务器对的用户归档函数，则更改主机时用户归档连接不能自动切换至新主机。在这种情况下，所有用户归档调用将传送 LastError UA_ERROR_SERVER_UNKNOWN = 1004。结果，用户程序必须运行新的 uaQueryArchive() 或 uaQueryArchiveByName() 和 uaArchiveOpen()。

uaReleaseArchive

描述：

终止与当前用户归档的连接。

```
BOOL uaReleaseArchive (
 UAHARCHIVE hArchive )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。 该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE： 成功断开与用户归档的连接。

FALSE： 错误

注释

成功解除连接后必须将句柄“hArchive”设置为“NULL”。这样，如果还在使用已经失效的句柄，而 COM 界面中又没有对应的函数，就一定会显示错误“UA_ERROR_INVALID_HANDLE”。

uaSetLocalEvents

描述：

设置本地事件。

```
void uaSetLocalEvents (
UAHCONNECT hConnect
BOOL bLocalEvents )
```

参数：

`UAHCONNECT hConnect`

用户归档的句柄。该句柄通过“uaConnect”生成。

`BOOL bLocalEvents`

本地事件

uaUsers

描述：

运行系统函数“uaUsers”返回使用“uaConnect”与用户归档连接的所有用户的数量。请注意，这也将包括用户归档的 WinCC 内部调用，但不包括由用户启动的调用（比如，来自脚本的调用等）。

```
LONG uaUsers (
UAHCONNECT hConnect )
```

参数：

`UAHCONNECT hConnect`

用户归档的句柄。该句柄通过“uaConnect”生成。

返回值：

活动连接或用户的数量。

2.3.7.5 归档专用的运行系统函数

归档专用的运行系统函数

用户归档函数

这些函数用于在运行系统中操作用户归档和视图。

函数	描述
uaArchiveClose	终止与当前用户归档的连接。
uaArchiveDelete	从当前用户归档中删除数据记录
uaArchiveExport	导出当前用户归档
uaArchiveGetCount	读取数据记录的数量。
uaArchiveGetFieldLength	读取当前域的长度
uaArchiveGetFieldName	读取当前域的名称
uaArchiveGetFields	读取域的数量
uaArchiveGetFieldType	读取当前域的类型
uaArchiveGetFieldValueDate	读取日期和时间并将其放入当前的数据域
uaArchiveGetFieldValueDouble	读取当前数据域的“双精度”值
uaArchiveGetFieldValueFloat	读取当前数据域的“浮点”值
uaArchiveGetFieldValueLong	读取当前数据域的“长整型”值
uaArchiveGetFieldValueString	读取当前数据域的“字符串”值
uaArchiveGetFilter	读取当前数据域的过滤器
uaArchiveGetID	读取当前数据域的 ID
uaArchiveGetName	读取当前数据域的名称
uaArchiveGetSort	读取当前数据域的排序
uaArchiveImport	导入用户归档
uaArchiveInsert	将新数据记录插入用户归档中
uaArchiveMoveFirst	跳转到第一条数据记录
uaArchiveMoveLast	跳转到最后一条数据记录

uaArchiveMoveNext	跳转到下一条数据记录
uaArchiveMovePrevious	跳转到前一条数据记录
uaArchiveOpen	建立与当前用户归档的连接
uaArchiveReadTagValues	读取变量值
uaArchiveReadTagValuesByName	基于名称读取变量值
uaArchiveRequery	新查询
uaArchiveSetFieldValueDate	写入当前数据域
uaArchiveSetFieldValueDouble	写入当前数据域的“双精度”值
uaArchiveSetFieldValueFloat	写入当前数据域的“浮点”值
uaArchiveSetFieldValueLong	写入当前数据域的“长整型”值
uaArchiveSetFieldValueString	写入当前数据域的“字符串”值
uaArchiveSetFilter	设置过滤器
uaArchiveSetSort	设置排序标准
uaArchiveUpdate	更新打开的用户归档。
uaArchiveWriteTagValues	将当前数据记录的值写入变量中
uaArchiveWriteTagValuesByName	基于名称将当前数据记录的值写入变量中

uaArchiveClose

描述：

终止与当前用户归档的连接。

```
BOOL uaArchiveClose (
UAHARCHIVE hArchive )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：已成功关闭用户归档

FALSE：错误

uaArchiveDelete

删除归档

WinCC 脚本语言的标准函数！AL(UAEditorScriptAllgemein,3,"","")

用户归档标准函数的参考！AL(UAEditorReferenzenScript,3,"","")

描述：

从用户归档中删除数据。然而，已组态的用户归档将保留下来。

```
BOOL uaArchiveDelete (
 UAHARCHIVE hArchive,
 LPCSTR pszWhere )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LPCSTR pszWhere`

该字符串包含 SQL 选择表达式。它定义要删除哪些数据记录。该表达式与“DELETE FROM <archive> WHERE pszWhere”SQL 语句中的表达式相似。

□□□ 如字符串为空，则将删除整个用户归档。

返回值：

TRUE：已成功删除用户归档

FALSE：错误

uaArchiveExport

描述：

将当前用户归档以 CSV 格式导出至另一个归档中。

```
BOOL uaArchiveExport (
 UAHARCHIVE hArchive,
```

```
 LPCSTR pszDestination,
 LONG lType,
 LONG lOptions )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LPCSTR pszDestination

目标归档的文件名。 在客户机上调用该函数时，该参数指定的路径表示服务器计算机。

LONG lType

目标归档的数据格式。两种格式可用：

UA_FILETYPE_DEFAULT = 0：默认文件格式 = CSV

UA_FILETYPE_CSV = 1：CSV 文件格式

LONG lOptions

选项

保留供以后扩展。必须为 0。

返回值：

TRUE：已成功导出用户归档

FALSE：出错

uaArchiveGetCount

描述：

读取数据记录的数量。

```
LONG uaArchiveGetCount(
 UAHARCHIVE hArchive,
 LONG * plCount )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

```
LONG plCount
```

指向用于储存数据记录数量的变量的指针。

返回值：

数据记录的数量。

0 = 归档为空或发生了错误。必须用 uaGetLastError() 进行查询。

uaArchiveGetFieldLength

描述：

读取当前数据记录中域的长度。

```
LONG uaArchiveGetFieldLength(
 UAHARCHIVE hArchive,
 LONG lField )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

```
LONG lField
```

域编号，其中 lField = 1 表示指向第一个域的地址。

返回值：

当前域的长度。

uaArchiveGetFieldName

描述：

读取当前数据记录中域的名称。

```
VOID uaArchiveGetFieldName (
 UAHARCHIVE hArchive,
 LONG lField,
 LPCSTR pszName,
 LONG cMaxLen )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LONG lField`

域编号，其中 lField = 1 表示指向第一个域的地址。

`LPCSTR pszName`

域名称

`LONG cMaxLen`

最大长度

uaArchiveGetFields

描述：

读取已组态数据域（包括“ID”、“上个用户”和“上次访问”域）的数量。在运行系统调用中，已组态域的索引标记为 1 至 N。域 ID 索引为 0。“上个用户”和“上次访问”域附加在已组态域的末尾。

```
LONG uaArchiveGetFields (
 UAHARCHIVE hArchive )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

已组态域的数量。

uaArchiveGetFieldType

描述：

读取当前数据记录中域的类型。

```
LONG uaArchiveGetFieldType (
UAHARCHIVE hArchive,
LONG lField )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

```
LONG lField
```

域编号，其中 lField = 1 表示指向第一个域的地址。

返回值：

当前域的类型。

域类型的符号定义是：

UA_FIELDTYPE_INTEGER

UA_FIELDTYPE_DOUBLE

UA_FIELDTYPE_STRING

UA_FIELDTYPE_DATETIME

uaArchiveGetFieldValueDate

描述：

读取当前数据记录中域的日期和时间。

```
BOOL uaArchiveGetFieldValueDate (
UAHARCHIVE hArchive,
LONG lField,
LPSYSTEMTIME pstDateTime )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LONG lField`

域编号，其中 lField = 1 表示指向第一个域的地址。

`LPSYSTEMTIME pstDateTime`

SYSTEMTIME 类型变量的指针。

返回值：

TRUE：成功读取日期和时间

FALSE：错误

uaArchiveGetFieldValueDouble

描述：

读取当前数据记录中域的双精度值。

```
BOOL uaArchiveGetFieldValueDouble (
UAHARCHIVE hArchive,
LONG lField,
double* pdValue )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

```
LONG lField
```

域编号，其中 lField = 1 表示指向第一个域的地址。

```
double* pdValue
```

当前域内容的变量的指针。

返回值：

TRUE：成功读取域值

FALSE：错误

uaArchiveGetFieldValueFloat

描述：

读取当前数据记录中域的浮点值。

```
BOOL uaArchiveGetFieldValueFloat (
UAHARCHIVE hArchive,
LONG lField,
FLOAT* pfValue )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

```
LONG lField
```

域编号，其中 lField = 1 表示指向第一个域的地址。

```
FLOAT* pfValue
```

当前域内容的浮点变量的指针。

返回值：

TRUE：成功读取域值

FALSE：错误

uaArchiveGetFieldValueLong

描述：

读取当前数据记录中域的长整型值。

```
BOOL uaArchiveGetFieldValueLong (
UAHARCHIVE hArchive,
LONG lField,
LONG* pdValue )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LONG lField`

域编号，其中 lField = 1 表示指向第一个域的地址。

`LONG* pdValue`

当前域内容的长整型变量的指针。

返回值：

TRUE：成功读取域值

FALSE：错误

uaArchiveGetFieldValueString

描述：

读取当前数据记录中域的字符串。

```
BOOL uaArchiveGetFieldValueString (
```

```
UAHARCHIVE hArchive,
LONG lField,
LPSTR pszString,
LONG cMaxLen )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LONG lField

域编号，其中 lField = 1 表示指向第一个域的地址。

LPCSTR pszString

域值为字符串。

LONG cMaxLen

字符串的最大长度。

返回值：

TRUE：成功读取域值

FALSE：错误

uaArchiveGetFilter

描述：

读取当前数据记录的过滤器。可在附录中“SQL 语句”下找到附加信息。

```
VOID uaArchiveGetFilter (
 UAHARCHIVE hArchive,
 LPSTR pszFilter,
 LONG cMaxLen )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

```
LPSTR pszFilter
```

读取过滤器。

```
LONG cMaxLen
```

最大长度

uaArchiveGetID

描述：

uaArchiveGetID 读取用户归档的 ID。

```
LONG uaArchiveGetID (
 UAHARCHIVE hArchive )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

用户归档的 ID

uaArchiveGetName

描述：

读取用户归档的名称。

```
VOID uaArchiveGetName (
UAHARCHIVE hArchive,
LPSTR pszName,
LONG cMaxLen )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LPSTR pszName

用户归档名称缓冲区的指针。

LONG cMaxLen

最大长度

实例：

```
char Filling [40];
uaArchiveGetName( hArchive, Filling, 39 );
```

uaArchiveGetSort

描述：

uaArchiveGetSort 读取用户归档的排序。可在附录中“SQL 语句”下找到附加信息。

```
VOID uaArchiveGetSort (
UAHARCHIVE hArchive,
LPSTR pszSort,
LONG cMaxLen )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LPCSTR pszSort

排序...

LONG cMaxLen

最大长度

uaArchiveImport

描述：

uaArchivImport 用 CSV 数据格式导入用户归档。目标归档的结构必须与导入的 CSV 归档结构相同。

```
BOOL uaArchiveImport (
UAHARCHIVE hArchive,
LPCSTR pszSource,
LONG lType,
LONG lOptions )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LPCSTR pszSource`

源归档的文件名。

`LONG lType`

源归档的数据格式。两种格式可用：

UA_FILETYPE_DEFAULT = 0：默认文件格式 = CSV

UA_FILETYPE_CSV = 1：CSV 文件格式

`LONG lOptions`

保留供以后扩展。必须为 0。

返回值：

TRUE：已成功导入用户归档

FALSE：错误

uaArchiveInsert

描述：

将本地数据记录缓冲区插入当前数据库中。要在新数据记录中获得有用数据，则必须在调用“uaArchiveInsert”前，使用“uaArchiveSetFieldValue...”函数写本地数据记录缓冲区的域。
必须用函数“uaArchiveSetFieldValueLong”将内部列“ID”写入当前数据集中。

```
BOOL uaArchiveInsert (
UAHARCHIVE hArchive )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：成功插入数据记录

uaArchiveMoveFirst

描述：

跳转到第一条数据记录。

```
BOOL uaArchiveMoveFirst (
UAHARCHIVE hArchive )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：成功在用户归档中进行移动

FALSE：错误

uaArchiveMoveLast

描述：

跳转到最后一条数据记录。

```
BOOL uaArchiveMoveLast (
UAHARCHIVE hArchive )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：成功在用户归档中进行移动

FALSE：错误

uaArchiveMoveNext

描述：

跳转到下一条数据记录。

```
BOOL uaArchiveMoveNext (
UAHARCHIVE hArchive )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：成功在用户归档中进行移动

FALSE：错误

uaArchiveMovePrevious

描述：

跳转到前一条数据记录。

```
BOOL uaArchiveMovePrevious (
UAHARCHIVE hArchive )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：成功在用户归档中进行移动

FALSE：错误

uaArchiveOpen

描述：

uaArchiveOpen 用于打开现有的用户归档。执行与该用户归档有关的读写操作时，必须始终调用 uaArchiveOpen。例如，必须先调用 uaArchiveOpen，然后才能调用 uaArchiveMoveFirst、uaArchiveMoveLast、uaArchiveMoveNext、uaArchiveMovePrevious、uaArchiveDelete、uaArchiveUpdate、uaArchiveInsert 函数。

说明

对用户归档进行排序和过滤

可以对用户归档使用“uaArchiveSetSort”和“uaArchiveSetFilter”函数，而无需使用“uaArchiveOpen”打开用户归档。

```
BOOL uaArchiveOpen (
UAHARCHIVE hArchive )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：已成功打开用户归档

FALSE：错误

uaArchiveReadTagValues

描述：

从域变量中读取当前值。

```
BOOL uaArchiveReadTagValues (
UAHARCHIVE hArchive,
LONG* pnFields,
LONG cFields,
LONG lOptions )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LONG* pnFields

保留供将来应用 (NULL)。

LONG cFields

已传送的域索引数量（数组 pnFields 的大小）。

保留供将来使用 (0)。

LONG lOptions

保留供将来使用 (0)。

如果 lOptions 值为其它值，则数据在指针的位置插入。

返回值：

TRUE：成功实现用户归档的读取

FALSE：错误

uaArchiveReadTagValuesByName

描述：

读取当前数据中的变量值。

```
BOOL uaArchiveReadTagValuesByName (
UAHARCHIVE hArchive,
LPCSTR pszFields,
LONG lOptions )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LPCSTR pszFields`

保留供将来应用 (NULL)。

`LONG lOptions`

保留供将来使用 (0)。

返回值：

TRUE：成功实现用户归档的读取

FALSE：错误

uaArchiveRequery

描述：

调用 uaArchiveSetFilter 和 uaArchiveSetSort 后，必须用 uaArchiveRequery 重新加载归档。

说明

对用户归档进行排序和过滤

可以对用户归档使用“uaArchiveSetSort”和“uaArchiveSetFilter”函数，而无需使用“uaArchiveOpen”打开用户归档。在此情况下，请不要调用“uaArchiveRequery”函数。

在下列情况下，请调用 uaArchiveRequery：

- 已通过用户归档表格控件输入了内容。
- 已在要传送到表格域的用户归档编辑器中输入了内容。

```
BOOL uaArchiveRequery(
UAHARCHIVE hArchive )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：重新查询成功

FALSE：错误

uaArchiveSetFieldValueDate

描述：

将日期和时间写入当前数据记录的域中。

```
BOOL uaArchiveSetFieldValueDate (
UAHARCHIVE hArchive,
```

```
LONG lField,
LPSYSTEMTIME pstDateTime )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LONG lField`

域编号，其中 lField = 1 表示指向第一个已组态域的地址。lField = 0 表示指向 ID 域的地址。

`LPSYSTEMTIME pstDateTime`

日期和时间

返回值：

TRUE：成功写入日期和时间

FALSE：错误

uaArchiveSetFieldValueDouble

描述：

将双精度值写入当前数据记录的域中。

```
BOOL uaArchiveSetFieldValueDouble (
UAHARCHIVE hArchive,
LONG lField,
double dValue )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LONG lField`

域编号，其中 lField = 1 表示指向第一个已组态域的地址。lField = 0 表示指向 ID 域的地址。

```
double dValue
```

域值

返回值：

TRUE：成功写入域值

FALSE：错误

uaArchiveSetFieldValueFloat

描述：

将浮点值写入当前数据记录的域中。

```
BOOL uaArchiveSetFieldValueFloat (
UAHARCHIVE hArchive,
LONG lField,
float fValue )
```

参数：

```
UAHARCHIVE hArchive
```

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

```
LONG lField
```

域编号，其中 lField = 1 表示指向第一个已组态域的地址。lField = 0 表示指向 ID 域的地址。

```
float fValue
```

域值

返回值：

TRUE：成功写入域值

FALSE：错误

uaArchiveSetFieldValueLong

描述：

将长整型值写入当前数据记录的域中。

```
BOOL uaArchiveSetFieldValueLong (
UAHARCHIVE hArchive,
LONG lField,
LONG dValue )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LONG lField`

域编号，其中 lField = 1 表示指向第一个已组态域的地址。lField = 0 表示指向 ID 域的地址。

`LONG dValue`

域值

返回值：

TRUE：成功写入域值

FALSE：错误

uaArchiveSetFieldValueString

描述：

将字符串写入当前数据记录的域中。

```
BOOL uaArchiveSetFieldValueString (
UAHARCHIVE hArchive,
LONG lField,
LPCSTR pszString )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LONG lField

域编号，其中 lField = 1 表示指向第一个已组态域的地址。lField = 0 表示指向 ID 域的地址。

LPCSTR pszString

域值

返回值：

TRUE：成功写入域值

FALSE：错误

uaArchiveSetFilter

描述：

设置过滤器。无需使用“uaArchiveOpen”打开用户归档，就可以调用此函数。

说明

如果已经使用“uaArchiveOpen”打开了用户归档，那么过滤后请使用“uaArchiveRequery”重新加载用户归档。可在附录中“SQL 语句”下找到附加信息。

```
VOID uaArchiveSetFilter (
 UAHARCHIVE hArchive,
 LPSTR pszFilter )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LPSTR pszFilter

要设置的过滤器。

uaArchiveSetSort

描述：

设置用户归档的排序。无需使用“uaArchiveOpen”打开用户归档，就可以调用此函数。

说明

如果已经使用“uaArchiveOpen”打开了用户归档，那么排序后请使用“uaArchiveRequery”重新加载用户归档。可在附录中“SQL 语句”下找到附加信息。

```
BOOL uaArchiveSetSort (
UAHARCHIVE hArchive,
LPSTR pszSort )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LPCSTR pszSort`

排序...

返回值：

TRUE：成功设置排序

FALSE：错误

uaArchiveUpdate

描述：

更新打开的用户归档。将用户归档的所有数据更改都合并到数据库中。用户归档的组态保持不变。

```
BOOL uaArchiveUpdate (
UAHARCHIVE hArchive )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

返回值：

TRUE：已成功更新用户归档

FALSE：错误 "Update_failed" = 106

如出现不一致情况，即会发生该错误，例如：

在域中设置了“域必须包含值”标记，

却未在该域中设置任何值。

uaArchiveWriteTagValues

描述：

将当前数据记录的值写入变量中。

```
BOOL uaArchiveWriteTagValues (
UAHARCHIVE hArchive,
LONG* pnFields,
LONG cFields,
LONG lOptions )
```

参数：

UAHARCHIVE hArchive

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

LONG* pnFields

保留供将来应用 (NULL)。

LONG cFields

保留供将来使用 (0)。

LONG lOptions

保留供将来使用 (0)。

返回值：

TRUE：成功实现用户归档的读取

FALSE：错误

uaArchiveWriteTagValuesByName

描述：

将当前数据记录的值写入变量中。访问基于用户归档和域的名称。

```
BOOL uaArchiveWriteTagValuesByName (
UAHARCHIVE hArchive,
LPCSTR pszFields,
LONG lOptions )
```

参数：

`UAHARCHIVE hArchive`

用户归档的句柄。该句柄通过“uaQueryArchive”或“uaQueryArchiveByName”生成。

`LPCSTR pszFields`

保留供将来应用 (NULL)。

`LONG lOptions`

保留供将来使用 (0)。

返回值：

TRUE：成功实现用户归档的读取

FALSE：错误

2.4 WinCC 用户归档控件

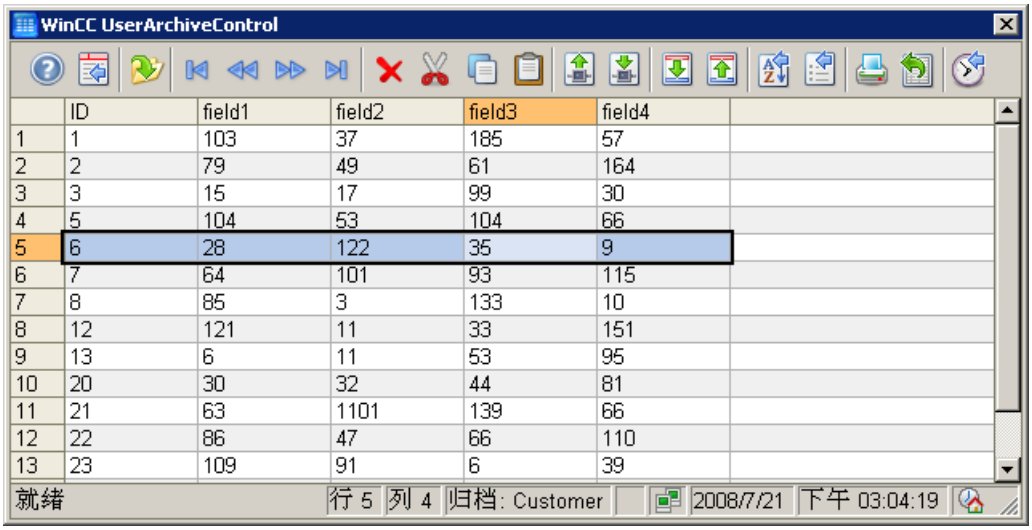
2.4 资源

2.4.1 WinCC 用户归档控件

功能范围

WinCC 用户归档控件提供了访问用户归档的归档和视图的功能。 在运行系统中，可以执行以下操作：

- 创建、删除或修改新的数据记录
- 浏览用户归档
- 读写直接变量链接的变量
- 导入和导出用户归档
- 定义选择标准以便仅显示用户归档的特定部分
- 定义显示列的排序条件



	ID	field1	field2	field3	field4
1	1	103	37	185	57
2	2	79	49	61	164
3	3	15	17	99	30
4	5	104	53	104	66
5	6	28	122	35	9
6	7	64	101	93	115
7	8	85	3	133	10
8	12	121	11	33	151
9	13	6	11	53	95
10	20	30	32	44	81
11	21	63	1101	139	66
12	22	86	47	66	110
13	23	109	91	6	39

属性

在组态期间，将用户归档控件与选定的用户归档和视图相连接。 要进行访问，必须启用用户归档或视图。 如果删除了访问保护，则必须重新在组态对话框中将用户归档控件与用户归档相连接。

当打开用户归档控件的画面时，将查询用户归档或域的访问保护：

- 如果用户没有用户归档的读权限，则不会显示任何数据，但会显示表格中的列标题。
- 如果用户没有域的读权限，则表格中不会显示相应列。
- 如果用户没有用户归档的写权限，用户则不能编辑表格中的数据。
- 如果用户没有域的写权限，用户则不能编辑表格中的相应列。

必须通过对象属性（例如，画面、IO 域或按钮的属性）来对保护性归档的控制变量单独组态访问保护。

2.4.2 组态用户归档控件

2.4.2.1 如何组态用户归档控件

简介

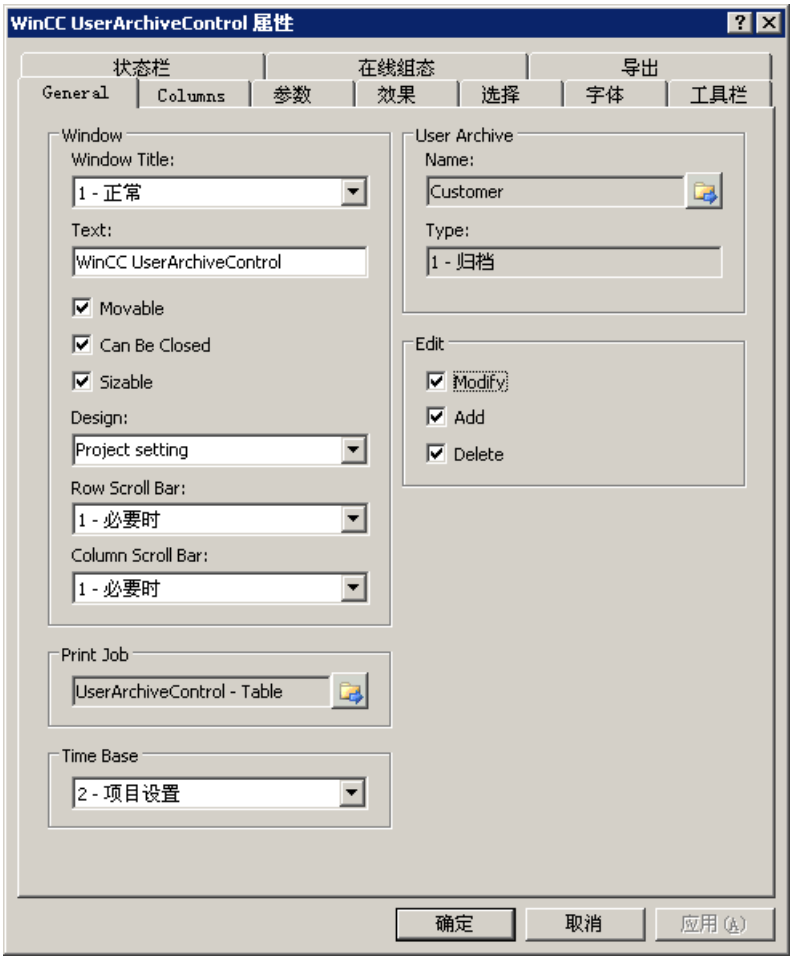
用户归档值显示在运行系统的 ActiveX 控件中。 可以在图形编辑器中为用户归档组态 WinCC 用户归档控件。

要求

- 已在用户归档中组态归档或视图。

组态步骤

1. 将 WinCC 用户归档控件链接到图形编辑器画面。
2. 在“常规”选项卡中，组态用户归档控件的基本属性。
   - 表格窗口属性
   - 控件的常规属性
   - 控件的时间基准
   - 控件中内容的编辑功能



3. 将用户归档控件与用户归档的归档或视图相连接。
4. 定义用户归档控件中表格的内容，在其中组态用户归档中的选定列。
5. 在“参数”、“显示”和“标记”选项卡中，组态表格的显示和属性。
6. 在相应选项卡中，组态表格窗口的工具栏和状态栏
7. 保存该组态。

2.4.2.2 如何定义用户归档控件的内容

简介

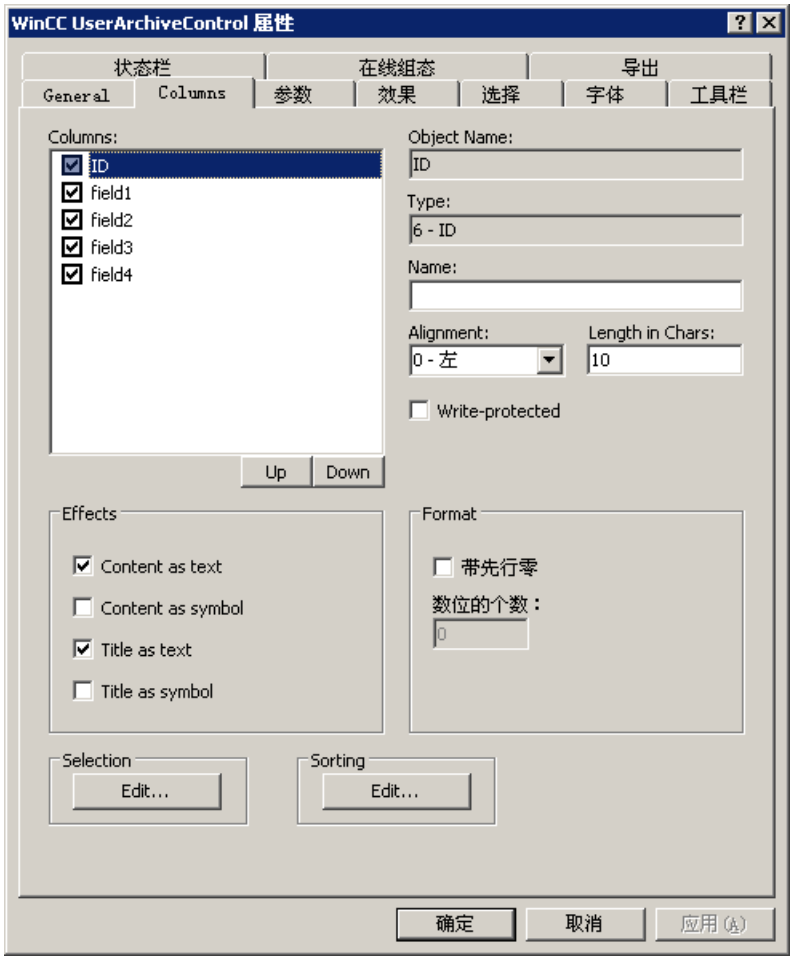
WinCC 用户归档控件在表格中显示所连接用户归档的数据。 表格中的显示内容取决于所选择的用户归档列以及选定的列内容。

要求

- 已经创建一个或多个用户归档或视图。
- 已将用户归档控件与用户归档或视图相连接。

组态用户归档的列。

1. 转到“列”选项卡。



2. 在“列”列表中，可以看到所连接归档或视图的域。 如果列名称前的框被选中，则该列将显示在表格中。 如果不想显示该列，则禁用该复选框。

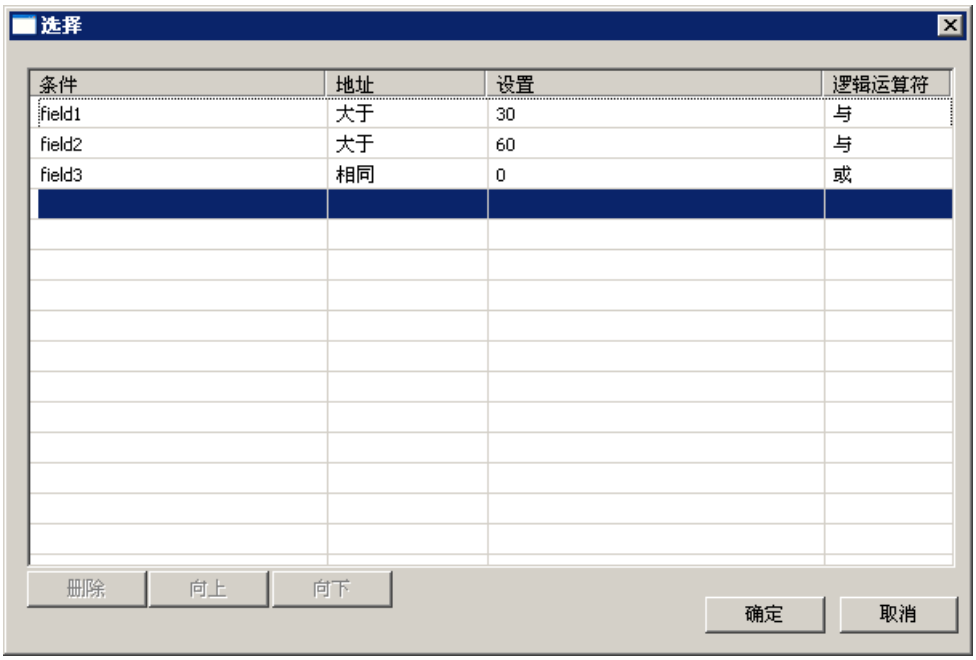
3. 使用“向上”和“向下”按钮确定表格中列的顺序。
4. 选择要组态其属性和格式的列。
5. 如有必要，则更改表格中列的宽度。 在“长度（字符）”域中输入数值。
6. 某些列还可以以符号的形式显示内容和标题。 在“显示”域中确定这些列的显示形式。 可同时显示文本和符号。
7. 保存该组态。

选择要在表格中显示的列内容

在“选择”区域中，组态用于显示列内容的标准。

步骤

1. 单击“编辑...”。 将打开选择对话框。

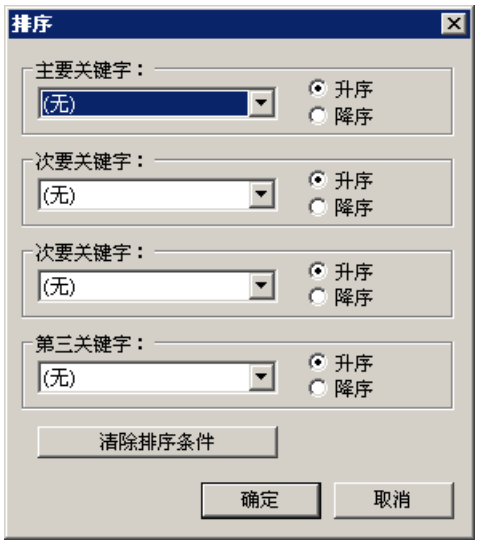


2. 指定显示标准。 有关列选择的更多信息，可参阅。
3. 单击“确定”关闭选择对话框。 所选内容即会在运行系统启动时应用到用户归档控件的表格中。

组态列的排序

在“排序”区域中，组态各列在用户归档控件表格中的排序。 也可使用键功能在运行系统中指定排序标准。

1. 单击“编辑...”。 排序对话框打开。



2. 设置排序顺序。 有关列排序的更多信息，可参阅。
3. 单击“确定”关闭排序对话框。
4. 保存用户归档控件的组态内容。

2.4.2.3 如何组态表格显示

如何组态表格元素的属性

简介

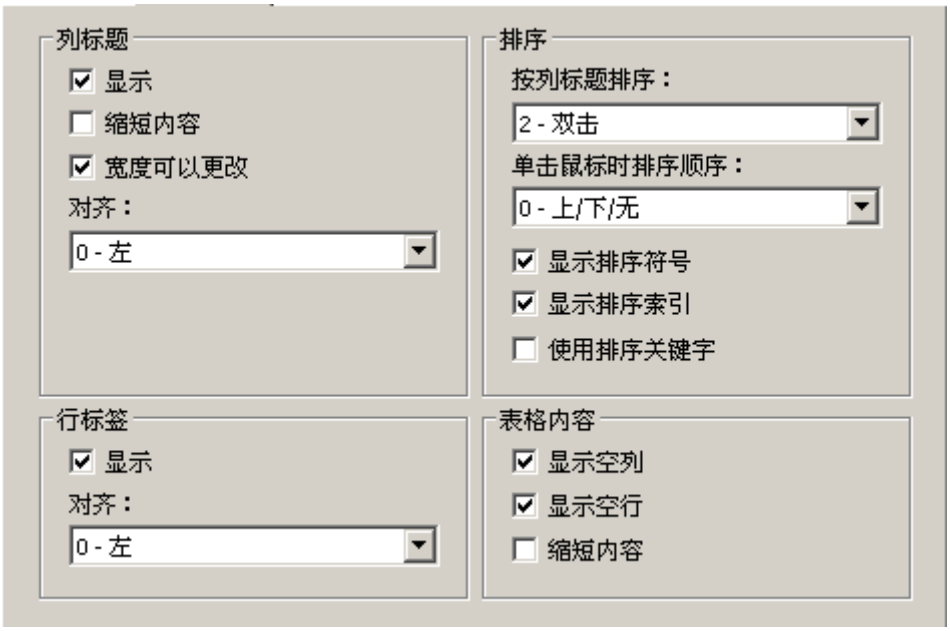
可以在 WinCC 控件中调整表格元素的属性，以满足各种要求。

先决条件

- 打开了图形编辑器，并组态了具有 WinCC 控件的画面。
- 打开了 WinCC 控件的组态对话框。

步骤

1. 转到“参数”选项卡。



2. 指定以下各项的属性
   - 列标题
   - 行标签
   - 排序
   - 表格内容
3. 保存该组态。

如何组态表格元素的颜色

简介

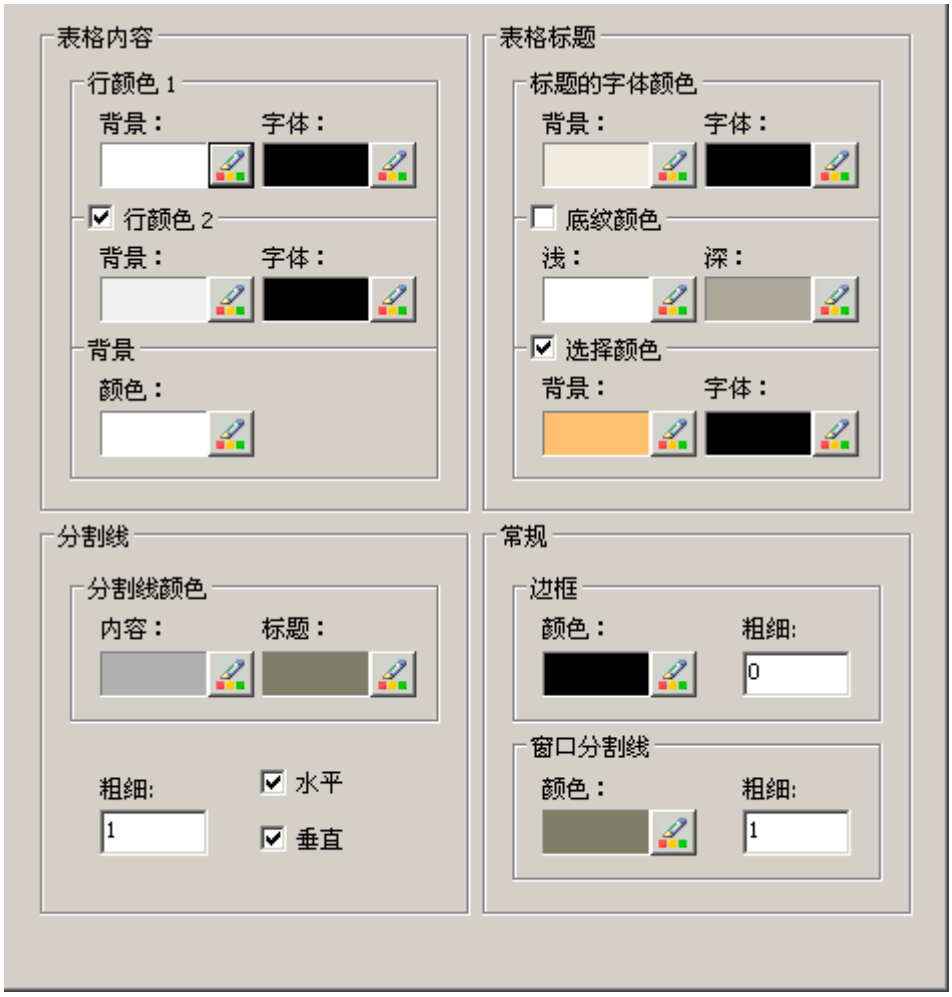
可以在 WinCC 控件中调整表格元素的颜色，以满足各种要求。

先决条件

- 打开了图形编辑器，并组态了具有 WinCC 控件的画面。
- 打开了 WinCC 控件的组态对话框。

步骤

1. 转到“显示”选项卡。



2. 在此，为以下各项定义背景或文本的颜色：
   - 表格内容。 可以为编号为偶数和奇数的行定义不同的颜色，以增强二者间的区别。
   - 表格标题的内容
   - 表格中的分隔线以及表格标题的分隔线
3. 在“常规”区域中，定义以下各项的颜色和线条粗细：
   - 控件边框
   - 控件元素的窗口分隔线
4. 保存该组态。

如何组态选定单元格和行的标记

简介

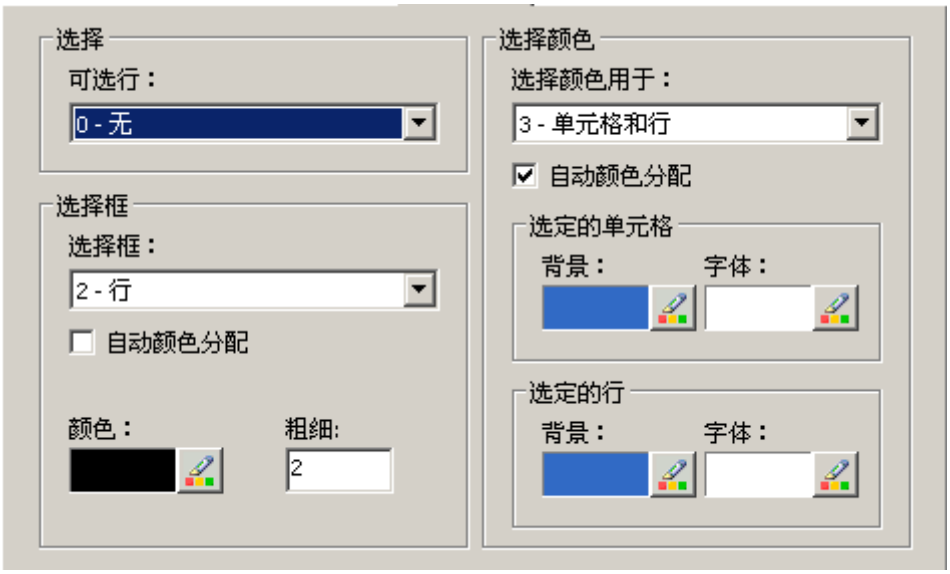
可以在 WinCC 控件中自定义选定单元格和行的标记，以满足各种要求。

先决条件

- 打开了图形编辑器，并组态了具有 WinCC 控件的画面。
- 打开了 WinCC 控件的组态对话框。

步骤

1. 转到“标记”选项卡。



2. 定义是使用鼠标选择行还是仅选择单元格。
3. 组态选取矩形的属性，该选取矩形可以围绕选定的表格单元格或行显示。
4. 组态可选择单元格和/或行的标记颜色。 系统颜色与属性“自动着色”一起用于进行标记。
5. 保存该组态。

如何通过列标题组态排序

简介

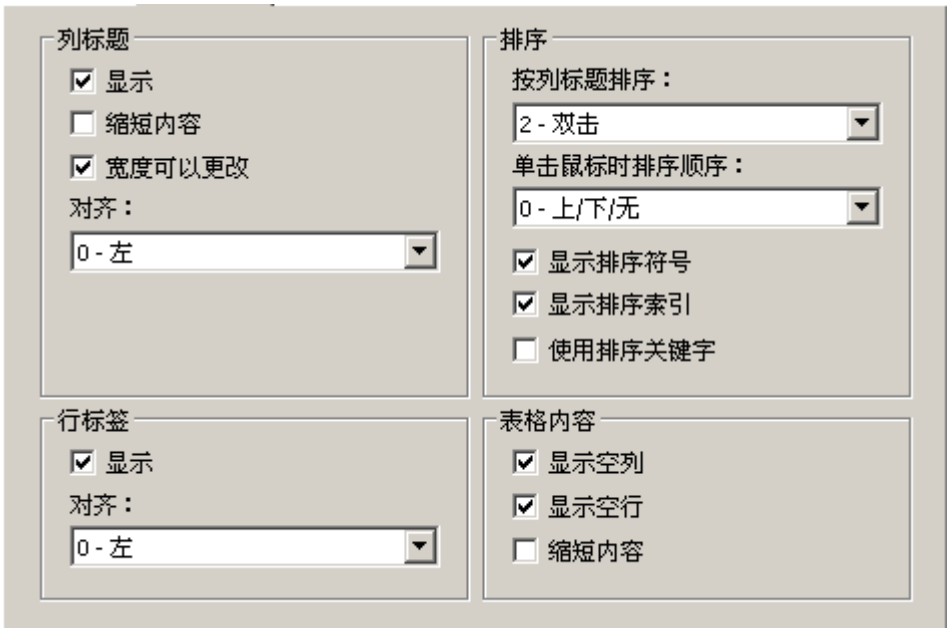
可以在 WinCC 控件中通过表格列标题调整排序顺序，以满足各种要求。

先决条件

- 打开了图形编辑器，并组态了具有 WinCC 控件的画面。
- 打开了 WinCC 控件的组态对话框。

步骤

1. 转到“参数”选项卡。



2. 定义是否启用排序，以及是否启用按列标题进行排序的排序方法。 在 WinCC 报警控件中，仅当禁用了“自动滚动”，才能按列标题进行排序。 在“常规”选项卡中或者使用 WinCC 报警控件的“自动滚动”工具栏图标，都可以禁用“自动滚动”。
3. 通过鼠标单击列标题确定排序顺序。 选择升序、降序或无排序顺序。
4. 以右对齐的方式组态要在列标题中显示的排序图标和索引。 这些会显示排序顺序以及各列的顺序。
5. 激活“使用排序按钮”，以使排序图标作为排序按钮显示在垂直滚动条上方。 单击此排序按钮，可以为选定列激活已组态的排序顺序。 如果缺少垂直滚动条，则不会显示排序按钮。
6. 保存该组态。

2.4.2.4 如何组态工具栏和状态栏

简介

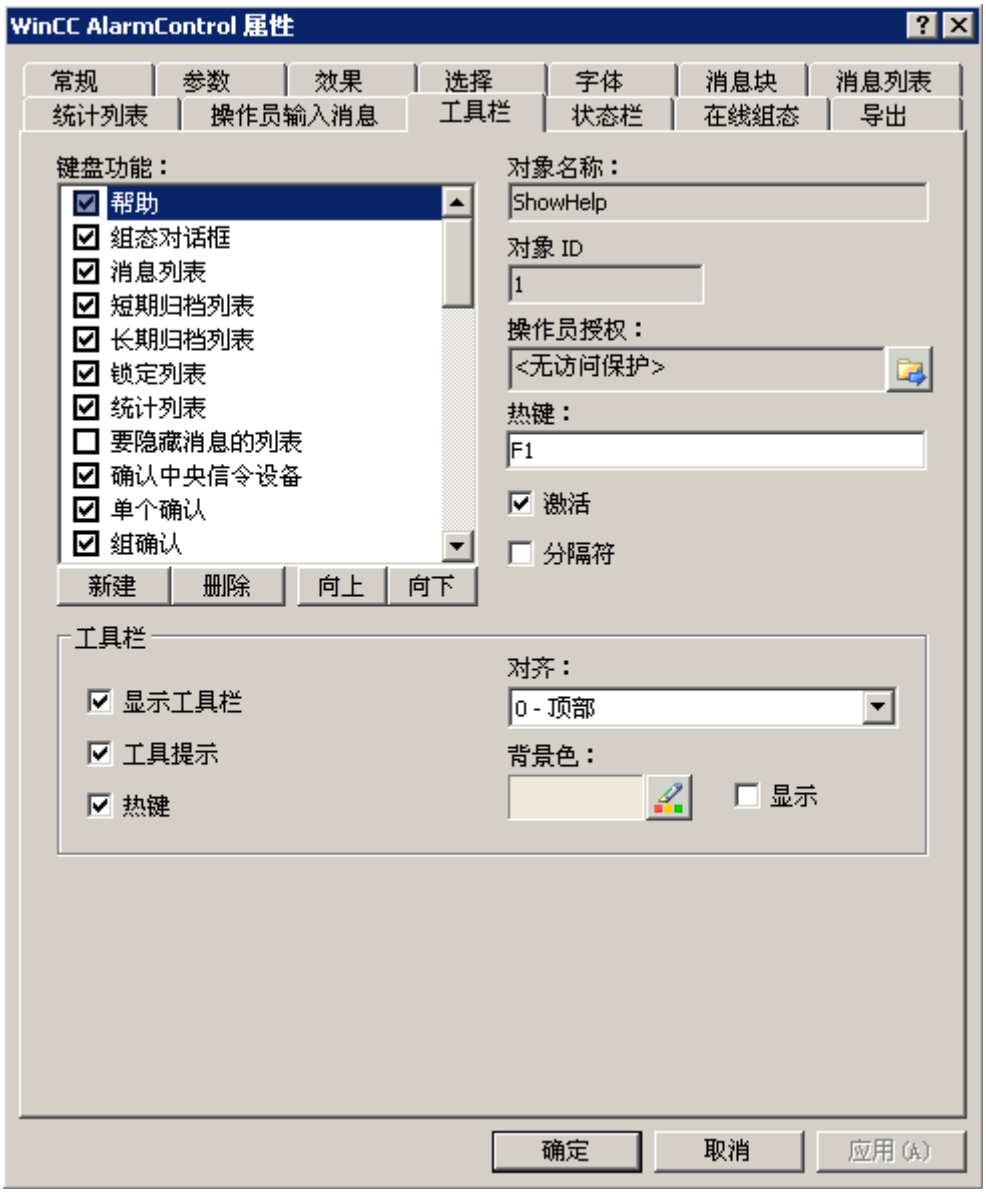
在运行期间，使用工具栏按钮的功能对 WinCC 控件进行操作。 状态栏包含了有关 WinCC 控件当前状态的信息。 可以在进行组态时或者在运行期间调整所有 WinCC 控件的工具栏和状态栏。

先决条件

- 在进行组态时，显示 WinCC 控件的画面在图形编辑器中打开。
- WinCC 控件分配有用于在运行期间打开组态对话框的“组态对话框”按钮功能。
- 打开了 WinCC 控件的组态对话框。

如何组态工具栏

1. 转到“工具栏”选项卡。 以在 WinCC 报警控件中为例：

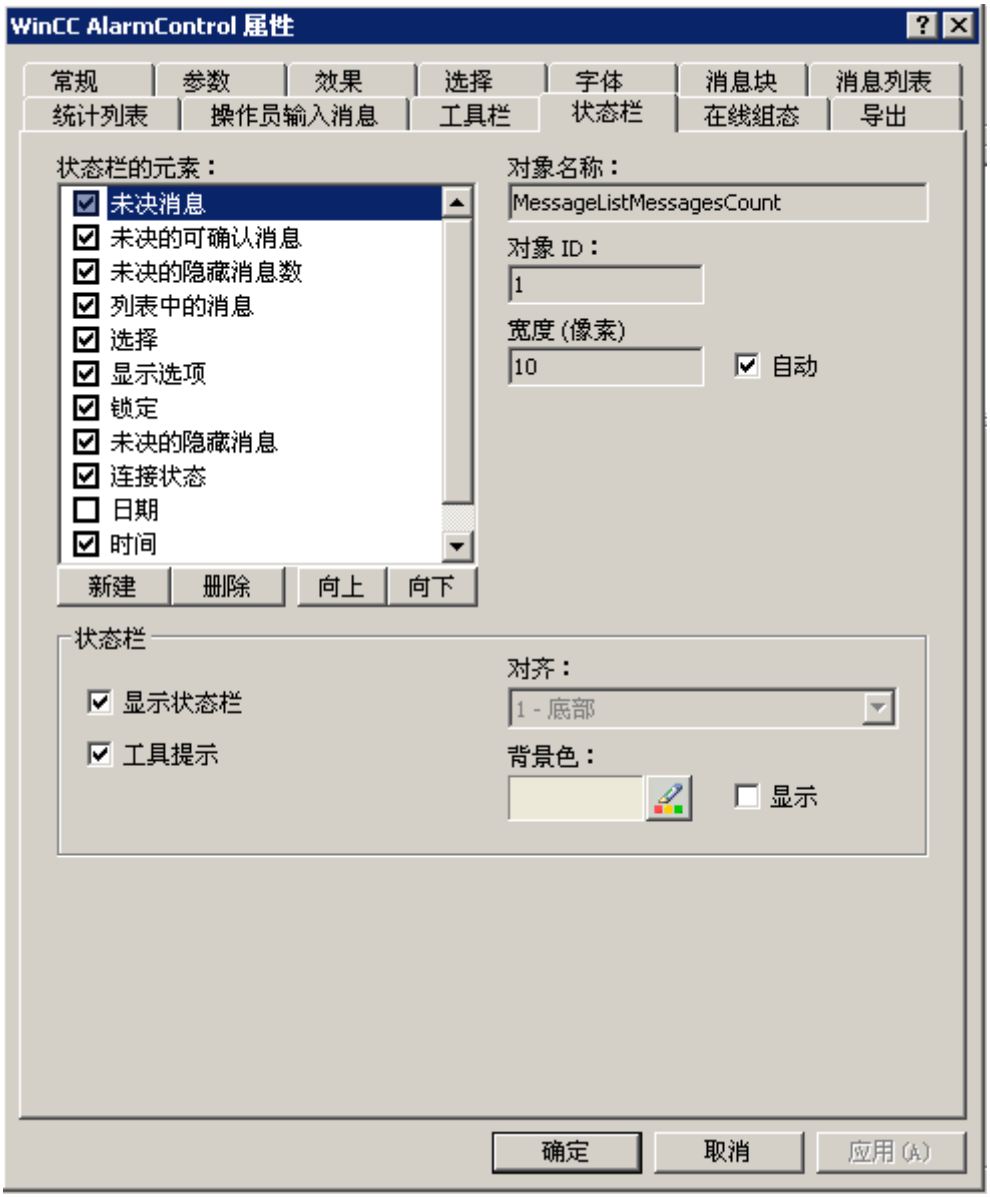


2. 在列表中，激活在运行期间操作 WinCC 控件所需的按钮功能。 有关按钮功能的信息，请参阅“运行系统中的操作”中相应 WinCC 控件的描述。

3. 确定用于显示工具栏中按钮功能的排序顺序。 从列表中选择按钮功能，并使用“向上”和“向下”按钮移动这些功能。
4. 为工具栏按钮的功能定义热键。
5. 任何分配有操作员权限的按钮功能只能在运行系统中由获得授权的用户使用。
6. 如果禁用了已激活按钮功能的“激活”选项，则会在运行系统中显示该按钮功能，但无法对其进行操作。
7. 可以在按钮功能间设置分隔符。 激活按钮功能的“分隔符”选项，以由分隔符对其进行限制。
8. 组态工具栏的常规属性，例如，对齐或背景颜色。

如何组态状态栏

1. 转到“状态栏”选项卡。 以在 WinCC 报警控件中为例：



2. 在状态栏元素的列表中，激活运行系统中所需的元素。 有关状态栏元素的更多信息，请参阅“运行系统中的操作”中对相应 WinCC 控件的描述。
3. 确定用于显示状态栏元素的排序顺序。 从列表中选择元素，并使用“向上”和“向下”按钮对其进行移动。
4. 要调整状态栏元素的宽度，请禁用“自动”选项，并输入宽度的像素值。
5. 组态状态栏的常规属性，例如，对齐或背景颜色。

2.4.2.5 如何导出运行系统数据

简介

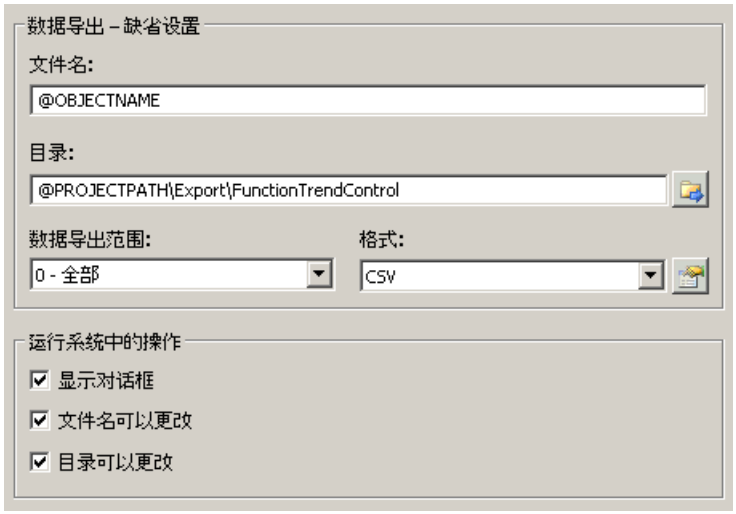
WinCC 控件中显示的运行系统数据可以通过一个按钮功能导出。 在组态对话框中设置运行系统中的数据导出操作。

要求

- 在进行组态时，显示 WinCC 控件的画面在图形编辑器中打开。
- 打开了 WinCC 控件的组态对话框。

如何组态数据导出操作

1. 转到“导出”选项卡。



2. 已在“数据导出默认设置”中输入标准文件名称和标准目录。 在这里指报警控件。 必要时，定义导出文件的文件名称和目录。
3. CSV 当前可用作数据格式。 单击  ，定义 CSV 文件中的分隔符。
4. 定义数据导出的范围：
   – 导出所有运行系统数据
   – 选定的运行系统数据会导出。 该数据导出仅能在 WinCC 控件中以表格形式显示。
5. 组态运行系统中的数据导出操作。 定义以下各项：
   – 是否允许用户重命名该文件或更改目录。
   – 是否在运行系统中显示“数据导出默认设置”对话框。
6. 如果禁用“显示对话框”，“导出数据”按钮功能操作对应的数据就会立即导出到定义的导出文件。

7. 保存该组态。
8. 转到“工具栏”选项卡，激活运行系统的“导出数据”按钮功能。

结果

可以在运行期间使用按钮功能  将所有数据或选定数据导出到定义的文件。

2.4.2.6 如何定义在线组态的结果

简介

用户可以在运行期间对 WinCC 控件进行参数化。 必须定义在线组态的运行结果。

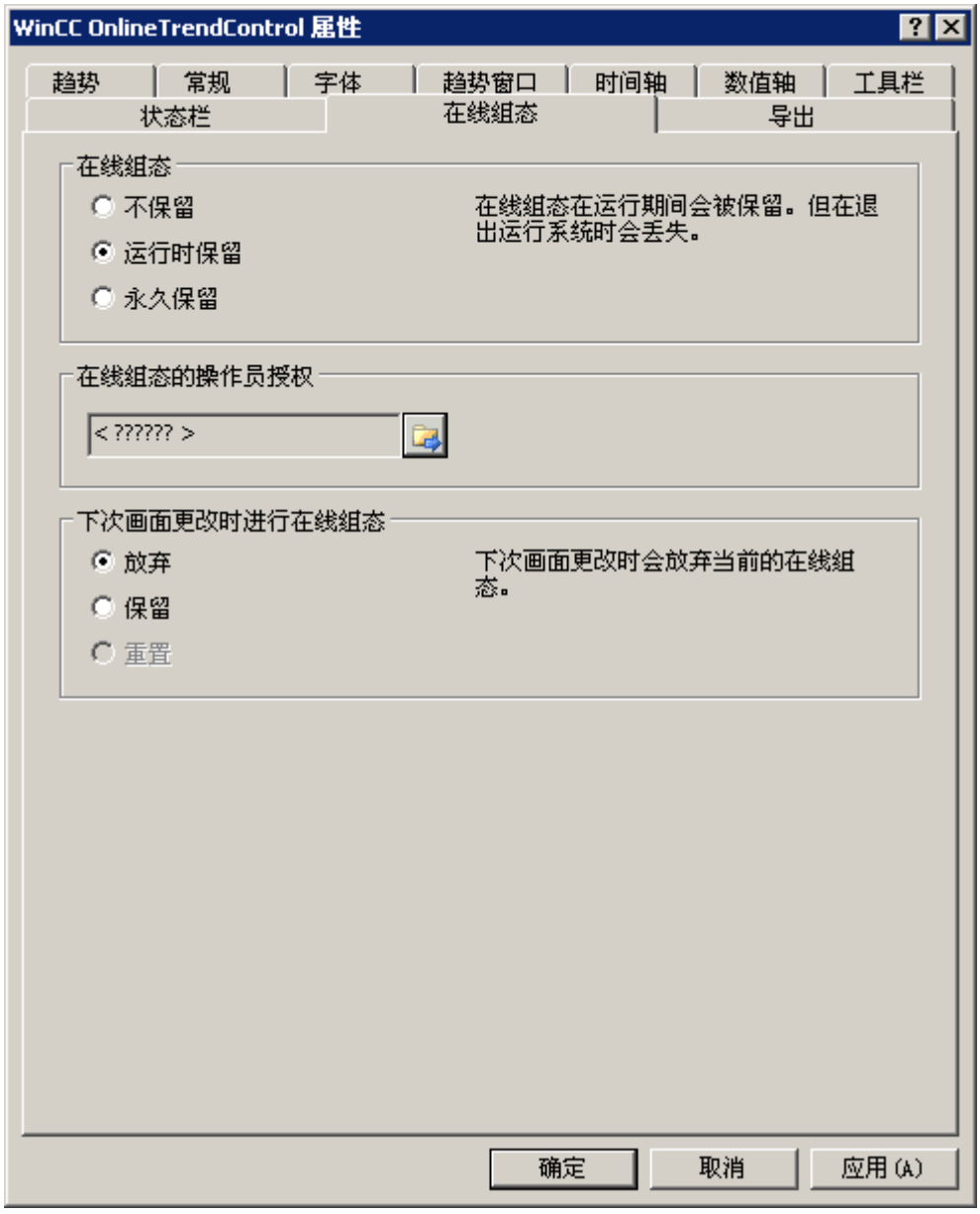
运行期间组态的更改保存在组态系统的独立画面中。 原始画面组态保留在组态系统中。

先决条件

- 在进行组态时，显示 WinCC 控件的画面在图形编辑器中打开。
- 打开了 WinCC 控件的组态对话框。

步骤

1. 转到“在线组态”选项卡。 例如，在在线趋势控件中：



2. 用于设置在线组态默认值的“在线组态”域中的选项按钮仅在组态系统中可用。 这些选项按钮在运行系统中不可用。
选择在线组态的三种结果之一：
   – “不保留”。 在线组态不保留在运行系统中。 此默认设置为运行系统用户禁用了所有选项。 在线组态于下次画面更改以及激活/禁用该项目时会丢失。
   – “在运行期间保留”。 此默认设置为运行系统用户启用了“放弃”、“保留”或“重置”选项。 如果启用“保留”选项，则在线组态于下次画面更改时会保留，但是在激活/禁用该项目时会丢失。

– “永久保留”。 此默认设置为运行系统用户启用了“放弃”、“保留”或“重置”选项。 如果启用“保留”选项，则在线组态于下次画面更改以及激活/禁用该项目时会保留。

3. 为在线组态定义相应的用户权限。
4. 通过设置默认值“运行期间保留”和“永久保留”，可以启用“下次画面更改时的在线组态”中的选项按钮，以便在组态系统中以及运行期间进行操作。 “重置”操作只能在运行期间使用，因为组态系统包含原始组态。
选择“下次画面更改时的在线组态”中的三个结果之一：
   – 如果要在下次画面更改时放弃在线组态，则选择“放弃”。
   – 激活“保留”，以在下次画面更改或激活/禁用该项目时根据默认设置激活在线组态。
   – 激活“重置”，以为运行系统激活一个版本早于组态系统中所存储版本的画面。 所有在线更改会丢失。

   说明

   如果将该画面保存在图形编辑器中，则在运行时或者于在线模式下加载 delta 时，也会替换该画面。 所有在线更改会丢失。

5. 保存该组态。

说明

执行了画面更改后，只会为新用户激活其它组态。

2.4.2.7 如何使用户归档控件的工具栏动态化

简介

自 WinCC V7.0 起，新的 WinCC 用户归档控件不再支持用于操作 WinCC 用户归档控件的默认函数。例如，可通过脚本使用 WinCC 动态类型来操作工具栏的键功能。

概述

自 V7.0 后的 WinCC 控件，由于可以实现工具栏的动态化，因此操作控件时不再需要特殊的函数。 不再支持以前使用的标准函数“Tlg...”。

如果不想通过工具栏来操作控件，则可使用可选的动态类型在“ToolbarButtonClick”对象属性中写入所需按钮的“ID”。

可通过以下方式确定工具栏按钮的“ID”：

- 使用页面“运行系统中用户归档控件的操作”上的表格。
- 通过用户归档控件组态对话框中的“工具栏”选项卡上的域“对象 ID”。

实例： 打开控件的组态对话框

要打开控件的组态对话框，可使用下列动态方式：

- VBScript：
  - ScreenItems("Control1").ToolbarButtonClick = 2
  - 作为“ToolbarButtonClick”属性的替代形式，还可以使用 VBS 方法来操作工具栏：
    ScreenItems("Control1").ShowPropertyDialog
  - 或者，如果支持“智能感知”则可使用以下表示法：
    Dim obj
    Set obj = ScreenItems("Control1")
    obj.ShowPropertyDialog
- C 脚本：
  - SetPropWord(lpszPictureName, "Control1", "ToolbarButtonClick", 2);
- 直接连接
  - 在源的直接连接对话框中，输入“2”作为常量
  - 为对象“Control1”选择属性“ToolButtonClick”以用作目标“画面中的对象”

参见

在运行系统中操作用户归档控件 (页 456)

2.4.3 运行系统中的操作

2.4.3.1 在运行系统中操作用户归档控件

简介

工具栏上的按钮用于在运行系统中操作 WinCC 用户归档控件。 如果不想通过工具栏来操作表格窗口，则可使用可选的动态类型在“ToolbarButtonClick”对象属性中写入所需按钮的“ID”。

概述

符号	描述	ID
	“帮助” 调用 WinCC 用户归档控件的帮助。	1

	“组态对话框” 打开组态对话框，在其中可更改用户归档控件的属性。	2
	“选择数据连接” 打开可以在其中选择用户归档的对话框。 所选用户归档的内容将显示在用户归档控件的表格中。	3
	“第一行” 用户归档的第一个值通过该按钮显示在表格中。	4
	“上一行” 用户归档的上一个值通过该按钮显示在表格中。	5
	“下一行” 用户归档的下一个值通过该按钮显示在表格中。	6
	“最后一行” 用户归档的最后一个值通过该按钮显示在表格中。	7
	“删除行” 删除所标记行的内容。	8
	“剪切行” 剪切所标记行的内容。	9
	“复制行” 复制所标记行的内容。	10
	“插入行” 从所标记行开始插入已复制或剪切下的行的内容。	11
	“读取变量” 该按钮用于读取已连接 WinCC 变量的内容，并将其写入列中。 要使用该按钮，必须在用户归档中激活“通过 WinCC 变量通讯”通讯类型。 列必须与变量相连接。	12
	“写入变量” 该按钮用于将列的内容写入已连接的 WinCC 变量。 要使用该按钮，必须在用户归档中激活“通过 WinCC 变量通讯”通讯类型。 列必须与变量相连接。	13
	“导入归档” 该按钮用于将用户归档的内容导入到用户归档控件的表格中。 用户归档必须以 CSV 文件的形式存在于项目文件夹的“ua”目录下。	14

	“导出归档” 该按钮用于导出用户归档控件中表格的内容。 用户归档以 CSV 文件的形式保存在项目文件夹的“ua”目录下。	15
	“排序对话框” 打开一个对话框，用于设置所显示用户归档列的用户定义排序标准。	16
	“选择对话框” 定义要在表格中显示的用户归档列的选择标准。	17
	“打印” 启动显示值的打印输出。 用于打印的打印作业在组态对话框的“常规”选项卡上定义。	18
	“导出数据” 此按钮用于将全部或选定的运行系统数据导出到“CSV”文件。 如果激活“显示对话框”选项，则会打开一个对话框，从中可查看导出设置并启动导出。 如果具有相应权限，还可以选择要导出的文件和目录。 如果显示了一个对话框，则立即开始将数据导出到预定义文件。	20
	“时间基准对话框” 打开一个对话框，用于为用户归档中使用的时间数据设置时间基准。	19
	“用户定义 1” 显示由用户创建的第一个键功能。 此按钮的功能是用户定义的。	1001

状态栏的可能元素

下列元素可显示在用户归档控件的状态栏中：

符号	名称	描述
归档: Customer	归档名称	显示所选用户归档的名称。
行 45	行	显示标记行的编号。
列 2	列	显示标记列的编号。
08.01.2008	日期	显示系统日期。

符号	名称	描述
09:02:20	时间	显示系统时间。
	时间基准	显示用于显示时间的时间基准。

如何在 WinCC 用户归档控件的表格中导航

可按以下说明在表格中导航：

- 使用“ENTER”键或“向右”光标键进入下一单元格。
- 使用“SHIFT+ENTER”键或“向左”光标键进入上一单元格。
- 使用鼠标在行中单击或使用“向下”光标键进入下一行。
- 使用鼠标在行中单击或使用“向上”光标键进入上一行。

2.4.3.2 在用户归档控件中处理数据：

简介

可以在 WinCC 用户归档控件中编辑数据。具有以下选择：

- 输入新数据
- 更改现有数据
- 删除行
- 剪切、复制和插入行

要求

- 已在组态对话框的“常规”选项卡中允许编辑操作。
- 已在组态对话框的“列”选项卡中禁用要编辑列的“写保护”属性。
- “ID”列不可编辑。
- 如果用户归档控件已连接视图，则不能删除或剪切行。

在表格中输入新数据。

1. 单击  移动到最后一行。 该行将被标记。
2. 双击标记行的第一个单元格。 也可在单元格中按“F2”、“Alt+Enter”或“Ctrl+Enter”。

3. 逐个在单元格中输入值，每次按 Enter 键进行确认。 在行中输入完所有值并已标记另一行后，新的数据记录将写入到用户归档。 使用鼠标单击、使用“ENTER”键或“向上”和“向下”光标键移动到另一行。
4. 可使用“CTRL+C”或“CTRL+X”将标记行的数据复制到剪贴板。 使用“CTRL+V”可将复制的数据插入标记行。

在表格中更改现有数据

1. 单击  或  移动到所需行。 也可使用滚动条移动到所需行。
2. 双击标记行的所需单元格。 也可在单元格中按“F2”、“Alt+Enter”或“Ctrl+Enter”。
3. 逐个在单元格中输入值，每次按 Enter 键进行确认。 在行中输入完所有值并已标记另一行后，更改的数据记录将写入到用户归档。

在表格中删除行

1. 单击  或  移动到所需行。 也可使用滚动条移动到所需行。
2. 单击  删除标记行。

剪切、复制和插入行

1. 单击  或  移动到所需行。 也可使用滚动条移动到所需行。
2. 单击  或  剪切或复制行中的数据。 也可使用组合键“CTRL+ALT+X”或“CTRL+ALT+C”执行该操作。
3. 转到要将数据复制到其中的目标行。 单击  插入剪切或复制的数据。 如果不想覆盖标记行的数据，可移入最后一行插入数据。

2.4.3.3 如何选择用户归档的数据

简介

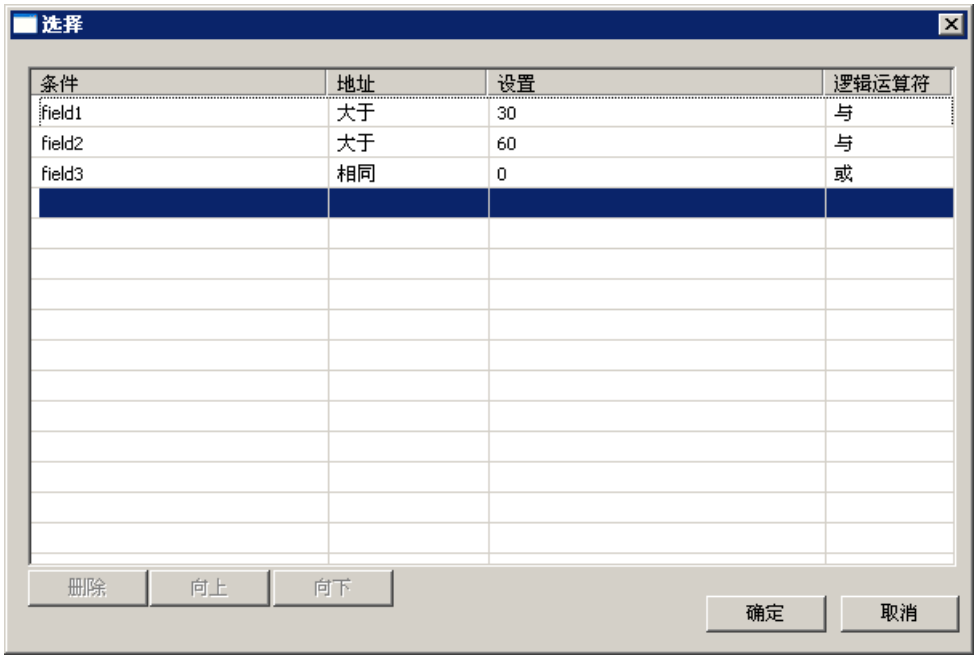
要显示或导出到 WinCC 用户归档控件表格的用户归档内容可以在运行系统中通过选择对话框进行定义。 在选择对话框中，定义所显示用户归档列的选择标准。

先决条件

- 已在用户归档控件的“工具栏”选项卡上组态了“选择对话框”按钮功能。

步骤

1. 单击  运行系统。 “选择”对话框会打开。



2. 双击“标准”列的第一个空白行。 将显示包含用户归档列的列表。 选择所需列，如，“field1”。
3. 在“操作数”列中双击选择操作数。
4. 在“设置”列中双击输入比较值。
5. 在“逻辑运算符”列中双击选择“AND”或“OR”功能。
6. 如果想定义更多标准，请重复上述步骤。
7. 单击“确定”关闭选择对话框。 所选内容即会显示在用户归档控件的表格中。

注意
确保列内容的显示 确保正确使用标准的设置和连接。 链接错误将导致用户归档控件中不显示已连接用户归档的数据。 必须单独测试各标准，然后，在链接标准前还需测试各个被链接标准。 检查所有预期内容是否也都以组合形式显示。 这样可确保所选内容完全显示在用户归档控件中。

2.4.3.4 如何对用户归档数据的显示进行排序

简介

在运行系统中，可按列对用户归档控件中的数据进行排序。 既可使用“排序对话框”按钮功能对列进行排序，也可直接通过列标题执行排序。

说明

在用户归档控件的组态期间，还可以通过单击“列”选项卡上“排序”下的“编辑...”按钮指定排序标准。

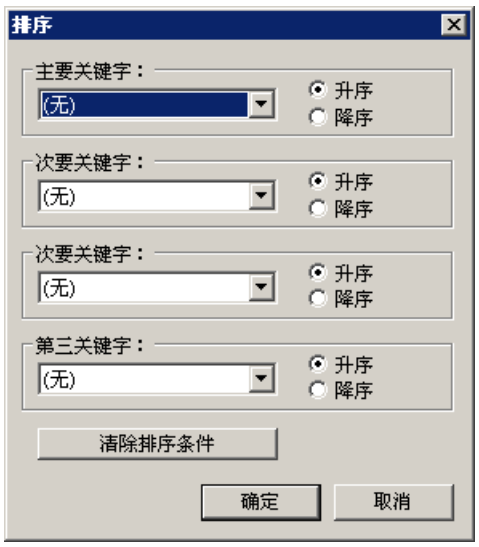
如何使用“排序”对话框排序

要求

- 已在用户归档控件的“工具栏”选项卡上组态了“排序对话框”按钮功能。

步骤

1. 单击“排序对话框”按钮  。
2. 在“主要关键字”域中，选择将首先按其进行排序的已连接用户归档的列。选中相应的复选框，指定排序次序（升序或降序）。 如果期望依据多个列进行排序，则可按照所需的次序从“次要关键字”列表中选择其它列。



如何使用列标题对内容进行排序

当使用列标题进行排序时，可为四个以上的列指定排序次序。 在列标题中显示为右对齐的排序图标和排序索引分别显示列内容的排序次序和排序顺序。

要求

- 已通过在 WinCC 用户归档控件的“参数”选项卡中单击或双击，启用按照“单击列标题排序”列表域进行排序。
- 已激活“显示排序图标”和“显示排序索引”复选框。

步骤

1. 单击期望首先按其进行排序的列的列标题。 将显示排序索引“1”，并显示向上排序图标表示升序排序次序。
2. 如果期望按照降序次序进行排序，则可再次单击列标题。
3. 如果已将排序次序定义为“上/下/无”，则可通过第三次单击来撤消对列的排序。
4. 如果期望根据多个列进行排序，可按照期望的顺序单击各个列标题。

2.5 在 WinCC V7 之前的版本中：WinCC 用户归档表格元素

2.5 资源

2.5.1 用户归档表格元素

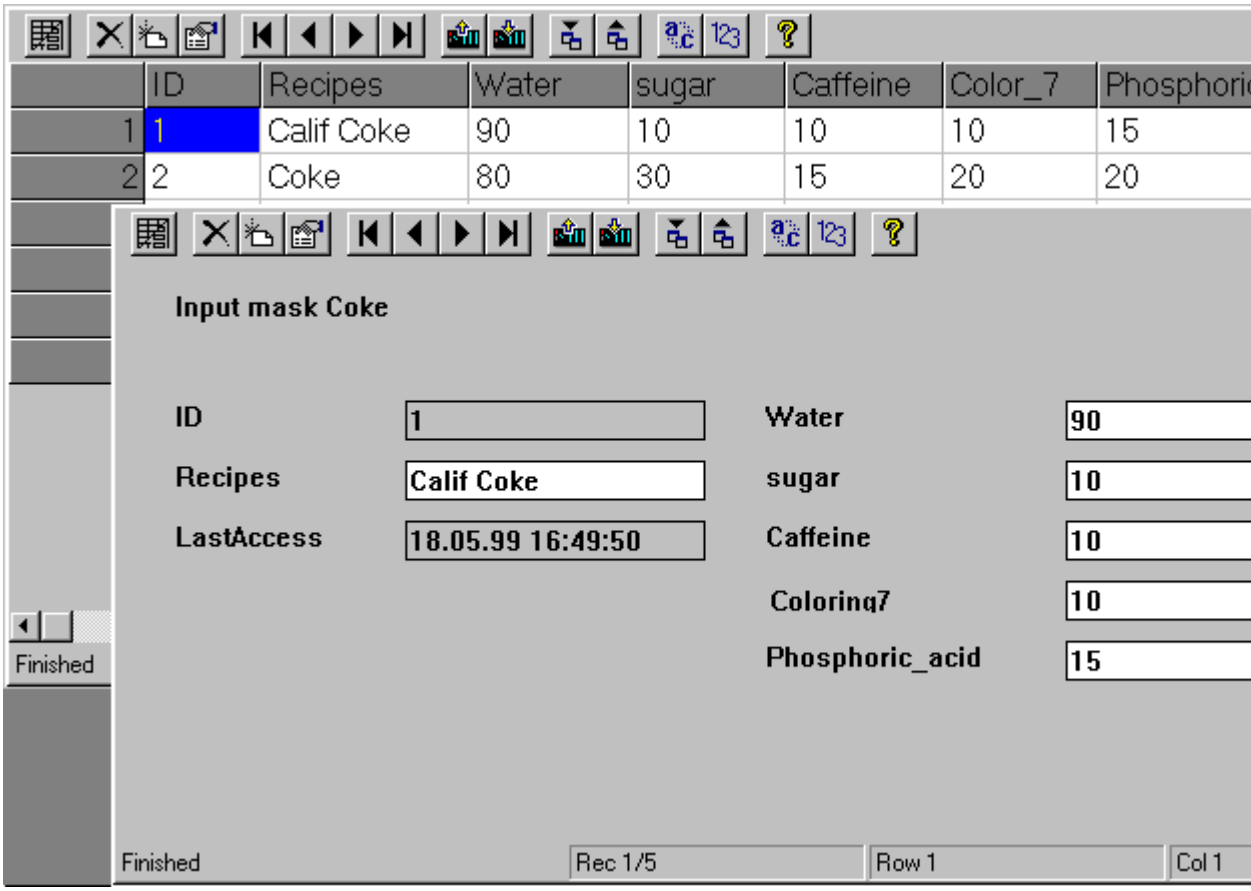
2.5.2 功能

功能范围

用户归档表格元素提供访问用户归档及其视图的选项。使用用户归档表格元素，在运行期间您可以：

- 创建、删除或修改新的数据记录
- 浏览用户归档
- 读写直接变量链接的变量

- 导入和导出用户归档
- 定义过滤和排序条件



视图

用户归档表格元素提供两个视图：表格视图和窗体视图。

- 表格视图
  表格视图用于以表格形式显示用户归档。每条记录占据一行，记录的数据域显示为列。
- 窗体视图
  窗体视图提供可由用户设计的用户界面。用户归档的窗体视图提供三种域类型：静态文本、输入域以及按钮。

说明

在组态过程中，用户归档表格元素连接到所选用户归档或视图，并且只能访问该用户归档或视图。要进行访问，必须启用用户归档/窗体（访问保护）。在用户管理器中，可以将特定授权分配给控件。

如果取消了该访问保护，则必须在图形编辑器中将控件重新连接到用户归档，以便控件能检测已取消的访问保护。

当打开用户归档表格元素的画面时，查询归档或域的访问保护。必须通过对象属性（例如画面、I/O 域或按钮的属性）来对受保护归档的控制变量单独执行访问保护。

参见

组态用户归档表格元素 (页 465)

2.5.3 组态用户归档表格元素

2.5.3.1 组态用户归档表格元素

步骤

要组态 WinCC 用户归档表格元素，请按下列步骤进行操作：

1. 使用用户归档编辑器或使用 WinCC 脚本语言函数组态用户归档。 在用户归档编辑器的说明中，可以了解用户归档“Cola”是如何组态的。
2. 在图形编辑器画面中放置新的用户归档表格元素。
3. 组态用户归档表格元素的属性。
4. 组态用户归档窗体视图。

参见

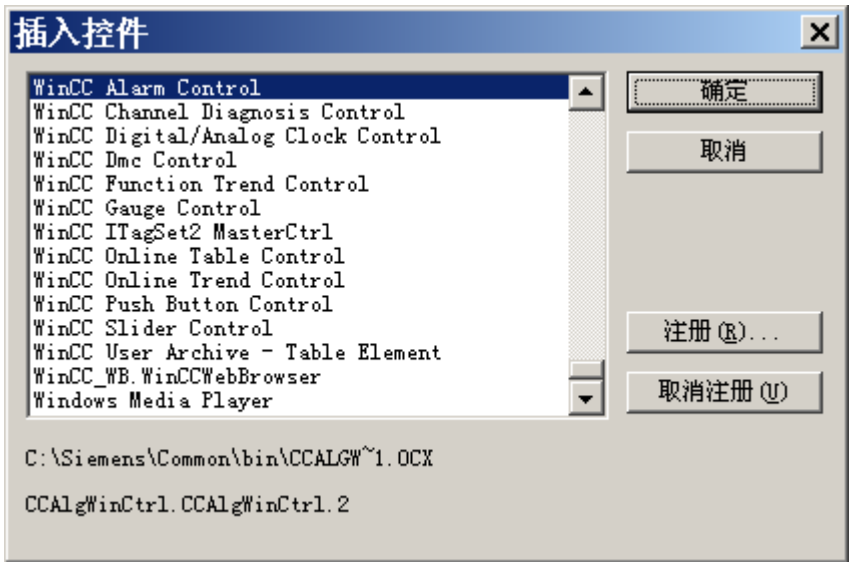
创建新用户归档 (页 321)

2.5.3.2 在过程画面中放置用户归档表格元素

步骤

为了在过程画面中建立用户归档表格元素，需要在图形编辑器中进行组态。 通过以下步骤可完成此操作：

1. 从对象选项板中选择对象组“智能对象”。
2. 单击“控件”对象，并将适当大小的窗口拖入图像区域。
3. 在当前已显示的“添加控件”选择对话框中，选择“WinCC 用户归档表格元素”选项，然后单击“确定”确认选择。



可选步骤

- 在对象选项板“控件”标签中的对象选项板窗口中，将显示某些标准控件以供选择。
- 选择 WinCC 用户归档表格元素。

参见

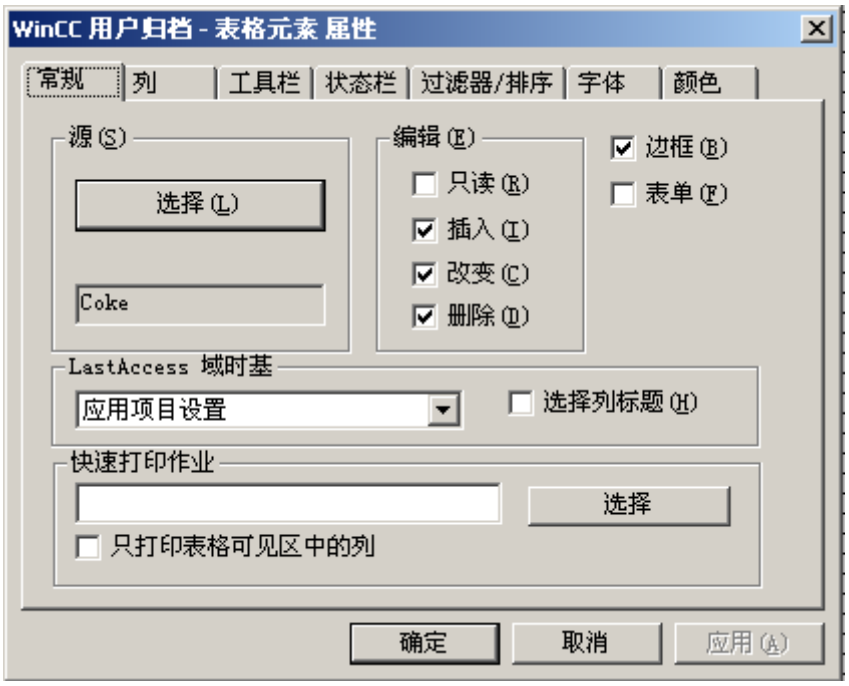
定义用户归档表格元素的属性 (页 467)

2.5.3.3 定义用户归档表格元素的属性

步骤

以下指南详细介绍了如何使用“WinCC 用户归档 - 表格元素属性”对话框在图形编辑器中组态用户归档“Cola”的用户归档表格元素。

1. 在“WinCC 用户归档表格元素”的区域内双击。 将显示“WinCC 用户归档 - 表格元素属性”对话框，其中包含“常规”标签。



2. 在源输入域中，定义要在控件中显示的归档或视图。 单击“选择”并在数据包浏览器对话框中选择用户归档“Cola”。
3. 在运行期间可以在编辑域中定义访问类型。 缺省状态下，启用访问类型“添加”、“修改”和“删除”。 也可以激活“只读”。
4. 使用“边框”复选框，可以定义显示的控件对话框是否带有框架。 激活这些选项。
5. 可以接受其它标签中的所有预设置，而不作任何改变。

参见

“常规”标签 (页 470)

删除用户归档表格元素 (页 468)

2.5.3.4 删除用户归档表格元素

步骤

在图形编辑器中删除用户归档表格元素分两步进行：

1. 单击以选择要删除的用户归档表格元素
2. 按下 Delete 键或选择“编辑 - 删除”菜单。

注意
随后立即执行删除操作，系统不会发出任何警告！ 只能使用“编辑 - 撤消”菜单或“Ctrl+Z”来撤消删除操作。

2.5.4 WinCC 用户归档表格元素的属性

2.5.4.1 WinCC 用户归档表格元素的属性

步骤

1. 通过右键单击对象并从打开的弹出式菜单中选择“属性”菜单项，可以修改用户归档表格元素属性。
   可以编辑过滤器、窗体、PressTBButton 和排序属性的静态。 为了避免数据库中的不一致，可以通过“WinCC 用户归档 - 表格元素属性”（双击控件）对话框更改其它对象属性。
2. 在打开的“对象属性”框的“属性”标签中，选择组“控件属性”。

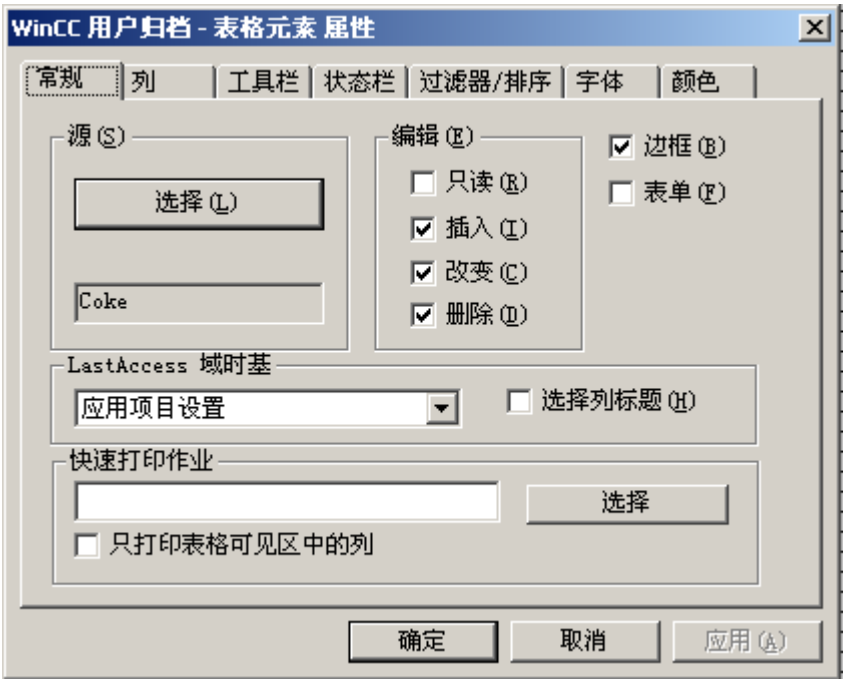
通常通过双击其中一个控件在图形编辑器中组态用户归档表格元素。 可以在打开的对话框中进行期望的更改。 由于各标签的对话框中提供了现有用户归档、视图、变量等以供选择，因此用户可以轻松安全地进行各种更改。

参见

组态用户归档表格元素 (页 465)

2.5.4.2 “常规”标签

组态



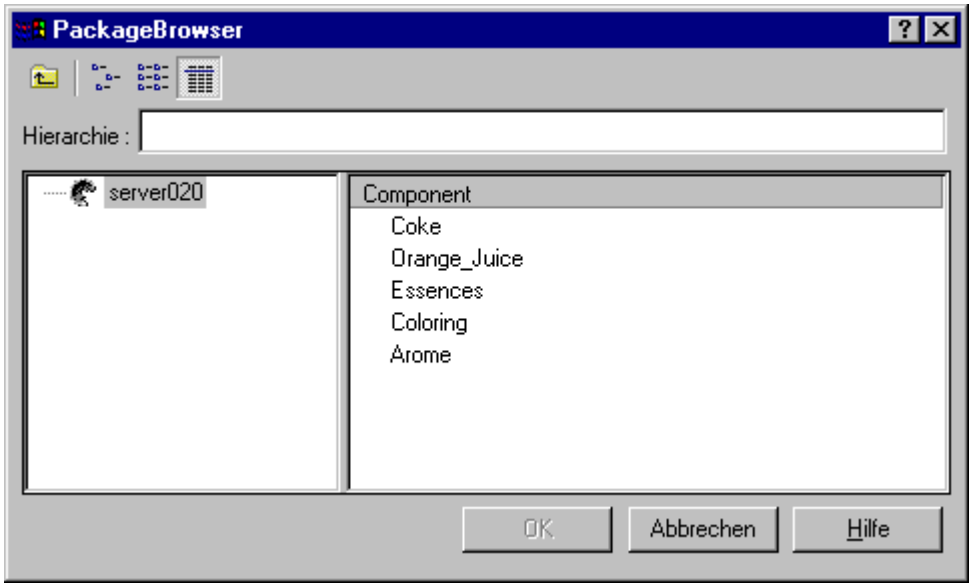
图标	描述
标题栏	在“标题栏”域中定义窗口标题。 在此定义是否显示标题栏，是关闭还是移动窗口。
源	单击“选择”按钮转到数据包浏览器，在此可以选择先前已组态的用户归档或视图。
编辑	在运行期间可以在编辑域中定义访问类型。 取消选中“只读”复选框后，系统会释放用户归档的访问类型“添加”、“修改”和“删除”。 对于视图，仅释放“修改”复选框。
边框	使用“边框”复选框，可以定义显示的控件对话框是否带有框架。
窗体	使用该复选框可定义窗体视图是否是控件窗口的开始视图。
“上次访问”域的时间基准	在此选择域中，可以在上次访问域中定义用于时间显示的时间基准。
快速打印作业	在此域中，定义将用于打印显示数据的打印作业。

说明

如果用户归档的组态在用户归档编辑器中已被修改（例如删除访问保护），则必须将图形编辑器中的控件重新链接到此用户归档。 然后，控件就可以探测到修改的归档组态。

数据包浏览器

通过单击用户归档表格元素属性对话框中的选择按钮可激活数据包浏览器。 可从已组态的用户归档和视图中进行选择。



在 WinCC 客户机中，可以从数据包浏览器的浏览域中选择某些服务器，此类服务器的数据包已加载并且其中已使用变量组态用户归档。 在 WinCC 客户机项目中，可以访问项目链接的所有服务器的用户归档。 某些用户归档不适用于 WinCC 客户机。 在“Hierarchy”（体系）区中将显示所选服务器的路径。 可以对其进行编辑，以便手动输入所期望服务器的路径。

如果要求的服务器未列在缺省列表中，则必须使用服务器数据功能来加载该服务器的数据包。关于 WinCC 客户机功能的其它信息，请参见 WinCC 项目管理器帮助。

说明

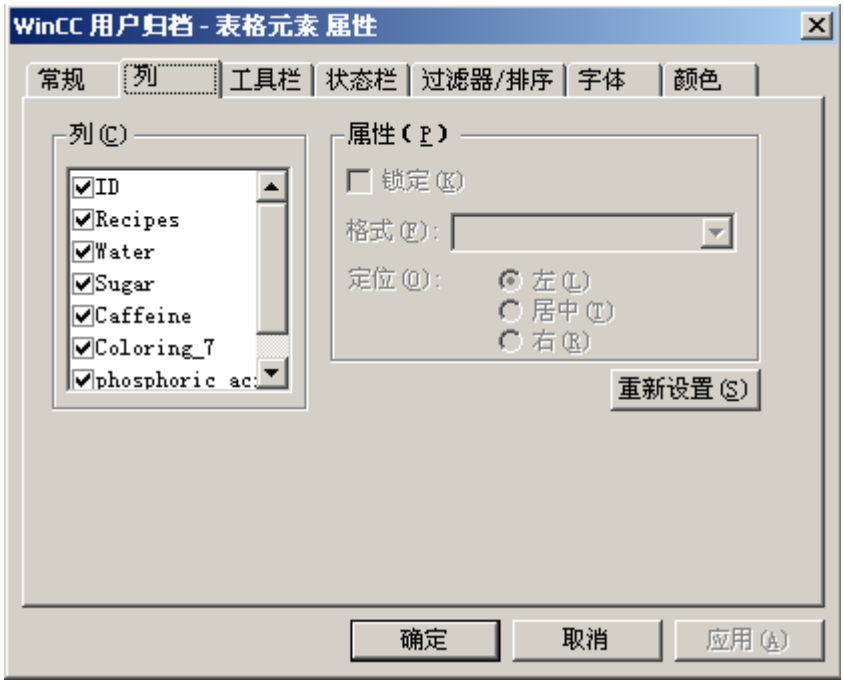
如果控件没有链接到现有的用户归档或视图上，当改变运行系统时，将显示出错消息“连接数据时出错！”。

参见

用户归档表格元素的属性列表 (页 494)

2.5.4.3 “列”标签

组态



图标	描述
列	在“列”输入域中，可以定义将在过程画面中显示的在用户归档编辑器中插入的域。
特征	在“属性”输入域中，可以定义当前在“列”域中所选域的属性。
锁定	可以使用锁定复选框防止选择域被覆盖。
格式	使用格式域来定义显示值的格式： • 固定 （固定小数位数“%.2f”） • 科学制 （指数显示“%e”） • 日期 （仅限于日期输出“%x”） • 时间 （仅限于时间输出“%X”） • 时间标志 （输出日期和时间“%c”） 日期域按操作系统中设置的日期格式来显示。

图标	描述
对齐	在“对齐”域中，可以在左、居中和右之间选择。
复位	使用复位按钮重新建立先前的设置。

说明

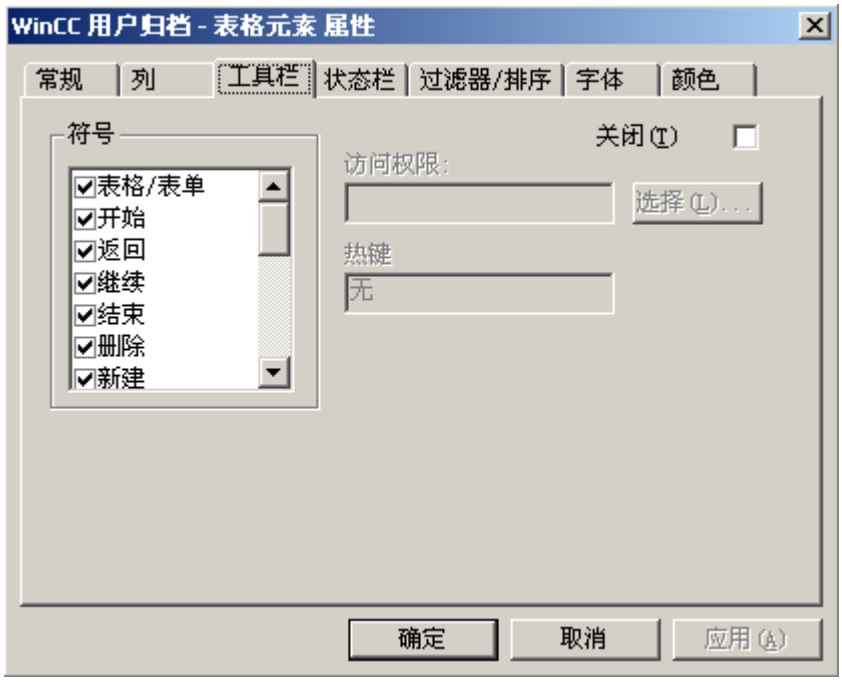
在格式域中，也可以设置整型值的十进制位数（例如，“%3f”表示三位十进制数值）或者十六进制格式“%x”。

参见

用户归档表格元素的属性列表 (页 494)

2.5.4.4 “工具栏”标签

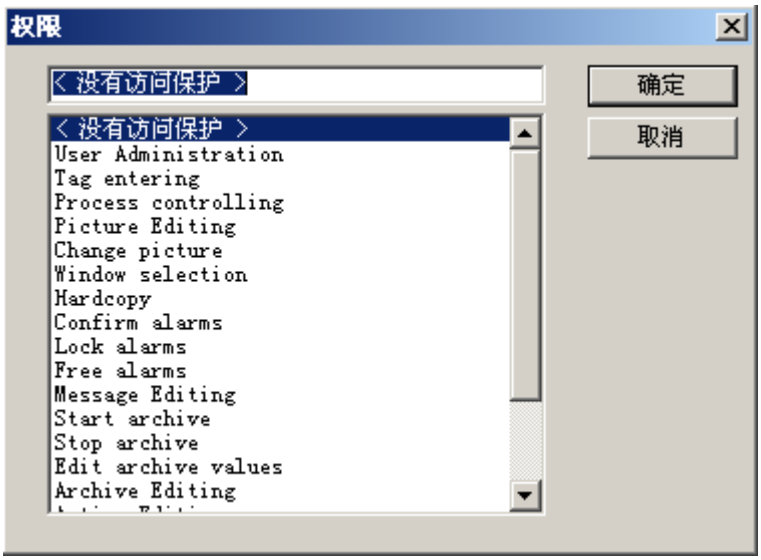
组态



图标	描述
符号	在“符号”下定义将包含在工具栏中的符号。
访问权限	在“访问权限”域中，将显示所选符号的访问权限。

图标	描述
选择	单击“选择”按钮以显示“权限”对话框，在此对话框中可定义期望的访问。
关闭	“关闭”域用于打开或关闭工具栏显示。
热键	在“热键”域中，可以为热键分配单个功能。

权限



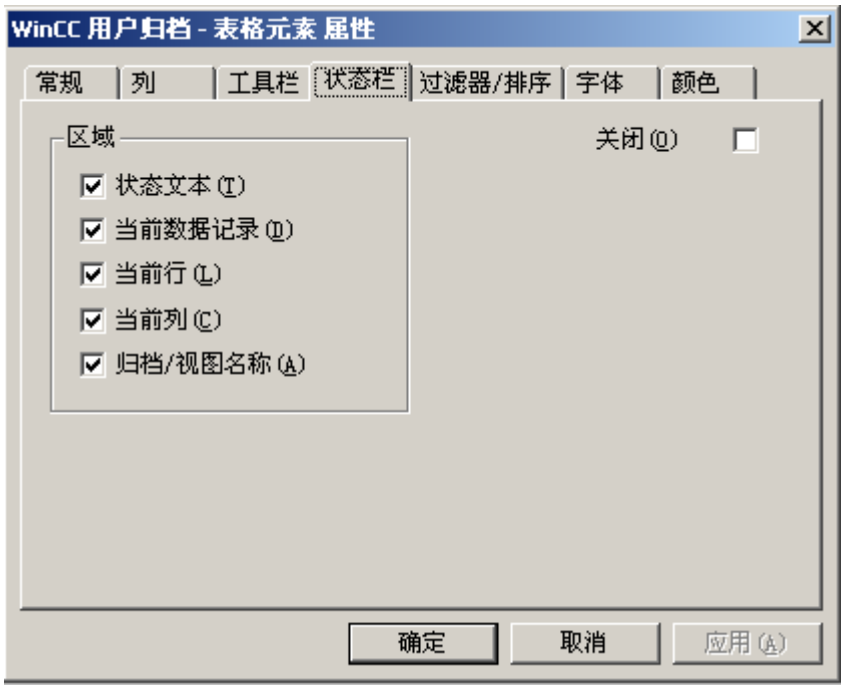
在“权限”对话框中，可定义期望的权限。 对话框中显示的权限是先前已在用户管理器中组态的权限。

参见

用户归档表格元素的属性列表 (页 494)

2.5.4.5 “状态栏”标签

组态



图标	描述
区域	在“范围”复选框中，可定义将包含在控件状态栏中的元素。
关闭	“关闭”域用于打开或关闭状态栏显示。

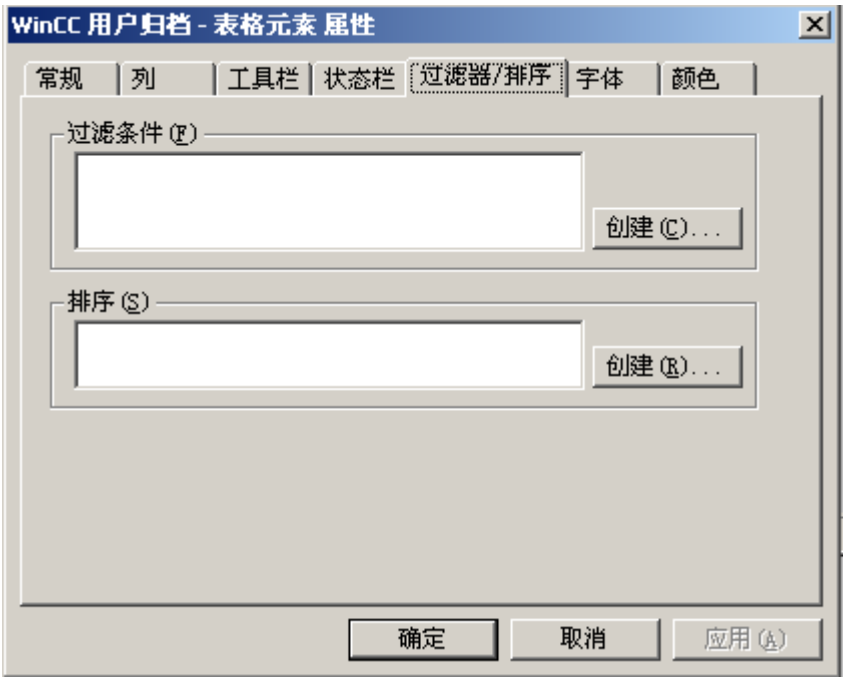
激活状态栏的所有区域后，状态栏显示如下：

参见

用户归档表格元素的属性列表 (页 494)

2.5.4.6 “过滤器/排序”标签

组态

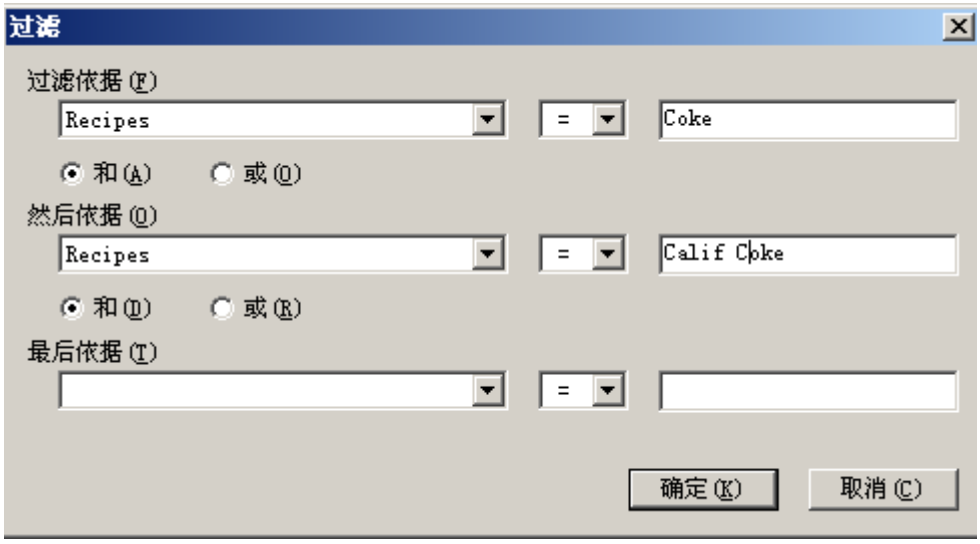


过滤标准

在“过滤标准”对话框中定义过滤条件。 直接输入过滤标准的规则。 这些条件可以使用数据库编程语言 SQL（结构化查询语言）编写。 关于包含许多实用实例的 SQL 描述，请参见附录。

实例： FieldC > 100
选择所有数据集，其在“FieldC”列中包含的值大于 100。

单击“创建...”按钮后，将显示自动过滤画面，在此画面中可定义过滤标准。

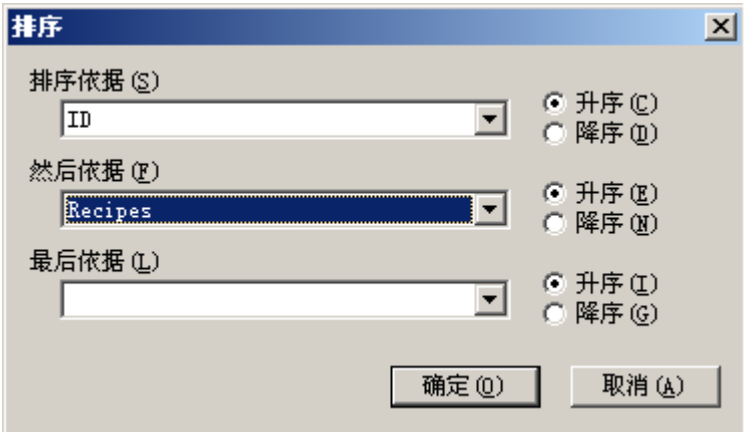


在“过滤依据”行，可定义过滤标准，在左侧的选择域中将显示用户归档的各数据域。 使用“然后依据”和“最后依据”行来定义其次的过滤标准。 按此顺序处理过滤。

排序

在“排序...”对话框中定义排序条件。 使用数据库编程语言 SQL 直接输入排序规则。

单击“创建...”按钮后，将显示自动过滤画面，在此画面中可定义排序条件。



在“排序依据”选择域中，可定义排序条件，并提供了用于用户归档的各数据域以供选择。 使用“然后依据”和“最后依据”选择域来定义其次的排序条件。 按此顺序处理过滤。 如果单击“升序”，将按升序排序；如果单击“降序”，将按降序排序。

参见

用户归档表格元素的属性列表 (页 494)

SQL 语言 (页 366)

2.5.4.7 “字体”标签

组态

在“字体”标签中，可以定义将在控件中使用的字体。

参见

用户归档表格元素的属性列表 (页 494)

2.5.4.8 “颜色”标签

组态

在“颜色”标签中，可以定义将在控件中使用的颜色。

参见

用户归档表格元素的属性列表 (页 494)

2.5.5 组态窗体视图

2.5.5.1 组态窗体视图

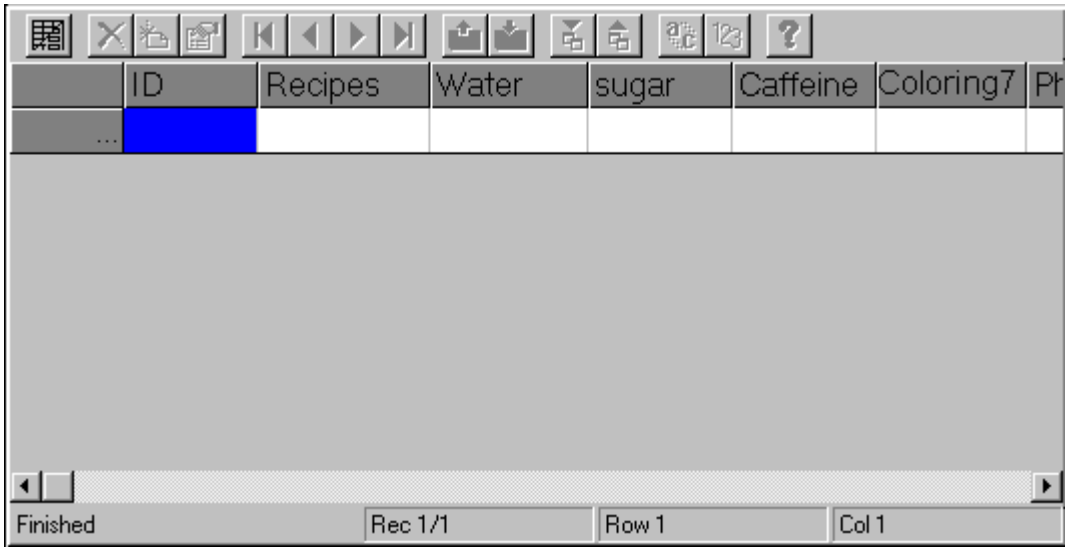
步骤

用户归档表格元素的窗体可由用户在图形编辑器中手动组态，并可用于在运行系统中编辑和显示用户归档数据。

创建窗体视图需要先组态用户归档表格元素。

下列指南说明如何在图形编辑器中组态新窗体视图。

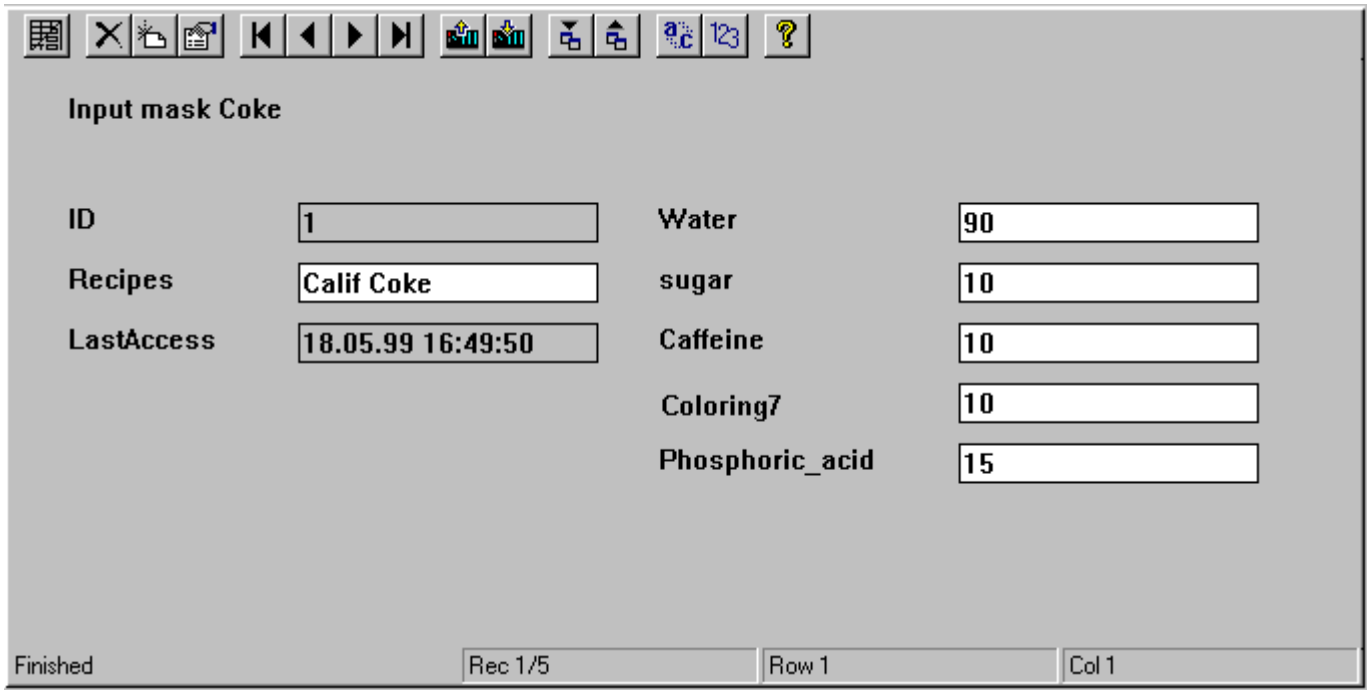
1. 按住 Control 键的同时，双击用户归档表格元素。将显示控件的表格视图。现在可以定义运行系统中各个列的宽度。



2. 使用  图标在窗体和表格视图之间进行切换。单击该图标以转到窗体视图。

现在可以开始组态窗体了。

现在我们创建一个窗体：



说明

在空窗体中单击右键，可以使用弹出式菜单中的“创建，全部”功能，以自动生成用户归档中所有现有数据域的窗体域。 对于各数据域，还会为其插入具有相应别名的文本域。 “创建，选定项”选项用于仅为在“列”选项卡中选定的列生成窗体域。

说明

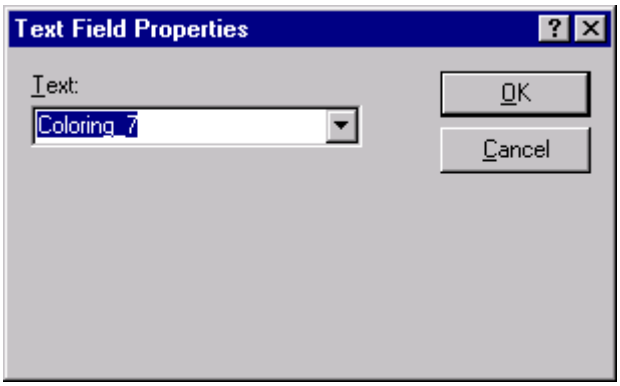
用户归档表格元素不支持缩放功能。 组态缩放功能将导致运行系统中出现显示问题。

2.5.5.2 插入“文本”窗体域

步骤

1. 如果尚未执行此操作，请打开窗体视图。
2. 要插入新的“文本”窗体域，可在图形编辑器的用户归档表格元素工作区中，右键单击希望放置文本的地方。 将显示以下列表框：

3. 选择“Add Text Field”（添加文本域）后，将转至“Text Field Properties”（文本域属性）对话框。



4. 可以在“Text”（文本）域中输入期望的文本。 在此输入文本“输入窗体 Cola”作为窗体的标题。

说明

如果通过“文本”扩展选择域，则将以静态文本形式显示归档的所有域名。 如果在文本库的文本引用中已输入用于语言切换的文本引用，则会提供相同的内容以供选择。

参见

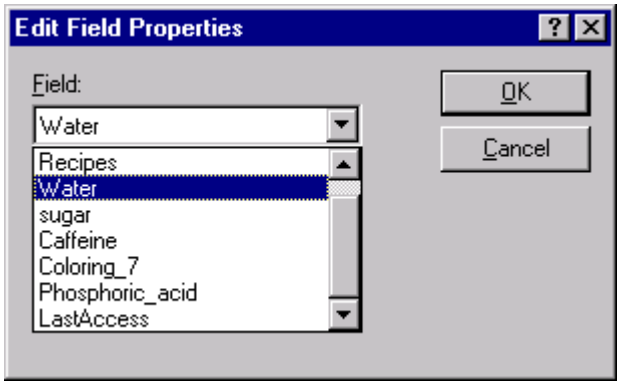
插入“编辑”窗体域 (页 482)

2.5.5.3 插入“编辑”窗体域

步骤

1. 如果尚未执行此操作，请打开窗体视图。
2. 要插入新的“编辑”窗体域，可在图形编辑器的用户归档表格元素工作区中，右键单击希望放置编辑域的地方。 将显示以下列表框：

3. 选择“Add Edit Field”（添加编辑域）后，将转至“Edit Field Properties”（编辑域属性）对话框：



在选择域对话框中，可以从用户归档的所有已组态域中进行选择。

4. 选择“Water”（水）。 现在可插入其它编辑域，例如“Sugar”（糖）、“Dyestuff 7”（色素 7）、“Caffeine”（咖啡因）和“Phosphoric Acid”（磷酸）。

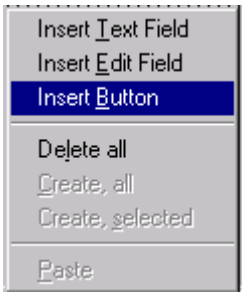
参见

插入“按钮”窗体域 (页 483)

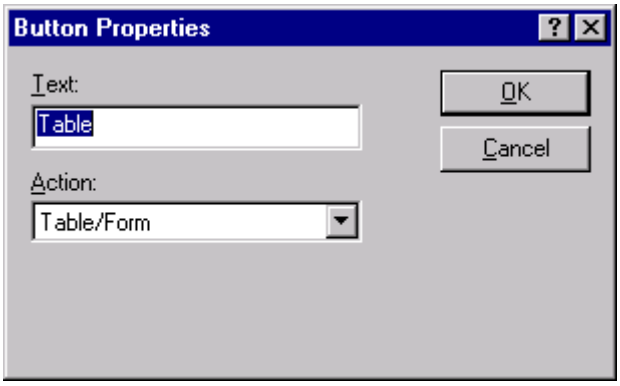
2.5.5.4 插入“按钮”窗体域

步骤

1. 如果尚未执行此操作，请打开窗体视图。
2. 要插入新“按钮”，可在图形编辑器的用户归档表格元素工作区中，右键单击希望放置按钮的地方。 将显示以下列表框：



3. 选择“Add Button”（添加按钮）后，将转至“Button Properties”（按钮属性）对话框：



4. 在文本域中，可以定义要在新按钮上显示为标签的文本。 输入文本“表格视图”。
5. 在“Action”（动作）域中，可以选择窗体视图的图标。 新组态的按钮将执行与工具栏中相关图标相同的操作。 选择“Form”（窗体）以切换到表格视图。

说明

从窗体视图到按钮均可链接工具栏功能。 也可设计按钮的大小和布局以通过触摸屏操作工具栏的某些功能。

参见

随后编辑窗体域 (页 484)

2.5.5.5 随后编辑窗体域

步骤

1. 随后，为了修改窗体域，可右键单击预组态的窗体域，然后单击“属性”按钮。
   或
   双击预组态的窗体域。

将显示相应的对话框以用于修改窗体域（如同文本、编辑和按钮窗体域章节中所述）。

参见

删除窗体域 (页 484)

2.5.5.6 删除窗体域

步骤

1. 为了删除窗体域，请右键单击预组态的窗体域。
2. 单击菜单项“删除”。

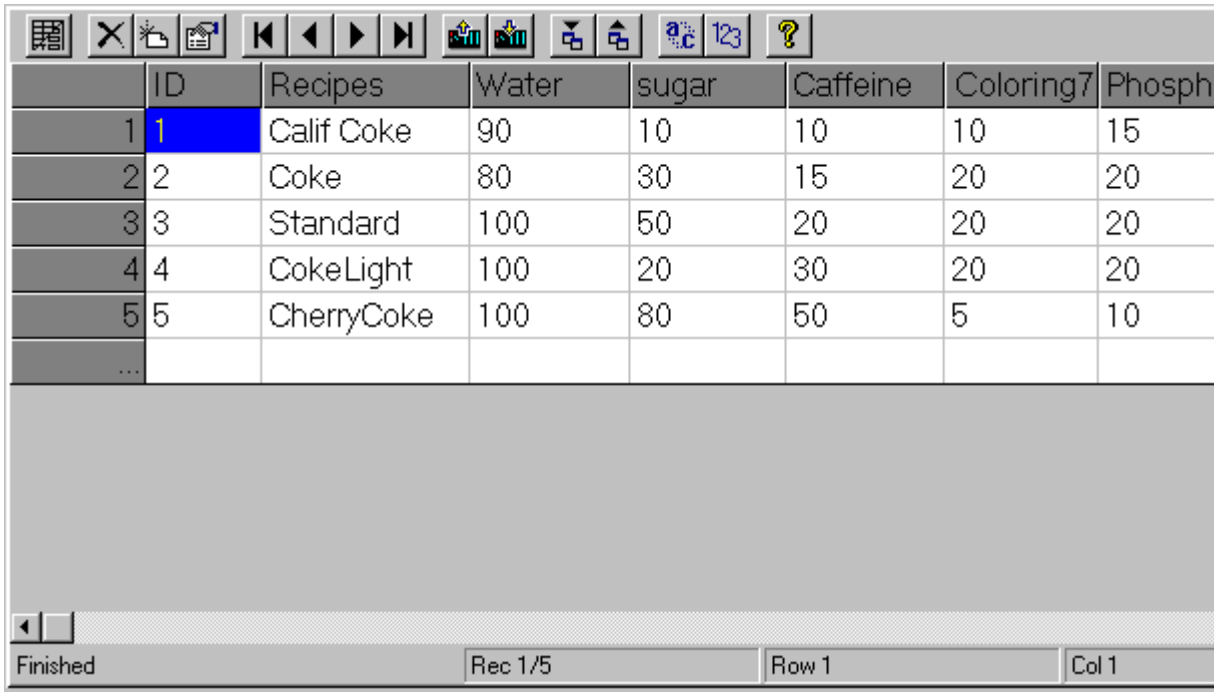
窗体域即删除。 请勿使用 Delete 键，因为这将删除整个控件。

2.5.6 运行系统中的用户归档表格元素

2.5.6.1 用户归档表格元素表格

应用

在运行期间，用户归档表格元素表格将以表格形式显示和输入用户归档数据。 通过按下组合键 <CTRL+ ENTER> 可访问一个单元格中的多行文本。 一个单元格中的多行文本将在表格视图中显示为一行，所有的行将在一个单行中总结。



使用此工具栏操作控件表格和窗体窗口：

在表格中处理的方式与在用户归档编辑器中处理表格窗口的方式相同。

说明

如果在控件表格中有一个或多个数值发生了改变，则必须退出数据记录（即转换到另一表格单元或行），以使该数值被数据库接受并在其它显示中更新。

WinCC 脚本语言中的操作必须确保记录的选择。 无法使用控件来选择记录。

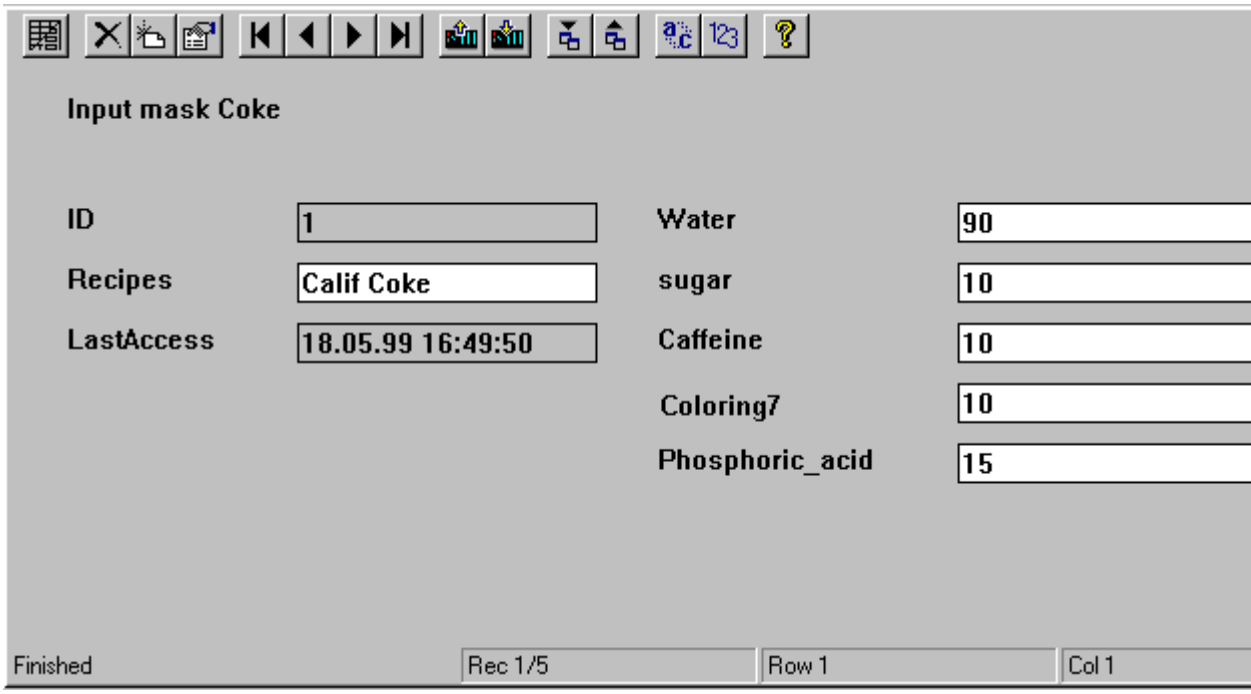
参见

组态用户归档表格元素 (页 465)

2.5.6.2 用户归档表格元素窗体

应用

用户归档表格元素的窗体可由用户在图形编辑器中手动组态，并可用于在运行系统中显示和输入用户归档数据。 文本可以在多个行中访问和显示。



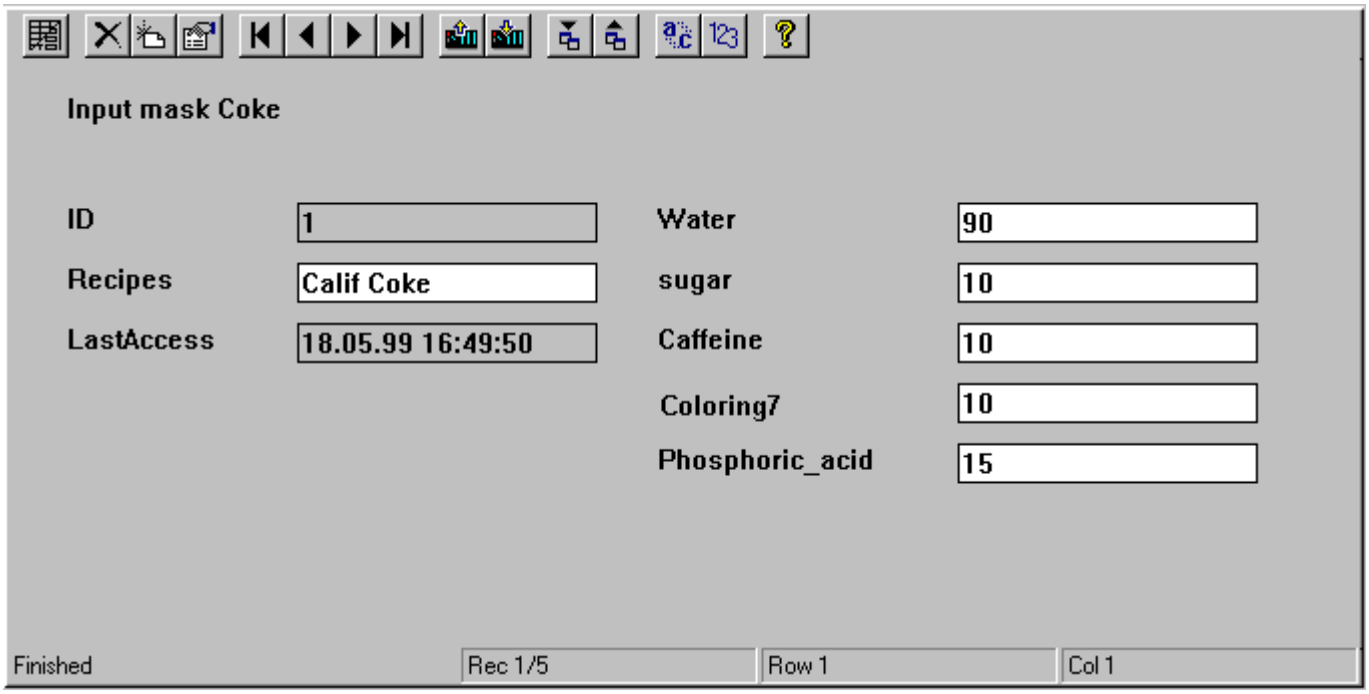
说明

如果在用户归档表格元素窗体中更改了一个或多个数值，那么在完成数据条目后必须滚动到另一记录，以使该数值被数据库接受并在其它显示中更新。

2.5.6.3 WinCC 用户归档表格元素工具栏

功能

工具栏提供下列选项：



图标	描述
	切换
	删除记录
	插入新记录
	修改现有域
	在表格窗口中浏览
	读写变量
	导入和导出用户归档
	定义过滤标准
	定义排序条件

图标	描述
	“上次访问”域的时间基准
	打印
	请求帮助

切换

使用此图标在窗体和表格视图之间进行切换。

删除记录

删除所选记录。

插入新记录

逐个输入数据域的数值，每次按下 Enter 键进行确认。 输入所有数据域后，包含插入数值的新纪录即创建成功。

修改现有域

单击此图标后，单击想要修改的域。 将显示文本标记 - 现在即可看见可以编辑此域的信息。 激活“修改现有域”后，即可在修改模式下修改用户归档表格元素。 然后将光标移入表格，并立即进行修改。 如果“修改”模式已关闭，则只能通过按下 F2 热键或双击要修改的域进行更改。

在表格窗口中浏览

可以使用这些按钮在表格窗口中向后向前滚动或浏览，并可跳至用户归档的起始或结束位置。

读写变量

这些按钮用于读写 WinCC 变量。

在“归档属性”对话框的“通讯”标签中设置用户归档时，可以选择通讯类型“通过 WinCC 变量进行通讯”。

导入和导出归档

单击这些按钮之一后，将以 CSV 格式导入或导出用户归档（用逗号分隔数值）。

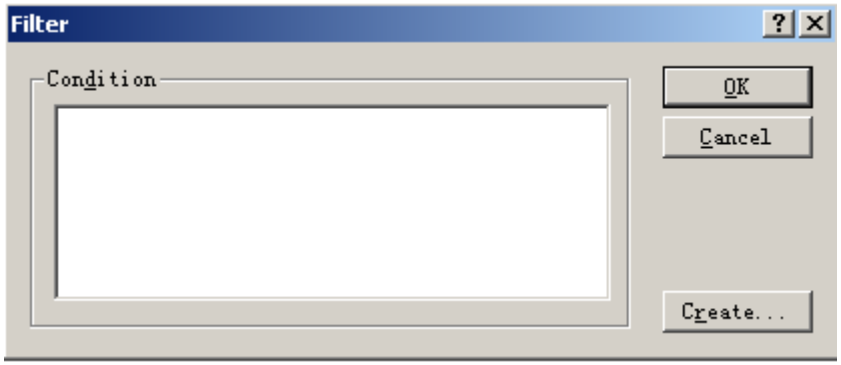
注意
在 Excel 中读取它们前，需要将数据类型指定为 CSV，否则 Excel 无法正确读取从 WinCC 导出的 CSV 文件。

说明

多用户项目还必须考虑以下内容： 如果服务器上有用户归档（例如在“c:\Projects\Test\UA”下），通过此指定路径则可启用它。 客户机通过网络驱动器（例如“I:\Test\UA”）来映射该激活。 此后，在客户机上用户归档的标准路径为“I:\Test\UA”。 但是，在服务器上符合此描述的目录不存在。 如果想要导入/导出用户归档数据，则必须改变客户机上的标准路径，在本实例中改为“C:\Projects\Test\UA”。

定义过滤标准

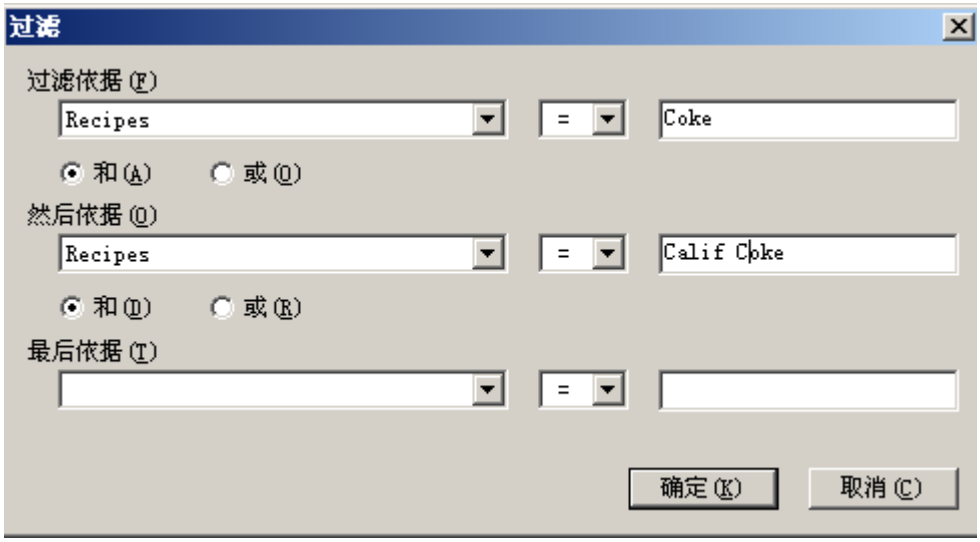
使用该选项来输入过滤标准。 将导出所有显示的数据。 要导出子集，则需要以仅显示期望的数据的形式表达过滤标准。 然后导出已过滤的数据。



这些条件可以使用数据库编程语言 SQL（结构化查询语言）编写。 关于包含许多实用实例的 SQL 描述，请参见附录。 有关详细信息，请参阅相关技术文献。

实例： ID < 100
仅选择 ID 为 1 到 99 的数据域；不显示其它所有的数据域。

单击“创建...”按钮后，将显示自动过滤画面，在此画面中可定义过滤标准。



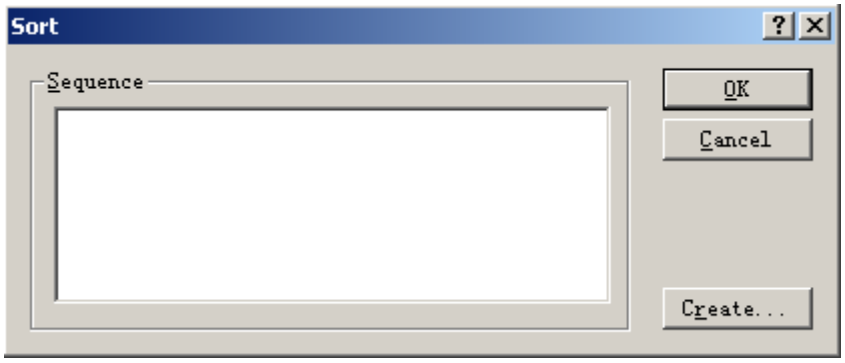
在“过滤依据”行，可定义过滤标准，在左侧的选择域中将显示用户归档的各数据域。 使用“然后依据”和“最后依据”行来定义其次的过滤条件。 按此顺序处理过滤。

说明

此处定义的过滤条件是暂时的，即在构建新画面后，属性对话框中定义的过滤条件将再次有效。

定义排序条件

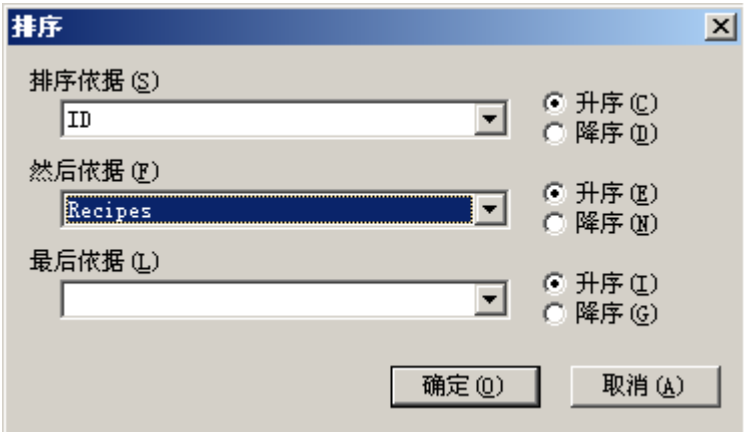
使用该选项来输入排序条件。



使用数据库编程语言 SQL 直接指定排序规则。

也可参考附录中 SQL 的说明。 有关详细信息，请参阅相关技术文献。

单击“创建...”按钮后，将显示自动过滤画面，在此画面中可定义排序条件。



在“排序依据”选择域中，可定义排序条件，并提供了用于用户归档的各数据域以供选择。使用“然后依据”和“最后依据”选择域来定义其次的排序条件。 按此顺序处理过滤。 如果单击“升序”，将按升序排序；如果单击“降序”，将按降序排序。

说明

此处定义的排序条件是暂时的，即在构建新画面后，属性对话框中定义的过滤标准将再次有效。

“上次访问”域的时间基准

使用该选项，可以改变“上次访问”域的时间基准。

打印

此选项开始打印显示的数值。

请求帮助

单击此帮助按钮以请求关于用户归档表格元素的帮助。

2.5.6.4 使用动态对象操作控件

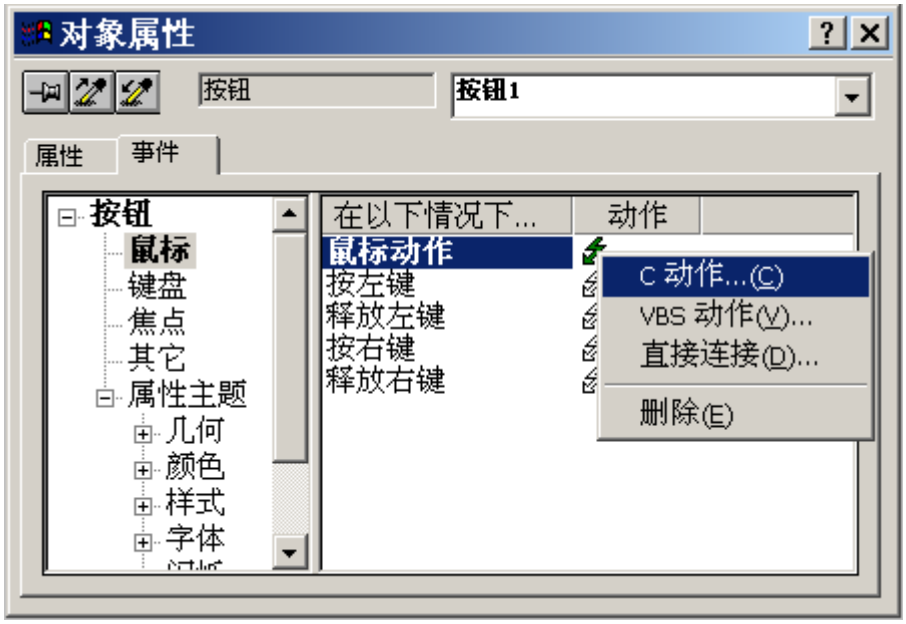
操作选项

使用用户归档表格元素提供的选项可将所有工具栏按钮的功能转移到自定义按钮或 I/O 域。可以定义每个按钮的大小和外观，从而可以操作表格元素，例如通过触摸屏操作。

使用“按下 TB 按钮”属性的实例

要将用户归档表格元素与按钮相关联，需要执行以下步骤：

1. 在图形编辑器中创建一个按钮，然后通过右键单击对象属性来调用它。
2. 在“事件”选项卡中，选择“鼠标”选项。 在右窗口中单击，选择“鼠标单击”。右键单击“动作”列中的箭头后，将显示一个对话框；选择“直接连接”选项。



3. 在“源”区域中，选择“常量”并在此输入常量，例如“Form”（请参见下面有关用户归档表格元素可用常量的概述）。
4. 在“目标”区域中，单击“画面中的对象”选项并在“对象选择”域中选择要链接的表格元素。在“属性”框中，选择“PressTBButton”并单击“确定”确认对话框。
5. 在图形编辑器中保存画面并转到运行系统。 如果此时激活已组态的“窗体”按钮，控件显示将从表格视图切换到窗体视图，反之亦然。

用于直接链接用户归档表格元素的常量

对于上述用户归档表格元素的直接链接，控件的每个按钮都有一个可用常量。可以根据下表分配单个按钮。

常量	相应按钮
Form	
Delete	
New	
Edit	
First	

常量	相应按钮
Previous	
Next	
Load	
WriteVar	
ReadVar	
Import	
Export	
Filter	
Sort	
Timezone	
Print	
Help	

说明

当使用键盘操作表格窗口时，按下“Tab”和“Position 1”键后，所选记录所在单元格的光标将不再可见。要使显示返回上次编辑的记录，可根据上述步骤插入按钮并选择“VTB_Focus”常量。按下此按钮后，单元格光标会跳回上次编辑的位置。

参见

用户归档表格元素的属性列表 (页 494)

2.5.6.5 用户归档表格元素的属性列表

概述

可以为用户归档表格控件设置以下属性：

属性	描述	可以动态显示
BackColor	定义用户归档表格元素中表格窗口的背景颜色。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“颜色”选项卡中可以编辑这些设置。	否
Border	确定运行系统中显示的用户归档表格元素的窗体视图是否带有边框。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“常规”选项卡中可以编辑这些设置。	否
Buttons	指定将输出工具栏中激活的按钮所对应的由软件生成的指针。 为了避免数据库中的不一致，禁止编辑该属性的静态设置。	否
Caption	定义用户归档表格元素中标题栏的标签。	否
Closable	确定是否可通过标题栏中的“X”关闭用户归档表格元素。	否
Delete	确定在运行系统中是否允许在用户归档表格元素中执行删除过程。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。“常规”选项卡的“编辑”区域中提供了一个针对此属性的复选框。	否
Filter	定义数据库的过滤条件。这些条件可以使用数据库编程语言 SQL（结构化查询语言）编写。 实例：FieldC > 100 选择在“FieldC”列中的值大于 100 的所有数据集。 也可在“过滤器/排序”选项卡中输入这些过滤条件。	是，使用名称 Filter
Form	定义在运行系统中启动时用户归档表格元素的视图： 状态“是”：输出窗体视图 状态“否”：输出表格视图。 也可以在“常规”选项卡中更改这些设置	是，使用名称 Form
GridBackColor	定义用户归档表格元素中数据集的背景颜色。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“颜色”选项卡中可以编辑这些设置。	否

属性	描述	可以动态显示
GridFont	定义用户归档表格元素中的字体。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“字体”选项卡中可以编辑这些设置。	否
GridForeColor	定义用户归档表格元素中数据集的字体颜色。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“颜色”选项卡中可以编辑这些设置。	否
HeaderBackColor	定义用户归档表格元素中标题和带有连续行号的列的背景颜色。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“颜色”选项卡中可以编辑这些设置。	否
HeaderForeColor	定义用户归档表格元素中标题和带有连续行号的列的字体颜色。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“颜色”选项卡中可以编辑这些设置。	否
Insert	定义是否可在用户归档表格元素中输入条目。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。“常规”选项卡的“编辑”区域中提供了一个针对此属性的复选框。	否
LocaleSpecificSettings	定义文本和字体的特定语言的响应，可在属性对话框中进行组态。 值 =“Yes”：可为各运行系统语言分配单独的文本和字体。 要执行此操作，可以在图形编辑器的“视图/语言”菜单中选择一种语言并在控件中选择期望的字体。 值 =“No”：无法定义特定语言的文本和字体。 控件的组态始终适用于所有可用的运行系统语言。	是，使用名称 LocaleSpecificSettings
Movable	定义是否可以移动用户归档表格元素。	否
Name	定义将显示的用户归档或视图。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“常规”选项卡的“源”区域中，将显示所有已组态用户归档和视图的选项。	否
PressTBButton	将表格元素工具栏的所有按钮与自定义按钮或 I/O 域连接起来。	是，使用名称“按下 TB 按钮”
PrintJob	指定哪种布局应用于打印输出。	否
PrintVisColsOnly	定义是否在草图打印模式下只打印当前可见列。	否

属性	描述	可以动态显示
Read only	定义用户归档表格元素在运行系统中是可以编辑还是为只读。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。“常规”选项卡的编辑区域中提供了一个针对此属性的复选框。	否
SelBackColor	定义用户归档表格元素中所选数据集的背景颜色。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“颜色”选项卡中可以编辑这些设置。	否
SelForeColor	在用户归档表格元素中定义所选数据集的字体颜色。 为了避免数据库中的不一致，该属性的静态只能在属性对话框中编辑。在“颜色”选项卡中可以编辑这些设置。	否
SelectedID	显示控件窗口中所选数据集的 ID。 SelectedID =“0”： 如果未选择有效数据集（如连接出错时） 如果选择了编辑行，SelectedID =“-1”。	否
Sort	定义数据库的排序条件。 这些条件可以使用数据库编程语言 SQL（结构化查询语言）编写。 也可以在“过滤器/排序”选项卡中输入过滤条件。	是，使用名称 Sort
StatusbarDisabled	定义是否在运行系统中激活用户归档表格元素的状态栏。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。 “状态栏”选项卡的区域中有一个针对该属性的复选框“关闭”。	否
StatusbarShowArc	定义是否在用户归档表格元素的状态栏中显示归档名称。	是，使用名称 StatusbarShowArc
StatusbarShowCol	定义是否在用户归档表格元素状态栏中显示当前所选数据集列的连续编号。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。 可以在“状态栏”选项卡的“当前列”中更改此设置。	否
StatusbarShowRecord	定义是否在用户归档表格元素状态栏中显示当前所选数据集列的域坐标。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。 可以在“状态栏”选项卡的“当前数据记录”中更改此设置。	否

属性	描述	可以动态显示
StatusbarShowRow	定义是否在用户归档表格元素状态栏中显示当前所选数据记录行的连续编号。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。 可以在“状态栏”选项卡的“当前行”中更改此设置。	否
StatusbarShowText	定义是否在用户归档表格元素的状态栏中显示数据库的当前状态。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。 可以在“状态栏”选项卡的“状态文本”中更改此设置。	否
Titleline	定义是否在用户归档表格元素中显示标题栏。	否
TimeZone	确定在运行系统中用于显示时间的时间基准。 通过以下数值设置时间基准： 值 = 0：应用项目设置 值 = 1： 服务器时区 值 = 2：本地时区 值 = 3：协调世界时 (UTC) 建议使用默认组态“应用项目设置”。这表示采用与项目其余部分相同的时区进行显示。	否
TimeZoneMark	确定 LastAccess 域的列标题是否应包含所设置的时区。 将以下首字母缩写词用于时区： LOC：本地时区 UTC：协调世界时 SVR：服务器时区	否
ToolbarDisabled	定义是否在运行系统中激活用户归档表格元素中的工具栏。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。 “工具栏”选项卡的区域中有一个针对该属性的复选框“关闭”。	否

属性	描述	可以动态显示
Type	定义在用户归档表格元素中是显示用户归档，还是显示视图 数值类型 = 0：代表用户归档 数值类型 = 1：代表视图。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。在“常规”选项卡的“源”区域中，将显示所有已组态用户归档和视图的选项。	否
Update	定义是否可在用户归档表格元素中进行更改。 为了避免数据库中的不一致，只能在属性对话框中编辑该属性的静态设置。“常规”选项卡的“编辑”区域中提供了一个针对此属性的复选框。	否

2.5.6.6 布局中的动态属性概述

过滤

可以使用“过滤”属性来定义数据库的过滤条件。 必须使用数据库编程语言 SQL 来表达条件。
格式： SQL 文本
变量类型： 文本变量

排序

可以使用“排序”属性来定义数据库的排序条件。 必须使用数据库编程语言 SQL 来表达条件。
格式： SQL 文本
变量类型： 文本变量

时区

确定运行系统中用于显示时间的时间基准。
格式： 数字

数值	描述
0	本地时区
1	服务器时区
2	协调世界时（UTC）
3	应用项目设置

变量类型： 所有变量类型，除二进制、文本和原始数据变量以外

布局编辑器中的 COM 服务器 3

3 资源

3.1 布局编辑器中的 COM 服务器

内容

在 WinCC 报表系统中，可以使用 COM 接口集成用户指定的报表对象。 因此，不在 WinCC 中产生的数据可以输出到 WinCC 报表中。 文档提供以下方面的信息：

- COM 服务器的应用程序选件
- 将 COM 服务器集成到 WinCC 中
- 使用 COM 对象输出数据
- 用于报表的 COM 接口的详细资料

3.2 使用 COM 服务器对象

引言

为了在 WinCC 日志中集成用户指定的数据，可将 COM 服务器集成在报表系统中。 此 COM 服务器提供可以在页面布局编辑器的对象选项板中选择的 COM 对象，该对象可添加到页面布局。 COM 对象便可提供用于日志输出的用户指定数据。 COM 对象可以具有文本、表格或画面类型。

此处，不能对 COM 对象本身进行任何注释。有关信息由 COM 服务器的作者提供。

COM 服务器对象的集成

在报表系统中使用 COM 对象时，必须执行下列步骤：

1. 注册 COM 对象
2. 如有必要，根据 COM 对象运行 COM 对象的注册表文件。
3. 通过将 COM 对象导入注册表来将其插入页面布局编辑器的对象选项板中

4. 在对象选择中选择 COM 对象
5. 在布局中定位 COM 对象并将其参数化

在每台服务器和想要在其上使用该 COM 对象的 WinCC 客户机上，都必须执行步骤 1 至 3。如果 COM 对象只应用于运行系统中，则也必须执行这些步骤。

要为 WinCC 报表系统创建 COM 服务器，必须在开发计算机上注册类型库 “WinCCProtProvider.tlb”。类型库自动在安装了 WinCC 的计算机上注册。要在没有安装 WinCC 的计算机上注册，可如下操作：

从安装了 WinCC 的计算机复制“WinCCProtProvider.tlb”文件。 该文件位于目录“..\Siemens \WinCC\Interfaces”。 将文件添加到目标计算机并进行注册。

正确集成的 COM 对象显示在页面布局编辑器对象选项板的“COM 服务器”标签中。可以从该处进行选择并添加到布局。

有关详细信息，请参阅章节“用于报表的 COM 接口的详细资料”。

参见

如何在报表中输出 COM 服务器的数据 (页 502)

COM 服务器集成实例 (页 503)

用于报表的 COM 接口的详细资料 (页 504)

3.3 如何在报表中输出 COM 服务器的数据

引言

为了在 WinCC 日志中集成用户指定的数据，可将 COM 服务器集成在报表系统中。 该 COM 服务器使日志对象可在对象选项板中使用，可在页面布局编辑器中选择并插入到页面布局中。 COM 对象便可提供用于日志输出的用户指定数据。

可用的日志对象

用户定义的 COM 对象	用于输出来自用户数据源的数据到 WinCC 日志中。

要求

- 知道如何创建布局和插入日志对象

步骤

1. 创建新的页面布局，并在页面布局编辑器中将其打开。
2. 在 COM 服务器标签的对象选项板中，选择由用户集成的 COM 对象，并在工作区中将其拖动为所需的大小。
3. COM 对象的创建者将对数据的连接和选择进行详细说明。
4. 根据此处的说明来组态 COM 对象。
5. 保存布局。
6. 创建打印作业，并选择此处所组态的页面布局。
7. 例如，通过 WinCC 项目管理器中的打印作业或通过在 WinCC 画面中所组态的调用来启动输出。

输出选项

无论谁编写 COM 对象，都将接收到可能的输出选项的信息。

参见

使用 COM 服务器对象 (页 501)

用于报表的 COM 接口的详细资料 (页 504)

COM 服务器集成实例 (页 503)

3.4 COM 服务器集成实例

引言

WinCC 光盘中提供了两个实例，每个实例都包含 COM 服务器。 一个实例用 Visual Basic 编写，另一个实例用 Visual C 编写。实例为 zip 格式，位于 WinCC 光盘的“Options\ODK\Samples”下。“CCProtTableServerExampleVB.zip”文件包含 Visual Basic 编写的实例。“CCProtPicturerExampleCPP.zip”文件包含 Visual C++ 编写的实例。

步骤

将压缩文件解压到临时目录中。将文件集成到系统中。

1. 提供类型库
2. 编译实例
3. 注册 COM 服务器
4. 将 COM 服务器集成到报表系统中

要将 COM 服务器集成到系统中需要广泛的编程技能，而无法在本文档中提供这些内容。章节“用于报表的 COM 接口要求”包含了有关正式 COM 服务器要求的说明。

实例提供的“*.REG”文件不是 Visual Basic 或 Visual C 编写的实例项目的一部分。它们用于将 COM 服务器插入页面布局编辑器的对象选项板中。

参见

用于报表的 COM 接口的详细资料 (页 504)

如何在报表中输出 COM 服务器的数据 (页 502)

3.5 用于报表的 COM 接口的详细资料

引言

本章节提供用于报表的 COM 接口的信息和要求。 更多说明可参见章节“COM 服务器数据输出”和“COM 服务器集成实例”。

调用接口

对象可以/必须提供下列 COM 接口，以便 WinCC 报表系统可以使用该对象：

```
interface IWinCCProtProvider :IDispatch
{
HRESULT Register([in]IDispatch* pIDispWinCCProtReportParams);
HRESULT Unregister();
HRESULT GetName([out, retval]BSTR* pName);
HRESULT ShowPrivateDialog([in]long hwndParent, [out, retval]BOOL* pfOK);
HRESULT SetPrivateData([in]VARIANT PrivateInfo);
HRESULT GetNameOfPrivateData([out, retval]BSTR* pPrivateInfoName);
```

```
HRESULT GetPrivateData([out, retval]VARIANT* pPrivateData);
};
interface IWinCCProtProviderText :IDispatch
{
HRESULT GetText([out, retval]BSTR* pName);
};
interface IWinCCProtProviderTable :IDispatch
{
HRESULT GetNumCols([out, retval]int* pnNumCols);
HRESULT GetNumLines([out, retval]int* pnNumLines);
HRESULT GetText([in]int nLine, [in]int nCol, [out, retval]BSTR* pName);
HRESULT HasHeader([out, retval]BOOL* pfHasHeader);
HRESULT GetHeader([in]int nCol, [out, retval]BSTR* pName);
};
Ìnterface IWinCCProtProviderPicture :IDispatch
{
HRESULT Draw( [in]long hdc, [in]int lef, [in]int top,
[in]int right, [in]int bottom);
};
```

对象必须支持接口 IWinCCProtProvider 和以下接口之一：IWinCCProtProviderText、IWinCCProtProviderTable 和 IWinCCProtProviderPicture。

CR+ 在 IWinCCProtProviderText 接口处用作换行符。

LF（CR =“回车”，LF =“换行符”）。

Interface IWinCCProtProvider

Register	在启动 COM 服务器后调用，将指向 IWinCCProtReportParams 的指针传递到服务器。
Unregister	调用来通知 COM 服务器释放指向接口 IWinCCProtReportParams 的指针。

GetName	返回 COM 服务器的名称，以将其显示在组态接口上。
ShowPrivateDialog	打开 COM 服务器选择对话框。
SetPrivateData	将保存在布局中的 SelCrit 数据传递给 COM 服务器。
GetPrivateData	从 COM 服务器读取 SelCrit 数据，将其保存在布局中。
GetNameOfPrivateDat a	返回选择规范的名称，将其显示在组态接口上。

Interface IWinCCProtProviderTable

GetNumCols	返回将打印在报表中的列数。
GetNumLines	返回将打印在报表中的行数。
HasHeader	返回关于是否应将表头打印在报表中的信息。
GetHeader	返回将打印在报表中的表头文本。
GetText	返回将打印在报表中的文本。

数据的特殊外观：

颜色、对齐等控制符始终位于输出文本之前，并可互相组合（例如“<B><U>输出文本”）。它们不区分大小写。

<END>	结束对控制序列的说明。 其余文本根据指定被接受。
<COLOR=#rrggbb>	十六进制符号的字体颜色（缺省值 = 用于表格的设置）
<BGCOLOR=#rrggbb >	十六进制符号的背景色（缺省值 = 用于表格的设置）
<B>	粗体字
<U>	下划线
<I>	斜体字
<STRIKE>	删除线
<ALIGN=left>	左对齐
<ALIGN=center>	居中
<ALIGN=right>	右对齐

Interface IWinCCProtProviderText

GetText	返回将打印在报表中的文本

Interface IWinCCProtProviderPicture

Draw	将选择标记移到与 COM 服务器连接的设备，并协调为可在其中进行绘制的形式。

此处，在指定的相关设备环境中绘制输出域。这是增强型图元文件。 在 MM_HIMETRIC 模式中进行绘制。

相关报表设备环境中读取参数的接口

报表系统提供从报表设备环境（报表系统和作业属性的设置）读取参数的接口。

```
Interface IWinCCProtReportParams :IDispatch
{
HRESULT GetParameter ( [in]BSTR PropertyName, [out]VARIANT* Value );
};
```

当前可以读取下列属性：

TimeFrom	打印作业参数
TimeTo	打印作业参数
PrivateSelCrit	例如，在按下 WinCC 控件中的“打印”按钮时使用。使用此 PrivateSelCrit，WinCC 控件的当前选择被发送到 COM 服务器。
ProjectName	WinCC 项目名称
LCID_APP	COM 服务器调用的当前在应用程序中设置的语言（PrintIt/ProtCS）。WinCC? 系统语言可能不同，因为 WinCC 不主动支持运行系统语言。
LCID_RT	WinCC 的当前运行系统语言。 该设置仅在运行系统中可见。

注册表条目

注册表中的条目根据 COM 对象自动进行输入，或者必须在注册表中通过调用注册表文件来输入。注册表文件必须由 COM 对象提供。如果没有这些注册表条目，即使注册了 COM 对象，它也不可用于报表系统。COM 对象由用户注册。

COM 服务器注册表条目实例：

HKEY_LOCAL_MACHINE\SOFTWARE\SIEMENS\WinCC\Report Designer \ReportClientDLLs\{4BF175C2-8BFF-11D0-840D-0080AD1374C8}]（COM 对象的 GUI-ID 作为唯一键）

"DllClientGUID"="{4BF175C2-8BFF-11D0-840D-0080AD1374C8}" (COM 对象的 GUID)

"DllFileName"="CCPComProvider.dll"

"NeedsRuntime"="NO"

„RunsOnServer„=„YES„

„RunsOnServer„=„YES„

„RunsOnMultiClient„=„YES„

"UseReportDesignerObjTab"="COM-Server"

应用程序对象的特性

COM 服务器中的调用被定时。

调试支持：

因此，超时特性使调试 COM 服务器变得不再困难，可以设置超时时间。如果未定义注册键，使用缺省值（10000 毫秒）。

HKEY_CURRENT_USER\Software\SIEMENS\WINCC\ReportSystem\TimeOuts\

InvokeTimeOut (vom Typ DWORD) --> 超时时间以毫秒计

如果输入 0xffffffff（-1）作为超时时间值，报表系统将无限等待其运行。

参见

如何在报表中输出 COM 服务器的数据 (页 502)

COM 服务器集成实例 (页 503)

OPC - 开放式互连

4

4 资源

4.1 OPC - 开放式互连

内容

使用 OPC 标准软件接口，各个生产商的设备和应用程序就能以统一的方式连接起来。

WinCC 可以用作 OPC 服务器或 OPC 客户机。 “OPC”通道表示 WinCC 的 OPC 客户机应用程序。

本章描述了

- WinCC 的 OPC 服务器
- 如何在 WinCC 中使用 OPC
- 如何建立不同的 OPC DA 链接
- 如何对 WinCC 消息系统的访问进行组态
- WinCC 消息系统如何在 OPC A&E 上显示
- 如何建立对 WinCC 归档系统的访问

4.2 OPC 的功能

简介

OPC（开放式互连）是一种一致的独立于制造商的软件接口。OPC 接口基于 Microsoft Windows 的 COM（组件对象模型）和 DCOM（分布式组件对象模型）技术。 另一方面，OPC XML 则基于 Internet 标准 XML、SOAP 和 HTTP。

COM

COM 是位于同一计算机上且属于不同程序的对象之间进行通讯时采用的标准协议。服务端是提供服务的对象，比如提供数据。客户端是使用由服务端提供的服务的应用程序。

DCOM

DCOM 代表 COM 功能的扩展，从而允许对远程计算机上的对象的访问。

该接口允许在企业、管理办公室和生产的应用程序之间进行标准化的数据交换。

以前，访问过程数据的应用程序受限于通讯网络的访问协议。使用 OPC 标准软件接口，各个生产商的设备和应用程序就能以一致的方式连接起来。

OPC 客户端是访问过程数据、消息和 OPC 服务端归档的应用程序。访问需要通过 OPC 软件接口。

OPC 服务端是一个程序，它为不同制造商的应用程序提供一个标准的软件接口。OPC 服务端是在处理过程数据的应用程序、各种网络协议和用于访问这些数据的接口之间的中间层。

只有当设备的操作系统是基于 Windows COM 和 DCOM 技术时，才能使用 OPC 软件接口进行数据交换。 目前，Windows 2000、Windows XP、Windows 2003 Server 和 Windows VISTA 具有这些软件接口。

XML

通过 DCOM 的通讯仅限于局域网。通过 XML 的数据交换使用 SOAP（简单对象访问协议）。SOAP 是独立于平台的、基于 XML 的协议。SOAP 可用于允许应用程序依靠 HTTP（超文本传送协议），通过 Internet 或在多机种计算机网络内相互进行通讯。

4.3 OPC 规范

简介

OPC 标准软件接口由 OPC 基金会定义。OPC 基金会是工业自动化领域中处于领先地位的各公司的联盟。WinCC 的 OPC 服务端支持下列规范。

- OPC 数据访问 1.0、2.05a 和 3.0
- OPC XML 数据访问 1.01
- OPC 历史数据访问 1.20
- OPC 报警和事件 1.10

OPC 数据访问 (OPC DA)

OPC 数据访问 (OPC DA) 是针对管理过程数据的规范。 WinCC OPC DA 服务端符合 OPC DA 规范 1.0、2.05a 和 3.0。

OPC 可扩展标记语言 DA (OPC XML DA)

OPC XML 标准支持通过 Internet 采用独立于平台的协议进行通讯。 客户端不再局限于 Windows 环境 (DCOM)。 其它操作系统（如 LINUX）可以使用 HTTP 协议和 SOAP 接口在 Internet 上监视和交换 OPC 数据。

通过 OPC XML 进行数据访问具有与 OPC 数据访问相似的功能范围。没有为 OPC XML 设计涉及数据修改（例如对 DCOM 接口的修改）的更改相关反馈消息，因为它们会使 Internet 连接变得不稳定。

OPC 历史数据访问 (OPC HDA)

OPC 历史数据访问 (OPC HDA) 是针对管理归档数据的规范。该规范是 OPC 数据访问规范的扩充。 WinCC V6.2 或更高版本的 WinCC OPC HDA 服务端符合 OPC HDA 规范 1.20。

OPC 报警和事件 (OPC A&E)

OPC 报警和事件是发送过程报警和事件的补充规范。 WinCC V7.0 或更高版本的 WinCC OPC A&E 服务端符合 OPC A&E 规范 1.10。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

4.4 兼容性

简介

OPC 基金会向会员提供用于 OPC 服务端产品的自动化兼容性测试的软件工具。

提交的结果会在 OPC 基金会的网站上公布。可以从该网站使用搜索术语“OPC Self-Certified Products”来调用结果。

该工具使用 OPC 基金会的 OPC 规范来检查 WinCC OPC 服务端的合规性。

此外还会进行定期测试，以确定 WinCC OPC 服务端和 WinCC OPC DA 客户端是否与其它 OPC 产品兼容。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

4.5 在 WinCC 中使用 OPC

引言

WinCC 可以用作 OPC 服务器和 OPC 客户机。 在 WinCC 安装期间，可选择下列 WinCC OPC 服务器进行安装：

- WinCC OPC DA 服务器
- WinCC OPC XML DA 服务器
- WinCC OPC HDA 服务器
- WinCC OPC A&E 服务器

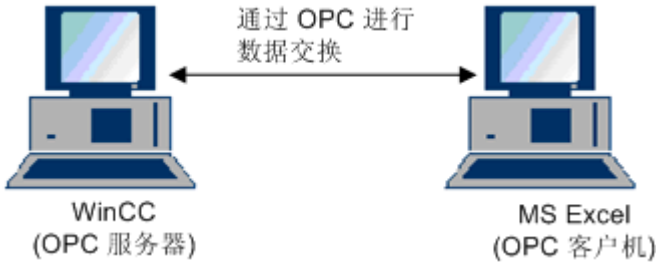
许可：

要使用 WinCC OPC HDA、WinCC-OPC-XML-DA 服务器和 WinCC OPC A&E 服务器，必须购买许可证。 “连接组件包”许可证必须安装在用作 WinCC OPC HDA 服务器、WinCC OPC XML DA 服务器或 WinCC OPC A&E 服务器的 WinCC 服务器上。 有关详细信息，请参阅章节“许可证颁发”。

应用程序

WinCC 作为 OPC DA 服务器

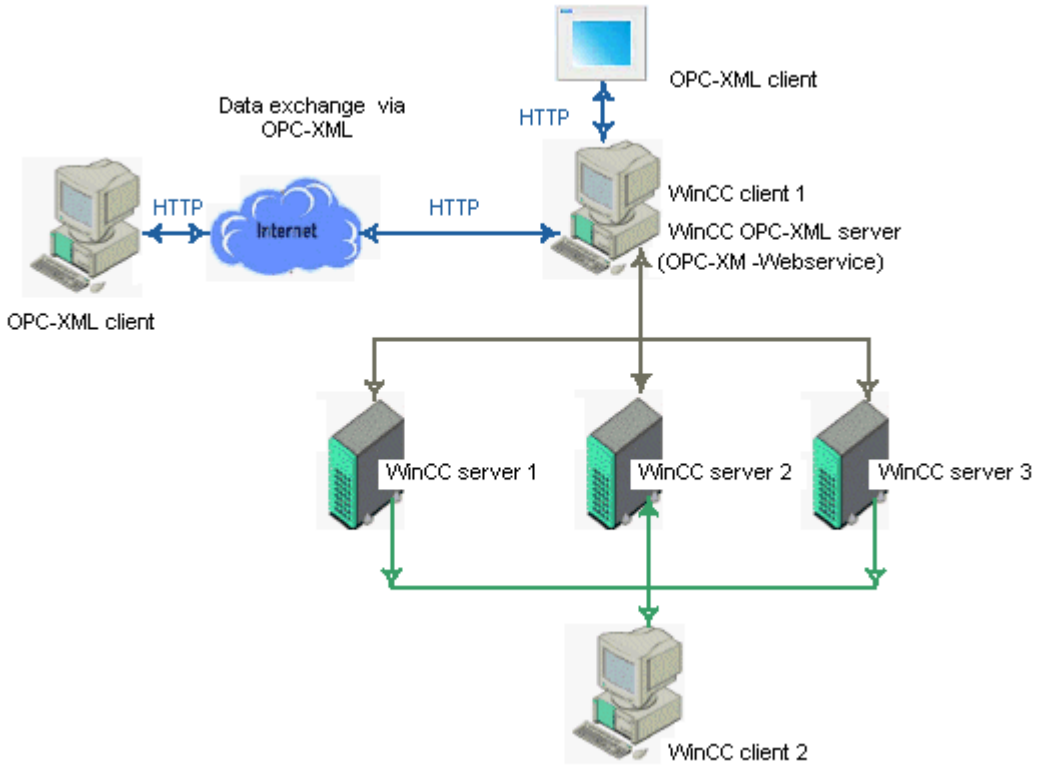
WinCC OPC DA 服务器为其它应用程序提供了 WinCC 项目的数据。 应用程序能够在同一计算机上运行或在已联网的计算机上运行。 例如，以这种方法，WinCC 变量可导出到 Microsoft Excel。



WinCC 作为 OPC-XML-DA 服务器

在分布式系统中，WinCC 客户机具有多个 WinCC 服务器的视图。WinCC OPC XML DA 服务器以 Web 页面形式为 OPC XML 客户机提供 OPC 过程数据。可以使用 HTTP 通过

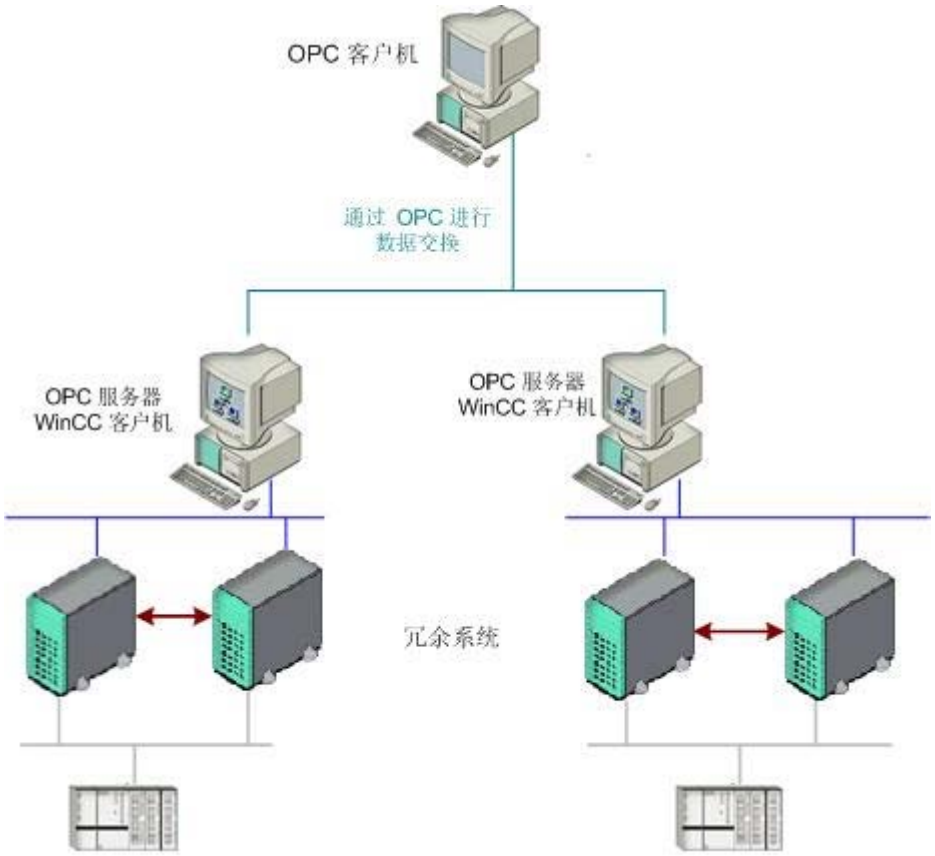
Internet 访问 Web 页面。 OPC XML 客户机不再限于局域网。 这样，OPC XML 客户机便可通过任何平台以及通过 Intranet 或 Internet 访问 WinCC 运行系统数据。



冗余系统中的 WinCC OPC 服务器

在冗余系统中，WinCC 服务器在运行系统中相互监视以便尽早识别服务器故障。 使用 OPC 软件接口，WinCC OPC 服务器允许 OPC 客户机访问 WinCC 运行系统数据。

任何基于相应的 OPC 规范的软件程序都可用作 OPC 客户机。 因此可以使用 OPC 客户机来集中地监控各种冗余系统。 通过创建专有 OPC 客户机，可以满足大多数用户特定需求。



4.6 WinCC OPC XML DA 服务器

4.6.1 WinCC OPC XML DA 服务器的功能

引言

WinCC 的 OPC XML DA 服务器是 Microsoft Internet 信息服务（IIS）的 Web 服务的一部分。

WinCC OPC XML DA 服务器以 Web 页面形式为 OPC XML 客户机提供 OPC 过程数据。可以使用 HTTP 通过 Internet 访问 Web 页面。 WinCC OPC XML DA 服务器的地址是：<http://<xxx>/WinCC-OPC-XML/DAWebservice.asmx>

WinCC OPC XML DA 服务器在 WinCC 中不可见。 OPC XML 客户机请求数据时，Web 服务器便自动启动 Web 服务。

为了建立成功的 OPC 通讯，必须遵守以下内容：

- 必须激活 WinCC OPC XML DA 服务器的 WinCC 项目。
- 必须能够通过 HTTP 访问 WinCC OPC XML DA 服务器的计算机。

许可证

为了操作 WinCC OPC XML DA 服务器，必须在每台作为 OPC XML 服务器的 WinCC 计算机上安装下列许可证：

- WinCC 基本系统
- WinCC 选件数据连通性软件包（Connectivity Pack）

“字符串”类型变量的特殊功能

如果使用逻辑上代表浮点值的“字符串”类型的变量，OPC 客户机要写入和读取时可能会出现问题。

描述

OPC 客户机写入到字符串变量，并不会以字符串形式输入新值，而是作为浮点、双精度或十进制处理。

问题

表示小数值（德语）的逗号可能会丢失。 这会导致错误的值。

如果以浮点、双精度或十进制格式请求读取值，还会影响对字符串变量的读访问。

纠正方法

只使用相应的浮点型变量用于浮点值。 仅以字符串格式访问字符串变量。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

4.6.2 安装

4.6.2.1 安装

引言

OPC-XML 允许通过 Internet 访问过程变量。 为了操作 OPC XML，必须安装各种附加的软件组件。

说明

必须遵守此处所述安装步骤的顺序。 否则，在安装期间可能会出现问题。

在 Windows 2000 和 Windows XP 下安装 WinCC-OPC-XML-Server

1. 安装 Internet 信息服务器（IIS）
2. 在 Windows Server 2003 下安装 ASP.NET
3. 安装 Microsoft .NET-Framework V2.0
4. 使用 WinCC 安装程序安装 WinCC OPC XML 服务器

4.6.2.2 安装 Internet 信息服务（IIS）

引言

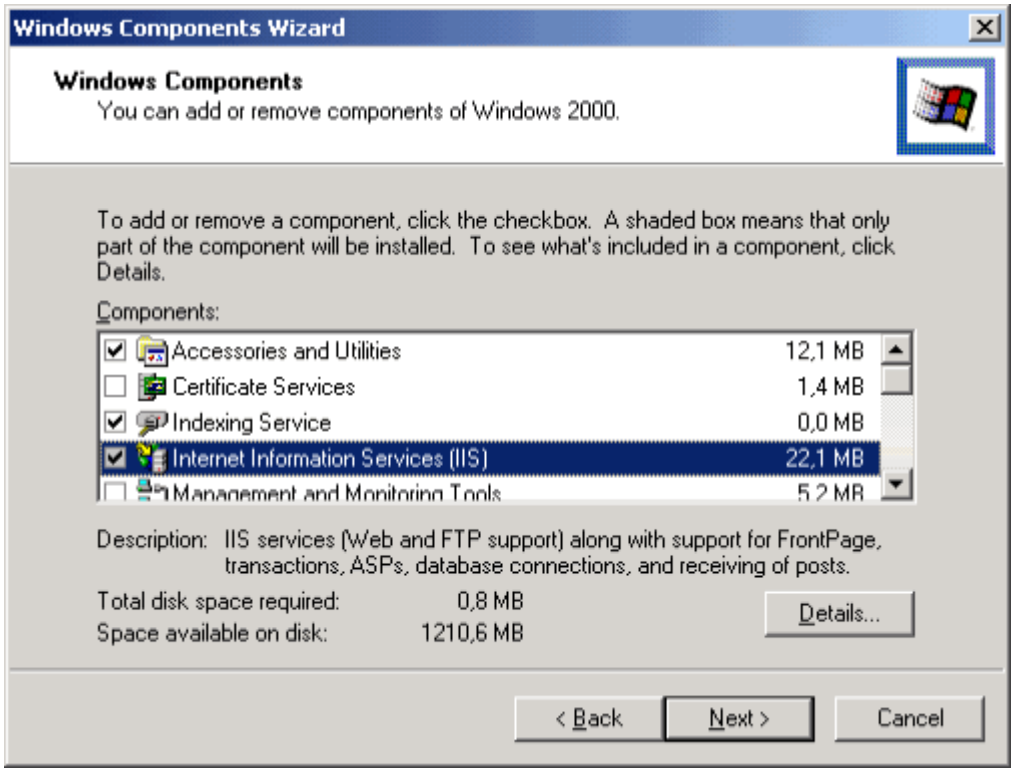
在 Windows 2000 Professional、Windows 2000 Server、Windows 2003 Server 或 Windows XP 下，安装 WinCC OPC XML 服务器之前必须先安装 Internet 信息服务（IIS）。

说明

要在 Windows 中安装 Internet 信息服务，必须具有必要的注册表数据库写权限。 为此，必须具有管理员权限。

步骤

1. 打开“添加或删除程序”对话框，并单击“添加或删除 Windows 组件”。 以下对话框将会打开。
2. 在选择框中选择“Internet 信息服务（IIS）”条目前面的复选框。 在 Windows 2003 服务器中，可通过“Windows 组件”>“应用程序服务器”>“Internet 信息服务”来找到 IIS。 在 Windows Server 2003 下还必须安装 ASP.NET



3. 单击“下一步”。 然后，Windows 会传送所需的数据，并完成必需的组态。
4. 单击“完成”，关闭助手。

4.6.2.3 安装 Microsoft .NET Framework

Microsoft .NET Framework 是使用 .NET 应用程序所必需的软件。 它还需要运行 OPC XML Web 服务。

安装大约需要 80 MB 的硬盘存储空间。

可以从下列位置获得 Microsoft .NET Framework 2.0：

- 从 http://msdn.microsoft.com/downloads 下载。
- Microsoft Windows 2003 Server 光盘

根据 Microsoft 的指示安装软件。

4.6.2.4 安装 WinCC OPC XML DA 服务器

要求

- Internet 信息服务器（IIS）
- 在 Windows Server 2003 下安装 ASP.NET
- Microsoft .NET Framework 2.0

安装

在安装 WinCC 过程中，可选择 WinCC OPC XML DA 服务器。 有关更多信息，请参阅 WinCC 信息系统中的“安装注意事项”>“安装 WinCC”一章。

安装期间必须进行下列设置：

- 创建虚拟目录“WinCC-OPC-XML”。
- 为目录定义访问权限。

参见

如何测试安装 (页 596)

定义 IIS 的安全性设置 (页 594)

4.7 WinCC OPC DA 服务器

4.7.1 WinCC OPC DA 服务端的功能

简介

WinCC OPC DA 服务端支持 OPC 数据访问规范 1.0、2.05a 和 3.0。这已经由兼容性测试确认。 WinCC V6.0 SP2 或更高版本的 WinCC OPC DA 服务端支持 OPC 数据访问 3.0 规范。

WinCC OPC DA 服务端是一个 DCOM 应用程序。WinCC OPC DA 服务端使用该接口向 WinCC 客户端提供关于 WinCC 变量的所需信息。

如果 WinCC OPC DA 客户端正在通过连接访问 WinCC OPC DA 服务端，则服务端处于活动状态。为了建立成功的 OPC 通讯，必须遵守以下内容：

- WinCC OPC DA 服务端的 WinCC 项目必须激活。
- 运行 WinCC OPC DA 服务端的计算机必须能够通过其 IP 地址访问。

安装

在安装 WinCC 过程中，可选择 WinCC OPC DA 服务端。安装后，WinCC OPC DA 服务端无需进一步组态便立即可用。

可以在 WinCC 服务器或 WinCC 客户机上安装 WinCC OPC DA 服务端。

组态提示

在 WinCC 项目中，可将变量汇总在变量组里，以实现结构化。变量名称不能与组名称相同。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

兼容性 (页 511)

查询 OPC DA 服务器名称 (页 521)

使用多个 OPC DA 服务器 (页 520)

WinCC 至 WinCC 的连接实例 (页 523)

WinCC - SIMATIC NET FMS OPC 服务器连接的实例 (页 527)

WinCC - SIMATIC NET S7 OPC 服务器连接的实例 (页 529)

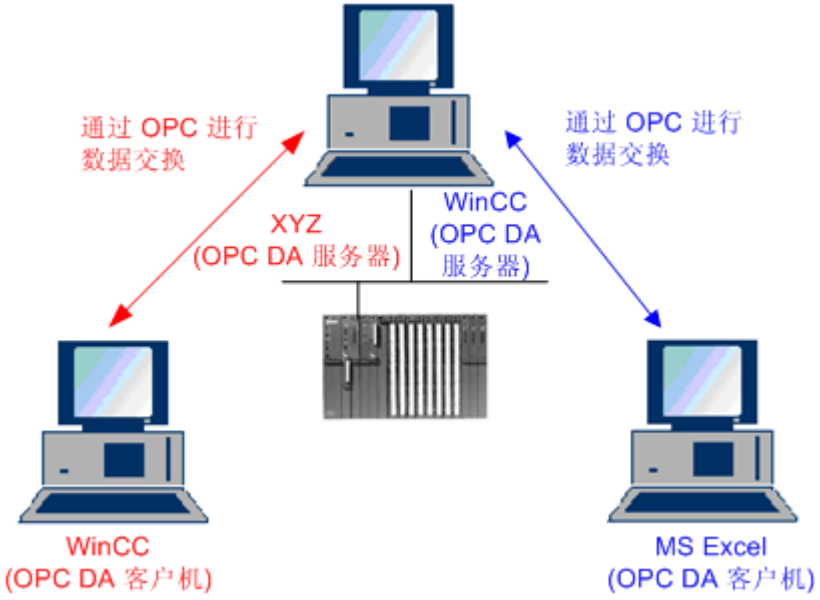
WinCC - Microsoft Excel 连接的实例 (页 535)

4.7.2 使用多个 OPC DA 服务器

简介

在一台计算机上可以安装一个以上的 OPC DA 服务端，且任意数目的服务端都可并行地工作。
通过这种方式，WinCC 的 OPC DA 服务端和另一（第三方）供应商的 OPC DA 服务端可在同一台计算机上彼此独立地运行。

WinCC OPC DA 客户端能通过第三方供应商的 OPC 服务端，访问自动化设备的过程数据。Microsoft Excel 的 OPC DA 客户端能使用 WinCC OPC DA 服务端访问 WinCC 数据。



有不同生产商提供的许多 OPC DA 服务端可用。其中每个 OPC DA 服务端都有唯一的名称 (ProgID) 以便识别。 OPC DA 客户端必须使用该名称对 OPC 服务端进行寻址。

OPC 条目管理器可用来查询 OPC DA 服务端的名称。 WinCC V7 的 OPC DA 服务端的名称为：“OPCServer.WinCC”。

参见

查询 OPC DA 服务器名称 (页 521)

4.7.3 查询 OPC DA 服务器名称

引言

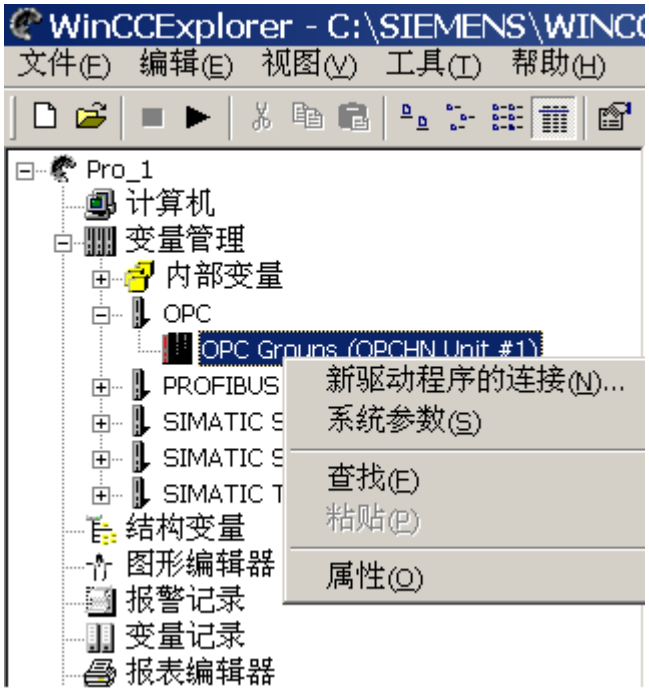
多个 OPC DA 服务器可安装在一台计算机上。 OPC 条目管理器在选择窗口中显示可用于工作站的 OPC DA 服务器的名称。 这些 OPC DA 服务器能够在同一计算机上运行或在已联网的计算机上运行。

要求

在 WinCC OPC DA 客户机的 WinCC 项目中添加“OPC”通道。

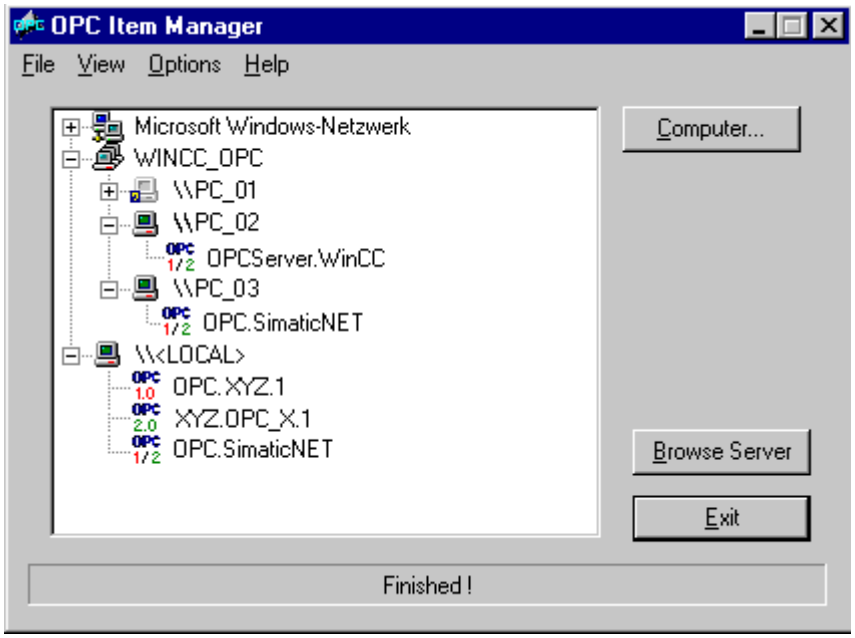
步骤

1. 在 WinCC OPC DA 客户机上，从“OPC 组 (OPCHN Unit#1)”通道单元的快捷菜单中选择“系统参数”。



将打开 OPC 条目管理器。

2. 在 OPC 条目管理器的浏览窗口中，选择要访问的计算机名称。
3. OPC 条目管理器在选择窗口中显示可用于计算机的 OPC DA 服务器的名称。



4.7.4 OPC DA 连接的实例

4.7.4.1 WinCC - WinCC 连接

WinCC 至 WinCC 的连接实例

引言

当建立 WinCC - WinCC 连接时，WinCC OPC DA 服务器和客户机之间的数据交换通过 WinCC 变量“OPC_Server_Tag”进行。 客户机上的 WinCC 变量“Client_OPC_Server_Tag_xyz”读取服务器上的 WinCC 变量“OPC_Server_Tag”。 如果 WinCC OPC 服务器上的“OPC_Server_Tag”变量的数值改变，WinCC OPC DA 客户机上的 WinCC 变量“Client_OPC_Server_Tag_xyz”的值也改变。 客户机上的改变也会在服务器上反映出来。

变量值显示在两台计算机的 I/O 域中。

要求

- 两台带 WinCC 项目的计算机。
- 这两台计算机必须能够通过其 IP 地址被访问。

组态过程

建立 WinCC - WinCC 的连接需要以下组态：

1. 组态 WinCC OPC DA 服务器上的 WinCC 项目
2. 组态 WinCC OPC DA 客户机上的 WinCC 项目

参见

如何组态 WinCC OPC DA 服务器上的 WinCC 项目 (页 524)

组态 WinCC OPC DA 客户机上的 WinCC 项目 (页 525)

如何组态 WinCC OPC DA 服务器上的 WinCC 项目

引言

本节中介绍在 WinCC OPC DA 服务器上的 WinCC 项目中创建一个 WinCC 变量并显示在 I/O 域内。

步骤

1. 从 WinCC OPC DA 服务器上“内部变量”图标的快捷菜单中选择“新建变量”。 创建一个类型为“有符号 16 位数”，名为“OPC_Server_Tag”的新变量。
2. 启动“图形编辑器”，打开一个新的画面。
3. 在画面中添加一个 I/O 域。从“智能对象”下的对象列表中选择“I/O 域”对象。 将打开“I/O 域组态”对话框。

4. 在“变量”域中，输入名称“OPC_Server_Tag”。
5. 设置更新周期为“2 秒”，域类型为“I/O 域”。
6. 单击“确定”关闭对话框并保存画面。
7. 单击图形编辑器中的“激活”按钮以启用 WinCC 项目。

参见

组态 WinCC OPC DA 客户机上的 WinCC 项目 (页 525)

组态 WinCC OPC DA 客户机上的 WinCC 项目

引言

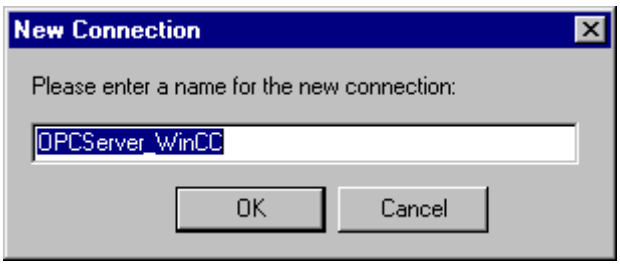
在本节中，将在 WinCC OPC DA 客户机上创建 WinCC 变量，以读取 WinCC OPC DA 服务器上的 WinCC 变量。变量值显示在 I/O 域中。

要求

- 在 WinCC OPC DA 客户机的 WinCC 项目中添加“OPC”通道。
- 在 WinCC OPC DA 服务器的 WinCC 项目中，组态一个数据类型为“有符号 16 位数”，名为“OPC_Server_Tag”的内部变量。
- 启用 WinCC OPC DA 服务器的 WinCC 项目。

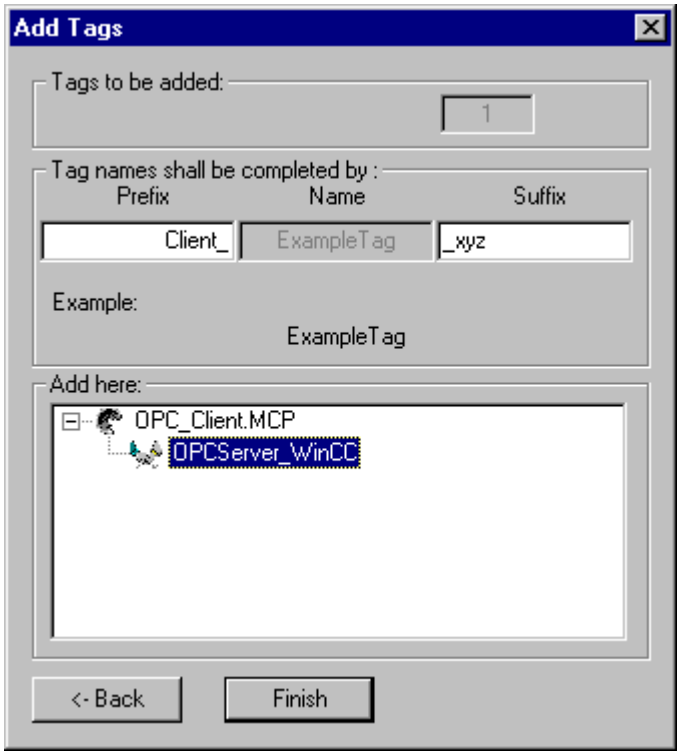
步骤

1. 在 WinCC OPC DA 客户机上，从“OPC 组 (OPCHN Unit#1)”通道单元的快捷菜单中选择“系统参数”。将打开 OPC 条目管理器。
2. 从选择对话框中选择要用作 OPC DA 服务器的计算机名称。从列表中选择“OPCServer.WinCC”。单击“浏览服务器”按钮。将打开“过滤标准”对话框。
3. 在“过滤标准”对话框中，单击“下一步->”按钮。在“OPCServer.WinCC ...”对话框中选择“OPC_Server_Tag”变量。单击“添加条目”按钮。
4. 如果到 OPC DA 服务器的连接已经存在，继续步骤 5。
   如果尚未组态连接，则会显示相应的消息。
   单击“是”。显示“New Connection”（新建连接）对话框。



   输入“OPCServer_WinCC”作为连接的名称。单击“OK”（确定）。

5. 显示“Add Tags”（添加变量）对话框。
在前缀域中输入“Client_”，在后缀域中输入“_xyz”。 选择连接“OPCServer_WinCC”。 单击“Finish”（完成）。



6. 在“OPCServer.WinCC ...”对话框中，单击“<-返回”按钮。在“OPC 条目管理器”中，单击“退出”关闭 OPC 条目管理器。
7. 启动“图形编辑器”，打开一个新的画面。在画面中添加一个 I/O 域。从“智能对象”下的对象列表中选择“I/O 域”对象。 将打开“I/O 域组态”对话框。
8. 在“变量”域中输入名称“Client_OPC_Server_Tag_xyz”。 将更新周期设置为“2 秒”。 将域类型设置为“I/O 域”。 关闭对话框并保存画面。 单击图形编辑器中的“激活”按钮以启用 WinCC 项目。
9. 在 WinCC OPC DA 服务器和客户机上的 I/O 域中，将显示组态变量的数值。 在 WinCC OPC DA 服务器的 I/O 域中输入新的数值。 新数值将显示在 WinCC OPC DA 客户机上的 I/O 域内。

参见

如何组态 WinCC OPC DA 服务器上的 WinCC 项目 (页 524)

4.7.4.2 WinCC - SIMATIC NET FMS OPC 服务器的连接

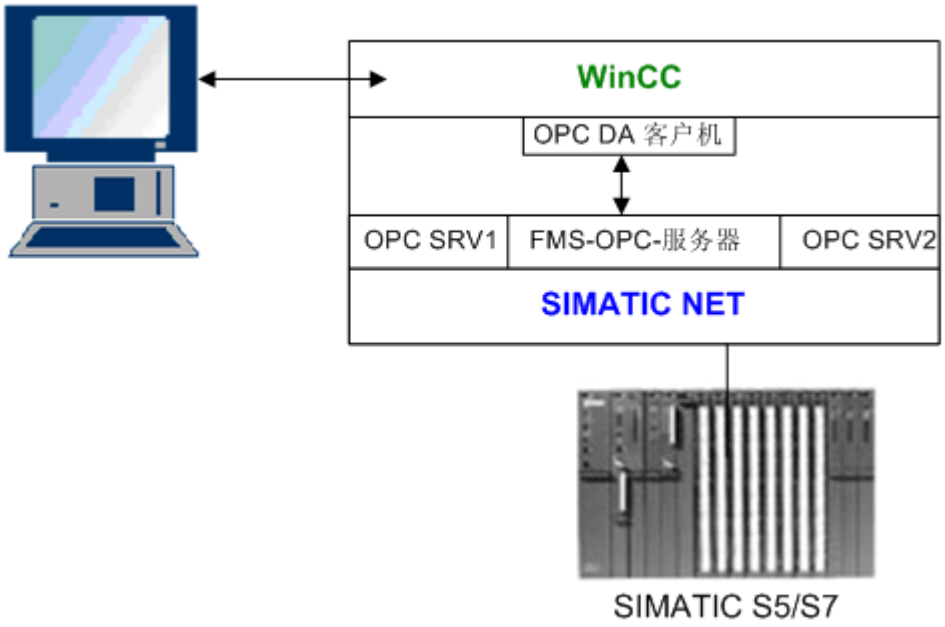
WinCC - SIMATIC NET FMS OPC 服务器连接的实例

简介

在 SIMATIC NET 的安装期间，可以选择要安装的 OPC 服务端。在以下实例中，将组态 WinCC 和 SIMATIC NET FMS OPC 服务端之间的连接。通过 SIMATIC NET FMS OPC 服务端，WinCC 可使用来自自动化设备的数据。

在本实例中，WinCC 用作 WinCC OPC DA 客户端。OPC 条目管理器显示为自动化设备组态的对象列表索引。

变量的当前值显示在一个 I/O 域中。SIMATIC NET FMS OPC 服务端上的变量值改变时，新的数值将会在 WinCC OPC DA 客户端的过程画面中反映出来。反之，在 I/O 域中输入的数值会发送到自动化设备上。



要求

- 安装有 WinCC、SIMATIC NET 软件的计算机。
- 已组态 SIMATIC NET FMS OPC 服务端。有关 SIMATIC NET S7 OPC 服务端设置的更多信息，请参阅 SIMATIC NET 文档。

组态步骤

在 WinCC OPC DA 客户端的 WinCC 项目中，需要下列组态：

1. 组态 WinCC - SIMATIC NET FMS OPC 服务端的连接

通讯手册

通讯手册包含通道组态的更多信息和扩展实例。 本手册可在 Internet 上下载，地址如下：

- http://support.automation.siemens.com/

搜索订货号：

- A5E00391327

如何组态 WinCC - SIMATIC NET FMS OPC 服务器的连接

引言

本节中，在 WinCC OPC DA 客户机的 WinCC 项目中组态一个访问 FMS 索引的 WinCC 变量。变量值显示在 I/O 域中。

要求

- 在 WinCC OPC DA 客户机的 WinCC 项目中添加“OPC”通道。

步骤

1. 在 WinCC OPC DA 客户机上，从“OPC 组 (OPCHN Unit#1)”通道单元的快捷菜单中选择“系统参数”。将打开 OPC 条目管理器。
2. 从选择对话框中选择要用作 OPC DA 服务器的计算机名称。从列表中选择“OPC.SIMATICNet”。
   单击“浏览服务器”按钮。将打开“过滤标准”对话框。
3. 在“过滤标准”对话框中，单击“下一步->”按钮。 “OPC.SIMATICNet..”对话框打开。 所有组态的 FMS 索引将显示在选择列表中。 选择索引。 单击“添加条目”按钮。

4. 如果到 SIMATIC NET FMS OPC 服务器的连接已经存在，继续步骤 5。
如果尚未组态连接，则会显示相应的消息。
单击“是”。 显示“New Connection”（新建连接）对话框。

输入“OPC_SImaticNET”作为连接的名称。 单击“OK”（确定）。

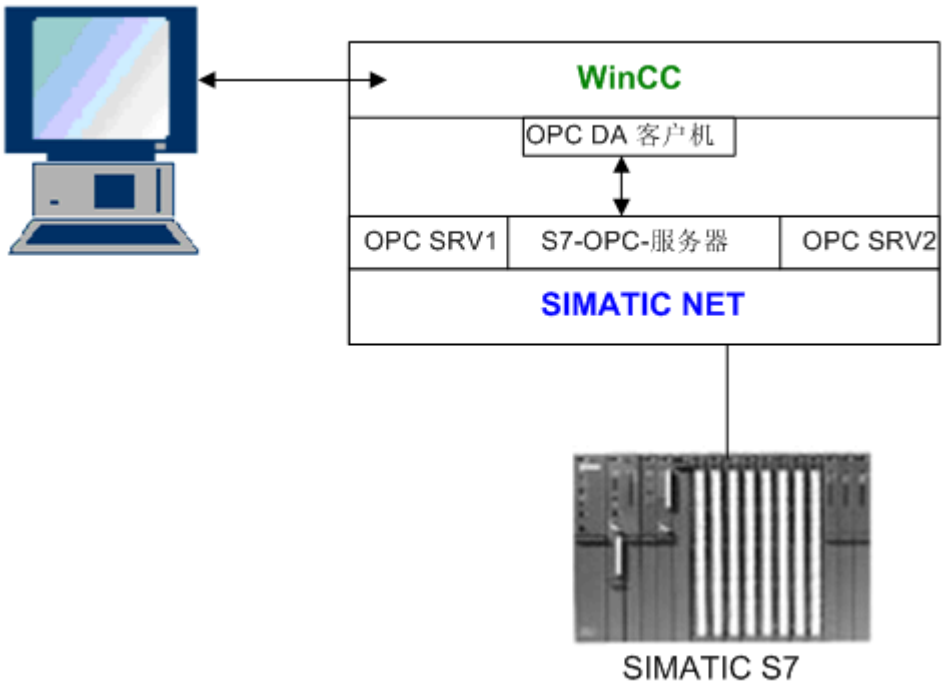
5. 将打开“添加变量”对话框。
在前缀域中输入“Client_”，在后缀域中输入“_xyz”。 选择连接“OPC_SimaticNET”。 单击“完成”。
6. 在“OPC.SIMATICNet..”对话框中，单击“<- Back”（<- 返回）按钮。在“OPC 条目管理器”中，单击“退出”关闭 OPC 条目管理器。
7. 启动“图形编辑器”，打开一个新的画面。在画面中添加一个 I/O 域。从“智能对象”下的对象列表中选择“I/O 域”对象。 将打开“I/O 域组态”对话框。
8. 在“变量”域中，输入变量的名称。 将更新周期设置为“2 秒”。 将域类型设置为“I/O 域”。
9. 单击“确定”关闭对话框并保存画面。 单击图形编辑器中的“激活”按钮以启用 WinCC 项目。
10. 在 I/O 域中，将显示 FMS 索引的当前值。 数值每 2 秒更新一次。 在 I/O 域中输入数值。更改后的数值将传递到自动化设备。

4.7.4.3 WinCC - SIMATIC NET S7-OPC 服务器的连接

WinCC - SIMATIC NET S7 OPC 服务器连接的实例

在 SIMATIC NET 的安装期间，可以选择要安装的 OPC 服务端。在下面的实例中，将组态 WinCC 到 SIMATIC NET S7 OPC 服务端的连接。通过 SIMATIC NET S7 OPC 服务端，WinCC 客户机可使用来自自动化设备的数据。

变量的当前值显示在 WinCC OPC 客户端的一个 I/O 域中。一旦 SIMATIC NET S7 OPC 服务端上的变量值改变，更改后的数值将显示在过程画面上。反之，在 I/O 域中输入的数值会发送到自动化设备上。



要求

- 安装有 WinCC、SIMATIC NET 软件的计算机。
- 已组态 SIMATIC NET S7 OPC 服务端。有关 SIMATIC NET S7 OPC 服务端设置的更多信息，请参阅 SIMATIC NET 文档。

组态步骤

为了建立 WinCC 到 SIMATIC NET S7 OPC 服务端的连接，需要下列组态：

1. 添加变量到 SIMATIC NET S7 OPC 服务端
2. 组态对 SIMATIC NET S7 OPC 服务端变量的访问

通讯手册

通讯手册包含通道组态的更多信息和扩展实例。 本手册可在 Internet 上下载，地址如下：

- http://support.automation.siemens.com/

搜索订货号：

- A5E00391327

添加变量到 SIMATIC NET S7 OPC 服务器

引言

为了用 OPC 条目管理器显示变量，必须添加变量至 SIMATIC NET S7 OPC 服务器的地址空间。 使用“OPC Scout”程序进行组态。 使用 SIMATIC NET 安装程序安装 OPC Scout。 对于本实例，将为自动化设备上的标记字“0”编址。

所用参数表

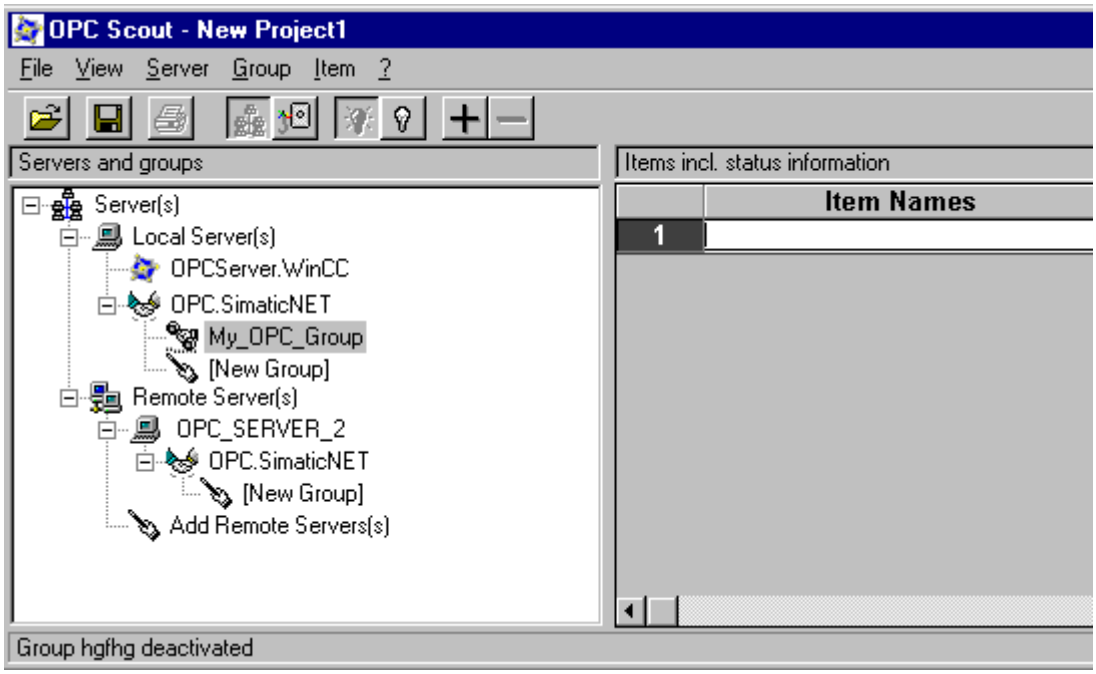
参数	数值
Data type	W
Range Byte	0
No. Values	1
Item alias	MW0

要求

- 在 SIMATIC NET 软件中组态 S7 连接。 有关更多的信息，请参考 SIMATIC NET 文档。

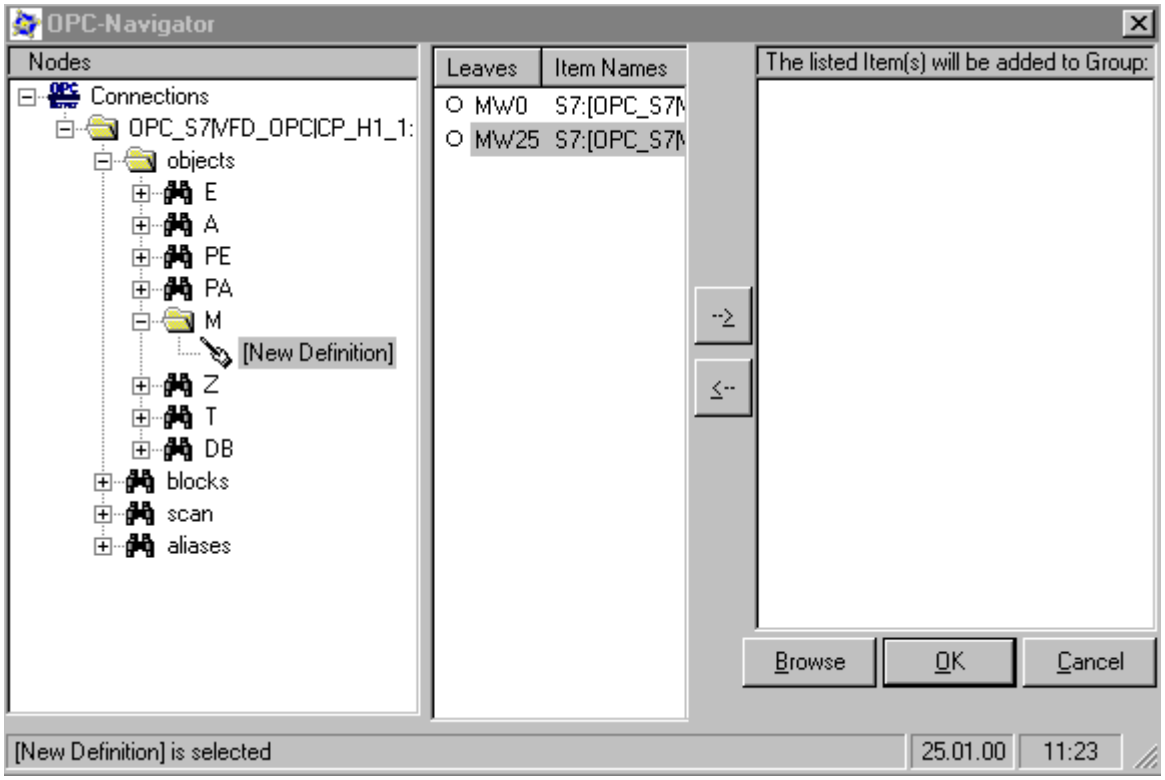
步骤

1. 通过“开始” → “程序” → “SimaticNet” → “OPCServer” → “OPCScout”打开“OPC Scout”。

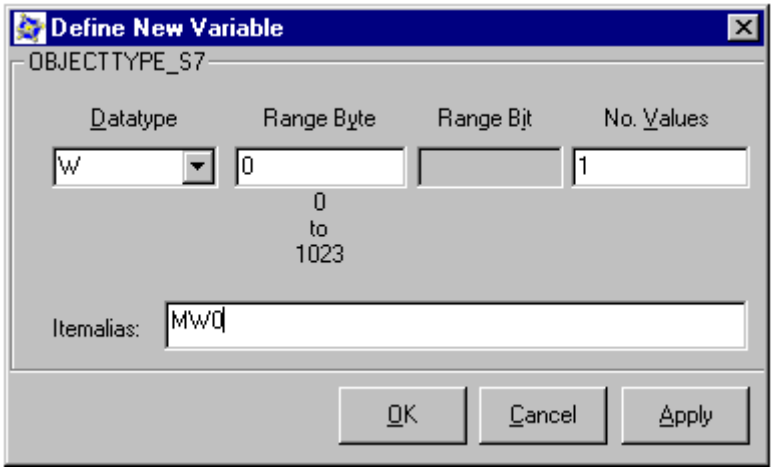


2. 选择“Local Server(s)”（本地服务器）下的“OPC.SimaticNet”。 如果 SIMATIC S7 OPC 服务器不是运行在同一台计算机上，则选择“Server(s)”（服务器）快捷菜单中的“Add Remote Server(s)”（添加远程服务器）。 在“Add Remote Server(s)”（添加远程服务器）对话框中输入用作 OPC 服务器的计算机名称，然后单击“OK”（确定）关闭对话框。
3. 选择“OPC.SimaticNet”快捷菜单中的“Connect”（连接）。 显示“Add Group”（添加组）对话框。 为该组输入一个名称。 单击“OK”（确定）关闭对话框。

4. 从所添加组的快捷菜单中选择“Add Item”（添加条目）。 “OPC Navigator”（OPC 浏览器）打开。



5. 在“OPC Navigator”（OPC 浏览器）中选择“Objects”（对象）下的“M”(标记)。 双击“(New Definition)”（新的定义）打开“Define New Tag”（定义新的变量）对话框。
6. 在“Define New Tag”（定义新的变量）对话框中输入表格中的参数。



单击“OK”（确定）关闭“Define New Tag”（定义新的变量）对话框。

7. 在 OPC 浏览器的“Leaves”（叶）区域中标记变量“MW0”。 单击“-->”按钮。 单击 OPC 浏览器中的“OK”（确定）。

参见

组态对 SIMATIC NET S7 OPC 服务器变量的访问 (页 534)

组态对 SIMATIC NET S7 OPC 服务器变量的访问

引言

本节中，将在 WinCC OPC DA 客户机的 WinCC 项目中组态 WinCC 变量。 该变量将访问 SIMATIC NET S7 OPC 服务器地址空间中的变量“MW0”。 变量值显示在 I/O 域中。

要求

- 使用 OPC Scout 创建“MW0”变量。
- 在 WinCC OPC DA 客户机的 WinCC 项目中添加“OPC”通道。

步骤

1. 选择“OPC 组（OPCHN Unit#1）”快捷菜单中的“系统参数”。 将打开 OPC 条目管理器。
2. 从选择对话框中选择用作 OPC 服务器的计算机名称。 从列表中选择“OPC.SIMATICNet”。
   单击“浏览服务器”按钮。 将打开“过滤标准”对话框。
3. 在“过滤标准”对话框中，单击“下一步->”按钮。 “OPC.SIMATICNet..”对话框打开。 选择“MW0”变量。 单击“添加条目”按钮。
4. 如果到 SIMATIC NET FMS OPC 服务器的连接已经存在，继续步骤 5。
   如果尚未组态连接，则会显示相应的消息。
   单击“是”。 显示“New Connection”（新建连接）对话框。

   输入“OPC_SIimaticNET”作为连接的名称。 单击“OK”（确定）。
5. 将打开“添加变量”对话框。
   在前缀域中输入“Client_”，在后缀域中输入“_xyz”。 选择连接“OPC_SimaticNET”。 单击“完成”。
6. 在“OPC.SIMATICNet..”对话框中，单击“<- Back”（<- 返回）按钮。 在“OPC 条目管理器”中，单击“退出”关闭 OPC 条目管理器。
7. 启动图形编辑器并打开画面。 在画面中添加一个 I/O 域。 从“智能对象”对象列表中选择“I/O 域”对象。 将打开“I/O 域组态”对话框。
8. 在“变量”域中，输入名称“Client_MW0_xyz”。 将更新周期设置为“2 秒”。 将域类型设置为“I/O 域”。

9. 关闭对话框并保存画面。 单击图形编辑器中的“激活”按钮以启用 WinCC 项目。
10. S7 变量的当前值显示在 WinCC OPC DA 客户机的 I/O 域中。 数值每 2 秒更新一次。 在 I/O 域中输入数值。 更改后的数值将传递到自动化设备。

参见

添加变量到 SIMATIC NET S7 OPC 服务器 (页 531)

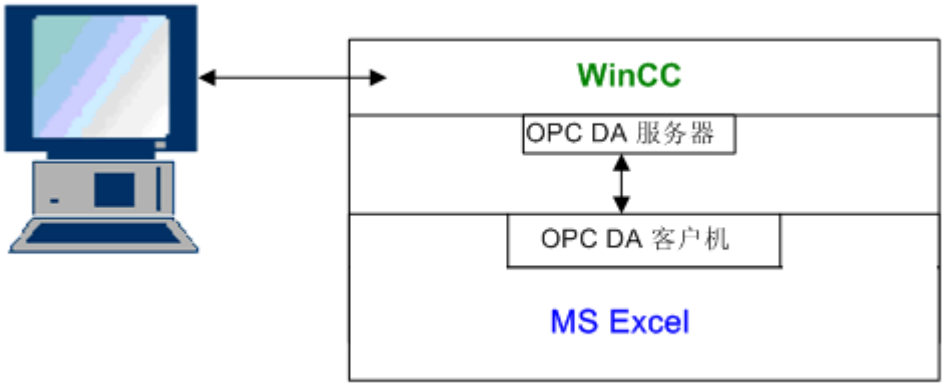
4.7.4.4 WinCC - Microsoft Excel 的连接

WinCC - Microsoft Excel 连接的实例

简介

在本实例中，将在 Microsoft Excel 中使用 Visual Basic 编辑器创建一个 OPC DA 客户机。该 OPC DA 客户机读取 WinCC OPC DA 服务器上 WinCC 项目中的 WinCC 变量，并将值写入单元格中。如果在该单元格中输入一个新值，该值将会传递到 WinCC OPC DA 服务器上。

需使用同时安装了 WinCC 和 Microsoft Excel 的计算机来进行连接。



组态步骤

必须在 Microsoft Excel 中进行下列组态：

1. 在 Microsoft Excel 的 Visual Basic 编辑器中创建一个 OPC DA 客户机
2. 在 Microsoft Excel 中组态对 WinCC 变量的访问

参见

如何组态在 Microsoft Excel 中访问 WinCC 变量 (页 540)

在 Microsoft Excel 中创建 OPC DA 客户机 (页 536)

在 Microsoft Excel 中创建 OPC DA 客户机

引言

要将 Microsoft Excel 用作 OPC DA 客户机，必须在 Microsoft Excel 的 Visual Basic 编辑器中创建特殊的脚本。

要求

具有基本的 Microsoft Excel Visual Basic 编辑器知识。

步骤

1. 打开 Microsoft Excel，将出现新的工作簿。
2. 在“Visual Basic 编辑器”的“工具”菜单中， → 单击“宏”。 将打开 Microsoft Excel 的 Visual Basic 编辑器。
3. 在“Visual Basic 编辑器”的“工具”菜单中选择“引用...”。 将显示“引用 - VBAProject”对话框。 在可用引用的列表中找到条目“Siemens OPC DAAutomation 2.0”。 选中相应的复选框。 单击“确定”。
4. 复制以下所显示的脚本。 该脚本仅在在线帮助中可用。
5. 双击 Visual Basic 编辑器项目窗口中的“Sheet1”，打开新的代码窗口。
6. 将脚本粘贴到代码窗口中。
7. 从“文件”菜单中选择“保存”。 从“文件”菜单中选择“关闭并返回到 Microsoft Excel”。

脚本

```
Option Explicit
Option Base 1

Const ServerName = "OPCServer.WinCC"

Dim WithEvents MyOPCServer As OpcServer
```

```
Dim WithEvents MyOPCGroup As OPCGroup
Dim MyOPCGroupColl As OPCGroups
Dim MyOPCItemColl As OPCItems
Dim MyOPCItems As OPCItems
Dim MyOPCItem As OPCItem

Dim ClientHandles(1) As Long
Dim ServerHandles() As Long
Dim Values(1) As Variant
Dim Errors() As Long
Dim ItemIDs(1) As String
Dim GroupName As String
Dim NodeName As String

'-----------------------------------------------------------------
-----
' Sub StartClient()
' □□□ □□□ OPC_server□□□□□□□□□
'-----------------------------------------------------------------
-----
Sub StartClient()
  ' □□□□□□□ ErrorHandler
  '----------- □□□□□□ ClientHandle □ GroupName
  ClientHandles(1) = 1
  GroupName = "MyGroup"
  '----------- □□□□"A1"□□ ItemID
  NodeName = Range("A1").Value
  ItemIDs(1) = Range("A2").Value
  '----------- □□ OPC-Server □□
```

```
  Set MyOPCServer = New OpcServer
  MyOPCServer.Connect ServerName, NodeName

  Set MyOPCGroupColl = MyOPCServer.OPCGroups
  '----------- □□□□□□□□□□□□□□□□
  MyOPCGroupColl.DefaultGroupIsActive = True
  '----------- □□□□□□□□□□□□
  Set MyOPCGroup = MyOPCGroupColl.Add(GroupName)

  Set MyOPCItemColl = MyOPCGroup.OPCItems
  '----------- □□□□□□□□□□ ServerHandle
  MyOPCItemColl.AddItems 1, ItemIDs, ClientHandles,
ServerHandles, Errors
  '----------- □□□□□□□□□□
  MyOPCGroup.IsSubscribed = True
  Exit Sub

ErrorHandler:
  MsgBox "Error: " & Err.Description, vbCritical, "ERROR"
End Sub

'-------------------------------------------------------------------
-----
' Sub StopClient()
' □□□ □□□□□□□□□□□□□□□
'-------------------------------------------------------------------
-----
Sub StopClient()
  '----------- □□□□□□□□□
```

```
  MyOPCGroupColl.RemoveAll
  '----------- □□□□□□□□□□□□□
  MyOPCServer.Disconnect
  Set MyOPCItemColl = Nothing
  Set MyOPCGroup = Nothing
  Set MyOPCGroupColl = Nothing
  Set MyOPCServer = Nothing
End Sub

'-------------------------------------------------------------------
-----
' Sub MyOPCGroup_DataChange()
' □□□ □□□□□□□□□□□□□□□□□□□□□□□
'-------------------------------------------------------------------
-----
'----------- □□□□□ OPC-DA Automation 2.1□□□□
Private Sub MyOPCGroup_DataChange(ByVal TransactionID As Long,
ByVal NumItems As Long, ClientHandles() As Long, ItemValues() As
Variant, Qualities() As Long, TimeStamps() As Date)
 '----------- □□□□□□□□□□□□□□□□□□
  Range("B2").Value = CStr(ItemValues(1))
  Range("C2").Value = Hex(Qualities(1))
  Range("D2").Value = CStr(TimeStamps(1))
End Sub

'-------------------------------------------------------------------
-----
' Sub worksheet_change()
' □□□ □□□□□□□□□□□□□□□□□□□□□□□□□
```

```
'-------------------------------------------------------------------
-----
Private Sub worksheet_change(ByVal Selection As Range)
  '----------- □□□□"B3"□□□□□□□
  If Selection <> Range("B3") Then Exit Sub
  Values(1) = Selection.Cells.Value
  '----------- □□□□□□□□□□□
  MyOPCGroup.SyncWrite 1, ServerHandles, Values, Errors
End Sub
```

参见

如何组态 WinCC OPC DA 服务器上的 WinCC 项目 (页 524)

如何组态在 Microsoft Excel 中访问 WinCC 变量

引言

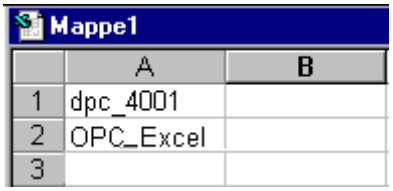
Excel OPC DA 客户机读取 WinCC OPC DA 服务器的 WinCC 变量并将变量值写入单元格中。 在 WinCC OPC DA 服务器的 WinCC 项目中，变量值显示在一个 I/O 域内。 如果单元格中的变量值改变，则在 WinCC OPC DA 服务器 I/O 域中的数值也随之改变。

要求

- 在 WinCC OPC DA 服务器的 WinCC 项目中，组态名称为“OPC_Excel”且数据类型为“有符号 16 位数”的内部变量。
- 将“OPC_Excel”变量的数值写入 WinCC OPC DA 服务器的 WinCC 项目的 I/O 域内。
- 启用 WinCC OPC DA 服务器的 WinCC 项目。

步骤

1. 在 Microsoft Excel 的单元格 A1 中，输入用作 OPC 服务器的计算机名称。 在单元格 A2 中，输入变量名称“OPC_Excel”。



2. 在 Excel 的“工具”菜单中，选择“宏” → “宏”。“宏”对话框将打开。 从宏列表中选择条目“Sheet1.StartClient”。 单击“运行”以启动 OPC 客户机。
3. 变量的值会写入单元格 B2 中，质量代码写入 C2 中，时间标志写入 D2 中。
4. 在单元格 B3 中输入新的数值。 更改后的数值显示在 WinCC OPC 服务器上的 I/O 域内。
5. 在 Excel 的“工具”菜单中，选择“宏” → “宏...”。“宏”对话框将打开。 从宏列表中选择条目“Sheet1.StopClient”。 单击“运行”停止 OPC 客户机。

4.8 WinCC OPC HDA 服务器

4.8.1 WinCC OPC HDA 服务器的功能

引言

WinCC OPC HDA 服务器是 DCOM 应用程序，为 OPC HDA 客户机提供来自归档系统的必需数据。 使用条目句柄访问数据。 可以进行读访问或写访问。 还可以分析数据。

WinCC OPC HDA 服务器支持 OPC 历史数据访问 1.20 规范。 这已经由适应性测试确认。

下列章节说明数据结构的设计，以及 WinCC OPC HDA 服务器支持的属性、集合和功能。 这不是详细的说明，而是最重要信息的摘要。 有关更多信息，可参见“OPC 历史数据访问 1.20”规范。

安装

在安装 WinCC 期间，可以选择 WinCC OPC HDA 服务器。 可以选择对 WinCC 归档系统的访问是否具有写功能。 安装后，WinCC OPC DA 服务器无需进行组态，立即可用。

如果安装时未选择写访问，则只能读取和分析 WinCC 归档系统中的数据。 如果选择了写访问，可以分析、添加、删除和更新 WinCC 归档系统中的数据。

可以在 WinCC 服务器或 WinCC 客户机上安装 WinCC OPC HDA 服务器。

许可证

为了操作 WinCC OPC HDA 服务器，必须在每台作为 OPC HDA 服务器的 WinCC 计算机上安装下列许可证：

- WinCC 基本系统
- WinCC 选件数据连通性软件包（Connectivity Pack）

OPC HDA 客户机

所有符合 OPC 历史数据访问 1.20 规范的 OPC HDA 客户机都可以访问 WinCC OPC HDA 服务器。 也可以使用专有的 OPC HDA 客户机。 通过创建专有 OPC HDA 客户机，可以满足大多数用户特定需求。

如何使用 OPC HDA 客户机的实例包括：

- 分析和判断归档数据
- 从不同的 OPC HDA 服务器对归档进行统计过程控制

要使用 OPC HDA 客户机请求历史数值，需要在组态过程中注意以下几点：

- 选择的查询周期应能使客户机在发送下一次查询之前收到所请求的数据。 周期太短可能造成接收数据时时间延迟过长。
- WinCC 服务器的 CPU 负载取决于每次查询的变量数。

使用组态的交换写访问循环归档

在运行系统中，数据在 WinCC 服务器的循环归档中进行修改。

只有在创建数据后立即进行更改时才会将更改传送给换出归档。

如果循环归档的相关归档段已经换出，则随后不会在换出归档中完成更改。 在 WinCC 服务器上删除了归档段，修改的数据也将删除。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

兼容性 (页 511)

质量代码 (页 548)

WinCC OPC HDA 服务器的数据结构 (页 543)

4.8.2 WinCC OPC HDA 服务器的数据结构

4.8.2.1 WinCC OPC HDA 服务器的数据结构

引言

WinCC OPC HDA 服务器上的数据是结构化的。 以下列出了可用的数据结构。这不是详细的说明，而是最重要信息的摘要。 有关更多信息，可参见“OPC 历史数据访问 1.20”规范。

数据结构

	描述
属性	为原始数据提供附加的质量特性。 属性包括关于归档的数据类型、规范等。有关更多信息，请参见所支持属性的概述。
配件	指定时间间隔中原始数据的摘要。 集合包括平均值、最小值和最大值等。有关更多信息，请参见所支持集合的概述。
开始时间/结束时间	设置时间间隔的开始点和结束点。
范围值	在开始和结束时记录的值。 如果没有可用的范围值，靠近这些时间的数值将用作范围值。
原始数据	WinCC 归档系统中特定时间间隔的数据。 这些数据包含时间标志和质量等级。
条目句柄	唯一指定给 WinCC 归档变量。
条目标识号	WinCC 归档变量的唯一标识符。 条目标识号可用于获取条目句柄。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

受支持函数的概述 (页 546)

WinCC OPC HDA 服务器的时间格式 (页 546)

受支持属性的概述 (页 544)

受支持配件的概述 (页 544)

4.8.2.2 受支持属性的概述

引言

下表包括了 WinCC OPC HDA 服务器支持的属性。 有关更多信息，可参见“OPC 历史数据访问 1.20”规范。

属性

属性	属性标识号	描述
条目标识号	OPCHDA_ITEMID	指示要访问的 WinCC 归档变量。
条目数据类型	OPCHDA_DATA_TYPE	指示 WinCC 归档变量的数据类型。
描述	OPCHDA_DESCRIPTION	返回 WinCC 归档变量的说明。 该说明在 WinCC 变量记录中定义。
工程单位	OPCHDA_ENG_UNITS	设置度量单位的显示。 设定标签在 WinCC 变量记录中定义。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

WinCC OPC HDA 服务器的数据结构 (页 543)

4.8.2.3 受支持配件的概述

引言

下表列出了 WinCC OPC HDA 服务器支持的集合。 有关更多信息，可参见“OPC 历史数据访问 1.20”规范。

配件

配件	描述
OPCHDA_COUNT	返回指定时间间隔中原始数据的计数。
OPCHDA_START	返回时间间隔开始时原始数据的初始值。
OPCHDA_END	返回时间间隔结束时原始数据的最终值。

配件	描述
OPCHDA_AVERAGE	返回指定时间间隔中原始数据的平均值。
OPCHDA_TIMEAVERAGE	返回指定时间间隔中原始数据按时间的平均值。
OPCHDA_TOTAL	返回指定时间间隔中的总和值。
OPCHDA_STDEV	返回指定时间间隔中原始数据的标准偏差。
OPCHDA_MINIMUMACTUALTIME	返回指定时间间隔中原始数据的最小值及其时间标志。
OPCHDA_MINIMUM	返回指定间隔中原始数据的最小值。
OPCHDA_MAXIMUMACTUALTIME	返回指定时间间隔中原始数据的最大值及其时间标志。
OPCHDA_MAXIMUM	返回指定间隔中原始数据的最大值。
OPCHDA_DELTA	返回指定时间间隔中原始数据中第一个值和最后一个值的差。
OPCHDA_REGSLOPE	返回指定时间间隔中原始数据回归线的斜率。
OPCHDA_REGCONST	返回起始点处原始数据的回归值。
OPCHDA_REGDEV	返回指定时间间隔内原始数据回归的标准偏离。
OPCHDA_VARIANCE	返回指定时间间隔内原始数据的变化。
OPCHDA_RANGE	返回指定时间间隔内原始数据的 OPCHDA_MAXIMUM 和 OPCHDA_MINIMUM 的差。
OPCHDA_DURATIONGOOD	返回原始数据质量好的时间段。 时间段以秒为单位。
OPCHDA_DURATIONBAD	返回原始数据质量差的时间段。 时间段以秒为单位。
OPCHDA_PERCENTGOOD	返回质量好的原始数据的百分比。
OPCHDA_PERCENTBAD	返回质量差的原始数据的百分比。
OPCHDA_WORSTQUALITY	返回指定时间间隔内原始数据的最差质量。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

WinCC OPC HDA 服务器的数据结构 (页 543)

WinCC OPC HDA 服务器的功能 (页 541)

4.8.2.4 受支持函数的概述

引言

下表列出了 WinCC OPC HDA 服务器支持的函数。 OPC HDA 客户机可使用这些函数进行数据交换。 有关更多信息，可参见“OPC 历史数据访问 1.20”规范。

读

函数	描述
ReadRaw	返回指定时间间隔内的原始数据及其质量和时间标志。
ReadProcessed	返回计算出的数值、数值质量和指定时间间隔的时间标志。 计算出的数值取决于所选的集合。
ReadAtTime	返回特定时间间隔内的原始数据及其质量和时间标志。 如果没有数值可用，将插入该点的数值。
ReadAttribute	返回指定时间间隔的条目属性和时间标志。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

WinCC OPC HDA 服务器的功能 (页 541)

4.8.2.5 WinCC OPC HDA 服务器的时间格式

引言

在 WinCC OPC HDA 服务器上通过设置开始时间和结束时间指定时间间隔。 指定的时间间隔确定历史数据的观察期。 在指定时间时，必须保持特定的格式。

下列选项可用于指定时间格式：

- 基于 UTC 的绝对时间
- 相对于服务器当地时间的相对时间

根据 UTC 的绝对值

缺省状态下，WinCC OPC HDA 服务器使用相同的世界时间（UTC）作为其时间基准。 该时间对应于格林威治标准时间（即中欧时间减去一小时）。

时间格式

YYYY/MM/DD hh:mm:ss.msmsms

参数

YYYY = 年份

MM = 月份

DD = 日

hh = 小时

mm = 分钟

ss = 秒

ms = 毫秒

输入实例

2002/06/10 09:27:30.000

指定相对于当地时间的时间

对于该选项，输入的时间相对于服务器的当地时间。 当地时区在计算机的“日期/时间”控制面板中设置。

时间格式

关键字+/-误差 1 +/-误差(n)

误差是与服务器当地时间的差。

关键字

NOW = 服务器上当前的当地时间

SECOND = 当前秒钟

MINUTE = 当前分钟

HOUR = 当前小时

DAY = 当前日

WEEK = 当前星期

MONTH = 当前月份

YEAR = 当前年份

误差

+/-S = 秒钟差

+/-M = 分钟差

+/-H = 小时差

+/-D = 日差

+/-W = 星期差

+/-MO = 月份差

+/-Y = 年份差

实例：

DAY - 1D = 前一日

DAY-1D + 7H30 = 前一日的 7:30

MO-1D+5H = 上个月最后一天的 5:00

NOW-1H15M = 1 小时 15 分钟前

YEAR+3MO= 本年度的四月

参见

www.opcfoundation.org (http://www.opcfoundation.org)

WinCC OPC HDA 服务器的功能 (页 541)

4.8.3 质量代码

引言

质量代码用来判断原始数据的状态和质量。 下表显示了 OPC HDA 的质量代码。 有关更多信息，可参见“OPC 历史数据访问 1.20”规范。

WinCC OPC HDA 服务器的质量代码

代码	OPC	描述	质量
0x00040000	OPCHDA_RAW	指示原始数据传输的质量。	好 劣 不确定
0x00080000	OPCHDA_CALCULATED	指示计算出的数据传输的质量。	好 劣 不确定
0x00100000	OPCHDA_NOBOUND	在开始点或结束点处未发现范围值。	劣
0x00200000	OPCHDA_NODATA	指定时间间隔内没有发现任何原始数据。	劣
0x00400000	OPCHDA_DATALOST	所选间隔内的原始数据未完全归档。	劣

参见

www.opcfoundation.org (http://www.opcfoundation.org)

4.8.4 支持的写访问

引言

下表列出了 WinCC OPC HDA 服务器支持的写访问。

表格元素：

	描述
周期性归档	要归档的过程值存储在循环归档中。 循环归档由数目可组态的数据缓冲区组成。 定义数据缓冲区的大小和时间段（例如以天为单位）。 当所有数据缓冲区都满时，第一个数据缓冲区中的过程数据被覆盖。
交换之后循环归档	为了使数据缓冲区中的过程数据不被覆盖，可以将其先交换出来（导出）。

	描述
☺	受 WinCC 支持。
☹	不受 WinCC 支持。

写访问

以后添加过程值

周期性归档	交换之后循环归档	受 WinCC 支持	描述
是	否	☺	如果时间段包含在循环归档中，以后可以添加过程值。
是	是	☹	相应时间段的数据缓冲区被交换至归档备份。 以后，不能将过程值添加到归档备份。
否	否	☹	循环归档不可用。不能存储过程值。
不可以	可以	☹	循环归档不可用。 不能存储过程值。

在运行系统中添加过程值

周期性归档	交换之后循环归档	受 WinCC 支持	描述
是	否	☺	过程值被添加到循环归档当前有效的数据缓冲区中。

插入将来的过程值

周期性归档	交换之后循环归档	受 WinCC 支持	描述
是	否	☹	在写访问期间，在未来不能添加值。
否	否	☹	写访问时，在未来不能添加值。

删除过程值

周期性归档	交换之后循环归档	受 WinCC 支持	描述
是	否	☺	时间段包含在循环归档中时，可以删除过程值。
是	是	☹	相应时间段的数据缓冲区被交换至归档备份。 可以从归档备份删除过程值。
否	否	☹	循环归档不可用。不能存储过程值。
不可以	可以	☹	循环归档不可用。 不能存储过程值。

编辑过程值

周期性归档	交换之后循环归档	受 WinCC 支持	描述
是	否	☺	时间段包含在循环归档中时，可以编辑过程值。
是	是	☹	相应时间段的数据缓冲区被交换至归档备份。 可以在归档备份中编辑过程值。

周期性归档	交换之后循环归档	受 WinCC 支持	描述
否	否	☹	循环归档不可用。不能存储过程值。
不可以	可以	☹	循环归档不可用。不能存储过程值。

4.8.5 OPC HDA 连接实例

4.8.5.1 OPC HDA 连接实例

引言

在下面的实例中，将组态 WinCC 和 OPC HDA 客户机之间的连接。 来自 WinCC 归档系统的数据通过 WinCC OPC HDA 服务器变得可用。 OPC HDA 客户机通过条目句柄访问数据。 要简化组态过程，将使用 OPC HDA 浏览器。

将使用来自 OPC 基金会的 OPC HDA 客户机。 所有符合 OPC 历史数据访问 1.20 规范的 OPC HDA 客户机都可以访问 WinCC OPC HDA 服务器。

要求

- 在 WinCC OPC HDA 服务器的 WinCC 项目中，创建名称为“OPC_HDA”、数据类型为“无符号 16 位数”的内部变量。
- 创建在 WinCC 归档系统中称为“HDA_ProcessValueArchive”的过程值归档。
- 在过程值归档“HDA_ProcessValueArchive”中创建称为“OPC_HDA_Tag”的 WinCC 归档变量。 将 WinCC 归档变量与内部变量“OPC_HDA”链接。
- 在运行系统列表中，启动变量记录运行系统，禁用图形运行系统。
- 启动 WinCC OPC HDA 服务器的 WinCC 项目。

组态步骤

将 WinCC 连接到 OPC HDA 客户机需要以下组态：

1. 使用 HDA 服务器浏览器组态对 WinCC 归档变量的访问
2. 读取 WinCC 归档变量的数值

参见

如何使用 HDA 服务器浏览器来组态对 WinCC 归档变量的访问 (页 555)

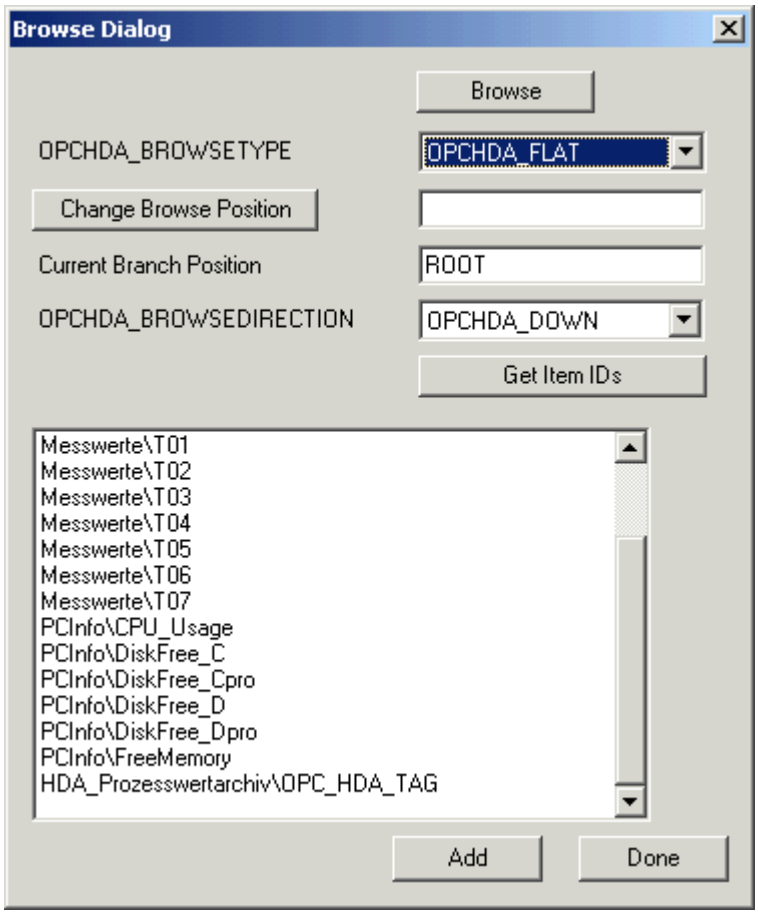
HDA 服务器浏览器 (页 554)

读取 WinCC 归档变量的数值 (页 556)

4.8.5.2 HDA 服务器浏览器

引言

OPC HDA 客户机通过条目句柄访问变量值。 为了易于组态，WinCC OPC HDA 服务器支持浏览器功能。 OPC HDA 客户机可以使用 HDA 服务器浏览器来搜索 WinCC OPC HDA 服务器的地址空间。 数据以过程值归档分级列出。



说明

如果不通过 HDA 服务器浏览器来访问 WinCC 归档变量，就需要手动组态条目标识号。 寻址 WinCC 归档变量时，计算机名称（服务器前缀）包含在路径中。 条目标识号具有下列语法： Server-prefix::process_value_archive\WinCC_archive_tag。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

如何使用 HDA 服务器浏览器来组态对 WinCC 归档变量的访问 (页 555)

4.8.5.3 如何使用 HDA 服务器浏览器来组态对 WinCC 归档变量的访问

简介

在本部分中，OPC HDA 客户机将用于访问 WinCC 归档变量。将使用来自 OPC 基金会的 OPC HDA 客户机。HDA 服务器浏览器将用于组态访问。

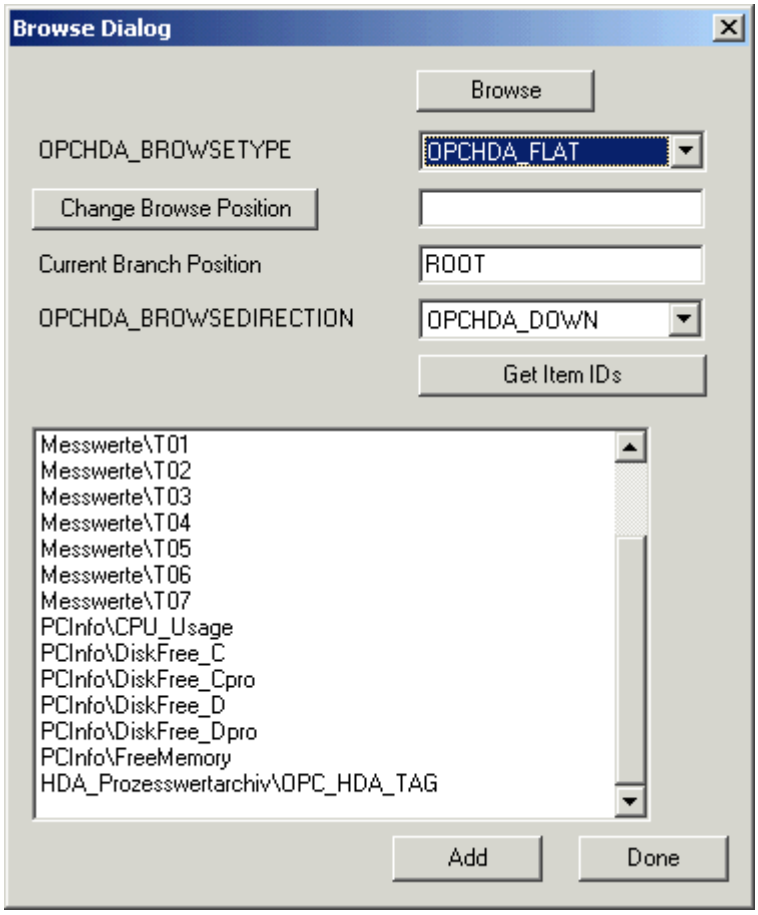
说明

此处的 OPC HDA 客户机是来自 OPC 基金会的演示客户机。源代码可以参见 Internet 网址 http://www.opcfoundation.org。

步骤

1. 将文件“SampleClientHDA.exe”复制到所选的目录中。只有在线帮助中才提供了此应用程序。
2. 双击“SampleClientHDA.exe”文件。HDA 客户机程序将启动。
3. 在“服务器名称”区域中，选择条目“OPCServerHDA.WinCC.1”。单击“连接”。确认随后的对话框。

4. 在 HDA 客户机中单击“浏览”。将打开“浏览对话框”对话框。在“OPCHDA_BROWSETYPE”域中选择“OPCHDA_FLAT”。



5. 在选择窗口中，选择“HDA_ProcessValueArchive_HDA_TAG”条目。单击“添加”，然后单击“完成”关闭对话框。

有关详细信息，请访问 http://www.opcfoundation.org。

参见

读取 WinCC 归档变量的数值 (页 556)

www.opcfoundation.org (http://www.opcfoundation.org)

4.8.5.4 读取 WinCC 归档变量的数值

引言

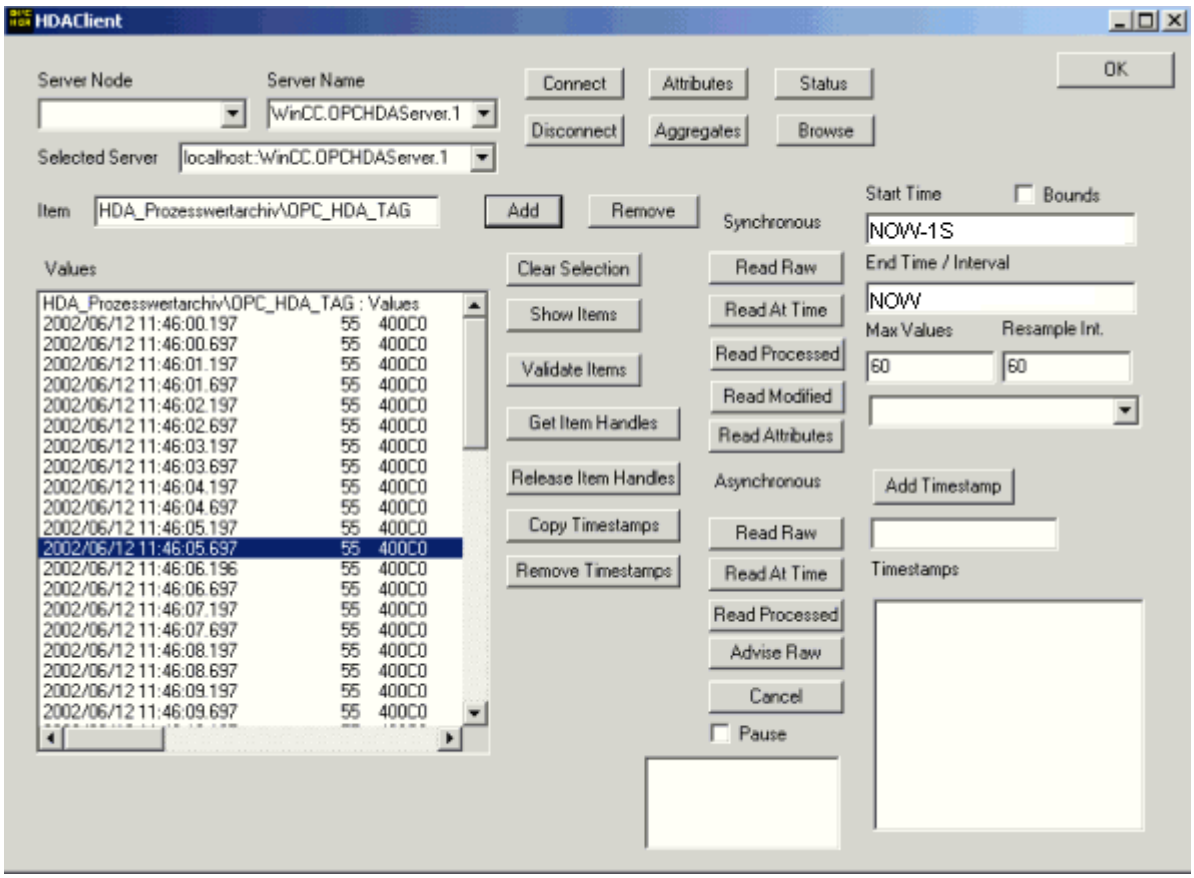
本节说明如何访问和读取 WinCC 归档变量。

要求

- OPC HDA 客户机必须在运行。

步骤

1. 在 HDA 客户机中单击“Show Items”（显示条目）。
2. 在 HDA 客户机中单击“Get Item Handles”（获取条目句柄）。
3. 双击“Value”（数值）选择域中的“HDA_ProcessValueArchive_HDA_Tag”。
4. 在“Start Time”（开始时间）域中输入“NOW-10S”。 在“End Time”（结束时间）域中输入“NOW”。



5. 单击“Read Raw”（读取原始数据）。 数值及其质量代码和时间标志显示在“Values”（数值）选择域中。

4.8.6 WinCC 中 OPC HDA 服务器针对非周期性记录的特殊功能

导言

在 WinCC 中，会周期性或非周期性地执行变量记录。 WinCC OPC HDA 服务器会视变量记录方法而定以不同的方式工作：

- 对于所有周期性记录的值，OPC HDA 服务器都会遵照 OPC Foundation 的 HDA 规范运行。 OPC 总计将进行线性内插。
- OPC Foundation 的 HDA 规范中不包括非周期性记录的变量。 OPC 总计将进行增量式内插。 尤其是如果变量长时间未经历变化，某个时间段内将没有可用的数据。 应对以下事项加以考虑，从而在出现上述情况时仍可获得有效的数据。

注意
对于非周期性记录的变量，OPC HDA 服务器不遵从 OPC 的规定。 OPC Foundation 的 HDA 规范不认可非周期性记录的变量，因此任何归档服务器都无法处理非周期性记录的变量。 所支持的总计将依照 OPC HDA 规范进行计算。 不支持任何非显式调用的函数。

组态非周期性记录的变量

为组态非周期性记录的变量，需要为变量启用“进行段更改后归档”(Archive after segment change) 设置。 段发生变化时，此操作会在新记录中输入最新的有效值。

对于非周期性记录的变量 WinCC OPC HDA 服务器所支持的总计

OPC HDA 服务器支持以下总计：

- OPCHDA_MINIMUM
- OPCHDA_MAXIMUM
- OPCHDA_AVERAGE
- OPCHDA_END
- OPCHDA_INTERPOLATIVE
- OPCHDA_TIMEAVERAGE
- OPCHDA_TOTAL
- OPCHDA_DURATIONGOOD
- OPCHDA_PERCENTGOOD

对于非周期性记录的变量 WinCC OPC HDA 服务器所支持的函数

- 仅限有“约束”的 ReadRaw。 针对变量的 ReadRaw 在执行时必须始终施加“约束”，以找到无记录值变化区域的最后一个实际存储值。
- ReadProcessed
- DeleteRaw
- DeleteAtTime
- Insert
- InsertReplace
- Replace

计算非周期性记录变量的总计

总计的计算基于扩展的“RawData”数据记录，该记录除包含实际存储值外，还包含用于计算的虚拟数据点。 WinCC OPC HDA 服务器会根据“ReadProcessed”的要求准备包含的“RawData”。 计算所需的虚拟数据点由边界实际数据点组成。 虚拟数据点包括以下重要点：

- “StartTime”的值
- “EndTime”的值
- 间隔限制的值

实例

“00:59:00”、“01:02:00”和“01:03:00”的值是为非周期变量记录变量存储的。 OPC HDA 客户机通过“ReadProcessed”要求具有以下参数的总计：

- StartTime = 01:00:00
- EndTime = 01:04:00
- Interval = 00:02:00

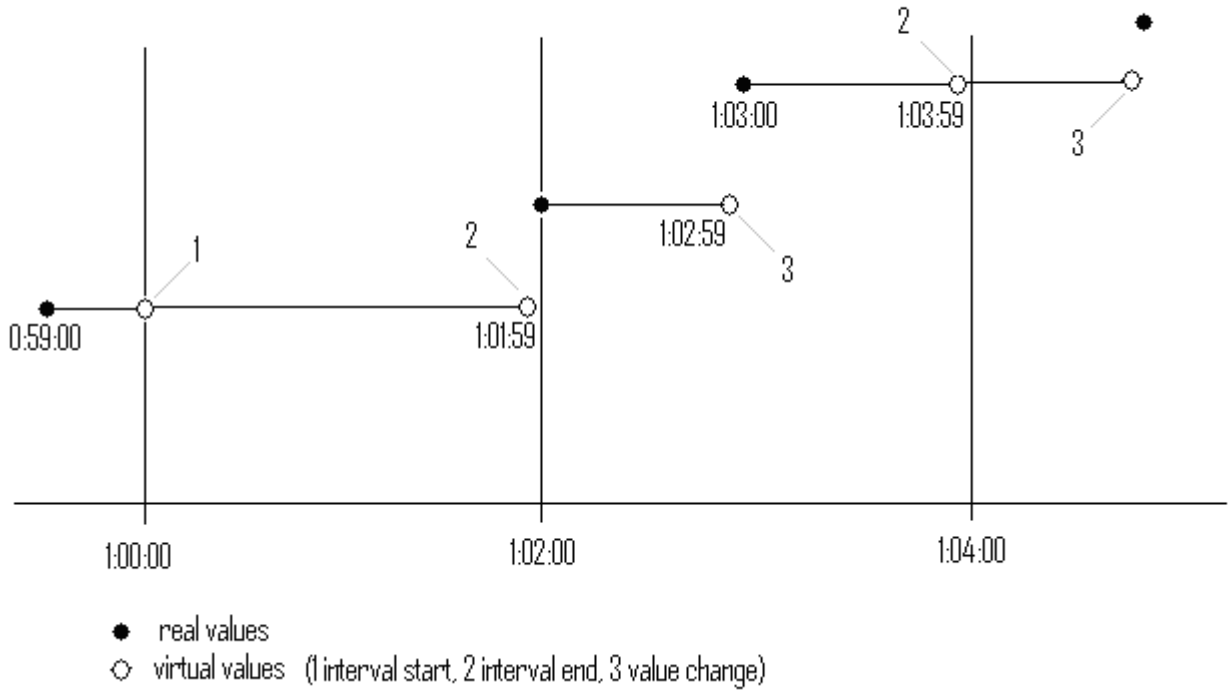
说明

在限制（“EndTime”/“Interval”）处生成虚拟值时，时间段长度始终比限制处计算的时间戳短 1 µs。

下表中使用了 1 秒的增量，以提高总览效果。 下图图解了该实例。

OPC 服务器使用下面的“RawData”来计算总计：

编号	时间戳	实际存储值	生成的虚拟值
1	00:59:00	1.00	
2	01:00:00		1.00
3	01:01:59		1.00
4	01:02:00	2.00	
5	01:02:59		2.00
6	01:03:00	3.00	
7	01:03:59		3.00



4.9 WinCC OPC A&E 服务器

4.9.1 WinCC OPC A&E 服务器的功能

引言

WinCC OPC A&E 服务器是 DCOM 应用程序。 OPC A&E 客户机通过订阅随时了解 WINCC 消息的状态变化情况。 OPC A&E 客户机可以对订阅使用过滤器。 该过滤器确定显示哪些消息和属性。

WinCC OPC A&E 服务器支持规范 OPC Alarm&Event 1.10。 这已经由适应性测试确认。

下列章节说明 WinCC 消息系统在 OPC A&E 上的显示，以及 WinCC OPC A&E 服务器支持的属性。这不是详细的说明，而是最重要信息的摘要。 有关更多信息，可参见“OPC 报警 & 事件 1.0”规范。

安装

可以在安装 WinCC 期间选择 WinCC OPC A&E 服务器。 安装后，WinCC OPC A&E 服务器无需其他组态便立即可用。

从 WinCC V6.2 开始，可在 WinCC 服务器和 WinCC 客户机上安装 WinCC OPC A&E 服务器。

许可证

为了运行 WinCC OPC A&E 服务器，必须在每台作为 OPC A&E 服务器的 WinCC 服务器上安装下列许可证：

- WinCC 基本系统
- WinCC 选件数据连通性软件包（Connectivity Pack）

服务器类型

WinCC OPC A&E 服务器支持条件事件和简单事件。 此外，还支持跟踪事件。

与条件相关的事件服务器

对于条件相关的事件服务器，事件与某个条件相关。 例如，条件可能是超出变量边界值。一旦超出边界值，便会在 WinCC 中生成消息。 此消息在 OPC A&E 中显示为报警。

简单事件服务器

简单事件是提示 OPC A&E 客户机关于事件的消息。 简单事件包括启动和退出程序。

说明

在使用冗余系统时，请注意如下事项：
比较变量时，切换到内部变量的简单事件会发送两次。
第一个消息由主机服务器触发，第二个消息由备用服务器触发。

跟踪事件服务器

如果在过程中发生了变化，OPC A&E 客户机会接收到消息。 例如，这一变化可能是调节器调整。

OPC A&E 客户机

所有符合 OPC Alarms & Events 1.10 规范的 OPC A&E 客户机都可以访问 WinCC OPC A&E 服务器。 OPC A&E 客户机可以是专有客户机。 通过创建专有 OPC 客户机，可以满足大多数用户特定需求。 例如，OPC A&E 客户机可用于对来自多重 OPC A&E 服务器的报警进行分析和一般归档。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

兼容性 (页 511)

OPC A&E 质量代码 (页 570)

在 OPC A&E 上映射 WinCC 消息系统 (页 563)

4.9.2 在 OPC A&E 上映射 WinCC 消息系统

4.9.2.1 在 OPC A&E 上映射 WinCC 消息系统

导言

组态 WinCC 消息系统期间，将进行设置以确定哪些过程事件产生消息。 此消息在 OPC A&E 中显示为报警。下面的表格列出了报警最重要的参数。 还描述了如何通过 WinCC 消息系统使信息可用。 更多信息，请参阅“报警结构”。

概述

OPC	WinCC 消息系统
Source	指示消息源。 消息源的格式为“<服务器前缀>::@LOCALMACHINE::”。
Time	发布已接收、已发送和已确认消息的时间标志。 发布 UTC（协调世界时）格式的时间标志。
Type	指示是简单事件、跟踪事件还是条件相关事件。 WinCC - POC A&E 服务器支持简单事件、条件相关事件和跟踪事件。
Severity	指示 WinCC 消息的优先级。
EventCategory	返回消息类别。 有关该主题的更多信息，请参阅“显示消息等级和类型”。
Message	指示相应消息号的消息文本。
ConditionName	指示消息号。
ChangeMask	指示消息改变后的状态。 更多信息，请参阅“确认方法”。
NewState	返回消息状态。 更多信息，请参阅“确认方法”。
ConditionQuality	返回消息质量。 更多信息，请参阅“质量代码”。
AckRequired	指示消息是否需要确认(接收)。
ActiveTime	返回已接收消息的时间标志。
EventAttribute	列出相关消息所需的属性。 更多信息，请参阅“WinCC 消息系统的属性”。
Quality	返回消息的质量代码。
Cookie	从 OPC A&E 服务器返回 cookie。 cookie 对应于 WinCC 报警系统中的消息号

参见

确认方法 (页 567)

WinCC 消息系统的属性 (页 565)

映射 WinCC 消息类别和消息类型 (页 564)

4.9.2.2 映射 WinCC 消息类别和消息类型

引言

WinCC 消息系统向用户提示过程中的使用干扰和操作情况。 WinCC 消息始终属于与事件类别有关的特定消息等级和消息类型。

通过 CcAeProvider.ini 文件组态 OPC 上 WinCC 消息系统的映射。

事件类别

在 WinCC OPC A&E 服务器上，为每种消息等级和类型的组合创建一个事件类别。

事件类别由类别 ID 和描述性的“类别说明”确定。 类别 ID 由消息等级和消息类型的 WinCC 内部 ID 组成；类别说明由消息等级和消息类型组成。

注意
如果 OPC A&E 服务器在连接站的 WinCC 客户机上运行，则链接到其上的 OS 服务器必须具有相同的消息等级和消息类型组态。 如果不是这样，则输入的 OPC 客户机将直接访问 OS 服务器。

可以通过报警属性 CLASSNAME 和 TYPENAME 确定消息等级和消息类型的名称。

4.9.2.3 映射 WinCC 消息优先级

引言

通过 OPC 服务器的属性“Severity”来显示 WinCC 消息的优先级。

在 WinCC 消息系统中组态报警时，可以组态一个在 0 到 16 之间的优先级。OPC A&E 规范定义了 severity 的数值范围为 1 到 1000，其中 1 代表最低 severity，1000 代表最高 severity。

因此，WinCC 优先级的值相应显示为 OPC severity。 在标准映射中，WinCC 优先级 0 将映射为 OPC severity 1。所有其它的优先级值都将以线性方式插入一直到 severity 1000。可在 CcAeProvider.ini 文件中组态其它优先级映射规则。

4.9.2.4 WinCC 消息系统的属性

简介

下表列出了 WinCC 消息系统的 OPC 属性。这些属性在 WinCC 消息系统中组态。某些属性仅供 WinCC 内部使用，与 OPC A&E 客户机无关。因此未列出这些属性。

属性

OPC 属性	WinCC 消息系统	数据类型
CLASSNAME	返回消息类别名称。	VT_BSTR
TYPENAME	返回消息类型名称。	VT_BSTR
FORECOLOR	返回用于显示已接收、已发送和已确认消息的文本颜色。	VT_I4
BACKCOLOR	返回用于显示已接收、已发送和已确认消息的背景色。	VT_I4
FLASHCOLOR	返回闪烁颜色。	VT_I4
FLAGS	指示消息是否需要确认（回执）。	VT_I4
TEXT01	返回用户文本块 01 的内容。	VT_BSTR
TEXT02	返回用户文本块 02 的内容。	VT_BSTR
TEXT03	返回用户文本块 03 的内容。	VT_BSTR
TEXT04	返回用户文本块 04 的内容。	VT_BSTR
TEXT05	返回用户文本块 05 的内容。	VT_BSTR
TEXT06	返回用户文本块 06 的内容。	VT_BSTR
TEXT07	返回用户文本块 07 的内容。	VT_BSTR
TEXT08	返回用户文本块 08 的内容。	VT_BSTR
TEXT09	返回用户文本块 09 的内容。	VT_BSTR
TEXT10	返回用户文本块 10 的内容。	VT_BSTR
PROCESSVALUE01	返回过程值块 01 的内容。	VT_VARIANT

OPC 属性	WinCC 消息系统	数据类型
PROCESSVALUE02	返回过程值块 02 的内容。	VT_VARIANT
PROCESSVALUE03	返回过程值块 03 的内容。	VT_VARIANT
PROCESSVALUE04	返回过程值块 04 的内容。	VT_VARIANT
PROCESSVALUE05	返回过程值块 05 的内容。	VT_VARIANT
PROCESSVALUE06	返回过程值块 06 的内容。	VT_VARIANT
PROCESSVALUE07	返回过程值块 07 的内容。	VT_VARIANT
PROCESSVALUE08	返回过程值块 08 的内容。	VT_VARIANT
PROCESSVALUE09	返回过程值块 09 的内容。	VT_VARIANT
PROCESSVALUE10	返回过程值块 10 的内容。	VT_VARIANT
STATETEXT	返回状态消息。	VT_BSTR
INFOTEXT	返回消息的信息文本。	VT_BSTR
LOOPINALARM	指示是否组态报警回路。	VT_I4
CLASSID	返回消息类别 ID。	VT_I4
TYPEID	返回消息类型 ID。	VT_I4
MODIFYSTATE	输出消息的状态变量值。	VT_I4
AGNR	返回产生消息的自动化设备的编号。	VT_I2
CPUNR	返回产生消息的 CPU 的编号。	VT_I2
DURATION	指示消息接收、发送和确认的时间间隔。	VT_I4
COUNTER	输出自运行系统启动以来的消息数。	VT_I4
QUITSTATETEXT	指示消息是否被确认。	VT_BSTR
QUITCOUNT	输出未确认的活动消息数。	VT_I4

OPC 属性	WinCC 消息系统	数据类型
PARAMETER	输出消息参数。（消息组态画面）	VT_BSTR
BLOCKINFO	返回消息块的当前内容。	VT_BSTR
ALARMCOUNT	输出未决消息数。	VT_I4
LOCKCOUNT	输出锁定消息数。	VT_I4
PRIORITY	指示已组态的消息优先级。	VT_I4
APPLICATION	输出触发了消息的应用程序。	VT_BSTR
COMPUTER	输出处理了消息的计算机的名称。	VT_BSTR
USER	输出处理了消息的用户的名称。	VT_BSTR
COMMENT	输出消息注释。	VT_BSTR

4.9.2.5 确认方法

引言

对于 WinCC，确认方法是如何显示和处理消息从“进入”到“离开”。 在 WinCC OPC A&E 服务器上，该消息状态用参数“ChangeMask”和“NewState”管理。

条件、简单和跟踪事件

通常将来自 WinCC 系统的消息作为条件事件发送到客户机。 为了使消息作为简单事件处理，在组态消息等级时必须满足下列条件：

- “确认进入”未激活。
- “无离开状态的消息”激活。

根据映射组态，消息等级“没有确认的系统”和消息类型“操作消息”的消息将作为 OPC 跟踪事件传送。

ChangeMask

“ChangeMask”参数将记录消息状态改变的地址。

参数值：

- OPC_CHANGE_ACTIVE_STATE
- OPC_CHANGE_ENABLE_STATE
- OPC_CHANGE_ACK_STATE

NewState

“NewState”参数指示改变后的消息状态。

参数值：

- OPC_CONDITION_ACTIVE
- OPC_CONDITION_ENABLED
- OPC_CONDITION_ACKED

概述

WinCC	NewState	ChangeState
已接收的消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE
带回执的已发送消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE
不带回执的已发送消息	OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE

WinCC	NewState	ChangeState
已确认的消息（消息未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
已确认的消息(消息不再未决)	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
锁定的消息	----------	OPC_CHANGE_ENABLED_STATE
解锁的消息	OPC_CONDITION_ENABLED	OPC_CHANGE_ENABLED_STATE
已接收的、确认的消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE
已接收的、带回执的已发送消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
已接收的、不带回执的已发送消息	OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE

WinCC	NewState	ChangeState
由系统确认的消息（消息未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
由系统确认的消息（消息不再未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
紧急确认的消息（消息未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
紧急确认的消息（消息不再未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE

参见

www.opcfoundation.org (http://www.opcfoundation.org)

4.9.3 OPC A&E 质量代码

引言

质量代码用于判断消息的状态和质量。 下表列出了 OPC A&E 的质量代码。

质量代码

代码	质量	状态
0xC0	OPC_GOOD	确定
0x40	OPC_UNCERTAIN	在不确定的情况下返回，例如如果发生延迟确认（回执）。
0x00	OPC_BAD	在与源的连接中断时返回。

4.9.4 OPC A&E 连接实例

4.9.4.1 OPC A&E 连接实例

简介

在下面的实例中，将组态 WinCC 和 OPC A&E 客户机之间的连接。 WinCC OPC A&E 服务器使来自 WinCC 消息系统的数据变得可用。

OPC A&E 客户机通过订阅来获知 WinCC 消息的状态改变。

所有符合 OPC Alarms&Events 1.1 规范的 OPC A&E 客户机都可以访问 WinCC OPC A&E 服务器。

组态步骤

WinCC 和 OPC A&E 客户机之间的连接需要下列组态：

1. 组态对 WinCC 消息系统的访问

参见

www.opcfoundation.org (http://www.opcfoundation.org)

如何组态访问 WinCC 消息系统 (页 572)

4.9.4.2 如何组态访问 WinCC 消息系统

引言

本节中，OPC 基金会的 OPC A&E 客户机将访问 WinCC 消息系统。

说明

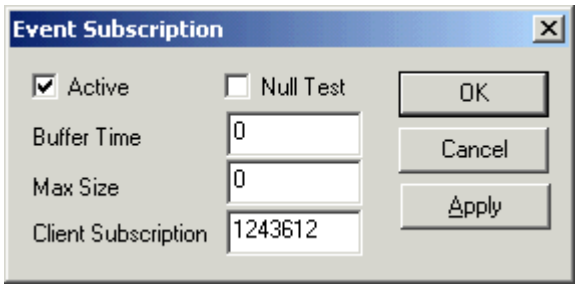
此处描述的 OPC A&E 客户机是来自 OPC 基金会的演示客户机。 源代码可以参见 Internet 网址 http://www.opcfoundation.org。

要求

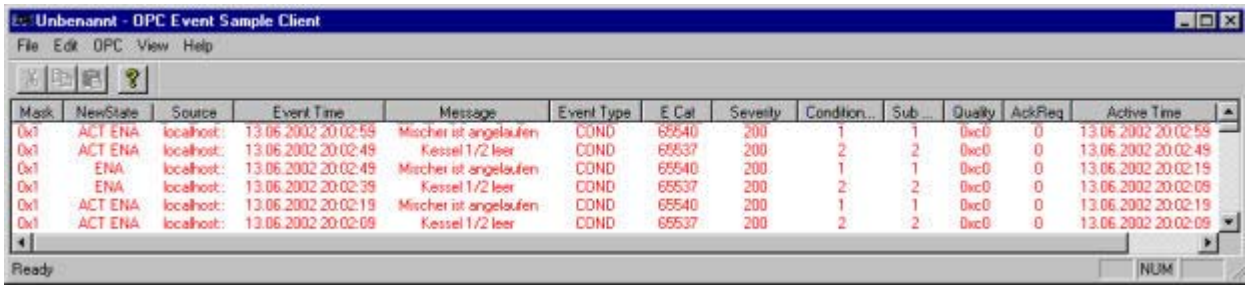
- 在 WinCC OPC A&E 服务器的 WinCC 项目中创建多个“二进制”数据类型的内部变量。
- 在 WinCC OPC A&E 服务器的 WinCC 项目中组态 WinCC 消息系统。 将消息链接至内部变量。
- 用图形编辑器组态画面。 将 WinCC 报警控件和 I/O 域添加到画面中。 将消息变量链接到图形对象。
- 在启动列表中启用“报警记录运行系统”。
- 启用 WinCC OPC A&E 服务器的 WinCC 项目。

步骤

1. . 将“SampleClientAE.exe”“二进制”文件复制到所选的目录中。 该应用程序仅在在线帮助中可用。
2. 在菜单栏中选择“OPC” > “连接...”。 在“OPC 报警服务器”对话框中选择“OPC.WinCC-AlarmsEvent”。 单击“确定”关闭对话框。
3. 从菜单栏中选择“OPC” > “事件订阅...”。 将打开“事件订阅”对话框。
4. 选中对话框中标签为“激活”的复选框。 在“缓冲时间”和“最大尺寸”域中输入“1000”。 单击“确定”关闭“事件订阅”对话框。



5. 来自 WinCC 消息系统的消息将显示在 OPC 事件实例客户机中。



6. 从菜单栏中选择“OPC” > “过滤器”。 将打开“过滤器”对话框。 从“事件类别”域中选择类别。 单击“确定”关闭“过滤器”对话框。
7. 符合过滤标准的消息将显示在 OPC 事件实例客户机中。

“缓冲时间”和“最大大小”参数

根据 OPC 规范，在 WinCC 中按如下组态“缓冲时间”和“最大大小”参数：

OPC 客户机要求返回值	WinCC 使用
缓冲时间 < 100 OPC_S_INVALIDBUFFERTIME	修改后的缓冲时间 = 100
100 <= 缓冲时间 <= 600000 S_OK	修改后的缓冲时间 = 缓冲时间
缓冲时间 > 600000 OPC_S_INVALIDBUFFERTIME	修改后的缓冲时间 = 600000
最大大小 = 0 OPC_S_INVALIDMAXSIZE	修改后的最大大小 = 1000
0 < 最大大小 < 10 OPC_S_INVALIDMAXSIZE	修改后的最大大小 = 10
10 <= 最大大小 <= 1000 S_OK	修改后的最大大小 = 最大大小
最大大小 = 1000 OPC_S_INVALIDMAXSIZE	修改后的最大大小 = 1000

创建用户时，可以设置参数。 然而，在创建之后，不能使用 SetState() 修改已存在的用户。

更多的信息，请参阅 http://www.opcfoundation.org。

参见

www.opcfoundation.org (http://www.opcfoundation.org)

4.9.5 带层级访问的 OPC A&E 服务器

4.9.5.1 OPC A&E 服务器的功能

导言

OPC-A&E 服务器使用 DCOM 服务在具有 OPC 能力的应用程序之间传送消息。 OPC A&E 服务器支持规范 OPC Alarm&Event 1.10。

下一章说明 WinCC 消息系统在带层级访问的 OPC A&E 上的映射，以及 OPC A&E 服务器支持的属性。 本文档包括对特定信息的概述。 有关更多信息，可参见“OPC 报警 & 事件 1.0”规范。

操作原理

OPC-A&E 客户机通过订阅接收 WinCC 消息。 可以使用订阅过滤器减少随订阅传送的事件数。 OPC-A&E 客户机可针对每个显示消息属性的事件类别进行设置。

安装

可以在安装 WinCC 期间选择 WinCC OPC A&E 服务器。 安装后，WinCC OPC A&E 服务器无需进行任何附加组态便可立即使用。

从 WinCC V6.2 开始，可在 WinCC 服务器和 WinCC 客户机上安装 WinCC OPC A&E 服务器。

许可证

要运行 OPC A&E 服务器，必须在每台运行 OPC A&E 服务器的计算机上安装下列许可证之一：

- WinCC 基本系统
- WinCC 连通性软件包插件

事件类型

带层级访问的 OPC A&E 服务器支持条件事件、简单事件和跟踪事件。

与条件有关的事件

对于与条件有关的事件，事件会与某个条件相关联。 例如，条件可以是违反了某个变量的限制。 这种违反限制的情况会产生一条消息，该消息在 OPC A&E 中显示为报警。

简单事件

简单事件是提示 OPC A&E 客户机关于事件的消息。 简单事件包括启动和关闭程序。

说明

在使用冗余系统时，请注意如下事项：
更新变量时，与内部变量互连的简单事件会发送两次。
第一个消息由主机服务器触发，第二个消息由备用服务器触发。

跟踪事件

跟踪事件连同操作员输入消息一并发送到 OPC A&E 客户机。 通过在过程中进行人工干预触发操作员输入消息。

OPC A&E 客户机

所有符合 OPC Alarms & Events 1.10 规范的 OPC A&E 客户机都可以访问 OPC A&E 服务器。 OPC A&E 客户机还可以是专有客户机。 通过创建专有 OPC 客户机，可以满足大多数用户特定需求。例如，可使用 OPC A&E 客户机对来自不同 OPC A&E 服务器的报警进行分析和联合归档。 无法对已归档消息进行确认；只能对当前报警和事件进行确认。

说明

OPC 上的文档

可以在“接口 > OPC - 用于过程控制的 OLE”一章中找到有关 OPC 的附加信息。

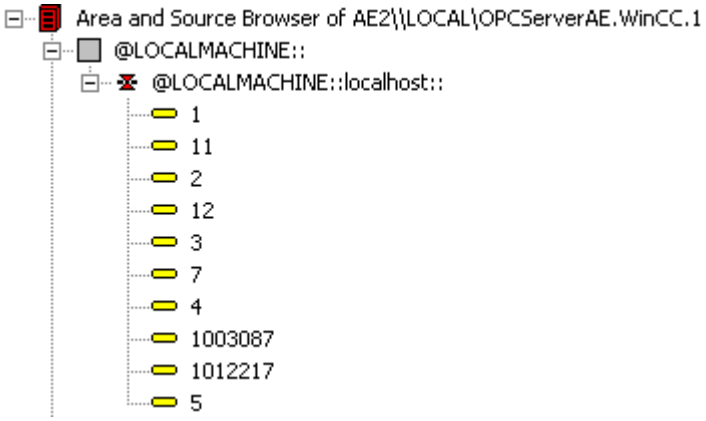
4.9.5.2 WinCC V6.2 SP2 或更高版本的 OPC A&E 服务器

OPC A&E 与带层级访问的 OPC A&E 之间的差异

使用 OPC A&E 显示消息

OPC A&E 服务器支持用于访问消息系统的“条件事件”和“简单事件”。对于“条件事件”，将针对每个来源显示消息号。由于一台 WinCC 服务器可容纳许多消息号，因此难以维护消息概览。

下图显示的是 OPC 浏览器中的显示实例：



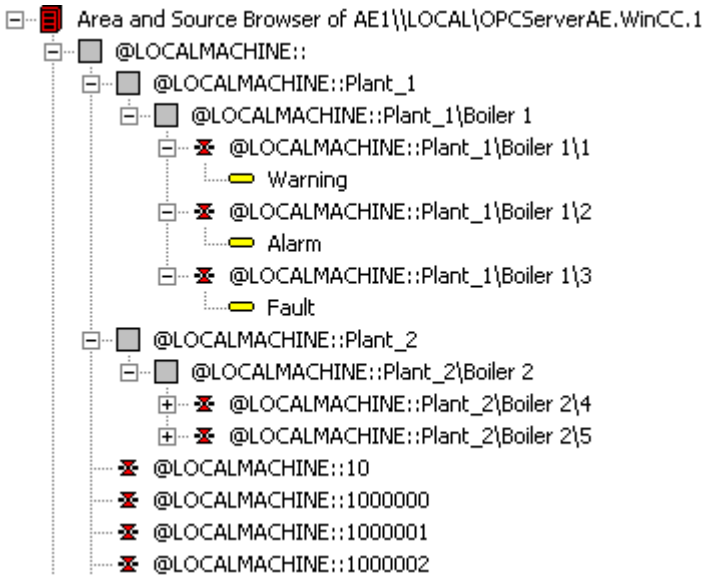
使用 OPC A&E 和层级访问显示消息

带层级访问功能的 OPC A&E 服务器支持的事件类型有条件事件、简单事件和跟踪事件。

用户文本块 2 决定“条件事件”消息的来源。使用缺省设置时，用户文本块 2 对应于故障位置。要以层级形式呈现消息，必须将这些消息合并到报警记录消息的用户定义组消息中。组消息的结构由 OPC A&E 中的区域决定。

在系统中触发操作员输入消息时，会发生跟踪事件。

下图显示的是 OPC 浏览器中条件事件的显示实例。除了“区域”和“来源”外，还显示了“条件”：



建议

请在创建新项目时使用带层级访问功能的 OPC A&E 服务器。

在项目升级到 WinCC V6.2 SP2 或更高版本后，OPC A&E 服务器可像以前那样使用，也可进行转换从而实现层级访问功能。可再次撤消转换而不会丢失任何数据。 有关更多信息，请参阅“如何将 OPC A&E 升级到 WinCC V6.2 SP2 或更高版本”。

如何将 OPC A&E 升级到 WinCC V6.2 SP2 或更高版本

简介

WinCC V6.2 SP2 或更高版本的 OPC A&E 由于能够以层级方式访问消息系统而得到了增强。不带层级访问功能的 OPC-A&E 服务器仍然是标准服务器。

从 OPC A&E 升级

如果使用带层级访问功能的 OPC A&E，并想要使用所有功能，则可能需要修改当前使用的 OPC A&E 客户机。

下面从项目的 WinCC 版本开始介绍可行的 OPC A&E 升级方案：

- 将项目从 WinCC V6.2 升级到 WinCC V6.2 SP2
- 项目在 WinCC V6.2 SP2 或更高版本中创建

从 WinCC V6.2 升级到 WinCC V6.2 SP2

对于 OPC A&E，可按以下方式将 WinCC V6.2 项目升级到 WinCC V6.2 SP2：

保留以前不带层级访问功能的 OPC A&E

如果想要继续使用以前使用的 OPC A&E 服务器，则下列方案可行：

- 如果没有更改标准的“CcAeProvider.ini”文件，则无需进行任何其它设置。
- 如果更改了标准的“CcAeProvider.ini”文件，并想要保留这些更改，请进行如下操作：
  - 将 WinCC 安装路径中的“CcAeProvider.ini”文件保存到“OPC\AlarmEvent\bin”文件夹中。
    如果使用的是分布式系统或集成到 STEP 7 中的系统，请将项目中的该文件保存到 WinCC 客户机或 OS 上。
  - 升级后，将该文件复制到 WinCC 项目目录。
    如果使用的是分布式系统或集成到 STEP 7 中的系统，请将 WinCC 服务器或 ES 上的该文件复制到客户机项目或 OS 项目的项目目录中。
- 如果更改了标准的“CcAeProvider.ini”文件，并想要使用产品随附的标准“CcAeProvider.ini”文件替换该文件，请在升级前删除服务器和客户机或 ES 计算机和 OS 计算机上的相应文件。在 ES 上，项目文件夹位于“wincproj”子文件夹中。

切换到带层级访问功能的 OPC A&E

如果想要使用带层级访问功能的 OPC A&E 服务器，请按以下步骤操作：

1. 升级后，将“CcAeProvider.ini”文件复制到项目文件夹中。该文件位于 WinCC 安装路径的“OPC\AlarmEvent\Hierarchical-Access”文件夹中。
2. 更新客户机或对 OS 服务器执行完整下载。

在 WinCC V6.2 SP2 或更高版本中创建项目

在 WinCC V6.2 SP2 或更高版本中创建的新项目仍使用无层级访问功能的 OPC A&E 服务器。不必进行任何附加设置。

如果想要使用带层级访问功能的 OPC A&E 服务器，请按以下步骤操作：

1. 升级后，将“CcAeProvider.ini”文件复制到 ES 项目的项目文件夹中。可在 WinCC 安装路径下的“OPC\AlarmEvent\Hierarchical-Access”文件夹中找到此文件。
2. 更新客户机或对 OS 服务器执行完整下载。

4.9.5.3 在 OPC A&E 上映射 WinCC 消息系统

映射 WinCC 消息系统

简介

因进行组态而产生的 WinCC 消息系统定义过程中的哪个事件将会生成消息。此消息在 OPC A&E 中显示为事件通知。

在带层级访问功能的 OPC A&E 上映射 WinCC 消息系统

WinCC 用户文本块 2 的 OPC 源和 WinCC 用户文本块 1 的 OPC 消息在 WinCC 中用作映射 WinCC 消息系统的默认设置。

概述

下表显示最重要的事件通知属性及来自 WinCC 消息系统的相应信息。

使用已组态属性的事件显示在表的第三列中：

- “S”表示简单事件
- “C”表示条件事件
- “T”表示跟踪事件

OPC	WinCC 消息系统	事件类型
Area	组消息的结构决定 OPC A&E 中的区域。如果没有为消息组态组消息，则只有与服务器前缀对应的 OPC 区域可用。	S、C、T
Source	指示消息源。消息源的格式为“<服务器前缀>::Area\用户文本块 2”。本地计算机的服务器前缀为“@LOCALMACHINE”。服务器前缀始终显示服务器层级中的顶级 Areas。	S、C、T
Time	发布已接收、已发送和已确认消息的时间戳。以 UTC（协调世界时）格式发布时间戳。	S、C、T
Type	指示该事件是简单事件、跟踪事件还是条件事件。	S、C、T
Severity	返回消息的优先级。	S、C、T
EventCategory	指示消息类别。“事件类别”由“类别 ID”和“类别说明”组成。“类别 ID”对应于消息类别的内部 ID。“类别说明”对应于消息类别的名称。	S、C、T
Message	指示相应消息号的消息文本。	S、C、T

OPC	WinCC 消息系统	事件类型
Condition	指示消息类型。	C
Sub-condition	对应于“Condition”参数。	C
ChangeMask	指定条件变化。 有关详细信息，请参阅“确认理论”。	C
NewState	指示条件的当前状态。 有关详细信息，请参阅“确认理论”。	C
ConditionQuality	返回消息质量。有关详细信息，请参阅“质量代码”。	C
AckRequired	指示消息是否需要确认。	C
EventAttribute	列出相关消息所需的属性。有关详细信息，请参阅“WinCC 消息系统的属性”。	C
Quality	返回消息的质量代码。	C
Cookie	不包括客户机的任何可用信息	C
ActorID	指示确认消息的用户。	T

说明

如果将不带通配符的文本指定为该区域的过滤器，则只返回该区域的消息。如果想要包括位于该指定区域以外区域中的源，则需要使用通配符。

注意

如果如下所示运行 OPC A&E 服务器，则**必须**在所连接的 OS 服务器上对消息类别和消息类型进行完全相同的组态：

- 在 WinCC 客户机上
- 在连通站上

如果 OS 服务器未进行完全相同的组态，所使用的 OPC 客户机必须直接访问相应的 OS 服务器。

映射消息优先级

导言

消息优先级由 OPC A&E 服务器映射到属性“Severity”。

在消息传递系统中组态报警时，可以组态“0”和“16”之间的优先级。 OPC A&E 规范为严重性定义的取值范围是“1”到“1000”。 在这种情况下，“1”代表严重性最低，“1000”代表严重性最高。

因此，优先级的值相应显示为 OPC 严重性。 在标准映射中，优先级“0”指定给 OPC 严重性“1”，优先级“16”指定给 OPC 严重性“1000”。 所有其它优先级值在“0”与“1000”之间进行线性插值。

WinCC 消息系统的属性

简介

下表列出了 WinCC 消息系统的 OPC 属性。这些属性在 WinCC 消息系统中组态。某些属性仅供 WinCC 内部使用，与 OPC A&E 客户机无关。因此，这些属性未包含在该表中。

属性

OPC 属性	WinCC 消息系统	数据类型
CLASSNAME	输出消息类别名称。	VT_BSTR
TYPENAME	输出消息类型名称。	VT_BSTR
FORECOLOR	输出已激活、已禁用和已确认消息的文本颜色。	VT_I4
BACKCOLOR	输出已激活、已禁用和已确认消息的背景色。	VT_I4
FLASHCOLOR	输出闪烁颜色。	VT_I4
FLAGS	指示必须确认该消息	VT_I4
TEXT01	输出用户文本块 01 的内容。	VT_BSTR
TEXT02	输出用户文本块 02 的内容。	VT_BSTR
TEXT03	输出用户文本块 03 的内容。	VT_BSTR
TEXT04	输出用户文本块 04 的内容。	VT_BSTR
TEXT05	输出用户文本块 05 的内容。	VT_BSTR
TEXT06	输出用户文本块 06 的内容。	VT_BSTR
TEXT07	输出用户文本块 07 的内容。	VT_BSTR
TEXT08	输出用户文本块 08 的内容。	VT_BSTR
TEXT09	输出用户文本块 09 的内容。	VT_BSTR
TEXT10	输出用户文本块 10 的内容。	VT_BSTR
PROCESSVALUE 01	输出过程值块 01 的内容。	VT_VARIANT
PROCESSVALUE 02	输出过程值块 02 的内容。	VT_VARIANT

OPC 属性	WinCC 消息系统	数据类型
PROCESSVALUE 03	输出过程值块 03 的内容。	VT_VARIANT
PROCESSVALUE 04	输出过程值块 04 的内容。	VT_VARIANT
PROCESSVALUE 05	输出过程值块 05 的内容。	VT_VARIANT
PROCESSVALUE 06	输出过程值块 06 的内容。	VT_VARIANT
PROCESSVALUE 07	输出过程值块 07 的内容。	VT_VARIANT
PROCESSVALUE 08	输出过程值块 08 的内容。	VT_VARIANT
PROCESSVALUE 09	输出过程值块 09 的内容。	VT_VARIANT
PROCESSVALUE 10	输出过程值块 10 的内容。	VT_VARIANT
STATETEXT	输出状态消息。	VT_BSTR
INFOTEXT	输出消息信息文本。	VT_BSTR
LOOPINALARM	指示是否组态报警回路。	VT_I4
CLASSID	输出消息类别 ID。	VT_I4
TYPEID	输出消息类型 ID。	VT_I4
MODIFYSTATE	输出消息的状态变量值。	VT_I4
AGNR	输出产生消息的 AS 的编号。	VT_I2
CPUNR	输出产生消息的 CPU 的编号。	VT_I2
DURATION	输出激活、禁用和确认消息之间的时间间隔。	VT_I4
COUNTER	输出自运行系统启动以来的消息数。	VT_I4
QUITSTATETEXT	指示消息是否被确认。	VT_BSTR
QUITCOUNT	输出未确认的活动消息数。	VT_I4
PARAMETER	输出消息参数。（消息组态画面）	VT_BSTR
BLOCKINFO	输出消息块的当前内容。	VT_BSTR
ALARMCOUNT	输出未决消息数。	VT_I4
LOCKCOUNT	输出锁定消息数。	VT_I4
PRIORITY	指示已组态的消息优先级。	VT_I4

OPC 属性	WinCC 消息系统	数据类型
APPLICATION	输出触发了消息的应用程序。	VT_BSTR
COMPUTER	输出处理了消息的计算机的名称。	VT_BSTR
USER	输出处理了消息的用户的名称。	VT_BSTR
COMMENT	输出消息注释。	VT_BSTR
HIDDEN-COUNT	输出隐藏消息数。	VT_I4
BIG COUNTER	输出自运行系统启动以来的消息数。	VT_CY
OS-HIDDEN	输出消息的隐藏状态。	VT_BOOL
OS-EVENTID	输出消息的已组态消息 ID。	VT_I4

确认方法

导言

WinCC 中的确认策略是如何显示和处理消息从“进入”到“离开”的过程。 在 OPC A&E 服务器上，该消息状态用“ChangeMask”和“NewState”参数来显示。

条件事件、简单事件和跟踪事件

来自系统的消息将作为有确认的条件事件发送到客户机。

为了使消息作为简单事件接受处理，消息的消息等级必须符合下列条件：

- “确认进入”未激活。
- “无离开状态的消息”激活。

在 WinCC 中，消息等级“系统，不需要确认”和消息类型“操作员输入消息”的消息作为跟踪事件传送。

注意
具有“系统，不需要确认”消息等级和“过程控制系统”消息类型的消息作为“系统消息”事件类别的简单事件进行传送。

ChangeMask

“ChangeMask”参数将记录消息状态改变的地址。

参数值：

- OPC_CHANGE_ACTIVE_STATE
- OPC_CHANGE_ENABLE_STATE
- OPC_CHANGE_ACK_STATE

NewState

“NewState”参数指示改变后的消息状态。

参数值：

- OPC_CONDITION_ACTIVE
- OPC_CONDITION_ENABLED
- OPC_CONDITION_ACKED

概述

WinCC	NewState	ChangeState
已接收的消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE
有确认的离开消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE
无确认的离开消息	OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE

WinCC	NewState	ChangeState
已确认的消息（消息未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
已确认的消息(消息不再未决)	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
锁定的消息	----------	OPC_CHANGE_ENABLED_STATE
解锁的消息	OPC_CONDITION_ENABLED	OPC_CHANGE_ENABLED_STATE
已确认的进入消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACTIVE_STATE
有确认的进入、离开消息	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
无确认的进入、离开消息	OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE

WinCC	NewState	ChangeState
由系统确认的消息（消息未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
由系统确认的消息（消息不再未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
紧急确认的消息（消息未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ACKED OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE
紧急确认的消息（消息不再未决）	OPC_CONDITION_ACTIVE OPC_CONDITION_ENABLED	OPC_CHANGE_ACK_STATE

说明

历史报警和事件不会被确认。 OPC A&E 历史事件接口仅有读取访问权限。

4.9.5.4 OPC A&E 质量代码

引言

质量代码用于判断消息的状态和质量。 下表列出了 OPC A&E 的质量代码。

质量代码

代码	质量	状态
0xC0	OPC_GOOD	确定
0x40	OPC_UNCERTAIN	在不确定的情况下返回，例如：发生延迟确认（回执）时。
0x00	OPC_BAD	在与源的连接中断时返回。

4.9.6 读取归档消息

4.9.6.1 访问归档消息

简介

可以使用 OPC 客户机，通过 OPC A&E 服务器访问归档消息。支持使用以下两种方法来访问归档消息：

- 输出过去某个时间段的归档消息
- 输出过去某个时间段的归档消息，但不指定截止时间。 输出归档消息后，其它所有新生成的消息会自动发送到 OPC 客户机。

注意
读取归档消息后，无法使用消息的已返回“ActiveTime”来确认该消息或跟踪消息转变。为确保这一点，OPC A&E 客户机必须检查带有额外标记“OPC_HAE_HISTORICAL_EVENTFLAG”的消息的“EventType”。归档消息的“ActiveTime”是不正确的。可以在“标识归档消息”下找到有关额外标记的信息。

查询“历史报警和事件”功能

除了标准过滤器外，WinCC 的扩展 OPC A&E 服务器还随附了以下过滤器：

滤波器	过滤器的值	描述
OPC_HAE_FILTER_BY_TIMEFRAME	0x80000000	与用于 OPC 历史数据访问的“ReadRaw”函数对应
OPC_HAE_FILTER_BY_STARTTIME	0x40000000	与用于 OPC 历史数据访问的“AdviseRaw”函数对应

源过滤器和历史报警请求

要请求归档消息，OPC 客户机必须支持“SetFilter”的订阅功能。如果将关键字“OPCHAEServer”也插入到订阅的“源过滤器”序列中，则 OPC 服务器还会发送归档消息。除此关键字外，还可使用其它参数定义要读取的消息：

- 方法
- 时间段
- 有限制或没有限制

在过滤器中指定的源列表除了“OPCHAEServer”源以外，还可以包括其它源名称。在此情况下，订阅将仅传送给定来源的历史事件。源名的顺序不连贯。

组态源过滤器后，所选时间段可从客户机上通过“Refresh”调用进行调用。

4.9.6.2 使用 OPC 访问归档消息的语法

语法

```
OPCHAEServer hMode=(read|advise) htStartTime=szTime
[hEndTime=szTime] [bBounds=(TRUE|FALSE)]
```

参数

hMode = [read|advise]

该参数是必需的。 定义如何读取归档消息和事件。

Read： 输出过往定期的归档消息和事件（如果是 OPC 历史数据访问，则可与 ReadRaw 相比）。

下面是设置用于在最后 30 分钟进行读取的过滤器的实例：

```
OPCHAEServer hMode=read htStartTime=NOW-30M bBounds=TRUE
```

Advise： 输出某个定期的归档消息和事件。接收所有归档消息后，将以与激活订阅情况下相同的方式发送新消息（相当于 OPC 历史数据访问情况下的 AdviseRaw）。

在下列实例中，将读取最后 30 分钟的消息（订阅必须激活）：

```
OPCHAEServer hMode=advise htStartTime=NOW-30M
```

说明

参数“htStartTime”和“htEndTime”支持以下符号：

- 相对符号，例如 NOW
- 符号值，例如 NOW、YEAR、MONTH
- 根据 XML 符号指定绝对 UTC 数据/时间值： 2006-09-01T10:00:00.000Z

使用符号表示法对应于 OPC 历史数据访问语法。

htStartTime =

该参数是必需的。 定义从归档中读取消息和事件的开始时间。

htEndTime =

该参数是可选的。 定义从归档中读取消息和事件的结束时间。 如果“hMode = read”，使用缺省设置“NOW”。

bBounds = [TRUE|FALSE]

该参数是可选的。 定义如何处理接近开始时间和结束时间的消息。 此功能与 OPC 历史数据访问完全相同。

bBounds=FALSE：

- 传送的第一个消息的时间标志 >= htStartTime
- 传送的最后一个消息的时间标志 >= htEndTime

bBounds=TRUE：

- 传送的第一个消息的时间标志 <= htStartTime
- 传送的最后一个消息的时间标志 >= hEndTime

缺省设置为 FALSE。

4.9.6.3 归档消息的读方法

导言

可以使用两种读模式之一来读取归档消息：

- 读
- 建议

“读”模式

“读”模式用于读取过去某个已定义时间段的归档消息。 读取消息的顺序始终根据每个 OS 服务器按时间顺序从报警进行读取。 通过设置开始和结束时间，可以指定是最先还是最后读取最后一条消息。 如果开始时间早于结束时间，将最后输出最后一条消息。

要使用“读”模式，需要对订阅运行下列函数：

1. SetFilter
2. Refresh

“Refresh”期间运行的“SetFilter”会被拒绝。 在“Refresh”期间激活订阅不会对刷新造成任何影响。

将继续以 Refresh 标记传送历史事件。

根据激活订阅的标准反应来传送新近生成的事件：

- 考虑除“历史”源“OPCHAEServer”外的 set filter 值
- 没有 Refresh 标记

客户机因而可以根据 Refresh 标记区分已接收的事件。 事件数据包从未同时包含历史事件和新事件。

- 具有 Refresh 标记的事件数据包仅包含历史事件。 这些事件还可以按队列排列。
- 没有 Refresh 标记的事件数据包仅包含新近生成的事件。

“建议”模式

“建议”模式用于读取过去某个已定义时间段之后的归档消息。 读取所有归档消息后，将按照操作已激活订阅的方式发送新消息。 将根据每台 OS 服务器按时间顺序传送归档消息：将传送开始时间以后的归档消息。 然后，将传送新近的归档消息。

请注意，不应为“建议”定义结束时间。

激活的订阅将用于“建议”模式。 如果对激活的订阅运行“SetFilter”函数，则会立即传送历史报警。

如果对未激活的订阅运行“SetFilter”函数，则只会在激活订阅后发送归档消息。 如果想要对未激活的订阅使用“建议”读模式，请按如下步骤操作：

1. SetFilter
2. 使用 SetState 将订阅设置为激活

如果将订阅取消激活，将中断传送。

如果将订阅设置为“未激活”，则将结束传送。 订阅激活时，“SetFilter”会被拒绝。

在“建议”模式下对激活的“历史”订阅运行“Refresh”时，其作用与其对标准订阅的作用相同：

所有排队等候的与条件相关的事件都将传送到具有 Refresh 标记的数据包。 最后一个数据包还包含“Last Refresh”附加标记。

“Refresh”调用对在“建议”模式下读取历史报警没有影响。

4.9.6.4 识别归档消息

一般步骤

使用 EventType 中的附加标记来区分归档消息。 该标记通过 OR 链接来链接到有效的 EventType。

名称	EventType	EventType（归档消息）
OPC_SIMPLE_EVENT	0x01	0x81
OPC_CONDITION_EVENT	0x04	0x84
OPC_TRACKING_EVENT	0x02	0x82
OPC_HAE_HISTORICAL_EVENTFLAG		0x80

实例

实例 1

下列源过滤器用于以“读取”模式输出前 30 分钟的归档消息和事件。 每台 OS 服务器日期最早的消息将第一个输出。 也会发送下限值。

```
OPCHAEServer hMode=read htStartTime=NOW-30M bBounds=TRUE
```

实例 2

下列源过滤器用于以“读”模式输出 2006 年 9 月 1 日 10:00 至 12:00 的归档事件。 每台 OS 服务器的最新消息将第一个输出。 也会发送此时间段的极限。

```
OPCHAEServer hMode=read htStartTime=2006-09-01T12:00:00.000Z
htEndTime=2006-09-01T10:00:00.000Z bBounds=TRUE
```

实例 3

下列源过滤器用于以“建议”模式输出前 30 分钟的归档消息和事件。 读取归档消息后，将按照操作已激活订阅的方式发送新近生成的消息。

```
OPCHAEServer hmode=advise htStartTime=NOW-30M
```

4.10 调试

4.10.1 OPC 调试

简介

通过 DCOM 完成 WinCC OPC 服务器和 OPC 客户机之间的数据交换。

安装 WinCC 后，将正确组态 WinCC OPC 服务器的 DCOM 设置。

如果 WinCC OPC 服务器或客户机与外部 OPC 系统进行通讯，则必须执行相应调整。

必须在用户管理的“DCOM/工作区/COM 安全/访问权限/编辑默认值”中为用户输入“本地访问”和“远程访问”权限。

4.10.2 组态 Windows

4.10.2.1 此为如何组态 Windows 帐号以使用 WinCC OPC

简介

OPC 客户机和 OPC 服务器是 DCOM 应用程序。 分布式 DCOM 应用程序只能使用同一用户帐户运行。 因此，OPC 服务器必须识别 OPC 客户机的用户帐户，反之亦然。

如果 WinCC OPC 服务器和 WinCC OPC 客户机一起使用，则在安装时已保证组态正确。

如果使用外部 OPC 服务器或客户机，请声明用户帐户。

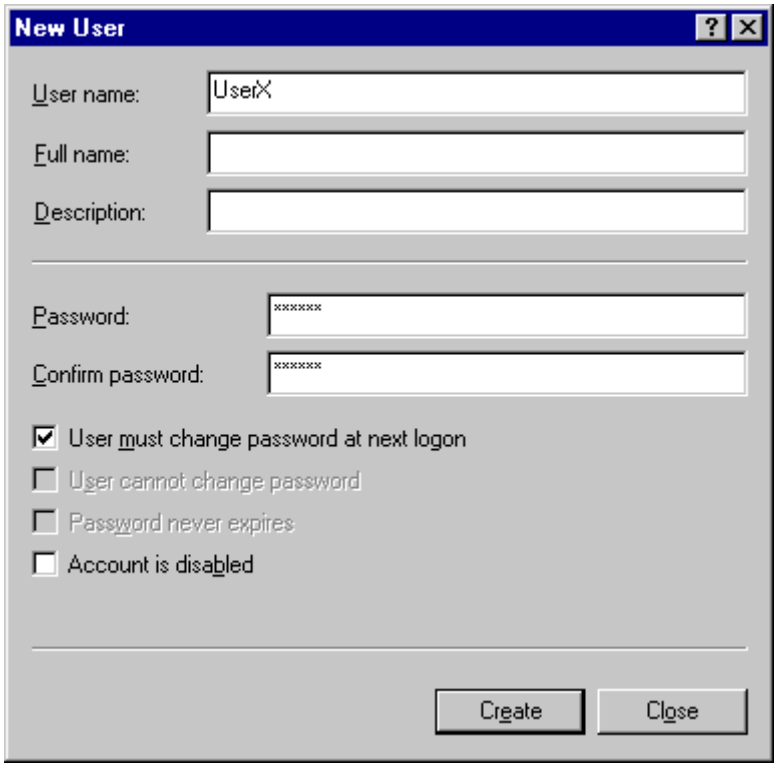
有关授予用户权限的更多信息，请参阅 Windows XP 文档。

要求

作为管理员登录到 WinCC OPC 服务器和 OPC 客户机工作站来组态用户权限。

步骤

1. 转到“控制面板”>“管理”>“计算机管理”>“本地用户和组”。
2. 在“用户”快捷菜单中，选择“新用户”。
   在“新用户”对话框中，输入通讯伙伴的用户帐户详细信息。 单击“创建”，关闭对话框。



3. 单击“用户”图标。 双击相关的用户。 将显示该用户的“属性”对话框。
4. 选择“成员”选项卡。 单击“添加”。 将打开“选择组”对话框。
5. 添加组“users”。
   如果是在安装了 WinCC 的计算机上，还要添加组“SIMATIC HMI”。 单击“确定”关闭所有打开的对话框。

4.10.2.2 如何调整 Windows 防火墙设置

引言

安装 WinCC 后，将正确组态 WinCC OPC 服务器的 Windows 防火墙设置。

如果 OPC 客户机访问不同子网中的 OPC 服务器，则必须调整 OPC 服务器所允许网络区域的组态。

4.10.3 XML

4.10.3.1 调试 - OPC XML

引言

WinCC 的 OPC XML 服务器用作 Web 服务。 这使得可以通过 Internet 访问 PC。 因此，必须定义相应的访问权限。

4.10.3.2 定义 IIS 的安全性设置

简介

Internet 信息服务使得可以通过 Internet 访问 PC。因此，必须定义相应的访问权限。

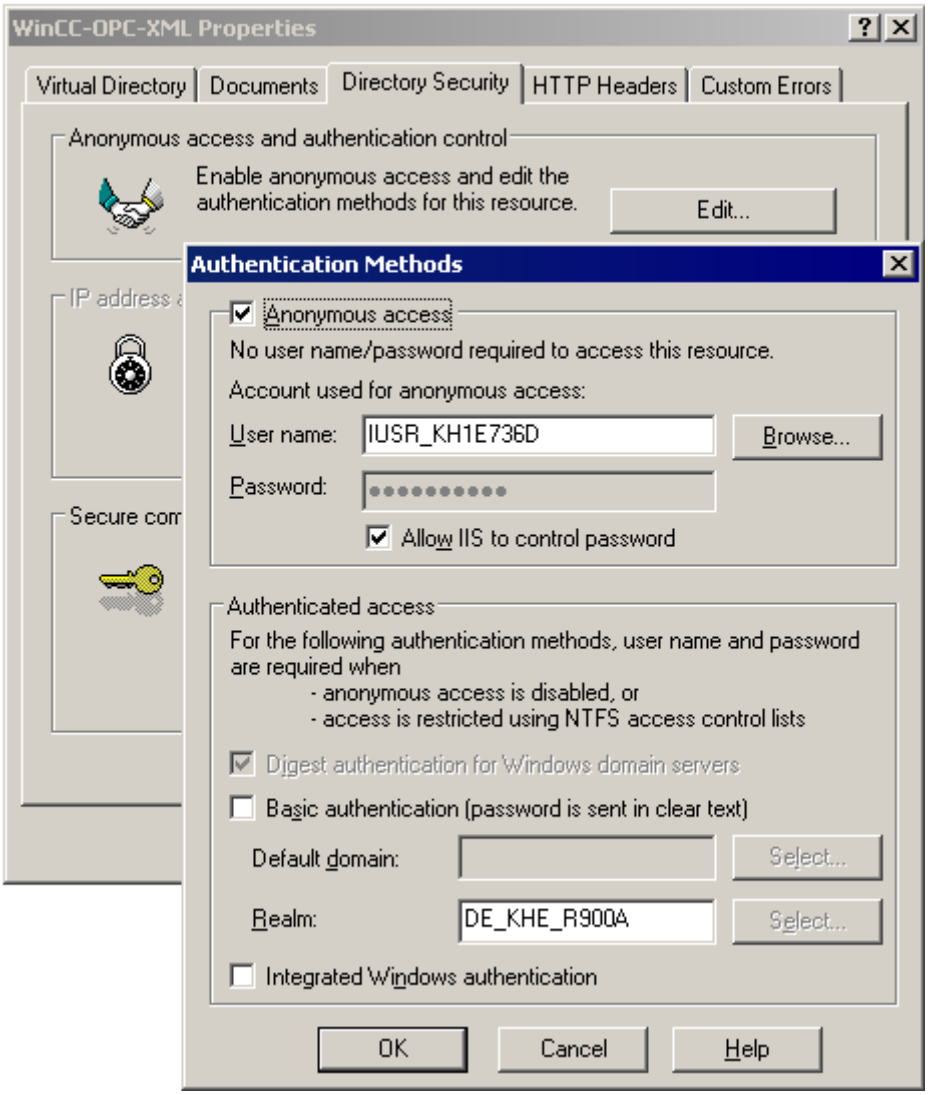
说明

如果对下列设置有任何疑问或遇到任何问题，请联系您的 Intranet/Internet 管理员。

步骤

1. 通过“控制面板”>“管理”>“Internet 信息服务管理器”启动 Windows 2003 中的“管理控制台”。在 Windows XP 中，选择“控制面板”>“管理”>“Internet 信息服务”并激活“管理控制台”。
2. 选择虚拟目录“WinCC-OPC-XML”。选择快捷菜单中的“属性”选项。将打开“WinCC OPC XML 属性”对话框。
3. 单击“目录安全”选项卡。在此选项卡上，选择相关的 Web 服务器安全特性。

4. 单击“用于匿名访问和身份验证的 PLC”中的“编辑”按钮。将显示“身份验证方法”对话框。



5. 在“验证访问”区域中激活“集成的 Windows 身份验证”选项。可以对 Web 服务进行匿名访问，但是出于安全性原因，不应该激活匿名访问。
6. 关闭所有打开的对话框。

参见

如何测试安装 (页 596)

4.10.3.3 如何设置正确的 ASP.NET 版本

简介

如果想要使用 WinCC-OPC-XML-DA 服务器，请确保安装时为 Web 站点设置了正确的“ASP.NET”版本，WinCC-OPC-XML Web 服务通过该版本进行链接。

说明

如果对下列设置有任何疑问或遇到任何问题，请联系您的 Intranet/Internet 管理员。

步骤

1. 通过“控制面板”>“管理”>“Internet 信息服务管理器”启动 Windows 2003 中的“管理控制台”。在 Windows XP 中，选择“控制面板”>“管理”>“Internet 信息服务”并激活“管理控制台”。
2. 选择虚拟目录“WinCC-OPC-XML”。选择快捷菜单中的“属性”选项。将打开“WinCC OPC XML 属性”对话框。
3. 单击“ASP.NET”选项卡。在此选项卡中，组态 Web 服务器“ASP.NET”的设置。
4. 单击“ASP.NET 版本”的文本选择框。如果尚未设置版本“2.x”，请选择版本“2.x”。
5. 关闭所有打开的对话框。
6. 进行更改之后需要重新启动 Web 服务。

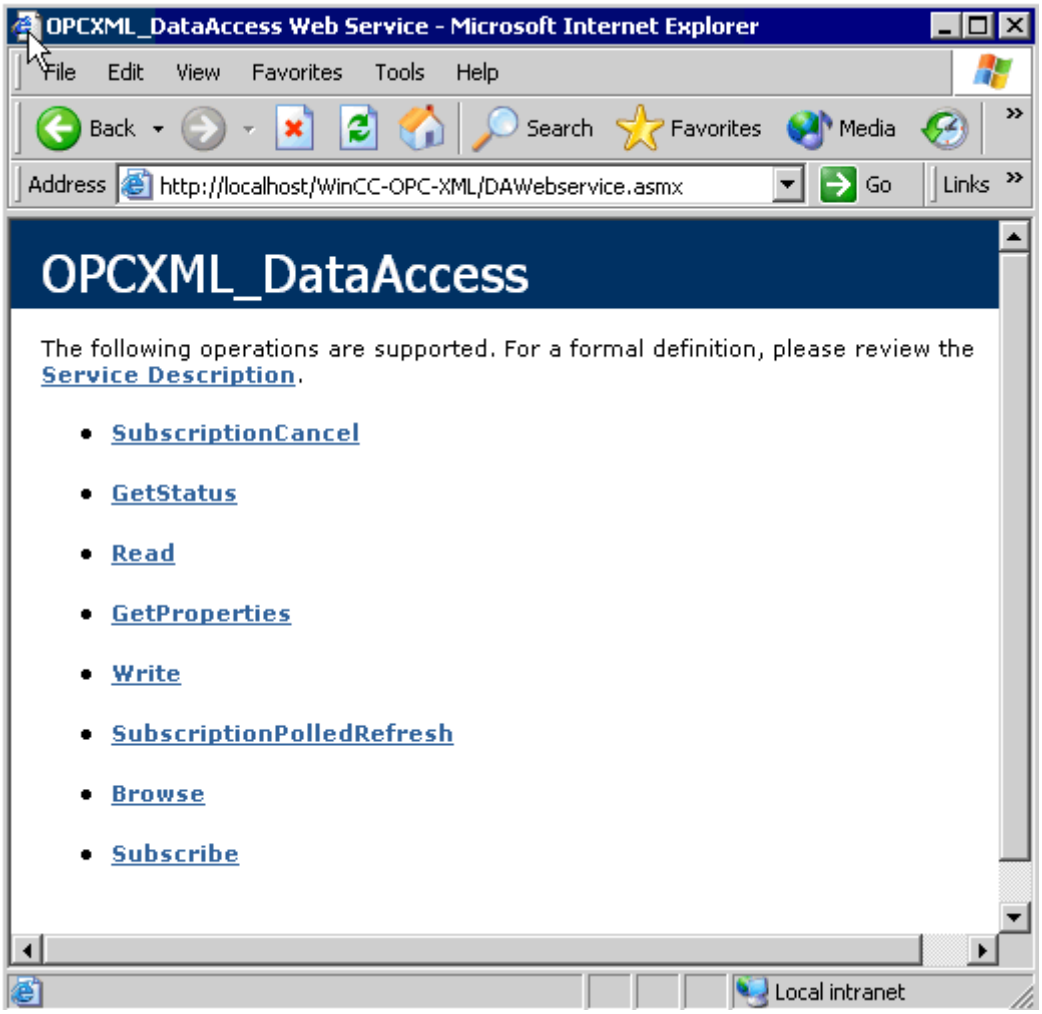
4.10.3.4 如何测试安装

引言

OPC XML-DA 使得 OPC 过程数据可显示为 Web 页面。 可以使用 HTTP 通过 Internet 访问 Web 页面。 下列部分说明如何测试安装。

步骤

1. 在作为 WinCC OPC XML 服务器的计算机上启动 Internet Explorer。
2. 在地址栏中输入 URL“http://localhost/WinCC-OPC-XML/DAWebservice.asmx”。 使用 <ENTER> 键确认输入。
3. 显示 OPC XML DA 功能请求时，安装成功完成。



参见

定义 IIS 的安全性设置 (页 594)

4.10.4 跟踪

引言

“跟踪”功能可用于记录变量值和函数调用，以进行测试和错误分析。

条目存储在跟踪文件中。 跟踪显示逐步建立连接的过程，从而更易于识别连接故障的来源。

设置

跟踪输出必须在操作系统的注册表中设置。 有关更多信息，请参阅“SIMATIC 客户支持”。

索引

P

S

U

W

报

变

标

表

菜

操

处

窗

创

打

单

导

地

电

调

定

动

读

对

范

访

附

改

工

功

关

归

滚

过

函

缓

交

系

下

限

项

消

写

新

修

压

要

用

原

运

在

诊

振

正

质

属

注

状

自

组

最