CODESYS Driver

目录

CODESYS Driver

帮助版本 1.025

目录

概述

CODESYS Driver 提供将 CODESYS 兼容控制器连接至客户端应用程序的可靠方式；其中包括 HMI、SCADA、Historian、MES、ERP 和无数自定义应用程序。

设置

通信协议
基于 ARTI 的 CODESYS V2.3 以太网

信道和设备的最大数量
此驱动程序支持的最大信道数量为 1024。每个信道的最大设备数量为 256 个。

标记数据库创建
此驱动程序的“自动标记数据库生成”功能可实现以更少的耗时完成 OPC 应用程序设置。可将此驱动程序配置为在服务器内自动构建服务器标记列表 (标记与特定于设备的数据相对应)。随后可从 OPC 客户端浏览自动生成的 OPC 标记。

设备超时
驱动程序不支持可配置定时参数。这些参数将始终设置为以下值：

- **连接超时**：20 秒
- **请求超时**：10000 毫秒
- **重试次数**：3 次尝试

注意：此驱动程序不允许配置请求超时。这会导致设备在连接丢失后需要花费很长时间来更新标记质量。

通道属性 - 常规
此服务器支持同时使用多个通信驱动程序。服务器项目中使用的各个协议或驱动程序称为信道。服务器项目可以由具有相同通信驱动程序或具有唯一通信驱动程序的多个信道组成。信道充当 OPC 链路的基础构建块。此组用于指定常规信道属性，如标识属性和操作模式。

“标识”

“名称”：此信道的用户定义标识。在每个服务器项目中，每个信道名称都必须是唯一的。尽管名称最多可包含 256 个字符，但在浏览 OPC 服务器的标记空间时，一些客户端应用程序的显示窗口可能不够大。信道名称是 OPC 浏览器信息的一部分。
有关保留字符的信息，请参阅服务器帮助中的“如何正确命名信道、设备、标记和标记组”。

“说明”：有关此信道的用户定义信息。
这些属性 (包括 Description) 当中有很多具有关联的系统标记。

“驱动程序”：为该信道选择的协议/驱动程序。该属性指定在信道创建期间选择的设备驱动程序。它在信道属性中为禁用设置。

注意：服务器全天在线运行时，可以随时更改这些属性。其中包括更改信道名称以防止客户端向服务器注册数据。如果客户端在信道名称更改之前已从服务器中获取了项，那么这些项不会受到任何影响。如果客户端应用程序在信道名称更改之后发布项，并尝试通过原来的信道名称重新获取项，则该项将不被接受。考虑到这一点，一旦开发完成大型客户端应用程序，就不应对属性进行任何更改。利用“用户管理器”可防止操作员更改属性并限制对服务器功能的访问权限。

诊断

"诊断数据捕获"：启用此选项后，信道的诊断信息即可用于 OPC 应用程序。由于服务器的诊断功能所需的开销处理量最少，因此建议在需要时使用这些功能，而在不需要时禁用这些功能。默认设置为禁用状态。
注意：如果驱动程序不支持诊断，则该属性将被禁用。
有关详细信息，请参阅服务器帮助中的"通信诊断"。

通道属性 - 以太网通信

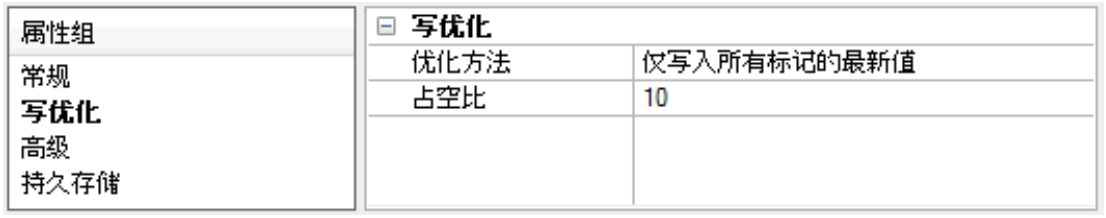
以太网通信可用于与设备进行通信。

以太网设置

"网络适配器"：指定要绑定的网络适配器。如果选择"默认"，则操作系统将选择默认适配器。

通道属性 - 写入优化

与任何 OPC 服务器一样，将数据写入设备可能是应用程序应具备的最重要的功能。服务器旨在确保从客户端应用程序写入的数据能够准时发送到设备。为了达到此目标，服务器提供了可用来满足特定需求以提高应用程序响应能力的优化属性。



写入优化

"优化方法"：控制如何将写入数据传递至底层通信驱动程序。选项包括：

- **"写入所有标记的所有值"**：此选项可强制服务器尝试将每个值均写入控制器。在此模式下，服务器将持续收集写入请求并将它们添加到服务器的内部写入队列。服务器将对写入队列进行处理并尝试通过将数据尽快写入设备来将其清空。此模式可确保从客户端应用程序写入的所有数据均可发送至目标设备。如果写入操作顺序或写入项的内容必须且仅能显示于目标设备上，则应选择此模式。
- **"写入非布尔标记的最新值"**：由于将数据实际发送至设备需要一段时间，因此对同一个值的多次连续写入会存留于写入队列中。如果服务器要更新已位于写入队列中的某个写入值，则需要大大减少写入操作才能获得相同的最终输出值。这样一来，便不会再有额外的写入数据存留于服务器队列中。几乎就在用户停止移动滑动开关时，设备中的值达到其正确值。根据此模式的规定，任何非布尔值都会在服务器的内部写入队列中更新，并在下一个可能的时机发送至设备。这可以大大提高应用性能。
  注意：该选项不会尝试优化布尔值的写入。它允许用户在不影响布尔运算的情况下优化 HMI 数据的操作，例如瞬时型按钮等。
- **"写入所有标记的最新值"**：该选项采用的是第二优化模式背后的理论并将其应用至所有标记。如果应用程序只需向设备发送最新值，则该选项尤为适用。此模式会通过在当前写入队列中的标记发送前对其进行更新来优化所有的写入操作。此为默认模式。

"占空比"：用于控制写操作与读操作的比率。该比率始终基于每一到十次写入操作对应一次读取操作。占空比的默认设置为 10，这意味着每次读取操作对应十次写入操作。即使在应用程序执行大量的连续写入操作时，也必须确保足够的读取数据处理时间。如果将占空比设置为 1，则每次读取操作对应一次写入操作。如果未执行任何写入操作，则会连续处理读取操作。相对于更加均衡的读写数据流而言，该特点使得应用程序的优化可通过连续的写入操作来实现。

注意：建议在将应用程序投入生产环境前使其与写入优化增强功能相兼容。

通道属性 - 高级

此组用于指定高级信道属性。并非所有驱动程序都支持所有属性，因此不会针对不支持的设备显示“高级”组。

“非规范浮点数处理”：通过非规范浮点数处理，用户可以指定驱动程序处理非规范 IEEE-754 浮点数据的方式。非规范值定义为无穷大、非数字 (NaN) 或不正规编号。默认值为“替换为零”。具有原生浮点数处理功能的驱动程序可能会默认设置为“未修改”。选项说明如下：

- **“替换为零”**：此选项允许驱动程序在将非规范 IEEE-754 浮点值传输到客户端之前，将其替换为零。
- **“未修改”**：此选项允许驱动程序向客户端传输 IEEE-754 不正规、规范、非数字和无穷大值，而不进行任何转换或更改。

注意：如果驱动程序不支持浮点值或仅支持所显示的选项，则将禁用此属性。根据信道的浮点规范化设置，将仅对实时驱动程序标记 (如值和数组) 进行浮点规范化。例如，此设置不会影响 EFM 数据。

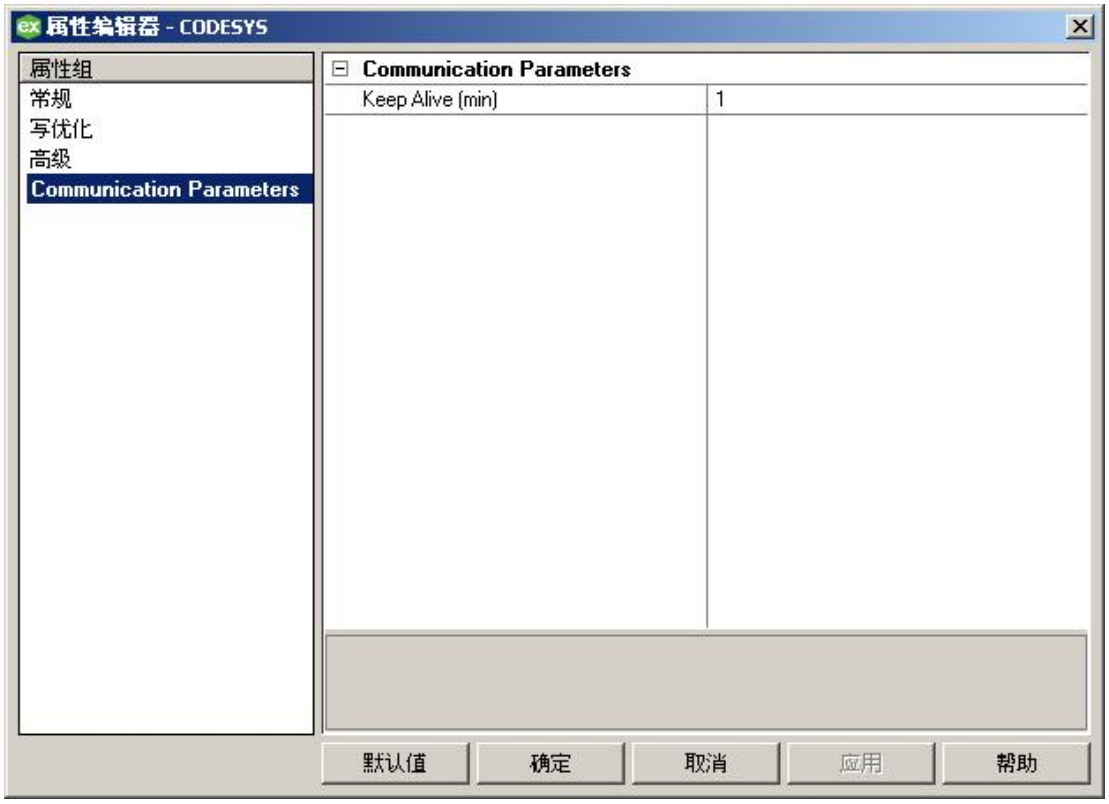
有关浮点值的详细信息，请参阅服务器帮助中的“如何使用非规范化浮点值”。

“设备间延迟”：指定在接收到同一信道上的当前设备发出的数据后，通信信道向下一设备发送新请求前等待的时间。设置为零 (0) 将禁用延迟。

注意：此属性并不适用于所有驱动程序、型号和相关设置。

信道属性 - 通信参数

“通信参数”组是一个属性集合，用于配置信道中所有设备所应用的通信参数。



通信参数

"保持连接"(Keep-Alive)：配置驱动程序在移除所有客户端参考后要与设备保持开放连接的时长 (分钟)。如果在超时之前未添加任何客户端参考，则超时到期时将关闭连接。如果在计时器处于活动状态期间添加了客户端参考，则会随即取消超时，以防止连接被关闭。

设备属性 - 常规

一个设备代表通信信道上的单一目标。如果驱动程序支持多个控制器，则用户必须为每个控制器输入一个设备 ID。

标识

名称：此属性用于指定设备的名称。此为用户定义的逻辑名称，最长可达 256 个字符，并且可以用于多个信道。

注意：尽管描述性名称通常是很好的选择，但浏览 OPC 服务器的标记空间时，一些 OPC 客户端应用程序的显示窗口可能不够大。设备名称和信道名称也成为浏览树信息的一部分。OPC 客户端中，信道名称和设备名称的组合将显示为"信道名称.设备名称"。

有关详细信息，请参阅服务器帮助中的"如何为信道、设备、标记和标记组正确命名"。

说明：有关此设备的用户定义信息。
在这些属性中，有很多属性 (包括“说明”) 具有关联的系统标记。

信道分配：该设备当前所属信道的用户定义名称。

驱动程序：为该设备选择的协议驱动程序。该属性指定在信道创建期间选择的驱动程序。它在信道属性中是禁用的。

型号：此属性指定与此 ID 关联的特定设备类型。下拉菜单中的内容取决于正在使用的通信驱动程序类型。驱动程序不支持的型号将被禁用。如果通信驱动程序支持多个设备型号，则只有当设备未与任何客户端应用程序连接时，才能改变型号的选择。

注意：如果通信驱动程序支持多种型号，则用户应将型号选择与物理设备进行匹配。如果下拉列表菜单中未显示该设备，则选择与目标设备最相近的型号。一些驱动程序支持名为“开放式”的型号选择，该选择使用户无需了解目标设备的具体信息即可进行通信。有关详细信息，请参阅驱动程序帮助文档。

ID：此属性指定设备的工作站/节点/标识/地址。输入的 ID 类型取决于正在使用的通信驱动程序。对于许多驱动程序而言，ID 是一个数值。支持数字 ID 的驱动程序使用户能够输入格式可更改的数值，以适应应用程需要或所选通信驱动程序特点。ID 格式可以是十进制、八进制和十六进制。如果驱动程序基于以太网，或者支持非常规工作站或节点名称，则可使用设备的 TCP/IP 地址作为设备 ID。TCP/IP 地址包含四个由句点分隔的值，每个值的范围在 0 至 255 之间。某些设备 ID 基于字符串。根据不同驱动程序，也可以在 ID 字段中配置其他属性。

操作模式

数据收集：此属性控制设备的活动状态。尽管默认情况下会启用设备通信，但可使用此属性禁用物理设备。设备处于禁用状态时，不会尝试进行通信。从客户端的角度来看，数据将标记为无效，且不接受写入操作。通过此属性或设备系统标记可随时更改此属性。

模拟：此选项可将设备置于模拟模式。在此模式下，驱动程序不会尝试与物理设备进行通信，但服务器将继续返回有效的 OPC 数据。模拟停止与设备的物理通信，但允许 OPC 数据作为有效数据返回到 OPC 客户端。在“模拟模式”下，服务器将所有设备数据处理为反射型：无论向模拟设备写入什么内容，都会读取回来，而且会单独处理每个 OPC 项。项的内存映射取决于组更新速率。如果服务器移除了项 (如服务器重新初始化时)，则不保存数据。默认值为“否”。

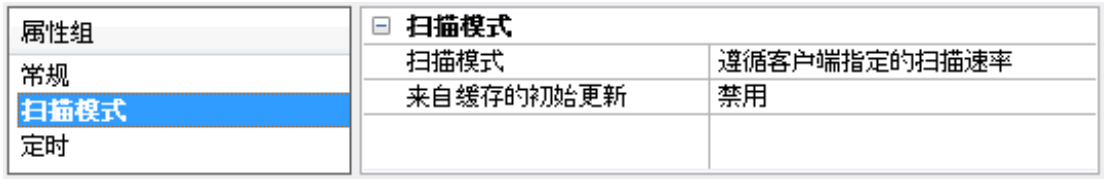
注意：

1. “系统”标记 (_Simulated) 为只读且无法写入，从而达到运行时保护的目的。“系统”标记允许从客户端监控此属性。
2. 在“模拟”模式下，项的内存映射取决于客户端更新速率 (OPC 客户端的“组更新速率”或本机和 DDE 接口的扫描速率)。这意味着，参考相同项、而采用不同更新速率的两个客户端会返回不同的数据。

“模拟模式”仅用于测试和模拟目的。该模式永远不能用于生产环境。

设备属性 - 扫描模式

“扫描模式”为需要设备通信的标记指定预订客户端请求的扫描速率。同步和异步设备的读取和写入会尽快处理；不受“扫描模式”属性的影响。



“扫描模式”：为发送到预订客户端的更新指定在设备中扫描标记的方式。选项说明如下：

- **“遵循客户端指定的扫描速率”**：此模式可使用客户端请求的扫描速率。
- **“不超过扫描速率请求数据”**：此模式可指定要使用的最大扫描速率。有效范围为 10 至 99999990 毫秒。默认值为 1000 毫秒。

注意：当服务器有活动的客户端和设备项且扫描速率值有所提高时，更改会立即生效。当扫描速率值减小时，只有所有客户端应用程序都断开连接，更改才会生效。

- **"以扫描速率请求所有数据"**：此模式将以预订客户端的指定速率强制扫描标记。有效范围为 10 至 99999990 毫秒。默认值为 1000 毫秒。
- **"不扫描，仅按需求轮询"**：此模式不会定期轮询属于设备的标签，也不会在一个项变为活动状态后为获得项的初始值而执行读取操作。客户端负责轮询以便更新，方法为写入 _DemandPoll 标记或为各项发出显式设备读取。*有关详细信息，请参阅服务器帮助中的"设备需求轮询"。*
- **"遵循标签指定的扫描速率"**：此模式将以静态配置标记属性中指定的速率强制扫描静态标记。以客户端指定的扫描速率扫描动态标记。

"来自缓存的初始更新"：启用后，此选项允许服务器为存储 (缓存) 数据的新激活标签参考提供第一批更新。只有新项参考共用相同的地址、扫描速率、数据类型、客户端访问和缩放属性时，才能提供缓存更新。设备读取仅用于第一个客户端参考的初始更新。默认设置为禁用；只要客户端激活标记参考，服务器就会尝试从设备读取初始值。

通过将响应设备脱机放置到特定时间段，驱动程序可以继续优化与同一通道上其他设备的通信。如果设备响应，则该设备会进入开启扫描状态；否则，设备将再次开始其关闭扫描时间段。

"故障时降级"：启用后，设备将自动 off-scan，直到再次响应。
提示：通过使用 _AutoDemoted 系统标记监视其降级状态，确定设备何时 off-scan。

"降级超时"：指定在 off-scan 设备放置之前，请求超时和重试次数的连续周期数。有效范围是1到30连续的失败。默认值为 3。

Demotion Period：指示在达到超时值时设备应放置 off-scan 的时间。在此期间，没有将读取请求发送到设备，并且与读取请求关联的所有数据都被设置为不良质量。当此期间过期时，驱动程序将设备放在扫描位置，并允许进行其他通信尝试。有效范围为 100 至 3600000 毫秒。

"降级时放弃请求"：选择在 off-scan 期间是否应尝试写入请求。禁用以始终发送写入请求，而不考虑降级期间。允许丢弃写入;服务器会自动使从客户端收到的任何写入请求失败，并且不会将消息张贴到事件日志中。

设备属性 - 标记生成

自动标记数据库生成功能使设置应用程序成为一项即插即用操作。选择可以配置为自动构建标记列表的通信驱动程序 (标记与特定于设备的数据相对应)。可以从客户端浏览这些自动生成的标记 (这取决于支持驱动程序的性质)。

如果目标设备支持其自身的本地标记数据库，则驱动程序会读取设备的标记信息，并使用该数据来在服务器中生成标记。如果该设备本身不支持已命名的标记，则驱动程序会根据特定于驱动程序的信息来创建标记列表。这两个条件的示例如下：

1. 如果数据采集系统支持其自身的本地标记数据库，则通信驱动程序将使用在设备中发现的标记名称来构建服务器的标记。
2. 如果以太网 I/O 系统支持其自身可用 I/O 模块类型的检测，则通信驱动程序会基于插入以太网 I/O 机架的 I/O 模块类型在服务器中自动生成标记。

注意：自动标记数据库生成的操作模式可进行完全配置。有关详细信息，请参阅下方的属性说明。

"设备启动时"
此属性指定自动生成 OPC 标记的时间。选项说明如下：

- **"启动时不生成"**：此选项可防止驱动程序向服务器的标记空间添加任何 OPC 标记。这是默认设置。
- **"始终在启动时生成"**：此选项可使驱动程序评估设备，以便获得标记信息。每次启动服务器时，它还会向服务器的标记空间添加标记。
- **"首次启动时生成"**：此选项可使驱动程序在首次运行项目时评估目标设备，以便获得标记信息。它还可以根据需要向服务器标记空间添加任何 OPC 标记。

注意：如果选择自动生成 OPC 标记的选项，添加到服务器标记空间的任何标记都必须随项目保存。用户可以在**"工具"|"选项"**菜单中将项目配置为自动保存。

"对于重复标签"
启用自动标记数据库生成后，服务器需要了解如何处理先前已添加的标记，或在初始创建通信驱动程序后已添加或修改的标记。此设置可控制服务器处理自动生成的以及当前存在于项目中的 OPC 标记的方式。它还可以防止自动生成的标记在服务器中累积。

例如，如果用户更改机架中的 I/O 模块，并且服务器配置为**"始终在启动时生成"**，则每当通信驱动程序检测到新的 I/O 模块时，新标记就会添加到服务器。如果未移除旧标记，则许多未使用的标记可能会在服务器的标记空间中累积。选项包括：

- **"创建时删除"**：此选项可在添加任何新标记之前，将先前添加到标记空间的任何标记删除。这是默认设置。
- **"根据需要覆盖"**：此选项可以指示服务器仅移除通信驱动程序要用新标记替换掉的标记。所有未被覆盖的标记仍将保留在服务器的标记空间中。
- **"不覆盖"**：此选项可以防止服务器移除任何之前生成的标记或服务器中已存在的标记。通信驱动程序只能添加全新的标记。
- **"不覆盖，记录错误"**：此选项与前一选项有相同效果，并且在发生标记覆盖时，也会将错误消息发布到服务器的事件日志。

注意：删除 OPC 标记会影响通信驱动程序已自动生成的标记以及使用匹配已生成标记的名称添加的任何标记。如果标记所使用的名称可能与驱动程序自动生成的标记相匹配，则用户应避免将此类标记添加到服务器。

"父组"：此属性通过指定将要用于自动生成标记的组，来防止自动生成的标记与已手动输入的标记发生混淆。组名称最多可包含 256 个字符。此父组具有一个根分支，可将所有自动生成的标记添加到其中。

"允许自动生成的子组"：此属性用于控制服务器是否为自动生成的标记自动创建子组。这是默认设置。如果禁用，则服务器会在没有任何分组的简单列表中生成设备标记。在服务器项目中，生成的标记使用地址值命名。例如，生成过程中不会保留标记名称。

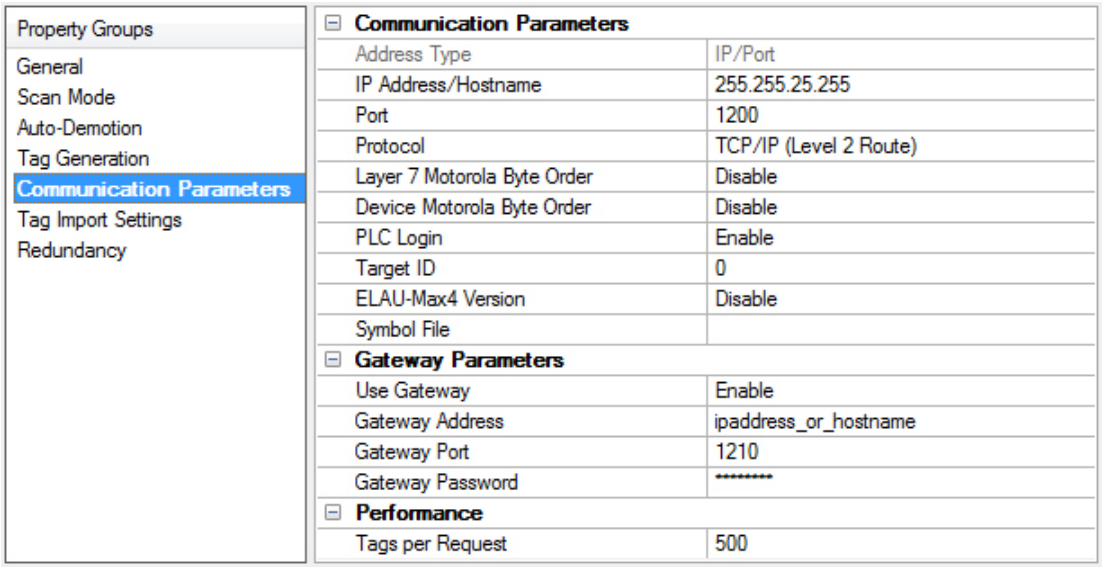
注意：如果在服务器生成标记的过程中，分配给标记的名称与现有标记的名称相同，则系统会自动递增到下一个最高数字，以免标记名称发生重复。例如，如果生成过程中创建了名为 "AI22" 的标记且该名称已存在，则会将标记创建为 "AI23"。

"创建"：开始创建自动生成的 OPC 标记。如果已修改设备的配置，则**"创建标记"**可强制驱动程序重新评估设备以发现可能的标记更改。由于该选项可以通过系统标记进行访问，这使得客户端应用程序能够启动标记数据库创建。

注意：当"配置"对项目进行离线编辑时，会禁用**"创建标记"**。

设备属性 - 通信参数

通信参数属性可用于建立到设备的连接。



通信参数

“地址类型”(Address Type)：指定连接到设备 (仅限 V3 模型) 时使用的地址类型。

“地址”(Address)：指定设备 (仅限 V3 模型) 的逻辑地址或 PLC 名称。

“IP 地址”(IP Address)：指定目标设备的 IP 地址。

“端口”(Port)：指定目标设备上的 Ethernet/IP 端口号。

“协议”(Protocol)：指定此设备中使用的协议。

“层 7 Motorola 字节顺序”(Layer 7 Motorola Byte Order)：指定层 7 是否使用 Motorola 字节顺序 (大端字节序)。

“设备 Motorola 字节顺序”(Device Motorola Byte Order)：指定目标设备是否使用 Motorola 字节顺序 (大端字节序)。在大多数情况下，此选项将与“层 7 Motorola 字节顺序”(Layer 7 Motorola Byte Order) 设置为相同的值。

“PLC 登录”(PLC Login)：指定驱动程序连接后是否应在 PLC 中保持登录状态。如果 PLC 仅支持单个客户端连接，则应禁用此设置，否则应启用。

“目标 ID”(Target ID)：如果这是一个子 PLC 设备，则请指定标识。需要通过另一个 PLC 路由通信的 PLC，称为子 PLC。目标 ID 可提供与子 PLC 通信所需的其他地址信息。当未与子 PLC 通信时，此设置应为 0。

“ELAU-Max4 版本”(ELAU-Max4 Version)：指定目标 PLC 的硬件修订版本。除非设备为 ELAU Max 4 1100 或 1200，否则都应禁用此值。

“符号文件”(Symbol File)：如果符号文件无法存储在设备上，请指定将使用的完整文件名，包括路径。此符号文件必须与设备上存储的符号相匹配。如果符号不匹配，则与该设备间的通信会失败。如果符号文件已存储在设备上，则此处可留空。V3 设备会将符号文件存储在设备上，因此不需要此属性。

即使在不同信道下，所有设备的 IP 地址和端口组合都必须唯一。大多数 CODESYS PLC 不支持并行通信信道，因此，通过在同一 CODESYS PLC 处指向多个通道并不会提高性能。目前，通过在服务器中复制和粘贴信道而创建的设备将具有相同的 IP 地址和端口组合。已复制信道下的所有设备都应进行更改，以移除所有 IP 地址和端口组合重叠。如果未能解决上述冲突，则可能存在未定义的行为。

网关参数

“使用网关”(Use Gateway)：指定连接到设备时是否应使用网关。

“网关地址”(Gateway Address)：指定连接至网关时要使用的 IP 地址或主机名。

“网关端口”(Gateway Port)：指定连接至网关时要使用的端口。

“网关密码”(Gateway Password)：指定连接到网关 (仅限 V2.3 模型) 时要使用的密码。

通过网关进行设备连接时，要求 CODESYS 网关与服务器安装在同一主机上。如果 CODESYS 网关与主机未安装在同一服务器上，则读取标记会导致事件日志中出现一则消息，显示“设备未响应”。

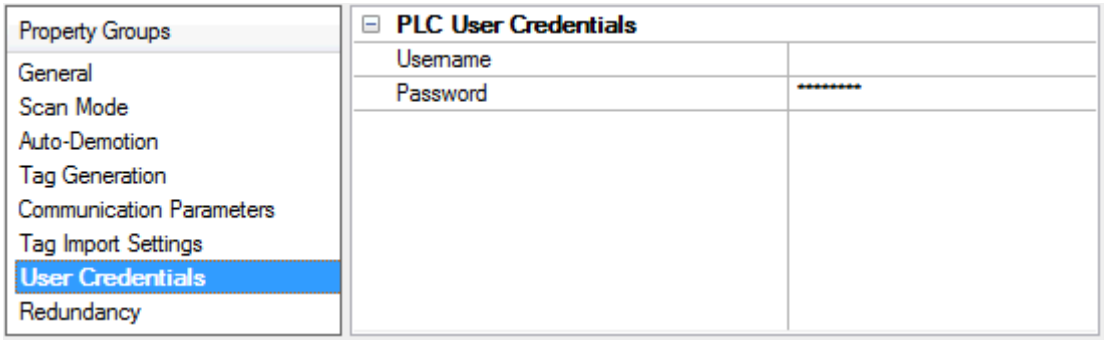
性能

“每个请求的标记数”(Tags per Request)：单个请求中最多可包含的标记数。每个请求都会产生一些服务费用。通常，最好在一个请求中处理多个项，而不是单独请求每个项。但是，在一个请求中请求多个项会导致往返时间 (应答时间) 变长。根据不同应用程序，最好选择将大的列表分割成多个小的请求。

注意：每个请求的写入数是有限的，取决于信道设置中指定的“占空比”。

设备属性 - 用户凭证

CODESYS V3 设备可以要求进行身份验证。如果要求进行身份验证，则用户名和密码属性需要配置正确设置以进行连接。这些设置不适用于 V2.3 设备。

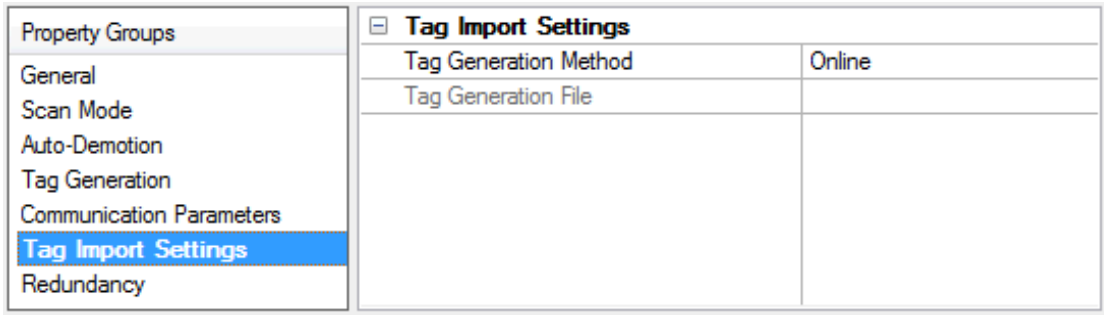


“用户名”(Username)：指定要用于身份验证的帐户名称。支持的用户名最大长度为 254 个字符。

“密码”(Password)：指定与指定的用户名关联的密码。支持的密码最大长度为 251 个字符。

设备属性 - 标记导入设置

“标记导入设置”为用于生成标记的属性。



标记导入设置

“标记生成方法”(Tag Generation Method)：可通过以下两种方法之一进行标记导入。选择**“在线”(Online)** 以从导入时对驱动程序可见的网络设备导入标记。选择**“离线”(Offline)**，在没有设备连接的情况下，通过向服务器提供编译 CODESYS 项目时所创建的相应纯文本符号 (.SYM) 文件导入标记。

“标记生成文件”(Tag Generation File)：指定符号文件的路径和文件名以进行导入。单击**“浏览”(Browse)** (...) 按钮，查找并选择 .SYM 文件。如果选择“在线”(Online) 方法导入，则此属性将会被禁用。

提示：选择文件后，即可开始使用**“标记生成”(Create tags)** 组中的“创建标记”(Create tags) 命令生成标记。

设备属性 - 冗余

Media-Level Redundancy 插件提供冗余。

有关详细信息，请参阅网站、向销售代表咨询或查阅用户手册。

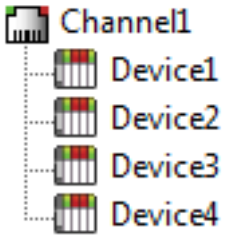
性能优化

优化通信

对于任何可编程控制器，优化系统吞吐量的方式都非常独特，但是，CODESYS Driver 都是相同的。CODESYS Driver 旨在优化读取和写入操作。对于所有数据类型的标记，请求将归组为单个事务。与单个标记事务相比，这可显著提高性能。唯一的限制是需要调整单个事务的标记数。

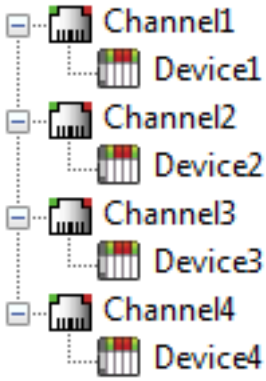
优化应用程序

即使驱动程序速度很快，也可以利用一系列指南来优化应用程序，以获得最佳性能。通信协议 (例如 CODESYS) 称为信道。应用程序中定义的每个信道都表示服务器中一个单独的执行路径。一旦定义了信道，便可在该信道下定义一系列设备。每一个此类设备都代表一个可从中收集数据的 CODESYS 控制器。虽然这种定义应用程序的方法提供了高水平的性能，但它不能充分利用驱动程序或网络。下面显示了使用单个信道配置时应用程序所呈现效果的示例。



在此示例中，每个设备均出现在单个信道下。在此配置中，驱动程序必须尽快从一个设备移动到下一个设备，以有效速率收集信息。随着更多设备的添加或从单个设备请求的信息的增加，整体更新速率会受到不利影响。

如果 CODESYS Driver 仅可以定义一个单信道，如上所示的示例为唯一可用的选项。但是，驱动程序最多可以定义 1024 个信道。使用多个信道，可通过同时向网络发出多个请求来分发数据集合工作载荷。下面显示了使用多个信道来提高性能时相同应用程序所呈现效果的示例。



当前，每个设备已在其自身的信道下定义。在此配置中，单个执行路径专用于从每个设备收集数据。如果应用程序的设备数小于等于 1024 个，则可对其进行精确优化，如此处所示。

即使应用程序设备数大于 1024 个，也可改善性能。虽然设备数小于等于 1024 个时可能是理想情况，但附加信道仍会对应用程序有益。尽管在全部信道上分散设备载荷会使服务器再次从一个设备移动到另一个设备，但是，它现在可以用极少的设备在单信道上进行处理。

数据类型说明

数据类型	说明
布尔型	单个位
字节	无符号 8 位值
字符型	有符号 8 位值
字	无符号 16 位值
短整型	有符号 16 位值
双字型	无符号 32 位值
长整型	有符号 32 位值
四字型	无符号 64 位值
长整型	有符号 64 位值
浮点型	32 位 IEEE 浮点值
双精度	64 位 IEEE 浮点值
日期	64 位 IEEE 日期和时间值
字符串	空终止字符数组

*有关特定于 CODESYS 平台的数据类型说明，请参阅 **CODESYS 数据类型**。*

不支持的数据类型
不支持的数据类型包括 LBCD 和 BCD。

***另请参阅：**地址说明、原子型数据类型*

基于符号标记的寻址
CODESYS Driver 对标记使用符号寻址结构。这些标记 (通常为“原生标记”) 与常规 PLC 数据项的区别在于，标记名称本身是地址，而不是物理或逻辑地址。

客户端/服务器标记地址规则
CODESYS 变量名称对应于客户端/服务器标记地址。CODESYS 变量名称 (通过 CODESYS PLC 输入) 遵循 IEC 61131-3 标识符规则。客户端/服务器标记地址也遵循这些规则，如下所示：

- 必须以字母 (A-Z、a-z) 字符或下划线 (_) 开头。
- 只能包含字母数字字符和下划线。
- 不区分大小写。

客户端/服务器标记名称规则
由于标记名称不可以下划线开头，因此服务器的标记名称分配规则不同于地址分配。

标记范围
全局标记
全局标记是在控制器中具有全局范围的 CODESYS 变量。任何程序或任务都可以访问使用以下符号的全局标记：

.<标记名称>
.<结构体名称>.<标记名称>

注意：结构可嵌套在其他结构中，因此标记名称的前缀可能由多个结构条目组成 (<外部结构体>.<内部结构体>.<我的标记>)。

程序标记
程序标记与全局标记相同，但程序标记的范围局限于定义该标记的程序内。程序标记的寻址规则和限制与全局标记相同。唯一的区别是，程序标记使用程序名称作为前缀：

<程序名称>.<标记名称>
或
<程序名称>.<外部结构体>.<内部结构体>.<标记名称>

示例：
"prog_1.tag_1"：程序 "prog_1" 中包含一个名为 "tag_1" 的标记

标记寻址

CODESYS V2.3

CODESYS 支持两种类型的变量：全局和专用。这两个类别的格式略有不同。下表提供了不同 CODESYS 变量及其相应的服务器寻址语法的示例。

地址类型	地址语法
全局变量	.MyTag
结构体中的全局变量*	.MyStruct.MyTag
变量中的全局位	.MyTag.[0]
专用变量	Program.MyTag
结构体中的专用变量*	Program.MyStruct.MyTag
变量中的专用位	Program.MyTag.[0]

* 支持多级别结构体 (最多八个级别)。

CODESYS V3

CODESYS 支持两种类型的变量：全局和专用。这两个类别的格式略有不同。下表提供了不同 CODESYS 变量及其相应的服务器寻址语法的示例。

地址类型	地址语法
全局变量	Application.GVL.MyTag
结构体中的全局变量*	Application.GVL.MyStruct.MyTag
变量中的全局位	Application.GVL.MyTag.[0]
专用变量	Application Program.MyTag
结构体中的专用变量*	Application Program.MyStruct.MyTag
变量中的专用位	Application Program.MyTag.[0]

* 支持多级别结构体 (最多八个级别)。

嵌套结构

符号标记有可能深度嵌套在任意数量的结构中 (即 .Struct1.Struct2.Struct3.Struct4.Struct5.Struct6.Struct7.Struct8.Struct9.tag)。为避免标记层次因嵌套结构而变得过于复杂，“自动标记生成”可防止服务器生成深度超过 8 个组的标记。如果设备上的标记层次超过了 8 个组，则会在标记名称中放置剩余的组。对于自动生成的标记，第一个组是“全局”或“程序组织单元”(POU) 名称。

注意：

1. 仅当“允许自动生成的子组”(Allow Automatically Generated Subgroups) 属性设置为“已启用”(Enabled) 时才可用。
2. 符号文件包含整体表示为字节数组的结构的数据点。服务器不支持这些标记。

数组

CODESYS Driver 支持所有基本类型的数组和用户定义的结构。此驱动程序对可在 PLC 中定义的数组提供部分支持。除字符串和结构之外，所有类型均支持一维和二维数组。必须使用数组元素语法访问字符串和结构。三维数组和嵌套数组都不能读取为数组。必须使用数组元素语法按元素逐一对其进行访问。

必须提供每个数组维度的大小。服务器中的数组大小必须与 PLC 中的数组大小相匹配，才能对数组进行读取或写入。*请参阅下表查看一些示例*。

地址类型	地址语法
全局 1 维数组 (10 个元素)	.myArray{10}
专用 1 维数组 (100 个元素)	PLC_PRG.myArray {100}
全局 2 维数组 (9 个元素)	.myArray{3}{3}
专用 2 维数组 (25 个元素)	PLC_PRG.myArray {5}{5}

注意：数组所支持的元素数最多为 65535。

数组元素

如果没有必要或不支持读取整个数组，则可以直接访问特定的数组元素。一维、二维和三维数组以及嵌套数组的元素可供访问。请参阅下表查看地址语法示例。请务必注意，尽管必须以数组元素的形式访问结构，但由于 OPC 不支持将整个结构作为单一标记进行读取，因此必须使用结构的各个成员变量对结构进行访问。

地址类型	地址语法
一维数组元素	.myArray[0]
二维数组元素	.myArray[1,7]
三维数组元素	.myArray[2,4,5]
嵌套数组元素 (5 级)	.myNestedArray[1][0][4][5][9]
嵌套三维数组 (2 级)	.my3DNestedArray[4,8,1][3,2,0]

注意：支持非零索引数组和第一个元素以非零起始的数组。示例：要获取以 1 为索引的 1 维数组的第一个元素，请使用 .myArray[1] 作为地址。

位寻址

CODESYS 驱动程序支持针对以下控制器数据类型进行位寻址：无符号短整型、短整型、字节、无符号整型、整型、字、无符号双整型、双整型和双字型。位寻址可提供访问控制器中 1、2 或 4 字节数据类型中的单个位的权限。

写入位标记时，驱动程序会读取控制器中数据类型的内容、更改正在写入的位值并将数据类型的完整内容写回到控制器。这也称为读取/修改/写入 (RMW) 行为。

地址类型	地址语法
全局位变量	Application.GVL.MyTag.[0]
结构中的全局位变量	Application.GVL.MyStruct.MyTag.[1]
专用位变量	.Program.MyTag.[2]
结构中的专用位变量	.Program.MyStruct.MyTag.[2]
数组元素中的全局位变量	Application.GVL.MyArray[0].[5]

注意：位地址的括号内的位值介于 0 到 31 之间。

联合

与 V3 设备进行通信时，CODESYS Driver 支持联合数据结构。在联合中定义的所有组件均具有相同的内存。

如果更改联合中某一个变量的值，则无论数据类型如何，均会对联合中的所有其他变量产生影响。

访问权限

程序的符号文件描述的每个标记均有特定的访问权限：无、读取、写入或两者 (读取和写入)。服务器将依据这些权限确定 PLC 程序对与客户端交互的各个标记的计划，但这些权限可由服务器进行覆盖。这可通过每个标记的“客户端访问”属性实现。如果服务器中的标记配置为“只读”，则无论设备所报告的“访问权限”如何，服务器均只允许客户端执行“读取”操作。如果标记配置为“读/写”，则无论设备所报告的“访问权限”如何，客户端均可以读取并写入标记。

当用户执行自动标记生成时，将根据设备所报告的访问权限为标记配置正确的客户端访问权限。在以下两种情况下不会出现此情况。

只写：报告为“只写”状态的标记将生成为同时具有“读取”和“写入”权限的“读/写”标记，因为不存在“只写”客户端访问权限设置。

无：自动标记生成和手动标记创建操作均不支持报告为具有“无”权限的标记，因为不存在能够正确表示这种访问权限的客户端访问设置。

寻址原子型数据类型

下表显示了可用的 CODESYS 数据类型和 OPC 对等值。

原子型数据类型 (CODESYS 类型)	OPC 数据类型
布尔型	布尔型
无符号短整型	字节、字符
短整型	字节、字符
字节	字节、字符
无符号整型	短整型、字
整型	短整型、字
字	短整型、字
双字型	双字型、长整型
无符号双整型	双字型、长整型
双整型	双字型、长整型
无符号长整型	四字型、双长整型
长整型	四字型、双长整型
长字型	四字型、双长整型
实型	浮点数
LREAL	双精度
时间	双字型
TIME_OF_DAY	双字型
LTIME	四字型
日期	双字型、日期
DATE_AND_TIME	双字型、日期
字符串	字符串
宽字符串	字符串
枚举	短整型、字

不支持的 CODESYS 数据类型

唯一不受支持的数据类型是指针。

另请参阅：地址说明

事件日志消息

以下信息涉及发布到主要用户界面中“事件日志”窗格的消息。请参阅有关筛选和排序“事件日志”详细信息视图的服务器帮助。服务器帮助包含许多常见的消息，因此也应对其进行搜索。通常，其中会尽可能提供消息的类型 (信息、警告) 和故障排除信息。

Tag generation failed because the device symbols could not be loaded. | Device = '<ChannelName.DeviceName>'.

错误类型：
错误

可能的原因：
The automatic tag generation operation failed because the server could not access the symbol information on the device.

可能的解决方案：

1. Verify that the device is capable of storing its own symbol information. Some devices do not have this capability, requiring symbol information to be manually exported and accessed by the server.
2. Ensure that the device communication parameters are correctly configured in the server.

Tag generation failed because communications could not be established with the device. | Device = '<ChannelName.DeviceName>'.

错误类型：
错误

可能的原因：
The automatic tag generation operation failed because the server was unable to connect to the device.

可能的解决方案：

1. Ensure that the device communication parameters are correctly configured in the server.
2. Check the physical connection between the server and the device.
3. If connecting to the device via a gateway, verify that the CODESYS Gateway is installed on the same host as the server.

Tag generation failed because an unexpected failure occurred. | Device = '<ChannelName.DeviceName>'.

错误类型：
错误

可能的原因：
The automatic tag generation operation failed due to an unknown reason.

可能的解决方案：

1. Ensure that the device communication parameters are correctly configured in the server.
2. Check the physical connection between the server and the device.
3. Ensure the device is functioning properly.
4. Contact technical support.

Internal Error. An unexpected error occurred. Resetting the PLC connection. | Transaction info = '<Transaction Type and Details>'.

错误类型：
错误

可能的原因：
An unknown error occurred.

可能的解决方案：
Attempt the operation again or contact technical support.

Device discovery failed because an unexpected failure occurred.

错误类型：
错误

可能的原因：
The device discovery operation failed due to an unknown reason.

可能的解决方案：

1. Contact technical support.
2. Check the physical connection between the server, gateway if one is being used, and the devices.
3. Ensure the devices, and gateway if one is being used, are functioning properly.

Error occurred while attempting to write tag. Unable to connect to the device. | Tag address = '<.mystruct.innerstruct.tag>'.

错误类型：
错误

可能的原因：
The server failed to connect to the device.

可能的解决方案：

1. Ensure that the device communication parameters are correctly configured in the server.
2. Check the physical connection between the server and the device.
3. If connecting to the device via a gateway, verify that the CODESYS Gateway is installed on the same host as the server.

Error occurred while attempting to connect to device. Failed to retrieve symbol list from device or file.

错误类型：
错误

可能的原因：

1. The server could not access the symbol information on the device.
2. The symbols do not match between the device and the specified symbol file in the device communication parameters.

可能的解决方案：

1. Ensure that the device communication parameters are correctly configured in the server.
2. Verify that the device is capable of storing its own symbol information. Some devices do not have this capability, and a symbol file must be specified in the device communication parameters.

Internal error occurred while attempting to read tag. | Tag address = '<.mystruct.innerstruct.tag>'.

错误类型：
错误

可能的原因：
The read failed due to an unknown reason.

可能的解决方案：
Contact technical support.

Internal error occurred while attempting to write tag. | Tag address = '<.mystruct.innerstruct.tag>'.

错误类型：
错误

可能的原因：

1. Non-ASCII characters were written to the string tag.
2. The write failed due to an unknown reason.

可能的解决方案：

1. Contact technical support.
2. Only write ASCII characters to the string tag.

Error occurred while attempting to read tag. Unsupported data type or invalid address specified. | Tag address = '<.mystruct.innerstruct.tag>', data type = '<type>'.

错误类型：
错误

可能的原因：

1. The specified data type is not supported.
2. The specified address is not valid.

可能的解决方案：
Ensure that the correct data type and address are specified.

Error occurred while attempting to read tag. The specified tag address was not found on the device. | Tag address = '<.mystruct.innerstruct.tag>'.

错误类型：
错误

可能的原因：
The tag address was not found on the device.

可能的解决方案：

1. Verify that the correct address is specified.
2. Verify that the address exists on the device.

Error occurred while attempting to write tag. The specified tag address was not found on the device. | Tag address = '<.mystruct.innerstruct.tag>'.

错误类型：
错误

可能的原因：
The tag address was not found on the device.

可能的解决方案：

1. Verify that the correct address is specified.
2. Verify that the address exists on the device.

Error occurred while attempting to read tag. The specified server data type is not compatible with the device data type. | Tag address = '<.mystruct.innerstruct.tag>', server data type = '<type>', device data type = '<type>'.

错误类型：
错误

可能的原因：
The specified server data type is not compatible with the device data type.

可能的解决方案：
Change the server data type to one that is compatible with the device data type for this address.

Error occurred while attempting to write tag. The specified server data type is not compatible with the device data type. | Tag address = '<.mystruct.innerstruct.tag>', server data type = '<type>', device data type = '<type>'.

错误类型：
错误

可能的原因：
The specified server data type is not compatible with the device data type.

可能的解决方案：
Change the server data type to one that is compatible with the device data type for this address.

Internal error occurred while attempting to connect to device. The configuration provided is not valid.

错误类型：
错误

可能的原因：
The configuration provided is not valid.

可能的解决方案：
Contact technical support.

Error occurred while attempting to read tag. The array size must match between the server and device. | Tag address = '<.mystruct.innerstruct.tag>', server array size = '<length>', device array size = '<length>'.

错误类型：
错误

可能的原因：
The array size does not match between the server and device.

可能的解决方案：
Specify the same array size for both the server and device.

Error occurred while attempting to write tag. The array size must match between the server and device. | Tag address = '<.mystruct.innerstruct.tag>', server array size = '<length>', device array size = '<length>'.

错误类型：
错误

可能的原因：
The array size does not match between the server and device.

可能的解决方案：
Specify the same array size for both the server and device.

Error occurred while attempting to read tag. The specified tag address has a string length that is larger than the maximum supported by the server. | Tag address = '<.mystruct.innerstruct.tag>', Max length = '<number>' characters.

错误类型：
错误

可能的原因：
The corresponding tag on the device has a string length larger than the maximum supported by the server.

可能的解决方案：
Shorten the string length of the tag on the device to a value that is supported by the server.

Error occurred while attempting to write tag. The specified tag address has a string length that is larger than the maximum supported by the server. | Tag address = '<.mystruct.innerstruct.tag>', Max length = '<number>' characters.

错误类型：
错误

可能的原因：
The corresponding tag on the device has a string length larger than the maximum supported by the server.

可能的解决方案：
Shorten the string length of the tag on the device to a value that is supported by the server.

Data type for the given address is not supported. A tag is not generated for this data point. | Tag address = '<.mystruct.innerstruct.tag>'.

错误类型：
警告

可能的原因：
This is caused by having an unsupported data type in the PLC program.

可能的解决方案：
Verify that the tag is the correct data type in the programming software. Correct or use a different data type (supported by the server) or data the unsupported type contains is not accessible.

The tag could not be added to the server because the address exceeds the maximum length. | Tag address = '<.mystruct.innerstruct.tag>', Max length = '<1024>' characters.

错误类型：
警告

可能的原因：
This is caused by having a tag address in the PLC program that exceeds the maximum length supported by the server.

可能的解决方案：
Restructure the PLC program so that the tag address is shorter than the maximum length.

The tag could not be added to the server because it failed address validation. | Tag address = '<.mystruct.innerstruct.tag>'.

错误类型：
警告

可能的原因：
The tag address is malformed or is not supported by the server.

可能的解决方案：

1. Verify the integrity of the symbol file or make a correction before trying again.
2. Verify or correct the tag address before trying again.

The value read from the string tag was truncated. | Tag address = '<.mystruct.innerstruct.tag>', maximum length = '<number>' characters.

错误类型：
警告

可能的原因：
The value read from the string tag was longer than the buffer size reported by the device.

可能的解决方案：
Restart the server runtime.

The value written to the string tag was truncated. | Tag address = '<.mystruct.innerstruct.tag>', maximum length = '<number>' characters.

错误类型：

警告

可能的原因：
The value written to the string tag was longer than the buffer on the device.

可能的解决方案：

1. Write a value that has a length less than or equal to the buffer on the device.
2. Make the buffer on the device large enough to fit the value being written.

Devices across channels must have unique address (IP address or hostname) and port combinations. | Address = '<number>', Port = '<number>', Overlapping device = '<device>'.

错误类型：
警告

可能的原因：
The address and port combination has already been used for another device.

可能的解决方案：
Change the address and/or port for this device.

Failed to open the symbol file. | File = '<path to file>'.

错误类型：
警告

The file path cannot be empty.

错误类型：
警告

The symbol file was invalid or corrupt. | File = '<path to file>'.

错误类型：
警告

Error opening file for tag database import. | OS Error = '<OS Supplied Message>'.

错误类型：
警告

Error occurred while attempting to write tag. The bit value exceeds the size of the controller data type. | Tag address = '<.mystruct.innerstruct.tag>', bit value = '<bit location>', controller data type size (bytes) = '<size in bytes>'.

错误类型：
警告

Error occurred while attempting to read tag. The bit value exceeds the size of the controller data type. | Tag address = '<.mystruct.innerstruct.tag>', bit value = '<bit location>', controller data type size (bytes) = '<size in bytes>'.

错误类型：
警告

Invalid address. Please enter a valid Logical Address or PLC Name.

错误类型：
警告

Devices across channels must have unique Logical Address/PLC Names. | Address = '<address>', Overlapping device = '<device>'.

错误类型：
警告

Tags generated. | Tag count = <count>.

错误类型：
信息化

索引

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W