

Driver ELIPSE RESTAPIClient

Nome do Arquivo	RESTAPIClient.dll
Fabricante	Elipse Software Ltda
Equipamentos	APIs RESTful
Protocolo	HTTP ou HTTPS
Versão	1.0.3
Última Atualização	10/08/2026
Plataforma	Win32
Dependências	IOKit versão 3.0
Leitura com Superblocos	Não
Nível	0

Introdução

Este Driver implementa o protocolo HTTP ou HTTPS, permitindo que aplicações desenvolvidas pela **Elipse Software** se comuniquem com APIs RESTful, utilizando métodos HTTP padrão e suportando o formato de dados **JSON** (*JavaScript Object Notation*).

Parâmetros de Configuração do Driver

Os parâmetros **[P]** de configuração deste Driver não são utilizados. Todas as configurações são executadas na janela de configurações, mostrada na figura a seguir.

Driver ELIPSE RESTAPIClient v1.00.1

RESTAPIClient

Setup

API URL:

Auth

Routes

Templates

Auth Type: No Auth

Token Type: Expiring (From Endpoint)

Username:

Password:

Endpoint:

☐ Use Fixed Expire

Token Key:

Duration:

Expire Key:

Scale: Minute(s)

Bearer Token:

OK

Cancel

Apply

Aba RESTAPIClient

As opções disponíveis na aba RESTAPIClient estão descritas na tabela a seguir.

Opções disponíveis na aba RESTAPIClient

OPÇÃO	DESCRIÇÃO
API URL	Caminho da API (<i>Application Programming Interface</i> ou <i>Interface de Programação de Aplicações</i>), composta pelo protocolo, HTTP ou HTTPS , pelo domínio, como por exemplo "https://api.exemplo.com", e, quando necessário, por uma porta TCP/IP, como por exemplo "https://api.exemplo.com:8080", definida antes das rotas específicas que identificam os recursos de uma API
Auth	Selecione esta aba para configurar a autenticação de uma API. Para mais informações, consulte a tabela Opções disponíveis na aba Auth
Routes	Selecione esta aba para configurar as rotas específicas de uma API. Para mais informações, consulte a tabela Opções disponíveis na aba Routes

OPÇÃO	DESCRIÇÃO
Templates	Selecione esta aba para configurar os modelos no formato JSON de requisição e resposta de uma API. Para mais informações, consulte a tabela Opções disponíveis na aba Templates

Auth Routes Templates

Auth Type: No Auth
Token Type: Expiring (From Endpoint)

Username:
Password:

Endpoint:
☐ Use Fixed Expire

Token Key:
Duration:

Expire Key:
Scale: Minute(s)

Bearer Token:

Aba Auth

As opções disponíveis na aba **Auth** estão descritas na tabela a seguir.

Opções disponíveis na aba Auth

OPÇÃO	DESCRIÇÃO
Auth Type	Define o tipo de autenticação utilizado por este Driver. As opções disponíveis são No Auth , Basic Auth ou Bearer Token
Token Type	Define o tipo de <i>Bearer Token</i> . Esta opção só é habilitada quando a opção Auth Type está configurada como Bearer Token . As opções disponíveis são Expiring (From Endpoint) , Non-Expiring (From Endpoint) ou Non-Expiring (Provided)
Username	Define o nome de um usuário. Disponível apenas para autenticações dos tipos Basic Auth ou Bearer Token dos tipos Expiring (From Endpoint) ou Non-Expiring (From Endpoint)
Password	Define a senha de um usuário. Disponível apenas para autenticações dos tipos Basic Auth ou Bearer Token dos tipos Expiring (From Endpoint) ou Non-Expiring (From Endpoint)

OPÇÃO	DESCRIÇÃO
Endpoint	Define o <i>endpoint</i> usado para obter um <i>Bearer Token</i> , como por exemplo <code>"/auth/token"</code> . Disponível apenas para a autenticação do tipo Bearer Token dos tipos Expiring (From Endpoint) ou Non-Expiring (From Endpoint)
Token Key	Define uma chave no arquivo do tipo JSON de resposta do <i>endpoint</i> que contém um <i>Bearer Token</i> , como por exemplo <code>"access_token"</code> . Disponível apenas para a autenticação do tipo Bearer Token dos tipos Expiring (From Endpoint) ou Non-Expiring (From Endpoint)
Expire Key	Define uma chave no arquivo do tipo JSON de resposta do <i>endpoint</i> que contém a duração de um <i>token</i> , em segundos, como por exemplo <code>"expires_in"</code> . Disponível apenas para a autenticação do tipo Bearer Token do tipo Expiring (From Endpoint)
Use Fixed Expire	Define um tempo fixo de expiração para um <i>token</i> fornecido manualmente, como por exemplo 3600 segundos. Selecionar esta opção desativa a opção Expire Key . Disponível apenas para a autenticação do tipo Bearer Token do tipo Expiring (From Endpoint)
Duration	Define o tempo de validade de um <i>token</i> , que deve ser um valor numérico. Disponível apenas quando a opção Use Fixed Expire está selecionada
Scale	Define a unidade de tempo utilizada para a opção Duration . As opções disponíveis são Second(s) , Minute(s) , Hour(s) ou Day(s) . Disponível apenas quando a opção Use Fixed Expire está selecionada
Bearer Token	Define um <i>token</i> fixo que é utilizado na autenticação. Disponível apenas para a autenticação do tipo Bearer Token do tipo Non-Expiring (Provided)

Auth

Routes

Templates

Endpoint:

Route Alias:

☐ Auth Required

Add Route

Alias	Endpoint	Has Auth
exemple	/exemple	True

Aba Routes

As opções disponíveis na aba **Routes** estão descritas na tabela a seguir.

Opções disponíveis na aba Routes

OPÇÃO	DESCRIÇÃO
Endpoint	Define os <i>endpoints</i> que são utilizados em requisições. Cada <i>endpoint</i> deve começar com uma barra seguida de um caminho, como por exemplo <code>"/api/usuarios"</code> . Também é possível definir segmentos dinâmicos em um <i>endpoint</i> , que são substituídos por valores configurados via Tags em uma requisição. Estes segmentos devem ser indicados por dois pontos, como por exemplo <code>"/api/empresa/:empresald/usuario/:usuariold"</code>
Route Alias	Define um apelido para uma rota. Este apelido é utilizado posteriormente para vincular Tags de uma aplicação a uma rota específica. O valor desta opção diferencia entre maiúsculas e minúsculas
Auth Required	Esta opção indica que uma rota específica utiliza o método de autenticação configurado na opção Auth Type da aba Auth
Add Route	Adiciona uma rota à tabela de rotas configuradas

Clicar com o botão direito do mouse em uma rota mostra as opções **Edit**, que abre a janela Params Config, e **Delete**, que permite remover a rota selecionada. Clicar duas vezes em uma rota também abre a janela Params Config, mostrada na figura a seguir, que permite configurar os parâmetros específicos de uma rota.

Params Config

Param Key:

Default Value:

Regex Validator:

☐ Is Mandatory

Query Params

Add Param

Key	Regex Validator	Default Value	Mandatory
-----	-----------------	---------------	-----------

Header Params

Add Param

Key	Regex Validator	Default Value	Mandatory
Content-Type	.*		False
Authorization	.*		False

Save and Close

Janela Params Config

As opções disponíveis na janela Params Config estão descritas na tabela a seguir.

Opções disponíveis na janela Params Config

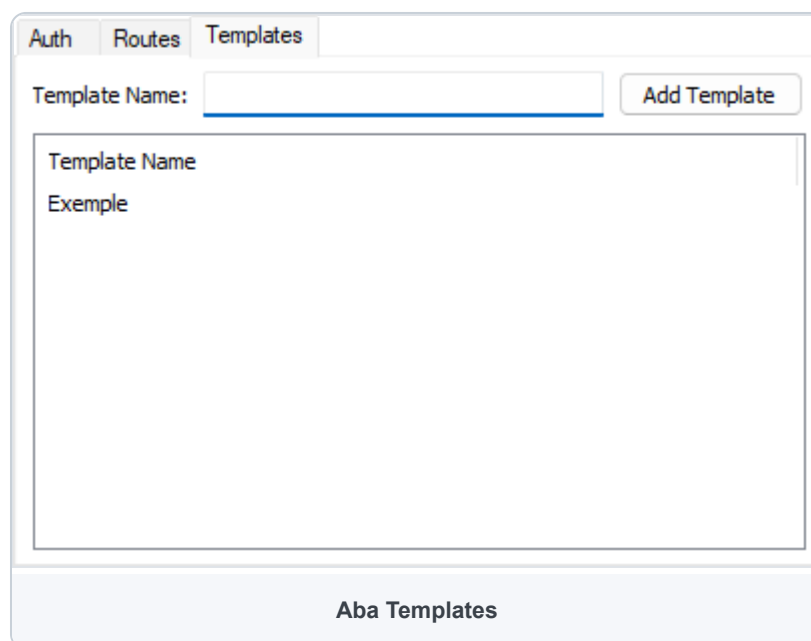
OPÇÃO	DESCRIÇÃO
Param Key	Define o nome de um parâmetro a ser utilizado em uma requisição
Default Value	Define um valor padrão para um parâmetro. Se nenhum valor é fornecido, considera-se esta opção como vazia
Regex Validator	Permite definir uma expressão regular (<i>regular expression</i>) para validar um valor atribuído a um parâmetro, se necessário
Is Mandatory	Selecionar esta opção impede o envio de uma requisição se um parâmetro não está configurado ou está configurado com um valor vazio

OPÇÃO	DESCRIÇÃO
Add Param	Adiciona um novo parâmetro às tabelas Query Params ou Header Params. Cada uma destas tabelas possui a respectiva opção Add Param , que permite inserir um parâmetro de consulta (<i>query</i>) ou de cabeçalho (<i>header</i>), respectivamente
Save and Close	Salva os dados configurados e fecha esta janela

Clicar com o botão direito do mouse em um parâmetro de qualquer uma das tabelas mostra a opção **Delete**, que permite remover o parâmetro selecionado.

Os parâmetros **Content-Type** e **Authorization** são inseridos automaticamente como parâmetros do tipo *header* na tabela Header Params. Estes parâmetros não podem ser excluídos nem configurados posteriormente por meio de Tags.

Em caso de se utilizar um tipo de autenticação diferente das opções disponíveis na aba Auth, como por exemplo usar uma chave de API, uma rota pode ser configurada para não usar autenticação, ou seja, desmarcar a opção Auth Required, e neste caso os parâmetros *header* podem ser configurados, definindo os respectivos valores da coluna **Default Value** tanto manualmente quanto em tempo de execução usando Tags.

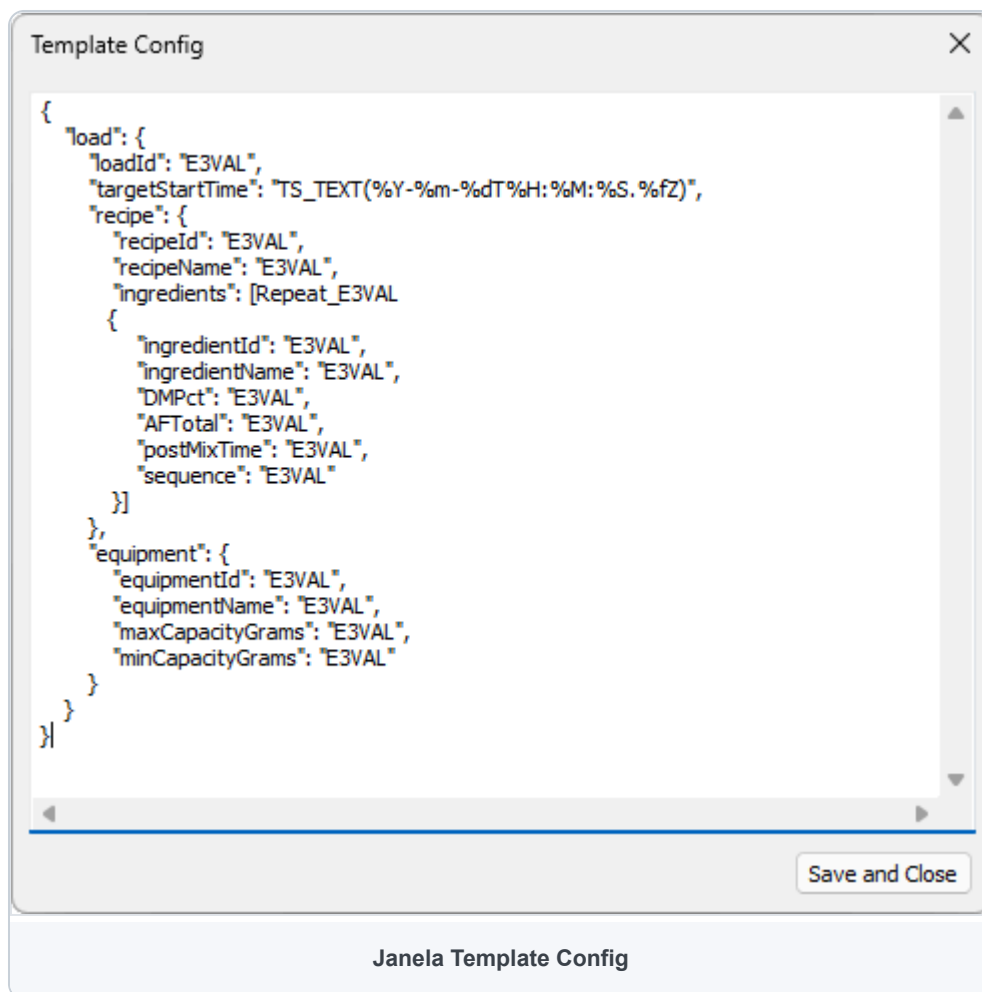


As opções disponíveis na aba **Templates** estão descritas na tabela a seguir.

Opções disponíveis na aba Templates

Opção	Descrição
Template Name	Define o nome de um modelo, no formato JSON , que é utilizado para as requisições e respostas de uma API
Add Template	Adiciona um modelo à tabela de modelos configurados

Clicar com o botão direito do mouse em um modelo mostra as opções **Edit**, que abre a janela Template Config, e **Delete**, que permite remover o modelo selecionado. Clicar duas vezes em um modelo também abre a janela Template Config, mostrada na figura a seguir, que permite configurar um modelo no formato **JSON**, conforme o padrão de sintaxe deste formato descrito no tópico a seguir.



A opção disponível na janela Template Config é **Save and Close**, que salva os dados configurados e fecha esta janela.

Sintaxe de um Modelo JSON

Para extrair ou montar o conteúdo de uma mensagem é necessário declarar um modelo, ou *template*, que permite informar o formato desta mensagem e quais partes devem ser transformadas em dados. Cada *template* deve utilizar palavras-chave colocadas no lugar dos valores a serem extraídos. As palavras-chave disponíveis estão descritas na tabela a seguir.

Palavras-chave disponíveis para templates

PALAVRA-CHAVE	DESCRIÇÃO
TS_TEXT(format)	Estampa de tempo, no formato Texto , que é usada como estampa de tempo de um Tag Bloco ou Tag de Comunicação. O significado de cada campo está descrito na tabela Opções disponíveis para a palavra-chave TS_TEXT
TS_UNIX	Estampa de tempo, no formato de segundos desde 1970, ou formato UNIX . Este valor pode ser um número ou um texto, e é usado como estampa de tempo de um Tag Bloco ou Tag de Comunicação, como por exemplo 1504198675 ou "1504198675"
QL_OPC	Qualidade, no padrão OPC DA, utilizando 1 (um) byte. Este valor é usado diretamente como valor de qualidade em um Tag Bloco ou Tag de Comunicação, sem transformações. Para mais informações, consulte a tabela Padrão OPC DA
QL_BOOL	Qualidade, no padrão Booleano. Se este valor é maior que 0 (zero) ou a expressão "true" ou "TRUE", a qualidade é boa. Se o valor é igual a 0 (zero) ou a expressão "false" ou "FALSE", a qualidade é ruim
E3VAL	Especifica um valor que é extraído na sequência que ocorre para um Tag Bloco, cada valor em um Elemento de Bloco. Se há apenas um valor E3VAL em uma mensagem, o <i>template</i> também pode ser usado com um Tag de Comunicação
DUMMY	Campo variável, mas cujo valor não deve ser enviado para Tags de Comunicação
Repeat_E3VAL	Palavra-chave que só pode ser usada dentro de um <i>array</i> no formato JSON para indicar que os elementos repetidos dentro deste <i>array</i> devem ser processados de forma independente por este Driver, retornando uma leitura para cada conjunto. Um <i>template</i> pode ter esta palavra-chave apenas uma vez

Opções disponíveis para a palavra-chave TS_TEXT

OPÇÃO	DESCRIÇÃO
%a	Dia da semana abreviado, em inglês
%A	Dia da semana completo, em inglês
%b	Mês abreviado, em inglês
%B	Mês completo, em inglês
%C	Século

OPÇÃO	DESCRIÇÃO
%d	Dia do mês, começando com 0 (zero)
%e	Dia do mês, começando com um espaço
%f	Milissegundos, de 0 (zero) a 999
%h	Hora, no formato de 12 horas
%H	Hora, no formato de 24 horas
%m	Mês
%M	Minuto
%p	AM ou PM
%S	Segundos
%y	Ano com dois dígitos
%Y	Ano com quatro dígitos
%Z	Nome do fuso horário, um código internacional que transforma o horário para o formato GMT (<i>Greenwich Mean Time</i>)
%+	Deslocamento ou <i>offset</i> do horário GMT no formato ±HH:MM

O código a seguir contém exemplos de estampas de tempo formatados pela palavra-chave **TS_TEXT**.

```
"2014-07-11T15:26:37Z" -> "TS_TEXT(%y-%m-%dT%H:%M:%SZ)"
"Mon Jul 10 11:04:47 BRT 2017" -> "TS_TEXT(%a %b %d %H:%M:%S %Z %Y)"
"2018-05-02T10:29:28.622-02:00" -> "TS_TEXT(%Y-%m-%dT%H:%M:%S.%f%+)"
```

Padrão OPC DA

Bit	7	6	5	4	3	2	1	0
Descrição	Q	Q	S	S	S	S	L	L

Opções disponíveis para o padrão OPC DA

OPÇÃO	DESCRIÇÃO
QQ	Dois bits de qualidade
SSSS	Quatro bits de sub-status
LL	Dois bits de limite
QQ	Os valores possíveis são 0 : BAD, 1 : UNCERTAIN ou 3 : GOOD
SSSS	Os valores possíveis são 0 : BAD_NONSPECIFIC, 1 : BAD_CONFIGERROR, 2 : BAD_NOTCONNECTED, 3 : BAD_DEVICEFAILURE ou 4 : BAD_SENSORFAILURE
SSSS	Os valores possíveis são 0 : UNCERT_NONSPECIFIC, 1 : UNCERT_LASTUSABLEVALUE ou 4 : UNCERT_SENSORNOTACCURATE
SSSS	Os valores possíveis são 0 : GOOD_NONSPECIFIC, 1 : GOOD_LOCALOVERRIDE ou 6 : GOOD_NONSPECIFICLOCALTIMESTAMP
LL	Os valores possíveis são 0 : FREE, 1 : LOW, 2 : HIGH ou 3 : CONST

Exemplos de Configuração de Modelos no Formato JSON

Um arquivo no formato **JSON** contendo um *array* com múltiplos conjuntos de dados deve incluir a palavra-chave **Repeat_E3VAL** no *template* imediatamente após o colchete que inicia este *array*, conforme o exemplo a seguir.

```
{
  "MessageType": "DUMMY",
  "TagData": [
    Repeat_E3VAL {
      "Temperature": "E3VAL",
      "Humidity": "E3VAL"
    }
  ],
  "MessageDesc": "DUMMY"
}
```

Neste caso, ao receber uma mensagem com 2 (dois) conjuntos de dados, por exemplo, o evento **OnRead** de um Tag Bloco é disparado para cada conjunto, como mostrado no código a seguir.

```
Sub [Sensor-001_OnRead]()
  Application.Trace Item("Temperature").Value
  Application.Trace Item("Humidity").Value
End Sub
```

Se outros valores do tipo **E3VAL** são declarados antes ou depois de um valor do tipo **Repeat_E3VAL**, estes são replicados para cada conjunto, com o valor do tipo **Repeat_E3VAL** sempre como o último valor no Tag Bloco de recebimento, como no exemplo a seguir.

```
{
  "MessageType": "E3VAL",
  "TagData": [
    Repeat_E3VAL {
      "Temperature": "E3VAL",
      "Humidity": "E3VAL"
    }
  ],
  "MessageDesc": "E3VAL"
}
```

Um Tag Bloco com os Elementos de Bloco **Tipo**, **Descrição**, **Temperatura** e **Umidade** pode gerar, a cada evento **OnRead**, valores como no código a seguir.

```
"Teste";"Descrição";11;12 => Primeiro evento OnRead
"Teste";"Descrição";14;66 => Segundo evento OnRead
"Teste";"Descrição";70;55 => Terceiro evento OnRead
```

Caso sem Repetição de Leitura por Conjunto

Se não há necessidade de disparar um evento **OnRead** para cada conjunto, um *template* pode ser declarado sem um valor do tipo **Repeat_E3VAL**, usando apenas uma palavra-chave do tipo **E3VAL** para capturar o *array*, conforme o exemplo a seguir.

```
{
  "MessageType": "DUMMY",
  "TagData": "E3VAL",
  "MessageDesc": "DUMMY"
}
```

Nesse caso, o *array* recebido é expandido com base na quantidade de elementos, resultando em uma única leitura no evento **OnRead**, no formato do exemplo a seguir.

```
array(11, 12); array(14, 66); array(70, 55) => Evento OnRead único
```

Se um *template* contém mais de uma palavra-chave do tipo **E3VAL**, ou seja, não **DUMMY**, o *array* não é expandido, conforme o exemplo a seguir.

```
{
  "MessageType": "E3VAL",
  "TagData": "E3VAL",
  "MessageDesc": "E3VAL"
}
```

A leitura é realizada em um único evento **OnRead**, no formato do exemplo a seguir.

```
"Teste"; array(array(11, 12), array(14, 66), array(70, 55)); "Descrição"=> Evento OnRead único
```

Referência de Tags

Esta seção contém informações sobre a configuração dos Tags PLC e Bloco deste Driver.

Tag Bloco de Recebimento

Somente Leitura

Utilize este Tag Bloco para leitura dos dados recebidos de uma requisição à uma API, configurando os parâmetros conforme a tabela a seguir.

Item	<Route Alias>;<Template Name>
B1	0 (zero)
B2	0 (zero)
B3	0 (zero)
B4	0 (zero)

O tamanho deste Tag Bloco depende do *template* associado e corresponde à quantidade de elementos no arquivo no formato **JSON** cuja palavra-chave seja diferente de **DUMMY**, exceto quando a única palavra-chave diferente de **DUMMY** existente seja um *array*. Neste caso, o *array* é expandido e este Tag Bloco tem o mesmo tamanho do *array* recebido.

NOTA

No caso de um Tag Bloco de Recebimento cujo *template* associado contém a palavra-chave **Repeat_E3VAL**, o conjunto de dados referente a esta palavra-chave vem nas últimas posições deste Tag Bloco.

Tag Bloco de Envio

Somente Escrita

Utilize este Tag Bloco para a escrita dos dados referentes à uma requisição para uma API, configurando os parâmetros conforme a tabela a seguir.

Item	<Route Alias>;<Template Name>
B1	0 (zero)

B2	0 (zero)
B3	0 (zero)
B4	0 (zero)

O tamanho deste Tag Bloco é a soma da quantidade de parâmetros necessários em uma requisição, ou seja, *header*, *route*, *query* e *body*.

Os primeiros Elementos de Bloco correspondem aos parâmetros de *header* adicionados na janela Params Config, exceto os parâmetros **Content-Type** e **Authorization**, que são ignorados.

Em seguida vêm os parâmetros de rota, definidos na aba Routes e associados ao *endpoint*, como por exemplo `/api/empresa:empresald/usuario:usuariold`, em que os parâmetros são **:empresald** e **:usuariold**.

Logo após são definidos os parâmetros de *query*, também configurados na janela Params Config.

Os Elementos restantes deste Tag Bloco são destinados à definição dos dados do corpo (*body*) de uma requisição, ou seja, os valores definidos pelo *template* associado ao Tag Bloco, caso exista.

A quantidade de Elementos de Bloco destinados ao corpo de uma requisição corresponde à quantidade de itens no *template* associado, e todos os Elementos devem ser preenchidos. No entanto, caso o *array* especial **Repeat_E3VAL** esteja presente, este *array* é considerado como apenas um único Elemento de Bloco, diferente do que ocorre no Tag Bloco de Recebimento, onde este *array* é expandido. Além disto, este *array* permanece na posição original do *template* e não é movido para o final deste Tag Bloco.

Para escrever neste *array*, deve-se informar na aplicação um *array* de *arrays*, em que cada elemento representa um conjunto de dados a ser inserido. Cada conjunto é interpretado como um *array* e os valores são associados por posição com os campos definidos no *template*.

Exemplo de um *template*.

```
{
  "TagData": [
    Repeat_E3VAL {
      "Temperature": "E3VAL",
      "Humidity": "E3VAL"
    }
  ]
}
```

Exemplo de escrita do elemento correspondente neste Tag Bloco.

```
array(array(11, 12), array(14, 66), array(70, 55))
```

Exemplo do resultado final, ou seja, o código no formato **JSON** gerado.

```
{
  "TagData": [
    {
      "Temperature": 11,
      "Humidity": 12
    },
    {
      "Temperature": 14,
      "Humidity": 66
    },
    {
      "Temperature": 70,
      "Humidity": 55
    }
  ]
}
```

NOTA

A escrita de dados inválidos nos parâmetros, seja por não atenderem às expressões regulares configuradas ou por apresentarem um tipo de dados incompatível, falha esta escrita.

Tag PLC

Somente Escrita

Utilize um Tag PLC para enviar uma requisição à uma API configurando os parâmetros conforme a tabela a seguir.

Item	<Route Alias>;<Template Name>
N1	1 (um)
N2	Entre 0 (zero) e 3 (três)
N3	0 (zero)
N4	0 (zero)

Qualquer escrita neste Tag executa uma requisição utilizando a rota configurada no parâmetro *Item* e, quando necessário, envia o corpo (*body*) da mensagem com base no *template* associado.

O parâmetro *N2* mapeia o método da requisição HTTP. Os valores possíveis são **0**: Método **GET**, **1**: Método **POST**, **2**: Método **PUT** ou **3**: Método **DELETE**.

NOTA

Se nem todos os elementos obrigatórios para uma requisição estão configurados no Tag Bloco de Envio, esta escrita falha. Da mesma forma, se uma requisição falha por qualquer outro motivo, esta escrita também não é concluída com sucesso.

Documentação das Interfaces de Comunicação

Esta seção contém a documentação das Interfaces de Comunicação referentes ao Driver **RESTAPIClient**.

Esta documentação é compartilhada e mantida no manual do IOKit:

- Configurações de um Driver
- Configurações Gerais
- Configuração de Estatísticas
- Configuração da Interface Ethernet
- Configuração da Interface Modem
- Configuração da Interface RAS
- Configuração da Interface Serial

Histórico de Revisões do Driver

Versão	Data	Autor	Comentários
1.0.3	10/08/2026	A. Fetzner	• Correção de falhas no <i>parsing</i> dos dados no formato JSON e na comunicação usando o protocolo HTTPS (<i>Case 39116</i>).
1.0.1	28/10/2025	A. Fetzner	• Versão inicial deste Driver.

Tópicos Adicionais

Topics not linked from the main table of contents:

- Importação OpenAPI/Swagger

Importação OpenAPI/Swagger

Este Driver oferece a possibilidade de configurar automaticamente todas as rotas, parâmetros, templates e o esquema de autenticação a partir de uma especificação **OpenAPI 3.x** ou **Swagger 2.0**. A especificação pode estar em formato **JSON**.

Como utilizar

1. Abra as propriedades do Driver e selecione a aba **Routes**.

2. Clique em **Import OpenAPI/Swagger...** e selecione o arquivo da especificação.
3. Confirme a substituição. A configuração atual de rotas, templates e autenticação será apagada.
4. Após a confirmação, o Driver cria automaticamente uma rota para cada operação descrita no documento. O alias da rota é gerado a partir do campo **operationId** quando presente, caso contrário utiliza o padrão **<método>_<caminho>**.

Itens importados

- **URL base** a partir de `servers[0].url` (3.x) ou `host+basePath` (2.0).
- **Rotas** para cada operação **GET**, **POST**, **PUT** e **DELETE**. Parâmetros de caminho no formato `{id}` são convertidos para `:id`.
- **Parâmetros** de query e header, com regex automático conforme o tipo declarado.
- **Templates** gerados a partir de `components.schemas` (3.x) ou `definitions` (2.0). Cada campo primitivo é preenchido com E3VAL.
- **Autenticação** detectada a partir de `securitySchemes`. Esquemas suportados: HTTP Basic, HTTP Bearer e OAuth2 (fluxos `password/clientCredentials`). Credenciais e tokens devem ser configurados manualmente após a importação.

Observações

- Para schemas de arrays de objetos, ajuste o template gerado manualmente para utilizar a sintaxe `[Repeat_E3VAL { ... }]` quando a resposta da API for uma lista repetida.
- Esquemas com `oneOf`, `anyOf` ou `allOf` usam apenas o primeiro variante.