

Xlightdata

INSTALAÇÃO E MANUTENÇÃO

ARTUR VINICIUS OLIVEIRA
JOÃO VITOR SZLANDA

FURUKAWA
SOLUTIONS

Xlightdata INSTALAÇÃO



Xlightdata

DESCRIÇÃO

O que é?

É um software utilizado para gravação de inventário de equipamentos GPON.

Equipamentos GPON

- OLT
- ONU
- STM1-GW
- MSDD

The screenshot displays the XLightData 3.0.8 application window. At the top, there's a title bar with the application name and standard window controls. Below the title bar, on the left, is a 'Código de Barras:' label above a text input field. On the right side of the top section, there are four buttons: 'Gerar Inventário', 'Enviar Inventário', 'Ler Inventário', and 'Resetar Campos'. A 'Sair' button with a red arrow icon is also present. Below these buttons is a tabbed interface with six tabs: 'OLT Inventory', 'ONU inventory', 'STM1-GW', 'OLT Blade inventory', 'MSDD inventory', and 'OLT Maple Inventory'. The 'OLT Inventory' tab is currently selected. The main area of the form contains several input fields with labels: 'MAC:', 'Código do produto:', 'Número de Série:', 'Data de Fabricação:', 'Numero do pedido:', 'Descrição do Produto:', 'Anotações Assist. Tec.:', and 'Número de Resets:'. At the bottom, there are two sections: 'Versões' with 'Hardware:' and 'Firmware:' labels and input fields, and 'Pacotes de Software' with 'Startup:', 'System:', and 'AsGOS:' labels and input fields.

Setup

INSTALAÇÃO

Para a instalação é necessário:

- Repositório do XlightData;
- Windows 10 ou superior;
- Descompactador de arquivo.

Link: <https://git.furukawalatam.com/sistemas-da-fio-sr-equipamentos/x-light-data>

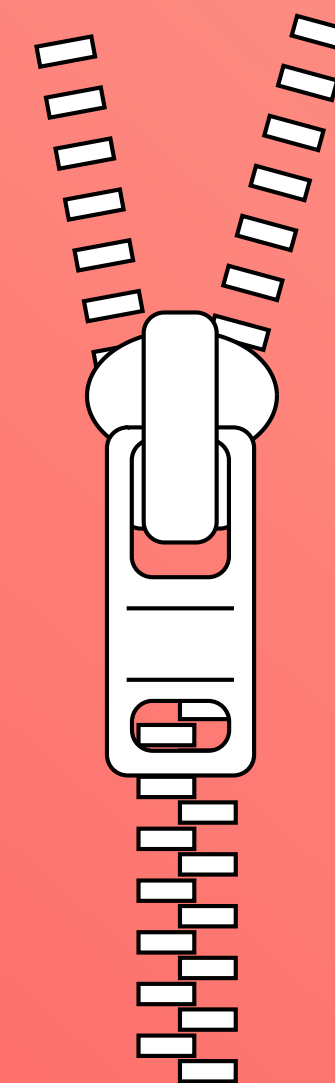


Descompactação

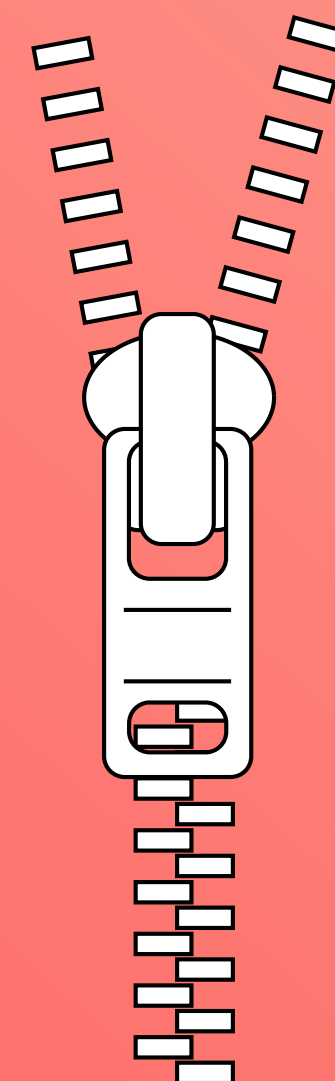
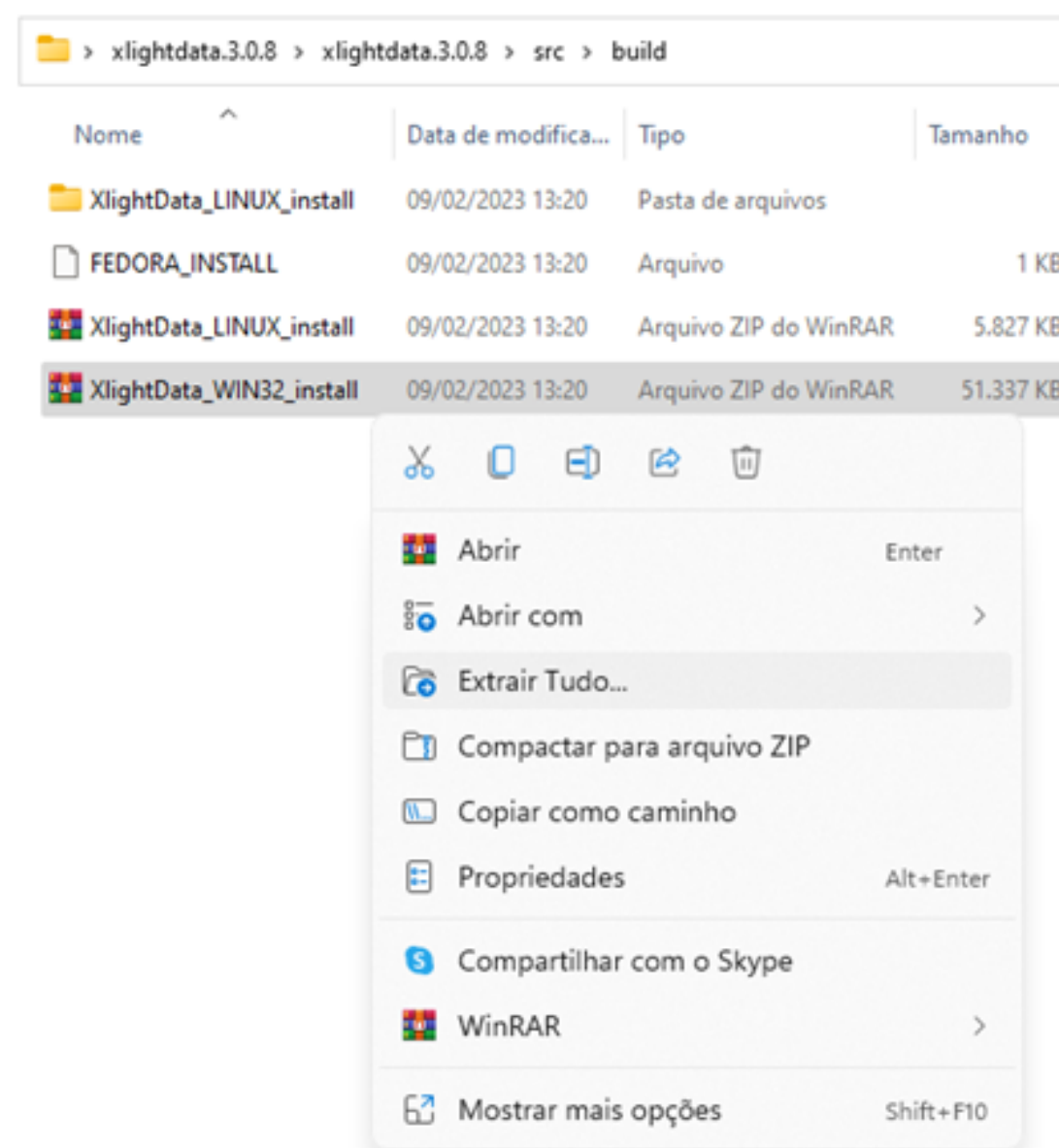
INSTALAÇÃO

Os softwares necessários para o correto funcionamento do XlightData se encontram no diretório “..\xlightdata.3.0.8\xlightdata.3.0.8\setup\XlightData_WIN 32_install”.

Obs.: Pode haver alteração no nome dos diretórios, “..\xlightdata.3.0.8\xlightdata.3.0.8”, de acordo com a versão do software.



Descompactação INSTALAÇÃO



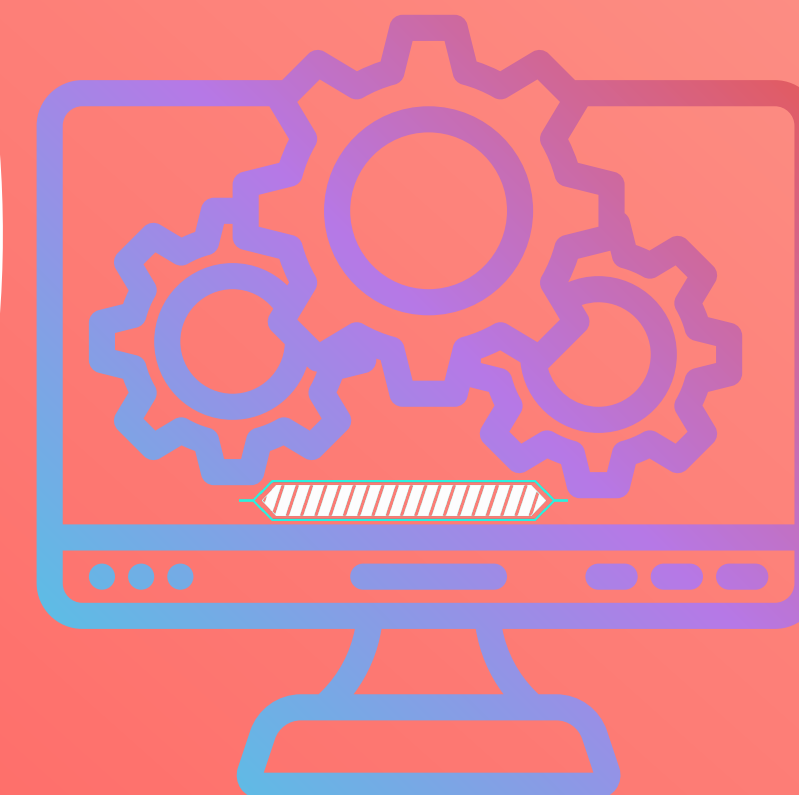
Software INSTALAÇÃO

Após a descompactação, é possível visualizar uma série de arquivos em “..\src\build\XlightData_WIN32_install\XlightData_WIN32_install”.

Realize a instalação dos seguintes arquivos:

- python-2.7.3;
- Firebird-2.5.1.26351_1_Win32;
- kinterbasdb-3.3.0.win32-py2.7;
- pycpg2-2.6.0.win32-py2.7-pg9.4.1-release;
- PyQt-Py2.7-x86-gpl-4.9.4-1.

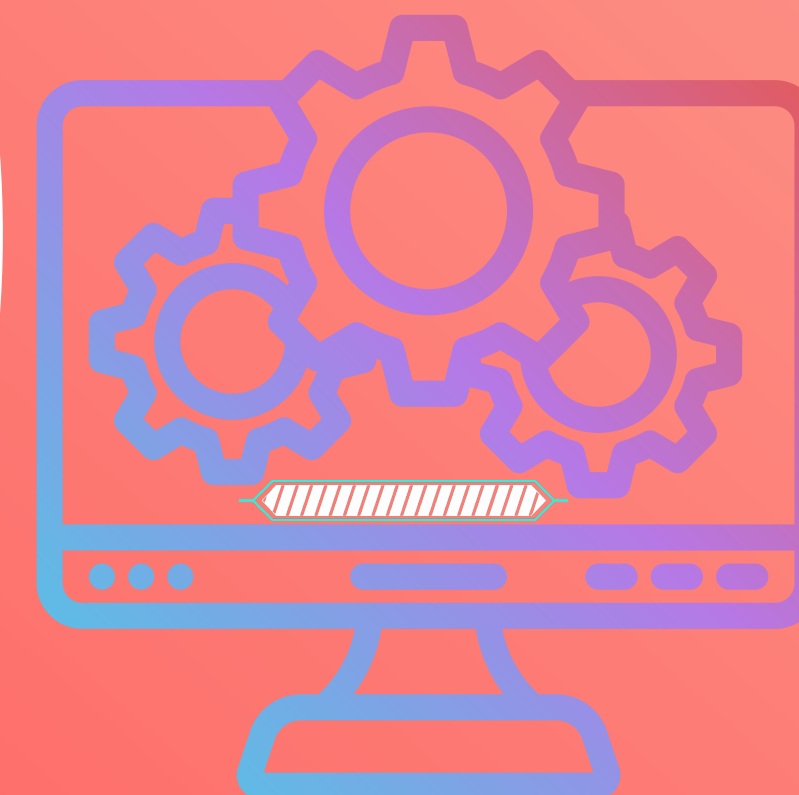
Obs.: É recomendado o Python na versão 2.7.3 para o melhor funcionamento do XlightData. Realize a declaração do Python nas variáveis de ambiente e certifique-se através do comando “python –version”.



Software INSTALAÇÃO

Descrição dos softwares:

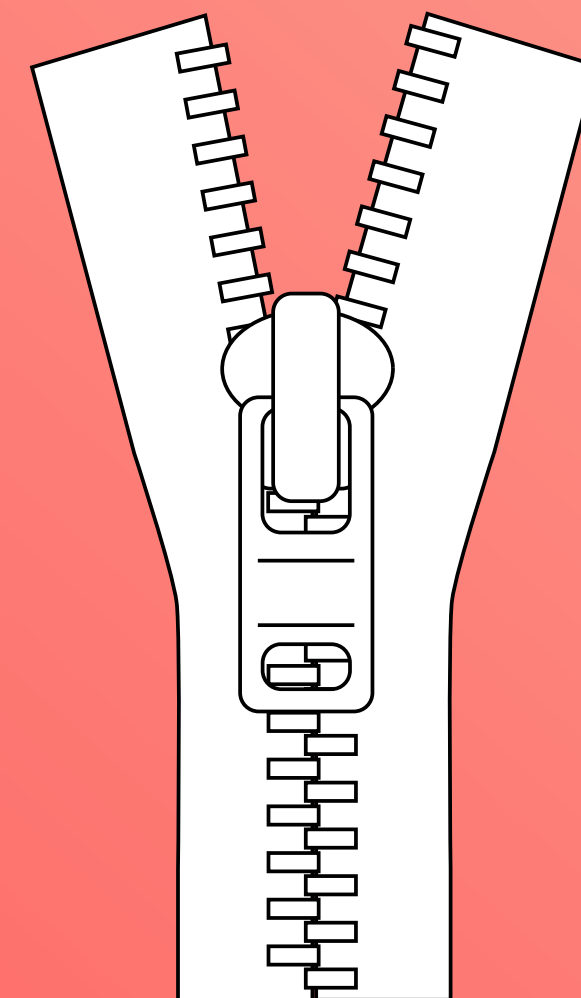
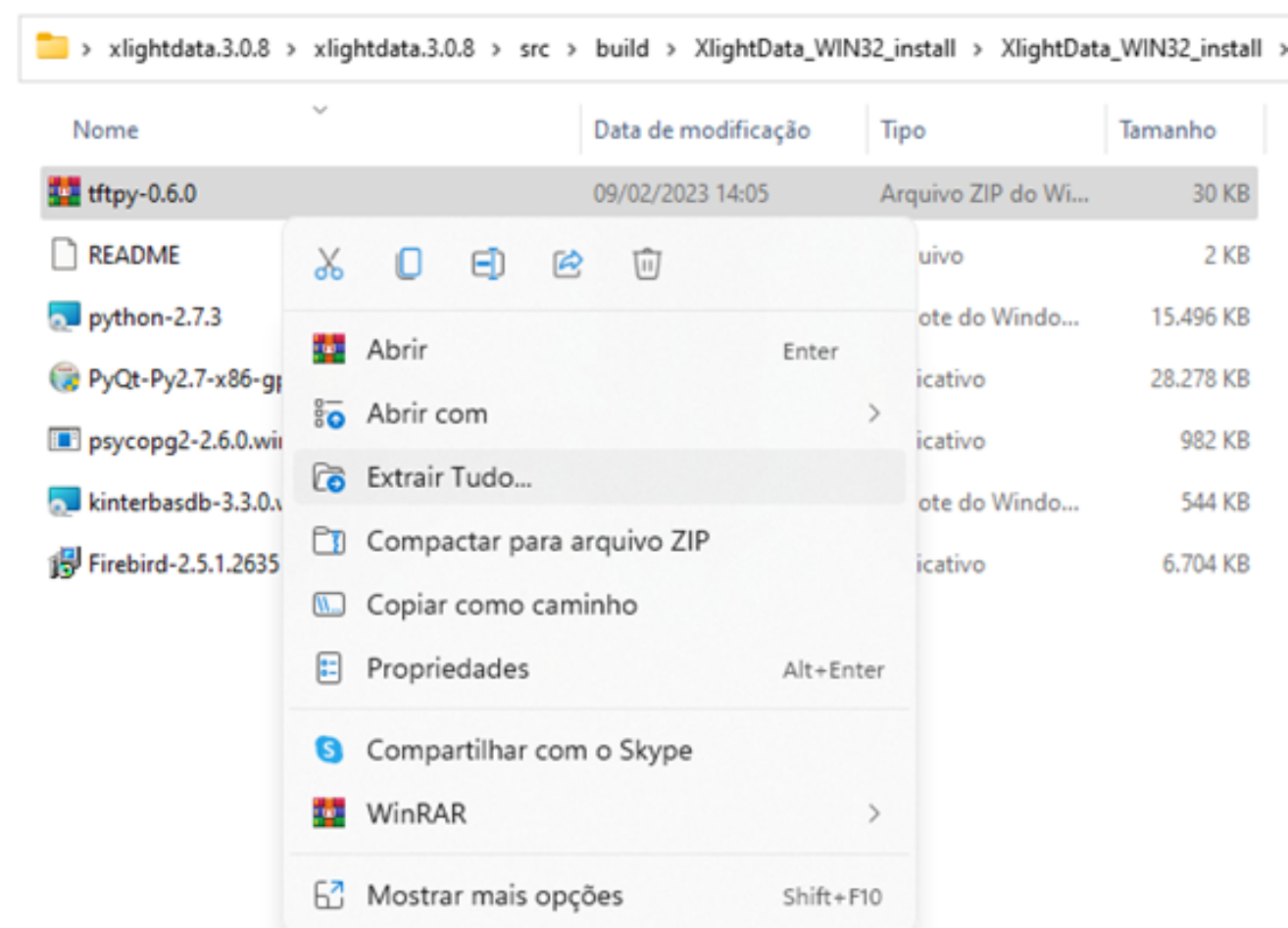
- Python: É uma linguagem de programação alto nível interpretada, que se destaca pela sua simplicidade e legibilidade do código;
- Firebird: Sistema de gerenciamento de banco de dados relacional;
- Kinterbasdb: Módulo de Python que oferece uma interface de programação para se conectar a bancos de dados Firebird;
- Psycopg2: Módulo de Python que oferece uma interface de programação para se conectar a bancos de dados PostgreSQL;
- PyQt: Módulo para desenvolvimento de interfaces gráficas de usuário (GUI) em Python.



Descompactação do TftPy

INSTALAÇÃO

Para o pleno funcionamento do software também é necessário a instalação do Tftpy, responsável pela interação entre cliente e servidor, para isso, realize a descompactação do arquivo “tftpy-0.6.0.zip”.



Descompactação do TftPy

INSTALAÇÃO

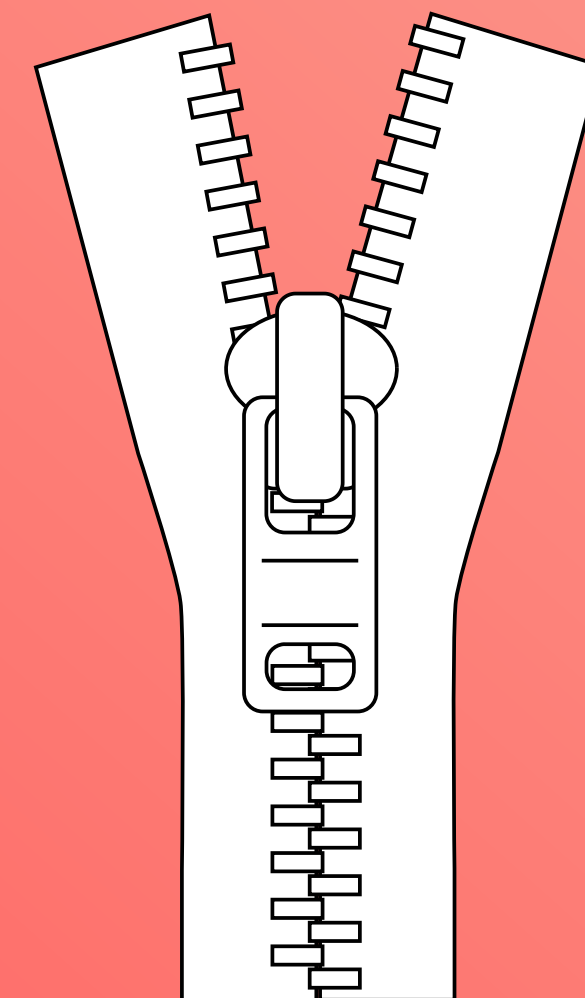
Após a descompactação avance pelos diretórios “tftpy-0.6.0” e execute o script “tftp.bat”.

📁 > xlightdata.3.0.8 > xlightdata.3.0.8 > src > build > XlightData_WIN32_install > XlightData_WIN32_install > tftpy-0.6.0 > tftpy-0.6.0

Nome	Data de modificação	Tipo	Tamanho
📁 tftp	20/12/2012 18:01	Arquivo em Lotes ...	1 KB
📄 setup	24/07/2011 23:08	Arquivo Fonte Pyt...	1 KB
📄 README	24/07/2011 23:09	Arquivo	4 KB
📄 PKG-INFO	24/07/2011 23:13	Arquivo	1 KB
📄 COPYING	02/03/2011 13:59	Arquivo	2 KB
📄 ChangeLog	24/07/2011 23:07	Arquivo	10 KB
📁 tftpy	13/03/2023 13:52	Pasta de arquivos	
📁 bin	13/03/2023 13:52	Pasta de arquivos	

Obs.: Caso o script não funcione como o espera há a alternativa de enviar os seguintes comandos via cmd:

- setup.py clean
- setup.py build
- setup.py install



Rede

CONEXÃO COM O BANCO

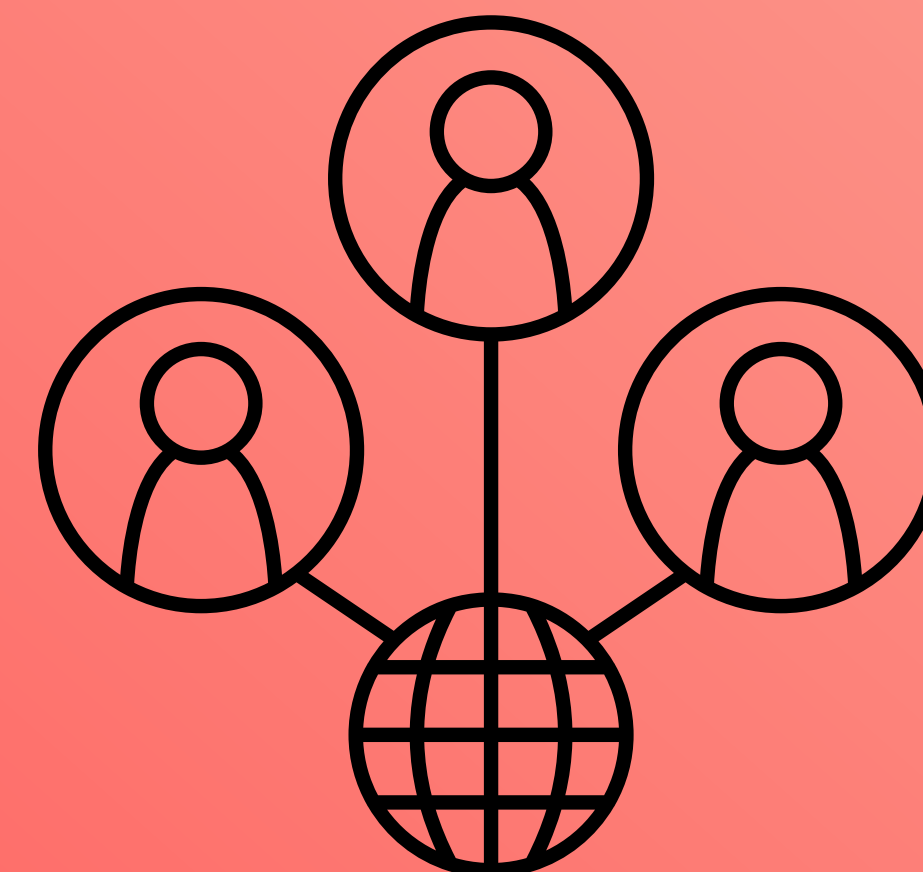
Execute o comando "ping 10.41.122.204" para se certificar da conexão com a rede do banco.

```
C:\Users\artur>ping 10.41.122.204

Disparando 10.41.122.204 com 32 bytes de dados:
Resposta de 10.41.122.204: bytes=32 tempo=3ms TTL=127
Resposta de 10.41.122.204: bytes=32 tempo=4ms TTL=127
Resposta de 10.41.122.204: bytes=32 tempo=5ms TTL=127
Resposta de 10.41.122.204: bytes=32 tempo=5ms TTL=127

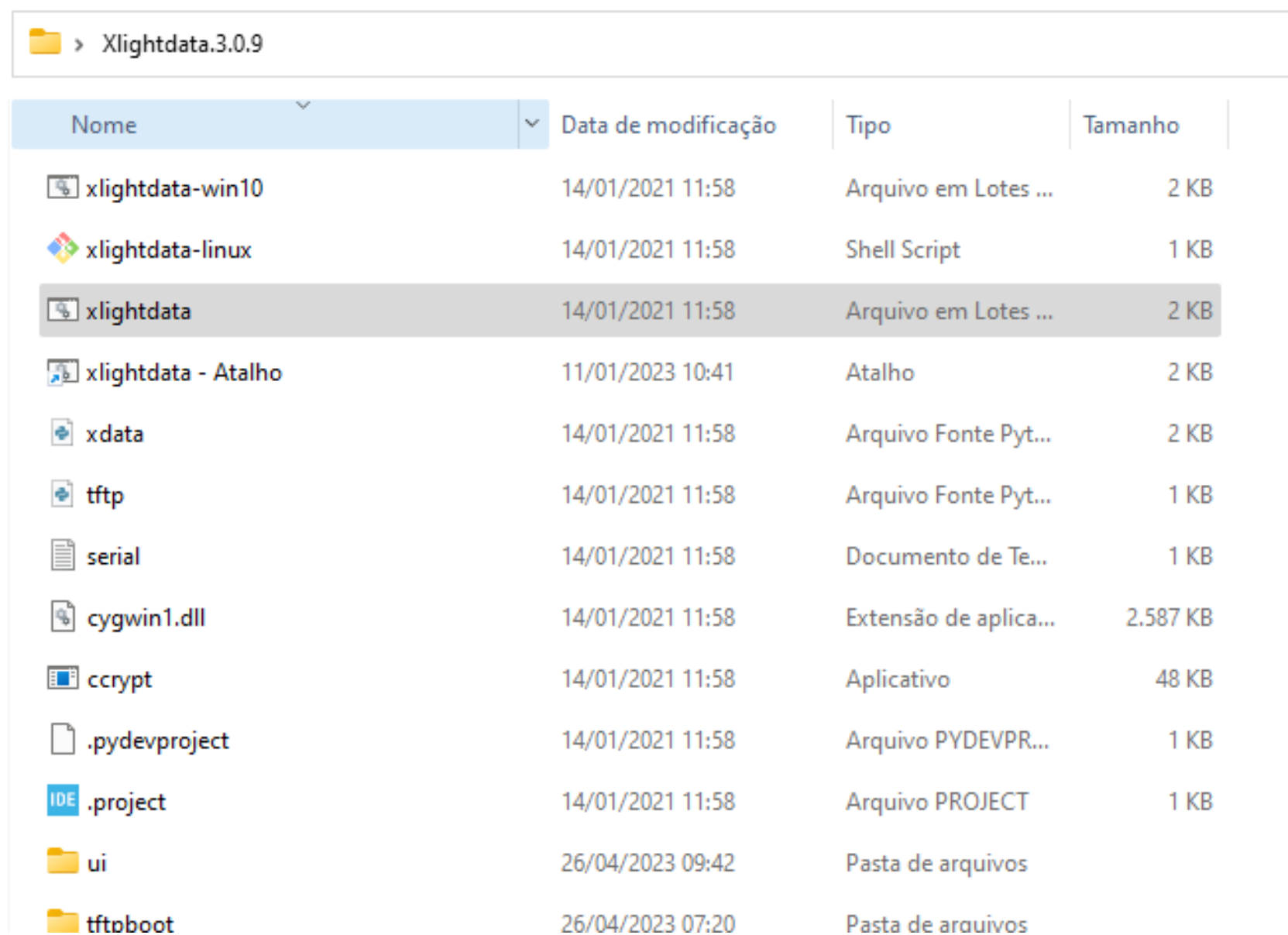
Estatísticas do Ping para 10.41.122.204:
    Pacotes: Enviados = 4, Recebidos = 4, Perdidos = 4 (0% de
    perda),
    Aproximar um número redondo de vezes em milissegundos:
    Mínimo = 3ms, Máximo = 5ms, Média = 4ms

C:\Users\artur>
```



Xlightdata

ABRINDO O XLIGHTDATA



Xlightdata.3.0.9				
Nome	Data de modificação	Tipo	Tamanho	
xlightdata-win10	14/01/2021 11:58	Arquivo em Lotes ...	2 KB	
xlightdata-linux	14/01/2021 11:58	Shell Script	1 KB	
xlightdata	14/01/2021 11:58	Arquivo em Lotes ...	2 KB	
xlightdata - Atalho	11/01/2023 10:41	Atalho	2 KB	
xdata	14/01/2021 11:58	Arquivo Fonte Pyt...	2 KB	
tftp	14/01/2021 11:58	Arquivo Fonte Pyt...	1 KB	
serial	14/01/2021 11:58	Documento de Te...	1 KB	
cygwin1.dll	14/01/2021 11:58	Extensão de aplica...	2.587 KB	
ccrypt	14/01/2021 11:58	Aplicativo	48 KB	
.pydevproject	14/01/2021 11:58	Arquivo PYDEVPR...	1 KB	
.project	14/01/2021 11:58	Arquivo PROJECT	1 KB	
ui	26/04/2023 09:42	Pasta de arquivos		
tftboot	26/04/2023 07:20	Pasta de arquivos		

Welcome

Xlightdata

TELA INICIAL

XLightData 3.0.8

Código de Barras:

01078923862777291112082021000080

Gerar Inventário

Enviar Inventário

Ler Inventário

Resetar Campos

Sair

OLT Inventory | ONU inventory | STM1-GW | OLT Blade inventory | MSDD inventory | OLT Maple Inventory

MAC: 0014FA034E37

Código do produto: 27772

Número de Série: 000080

Número de Série PON: 46524b5701000050

Data de Fabricação: 20/08/2012

Numero do pedido:

Descrição do Produto:

Anotações Assist. Tec.:

Número de Resets:

Versões

Hardware:

Firmware:

Pacotes de Software

System:

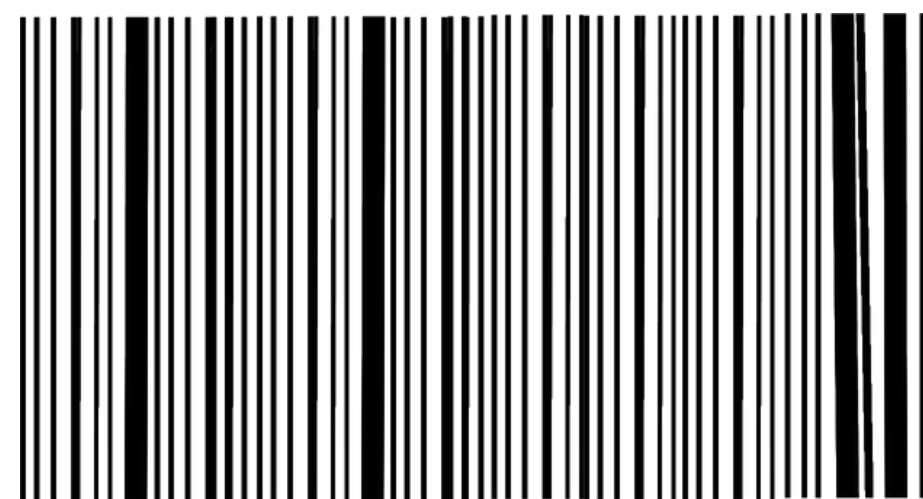
GTIN Cod Bar

GTIN:

Welcome

Xlightdata

CÓDIGO DE BARRAS



EAN

CÓDIGO DE BARRAS

O código EAN possui 13 caracteres que são utilizados para identificação do produto. O código é processado pelo software com a inserção do código de barras (32 caracteres) e após o botão "Enter" ser pressionado. Dessa forma, o Xlightdata realiza a busca do produto.



EAN

CÓDIGO DE BARRAS

XLIGHTData 3.0.8

Código de Barras:

01078923862777291112082021000080

Gerar Inventário

Enviar Inventário

Ler Inventário

Resetar Campos

Sair

OLT Inventory ONU inventory STM1-GW OLT Blade inventory MSDD inventory OLT Maple Inventory

MAC: 0014FA034E37

Código do produto: 27772

Número de Série: 000080

Número de Série PON: 46524b5701000050

Data de Fabricação: 20/08/2012

Numero do pedido:

Descrição do Produto:

Anotações Assist. Tec.:

Número de Resets:

Versões

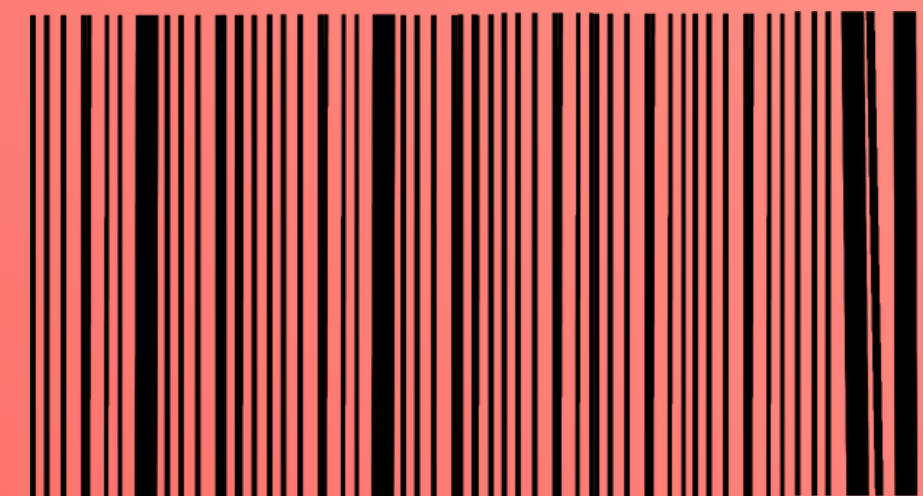
Hardware: Firmware:

Pacotes de Software

System:

GTIN Cod Bar

GTIN:



EAN

CÓDIGO DE BARRAS

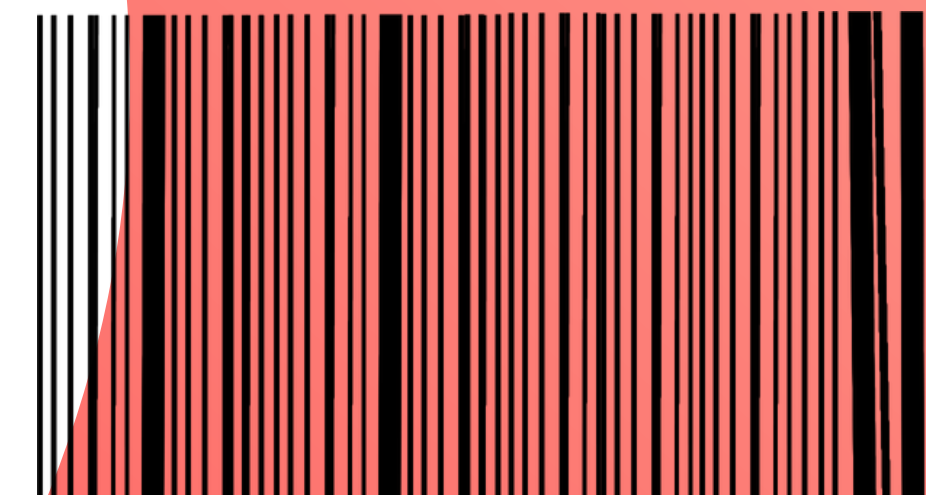
O EAN é extraído do 4º ao 16º número do código de barras.
Existe 4 subdivisões:

- País de origem;
- Empresa fabricante;
- Código do produto;
- Verificador.

Exemplo:

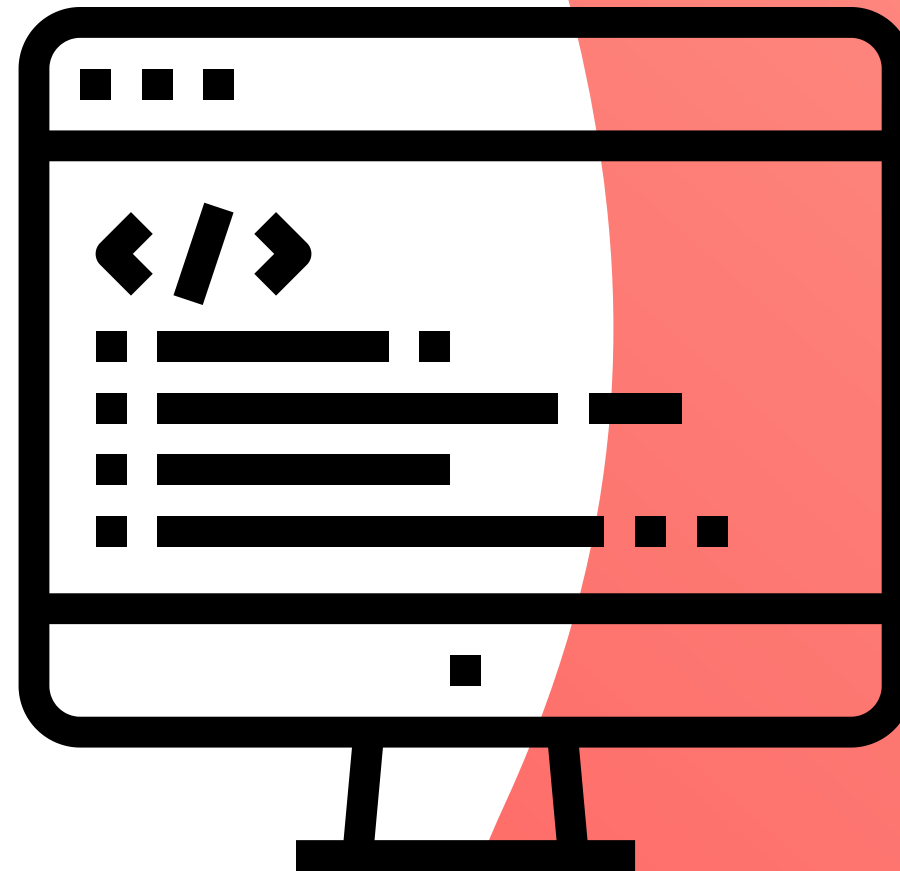
■ 01078923861510311108040121000006

04	05	06	07	08	09	10	11	12	13	14	15	16
7	8	9	2	3	8	6	1	5	1	0	3	1



Xlightdata

CÓDIGO



Setup CÓDIGO

XLightData 3.0.8

Código de Barras:

Gerar Inventário

Enviar Inventário

Ler Inventário

Resetar Campos

Sair

OLT Inventory ONU inventory STM1-GW OLT Blade inventory MSDD inventory OLT Maple Inventory

MAC:

Código do produto:

Número de Série:

Data de Fabricação:

Numero do pedido:

Descrição do Produto:

Anotações Assist. Tec.:

Número de Resets:

Versões

Hardware:

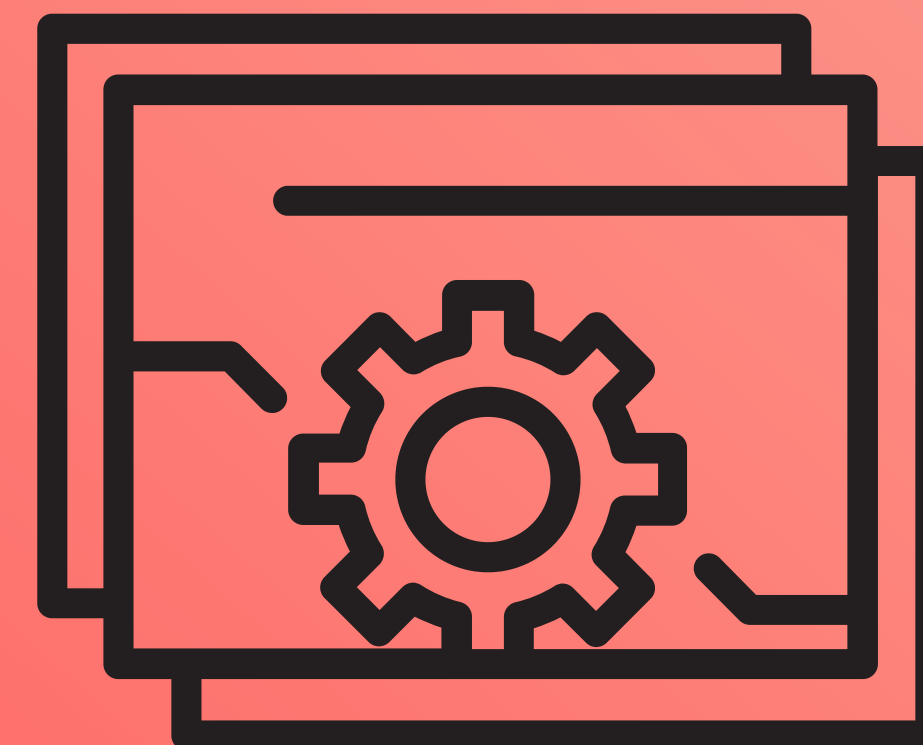
Firmware:

Pacotes de Software

Startup:

System:

AsGOS:



Tab

CÓDIGO | UI

No arquivo "pondata.ui" as definições da interface são definidas.

```

pondata.py 3  pondata.ui X
ui > pondata.ui
431 </widget>
432 <widget class="QWidget" name="TabOnu">
433 <attribute name="title">
434 <string>ONU inventory</string>
435 </attribute>
436 <widget class="QLabel" name="LaOnuData">
437 <property name="geometry">
438 <rect>
439 <x>20</x>
440 <y>160</y>
441 <width>181</width>
442 <height>21</height>
443 </rect>
444 </property>
445 <property name="text">
446 <string>Data de Fabricação:</string>
447 </property>
448 <property name="alignment">
449 <set>Qt::AlignRight|Qt::AlignTrailing|Qt::AlignVCenter</set>
450 </property>
451 </widget>

```

Tab

CÓDIGO | UI

A class "UI_PDMainWindow(object)" é uma das classes mais importantes do Xlightdata. Nela há o método "setupUI", onde os parâmetros da UI são armazenados em atributos Python.

```

src > pondata.py > Ui PDMainWindow > setupUi
201
202     #ONU Inventory fields
203     self.TabOnu = QWidget()
204     self.TabOnu.setObjectName(_fromUtf8("TabOnu"))
205
206     self.LeOnuMac = QLineEdit(self.TabOnu)
207     self.LeOnuMac.setEnabled(False)
208     self.LeOnuMac.setGeometry(QRect(210, 10, 161, 31))
209     self.LeOnuMac.setMaxLength(18)
210     self.LeOnuMac.setObjectName(_fromUtf8("LeOnuMac"))
211
212     self.LeOnuDproduto = QLineEdit(self.TabOnu)
213     self.LeOnuDproduto.setGeometry(QRect(210, 220, 341, 31))
214     self.LeOnuDproduto.setMaxLength(30)
215     self.LeOnuDproduto.setObjectName(_fromUtf8("LeOnuDproduto"))
216
217     self.LeOnuNserie = QLineEdit(self.TabOnu)
218     self.LeOnuNserie.setEnabled(False)
219     self.LeOnuNserie.setGeometry(QRect(210, 80, 161, 31))
220     self.LeOnuNserie.setMaxLength(6)

```

Processamento dos Eventos

CÓDIGO | EVENTOS

Ações em textEdit, botões e no pressionamento do botão "Enter" (quando o cursor no textEdit), chamam métodos para que os eventos sejam processados.

```
pondata.py 3 X
src > pondata.py > Ui_PDMainWindow > setupUi
862
863     self.LeCodbar.textEdited.connect(self.leCodeBarCheckIfDigit)
864     self.LeCodbar.returnPressed.connect(self.calculateBarCode)
865     self.PbWrite.clicked.connect(self.invGenerate)
866     self.PbSend.clicked.connect(self.invGenerateToSend)
867     self.PbClean.clicked.connect(self.clearButton)
868     self.PbRead.clicked.connect(self.invRead)
869     self.PbQuit.clicked.connect(self.exitApp)
870
```

textEdit

CÓDIGO | EVENTOS

Quando alguma ação de escrita é realizada no campo do código de barras o caractere digitado é verificado e só é escrito caso seja um número.

```
self.LeCodbar.textEdited.connect(self.leCodeBarCheckIfDigit)

src > pondata.py > Ui_PDMainWindow

2406         def leCodeBarCheckIfDigit(self):
2407             self.checkIfDigit(self.LeCodbar)

src > pondata.py > Ui_PDMainWindow > checkIfDigit

2418         def checkIfDigit(self, field):
2419             pattern = r'^0-9'
2420             if re.search(pattern,field.text()):
2421                 field.setText(field.text()[:-1])
```


returnPressed

CÓDIGO | EVENTOS

Quando o botão "Enter" é pressionado o processamento do código de barras digitado é realizado.

```
self.LeCodbar.returnPressed.connect(self.calculateBarCode)
```

Dentre os processamentos estão:

- Verificação do tamanho;
- Validação do código;
- Split por categoria;
- Verificação do produto;
- Leitura/Escrita no banco;
- Exibição das infos.

The screenshot shows the XLightData 3.0.8 application window. At the top, there's a 'Código de Barras:' label and a text input field. To the right of the input field are four buttons: 'Gerar Inventário', 'Enviar Inventário', 'Ler Inventário', and 'Resetar Campos'. Below these buttons is a 'Sair' button with a red icon. A tabbed interface below the buttons shows several tabs: 'OLT Inventory', 'ONU inventory', 'STM1-GW', 'OLT Blade inventory', 'MSDD inventory', and 'OLT Maple Inventory'. The 'ONU inventory' tab is currently selected. Under this tab, there are several labels and corresponding input fields: 'MAC:', 'Código do produto:', 'Número de Série:', 'Data de Fabricação:', 'Numero do pedido:', 'Descrição do Produto:', 'Anotações Assist. Tec.:', and 'Número de Resets:'. At the bottom of the window, there are two sections: 'Versões' with 'Hardware:' and 'Firmware:' labels and input fields, and 'Pacotes de Software' with 'Startup:', 'System:', and 'AsGOS:' labels and input fields.

Botão "Gerar Inventário"

CÓDIGO | EVENTOS

Armazena as informações inseridas, realiza um processamento e atribui as definições ao inventário.

```
self.PbWrite.clicked.connect(self.invGenerate)

def invGenerate(self):
    self.removeInvFile()
    gfile = invfile()

    #Verify for ONU with multiple licenses, if ther is any of them checked (L2, L3
    Lite or L3 Full)
    #   name                                PCode
    # ONU LD 582 L2+ L3Lite+ L3Full (DEFINITIVO)    61392
    #-----
    # ONU LD 581 L2+L3 (Fictício)    00033
    '''FIXME: colocar códigos corretos'''

    flag_newOrder = False
    flag_OrderCompleted = False

    if (self.LeOnuCnrod.text() == '61392' or self.LeOnuCnrod.text() == '00033'
```

Botão "Enviar Inventário"

CÓDIGO | EVENTOS

É o botão responsável pelo envio do inventário. Só é habilitado na OLT MAPLE 3508 e 3516.

```
self.PbSend.clicked.connect(self.invGenerateToSend)

def invGenerateToSend(self):
    gSend = invSend()
    # name      PCode
    # OLT MAPLE 3516      7899936006131
    # OLT MAPLE 3508      7893137283605
    if (self.LeOltMapleCprod.text() == '7899936006131'
    or self.LeOltMapleCprod.text() == '7893137283605'):
        if str(self.LeOltMapleMac.text()).__len__() == 0:
            self.showInformDialog('Impossivel gravar inventario. Falta mac-address')
            return None
    retStr = gSend.oltMaple(self)
    if retStr[0] == 'S':
        self.showInformDialog(retStr)
    elif retStr[0] == 'E':
        self.showErrorDialog(retStr)
    elif retStr[0] == 'I':
        self.showAcceptDialog(retStr)
        self.clearFields('all')
```

"Sem conexao com a OLT TimeOut"

"Erro ao gravar inventario"

"Inventario gravado com sucesso"

Botão "Enviar Inventário"

CÓDIGO | EVENTOS

```
class invSend(object):
    def oltMaple(self, ui):
        inv = inventory()
        inv.oltMaple(str(ui.LeOltMapleMac.text()), str(ui.LeOltMapleCprod.text()),
                    str(ui.LeOltMapleNserie.text()), str(ui.LeOltMapleData.text()))

        UDP_IP = '192.168.20.1'
        UDP_PORT = 10000

        t = datetime.datetime.now()

        try:
            sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
            sock.sendto(str(t), (UDP_IP, UDP_PORT))

            sock.settimeout(5)
            ret = sock.recvfrom(1)

            if ret[0] == '0':
                #Mac
                data = inv.mac

                #Code of product
                data = data + str(inv.codprod)

                #Serial Number
                data = data + str(inv.snumber).zfill(6)

                #Manufacturing date
                data = data + inv.dataf
                sock.sendto(str(data), (UDP_IP, UDP_PORT))
                return "Inventario gravado com sucesso"
            else:
                return "Erro ao gravar inventario"

        except socket.timeout:
            return "Sem conexao com a OLT TimeOut"
```

Botão "Ler Inventário"

CÓDIGO | EVENTOS

Quando pressionado se o arquivo "xxxinv.cpt" existe no diretório "tftpboot" e caso exista executa o arquivo "ccrypt.exe" com parâmetros para descriptografar, sobreescrição e leitura da key (-dfk), além da key e do endereço do arquivo.

```
self.PbRead.clicked.connect(self.invRead)
def invRead(self):
    #self.clearFields('all')

    if os.path.isfile('tftpboot/oltinv.cpt') == True:
        if sys.platform == "win32":
            os.system("ccrypt.exe -dfK dp2a3r2mhbolt tftpboot/oltinv.cpt")
        else:
            os.system("ccrypt -dfK dp2a3r2mhbolt tftpboot/oltinv.cpt")
```

Botão "Resetar Campos"

CÓDIGO | EVENTOS

Quando pressionado apaga todos os campos e os arquivos existentes em "tftpboot".

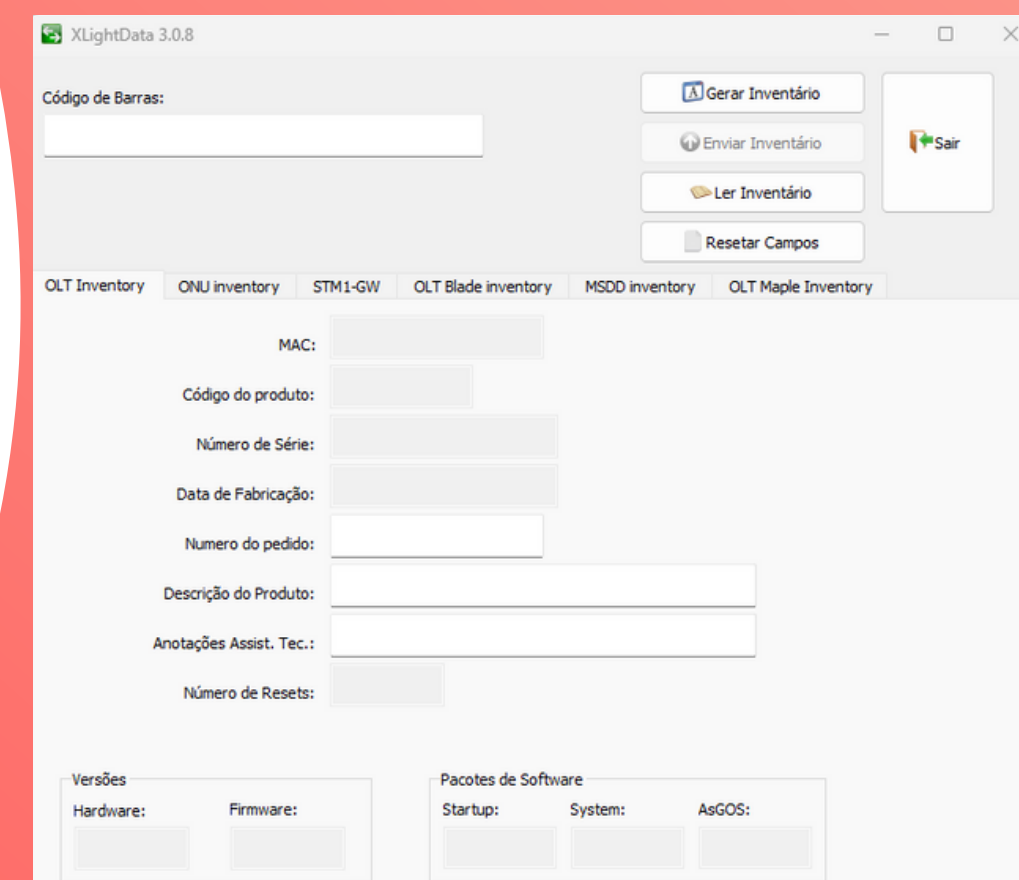
```
self.PbClean.clicked.connect(self.clearButton)
def clearButton(self):
    self.clearFields('all')
    self.removeInvFile()
def removeInvFile(self):
    try:
        os.remove("tftpboot/oltinv.cpt")
    except:
        pass
```

Botão "Sair"

CÓDIGO | EVENTOS

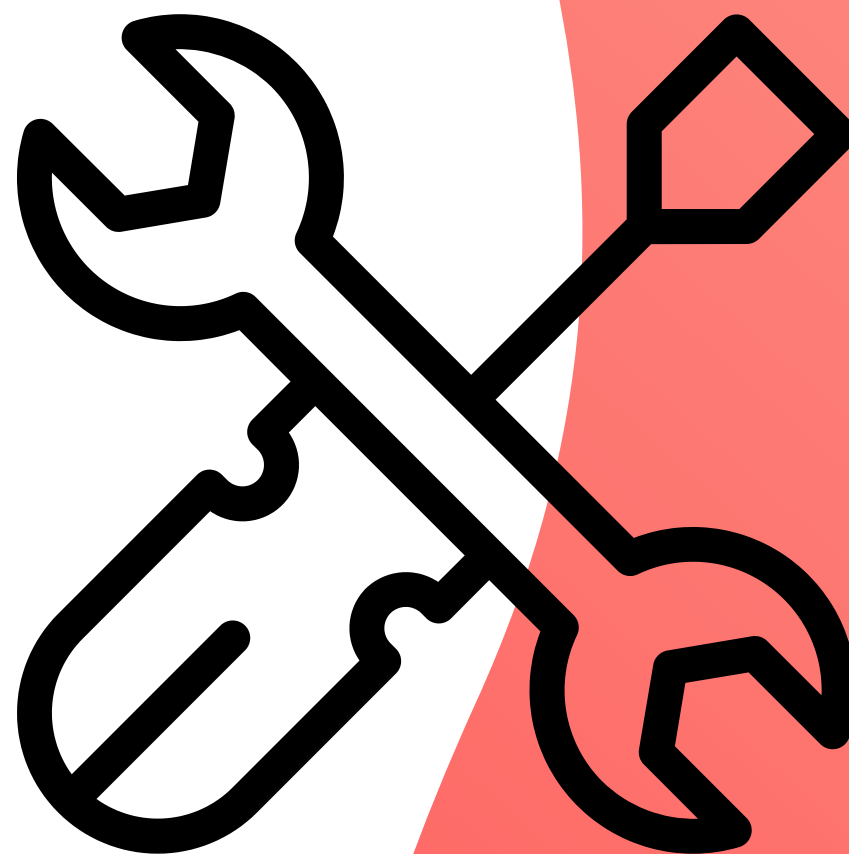
Quando pressionado o botão remove todos os arquivos em "tftpboot" e fecha a aplicação.

```
self.PbQuit.clicked.connect(self.exitApp)
def exitApp(self):
    self.removeInvFile()
    self.PDMainWindow.close()
```



Xlightdata

MANUTENÇÃO E COMPILAÇÃO



Inserção de novos produtos

MANUTENÇÃO E COMPILAÇÃO

De modo geral os novos código devem ser inseridos nos seguintes métodos:

- `def insertEAN(self, ean, ProdCode, isFurukawa, Gtin);`
- `def invGenerate(self);`
- `def calculateBarCode(self):`
- `def verifyProdCode(self,ProdCode);`

Obs.: Não é regra, portanto é necessário avaliar código a código.



Exemplo de novo produto

MANUTENÇÃO E COMPILAÇÃO

Exemplo:

■ 01078923862777291112082021000080

insertEAN:

```
src > asgamac.py > MAC > insertEAN
66     if (ProdCode == '26676' or ProdCode == '26677' or ProdCode == '61952'
67         or ProdCode == '80259' or ProdCode == '80260' or ProdCode == '80262'
68         or ProdCode == '80277' or ProdCode == '80261' or Gtin == '7899936006131'
69         or Gtin == '7893137283605'):
70         n_macs = 20
71     elif (ProdCode == '18094' or ProdCode == '27772' or ProdCode == '56254'
72         or ProdCode == '55982' or ProdCode == '58938' or ProdCode == '56109'
73         or ProdCode == '00033'):
74         n_macs = 4
75     elif (ProdCode == '55980' or ProdCode == '57107' or ProdCode == '80275'):
76         n_macs = 8
77     elif ( ProdCode == '62441' or ProdCode == '62439' or ProdCode == '61728'
78         or ProdCode == '62445' or ProdCode == '62620' or ProdCode == '62448'
79         or ProdCode == '62450' or ProdCode == '00022'):
80         n_macs = 8
81     elif (ProdCode == '61392' or ProdCode == '80273' or ProdCode == '80274'):
82         n_macs = 5
83     elif ( Gtin == '7899936800050' or Gtin == '7899936002782'
84         or Gtin == '7899936802665' or Gtin == '7899936006063'
85         or Gtin == '7899936006865' or Gtin == '789993600610'
86         or Gtin == '7893137319106'):
87         n_macs = 8
88     else:
89         return None
```



Exemplo de novo produto

MANUTENÇÃO E COMPILAÇÃO

Exemplo:

▪ 01078923862777291112082021000080

invGenerate:

```
'''FIXME: colocar códigos corretos'''
elif ( self.LeOnuCprod.text() == '27772' or self.LeOnuCprod.text() == '18094'
      or self.LeOnuCprod.text() == '56254' or self.LeOnuCprod.text() == '55982'
      or self.LeOnuCprod.text() == '58938' or self.LeOnuCprod.text() == '56109'
      or self.LeOnuCprod.text() == '55980' or self.LeOnuCprod.text() == '57107'
      or self.LeOnuCprod.text() == '61392' or self.LeOnuCprod.text() == '62441'
      or self.LeOnuCprod.text() == '62439' or self.LeOnuCprod.text() == '61728'
      or self.LeOnuCprod.text() == '62445' or self.LeOnuCprod.text() == '62620'
      or self.LeOnuCprod.text() == '62448' or self.LeOnuCprod.text() == '62450'
      or self.LeOnuCprod.text() == '00033' or self.LeOnuCprod.text() == '80275'
      or self.LeOnuCprod.text() == '80273' or self.LeOnuCprod.text() == '80274'):
    if str(self.LeOnuMac.text()).__len__() == 0:
        self.showInformDialog('Impossivel gravar inventario. Faltam mac-address')
    return None
```



Exemplo de novo produto

MANUTENÇÃO E COMPILAÇÃO

Exemplo:

▪ 01078923862777291112082021000080

calculateBarCode:

```
elif (ProdCode == '18094' or ProdCode == '27772'  
or ProdCode == '56254' or ProdCode == '55982'  
or ProdCode == '58938' or ProdCode == '56109'  
or ProdCode == '55980' or ProdCode == '57107'  
or ProdCode == '61392' or ProdCode == '62441'  
or ProdCode == '62439' or ProdCode == '61728'  
or ProdCode == '62445' or ProdCode == '62620'  
or ProdCode == '62448' or ProdCode == '62450'  
or ProdCode == '00033' or ProdCode == '80275'  
or ProdCode == '80273' or ProdCode == '80274'):  
    self.PDtabWidget.setCurrentIndex(1)  
    if isFurukawa == 1:  
        self.LeOnuMac.setText(strmac.upper())  
    else:  
        self.LeOnuMac.setText('00'+strmac.upper())
```



Exemplo de novo produto

MANUTENÇÃO E COMPILAÇÃO

Exemplo:

▪ 01078923862777291112082021000080

verifyProdCode:

```
if not (ProdCode == '18094' or ProdCode == '27772'  
    or ProdCode == '56254' or ProdCode == '55982'  
    or ProdCode == '58938' or ProdCode == '56109'  
    or ProdCode == '55980' or ProdCode == '57107'  
    or ProdCode == '28110' or ProdCode == '26676'  
    or ProdCode == '26677' or ProdCode == '61392'  
    or ProdCode == '62441' or ProdCode == '62439'  
    or ProdCode == '61728' or ProdCode == '62445'  
    or ProdCode == '62620' or ProdCode == '62448'  
    or ProdCode == '62450' or ProdCode == '00033'  
    or ProdCode == '61952' or ProdCode == '80259'  
    or ProdCode == '80260' or ProdCode == '80262'  
    or ProdCode == '80277' or ProdCode == '80261'  
    or ProdCode == '80275' or ProdCode == '80273'  
    or ProdCode == '80274'):

    return ProdCode
else :
    return '1'
```



Bibliotecas em bytecode

MANUTENÇÃO E COMPILAÇÃO

Como garantia de eficiência no processamento, deve ser feita a compilação dos arquivos .py para bytecode. O comando utilizado que possibilita a compilação é "python -m compileall". A partir deste comando arquivos .pyc são gerados.

Após modificações nos arquivos .py no diretório "..\src\" execute o comando "python -m compileall" para que novos arquivos .pyc sejam gerados.

Obs.: É recomendado que seja compilado em uma versão 2.x do Python. Diferentes versões geram arquivos com características diferentes.

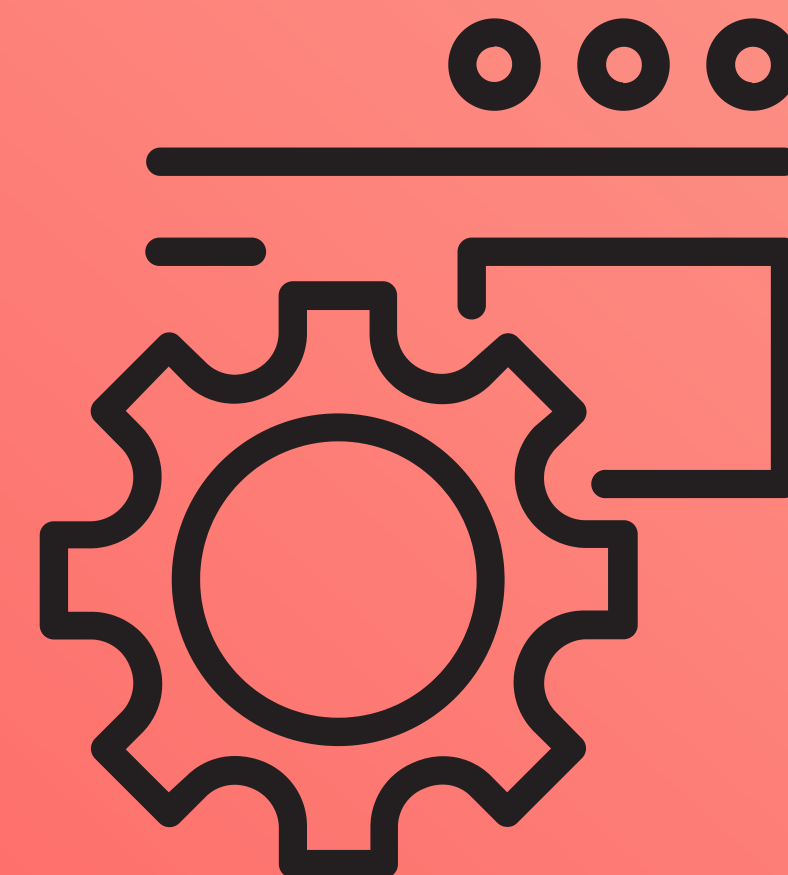


Compilação

MANUTENÇÃO E COMPILAÇÃO

```
PS C:\Users\user\Desktop\xlightdata.3.0.8\xlightdata.3.0.8\src> python -m compileall
Listing 'C:\\Users\\user\\Desktop\\xlightdata.3.0.8\\xlightdata.3.0.8\\src'...
Compiling 'C:\\Users\\user\\Desktop\\xlightdata.3.0.8\\xlightdata.3.0.8\\src\\asgamac.py'...
Compiling 'C:\\Users\\user\\Desktop\\xlightdata.3.0.8\\xlightdata.3.0.8\\src\\pondata.py'...
Listing 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\python311.zip'...
Can't list 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\python311.zip'
Listing 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\__hello__.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_aix_support.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_bootsubprocess.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_collections_abc.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_osx_support.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_py_abc.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_pydecimal.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_pyio.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_sitebuiltins.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\_threading_local.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\abc.py'...
Compiling 'C:\\Users\\user\\AppData\\Local\\Programs\\Python\\Python311\\Lib\\aifc.py'...
```

Obs.: Em caso da falta do pacote execute "pip install compileall2".



Xlightdata

FIM

ARTUR VINICIUS OLIVEIRA
JOÃO VITOR SZLANDA

FURUKAWA
SOLUTIONS