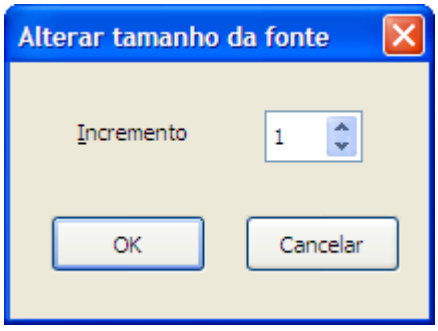
Tutorial

PyUNO
e as caixas de diálogo

por ***Noelson Duarte***

No artigo anterior, apresentei macros Python para aumentar e reduzir o tamanho da fonte do texto selecionado em uma unidade. Hoje, vamos permitir que o próprio usuário defina o valor do incremento.
Num aplicativo dirigido por uma interface gráfica, isto pode não ser uma tarefa trivial. Felizmente, a API do OpenOffice.org possui vários objetos voltados para o desenvolvimento de janelas. Eles estão agrupados no módulo awt. O IDL Reference Guide contém uma descrição detalhada destes objetos.
Em nosso caso, para interagir com o usuário, precisamos de uma caixa de diálogo com, no mínimo, dois controles, um campo para a entrada do valor e um botão de comando. Mas, para os usuários finais, o mínimo é pouco, então vamos projetar um diálogo com mais informações:



A determinação das propriedades do tamanho e posição dos controles é tediosa sem uma ferramenta adequada. Assim, utilizei o editor de diálogos do BrOffice.org – parte do IDE Basic – para desenhar e testar a aparência final da caixa de diálogo. Depois, copiei os valores destas propriedades para o código. Para testar seus próprios diálogos com o editor selecione:
Ferramentas| Macros| Organizar caixas de diálog| Novo| Editar

Inclusive, podemos utilizar diálogos de uma biblioteca Basic, com qualquer uma das linguagens suportadas pelo ambiente de macros do BrOffice.org. Contudo, vamos montar o diálogo usando código Python.

Os controles UNO adotam uma variação do modelo *MVC* (Model – View – Controller), com um objeto para a camada Model – que trata dos dados – e outro para as camadas View / Controller – que trata da apresentação dos dados e da interação com o usuário.
Cada um destes objetos possui diversas propriedades. Algumas são comuns a todos os controles, enquanto outras são específicas. Isto pode ser notado no código da rotina responsável pela caixa de diálogo.

```
# tamanhoFonte.py
# macro Python para o BrOffice.org

# importa constantes da enum
from com.sun.star.awt.PushButtonType import OK, CANCEL

def DialogoAlterar():
    """Exibe um dialogo para obter o valor do incremento do tamanho da fonte."""
    try:
        ctx = XSCRIPTCONTEXT.getComponentContext()
        smgr = ctx.ServiceManager
#
        # cria o modelo do dialogo
        nomeSv = "com.sun.star.awt.UnoControlDialogModel"
        oDlgModel = smgr.createInstanceWithContext(nomeSv, ctx)
        # define algumas propriedades
        oDlgModel.PositionX = 155
        oDlgModel.PositionY = 95
        oDlgModel.Width = 107
        oDlgModel.Height = 79
        oDlgModel.Title = "Alterar tamanho da fonte"
        #
        # cria o modelo do rotulo
        nomeSv = "com.sun.star.awt.UnoControlFixedTextModel"
        tmpModel = oDlgModel.createInstance(nomeSv)
        # define algumas propriedades
        tmpModel.PositionX = 16
        tmpModel.PositionY = 17
        tmpModel.Width  = 41
        tmpModel.Height = 15
        tmpModel.Name = "lblIncremento"
        tmpModel.TabIndex = 0
        tmpModel.Label = " Incremento "
        # insere o modelo do controle no modelo do dialogo
        oDlgModel.insertByName("lblIncremento", tmpModel)
        #
        # cria o modelo do campo numerico
        nomeSv = "com.sun.star.awt.UnoControlNumericFieldModel"
        tmpModel = oDlgModel.createInstance(nomeSv)
        # define algumas propriedades
        tmpModel.PositionX = 65
        tmpModel.PositionY = 15
        tmpModel.Width  = 25
        tmpModel.Height = 16
        tmpModel.Name = "nfIncremento"
        tmpModel.TabIndex = 1
        tmpModel.DecimalAccuracy = 0
```

```
tmpModel.ShowThousandsSeparator = False
        tmpModel.Spin  = True
        tmpModel.Value = 1
        tmpModel.ValueMin = -5
        tmpModel.ValueMax = 5
        tmpModel.HelpText = "Valor a aumentar ou reduzir"
        # insere o modelo do campo numerico no modelo do dialogo
        oDlgModel.insertByName("nfIncremento", tmpModel)
        #
        # cria o modelo do botao OK
        nomeSv = "com.sun.star.awt.UnoControlButtonModel"
        tmpModel = oDlgModel.createInstance(nomeSv)
        # define algumas propriedades do botao OK
        tmpModel.PositionX = 10
        tmpModel.PositionY  = 47
        tmpModel.Width = 40
        tmpModel.Height = 18
        tmpModel.Name = "btnAlterar"
        tmpModel.TabIndex = 2
        tmpModel.Label = "OK"
        tmpModel.PushButtonType = OK
        tmpModel.DefaultButton = True
        # insere o modelo do botao OK no modelo do dialogo
        oDlgModel.insertByName("btnAlterar", tmpModel)
        #
        # cria o modelo do botao CANCELAR
        tmpModel = oDlgModel.createInstance( nomeSv )
        # define algumas propriedades
        tmpModel.PositionX = 60
        tmpModel.PositionY  = 47
        tmpModel.Width = 40
        tmpModel.Height = 18
        tmpModel.Name = "btnCancelar"
        tmpModel.TabIndex = 3
        tmpModel.Label = "Cancelar"
        tmpModel.PushButtonType = CANCEL
        # insere o modelo do botao no modelo do dialogo
        oDlgModel.insertByName("btnCancelar", tmpModel)
        #
        # cria o controle do dialogo e define o seu modelo
        nomeSv = "com.sun.star.awt.UnoControlDialog"
        oDlgControl = smgr.createInstanceWithContext(nomeSv, ctx)
        oDlgControl.setModel(oDlgModel)
        # obtem o modelo do campo numerico
        oNFIncremento = oDlgControl.getControl("nfIncremento").getModel()
        #
        # cria o objeto toolkit
        nomeSv = "com.sun.star.awt.ExtToolkit"
        oToolkit = smgr.createInstanceWithContext(nomeSv, ctx)
        # cria uma janela filha para a tela
        oDlgControl.setVisible(False)
        oDlgControl.createPeer(oToolkit, None)
        #
        # ativa o dialogo e aguarda uma acao num dos botoes
        resp = oDlgControl.execute()
        # obtem o valor do campo numerico
        valorCampo = oNFIncremento.Value
        # elimina o objeto dialogo
        oDlgControl.dispose()
```

```
        #
        # botao OK pressionado e valor <> 0 ?
        if (resp == 1 and valorCampo != 0):
            AlteraTamanhoFonte(valorCampo)
    #
    except Exception,e:
        print str(e)
```

Inicialmente, criamos os modelos da caixa de diálogo e dos seus controles, definindo as suas propriedades. O diálogo funciona como um *container* para os outros objetos, que são inseridos pelas chamadas ao seu método *insertByName*.
Note que usamos o *ServiceManager* para criar o diálogo. Este, por sua vez, é usado para criar os controles – um rótulo, um campo numérico e dois botões de comando – com chamadas ao método *createInstance*.
A seguir, temos a criação do objeto controlador do diálogo e, chamando o seu método *setModel*, definimos qual modelo será monitorado.
Como a variável *tmpModel* foi usada para criar todos os controles e precisamos tratar o valor do campo, guardo o modelo do campo numérico na variável *oNFIncremento*.
Antes da exibição do diálogo, criamos uma janela na tela com uma chamada ao método *createPeer*. Note que passamos o *toolkit* como primeiro argumento. O segundo argumento, *None*, indica que a janela é filha da janela principal(*Desktop*). Finalmente, ativamos o diálogo chamando o método *execute* do controlador.
As constantes *OK* e *CANCEL*, atribuídas à propriedade *PushButtonType* dos botões de comando, definem o comportamento dos mesmos. Neste caso, ao clicar sobre um deles, a caixa de diálogo será desativada e a variável *resp* receberá um valor de retorno (1 => OK e 0 => CANCEL). Isto permite um tratamento posterior, sem a necessidade do uso de um *listener* – um objeto que monitora eventos em outros objetos.
O tratamento de eventos pela API do OpenOffice.org tem dois sabores – *listener* e *handler* – a diferença é que o handler pode ou não consumir o evento, enquanto o listener permanece ativo até que seja removido do cadastro. Os leitores ansiosos por um exemplo de tratamento de eventos, em Python, podem transferir a documentação PyUNO e analisar o arquivo *dynamicDialog* da pasta *scripts*.
Segue o restante do código fonte, que lida com a alteração do tamanho da fonte e foi apresentado no primeiro artigo da série.

```
    def AlteraTamanhoFonte(valor):
"""" adiciona o valor ao tamanho da fonte do texto selecionado"""
# obtem o modelo do documento
oDoc = XSCRIPTCONTEXT.getDocument()
# obtem a selecao
oSel = oDoc.getCurrentSelection()
# objetos como gráficos, molduras e tabelas podem ser selecionados
# mas apenas o objeto TextRanges nos interessa
if (oSel.supportsService("com.sun.star.text.TextRanges")):
    # visita um a um o conteudo selecionado
    for i in range(oSel.getCount()):
        oCur = oSel.getByIndex(i)
        oEnum = oCur.createEnumeration()
        while (oEnum.hasMoreElements()):
            oTxt = oEnum.nextElement
```

```
        if (oTxt.supportsService("com.sun.star.text.Paragraph")):
                    oEnum2 = oTxt.createEnumeration()
                    while (oEnum2.hasMoreElements()):
                        oTxtParte = oEnum2.nextElement()
                        tamFonte = oTxtParte.getPropertyValue("CharHeight") +
valor
                        oTxtParte.setPropertyValue("CharHeight", tamFonte)
    return None

def IncrementaTamanhoFonte():
    """ adiciona 1 ao tamanho da fonte do texto selecionado"""
    AlteraTamanhoFonte(1)
    return None

def DecrementaTamanhoFonte():
    """ adiciona -1 ao tamanho da fonte do texto selecionado"""
    AlteraTamanhoFonte(-1)
    return None

# define as funcoes visiveis na interface grafica do BrOffice.org
g_exportedScripts = (DialogoAlterar, IncrementaTamanhoFonte,
DecrementaTamanhoFonte)
```

Note que inclui a função *DialogoAlterar* na variável *g_exportedScripts*, deste modo ela pode ser disparada pelo diálogo *Seletor de Macros* (Ferramentas | Macros | Executar macro).
Por hoje é só, as nossas macros PyUNO estão prontas. No próximo artigo, veremos como transformá-las num suplemento (add-on), com uma maior integração na interface gráfica do BrOffice.org.