Módulo TQSEAG

O Módulo TQSEag permite o desenvolvimento de programas de dentro dos editores gráficos, com acesso aos desenhos e janelas de edição durante a sessão gráfica. Naturalmente usamos no TQSEag os módulos já mostrados para manipular desenhos (TQSDwg) e janelas (TQSJan). Inicialmente a interface de usuário para chamada de programas está restrita aos menus suspensos (*dropdown menus*).

Embora o Editor de Aplicações Gráficas TQS (EAG) seja um único programa executável, EAGW.EXE, o editor é estendido para diferentes aplicações, com diferentes comandos e menus. Atualmente existem mais de 20 aplicações diferentes de edição. A programação do editor é comandada por menus, e a programação em Python, da mesma maneira, é chamada conforme a aplicação. Mostraremos antes como são chamados os menus.

Como uma aplicação do EAG é escolhida

Todo desenho TQS tem os atributos de sistema e subsistema definidos. Estes atributos são acessados pelas propriedades de TQSDwg:

```
settings .systemId
```

Número do sistema

```
settings .subSystemId
```

Número do subsistema

Quando damos um duplo clique em um desenho no gerenciador, o editor EAG será chamado e um menu de aplicação carregado conforme as propriedades acima. Eis alguns menus disponíveis:

Menu	Sistema TQSDwg.	Sub	Aplicação
EAG.MEN	IEDOUT	1	Desenho qualquer
EAGEDP.MEN	IEDOUT	4	Layout de plantas
EAGFOR.MEN	IEDFOR	2	Planta de formas
EAGGRE.MEN	IEDFOR	5	Edição de desenhos de grelhas
EAGME.MEN	IEDFOR	11	Modelador Estrutural
EAGSCAT.MEN	IEDFOR	14	Seções catalogadas
EAGFER.MEN	IEDLAJ	4	Armaduras de lajes
EAGPIL.MEN	IEDPIL	3	Editor de seções de pilares
EAGTELS.MEN	IEDTAB	5	Planta de telas
EAGTESQ.MEN	IEDTAB	11	Esquema de telas soldadas
EAGMCRUZP.MEN	IEDMAD	1	Edição de formas de madeira

EAGMAD.MEN	IEDMAD	2	Desenho de painéis de lajes
EAGELMS.MEN	IEDMAD	6	Desenho de painéis de vigas
EAGALV.MEN	IEDALV	1	Alvenaria em planta
EAGELV.MEN	IEDALV	12	Alvenaria em elevação
EAGADG.MEN	IEDALV	3	Diversas funções de alvenaria
EAGFAC.MEN	IEDPRE	5	Fachadas pré-moldadas
EAGPOR.MEN	IEDPOR	1	Visualizador de pórtico/Grelha
EAGPRCA.MEN	IEDPAR	1	Entrada gráfica de Paredes de concreto
EAGELVPC.MEN	IEDPAR	2	Elevação de Parede de concreto
EAGGDG.MEN	IEDPAR	3	Diversas funções de alvenaria
EAGELVPC.MEN	IEDPAR	17	Paredes de concreto

O que é um menu de editor? É um arquivo de script, que define as interfaces gráficas de menus e barras de ferramentas de botões, acionando comandos (Os menus tipo Ribbon tem uma programação à parte). Todos os menus do editor são armazenados na pasta `TQSW\EXEC\EAGMENU`. Os comandos são funções com assinatura padrão, dentro de DLLs C++. O arquivo .MEN é dividido em seções. A primeira seção declara as DLLs que serão chamadas:

```
[DLL]
EAGME
```

Nesta declaração, temos a seção `[DLL]` e a declaração da EAGME.DLL. As DLLs ficam obrigatoriamente na pasta TQSW\EXEC. Em seguida, temos a seção `[CMD]` de comandos, num formato como este:

```
[CMD]
ID_ARQUIVO_SALVAR,EAGME,aplic_arquivosalvar
Salvar desenho na tela em arquivo DWG
ID_ARQUIVO_SALVARCOMO,EAGME,aplic_arquivosalvar
Salvar o desenho atual com um nome a ser fornecido
```

Um comando é formato por 4 campos em duas linhas: na primeira linha temos um identificador arbitrário que será usado em outros lugares, o nome da DLL, e o nome de uma função C++ dentro da DLL que será chamada se o comando for acionado. Na segunda linha temos uma mensagem de ajuda que aparece em tooltips e na área de status do editor.

Finalmente, os identificadores entram na descrição de menus e barras de ferramentas, dentro da seção `[MENU]`:

```
[MENU]
SUBMENU,&Arquivo
MENUITEM,ID_ARQUIVO_SALVARCOMO,Salvar &DWG
```

```
MENUITEM,ID_ARQUIVO_SALVARCOMOIMAGEM,Salvar como ima&gem
SEPARADOR
SUBMENU,&Utilidades
MENUITEM,ID_ARQUIVO_UTILIDADES_PURGEBLK,Limpar blocos
FIMSUBMENU
FIMSUBMENU
```

O submenu "Arquivo" entrará no final da lista de menus, e os itens restantes serão pendurados abaixo. Os identificadores apontam para os comandos dentro das DLLs, e assim elas são chamadas.

Mostramos em linhas gerais a programação dos editores por DLLs em C++. A programação em Python é semelhante.

Como uma extensão em Python é carregada

Junto com os módulos para interface Python são distribuídos exemplos. Usaremos o menu `EAGFor.PYMEN` para mostrar a programação do editor.

Após a carga de um menu tipo .MEN, o EAG verifica se existem menus Python, e se existirem, os carrega no final da lista de menus. O nome do menu Python é o mesmo do arquivo .MEN carregado, mas com tipo .PYMEN. A localização do arquivo é a pasta `TQSW\EXEC\Python`.

O arquivo `EAGFor.PYMEN` está na pasta `\TQSW\EXEC\Python`. Isto significa que quando o menu `EAGFOR.MEN` for carregado, o EAG tentará a carga do `EAGFor.PYMEN`. Este arquivo será carregado sempre que um desenho de formas for lido. Para gerar um desenho de formas e chamar este menu, vá no "TQS Formas", "Processar", "Geração de desenhos", "Planta de formas". Depois de gerado, edite o desenho com nome tipo `FORnnnn.DWG`.

Chamada do programa Python pelo editor

O menu `EAGFor.PYMEN` de exemplo tem estrutura semelhante aos dos outros menus:

```
[PYTHON]
EAGFOR.PY
[CMD]
ID_PYTHON_CMD1,EAGFOR.PY,aplic_cmd1
Teste de geração de desenhos
ID_PYTHON_CMD2,EAGFOR.PY,aplic_cmd2
Acionamento do editor por identificadores de comando
ID_PYTHON_CMD3,EAGFOR.PY,aplic_cmd3
Teste de interação
ID_PYTHON_CMD4,EAGFOR.PY,aplic_cmd4
Teste de seleção e movimentação de elementos
ID_PYTHON_CMD5,EAGFOR.PY,aplic_cmd5
Mostrar os estados do editor
[MENU]
SUBMENU,Teste Python
```

```
MENUITEM,ID_PYTHON_CMD1,&Geração de desenho
MENUITEM,ID_PYTHON_CMD2,&Comandos do editor
SEPARADOR
MENUITEM,ID_PYTHON_CMD3,&Entrada de pontos
MENUITEM,ID_PYTHON_CMD4,&Seleção e movimentação
SEPARADOR
MENUITEM,ID_PYTHON_CMD5,Estados do edito&r
FIMSUBMENU
```

A diferença é que agora em vez do nome de DLLs, temos a seção `[PYTHON]`, com nomes dos diversos módulos em Python chamados, e nos identificadores, temos o nome de um módulo seguido do nome da função que tratará o comando. Esta função tem uma assinatura padrão que mostraremos em seguida.

Rotinas Python chamado pelo .pymen

Vemos acima a seção `[MENU]` com nome "Teste Python", declarando 5 comandos (com dois separadores entre eles). No editor de planta de formas `(EAGFOR)`, o resultado será como este:

A declaração da primeira linha deste menu é:

```
MENUITEM,ID_PYTHON_CMD1,&Geração de desenho
```

O sinal "&" na frente da letra "G" em "Geração de desenho", simplesmente faz com que o editor sublinhe esta letra, e aceite as teclas `<Alt><G>` de atalho para acionamento deste menu, quando estiver aberto.

Quando o usuário acionar este comando, entrará em ação o identificador:

```
ID_PYTHON_CMD1
```

Este por sua vez foi declarado na seção `[CMD]` como:

```
ID_PYTHON_CMD1,EAGFOR.PY,aplic_cmd1
Teste de geração de desenhos
```

Isto fará com que a rotina `aplic_cmd1 seja` chamada do módulo Python `EAGFOR.PY`, necessariamente localizado na pasta `\TQSW\EXEC\Python`. Neste arquivo de exemplo, está é a rotina:

```
def aplic_cmd1 (eag, tqsjan):
TSTDWG.Test(tqsjan.dwg)
tqsjan.ZoomTotal()
```

A assinatura da rotina é comum à todas as chamadas que o editor faz em Python: o primeiro parâmetro é um objeto da classe `TQSEag.Eag`, que permite acesso ao editor gráfico; o segundo parâmetro é da classe `TQSJan.Window` (já mostrado), que permite a manipulação de um desenho em uma janela Windows. Os dois parâmetros são separados, pois o editor gráfico tem uma arquitetura multidocumento e multijanela. A janela e desenho recebidos pela rotina são os ativos no momento da chamada.

Complementado, para manipular o desenho associado à janela, basta obter o objeto da classe `TQSDwg.Dwg` a partir de tqsjan:

```
dwg = tqsjan.dwg
```

Assim, o primeiro comando da rotina acionou o teste inteiro `TSTDWG.py`, passando o dwg aberto. Veja que este teste pode rodar tanto dentro quanto fora do editor gráfico para gerar desenhos.

Em linhas gerais, dentro do editor gráfico, você pode manipular o desenho editado, lendo e gravando. Você pode controlar a janela gráfica do editor, e chamar programas da interface do editor, `TQSEag` para interagir com o usuário.

Pré-entrada de dados e o TQSEag.entry

O objeto tipo `TQSEag.Eag` tem um subobjeto `entry` da classe `TQSEag.Entry`, que serve para simular a entrada de dados do usuário. A ideia é simular a entrada de coordenadas e teclas por parte do usuário para em seguida acionar comandos disponíveis do editor gráfico.

É importante ressaltar que os comandos usam os dados definidos previamente – então, primeiro alimentamos as respostas esperadas por um comando a ser chamado, depois chamamos o comando.

Por exemplo, para uma definir uma linha entre dois pontos, acionamos o comando de desenho de linhas, entramos com dois pontos e terminamos com `<Enter>`. Isto pode ser simulado como abaixo:

```
eag.entry.Point (0., 0.)

eag.entry.Point (200., 0.)

eag.entry.KeyEnter ()

eag.exec.Command ("ID_INSERIR_LINHA")
```

As funções que simulam entrada do usuário são:

```
entry .KeyEscape ()
```

Entra tecla de <Escape>

```
entry .KeyEnter ()
```

Entra tecla de <Enter>

```
entry .KeyFunction( ishift,icontrol,ialt,ifn)
```

Entra tecla de função Fn, opcionalmente com <Shift>, <Control> e <Alt> acionados

```
entry .String (str)
```

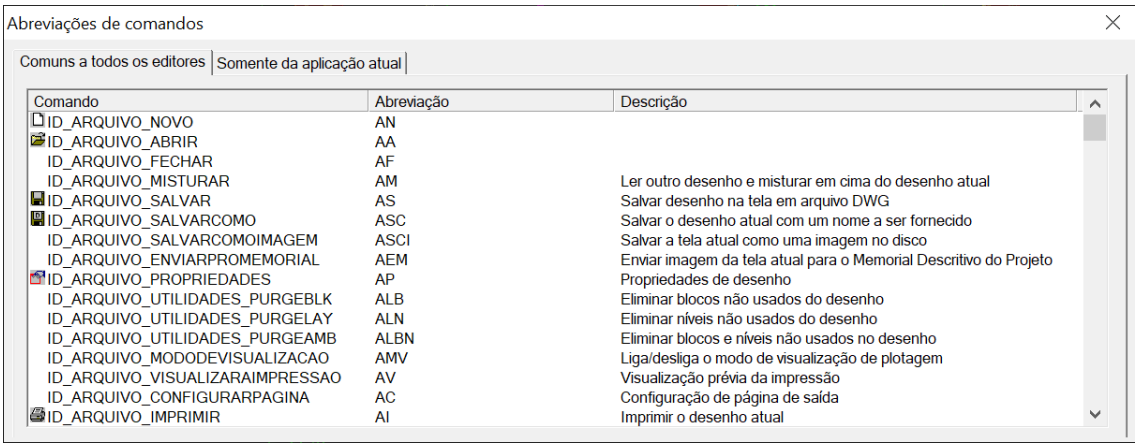
Entra um texto, que será interpretado conforme o comando

```
entry .Point (x, y)
```

Entra um par de coordenadas

Execução de comandos com TQSEag.exec

O subobjeto `exec` do objeto `TQSEag.Eag` permite executar comandos e controlar a interface deles no editor. Os identificadores de comandos disponíveis podem ser determinados examinando-se os arquivos .MEN do TQS, ou simplesmente acionando o comando "Editar, Abreviações".




```
exec .Command (ident)
```

Executa um comando do editor.

`ident` identificador tipo `ID_xxx` declarado no arquivo .MEN

```
exec .EnableCommand (ident, ienable)
```

Habilita ou desabilita um comando do editor.

`Ident`identificador tipo ID_xxx declarado no arquivo .MEN

`Ienable`(0) desabilita (1) habilita comando

```
exec .CheckCommand (ident, icheck)
```

Coloca marca "Check" no comando, quando aplicável

`Ident`identificador tipo ID_xxx declarado no arquivo .MEN

`Icheck`(0) não (1) marca check

```
exec .EditW (name)
```

Chama o editor `EDITW.EXE` para edição de um arquivo Ascii fornecido

`Name`Nome do arquivo, com ou sem path

Retorna (0) se editou (1) não

Mensagens na janela do editor com TQSEag.msg

O subobjeto `msg` da classe `TQSEag.Msg` permite a emissão de mensagens na janela de mensagens do editor e na área de status.

```
msg .Print (*args)
```

Emite mensagens no estilo do "print" do Python, na janela do editor

```
msg .ClearMessageWindow ()
```

Limpa a janela de mensagens do editor

```
msg .PrintStatus (*args)
```

Emite mensagens no estilo do "print" do Python, na área de status do editor

```
msg .WindowTitle (tqsjan, title)
```

Define o título da janela atual de desenho

`Tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`Title`Novo título da janela

Leitura de pontos e objetos com TQS.locate

O subobjeto `locate` da classe `TQS.Locate` tem funções para interação com o usuário – leitura de coordenadas e seleção de elementos.

```
locate .GetPoint( tqsjan, msg)
```

Lê um ponto do mouse/teclado.

`tqsjan`objeto da classe `TQSJan.Window` passado pelo EAG

`msg`Mensagem para leitura do ponto

Retorna : `icod, x, y`

`icod` (0) `<Enter>` (-1) `<Esc>` (1) `<B1>` (2) `<B2>` (3) `<B3>`

(`>20`) Teclas de função

(`<-1`) Código Ascii negativo ("A"=-65)

```
locate .GetSecondPoint (tqsjan,x1,y1,irubber,itprubret,msg)
```

"""

Lê um segundo ponto com mouse/teclado, fazendo rubberband com x1,y1

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`x1, y1`Ponto de referência para rubberband

`irubber`Tipo de linha elástica – uma das constantes abaixo.

`Itprubret`Preenchimento de linha elástica retangular –constantes abaixo

`msg`Mensagem para leitura do ponto

Retorna `icod, x, y`

`icod` (0)`<Enter>`(-1)`<Esc>`(1) `<B1>` (2) `<B2>`(3) `<B3>`

(`>20`) Teclas de função

(`<-1`) Código Ascii negativo ("A"=-65)

Constantes de linha elástica	Finalidade
`TQSEag.EAG_RUBNAO`	Sem linha elástica
`TQSEag.EAG_RUBLINEAR`	Linha elástica linear
`TQSEag.EAG_RUBRETANG`	Linha elástica retangular
`TQSEag.EAG_RUBPANDIN`	Pan dinâmico

Preenchimento de linha elástica	Finalidade
`TQSEag.EAG_RUBRET_NAOPREEN`	Não preencher

TQSEag.EAG_RUBRET_JANW	Preenche como uma janela tipo W
TQSEag.EAG_RUBRET_JANC	Preenche como janela tipo C
TQSEag.EAG_RUBRET_JANUSR	Preenche como W (à direita) ou C (à esquerda)

locate **.SetNextZ** (z)

Define coordenadas Z que podem ser usadas pelo próximo comando.

A coordenada é zerada após o próximo locate.GetPoint.

locate.GetLastZ()

Retorna (idisp, z) o valor da cota Z da última entrada, se disponível

Idisp(1) Se cota Z dizponível

ZValor, se disponível

locate **.GetLabel** (tqsjan, msg)

Lê texto do usuário, com mensagem.

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

msgMensagem

Retorna str, istat

StrTexto lido

Istat (0) Ok (1) <Enter> (2) <Esc>

"""

locate **.GetValue** (tqsjan,msg,valmin,valmax,valdef)

Leitura de número em ponto flutuante.

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

msgMensagem

valminValor mínimo

valmaxValor máximo

valdefValor padrão

Retorna val, istat

ValValor lido

Istat (0) Ok (1) Não leu

locate **.GetAngle** (tqsjan,msg,angdef)

Leitura de um ângulo em graus. Aceita entrada de <R> 3 pontos e <L> linha.

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

msgMensagem

angdefValor padrão

Retorna `val, istat`

`Ang`Ângulo lido

`Istat` (0) Ok (1) Não leu

`locate` **`.GetEntryType`** `()`

Retorna (0) para última entrada alfanumérica ou (1) gráfica

`locate` **`.GetYesNo`** `(tqsjan, msg, r1, r2)`

Mostrar mensagem `msg` e retorna se r1 ou r2 foram escolhidos

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`msg`Mensagem

`r1,r2`String de resposta, como "S" e "N"

Retorna `istat`

`Istat` (0) `r1` (1) `r2` (2) `<Escape>`

`locate` **`.GetCursor`** `(tqsjan)`

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna coordenadas `x,y` na posição do cursor no mundo real

`locate` **`.SetRubberIdle`** `(tqsjan,irubberidle,xrubbidle,yrubbidle)`

Liga ou desliga linha elástica com ponto de referência fornecido.

A linha fica acesa até o próximo comando ou `<Esc>`

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`irubberidle(0)` desligada `(1)` ligada

`xrubbidle`Coordenadas para ativar a linha elástica

`yrubbidle` Coordenadas para ativar a linha elástica

`locate` **`.GetPolyline(`** `tqsjan)`

Lê uma poligonal fechada ou aberta por pontos.

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna

`xy[][2]` Matriz de pontos lidos. Não leu se size== 0

`locate.` **`Select`** `(tqsjan,msg,itploc)`

Faz seleção de elementos conforme `itploc`. Emite mensagem do usuário. Retorna endereço do 1o elemento e outros dados. Para mais elementos, use `BeginSelect/GetSelect`.

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`msg`Mensagem ao usuário

`itploc`Tipo de seleção – uma das constantes abaixo

Retorna:

`addr`Handle do 1o elemento gráfico selecionado (`Dwg`)

`x`Coordenadas de seleção

`y`Coordenadas de seleção

`np`Índice do ponto se poligonal

`istat` (!=0) Se não selecionou

Tipo de seleção	Uso
`TQSEag.EAG_INORM`	Padrão (por ponto, modificável)
`TQSEag.EAG_IJANEL`	Seleção por janela
`TQSEag.EAG_ICURS`	Coordenada do cursor
`TQSEag.EAG_IMULTP`	Seleção múltipla
`TQSEag.EAG_IALTJAN`	Tenta seleção por ponto, depois janela

`locate .BeginSelection (tqsjan)`

Prepara para a leitura de elementos selecionados em l `ocate.Select`.

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna(1) se não há elementos

`locate .NextSelection( tqsjan)`

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna `handle` para o próximo elemento gráfico selecionado ou `None`

`locate .DragSelection( tqsjan,xref,yref)`

Acende elementos os selecionados e os arrasta junto com o cursor

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`xref,yref` é o ponto de inserção inicial da lista

`locate .DragOff (tqsjan)`

Desliga elementos ligados em `DragSelection`.

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Estados do editor com TQSEag.state

O subobjeto `state` da classe `TQSEag.State` permite controlar diversos estados globais do editor que afetam a entrada de dados gráficos.

`state .GetOrtho (tqsjan)`

Retorna o modo ortogonal

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna:

`Iortho` (0) desligado (1) modo ortogonal

`Angle`Ângulo em graus do modo ortogonal

`state` **`.SetOrtho`** `(tqsjan,iortho,angle)`

Define o modo ortogonal

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`iortho` (0) Desligado (1) Modo ortogonal

`angle`Ângulo em graus do modo ortogonal

`state` **`.GetLevelLock`** `(tqsjan)`

Retorna o nível travado

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna

`Ilevellock`(0) Desligado (1) Nível travado

`state` **`.SetLevelLock`** `(tqsjan,ilevellock)`

Define o nível travado

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`Ilevellock`(0) Desligado (1) Nível travado

`state` **`.ClearAllLevelsLock`** `(tqsjan)`

Limpa travas individuais de níveis

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`state` **`.GetOneLevelLock`** `(tqsjan, ilevel)`

Lê a trava de um nível

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`ilevel`Nível considerado

Retorna:

`Ilock` (1) Se este nível está travado

`state` **`.SetOneLevelLock`** `(tqsjan, ilevel, ilock)`

Define a trava de um nível

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`ilevel`Nível considerado

`Ilock` (1) Se este nível está travado

`state` **`.GetActiveLevel`** `(tqsjan)`

Retorna o nível atual

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna:

`iactivelevel`(0) Desligado (1) Nível travado

state.SetActiveLevel(tqsjan,iactivelevel)

Define o nível atual

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

iactivelevel(0) Desligado (1) Nível travado

state .**GetFastCurve(** tqsjan)

Retorna (1) se curva rápida ligada

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

Retorna:

ifastcurve(0) modo normal (1) curva rápida (curvas mostradas como linhas)

state .**SetFastCurve** (tqsjan,ifastcurve)

Define (1) se curva rápida ligada

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

ifastcurve(0) modo normal (1) curva rápida (curvas mostradas como linhas)

state .**GetGrid** (tqsjan)

Retorna dados de grade

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

Retorna:

igrade (0) Não (1) Grade ligada

grdorxX origem da grade cm

grdoryY origem da grade cm

grdesxX espaçamento da grade cm

grdesyY espaçamento da grade cm

grdangÂngulo da grade, em graus

iespacpontoNúmero de espaçamentos por ponto da grade

igraderubber(1) Se grade em linha elástica

state .**SetGrid** (tqsjan,igrade,grdorx,grdory,grdesx,grdesy,grdang,

iespacponto, igraderubber)

Define dados da grade

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

igrade (0) Não (1) Grade ligada

grdorxX origem da grade cm

grdoryY origem da grade cm

grdesxX espaçamento da grade cm

grdesyY espaçamento da grade cm

grdangÂngulo da grade, em graus

iespacpontoNúmero de espaçamentos por ponto da grade

igraderubber(1) Se grade em linha elástica

```
state .InvertCapture( tqsjan)
```

Inverte a captura de coordenadas (ligada ou desligada).

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

```
state .GetFatText (tqsjan)
```

Retorna o modo de texto rápido (1) ligado

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

```
state .SetFatText (tqsjan,ifast)
```

Define o modo de texto rápido

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

ifast (1) se modo de texto rápido

```
state .GetBlockInsertion (tqsjan)
```

Retorna dados para inserção de um bloco novo

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

Retorna:

xscaleEscala X

yscaleEscala Y

angleÂngulo de inserção em graus

```
state .SetBlockInsertion (tqsjan,xscale,yscale,angle)
```

Define dados para inserção de um bloco novo

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

xscaleEscala X

yscaleEscala Y

angleÂngulo de inserção em graus

```
state .GetParallelDist (tqsjan)
```

Retorna distância padrão no comando "Paralela a elemento".

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

```
state .SetParallelDist (tqsjan,distance)
```

Define distância padrão no comando "Paralela a elemento".

tqsjanObjeto da classe TQSJan.Window passado pelo EAG

```
state .GetFilletRadius (tqsjan)
```

Retorna a distância de arredondamento padrão de arcos.

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`state` **`.SetFilletRadius`** `(tqsjan,radius)`

Define a distância de arredondamento padrão de arcos

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

`state` **`.GetFilletlDist`** `(tqsjan)`

Retorna as distâncias padrão de chanfragem

`tqsjan`Objeto da classe `TQSJan.Window` passado pelo EAG

Retorna:

`dist1, dist2`Distâncias de chanfragem

`state` **`.SetFilletlDist`** `(tqsjan,dist1,dist2)`

Define as distâncias padrão de chanfragem

tqsjanObjeto da classe `TQSJan.Window` passado pelo EAG

`dist1, dist2`Distâncias de chanfragem

Menus de demonstração EAGFOR.PYMEN e programa EAGFor.py.

Como já mostramos, o menu `EAGFor.PYMEN` e o programa `EAGFor.py` ligado a ele devem ficar na pasta `TQSW\EXEC\Python`, ou na pasta correspondente conforme a instalação TQS. Já falamos sobre o princípio de funcionamento de um menu Python para os editores EAG. Vamos descrever apenas os cinco comandos neste menu de exemplo.

Comando "Geração de desenho"

Este comando mostra a facilidade de gerar desenhos usando o objeto de desenho `TQSDwg.Dwg`, que pode ser extraído do `TQSJan.Window`. Neste exemplo, o exemplo inteiro definido no módulo `TQSDwg` é rodado, passando o objeto de desenho aberto. Uma longa listagem de extração de elementos gráficos é emitida por este comando, que também acrescenta alguns elementos gráficos no desenho atual, e mistura e insere um desenho como referência externa.

Comando "Acionamento de comandos do editor"

Aqui mostramos como acionar comandos padrão do editor, fornecendo antes os dados que o usuário entraria interativamente. O programa simula a entrada interativa, usando funções da classe `TQSEag.Entry`. Um comando do editor é marcado como "Checado" e a janela de mensagens é limpa.

Comando "Entrada de pontos"

Aqui são testados os comandos de entrada de pontos, ângulos, valores e texto. Um texto, uma linha e uma linha poligonal fechada e preenchida são construídos. No final, uma linha elástica é deixada ligada nas coordenadas do texto.

Comando "Seleção de elementos e movimentação"

Testamos o comando para locação de elementos, e mostramos como iterar pelos elementos selecionados. No final, a lista selecionada se acende, e movimentamos os elementos com o fornecimento de novo ponto.

Comando "Estados do editor"

Diversos estados do editor são alterados, e o resultado é mostrado antes e depois da alteração. A grade de coordenadas é deixada ligada no final do comando.