

Programação em Python

Os sistemas TQS tiveram origem na década de 1970 e se tornaram ao longo dos anos um complexo sistema de modelagem, análise, dimensionamento, detalhamento e desenho de estruturas de concreto armado, protendido, pré-moldado, alvenarias e paredes estruturais. Em geral, a introdução de novos elementos, sistemas estruturais e critérios de cálculo estiveram a cargo da TQS. A possibilidade de desenvolver ferramentas por parte do engenheiro usuário do sistema permite estendê-lo para atender novas necessidades, calcular novos elementos, e facilitar o detalhamento.

A TQS já possui hoje dois sistemas de programação de desenhos que podem ser usados pelo usuário. Este manual detalha um terceiro, criado a partir da versão 23, que usa a linguagem de programação Python® como ferramenta.

Nos anos 1990 a TQS introduziu o sistema DP de desenho paramétrico. O DP é uma linguagem interpretada orientada para construções geométricas, que permite gravar desenhos em geral, e gerar objetos de ferros inteligentes (<http://docs.tqs.com.br/Docs/Details?id=3854&language=pt-BR>)

Nos anos 2000 a TQS criou os componentes tipo OCX, TQSDWG, TQSJAN e TQSGEO, que permitem programar em qualquer linguagem que aceite componentes OCX. O componente TQSDWG permite a geração de desenhos, o TQSJAN mostra estes desenhos em uma janela Windows, e o TQSGEO complementa com uma biblioteca de geometria analítica 2D. A linguagem mais usada na época com estes componentes era o VB, mas também podiam ser usadas linguagens como C# e Delphi (<http://docs.tqs.com.br/Docs/Details?id=3536&language=pt-BR>)

A mais nova interface com TQS foi escrita na linguagem Python, que além das funções criadas nos componentes anteriores estende a programação para outros sistemas como Edição Gráfica e modelagem. Novos componentes estão sendo desenvolvidos para acessar mais funcionalidades do TQS.

Python é uma linguagem de programação de propósito geral de alto nível, interpretada de script, multiparadigma, imperativa, orientada a objetos, funcional, de tipagem dinâmica forte, lançada por Guido van Rossum em 1991 (Wikipedia <https://pt.wikipedia.org/wiki/Python>). Uma de suas principais características é permitir a fácil leitura do código e exigir poucas linhas se comparada ao mesmo programa em outras linguagens – é um dos motivos de ser tão difundida atualmente.

Pacote e Módulos

A distribuição é feita através de um pacote instalável `TQS`. Dentro deste pacote são definidos os seguintes módulos documentados neste manual:

`TQSDwg`: para a criação e leitura de desenhos DWG TQS. Permite criar desenhos como qualquer programa TQS, e também ler e extrair informações de desenhos. Na programação foram incluídos os objetos de cotagem associativa e ferros inteligentes.

`TQSJan`: para mostrar um desenho TQS em uma janela Windows. Um programa qualquer com uma janela Windows pode mostrar um desenho TQS controlado por `TQSDwg` através de uma janela ligada ao módulo `TQSJan`.

`TQSEag`: interface para programação dentro dos editores gráficos. Um ou mais comandos em menus podem ser introduzidos em um dos editores gráficos TQS para tratar um desenho. A programação dentro dos editores gráficos se faz com uma combinação do módulo `TQSDwg` (para tratar o desenho), `TQSJan` (para tratar a janela) e `TQSEag` (iteração com o usuário e estados do editor).

`TQSGeo`: biblioteca auxiliar de geometria analítica 2D para usar na programação de desenhos.

`TQSUtil`: outros utilitários comuns aos componentes

`TQSBuild`: Leitura e gravação de dados do edifício. São informações de pisos, cotas, materiais, cobrimentos,

cargas e critérios. Permite fazer tudo o que o Editor de Dados do Edifício faz.

TQSGrid: Montagem de tabelas em desenho DWG, HTML e LST.

TQSExec: Execução automática de tarefas com o gerenciador TQS.

A interface dentro dos módulos é feita através de funções, objetos e propriedades. Devido ao TQS ser um sistema traduzido para mais de uma língua, foi escolhida a língua inglesa para dar nome a todos os componentes da interface.

Instalação e uso

A programação em Python exige a instalação de um interpretador Python compatível. Você precisa instalar uma versão do Python para Windows, 32 bits, versão 3.7 ou superior. Um site sugerido para obter o interpretador é o python.org.

O pacote **TQS** é distribuído através de um arquivo tipo *Wheel* e instalado pelo utilitário **PIP**, que vem junto com a instalação do Python. Todos os arquivos relacionados à interface Python são distribuídos na pasta

`TQSW\EXEC\Python` e copiados para esta pasta durante a instalação do TQS.

Atualmente existe a dependência do Python 3.8 para o teste feito com o módulo `TSTJAN.py`. Todos os programas que rodam dentro dos editores gráficos precisam estar na pasta `TQSW\EXEC\Python`. Outros programas podem rodar em qualquer pasta, usando o módulo TQS.

Este manual trata da versão 1.2.1 da interface TQS-Python. A instalação do TQS se possível instala automaticamente o pacote. Se a instalação não for possível, você deve fazer isto, entrando na pasta `TQSW\EXEC\Python` e rodando o utilitário PIP, como abaixo:

```
pip install TQSPythonInterface-1.2.1-py310-none-any.whl
```

Se o PIP não estiver no PATH da instalação, é possível também aciona-lo através do Python:

```
py -m pip install ....
```

O nome do arquivo *Wheel* pode variar de versão para versão. Após a instalação, os módulos `TQSDwg`, `TQSJan`, `TQSEag`, `TQSGeo`, `TQSUtil`, `TQSBuild`, `TQSGrid` e `TQSExec` ficarão disponíveis para importação dentro do pacote TQS. Os arquivos distribuídos junto com o TQS na pasta `TQSW\EXEC\Python` tem a seguintes finalidades:

Arquivo	Finalidade
<code>Programacao_TQS_Python.pdf</code>	Este manual
Aquivo <code>.whl</code>	Instalação do pacote TQS para Python
<code>EAGPpy.py</code>	Ponte entre os editores gráficos e os comandos nos menus Python tipo <code>.PYMEN</code>
<code>EAGFOR.py</code>	Teste de programação do Editor Gráfico de Formas EAGFOR
<code>EAGFOR.PYMEN</code>	Menu de edição gráfica que usa o <code>EAGFOR.py</code>
<code>TSTDWG.py</code>	Programa de teste do módulo <code>TQSDwg</code>
<code>TSTFER.py</code>	Programa de teste do módulo <code>TQSDwg</code> com Ferros Inteligentes
<code>TSTJAN.py</code>	Programa de teste do módulo <code>TQSJan</code>

<code>TSTGEO.py</code>	Programa de teste do módulo <code>TQSGeo</code>
<code>TSTUTIL.py</code>	Programa de teste do módulo <code>TQSUtil</code>
<code>TSTBUILD.py</code>	Teste do módulo <code>TQSBuild</code>
<code>TSTGrid.py</code>	Teste do módulo <code>TQSGrid</code>
<code>TSTExec.py</code>	Teste do módulo <code>TQSExec</code>
<code>TSTLISFER.PY</code>	Funções de listagem de linhas de ferro
<code>TSTITERATOR.PY</code>	Teste de múltiplos iteradores de DWG

O programa `TSTJAN.PY` que testa o módulo `TQSJan` faz a montagem de uma janela Windows para teste. Para este teste foi usado o Módulo `wxPython`, que deve ser baixado compatível com a versão Python atual pelo comando:

```
pip install -U wxPython.
```

Organização dos módulos

Os módulos `TQSGeo` e `TQSUtil` definem apenas funções. Os demais definem objetos e funções e/ou propriedades com o propósito do módulo. Para evitar a poluição de nomes de rotinas e propriedades, as funcionalidades foram divididas em objetos menores (subobjetos). Por exemplo:

Um objeto `dwg` representa um desenho e é criado com o módulo `TQSDwg`:

```
from TQS import TQSDwg
dwg = TQSDwg.Dwg ()
```

Para desenhar uma linha, usamos uma função dentro do subobjeto `draw`:

```
dwg.draw.Line (500, 300, 0, 300)
```

Mas para salvar um desenho, temos o subobjeto `file`:

```
dwg.file.SaveAs ("TESTE")
```

Esta organização é mostrada dentro de cada capítulo.

Algumas convenções usadas

Trechos e sintaxe de programas, comandos acionados em linha de comando, nomes de pastas no computador – são mostrados com fonte `Courier New`.

Nomes de constantes, propriedades e funções são mostrados em **negrito**.

Constantes são adotadas em letra maiúscula, como em `TQSUtil` **.PI**.

Propriedades começam em letra minúscula, como em `dim` **.idmnil**. A maior parte das propriedades permite leitura e gravação, a menos quando indicado diferente.

Funções começam sempre em letra maiúscula, como em `TQSUtil` **.SupportFolder** ().