NAME test-tc.py

Test driver to exercise the `tconfpy` Python module.

SYNOPSIS

test-tc.py [symtbl] [inmem] [nonewvars] [templates] [temponly] [limitns] [litvars] [nopredefs] [debug] cfg-file, ...

OPTIONS

symtbl This option causes the test driver to include a symbol table with some predefined test variables in it. These are useful for experimenting with variable dereferencing, substitution, and type/value enforcement. The variables created are:

Name	Val	RO	Type	Def	Legal Values	Min/Max	
int1	1	True	int	0	[1, 2, 23]	None	None
int2	1	True	int	0	[]	1	30
float1	1.0	None	float	0.5	[3.14, 2.73]	None	None
float2	1.0	None	float	0.5	[]	-1.2	0.5
str1	"stringy"	None	None	None	[r'^box$', r'^Bax', r'a+bc']	3	8
cmplx1	4+5j	None	complex	0-0j	[]	None	None
cmplx2	4+5j	None	complex	0-0j	[1-1j 1+1j]	None	None
bool1	True	None	boolean	None	None	None	None
ro1	"ReadVar"	False	None		None None	None	None

Notice that some of the fields are defined as `None`. In this case, that parameter is set to its default value when the variable is created. (See the `tconfpy` documentation regarding the `VarDescriptor` object for the details.)

The default is to not predefine any such variables. (Note that whether this option is present or not, `tconfpy` always creates a number of predefined variables of its own internally. See the `tconfpy` documentation for the details.)

You can see all the predefined variables (and their attributes) by running `test-tc.py` on an empty configuration file. This will show you both the variables automatically defined by `tconfpy` as well as any variables created with this option, if present.

inmem Read each cfgile specified on the command line into an in-memory list, and pass this list to the parser. This exercises the in-memory parsing capabilities of `tconfpy`. It is primarily provided as a diagnostic tool for anyone modifying the parser code.

nonewvars

This option disables the creation of new variables in the configuration file (via the `AllowNewVars` API option). The user is limited to referencing and modifying only those variables already present in the symbol table. Typically used when passing an initial symbol table to the parser to limit the user to only those variables. The default is to permit new variable creation.

templates

This option creates a default set of variable templates for use in the configuration file. These are passed to the `tconfpy` parser via the `Templates={}` API option.

The following variable templates are predefined when this option is used:

```
Name     Val       RO Type    Def   Legal Values                 Min/Max
--------------------------------------------------------------------------


templb   1         RW boolean False []                           None None
templc   4+5j      RW complex 0-0j  [1-1j, 1+1j]                 None None
templf   1.0       RW float   0.5   [3.14, 2.73]                 None None
templi   1         RW int     0     [1, 2, 23]                   None None
templs   "stringy" RW string  ""    [r'^box$', r'^Bax', r'a+bc'] 3    8
```

temponly

This option will only permit new variable creation if a template for that variable exists (via the `TemplatesOnly=True` parser API option). This is used in conjunction with the `templates` option above.

limitns This option will populate the initial symbol table so that only a limited number of namespaces can be used. If this option is present, only the root namespace, and namespaces beginning with the string "NS" will be allowed. In this case, namespaces will be required to be from 3 to 8 characters long. The default without this option is to allow new namespaces to be used without restrictions on name or length. The initial namespace is set to root ("") whether or not this option is selected.

litvars By default, `tconfpy` does nothing to text encountered inside of `.literal` blocks in a configuration file. If this option is present, the test driver tells `tconfpy` to replace any variable references present in a `.literal` block. See the `tconfpy` documentation for a more complete description.

nopredefs

This option tells the parser (via the `ReturnPredefs=False` API option) to **not** include any of its own predefined variables in the symbol table it returns.

debug This option causes the test driver to invoke `tconfpy` with debugging enabled and then display debug output when parsing is complete. The default is for debug output to be disabled.

ADDING OR CHANGING PREDEFINED VARIABLES IN THE TEST DRIVER

This test driver uses a simple table-driven scheme for predefining variables should you desire this feature. This can be found near the beginning of the `test-tc.py` file. All this should be fairly self-explanatory once you understand the various variable attributes recognized by `tconfpy`. Edit this table as desired to create more variables or different attribute values for your testing purposes.

The exact same scheme is used for predefining variable templates and you can add/delete/change these to suit your own tastes in the same way.

OTHER

Requires Python 2.3 or later.

BUGS AND MISFEATURES

The symbol table dump at the end of a program run could be formatted better.

COPYRIGHT AND LICENSING

AUTHOR

```
Tim Daneliuk
tconfpy@tundraware.com
```

DOCUMENT REVISION INFORMATION

$Id: test-tc.1,v 1.113 2005/01/20 10:35:04 tundra Exp $