

Variables (access)

```
Set VsUtil s.v ari abl es[ '<n ame
>'] = v

Get v = VsUtil s.v ari abl es[ '<n
ame >']
```

Tip: use `<from VsUtils import variables as vs>` to access variables easily with `<vs['...']>`

Assertion

```
assert True(expr)

assert False(expr)

assert Equal(fisrt,second)

assert Not Equal(fisrt,second)

assert Alm ost Equal(fisrt,second,places=7)

assert Not Alm ost Equal(fisrt,second,places=7)

assert Raises(excClass,callObj, *args,gs)

fail()
```

all assertions with additional parameters:

```
(..., msg=None, continueTest = True)

msg could use format string: "...% i" % vs[...]
```

Async assertion

```
assert Tru eIn ter val (expr)

assert Fal seI nte rva l(expr)

assert Alm ost Equ alI nte rva l(f is t,s ec, places)

assert Not Alm ost Equ alI nte rva l(f is t,s ec, places)

assert Tru eAt Lea stI nte rva l(expr)

assert AtL eas tOn ceT rue (expr)
```

all assertions with additional parameters:

```
(..., msg=None)
```

Async usage

```
async = ATLAsyncTest(self)

async.a ss ert...( la mbd -
a:vs[varname]...,msg)

async.a ss ert...
( 'vs[varname]...',msg)

async.a ss ert...f(nction,msg)

async.e xe cut e(t imeout)
```

Interact with Logbook

```
rtc.setLogLevel(level)

rtc.st art Log Ses -
sion('Sessio nName', 'comment')

rtc.lo gMe ssa ge( RTC.Lo -
gger.level, 'UserModule', 'msg')

...

rtc.st opL ogS ess ion()
```

Levels are: *Debug, Info, Warning, Error, Critical*

RTC services

Creation

```
rtc = RTC.RT CMa nager(ip [, port])
```

Connection

```
rtc.co nnect(ip [, port])
```

Configure if not done by mmi

```
rtc.co nfi gure(client name, dbPath,
**k war
```

Usercode control

```
rtc.[s tar t/s top ]Code(codeName)
```

Monitoring control

```
rtc.[s tar t/s top ]Mo nitor(taskName
)
```

Archiver control

```
rtc.[s tar t/s top ]Re cor dset(rsvam
e)
```

Manage errors

```
try:... except: System Error , detail
:
```

VS callback

```
class Notifier(Vs.Callback):

    def variab leD idC han -
ge( *args):

        print 'callback
called'

callback = notifier()

notif = VsUtil s.vc.

    reg ist erC all bac kVa -
lue Cha nge d(n ame ,ca llback)

...

del notif
```

Callback class should be created outside of *register* method.

Other callback are available.

Tips

```
import time

time.s lee p(s econds)
```

Testcase

```
class <TestCase
name>(ATLTestCase):

    def initia liz e(s elf):

        # no exception inside
this method

    def finali ze( self):

        # no exception inside
this method

    def execut e(s elf):

        # test steps here, see
Assertions
```

Test structure (tsdef)

TestSuite (tsdef)

Root TestGroup

TestGroup

TestCase

TestCase

Groups are optional

Running tests

```
$PYTHON- /opt/gc/lib/python
NPATH

/opt/vs/lib

python utest/ATL/ATLTestRunner.py

-c config.xml

-r result.xml

-v verbose in console

[--server] xmlrpc server
```

Test results

Passed tests within limits

Failed test not in limits

Aborted not completed (*exception*)

expr and *msg* could be lamda, string or function
expr should contains only one variable.
expr is evaluated one time to get the variable

Advanced tips

How to disable warning import msg

```
sys.modules['hook s'].current_importer.add_ok_module('xxx')
```

In custom assert, how to get correct file/line

Use *tbskip* param in *self.result.addFailure*

How to interact with operator ?

Use *wxpython* binding

Variables (creation)

```
VsUtil s.v c.c rea teV ari abl e('<name>', Vs.<type>, [capacity=1])
```

```
VsUtil s.v c.d ele teV ari abl e('<name>')
```

Types are: *INTEGER*, *DOUBLE*, *OPAQUE*



By **Vicnet**
cheatography.com/vicnet/

Published 30th November, 2016.
Last updated 30th November, 2016.
Page 1 of 2.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>