def \_init\_(self, namelist=None, pathlist=None):

if namelist is None:

namelist = []

self.namelist = namelist

if pathlist is None:

pathlist = []

self.pathlist = pathlist

try:

# blacklist all files in org.python.pydev/pysrc

import pydev\_pysrc, inspect

self.pathlist.append(os.path.dirname(pydev\_pysrc.\_file\_))

except:

pass

self.previous\_modules = list(sys.modules.keys())

def is\_module\_blacklisted(self, modname, modpath):

for path in [sys.prefix] + self.pathlist:

if modpath.startswith(path):

return True

else:

return set(modname.split('.')) & set(self.namelist)

def run(self, verbose=False):

"""

Del user modules to force Python to deeply reload them

Do not del modules which are considered as system modules, i.e.

modules installed in subdirectories of Python interpreter's binary

Do not del C modules

"""

log = []

modules\_copy = dict(sys.modules)

for modname, module in modules\_copy.items():

if modname == 'aaaaa':

print(modname, module)

print(self.previous\_modules)

if modname not in self.previous\_modules:

modpath = getattr(module, '\_file\_', None)

if modpath is None:

# module is a C module that is statically linked into the

# interpreter. There is no way to know its path, so we

# choose to ignore it.

continue

if not self.is\_module\_blacklisted(modname, modpath):

log.append(modname)

del sys.modules[modname]

if verbose and log:

print("\x1b[4;33m%s\x1b[24m%s\x1b[0m" % ("UMD has deleted",

": " + ", ".join(log)))

\_umd\_ = None

\_get\_globals\_callback = None