GitHub This repository Search Explore Features Enterprise Pricing Sign up Sign in

dshikashio / **Pybag**

Watch 4 Star 9 Fork 1

Branch: **master** **Pybag** / examples / **pytest.py**

dshikashio on Jul 17, 2011 Importing code

1 contributor

217 lines (186 sloc) 8.59 KB

Raw Blame History

```
import sys
import pydbgeng


def ev_beakpoint(*args):
    print "DEBUG_EVENT_BREAKPOINT"
    print args

def ev_exception(*args):
    print "DEBUG_EVENT_EXCEPTION"
    print "ExceptionRecord: ", args[0]
    print "FirstChance: ", args[1]

def ev_loadmodule(*args):
    print "DEBUG_EVENT_LOAD_MODULE"
    print "ImageFileHandle: ", args[0]
    print "BaseOffset: %x" % args[1]
    print "ModuleSize: ", args[2]
    print "ModuleName: ", args[3]
    print "ImageName:  ", args[4]
    print "CheckSum:   ", args[5]
    print "TimeDate:   ", args[6]
    #return pydbgeng.DEBUG_STATUS_BREAK

def ev_unloadmodule(*args):
    print "DEBUG_EVENT_UNLOAD_MODULE"
    print "ImageBaseName: ", args[0]
    print "BaseOffset: %x" % args[1]

def ev_createprocess(*args):
    print "DEBUG_EVENT_CREATE_PROCESS"
    print "ImageFileHandle: ", args[0]
    print "Handle: ", args[1]
    print "BaseOffset: %x" % args[2]
    print "ModuleSize: ", args[3]
    print "ModuleName: ", args[4]
    print "ImageName: ", args[5]
    print "CheckSum: ", args[6]
    print "TimeDate: ", args[7]
    print "InitialThreadHandle: ", args[8]
    print "ThreadDataOffset: %x" % args[9]
    print "StartOffset: %x" % args[10]
    #return pydbgeng.DEBUG_STATUS_BREAK

def ev_exitprocess(*args):
    print "DEBUG_EVENT_EXIT_PROCESS"
    print "ExitCode: ", args[0]

def ev_createthread(*args):
    print "DEBUG_EVENT_CREATE_THREAD"
    print "Handle: ", args[0]
    print "DataOffset: %x" % args[1]
    print "StartOffset: %x" % args[2]
    #return pydbgeng.DEBUG_STATUS_BREAK

def ev_exitthread(*args):
    print "DEBUG_EVENT_EXIT_THREAD"
    print "ExitCode: ", args[0]
    return pydbgeng.DEBUG_STATUS_BREAK

def ev_systemerror(*args):
    print "DEBUG_EVENT_SYSTEM_ERROR"
    print "Error: ", args[0]
    print "Level: ", args[1]

def ev_sessionstatus(*args):
    print "DEBUG_EVENT_SESSION_STATUS"
    sdict = {pydbgeng.DEBUG_SESSION_ACTIVE: "DEBUG_SESSION_ACTIVE",
             pydbgeng.DEBUG_SESSION_END_SESSION_ACTIVE_TERMINATE:
                "DEBUG_SESSION_END_SESSION_ACTIVE_TERMINATE",
             pydbgeng.DEBUG_SESSION_END_SESSION_ACTIVE_DETACH:
                "DEBUG_SESSION_END_SESSION_ACTIVE_DETACH",
             pydbgeng.DEBUG_SESSION_END_SESSION_PASSIVE:
                "DEBUG_SESSION_END_SESSION_PASSIVE",
             pydbgeng.DEBUG_SESSION_END: "DEBUG_SESSION_END",
             pydbgeng.DEBUG_SESSION_REBOOT: "DEBUG_SESSION_REBOOT",
             pydbgeng.DEBUG_SESSION_HIBERNATE: "DEBUG_SESSION_HIBERNATE",
             pydbgeng.DEBUG_SESSION_FAILURE: "DEBUG_SESSION_FAILURE"}
    print "Status: ", sdict[args[0]]

def ev_debuggeestate(*args):
    print "DEBUG_EVENT_CHANGE_DEBUGGEE_STATE"
    fdict = {pydbgeng.DEBUG_CDS_ALL: 'DEBUG_CDS_ALL',
            pydbgeng.DEBUG_CDS_REGISTERS: 'DEBUG_CDS_REGISTERS',
            pydbgeng.DEBUG_CDS_DATA: 'DEBUG_CDS_DATA'}
    print "Flags: ", fdict[args[0]]
    if args[0] == pydbgeng.DEBUG_CDS_DATA:
        adict = {pydbgeng.DEBUG_DATA_SPACE_VIRTUAL: 'DEBUG_DATA_SPACE_VIRTUAL',
               pydbgeng.DEBUG_DATA_SPACE_PHYSICAL: 'DEBUG_DATA_SPACE_PHYSICAL',
               pydbgeng.DEBUG_DATA_SPACE_CONTROL: 'DEBUG_DATA_SPACE_CONTROL',
               pydbgeng.DEBUG_DATA_SPACE_IO: 'DEBUG_DATA_SPACE_IO',
               pydbgeng.DEBUG_DATA_SPACE_MSR: 'DEBUG_DATA_SPACE_MSR',
               pydbgeng.DEBUG_DATA_SPACE_BUS_DATA: 'DEBUG_DATA_SPACE_BUS_DATA'}
        print "Argument: ", adict[args[1]]
    else:
        print "Argument: ", args[1]

def ev_enginestate(*args):
    print "DEBUG_EVENT_CHANGE_ENGINE_STATE"
    fdict = {pydbgeng.DEBUG_CES_CURRENT_THREAD: "DEBUG_CES_CURRENT_THREAD",
            pydbgeng.DEBUG_CES_EFFECTIVE_PROCESSOR: "DEBUG_CES_EFFECTIVE_PROCESSOR",
            pydbgeng.DEBUG_CES_BREAKPOINTS: "DEBUG_CES_BREAKPOINTS",
            pydbgeng.DEBUG_CES_CODE_LEVEL: "DEBUG_CES_CODE_LEVEL",
            pydbgeng.DEBUG_CES_EXECUTION_STATUS: "DEBUG_CES_EXECUTION_STATUS",
            pydbgeng.DEBUG_CES_ENGINE_OPTIONS: "DEBUG_CES_ENGINE_OPTIONS",
            pydbgeng.DEBUG_CES_LOG_FILE: "DEBUG_CES_LOG_FILE",
            pydbgeng.DEBUG_CES_RADIX: "DEBUG_CES_RADIX",
            pydbgeng.DEBUG_CES_EVENT_FILTERS: "DEBUG_CES_EVENT_FILTERS",
            pydbgeng.DEBUG_CES_PROCESS_OPTIONS: "DEBUG_CES_PROCESS_OPTIONS",
            pydbgeng.DEBUG_CES_EXTENSIONS: "DEBUG_CES_EXTENSIONS",
            pydbgeng.DEBUG_CES_SYSTEMS: "DEBUG_CES_SYSTEMS",
            pydbgeng.DEBUG_CES_ASSEMBLY_OPTIONS: "DEBUG_CES_ASSEMBLY_OPTIONS",
            pydbgeng.DEBUG_CES_EXPRESSION_SYNTAX: "DEBUG_CES_EXPRESSION_SYNTAX",
            pydbgeng.DEBUG_CES_TEXT_REPLACEMENTS: "DEBUG_CES_TEXT_REPLACEMENTS"}
    try:
        print "Flags: ", fdict[args[0]]
    except:
        pass
    if args[0] == pydbgeng.DEBUG_CES_EXECUTION_STATUS:
        adict = {pydbgeng.DEBUG_STATUS_NO_DEBUGGEE: "DEBUG_STATUS_NO_DEBUGGEE",
                 pydbgeng.DEBUG_STATUS_BREAK: "DEBUG_STATUS_BREAK",
                 pydbgeng.DEBUG_STATUS_STEP_INTO: "DEBUG_STATUS_STEP_INTO",
                 pydbgeng.DEBUG_STATUS_STEP_BRANCH: "DEBUG_STATUS_STEP_BRANCH",
                 pydbgeng.DEBUG_STATUS_STEP_OVER: "DEBUG_STATUS_STEP_OVER",
                 pydbgeng.DEBUG_STATUS_GO_NOT_HANDLED:
                    "DEBUG_STATUS_GO_NOT_HANDLED",
                 pydbgeng.DEBUG_STATUS_GO_HANDLED: "DEBUG_STATUS_GO_HANDLED",
                 pydbgeng.DEBUG_STATUS_GO: "DEBUG_STATUS_GO",
                 pydbgeng.DEBUG_STATUS_IGNORE_EVENT:
                    "DEBUG_STATUS_IGNORE_EVENT",
                 pydbgeng.DEBUG_STATUS_RESTART_REQUESTED:
                    "DEBUG_STATUS_RESTART_REQUESTED",
                pydbgeng.DEBUG_STATUS_NO_CHANGE:
                    "DEBUG_STATUS_NO_CHANGE"}
        a = args[1] & 0xffffffff
        b = args[1] & pydbgeng.DEBUG_STATUS_INSIDE_WAIT
        if b:
            b = "DEBUG_STATUS_INSIDE_WAIT"
        c = args[1] & pydbgeng.DEBUG_STATUS_WAIT_TIMEOUT
        if c:
            c = "DEBUG_STATUS_WAIT_TIMEOUT"
        print "Argument: ", adict[a], b, c
    else:
        print "Argument: ", args[1]

def ev_symbolstate(*args, **kw):
    print "DEBUG_EVENT_CHANGE_SYMBOL_STATE"
    fdict = {pydbgeng.DEBUG_CSS_LOADS: "DEBUG_CSS_LOADS",
             pydbgeng.DEBUG_CSS_UNLOADS: "DEBUG_CSS_UNLOADS",
             pydbgeng.DEBUG_CSS_SCOPE: "DEBUG_CSS_SCOPE",
             pydbgeng.DEBUG_CSS_PATHS: "DEBUG_CSS_PATHS",
             pydbgeng.DEBUG_CSS_SYMBOL_OPTIONS: "DEBUG_CSS_SYMBOL_OPTIONS",
             pydbgeng.DEBUG_CSS_TYPE_OPTIONS: "DEBUG_CSS_TYPE_OPTIONS"}
    print "Flags: ", fdict[args[0]]
    print "Argument: ", args[1]


#c = pydbgeng.DebugClient(pydbgeng.DEBUG_CREATE)
#c = pydbgeng.DebugCreate()

#c.SetEventCallback(pydbgeng.DEBUG_EVENT_EXCEPTION, ev_exception)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_LOAD_MODULE, ev_loadmodule)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_UNLOAD_MODULE, ev_unloadmodule)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_CREATE_PROCESS, ev_createprocess)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_EXIT_PROCESS, ev_exitprocess)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_CREATE_THREAD, ev_createthread)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_EXIT_THREAD, ev_exitthread)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_SYSTEM_ERROR, ev_systemerror)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_SESSION_STATUS, ev_sessionstatus)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_CHANGE_DEBUGGEE_STATE, ev_debuggeestate)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_CHANGE_ENGINE_STATE, ev_enginestate)
#c.SetEventCallback(pydbgeng.DEBUG_EVENT_CHANGE_SYMBOL_STATE, ev_symbolstate)

#print "Creating Process"
#c.CreateProcess(r"c:\windows\system32\calc.exe", pydbgeng.DEBUG_PROCESS)
#c.SetOutputCallbacks(sys.stdout.write)

#syms = pydbgeng.DebugSymbols(c)
#print "%x" % syms.GetSymbolOptions()
#syms.RemoveSymbolOptions(pydbgeng.SYMOPT_DEFERRED_LOADS)


#print "Create Debug Control"
#ctrl = pydbgeng.DebugControl(c)
#ctrl.AddEngineOptions(pydbgeng.DEBUG_ENGOPT_INITIAL_BREAK)

#data = pydbgeng.DebugDataSpaces(c)
#regs = pydbgeng.DebugRegisters(c)
#adv  = pydbgeng.DebugAdvanced(c)
#ds = pydbgeng.DebugSystems(c)

#ctx = adv.GetThreadContext(pydbgeng.IMAGE_FILE_MACHINE_AMD64)
# ctx.Rip += 20
# adv.SetThreadContext(pydbgeng.IMAGE_FILE_MACHINE_AMD64, ctx)
#regs.OutputRegisters()
#ctrl.OutputDisassembly(regs.GetValue(16))
#ctrl.OutputDisassemblyLines(5, 10, regs.GetValue(16))

#ctrl.SetExecutionStatus(pydbgeng.DEBUG_STATUS_GO)
#ctrl.WaitForEvent(-1)
```

```

import pywindbg
w = pywindbg.Userdbg()
w.events.module_load(verbose=True)
w.events.exception(verbose=True)
#w.create(r"c:\windows\system32\calc.exe", False)
w.create(r"C:\Program Files (x86)\Internet Explorer\iexplore.exe", False)
w.events.nomodule_load()
w.go(1)


#pywindbg.iter_mod(w, 'lpk', pywindbg.find_branch)
x = pywindbg.iter_mod(w, 'lpk', pywindbg.find_mov)



```