IDAPython 6.8

```
long char void bool iterator
```

See file `IDA_DIR\python\idc.py`

Some constants

```
BADADDR = 0xffffffffL
BADSEL = 0xffffffffL
MAXADDR = 0xFF000000
SIZE_MAX = 0xffffffffL
```

User interface

```
Message(format, ...)        AskStr(default, prompt)
Warning(format, ...)        AskFile(doSave, mask, prompt)
ChooseFunction(title)       AskYN(default, prompt)
```

GUI-dialogs

```
Jump(ea)
ScreenEA()
here()
```

Get or set current position

Debugger Hooks

```
AddBpt(ea) / DelBpt(ea) / EnableBpt(ea)
GetBptQty() / GetBptEA(n)
SetRegValue(val, reg) / GetRegValue(reg)
```

Analysis

```
AnalyseArea(sEA, eEA) / AutoMark(ea, QType)
DeleteAll() / AutoMark2(start, end, QType)
Demangle(name, disableMask)
```

Cross references

```
CodeRefsTo(ea, flow)
CodeRefsFrom(ea, flow)
```

Code refs to/from address

`flow` Need follow the code?

```
DataRefsTo(ea)
DataRefsFrom(ea)
```

Data refs to/from address

```
XrefsFrom(ea, flags=0)
XrefsTo(ea, flags=0)
```

All references to/from address

Functions

```
LocByName(name)
```

Line address of label *name*

```
Functions(startEA, endEA)
NextFunction(ea) / PrevFunction(ea)
```

Iterate over functions

```
GetFunctionName(ea)
```

Return function name or empty string

```
GetFuncOffset(ea)
```

Return *FuncName+OffsetFromFuncStart*

```
MakeFunction(begin, end)
DelFunction(ea)
SetFunctionEnd(ea, end)
```

Manage function

```
SetFunctionCmt(ea, cmt, rpt)
GetFunctionCmt(ea, rpt)
```

Function comment

`rpt` 0 – Non repeatable, 1 - Repeatable

Manage function attributes

```
GetFunctionFlags(ea)
SetFunctionFlags(ea, flags)
```

FUNC_NORET	FUNC_FAR
FUNC_LIB	FUNC_STATIC
FUNC_FRAME	FUNC_USERFAR
FUNC_HIDDEN	

Stack pointer interaction (SP-register)

```
GetSpd(ea) / GetSpDiff(ea) / SetSpDiff(ea, delta)
```

Function frame interaction

```
GetFrame(ea) / MakeFrame(ea, nVars, nRegs, nArgs)
```

`GetFrameSize(ea)`	Whole frame
`GetFrameRegsSize(ea)`	Registers in frame
`GetFrameLvarSize(ea)`	Local variables
`GetFrameArgsSize(ea)`	Arguments

Size (in bytes)

Search in database

```
FindBinary(ea, flags, binary)
FindText(ea, flags, row, col, text)
FindImmediate(ea, flag, value)
FindCode(ea, flags) / FindFuncEnd(ea)
FindVoid(ea, flag) / FindData(ea, flags)
```

`id` – unique identifier, `idx` – index,
`rpt` – is this element repeatable?,
`mOff` – struct field offset

Enums

```
GetEnumQty()                GetnEnum(idx)
GetEnumIdx(id)              GetEnum(name)
GetEnumName(id) /           GetEnumSize(id)
GetEnumCmt(id, rpt)
GetConstByName(name) / GetConstValue(id)
GetConstName(id) / GetConstCmt(id, rpt)
```

Read/Write in database

```
*Byte(ea)          Patch*Byte(ea, value)
*Word(ea)          PatchWord(ea, value)
*Dword(ea)         PatchDword(ea, value)
isLoaded(ea)
```

Actual address? / * `Dbg` prefix may be

Instructions

`FuncItems(start)`	Function insrtuctions' addresses
`ItemHead(ea)`	Instruction/data start address
`ItemEnd(ea)`	Instruction/data end address
`ItemSize(ea)`	Item size from *ea* to the end

Colors (of background)

```
GetColor(ea, what)
SetColor(ea, what, color)
```

`what`

```
CIC_ITEM (1)
CIC_FUNC (2)
CIC_SEGM (3)
```

```
RGB (hex 0xBBGGRR)
```

Entry points

```
AddEntryPoint(ord, ea, name, makeCode)
RenameEntryPoint(ord, name) / GetEntryPoint(ord)
GetEntryOrdinal(index) / GetEntryPointQty()
```

Segments

```
Segments()
```

Iterate over segments

```
FirstSeg() / NextSeg(ea)
SegName(ea) / SegByName(name)
SegStart(ea) / SegEnd(ea)
```

Structures

```
AddStrucEx(idx, name, isUnion) / DelStruc(id) / IsUnion(structId)
GetStrucQty() / GetStrucId(idx) / GetStrucIdByName(name) / GetStrucSize(id)
GetStrucIdx(id) / GetStrucName(id) / GetStrucComment(id, rpt)
```

Structure

```
AddStrucMember(id, name, mOff, flag, typeid, nbytes)
SetMemberName(id, mOff, name) / SetMemberComment(id, mOff, cmnt, rpt)
GetMemberQty(id) / GetMemberName(id, mOff) / GetMemberComment(id, mOff, rpt)
GetMemberSize(id, mOff) / GetMemberStrId(id, mOff) / DelStrucMember(id, mOff)
```

Structure fields

Listing, comments, operands

```
MakeComm(ea, cmnt)
MakeRptCmt(ea, cmnt)
CommentEx(ea, type)
```

Comments

`type` 0 - usual, 1 - repeatable

```
ExtLin*(ea, n, line)
DelExtLn*(ea, n)
Line*(ea, n)
```

Multiline comment in code

* – *should be `A` or `B`*
`n` – Line number

```
MakeCode(ea)
MakeByte(ea)
MakeUnkn(ea, flags)
```

Word
Dword

```
DOUNK_SIMPLE     0
DOUNK_EXPAND     1
DOUNK_DELNAMES   2
```

Code / data interpretation

Strings

```
MakeStr(begin, end)
GetString(ea, len=-1, sType=0)
```

```
ASCSTR_TERMCHR  0   ASCSTR_C       0
ASCSTR_PASCAL   1   ASCSTR_LEN2    2
ASCSTR_UNICODE  3   ASCSTR_LEN4    4
ASCSTR_ULEN2    5   ASCSTR_ULEN4   6
ASCSTR_LAST     6
```

```
MakeName(ea, name)
Name(ea)
```

Name (label) of code or data

Listing interaction

`GetDisasm(ea)`	`mov   ebp, 100h`
`GetMnem(ea)`	`mov`
`GetOpnd(ea, n)`	`0) ebp 1) 100h`
`GetOperandValue(ea, n)`	`0) 0    1) 256`
`GetOpType(ea, n)`	Depends on architecture

Operand interpretation

```
OpBinary(ea, n)          OpOctal(ea, n)
OpDecimal(ea, n)         OpHex(ea, n)
OpChr(ea, n)             OpSign(ea, n)
OpSeg(ea, n)             OpStkvar(ea, n)
OpOff(ea, n, base) / OpOffEx /
OpEnumEx(ea, n, enumid, serial)
OpStroffEx(ea, n, structId, delta)
OpAlt(ea, n, val)
AltOp(ea, n)
```

User-defined string operand

`n` Number (from zero), *-1* - All
`base` Base address (in bytes)

Pavel Rusanov
v.1.0

IDAPython 7.x

Python 2.7

long char void bool iterator

More info in modules at IDA_DIR\python\

Backward compatibility: IDA_DIR\python\idc_bc695.py

Some constants

BADADDR = BADSEL = 0xffffffffL
MAXADDR = 0xFF000000
SIZE_MAX = 0xffffffffL

Analysis

plan_and_wait(start, end) *AnalyzeRange*
auto_mark_range(start, end, QType)
delete_all_segments()
demangle_name(name, disableMask)

Entry points

*Entries()
add_entry(ord, ea, name, makeCode)
rename_entry(ord, name) | get_entry(ord)
get_entry_ordinal(index) | get_entry_qty()

Cross references (XRef)

= Code refs = *to / from addr*
*CodeRefsTo(ea, flow)
*CodeRefsFrom(ea, flow)

Follow normal code flow *?*

= Data refs = *to / from addr*
*DataRefsTo(ea)
*DataRefsFrom(ea)

= All refs = *to / from addr*
*XrefsTo(ea, flags=0)
*XrefsFrom(ea, flags=0)

```
ida_xref.XREF_*
USER TAIL BASE MASK PASTEND ALL FAR DATA
```

*
idautils
High level module

Functions

iterate = over = functions
*Functions(ea, end)
get_ prev / next _func(ea)

get_name_ea_simple(name) Line address of *name*
get_func_name(ea) Func. name or empty string
get_func_off_str(ea) *'funcname+offset'* string

= Manage function = set_func_end(ea, end)
add_func(ea, end=BADADDR) | del_func(ea)

= Function comment = get_func_cmt(ea, rpt)
set_func_cmt(ea, cmt, rpt)

Not repeatable	(0)
Repeatable	(1)

= Manage function attributes =
get_func_attr(ea, attr)
set_func_attr(ea, attr, val)

```
      FUNCATTR_*
START  END     FLAGS  FRAME
FRSIZE FRREGS  REFQTY FPD
COLOR  OWNER   ARGSIZE
```

= Stack pointer interaction = get_sp_delta(ea)
add_user_stkpnt(ea, delta) | get_spd(ea)

= Function frame interaction =
set_frame_size(ea, lvsize, frregs, args)
Local vars *Saved regs* *Args size*

get_frame_size(ea)	Whole frame	*Size (bytes)*
get_frame_regs_size(ea)	Registers	
get_frame_lvar_size(ea)	Local vars	
get_frame_args_size(ea)	Arguments	

get_frame_id(ea) ID of function frame structure

User interface

= Get / set current position =
get_screen_ea() | here() | jumpto(ea)

= GUI-dialogs = msg(msg) | warning(msg)
ask_yn(dflt, prompt) | choose_func(title)
idaapi.ask_str(dflt, history, prompt)
idaapi.ask_file(doSave, mask, prompt)

Search in database

find_binary(ea, flag, bin, rdx=16)
find_text(ea, flag, row, col, text)
find_imm(ea, flag, value)
find_ code / data / suspop (ea, flag)
find_func_end(ea)

```
SEARCH_*  UP  DOWN
NEXT  CASE  REGEX
NOBRK NOSHOW
```

Items

next / prev _addr(ea) | next / prev _not_tail(ea)
get_item_ head / end (ea) *item* start/end addr*
get_item_size(ea) *from* **ea** *to the end*
next / prev _head(ea, maxea=BADADDR / minea=0)
*FuncItems(ea) *Function items' addrs*
*Heads(ea=None, end=None) *Items' addrs from* **ea** *to* **end**

item – instruction or data

idc
idaapi
Modules for most functions

Segments

get_segm_attr(ea, attr) | get_segm_ start / end (ea)
set_segm_attr(ea, attr, value)

```
  SEGATTR_*
START , END   , ORGBASE , ALIGN , COMB , PERM , BITNESS ,
FLAGS , SEL , ES , CS  , SS  , DS  , FS  , GS  , TYPE , COLOR
```

get_segm_name(ea) | selector_by_name(name)
= iterate over segments = *Segments()
get_first_seg() | get_next_seg(ea)

Read / Write in database

get_wide_ byte / word / dword (ea) | patch_ byte / word / dword / qword (ea, val)
get_qword(ea)
get_bytes(ea, size, useDbg=False)
is_loaded(ea) *is byte initialized?*

Colors (background)

get_color(ea, what)
set_color(ea, what, color)

```
CIC_*
*ITEM (1)
*FUNC (2)
*SEGM (3)
```

```
 RGB
 hex
0xBBGGRR
```

Debugger Hooks

add_bpt(ea, size=0, type=BPT_DEFAULT) | del_bpt(ea)
enable_bpt(ea, flag)
get_bpt_qty() | get_bpt_ea(n)

```
BPT_*
WRITE (1) RDWR (3) SOFT (4)
EXEC (8) DEFAULT (12)
```

get_reg_value(reg) | read_dbg_memory(ea, size)
set_reg_value(val, reg) | write_dbg_memory(ea, data)
step_ into / over () | run_to(ea) | read_dbg_ byte / word / dword / qword (ea)
step_until_ret()
start_process(path, args, sdir)

Enums

get_enum_qty() | getn_enum(idx)
get_enum_ idx / size (id)
get_enum _member_by_name (name)
get_enum_member_value(id)
get_enum_ member_name (id)
get_enum_ member_cmt (id, rpt)

id *– unique identifier*
idx *– index*
rpt *– repeatable element?*
mOff *– struct field (member) offset*

Structures

= Structure = add_struc(idx, name, isUnion)
*Structs() | del_struc(id) | is_union(id)
get_struc_qty() | get_struc_by_idx(idx)
get_struc_id(name) | get_struc_name(id)
get_struc_ idx / size (id) | get_struc_cmt(id, rpt)

= Structure fields = get_member_qty(id)
get_member_ size / strid (id, mOff)
get_member_name(id, mOff)
get_member_cmt(id, mOff, rpt)
set_member_name(id, mOff, name)
set_member_cmt(id, mOff, cmnt, rpt)

add_struc_member(id, name, mOff, flag, type, nbytes, tgt=-1, tdelta=0, rtype=REF_OFF32)
del_struc_member(id, mOff)

FF_* const. see below · *Depends on* ***flag*** · *Size of member* · *Target addr.* · *Offset target delta* · *REF_* constant see below*

Listing, comments, operands

n Number (from zero) -1 - All

= Comments = set_cmt(ea, cmnt, rpt)
get_cmt(ea, rpt) rpt 0 - usual / 1 - repeatable

Multiline = comment = in code
Line number
get_extra_cmt(ea, n)
del_extra_cmt(ea, n)
update_extra_cmt(ea, n, line)

= Code / data view = create_insn(ea)
create_ byte / word / dword / qword (ea)

```
DELIT_*   SIMPLE   (0)
EXPAND (1) DELNAMES (2)
```

del_items(ea, flags=0, size=1)
make_array(ea, n) *Array of* ***n*** *items with type of item at* ***ea***
create_data(ea, flags, size, tid) *Struct ID*

```
FF_*  BYTE  WORD  DWORD QWORD TBYTE OWORD
STRLIT STRUCT FLOAT  DOUBLE PACKREAL ALIGN
```

= Name of code / data = *Names()
set_name(ea, name, sn_flags=SN_CHECK)
get_name(ea, gtn_flags=0)

```
GN_*    VISIBLE
COLORED DEMANGLED
STRICT  SHORT LONG
LOCAL   ISRET NOT_ISRET
```

```
SN_*   LOCAL  CHECK  NOCHECK
PUBLIC NON_PUBLIC    WEAK
NON_WEAK  AUTO  NON_AUTO
NOLIST    NOWARN
```

= Strings = *Strings() | get_str_type(ea)
get_strlit_contents(ea, len=-1, sType=0)
ida_bytes.create_strlit(ea, len, sType) | *Create string with default global type* create_strlit(ea, end)

```
   STRTYPE_*
TERMCHR (0) C (0) C16 (1) C_32 (2) PASCAL (64) PASCAL_16 (65)
LEN2 (128) LEN2_16 (129) LEN4 (192) LEN4_16 (193)
```

```
end=BADADDR
for auto-end
```

= Listing interaction =

"mov ebp, 100h"	generate_disasm_line(ea, flags)
"mov"	print_insn_mnem(ea)
0)"ebp" 1)"100h"	print_operand(ea, n)
0) 0 1) 256	get_operand_value(ea, n)
Depends on arch.	get_operand_type(ea, n)

```
GENDSM_*
FORCE_CODE
MULTI_LINE
```

Manual representation set_manual_insn(ea, insn) / get_manual_insn(ea)

= Operand interpretation =
toggle_ sign / bnot (ea, n) *Change sign / bitwise not*
op_ bin / dec / chr / seg / oct / hex / stkvar (ea, n)
op_stroff(ea, n, stId, delta)
op_enum(ea, n, enmid, serial)
op_plain_offset(ea, n, base)
op_offset(ea, n, rtype, tgt, base, tdelta)
expression target · *the offset base* · *displacement from the target*

```
REF_*
OFF8   OFF16  OFF32
LOW8   LOW16  OFF64
HIGH8  HIGH16
```

User-defined operand op_man(ea, n, val)
get_forced_operand(ea, n)

Pavel Rusanov

v.1.0