

gdb 入门

1 GDB

1.1

```
SH$ gdb program          # program[...]  
$ gdb program core       # gdb[...core[...]  
$ gdb program pid        # [...]D[gdb[...attach[...program[...PATH[...]
```

1.2

```
SH(gdb) r/run            # [...]  
(gdb) c/continue         # [...]  
(gdb) n/next             # [...]  
(gdb) s/step             # [...]  
(gdb) ni/si              # [...]ni[...si[...]  
(gdb) fini/finish        # [...]fini[...finish[...]  
(gdb) u/util              # [...]u[...]  
  
(gdb) set args           # [...]set args 10 20 30  
(gdb) show args          # [...]  
(gdb) path <dir>         # [...]  
(gdb) show paths         # [...]  
(gdb) set env <name=val> # [...]set env USER=chen  
(gdb) show env [name]    # [...]  
(gdb) cd <dir>           # [...]shell[...cd[...]  
(gdb) pwd                # [...]  
  
(gdb) shell <command>    # [...]shell[...]
```

2 断点(breakpoint)

```
SH(gdb) b/break linenum/func # [...]linenum[...function[...]  
(gdb) b/break +/-offset     # [...]offset[...]  
(gdb) b/break filename:linenum # [...]filename[...linenum[...]  
(gdb) b/break filename:func  # [...]function[...]  
(gdb) b/break *address       # [...]address[...]  
(gdb) b/break                # [...]  
(gdb) b/break if <condition> # [...]break if i=100  
  
(gdb) info break [n]         # [...] n[...]
```

□□/□□□□(watchpoint)

```

SH# [20 slots]
(gdb) watch <expr>          # [6 slots]
(gdb) rwatch <expr>         # [1+6 slots]
(gdb) awatch <expr>         # [1+6+6 slots]
(gdb) info watchpoints      # [6 slots]

```

□□/□□□□(catchpoint)

```
SH# [ ]C++ [ ]
(gdb) tcatch <event> # [ ]
(gdb) catch <event> # [ ]event[ ]
# throw [ ]c++[ ]throw[ ]
# catch [ ]C++[ ]catch[ ]
# exec [ ]exec[ ]HP-UX[ ]
# fork [ ]fork[ ]HP-UX[ ]
# vfork [ ]vfork[ ]HP-UX[ ]
# load [file] [ ]HP-UX[ ]
# unload [libname] [ ]HP-UX[ ]
#
```

--	--	--	--	--

```
SH# [ ]delete[ ]clear[ ]disable[ ]enable[ ]
(gdb) clear # [ ]
(gdb) clear <function> # [ ]
(gdb) clear <file:line> # [ ]
(gdb) d/delete [n]/[m-n] # [ ]m-n

# [ ]disable[ ]disable[ ]GDB[ ]enable[ ]
(gdb) disable [n]/[m-n] # disable[ ]n[ ]disable[ ]m-n

(gdb) enable [n]/[m-n] # enable[ ]n[ ]m-n
(gdb) enable once [n]/[m-n] # enable[ ]n[ ]disable[ ]m-n
(gdb) enable delete [n]/[m-n] # enable[ ]m-n
```

--	--	--	--	--	--	--

```

SH# [ ]breakpoint[ ]
(gdb) condition <bunm> <expr> # [ ]bnum[ ]
(gdb) condition <bnum> # [ ]bnum[ ]

# ignore [ ]
(gdb) ignore <bnum> <count> # [ ]bnum[ ]count[ ]

```



```
SH# [ ]command[ ]
(gdb) commands [bnum]
> ... commands list ...
> end          # [ ]bnum[ ]

[ ]
(gdb) break foo if x>0
(gdb) commands
> printf "x is %dn",x
> continue
> end
# [ ]foo[ ]x>0[ ]x[ ]foo[ ]0[GDB[ ]x[ ]
# [ ]commands[ ]end[ ]
```



```
SH# [ ]c++[ ]gdb[ ]
[ ]
(gdb) b String::after
[0] cancel
[1] all
[2] file:String.cc; line number:867
[3] file:String.cc; line number:860
[4] file:String.cc; line number:875
[5] file:String.cc; line number:853
[6] file:String.cc; line number:846
[7] file
> 2 4 6
Breakpoint 1 at 0xb26c: file String.cc, line 867.
Breakpoint 2 at 0xb344: file String.cc, line 875.
Breakpoint 3 at 0xafcc: file String.cc, line 846.
```



```
SH# [ ]c/continue[ ]step[ ]next[ ]
(gdb) c/continue [ignore-count]  # [ ]ignore-count[ ]

# [ ]debug[ ]VC[ ]stepin[ ]count[ ]count[ ]
(gdb) step <count>

# [ ]step-mode[ ]debug[ ]
(gdb) set step-mode on

# [ ]
(gdb) u/until
```

```
# [redacted]stepi [redacted]nexti [redacted]"display/i $pc" [redacted]
(gdb) si/stepi
(gdb) ni/stepi
```

❏

```
SH# [redacted]UNIX [redacted]UNIX [redacted]SIGINT [redacted]Ctrl+C [redacted]S
# GDB [redacted]GDB [redacted]GDB [redacted]GDB [redacted]handle [redacted]
(gdb) handle <signal> <keywords...>
# [redacted]GDB [redacted]<signal> [redacted]SIG [redacted]SIG [redacted]SIGIO-SIGKILL [redacted]SIGIO [redacted]SIGKILL [redacted]SIGIO
nstop      # [redacted]GDB [redacted]
stop       # [redacted]GDB [redacted]
print      # [redacted]GDB [redacted]
noprint     # [redacted]GDB [redacted]
pass/noignore # [redacted]GDB [redacted]GDB [redacted]
nopass/ignore # [redacted]GDB [redacted]

# [redacted]GDB [redacted]
(gdb) info signals
(gdb) info handle
```

❏❏❏❏

```
SH# [redacted]singal [redacted]Ctrl+C [redacted]GDB [redacted]
(gdb) signal <singal>
# UNIX [redacted]1 [redacted]15 [redacted]<singal> [redacted]
# single [redacted]shell [redacted]kill [redacted]kill [redacted]GDB [redacted]single [redacted]
```

❏❏

```
SH# [redacted]
(gdb) info threads      # [redacted]
(gdb) break <line> thread <threadno>      # [redacted]line [redacted]threadno [redacted]
(gdb) break <line> thread <threadno> if... # [redacted]line [redacted]threadno [redacted]

[redacted]
(gdb) break frik.c:13 thread 28 if bartab > lim
```

❏❏❏❏

```
SH# [redacted]"[redacted]"Stack [redacted]GDB [redacted]
(gdb) bt/backtrace      # [redacted]
(gdb) bt/backtrace <n>  # [redacted]n [redacted]n [redacted]n [redacted]

# [redacted]
(gdb) f/frame <n>      # n [redacted]0 [redacted]
```

```
(gdb) up <n>          # [ ]n[ ]n[ ]
(gdb) down <n>         # [ ]n[ ]n[ ]
```

```
# [ ]
```

```
(gdb) f/frame
```

```
# [ ]
```

```
(gdb) info f/frame
```

```
(gdb) info args      # [ ]
```

```
(gdb) info locals    # [ ]
```

```
(gdb) info catch     # [ ]
```

[]

```
SH# [ ]-g[ ]
```

```
(gdb) l              # [ ]
```

```
(gdb) l -             # [ ]
```

```
(gdb) l +             # [ ]
```

```
(gdb) l/list <linenum/func> # [ ]linenum[ ]function[ ]10
```

```
(gdb) l/list          # [ ]list[ ]10
```

```
(gdb) l/list m,n      # [ ]m[ ]n[ ]m[ ]n
```

```
(gdb) l/list -/+offset # [ ]offset[ ]
```

```
(gdb) l/list <file:line> # [ ]file[ ]line[ ]
```

```
(gdb) l/list <func>     # [ ]func[ ]
```

```
(gdb) l/list <file:func> # [ ]file[ ]func[ ]
```

```
(gdb) l/list <*address> # [ ]address[ ]
```

```
(gdb) set listsize     # [ ]
```

```
(gdb) show listsize    # [ ]listsize[ ]
```

[]

```
SH(gdb) forward-search <regexp> # [ ]regexp[ ]
```

```
(gdb) search <regexp>          # [ ]
```

```
(gdb) reverse-search <regexp>  # [ ]
```

[]

```
SH# [ ]-g[ ]GDB[ ]GDB[ ]
```

```
(gdb) dir/directory <dirname ... > # [ ]UNIX[ ]:"[ ]Windows[ ]";[ ]
```

```
(gdb) dir/directory          # [ ]
```

```
(gdb) show directories      # [ ]
```

[]

```

SH(gdb) info line          # [ ]:[ ]:
(gdb) info line <num>
(gdb) info line <file:num>
(gdb) info line <func>
(gdb) info line <file:func>

# [ ]disassemble[ ]dump[ ]func[ ]
(gdb) disassemble func

```

[]

```

SH# <expr>[ ]GDB[ ]<f>[ ]16[ ]/x[ ]
(gdb) p/print <expr>      # expr[ ]const[ ]
(gdb) p/print /<f> <expr>

# [ ]GDB[ ]
@      [ ]
::      [ ]
{<type>} <addr>  [ ]<addr>[ ]type[ ]
# [ ]GDB[ ]
SH# [ ]
# [ ]GDB[ ]GDB[ ]GDB[ ]
x [ ]
d [ ]
u [ ]
o [ ]
t [ ]
a [ ]
c [ ]
f [ ]

[ ]
(gdb) p i
$21 = 101
(gdb) p/a i
$22 = 0x65

```

[]

```

SH# [ ]examine[ ]x[ ]x[ ]
(gdb) x/<n/f/u> <addr>      # n, f, u[ ]

n      [ ]
f      [ ]s[ ]i[ ]
u      [ ]GDB[ ]4[ ]bytes[ ]u[ ]b[ ]h[ ]w[ ]g[ ]GDB[ ]
<addr> [ ]

# n/f/u[ ]
(gdb) x/3uh 0x54320          # [ ]0x54320[ ]h[ ]3[ ]u[ ]

```



```
SH# GDB display
(gdb) display <expr>
(gdb) display/<fmt> <expr>
(gdb) display/<fmt> <addr>
# expr<fmt><addr> display GDB

# i s display
(gdb) display/i $pc
# $pc GDB/i
display GDB
(gdb) undisplay <dnums...>
(gdb) delete display <dnums...>
# dnums
# 2-5
(gdb) disable display <dnums...>
(gdb) enable display <dnums...>
# disable enable
(gdb) info display
# display GDB enable
```



```
SH# GDB
# 1 GDB
(gdb) set print address
(gdb) set print address on
(gdb) set print address off # 
(gdb) show print address # 

# 2
(gdb) set print array
(gdb) set print array on
(gdb) set print array off
(gdb) show print array
(gdb) show print elements # print elements
(gdb) set print elements <number-of-elements>
# <number-of-elements> GDB 0

# 3 off
(gdb) set print null-stop <on/off>

# 4 printf pretty GDB
(gdb) set print pretty on
(gdb) show print pretty # GDB

# 5 "nnn" nnn "65"
(gdb) set print sevenbit-strings <on/off>
(gdb) show print sevenbit-strings #
```

```
# 6
(gdb) set print union <on/off>
(gdb) show print union      # 

```

```
$1 = {it = Tree, form = {tree = Acorn, bug = Cocoon}}  # 
$1 = {it = Tree, form = {...}}                      # 
```

```
# 7 C++GDBGDBGoff
(gdb) set print object <on/off>
(gdb) show print object    # 
```

```
# 8 C++on
(gdb) set print static-members <on/off>
(gdb) show print static-members  # 
```

```
# 9GDBG
(gdb) set print vtbl <on/off>
(gdb) show print vtbl        # 
```

```


```

```
GDBGprintprintGDBGDBG$1, $2, $3 .....print$1
```

```


```

```
SH# GDBGDBGDBGsetGDBGUNIX$
(gdb) set $foo = *object_ptr
# GDBG
(gdb) show convenience
# 
(gdb) set $i = 0
(gdb) print bar[$i++]->contents
# print bar[0]->contents, print bar[1]->contents
```

```


```

```
SH# ipspprint$peip
(gdb) info registers      # ( )
(gdb) info all-registers  # 
(gdb) info registers regname# info rbp
```

```


```

```
SH# GDBGDBG
# 
```

```
# [REDACTED]GDB[REDACTED]GDB print[REDACTED]
(gdb) print x=4
# x=4[REDACTED]C/C++[REDACTED]x[REDACTED]4[REDACTED]Pascal[REDACTED]Pascal[REDACTED]x:=4[REDACTED]
# [REDACTED]GDB[REDACTED]
(gdb) whatis width
type = double
(gdb) p width
$4 = 13
(gdb) set width=47
Invalid syntax in expression.
# [REDACTED]set width[REDACTED]GDB[REDACTED]"Invalid syntax in expression"[REDACTED]set
# var[REDACTED]GDB width[REDACTED]GDB[REDACTED]
(gdb) set var width=47
# [REDACTED]GDB[REDACTED]set var[REDACTED]GDB[REDACTED]
```

[REDACTED]

```
SH# [REDACTED]GDB[REDACTED]GDB[REDACTED]GDB jump[REDACTED]
(gdb) jump <linespec>
# [REDACTED]<linespec>[REDACTED]file:line[REDACTED]+num[REDACTED]
(gdb) jump <address>
# [REDACTED]<address>[REDACTED]jump[REDACTED]C
# [REDACTED]jump[REDACTED]"set $pc"[REDACTED]
(gdb) set $pc = 0x485
```

[REDACTED]

```
SH# [REDACTED]return[REDACTED]
(gdb) return
(gdb) return <expression>
# [REDACTED]return[REDACTED]<expression>[REDACTED]
```

[REDACTED]

```
SH(gdb) call <expr>
# [REDACTED]void[REDACTED]
# [REDACTED]—print[REDACTED]print[REDACTED]print[REDACTED]call[REDACTED]void[REDACTED]call[REDACTED]print[REDACTED]
```

GDB [REDACTED]

```
SH(gdb) show language
# [REDACTED]GDB[REDACTED]C[REDACTED]
(gdb) info frame
# [REDACTED]
(gdb) info source
```

```
# GDBset languageset languageGDB
(gdb) set language
The currently understood settings are:
local or auto Automatic setting based on source file
c Use the C language
c++ Use the C++ language
asm Use the Asm language
chill Use the Chill language
fortran Use the Fortran language
java Use the Java language
modula-2 Use the Modula-2 language
pascal Use the Pascal language
scheme Use the Scheme language
# set language
```



```
(gdb) q/quit      # GDB
```