

GDB cheatsheet - page 1

Running

```
# gdb <program> [core dump]
    Start GDB (with optional core dump).

# gdb --args <program> <args...>
    Start GDB and pass arguments

# gdb --pid <pid>
    Start GDB and attach to process.

set args <args...>
    Set arguments to pass to program to
    be debugged.

run
    Run the program to be debugged.

kill
    Kill the running program.
```

Breakpoints

```
break <where>
    Set a new breakpoint.

delete <breakpoint#>
    Remove a breakpoint.

clear
    Delete all breakpoints.

enable <breakpoint#>
    Enable a disabled breakpoint.

disable <breakpoint#>
    Disable a breakpoint.
```

Watchpoints

```
watch <where>
    Set a new watchpoint.

delete/enable/disable <watchpoint#>
    Like breakpoints.
```

<where>

function_name
Break/watch the named function.

line_number
Break/watch the line number in the current source file.

file:line_number
Break/watch the line number in the named source file.

Conditions

```
break/watch <where> if <condition>
    Break/watch at the given location if the
    condition is met.
    Conditions may be almost any C ex-
    pression that evaluate to true or false.

condition <breakpoint#> <condition>
    Set/change the condition of an existing
    break- or watchpoint.
```

Examining the stack

```
backtrace
where
    Show call stack.

backtrace full
where full
    Show call stack, also print the local va-
    riables in each frame.

frame <frame#>
    Select the stack frame to operate on.
```

Stepping

```
step
    Go to next instruction (source line), di-
    ving into function.
```

```
next
    Go to next instruction (source line) but
    don't dive into functions.

finish
    Continue until the current function re-
    turns.

continue
    Continue normal execution.
```

Variables and memory

```
print/format <what>
    Print content of variable/memory locati-
    on/register.

display/format <what>
    Like „print“, but print the information
    after each stepping instruction.

undisplay <display#>
    Remove the „display“ with the given
    number.

enable display <display#>
disable display <display#>
    En- or disable the „display“ with the gi-
    ven number.

x/nfu <address>
    Print memory.
    n: How many units to print (default 1).
    f: Format character (like „print“).
    u: Unit.

    Unit is one of:
        b: Byte,
        h: Half-word (two bytes)
        w: Word (four bytes)
        g: Giant word (eight bytes)).
```

Format

<i>a</i>	Pointer.
<i>c</i>	Read as integer, print as character.
<i>d</i>	Integer, signed decimal.
<i>f</i>	Floating point number.
<i>o</i>	Integer, print as octal.
<i>s</i>	Try to treat as C string.
<i>t</i>	Integer, print as binary (<i>t</i> = „two“).
<i>u</i>	Integer, unsigned decimal.
<i>x</i>	Integer, print as hexadecimal.

<what>*expression*

Almost any C expression, including function calls (must be prefixed with a cast to tell GDB the return value type).

file_name::variable_name

Content of the variable defined in the named file (static variables).

function::variable_name

Content of the variable defined in the named function (if on the stack).

{type}address

Content at *address*, interpreted as being of the C type *type*.

\$register

Content of named register. Interesting registers are \$esp (stack pointer), \$ebp (frame pointer) and \$eip (instruction pointer).

Threads

thread <thread#>

Chose thread to operate on.

Manipulating the program

```
set var <variable_name>=<value>
    Change the content of a variable to the
    given value.

return <expression>
    Force the current function to return im-
    mediately, passing the given value.
```

Sources

```
directory <directory>
    Add <directory> to the list of directories
    that is searched for sources.
```

list

```
list <filename>:<function>
list <filename>:<line_number>
list <first>,<last>
    Shows the current or given source con-
    text. The <filename> may be omitted. If
    <last> is omitted the context starting at
    <start> is printed instead of centered a-
    round it.
```

```
set listsize <count>
    Set how many lines to show in „list“.
```

Signals

```
handle <signal> <options>
    Set how to handle signles. Options are:

    (no)print: (Don't) print a message when
    signals occurs.

    (no)stop: (Don't) stop the program
    when signals occurs.

    (no)pass: (Don't) pass the signal to the
    program.
```

Informations

```
disassemble
disassemble <where>
    Disassemble the current function or
    given location.
```

info args

Print the arguments to the function of the current stack frame.

```
info breakpoints
    Print informations about the break- and
    watchpoints.
```

```
info display
    Print informations about the „displays“.
```

info locals

Print the local variables in the currently selected stack frame.

```
info sharedlibrary
    List loaded shared libraries.
```

```
info signals
    List all signals and how they are cur-
    rently handled.
```

```
info threads
    List all threads.
```

```
show directories
    Print all directories in which GDB sear-
    ches for source files.
```

```
show listsize
    Print how many are shown in the „list“
    command.
```

```
what is <variable_name>
    Print type of named variable.
```