



GDB
The GNU Project
Debugger

Presenter: Phil Muldoon

IRC: gdb at freenode

EMAIL: gdb@sourceware.org

- Current release 7.6.1
- New release 7.7 “real soon now”.

GDB

- Frame filters and frame decorators.
- Temporary breakpoints supported.
- Line tables representation.
- Types can be pretty-printed via a Python API.
- Python 3 is now supported (in addition to Python 2.4 or later).
- New convenience variable `$_exception`. Holds the exception for an exception-related catchpoint.
- Catchpoints, like "catch throw", accept a regular expression which can be used to filter exceptions by type.

Frame filters

- Allows you to “filter out” frames from a backtrace, sort, elide, and decorate the contents of each frame.

Frame Decorator

```
import gdb
import itertools
from gdb.FrameDecorator import FrameDecorator
import copy

class UpperCase_Decorator (FrameDecorator):

    def __init__(self, fobj):
        super(UpperCase_Decorator, self).__init__(fobj)
        self.fobj = fobj

    def function (self):
        fname = str (self.fobj.function())
        fname = fname.upper()
        return fname
```

Frame Filter

```
class ExampleFrameFilter ():

    def __init__ (self):
        self.name = "Function uppercase"
        self.priority = 100
        self.enabled = True
        gdb.frame_filters [self.name] = self

    def filter (self, frame_iter):
        frame_iter = itertools.imap
        (UpperCase_Decorator,frame_iter)
        return frame_iter
```

Frame Filters

Output from example:

```
(gdb) bt full
```

```
#0 0x00000000004005ff in END_FUNC (foo=21, bar=0x40097d "Param") at test.c:47
    h = 9
    f = 42
    bar = 0x40095d "Inside block x2"
#1 0x000000000040072f in FUNCA () at test.c:49
    fb = {nothing = 0x40096d "Foo Bar", f = 42, s = 19}
#2 0x000000000040062b in FUNCB (j=10) at test.c:69
```

Elided example:

```
(gdb) bt full
```

```
#0 0x00000000004005ff in END_FUNC (foo=21, bar=0x40097d "Param") at test.c:47
    h = 9
    f = 42
    bar = 0x40095d "Inside block x2"
#1 0x000000000040072f in FUNCA () at test.c:49
    fb = {nothing = 0x40096d "Foo Bar", f = 42, s = 19}
#2 0x000000000040062b in FUNCB (j=10) at test.c:69
```

Type pretty printers

- Allows the pretty printing of types in GDB

```
class Recognizer(object):
    def __init__(self):
        self.enabled = True

    def recognize(self, type_obj):
        if type_obj.tag == 'basic_string':
            return 'string'
        return None

class StringTypePrinter(object):
    def __init__(self):
        self.name = 'string'
        self.enabled = True

    def instantiate(self):
        return Recognizer()

gdb.type_printers.append(StringTypePrinter())
```

Type pretty printers

(gdb) ptype s

```
type = class templ<string> [with T = string] {  
  public:  
    T x;  
    templ<T> *value;  
}
```

(gdb) ptype/r s

```
type = class templ<basic_string> {  
  public:  
    basic_string x;  
    templ<basic_string> *value;  
}
```

```
template<typename T>  
class templ  
{  
  public:  
    T x;  
    templ<T> *value;  
};  
  
templ<basic_string> s;
```

C++ Exception handling

- Catchpoints now take a regex, allowing you to filter what exceptions you are interested in.
- Catch catch
- Catch throw
- Catch rethrow