

Déboguer confortablement en ligne de commande

GDB, Valgrind, LLDB, PDB, IPDB, JDB



2^{ème} session LIFTech'

Auteur: Florent Jaillet

Date: 9 juin 2016

Plan

- Motivation
- Intérêt du débogage en ligne de commande
- GDB (*C, C++, Objective-C, Fortran, Go, Rust...*)
- Valgrind + GDB
- LLDB (*C, C++, Objective-C*)
- PDB et IPDB (*Python*)
- JDB (*Java*)

Motivation

- Présentation à CppCon 2015 de Greg Law : "Give me 15 minutes & I'll change your view of GDB"
- Outils en ligne de commande difficiles à apprendre et peu intuitifs, mais plus pratiques que laisse paraître leur première utilisation
- Objectif ici n'est pas de présenter en détail le fonctionnement des outils, mais plutôt d'identifier les fonctionnalités clefs qui rendent l'utilisation plus facile et pratique
- Utiliser ce support de présentation comme mémo pour creuser les points intéressants par la suite

Intérêt du débogage en ligne de commande

- Ligne de commande disponible partout (par défaut sous GNU/Linux, BSD, MAC OS X, [Windows Subsystem for Linux](#) sous Windows 10, [installable](#) sous android)
- Parfois la seule solution (machine distante, grappe de calcul, système embarqué...)
- Puissance des outils
- Éventuelle préférence pour la ligne de commande

GDB : Exemple d'utilisation

- Compilation :

```
g++ -g example.cpp -o example
```

- Lancement du débogage :

```
gdb ./example
```

GDB : Utilisation basique

Aide	help
Lancement	start, run
Points d'arrêt	break , disable , enable , clear , delete
Flot d'exécution	step , next , finish , continue
Affichage informations	print , backtrace (<i>ou where</i>), list , info
Arrêt et fin	kill , quit

GDB : Plus de confort avec tui

- Interface avec fenêtres textuelles multiples via le raccourci Ctrl+X+A ou la commande `tui enable`
- Informations affichables en plus de la ligne de commande de GDB :
 - code source
 - code assembleur
 - registres
- Choix de la disposition des fenêtres avec les raccourcis Ctrl+X+2 et Ctrl+X+1 ou la commande `layout`
- Raccourcis :
 - Ctrl+L : rafraichissement interface
 - Ctrl+P, Ctrl+N, Ctrl+B, Ctrl+F : remplacement des touches flèches

GDB : Plus de puissance

Points d'arrêt	watch , catch , break <where> if <condition>, condition , commands , <i>sauvegarde</i> (save, source)
Affichage	dprintf
Appel à la volée	call
Rétro-débugage	record , reverse-step , reverse-next , reverse-finish , reverse-continue
Script et extension	python , python-interactive
Historique	set history save (<i>à mettre dans \$HOME/.GDBinit</i>)
Commandes système	! (<i>ou shell</i>)

Valgrind

Suite d'outils pour l'analyse dynamique (en particulier mémoire), notamment :

- détection de l'utilisation de mémoire non allouée ou non initialisée

```
Invalid read of size 1
```

```
at 0x4005C8: main (example_valgrind.c:18)
```

```
Address 0x5202100 is 0 bytes after a block of size 16 alloc'd
```

```
at 0x4C2BBCF: malloc (vg_replace_malloc.c:299)
```

```
by 0x4005BF: main (example_valgrind.c:15)
```

- détection de fuites de mémoire

```
LEAK SUMMARY:
```

```
definitely lost: 35 bytes in 2 blocks
```

```
indirectly lost: 0 bytes in 0 blocks
```

```
possibly lost: 0 bytes in 0 blocks
```

```
still reachable: 0 bytes in 0 blocks
```

```
suppressed: 0 bytes in 0 blocks
```

- profilage de l'utilisation mémoire (`--tool=massif`)

Valgrind + GDB

Utilisation interactive de Valgrind possible via GDB :

- Lancement de Valgrind : `valgrind --vgdb-error=0 ./example`
- Lancement de GDB dans une autre invite de commande : `gdb ./example`
- Puis dans GDB : `(gdb) target remote | vgdb`

Aide	monitor help
Fuite mémoire	monitor leak-check
Validité mémoire	monitor get-vbits , monitor check_memory
Occupation mémoire	monitor snapshot <i>(nécessite l'option --tool=massif de Valgrind)</i>

Plus de détails dans la [documentation de Valgrind](#) et ce [tutoriel](#)

LLDB : Utilisation basique

	GDB	LLDB
Aide	help	help
Lancement	start, run	run -s, run
Points d'arrêt	break , disable , enable , clear , delete	breakpoint set , breakpoint disable , breakpoint enable , breakpoint clear , breakpoint delete
Flot d'exécution	step , next , finish , continue	step , next , finish , continue
Affichage informations	print , backtrace (<i>ou where</i>), list , info	print , bt , list , (frame variable , breakpoint list , watchpoint list ,...)
Arrêt et fin	kill , quit	kill , quit

LLDB : Plus de puissance 1/2

	GDB	LLDB
Multifenêtre	tui	gui
Points d'arrêt	watch , catch , break <where> if <condition>, condition , commands , <i>sauvegarde</i> (save, source)	watchpoint set variable , breakpoint set -F , breakpoint set -c , breakpoint modify -c , breakpoint command add , \emptyset
Affichage	dprintf	\emptyset
Appel à la volée	call	expression
Rétro-débugage	record ,...	\emptyset

LLDB : Plus de puissance 2/2

	GDB	LLDB
Script et extension	python , python-interactive	script , script
Historique	set history save (<i>à mettre dans \$HOME/.GDBinit</i>)	<i>disponible par défaut</i>
Commandes système	! (<i>ou shell</i>)	platform shell

[Page avec plus d'infos sur les correspondances GDB / LLDB](#)

PDB et IPDB

	GDB	PDB et IPDB
Aide	help	help
Lancement	start, run	ø, run
Points d'arrêt	break , disable , enable , clear , delete , condition , commands	break , disable, enable, clear , clear , condition, commands
Flot	step , next , finish , continue	step , next , return , continue
Affichage informations	print , backtrace (<i>ou</i> where), list , info	(p, pp, print), where , list , (dir(), locals(), globals(), break ,...)
Arrêt et fin	kill , quit	ø, quit
Appel, script	call , python , python-interactive	[!]
Système	! (<i>ou</i> shell)	os.system("<cmd>")

Voir [pdb++](#) ou [pudb](#) pour des fonctionnalités plus avancées

JDB

Fonctionnalités limitées, pas de fonctionnalités avancées :

	GDB	JDB
Aide	help	help
Lancement	start, run	ø, run
Points d'arrêt	break, disable, enable, clear, delete, watch, catch	stop, ø, ø, clear, clear, watch, catch
Flot d'exécution	step, next, finish, continue	step, next, step up, cont
Affichage informations	print, backtrace (<i>ou</i> where), list, info	(print, dump), where, list, (locals, clear,...)
Appel à la volée	call	eval (<i>ou</i> print)
Arrêt et fin	kill, quit	kill, quit