

Exemple de débogage

Nous avons rencontré un bug lors de la conception de l'interface graphique pour jouer contre l'IA. Choisir le mode difficile renvoyait une erreur de segmentation. Voici comment nous avons repéré le bug avec le débogueur gdb :

```
~/.../othello-main/src$ gdb ./othello
GNU gdb (GDB) 14.2
Copyright (C) 2023 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-unknown-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./othello...
(gdb) run
Starting program: /home/runner/workspace/othello-main/src/othello
warning: Error disabling address space randomization: Operation not permitted
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/nix/store/k7zgvpz2r31zkg9xqgjjm7mbknryv6bs-glibc-2.39-52/lib/libthread_db.so.1".
MESA: error: ZINK: vkCreateInstance failed (VK_ERROR_INCOMPATIBLE_DRIVER)
libEGL warning: egl: failed to create dri2 screen
[New Thread 0x7f36a6e176c0 (LWP 290)]
[New Thread 0x7f36a66166c0 (LWP 291)]
[New Thread 0x7f36a5e156c0 (LWP 292)]
[New Thread 0x7f36a56146c0 (LWP 293)]
[New Thread 0x7f36a4e136c0 (LWP 294)]
[New Thread 0x7f368ffff6c0 (LWP 295)]
[New Thread 0x7f368f7fe6c0 (LWP 296)]
[New Thread 0x7f368effd6c0 (LWP 297)]
[New Thread 0x7f368e7fc6c0 (LWP 298)]
[New Thread 0x7f368dffb6c0 (LWP 299)]
[New Thread 0x7f368d7fa6c0 (LWP 300)]
[New Thread 0x7f368cff96c0 (LWP 301)]
[New Thread 0x7f366ffff6c0 (LWP 302)]
[New Thread 0x7f366f7fe6c0 (LWP 303)]
[New Thread 0x7f36a40fc6c0 (LWP 306)]
[New Thread 0x7f366effd6c0 (LWP 307)]
Mode de jeu: Joueur vs Ordi - Niveau 4
Couleur sélectionnée: 0

Thread 1 "othello" received signal SIGSEGV, Segmentation fault.
0x000000003e38260 in ?? ()
(gdb) print difficultyLevel
'difficultyLevel' has unknown type; cast it to its declared type
(gdb) print (int)difficultyLevel
$1 = 3
(gdb) ]
```

erreur détectée

vérification

Le problème se trouvait dans la valeur attribuée à difficultyLevel. Les valeurs étaient passées "en dur" au lieu d'utiliser l'énumération t_niveau.

```

if (e->type == SDL_MOUSEBUTTONDOWN) {
    if (mouseX >= niveau1.x && mouseX <= niveau1.x + niveau1.l &&
        mouseY >= niveau1.y && mouseY <= niveau1.y + niveau1.h) {
        difficultyLevel = 1; // Facile
    } else if (mouseX >= niveau2.x && mouseX <= niveau2.x + niveau2.l &&
        mouseY >= niveau2.y && mouseY <= niveau2.y + niveau2.h) {
        difficultyLevel = 2; // Moyen
    } else if (mouseX >= niveau3.x && mouseX <= niveau3.x + niveau3.l &&
        mouseY >= niveau3.y && mouseY <= niveau3.y + niveau3.h) {
        difficultyLevel = 3; // Difficile
    }
}

```

Ceci posait problème car t_niveau était défini ainsi à cette phase du projet :

```

// Définition des types
typedef enum { FACILE = 0, MOYEN, DIFFICILE } t_niveau;

```

Nous avons donc corrigé le bug en attribuant les bonnes valeurs à difficultyLevel.

```

if (e->type == SDL_MOUSEBUTTONDOWN) {
    if (mouseX >= niveau1.x && mouseX <= niveau1.x + niveau1.l &&
        mouseY >= niveau1.y && mouseY <= niveau1.y + niveau1.h) {
        difficultyLevel = FACILE;
    } else if (mouseX >= niveau2.x && mouseX <= niveau2.x + niveau2.l &&
        mouseY >= niveau2.y && mouseY <= niveau2.y + niveau2.h) {
        difficultyLevel = MOYEN;
    } else if (mouseX >= niveau3.x && mouseX <= niveau3.x + niveau3.l &&
        mouseY >= niveau3.y && mouseY <= niveau3.y + niveau3.h) {
        difficultyLevel = DIFFICILE;
    }
}

```