La ligne de commande

Programmation séquentielle en C, 2023-2024

Orestis Malaspinas (A401)

2023-11-02

Informatique et Systèmes de Communication, HEPIA

La ligne de commande (1/4)

- Ou comment passer des arguments à un programme en C.

Point d'entrée d'un programme

- Le point d'entrée est la fonction `main()`.
- Elle peut être déclarée de 4 façon différentes:
  1. `void main().`
  2. `int main().`
  3. `void main(int argc, char **argv).`
  4. `int main(int argc, char *argv[]).`
- `argc` est le nombre d'arguments passés à la ligne de commande: **le premier est celui du programme lui-même**.
- `argv` est un tableau de chaînes de caractères passés sur la ligne de commande.

La ligne de commande (2/4)

Exemple d'utilisation

Pour la fonction dans le programme `prog`

```
int main(int argc, char **argv)
```

Pour l'exécution suivante on a

```
$ ./prog -b 50 file.txt
```

```
argc == 4
argv[0] == "prog"
argv[1] == "-b"
argv[2] == "50"
argv[3] == "file.txt"
```

La ligne de commande (3/4)

Conversion des arguments

- Les arguments sont toujours stockés comme des **chaînes de caractères**.
- Peu pratique si on veut manipuler des valeurs numériques.
- Fonctions pour faire des conversions:

```
int atoi(const char *nptr);  // de la chaîne en entier
double atof(const char *nptr); // de la chaîne en nombre à virgule
     flottante
int snprintf(char *str, size_t size,
             const char *format, ...);
// str: buffer, size: taille en octets max à copier,
// format: cf printf(), ret: nombre de char lus
```

La ligne de commande (4/4)

Exemple d'utilisation

```c
#include <stdio.h>
#include <stdlib.h>
#include <libgen.h>
int main(int argc, char **argv) {
    if (argc != 3) {
        char *progname = basename(argv[0]);
        fprintf(stderr, "usage: %s name age\n", progname);
        return EXIT_FAILURE;
    }
    // argv[0] est le nom du programme on l'ignore
    // le 1er argument est une chaîne de caractères (pas de conversion)
    char *name = argv[1];
    int age = atoi(argv[2]); // le 2e argument est un entier (conversion)
    printf("Hello %s, you are %d years old.\n", name, age);
    return EXIT_SUCCESS;
}
```

La ligne de commande (4/4)

Exemple d'utilisation

```c
#include <stdio.h>
#include <stdlib.h>
#include <libgen.h>
int main(int argc, char **argv) {
    if (argc != 3) {
        char *progname = basename(argv[0]);
        fprintf(stderr, "usage: %s name age\n", progname);
        return EXIT_FAILURE;
    }
    // argv[0] est le nom du programme on l'ignore
    // le 1er argument est une chaîne de caractères (pas de conversion)
    char *name = argv[1];
    int age = atoi(argv[2]); // le 2e argument est un entier (conversion)
    printf("Hello %s, you are %d years old.\n", name, age);
    return EXIT_SUCCESS;
}
```

```
$ ./prog Paul 29
Hello Paul, you are 29 years old.
```