Programación de Servicios y Procesos

U.D.2 – Programación Multihilo (III)

Hilos(o hebras) POSIX

- Los hilos POSIX se identifican por la estructura (struct) pthread_t
- Dada la dificultad abstracta identificar un hilo se usa la siguiente función para identificarlos.
- La función devuelve valor “nonzero” cuando son iguales y “zero” cuando no lo son.

```
#include <pthread.h>

int pthread_equal(pthread_t tid1, pthread_t tid2);
```

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/

Hilos(o hebras) POSIX

- La función pthread_self(void) se usa cuando queremos identificar un hilo.

```
#include <pthread.h>

pthread_t pthread_self(void);
```

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/

Hilos(o hebras) POSIX

```
#include <pthread.h>

int pthread_create(pthread_t *restrict tidp, const pthread_attr_t *restri
```

- **El primer argumento es una dirección de tipo pthread_t. Una vez que la función es llamada con éxito, la variable cuya dirección se pasa como primer argumento contendrá el ID del hilo recién creado.**
- **El segundo argumento puede contener ciertos atributos que queremos que contenga el nuevo hilo. Podría ser la prioridad, etc.**
- **El tercer argumento es un puntero de función. Esto es algo a tener en cuenta ya que cada hilo comienza con una función y la dirección de esa función se pasa aquí como tercer argumento para que el kernel sepa desde qué función iniciar el hilo.**
- **Como la función (cuya dirección se pasa en el tercer argumento anterior) puede aceptar también algunos argumentos, podemos pasar estos argumentos en forma de puntero a un tipo void. Ahora, ¿por qué se eligió un tipo void? Porque si una función acepta más de un argumento, entonces este puntero puede ser un puntero a una estructura que puede contener estos argumentos.**

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/

Hilos(o hebras) POSIX

- **A continuación vamos a ver un ejemplo de uso de hilos POSIX. El programa hilos_posix hace lo siguiente:**
  - **Primero: usa la función ptread_create() para generar dos hebras. La función de inicio que pasamos a ambos hilos, es la misma.**
  - **En la función doSomeThing() se usa las funciones ptread_self() y ptread_equal() para identificar cuál de los hilos generados es.**
  - **Además, en la función doSomeThing() se ejecuta un bucle for para realizar una simulación de carga de trabajo para el hilo correspondiente.**

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/

Hilos(o hebras) POSIX

- **Veamos qué pasa cuándo quitamos la función sleep() del código.**
  - **EFECTIVAMENTE, TENEMOS UN PROBLEMA DE CONCURRENCIA.**
  - **EL PROCESO QUE GENERÓ LOS HILOS MURIÓ ANTES DE ACABAR EL ÚLTIMO HILO.**

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/

Hilos(o hebras) POSIX

```
#include <pthread.h>

int pthread_join(pthread_t thread, void **rval_ptr);
```

- La función anterior se asegura de que su hilo padre no termine hasta que termine. Esta función se llama desde dentro del hilo padre y el primer argumento es el ID del hilo en el que esperar y el segundo argumento es el valor de retorno del hilo en el que queremos que espere el hilo padre. Si no estamos interesados en el valor de retorno podemos poner este puntero a NULL.

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/

Hilos(o hebras) POSIX → Finalizarlos

Desde un de vista más abstracto (más arriba) hay tres maneras de que un hilo acabe:

- **El hilo vuelve desde su rutina de inicio.**
- **El hilo es cancelado por otro hilo. Esto se puede hacer con la función pthread_cancel().**
- **El proceso decide llamar a la función pthread_exit() por sí mismo.**

```
#include <pthread.h>

void pthread_exit(void *rval_ptr);
```

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/

Hilos(o hebras) POSIX

- **A continuación vamos a ver un segundo ejemplo de uso de hilos POSIX. El programa hilos_posix_II hace lo siguiente:**
  - **Primero: Creamos dos hilos usando pthread_create().**
  - **La función de inicio para ambos hilos es la misma, es decir, doSomeThing()**
  - **Los hilos salen de la función de inicio usando la función pthread_exit() con un valor de retorno.**
  - **En la función principal, una vez creados los hilos, se llama a las funciones pthread_join() para esperar a que se completen los dos hilos.**
  - **Una vez que ambos hilos se han completado, se accede a su valor de retorno mediante el segundo argumento de la llamada pthread_join().**

https://www.thegeekstuff.com/2012/04/create-threads-in-linux/