Memoria Practica (III)

Programación de Servicios y Procesos | Practica 01

Que

Escribe una clase llamada Ejecuta que reciba como argumentos el comando y las opciones del comando que se quiere ejecutar. El programa debe crear un proceso hijo que ejecute el comando con las opciones correspondientes mostrando un mensaje de error en el caso de que no se realizase correctamente la ejecución. El padre debe esperar a que el hijo termine e informar si se produjo alguna anomalía en la ejecución del hijo.

Para que

Para ayudarnos a hacer la práctica 3, ya que esta práctica es un poco como el tutorial básico para la tercera práctica.

Pseudocodigo

```
Clase Ejecuta
  Función ejecutar(comando)
    retorno = ejecutar el comando usando la función system()
    Si retorno es igual a -1
      Mostrar "Error: Ha habido un error en la ejecución"
    Fin Si
    Devolver retorno
  Fin Función
Fin Clase

Función principal main
  Si el número de argumentos de línea de comandos no es igual a 2
    Mostrar "Uso: <nombre_programa> "<comando>""
    Salir con código de error 1
  Fin Si

  Crear una instancia de la clase Ejecuta llamada ej
  Crear una variable pid para el identificador de proceso
  Crear una variable stcomand para el estado de ejecución del comando
  pid = bifurcar()  # Crear un proceso hijo

  Según el valor de pid
    Caso -1:
      Mostrar "Error: Ha habido un error en la creación del proceso"
    Caso 0:
      stcomand = ej.ejecutar(Argumento[1])  # Ejecutar el comando proporcionado como
argumento
    Por defecto:
      Esperar a que el proceso hijo termine y almacenar el estado en stcomand

  Si stcomand no es igual a -1
    Mostrar "Ok: Comando ejecutado correctamente"
  De lo contrario
    Mostrar "War: El comando no se ha ejecutado correctamente"
  Retornar 0
Fin Función
```

Como

Para empezar, tengo una clase llamada "Ejecuta" con una única función pública llamada "ejecutar". Esta función tiene como parámetro de entrada un array de caracteres. La función "ejecutar" llama a otra función llamada "system" a la que se le pasa el array de caracteres como parámetro. Si hay un error en la función "system", esta devolverá un valor de -1, que se guarda en una variable "i". Se comprueba con una instrucción "if" si "i" es igual a -1 para mostrar un mensaje en pantalla.

En la función principal, se guarda el argumento ingresado entre comillas dobles desde la terminal. Luego, se verifica si la longitud de "argc" (la longitud del array) es diferente de dos. Si es así, significa que el formato no es correcto, se notifica al usuario y se cierra el programa. De lo contrario, se crea un objeto de tipo "Ejecuta", se crea un proceso con la función "fork()", y se declara una variable que almacenará el estado del comando "stcomand".

Se crea un switch según el valor de "pid", que es la variable donde se almacena el valor de retorno de "fork". En caso de que sea -1, se trata de un error. Si es 0, significa que estamos en el proceso hijo, y se llama a la función "ejecutar" del objeto "Ejecuta" creado anteriormente, pasándole como parámetro el valor introducido entre comillas dobles. En otro caso, estamos en el proceso padre, que espera a que el hijo termine. Se verifica el valor de retorno de la función "ejecutar", que se ha guardado en la variable "stcomand" en el proceso hijo. Si es diferente de -1, significa que todo ha ido bien y se imprime un mensaje de "OK". En caso contrario, se imprime un mensaje de error.

Conclusión

Esta práctica me ha resultado un poco más sencilla que las demás, además, también me ha ayudado a realizar las otras dos prácticas.