

Programa comandos personalizados para el sistema operativo

Código del proyecto: ICX0_P4

Producto 3. Automatización básica de la administración de un sistema operativo

Alumno:	Jesús Real Tovar
Consultor/ra:	Óscar Baltà Fabró
Fecha de entrega:	31 de mayo de 2019

1. Función main, entry point de nuestra aplicación.

```
1  /*
2  Función main, entry point de la aplicación de consola.
3  */
4  int main()
5  {
6      printf("%s\n", "
↳ =====
↳ ");
7      printf("%s\n", "Introduce la ruta del archivo hosts.");
8      const char * path[255];
9      // Obtenemos mediante consola, la ruta del archivo
10     read_string(path, 255);
11     // Mostramos el contenido del archivo por consola
12     show_content_of_file(path);
13
14     printf("%s\n", "
↳ =====
↳ ");
15     printf("%s\n", "Introduce la ruta del archivo pares.");
16     const char * diff_path[255];
17     // Obtenemos mediante consola, la ruta del archivo
18     read_string(diff_path, 255);
19     // Hacemos la comparación de los dos archivos y copiamos la diferencia
20     diff_two_files(path, diff_path);
21     // Mostramos el contenido del archivo por consola
22     show_content_of_file(path);
23
24     system("pause");
25     return 0;
26 }
```

Desde la función main, como podemos ver, llamamos a las diferentes funciones para realizar las distintas tareas que requerimos realizar sobre los distintos ficheros.

- Pedimos la ruta del archivo hosts
- Mostramos el contenido del archivo hosts
- Pedimos la ruta del archivo con los pares
- Hacemos la comparación de los dos archivos y aplicamos las diferencias al archivo hosts
- Mostramos el contenido del archivo hosts

2. Función para mostrar el contenido de un archivo.

```
1  /*
2  Función para mostrar el contenido de un archivo
3  */
4  void show_content_of_file(const char * path) {
5      // Abrimos el archivo
6      FILE * ptr_file = open_file(path);
7      // Si el archivo no existe retornamos
8      if (ptr_file == NULL)
9      {
10         printf("%s\n", "El archivo no existe.");
11         return;
12     }
13     printf("%s %s %s\n", "=====" "Contenido del archivo: ", path, "
    ↪ =====");
14     char c;
15     c = fgetc(ptr_file);
16     // Mientras que no llegemos al final del archivo
17     while (c != EOF)
18     {
19         // Escribimos en consola cada carácter del archivo
20         printf("%c", c);
21         c = fgetc(ptr_file);
22     }
23     printf("\n");
24     // Cerramos el archivo
25     fclose(ptr_file);
26 }
```

Esta función está diseñada para mostrar todo el contenido de un archivo por consola, carácter por carácter. Nos sirve para mostrar los cambios realizados en el archivo hosts y verificar que estos están correctos.

3. Función para chequear el contenido de un archivo.

```
1  /*
2  Función para chequear el contenido de un archivo
3  */
4  bool check_content_of_file(const char * path, const char * find) {
5
6      // Abrimos el archivo
7      FILE * ptr_file = open_file(path);
8      // Si el archivo no existe retornamos
9      if (ptr_file == NULL)
10     {
11         printf(" %s\n", "El archivo no existe.");
12         return false;
13     }
14
15     bool found = false;
16     char * line[1024];
17
18     // Por cada linea del archivo
19     while (fgets(line, 1024, ptr_file) != NULL) {
20         // Comprobamos si contiene la palabra o frase a buscar
21         if ((strstr(line, find)) != NULL) {
22             // Si la contiene, seteamos como encontrada y rompemos el bucle
23             found = true;
24             break;
25         }
26     }
27
28     // Cerramos el archivo
29     fclose(ptr_file);
30
31     // Retornamos found
32     return found;
33 }
```

Con esta función podemos verificar que un archivo contiene cierta palabra o conjunto de palabras. Nos es útil para saber si el archivo hosts contiene alguno de los pares que debemos incorporar desde el archivo de pares.

Función para comparar dos archivos, agregar las diferencias a un archivo temporal y aplicarlos al archivo principal.

4. Función para comparar dos archivos, agregar las diferencias a un archivo temporal y aplicarlos al archivo principal.

```
1  /*
2  Función para comparar dos archivos, agregar las diferencias a un archivo temporal y aplicarlos al archivo
   ↪ principal
3  */
4  void diff_two_files(const char * path, const char * diff_path) {
5      // Abrimos el archivo diff
6      FILE * ptr_file_diff = open_file(diff_path);
7      // Si el archivo no existe retornamos
8      if (ptr_file_diff == NULL)
9      {
10         printf("%s\n", "El archivo no existe.");
11         return;
12     }
13
14     // Creamos el archivo temporal
15     FILE * ptr_file_tmp = tmpfile();
16     // Si el archivo no existe retornamos
17     if (ptr_file_tmp == NULL)
18     {
19         printf("%s\n", "Error al crear el archivo temporal.");
20         return;
21     }
22
23     char * line[1024];
24
25     // Por cada linea del archivo diff
26     while (fgets(line, 1024, ptr_file_diff) != NULL) {
27         // Comprobamos si contiene el contenido de la linea
28         if (!check_content_of_file(path, line)) {
29             // La añadimos al archivo temporal
30             fputs(line, ptr_file_tmp);
31         }
32     }
33     // Posicionamos el puntero al principio
34     fseek(ptr_file_tmp, 0, SEEK_SET);
35
36     // Abrimos el archivo
37     FILE * ptr_file = open_file(path);
38     // Si el archivo no existe retornamos
39     if (ptr_file == NULL)
40     {
41         printf("%s\n", "El archivo no existe.");
42         return;
43     }
44     // Posicionamos el puntero al final
45     fseek(ptr_file, 0, SEEK_END);
46     fputs("\n", ptr_file);
47     // Por cada linea del archivo
48     while (fgets(line, 1024, ptr_file_tmp) != NULL) {
49         // La añadimos al archivo
50         fputs(line, ptr_file);
51     }
52
53     // Cerramos los archivos
54     fclose(ptr_file);
55     fclose(ptr_file_diff);
56     fclose(ptr_file_tmp); //El archivo temporal se borra solo al cerrarlo
57 }
```

Con esta función realizamos la tarea más importante de la aplicación, incorporar al archivo hosts los pares del archivo de pares. Para esto seguimos una serie de acciones:

- Abrimos el archivo de pares
- Creamos el archivo temporal donde guardaremos los pares que no contenga el archivo hosts
- Comprobamos que el archivo hosts contenga los pares del archivo de pares, los que no contiene los añadimos al archivo temporal
- Abrimos el archivo hosts
- Añadimos al archivo hosts cada par guardado en el archivo temporal
- Cerramos los archivos
- El archivo temporal es automáticamente borrado por el propio sistema ya que usamos la función *tmpfile*

5. Salida

```
=====
Introduce la ruta del archivo hosts.
d:\hosts
===== Contenido del archivo: d:\hosts =====
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97      rhino.acme.com          # source server
#      38.25.63.10      x.acme.com              # x client host
#
# localhost name resolution is handled within DNS itself.
#      127.0.0.1        localhost
#      ::1              localhost
127.0.0.1 localhost
::1 localhost
=====
Introduce la ruta del archivo pares.
d:\pares.txt
===== Contenido del archivo: d:\hosts =====
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97      rhino.acme.com          # source server
#      38.25.63.10      x.acme.com              # x client host
#
# localhost name resolution is handled within DNS itself.
#      127.0.0.1        localhost
#      ::1              localhost
127.0.0.1 localhost
::1 localhost
127.0.0.1 www.lavanguardia.com
127.0.0.1 www.elpais.es
127.0.0.1 www.ccoo.es
127.0.0.1 www.google.cat
Presione una tecla para continuar . . .
```