Visual SlickEdit – Modern Editor Theme for OS/2

Version

0.1

Table of Contents

MODERN PROGRAMMER'S EDITOR THEME

Visual SlickEdit – OS/2 Status

Editor status on the OS/2 platform

Visual SlickEdit is a premiere Programmer's Editor. The product has a long history, it was available on the OS/2 platform between 1993 – 1998. In February of 1999 MicroEdge Inc. released the final patch for the product on the OS/2 platform, bringing it to version 4.0b. No further upgrades and/or patches have been released since.

In the meantime, the product itself has continued to evolve and is currently available across multiple platforms. The most recent product release level is SlickEdit 2020. Trial versions can be downloaded from here.

Goal of the Visual SlickEdit Modern Editor Theme for OS/2

Utilized in many source code editors, or general applications for that matter, a concept of a User Presentation Space Theme is present in Visual SlickEdit as well. The 'Color Scheme' functionality in SlickEdit allows you, the end user, to adjust various colour mappings. Additional parts of the editor's configuration allow you to set fonts, certain key mappings, and so forth. However, due to SlickEdit's age, these remain individual settings which are rather unwieldly to deal with. If the default settings are not acceptable, the changes require detailed manual configuration steps for each programming language one intends to utilize.

Therefore, the concept of a Theme is of particular interest when it comes to an implementation within a programmer's editor. After all, such an editor will most likely be utilized to support source code writing in multiple languages. Further on, the various programming languages utilize their own (and specific) syntax, which at times may make it difficult to interpret the logic behind the source code when one simply looks at a stream of flat text. Ultimately, Theme implementation may provide not just a visually more appealing interface, but it can in fact make the programmer more productive by simplifying and speeding up the manipulation of the source code.

In the case of the latest publicly available release of Visual SlickEdit for OS/2, there are several built-in themes present in the editor itself. These, while being incredibly configurable due to SlickEdit's flexiblity, are primarily limitted to font and colour substitutions.

Further on, Visual SlickEdit's ability to support (syntax recognition, etc.) the OS/2 APIs is limitted to a pre-defined subset of such APIs. This once again is entirely driven by the product's age and the published API information at the time the last version of SlickEdit was available for the OS/2 platform. What this means is that as the platform continued to change and new libraries, and therefore APIs and programming elements, became available on OS/2, they are not readily recognized within SlickEdit's environment.

Therefore, the goal of 'Visual SlickEdit - Modern Editor Theme for OS/2' is to close some of these existing gaps by introducing:

1) modern syntax highlighting for the C/C++ language

2) recognition of the wider OS/2 Development Toolkit APIs and elements

3) packaging of the individual SlickEdit's configuration options into a pre-made set of application configuration files

Visual SlickEdit's Configuration Options

User Customizable Controls

This section provides a high-level summary of the SlickEdit's built-in configuration options. This is done primarily so that as specific changes are highlighted later on, the reader has a good understanding of what changes are being made. No specific application navigation paths are provided given the assumption that you are familiar enough with the editor to otherwise implement the required configuration manually.

A great starting point is the 'Extension Options' configuration screen in SlickEdit, see Fig 2.1 below:

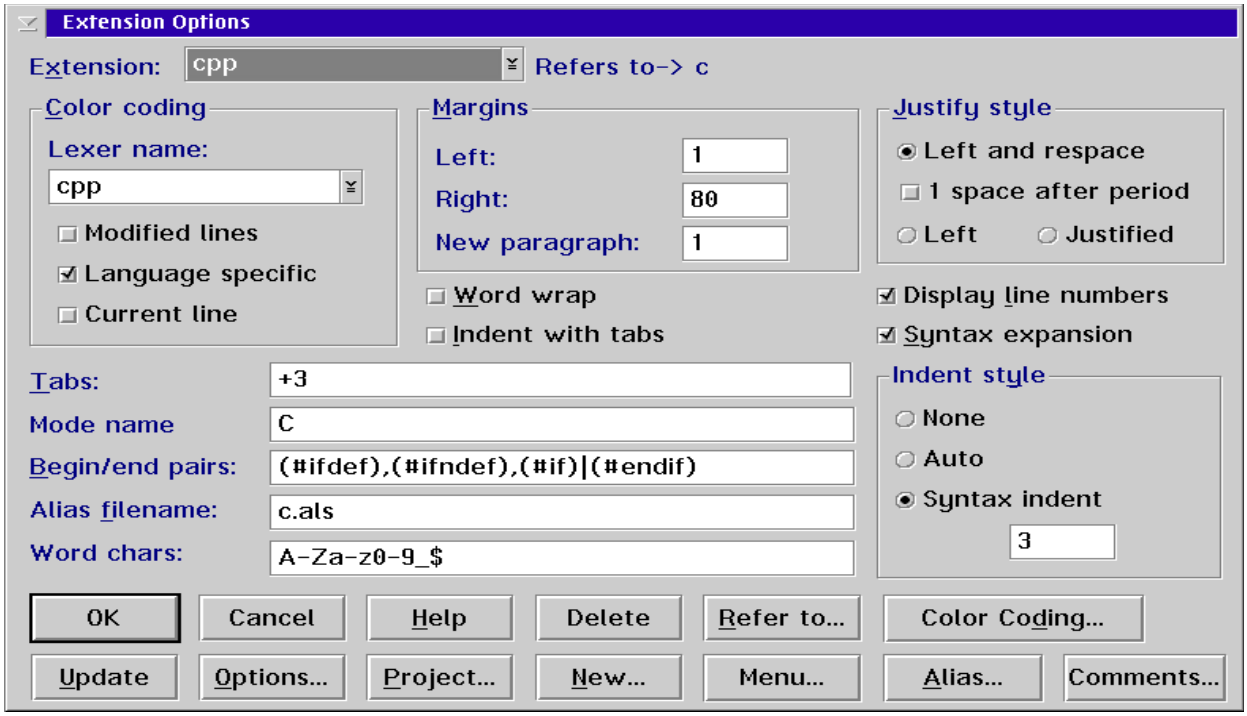

FIG 2.1 – Extension Options screen

All of the options listed above are of course configurable, however for our purposes we will be looking at implementing the changes applicable to the editor functionality controlled through 'Color Coding...' button, see Fig 2.2 below:

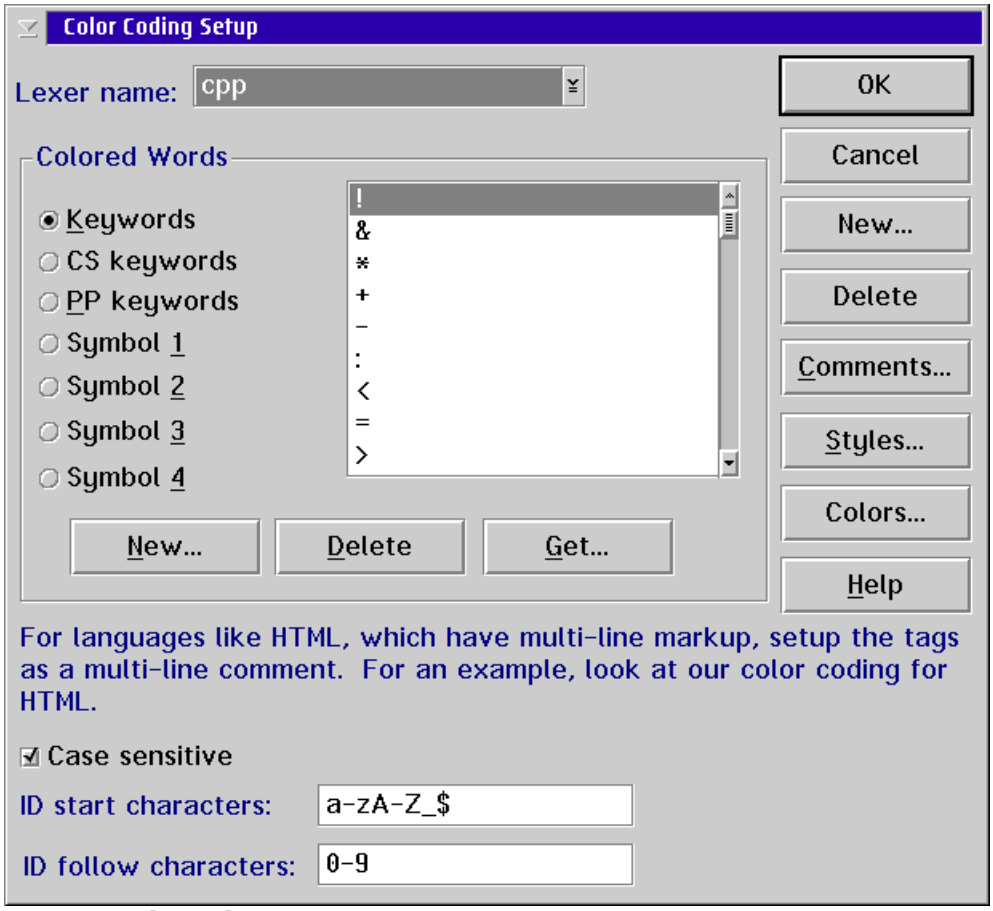

FIG 2.2 – Color Coding Setup screen

Specifically since we are only adjusting the C/C++ configuration one should look up the appropriate Lexer name, most likely both C and C++ are mapped to the same Lexer name, that being 'cpp', as shown in the above image.

Here we have the option to change the 'Colored Word' definitions, as well as the specific 'Colors' that are to be used. Once implemented, the 'Colored Word' definitions will map specific language keywords, structures and environment variables to a particular 'Colors...' definitions. This is the power of SlickEdit's user extendible configuration which enables us to play a bit of a catch-up game on the OS/2 platform.

The below section will review the changes implemented by the 'Visual SlickEdit - Modern Editor Theme for OS/2', how each of these editor elements is adjusted, and present a visual comparison of a sample source code (DISKIO utility) showcasing the source code rendering using both the default editor settings and those provided by the 'Visual SlickEdit - Modern Editor Theme for OS/2'.

Visual SlickEdit Modern Theme Changes

Implementation Review

The changes packaged into the 'Visual SlickEdit - Modern Editor Theme for OS/2' utilize the above mentioned editor customization controls. However, as opposed to spelling out the details that one must implement manually, the configuration is captured using a set of standard Visual SlickEdit configuration files. These files are the USER customization files, which are normally created by the editor whenever you manually change any of the default configurations. Therefore, it is important that you first check whether your installation already has these present. If they are, you may want to merge your existing changes along with the Modern Theme changes. If you do not, the Modern Theme changes will overwrite your previous customizations. Either way, backup would be prudent.

Colour Scheme Configuration File

The *VSSCHEME.INI* file is the default SlickEdit 'Color Scheme' file. The user defined changes are stored in the *USSCHEME.INI* file.

Colour Coding Scheme Configuration File

The *VSLICK.VLX* file is the default SlickEdit 'Color Coding Scheme' file. The user defined changes are stored in the *USER.VLX* file.

Visual SlickEdit Modern Theme for OS/2 “Look'n'Feel”

Following are a few examples of viewing a couple of different C/C++ source files with the default SlickEdit configuration, as well as the modified 'Visual SlickEdit - Modern Editor Theme for OS/2' configuration.

MODERN PROGRAMMER'S EDITOR THEME

Sample main() Declaration

```
int main(int argc, char **argv)
{
   char szVersion[32];
   int nDisks, nCDROMs, nCount, xHD[256], xCD[256], cHD = 0, cCD = 0;
   int rc;

   strcpy(szVersion, rcsrev + sizeof("$Revision: ") - 1);
   *strchr(szVersion, ' ') = 0;

   printf("DISKIO - Fixed Disk Benchmark, Version %s"
          "\n(C) 1994-1998 Kai Uwe Rommel"
          "\n(C) 2004 madded2\n"
          "\n    2020 Dariusz Piatkowski\n", szVersion);

#ifdef OS2
   init_timer ();
   CDInit ();
#endif

#ifdef OS2CPUPERF
   rc = CpuPerfInit ();
   if (rc == NO_ERROR)
   {
      flPerf0k = 1;
   }
   else
   {
      flPerf0k = 0;
      printf("\nOS/2 CPU load measuring mechanism disabled, rc=%d\n", rc);
   }
#endif
```

FIG 3.1 – DEFAULT View

```
int main(int argc, char **argv)
{
   char szVersion[32];
   int nDisks, nCDROMs, nCount, xHD[256], xCD[256], cHD = 0, cCD = 0;
   int rc;

   strcpy(szVersion, rcsrev + sizeof("$Revision: ") - 1);
   *strchr(szVersion, ' ') = 0;

   printf("DISKIO - Fixed Disk Benchmark, Version %s"
          "\n(C) 1994-1998 Kai Uwe Rommel"
          "\n(C) 2004 madded2\n"
          "\n    2020 Dariusz Piatkowski\n", szVersion);

#ifdef OS2
   init_timer ();
   CDInit ();
#endif

#ifdef OS2CPUPERF
   rc = CpuPerfInit ();
   if (rc == NO_ERROR)
   {
      flPerf0k = 1;
   }
   else
   {
      flPerf0k = 0;
      printf("\nOS/2 CPU load measuring mechanism disabled, rc=%d\n", rc);
   }
#endif
```

FIG 3.2 – Modern Theme View

Sample #INCLUDE and #DEFINE Declaration

```
/*
 * $Log: dhry_tmr.c,v $
 * Revision 1.2  1999/10/28 17:35:29  rommel
 * fixed timer code
 *
 * Revision 1.1  1999/10/25 08:36:19  rommel
 * Initial revision
 *
 */

#include <stdio.h>

extern unsigned long Number_Of_Runs;

#define THREADSTACK 65536

#ifdef OS2

   #define INCL_DOS
   #define INCL_NOPM
   #include <os2.h>

typedef QWORD TIMER;

VOID APIENTRY timer_thread(ULONG nArg)
{
   HEV hSem;
   HTIMER hTimer;

   DosCreateEventSem(0, &hSem, DC_SEM_SHARED, 0);
   DosAsyncTimer(nArg, (HSEM) hSem, &hTimer);
   DosWaitEventSem(hSem, SEM_INDEFINITE_WAIT);
   DosStopTimer(hTimer);
   DosCloseEventSem(hSem);

   Number_Of_Runs = 0;

   DosExit(EXIT_THREAD, 0);
}
```

FIG 3.3 – DEFAULT View

```
/*
 * $Log: dhry_tmr.c,v $
 * Revision 1.2  1999/10/28 17:35:29  rommel
 * fixed timer code
 *
 * Revision 1.1  1999/10/25 08:36:19  rommel
 * Initial revision
 *
 */

#include <stdio.h>

extern unsigned long Number_Of_Runs;

#define THREADSTACK 65536

#ifdef OS2

   #define INCL_DOS
   #define INCL_NOPM
   #include <os2.h>

typedef QWORD TIMER;

VOID APIENTRY timer_thread(ULONG nArg)
{
   HEV hSem;
   HTIMER hTimer;

   DosCreateEventSem(0, &hSem, DC_SEM_SHARED, 0);
   DosAsyncTimer(nArg, (HSEM) hSem, &hTimer);
   DosWaitEventSem(hSem, SEM_INDEFINITE_WAIT);
   DosStopTimer(hTimer);
   DosCloseEventSem(hSem);

   Number_Of_Runs = 0;

   DosExit(EXIT_THREAD, 0);
}
```

FIG 3.4 – Modern Theme View

OS/2 Development Toolkit Integration

The updated colour scheme is not the only enhancement. The 'Visual SlickEdit - Modern Editor Theme for OS/2' also brings close integration with the officially available OS/2 Development Toolkit. Specifically what this means is that the various macros, TYPEDEFs, DEFINEs and STRUCTs that are created within the Toolkit header files are accounted for and allow Visual SlickEdit to readily recognize such elements.

In combination with the colour scheme settings, this allows the programmer to easily pint-point code elements that are specific to the source being worked on, and which do NOT belong to the Toolkit itself. Such an approach enables quicker differentiation of project source code elements from the existing Toolkit elements, hopefully simplifying the task of both reading the existing code, as well as developing new code.

Note

What the above paragraph means is that all of the source files contained in the OS/2 Development Toolkit have been scanned and the applicable element definitions captured in the USER.VLX configuration file. The placement of each element definition is further detailed in the following 'Colored Words Configuratoin' section.

Colored Words Configuration

There are several choices listed on this screen, see Fig 3.1 below. We will review each one and explain how these are used.

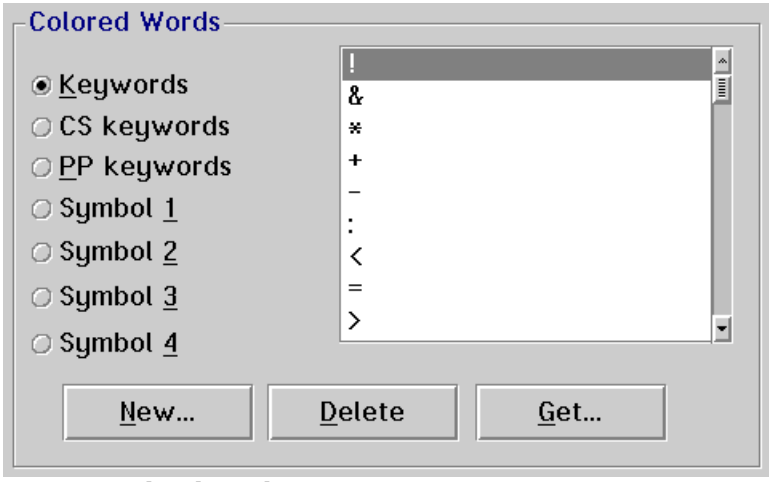

FIG 3.1 – Colored Words screen

Keywords

These are language specific keywords which are going to use a particular colour.

CS keywords

These are language specific keywords that are Case Sensitive, they always remain case sensitive and will use a particular colour.

PP keywords

These are language specific keywords that are XXX and will use a particular colour.

Symobl 1

These are language specific keywords that are mapped to the OS/2 Development Toolkit Macro definitions, and will a particular colour.

Symbol 2

These are language specific keywords that are intended to display using an easily identifiable colour.

Symbol 3

These are language specific keywords that are mapped to the OS/2 Development Toolkit **DEFINE** definitions, and will use a particular colour.

Symobl 4

These are language specific keywords that are mapped to the OS/2 Development Toolkit Macro **TYPEDEF** or **STRUCT** definitions, and will use a particular colour.

Color Settings

The colour settings implemented by the Visual SlickEdit Modern Theme for OS/2 are stored in a SlickEdit 'Color scheme' named “VSE – OS/2 – Modern Dark”.

Beyond the changes introduced by the 'Visual SlickEdit Modern Theme for OS/2', this scheme can be further modified as required.

Installation

The installation process is manual. This is largely because of the pre-requisite review of your existing Visual SlickEdit customizations, and the need to check whether the configuraiton files provided as part of the Modern Theme already exist in your installation.

File Locations

The files already mentioned in this document (see 'Implementation Review' section above) simply need to be copied from the ZIP file to the location where your Visual SlickEdit is installed. The SlickEdit program should of course be shut down and not executing during the deployment of the configuration files.

Index