Adding an Lua-based integrated character-based menu system into the SciTE editor

Robin Snyder, robin@robinsnyder.com

Slide notes from the Lua Workshop in Reston, VA, November 29-30, 2012

Printed: 2012/12/01

1. Abstract

The multi-platform open source SciTE editor provides support for interactive editor-based scripting using Lua and provides a standard but limited pull-down menu system into which some selected actions can be added. In order to mimic actions of the original Borland Sprint editor menu system, a general purpose multi-platform and extensible character-based menu system was added to SciTE using Lua scripting of the call tip feature of SciTE. Problems encountered and resolved during the process included QUERTY/Dvorak keyboard layout issues, context-sensitive help auto-location in the Lua source, and integrating support for the Lua-supported Logitech G-13 gaming keypad. Related issues involved custom lexing support via the Lua-based lexer.

2. Borland Sprint: 1988

Borland's DOS-based Sprint word processor (1988):

- Formatter (Scribe-like formatter macros)
- Editor (EMACS-like macros via forth-like macro language)
- Printer customizations
- Screen customizations

3. Formatting

- TeX (Donald Knuth)
- C++ macro preprocessor

Formatter macros are not the same as editor macros.

What is WYSIWYG?

4. WYSIWYG

Microsoft Word is WYSIWYG:

- What you see is what you get.

Microsoft Word is WYSIWYG:

- What you see is what you get.
- What you see is what you hope to get.
- What you see is all you get.

- Editor macros can change what you see.
- Formatter macros can change what you get.

5. Edit-Compile-Run

Formatting introduces another step.

- Edit Compile Run
- Edit/Design Compile/Link Run
- Edit Format Compile Run

Formatting helps turn non-computer-checked redundancy into computer-managed redundancy. Example: Menu system key mappings.

Formatting makes it easy to add features that are part of AOP (Aspect Oriented Programming).

6. Sprint macro system

Screen: color-coded characters (DOS-based text screen).

Formatter: single-letter character macros mapped to screen color-coding.

7. Windows XP to 7

- Need: 2+GB to 16GB memory space.
- DOS-based Sprint editor not workable.
- DOS-based Sprint formatter still used (but being replaced).

8. Sprint replacement

- Open Source SciTE based on Scintilla components.
- Lua script support

9. SciTE

- Has menu system.
- Has hooks for adding commands that call Lua.
- Has output area

Has many nice features, but one size does not fit all.

10. Complete customization

Complete customization, like Sprint, requires the ability to handle each and every keystroke.

The Lua adaptation in SciTE allows and supports this.

11. Key sources

- Keyboard
- Numeric keypad
- Special keys
- Logitech G-13 Keypad
- Logitech MK 350/550 Wave Keyboard

12. Logitech G13

13. Logitech Wave Keyboard

14. Systems approach

- Map special keysto Shift-Ctrl-Alt keys (60+ keys)
- If more needed, use Shift-Ctrl, Ctrl-Alt, or Shift-Alt.
- Map G13 (using Lua) to AutoHotKey.
- Use AutoHotKey to map Shift-Ctrl-Alt keys, etc. to applications.
- In SciTE, use Lua to map keys to actions.

15. Menus

- Traditional SciTE menu bar (leave in place).
- Completion select use - co-opt for popup menu system.

16. Special keys

- Press **`F10`**, **`k`** to toggle showing each key pressed (in the output window).
- Press **`Alt-~`**. The next key pressed takes one to the Lua code for that key.
- In a menu, press **`F1`** to go to the Lua code for that menu.

17. Considerations

- Linux support added after Windows support (bottom up manner)
- Dvorak keyboard layout (mappings for raw keys, when needed)

18. Autocompletion

The auto-completion facility of SciTE is used to implement a menu system - in addition to the traditional menu system.

These menus appear to be like a context-sensitive menu.

19. Keystrokes

```
function OnKey(key, shift, ctrl, alt)
   if not is_win then
      if (key > 96) and (key < 123) then
         key = key - 32
         shift = false
      elseif (key > 65280) and (key < 65308) then
         key = key - 65280
      elseif (key > 65469) and (key < 65482) then
         key = key - 65358
      elseif shift then
         key = gtkShiftMap1[key]
      else
         key = gtkToWinKeyMap1[key]
         end
      putKey("OnKey", key, shift, ctrl, alt)
      end
   local ok1, result1 = pcall(OnKey1,key, shift, ctrl, alt)
   if not ok1 then
      putKey("OnKey", key, shift, ctrl, alt)
      print(result1)
      sciteErrorGo()
      end
   return false
   end
```

20. Keystrokes

```
function OnKey1(key, shift, ctrl, alt)
   local k = keys1[key]
   if state1.showkeys == 1 then
      putKey("OnKey1.1", key, shift, ctrl, alt)
      end
   if not k then
      if state1.showkeys == 0 then
         putKey("OnKey1.2",key, shift, ctrl, alt)
         end
      sciteError("Key " .. key .. " has no mapping [0x" .. tonumber(key, 16) .. "]",
"D:\F\LUX\scite4.lux", 1069,1)
   else
      local key3 = keys2[k]
      local is_help1 = 0
      if (key ~= 16) and (key ~= 17) and (key ~= 18) then
         if state1.keyHelp then
            is_help1 = 1
         elseif alt and ((key > 64) and (key < 91) or (key > 47) and (key < 58)) and
editor:AutoCActive() then
            is_help1 = 2
            end
         end
      if is_help1 == 1 then
         print("OnKey :: keyHelp")
         if state1.gkeyHelp then
            gkeyHelpGo(key)
            state1.keyHelp = false
            state1.gkeyHelp = false
         else
            if (key == 54) and shift and ctrl then
               state1.gkeyHelp = true
            else
               keyHelpGo(key, shift, ctrl, alt)
               state1.keyHelp = false
               end
            end
      elseif is_help1 == 2 then
         local ch1 = string.char(key)
         local i
```

```
         local j = 0
         for i=1,menuLen1 do
            if menuKeys1[i] == key then
               j = i
               break
               end
            end
         if j == 0 then
            print("Menu key [" .. ch1 .. "] not in menu.")
         else
            if menuKey1 and (menuKey1 ~= 0) then
               menuItemFind1(menuKey1, menuCode1[j])
               menuClose()
               end
            end
      else
         if editor:AutoCActive() and (key3[2] ~= 1) then
            local b = ((key >= 48) and (key <= 57)) or ((key >= 65) and (key <= 90)) or
(key == 13) or (key == 112)
            if shift or ctrl or alt or (not b) then
               putKey("OnKey1.4", key, shift, ctrl, alt)
               print("key3[2]=[" .. key3[2] .. "]")
               menuClose()
               end
            end
         if (key3[2] == 0) and editor:AutoCActive() then
            menuKey(key, shift, ctrl, alt)
         else
            if state1.is_status ~= 0 then
               if editor:AutoCActive() then
                  editor:AutoCCancel()
                  end
               state1.is_status = 0
               end
            if shift and ctrl and alt and key3[8] then
               key3[10]() -- 111
            elseif ctrl and alt and key3[8] then
               key3[9]() -- 110
            elseif shift and alt and key3[6] then
               key3[8]() -- 101
            elseif shift and ctrl and key3[6] then
               key3[6]() -- 011
            elseif alt and key3[7] then
               key3[7]() -- 100
            elseif ctrl and key3[5] then
               key3[5]() -- 010
            elseif shift and key3[4] then
               key3[4]() -- 001
            elseif key3[3] then
               key3[3]() -- 000
            else
               putKey("OnKey1.6", key, shift, ctrl, alt)
               sciteError("key " .. key .. " not mapped", "D:\F\LUX\..\LUX\scite4.lux",
1169,1)
               end
            end
         end
      end
   return true
   end
```

21. Keys

```
keys2 = {
-- ...
{65,0,key_065_000,key_065_001,key_065_010,key_065_011,key_065_100,key_065_101,key_065_110
,key_065_111},
{66,0,key_066_000,key_066_001,key_066_010,key_066_011,key_066_100,key_066_101,key_066_110
,key_066_111},
{67,0,key_067_000,key_067_001,key_067_010,key_067_011,key_067_100,key_067_101,key_067_110
,key_067_111},
-- ...
{112,0,key_112_000,key_112_001,key_112_010,key_112_011,key_112_100,key_112_101,key_112_11
0,key_112_111},
-- ...
}
```

22. Handlers

```
-- ...
function key_065_000() keyAdd("a") end -- a
function key_065_001() keyAdd("A") end -- Shift- a or A
function key_065_010() selectToggle2() end -- Ctrl-a
function key_065_011() end -- Shift-Ctrl-a
function key_065_100() altToggle(65) end -- Alt-a
-- ...

function key_112_000() doMenu(112) end -- F1
function key_112_001() unAssigned() end -- Shift-F1
function key_112_010() doMenu(1121) end -- Ctrl-F1
function key_112_011() unAssigned() end -- Shift-Ctrl-F1
function key_112_100() documentGoto("y.$") end -- Alt-F1
function key_112_110() ctrlAltTest1() end -- Ctrl-Alt-F1
```

23. Menu data

```
,{112, "F1", 0,
   {{"0", " ", menu1120 }
   ,{"1", "... ", menu1121 }
   ,{"A", "Atomic", menu112A }
   ,{"S", "ASPX", menu112S }
   ,{"C", "Comment", menu112C }
   ,{"D", "Definition goto", menu112D }
   ,{"G", "Block specify", menu112G }
   ,{"J", "JavaScript", menu112J }
   ,{"L", "Longlines1", menu112L }
   ,{"N", "Nest", menu112N }
   ,{"P", "Prog", menu112P }
   ,{"X", "XSL", menu112X }
   ,{"T", "Test", menu112T }
   }}
,{1121, "Ctrl-F1", 0,
   {{"0", "Link insert", menu11210 }
   ,{"1", "Link insert (comment)", menu11211 }
   }}


-- Menu functions for F1 (112)

function menu1120()
print("menu1120 ::  ")
   addBlockIf(0)
   end

function menu1121()
   print("menu1121 :: ... ")
   addBlockIf(1)
   end

function menu112A()
   print("menu112A :: Atomic")
   addBlock("Atomic")
   end
```

24. Show a menu

```
function menuShow(omit1, escapeAction0)

   if not escapeAction0 then
      escapeAction0 = escapeNothing1
      end

   if editor:AutoCActive() then
      print("*** Warning: menu already active")
      editor:AutoCComplete()
      end

   props["calltip.fore"] = "FFFF00"
   props["calltip.back"] = "FF0000"
   editor.CallTipFore = 0xFFFF00
   editor.CallTipBack = 0xFF0000
```

```
   editor.AutoCMaxHeight = 40
   editor.AutoCMaxWidth = 120
   editor.AutoCSeparator = string.byte(";")

   if not omit1 then

      used1 = {}

      local i
      for i=48,90 do
         used1[i] = false
         end
      for i=58,64 do
         used1[i] = true
         end

      local u = ""

      local k
      for i=1,#menuKeys1 do
         k = menuKeys1[i]

         if used1[k] then
            rmsLua.msgbox("Menu letter \"" .. string.char(k) .. "\" used more than once
in \"" .. menuTitle1 .. "\"")
            u = u .. string.char(k)
         else
            used1[k] = true
            end
         end

      local t = "+"
      for i=48,90 do
         if not used1[i] then
            t = t .. string.char(i)
            end
         end

      menuAdd("", t, t)
      if u ~= "" then
         u = "*** " .. u .. " ***"
         menuAdd("", u, u)
         end
      end

   menuEscapeAction1 = escapeAction0

   state1.in_menu = 1
   editor:UserListShow(99,menuText1)

   if menuPos1 == 0 then

      editor:LineDown()
   else
      local i
      for i = 1,menuPos1 do
         editor:LineDown()
         end
      end

   editor.AutoCSeparator = string.byte(" ")
   end
```

25. Menu escape

```
function menuEscape()
   print("menuEscape")
   if menuEscapeAction1 then
      menuEscapeAction1()
      menuEscapeAction1 = nil
      end
   end
```

26. Menu close

```
function menuClose()
   print("menuClose")
   if editor:AutoCActive() then
      editor:AutoCComplete()
      menuKey1 = 0
      menuEscape()
      end
   end
```

27. Menu key

```
function menuKey(key, shift, ctrl, alt)

   local done1 = false
   local action1 = nil

   if key == 27 then
      print("menu Escape")
      menuClose()
   elseif key == 13 then
      local i = editor.AutoCCurrent
      local map1 = menuMap1[i]
      if map1 then
         menuPos1 = map1
         action1 = menuCmds1[menuPos1]
         done1 = true
         end
   elseif key == 112 then
      print("Menu definition help")
      if menuKey1 ~= 0 then
         menuItemFind1(menuKey1)
         end
      done1 = true
   elseif key == 113 then
      print("Menu action help")
      if menuKey1 ~= 0 then
         end
      sciteCodeGo("")
      done1 = true
   else
      local n = #menuKeys1
      local i
      for i=1,n do
         if menuKeys1[i] == key then
            print("menuKey ch=[" .. string.char(key) .. "] (found)")
            menuPos1 = i
            action1 = menuCmds1[menuPos1]
            done1 = true
            end
         end
      end

   if done1 then
      if true then -- quick action
         editor:AutoCComplete()
         if action1 then
            action1()
            menuEscapeAction1 = nil
            end
         menuKey1 = 0
         end
      end
   end
```

28. Lua errors

Errors in the edit-handling code can be problematic when in is using that editor to change the code with the errors - and which now does not work.