```
************************************************************
*                                                          *
*            WPSetSave20 - An AppleWorks Init              *
*                                                          *
*      - When working in the OA-N 'Change Filename'        *
*           command dialog in the Word Processor           *
*           REVIEW/ADD/CHANGE Window, adds the             *
*             ability to specify the both the              *
*           type of saved file (AWP vs. Text) and,         *
*           in the case of Text files, the type of         *
*           line ending characters, as follows:            *
*                                                          *
*            1. AWP (Word Processor File)                  *
*            2. TXT (Apple Text File - CR)                 *
*            3. DOS (MS-DOS/Windows Text File - CR/LF)     *
*            4. NIX (Unix Text File - LF)                  *
*            5. MLN (Merlin Assembler Source File)         *
*                    (high bits set / <TAB> = $A0)         *
*                                                          *
*       - After providing the opportunity to change        *
*          the name of the file via the <OA-N> key         *
*          sequence, pressing <OA-Return> presents         *
*          a horizontal menu allowing the selection        *
*          of the desired save type.                       *
*                                                          *
*       - Type of file is displayed after the              *
*          REVIEW/ADD/CHANGE in top line of Display        *
*          (provided WPStatus30 Init is installed)         *
*                                                          *
*          Version 2.0 (for AppleWorks Version 5.1)        *
*                    (c) 2017 Hugh Hood                    *
*                                                          *
*       - Seg $17 is patch-in segment.                     *
*                                                          *
************************************************************

               TR             ADR            ; truncate bank address

               XC                            ; enable 65C02 code

* Equates *

FoundRtn       EQU            $87            ; non-zero if <Rtn> pressed
                                             ;  (but, does NOT set for <OA-Rtn>
FoundEsc       EQU            $88            ; non-zero if <ESC> pressed
CArg           EQU            $9E            ; & $9F - pointer
                                             ; (use to dereference pointer)
LastChar       EQU            $360           ; last character entered from keyboard
                                             ;  (high bit set for 'OA' characters
DTFStatus      EQU            $C6C           ; current file status flag byte:
                                             ;  ($01/0001) or $03/0011) - new
                                             ;  ($02/0010) or $06/0110) - changed
                                             ;  ($04/0100) - saved
                                             ;  ($00/0000) - unchanged
NewStr         EQU            $D85           ; used by WP to contain 'Line/Column' String
OldStr         EQU            $E05           ; used by WP to contain 'Line/Column' String
```

```
AWVersion      EQU            $1003          ; $33/51 = 5.1 / $28/40 = 4.0  /
                                             ;  $1E/30 = 3.0
MadeAChange    EQU            $113F          ; posts changed flag on current file
MvLeftRtn      EQU            $1148          ; memory move / follow with TO/FROM/LENGTH
PopStack       EQU            $114E          ; removes the last Escape level
WriteCom       EQU            $118A          ; writes a string on the command line
                                             ;  (follow JSR string address)
imSavePatch    EQU            $3006          ; Patch Manager save routine in SEG.IM
InitAdr        EQU            $4000          ; load address for Init files
ErrorHandler   EQU            $8FD4          ; process bad keys in 'Change Filename'
PatchPoint     EQU            $8FD9          ; patch point in SEG $17 (returns)
PatchPoint2    EQU            $8FCE          ; patch point in SEG $17 (bad keys)
MMInDoc        EQU            $7C68          ; AWP file flag byte:
                                             ;  bit 0 (%00000001)/$01 = Mail Merge (AWP)
                                             ;  bit 2/5 (%00100100)/$24 = Merlin Source
                                             ;  bit 3/5 (%00101000)/$28 = Unix TEXT
                                             ;  bit 4/5 (%00110000)/$30 = MS-DOS TEXT
                                             ;  bit 5 (%00100000)/$20 = TEXT
Patch2Run      EQU            $9D00          ; final destination for new code
                                             ; NOTE: Seg $17 runs from $7E00 - $9BD5.
                                             ;        Other Segs loading at $7E00
                                             ;        use adjacent memory up through
                                             ;        $A70D, but Seg $17 will re-load
                                             ;        after those have finished.
newCC2S        EQU            $AB40          ; concatenate (2) strings
                                             ;  (first String has second added
                                             ;    at end)
newPushStack   EQU            $AB4F          ; ; add a level to Escape road map
                                             ;  (follow with string address)
PatchAdr       EQU            $BB00          ; load address for patch code
                                             ;  (NOTE: uses ProDOS I/O buffer -
                                             ;          1K max length -
                                             ;           $BB00 - $BEFF)
GetMenuBar     EQU            $D032          ; build inverse horizontal menu bar at
                                             ;  bottom of screen
                                             ; - select an item by typing its first
                                             ;    character or pressing return when
                                             ;    the item is highlighted.  Use arrows
                                             ;    to move highlighting.
                                             ; -  Upon exit, capitalize the select one.
                                             ; - follow JSR with:
                                             ;   - DA (where to go if escape)
                                             ;   - DB (number of items)
                                             ;   - DA (description #1)
                                             ;   - DA (routine #1)
                                             ;   [Up to (7) more pairs - (8) MAX]
GetMenuBar2    EQU            $D04A          ; same as GetMenuBar except LDA #x prior
                                             ;  to JSR for default selection
                                             ;  - NOTE: Selection # (both default and
                                             ;           final) is stored (2) bytes
                                             ;           before the beginning of the
                                             ;           actual GetMenuBar routine,
                                             ;           and (4) bytes before actual
                                             ;           GetMenuBar2 routine, which
                                             ;           is JMPed to by $D04A
                                             ;           [dereference pointer to use]
                                             ;
               ORG            InitAdr        ; ($4000)
               TYP            $06            ; create binary file

********************************
*          Init Header         *
********************************
START
               JMP            IStart         ; skip over header

**--------------------------------------------------------------

               ASC            'mb'           ; Init ID Bytes (AW 5.1)
               DB             $14            ; Init Version - programmer assigned
                                             ;  e.g. - $0A/1.0 $0C/1.2 $19/2.5
               STR            'WPSetSave20'  ; Init Screen Name
               HEX            0000           ; Header End Bytes

**--------------------------------------------------------------

IStart

               LDA            AWVersion      ; AppleWorks version #
               CMP            #$33           ; Is it Version 5.1?
               BNE            Done           ; disregard - wrong version

PatchH17       JSR            imSavePatch    ; call patch manager
               DW             Code1          ; beginning of patch1 code ($40xx)
               DW             Patch1End-PatchAdr
; length of patch code
               DW             $0017          ; SEG number to patch
                                             ;  ($17 = AWP Util SEG)


PatchH15       JSR            imSavePatch    ; call patch manager
               DW             Code1+Patch1End-PatchAdr
; beginning of patch2 code ($40xx)
               DW             MoveStart-PatchAdr+Patch2End-Patch2Run
; length of patch code
               DW             $0017          ; SEG number to patch
                                             ;  ($17 = AWP Util SEG)

Done           RTS                           ; back to Init Manager

**--------------------------------------------------------------

Code1          EQU            *              ; (will be $40xx)

               ORG            PatchAdr       ; (Patching Code is moved and run
                                             ;   @ $BB00 by Init Manager)

HookBytes      HEX            0000           ; first bytes for $17 Patch
               LDA            #$4C           ; JMP instruction
               STA            PatchPoint     ; $8FD9 in SEG $17
               LDA            #Patch2Run     ; low byte of new code
               STA            PatchPoint+1
               LDA            #>Patch2Run    ; high byte of new code
               STA            PatchPoint+2

               LDA            #$4C           ; JMP instruction
               STA            PatchPoint2    ; $8FCE in SEG $17
               LDA            #CmdHandler    ; low byte of new code
               STA            PatchPoint2+1
               LDA            #>CmdHandler   ; high byte of new code
               STA            PatchPoint2+2

               RTS

Patch1End      EQU            *

**--------------------------------------------------------------

               ORG            PatchAdr       ; (Patching Code is moved and run
                                             ;   @ $BB00 by Init Manager)

MoveCall       HEX            0000           ; first bytes for $17 Patch
               JSR            MvLeftRtn      ; move new code to run location
               DA             #Patch2Run     ; ($9D00)
               DA             #MoveStart     ;
               DA             Patch2End-Patch2Run

               RTS                           ; patch hook-in done

**--------------------------------------------------------------

MoveStart      EQU            *              ; (will be $BBxx)

               ORG            Patch2Run      ; ($9D00)

               LDA            LastChar       ; check last keypress
               CMP            #$8D           ; was it <OA-Rtn>?
               BEQ            PatchStart     ; do 'Change Save Type'

SkipType       JMP            PopStack       ; skip 'Change Save Type'

PatchStart     JSR            PopStack       ; remove last 'title' ('Change Filename')

               LDA            MMInDoc        ; get file flag byte
               CMP            #$20           ; is it Apple Text (CR)?
               BEQ            TXTTyp         ;
               CMP            #$24           ; is it Merlin Source?
               BEQ            MLNTyp
               CMP            #$28           ; is it Unix Text (LF)?
               BEQ            NIXTyp
               CMP            #$30           ; is is MS-DOS Text (CR/LF)?
               BEQ            DOSTyp

AWPTyp         LDA            #1             ; menu item #1
               STA            Default+1      ; default for menu routine
               BRA            :B

TXTTyp         LDA            #2             ; menu item #2
```

```
               STA            Default+1      ; default for menu routine
               BRA            :B

MLNTyp         LDA            #5             ; menu item #5
               STA            Default+1      ; default for menu routine
               BRA            :B

NIXTyp         LDA            #4             ; menu item #4
               STA            Default+1      ; default for menu routine
               BRA            :B

DOSTyp         LDA            #3             ; menu item #3
               STA            Default+1      ; default for menu routine


:B             JSR            newPushStack   ; add level to Escape road map
               DA             TitleStr
               JSR            WriteCom
               DA             PromptStr
Default        LDA            #1             ; default selection # (changes)
               JSR            GetMenuBar2
               DA             GoBack
               DB             5              ; # of selections
               DA             MenuAWPStr
               DA             SetType
               DA             MenuTXTStr
               DA             SetType
               DA             MenuDOSStr
               DA             SetType
               DA             MenuNIXStr
               DA             SetType
               DA             MenuMLNStr
               DA             SetType



SetType        LDA            GetMenuBar+1   ; low byte of pointer to actual routine
               STA            CArg           ; low byte pointer for indirect reference
               LDA            GetMenuBar+2   ; high byte of pointer to actual routine
               STA            CArg+1         ; high byte pointer for indirect reference
               DEC            CArg           ; subtract first of (2) bytes for location
                                             ;  of stored menu selection
               LDA            CArg           ; load for CMP operation
               CMP            #$FF           ; did we cross page boundary?
               BNE            :C             ; if not, do one more decrement
               DEC            CArg+1         ; cross page - subtract (1) from high byte
:C             DEC            CArg           ; subtract second of (2) bytes
               CMP            #$FF           ; did we cross page boundary?
               BNE            :D             ; if not, proceed with rest
               DEC            CArg+1         ; cross page - subtract (1) from high byte
:D             LDA            (CArg)         ; retrieve selection #
               TAX                           ; put it in 'X' for CPX operations
               LDA            MMInDoc        ; read AWP file flag byte
               CPX            Default+1      ; was selection the initial default?
               BEQ            GoBack         ; if so, then done
               JSR            MadeAChange    ; flag change (DTFStatus)
               CPX            #2             ; was selection TXT?
               BEQ            SetTXT         ; set TXT flag
               CPX            #3             ; was selection MS-DOS?
               BEQ            SetDOS         ; set DOS flag
               CPX            #4             ; was selection UNIX?
               BEQ            SetNIX         ; set NIX flag
               CPX            #5             ; was selection Merlin Source?
               BEQ            SetMLN         ; set MLN flag

                                             ; if not TXT/DOS/NIX/MLN, then is AWP
SetAWP         AND            #%00000001     ; force all bits low except Bit 0,
                                             ;  in case this is a Mail Merge AWP file
               STA            MMInDoc        ;
               BRA            GoBack

SetTXT         LDA            #%00100000
               STA            MMInDoc
               BRA            GoBack

SetDOS         LDA            #%00110000
               STA            MMInDoc
               BRA            GoBack

SetNIX         LDA            #%00101000
               STA            MMInDoc
               BRA            GoBack

SetMLN         LDA            #%00100100
               STA            MMInDoc


**--------------------------------------------------------------

GoBack         JMP            PopStack       ; go back to original code

**--------------------------------------------------------------

CmdHandler     EQU            *

               LDA            LastChar       ; check last keypress
               CMP            #$8D           ; was it <OA-Rtn>?
               BNE            :NextCheck1
               JMP            PatchStart     ; proceed to 'Change Save Type'

:NextCheck1    LDA            FoundEsc       ; was it <ESC> key?
               ORA            FoundRtn       ; check <Rtn> key as well
               BNE            :NextCheck2

               JMP            ErrorHandler   ; back to 'Change filename' routine

:NextCheck2    JMP            SkipType       ; back to AppleWorks


**--------------------------------------------------------------

PromptStr      STR            'Save Type'    ; prompt for horizontal menu
```

```
MenuAWPStr     STR            'Word Processor'
MenuTXTStr     STR            'Apple Text'
MenuDOSStr     STR            'MS-DOS Text'
MenuNIXStr     STR            'Unix Text'
MenuMLNStr     STR            'Source-Merlin'
TitleStr       STR            'Change Save Type'

**--------------------------------------------------------------

Patch2End      EQU            *              ;
               SAV            I.WPSETSAVE20
               LST            OFF

               END


```