MacOSX_UNIX_Linux Dual Safe Way, Technical Note

Please refer to "**Intro.PDF(PC/Mac)**" for introductory concept about Dual Safe Way. Following are technical note to run MacOSX_Unix_Linux versions successfully. This protocol is tested by a few representative platforms (MacOSX Panther; Sun UNIX Solaris 8; RedHat Linux 6.2). Excuse TuneFind that this is not a double-click version but a command-line version, and if yours doesn't respond as described below. One more thing. Unfortunately, a poor-but-friendly TuneFind's chief technology officer still hasn't figured out how to mount ZIP drive on UNIX and Linux machines. This protocol often said "ZIP" all the way, but it is only tested by "Floppy" drive in case of UNIX_Linux.

CONTENT

1. Downloading and Unzipping:

This is a StuffIt Expander archives (MacOSX) / GNU-style zipped archives (UNIX_Linux). In MacOSX most of Web browsers automatically un-stuff it behind the scene. While in UNIX_Linux, please do unzipping it manually in **Terminal** as below,

```
 [lan] % gunzip -d unix_dualway.tar.gz  ↵  (or linux_dualway.tar.gz)
 [lan] % tar xvf unix_dualway.tar  ↵     (or linux_dualway.tar)
```

This makes three executable files (**`1Bat, 2Er, 3Diff`**), which are explained in the next paragraph.

2. Summary Protocol of MacOSX_UNIX_Linux DualWay:

Sorry, you can't use Double Clicks. Please run three executable files (**`1Bat, 2Er, 3Diff`**) in **Terminal** (text mode, see next paragraph).

`[lan] %` (Set Path)	Please see below, § 5 How to set your working folder as one of executable Paths
`[lan] %` (Make 'Header.txt' file)	Please see below, § 3 How to make **Header.txt**.

[lan] % **1Bat** *1) ↵	Creates a Batch File -> needs **Header.txt**
[lan] % chmod 777 Batch.txt ↵	Prepares to run **Batch.txt**
[lan] % (may need to refresh Terminal) *2)	
[lan] % **Batch.txt** ↵	Makes catalogs of ZIP (or alternative) and SDW
[lan] % **2Er** ↵	Erases part(s) of owner Stamp in catalogs -> needs **Header.txt**
[lan] % **3Diff** ↵	Reports difference between SDW and ZIP -> needs **Header.txt**

1. Unlike PC/MacOS7:8:9, MacOSX_UNIX_Linux DualWay has steps of 1, 2, and 3. As usual, after 1Bat (Batch generator) and before 2Er (Stamp eraser), "Batch.txt" must be run. For typing convenience, Commands are shorten.

2. In a certain system, you need to refresh the Terminal. Just go ahead the next command line [% Batch.txt ↵], and if you get an error message claiming 'can't execute Batch.txt', then close Terminal, restart Terminal, and do it again.

3. Before running MacOSX_UNIX_Linux DualWay, please play around on Terminal (Console) and Shell:

Terminal is the text-based screen interface, and years ago many UNIX people (originated from Cambridge U. and AT&T Bell Labs) run the computer far from the machine in text-mode, somewhere in a cozy quiet corner with just only Terminal screen, then it got the name 'Terminal'. Please get started **Terminal** on your machine.

[MacOSX] Finder -> Go -> Utilities -> (Utilities ->) **Terminal** (a black screen icon).
[Solaris] (If CDE) Right-Click on mouse -> Workspace Menu -> Hosts -> **Terminal Console**
[RedHat] (If GNOME) Screen-icon at bottom tool-bar; OR Right-click of mouse at Desktop ->
New -> **Terminal**

While in **Terminal**, the most important switch is **Shell**. **Shell** is an encapsulated working environment where command lines can be run. There are a couple of **Shells** available to switch. Historically speaking, Bourne shell, C-shell, BASH (Bourne again Shell), Tcsh (Tenex C-shell), have being developed gradually. Before Graphical User Interface (GUI, pronounced 'gu-ai') has been developed, **Shell** was the main interface.

Currently TuneFind doesn't have a sophisticated GUI-based compiler, and besides, DualWay only needs a simple Text input/output. Thus, PC/MAC-Classic DualWay was ported into the Terminal-Shell interface. Sorry, you can't use Double Clicks, but typing commands wouldn't be so bad, and you may feel Back-To-the-Past time travel.

If you are not familiar with Terminal and Shell, please play around a bit, before jump into DualWay. In case of MacOSX, it is convenient to drag **Terminal** at the bottom menu for a quick access, then it stays right at your hand. Try typing a couple of commands, such as below,

```
[lan] %*3)  pwd     (print working directory)
[lan] %  env        (show Environmental settings)
[lan] %  set        (show current settings)
[lan] %  ls         (list current directory contents)
[lan] %  ls -F      (list current directory contents with a Flag mark, folder name with slash (/);
                      executable file with asterisk (*).
[lan] %  ls -l      (list current directory contents in a long format)
[lan] %  ls -a      (list ALL current directory contents including hidden dot files)
[lan] %  cd         (change directory, 'cd ..' for upper directory, 'cd foldername' for lower
                      directory)
```

[3] Generally speaking, C-shell and its lineage often has **'%'** prompt, while Bourne shell and its lineage often has **'$'** prompt.

When commands are responding well, let us get into a little business. If commands are not responding, see your System Administrator or a local Professional. Now, please carefully review the following environmental variable settings at your machine after you type **set** OR **env**.

```
HOME (home)
SHELL (shell)
PATH (path)
```

You will see a lot of slashes (/) with some words like **bin**, **sbin**. The left most slash (/) is called **root** of the computer, and anything after that is sub- and sub-sub-folder names, delimited by the slash (/). Thus, they points to 'what is where' in the machine. The **bin** stands for 'binary', and traditionally most of executable files are stored there in **bin** because it is good idea to store Binary Executable file separately from other Text files. The small commands like **set**, **env**, **ls** are also stored in one of those **bin**. And a big command like 'Shell' is also there: Bourne-shell (**bin/sh**); BASH (**bin/bash**); C-shell (**bin/csh**); Tenex C-shell (**bin/tcsh**). Find out which **Shell** is your current setting.

Did you play around enough? Then, let us do a real business of DualWay. As explained at "Intro.PDF(PC/Mac)", please copy the entire content of your ZIP disks or Floppy diskettes into one of hard disk under the folders which has the same name of the disk label. Your ZIP (Floppy) disks stay as the master, since the removable disks move around machines, and change always. The entire copies on the hard disk is called "shadow" (SDW), and here comes the name of DualWay. And the three executable files (**1Bat, 2Er, 3Diff**) are the step-by-step scanning program to detect any difference between ZIP (master removable disk) and SDW (shadow copies on hard disk).

4. How to make "Header.txt" file.

This DualWay program needs your ZIP (Floppy) disk's directory/folder names information.

Protocol :

(1) Please mount a disk.

[MacOSX] Just insert a ZIP disk or Floppy diskette, then it is automatically mounted.
[Solaris 8] Start File Manager: File -> Open Floppy.
[RedHat 6.2/Gnome] Right-click mouse on "Floppy 0" icon -> Mount device.

(2) Start **Terminal**, and please go onto the disk by `cd` (change directory).

[MacOSX] To find out the ZIP disk's mounting place, run Disk Utility (under Go -> Utilities). In the left pane, it shows all mounted disks (Hard_Disk, ZIP disk, or Floppy disk). Click on the ZIP disk, then at the bottom of right pane, you will see

```
"Mount Point: /Volumes/ZIPdiskname"
[lan] % cd /Volumes/ZIPdiskname   ↵
```

[Solaris 8] In File Manager you will see the floppy disk name such as "noname".

```
[lan] % cd /floppy/noname   ↵
```

[RedHat 6.2/Gnome] You can see it by File Manager, and it is usually,

```
[lan] % cd /mnt/floppy   ↵
```

(3) [All three platforms, exactly the same] If you want to scan the folders at the root of ZIP (Floppy) disk, don't change directory any more. If your scanning folder is under sub-, or sub-sub-folders, then change directory until you reach there. Then issue one of the following command line. Please name the resultant file "Header.txt", which has a list of folder names to be scanned, each one per each line.

```
[lan] % ls > Header.txt
```

(Then bring this file to the same folder where three executable files exist.)

OR do below, for example, three executable files are at /DualWay at your home.

```
[lan] % ls > ~/DualWay/Header.txt
```

(4) (Optional) To review the content of **Header.txt**

At GUI,

[MacOSX] Double clicks **Header.txt** in Finder.
[Solaris 8] Double clicks **Header.txt** in File Manager.
[RedHat 6.2/Gnome] Double clicks **Header.txt** in File Manager (Midnight Commander); OR right-mouse click on the file, and open with gEdit.

At Terminal, for example,

```
[lan] % cd ~/DualWay
[lan] % more Header.txt
```

Alternative MacOSX Protocol:

You can use the same technique as described in Classic Macintosh to make **Header.txt**. Select all folders; [Apple-Command] + C-key to copy folder names into a buffer; then open TextEdit (plain mode) and [Apple-Command] + V-key to paste. However, due to end of line bad INVISIBLEs, **1Bat** doesn't function well to write a good **Batch.txt**. When you made **Header.txt** by this COPY-PASTE way, you have to place Hard-Return manually to the end of each folder name. The look of **Header.txt** becomes sparse, due to extra blank followed by each folder name, but this works well.

Tip: Bad INVISIBLEs at the end of lines are common phenomena, when you copy PC text file to Mac, or Linux file to PC, etc. So please get used to it how to convert Bad INVISIBLEs to Good INVISIBLEs.

5. How to set your working folder as one of executable Paths:

5.1 Setting Path Temporarily:

For example your default Paths are four locations in Shell,

```
 path (/bin /sbin /usr/bin /usr/sbin)   OR
 PATH=/bin:/sbin:/usr/bin:/usr/sbin
```

(Those two lines are the same meaning, just different expression in a Shell)

we need to add one more for DualWay.

The OSX_UNIX_Linux DualWay is assuming that you are working at your Home (the shortcut is '**~/**' (tilde and slash). Home in UNIX/Linux/MacOSX means your login workplace on the computer, not your HOME where you sleep. To examine it, at the **Terminal**, type **pwd** (Print Working Directory). It tells where your login Home is. When you refer to your Home location, you can type exactly as it is, or you can conveniently use a special mark '**~/**' (tilde slash) which is exactly the same meaning of Home location.
For example, your Home location is '**/Users/johngiraffe**', then you can refer it just by '**~/**'. They are synonym. This provides convenience for typing and avoid a typo.

[Example MacOSX]
Suppose you made a working directory called 'DualWay' under your Home, and three executable files are there. Below, there are a few examples of adding your own Path in two different Shell ('tcsh' and 'bash').

<<tcsh>>
[lvl-mac] giraffej% **set path = ($path /Users/johngiraffe/DualWay)** (In here, **$path** represents all four Paths /bin, /sbin, /usr/bin, /usr/sbin. And adding a new one.) OR
[lvl-mac] giraffej% **set path = ($path ~/DualWay)** (in here, **~/** is /Users/johngiraffe/)

<<BASH>>
(While typing, no any blank space all the way, please.)
[lausd-Mac] giraffej$ **PATH=$PATH:/Users/johngiraffe/DualWay** (Likewise, **$PATH** represents all four Paths /bin, /sbin, /usr/bin, /usr/sbin. And adding a new one.) OR
[lausd-Mac] giraffej$ **PATH=$PATH:$HOME/DualWay** (in here, $HOME is /Users/johngiraffe)

```
[lausd-Mac] giraffej$ PATH=$PATH:~/DualWay
```

(in here, **~/** is /Users/johngiraffe/)

All are the same meaning: "Add one more Path, **/Users/johngiraffe/DualWay** as an executable Path. When you successfully add one more your own Path to the previous four paths, your path setting will become five, and it looks like below when you examined by '**set**' or '**env**' commands.

```
 path (/bin /sbin /usr/bin /usr/sbin /Users/johngiraffe/DualWay)
```

Now, you can run executable files of DualWay and **Batch.txt** file in your newly created Folder 'DualWay'. The magic environmental word $path, $PATH are case sensitive in each shell, so please type carefully.***4)** They ($path, $PATH) hold all default Paths **(/bin /sbin /usr/bin /usr/sbin)**, so make sure to include it.

5.2 Setting Path Permanently:

The temporarily Path setting above works at all three Platforms. Furthermore, it is a clever idea to write down this Path Setting command line into your login file. Depend on your Shell environment, your login file is different.

Your Shell	Login file
C-shell (csh) /Tenex C-shell (tcsh) ->	**.login**
BASH (bash) ->	**.profile**

Please make a note that those files are a kind of your personal system extensions for environmental setting. For this reason, they have a dot **"."** at the beginning of filename, and they are hidden by a regular **ls** command except **ls -a** (List **A**ll including hidden files). They must be placed at your Home directly. The dot file specifies your environment during your login, and it doesn't affect other users. Once your path setting command line was placed into one of those login files (**.login** or **.profile**), whenever you login, the path is automatically set to include your customization. It is just one time labor and you can use it forever. This permanent path setting works exactly the same way in three platforms. Below is a sample in case of C-shell(csh)/Tenex-C-shell(tcsh). In **.login** file, just place the command line. That's it. Noting else is needed. To write this line in **Terminal**, any UNIX editor works (ex, vi, pico, emacs); at GUI you may use any simple text editor.

.login file

```
set path = ($path ~/DualWay)
```

4. Generally speaking, in UNIX/Linux words are case-sensitive, upper letters and lower letters means differently. While MacOSX, some Paths and folder names are not case-sensitive. However, typing in case sensitive manner is a safe practice.

6. How to set 'Batch.txt' as an executable file as well as read and write:

In brief, you need to 'change the mode' of the **Batch.txt** file. The command is `chmod`, and a good mode is '777', for example. For more details about which number refers which mode, and how to calculate, refer to an UNIX introductory book. Briefly, read permission is 4; write permission is 2; executable permission is 1. Thus, to have them all, add up 4 + 2 + 1 = 7. Once you have them all, then you allow to have them to you, to group, and to the public. So, '777'.

```
 % chmod 777 Batch.txt  ↵
```

Practical Note: To run this **Batch.txt** after the creation of Batch.txt and chmod 777, in certain OSX it often requires to close and re-start **Terminal** to take effect of your change. In some other cases, it doesn't require the renewal.

7. Trouble shooting:

Messy **Batch.txt** comes from **Header.txt**.
When you make **Header.txt** by the way as described in "Intro.PDF(PC/MAC)", namely, SELECT folders -> COPY -> PASTE, then this happens. Please read § **4. How to make "Header.txt" file of this .PDF**, which tells what to do. You can fix the file by manually entering Hard returns (Enter) at all folder names. This makes **Header.txt** file sparse: folder name, blank line, folder name, blank line, ... but it works well.

```
 --- : Command not found
```

When you try to run an executable file, and get the message above, this likely means executable Path isn't set well.

```
 --- : No such file or directory
```

When you try to run the **Batch.txt** executable file, and get the message above, this likely means scanning Path written in **Batch.txt** is not accurate. So carefully review the **Batch.txt**. Make sure that your ZIP(Floppy) disk is inserted (and mounted).

Footnote:

- MacOSX DualWay (three executable files) was compiled by Apple's version of GCC-3.3 compiler by GNU group (G! not UNIX).
- UNIX DualWay was compiled under Solaris 8 by Sun UltraSPARC-IIi Chip with GCC-2.95.3 compiler.
- Linux DualWay was compiled under RedHat Linux 6.2 by Intel i686 Chip with GCC-2.0 compiler.
  The codes will be posted soon in order to work with other platforms (one moment).

Revised July 2005