Inessential Dotfiles[1]

Version 0.8

`/afs/sipb.mit.edu/project/doc/idotfiles/iDotfiles.dvi`

The Student Information Processing Board

Heather Drake <pipa@mit.edu>, Camilla Fox <cfox@mit.edu>, David Golombek <daveg@mit.edu>

July 16, 1999

Contents

1 Introduction

The aim of this document is to help you understand what goes on when you log in on an Athena terminal, and to help you to customize your own login session. This sequence of events is controlled by the scripts that are run when you log in, and the configuration files associated with some programs. Of these scripts and configuration files, your dotfiles are those which live in your home directory and have names which begin with a '.' (out of convention). You can list what dotfiles you have by doing `ls -a`, which tells ls to show files which it normally hides. This document will cover what happens to a default account when it's user logs on, and the ways in which you can modify this default behavior.[2] Note that all of the material covered here is subject to change without notice. We will try to keep it current, but may fall behind, so beware.

1.1 What do I have to know?

You don't need to already know C or any other programming language to customize your login. Most Athena dotfiles are written in csh, which is a scripting language, whose syntax is intended to loosely resemble C. Learning csh can help in writing dotfiles, but is not necessary for simple modifications. Generally, all that is needed is an interest in what the computer is doing, and the willingness to spend some time experimenting.

1.2 What do I gain from knowing about dotfiles?

By modifying your dotfiles, you will be able to customize your login session so that it's more aesthetically pleasing, more productive, and more personalized. By adding aliases, you can save keystrokes on the command line. By adding commands to a startup file, you avoid having to manually start the same programs every time you log on. Modifying your dotfiles is a great way to begin deeper explorations of computers and programming.

1.3 Other sources of information

Because of its scope, this document makes certain assumptions about your knowledge of UNIX. If you want to learn more about UNIX, there are a variety of sources of information. Man pages should exist for all of the commands used in this file (if they don't, it's a bug, which should be reported using `sendbug`). Other SIPB Inessential guides and the publications from IS are also great sources of information; SIPB Inessential Guides can be found at `http://www.mit.edu/sipb/docs.html` or `/mit/sipb/doc/` and you can visit `http://web.mit.edu/is/pubs` for a listing of IS publications. It is recommended that you understand each of the commands that you are using before you put them in your dotfiles, so that you can fix your dotfiles if things go wrong.

2 What happens before I log on?

When you walk around an Athena cluster, you might notice that the unused machines are running a program called xlogin. This program is responsible for the message that bounces around the screen and for the username and password box that pop up when you press a key. Xlogin also allows you to run certain programs without logging in, such as xcluster, as well as to specify several different login options.

Similarly, a machine which you can telnet to, such as the athena dialups, is running a program, telnetd, which accepts the connection, then runs a program called login, similar to xlogin, described above. Such a login (which also occurs when using rlogin, ssh, or the **Terminal-style session** from xlogin) is called a TTY login. The sequence of dotfile execution is substantially different for TTY logins than for X logins, and is covered separately in Section 6.

[2] Both terminal and dialup logins are covered.

3 Overview

There is a complex sequence of different files that are run (or sourced[3]) during a login session. This sequence is described by the following diagram.

- User types their username and password into xlogin.
- xlogin runs the sh script, /etc/athena/login/Xsession, which:
  1. Makes Athena keymap modifications, if required
  2. Sources ~/.xsession[4], if it exists, otherwise sources /usr/athena/lib/init/xsession, which:
     (a) Checks for ~/.hushlogin
     (b) Sets the XSESSION environment variable
     (c) Sources ~/.cshrc[5]
        i. Sources /usr/athena/lib/init/cshrc, which:
           A. Sets the variables athena_path, hosttype, bindir, athena_manpath, home
           B. Sets the environment variables: MORE, EDITOR, VISUAL, MM_CHARSET, MANPATH, ATHENA_SYS, ATHENA_SYS_COMPAT, PRINTER
           C. Sources ~/.environment
           D. Sets path using $athena_path
           E. Sets prompt, history, cdpath, interactive
           F. Aliases renew, term, mail, Mail, xresize, logout, add, setup, setup_X, setup_tty, remove
           G. Sources ~/.cshrc.mine
     (d) Loads the Xresources, by running `xrdb -merge ~/.Xresources`
     (e) Starts window manager, mwm[6] unless another has been specified
     (f) Starts dash, unless skip_dash is set
     (g) Creates logout button, unless skip_xlogout is set
     (h) Starts initial xterm, unless skip_initial_xterm is set
     (i) Starts zwgc (a zephyr client), checks for a new message of the day, checks for new mail, and pings the OLC server, all unless skip_x_startup is set
     (j) Sources ~/.startup.X[7]
     (k) Prepares for logout with "exec session_gate -logout"
     (l) Waits for user to logout
     (m) Sources ~/.logout
  3. Runs kdestroy, unlog, and fsid -u -f $USER

The rest of this document will attempt to cover these dotfiles in greater depth, and to give you a feel for ways to make modifications.

[3]To source a file means to run `source filename`, which has the same effect as executing that file, except that it modifies environment variables in your current environment, rather than creating a new shell and keeping all such modifications within it.

[4]This will substantially change the behavior of your login.

[5]If you change your shell, this will be a different file, but in the case of a supported shell, the effect should be similar.

[6]4dwm is run instead of mwm, if you are on an SGI, and have not set skip_sgi.

[7]~/.startup.sgi is sourced instead, if you are logging in on an SGI, and skip_sgi is not set.

4 The default dotfiles

When you logged into Athena for the first time, there was already a default set of dotfiles in your home directory. If you have modified or deleted them, and you want to get the original ones back, you can make copies of the ones in /usr/prototype_user/. This section covers what happens when you log in with these default dotfiles.

4.1 /etc/athena/login/Xsession

The xlogin program runs a script called Xsession, located in /etc/athena/login on all standard Athena workstations. This script does very little under the normal login sequence. It sets the TTY (Teletype Terminal) characteristics to defaults for interactive use, and uses xmodmap to set Athena keymap modifications if required for that type of computer, then executes the script below. After /usr/athena/lib/init/xsession returns, Xsession is also responsible for clean up tasks such as destroying your kerberos tickets and your afs tokens.

4.2 /usr/athena/lib/init/xsession

This is probably the most important of the startup scripts. This script both starts some of the most important programs and sources other files that control the rest of your login.

This script first checks for the existence of a ~/.hushlogin, which tells the login scripts to avoid printing status messages. It then sets the XSESSION environment variable, which is later used to determine if this is a X or TTY login.

Next, this script sources the ~/.cshrc file, covered below, which in turn sources many user controllable files (most importantly, ~/.cshrc.mine and ~/.environment). Among the possible uses of these files, they also serve as a place to set configuration parameters for the continued running of /usr/athena/lib/init/xsession.

After completing the sourcing of these user files, it begins to configure the X Window system. It does this with the command

```
xrdb -merge ~/.Xresources
```

For more information on this file, see Section 5.5. Finally, the window manager itself is started. The default Athena window manager is the Motif Window Manager, but users can select others by setting the WINDOW_MANAGER environment variable (see Section 5.2).

Next, xsession goes through and starts a wide variety of programs. Each of these programs has a skip_*programname* variable associated with it, which, if set, prevents the program from being run. Each of these programs are listed below, along with a brief description, and the skip_*programname* variable to set to skip it.

- Dash: (skip_dash) The menu bar appearing at the top of the screen, allowing programs to be run by clicking on their menu item.
- Xlogout: (skip_xlogout) The button you can click to log out of a workstation.
- Initial Xterm: (skip_initial_xterm) The first xterm to be started on your screen.
- X Startup Activities: (skip_x_startup) This runs zwgc (the zephyr windowgram client), gets the message of the day (motd), runs a mail check, and pings the OLC server to check for outstanding messages from OLC.
- Lert: (skip_lert) Shows any Athena Accounts messages to you[8] (such as account deactivation warnings).

If you set variables to skip any aspect of xsession's usual sequence make sure that you have a method of replacing its functionality. For example, if you set skip_dash, you will need to remember where to find all

[8] It is recommended that you do not skip lert unless you are planning to run it somewhere else in your dotfiles.

the programs that you normally use, and if you skip_initial_xterm as well, you will need to provide yourself with a way to get a shell.

Once the selected combination of programs have been run, xsession runs any programs that the user specifies in their ~/.startup.X file (see Section 5.6).

Finally, the xsession file goes to sleep until the user is done with the login session. This is done using the command

```
exec session_gate -logout
```

which waits until the Athena session is done, then executes the ~/.logout file (covered below) and exits.

4.3 ~/.cshrc

The ~/.cshrc sources other files and sets your path (the directories that are searched for programs you type on the command line). The recommended method of adjusting the csh initialization sequence is to create any of the following files (listed with the section in which it is covered):

- ~/.environment: Section 5.2
- ~/.cshrc.mine: Section 5.4

While your ~/.cshrc seems an obvious file to modify, remember that unless you are removing or changing functionality, you will be better off making a ~/.cshrc.mine and putting your customizations there. If you do modify your ~/.cshrc, be careful, because if you don't in turn source /usr/athena/lib/init/cshrc (see below), you will not automatically get any changes that Athena may make to the system-wide file at a later date. Be sure you know what you are doing, and plan on keeping track of changes to Athena. In reality, these sytem-wide files are changed very rarely.

Also note that this file will be sourced by your shell, every time you start a new shell instantiation (such as an xterm window). Thus, the less you do within this file (and the files it sources), the faster your xterms will load.

4.4 /usr/athena/lib/init/cshrc

This file is responsible for setting up several default variables for all Athena users. It sets both shell and environment variables, which are listed below, with a brief description of what they do. More information on the effects of these variables on the shell can be found with `man tcsh`. Since tcsh is based on csh, `man csh` will also be useful.

Shell Variables

- `athena_path` Generic path before any lockers are added
- `hosttype` Machine type
- `bindir`[9] Directory in which to find binaries for this platform
- `athena_manpath` Default search path for man files
- `host` The hostname of the machine
- `autolist` List choices when tab completing
- `home` Home directory (~) for user
- `noclobber` Don't overwrite files with redirection
- `prompt` The default shell prompt

[9]This is deprecated.

- `history` Number of commands saved in shell history
- `cdpath` Path to search for directory changes
- `interactive` Allow editing of commands typed to the shell
- `nostat` Don't tab complete within specific directories (on Athena, this is used to prevent accidentally stating /afs which would take a long time and tie up your shell)

Environment Variables

- `EDITOR` Sets the editor that programs will start by default
- `VISUAL` Similar to EDITOR, but used by programs which do not wish to assume that your editor is not a line editor
- `MM_CHARSET` Default character set
- `MANPATH` Default search path for man files (set to same as $athena_manpath)
- `ATHENA_SYS, ATHENA_SYS_COMPAT` Used by the add alias when adding lockers and modifying path and manpath
- `PRINTER` Default printer to use

Aliases

- `renew` Renew Kerberos tickets after they have expired
- `term` Change terminal type
- `xresize` Update terminal settings to the current size of the xterm
- `logout` Kill the session gate, and exit the current shell, causing the user to logout
- `add` Attach a locker, then add it to path and manpath
- `setup` Add a class locker, and run its startup script
- `remove` Remove a class locker

As well as setting these variables and aliases, ~/.cshrc sources your ~/.environment, and your ~/.cshrc.mine, should they happen to exist.

5 Optional dotfiles

The section outlines the dotfiles which are not created by default, but which are used, if they exist.

5.1 ~/.hushlogin

The default dotfiles will avoid printing status messages if this file exists.

5.2 ~/.environment

This is a good place in which to add or attach lockers that you expect to need during your session. Due to the slowness of invoking attach, it is recommended that you do all of your adds or attaches on a single line. For example:

```
add graphics outland sipb
```

5.3 ~/.path

The usage of ~/.path is deprecated, although if you have added customizations already, you might have one. If you are using it, it will be sourced by your ~/.cshrc. The recommended tool for customizing your path shell variable is add, which increased in functionality with the 8.2 (1998) release, such that you can use it to put a specific directory in your path, instead of only lockers.

5.4 ~/.cshrc.mine

This is the suggested place to set custom variables for (t)csh, create aliases, unset sytem defaults, and other shell customizations. Some popular tcsh specific[10] things to do in your ~/.cshrc.mine include:

```
set prompt = '[%n@%m]%c%# '      # initialize prompt
set fignore = '~'                # ignore ~ files when tab completing
alias zw 'zwrite'                # shorten frequently used commands
alias cp 'cp -i'                 # don't accidentally overwrite files with cp
```

5.5 ~/.Xresources

The xsession runs xrdb on /.Xresources. In brief, xrdb loads the specifications listed in this file, and controls properties of X applications that are run during your session. A full discussion of xrdb is beyond the scope of this document, but you can do `man xrdb` to learn more about it on your own. A few sample lines from a .Xresources follow:

```
XTerm*background:                black       ! Black background
XTerm*foreground:                green       ! with Green text
XTerm*scrollBar:                 true        ! Use scrollbars
xclock.geometry:                 100x100+0-0 ! Set location and size
```

You may notice that this is the first of the dotfiles we have mentioned which is not simply a (t)csh script that gets run in a particular order. Also note that the capitalization of the program specification is very important (and confusing), so look back to the manpage for reference.

5.6 ~/.startup.X

This is the designated file to start X programs from, because at this point the window manager has been started, Xresources loaded, and all environment variables set. An important point to remember is to put a '&' after commands which don't return immediately, so that the command is started in the background, while the main process continues on. There are a wide variety of popular programs to start in the background so only a couple will be listed, namely:

```
xloadimage -fullscreen -onroot Graphics/background.gif > /dev/null &
xterm -name Main -geometry +300+300 &
xscreensaver -l &
```

Which respectively set the background of your screen to be the specified image, create an xterm with a specific name and location, and start the SIPB xscreensaver, so that you can click on its icon to lock your machine.

5.7 ~/.logout

This file is sourced by session_gate when you log off. Typically there isn't much to be done in this file, but there are some jobs it's good at. For example, if you want to record how long you're logged on every day, you can append the date into a file for later use. Or you can leave friends a more accurate version of lastlog,

[10] If you ever have reason to run csh, it will also source this file, which may cause problems, if you use tcsh specific features.

by appending the date into a file into your Public directory. If you've changed the default location of your kerberos tickets, your ~/.logout is a good place to insure that they are destroyed when you log off (kdestroy will get run by /etc/athena/login/Xsession, so unless you have extra sets of ticket files, you don't need to worry about this).

6 TTY logins

Until this point, this document has covered what happens when you log in via X. However, during TTY logins, a slightly different sequence of files are sourced. A TTY login occurs when you telnet, ssh or rlogin to a machine, or if you choose the *Terminal-style session* option from xlogin.

Whichever program handles your login will startup a copy of your default shell, with the `-l` option, signifying that it should be a login shell[11]. This shell will then run ~/.cshrc[12] (Section 4.3, which will follow the same path as in a X login. When this is done, the shell will then source ~/.login (see below). When the login shell finally exits, it will source ~/.logout (Section 5.7).

6.1 ~/.login

This file by default simply sources /usr/athena/lib/init/login then returns. This is a good place to make modifications of your own for TTY logins, but the same warnings apply as to ~/.cshrc; if you modify this file, you have to make sure to keep it up to date with Athena releases.

6.2 /usr/athena/lib/init/login

This script first checks for the existence of a ~/.hushlogin, then it sets terminal characteristics appropriate to the particular machine type. Then, if the skip_tty_startup shell variable isn't set, login will run the following sequence of commands

```
zwgc                            # Start Zephyr client
from -t -n                      # Check for mail
(olc who &) >& /dev/null        # Ping the OLC server
```

Finally, unless the skip_lert shell variable is set, login runs `lert -q`, which shows any Athena Accounts messages to you (such as account deactivation warnings).

6.3 ~/.startup.tty

This is the equivalent of ~/.startup.X, but is only run when you log in on a TTY session. Traditionally, less is done in this file than in ~/.startup.X, due to the single screen nature of a TTY login.

7 Customizing your login

Up to this point, all discussion has assumed no modifications to the default dotfiles (aside from the possible addition of some files).

7.1 Things to remember

Remember, messing with your dotfiles can be hazardous to your ability to log in :-) However, at any point you can still log in by choosing the *Ignore Customizations* option at the xlogin. This will ignore any changes that you have made to your dotfiles, and log you in safely. And if things go really bad, you can delete the copies of your dotfiles that you have been modifying and get new copies of the defaults from /usr/prototype_user

[11] The login shell on a TTY session is equivalent to the session gate under X, so that if you exit it the login shell, you will be logged out.

[12] Unless you aren't a (t)csh user, of course.

(for .cshrc, .login, and .mh_profile), /usr/athena/lib/X11 (for the system wide .mwmrc, which will be called system.mwmrc), or /usr/athena/lib/zephyr (for zwgc.desc). Finally, you can and should test all dotfiles you are modifying by either running or sourcing them while still logged in.

7.2 Normal Customizations

The best way to customize your account is to create and modify the ~/.cshrc.mine, ~/.environment, ~/.Xresources, and ~/.startup.X files mentioned above.

A modification which makes a big difference to the look of your login session is the Xresources that many of the programs you run use. If you have the option of modifying an attribute such as background color on the command line, every time you start the program, or setting an Xresource, it's probably easiest to set the Xresource. If you decided not to run dash because it has an ugly color, first, try experimenting with different color choices. For example, you might put the following in your ~/.Xresources:

```
Dash*background: grey20
Dash*foreground: yellow
```

Once you start looking around, you will be surprised at what modifications many X applications will allow.

From an aesthetic stand point, your choice and configuration of window manager is also important (Section A.4).

Another area of common modification includes desktop toys. Some common ones are xclock, xload, xcalc, xzul (in the outland locker), and xzewd (also in the outland locker). Any other X applications that you find yourself using every time you login are also candidates for being started out of your ~/.startup.X. Setting a geometry resource will save you having to place each application that you start up, which is especially important if you start up them up in your ~/.startup.X.

7.3 Quickstation Customizations

Unless it is carefully done, a heavily customized login is often a slow login. For this reason, some people prefer to make a different set of customizations get executed when they login to a quickstation. To make this possible, the environment variable `ATHENA_QUICK` is automatically set in the default dotfiles. Here's an example of how you might use this in your ~/.startup.X.

```
if(! $?ATHENA_QUICK) then
  add infoagents sipb
  xzewd &
  netscape &
endif
```

7.4 SGI Differences

When you login on an SGI, the default sequence of dotfiles differs slightly from that on other platforms. If you want your login on SGIs to look just like your login on other platforms, you can put `set skip_sgi` in your ~/.environment. However, there are reasons why you might want to keep the customizations separate.

The main differences in the login sequence are that your ~/.startup.sgi (if it exists) will get sourced instead of your ~/.startup.X, and that the windowmanager 4dwm will get run instead of mwm (or whatever windowmanager you have set in your ~/.environment).

If you want to customize your SGI login separately from your normal login, it's likely that you'll have to do very little work. The window manager 4dwm will create a directory[13] ~/.desktop_indy or ~/.desktop_O2, containing its configuration files, and saving information about what programs you were running, and where they were on the desktop. The next time it starts up, it will launch all the programs that were running when it was last shut down. This means that you don't want to start up programs in your ~/.startup.sgi, since unless you remember to exit each of them before you logout, you will end up getting several copies started, each time you login.

[13] The files will actually be of the form ~/.desktop-*hostname*, but they will be symlinks to ~/.desktop_indy or ~/.desktop_O2.

7.5 Using Other Shells

If you are used to a shell other than tcsh, on other systems, it may make sense to configure your Athena account to use that shell also. In particular, with the 8.3 Athena release, bash will also be supported, and all new accounts will start with a set of dotfiles for bash, as well as those for tcsh. If you currently use tcsh, and wish to switch to bash, you will need to copy in a .bashrc and .bash_login from /usr/prototype_user, then run chsh to change the information about your default shell in moira.

If you are running bash as your shell, your ~/.cshrc will never get sourced, but your ~/.bashrc will, and your ~/.bash_environment will get sourced instead of your ~/.environment. Your ~/.startup.X or ~/.startup.tty will get sourced, as appropriate. Since the syntax of bash differs from that of tcsh, if you already have customizations, you might need to rewrite them. For information on bash, `man bash` or read the node in the info database[14].

8 Conclusion

We hope that this document has taught you a bit about dotfiles, and given you some ideas for customizing your own.

[14]To do this, you can run `info -f bash`, or from within emacs, you can do `C-h i m bash <RET>`.

A Other Dotfiles

The files listed above are the ones used during every login. There is another set of dotfiles, which exist in almost all user's home directories, but which pertain more to the execution of various programs than to the actual login process. It is typical for many programs to use such dotfiles to store user configuration between sessions, and some users have several hundred such dotfiles. Less time will be spent on these than those above, but a few are worth mentioning due to their ubiquitous nature.

A.1 ~/.emacs

Emacs reads and loads this file every time it is started. This file is composed of emacs lisp commands, the native language of gnu emacs. The capabilities of emacs are well beyond the scope of this document, and a .emacs may invoke all of these capabilities. Several examples of .emacs files may be found in the dotfiles locker under the Emacs directory. Help with emacs may be found by doing C-h i inside an emacs window, and selecting the emacs menu option.

A.2 ~/.meetings, ~/.newsrc

These files are used by discuss and newsreader programs to store the list of meetings or groups which the user wishes to read. The SIPB documents Using Discuss and One Page NetNews[15] will get you started with discuss and netnews.

A.3 ~/.mh_profile

This is the default configuration file for MH, which is the set of programs (including inc and comp) used for handling mail. For more information on this file, read its manual pages with the commands `man mh-profile` and `man mh`.

A.4 ~/.mwmrc, ~/.twmrc, etc.

These are configuration files for Motif Window Manager (mwm), Tab Window Manager (twm) respectively, although several other window managers, such as ctwm and vtwm.gamma will use ~/.twmrc. Each window manager has its own configuration file which determines how windows on your screen are manipulated, how the title bars and borders look, what menus are available, etc. To use a window manager which isn't the default, you should set the environment variable in your ~/.environment. For example, to use the ctwm window manager, you would need to do:

```
add windowmanagers
setenv WINDOW_MANAGER ctwm
```

Some examples of customized window manager dotfiles can be found in the dotfiles locker[16]. More information can also be found by reading the manual page of the window manager in question.

A.5 ~/.plan

On many systems, it's traditional to leave your current plans, information about how to get in touch with you, pgp public key, or anything that you want to tell lots of people in your ~/.plan. Other people can then read your ~/.plan by fingering you on a finger server which will display your ~/.plan. Because of how afs file permissions work, unless you have made the top level of your home directory world readable, you will need to put your ~/.plan in your Public directory, then symlink it to the top level of your home directory. To see your ~/.plan, or someone else's, `finger username@monk.mit.edu`. The Electric Monk is a public

[15] You can get these from /mit/sipb/doc, `http://www.mit.edu/sipb/docs.html`, or by asking at the SIPB office.

[16] The dotfiles locker is a repository for various dotfiles so that people can learn by example. Caution is advised, if you intend to use them in their entierty, since they are not guaranteed to be up to date.

finger service which serves up finger information for the MIT community. There is also a discuss meeting which archives an assortment of ~/.plan files[17], so be aware that your ~/.plan could get archived there.

A.6 ~/.zwgc.desc, ~/.zephyr.subs, ~/.zephyr.vars

These are configuration files for zephyr programs. ~/.zwgc.desc is used by zwgc to describes how windowgrams will appear on your screen, while ~/.zephyr.subs determines what instances and classes you are subscribed to. ~/.zephyr.vars is used to set zephyr signatures, visibility, and other zephyr variables. All of these files are covered in Inessential Zephyr[18].

[17]Point your discuss reader at `bloom-picayune.mit.edu:/usr/spool/discuss/ca`.

[18]Inessential Zephyr is published by The SIPB. You can find it at /mit/sipb/doc/izephyr.dvi, or you can pick up a copy at the SIPB office.