Joanna Klukowska
joannakl@cs.nyu.edu

Recitation 1: The Basics

Contents

1 Eclipse and Ocaml Installations 1

2 Using Command Line 3

3 Accessing Your Linux/Unix Account From Other Locations 6

1 Eclipse and Ocaml Installations

Note 1: When you "install" Eclipse, you download either a zip file or tar.gz file (depending on the operating system) that contains all files ready to run your the IDE. You only need to extract it and move it to a place where you and your OS can find it easily.

Note 2: Most modern operating systems will include Java Development Kit. You should also have it installed back from when you took CSCI-UA-101. If you are missing it for whatever reason, make sure you install it. Post questions to the course Forums on NYU Classes if you need help with that.

Windows Instructions:

Eclipse:

1. Download Eclipse from http://www.eclipse.org/downloads/.
2. Extract the zip file and place it in `C:\Program Files\`
3. Make a shortcut to `C:\Program Files\eclipse\eclipse` (note that the first eclipse is the directory, second is the name of the executable file) in your `C:\Documents and Settings\USERNAME\Start Menu\Programs` (in which `USERNAME` is your username on the Windows system) so that eclipse appears in your Start menu.

Objective Caml:

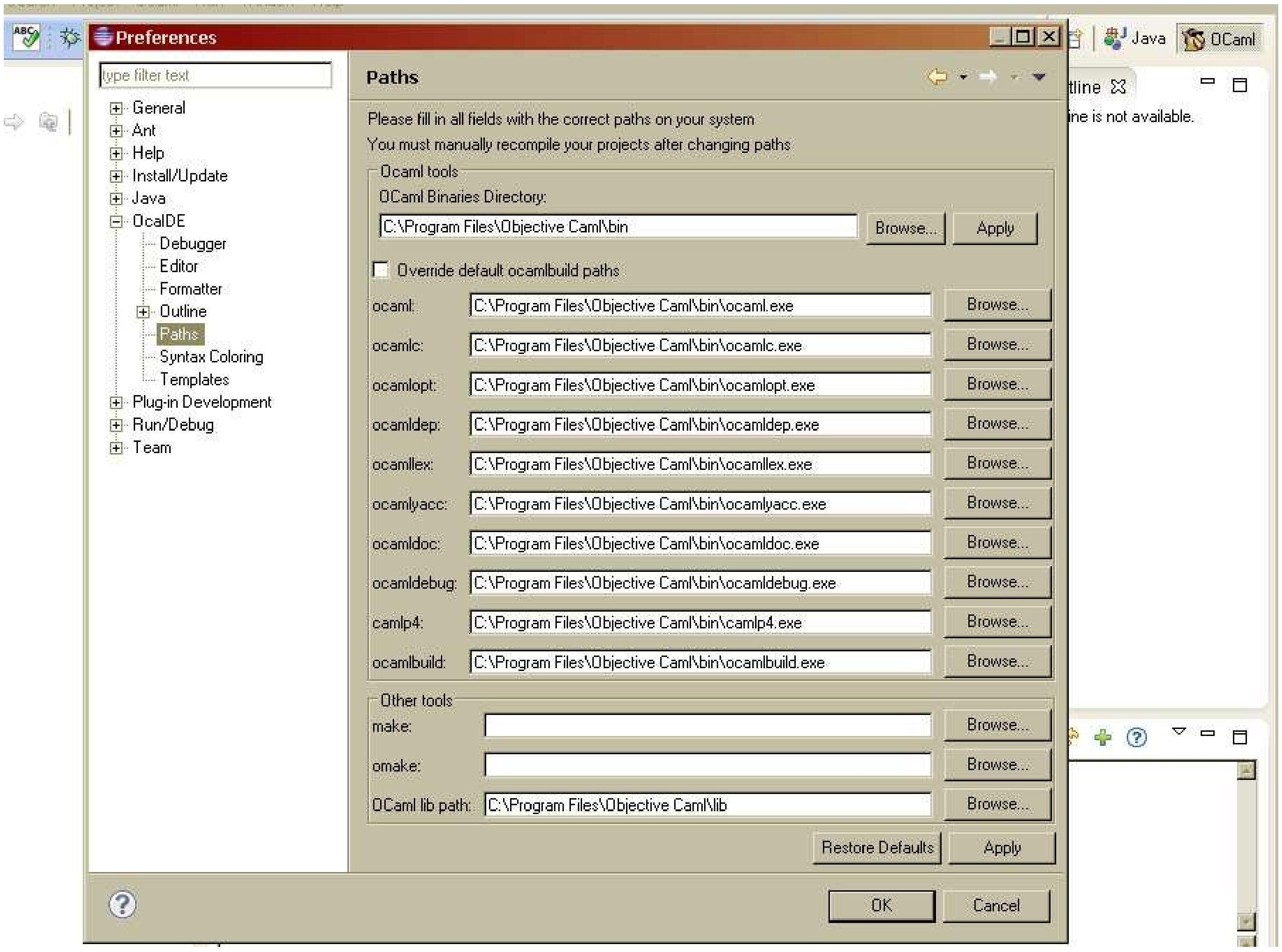
1. Download the http://caml.inria.fr/pub/distrib/ocaml-3.11/ocaml-3.11.0-win-msvc.exe file.
2. Run the file and accept all the defaults.
3. You can either open a terminal-like interface using the shortcuts in your Start menu or run `ocaml` in the terminal.

OcaIDE:

Do not attempt the OcaIDE installation unless both of the above installations succeeded.

Follow instructions in step 3 on http://www.seas.upenn.edu/∼cis120/current/ocaml_setup.shtml Assuming that Objective Caml has been installed in the default location, the paths should be filled in. See the figure below for an example. WARNING: do not just copy the paths from the figure into your setup without making sure that these files really exist on your system.

Joanna Klukowska
joannakl@cs.nyu.edu



Paths

Please fill in all fields with the correct paths on your system
You must manually recompile your projects after changing paths

Ocaml tools

OCaml Binaries Directory: C:\Program Files\Objective Caml\bin

[ ] Override default ocamlbuild paths

ocaml:	C:\Program Files\Objective Caml\bin\ocaml.exe
ocamlc:	C:\Program Files\Objective Caml\bin\ocamlc.exe
ocamlopt:	C:\Program Files\Objective Caml\bin\ocamlopt.exe
ocamldep:	C:\Program Files\Objective Caml\bin\ocamldep.exe
ocamllex:	C:\Program Files\Objective Caml\bin\ocamllex.exe
ocamlyacc:	C:\Program Files\Objective Caml\bin\ocamlyacc.exe
ocamldoc:	C:\Program Files\Objective Caml\bin\ocamldoc.exe
ocamldebug:	C:\Program Files\Objective Caml\bin\ocamldebug.exe
camlp4:	C:\Program Files\Objective Caml\bin\camlp4.exe
ocamlbuild:	C:\Program Files\Objective Caml\bin\ocamlbuild.exe

Other tools

make:	
omake:	
OCaml lib path:	C:\Program Files\Objective Caml\lib

Figure 1: Paths listing for OcaIDE in Eclipse on a Windows system.

Mac Instructions:

Eclipse:

1. Download Eclipse from http://www.eclipse.org/downloads/.
2. Double click on the tar.gz file. This will extract the eclipse from the compressed file. You might want to move the extracted eclipse folder to your Applications folder for easy access.
3. Double click the Eclipse executable in the eclipse folder to run the IDE.

Objective Caml:

1. Download the dmg file
2. Open the dmg file. The `Readme.txt` file has some basic instructions and information. Make sure you read it.

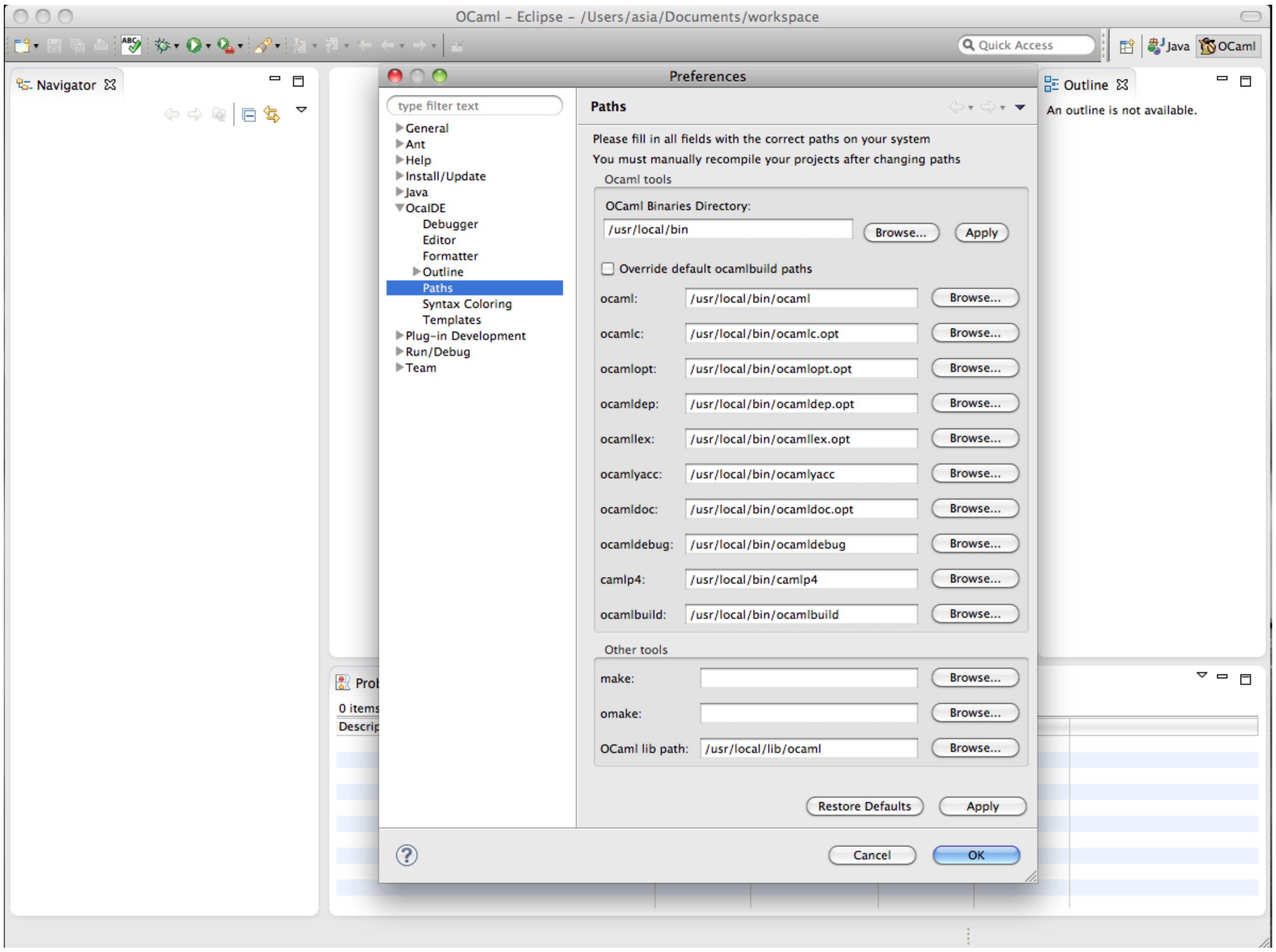
Joanna Klukowska
joannakl@cs.nyu.edu

3. Double click on `ocaml.pkg` to start the installation. Accept all the defaults.
4. Make sure that ocaml is installed by running the command `ocaml` in the terminal window.

OcaIDE:

Do not attempt the OcaIDE installation unless both of the above installations succeeded.

Follow instructions in step 3 on http://www.seas.upenn.edu/∼cis120/current/ocaml_setup.shtml Assuming that Objective Caml has been installed in the default location, the paths should be filled in. See the figure below for an example. WARNING: do not just copy the paths from the figure into your setup without making sure that these files really exist on your system.



Please fill in all fields with the correct paths on your system
You must manually recompile your projects after changing paths

Ocaml tools

OCaml Binaries Directory: /usr/local/bin

Override default ocamlbuild paths

ocaml:	/usr/local/bin/ocaml
ocamlc:	/usr/local/bin/ocamlc.opt
ocamlopt:	/usr/local/bin/ocamlopt.opt
ocamldep:	/usr/local/bin/ocamldep.opt
ocamllex:	/usr/local/bin/ocamllex.opt
ocamlyacc:	/usr/local/bin/ocamlyacc
ocamldoc:	/usr/local/bin/ocamldoc.opt
ocamldebug:	/usr/local/bin/ocamldebug
camlp4:	/usr/local/bin/camlp4
ocamlbuild:	/usr/local/bin/ocamlbuild

Other tools

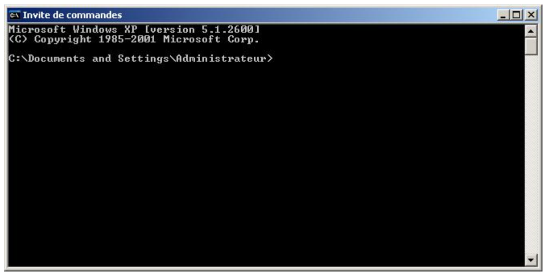
make:	
omake:	
OCaml lib path:	/usr/local/lib/ocaml

Figure 2: Paths listing for OcaIDE in Eclipse on a Mac system.

2 Using Command Line

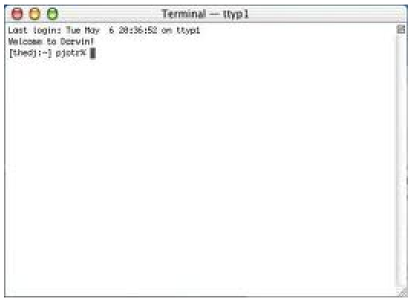
You should know how to compile and run your Java and Ocaml programs from the command line, i.e. the small black or white window with nothing but the text of a prompt in it that looks something like this:

Joanna Klukowska
joannakl@cs.nyu.edu



```
Microsoft Windows XP [version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrateur>
```



```
Last login: Tue May  6 20:36:52 on ttyp1
Welcome to Darwin!
[thed]:~] pjotr%
```

When you log in to a computer system remotely, very often this is the only way of interacting with the computer.

Assuming you have the JDK and Ocaml installed, the commands that you need to run are pretty much the same for Windows, Mac and Linux/Unix systems. But the way you are ”getting around”, i.e. moving from one directory to another, in the command line is different in Windows than it in Mac/Linux/Unix. Since you will need to login to a Linux system in order to test your assignments, the rest of this section assumes a Linux like environment.

Finding your way around Unix/Linux

Note: These notes refer to the names of directories and machines that I used to use in a different school. Once we have the accounts ready on the NYU system, I will revise these notes accordingly.

Whenever you are logged into a Unix/Linux system, you have a unique, special directory called your **current** or **present working directory** (PWD for short). The present working directory is where you "are" in the file system at the moment, i.e., the directory that you feel like you are currently "in". This is more intuitive when you are working with the command line, but it carries over to the graphical user interface (GUI) as well.

Many commands operate on the PWD if you do not give them an argument. We say that the PWD is their default argument. (Defaults are fall-backs – what happens when you don’t do something.) For example, when you enter the "`ls`" command (list files) without an argument, it displays the contents of your PWD. The dot "`.`" is the name of the PWD:

```
ls .
```

and

```
ls
```

both display the files in the PWD, first one because the name of the directory is provided, second one because it is the default behavior for `ls`.

When you first login, your present working directory is set to your **home directory**. Your home directory is a directory created for you by the system administrator. It is the top level of the part of the file system that belongs to you. You can create files and directories within your home directory, but usually nowhere else. Usually your home directory’s name is the same as your username.

In Unix/Linux, files are referred to by **pathnames**. A pathname is like a generalization of the file’s name. A pathname specifies the location of the file in the file system by its position in the hierarchy. In Unix/Linux, a forward slash "/" separates successive directory and file names in the pathname, as in: jklukow/cs135_jk/cs136_labs/lab01_tutorial.pdf

There are two kinds of pathnames: absolute pathnames, and relative pathnames. The absolute pathname is the unique name for the file that starts at the root of the file system. It always starts with a forward slash.

A relative pathname is a pathname to a file relative to the PWD. It never starts with a slash! If `afuller` has a directory called `cs136` in her home direcory which contains another directory called `lab01` which contains the file called `lab01_solution.pdf`, then the relative pathname of that file from PWD is

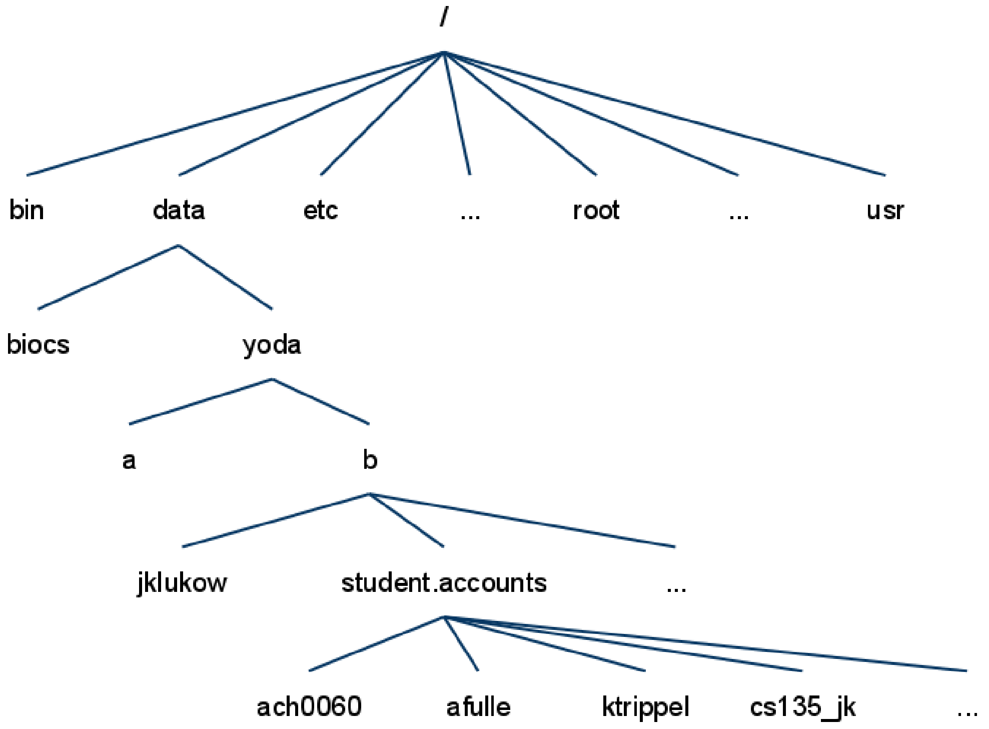


Figure 3: Partial directory structure on *eniac* - this is not the system you will be using in NYU.

```
cs136/lab01/lab01_solution.pdf
```

and it's absolute pathname is

```
/data/yoda/b/student.accounts/afuller/cs136/lab01/lab01_solution.pdf
```

Dot-dot or **..** is a shorthand name for the parent directory. The parent directory is the one in which the directory is contained. It is the one that is one level up in the file hierarchy. If my PWD is currently `/data/yoda/b/jklukow/cs136/`, then `..` is the directory `/data/yoda/b/jklukow/` and `../../` is the directory `/data/yoda/b/`.

Unix/Linux commands to get you started

`ls`	list content of the current working directory
`ls dir_name`	list content of the directory named `dir_name`
`cd dir_name`	`cd` stands for change directory, changes the current working directory to `dir_name`
`cd ..`	move one directory up in the directory tree
`cd`	change the current working directory to your home directory
`pwd`	print the name of the present working directory
`cp file1 file2`	copy `file1` into `file2`, where `file1` and `file2` can be either relative or complete path names
`mv file1 file2`	move `file1` into `file2`, where `file1` and `file2` can be either relative or complete path names

`rm file`	remove a file (there is no undoing it, so be very careful!)
`mkdir path`	make a directory at the specified `path`
`rmdir path`	remove the directory specified by the `path` (there is no undoing it, so be very careful!)
`man command`	display a manual page (or simply *help*) for the `command` (this is the easiest way to learn about options to the commands that you know and about new commands).

How to Compile Your Java Programs Using Command Line

One of the things that you going to do is compiling your code. We will use one of the JDK tools called `javac`. Simply type javac followed my the name of the file with your java source code at the prompt:

```
javac MyFirstCommandLineProgram.java
```

This assumes that your PWD contains the source code file. If it does not use `cd` to navigate to the appropriate directory. Once you successfully compile the code, you will have `MyFirstCommandLineProgram.class` in the same directory. To run it, use the Java interpreter called `java` followed by the name of the class you are running (without the `.class` extension) as follows:

```
java MyFirstCommandLineProgram
```

How to Use Ocaml in Command Line

When you run the command ocaml from your command line, you will be taken to an ocaml interpreter prompt. You can now proceed entering command one at a time. The problem with that approach is that you will not be able to edit code that you entered on previous line. A more programmer friendly approach is to type the program in a text file (use your favorite text editor, just make sure it produces plain text) and then run the program from such file. This is done using the #use directive followed by the name of the source code file in double quotes followed by two semicolons:

```
[asia@asia rec1]$ ocaml
Objective Caml version 3.11.2
# #use "hello_world.ml" ;;
hello world
- :  unit = ()
#
```

3 Accessing Your Linux/Unix Account From Other Locations

To access you Linux account at NYU you will need an SSH and SFTP clients. The following is a list of just a few free packages that you can use:

- PUTTY, Windows, SSH client
- FileZilla, cross-platform, SFTP client
- Fugu, Mac, SFTP client
- Apple systems should come with SSH client built in, you may need to enable it. Once this is done, you simply type

  ```
  ssh USERNAME@NYU_LINUX_SERVER_NAME
  ```

- Linux systems come with SSH client built in, and many SFTP clients (if you need some names, let me know).