

CISC 1115 MY9F

Programming Environment

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

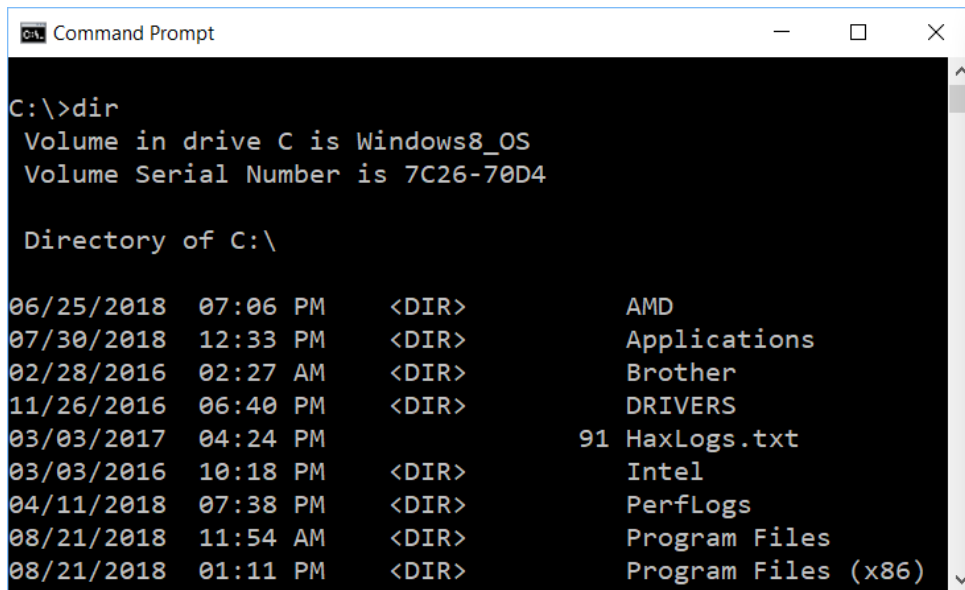
- Get around in Operating Systems
 - Unix-like: Unix, Linux, OS X
 - Windows
- Terminal and command line
- JRE and JDK
- Authoring Java programs
- Compiling and running Java programs

Operating Systems

- Do you know the answers to the questions:
 - Where are my files?
 - What is a folder or a directory?
 - How do I install an computer application?
 - How do I open a terminal window?
 - How do I copy/delete/rename a file, or a folder/directory. How about several files or folders/directories?
- To learn programming, it is necessary to be proficient in using your computer. Minimally, you need to know the answers to the questions like these.
 - If not yet, no sweat, we will get there this semester.

Terminal and Command Line

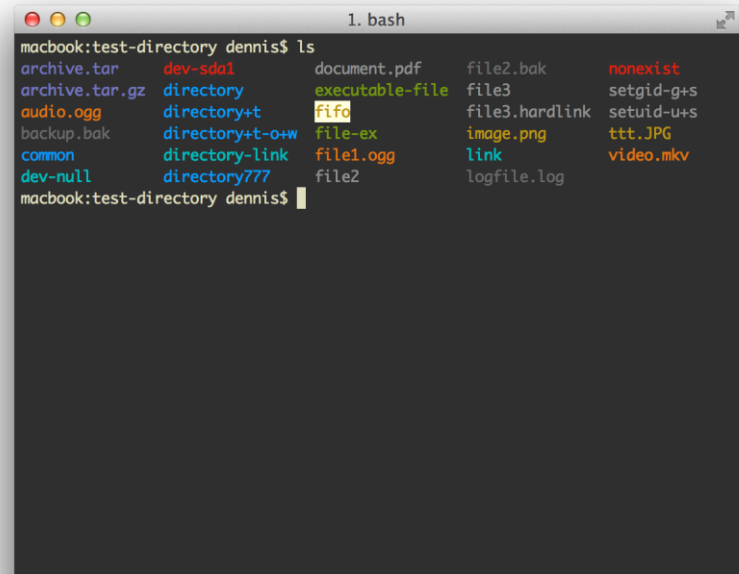
- Use a Command Line
- Why?



```
C:\>dir
Volume in drive C is Windows8_OS
Volume Serial Number is 7C26-70D4

Directory of C:\

06/25/2018  07:06 PM    <DIR>          AMD
07/30/2018  12:33 PM    <DIR>          Applications
02/28/2016  02:27 AM    <DIR>          Brother
11/26/2016  06:40 PM    <DIR>          DRIVERS
03/03/2017  04:24 PM                91 HaxLogs.txt
03/03/2016  10:18 PM    <DIR>          Intel
04/11/2018  07:38 PM    <DIR>          PerfLogs
08/21/2018  11:54 AM    <DIR>          Program Files
08/21/2018  01:11 PM    <DIR>          Program Files (x86)
```



```
macbook:test-directory dennis$ ls
archive.tar      dev-sda1      document.pdf   file2.bak      nonexistent
archive.tar.gz  directory     executable-file file3           setgid-g+s
audio.ogg        directory+t   fifo          file3.hardlink setuid-u+s
backup.bak       directory+t+o+w file-ex       image.png     ttt.JPG
common           directory-link file1.ogg     link          video.mkv
dev-null         directory777  file2        logfile.log
macbook:test-directory dennis$
```

Common Tasks on Command Line

Windows

- Display working directory: `cd`
- Display content of a directory: `dir`
- Change directory: `cd directory_to_go`
- Make directory: `mkdir dir_to_make`
- Move files: `move this_file to_dir`
- Delete file: `del file_to_delete`
- Delete directory: `rmdir dir_to_delete`

Unix-like

- Display working directory: `pwd`
- Display content of a directory: `ls`
- Change directory: `cd directory_to_go`
- Make directory: `mkdir dir_to_make`
- Move files: `move this_file to_dir`
- Delete file: `rm file_to_delete`
- Delete directory: `rmdir dir_to_delete`

JRE and JDK

- JRE: Java Runtime Environment
 - Required to run a Java program
- JDK: Java Development Kit
 - Required to author a Java program
 - Compile and run Java programs
- Have you had JDK installed/set up in your system?

Oracle JDK vs Open JDK

- Java
 - A specification. There are more than one implementations and packaging.
 - <https://docs.oracle.com/javase/specs/index.html>
- Oracle JDK
 - <https://www.oracle.com/java/technologies/javase-downloads.html>
- Open JDK
 - <https://openjdk.java.net/>

Verify JRE is Present

- On Command Line
 - `java -version`

```
Command Prompt
C:\>java -version
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
C:\>
```



```
Command Prompt
C:\>jave -version
'jave' is not recognized as an internal or external command,
operable program or batch file.
C:\>
```



Verify JDK is Present

- On Command Line
 - javac -version

```
Command Prompt
C:\>javac -version
javac 1.8.0_251
C:\>_
```



```
Command Prompt
C:\>javac -version
'javac' is not recognized as an internal or external command,
operable program or batch file.
C:\>_
```



Verify Versions of JVM and Java Compiler

- On command line

```
Command Prompt
C:\Users\hui>java -version
java version "1.8.0_301"
Java(TM) SE Runtime Environment (build 1.8.0_301-b09)
Java HotSpot(TM) 64-Bit Server VM (build 25.301-b09, mixed mode)

C:\Users\hui>javac -version
javac 14.0.2

C:\Users\hui>_
```



```
Command Prompt
C:\Applications\Java>java -version
openjdk version "16.0.2" 2021-07-20
OpenJDK Runtime Environment (build 16.0.2+7-67)
OpenJDK 64-Bit Server VM (build 16.0.2+7-67, mixed mode, sharing)

C:\Applications\Java>javac -version
javac 16.0.2

C:\Applications\Java>_
```



Trouble with JDK?

- “I have installed JDK, but still I got this!”

```
Command Prompt
C:\>javac -version
'javac' is not recognized as an internal or external command,
operable program or batch file.
C:\>_
```



- You need to set up the search path for JDK’s executables (such as, javac)
 - System specific, but we will do it via the user profile of “Git Bash”

Your Exercise

- Let's complete the following tasks
 - Open a terminal
 - Show working directory
 - Display content in the working directory
 - Switch to a different directory
 - Create a directory
 - Delete the directory
 - Verify if JRE is present
 - Verify if JDK is present (if inaccessible on command line, it is OK for this exercise; we shall address this in a little while)
 - Write a short manual about the steps in the journal in directory C0826

Get Organized!

1. Create a folder (i.e., directory) called “journal” somewhere on your computer
2. In the “journal” directory, create a subfolder (i.e., subdirectory) with a meaningful name, like C0831
3. Go to the directory (“cd C0831”)
4. Do something, e.g., create a batch file (on Windows systems)/shell script (on UNIX systems)

- Windows

```
echo java -version > checkjava.cmd
```

```
echo javac -version > checkjavac.cmd
```

- UNIX

```
echo “java -version” > checkjava.sh
```

```
echo “javac -version” > checkjavac.sh
```

Questions?

- Do you have any questions?
- Is there any tasks you wish to do, but not introduced in the class?

Additional Resources

- Several good tutorials
 - <https://www.sci.brooklyn.cuny.edu/~goetz/java/>
- Online IDEs, e.g.,
 - <https://replit.com/>

Questions?

- For CISC 1115, you are allowed to use IDEs.
- What is your preference?

Authoring a Java Program

1. **Requirement:** write a shortest java program, and compile and run it.
2. **Design:** a Java program that prints out “Hello, World!” on the standard output
3. **Implement**
 - A. Create/Revise a HelloWorld.java using an editor
 - A. Using: the Atom editor, the Visual Studio Code, notepad++ for Windows; SlickEdit (\$\$\$) for Mac OS X, ...
 - B. The instructor will use Atom for demo in class.
 - B. Compile the program, if error, go to step A
4. **Test**
 - Test the program, if failed, go to step 2 (can also be steps 1 and 3)

Demo for Authoring a Java Program

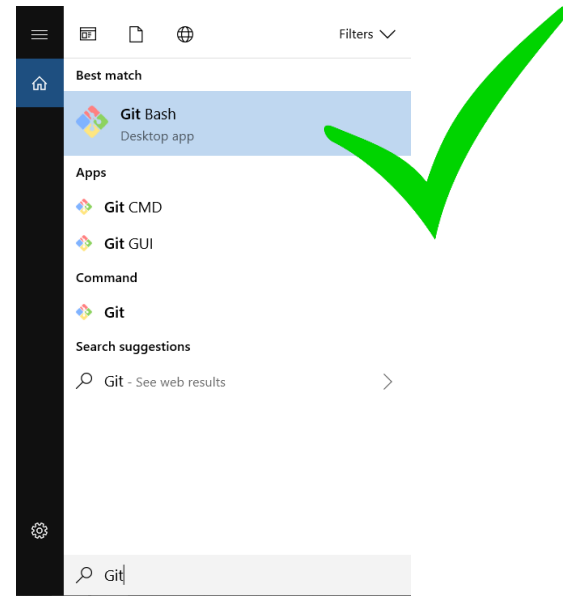
1. Prepare the working environment
 - a) Install the git client (if not already installed)
 - b) Install the Atom editor (if not already installed)
2. Create HelloWorld.java using the Atom editor
3. Compile and run the program

Prepare the Working Environment

1. Install the git client (if not already installed)
2. Install the Atom editor (if not already installed)

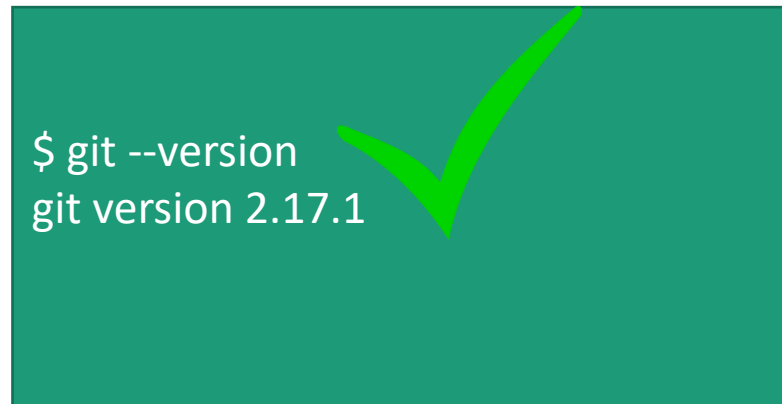
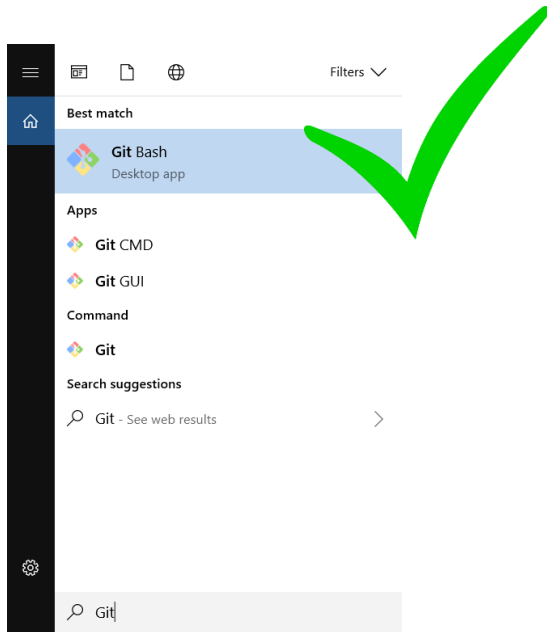
Verify Whether You Have Git Client

- Verify if you have had the Git client installed already
- Windows
 - Attempt to run “Git Bash”
- Unix (OS X or Linux):
 - Open a terminal window
 - Run “git --version”, i.e., type “git --version” (without quotes) and hit the Enter key



Have I Had Git Client Installed?

- Windows and Unix



- If not, download and install it

Download Git Client

- Visit <https://git-scm.com/downloads> using your favorite Web browser

The image shows a screenshot of the Git Downloads page. The top section is titled "Downloads" and features three platform-specific download buttons: "Mac OS X", "Windows", and "Linux/Unix". A large green checkmark is placed over the "Windows" button. To the right of these buttons is a computer monitor displaying the "Latest source Release 2.18.0" and a "Download 2.18.0 for Windows" button, also marked with a green checkmark. Below the platform buttons, a note states: "Older releases are available and the Git source repository is on GitHub." The bottom section of the page is divided into two columns: "GUI Clients" and "Logos". The "GUI Clients" section mentions built-in tools like "git-gui" and "gitk", and includes a link "View GUI Clients →". The "Logos" section mentions various Git logos in PNG and EPS formats and includes a link "View Logos →". Both the "View GUI Clients" and "View Logos" links are crossed out with large red X's.

Downloads

Mac OS X Windows Linux/Unix

Latest source Release
2.18.0
Release Notes (2018-06-21)
Download 2.18.0 for Windows

Older releases are available and the Git source repository is on GitHub.

GUI Clients
Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.
[View GUI Clients →](#)

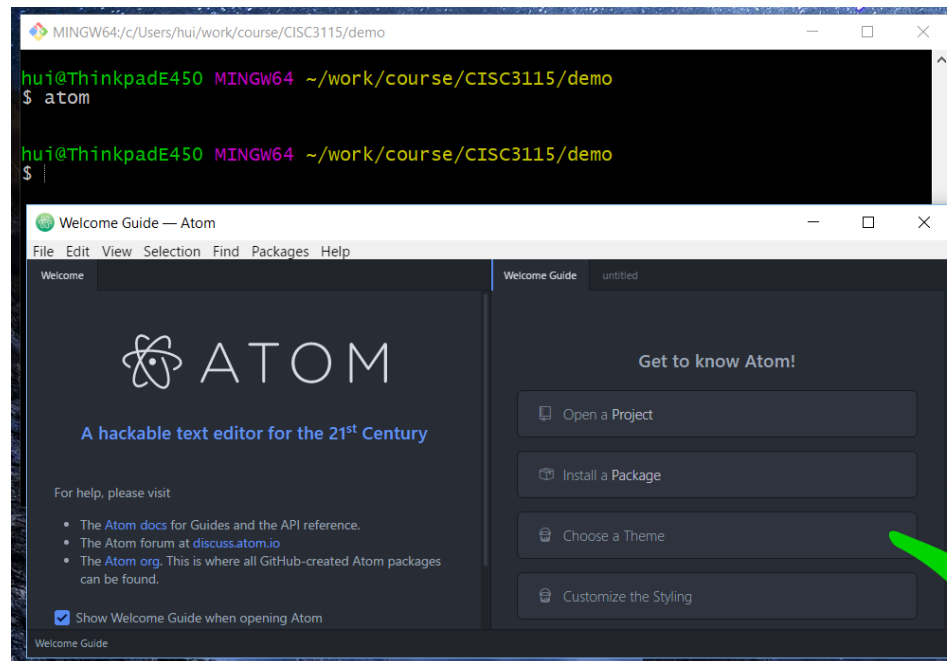
Logos
Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.
[View Logos →](#)

Git Bash on Windows

- Provides a terminal where you can run Unix commands
- The instructor shall use the Git Bash from now on so that the instructions are identical to both Windows and Unix (e.g., OS X) users
- Window users: Use the Git Bash terminal
- Unix users: just use a terminal (e.g., the terminal on OS X)

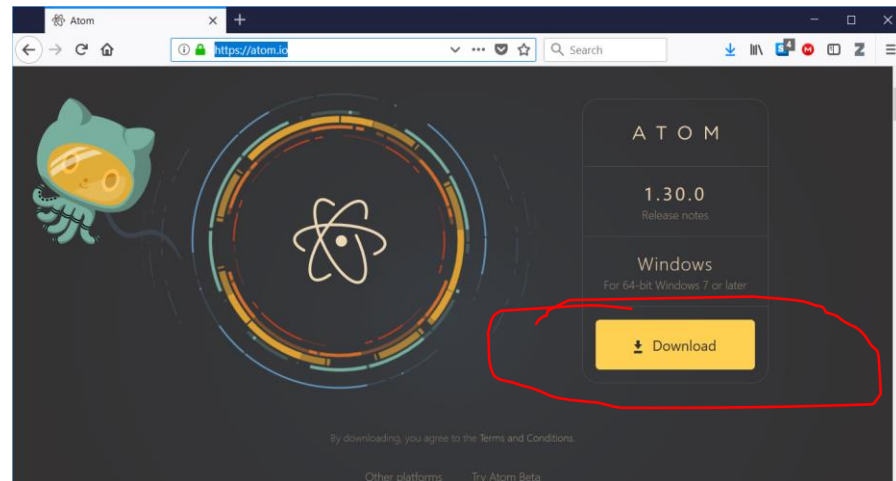
Verify Whether You Have Atom Installed

- Verify if you have had the Atom editor installed already
 - Type atom on the Command Line



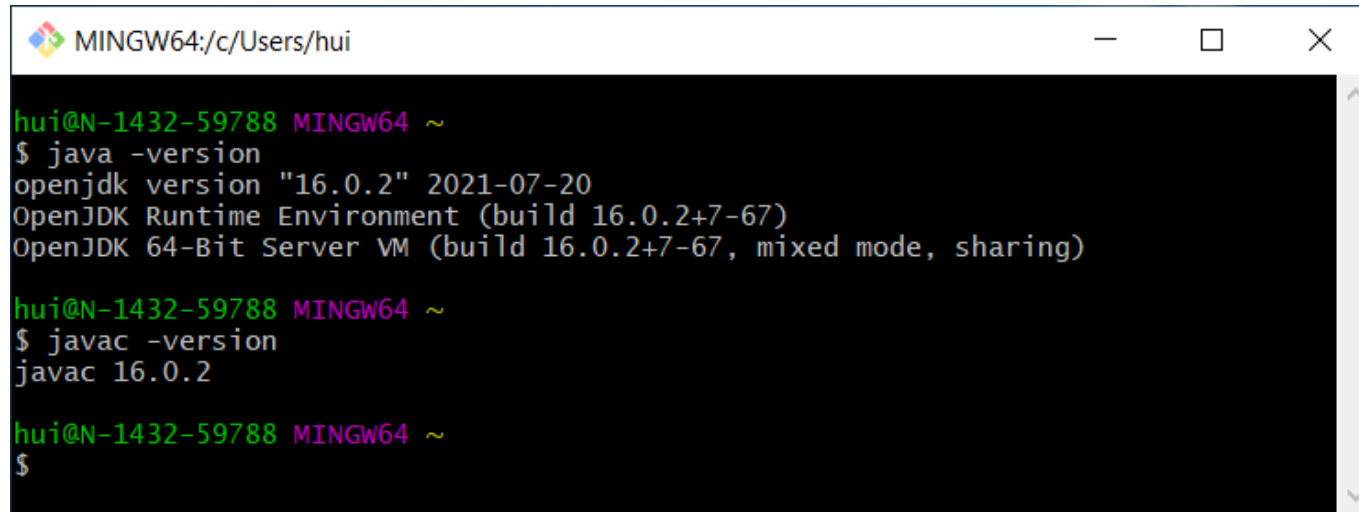
Download and Install the Atom Editor

- If you have not had the Atom Editor installed, download and install the Atom editor
- Visit <https://atom.io/> using your favorite Web browser



Checking on Java and Javac

- Check whether both java & javac are found, and have an identical version. Otherwise, next slide

A terminal window titled 'MINGW64:/c/Users/hui' with standard window controls. The terminal shows three commands and their outputs: 1. '\$ java -version' outputs 'openjdk version "16.0.2" 2021-07-20', 'OpenJDK Runtime Environment (build 16.0.2+7-67)', and 'OpenJDK 64-Bit Server VM (build 16.0.2+7-67, mixed mode, sharing)'. 2. '\$ javac -version' outputs 'javac 16.0.2'. 3. '\$' is entered on a new line, and the prompt '\$' appears again on the next line.

```
hui@N-1432-59788 MINGW64 ~
$ java -version
openjdk version "16.0.2" 2021-07-20
OpenJDK Runtime Environment (build 16.0.2+7-67)
OpenJDK 64-Bit Server VM (build 16.0.2+7-67, mixed mode, sharing)

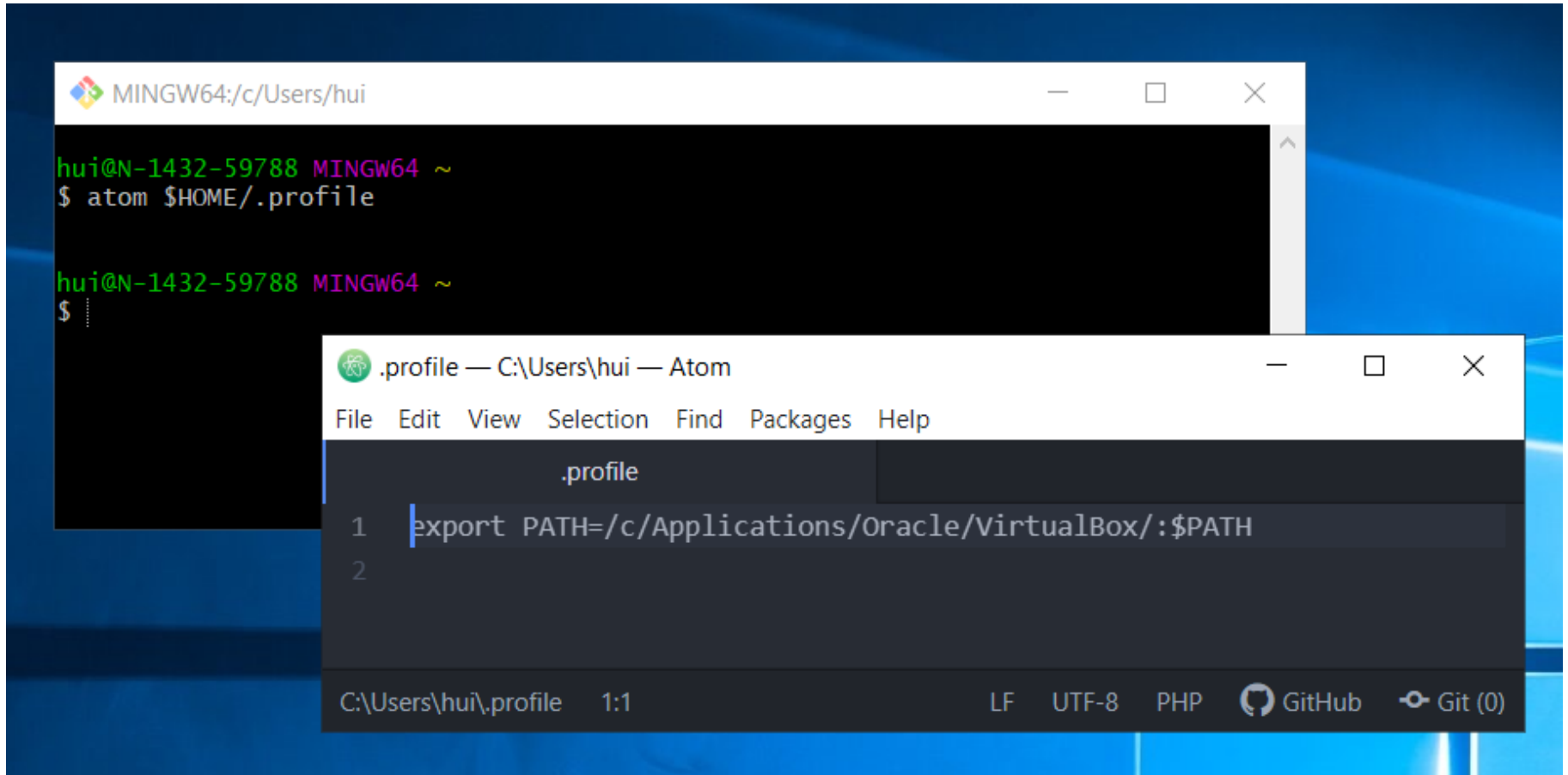
hui@N-1432-59788 MINGW64 ~
$ javac -version
javac 16.0.2

hui@N-1432-59788 MINGW64 ~
$
$
```

Setting up Search Path for Java and Javac

- In “Git Bash” terminal, create (if not already exists) or edit the .profile file on your “home directory” (see next slide)
- Then, restart “Git Bash” terminal, and check accessibility and versions of Java and Javac

Edit/Create .profile File



The image shows a Windows desktop environment with a blue background. In the foreground, there are two windows. The top window is a terminal window titled "MINGW64:/c/Users/hui". It shows the following commands and output:

```
hui@N-1432-59788 MINGW64 ~  
$ atom $HOME/.profile  
  
hui@N-1432-59788 MINGW64 ~  
$ .....
```

The bottom window is the Atom text editor, titled ".profile — C:\Users\hui — Atom". It shows the following content:

```
File Edit View Selection Find Packages Help  
  
.profile  
1 export PATH=/c/Applications/Oracle/VirtualBox/:$PATH  
2
```

The status bar at the bottom of the Atom window shows "C:\Users\hui\.profile 1:1", "LF UTF-8 PHP", and "GitHub Git (0)".

Implement the HelloWorld Java Program

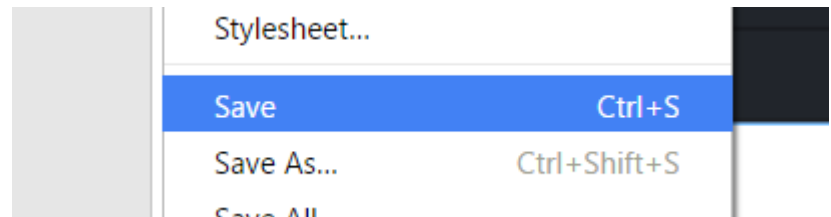
- Open a terminal Window
- (Optional) Create a subdirectory under a desired directory
- Run “atom HelloWorld.java” from the Command Line at the subdirectory
- Type the code
- Save the file

```
MINGW64:/c/Users/hui/CISC1115/examples
hui@N-1432-59788 MINGW64 ~
$ pwd
/c/Users/hui
hui@N-1432-59788 MINGW64 ~
$ mkdir -p CISC1115/examples
hui@N-1432-59788 MINGW64 ~
$ cd CISC1115/examples
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ ls
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ atom HelloWorld.java
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ |
```

```
HelloWorld.java — C:\Users\hui\CISC1115\examples — Atom
File Edit View Selection Find Packages Help
HelloWorld.java
1 public class HelloWorld {
2     public static void main(String[] args) {
3         System.out.println("Hello, World!");
4     }
5 }
```

C:\Users\hui\CISC1115\examples\HelloWorld.java* 3:41 CRLF UTF-8 Java GitHub Git (0)

- Press “CTRL-S” or click “Save” from the “File” menu to save the file



Compile and Run the Program

```
MINGW64/c/Users/hui/CISC1115/examples
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ ls
HelloWorld.java
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ javac HelloWorld.java
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ ls
HelloWorld.class HelloWorld.java
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ java HelloWorld
Hello, world!
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ |
```

Verify the program file exists

Compile the program

Verify the class file was created

Run the program

Did the program pass the test?

```
MINGW64 ~/Users/hui/CISC1115/examples
• Do I see "Hello World!" when I run the program?

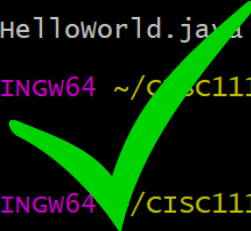
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ ls
HelloWorld.java

hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ javac HelloWorld.java

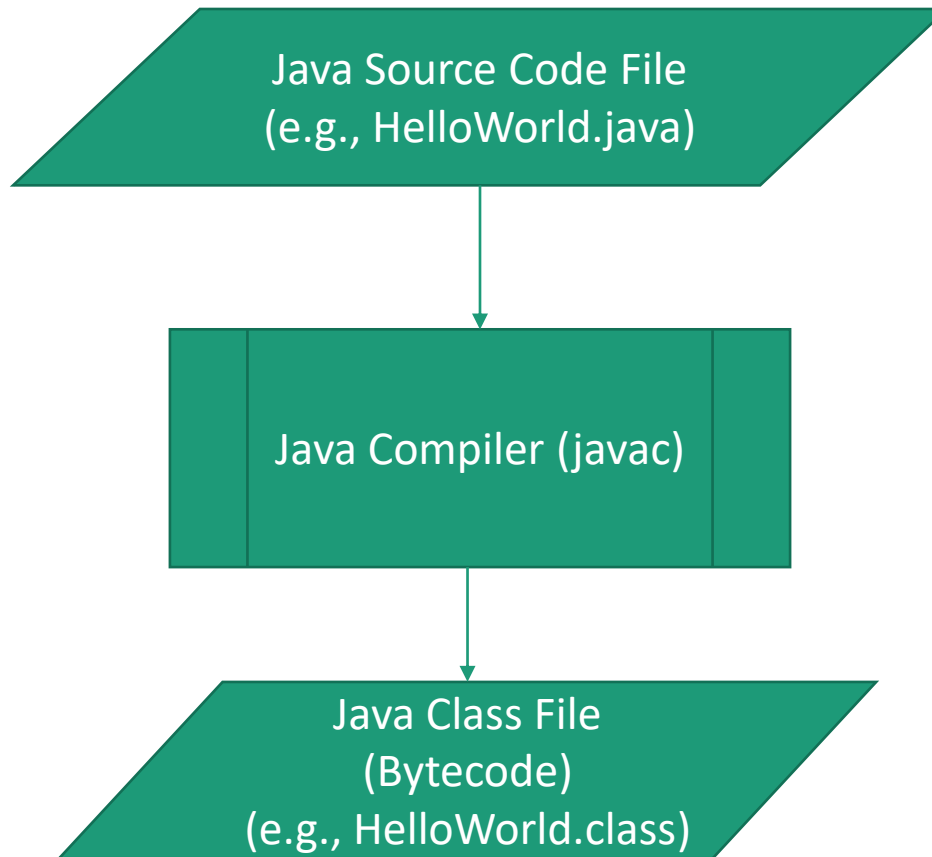
hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ ls
HelloWorld.class HelloWorld.java

hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ java HelloWorld
Hello, world!

hui@N-1432-59788 MINGW64 ~/CISC1115/examples
$ |
```



Compilation



Running Java Program

- You are running Java class files containing Java bytecode
- Example: `java HelloWorld`
 - The java program launches a Java Virtual Machine (JVM)
 - load the `HelloWorld.class` (and its dependencies), and start executing the bytecode in the class files

Troubleshooting

- Read the compilation error message carefully
 - Caveat:
 - The error message sometimes is accurate about what went wrong; sometimes not.
 - The compiler is more accurate at pinpointing where an error was found than telling what went wrong.
- Figure out what might be wrong, revise and compile it again
- Best practice: save often, compile often, don't have to wait.

Questions

- Prepare the environment to write Java programs
 - Git and Git Bash
 - Atom (or other your favorite editors)
 - In this class, the instructor prefer not to use an Integrated Developer Environment software (IDE, e.g., Net Beans, Eclipse, IntelliJ)
- Review the process of authoring a simple Java program

CodeLab Registration

- Course Section Code is in Blackboard