Working from Home

CS 15 Fall 2016
Why?
Beautiful Graphics
No lag or pixelated text
DISCLAIMERS

1. Before handing in a project, you MUST MUST MUST test your program by running it on a department machine, either on a Sunlab computer or through ssh.

2. It is possible to accidentally overwrite your new files with your old ones - and your terminal won’t always warn you!

TL;DR: PROCEED AT YOUR OWN RISK
DISCLAIMERS (cont.)

- If your code doesn’t run on a Sunlab computer, then the TAs won’t be able to grade it!!
  - This is true even if it works at home
- So you will receive a 0
“Not [cs015TAs’] chair, not [cs015TAs’] problem”

- BUT IT IS *YOUR* PROBLEM
This guide is both for Windows users and Mac users:

- Slides with Mac instructions will have titles in this color
- Slides with Windows instructions will have titles in this color
- Slides with generic instructions will have titles in white
IMPORTANT

This setup is designed for 64-bit Mac and 32 or 64-bit Windows computers.

If you have a 32-bit Mac, you can still follow the instructions, but you won’t be able to complete some steps, and the setup may not work (we haven’t tested, so we don’t know!)
How can you tell? 32 or 64

Click on More Info...
Locate the Processor Name

If you don’t have any of these two, you have a 64-bit Mac!

<table>
<thead>
<tr>
<th>Processor Name</th>
<th>Bitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Core Solo</td>
<td>32 bit</td>
</tr>
<tr>
<td>Intel Core Duo</td>
<td>32 bit</td>
</tr>
</tbody>
</table>
How can you tell? 32 or 64

1. Click on Windows icon
2. Search for “System” application (works for win. 8, 10)
3. Look at “System type”

For Windows 7, click here to see if you have 32 bit or 64 bit
GET READY…

- Want to be able to run Atom and Eclipse on your own computer?
- Don’t want to deal with `ssh` lag?
- Then pay **CLOSE** attention to these slides :)
  - WARNING: if you don’t pay close attention, your code may break on Sunlab computers!
  - and you will get a 0
What the f*** is ssh?

```bash
ssh -Y <cslogin>@ssh.cs.brown.edu
```

- **ssh** allows you to login to a CS department machine (AKA a Linux computer in the CIT)
- To set up **ssh**, reference this [online guide](#)
- **ssh** - a terminal command, like `cd` or `ls`, that stands for Secure Shell
- `-Y` - a flag that allows you to use graphical applications (like Atom) over ssh
- `<cslogin>@ssh.cs.brown.edu` - the account to which you will login via ssh
What the f*** is ssh?

If you don’t have ssh set up right now, you can’t complete this guide :/

You will need a department machine to set up ssh, so head over to Sunlab and follow these slides afterwards!
How can I tell if I’m in ssh?

- If you see your laptop name before the $, you are **NOT** in ssh.
- If you see `cslabXX` or `mslabXX` (where XX represent two digits, usually a number and a letter), you **ARE** “ssh’ed” into your CS account.
How can I tell if I’m in ssh?

If you open PuTTY, select **Brown CS**, and see this screen, you **ARE** “ssh’ed” into your CS account.
Open Up the Terminal!

1. Search for “Terminal” on your computer and open up this application.
2. Type `cd` to go to your home directory.

This is your username!
# Terminal Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cd</td>
<td>change working directory</td>
</tr>
<tr>
<td>ls</td>
<td>list files in directory</td>
</tr>
<tr>
<td>pwd</td>
<td>print current working directory</td>
</tr>
</tbody>
</table>
| mv      | move file (ex. mv file1 dir1: moves file1 to dir1 directory)  
rename file (ex. mv file1 file2: renames file1 to file2) |
| cp      | copy file content to another file  
(ex. cp file1 file2: copy file1 content to file2)  
copy files to another directory  
(ex. cp -r dir1 dir2: copy files in dir1 to dir2) |
| rm      | remove file |
| rm -r   | remove directory and its contents (ex. rm -r dir -r means recursive) |
| cat     | print file contents to terminal (ex. cat text.txt) |
Open Up the Command Prompt!

1. Click on Windows icon
2. Search for “Command Prompt”
3. Open

This is your username!
Command Prompt Commands

- `cd`: change working directory
- `dir`: displays contents of a directory
- `del`: delete file
- `rd`: remove empty directory
- `rename`: rename a file
- `type`: print file contents to terminal
1. Let’s create this folder(*not directly* linked to the department machines) on Desktop!
   **Type** `cd Desktop`

2. **Type** `mkdir cs015` to create a new folder named *cs015*. Please DON’T put *space* in your folder name!

3. In the future, to go into this folder *from home directory*,
   **type**
   
   `cd Desktop/cs015`
Create a local cs015 folder

The folder you just created will hold all the CS15 projects that you work on at home, and will resemble the structure of your cs015 folder on the department machines.
cs015.jar

- Huh?
  - The cs015.jar contains all of our support code
  - You **need** it on your computer *if you want your code to run properly* (or at all for that matter)
  - To get the cs015.jar, we need to copy it over from your department account to your local computer!
Get the .jar!

● Open your Terminal.
● Do **NOT** ssh into your CS account
● In your terminal, navigate to your local cs015 folder that you just created using

```bash
cd Desktop/cs015
```
Get the .jar!

● Once you are in your cs015 folder, type this into the Terminal and press enter:

● `scp <cslogin>@ssh.cs.brown.edu:/course/cs015/lib/cs015.jar .`

● On one line it should look exactly like this:

`scp <cslogin>@ssh.cs.brown.edu:/course/cs015/lib/cs015.jar .`
WOAH, HOLD UP!

- WTF IS SCP
- The `scp` (secure copy) command allows you to copy files to a local host (your laptop) from a remote host (your CS account) and vice versa
  - we just used it to copy the `cs015.jar` from the CIT computers to your personal computer!
  - type `ls` to confirm that you successfully `scp`'ed the `.jar`!
  - for more, see our `scp` handout!
Get the .jar!

- To get the cs015.jar, we are going to use WinSCP
- To download, go to this link. If you already have it set up, you can skip the next few slides.
- Click on “Installation package”
Set Up WinSCP!

1. Download and open WinSCP installer.
2. Press “Yes” and “Accept”
3. Choose “Custom Installation”
4. Accept default destination
5. At the “Select Components” screen, uncheck everything except “WinSCP application”, click next
6. At “Select Additional Tasks” screen, click next
7. Make sure you check “Commander” interface, click next
8. Press “Install”
9. Open WinSCP.
10. SSH into your CS account.
11. Click “Brown CS” & “edit”
12. Make sure “Host Name” box says “ssh.cs.brown.edu”
13. Enter your login at “User name” box. Leave the “Password” box blank.
Set Up WinSCP!

14. Click on “Advanced…”
15. Click on “Authentication” under “SSH”
16. Upload your private key saved when setting up SSH under “Private key file:”
Set Up WinSCP!

17. Click on “SFTP” under “Environment”
18. SFTP server should be “Default”
19. Make sure “Allow SCP fallback” is checked.
20. Press “OK”
Using WinSCP!

1. Click the Login button!

2. Type your SSH setup key password if it asks.
Using WinSCP!

On the left are the files from your own computer, and on the right are your files on the department machine!
Get the .jar!

- On the right half of the screen, click on the dropdown menu that shows your current folder, and click main (it should be near the top above gpfs)
Get the .jar!

- Double-click on course, then on cs015, and then on lib
- On the left side of WinSCP (your local files), navigate to your newly created cs015 folder on Desktop
Get the .jar!

- Locate the file called “cs015.jar” in the remote lib folder (right side)
  - make sure it’s the correct file, because there other versions of the .jar e.g. cs015_2014.jar
- Drag cs015.jar from the right side into your local cs015 folder (left side)
- If a window pops up, click “OK”
Get the .jar!

If your screen looks something like this, you’re done!

Make sure you have the .jar file
Now, we need to make sure our computer has the same version of Java as the Sunlab computers!

- This is **SUPER DUPER IMPORTANT**.
- If you do not follow these steps *exactly,* your code **will not work** and...
- **YOU WILL RECEIVE A 0**
Please download the following JDK and JRE:

Check "Accept License Agreement" in the gray box, click the links next to Mac OS X x64 to download the .dmg files.

You must have a 64-bit Mac (click here to check)


This one!
Installing Java

- Please download the following JDK:

  Check **Accept License Agreement** in the gray box.

  You must have a 64-bit Mac (click [here](http://www.oracle.com/technetwork/java/javase/downloads/java-archive-javase8-2177648.html#jdk-8u25-oth-JPR) to check)

Please download the following JRE:

Check "Accept License Agreement" in the gray box.

You must have a 64-bit Mac (click here to check)

Installing Java


- When taken through the installation wizards, don’t modify anything and continue to click “Next” until the files are installed
Please download the following JDK and JRE:
  ○ Check “Accept License Agreement” in the gray box, click the links next to Windows x86* or x64 to download the .exe files
    *To find out if you have x86 or x64, see slide 8
    *If you have x86, download the x86 Offline .exe file
  *Make sure to get stuff that says “8u25” at end


When taken through the installation wizards, don’t modify anything and click “Next” until the files are installed
Let’s Check!

In your terminal, type `java -version` and press Enter.
If it says `java version "1.8.0_25"`, you downloaded the right version!
Run Java from Command Prompt

1. Click on Windows icon
2. Search for “Edit the system environment variables”
3. Click on “Environment Variables.”
3. In the system variables box, scroll down and click “Path”, then click “Edit…”
4. Add `;C:\Program Files\Java\jdk1.8.0_25\bin` to the end of the variable value
Let’s Check!

On your Command Prompt, type `javac`

If it doesn’t recognize this, then setting up the PATH didn’t work….

Close the Command Prompt, open it up again, and type this and press Enter.

```
set path=%path%;C:\Program Files\Java\jdk1.8.0_25\bin
```
Now that we have all the necessary pieces in place, we can get started with Atom!
To use Atom on your laptop, you’ll need to download it.
Click the large red button
Click on the downloaded app and atom will open automatically
To use Atom on your laptop, you’ll need to download it.

We want the Windows option

To determine whether your PC is 32 (x86) or 64 (x64)

see slide 8

click here to download!
Configuring Atom

Now that we have Atom, we need to set up a path so that when you run your java files, they know to include the cs015.jar support code.

To do this, we need to set up a $CLASSPATH variable!
Configuring Atom

1. Open up terminal and type `cd`
2. Type `open .bash_profile` → this opens up a text file
3. If the file does not exist, use `touch .bash_profile` to create it, then try line 3 again
4. Type below, save file, and close it!
   ```
   export CLASSPATH=$CLASSPATH:/Users/<your user name, not CS login>/Desktop/cs015/cs015.jar:..
   ```
5. Close your terminal and open it again.
.bash_profile explained

So what just happened?!

- `.bash_profile` is a text file that contains code that runs whenever you start your Terminal.
- `export CLASSPATH=...cs015.jar:` means that whenever Terminal opens, the `$CLASSPATH` variable is set to `cs015.jar`!
- But what’s `$CLASSPATH`? Read on...
Let’s Check!

On your terminal, type

```bash
echo $CLASSPATH
```

If it returns

```
:/Users/<your user name>/Desktop/cs015/cs015.jar:...
```

you’re all set!
Configuring Atom

1. Click on Windows icon
2. Search for “Command Prompt”
3. Open and type

```
set CLASSPATH=%CLASSPATH%;C:\Users\<your user name, not CS login>\Desktop\cs015\cs015.jar;
```
CLASSPATH what?

- CLASSPATH is a variable that tells Java “look here when you compile code”
- Useful when you’re using a library of pre-defined classes like our support code
- When Java compiles, it looks at the CLASSPATH to find the support code - without it, cs015 code won’t compile!!
So what now?
Phew! Atom is all set up!

Now let’s get a project transferred over!
From now on, we will teach you how to use sftp to transfer files. sftp doesn’t require you to set up any application, but everything is done through terminal commands. If you don’t like using the terminal, go to slide 80 to use FUSE!
Copying Files Over

- To copy your files over from your CS account to your laptop, we are going to use **sftp**
  - **sftp** - Secure File Transfer Protocol
  - **sftp** will only work if you have **ssh** setup on your computer

- To get started, pull up your Terminal and make sure you are NOT in **ssh**
  - if you are, type “exit” in the command line and press Enter

- Then type the following:
  
  ```bash
  sftp <cslogin>@ssh.cs.brown.edu
  ```

- and press Enter
What did that do?

***If prompted to do so, enter your CS account pass phrase

● Your Terminal should look something like this:

![Terminal screenshot]

● Basically, you are now logged into **both** your CS account **AND** your local computer
  ○ for more about `sftp`, see our [helpful handout](#)
sftp Magic

● To navigate through your CS account, use normal commands like “cd” or “ls”

● To do the same on your local computer, add an “l” (that’s a lowercase L) before the commands
  ○ i.e. lls, lcd, lpwd
  ○ Click here for a commands cheat sheet!
lcd and cd (1)

- Use `lcd` and `cd` to navigate to your cs015 folder on your laptop and your cs015 folder on your CS account respectively.

```
Nickies-MacBook-Pro-2:Desktop nipucel$ sftp npucel@ssh.cs.brown.edu
Connected to ssh.cs.brown.edu.
sftp> cd course/cs015
sftp> ls
AndyBot    LiteBrite    TASafeHouse
sftp> lcd cs15_TA_stuff/cs015
sftp> ls
AndyBot    bin   gfx
Tetris_   cs015.jar    src
```

items in my CS account
items in my local cs015 folder

items in my cs015 folder
To walk you through copying files from the remote computer and your local computer (and vice versa), we are going to practice by copying AndyBot!

First, use `lcd` and `cd` to navigate to your AndyBot folders locally and remotely
- If you do not have a local AndyBot folder, make one!
If you type `ls` and press enter, you should see your `.java` files (and possibly `.class` files)
- the `.java` files are the files we want to copy!
- if you type `lls`, you should see no files
To get these files, we are going to use the...(wait for it)...`get` command!
Type `get *.java` into your Terminal, and press enter
The get command

- What did we just do?

  \[ \text{get } * . \text{java} \]

  * get is the sftp command for copying files from your remote directory to your local directory

  * basically means "all," so \(* . \text{java} \) means "all java files (in this directory)"

  Normally, we would put the location of the desired destination for our transferred files here, but since we leave it blank, sftp assumes we want to copy the files in our current directory (which, for us, is our AndyBot folder)
The `get` command

- Your screen should spit out a few lines that look something like this:

```bash
sftp> get *.java
Fetching /gpfs/main/home/npucel/course/cs015/AndyBot/App.java to App.java
/gpfs/main/home/npucel/course/cs015/AndyBot 100% 466 0.5KB/s 00:00
Fetching /gpfs/main/home/npucel/course/cs015/AndyBot/Maze.java to Maze.java
/gpfs/main/home/npucel/course/cs015/AndyBot 100% 792 0.8KB/s 00:00
```

- These lines show the progress of every file that is being transferred from your remote computer to your local computer.
The get command

● If you type `lls` to view the files in your local AndyBot folder, you should now see your `.java` files!

```
   sftp> ll
   App.java   Maze.java
```

● To learn more about the `get` command, see our nifty handout [here](#)
WinSCP: Copying Files

- To get back to your home directory (on the right side), navigate to
  `/gpfs/main/home/<cslogin>/`
- From there, navigate to your AndyBot folder
  (`~/course/cs015/AndyBot`)  
- On the left side, navigate to your local cs015 folder, and create a `local` AndyBot folder
WinSCP: Copying Files

- Drag your App.java file and Maze.java files from your remote AndyBot folder into your local AndyBot folder.
Did it Work?!

Try to open your files in your local Atom to make sure that AndyBot and the appropriate files appear, then run your code to see if it worked!
Moving the Files Back

- After you’ve edited your .java files locally, you’ll have to **put** them back into your CS account.
- So, naturally, we’re going to use the **put** command!
- Type `put *.java` into Terminal, and press Enter.
The `put` command

- The `put` command works just like the `get` command, except in the opposite direction.
- The `get` command “gets” files from your remote host and copies them to your local computer.
- The `put` command “puts” files from your local computer into your remote host.
Here is the basic structure for each command:

- **get** <remote files> <copy to this local location>
- **put** <local files> <in this remote location>

We left this part blank because `sftp` assumes we want to move files to our current directories.

For more detail about these commands, see our nifty handout.
WinSCP: Moving Files Back

- After you’ve edited your .java files locally, you’ll have to put them back into your CS account.
- On WinSCP, drag your files from your local folder (left side) to the remote folder (right side).
- If you get a popup window confirming you want to Overwrite, click Yes.
Which Files to Copy

Starting a new project?

- Connect over ssh and run `cs015_install <project>`
- Make sure you copy over any stencil code given to you by the install script to local folder

Continuing a project?

- No problem, but remember that files you edit locally will not be synced with files on the department computers--so you must copy the updated files into your CS account (note: this will overwrite the files in your CS account!)
Important Reminder

If you accidentally `sftp` (or use WinSCP) in the wrong direction, you will overwrite your new files with your old ones, and *you could lose hours of work.*

Make sure to be *very careful* when going back and forth, and consider saving a backup copy of your work in another directory.
...And Back to Reality!

When you finish your project or decide to return to the Sunlab, you’ll have to move your project back… and check to see if it still works.

Because if it does not work on department computers…. **YOU. WILL. RECEIVE. A. 0.**
To Run the Program...

Open your terminal, cd into your cs015 folder, and then into your project folder. Run the program like how you would in a department machine.

```
javac *.java
java <Project Name>.App
```
In the future, for running any projects that do not use our support code, you need to:

```bash
javac *.java
cd ..
java <Project Name>.App
```
To Run the Program...

Open your terminal, cd into your cs015 folder, and then into your project folder.

javac *.java

cd ..

java <Project Name>.App
Run and Done!

Run your programs on a department machine (or over ssh) to see if they work as intended!

If they do run as expected, you are DONE!
More Options: FUSE

- FUSE is a way to “mount” the CS department file system onto your own local files! *(optional)*
- What does this mean?
  - you can navigate to your CS files in Finder, just like with any other files
  - when you edit a file, changes are automatically saved in your department account - no copying files over!
  - open cs15 project files on a lag-free local Atom (or Eclipse), edit away, then save to your CS account
- Sounds great, how do I get it?
  - check out our guide [here](#)
More Options: Aliases

- What if I don’t want to type the long command for `ssh`?
- Introducing… aliases! *(optional)*
- An alias is a short terminal command that can be used to replace a longer one (like logging into ssh)
  - For example, if `myssh` is your alias for `ssh -Y <login>@ssh.cs.brown.edu`, running `myssh` from your Terminal would log you into ssh
- To learn how to set up an alias on Mac, see [this guide](#)!
Sneak Peek:

Next week on “CS Swag You Need”:

*Life With Your Own Eclipse*

Check back soon for how to setup the Java IDE you will grow to love: Eclipse!

(Slides will be available online)
Best of Luck!