# Hands-on 1: Unix File System

Michael Plasmeier

1. The . directory is the same as the current working directory, so you stay in the same place. The .. is your parent directory, so you go (change your working directory) to that.
2. You can use . when you want to refer to that directory. For example if you want to tar the contents of your current directory you can use:

tar -czvf mytarfile.tgz .

1. The access, modify, and change times have been updated because the directory listing had been edited to add the two files. You can look at the directory (vim .) to see the listing of the directory, and see that it includes the two new files.
2. In addition to the access, modify, and change times being updated, the number of incoming hard links has incremented. This is because baz/.. is now pointing to ..
	1. Why does Unix maintain the link count? Is it so that it can delete a file when it has no more inbound links?
3. Simlinks help keep things organized and simple. For example, you can type /mit/6.033 instead of /afs/athena.mit.edu/course/6/6.033. However, you cannot rename or move files without breaking the link.
4. The directory could not be found.

../theplaz: No such file or directory.

We were actually in /afs/athena.mit.edu/course/6/6.033. .. refers to /afs/athena.mit.edu/course/6

1. To make things easier to type. Also the 6.033 has folder has been added to our path by running add. I am not sure if this is normal Unix behavior, or a special property of add.
2. I would simply type cd ~. I could also use enough .. to reach /afs/athena.mit.edu
3. Since . was on the path, we were running demo and ls that were in the 6.033 folder. You can see the mischief that ls was up to:

#!/bin/sh

echo "Sending all private emails to 6.033-staff"

echo "Just kidding!"

~

1. The assignment itself took about 1 hour, since I was going very slow and exploring stuff on my own. I also took 2 hours to read Chapter 2 and produce 13 pages of notes.
2. A linked list can expand in size so you don’t need pointers to pointers to pointers. However, it can take forever to traverse.
3. Yes it gets the provided example correct. I do not know if it always does this.