Paths

When you use the shell (bash) to move around in the directory tree, you will have to tell the operating system where you want to move to. You do this by specifying the so called *Path*. This is nothing else but a hierarchical list of directories starting at the root that identifies the location of a directory or a file you want to access.

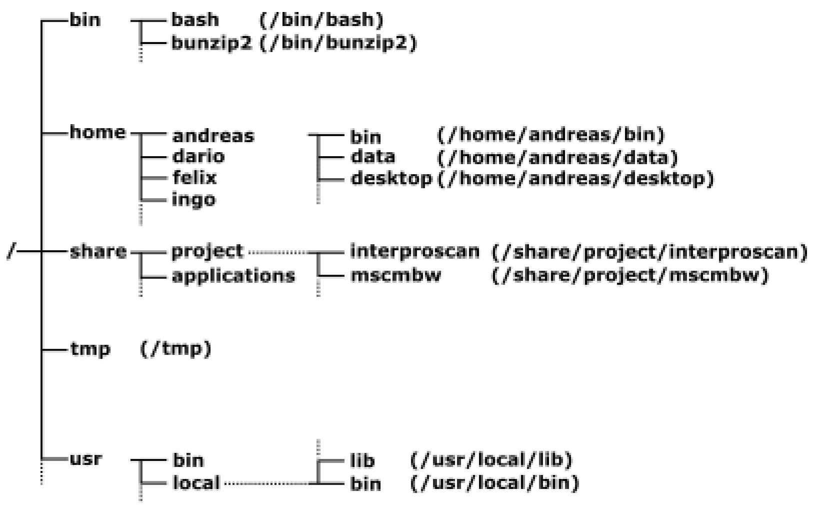

Figure 1: The directory tree in a linux system. The '/' to the left indictates the root directory. You can reach each directory or file from each place in the tree by specifying the absolute path starting from the root, or by specifying the shortest path starting from the directory you are currently located in. The absolute paths for individual locations are given in parentheses.

Note, the Path uniquely identifies a directory or a file in the linux file system (Fig. 1). There are four characters that are helpful to memorize

1. The '/' denotes the root of the file system if it is placed at the beginning of the path. Otherwise it just separates directories
2. The '.' (yes, it is a dot) denotes the directory where you are currently located
3. The '..' denotes one directory above your current location
4. The '~' denotes your home directory.

In Linux systems, there are two different kinds of paths, absolute and relative paths.

Absolute paths

An absolute path always starts at the root of the directory tree, and thus start with a '/'. Absolute paths are valid from any position in the directory tree. For example, the absolute path of the directory desktop in Figure 1 is

```
/home/andreas/Desktop
```

Remember, ~ is a shortcut for the home directory. In essence, any path starting with ~ is an absolute path starting at the home instead of the root directory.

User *Andreas* has a therefore an alternative to specify the absolute path to her *play* directory. He can write

```
~/Desktop
```

to access *Desktop* from anywhere in the directory tree, as the ~ is a short cut to Andreas' home directory.

Note, in principle other users can use the same trick. They just have to specify whose home directory they want to access by appending the user name to the '~'.

```
~Andreas/Desktop
```

Relative Paths

Relative paths typically start at the current position in the directory tree, and thus start with a ./. It is for this reason that they depend on the current position and change once you move to a different directory in the tree.

For example, if the user *Andreas* is in his home directory and wants to specify the path to the directory Desktop, he just has to write

```
./play
```

where the dot identifies the directory the user *Andreas* is currently in.

Sometimes when using relative paths, it is desired to go up a level in the file tree, e.g. from */home/andreas/Desktop* / to */home/Andreas/bin*. The relative path has to point first back to */users/carol* and from there to the *work* folder. This can be done with '.. '. In our example the path would look like

```
../bin/
```

Moving around in the directory tree

Moving in the tree is easy, just type the command *cd* followed by a space and then the path identifying the position you want to move to. For example

```
cd /
```

will bring you to the root directory.

```
cd ~
```

will bring you to your home directory. Note that calling the function *cd* without any path has the same effect.

```
cd ..
```

will bring you one directory above from where you are currently located.

```
cd .
```

will do… nothing, as you specify the directory where you are currently in as the target directory.