ezkpp Documentation

Release 0.1

Tomas Chor

Oct 16, 2016

CONTENTS

Contents:

CHAPTER

ONE

INTRODUCTION

This guide is aimed at providing additional information not covered by the official KPP manual. We focus on the latest version of the pogram, release 2.2.3, which can be freely downloaded at its official webpage. We recommend any person reading this manual to keep a copy of the original mnaual (which can be downloaded here) since this guide is not meant to replace the original manual, but to suplement it.

CHAPTER

TWO

ABOUT BASH

The official KPP manual is entirely based on Unix Shell, which is command language which most of the Linux distrobutions use to interact with the system without a Graphical User Interface. The manual, however, assumes a non-trivial knowledge of this tool, which makes it difficult for users not experienced with terminals and command line interfaces (which includes bash, C shell, MS-DOS, PowerShell, ksh etc.) to install and run the simulations effectively. The approach adopted in this guide will be to go through the steps necessary to compile and run KPP, as stated in the manual, but taking the time to explain them a little better how to do them, and what exactly it is that they do.

We will first go over a few basic notions necessary to understand what is going to be done in the guide. If you are familiar with the concepts of system shells, you may skip the next sections.

What is Bash?

But what is a shell? A system shell is the name what computer engineers use to refer to the outter layer of an Operaional System (OS). It is said outter layer because it separates the user (you) from the core of your OS. So it separates you from the intricate group of codes that ultimately governs your machine and lets you interact with your computer using a human-readable language (and not, for example, binary!). So a shell is a bridge between you and your machine.

These shells can be either graphical shells (called Graphical User Interface, GUI, just like what you use during mundane tasks such as browsing the web and reading a PDF document) or text shells (also called terminals or Command Line Interface, CLI). Graphical shells are easier and extremely intuitive (most people use the mouse in a GUI and never needed to be told how to do it), but they are very limited. Basically all you can do is click on buttons that were previously programmed to to some task and input text.

Texts shells (terminals), however, are extremely powerful. You can do virtually anything with your computer using them. That comes to the cost of terminals not being intuitive at all. Since KPP is a complicated code for which there is no graphical interface, we need to use a terminal to compile ("install") and run it, simply because this task requires a more powerful tool then your mouse.

Bash (acronym for Bourne Again Shell) is a kind of Unix Shell used by most of the Linux systems and some Mac OSs. Some other shells can be used to perform the same tasks (the KPP manual itself also gives some commands in C Shell, which is another Unix Shell), but we focus on Bash here because it is the most common and most easily accessible. Most of Linux distributions use it, and some Mac OSs use it as well. Furthermore, it can be natively installed into Windows 10, as we will explain in the next section.

Accessing Bash

To access and use Bash, you either need a Bash emulator or to be in an operational system that supports it natively. Various emulators exist (Cygwin, cmder, MinGW, etc.) but they are not recommended because some of them contain many bugs. If you would like to try those anyway, chances are that it'll work, since we're going to be doing simple

tasks and many people use those. However, running it natively is a always a garantee of no bugs, so (in the spirit of keeping it general) that's why it's the most recommended option for this guide.

We will briefly go through your options for each of the 3 most common operational systems.

From Windows

Windows doesn't support Unix Shells natively by default, so here are the options.

If you're using Windows 10, you can natively install the Ubuntu 14.04 inside your Windows machine with the Windows 10 anniversary update, which is available for every up-to-date Windows 10 computer. Directions to do this are very somple and are given in many places (such as here) so for now we will not explain them in detail. This will give you Bash running natively on Windows. But only works for up-to-date Windows 10 computers.

If you do not have Windows 10, you can either install one of the many Bash emulators for Windows or you can install a Linux virtual machine inside your Windows computer. You can do that using Virtual Box and installing a Ubuntu-based distribution (we recommend installing either a recent version of Ubuntu or Linux Mint 18 (or greater), since these two are most suited for beginners in Linux). Again, directions on how to do this are straightfoward and exist all over the internet, so we will not spend time on steps on how to do that.

From Mac OS

If you have a Mac, you might already have Bash natively installed, since all Macs are based on Unix. To find out what your shell is, you need to open a terminal application (generally under utilities). Then type the command *echo $SHELL* and press enter. If the output of the shell is something ending in Bash, like */bin/bash*, then you're already running Bash. If it ends in something else, like */bin/ksh*, then you're running a different Unix Shell. Most commands should be the same, but if you want to use this shell you might have to translate some (which should be easy with the help of Google).

If you're running another terminal and would like to try Bash, you can either get an Bash emulator for Mac, install a Linux virtual machine (as described in the Windows section) or change your terminal to Bash. The most recomended here is to change your Shell to Bash. Instructions on how to do this are easy and can again be found in many places, including here.

From Linux

If you're running Linux you can open a terminal and run the command *echo $SHELL* to find out if you're running Bash or not. If you're not you can try to keep going with your Shell (some commands may need to be translated) or you can change your default Shell with the *chsh* command. You can find more detailed information on that in many places, such as here.

CHAPTER

THREE

COMPILING KPP

In this chapter we detail how to successfully download and compile KPP on your system under the Bash environment.

Downloading into your folder

One of the first things to be said is: most of the commands we will use will only work if you're in the right directory (which we will always tell what it is). So when you open a terminal, that terminal is "running" in some directory in your computer. You can find out which directory that is by entering the command *pwd* which stands for "Print Working Directory". That will show you exactly where you are on your computer. You can also enter *ls*, which will "list" everything you have on that directory. To change directories, you can use the command *cd*, which stands for "Change Directory". So if you want to go to your downloads directory, you can type *cd Downloads*, or *cd /home/user/Downloads* depending on where you are on your terminal (the first is a relative and the second is an absolute or full path).

If you prefer to download KPP through its website manually and unpack it somewhere, you'll have to go there with your terminal. So, if I unpack it in my home directory, as soon as I open my terminal I'll have to use *cd /home/myname/kpp-2.2.3*. This command will only work if the path is correct (it won't work on Windows, for example, which does not have a */home* location. If you're using Bash on Windows it's better to go with the following alternative.

Alternatively, you can open a terminal and run

```
wget http://people.cs.vt.edu/~asandu/Software/Kpp/Download/kpp-2.2.3_Nov.2012.zip
unzip kpp-2.2.3_Nov.2012.zip
cd kpp-2.2.3
```

which will automatically download the software, unpack it and move to the correct directory (which was created when unpacking).

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search