The LHCb Starterkit

Albert Puig, on behalf of the LHCb Starterkit Team
Universität Zürich (Switzerland)
E-mail: albert.puig@cern.ch

Abstract. The vast majority of high-energy physicists use and produce software every day. Software skills are usually acquired “on the go” and dedicated training courses are rare. The LHCb Starterkit is a new training format for getting LHCb collaborators started in effectively using software to perform their research. The course focuses on teaching basic skills for research computing. Unlike traditional tutorials we focus on starting with basics, performing all the material live, with a high degree of interactivity, giving priority to understanding the tools as opposed to handing out recipes that work “as if by magic”. The LHCb Starterkit was started by two young members of the collaboration inspired by the principles of Software Carpentry, and the material is created in a collaborative fashion using the tools we teach. Three successful entry-level workshops, as well as an advance one, have taken place since the start of the initiative in 2015, and were taught largely by PhD students to other PhD students.

1. Introduction
When the field of high-energy physics (HEP) started out in the previous century, the process of analyzing data looked vastly different than it does today. Events were counted by hand, and discoveries were made using oscilloscopes and photographs; physical tools, such as rulers, were used to determine properties of photographed tracks. Up to 50 years ago, universities employed teams of people to study such photographs. Over the course of the XX century, data taking and analysis techniques have evolved immensely and manual methods were replaced, on a large scale, by digital readout and analysis using computers. The size of the collected datasets has increased over time—in parallel with the affordable computing capacity—and with it the complexity of the software stacks needed to guarantee their efficient processing. Experimental physicists have thus become computer experts, and for for people to take that step confidently, training becomes indispensable.

It is much too common, however, that computing training in high-energy physics experiments is lacking. Many will recognize the pain of working through broken tutorials and the effort of using outdated, or even completely missing, documentation. Hours are wasted by people running into the same types of problems, and experts answering the same questions, over and over again. This can be mitigated by adopting a centralized setup to provide working, up-to-date tutorials and answers to frequently asked questions. The LHCb Starterkit aims to accommodate such a setup by replacing the traditional static, often-unmaintained web sites\(^1\) by a modern, collaborative approach.

\(^1\) TWiki pages [1], editable by all members of the collaboration, are the most used way in the LHC experiments to collect documentation and tutorials.
2. The LHCb Starterkit
The LHCb Starterkit [2] is an initiative started in 2015 with the aim to provide software training, by and for the LHCb collaboration members, with special emphasis on helping new members—usually young PhD students or even MSc students. Software includes general tools, such as Python [3] or Git\textsuperscript{TM} [4], as well as collaboration-specific packages, and focus is put on learning the building blocks of each piece of software while avoiding recipes. As a consequence, training entails ensuring the availability of up-to-date tutorials on the covered subjects, as well as organizing workshops where participants obtain hands-on experience with these tutorials under the supervision of more experienced collaborators.

The Starterkit initiative is based on three founding goals:

- \textit{Improve software literacy.} New collaboration members come from many different backgrounds and have varying levels of experience, and both the tutorials and workshops help to get everyone on the same level. This, in turn, makes communication and problem-solving easier for everyone.

- \textit{Teach good practices.} There are many ways to write software, and for each there are numerous reasons to pick or eschew them. By getting people on the same track in this regard, especially young members of the experiment, collaborating becomes more straightforward which contributes, in the long run, to the quality and maintainability of the LHCb software stack. Special emphasis is put on the importance of documentation.

- \textit{Socialization.} The organization of workshops allows young collaboration members, especially new ones, to get to know each other, as well as some of the experts in key areas of the software; this is further effectuated through social events during those workshops. Achieving a good socialization fosters a climate of communication and makes the young members feel welcome to ask questions and share their ideas.

While one of the main outcomes of the Starterkit initiative is a central repository for the creation and maintenance of teaching and self-help material, the organization itself is entirely decentralized. There is no strict hierarchy within the group of people working on the Starterkit—decisions are made democratically. Because there is no single person in charge, which could be damaging to the project were they to leave the collaboration, work can always continue with people present at any particular time. However, it is important to note that most of the organizers, like the target audience, are PhD students, which implies a very heavy rotation of the people involved. To ensure the survival of the setup, former attendees of the workshops are encouraged to take responsibilities within the initiative in subsequent editions. Knowledge and experience is continuously transferred, and the project endures.

2.1. The tutorials
The LHCb software, while based on relatively simple building blocks, is quite intimidating to newcomers and has a steep learning curve. Prior to the Starterkit, it was very common for new students to “inherit” scripts from their supervisors and hack them to try to adapt them to their needs; this approach is very error-prone and results in many questions on mailing lists that could have been easily avoided with a good basic training on the principles of the software stack. To solve this problem, the software tutorials built by the Starterkit initiative focus on providing newcomers with the understanding of basic software building blocks, giving them the tools necessary to build on and perform complex tasks later on.

The tutorials, inspired—both in structure and content—by the well-established Software Carpentry tutorials [5], are freely accessible, both within and outside of the LHCb collaboration [6, 7]. They are hosted as GitHub [8] webpages [9, 10], available under a Creative Commons Attribution License [11], and anyone can—and is encouraged to—submit issues and
pull requests to update them. This approach offers two advantages: on one side, its openness makes it very easy to collaborate, and as a consequence many LHCb experts, as well as users, have contributed to its improvement; on the other, the public availability of the tutorials and the use of a tool like Github to maintain them has ensured that they have stayed up-to-date and in an optimal shape, even when significant changes have occurred in the LHCb software stack. In total, more than 100 issues and more than 100 pull requests have been received from over 35 contributors.

The following topics are covered:

- **Bash.** Using a shell allows one to quickly navigate folders and perform (scriptable) operations on files. Since this is something every experimental particle physicist will have to do on a daily basis, an introduction on how to use the shell is given. Basic bash scripting is also included to cover typical use cases, such as invoking a program multiple times with different options or submitting jobs to a batch system.

- **Git.** Modern version control is paramount to the success of large software packages. LHCb recently adopted Git, and teaching people the ins and outs of this version control system helps them develop and share code more efficiently. Additionally, the specifics of interacting with the LHCb repositories are also covered, easing the students’ way into contributing to the LHCb core software.

- **Python.** This language is widely used in LHCb, both for data analysis and to configure the LHCb applications, which use C++ to do the underlying work. Python 3 is used as a version of choice, with discussion on backwards compatibility with Python 2.

- **LHCb Software stack.** At first, an introduction to the LHCb data flow, from data taking to final analysis, is given, putting emphasis on pointing to the relevant detailed documentation, followed by an introduction to the LHCb software and its basic ideas and tools. After this, the reader is taken through a set of lectures that build on and take them from starting the software to producing simple files containing physics data; to achieve this, the topics covered by these lectures include the interaction with the Grid [12], the interactive exploration of data sets, and the script-based production of ROOT-based NTuples [13] using the LHCb software. This is the largest body of material, and is built such that at no point a magic recipe is given for a particular task—the building blocks are shown and explained, so the student will be able to build on them to solve any task at hand.

Once the basic software tutorial [6] has been mastered, a second one [10], featuring more advanced and LHCb-specific topics, is available.

### 2.2. The workshops

Once per year, the **Starterkit workshop** is held at CERN. It has two main organizers, which change every time and are usually drafted from previous Starterkit workshop participants, with the support from 10–15 volunteer instructors and assistants. As previously discussed, the target audience of the workshop, limited to 40 students divided into two groups, are PhD and MSc students who are new to the collaboration. The LHCb collaboration has about 80 new students each year, so the workshop includes about half the students—enough to make a noticeable impact on the software literacy of the collaboration as a whole.

The workshop takes four days, divided in two sections of two days each: the first section covers the general computing tools, i.e., bash, git and python, while the second section—which lasts two or two-and-a-half days—focus on LHCb-specific software. A hands-on approach, inspired by the Software Carpentry Workshop Operations guidelines [14], is followed: the participants follow a main instructor in completing each of the tutorials, and can ask for help at any time from the
3–4 assistants in the room. Teaching tools such as the use of sticky notes to communicate with the helpers and instructors, as well as to give feedback, are extensively used. The sessions are designed to be very interactive and informal: the instructor asks lots of questions and encourages the students to ask too; additionally, there is no copy-paste of pieces of code, and the instructor always types everything to make the lessons easy to follow. The high ratio of experts to students, one per 4–5 students, ensures this interactivity and level of help can be maintained even in the most complex lessons, at the cost of limiting the attendance capacity of the workshop.

As previously discussed, a social event is organized during the workshop to help establish a crucial contact among peers, easing their way into a new collaboration and making them feel welcome by the LHCb experiment.

In 2016, by the way of experiment, the first Impactkit workshop was organized. This was a three-day follow-up workshop, focused on the more advanced LHCb-specific software tutorials. Twenty participants, mostly attendees of previous Starterkit workshops, were given two days of advanced lectures on LHCb topics, much in the same manner as during the Starterkit workshop. The last day was organized as a hackathon: in groups of 2–4 people, the participants were tasked, guided by experts, with solving real problems involving the LHCb software over the course of one day. The issues ranged from solving longstanding bugs in the software to writing missing documentation for important packages and creating new scripts and interfaces—all with the aim to be actually used in production. The idea behind this session was not only to put in practice all the learned concepts, but to ease the students into contributing to the software.

The hackathon proved successful, with most of the issues being solved and many solutions now being used in production. The completion of the Impactkit challenges represents the crystallization of the teachings and goals of the Starterkit initiative: starting six months prior in the Starterkit, the students were able to make meaningful, well-documented contributions to the very complex LHCb software stack in an autonomous way. This success has lead to the Impactkit workshop joining the Starterkit in its annual cycle, with the two being separated by six months.

3. Conclusion

The Starterkit initiative has been a huge success and has received overwhelmingly positive feedback from across the collaboration. The tutorials are being actively followed and updated by many—including senior physicists coming back to physics analysis after a few years—and the workshops are always oversubscribed. Workshop participants, over a hundred since the start of the Starterkit, have hailed the clarity and usefulness of the tutorials and workshops, and many have since joined in organizing subsequent workshops.

The Starterkit has enabled new collaboration members to kickstart their career within high-energy physics, and prevented them from feeling lost in the myriad of software that is used in the everyday life of an experimentalist. As a result, the efficiency with regard to software has increased collaboration-wide—an improvement that is beneficial to everyone.

All in all, the need for a centrally organized, functioning educational system within HEP collaborations could not have been more evident, and we hope similar initiatives arise in other experiments.

4. Acknowledgements

The Starterkit Team wants to thank all the people that have contributed to making it a reality, both through tutorial writing, checking and updating, and by helping or teaching in the workshops. Special thanks also go to the LHCb Secretariat, which have helped in many

---

2 Helpers in Software Carpentry language.
organizational aspects. Finally, we want to thank Marco Clemencic for presenting this oral contribution at the CHEP conference.

References
[1] TWiki – the Open Source Enterprise Wiki and Web Application Platform. http://twiki.org/
[2] LHCb Starterkit. http://lhcb.github.io/starterkit
[3] Python Software Foundation. Python Language Reference. Available at http://www.python.org
[4] Git distributed version control system. Available at https://git-scm.com/
[5] Wilson G. Software carpentry: Lessons learned. http://f1000research.com/articles/3-62/v2
[6] First Steps in LHCb. https://lhcb.github.io/first-analysis-steps/
[7] Second analysis steps in LHCb. https://lhcb.github.io/second-analysis-steps/
[8] GitHub. http://www.github.com
[9] https://github.com/lhcb/first-analysis-steps.
[10] https://github.com/lhcb/second-analysis-steps.
[11] Creative Commons Attribution 4.0 International License. https://creativecommons.org/licenses/by/4.0/legalcode
[12] Worldwide LHC Computing Grid. http://wlcg.web.cern.ch/
[13] Brun R and Rademakers F 1997 Nucl. Instrum. Meth. A389 81–86
[14] Software Carpentry Workshop Operations. https://software-carpentry.org/workshops/operations/