Thoughts on Tutorials and Training for Geant4

- **Caveat**: largely a brain/notes dump that I've collected over the past year or so!
  - *… so there'll be plenty of points you'll be saying "yes, but…"*
  - *… nor may it all be particularly coherent*
- **Not a direct proposal** - just ideas to start a discussion and get people's thoughts.
  - *… in particular requires time/effort which we know are limited…*
  - *… but it **might** save these in the future*
- **Not a replacement** for Documentation/Examples
  - *… though it is related*

Motivations

- **Go to YouTube and search for “Geant4”**
  - *… no, seriously, go and do it!*
  - *… or “Geant4 tutorial” on Google*
- We are getting asked about these on the forum both directly/indirectly
  - *… and the authors of said material don’t contribute/answer questions here*
  - *… might not show “best/current practice”*
- We ***do*** hold tutorials/courses, but these are naturally limited in availability
- Examples/Documentation can be a steep learning curve or intimidating

Thoughts…

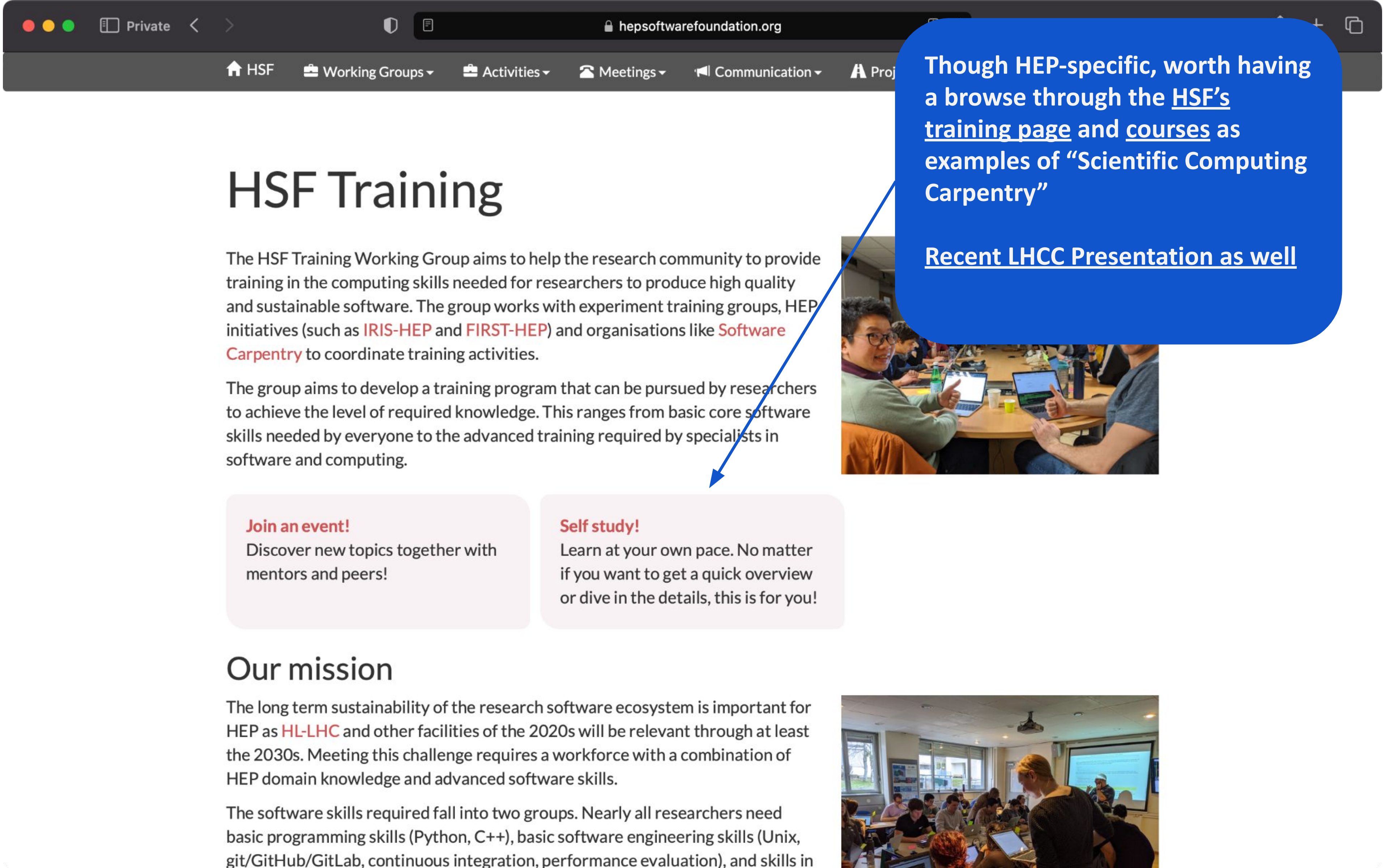
- **Is there scope for us to gather our existing courses together into a single syllabus**?
  - *Beginner, Advanced "modules" like CERN?*
  - *"Optional" ones for HEP/Medical/Space?*
- Could it be developed as an open, community project similar to Software Carpentry?
  - ***We*** *develop it, but* ***invite contributions*** *to improve/add, reviewed of course.*
  - *Use Carpentry "teach the teachers" method to expand/maintain pool of tutors*
- Have at least part of the course(s) **online or in suitable format for self-study**

Why a single syllabus?

- **Reduce duplication of material and thus effort**
  - *… including keeping it current*
  - *Clearer “beginner”, …, “advanced” parts, potentially “For Domain X” components*
  - *Better co-working with docs/examples*
- Easier to engage with users for feedback or who want to contribute material/training
  - *… HowTos, **translations**, “this isn’t clear..”*
  - *Hence Geant4-led community type project*
- **Make it easier to hold courses more widely**
  - *… broaden pool of tutors across the world*
  - *… even online*
  - *… maybe even get funding for this*

Why online / self-study?

- **Not all users will be able attend a tutorial in person, even if we broaden the availability**
- Docs/Examples *somewhat* throw users in the deep end, or at least *may* be intimidating
  - *… questions on Discourse are often in the form "I modified example X…"*
- One thing some YouTube tutorials do well is **incrementally building up an application**
  - *… if we teach this way, the material naturally lends itself to a self-study format*
  - *… and can be neatly integrated with Docs, partially there in App Developer Guide already*
  - *… with time to expand on the "why" as much as the "how"*



HSF Training

The HSF Training Working Group aims to help the research community to provide training in the computing skills needed for researchers to produce high quality and sustainable software. The group works with experiment training groups, HEP initiatives (such as IRIS-HEP and FIRST-HEP) and organisations like Software Carpentry to coordinate training activities.

The group aims to develop a training program that can be pursued by researchers to achieve the level of required knowledge. This ranges from basic core software skills needed by everyone to the advanced training required by specialists in software and computing.

Join an event!
Discover new topics together with mentors and peers!

Self study!
Learn at your own pace. No matter if you want to get a quick overview or dive in the details, this is for you!

Our mission

The long term sustainability of the research software ecosystem is important for HEP as HL-LHC and other facilities of the 2020s will be relevant through at least the 2030s. Meeting this challenge requires a workforce with a combination of HEP domain knowledge and advanced software skills.

The software skills required fall into two groups. Nearly all researchers need basic programming skills (Python, C++), basic software engineering skills (Unix, git/GitHub/GitLab, continuous integration, performance evaluation), and skills in

Though HEP-specific, worth having a browse through the HSF's training page and courses as examples of "Scientific Computing Carpentry"

Recent LHCC Presentation as well

Tooling/Tech

- **Known “first install Geant4” startup cost**
  - *Plus associated development/graphical environment*
- Have solutions for this in form of Virtual Machines, Docker/Apptainer, Packaging, e.g.
  - *https://geant4.lp2ib.in2p3.fr*
  - *https://gitlab.cern.ch/geant4/geant4-dev/-/issues/80 and as shown in Plenary 2*
- Like course material, can we unify these and co-work with the syllabus?
  - *“Layered” use, cmake -> rpm/deb/brew -> easy install || VM/Docker build*

Yes, but…

- Start up cost of everything…
  - *… even if it’s only collation*
  - *…with our resources stretched as it is*
  - *I think there’s a long term **saving** though*
- Domain-specific (HEP, Medical, etc) aspects
  - *Maybe as “optional modules”?*
- Prerequisites on tutee’s C++/etc knowledge
  - *Just have to be honest, link to other courses*
  - *Don’t hand-hold, e.g “Geant4 without C++” examples, except for teach/dev environment*
  - *Python/Julia not a solution/panacea here*
- Keeping material up-to-date…
  - *… just the same with Documentation*
  - *…collation might help to share load though*

Post-Parallel 2A: Plan for next steps

- Ben, Susanna, Carlo, John, Gabriele discussed what we could do next
  - *Preliminary - welcome further input/ideas from you!*
  - *Come and chat with us in the coffee or lunch break!*
- **Step 1a**: Gather list of current courses and material
  - *Understand course structure and contents with authors*
  - *What worked/works, what doesn't*
  - *Balance of "lecture" material (e.g. core concepts/algorithms) vs hands-on coding, how each is timetabled*
  - ***Aim**: Determine if common/extended syllabus possible, outline a basic structure with lectures/hands-on. See if hands-on part can work as a standalone "self-taught" component*
- **Step 1b**: Investigate tools/resources for development environment for teaching/learning.
  - *E.g. "standard" CLI, IDEs like VSCode, Containers, VMs, Jupyter*
  - *Online learning/IDEs systems like Open EDX, Moodle, …*

Questions, ~~Arguments~~ Discussion