Want Students Ready to Contribute?
Let’s Discuss What They Should Know
Hello!

I am Heidi Ellis
I am a CS & IT Prof at Western New England University
Co-maintainer of Gnome MouseTrap
You can find me at @heidijcellis
Hello!

I am Greg Hislop

I am a Professor in the College of Computing and Informatics at Drexel
You can find me at @GregoryHislop
Hello!

I am Gina Likins
I am on the Red Hat University Outreach team
You can find me at @lintqueen
Topics

◇ What’s the problem?
◇ What should students know?
  ▪ Breaking it down: Technical knowledge
  ▪ Breaking it down: Professional skills
◇ Bad news: Blockers
◇ Good news: possible paths to a solution?
◇ Conclusion
What’s Going On?

Also known as the problem statement
Focus on Open Source in Industry v. Higher Ed

Higher Ed → Open Source → Industry

Students who learn FOSS → Companies that use FOSS
Idea:

Students can gain experience by participating in open source communities while in college.
What should students know to contribute?

Technical Skills

Professional Skills
What should students know to contribute?

Technical Skills

- coding
- version control (GitHub, git)
- IRC & mailing lists
- bug & feature trackers
- open source licenses
What should students know to contribute?

Professional Skills

- how to understand community structure
- teamwork
- oral & written communication
Problem Solving = Find a Solution
Critical Thinking = Evaluation
Information Processing = Research
Management = Self Organization
Assessment = Post-mortem
Summary

- Student participation in FOSS is a no-brainer
- Need technical & professional skills for student success
- Why aren’t more schools teaching FOSS?
  - IOW, what are the “blockers”
Reason !: “Publish or perish” – tough when there are no dedicated journals or conferences
Reason 2: Can be tough to create new classes
Reason 3: Open source & academic timelines: out of sync

Fedora Project
- Release Cycle: 6 mo
- Planning Cycle: ~1 yr
- Execution Time: 6–12 mo

Academia
- Release Cycle: 1–4 yrs
- Planning Cycle: 2–5 yrs
- Execution Time: 4 yrs
Reason 4: There’s too much to teach already
518 pages;
18 “Knowledge Areas”
518 pages;
18 "Knowledge Areas"
1 of 18 Knowledge Areas → Social Issues & Professional Practice (SIPP)
518 pages;

18 “Knowledge Areas”

1 of 18 Knowledge Areas → Social Issues & Professional Practice (SIPP)

10 Topics in SIPP →

1 is Intellectual Property (IP)
518 pages;

18 “Knowledge Areas”

1 of 18 Knowledge Areas → Social Issues & Professional Practice (SIPP)

10 Topics in SIPP →

1 is Intellectual Property (IP)

Within IP → 9 Subtopics (7 “Tier 1” & 2 “Elective”)
518 pages;

18 “Knowledge Areas”

1 of 18 Knowledge Areas → Social Issues & Professional Practice (SIPP)

10 Topics in SIPP →

1 is Intellectual Property (IP)

Within IP → 9 Subtopics (7 “Tier 1” & 2 “Elective”)

1 of the “Elective” Sub-Topics →
518 pages;

18 “Knowledge Areas”

1 of 18 Knowledge Areas → Social Issues & Professional Practice (SIPP)

10 Topics in SIPP →

1 is Intellectual Property (IP)

Within IP → 9 Subtopics (7 “Tier 1” & 2 “Elective”)

1 of the “Elective” Sub-Topics →

“Foundations of the Open Source Movement”
The Biggest Reason:
Instructors do not have experience with open source
A Further Blocker: Instructors are expected to “know everything”
The Good News....
Solutions....
Open Path

Pathways Model Workshop 2016

Contents [hide]
1 Pathways Model Workshop
   1.1 Goals
   1.2 Resources
2 Agenda

Pathways Model Workshop
teachingopensource.org

- Resource collection
- Mailing list
- Place for people to find their community
Workshops & BOFs

Presented at conferences for instructors

- Consortium for Computing Sciences in Colleges
- SIGCSE (Special Interest Group on Computer Science Education)
Rensselaer Center for Open Source Software (Troy, NY)

- Project-based work
- Students work for credit or stipend
Rochester Institute of Technology (RIT) (Rochester, NY)

- Academic minor in Free & Open Source Software & Free Culture
- Both technical & non-technical tracks
And community?
Some ideas:

- Identify clear path for new contributors
- Documentation, documentation, documentation...
- Clearly marking bugs “easy fix” (or equivalent) is handy
- List of nice-but-not-blocking-small-to-medium size features would be wonderful
- Project-sized hunks (suitable for team over course of semester) + mentor = WOW!
- Have an “academic liaison”
We had an unexpected interaction with a developer for a different project that was a dependency of the pizzadash project. We needed a version update of the node-dash-button api and were able to get the developer to update it with almost no hassle.
What we learned

- Abiding by a Style Guide
- Proper Debugging/Unit Testing Methods
- New Data Structures and Algorithms!
- Markdown
- Proper Documentation Techniques
- Communication via GitHub
  - Opening a pull request
  - Resolving merge conflicts
  - Coordination with other contributors
  - Making relevant commits, with proper commit messages
Has inspired some of us to try and mimic the objective of the project to promote the development of such educational resources for other programming languages.
Thanks!

Any questions?

You can find us at:

- @lintqueen (gina)
- @heidijcellis
- @GregoryHislop
- education@redhat.com
References

This presentation uses the following typographies and colors:

- [ ] http://www.slideshare.net/North_Bridge/2015-future-of-open-source-study
- [ ] https://books.google.com/books?id=806hBgAAQBAJ&pg=PA65&lpg=PA65&dq=age+of+open+source+contributors&source=bl&ots=O2NkdBfTSd&sig=o5aOXVtIZuaafEcNPb1aVYfK%E18&hl=en&sa=X&ved=0ahUKEwi0tpCpitDMAhWMNiYKHS_ZD6wQ6AEIKzAC#v=onepage&q=age%20of%20open%20source%20contributors&f=false