Volunteer Handbook

Ada is a community-supported organization, and we rely on the talents of volunteers to help us in reaching our mission to close the gender gap in tech.

Location
Moz
1100 2nd St
Seattle, WA 98101

When you get to the building, take the elevator to the 5th floor and ask the receptionist to direct you to Ada Developers Academy. Alternatively, from the lobby, text/call 206-939-5755 to let us know you've arrived and an Ada Staff member will come get you from the lobby on the first floor.

Contact Information
Brooke (brooke@adadevelopersacademy.org) is the Ada Program Coordinator. She handles most schedule updates and reminders for volunteers; however, she works very limited hours (2-3 hours/week) so if you need immediate response to an issue, need to cancel a scheduled time, or etc, you should contact either/both **Crystal (crystal@adadevelopersacademy.org)** the Program Manager or **Karen (karen@adadevelopersacademy.org)** the Sponsorship and Engagement Manager.

Table of Contents

Volunteer Handbook

Ways you can be involved

- Guest Lecturer - Speak to a specific topic or tool
- Teaching Assistant - Helping students problem solve/learn in the classroom
- Code Reviewer - Provide students with comments/code reviews either in person or remotely
- Application Reviewer - Review candidate application submissions and score
- Sponsoring Company Mentor - Provide regular professional/technical support for a specific student
- Ada Student/Alum Mentor - Provide emotional/technical support for a specific student

How to get involved

1. Complete the Volunteer Information Survey: http://goo.gl/forms/NpoVLLMQt2
2. Return a signed copy of the Volunteer Code of Conduct to contact@adadevelopersacademy.org

You should hear back from Ada staff shortly afterwards to schedule a no-obligation visit to the classroom so you can check out our program up-close!

Volunteer Handbook

Guest Lecturer

As a **Guest Lecturer**, you will provide training to students on coding skills/tools and/or soft skills.

Your role will be to:

- know your audience is a group of women who are new to coding--students began their cohort with very little to no experience coding; Ada provides them with 7 months of intensive classroom experience in Ruby, Rails, HTML/CSS, and JS followed by a 5 month industry internship; please define terms, do not assume that they know something already, and do not refer to concepts as something that "programmers should already know", as something that "you can't get a job if you don't understand", or as “easy”. Things that are easy to you may be difficult to someone else, and trivializing them trivializes a person who struggles with them.
- prepare an interactive presentation for our students that broadens their skill set or increases their awareness of topics related to the tech industry
- if your lecture is teaching a tool or skill, scaffold the lecture so that students are first presented with an intro to the topic and smaller problems; build to more challenging problems after guided instruction and walk through of how to solve the problem; providing plenty of simpler examples and problems is a perk
- field questions at the end of the lecture and provide resources for students who are interested in learning more

You bring expertise to our classroom that gives students the ability to learn more and do more. A diverse collection of guest lecturers allows each student to find specific topics that they are passionate about and also provide assistance to our instructors by covering topics they may not be familiar with/have the time to prepare for.

What to expect

- Guest Lectures happen on Tuesday afternoons between 1 and 5. We aim for 1-2 lectures each Tuesday.
- Approximately 20-25 students, teachers, staff
- With your permission, absent students usually request that we record speakers so they can watch them later. Generally we use QuickTime’s Screen Capture to do this and then upload the lecture to http://lectures.adadevelopersacademy.org, or we can just share them internally.

What Ada will provide

When you come for your presentation, we will provide:

- White boards
- Projector
- HDMI/Mini DisplayPort connectors

Format of your presentation

We understand presentations are highly individual, but here are some general guidelines to consider:

- Presentations should be between 30-60 minutes, including prep, the presentation itself, and Q&A at the end.
- We will block 1 hour in our curriculum for your presentation. You do not need to use all this time. You should definitely NOT talk for this entire time without interaction from/activities with the students.

Volunteer Handbook

Ada DEVELOPERS ACADEMY

Tips for being an effective Guest Lecturer

General Tips

- **Start and finish strong** - Begin with a bang. Listeners tend to remember what they hear first and last. Begin with a surprising statistic, a startling statement, a vivid anecdote, a rhetorical question, or a provocative quotation.
- **Address the "So What" Question** - Explain why listeners need to know about your topic—why they should care?
- **Offer a roadmap at the beginning of the presentation**
    - Share the structure of your talk with your listeners
    - Direct your audience's attention to the most important points you are making
- **Sustain audience engagement**
    - Break up your talk every 10-15 minutes to revitalize the audience's attention. At this break, ask the audience a question, give them a simple problem to solve, have them discuss concepts with a neighbor and then share back their thoughts, or have them try something on their own.
    - Use concrete examples and simple syntax. Speaking in theory/abstractly for a long time is hard to follow
    - Integrate active learning into your lecture.
- **Remember that DELIVERY is more important than CONTENT** - If your delivery isn't interesting, your content will be hard to process.

For Research-Based Presentations

- **Consider turning your research into a story** - The most common way to present one's research is to give it in the form of a research report. You identify a research question, a hypothesis, your methods, your findings, and your conclusions. Try something different: present your research as a story.
    - Explain why did you decided to study this topic.
    - Describe the struggles and difficulties you encountered.
    - Tell your audience about the discoveries that surprised you.
    - Describe the "ah hah!" moment when you suddenly understood your work's significance
    - Conclude by telling your audience what you discovered and why it is significant.

For Technical Tool Presentations

- **Provide opportunities for students to try the tool** - build in opportunities for the students to try the tool on their own. Our students learn best by doing, so your presentation should include time for the students to try out the tool.
- **Provide simple problems** - giving the students some sample scenarios/problems to solve using the tool you are teaching will scaffold the learning so that students can build toward more difficult problems.

Volunteer Handbook

Teaching Assistant

As a **Teaching Assistant**, you will be asked to help students during project time.

Your role will be to:

- know your audience is a group of women who are new to coding--students began their cohort with very little to no experience coding; Ada provides them with 7 months of intensive classroom experience in Ruby, Rails, HTML/CSS, and JS followed by a 5 month industry internship; please define terms, do not assume that they know something already, and do not refer to concepts as something that "programmers should already know", as something that "you can't get a job if you don't understand", or as “easy”. Things that are easy to you may be difficult to someone else, and trivializing them trivializes a person who struggles with them.
- provide expertise knowledge to students on why/when/how coding structures are used
- help students get "unstuck" by asking them probing questions, providing suggestions, and scaffolding the process of solving a problem
- provide specific feedback to students about their code and problem solving approach
- encourage and support students on their journey to being professional developers

Your knowledge and expertise as a seasoned programmer is invaluable to our students. We ask that you model your thinking to students by not just giving them the answers, but by asking them leading questions that help them find the solution with your guidance. Remember that our students are completely new to coding and that your support and guidance will help them achieve quicker and with more success!

What to expect

- TA during class hours of 1-5pm or off hours from 5:30-7pm
- Help students on a 1:1 or 1:few basis on project work, doing: code review, talking through approaches, or answering specific questions.

If there is down time while volunteering at Ada, you are more than welcome to work on projects of your own, but please ensure that students know that you are still available and that they are not "bugging you" when they ask for help.

Tips for being an Effective TA

Treat everyone fairly

- We all form bonds with people at different levels. Do not allow this to influence who you help or how you answer questions. Be available to all students and make it a priority to spread out your help.

Cultivate independence

- Demonstrate HOW to do something, and then ask the student to re-teach you.
- Make sure the student is the one actually doing the work (i.e. Don't just take over! Let the student “drive” the keyboard.)
- Offer additional examples/questions to “quiz” the student
- Aim for improvement and long-term growth—not perfection!
- Help them do the best work THEY can do (not the best YOU can do)

Volunteer Handbook

Be a good listener

- Listen to what the student is asking for help with. Make sure you know the student's current learning goals and current questions.
- Ask for explanations of answers—particularly incorrect ones. Try to lead students to identifying their own misconceptions when possible.
- Observe nonverbal cues (they might verbally say they get it, but do their body cues say the same?)
- If you hear the same question numerous times, feel free to do an impromptu lesson or let an instructor know so they can do an impromptu lesson
- Know when to answer student questions regarding social, academic, or personal issues, and when to direct the student to another resource.**

**Please inform Ada staff should you have concerns about a student emotionally or academically.

Offer praise and encouragement, but without being phony

- Acknowledge progress, even if there are still struggles/errors
- Never ridicule incorrect answers
- Share your own struggles and strategies

Provide constructive feedback

- **Demonstrate Positive Intention:** Give comments with care and kindness.
- **Describe, Don't Evaluate:** Feedback should describe, rather than judge, a student's work. For example, "You've broken this problem down into really good functions that eliminate redundancy of code and allows for specific code functionality testing" rather than "This code looks good" (which is a judgment and doesn't describe what makes the code "good".)
- **Be Specific, Not General:** Give specific comments rather than general ones. For example, "In this section of code we could eliminate some of the loops if we…" rather than "This algorithm could be more efficient."
- **Balance Feedback, Use the "sandwich" approach:**
    1. Say something that worked well,
    2. Offer a constructive suggestion for improvement, and
    3. Mention another aspect that worked well.

    For example, "I really like the UI for your form, but perhaps the controller could store the data using a…. Overall, the breakdown of the problem is good, though!"

    This approach will build a student's confidence levels while they are developing their skills.
- **Check Understanding:** Check that the student understood your feedback. Ask the student to paraphrase what you've explained. Sometimes if the student verbalizes it, it can either affirm understanding or lead to additional questions.
- **Answer the Question:** Your experience will enable you to see potential complications and inefficiencies in a student's approach or architecture before the student does. Don't deflect the conversation toward those unearthed issues. Work with the student to address their immediate concerns.
- **Pace Feedback:** Only provide feedback to students at a rate they can integrate. For example, work on perhaps one main issue per session and suggest one way to strengthen the code.
- **Cultivate Two-Way Feedback:** While part of your job as a TA is to provide helpful suggestions to students, you can also solicit their input on your feedback. For example, "What's working for you in my explanations? How might I improve my delivery to better assist you?"

Be OK with not knowing all the answers

- If a student asks a question you cannot answer, don't be afraid to tell them you don't know the answer and either will follow-up later or take the time to find the answer right then.
- Show students how you find answers when you're stuck, what resources are valuable to you, and how to evaluate the quality of potential solutions or explanations.

Near the end

- Review main learning points of the session

Volunteer Handbook

Code Reviewer

As a **Code Reviewer**, you will assess the quality of students' code and provide feedback to them. This can be done in person or remotely.

Your role will be to:

- know your audience is a group of women who are new to coding--students began their cohort with very little to no experience coding; Ada provides them with 7 months of intensive classroom experience in Ruby, Rails, HTML/CSS, and JS followed by a 5 month industry internship; please define terms, do not assume that they know something already, and do not refer to concepts as something that "programmers should already know", as something that "you can't get a job if you don't understand", or as “easy”. Things that are easy to you may be difficult to someone else, and trivializing them trivializes a person who struggles with them.
- read the provided assignment description and grading rubric
    - when students complete assignments, they will have options on the requirements of the assignment. All students should complete assignments at a ***bronze*** minimum, but they may complete assignments all the way through ***silver***, ***gold***, or ***platinum***
- complete an industry-standard code review of student’s code using the student’s github repo
- assign a "level" to the student's code -- ***beginner***, ***proficient***, or ***master*** -- for each of the categories on the rubric
    - expect that students will all start as *beginners* and more towards *proficient* later in their coursework
    - *master* should be reserved for truly exceptional work, and also be used as an indicator to students that they should consider moving up a medal level for the next assignment
- leave encouraging, as well as constructive, comments/suggestions/feedback to students
- allow students to improve their code; reassessing leveling is fine, even encouraged. Our goal is growth, not necessarily getting things right the first time.

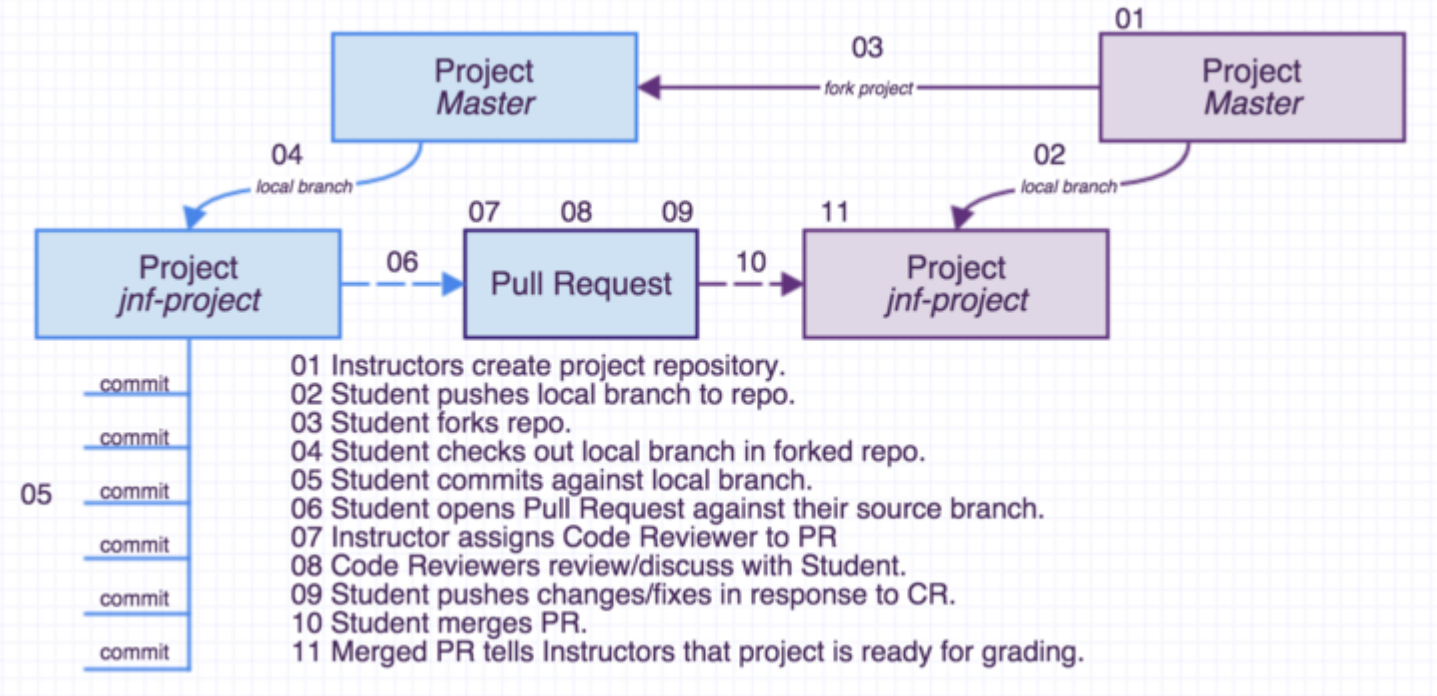
What to expect

- Within your availability, code review as much student work as possible each week using the process described below.

Code Review Google Group

Please join/use this group as a reference for specific questions about the code review process and specific assignments. https://groups.google.com/d/forum/ada-code-reviewers

Code Review Process



Volunteer Handbook

Application Reviewer

As a **Application Reviewer**, you will help evaluate candidates applying for admission to Ada Developers Academy.

Your role will be to:

- Review applications through a web portal, including resumes, technical questions, and short video introductions.
- Be timely about the review process.

Volunteer Handbook

Sponsoring Company Mentor

As a **Mentor**, you will provide 1-on-1 support to specific Ada student(s).

Your role will be to:

- know your audience is a group of women who are new to coding--students began their cohort with very little to no experience coding; Ada provides them with 7 months of intensive classroom experience in Ruby, Rails, HTML/CSS, and JS followed by a 5 month industry internship; please define terms, do not assume that they know something already, and do not refer to concepts as something that "programmers should already know" or as something that "you can't get a job if you don't understand"
- Provide 1-on-1 technical and professional guidance to a designated Ada student
- Meet with student at regular intervals, preferably once per week (at the start at Ada, later at the Job Site)
- Be available for non-mentee support in the classroom (this provides ability to meet more students)
- Be available for email or phone questions from mentee
- Know what you have to offer. A student may not know what they would like to do with their mentor, suggesting a few different options (code help, career advice, pair programming) can help them figure out what they'd get the most use out of

You create an emotional tie with our students and help support them on a deeper and very meaningful level. You provide students with regular support that encourages them to survive an emotionally intense career change.

Volunteer Handbook

Ada Student/Alum Mentor

As a **Student/Alum Mentor**, you will provide 1-on-1 support to a specific Ada student.

Your role will be to:

- Share your skills and knowledge.
- Understand that your tone and attitude will influence your mentee's experience.
- Provide guidance and constructive feedback.
- Support your mentee.
- Report any concerns to Ada Staff.

How to be a ROCKSTAR Mentor

- **Schedule regular check-ins with your mentee** - Coffee dates, meetups, text messages, or chat are all great ways to keep communication lines open and available. Sometimes mentees might feel uncomfortable "taking your time" so showing her that you are available and invested is important.
- **Be a positive role model** - A mentee can learn a lot from her mentor simply by watching how her mentor behaves in a particular situation. Good mentors will look out for, or even create, experiences and situations in which her mentee can become involved to learn new things. e.g. inviting your mentee to meetups or working on a learning project together.
- **Be genuinely interested in your mentee as an individual** - A good mentor will get to know her mentee personally, including her personal and career aspirations and goals.
- **Share your experiences and insights** - A good mentor shares stories that you feel are appropriate and helpful, but is conscientious as to how these stories can influence the mentee in both good and bad ways. Be open to sharing your mistakes and failures, as these are often where our biggest lessons are learned. This will also help your mentee be aware that challenges will arise, and the way you dealt with the situation might help her learn how to build resilience.
- **Ask open questions** - Good mentors ask open questions to aid in identifying needs, values and passions. It's also a great way to get your mentee to think through situations herself and draw out the consequences of the various choices or courses of action she can take. During these conversations, you should share your wisdom without making decisions for your mentee.
- **Act as a sounding board** - Mentees benefit greatly from the opportunity of having a good mentor listen to them. You should allow your mentee to explore her thoughts and ideas openly with you.
- **Provide a fresh perspective** - A good mentor will often provide her mentee with a fresh perspective on an issue. You will often have the clarity of distance from an issue or problem that's needed to provide objective feedback. You can also hold up a 'mirror' to the mentee. For example, when appropriate and when the relationship allows, you can help the mentee see what their behavior might look like to others.
- **Provide helpful feedback** - Not all feedback is helpful. You should deliver feedback in a way that will help your mentee gain insight to further develop specific qualities or skills. You should always ask for permission to give feedback. Giving unwelcome feedback can be detrimental to any mentoring relationship.
- **Acknowledge achievements** - Highlight for your mentee any achievements she might have forgotten to help build her confidence.

Volunteer Handbook

Volunteer Code of Conduct

Vision
Ada Developers Academy is designed to increase the number of skilled software developers while redefining and reinventing the way we prepare individuals to be successful in information technology fields. By creating an intensive top-quality state-of-the-art learning community that is inclusive, positive, and enabling, we aim to directly address both the information technology labor shortage in Washington State and the gender imbalance in the software industry.

Mission
Ada Developers Academy prepares students to succeed in careers in software development by training them through a year-long, tuition-free, comprehensive program that teaches cutting-edge web technologies, provides high-caliber internships, and links students with hiring companies.

Inclusivity
Ada Developers Academy is designed to be an inclusive, positive learning environment. Everyone is welcome, regardless of sexual orientation, disability, physical appearance, education, age, race, or religion. All women (cis and trans) and people with non-binary gender who feel a part of women's community are encouraged to apply.

For the safety of our students and the integrity of our program, we ask that all volunteers read and sign this code of conduct.

As a volunteer partner of Ada Developers Academy, **I will**:

- Be conscious of the fact that everything I do, directly or indirectly, has the potential to reflect upon the Ada Program as a whole.
- Accurately represent my professional qualifications, experience, and knowledge.
- Treat everyone with dignity, worth, respect, and fairness regardless of sexual orientation, disability, physical appearance, socioeconomic background, education, age, race, or religion.
- Use positive techniques of guidance, positive reinforcement, and encouragement rather than competition, comparison, or criticism.
- Consider the meaning and impact of my own ethnic and cultural background, gender, class, age, and sexual orientation and their effects on my views of society, women, and women in technology.
- Take appropriate precautions to distinguish between personal, organizational and societal views.

As a volunteer partner of Ada Developers Academy, **I will not**:

- Show favoritism towards any student.
- Humiliate, ridicule, threaten, or degrade anyone involved in the Ada Program.
- Generalize women using gross stereotypes about women (catty, hard to deal with, e.g.) or women in technology.
- Use derogatory, patronizing, judgmental, insulting, humiliating or ridiculing words or phrases for women/girls in any form.
- Engage in or condone any sexual harassment or other forms of harassment that contribute to an uncomfortable, ineffective, or hostile environment.

I agree to abide by the Code of Conduct outlined above in order to best strive for social change in line with Ada Developers Academy's vision, mission, and inclusivity standards. I understand that failure to adhere to these guidelines may result in dismissal as a volunteer.

_____	_____	_____
Printed Name	Signature	Date