Code Fellows

Teaching Assistant Handbook

A guide to the inner workings of being a Code Fellows TA

Chapter 1

Welcome to Code Fellows!

Greetings and Congratulations!

Welcome to our team of world-class teaching assistants, we are glad to have you on board! This handbook is meant to help orient and teach you how and why we do things the way that we do. This handbook contains best practices and linked resources, it should be read before, during, and after you assist for a course.

Use Outline View

When using this document, we recommend taking advantage of the Document Outline feature **(View > Show document outline)**. This feature will help you navigate the handbook as you're scrolling its pages.

Also, view the pages in Print layout (**View > Print layout**) so that all page numbers, headers, and footers are visible.

The TA Guiding Principle

A guiding principle is a statement that helps to guide our decisions and the way in which we approach training. As a TA, your guiding principle is to lead with humility and empathy to create safe and inclusive environments where people learn and lives are transformed.

The Code Fellows Way

Based on educational theory (***andragogy**: the method and practice of teaching adult learners*), industry standards, professional best practices, and LOTS of iteration, we have identified effective instructional methods.

Code Fellows is centered upon **empathetic teaching**. We believe empathy is the most important skill you can practice and will lead to greater success personally, professionally, and as a bonus, make you happier the more you practice.

You were hired as a TA because we detected some degree of empathy in you already. In order to help you grow this skill, please read the following articles:

- Your Most Important Skill: Empathy by esteemed developer, Chad Fowler.
- Learning Styles and Experiential Learning Cycle by David Kolb

We are aligned with student goals: Our students have big plans and goals, typically involving career and life changes. The position of TA is a leadership role that we actively encourage you to take ownership and command of within our program, which allows you to take accountability for the experience of our students.

We utilize a non-traditional instructional style: This style is more like mentoring, or coaching, or a personal trainer. We expect students to learn by doing, and to teach each other. Class time is discussion-oriented, to unblock student progress and thinking, to address common hazards, and lightly introduce new topics. Stacked learning is our overriding methodology, familiarize yourself with this concept here.

We are ruthlessly focused on practical skill: Code Fellows is in constant dialogue with industry partners to stay current with the abilities and aptitudes that are needed **now**, and to read the winds on technologies, concepts, and frameworks that are gaining traction. As a result, our curriculum is in a state of constant evolution. Pair-programming and team projects in our courses are designed to emulate real-world work environments. We emphasize professionalism, accountability, and communication skills alongside our technical curriculum,

striving to produce graduates who can immediately provide value for employers and become key team players amongst their colleagues.

Why We Do This

We have a singular goal: Great people into great jobs! People come to us, looking for fulfillment of their hopes and dreams and they are making a massive investment of both time and energy to be here. It is deeply rewarding to help people learn new skills that serve to improve their lives.

What We Do

Students will expect you to know about the inner workings of Code Fellows. The FAQ page here is a great place to start.

Don't worry if you cannot answer a question. Take the lead by helping the student find the answer, and learn it yourself, so you can answer next time. If a question is related to course discounts and/or is regarding specific aspects of the admissions process then please have students email admissions@codefellows.com to get the correct answer(s) from the Admissions department. If a question is related to billing, invoices and/or other financially related issues then please have students email finance@codefellows.com to get the correct answer(s) from the Finance department. Always be willing and ready to learn from your interactions with students! The best educators are constantly learning new things.

TAs should understand our admissions processes: students apply via the website, do an initial phone interview, prepare for their entrance test, take the test online via Canvas LMS), and are enrolled if they achieve the minimum passing score of 80%. Students with more advanced skills are able to skip courses by testing into the intermediate or advanced courses and successfully completing an in-person interview.

Familiarize yourself with our Course Offerings and Course Catalog, as well as the policies contained therein, found on our website:

Course Offerings

Course Catalog

TAs should know exactly what our Code of Conduct says and be able to answer student questions about what we consider harassment and how to treat others, as well as plagiarism, proper use of citations, open source practices, etc.

Chapter 2

Getting Started: The Team, Onboarding and Initial Training

The Team Setup

The TA Team consists of two types of TAs, graders and lab support TAs. Each type of TA is an important part of the team and will help our students succeed during their time at Code Fellows. All TAs are a part of a pool that provides resources, guidance and encouragement to the students. In either role a TA will gain valuable experience, whether it be debugging and troubleshooting skills to code review.

There are two exceptions to the way the team is set up though. Both the 101 and 102 courses have dedicated TAs that will focus completely on those students. This is to ensure that these students, who have mostly never written code, get help with the basic fundamentals that are taught in these courses.

Introduction to Our Tools

The team communicates in the following ways: via email, Slack and Google Docs. In the Code Fellows Slack workspace, each class has a private Slack channel for the entire class (by invitation from Code Fellows Developers admin account). Each class also has a private Slack channel for the instructor and TAs (again, by invitation). TAs are welcome to join and participate in any other public channels in the Code Fellows Slack workspace. In addition to Slack's web

functionality, it also has a mobile app, and Mac desktop app. **Please ensure that your Slack profile is up to date with a professional looking picture and includes some bio info to help students get to know you.**

Google Docs, Google Drive, or other tools may be used within a course, depending upon the instructor.

The team uses GitHub and Canvas for managing courses and students. The Principal Teaching Assistant will add you to the TA Team in the GitHub Code Fellows Organization. Classes will have repos on the master Code Fellows account that accumulate the code written in class, plus contain other code samples supplied by the instructor. TAs do not have admin permissions for this repo, but can contribute via Issues and Pull Requests. Students generally will submit assignments by links to their GitHub repos of coursework. GitHub Gists are also used to share code snippets at times.

Our learning management system, Canvas, can be overwhelming for first-time users thrown at it with no guidance. Moreover, we use it in a very specific way, to complement the assignment review we do in GitHub. Many specific aspects of Canvas require their own introduction (specifically important: using the Gradebook and SpeedGrader). You can learn a lot by just clicking around and exploring Canvas, and it is encouraged that you set aside some time to do so. Investigate each of the following components:

- Syllabus
- Roster
- Assignments
- Quizzes
- Gradebook
- Discussion forums
- Supplemental files

You will be invited to your pooled Lab Time/Grading course(s) Canvas pages, as well as the class and team Slack channels, the Principal Teaching Assistant is your first point of contact for this. To get set up properly, start by setting your timezone (the default is Mountain time, not

Pacific!). There are a variety of notification settings that can also be applied, it is important to configure settings so that they are actually useful to you.

Lab Support TA

Lab support TAs are present to support students that are in lab time and during project week. For daytime classes there are two different shifts that will put a TA either supporting the morning or afternoon labs for the entire lab session and for a portion of the other lab session. As a Lab TA you could at times be supporting a 401 level class that you did not complete, however there are ways in which you can support that class. Having a good understanding of data structures and algorithms and general logic translates between all languages and can be used to help all students.

Grading TA

Grading TAs are responsible for the grading of assignments using Canvas and GitHub. Graders will be responsible for grading a number of students and they can be from multiple classes. It is very important for graders to maintain the 24 hour turnaround on assignments, this helps both the Student and the instructor know where the Student stands in the course and helps to identify possible action plans for the Student.

Training Session

Prior to beginning work as a TA at Code Fellows, all new TAs must complete their TA Training course on Canvas.

Training includes:

- A review of this handbook followed by a short quiz.
- A review of the Grading and Expectations document followed by a short quiz.
- A review of the Code of Conduct document followed by a short quiz.

- Links to helpful Canvas tutorials on:
    - Using the Speedgrader.
    - Taking attendance.
    - Understanding the grade book.
- A 15 minute video where two of our instructors talk you through grading a student's assignment submission.
- A short TEDx talk explaining the importance of a growth mindset.

Grading TAs will receive individual training on standard grading practices from the Principal TA. This will occur during the first week of working as a Grading TA and is seperate from the TA Training course found in Canvas.

Grading training will consist of a review of the grading expectations and best practices with the Principal Teaching Assistant, as well as a paired grading session with the Principal Teaching Assistant. More training sessions may occur depending on the course level and assignments available to grade.

Shadowing

Though not always possible, it is encouraged that new TAs spend a portion of a lab session shadowing an experienced TA to get a feel for the role and have an opportunity to observe techniques used by experienced TAs. Student interaction operates on many levels, and each moment can be focused on a variety of topics: specific course content, the details of an assignment, the flow of the course, an issue with workflow (such as a Git or IDE problem), the mindset of a student, just shooting the breeze, and so on. By shadowing, a new TA gets to observe these interactions and consider how they would have engaged with the student. After shadowing, the TAs can reflect on the session and discuss the student interactions that took place.

Course Audits

TAs have the opportunity to audit courses in special situations, in order to most effectively serve students. A course audit is a way of enrolling in a class in exchange for feedback and TA support of students in lab time. TA's may audit courses with recent major revisions to the curriculum, or under other special circumstances. This must be approved by the Principal Teaching Assistant, who is the primary contact for questions about auditing.

As a TA auditing a course, it is important to treat the audit as if you are an enrolled student in the course. Be on time for lectures and keep to the required attendance standards. Maintain a high level of professionalism, and be respectful towards the instructors and the students enrolled in the course. If for any reason you need to miss a class or part of a class, notify both the instructor and the Principal TA.

While attending lectures, keep any questions you may have during lecture time tightly within topic scope. Contact the instructor outside of lecture if you have more complex questions. To optimally position yourself to help students understand the problems they encounter during lab, work through the labs a day or two ahead of time. Submit your lab work on Canvas for review. When creating repositories for your work, you will need to set the visibility to Private and invite the instructor(s) and grading TA to be collaborators.

Outside of lecture, you will serve as the first point of contact for students working through assignments, and will be expected to offer help and troubleshooting insights, even if you haven't yet fully completed the lab yourself. Ideally, you can work ahead, and have labs done by the time students access them. When that's not the case, you will still offer meaningful assistance, in guiding students through the problem-solving process, or connecting them to helpful resources, like online documentation, the right segment of the lecture video, or getting them in touch with the instructor to resolve the issue.

During your time auditing the course, you will daily provide feedback on lectures, labs or course documentation to instructional staff. Maintain professionalism with proper feedback channels, only sharing critical feedback with instructional staff. This will often include meetings

after lecture time to review the lecture. Your input and feedback is important for the whole team and will be used to help improve the course for the students, instructors, partner schools and of course, yourself and other TAs. With that, it is important that you are focused on the course.

Instructional Team Sync

Before a new class starts, there will be an instructional team sync that happens usually the week before the course. This sync will consist of any lab TAs, grading TAs, Assistant Instructors, instructors and the Principal TA. The purpose of these syncs is to make sure proper communication is established within the team, bring up any potential student concerns and set expectations for the course.

Remote TAing

While we do not have dedicated remote positions, there are times when remote TAing can happen. In these situations we use Zoom for lectures and Remo for lab times to create the best possible virtual environment possible. In these situations you must keep the same level of professionalism expected on campus while in a virtual environment. The current Remo event space for Code Fellows HQ can be found here.

Within the Remo virtual environment, we have developed a Help Queue application to better manage when students need help. This guide has setup instructions and a quick overview of how the help app works. It is important to encourage students to write helpful and descriptive help tickets.

During a required remote situation, the schedule of current class lecture and lab times might be altered. When this happens, it is usually set to have all classes on the same schedule which might require a change in your current scheduled times. For instance, all classes could be adjusted to be morning lectures (9am - 12pm) with afternoon labs (1pm - 6pm). If this occurs you will be added to the additional class Slack channels.

There are a few additional things to take into consideration while helping students in a remote environment. The lack of physically being in the same workspace can allow students to isolate and not seek help when they need it. It is even more important to keep open and clear lines of communication, utilize the course Slack channel and check in with students often. Problem solving can become more difficult. Some students may have issues with their microphones or webcams, the possible technology issues are endless. Stay patient and be encouraging if a student is having these problems. If it seems overwhelming, it is always ok to reach out to the instructor and see if they are available to help.

Working In Remo

Our online Virtual Campus environment is based on an event platform called Remo. Each floor has multiple workspaces that offer students the option to collaborate with each other and the instructional staff. In addition to the tools built into Remo, we have implemented a Help Queue that allows students to create help tickets that are easily seen by the instructional team. Upon onboarding, the Principal Teaching Assistant will give you access to the TA side of the Help Queue and will be available to answer any questions you may have.

Using the Help Queue: Tickets should be handled in the order that they come in. The only exception to this is when there is only one language-specific lab TA available to handle a 401-level ticket. For example, if you are the only Python TA and there is a Python specific ticket 4th in the list behind a few 201, 301 or another 401 ticket. It is important to click on the “Pending” button to show others that the ticket is “In Progress” with you. If for some reason you are not able to help the student reach the solution, you can select the button again and put it back into the “Pending” status. If this occurs be sure to inform another TA or the instructor of what was attempted and what the current situation is. When the ticket is completed, select the ‘x’ in the top right corner to close the ticket.

What if I am not sure I can help out with a 401 ticket? If you aren’t sure if you can help out with a ticket from a 401-level class that you didn’t take, leave the ticket in the “Pending” status. Then join the student at the floor and table indicated in the ticket, and do a quick check in. If you find that you are able to assist them, move the ticket into “In Progress”. If you are unable to

assist them, reach out to a TA that has been through the course, the instructor or the Principal TA, this will help ensure the student gets the help they need. It is important to let the student know that they are important to us, and we are here to help and support them in whatever way we can.

If at any time you are unable to locate a student who has submitted a ticket within Remo, reach out to them directly through Slack to see if they are still requiring assistance. If they do not need help or if you do not receive a prompt response from them, send them another message letting them know that you are closing the ticket and that they will need to open a new ticket if they are still needing help, and then close the ticket.

A few points on good practises while working in Remo:

- Have your camera on if you can. It is important for the students to see us and know who we are.
- Check in with students in the order of the Help Queue Tickets.
    - If you aren't sure if you will be able to assist them sufficiently, leave the ticket in "Pending". You can move it to "In Progress" if you are able to assist.
    - If you are unable to assist them, reach out to other TAs in the TA Slack channel, contact their instructor or the Principle TA to ensure the Student gets the help they need.
- The 15 Minute Rule applies to TAs as well.
    - Give the student a work plan that will help get them to where they are going, and tell them they may open a new ticket if they remain stuck.
    - If you are not sure of a path to get to the solution, escalate the problem to the instructor. If there are no other open tickets, sit in on the troubleshooting so you know in the future. If you can't sit in, check in with the student and instructor at a later time to find out the solution.
    - When there are no tickets in the Help Queue, take a moment to check in with students you've previously helped. You might troubleshoot some more or to celebrate a solution, either way checking back in with the

student will let them know that they are important and that we care about their success.

- Be patient. The remote environment brings new and unique challenges to debugging. Try to maintain a calm voice and remember to listen to the student.
    - The student should explain what their problem is in as much detail as they can.
    - Ask the student leading questions if needed, “What is the expected outcome?”, “What have you tried to get the desired outcome?” or “Can you walk me through your code?”.
- Require students to use the Help Queue within Remo to get your assistance. If a student directly messages you, let them know you would be happy to help and have them create a Help Ticket (even if there are no tickets in the queue).

Administrative Policies

Compensation: The primary point of contact for your TA employment agreement is the Principal Teaching Assistant. For your convenience we prepared a Frequently Asked Questions document, to help you understand our expectations around time keeping, time off, and payroll. You will find this **here**.

What if a TA gets a job while a course is in session? It happens, and since most of our TAs are also alums, this actually makes us extra pleased for you! We ask that you please give the instructor, the Principal Teaching Assistant, and Campus Director as much notice as possible so that we can arrange for a replacement.

How to handle student concerns and grievances: If it is an issue of an academic nature, strive to confer with the instructor first. In the event it is an academic concern that would not be appropriate or helpful to report to an instructor, please report to the VP of Education. For all other issues, report to one of the following, depending upon the nature of the matter: the Campus Director, Principal Teaching Assistant, Director of Admissions, or direct the student to do so directly.

Chapter 3

Ideals in Action: Definitions of Success as a TA

As a Teacher

If, as a TA, you notice a student struggling due to pre-work not being complete or if the student did not adequately learn from it, notify the instructor either in person or via Slack immediately. The sooner these issues are addressed, the better.

When working with students on code, DO NOT simply take over their laptop and start typing. It is extremely important that each student maintains agency over their tools and their work. Think of it as a pair-programming exercise with the student as the driver, and the TA the navigator. Yes, it will take longer, but it also builds a lot of important skills in listening, thinking, typing, and coding.

Understanding the components of course structure

Assignments (reading and coding assignments):

- Assignments are structured by Day or Class Number.
- TAs should help the instructor ensure that dates are accurate.
- TAs have the ability to publish assignments, but this should be left to the instructor unless otherwise assigned.
- TAs should help the instructor keep to an appropriate schedule for publishing reading and coding assignments (often, reading assignments are published a week in advance, whereas coding assignments tend to trickle out more slowly).

- TAs should ensure there is a solid mental road map for the course; anything that is unclear to you will probably be unclear to the students, too, so communicate with the instructor about this.

Quizzes:

- TAs may be asked by the instructor to write quizzes.
- Quizzes are used to reinforce concepts from lectures.
- Set quizzes to autograde as much as possible (multiple choice, True/False, etc.).
- Allow the students unlimited retakes.
- Use their highest quiz score as their grade for that assignment.

Attendance:

- Attendance will be recorded daily in Canvas.
- The Instructional Team as a whole is responsible for taking attendance.
- Students must be present for 90% of class sessions to pass a course.
- For daytime courses if attendance has not been taken by 11:30am then a reminder will be sent to the relevant team Slack channel.

How to be part of a successful Project Week: Students will work in teams leading up to a pitch day, in which students pitch potential projects to contribute to for the remainder of the course. Once a set of projects is chosen, they should be scoped for a three to four person team over a week. TAs are encouraged to be part of this process. It is vital to work with students on the projects so that they can show a minimal set of functionality and demonstrate that they understand the concepts taught in the course. Students will tend toward ambition, it is on us to create realistic expectations. Minimal Viable Product (MVP) is the initial goal.

As a Team Member

Always reflect and then communicate your thoughts and observations to the instructional team:

- What went well today?

- What can be improved for tomorrow?
- What are you doing to enhance your performance as a TA?

Regular teaching assistant meetings will be led by the instructor, where you'll report on how the week went from each person's perspective, discuss any students needing extra attention, and whether grading is up to date. If you haven't had at least one team meeting in any given week then politely bring this up with your instructor.

As a Human

Empathy!

Be yourself. There is a reason you have been recognized as someone who should be "on the bus". Students want to hear from you, your experience, and your opinions. Let your personality shine.

Don't pretend to the students that you know more than you do but also be confident so they have an anchor to depend on when the coding gets tough. Show them how you find answers to tough questions, so they can learn how to learn. Put the needs of your students above your own and remember that you are to lead them by serving them and hopefully provide a model for them to follow in the future. You are here to help them achieve their goals.

Chapter 4

Operations: Responsibilities & Expectations

Overview of Daily Activities

The listing of TA responsibilities in this section of the handbook will consider an example day in a variety of divisions. Depending upon the course and the needs of the instructor, an individual will likely only have duties in some subset of these timeframes: class time (before, during, or after), lab hours, and all other times.

Daily expectations of teaching assistants, in general:

- Be present and be engaged to help students as needed.
- Be available and approachable at all times during labs. Even if you are with a student and therefore unavailable you can still project a feeling of approachability.
- Grade and comment upon assignments within 24 hours of initial submission. Final Project Week assignments are graded by the Lead and/or Assistant Instructor (if the course has been assigned an Assistant Instructor).
- Understand the instruction plan of the day, and how it fits into the plan for the week. TAs should always know what is going on.

BE ON TIME. Timekeeping is done through the TSheets application. Any issues, such as the services being down or incorrect clockings, should be noted in the application and through email with the Principal Teaching Assistant. If you have an issue that is going to cause you to

be late or miss a shift, contact your instructor and hiring manager as soon as possible, via Slack or another agreed-upon method. The shift schedules are as follows:

Daytime Lab/Grader TA:
    Morning Lab M-F: 9:00am - 2:00pm
    Afternoon Lab M-F: 1:00pm - 6:00pm
Nights and weekends Lab/Grader TA:
    Mondays and Thursdays: 7:00pm - 9:30pm
    Tuesdays: 8:00pm - 9:30pm
    Wednesdays: 6:30pm - 9:30pm
    Saturdays: 1:00pm - 6:00pm

Daytime TAs are scheduled up to 30 hours per week and Nights and Weekends TAs are scheduled up to 20 hours per week. Both are allowed a 10 minute break while on the clock and an off the clock lunch break. All requests for time off should be requested through the timekeeping service, as well as through email with the Principal Teaching Assistant.

TAs will occasionally be assigned other duties while on-site and in a shift at Code Fellows, such as cleaning whiteboards, assisting staff in facilities support (rearranging a room, or getting it straightened up prior to an event), or other similar tasks. This is coordinated with the Principal Teaching Assistant. Always make certain that the instructor for your course is aware of these activities.

Preparation helps TAs to be more effective, therefore it is strongly recommended that the TAs work through the class assignments and code their own versions, in advance, unless provided with completed examples. This provides the TA a ready-made and familiar body of code to show students as an example and gives them specific insight into what the students are seeking to accomplish with each assignment. By being able to show students multiple working versions of code, we directly illustrate to them that there are multiple approaches to solutions.

On a weekly basis, your performance will be reviewed by the instructional staff. Here are some examples of things that will result in a better review, and things that won't:

You'll get a better review if you exhibit the following competencies:

1. **Professionalism:** examples include proactively checking in with students and exhibiting positivity.
2. **Craft:** examples include being able to effectively and accurately assist students on your own, giving good mentorship, having good time management skills, giving good feedback, and knowing when to ask for help.
3. **Growth Mindset:** examples include showing empathy and being able to give and take good candid feedback.

The following will result in the opposite:

1. Showing a lack of professionalism by: being late on grading, not being engaged with students, not communicating well or early enough with the instructor, too many absences, or not being prepared for the day's course agenda.
2. Showing a lack of attention to quality by: grading without giving feedback, grading inconsistently.
3. Showing a lack of leadership.

Upon a first poor performance review, the Principal Teaching Assistant will sync with the individual TA. A performance plan will be developed, and it will be expected of the TA to follow that plan. Follow up syncs will be scheduled weekly and performance will be reviewed. Multiple reviews showing poor performance, or failure to follow the developed plan will result in the release of the TA.

If you are a graduate of Code Fellows, you will be sent a monthly job placement survey, just like every other Code Fellows graduate. It's very important that you submit this report accurately every month. If you are a TA or know you'll be a TA in the upcoming month, choose the option "Employed at Code Fellows In-Field" for your placement status. In subsequent months, choose the selection that most accurately reflects your placement status (i.e. if you're TA-ing again, choose the same option, if you stop TA-ing and are seeking a job, choose that option, etc). For your Title, use one of the following options that best applies exactly (copy/paste):

1. Lab TA Daytime
2. Lab TA N/W
3. Grading TA Daytime

4. Grading TA N/W

Lab TA

Circulate among the students regularly. Take care to not get bogged down working with an individual student, utilize the 15 minute rule and escalate to another TA or instructor, sometimes a fresh set of eyes can find the problem. It is your responsibility to help as many students as possible. Instructors might call a TA meeting during lab time. Be available and approachable at all times during labs. Even if you are with a student and therefore technically unavailable you can still project a feeling of approachability so that the other student knows that you will certainly accommodate them once you become available again.

DO NOT wear headphones or earbuds during lab time. It is important to maintain a sense of approachability at all times.

When helping students troubleshoot a problem it is important to guide the student and not just solve their problem for them. This will help the student learn to use the resources available to help with future problems they run into. When a student comes for help they should be able to tell you what the error message says if there is one or what the expected outcome is, what they have tried and what resources they have researched to find a solution. This will help the student in learning to “speak developer” and articulate their problems.

As you are guiding a student to a solution, have them open any documentation you are suggesting, it is not as helpful if you are opening up the documentation on your machine. Utilize breakpoints and have them talk through what their code is doing, this can bring upon “A HA” moments for students and encourage the use of breakpoints in the future.

If you find yourself working with a single student for more than 15 minutes and have not been able to find a solution or a path to a solution, you should escalate the problem to the instructor. If there are no other students waiting for assistance, work with the student and instructor so you also know the solution to that problem. If the instructor is not available, utilize the lab Slack channel and/or the TA slack channel to see if anyone has run into the problem before.

Grading TA

Our goal is to prepare our students for job readiness. For us that means graduates of the 401 class—not just finishers. Keeping our placement rate high in the market requires that being a "Code Fellows Grad" equals a high standard to hiring companies.

We want them to test into the best class for them. They don't have to run through every one of our programs and our goal isn't to push a specific percent through to the next program. Over the next year we'll see students at the end of each class exit into industry jobs. We will be able to provide some additional guidance to the best role for them based on the class (internship, Junior Web Dev, etc). This means that we're different from most academic institutions that provide a certificate or let people "graduate" and pass them along.

Each course is designed to prepare students for the next course. However, it doesn't mean that all students will be ready. Often, 20% of the students won't be ready to pass without further prep because they are coming in with less knowledge and experience than other students. We want to help both groups, and to do so we need to set realistic expectations for them.

When we hear from hiring companies, they don't just want people that are trained in tech. They also want candidates that are a good culture fit with their businesses. A well-rounded graduate can show:

- Meaningful life experience
- Diverse thinking
- People skills
- Leadership
- Problem solving capabilities

Our partners ask: "Who are the top 5?" They want to know from both a technical and soft skills perspective. Don't underestimate your ability to help students be job ready on that basis as well.

Grading is a hugely important part of Code Fellows, as it is important to ensure that students have a good understanding of how their overall grade will be determined, according to the rubric. It is our primary means of providing feedback to the students, and these interactions are essential. Students want to know, and ***need*** to know how they are doing in our classes. If there are problems, we want to be sure that they are identified and addressed rapidly.

All assignments are to be graded and commented upon within 24 hours of the due date, or 24 hours from the time of submission if an assignment is turned in late. Be aware that this 24 hour turnaround does not apply to resubmissions. Resubmissions should be graded as soon as possible and with the same level of care as the original submission. If there is an abundance of resubmitted assignments it is suggested to grade the students who are furthest away from the passing mark first, this gives them and the instructor a better idea of where they stand in the class. This level of feedback is exactly why students come to take our classes.

Every single assignment should receive an individualized response, whether a sentence or two, a paragraph, links to helpful resources, or the like. Strive to give students encouragement and to acknowledge their efforts as frequently as possible. We want our students to know that we're here to help them, and that we appreciate their dedication to their work.

To grade reading assignments first read the student responses thoroughly and carefully. If a student asks a question, BE CERTAIN to answer it in your response, if you do not know the answer inform the instructor. If a student makes an observation that merits a reaction, then react to it. If a student seems to have misunderstood something, provide clarity. Encourage students to be thoughtful, thorough, and reflective in their responses. While avoiding being hypercritical and overly pedantic, help students to use the right terminology and organization techniques to express themselves clearly. This is part of guiding them toward well-rounded professional competency.

Code assignments are typically submitted as merged GitHub pull requests. Your response should include inline comments on the pull request, either indicating issues with the code or highlighting exceptional work. Within the comments section in Canvas you should summarize the inline comments and provide any other feedback pertaining to the submission.

Follow these guidelines:

- Review assignment requirements.
- Unless the assignment is small, it is best to clone the assignment and run it to see if it works as expected. If it does not, comment to that fact and require resubmission for grading. Unfortunately, students will submit faulty code in order to meet the deadline. This allows you to quickly filter out such submissions.
- Review the assignment in more detail. Make sure each requirement for the assignment is met and, in the case of pass or fail only grade, return assignments that do not meet the requirements to the student for correction and re-submission.
- Review the code for common errors and point them out in a pull request or via comments (in Canvas, we do not recommend making the comments on the student GitHub repo).
- Try to find something encouraging to comment upon to temper the corrections made to assignments. Challenge strong students and encourage the weaker ones.
- If an assignment needs to be resubmitted and has not been for a couple of days after the due date and initial review, mark it as incomplete.
- If a student is consistently making the same mistakes, provide them with a Gist, including a copy of their code with your comments and a copy of their code that you've refactored the way you would have written it. This takes much more time to produce and deliver than common pull request comments.
- **When plagiarism is suspected or identified, seek to identify the suspected source or otherwise be prepared to explain your suspicion, and then consult with the instructor as soon as possible. Plagiarism is a Code of Conduct violation and can result in release of the student.**

General Guidelines on Grading

- Pass: 90%+
    - Quality of work.
    - Assignments turned in on time.
    - Contribution to Team.
        - Are they helping other people in the cohort?
    - Leadership/attitude.

        - Are they setting a good example someone would want to hire? On time, positive contribution.
- Fail: <89%
    - Quality of work is just OK, or lacking in completeness or presentability.
    - Assignments are spotty or late.
    - Contribution to Team.
        - Doing only their own work, not participating or helping.
    - Leadership/attitude.
        - Are they setting a good example someone would want to hire? On time, positive contribution.

Complete Submissions

When you are grading, you should only grade submissions that are complete. For a submission to be deemed complete:

- The student must answer the required submission questions, which might include:
    - How long did you spend on this assignment?
    - Describe your process of completing this assignment?
    - What was the hardest/easiest part of this assignment?
- The code must meet professional criteria as demonstrated in the class, so it is expected that the code will:
    - Build.
    - Pass a linter.
    - Pass a style checker.
    - Pass all tests as appropriate to the level of course.
    - Have a completed README

If a submission is incomplete (in any way), 0 to 5 points can be assigned after a cursory look over the code to determine how much effort was applied.

Assignment Rubric

Points are granted to each assignment based on how the submission meets the following criteria:

	Met all assignment requirements	**Idiomatic style used**	**Proper git workflow utilized**	**Other adjustments**	**Total possible**
Lab assignment	Up to 6 points	Up to 3 points	Up to 1 point	-20% if late, -2 to +2 points for effort	**10 points**

Career Coaching Assignments

Career Coaching assignments are a pivotal piece of the students future success. It is important to spend time accurately assessing completion of these assignments. If students are missing these assignments, reach out to them to encourage them to complete the assignment, as well as notify the instructor of the missed assignment.

We are not all resume experts, we do however know the technologies that are used during the courses. When reviewing students submitted resumes, take the time to check for formatting issues, grammar mistakes and spelling errors, including things such as JavaScript, CSS, etc.. You are of course welcome to suggest things to the student that they may have forgotten, if you know they used something in a project, you can encourage them to add that. Use the Materials Review SOP to help ensure the student is following the guidelines for their resume.

Code 201 Assignments

A few submissions through a Code 201 course are done through forking a repository, and then submitting a pull request for the submission on Canvas. As the grader, once the submission has been fully graded, you will need to then close the pull request. Any re-submits will be done through a new pull request and closed out once graded. All closed pull requests should have comments regarding the grading as normal.

Late Work and Assignment Resubmission Policy

- At the time of the submission deadline:
    - Assignments that are not turned in or are substantially incomplete will be graded as **0 points**.
- After the submission deadline:
    - Any assignment previously earning at least 2 points can be resubmitted for regrading with **no penalty**.
    - Any assignment initially awarded 1 point or less can be resubmitted for regrading and will incur a **20% penalty**.
    - For students transitioning to an immediately-following class, work must be completed by 5:00pm Tuesday after the end of class. Contract and deposit must be completed the following day.
    - Resubmissions are allowed at any time prior to the start of the next Project Week.
    - For students who need more time after the course, they will email a plan to the instructor with what's needed to be done and target submission dates. Instructors can approve the plan if it looks reasonable, and let admissions know.
        - For 201 and 301, we should communicate this as being a strictly "as-needed" policy.
        - Students must turn in any incomplete work prior to being accepted to the next Code Fellows course in the program.
- For handling student work that's late due to illness or pre-arranged absences, please see the State Catalog, Section 8: Makeup Work.

Fudge points allowed for effort

- If someone spends 15 minutes, and hits just the minimum, you can optionally take off up to 2 points. Include a comment encouraging them to push their learning further by digging deeper into the task at hand.
- If someone spends 8 hours, and makes clear progress but does not meet all the criteria, you can add up to 2 points.

- If someone puts in meaningful effort (at least 3-5 hours) and hits stretch goals, or elaborates on the solution independently, up to 2 points can be added as optional credit.
- Sometimes, experimentation and attempting different approaches is more important than getting the right answer.

A great submission follows good idiomatic code style and best practices

- JavaScript code looks like quality JavaScript code. C# code should look like quality C# code.
- Proper use of whitespace:
    - Indentation.
    - Appropriate new lines between methods and logical sections of code.
- Well-factored functions.
    - Attached to objects, or scoped properly.
    - Meaningfully named.
- Meaningful variable names.
- Meaningful/standard folder names and folder structure (e.g., scripts, styles, partials, …).
- Meaningful/standard file names (e.g. index.html, server.js, app.js, ...).
- Consistent and appropriate case usage (camelCase in JS, kabob-case in html, etc).
- Appropriate level of comments, especially comments that explain WHY the line or section of code exists.

A great submission follows Git best practices

- Meaningful commit messages that explain WHY—not what—the changes are.
- Proper use of branches.
- No commented-out code included.
- Reasonable number of commits.
- Proper files included/ignored (e.g.: no `.DS_Store` files or `node_modules` flotsam).

Code should be original and attributed

The above assumes that work falls within the guidelines of the Code of Conduct, or consequences there should be followed.

Step-by-Step Grading Process

1. Before you start grading:
    a. Verify with the instructor that no in class changes were made to the assignment during class.
    b. Create a directory for the assignment you are about to grade.
    c. In that directory, create directories for each of the students you are responsible for grading.
2. Grading:
    a. Did they answer all the questions in their comments on Canvas?
        i. If not then do not grade and instead request that they answer the questions on Canvas.
        ii. If yes then continue with the grading process.
    b. Clone the student's Github repository.
    c. Does their submission pass the linter?
        i. If not and it is clear they made little effort:
            1. Award 0 points.
            2. Direct them to resubmit.
        ii. If not but it is clear they made an effort:
            1. Award up to 5 points.
            2. Direct them to resubmit.
        iii. If yes then continue with the grading process.
    d. Check for meaningful variable names:
        i. Do they accurately and succinctly describe the data they are storing?
            1. If not:
                a. Leave a comment on their pull request commit message on Github.
                b. Deduct 1 point.
            2. If yes then continue with the grading process.

e. Check for meaningful function names:
    i. Do they follow industry best practice (camel case, describe the action they perform etc.):
        1. If not:
            a. Leave a comment on their pull request commit message on Github.
            b. Deduct 1 point.
        2. If yes then continue with the grading process.
f. Check for well factored functions:
    i. Is the code well factored commensurate with what they have been taught to this point?
        1. If not:
            a. Leave a comment on their pull request commit message on Github.
            b. Deduct 1 point.
        2. If yes then continue with the grading process.
g. Check for meaningful project structure:
    i. Are the folder and file names meaningful and follow best industry practice?
        1. If not:
            a. Leave a comment on their pull request commit message on Github.
            b. Deduct 1 point.
        2. If yes then continue with the grading process.
h. Check for appropriate level of comments:
    i. Could their comments be improved significantly in terms of being meaningful, too excessive, too sparse etc.?
        1. If yes:
            a. Leave a comment on their pull request commit message on Github.
            b. Deduct 1 point.
        2. If not then continue with the grading process.
i. Check for code that has been commented out:

        i. Does commented out code exist?
            1. If yes:
                a. Leave a comment on their pull request commit message on Github.
                b. Deduct 1 point.
            2. If not then continue with the grading process.
    j. Check for proper git workflow:
        i. Do they have meaningful commit messages, proper use of branches, reasonable number of commits, a gitignore etc.?
            1. If not:
                a. Leave a comment on their pull request commit message on Github.
                b. Deduct 1 point.
            2. If yes then continue with the grading process.
    k. Check for late assignment submission:
        i. Was the assignment submitted past the deadline set?
            1. If yes:
                a. Deduct 2 points.
            2. If not then continue with the grading process.
    l. Check for bonus points:
        i. Did they achieve any/all of the stretch goals specified for the assignment?
            1. Award additional points using best judgement.

Chapter 5

Conclusion

How Can WE Help YOU?

As a TA, you are a vital and valued member of the Code Fellows team. We want to do whatever we can to make the experience as rewarding, enjoyable, and enriching as possible. Be sure to read through the supplemental links at the end of this handbook. The more you learn about our educational processes and their underlying theories and practices, the better learner you will be. Good developers are always learning. Our instructors and Principal Teaching Assistant will be available to you during many of the off-hours in the event that you have any questions or observations.

Your Input is Welcome!

While assembling this handbook, we have tried to think of everything that you need to know. However, we know we have probably missed a thing or two. Plus, things change, and it is easy to overlook details from time to time. If you see anything in this handbook that could be explained more clearly, or have an idea for something to improve or add, let us know! Your primary point of contact for feedback is the Principal Teaching Assistant.

In Conclusion

A reminder of the TA Guiding Principle, “We lead with humility and empathy to create safe and inclusive environments where people learn and lives are transformed.”

Being a TA is a leadership position. Students look at you and understand that where they are now, you used to be and where you are now, is where they are striving to get to. Having that real life example in front of them, showing that it CAN be done is huge and with that also comes a hefty weight of responsibility to model for students what it looks like to be a Code Fellows Alumni.

Be sure to list your TA experience on your resume and on LinkedIn.

Have fun. Be professional. Be kind.

Appendixes

Resources: Links Referenced Above

- David Kolb's adult learning cycle model: http://www.simplypsychology.org/learning-kolb.html
- Stacked learning: https://www.codefellows.org/blog/how-to-accelerate-your-learning-with-stacked-modules

Code & Ops 101 Guide

Thanks in advance for helping introduce people to the world of code! Code 101 is truly an adventure - both for you, and for the students who are often going way out of their comfort zones to take this class. You are representing Code Fellows to new people for the first time, quite often. Here are some important things to know about this class, and your role in making it successful:

- Unlike some other classes, the primary goal of this class isn't learning to code. Yes, we teach students HTML and CSS, but the goal is broader: to help them determine if learning to code, and perhaps becoming a software developer, is something that's right for them. As such, we try to give students an accurate picture of what the industry is like, what kinds of jobs are out there, and what it's like to learn to code.
- These students are beginners. It sounds obvious, but there's a profound difference between true beginners and even advanced-beginners: advanced-beginners know enough to ask questions. True beginners don't even know they should have questions yet. This means that you can't wait for students to ask you for help; you'll be more effective crouching down next to someone, and saying "let me see what you have." You'll find things to help them with in no time.
- Learning to code steps on everyone's buttons at times. This means that all kinds of emotions come up in this class: anger, fear, frustration, saying "this is stupid", needing to rest on their laurels and tell you about the other things they're good at...we see it all. Instead of reacting to their anger/fear/whatever, normalize the experience of "not getting it" - let them know everyone (including yourself) has challenges, too. This is why the Code Fellows TA mantra is: "We lead with humility and empathy to create safe and inclusive environments where people learn and lives are transformed." It's up to us to create that safe and inclusive environment.
- This class isn't about learning to code, It's about trying on this field for size and making a decision about whether they want to learn more. The more you keep the focus on that, the less people are worried about not learning enough code, or getting something wrong, or not having amazing CSS, etc.

- Be really forward about the fact that the coding in this class will start at the very beginning and not assume any prior knowledge. This makes your beginners feel safer and your more advanced folks know what to expect.
- Timing: there is some play in the schedule, but if students get too far behind, they'll never recover. This is definitely one of those classes where you have to be on the ball, be careful about which stories you tell, and stay away from all those tempting diversions (penguins! Code Fellows jokes! Cool topics you like!).

Workshop Outline

See the structure of the day in the appropriate Facilitator Guide repository:

- Code 101
- Ops 101

Remote 101

If a 101 needs to be held virtually, there will be some minor changes to how the course is run. The content will be the same as in the guide, however the pods of desks will now be breakout rooms within Zoom. Here are the key changes:

- TAs should add TA to both their Zoom and Slack profiles to help students identify who they can ask for help
- At the start of the workshop, students will be moved to breakout rooms so the TAs can verify and help with any setup issues.
- Groups of students will be moved into breakout rooms to work on their projects. The ideal group size will be four students.
- TAs will move between the breakout rooms to check on students, help with any questions, provide feedback and encouragement.