Join The Odin Community

Foundations Course

Introduction

Working and collaborating with other people is an important part of working as a web developer. Therefore, we at The Odin Project encourage you to participate in our online chat community, which we'll talk more about below. By joining the community, you can grow alongside other Odinites and help each other learn web development. While you're at it, you can check out our Facebook page, follow us on Twitter and catch up on Instagram. Use #TheOdinProject to share your Odin Project progress, updates, thoughts and to see what other Odin students are up to!

Why a Community Is Awesome for You

Learning web development will be a long and arduous journey, but you can make the marathon a lot more fun by getting involved in our Discord community. No matter what pace you are doing our curriculum, there will always be people a few steps ahead of you that are willing to help. Furthermore, helping others that are a few steps behind you is a great way to deepen your own understanding and make your learning stick.

When you're slogging through the desert of despair, where your code doesn't work or even make sense to you anymore, you can find an oasis of knowledge and encouragement in our community. Veterans at The Odin Project love to help fill in knowledge gaps and provide new insights and perspectives on how to improve your code. We’ve all been there before!

Remember that project you struggled so hard to figure out and that you’re so proud of finishing? Through our community, you will get to share your work and progress with those who fully appreciate how much hard work went into it.

Why a Community Is Awesome for Odin

We are working hard to update existing lessons and produce new content, so we would love to hear your feedback on the lessons and projects. We hope you find the lessons fun, engaging, and informative and find the projects challenging but achievable. So please let us know your thoughts!

Before Asking for Help

As most of the projects are designed to push you to your limit, please remember that there is always a community to turn to! You don’t need to know how to solve every problem straight away, BUT you do need to have a general idea of where you are going. This becomes really important when asking your questions because sometimes the problem is your **approach** and not your code.

If you’re feeling stuck, it’s a good time to pause and take a breather. Break the problem down into little pieces and then decide what is really holding you back. We call this technique rubber duck debugging.

You should also do a Google search to find relevant information for your problem. Read about how to use Google to solve your programming questions. You can also look back at previous lessons for tools that you can apply to the current task.

If these methods don't yield a solution for you, then it's time to reach out to the Odin community and ask for help.

Asking for Help

So you've spent some time struggling to solve the problem on your own, and now it's time to fire up the Odin Discord and ask for help. The first thing to keep in mind is don't ask to ask. While this is a simple idea (with a pretty catchy motto!), it can help you get answers to your questions much faster and will make it easier to others to feel comfortable offering you help.

In addition, when you ask your questions, you should help the community help you by putting together a great question.

When asking your question, please remember to include the context:

- What do you think the problem is?
- What exactly do you want to happen?
- What is actually happening?
- How did you get there?
- What have you tried so far?

If you can't pinpoint the problem, you can share a **screenshot**. This is especially useful for showing the output of commands in the command line. In Discord, drag and drop your screenshot image file into the chat box to upload it or simply use the PrtScn and paste keyboard shortcuts. If you don't know how to take a screenshot on your computer, this is a good time to ask Google.

Screenshots are great for showing the output of commands or error messages in the command line. Screenshots are also great for showing the output from your code such as how the output looks visually on a webpage or console outputs in the browser. However, you should always include the corresponding files containing the error. Even if it is a short amount of code, providing it in the discord server in the proper format along with a screenshot of the output is helpful to those debugging it rather than just a screenshot. When you do share a screenshot of the output or how it looks visually make sure to push your project to GitHub or put your corresponding code in a replit so that others can comb through and debug the code. The screenshot of the output and the correlating code that can recreate the problem will help make it easier to understand the problem for people helping you. You'll learn all about GitHub very soon.

Sometimes there might be no one around to help you with your issue. That is the ideal time to get familiar with the Discord search function. Search for specific keywords or error messages to see if anyone else had a similar issue before and how they solved it!

Formatting Your Questions

Asking your questions in a readable format helps everyone debug them better. Here are some ways to go about that:

If you're having trouble on the command line, make sure to include both your input and the error message you're getting.

In the chat rooms, code can be displayed differently from normal sentences by using **backticks** (``), which can be found above the Tab key on US and UK keyboards. Backticks are not the same as single quotation marks (''), which are found to the left of the Enter key.

For a single line of code: use one backtick before and after your code.

`Your Code`

For multiple lines of code: use three backticks *on a separate line* above and below your code.

```

Your Multiple Lines of Code

```

You can also use *code highlighting* to add color to your multi-line code by specifying the language:

**```javascript**

Your Multiple Lines of Colorful Code

```

## Chat Features

- Have fun with giphys: type `/giphy hi` to say hi to everyone.
- Type `!help` for more information on chat commands.
- Show your appreciation to those who help you with `@username ++`.
- Don't forget to visit all the available rooms!

## How to Help Others Solve Coding Problems

Not only is it important to know how to ask an effective question, it is also important to know how to help others effectively. Please take a moment to review these guidelines so that you will have proper expectations of the help you will receive in our Discord community. In addition, come back and review these guidelines when you are ready to start helping others.
```

1. Instead of Answering the Question, Guide Them to the Answer

Unless the problem is a simple typo or syntax error, it is more beneficial to guide them to find their own answer. This approach will teach good debugging skills and will increase their ability to solve future problems.

Start by asking probing questions, such as “What have you already tried?”, “What do you expect this function to do?”, or “What do you think that error means?”.

2. Help Only When You Are Certain of the Answer

If you are not 100% certain of the answer, you may end up doing more harm than good, so please let someone else answer it.

Do not worry about how long someone has to wait for an answer. The right answer is worth the wait.

3. Help Only When No One Else Is Currently Helping

If somebody is already getting help, do not jump in the middle of the conversation. We know you mean well, but it is overwhelming for the person receiving help to follow multiple conversations.

4. Help Only When You Have Plenty of Time

If you do not have much time to help, please let someone else answer the question.

5. Adjust Your Expectations to Their Level

If the question does not reveal where they are in the curriculum, ask them so that you can adjust your expectations to their knowledge level.

6. Ask for Clarifications

If the question seems confusing or ambiguous, ask for more clarity, or politely link them to our bot command `!question`, which links to the How to be great at asking coding questions article.

7. Ask for Live Code

If the question needs to have live code to fully understand or debug, ask them to use replit to provide it. If the problem is difficult to isolate, they should recreate the problem with isolated code.

8. Do Not Answer Googleable Questions

Learning how to research these questions is a very important skill for developers, so we need to empower them to find their own answer. When we answer these questions, it hinders their personal growth and makes them codependent on our community.

Instead of answering these questions, politely ask them to google their question or use our bot command `!google` with the search terms.

9. Do Not Answer Questions Covered in Our Curriculum

If you know that the answer is provided in our curriculum, ask them where they are at in the curriculum.

If they have not reached that portion of the curriculum, let them know they will learn it in the future.

If they have already been through that portion of the curriculum, politely direct to review that lesson.

10. Answer the Question Before Pointing Out Other Problems

When helping someone it can be easy to spot other problems in their code. Resolve the original question, before pointing out any other problems that need attention.

11. Encourage Students to Use a Debugger

It is common for students to not understand the importance of using a debugger to look at the values of their variables at different points in their program. When students are getting unexpected values, politely encourage them to use a debugger with our bot command `!debug`.

12. Watch for Students That Need to Take a Step Back

It is common for students to focus too hard on a problem and not be able to clearly see everything. When this situation arises, politely encourage them to step back from the problem and take a break. Often times, stepping away from a problem will help them see the bigger picture and how to solve it.

13. Watch for Students That Are in over Their Head

It is common for students to skip a lesson/project or think they know more than they actually do. When this situation arises, politely encourage them to go back and reread a section of the curriculum for more understanding.

14. Admit When the Problem Goes Beyond Your Current Knowledge

It is common for the actual issue to go beyond the initial question. If it goes beyond your current knowledge, it is important to admit that you are unsure of the correct answer and let someone else help.

After digging deeper into the problem, they might be able to continue troubleshooting on their own or they can wait for someone more experienced to help.

15. Be Patient

Helping others solve a problem is not always easy. Remember to be patient as they struggle through the problem.

16. Duck Out of the Conversation If You Get Frustrated

Sometimes there are misunderstandings and interactions go poorly. You are a volunteer and are not obligated to help when things get out of hand. Politely duck out of the conversation and let someone else step up.

Assignment

2. First, create a free GitHub account. As you will discover, GitHub is an integral part of the development workflow.

3. Now, sign in to our Discord server. Pop in and say hello! We've created an introductions room which is a great place to introduce yourself and we're always happy to welcome new community members. We have chat rooms for every development topic covered in our curriculum. Log into the chat and start exploring!

   - **Link your GitHub to your Discord profile:** Go to `Discord Settings > Connections`, then click the GitHub icon. In the new tab that opens click "Allow Access", then back in Discord make sure "Display on profile" is toggled on. This will allow others to see what you're working on and vice versa!

4. Here are some guidelines before you dive in:

   - **Ping (@user) With a Purpose:** Only @ another user when it is necessary. Include your question or comment in the message. Wait until they reply before pinging again.

   - **Don't 'Bomb' Chats:** Don't send multiple messages in a row; type out your whole message, then push send.

- **Don’t Exclude Anyone:** These are public chats; if someone joins in on a conversation, include them!
- **Don’t Disappear Right After Asking for Help on Code:** If you’re posting a question, make sure you have time to stick around and discuss it with those trying to help!
- **Remember the Human:** Behind every username there is a person with feelings! Be kind! If you don’t have anything nice to say, don’t say anything at all.
- **If You Wouldn’t Say It Out Loud Don’t Type It:** Plain and simple.
- **Read the Rules and FAQ:** Upon joining, you will find yourself having to read through our rules. Please take the time to read and understand our rules and FAQ.

Additional Resources

This section contains helpful links to related content. It isn’t required, so consider it supplemental.

- It looks like this lesson doesn’t have any additional resources yet. Help us expand this section by contributing to our curriculum.

View Course

Login to track progress

Next Lesson

Improve this lesson on GitHub

Have a question?

Chat with our friendly Odin community in our Discord chatrooms!

Open Discord

Are you interested in accelerating your web development learning experience?

Get started

Thinkful

5-6 months

Job Guarantee

1-on-1 Mentorship