Level Up Your Skills With the Real Python Slack Community

by Ricky White Oct 21, 2020 3 Comments best-practices community

Mark as Completed

Table of Contents

The *Real Python* Community Slack is an English-speaking Python community with members located all over the world. It's a welcoming group in which you're free to discuss your coding and career questions, celebrate your progress, vote on upcoming tutorial topics, or just hang out with us at the virtual water cooler.

As a community member, you also get access to our weekly Office Hours, a live online Q&A session with the *Real Python* team where you'll meet fellow Pythonistas to chat about your learning progress, ask questions, and discuss Python tips and tricks via screen sharing.

The aim of this guide is to help you:

- Navigate some of **Slack's most useful features**
- Get the most out of the *Real Python* **Slack community**
- Get your **questions answered** by other *Real Python* members
- Learn how to **communicate technical problems** to your peers
- Get comfortable with the tools you'll use when you get your first (or next) **developer**

Table of Contents

- Successfully Posting to Slack
- Finding the Best Channel to Ask Your Question
- Communicating Your Problem Effectively
- Making Your Code More Readable in Slack
- Getting Responses to Your Questions
- Helpful Resources
- Conclusion

Mark as Completed

Tweet Share Email

job

We'll update this guide periodically and welcome any recommendations or questions that you may have. You can share them with me (@Ricky White) in Slack or in the comments below. We'll make all update announcements in the **#hangouts** channel of Slack.

> **Free Bonus: 5 Thoughts On Python Mastery**, a free course for Python developers that shows you the roadmap and the mindset you'll need to take your Python skills to the next level.

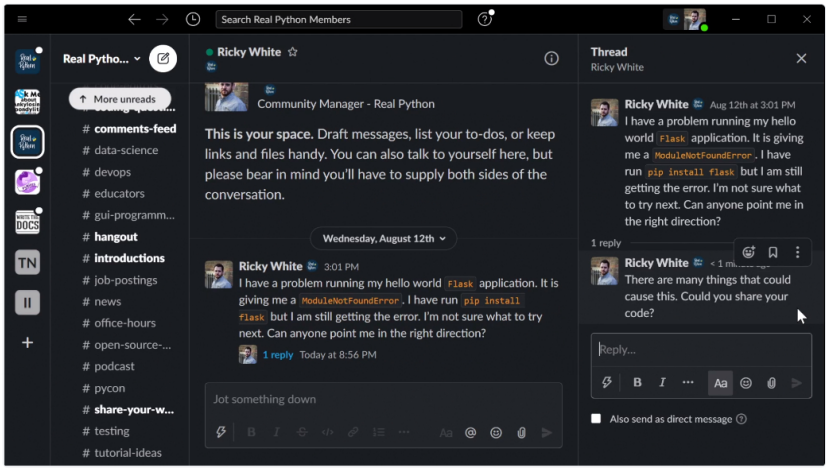
Successfully Posting to Slack

Slack lets you post messages and ask questions that everyone within the channel can see. You can also send direct messages to individual users. However, there are a few things to consider before you hit that send button. Let's discuss those considerations in the context of the three most common ways you'll post messages in Slack: replying, posting, and cross-posting.

Answering a Member's Question

The best way to respond to another member's post is to use the *Reply in thread* button. Using **threads** has the advantage of keeping the entire conversation in one place, which won't happen if you reply with a new post.

Here's an example of how to use *Reply in thread*:



The thread feature is an excellent way to ensure that questions from other members don't get buried in a flurry of answers to a previous question, which can result in questions being left unanswered.

Another benefit of threads is that they make it very clear to community helpers which questions have already been answered and which remain unanswered. This helps members determine where to focus their time and energy.

Posting Your Question

When you run into a problem with your code, you may be tempted to jump on Slack, write out your problem, hit *Send message*, then copy in your code and hit *Send* again. You may even want to write a more detailed question or explain the solutions you've already tried and then—yep, you guessed it—hit *Send* again.

That's three posts for the same question.

This approach seems harmless and is technically possible in Slack. But which post are

This approach seems harmless and is technically possible in Slack. But which post are people supposed to respond to? The question, the code, or maybe the initial post where you stated the problem? It's unclear.

Instead, you should make sure your problem, question, and code are all contained in just **one post**. This allows people to follow the guidelines for replying in one succinct thread instead of across multiple threads, which can lead to repetition in responses.

You'll learn more about the best ways to structure your questions and format your code in a bit. For now, all you need to know is that limiting your question to a single post will benefit you and the rest of the *Real Python* Slack community.

Cross-Posting

As a general rule, you should try to avoid **cross-posting** your question to several channels. Cross-posting might seem like an efficient way to get more people to see your question so that you're more likely to get an answer, but it often has the opposite effect.

If members see you cross-posting all the time, then they may assume that someone else has already answered your post elsewhere and be less inclined to respond. Or a member may take considerable time to read your code and write a helpful response, only to find that your question has already been answered in another channel.

There's one exception to the rule against cross-posting: if your question goes unanswered, and the reason might be that you accidentally posted in the wrong channel (more on that later), then that would be a good time to cross-post to an appropriate channel.

If you find yourself in this situation, then you should use the *Share message* feature. This will automatically format your question in a way that clearly shows readers that it's a cross-post, and it will link back to the original post so people can reply to the original thread.

Unless you meet the exception above, though, we encourage you to avoid cross-posting and instead post your question in the most appropriate channel for your question.

Finding the Best Channel to Ask Your Question

There are several Slack **channels** in which you can ask questions. Most channels focus on a particular topic or specialty. When you join Slack, you'll automatically get added to a few general channels. You're also free–and encouraged–to join any additional channels that exist within the community Slack.

You can find the list of available channels by clicking the **+** icon next to the Channels menu. Select the channel you wish to join and click the appropriately named *Join* button. For instance, you may wish to join the **#webdev**, **#data-science**, or **#careers** channels if those subjects interest you.

Once you've joined a few channels, you can ask your question in the channel that is most closely related to your problem. If your question doesn't fit into a special category, then **#coding-questions** is the channel for you.

Communicating Your Problem Effectively

We highly encourage you to reach out to the *Real Python* community whenever you have a coding problem. The adage that there's no such thing as a stupid question most definitely applies within the community. However, it's important that you communicate your questions clearly and succinctly.

To get the best help, make sure to include all the relevant information in your post. Your question should contain **four parts**:

1. A detailed description of what you're trying to achieve and the problem you've

encountered

2. A summary of approaches you've already tried to fix the issue
3. Any thoughts you have on what the problem might be or what's causing it
4. The traceback or error message, the relevant code, and any terminal commands you used to run your code

Running over this short checklist of information to include in your question will save you time and increase the chance that members can help you solve your problem.

Communicating your thought process and problems effectively is a skill and will get easier the more you do it. So it's especially important that, as a beginner, you make sure to ask thoughtful, well-structured questions. That way you'll form good habits and positive relationships right from the start.

Making Your Code More Readable in Slack

The ability to share your code is one of Slack's greatest strengths and makes it a great platform for the *Real Python* community. There are **three ways** that you can share formatted code on Slack:

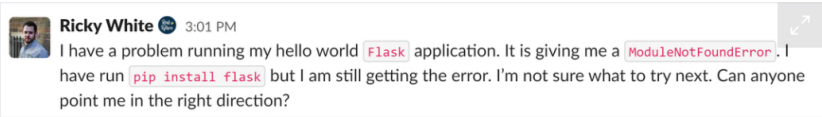
1. Inline code
2. Code blocks
3. Code snippets

Each has its use case, and you should aim to adhere to the principles laid out below.

Inline Code

Inline code is the best way to format small bits of code that you want to mention in your question or comments. It's not intended to be used with whole code blocks or functions—it's primarily for mentioning specific **variables** and **objects** within your question.

Here's an example of inline code:



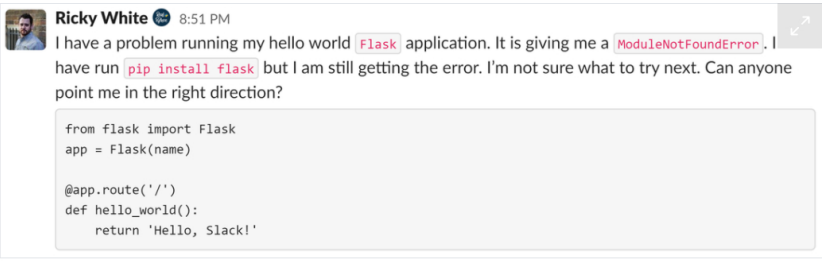
The words `Flask`, `ModuleNotFoundError`, and `pip install flask` are formatted as inline code. Formatting your code questions like this adds context that other members can use to quickly parse the import bits of information.

To add inline code to your question, wrap your variable or object name in backticks (`) or click the *Code* formatting button on the message editor.

Code Blocks

Code blocks are an intermediate formatting technique that you can use for **small code examples** like single functions, REPL and terminal outputs, and short code excerpts.

Here's an example question that uses inline code followed by a code block:



The question itself uses inline code for objects and variable names and is followed by a code block showing the code in question. To create a code block, wrap your code in triple backticks(```) or click the *Code block* button (not the *Code* button!) in the Slack message editor. That's it.

Code blocks are best used for showing ten or fewer lines of code and should never be used with code involving more than twenty lines. This is for a few reasons:

- There's **no syntax highlighting**, which makes long passages of code more difficult to read.
- It fills the screen to the point that **other member's questions can get buried** or missed.
- It makes for an **unpleasant reading experience** for anyone scrolling through the feed.

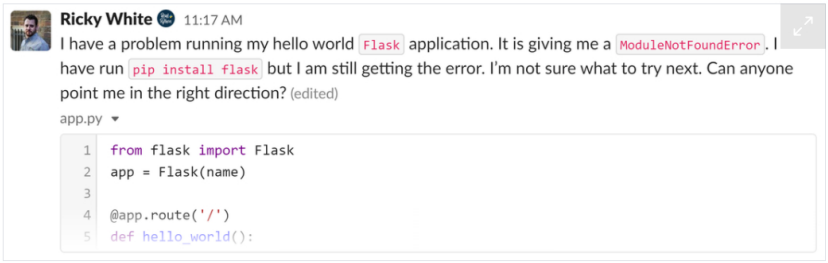
If you need to share large portions of code, then there's a better way: code snippets!

Code Snippets

Code snippets, which are not to be confused with code blocks, are perhaps the most underused feature in Slack. They make sharing and reading large portions of code in Slack a much more pleasant experience. Snippets should be your preferred method of sharing most of your code.

You can use both inline code and code blocks within a code snippet message. This is especially useful when you want to share your code and an error message or traceback at the same time, which adheres to the guideline for keeping questions wrapped up in a single, well-formatted post.

Here's an example of a code snippet:



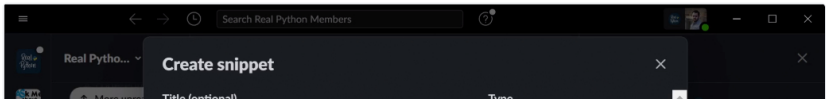
One significant benefit of using code snippets is that they support **code syntax highlighting** for almost any programming language. They're also collapsible, making long code samples less intrusive. Interested members can expand and read the snippet in full with just a single click.

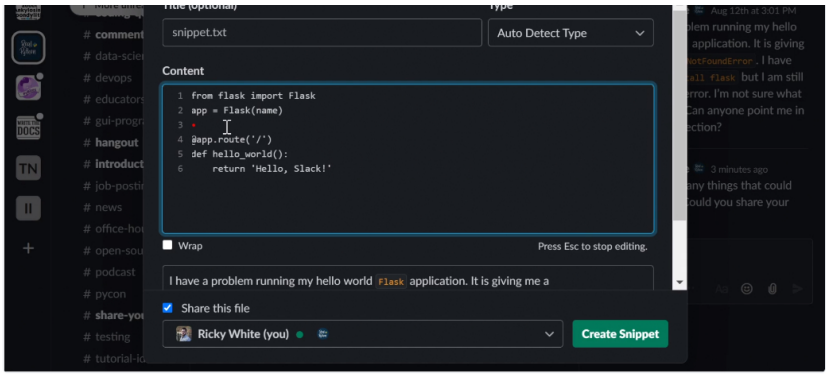
One of my favorite features of code snippets is that they're also **downloadable**. This means that anyone helping you with a coding problem can download your code file and open it in their code editor, potentially enabling them to replicate your error much more easily. This is a really powerful feature!

> **Important:** Use the download feature sparingly. It's impractical to expect someone to download your entire project if it's larger than a single file. If the person helping you needs access to *all* your code, then please share the link to your project's Git repo instead.

To create a code snippet, click the lightning bolt icon in the Slack message area and select *Create a text or code snippet*. This will open an overlay in which you can type your question and copy your code into the content area. You can give the snippet a title (which you can treat like a filename) and choose the language for syntax highlighting.

Here's a quick demonstration of how to create a code snippet:





Code snippets require a few more steps than inline code or code blocks, but they have a much greater readability.

If you have any further questions about formatting your code in Slack, let me know in the comments below, or reach out to me on the *Real Python* Community Slack (@Ricky White).

Getting Responses to Your Questions

Not getting your question answered can be frustrating. If you find that your questions aren't generating responses, then it might be worth asking yourself a few questions to see if there is anything you can do to remedy that situation:

- How much time has passed?
- Did I include all relevant information?
- Did I post to the most appropriate channel?
- Where else can I look for an answer?

Let's look at each of these in a little more detail.

When you ask yourself how much time has passed, if the answer is just a few hours, then give it more time. Most members have full-time jobs and are checking Slack only when they have spare time to work on their Python skills.

Time zones also play a key factor into when your question may get answered. While there's no right time to post, you may find questions that are posted in the evenings or on weekends get answered sooner. Don't let that stop you from posting your question, though. The best time to post is when the problem is fresh in your mind.

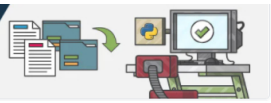
Earlier, you covered what information to include in your question. If you're not getting answers, then check if there's something you left out. Did you include your traceback or error message? Is all relevant code included and properly formatted? These may all affect members' ability to help, so it's worth looking to see if you can revise your question and make it easier for others to understand.

When you're a beginner, it can be hard to know what information to include and what to leave out, and that's okay. It'll get easier with practice. Just try to follow the guidelines above and don't let doubt stop you from posting your question. The *Real Python* community is full of welcoming, generous people who will make every effort to help you.

If you've waited sufficient time and are sure your question is clear and descriptive, then you might also consider if you asked your question in the right place. Would it be better suited in a different channel? If so, then cross-post it to the new channel using the *Share* button mentioned earlier. This will show people where you posted originally, and they'll see that you still need help.

If you've exhausted all these options, then you could consider asking your question in the members-only Office Hours. If you can't attend live to ask your question, then you can share your post in the **#office-hours** Slack channel and ask for it to be added to the list of questions for the session.

Finally, if none of the above yields a result, then reach out to me directly in Slack (@Ricky White) and I can either help you find a solution or point you in the right direction.



Helpful Resources

If you would like to learn all the formatting options for your posts and messages, then the Slack documentation has a few articles to help. You can find general formatting help on their article titled Format Your Messages. If you'd like to learn more advanced formatting techniques, then you can learn how to use markup to format your posts instead.

If you're a beginner, then I would also encourage you check out Understanding the Python Traceback. This will help you better understand what's going on in your code, which will help you ask better questions in the community Slack.

Conclusion

Participating in a vibrant community like the *Real Python* Slack can have enormous benefits for your learning and career journey. In this tutorial, you've learned how to start posting in Slack and how to use some lesser-known features that can help you get the most from the community.

Learning how to communicate your technical problems is a significant step toward becoming a proficient developer. It's my hope that this guide will help you achieve this so that you can start tapping into *Real Python* members' extensive knowledge base.

The *Real Python* community is diverse, with members of all backgrounds and experiences contributing from all across the globe. Making connections with such a varied network of people will help you enhance your **technical skills**, expose you to new **areas of interest**, and allow you to **network** with other Pythonistas.

Connecting with your peers isn't just an online activity, either, as you can see from our PyCon 2019 meetup:

If you've yet to take advantage of the benefits that the Slack community offers you as a *Real Python* member, then you can sign up for an account today to connect with your peers and start advancing your Python knowledge and career!

```
>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Email Address

Send Me Python Tricks »

About **Ricky White**

Ricky is a Python developer and writer who has a tendency to refer to himself in the third person. By day he is a stay-at-home dad, and by night he's a crime-fighting ninja. You can often find him on Twitter, or at endlesstrax.com.

» More about Ricky

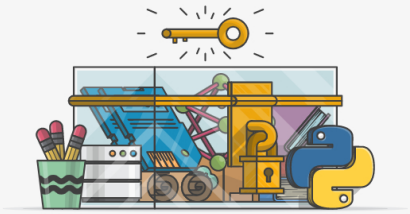
Each tutorial at Real Python is created by a team of developers so that it meets our high quality standards. The team members who worked on this tutorial are:

 Aldren

 Joanna

 Jacob

Master Real-World Python Skills With Unlimited Access to Real Python



Join us and get access to hundreds of tutorials, hands-on video courses, and a community of expert Pythonistas:

Level Up Your Python Skills »

What Do You Think?

Tweet Share Email

Real Python Comment Policy: The most useful comments are those written with the goal of learning from or helping out other readers—after reading the whole article and all the earlier comments. Complaints and insults generally won't make the cut here.

What's your #1 takeaway or favorite thing you learned? How are you going to put your newfound skills to use? Leave a comment below and let us know.

Keep Learning

Keep Learning

Related Tutorial Categories: best-practices community