

Developing Discourse Plugins - Part 7 - Publish your plugin



- [Documenting your plugin](#)
 - [GitHub Documentation](#)
 - [Meta Documentation](#)
 - [Updating your documentation](#)
 - [Supporting your plugin](#)
 - [More in the series](#)
-

Previous tutorial: <https://meta.discourse.org/t/developing-discourse-plugins-part-6-add-acceptance-tests/32619>

You've [created your plugin](#), you've [uploaded it to GitHub](#) and you've [added tests](#). Great! Problem is, no one else knows about it.

Documenting your plugin

All plugins require good documentation. Users need to know what the plugin does, how to install it, important settings/configuration changes needed, and how to use it. Plugins should be documented in two different locations: the `README.md` file within your git repo, and a dedicated topic in the `#customization:plugin` category.

GitHub Documentation

The `README.md` file on GitHub is important as it's shown by default when a user visits your repo. At minimum, your readme should include the plugin's title and a short description. A more complete readme will also include installation instructions, a more detailed description, basic usage instructions, license, and (if applicable) screenshots. If additional instructions are included in your

plugin's Meta topic, be sure to include a link to the topic.

Example of a minimally documented plugin: [Discourse Data Explorer](#) Examples of plugins with more complete documentation: [Discourse Solved](#), [Discourse OAuth2 Basic](#), and [Discourse Ads](#).

[details="Sample plugin README template"] *Be sure to update text surrounded by `<>` with your plugin specifics.*

```
## <Plugin Title>
```

```
<Plugin Description>
```

```
## Installation
```

```
Follow the [plugin installation guide](https://meta.discourse.org/t/install-a-plugin/19157).
```

```
## How to use
```

```
<Explain how to enable the plugin, necessary configuration steps, and how to use it>
```

```
## Screenshots
```

```
<Include screenshots if necessary to demonstrate your plugin's usage>
```

```
## Read More
```

```
<Include a link to your Meta topic if more information is detailed there>
```

```
## License
```

```
<Note your code license, most Discourse plugins use MIT>
```

[/details]

Meta Documentation

Where GitHub documentation tends to be short and "to the point", Meta documentation is where you get to share the full details of your plugin, and convince users why they should use it. At minimum, your Meta topic should include a short description of the plugin, and a link to the plugin's repo on GitHub (so users can install it). More complete documentation will also include a detailed description include example use cases, detailed usage instructions, documentation of all settings and configuration options, and screenshots. Screenshots are key as users want to see what your

plugin does, not just read about it. A plugin adding an auth provider likely needs only 1 screenshot, where a plugin that adds a new interface, or makes major changes should have more quite a few.

Meta documentation tends to be more personal than GitHub, each plugin author has their own documentation style. Whatever style you choose, ensure it is clear, easy to read, and organized. Use headers as appropriate, annotate screenshots to explain complex interactions, and be sure to be consistent in your formatting. Avoid the temptation to write a single giant paragraph.

Updating your documentation

After you write your initial documentation, it's important to keep it up to date.

Added a new feature to the plugin? Be sure to save some time to document it. Answered the same question multiple times? Consider adding a FAQ section to your Meta topic. Known issue that is complicated to fix? Detail it in your topic so users know what to expect.

Supporting your plugin

After you publish your plugin and it's documentation, site admins who find it useful may install it on their site and start to use it. This usage requires ongoing support by the plugin developer. You'll want to answer questions that site admins have, helping them to use your plugin. Something that made perfect sense to you may be unclear to others, so you'll want to update the code and/or documentation to clarify it. You may receive feature requests, which you'll have to decide whether or not to add.

Lastly, Discourse itself is under constant development. While your plugin may work perfectly today, tomorrow something might change that breaks it. Be sure to stay up to date on Discourse development so that you can update your plugin to support the latest version of Discourse when a change affects your plugin.

More in the series

Part 1: [Plugin Basics](#) Part 2: [Plugin Outlets](#) Part 3: [Site Settings](#) Part 4: [git setup](#) Part 5: [Admin interfaces](#) Part 6: [Acceptance tests](#) **Part 7: This topic**

📄 #2

📄 📄 📄 2026-10-31 13:29:33 UTC

📄 📄 📄 2026-10-31 13:29:34 UTC