

Django Open Source Contribution Guide for Beginners

This comprehensive guide is designed to help beginners understand and participate in the Django open source contribution process. By following a real-world example of fixing an actual Django ticket, you'll learn the complete workflow from finding an issue to getting your code merged.

Key Concepts Refresher

Before diving into the contribution process, let's review some key concepts:

What is Django?

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It's built by experienced developers and takes care of much of the hassle of web development.

```
# A simple Django view
from django.http import HttpResponse

def hello_world(request):
    return HttpResponse("Hello, World!")
```

What is Open Source?

Open source software is code that is designed to be publicly accessible—anyone can see, modify, and distribute the code. Django is open source, which means its development relies on community contributions.

What is Git/GitHub?

Git is a distributed version control system that tracks changes in source code during software development. GitHub is a platform that hosts Git repositories and provides tools for collaboration.

```
# Basic Git commands
git clone https://github.com/username/repo.git      # Clone a repository
git checkout -b new-branch                           # Create a new branch
git add file.py                                     # Stage changes
git commit -m "Fixed issue #123"                   # Commit changes
git push origin new-branch                          # Push to remote
```

Learning Objectives

By the end of this guide, you will:

1. Understand how to find beginner-friendly issues in Django
2. Know how to set up a Django development environment

3. Learn the process of fixing a real Django issue
4. Understand how to submit a pull request
5. Learn how to respond to feedback from maintainers

The Complete Contribution Process

This guide is divided into three main parts:

1. [Django Contribution Guide](#) - An overview of the contribution process and how to get started
2. [Step-by-Step Ticket Fix Walkthrough](#) - A detailed walkthrough of fixing a specific Django ticket
3. [This Summary Document](#) - Tying everything together with best practices and additional resources

Selected Ticket: #35924 - FilteredSelectMultiple Widget Arrows

For this guide, we selected ticket [#35924](#), which deals with the responsiveness of arrow buttons in the FilteredSelectMultiple widget in Django's admin interface.

This ticket was chosen because:

- It's labeled as "Easy pickings"
- It involves a UI/UX improvement that's easy to understand
- It requires minimal code changes
- It provides a good introduction to Django's admin interface
- It's a real issue that needs fixing

Best Practices for Contributing to Django

Based on our walkthrough, here are some best practices for contributing to Django:

1. **Start Small:** Begin with "Easy pickings" tickets to familiarize yourself with the contribution process
2. **Read the Documentation:** Django has excellent [contribution documentation](#)
3. **Ask Questions:** Use the [Django Forum](#) or [Discord](#) if you're stuck
4. **Write Tests:** Django has a strong testing culture; always include tests with your changes
5. **Follow the Style Guide:** Django follows [PEP 8](#) with some modifications
6. **Be Patient:** The review process can take time, especially for larger changes
7. **Be Responsive:** Respond promptly to feedback on your pull requests

Common Challenges for Beginners

New contributors often face these challenges:

1. **Setting up the development environment:** Django's setup can be complex for beginners
2. **Understanding the codebase:** Django is a large project with many components
3. **Writing tests:** Django's testing framework has a learning curve
4. **Navigating the contribution process:** The workflow can seem overwhelming at first

Our walkthrough addresses these challenges by providing clear, step-by-step instructions.

Beyond Your First Contribution

After your first contribution is merged, you can:

1. **Take on more complex tickets:** As you gain confidence, try more challenging issues
2. **Become a regular contributor:** Build relationships with core developers
3. **Help with documentation:** Improving docs is always valuable
4. **Review other pull requests:** Helping review others' code is a great way to learn
5. **Join the community:** Participate in Django events and discussions

Additional Resources

- [Django Documentation](#)
- [Django Source Code](#)
- [Django's Trac Instance](#)
- [Django Forum](#)
- [Django Discord](#)
- [Django Chat Podcast](#)
- [Django News](#)

Conclusion

Contributing to Django is a rewarding experience that helps you grow as a developer while improving a framework used by millions. By starting with beginner-friendly tickets like #35924, you can gradually build confidence and expertise to tackle more complex issues.

Remember that everyone in the Django community was once a first-time contributor. Don't be afraid to ask questions and learn from the feedback you receive.

Happy contributing!