GitHub Tutorial: A Comprehensive Guide

Introduction

GitHub is a widely used platform for version control and collaboration, allowing multiple developers to work on a project simultaneously.

This tutorial will cover the fundamental Git and GitHub commands to help you manage repositories effectively.

Setting Up Git

Before using Git, you need to configure your local environment with your GitHub credentials:

```
git config --global user.email "your_email@example.com"
git config --global user.name "Your Name"
```

If this is your first time using Git, you may also need to authenticate your account when pushing changes.

Adding a Local Project to GitHub

1. Create a new repository on GitHub without initializing it with any files.
2. Navigate to your project folder in the terminal:
   ```bash
   cd /path/to/your/project
   ```
3. Initialize a new Git repository:
   ```bash
   git init
   ```
   Or, to set the branch name explicitly:
   ```bash
   git init -b main
   ```
4. Prevent large files from being uploaded (>100MB):
   ```bash
   find . -size +100M | cat >> .gitignore
   ```

   If a large file is accidentally staged, remove it:

   ```bash
   git rm --cached filename
   ```

5. Stage all files for commit:

   ```bash
   git add .
   ```

6. Check the status of staged files:

   ```bash
   git status
   ```

7. Commit changes with a descriptive message:

   ```bash
   git commit -m "Initial commit"
   ```

8. Link your local repository to the GitHub repository:

   ```bash
   git remote add origin https://github.com/yourusername/your-repo.git
   ```

9. Push changes to GitHub:

   ```bash
   git push -u origin main
   ```

Updating an Existing Repository

1. Ensure your local repository is up to date:

   ```bash
   git pull origin main
   ```

2. Stage new or modified files:

   ```bash
   git add .
   ```

3. Commit changes:

   ```bash
   git commit -m "Updated feature X"
   ```

4. Push changes:

   ```bash
   git push origin main
   ```

Cloning a Repository

If you want to work on an existing GitHub repository:

```bash
git clone https://github.com/username/repository.git
```

Navigate into the cloned repository:

```bash
cd repository
```

Working with Branches

Creating a New Branch

```bash
git branch new_branch_name
```

Switch to the new branch:

```bash
git checkout new_branch_name
```

Or create and switch in one command:

```bash
git checkout -b new_branch_name
```

Checking Active Branch

```bash
git branch
```

Switching Branches

```bash
git checkout branch_name
```

Or:

```bash
git switch branch_name
```

Merging a Branch to Main

1. Switch to the main branch:
   ```bash
   git checkout main
   ```
2. Merge the feature branch:
   ```bash
   git merge other_branch
   ```

Handling Experimental Changes

Discarding Unwanted Changes

```bash
git reset --hard origin/main
```

Keeping Experimental Changes in a Branch

1. Create a new branch:
   ```bash
   git branch experimental_branch
   ```

2. Switch to it:

```bash
git checkout experimental_branch
```

3. Make changes, then commit:

```bash
git add .
git commit -m "Testing feature X"
```

4. Push the branch to GitHub:

```bash
git push origin experimental_branch
```

5. If changes are good, merge them back into `main`:

```bash
git checkout main
git pull origin main
git merge experimental_branch
```

6. Resolve conflicts if necessary, then:

```bash
git add resolved_files
git merge --continue
git commit -m "Merged experimental changes"
git push origin main
```

Collaborating on a Shared Repository

Always **pull the latest changes** before making edits:

```bash
git pull origin main
```

After making changes, push them:

```bash
git add .
git commit -m "Updated X"
git push origin main
```

To check the repository's commit history:

```bash
git log --oneline --graph --decorate --all
```

Conclusion

This tutorial covers essential Git commands for managing repositories, branching, and collaboration. Regularly pulling, committing with detailed messages, and working in branches will keep your repository well-maintained and organized.

For more detailed Git documentation, visit [GitHub Docs](https://docs.github.com/en).