Ai Tools For Coding And How To Use Them Step By Step Guide

There are several **AI tools for coding** that can significantly boost your productivity, automate repetitive tasks, help you write better code, and even assist in debugging. These tools range from **code completion** and **refactoring** to **automated testing** and **documentation generation**. Here's a guide on how to use them, step by step.

---

1. GitHub Copilot

GitHub Copilot is one of the most popular AI tools for coding, built by GitHub and OpenAI. It helps developers by providing code suggestions, documentation, and completing functions automatically.

How to Use GitHub Copilot:

1. **Install GitHub Copilot**:
   - Open **Visual Studio Code (VS Code)**.
   - Go to the GitHub Copilot extension page and click **Install**.
2. **Enable Copilot**:
   - After installation, you might need to log in with your **GitHub account** and enable Copilot by selecting **GitHub Copilot** in the VS Code extension panel.
3. **Start Coding**:
   - Start typing your code in a supported language (e.g., Python, JavaScript, TypeScript, Ruby, etc.).
   - Copilot will automatically provide suggestions based on the context of your code.
4. **Accept Suggestions**:
   - You can accept a suggestion by pressing **Tab**. If multiple suggestions are available, you can cycle through them using the arrow keys.

5. **Write Functions**:
    - Copilot can also help you write entire functions. Just type a comment or a description, like:

      ```
      Python:-

      # Function to calculate the factorial of a number
      ```

      Copilot will generate the code for you.

Example:

```
Python:-

# Function to calculate factorial of a number
def factorial(n):
    # Copilot will suggest code here
```

---

2. Tabnine

Tabnine is another AI code completion tool that integrates with popular code editors like VS Code, JetBrains, and Sublime Text.

How to Use Tabnine:

1. **Install Tabnine**:
    - Visit Tabnine's website and download the extension for your IDE (VS Code, JetBrains, etc.).
2. **Configure**:
    - Once installed, open your IDE, and Tabnine should be active. You can configure settings through the IDE to adjust how aggressively it suggests completions.
3. **Start Coding**:
    - Tabnine uses machine learning to analyze your code and provide real-time suggestions.
    - The more you use it, the more personalized the suggestions become based on your coding style.
4. **Accept Suggestions**:
    - Press **Tab** or **Enter** to accept the suggestion.
    - You can also trigger Tabnine manually with `Ctrl + Space` (Windows/Linux) or `Cmd + Space` (Mac).

Example:

- Type `for` in Python, and Tabnine will suggest a loop structure based on your previous code and coding style.

---

3. Kite

Kite is another AI-powered code completion tool that provides suggestions and documentation as you code. It works with multiple languages like Python, JavaScript, Go, and more.

How to Use Kite:

1. **Install Kite**:
   - Visit Kite's website and download the installer for your operating system.
   - Follow the instructions to install Kite.
2. **Install the Kite Plugin for VS Code**:
   - Open VS Code, go to the Extensions marketplace, and search for **Kite**. Install the plugin.
3. **Use Kite for Code Suggestions**:
   - Once the setup is complete, Kite will automatically start suggesting completions as you type.
   - It also shows relevant documentation for functions and classes.
4. **Code Examples**:
   - Type `def` in Python, and Kite will suggest function templates.
   - In JavaScript, Kite can suggest API calls and help with writing asynchronous code.

---

4. Codex by OpenAI

Codex is the engine behind GitHub Copilot, but you can also use it directly through OpenAI's API to generate code from text prompts.

How to Use Codex:

1. **Create an OpenAI Account**:
   - Go to OpenAI and create an account.
   - Once logged in, get your **API key**.
2. **Set Up the OpenAI API**:
   - Install the OpenAI Python package:

```
bash
CopyEdit
pip install openai
```

- Set up your API key in your code:

```
Python:-

import openai
openai.api_key = "your-api-key"
```

3. **Use Codex for Code Generation**:
   - Pass a prompt to Codex and it will generate code based on the instructions. For example:

```
Python:-

response = openai.Completion.create(
  engine="code-davinci-002",  # Codex engine
  prompt="Write a Python function to calculate Fibonacci
sequence",
  max_tokens=100
)
print(response.choices[0].text.strip())
```

4. **Integrate into Your Projects**:
   - Use Codex to automate repetitive coding tasks, generate documentation, or even write new features based on high-level descriptions.

Example:

- **Input Prompt**: "Create a function that reverses a string in Python."
- **Output**:

```
Python:-

def reverse_string(s):
    return s[::-1]
```

---

5. AI-Powered Refactoring Tools (e.g., Sourcery for Python)

Refactoring tools like **Sourcery** help you write cleaner, more efficient code by suggesting better ways to implement existing logic.

How to Use Sourcery (Python):

1. **Install Sourcery**:
   - Install Sourcery via pip:

```
pip install sourcery-cli
```

2. **Integrate with VS Code**:

- Sourcery has a plugin for VS Code. Install it from the VS Code marketplace.

3. **Refactor Code Automatically**:
   - Sourcery will analyze your code and suggest improvements, like simplifying complex expressions or replacing loops with more efficient data structures.
4. **Use Sourcery's Suggestions**:
   - It will show suggestions like:

```
Python:-

# Old code
if x == True:
    return 'Yes'

# Refactored by Sourcery
return 'Yes' if x else 'No'
```

---

6. AI-Powered Testing Tools (e.g., Test.ai)

Test.ai automates testing using AI by generating tests for your app. It simulates user interactions and ensures your app behaves as expected.

How to Use Test.ai:

1. **Sign Up for Test.ai**:
   - Go to the Test.ai website and create an account.
2. **Connect to Your Application**:
   - Test.ai supports integration with web apps, mobile apps (iOS/Android), and APIs.
3. **Automate Tests**:
   - Once integrated, Test.ai will automatically generate tests based on user behavior or given scenarios.
4. **Run Tests & Optimize**:
   - It runs tests on your codebase and returns issues or anomalies for improvement.

Conclusion

AI tools are revolutionizing the way developers write and manage code. By integrating these tools into your workflow, you can automate repetitive tasks, improve your coding speed, and ensure higher code quality. Here's a recap:

Tool	Functionality
GitHub Copilot	Code suggestions, completions, and documentation
Tabnine	Code completions across multiple IDEs

Tool	Functionality
Kite	AI-powered code completions and documentation
Codex (OpenAI)	Code generation from prompts (via API)
Sourcery	Refactor Python code for efficiency
Test.ai	AI-driven test creation and automation