Inline Chat with GitHub Copilot

Quick, contextual AI assistance directly in your editor without leaving your code.

🎯 Overview

Inline Chat provides immediate AI assistance at your cursor position: - Quick questions about code - Targeted refactoring - Code generation in place - Explanations without context switching

---

⌨️ Keyboard Shortcuts

Action	macOS	Windows/Linux
Open Inline Chat	`Cmd + I`	`Ctrl + I`
Accept suggestion	`Enter` or `Tab`	`Enter` or `Tab`
Dismiss	`Escape`	`Escape`

🚀 Basic Usage

Quick Code Generation

1. Place cursor where you want code
2. Press `Cmd + I` (Mac) or `Ctrl + I` (Windows)
3. Type your request
4. Press Enter to generate
5. Accept or modify the result

Example:

```
public class ExpenseService {
    // Place cursor here, press Cmd+I
    // Type: "add a method to calculate total expenses by category"
}
```

Inline Explanation

1. Select code you want explained
2. Press `Cmd + I`
3. Type: "explain" or "what does this do?"
4. Read the explanation inline

Quick Fixes

1. Select problematic code
2. Press `Cmd + I`
3. Type: "fix" or describe the issue
4. Review and accept the fix

💬 Common Inline Chat Prompts

Code Generation

```
add a validation method for expense amount
```

```
create a constructor with all fields
```

```
add getter and setter for this field
```

Refactoring

```
extract this to a separate method
```

```
convert to stream API
```

```
simplify this conditional logic
```

Documentation

```
add javadoc
```

```
add inline comments explaining this logic
```

Testing

```
generate unit test for this method
```

```
add test case for null input
```

🔧 Contextual Actions

With Selection

Select code first, then `Cmd + I`:

```
Prompt: make this null-safe
```

```
Prompt: optimize for performance
```

```
Prompt: add error handling
```

Without Selection

Cursor at position, `Cmd + I`:

```
Prompt: add a method to find expenses by date range
```

```
Prompt: implement toString method
```

📝 Practical Examples

Example 1: Add Validation

Before:

```
public void setAmount(double amount) {
    this.amount = amount;
}
```

Inline Chat prompt: `add validation for positive amount`

After:

```
public void setAmount(double amount) {
    if (amount < 0) {
        throw new IllegalArgumentException("Amount must be non-negative");
    }
    this.amount = amount;
}
```

Example 2: Convert Loop to Stream

Before:

```
List<Expense> filtered = new ArrayList<>();
for (Expense expense : expenses) {
    if (expense.getAmount() > 100) {
        filtered.add(expense);
    }
}
```

Select code, Inline Chat: `convert to stream`

After:

```
List<Expense> filtered = expenses.stream()
    .filter(expense -> expense.getAmount() > 100)
    .collect(Collectors.toList());
```

Example 3: Add Builder Pattern

Cursor inside class, Inline Chat: `add builder pattern for this class`

```
public static class Builder {
    private Long id;
    private String description;
    private double amount;

    public Builder id(Long id) {
        this.id = id;
        return this;
    }
    // ... more builder methods

    public Expense build() {
        return new Expense(this);
    }
}
```

⚡ Quick Reference Prompts

Task	Prompt
Add null check	`make null-safe`
Add logging	`add logging`
Add try-catch	`add error handling`
Convert to Optional	`use Optional`
Add validation	`validate input`
Simplify	`simplify`
Explain	`explain`

Task	Prompt
Fix bug	`fix`
Add test	`generate test`
Add documentation	`add javadoc`

🔄 Inline Chat vs Chat Panel

Feature	Inline Chat (`Cmd+I`)	Chat Panel
Context	Current cursor/selection	Can reference multiple files
Speed	Very fast	More comprehensive
Output	Directly in editor	Side panel with code blocks
Best for	Quick, targeted changes	Complex questions, exploration

When to Use Inline Chat

- Quick code generation at cursor
- Small refactoring tasks
- Adding single methods
- Quick explanations
- Simple fixes

When to Use Chat Panel

- Multi-file questions
- Architecture discussions
- Complex debugging
- Learning new concepts
- Detailed explanations

🔧 Practical Exercises

Exercise 1: Quick Generation

1. Create an empty class `Budget`
2. Use `Cmd+I` to add: "private fields for name, amount, startDate, endDate"
3. Use `Cmd+I` to add: "constructor with all fields"
4. Use `Cmd+I` to add: "getters and setters"

Exercise 2: Inline Refactoring

1. Write a method with nested if-else statements
2. Select the code
3. Use `Cmd+I`: "simplify using early returns"
4. Accept the refactored version

Exercise 3: Quick Documentation

1. Open a class with undocumented methods
2. Place cursor above each method
3. Use `Cmd+I`: "add javadoc"
4. Accept generated documentation

✅ Best Practices

- **Be Concise**: Short prompts work best for inline chat
- **Select Relevant Code**: Better context = better results
- **Iterate Quickly**: Use `Cmd+I` again to refine
- **Accept Partially**: You can edit the result before accepting
- **Know When to Switch**: Use Chat Panel for complex tasks

🔗 Related Resources

- Copilot Chat Cookbook - More chat patterns

- Slash Commands - Commands available in chat
- Keyboard Shortcuts - Complete shortcut reference