

Exercise 3: Coding Mode for VS Code

🕒 15 minutes • LOCAL & PHI-SAFE

Learning Objective

Create a custom mode that transforms your spoken descriptions into code, comments, and technical documentation—optimized for software development workflows.

What This Mode Does

Describe a Function

"Create a function called validate email that takes a string and returns true if it's a valid email format"

```
function validateEmail(email)
{ const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/; return
  regex.test(email); }
```

Add a Comment

"Add a comment explaining that this function handles user authentication and should be called before accessing protected routes"

```
// This function handles user
authentication // Call before
accessing protected routes
```

Step-by-Step Setup

1

Create New Custom Mode

Settings → Modes → Create Custom Mode

Name: "Coding Assistant" or "Dev Mode"

2

Select Local Models

Voice Model: Fast or Standard (LOCAL)

Language Model: Any LOCAL model available

3

Configure Auto-Activation

Add rules for your development tools:

Applications:

- ✓ Visual Studio Code
- ✓ Cursor
- ✓ Terminal / iTerm2
- ✓ Any JetBrains IDE

4

Enable All Context Types

- ☒ **Application Context** - Detects language from file extension
- ☒ **Selected Text** - Reference existing code
- ☒ **Clipboard** - Use copied code as style reference

5

Enter AI Instructions

<role>

You are a senior software developer assistant.

</role>

<instructions>

The User Message is dictated speech about code. Determine intent and format appropriately:

IF describing a function/class → Output clean, ready-to-paste code

IF asking for a comment → Output as code comments (// or # based on language)

IF explaining changes → Output the modified code

IF asking a question → Provide a brief technical answer

Detect programming language from:

1. Application Context (file extension, IDE)
2. Clipboard Context (existing code style)
3. Keywords in dictation ("Python function", "JavaScript array")

</instructions>

<requirements>

- Output ONLY code or comments - no explanations unless asked
- Match the detected language syntax exactly
- Use consistent naming conventions (camelCase for JS, snake_case for Python)
- If language unclear, default to JavaScript
- Keep variable/function names exactly as spoken

</requirements>

Test Your Coding Mode

Open VS Code with a JavaScript file, then try these:

Test 1: Create a Function

"Create a function called calculate total that takes an array of prices and returns the sum"

```
function calculateTotal(prices)
{ return prices.reduce((sum, price)
=> sum + price, 0); }
```

Test 2: Add Comments

"Add a comment above this function explaining it validates user input and throws an error if invalid"

```
// Validates user input // Throws an
error if validation fails
```

Test 3: With Selected Code

Select existing code, then: "Add error handling to this function"

Test 4: Quick Question

"What's the difference between let and const"

Pro Tip: Copy a block of your existing code before dictating. The AI will use it as a style reference for consistent formatting.

Vocabulary Additions

Add these to Settings → Vocabulary for better recognition:

```
API, JSON, async, await, useState, useEffect, const, let, var, npm, yarn,
git, GitHub, TypeScript, JavaScript, Python, React, Node, Express, MongoDB,
PostgreSQL, SQL, HTML, CSS, REST, GraphQL
```

Completion Checklist

- ☐ Created "Coding Assistant" mode with LOCAL models
- ☐ Added auto-activation for VS Code, Cursor, Terminal
- ☐ Enabled all three context types
- ☐ Pasted the coding prompt
- ☐ Added technical vocabulary words
- ☐ Tested creating a function
- ☐ Tested adding comments

Next Level: Combine this with Claude Code! Dictate complex instructions with SuperWhisper, paste into Claude Code terminal, watch it execute. Voice → Claude → Code in seconds.