Context Is All You Need: AI Engineering with the Petoi 'Bittle X' Robot Dog

Overview

In this workshop, students will experience firsthand how AI becomes powerful through context. Starting with an AI chat interface that only knows how to send a few commands to the robot dog (Petoi 'Bittle X'), students will progressively teach the AI new capabilities by providing it with more context about the robot's full range of abilities. This hands-on approach demonstrates the core principle of AI Engineering: connecting AI to specific knowledge to solve real problems.

Key Objectives

By the end of this workshop, students will:

1. **Understand AI Engineering**: Experience how providing context to AI transforms its capabilities from limited to comprehensive.
2. **Build Iteratively**: Add new robot capabilities one by one, seeing immediate results with each addition.
3. **Read Technical Documentation**: Learn to extract key information from documentation and translate it for AI use. This is a core AI Engineering skill. Understanding a system and distilling that knowledge into context that makes AI powerful.
4. **Practice Responsible AI**: Implement transparency features showing exactly what commands AI sends to hardware.
5. **Problem-Solve with AI Tools**: Use GitHub Copilot and other AI assistants to understand and modify code. Students learn that giving up is not an option when AI tools can help them think through problems and find solutions.

Materials

- Petoi 'Bittle X' Robot Dog (1 per 5-6 students).
- Laptops with Python and VS Code installed.
- The Python files required to control the robot dog found here.
- The Python packages: `openai`, `python-dotenv`, and `pyserial` installed via pip.
- An OpenAI API key in a `.env` file. You can get one here. Create a `.env` file in the `lib` folder with the contents `OPENAI_API_KEY=YOUR_API_KEY_HERE`. The program will auto-create this file if missing.
- The robot dog command reference sheet available in the `challenges` folder as `robot-dog-commands.md`.

Background: The Power of Context

Imagine asking someone to "make coffee" without any context. They might not know where the coffee maker is, what kind of coffee you like, or even what coffee is. But give them a manual for your specific coffee maker and tell them your preferences, and suddenly they can make perfect coffee every time.

This is exactly what we're doing with AI and the robot dog. AI Models are brilliant but need specific context to be useful. That's where AI Engineers come in, they connect AI to the specific knowledge it needs.

Why This Matters

As proclaimed by Andrej Karpathy (former Director of AI at Tesla and founding member of OpenAI), we're entering the age of AI Engineering. The future isn't just about using AI, it's about making AI useful by connecting it to real-world systems and knowledge.

Traditional software engineers write code to directly control systems. AI Engineers take a different approach, they write code that teaches AI how to control those systems. It's still engineering, but instead of coding every action, you're providing AI with the knowledge it needs to figure things out.

Software engineers, product managers, data analysts, data scientists, and more are all becoming AI Engineers when they focus on empowering AI for specific use cases. Even tools like Microsoft Copilot Studio now let people do this without writing much code at all.

In this workshop, students will experience both worlds. They'll see how traditional code controls the robot directly, then transform into an AI Engineer by teaching AI to control it.

Workshop Instructions

Setup

1. **Connect the robot dog**:
   - **USB-C**: Use the included USB-C cable to connect the robot dog to your laptop. This is the most reliable connection.
   - **Bluetooth (Alternative)**: You can also connect via Bluetooth, but the connection can be spotty.
2. **Open VS Code**: Navigate to the workshop folder and open the `challenges` folder.
3. **API Key Setup**: The program will automatically create a `.env` file in the `lib` folder if one doesn't exist. Students need to edit this file and replace `your_key_here` with their actual OpenAI API key.

Workshop Structure

This workshop uses a progressive challenge system where each challenge builds on the previous one:

- **Challenge 1**: Teaching AI New Commands (Context Engineering)
- **Challenge 2**: Responsible AI Transparency
- **Challenge 3**: Command Sequences
- **Challenge 4**: RAG Implementation (Simplified Retrieval-augmented Generation (RAG))
- **Challenge 5**: Robot Arm Control (OPTIONAL - requires Bittle X+Arm model)

Students work through challenges in the `challenges` folder while all system files remain hidden in the `lib` folder.

Note for Facilitators: Challenge 5 requires the **Bittle X+Arm** model (robot dog with arm attachment). If your robots don't have arms, skip this challenge. For more information about the robot arm, see:

- Bittle X+Arm Setup Guide
- Robot Arm Documentation

Facilitator Instructions

Opening Demonstration (10 minutes)

1. **Open Challenge 1**: Show `challenge-1-commands.py` in VS Code

2. **Run the program**: Click the play button and demonstrate the limited AI

3. **Try these commands**:

```
You: stand up
AI: Standing up! [Robot stands]

You: sit down
AI: Making the robot sit! [Robot sits]

You: jump
AI: I don't know that command yet. I can: stand up, sit down, and rest
```

4. **Show the command reference**: Open `robot-dog-commands.md` to show the 50+ available commands

5. **Set the challenge**: "Your job as AI Engineers is to teach the AI these commands by adding them to the code."

Challenge 1: Teaching AI New Commands (15 minutes)

Learning Goal: Understand how context makes AI powerful

Instructions for Students:

1. Look at the `ROBOT_COMMANDS` dictionary in the code
2. Open `robot-dog-commands.md` for available commands
3. Add "jump" and other commands to make the AI smarter
4. Test by running the program and trying new commands

Facilitator Notes:

- Emphasize that students should use GitHub Copilot for help
- Walk around and help students who are stuck
- Point out how each new command immediately makes the AI more capable

Challenge 2: Responsible AI Transparency (10 minutes)

Learning Goal: Make AI systems transparent and trustworthy

Instructions for Students:

1. Change `SHOW_COMMANDS = False` to `True`
2. Run the program and notice the transparency features
3. Try various commands and observe what the AI actually sends to the robot

Facilitator Notes:

- Discuss why transparency matters for AI controlling physical devices
- Point out the difference between what AI says and what it actually does
- Connect to real-world examples like self-driving cars

Challenge 3: Command Sequences (15 minutes)

Learning Goal: Create complex behaviors from simple building blocks

Instructions for Students:

1. Change `ENABLE_SEQUENCES = True`
2. Add sequence commands using the format shown in comments
3. Create creative sequences like "dance" or "morning routine"
4. Test the multi-step behaviors

Facilitator Notes:

- Help students understand the difference between single commands and sequences
- Encourage creativity in sequence design
- Show how simple commands combine into complex behaviors

Challenge 4: RAG Implementation (15 minutes)

Learning Goal: Implement Retrieval Augmented Generation (RAG) - simplified version

Instructions for Students:

1. Change `ENABLE_FULL_API = True`
2. Implement the `fetch_api_documentation()` function
3. Test the AI with access to full robot capabilities

Facilitator Notes:

- This is a simplified RAG implementation - real RAG systems are more complex
- Help students understand the concept of connecting AI to live data sources
- Some students may need more guidance on network programming
- Emphasize how this connects AI to live, up-to-date information
- Note: This is a basic example - production RAG involves vector databases, embeddings, and more sophisticated retrieval

Challenge 5: Robot Arm Control (15 minutes) - OPTIONAL

Prerequisites: Bittle X+Arm model required (robot dog with arm attachment)

Learning Goal: Master complex robot control through expert prompting and documentation reading

Instructions for Students:

1. Read `robot-arm-commands.md` to understand arm capabilities
2. Click ▶ to run Challenge 5 - AI becomes robot arm expert
3. Find a small object in the room
4. Use natural language to control both movement and arm
5. Complete: Navigate → Pick up → Move → Put down
6. Try complex multi-step prompts

Facilitator Notes:

- **Check robot model first** - only works with Bittle X+Arm
- This is pure prompt engineering - no code changes needed
- Emphasize reading documentation before prompting
- Safety reminder: keep hands away from robot claws
- Encourage complex, natural language prompts
- Show how documentation reading enables effective AI interaction
- Skip this challenge if robots don't have arm attachments

Key Teaching Points

Context Engineering

Demonstrate how the same AI becomes dramatically more capable when given better context. This is the core principle of AI Engineering.

GitHub Copilot Usage

Actively encourage students to use GitHub Copilot throughout the workshop. The terminal in VS Code serves as the chat window for this workshop experience.

For Facilitators: Students should learn about GitHub Copilot modes. Direct them to the VS Code Copilot Chat documentation to understand the different interaction modes:

- **Ask Mode** (Safest): Provides suggestions without making changes to files
- **Edit Mode** (Safe): Only modifies the currently open file
- **Agent Mode** (Most Powerful): Can make changes across multiple files. Most useful for students but requires responsibility and careful review

Show students how to:

- Click the Copilot button in VS Code to open the chat panel
- Ask specific questions about their code
- Use different modes appropriately for their skill level
- Review any suggested changes before accepting them

Problem-Solving Mindset

When students get stuck, guide them to:

1. Read error messages carefully
2. Ask GitHub Copilot for help
3. Break problems into smaller pieces
4. Ask instructors for guidance, not answers

Responsible AI

Use Challenge 2 to discuss:

- Why transparency matters in AI systems
- How to build trust with users
- The importance of showing what AI is actually doing

Troubleshooting

Robot Connection Issues:

- Check USB cable connection
- Verify robot is powered on (green light)
- Try reconnecting the cable
- Switch to Bluetooth if USB fails

API Key Problems:

- Check that `.env` file is in the `lib` folder
- Verify the API key format is correct
- Ensure no extra spaces in the `.env` file

Import Errors:

- Verify all files are in the correct folders
- Check that `lib` folder contains all required files
- Restart VS Code if needed

Wrap-up Discussion (10 minutes)

Key Questions:

- How did adding more context change what the AI could do?
- Why is transparency important when AI controls physical devices?
- How could you apply these AI Engineering principles to other projects?
- What did you learn about using AI tools to solve problems?

Extension Ideas:

- Add custom robot behaviors
- Create voice control using speech recognition
- Connect multiple robots for coordinated actions
- Build a web interface for robot control

Assessment

Students demonstrate understanding by:

- Successfully adding multiple robot commands (Challenge 1)
- Explaining the importance of AI transparency (Challenge 2)
- Creating working command sequences (Challenge 3)
- Implementing basic network requests (Challenge 4)
- Using GitHub Copilot to solve coding problems throughout
- **Optional**: Controlling robot arm through expert prompting (Challenge 5 - if arm available)