LESSON 7 · 19 NOVEMBER 2026

Applied Systems Sciences 1

LLM APIs from Python

Copilot and Claude Code at work, then a new project with uv that asks a language model

David Maier
Department of Environmental Systems Sciences
University of Graz

A space-time diagram, drawn by a coding agent

The brief is already written

> *“the space-time diagram (one row per step), the average speed, the number of standing cars.”*

ODD.md, element 4, Observation

An agent reads `ODD.md` like any other reader. A good ODD is a good brief.

THE PROMPT

Read ODD.md and model.py. Under Observation the ODD names a space-time diagram. Make it as a picture: write a new file plot.py that runs simulate() from model.py with 20 cars, 30% dawdling, 200 steps and seed 1, and draws one row per step with time going down and one column per cell. An empty cell is white, a car has the colour of its speed. Save the picture as jam.png. Use matplotlib and add it with uv. Do not change model.py. Then run plot.py.

In VS Code: give the prompt to your **coding agent**. Read every change before you keep it.

Live: Copilot and Claude Code

Live in class: the presenter's screen appears here.

How your code asks a language model

1 Send

Your key and a message, to an address on the web.

→

2 Write

The server checks the key, the model writes.

→

3 Read

JSON comes back, and Python makes it a dictionary.

Your code sends this JSON

```
{
  "messages": [
    {
      "role": "user",
      "content": "What is emergence?"
    }
  ]
}
```

The answer, shortened

```
{
  "model": "xiaomi/mimo-v2.6-flash",
  "choices": [{
    "message": {
      "role": "assistant",
      "content": "Emergence is …"
    }
  }],
  "usage": {"total_tokens": 49}
}
```

Your key

Now **Open asw1-ws26.dev.davidmaier.at/llm-key and log in with GitHub.**

It shows your key as two lines to copy into your code.

A password for a program

The key says who is asking. The course pays for every call, so a key allows 30 calls a minute and 400 a day.

Keep it to yourself

Never push a key to GitHub. If it got out, make a new one on the same page: the old one stops working.

With a provider of your own (OpenAI, Anthropic, OpenRouter) a key is money: whoever has it can spend it.

A new project with uv

1

Open

In VS Code: **File, Open Folder...**, make a new folder `llm-demo` and open it.

→

2

Create

Terminal, New Terminal, then:

```
uv init
```

→

3

Run

```
uv run main.py
```

It prints “Hello from llm-demo!”.

→

4

Add

```
uv add requests
```

A package that sends requests to the web.

`pyproject.toml` says what your project is and which packages it needs. `uv add` writes a package in and installs it into `.venv`.

Live: our first API call

Live in class: the presenter's screen appears here.

The code from today

Asking the model

```python
import requests

URL = "…"    # both lines
KEY = "…"    # from /llm-key


def ask(question):
    message = {"role": "user", "content": question}
    response = requests.post(
        URL,
        headers={"Authorization": "Bearer " + KEY},
        json={"messages": [message]},
    )
    answer = response.json()
    return answer["choices"][0]["message"]["content"]
```

Using the answer

```python
def main():
    print(ask("What is emergence? One sentence."))

    reply = ask(
        "A lake has 24 fish. You may catch 0 to 5. "
        "How many do you catch? "
        "Answer with one number only."
    )
    catch = int(reply)
    print("I catch", catch)


main()
```

Run it with `uv run main.py`, then run it again: other words, and maybe another catch.

Python challenge: a function

```
def brake(speed, gap):
    if speed > gap:
        speed = gap
    return speed

v = 5
w = brake(v, 2)
print(v, w)
```

What will be printed?

Python challenge: a dataclass

```python
from dataclasses import dataclass

@dataclass
class Car:
    position: int
    speed: int = 0

a = Car(position=3)
b = Car(position=7, speed=2)
a.speed = b.speed + 1

print(a.position, a.speed, b.speed)
```

What will be printed?

Today, and next week

You can now

- start a project with `uv init` and add a package
- ask a language model from your own code
- read its answer from a dictionary and use it
- give a coding agent a brief and read what it changed

Before next Thursday

- [ ] Your `main.py` answers. Ask it a question of your own.
- [ ] If you like: give a coding agent the space-time prompt, and check what it draws. Or work with the coding agent on making the LLM demo more interesting.

Thu 26 Nov **Project kick-off.** The topics go online with lesson 8. Groups, a research question, a first ODD and a shared repository.