Python

Overview

Get started
Quickstart
Customization

Core capabilities
Agent harness
Backends
Subagents
Human-in-the-loop
Long-term memory
Middleware

Command line interface
Use the CLI

Get started

Quickstart

Copy page

Build your first deep agent in minutes

This guide walks you through creating your first deep agent with planning, file system tools, and subagent capabilities. You'll build a research agent that can conduct research and write reports.

Prerequisites

Before you begin, make sure you have an API key from a model provider (e.g., Anthropic, OpenAI).

Step 1: Install dependencies

pip uv poetry

```
pip install deepagents tavily-python
```

Step 2: Set up your API keys

```
export ANTHROPIC_API_KEY="your-api-key"
export TAVILY_API_KEY="your-tavily-api-key"
```

Step 3: Create a search tool

```
import os
from typing import Literal
from tavily import TavilyClient
from deepagents import create_deep_agent

tavily_client = TavilyClient(api_key=os.environ["TAVILY_API_KEY"])

def internet_search(
    query: str,
    max_results: int = 5,
    topic: Literal["general", "news", "finance"] = "general",
    include_raw_content: bool = False,
):
    """Run a web search"""
    return tavily_client.search(
        query,
        max_results=max_results,
        include_raw_content=include_raw_content,
        topic=topic,
    )
```

Step 4: Create a deep agent

```
# System prompt to steer the agent to be an expert researcher
research_instructions = """You are an expert researcher. Your job is to condu

You have access to an internet search tool as your primary means of gathering

## `internet_search`

Use this to run an internet search for a given query. You can specify the max
"""

agent = create_deep_agent(
    tools=[internet_search],
    system_prompt=research_instructions
)
```

Step 5: Run the agent

```
result = agent.invoke({"messages": [{"role": "user", "content": "What is lang

# Print the agent's response
print(result["messages"][-1].content)
```

What happened?

Your deep agent automatically:

1. **Planned its approach**: Used the built-in `write_todos` tool to break down the research task
2. **Conducted research**: Called the `internet_search` tool to gather information
3. **Managed context**: Used file system tools (`write_file` , `read_file`) to offload large search results
4. **Spawned subagents** (if needed): Delegated complex subtasks to specialized subagents
5. **Synthesized a report**: Compiled findings into a coherent response

Next steps

Now that you've built your first deep agent:

- **Customize your agent**: Learn about **customization options**, including custom system prompts, tools, and subagents.
- **Understand middleware**: Dive into the **middleware architecture** that powers deep agents.
- **Add long-term memory**: Enable **persistent memory** across conversations.
- **Deploy to production**: Learn about **deployment options** for LangGraph applications.

Edit the source of this page on GitHub.

Connect these docs programmatically to Claude, VSCode, and more via MCP for real-time answers.

Was this page helpful? Yes No