# Use AG2 with Radium

AG2 1.x, the current successor to AutoGen, supports OpenAI-compatible endpoints through `OpenAIConfig`. This guide uses the current async `ag2.Agent` API.

## Before you start

You need Python 3.10 or newer, a Radium API key, and a terminal or PowerShell window.

## 1. Create a project and install AG2

macOS or Linux:

```
mkdir radium-ag2
cd radium-ag2
python3 -m venv .venv
source .venv/bin/activate
python -m pip install "ag2[openai]"
```

Windows PowerShell:

```
mkdir radium-ag2
cd radium-ag2
py -m venv .venv
.venv\Scripts\Activate.ps1
python -m pip install "ag2[openai]"
```

## 2. Set your Radium credentials

macOS or Linux:

```
export RADIUM_API_KEY="your-radium-api-key"
export RADIUM_MODEL="hal-1.0"
```

Windows PowerShell:

```
$env:RADIUM_API_KEY = "your-radium-api-key"
$env:RADIUM_MODEL = "hal-1.0"
```

# 3. Create `main.py`

```
import asyncio
import os

from ag2 import Agent
from ag2.config import OpenAIConfig



api_key = os.getenv("RADIUM_API_KEY")
if not api_key:
    raise SystemExit("Set RADIUM_API_KEY before running this program.")



async def main() -> None:
    assistant = Agent(
        "radium_assistant",
        prompt="Follow the user's instruction exactly.",
        config=OpenAIConfig(
            model=os.getenv("RADIUM_MODEL", "hal-1.0"),
            api_key=api_key,
            base_url="https://api.radium.cloud/v1",
            max_tokens=512,
            temperature=0,
            timeout=90,
        ),
    )
    reply = await assistant.ask("Reply with exactly: Radium connected.")
    if not reply.body or not reply.body.strip():
        raise RuntimeError("Radium returned no visible text.")
    print(f"RADIUM_RESPONSE: {reply.body.strip()}")



asyncio.run(main())
```

This connection example does not enable code-execution tools.

# 4. Run it

```
python main.py
```

Expected output:

```
RADIUM_RESPONSE: Radium connected.
```

# Choose a model

```
export RADIUM_MODEL="tycho-1.0"  # or hal-1.0 or clarke-1.0
python main.py
```

# Migrate an existing AG2 agent

Create a Radium configuration and pass it to the agent:

```
config = OpenAIConfig(
    model="hal-1.0",
    api_key=os.environ["RADIUM_API_KEY"],
    base_url="https://api.radium.cloud/v1",
    max_tokens=512,
)
agent = Agent("radium_assistant", config=config)
```

AG2 1.x removed the legacy `autogen.AssistantAgent` interface. If an older project imports `autogen`, it uses AG2 Classic and needs either the classic configuration format or a migration to the current API shown here.

# Troubleshooting

- `ImportError` for `autogen`: follow the current `ag2.Agent` imports shown above.
- Authentication errors: set the Radium key in the active terminal.
- `404`: check the base URL and model ID.
- Code execution concerns: do not add execution tools until they have been reviewed and sandboxed.

# Validation

Version note: these instructions were verified with Python 3.12.7 and `ag2[openai]==1.0.2` on August 20, 2026. You do not need that exact package version. Text generation and callable-tool execution passed with `hal-1.0`, `clarke-1.0`, and `tycho-1.0`.

Reference: AG2 model configuration.

# Next

- API quickstart, for calling Radium directly
- Tool calling, for the `tool_use` and `tool_result` contract

---

Created 2026-08-23 21:34:23 UTC by Admin
Updated 2026-08-23 21:53:01 UTC by Admin