

OMP to rekomendowany agent AI. Po instalacji podłączasz go do OmniRoute pluginem omnisync. Plugin sam pobiera listę modeli, więc nie edytujesz plików konfiguracyjnych ręcznie.

Czas: ok. 10 minut.

Wymagania

- Klucz API do OmniRoute ([jak zdobyć](#)).
- git.
- Bun $\geq$ 1.3.14. Instalator OMP poradzi sobie bez Buna (pobierze gotowy plik binarny), ale wtyczki instalują się przez Buna, więc warto go mieć.

Linux

1. Zainstaluj Buna:

```
curl -fsSL https://bun.sh/install | bash
```

2. Otwórz nowy terminal i zainstaluj OMP:

```
bun install -g @oh-my-pi/pi-coding-agent
```

Alternatywa bez Buna: `curl -fsSL https://omp.sh/install | sh`

3. Sprawdź instalację:

```
omp --version
```

Oczekiwany wynik: `omp/<wersja>`, np. `omp/18.10.3`.

Windows (PowerShell)

1. Zainstaluj Buna:

```
powershell -c "irm bun.sh/install.ps1 | iex"
```

2. Otwórz nowe okno PowerShell i zainstaluj OMP:

```
bun install -g @oh-my-pi/pi-coding-agent
```

- Alternatywa bez Buna: `irm https://omp.sh/install.ps1 | iex`
3. Sprawdź instalację:

```
omp --version
```

Podłączenie do OmniRoute

1. Zainstaluj wtyczki według strony [OMP: wtyczki](#). Bez wtyczki `omnisync` OMP nie zobaczy modeli OmniRoute.
2. Uruchom `omp` w katalogu projektu.
3. Wpisz `/omni setup` i podaj:
 - URL serwera: `https://omni.suncode.dev`
 - klucz API od działu administracji i bezpieczeństwa
4. Wpisz `/model` i wybierz `omni/paid-premium`. To firmowy model domyślny.

Plugin zapisuje klucz we własnych ustawieniach. Klucz nie trafia do `models.json`.

Sprawdzenie

W OMP wpisz:

```
/omni test paid-premium
```

Oczekiwany wynik: krótka odpowiedź modelu bez błędu.

Gdzie OMP trzyma konfigurację:

System	Katalog
Linux	<code>~/ .omp/</code>
Windows	<code>%USERPROFILE%\ .omp\</code>

Bezpieczeństwo: wyłącz tryb yolo

Domyślnie OMP działa w trybie **yolo**: sam zatwierdza każde narzędzie, także komendy w terminalu. Ustaw tryb, w którym agent pyta przed wykonaniem komendy.

1. Ustaw tryb zatwierdzania (Linux i Windows PowerShell):

```
omp config set tools.approvalMode write
```

2. Dodaj komendy, przed którymi OMP ma zawsze pytać albo które ma blokować:

```
omp config set bash.patterns '[{"match":"git
push*","approval":"prompt"}, {"match":"git reset
--hard*","approval":"prompt"}, {"match":"git
clean*","approval":"prompt"}, {"match":"rm
-rf*","approval":"prompt"}, {"match":"* |
sh","approval":"deny"}]'
```

Windows: PowerShell 5.1 gubi cudzysłowy " w tej komendzie. Zamiast niej wklej blok YAML z końca tej sekcji do pliku %USERPROFILE%\omp\agent\config.yml.

3. Uruchom ponownie omp.

Tryby (tools.approvalMode)

Tryb	Bez pytania	Pyta o zgodę
always-ask	tylko odczyt plików	zapis plików i komendy
write (zalecany)	odczyt i zapis plików w projekcie	komendy w terminalu, kod (eval), przeglądarka, subagenci
yo!o (domyślny)	wszystko	tylko wzorce z bash.patterns i komendy krytyczne

Niezależnie od trybu OMP zawsze pyta przy komendach krytycznych, m.in. `rm -rf /`, `sudo rm`, `mkfs`, `dd ... of=/dev/...`, `curl ... | sh`, `shutdown`.

Wzorce komend (bash.patterns)

- match: cała komenda, * oznacza dowolny ciąg znaków. `git push*` pasuje do `git push origin main`, ale nie do `git status`.
- approval: prompt (zapytaj), deny (zablokuj), allow (wykonaj bez pytania).
- Wygrywa pierwszy pasujący wzorec. Dlatego deny wpisuj przed allow.

Przykład: w trybie write testy mogą się wykonywać bez pytania:

```
omp config set bash.patterns '[{"match":"git
push*","approval":"prompt"}, {"match":"npm
test*","approval":"allow"}]'
```

`omp config set bash.patterns` nadpisuje całą listę. Za każdym razem podawaj komplet wzorców.

Tryb na jedną sesję

```
omp --approval-mode always-ask
```

Ustawienie na stałe sprawdzisz komendą `omp config get tools.approvalMode`.
Oczekiwany wynik: `write`.

Ten sam efekt daje wpis w `~/.omp/agent/config.yml` (Windows: `%USERPROFILE%\\.omp\agent\config.yml`):

```
tools:
  approvalMode: write
bash:
  patterns:
    - match: git push*
      approval: prompt
    - match: git reset --hard*
      approval: prompt
    - match: git clean*
      approval: prompt
    - match: rm -rf*
      approval: prompt
    - match: "*" | sh"
      approval: deny
```

Nie uruchamiaj **omp --auto-approve**. Ta flaga wyłącza wszystkie pytania o zgodę.

Problemy: [Rozwiązywanie problemów](#).