

Agent/Client Protocol

Sergey Ignatov
Director of Engineering, JetBrains



JETBRAINS



IntelliJ IDEA

IDE for Java and Kotlin developers

Java Kotlin Spring



PyCharm

IDE for Python developers and data scientists

Python Django Jupyter



PhpStorm

IDE for PHP developers

PHP Laravel Symfony



GoLand

IDE for Go developers

Go (Golang) JavaScript TypeScript



Rider

IDE for .NET and game developers

C# .NET ASP.NET



CLion

IDE for C and C++ developers

C C++ CMake



RustRover

IDE for Rust developers

Rust SQL JavaScript



WebStorm

IDE for JavaScript and TypeScript developers

JavaScript TypeScript React



RubyMine

IDE for Ruby and Rails developers

Ruby on Rails (RoR) Hotwire RuboCop



DataGrip

Tool for multiple databases

Databases SQL NoSQL



DataSpell

Your AI-powered tool for analyzing data and sharing insights

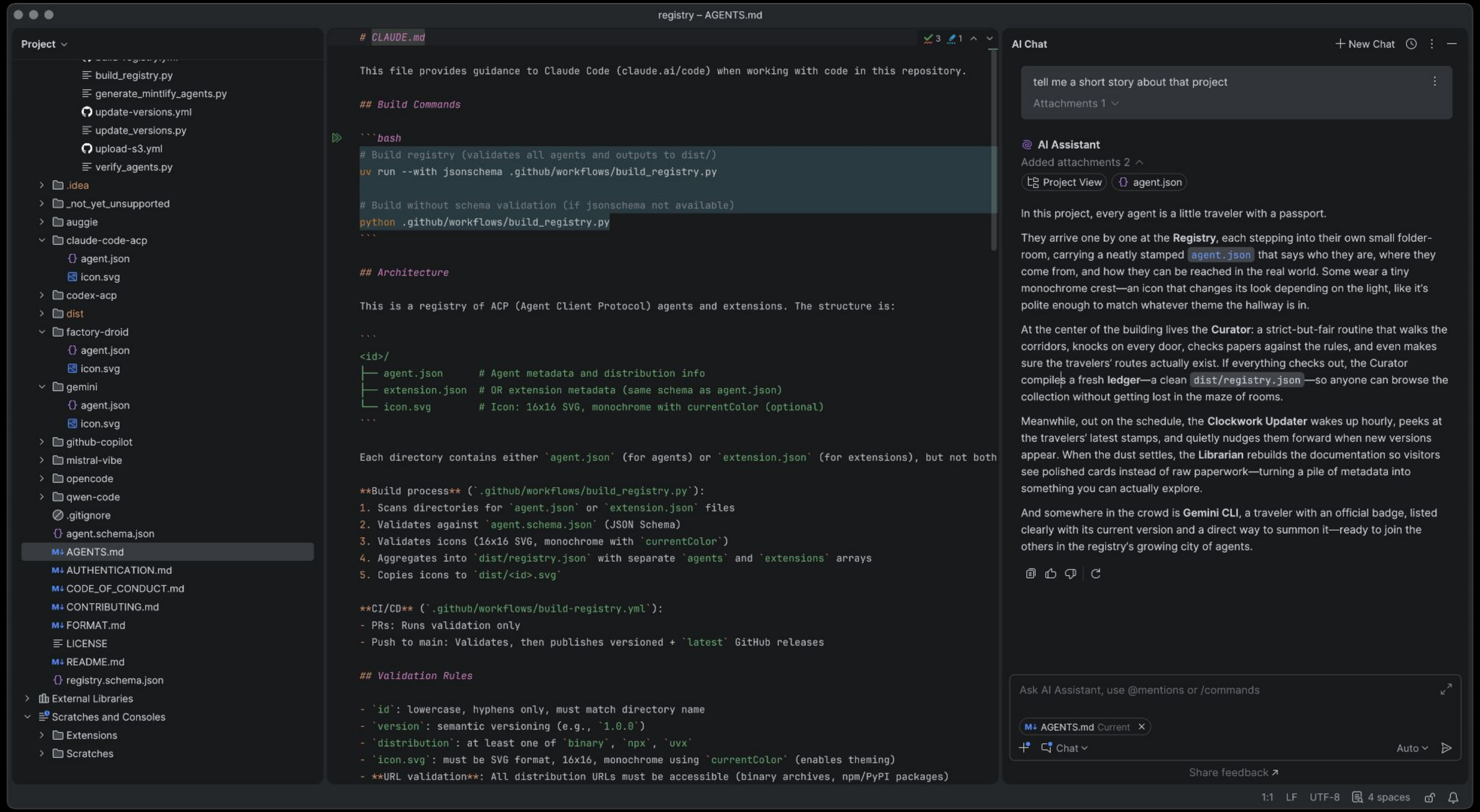
Python Jupyter SQL cells

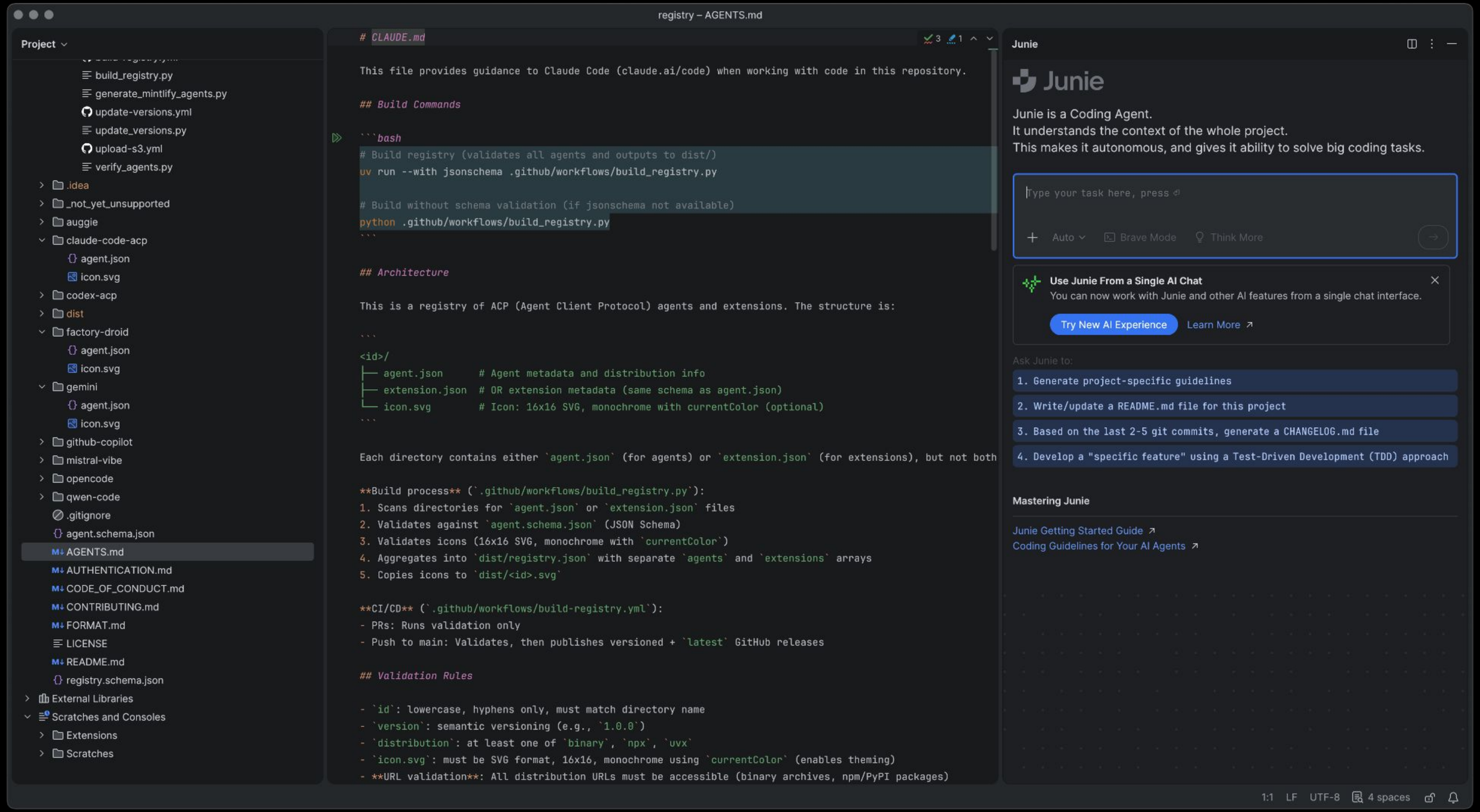


ReSharper

Extension that makes Visual Studio a much better IDE

C# .NET ASP.NET


registry – AGENTS.md
Project
build_registry.py
generate_mintlify_agents.py
update-versions.yml
update_versions.py
upload-s3.yml
verify_agents.py
.idea
_not_yet_unsupported
auggie
claude-code-acp
agent.json
icon.svg
codex-acp
dist
factory-droid
agent.json
icon.svg
gemini
agent.json
icon.svg
github-copilot
mistral-vibe
opencode
qwen-code
.gitignore
agent.schema.json
AGENTS.md
AUTHENTICATION.md
CODE_OF_CONDUCT.md
CONTRIBUTING.md
FORMAT.md
LICENSE
README.md
registry.schema.json
External Libraries
Scratches and Consoles
Extensions
Scratches
CLAUDE.md
This file provides guidance to Claude Code (claude.ai/code) when working with code in this repository.
Build Commands
```bash
# Build registry (validates all agents and outputs to dist/)
uv run --with jsonschema .github/workflows/build_registry.py
# Build without schema validation (if jsonschema not available)
python .github/workflows/build_registry.py
```
Architecture
This is a registry of ACP (Agent Client Protocol) agents and extensions. The structure is:
```
<id>/
├── agent.json      # Agent metadata and distribution info
├── extension.json  # OR extension metadata (same schema as agent.json)
└── icon.svg        # Icon: 16x16 SVG, monochrome with currentColor (optional)
```
Each directory contains either `agent.json` (for agents) or `extension.json` (for extensions), but not both
Build process (`.github/workflows/build_registry.py`):
1. Scans directories for `agent.json` or `extension.json` files
2. Validates against `agent.schema.json` (JSON Schema)
3. Validates icons (16x16 SVG, monochrome with `currentColor`)
4. Aggregates into `dist/registry.json` with separate `agents` and `extensions` arrays
5. Copies icons to `dist/<id>.svg`
CI/CD (`.github/workflows/build-registry.yml`):
- PRs: Runs validation only
- Push to main: Validates, then publishes versioned + `latest` GitHub releases
Validation Rules
- `id`: lowercase, hyphens only, must match directory name
- `version`: semantic versioning (e.g., `1.0.0`)
- `distribution`: at least one of `binary`, `npx`, `uvx`
- `icon.svg`: must be SVG format, 16x16, monochrome using `currentColor` (enables theming)
- **URL validation**: All distribution URLs must be accessible (binary archives, npm/PyPI packages)
AI Chat
New Chat
tell me a short story about that project
Attachments 1
AI Assistant
Added attachments 2
Project View
agent.json
In this project, every agent is a little traveler with a passport.
They arrive one by one at the Registry, each stepping into their own small folder-room, carrying a neatly stamped agent.json that says who they are, where they come from, and how they can be reached in the real world. Some wear a tiny monochrome crest—an icon that changes its look depending on the light, like it's polite enough to match whatever theme the hallway is in.
At the center of the building lives the Curator: a strict-but-fair routine that walks the corridors, knocks on every door, checks papers against the rules, and even makes sure the travelers' routes actually exist. If everything checks out, the Curator compiles a fresh ledger—a clean dist/registry.json—so anyone can browse the collection without getting lost in the maze of rooms.
Meanwhile, out on the schedule, the Clockwork Updater wakes up hourly, peeks at the travelers' latest stamps, and quietly nudges them forward when new versions appear. When the dust settles, the Librarian rebuilds the documentation so visitors see polished cards instead of raw paperwork—turning a pile of metadata into something you can actually explore.
And somewhere in the crowd is Gemini CLI, a traveler with an official badge, listed clearly with its current version and a direct way to summon it—ready to join the others in the registry's growing city of agents.
Ask AI Assistant, use @mentions or /commands
AGENTS.md Current
Chat
Auto
Share feedback
1:1
LF
UTF-8
4 spaces


registry – AGENTS.md
Project
build_registry.py
generate_mintlify_agents.py
update-versions.yml
update_versions.py
upload-s3.yml
verify_agents.py
.idea
_not_yet_unsupported
auggie
claude-code-acp
agent.json
icon.svg
codex-acp
dist
factory-droid
agent.json
icon.svg
gemini
agent.json
icon.svg
github-copilot
mistral-vibe
opencode
qwen-code
.gitignore
agent.schema.json
AGENTS.md
AUTHENTICATION.md
CODE_OF_CONDUCT.md
CONTRIBUTING.md
FORMAT.md
LICENSE
README.md
registry.schema.json
External Libraries
Scratches and Consoles
Extensions
Scratches
CLAUDE.md
This file provides guidance to Claude Code (claude.ai/code) when working with code in this repository.
Build Commands
```bash
# Build registry (validates all agents and outputs to dist/)
uv run --with jsonschema .github/workflows/build_registry.py
# Build without schema validation (if jsonschema not available)
python .github/workflows/build_registry.py
```
Architecture
This is a registry of ACP (Agent Client Protocol) agents and extensions. The structure is:
```
<id>/
├── agent.json      # Agent metadata and distribution info
├── extension.json  # OR extension metadata (same schema as agent.json)
└── icon.svg        # Icon: 16x16 SVG, monochrome with currentColor (optional)
```
Each directory contains either `agent.json` (for agents) or `extension.json` (for extensions), but not both
Build process (`.github/workflows/build_registry.py`):
1. Scans directories for `agent.json` or `extension.json` files
2. Validates against `agent.schema.json` (JSON Schema)
3. Validates icons (16x16 SVG, monochrome with `currentColor`)
4. Aggregates into `dist/registry.json` with separate `agents` and `extensions` arrays
5. Copies icons to `dist/<id>.svg`
CI/CD (`.github/workflows/build-registry.yml`):
- PRs: Runs validation only
- Push to main: Validates, then publishes versioned + `latest` GitHub releases
Validation Rules
- `id`: lowercase, hyphens only, must match directory name
- `version`: semantic versioning (e.g., `1.0.0`)
- `distribution`: at least one of `binary`, `npx`, `uvx`
- `icon.svg`: must be SVG format, 16x16, monochrome using `currentColor` (enables theming)
- **URL validation**: All distribution URLs must be accessible (binary archives, npm/PyPI packages)
Junie
Junie
Junie is a Coding Agent.
It understands the context of the whole project.
This makes it autonomous, and gives it ability to solve big coding tasks.
Type your task here, press
Auto
Brave Mode
Think More
Use Junie From a Single AI Chat
You can now work with Junie and other AI features from a single chat interface.
Try New AI Experience
Learn More
Ask Junie to:
1. Generate project-specific guidelines
2. Write/update a README.md file for this project
3. Based on the last 2-5 git commits, generate a CHANGELOG.md file
4. Develop a "specific feature" using a Test-Driven Development (TDD) approach
Mastering Junie
Junie Getting Started Guide
Coding Guidelines for Your AI Agents
1:1
LF
UTF-8
4 spaces

Gemini - ignatov

Claude Code

Welcome back Sergey!



Opus 4.5
API Usage Billing
/Users/ignatov

❯ Try "refactor <filepath>"

ignatov | Opus 4.5 | 0%

GEMINI

Tips for getting started:
1. Ask questions, edit files, or run commands.
2. Be specific for the best results.
3. /help for more information.

ℹ Gemini CLI update available! 0.22.3 → 0.26.0
Locally installed. Please update via your project's package.json.

You are running Gemini CLI in your home directory. It is recommended to run in a project-specific directory.

Using:
- 1 GEMINI.md file

> Type your message or @path/to/file

~ no sandbox Auto (Gemini 2.5) /model

✨ Update available! 0.77.0 → 0.92.0
Run npm install -g @openai/codex to update.

See full release notes:
https://github.com/openai/codex/releases/latest

>_ OpenAI Codex (v0.77.0)

model: gpt-5.2-codex high /model to change
directory: ~

To get started, describe a task or try one of these commands:

/init - create an AGENTS.md file with instructions for Codex
/status - show current session configuration
/approvals - choose what Codex can do without approval
/model - choose what model and reasoning effort to use
/review - review any changes and find issues

⚠ MCP client for `idea` failed to start: MCP startup failed: handshaking with MCP
server failed: Send message error Transport
[rmcp::transport::worker::WorkerTransport<rmcp::transport::streamable_http_clien
t::StreamableHttpClientWorker<reqwest::async_impl::client::Client>>] error:
Client error: HTTP status client error (404 Not Found) for url
(http://127.0.0.1:64342/stream), when send initialize request

⚠ MCP startup incomplete (failed: idea)

› Implement {feature}

100% context left · ? for shortcuts

We decided to build a protocol between IDEs and agents

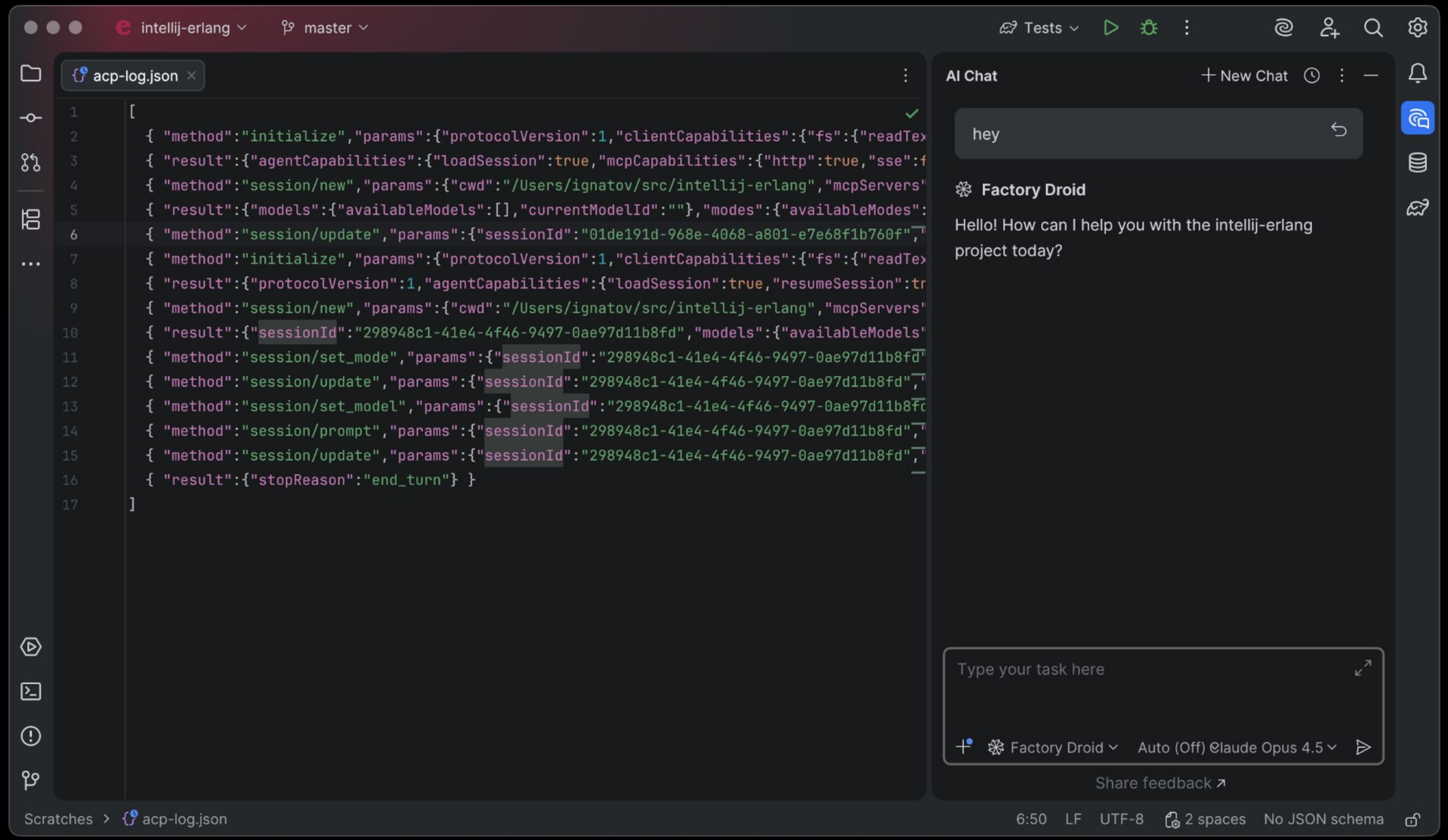
Time matters



Agent/Client Protocol

JSON RPC Spec


```
{ "method":"initialize","params":{"protocolVersion":1,"clientCapabilities":{"fs":{"readTextFile":true
{ "result":{"agentCapabilities":{"loadSession":true,"mcpCapabilities":{"http":true,"sse":false},"pro
{ "method":"session/new","params":{"cwd":"/Users/ignatov/src/intellij-erlang","mcpServers":[]} },
{ "result":{"models":{"availableModels":[],"currentModelId":""},"modes":{"availableModes":[{"descript
{ "method":"session/update","params":{"sessionId":"01de191d-968e-4068-a801-e7e68f1b760f","update":{"a
{ "method":"initialize","params":{"protocolVersion":1,"clientCapabilities":{"fs":{"readTextFile":true
{ "result":{"protocolVersion":1,"agentCapabilities":{"loadSession":true,"resumeSession":true,"session
{ "method":"session/new","params":{"cwd":"/Users/ignatov/src/intellij-erlang","mcpServers":[]} },
{ "result":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","models":{"availableModels":[{"modelId
{ "method":"session/set_mode","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","modeId":"
{ "method":"session/update","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","update":{"s
{ "method":"session/set_model","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","modelId"
{ "method":"session/prompt","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","prompt":[{"
{ "method":"session/update","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","update":{"s
{ "result":{"stopReason":"end_turn"} }
```




```
[
  { "method":"initialize","params":{"protocolVersion":1,"clientCapabilities":{"fs":{"readTex
  { "result":{"agentCapabilities":{"loadSession":true,"mcpCapabilities":{"http":true,"sse":f
  { "method":"session/new","params":{"cwd":"/Users/ignatov/src/intellij-erlang","mcpServers"
  { "result":{"models":{"availableModels":[],"currentModelId":""},"modes":{"availableModes":
  { "method":"session/update","params":{"sessionId":"01de191d-968e-4068-a801-e7e68f1b760f","
  { "method":"initialize","params":{"protocolVersion":1,"clientCapabilities":{"fs":{"readTex
  { "result":{"protocolVersion":1,"agentCapabilities":{"loadSession":true,"resumeSession":tr
  { "method":"session/new","params":{"cwd":"/Users/ignatov/src/intellij-erlang","mcpServers"
  { "result":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","models":{"availableModels"
  { "method":"session/set_mode","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd"
  { "method":"session/update","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","
  { "method":"session/set_model","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd
  { "method":"session/prompt","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","
  { "method":"session/update","params":{"sessionId":"298948c1-41e4-4f46-9497-0ae97d11b8fd","
  { "result":{"stopReason":"end_turn"} }
]
```



Gemini CLI

Google's reference implementation showing the full potential of ACP agent integration.

Via Adapter



Claude Code

Anthropic's Claude integrated through Zed's SDK adapter, available to all ACP clients.

Via Adapter



Codex

OpenAI's code generation agent with streaming terminal output and community implementations.

In Progress



JetBrains Junie

JetBrains' AI agent coming soon with ACP support across their entire IDE ecosystem.



Goose

Square's open-source agent with native ACP implementation, available to any ACP client.

KIMI CLI

Kimi CLI

A CLI AI agent implementing the Agent Client Protocol with support for various development workflows.



OpenCode

An open-source coding agent with ACP implementation, designed for flexible integration with various editors and IDEs.



Stakpak

An ACP-compatible agent focused on providing comprehensive code assistance and collaboration features.

VT CODE

VT Code

A versatile coding agent implementing ACP for seamless integration with compatible development environments.



Augment Code

Auggie gives you powerful agentic capabilities to analyze code, make changes, and execute tools.

CODE ASSISTANT

Code Assistant

An AI coding assistant built in Rust for autonomous code analysis and modification.



cagent

A powerful, easy-to-use, customizable multi-agent runtime that orchestrates AI agents.

FAST-AGENT

fast-agent

Create and interact with sophisticated Agents and Workflows in minutes.



LLMling-Agent

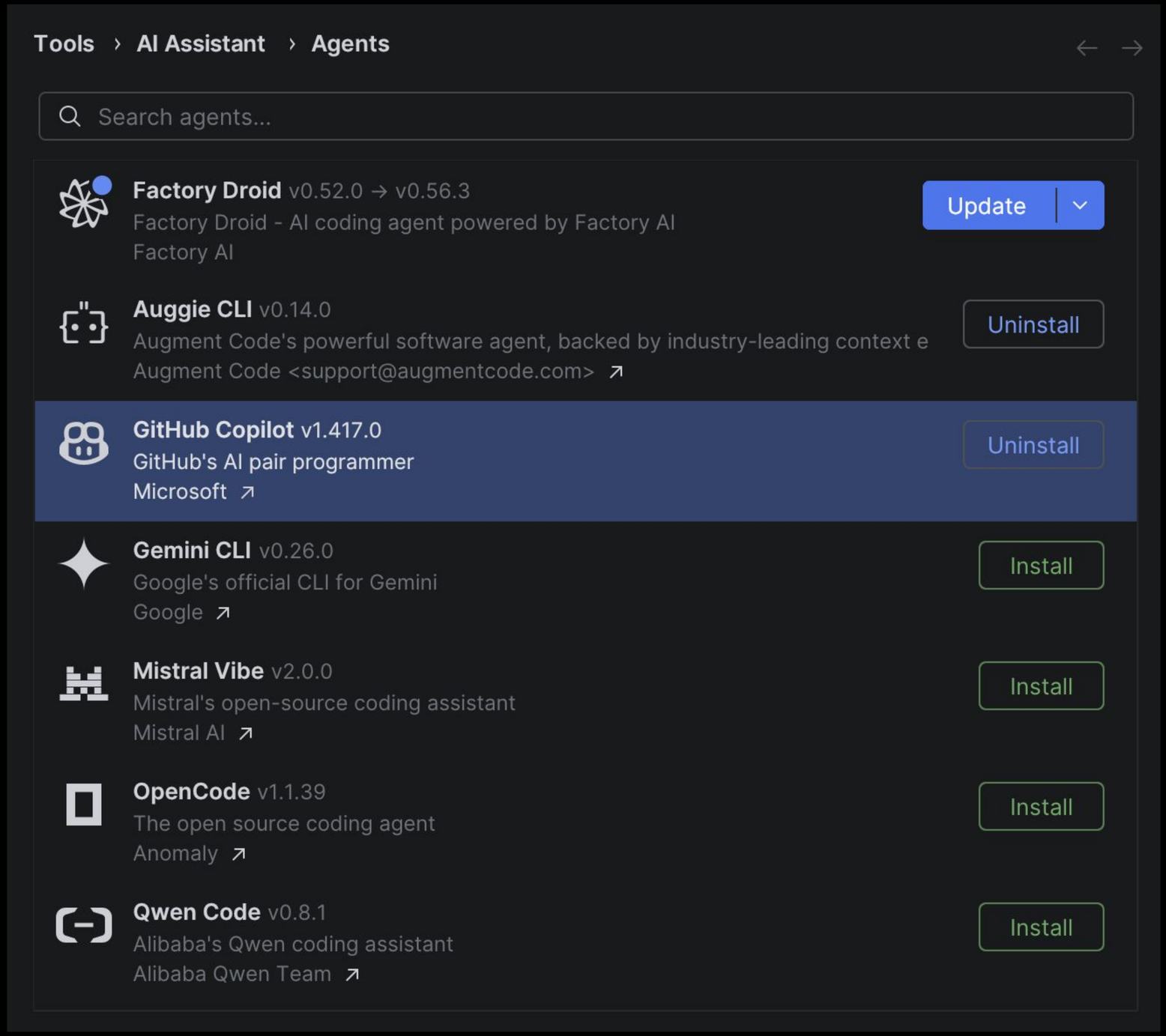
A framework for creating and managing LLM-powered agents to provide structured interactions.

IMPROVEMENT

JANUARY 28, 2026 • 1 MINUTE READ

ACP support in Copilot CLI is now in public preview




Tools › AI Assistant › Agents
Search agents...
Factory Droid v0.52.0 → v0.56.3
Factory Droid - AI coding agent powered by Factory AI
Factory AI
Update
Auggie CLI v0.14.0
Augment Code's powerful software agent, backed by industry-leading context e
Augment Code <support@augmentcode.com>
Uninstall
GitHub Copilot v1.417.0
GitHub's AI pair programmer
Microsoft
Uninstall
Gemini CLI v0.26.0
Google's official CLI for Gemini
Google
Install
Mistral Vibe v2.0.0
Mistral's open-source coding assistant
Mistral AI
Install
OpenCode v1.1.39
The open source coding agent
Anomaly
Install
Qwen Code v0.8.1
Alibaba's Qwen coding assistant
Alibaba Qwen Team
Install

Zed

Native ACP support with real-time editing, syntax highlighting, and agent following.

JetBrains

Full ACP integration coming soon to IntelliJ IDEA, PyCharm, and other JetBrains IDEs.

In Progress

Sidequery

Browser-based development environment with ACP support coming soon.

AionUi

Modern editor with integrated ACP support for seamless agent interaction.

Emacs

Agent-shell.el plugin brings ACP agents to Emacs through a native shell interface.

marimo

Python notebook environment with integrated ACP agent support for data science workflows.

Via Plugin

Neovim

Through two available plugins: CodeCompanion & yetone/avante.nvim.

Via Plugin

Obsidian

Bring your AI agents directly into Obsidian. Chat with Claude Code, Codex, and others from your vault.

aizen

Manage multiple Git branches simultaneously with dedicated terminals and agents in parallel.

DeepChat

A feature-rich open-source AI chat platform supporting multiple cloud and local LLMs.

Tidewave

The coding agent for full-stack web app development. Integrate agents with your web app at every layer.

Via AI SDK Adapter

Web Browser

AI SDK-powered browser client with ACP support via @mcpc/acp-ai-provider.

00:40

Search

Agmente

Agent Client Protocol client

Open

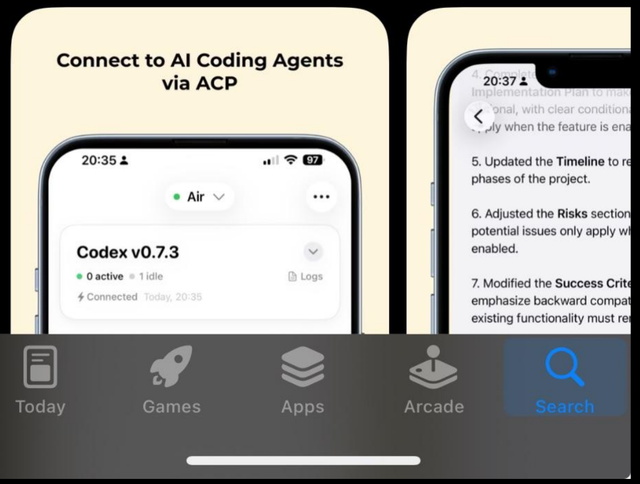
AGE	CATEGORY	DEVELOPER	L/
4+	Developer Tools	Peng Lyu	
Years Old			

What's New

Version 1.2 3w ago

- Faster and more responsive session switching
- Better stability when managing multiple agent connections more

Preview



00:34

Qwen

Search sessions

Qwen Code v0.8.1

0 active 2 idle Logs

Connected Today, 00:34

TODAY

hey
The project appears to be an agent-cli...

hey
Hello! How can I assist you with the A...

+ New Session

00:34

Desc the project

The project appears to be an agent-client-protocol implementation, which is a Rust/Cargo-based project with a focus on defining and managing communication protocols between agents and clients. It includes a `Cargo.toml` file, suggesting it's a Rust crate, and a `package.json` file, indicating it might also have JavaScript/TypeScript components. The presence of `.release-plz.toml` suggests automated release management. The project also includes documentation files like `AGENTS.md`, `CHANGELOG.md`, `CONTRIBUTING.md`, and `README.md`, which provide insights into the project's purpose, contribution guidelines, and changelog. The directory structure indicates a focus on protocol definition and implementation, possibly for AI agents or similar systems.

Default

Message the agent...

github.com

agentclientprotocol

Type / to search

Overview | Repositories 11 | Discussions | Projects | Packages | Teams 7 | People 15 | Insights | Settings

Agent Client Protocol (ACP)

A protocol for connecting any editor to any agent

Verified

127 followers | https://agentclientprotocol.com/

Follow

README.md



Agent Client Protocol

The Agent Client Protocol (ACP) standardizes communication between *code editors* (interactive programs for viewing and editing source code) and *coding agents* (programs that use generative AI to autonomously modify code).

Learn more at agentclientprotocol.com.

Integrations

- Schema
- Agents
- Clients
- Official Libraries
  - **Kotlin**: `acp-kotlin` - Supports JVM, other targets are in progress
  - **Python**: `python-sdk` - See examples
  - **Rust**: `agent-client-protocol` - See examples/agent.rs and examples/client.rs
  - **TypeScript**: `@agentclientprotocol/sdk` - See examples/
- Community Libraries

Pinned

Customize pins

agent-client-protocol Public
A protocol for connecting any editor to any agent
Rust 1.8k 138

kotlin-sdk Public
Agent Client Protocol Kotlin SDK
Kotlin 38 9

python-sdk Public
Python SDK for ACP clients and agents.
Python 135 12

rust-sdk Public
Rust SDK for ACP clients and agents.
Rust 46 10

View as: Public

You are viewing the README and pinned repositories as a public user.

Get started with tasks that most successful organizations complete.

Top discussions this past month

- **Support for questions and "wizards" like AskUserQuestionTool** ↑ 4 💬 4
- **Remote ACP Agents** ↑ 2 💬 0
- **How to "always approve" tool calls for specific MCP server** ↑ 1 💬 0

View all discussions

People

Invite someone

Top languages

Rust MDX Java Kotlin TypeScript

Most used topics Manage

acp ai agent-client-protocol agentclientprotocol agents

Requests for Dialog (RFDs)

Copy page

Our process for introducing changes to the protocol

A “Request for Dialog” (RFD) is ACP’s version of the RFC process. RFDs are the primary mechanism for proposing new features, collecting community input on an issue, and documenting design decisions.

When to write an RFD

You should consider writing an RFD if you intend to make a “substantial” change to ACP or its documentation. What constitutes a “substantial” change is evolving based on community norms and varies depending on what part of the ecosystem you are proposing to change.

Some changes do not require an RFD:

- Rephrasing, reorganizing or refactoring
- Addition or removal of warnings
- Additions that strictly improve objective, numerical quality criteria (speedup, better browser support)
- Fixing objectively incorrect behavior

The RFD Process

1. Propose by opening a PR

Fork the repo and copy `docs/rfds/TEMPLATE.md` to `docs/rfds/my-feature.md` (using kebab-case naming). The RFD can start minimal -just an elevator pitch and status quo are enough to begin dialog. Pull requests

Use any agent you like in the editor you love ❤