AgentZero

1. **General-purpose assistant**

* Agent Zero is not pre-programmed for specific tasks (but can be). It is meant to be a general-purpose personal assistant. Give it a task, and it will gather information, execute commands and code, cooperate with other agent instances, and do its best to accomplish it.
* It has a persistent memory, allowing it to memorize previous solutions, code, facts, instructions. to solve tasks faster and more reliably in the future.

1. **Computer as a tool**

* Agent Zero uses the operating system as a tool to accomplish its tasks. It has no single-purpose tools pre-programmed. Instead, it can write its own code and use the terminal to create and use its own tools as needed.
* The only default tools in its arsenal are online search, memory features, communication (with the user and other agents), and code/terminal execution. Everything else is created by the agent itself or can be extended by the user.
* Tool usage functionality has been developed from scratch to be the most compatible and reliable, even with very small models.
* **Default Tools:** Agent Zero includes tools like knowledge, webpage content, code execution, and communication.
* **Creating Custom Tools:** Extend Agent Zero's functionality by creating your own custom tools.
* **Instruments:** Instruments are a new type of tool that allow you to create custom functions and procedures that can be called by Agent Zero.

1. **Multi-agent cooperation**

* Every agent has a superior agent giving it tasks and instructions. Every agent then reports back to its superior.
* In the case of the first agent in the chain (Agent 0), the superior is the human user; the agent sees no difference.
* Every agent can create its subordinate agent to help break down and solve subtasks. This helps all agents keep their context clean and focused.

1. **Completely customizable and extensible**

* Almost nothing in this framework is hard-coded. Nothing is hidden. Everything can be extended or changed by the user.
* The whole behavior is defined by a system prompt in the **prompts/default/agent.system.md** file. Change this prompt and change the framework dramatically.
* The framework does not guide or limit the agent in any way. There are no hard-coded rails that agents have to follow.
* Every prompt, every small message template sent to the agent in its communication loop, can be found in the **prompts/** folder and changed.
* Every default tool can be found in the **python/tools/** folder and changed or copied to create new predefined tools.
* Of course, it is open-source (except for some tools like Perplexity, but that will be replaced with an open-source alternative as well in the future).

1. **Communication is key**

* Give your agent a proper system prompt and instructions, and it can do miracles.
* Agents can communicate with their superiors and subordinates, asking questions, giving instructions, and providing guidance. Instruct your agents in the system prompt on how to communicate effectively.
* The terminal interface is real-time streamed and interactive. You can stop and intervene at any point. If you see your agent heading in the wrong direction, just stop and tell it right away.
* There is a lot of freedom in this framework. You can instruct your agents to regularly report back to superiors asking for permission to continue. You can instruct them to use point-scoring systems when deciding when to delegate subtasks. Superiors can double-check subordinates' results and dispute. The possibilities are endless.

**Nice features to have**

* Output is very clean, colorful, readable and interactive; nothing is hidden.
* The same colorful output you see in the terminal is automatically saved to HTML file in **logs/** folder for every session.
* Agent output is streamed in real-time, allowing the user to read along and intervene at any time.
* No coding is required, only prompting and communication skills.
* With a solid system prompt, the framework is reliable even with small models, including precise tool usage.