How the Agent Class Works Behind the Scenes (OpenAI Agents SDK)

- Agent Definition:

  An Agent is an AI system configured with a name, instructions (system prompt), a language model (default GPT-4o), and optionally tools, handoffs, guardrails, and context data126.

- Core Components:
  - Name: Identifier for logging and debugging.
  - Instructions: Define the agent's behavior and task (like a system prompt).
  - Model: The underlying LLM powering the agent (e.g., GPT-4o). Tools: Python
  - functions or APIs the agent can call to extend capabilities.
  - Handoffs: Mechanism to delegate tasks to other agents for multiagent workflows.
  - Guardrails: Input validation and safety checks to ensure valid and safe interactions.
  - Context: Custom Python objects or data passed to the agent to maintain state or provide additional info126.

- Initialization:

  When an Agent instance is created, it is configured with the above components, especially instructions that guide how the LLM should respond16.

- Input Processing:

  The agent receives user input and combines it with its instructions to create a prompt for the LLM. This prompt guides the model's response generation16.

- Agent Loop:

  The agent runs an internal loop where it:

  1. Sends the current prompt to the LLM.
  2. Checks if the LLM's response requires calling any tool.
  3. Calls the tool if needed and feeds the tool's output back to the LLM.
  4. Repeats until the LLM produces a final output without needing further tool calls26.

- Tools Integration:

  Tools are Python functions or APIs registered with the agent. When the LLM's response indicates a tool call, the agent executes the tool and returns the result to the LLM to refine the answer16.

- Handoffs:

  In multi-agent setups, the agent can hand off control to another specialized agent to handle specific subtasks. This can be done either by orchestrator-subagent pattern or by full control handoff, allowing subagents to respond directly to users[2][5][6].

- Guardrails:
  Guardrails act as safety layers that validate inputs before processing and outputs before returning to users, preventing malicious or invalid data from affecting the system[2][6].
- Tracing and Monitoring:
  The SDK supports tracing agent activity, tool usage, and outputs via dashboards, helping developers debug and optimize agent workflows[6].
- Customization:
  Agents can be finely tuned via model settings (temperature, max tokens, etc.), custom tools, handoffs, and context objects to build complex and reliable AI systems