ADK Tools Overview

The Agent Development Kit (ADK) provides a rich and extensible ecosystem of tools that empower AI agents with diverse capabilities. Tools allow agents to interact with external systems, APIs, data sources, and perform complex computations.

Core Concepts

`BaseTool`

The fundamental class for all tools in ADK. Developers can subclass `BaseTool` to create custom tools with specific functionalities.

- **Module:** `google.adk.tools.base_tool`
- **Key Attributes:** `name`, `description`, `is_long_running`.
- **Key Methods:** `_get_declaration()` (to define how the tool appears to the LLM)

General Purpose Tools

Google Search (`google_search`)

A built-in tool that allows agents to perform Google searches. This tool is automatically invoked by Gemini models when web search is needed.

- **Module:** `google.adk.tools.google_search_tool`
- **Class:** `GoogleSearchTool`
- **Note:** For Gemini 1.x, it uses `GoogleSearchRetrieval`; for Gemini 2.x, it uses `GoogleSearch`.

Load Web Page (`load_web_page`)

Fetches the content of a given URL and returns the extracted text. Useful for agents that need to process information from web pages.

- **Module:** `google.adk.tools.load_web_page`

Flow Control Tools

Exit Loop (`exit_loop`)

A tool used within a `LoopAgent` to signal that the loop should terminate.

- **Module:** `google.adk.tools.exit_loop_tool`
- **Usage:** `exit_loop(tool_context)` (sets `tool_context.actions.escalate = True`)

Transfer to Agent (`transfer_to_agent`)

Allows an agent to hand off the current task or query to another specified agent within the agent hierarchy.

- **Module:** `google.adk.tools.transfer_to_agent_tool`
- **Usage:** `transfer_to_agent(agent_name="specialist_agent_name", tool_context)`

Memory and Artifact Tools

Load Artifacts (`load_artifacts`)

Loads specified artifacts that are attached to the current session, making their content available to the agent.

- **Module:** `google.adk.tools.load_artifacts_tool`
- **Class:** `LoadArtifactsTool`
- **Usage:** Model calls `load_artifacts(artifact_names=["file1.txt", "image.png"])`. The tool then makes these available in subsequent LLM requests.

Load Memory (`load_memory`)

Queries the agent's memory (associated with the current user) for relevant past interactions or information.

- **Module:** `google.adk.tools.load_memory_tool`

API Integration Tools & Toolsets

OpenAPI

ADK provides robust support for integrating with APIs defined by OpenAPI specifications.

- **`OpenAPIToolset`**:
  - **Module:** `google.adk.tools.openapi_tool.openapi_spec_parser`
  - **Purpose:** Parses an OpenAPI specification (JSON or YAML) and generates a collection of `RestApiTool` instances, one for each operation defined in the spec.
  - **Usage:** `toolset = OpenAPIToolset(spec_str=openapi_string, spec_str_type="json")`
- **`RestApiTool`**:
  - **Module:** `google.adk.tools.openapi_tool.openapi_spec_parser`
  - **Purpose:** Represents a single REST API operation. It handles request parameter

Enterprise & Search Tools

Vertex AI Search Tool (`VertexAiSearchTool`)

A built-in tool that allows agents to use Vertex AI Search for information retrieval from specified data stores or search engines.

- **Module:** `google.adk.tools.vertex_ai_search_tool`
- **Class:** `VertexAiSearchTool`
- **Usage:** `search_tool = VertexAiSearchTool(data_store_id="projects/.../dataStores/...")`
- **Note:** For Gemini 2.x, this tool configures the LLM request to use the built-in Vertex AI RAG capabilities. For other models, it might execute a search query directly.

Enterprise Web Search Tool (`EnterpriseWebSearchTool`)

A Gemini 2+ built-in tool that uses web grounding with enterprise compliance.

Third-party Framework Integration

Langchain Tool (`LangchainTool`)

An adapter to wrap tools from the Langchain framework, making them compatible with ADK.

- **Module:** `google.adk.tools.langchain_tool`
- **Class:** `LangchainTool`
- **Usage:** `wrapped_lc_tool = LangchainTool(langchain_search_tool)`

CrewAI Tool (`CrewaiTool`)

An adapter to wrap tools from the CrewAI framework.

- **Module:** `google.adk.tools.crewai_tool`
- **Class:** `CrewaiTool`

Specialized Tools

Long-Running Function Tool (`LongRunningFunctionTool`)

A specialized `FunctionTool` for Python functions that are long-running or asynchronous. The framework calls the function, and the result is returned asynchronously.

- **Module:** `google.adk.tools.long_running_tool`
- **Class:** `LongRunningFunctionTool`
- **Usage:** `async_tool = LongRunningFunctionTool(my_long_python_function)`

Code Execution

While not strictly a "tool" in the `google.adk.tools` sense that an LLM calls, ADK supports code execution capabilities for agents:

- **`BuiltInCodeExecutor`**: (Module: `google.adk.code_executors`)
  This executor leverages the model's (e.g., Gemini 2.0+) built-in code execution capabilities.
- **Other Executors**: ADK also provides `UnsafeLocalCodeExecutor`, `VertexAiCodeExecutor`, and `ContainerCodeExecutor` for different code execution environments.
- **Deprecation Note:** The older `BuiltInCodeExecutionTool` (from `google.adk.tools`) has been deprecated and replaced by `BuiltInCodeExecutor`.

Creating Custom Tools

You can create custom tools in ADK in two main ways:

1. **Using `FunctionTool`**: The easiest way for simple Python functions.

```
from google.adk.tools import FunctionTool

def my_custom_function(param1: str, param2: int) -> str:
    """This is my custom function that does something."""
    return f"Processed <{param1}> with <{param2}>"

custom_tool = FunctionTool(my_custom_function)
```

2. **Subclassing `BaseTool`**: For more complex tools requiring custom logic for declaration or execution.

```
from google.adk.tools import BaseTool, ToolContext
from google.genai import types

class MyAdvancedTool(BaseTool):
    def __init__(self):
```

This overview provides a glimpse into the diverse range of tools available within the ADK. By leveraging these tools, developers can build sophisticated and capable AI agents.