Custom agent

This notebook goes through how to create your own custom agent.

An agent consists of two parts:

```
- Tools: The tools the agent has available to use.
- The agent class itself: this decides which action to take.
```

In this notebook we walk through how to create a custom agent.

```
from langchain.agents import Tool, AgentExecutor, BaseSingleActionAgent
from langchain import OpenAI, SerpAPIWrapper
```

API Reference:

- Tool from `langchain.agents`
- AgentExecutor from `langchain.agents`
- BaseSingleActionAgent from `langchain.agents`

```
search = SerpAPIWrapper()
tools = [
    Tool(
        name="Search",
        func=search.run,
        description="useful for when you need to answer questions about
current events",
        return_direct=True,
    )
]
```

```
from typing import List, Tuple, Any, Union
from langchain.schema import AgentAction, AgentFinish


class FakeAgent(BaseSingleActionAgent):
```

```
    """Fake Custom Agent."""

    @property
    def input_keys(self):
        return ["input"]

    def plan(
        self, intermediate_steps: List[Tuple[AgentAction, str]],
**kwargs: Any
    ) -> Union[AgentAction, AgentFinish]:
        """Given input, decided what to do.

        Args:
            intermediate_steps: Steps the LLM has taken to date,
                along with observations
            **kwargs: User inputs.

        Returns:
            Action specifying what tool to use.
        """
        return AgentAction(tool="Search", tool_input=kwargs["input"],
log="")

    async def aplan(
        self, intermediate_steps: List[Tuple[AgentAction, str]],
**kwargs: Any
    ) -> Union[AgentAction, AgentFinish]:
        """Given input, decided what to do.

        Args:
            intermediate_steps: Steps the LLM has taken to date,
                along with observations
            **kwargs: User inputs.

        Returns:
            Action specifying what tool to use.
        """
        return AgentAction(tool="Search", tool_input=kwargs["input"],
log="")
```

API Reference:

- AgentAction from `langchain.schema`
- AgentFinish from `langchain.schema`

```
agent = FakeAgent()
```

```
agent_executor = AgentExecutor.from_agent_and_tools(
    agent=agent, tools=tools, verbose=True
)
```

```
agent_executor.run("How many people live in canada as of 2023?")
```

```
    > Entering new AgentExecutor chain...
    The current population of Canada is 38,669,152 as of Monday, April 24, 2023, based on Worldometer elaboration of the latest United Nations data.

    > Finished chain.
```

```
    'The current population of Canada is 38,669,152 as of Monday, April 24, 2023, based on Worldometer elaboration of the latest United Nations data.'
```