LangChain Expression Language   Cookbook

Adding memory

Adding memory

This shows how to add memory to an arbitrary chain. Right now, you can use the memory classes but need to hook it up manually

```
%pip install --upgrade --quiet  langchain
langchain-openai
```

```
from operator import itemgetter

from langchain.memory import
ConversationBufferMemory
from langchain_core.prompts import
ChatPromptTemplate, MessagesPlaceholder
from langchain_core.runnables import
RunnableLambda, RunnablePassthrough
from langchain_openai import ChatOpenAI

model = ChatOpenAI()
prompt = ChatPromptTemplate.from_messages(
    [
        ("system", "You are a helpful
chatbot"),

MessagesPlaceholder(variable_name="history"),
        ("human", "{input}"),
    ]
)
```

```
memory =
ConversationBufferMemory(return_messages=True)
```

```
memory.load_memory_variables({})
```

```
{'history': []}
```

```
chain = (
    RunnablePassthrough.assign(

history=RunnableLambda(memory.load_memory_varia
| itemgetter("history")
    )
    | prompt
    | model
)
```

```
inputs = {"input": "hi im bob"}
response = chain.invoke(inputs)
response
```

```
AIMessage(content='Hello Bob! How can I
assist you today?', additional_kwargs={},
example=False)
```

```
memory.save_context(inputs, {"output":
response.content})
```

```
memory.load_memory_variables({})
```

```
{'history': [HumanMessage(content='hi im
bob', additional_kwargs={}, example=False),
  AIMessage(content='Hello Bob! How can I
assist you today?', additional_kwargs={},
example=False)]}
```

```
inputs = {"input": "whats my name"}
response = chain.invoke(inputs)
response
```

```
AIMessage(content='Your name is Bob.',
additional_kwargs={}, example=False)
```