Chat Prompt Template

Chat Models takes a list of chat messages as input - this list commonly referred to as a prompt. Typically this is not simply a hardcoded list of messages but rather a combination of a template, some examples, and user input. LangChain provides several classes and functions to make constructing and working with prompts easy.

```
from langchain.prompts import (
    ChatPromptTemplate,
    PromptTemplate,
    SystemMessagePromptTemplate,
    AIMessagePromptTemplate,
    HumanMessagePromptTemplate,
)
from langchain.schema import (
    AIMessage,
    HumanMessage,
    SystemMessage
)
```

You can make use of templating by using a `MessagePromptTemplate`. You can build a `ChatPromptTemplate` from one or more `MessagePromptTemplates`. You can use `ChatPromptTemplate`'s `format_prompt` – this returns a `PromptValue`, which you can convert to a string or Message object, depending on whether you want to use the formatted value as input to an llm or chat model.

For convenience, there is a `from_template` method exposed on the template. If you were to use this template, this is what it would look like:

```
template="You are a helpful assistant that translates {input_language} to
{output_language}."
system_message_prompt = SystemMessagePromptTemplate.from_template(template)
human_template="{text}"
human_message_prompt = HumanMessagePromptTemplate.from_template(human_template)
```

```
chat_prompt = ChatPromptTemplate.from_messages([system_message_prompt,
human_message_prompt])
```

Skip to main content

```
chat_prompt.format_prompt(input_language="English", output_language="French",
text="I love programming.").to_messages()
```

```
[SystemMessage(content='You are a helpful assistant that translates English to
French.', additional_kwargs={}),
 HumanMessage(content='I love programming.', additional_kwargs={})]
```

If you wanted to construct the MessagePromptTemplate more directly, you could create a PromptTemplate outside and then pass it in, eg:

```
prompt=PromptTemplate(
    template="You are a helpful assistant that translates {input_language} to
{output_language}.",
    input_variables=["input_language", "output_language"],
)
system_message_prompt = SystemMessagePromptTemplate(prompt=prompt)
```