LLMObjects: Simplifying Platform and Model Integration

When switching between different platforms and their respective LLM models, minor code changes are often necessary. Although these changes are small, they can still cause disruptions. `LLMObjects` provides a layer on top of LangChain that offers a unified interface for setting up LangChain platform objects with the desired model. This abstraction simplifies the process of integrating and switching between various LLM platforms and models.

Key Features of LLMObjects

- **Unified Interface**: A single, consistent interface for interacting with various LLM platforms and models.
- **Simplified Configuration**: Reduces the need for platform-specific code changes.
- **Extended Parameter Support**: Easily configure common and advanced parameters across different models.

Supported Parameters

The following table summarizes the parameters supported by `LLMObjects` for different LLM platforms:

Parameter	Description	Default Value	Platforms Supported
`model`	The specific model identifier for the platform.	Required	All
`temperature`	Controls the randomness of the model's responses.	0.7	All
`streaming`	Enables streaming of responses if supported by the platform.	False	All
`max_tokens`	Limits the maximum number of tokens generated in the response.	256	OpenAI, Anthropic, Groq, DeepInfra, Cohere, Ollama, NVIDIA
`max_output_tokens`	Similar to `max_tokens`, used specifically for platforms that use this parameter name.	256	GoogleGenerativeAI
`repetition_penalty`	Penalizes repetition to reduce the likelihood of generating repetitive text.	N/A	DeepInfra
`max_new_tokens`	Limits the number of new tokens generated in the response.	N/A	DeepInfra
`top_p`	Controls nucleus sampling by considering the cumulative probability distribution over all tokens.	N/A	DeepInfra, NVIDIA
`api_key`	The API key for authentication with the platform.	Required	All (except Ollama if not required)

base_url	The base URL of the NIM to connect to.	N/A	NVIDIA
seed	A seed for deterministic results.	N/A	NVIDIA
stop	A list of cased stop words.	N/A	NVIDIA

Defining the `model` Parameter

The `model` parameter in `LLMObjects` specifies the identifier of the model to be used within a given platform. This parameter follows a specific format:

```
<Platform>:<ModelSpecifier>
```

- **Platform**: Indicates the platform or provider of the language model, such as `ChatOpenAI`, `ChatAnthropic`, `ChatGroq`, `ChatDeepInfra`, `ChatCohere`, `ChatGoogleGenerativeAI`, `ChatNVIDIA`, or `ChatOllama`.
- **ModelSpecifier**: Specifies the particular model offered by the platform, such as `gpt-4o`, `claude-1.3`, `meta-llama/Meta-Llama-3-70B-Instruct`.

Example Values

- `ChatOpenAI:gpt-4o`
- `ChatAnthropic:claude-1.3`
- `ChatDeepInfra:meta-llama/Meta-Llama-3-70B-Instruct`
- `ChatNVIDIA:llama2_13b`

Sample Usage

Here is a sample code to illustrate how `LLMObjects` simplifies the process of integrating and using different LLM platforms and models. It includes obtaining the platform and model via environment variables:

Environment Usage

```python
import os
from llmobjects import LLMObject
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.output_parsers import StrOutputParser

# Note: This should be done in a script
variable_name = "LLMMODEL"
variable_value = "ChatAnthropic:claude-1.3"
os.environ[variable_name] = variable_value

# Python code
llm = os.getenv(variable_name)
print("Platform:", llm)

# Define a prompt
prompt = ChatPromptTemplate.from_template("tell me a short joke about {topic}")
output_parser = StrOutputParser()

# Initialize LLMObject with the environment variable
```

```
llm_instance = LLMObject(llm, temperature=1, max_tokens=4096).llm_instance
chain2 = prompt | llm_instance | output_parser

print(chain2.invoke({"topic": "ice cream"}))
```

Sample Output

```
Platform: ChatAnthropic:claude-1.3
Here's a short ice cream joke for you:

Why did the ice cream cone blush?

Because it saw the banana split!
```

Full Example with Multiple Models

```
import getpass
import os
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.output_parsers import StrOutputParser
from llmobjects import LLMObject

prompt = ChatPromptTemplate.from_template("tell me a short joke about {topic}")
output_parser = StrOutputParser()

llmlist = [
    "ChatOpenAI:gpt-4o",
    "ChatAnthropic:claude-1.3",
    "ChatDeepInfra:meta-llama/Meta-Llama-3-70B-Instruct",
    "ChatDeepInfra:mistralai/Mixtral-8x22B-Instruct-v0.1",
    "ChatDeepInfra:google/gemma-1.1-7b-it",
    "ChatDeepInfra:01-ai/Yi-34B-Chat",
    "ChatOllama:llama3:latest",
    "ChatGroq:mixtral-8x7b-32768",
    "ChatNVIDIA:llama2_13b",
]

for llm in llmlist:
    print()
    print("Platform:", llm)
    llm_instance = LLMObject(llm, temperature=1, max_tokens=4096).llm_instance
    chain2 = prompt | llm_instance | output_parser
    print(chain2.invoke({"topic": "ice cream"}))
    print()
    print("+" * 40)
```

Sample Output

```
Platform: ChatOpenAI:gpt-4o
Why did the ice cream truck break down? Because of the rocky road.

++++++++++++++++++++++++++++++++++++++++
```

```
Platform: ChatAnthropic:claude-1.3
Why did the ice cream go to the party? Because it was cool and sweet!

++++++++++++++++++++++++++++++++++++++++
Platform: ChatDeepInfra:meta-llama/Meta-Llama-3-70B-Instruct
Why did the ice cream cone become a reporter? It always wanted the scoop.

++++++++++++++++++++++++++++++++++++++++
Platform: ChatDeepInfra:mistralai/Mixtral-8x22B-Instruct-v0.1
Why did the ice cream bring a suitcase? It wanted to go on a Sundae.

++++++++++++++++++++++++++++++++++++++++
Platform: ChatDeepInfra:google/gemma-1.1-7b-it
Why did the ice cream cone join the circus? To show off its sprinkles.

++++++++++++++++++++++++++++++++++++++++
Platform: ChatDeepInfra:01-ai/Yi-34B-Chat
Why did the ice cream bring an umbrella? Because it wanted to drizzle.

++++++++++++++++++++++++++++++++++++++++
Platform: ChatOllama:llama3:latest
Why don't ice cream trucks ever get lost? They always follow the Rocky Road.

++++++++++++++++++++++++++++++++++++++++
Platform: ChatGroq:mixtral-8x7b-32768
Why did the ice cream cone become a musician? Because it had the coolest jams.

++++++++++++++++++++++++++++++++++++++++
Platform: ChatNVIDIA:llama2_13b
Why did the ice cream cone blush? Because it saw the banana split.

++++++++++++++++++++++++++++++++++++++++
```

By abstracting the complexities of platform-specific configurations, `LLMObjects` allows you to focus on developing and deploying applications without worrying about the underlying differences between various LLM platforms.
