

Spring AI Alibaba 实现了与阿里云通义模型的完整适配，接下来，我们将学习如何使用 spring ai alibaba 开发一个基于通义模型服务的智能聊天应用。

快速体验示例

注意：因为 Spring AI Alibaba 基于 Spring Boot 3.x 开发，因此本地 JDK 版本要求为 17 及以上。

1. 下载项目

运行以下命令下载源码，进入 helloworld 示例目录：

```
git clone --depth=1 https://github.com/springaialibaba/spring-ai-alibaba-examples.git
cd spring-ai-alibaba-examples/spring-ai-alibaba-helloworld
```

2. 运行项目

首先，需要获取一个合法的 API-KEY 并设置 `AI_DASHSCOPE_API_KEY` 环境变量，可跳转 [阿里云百炼平台](#) 了解如何获取 API-KEY。

```
export AI_DASHSCOPE_API_KEY=${REPLACE-WITH-VALID-API-KEY}
```

启动示例应用：

```
./mvnw compile exec:java -
Dexec.mainClass="com.alibaba.cloud.ai.example.helloworld.HelloworldApplication"
```

访问 `http://localhost:18080/helloworld/simple/chat?query=给我讲一个笑话吧`，向通义模型提问并得到回答。

示例开发指南

以上示例本质上就是一个普通的 Spring Boot 应用，我们来通过源码解析看一下具体的开发流程。

1. 添加依赖

首先，需要在项目中添加 `spring-ai-alibaba-starter` 依赖，它将通过 Spring Boot 自动装配机制初始化与阿里云通义大模型通信的 `ChatClient`、`ChatModel` 相关实例。

```
<dependency>
  <groupId>com.alibaba.cloud.ai</groupId>
  <artifactId>spring-ai-alibaba-starter</artifactId>
  <version>1.0.0-M5.1</version>
</dependency>
```

注意：由于 spring-ai 相关依赖包还没有发布到中央仓库，如出现 spring-ai-core 等相关依赖解析问题，请在您项目的 pom.xml 依赖中加入如下仓库配置。

```

<repositories>
  <repository>
    <id>spring-milestones</id>
    <name>Spring Milestones</name>
    <url>https://repo.spring.io/milestone</url>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
  </repository>
</repositories>

```

2. 注入 ChatClient

接下来，在普通 Controller Bean 中注入 `ChatClient` 实例，这样你的 Bean 就具备与 AI 大模型智能对话的能力了。

```

@RestController
@RequestMapping("/helloworld")
public class HelloworldController {
    private static final String DEFAULT_PROMPT = "你是一个博学的智能聊天助手，请根据用户提问回答！";

    private final ChatClient dashScopeChatClient;

    public HelloworldController(ChatClient.Builder chatClientBuilder) {
        this.dashScopeChatClient = chatClientBuilder
            .defaultSystem(DEFAULT_PROMPT)
            // 实现 Chat Memory 的 Advisor
            // 在使用 Chat Memory 时，需要指定对话 ID，以便 Spring AI 处理上下文。
            .defaultAdvisors(
                new MessageChatMemoryAdvisor(new
InMemoryChatMemory())
            )
            // 实现 Logger 的 Advisor
            .defaultAdvisors(
                new SimpleLoggerAdvisor()
            )
            // 设置 ChatClient 中 ChatModel 的 Options 参数
            .defaultOptions(
                DashScopeChatOptions.builder()
                    .withTopP(0.7)
                    .build()
            )
            .build();
    }

    @GetMapping("/simple/chat")
    public String simpleChat(String query) {
        return dashScopeChatClient.prompt(query).call().content();
    }
}

```

以上示例中，ChatClient 使用默认参数调用大模型，Spring AI Alibaba 还支持通过 DashScopeChatOptions 调整与模型对话时的参数，DashScopeChatOptions 支持两种不同维度的配置方式：

1. 全局默认值，即 ChatClient 实例初始化参数

可以在 application.yaml 文件中指定 spring.ai.dashscope.chat.options.* 或调用构造函数 ChatClient.Builder.defaultOptions(options)、DashScopeChatModel(api, options) 完成配置初始化。

2. 每次 Prompt 调用前动态指定

```
String result = dashScopeChatClient
    .prompt(query)
    .options(DashScopeChatOptions.builder().withTopP(0.8).build())
    .call()
    .content();
```

关于 DashScopeChatOptions 配置项的详细说明，请查看参考手册。

更多资料

基础示例与API使用

- [ChatClient 详细说明](#)
- [Prompt Template 提示词模板](#)
- [Function Calling](#)

高级示例

- [使用 RAG 开发 Q&A 答疑助手](#)
- [具备连续对话能力的聊天机器人](#)