dgroove
Tecnologia emozionale.

ICT Training Center

Il tuo partner per la Formazione e la Trasformazione digitale della tua azienda

Spring AI

Generative Artificial Intelligence con Java

Simone Scannapieco

Corso base per Venis S.p.A, Venezia, Italia

Settembre 2025

🔧 JSON OUTPUT CONVERTER PROMPT TEMPLATES

Progetto Spring AI
Passaggi

1. Nuovo *string template* per richiesta definizione da dizionario con formato di *output* automatizzato
2. Creazione modelli per riposta da LLM
3. Modifica *file* `GeminiFromClientService.java` e `GeminiFromClientServiceImpl.java`
4. Modifica *file* `QuestionFromClientController.java`
5. *Test* delle funzionalità con Postman/Insomnia

Progetto Spring AI
Modello di richiesta

⚠ Il *path* `resources/templates` viene creato automaticamente da Spring AI in fase di creazione del progetto

File **`get-json-output-converter-format-definition-prompt.st`**

```
Dammi una definizione del termine {lemma}, come riporterebbe un dizionario. {formato}
```

Progetto Spring AI
Risposta

Modello di risposta

```java
package it.venis.ai.spring.demo.model;

import java.util.List;

import com.fasterxml.jackson.annotation.JsonPropertyDescription;

public record DefinitionResponse(
        @JsonPropertyDescription("Il lemma da definire") String lemma,
        @JsonPropertyDescription("La definizione del lemma") String description,
        @JsonPropertyDescription("Esempi di utilizzo") List<DefinitionExample> examples) {
}
```

Modello di sotto-risposta (esempi)

```java
package it.venis.ai.spring.demo.model;

public record DefinitionExample(String example) {}
```

Interfaccia servizio Gemini (da *client*)

```java
package it.venis.ai.spring.demo.services;

import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.DefinitionRequest;
import it.venis.ai.spring.demo.model.DefinitionResponse;
import it.venis.ai.spring.demo.model.Question;

public interface GeminiFromClientService {

    String getAnswerFromClient(String question);

    Answer getAnswerFromClient(Question question);

    Answer getDefinitionFromClient(DefinitionRequest definitionRequest);

    Answer getCustomFormatDefinitionFromClient(DefinitionRequest definitionRequest);

    Answer getJSONUserFormatDefinitionFromClient(DefinitionRequest definitionRequest);

    DefinitionResponse getJSONOutputConverterFormatDefinitionFromClient(DefinitionRequest definitionRequest);

}
```

Implementazione servizio Gemini (da *client*) - I

```java
package it.venis.ai.spring.demo.services;

import java.util.Map;
import org.springframework.ai.chat.client.ChatClient;
import org.springframework.ai.template.st.StTemplateRenderer;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.core.io.Resource;
import org.springframework.stereotype.Service;
import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.DefinitionRequest;
import it.venis.ai.spring.demo.model.Question;

@Service
public class GeminiFromClientServiceImpl implements GeminiFromClientService {

    private final ChatClient chatClient;

    public GeminiFromClientServiceImpl(ChatClient.Builder chatClientBuilder) {
        this.chatClient = chatClientBuilder.build();
    }

    @Override
    public String getAnswerFromClient(String question) {
        return this.chatClient.prompt()
                .user(question)
                .call()
                .content();
    }
    ...
```

Implementazione servizio Gemini (da *client*) - II

```java
    ...

    @Value("classpath:templates/get-json-output-converter-format-definition-prompt.st")
    private Resource JSONOutputCOnverterFormatDefinitionPrompt;

    @Override
    public DefinitionResponse getJSONOutputConverterFormatDefinitionFromClient(DefinitionRequest definitionRequest) {

        BeanOutputConverter<DefinitionResponse> converter = new BeanOutputConverter<>(DefinitionResponse.class);

        String format = converter.getFormat();

        System.out.println(format);

        String chatResponse = this.chatClient.prompt()
                .user(u -> u.text(this.JSONOutputCOnverterFormatDefinitionPrompt)
                        .params(Map.of("lemma", definitionRequest.lemma(), "formato", format)))
                .templateRenderer(StTemplateRenderer.builder().startDelimiterToken('{')
                        .endDelimiterToken('}')
                        .build())
                .call()
                .content();

        return converter.convert(Objects.requireNonNull(chatResponse));

    }

}
```

Implementazione controllore REST (da *client*)

```
package it.venis.ai.spring.demo.controllers;

import org.springframework.web.bind.annotation.RestController;
...
import it.venis.ai.spring.demo.services.GeminiFromClientService;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;

@RestController
public class QuestionFromClientController {

    private final GeminiFromClientService geminiService;

    ...

    @PostMapping("/client/definition/custom")
    public Answer getCustomFormatDefinition(@RequestBody DefinitionRequest definitionRequest) {
        return this.geminiService.getCustomFormatDefinitionFromClient(definitionRequest);
    }

    @PostMapping("/client/definition/json/user")
    public Answer getJSONUserFormatDefinition(@RequestBody DefinitionRequest definitionRequest) {

        return this.geminiService.getJSONUserFormatDefinitionFromClient(definitionRequest);

    }

    @PostMapping("/client/definition/json/converter")
    public DefinitionResponse getJSONOutputConverterFormatDefinition(@RequestBody DefinitionRequest definitionRequest) {
        return this.geminiService.getJSONOutputConverterFormatDefinitionFromClient(definitionRequest);
    }

}
```

Codice

Branch di riferimento

`https://github.com/simonescannapieco/spring-ai-base-dgroove-venis-code.git`

Branch:

`10-spring-ai-gemini-json-output-converter-format-prompt-template`