



ICT Training Center

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



Note



SPRING AI

GENERATIVE ARTIFICIAL INTELLIGENCE CON JAVA

Simone Scannapieco

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

Novembre 2025

Note



TOOL CALLING

CONCETTI BASE

Note

- ➔ Servizio *tool calling* per richiesta ora locale/per fuso orario
 - 1 Creazione *tool* per supporto ora/locale/*timezone*
 - 2 Configurazione *client* Ollama e Gemini per utilizzo *tool*
 - 3 Creazione interfaccia e implementazione servizio
 - 4 Test delle funzionalità con Postman/Insomnia

Note

Metodi di estrazione ora/locale/timezone

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

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
...
import java.time.LocalTime;
import java.time.ZoneId;

@Component
public class TimeTools {

    private static final Logger LOGGER = LoggerFactory.getLogger(TimeTools.class);

    @Tool(name="getCurrentLocalTime", description = "Ottieni l'orario corrente nella timezone dell'utente.")
    String getCurrentLocalTime() {
        LOGGER.info("Ritorno dell'orario corrente nella timezone dell'utente");
        return LocalTime.now().toString();
    }

    @Tool(name = "getCurrentTime", description = "Ottieni l'orario corrente nella timezone specificata.")
    public String getCurrentTime(@ToolParam(description = "Valore che rappresenta la timezone.") String timeZone) {
        LOGGER.info("Ritorno dell'orario corrente nella timezone {}", timeZone);
        return LocalTime.now(ZoneId.of(timeZone)).toString();
    }
}
```

Note

Configurazione Ollama e Gemini per tool calling - I

```
package it.venis.ai.spring.demo.config;  
...  
  
@Configuration  
public class ToolConfig {  
  
    @Bean  
    public ChatClient geminiToolChatClient(OpenAiChatModel geminiChatModel, TimeTools timeTools) {  
  
        ChatClient.Builder chatClientBuilder = ChatClient.builder(geminiChatModel);  
  
        return chatClientBuilder  
            .defaultTools(timeTools)  
            .defaultAdvisors(List.of(new SimpleLoggerAdvisor()))  
            .defaultSystem(  
                """  
                    Sei un assistente AI di nome ToolGeminiBot, addestrato per intrattenere una  
                    conversazione con un umano.  
                    Usa i tool forniti dall'utente per fornire la tua risposta.  
                """)  
            .defaultUser(  
                """  
                    Come puoi aiutarmi?  
                """)  
        .build();  
  
    }  
...  
}
```

Note

Configurazione Ollama e Gemini per tool calling - II

```
...
@Bean
public ChatClient ollamaToolChatClient(OllamaChatModel ollamaChatModel, TimeTools timeTools) {

    ChatClient.Builder chatClientBuilder = ChatClient.builder(ollamaChatModel);

    return chatClientBuilder
        .defaultTools(timeTools)
        .defaultAdvisors(List.of(new SimpleLoggerAdvisor()))
        .defaultSystem(
            """
                Sei un assistente AI di nome ToolLlamaBot, addestrato per intrattenere una
                conversazione con un umano.
                Usa i tool forniti dall'utente per fornire la tua risposta.
            """
        )
        .defaultUser(
            """
                Come puoi aiutarmi?
            """
        )
        .build();
}

}
```

Note

Interfaccia servizio

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

import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.QuestionRequest;

public interface ToolService {

    public Answer getGeminiToolLocalTimeAnswer(QuestionRequest request);

    public Answer getOllamaToolLocalTimeAnswer(QuestionRequest request);

}
```

Note

Implementazione servizio - I

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

...
@Service
public class ToolServiceImpl implements ToolService {

    private final ChatClient geminiToolChatClient;
    private final ChatClient ollamaToolChatClient;

    public ToolServiceImpl(
        @Qualifier("geminiToolChatClient") ChatClient geminiToolChatClient,
        @Qualifier("ollamaToolChatClient") ChatClient ollamaToolChatClient) {
        this.geminiToolChatClient = geminiToolChatClient;
        this.ollamaToolChatClient = ollamaToolChatClient;
    }

    @Override
    public Answer getGeminiToolLocalTimeAnswer(QuestionRequest request) {
        return new Answer(this.geminiToolChatClient
            .prompt()
            .user(request.body().question())
            .call()
            .content());
    }
}
...
```

Note

Implementazione servizio - II

Note

Implementazione controllore REST

```
package it.venis.ai.spring.demo.controllers;  
...  
  
@RestController  
public class QuestionController {  
  
    ...  
    private final ToolService toolService;  
  
    public QuestionController(QuestionService service, RAGService ragService, ToolService toolService) {  
        this.service = service;  
        this.ragService = ragService;  
        this.toolService = toolService;  
    }  
  
    ...  
  
    @PostMapping("/gemini/ask/time-tools/local-time")  
    public Answer getGeminiToolLocalTimeAnswer(@RequestBody QuestionRequest request) {  
        return this.toolService.getGeminiToolLocalTimeAnswer(request);  
    }  
  
    @PostMapping("/ollama/ask/time-tools/local-time")  
    public Answer getOllamaToolLocalTimeAnswer(@RequestBody QuestionRequest request) {  
        return this.toolService.getOllamaToolLocalTimeAnswer(request);  
    }  
}
```

Note

<https://github.com/simonescannapieco/spring-ai-advanced-dgroove-venis-code.git>

Branch: 12-spring-ai-gemini-ollama-time-tools

Note