



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



Note			



## **SPRING AI**

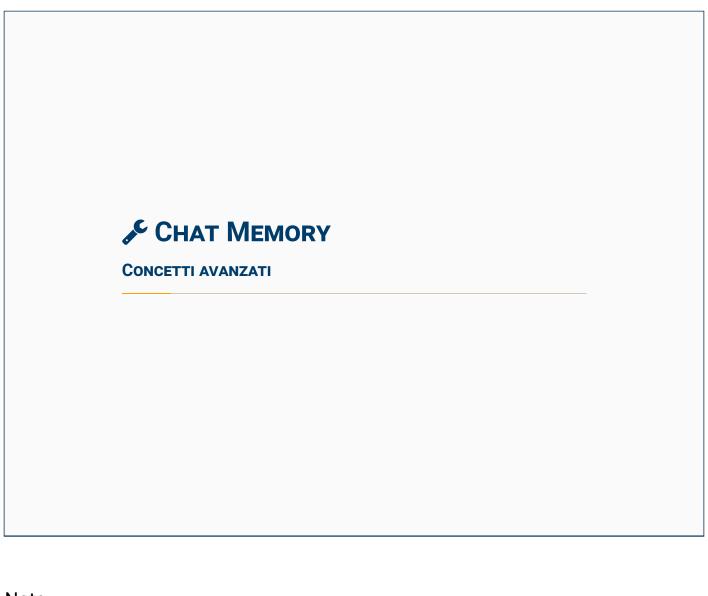
## GENERATIVE ARTIFICIAL INTELLIGENCE CON JAVA

#### Simone Scannapieco

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

Novembre 2025

Note	



Note	

### CONCETTI AVANZATI PER-USER CHAT MEMORY



- Ocome tenere traccia di diverse conversazioni?
  - ChatMemory definita secondo una logica multi-conversazione
    - ▲ CONVERSATION\_ID chiave che il sistema ricerca nel contesto per CRUD storico messaggistica
    - ▲ DEFAULT\_CONVERSATION\_ID valore di default fornito alla chiave se non popolata programmaticamente

## **Interfaccia Chat Memory**

```
public interface ChatMemory {
   String DEFAULT_CONVERSATION_ID = "default";
   /**
   * The key to retrieve the chat memory conversation id from the context.
   */
   String CONVERSATION_ID = "chat_memory_conversation_id";
   ...
```

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

3/19

#### **CONCETTI AVANZATI PER-USER CHAT MEMORY**



- Come sovrascrivere il valore di default?
  - Agire a livello di chat memory advisor
  - Utilizzo delle advisor specifications di ChatClient (ChatClient\$AdvisorSpec)

## Interfaccia per ogni chat memory advisor

```
public interface BaseChatMemoryAdvisor extends BaseAdvisor {
     * Retrieve the conversation ID from the given context or return the default conversation ID when not found.
    default String getConversationId(Map<String, Object> context, String defaultConversationId) {
         Assert.notNull(context, "context cannot be null");
         Assert.noNullElements(context.keySet().toArray(), "context cannot contain null keys");
         Assert.hasText(defaultConversationId, "defaultConversationId cannot be null or empty"); return context.containsKey(ChatMemory.CONVERSATION_ID) ? context.get(ChatMemory.CONVERSATION_ID).toString()
                  : defaultConversationId;
    7
```

#### Customizzazione id conversazione

```
String conversation_id = "custom_conversation_id";
ChatClient
    .prompt()
    .advisors(advisorSpec -> advisorSpec.param(ChatMemory.CONVERSATION_ID, conversation_id))
    .build();
```

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

4/19

# PROGETTO SPRING AI APPLICAZIONE E PASSAGGI



- Onfigurazione per-user chat memory per ChatClient Ollama
  - 1 Creazione modello per richiesta domanda con username
  - 2 Modifica interfaccia e implementazione del servizio di risposta
  - Modifica controllore MVC
  - 4 Test delle funzionalità con Postman/Insomnia

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

Note		



## Modello richiesta con username

```
package it.venis.ai.spring.demo.model;
import java.util.Objects;
import java.util.UUID;

public record QuestionRequest(UUID id, String username, Question body) {
    public QuestionRequest(String username, Question body) {
        this(UUID.randomUUID(), username, body);
    }

    @Override
    public String username() {
        return Objects.requireNonNullElse(this.username, "default");
    }
}
```

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

6/19



```
Interfaccia servizio

package it.venis.ai.spring.demo.services;
import it.venis.ai.spring.demo.model.Ansver;
import it.venis.ai.spring.demo.model.Question;
import it.venis.ai.spring.demo.model.Question;
import it.venis.ai.spring.demo.model.QuestionRequest;

public interface QuestionService {

    Answer getGeminiAnsver(Question question);
    Answer getOllamaAnsver(Question question);
    Answer getOllamaMemoryAwareAnsver(Question question);
    Answer getOllamaPerUserMemoryAwareAnsver(QuestionRequest request);
}

Simone Scannapieco

Spring Al-Corso avanzato
```

Note		



### Implementazione servizio

```
package it.venis.ai.spring.demo.services;
@Service
@Configuration
public class QuestionServiceImpl implements QuestionService {
    private final ChatClient geminiChatClient;
    private final ChatClient ollamaChatClient;
    private final ChatClient ollamaMemoryChatClient;
    public\ Question Service Impl(@Qualifier("geminiChatClient")\ ChatClient\ geminiChatClient,
             @Qualifier("ollamaChatClient") ChatClient ollamaChatClient,
             @Qualifier("ollamaMemoryChatClient") ChatClient ollamaMemoryChatClient) {
        this.geminiChatClient = geminiChatClient;
this.ollamaChatClient = ollamaChatClient;
        this.ollamaMemoryChatClient = ollamaMemoryChatClient;
    @Override
    public Answer getOllamaPerUserMemoryAwareAnswer(QuestionRequest request) {
        return new Answer(this.ollamaMemoryChatClient
                 .advisors(advisorSpec -> advisorSpec.param(ChatMemory.CONVERSATION_ID, request.username()))
                 .user(request.body().question())
                 .content());
```

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

8/19

Simone Scannapieco



m Venis S.p.A, Venezia, IT

9/19

```
Implementazione controllore REST

package it.venis.ai.spring.demo.controllers;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;
import it.venis.ai.spring.demo.model.Answer;
import it.venis.ai.spring.demo.model.Question;
import it.venis.ai.spring.demo.model.Question;
import it.venis.ai.spring.demo.services.QuestionService;

@RestController
public class QuestionController {

...

@PostMapping("/ollama/ask/memory")
public Answer getOllama/MemoryAwareAnswer(QRequestBody Question question) {

return this.service.getOllamaMemoryAwareAnswer(QRequestBody QuestionRequest request) {

return this.service.getOllamaPerUserMemoryAwareAnswer(request);
}
```

Note			

Spring AI - Corso avanzato



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

Branch: 5-spring-ai-gemini-ollama-per-user-chat-memory

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

Ν	ote
---	-----

### **CONCETTI AVANZATI PERSISTENT CHAT MEMORY**



- Storico della conversazione valida per singola sessione
  - Opzione valida per demo/applicazioni a bassa criticità
  - Possibile persistere le informazioni di storico su DB
    - PostgreSQL
    - MySQL/MariaDB
    - SQL Server
    - HSQLDB

## Dipendenza di progetto

<dependency>

<groupId>org.springframework.ai

<artifactId>spring-ai-starter-model-chat-memory-repository-jdbc</artifactId>

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

N	O.	te



#### Definizione bean

```
@Autowired
JdbcChatMemoryRepository chatMemoryRepository;

ChatMemory chatMemory = MessageWindowChatMemory.builder().chatMemoryRepository(chatMemoryRepository)
    .maxMessages(10)
    .build();
```

1 Il sistema inferisce la tipologia di DB dalle proprietà dell'applicativo

#### File application.yml

```
spring:
    datasource:
        url: jdbc:postgresql:...
```

Il sistema applica lo schema relativo al DB inferito

### File schema-postgresql.sql

```
CREATE TABLE IF NOT EXISTS SPRING_AI_CHAT_MEMORY (
    conversation_id VARCHAR(36) NOT NULL,
    content TEXT NOT NULL,
    type VARCHAR(10) NOT NULL CHECK (type IN ('USER', 'ASSISTANT', 'SYSTEM', 'TOOL')),
    "timestamp" TIMESTAMP NOT NULL
    );

CREATE INDEX IF NOT EXISTS SPRING_AI_CHAT_MEMORY_CONVERSATION_ID_TIMESTAMP_IDX
ON SPRING_AI_CHAT_MEMORY(conversation_id, "timestamp");
```

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

12/19

## CONCETTI AVANZATI DEFINIZIONE NUOVO CONNETTORE

Simone Scannapieco



m Venis S.p.A, Venezia, IT

13 / 19

1 Definire un nuovo schema per il nuovo connettore

```
File ./resources/schema/schema-h2.sql

CREATE TABLE SPRING_AI_CHAT_MEMORY (
    conversation_id VARCHAR(36) NOT NULL,
    content LONGVARCHAR NOT NULL,
    type VARCHAR(10) NOT NULL,
    "timestamp" TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL
);

CREATE INDEX SPRING_AI_CHAT_MEMORY_CONVERSATION_ID_TIMESTAMP_IDX
ON SPRING_AI_CHAT_MEMORY(conversation_id, timestamp DESC);

ALTER TABLE SPRING_AI_CHAT_MEMORY ADD CONSTRAINT TYPE_CHECK CHECK (type IN ('USER', 'ASSISTANT', 'SYSTEM', 'TOOL'));
```

Note		

Spring AI - Corso avanzato

## CONCETTI AVANZATI DEFINIZIONE NUOVO CONNETTORE



13/19

2 Impostare i corretti parametri dell'applicativo

```
File application.yml
spring:
   ai:
       chat:
           memory:
               repository:
                  jdbc:
                       initialize-schema: always # always, never, embedded (default)
                       schema: classpath:/schema/schema-h2.sql
       url: jdbc:h2:file:./db/chatmemory
       driver-class-name: org.h2.Driver
       username: user
       password: pwd
   h2:
       console:
           enabled: true
```

te

# PROGETTO SPRING AI APPLICAZIONE E PASSAGGI



- Onfigurazione JDBC H2 chat memory per ChatClient Ollama
  - 1 Modifica al pom.xml per dipendenze Spring Al JDBC, H2 e devtools
  - 2 Modifica ad application.yml
  - 3 Creazione schema H2
  - 4 Modifica ai bean ChatClient per Ollama
  - 5 Test delle funzionalità con Postman/Insomnia

Note		

# PROGETTO SPRING AI JDBC CHAT MEMORY

Simone Scannapieco



m Venis S.p.A, Venezia, IT

15/19

Spring AI - Corso avanzato

Note		

# PROGETTO SPRING AI JDBC CHAT MEMORY



## Configurazione applicativo

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

16/19

# PROGETTO SPRING AI JDBC CHAT MEMORY



### Schema H2

🐸 Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

N	ot	e

# PROGETTO SPRING AI JDBC CHAT MEMORY



m Venis S.p.A, Venezia, IT

18/19

## Configurazione Gemini + Ollama

Spring AI - Corso avanzato

#### Note

👺 Simone Scannapieco



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

Branch: 6-spring-ai-gemini-ollama-jdbc-chat-memory

Simone Scannapieco

Spring AI - Corso avanzato

m Venis S.p.A, Venezia, IT

Ν	ot	e
---	----	---