Documentazione applicazione "*Cartoons Catalog*"

Corso di Programmazione Avanzata a.a.2024/2025

Studente Guidi Dario

1 Caso d'uso

1.1 Descrizione caso d'uso e manuale utente

L'applicazione distribuita *CartoonsCatalog* è un semplice *catalogo dei personaggi del cartone "Rick and Morty"*. L'utente avvia l'applicazione, preme il bottone "*Carica catalogo*" della prima finestra, per caricare appunto le informazioni dei personaggi, e successivamente compare un'altra finestra dove visualizza una tabella dell'id e del nome dei personaggi caricati. Selezionando con il tasto destro del mouse un elemento della tabella compare un menu a scomparsa con la voce "*Rimuovi*" per poterlo rimuovere. Con il click del bottone "*Sfoglia catalogo*" si apre una nuova finestra dove ci sono i dettagli in più su ogni personaggio accompagnati con un'immagine relativa sempre allo stesso. I dettagli su ogni personaggio appare singolarmente nella finestra ed è necessario utilizzare i bottoni "<<" e ">>" per scorrerli tutti. Per tornare alla finestra dove è presente la tabella dei personaggi premere il bottone "*Indietro*".

2 Scambio dei dati

2.1 Metodo di scambio dati tra applicazione e servizio

I dati scambiati sono codificati in formato *JSON*. L'applicazione *carica*, *legge* e *rimuove* i dati sui personaggi tramite delle richieste http al servizio *CartoonsCatalogServer*. Il servizio risponde all'URL "*http://localhost:8080/607453/<azione>"*. Il tipo di azione richiesta è codificata nell'URL: per *caricare i dati dal servizio* si specifica come azione *"/caricadati*"; per ottenere tutti i personaggi disponibili dal servizio si specifica come azione "*/tutti*"; per rimuovere un certo elemento selezionato dal servizio si specifica "*/rimuovi?id=<id personaggio>*".
L'immagine di ciascun personaggio è raggiungibile tramite un altro URL che è fornito dal servizio insieme ai dettagli del personaggio visualizzato. Quindi per essere scaricata è necessaria un'altra richiesta http.

3 Servizio

3.1 Struttura del software e scelte progettuali

Il servizio è stato realizzato come un progetto Maven utilizzando il framework Spring. I dati caricati nel database "607453", all'inizio del caso d'uso dell'applicazione, sono prelevati dall'API al sito Web https://rickandmortyapi.com/api/character. È stato scelto un progetto con l'uso di Spring poiché l'applicazione è molto semplice e richiedeva solamente la lettura e la scrittura di query SQL molto semplici, come leggere tutte le entrate della tabella, ritornare il conto delle entrate presenti e rimuovere un'entrata tramite l'Id.

La classe per la memorizzazione dei personaggi è definita come *Character.java*. Per ogni istanza di questa classe, si memorizza: un identificatore *id*; uno *status*; la specie *species*; il genere *gender*; l'origine *origin*; l'url dell'immagine del personaggio *imageUrl*. Inoltre, sono definiti il costruttore, i getter e setter.

Il codice dove sono implementati i metodi esposti dal servizio per l'applicazione si trovano nel file *MainController.java*. La classe *MainController* utilizza l'interfaccia *characterRepository,* della classe interfaccia *CharacterRepository* che estende *CrudRepositry<Character, Integer>*, per interagire con il database.

La funzione "*long loadCharacters*" (metodo richiesta http GET) legge tutti i dati dal servizio web pubblico, con una richiesta http, li memorizza nella tabella "*characters*" del database e ritorna il valore numerico delle entrate memorizzate.

La funzione "*Iterable<Character> getAllCharacters()*" (metodo richiesta http *GET*) legge tutti i dati dal database ritornandoli in formato JSON.

La funzione "*String removeCharacter(Integer id)*" (metodo richiesta http *POST*) rimuove l'entrata della tabella con l'Id = *id* e ritorna una stringa di conferma che la rimozione è avvenuta con successo.

3.2 Fonti utilizzate

Il link di riferimento dove trovare i metodi '*void deleteById(Integer id)*' e '*long count()*' dell'interfaccia estesa *CrudRepository<Character, Integer>* utilizzati nel progettino è il seguente:

https://docs.spring.io/spring-data/commons/docs/current/api/org/springframework/data/repository/CrudRepository.html

3.3 Unit Test del metodo "/tutti"

Nella cartella dei Test Packages è implementato lo Unit Test della richiesta http del metodo "*/tutti*", "*void getAllCharacters_response200()*". Se lo status della risposta alla richiesta http è 200, allora il test è andato a buon fine.

3.4 Unit Test del metodo "/rimuovi"

Nella cartella dei Test Packages è implementato uno Unit Test "" della richiesta http del metodo "*/rimuovi?id=20*", "*void removeCharacter_response200()*". Se lo status della risposta alla richiesta http è 200, allora il test è andato a buon fine.

4 Applicazione

4.1 Struttura del software e scelte progettuali

L'applicazione è un progetto Maven che utilizza il framework di JavaFX per la realizzazione di un'applicazione con interfaccia grafica.

La classe per la memorizzazione dei personaggi è definita come *Character.java*. Per ogni istanza di questa classe, si memorizza: un identificatore *id*; uno *status*; la specie *species*; il genere *gender*; l'origine *origin*; l'url dell'immagine del personaggio *imageUrl*. Inoltre, sono definiti i getter ed il costruttore.

Utilizza tre finestre: "*primary.fxml*", "*secondary.fxml*" e "*catalog.fxml*". La logica che gestisce le interazioni dell'utente con le finestre è implementata rispettivamente nelle classi *PrimaryController*, *SecondaryController* e *CatalogController*.

La classe *PrimaryController* inizializza la finestra con il logo del cartone animato nel metodo "*void initialize()*". Il logo è salvato localmente nel progetto in "*it.unipi.CartoonsCatalog.img*". Inoltre, è definito il metodo "*void loadAndswitchToSecondary()*" associato alla pressione del bottone con id = *loadButton*. Questo metodo fa una richiesta http al servizio con l'azione nell'URL "*./caricadati*" per richiedere di caricare i dati nel database e poi passa alla finestra "*secondary.fxml*".

La classe *SecondaryController* con il metodo "*void initialize()*" si occupa di caricare di nuovo il logo del cartone animato come fa la classe PrimaryController ed in più carica nella *TableView charactersTable* tutti i dati ottenuti facendo una richiesta http al servizio con metodo nell'URL "*./tutti*". Il metodo della classe "*void remove()*" rimuove un elemento dal database tramite una richiesta http al servizio con l'azione nell'URL *"/remove?id=<id elemento selezionato>".* Questo metodo è associato alla selezione, sopra un elemento della tabella, della voce "Rimuovi" nel menu a scomparsa. L'ultimo metodo definito è quello associato alla pressione del bottone con id = *browseButton* per sfogliare il catalogo con i personaggi in dettaglio nella finestra "*catalog.fxml*".

La classe *CatalogController* inizializza, col metodo "*void initialize()*", una variabile *int* che memorizza l'indice del personaggio "sfogliato" e un *ArrayList* di *Character* tutti i personaggi ottenuti dal servizio con la richiesta http e azione specificata "*./tutti*". Dopodiché invoca il metodo "*goToNextCharacter()*" che caricherà il prossimo personaggio (il primo) visualizzando immagine (ottenuta tramite richiesta http all'URL del campo *imageURL* dell'istanza *Character*). Con la pressione dei bottoni id = *nextButton* e id = *previousButton*, vengono invocati i metodi corrispettivi "*goToNextCharacter()*" e "*goToPreviousCharacter()*" per scorrere i personaggi a destra e sinistra. Con la pressione del bottone id = *backButton* viene invocato il metodo "*goBackToSecondary()*" per tornare alla finestra della tabella dei personaggi.