

Introduzione alla Programmazione Web del 21 luglio 2020

Un giornale on line effettua un servizio di webcasting di notizie sportive: con una certa frequenza (circa ogni minuto, o meno) viene dato un aggiornamento su un dato evento (una partita di calcio, una gara di F1...).

Si implementi la web app che permette di mettere in linea un servizio analogo.

	Brescia	0	45'+1' Half-time Live	0	Roma	
BRE						BRE
ROM						ROM
COMMENTARY						
<hr/>						
45'+1'	Fine primo tempo - Si va al riposo sullo 0-0					
45'	1° di recupero					
45'	Lancio sulla destra per Pellegrini: palla troppo lunga che sfila sul fondo					
44'	Lancio in area per Torregrossa, anticipato da Ibanez					
43'	Su corta respinta della difesa biancazzurra, prova il destro al volo da 25 mt Diawara : tiro lento che, però, si perde appena mezzo mt a lato alla destra di Andrenacci					
42'	Sull'angolo dalla sinistra di Pellegrini, Bjarnason anticipa Ibanez e allontana il pericolo					

1) Il servizio ha una componente di feeding: un addetto usa una pagina web per:
- inserire in una coda dei brevi messaggi di testo identificati da un numero progressivo (il primo messaggio inserito crea la coda, se necessario).
- chiudere il webcasting dell'evento tramite uno speciale messaggio "END"
Il sistema è in grado di gestire un solo webcasting alla volta.

2) La componente vista dagli utenti è una pagina web che mostra i nuovi eventi, dall'ultimo visualizzato: se vi sono 10 eventi e l'utente ne ha visti 7, la pagina mostrerà gli ultimi 3. Se l'utente ha già visto tutti gli eventi, gli verrà dato il messaggio "Niente di nuovo al momento".

Ovviamente ciascun utente avrà la propria vista individualizzata. La pagina utente dovrà avere una funzione che permetta di ripristinarne lo stato "dimenticando" gli eventi già visti, e quindi tornando allo stato iniziale.

Se non vi è un webcast attivo, la pagina mostra il messaggio: "Nessun evento disponibile ora".

3) Si curi la formattazione delle pagine.

Si evidenzino con un particolare stile le righe che contengono delle parole chiave (es: gol, rigore, espulso).

4) Per la condivisione di dati tra la parte di feeding e quella vista dall'utente si scelga la soluzione più efficiente (es., File System, Cookie, Session, Database, ServletContext, Variabili di istanza...). Si motivi brevemente la scelta fatta in un file HTML chiamato "motivazione.html" incluso nel progetto (la formattazione del file non è importante).

5) Si aggiunga un refresh automatico della pagina che venga effettuato ogni x secondi (default x=30), dove x è il valore presente in un campo numerico editabile presente nella pagina utente.

Quando il valore di x viene cambiato, la pagina deve cambiare la frequenza di aggiornamento rispettando tale valore.

Quando la pagina “dimentica” gli eventi già visti e torna allo stato iniziale, resetta anche il refresh automatico al valore di default.

Si consiglia di usare due diversi browser per testare le viste di più utenti.

Richiami sulla ServletContext:

ServletContext is an object from Servlet world. It is a shared repository for all servlets belonging to a web application. There is only one ServletContext per web application (so it is shared by all servets and all users).

The ServletContext object is the same as the “application” object in JSPs belonging to the same web application. (https://www.tutorialspoint.com/jsp/jsp_implicit_objects.htm)

It can be acquired by calling

getContext();

in the init, destroy, doGet, doPost, doFilter and all other doX methods. (<https://docs.oracle.com/javaee/5/tutorial/doc/bnagl.html>)

Once you have the ServletContext object, you can add to/retrieve from it objects like you do in an HttpSession, by using the methods

*void setAttribute(String name, Object object)
Object getAttribute(String name)*