Shared Calendar

Onkar Singh (B13321)

Samya Ranjan (B13225)

1st June 2016

Table of Contents

1. Introduction

Shared Calendar refers to a calendar that is shared among multiple clients which are connected to a single server. This utility helps different clients to share calendar events, no matter where they are. The client just needs an active Internet connection, and the server will update the client's calendar within 10 seconds of a successful authentication. So, the consistency is ensured. Every-time a new client connects to a server, the client has to enter his/her username, and the server will check that the client is registered or not. If registered, would authenticate the client, and if not, would deny the connection. So the security issues are handled too.

2. Procedure

The server has a file named as “authenticate.txt” which stores the information of each and every client that can connect to the server. Each client has a file named as “username” which contains information regarding client's username and password.

The server spawns a new thread to handle each client. When the client firstly activates his/her device (here, executing the client program) and he/she has an active Internet connection, he/she has to give his/her username, that is required for authentication. The client program sends this username to the server. The server maintains a list of clients that can connect to the server(i.e. can share the calendar). The server checks this username sent by the client in its list, and if username exists in the list, the client would be authenticated and the client would be synced to the server within 10 seconds of successful authentication, else the server would not allow the client to share the calendar.

After successful authentication, the following menu pops up to the client :

1. Add an Event 2. Delete an Event

3. List events between two timestamps 4. List timestamps of events

If client wants to manipulate calendar in any way, he/she would enter the corresponding choice.

2.1. Add an event

To add an event, the client adds an event to the local temporary file present on the client named as "temp_client_username". After this, client checks the Internet connection, if Internet is active on his/her device, the client requests the server to add an event, the server responds with the acknowledgement to add an event. The client sends this temporary file to the server which contains the event added by the client.

The server receives this file and make a copy of this file and stores it. Now the server transfers the calendar events from this file to the permanent server file named as "perm_server_.ics". The event would be added to the permanent file only if the event's timestamps does not clash with the timestamps of the events present in the permanent file of the server. If the event is added to the permanent file, the entry would also be recorded in the log file maintained by server named as "server_perm_log_file.txt". The entry in the log file would consist of client's username from which the request has come, status of event (i.e. the request is for adding an event), the start and end timestamps of the event added. The server contains another file named as "server_version.txt" which contains the version no of the server which is increased by one, everytime the permanent file is modified.

The server maintains a temporary file which contains all the events added to the permanent file (this file is needed because there might be clashing events, which are not added to the permanent file). Now the server broadcasts this file to all the connected clients. And after this broadcast, sends the copy of the received file (initially a copy received file is stored) to the client which has requested these events.

The broadcast is received by all the connected clients. The clients also contains a permanent calendar file named as "perm_client_username.ics". The client adds the broadcasted events to this permanent file ensuring the timestamps do not clash (necessary, the file should not contain multiple events with same timestamps). Everytime this file is modified the client version number increases by one and stored back in the file named as "client_version_username.txt". The client also receives the copy of the file which it has sent to the server. The client removes all the events present in this file from the temporary file which was sent to the server. The add operation completes with no flaws.

2.2. Remove an event

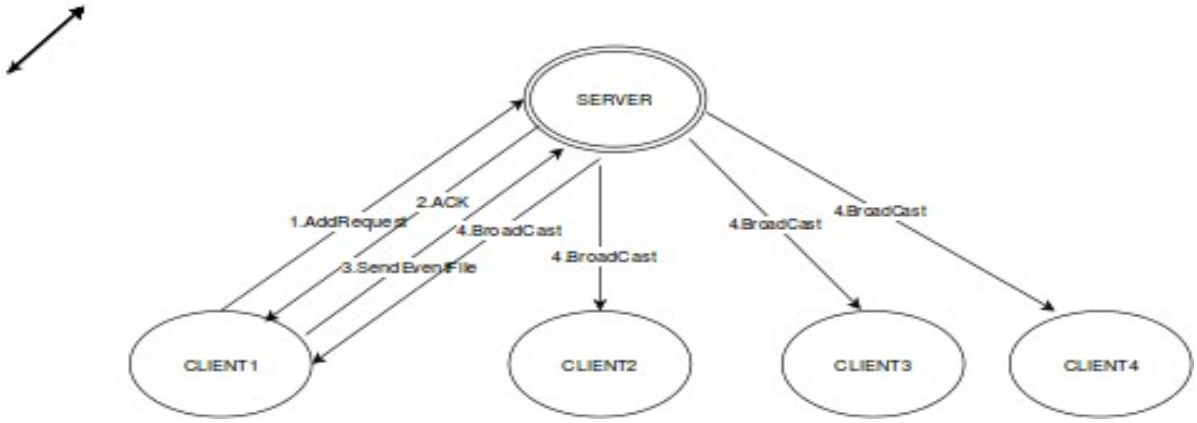
To remove an event, the client program asks for a timestamp of an event to remove. This adds a this timestamp to the local temporary file present on the client named as "username_remove.txt". After this, client checks the Internet connection, if Internet is active on his/her device, the client requests the

server to remove an event, the server responds with the acknowledgement to remove an event. The client sends this temporary file to the server which contains the timestamps of the events requested to be removed by the client.

The server receives this file, finds the events with requested timestamps in the permanent server file named as “perm_server_.ics”. If the events exist in the permanent file, the server removes the events from the permanent file. And if the event is deleted from the permanent file, the entry would also be recorded in the log file maintained by server named as “server_perm_log_file.txt”. The entry in the log file would consist of client's username from which the request has come, status of event (i.e. the request is for deleting an event), the start and end timestamps of the event deleted. The server contains another file named as “server_version.txt” which contains the version no of the server which is increased by one, everytime the permanent file is modified. Now the server broadcasts the file received by the client to all the connected clients.

The broadcast is received by all the connected clients. The clients also contains a permanent calendar file named as “perm_client_username.ics”. The client deletes the broadcasted events from this permanent file. Everytime this file is modified the client version number increases by one and stored back in the file named as “client_version_username.txt”. The client also receives the copy of the file which it has sent to the server. The client removes all the timestamps of the events present in this file from the temporary file which was sent to the server. The remove operation completes with no flaws.

Fig. 1 Basic diagram of the shared calendar.



2.3. Handling disconnections

When a client gets disconnected from the server (due to no active Internet connection), the client program can still add, delete or list events from the local copy. While adding an event, the event gets added to the “temp_client_username.ics” file. The event is stored in the temporary copy until it regains active Internet connection and after getting connected to the server , the Sync function is called , which syncs the stale local copy with recent server copy. After the local copy is updated , the events added to the temporary local copy are propagated to the server, the server updates its permanent copy and

broadcasts the events to all the clients. The same concept works for delete event as well. For listing events a clients can see all the events that are added to the permanent copy of the client. A Client can request events between two timestamps . The Sync function on the server reads the server log file to find out how events which were added and which were removed . The removed event timestamps are stored in a text file and sent to the client. While for the events that were added during the disconnection the server reads log file and finds the corresponding events and adds it into an ics file. Later the ics file is sent to the client. The client adds the events into its permanent file and increases its version no.

3. Conclusions

Hence, we summarize our work and provide a utility of a shared calendar. This shared calendar utility helps multiple clients to share a calendar using Internet.

Contribution of Each Member

Onkar Singh : UI/UX Design, Checking consistency and providing Durability, Synchronization of files File Handling (ics files using iCal4j Library) on client as well as server.

Samya Ranjan : Consistency Protocols, Handling concurrency on both server and client. Communication between server and client

Shared :

- Sending and Receiving Protocols.
- Design of algorithm.

Libraries Included

- javamail.jar
- activation.jar
- backport-util-concurrent.jar
- commons-lang3-3.4.jar
- commons-logging-1.2.jar
- ical4j-2.0-beta1.jar
- slf4j-api-1.7.13.jar
- slf4j-simple-1.7.13.jar

References

http://cs.lmu.edu/~ray/notes/javanetexamples/#chat

http://ical4j.sourceforge.net/introduction.html

http://www.programcreek.com/java-api-examples/index.php?api=net.fortuna.ical4j.model.DateTime

http://www.tutorialspoint.com/

http://stackoverflow.com/

http://10.8.1.4/~tag/#Tutorials

Declaration

We hereby declare that, we have not copied anything from any sources.