Creating a data backup of a WebSQL database

by Raymond Camden on | Comments

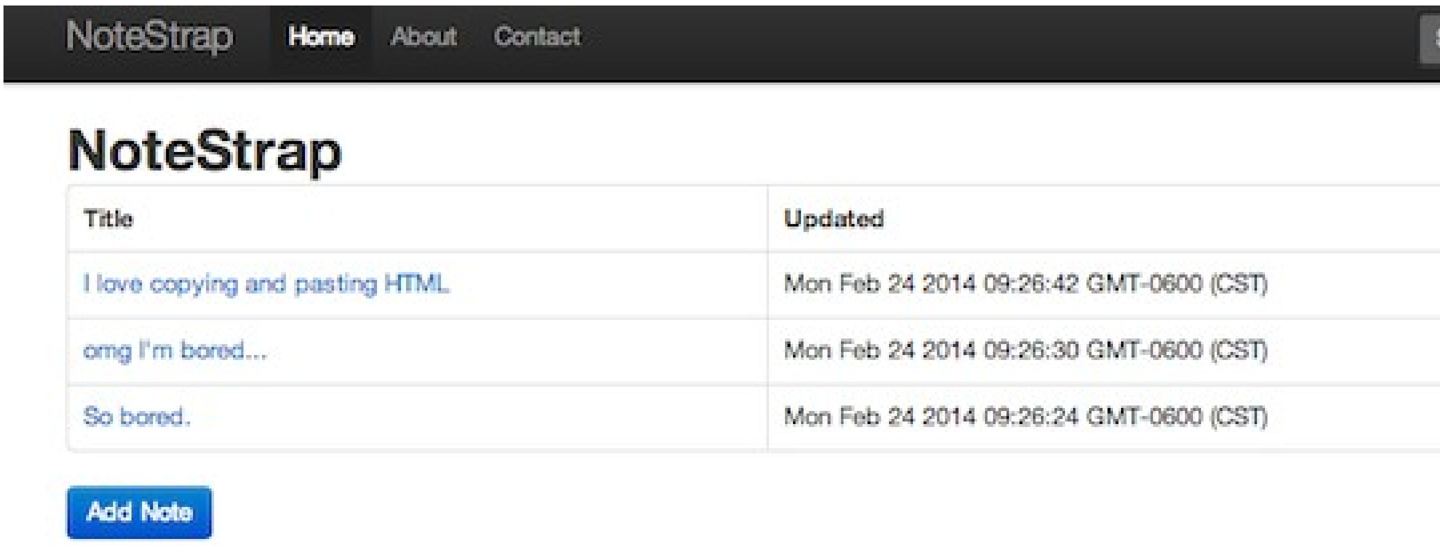
This post is more than 2 years old.

I haven't written much about WebSQL lately, mainly because it is Dead Spec Walking. However, it still works in Cordova (for now), and I get questions from time to time. This one in particular was kind of interesting. Plus, the guy asking me for help with this was super nice even though I kept delaying my answer over a few weeks. :)

His question was simple - how to create a *data* backup of the database. I'm stressing "data" backup because the WebSQL database is stored in a single file. You could - potentially - just grab the bits. But the assumption here is that you want the data from the various tables and not the actual bits.

With that in mind then the solution is rather simple. For each table you select * from it, convert the SQLRecordSet object into a simple array of structs, and then... well... do whatever you want. For my demo I'm converting it to a JSON string with the assumption that you want to send it over the wire.

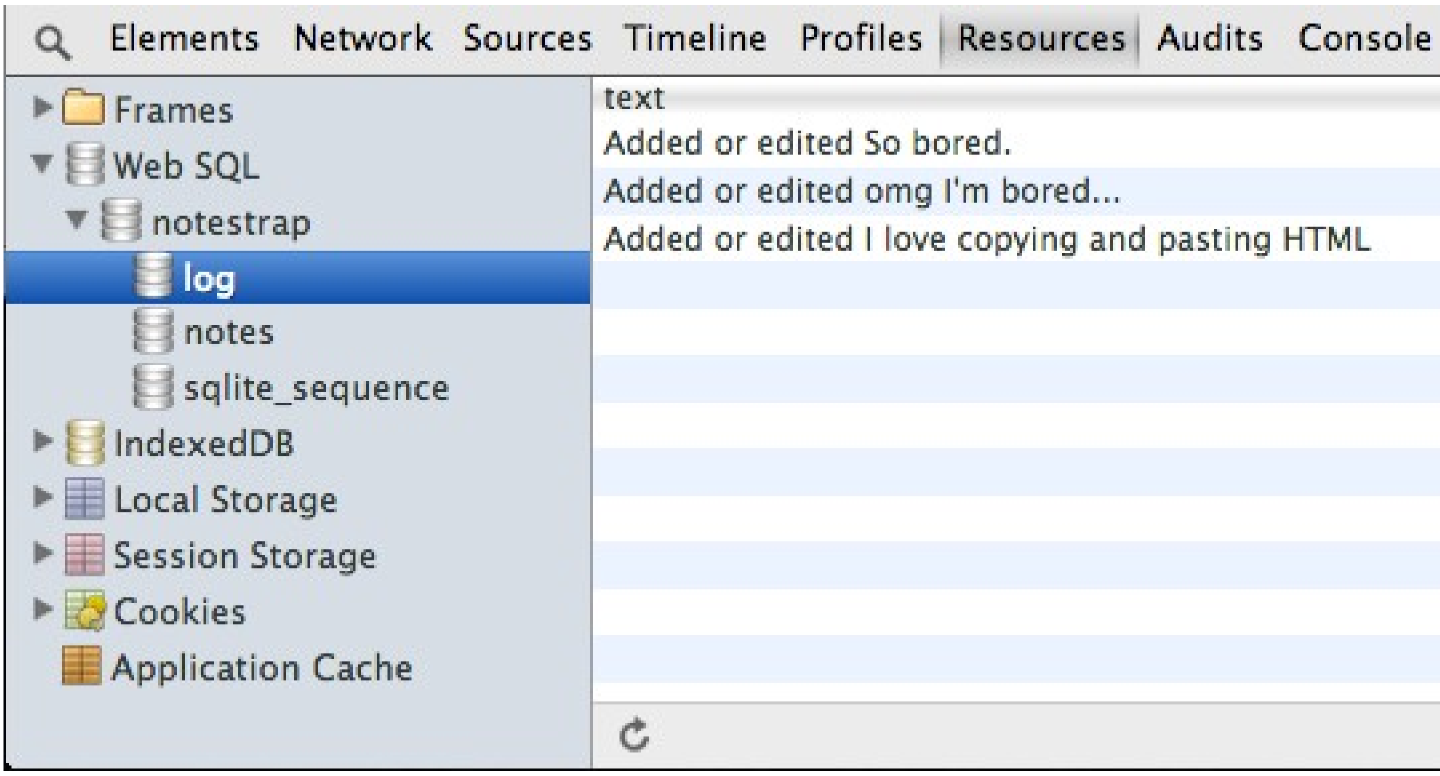
I took an existing demo (NoteStrap) that simply supported saving notes. It has basic CRUD (create, read, update, delete):



NoteStrap

Title	Updated
I love copying and pasting HTML	Mon Feb 24 2014 09:26:42 GMT-0600 (CST)
omg I'm bored...	Mon Feb 24 2014 09:26:30 GMT-0600 (CST)
So bored.	Mon Feb 24 2014 09:26:24 GMT-0600 (CST)

To make things a bit more interesting, I added a second table, called log, where I store log messages for deletes and add/edits.



text
Added or edited So bored.
Added or edited omg I'm bored...
Added or edited I love copying and pasting HTML

Ok, so how do we handle the export? In theory doing a simple SQL select operation is trivial, but remember that the call is asynchronous. Since we have two tables, we need a nice way of handling them and running only after both are done. For that, I used promises. Here is the solution I came up with. (Note, I also added a simple "Backup" button to the UI for testing.)

```
function backup(table) {
        var def = new $.Deferred();
        db.readTransaction(function(tx) {
                tx.executeSql("select * from "+table, [], function(tx,results) {
                        var data = convertResults(results);
                        console.dir(data);
                        def.resolve(data);
                });
        }, dbError);

        return def;
}

$(document).on("click", "#doBackupBtn", function(e) {
        e.preventDefault();
        console.log("Begin backup process");

        $.when(
                backup("notes"),
                backup("log")
        ).then(function(notes, log) {
                console.log("All done");
                //Convert to JSON
                var data = {notes:notes, log:log};
                var serializedData = JSON.stringify(data);
                console.log(serializedData);
        });

});
```

Note that this does *not* properly handle errors, but you get the basic idea. The backup function works with jQuery promises so if I add a new table I just need to append the call in my when block and add the corresponding data to my result.

You can test this yourself here, but please remember it will only work in browsers that support WebSQL.

https://www.raymondcamden.com/2014/02/24/Creating-a-data-backup-of-a-WebSQL-database