Module 05 Activity – Data Models

Patrick A Brunner

Rasmussen College

For the module 05 activity, we will be looking at different technologies for storing data on the web. The three technologies that will be reviewed here include Web SQL Database, Indexed Database, and File Access. For each of these technologies, we will review how each can be used as well as looking at examples of the data models for the different technologies.

To begin we will look at the Web SQL Database. If we look at the article, Introducing Web SQL Databases by Remy Sharp, we see that a Web SQL Database is a specification that can bring SQL databases to the client side. (Sharp, 2010). Some of the actions that can be carried out on the client-side include creating databases, opening the transaction, creating tables, inserting values into tables, deleting values from tables and reading the data. The Web SQL Database API is not actually a part of HTML5specifications, however its part of a suite that allows developers to build web applications. The Web SQL Database specification is built around SQLite and is capable of manipulating client-side databases using SQL. Looking even deeper into the Web SQL Database, we look at the article, Client-side Storage by Michael Mahemoff. This article outlines some of the strengths and weaknesses of Web SQL Database (Mahemoff, 2010). Some of the strengths include:

* Supported on all major mobile browsers (Android and IOS) as well as several desktop browsers (Chrome, Safari).
* Good performance generally, being an asynchronous API. Database interaction won't lock up the user interface.
* Search performance is good as data can be indexed according to search keys.
* Robust since it supports a transactional database model.
* Based on a rigid data structure, it is easier to maintain the integrity of the data.

While weaknesses include:

* Will not be supported on IE or Firefox and may be phased out on other browsers at some time.
* Significant learning curve with need for an understanding of relational databases and SQL.
* Suffers from object-relational impedance mismatch.
* Diminishes agility, as database schema must be defined upfront, with all records in a table matching the same structure.

An example of a transactional database model can be seen below:



Next, we move on to Indexed Database. Continuing with Michael Mahemoff’s article, we see that an Indexed Database is a “collection of "object stores" which you can just drop objects into.” The stores are similar to SQL tables, however there are constraints on the object structure meaning there is no need to define anything up front. “So this is similar to Web Storage, with the advantage that you can have as many databases as you like, and as many stores within each database. But unlike Web Storage, there are important performance benefits: An asynchronous API, and you can create indexes on stores to improve search speed.” Some advantages to Indexed Database include:

* Good performance being an asynchronous API meaning that database interaction will not lock up the user interface.
* Good search performance as data can be indexed according to search keys.
* Supports versioning.
* Robust since it supports a transactional database model.
* Easy to learn based on a simple data model.
* Good browser support.

While weaknesses include:

* Very complex API resulting in large amounts of nested callbacks.

Once again, Indexed Database supports a transactional database model, an example of which is seen above.

Finally, we look at File Access or FileSystem. The FileSystem API is designed to store large files and binary content. “It gives each domain a full hierarchical filesystem, and in Chrome at least, these are real files sitting on the user's hard drive. For reading and writing of individual files, the API builds on the existing File API.” Strengths of the File Access API include:

* Stores large files or binary content so it is ideal for images, audio, video, PDFs, etc.
* Good search performance as data can be indexed according to search keys. Good performance being an asynchronous API meaning that database interaction will not lock up the user interface.

While weaknesses include:

* Only available in Chrome and Opera.
* There is no built-in indexing or search support.

It appears that File Access is able to make use of a more document-based data model, an example of which is below:



As shown here, each of these technologies has both advantages and disadvantages for developers.

References

Sharp, R. (2010).Introducing Web SQL Databases. Retrieved from http://html5doctor.com/introducing-web-sql-databases/.

Mahemoff, M. (2010). Client-side Storage. Retrieved from https://www.html5rocks.com/en/tutorials/offline/storage/.