Technology Report

Adapa Stock Watcher

Technology expert: Nico Boh

Backend

Erlang OTP R16B01

There was never any thought of using another programming language than Erlang for the ETL development. Further information can be found in the Design and Architecture documents.

Apache CouchDB 1.4.0

Instead of using a traditional SQL database, we decided to use a document based approach with Apache CouchDB. This database stores the data in JSON documents, which are accessible via HTTP requests. The query language used is JavaScript.

Pros

CouchDB is a good tool to store an undefined and incalculable amount of data. It allows us to change the schema of our data at any time. Since the data is stored in big documents we can get the data needed in a single query/view. It allows any number of parallel readers and writers. CouchDB serves our applications without using any middle layer. Also updates to the database/documents are handled on the server.

Cons

We need to create views (mostly using JavaScript) for each and every query, what can be pretty tricky since it is not as straight forward as SQL. Queries like WHERE or SORT are not available. We will probably end up with redundant data. It's kind of difficult to merge data of a set of views. However, after a while we concluded that there was no need for this in our system. If there are any changes to be made, we do this in the application layer.

To use CouchDB in Android or Desktop you need to run an Erlang VM on the device. This means every time you start your app, you would need to start the VM as well. This causes slight loss in boot performance and memory used by the app. We fixed this by just using http requests to request the data from the server. We know that this means that our application won't be showing any data without internet connectivity. Regarding the fact that we only want to show the newest data, this is a missing feature we can live with.

Challenges/Changes

It took us a while to get used to a non SQL database environment. We were forced to do a lot of research and learning. In the beginning the learning curve was pretty high, what was a result of the easy and straightforward way CouchDB works. However, in the middle of the development phase we got stuck in updating and querying the database properly. This was most of all caused by the data structure of the documents in our database. We didn't give a big thought to that from the beginning, what was a big mistake. For a short period we even considered using another alternative, like MySQL. Fortunately, restructuring and optimizing the entire database solved this issue.

In the end we are very happy that we kept using CouchDB. It is much handier than RDBMS when it comes to storing big chunks of data. As soon as we got our head around it, querying the database allowed us to evaluate the data very easily and efficiently.

Desktop

Visual C# 5.0 (.NET 4.5)

The main reason we chose C# over other object oriented programming languages like Java or VB.NET, was to challenge ourselves by using an environment nobody worked with before.

Pros

Due to its similarity to Java and the other C languages, the learning progress was very quick. Because of its popularity, it is very easy to find documentation and reliable resources such as tutorials and forums. Visual Studio as an IDE is very straight forward to develop desktop applications with user friendly interfaces. The .NET framework provides us with a great repository of predetermined classes, functionality and support. This allows us to develop the application fast and with less coding effort.

Cons

Since we are developing in a Windows environment and C# applications are designed for Windows operating systems, it is going to be difficult to run our application on other OS. The user would have to install the latest .NET framework on his machine. And even then it is not provided that the app will run. But considering that about 90% of the world's PC users run a Windows operating system it is only a little price to pay.

The standard Windows Forms application gives the designer very little room for creativity. It is hard to change the design of standard controls and the application is most likely going to look like any other Windows application.

WPF 4.5, XAML

Because of this reason we decided to use WPF (Windows Presentation Foundation) application instead. This next generation UI framework allows us to create unique and creative interfaces. The UI is much faster, scalable and resolution independent. One of its biggest strength is that it separates the appearance from the behavior. The appearance is specified in XAML (Extensible Application Markup Language) and the behavior is implemented through C#. The two parts are communicating via data bindings, events and commands. This allows for example to create the UI design completely separate from the functionality.

Challenges/Changes

In the beginning we could see fast progress, especially in the UI design. Although it is fairly easy to construct a UI in WPF using Visual Studio, we figured soon that, if you didn't give enough thoughts to the structure from the beginning, you will most likely run into problems later. This happened to us during the development of the second prototype. Fortunately we could build up the interface from scratch in reasonable time. This was also due to the experience we gained while developing the first prototype.

Mobile App

Android

We went with android because we had some previous experience in the group with making applications of moderate complexity. Also, Android is open, free and relatively easy to use and learn. Documentation is great and also the community on forums as well as the official pages are extensive.

The one thing we've spent more time on than on any other bit of the mobile project were charts. Free charts libraries are hard to come by, and by the few free around not many support candlestick charts. The one we ended up using greatly lacks documentation and support, but it works.

Website

HTML 5

We use HyperText Markup Language Version 5 to provide a general structure of the web interface. CSS 3 provides the styling. The dynamic content is handled by JavaScript and jQuery.

Pros

HTML 5 helps to reduce browser compatibility issues since it is supported by almost all modern web browsers i.e. Internet Explorer 9, Firefox and Google Chrome. It has a cleaner and neater syntax structure what makes code easier, especially for the developers. For instance it reduces the need of Java Script. Many new features and standards have emerged as part of HTML 5. Once you detect the available features in today's browsers, you can take advantage of those features in your application. Main focus of HTML5 is to make easier application with easy front-ends, drag and drop tools, discussion boards, wikis and other useful elements.

Cons

Unfortunately there can be still problems with the browser support. If users use outdated web browsers they can get compatibility issues. HTML 5 is considered a work in progress, so technically any of the elements could change at any time. For the elements we use in the application it is most unlikely that this could have any effect on us.

JavaScript 1.8.5

JavaScript is the primary client side scripting language. Other tools (jQuery, Ajax) are used to make the content dynamic. This allows us to add and remove HTML and CSS elements without reloading the entire page.

jQuery 1.9.1

This API library for JavaScript, makes it easier for us to reuse code. jQuery provides many graphical enhancements like smooth transition of HTML elements, fade etc. Using jQuery also reduces JavaScript compatibility issues. JavaScript does not always behave the same consistently in every browser, unless the developer knows workarounds. Due to jQuery, we do not have to spend time on those.

Ajax

We use Ajax to communicate with our database. It makes it very easy and handy for us to work with the JSON objects we receive from CouchDB.