

Flood Simulation Browser

Richard Torenvliet - 6138861

March 20, 2012

1 Introduction

This project concerns the flood simulation browser. The concept of this browser is to illustrate/visualize a flood in a particular area. With this technology people can see the flow that the water will take. When a dike breaks it is important to know where the water will flow. That way the information about which locations in the area will be under water first. By using this information the government can be informed. People with the authority to take measures against these floods are provided with the right information. This system already exists but this project involves taking advantage of this system and make it easier available and accessible. The implementation of this application will be on a multi-touch device. In particular iPads and Android tablets. but can be launched on other tablets as well. Users can use this application in an intuitive way and get more intelligence about the complex situation at hand. They have the ability to choose from several simulations that where already made for a specific area. This document covers design choices/options in order to reach that goal. The first step is to get an idea of the existing simulation system of which this application will take use of. This includes a clarification of what is already made and what is not. For instance, what type of data is already made at the system side. The next step will be finding what out the requirements of the tablet implementation, like what features does it have to contain in order to be to maintain the intuitiveness. Just by showing the data would not be sufficient. The function of this App is to extend the simulation system with a use-ability factor.

2 Flood Simulation System

The system that already exists runs in the Cloud. That means it does all the calculation for the client and it is not important to know at physical computer it runs. The system can calculate other simulation by providing it parameters about a certain area. When a simulation is run it will be stored on in the system. With other calls the system return a set of simulation that are present in the current database. Now the important thing to note is that the simulation consists of images that have to be displayed on a map, let's say Google Maps. The flood simulation system preforms a complex simulation with the height map of the area. The height map consists of detailed information about the height of an area. With this data the flow of the water can be calculated and turned in to images.

3 Flood API

The API is a REST api that returns a JSON formatted string. It's a client server system where requests are done by the client and the appropriate data is transferred back to the client.

4 Flood Simulation App

4.1 Platform considerations

As stated in the introduction the app is meant for tablets, but not specific for iOS or Android. How can we maintain this demand on the application. This means that the application is not native programmed but consists of one code-base. There is a technology called PhoneGap that uses HTML5 CSS and Javascript to build the app. In essence the developer build a mobile website and PhoneGap can wrap this in an application. The native web engine is that used to render the mobile website. PhoneGap can result in a huge speed up in development which is a huge advantage. Also by the skills gained by webdevelopment can be re-used. Therefore there is no need to learn a new programming environment used for native apps in Objective C with Xcode for iOS and Java with Eclipse for Android. The advantage that is the learning curve is not high, but it can still feel like a mobile website in stead of a native application. Luckily frameworks exists where native-like elements are created for you.

Another option can be a Titanium Mobile, the Appcelerator. Titanium Mobile can build mobile applications that are in fact native applications. By programming in Javascript and call functions to create native elements. The framework builds native code, in contrary to PhoneGap what is really a mobile webpage in an application. The development for this platform is fast and an application for both iOS and Android are quickly from the ground. But there is also a disadvantage by using Titanium. The applications are big, around 11MB no matter what you did. Not only that, the native elements look good on iOS but no so much on Android. So true cross platform is never reached.

Javascript frameworks

As already explained, PhoneGap only provides the possibility to make an app out of a website. This website can be build in any way the developer likes using webtechnologies supported on the native device. There are frameworks that can speed up the development to make an intuitive application. Two frameworks that can be considered jQuery Mobile and Sencha Touch.

jQuery Mobile This Javascriptwork is build out of one Javascript file and one css file that the developer includes. By giving certain HTML elements a data attribute, which is a HTML5 element, the framework uses this to create views. A page is made by declaring a div adding data-role="page". Such a page can be given a footer like div data-role="footer".

Switching frame one page to the other can be as simple as giving a html anchor an href to an id of a page.

The downside of jQuery mobile is that in order to create an application where views look a like, you have to repeat yourself. Every page gets

```

<div data-role="page" id='bar'>
    <a #href='foo ' data-role="button">switch to foo</a>
</div>

<div data-role='page' id='foo'>
    <a #href='bar ' data-role="button">switch to bar</a>
</div>

```

Figure 1: jQuery Mobile, pages example

the same footer. It can be a burden when changes have to be made in the footer. The conclusion is that jQuery mobile is really easy and the developer does indeed use html, css to create views. But is is not easy to build a modular application.

Sencha Touch Sencha touch is a framework focused on the Model View Controller design pattern. It encourages the developer to use this pattern. The way of developing really differs from jQuery mobile. There where jQuery mobile sets you free to do whatever you like, sencha touch constraints the developer. The difference can be easily explained with an example.

```

Ext.application({
    name: 'foo ',

    launch: function() {
        Ext.create("Ext.tab.Panel", {
            fullscreen: true,
            items: [
                {
                    title: 'bar ',
                    iconCls: 'bar ',
                    html: 'bar '
                }
            ]
        });
    }
});

```

Figure 2: Sencha Touch application example

This example creates an application with the

4.2 App Design