Part 1 : Introduction

In this article the aim is to give an overview of how one might go about constructing an on-line simulation of a calculator using HTML and JavaScript. It is, of course, not the only approach one might take but it's based on the experience I have gained modelling various Sumlock Anita calculators. Hopefully the information is useful to someone who wishes to construct a simulator of a favourite vintage calculator but doesn't know where to start. I'm sure, if you try to build your own model, you'll find improvements to the process over that laid out below. If you do, drop me a line and let me know what they are!

The article will go through a worked example of a simple LCD calculator so that it is illustrated exactly what is done at each stage, and how that makes the final model work. The only tools needed are a text editor and a browser with JavaScript enabled. In order to follow the description below a little experience with programming languages, such as C, is preferable, but only a rough understanding of HTML is needed.

The calculator will be made up of a collage of images, broken down from a source image, with regions defined for input (the keys and switches) and regions defined for output (the display). Data is passed to the javascript when a key or switch region is activated, and the javascript displays results by altering the images used in the display regions. We will discuss how the image is fragmented into these regions, and how the HTML and javascript communicate to pass in user input and output new images.

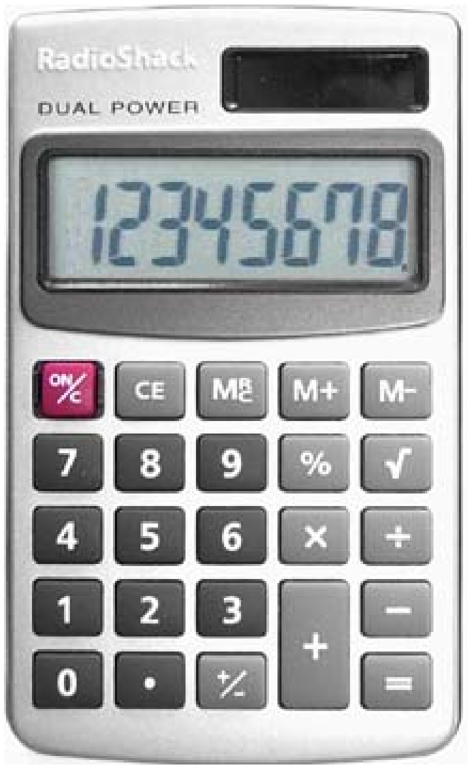
The example

The example we'll use is a very simple Radio Shack, 8 digit LCD calculator with memory, a square root and percent.

One of the keys to a successful simulation is the availability of a manual. Even a simple calculator can have some quirky features, and it is these features that often give the calculator its uniqueness, so it is important to model these if the simulations is meant to document a piece of calculator history. Fortunately the calculator in the example we're using is both sparse in quirky features, and has a manual available on the internet here.

However, having started to model the features of this fine calculator the code started to get bloated with details of the particular features of this model, and this served only to hide the basic structure of the code. So a decision has been made to simplify the code so that only the pertinent features of a simple calculator are modelled. You can always have a go at modifying the code to model the actual calculator features---the manual is available above after all!

The image we'll use of the calculator to be modelled is shown below.



All the basic expected operations will be modelled. So, in general, the basic "<num> <op> <num> =" sequence yields a correct value. Chaining is cut down a little. Something like "2 + 3 +" will still work but "2 + 3 * 6 = " yields 30 not 36---i.e. we won't maintain separate chain registers for adds/subtracts and multiply/divides. Overflow and error handling is much simplified. In these cases "-E-" is displayed, and only an "ON/C" will clear it and initialise the calculator again. No overflow rescaling is done (as per the manual). The manual also states that pressing '=' repeats the last operation, but as repeated pressing of the relavant operation key has the same effect, we won't model this. With these things in mind, we can start constructing a model.

"How to write a calculator simulator"
<- Prev Page Next Page->