Loading from external sources

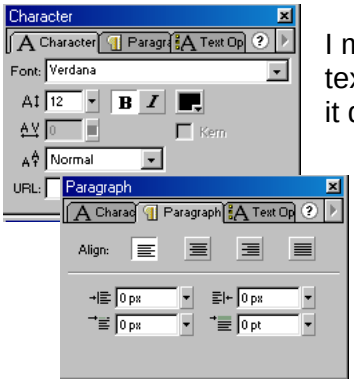
You have learnt some quite complicated things with Flash and it is true that most flash applications and websites don't contain half the actionscriping we have looked at, but to become the highest skilled and therefore highest paid Flasher there has ever been the range of skills you posses will better you in the long run.

Lets look at loading in variables from external sources now to infinitely improve the dynamics within your flash movie and allow for easy updateability.

To do this we create what are technically know as 'Flat File Databases', which basically means a text file full of information that will be pulled into some package or other and read from start to finish. Not a economically sound way of handling data, from a computers point of view but if the file is simple enough and not to long then this is probably the easiest way to get database like activity happening within your flash website.

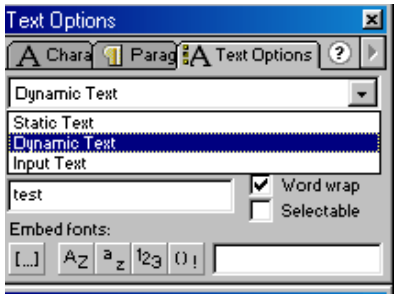
First of all we need to create the flash file. I have created a very simple flash file with a 'Dynamic Text Box' that fills the screen and one key frame that holds the actions to load in the text file.

Text options with Flash 5 and MX

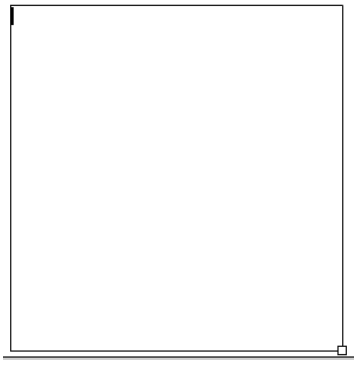


I mentioned 'Dynamic Text Box's' then and up until know text is just something you use to type directly into flash well it does have other uses.

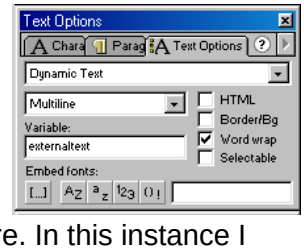
Up until know you have been looking at and probably using these two tabs in the text box options and probably only wondering what 'Text Options' was. On the right here you can see the 'Text Options' tab display. You will see from the drop down box shown that you have 3 main options, Static Text (The one you are most used to using), Dynamic text and Input text (allowing flash to take in information like HTML forms).



This lesson looks at Dynamic text and this allows us to use text pulled from other sources and allows us to control the way text can bee seen to the viewer when the flash file reaches the real world.



Create a text field that can hold a few sentences, like this:



Set your text options to be the same as the box on the right here. In this instance I have set the text to be multi line (allowing multiple lines of text like in html forms). Single Line would be the other option.

I have also ticked ‘Word Wrap’ because I would like the text to return carriage and continue typing whenever the box ends.

HTML – Displays text using only the standard html fonts no matter what you have chosen as well as recognising html tags and using them!
Border/Bg – gives the text box an edge and fill if you want.
Word Wrap – Continues typing on to the next line and so forth.
Selectable – Allows the text to be highlighted and selected to be copy and pasted maybe.

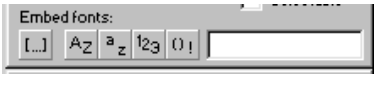
Variable allows us to make all these things shown in the example work. Variable allows us to give the Dynamic and Input text boxes identities, as we gave instances names in previous weeks. By giving these text boxes identities they can be used and called upon using these names and allowing us to load into them from external sources are also done by using these names.

Call your dynamic text box – *externaltext* ***as I have (no spaces!)***

The last section here that you might want to use is ***‘Embed Fonts’*** this allows us to create flash files that use fonts other than what are normally present on a standard machine and know that they will be seen as intended by the outside world.

N.B. It does up the file size so keep this to a minimum for the web!

We can choose to embed the font chosen in a number ways.



Starting from the left of the picture:

[….] – Embeds the whole font family whether we use a character or not, good for forms not good if you know you are only using a select number of characters or cases.
AZ – All uppercase letters in that font family
az – All lowercase letters in that font family
123 – The numbers only
()! – The other characters

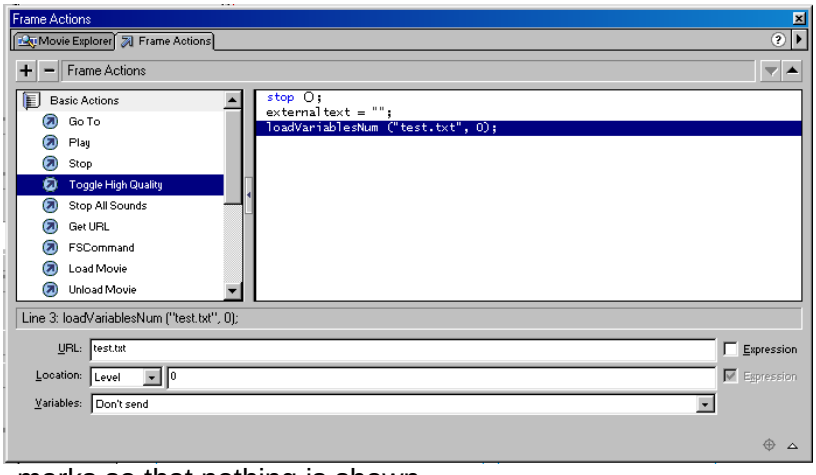
The box allowing you to choose only the letters or characters to store to minimise file size.

You will see that if the embed is used you can vary the effects on the text dynamically and get effects like anti-aliasing or aliasing.

The Actionscripting

Back to the flash file, on either a new layer called actions or on the keyframe that contains the Dynamic text box double click to add actions.

As you can see from the box below the actions are as follows.



Stop (); out of habit to make the flash file stop just in case I wish to add further keyframes later.

Externaltext=""; is the variable name created by choosing Set variable from the actions book. Defined here with nothing in between the speech marks so that nothing is shown.

LoadVariablesNum ("test.txt",0); is the line that loads in the external variables in a text file called 'test.txt' we will create in a minute. You will find LoadVariables in the 'Actions' book on the left hand side of the actionscripting window. As you can see from previous lessons that it is asking for a URL in this case a text file, a location 0 being the base movie and variables (this deals with sending variables not receiving as we are).

Save this and then create text file called 'test.txt' placing it in the same folder as the exported flash files. The text file needs to look like this:

externaltext=this is a tutorial to allow you to create externally loading text files into flash websites and CDROM's. It is really easy you know!&another=this is another variable called another

Save this file called test.txt and then go back to your flash file and either Control > Test Movie or Publish selecting .swf and .html remembering to save the files to same place as the text file you just made.

What you should find is that the text for the variable externaltext is displayed in the box like this: (I have embellished my text slightly)

This is loading the text held under the title 'externaltext' into the dynamic text box of the same name. However, you may be wondering why the next section is missing?

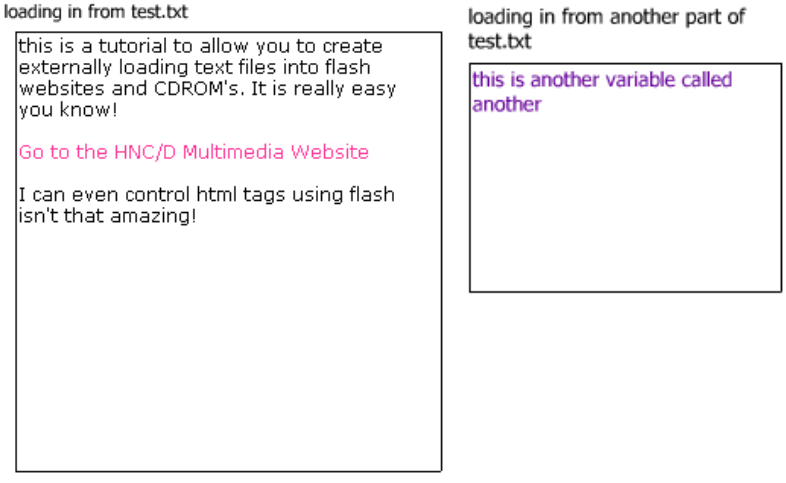
&another=this is another variable called another

Where is this bit?
Well this is text file was written not only as a text file but as a flat file database and this allows us to use it to store more than one variable at once.

By using the '&' I have split the text file into 2 variables one called 'externaltext' and another one called 'another'. Therefore, you can use this method to hold multiple variables and their contents externally and pull them into different places in the same flash file.

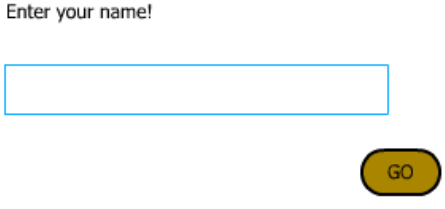
Have a go at creating another Dynamic text box and calling it 'another'. Now Publish or Control > Test Movie. You should now find that the variable 'externaltext' has its contents displayed and now 'another' is showing its contents in another text box!

Amazing! The power there is phenomenal don't you think?



Input text example

Just to show you a possible use of input text here is an example using a name entry, which is stored and then used within a game like scenario.



Say you have a game that as they enter is takes the entrants name. I have placed a simple screen like this in my example.

The Input box shown with an edge around it is called 'name' and the button goes to another keyframe when pressed.

If I enter my name in that middle box I would like it to acknowledge me at the next keyframe and the way to do this is simple.

In the first key frame where the picture above is stored I have a 'stop' command and I have defined 'name' to contain nothing at the moment the flash file loads.

```
Stop();
Name = "";
```

The button command is simple:

```
On (release);{
GotoAndStop (9);
}
```

In frame ‘9’ I have another text box, this time dynamic called ‘hello’ and the keyframe action is as follows:
Stop();
Hello = “Hello and Welcome “ + name;

The stop command you are already familiar with but the next line is setting up another variable called the same as the dynamic text box as to guarantee display.

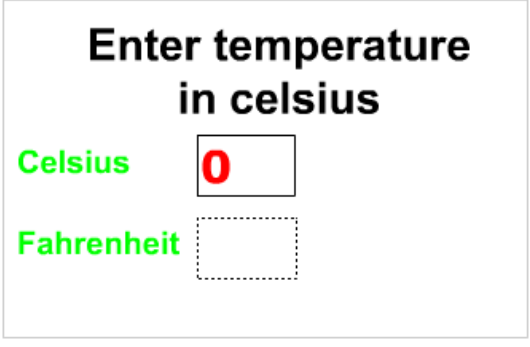
The variable here using what is known as concatenation, the combination of two variables to change the value of the first. In this case the concatenation is simple, the variable called ‘hello’ has some words and the contents of the previous variable combined together to create a sentence of some sort. It will display when run:

Hello and Welcome what ever the contents of ‘name’ happens to be.

Looking at Variables

As in the Input text example previously we can use variables to extend the range of Flash's capabilities further. In the previous example we looked at concatenating words together with variables we are now going to look at applying maths and functionality to variables that contain numbers.

I have created 2 uses of variables using numbers to show you and we will work through those now.



I have created a Fahrenheit calculator that will work out the Fahrenheit from an input Celsius. This is much simpler than it looks!

I have an 'input' text box along side celsius and I have given it the variable name of 'celsius'. I also have a 'dynamic' text box which is not selectable which is called Fahrenheit, they sit within a movie clip so that the contents can be updated immediately by using the OnClipEvent control.

The movie clip is then placed on the timeline and you can add a stop command here if you want to.

```
onClipEvent (load) {
  <not set yet> = "";
}
```

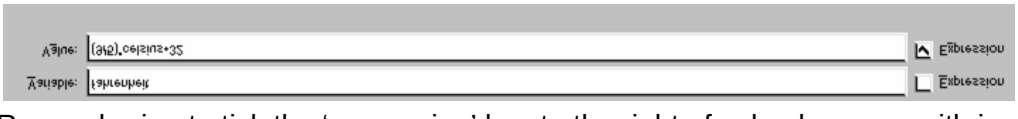
OnClipEvent is another way of activating controls as we did with drag and drop last week (the following example could work just as well with a button to activate the calculation).

The next line is waiting for us to create a variable.

I am going to give the variable the same name as the 'Dynamic' text box so it will display the results in it (as before) and give it some maths to do.

fahrenheit = (9/5) * celsius + 32;

This is the maths, I know I can work out the Fahrenheit by doing 9 divided by 5 times by the Celsius and then adding 32 and the calculation is written like above.



Remembering to tick the 'expression' box to the right of value because with is an expression to be carried out not to be displayed as is. If you didn't tick the 'expression' box the value would be:

"fahrenheit = (9/5) * celsius + 32"; - which is very different and would display this calculation in the Fahrenheit box rather than the sum of the calculation.

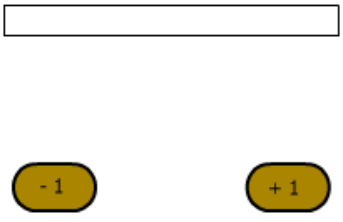
Try Control > Test Movie see what happens.

As you enter a figure into the Celsius box the Fahrenheit box changes automatically.

Scoring using Variables

Scoring or indeed counting is becoming a regular activity in Flash and certainly as it becomes more for gaming so lets look at a simple counting mechanism that can add to our game and maybe create a score tally.

I have created a Movie clip with a ‘Dynamic’ text box called ‘count’ and two buttons, one will add and one will subtract from the total displayed in the text box.



What I would like to happen is, when I click on the ‘-1’ button the total displayed is minus 1 and vice versa with the ‘+1’ button.

When the total being displayed in the dynamic text box reaches a certain number I would like Flash to do something (the number I would like it to reach is 6).

The first thing to do is to set up the actions on the buttons inside the movie clip.

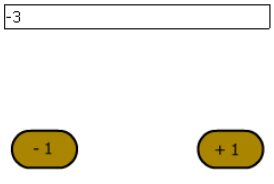
```
on (press) {
count = count-1;
}

on (press) {
count = count+1;
}
```

These are the actions to be played on the buttons, hopefully it is obvious which action goes on which button!

The SetVariable action has been called again and it is setting up the variable with the same name as the dynamic text box as to display the count and is readdressing itself every time a button is pressed. It works by getting its own value and then adding 1 or taking 1 to produce a new value.

If you Try Control > Test Movie now you should find that you can add and take a way numbers incrementing and decrementing by one to your hearts content.



Now we need to have this score doing something useful.

Come back to the main timeline and add actions to the movie clip as we did earlier.

Here is what the action script is going to look like:

```
onClipEvent (enterFrame) {
total = count;
if (total ==6) {
loadMovieNum ("welldone.swf", 0);
}
}
```

Some of this you will be familiar with and other bits need explaining.

```
onClipEvent (enterFrame) {
total = count;
if (total ==6) {
loadMovieNum ("welldone.swf", 0);
}
}
```

What I propose with happen is that when the count = 6 a new movie will load in to replay the existing one and congratulate the user, and it is done like this:

total = count; - I have declared a new variable and because the OnClipEvent happens every time I enter the frame (which I do constantly, because there is only one frame and flash loops) it is constantly updated, I have called it 'total'.

The reason why we declare a new variable to hold the contents of another variable as we are in this case is so we can work with it from a fixed point. Also because 'count' is simply incremental and displaying in a text box flash doesn't actually know its contents to use, only to display so we create a new variable that holds the 'count' contents.

if (total ==6) {loadMovieNum ("welldone.swf", 0);} – Next we have an 'if' statement again calling upon the contents of 'total' to be tested. We say if total is equal to 6 then do something otherwise let the user carry on incrementing and decrementing.

If 'total' does equal 6 load a new movie called 'welldone.swf' into the base level '0' and therefore replace the counting movie.

Try Control > Test Movie

You should find that the minute you reach '6' the new movie loads.

Other options

You might want to try adding to this 'if' statement by using the 'OR' operator within the 'if' statement.

Try

```
if ((total == 6) || (total == -6)) {
loadMovieNum ("welldone.swf", 0);
}
```

This is the 'OR' operator in action and it says to Flash. If the contents of total is equal to 6 OR the contents of total is equal to –6 then go and load in that movie.

You can also try these other Operators:

< - Less than
eg. total < 6 – means the contents of total less than 6

> - Greater than
eg. total > 6 – means the contents of total greater than 6

>= - Greater than or equal to
eg. total >= 6 – means the contents of total greater than or equal to 6

<= - Less than or equal to
eg. total <= 6 – means the contents of total less than or equal to 6

<> - Not equal to
eg. total <> 6 – means the contents of total is not equal to 6

& - Logical AND

! – Logical NOT

|| – Logical OR