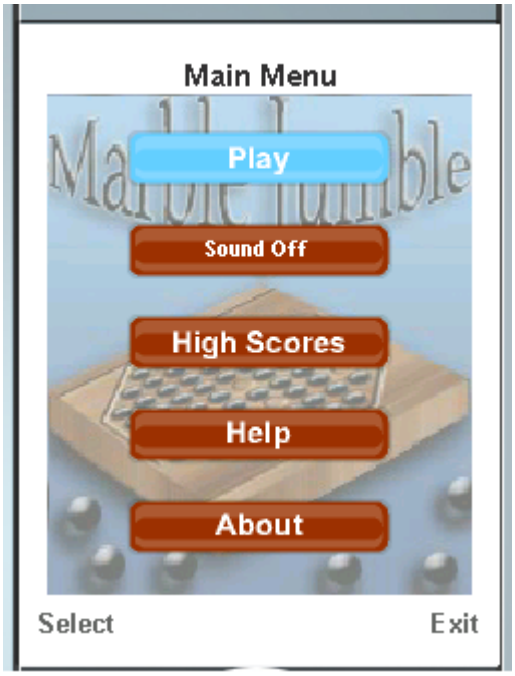
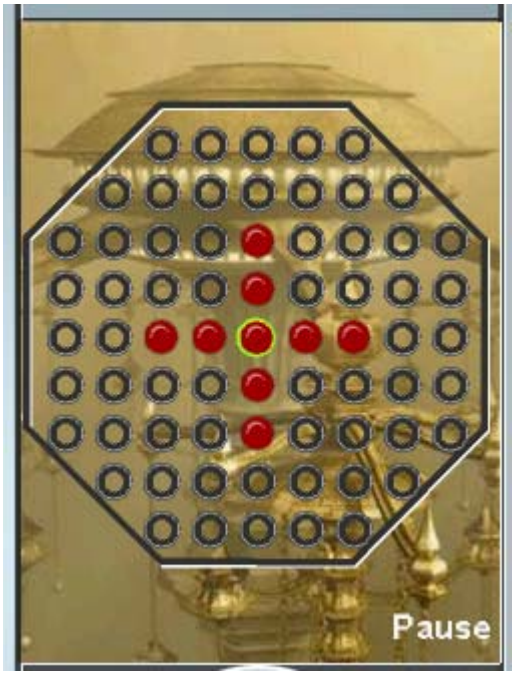
L&T Infotech Ports App to Adobe Mobile Client on Brew Mobile Platform

One of the brightest promises of the Brew Mobile Platform (Brew MP) is that it enables developers to integrate Flash content directly into mobile applications for a richer user experience. Developer L&T Infotech Ltd. recently did just that, converting its J2ME-based game application, Marble Jumble, to a Brew MP application using **Brew MP SDK 1.0, ActionScript 1.0** and **Adobe Mobile Client (AMC) 1.1**.

Marble Jumble is a puzzle board game originally developed for J2ME and Flash Lite 2.1. L&T chose the app for this proof of concept on Brew MP because it is interactive, featuring user input, levels of play, key events and low-resolution graphics.





Preparation and Impact Analysis

L&T engineers spent about a week studying Brew MP and Adobe Mobile Client® (AMC), acquainting themselves with the tools and workflow. They used the Flash Application Primer and sample code from the Brew MP Developer Network to set up the necessary development environment. They also used the workflow overview in the Flash Application Primer to understand the relationships among Adobe® Flash® CS3/CS4, the Flash plug-in for Brew MP, the Brew MP AppCreator, and Simulator.

Next, they conducted an impact analysis to identify areas of change in converting their code to Flash and Brew MP:

- Development Tools – The engineers used Adobe Flash Player 6 and AMC 1.1.
- Graphics – Marble Jumble uses PNG images at each stage of play, which L&T converted to movie clips for use with AMC 1.1. They added them to the project library and called each object as a movie clip file to the screen. Also, they took advantage of the porting project to increase target screen size and image resolution from 176 x 220 up to 240 x 320 (QVGA).

- Asset Container – In AMC, Marble Jumble continuously renders UI objects in timeline frames. When changing the objects on the screen, L&T called the rendering method at each step (e.g., one timeline frame for the menu, another for the game screen, and so on.).
- Programming Language – For the game logic and the code required to move from one screen to another, L&T wrote Marble Jumble for Flash/Brew MP using Adobe ActionScript® 1.0.

The impact analysis showed that the format of the game's sound assets – MIDI – would suffice for both platforms.

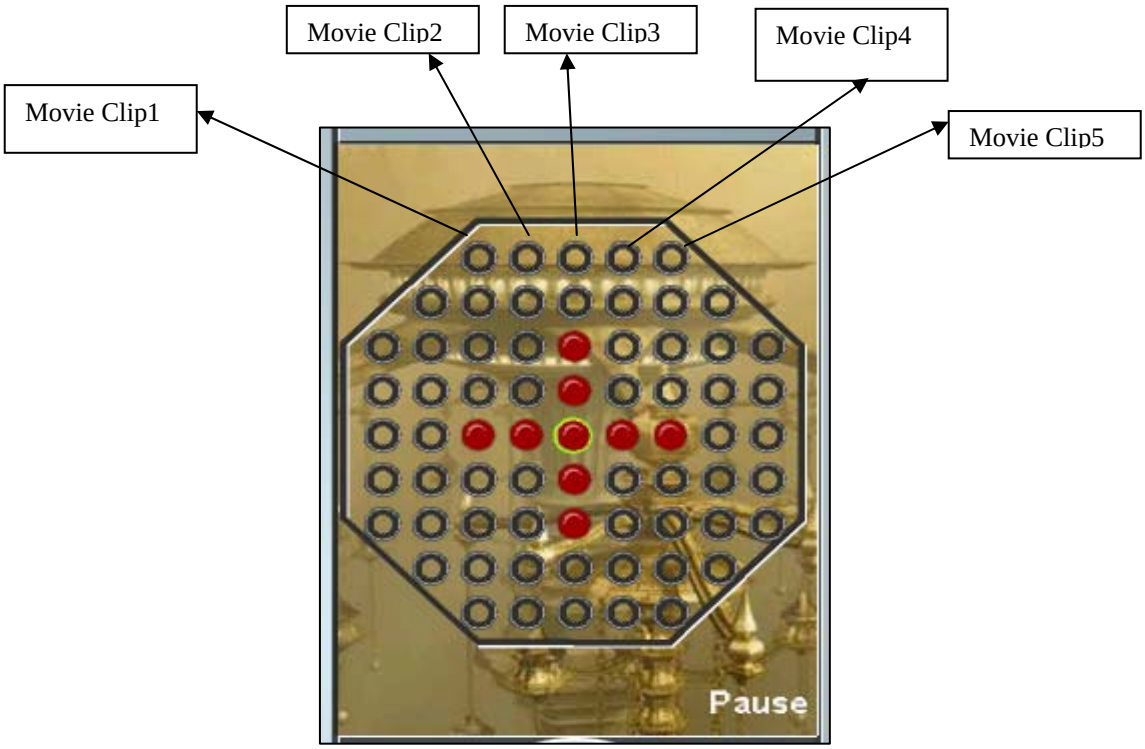
One Person-Month and 400 Lines of Code

Once the engineers had used Flash to create the UI components and assets for each layer, they wrote ActionScript code to link the components and assets together in an action layer, as well as to implement key listeners for up/down/left/right/fire:

```
var myListener = new Object();
Key.addListener(myListener);
myListener.onKeyDown = function() {
        if (Key.getCode() == Key.UP || Key.getCode() == 50) {
                //Key up functionality
        }
        if (Key.getCode() == Key.DOWN || Key.getCode() == 56) {
                //Key down functionality
        }
        if (Key.getCode() == Key. LEFT|| Key.getCode() == 52) {
                //Key Left functionality
        }
        if (Key.getCode() == Key. RIGHT|| Key.getCode() == 54) {
                //Key Right functionality
        }
if (Key.getCode() == Key. ENTER|| Key.getCode() == 53) {
        }
}
```

Similarly, they used the attachMovie functions to add the movie clips of the UI objects to the stage:

```
        attachMovie(arg1,arg2,arg3)
        //where arg1 = Movie ID, arg2 = Movie Reference name and arg3 = color depth for the movie object
```



Following is sample ActionScript code from the game (offered as a reference for syntax only):

```
//To attach a movie to the stage
attachMovie("game_Over","game_Over",this.getNextHighestDepth());


// To set the new color depth, as the new object's color depth should be
//higher than that of the previous object to make it visible on the stage
game_Over.swapDepths(q+200);

// To set the coordinates of the object on the stage
game_Over._x = (Stage.width/2)-(game_Over._width)/2;
game_Over._y = (Stage.height/2)-(game_Over._height)/2;
```

The game logic in the action layer took up 100-200 lines of code. Other frames required only enough code for program flow among screens. In all, Marble Jumble represents about 400 lines of ActionScript code.

The engineers implemented continuous program flow by placing their code in the key frames. Originally, a separate ActionScript file accessed the array variables, and this worked fine in ActionScript 2.1. In the conversion to AMC 1.1, which uses Action script 1.0, the engineers transferred these array variables to the ActionScript file containing key frame logic in order to avoid errors in accessing array variables declared in the separate ActionScript file.

After their initial study of Brew MP and AMC, the engineers spent about one person-month spread over two weeks converting the app, including Impact Analysis, UI creation, coding, testing on Brew MP, fixing bugs and releasing.

Tips for Other Developers

For porting to Flash/Brew MP, L&T recommends an app like Marble Jumble, which relies more on game logic rather than on animation and rich graphics.

The engineers also emphasize the Impact Analysis as the important first step for any developer, characterizing it as the only way to know how much change is involved in converting the code. Which types of assets do the target platforms (AMC, Flash Lite) support and how many must the developer convert? What will be the final output? How will the developer implement global variables and arrays and listen for events? Is the reference code object-oriented, requiring changes in the timeline frame?

L&T notes that the Brew MP documentation offers a good overview and a good set of starter code samples on linking Brew MP to Flash using AMC. In particular, the Flash Application Primer helped them understand the model for building and publishing an executable. They look forward to seeing more sample code from other developers to broaden the knowledgebase around porting apps to Flash/Brew MP, especially on handling UI-related events like sound.

➔YOU TOO CAN DO THIS! Download and install the Brew MP SDK. Visit the Library in the Brew MP Developer Network for the Flash Application Primer, ActionScript Language Reference and other documents and code samples in the Flash language.

Code samples and screenshots courtesy of L&T Infotech Ltd.

###