## Taking Control

### The final part of this keyboard programming tutorial, by Damian Walker

At the end of last month's tutorial, we had a fully working keyboard routine that allowed the player to move the ball smoothly, without key repeat delay, and in any of eight directions. All that remains now is to remove the annoying key click. This can be done with three alterations. Firstly, at the top of the program:

```
INCLUDE "System.oxh"
```

Secondly, at the top of the *MoveBall* procedure, just after all the *LOCAL* variable declarations:

```
    SetKeyClickEnabled:(0)
```

And finally, at the end of the *MoveBall* procedure, just before the *ENDP* line:

```
    SetKeyClickEnabled:(1)
```

The use of OPXs was covered in the *Animating OPL* tutorial. The System OPX contains the procedure to enable or disable the key click, *SetKeyClickEnabled*. This is easy to use: simply pass it a 0 to disable the key click and a 1 to enable it. We use it this way to disable the key click before we start moving the ball, and re-enable it again when we're done.

Unfortunately there is no way, without using a third-party OPX, to check whether the key click was enabled in the first place, so as it stands, people who don't want the key click end up having it switched on anyway. You may therefore decide you don't want this functionality in your game, leaving it to the player instead to switch off the key click in the Control Panel if they so desire.

That ends the practical side of this tutorial. Before taking my leave of the subject I'd like to look at a few ways the techniques can be used and varied. First of all, I'd say that the use of KEY and GET should be avoided on EPOC32 machines, because of their inability to deal with system events. I'm mindful of the *DO UNTIL KEY=27* still in the *Bouncer* procedure, and you might like to think about replacing this with something better.

The use of *GETEVENTA32* and key codes, as showed in the third part of the tutorial, is sufficient for a lot of games, and for some games it is more appropriate than the use of scan codes as in the most recent part. In fact it is only really for arcade games that one would need to overcome the key repeat delay problem or check for multiple keys at once. For puzzle games and strategy games the simpler method is probably better suited. In fact, if your game doesn't involve timers, or enemies that move while you are idle, you could even investigate the use of *GETEVENT32* instead, which waits indefinitely for an event and therefore doesn't need multiple calls to *IOYIELD*.

There are many other ways that these routines can be used. The most obvious is to combine stylus and keyboard control, allowing the player to use either method (or both). The OPL manual explains very well how to check for events with the stylus. Other things you might do include restricting movement: not all games benefit from diagonal movement, and some may even be restricted to left and right. But even in these cases, you'll might need to check for multiple keys to implement jumping or firing.

These are just some ideas to get you thinking about good use of keyboard control in your games. I'd be interested in any questions or comments you have: just send them to the usual email address.

EPOC
# ENTERTAINER
Issue 12: November 2008
Edited by Damian Walker
*This month... Mahjongg*

This month I am pleased to be able to announce a new game for EPOC32, which you can see below on this page. To my knowledge this is the first new game of the year. With the number of people who have sent feedback for the programming tutorial articles, are we likely to see any more? If you've been following the tutorials then you should now be able to move sprites around the screen and scan the keyboard—the principal ingredients of a game.

Talking of programming tutorials, our latest one is coming to an end this month. If you've enjoyed the programming tutorials so far then don't worry, there's another one lined up for next month.

Finally, there are two game reviews this month, including a very good mah jongg game. No, not the solitaire puzzle, but the proper traditional Chinese game that you can play against other people. The other review is of Worm, I think the first snake game that's been reviewed in *EPOC Entertainer*.

I now have a little prize ready for the start of our first competition, which I hope to announce over the next couple of months. The nature of the competition has yet to be decided. It might be a programming competition, or a level or scenario design competition for one of the many games that support this feature. I'd be interested over the next month in hearing how many people would be interested in entering, and which type of competition they'd prefer. Answers and comments are welcome at the usual email address.

***entertainer@snigfarp.karoo.co.uk***

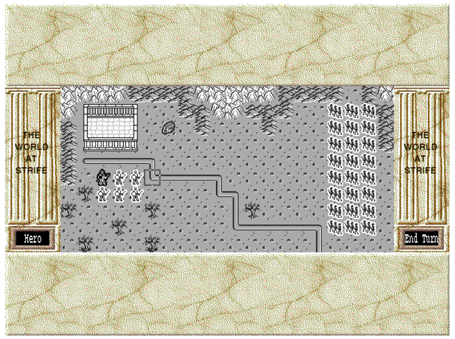
## News: a New Game Release

### Damian Walker introduces The World at Strife, a game released for EPOC32 in the past month.

A new game has been released last month: The World at Strife, a game of my own. This is the last game in the Strife series. These are strategy games in which you have to defeat the enemy or seize strategic positions on a series of battlefields.

There are four supplied campaigns, ranging from ancient mythology to science fiction, or you can create your own campaign using the built in campaign editor. The appearance of the graphics can be altered to suit the era of your campaign.

The World at Strife can be downloaded from my own web site, at the URL shown below right. It is intended that new campaigns by me and others will be added to the site as time goes on.



| | |
|---|---|
| By | Damian Walker |
| URL | psion.snigfarp.karoo.net |

## The Real Mah Jongg

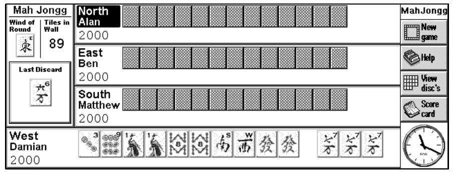
### A review of Adrian Collister's Mah Jong game, by Damian Walker.

Most computer games players know of Mah Jongg as a solitaire tile game, in which you remove pairs of matching tiles from a pile in the hope of clearing all the tiles from the table. But those interested in traditional games will know that these computer solitaire games are not Mah Jongg, because Mah Jongg is actually a game for four players.

The game plays like many card games familiar to the western world. The tiles are split into in suits, and each tile, as with cards, has a rank. The exceptions are tiles which represent seasons, flowers and dragons. The object of the game is to create a hand of fourteen cards termed Mah Jongg, made up of an identical pair of tiles, plus other sets of three or four tiles. These sets have names link *pong, kung* and *chow*, representing three or four identical tiles, or a sequence of three tiles in the same suit. The first player to declare Mah Jongg wins the round. A full game consists of four rounds.

It is this four-player game that Adrian Collister has implemented on EPOC32, under the straightforward title Mah Jongg. The version I tested is the unregistered “demo” version, which allows play of the first round of four. This runs on all EPOC32 machines apart from the Osaris, whose screen is too small to accommodate the necessary graphics. On colour machines Mah Jongg runs in monochrome, though it does scale up to fill the larger screen of the Series 7 and netBook (and presumably the Geofox too).

Leaving aside compatibility, I found the program to be generally well-written. It conforms to the standards of EPOC32, responding to system events, and using the familiar menus, toolbar and dialog boxes. It runs along at a decent speed on everything including the Series 5 “classic”, and it takes up a very small amount of memory—both in disk space and when running. As for reliability,



I've not come across any problems.

Playability is the most important aspect of this game. While we can't credit the author for making the rules of Mah Jongg entertaining, we can commend his good taste in choosing to implement this particular game game. Like many traditional games, Mah Jongg is good fun and can be played again and again. It is quite a quick, light game, especially in the unregistered version which allows only the first hand to be played.

Apart from this limitation, the rules of Mah Jongg are followed exactly. The three computer opponents have so far managed to beat me each time, so the artificial intelligence is adequate at least to give an introduction to the game. As time goes on I may find that the computer players are even better than that, or that my ability to learn the game's tactics is worse than I expected.

Mah Jongg has good presentation, though limited to monochrome even on colour machines. The tiles are clear, and well-drawn, with a 3D shadow effect that prevents the display looking flat. While the display is therefore not exciting, it is perfectly adequate for the game. Sound is limited, but appropriate, with the tiles making an attractive clicking noise when laid. Sound can be switched off if you wish.

This version of Mah Jongg provides a refreshing change from the usual solitaire tile-matching game available on EPOC32 and other platforms. For people who like traditional games, I fully recommend it.

| | |
|---|---|
| By | Adrian Collister |
| URL | pages.eidosnet.co.uk/~a.collister |
| Licence | Shareware |
| Compatibility | Revo, 5/5mx, Geofox, 7/netBook |
| Rating | ☆☆☆☆ |

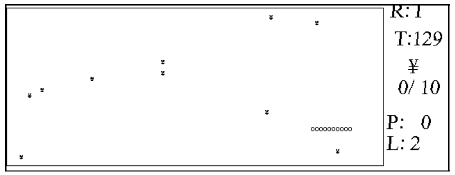
## Turning the Worm

### Damian Walker reviews Worm, a snake game by Ferenc Sarlós.

The snake game is one of the more “traditional” genres of video game, considered with the likes of pong and asteroids. The player guides a snake around a largely blank playing area, where small pieces of food are waiting to be gobbled up. Each time the snake takes a snack, points are scored, and the snake grows in length, making the game harder. When all the food is eaten, one progresses to the next screen, which may have more food, or in some cases, obstacles. Some games present the food one morsel at a time, while others lay out a feast at the start of the game.

Worm by Ferenc Sarlós is one of several snake games available on EPOC32. It runs properly only on the Series 5. The Osaris and Revo can't run the game, as their screens are too small. And while the game runs in letter box mode on the Series 7, it runs a little too fast on that machine—indeed, on any machine faster than a Series 5 “classic”. Of other EPOC machines, only on the Geofox might you be able to run Worm, as with the Series 7 in letter box mode.

Worm implements the traditional game with a few differences. Where in some



versions of the game you have to eat all the food, in Worm you have to eat ten pieces, and then an exit appears. You can eat a few more pieces, or head straight to the exit. Worm appears to lack the walls and other obstacles that some snake games have.

The graphics in Worm are extremely lacking, as they are made up of characters instead of real graphics. There is no reason why this should be the case; high-resolution graphics are very easy on EPOC32, and text characters are not an attractive substitute. In addition to this, the characters are that small that it is difficult to distinguish between, for example, pieces of food and the exit when it appears. The presentation of the score table is quite unattractive too, with details such as the room number, score and lives left scattered seemingly randomly.

Worm features no sound, and a rudimentary user interface. While there is a menu available at the title screen, once in the game, there is neither menu nor any indication of what keys are used. The worm is obviously guided using the cursor keys, but one might wish to access other features such as pausing the game or finishing early.

I wouldn't recommend Worm. For fans of snake games, there are a number of others available for EPOC32, which I hope to review in future editions of *EPOC Entertainer*.

| | |
|---|---|
| By | Ferenc Sarlós |
| URL | psion.snigfarp.karoo.net |
| Licence | Freeware |
| Compatibility | S5 |
| Rating | ☆☆ |