Taking Control

This month's article, by Damian Walker, discusses keyboard and event control.

In last month's tutorial we introduced code to scan the keyboard using *KEY*, but also identified some limitations with that method, namely, the fact that non-key events can't be scanned for. For this there is a statement *GETEVENTA32*, which is a little more complicated to use, but allows you to scan for system events and pen input, as well as key presses. It is *GETEVENTA32* that we'll be using this month.

Firstly you need to remove all the key scanning code that you added last month, returning the program to the state it was in at the start of the first instalment. Then you need to add the following to the top of the *MoveBall* procedure:

```
LOCAL status%,ev&(16)
```

Add the following after the line starting *progbegin&=*:

```
GETEVENTA32 status%,ev&()
```

And the following after *start&=DTNow&:*

```
movex%=0
movey%=0
IOYIELD
WHILE status%<>-46
  IF ev&(1)=4105
    movey%=-4
  ELSEIF ev&(1)=4106
    movey%=4
  ELSEIF ev&(1)=4104
    movex%=4
  ELSEIF ev&(1)=4103
    movex%=-4
  ENDIF
  GETEVENTA32 status%,ev&()
  IOYIELD
ENDWH
```

The function *GETEVENTA32* differs very much from *KEY* in the way it works, so it isn't just a drop-in replacement. The way it operates is this. Firstly, a call to *GETEVENTA32* is made to set up event checking. The *status%* variable is initialised at -46, and stays that way until an event occurs. IOYIELD is called repeatedly to allow events to occur; during ordinary processing events are locked out. Any event sets *status%* to 0, at which point the elements of the *ev&()* array can be checked for the nature of the event. Once the event has been processed, *GETEVENTA32* must be called again to check for a subsequent event. Notice from the IF section in the code above that the key codes returned by *GETEVENTA32* differ from those returned by *KEY*.

Why do we use a *WHILE* loop above, rather than an *IF* as with *KEY*, to check for events? This is because a key press generates three events in a row: a key down event, a key code event, and a key up event. Given that our outer loop has a 1/8 second pause between iterations, we really need to process all three events together—even though we're ignoring two of them—otherwise a noticeable time lag is introduced. So the WHILE loop continues to process as many events as are waiting to be dealt with. This is why there is an extra *IOYIELD* in the loop. You might want to try replacing the WHILE with an IF to see this time lag in action.

There are still problems with this approach. Key events conform to the system's key delay and repeat speeds. This is great for typing, but for movement in games, the delay after the sprite starts moving can be a nuisance. And wouldn't it be nice if we could check for multiple key presses for diagonal movement? Next month's instalment will deal with these issues.

EPOC ENTERTAINER

Issue 10: September 2008

Edited by Damian Walker

This month... Castle III

EPOC Entertainer has now reached double figures, with issue 10 in your hands or on your screen. In this issue there are two reviews: *Castle III* by Darren Prescott, and *Scorched Cannons* by Global Posse. Readers who were around in the earlier days of PCs might remember the Scorched Earth game which is an obvious inspiration to Scorched Cannons.

There also continues our series on programming for keyboard control, this month showing you how to scan the keyboard in a more "professional" way, allowing for pointer and system events if you wish.

As always, I am interested in feedback for past, current and future issues of *EPOC Entertainer*. If you have any comments, ideas, wish lists, or perhaps even contributions to make, then do get in touch with me at the address below.

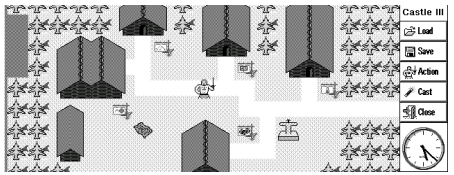
entertainer@snigfarp.karoo.co.uk

King of the Castle

Damian Walker takes a look at Darren Prescott's shareware role-playing adventure game, Castle III.

Compared with other types of game, such as arcade games or board games, the role-playing game is underrepresented on EPOC32. One of the best of these games is Castle III, by Darren Prescott. This series of games started life on the EPOC16 platform, and versions 1 and 2 of the game are available only on that system. Castle III on the other hand is exclusive to EPOC32.

This is a traditional style computer role-playing game. After choosing your character from three types (fighter, mage or thief), you are taken to a village. Here you can equip your character with weaponry and other items, although your choice at the start of the game is limited to a wooden club and a pair of boots, since you have little gold to buy things with. When you're ready you progress to the dungeon, where there is a variety of monsters to fight, and treasure to collect. The view is a top-down map view which scrolls as you move around the dungeon, keeping your character at the centre of the screen.



Game play is quite addictive. It certainly had me hooked, and I keep having to tear myself away from the game to work on this review. There are a few niggles, though. Sudden death

by invisible traps, without warning, doesn't add to the fun for me, and I don't like the fact that you can become easily trapped by taking a wrong turning, leaving you wandering in a futile manner around the dungeon. This is probably more a fault of the genre than with this particular game, and it certainly doesn't stop me wanting to play again. There are plenty of levels to keep a player amused for quite some time, too.

One omission I found is that, while many aspects of the game are well-described by the help system or by the various dialogue boxes, there is little information anywhere on the effect of the various weapons. Common sense told me that the axe I'd just picked up is better than the club I started with, but only by the price of the weapons could I work out if it was worth swapping my axe for a sword at the next weapon shop.

The graphics are simplistic and in some cases not very well drawn. Many items are recognisable, particularly chests, healing potions and locations on the map. Others are more difficult to see, however. None of the shop signs are recognisable without a key, being rather too small. While all of the monsters are recognisable as monsters, it isn't always possible to recognise what a monster is before fighting one for the first time. But on the positive side, it is possible on all machines to see where you are, and where the various items are, without much difficulty.

The user interface is good, with the standard EPOC menus and tool bar making things easy to locate. For this type of game the keyboard is the most convenient method of moving around the dungeon, and my only criticism is that many of Castle III's dialogue boxes lack keyboard shortcuts, forcing you to reach for the stylus more often than should be necessary.

One thing that always impresses me is compatibility across the whole EPOC32 range, and Castle III certainly has that. It runs on anything from the Osaris and Revo to the Series 7 and Netbook. The game adjusts itself to any reasonable size of screen, so the Osaris screen was supported without any direct effort on the part of the author.

The game supports EPOC in one important way, that of being obedient to a request to close from the system screen, so Castle III shouldn't get in the way of making backups. It doesn't support EPOC conventions for documents, however, so saved games can't be automatically loaded from the system screen as with some other games. Instead, you have to launch Castle III from the Extras bar and load your game from within the program.

I found one issue with reliability. Selecting Help from the Actions dialogue box gave me a *Not Found* error. Presumably some file is missing from the distributed version.

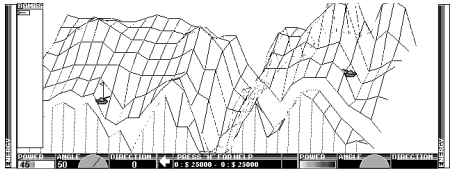
Performance is good. On the slowest machines you can move about the dungeon as fast as you could reasonably wish, although drawing a new dungeon level does cause a delay on all machines. This only serves to increase anticipation! It does take up a fair amount of memory though. While with its simple graphics it takes only 750 kilobytes of disk space on a colour machine, when running it consumes a further 1.8 megabytes.

For those who like a good old-fashioned dungeon romp, this is probably a good and entertaining choice of game for the EPOC32 platform. Those used to similar games on the PC and on consoles might not like the simple graphics, but the game play is there, and I recommend giving it a try.

By	Darren Prescott/Neuon
URL	www.skankee.com/neuon/
Licence	Shareware
Compatibility	Osaris Revo 5/5mx Geofox 7 netBook
Rating	☆☆☆☆

What a Scorcher!

Damian Walker reviews Global Posse's artillery game, Scorched Cannons.



One type of game that has been very popular over the years is the artillery game. A player controls the elevation and power of a gun, and in one turn after another fires shots at an enemy, each time refining the aim of his weapon. Variations pit whole teams of units one against another, support different weapons, or introduce 3-dimensional terrain. The timelessness of these games may rest on the fact that graphically, they are very adaptable: it is possible to use anything from stunning 3D graphics to a simple text interface.

EPOC32 is well served by these artillery games. *Wrippers* has been examined previously, and the game featured in this review is *Scorched Cannons*. A two-player game, Scorched Cannons runs on the Series 5 and 5mx, and pits two tanks against each other in a 3-dimensional landscape. At the beginning of a game, each player can equip his tank with a choice of weapons, before proceeding to the battlefield. At the end of the battle, extra money is awarded to the players, who can re-equip their tanks and fight again.

The biggest positive about this game is its graphics. From the title screen to the battlefield, everything is well-drawn, with only the weapon purchase screen looking a bit spartan. The 3-dimensional mountainous landscape is drawn in clear wire-frame graphics, while the control panel is attractively drawn and shaded. The game runs at a good speed too.

There is a range of options at the beginning of the game. You can set the economics of the game: how much money is available to equip tanks for each battle, and how much is given as a bonus when a player wins. You can set the size of the battlefield. You can also control the magnification and viewing angle, but sadly only at the title screen, not during the game.

As with many artillery games, the action is quite addictive. However, Scorched Cannons loses out by not having a computer player, so if you're on your own, it's not quite so entertaining.

The game scores quite highly in its memory footprint too. For all its pretty graphics, the game occupies only a couple of hundred kilobytes on drive C: and about as much again when running. Unfortunately, the game fails to work when installed to drive D: so it's a good job it's not too big.

There are a couple of issues with general reliability of the game. It doesn't crash, but there are certain options and dialogues that don't work as they should. Sound doesn't work at all, though perhaps that's because there's an incomplete archive floating around the net. And in the final "Play again?" dialogue box, *No* appears not to be a valid option!

There are some serious down sides to the game too. The first is compatibility: the game supports only the Series 5, 5mx and MC218. It will run in letterbox mode on the Series 7 and Geofox, though. The user interface is terrible, in spite of all its prettiness. Once past the title screen, the menus and standard EPOC keys disappear, and if you've not read the dialogue-based help at the title screen, you're pretty stuck when you get to the battlefield.

But despite these problems, Scorched Cannons is quite an interesting game for two players. Its rating here is affected mainly by the poor interface and lack of a computer opponent.

By	Global Posse
URL	psion.snigfarp.karoo.net
Licence	Shareware
Compatibility	5/5mx
Rating	☆☆