Portfolio: Ekans

Harry Mumford-Turner & Kristian Shaw

Game concept

Game overview:

The player controls a group of connected blocks called ‘Ekans’ which they direct around the screen with the goal of colliding with a single block called a ‘Gem’. The movement of the Ekans is much like a real snake, where the player moves the head and the body follows the exact path of the head. On the collision with the head of the Ekans and the Gem, a block is added to the Ekans tail, (so the end of the body) the current Gem is destroyed and a new Gem is created which is again randomly placed on the screen. If the Ekans collides with itself or the edge of the screen the player loses.

Speciﬁcation:

Functional requirements

0.1 The System must produce a menu, to guide the player to the main game

0.1.1 The system must produce enough text for a difficulty setting, quit option and play option.

0.1.2 The system must lead the player out of the game with a retry option and a quit option

1.1 The System must enable the client to control a group of connected blocks called ‘Ekans’

1.1.1 The Ekans will move on a grid where all the elements are locked onto

1.1.1.1 The grid will end at the border and each block will be 25 pixels by 25 with the board size of 500 pixels by 500.

1.2 The System must enable the client to move these blocks with the goal of colliding with a single block called a ‘Gem’.

1.2.1 The movement of the Ekans will be much like a real snake, where the client moves the head and the body follows the exact path of the head. Ekans will be made up of equal segments that follow each other through movement, each occupying a square on an invisible grid.

1.2.2 Each tic of the fps clock the Ekans will be redrawn with updated positions, based on all the elements (segments) in the array (Ekans)

1.2.3 The system will read user input from the directional keys

1.2.4 The direction of the Ekans will only be changed if the player has selected a direction with a 90° difference to the current direction.

1.2.5 If a direction key pressed is NOT the opposite direction Ekans then change to that corresponding direction

1.2.6 The system should only allow Ekans to move either up, down, left or right. Not diagonally.

1.2.7 The system should only take the first user input, e.g. Pressing up then left, within the clock tic time still only moves up not left

1.2.8 A function will shift the tail of the array working backwards through the array making each one the one in front until index 1

1.2.9 The gems will also take up a square of the same size to make alignment more visible to the player.

1.2.10 When the player collides with a Gem, another segment is added to Ekans by appending to the array of x’s and y’s.

1.2.11 The system must enable the Ekans to collide with a Gem

1.2.11.1 A block will be added to the Ekans tail, (so the end of the body)

1.2.11.2 The current Gem will be destroyed and a new Gem will be created which will be randomly placed on the screen.

1.2.11.2.1 A random gem function will create a random pair of coordinates forced to a multiple of 25 to keep it aligned with the grid. A random position will be created and if the new position overlaps a current Ekans segment, it will generate another.

1.2.11.3 A function will check to see if a pair of coordinates collide, if the gem coordinates collide with the Ekans head coordinates then the current gem is not displayed and another is placed at a random coordinate that is not the Ekans.

1.2.12 The system will check to see if the Ekans collides with itself or the edge of the screen

1.2.13 The player will lose if the Ekans collides with itself or the edge of the screen

1.2.13.1 If the coordinates of the Ekans head collides with the edge block or any of its body parts then break from the game loop

1.3 The System must enable the client to move these blocks with the bonus goal of colliding with a single block called a 'Super Gem'.

1.3.1 The system will spawn Super Gems for every 5 gems the player collects

1.3.1.1 A modulus of 5 will be used when calculating this

1.3.2 The Super Gems will be given random coordinates every time one spawns using the random coordinate function (see 1.2.11.2.1)

1.3.2.1 These will be locked to the same grid as the Ekans and will be one block in size.

1.3.2.2 These will not spawn on the currently spawned Gem or the Ekans

1.3.3 Only a single Gem will be on the screen at any given time and when one is destroyed another is created.

1.3.4 Super Gems are worth 10 points each, times the number of a points for a Gem.

1.3.5 Super Gems will disappear after 5 seconds of spawning.

1.3.6 Super Gems will not increase the size of the Ekans.

1.4 The system should enable a difficulty setting

1.5.1 The system must be able to allow for a difficulty setting of Slow, medium or fast.

1.5.1.1 The difficulty setting will enable a higher multiplier of points when the player collects a Gem and a Super Gem (see 1.5.1)

1.5.1.2 The higher the difficulty will increase the speed of the Ekans.

1.5 The system should enable a scoring system

1.5.1 When collecting a Gem the player should get 1 points times the game speed.

1.5.1.1 The game speed will be 1, 2 or 3, representing slow medium or fast

1.5.2 The score will be printed on the screen as the game progresses

1.5.3 The score will be printed at the end including the number of Gems and number of Super Gems the player has collected.

Non-functional requirements:

2.1 The frame rates during run time will be 5, 10 and 15 in accordance to the game speeds 1, 2 and 3 respectively. (see 1.5.1.2)

2.2 The average response time between the directional key and reaction must be less than the speed divided by the clock tic. This is before the Ekans is re-drawn otherwise the direction wont change.

Game Design

Rules:

1. Direct a Snake called Ekans around the screen with the arrow keys, you can’t turn backwards on yourself, so only turning 90\* of your current direction.
2. Avoid a collision with the edges of the screen and yourself.
3. Pick up Gems with your Ekans to extend your snake and get points.
4. Picking up Super Gems gives you bonus points and appears after every 5 Gems have been collected.

Controller behavior:

The snake will move either up, down, left or right. Not diagonally. The direction will be changed by pressing arrow keys, but will only change if you have selected a direction with 90° difference to the current direction. The Ekans will move on a grid where all the elements are locked onto, this grid will end at the border and each block is 25 pixels by 25 with the board size of 500 pixels by 500.

The player directs the Ekans with the directional arrow keys from the keyboard and any other input is ignored. The Ekans reacts after each tic of the frames per second tick.

Pickups:

The pickups will be given random coordinates every time one spawns, these will be locked to the same grid as the Ekans and will be one block in size. Only a single Gem will be on the screen at any given time and when one is destroyed another is created. When 5 Gems are destroyed a ‘SuperGem’ is created along with a Gem and a ‘SuperGem’ is worth 10 points, times the number of a points for a Gem, but disappears after 5 seconds of spawning.

Object structure:

Ekans will be made up of equal segments that follow each other through movement, each occupying a square on an invisible grid. The gems will also take up a square of the same size to make alignment more visible to the player.

Object looks:

Ekans will be block colour of purple and will be played upon a green background with blue Gems and SuperGem of blue and red.

Options:

A difficulty changer will be available which will simply change the speed at which the snake moves. 1 being slow 3 being fast. This will alter the score and give the player an advantage to playing faster. The number of segments added per gem and the speed multiplier per gem can be changed as an additional option.

Scoring:

The scoring system will be simple. On the easiest difficulty the user will score 1 point per gem, and for harder difficulties score multipliers will be made active. For example 2x speed will also mean a 2x score multiplier, meaning 2 points per gem.

Prototype Evaluation:

This was a half functioning program with the snake moving a single direction, the starting one. Movement from the player wasn’t added yet, but the snake still moved by shifting each block the current direction the snake was moving. The program still couldn’t properly exit and didn’t have a menu function.

Overall the prototype reflected the game somewhat, and the client had a jest of the functioning game.

Function Evaluation:

|  |  |  |
| --- | --- | --- |
| Function Name: | Description: | Desirability: |
| randomGem | Generates a random position for the gem | desirable |
| collide | Check to see if 2 sets of coordinates overlap | mandatory |

Testing

Test Cases:

Black box testing was decided and used, but while White Box testing validates the actual software code, the main focus of black box testing is on the validation of the functional requirements, which are and should be the main focus.

|  |  |
| --- | --- |
| Initial state- SS01  From the very start of the program, Menu items are displayed correctly and the first menu item is highlighted correctly.  From SS01 - pressed up - no change | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS01.png |
| From SS01 - pressed down - SS02  Highlighted Speed. | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS02.png |
| From SS02 - pressed return - SS03  Speed changes up one, to 2.  From SS03 - pressed return - SS04  Speed changes to 3  From SS04 - pressed return - SS02  Speed changes back to 1 | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS03.png |
| From SS02 - pressed down - SS05  Quit is selected.  From SS05 - pressed enter - window closes  From SS01, 02, 03, 05 - pressed 'x' in window bar  Game quits correctly on 2 above cases | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS05.png |
| From SS01 - pressed return - game starts - SS06  Snake is drawn with a single Gem, Score is set at 0 and Snake starts on a downward direction  From SS06 - pressed up - no change  Snake cannot move into itself. | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS07.png |
| From SS06 - pressed left - SS07  From SS06 - pressed right - SS08  Snake is controlled by the arrow keys correctly and moves around the screen as expected. | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS08.png |
| From SS06 - picked up gem - score and length increase - SS09  From SS06 - picked up 5 gems – super gem appears - SS10 | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS10.png |
| From SS10 - waited for 50 ticks – super gem disappears - SS11  50 ticks of the body means the super gem disappears, a tick being a shift of the Ekans body and a frame tick. | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS11.png |
| From SS10 - picked up super gem - Ekans does not grow - SS12  The Ekans only grows when normal Gems are picked up | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS12.png |
| From SS06 - collision with wall - SS13  Game over screen is displayed with results. | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS13.png |
| From SS06 - collision with self - SS14  Same as above. | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS14.png |
| From SS06 - picked up 5 gems, 1 super gem and died - SS15  Collided with the side after collecting the shown Gems.  From SS15 - pressed return - SS01  Returns to Menu | Macintosh HD:Users:Harry:Downloads:Ekans Testing:SS15.png |

Evaluation and Conclusion

User evaluation:

We tested the game with a variety of clients this is an example client evaluation of the game:

Evaluation of client feedback:

|  |
| --- |
| “Very simple but fun game, couldn't find any glitches anywhere. Visually colourful but rather boring, everything is made of coloured blocks with no detail at all.” |

The block coloured non-detail blocks were used to create a funny stand alone application, so no sprites had to be used. This means the game is completely portable and can be played in almost any device.

|  |
| --- |
| “Difficulties were set very well, I found 1 easy, 2 about my skill level and 3 I found difficult. No way of pausing the game which seems like an important thing to have, but a single game does not last very long so not too much of a problem.” |

The ability to pause the game could be implemented by stopping the frame upon a certain key press from the player, but because the Ekans game generally last 1 or 2 minutes, we found this an unnecessary addition.

|  |
| --- |
| “I found the scoring scaled well with the difficulty, I was scoring the most on difficulty 2. Supergems were easy to get at first, but once Ekans gets quite long they are sometimes unreachable. Very easy to keep playing, once a game is finished starting a new one takes a couple of key presses. I always found the original Snake a very addictive and this has that same effect. It's fun trying to top my personal best after every game.” That tense desperate struggle when you get very far and you are running out of room is also very fun” |

Super Gems are meant to be sometimes unreachable at a later stage, so we felt that you have to navigate around with that in mind to add a difficult element to the game.

Conclusion:

In conclusion there were a few points that we found through testing and evaluating the game, these are as follows:

* The difficulty or lack of was hard to find, increasing the speed did increase the point multiplier and also increased the difficulty. But the clients felt that a more challenging environment could be further implemented, for example having walls for the Ekans to maneuver around, this could be implemented as a level system.
* The shifting of the Ekans body was a trying factor throughout, as the segments needed to be stored in multiple arrays. Although when adding new segments when colliding with a Gem meant a shift of all the segments in the array and the addition of a single segment. So this was an overall easier method than just removing the last element and adding another in the current eaten Gem’s place.
* When producing random coordinates for the Gem and Super Gem, we came across a bug where the Gem was spawning on top of the current Ekans body. This had to be changed by checking all the coordinates in the elements in the Ekans array, with the random coordinates. This method was the most efficient we found and successfully created random coordinates.
* Block Graphics-Vs.-Sprites: We found that using block colour was an easier method of drawing the Ekans body to the screen than sprites. This meant that it was a complete stand-alone application, which doesn’t rely on any outside resources. This also means less storage space for the application to run in, and a more generic game in terms of portability.
* Keyboard input was well maintained and only minor problems were brought up through development. On key press the Ekans body positions are updated and anymore keyboard input is rejected until the Ekans is redrawn at the end of each timer tick. So if a player presses up then left in quick succession (before the end of the timer tick) it would’ve registered left as the direction. So using else if’s instead of if statements that was altered to now only accept the very first keyboard press. – Note this was only a noticeable problem on very easy difficulties where the Ekans moved slowly so the tick was slower.
* Collision with the outside border was problematic if the border was automatically redrawn to a different size each time, so a fixed border size was introduced. We found that the Ekans was drawn on top of the outside wall then the game ends, which leads the player seeing that the Ekans is over the wall. This could be fixed by redrawing the wall at collision and on top of the Ekans.
* Collision with itself was similar to colliding with the wall. Where the Ekans collided with itself, and the head overlapped the body making the Ekans appeared to be shrinked, this could be prevented by leaving the loop before the drawing of the next frame.

In conclusion, the majority of the client requirements matched the implementation, thus producing a fulfilling game. In addition to the client feedback and the black box testing, the game matched all the client standards.