IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

REVIEW OF COMPUTER ENGINEERING STUDIES
Vol.1, No.2, 2014, pp.11-14
http://dx.doi.org/10.18280/rces.010203

BASED ON THE NINE SUB-LOCK BEETLES FLASH GAME DESIGN

Huimei Yan, Kunliang Xu

School of Computer Science and Engineering, QuJing Normal University, QuJing, 655011, China.

Email: xkunl_2006@163.com

ABSTRACT

This paper introduces a flash software, one designed as network of small game based on an ancient game, after the game starting, there will be a 15 × 15 checkerboard, and the central of board has a beetle, nine pieces on the below of chessboard, one player each falling a chess piece, the beetle will move to the lattice of no chess piece until it is surrounded. The game is easy to operate, no need to install, small volume of file, suitable for network online use.

Keywords: FLASH, Chessboard, Game.

1. INTRODUCTION

Flash game is the rising of a new form of game, because the game has the advantage of easy operation, green, no installation, small volume of file, etc. are loved by the majority of users [1]. FLASH game in the form of performance is basically the same as the traditional game, because of its small volume, fast spread, beautiful screen, as long as the user's browser install the ADOBE Flash player can play all Flash games, having the trend of replacing traditional web online games. Thus, the domestic and external production for online games with FLASH has become a trend [2].

In this paper, the design of the game's protagonist is a beetle, this is an old game. Legendary supporting sky four pillars have a broken, so the Jade Emperor sent a fairy to repair, he used nine stones to fix a seaside beetle, used to make the cornerstone of broken columns, since then the sky stabilized, so this game is handed down. After the start of the game, a 15 × 15 checkerboard appears on the stage, there is a beetle on the central of board, nine pieces on the below of chessboard, one player each falling a chess piece, the beetle will move to the lattice of no chess piece until it is surrounded. Players use a maximum of nine pieces to surround the beetles and win the victory, if a player runs out of nine pieces, and beetles have not been surrounded, the game failed [3-4]

2. FLASH SOFTWARE DESCRIPTION

The predecessor of Flash is Future Wave's Future Splash, and it is the world's first commercially available two-dimensional vector animation software for designing and editing Flash document. In November 1996, the American company named Macromedia bought Future Wave, and renamed Flash. Later Macromedia was acquired by Adobe. The latest version is Adobe Flash CS5 [5-6].

Flash is a vector-based graphics system, the elements are vector, just use a small amount of vector data, and you can describe a complex object, the storage space occupied by graphics is only a few thousandths of a bitmap and the zoom in and out of graphic can't reduce picture quality. It uses plug-in mode to work. Users simply install a plug-in, and then you can quickly start and watch the animation. Supports bitmap, sound, gradient color, Alpha, transparency, etc., supports the network in real-time playback, very suitable for transmission over the network and online [5-6].

The applications field of Flash is very wide, having obtained the very good application in movies, television, cartoons, vocal music and other fields, saving costs and ensuring the smooth and picture quality, and promote the integration and development of traditional media and the Internet media. With the further development of network technology, FLASH will be well applied in application development, software interface development, the development of mobile phones in the field of game development, Web application services, site construction, media and entertainment, education system, will be "the most flexible front desk." [5-6].

3. GAME INTERFACE DESIGN

The game interface mainly includes game description, chessboard, beetles, several components material of beetles move, the voice prompts of success or failure and control buttons, running interface of the game shown in Figure 1.

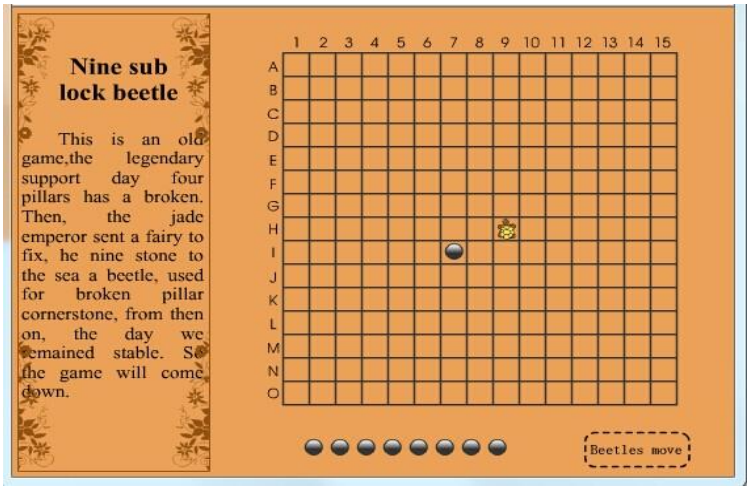

Figure 1. Game running interface map

Component design specific steps involved in the game are as follows [7-10]:

(1) New movie clip symbol named “game description”, draw a rectangle border, draw some pattern on the inside of the frame, insert static text on the inside “Nine sub lock beetle”, as well as a brief game text.

(2) New movie clip symbol named “chessboard”, draw a square and a square divided into 15 × 15 grid, length of each grid is 20 square, above the square input the number “1-15” representing chessboard abscissa, enter the letters “A-O” in the left side of the square representing the board ordinate.

(3) New movie clip symbol named “bettles”, the layer renamed “bettles”, draw a slightly smaller “bettles” than the grid of the checkerboard, insert key frame in the 7th,13th,19th frame, blank key frame inserted in the 4th,10th,16th frame, form the flashing animation of beetles. Create a new layer “actions”, in 19th frame insert a key frame, and on the frame add the code “stop ()”.

(4) New movie clip symbol named “pieces”, draw a black piece.

(5) New button element named “Hide button”, in the button symbol “click” insert key frame, slightly smaller circle than the grid of the chessboard.

(6) New movie clip symbol named “Hide button MC”, put the button element “Hide button” into the first frame.

(7) New movie clip symbol named “beetles move”, rename the layer “bottom”, with a dotted line to draw a rounded rectangle not filled with color, insert the key frame in the fifth frame, fill yellow to the rounded rectangle. Create a new layer, named “beetles move”, enter the text “beetles move” on the rounded rectangle. New Layer “actions”, in the first frame add code to “stop ();”. When the beetles move, this component starts playing.

(8) New button element named “Start.”

(9) New button element named “re-play.”

(10) New movie clip symbol named “result”, rename the layer “Results”, insert the key frame in the second frame, draw a rounded rectangle, and in the above enter text “failure”, in the third frame insert key frame, change the text “failure” to “success.” New Layer “actions”, respectively, in the 1th, 2th, 3th frame add code “stop ();”.

4. THE MAIN PROGRAM CODE OF THE GAME

The code of beetles move below:

```
If (_root.playnow == “com puter”) { // If you allow computer to operate
        xz = _root.xxx; //Let the lattice number of transverse assign to xz

        yz = _root.yyy; //Let the lattice number of longitudinal assign to yz
        dong = 0; // Determine in which direction it should move a pawn variables
        _root.table[yz][xz] = 0;
        dongArray = new Array(1, 2, 3, 4);//New array
        long = dongArray.length;//Long is the length of array dongArray
        if (_root.table[yz-1][xz] == 1) {
                dongArray[0] = 0;          }// If there is a pawn on the top of beetles, the first element of the array dongArray of 1 to 0
        if (_root.table[yz+1][xz] == 1) {
                dongArray[1] = 0;
                        }// If there is a pawn on the bottom of beetles, the second elements of the array dongArray of 2 to 0
        if (_root.table[yz][xz-1] == 1) {
                dongArray[2] = 0;
   // If there is a pawn on the left of beetles, the third elements of the array dongArray of 3 to 0
        }
        if (_root.table[yz][xz+1] == 1) {
                dongArray[3] = 0;
                // If there is a pawn on the right of beetles, the fourth elements of the array dongArray 4 to 0      }
        for (num1=3; num1>0; num1--) {
                for (i=0; i<=num1; i++) {
                        if (dongArray[i] == 0) {
                                dongArray.splice(i,      1); //Delete all the element equal 0 in array dongArray

                                long--;
                                break;
                        }
                }
        }
```

```
        if (_root.table[yz-1][xz-1] == 1 && _root.table[yz-1][xz+1] == 1 && _root.table[yz-2][xz] == 1 && long>1) {
                // If the empty lattice above the beetles has been surrounded by three sub-pieces, and there are other spaces around beetles
                for (i=0; i<=3; i++) {
                        if (dongArray[i] == 1) {
                                dongArray.splice(i, 1); //Delete the element equal 1 in array dongArray
  long--;
                                break;
                        }
                }
        }
        // If the empty lattice above the beetles has been surrounded by three sub-pieces, and there are other spaces around beetles, beetles will not jump above grid
        if (_root.table[yz+1][xz-1] == 1 && _root.table[yz+1][xz+1] == 1 && _root.table[yz+2][xz] == 1 && long>1) {
                // If the empty lattice below the beetles has been surrounded by three sub-pieces, and there are other spaces around beetles
                for (i=0; i<=3; i++) {
                        if (dongArray[i] == 2) {
                                dongArray.splice(i, 1); //Delete the element equal 2 in array dongArray
                                long--;
                                break;
                        }
                }
        }
        if (_root.table[yz-1][xz-1] == 1 && _root.table[yz+1][xz-1] == 1 && _root.table[yz][xz-2] == 1 && long>1) {
                // If the empty lattice on the left of the beetles has been surrounded by three sub-pieces, and there are other spaces around beetles
                for (i=0; i<=3; i++) {
                        if (dongArray[i] == 3) {//Delete the element equal 3 in array dongArray
                                dongArray. splice (i, 1);
                                Long--;
                                break;
                        }
                }
        }
        if (_root.table[yz-1][xz+1] == 1 && _root.table[yz+1][xz+1] == 1 && _root.table[yz][xz+2] == 1 && long>1) {
                // If the empty lattice on the right of the beetles has been surrounded by three sub-pieces, and there are other spaces around beetles
                for (i=0; i<=3; i++) {
                        if (dongArray[i] == 4) {
                                dongArray.splice(i, 1);//Delete the element equal 4 in array dongArray
                                long--;
                                break;
                        }
                }
        }
        // In order to make the computer more "smart", let beetles not move to the grid surrounded by three pieces, unless you have nowhere to go.
        suiji = random(long);//randomly get the serial number of array dongArray
        dong = dongArray[sui ji];//Let this serial number of the corresponding element assign to the variable dong
        if (dong == 1) {
                _root.gui._y -= 20;
                yz = yz-1; // If the dong is equal to 2, turtle walking down, the lattice number of longitudinal reduce 1
         }
        if (dong == 2) {
                _root.gui._y += 20;
                yz = yz+1; // If the dong is equal to 2, turtle walking down, the lattice number of longitudinal plus 1
        }
        if (dong == 3) {
                _root.gui._x -= 20;
                xz = xz-1;// If the dong is equal to 3, turtles turn left, the lattice number of transverse reduce 1     }
        if (dong == 4) {
                _root.gui._x += 20;
                xz = xz+1;// If dong equals four, turtle right away, the lattice number of transverse plus 1
  }
        win = 0;//initialize the variable win
        if (_root.table[yz][xz+1] == 1 && _root.table[yz][xz-1] == 1 && _root.table[yz+1][xz] == 1 && _root.table[yz-1][xz] == 1) {
                win = 1;
        }// If the upper and lower of beetles have a piece about turtles and the beetles have been surrounded, the variable win becomes 1
  }
  _root.xxx = xz;
  _root.yyy = yz;//Let the lattice number of transverse and longitudinal respectively assign to xxx and yyy
  guinum=yz*15+xz;//obtaining the number of grid beetles currently located and assign to guinum
  _root["dian"+_root.ti]._visible = 100;//Beetles walked to the place where you can put pieces
  _root.ti = guinum;// The position after the change of beetles assigned to _root.ti
  _root["dian"+guinum]._visible = 0;// Beetles walk to the place where you can't put pieces
  _root.playnow = "player";// change the game player to begin operation
  if (_root.step == 9 && win == 0){// If you use up nine pieces and haven't surrounded beetles
        _root.jieguo.gotoAndPlay(2);        //The emergence of failed feedback
        _root.losesound.start(); //play failed voice of player library
        for (num=0; num<225; num++) {
                removeMovieClip(_root["dian"+num]);
        }       // Delete all the hidden buttons on the chessboard, not recapture pieces any more
  }
  if (win == 1) {           // If win is equal to one a beetle that has been surrounded beetles
        _root.jieguo.gotoAndPlay(3); //The emergence of successful feedback
```

```
_root.winsound.start();//play successful voice of player library
for (num=0; num<225; num++) {
        removeMovieClip(_root["dian"+num]);
}       // Delete all the hidden buttons on the chessboard, not recapture pieces any more
```

5. CONCLUSION

There are many flaws and shortcomings in the game, such as the rough interface, not quite fine, simple level design, challenging not high. But this game design is an important test for my university learning outcomes, encountered many difficulties in the development process, but when solving a difficult time, I will feel very happy, and also get a very valuable experience in the development process.

REFERENCES

1. Yong Li, Wei Li, Zhenglin Song, Falsh Interactive Game Production Paradigm Navigation, Tsinghua University Press, Beijing, 2008.
2. Qigui Fang, Flash Multimedia Courseware Case Tutorial, Tsinghua University Press, Beijing, 2012.
3. Song Hu, Xiaokang Liang, Juan Zhao, Chinese Version of the Standard Flash CS5 Tutorial, China Youth Press Beijing, 2011.
4. Jingjing Chen, Talking about the Process of Teaching Flash Animation Game Created, *Heilongjiang Science and Technology Information*, vol.1, No.08, pp.50-55, 2012.
5. Guozhi Zhao, Lu Zhao, The Color Composition and Graphic Design Art, Fine Arts Publishing House, Liaoning, 1999.
6. Yu Hao, The Science of Happiness Research Group of Words Flash Game and Making, Wuhan University, Wuhan, 2011.
7. Ke Liang, Xingying Fu, Liu Yuan, Flash Multimedia Courseware Design, *Software Guide*, vol.4, No.2, pp.20-30, 2014.
8. Weihua Zhu, The Advantage of Analysis Flash Animation Design, *Industry Science and Technology Forum*, vol.15, No.8, pp.30-35, 2013.
9. Haihua Liu, Research and Application of Flash Game, *Computer CD Software and Application*, vol.20, No.17, pp.15-20, 2011.