Licor

CATCH BUGS

ENVIRONMENT

Xcode on Mac

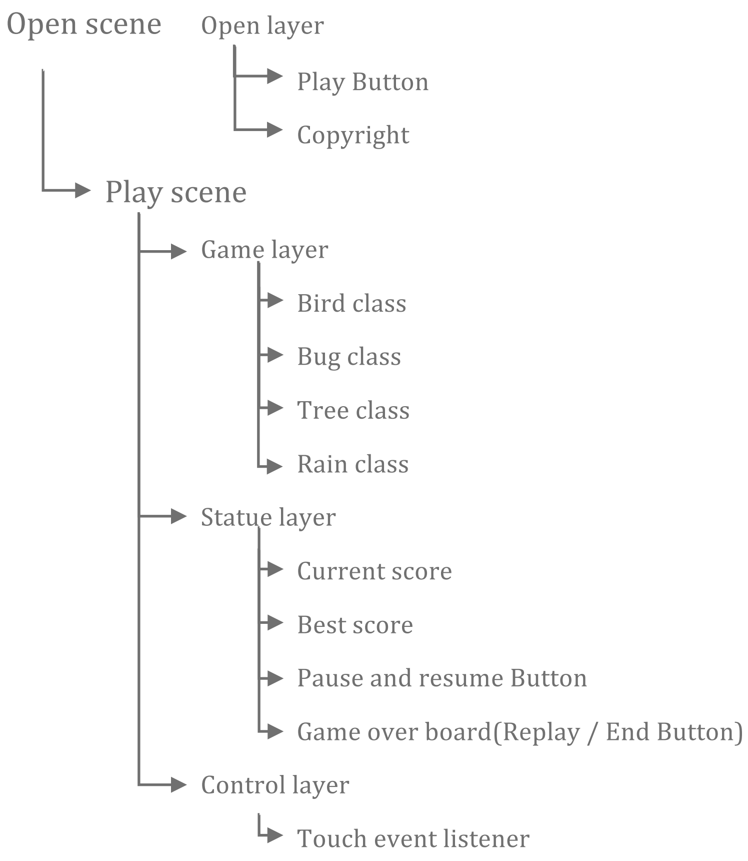
Desktop version

Build and test environment: 2.2 GHz Intel Core i7
16 GB 1600 MHz DDR3

INTRODUCTION

A simple game runs on Mac. Click to control the bird, every click will enable the bird to fly higher, otherwise it will fall down. Stay away from the trees and don't hit the ground, or you'll lose the game. Catch bugs to get the score, each bug worth 1. And it'll rain randomly, so don't be influenced by the rain drops.

STRUCTURE



- Open scene
  - Open layer
    - Play Button
    - Copyright
  - Play scene
    - Game layer
      - Bird class
      - Bug class
      - Tree class
      - Rain class
    - Statue layer
      - Current score
      - Best score
      - Pause and resume Button
      - Game over board(Replay / End Button)
    - Control layer
      - Touch event listener

VARIABLE NAMING

All the variables and functions are little camel-case, while the class names are big camel-case.

CLASS USED FROM COCOS

Layer, Sprite, Node, PhysicsBody, PhysicsShapeEdgeSegment, Director, RandomHelper, SimpleAudioEngine, Size…

DETAIL

Scene manager to arrange the scenes, though there're just 2 scenes at present. In this class 2 scenes are created and a pointer is added into the scene for the replacement.

Open scene includes the name of the game play button and the copyright.

Play scene is made up of three layers.

Game layer includes all the game objects, like the bird, bugs, trees and rain, deals all the matters between these objects, such as their contact, their states change.

Status layer shows the current score, best score, and buttons like pause during the game playing and replay or end when the game is over.

Control layer just listen the touch event and pass it to the game layer to control the bird.

Game layer is a class derived from the **control delegator**, which is synthesized in control layer. The touch event is passed to the game layer by the control delegator.

Status layer derived from the **status delegator**, which is synthesized in game layer, pass the data like the score to the status layer. And a pointer to the game layer is passed to the status layer by the status delegator for the control of pause and resume of the game layer.

Bird class controls everything related to the bird, such as the states, action, its appearance and physics body. A static pointer called shareBird to the Bird class in the class guaranteed that there is only one bird created and the bird added in the game layer can be controlled just by this pointer.

Bug class controls the bug's position, visibility, and its state, and **bug manager** create several bugs and added them to the scene, using a vector to store and manage the bugs.

Tree class creates 4 types of trees, and **tree manager** randomly add 3 trees to the scene, at random position.

Rain class creates the rain drops which are stored in a vector and add them to the scene to make it like rainy. After the rain drops under the ground the drop will be collected and reuse. It will rain at random while playing the game, the rain will not only have an impact on the bird, but make the sight more bewildering.

IDEAS FROM THE CREATOR

We are so often experience the frustration results from the failures of being unable to run our code successively, and sometimes only to find a bunch of bugs in it.

So, I came up with an idea to make a game to show our everlasting struggle with the bugs. We can catch them, kill them, however, they can't be eliminated totally. What we can do is to find as many of them as we can.

And every project is a new journey, an arduous journey. Once you start making a new project, you just start a new journey. There is no need to look back at how glorious and splendid past you've made, but just focus on the present, on the future, so, the best score in the game is temporary, it won't be stored forever, and every time you play the game, you just start a new journey, trying to do better.

CREATOR

Licor (Li chuanrong, 1552653)