Making Pong Game using LÖVE

LÖVE is a framework you can use to make 2D games in Lua. It's free, open-source, and works on Windows, Mac OS X, Linux, Android and iOS. Not only is Löve2D free of charge and open-source, but it's also cross-platform. This means that **players and developers of all stripes and systems can get in on the game development goodness**. As long as you can code in one of the two different languages (C++ and Lua), you'll be making Löve2D games in no time.

is this article we are going to talk a about LOVE framework and the Lua language, also we will discover its architecture and how to make a simple game in it

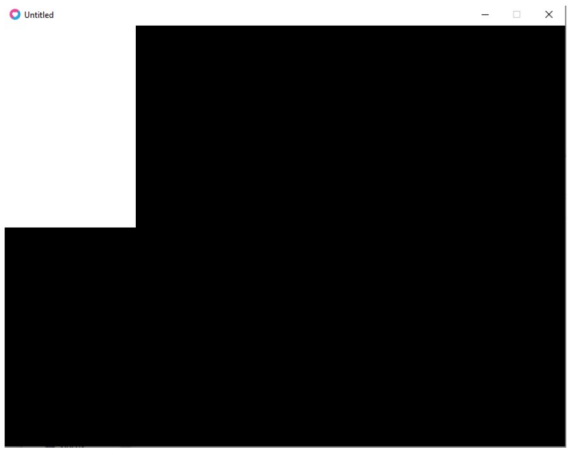
LOVE:

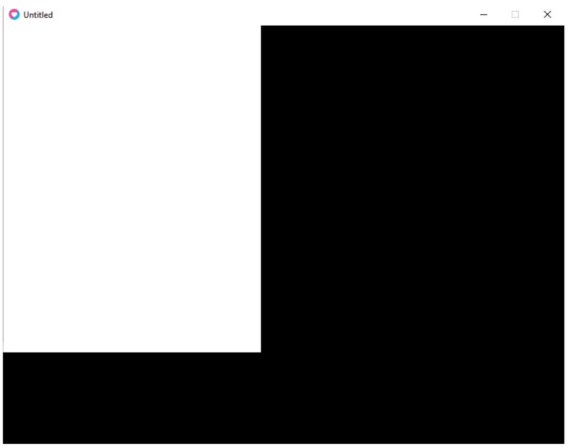
love architecture:

```
function love.load()
    height = 100
    width = 200
end
function love.update(dt)
```

```
    height=height+1
    width = width+1
end
function love.draw()
   love.graphics.rectangle("fill",0,0,height,width)
end
```

if we execute this project in the love.exe the app will show a getting big white rectangle:





let’s explain:

the first function love.load( ) ⇒ is an initialisation for all variables we need to use

the second function love.update( )⇒ called in every frame to do a certain thing (example : add 1 to the height and width)

the third function love.draw( ) ⇒ for drawing the shapes in the game , called also in every frame

this is simple explanation for the functions used in the LOVE architecture.

Lua benefice:

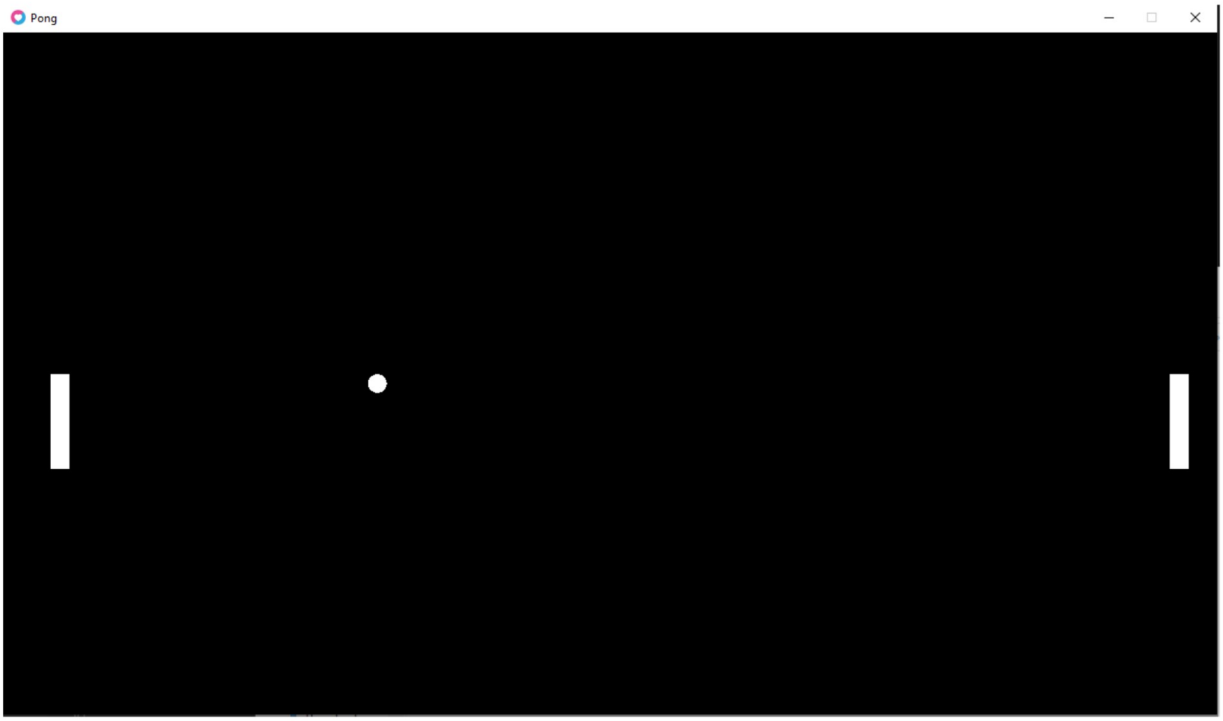
- It has been efficient in-memory usages.
- It has an open-source license.
- It is very easy to insert in C programs.
- It is usually a very good scripting for games.

- It is very simple to use and also to learn.
- Its syntax is pretty good.

Pong Game:

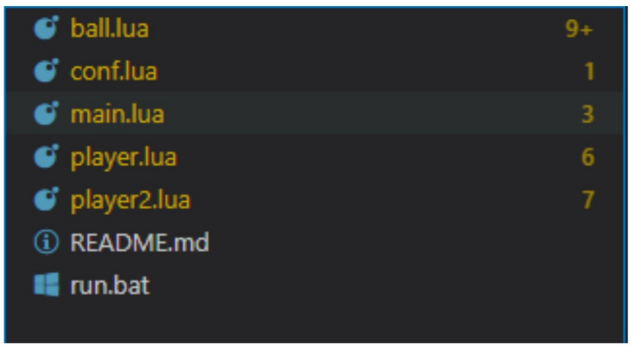
“Pong is one of the first computer games that ever created, this simple "tennis like" game features two paddles and a ball, the goal is to defeat your opponent by being the first one to gain10 point, a player gets a point once the opponent misses a ball. The game can be played with two human players, or one player against a computer-controlled paddle. The game was originally developed by Allan Alcorn and released in 1972 by Atari corporations. Soon, Pong became a huge success, and became the first commercially successful game, on 1975, Atari release a home edition of Pong (the first version was played on Arcade machines) which sold 150,000 units. Today, the Pong Game is considered to be the game which started the video games industry, as it proved that the video games market can produce significant revenues.”

we will try to make one with 2 players:



Spoileeeer !!

Project architect:



every file contains its functions and attribute :

we start with :

player.lua

```
Player = {}

function Player:load()
   self.x = 50
   self.y = love.graphics.getHeight() / 2
   self.width = 20
   self.height = 100
   self.speed = 500
end


function Player:update(dt)
   self:move(dt)
   self:checkBoundaries()
end


function Player:move(dt)
   if love.keyboard.isDown("w") then
      self.y = self.y - self.speed * dt
   elseif love.keyboard.isDown("s") then
      self.y = self.y + self.speed * dt
   end
end

function Player:checkBoundaries()
```

```
   if self.y < 0 then
      self.y = 0
   elseif self.y + self.height > love.graphics.getHeight() then
      self.y = love.graphics.getHeight() - self.height
   end
end


function Player:draw()
   love.graphics.rectangle("fill", self.x, self.y, self.width, self.height)
end
```

the new function move and checkBoundries are for making the pad move and to check if it super pass the boundaries

player2.lua

```
Player2 = {}

function Player2:load()
   self.x = love.graphics.getWidth()-50
   self.y = love.graphics.getHeight() / 2
   self.width = 20
   self.height = 100
   self.speed = 500
end


function Player2:update(dt)
   self:move(dt)
   self:checkBoundaries()
end


function Player2:move(dt)
   if love.keyboard.isDown("up") then
      self.y = self.y - self.speed * dt
   elseif love.keyboard.isDown("down") then
      self.y = self.y + self.speed * dt
   end
end

function Player2:checkBoundaries()
   if self.y < 0 then
      self.y = 0
   elseif self.y + self.height > love.graphics.getHeight() then
      self.y = love.graphics.getHeight() - self.height
   end
end
```

```
function Player2:draw()
   love.graphics.rectangle("fill", self.x, self.y, self.width, self.height)
end
```

same as player, but different setting of positioning and controlling

ball.lua

```
Ball = {}


function Ball:load()
   self.x = love.graphics.getWidth() / 2
   self.y = love.graphics.getHeight() / 2
   self.width = 20
   self.height = 20
   self.speed = 200
   self.xVel = -self.speed
   self.yVel = 0
end


function Ball:update(dt)
   self:move(dt)
   self:collide()
end


function Ball:collide()
   if checkCollision(self, Player) then
      self.xVel = self.speed
      local middleBall = self.y + self.height / 2
      local middlePlayer = Player.y + Player.height / 2
      local collisionPosition = middleBall - middlePlayer
      self.yVel = collisionPosition * 5
      if (self.speed<0)
         then
         self.speed=self.speed-10
         else
            self.speed=self.speed+10
         end
   end

   if checkCollision(self, Player2) then
      self.xVel = -self.speed
      local middleBall = self.y + self.height / 2
      local middleplayer2 = Player2.y + Player2.height / 2
      local collisionPosition = middleBall - middleplayer2
      self.yVel = collisionPosition * 5
      if (self.speed<0)
```

```
         then
         self.speed=self.speed-10
         else
            self.speed=self.speed+10
         end
   end

   if self.y < 0 then
      self.y = 0
      self.yVel = -self.yVel
   elseif self.y + self.height > love.graphics.getHeight() then
      self.y = love.graphics.getHeight() - self.height
      self.yVel = -self.yVel
   end

   if self.x < 0 then
      self.x = love.graphics.getWidth() / 2 - self.width / 2
      self.y = love.graphics.getHeight() / 2 - self.height / 2
      self.yVel = 0
      self.xVel = self.speed
   end

   if self.x + self.width > love.graphics.getWidth() then
      self.x = love.graphics.getWidth() / 2 - self.width / 2
      self.y = love.graphics.getHeight() / 2 - self.height / 2
      self.yVel = 0
      self.speed=200
      self.xVel = -self.speed
   end
end


function Ball:move(dt)
   self.x = self.x + self.xVel * dt
   self.y = self.y + self.yVel * dt
end


function Ball:draw()
   love.graphics.rectangle("fill", self.x, self.y, self.width, self.height,100)
end
```

the new function move and collide are meant to be the function controlling the movement of the ball and check collision with player 1 and 2

main.lua

```
require("player")
require("player2")
require("ball")
```

```
function love.load()
   Player:load()
   Player2:load()
   Ball:load()

end


function love.update(dt)
   Player:update(dt)
   Player2:update(dt)
   Ball:update(dt)

end


function love.draw()
   Player:draw()
   Player2:draw()
   Ball:draw()
   --AI:draw()
end


function checkCollision(a, b)
   if a.x + a.width > b.x and a.x < b.x + b.width and a.y + a.height > b.y and a.y < b.y +
b.height then
      return true
   else
      return false
   end
end
```

the function checkCollision is for checking collision between two objects a and b

conf.lua

```
function love.conf(t)
    t.title = "Pong"                -- The title of the window the game is in (string)
    t.version = "11.3"                 -- The LÖVE version this game was made ring)
    t.console = true                  -- Attach a console (boolean, Windows only)
    t.window.width = 1280              -- The window width (number)
    t.window.height = 720              -- The window height (number)
end
```

configuration file

Voilà: