Files Structure

The files are structured as follow

Folder	file	content
/		
	main.lua	Load the unittest and the application
	main_game.lua	Load the game through the /scenes/intro.lua
	main_test.lua	Load the unittests
	Build.settings	Used by Corona SDK for build
	config.lua	Used by Corona SDK for run
	*.ttf	Fonts used on this game
	*.png	Icos and lauchscreen for Android and iPhone
core		Holds the classes responsible for the in play game functions
	board.lua	Generic board game class for any board grid game
	boardgame.lua	Generic board game, uses for control a board
	Item.lua	Generic visual item to be use in a board
	gemboard.lua	Inherits from board and add more functionalities like settle gems and disposal gems
	gemgame.lua	Inherits from boardgame , control a gemboard adding all visual necessary for the gameplay
	gem.lua	Inherits from item, adds the skin of a gem
	utils.lua	Global helpers
scenes		Holds the storyboards/scenes
	game.lua	Create the scene and HUDs for the game
	start.lua	The first scene displayed
	restart.lua	Empty scene for let the game fully restart, it is the easiest way for clean up
scenes/hud		Holds the HUD used by the game scene
	gameintro.lua	The countdown at the start of the game
	timer.lua	Controls the time and all visuals related to it
	score.lua	Controls the score and all visuals related to it
	gameover.lua	Show the stars and reset button at the end
sounds		Holds the sounds and musics
unittest		Holds the unittests for core classes
vendors		Holds external librarys

BOARD.lua

new(properties)	constructor
fill(model, random)	Fill the board with empty items or random items
clear()	Remove all items
draw(parent)	Draw it self and all items
updated()	Not implemented
getItemAt(col, row)	Find the item in that location and returns it
getItemAtIndex(index)	Find the item in that index and returns it
getIndex(col,row)	Return the index of the location
getPosition(index)	Return an object with {col=x, row=x}
getIndexOfItem(item)	Return the index of the item
addItemAtIndex(item, index)	Add an item at the specific index
addItemAt(item, col, row)	Add an item at the specific location
addItem(item)	Add an item at the next available location or replace the last
createItem(index, addToBoard)	Create a new item [add to the board] and return it
removeItemAtIndex(index)	Remove item at the specific index
removeItemAt(col, row)	Remove item at the specific location
removeItem(item)	Remove item
removeItems(items)	Remove an array of items
swapItems(item1, item2)	Swap the position of two items
swapFromIndexToIndex(i1,i2)	Swap the position of two items at the indexes
moveItemTo(item, col, row)	Move an item to the specific location
moveFromIndexToIndex(i1,i2)	Move an item form a location to another by indexes
destroy()	Remove self
print()	Print the content of the board on console for debugging

Boardgame.lua

new(properties)	constructor
reset()	Stop the game and clean it up
restart( )	Stop, clean up and start again
start()	Start the game
stop()	Stop the game
pause()	Pause de game
destroy()	Destroy the board

Item.lua

new(properties)	constructor
draw(parent,x,y)	Draw it self to the parent at coordinates
remove()	Remove it self

Gemboard.lua (implements board.lua)

new(properties)	constructor
getItemByXY(x, y)	Get item at the coordinates
getIndexByXY(x, y)	Get index at the coordinates
testForMatchs(minmatchs, stopimediatly)	The core function of the game, it run through the items to find possible matchs, it return a bool if found or not plus a disposal collection
settleGems()	Move the gems down and add new one at the top
fill(...)	Fill the board and make sure it has no matchs on it
disposeItems(disposal, replace)	Remove the list of items on a disposal collection [and add new item in the place]

Gemgame.lua (implements boardgame.lua)

new(properties)	**constructor**
tryToSwapToIndex(index)	Try to swap the item on focus with the index
createBoard(options)	Create and draw a new board
start( )	Start game and add touch listener
restart( )	Stop, clean, fill and start a game
stop()	Stop the game
pause()	Pause the game
draw()	Draw the items
setItemOnFocus(index)	Set focus to the item been move or click
timer()	Delay listener to improve the fell of the game
swapGems()	Call to animate a swap movement
update()	The core of the gemgame, where everything hapen
countpoints(disposal)	Calculate the points and dispatch the result
animateDie(disposal)	Create a die animation for items in a disposal collection
animate(bounce)	The graphical runtime of the game

Gem.lua   (implements item.lua)

new(properties)	**constructor**
draw(parent, x, y)	Draw a gem in the parent at the coordinates

Utils.lua

copy()
Clone an item

fill(obj,with)
apply the properties of an object to another

len()
get the total number of items in a table including pars and ipars

sleep()
give some break