Inhoud

[Basic Cocos2d-x concepts 3](#_Toc492852050)

[Main components 3](#_Toc492852051)

[Scene graph 3](#_Toc492852052)

[Sprite 3](#_Toc492852053)

[Actions 4](#_Toc492852054)

[Action 4](#_Toc492852055)

[Sequence of actions 4](#_Toc492852056)

[Spawn 5](#_Toc492852057)

[Parent-child 5](#_Toc492852058)

[Logging 5](#_Toc492852059)

[All about Sprites 5](#_Toc492852060)

[Creation of a sprite 5](#_Toc492852061)

[Sprite Sheet 5](#_Toc492852062)

[Loading in a sprite 6](#_Toc492852063)

[Creating a sprite from a SpriteFrameCache 6](#_Toc492852064)

[Creating a sprite from a SpriteFrame 6](#_Toc492852065)

[Sprite manipulation 6](#_Toc492852066)

[The position of a sprite 6](#_Toc492852067)

[Skewing the sprite 7](#_Toc492852068)

[Changing color: 7](#_Toc492852069)

[Changing opacity (fading?) 7](#_Toc492852070)

[Polygon sprite 7](#_Toc492852071)

[All about actions 8](#_Toc492852072)

[Difference between MoveTo and MoveBy 8](#_Toc492852073)

[Rotation: 8](#_Toc492852074)

[Scaling: 8](#_Toc492852075)

[Fading in/Out: 9](#_Toc492852076)

[Tinting: 9](#_Toc492852077)

[Animation 9](#_Toc492852078)

[Animate 9](#_Toc492852079)

[Easing 10](#_Toc492852080)

[Cloning 10](#_Toc492852081)

[Reversing 11](#_Toc492852082)

[All about Scenes 11](#_Toc492852083)

[Creating scenes 11](#_Toc492852084)

[Transition scenes 12](#_Toc492852085)

[UI Components 12](#_Toc492852086)

[Labels 12](#_Toc492852087)

[BMFont 12](#_Toc492852088)

[TTF 12](#_Toc492852089)

[SystemFont 13](#_Toc492852090)

[Shadow 13](#_Toc492852091)

[Outline 13](#_Toc492852092)

[Glow 13](#_Toc492852093)

[Menu 13](#_Toc492852094)

[Menu callbacks with lamba function 14](#_Toc492852095)

[Buttons 14](#_Toc492852096)

[Checkbox 14](#_Toc492852097)

[LoadingBar 15](#_Toc492852098)

[ScrollView 15](#_Toc492852099)

[Slider 15](#_Toc492852100)

[TextField 15](#_Toc492852101)

# Basic Cocos2d-x concepts

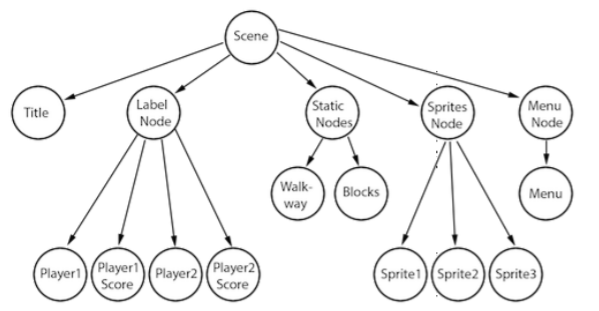
Cocos2d-x is a cross-platform game engine. It is an easy to use API to make the game you want without too much technical insight. Cocos2d-x includes a renderer, 2d/3d graphics, collision detection, sound, controller support, animations…

# Main components

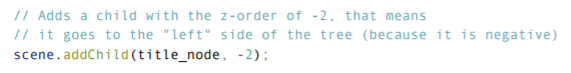
The most important components are Scene, Node, Sprite, Menu and Action. These components make up your game. You can see a scene as one moment in your game, like the first level or main menu. A Node is an object and a sprite is something that can be moved, like your main character.

The director controls the flow of your game, which Scenes comes next and how the transition goes. You can see it as the game flow.

## Scene graph

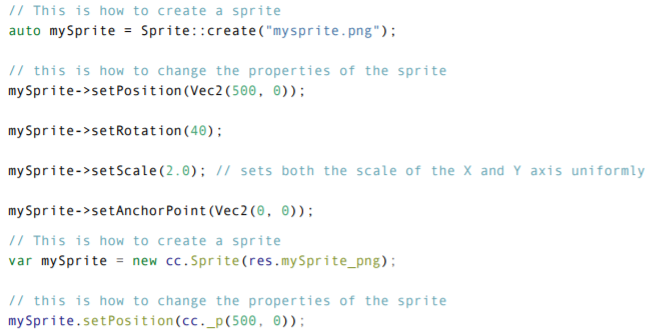


The scene graph is presented as a tree. This also gives the order in which the nodes are rendered and displayed. The scene graph uses the in-order, that means that the renderer first looks at the left side of the tree, then the root node and then the right side. So the nodes at the right side will be placed on the top of the scene. You can also compare this with the z-order. The higher the z value, the more to the foreground. Keep that in mind. You can also influence this z-value yourself:



## Sprite

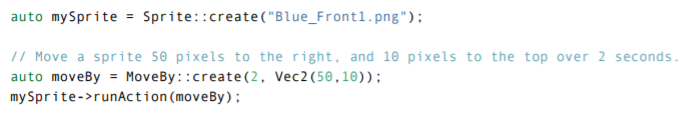
Sprites are a very important aspect of your game. Your main character is probably a sprite.  
You can do all sorts of things with your sprites…Change position, rotation, scale, opacity, color…



# Actions

## Action

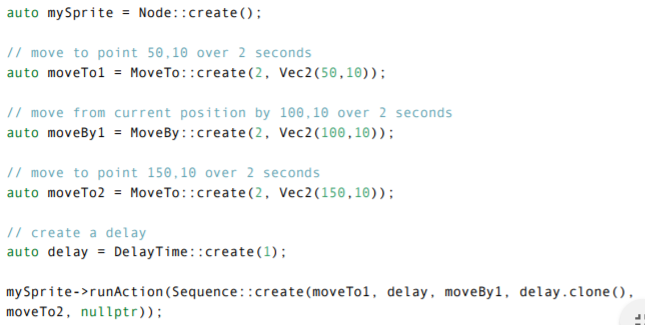
Actions allow transformations on Nodes, it is a very important aspect of any game. To have to change the position or scale of a node? No problem, Actions can do that.



## Sequence of actions

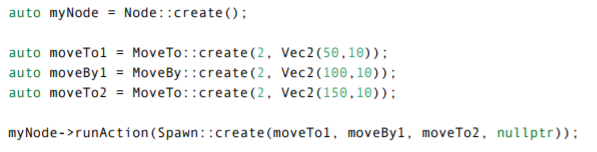
Maybe you want to have a sequence of action that happen after each other, that is also possible!





## Spawn

But what if you want to do multiple actions AT THE SAME TIME? Actions have your back!

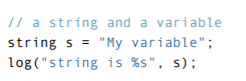


## Parent-child

You can make sure that other nodes copy each other, when one does something the others follow.

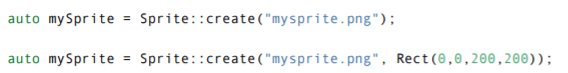
## Logging

It might be handy to log certain variables, we can do this thanks to log().



# All about Sprites

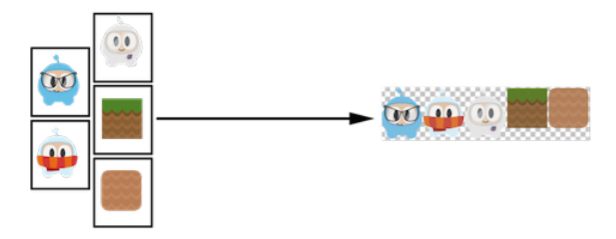
## Creation of a sprite



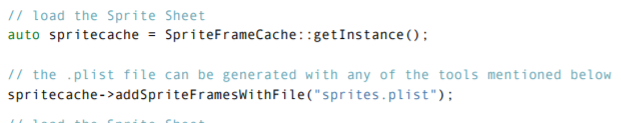
You can create a sprite in a few different ways. In the first one you create a sprite that is identical to the image file you opened. But when you create a sprite with a rectangle, you can choose the dimension of the sprite. Rect(ori.x, ori.y, width, height).

## Sprite Sheet

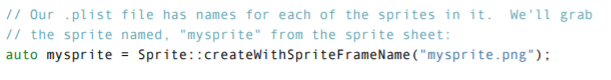
A sprite sheet is a file where all your images are combined in one. You can get the ones out you need.



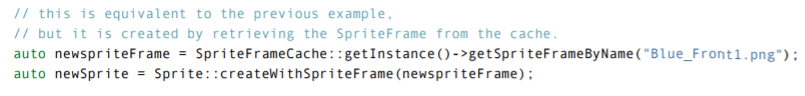
### Loading in a sprite



### Creating a sprite from a SpriteFrameCache



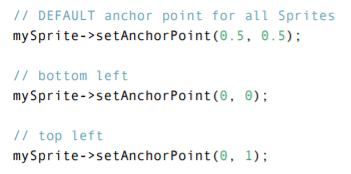
### Creating a sprite from a SpriteFrame



You can make your own sprite sheets with Cocos Studio!

## Sprite manipulation

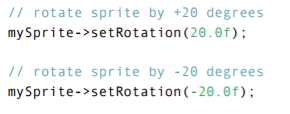
The anchor point is used to specify what part of the sprite will be used for setting the position.



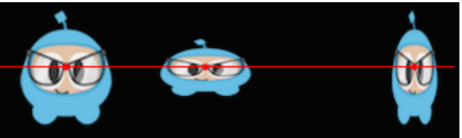
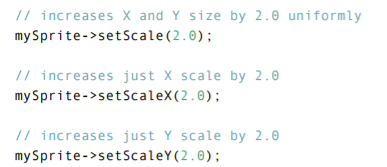
The position of a sprite is pretty self-explanatory. You can alter its position by using:  


Notice that changing the anchor point also slightly changes the position.

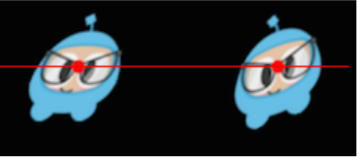
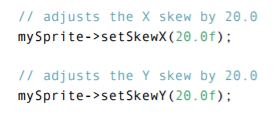
You can rotate your sprite, position value to go clockwise, negative to go counter-clockwise.



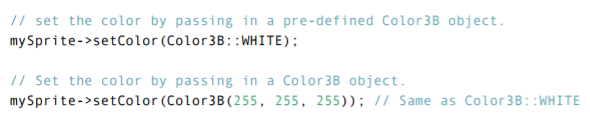
You can alter the scale of a sprite, only the x, only the y or together.



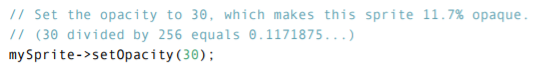
Skewing the sprite, you can see this as a stretch of your sprite. For example:



### Changing color:

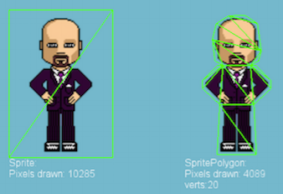
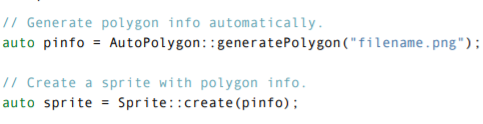


### Changing opacity (fading?)

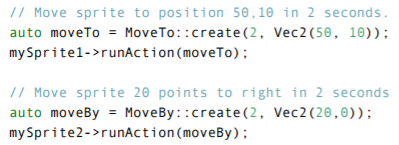


## Polygon sprite

A polygon sprite is a sprite that is made of multiple triangles, unlike a normal sprite that is made out of a rectangle of just 2 triangles. The purpose for this is mainly performance reasons, but it is also very helpful with collision detection. Using 1 big box around the sprite isn’t very helpful.

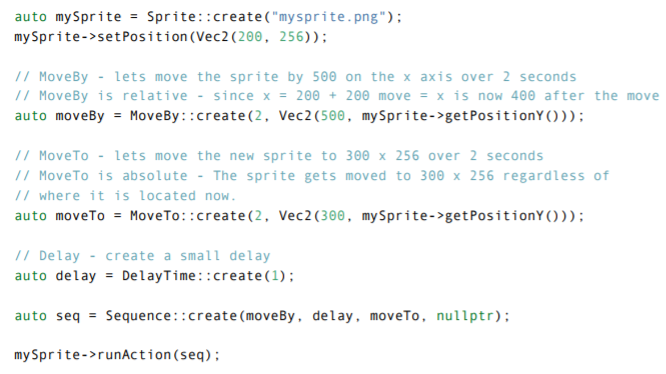
 

# All about actions

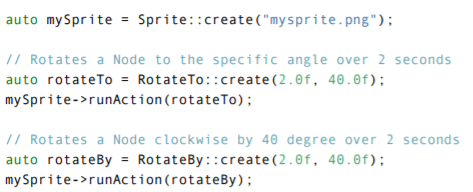


## Difference between MoveTo and MoveBy

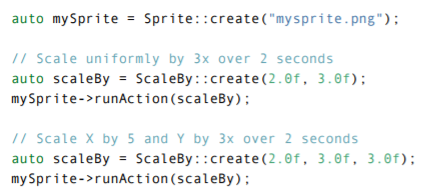
By is relative to the current state of the Node, To is absolute.



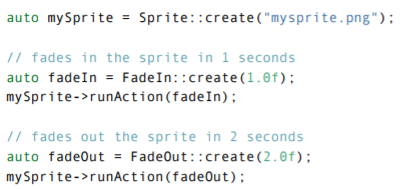
## Rotation:



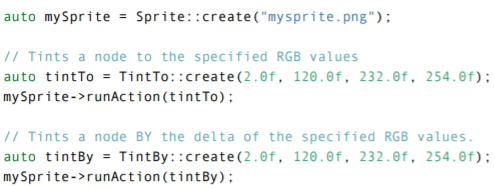
## Scaling:



## Fading in/Out:



## Tinting:



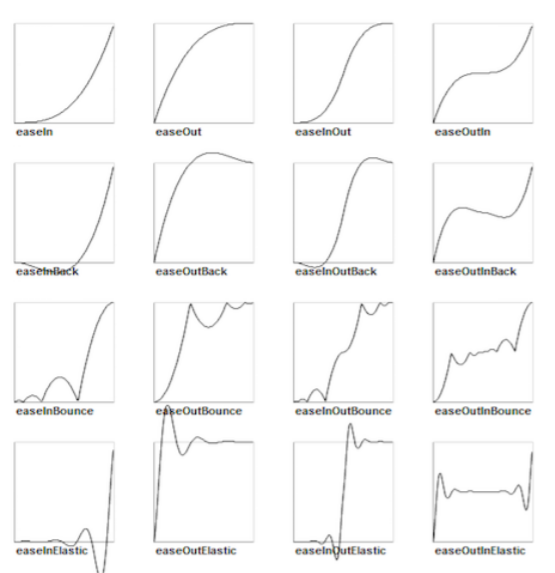
# Animation

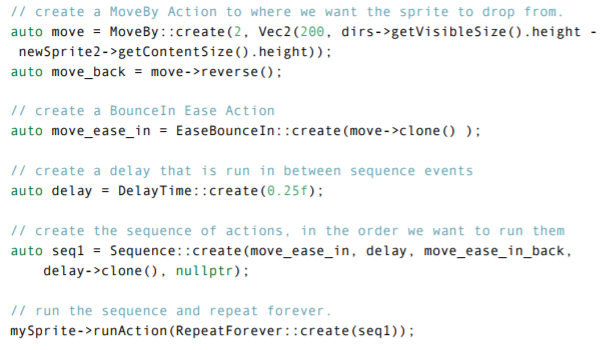
## Animate





## Easing





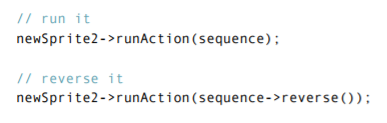
## Cloning

Because of the internal state of the action, using the same action twice can give you unexpected results.  
That’s why you have to clone them.



## Reversing

It is just as is sound, when you have a sequence of actions, with the reverse action, you can play them backwards.



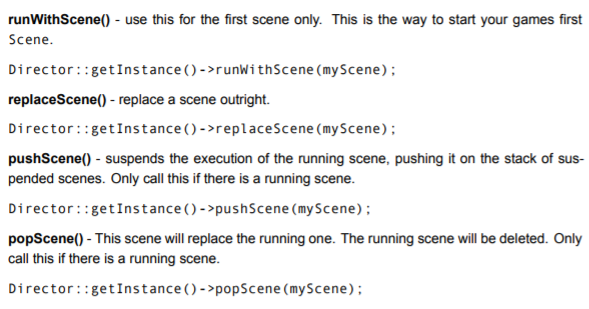
# All about Scenes

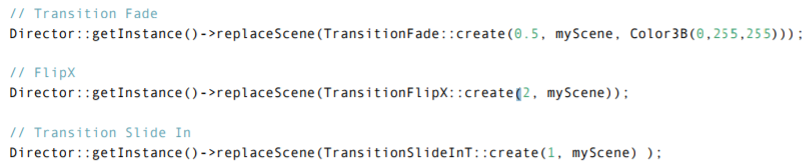
## Creating scenes

Simple example of creating a scene:



## Transition scenes





# UI Components

## Labels

### BMFont



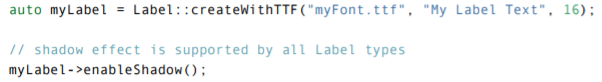
### TTF

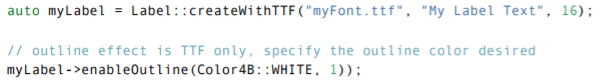

### SystemFont



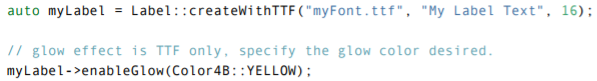
### Shadow



### Outline

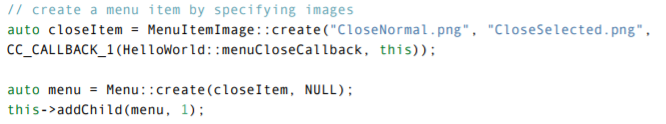


### Glow

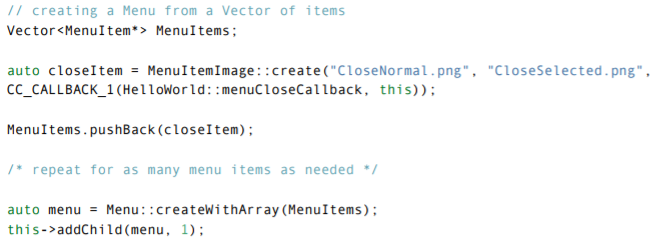


## Menu

Every game has menus, it’s a vital component for your game. A menu is made out of Menu Items, when you click on an item, a callback function is called for executing the choice.  
There are a few ways to make a menu.



With vectors:

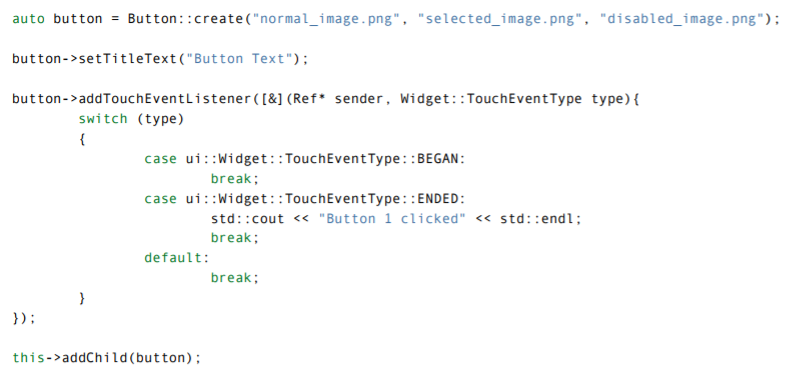


### Menu callbacks with lamba function



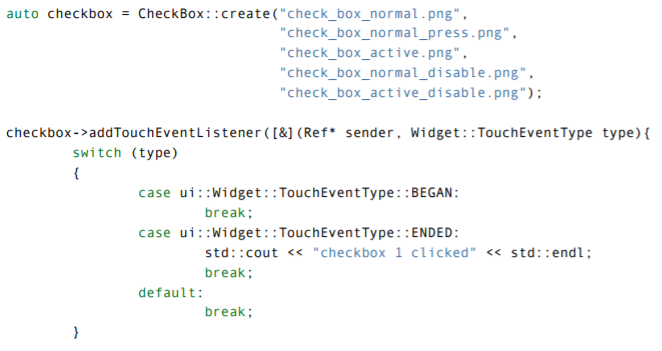
## Buttons

Buttons have a normal and a selected state. When pressed they call a predefined callback function.



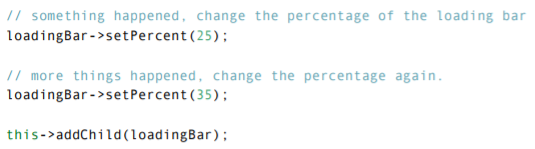
## Checkbox

Checkboxes are used for a binary choice.



## LoadingBar





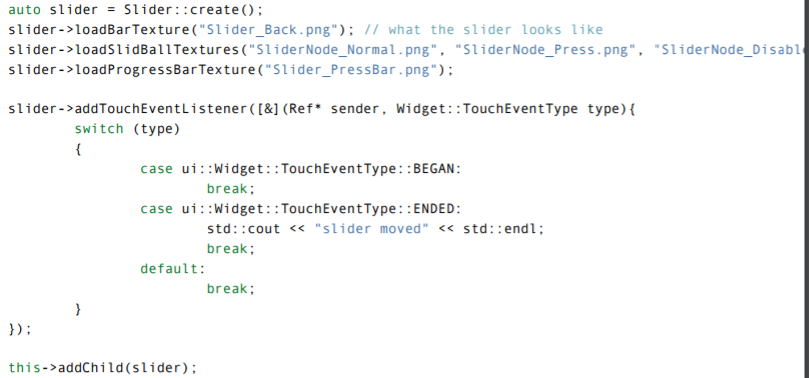
## ScrollView

A scrollview might be handy to use for the tower menu. Not all the towers can fit, U can scroll down.



## Slider

Good for sound adjustment or speed?



## TextField

Good for entering names/information.

