

THE CALLBACKS

`love.load(args)` Run once when the game starts.
`love.update(dt)` Used to update the game's state.
`love.draw()` Used to draw to the screen.
`love.quit()` Run when the game is quit.
Return true to cancel quitting.
`love.focus(focused)`
Run when the window gains/loses focus.

THE KEYBOARD

Callbacks

`love.keypressed(key, unicode)`
`love.keyreleased(key, unicode)`

Module `love.keyboard`

`isDown(key1, ...)`
`delay, interval = getKeyRepeat()`
`setKeyRepeat(delay, interval)`

THE MOUSE

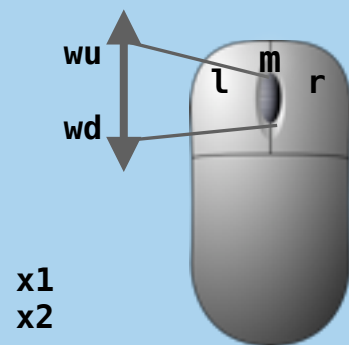
Callbacks

`love.mousepressed(x, y, button)`
`love.mousereleased(x, y, button)`

Module `love.mouse`

`x, y = getPosition()`
`getX()` `getY()`
`isDown(button1, ...)`
`isGrabbed()`
`isVisible()`
`setGrab(grab)`
`setPosition(x, y)`
`setVisible(visible)`

Enum `MouseConstant`



AUDIO

Module `love.audio`

<code>pause()</code>	<code>pause(source)</code>
	<code>play(source)</code>
<code>resume()</code>	<code>resume(source)</code>
<code>rewind()</code>	<code>rewind(source)</code>
<code>stop()</code>	<code>stop(source)</code>
<code>getNumSources()</code>	
<code>newSource(file, type)</code>	
<code>newSource(data)</code>	
<code>newSource(decoder, type)</code>	

Getters/Setters (prefix names with "get" or "set")

`Orientation(fx, fy, fz, ux, uy, uz)`
`Position(x, y, z)`
`Velocity(x, y, z)`
`Volume(volume)`

MODULE LOVE.GRAPHICS

Drawing

`draw(drawable, x, y, r, sx, sy, ox, oy)`
`drawq(img, quad, x, y, r, sx, sy, ox, oy)`

New Objects (prefix names with "new")

`Font(filename, size)` `Font(size)`
`Framebuffer(width, height)`
`Image(filename)` `Image(file)` `Image(data)`
`ImageFont(imageOrFilename, glyphs)`
`ParticleSystem(image, buffer)`
`Quad(x, y, width, height, sw, sh)`
`Screenshot()` -- returns `ImageData`
`SpriteBatch(image, size)`

Text/Shapes

`print(text, x, y, r, sx, sy)`
`printf(text, x, y, limit, align)`
`circle(mode, x, y, radius, segments)`
`line(x1, y1, x2, y2, ...)`
`point(x, y)`
`polygon(mode, ...)`
`quad(mode, x1, y2, x2, y2, x3, y3, x4, y4)`
`rectangle(mode, x, y, width, height)`
`triangle(mode, x1, y1, x2, y2, x3, y3)`

Transformations

`push()` `pop()`
`translate(dx, dy)`
`rotate(angle)`
`scale(sx, sy)`

Getters/Setters (prefix names with "get" or "set")

`BackgroundColor(r, g, b)`
`Color(r, g, b, a)`
`Font(font)` `Font(file, size)` `Font(size)`
`Icon(drawable)`
`Mode(w, h, fullscreen, vsync, fsaa)`

MODULE LOVE.TIMER

<code>getDelta()</code>	<code>getFPS()</code>
<code>getMicroTime()</code>	<code>getTime()</code>
<code>sleep(milliseconds)</code>	
<code>step()</code>	

Class `Source`

`play()` `resume()` `pause()` `stop()` `rewind()`
`isLooping()` `isStatic()` `isStopped()`

Getters/Setters (prefix names with "get" or "set")

`Looping(loop)`
`Pitch(pitch)`
`Position(x, y, z)`
`Velocity(x, y, z)`
`Volume(volume)`

Enum `SourceType`

`static` Decode the entire sound at once.
`stream` Decode the sound gradually.

FILESYSTEM

Module love.filesystem

```
enumerate(dir)
exists(filename)
isDirectory(filename)
isFile(filename)
iterator      = lines(filename)
chunk         = load(filename)
ok            = mkdir(name)
contents, size = read(filename, size)
ok            = remove(filename)
ok            = write(filename, data, size)
```

New Objects

```
newFile(filename)
newFileData(contents, name, decoder)
```

Getters/Setters

```
getAppdataDirectory()
modtime, errmsg = getLastModified(name)
getSaveDirectory()
getUserDirectory()
getWorkingDirectory()
setIdentity(name)
```

Class File

```
ok          = close()
eofReached  = eof()
iterator    = lines()
ok          = open(mode)
contents    = read(bytes)
ok          = seek(pos)
pos         = tell()
ok          = write(data)
getSize()
```

Enum FileMode

r Allows you to (only) read from a file.
w Allows you to (only) write to a file.
a Same as w but data is appended to the end of file.

GRAPHICS CLASSES AND ENUMS

Class Font

```
getHeight()
getWidth(text)
getLineHeight()
setLineHeight(height)
```

Class Framebuffer

```
getImageData()
renderTo(func)
```

Class Image

```
min, mag = getFilter()
getHeight()
getWidth()
horiz, vert = getWrap()
setFilter(min, mag)
setWrap(horiz, vert)
```

Class Quad

```
flip(flipX, flipY)
x, y, w, h = getViewport()
setViewport(x, y, w, h)
```

Class SpriteBatch

```
add(x, y, r, sx, sy, ox, oy)
addq(quad, x, y, r, sx, sy, ox, oy)
clear()
```

Enum BlendMode

additive
alpha
subtractive
multiplicative

Enum ColorMode

modulate Images (etc) are affected by the
 current color.
replace Opposite of modulate.

Enum DrawMode

fill Draw filled shape.
line Draw outlined shape.