Follow the mouse

When you move the mouse, the asset moves towards the cursor.

In this example we are focusing on the movements and we use an asterisk instead of an asset.

At each `update()`, between drawing each frame of the animation, the program calculates the difference between the position of the asset and the cursor. If the distance is larger than 0, the asset moves part of the distance (95% per second) from its current position toward the cursor.

Create a `main.lua` file that:
- Gets the coordinates of the mouse pointer (`target.x`, `target.y`).
- Calculates the `x` and `y` distances (`dx`, `dy`) between the asterisk and the mouse.
- adds 95% of the distance to the current position of the asterisk.

Follow the mouse

When you move the mouse, the asset moves towards the cursor.

In this example we are focusing on the movements and we use an asterisk instead of an asset.

At each `update()`, between drawing each frame of the animation, the program calculates the difference between the position of the asset and the cursor. If the distance is larger than 0, the asset moves part of the distance (95% per second) from its current position toward the cursor.

Create a `main.lua` file that:
- Gets the coordinates of the mouse pointer (`target.x`, `target.y`).
- Calculates the `x` and `y` distances (`dx`, `dy`) between the asterisk and the mouse.
- adds 95% of the distance to the current position of the asterisk.

Follow the mouse

When you move the mouse, the asset moves towards the cursor.

In this example we are focusing on the movements and we use an asterisk instead of an asset.

At each `update()`, between drawing each frame of the animation, the program calculates the difference between the position of the asset and the cursor. If the distance is larger than 0, the asset moves part of the distance (95% per second) from its current position toward the cursor.

Create a `main.lua` file that:
- Gets the coordinates of the mouse pointer (`target.x`, `target.y`).
- Calculates the `x` and `y` distances (`dx`, `dy`) between the asterisk and the mouse.
- adds 95% of the distance to the current position of the asterisk.

Follow the mouse

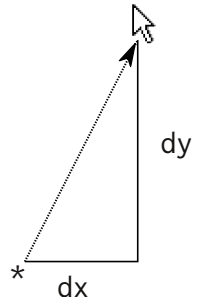
When you move the mouse, the asset moves towards the cursor.

In this example we are focusing on the movements and we use an asterisk instead of an asset.

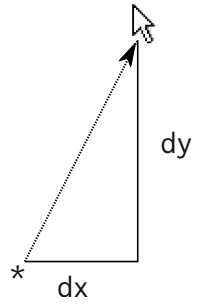
At each `update()`, between drawing each frame of the animation, the program calculates the difference between the position of the asset and the cursor. If the distance is larger than 0, the asset moves part of the distance (95% per second) from its current position toward the cursor.

Create a `main.lua` file that:
- Gets the coordinates of the mouse pointer (`target.x`, `target.y`).
- Calculates the `x` and `y` distances (`dx`, `dy`) between the asterisk and the mouse.
- adds 95% of the distance to the current position of the asterisk.

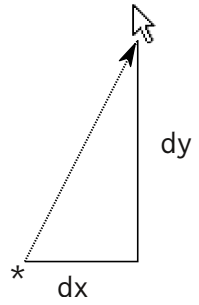
```
star = {x = 175, y = 200, easing = 0.95, char = '*'}

function love.load(arg)
 love.graphics.setFont(love.graphics.newFont(36))
end

function love.update(dt)
  target = {x = 0, y = 0}
  target.x, target.y = love.mouse.getPosition()

  dx = math.floor(target.x - star.x)
  dy = math.floor(target.y - star.y)

  star.x = star.x + (dx * star.easing * dt)
  star.y = star.y + (dy * star.easing * dt)
end

function love.draw()
  love.graphics.print(star.char, star.x, star.y)
end
```



```
star = {x = 175, y = 200, easing = 0.95, char = '*'}

function love.load(arg)
 love.graphics.setFont(love.graphics.newFont(36))
end

function love.update(dt)
  target = {x = 0, y = 0}
  target.x, target.y = love.mouse.getPosition()

  dx = math.floor(target.x - star.x)
  dy = math.floor(target.y - star.y)

  star.x = star.x + (dx * star.easing * dt)
  star.y = star.y + (dy * star.easing * dt)
end

function love.draw()
  love.graphics.print(star.char, star.x, star.y)
end
```



```
star = {x = 175, y = 200, easing = 0.95, char = '*'}

function love.load(arg)
 love.graphics.setFont(love.graphics.newFont(36))
end

function love.update(dt)
  target = {x = 0, y = 0}
  target.x, target.y = love.mouse.getPosition()

  dx = math.floor(target.x - star.x)
  dy = math.floor(target.y - star.y)

  star.x = star.x + (dx * star.easing * dt)
  star.y = star.y + (dy * star.easing * dt)
end

function love.draw()
  love.graphics.print(star.char, star.x, star.y)
end
```



```
star = {x = 175, y = 200, easing = 0.95, char = '*'}

function love.load(arg)
 love.graphics.setFont(love.graphics.newFont(36))
end

function love.update(dt)
  target = {x = 0, y = 0}
  target.x, target.y = love.mouse.getPosition()

  dx = math.floor(target.x - star.x)
  dy = math.floor(target.y - star.y)

  star.x = star.x + (dx * star.easing * dt)
  star.y = star.y + (dy * star.easing * dt)
end

function love.draw()
  love.graphics.print(star.char, star.x, star.y)
end
```