



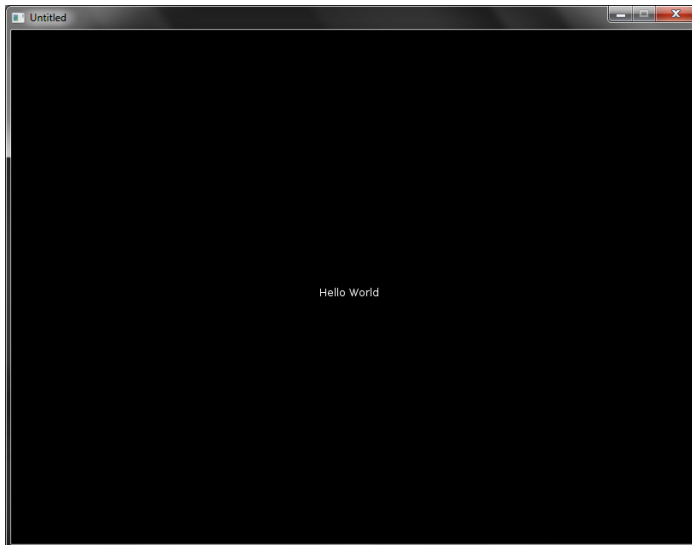
# Jogos Eletrônicos

## Aula 05 – Introdução ao Löve

Edirlei Soares de Lima  
<edirlei.lima@uniriotec.br>

# “Hello World” em Löve

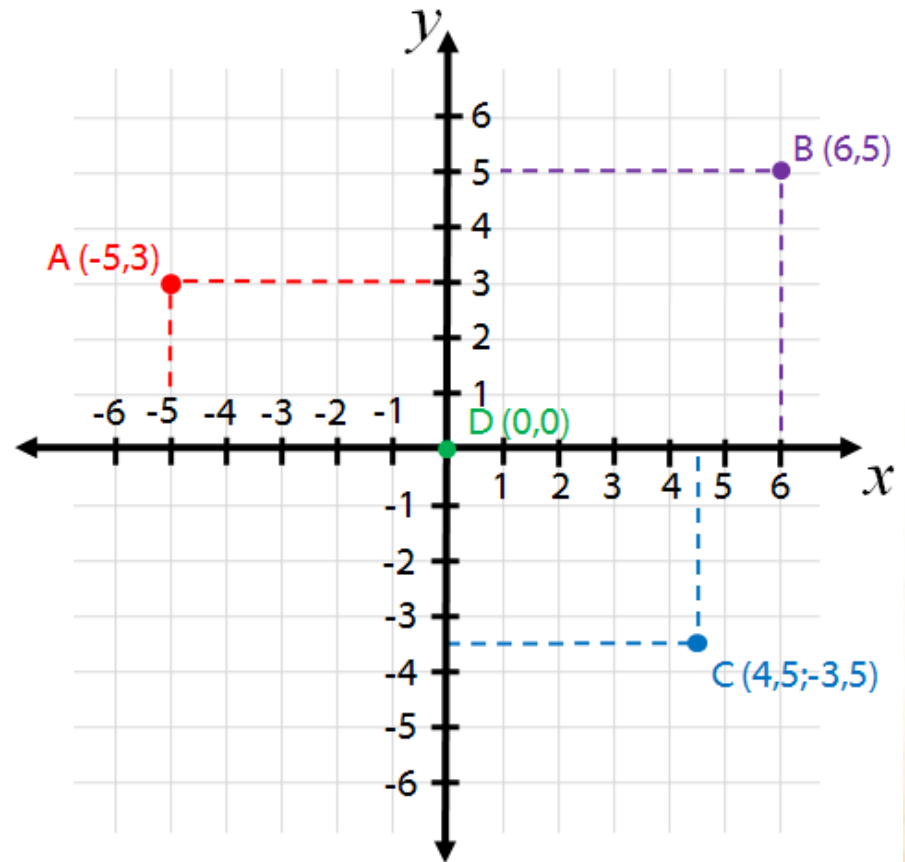
```
function love.draw()  
    love.graphics.print("Hello World", 360, 300)  
end
```



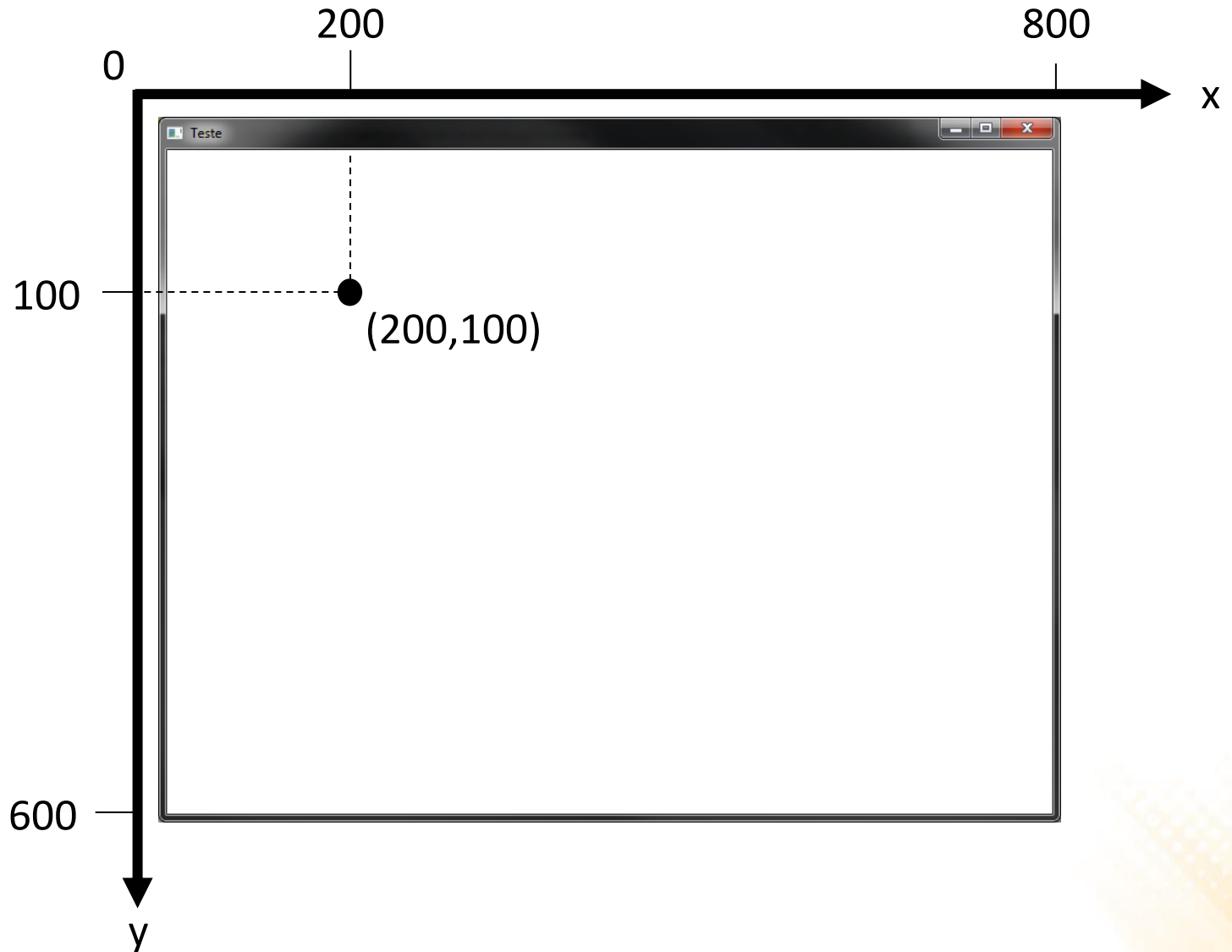
O comando `love.graphics.print` é usado para escrever um texto na tela. Os dois últimos parâmetros representam a posição x e y da tela onde o texto será escrito.

# Coordenadas de Tela

- Sistema de Coordenadas Cartesiano
- Duas dimensões (2D)
- Coordenas X e Y



# Coordenadas de Tela



# Programando em Löve

- Programar em Löve envolve a implementação de funções de **callback**. Essas funções são executadas automaticamente em determinados momentos de acordo com as suas funcionalidades.

- **Exemplo:**

```
love.draw()
```

- A callback `love.draw` é executada continuamente para a renderização dos frames que serão exibidos na tela.

# Löve e Callbacks

- O Löve disponibiliza diversas callbacks que podem ser utilizadas em um jogo para diversas funcionalidades:
  - Inicialização, renderização, atualização, interação pelo teclado/mouse/joystick, etc.
- É importante conhecer e entender o funcionamento das callbacks
  - No decorrer do curso nós vamos explorar as principais callbacks do Löve
- É possível consultar a lista completa de callbacks em:  
<https://www.love2d.org/wiki/Category:Callbacks>

# Callback `love.load()`

- A callback `love.load()` é executada apenas uma vez no momento que o jogo é iniciado.
- A função é geralmente usada para:
  - Carregar recursos (imagens, áudio, etc.)
  - Inicializar variáveis
  - Definir configurações

```
function love.load()  
    image = love.graphics.newImage("cake.jpg")  
    love.graphics.setColor(0,0,0)  
    love.graphics.setBackgroundColor(255,255,255)  
end
```

# De Volta ao “Hello World”

```
function love.load()  
    love.graphics.setColor(0,0,0)  
    love.graphics.setBackgroundColor(255,255,255)  
end  
  
function love.draw()  
    love.graphics.print("Hello World", 360, 300)  
end
```



O comando `love.graphics.setColor` é usado para definir a cor usada para desenhar e escrever na tela (modelo RGB)

O comando `love.graphics.setBackgroundColor` é usado para definir a cor de fundo da tela (modelo RGB)



# Modelo de Cores RGB

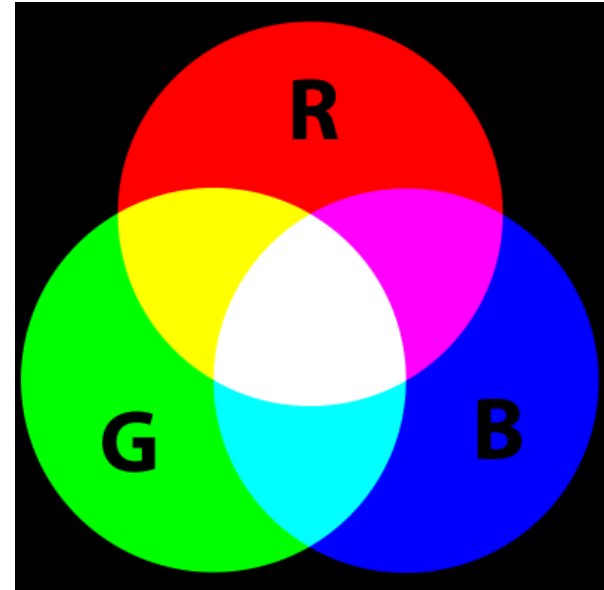
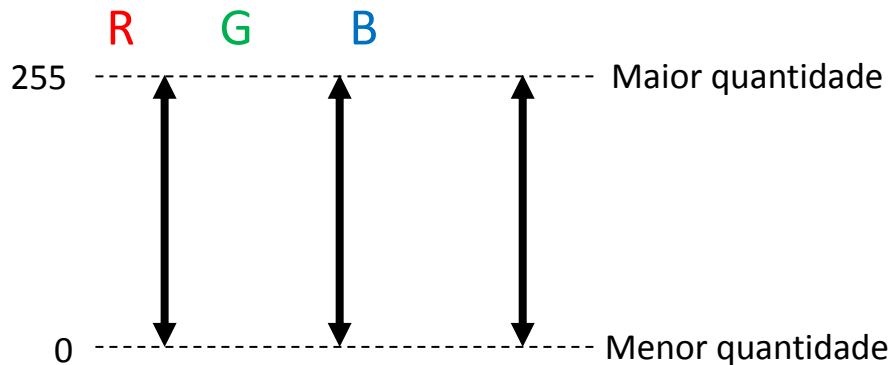
- **Formato de cor RGB:**

**R** = Red (Vermelho)

**G** = Green (Verde)

**B** = Blue (Azul)

- **Escala RGB:**



Não sabe o valor RGB da cor que  
você quer?

<http://www.colorpicker.com/>

# Callback `love.update(dt)`

- A callback `love.update(dt)` é continuamente executada em loop enquanto o jogo estiver aberto. O parâmetro `dt` indica o tempo que se passou desde a última vez que essa função foi chamada (usualmente um valor bem pequeno como 0.015714)
- A função é geralmente usada para:
  - Animação
  - Cálculos de física
  - Inteligência artificial de inimigos

Calcula o deslocamento em X de forma independente da velocidade de execução do programa

```
function love.update(dt)
  px = px + (100 * dt)
end
```

# De Volta ao “Hello World”

```
local px      -- posição x do texto

function love.load()
    love.graphics.setColor(0, 0, 0)
    love.graphics.setBackgroundColor(255, 255, 255)
    px = 0
end

function love.update(dt)
    px = px + (100 * dt)
end

function love.draw()
    love.graphics.print("Hello World", px, 300)
end
```

# Módulos Löve

- O Löve é dividido em **módulos**:
  - Cada módulo possui um conjunto de funções e tipos de dados que podem ser utilizados.
- Todos os módulos estão contidos em um **módulo global** chamado `love`.
- **Exemplo de módulo:** `love.graphics`
  - Nos exemplos anteriores nós utilizamos algumas funções do módulo `love.graphics`
  - A função `love.graphics.print` pertence ao módulo `love.graphics`

# Módulo `love.graphics`

- O módulo `love.graphics` contém funções dedicadas a operações gráficas:
  - Desenho de linhas, formas geométricas, texto, imagens, etc.
  - Carregar arquivos externos (imagens, fontes, etc.) para a memória.
  - Criar objetos especiais (sistemas de partículas, canvas, etc.)
  - Manipular a tela
- É possível consultar a lista completa de funções do módulo `love.graphics` no seguinte endereço:

<http://love2d.org/wiki/love.graphics>

# Módulo `love.graphics`

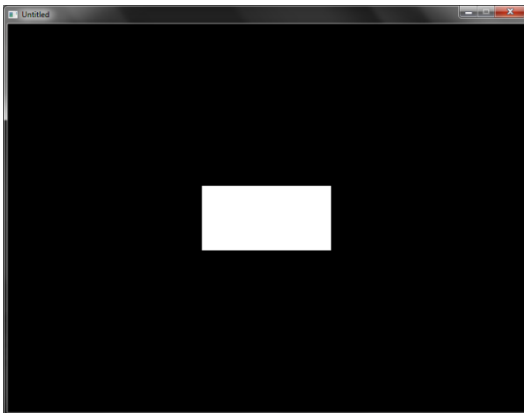
- Desenhando **formas geométricas** básicas:

- **Retângulo:**

```
love.graphics.rectangle(mode, x, y, width, height)
```

Exemplo:

```
love.graphics.rectangle("fill", 300, 250, 200, 100)
```



mode: "fill" para desenhar a forma preenchida ou "line" para desenhar somente os contornos.

# Módulo `love.graphics`

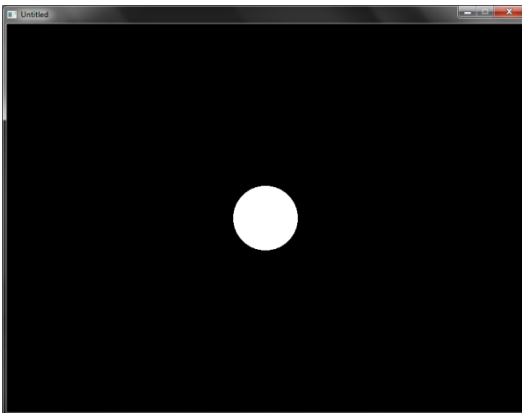
- Desenhando **formas geométricas** básicas:

- **Círculo:**

```
love.graphics.circle(mode, x, y, radius, segments)
```

Exemplo:

```
love.graphics.circle("fill", 400, 300, 50, 100)
```



Número de segmentos usados para desenhar um círculo.

# Módulo `love.graphics`

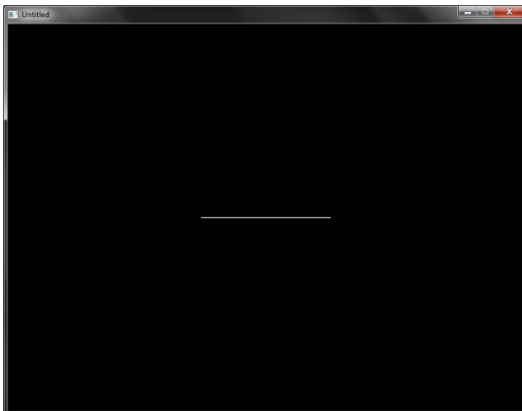
- Desenhando **formas geométricas** básicas:

- **Linha:**

```
love.graphics.line(x1, y1, x2, y2, ...)
```

Exemplo:

```
love.graphics.line(300, 300, 500, 300)
```



É possível passar mais pontos como parâmetro.



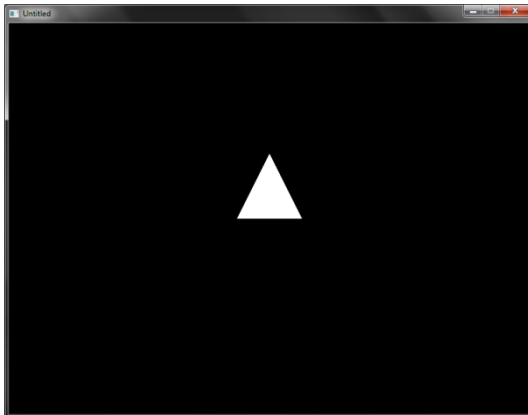
# Módulo `love.graphics`

- Desenhando **formas geométricas** básicas:
  - **Polígono:**

```
love.graphics.polygon(mode, ...)
```

Exemplo:

```
love.graphics.polygon("fill", 350, 300, 450, 300, 400, 200)
```



É possível passar mais pontos como parâmetro.

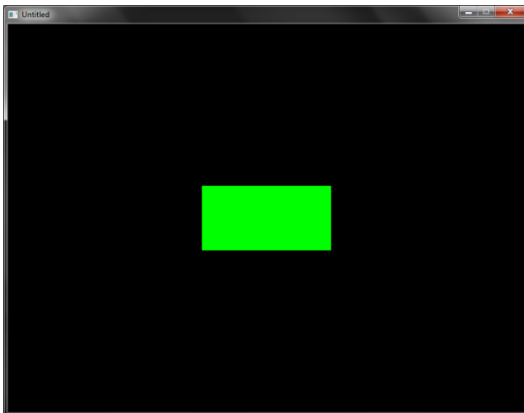
# Módulo `love.graphics`

- Desenhando **formas geométricas** básicas:
  - **Modificando a cor das formas geométricas:**

```
love.graphics.setColor(red, green, blue, alpha)
```

Exemplo:

```
love.graphics.setColor(0, 255, 0)  
love.graphics.rectangle("fill", 300, 250, 200, 100)
```



O parâmetro `alpha` é opcional e pode ser utilizado para definir cores com transparência.

# Exemplo 1 – Primitivas Geométricas

```
function love.draw()  
  
  -- desenha retangulo  
  love.graphics.setColor(0, 134, 0)  
  love.graphics.rectangle("fill", 100, 100, 600, 400)  
  
  -- desenha losango  
  love.graphics.setColor(252, 252, 0)  
  love.graphics.polygon("fill", 120, 300, 400, 120,  
                        680, 300, 400, 480)  
  
  -- desenha circulo  
  love.graphics.setColor(0, 0, 140)  
  love.graphics.circle("fill", 400, 300, 120, 100)  
  
end
```

# Exemplo 1 – Primitivas Geométricas



# Exercício 1

**1)** Usando as primitivas básicas (linha, retângulo, círculo, polígono) crie um programa que desenhe um cenário semelhante ao mostrado na figura abaixo:

