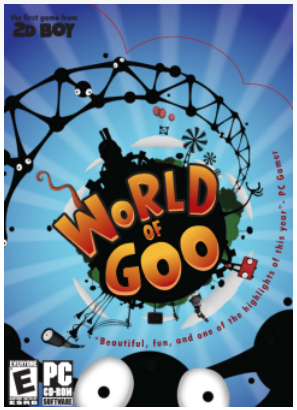
SDL

- *Simple DirectMedia Layer*
- Acesso “baixo nível” ao áudio, teclado, mouse, joystick, gráficos, etc.
- Jogos, vídeos e aplicações gráficas
- Multi-plataforma:
  - *Windows, Linux, MacOS*
  - *Android, iOS*
- Implementado em C
  - Bibliotecas para *Python*, *Lua*, etc.
  - Game engines: *Pygame*, *Löve*

Jogos

- Código aberto
  - *Freeciv, SuperTux*
- Independente (Indie)
  - *World of Goo, FEZ*
- Profissional (AAA)
  - *Quake 4, Unreal*



Artigos & Videos - 04

- Open Source Tools for Game Development
  - `https://www.youtube.com/watch?v=r3wDnOAjrtk`
- Indie Game: The Movie
  - `http://vimeo.com/ondemand/indiegamethemovie/25268139`

Hello World

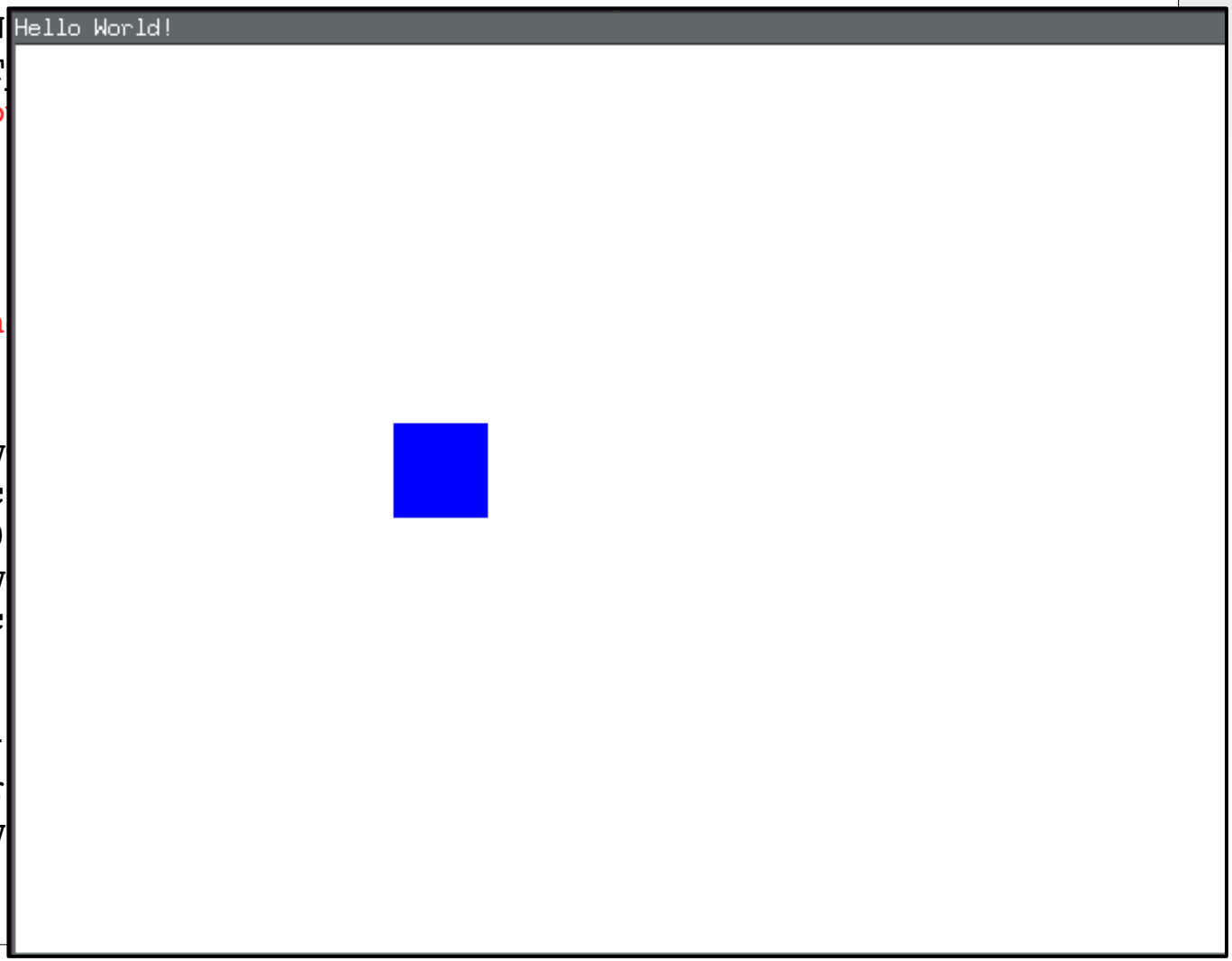
```
#include <SDL2/SDL.h>
int main (int argc, char* args[])
{
    /* INITIALIZATION
    SDL_Init(SDL_INIT
    SDL_Window* windo




    SDL_Renderer* ren

    /* EXECUTION */
    SDL_SetRenderDraw
    SDL_RenderFillRec
    SDL_Rect r = { 20
    SDL_SetRenderDraw
    SDL_RenderFillRec
    SDL_Delay(5000);

    /* FINALIZATION *
    SDL_DestroyRender
    SDL_DestroyWindow
    SDL_Quit();
}
```



Hello World - Input

```
    /* INITIALIZATION */

    /* EXECUTION */
    SDL_Rect r = { 200,200, 50, 50 };
    SDL_Event e;
    while (1) {
        while (SDL_PollEvent(&e) == 0);
        if (e.type == SDL_QUIT) {
            break;
        } else if (e.type == SDL_KEYDOWN) {
            switch (e.key.keysym.sym) {
                case SDLK_UP:
                    r.y -= 10;
                case SDLK_DOWN:
                    r.y += 10;
                case SDLK_LEFT:
                    r.x -= 10;
                case SDLK_RIGHT:
                    r.x += 10;
            }
        }
        SDL_SetRenderDrawColor(renderer, 0xFF,0xFF,0xFF,0x00);
        SDL_RenderFillRect(renderer, NULL);
        SDL_SetRenderDrawColor(renderer, 0x00,0x00,0xFF,0x00);
        SDL_RenderFillRect(renderer, &r);
        SDL_RenderPresent(renderer);
    }

    /* FINALIZATION */
```

O “loop” de eventos

```
/* INITIALIZATION */

/* EXECUTION */
while (1) {
    /* events  */
    /* logic   */
    /* redraw  */
}

/* FINALIZATION */
```

- Modelo síncrono

O “loop” de eventos

```
/* INITIALIZATION */

/* EXECUTION */
while (1) {
    /* events  */
    SDL_PollEvent(&e);
    SDL_WaitEvent(&e);
    SDL_WaitEventTimeout(&e, 25);

    /* logic   */
    /* redraw  */
}

/* FINALIZATION */
```

Exercício - “Animação”

- Mover dois retângulos “em quadrados” com velocidades diferentes
- Parar o retângulo quando clicado com o mouse

SDL_GetTicks

Use this function to get the number of milliseconds since the SDL library initialization.

Contents
- SDL_GetTicks
  - Syntax
  - Return Value
  - Code Examples
  - Remarks
  - Related Functions

Syntax

Toggle line numbers

```
Uint32 SDL_GetTicks(void)
```

Return Value

Returns an unsigned 32-bit value representing the number of milliseconds since the SDL library initialized.

```
switch (e.type) {
  <...>
  SDL_MOUSEBUTTONDOWN:
    SDL_MouseButtonEvent* me =
     (SDL_MouseButtonEvent*) &e;
    me->x;
    me->y;
    <...>
    break;
}
```

Animações

```
int speed = 10;

while (1) {
    SDL_PollEvent(&e);

    // eventos
    switch (e.type) {
        <...>
    }

    // animacoes
    x += speed;

    // redesenho
    <...>
}
```

```
int speed = 10;

while (1) {
    SDL_PollEvent(&e);

    // eventos
    switch (e.type) {
        <...>
    }

    // animacoes
    x += DT*speed;
          // DT: diferenca de tempo
          // entre dois frames (SDL_Ticks)

    // redesenho
    <...>
}
```

Compilando / Executando

```
# clone
> git clone https://github.com/fsantanna-uerj/reativos

# update
> cd reativos/
> git pull

# compile
> cd reativos/code/sdl
> gcc 00_hello.c -lSDL2 -o 00_hello   # (01_input.c 01_input)

# run
> ./00_hello                          # (./01_input)
```

Documentação

- Lazy Foo Tutorial
  - `http://lazyfoo.net/tutorials/SDL/index.php`
- Manual de referência
  - `https://wiki.libsdl.org/APIByCategory`
- Fórum do SDL
  - `http://forums.libsdl.org/`