

# Programação com a linguagem Céu

## Movimentar um pixel

**Anny Caroline**

`annycarolinegnr@gmail.com`

**Francisco Sant'Anna**

`francisco@ime.uerj.br`

# Player

- Para começar, vamos criar uma abstração, chamada `Player`

```
code/await Player(none) -> none do  
  
end  
  
spawn Player();  
await FOREVER;
```

# Teclas

- Desenhar um pixel após pressionar uma tecla do teclado

```
code/await Player(none) -> none do  
    await KEY_PRESS;  
    emit GRAPHICS_DRAW_PIXEL(0,0);  
end
```

```
spawn Player();  
await FOREVER;
```

1-KEY\_PRESS.ceu

# Teclas

- Desenhar um pixel após pressionar a tecla “a”

# Teclas

## ■ Solução 1

```
code/await Player(none) -> none do
  var int key = await KEY_PRESS;
  if (key==KEY_a) then
    emit GRAPHICS_DRAW_PIXEL(0,0);
  end
end

spawn Player();
await FOREVER;
```

# Teclas

## ■ Solução 2

```
code/await Player(none) -> none do
    var int key = await KEY_PRESS until key==KEY_a;
    emit GRAPHICS_DRAW_PIXEL(0,0);
end

spawn Player();
await FOREVER;
```

# Teclas

- lista de teclas

```
var int key = await KEY_PRESS until key == KEY_a;
```

```
var int key = await KEY_PRESS;  
if (key == KEY_a) then  
    emit GRAPHICS_DRAW_PIXEL(0,0);  
end
```

# Perguntas

- Há uma pequena diferença entre as duas soluções. Qual é?



# Every

```
code/await Player(none) -> NEVER do
```

```
    var int i=0;
```

```
    var int key;
```

```
    every key in KEY_PRESS do
```

```
        emit GRAPHICS_DRAW_PIXEL(i,i);
```

```
        i = i+1;
```

```
    end
```

```
end
```

```
spawn Player();
```

```
await FOREVER;
```

# Every

```
code/await Player(none) -> NEVER do
```

```
  var int x=0;
```

```
  var int y=0;
```

```
  var int key;
```

```
  every key in KEY_PRESS do
```

```
    if (key==KEY_UP) then
```

```
      emit GRAPHICS_DRAW_PIXEL(x,y);
```

```
      y = y+1;
```

```
    end
```

```
  end
```

```
end
```

# Exercício

- Modifique o exemplo anterior para movimentar o pixel ao longo da tela
- Utilize as teclas
  - KEY\_UP
  - KEY\_RIGHT
  - KEY\_DOWN
  - KEY\_LEFT
- Comportamento esperado

```
if Exp then
    //Block
else/if Exp then
    //Block
else
    //Block
end
```

# Exercício

- limpar a última posição do pixel

# FRAMES\_REDRAW

- limpar a última posição do pixel
- `FRAMES_SET`
  - Habilita ou desabilita a geração periódica de eventos `FRAMES_UPDATE` e `FRAMES_REDRAW` para a aplicação
- `emit FRAMES_SET(yes);`

# FRAMES\_REDRAW

- limpar a última posição do pixel
- `FRAMES_SET`
- Habilita ou desabilita a geração periódica de eventos `FRAMES_UPDATE` e `FRAMES_REDRAW` para a aplicação

- ```
every FRAMES_REDRAW do
  //...
end
```

```
emit FRAMES_SET(yes);

code/await Player(none) -> NEVER do
    var Point pt = val Point(0,0);

    par do
        var int key;
        every key in KEY_PRESS do
            //atualiza pt
        end
    with
        every FRAMES_REDRAW do
            emit GRAPHICS_SET_COLOR_NAME(COLOR_WHITE);
            emit GRAPHICS_DRAW_PIXEL(pt.x, pt.y);
        end
    end
end

end
```

```
emit FRAMES_SET(yes);
```

```
code/await Player(none) -> NEVER do
```

```
  var Point pt = val Point(0,0);
```

```
  par do
```

```
    var int key;
```

```
    every key in KEY_PRESS do
```

```
      //atualiza pt
```

```
    end
```

```
  with
```

```
    every FRAMES_REDRAW do
```

```
      emit GRAPHICS_SET_COLOR_NAME(COLOR_WHITE);
```

```
      emit GRAPHICS_DRAW_PIXEL(pt.x, pt.y);
```

```
    end
```

```
  end
```

```
end
```

```
  if key == KEY_LEFT then
```

```
    pt.x = pt.x - 1;
```

```
  else/if key == KEY_RIGHT then
```

```
    pt.x = pt.x + 1;
```

```
  else/if key == KEY_UP then
```

```
    pt.y = pt.y + 1;
```

```
  else/if key == KEY_DOWN then
```

```
    pt.y = pt.y - 1;
```

```
  end
```



# Exercício

- limitar a movimentação do pixel à janela

# Exercício

- movimentar dois pixels ao mesmo tempo
- usar um conjunto de teclas diferentes
  - como se fossem dois jogadores