Programação com a linguagem Céu

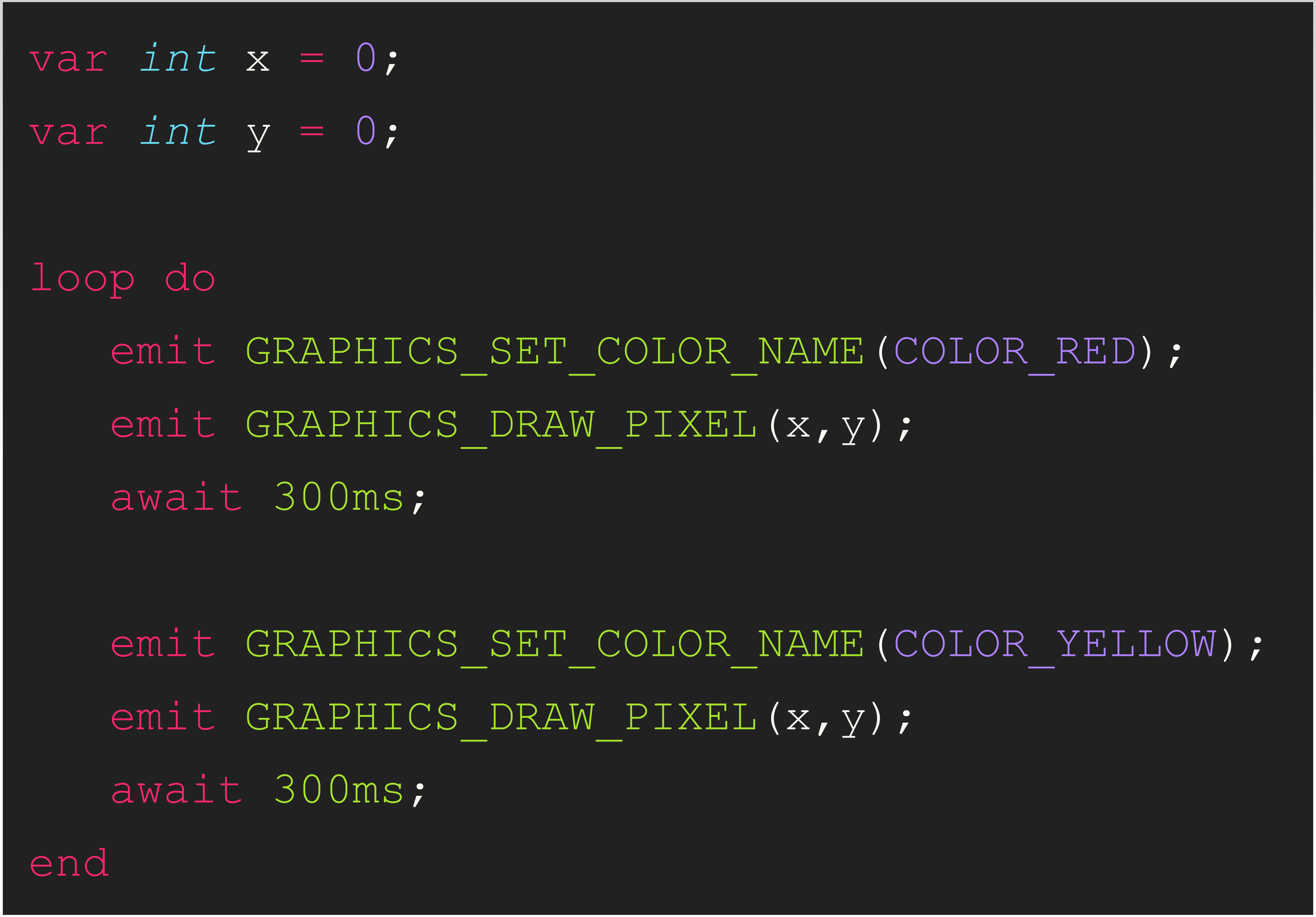
code/await - pixels piscando

Anny Caroline
annycarolinegnr@gmail.com

Francisco Sant'Anna
francisco@ime.uerj.br

Pixels piscando

- pixel que muda de cor com o passar do tempo (piscando)



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

loop do
   emit GRAPHICS_SET_COLOR_NAME(COLOR_RED);
   emit GRAPHICS_DRAW_PIXEL(x,y);
   await 300ms;

   emit GRAPHICS_SET_COLOR_NAME(COLOR_YELLOW);
   emit GRAPHICS_DRAW_PIXEL(x,y);
   await 300ms;
end
```

Pixels piscando

- E se fossem 2 pixels?

```
par do
   var int x = 0;
   var int y = 0;

   loop do
       emit GRAPHICS_SET_COLOR_NAME(COLOR_RED);
       emit GRAPHICS_DRAW_PIXEL(x,y);
       await 300ms;

       emit GRAPHICS_SET_COLOR_NAME(COLOR_YELLOW);
       emit GRAPHICS_DRAW_PIXEL(x,y);
       await 300ms;
   end
with
   var int x = 10;
   var int y = 10;

   loop do
       emit GRAPHICS_SET_COLOR_NAME(COLOR_RED);
       emit GRAPHICS_DRAW_PIXEL(x,y);
       await 300ms;

       emit GRAPHICS_SET_COLOR_NAME(COLOR_YELLOW);
       emit GRAPHICS_DRAW_PIXEL(x,y);
       await 300ms;
   end
end
```

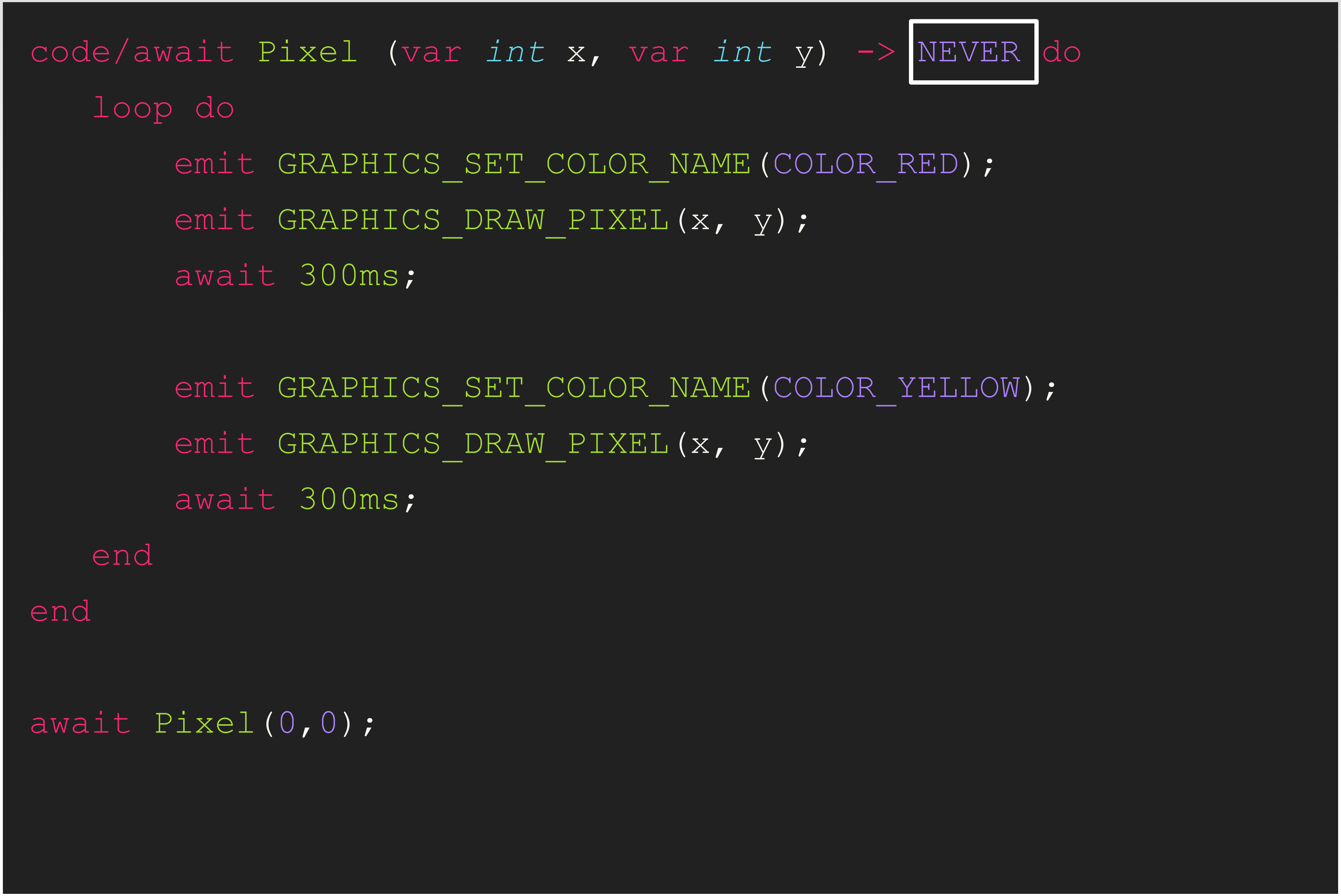
Pixels piscando

- E se fossem 10?

Pixels piscando

- E se fossem 10?
- Podemos usar um `code/await`

code/await

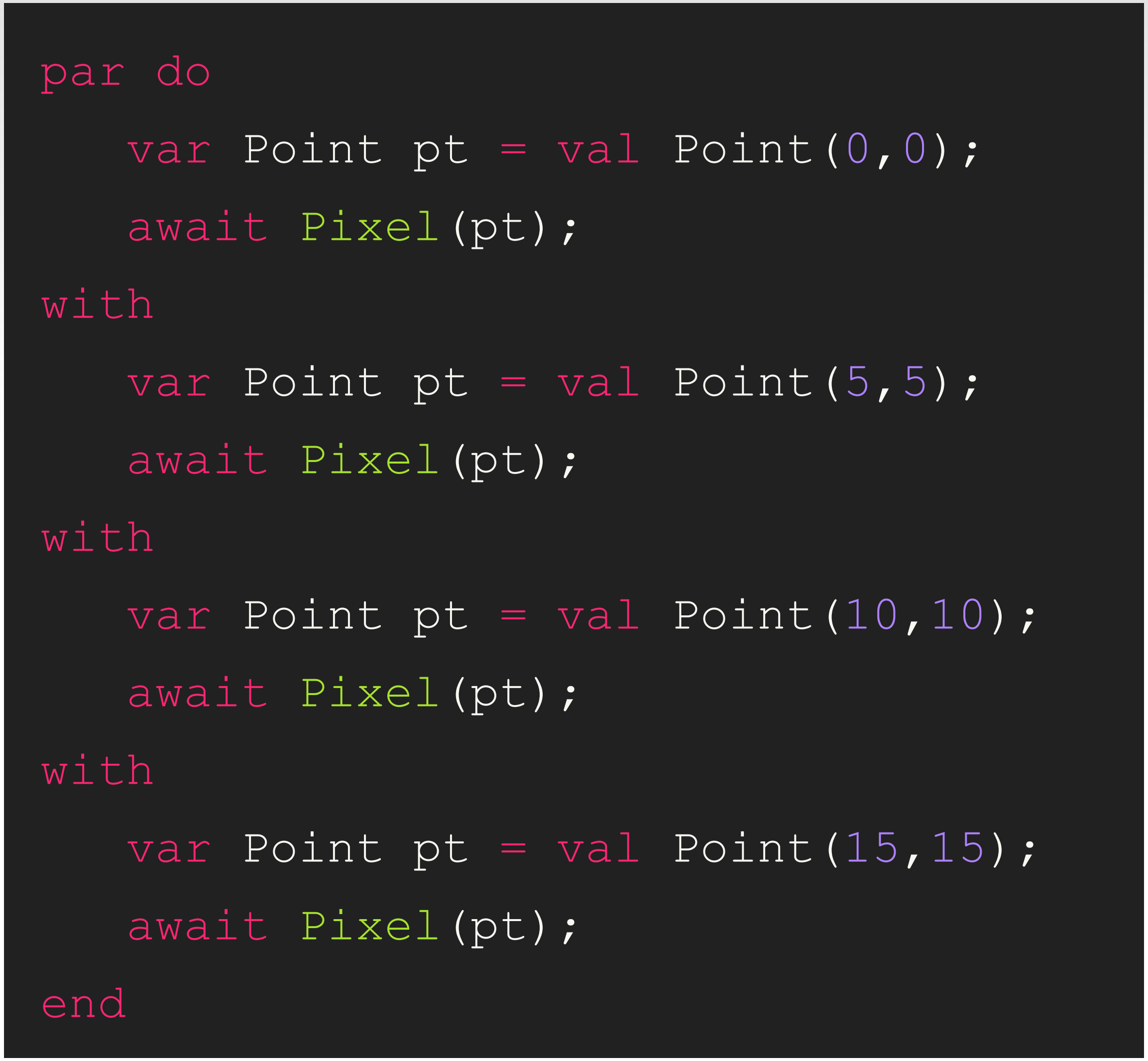


```
code/await Pixel (var int x, var int y) -> NEVER do
   loop do
       emit GRAPHICS_SET_COLOR_NAME(COLOR_RED);
       emit GRAPHICS_DRAW_PIXEL(x, y);
       await 300ms;

       emit GRAPHICS_SET_COLOR_NAME(COLOR_YELLOW);
       emit GRAPHICS_DRAW_PIXEL(x, y);
       await 300ms;
   end
end

await Pixel(0,0);
```

code/await

```
code/await Pixel (var Point pt) -> NEVER do
    loop do
        emit GRAPHICS_SET_COLOR_NAME(COLOR_RED);
        emit GRAPHICS_DRAW_PIXEL(pt.x, pt.y);
        await 300ms;

        emit GRAPHICS_SET_COLOR_NAME(COLOR_YELLOW);
        emit GRAPHICS_DRAW_PIXEL(pt.x, pt.y);
        await 300ms;
    end
end

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

await + par



```
par do
    var Point pt = val Point(0,0);
    await Pixel(pt);
with
    var Point pt = val Point(5,5);
    await Pixel(pt);
with
    var Point pt = val Point(10,10);
    await Pixel(pt);
with
    var Point pt = val Point(15,15);
    await Pixel(pt);
end
```

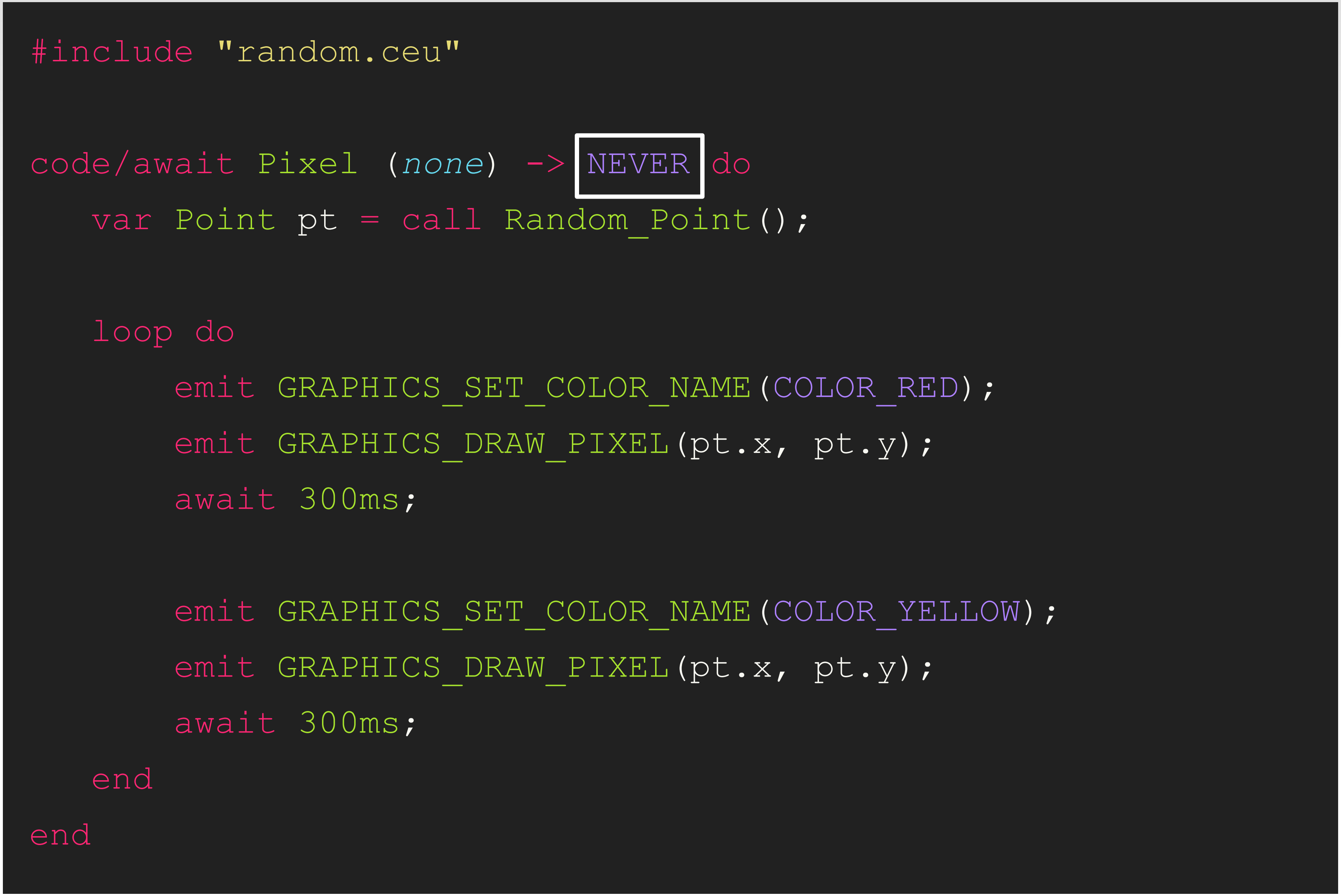
Perguntas

- O exemplo anterior funcionaria com
  - `par/or`

Perguntas

- O exemplo anterior funcionaria com
  - `par/or`
  - `par/and`

code/await



```
#include "random.ceu"

code/await Pixel (none) -> NEVER do
    var Point pt = call Random_Point();

    loop do
        emit GRAPHICS_SET_COLOR_NAME(COLOR_RED);
        emit GRAPHICS_DRAW_PIXEL(pt.x, pt.y);
        await 300ms;

        emit GRAPHICS_SET_COLOR_NAME(COLOR_YELLOW);
        emit GRAPHICS_DRAW_PIXEL(pt.x, pt.y);
        await 300ms;
    end
end
```

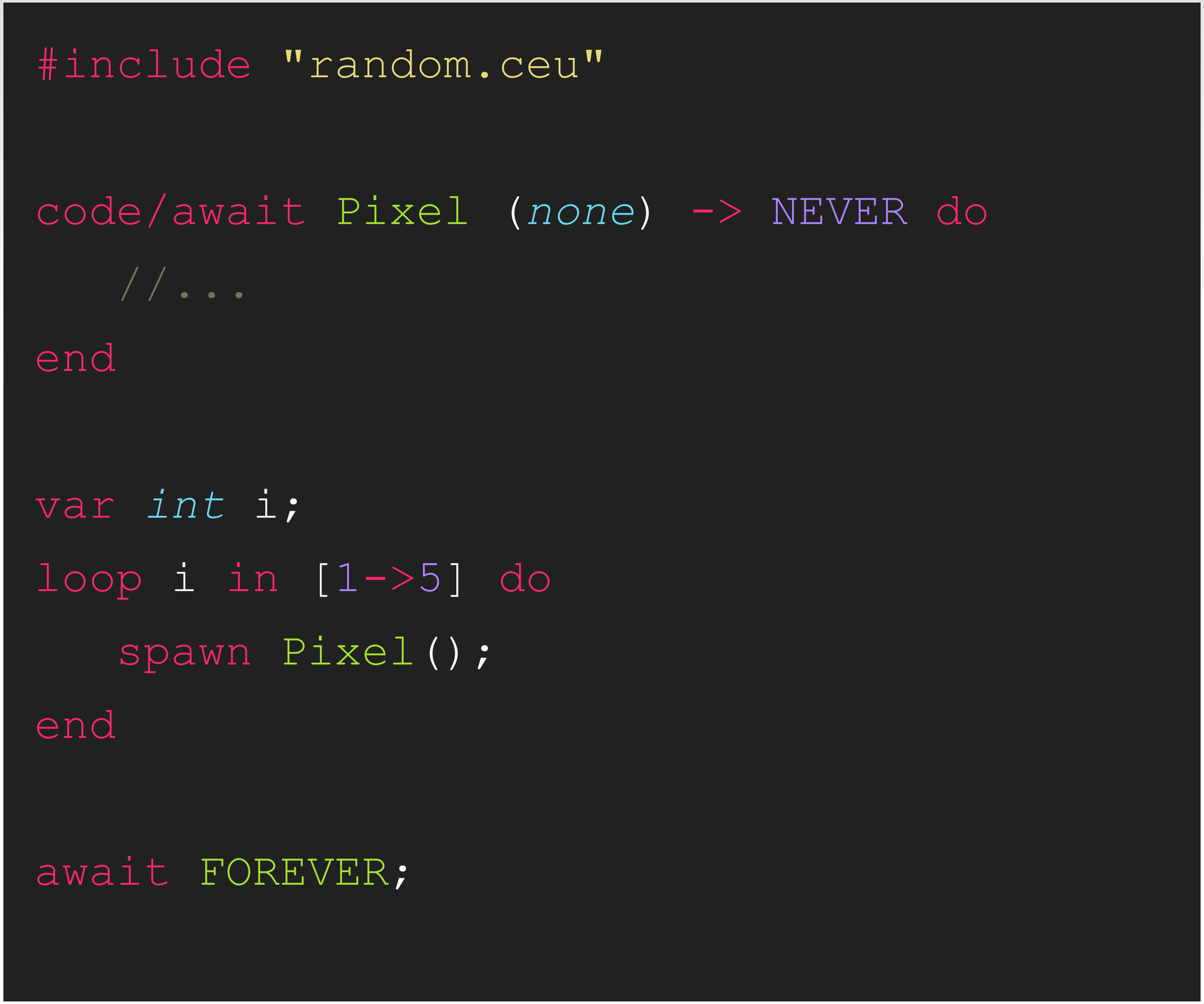
spawn

- A instrução `spawn` começa a executar um bloco em paralelo ao bloco “pai”
- Quando o bloco “pai” termina, o bloco gerado é abortado

spawn

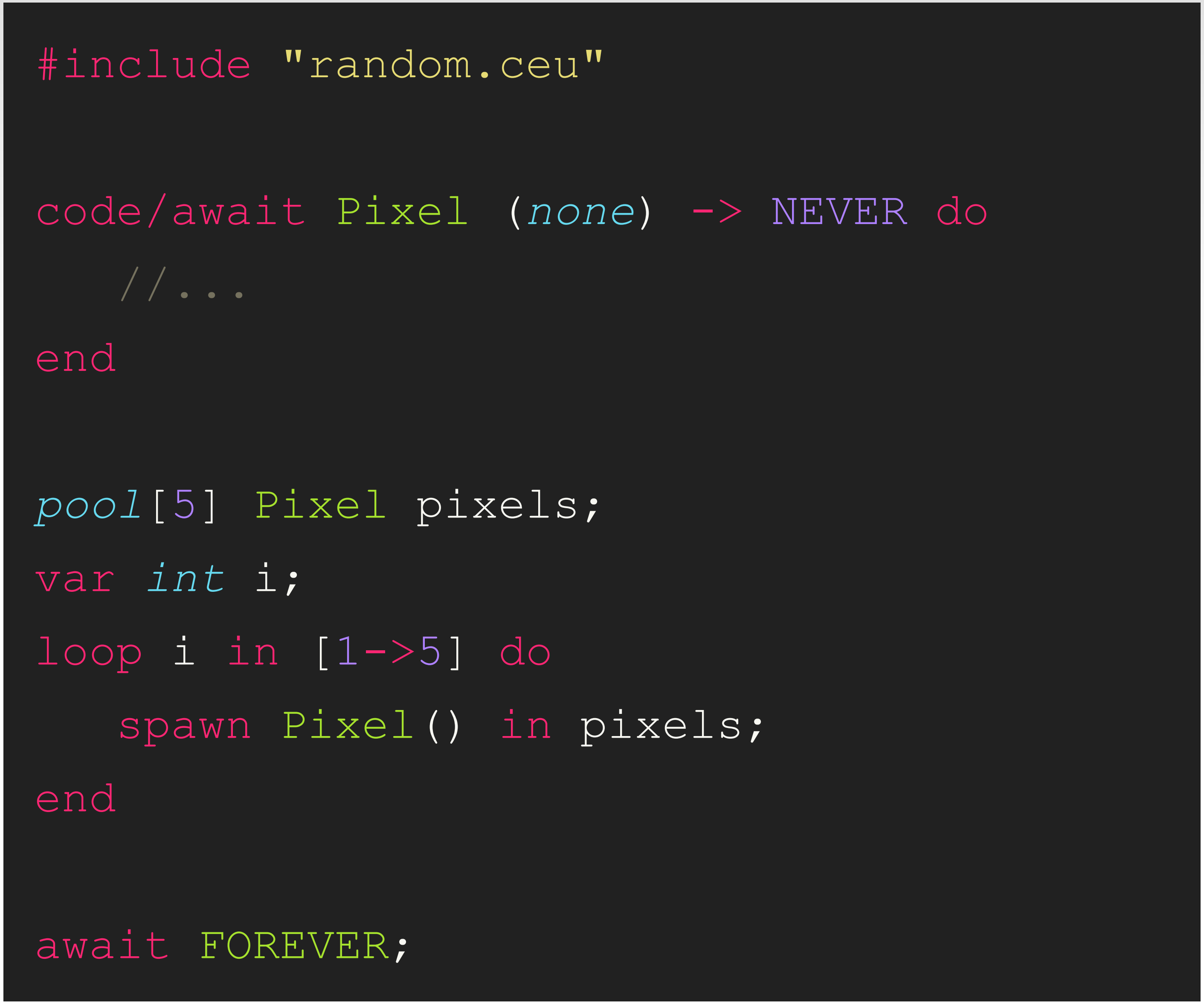
```
spawn Pixel();
spawn Pixel();
spawn Pixel();
spawn Pixel();
spawn Pixel();


await FOREVER;
```

spawn - usando um loop



```
#include "random.ceu"

code/await Pixel (none) -> NEVER do
    //...
end

var int i;
loop i in [1->5] do
    spawn Pixel();
end

await FOREVER;
```

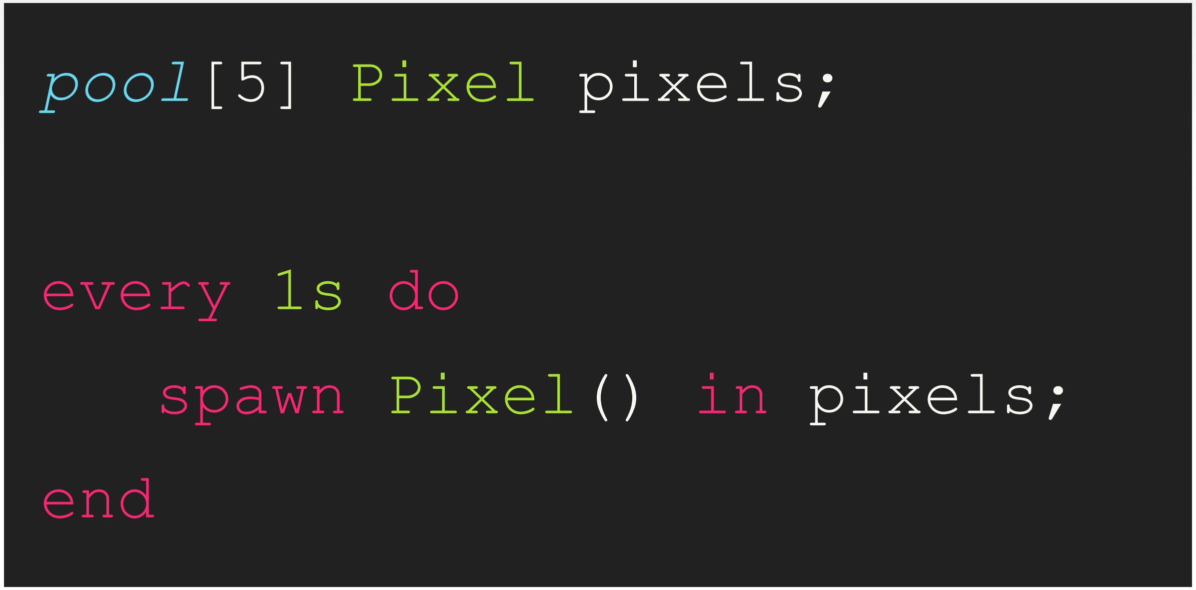
spawn - usando um loop



```
#include "random.ceu"

code/await Pixel (none) -> NEVER do
    //...
end

pool[5] Pixel pixels;
var int i;
loop i in [1->5] do
    spawn Pixel() in pixels;
end

await FOREVER;
```

Pixels piscando

- E se fosse necessário criar 1 pixel a cada segundo?



```
pool[5] Pixel pixels;

every 1s do
    spawn Pixel() in pixels;
end
```

Pixels piscando

- E se o `pool` fosse ilimitado?

```
pool[] Pixel pixels;

every 1s do
    spawn Pixel() in pixels;
end
```

Exercício

- Com um clique do mouse, parar a execução de todos os pixels

Exercício

- Modifique o exemplo anterior para limpar a tela após o clique do mouse