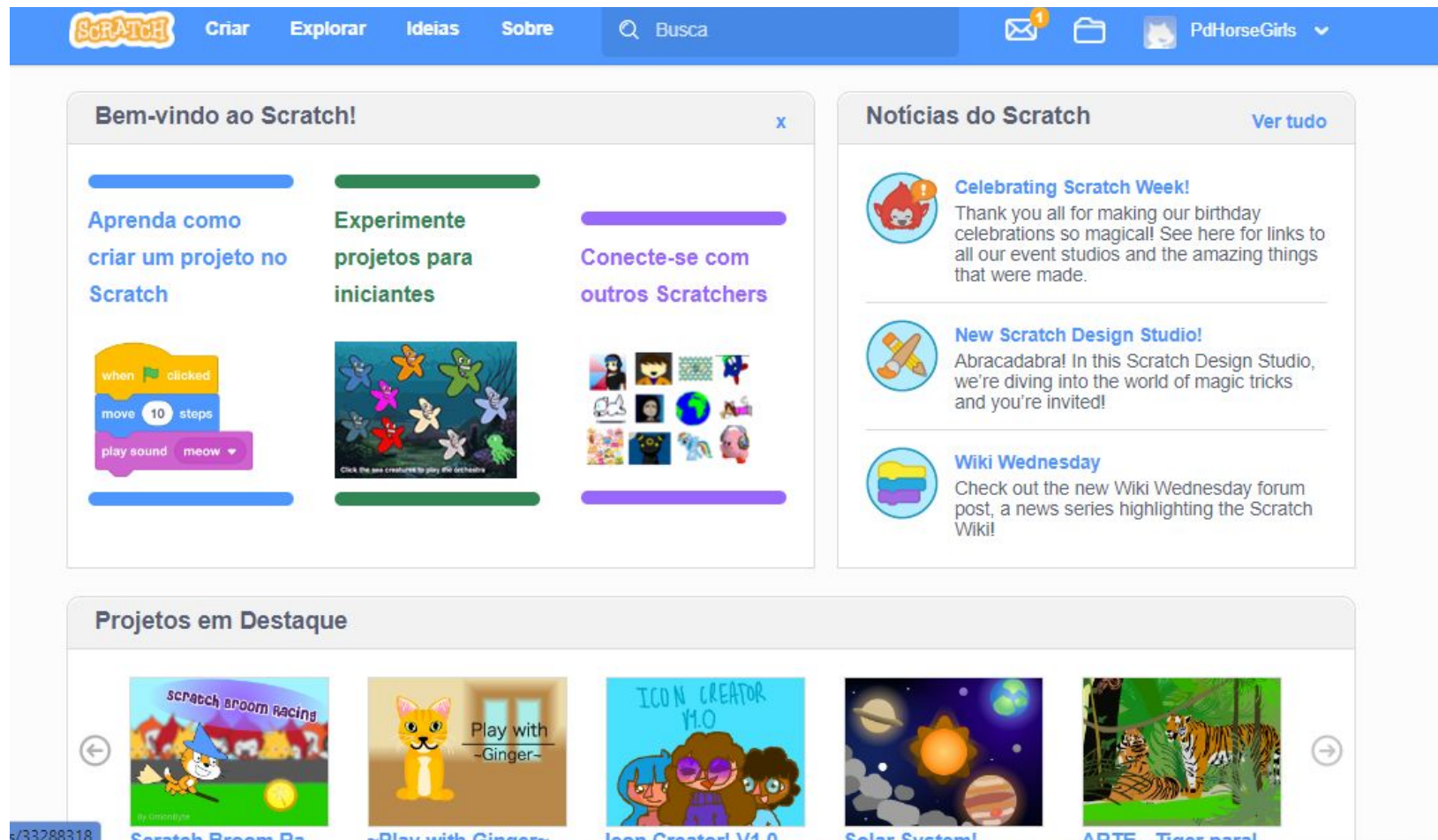


Java(Modularização)

Patrícia Dourado

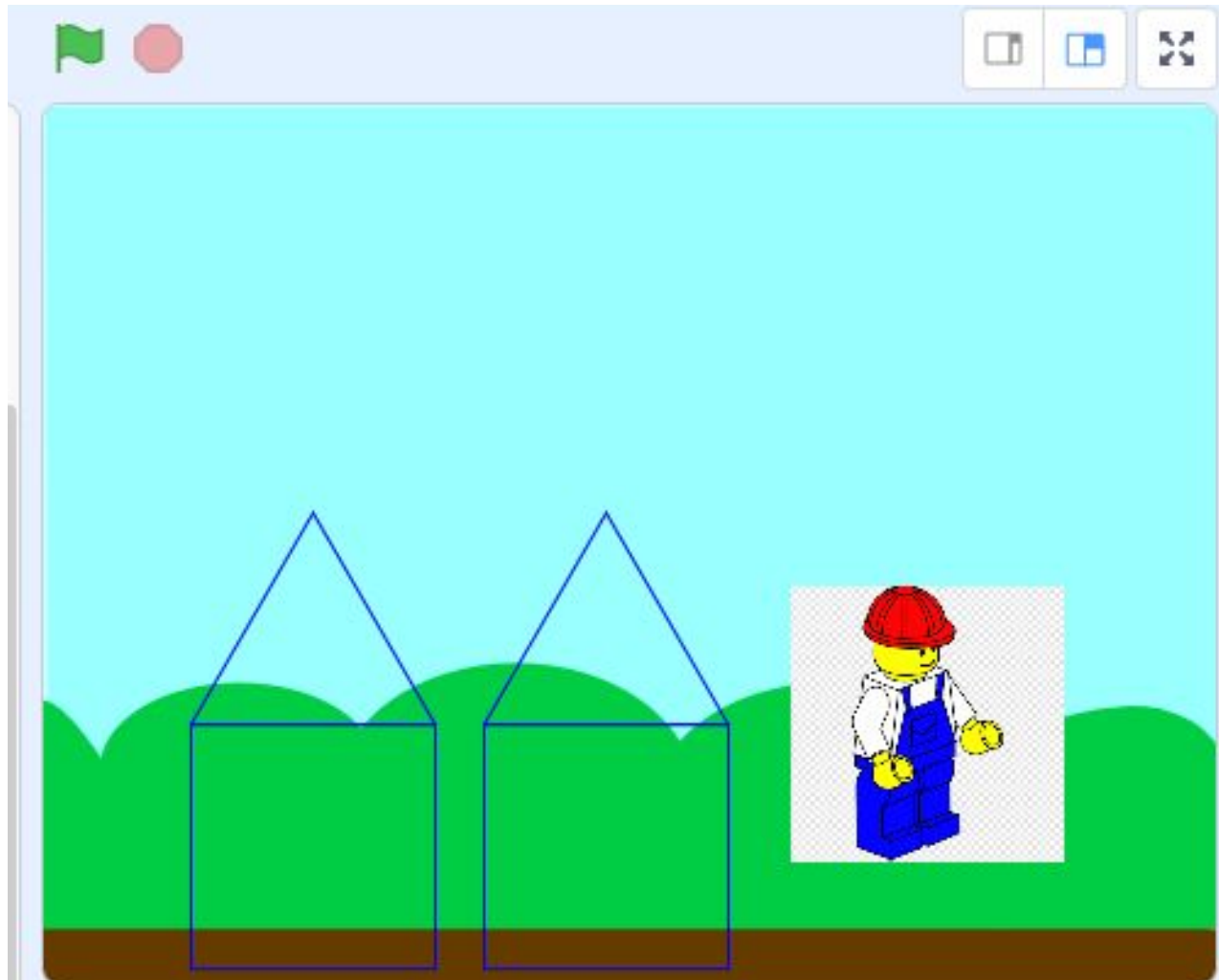


Projeto em Scratch!




<https://scratch.mit.edu/>

Scratch – construir casa!



quando  for clicado

 apague tudo

 use a caneta

Casa

 levante a caneta

mova 120 passos

 use a caneta

Casa

defina Casa

quadrado

triangulo

defina triangulo

repita 3 vezes

mova 100 passos

espere 1 seg

gire  120 graus

espere 1 seg

defina quadrado

repita 4 vezes

 use a caneta

mova 100 passos

espere 1 seg

gire  90 graus

espere 1 seg



Modularização

Técnica de programação que consiste em decompor a solução de um problema em blocos, chamados módulos, que interagem com um bloco principal.

[Xavier, G.F.C. Lógica de programação. São Paulo: Editora SENAC, 1998.]

Módulos:

“Conjunto de comandos que constitui uma parte de um algoritmo principal e que tem uma tarefa bem-definida e independente em relação ao resto do algoritmo.”

[Xavier, G.F.C. Lógica de programação. São Paulo: Editora SENAC, 1998.]

Características desejadas

- Módulos devem ter finalidade bem definida
- Independência do módulo em relação ao resto do programa
- Implementação em época diferente do restante do programa
- Manutenção mais simples, evitando efeitos colaterais
- Reuso de módulos em sistemas diferentes exigindo pouca ou nenhuma adaptação

[Xavier, G.F.C. Lógica de programação. São Paulo: Editora SENAC, 1998.]

Tipos de Módulo

- **Módulo principal** → Todo programa deve ter um módulo principal a partir do qual o programa começa a ser executado. Os demais módulos somente serão executados se forem “chamados” a partir do módulo principal.
- **Procedimento** → módulo que não retorna valor.
- **Função** → módulo que retorna valor no nome da função.

Implementação de módulos em Java

```
public static <tipo> <identificador> (<parâmetros>){  
    <Bloco de instruções>  
}
```

- Exemplo:

```
public static void mensagem() {  
    System.out.println("Olá, mundo");  
}
```

Estrutura do programa

```
package olaMundo;

public class OlaMundo {

    public static void main(String[] args) {
        mensagem();
    }

    public static void mensagem() {
        System.out.println("Olá, mundo");
    }

}
```

Exemplo – Olá, mundo

E se quisermos personalizar para direcionar o olá para alguém?

* Dados são passados pelo módulo principal (ex: main) a sub-rotinas (procedimentos ou funções), por meio de **parâmetros**.

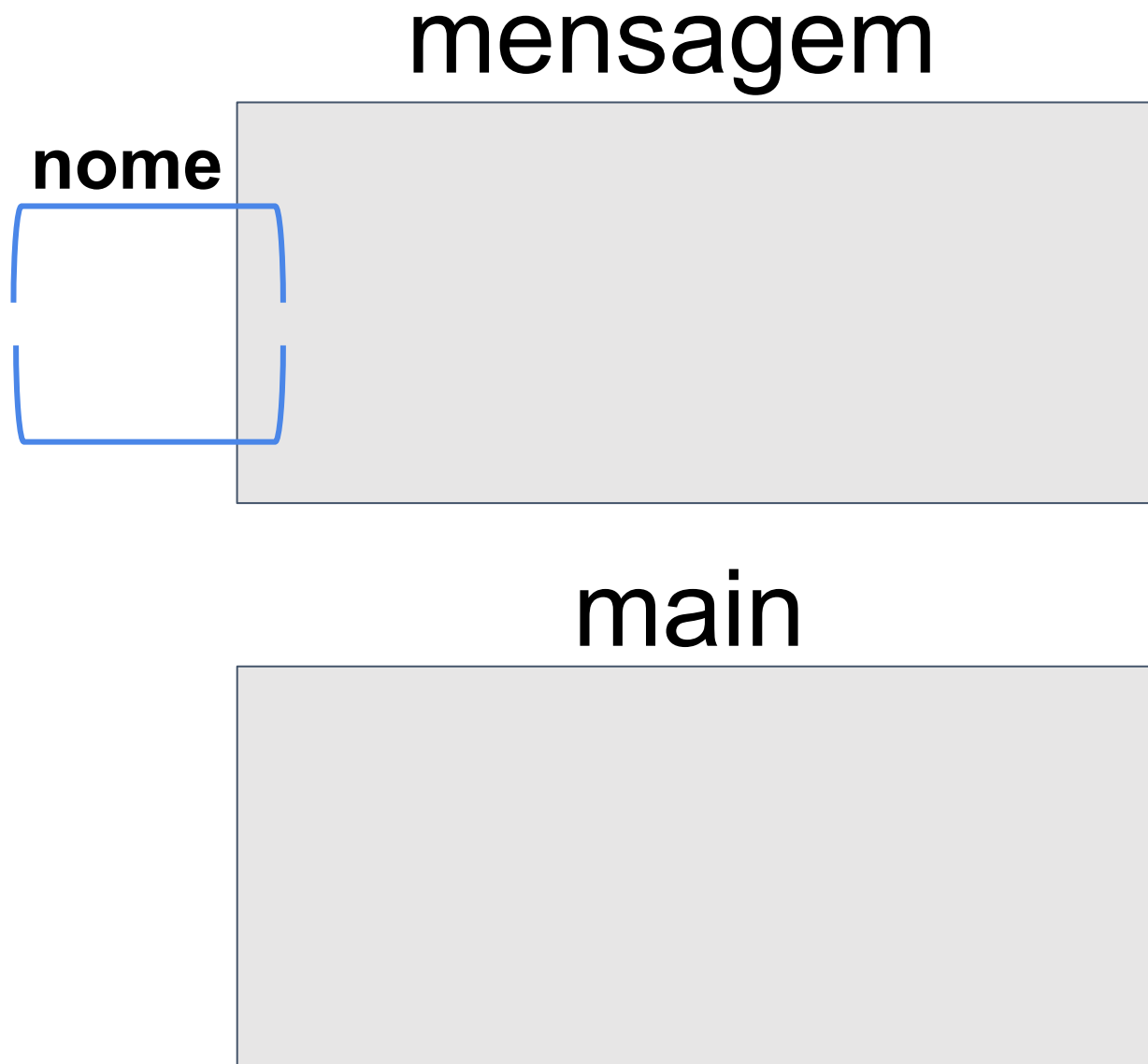
```
package olaMundo;
```

```
public class OlaMundo {
```

```
    public static void main(String[] args) {  
        mensagem("João");  
    }
```

```
    public static void mensagem(String nome) {  
        System.out.println("Olá, " + nome + "!");  
    }  
}
```

Parâmetros



Exemplo – soma

```
public static void main(String[] args) {  
    int x = 2;  
    soma(x, 5);  
}
```

```
public static void soma(int a, int b) {  
    int c;  
    c = a + b;  
    System.out.println(c);  
}
```

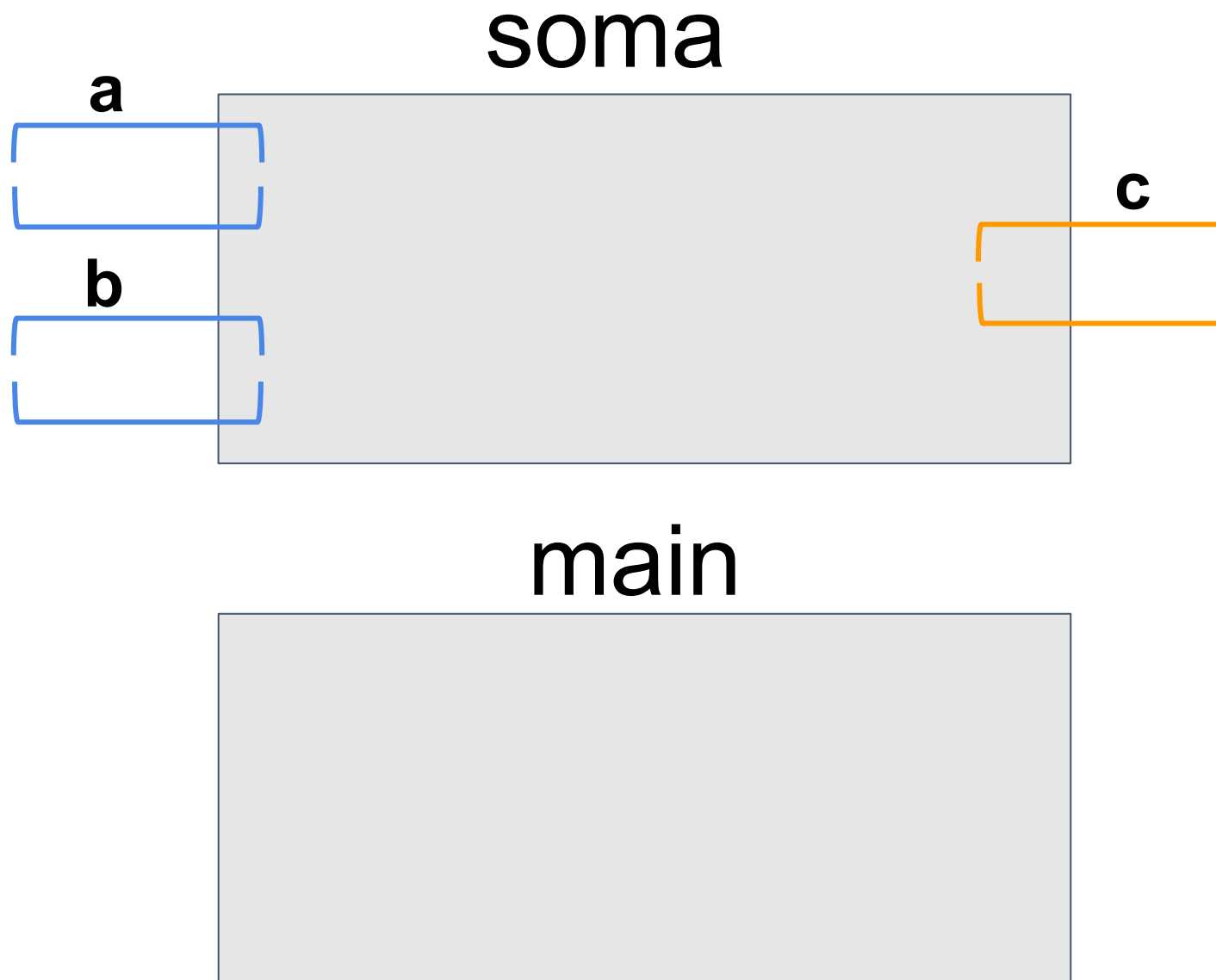
Exemplo – soma

```
public static void main(String[] args) {  
    int x = 2;  
    soma(x, 5);  
}
```

```
public static void soma(int a, int b) {  
    int c;  
    c = a + b;  
    System.out.println(c);  
}
```

E se quisermos devolver o valor da soma para quem chamou o módulo soma?

Parâmetros



Exemplo – soma

```
public static void main(String[] args) {  
    int x = 2, s;  
  
    s=soma(x, 5);  
}
```

```
public static int soma(int a, int b) {  
    int c;  
    c = a + b;  
  
    return c;  
}
```

Referências

Agradecimentos ao Prof André santanchè e Christianne Dalforno por ceder exemplos e alguns slides que foram adaptados.

Imagem lego:

https://upload.wikimedia.org/wikipedia/commons/6/61/Lego_blocks.jpg