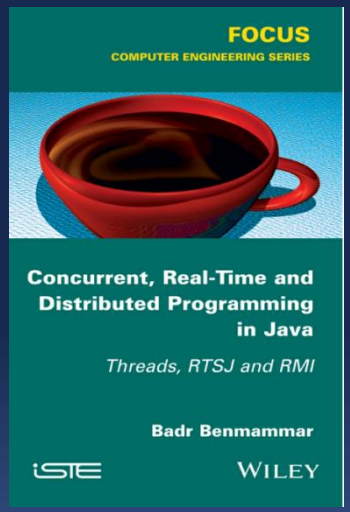
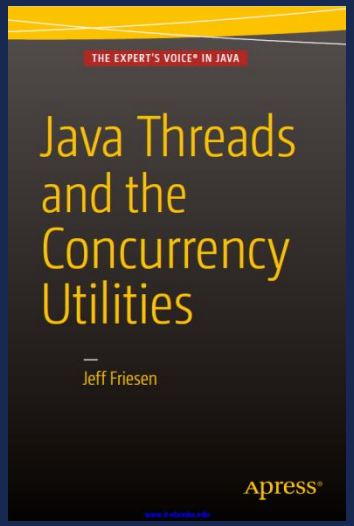
Programação Paralela e Concorrente

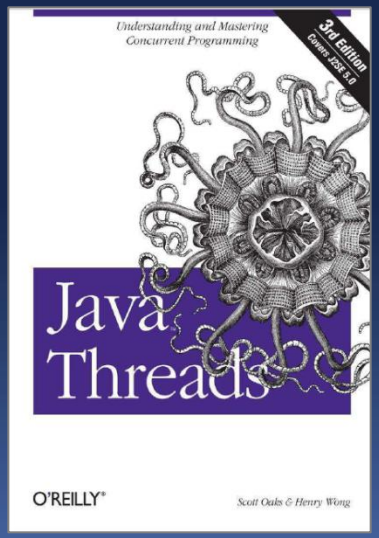
Unidade 5 – Trabalhando com diversos Threads

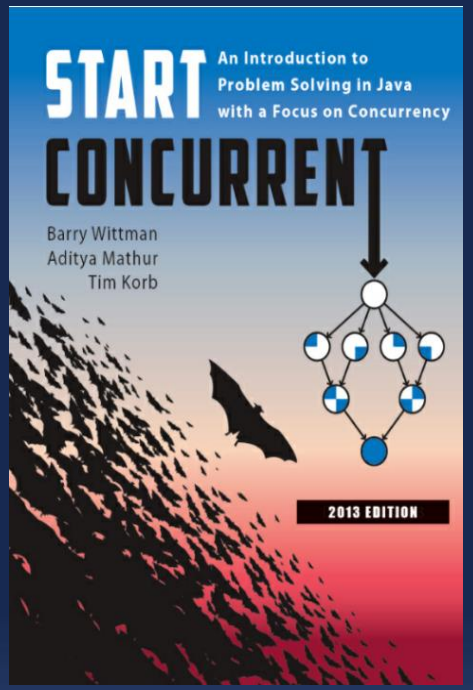
Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUSP
aparecidovfreitas@gmail.com

Bibliografia









Introdução

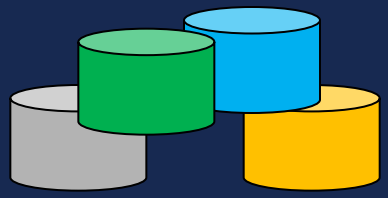
- Vimos nas unidades anteriores que a **Máquina Virtual Java,** desde a sua primeira implementação, tem o suporte nativo à **threads**;
- Assim, ao se criar uma aplicação java, a função main() por si só já é um thread.
- Em tempo de execução, pode-se criar a partir do main() diversos outros threads que serão processados sob a máquina virtual;
- Esses threads são indepedentes e cada um tem a sua própria pilha de métodos.

Manuseando vários threads

- Nesta unidade, faremos uma aplicação que irá consultar **4 arquivos**, formato **txt**, para encontrar um **texto** qualquer, que será informado pelo usuário, em tempo de execução;

- Os arquivos correspondem à estudantes que estão inscritos em cursos online e possuem os seguintes nomes:

  - ✓ `inscritos_C_2019.txt`
  - ✓ `inscritos_C_2020.txt`
  - ✓ `inscritos_Python_2019.txt`
  - ✓ `inscritos_Python_2020.txt`



Classe Principal

- No projeto **BuscaTexto**, criaremos inicialmente, a nossa classe **Principal** com a função **main()** que irá receber do usuário o texto a ser pesquisado.

```
package br.uscs;

public class Principal {

    public static void main(String[] args) {

    }

}
```

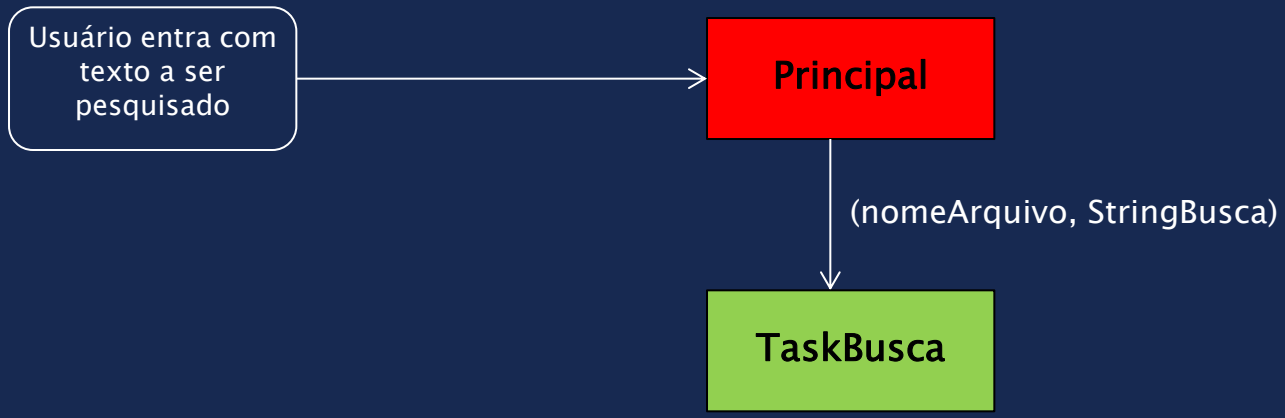
Classe Principal

- Na classe **Principal**, na função **main()** vamos escrever o código que faz a interface com o usuário para receber dele o texto a ser pesquisado.

```
package br.uscs;

import java.util.Scanner;

public class Principal {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);

        System.out.println("Entre com o texto a ser pesquisado: ");

        String texto = in.nextLine();

        System.out.println("Texto a ser pesquisado =>  " + texto);

        in.close();

    }
}
```

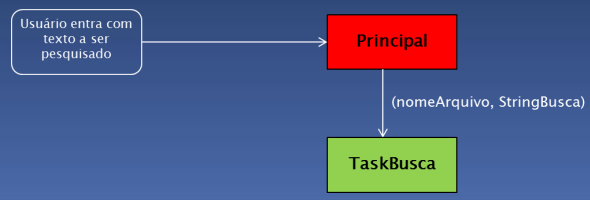
Criação das Tarefa de Busca

- Vamos agora criar **uma classe chamada TaskBusca que corresponde a um thread que recebe da classe principal o nome do arquivo a ser pesquisado e o string de pesquisa informado pelo usuário.**



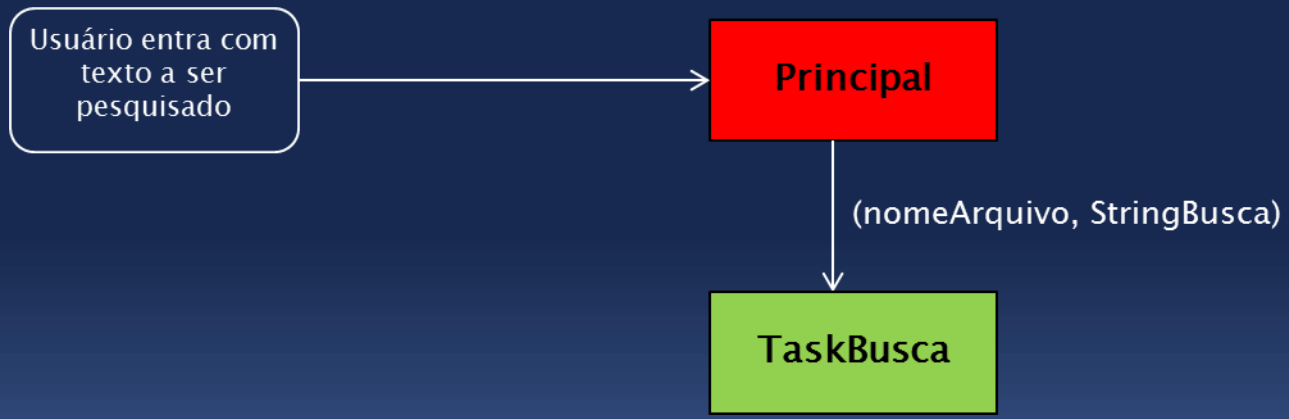
Classe TaskBusca

```
package br.uscs;

public class TaskBusca implements Runnable {

    private String nomeArquivo;
    private String textoBusca;

    public TaskBusca(String nomeArquivo, String textoBusca) {
        this.nomeArquivo = nomeArquivo;
        this.textoBusca = textoBusca;
    }
    @Override
    public void run() {

    }

}
```



Implementação do método run()

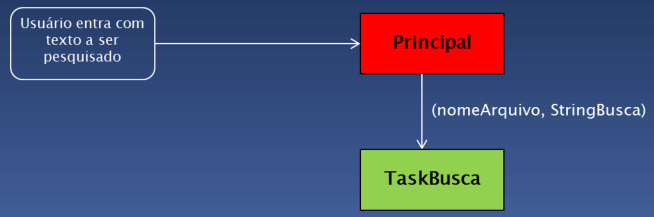
- Vamos agora implementar a tarefa que será processada pelo método run() na classe TaskBusca.
- O método run() deverá ler o arquivo texto recebido e pesquisar seu conteúdo para localizar o texto informado pelo usuário.



Implementação do método run()

- Inicialmente, no método **run()** vamos declarar o arquivo a ser lido.

```
@Override
public void run() {

    File file = new File(this.nomeArquivo);

    try {
        Scanner scanner = new Scanner(file);
        scanner.close();

    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
}
```



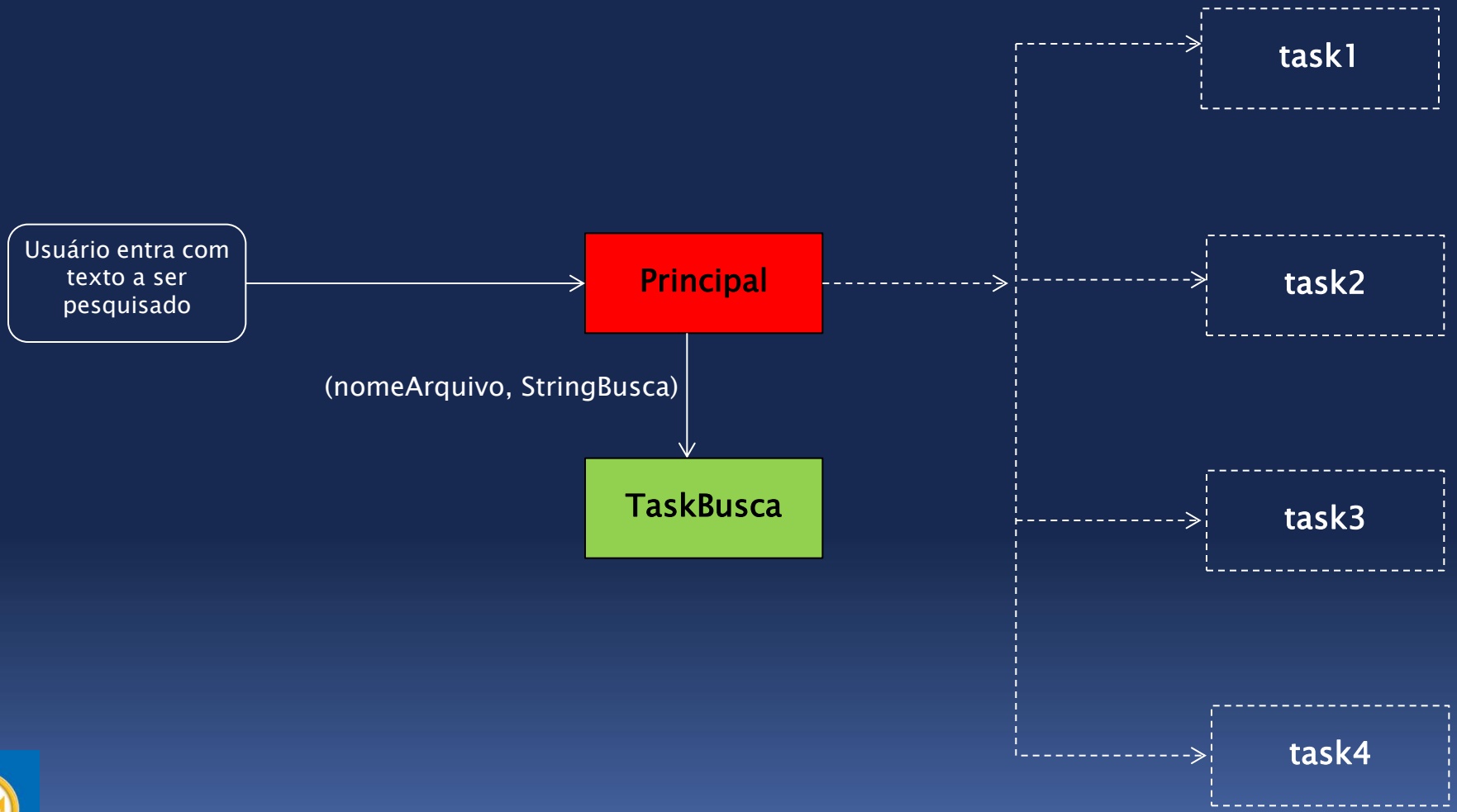
Implementação do método run()

- Vamos agora, no método **run()** fazer a busca do texto no arquivo.

```
@Override
public void run() {

    File file = new File(this.nomeArquivo);
        try {
        Scanner scanner = new Scanner(file);

        int numeroLinha= 1;
        while (scanner.hasNextLine()) {
            String linha = scanner.nextLine();
            if (linha.contains(textoBusca)) {
                System.out.println("Arquivo: " + nomeArquivo + " ** " +
                            "Linha: "+ numeroLinha + " ** " + "Registro: " + linha);
            }
            numeroLinha++;
        }
        scanner.close();

    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
}
```

Criação dos Threads em main()

- Vamos agora, no método **main()** criar quatro instâncias da **TaskBusca** uma vez que estas instâncias representam as tarefas a serem executadas pelos threads.



Criação dos Threads em main()

```java
import java.util.Scanner;

public class Principal {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);
        System.out.println("Entre com o texto a ser pesquisado: ");
        String texto = in.nextLine();
        System.out.println("Texto a ser pesquisado =>  " + texto);
        in.close();

        Runnable task1 = new TaskBusca("Inscritos_C_2019", texto);
        Runnable task2 = new TaskBusca("Inscritos_C_2020", texto);
        Runnable task3 = new TaskBusca("Inscritos_Python_2019", texto);
        Runnable task4 = new TaskBusca("Inscritos_Python_2020", texto);

        Thread thread1 = new Thread(task1, "Thread - Task1");
        Thread thread2 = new Thread(task1, "Thread - Task2");
        Thread thread3 = new Thread(task1, "Thread - Task3");
        Thread thread4 = new Thread(task1, "Thread - Task4");
    }
}
```

Disparando a execução dos threads

```
package br.uscs;
import java.util.Scanner;
public class Principal {

    public static void main(String[] args) {

        Scanner in = new Scanner(System.in);
        System.out.println("Entre com o texto a ser pesquisado: ");
        String texto = in.nextLine();
        System.out.println("Texto a ser pesquisado =>  " + texto);
        in.close();

        Runnable task1 = new TaskBusca("Inscritos_C_2019.txt", texto);
        Runnable task2 = new TaskBusca("Inscritos_C_2020.txt", texto);
        Runnable task3 = new TaskBusca("Inscritos_Python_2019.txt", texto);
        Runnable task4 = new TaskBusca("Inscritos_Python_2020.txt", texto);

        Thread thread1 = new Thread(task1, "Thread - Task1");
        Thread thread2 = new Thread(task2, "Thread - Task2");
        Thread thread3 = new Thread(task3, "Thread - Task3");
        Thread thread4 = new Thread(task4, "Thread - Task4");

        thread1.start();
        thread2.start();
        thread3.start();
        thread4.start();

    }
}
```

Executando a aplicação

- Verifique na log de execução a impressão entrelaçada correspondente aos quatro threads da aplicação.

```
Entre com o texto a ser pesquisado:
Felipe
Texto a ser pesquisado =>  Felipe
Arquivo: Inscritos_C_2020.txt ** Linha: 12 ** Registro: Felipe  Leite de Souza  felipe_leitesouza@hotmail.com          29
Arquivo: Inscritos_Python_2019.txt ** Linha: 25 ** Registro: Ricardo Felipe     da Silva        ricardo.silva@ndc.com.br
Arquivo: Inscritos_Python_2019.txt ** Linha: 44 ** Registro: Felipe     Leite de Souza  felipe_leitesouza@hotmail.com
Arquivo: Inscritos_Python_2020.txt ** Linha: 11 ** Registro: Felipe     Leite de Souza  felipe_leitesouza@hotmail.com
Arquivo: Inscritos_Python_2020.txt ** Linha: 26 ** Registro: Felipe     Carnevale Fornaziero    felipevictorcarnevale@hotma
Arquivo: Inscritos_Python_2020.txt ** Linha: 50 ** Registro: Felipe     de Oliveira Giuriollo   felipegiuriollo@gmail.com
Arquivo: Inscritos_C_2019.txt ** Linha: 31 ** Registro: Felipe  Leite de Souza  felipe_leitesouza@hotmail.com          29
Arquivo: Inscritos_C_2020.txt ** Linha: 26 ** Registro: Felipe  Carnevale Fornaziero    felipevictorcarnevale@hotmail.com
Arquivo: Inscritos_C_2020.txt ** Linha: 51 ** Registro: Felipe  de Oliveira Giuriollo   felipegiuriollo@gmail.com
```