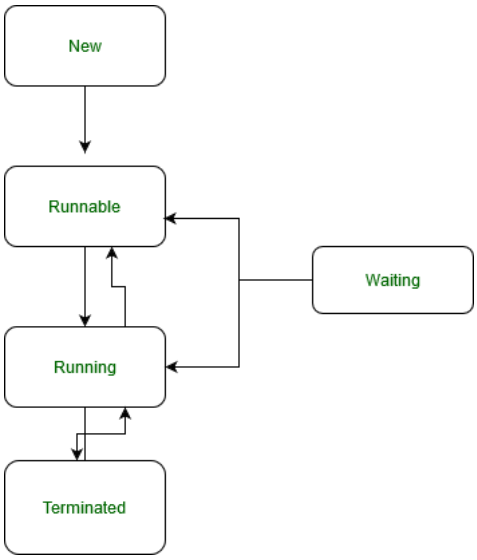
PROGRAMA JAVA PARA USAR EXCEÇÕES COM THREAD

Exceções são os eventos que ocorrem devido a erro do programador ou erro da máquina que causa uma perturbação no fluxo normal de execução do programa. Quando um método encontra uma condição anormal que não pode manipular, uma exceção é lançada como uma instrução de exceção. As exceções são capturadas por manipuladores (aqui, captura de bloco). As exceções são capturadas por manipuladores posicionados junto com a pilha de invocação de método do encadeamento. Se o método de chamada não estiver preparado para capturar a exceção, ele lançará a exceção até seu método de chamada e assim por diante. Portanto, no programa java, os manipuladores de exceção devem ser posicionados estrategicamente, para que o programa capture todas as exceções das quais o programa deseja se recuperar.

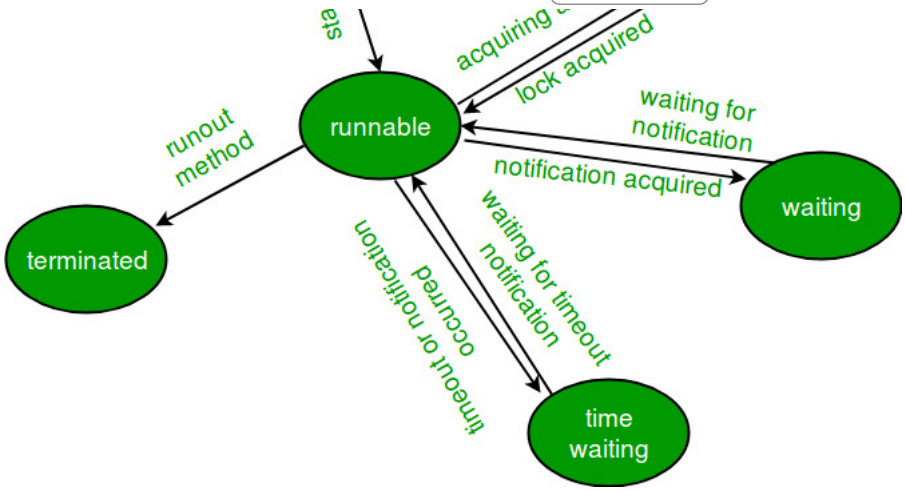
Ciclo de vida de um fio : Os implementos de classe a classe Thread ou de interface Runnable, em seguida, a classe estendido método start() executar o fio, sleep() métodos de fazer com que o thread atualmente em execução no sono para o número especificado de milissegundos **,** e muitos mais.



Antes de discutir as abordagens, indique. as transações do thread devem ser conhecidas para lidar com as exceções para uma melhor compreensão. Um encadeamento em Java existe em qualquer ponto no tempo em qualquer um dos seguintes estados. Um segmento encontra-se apenas em um dos estados mostrados a qualquer momento:

1. Novo
2. Executável
3. Bloqueado
4. Esperando
5. Espera cronometrada
6. Rescindido

Design ruim
Inadequado
Repetitivo
Irrelevante



1. Um nome de classe RunnableThread implementa a interface Runnable que fornece o método run() executado pelo thread. O objeto desta classe agora pode ser executado

2. O construtor Thread é usado para criar um objeto da classe RunnableThread passando o objeto executável como um parâmetro.

3. O método *start()* é invocado no objeto Thread, pois retorna imediatamente após a geração de um thread.

4. O encadeamento termina quando o método run() termina, o que deve ser uma finalização normal ou exceção capturada.

5. Agora, para criar um novo tópico

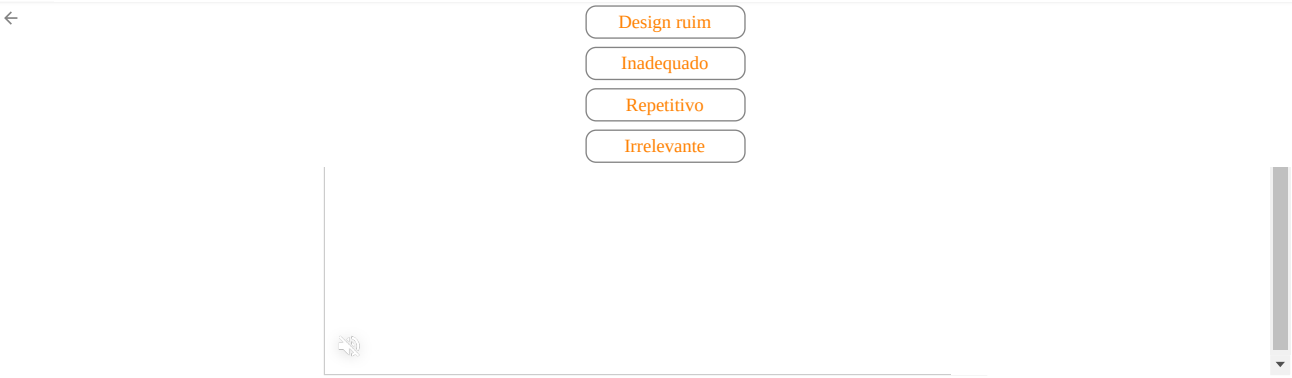
```
runner = new Thread(this,threadName) ;
```

6. Para iniciar o novo tópico.

```
runner. start() ;
```

7. public void run() é um método substituível usado para exibir as informações de um determinado segmento.

8. Thread.currentThread(). Sleep (2000) é usado para desativar o encadeamento até que o próximo encadeamento *inicie* a execução ou usado para atrasar o encadeamento atual.



O manipulador de exceção não capturada será usado para demonstrar o uso de exceção com thread. É uma interface específica fornecida por Java para lidar com exceção no método de execução de encadeamento.

Existem dois métodos para criar um thread:

1. **Estenda a classe de thread (java.lang.thread)**
2. **Implementar interface executável (java.lang.thread)**

1. Exceção e manipulação de exceção com threads

Aqui, um novo thread é criado na classe que está estendendo a classe de thread na qual o método *run()* é sobrescrito. Isso invoca o ponto de entrada do novo encadeamento criado na classe que estava estendendo a classe de encadeamento. Além disso, o método *start()* é usado para iniciar e executar a thread no programa.

```
// Java program to Use exceptions with thread

// Importing Classes/Files
import java.io.*;

// Child Class(Thread) is inherited from parent Class(GFG)
class GFG extends Thread {

    // Function to check if thread is running
    public void run()
    {
        System.out.println("Thread is running");

        // Using for loop to iterate
        for (int i = 0; i < 10; ++i) {
            System.out.println("Thread loop running " + i);
        }
    }

    // Main Driver Method
    public static void main(String[] args)
    {

        // Try-catch block to detect exception
        try {

            // Creating new thread
            GFG ob = new GFG();

            throw new RuntimeException();
        }

        // Catch block to handle exception
        catch (Exception e) {

            // Exception handler
            System.out.println(
                "Another thread is not supported");
        }
    }
}
```

Design ruim
Inadequado
Repetitivo
Irrelevante

2. Tratamento de exceções com o método sleep(): o método sleep() da classe thread é usado onde há uma demanda para suspender o thread por um determinado período de tempo para o fluxo de trabalho adequado do código.

Sintaxe:

sono vazio público estático (milissegundos longos); // geralmente usado

sono vazio público estático (milissegundos longos, nanosegundos int); // usado para ilustrar apenas a precisão

Parâmetro:

NOME	AÇÃO EXECUTADA
milissegundos	Duração de tempo para fazer um tópico dormir
nanossegundos	Duração adicional de tempo para suspender o thread, mas restrito a 999999

Tipo de retorno: como visto na própria sintaxe, não retorna nenhum valor.

Exceção: muitas vezes gera exceções, já que a linguagem java envolve o conceito de multithreading

1. *IllegalArgumentException* é lançada quando o valor paramétrico é negativo, pois é limitado, conforme discutido entre [0 - +999999]
2. *InterrupteException* é lançada quando um thread é interrompido com um thread em andamento, conforme discutido em java, que oferece suporte aos conceitos de multithreading.

Implementação:

```
// Java program to Use exceptions with thread

/* Note: Dont confuse main method with Main class*/

// Importing Classes/Files
import java.io.*;

// Child Class(Thread) is inherited
// from parent Class(GFG)
class GFG extends Thread {
    public void run()
    {
        System.out.println("Throwing in "
                           + "MyThread");
        throw new RuntimeException();
    }
}

// Main driver method
public class Main {
    public static void main(String[] args)
    {
        GFG t = new GFG();
        t.start();

        // try block to deal with exception
        try {
            Thread.sleep(2000);
        }

        // catch block to handle the exception
```

Design ruim
Inadequado
Repetitivo
Irrelevante

```
        // Print command just to show program
        // run successfully
        System.out.println("Completed");
    }
}
```

Saída:

```
Throwing in MyThread
Exception in thread "Thread-0" java.lang.RuntimeException
        at testapp.MyThread.run(Main.java:19)
Completed
```

Anterior Próximo

Java-Exception Handling Technical Scripter 2020 Java Java Programs Technical Scripter

LATEST POSTS

Experiência de Entrevista Epicor
Experiência de entrevista Ebix
Experiência de entrevista com Pickyourtrail (SET 1)
C-DOT (experiência de entrevista em tempo integral)
Diferença Máxima de Peso

MOST POPULAR POSTS

Lidando com linhas e colunas no Pandas DataFrame
Projetos Python - do iniciante ao avançado
7 ideias interessantes de projetos em Python para desenvolvedores intermediários
As 7 principais ideias de projetos Java para aprimorar as habilidades de programação
As 10 principais bibliotecas Python para ciência de dados em 2020