Pilha: um programa em Java sempre começa no método main(). Depois, ele cria a pilha e coloca os blocos de código um em cima do outro, na pilha. Uma pilha Java faz parte da JVM e armazena os métodos que estão sendo executados. Além do bloco de código, a pilha guarda as variáveis e as referências deste bloco. Assim a JVM organiza a execução e sabe exatamente qual método está sendo executado, que é sempre o método no topo da pilha. A JVM também sabe quais outros precisam ser executados ainda, que são justamente os métodos abaixo. A pilha também é chamada de stack ou pilha de execução ou call stack.

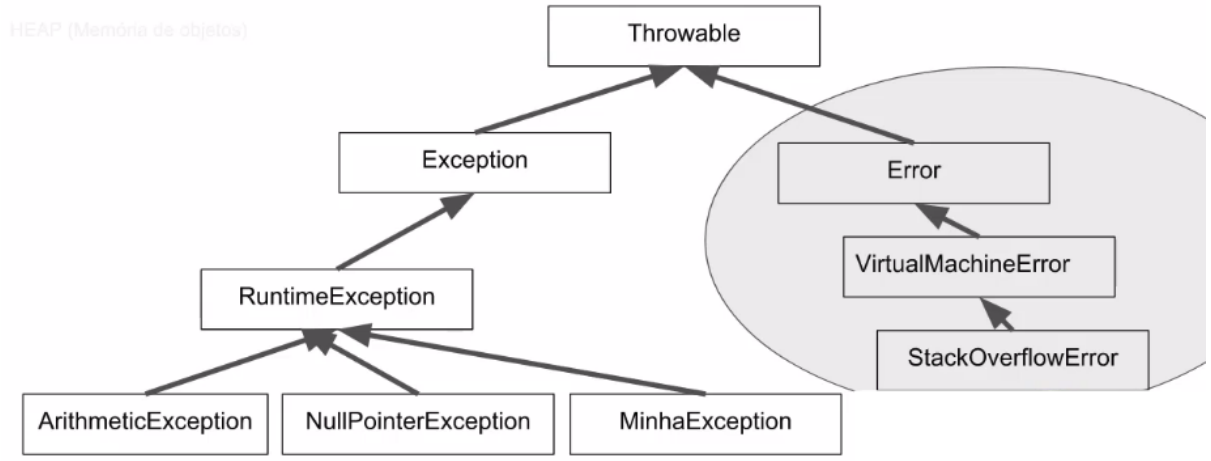
Exceções: Exceção é um objeto. São baseados em classes (observe o nome)

```
try {
        tentar isso…
} catch (ArithmeticException ex) //se der ruim, fazer isso
ex: variável criada por nós. Serve para referência
}
```

Toda exceção em Java possui um nome que a identifica. Essa abordagem torna seu entendimento mais fácil do que o uso de códigos de erros como 15, 7012 ou 16.
Exceções não tratadas caem na pilha de execução procurando por alguém que saiba lidar com elas. Não tendo nenhuma solução no método atual, esse é “jogado fora”, todos os métodos da pilha são testados à procura de soluções. Por isso, uma exceção muda o fluxo de um programa.

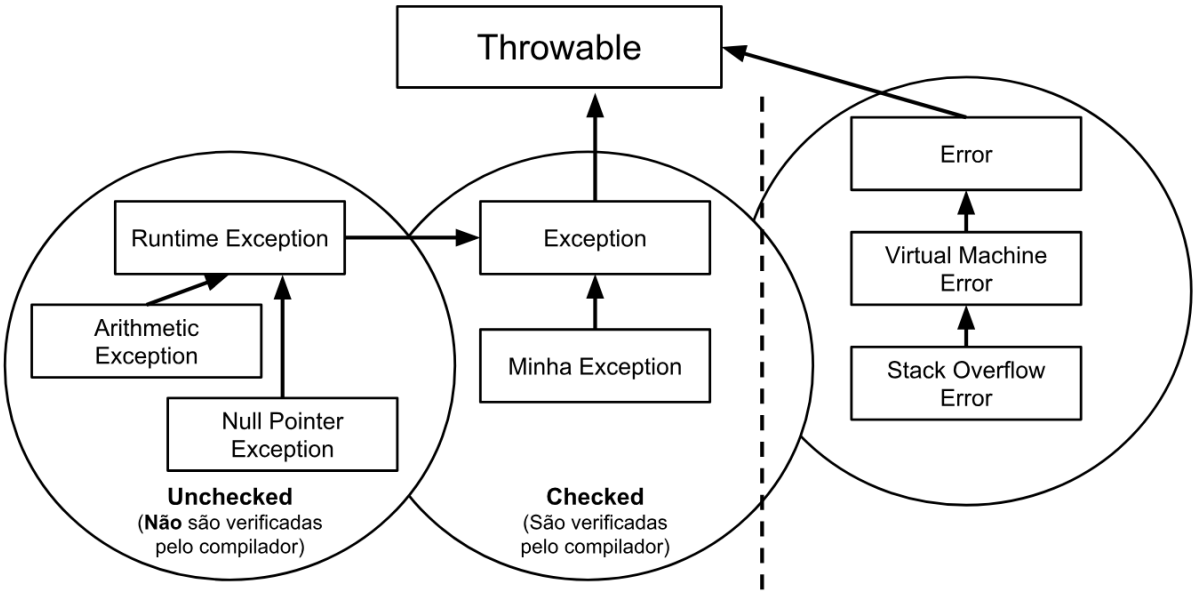
Ao estudarmos a hierarquia das exceções vimos que é a classe do topo — Throwable — quem realmente faz as coisas. Exception e

RuntimeException não possuem utilidade e cada uma só possui alguns construtores. Então qual a diferença entre estender de RuntimeException ou Exception?



Se estendemos de RuntimeException, temos uma exceção do tipo Unchecked, ou seja, o compilador não toma atitude. Tendo throw ou não, ele não se importa.

Se estendermos diretamente da classe Exception, o compilador ficará de olho e nos obrigará a colocar throws na assinatura do método, para sinalizar quem chama o método, que ele é perigoso ou tratar a exceção no próprio método com o try-catch. Essa exceção é do tipo checked.



Na hora de rodar (executar) **não** tem diferença!!!

- Atenção: Checked e Unchecked é algo específico do mundo Java e estão relacionados somente ao processo de compilação. Na hora de executar o código, não tem diferença!

Existe uma outra hierarquia de classes que estende Throwable, como a classe Error. Entretanto, nós não a conhecemos muito bem, porque é uma hierarquia pensada para desenvolvedores de máquina virtual. Nós, desenvolvedores Java, não trabalhamos com essas classes diretamente. Quem cria e joga esses objetos na pilha é a máquina virtual, quando algo muito grave acontece. O erro mais comum é o StackOverFlowError.