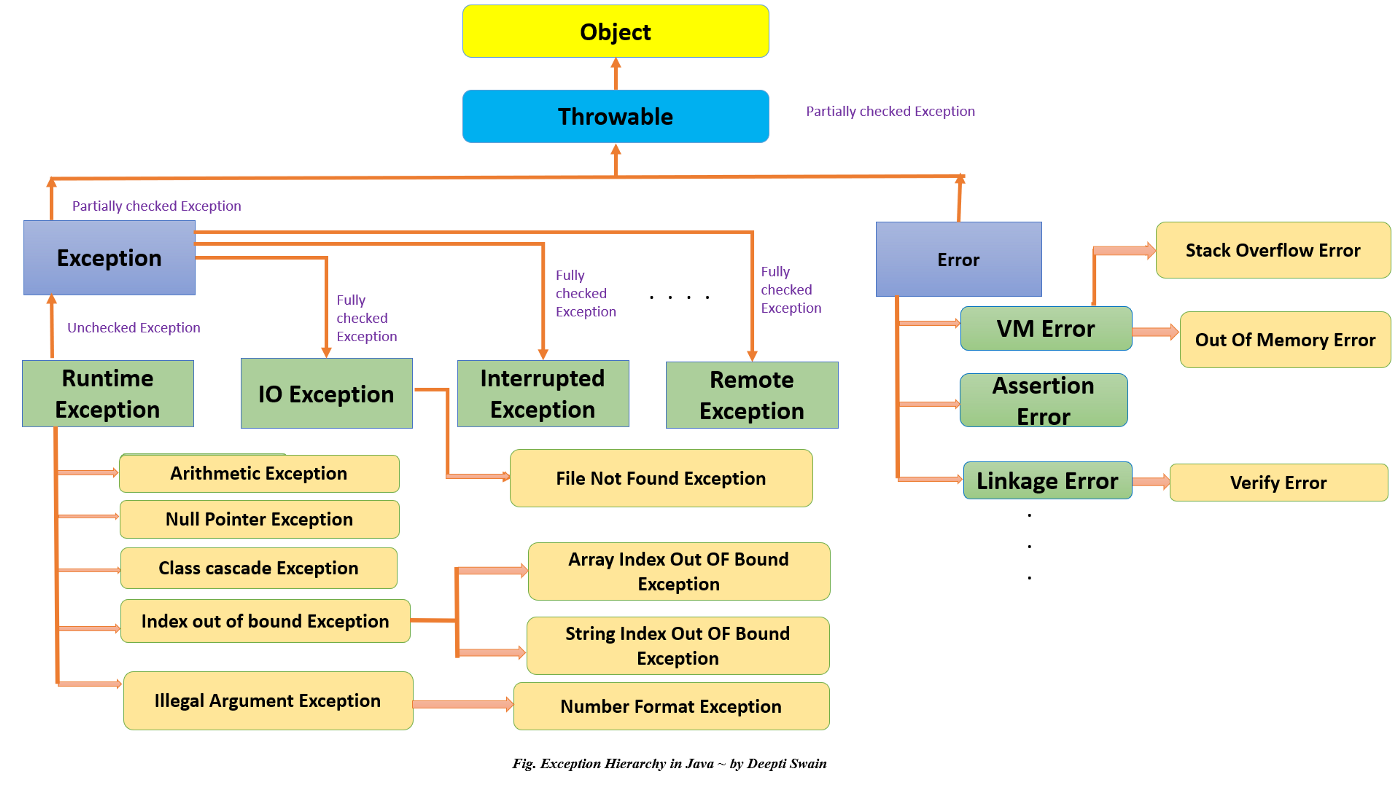
**Excessão** é um evento que interrompe o fluxo normal de processamento de uma classe, isso torna um programa mais robusto e confiavel e evita o programa de parar de funcionar por erros, sua principal função é garantir que o programa continue e rodando e que qualquer bug no sistema seja corrigido (através do uso de exceções).

**Error** é usado pelo JVM que indica **um problema de recurso do programa (**como falta de RAM), tornando a execução possivel de continuar, error não é excessão **e não pode ser corrigido nem identificado**.

**Unchecked (runtime)** exceptions que **podem** ser evitados se forem tratados e analisados pelo desenvolvedor, é causada principalmente por **loops infinitos que geral um uso muito grande do sistema**. São exceções herdadas da classe RuntimeExceptions ou da classe Error, se uma excessão ocorrer e não for capturada, irá quebrar o programa, em geral é evitada através de um código limpo.

**Checked exceptions** exceptions que **devem** ser evitados e tratados pelo desenvolvedor para o programa funcionar. Herdam da classe Exception, mas não da RuntimeException, o compilador lança captura a excessão e lança throws de acordo, caso ele não capture ou não seja declarado o throw, ocorrerá um erro de compilação.

**Exception personalizada** programadores podem achar util ter sua própria declaração de excessão, elas são especificas ao problema e podem ocorrer quando outro programador empregar suas classes reutilizaveis, essa nova classe deve extender uma existente que assegura que a classe possa ser utilizada como mecanismo de tratamento de excessão.

O tratamento de exceções é feito através das seguintes palavras reservadas:

**Try, catch** & **finally** que definem blocos que tratam a excessão, **tentando** verificar o erro, **pegando** o erro e o tratando para **finalmente** fazer o que for necessário na situação (transformar o valor ou tentar de novo se necessário). Note que o finally **não é obrigatório**, muitas vezes o catch sendo o suficiente para tratar o problema.

**Throws** declara que o método pode lançar uma ou mais exceções. Throws é usada na assinatura do método, usado apenas para exceções checked e serve para informar ao “chamador” que essa estrutura pode lançar exceções, obrigando o uso dos blocos try-catch.

**Throw** lança explicitamente uma exception (aquela caixa de erro grande), usado principalmente para instruções personalizadas, ele procura o try catch mais próximo do programa e joga o problema para ele.

Sobre os tipos de erros, segue alguns exemplos:

A classe **FileReader** pede que a excessão **FileNotFoundException** seja chamada para caso não haja o arquivo requisitado.

**NumberFormatException** é requisitado quando um número é requisitado onde uma string é colocada.