Aula 06

Tratamento de exceção
e
Fluxo de Dados

AMBIENT
INFORMÁTICA

Pauta

- Tratamento de exceção
- Lançando exceções
- Hierarquia das exceções:
- Exceções verificadas
- Exceções não verificadas
- Declarando novos tipos de exceções
- Assertivas

Tratamento de exceção

- Faça o código 01 do material
- Tente informar ZERO no denominador
- O que aconteceu?

Tratamento de exceção

- Todo código que pode lançar exceção deve ficar dentro do bloco try;
- Dentro do bloco catch fica o código que deve ser executado quando alguma exeção for lançada;
- Ordem de colocação do catch
- try{
- ... codigo ...}
- catch{

Tratamento de exceção

```
try{
... codigo ...}
catch(TipoExcecao e){
... codigo ...
}
```

Tratamento de exceção

- Já vimos que quando uma parte do código lança uma exceção a execução é interrompida;
- O que fazer quando for necessário executar algum código, mesmo quando ocorrer exceções, como por exemplo:
- Fechamento de arquivos
- Fechamento de banco de dados
- Descarga de recursos
- ???

AMBIENT INFORMÁTICA

Tratamento de exceção

```
try{
  ... codigo ...}
catch(TipoExcecao e){
  ... codigo ...
}finally{
    ... codigo ...
}
```

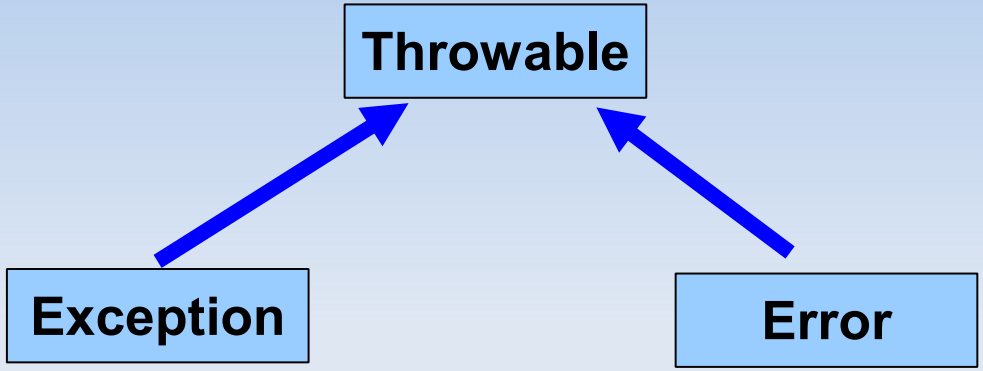
Tratamento de exceção

- O java garante que um bloco finally executará se uma exceção for lançada;
- Existe apenas um bloco finally para cada try;
- Não é obrigatório;
- Sempre será executado, sendo ou não lançada uma exceção;

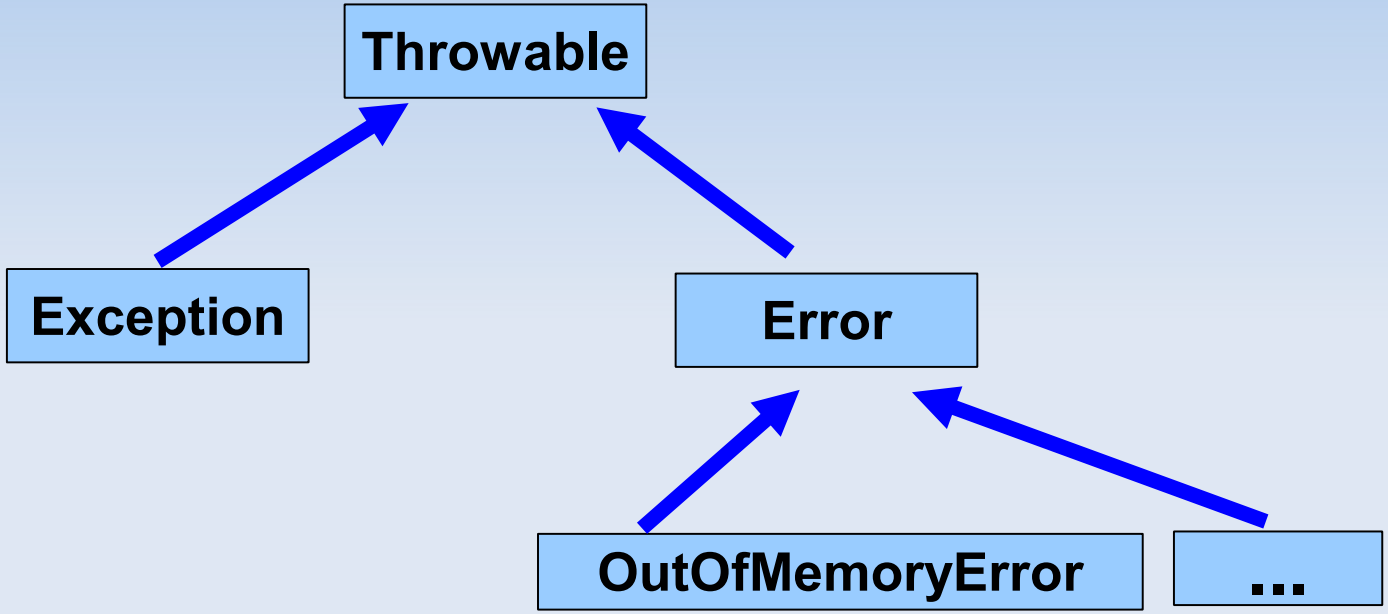
Lançando exceções

- Quando for necessário lançar uma nova exceção utilizamos a palavra chave: throw;
- Todo método que lançar uma exceção deve utilizar a palavra chave throws para tornar explícito quais exceções este método lança.
- Exceção verificada

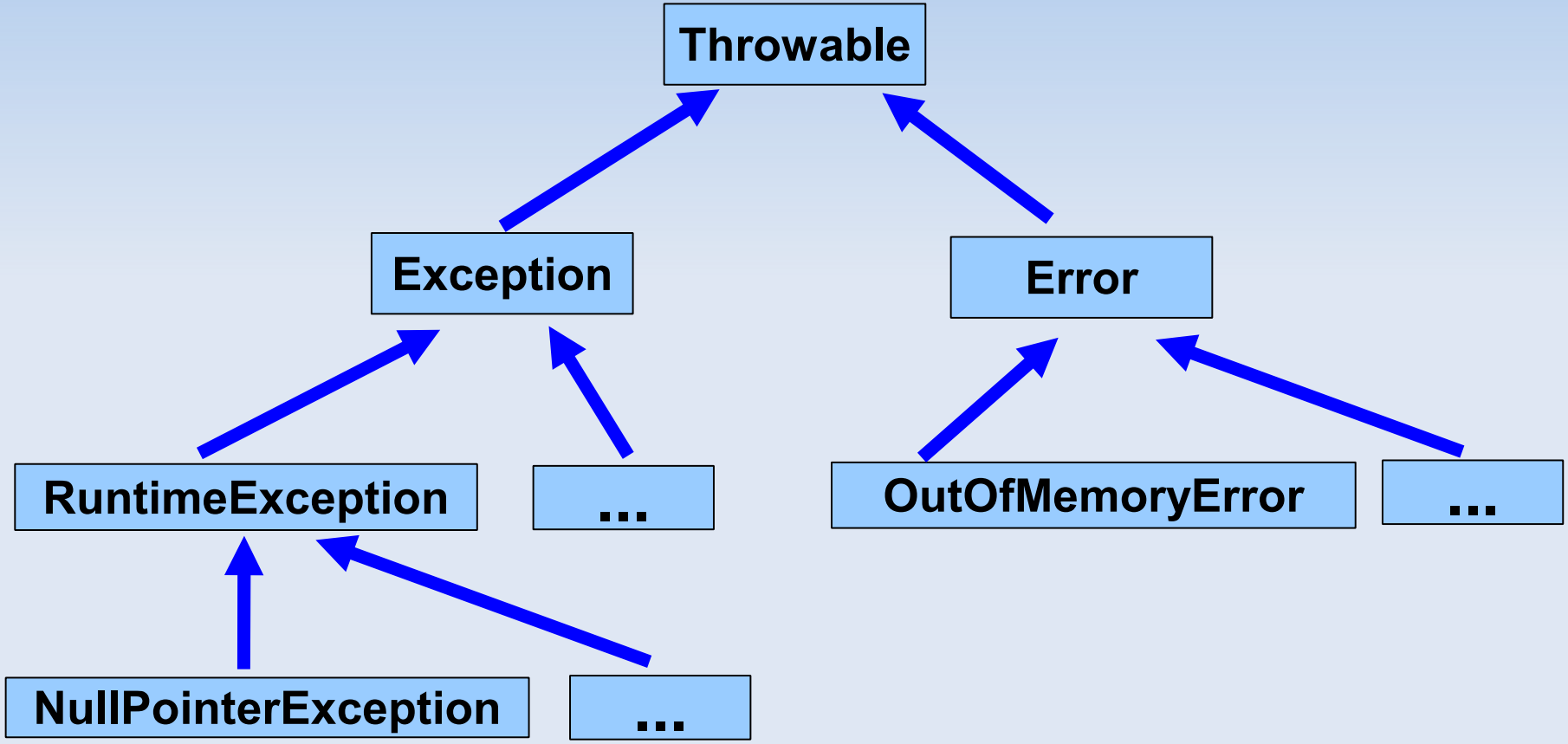
Hierarquia das exceções:



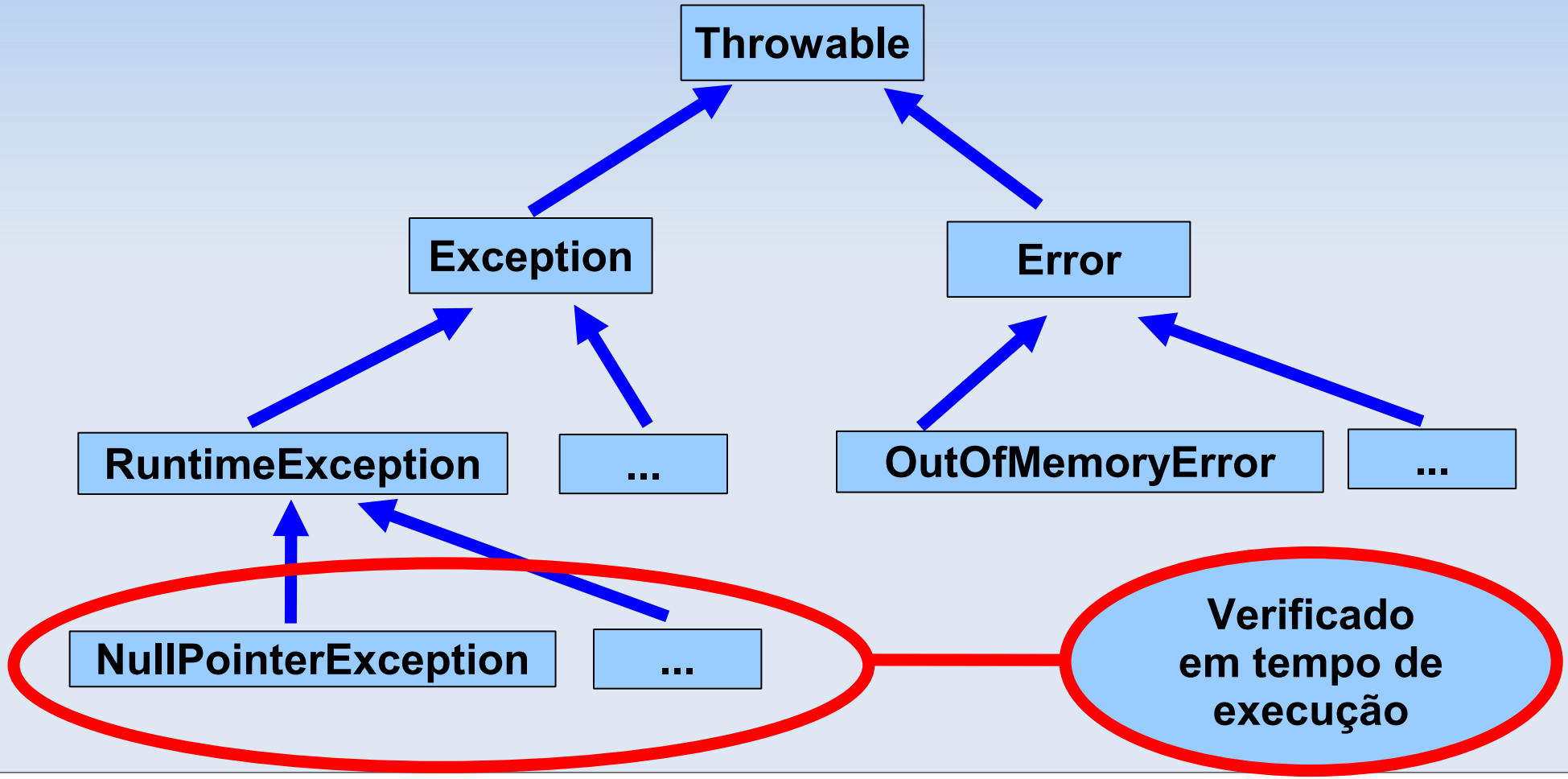
Hierarquia das exceções:



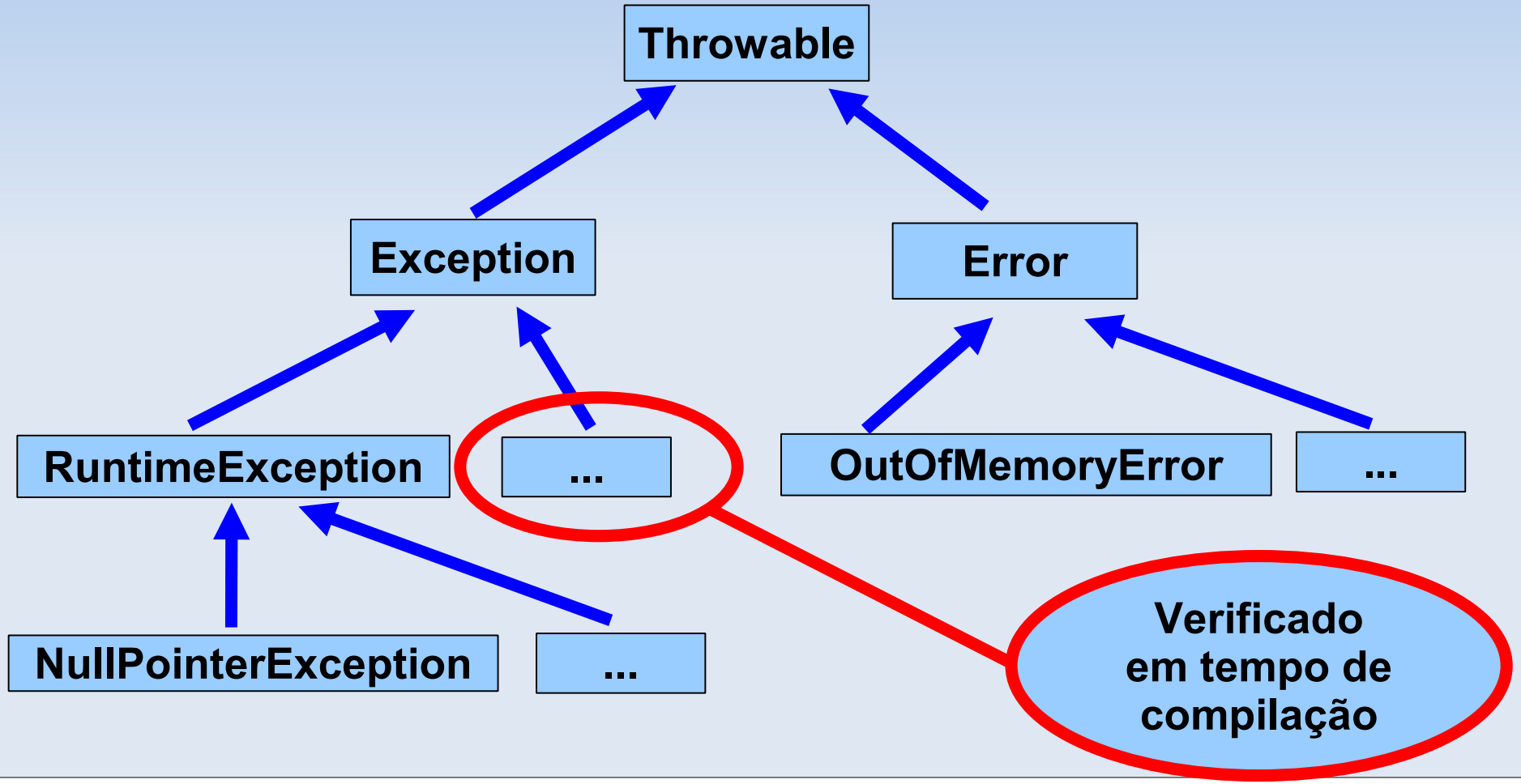
Hierarquia das exceções:



Hierarquia das exceções:



Hierarquia das exceções:



Hierarquia das exceções:

- Métodos da classe Throwable:
  - printStackTrace();
  - getStackTrace();
  - getMessage();
  - ... outros ...

Exceções verificadas

- São exceções que devem ser tratadas obrigatoriamente;
- São verificadas em tempo de compilação;
- Tratamos de duas formas:
- NÃO herdam de RunTimeException
- try ... catch
- throws

Exceções não verificadas

- Não são verificadas em tempo de compilação
- São verificadas em tempo de execução;
- São subclasses da classe RuntimeException;

Exemplo

- Faça o exemplo do código 02

Declarando novos tipos de exceções

- Todas as exceções devem ser do tipo Trowable;
- Se extendermos a classe Exception teremos exeções verificadas em tempo de compilação;
- Se extendermos a classe RuntimeException teremos exeções verificadas em tempo de execução;

Exemplo

- Faça o exemplo do código 03

Assertivas

- Utilizadas para auxiliar o programador a detectar bugs na implementação;
- **assert expressao;**
- Lança uma exceção do tipo AssertionError se a expressão for false;
- **assert expressao1: expressao2;**
- Lança uma exceção do tipo AssertionError com expressao2 como mensagem se a expressão1 for false;

Assertivas

- Java 1.4
- Parâmetros de execução: -ea
- Exemplo código 04

Arquivos e fluxos

Pauta

- Arquivos e fluxos
- Classe File
- Escrevendo em arquivos
- Lendo em arquivos
- JfileChooser
- Acesso aleatório
- Serialização de objetos

Arquivos e fluxos

- Tipos de fluxo:
  - Arquivos
  - Rede
  - STREAM
- Cada arquivo é visto como um fluxo;
  - Fluxos baseados em bytes:
  - Fluxos baseados em caracteres;

Classe File

- Recupera informações sobre arquivos ou diretórios;
- Alguns métodos úteis
- canRead();
- exist();
- getName();
- getPath();
- etc...

Escrevendo em arquivos

- A classe Formatter
- Método format(); aceita formatação
- Método close(); fecha o fluxo
- Existem outras maneiras de escrever em um arquivo;

Lendo em arquivos

- A classe Scanner
- O construtor Scaner(new File(...))
- Método hasNext()
- Método next();
- Método close();
- Existem outras formas de ler um arquivo

JFileChooser

- Métodos:
- showSaveDialog();
- getSelectedFile();
- Campos:
- CANCEL_OPTION

Acesso aleatório

- Permite leitura e escrita em arquivos
- RandomAccessFile

Serialização de objetos

- A interface Serializable;
- A classe ObjectOutputStream
- Método writeObject();
- A classe ObjectInputStream
- Método readObject();

Serialização de objetos

- Todos os objetos com campos serializaveis podem ser serializados;
- Palavra-chave transient;

DÚVIDAS ??

AMBIENT
INFORMÁTICA

Fim

AMBIENT
INFORMÁTICA