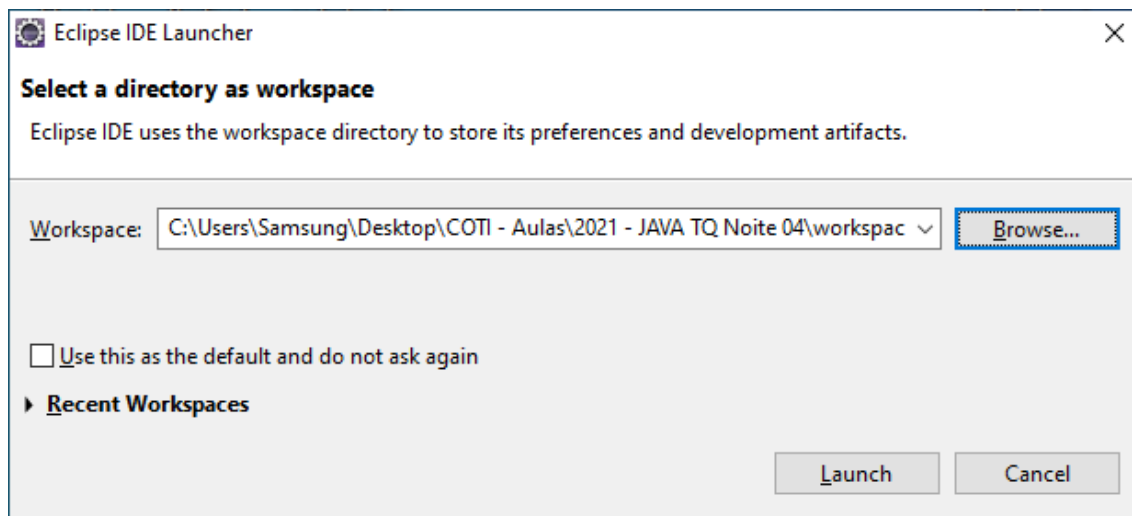
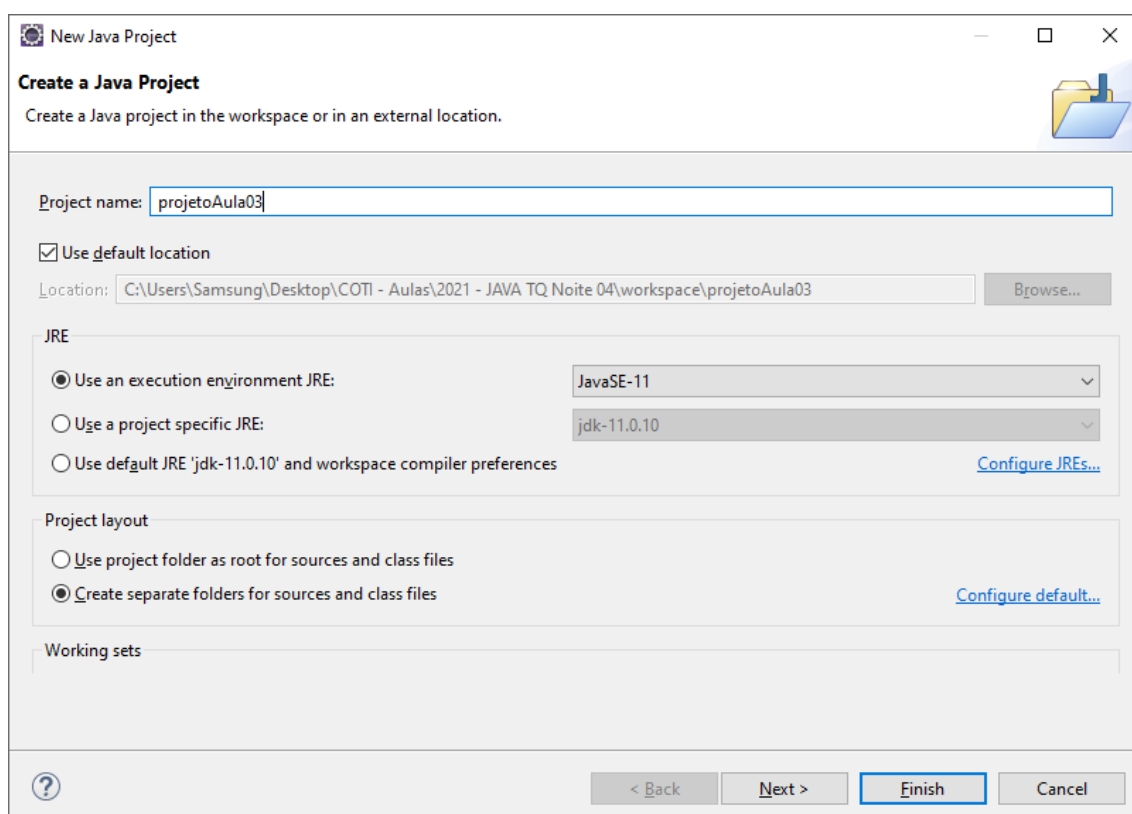
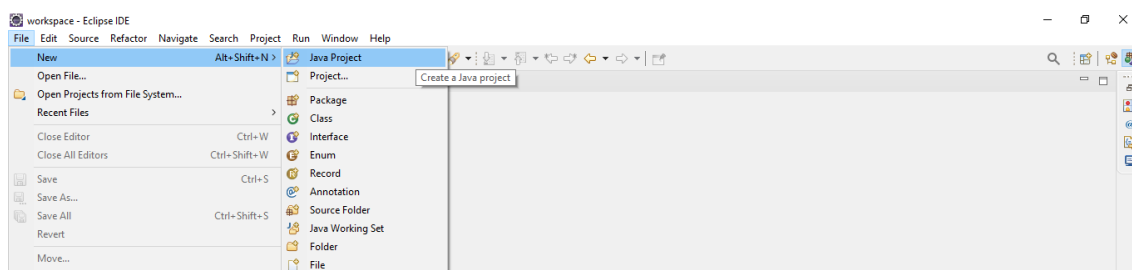


Abrindo o eclipse:



Criando um projeto:



Modelo de entidades

Diagrama de classes

- Empresa
 - IdEmpresa
 - NomeFantasia
 - RazaoSocial
 - Cnpj
- Funcionário
 - IdFuncionario
 - Nome
 - Salário
 - Cpf

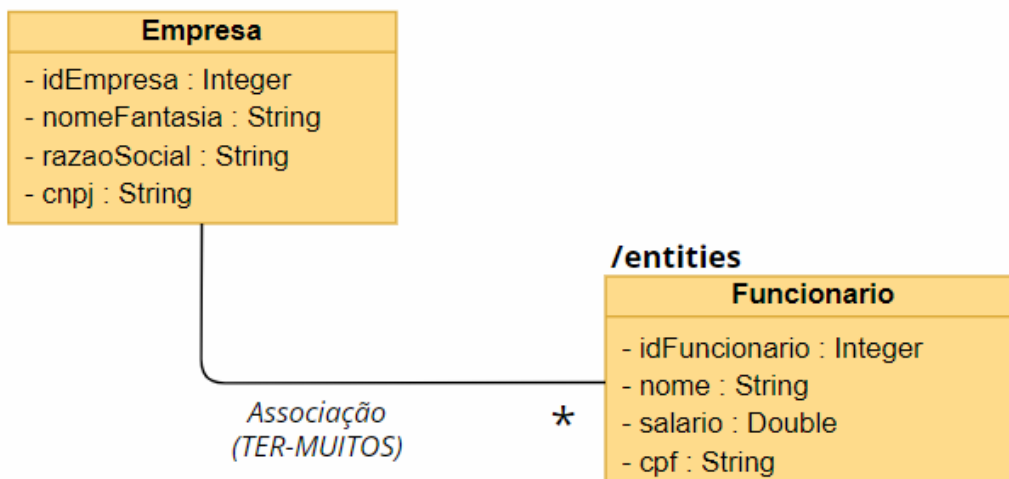
Regra: Uma empresa pode ter muitos funcionários.

JAVABEAN

Nome dado a classes de um projeto Java voltadas para **modelagem de entidades** do sistema, são caracterizados pelos seguintes itens:

- Atributos privados
- Sobrecarga de construtores
 - Construtor sem argumentos
 - Construtor com entrada de argumentos
- Métodos de encapsulamento
 - Set (entrada de dados / atribuição)
 - Get (saída de dados / retorno)
- Sobrescrita de métodos da classe Object
 - toString()
 - equals()
 - hashCode()

/entities



/entities/**Funcionario.java**

```
package entities;

public class Funcionario {

    private Integer idFuncionario;
    private String nome;
    private Double salario;
    private String cpf;

    public Funcionario() {
        // TODO Auto-generated constructor stub
    }

    public Funcionario(Integer idFuncionario, String nome,
        Double salario, String cpf) {
        super();
        this.idFuncionario = idFuncionario;
        this.nome = nome;
        this.salario = salario;
        this.cpf = cpf;
    }

    public Integer getIdFuncionario() {
        return idFuncionario;
    }

    public void setIdFuncionario(Integer idFuncionario) {
        this.idFuncionario = idFuncionario;
    }

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public Double getSalario() {
        return salario;
    }

    public void setSalario(Double salario) {
        this.salario = salario;
    }

    public String getCpf() {
        return cpf;
    }
}
```

```
public void setCpf(String cpf) {
    this.cpf = cpf;
}

@Override
public String toString() {
    return "Funcionario [idFuncionario=" + idFuncionario
        + ", nome=" + nome + ", salario=" + salario + ", cpf="
        + cpf + "]";
}
}
```

Collections

São classes e interfaces do Java utilizados para implementação de coleções de objetos, por exemplo: Listas, filas, mapas etc.

/entities/Empresa.java

```
package entities;

import java.util.List;

public class Empresa {

    private Integer idEmpresa;
    private String razaoSocial;
    private String nomeFantasia;
    private String cnpj;
    private List<Funcionario> funcionarios;

    public Empresa() {
        // TODO Auto-generated constructor stub
    }

    public Empresa(Integer idEmpresa, String razaoSocial,
        String nomeFantasia, String cnpj,
        List<Funcionario> funcionarios) {
        super();
        this.idEmpresa = idEmpresa;
        this.razaoSocial = razaoSocial;
        this.nomeFantasia = nomeFantasia;
        this.cnpj = cnpj;
        this.funcionarios = funcionarios;
    }

    public Integer getIdEmpresa() {
        return idEmpresa;
    }
}
```

```
public void setIdEmpresa(Integer idEmpresa) {
    this.idEmpresa = idEmpresa;
}

public String getRazaoSocial() {
    return razaoSocial;
}

public void setRazaoSocial(String razaoSocial) {
    this.razaoSocial = razaoSocial;
}

public String getNomeFantasia() {
    return nomeFantasia;
}

public void setNomeFantasia(String nomeFantasia) {
    this.nomeFantasia = nomeFantasia;
}

public String getCnpj() {
    return cnpj;
}

public void setCnpj(String cnpj) {
    this.cnpj = cnpj;
}

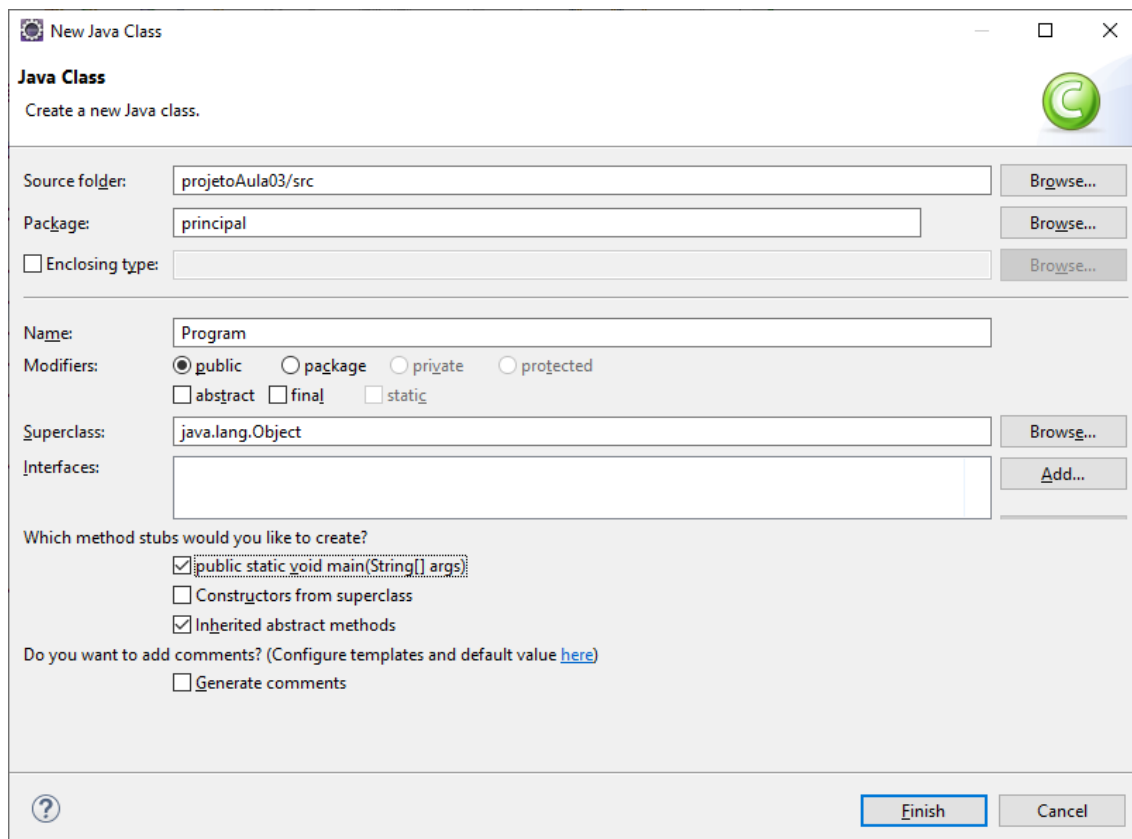
public List<Funcionario> getFuncionarios() {
    return funcionarios;
}

public void setFuncionarios(List<Funcionario> funcionarios) {
    this.funcionarios = funcionarios;
}

@Override
public String toString() {
    return "Empresa [idEmpresa=" + idEmpresa
        + ", razaoSocial=" + razaoSocial + ", nomeFantasia="
        + nomeFantasia
        + ", cnpj=" + cnpj + ", funcionarios="
        + funcionarios + "]\n";
}
}
```

/principal/**Program.java**

Classe para executar o projeto. Para isso ela deverá ter o método **void main(String[] args)**



java.util.Scanner

Classe Java utilizada para fazer captura de dados informados pelo usuário em aplicações locais.

```
package principal;

import java.util.Scanner;

import entidades.Empresa;

public class Program {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("\n *** EXPORTADOR DE DADOS *** \n");
        System.out.println("ENTRE COM OS DADOS DA EMPRESA:");

        //criando uma variável de instância para empresa:
        Empresa empresa = new Empresa();
```

```

System.out.print("ID da empresa.....: ");
empresa.setIdEmpresa
    (Integer.parseInt(scanner.nextLine()));

System.out.print("Nome Fantasia da empresa.....: ");
empresa.setNomeFantasia(scanner.nextLine());

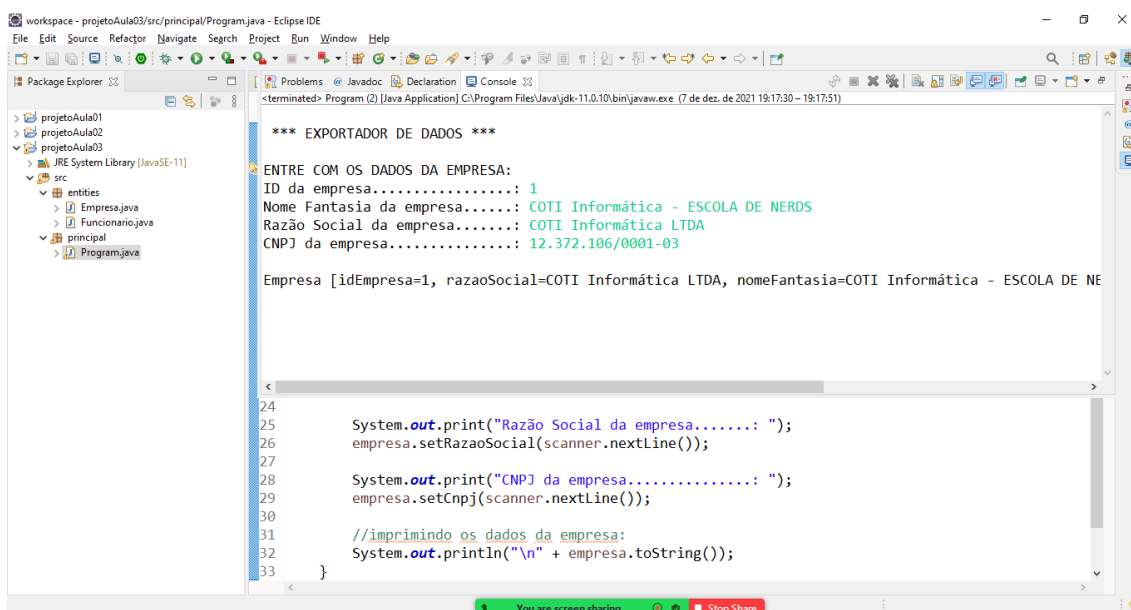
System.out.print("Razão Social da empresa.....: ");
empresa.setRazaoSocial(scanner.nextLine());

System.out.print("CNPJ da empresa.....: ");
empresa.setCnpj(scanner.nextLine());

//imprimindo os dados da empresa:
System.out.println("\n" + empresa.toString());
    }
}

```

Executando:



*** EXPORTADOR DE DADOS ***

ENTRE COM OS DADOS DA EMPRESA:

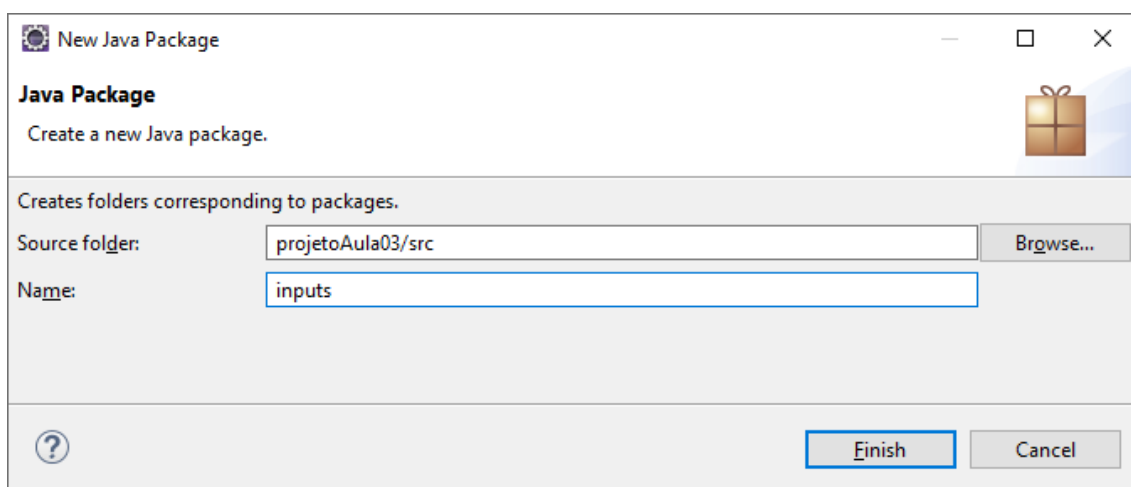
ID da empresa.....: 1
 Nome Fantasia da empresa.....: COTI Informática - ESCOLA DE NERDS
 Razão Social da empresa.....: COTI Informática LTDA
 CNPJ da empresa.....: 12.372.106/0001-03

Empresa [idEmpresa=1, razaoSocial=COTI Informática LTDA, nomeFantasia=COTI Informática - ESCOLA DE NERDS, cnpj=12.372.106/0001-03, funcionarios=null]



PRINCÍPIO DA RESPONSABILIDADE ÚNICA

De acordo com o princípio de SRP, iremos criar uma classe voltada especificamente para ler os dados de uma empresa e, posteriormente, iremos utilizá-la no método main() da classe Program.



```
package inputs;
```

```
import java.util.Scanner;
```

```
import entities.Empresa;
```

```
public class EmpresaInput {
```

```
    // método para ler e retornar todos os dados de uma empresa  
    // capturados através do console do eclipse (Scanner)
```

```
    public Empresa read() { // read() -> leitura
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        // criando uma variável de instância para empresa:  
        Empresa empresa = new Empresa();
```

```
        System.out.println("ENTRE COM OS DADOS DA EMPRESA:");
```

```
        System.out.print("ID da empresa.....: ");  
        empresa.setIdEmpresa(Integer.parseInt(  
            scanner.nextLine()));
```



```

System.out.print("Nome Fantasia da empresa.....: ");
empresa.setNomeFantasia(scanner.nextLine());

System.out.print("Razão Social da empresa.....: ");
empresa.setRazaoSocial(scanner.nextLine());

System.out.print("CNPJ da empresa.....: ");
empresa.setCnpj(scanner.nextLine());

return empresa;
}
}

```

Voltando na classe Program.java

```

package principal;

import entities.Empresa;
import inputs.EmpresaInput;

public class Program {

    public static void main(String[] args) {

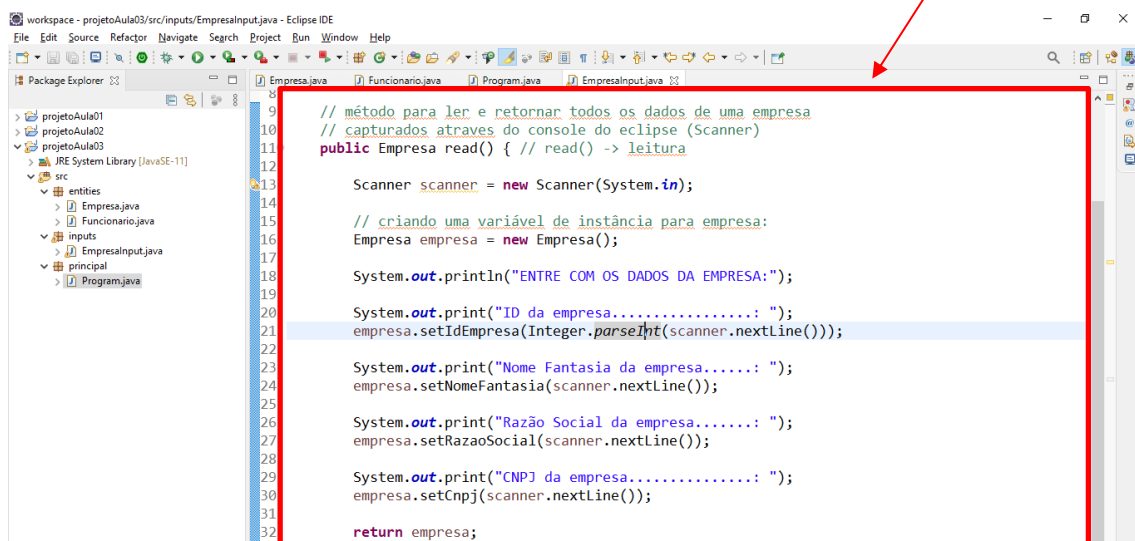
        System.out.println("\n *** EXPORTADOR DE DADOS *** \n");

        //Criando um objeto da classe EmpresaInput
        EmpresaInput empresaInput = new EmpresaInput();
        Empresa empresa = empresaInput.read();

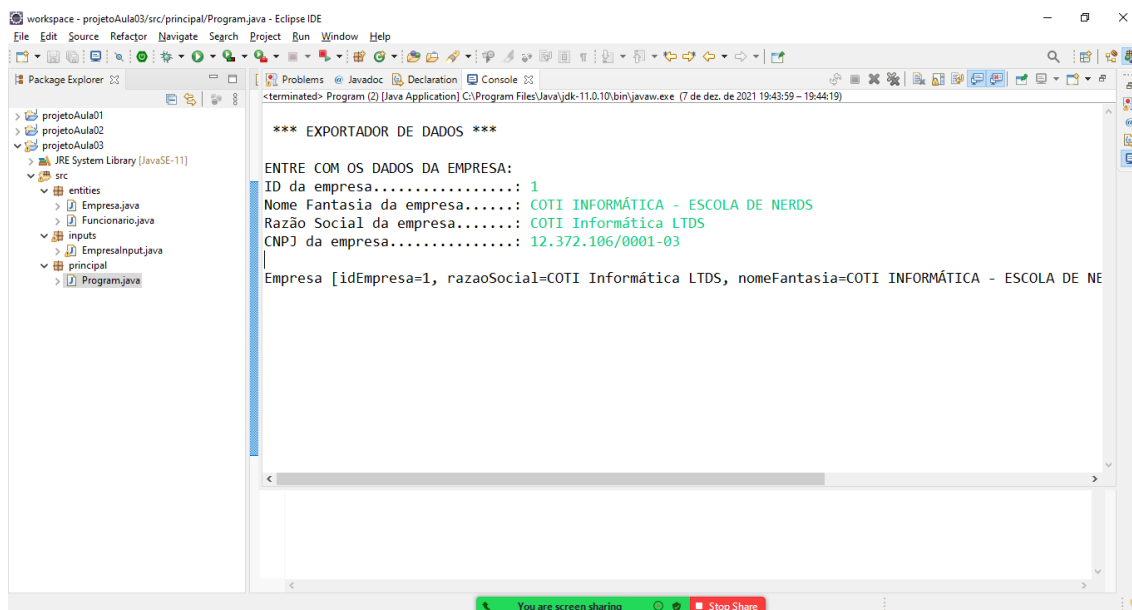
        //imprimindo os dados da empresa:
        System.out.println("\n" + empresa.toString());

    }
}

```



Executando:



*** EXPORTADOR DE DADOS ***

ENTRE COM OS DADOS DA EMPRESA:

ID da empresa.....: 1

Nome Fantasia da empresa.....: COTI INFORMÁTICA - ESCOLA DE NERDS

Razão Social da empresa.....: COTI Informática LTDS

CNPJ da empresa.....: 12.372.106/0001-03

Empresa [idEmpresa=1, razaoSocial=COTI Informática LTDS, nomeFantasia=COTI INFORMÁTICA - ESCOLA DE NERDS, cnpj=12.372.106/0001-03, funcionarios=null]

Métodos estáticos (**static**)

Métodos declarados com a palavra reservada static não precisam de uma variável de instancia para serem executados. São chamados diretamente a partir do nome da classe.

Forma padrão para executar métodos em JAVA:

```
package principal;
```

```
public class Teste {
```

```
    public static void main(String[] args) {
```

```
        Impressao impressao = new Impressao();
```

```
        impressao.imprimir("Sergio");
```


```
    }
```

```
}
```

```
class Impressao {  
  
    public void imprimir(String nome) {  
        System.out.println("Olá, " + nome);  
    }  
}
```

Forma de executar métodos estáticos em JAVA:

```
package principal;  
  
public class Teste {  
  
    public static void main(String[] args) {  
  
        Impressao.imprimir("Sergio Mendes");  
    }  
}  
  
class Impressao {  
  
    public static void imprimir(String nome) {  
        System.out.println("Olá, " + nome);  
    }  
}
```



Note que, o método **imprimir()** está sendo executado sem a necessidade de criarmos uma variável de instancia para a classe Impressao, isto porque o método é estático.

Modificando o método **read()** da classe **EmpresaInput** para ser declarado como método estático.

```
package inputs;  
  
import java.util.Scanner;  
  
import entities.Empresa;  
  
public class EmpresaInput {  
  
    // método para ler e retornar todos os dados de uma empresa  
    // capturados através do console do eclipse (Scanner)  
    public static Empresa read() { // read() -> leitura  
  
        Scanner scanner = new Scanner(System.in);
```

```
// criando uma variável de instância para empresa:
Empresa empresa = new Empresa();

System.out.println("ENTRE COM OS DADOS DA EMPRESA:");

System.out.print("ID da empresa.....: ");
empresa.setIdEmpresa(Integer.parseInt(
    scanner.nextLine()));

System.out.print("Nome Fantasia da empresa.....: ");
empresa.setNomeFantasia(scanner.nextLine());

System.out.print("Razão Social da empresa.....: ");
empresa.setRazaoSocial(scanner.nextLine());

System.out.print("CNPJ da empresa.....: ");
empresa.setCnpj(scanner.nextLine());

return empresa;
}
}
```

Voltando na classe Program.java

```
package principal;

import entities.Empresa;
import inputs.EmpresaInput;

public class Program {

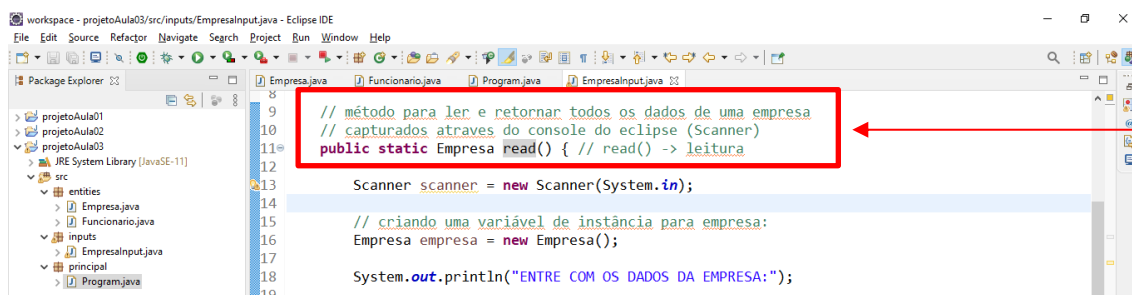
    public static void main(String[] args) {

        System.out.println("\n *** EXPORTADOR DE DADOS *** \n");

        Empresa empresa = EmpresaInput.read();

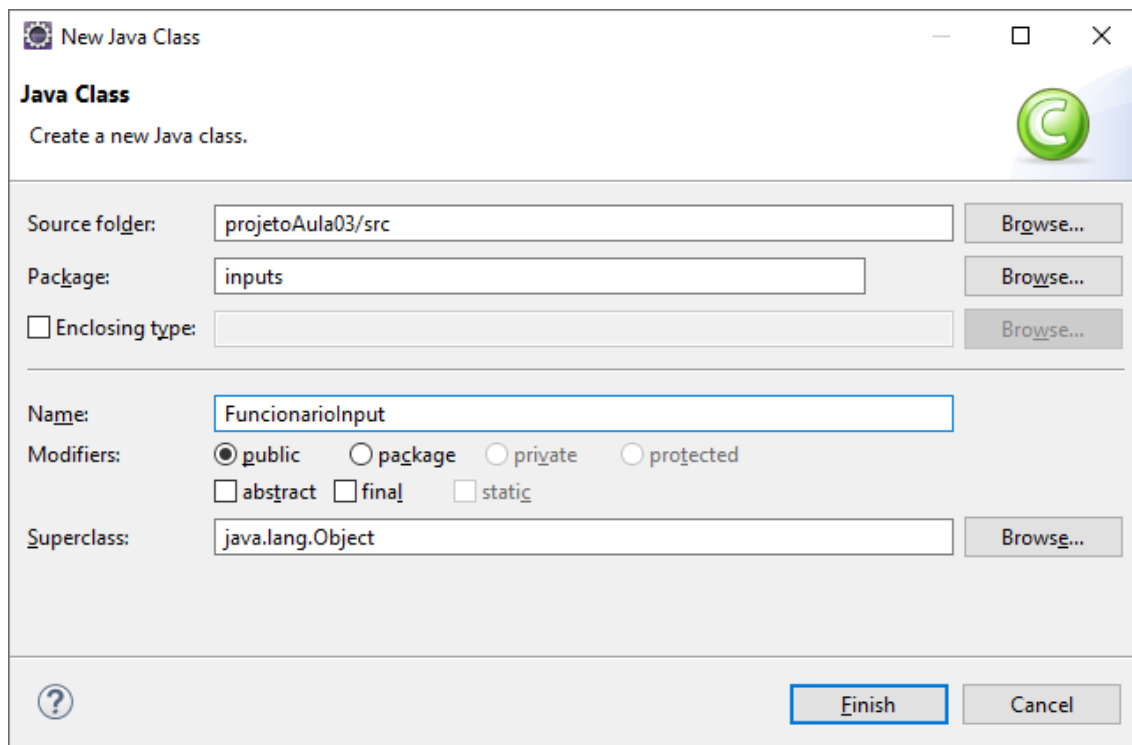
        //imprimindo os dados da empresa:
        System.out.println("\n" + empresa.toString());

    }
}
```



Criando uma classe Input para ler os dados de um funcionário informado pelo usuário do console:

/inputs/**FuncionarioInput.java**



```
package inputs;

import java.util.Scanner;
import entities.Funcionario;

public class FuncionarioInput {

    //método estático para capturar os dados de um funcionário
    //informados através do console do eclipse (scanner)
    public static Funcionario read() {

        @SuppressWarnings("resource")
        Scanner scanner = new Scanner(System.in);

        Funcionario funcionario = new Funcionario();

        System.out.println("ENTRE COM OS DADOS DO FUNCIONÁRIO:");

        System.out.print("ID do funcionário.....: ");
        funcionario.setIdFuncionario(Integer.parseInt(
            scanner.nextLine()));
    }
}
```

```

System.out.print("Nome do funcionário.....: ");
funcionario.setNome(scanner.nextLine());

System.out.print("Salário do funcionário.....: ");
funcionario.setSalario(Double.parseDouble
(scanner.nextLine()));

System.out.print("CPF do funcionário.....: ");
funcionario.setCpf(scanner.nextLine());

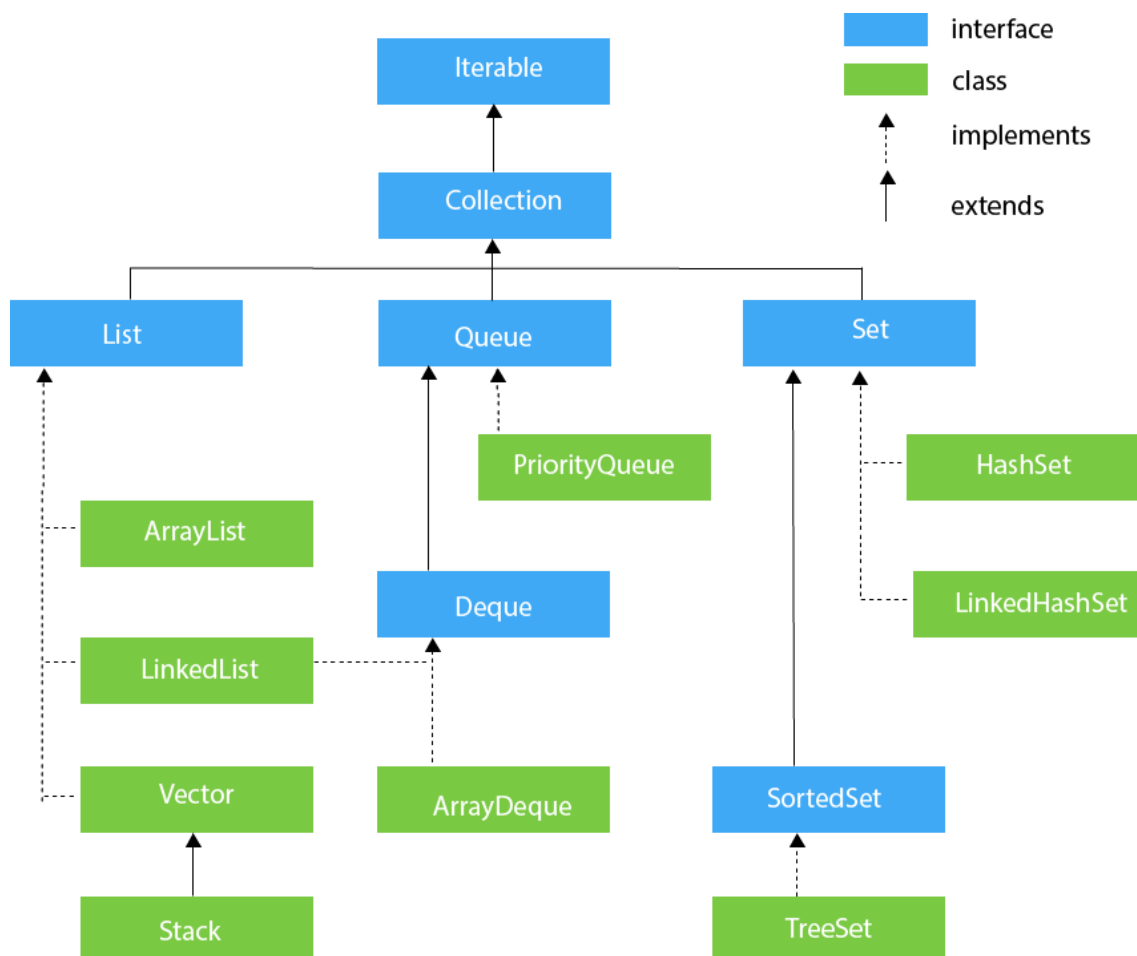
return funcionario;
}
}

```

Principais collections do Java são:

Classes ou interfaces para manipulação de coleções de dados

- **List** Listas de objetos
- **Set** Listas que não permitem objetos duplicados
- **Queue** Organizar filas (Primeiro que entra, primeiro que sai)
- **Map** Mapas baseados em chave / valor



Para o relacionamento entre Empresa e Funcionario optamos por utilizar a interface **List**, que pode ser implementada como um **ArrayList**.

Voltando na classe Program.java

Capturando dados de funcionários.

```
package principal;

import java.util.ArrayList;
import java.util.Scanner;

import entities.Empresa;
import entities.Funcionario;
import inputs.EmpresaInput;
import inputs.FuncionarioInput;

public class Program {

    public static void main(String[] args) {

        @SuppressWarnings("resource")
        Scanner scanner = new Scanner(System.in);

        System.out.println("\n *** EXPORTADOR DE DADOS *** \n");

        //lendo os dados de uma empresa
        Empresa empresa = EmpresaInput.read();

        //inicializando a lista de funcionários do objeto empresa
        empresa.setFuncionarios(new ArrayList<Funcionario>());

        System.out.print("Qtd de funcionários desejado..: ");
        Integer qtd = Integer.parseInt(scanner.nextLine());

        for(int i = 1; i <= qtd; i++) {

            System.out.println("\n" + i + "º Funcionário:");

            Funcionario funcionario = FuncionarioInput.read();

            //adicionando o funcionário na empresa
            empresa.getFuncionarios().add(funcionario);
        }

        //imprimindo os dados da empresa:
        System.out.println("\n" + empresa.toString());
    }
}
```

Executando:

*** EXPORTADOR DE DADOS ***

ENTRE COM OS DADOS DA EMPRESA:

ID da empresa.....: 1
 Nome Fantasia da empresa.....: COTI Informática ESCOLA DE NERDS
 Razão Social da empresa.....: COTI Informática LTDA
 CNPJ da empresa.....: 198389123976
 Qtd de funcionários desejado...: 2

1º Funcionário:

ENTRE COM OS DADOS DO FUNCIONÁRIO:

ID do funcionário.....: 1
 Nome do funcionário.....: Ana Paula
 Salário do funcionário.....: 3000
 CPF do funcionário.....: 12367812630

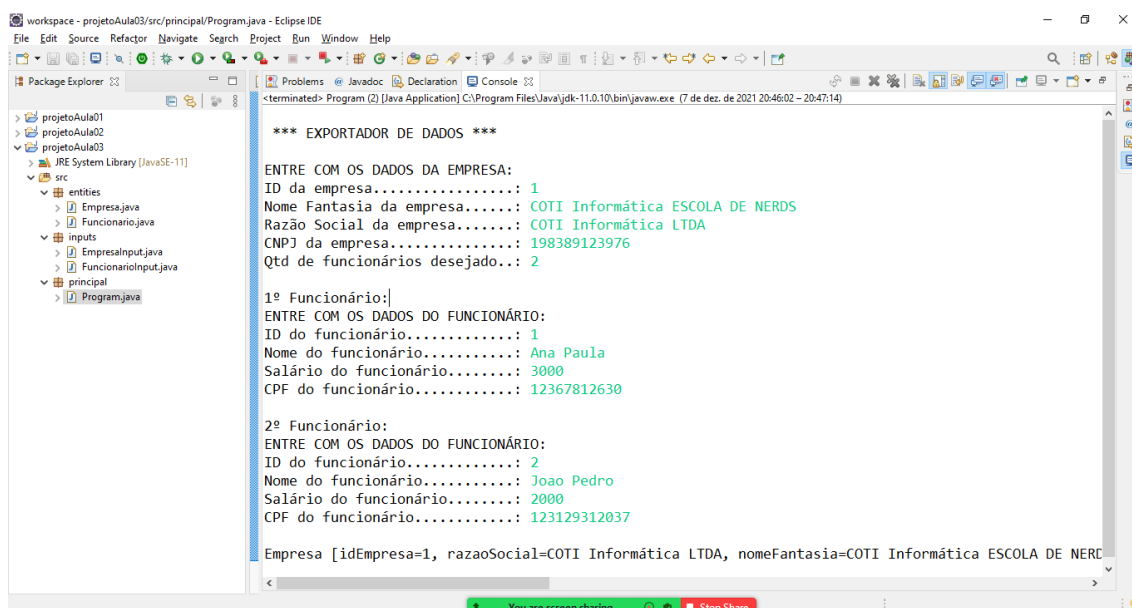
2º Funcionário:

ENTRE COM OS DADOS DO FUNCIONÁRIO:

ID do funcionário.....: 2
 Nome do funcionário.....: Joao Pedro
 Salário do funcionário.....: 2000
 CPF do funcionário.....: 123129312037

Empresa [idEmpresa=1, razaoSocial=COTI Informática LTDA, nomeFantasia=COTI Informática ESCOLA DE NERDS, cnpj=198389123976, funcionarios=[Funcionario [idFuncionario=1, nome=Ana Paula, salario=3000.0, cpf=12367812630], Funcionario [idFuncionario=2, nome=Joao Pedro, salario=2000.0, cpf=123129312037]]]

Exemplo:



```
workspace - projetoAula03/src/principal/Program.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer Problems Javadoc Declaration Console
<terminated> Program (2) [Java Application] C:\Program Files\Java\jdk-11.0.10\bin\javaw.exe (7 de dez. de 2021 20:46:02 - 20:47:14)

*** EXPORTADOR DE DADOS ***

ENTRE COM OS DADOS DA EMPRESA:
ID da empresa.....: 1
Nome Fantasia da empresa.....: COTI Informática ESCOLA DE NERDS
Razão Social da empresa.....: COTI Informática LTDA
CNPJ da empresa.....: 198389123976
Qtd de funcionários desejado...: 2

1º Funcionário:
ENTRE COM OS DADOS DO FUNCIONÁRIO:
ID do funcionário.....: 1
Nome do funcionário.....: Ana Paula
Salário do funcionário.....: 3000
CPF do funcionário.....: 12367812630

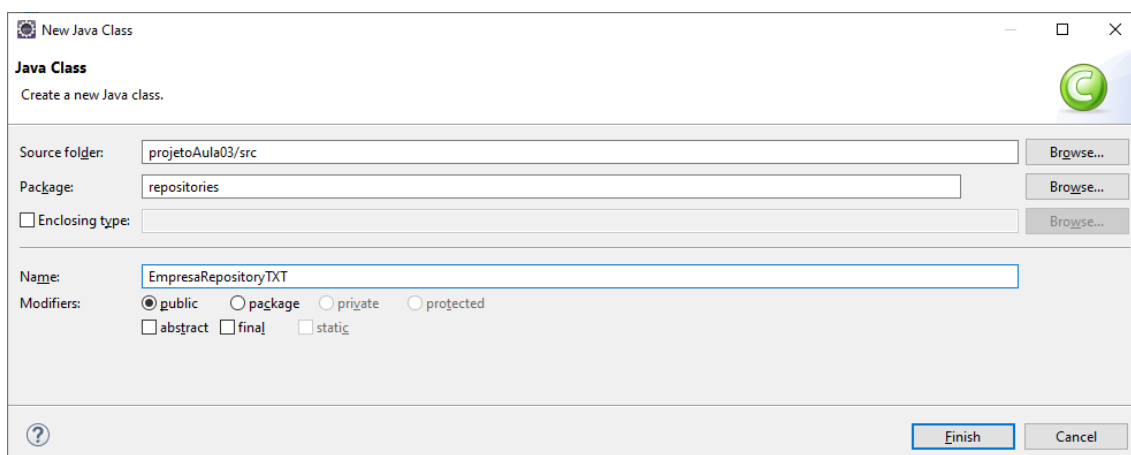
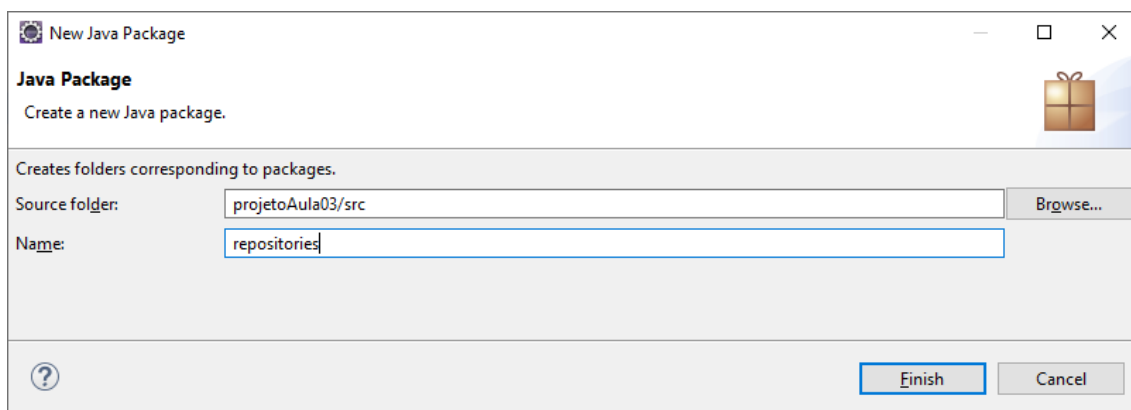
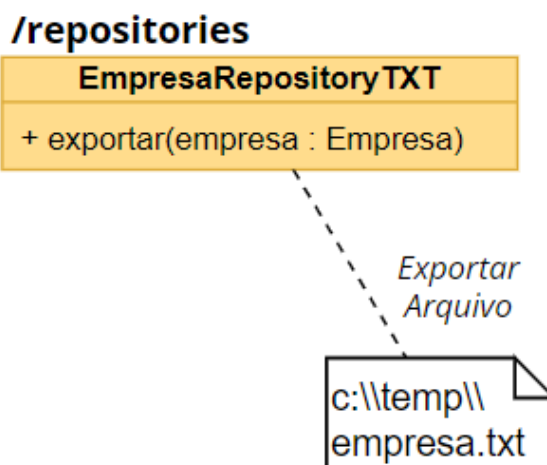
2º Funcionário:
ENTRE COM OS DADOS DO FUNCIONÁRIO:
ID do funcionário.....: 2
Nome do funcionário.....: Joao Pedro
Salário do funcionário.....: 2000
CPF do funcionário.....: 123129312037

Empresa [idEmpresa=1, razaoSocial=COTI Informática LTDA, nomeFantasia=COTI Informática ESCOLA DE NERD
```


Repository

Nome comumente utilizado para designar classes em projetos Java que irão fazer algum tipo de armazenamento de dados em meio físico, como arquivos, bancos de dados etc.

** Iremos criar uma classe de repositório para exportar os dados de empresa para um arquivo de extensão .TXT



```
package repositories;

import entities.Empresa;

public class EmpresaRepositoryTXT {

    //método para receber um objeto Empresa
    //e exportar os seus dados para um arquivo
    //de extensão .TXT
    public void exportar(Empresa empresa) {

    }

}
```

Tratamento de exceções

Em Java, exceções são erros que ocorrem em tempo de execução, ou seja, o código-fonte está compilado com sucesso mas durante a sua execução algum erro inesperado ocorre.

Quando criamos métodos em Java podemos declarar os métodos com uma diretiva que irá obrigar o programador a, quando o método for chamado, usar um bloco de tratamento de exceções chamado try / catch.

throws Exception

É uma diretiva utilizada na declaração de métodos, e faz com que o programador seja obrigado a utilizar um bloco try / catch na chamada do método (ou um outro throws Exception). Exemplo:

Execução de método comum:

```
package principal;

public class Teste {

    public static void main(String[] args) {

        A a = new A();
        a.imprimir();

    }

}

class A {

    public void imprimir() {
        System.out.println("Hello World!");
    }

}
```

Utilizando o **throws Exception**, a execução do método deverá ser feita obrigatoriamente dentro de um bloco try e catch ou de um outro método throws Exception:

```
package principal;

public class Teste {

    public static void main(String[] args) {

        A a = new A();

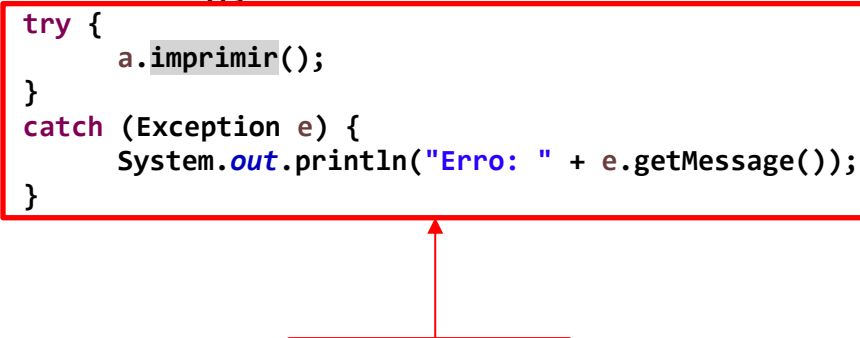
        try {
            a.imprimir();
        }
        catch (Exception e) {
            System.out.println("Erro: " + e.getMessage());
        }
    }
}

class A {
    public void imprimir() throws Exception {
        System.out.println("Hello World!");
    }
}
```

Note que, a chamada do método imprimir() precisou ser feita dentro de um bloco try e catch pois o método foi criado com a diretiva throws Exception.

```
package principal;
public class Teste {
    public static void main(String[] args) {
        A a = new A();
        try {
            a.imprimir();
        }
        catch (Exception e) {
            System.out.println("Erro: " + e.getMessage());
        }
    }
}

class A {
    public void imprimir() throws Exception {
        System.out.println("Hello World!");
    }
}
```



Voltando na classe **EmpresaRepositoryTXT.java**

```
package repositories;

import java.io.File;
import java.io.PrintWriter;

import entities.Empresa;
import entities.Funcionario;

public class EmpresaRepositoryTXT {

    // método para receber um objeto Empresa
    // e exportar os seus dados para um arquivo
    // de extensão .TXT
    public void exportar(Empresa empresa) throws Exception {

        PrintWriter print = new PrintWriter
            (new File("c:\\temp\\empresa.txt"));

        // imprimindo os dados da empresa:
        print.write("\nDADOS DA EMPRESA:\n");

        print.write("ID DA EMPRESA.....: "
            + empresa.getIdEmpresa());

        print.write("NOME FANTASIA.....: "
            + empresa.getNomeFantasia());

        print.write("RAZÃO SOCIAL.....: "
            + empresa.getRazaoSocial());

        print.write("CNPJ.....: "
            + empresa.getCnpj());

        print.write("QTD DE FUNCIONÁRIOS...: "
            + empresa.getFuncionarios().size());

        // imprimindo os dados dos funcionários:
        print.write("\nDADOS DOS FUNCIONÁRIOS:\n");

        for(Funcionario funcionario : empresa.getFuncionarios()) {

            print.write("ID DO FUNCIONÁRIO.....: "
                + funcionario.getIdFuncionario());

            print.write("NOME.....: "
                + funcionario.getNome());

            print.write("SALÁRIO.....: "
                + funcionario.getSalario());
```

```
        print.write("CPF.....: "
            + funcionario.getCpf());

        print.write("---");
    }

    //salvar e fechar o arquivo
    print.flush();
    print.close();
}
}
```

Testando:

Voltando na classe **Program.java**

```
package principal;

import java.util.ArrayList;
import java.util.Scanner;

import entities.Empresa;
import entities.Funcionario;
import inputs.EmpresaInput;
import inputs.FuncionarioInput;
import repositories.EmpresaRepositoryTXT;

public class Program {

    public static void main(String[] args) {

        @SuppressWarnings("resource")
        Scanner scanner = new Scanner(System.in);

        System.out.println("\n *** EXPORTADOR DE DADOS *** \n");

        //lendo os dados de uma empresa
        Empresa empresa = EmpresaInput.read();
        //inicializando a lista de funcionários do objeto empresa
        empresa.setFuncionarios(new ArrayList<Funcionario>());

        System.out.print("Qtd de funcionários desejado..: ");
        Integer qtd = Integer.parseInt(scanner.nextLine());

        for(int i = 1; i <= qtd; i++) {

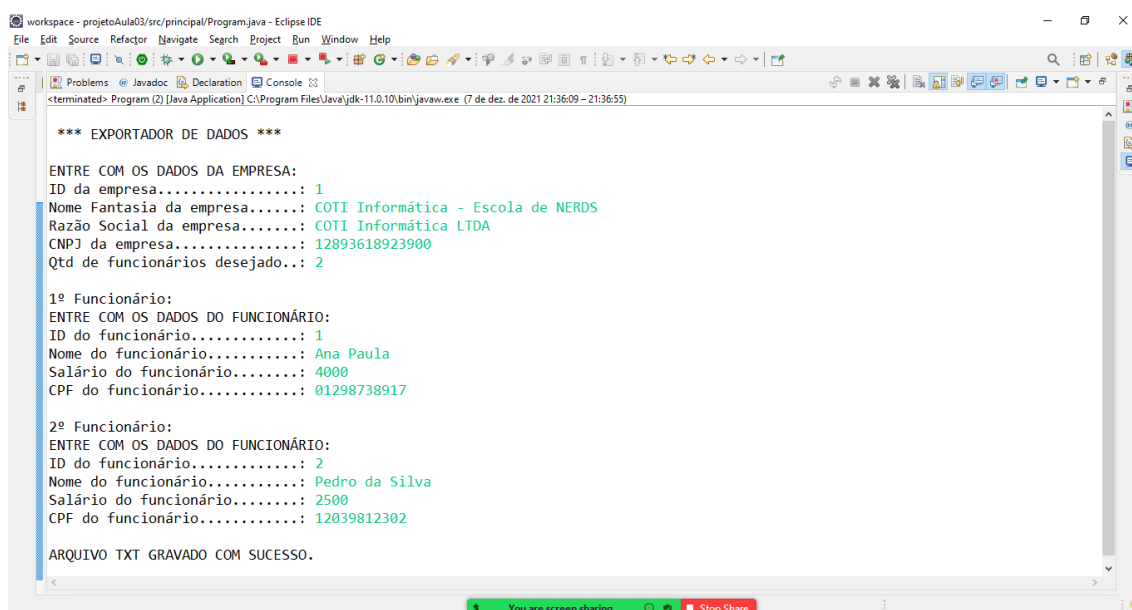
            System.out.println("\n" + i + "º Funcionário:");
            Funcionario funcionario = FuncionarioInput.read();

            //adicionando o funcionário na empresa
            empresa.getFuncionarios().add(funcionario);
        }

        //criando uma variavel de instancia para a classe do repositório
        EmpresaRepositoryTXT empresaRepository
            = new EmpresaRepositoryTXT();
    }
}
```

```
try {
    empresaRepository.exportar(empresa);
    System.out.println
        ("\nARQUIVO TXT GRAVADO COM SUCESSO.");
}
catch(Exception e) {
    System.out.println("\nERRO: " + e.getMessage());
}
}
```

Testando:



*** EXPORTADOR DE DADOS ***

ENTRE COM OS DADOS DA EMPRESA:

ID da empresa.....: 1

Nome Fantasia da empresa.....: COTI Informática - Escola de NERDS

Razão Social da empresa.....: COTI Informática LTDA

CNPJ da empresa.....: 12893618923900

Qtd de funcionários desejado..: 2

1º Funcionário:

ENTRE COM OS DADOS DO FUNCIONÁRIO:

ID do funcionário.....: 1

Nome do funcionário.....: Ana Paula

Salário do funcionário.....: 4000

CPF do funcionário.....: 01298738917

2º Funcionário:

ENTRE COM OS DADOS DO FUNCIONÁRIO:

ID do funcionário.....: 2

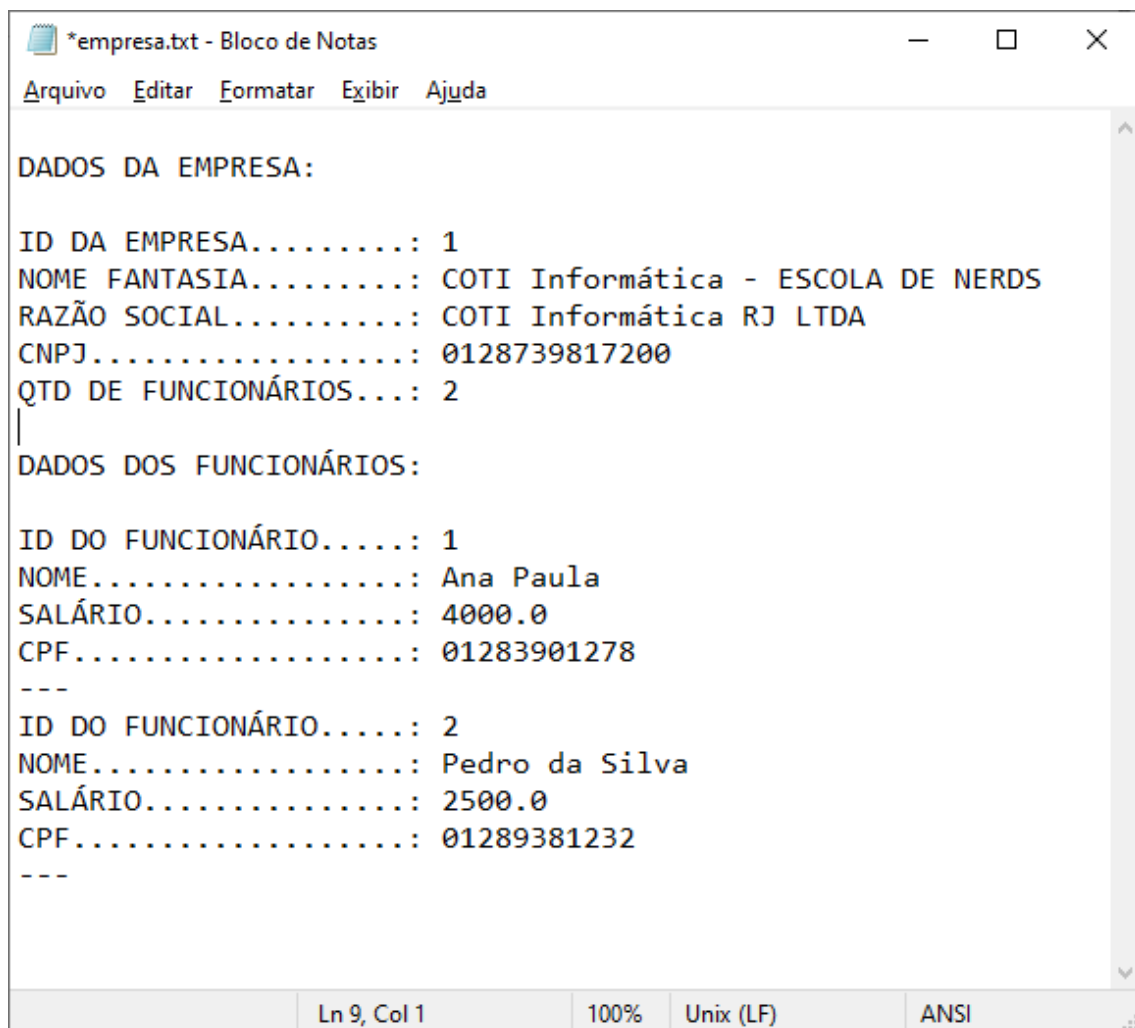
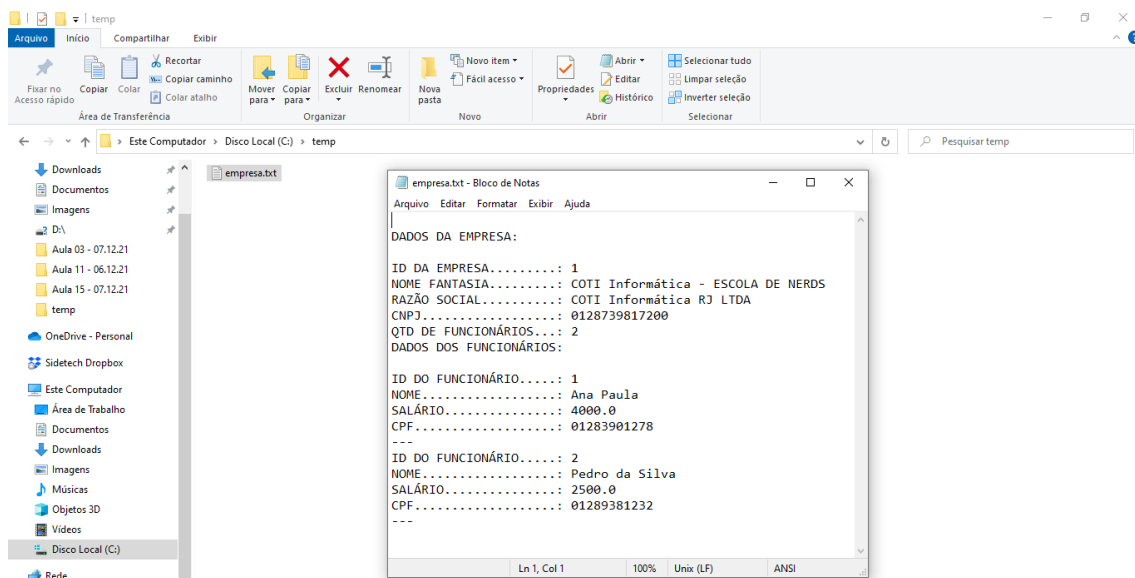
Nome do funcionário.....: Pedro da Silva

Salário do funcionário.....: 2500

CPF do funcionário.....: 12039812302

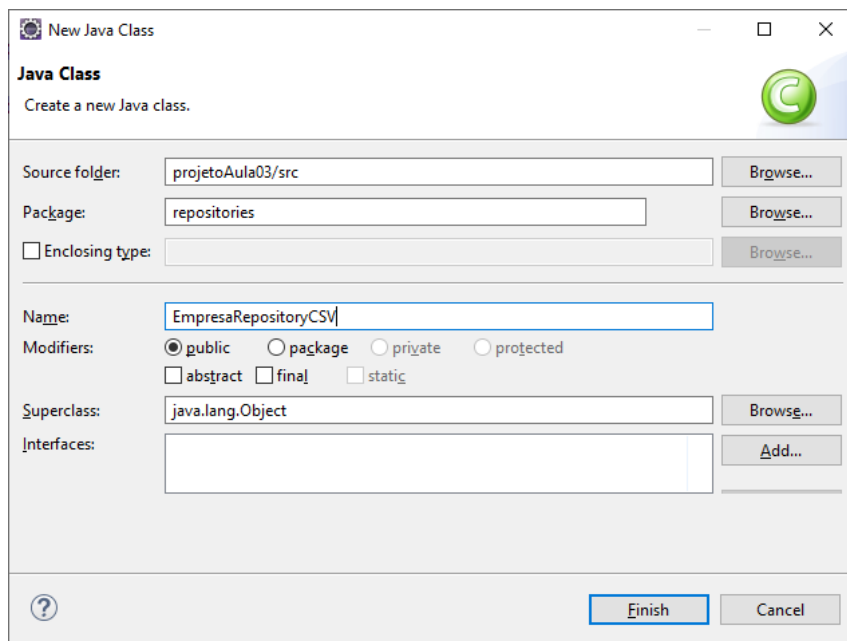
ARQUIVO TXT GRAVADO COM SUCESSO.

Arquivo gerado:



Criando uma classe para exportar os dados da Empresa para formato CSV (Excel)

/Repositories/**EmpresaRepositoryCSV.java**



/repositories

EmpresaRepositoryCSV

+ exportar(empresa : Empresa)

Exportar
Arquivo

c:\\temp\\
empresa.csv

```
package repositories;

import java.io.File;
import java.io.PrintWriter;

import entities.Empresa;
import entities.Funcionario;

public class EmpresaRepositoryCSV {

    public void exportar(Empresa empresa) throws Exception {

        PrintWriter print = new PrintWriter
            (new File("c:\\temp\\empresa.csv"));
```



```
//escrever os dados da empresa
print.write(empresa.getIdEmpresa() + ";");
print.write(empresa.getNomeFantasia() + ";");
print.write(empresa.getRazaoSocial() + ";");
print.write(empresa.getCnpj() + "\n");

//escrever os dados dos funcionários
for(Funcionario funcionario : empresa.getFuncionarios()) {

    print.write(funcionario.getIdFuncionario() + ";");
    print.write(funcionario.getNome() + ";");
    print.write(funcionario.getSalario() + ";");
    print.write(funcionario.getCpf() + "\n");

}

print.flush();
print.close();
}
```

Testando na classe **Program.java**

```
package principal;

import java.util.ArrayList;
import java.util.Scanner;

import entities.Empresa;
import entities.Funcionario;
import inputs.EmpresaInput;
import inputs.FuncionarioInput;
import repositories.EmpresaRepositoryCSV;
import repositories.EmpresaRepositoryTXT;

public class Program {

    public static void main(String[] args) {

        @SuppressWarnings("resource")
        Scanner scanner = new Scanner(System.in);

        System.out.println("\n *** EXPORTADOR DE DADOS *** \n");

        //lendo os dados de uma empresa
        Empresa empresa = EmpresaInput.read();
        //inicializando a lista de funcionários do objeto empresa
        empresa.setFuncionarios(new ArrayList<Funcionario>());

        System.out.print("Qtd de funcionários desejado..: ");
        Integer qtd = Integer.parseInt(scanner.nextLine());

        for(int i = 1; i <= qtd; i++) {

            System.out.println("\n" + i + "º Funcionário:");
            Funcionario funcionario = FuncionarioInput.read();
```

```
//adicionando o funcionário na empresa
empresa.getFuncionarios().add(funcionario);
}

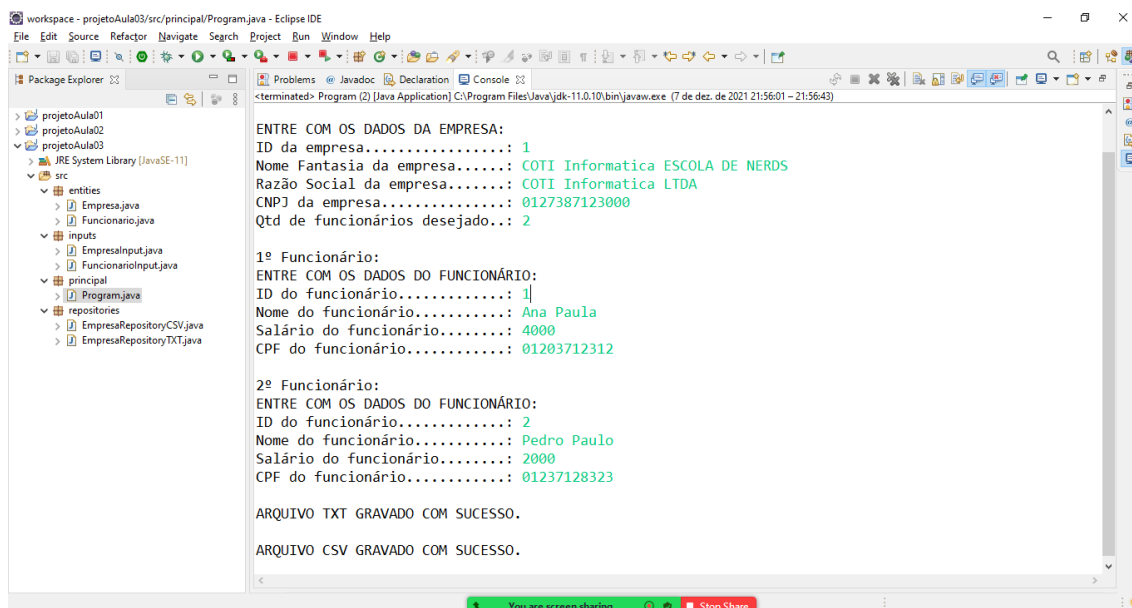
//criando uma variavel de instancia para a classe do repositorio
EmpresaRepositoryTXT empresaRepositoryTXT
= new EmpresaRepositoryTXT();

EmpresaRepositoryCSV empresaRepositoryCSV
= new EmpresaRepositoryCSV();

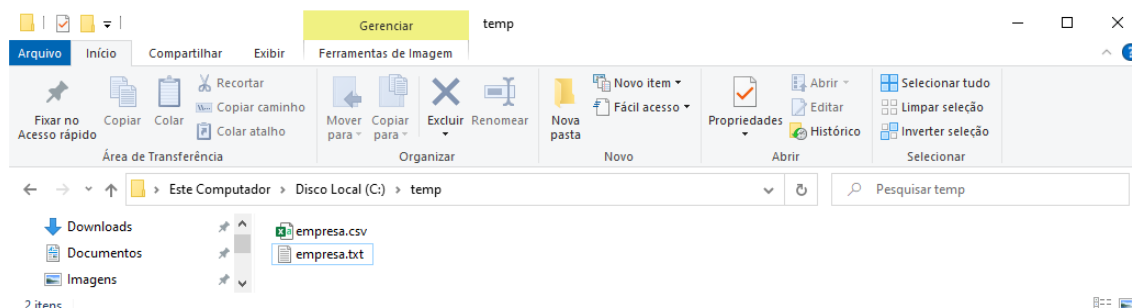
try {
    empresaRepositoryTXT.exportar(empresa);
    System.out.println("\nARQUIVO TXT GRAVADO COM SUCESSO.");

    empresaRepositoryCSV.exportar(empresa);
    System.out.println("\nARQUIVO CSV GRAVADO COM SUCESSO.");
}
catch(Exception e) {
    System.out.println("\nERRO: " + e.getMessage());
}
}
```

Resultado:



Arquivos gerados:



```

empresa.txt - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda

DADOS DA EMPRESA:

ID DA EMPRESA.....: 1
NOME FANTASIA.....: COTI Informatica ESCOLA DE NERDS
RAZÃO SOCIAL.....: COTI Informatica LTDA
CNPJ.....: 0127387123000
QTD DE FUNCIONÁRIOS...: 2
DADOS DOS FUNCIONÁRIOS:

ID DO FUNCIONÁRIO.....: 1
NOME.....: Ana Paula
SALÁRIO.....: 4000.0
CPF.....: 01203712312
---
ID DO FUNCIONÁRIO.....: 2
NOME.....: Pedro Paulo
SALÁRIO.....: 2000.0
CPF.....: 01237128323
---
Ln 1, Col 1 100% Unix (LF) ANSI

```

Salvamento Automático empresa.csv Pesquisar (Alt+Q) Sergio Mendes

Arquivo Página Inicial Inserir Layout da Página Fórmulas Dados Revisão Exibir Ajuda Comentários Compartilhar

Desfazer Área de Transfer... Fonte Alinhamento Número Estilos Células Edição

	A	B	C	D
1	1	COTI Informatica ESCOLA DE NERDS	COTI Informatica LTDA	1,27387E+11
2	1	Ana Paula	4000.0	1203712312
3	2	Pedro Paulo	2000.0	1237128323
4				

empresa Pronto 175%