



C125 – Programação Orientada a Objetos com Java

Pacotes

Prof. Phyllipe Lima
phyllipe@inatel.br

1



Agenda



- ☕ Usar Pacotes para organizar as Classes Java
- ☕ Entender o conceito de *fully-qualified-name* da Classe Java
- ☕ Verificar o que são **imports**

2

Organização de Classes



- Imagine que você tenha criado uma Classe “Matematica”, com alguns métodos que realizam operações matemáticas.
- Agora imagine que seu colega também criou uma classe “Matematica”, com algumas melhorias, e deseja lhe passar essa classe.
- Como o seu projeto pode lidar com classes que tenham o mesmo nome?

Organização de Classes



<pre>//Sua classe public class Matematica { int soma(int x, int y) { return x + y; } int subtrair(int x, int y) { return x - y; } double dividir(int x, int y) { return x/y; } }</pre>	<pre>//Classe do seu amigo public class Matematica { int soma(int x, int y) { return x + y; } int subtrair(int x, int y) { return x - y; } double dividir(double x, double y) { if(y == 0) { throw new RuntimeException(); //Nao se preocupem com isso } return x/y; } }</pre>
--	--

Organização de Classes



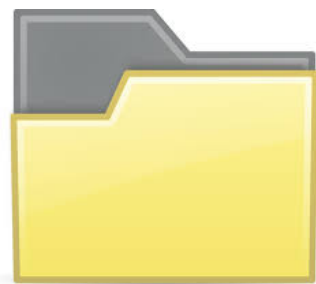
- ☕ Uma primeira ideia seria simplesmente “copiar” o código por cima. Mas essa solução não é escalável.
- ☕ Adicionalmente, a sua classe pode ter outros métodos úteis. Assim você deseja que as duas classes possam coexistir no mesmo projeto.
- ☕ Pensando em sistema operacional, é possível ter dois arquivos com o mesmo nome no mesmo diretório?
 - ☕ Não! E como organizamos nossos arquivos no computador?

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

5

5

Organização de Classes




- ☕ Usamos diretórios
- ☕ No Java, os ***pacotes*** se comportam exatamente como diretórios


C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

6

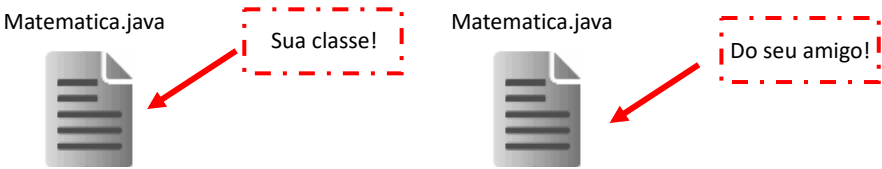
6



Organização de Classes




- Assim como os diretórios contém os nossos arquivos, os pacotes contém as classes.
- Na verdade, classes Java são arquivos com extensão “.java”. Então podemos ver **pacotes** como diretórios!




C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

7

7



Criando pacotes



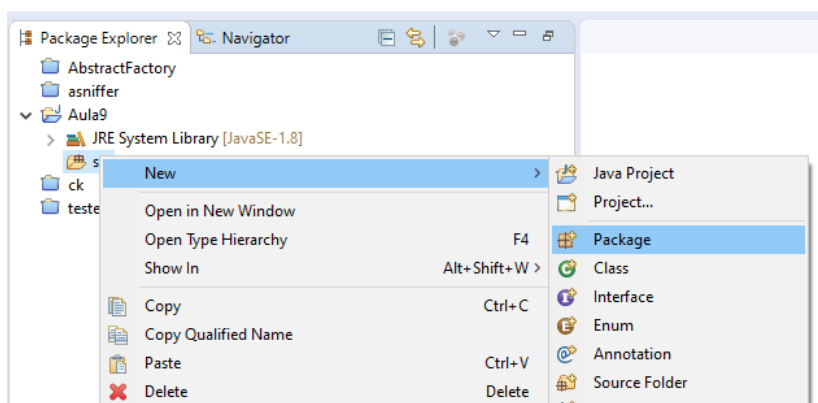
- Vamos criar nossos pacotes!
- O próximo exemplo considera que ainda não criamos a classe Matemática!
- Criaremos um pacote chamado “matemática” para colocar a nossa classe “Matematica”.

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

8

8

Criando pacotes



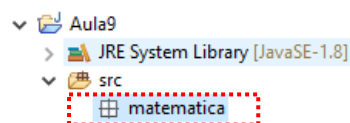
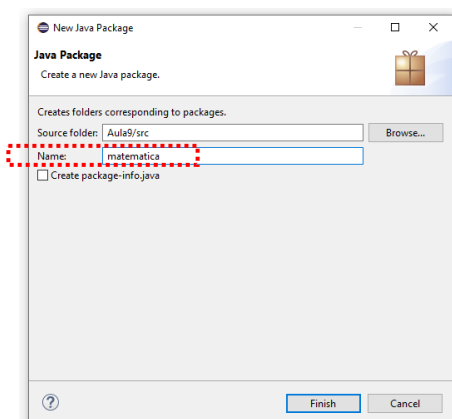
☕ Clique com o botão direito em “src”, depois vá em New -> Package

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

9

9

Criando pacotes



☕ Observe no “Package-Explorer” que agora temos dentro de “src” um novo diretório chamado “matemática”. Está em branco pois não tem nenhuma classe dentro

☕ Dê um nome para o pacote. No exemplo foi colocado “matemática”

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

10

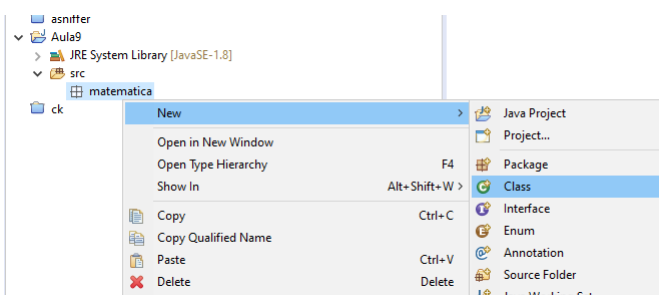
10



Criando pacotes



Vamos adicionar uma nova classe chamada “Matematica” nesse pacote!



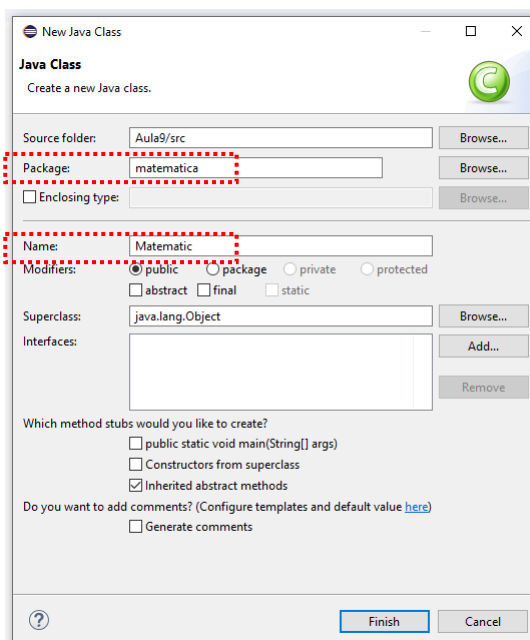
C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

11

11

Observe que o campo “Package” foi preenchido automaticamente!

O nome da classe precisamos preencher manualmente



C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

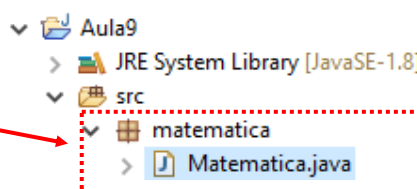
12

12

Criando pacotes



Observe como ficou a estrutura do projeto após a criação da classe



C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

13

13

Criando pacotes



```
package matematica;

public class Matematica {
}
```

Observe a palavra-chave “package”. Ela indica em qual pacote se encontra a classe.

Nesse exemplo a classe “Matematica” se encontra no pacote “matemática”

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

14

14



Criando pacotes – Convenção de Nome



O padrão da Sun (empresa que criou o Java) para dar nome aos pacotes é relativo ao nome da empresa que desenvolveu a classe. Exemplos:

- br.com.nomedaempresa.nomedoprojeto.subpacote
- br.com.nomedaempresa.nomedoprojeto.subpacote2
- br.com.nomedaempresa.nomedoprojeto.subpacote2.subpacote3

Cada “.” (ponto) indica um novo subpacote, isto é, pacote dentro de pacote. Pode ser visto como um diretório dentro de outro.

Vamos alterar o pacote “matemática” para “br.cdg.inatel.matemática”. Veremos como o Eclipse pode nos auxiliar

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

15

15



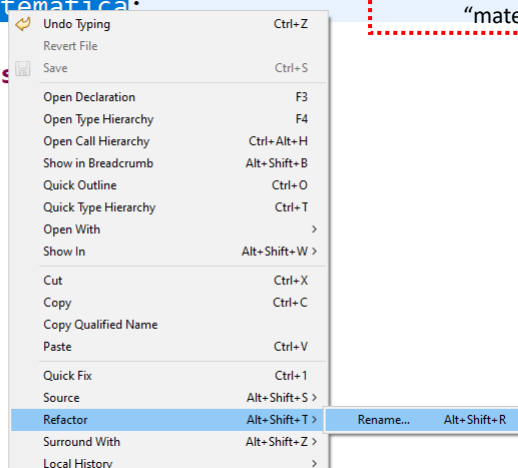
Criando pacotes – Convenção de Nome



package **matematica**;

public class

}



Clique com o Botão Direito em “matematica”

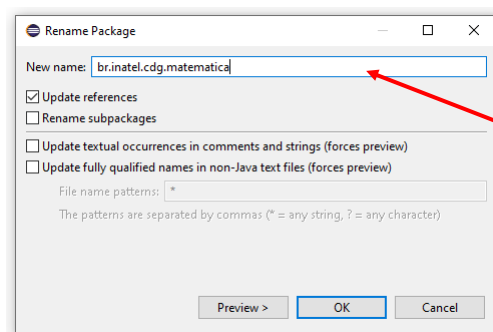
C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

16

16



Criando pacotes – Convenção de Nome

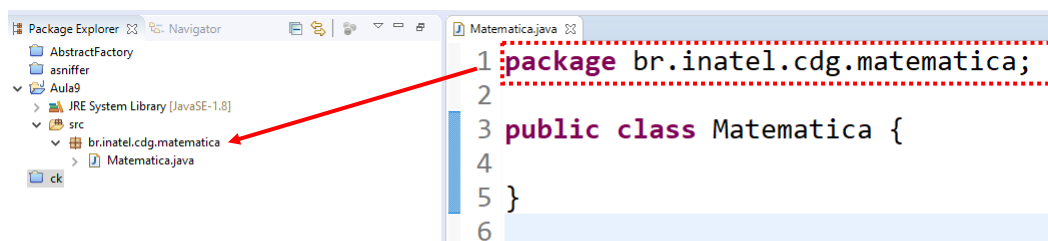


Coloque o novo nome do pacote

17




Criando pacotes – Convenção de Nome




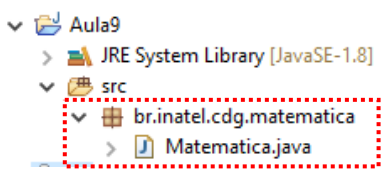
☕ O próprio Eclipse já cuidou de renomear o pacote, além de ter criado a nova estrutura “br.inatel.cdg.matematica”.

18



Criando pacotes – Convenção de Nome







☕ Não confunda e pense que foi criado um único pacote chamado “br.inatel.cdg.matemática”. Na verdade temos 4 pacotes.
☕ br, inatel, cdg, matematica.

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES 19

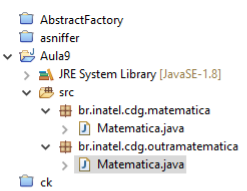
19



Criando Pacotes



☕ Vamos agora criar outro pacote para conter a classe “Matematica” criada pelo seu colega.



```

1 package br.inatel.cdg.outramatematica;
2
3 public class Matematica {
4
5 }
6

```

☕ Agora podemos ter duas classes chamadas “Matematica”, sem problema! Ambas estão em pacotes diferentes

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES 20

20




Fully-Qualified-Name (Nome Completo)




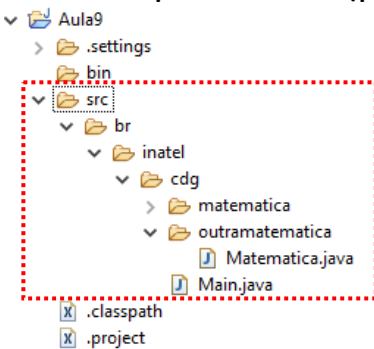
- ☕ Agora que temos as duas classe “Matematica”, como podemos utilizá-las?
- ☕ Precisamos informar o caminho completo da classe que desejamos.
- ☕ Vamos criar uma classe “Main” dentro do pacote br.inatel.cdg, para começarmos a testar




Fully-Qualified-Name (Nome Completo)

 Para reforçar a ideia de diretório, observe como está o projeto na aba “Navigator”. Essa aba mostra a hierarquia de arquivos, exatamente como no Sistema Operacional (pelo Windows Explorer ou Finder).




C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES 23


23



Fully-Qualified-Name (Nome Completo)

 Dentro do método “main” vamos criar uma instância de Matematica, mas qual delas?

```
public static void main(String[] args) {
    //O compilador não sabe qual Matematica queremos usar
    Matematica m = new Matematica();
}
```

 Uma opção é passar o nome completo, ou *Fully-Qualified-Name*.

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES 24

24



Fully-Qualified-Name (Nome Completo)



☕ O Fully-Qualified-Name contém o nome de todos os pacotes onde a classe está presente. Assim, o compilador sabe exatamente qual classe queremos.

```
public static void main(String[] args) {  
    //Utilizando o fully-qualified-name  
    br.inatel.cdg.matematica.Matematica  
    m = new br.inatel.cdg.matematica.Matematica();  
}
```

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

25

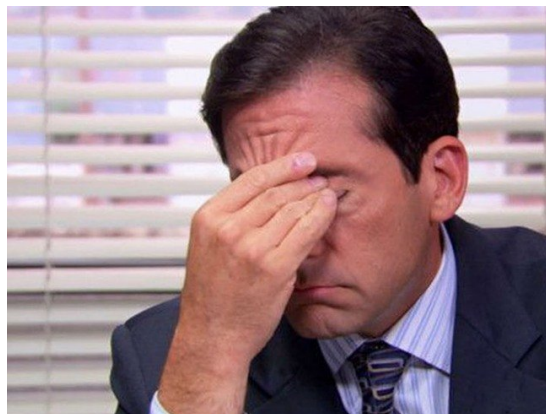
25



Fully-Qualified-Name (Nome Completo)




☕ Parece divertido programar assim!




C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES




26

26



Utilizando Imports




-  Podemos facilitar a nossa vida, utilizando a palavra chave *import*
-  Com ela podemos, no início da classe, descrever quais outras classes de outros pacotes vamos utilizar.
-  O Eclipse nos ajuda com isso!


C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES




27

27



Utilizando Imports

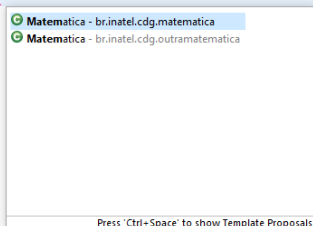


-  Comece a escrever “Matematica” e pressione Ctrl-Espaço
-  O Eclipse irá lhe mostrar algumas opções de potenciais classes.
-  Vamos escolher a primeira opção e pressionar “Enter”

```

public static void main(String[] args) {
    //Utilizando imports
    Matem
}


```




C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

28

28



Utilizando Imports



```

package br.inatel.cdg;

import br.inatel.cdg.matematica.Matematica;

public class Main {

    public static void main(String[] args) {
        //Utilizando imports
        Matematica m = new Matematica();
    }
}

```

O “package” fica logo no início da classe. Ele informa a qual pacote a classe atual (“Main”) pertence.

Em seguida temos o “import”. Informando quais outras classes estamos usando. Repare que ele possui o *fully qualified name* da classe “Matematica”.

Não há dúvidas de qual “Matematica” estamos usando!

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

29



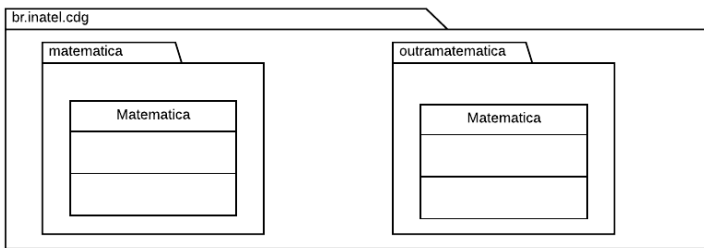


Diagrama UML com Pacotes



Observe que nesse caso não especificamos os membros e métodos das classes, pois estamos interessados na organização dos pacotes. Estamos uma camada acima.



```

graph TD
    br_inatel_cdg[br.inatel.cdg]
    matematica[matematica]
    outramatematica[outramatematica]
    Matematica1[Matematica]
    Matematica2[Matematica]

    br_inatel_cdg --> matematica
    br_inatel_cdg --> outramatematica
    matematica --> Matematica1
    outramatematica --> Matematica2

```

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

30

Exercício 1 – Space Shooter



C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

31

31

Exercício 1 – Space Shooter



- ☞ Crie classes Java que sigam o UML apresentado a seguir.
- ☞ Temos uma estrutura de Classes e uma de Pacotes.
- ☞ Crie getters e setter que julgar necessário
- ☞ Faça os seguintes testes
 - ☞ A Nave pode invocar o método atirar(), recebendo como parâmetro um Asteroide.
 - ☞ Existem dois tipos de asteroides: Pequeno e Grande
 - ☞ A nave possui dois tipos de tiro: Normal e Explosivo
 - ☞ Asteroides do tipo “Grande” são destruídos apenas com Naves que possuem um tipo de tiro “Explosivo”
 - ☞ Se o Asteroide for destruído, ele chama o método destruir(), que apenas imprime uma mensagem

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

32

32



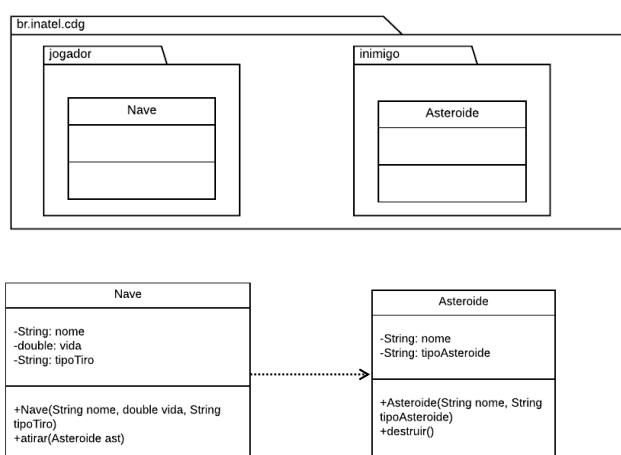
Exercício 1 – Space Shooter



- OBS: No diagrama de classes UML, quando temos uma seta, é uma relação de dependência. Assim, Nave depende de Asteroide. Pois um método da classe Nave, recebe uma variável do tipo Asteroide. Mas não temos um membro na classe Nave do tipo Asteroide, por isso não temos uma agregação e/ou composição!



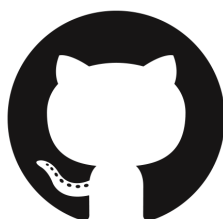
Exercício 1 – Space Shooter



Resolução dos Exercícios



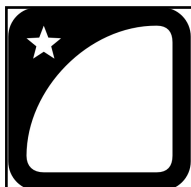
<https://github.com/phillima-inatel/C125>



C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

35


35



Material Complementar



 Capítulo 7 da apostila FJ-11 (pg 102)

 Pacotes - Organizando suas Classes e Bibliotecas

C125 – PROGRAMAÇÃO ORIENTADA A OBJETOS COM JAVA – INSTITUTO NACIONAL DE TELECOMUNICAÇÕES

36

36



C125 – Programação Orientada a Objetos com
Java

Pacotes

Prof. Phyllipe Lima
phyllipe@inatel.br

