Orientação a Objetos com Java: Pacotes, Números, Strings, Enumerações e Anotações

prof. Angelo C. Loula

Pacotes

- Pacotes são uma organização hierárquica de classes
  - similar a arquivos em diretórios
- Um pacote define um agrupamento de classes afins
  - Usado também para controle de acesso
- Classes com mesmo nome mas em pacotes diferentes são tratadas como classes diferentes
- Nome do pacote pode conter separações por ponto, definindo hierarquia.

Pacotes

```
//in the Draggable.java file
package graphics;
public interface Draggable {
    . . .
}

//in the Graphic.java file
package graphics;
public abstract class Graphic {
    . . .
}

//in the Circle.java file
package graphics;
public class Circle extends Graphic implements Draggable {
    . . .
}
```


Pacotes

```
//in the Draggable.java file
package br.uefs.ecomp.graphics;
public interface Draggable {
    . . .
}

//in the Graphic.java file
package br.uefs.ecomp.graphics;
public abstract class Graphic {
    . . .
}

//in the Circle.java file
package br.uefs.ecomp.graphics;
public class Circle extends Graphic implements Draggable {
    . . .
}
```

Pacotes

```
br.uefs.ecomp.graphics.Circle myshape;

import br.uefs.ecomp.graphics.Circle;
Circle myshape;

import br.uefs.ecomp.graphics.*;
Circle myshape;
```

- Pacotes e Diretórios

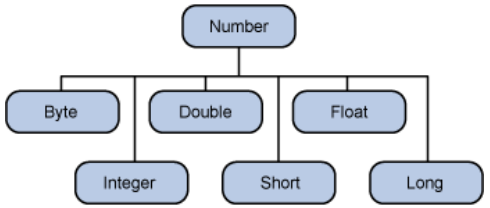
```
package graphics;
public class Rectangle()
```

`…\graphics\Rectagle.java`

- CLASSPATH

Números

- Classe *Number*
  - classe abstrata com subclasses de cada tipo de número
  - *wrappers* para os tipos primitivos



- O compilador por transformar entre tipos primitivos e objetos automaticamente

```
Integer x, y;
x = 12;
y = 15;
System.out.println(x+y);
```


Números

- Classe *Number*
  - Possibilidade de passar tipos primitivos como objetos para métodos que esperam objetos
  - Ter acesso a constantes das classes, como os valores limites MIN_VALUE e MAX_VALUE
  - Fazer conversões entre tipos

Caracter

```
char ch = 'a';
char uniChar = '\u039A'; // Unicode for uppercase Greek
  omega character
char[] charArray ={ 'a', 'b', 'c', 'd', 'e' }; // an
  array of chars
```

- *Wrapper* para caracter
  - Classe Character

```
Character ch = new Character('a');
```

  - Contém métodos de manipulação de caracteres

Strings

- Cadeias de Caracteres
  - Imutáveis

```
String greeting = "Hello world!";
```

  - 11 construtores, exemplo:

```
char[] helloArray = { 'h', 'e', 'l', 'l', 'o', '.'};
String helloString = new String(helloArray);

System.out.println(helloString);
```

  - Métodos: length(), getChars(), concat()

```
"My name is ".concat("Rumplestiltskin");
"Hello," + " world" + "!"
```

  - Conversão de número para string: subclasses de number valueOf(), parseFloat(), parseInt(), etc.

Strings

- Cadeias de Caracteres
  - Mais métodos
    - charAt()
    - substring()
    - split()
    - trim()
    - toLowerCase()
    - toUpperCase()
    - indexOf()
    - replace()
    - equals() comparação
- StringBuilder: cadeia de caracter mutável

Enumerações

- Conjunto de constantes

```
public enum Day {
    SUNDAY, MONDAY, TUESDAY, WEDNESDAY,
    THURSDAY, FRIDAY, SATURDAY
}
```

- Mas também é uma classe com métodos próprios
  - Todo enum estende java.lang.Enum

Enumerações

```
public enum Planet {
    MERCURY (3.303e+23, 2.4397e6),
    VENUS   (4.869e+24, 6.0518e6),
    EARTH   (5.976e+24, 6.37814e6),
    MARS    (6.421e+23, 3.3972e6);

    private final double mass;   // in kilograms
    private final double radius; // in meters
    Planet(double mass, double radius) {
        this.mass = mass;
        this.radius = radius;
    }
    private double mass()   { return mass; }
    private double radius() { return radius; }

    // universal gravitational constant  (m3 kg-1 s-2)
    public static final double G = 6.67300E-11;
```

Enumerações

```
    double surfaceGravity() {
        return G * mass / (radius * radius);
    }
    double surfaceWeight(double otherMass) {
        return otherMass * surfaceGravity();
    }
    public static void main(String[] args) {
      double earthWeight = Double.parseDouble(args[0]);
        double mass = earthWeight/EARTH.surfaceGravity();
        for (Planet p : Planet.values())
           System.out.printf("Your weight on %s is %f%n",
                             p, p.surfaceWeight(mass));
    }
}
```


Anotações

- Provêem informações sobre o código, mas não são instruções do código, mas podem auxiliar o compilador
- Novas anotações podem ser criadas

```
@Author(
   name = "Benjamin Franklin",
   date = "3/27/2003"
)
class MyClass() { }
```


Anotações

```
@SuppressWarnings(value = "unchecked")
void myMethod() { }
```

Se existe somente o campo value, seu nome pode ser omitido:

```
@SuppressWarnings("unchecked")
void myMethod() { }
```

Se não existem campos, os parânteses podem ser omitidos:

```
@Override
void mySuperMethod() { }
```


Anotações

- Anotações do compilador

```
@Deprecated
```

```
@Override
```

```
@SuppressWarnings
```

– seguido dos tipos de warnings