Enumerados
Módulos

Sumário

- Packages
- Acesso
- Enumerados
- … & stuff

Packages

- Packages = pastas = arrumação = poder ter 2 coisas com o mesmo nome em sítios diferentes

`package pt.iscte.poo.aula2; -> pt/iscte/poo/aula2`

```
class Car {…}
```

- Pode existir também

`package pt.iscte.ip.aula2 -> pt/iscte/ip/aula2`

```
class Car {…}
```

Packages

- zip pt/iscte (todos os programas feitos nas minhas aulas no ISCTE)
- zip pt/iscte/poo (todos os programas feitos nas minhas aulas de POO)
- Noutro programa qualquer … usar o Carro feito em POO:

```
import pt.iscte.poo.Car
```

Acesso, atributos

- **private**
- (package-private)
- protected
- public

Acesso, operações

- private
- (package-private)
- protected
- public

Classe = módulo

- Interface (public)
  - Operações e Constantes
- Implementação (private)
  - Operações e métodos privados
  - Atributos
  - Corpo dos métodos

Classe = módulo

- Contrato
  - Pré e pós-condições das operações e métodos
  - Verificação
- Manual de utilização
  - Comentários de documentação da classe (para programador)
  - Comentários de documentação de cada característica pública (para utilizador)

Enumerados

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

Weekday day = Weekday.MONDAY;

public enum Orientation {WEST, NORTH, EAST,
SOUTH}

Orientation o = Orientation.NORTH;
```

Enumerados

- Mensagem diferente por cada dia da semana …

```
WeekDay day = …
if (day == WeekDay.Monday) {
  System.out.println(“Weekend is over, get up!!!”);
} else if (day == WeekDay.Tuesday) {
  …
}  else …
```

Enumerados

```
WeekDay day = …
switch (day) {
   case WeekDay.Monday:
     System.out.println(…);
     break;
   case WeekDay.Tuesday:…
     break;
…
```

Enumerados

- Imprimir todos os dias da semana?

```
for (WeekDay day: Weekday.values()) {
  System.out.println(day);
}

WeekDay[] allDaysOfTheWeek = WeekDay.values()
```

Enumerados

- Quantos dias há entre dois dias da semana?
- Idealmente a chamada seria:

```
int n = nDaysBetween(Wednesday, Monday);
```

Como seria a declaração:

```
public static int nDaysBetween( ??? day1, ??? day2)
```

Enumerados

```
enum WeekDay {Monday, Tuesday, Wednesday, …

public static int nDaysBetween(WeekDay day, WeekDay day2) {
      return day2.ordinal() – day1.ordinal();
}
```

chamada …

```
int n = nDaysBetween(WeekDay.Monday, WeekDay.Wednesday);
```

Enumerados (funções de instância)

ou …

```
public int nDaysUntil(WeekDay day) {
    return day.ordinal() – ordinal();
}
int n =
WeekDay.Monday.nDaysUntil(WeekDay.Wednesday
);
```

Enumerados

- Que dia da semana é hoje?
- Ideal:

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

WeekDay day = keyboard.nextLine();
```

mas:

```
Type mismatch: cannot convert from String to Weekday
```

Enumerados

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


WeekDay day = Weekday.valueOf(keyboard.nextLine());


 java.lang.IllegalArgumentException: No enum constant
Weekday.ontem
```

… & stuff

- referências (revisões)
- Scanner (revisões?)
- toString (revisões?)
- ArrayList
- classes-pacote

Sumário

- Packages
- Acesso
- Enumerados
- … & stuff

Referências

- Y. Daniel Liang, "Introduction to Java Programming" 7th Ed. Prentice-Hall, 2010.