

Sesión 06: Modelización

Programación 2

Ángel Herranz

Febrero 2019

Universidad Politécnica de Madrid

En capítulos anteriores

- ...
 - Objetos, referencias y variables (y primitivos)
 - Clases: plantilla para crear objetos
- Encapsular
- datos y comportamiento
- Racionales

En capítulos anteriores

- ...
- Objetos, referencias y variables (y primitivos)
- Clases: plantilla para crear objetos

Encapsular

datos y comportamiento

- Racionales
- Terminología y ocultación



Terminología

atributo, clase, constructor, dato
básico, instancia, invocación, mensaje,
miembro, método, modificador,
objeto, observador, puntero,
referencia, tipo, variable



Ocultación

Para evitar que una modificación en la representación interna de una clase afecta a quien la usa, es muy importante ocultar dicha representación



Ocultación

Para evitar que una modificación en la representación interna de una clase afecta a quien la usa, es muy importante ocultar dicha representación

```
class Punto2D {  
    double x;  
    double y;  
    ...  
}  
  
public static void  
    main(String[] args) {  
    Punto2D p =  
        new Punto2D();  
    p.x = 1;  
    p.y = -1;  
}
```



Ocultación

Para evitar que una modificación en la representación interna de una clase afecta a quien la usa, es muy importante ocultar dicha representación

```
class Punto2D {  
    private double x;  
    private double y;  
    ...  
}
```

```
public static void  
    main(String[] args) {  
        Punto2D p =  
            new Punto2D();  
        p.x = 1; 👍 No compila  
        p.y = -1; 👍 No compila  
    }
```



Buenas prácticas i

En lugar de exponer los atributos se define un **API** de acceso a la clase

- `Punto2D()`: crea un punto $(0, 0)$ en coordenadas cartesianas
- `Punto2D(x,y)`: crea un punto (x, y) en coordenadas cartesianas (abscisa y ordenada)
- `x()`: devuelve la abscisa
- `y()`: devuelve la ordenada



Buenas prácticas ii

- `cambiarX(double x)`: modifica la abscisa
- `cambiarY(double y)`: modifica la ordenada
- `distancia(Punto2D b)`: devuelve la distancia desde **this** a b
- `rotar(double a)`: rota el punto alrededor del (0,0) la cantidad de a radianes



Buenas prácticas ii

- `cambiarX(double x)`: modifica la abscisa
- `cambiarY(double y)`: modifica la ordenada
- `distancia(Punto2D b)`: devuelve la distancia desde **this** a b
- `rotar(double a)`: rota el punto alrededor del (0,0) la cantidad de a radianes

¡Implementar la buena práctica!

The Cincinnati Kid

Texas hold'em

🎧 ¿Cómo modelizar una mano de *Texas hold'em*?

Texas hold'em

🎧 ¿Cómo modelizar una mano de *Texas hold'em*?

- Cada jugador recibe dos cartas
- Hay 5 cartas comunitarias
- Tres fases: *flop*, *turn* y *river*

Naipes de la baraja francesa




Modelizar los naipes ⌚ 15'

- Cada naipe tiene un palo: ♣ ♦ ♠ ♥
- Y un *valor*: A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K
- Ignoremos el comodín
- Pequeño programar que usa los naipes:
crear y mostrar naipes

Modelización de *enumeraciones*

```
public enum Palo {  
    TREBOLES, DIAMANTES, CORAZONES, PICAS;  
}
```

```
public enum Valor {  
    AS, DOS, TRES, CUATRO, CINCO,  
    SEIS, SIETE, OCHO, NUEVE, DIEZ,  
    VALET, DAMA, REY;  
}
```

 Explorar el método `ordinal()` y la
función `values()`