
 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 2</i>	

Presentació

Benvingut a Mokepon 2. En aquest exercici capturaràs Mokepon.

Herència

Pensem en que passa quan captures un Mokepon. Un Mokepon capturat segueix sent un Mokepon, però hauria de tenir informació nova, com el nom que li ha posat el seu entrenador, el nom de l'entrenador, i el nivell de felicitat (un Mokepon més feliç pega més). Podem crear una classe nova, però realment volem que segueixi sent un Mokepon, així que farem servir l'herència

Fem una nova classe MokeponCapturat, que serà **filla** de Mokepon.

```
//extends Mokepon implica que és filla de Mokepon, i hereda tots els mètodes i
atributs de Mokepon

public class MokeponCapturat extends Mokepon {

    //posem només els atributs nous, no els de Mokepon. És a dir, posem els atributs
que tindrà MokeponCapturat, pero no Mokepon



    String nomPosat;

    String nomEntrenador;

    int felicitat;

}
```

Podem provar que efectivament MokeponCapturat hereda tots els atributs i mètodes de Moképon. No només això, al mètode Mokepon.atacar(Mokepon atacat, int numAtac), que li has de dir el Mokepon al que ataques, pots posar el MokeponCapturat. Això es deu a que Java enten que MokeponCapturat, també és un

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 2</i>	

Moképon. Aquesta propietat a on una classe es comporta com si en fos dues simultàniament se'n diu **Polimòrfia**

```

MokeponCapturat elMeuMikachu = new MokeponCapturat();

System.out.println(elMeuMikachu.nom);

elMeuMikachu.diguesNom();

mulmasaur.atacar(elMeuMikachu, 0);

```

Constructors Heredats i super

Ara veiem que MokeponCapturat no té constructors més enllà de l'estandard. Imaginem que volem fer els constructors que tenia Mokepon. Per a ajudar-nos, podem fer servir super. Així com **this** fa referència a la nostra classe, **super** fa referència al nostre pare (en aquest cas, Mokepon)

```

public MokeponCapturat(String nom, Tipus tipus) {

    //cridem al constructor idèntic del pare

    super(nom, tipus);

    //la resta de variables les posem nosaltres

    this.nomPosat = nom;

    this.nomEntrenador = "Marc";



    this.felicitat = 50;

}

```

Fes la resta de constructors de Mokepon fent servir el super.

Fes a més, un nou constructor a on li passes (Mokepon mok, String nomPosat, String nomEntrenador) i creï un nou MokeponCapturat en base a les dades del Mokepon proporcionat (posa la felicitat a 50)

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 2</i>	

Felicitat, sobreescrivre mètodes

Comença fent un mètode acariciar a MokeponCapturat, que pujarà la seva felicitat 10 punts (fins a un màxim de 100). Fixa't que un Mokepon no podrà fer servir el mètode acariciar, ja que la herència només va de pares a fills.

Ara canvia el mètode atacar. Tot i que atacar ja existeix en Mokepon, pots canviar com funciona el mètode en MokeponCapturat tornant-lo a implementar. Implementa a MokeponCapturat un mètode atacar que sobreescriui el de Mokepon amb les següents característiques

si felicitat està a 50 o més, es multiplica el dany per 1.2. En cas contrari es multiplica per 0.8.

Implementa-ho escrivint el mètode sencer. Però no facis un nou mètode efectivitat, fes servir `super.efectivitat()` per a no tenir que tornar a escriure'l.

Capturar, instanceof

Fes a Mokepon un mètode capturar(`nomEntrenador`, `nomDonat`) que capturi aquell Mokepon i et torni un Mokepon capturat de les mateixes característiques. Fes-te valdre del constructor que has creat a MokeponCapturat per fer-ho.

No obstant, ens adonem que només volem capturar a aquells Mokepon que no estiguin capturats ja. Com podem saber si el Mokepon no és un MokeponCapturat?

Això es pot fer mitjançant `instanceof`. la comanda `instanceof` funciona de la següent forma

`nomObjecte instanceof nomDeClasse`



i et diu `true` si forma part de la classe i `false` si no

d'aquesta forma

```
Mokepon mikachu = new Mokepon();

System.out.println(mikachu instanceof Mokepon);
```

printaria `true`

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Mokepon - Part 2</i>	

Aquí tens una funció capturar a mig fer fent servir instanceof

```

MokeponCapturat capturar(String nomEntrenador, String nomDonat) {

    //si nosaltres mateixos no som de classe MokeponCapturat
    if(!(this instanceof MokeponCapturat)) {

        //capturar el Mokepon

    }else{

        System.out.println("No pots capturar un Mokepon que ja està capturat");

        //podem castejar perquè estem segurs de que és un MokeponCapturat gràcies
al instanceof

        return (MokeponCapturat) this;

    }

}

```

Acaba la funció capturar. Implementa també al main (és a dir, la classe Test) una funció static `MokeponCapturat capturar(Mokepon mok, String nomEntrenador, String nomDonat)` que faci el mateix, però amb el Mokepon donat en comptes del `this`.

Pensa en aquestes 3 coses

1. Volem retornar un `MokeponCapturat`
2. Tenim un constructor de `MokeponCapturat` que hem creat abans que li passes un `mokepon` i et torna un `MokeponCapturat`
3. La paraula per referir-nos a nosaltres mateixos és **this**