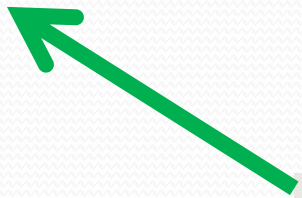
Podstawy i języki programowania

Dziedziczenie w praktyce
Redefiniowanie metod
Public, private, protected: podstawy enkapsulacji
Zmienne statyczne
Serializacja

Magiczne słowo „extends”

```
12  public class Zwierze {
13      int wiek;
14      double waga;
15      String imie;
16
17  }
18
```



```
12  public class Pies extends Zwierze{
13      /*
14       * Dostęp do wszystkiego
15       * Co było w klasie "Zwierze"
16       * oraz:
17       */
18      void szczekaj() {
19          System.out.println("Hau! Hau!");
20      }
21  }
22
```

Operacje I/O w Javie

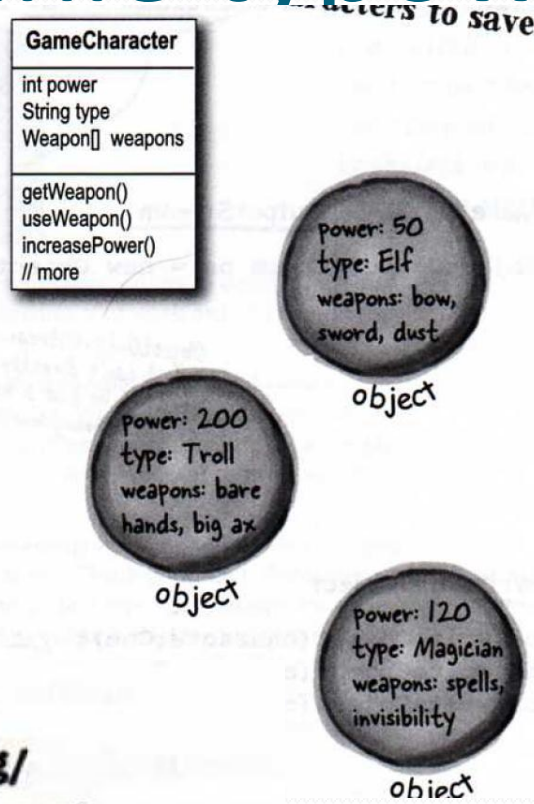
Serializacja

- Zapisuje całą klasę
- Plik „binarny”
- Delimiter nieokreślony
- Nie da się podglądać☺

Pliki tekstowe

- Zapisuje wybrane informacje
- Plik tekstowy
- Delimiter ustawiamy sami
- Można podejrzeć

Porównanie typów plików



Serializacja

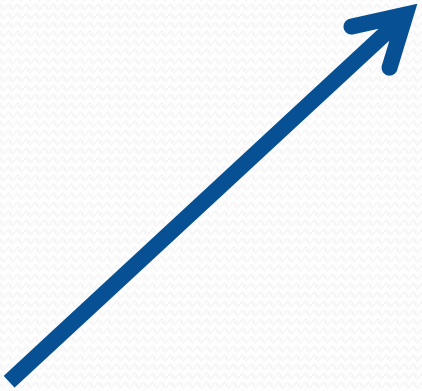
```
¬ìsrGameCharacter
¬%gê8MÛÎpowerLjava/lang/
String;[weaponst[Ljava/lang/
String;xp2tlfur[Ljava.lang.String;≠“VÁ
È{Gxptbowtswordtdustsq~»tTrolluq~tb
are handstbig axsq~xtMagicianuq~tspe
llstinvisibility
```

Plain tekst

```
50,Elf,bow, sword,dust
200,Troll,bare hands,big ax
120,Magician,spells,invisibility
```

Serializacja

Wymagane do serializacji obiektów!



```
3   public class Film implements Serializable {
4           String tytuł;
5           String gatunek;
6           int ocena;
7           static int liczba_filmow=0;
8
```

Enkapsulacja

- Ukrywanie zmiennych wewnętrznych.

```
Film film = new Film();
film.tytuł = "Jakiś tytuł";
```

```
97      Film film = new Film();
98      film.setTytuł("Jakiś tytuł");
99      }
```

```
12  String getTytuł() {
13      return tytuł;
14  }
15
16  void setTytuł(String tyt) {
17      this.tytuł=tyt;
18  }
```

Po co?

1. Możliwość kontroli błędów przed ustawieniem
2. Użytkownik nie widzi wewnętrznej reprezentacji pól.

Poziomy dostępu

- **public** – dostęp do atrybutu z każdego miejsca
- **Private** – dostęp do atrybutu tylko z wnętrza klasy bazowej/obiektu bazowego
- **Protected** – dostęp do atrybutu tylko z wnętrza klasy bazowej lub niższej w hierarchii dziedziczenia

Redefiniowanie metod

- W Javie wszystko co napiszemy dziedziczy po klasie Object. I tam jest taka fajna metoda toString(). Ale nie działa ona tak jak chcemy... No to nic trudnego!

Java dba, abyśmy się nie pomylili...

```
   @Override
13 public String toString() {
14 return "Tytuł: " + tytuł + "\nGatunek: " + gatunek + "\nOcena: " + ocena + "\n";
15     }
16
17 }
```

Filmoteka

```
3   public class Film implements Serializable {
4           String tytuł;
5           String gatunek;
6           int ocena;
7           static int liczba_filmow=0;
8
```

Filmoteka

```
3   public class Film implements Serializable {
4           String tytuł;
5           String gatunek;
6           int ocena;
7           static int liczba_filmow=0;
8
```

No to co to ten **static?**

Dodajemy film

```
24      static void utworz() {
25          Film nowy = new Film();
26          Scanner sc = new Scanner(System.in);
27          System.out.println("Podaj tytuł");
28          nowy.tytuł = sc.nextLine();
29          System.out.println("Podaj gatunek");
30          nowy.gatunek = sc.nextLine();
31          System.out.println("Podaj ocenę");
32          nowy.ocena = sc.nextInt();
33          kartoteka[Film.liczba_filmow++] = nowy;
34
35      }
```

Zapisujemy obiekt(y)

```
36      static void zapisz() {
37          try {
38          File plik_zapis = new File("dane.ser");
39          FileOutputStream strumien_wyjsciowy = new FileOutputStream(plik_zapis);
40          ObjectOutputStream strumien = new ObjectOutputStream(strumien_wyjsciowy);
41          for(int i=0; i<Film.liczba_filmow;i++) {
42              strumien.writeObject(kartoteka[i]);
43          }
44          strumien.close();
45          }
46          catch(Exception ex) {
47          System.out.println("Oups! Coś złego się stało");
            ex.printStackTrace();
49          }
50      }
```

… i je odczytujemy

```
51      static void odczytaj() {
52          try {
53          File plik_odczyt = new File("dane.ser");
54          FileInputStream strumien_wejsciowy = new FileInputStream(plik_odczyt);
55          ObjectInputStream strumien = new ObjectInputStream(strumien_wejsciowy);
56          Object obiekt = null;
57          Film.liczba_filmow = 0;
58           for(int i=0;i<10;i++)
59           {
60             obiekt = strumien.readObject();
61              kartoteka[Film.liczba_filmow++] = (Film) obiekt;
62          }
63          strumien.close();
64          }
65          catch (EOFException fp) {
66              System.out.println("Koniec pliku");
67          }
68          catch(Exception ex) {
69          System.out.println("Oups! Coś złego się stało");
            ex.printStackTrace();
71          }
72      }
```

Zadanie

- Napisz kompletny program do katalogowania kolekcji broni/znaczków pocztowych/kapsli od piwa
- Potrzebne funkcjonalności:
  - Klasa do opisu obiektu (typ, nazwa, cena, itp.)
  - Menu główne z możliwością wyświetlania, dodawania, usuwania, edytowania elementów kolekcji
  - Wyświetlanie i sortowanie po wybranych kryteriach.
  - Masowa zmiana ceny elementów z kolekcji (np. dodanie do wszystkich wartości 4% z powodu inflacji)
  - Zapis i odczyt z pliku obiektowego
  - Metoda toString() wyświetlająca ładnie sformatowany opis danego obiektu
  - Zapis całości do pliku tekstowego rozdzielanego średnikami