rozprávkové recyklovanie

@RoboNovotny

UINF/PAZ1c

epizóda 5

14/okt/15

plán na dnes

- rozprávkové algoritmy
- znovupoužiteľnosť kódu
- výstavba tried

kde sme?

návrh triedy začína schopnosťami

ČO pred AKO


ako navrhovať ako?

dobrý kód sa číta ako rozprávka


dobrý kód je rozšíriteľný
& znovupoužiteľný

nová funkcionalita

- pridávanie nových tried
- znovupoužitie existujúcich tried
- pridávanie metód
- prekrývanie zdedených metód

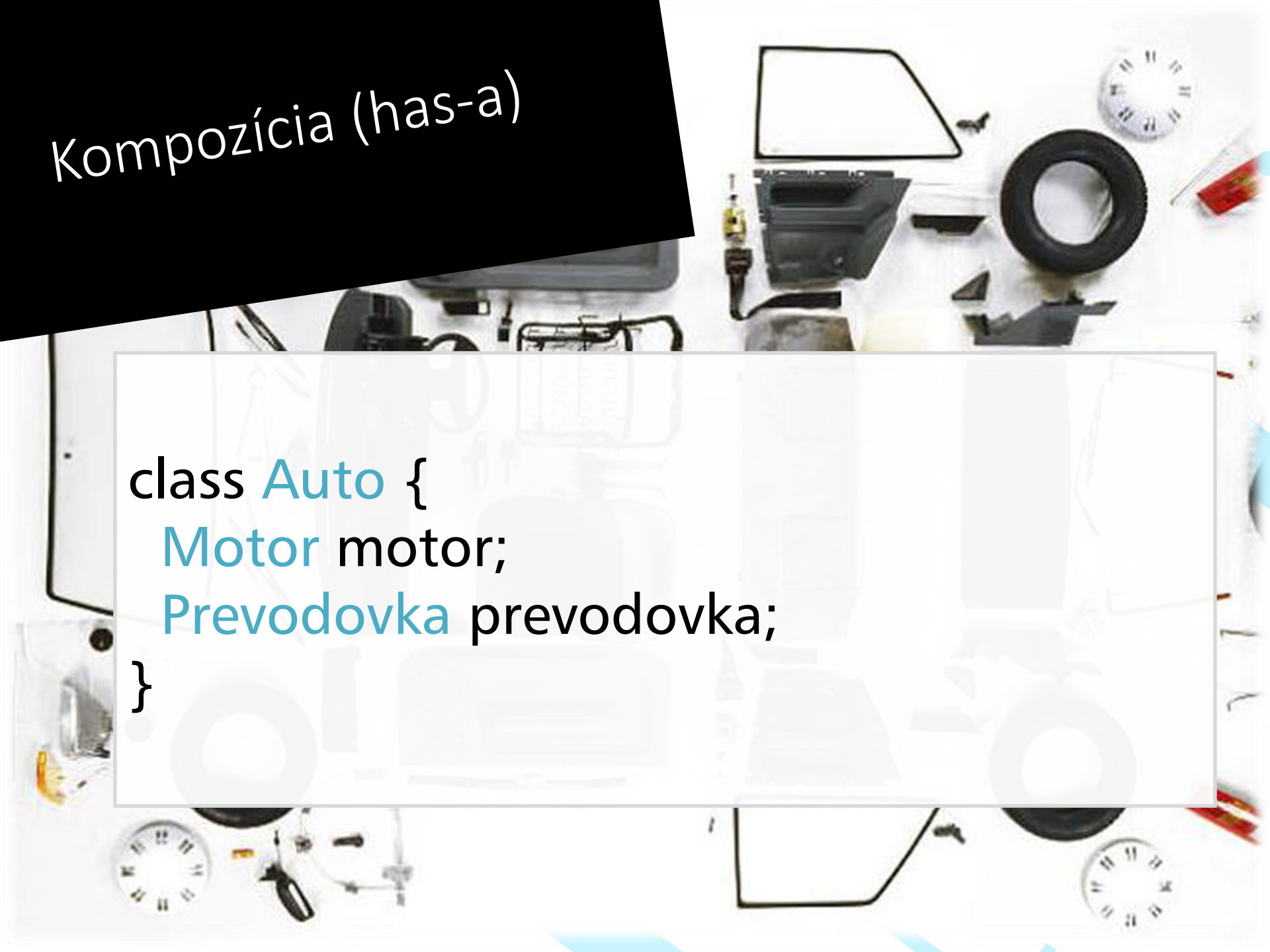
2 | 3 spôsoby výstavby tried

kompozícia

dedičnosť

delegácia

Kompozícia (has-a)



```
class Auto {
  Motor motor;
  Prevodovka prevodovka;
}
```


kompozícia je o stave a
kolaborátoroch

vízia rozšíriteľnosti

- aké správanie budú programátori modifikovať?
- kde priestor na dodanie funkcionality?

vízia rozšíriteľnosti

skúsenosti + predvídavosť + šťastie

demo bogosort


krátke metódy
s jedinou
zodpovednosťou

bogosort

```
public void bogosort(List<String> data) {
        while(!jeUtriedeny(data) {
                zamiesaj(data)
        }
}

private boolean jeUtriedeny(List<String> data)
{
}

private void zamiesaj(List<String> data) {
        Collection.sort(data);
}
```

bogosort

```
public class BogoSort {
        public void sort(List<String> d);
        private boolean jeUtriedeny(List<String> d);
        protected void zamiesaj(List<String> d) {
}
```

miesto rozšírenia: možné chytrejšia miešania

bogosort

```
public class MegaBogoSort extends BogoSort {

  protected void zamiesaj(List<String> d) {

  }

}
```

rozšíriteľnosť cez dedičnosť & prekrývanie

Open-Closed Principle

Triedy majú byť otvorené voči rozširovaniu, ale uzavreté voči zmenám.

Bertrand Meyer – Object Oriented Software Construction (1988)

Open for Extension

- možno rozširovať funkcionalitu
- pridávať nové správanie či stav

Closed for Modification

- neslobodno meniť zdrojový kód
- okrem opravy chýb

KISS

[Keep it simple, stupid]

Urobte tú najjednoduchšiu vec, ktorá vyzerá, že bude fungovať

Otázky?