Model podataka

1. Uvod

Do sada smo podatke u Angular aplikaciji čuvali u promenljivama u komponenti. Međutim, kompleksnije strukture podataka zahtevaju bolje strukturiranje. Kada bi trebalo predstaviti podatke iz realnog sveta, oslanjanje na promenljive bi bilo veoma nepraktično zbog kompleksne strukture tih podataka (npr. podaci studentske službe). Angular nam omogućava da podatke strukturiramo i modelujemo putem klasa.

2. Model podataka

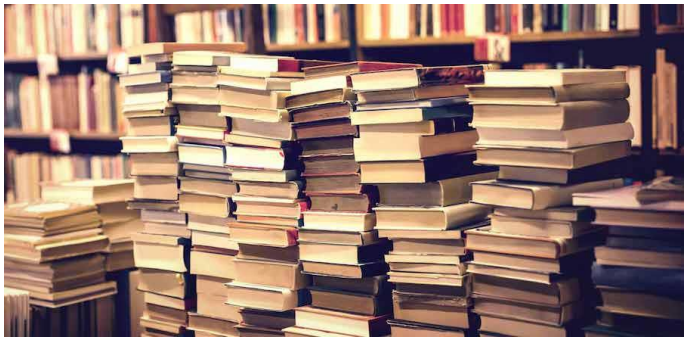
Model podataka se koristi kako bi se u aplikaciji efikasno organizovali podaci nad kojima se vrši obrada. Kada su svi relevantni podaci predstavljeni modelom, razvoj aplikacije je znatno lakši jer postoji jasna struktura nad kojom će se vršiti prikaz, pretraživanje, unos itd.

Model podataka na back-end i front-end delu se razlikuje:

- Front-end: podaci se modeluju tako da omoguće predstavljanje relevantnih podataka
- Back-end: podaci se modeluju kako bi se mogli trajno čuvati

! Model podataka u front-end segmentu **ne mora biti identičan** modelu podataka u back-end segmentu.

Entiteti (podaci) se u Angularu opisuju pomoću klasa i njihovih polja (svojstava).
Pojam klase ukazuje na klasifikaciju nekih primeraka, tako da za njihovo opisivanje koristimo iste skupove svojstava. Imajući ovo u vidu, model predstavlja apstrakciju podataka koji će biti korišćeni u aplikaciji.



```
export class Book {
  constructor(
    public title: string,
    public author: string,
    public published: number
  ) {}
}
```

Primer: Angular aplikacija omogućava pregled ponude neke knjižare. U modelu možemo definisati klasu *Book* sa poljima *title, author* i *published.*

2.1 Kreiranje modela

Klasa se definiše u zasebnom .ts fajlu. Naziv klase pišemo velikim početnim slovom, a naziv fajla malim početnim slovom.

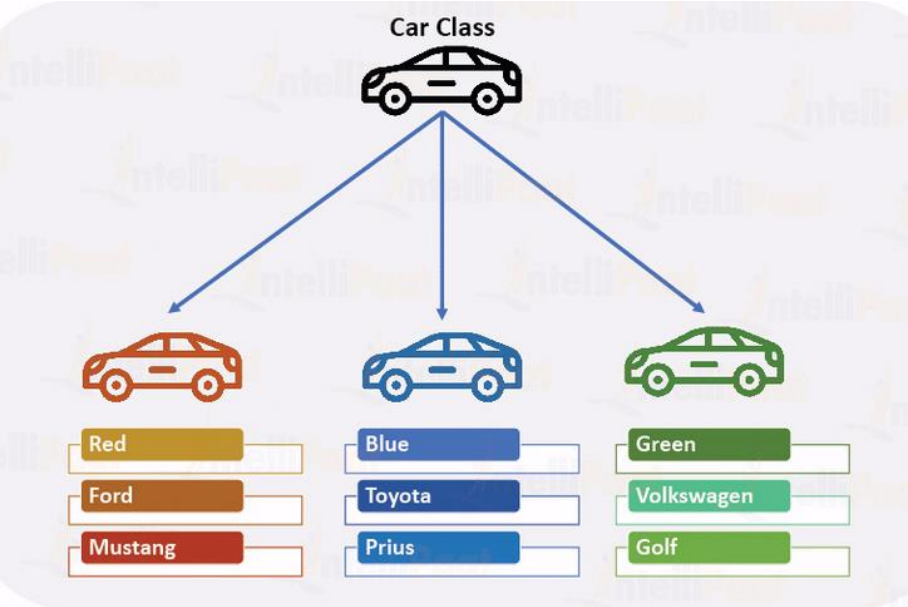
Praktičan način definisanja polja je ako ih navedemo kao parametre konstruktora (kao u primeru iznad, klasa Book) čime se olakšava instanciranje pojedinačnih objekata te klase. Pored naziva parametra,

definiše se i njegov tip. Modifikator **public** omogućava pristup ovim poljima i izvan same klase.

> ! Redosled navođenja parametara konstruktora je **bitan**, jer će se u tom redosledu navoditi i vrednosti tih parametara pri instanciranju.

2.2 Instanciranje

Instanciranje predstavlja kreiranje jednog primerka neke klase. Primerci klase se nazivaju instance tj. objekti.




```
export class Car {
  constructor(
    public color: string,
    public company: string,
    public model: string
  ) {}
}
```

Slikoviti prikaz klase *Car* i njenih instanci.

Da bi u nekoj komponenti koristili klasu *Car*, potrebno je importovati tu klasu.

```
import { Car } from './car';
```

Instanciranje se vrši navođenjem ključne reči new, nazivom klase i vrednostima parametara u istom redosledu kao u konstruktoru klase.

```
car1: Car = new Car('Red', 'Ford', 'Mustang');
```

Rezultat instanciranja je objekat koji čuvamo u nekoj promenljivoj (car1), a tip te promenljive je klasa koju smo instancirali.

Zadaci

Alat koji će se koristiti za pravljenje Angular aplikacija na vežbama je https://stackblitz.com/

1. Film je opisan nazivom, žanrom, trajanjem i linkom do stranice na IMDB. Instancirati po jedan objekat za omiljenu komediju, akciju i animirani film, pa omogućiti prikaz ovih podataka.

Domaći

1. Opisati osobu imenom, prezimenom, datumom rođenja i mestom rođenja. Instancirati tri objekta koji predstavljaju dete, majku i oca, pa putem šablona prikazati izvod iz matične knjige rođenih za dete.

Literatura

1. https://angular.io/
2. https://material.angular.io/
3. Materijali sa predavanja