

PGR208 Android Programming

Data class, List<T>

Rolando Gonzalez, 2024

Innhold

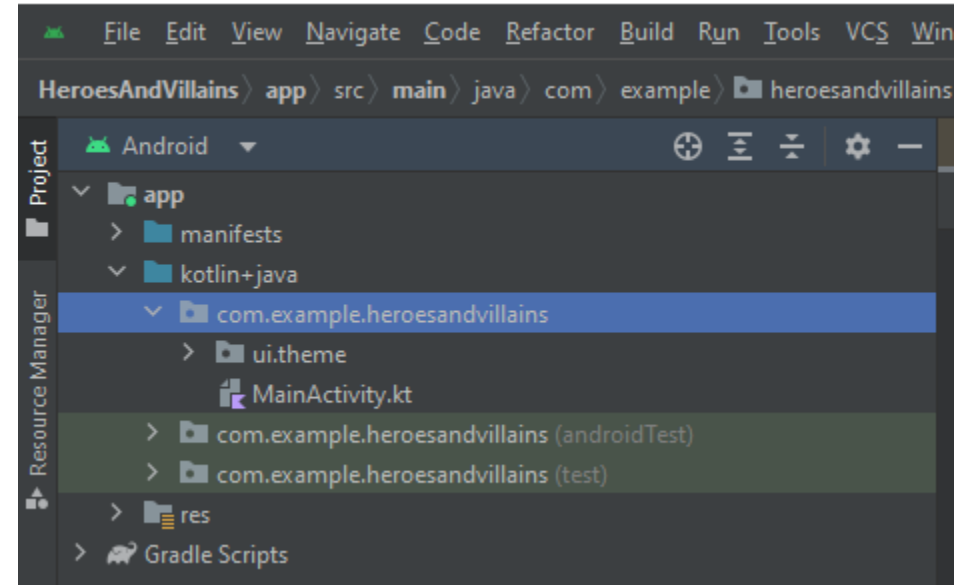
- Hva er en data class?
- Opprette mappe for data class
- Opprette data class-fil
- Initiere data class-objekt
- Liste med data class-objekter

Hva er en data class?

- En data class er en type klasse som har som hovedhensikt å holde på data.
- «Å holde på data» er spesielt relevant i sammenheng med at man bruker data class når man jobber med data fra en database eller når man henter data fra et Web API.
- En data class får ideelt sin egen fil; dvs. en fil med .kt-endelse
- En data class har innebygget «oppførsel» som gjør at de koder som man bruker i sammenheng med database og asynkront hentet informasjon lettere kan lages.

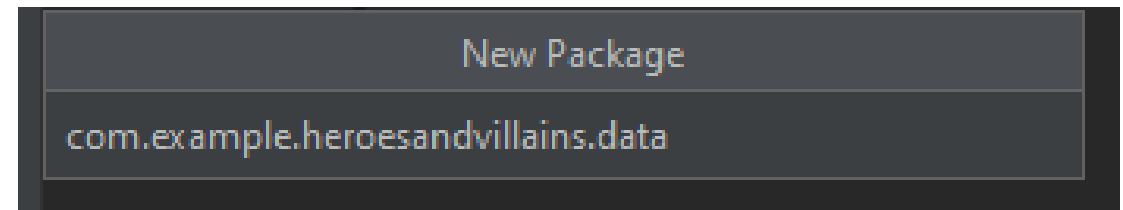
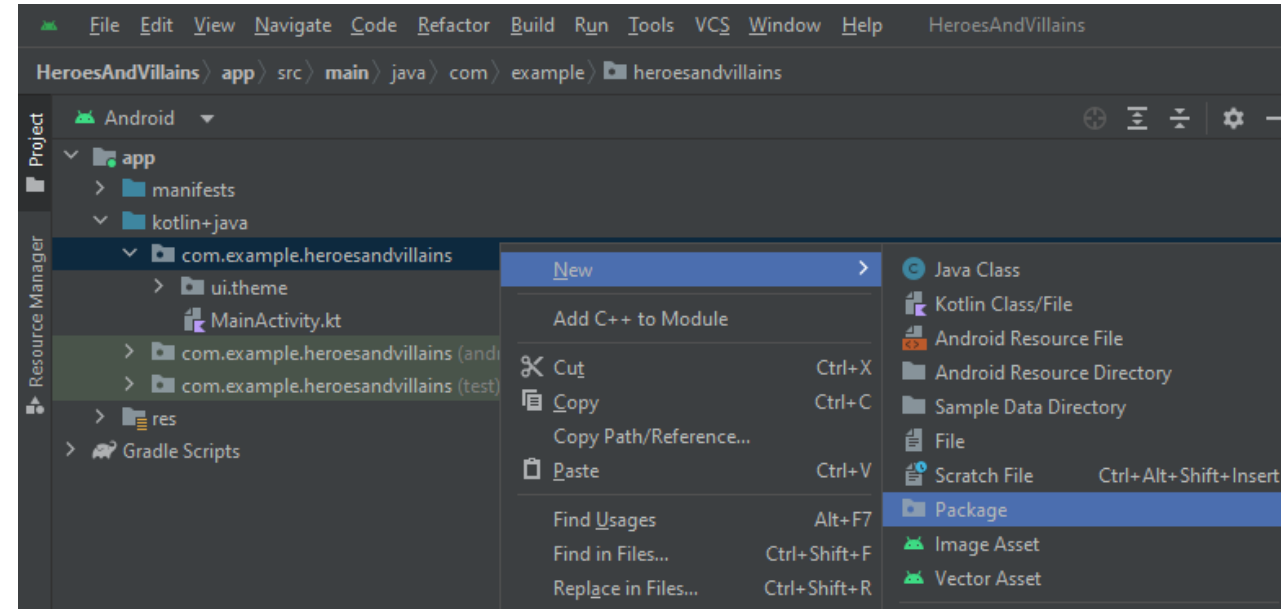
Konteksten for en data class

- Først har man en kontekst for data-klassen(e).
- Hvis man for eksempel har en app som har med superhelter og superskurker å gjøre så vil man i hvert fall jobbe med ting som har med de to tingene.
- Dataene om superhelter og superskurker vil enten komme fra en database lokalt på mobilen eller hentes fra ekstern datakilde som eksempelvis Web API.



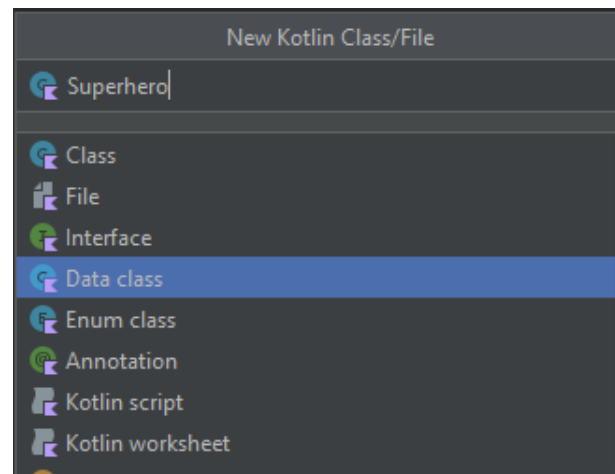
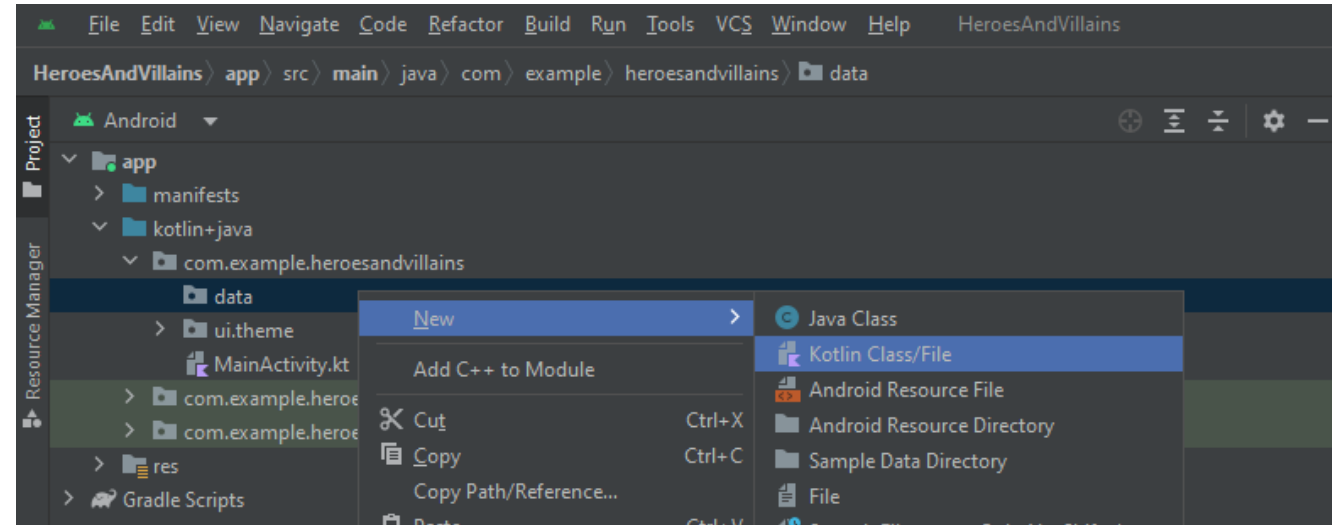
Opprette mappe for data classes

- Man oppretter en package ved å høyreklikke på hovedmappen for kodefilene og New -> Package.
- «data» er et typisk navn på hovedmappen for blant annet data classes.



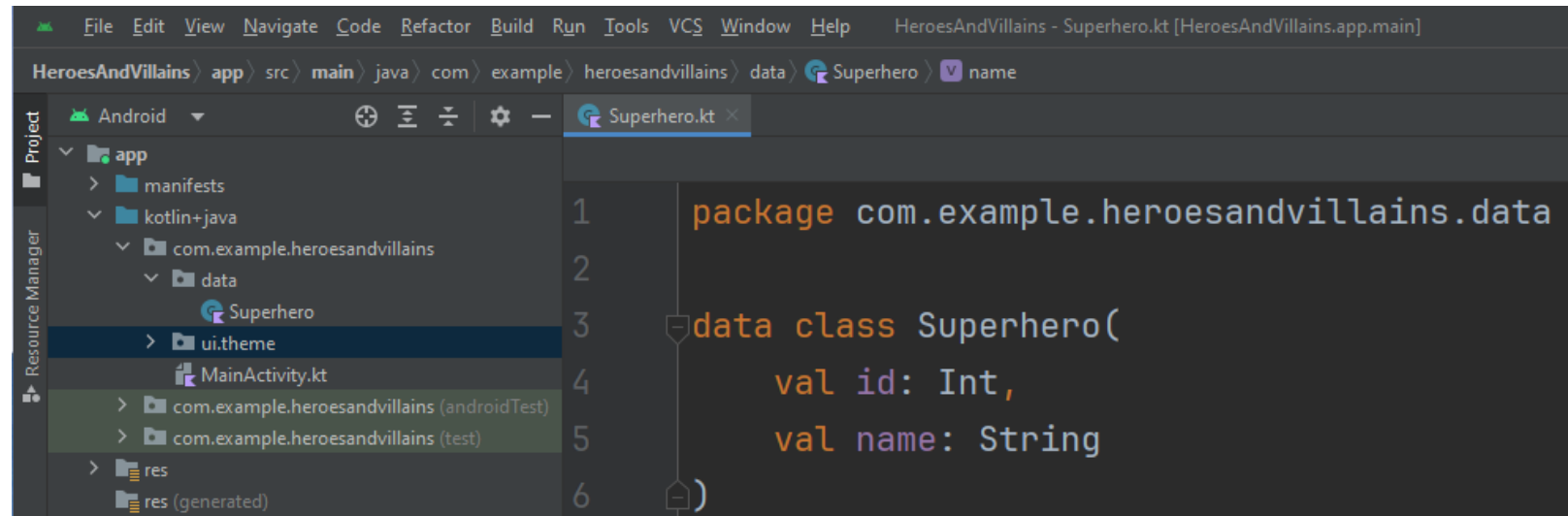
Opprette data class-fil

- Man høyreklikker så på data package og New -> Kotlin Class/File
- Deretter får man valgt Data class og satt navn



Kode data class

- En data class kan lages som vist på skjermbildet som lar det være mulig å deklarere og initiere objekter med property-setting direkte.
- Både val og var kan brukes på properties.



The screenshot shows an IDE window with the following structure:

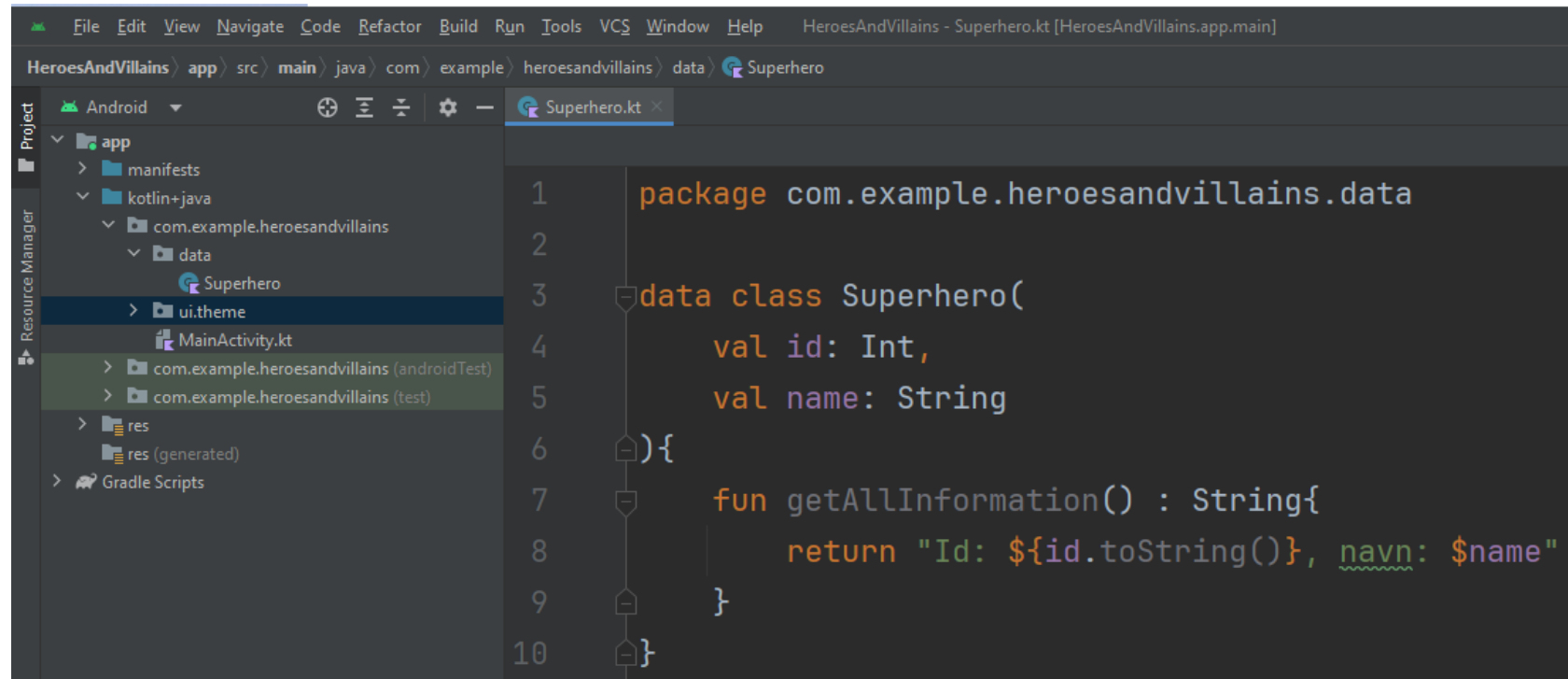
- Project: HeroesAndVillains
- Package: app > src > main > java > com > example > heroesandvillains > data
- File: Superhero.kt

The code in Superhero.kt is as follows:

```
1 package com.example.heroesandvillains.data
2
3 data class Superhero(
4     val id: Int,
5     val name: String
6 )
```

Kode data class - metoder

- Du kan også gi klassen metoder med *fun*.



The screenshot shows the Android Studio IDE with a project named 'HeroesAndVillains'. The left sidebar displays the Project and Resource Manager. The main editor window shows the file 'Superhero.kt' in the package 'com.example.heroesandvillains.data'. The code defines a data class 'Superhero' with two properties: 'id' of type 'Int' and 'name' of type 'String'. A method 'getAllInformation()' is defined, which returns a string formatted as 'Id: {id.toString()}, navn: {name}'.

```
1 package com.example.heroesandvillains.data
2
3 data class Superhero(
4     val id: Int,
5     val name: String
6 ){
7     fun getAllInformation() : String{
8         return "Id: ${id.toString()}, navn: $name"
9     }
10 }
```


Opprette enkeltobjekt av Superhero

- Koden viser hvordan vi i en composable, FavoriteSuperhero, initierer et objekt av Superhero.
- Når man initierer objekt kan man velge å inkludere property-navn eller ikke:
 - Superhero(id = 1)
 - Superhero(1)

```
@Composable
fun FavoriteSuperhero() {

    val superhero = Superhero(
        id = 1,
        name = "Superman"
    )

    Text(
        text = "My favorite superhero: ${superhero.name}!"
    )
}
```

```
val superhero = Superhero(
    id: 1,
    name: "Superman"
)
```

Opprette List<T> av Superhero

- Man kan lage lister med objekter av en data class.
- Man vil eksempelvis bruke en `forEach` for å gå gjennom arrayen og skrive ut til en `Column()`

```
val superheroes: List<Superhero> = listOf(  
    Superhero( id: 1, name: "Supermann"),  
    Superhero( id: 2, name: "Batman"),  
    Superhero( id: 3, name: "Spiderman"),  
    Superhero( id: 4, name: "Aquaman")  
)
```

```
Column {  
    superheroes.forEach{ superhero ->  
        Text(  
            text: "Navn: ${superhero.name}"  
        )  
    }  
}
```

Sette default-verdi

- Man kan sette en default-verdi på en property som vist på skjermbildet.
- Det betyr også at «level» ikke må settes ved initiering av objektet.
- Merk at «val level» leder til at level vil måtte forbli 0 hvis ikke en annen verdi settes. «var level» vil sette verdien til 0 og tillatte oppdateringer senere.

```
data class Superhero(  
    val id: Int,  
    val name: String,  
    var level: Int = 0  
)
```

Destructuring

- Destructuring går ut på at man kan opprette variabler som inneholder verdiene til properties i et Data class-objekt.
- Kan være praktisk for å eksempelvis slippe å måtte skrive slik flere ganger:
 - objektNavn.property

```
val (id, name, level) = superhero  
Text(name)
```

```
data class Superhero(  
    val id: Int,  
    val name: String,  
    var level: Int = 0  
)
```

sortedBy

- Man kan sortere en liste med objekter ved hjelp av sortedBy
- I kodeeksempelet blir listen sortert etter superheltens navn

```
val superheroes: List<Superhero> = listOf(  
    Superhero( id: 1, name: "Supermann"),  
    Superhero( id: 2, name: "Batman"),  
    Superhero( id: 3, name: "Spiderman"),  
    Superhero( id: 4, name: "Aquaman")  
).sortedBy { it.name }
```

Ressurser

- Om Data class
 - <https://kotlinlang.org/docs/data-classes.html>