EDAA45 Programmering, grundkurs

Läsvecka 6: Klasser

Björn Regnell

Datavetenskap, LTH

Lp1-2, HT 2016

6 Klasser

Vad är en klass?

- En mall för att skapa objekt.
- Objekt skapade med **`new`** `Klassnamn` kallas för **instanser** av klassen `Klassnamn`.
- En klass innehåller medlemmar (eng. *members*):
  - **attribut**, kallas även fält (eng. *field*): **`val`**, **`lazy val`**, **`var`**
  - **metoder**, kallas även operationer: **`def`**
- Varje instans har sin uppsättning värden på attributen (fälten).

Designexempel: Klassen Complex

TODO:

- Bygg upp `case class Complex(re: Double, im: Double)` steg för steg inspirerat av Pins3ed kap 6 i likhet med hur de gör med Rational
- Illustrera följande begrepp: this (behövs i max(that)), method overloading behövs för att plussa med både Complex och Double
- Till fördjupningsövning: dekorera Double med metoderna im och re samt (Double, Double) med metoden ir (för irrational) med implicit klass
- Till extrauppgift: implementera klassen Polar(r, fi) med polära koordinater https://sv.wikipedia.org/wiki/Pol%C3%A4ra_koordinater

Specifikationer av klasser i Scala

- Specifikationer av klasser innehåller information som *den som ska implementera* klassen behöver veta.
- Specifikationer innehåller liknande information som dokumentationen av klassen (scaladoc), som beskriver vad *användaren* av klassen behöver veta.

Specification **Person**

```
/** Encapsulate immutable data about a Person: name and age. */
case class Person(name: String, age: Int = 0){
  /** Tests whether this Person is more than 17 years old. */
  def isAdult: Boolean = ???
}
```

- Specifikationer av Scala-klasser utgör i denna kurs ofullständig kod som kan kompileras utan fel.
- Saknade implementationer markeras med ???
- Kommentarer utgör krav på implementationen.

Specifikationer av klasser och objekt

Specification **MutablePerson**

```
/** Encapsulates mutable data about a person. */
class MutablePerson(initName: String, initAge: Int){
  /** The name of the person. */
  def getName: String = ???

  /** Update the name of the Person */
  def setName(name: String): Unit = ???

  /** The age of this person. */
  def getAge: Int = ???

  /** Update the age of this Person */
  def setAge(age: Int): Unit = ???

  /** Tests whether this Person is more than 17 years old. */
  def isAdult: Boolean = ???

  /** A string representation of this Person, e.g.: Person(Robin, 25) */
  override def toString: String = ???
}
object MutablePerson {
  /** Creates a new MutablePerson with default age. */
  def apply(name: String): MutablePerson = ???
}
```

Specifikationer av Java-klasser

- Specificerar signaturer för konstruktorer och metoder.
- Kommentarerna utgör krav på implementationen.
- Används flitigt på extentor i EDA016, EDA011, EDA017...
- Javaklass-specifikationerna behöver kompletteras med metodkroppar och klassrubriker innan de kan kompileras.

`class Person`

```
/** Skapar en person med namnet name och åldern age. */
Person(String name, int age);

/** Ger en sträng med denna persons namn. */
String getName();

/** Ändrar denna persons ålder. */
void setAge(int age);

/** Anger åldersgränsen för när man blir myndig. */
static int adultLimit = 18;
```