# interface

### Voorbeeld

### Te gebruiken bij:

-

### Eigenschappen:

-

### Info

#### Overzicht

Een speciale vorm van abstracte classes zijn **interfaces**. Een interface heeft geen properties of geïmplementeerde methodes. Het enige dat erin staat, zijn lege methodes die ingevuld moeten worden door de class die de interface implementeert

Zo zouden we een interface kunnen maken die ervoor zorgt dat alle CreditCard-classes verplicht zijn om de methode isExpired te implementeren. Hiermee geef je als het ware een garantie aan de gebruiker (en ontwikkelaar) dat er altijd een implementatie van isExpired beschikbaar is.

java

interface Expiring {

boolean isExpired(int month, int year);

}

class CreditCard implements Expiring {

public boolean isExpired(int month, int year) {

// implementatie van deze methode...

}

}

Hierbij is het belangrijk om te weten dat wanneer je een interface in een abstracte class implementeert, dat de abstracte class niet verplicht is om deze methode te implementeren. De verantwoordelijkheid ligt dan bij de subclass van CreditCard:

abstract class CreditCard implements Expiring {

// geen implementatie van de isExpired-methode

}

Een goed voorbeeld van een interface die bepaalde functionaliteit belooft én afdwingt in de implementerende classes, is List in de Java standaard library: [List](https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/util/List.html).

#### aantekening

In dit voorbeeld definieert de interface "Afdrukbaar" de methode "afdrukken". De klassen "Boek" en "Document" implementeren de interface en bieden hun eigen implementatie van de "afdrukken" methode. In de "Hoofd" klasse worden objecten van zowel "Boek" als "Document" gemaakt en kan de "afdrukken" methode op elk van hen worden opgeroepen. Dit stelt je in staat om verschillende objecten met verschillende implementaties te verzamelen en ze op uniforme wijze aan te roepen door de interface te gebruiken.

java

interface Printable {

void print();

}

class Book implements Printable {

private String title;

public Book(String title) {

this.title = title;

}

@Override

public void print() {

System.out.println("Book: " + title);

}

}

class Document implements Printable {

private String name;

public Document(String name) {

this.name = name;

}

@Override

public void print() {

System.out.println("Document: " + name);

}

}

public class Main {