# Naslagmateriaal
# (beschikbaar bij toets)

### Notatie van klasse

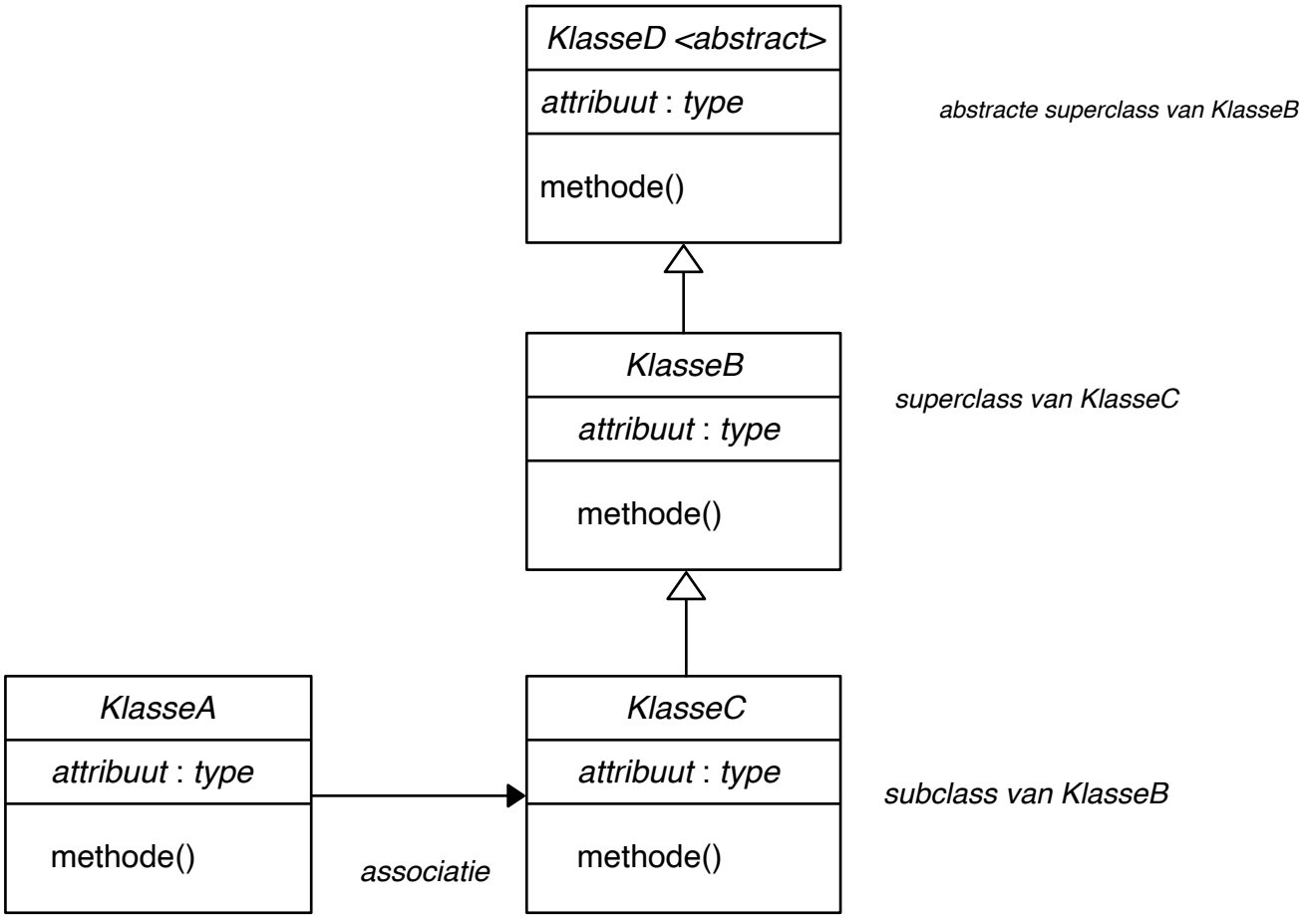
| *KlasseNaam* |
| --- |
| *attribuut_1* : *type*<br>*attribuut_2* : *type*<br>*-ditAttribuutIsPrivate : type*<br>*+ditAttribuutIsPublic : type* |
| methode()<br>methode(parameter_1, parameter_2)<br>methode() : *teruggeefwaarde*<br>setPrivateAttribuut(parameter : type)<br>getPrivateAttribuut() : type |

Afhankelijk van context kun je kiezen of je informatie over toegankelijkheid en types weglaat. Bij de toets wordt dit expliciet aangegeven

### Notatie van object / instantie

| *objectnaam : KlasseNaam* |
| --- |
| *attribuut_1* : *waarde*<br>*attribuut_2* : *waarde* |

### Ontwerpklassendiagram

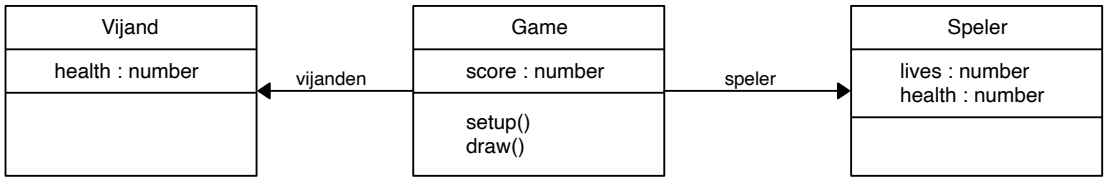


**Naslagmateriaal**
**(beschikbaar bij toets)**

#### Een class in JavaScript

```
class <Naam van de class> {
  #privateAttribuut;
  publicAttribuut;

  constructor(parameter1, parameter2) {
    this.#privateAttribuut = parameter1;
    this.publicAttribuut = parameter2;
  }

  methodenaam() {
    // code die uitgevoerd moet worden

    return <waarde>; // alleen als er een waarde teruggegeven moet worden
  }
}
```

#### Associatie in JavaScript

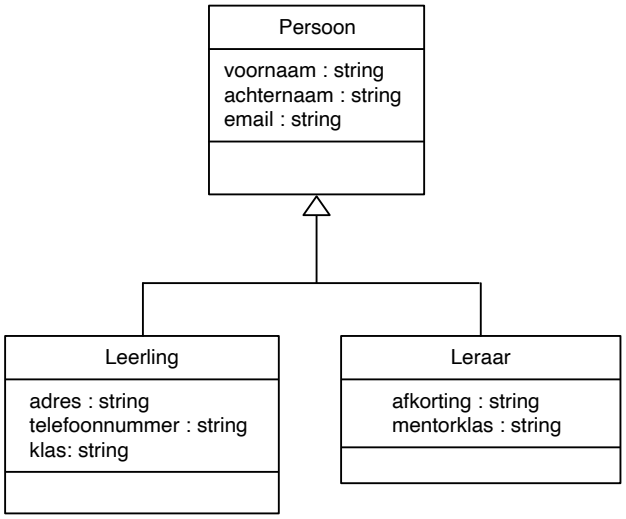



#### Mogelijke implementatie in JavaScript van de klasse Game

```
class Game {
  score;
  vijanden;
  speler;

  constructor() {
    this.score = 0;        // een nieuwe Game begint altijd met 0
    this.speler = new Speler();
    this.vijanden = [ new Vijand(), new Vijand(), new Vijand(), new Vijand() ];
  }

  setup() {
    // code die voorafgaand aan de game moet worden uitgevoerd
  }

  draw() {
    // code die voor ieder frame wordt uitgevoerd
  }

}
```

**Naslagmateriaal**
**(beschikbaar bij toets)**

**Overerving in JavaScript**




```
class Persoon {
  voornaam;
  achternaam;
  email;

  constructor(_voornaam, _achternaam, _email) {
    this.voornaam = _voornaam;
    this.achternaam = _achternaam;
    this.email = _email
  }

}
```

```
class Leerling extends Persoon {
  adres;
  telefoonnummer;
  klas;

  constructor(_voornaam, _achternaam, _email,
              _adres, _telefoonnummer, _klas) {

    super(_voornaam, _achternaam, _email);
    this.adres = _adres;
    this.telefoonnummer = _telefoonnummer;
    this.klas = _klas
  }

}
```

**Naslagmateriaal**
**(beschikbaar bij toets)**

#### Een class in C++ (versimpeld)

```
class <Naam van de class> {
  private:
    <type> naamPrivateAttribuut;

    <type> naamPrivateMethode(<type> parameter1, <type>parameter2) {
      // code van privateMethode1
    }

  public:
    <type> publicAttribuut;

    // constructor:
    <Naam van class>(<type> parameter1, <type>parameter2) {
        // code van de constructor
    }

    <type> naamPublicMethode(<type> parameter1, <type>parameter2) {
      // code van privateMethode1
    }
 };
```

#### Voorbeeld

```
class Teller {
  private:
    int pin;
    bool wasOnderbroken;

    bool isOnderbroken() {
      return !digitalRead(pin);
    }

  public:
    Teller(int _pin) {
      pinMode(_pin, INPUT_PULLUP);
      pin = _pin;
      wasOnderbroken = false;
      aantal = 0;
    }

    void update() {
      if (wasOnderbroken == false && isOnderbroken() == true) {
        aantal++;
      }

      wasOnderbroken = isOnderbroken();
    }

    int getAantal() {
      return aantal;
    }
};
```