* Geheugenmodel classes
  + De rabbit class wordt aangemaakt. Elke Rabbit class heeft een prototype class
  + De methodes van de prototype kun je aanroepen, bijvoorbeeld Rabbit.prototype.speak()
  + De methode \_\_proto\_\_ is gelijk aan de prototype van het object
* Abstracte classes is mogelijk, maar dan moet je een lege class definieren. In de subclasses definieer je dan de methodes en functies die in de classes behoren.
* Array.prototype.map.call
* Bij javascript verwijst this niet altijd naar de this in de class zelf, waar dit bij java wel zo is.
* Static nog verder uitzoeken
* Libraries die code in de prototype aanpassen kunnen erg gevaarlijk zijn.
* Je kunt javascript classes aanmaken in dev tools om te testen
* Bij een normale functie kan de this gebruikt worden, en bij de arrow functie niet. De arrow functie kijkt om zich heen in dit geval. De arrow functie weet zelf niet wat this is.
* Je kunt methodes in een object aanroepen. Er wordt naar de prototype gekeken of deze dan bestaat, en zo niet, dan wordt er naar de prototype van de superklasse gekeken. Zo gaat het maar door, totdat het niet meer verder kan.
* De eventlistener verandert altijd the this element
* De debugger stopt het programma met uitvoeren op de regel waar deze gedefinieerd is.
* Function is de hoogste class in javascript.