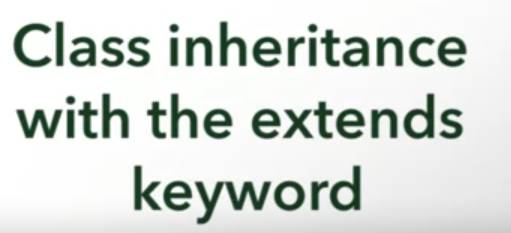
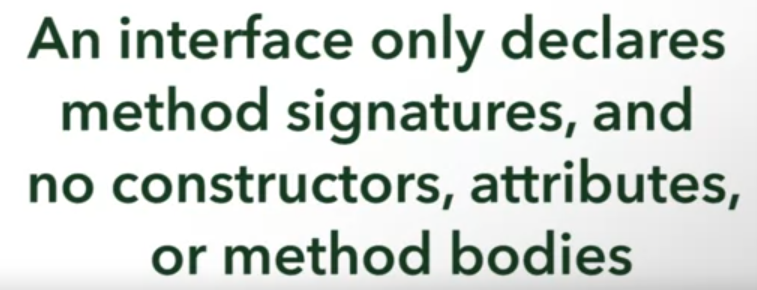




* Subclasses
  + Sample: Dog type must also behave as Animal type

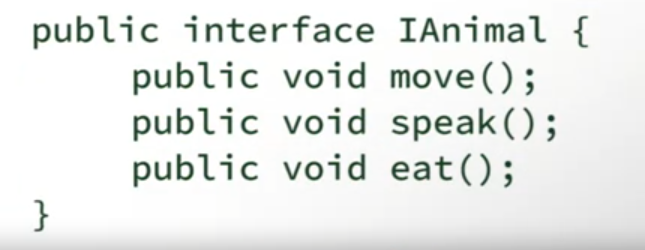


Java Interface

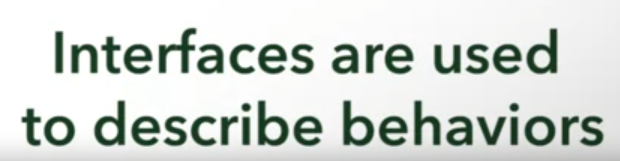


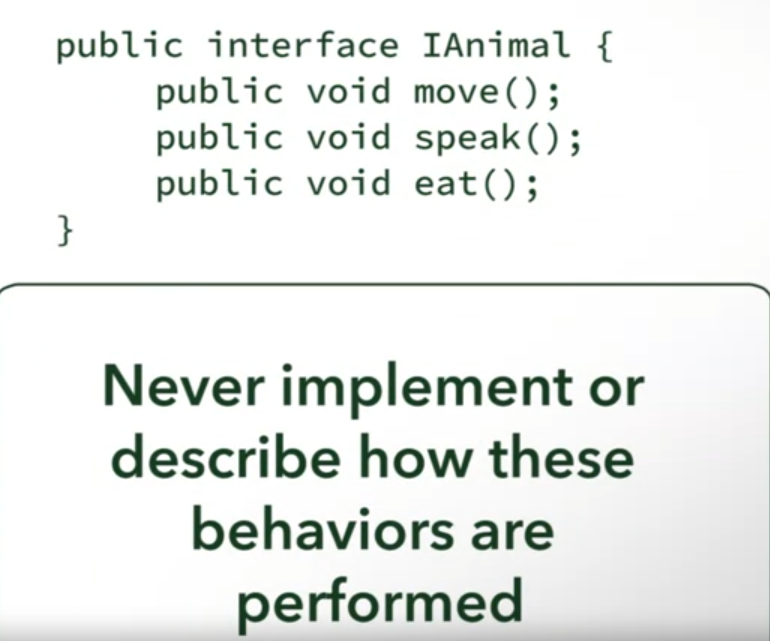


* Like a **contract** to be fulfilled in implementing classes
  + Like types
* Place letter I before the actual name to indicate that it is an interface

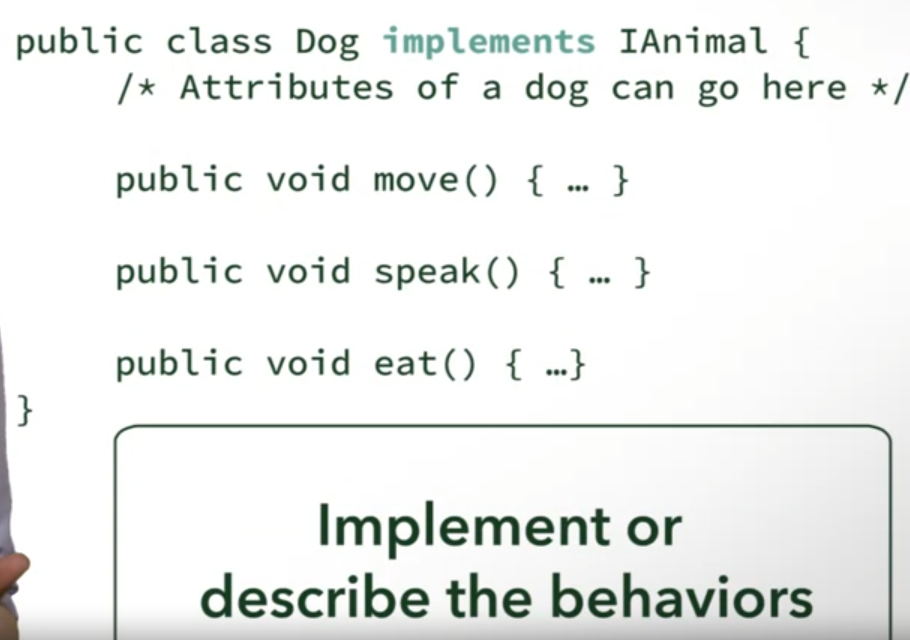


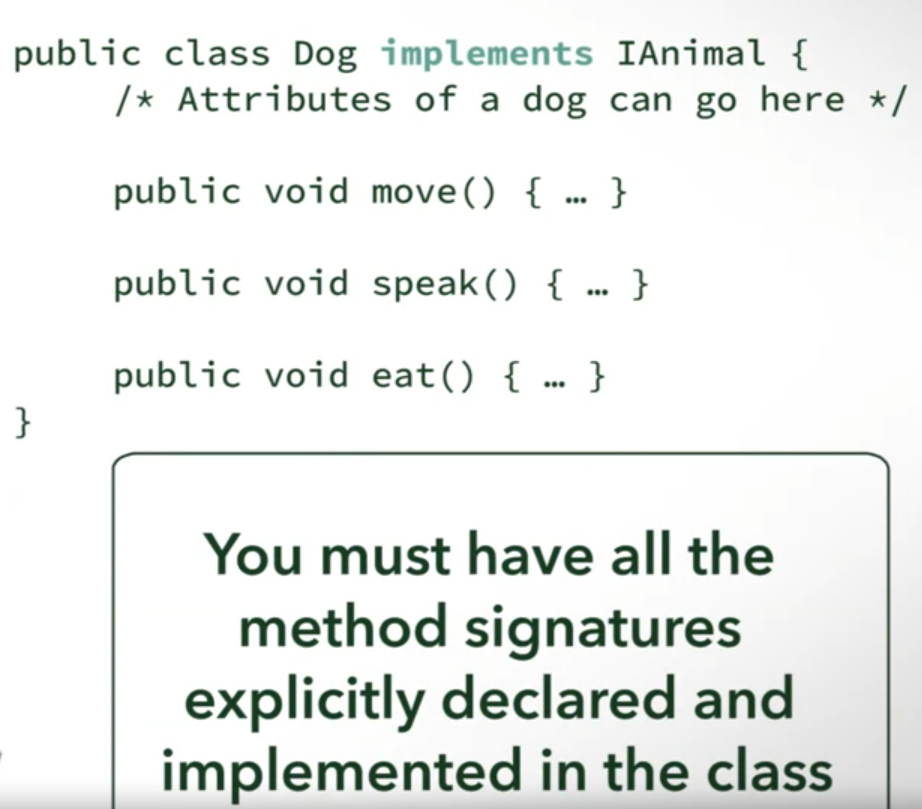


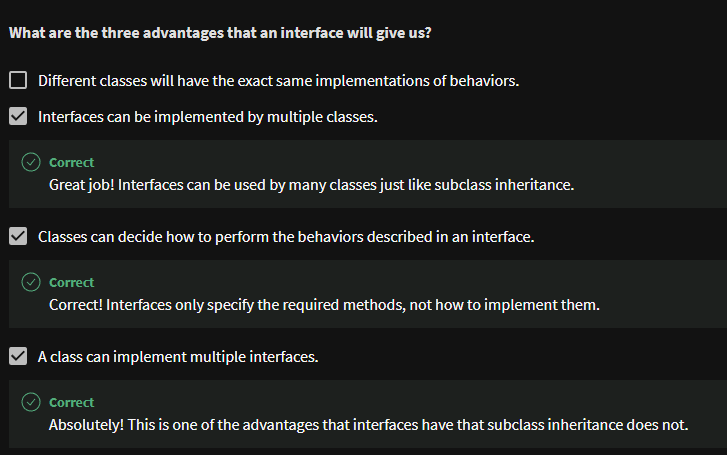




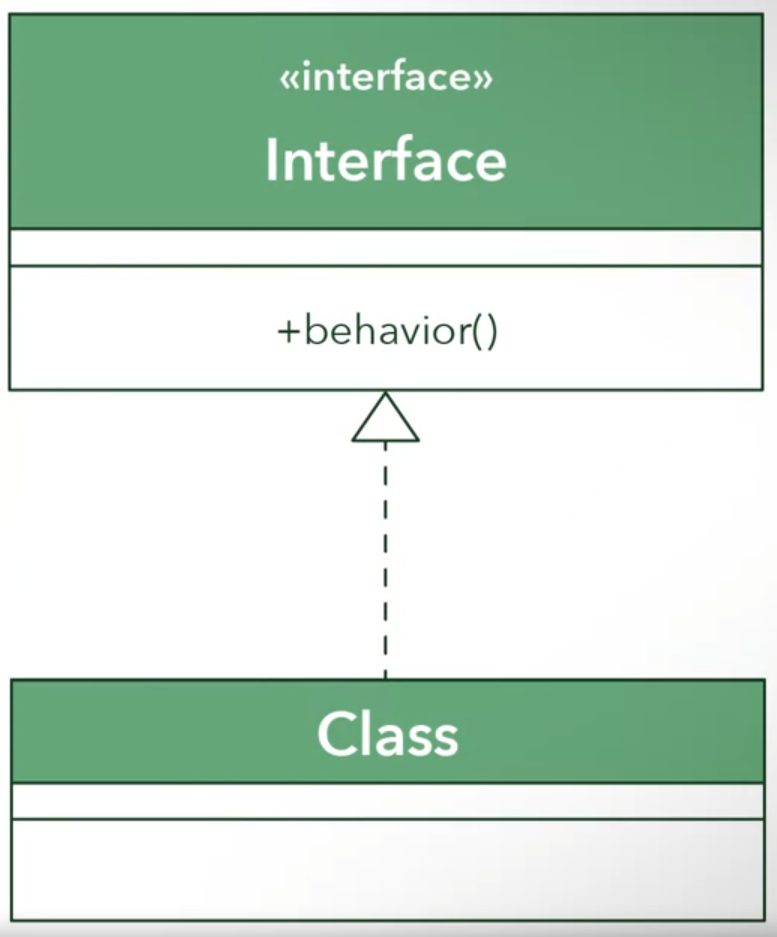
Keyword: **implements**

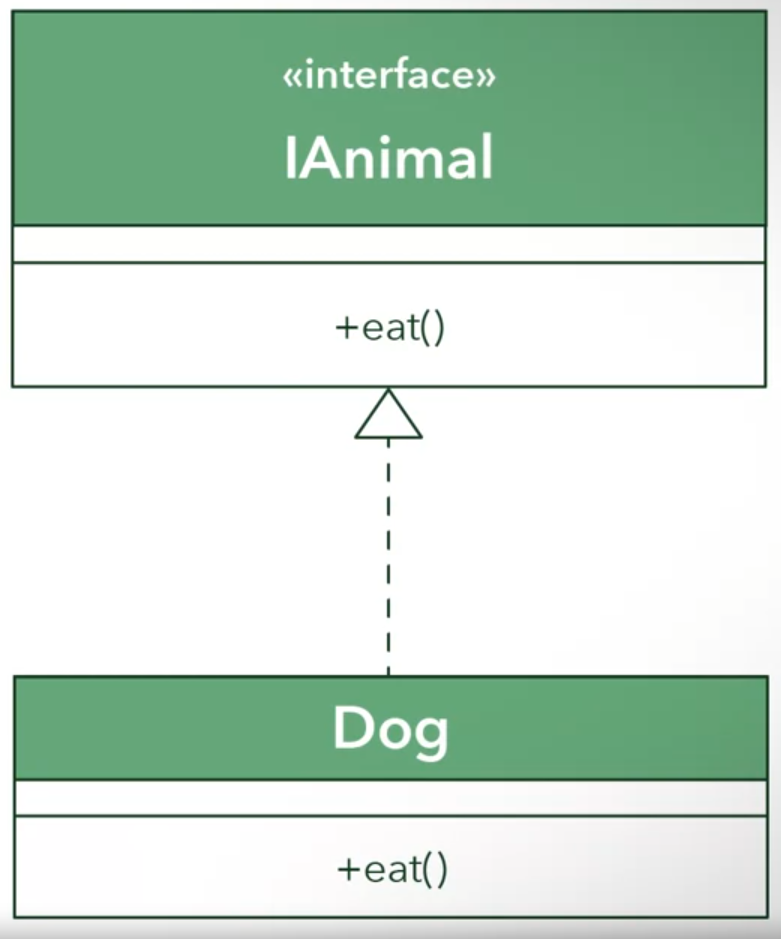






In UML:

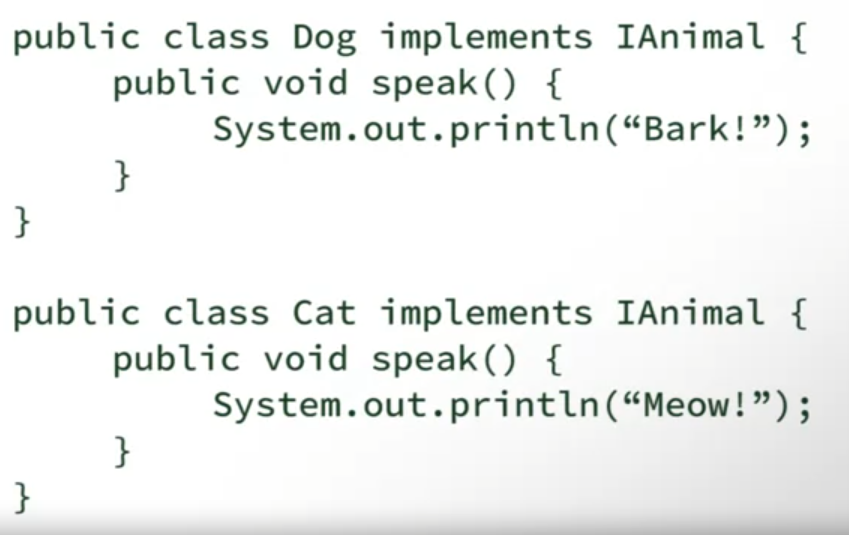


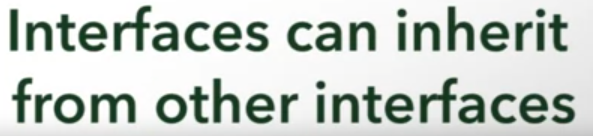
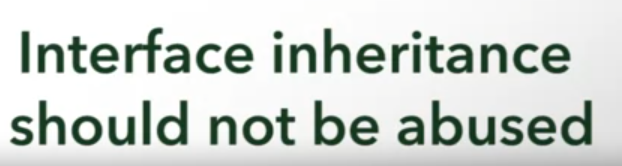


When to use interface or inheritance?

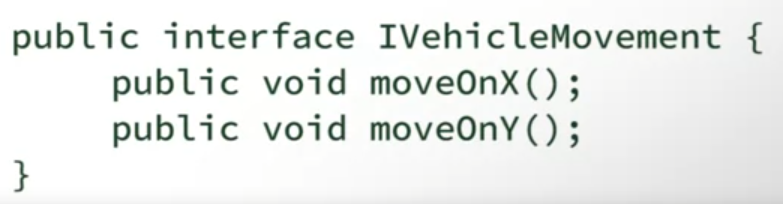
Interface

* Allows polymorphism like abstract classes

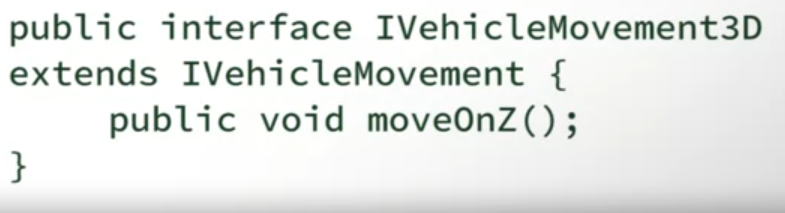


* 
* 

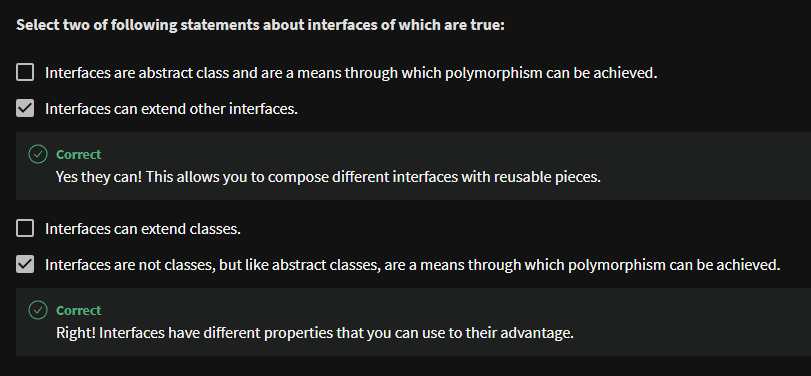
Another example of interface



* For 2d movements



* Extends the current interface to allow 3d movements without having to add the moveOnZ() on the first one
* This will not require vehicles with only 2d movements to have moveOnZ()



Multiple Inheritance

* Java doesn’t support this
* Introduce data ambiguity

Multiple Interfaces



