# Interfaces

* Close relatives to abstract classes.
* An interface introduces methods that the implementing class should implement.
* When a class implements an interface, it complies to behave like the interface instructs.
* An interface can have (from version 8.0) two kind of implementations of methods: default and static. All other methods must be implemented by the implementing class.
* A default method is what the name says. It makes it possible to update the interface without old code breaking.
* A static method is an interface dependent method which acts like a one declared in a class. Hard for me to grasp why these had to be included in the language.
* If an interface has properties, they are always static finals even if you don’t specify it explicitly.
* A class can implement more than one interface. I do not recommend it!
* An interface can be extended just like a class, but I rarely have found any need for it. That might be cause I’m stupid.
* An example interface for a deck of cards:

interface Deck {

public void shuffle();

public void deal();

}

* If a class implements this interface it agrees to behave like a deck of Cards. It also means that the implementing class must implement methods shuffle and deal. Remember that the interface does not have to know how the implementations look like.
* If you got interested about the details of default and static methods in interfaces, see: <https://softwareengineering.stackexchange.com/questions/233053/why-were-default-and-static-methods-added-to-interfaces-in-java-8-when-we-alread>
* Examples: ErrorUser, Error, Abs, RMSE
* Examples: InterfaceUser, InterfaceBase, InterfaceExtender, InterfaceImplementerBase, InterfaceImplementerExtender