# Abstract classes and private constructors

* The upper the class is in the inheritance path, the simpler it is. For example, the class Object is the base class for all other classes, and it does not represent anything in particular.
* If the class is simple enough it can be made *abstract*.
* If a class is made abstract it only provides a partial solution to a problem. Therefore, it cannot be instantiated. So, the purpose of an abstract class is that some other class extends it.
* An abstract class can however have fully functional properties and methods. The only rule is that the class must have at least one method without a solution. It has methods without solution, because it does not know how to implement them.
* Private constructors are defined when you want to fully control the creation of objects, or you don’t want your class to be extended.
* If you make a class that cannot be extended you probably are coding it the wrong way. My opinion.
* Usually when you want to control the creation of your objects, you want to somehow restrict the number of objects created.
* Examples: AbstractTest ja SingletonInt.