11.7 Classes and Methods - Nouns and Verbs

One way to think about classes and objects is that they represent things – cards, ArrayLists, Scanners, bank accounts, details about people. Some of these things are software models of real things (cards); and some are only software things (ArrayList) – but they are all in at least some sense *things*. This means that the names of classes (and objects) tend to be **nouns** – words used to name things.

Conversely, the methods in a class represent actions – you call them to do something. This means their names tend to be **'verbs'** – words used to represent actions. I've put 'verbs' in '' because most method names are made up of more than one word so, strictly, they can't in linguistic terms be actual verbs – but they do represent *actions*. For example, consider this example from the last chapter:

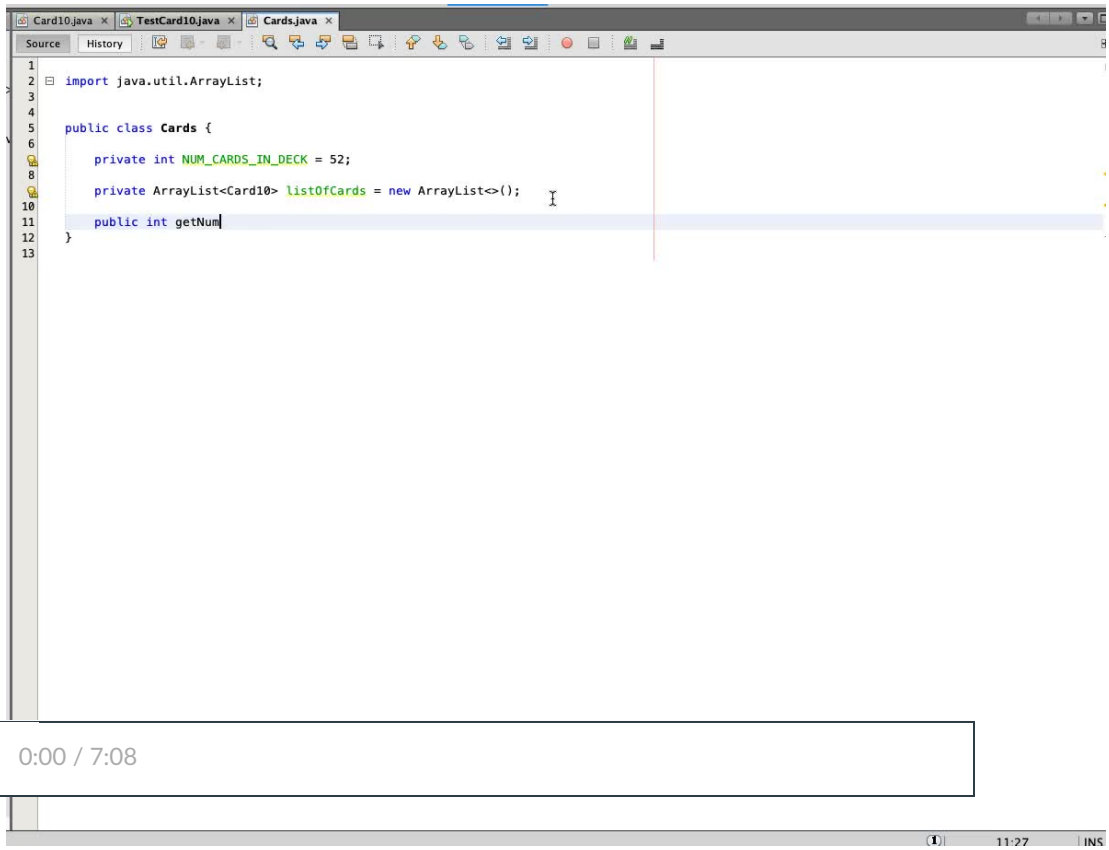
```
Card card = new Card();
card.setSuite("Hearts");
card.setNameValue("Ace");

String cardName = card.getName();
```

The names of the class (`Card`) and object (`card`) represent *things* so they have names, which are *nouns*. On the other hand, all the methods describe *actions* (they are 'verb-like') because they all represent *something that is being done.*

KEY POINT: Naming Classes, Objects and Methods

- When choosing the names of classes and objects use **nouns** – words that represent names of things.
- If the class represents one thing, pick a *singular* noun - not a plural one. So our Card classes represent one card - and they are called 'Card' *not* 'Cards'. If on the other hand I was writing class the represented a *list* of cards, then 'Cards' would be the correct name.
- When choosing the names of methods either use **verbs** for single-word names; or sequences of words that describe **actions** – also see the two points below.
- When choosing names to set or get the values of properties (variables or things that could be variables),start them with **set** and **get** respectively unless they return values which are boolean.
- When choosing the names of methods that return boolean values, begin them with **is**.

The video below talks about how we should name classes, methods; and it also talks about static:



11.7.1 Is this a bit much?

You might be thinking that we are writing a lot of code for - in some cases - quite simple things. Do we have to do this? Well - it depends. If *and only if* you are dealing with data that will *never* be exposed to the outside world, there is an alternative called a *record.* This is basically a version of a class that is intended to be used as simple data storage and not to have a lot of extra functionality (methods basically - though you can have those too if you want). This can cut down the amount of code you have to write (a lot) - but it does come with the disadvantage that the "workings" of the code are exposed. So you will end up with problems later because - inevitably - other code will be written that depends on that "working". But *provided* it is for things that are only ever going to be internal to your code, it's useful. Records come with some other restrictions too - for example they are *immutable* (final). We're not going into detail here - but you can read about them online.