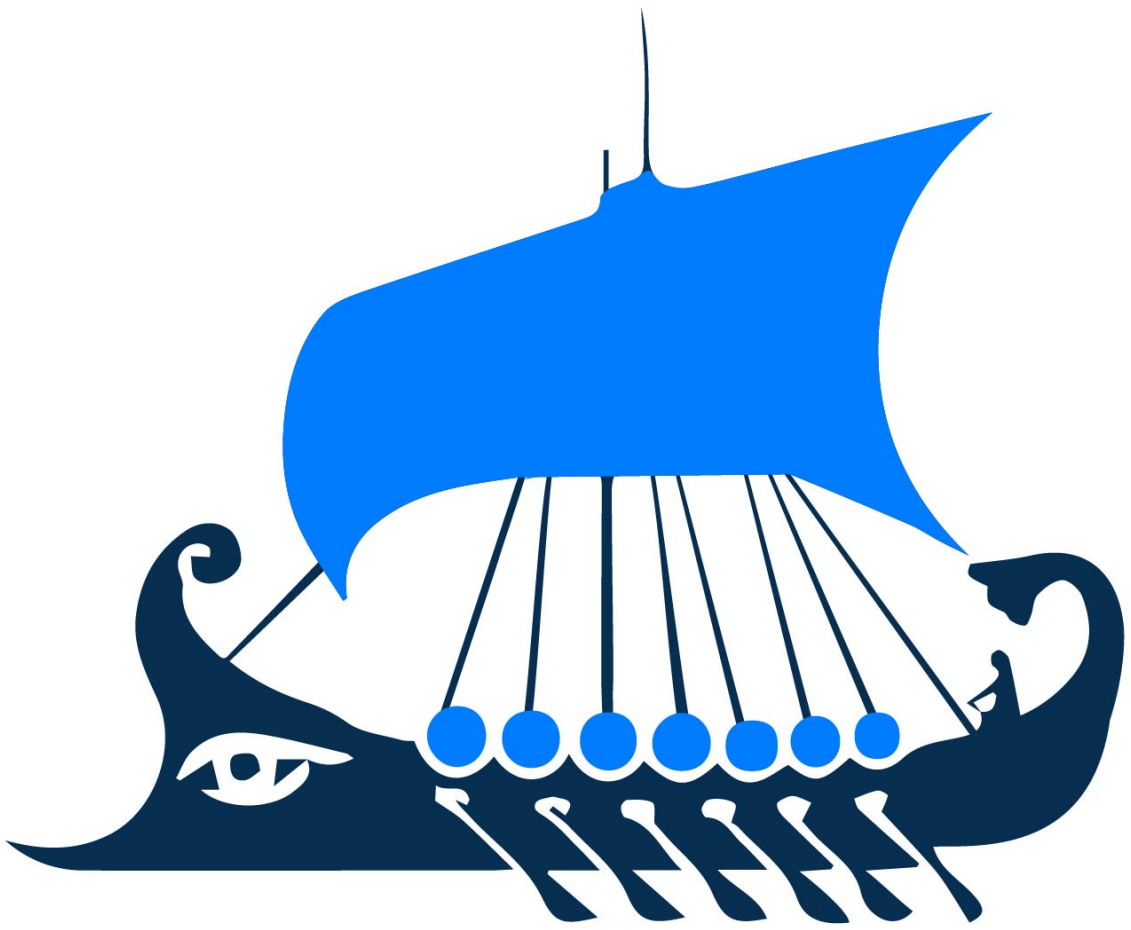
Theory: Immutability

9 minutes 7 / 7 problems solved

Start practicing

In philosophy, there is a thought experiment called "Ship of Theseus". It is one of the oldest concepts in Western philosophy and it goes like this.

The famous ship of the hero Theseus has been kept as a museum piece in a harbor. Over the years, the wooden parts have rotted and been replaced. A hundred years later, all of the wooden parts of the ship have been replaced. The question is: is it still the same ship?



There are many ways to answer this question and you can check them out on Wikipedia. What is of interest to us is that this question of identity can be applied to programming as well. One of the most important concepts in programming, the one directly connected to change and identity, is the concept of **(im)mutability.**

Current topic:

✓ Immutability ...

Topic is required for:

✓ Boxing and unboxing ...

✓ Hashable ...

✓ Objects in Python ...

Table of contents:

§1. Mutability and immutability

Mutability literally means "the quality of being changeable" and, in programming, it refers to the idea of changing the state of the object after it has been created.

Along this line, we can distinguish between **mutable** and **immutable** objects. To put it simply, mutable objects can be altered once they've been created and immutable cannot. This is the key difference between mutable and immutable objects.

What does this mean in practice? Immutable objects always represent the same value: if you want to have a different value, you need to create a completely new object or reassign the value of the existing one. With mutable objects, things are much easier and we can change the values they contain without reassigning the variable or creating a new object.

Returning to the ship of Theseus, if we were to consider it an immutable object, then the answer to the question of identity would be no. Once you change something, you have a different object. In other words, the ship of Theseus is no longer the ship of Theseus even though it has the same name!

Alternatively, if we regard the ship as a mutable object, then, yes, it is still the same ship. The changes that we made did not affect its identity.

Depending on the programming language you're using, different types of objects may be immutable. For instance, strings are immutable in Python and Java, but Java also has another type for strings which is mutable. In Ruby and PHP strings are mutable. When writing a program in your favorite language, you need to take into account which objects are mutable and which are immutable in that particular language.

§2. Custom objects and immutability

In general, objects of custom classes are mutable. However, there are cases when we would want to make them immutable: immutable objects are thread-safe, easier to test and they may be more secure.

> Immutable objects can be shared between different threads without additional protection. The state of mutable objects is hard to follow as long as they can be changed by any of the working threads.

In the context of custom objects, we can also talk about **weak immutability** and **strong immutability.** Weak immutability is when some fields of an object are immutable and others are mutable. Strong immutability is when all fields of an object are immutable.

Specific instructions on how to make a custom class immutable depend on the language, but we can give general guidelines. Basically, you need to forbid changing the value of the field once it has been created or forbid reassigning the value. This can be done, for example, by making the variable read-only or a constant. Another option is to modify the methods that set attribute values so that they throw exceptions. You can also work with access modifiers: make the fields unattainable from the outside of the class.

§3. Summary

To sum up, the difference between mutable and immutable objects lies in the fact that mutable objects can change their states after creation and immutable objects cannot. Languages have their own division into mutable and immutable objects. Custom classes are usually mutable but can be made immutable using language-specific tools and techniques if necessary.

Report a typo

736 users liked this theory. **12** didn't like it. **What about you?**

😍 🙂 😐 ☹️ 😡

Start practicing

Comments (19) Hints (2) Useful links (7) Show discussion