<https://www.youtube.com/watch?v=D4Dja5WSZoA>

Use a getCurrentLearner() method. Does a lookup based on current user’s ID. Return NullLearner if no user found. The NullLearner has default return values for its methods.

*null* is the misguided invention of British computer scientist Tony Hoare (most famous for his Quicksort algorithm) in 1964, who coined his invention of *null references* as his “billion dollar mistake”.

Every null check is a branch in your code. If the object is null, do one thing. If it is not null, do another.

The more branches in your code, the more complex the code becomes. This is measurable using a metric known as cyclomatic complexity. Cyclomatic complexity counts the number of branches in your code, say a given method, and then scores it. A higher score means more complex code. Simple.

Thus, every null check increases our cyclomatic complexity score.

Using the Null Object pattern appropriately will reduce the number of null checks needed in your code, thus the number of code branches reduces and your cyclomatic complexity reduces for this body of code.

Etc.

Super simple

The NOP replaces conditional checking with polymorphism as we’ll see.

Use args to a constructor

Implement a shared interface to create the Null type

“We have a real thing, but it does nothing when we use it”