# Overview:

* The Proxy design pattern provides a surrogate or placeholder for another object to control access to it.
  + Used when we want to provide controlled access for a functionality.
* The formal definition of a proxy is a person authorized to act for another person.
  + An agent or substitute.
  + The authority to act for another.
* There are situations in which a client does not or cannot reference an object directly, but wants to still interact with object.
  + Introduces a level of indirection when accessing an object.
* A proxy object can act as the intermediary between the client and the target object.
* Another common use case is to provide a wrapper implementation for better performance.

# Examples:

* A check or a credit card is a proxy for what is in a bank account
  + Can be used in place of cash.
    - Provides a means of accessing that cash when required.
  + Exactly what a proxy does, it controls and manage access to the object they are protecting.
* In a classroom, when one student is absent, during roll call; his best friend may try to mimic the student’s voice to try to keep his friend from being marked as absent.
* Let’s say we have a class that can run some command on a system:
  + If we are using it, it works fine.
  + If we want to give this program to a client application.
    - Can have severe issues because client program can issue commands to delete some system files or change some settings that you do not want to.
  + A proxy class can be created to provide controlled access of the program.