# Inversión de dependencia, arquitectura hexagonal y otras yerbas

La inversión de dependencia es uno de los 5 principios SOLID. Se lo suele definir como:

* *Las clases de alto nivel no deberían depender de las clases de bajo nivel. Ambas deberían depender de las abstracciones.*
* *Las abstracciones no deberían depender de los detalles. Los detalles deberían depender de las abstracciones*.

Los problemas a evitar son los siguientes:

* **Las parte más genérica de nuestro código (el dominio o lógica de negocio) dependerá por todas partes de detalles de implementación**. Esto no es bueno, porque no podremos reutilizarlo, ya que estará acoplado al framework de turno que usemos, a la forma que tengamos de persistir los datos, etc. Si cambiamos algo de eso, tendremos que rehacer también la parte más importante de nuestro programa.
* **No quedan claras las dependencias**: si las instancias se crean dentro del módulo que las usa, es mucho más difícil detectar de qué depende nuestro módulo y, por tanto, es más difícil predecir los efectos de un cambio en uno de esos módulos. También nos costará más tener claro si estamos violando algunos otros principios, como el de Responsabilidad Única.
* **Es muy complicado hacer tests**: Si tu clase depende de otras y no tienes forma de sustituir el comportamiento de esas otras clases, no puedes testearla de forma aislada. Si algo en los tests falla, no tendrías forma de saber de un primer vistazo qué clase es la culpable.

Ejemplo: se requiere que la clase ElectricalPowerSwitch definida en code\_01.py pueda manejar un ventilador ademas de una luz