Mocking

Mocking – техника, която се прилага, когато тестваме код с някакви dependency-та. Mocking-а ни позволява да тестваме клас, който зависи от други класове. Чрез „фалшифициране“ на дадени класове, ние можем да run-ваме unit test-овете, които зависят от тези класове

За да пишем тестваем код трябва да използваме **Dependency** **Inversion** (Inversion of Control Pattern) – предоставя decoupling между отделните модули на програмата и самата програма. Всеки модул може да се съсредоточи върху това което прави

* Public API should work with interfaces, not implementation classes (IEnumerable vs. List). Т.е. е важно колкото е възможно по- често да използваме интерфейси, а не имплементации на конкретни класове

Пример:

Bad code:

**public Card[] Cards { get; private set; }**

**или**

**public List<Card> Cards { get; private set; }**

Good code:

**public IList<ICard> Cards { get; private set; }**

**по-добре да работим с интерфейси отколкото с конкретни класове**

предимство: например ако искаме в Engine-а на дадена игра, да променим компонента, който рендерира обектите, то не трябва да бъркаме в Engine-а, а просто да променим компонента, който подаваме отвън

ако даден клас използва дадено dependency само веднъж, в даден метод, то е по-добре да подаваме това dependency само на конкретния метод, а не на целия клас през конструктора

Не винаги, в 100 % от случаите, трябва да създаваме интерфейс за всеки компонент, който създаваме. Понякога, ако знаем, че ще използваме само една конкретна имплементация, и че няма да ни трябва друга вариация, подобна на тази имплементация, то може да избегнем правенето на интерфейс за конкретния компонент

Moq (Mock You) can mock almost any type, not just interfaces. E.g. Random, DateTime, etc.