**Паттерны для QA**

**https://www.youtube.com/watch?v=hXFSlZM83Ns**

**Singleton –** 3-й вариант (для многопоточки)

**Builder**

**Decorator** – для QA это будет механизм аннотаций (как готовых, так и самими написанных)

**Observer** – варианты, когда используется: 1. Один класс зависит от другого; 2. слушатель, который в отличие от Decarator-а, будет реализован не всегда, а только в случае выполнения каких то условий/действий *(класс Observable и интерфейс Observer deprecated, см.примеры https://javarush.com/groups/posts/3421-shablon-nabljudateljh-observer)*; 3. ослабление связей между классами.

В тестировании применяется вариант\_1. при многократном заполнении полей сущностей разными параметрами, используя класс Random.

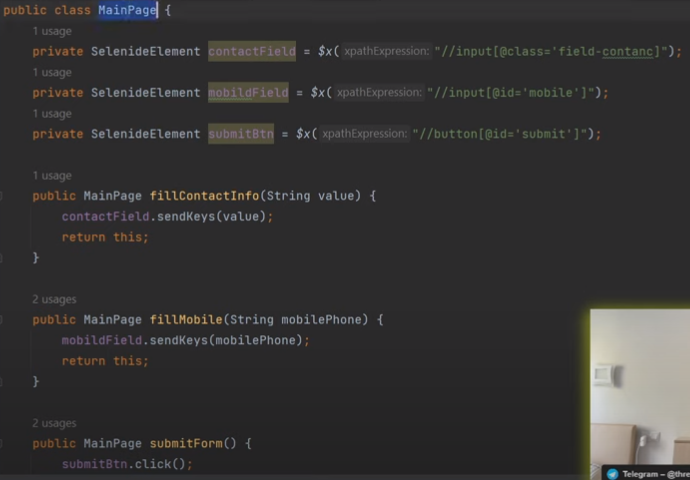
**Factory**

**Facade**

**Page object** – в QA используется, когда нужно многократно переиспользовать в разных классах одни и теже элементы на странице. При этом разносится по отдельным классам логика действия с элементами на странице и сами тестовые классы.

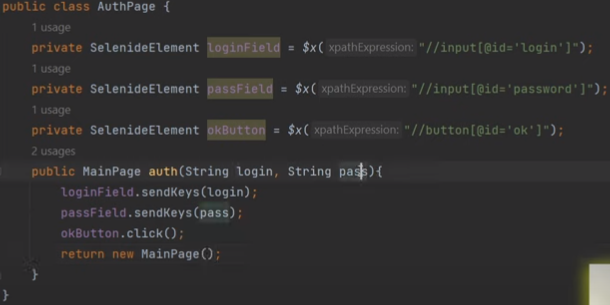
Вначале страница бьется на отдельные группы элементов с которыми выполняют какие-то действия отдельные классы:

- действия с элементами на странице:



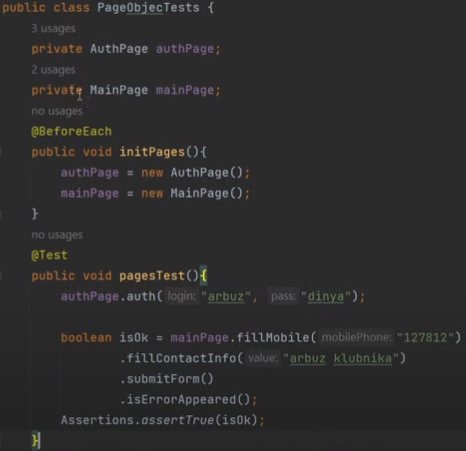


- авторизация на странице:



После чего в Тестах вызываются готовые методы этих классов:

Вначале авторизуемся, потом проводим какие то действия с элементами на странице



**Data Driven Test –** параметризированные тесты с ипользованием аннотации @ParameterizedTest над методом.