3.Abstracting Selenium

Introduction

Each test contains a set of Selenium commands that needs to be repeated in each test. When something needs to be changed to the provided command, that the change needs to be implemented in all tests and all classes.

Abstracting Selenium

In order to provide a way of constructing a set of general abstract methods over Selenium, a set of action-oriented abstraction over the main Selenium commands can be built. This means that if something is not proper writing, you can modify the general methods and re-use them in the code. A best approach could be when a new version of Selenium Webdriver is launched and you need to change only the abstraction class and not the entire tests.

As an example, we can have:

```
public class PageDriver {
public static WebDriver driver;

public static void fillText(By locator, String fieldValue){
 driver.findElement(locator).sendKeys(fieldValue);
}
public static void click(By locator){
 driver.findElement(locator).click();
}

public static void navigate(String URL) {
 PageDriver.driver.navigate().to(URL + "/");
}

public static void close(){
 PageDriver.driver.quit();
}
}
```