ТЕСТИРОВАНИЕ ДОБАВЛЕНИЯ ЗАКАЗОВ В КОРЗИНУ НА САЙТЕ DOMINOS.BY

Поставленные задачи

1. Зайти на домашнюю страницу сайта.
2. В разделе «Пицца» добавить в корзину пиццу «Маргарита».
3. В разделе «Напитки» добавить в корзину любой напиток.
4. Перейти в корзину.
5. Проверить, что пицца и напиток есть в заказе.

Применяемые фреймворки и паттерны:

Junit	Page Object
Selenium	Page Factory

Реализация добавления товаров в корзину

Проектом предусматриваются унифицированные методы класса AbstractPage для добавления товаров в корзину и перехода по категориям.

Для формирования локатора используется Enum содержащий наименования типов продуктов и уникальные значения локатора, используемого в шаблоне, для добавления товара в корзину.

Аналогичные методы реализованы для переходов по категориям товаров.

```
public DrinksPage addToShoppingCart(DrinkType type) {
    addToShoppingCart(type.getXpathValue());
    return this;
}
```

```
protected AbstractPage addToShoppingCart(String xpathValue) {
    String xpathPattern = "//div[@data-code='%s']//descendant::button[@data-action='a2b']";
    String xpath = String.format(xpathPattern, xpathValue);
    webDriver.findElement(By.xpath(xpath))
            .click();
    return this;
}
```

```
public enum DrinkType {
    COCA_COLA_1L("KK1", "Кока-Кола 1 л");

    public final String xpathValue;
    public final String name;

    DrinkType(String xpathValue, String name) {
        this.xpathValue = xpathValue;
        this.name = name;
    }

    public String getXpathValue() {
        return xpathValue;
    }

    public String getName() {
        return name;
    }
}
```

Получение позиций заказа

Для дальнейшей проверки правильности добавленных товаров в корзину, выполняемой в тесте, необходимо получить список наименований всех позиций корзины.

Для извлечения наименований необходимо открыть корзину, вызвать представленный метод и передать результат извлечения в список.

```
@FindBy(xpath = "//div[@class='order-products__added-products']//descendant::div[@class='product-card__title']")
private List<WebElement> ordersPositionFromShoppingCart;

public List<String> getOrderPositionNameFromShoppingCart() {
    List<String> ordersPositionName = new ArrayList<String>();
    for (WebElement position : ordersPositionFromShoppingCart) {
        ordersPositionName.add(position.getText());
    }
    return ordersPositionName;
}
```

Для сверки полученных наименований с ожидаемыми необходимо сформировать список наименований, извлеченных из Enum.

```
List<String> expectedOrderPositionName = Arrays.asList(PizzaType.MARGARITA.getName(), DrinkType.COCA_COLA_1L.getName());
```

Содержание тестового метода и структура проекта

```java
public class OrderVerificationTest extends AbstractTest {

    @Test
    public void orderVerification() {
        MainPage mainPage = new MainPage();
        mainPage.closePopupBanner()
                .openCategory(Categories.PIZZA);
        PizzaPage pizzaPage = new PizzaPage();
        pizzaPage.addToShoppingCart(PizzaType.MARGARITA)
                .openCategory(Categories.DRINKS);
        DrinksPage drinksPage = new DrinksPage();
        drinksPage.addToShoppingCart(DrinkType.COCA_COLA_1L);
        List<String> actualOrderPositionName = drinksPage.openShoppingCart()
                .getOrderPositionNameFromShoppingCart();
        List<String> expectedOrderPositionName = Arrays.asList(PizzaType.MARGARITA.getName(), DrinkType.COCA_COLA_1L.getName())
        Assert.assertEquals(expectedOrderPositionName, actualOrderPositionName);
    }
}
```