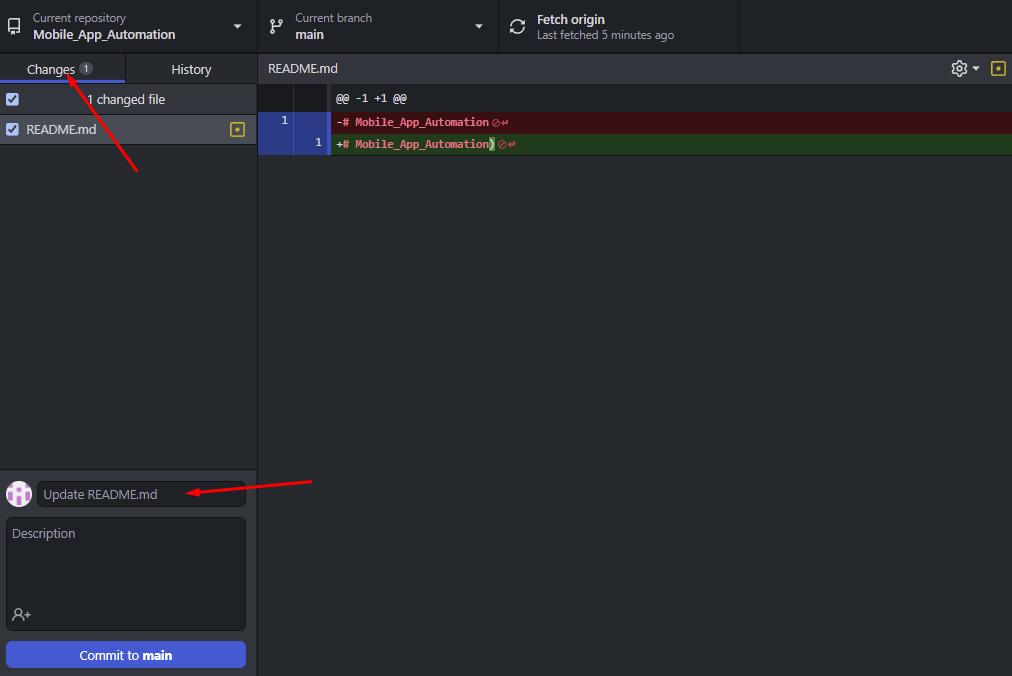
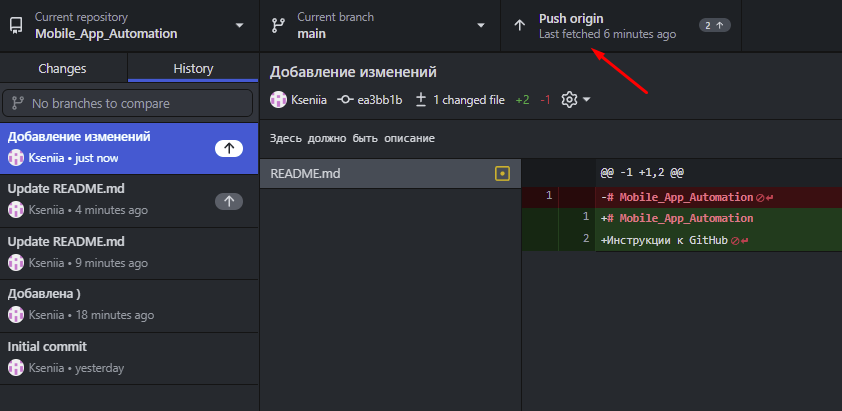
# Урок 0

## Инструкции к GitHub

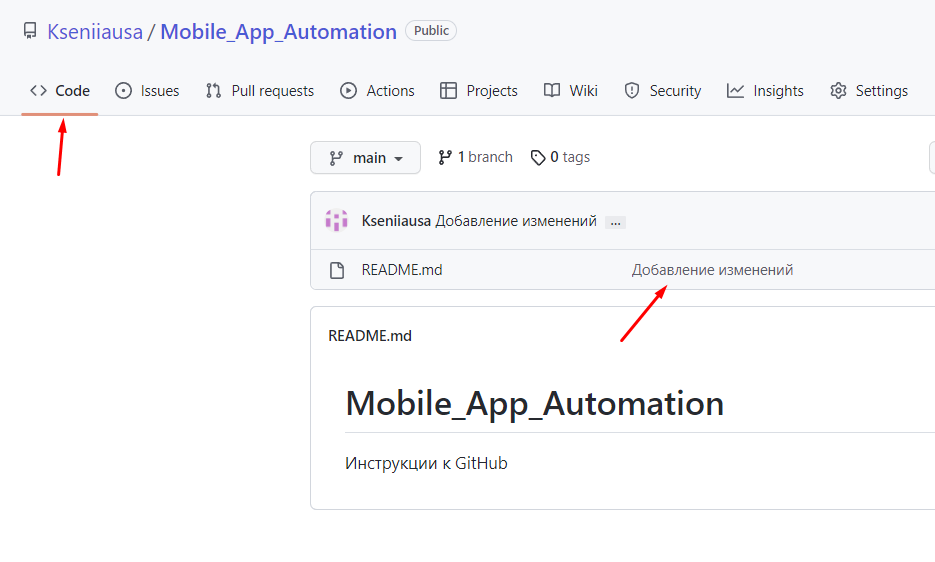
1. При внесении изменений можно посмотреть изменения и добавить комментарий:



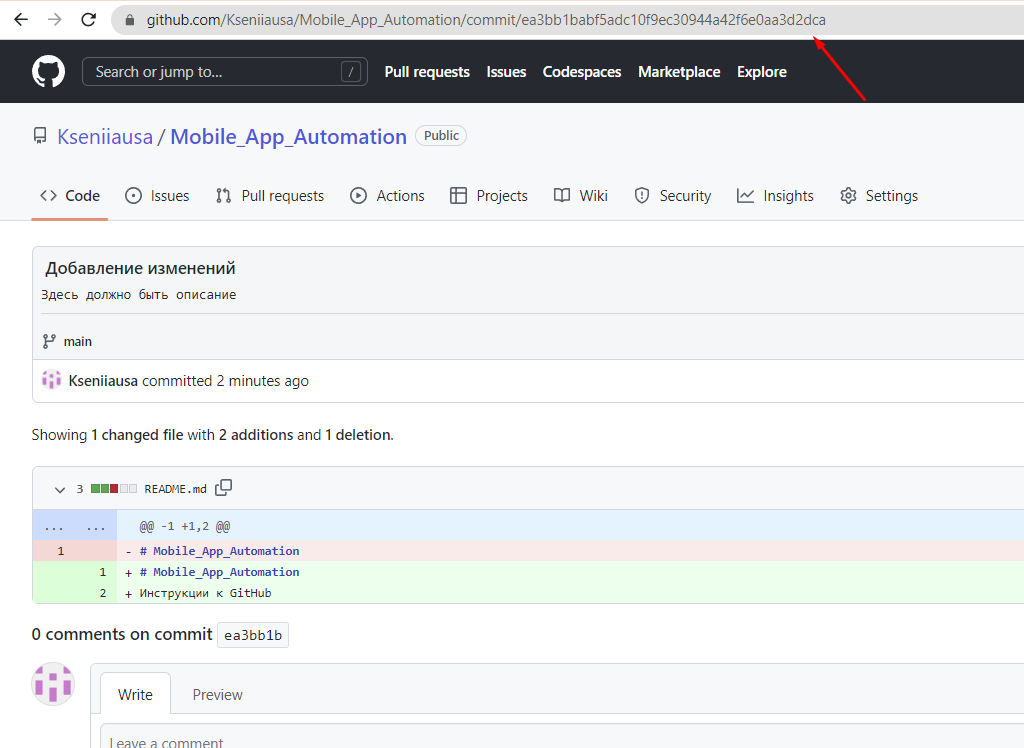
1. Добавляем комментарий и жмем кнопку коммит
2. Изменения сохранены локально, просмотр вкладка History
3. Для отправки на удаленную машину нажать PUSH – двухсторонняя синхронизация.



1. Изменения отображаются на удаленном доступе

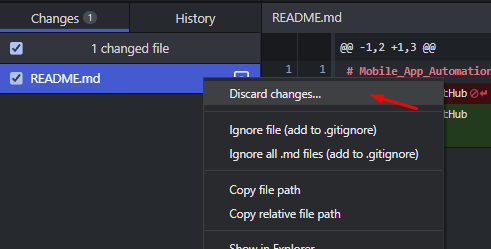


1. При переходе по коммиту открываются изменения, ссылку на которых необходимо отправлять на проверку



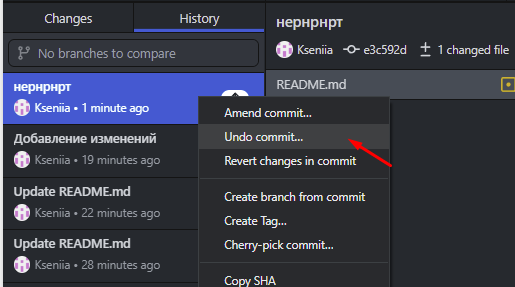
### Вернуть изменения

1. Внесли изменения в файл и сохранить
2. Перейти в ГитД и выбрать:



### Вернуть изменения с коммитом

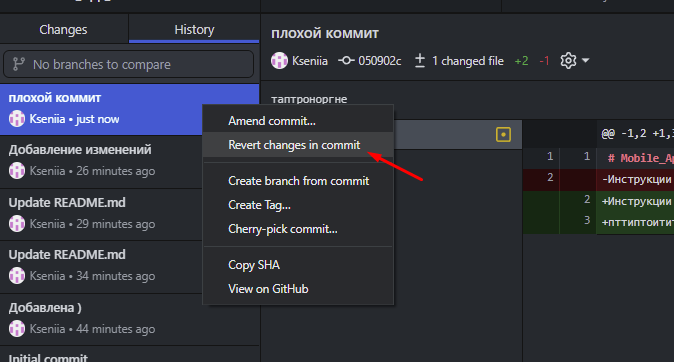
1. Внесли изменения в файл и сохранить
2. Отправить коммит
3. Перейти в ГитД и выбрать:



1. Для отмены изменений выполнить шаги [выше](#_Вернуть_изменения)

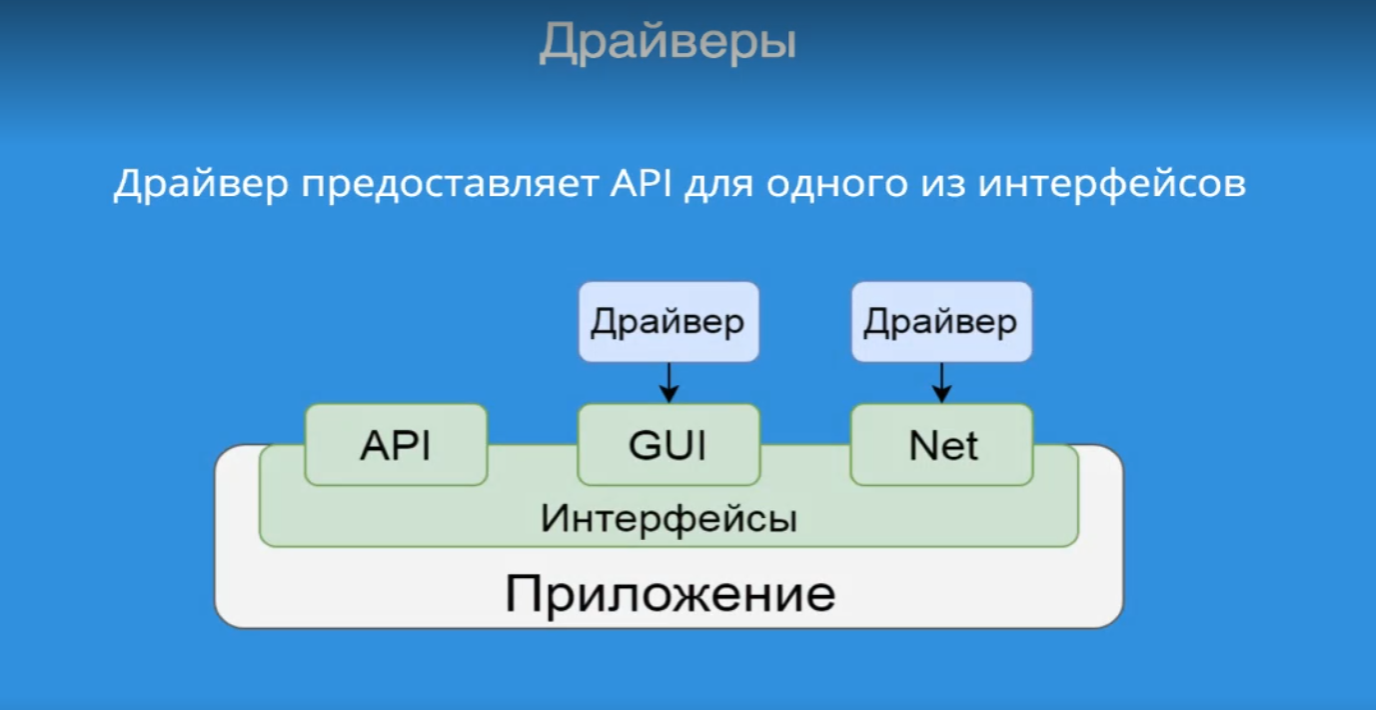
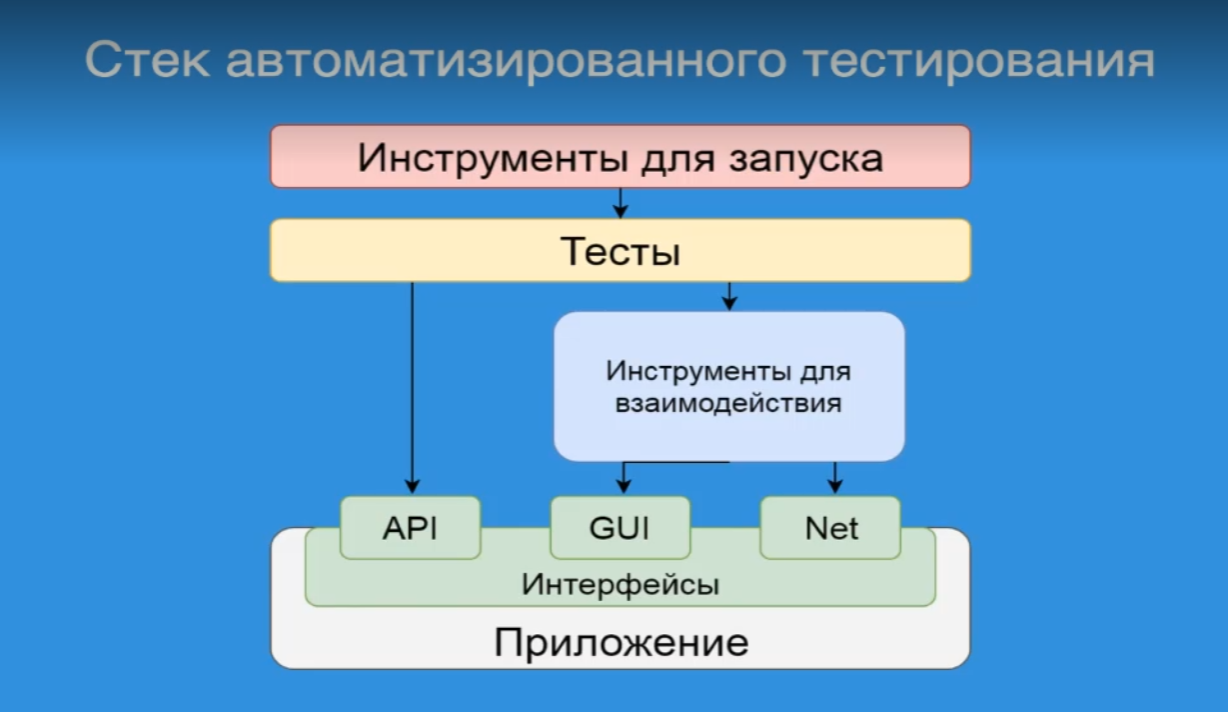
### Вернуть изменения с коммитом и уд. синхронизацией

1. Внесли изменения в файл и сохранить
2. Отправить коммит
3. Выполнить синхронизацию
4. Перейти в ГитД и выбрать Revert – локально изменения отменены
5. Для отправки на удаленную машину нажать PUSH

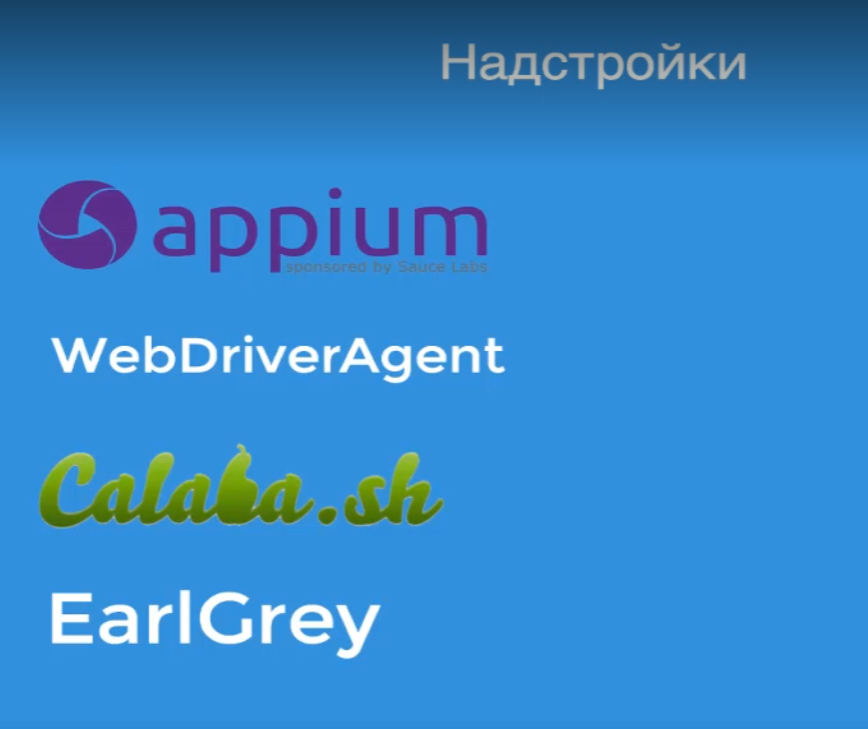
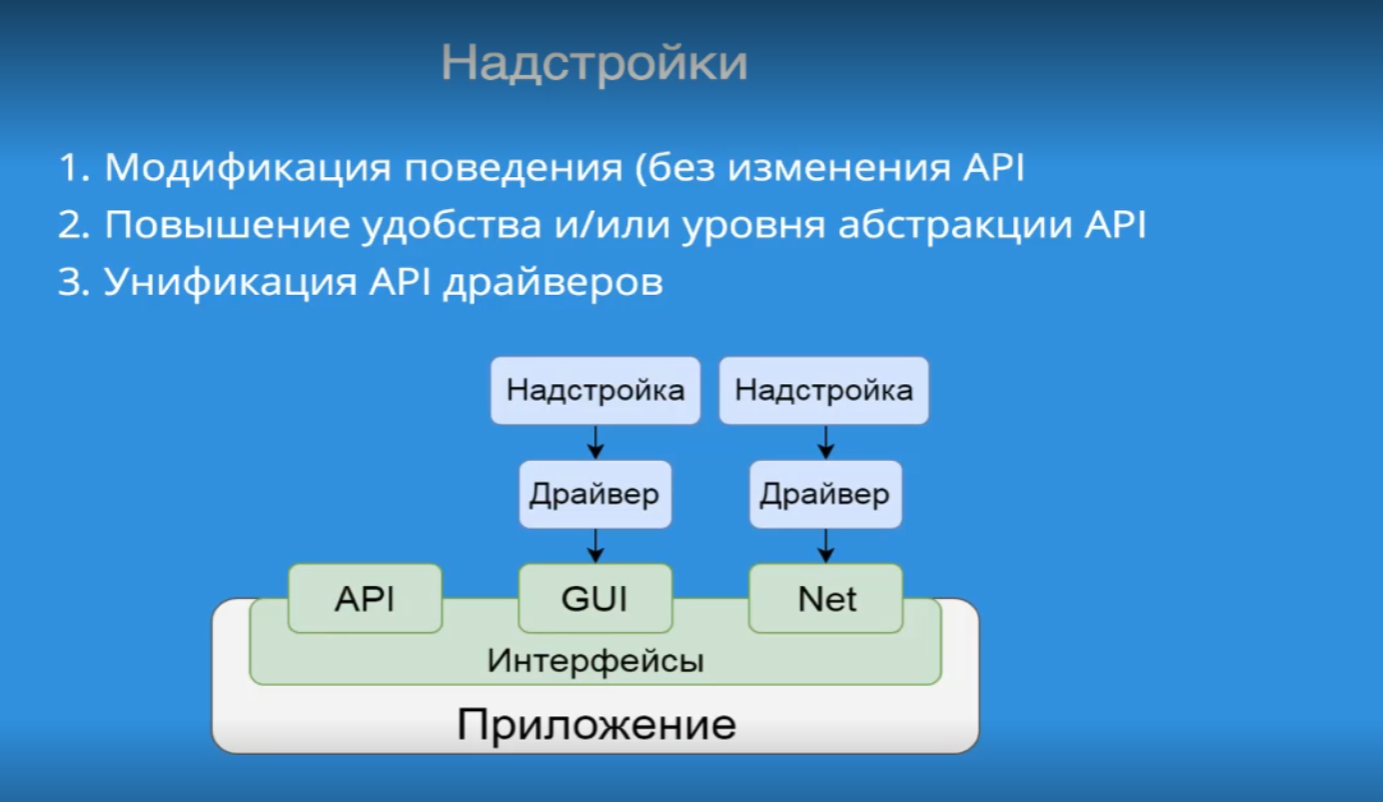


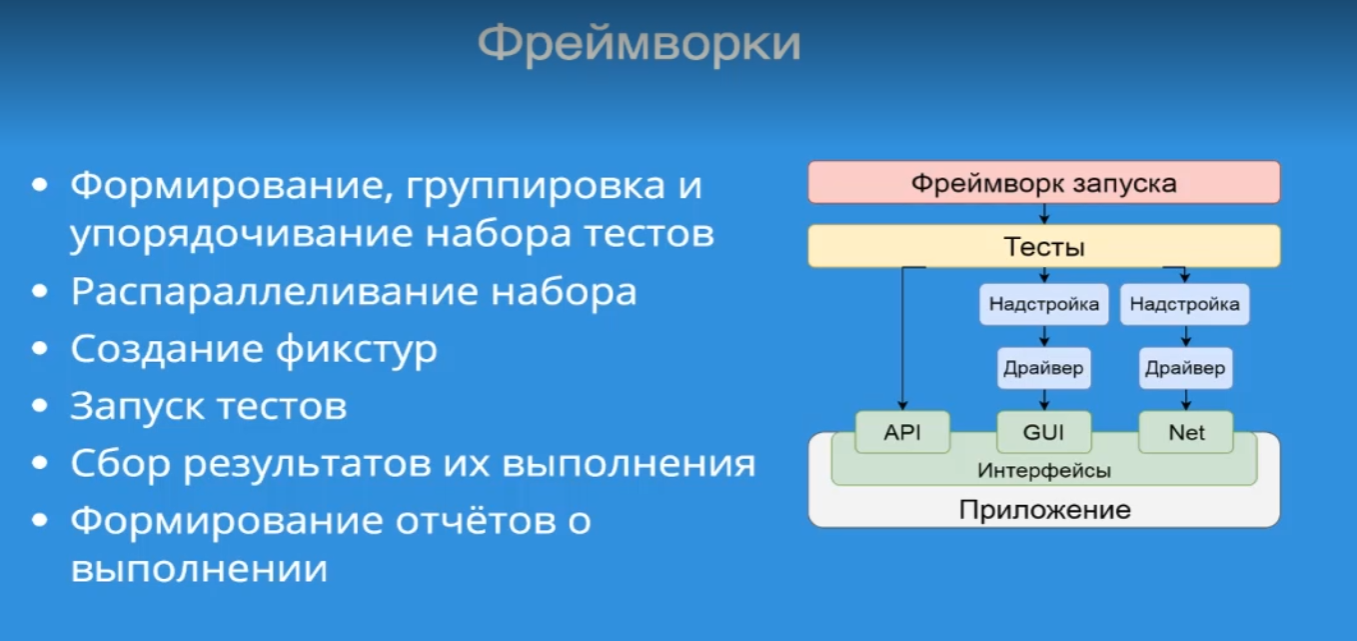
# Урок 1

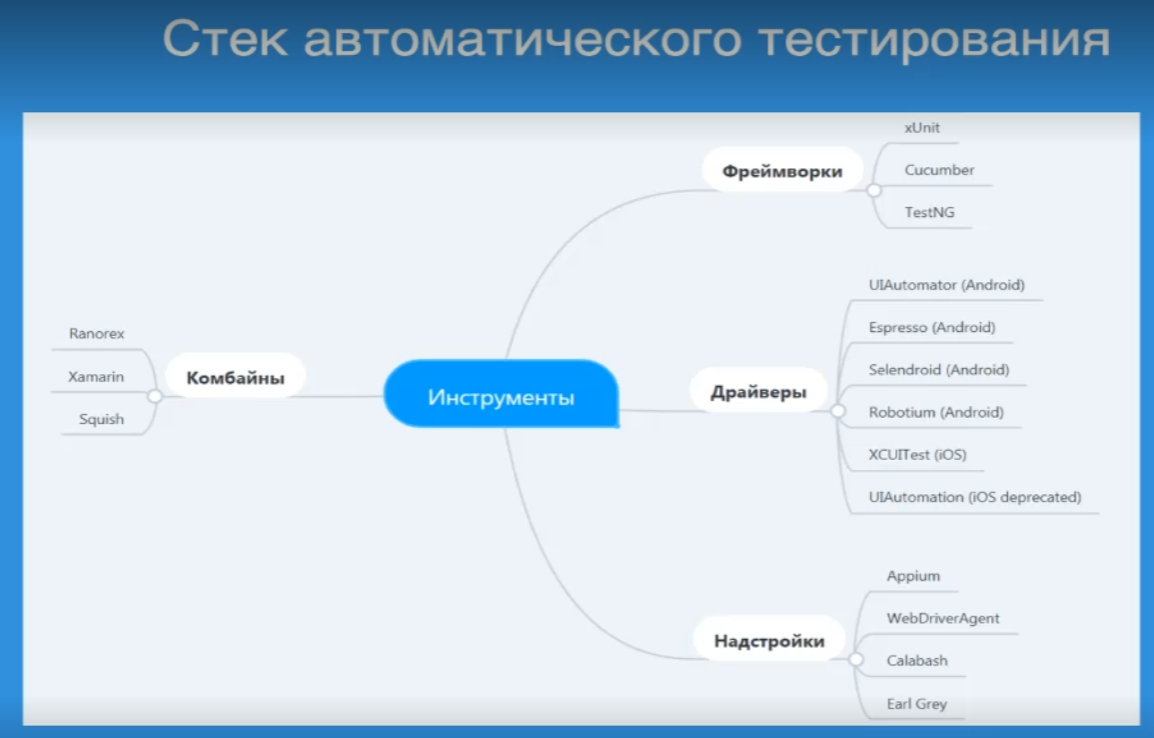
## Инструменты



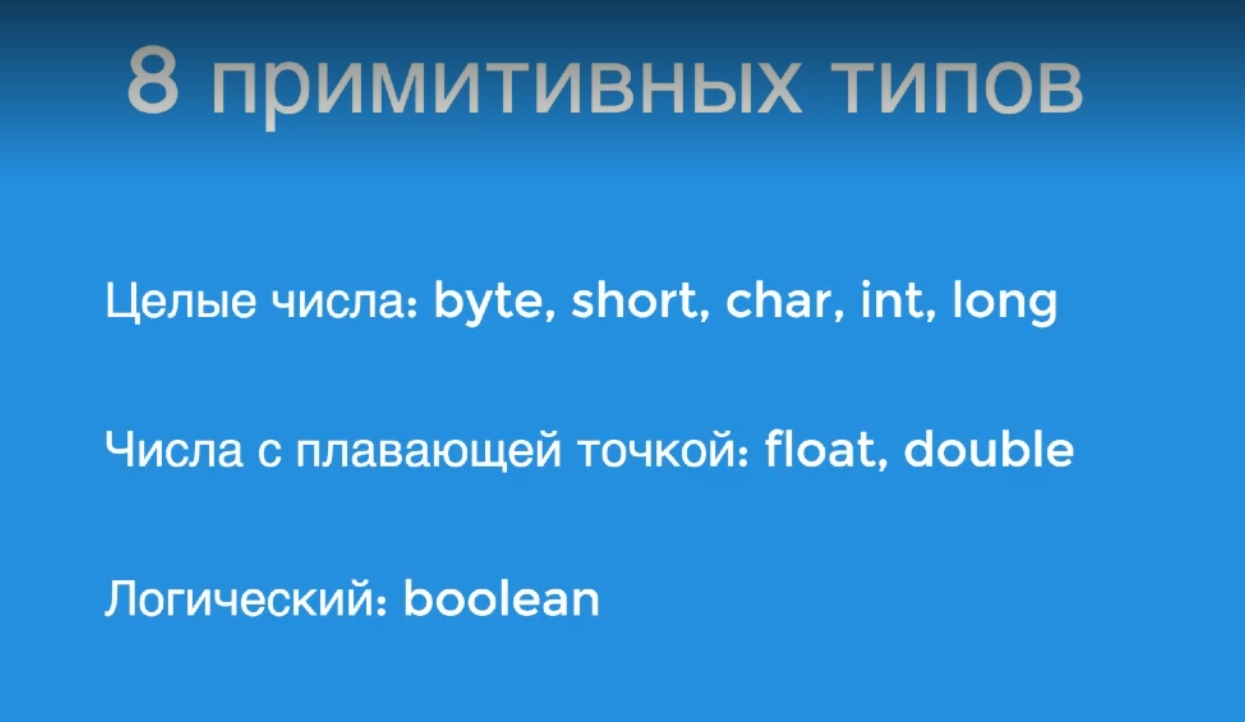




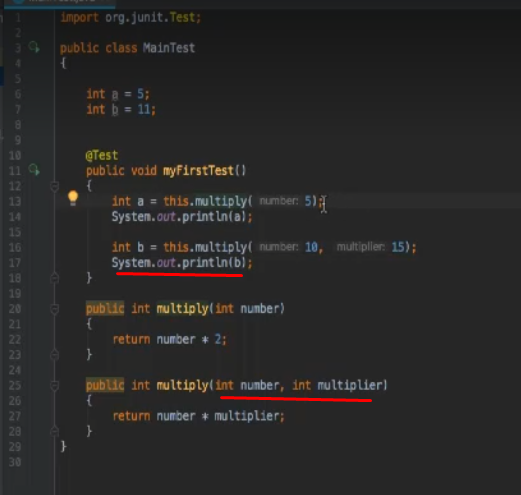
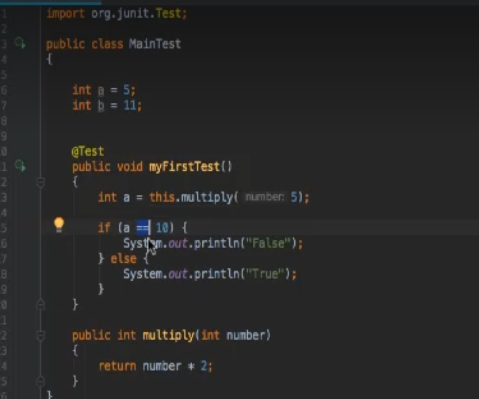




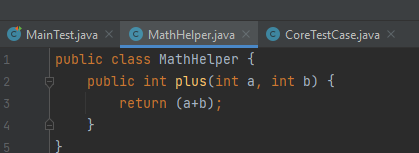
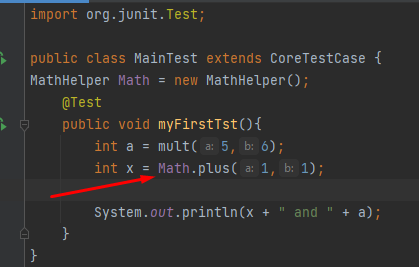
## Java



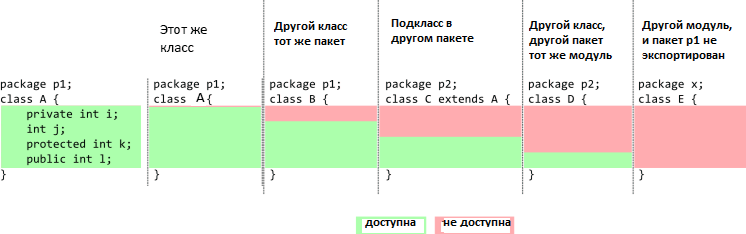
### Функции и this



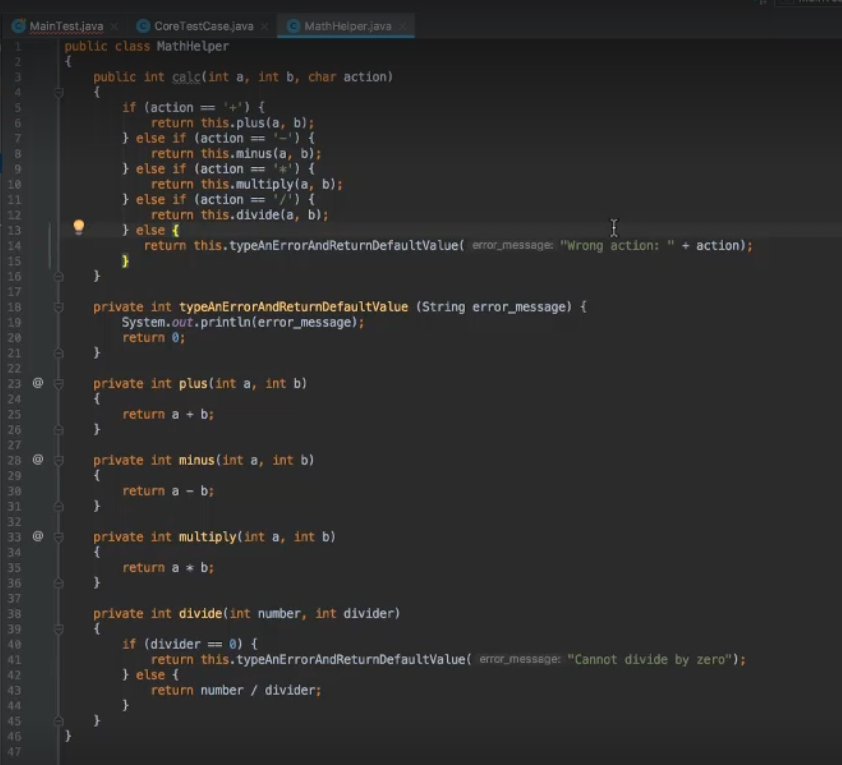
### Родительские классы



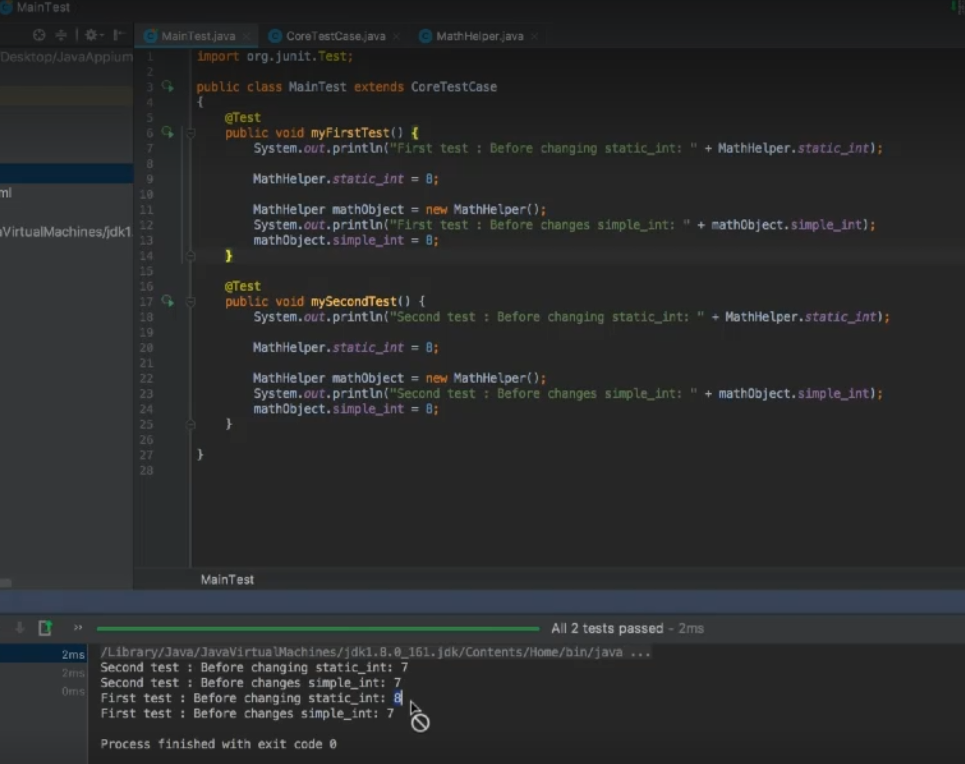
### Модификатор доступа



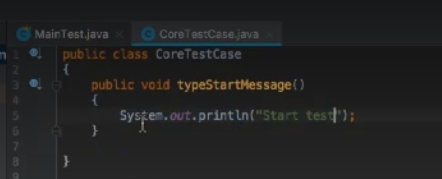
Пример:



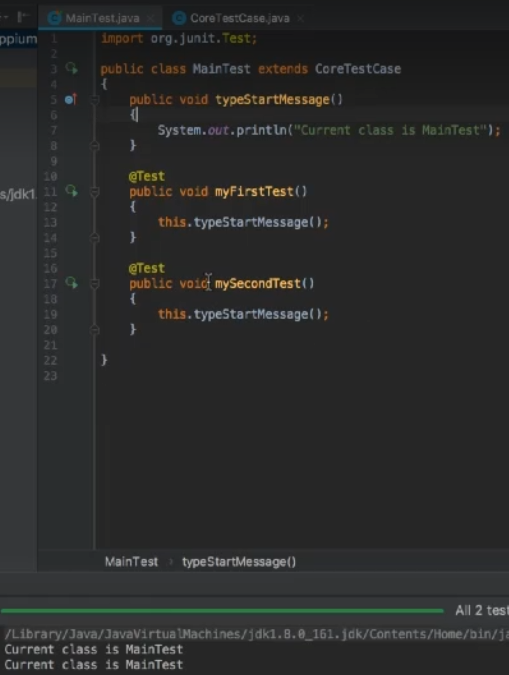
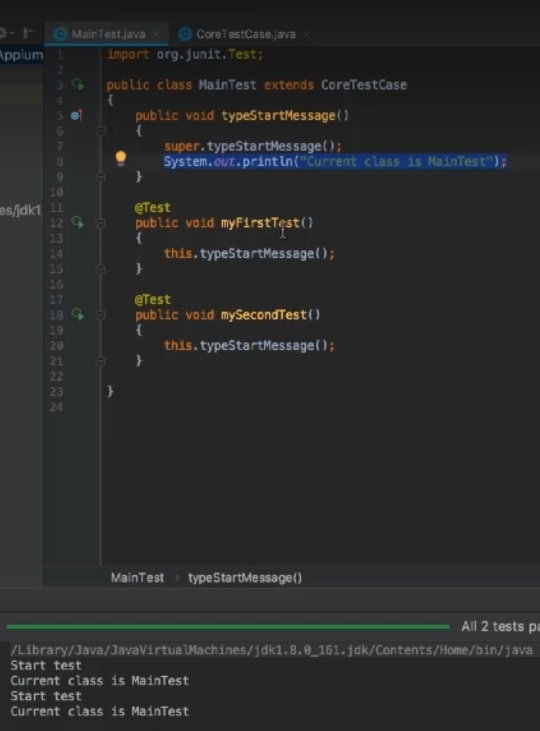
### Статические методы и поля?



### Super



При Super сначала родитель потом текущий. Для переопределения – убрать.



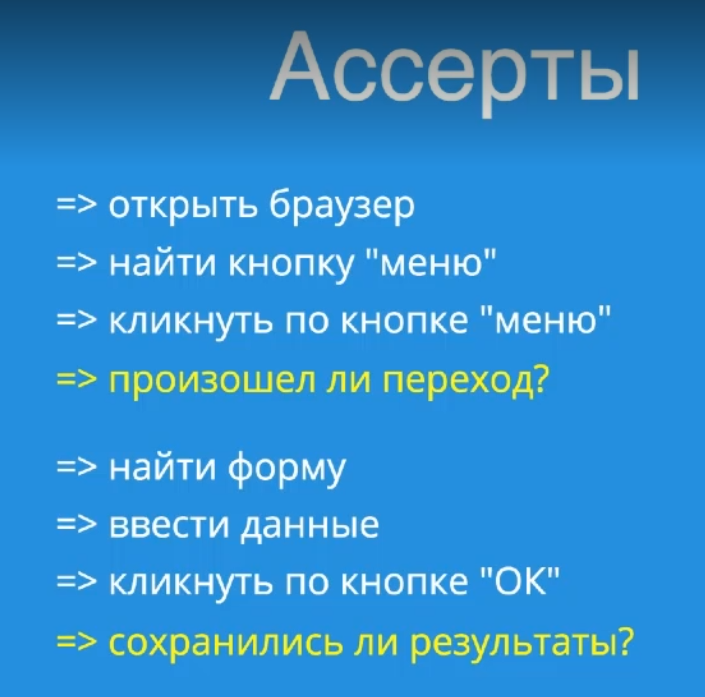
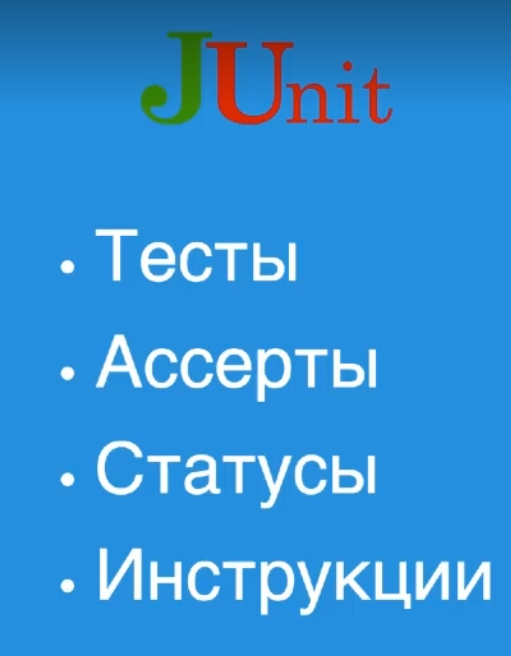
### ООП



### Junit

Ассерт - соответствие значения параметру предустановленного значения

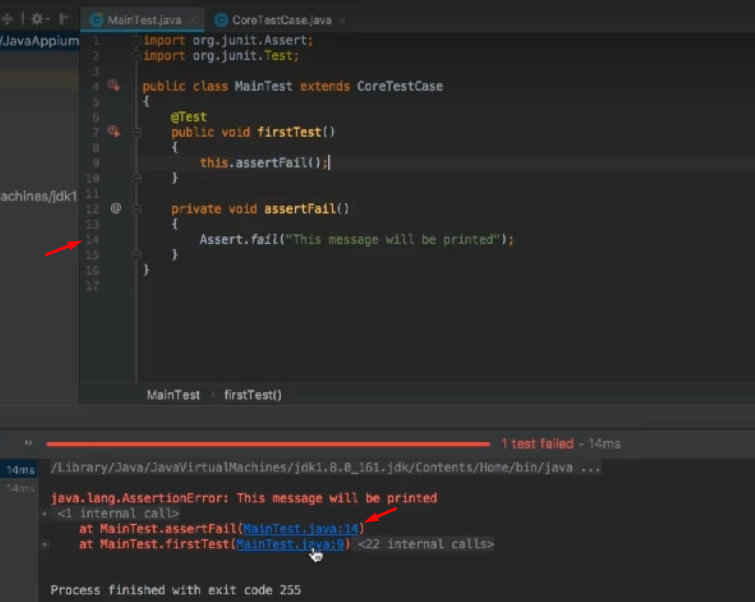
Инструкции – before, after



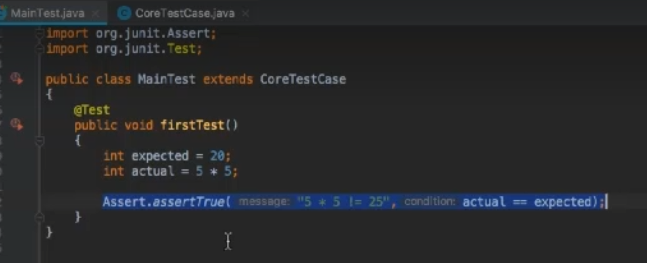
### Test

Alt(option) + enter – выбор

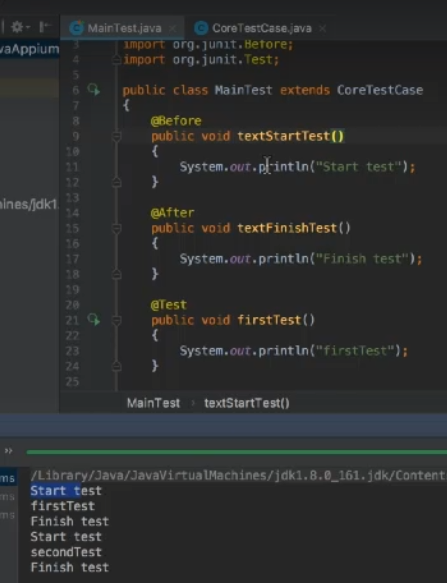
Trace показывает где ошибка



Пример успешного теста (справа как нужно писать):



Before – перед каждым тестом, After - после каждого теста:



# Урок 2

## Запуск эмулятор из Cmd

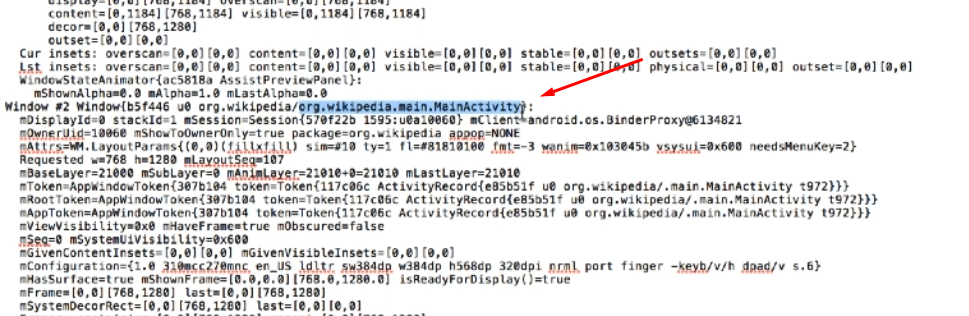
emulator @and80

## Packages and Capabilities

1. Запустить приложение
2. Проверить подключение : adb devices
3. Список всех пакетов всех приложений на устройстве: adb shell pm list packages
4. Найти пакет: adb shell pm list packages | grep "wiki"
5. Выдает активность на открытом экране приложения и записывает в файл:

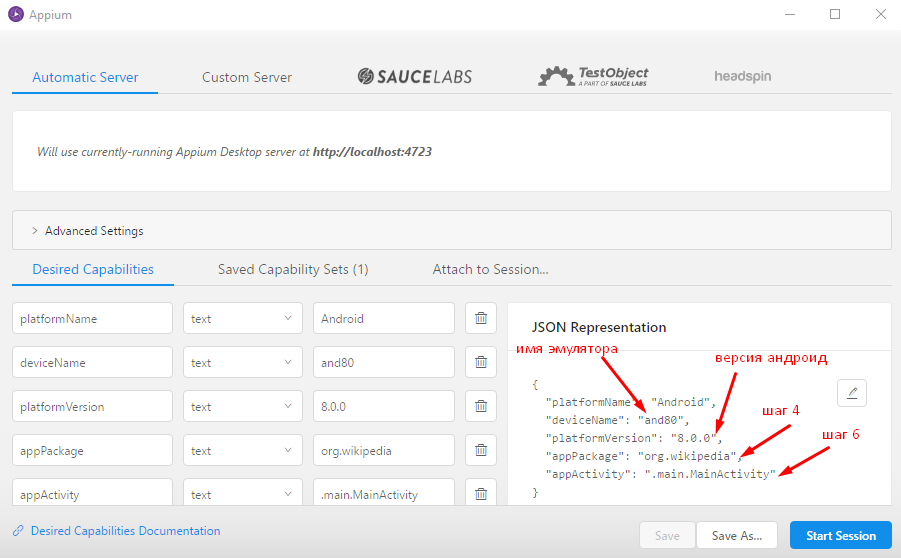
adb shell dumpsys window windows >> Desktop\activity.txt

1. Найти в файле по строке – шаг 4.



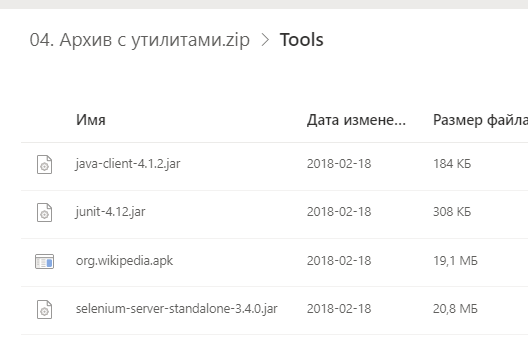
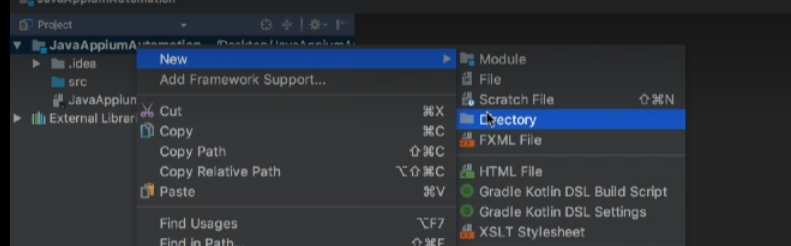
1. Перенести параметры в Appium

## Appium

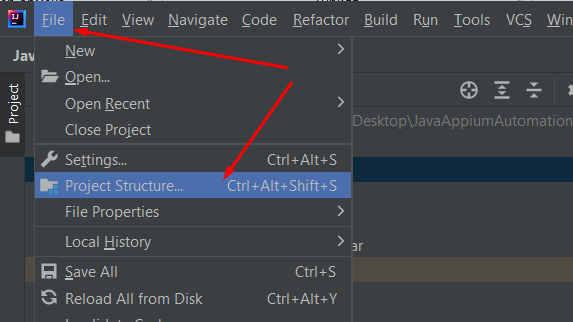
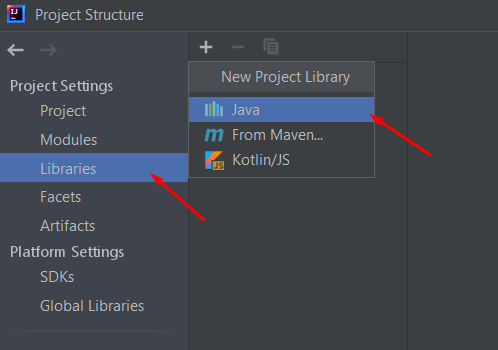


## Создание нового проекта

1. Скачать и импорт в проект в новую папку libs (фреймворки ниже)

1. Перейти в структуру и добавить их

1. Создать еще папку apks – перенести приложение

## Первый тест

В классе FirstTest

import io.appium.java\_client.AppiumDriver;  
import io.appium.java\_client.android.AndroidDriver;  
import org.junit.After;  
import org.junit.Before;  
import org.junit.Test;  
import org.openqa.selenium.remote.DesiredCapabilities;  
  
import java.net.URL;  
  
public class FirstTest {  
 private AppiumDriver driver;  
 @Before  
 public void setUp() throws Exception {  
 DesiredCapabilities capabilities = new DesiredCapabilities();  
 capabilities.setCapability("platformName","Android");  
 capabilities.setCapability("deviceName","and80");  
 capabilities.setCapability("platformVersion","8.0.0");  
 capabilities.setCapability("automationName","Appium");  
 capabilities.setCapability("appPackage","org.wikipedia");  
 capabilities.setCapability("appActivity",".main.MainActivity");  
 capabilities.setCapability("app","C:\\Users\\User\\Desktop\\JavaAppiumAutomation\\apks\\org.wikipedia.apk");  
  
 driver = new AndroidDriver(new URL("http://127.0.0.1:4723/wd/hub"),capabilities);  
 }  
  
 @After  
 public void tearDown ()  
 {  
 driver.quit();  
 }  
 @Test  
 public void firstTest(){  
 System.*out*.println("First run");  
 }  
}

# Урок 3

## Локаторы, клики, ввод в поле поиска

Узнать больше об XPath можно из этой довольно красочной документации: <http://pragmatictestlabs.com/2020/01/28/mastering-xpath-for-selenium-test-automation-engineers/>

(0)

//Поиск Строки поиска и клик:  
 WebElement element = driver.findElementByXPath( "//\*[contains(@text, 'Search Wikipedia')]");  
 element.click();

// Поиск строки ввода и ввод  
 WebElement element\_to\_enter\_search\_line = driver.findElementByXPath( "//\*[contains(@text, 'Search…')]");  
 element\_to\_enter\_search\_line.sendKeys( "Appium");

// - любая вложенность элементов \*- любой элемент и частичное совпадение

## Создадим свой Фреймворк (\*)

// Поиск элемента и ожидание его появления  
private WebElement waitForElementPresentByXpath(String xpath, String error\_message, long timeoutInSeconds)  
{  
WebDriverWait wait = new WebDriverWait(driver, timeoutInSeconds); // время ожидания  
wait.withMessage(error\_message + "\n"); // текст ошибки  
By by = By.*xpath*(xpath); // локатор  
return wait.until (ExpectedConditions.*presenceOfElementLocated*(by) ); // ожидаем элемент  
}

Можно переписать Поиск ввода (1)

// Поиск строки ввода и ввод с Фреймворком  
WebElement element\_to\_enter\_search\_line = waitForElementPresentByXpath(  
 "//\*[contains(@text, 'Search…')]",  
 "Cannot find search input",  
 5);  
  
 //было: driver.findElementByXPath("//\*[contains(@text, 'Search…')]");  
  
element\_to\_enter\_search\_line.sendKeys("Appium");

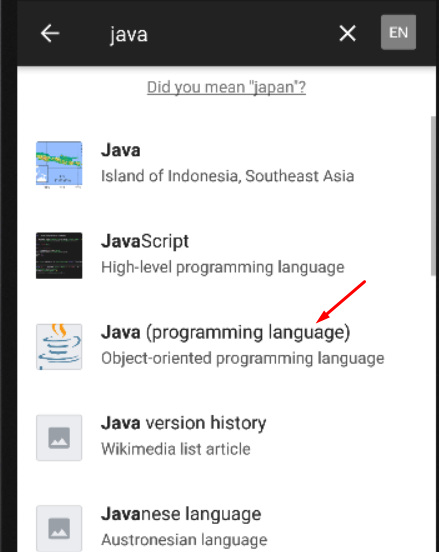
Переопределим Фреймворк (\*\*)

//Переопределим Фреймворк, чтобы не вводить время каждый раз  
private WebElement waitForElementPresentByXpath(String xpath, String error\_message){  
 return waitForElementPresentByXpath ( xpath, error\_message, 5);  
}

Можно переписать Поиск ввода (1\*)

// Поиск строки ввода и ввод с Фреймворком  
 WebElement element\_to\_enter\_search\_line = waitForElementPresentByXpath(  
 "//\*[contains(@text, 'Search…')]",  
 "Cannot find search input");  
 element\_to\_enter\_search\_line.sendKeys("Appium");  
}

## #5 Поиск строки в поиске по id и затем по тексту



1 или 1\* (поиск по java) + то, что ниже…

// Проверка наличия строки в поиске  
// от id искать текст ниже  
waitForElementPresentByXpath("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']",  
 "Cannot find 'Object-oriented programming language' topic searching by Java ",  
 15);

Промежуточный код:

@Test  
 public void firstTest() {  
 //Урок3  
 //Поиск Строки поиска и клик:  
 WebElement element = driver.findElementByXPath("//\*[contains(@text, 'Search Wikipedia')]");  
 element.click();  
  
 // Поиск строки ввода и ввод с Фреймворком  
 WebElement element\_to\_enter\_search\_line = waitForElementPresentByXpath(  
 "//\*[contains(@text, 'Search…')]", //было: driver.findElementByXPath("//\*[contains(@text, 'Search…')]");  
 "Cannot find search input");  
 element\_to\_enter\_search\_line.sendKeys("Java");  
  
 // Проверка наличия строки в поиске  
 // от id искать текст ниже  
 waitForElementPresentByXpath("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']",  
 "Cannot find 'Object-oriented programming language' topic searching by Java ",  
 15);  
 }  
  
 // Фреймворк Поиск элемента и ожадание его появления  
 private WebElement waitForElementPresentByXpath(String xpath, String error\_message, long timeoutInSeconds)  
 {  
 WebDriverWait wait = new WebDriverWait(driver, timeoutInSeconds); // время ожидания  
 wait.withMessage(error\_message + "\n"); // текст ошибки  
 By by = By.*xpath*(xpath); // локатор  
 return wait.until (ExpectedConditions.*presenceOfElementLocated*(by) ); // ожидаем элемент  
 }  
  
 //Переопределим Фреймворк, чтобы не вводить время каждый раз  
 private WebElement waitForElementPresentByXpath(String xpath, String error\_message){  
 return waitForElementPresentByXpath ( xpath, error\_message, 5);  
 }  
}

## #6 Методы Ожидание элемента клик и ввод

//Ожидание элемента и клик  
private WebElement waitForElementAndClick(String xpath, String error\_message, long timeoutInSeconds) {  
 WebElement element = waitForElementPresentByXpath (xpath, error\_message, timeoutInSeconds);  
 element.click();  
 return element;  
}  
  
//Ожидание элемента и ввод  
private WebElement waitForElementAndSendKeys(String xpath, String value, String error\_message, long timeoutInSeconds) {  
 WebElement element = waitForElementPresentByXpath (xpath, error\_message, timeoutInSeconds);  
 element.sendKeys(value);  
 return element;  
}

Стало:

//Поиск и клик  
waitForElementByXpathAndClick("//\*[contains(@text, 'Search Wikipedia')]",  
 "Cannot find Search Wiki input",  
 5);  
  
//Поиск и ввод  
waitForElementByXpathAndSendKeys( "//\*[contains(@text, 'Search…')]",  
 "Java",  
 "Cannot find search input",  
 5);

## #7 – 8 Поиск строки. Отмена поиска. Рефакторинг

@Test  
public void firstTest() {  
 //Поиск и клик  
 waitForElementAndClick(  
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);  
  
 //Поиск и ввод  
 waitForElementAndSendKeys(  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 "Java",  
 "Cannot find search input",  
 5);  
  
 // Проверка наличия строки в поиске  
 // от id искать текст ниже  
 waitForElementPresent(  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']"),  
 "Cannot find 'Object-oriented programming language' topic searching by Java ",  
 15);  
}

@Test  
public void testCancelSearch() { //Тест на отмену поиска. По id  
  
 //Дожидаемся эл-та и кликаем  
 waitForElementAndClick(  
 By.*id*("org.wikipedia:id/search\_container"),  
 "Cannot find Search Wiki input",  
 5  
 );  
  
 //Поиск и клик  
 waitForElementAndClick(  
 By.*id*("org.wikipedia:id/search\_close\_btn"),  
 "Cannot find X to cancel search",  
 5  
 );  
  
 //Проверка отсутсвия эл-та на странице  
 waitForElementNotPresent(  
 By.*id*("org.wikipedia:id/search\_close\_btn"),  
 "X is still on the page",  
 5  
 );  
}

// Фреймворк. Поиск элемента и ожидание его появления  
 private WebElement waitForElementPresent(By by, String error\_message, long timeoutInSeconds)  
 {  
 WebDriverWait wait = new WebDriverWait(driver, timeoutInSeconds); // время ожидания  
 wait.withMessage(error\_message + "\n"); // текст ошибки  
 return wait.until (ExpectedConditions.*presenceOfElementLocated*(by) ); // ожидаем элемент  
 }  
  
 //Переопределим Фреймворк, чтобы не вводить время каждый раз  
 private WebElement waitForElementPresent(By by, String error\_message){  
 return waitForElementPresent(by, error\_message, 5);  
 }  
  
 //Ожидание элемента и клик  
 private WebElement waitForElementAndClick(By by, String error\_message, long timeoutInSeconds) {  
 WebElement element = waitForElementPresent(by, error\_message, timeoutInSeconds);  
 element.click();  
 return element;  
 }  
  
 //Ожидание элемента и ввод  
 private WebElement waitForElementAndSendKeys(By by, String value, String error\_message, long timeoutInSeconds) {  
 WebElement element = waitForElementPresent(by, error\_message, timeoutInSeconds);  
 element.sendKeys(value);  
 return element;  
 }  
  
 //Проверка отсутсвия эл-та на странице по ID  
 private boolean waitForElementNotPresent(By by, String error\_message, long timeoutInSeconds){  
 WebDriverWait wait = new WebDriverWait(driver, timeoutInSeconds);  
 wait.withMessage(error\_message + "\n");  
 return wait.until(ExpectedConditions.*invisibilityOfElementLocated*(by));  
 }

## Очистка элемента

//Очистка элемента  
private WebElement waitАoкElementAndClear(By by, String error\_message, long timeoutInSeconds){  
WebElement element = waitForElementPresent(by, error\_message, timeoutInSeconds);  
element.clear();  
return element;  
}

## Сравнение заголовков

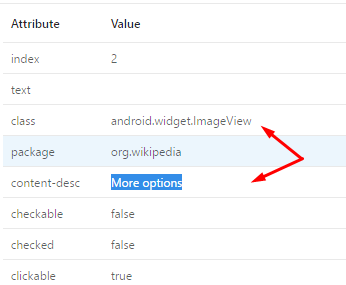
@Test  
//Падение тестов - сравнение заголовков  
public void testCompareArticleTitle() {  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);  
  
 waitForElementAndSendKeys( //Поиск и ввод  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 "Java",  
 "Cannot find search input",  
 5);  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']"),  
 "Cannot find Search Wiki input",  
 25);  
  
 // ожидаем элемент и сохраняем  
 WebElement title\_element = waitForElementPresent(  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "Cannot find article title",  
 15  
 );  
  
 // запишем текст из заголовка по атрибуту text  
 String article\_title = title\_element.getAttribute("text");  
  
 //Сравнение заголовков  
 Assert.*assertEquals*(  
 "We see unexpected title!",  
 "Java (programming language)",  
 article\_title  
 );  
}

## Просмотр элементов uit как апиум

<https://www.guru99.com/uiautomatorviewer-tutorial.html>

# Поиск элементов

## Поиск по классу и названию



By.*xpath*("//\*android.widget.ImageView[@content-desk='More options']"),

## Поиск по тексту

By.*xpath*("//\*[@text='Add to reading list']"),

## Поиск по тексту из переменной

By.*xpath*("//\*[@text='" + name\_of\_folder + "']"),

## Поиск по ID



By.*id*("org.wikipedia:id/onboarding\_button"),

## Родительский и дочерний

(/\* - на 1 элемент ниже)

String search\_result\_locator = "//\*[@resource-id='org.wikipedia:id/search\_results\_list']/\*[@resource-id='org.wikipedia:id/page\_list\_item\_container']";

# Методы

## Первый тест

В классе FirstTest

import io.appium.java\_client.AppiumDriver;  
import io.appium.java\_client.android.AndroidDriver;  
import org.junit.After;  
import org.junit.Before;  
import org.junit.Test;  
import org.openqa.selenium.remote.DesiredCapabilities;  
  
import java.net.URL;  
  
public class FirstTest {  
 private AppiumDriver driver;  
 @Before  
 public void setUp() throws Exception {  
 DesiredCapabilities capabilities = new DesiredCapabilities();  
 capabilities.setCapability("platformName","Android");  
 capabilities.setCapability("deviceName","and80");  
 capabilities.setCapability("platformVersion","8.0.0");  
 capabilities.setCapability("automationName","Appium");  
 capabilities.setCapability("appPackage","org.wikipedia");  
 capabilities.setCapability("appActivity",".main.MainActivity");  
 capabilities.setCapability("app","C:\\Users\\User\\Desktop\\JavaAppiumAutomation\\apks\\org.wikipedia.apk");  
  
 driver = new AndroidDriver(new URL("http://127.0.0.1:4723/wd/hub"),capabilities);  
 }  
  
 @After  
 public void tearDown ()  
 {  
 driver.quit();  
 }  
 @Test  
 public void firstTest(){  
 System.*out*.println("First run");  
 }  
}

## Ожидание элемента

private WebElement waitForElementPresent(By by, String error\_message, long timeoutInSeconds)  
{  
WebDriverWait wait = new WebDriverWait(driver, timeoutInSeconds); // время ожидания  
wait.withMessage(error\_message + "\n"); // текст ошибки  
return wait.until (ExpectedConditions.*presenceOfElementLocated*(by) ); // ожидаем элемент  
}  
  
//Переопределим Фреймворк, чтобы не вводить время каждый раз  
private WebElement waitForElementPresent(By by, String error\_message){  
 return waitForElementPresent(by, error\_message, 5);  
}

Применение

waitForElementPresent(  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "Cannot find article title",  
 35  
);

waitForElementPresent(  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']"),  
 "Cannot find 'Object-oriented programming language' topic searching by Java ",  
 15);

## Отсутствие элемента

//Проверка отсутствия эл-та на странице

private boolean waitForElementNotPresent(By by, String error\_message, long timeoutInSeconds){  
WebDriverWait wait = new WebDriverWait(driver, timeoutInSeconds);  
wait.withMessage(error\_message + "\n");  
return wait.until(ExpectedConditions.*invisibilityOfElementLocated*(by));  
}

Применение

waitForElementNotPresent(  
 By.*id*("org.wikipedia:id/search\_results\_list"),  
 "Search results area is still present",  
 5  
 );  
}

## Ожидание и клик

//Ожидание элемента и клик  
private WebElement waitForElementAndClick(By by, String error\_message, long timeoutInSeconds) {  
 WebElement element = waitForElementPresent(by, error\_message, timeoutInSeconds);  
 element.click();  
 return element;  
}

Применение

waitForElementAndClick(   
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);

waitForElementAndClick( //Поиск и клик поиск среди заголовков по тексту  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_title'][@text = 'Appium']"),  
 "Cannot find Search Appium input",  
 25);

## Ожидание и ввод

private WebElement waitForElementAndSendKeys(By by, String value, String error\_message, long timeoutInSeconds) {  
 WebElement element = waitForElementPresent(by, error\_message, timeoutInSeconds);  
 element.sendKeys(value);  
 return element;  
}

Применение

waitForElementAndSendKeys(  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 "Java",  
 "Cannot find search input",  
 5);

## Очистка элемента

private WebElement waitForElementAndClear(By by, String error\_message, long timeoutInSeconds){  
WebElement element = waitForElementPresent(by, error\_message, timeoutInSeconds);  
element.clear();  
return element;  
}

Применение

// Очищаем введенное значение в поле вода по переданному элементу  
waitForElementAndClear(  
 By.*id*("org.wikipedia:id/search\_src\_text"), // Поле ввода  
 "Cannot find search field",  
 5  
);

## Проверка текста

//Проверка наличие ожидаемого текста у элемента  
private WebElement assertElementHasText (By by, String expected\_text, String error\_message, long timeoutInSeconds){  
WebElement element = waitForElementPresent(by, error\_message, timeoutInSeconds);  
String search\_title = element.getAttribute("text");  
Assert.*assertEquals*(error\_message, expected\_text, search\_title);  
return element;

}

Применение

public void testElementHasText() {  
  
 assertElementHasText(  
 By.*xpath*("//\*[@resource-id='org.wikipedia:id/search\_container']//\*[@text='Search Wikipedia']"),  
 "Search Wikipedia",  
 "Couldn't find 'Search Wikipedia' text in Search field",  
 10)

;}

## Поиск числа элементов

//Число элементов на сранице  
private int getAmountOfElements(By by) {  
 List elements = driver.findElements(by);  
 return elements.size();}

Применение

// число эементов  
 int amount\_of\_search\_results = getAmountOfElements(  
 By.*xpath*(search\_result\_locator)  
 );

}

## Проверка что элементов больше 0

// Проверка что элементов больше 0  
Assert.*assertTrue*("We found too few results",  
 amount\_of\_search\_results > 0);

## Проверка что не нашлось ни одной строки

// Проверка что не нашлось ни одной строки  
private void assertElementNotPresent(By by, String error\_message) {  
 int amount\_of\_elements = getAmountOfElements(by); //число элементов

if (amount\_of\_elements > 0) {  
 String default\_message = "An element '" + by.toString() + "' supposed to be not present";

throw new AssertionError(default\_message + " " + error\_message);

}

}

Применение

// Проверка что не нашлось ни одной строки  
assertElementNotPresent (  
 By.*xpath*(search\_result\_locator),  
 "We've found some results by request " + search\_line );

## Получить заголовок статьи

// Получить заголовок статьи  
private String waitForElementAndGetAttribute(By by, String attribute, String error\_message, long timeInSeconds){  
 WebElement element = waitForElementPresent(by, error\_message, timeInSeconds);  
 return element.getAttribute(attribute);  
}

Применение

// Запишем атрибут (text) в строку  
 String title\_before\_rotation = waitForElementAndGetAttribute(  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "text",  
 "Cannot find title of article",  
 15  
 );  
}

## Поворот экрана

// Поворот экрана  
 driver.rotate(ScreenOrientation.*LANDSCAPE*);

## Сравнение ОР и ФР

// сравнение результатов  
Assert.*assertEquals*(  
 "Article title have been changed after screen rotation",  
 title\_after\_rotation,  
 title\_after\_rotation  
);

## Свернуть и развернуть МП

driver.runAppInBackground(5);

# Урок 4

## Свайп

@Test  
  
//Свайп  
public void testSwipeArticle() {  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);  
  
 waitForElementAndSendKeys( //Поиск и ввод  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 "Java",  
 "Cannot find search input",  
 5);  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']"),  
 "Cannot find Search Wiki input",  
 25);  
  
 waitForElementPresent( // ожидаем элемент  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "Cannot find article title",  
 15  
 );  
  
 //вызов свайпа  
 swipeUp(2000);  
  
}

Метод:

// Свайп вверх  
protected void swipeUp(int timeOfSwipe) { // передаем время между нажатиями  
 TouchAction action = new TouchAction(driver); //импорт из appium  
 //для создания отн-ых координат  
 Dimension size = driver.manage().window().getSize(); // получаем параметры девайса  
  
 int x = size.width / 2;//нач. переменная по Х = конеч. переменная по Х  
 int start\_y = (int) (size.height \* 0.8); //нач. переменная по У - на 80% экрана т.е. снизу  
 int end\_y = (int) (size.height \* 0.2); //конеч. переменная по У  
  
 action.press(x, start\_y).waitAction(timeOfSwipe).moveTo(x, end\_y).release().perform();  
}

## Свайп до футера

(конца страницы)

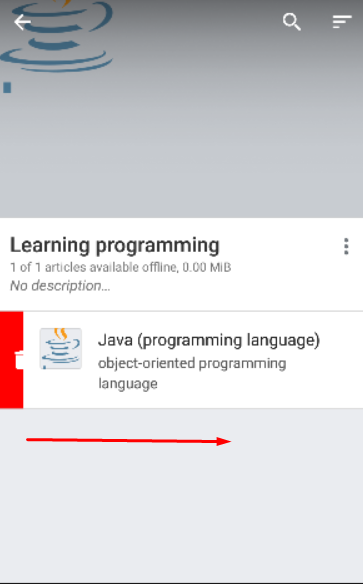
1. Проверяем есть ли элемент
2. Ели нет то свайпаем и снова проверяем
3. Находим эл-т и по нему понимаем, что конец

@Test  
//Свайп до конца страницы  
public void testSwipeArticle() {  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);  
  
 waitForElementAndSendKeys( //Поиск и ввод для Appium  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 "Appium",  
 "Cannot find search input",  
 5);  
  
 waitForElementAndClick( //Поиск и клик поиск среди заголовков по тексту  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_title'][@text = 'Appium']"),  
 "Cannot find Search Appium input",  
 25);  
  
 waitForElementPresent( // ожидаем элемент  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "Cannot find article title",  
 35  
 );  
  
 //Свайп до футера (конца страницы)с ограничением до 6 свайпов  
 swipeUpToFindElement(  
 By.*xpath*("//\*[@text = 'View page in browser']"),  
 "Cannot find article title",  
 6  
 );  
}

Метод

//Свайп пока не найдем эл-т с ограничением по свайпам  
protected void swipeUpToFindElement(By by, String error\_message, int max\_swipes){  
 // driver.findElements(by); - находит все эл-ты что есть  
 // driver.findElements(by).size(); - число найденных эл-ов  
 int already\_swipes = 0;  
 while (driver.findElements(by).size() == 0){  
  
 // проверяем что эл-та все еще нет  
 if (already\_swipes > max\_swipes){  
 waitForElementPresent(by, "Cannot find element by swiping up. \n" + error\_message, 0);  
 return;  
 }  
 swipeUpQuick();  
 ++already\_swipes;  
 }  
}

## Свай влево



// Свай влево  
protected void swipeElementToLeft(By by, String error\_message) {  
 WebElement element = waitForElementPresent(  
 by,  
 error\_message,  
 10);  
 int left\_x = element.getLocation().getX(); // получаем самую левую точку  
 int right\_x = element.getSize().getWidth(); // прибавляем ширину элемента  
  
 // Получаем середину элемента  
 int upper\_y = element.getLocation().getY();  
 int lower\_y = upper\_y + element.getSize().getHeight();  
 int middle\_y = (upper\_y + lower\_y) / 2;  
  
 TouchAction action = new TouchAction(driver); // инициализируем драйвер  
 action  
 .press(right\_x, middle\_y)  
 .waitAction(150)  
 .moveTo(left\_x, middle\_y)  
 .release()  
 .perform();  
}

## #3 Добавление в избранное и удаление

//Сохранить статью в избранное и удалить  
@Test  
public void saveFirstArticleToMyList(){  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);  
  
 waitForElementAndSendKeys( //Поиск и ввод  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 "Java",  
 "Cannot find search input",  
 5);  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']"),  
 "Cannot find Search Wiki input",  
 25);  
  
 waitForElementPresent( // ожидаем элемент  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "Cannot find article title",  
 15  
 );  
  
 // Переходим в меню (...) статьи  
 waitForElementAndClick(  
 By.*xpath*("//android.widget.ImageView[@content-desc='More options']"), //android.widget.ImageView[@content-desc='More options'  
 "Cannot find button to open article options",  
 15);  
  
 // Добавляем в чтение  
 waitForElementAndClick(  
 //By.xpath("//\*[@text='Add to reading list']"),  
 By.*xpath*("/hierarchy/android.widget.FrameLayout/android.widget.FrameLayout/android.widget.ListView/android.widget.LinearLayout[3]/android.widget.RelativeLayout/android.widget.TextView"),  
 "Cannot find option to add article to reading list",  
 15);  
  
 // Кликаем кнопку добавить - подверждение  
 waitForElementAndClick(  
 By.*id*("org.wikipedia:id/onboarding\_button"),  
 "Cannot find 'Got it' tip overlay",  
 5);  
  
 // После всплывает окно с введением названия списка для чтения  
  
 // Очищаем дефолтное значение  
 waitForElementAndClear(  
 By.*id*("org.wikipedia:id/text\_input"),  
 "Cannot find input to set name of articles folder",  
 5 );  
  
 // Вводим новое  
 String name\_of\_folder = "Learning programming"; // название папки с избранным  
 waitForElementAndSendKeys(  
 By.*id*("org.wikipedia:id/text\_input"),  
 name\_of\_folder,  
 "Cannot put text into articles folder input",  
 5 );  
  
 // Подтверждаем  
 waitForElementAndClick(  
 By.*xpath*("//\*[@text='OK']"),  
 "Cannot press OK button",  
 5  
 );  
  
 // Закрываем статю  
 waitForElementAndClick(  
 By.*xpath*("//android.widget.ImageButton[@content-desc='Navigate up']"),  
 "Cannot close article",  
 5  
 );  
  
 // Добавляем в избранное  
 waitForElementAndClick(By.*xpath*("//android.widget.FrameLayout[@content-desc='My lists']"),  
 "Cannot find navigation button to My list",  
 5  
 );  
  
 // Находим ее в избранном  
 waitForElementAndClick(  
 By.*xpath*("//\*[@text='Learning programming']"),  
 //By.xpath("//\*[@text='" + name\_of\_folder + "']"), // name\_of\_folder - переменная  
 "Cannot find created folder",  
 5  
 );  
  
 // Свайпаем влево для удаления  
 swipeElementToLeft(  
 By.*xpath*("//\*[@text='Java (programming language)']"),  
 "Cannot find saved article"  
 );  
  
 // Проверяем что статьи нет  
 waitForElementNotPresent(  
 By.*xpath*("//\*[@text='Java (programming language)']"),  
 "Cannot delete saved article",  
 10);  
}

## #5 Assert True

– проверка что условие соблюдается

// Проверка что элементов больше 0  
Assert.*assertTrue*("We found too few results",  
 amount\_of\_search\_results > 0);

## # Ротация (поворот)

// Проверка что заголовок не меняется при повороте экрана  
@Test  
public void testChangeScreenOrientationOnSearchResults(){  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);  
  
 String search\_line = "Java"; // что ищем  
 waitForElementAndSendKeys( //Поиск и ввод  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 search\_line,  
 "Cannot find search input",  
 5);  
  
 waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[@resource-id = 'org.wikipedia:id/page\_list\_item\_container']//\*[@text = 'Object-oriented programming language']"),  
 "Cannot find 'Object-oriented programming language' topic searching by " + search\_line,  
 25);  
  
 // Запишем атрибут (название строки статьи) в строку  
 String title\_before\_rotation = waitForElementAndGetAttribute(  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "text",  
 "Cannot find title of article",  
 15  
 );  
  
 // Поворот экрана  
 driver.rotate(ScreenOrientation.*LANDSCAPE*);  
  
 // Запишем атрибут (название строки статьи) в строку  
 String title\_after\_rotation = waitForElementAndGetAttribute(  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "text",  
 "Cannot find title of article",  
 15  
 );  
  
 // сравнение результатов  
 Assert.*assertEquals*(  
 "Article title have been changed after screen rotation",  
 title\_after\_rotation,  
 title\_after\_rotation  
 );  
  
 //Вернем ориентацию  
 driver.rotate(ScreenOrientation.*PORTRAIT*);  
  
 // Запишем атрибут (название строки статьи) в строку  
 String title\_after2\_rotation = waitForElementAndGetAttribute(  
 By.*id*("org.wikipedia:id/view\_page\_title\_text"),  
 "text",  
 "Cannot find title of article",  
 15  
 );  
  
 // сравнение результатов  
 Assert.*assertEquals*(  
 "Article title have been changed after screen rotation",  
 title\_before\_rotation,  
 title\_after2\_rotation  
 );  
}

## ДЗ старое

//\*\*\*  
  
 // EX 1  
 //Функция, которая проверяет наличие ожидаемого текста у элемента - assertElementHasText.  
 // На вход эта функция должна принимать локатор элемент, ожидаемый текст и текст ошибки,  
 // который будет написан в случае, если элемент по этому локатору не содержит текст, который мы ожидаем.  
 //  
 //И тест, который проверяет, что поле ввода для поиска статьи содержит текст  
 //(в разных версиях приложения это могут быть тексты "Search..." или "Search Wikipedia",  
 // правильным вариантом следует считать тот, которые есть сейчас).  
 @Test  
 public void testElementHasText() {  
  
 MainPageObject.assertElementHasText(  
 By.*xpath*("//\*[@resource-id='org.wikipedia:id/search\_container']//\*[@text='Search Wikipedia']"),  
 "Search Wikipedia",  
 "Couldn't find 'Search Wikipedia' text in Search field",  
 10  
 );  
 }  
  
 // EX 2  
 //Написать тест, который: Ищет какое-то слово  
 //Убеждается, что найдено несколько статей  
 //Отменяет поиск  
 //Убеждается, что результат поиска пропал  
 @Test  
 public void SearchForManySearchResultsAndCancel() {  
  
 MainPageObject.waitForElementAndClick( //Поиск и клик  
 By.*xpath*("//\*[contains(@text, 'Search Wikipedia')]"),  
 "Cannot find Search Wiki input",  
 5);  
  
 MainPageObject.waitForElementAndSendKeys( //Поиск и ввод  
 By.*xpath*("//\*[contains(@text, 'Search…')]"),  
 "Java",  
 "Cannot find search input",  
 5);  
  
 MainPageObject.waitForElementPresent( // ожидаем элемент и сохраняем  
 By.*id*("org.wikipedia:id/search\_results\_list"),  
 "Couldn't Search Results area",  
 15);  
  
 // Сохраняем список найднных заголовков у полей  
 By mySelector = By.*xpath*("//\*[@resource-id='org.wikipedia:id/page\_list\_item\_container']//\*[@resource-id='org.wikipedia:id/page\_list\_item\_title']");  
  
 List<WebElement> myElements = driver.findElements(mySelector);  
 System.*out*.println("Number of elements:" + myElements.size());  
 *assertTrue*(  
 "Several articles were not found in Search Results area",  
 myElements.size() >= 2);  
  
 MainPageObject.waitForElementAndClick( //Поиск и клик по Х  
 By.*id*("org.wikipedia:id/search\_close\_btn"),  
 "Couldn't find 'X' button to cancel search",  
 5);  
  
 MainPageObject.waitForElementNotPresent(  
 By.*id*("org.wikipedia:id/search\_results\_list"),  
 "Search results area is still present",  
 5  
 );  
}  
  
 //Тест, который делает поиск по какому-то слову. Например, JAVA. Затем убеждается, что в каждом результате поиска есть это слово.  
 //Работать только с теми результатами поиска, который видны сразу, без прокрутки.  
  
 @Test  
 public void ValidateSearchByJavaWord() {  
 MainPageObject.waitForElementAndClick(  
 By.*id*("org.wikipedia:id/search\_container"),  
 "Couldn't find Search field",  
 5  
 );  
  
 MainPageObject.waitForElementAndSendKeys(  
 By.*id*("org.wikipedia:id/search\_container"),  
 "Java",  
 "Couldn't find Search field",  
 5  
 );  
  
 WebElement search\_area = MainPageObject.waitForElementPresent(  
 By.*id*("org.wikipedia:id/search\_results\_list"),  
 "Couldn't Search Results area",  
 15  
 );  
 By mySelector = By.*xpath*("//\*[@resource-id='org.wikipedia:id/page\_list\_item\_container']//\*[@resource-id='org.wikipedia:id/page\_list\_item\_title']");  
  
 List<WebElement> myElements = driver.findElements(mySelector);  
 System.*out*.println("Number of elements:" +myElements.size());  
 int count = 0;  
 for(WebElement e : myElements) {  
 System.*out*.println(e.getText());  
 String string = e.getText();  
 String substring = "Java";  
 if (string.contains(substring))  
 {  
 count = count+1;  
 }  
 }  
 *assertTrue*("There are no 'Java' word in at least one search result",  
 count == 3);  
 }  
  
//\*\*\*