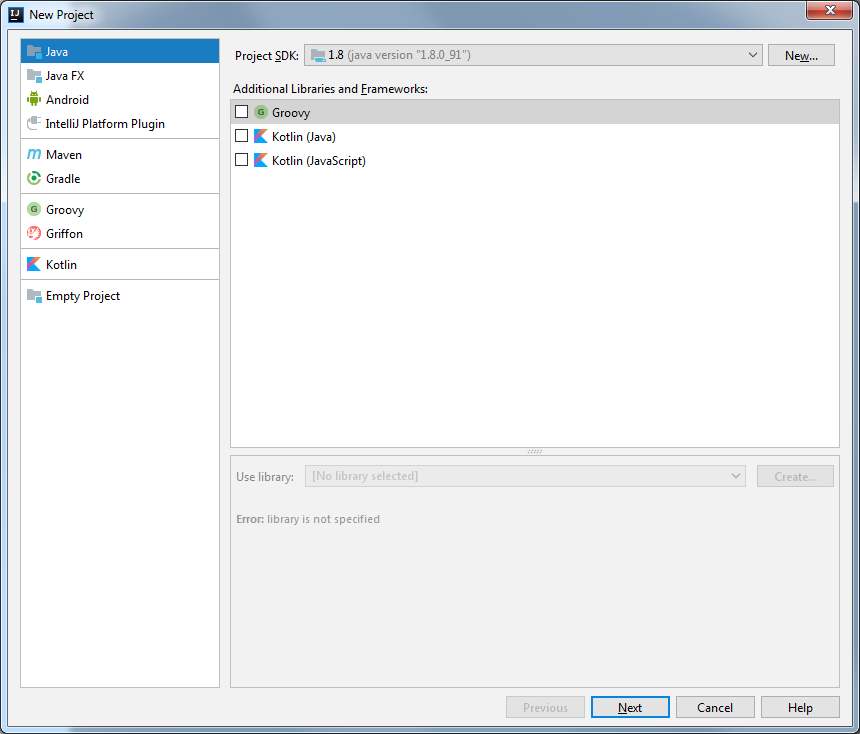
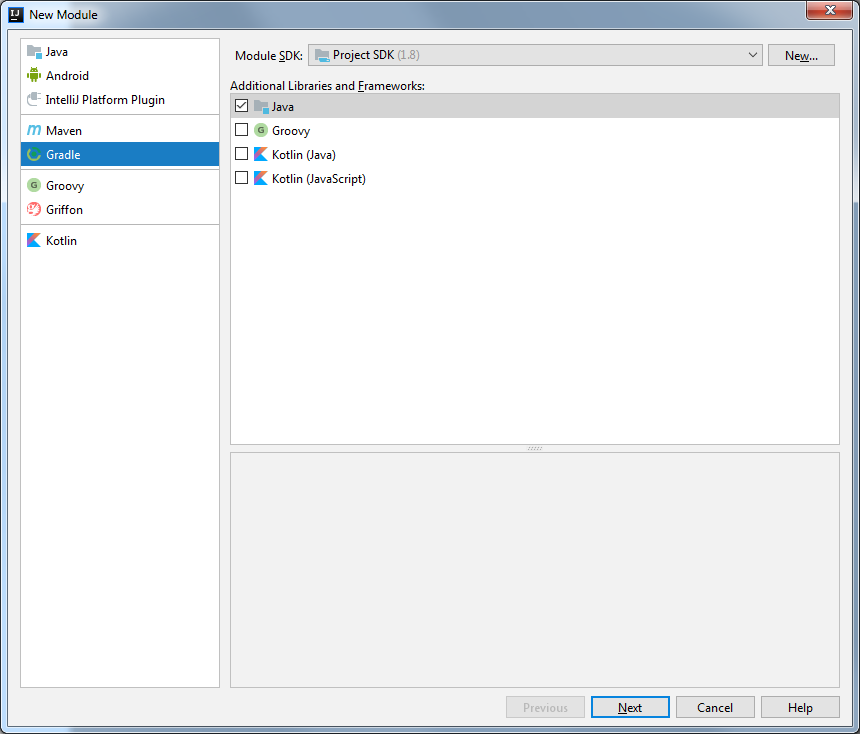
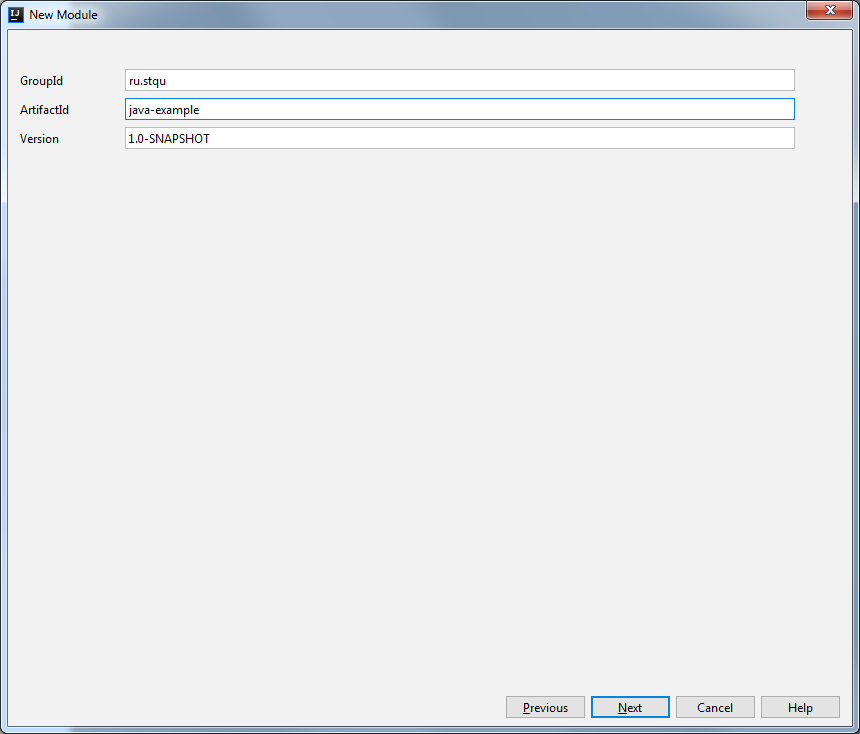
git init – инициализация папки для гита. Создание гит репозитория на локальной машине

Создать новый проект

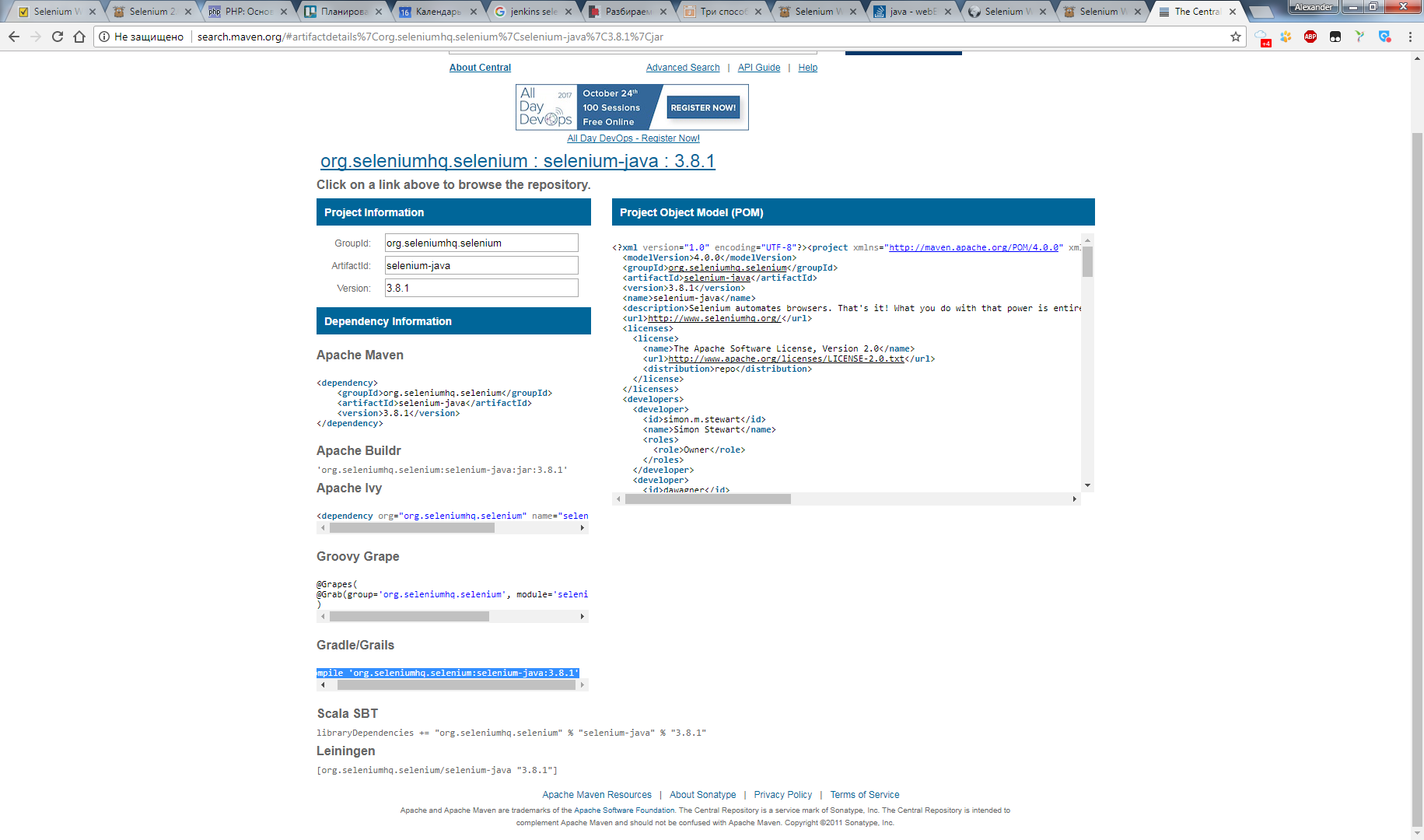


File-new-module – Gradle (это сборщик, он умеет сам подгружать библиотеки, если правильно указать зависимости)

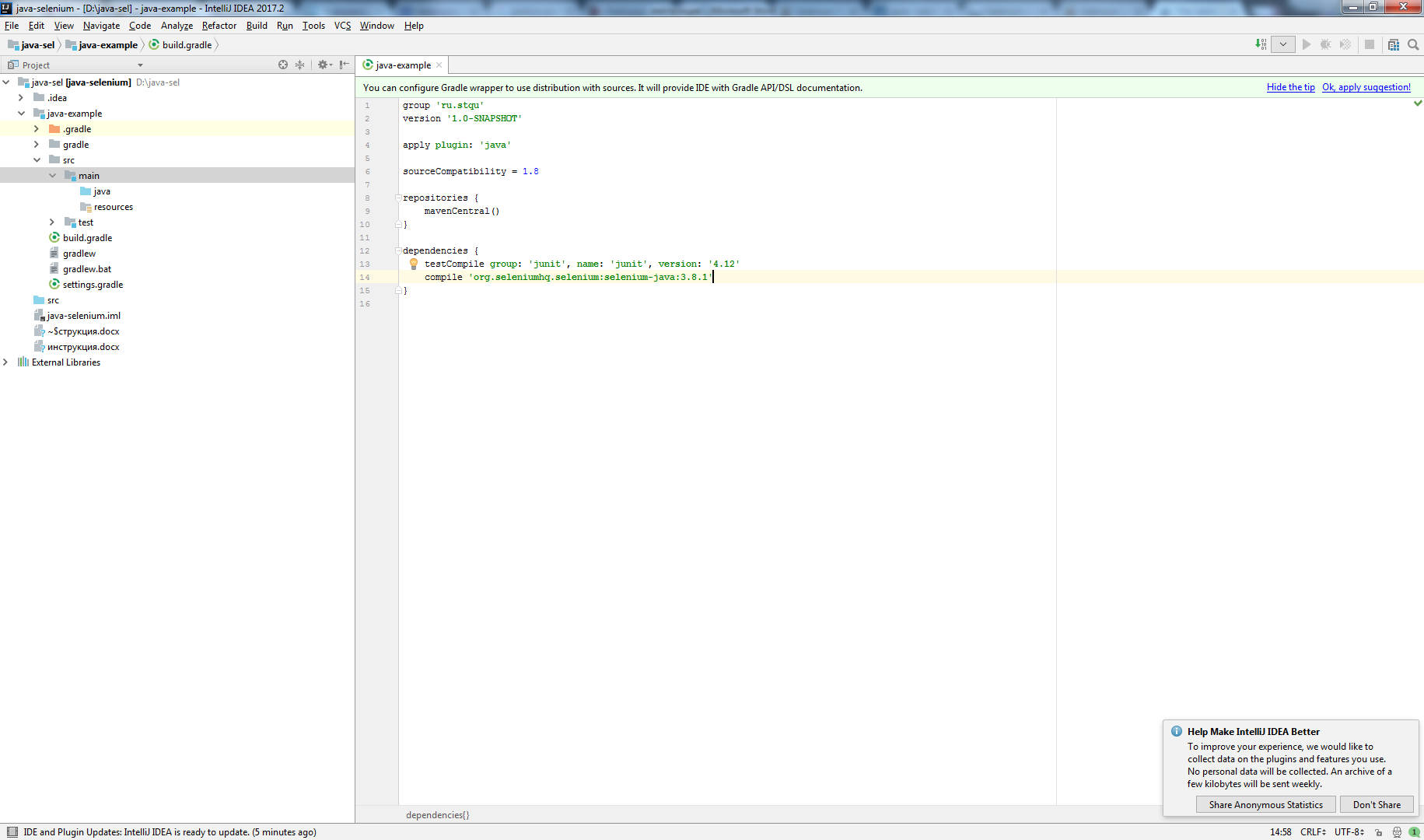




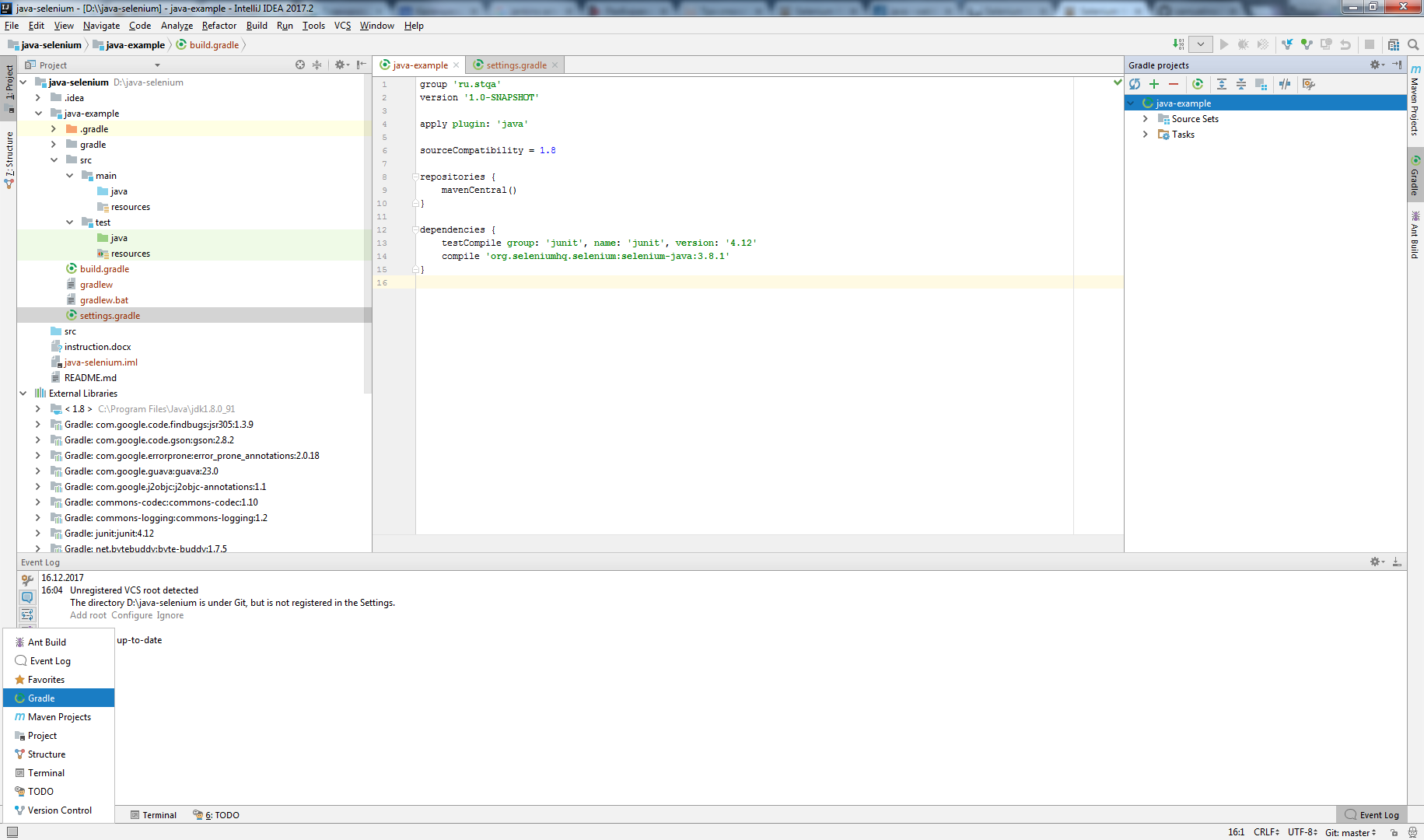
Зайти на сайт <http://search.maven.org> и найти selenium-java, добавить текст из пункта Gradle/Grails

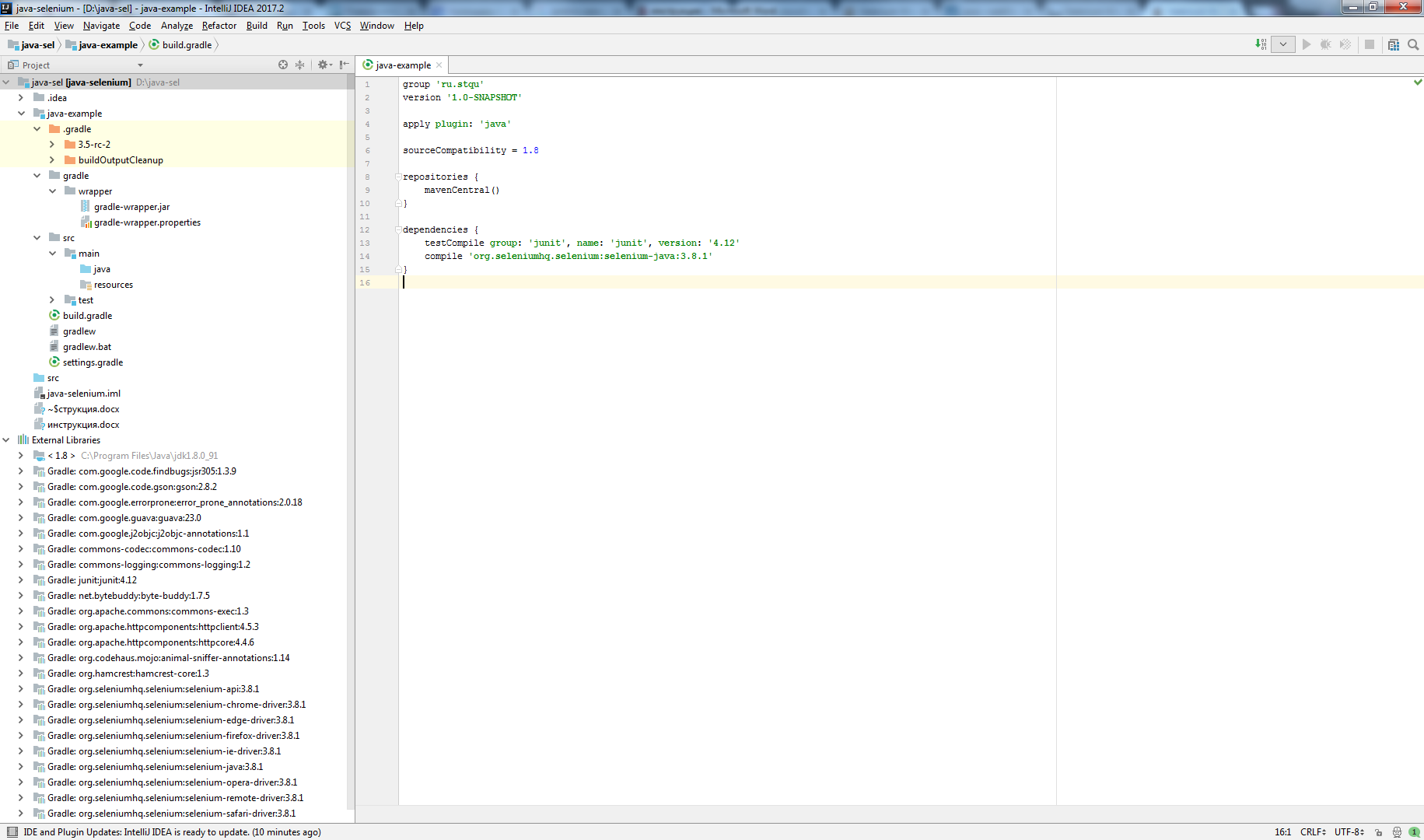


И добавить его в интеладжиа



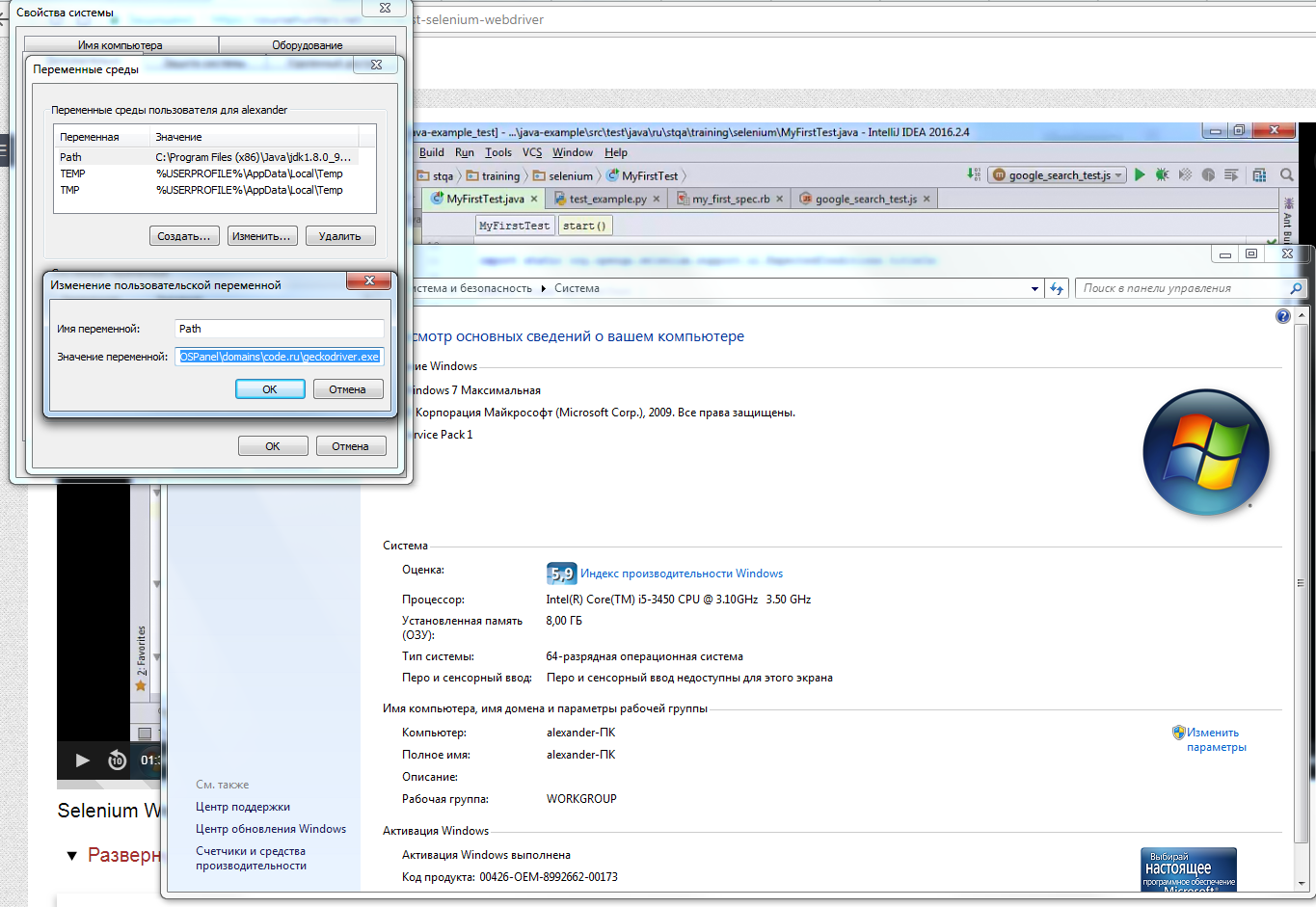
Обновить проект, подключатся куча библиотек





Перед работой скачать geckodriver и chromedriver и прописать пути до них Path переменные среды

C:\Program Files (x86)\Java\jdk1.8.0\_91\bin;D:\OSPanel\modules\php\PHP-5.5\php.exe;D:\OSPanel\domains\code.ru\chromedriver.exe;D:\OSPanel\domains\code.ru\geckodriver.exe



# Методы WebDriver

<http://www.autotest.org.ua/selenium-webdriver-search-elements-and-types-locators/> - поиск элемента

<https://kreisfahrer.gitbooks.io/selenium-webdriver/content/webdriver_api_slozhnie_vzaimodeistviya/kontrol_za_hodom_testa_kastomnie_ozhidaniya,_popapi,_alerti,_iframes.html> – локаторы

**Путь до вебдрайвера**

System.setProperty(“webdriver.chrome.driver”, “C:/Tools/chromedriver.exe”);

**Создать объект вебдрайвера и открыть браузер**

WebDriver driver;  
driver = newChromeDriver();

**Развернуть браузер на весь экране**

driver.manage().window().maximize();

**Открыть сайт/страницу**

driver.get("http://yandex.ru");

driver.navigate().to("http://yandex.ru");

**Создать объекта вебдрайвера и поиск элемента по id, названию элемента, названию класса, css, названию тэга, по ссылке**

WebElement email;

email = driver.findElement(By.id("user\_auth\_email"));

email = driver.findElement(By.name("user\_auth\_email"));

email = driver.findElement(By.className("user\_auth\_email"));

email = driver.findElement(By.**cssSellector**("user\_auth\_email"));

email = driver.findElement(By.cssSelector("[id=user\_auth\_email]"))

ul#menu li.active – 2 тэга ul и li, id=menu название класса тега li равно active

“[name=email]” – полностью совпадает значение

“[name\*=email]” – содержит текст

“[name^=email]” – начинается на текст

“[name$=email]” – заканчивается на текст

“label” – по тегу

“.error” – по классу

“label.error” = по тегу и классу

“label.error.fatal” – по тэгу и двум классам

“label.error[for=email” – по тегу, классу и атрибуту

“label:not(.error)” – ищет с тегом label не класса error

“div#main p” – p где-то внутри div#main

“div#main > p” – p непосредственно внутри div#main

“div#main li:first-child” – первый элемент

“div#main li:last-child” – последний элемент

“div#main li:first-child(n)” – n-ый элемент

email = driver.findElement(By.**xpath**("//\*[@name='user\_auth[email]']")).sendKeys("zamyatinrussia@mail.ru");

// - искать по всей странице

\* - любой тег

@ - перед началом атрибута

"//\*[contains(@title='Name')]" – тег title содержит текст Name

"//\*[starts-with(@title='Name')]" – тег title начинается с текста Name

“//div[@id]=’main’//p” - p где-то внутри div#main

“//div[@id]=’main’/p” - p непосредственно внутри div#main

“//div[@id]=’main’/div[2]” – второй элемент из найденных

driver.findElement(By.xpath("//a[contains(.,'Чек-лист 2')]")); - поиск по текст внутри

//form[.//input[@name=’password’]] - подзапросы

email = driver.findElement(By.tagName("user\_auth\_email"));

email = driver.findElement(By.linkText("user\_auth\_email"));

email = driver.findElement(By.partialLinkText("user\_auth\_email"));

**Получить заголовок страницы**

driver.getTitle();

**Нажатие левой кнопки мыши**

driver.findElement(By.name("btnK")).click();

**Ищет элемент в течении некоторого времени, но не больше 10 секунд**

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

**Получить адрес текущей страницы**

driver.gerCurrentUrl();

**Обновить страницу**

driver.navigate().refresh();

**Закрыть браузер**

driver.quit();

**Передать в элемент значение (ввести в поле)**

email = driver.findElement(By.id("user\_auth\_email"));

email.sendKeys("zamyatinrussia@mail.ru");

**Нажать кнопку Enter**

search = driver.findElement(By.id("search-input"));

search.sendKeys("Чек-лист");

search.sendKeys(Keys.RETURN);

**Очистить поле**

email = driver.findElement(By.id("user\_auth\_email"));

email.clear;

**Проверка:**

Ищет элемент search-items внутри которого список li и количество этих элементов = 8

Assert.*assertTrue*(

driver.findElement(By.*id*(**"search-items"**))

.findElements(By.*tagName*(**"li"**))

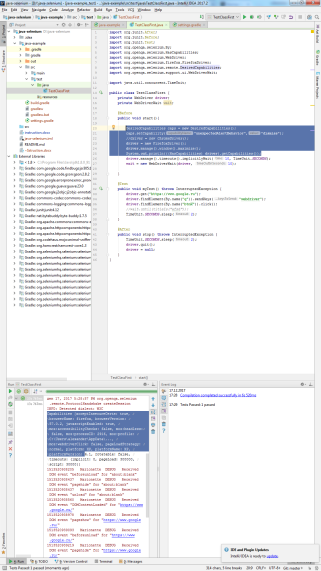
.size() == 8);

Смотрит, что находимся на странице с этим url

Assert.*assertTrue*(driver.getCurrentUrl().contains(**"https://chlist.sitechco.ru/search/project?query=%D0%A7%D0%B5%D0%BA-%D0%BB%D0%B8%D1%81%D1%82&project=10481"**));

**Настройки браузера (capabilities):**

DesiredCapabilities caps = **new** DesiredCapabilities();  
caps.setCapability(**"unexpectedAlertBehavior"**,**"dismiss"**);  
*//driver = new ChromeDriver();***driver** = **new** FirefoxDriver();  
**driver**.manage().window().maximize();  
System.***out***.println(((HasCapabilities) **driver**).getCapabilities());

****

**Работа с cookies:**

driver.manage().deleteAllCookies(); //удаляются только куки для текущей страницу

Стабильные версии браузера Firefox ESR (Extended Support Release) – обновляется примерно раз в год.

**Поиск одного или несколько элементов**

WebElement element;  
**driver**.findElement(<локатор>); - найдет первый подходящий элемент  
List<WebElement> element;  
**driver**.findElements(<локатор>); - найдет все элементы

List<WebElement> email = **driver**.findElements(By.*cssSelector*(**"label.small-label.block"**));  
WebElement email1 = email.get(1);  
*//wait.until(ExpectedConditions.stalenessOf(email));*String em = email1.getAttribute(**"textContent"**);  
System.***out***.println(em);

**Поиск внутри элемента**

WebElement form = driver.findElement(By.id("login-modal"));

form.findElement(By.name("Password"));

тоже самое, что и WebElement input = driver.findElement(By.cssSelector("#login-modal[name=password]"));

**Поиск нескольких элементов**

**Поиск по содержащему тексту**

WebElement rows = driver.findElement(By.cssSelector("span.abbr"));

String rowss = rows.getText();

**Ожидание появления элемента**

**driver**.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***); - ожидает до 10 секунд, если находит раньше то не будет ждать 10 секунд, а продолжит работу

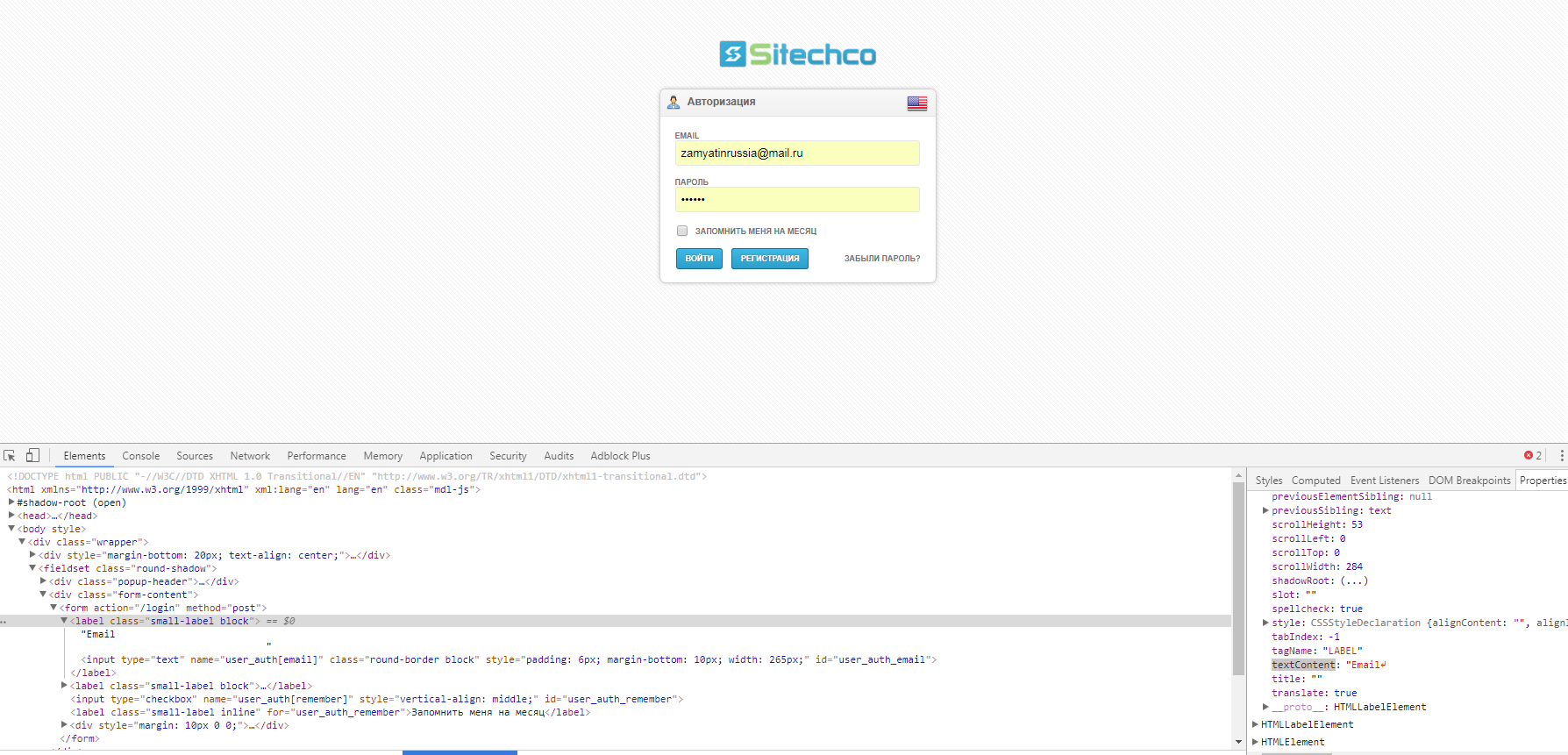
**private** WebDriverWait **wait**;

**wait** = **new** WebDriverWait(**driver**, 10);

**public boolean** isElementPresent(By locator){  
 **wait**.until((WebDriver d) -> d.findElement(locator)); - явное ожидание  
 **return true**;  
}

**Получение атрибутов элемента**

WebElement email = **driver**.findElement(By.*cssSelector*(**"label.small-label.block"**));  
String em = email.getAttribute(**"textContent"**);  
System.***out***.println(em);



**Определение видимости элемента**

**Получение стилей элемента**

**Размер и положение элемента**

getSize – размер элемента

getLocation – положение на странице

getRect – размер и положение

(Locatable element).getCoordinates().inView()

**Сложные действия Advanced Interactions API**

click

sendKeys

clickAndHold – зажать и не отпускать кнопку мыши

moveToElement / moveByOffset – перемещение мыши на какой-либо элемент и на какое-то расстояние в пикселях

release – отпускает кнопку мыши

keyDown – нажать клавишу

keyUp – отпустить клавишу

**Ожидание исчезновения элемента**

WebElement email = **driver**.findElement(By.*cssSelector*(**"label.small-label.block"**));  
**wait**.until(ExpectedConditions.*stalenessOf*(email));

**Ожидание видимости элемента**

WebElement email = **driver**.findElement(By.*cssSelector*(**"label.small-label.block"**));  
**wait**.until(visibilityOf(email));

**//wait**.until(not(visibilityOf(email)));

**Разные ожидания**

wait.until(titles(“Поиск в google”)); - ожидает заголовок “Поиск в google”

wait.until(urlContains(“login.php”)); - ожидает такой урл

wait.until(numberOfWindowsToBe(2)); - ожидает 2 открывшихся окна

wait.until(attributeContains(element, “class”, “visible”))

wait.until(textToBePresentInElement(element, “OK”))

wait.until(elementToBeSelected(element))

**Ожидание загрузки страницы**

**Alert**

Alert alert = **driver**.switchTo().alert(); - поиск окна алерта  
Alert alert1 = **wait**.until(alertIsPresent()); - поиск окна алерта с ожиданием  
alert.getText() / alert.sendKeys(); - заполнение в диалоговом окне  
alert.accept() / alert.dismiss(); - закрытие диалогового окна

caps.setCapability(“unexpectedAlertBehaviour”,”dismiss”) – по умолчанию закроет диалоговое окно

**Загрузка файлов на сервер**

Заполнять поле с помощью sendKeys

Если поле ввода скрыто, сделать его видимым

**Окна и вкладки**

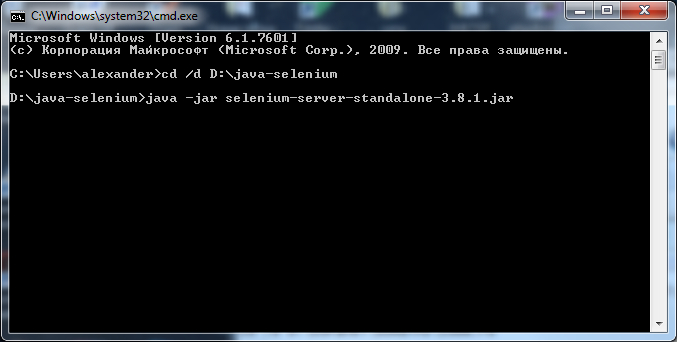
# УДАЛЕННЫЙ ЗАПУСК БРАУЗЕРА

Запустить селениум драйвер

перейти в папку где лежит селениум вебсервер

cd /d D:\java-selenium

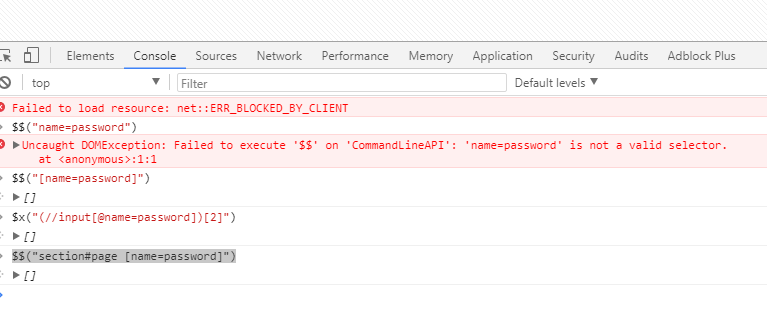
запустить java -jar selenium-server-standalone-3.8.1.jar



В коде поменять

*//driver = new FirefoxDriver();***driver** = **new** RemoteWebDriver(DesiredCapabilities.*firefox*());

# Поиск в консоли браузера



$$("[name=password]")

$x("(//input[@name=password])[2]")

$$("section#page [name=password]")