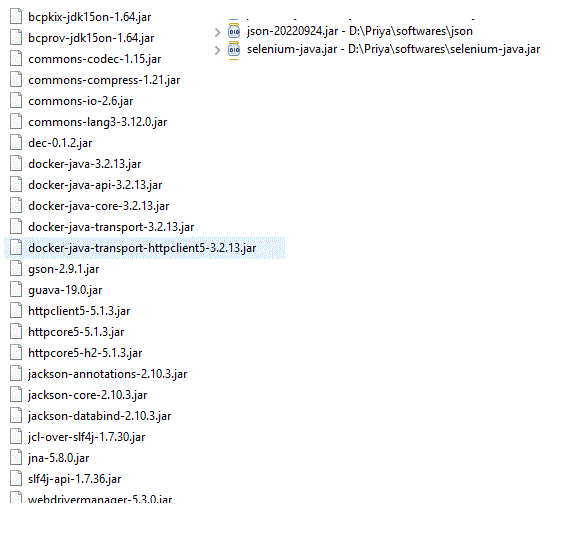
Selenium is used for cross-browser automated testing and as it supports a lot of browsers, they are very helpful. For using Selenium in the project we need separate kinds of jars which are presented in the below screenshot.



Otherwise, by using maven also we can take the necessary dependencies. As a web scraping way, we need to know each and every element’s internal representation of HTML. That will help to identify the element and prepopulate the values from Java code. Let’s see how to do it.

### Example Project

In our example let us see how to do that by using the chromedriver. i.e. in the chrome browser, by using selenium, we are automating the address-filling process.

**AmazonWebsiteAddressAutoSubmit.java**

* Java

|  |
| --- |
| **import** java.util.List;  **import** java.util.concurrent.TimeUnit;  **import** org.openqa.selenium.By;  **import** org.openqa.selenium.WebDriver;  **import** org.openqa.selenium.WebElement;  **import** org.openqa.selenium.chrome.ChromeDriver;  **import** org.openqa.selenium.chrome.ChromeOptions;  **import** org.openqa.selenium.support.FindBy;  **import** org.openqa.selenium.support.PageFactory;    **import** io.github.bonigarcia.wdm.WebDriverManager;    **public** **class** AmazonWebsiteAddressAutoSubmit {    **public** **static** **void** main(String[] args) {          ChromeOptions chromeOptions = **new** ChromeOptions();          WebDriverManager.chromedriver().setup();          WebDriver driver = **new** ChromeDriver(chromeOptions);            driver.manage().timeouts().pageLoadTimeout(15, TimeUnit.MINUTES);            // Launch the Online Store Website  **try** {              driver.get("<https://www.amazon.in/a/addresses/add?ref=ya_address_book_add_post>");                LoginPage login = **new** LoginPage(driver);                // Replace with your emailId              login.set\_username("tltxxx@gmail.com");              login.continueButtonClick();                // Replace with your password              login.set\_password("xxxx");              login.click\_button();                NewAddressDetails addAddress = **new** NewAddressDetails(driver);                // Replace the fields as per your requirement              addAddress.set\_fullname("xxxxx");              addAddress.set\_phonenumber("1234567890");              addAddress.set\_postalCode("625 016");              addAddress.set\_addressLine1("AddressLine1");              addAddress.set\_addressLine2("AddressLine2");              addAddress.set\_city("City");                // Actually state has to be set              // If not set, it will throw error while              // clicking on the next submit button              List<WebElement> allElements = driver.findElements(By.xpath("//input[@class='a-button-input']"));              WebElement clickableElement;  **for** (WebElement element : allElements) {  **if** (element.getAttribute("aria-labelledby").equals("address-ui-widgets-form-submit-button-announce")) {                      clickableElement = element;                      clickableElement.click();                  }              }                // Edit address part              Thread.sleep(5000);                // We can check whether address              // got added successfully              driver.get("<https://www.amazon.in/a/addresses>");                Thread.sleep(5000);                // Print a Log In message to the screen              System.out.println("Successfully performed the operation of adding address");                Thread.sleep(5000);          } **catch** (InterruptedException e) {              // TODO Auto-generated catch block              e.printStackTrace();          }            // Close the driver          driver.quit();          // driver.close();      }    }    **class** LoginPage {  **private** WebDriver driver;        @FindBy(id = "ap\_email")      WebElement username;      @FindBy(id = "ap\_password")      WebElement password;      @FindBy(id = "signInSubmit")      WebElement button;      @FindBy(id = "continue")      WebElement continueButton;    **public** LoginPage(WebDriver driver) {          // initialize elements          PageFactory.initElements(driver, **this**);      }    **public** **void** set\_username(String usern) {          username.clear();          username.sendKeys(usern);      }    **public** **void** set\_password(String userp) {          password.clear();          password.sendKeys(userp);      }    **public** **void** click\_button() {          button.submit();      }    **public** **void** continueButtonClick() {          continueButton.submit();      }  }    **class** NewAddressDetails {      // private WebDriver driver;      @FindBy(id = "address-ui-widgets-enterAddressFullName")      WebElement fullName;      @FindBy(id = "address-ui-widgets-enterAddressPhoneNumber")      WebElement phoneNumber;        @FindBy(id = "address-ui-widgets-enterAddressPostalCode")      WebElement postalCode;      @FindBy(id = "address-ui-widgets-enterAddressCity")      WebElement city;        @FindBy(id = "address-ui-widgets-enterAddressLine1")      WebElement addressLine1;      @FindBy(id = "address-ui-widgets-enterAddressLine2")      WebElement addressLine2;      // @FindBy(id = "address-ui-widgets-enterAddressStateOrRegion")      @FindBy(className = "a-button-inner")      WebElement stateOrRegion;      @FindBy(className = "a-button-input")      // @FindBy(xpath = "//input[@class='a-button-input']")      WebElement button;    **public** NewAddressDetails(WebDriver driver) {          // initialize elements          PageFactory.initElements(driver, **this**);      }    **public** **void** set\_fullname(String fname) {          fullName.clear();          fullName.sendKeys(fname);      }    **public** **void** set\_phonenumber(String phnumber) {          phoneNumber.clear();          phoneNumber.sendKeys(phnumber);      }    **public** **void** set\_postalCode(String pcode) {          postalCode.clear();          postalCode.sendKeys(pcode);      }    **public** **void** set\_city(String cty) {          city.clear();          city.sendKeys(cty);      }    **public** **void** set\_addressLine1(String line1) {          addressLine1.clear();          addressLine1.sendKeys(line1);      }    **public** **void** set\_addressLine2(String line2) {          addressLine2.clear();          addressLine2.sendKeys(line2);      }    **public** **void** set\_stateOrRegion(String region) {          stateOrRegion.clear();          stateOrRegion.sendKeys(region);      }    **public** **void** click\_button() {          button.click();          /\*           \* Actions builder = new Actions(AutoFill.driver);           \* builder.moveToElement(button).click(button); builder.perform();           \*/      }  }    **class** EditAddressDetails1 {  **private** WebDriver driver;        @FindBy(className = "a-button-input")      WebElement button;    **public** EditAddressDetails1(WebDriver driver) {          // initialize elements          PageFactory.initElements(driver, **this**);        }    **public** **void** click\_button() {          button.submit();      }  } |

*URL: [https://www.amazon.in/a/addresses/add?ref=ya\_address\_book\_add\_post](https://www.amazon.in/a/addresses/add?ref=ya_address_book_add_post" \t "https://www.geeksforgeeks.org/how-to-automate-amazon-like-e-commerce-website-with-selenium/_blank)*

Once selenium tries to access the page, according to the requested information like

* Username (It can be your registered email id)
* Password

are provided first. Once it gets validated, it will go to the next page i.e. address addition page. There need to provide the mandatory details like

* Name
* PhoneNumber
* PostalCode
* AddressLine1
* AddressLine2
* City
* State

and then need to press on Add address button. If all the above information is validated it will successfully add the address or else will show the error message. So in a way, instead of manually typing the data and submitting the form, all the data is automatically submitted via our code. Hence Selenium is the best choice for automated testing. Whenever there are errors showing as a means of validation, we need to rectify them via code only. So for that specific tags need to be known for addressing them. In the code under the “LoginPage” class,

* ap\_email
* ap\_password
* signInSubmit for submit
* continue for continue

All the above are ids and easily they are identified by means of

@FindBy(id = "ap\_email")

WebElement username;

@FindBy(id = "ap\_password")

WebElement password;

@FindBy(id = "signInSubmit")

WebElement button;

@FindBy(id = "continue")

WebElement continueButton;

Once the details are validated both in terms of client-side validations (like no blank spaces/no nulls/proper email id etc for email id and no blank/no null for password) and server-side validation like (email/password gets registered to amazon site), it will go to the address adding page

@FindBy(id = "address-ui-widgets-enterAddressFullName")

WebElement fullName;

@FindBy(id = "address-ui-widgets-enterAddressPhoneNumber")

WebElement phoneNumber;

@FindBy(id = "address-ui-widgets-enterAddressPostalCode")

WebElement postalCode;

@FindBy(id = "address-ui-widgets-enterAddressCity")

WebElement city;

@FindBy(id = "address-ui-widgets-enterAddressLine1")

WebElement addressLine1;

@FindBy(id = "address-ui-widgets-enterAddressLine2")

WebElement addressLine2;

// @FindBy(id = "address-ui-widgets-enterAddressStateOrRegion")

@FindBy(className = "a-button-inner")

WebElement stateOrRegion;

@FindBy(className = "a-button-input")

// @FindBy(xpath = "//input[@class='a-button-input']")

WebElement button;

Here also all data get validated and move to the next page. If any mandatory fields are not proper it will show error as well. If everything is fine only, it will move to the next page. For identifying the elements, for specific id, we need to mention @FindBy(id = “address-ui-widgets-enterAddressLine2”) where “address-ui-widgets-enterAddressLine2” is the exact id value and @FindBy(className = “a-button-inner”) where “a-button-inner” is identified with the class name. In case multiple times the class got repeated, by means of XPath, we can try to retrieve

List<WebElement> allElements = driver.findElements(By.xpath("//input[@class='a-button-input']"));

WebElement clickableElement;

for (WebElement element : allElements) {

if (element.getAttribute("aria-labelledby").equals("address-ui-widgets-form-submit-button-announce")) {

clickableElement = element;

clickableElement.click();

}

}