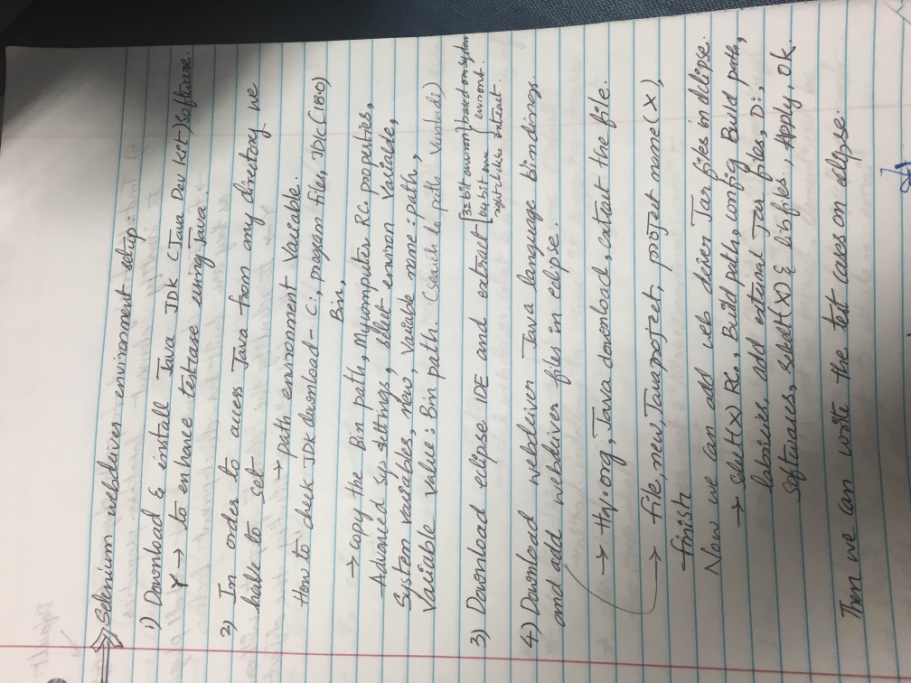
(1)what are limitations in selenium webdriver

Some limitations of Selenium Automation tool are as follows:

1. It does not support and non web-based applications, it only supports web based applications.  
2. Its and open source tool so in case of any technical issues you need to rely on the selenium community forums to get your issue resolved.  
3. You need to know at least one of the supported language very well in order to automate your application successfully.

4. No inbuilt reporting capability so you need plugins like JUnit and TestNG for test reports.  
5. Lot of challenges with IE browser.

(2) installing/configure selenium



(3) what are different ways of locating elements in selenium

There are some browser tools that you can use in order to identify web elements. These are:

* Firebug for Firefox
* Google Developer Tools for Chrome
* Web Inspector for Safari

Selenium WebDriver API supports different possibilities to identify elements: by ID, by CLASS, by NAME, by CSS selector, by XPath, by TAG name. Also you define your custom selector in order to interact with the elements.

(4) which is fastest way to identify elements in web page?

X-Paths are accurate, indeed. However, they are also very fragile. Any change to the website that moves things around has the potential of breaking your tests. IDs are by far the least fragile, since it doesn't matter if your changes to a , your tests will always find it.

(5) what is absolute path and relative path in xpath

bsolute Xpath :

1) start selection from the document node  
2) Starts with //  
3) e.g. “/html/body/p” matches all the paragraph elements  
  
Google Search Box : /html/body/div[1]/div[2]/div[1]/div[1]/div[3]/div/div/div/form/fieldset[2]/div/div/div

Relative Xpath:

1) start selection matching anywhere in the document  
2) Starts with /  
3) e.g. “//p” matches all the paragraph elements starts with p  
  
Google Search Box : //\*[@id='gbqfqwb']

(6) different types of waits or synchronization in selenium webdriver

When our automation execution starts, then there should be fair communication between tool and application.   
What i mean here is if the tool is too fast of execution and the application/objects are not fully loaded/not ready by that time, then our automation test cases will fail. So the tool should wait(appropriate) till the objects are present/ready in the application, so that communication/synchronization happens between the tool and application so the chances of our test cases will pass.  
  
Synchronization or Waits can be done in two ways.  
1. Explicit Waits  
2. Implicit Waits  
  
Explicit Waits:  
This can be achieved in 2 ways.  
  
Thread.sleep:   
Thread.sleep waits the specified time irrespective of the object state.  
Ex: Thread.sleep(30000);  
Here the execution is halted for 30 Sec., even if the object you are looking exists in 10 sec. So here tool unnecessarily waits for 20 sec.  
Execution wont wait after 30 sec.s even if the object does not available, so the chances of your Test fails.  
  
WebDriverWait:   
We can tell the tool to wait only till the Condition met. Once the condition is met, the tool proceed with the next step.  
This can be done with WebDriverWait in conjunction with ExpectedConditions Class.   
There are few methods supported in ExpectedConditions class to support synchronisation.  
Here is the example:  
WebDriverWait wait = new WebDriverWait(driver, 30);  
WebElement o\_element = wait.until(ExpectedConditions.elementToBeClickable(By.id("Object Id")));  
  
Here the tool waits a maximum time of 30 Sec., if the object you are looking is displayed in 10 sec. then the execution proceeds with the next step afte 10 secs. rather than waiting for 30 secs.  
  
If you dont want to include any methods in ExpectedConditions class, then you can use below code:  
WebDriver driver = new FirefoxDriver();  
driver.get("Your URL");  
WebElement o\_Element = (new WebDriverWait(driver, 30))  
  .until(new ExpectedCondition<WebElement>(){  
    @Override  
    public WebElement apply(WebDriver d) {  
        return d.findElement(By.id("Object Id"));  
    }});  
Here WebDriverWait by default calls the ExpectedCondition every 500 milliseconds until it returns successfully or wait for maximum of 30 sec.  
  
  
Implicit Waits:  
An implicit wait is to tell WebDriver to poll the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available. The default setting is 0. Once set, the implicit wait is set for the life of the WebDriver object instance.  
WebDriver driver = new FirefoxDriver();  
driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);  
driver.get("Your URL");  
WebElement o\_Element = driver.findElement(By.id("Object Id"));

(7) how to save screen shots using selenium webdriver?

public class TestSample

{

static WebDriver driver;

@Before

public void setUp() {

//start new webdriver session, for eg using firefox

driver = new FirefoxDriver();

}

@Test

public void aTest() {

driver.get("http://www.google.com/");

//more test logic - test might pass or fail at this point

}

@After

public void tearDown() {

// take the screenshot at the end of every test

File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);

// now save the screenshto to a file some place

FileUtils.copyFile(scrFile, new File("c:\\tmp\\screenshot.png"));

//quit WebDriver session

driver.quit();

}

}

(8) how to handle multiple windows in selenium webdriver?

**import**java.util.List;  
**import**org.junit.After;  
**import**org.junit.Before;  
**import**org.junit.Test;  
**import**org.openqa.selenium.By;  
**import**org.openqa.selenium.WebDriver;  
**import**org.openqa.selenium.WebElement;  
**import**org.openqa.selenium.firefox.FirefoxDriver;  
  
  
**public class**MultipleWindowsHandle {  
  
     
     WebDriver driver;    
     @Before    
     **public void**setup() **throws**Exception {    
     driver=**new**FirefoxDriver();    
     String URL="http://www.seleniummaster.com";     
     driver.get(URL);    
     driver.manage().window().maximize();    
     }    
     @Test    
     **public void**test() **throws**Exception {     
     // Opening site    
     driver.findElement(By.xpath("//img[@alt='SeleniumMasterLogo']")).click();    
     // Storing parent window reference into a String Variable    
     String Parent\_Window = driver.getWindowHandle();      
      // Switching from parent window to child window     
     **for**(String Child\_Window : driver.getWindowHandles())    
     {    
     driver.switchTo().window(Child\_Window);    
     // Performing actions on child window    
     driver.findElement(By.id("dropdown\_txt")).click();    
     List  dropdownitems=driver.findElements(By.xpath("//div[@id='DropDownitems']//div"));    
     **int**dropdownitems\_Size=dropdownitems.size();    
     System.out.println("Dropdown item size is:"+dropdownitems\_Size);    
     ((WebElement) dropdownitems.get(1)).click();    
     driver.findElement(By.xpath("//\*[@id='anotherItemDiv']")).click();    
     }    
     //Switching back to Parent Window    
     driver.switchTo().window(Parent\_Window);    
     //Performing some actions on Parent Window    
     driver.findElement(By.className("btn\_style")).click();    
     }    
      @After    
      **public void**close() {    
      driver.quit();    
      }     
     }

(9) how to lanuch webpage using chrome driver?

System.setProperty("webdriver.chrome.driver",

"C:/Program Files (x86)/Google/Chrome/Application/chrome.exe");

WebDriver driver=new ChromeDriver();

driver.get("http://www.yahoo.com");

(10) what is desired capabilities in selenium webdriver?

|  |  |
| --- | --- |
|  | 1. It is a class in org.openqa.selenium.remote.DesiredCapabilities package. 2. It gives facility to set the properties of browser. Such as to set BrowserName, Platform, Version of Browser. 3. Mostly DesiredCapabilities class used when do we used Selenium Grid. 4. We have to execute mutiple TestCases on multiple Systems with different browser with Different version and Different Operating System.   Example:  WebDriver driver;  String baseUrl , nodeUrl;  baseUrl = "https://www.facebook.com";  nodeUrl = "http://192.168.10.21:5568/wd/hub";  DesiredCapabilities capability = DesiredCapabilities.firefox();  capability.setBrowserName("firefox");  capability.setPlatform(Platform.WIN8\_1);  driver = new RemoteWebDriver(new URL(nodeUrl),capability);  driver.manage().window().maximize();  driver.manage().timeouts().implicitlyWait(2, TimeUnit.MINUTES); |