

## BaseClass.java

```
1 package com.mavenproject;
2
3 import java.awt.AWTException;
4 import java.awt.Robot;
5 import java.awt.event.KeyEvent;
6 import java.io.File;
7 import java.io.IOException;
8 import java.util.Set;
9 import java.util.concurrent.TimeUnit;
10
11 import org.apache.commons.io.FileUtils;
12 import org.openqa.selenium.JavascriptExecutor;
13 import org.openqa.selenium.OutputType;
14 import org.openqa.selenium.TakesScreenshot;
15 import org.openqa.selenium.WebDriver;
16 import org.openqa.selenium.WebElement;
17 import org.openqa.selenium.chrome.ChromeDriver;
18 import org.openqa.selenium.ie.InternetExplorerDriver;
19 import org.openqa.selenium.interactions.Actions;
20 import org.openqa.selenium.support.ui.ExpectedConditions;
21 import org.openqa.selenium.support.ui.Select;
22 import org.openqa.selenium.support.ui.WebDriverWait;
23
24 public class BaseClass {
25     public static WebDriver driver;
26
27 //Browser launch
28     public static WebDriver browserlaunch(String browser) {
29         if (browser.equalsIgnoreCase("chrome")) {
30
31             System.setProperty("webdriver.chrome.driver",
32                 System.getProperty("user.dir") + ("\\Driver\\
33                 \\chromedriver.exe"));
34             driver = new ChromeDriver();
35         }
36         else if (browser.equalsIgnoreCase("internetexplorer")) {
37             System.setProperty("webdriver.ie.driver",
38                 System.getProperty("user.dir") + ("\\Driver\\
39                 \\IEDriverServer.exe"));
40             driver = new InternetExplorerDriver();
41         }
42         driver.manage().window().maximize();
43         return driver;
44     }
45 // Get URL
```

BaseClass.java

```
46     public static void geturl(String url) {
47         driver.get(url);
48     }
49
50 //Click
51     public static void click(WebElement element) {
52         element.click();
53     }
54
55 //SendKeys
56     public static void sendkeys(WebElement element, String input) {
57         element.sendKeys("input");
58     }
59
60 //Close
61     public static void close() {
62         driver.close();
63     }
64
65 //Quit
66     public static void quit() {
67         driver.quit();
68     }
69
70 //navigate to URL
71     public static void navigatetourl(String url) {
72         driver.navigate().to(url);
73     }
74
75 // navigate to forward
76     public static void navigatetoforward() {
77         driver.navigate().forward();
78     }
79
80 //Navigate to back
81     public static void navigatetoback() {
82         driver.navigate().back();
83     }
84
85 //Navigate refresh
86     public static void navigaterefresh() {
87         driver.navigate().refresh();
88     }
89
90 //Alert accept
91     public static void alertaccept() {
92         driver.switchTo().alert().accept();
```

## BaseClass.java

```
93     }
94
95 //Alert dismiss
96     public static void alertdismiss() {
97         driver.switchTo().alert().dismiss();
98     }
99
100 //Alert sendKeys
101     public static void alertsendkeys(String input) {
102         driver.switchTo().alert().sendKeys("input");
103     }
104
105 //Mouse actions Click
106     public static void mouseclick(WebElement element) {
107         Actions ac = new Actions(driver);
108         ac.click(element).build().perform();
109     }
110
111 //Mouse actions Right Click
112     public static void mouserightclick(WebElement element) {
113         Actions ac = new Actions(driver);
114         ac.contextClick(element).build().perform();
115     }
116
117 //Mouse actions Double click
118     public static void mousedoubleclick(WebElement element) {
119         Actions ac = new Actions(driver);
120         ac.doubleClick(element).build().perform();
121     }
122
123 // Frame By id
124     public static void frameid(String id) {
125         driver.switchTo().frame(id);
126     }
127
128 // Frame By index
129     public static void frameindex(int index) {
130         driver.switchTo().frame(index);
131     }
132
133 // Frame By WebElement
134     public static void framewebelement(WebElement element) {
135         driver.switchTo().frame(element);
136     }
137
138 // From Frame To Main Page
139     public static void frametomainpage() {
```

BaseClass.java

```
140     driver.switchTo().defaultContent();
141 }
142
143 //Robot VK UP
144 public static void vkup() throws AWTException {
145     Robot r = new Robot();
146     r.keyPress(KeyEvent.VK_UP);
147     r.keyRelease(KeyEvent.VK_UP);
148 }
149
150 //Robot VK DOWN
151 public static void vkdown() throws AWTException {
152     Robot r = new Robot();
153     r.keyPress(KeyEvent.VK_DOWN);
154     r.keyRelease(KeyEvent.VK_DOWN);
155 }
156
157 //Robot VK ENTER
158 public static void vkenter() throws AWTException {
159     Robot r = new Robot();
160     r.keyPress(KeyEvent.VK_ENTER);
161     r.keyRelease(KeyEvent.VK_ENTER);
162 }
163
164 //Get windows handling
165 public static void getwindowshandling() {
166     Set<String> windowHandles = driver.getWindowHandles();
167     for (String s : windowHandles) {
168         String tittle = driver.switchTo().window(s).getWindowHandle();
169         System.out.println(tittle);
170     }
171 }
172
173 //Drop down by value
174 public static void dropdownbyvalue(WebElement element, String value) {
175     Select s = new Select(element);
176     s.selectByValue(value);
177 }
178
179 //Drop down by index
180 public static void dropdownbyindex(WebElement element, int index) {
181     Select s = new Select(element);
182     s.selectByIndex(index);
183 }
184
185 //Drop down by visible of text
186 public static void dropdownbyvisibleoftext(WebElement element, String
```

## BaseClass.java

```
text) {
187     Select s = new Select(element);
188     s.deselectByVisibleText(text);
189 }
190
191 //Check box
192 public static void checkbox(WebElement element) {
193     element.click();
194 }
195
196 //Is enabled
197 public static void isenabled(WebElement element) {
198     System.out.println(element.isEnabled());
199 }
200
201 //Is displayed
202 public static void isdisplayed(WebElement element) {
203     System.out.println(element.isDisplayed());
204 }
205
206 //Is Selected
207 public static void isselected(WebElement element) {
208     System.out.println(element.isSelected());
209 }
210
211 //Get options
212 public static void getoptions(WebElement element) {
213     Select s = new Select(element);
214     java.util.List<WebElement> options = s.getOptions();
215     for (WebElement webElement : options) {
216         System.out.println(webElement.getText());
217     }
218 }
219
220 //Get title
221 public static void gettitle() {
222     System.out.println(driver.getTitle());
223 }
224
225 //Get current URL
226 public static void getcurrenturl() {
227     System.out.println(driver.getCurrentUrl());
228 }
229
230 //Get text
231 public static void gettext(WebElement element) {
232     System.out.println(element.getText());
```

## BaseClass.java

```
233     }
234
235 //Get attribute
236     public static void getattribute(WebElement element, String input) {
237         System.out.println(element.getAttribute(input));
238     }
239
240 //Implicit wait
241     public static void implicitwait(int sec, TimeUnit unit) {
242         driver.manage().timeouts().implicitlyWait(sec, unit);
243     }
244
245 //explicit wait
246     public static void explicitwait(int sec, WebElement element) {
247         WebDriverWait wait = new WebDriverWait(driver, sec);
248         wait.until(ExpectedConditions.visibilityOf(element));
249     }
250
251 //Take screenshot
252     public static void takescreenshot(String location) throws IOException {
253         TakesScreenshot sc = (TakesScreenshot) driver;
254         File start = sc.getScreenshotAs(OutputType.FILE);
255         File end = new File(location);
256         FileUtils.copyFile(start, end);
257     }
258
259 // Scroll intoView
260     public static void scrollintoview(WebElement element) {
261         JavascriptExecutor js = (JavascriptExecutor) driver;
262         js.executeScript("arguments[0].scrollIntoView();", element);
263     }
264
265 // Scroll By
266     public static void scrollby(WebElement element) {
267         JavascriptExecutor js = (JavascriptExecutor) driver;
268         js.executeScript("window.scrollBy();", element);
269     }
270
271 //Is Multiple
272     public static void ismultiple(WebElement element) {
273         Select s = new Select(element);
274         System.out.println(s.isMultiple());
275     }
276
277 // Radio button
278     public static void radiobutton(WebElement element) {
279         element.click();
```

BaseClass.java

```
280     }  
281 }  
282
```