סיכום שיעור שמיני:

1. **Test Synchronization**

לפעמים נצטרך לחכות עד אשר תנאי כלשהו יתממש ע"מ להמשיך את הטסט לדוגמא:
לחכות עד אלמנט יופיע / ייעלם מהמסך.
דיברנו על מספר דרכים למימוש:

- **Thread.sleep()** – יחכה פרק זמן שיוגדר בלי להתחשב האם התנאי התקיים.

Thread.sleep causes the current thread to suspend execution for a specified period. This is an efficient means of making processor time available to the other threads of an application or other applications that might be running on a computer system. The sleep method can also be used for pacing, as shown in the example that follows, and waiting for another thread with duties that are understood to have time requirements, as with the SimpleThreads example in a later section.
Two overloaded versions of sleep are provided: one that specifies the sleep time to the millisecond and one that specifies the sleep time to the nanosecond. However, these sleep times are not guaranteed to be precise, because they are limited by the facilities provided by the underlying OS. Also, the sleep period can be terminated by interrupts, as we'll see in a later section. In any case, you cannot assume that invoking sleep will suspend the thread for precisely the time period specified.

- **Implicit wait** – יגרום לדרייבר להמתין פרק זמן שיוגדר עד אשר התנאי יתקיים, להבדיל מהקודם, במידה ויוגדר פרק זמן אך התנאי יתקיים קודם לכן התכנית תמשיך.
  Implicit wait יעבוד על כל מקום בקוד בו נבצע findElement / findElements.

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.

- **Explicit wait –** בדומה לקודם, יחכה עד אשר יופיע אלמנט, רק ש **Explicit wait** מיועד פר אלמנט ולא לכלל האלמנטים ע"י שימוש במחלקה **ExpectedConditions (סעיף 2).**

An explicit wait is code you define to wait for a certain condition to occur before proceeding further in the code. The worst case of this is Thread.sleep(), which sets the condition to an exact time period to wait. There are some convenience methods provided that help you write code that will wait only as long as required. WebDriverWait in combination with ExpectedCondition is one way this can be accomplished.

- Fluent wait – נותן לנו אפשרות לשנות את הפרמטרים הדיפולטיבים שמגיעים עם ה wait

```
import org.junit.*;
public class Synchronization {
@Test
   public static void test() {
          // wait 10 seconds regardless if element is shown.
          Thread.sleep(10000);
          driver.findElement(By.cssSelector("div[style='']")).isDisplayed();

          // wait up to 10 seconds, if elements are found before keep running – applies for all elements
          WebDriver driver = new FirefoxDriver();
          driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

          // wait up to 10 seconds for the below element only

           WebDriverWait wait = new WebDriverWait(driver,10);

           wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("statedropdown")));

          // fluent wait with custom parameters

          Wait<WebDriver> wait = new FluentWait<WebDriver>(driver) //Wait for the condition

             .withTimeout(30, TimeUnit.SECONDS)  // checks for the condition with 5 seconds interval

             .pollingEvery(5, TimeUnit.SECONDS)

             .ignoring(NoSuchElementException.class); //Which will ignore the NoSuchElementException

          }
}
```

2. **ExpectedConditions**:

כפי שראינו מקודם כשהשתמשנו ב **Explicit wait** נעזרנו במחלקה בשם **ExpectedConditions**
המחלקה מספקת לנו סט תנאים נפוצים ע"מ לחכות לאותם התנאים להתקיים.
להלן מספר דוגמאות:

- תנאי הבודק האם האלמנט נמצא:

```
WebDriverWait wait = new WebDriverWait(driver, waitTime);
wait.until(ExpectedConditions.presenceOfElementLocated(locator));
```

- תנאי הבודק האם האלמנט לחיץ:

```
WebDriverWait wait = new WebDriverWait(driver, waitTime);
wait.until(ExpectedConditions. elementToBeClickable (locator));
```

- תנאי הבודק האם האלמנט visible

```
WebDriverWait wait = new WebDriverWait(driver, waitTime);
wait.until(ExpectedConditions. visibilityOfElementLocated (locator));
```

- ועוד הרבה....
https://seleniumhq.github.io/selenium/docs/api/dotnet/html/T_OpenQA_Selenium_Support_UI_ExpectedConditions.htm

3. **Page load timeout**:

Page load timeout הוא זה שיקבע כמה זמן הדרייבר ימתין שהדף יסיים להיטען.

פרק זמן ברירת המחדל הינו 30 שניות.

ע"מ לשנות את פרק זמן זה נשתמש בקוד הבא:

```
driver.manage().timeouts().pageLoadTimeout(5, TimeUnit.SECONDS);
```

בו נגדיר את כמות הזמן (5) ואת יחידת הזמן (TimeUnit.SECONDS).
במידה ופרק זמן זה יעבור נקבל *TimeoutException*.

4. **Angular JS**:

הרבה מדפי האינטרנט עושים שימוש ב Angular JS, מכיוון שהרבה דברים קורים "on the fly" נוצר קושי באוטומציה מכיוון שלא תמיד ניתן להאזין לכל ה events שקורים.

אחד מהפתרונות שראינו הוא שניתן להשתמש בקוד Java בשימוש מחלקות אחרות ולנסות לעבוד בצורה זו (הדרך הפחות מומלצת):

```
public static ExpectedCondition<Boolean>
angularHasFinishedProcessing() {

    return new ExpectedCondition<Boolean>() {

        @Override

        public Boolean apply(WebDriver driver) {

            return Boolean.valueOf(((JavascriptExecutor)
driver).executeScript("return (window.angular !== undefined) &&
(angular.element(document).injector() !== undefined) &&
(angular.element(document).injector().get('$http').pendingRequests.l
ength === 0)").toString());

        }};}}

WebDriverWait wait = new WebDriverWait(driver, 15, 100);

wait.until(AdditionalConditions.angularHasFinishedProcessing()));
```

הדרך המומלצת לעבודה, היא לעבוד עם NgWebDriver שיודע לעבוד מול אלמנטים של Angular (לדוגמא: ...binding, repeater, model) ומשתמש ב Protactor .

```
driver.findElement(ByAngular.buttonText("Submit")).click();
```

https://github.com/paul-hammant/ngWebDriver
http://www.protractortest.org/#/

** חשוב להבין שניתן לעבוד מול Angular בעזרת הדרכים שלמדנו.

5. **Actions**:

לפעמים נצטרך לבצע פעולות "מורכבות" ע"מ לדמות אינטרקציות של משתמש אנושי, כמו לחיצות מרובות, פעולות עם העכבר וכו'.
ע"מ לעשות זאת נשתמש במחלקה שנקראת Actions.
להלן מס' דוגמאות:

- לחיצת עכבר כפולה:

```
WebElement doubleClickElement = driver.findElement(By.id("button"));

Action doubleClickAction = new Actions(driver);

doubleClickAction.doubleClick(doubleClickElement);

doubleClickAction.build.perform();
```

- לחיצה על מספר אלמנטים ברשימה:

```
List<WebElement> elementsList=
driver.findElements(By.Name("option"));

Actions builder=new Actions(driver);

builder.clickAndHold((WebElement)
elementsList.get(3)).clickAndHold((WebElement)
elementsList.get(5)).click();

builder.build().perform();
```

- מעבר עם עכבר מעל אלמנט:

```
Actions hoverAction = new Actions(driver);

WebElement firstHoverElement = driver.findElement(By.id(“first"));

WebElement secondHoverElement = driver.findElement(By.id(“second"));

hoverAction.moveToElement(firstHoverElement).moveToElement(second
HoverElement) .click().build().perform();
```

- Drag and drop:

```
WebElement locationElement = driver.findElement(By.id(“start"));

WebElement destinationElement = driver.findElement(By.id(“end"));

Actions dnd= new Actions(driver);

dnd.dragAndDrop(locationElement,
destinationElement).build().perform();
```

- גלילה לאלמנט מסוים:

```
WebElement element = driver.findElement(By.id("id"));

((JavascriptExecutor)
driver).executeScript("arguments[0].scrollIntoView(true);", element);
```

- העלאת קובץ:

```
driver.findElement(By.id(“browseButton")).sendKeys("<absolutePathToMyFile>");
```

- לחיצה על כפתור מסוים במקלדת (לדוגמא על אנטר):

```
driver.findElement(By.id("Value")).sendKeys(Keys. Enter);
```

ועוד הרבה..
https://seleniumhq.github.io/selenium/docs/api/java/org/openqa/selenium/interactions/Actions.html

תזכורת ממה שעשינו בכיתה:

1. ImplicitWait
2. Explicit wait
3. Thread.sleep
4. Ngwebdriver initialization
5. Finding Angular element
6. Using Fluent wait

```
import org.junit.BeforeClass;
import org.junit.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.testng.annotations.AfterClass;

public class SyncExample {

    public static ChromeDriver driver;

    @BeforeClass
    public static void openBrowser() {

        System.setProperty("webdriver.chrome.driver", "C:\\Users\\Daniel\\Desktop\\My
Course\\drivers and apps\\chromedriver_win32\\chromedriver.exe");
        driver = new ChromeDriver();
         new  NgWebDriver((JavascriptExecutor)driver);   4

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

        //  Implicit Wait
//        driver.manage().timeouts().implicitlyWait(1, TimeUnit.MINUTES);
        driver.navigate().to("https://www.youtube.com");
    }


    @Test
    public void syncronizationTest() {
2       //  Explicit Wait
        WebDriverWait wait = new WebDriverWait(driver, 10);
        WebElement guideElement = driver.findElement(By.id("guide-icon"));
        wait.until(ExpectedConditions.visibilityOf(guideElement));
        guideElement.click();

    }

3   @Test
    public void sleepTest() throws InterruptedException {
    //  sleep wait
    Thread.sleep(5000);
    WebElement guideElement = driver.findElement(By.id("guide-icon"));
    guideElement.click();


driver.findElement(ByAngular.exactRepeater("foo in foos"));   5


}

@Test
public void run(){

    WebElement e = driver.findElement(By.id("123"));   6

    new FluentWait<WebElement>(e).
            withTimeout(10, TimeUnit.SECONDS).
            pollingEvery(100,TimeUnit.MILLISECONDS).

            until(new Function<WebElement  , Boolean>() {
                    @Override
```