

## Sendkeys : Click : Clear : Submit ~ Element Input Operations

### Table of content

- [SendKeys in selenium](#)
- [Handle button with sendkeys](#)
- [What is CharSequence in Sendkeys?](#)
- [... in 'CharSequence... arg0'](#)
- [Experiment with sendkeys in selenium](#)
- [Keys.keys with Sendkeys in Webdriver](#)
- [clear\(\) method](#)
- [click\(\) method in webdriver](#)
- [How to write a custom click method in selenium?](#)
- [Reasons for click\(\) method to fail in selenium](#)
- [Submit](#)

## SendKeys in selenium

The **sendKeys()** method used to pass the Keyboard keys or text into editable elements (text bar, text area, button) without replacing the previously available content.

Using multiple sendkeys to a particular field, selenium appends all of the text one by one (example below).

Parameter(s) Accepted : sendKeys(CharSequence... keysToSend )

Return Type : void

### Complete Program for Sendkeys

```
public class Textbar {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // set value to textbar
        driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys("selenium webdriver");
    }
}
```

If we use Multiple **sendkeys** to a particular field, **selenium** appends them one by one to the field.

```
public class TextbarTwoSendkeys {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys("selenium webdriver");
        //send text to the same textbar and notice the change | text appended
        driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys("selenium webdriver");
    }
}
```

We can clear any editable field using clear() method present in selenium, most of the time clear() will be used along with sendkeys().

```
public class TextbarClearSendkeys {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // set value to textbar
        driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys("selenium webdriver");

        // clear() method clears the content of a editable item
        driver.findElement(By.xpath("//input[@id='textbar']")).clear();
    }
}
```

```

driver.findElement(By.xpath("//input[@id='textbar']")).clear();
//send text to the same textbar and notice the change
driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys("selenium webdriver");
}
}

```

Capture screenshot in selenium

## Handle button with sendkeys

There is a misconception among most of the testers that we can use send keys only for Textbar, TextAreas, but in reality, we can use sendKeys to handle button actions like clicking operation by sending enter key from the keyboard with Keys.keys

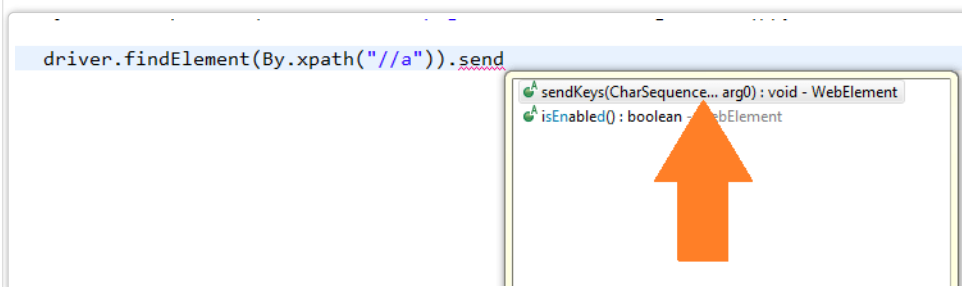
```

public class ButtonSendkeys {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // send enter key to button
        driver.findElement(By.xpath("//input[@id='btn-sendkeys']")).sendKeys(Keys.ENTER);
    }
}

```

## What is CharSequence in Sendkeys?

Have you ever noticed that the Sendkeys method accepts CharSequence... arg0 parameter, but what we pass to sendkeys is a String.

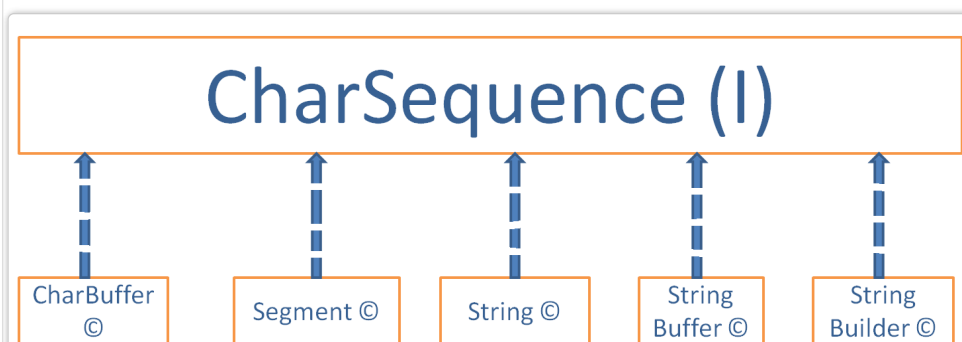


CharSequence : CharSequence is an interface that represents a sequence of characters (topmost class in string group ). Mutability is not enforced by this interface. Therefore, both mutable (**StringBuffer**, **StringBuilder**) and immutable (String) classes implement this interface.

As CharSequence is the topmost interface, we can cast all the subclasses into this type; this is the reason behind sendkeys accepting String. When we pass a String parameter, it will be automatically upcasted into CharSequence.

String - Immutable (leaves dead objects when we reassign the value)

StringBuilder and StringBuffer - Mutable (we can change the value without leaving dead object)



```
CharSequence csString = new String("chercher.tech");
```

```
CharSequence csStrBuffer = new StringBuffer("seleniumhq.org");
CharSequence csStrBuilder = new StringBuilder("testng.org");
```

In the above line **auto-upcasting** occurs, which similar to our Browser classes casting into WebDriver Interface.

**String** : String is a sequence of characters in Java. It is an immutable class and one of the most frequently used types in Java.

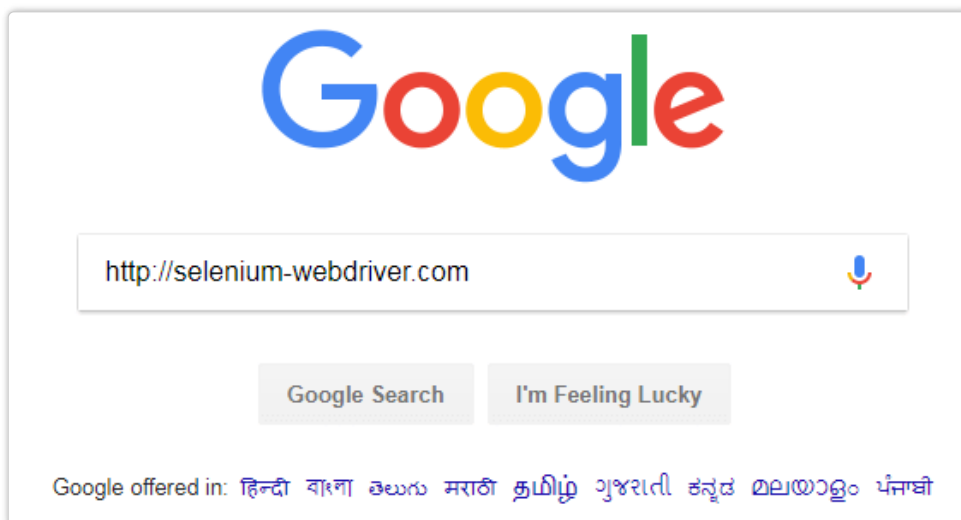
**Usage of CharSequence** : When we are not sure which kind of values is going to come and if we want to accept everything we would be using Object class, similarly when we want to limit the inputs only to String types (**String, StringBuilder, StringBuffer**) we can use CharSequence.

```
import org.testng.annotations.Test;

public class CharSequenceTest {
    @Test
    public void test(){
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://google.com");

        // define all the strings
        CharSequence csString = new String("https://");
        CharSequence csStrBuffer = new StringBuffer("selenium-");
        CharSequence csStrBuilder = new StringBuilder("webdriver.");
        CharSequence csString2 = "com";

        // send the string to search bar without clearing
        driver.findElement(By.name("q")).sendKeys(csString);
        driver.findElement(By.name("q")).sendKeys(csStrBuffer);
        driver.findElement(By.name("q")).sendKeys(csStrBuilder);
        driver.findElement(By.name("q")).sendKeys(csString2);
    }
}
```



The above program entered all the values into the search box of Google without clearing the field. So we can pass StringBuilder, StringBuffer instead of string in the **sendKeys()** method of webdriver

[Selenium Tricky Interview questions](#)

... in 'CharSequence... arg0'

What is the ... present in the Sendkeys() along with CharSequence, this symbol is known as Ellipsis in java.

Ellipsis denotes that this method accepts Zero or more parameter of the same type, Such methods can have a variable number of arguments and thus useful for the passing of dynamic data.

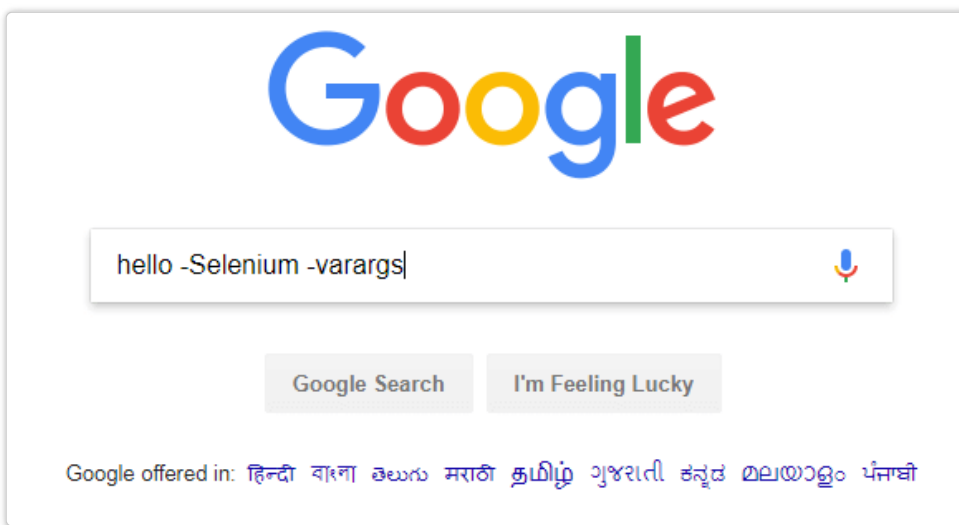
Using the varargs, we can avoid writing the overload methods (overload methods are nothing but the same methods accepting more number of parameters or different type of parameters in the same java class).

**sendKeys()** method in selenium accepts the varargs as a parameter, which means we can pass more than one string at a time, have you ever tried ?. Let's try.

```
import org.testng.annotations.Test;

public class VarargsTest {

    @Test
    public void test(){
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://google.com");
        // send multiple string in send keys
        driver.findElement(By.name("q")).sendKeys("hello -", "Selenium -", "varargs");
    }
}
```



If you have noticed the above program entered all three strings into the Search box.

Below code snippet gives details about how the Varargs internally works along with sendKeys, this is my assumption, and it could differ from the actual sendkeys() method in **webdriver**.

```
static String asOneValue(CharSequence... args) {
    // create object for StringBuilder
    StringBuilder totalString = new StringBuilder();
    // iterate through the input values
    for(CharSequence arg : args) {
        // append the total value to the total string
        totalString.append(arg);
    }
    // return the total string

    return totalString.toString();
}
```

## Experiment with sendkeys in selenium

This is not a big experiment but probably an idea, what happens when we pass an array of strings to sendkeys ? or What happens when you pass element sendkeys('string1', "string2", "string3");

Sendkeys method will add all the strings by iterating them and forms into one single string using StringBuilder and sends this value into a field in webdriver.

```
public class StringArraySendkeys {
```

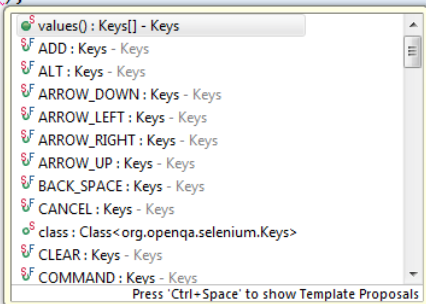
```
@Test
public void test(){
    System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
    WebDriver driver = new ChromeDriver();
    driver.get("https://google.com");
    // strings
    String[] strings = {"str1", "str2", "str3"};
    driver.findElement(By.name("q")).sendKeys(strings);
}
}
```

Create Headless Firefox Browser in webdriver

## Keys.keys with Sendkeys in Webdriver

We can stimulate the Keyboard using selenium with the help of Keys Enum; Keys Enum also implements CharSequence.

Fields (variable or keys) Present in the Keys Enum represents pressable keys which are not text, for example : a-z are text, but Shift, Ctrl, Alt are pressable keys but not text.



The screenshot shows an IDE with the code `driver.findElement(By.name("q")).sendKeys(Keys,);`. A dropdown menu is open, listing various keys from the `Keys` enum, including `ADD`, `ALT`, `ARROW_DOWN`, `ARROW_LEFT`, `ARROW_RIGHT`, `ARROW_UP`, `BACK_SPACE`, `CANCEL`, `class`, `CLEAR`, and `COMMAND`. The dropdown also shows the class `org.openqa.selenium.Keys` and a hint to 'Press Ctrl+Space to show Template Proposals'.

The below program presses 1 from the Numpad and uE01C, which is nothing but the Unicode value of number 2 from the Numpad.

```
public class KeysSendkeys {
    @Test
    public void test(){
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://google.com");
        // press number 1
        driver.findElement(By.name("q")).sendKeys(Keys.NUMPAD1);
        // press number 2 key
        driver.findElement(By.name("q")).sendKeys("uE01C");
    }
}
```

Output :



12

Every key represents a Unicode value, Instead of **Keys.value** we can also pass unicode to sendkeys method in selenium. Few are given below.

NULL	('����')	ESCAPE	('����')
CANCEL	('������')	SPACE	('������')
HELP	('��������')	PAGE_UP	('��������')
BACK_SPACE	('��������')	PAGE_DOWN	('��������')
TAB	('��������')	END	('����������')
CLEAR	('����������')	HOME	('������������')
RETURN	('������������')	LEFT	('������������')
ENTER	('��������������')	ARROW_LEFT	(Keys.LEFT)
SHIFT	('��������������')	UP	('��������������')
LEFT_SHIFT	(Keys.SHIFT)	ARROW_UP	(Keys.UP)
CONTROL	('��������������')	RIGHT	('��������������')
LEFT_CONTROL	(Keys.CONTROL)	ARROW_RIGHT	(Keys.RIGHT)
ALT	('����������������')	DOWN	('����������������')
LEFT_ALT	(Keys.ALT)	ARROW_DOWN	(Keys.DOWN)
PAUSE	('������������������')	INSERT	('������������������')
SEMICOLON	('��������������������')	EQUALS	('��������������������')

// Number pad keys

NUMPAD0	('��������������������')	NUMPAD8	('��������������������')
NUMPAD1	('��������������������')	NUMPAD9	('��������������������')
NUMPAD2	('��������������������')	MULTIPLY	('��������������������')
NUMPAD3	('��������������������')	ADD	('��������������������')
NUMPAD4	('��������������������')	SEPARATOR	('��������������������')
NUMPAD5	('��������������������')	SUBTRACT	('��������������������')
NUMPAD6	('��������������������')	DECIMAL	('��������������������')
NUMPAD7	('��������������������')	DIVIDE	('��������������������')

// Function keys

F1	('��������������������')	F7	('��������������������')
F2	('��������������������')	F8	('��������������������')
F3	('��������������������')	F9	('��������������������')
F4	('��������������������')	F10	('��������������������')
F5	('��������������������')	F11	('��������������������')
F6	('��������������������')	F12	('��������������������')

META ('                    ');  
ZENKAKU\_HANKAKU ('                    ');  
COMMAND (Keys.META);

Fluent wait in webdriver

## Multiple Keys with Keys.chord in SendKeys method

We can send multiple keys using sendkeys() in webdriver, for example if we want press shift button and enter some text to a field

```
public class SendMultipleKeys {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // enter text with capital letters by pressing Shift
        driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys(Keys.chord(Keys.SHIFT, "HELLO"));
    }
}
```

```

        sendKeys(Keys.chord(Keys.SHIFT, "UserName"));
    }
}

```

## clear() method

We can clear any editable field using **clear()** method present in webdriver, most of the time **clear()** will be used along with **sendKeys()**.

```

public class TextbarClearSendkeys {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // set value to textbar
        driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys("selenium webdriver");
        // clear() method clears the content of a editable item
        driver.findElement(By.xpath("//input[@id='textbar']")).clear();
        //send text to the same textbar and notice the change
        driver.findElement(By.xpath("//input[@id='textbar']")).sendKeys("selenium webdriver");
    }
}

```

Add Screenshots to the PDF file in selenium

## click() method in webdriver

Web page testing happens with clicks most of the time, selenium provides a method called click() which helps the user to click on the elements on a webpage.

To **click** any element on webpage selenium must **find the element** first, an element should have **width and height more than '0px'**.

The Element must be visible on the webpage.

```

public class Link {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // clicks the link which has id = 'button'
        driver.findElement(By.xpath("//a[@id='link']")).click();
    }
}

```

To click a button

```

public class Button {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // clicks the button which has id = 'button'
        driver.findElement(By.id("button")).click();
    }
}

```

We can use click() method to click the checkbox and uncheck the checkbox

```

public class Checkbox {

    public static void main(String[] args) {

```

```
// set chrome driver exe path
System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
WebDriver driver = new ChromeDriver();
driver.get("https://chercher.tech/selenium-webdriver-sample");
// clicks the check which has id = 'checkbox'
driver.findElement(By.id("checkbox")).click();
}
}
```

We can use click() method to click the radio button, as we know we cannot unselect a radio button by clicking it. To un-select a radio button we need to click another radio button.

```
public class Radio {
    public static void main(String[] args) {
        // set chrome driver exe path
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");
        WebDriver driver = new ChromeDriver();
        driver.get("https://chercher.tech/selenium-webdriver-sample");
        // clicks the radio button which has id = 'radio'
        driver.findElement(By.id("radio")).click();
    }
}
```

#### Type Casting in Method Overloading in Java Selenium

## How to write a custom click method in selenium?

Sometimes we may need to write our own methods to click so that before click or after click we can perform some operations, also we will have the clicking operation happening in one place.

In case if there is a change in click() method of **webdriver**, we can easily modify or update the method in one place instead of multiple places.

```
public class CustomClick {
    // open firefox and store it in driver and driver is static here (driver not webdriver)
    public static WebDriver driver = new FirefoxDriver();
    public static void main(String[] args) {
        driver.get("https://chercher.tech/java/explicit-wait-sample");
        WebElement button = driver.findElement(By.xpath("//button[@id='disable']"));
        customClick(button);
    }
    public static void customClick(WebElement element){
        System.out.println("you can write the custom login/ or print statements");
        element.click();
        System.out.println("you can write the custom login/ or print statements");
    }
}
```

#### Xpath in Selenium

## Reasons for click() method to fail in selenium

*There no such element*

*Element is not visible*

*Element Not clickable*

*Element Height and width is less than 0 (zero)*

*Check the xpath case(upper and lower) few it depends on the browser*

#### Exceptions in Selenium



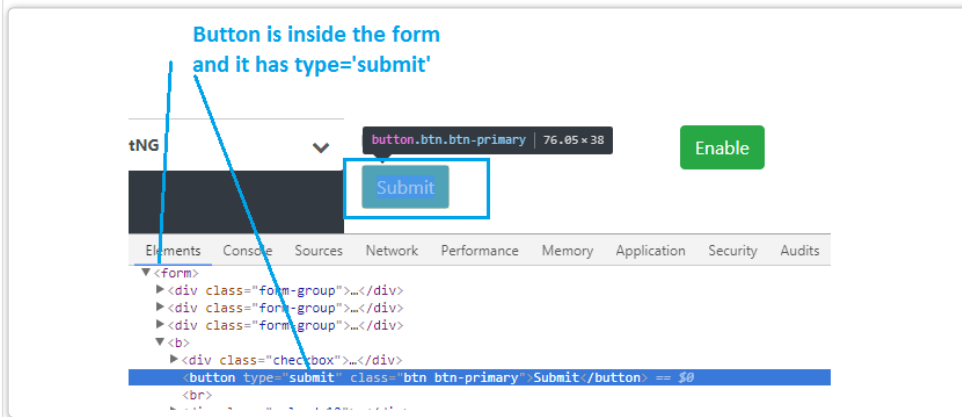
## Submit

**submit()** method in webdriver submits a form, user can use it only along with form or with form element only.

**submit()** method clicks a button is a misconception, it works only when that particular button is associated with the form.

**@parameters :** accepts nothing

**@return** : nothing



```
public class Submit {  
    public static void main(String[] args) {  
        // set chrome driver exe path  
        System.setProperty("webdriver.chrome.driver", "C:/~/chromedriver.exe");  
        WebDriver driver = new ChromeDriver();  
        driver.get("https://chercher.tech/selenium-webdriver-sample");  
        // submits the button which has type as submit  
        driver.findElement(By.id("//button[@type='submit']")).submit();  
    }  
}
```

## Ways to select a dropdown option in selenium

### Recommended Readings

## Strings in Java & Selenium

## Sleep in Selenium

## Page & Script Load Timeout in Selenium

## Implicit Wait in Selenium

## Find Element / Find Elements in Selenium

## Locators in Selenium

## Try Xpath Add on to FireFox 57+ [FireBug Replacement]

## FireBug in Selenium Webdriver