**Selenium Class**

# Chapter#15 - Keys Board Actions

During automation, we are often required to press enter, control, tab, arrow keys, function keys, and other non-text keys as well from the keyboard. In this chapter, we will find how to simulate the pressing of these non-text keys using Selenium WebDriver in Java. Here, we will be using the Keys enum provided by Selenium WebDriver for all the non-text keys.

1. **Press Enter/Return Key in Selenium**
2. **Actions Class Method for Keyboard Interaction**

## Press Enter/Return Key in Selenium

For pressing Enter key over a textbox we can pass Keys.ENTER or Keys.RETURN to the sendKeys method for that textbox.

WebElement textbox = driver.findElement(By.id("idOfElement"));

textbox.sendKeys(Keys.ENTER);

WebElement textbox = driver.findElement(By.id("idOfElement"));

textbox.sendKeys(Keys.RETURN);

|  |  |
| --- | --- |
| **Keyboard’s Key** | **Keys enum’s value** |
| Arrow Key – Down | Keys.ARROW\_DOWN |
| Arrow Key – Up | Keys.ARROW\_UP |
| Arrow Key – Left | Keys.ARROW\_LEFT |
| Arrow Key – Right | Keys.ARROW\_RIGHT |
| Backspace | Keys.BACK\_SPACE |
| Ctrl Key | Keys.CONTROL |
| Alt key | Keys.ALT |
| DELETE | Keys.DELETE |
| Enter Key | Keys.ENTER |
| Shift Key | Keys.SHIFT |
| Spacebar | Keys.SPACE |
| Tab Key | Keys.TAB |
| Equals Key | Keys.EQUALS |
| Esc Key | Keys.ESCAPE |
| Home Key | Keys.HOME |
| Insert Key | Keys.INSERT |
| PgUp Key | Keys.PAGE\_UP |
| PgDn Key | Keys.PAGE\_DOWN |
| Function Key F1 | Keys.F1 |
| Function Key F2 | Keys.F2 |
| Function Key F3 | Keys.F3 |
| Function Key F4 | Keys.F4 |
| Function Key F5 | Keys.F5 |
| Function Key F6 | Keys.F6 |
| Function Key F7 | Keys.F7 |
| Function Key F8 | Keys.F8 |
| Function Key F9 | Keys.F9 |
| Function Key F10 | Keys.F10 |
| Function Key F11 | Keys.F11 |
| Function Key F12 | Keys.F12 |

## Actions Class Method for Keyboard Interaction

**keyDown(Keys modifierKey)**

The keyDown(Keys modifierKey) method takes the modifier Keys as parameter (Shift, Alt and Control Keys – that modifies the purpose of other keys, hence the name). It is used to simulate the action of pressing a modifier key, without releasing. The expected values for the keyDown() method are – Keys.SHIFT, Keys.ALT and Keys.CONTROL only, passing key other than these results in IllegalArgumentException.

**keyDown(WebElement element, Keys modifierKey)**

This another implementation of keyDown() method in which the modifier key press action is performed on a WebElement.

**keyUp(Keys modifierKey)**

The keyUp() method is used to simulate the modifier key-up or key-release action. This method follows a preceeding key press action.

**keyUp(WebElement element, Keys modifierKey)**

This implementation of keyUp() method performs the key-release action on a web element.

**sendKeys(CharSequence KeysToSend)**

The sendKeys(CharSequence KeysToSend) method is used to send a sequence of keys to a currently focussed web element. Here, we need to note that it is different from the webElement.sendKeys() method. The Actions sendKeys(CharSequence KeysToSend) is particularly helpful when dealing with modifier keys as it doesn’t release those keys when passed(resulting in correct behaviour) unlike the webElement.sendKeys() method.

**sendKeys(WebElement element, CharSequence KeysToSend)**

This implementation of sendKeys() method is used to send a sequence of keys to a web element.

**Code snippet for Keyboard Actions**

**//WebElement to which the keyboard actions are performed**

WebElement textBoxElement = driver.findElement(By Locator of textBoxElement);

**//Creating object of Actions class**

Actions builder = new Actions(driver);

**//Generating an action to type a text in CAPS**

Action typeInCAPS = builder.keyDown(textBoxElement, Keys.SHIFT)

.sendKeys(textBoxElement, "artOfTesting")

.keyUp(textBoxElement, Keys.SHIFT)

.build();

**//Performing the typeInCAPS action**

typeInCAPS.perform();