Pressing Non-text Keys in selenium:

During automation, we are often required to press enter, control, tab, arrow keys, function keys and other non-text keys as well from keyboard.

Ways To Handle Keyboard Keys:

1. Handle Keyboard Keys using Action class.
2. Handle Keyboard Keys using sendkeys & chord.
3. Handle Keyboard keys using Robot class.

We use an enum called "Keys". (`import org.openqa.selenium.Keys;`)

1. *element.sendKeys(Keys.ENTER);*
2. *action.sendKeys(Keys.ENTER).perform();*
   //whenever we use action class we need to end statement with .perform() method or build().perform().

```
// Actions action = new Actions(driver);
```

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);
```

or

```
WebElement textbox = driver.findElement(By.id("idOfElement"));
textbox.sendKeys(Keys.RETURN);
```

Similarly, we can use Keys enum for different non-text keys and pass them to the sendKeys method. The following table has an entry for each of the non-text key present in a keyboard.

Keyboard's Key	Keys enum's value
Arrow Key - Down	Keys.ARROW_DOWN
Arrow Key - Up	Keys.ARROW_LEFT
Arrow Key - Left	Keys.ARROW_RIGHT
Arrow Key - Right	Keys.ARROW_UP
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 F12	Keys.F12

Scrolling An element without using "Java script executor":

Using action "Actions" class

The **Page Up and Page Down keys** (sometimes abbreviated as **PgUp** and **PgDn**) are two keys commonly found on computer keyboards.

The two keys are primarily used to scroll up or down in documents, but the scrolling distance varies between different applications.

The arrow keys (down arrow, left arrow) and the scroll wheel can also be used to scroll a document, although usually by smaller incremental distances.

Page scroll down

```
action.sendKeys(Keys.PAGE_DOWN).build().perform();
```

Page scroll up

```
action.sendKeys(Keys.PAGE_UP).build().perform();
```

Note-For Horizontal (and vertical also) scroll on the web page. We can use below method of Java script executer:

```
        //This will scroll the page Horizontally till the element is found
        js.executeScript("arguments[0].scrollIntoView();", element);

or Using action class as below:

action.sendKeys(Keys.ARROW_LEFT).build().perform();
```

```
action.sendKeys(Keys.ARROW_RIGHT).build().perform();
```

Pressing Multiple keys together:

Mouse Event: https://www.guru99.com/keyboard-mouse-events-files-webdriver.html

1. Using Actions Class:

```
Actions action = new Actions(driver);
action.keyDown(Keys.CONTROL).sendKeys("a").keyUp(Keys.CONTROL).perform();
```

Note-

- keyDown is press & hold command and its released by keyUp.
- when we use action.sendKeys("a") it presses the key "a" and immediately releases it.

Parameters: for keyDown/keyUp method

key Either `Keys.SHIFT`, `Keys.ALT` or `Keys.CONTROL`. If the provided key is none of those, `IllegalArgumentException` is thrown.

2. Using SendKeys Chord:

```
driver.findElement(By.xpath("//body")).sendKeys(Keys.chord(Keys.CONTROL, "a"));
```

3. Using Robot Class:

```
Robot rb = new Robot();
rb.keyPress(KeyEvent.VK_CONTROL);
rb.keyPress(KeyEvent.VK_A);
```

Press ctrl+shif+s in the given textbox and verify msg displayed

Links: https://jsfiddle.net/39850x27/2/

Press a key on the keyboard in the input field to find out if the Ctrl-SHIFT key was pressed or not.

The Ctrl-SHIFT-S keys were pressed!

```
// move a cursor to the field
   wait.until(ExpectedConditions.elementToBeClickable(inputField)).click();

   Actions a = new Actions(driver);

   // Press SHIFT-CTRL-S
   a.keyDown(Keys.SHIFT)
    .keyDown(Keys.CONTROL)
    .sendKeys("s")
    .build()
    .perform();


  //Wait for a message
 wait.until(ExpectedConditions.visibilityOfElementLocated(messageWeWaitFor));

   System.err.println("Success - Ctrl-Shift-S were pressed !!!");

   // Sleep some time (to see the message is really on the page)
   Thread.sleep(5000);

   // Release SHIFT+CTRL keys
   a.keyUp(Keys.CONTROL)
    .keyUp(Keys.SHIFT)
    .build()
    .perform();
```

In one go we can do it like this:

```
   a.keyDown(Keys.SHIFT)
    .keyDown(Keys.CONTROL)
    .sendKeys("s")
    .keyUp(Keys.CONTROL)
    .keyUp(Keys.SHIFT)
    .build()
    .perform();
```

Modifier Key:

In computing, a modifier key is a special key on a computer keyboard that temporarily modifies the normal action of another key when pressed together. By themselves, modifier keys usually do nothing; that is, pressing any of the ⇧ Shift, Alt, or Ctrl keys alone does not trigger any action from the computer.

- Shift+a to type “A”
- Ctrl+a to select all
- Ctrl+c to copy
- Ctrl+v to paste

1. Handle Keyboard Keys using Action class:

Actions Class Method for Keyboard:

keyDown and keyUp are the main methods in KeyBoard Events in Selenium Webdriver Actions class.

- **public Actions keyDown(Keys theKey) :** Performs a modifier key press (SHIFT,Keys.ALT or Keys.CONTROL) to Handle keyDown operation.
- **public Actions keyDown(WebElement element, Keys theKey) :** Performs a modifier key press (SHIFT,Keys.ALT or Keys.CONTROL) after focusing on an element perform keyDown using WebElement.
- **public Actions keyUp(Keys theKey) :** Performs a modifier key release (SHIFT,Keys.ALT or Keys.CONTROL) to Handle keyUp operation.
- **public Actions keyUp(WebElement element, Keys theKey) :** performs a modifier key release after focusing on an element to perform keyUp operation.
- **public Actions sendKeys(java.lang.CharSequence… keysToSend) :** The key sends the active element to the key, it is actively different from calling sendKeys two passes (CharSequence…) on an active element in two ways: Modifiers are not included in this call, and no one is able to focus the element again. Do not try. Then we will send some sendKeys(Keys.TAB) to switch the elements.
-

- **public Actions sendKeys(WebElement element, java.lang.CharSequence… keysToSend) :** Sends keys to the given element using sendKeys.