Frames

What is Frame or iFrame?

- An iFrame (Inline Frame) is an HTML document embedded inside the current HTML document on a website.
- <Frame> or<iFrame> HTML tag will be used to insert one html page in another html page.
- IFrame HTML element is used to insert content from another source, such as an advertisement, into a Web page.

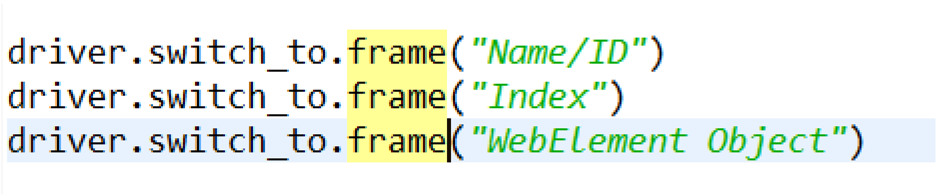
Note: A Web designer can change an iFrame content without making them reload the complete website. A website can have multiple frames on a single page. And a frame can also have inner frames (Frame inside a Frame).

- IFrame is defined by an *<iFrame></iFrame>* tag in HTML. With this tag you can identify an iFrame while inspecting the HTML tree.
- In order to embed (include) a web page inside another web page developer uses 'iFrame' or 'Frame' html tag.
- But, if an element is inside the frame then web driver cannot find the element.
- I e web driver can't perform an action on web element automatically when object or web element are inside the frame.
- In order to work with frame web elements we should pass driver control to the frame before performing an action.

Frame method is overloaded it takes one argument of type

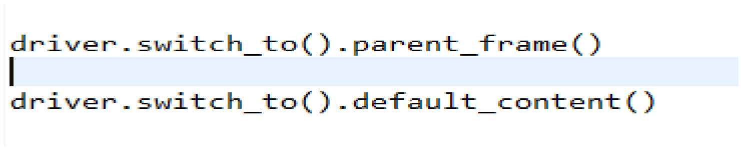
1. Int
2. String (ID/Name)
3. Web Element (Address)

There are three ways we can pass driver control to frame.



```
driver.switch_to.frame("Name/ID")
driver.switch_to.frame("Index")
driver.switch_to.frame("WebElement Object")
```

In order to give the control back to the main page, i.e. Change the control from frame or to exit from the frame we use the following methods.



```
driver.switch_to().parent_frame()

driver.switch_to().default_content()
```

Default content method is used to switch to main page directly.

Parent frame method is used to switch to its immediate parent page.

Q. How to find the frame in a web page?

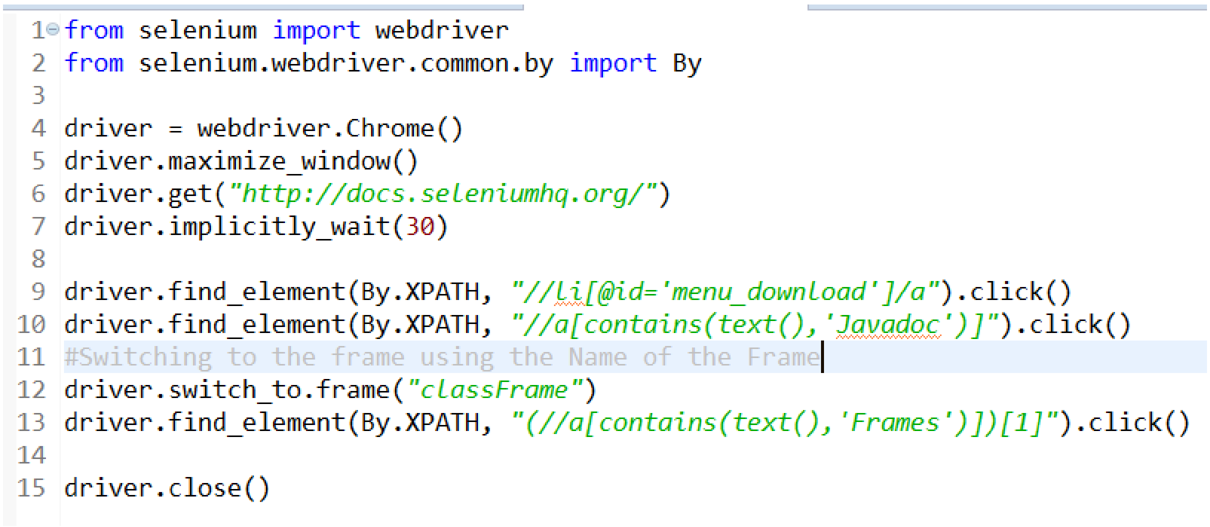
A. Inspect any element, in the search field, type “//Frame” or “//IFrame”. If any frame is present, it will show the number of frames present in the web page.

A. In order to find the object or web element is present inside the frame. Inspect the element. Move the cursor in the upside direction. We can find the tag “<frame>” or “<iFrame>”. If required element is present within that frame, than that was the frame where our element is present.

- Once after identifying the frame, we need to switch into the frame using the above mentioned methods. Then we can access the web elements present inside the frame.

Write a script to switch to FRAME using the **name** of the frame?

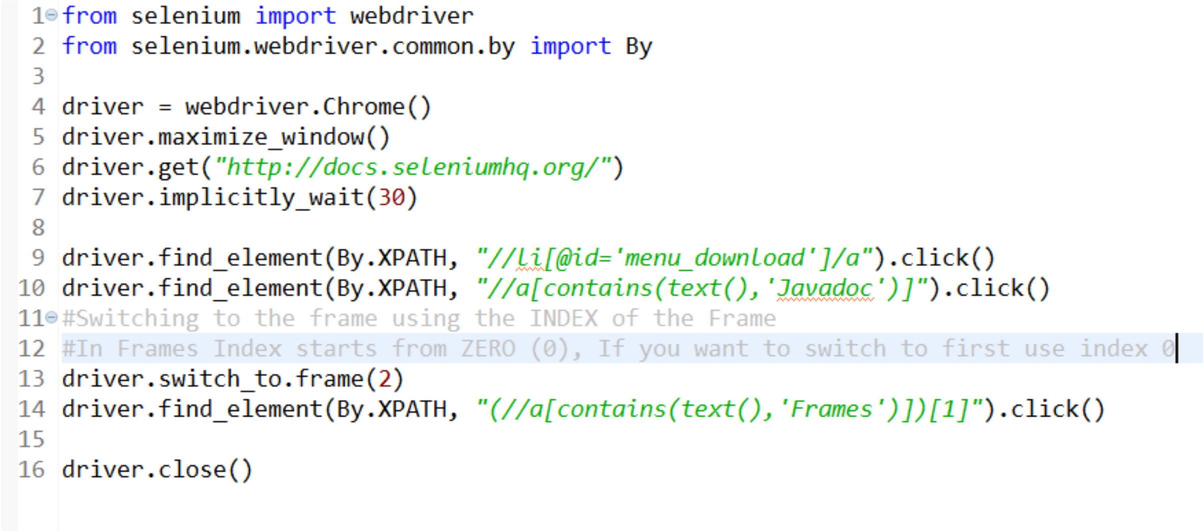
If you take a look at the HTML code of Frame tag you may find the ***Name*** *attribute*. If a frame tag contains the name or id than we can use these two attribute values directly to switch the control to frame.



```python
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()
#Switching to the frame using the Name of the Frame
driver.switch_to.frame("classFrame")
driver.find_element(By.XPATH, "(//a[contains(text(),'Frames')])[1]").click()

driver.close()
```

Write a script to switch to FRAME using the **INDEX** of the frame?

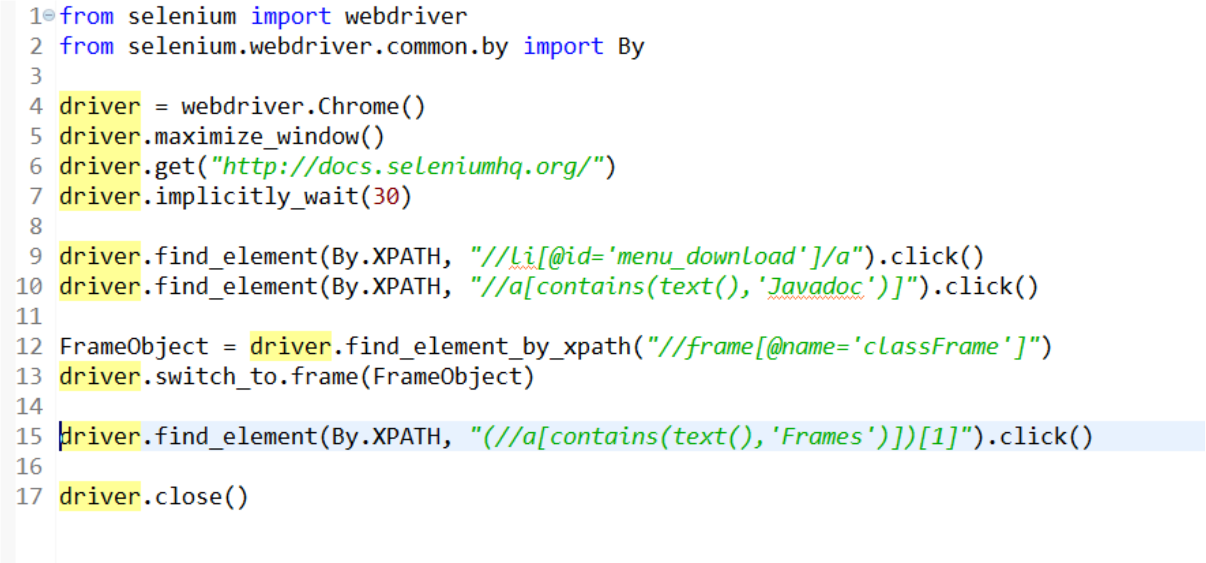
- Index of a Frame is the position at which it occurs in the HTML page. In the sample page we have three Frames, index of Frame starts from 0. So there are three Frames on the page with index 0, 1 and 2.



```python
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()
#Switching to the frame using the INDEX of the Frame
#In Frames Index starts from ZERO (0), If you want to switch to first use index 0
driver.switch_to.frame(2)
driver.find_element(By.XPATH, "(//a[contains(text(),'Frames')])[1]").click()

driver.close()
```

Write a script to switch to FRAME using the **WEB ELEMENT OBJECT** of the frame?

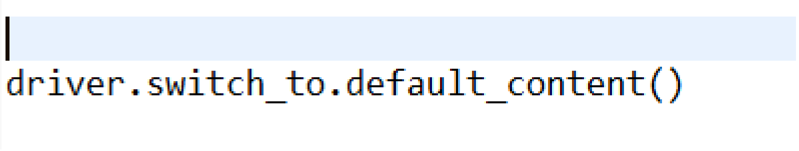
- We can switch to a Frame by simply passing the Frame Web Element to the **driver.switch_to_frame** command. First find the Frame element using any of the locator strategies and then passing it to switch_to command.



```python
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()

FrameObject = driver.find_element_by_xpath("//frame[@name='classFrame']")
driver.switch_to.frame(FrameObject)

driver.find_element(By.XPATH, "(//a[contains(text(),'Frames')])[1]").click()

driver.close()
```

Switching back to Main page from Frame

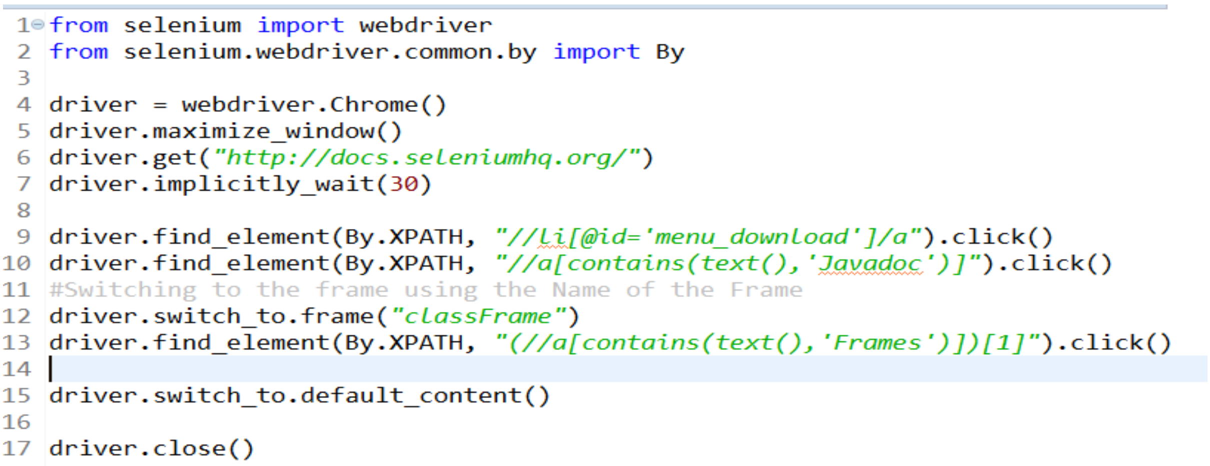
- There is one very important command that will help us to get back to the main page. Main page is the page in which all the FRAMES are embedded. Once you are done with all the task in a particular iFrame you can switch back to the main page using the below statement.



```
driver.switch_to.default_content()
```

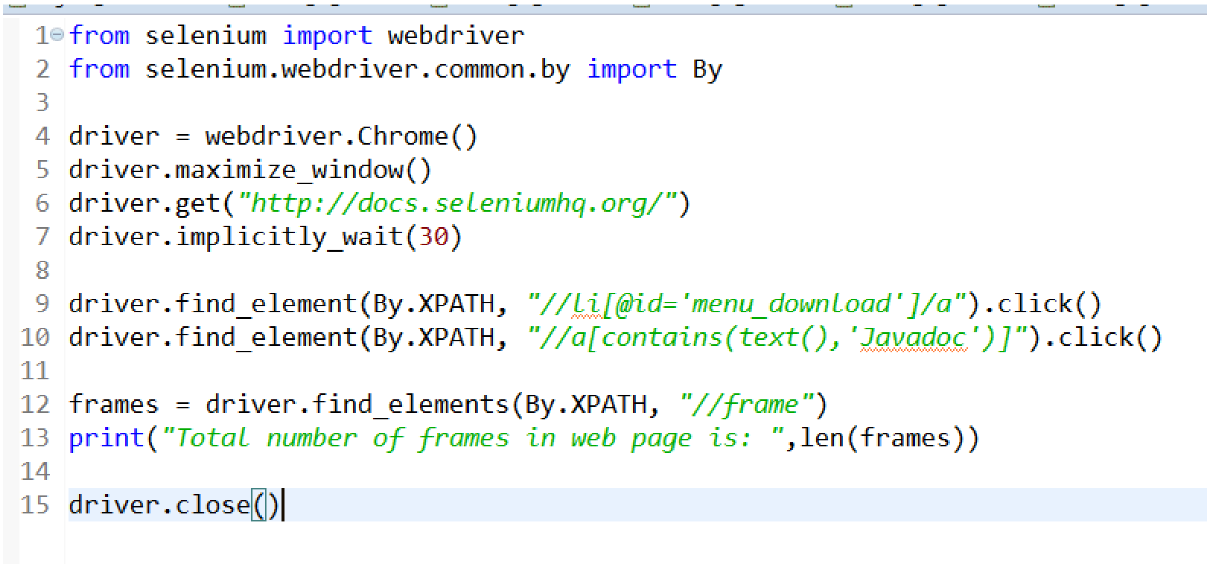
- The above statement switch the control from FRAME to main page

Write a script to switch the control to main page?



```
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()
#Switching to the frame using the Name of the Frame
driver.switch_to.frame("classFrame")
driver.find_element(By.XPATH, "(//a[contains(text(),'Frames')])[1]").click()

driver.switch_to.default_content()

driver.close()
```

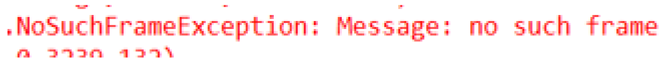
Write a script to find the total number of FRAMES in the web page?



```
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()

frames = driver.find_elements(By.XPATH, "//frame")
print("Total number of frames in web page is: ",len(frames))

driver.close()
```

Note:

- Now that we know what a Frame is and how we can switch between Frames, let's learn how to interact with elements inside an iFrame. Now will discuss in details about how to interact with elements in an iFrame.
- Now let's learn how to interact with elements inside an iFrame. Once we have switched to a particular Frame everything else after that can be done using regular WebDriver command.
- Whenever you try to switch to the frame. If the frame is not available or not loaded, we get the below exception.



```
.NoSuchFrameException: Message: no such frame
```

What is NoSuchFrameException?

- Whenever switchTo ().Frame () method fails to locate the frame in a HTML page. We get a NoSuchFrameException.

IMPORTANT: Whenever the frame page get refresh, control will automatically go back to the main page.

Now, let's create two scenario.

First scenario.

Step 1: Login to https://docs.seleniumhq.org/.

Step 2: Click on the "Download" tab.

Step 3: Click on the "Java doc" link.

Step 4: Once you are in "Overview" page.

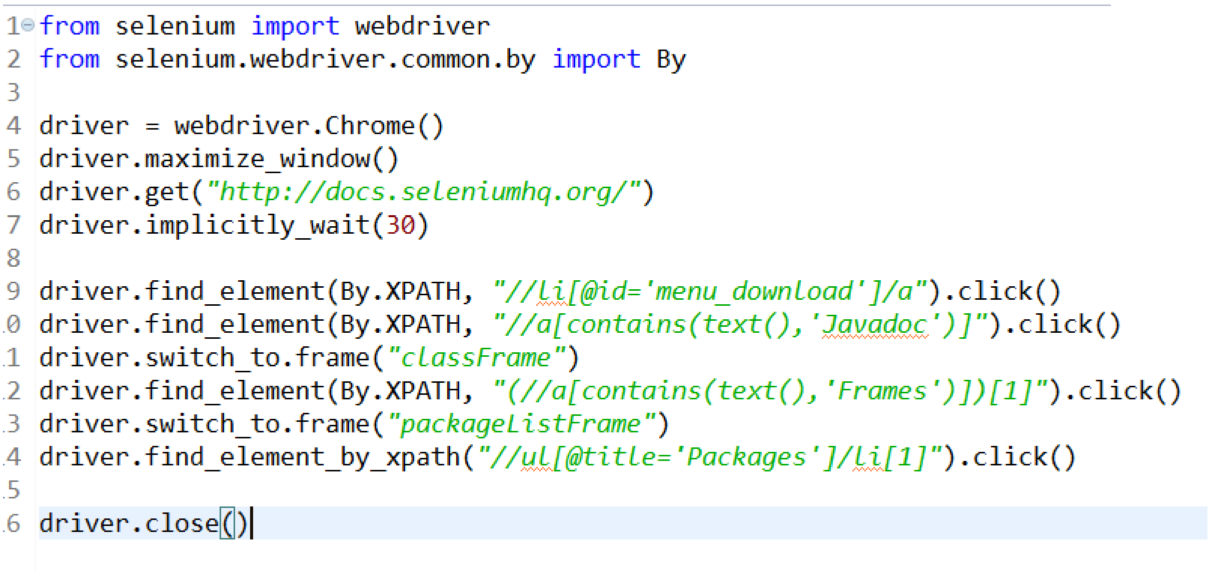
Step 5: Switch to the "ClassFrame" frame.

Step 6: Click on the "Frames" link

Step 7: Switch to "packageListFrame" frame.

Step 8: Click on the "com.thoughtworks.selenium" link

Below is the script for the above scenario.



```python
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()
driver.switch_to.frame("classFrame")
driver.find_element(By.XPATH, "(//a[contains(text(),'Frames')])[1]").click()
driver.switch_to.frame("packageListFrame")
driver.find_element_by_xpath("//ul[@title='Packages']/li[1]").click()

driver.close()
```

- Scripts executes successfully without any exception.

Second scenario.

Step 1: Login to https://docs.seleniumhq.org/.

Step 2: Click on the “Download” tab.

Step 3: Click on the “Java doc” link.

Step 4: Once you are in “Overview” page.

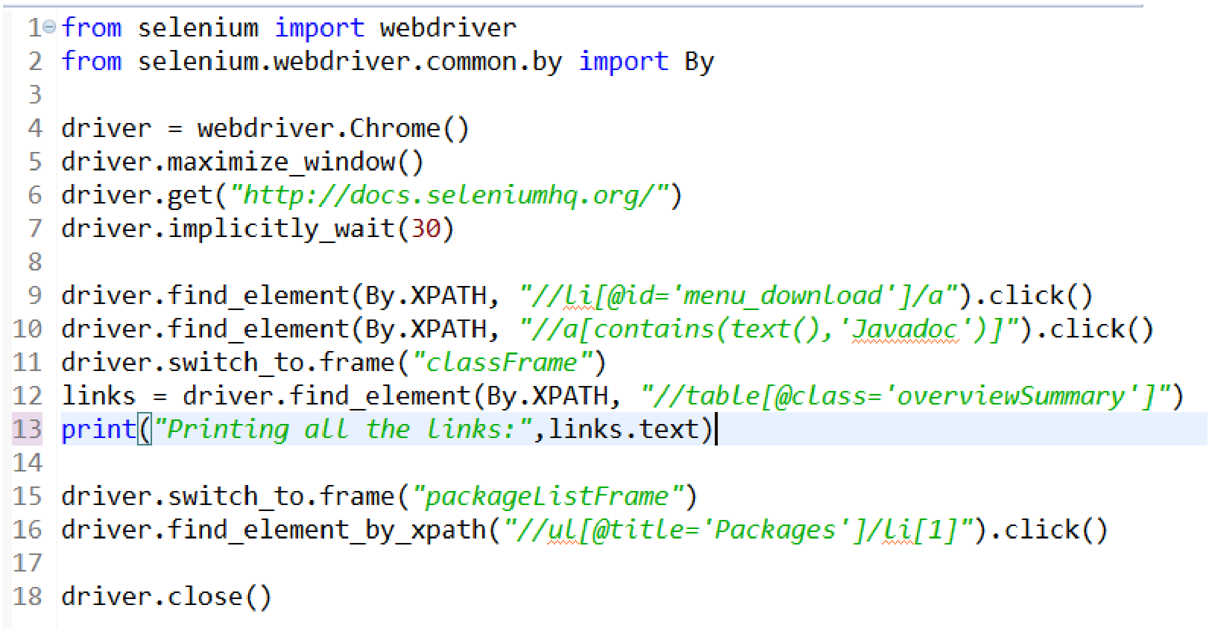
Step 5: Switch to the “ClassFrame” frame.

Step 6: Get all the links present in the frame and print the same

Step 7: Switch to “packageListFrame” frame.

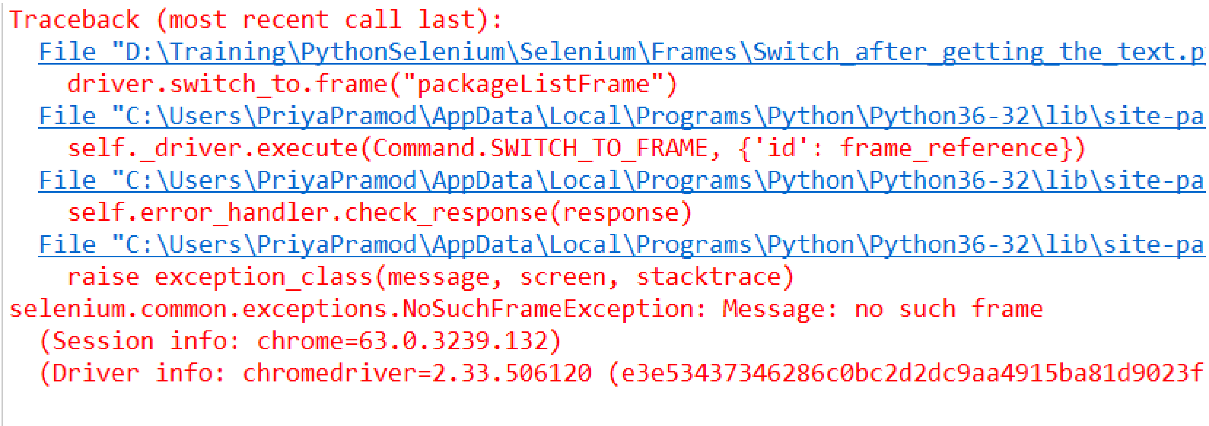
Step 8: Click on the “com.thoughtworks.selenium” link

Below is the script for the above scenario.



```python
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()
driver.switch_to.frame("classFrame")
links = driver.find_element(By.XPATH, "//table[@class='overviewSummary']")
print("Printing all the links:",links.text)

driver.switch_to.frame("packageListFrame")
driver.find_element_by_xpath("//ul[@title='Packages']/li[1]").click()

driver.close()
```

If you run the above scripts, you will get the following exception.



```
Traceback (most recent call last):
  File "D:\Training\PythonSelenium\Selenium\Frames\Switch_after_getting_the_text.p
    driver.switch_to.frame("packageListFrame")
  File "C:\Users\PriyaPramod\AppData\Local\Programs\Python\Python36-32\lib\site-pa
    self._driver.execute(Command.SWITCH_TO_FRAME, {'id': frame_reference})
  File "C:\Users\PriyaPramod\AppData\Local\Programs\Python\Python36-32\lib\site-pa
    self.error_handler.check_response(response)
  File "C:\Users\PriyaPramod\AppData\Local\Programs\Python\Python36-32\lib\site-pa
    raise exception_class(message, screen, stacktrace)
selenium.common.exceptions.NoSuchFrameException: Message: no such frame
  (Session info: chrome=63.0.3239.132)
  (Driver info: chromedriver=2.33.506120 (e3e53437346286c0bc2d2dc9aa4915ba81d9023f
```

- It gives an exception called “NoSuchFrameException”.

What is NoSuchFrameException?

- Whenever switch_to. Frame method fails to locate the frame in a HTML page. We get a NoSuchFrameException.

Note:

Now, let’s take an example of above two scenario, in both the scenario. We are following the same flow but in the first scenario we were able to run our scripts without getting any exception, but in the second scenario we are getting the exception.

Coz, once you switch to the frame, you can access the html elements that are inside the frame. Any attempt to access the elements which are inside another Frame without switching to that will result in WebDriver exception called NoSuchFrameException.

Now if you are inside frame called “ClassFrame”, now if you try to switch to another frame called “packageListFrame”, you will get the exception. Because frame “packageListFrame” is not part of the frame “ClassFrame”, but it is a part of the main page. Any attempt to directly switch to frame “packageListFrame” from frame “ClassFrame” will result in an exception.

So, after performing the actions inside the frame, we need to switch back to the main page using switch_to default_content, then if you try to switch to the frame “packageListFrame”, you can switch to

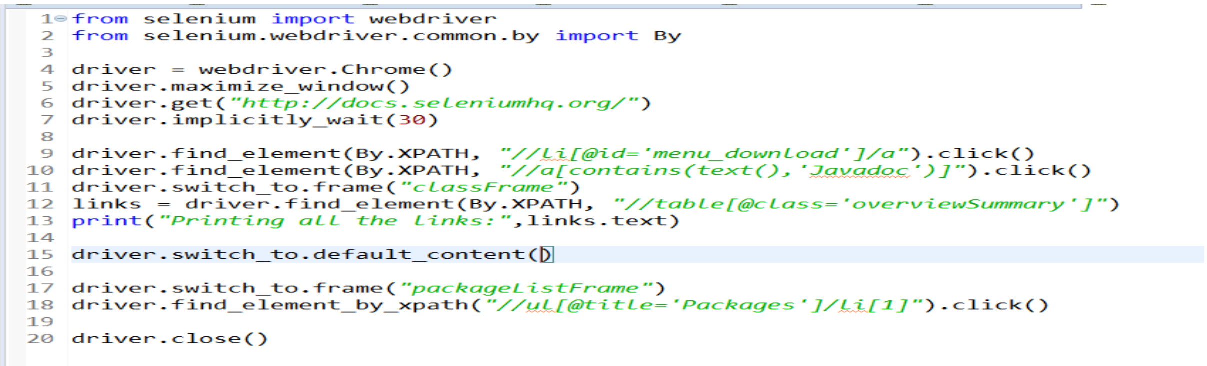
IMPORTANT: But in the first scenario, we were able to switch to the frame “packageListFrame” without going back to the main page.

Because whenever the frame page get refresh, control will automatically go back to the main page.

So only in the first scenario we are not getting the exception. Coz, once after clicking on the Frames link, frame got refreshed and control goes back to the main page.

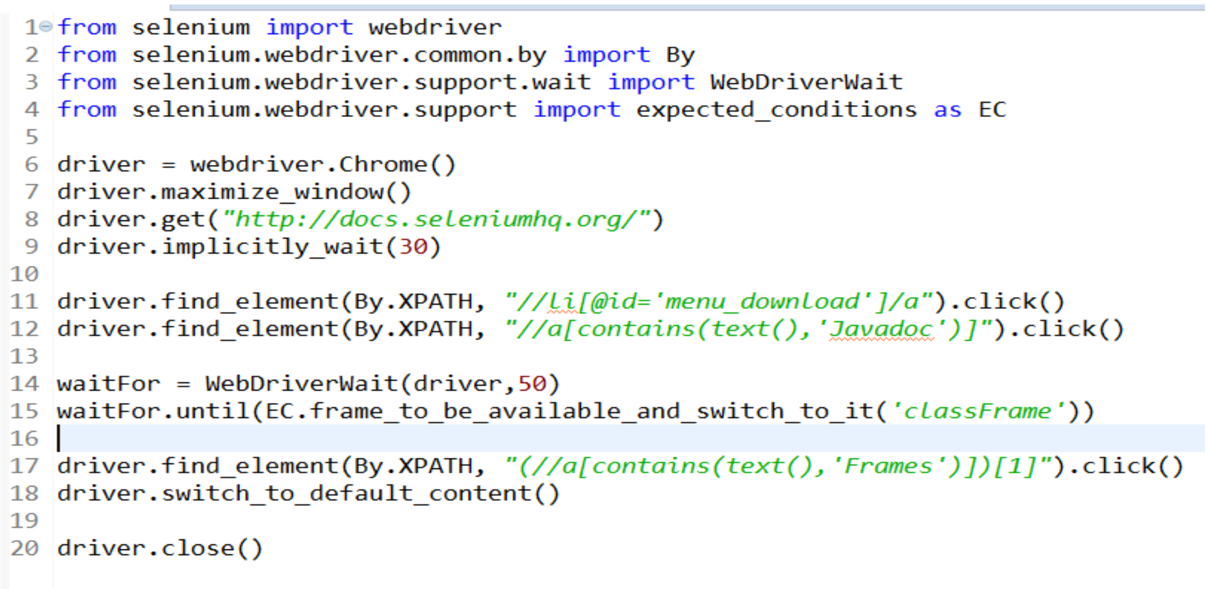
But in the second scenario, page not refreshed. If page is not refreshed means, the control still remains in the same frame. Any attempt to switch the frame without switching to main page, we end up getting the exception.

Below is the working script for the second scenario.



```
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()
driver.switch_to.frame("classFrame")
links = driver.find_element(By.XPATH, "//table[@class='overviewSummary']")
print("Printing all the links:",links.text)

driver.switch_to.default_content()

driver.switch_to.frame("packageListFrame")
driver.find_element_by_xpath("//ul[@title='Packages']/li[1]").click()

driver.close()
```

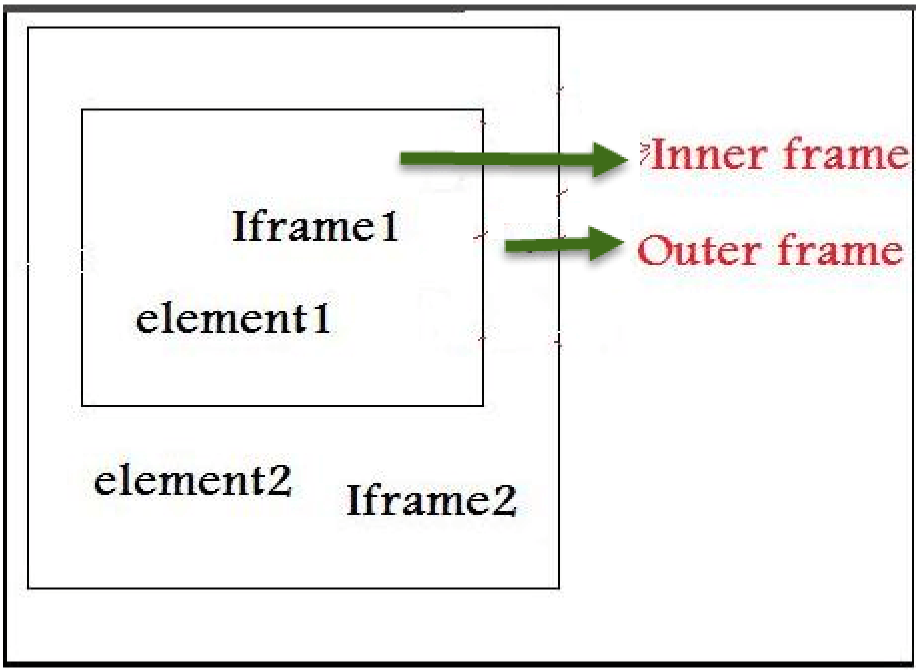
Write a script to switch to the frame using explicit wait?



```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

driver = webdriver.Chrome()
driver.maximize_window()
driver.get("http://docs.seleniumhq.org/")
driver.implicitly_wait(30)

driver.find_element(By.XPATH, "//li[@id='menu_download']/a").click()
driver.find_element(By.XPATH, "//a[contains(text(),'Javadoc')]").click()

waitFor = WebDriverWait(driver,50)
waitFor.until(EC.frame_to_be_available_and_switch_to_it('classFrame'))

driver.find_element(By.XPATH, "(//a[contains(text(),'Frames')])[1]").click()
driver.switch_to_default_content()

driver.close()
```

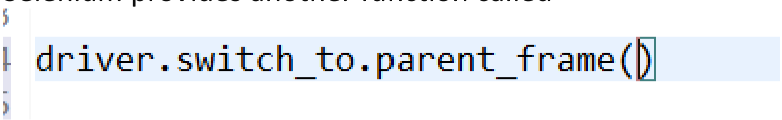
Nested Frames (Frames inside Frames):

Let's assume that there are two frames one inside other like shown in below image.



Just assume, we are in the inner frame, now I want to switch the control to outer frame, no need to switch to the main page.

Selenium provides another function called



```
driver.switch_to.parent_frame()
```

To switch focus to its parent. If the focus is in the outer frame, that time focus remains the same.