(Refer Notes for 29/12/18 before X path)

NO

DATE 05/01/2019

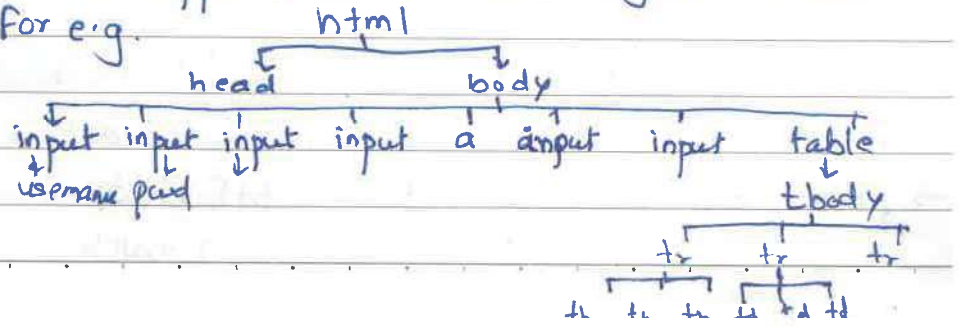
X-path :—

1) X-path is an expression, not an attribute.
2) X-path was introduced to support xml language where the xml stands for extensible markup language.
3) It is used to store the data, data is stored in the form of nodes. Nodes present in xml tree, The path of these nodes in xml tree is known as X-path.
4) X-path supports both xml & html language
5) When the elements are not getting identified with any other locators then we use x-path.
6) X-path is categorised into two →
   i) Absolute x-path
   ii) Relative x-path.

i) Absolute x-path :—

1) In absolute x-path, in order to identify any element, we need to start from the root of the application till the target element.

for e.g.



NO

DATE

Q. Write a x-path expression to identify the link present in above html structure.

→ /html/body/a → 1 match

Q. Write a x-path expression to identify username field from above html hirarchy

→ /html/body/input → 6 matches

As the above x-path will be giving multiple matches, in order to overcome that & identify username field, we will specify the position of the element in square bracket [].

For.e.g.

/html/body/input[1] → 1 match

Q. Write a x-path expression to identify a link presence in table (for reference see above html hirarchy)

→ /html/body/table/tr/td/a

/html/body/table/tbody/tr/td/a → 2 matchy

Write a xpath expression to identify a link present in 2nd row, 3rd column of the table.

→ /html/body/table/tbody/tr[2]/td[3]/a → 1 match

NO

DATE

All the given expressions are derived by using absolute x-path. We can write same expressions by using relative x-path.

ii> Relative X-path

In relative x-path, x-path path expression might start with double slash ("//") which indicates to search anywhere in the web-page, or the x-path expression might contain double slash (//) in betn which indicates to search anywhere in the decendent of current node.

For eg.

Q Write a x-path expression to identify the objects which are developed by using input tag.

→ //input → 8 matches

The above expression will give you the result from the webpage based on presence of input tag.

Q. Write a x-path expression to identify username field

→ //input[1]

NO

DATE

* //a → It will match with all the links present in entire webpage i.e. 3

* //body/a → It will give you the count of the links which are present directly inside the body tag.

Write a x-path expression to identify a link present in second row third column of the table.

→ //tr[2]/td[3]/a or
/html/body/table//tr[2]/td[3]/a

Note :—

- Weather we use absolute x-path or relative x-path, there deriving expression to identify object uniquely is difficult. In order to overcome this limitation, x-path supports an expression similar to css selector that is known as 'x-path by attribute'.
- Generally we use x-path by position along with x-path by attributes.

Syntax → //html tag [@Attribute Name = 'Attribute Value']

For e.g. if the html code is

<input id="username" type="text" name="username" Placeholder="username">

NO

DATE

Based on above html code, possible x-path expressions will be,

//input[@id='username'] // or
//input[@type='text'] // or
//input[@name='username'] // or
//input[@placeholder='Username']

In order to identify an object based on its inner text (which is getting displayed on UI), we can use text function of x-path.

Syntax ⇒ //html tag [text()='inner text'] → inner text

For e.g.

Forgot password?

//a[text()='forgot password?']
//or
//a[@class='forgotPasswordLink']

example 2 →

<button type="Submit" class="button buttonblue">Sign in</button>

//button[text()='sign in'] or
//button[@type='submit'] or
//button[@class='button buttonblue']

(Refer x-path notes on 5/01/19 before this)

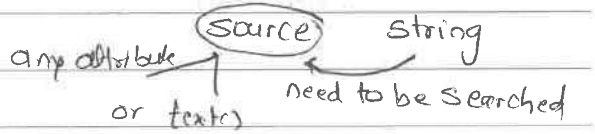
NO

DATE 06/01/2019

Contains() for x-path

1) While working with real time appl^n, you might come across a scenario where inner text of an object is too lengthy or contains additional spaces then in such cases text() might not work properly. To overcome this, we can use another function of x-path i.e. contains()

syntax ⇒ //html tag [contains (arg1, ar 2)]



2) In above syntax, argument 1 indicates source which can be any html attribute or text(), whereas argument 2 (arg2) is a string which need to be searched in source.

3) Contain function can also be used when you want to identify any object which has dynamic attribute values.

For e.g.

1) Link as text Template

→ //a [contains (text(), 'Link as text'] //or

//a [contains (text(), 'Template'] //or

//a [contains (text(), 'Link as'] //or ....

NO

DATE

example 2 - (for dynamic attribute)

<input id = "username123" type = "text">

<input id = "abc12username" type = "text">

<input id = "username12567" type = "text">

⇒

//input [contains (@id, 'username')]

Note :-

x-path possible syntax :

1) //html tag [@Attribute name = 'Attribute Value']

2) //html tag [text() = 'inner text']

3) //html tag [contains (arg1, arg2)]

4) //* [anyone of the above syntax]

5) OR . AND

//html tag [@AN = 'AV' OR @AN = 'AV']

//html tag [@AN = 'AV' AND @AN = 'AV']

6) starts-with (arg1, arg2)

//html tag [starts-with (arg1, arg2)]

dom → document object model.

NO
DATE

7) following
8) ancestor
9) child
10) parent
11) preceding
12) following-sibling
13) self
14) discendant
15) access parent element from child
16) dependent element

7) following →

It will allow to select all the elements from the html dom for the current node.

Foreg // input [@id = 'username'] // following:: input

Above x-path expression will give you 4 matching nodes by using "following" axes.

object 1 → password field

object 2, 3, 4 → input fields which are hidden.

(For application URL → xtiger)

If you want to focus on any particular element such as password field then x-path will be

// input [@id = 'username'] // following:: input [1]

NO
DATE

8) Ancestor :—

This axes selects all the ancestor elements (grand parent, parent etc) of the current node.

For. e.g. application Name - actitime.

// input [@id = 'username'] // ancestor:: div

The above expression will give you two maching nodes. If you want to focus on any perticular element then you can specify the element position in square bracket as mentioned below

// input [@id = 'username'] // ancestor:: div [1]

9) child :—

// div [@class = 'textwidth'] / child:: ul (immediate child)

// div [@class = 'textwidth'] // child:: li

// div [@class = 'textwidth'] // child:: li [2]

Using this, we can select all the child element or sub-child elements of current node.

11) Preceding :—

It selects all the nodes that comes before the curre node.

For. e.g. in actitime app^n if we are identifying

NO

DATE

login button & using 'preceding' for input tag then it will give you 5 matching nodes.

1, 2, 3 will be hidden, 3rd will be password username field, 4th password field & 5th will be checkbox.

e.g.

//a [@id = 'loginButton'] // preceding :: input

12) following-siblings :—

It selects the 'following-siblings' of the context node (current node), siblings are at the same label of current node

Fore.g. https://beginnersbook.com/java-collections-tutorials/

//a [text() = 'ArrayList'] // following-sibling :: a

The above x-path expression will return 12 maching nodes from "java collection-table of contents' from the appln.

10) Parent :—

It selects all the parents of current node.

For ex. https://beginnersbook.com/java-collections-tutorials/

//a [text() = 'ArrayList'] // parent :: p

NO

DATE

14) decendant :—

It selects the decendents of current nodes.

Foreg. Flipkart application

//a [@title = "Samsung Galaxy A7 (Blue, 128 GB')] //

decendant :: img → 1 match

//a [@title = "samsung Galaxy A7 (Blue, 128 GB')] //

decendant :: div → 3 maches

Identify parent element by using child element :—

syntax : // parenthtml tag [child xpath expression]

In order to access parent element tag, specify the x-path expression of child element within square bracket as mentioned in syntax above.

Q. Write a x-path expression to identify a row which contains table header

→ //tr [th]

//tr/th → We will get a cell of a table which are headers.

Q. Write a x-path expression to identify the links present in table.

→ //td [a]

NO

DATE

Imp

Dependent & independent element :-

* While working with real time appl^n, you might come across scenarios like -

i) get the company rating from a dynamic webtable.

ii) get the specific product details or price (when product location is dynamic)

iii) get the specific flight cost etc.

a) In order to cover these scenarios we need to follow below mentioned process,

Step 1 → Identify dependent & independent element. Independent means an object which is not going to be changed or getting identified by its own. whereas ~~ind~~ dependent means it requires other objects to get identified.

Step 2 → Get the html hirarchy of dependent & independent element.

- Right click on independent element, select inspect option, now in html code slowly move your mouse pointer upwards in the tree structure till both dependent & independent elements are covered or highlighted together & then note down the tree structure

NO

DATE

Step 3 → With the help of above tree structure write a x-path expression to identify independent element.

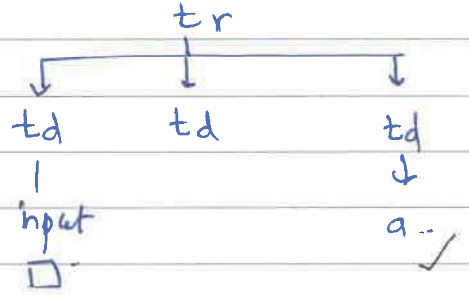
Step 4 → With the help of independent element's x-path expression, derive a expression for common parent.

Step 5 → After getting common parent expression derive a x-path expression for dependent element.

Q. Write a x-path expr^n to identify check box of google company.

→ Step 1 → dependent element - check box
indendent element - google

Step 2 → html tree structure

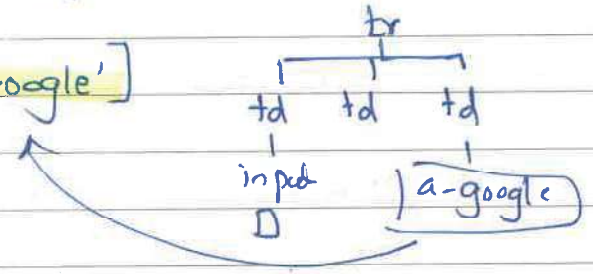


NO

DATE

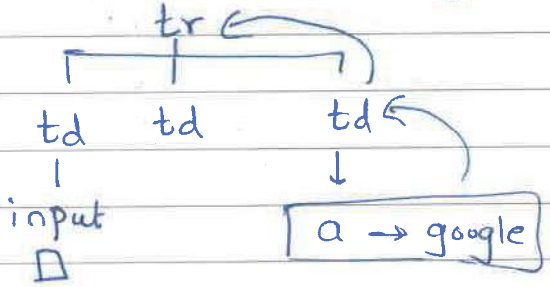
step3 → x-path expression for independent element

//a [text()='Google']



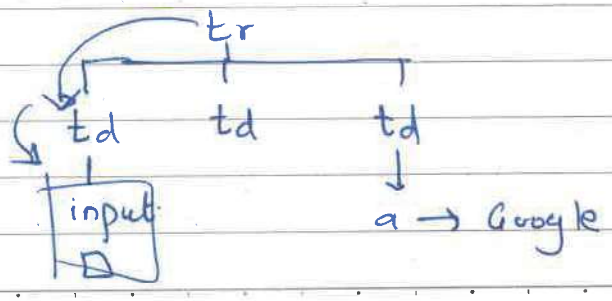
step4 → x-path expression for common parent

// tr [td [a [text()='Google']]]



step5 → x-path expression for dependent element-

// tr [td [a [text()='Google']]] /td[1]/input



NO

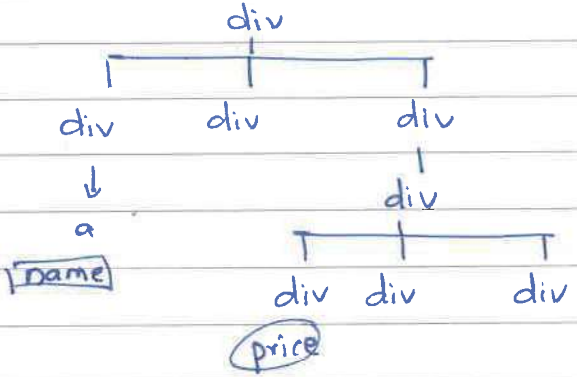
DATE

Q. Write a x-path expression to get the price of samsung galaxy J6 (Blue, 64 GB) from flipkart.com

→ step 1 : independent → mobile name

dependent → price.

Step 2 :



step3 : x-path of independent element

//a [text()='Samsung Galaxy J6 (Blue, 64 GB)']

step4 : x-path expression for common parent

//div [div [a [text()='Samsung Galaxy J6 (Blue, 64 GB)']]]

step5 : x-path expression for dependent element

//div [div [a [text()='Samsung ...)']]] /div/ div /div [1]