

# XPATH COMPENDIUM AND PRACTICE

XPath is the address of an HTML element.

There are examples of writing XPath in multiple ways in this document.

Two types of XPath:

- Absolute XPath
- Relative XPath

How to write an XPath?

1. single forward slash /
2. double forward slash //
3. XPath functions
4. Relative XPath using axes
5. Relative XPath without using axes

XPath syntax and Functions

- /tagName
- //tagName[@attributeName="attribute Value"]
- //tagName[@name,"subStringOfValue"]]
- AND, OR functions
- text()
- //tagname[text()='value']
- //tagname[contains(text(),'value')]
- normalize-space()
- //tagname[normalize-space()='value']
- Index, position, last, etc.

What is the right platform to write and verify XPath

1. Chrome DevTools
2. Console Sx
3. Using SelectorsHub

[XPath cheatsheet](#)

# Write XPath using XPath functions

single forward slash /

Website: <https://www.amazon.com/>

Element: hamburger menu button

A screenshot of the Amazon homepage. The hamburger menu button is highlighted with a red box. A red arrow points from the button to the browser's developer tools. In the developer tools, the XPath '/html/body/div[1]/header/div/div[4]/div[1]/a' is selected in the Elements tab.

Absolute XPath:

/html/body/div[1]/header/div/div[4]/div[1]/a

/tagName

(not recommended to use because if any of the above elements changes the XPath will break)

double forward slash //

Website: <https://www.amazon.com/>

Element: hamburger menu button

A screenshot of the Amazon homepage. The hamburger menu button is highlighted with a red box. A red arrow points from the button to the browser's developer tools. In the developer tools, the XPath '//a[@id="nav-hamburger-menu"]' is selected in the Elements tab.

Relative XPath:

//a[@id="nav-hamburger-menu"]

/tagName[@attributeName="attribute Value"]

XPath functions

Website: <https://www.amazon.com/>

Element: hamburger menu button

A screenshot of the Amazon homepage. The hamburger menu button is highlighted with a red box. A red arrow points from the button to the browser's developer tools. In the developer tools, the XPath '//a[contains(@id, 'hamburger')]' is selected in the Elements tab.

Using 'contains' method to shorten the XPath:

//a[contains(@id, 'hamburger')]

/tagName[@name, "subStringOfValue"]

## AND, OR functions

Website: <https://www.amazon.com/>

Element: hamburger menu button

The screenshot shows the Amazon homepage. A red box highlights the hamburger menu icon in the top-left corner. A red arrow points from this icon to the developer tools window, specifically the 'Elements' tab. In the 'Elements' tab, the XPath expression `//a[contains(@id, "hamburger") or @aria-label="Open Menu"]` is highlighted with a red box.

Add OR in case the first XPath won't work:

`//a[contains(@id, "hamburger") or @aria-label="Open Menu"]`

Add AND to make the XPath more robust:

`//a[contains(@id, "hamburger") and @aria-label="Open Menu"]`

Website: <https://www.google.com/>

Element: Google logo

The screenshot shows the Google homepage. A red box highlights the Google logo. A red arrow points from the logo to the developer tools window, specifically the 'Elements' tab. In the 'Elements' tab, the XPath expression `//img[contains(@class, "InXdpd") or @alt="Google"]` is highlighted with a red box.

`//img[contains(@class, "InXdpd") or @alt="Google"]`

Website: <https://www.google.com/>

Element: Google logo

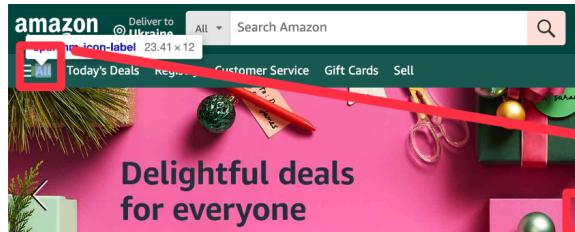
The screenshot shows the Google homepage. A red box highlights the Google logo. A red arrow points from the logo to the developer tools window, specifically the 'Elements' tab. In the 'Elements' tab, the XPath expression `//a[contains(@href, "/ipad/") and @aria-label="iPad"]` is highlighted with a red box.

`//a[contains(@href, "/ipad/") and @aria-label="iPad"]`

(if not to use AND function in this case, the XPath selects 8 elements including those in the dropdown submenu of iPad category)

## text() functions

Website: <https://www.amazon.com/>



Element: hamburger menu button with text

//a/span[text()='All']

//a/span[text()='All' and @class="hm-icon-label"]

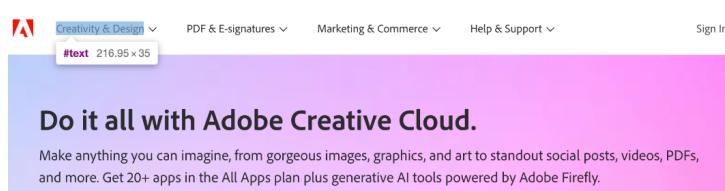
//a/span[contains(text(), 'All')]

//tagname[text()='value']

//tagname[contains(text(), 'value')]

## normalize-space()

Website: <https://www.adobe.com/>

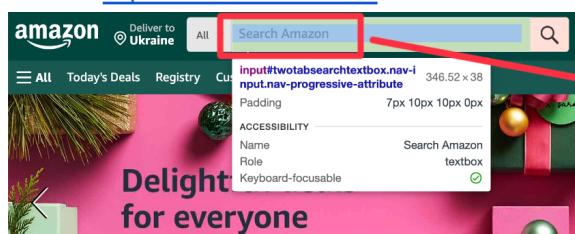


Element: menu item text

The 'normalize-space' function removes an unnecessary space in the text:

//div[section[button[normalize-space()="Creativity & Design"]]]//button[normalize-space()="PDF & E-signatures"]

Website: <https://www.amazon.com/>



Element: Search Amazon text inside the Search Field

The 'normalize-space' function removes an unnecessary space in the text:

//label[normalize-space()='Search Amazon']

//label[contains(normalize-space(),'Search Amazon')]

//label[contains(normalize-space(),'Search Amazon') and @style='display: none;']

(of course Amazon.com don't have unnecessary spaces in their code, or I didn't find one. So I had to modify HTML a little for this example)

//tagname[normalize-space()='value']

## Relative XPath using axes

Website: <https://www.shutterstock.com/>

Element: menu items

The screenshot shows the Shutterstock homepage. The top navigation bar includes links for 'Home', 'AI Generate', 'Catalog', and 'Create'. Below this is a search bar with placeholder text 'Find the content you need to make standout work—free for one month. If it's in your head, it's on our site.' and several search filters like 'Winter', 'Happy Birthday', 'Christmas', 'Christmas Tree', and 'Happy New Year'. A prominent red box highlights the 'Images' menu item in the top navigation bar. A red arrow points from this highlighted item to the developer tools element path in the bottom right.

Element path in developer tools:

```
//li[@role='menuitem' and .='Images']/ancestor::div//li[.='Video']
```

Find the 'Video' menu item by going from the 'Images' menu item to their common ancestor:

`//li[@role='menuitem' and .='Images']/ancestor::div//li[.='Video']`

Website: <https://www.amazon.com/>

Element: menu items

The screenshot shows the Amazon homepage. The top navigation bar includes links for 'Hello, sign in', 'Account & Lists', '>Returns & Orders', and a shopping cart icon. Below this is a large promotional banner for 'Shop last-minute holiday deals now'. A red box highlights the 'Today's Deals' menu item in the top navigation bar. A red arrow points from this highlighted item to the developer tools element path in the bottom right.

Element path in developer tools:

```
//a[.="Today's Deals"]/parent::div//a[.="Registry"]
```

Find the 'Registry' menu item by going from the 'Today's Deals' menu item to their common parent:

`//a[.="Today's Deals"]/parent::div//a[.="Registry"]`

## Relative XPath without axes

Website: <https://www.adobe.com/>

Element: menu items

```
//div[section[button[normalize-space()="Creativity & Design"]]]//button[normalize-space()="PDF & E-signatures"]
```

Using '[ ]' to find the parent of the menu items elements and jump to the next button:

```
//div[section[button[normalize-space()="Creativity & Design"]]]//button[normalize-space()="PDF & E-signatures"]
```

Website: <https://www.adobe.com/>

Element: list items

```
//a[@="Today's Deals"]/parent::div/a[.="Registry"]
```

Using '[ ]' to find the parent of the list items elements and jump to the next item:

```
//div[ul[li[a[.="Creative Cloud"]]]]//a[.="Adobe Express"]
```

Website: <https://account.next.ua/en/Login>

Element: Login fields input

```
//div[@id="passwordLogin"]//input[@id="EmailOrAccountNumber"]
```

One more way to write more unique parent XPath:

```
//div[@id="passwordLogin"]//input[@id="EmailOrAccountNumber"]
```