pavanpw-17

frames-

count total number of frames. returns an array of frames.

//total frames **const allframes=await page.frames() console.log("Number of frames:",allframes.length)**

using name of the frame to go into frame-

name attribute should be present.

//const var=await page.frame('name'); // if name is present

to identify frame by url-

we need to have src attribute.

see how to write it:

//const frame1=await page.frame({url:'https://ui.vision/demo/webtest/frames/frame_1.html'})

fill-

pass in the locator and value to fill.

//await frame1.fill("[name='mytext1']",'Hello');

frame locator-

we need to pass only xpath or css. name or url not acceptable.

then pass the locator of element inside the frame.

//appraoch 2- using frame locator **const inputbox=await page.frameLocator("frame[src='frame_1.html']").locator("[name='mytext1']") inputbox.fill("Hello")**

```
const { test, expect } = require('@playwright/test');

test('frames', async ({ page }) => {

  await page.goto('https://ui.vision/demo/webtest/frames/');

  //total frames
  //count total number of frames. returns an array of frames.
  const allframes=await page.frames()
  console.log("Number of frames:",allframes.length)

  //approach 1: using  name or url
  //const var=await page.frame('name'); // if name is present
  //const frame1=await page.frame
({url:'https://ui.vision/demo/webtest/frames/frame_1.html'})
  //await frame1.fill("[name='mytext1']",'Hello');


  //appraoch 2- using frame locator
  //we need to pass only xpath or css. name or url not acceptable.
  //then pass the locator of element inside the frame.
  const inputbox=await page.
frameLocator("frame[src='frame_1.html']")
.locator("[name='mytext1']")
  inputbox.fill("Hello")


  await page.waitForTimeout(5000);

});
```