Components

Monday, February 4, 2019 8:50 AM

WebAndWebApiAutomation.dll

The assembly that contains all class and interface definitions for this utility.

WebAndWebApiAutomation.xml

Xml documentation file that contains all the summary method documentation for Visual Studio intellisense

WebAutomation

Contains the utility and component instance creation methods for automating cross browser testing of a web site. This component is the only one that should be directly accessed and be 'newed up'.

Public constructor

- WebAutomation(string driverPath, int timeoutForWaitOverride)
  - driverPath -> This is the path where that driver service executables for each browser can be accessed from. This path is verified and, if it exists, will be checked for the appropriate executables.
  - timeoutForWaitOverride -> The default for this is 5 seconds and it is and optional parameter. It can be overridden by providing a positive, non-zero integer.

Methods

- IWebElement CheckChildElementExistsAndReturnIt(SelectorData parentSelectorData, SelectorData childSelectorData, IWebDriverManager webDriverManager)
  - Finds and returns the child IWebElement using the parameters provided, if none is found null is returned
  - parentSelectorData -> Data to find the parent element with
  - childSelectorData -> Data to find the child element with</param>

- By CheckElementExistsReturnCssSelector(SelectorData selectorData, IWebDriverManager webDriverManager)
  - Finds the IWebElement using the parameters provided and returns the CssSelector based By object, if none is found null is returned
  - selectorData -> Data to build the CssSelector with

- IWebElement CheckElementExistsReturnIWebElement(SelectorData selectorData, IWebDriverManager webDriverManager)
  - Finds and returns the IWebElement using the parameters provided, if none is found null is returned
  - selectorData -> Data to build the CssSelector with

- bool CheckElementsExist(SelectorDataSet selectorDataSet, IWebDriverManager webDriverManager)
  - Builds a CssSelector with the SelectorDataSet provided and uses it to check that an element exists with that data
  - selectorDataSet -> Data to check for in the current DOM instance

- IWebDriverManager Clear(IWebDriverManager webDriverManager, SelectorData selectorData)
  - Clears the text from the provided element after verifiying the element is visible
  - selectorData -> Object representing the element to be cleared

- IWebDriverManager Click(IWebDriverManager webDriverManager, SelectorData selectorData)
  - Clicks the provided element after verifying it exists and is clickable
  - selectorData -> Object representing the element to click

- bool DoesUrlContain(IWebDriverManager webDriverManager, string text)
  - Checks whether or not the current url contains the provided text
  - text -> Text to look for in the current Url

- List<By> GetAllBysUsingMatchingSelectorData(SelectorData selectorData, IWebDriverManager webDriverManager)
  - Finds all elements matching the provided selector data and returns a list of xpath by objects for each found elements
  - selectorData -> Data to check for in the current DOM instance

- IWebDriverManager GetIWebDriverManager()
  - Creates and returns and instance of the IWebDriverManager used to create and manage IWebDriver objects

- ILogger GetLogger(LoggerSettings loggerSettings)
  - Creates and returns and instance of the ILogger used to write to a common log file
  - loggerSettings -> Settings for the logger

- ITestExecutor GetTestExecutor(bool collectTestData, string resultsPath)
  - Creates and returns and instance of the ITestExecutor used to execute tests and manage the results
  - collectTestData -> Currently has no effect, false by default
  - resultsPath -> Desrination path for screenshots and test results, null by default

- IJavaScriptExecutor GetJavaScriptExecutor(IWebDriverManager webDriverManager)
  - Creates and returns an IJavaScriptExecutor object using the provided IWebDriver

- void HighlightElement(IWebDriverManager webDriverManager, SelectorData selectorData)
  - Modifies the style associated with the selector data object provided to highlight the element on the page
  - selectorData -> Object representing the element to highlight

- bool IsElementVisible(IWebDriverManager webDriverManager, SelectorData selectorData)
  - Checks whether or not the provided element is visible
  - selectorData -> Object representing the element to check

- void JavaScriptClick(IWebDriverManager webDriverManager, SelectorData selectorData)
  - Clicks the element associated with the selector data object provided using a JavaScript query. This is useful when the element being clicked may be obstructed
  - selectorData -> Object representing the element to highlight

- IWebDriverManager MoveToElement(IWebDriverManager webDriverManager, SelectorData selectorData)
  - Clears the text from the provided element after verifiying the element is visible
  - selectorData -> Object representing the element to move to

- string ParseInnerText(string htmlText)
  - This method will remove html data from the innerText of a given element and return only text NOT go beyon the first element. This method will not parse the innerText of a child element passed in as part of a larger set
  - htmlText -> Raw htmlText from an IWebElement that contains html to be removed

- IWebDriverManager SendText(IWebDriverManager webDriverManager, SelectorData selectorData, string textToEnter)
  - Enters text into the provided element after verifying it exists and is visible
  - selectorData -> Object representing the element to receive the text
  - textToEnter -> Text that will be entered

- IWebDriverManager SendTextWithDelay(IWebDriverManager webDriverManager, SelectorData selectorData, string textToEnter, int delay = 500)
  - Enters text into the provided element one character at a time with a delay between each after verifying it exists and is visible. Note: Only use this method to send text to elements that have autocomplete and require a delay on each letter typed.
  - selectorData -> Object representing the element to receive the text
  - textToEnter -> Text that will be entered
  - delay -> Interval before each character is entered

- void TakeScreenShot(IWebDriverManager webDriverManager, string sreenShotPath, string screenShotName)
  - Takes a screenshot and saves it in the provided path. The file name is in the form of "{testMethodName}_{DateTime.Now}.jpeg"
  - sreenShotPath -> Path to save the screenshot to
  - screenShotName -> Name of the screenshot file

- List<KeyValuePair<By, WebAutomationEnums.NavigationResult>> TestLinkNavigationForAllAnchorsFoundInPage(IWebDriverManager webDriverManager)
  - Validates that all the anchor tags <a /> found on the page the driver is currently navigated to. It returns the XPath By object and NavigationResult for every link found and tested

ILogger

- Thread safe logger to log test related messages according to a defined set of message types
  - No public constructor

Methods

- void Log(LogMessageType level, string message)
  - level -> One the values defined by in WebAutomationEnums.LogMessageType
  - Message -> The message to be logged

    Date information is added by the logger automatically. For example a message of 'Clicking Submit' with MessageType TESTINFO, logged within a test method named 'Execute' would be written as:

    1/25/2019 10:52:09 AM - Message Type:TESTINFO Test Method:Execute - Clicking Submit

- string ToXML(object obj);
  - Serializes the provided object to xml then returns the string representation of that xml serialization

ITestExecutor

- This interface provides access to thread safe test execution delegate methods.
  - These methods have built in logging to output test start, pass and failure messaging as well as exception logging.
  - Any exception thrown within the execute method will be logged and then rethrown to be handled by the caller.

Methods

- void Execute(Action testMethod, IWebDriverManager webDriverManager)
  - testMothod -> This is the test method to execute
  - webDriverManager -> This should have at least one driver instance created otherwise execution will fail and an exception will be thrown

IWebDriverManager

- This interface provides access to the methods available on the IWebDriver interface as well as the ability manage multiple driver instances
  - No pubic constructor

Methods

- void CreateDriverInstance(DriverType driverType, string[] driverOptions = null)
  - Creates an instance of IWebDriver that matches the type provided

- void QuitDriverInstance(DriverType driverType)
  - Quits an instance of IWebDriver that matches the type provided

- void SetActiveDriver(DriverType driverType)
  - Selects the driver to mark as active. The active driver is used for execution.

- DriverType GetActiveDriverType()
  - Returns the DriverType value indicating what driver is currently marked as active. The active driver is used for execution.

- string GetActiveDriverPageSource()
  - Gets the source of the page last loaded by the browser.

- string GetActiveDriverTitle()

        - Gets the title of the current browser window

    - string GetActiveDriverUrl()
        - Gets or sets the URL the browser is currently displaying.

    - string GetActiveDriverCurrentWindowHandle()
        - Gets the current window handle, which is an opaque handle to this window that uniquely identifies it within this driver instance.

    - ReadOnlyCollection<string> GetActiveDriverWindowHandles()
        - Gets the window handles of open browser windows.

    - void NavigateWithActiveDriver(string url)
        - Instructs the driver to navigate the browser to another location.

    - void CloseLastTabWithActiveDriver(string mainWindow)
        - Closes the farthest tab to the right in the current browser window

    - void CloseTabWithActiveDriver(string tabToClose, string targetTab)
        - Closes the tab designated in the tabToClose parameter and switches the context to the tab designated in the targetTab parameter

    - void SwitchToLastTabWithActiveDriver()
        - Switches the context to the farthest tab to the right in the current browser window

    - void CloseAllTabsExceptCurrentWithActiveDriver()
        - Closes all tabs except for the one currently with focus