Finding Page Elements

WebAii provides one of the richest markup identification infrastructures currently available on the market. It builds on top of commonly known element identification methods like 'getElementById', 'getElementByName' or 'XPath' and extends them to provide identification routines that cater more to application automation scenarios. In addition to maintaining a simple and easy to use set of APIs, WebAii introduces a consistent and extensible way to build identification and persist it using 'FindParam' objects.

It is important to understand how WebAii's identification methods works because that understanding will allow you to exploit the power of these identification methods to build robust automation quicker.

WebAii's element identification overview

WebAii supports the following identification methods:

Methods	Description	Example
Find.ById()	Searches for an element contained in a markup document using it's set 'id' attribute. When the desired 'id' matches an element's id, the element is returned - identical to getElementById	`// Find element with id=input1` `Element e = Find.ById("input1");`
Find.ByName()	Searches for an element contained in a markup document using it's set 'name' attribute. When the desired 'name' matches an element's name, the element is returned	`// Find element with name=goButton` `Element e = Find.ByName("goButton");`
Find.ByTagIndex()	Searches for an element using its tag name occurrence index. Finds the element at the specified occurrence index and returns it. This method uses zero based indexing.	`// Find the 3rd occurrence of table tag` `Element table = Find.ByTagIndex("table", 2);`

Find.ByAttributes(), Find.AllByAttributes()	Searches for an element or 'All' elements using an 'exact' or 'partial' list of attribute values (You can specify 1-N attribute/value pairs). When all attribute values match, the element or collection of elements is returned.	`// Find the first element with attribute class=myclass`
		`Element e = Find.ByAttributes("class=myclass");`
		
		`// Find the first element with attribute class=myclass`
		`// and 'src' has partial value foo.gif. (~ signifies partial)`
		`Element e = Find.ByAttributes("class=myclass", "src=~foo.gif");`
		
		`// Find all elements with class=myclass and src has a partial foo.gif`
		`IList<Element> allbtns = Find.AllByAttributes("class=myclass", "src=~foo.gif");`

Find.ByContent, Find.AllByContent()	Searches for an element or 'All' elements using 'exact', 'partial' or 'regex' of the element content. The element content can be: InnerText, InnerMarkup, OuterMarkup, TextContent (default), StartTagContent.	// Find element with TextContent has literal value: Education
		// l: signifies literal
		Element e = Find.ByContent("l:Education");
		
		// Find element with TextContent has patrial value: Education
		// p: signifies partial
		Element e = Find.ByContent("p:Education");
		
		// Find element with TextContent matches regex expression:
		// ^(<tr>\s*<td\s*scope=.*>\s*Education)
		// x: signifies regular expression
		Element e = Find.ByContent(@"x:^(<tr>\s*<td\s*scope=.*>
		\s*Education)");
		
		// Explicitly set the content type.
		Element e = Find.ByContent("p: ",
		FindContentType.StartTagContent);
		
		// Find all elements with TextContent having partial value: car
		IList<Element> alle = Find.AllByContent("p:car");
		
		// NOTE:
		//
		// There is a difference between FindContentType.InnerText &
		FindContentType.TextContent that is worth noting:
		//
		// Example: Text1 Text2</
		div>
		//
		// InnerText for div1 : Text1Text2 {recursive}
		// TextContent of div1 : Text1 {non-recursive}
		//
		// Default for ByContent is TextContent which is the most
		common usage.

Find.ByXPath, Find.AllByXPath()	Searches for an element or 'All' elements using an XPath expression. WebAii supports the .NET Framework XPath implementation. A good reference to XPath implementations can be found here.	// Find the banner img element
		Element img = Find.ByXPath("//body[1]/table[1]/tbody[1]/tr[1]/
		td[1]/img[1]");
		
		// Find all times with id=div
		IList<Element> allDivs = Find.AllByXPath("/
		descendant::node()[starts-with(@id,'div')]");

Find.AllByTagName()	Searches for 'All' elements with the specified tag name and returns it as a list of elements.	// Return all img elements IList<Element> allimg = Find.AllByTagName("img");

Find.ByNodeIndexPath()	Searches for an element using dom tree node index paths. This identification is done using an xpath like approach that simply describes the hierarchy path to a specific element using the node index within the hierarchy without having to specify the tag name at each level. This identification method can be chosen in cases where a segment of the DomTree hierarchy at a specific location is consistent but element type changes. For example, if an element you are trying to target is the direct child of another element that fluctuates between a span and div, you can choose this identification method to provide a consistent way to identify that element.	/// Given this DOM: /// /// <referenceElement> /// (0)<foo> /// <bar> /// </bar> /// <car> /// </car> /// (2)<bus> /// (0)<driver> /// <cap> /// </cap> /// (1)<target> /// </target> /// </driver> /// </bus> /// </foo> /// </referenceElement> // We can find the <target></target> element by: Element target = Find.ByNodeIndexPath("0/2/0/1");
Find.ByParam(), Find.AllByParam()	Searches for an element or 'All' elements using the **FindParam** object passed in. FindParam objects are described in more details below.	Samples are provide below.

Identification methods usage

WebAii identification methods are accessible using the **Find** object that is exposed as a property off the 'Browser' object : Manager.ActiveBrowser.Find.Byxx(...) and also a property off each TestRegion object : TestRegion.Find.Byxx(...).

The difference between the Find object off the Browser (Root Base Identification - **RBI**) and the Find object on each TestRegion (ReGion Base Identification - **RGBI**), is that the one off the Browser performs all searches starting from the root document element whereas the Find object off TestRegions searches only the elements contained in that region and uses the TestRegions open tag (i.e. <!--testregion...->) as the root element for reference based identification like tag name occurrence index and XPath identification.

With TestRegions, depending on the areas of the application that each automated test is targeting, you can use different Find objects to give each test a greater level of independence and shield it from product changes outside its target area.