What's the most efficient way to locate and set element values in an XDocument?

Asked 16 years, 1 month ago Modified 11 years ago Viewed 3k times

1

Given the following XML 'template':

```
<Request module="CRM" call="list_service_features" id="{ID}">
  <block name="auth">
     <a name="username" format="text">{USERNAME}</a>
     <a name="password" format="password">{PASSWORD}</a>
     <a name="client-id" format="counting">{CLIENT-ID}</a>
  </block>
  <a name="service-id" format="counting">{SERVICE-ID}</a>
</Request>
```

Using XDocument, what's the best way to set the values in curly braces. I got so far but got stuck on the best way to select each of the three `<a />` nodes inside the `<block/>` element. This is just a small fragment of XML, others may have up to 20 `<a name="..."></a>` elements.

The the way the XML is constructed wasn't my creation, it's what we have to sent to our supplier's 'web service'...before anyone has a laugh at the format="counting" attribute :)

@David - cheers for the response, appreciated. I was kinda hoping it'd be a bit more elegant, kinda along the lines of:

```
List<XElement> e = doc.Descendants("a").ToList();
e.Where(x => x.Attributes("name") == "username").Single().Value = "abc";
e.Where(x => x.Attributes("name") == "password").Single().Value = "abc";
```

Clearly the code above doesn't work but I thought there would be an elegant one liner for each of the `<a>` tags

c# .net xml linq-to-xml

Share
Improve this question
Follow

edited May 23, 2017 at 12:11
Community Bot
1 ● 1

asked Nov 18, 2008 at 18:52
Kev
120k ● 53 ● 305 ● 390

1 Answer

Sorted by: Highest score (default)

3

Does this do it for you? Good old Descendants property.

```
string xmlInput = ...;
XDocument myDoc = XDocument.Parse(xmlInput);
//
List<XElement> someElements = myDoc.Descendants("a").ToList();
someElements.ForEach(x => x.Value = "Foo");
//
Console.WriteLine(myDoc);
```

Hmm, I see you have an attribute in there. Can do that too:

```
string xmlInput = //...
XDocument myDoc = XDocument.Parse(xmlInput);
//
List<XText> someText =
  myDoc.Descendants()
  .Nodes()
  .OfType<XText>()
  .Where(x => x.Value.StartsWith("{") && x.Value.EndsWith("}"))
  .ToList();
//
List<XAttribute> someAttributes =
  myDoc.Descendants()
  .Attributes()
  .Where(x => x.Value.StartsWith("{") && x.Value.EndsWith("}"))
  .ToList();
//
someText.ForEach(x => x.Value = "Foo");
someAttributes.ForEach(x => x.Value = "Bar");
//
Console.WriteLine(myDoc);
```

Ah, now with what you're expecting, I will make it work:

```
List<XElement> e = myDoc.Descendants("a").ToList();
e.Where(x => x.Attribute("name").Value == "username").Single().Value = "abc";
e.Where(x => x.Attribute("name").Value == "password").Single().Value = "abc";
```

Share

Improve this answer

Follow

edited Nov 18, 2008 at 20:14

answered Nov 18, 2008 at 19:44

Amy B

110k ● 21 ● 139 ● 190

Yep that's what I'm looking for. I arrived at something similar but more clunky: e.Attributes("name").Where(a => a.Value == "username").Single().Parent.Value = "abcdef"; Much appreciated! – Kev Nov 18, 2008 at 20:22