# **Objetivo:** realizar consultas sobre contenido XML

* realizar consultas que devuelven colecciones de objetos XElement
* acceder a los valores de los elementos de dicha colección

# **Recursos:**

* [Propiedad de eje descendiente](https://msdn.microsoft.com/es-es/library/bb384876.aspx)
* [Value (Propiedad XML)](https://msdn.microsoft.com/es-es/library/bb384768.aspx)

**Ejemplo resuelto**

Trabajaremos a partir de un documento XML cargado en memoria y manipularemos sus datos. En primer lugar, contamos con un documento XML al que se ha llamado "xmlfile1.xml" que contiene la siguiente información:

<?xml version="1.0"?>

<datosGenerales>

<datosPersona>

<nombre>Fernando</nombre>

<edad>18</edad>

</datosPersona>

<datosPersona>

<nombre>Rosa</nombre>

<edad>17</edad>

</datosPersona>

<datosPersona>

<nombre>Carlos</nombre>

<edad>21</edad>

</datosPersona>

</datosGenerales>

El siguiente código muestra cómo cargar XML en memoria desde el archivo y efectuar una selección de sus datos. El ejemplo, en ejecución, dará como resultado los valores “*Fernando”* y “*Carlos”*.

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

'obtener origen de datos

Dim doc As XDocument = XDocument.Load("xmlfile1.xml")

'crear consulta

Dim Selección = \_

From datos In [[1]](#footnote-0)doc...<datosPersona>

Where datos.<nombre>.Value Like "??r\*"

Select datos.<nombre>.Value

'ejecutar consulta

For Each s In Selección

MessageBox.Show(s)

Next

End Sub

La expresión doc...<datosPersona> usa el “eje de descendientes” y básicamente dice: ***"busque todos los descendientes denominados “datosPersona"***. *En XML, un descendiente es un elemento anidado en uno o más niveles por debajo del elemento actual*. Observa que el XML del ejemplo tiene un nodo raíz denominado datosGenerales y que contiene los elementos datosPersona. La sintaxis de tres puntos se traduce básicamente en doc.Descendants("datosGenerales"). Más ejemplos en: [Propiedad de eje descendiente](https://msdn.microsoft.com/es-es/library/bb384876.aspx)

La expresión datos.<nombre>.Value: usa la propiedad Value. Puede que te estés preguntando por qué aquí necesitamos llamar explícitamente a .Value antes de hacer la comparación. La razón de ello es que **datos.<nombre> devuelve una colección de objetos XElement;** un IEnumerable(Of XElement). La propiedad de .Value proporciona acceso al **valor del primer elemento de dicha colección**. Consulta [Value (Propiedad XML)](https://msdn.microsoft.com/es-es/library/bb384768.aspx) para ver más ejemplos

|  |
| --- |
| **Puesta en práctica**  A partir de los datos de **Contactos.xml** creado en la Tarea1 mostrar:   * en un **listBox** los nombres de todos los contactos del archivo y * en otro **ListBox** los nombres y el teléfono móvil de aquellos cuyo nombre comience con “L”.   **Ojo!**, móvil es un atributo del elemento <Telefono> (<Telefono Móvil="666666666" Trabajo="911111111" />) |

**Lecturas recomendadas:**

[Cómo filtrar nombres de elemento (LINQ to XML)](http://msdn.microsoft.com/es-es/library/bb387071(v=vs.100).aspx)

[Cómo recuperar el valor de un atributo (LINQ to XML)](http://msdn.microsoft.com/es-es/library/bb387056(v=vs.100).aspx)

[Cómo recuperar un atributo en particular (LINQ to XML)](http://msdn.microsoft.com/es-es/library/bb387086(v=vs.100).aspx)

1. Origen de datos: los descendientes <datosPersona> [↑](#footnote-ref-0)