- **Introduction**
- **XLink vs. HTML Links**
- **XML Linking Language (XLink)**
  - **XLink Element**
  - **XLink Attributes**
  - **Simple Links**
  - **Extended Links**
- **XML Pointer Language (XPointer)**

## Introduction

- Linking in XML is divided into two parts**:** **XLink and XPointer**
- **External-Link: XLink** defines a standard way of **creating hyperlinks** in XML documents
  - It defines how one document **links** to another document
- **Internal-Link: XPointer** allows the hyperlinks to **point to more specific parts (fragments)** in the XML document
  - It defines how **individual parts** of a document are addressed

# XLink

- XLink is short for XML Linking Language
- Any element in an XML document can behave as a link.
- Xlink elements that specify linking information are called linking elements
- XLink supports simple links (like HTML) and extended links (for linking multiple resources together)
- With XLink, the links can be defined outside the linked files
- XLink is a W3C Recommendation
- XLink is capable of linking more than just documents; XLink links resources which includes documents, audio, video, database data,etc. Web Browsers will eventually support XLink.  However, XLink is intended for a broader base of applications, not just Web browsers.

## XPointer

- XPointer is short for XML Pointer Language
- XPointer allows the links to point to specific parts of an XML document
- XPointer uses XPath expressions to navigate in the XML document
- XPointer is a W3C Recommendation .

## XLink vs. HTML Links

- HTML:
  - HTML made it possible to **embed hypertext links** in documents
  - These links could **insert** images or let the user to **jump from** inside one document to another document
  - Or to **jump to** another part of the same document
  - Limitations:
    - Single URL only: URLs are **limited to pointing** at **a single document**
    - A **pre-set named anchor** is required before **the link** to any part of a document can be set
    - Links are purely **one-way communication**: from reader to targeted documents

## XLink vs. HTML Links

- XML **Linking**
  - Combining **XLink** and **XPointer** to embed **internal-** and **external-links** as in HTML.
  - Use **XSLT** for **rendering** the XML documents into HTML for viewing
  - XLink is a proposal for more **powerful links** between documents designed especially for use **with XML documents**
  - Supports **multidirectional links** where the links run in more than one direction.

## XLink vs. HTML Links

- XML **Linking**
  - **Any element** can become a link, not just the ***<a>*** element.
  - Links **do not even have to be stored** in the **same file** as the documents they connect.
  - These features make XLinks more **suitable** not only for new uses, but for things that can be done only with considerable effort in HTML,
    - Such as **cross-references**, **footnotes**, **end notes**, **interlinked data**, and more.

## XLink vs. HTML Links

- **Application Support**
  - XLinks have a **much broader base** of applicability than HTML links.
  - Specific linking processors can be found: i.e., **xlip** from **Fujitsu,** used to demonstrate the traverse of linking, for the detail information regarding the tools, visit:

    **http://www.w3.org/XML/2000/09/LinkingImplementations.html**

  - XLinks are **not just used** for **hypertext connections** and **embedding images** in documents
  - They can be used **by any custom application** that needs to **establish connections** between documents and parts of documents, for any reason.

## Linking Elements

- In HTML, a link is defined with the <A> tag. In XML, any element can be a link.
- Elements that include links are called linking elements.
- Linking elements are identified by an xlink:type attribute.

## XLink Element

- **XLink Syntax**
  - It is impossible for browsers to predict **what hyperlink elements** will be called in XML documents
  - The solution for **creating links** in XML documents was to **put a marker** on elements that should act as hyperlinks
  - In XML, **any element** can be **a link** or part of a link.

```
<?xml version="1.0"?>
<homepages xmlns:xlink="http://www.w3.org/1999/xlink">
  <homepage
    xlink:type="simple"
    xlink:href="http://www.w3schools.com">
    Visit W3Schools
    </homepage>
  <homepage xlink:type="simple“
    xlink:href="http://www.w3.org">
    Visit W3C
  </homepage>
</homepages>
```

**To get access to the XLink attributes and features we must declare the XLink namespace at the top of the document**

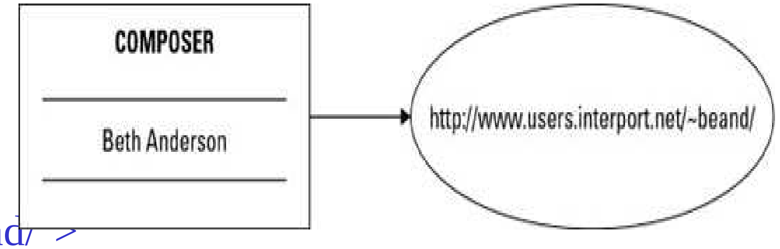
## Linking Element (Continued)

- xlink:type attribute values
  - simple
  - extended
  - locator
  - arc
  - resource
  - title

## XLink Element (cont.)

- An example:

```
<COMPOSER
     xmlns:xlink="http://www.w3.org/1999/xlink"
     xlink:type="simple"
     xlink:href="http://www.users.interport.net/~beand/">
     Beth Anderson
</COMPOSER>
```



- The xlink prefix must be bound to the **http://www.w3.org/1999/xlink namespace** URI
- **Linking information** of these elements **are included** in the attributes, not the element names
- **Attributes** define the **linking behavior**

## XLink and DTDs

- **DTDs** used with documents that use XLink
  - Validation
  - Reduce the number of XLink attributes in XML document

```
<car xmlns:xlink = "http://www.w3.org/1999/xlink"
     xlink:type = "simple" xlink:role = "MT4606"
     xlink:title = "The Latest Model">
```

  - Provide default values in DTD, and rewrite as:

```
<car xlink:role = "MT4606"
     xlink:title = "The Latest Model">
```

## XLink Element (cont.)

- To make the linking as **the default values** of an element, then add it to the **DTD file of an XML document**
- The previous example, defined the element in a DTD file:

```
<!ELEMENT COMPOSER (#PCDATA)>
    <!ATTLIST COMPOSER
      xmlns:xlink CDATA  #FIXED http://www.w3.org/1999/xlink
      xlink:type  CDATA  #FIXED "simple"
      xlink:href  CDATA  #REQUIRED>
```

- Rewrite the example element:

```
<COMPOSER
    xlink:href="http://www.users.interport.net/~beand/">
    Beth Anderson
</COMPOSER>
```

## XLink --Semantic Attributes

- Descriptions of the **Remote Resource**
- **A linking element** may have optional **xlink:role** and **xlink:title** attributes that **describe** the remote resource
  - The **document or other resource** to which the link points.
  - The **title** contains plain text that describes the resource.
  - The **role** contains a URI pointing to a document that more fully describes the resource.
  - For example, **the title** might describe what a page does and **the role** might point to **a help page** for the page

```
<SEARCH
   xlink:href="http://www.google.com/advanced_search"
   xlink:title="Search with Google"
   xlink:role="http://www.google.com/help.html">
   Search the Web with Google
</SEARCH>
```

## XLink – Behaviour Attributes

- Link Behavior Attributes (**show** and **actuate**)
  - The **show attribute** suggests how the content should be **displayed** when the link is activated
    - ***xlink: show = "new"*** indicates that **the resource** should be displayed in a new windows,
  - The **actuate attribute** suggests whether the link should be **traversed automatically** or whether a specific user request is required
    - ***Xlink:actuate="onRequest"*** indicates that **the resource** should not be retrieved until the users requests it (e.g. **by clicking on the link**)
- These are **application dependent**, however, applications are **free to ignore** the suggestions.
  - All these are based on the **XLink-aware** application

## xlink:show

- The show attribute is used to **communicate** the **desired presentation** of the **ending resource** on traversal from the **starting resource**

- **Constraint: show Value**

  - The **xlink:show** attribute has **five legal values**:
    - **replace**
    - **new**
    - **embed**
    - **other**
    - **none**

## xlink:actuate

- The **actuate** attribute is used to **communicate** the desired timing of traversal from the **starting resource** to the **ending resource**

- **Constraint: actuate Value**
  - If a value is supplied for an actuate attribute, it must be one of the values
    - **onLoad**
    - **onRequest**
    - **Other**
    - **none**

## XLink -- Links

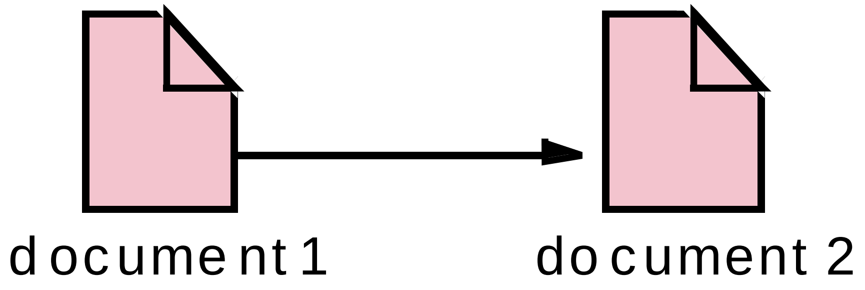
- XLink offers **two kinds** of links:
  - **Simple links**
    - Shorthand **syntax** for a common kind of link,
    - **an outbound link** with **exactly** two participating resources (into which category HTML-style ***A*** and ***IMG*** links fall).
    - Because simple links **offer less functionality** than extended links, they have **no special internal** structure.
  - **Extended links**
    - Extended links offer **full XLink functionality**, such as **inbound and third-party** arcs, as well as links that have **arbitrary numbers** of participating resources.
    - As a result, their structure can be **fairly complex**, including
      - Elements for pointing to **remote resources**,
      - Elements for containing **local resources**,
      - Elements for specifying **arc traversal rules**,
      - Elements for specifying **human-readable** resource and **arc** titles.

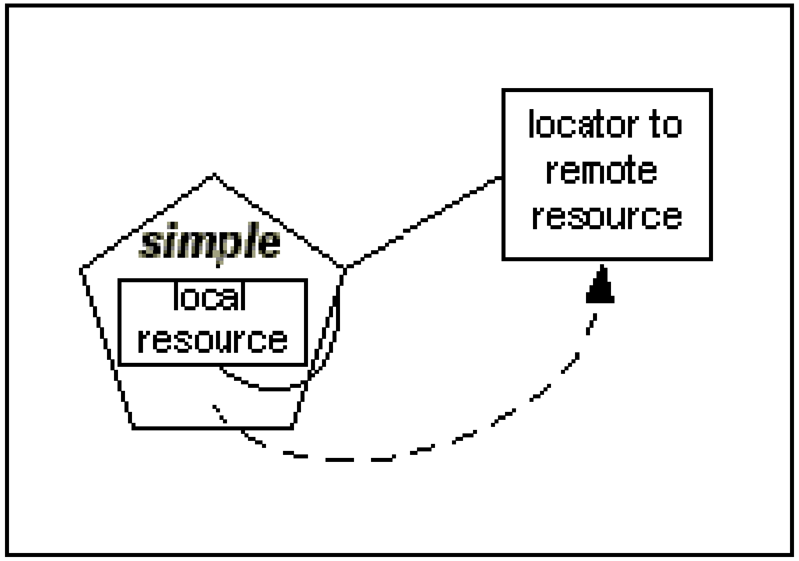
## Simple Links

- A **simple link** is a link that **associates exactly two resources**, one **local** and one **remote**, with an **arc** going from the former to the latter.
  - Thus, a simple link is always an **outbound link**
- Links one resource to another (**similarly to HTML link**)
- Linking elements
  - Specify linking information

    **<COMPOSER xlink:type = "simple"**
    **xlink:href = "**http://www.users.interport.net/~beand/**">**
  - Linking element (COMPOSER) is ***local resource***
  - **http://www.users.interport.net/~beand**/ is ***remote resource***

## Illustrating a simple link from document 1 to document 2.

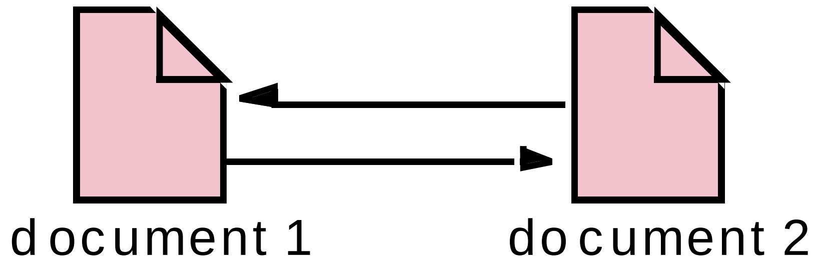




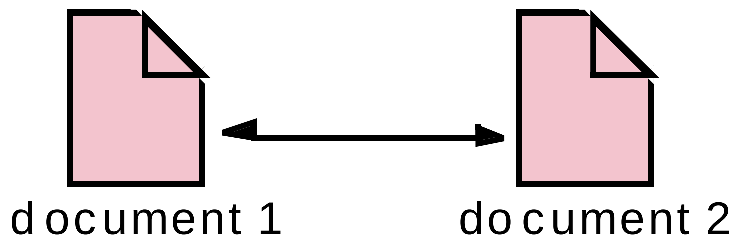
## Extended Links

- **Definition**: An **extended link** is a link that **associates** an arbitrary number of resources
  - The **participating resources** may be any combination of **remote and local**
  - Link **multiple** combinations of local and remote resources
- Remember the **"back"** button in homepage to traverse back to previous document? It is also **a browser function**.
- **Multidirectional links**
  - Traverse between resources
  - Can link any number of resources
  - ***Unidirectional links*** may not offer return to local resource

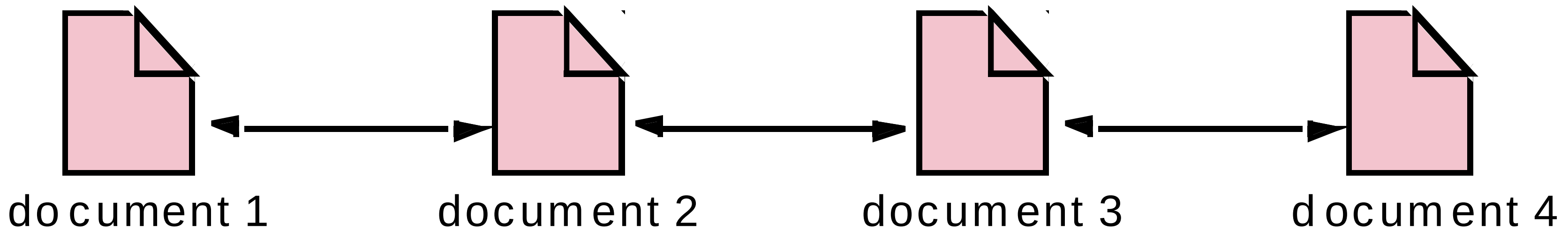
## Two unidirectional links.



## Multidirectional link.



## Multidirectional linking between four resources.



## Extended Links (cont.)

- **Syntax**
  - Extended links generally **point to** more than one target and **from** more than one source.
  - Both **sources and targets** are called by the more generic word ***"resource"***.
  - ***"Resources"*** are divided into remote resources and local resources.
  - A **local resource**
    - It is actually contained **inside the extended link elements**
    - It is the content of an element of arbitrary type that has an **xlink:type** attribute with the value ***resource***.

```
<WEBSITE xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="extended">
    <NAME xlink:type="resource">Cafe au Lait</NAME>
    ……..
</WEBSITE>
```

## Extended Links (cont.)

- Syntax
  - A **remote resource** :
    - It exists **outside the extended link element**, very possibly in another document.
    - The extended link element **contains locator** child elements that **point to** the **remote resource**.
    - These are elements with any name that have an ***xlink:typ***e attribute with the value ***“locator”***.
    - Each locator element has an ***xlink:href*** attribute whose value is **a URI** locating the remote resource

```
<WEBSITE xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="extended">
        <HOMESITE xlink:type="locator"
        xlink:href="http://ibiblio.org/javafaq/"/>

        ……………….
</WEBSITE>
```

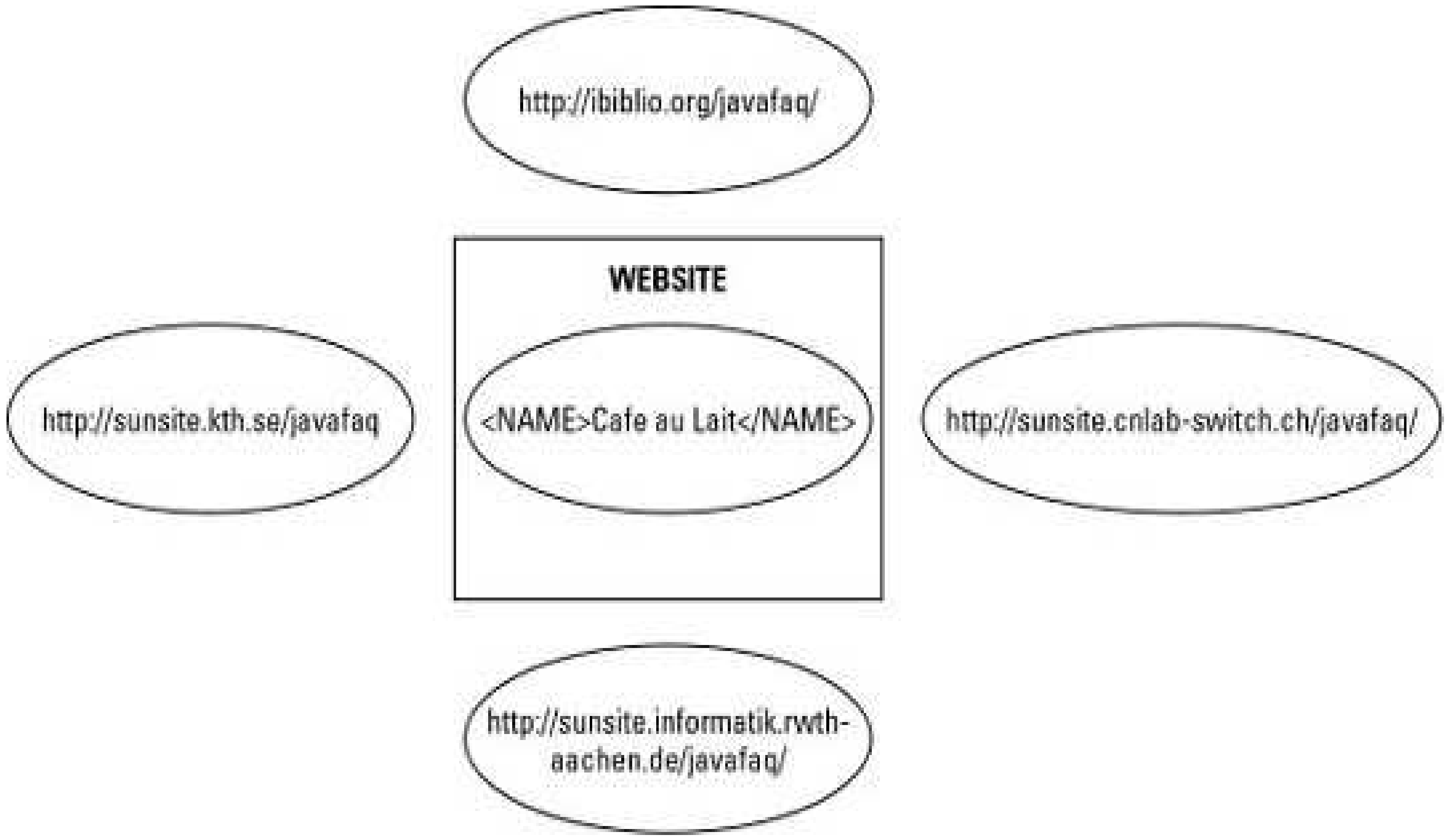
## Extended Links (cont.)

- For example, suppose you're writing **a page of links** to Java sites.
- One of the sites you want to link to is **Cafe au Lait** at **http://ibiblio.org/javafaq/.**
- However, there are also **three mirrors** of that site in three other countries.
- Some people coming to your site will want to **access the home site** while others will want to go to one of the **mirror sites**.
- Do it in **HTML**?  You have to write **four different links**

```
<WEBSITE xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="extended">
    <NAME xlink:type="resource">Cafe au Lait</NAME>
    <HOMESITE xlink:type="locator" xlink:href="http://ibiblio.org/javafaq/"/>
    <MIRROR xlink:type="locator" xlink:href="http://sunsite.kth.se/javafaq"/>
    <MIRROR xlink:type="locator" xlink:href="http://sunsite.informatik.rwth-
      aachen.de/javafaq/"/>
   <MIRROR xlink:type="locator" xlink:href="http://sunsite.cnlab-switch.ch/javafaq/"/>
</WEBSITE>
```

## Extended Links (cont.)

- Shows the **WEBSITE** extended link element and **five resources**.
- The **WEBSITE** element contains one **local resource** and refers to the other **four remote resources** by URLs.
- However, this just **describes** these resources. **No connections** are implied between them


http://ibiblio.org/javafaq/
WEBSITE
http://sunsite.kth.se/javafaq
<NAME>Cafe au Lait</NAME>
http://sunsite.cnlab-switch.ch/javafaq/
http://sunsite.informatik.rwth-aachen.de/javafaq/

## XML Pointer Language (XPointer)

- Defines an **addressing scheme** for individual parts of an XML document
- These addresses can be used by **any application** that needs to identify parts of or locations in an XML document.
  - For instance, an **XML editor** could use an **XPointer** to identify the current position of the **insertion point** or the range of the selection
- References fragments of XML document via **URI**
  - Link to **specific part** of resource, instead of linking to **entire resource**
  - Link to specific ***locations*** (i.e., **XPath** tree nodes)
  - Link to **ranges** of locations
- Uses **XPath** to **reference** XML document nodes
- Also used for **searching** XML documents via **string matching**

## XPointer Example

- **Traditional way** to reference part of document

  **<H2><A NAME="xtocid20.2">XPointer Examples</A></H2>**

  - You can then link to this position in the file by adding a **#** and **the name of the anchor** to the URL.
  - The piece of the URL after the # is called the **fragment identifier**.
  - For example, in this link the fragment identifier is **xtocid20.2**

  **<A HREF="http://www.ibiblio.org/xml/bible/20.html#xtocid20.2">**
  **XPointer Examples**
  **</A>**

- Problems:
  - This solution is kludge. It's not always possible to **modify the target document**
  - **Named anchors** violate the principle of **separating markup from content**

## XPointer Example (cont.)

- XPointers allow **much more sophisticated connections** between parts of documents.
- An XPointer can refer to **any element** of a document
  - To the first, second, or seventeenth element, and so on
- XPointers provide very **precisely targeted addresses** of particular parts of documents.
- They **do not require** the **targeted document** to contain additional markup just so its individual pieces can be linked to.
- Furthermore, unlike HTML anchors, **XPointers** don't point to just a single point in a document.
  - They can **point to entire elements**, to possibly discontinuous sets of elements, or to the range of text between two points.

1. Thus, you can use an XPointer to select **a particular part** of a document,
2. Perhaps so it can be **copied or loaded** into a program.

# XPointer Example (cont.)

Finds the element with the ID *"ebnf"*

- Here are a few examples of XPointers:
- Each of these seven XPointers **selects** a particular element in a document

Finds the second *"language"* element in the document

Finds the second child element of the fourteenth child element of the root element

**1. xpointer(id("ebnf"))**

**2. xpointer(descendant::language[position()=2])**

**3. element(/1/14/2)**

**4. xpointer(id("ebnf"))xpointer(id("EBNF"))**

Points to the element with the ID *"ebnf"*. However, if no such element is present, it then finds the element with the ID *"EBNF*

```
<SPECIFICATION xmlns:xlink="http://www.w3.org/1999/xlink"
        xlink:type="simple"
       xlink:href="http://www.w3.org/TR/1998/REC-xml-19980210.xml#xpointer(id('ebnf'))"
        xlink:actuate="onRequest" xlink:show="replace">
         Extensible Markup Language (XML) 1.0
</SPECIFICATION>
```

```
1  <?xml version = "1.0"?>
2  <!--  contacts.xml -->
3  <!-- contact list document     -->
4
5  <contacts>
6     <contact id = "author01">Deitel, Harvey</contact>
7     <contact id = "author02">Deitel, Paul</contact>
8     <contact id = "author03">Nieto, Tem</contact>
9  </contacts>
```

**Example contact list**

Lines 5-9

Mark up contact list that contains ids for three authors

## XML Pointer Language (cont.)

– Assume contact list has relative URI **`/contacts.xml`**

- XLink references entire contact list with URI

```
xlink:href = "/contacts.xml"
```

- XPointer references specific part:

  – Element contact with **id of author02**

```
xlink:href = "/contacts/xml#xpointer(
  //contact[@id = ‘author02]’)"
```

# XLink Attribute Reference

| Attribute | Value | Description |
|---|---|---|
| **xlink:actuate** | onLoad<br>onRequest<br>other<br>none | Defines when the linked resource is read and shown |
| **xlink:href** | *URL* | The URL to link to |
| **xlink:show** | embed<br>new<br>replace<br>other<br>none | Where to open the link. Replace is default |
| **xlink:type** | simple<br>extended<br>locator<br>arc<br>resource<br>title<br>none | The type of link |