Search for: Search

- Home
- About
- XTF Implementations
- XTF Community
- Tutorial
    - Quick Start
    - Fundamental Concepts
    - First Steps
    - The Essentials
    - The Exercises
        - Exercise 1: Add new content
        - Exercise 2: Change metadata
        - Exercise 3: Change logo/colors
        - Exercise 4: Results ranking
        - Exercise 5: Customize search
        - Exercise 6: Modify results
        - Exercise 7: Strucutral searching
        - Exercise 8: Hierarchical facets
        - Exercise 9: Change footnotes
- Documentation
    - Change Log
    - Deployment Guide
    - Programming Guide
    - Tag Reference
    - Tips & Tricks
    - Under the Hood
    - Experimental Features
    - Undocumented Features
    - Resources
        - Rowan Brownlee’s Beginner’s Guide to XTF
        - XTF Stylesheet Hierarchy
- FAQ
- Download
- Support

Rowan Brownlee’s Beginner’s Guide to XTF

Media collections at the University of Sydney Library: Rowan Brownlee’s beginner’s guide to configuring XTF for presenting non-textual content and metadata

Rowan Brownlee, Digital Project Analyst, University of Sydney Library

XTF version 2.1.1

Last updated, 17 February 2010

The University of Sydney Library is using XTF to enable access to several media collections. The collections are publicly available, and this initiative results from partnerships between the Library, academics and associated project managers.

When I began working with XTF, I modified the default user interface to cater for presentation of a plant sciences image collection. I then sought to test application of the model across two additional collections within visual arts and archaeology. This section of the XTF wiki provides an outline of the steps I followed, using a collection of image and video reproductions of works created by staff and students at Sydney College of the Arts (SCA). For the SCA Archive, my project partner is Jacqui Spedding, a ceramic artist and project manager for SCA Images Online.

The presentation model described is not intended to suit all needs. It does however provide an example of adapting XTF stylesheets to suit non-textual media while capitalising on the strength and functionality of the framework.

Tutorial Components

About the Collections

Choosing XTF

I was working with botanic taxonomists and other librarians on a plant sciences digital media project titled eBot. We originally contracted a developer to create a PHP web form and a relational database enabling record creation, image submission, search, display and taxonomy management. Despite everyone's best efforts, the results didn't match the team's expectations.

Seeking an alternative, I discovered that XTF was being used in April, an Australian Research Council project involving the Library and featuring Gary Browne (the Library's development programmer). XTF will also be an important component of a planned upgrade to the Library's text management platform. Although I wasn't initially sure that XTF would meet the eBot project's needs, there were a number of arguments in its favour. XTF is a 'known' technology backed by the Library, so there is a viable support model. Search and retrieval is very fast and configuration of presentation highly flexible. The default installation includes a number of example stylesheets suitable for adaptation for presentation of non-textual media such as images and video.

A key feature of XTF is its ability to cater for a wide range of metadata sets. My academic partners all have project- or domain-specific metadata. XTF requires no modification of source metadata and has readily accommodated metadata from the plant sciences, visual arts and archaeology. If an XSLT stylesheet can be written, source metadata can be presented.

XTF is not however a purpose-built media management system and we are investigating options for collection management and media processing. XTF also requires skills in related XML technologies such as XSLT and XPath. On balance, XTF is proving an excellent choice for our media search and presentation needs.

Next… Gathering content

Gathering content

The SCA media collection management system uses a Filemaker database which was initially developed by Anthony Green, (Visual Resources Librarian, Power Institute Visual Resources Library)for use within the Library. As part of an ongoing collaborative project, Anthony made the system available to SCA. Jacqui manages bibliographic metadata and related taxonomies for a number of research and teaching collections at SCA. For the XTF-based Archive, she exports records from Filemaker as a comma separated file (csv). I process the csv using a Python script to produce a set of XML records, ready for indexing by XTF.

Jacqui also provides sets of image and video files to be presented with their associated metadata. Using Photoshop batch processing techniques, Jacqui outputs thumbnail, view, powerpoint and zoomified versions from the original archival TIFF files. Flix provides Flash versions of video files. I use a Python script to check that each XML record is accompanied by its full complement of associated files and that each file is the correct format. (For web viewing of image files we need single-layered 8 bit per channel sRGB in jpeg format). Along with its excellent text processing tools, Python offers a very useful imaging library, and I'm considering additional process checks prior to submission of media and metadata.

Next… Installation and setup

Installation and setup

I followed the instructions in the XTF installation quick start to verify that initial setup was successful. I then deleted the sample data and created the following set of directories for metadata and media.

```
data/sca/records
media/sca/images/powerpoint
media/sca/images/thumbs
media/sca/images/view
media/sca/images/zoom
media/sca/video
```

Although the media files are currently located within the XTF directory structure, we will probably house them elsewhere on our storage area network.

Next… Telling XTF which records to index

Telling XTF which records to index

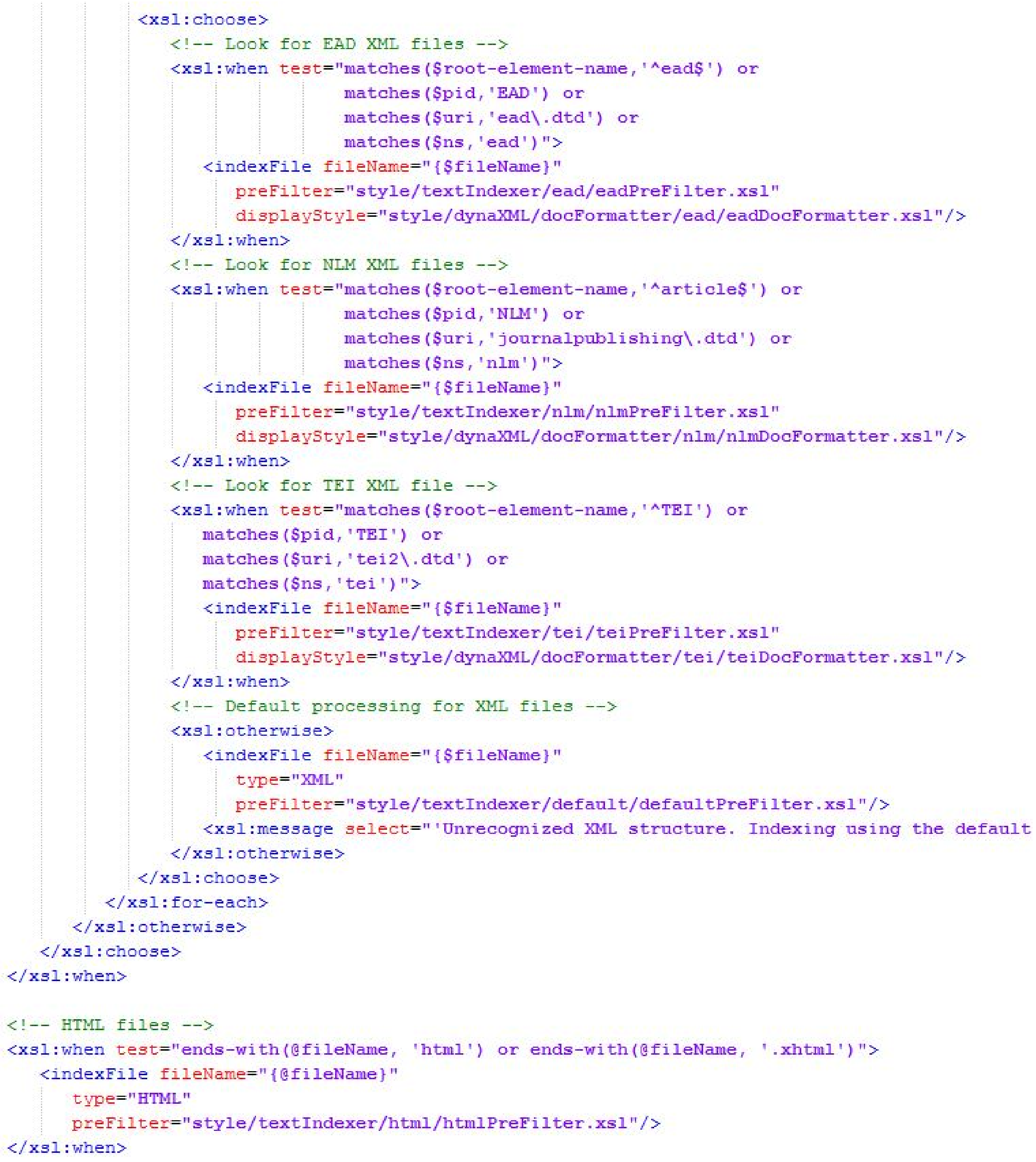
style/textIndexer/docSelector.xsl

The stylesheet *docSelector.xsl* identifies which files to index and which stylesheets to apply to each type of file.

- Default docSelector.xsl
- Modified docSelector.xsl

Default docSelector.xsl

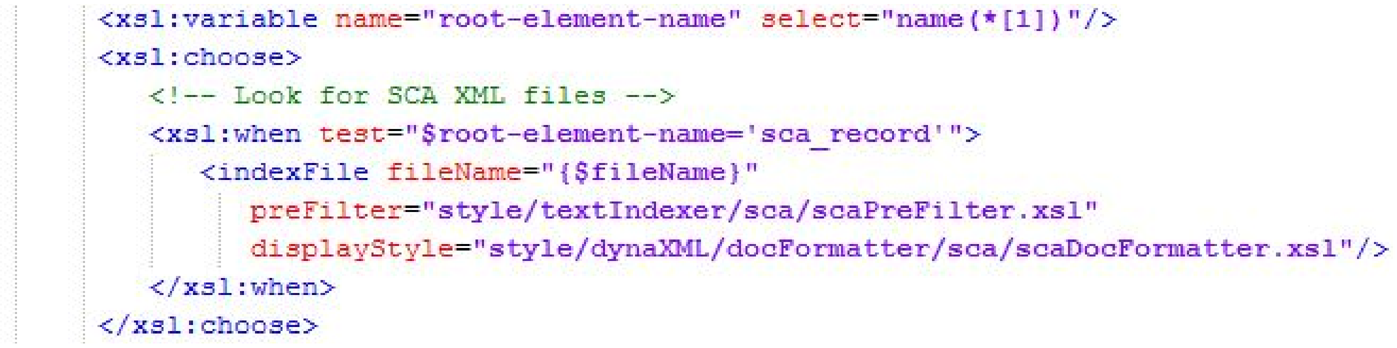
Given sufficient information to identify a category of file, XTF is able to apply the required stylesheets for data processing. The stylesheet *docSelector.xsl* includes a number of example tests enabling TextIndexer to identify a variety of XML, HTML and other files. In the following extract, one of the ways that XTF identifies an EAD XML file is to test if the string *ead* appears within a file’s root element.



```
            <xsl:choose>
               <!-- Look for EAD XML files -->
               <xsl:when test="matches($root-element-name,'^ead$') or
                               matches($pid,'EAD') or
                               matches($uri,'ead\.dtd') or
                               matches($ns,'ead')">
                  <indexFile fileName="{$fileName}"
                     preFilter="style/textIndexer/ead/eadPreFilter.xsl"
                     displayStyle="style/dynaXML/docFormatter/ead/eadDocFormatter.xsl"/>
               </xsl:when>
               <!-- Look for NLM XML files -->
               <xsl:when test="matches($root-element-name,'^article$') or
                               matches($pid,'NLM') or
                               matches($uri,'journalpublishing\.dtd') or
                               matches($ns,'nlm')">
                  <indexFile fileName="{$fileName}"
                     preFilter="style/textIndexer/nlm/nlmPreFilter.xsl"
                     displayStyle="style/dynaXML/docFormatter/nlm/nlmDocFormatter.xsl"/>
               </xsl:when>
               <!-- Look for TEI XML file -->
               <xsl:when test="matches($root-element-name,'^TEI') or
                  matches($pid,'TEI') or
                  matches($uri,'tei2\.dtd') or
                  matches($ns,'tei')">
                  <indexFile fileName="{$fileName}"
                     preFilter="style/textIndexer/tei/teiPreFilter.xsl"
                     displayStyle="style/dynaXML/docFormatter/tei/teiDocFormatter.xsl"/>
               </xsl:when>
               <!-- Default processing for XML files -->
               <xsl:otherwise>
                  <indexFile fileName="{$fileName}"
                     type="XML"
                     preFilter="style/textIndexer/default/defaultPreFilter.xsl"/>
                  <xsl:message select="'Unrecognized XML structure. Indexing using the default
               </xsl:otherwise>
            </xsl:choose>
         </xsl:for-each>
      </xsl:otherwise>
   </xsl:choose>
</xsl:when>

<!-- HTML files -->
<xsl:when test="ends-with(@fileName, 'html') or ends-with(@fileName, '.xhtml')">
   <indexFile fileName="{@fileName}"
      type="HTML"
      preFilter="style/textIndexer/html/htmlPreFilter.xsl"/>
</xsl:when>
```

Modified docSelector.xsl

Within the *file* template, I added a section describing SCA XML metadata files. Since all of the records contain *sca_record* within the root element, I used this as an identifying feature. I also included path references to *scaPreFilter.xsl* and *scaDocFormatter.xsl*. (More on these two files in identifying metadata elements for indexing and associating stylesheets with metadata records.)



```
<xsl:variable name="root-element-name" select="name(*[1])"/>
<xsl:choose>
   <!-- Look for SCA XML files -->
   <xsl:when test="$root-element-name='sca_record'">
      <indexFile fileName="{$fileName}"
         preFilter="style/textIndexer/sca/scaPreFilter.xsl"
         displayStyle="style/dynaXML/docFormatter/sca/scaDocFormatter.xsl"/>
   </xsl:when>
</xsl:choose>
```

Next… Identifying metadata elements for indexing and faceted browsing

Identifying metadata elements for indexing and faceted browsing

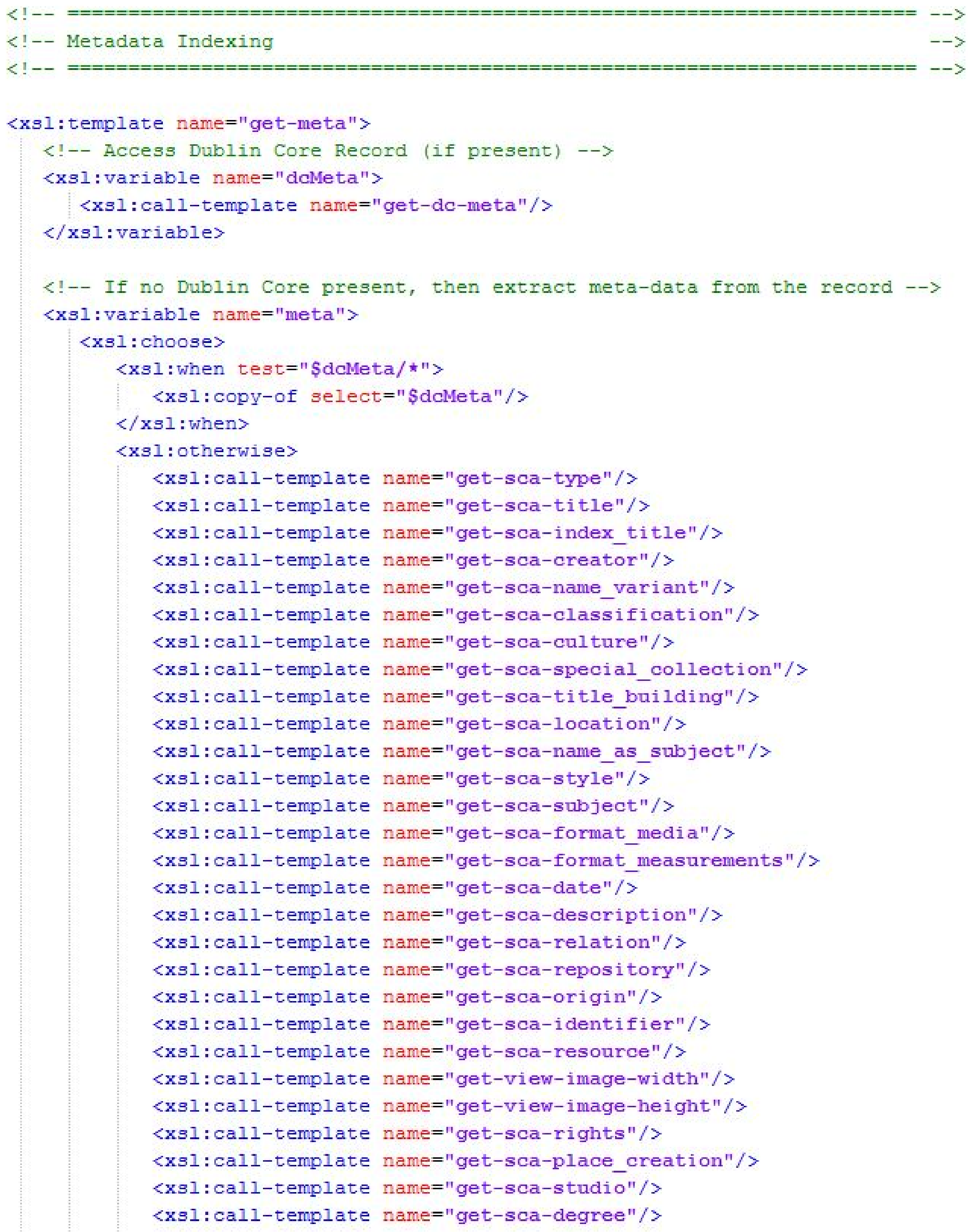
style/textIndexer/sca/scaPreFilter.xsl

scaPreFilter.xsl identifies metadata elements for indexing and for faceted browsing. I adapted the example *nlmPreFilter.xsl*, retaining the *import common templates*, *output parameters*, *identify transformation* and *root template* sections. I removed the NLM indexing section but kept the same kind of structure for the *get-meta* template. Most of the stylesheet comprises named templates, each of which target a particular metadata element. The syntactical requirements of *metadata indexing* templates are slightly different to those for *facet* templates, so I created two sets of templates – one to suit each type.

- Calling named templates: identification of metadata elements
- Example named templates for metadata indexing
- Example named templates for faceted browsing

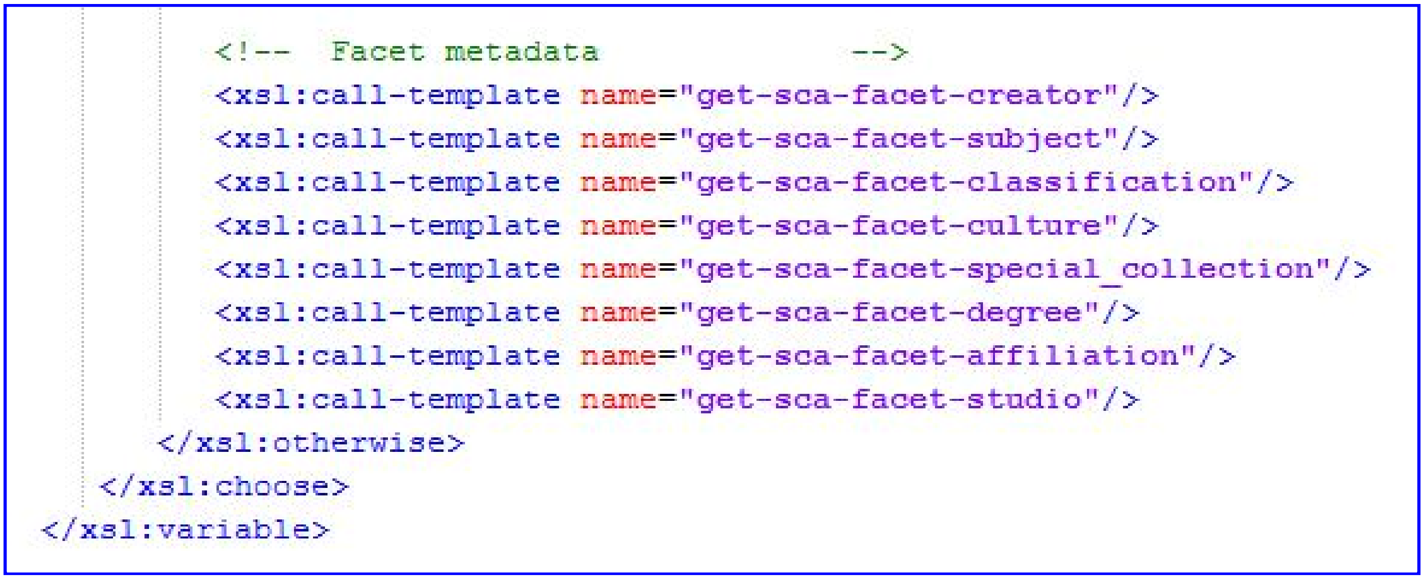
Calling named templates: identification of metadata elements

Extract from the *get-meta* template, listing calls to named templates targeting particular metadata elements for indexing.



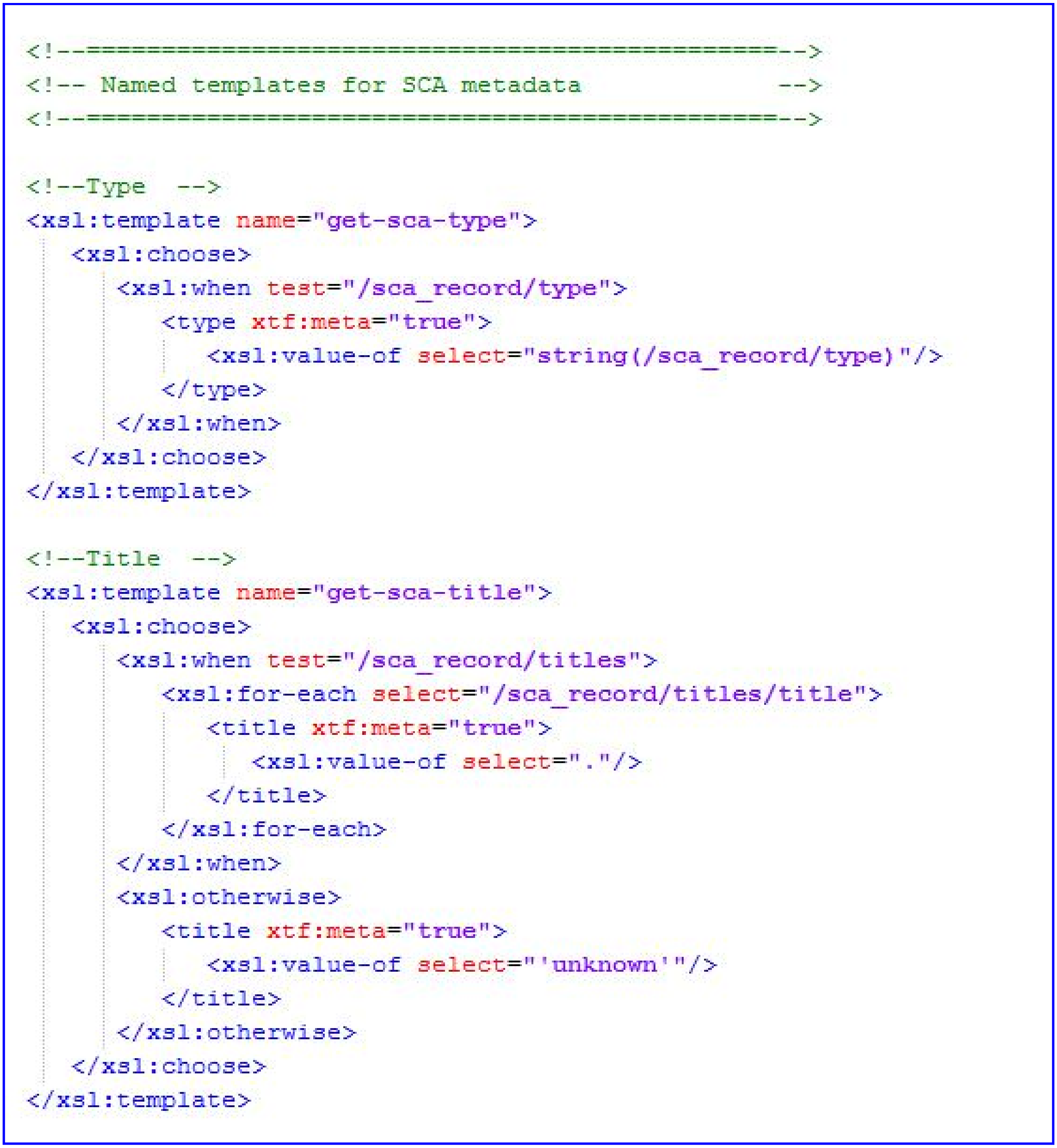
```
<!-- ============================================================ -->
<!-- Metadata Indexing                                            -->
<!-- ============================================================ -->

<xsl:template name="get-meta">
   <!-- Access Dublin Core Record (if present) -->
   <xsl:variable name="dcMeta">
      <xsl:call-template name="get-dc-meta"/>
   </xsl:variable>

   <!-- If no Dublin Core present, then extract meta-data from the record -->
   <xsl:variable name="meta">
      <xsl:choose>
         <xsl:when test="$dcMeta/*">
            <xsl:copy-of select="$dcMeta"/>
         </xsl:when>
         <xsl:otherwise>
            <xsl:call-template name="get-sca-type"/>
            <xsl:call-template name="get-sca-title"/>
            <xsl:call-template name="get-sca-index_title"/>
            <xsl:call-template name="get-sca-creator"/>
            <xsl:call-template name="get-sca-name_variant"/>
            <xsl:call-template name="get-sca-classification"/>
            <xsl:call-template name="get-sca-culture"/>
            <xsl:call-template name="get-sca-special_collection"/>
            <xsl:call-template name="get-sca-title_building"/>
            <xsl:call-template name="get-sca-location"/>
            <xsl:call-template name="get-sca-name_as_subject"/>
            <xsl:call-template name="get-sca-style"/>
            <xsl:call-template name="get-sca-subject"/>
            <xsl:call-template name="get-sca-format_media"/>
            <xsl:call-template name="get-sca-format_measurements"/>
            <xsl:call-template name="get-sca-date"/>
            <xsl:call-template name="get-sca-description"/>
            <xsl:call-template name="get-sca-relation"/>
            <xsl:call-template name="get-sca-repository"/>
            <xsl:call-template name="get-sca-origin"/>
            <xsl:call-template name="get-sca-identifier"/>
            <xsl:call-template name="get-sca-resource"/>
            <xsl:call-template name="get-view-image-width"/>
            <xsl:call-template name="get-view-image-height"/>
            <xsl:call-template name="get-sca-rights"/>
            <xsl:call-template name="get-sca-place_creation"/>
            <xsl:call-template name="get-sca-studio"/>
            <xsl:call-template name="get-sca-degree"/>
```

Extract from *get-meta* template, listing calls to named templates targeting particular metadata elements for faceted browsing.



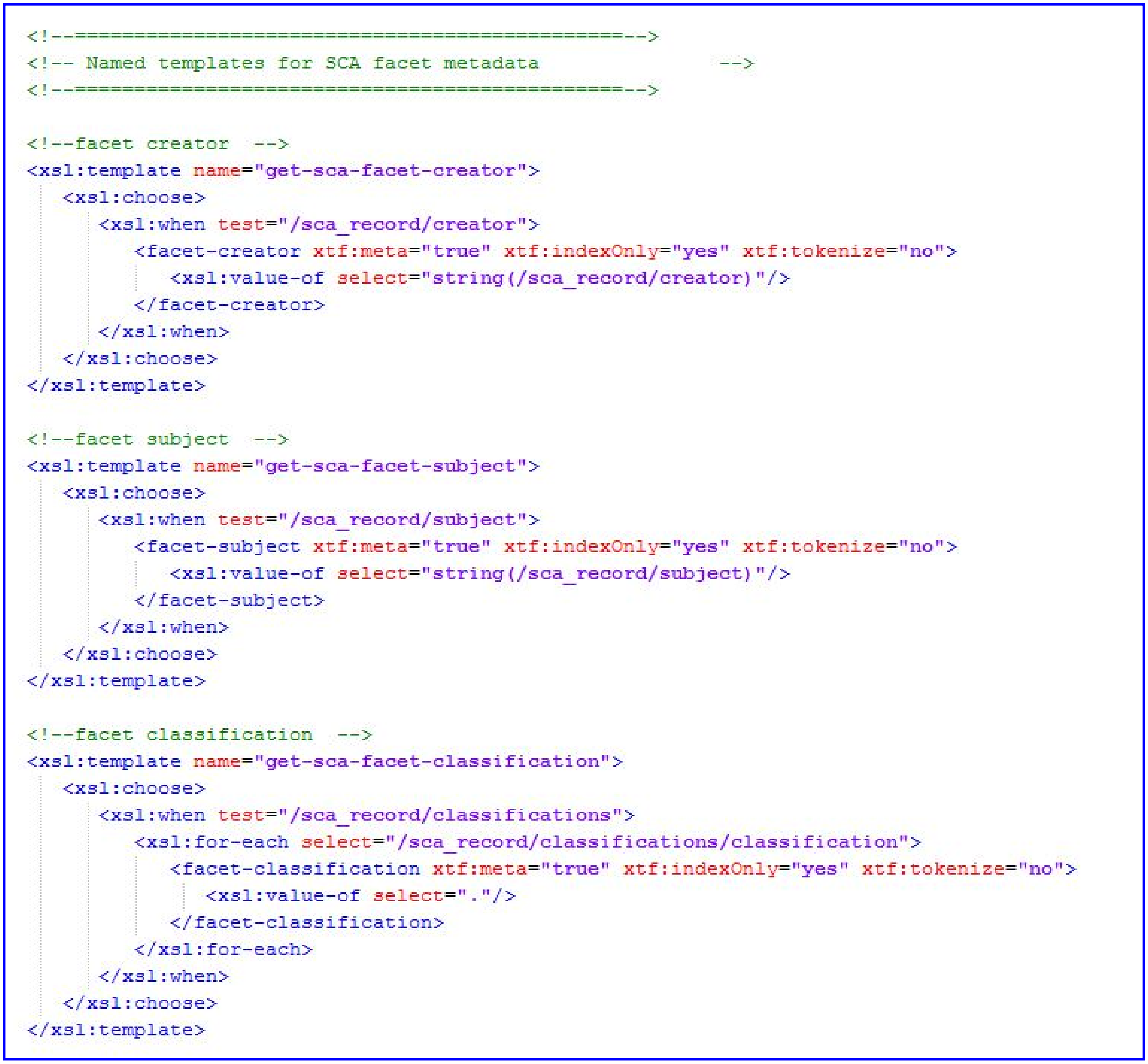
```
         <!--  Facet metadata             -->
         <xsl:call-template name="get-sca-facet-creator"/>
         <xsl:call-template name="get-sca-facet-subject"/>
         <xsl:call-template name="get-sca-facet-classification"/>
         <xsl:call-template name="get-sca-facet-culture"/>
         <xsl:call-template name="get-sca-facet-special_collection"/>
         <xsl:call-template name="get-sca-facet-degree"/>
         <xsl:call-template name="get-sca-facet-affiliation"/>
         <xsl:call-template name="get-sca-facet-studio"/>
      </xsl:otherwise>
   </xsl:choose>
</xsl:variable>
```

Example named templates for metadata indexing



```
<!--==================================================-->
<!-- Named templates for SCA metadata               -->
<!--==================================================-->

<!--Type  -->
<xsl:template name="get-sca-type">
   <xsl:choose>
      <xsl:when test="/sca_record/type">
         <type xtf:meta="true">
            <xsl:value-of select="string(/sca_record/type)"/>
         </type>
      </xsl:when>
   </xsl:choose>
</xsl:template>

<!--Title  -->
<xsl:template name="get-sca-title">
   <xsl:choose>
      <xsl:when test="/sca_record/titles">
         <xsl:for-each select="/sca_record/titles/title">
            <title xtf:meta="true">
               <xsl:value-of select="."/>
            </title>
         </xsl:for-each>
      </xsl:when>
      <xsl:otherwise>
         <title xtf:meta="true">
            <xsl:value-of select="'unknown'"/>
         </title>
      </xsl:otherwise>
   </xsl:choose>
</xsl:template>
```

Example named templates for faceted browsing



```
<!--==============================================-->
<!-- Named templates for SCA facet metadata               -->
<!--==============================================-->

<!--facet creator  -->
<xsl:template name="get-sca-facet-creator">
   <xsl:choose>
      <xsl:when test="/sca_record/creator">
         <facet-creator xtf:meta="true" xtf:indexOnly="yes" xtf:tokenize="no">
            <xsl:value-of select="string(/sca_record/creator)"/>
         </facet-creator>
      </xsl:when>
   </xsl:choose>
</xsl:template>

<!--facet subject  -->
<xsl:template name="get-sca-facet-subject">
   <xsl:choose>
      <xsl:when test="/sca_record/subject">
         <facet-subject xtf:meta="true" xtf:indexOnly="yes" xtf:tokenize="no">
            <xsl:value-of select="string(/sca_record/subject)"/>
         </facet-subject>
      </xsl:when>
   </xsl:choose>
</xsl:template>

<!--facet classification  -->
<xsl:template name="get-sca-facet-classification">
   <xsl:choose>
      <xsl:when test="/sca_record/classifications">
         <xsl:for-each select="/sca_record/classifications/classification">
            <facet-classification xtf:meta="true" xtf:indexOnly="yes" xtf:tokenize="no">
               <xsl:value-of select="."/>
            </facet-classification>
         </xsl:for-each>
      </xsl:when>
   </xsl:choose>
</xsl:template>
```

Next… Associating stylesheets with metadata records

Associating stylesheets with metadata records

style/dynaXml/docReqParser.xsl

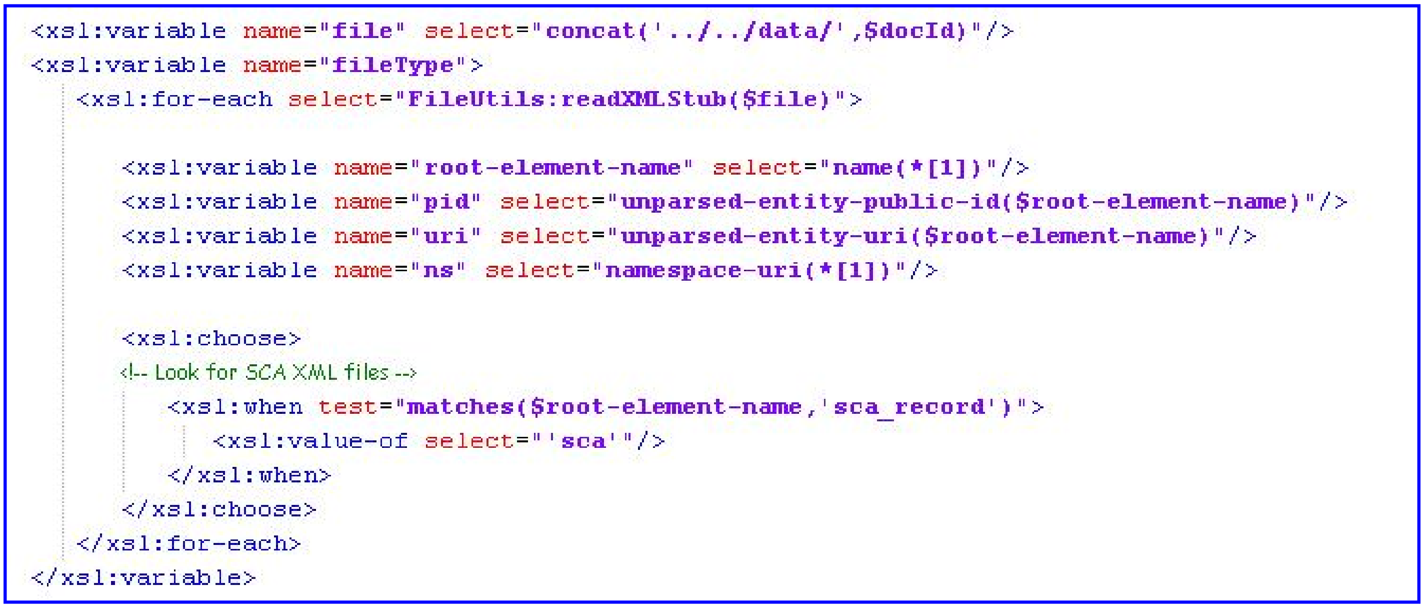
XTF needs to know which stylesheets to apply to the SCA XML records, both for indexing as well as presentation as html pages. Within the root template, I include instructions to associate SCA-specific *preFilter* and *docFormatter* stylesheets with SCA XML records. As described in the previous section, *scaPreFilter.xsl* identifies metadata elements for indexing and faceted browsing. (*scaDocFormatter.xsl* is described in displaying full records).

The document root element of each XML file is tested. If it contains *sca_record*, a variable *fileType* is assigned the value *sca*. Statements within the *style* tag and *preFilter* tag test the value of *fileType*. If fileType = sca, *scaPreFilter.xsl* and *scaDocFormatter.xsl* will be applied to the file.

- Testing for an SCA XML record
- Associating scaPreFilter.xsl with SCA XML records (for indexing)
- Associating scaDocFormatter.xsl with SCA XML records (for presentation as html pages)

Testing for an SCA XML record

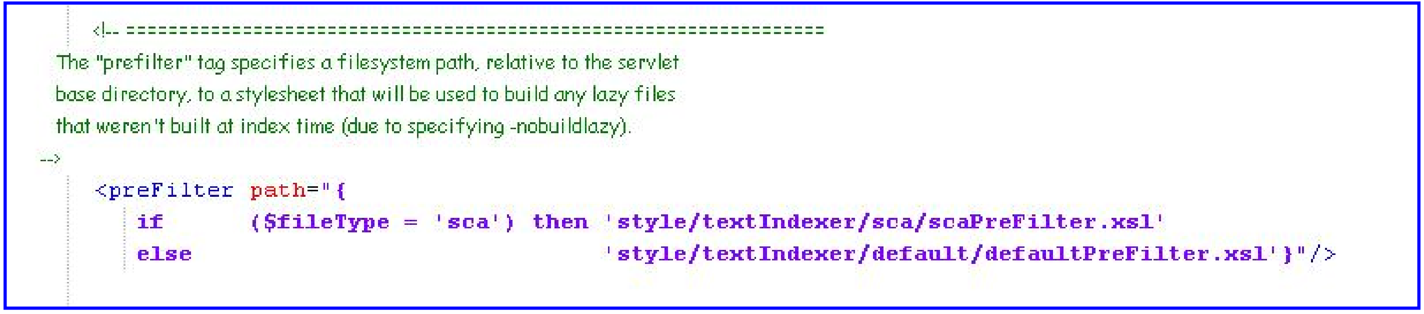
If the XML record contains *sca_record* within the root element, the *fileType* variable is assigned the value *sca*.



```
<xsl:variable name="file" select="concat('../../data/',$docId)"/>
<xsl:variable name="fileType">
   <xsl:for-each select="FileUtils:readXMLStub($file)">

      <xsl:variable name="root-element-name" select="name(*[1])"/>
      <xsl:variable name="pid" select="unparsed-entity-public-id($root-element-name)"/>
      <xsl:variable name="uri" select="unparsed-entity-uri($root-element-name)"/>
      <xsl:variable name="ns" select="namespace-uri(*[1])"/>

      <xsl:choose>
      <!-- Look for SCA XML files -->
         <xsl:when test="matches($root-element-name,'sca_record')">
            <xsl:value-of select="'sca'"/>
         </xsl:when>
      </xsl:choose>
   </xsl:for-each>
</xsl:variable>
```

Associating scaPreFilter.xsl with SCA XML records (for indexing)

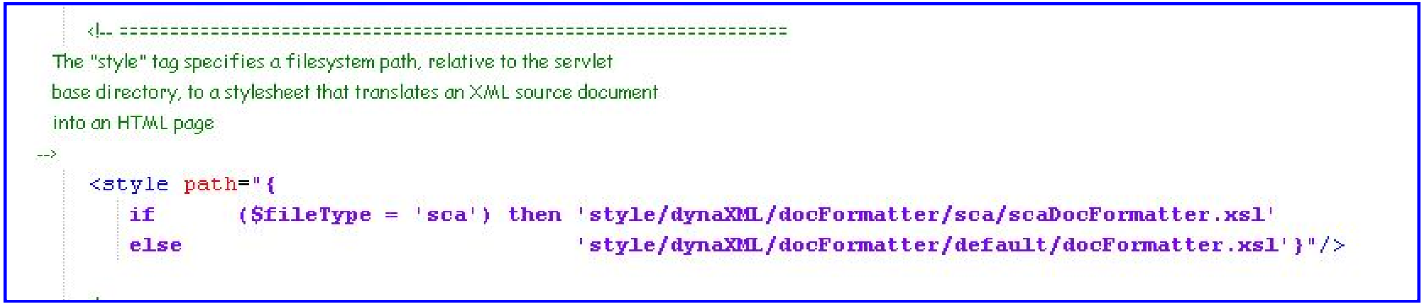
If the *fileType* variable contains the value *sca*, *scaPreFilter.xsl* will be applied to the file.



```
    <!-- ===================================================================
  The "prefilter" tag specifies a filesystem path, relative to the servlet
  base directory, to a stylesheet that will be used to build any lazy files
  that weren't built at index time (due to specifying -nobuildlazy).
-->
    <preFilter path="{
       if       ($fileType = 'sca') then 'style/textIndexer/sca/scaPreFilter.xsl'
       else                              'style/textIndexer/default/defaultPreFilter.xsl'}"/>
```

Associating scaDocFormatter.xsl with SCA XML records (for presentation as html pages)

If the *fileType* variable contains the value *sca*, *scaDocFormatter.xsl* will be applied to the file.



```
    <!-- ===================================================================
  The "style" tag specifies a filesystem path, relative to the servlet
  base directory, to a stylesheet that translates an XML source document
  into an HTML page
-->
    <style path="{
       if       ($fileType = 'sca') then 'style/dynaXML/docFormatter/sca/scaDocFormatter.xsl'
       else                              'style/dynaXML/docFormatter/default/docFormatter.xsl'}"/>
```

Next… Displaying brief search results

Displaying brief search results

style/crossQuery/resultFormatter/default/resultFormatter.xsl

For a brief record display showing an image thumbnail and metadata, most of the requirements are met by XTF's default *resultFormatter.xsl* stylesheet.

- Adding a column for image thumbnails
- Making thumbnails clickable
- Altering metadata field labels
- Screenshot: Default brief record display
- Screenshot: Modified brief record display

Adding a column for the image thumbnails

Within the *docHit* template, the default XTF brief record display is structured as an html table. In the first row I inserted an additional cell between the record result number and the *Artist* metadata field. I made the cell a fixed width and included a rowspan attribute to ensure that all of the associated metadata fields would align to the right of the image.

```
<!-- Thumbnail image row -->
<td rowspan="20" width="250px" align "top" valign"middle">
```

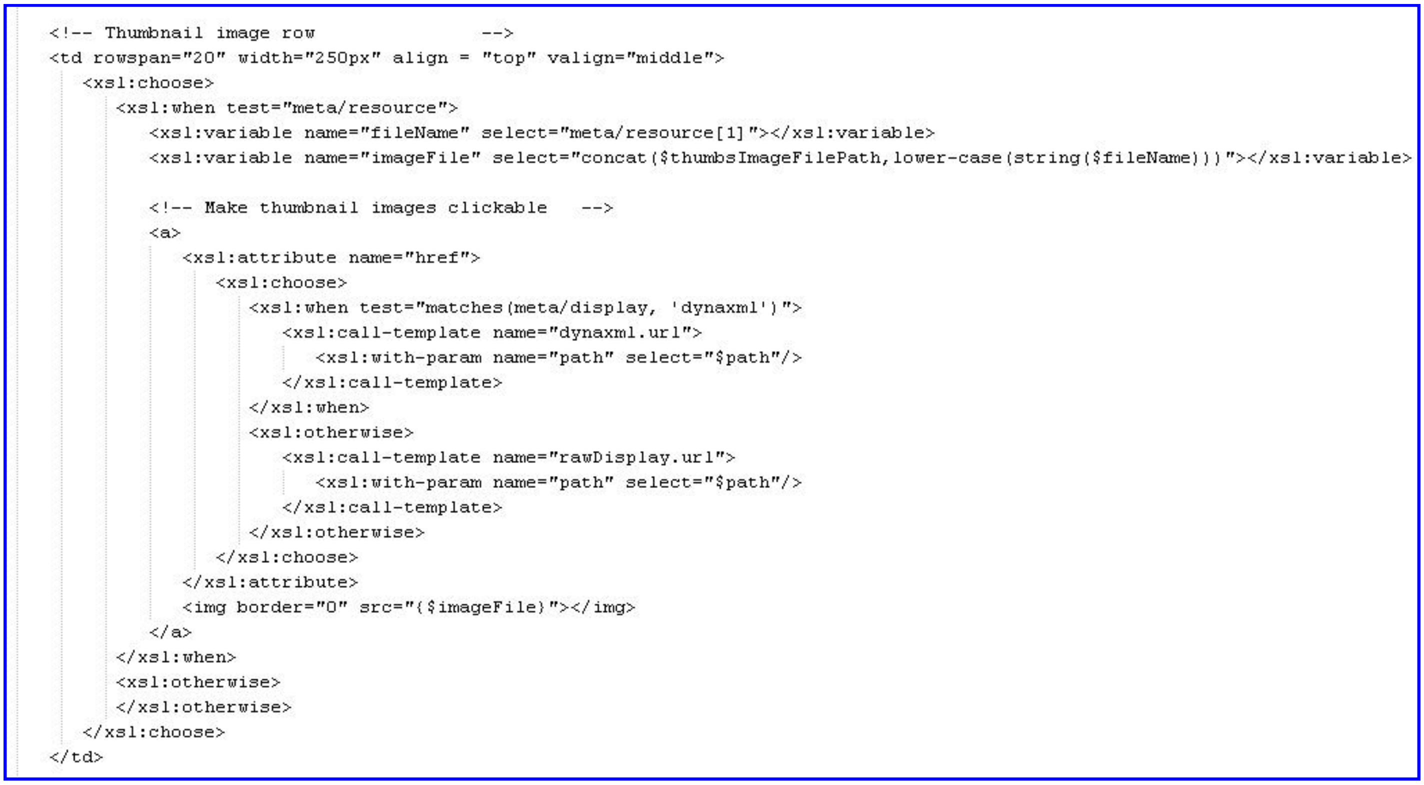
Making the thumbnails clickable

For each retrieved record, XTF needs to know where to find the associated thumbnail image file. In the *Local parameters* section I added a parameter named *thumbsImageFilePath*.

```
<!-- sca thumbnail images-->

<xsl:param name="thumbsImageFilePath" select="concat($xtfURL, 'media/sca/images/thumbs')"/>
```

Within the *docHit* template, I used a variable *fileName* to reference the name of the thumbnail file associated with the retrieved metadata record. For the SCA metadata records, the filename is contained within the *resource* metadata field. By concatenating the *fileName* and *thumbsImageFilePath* within the variable *imageFile*, I provide a thumbnail reference specific to each retrieved record. Each time the template renders a brief record, it has sufficient information (contained within the *imageFile* variable) to find the location of that record's thumbnail.

To make the thumbnail clickable, I copied the xsl already provided for the default clickable title field.

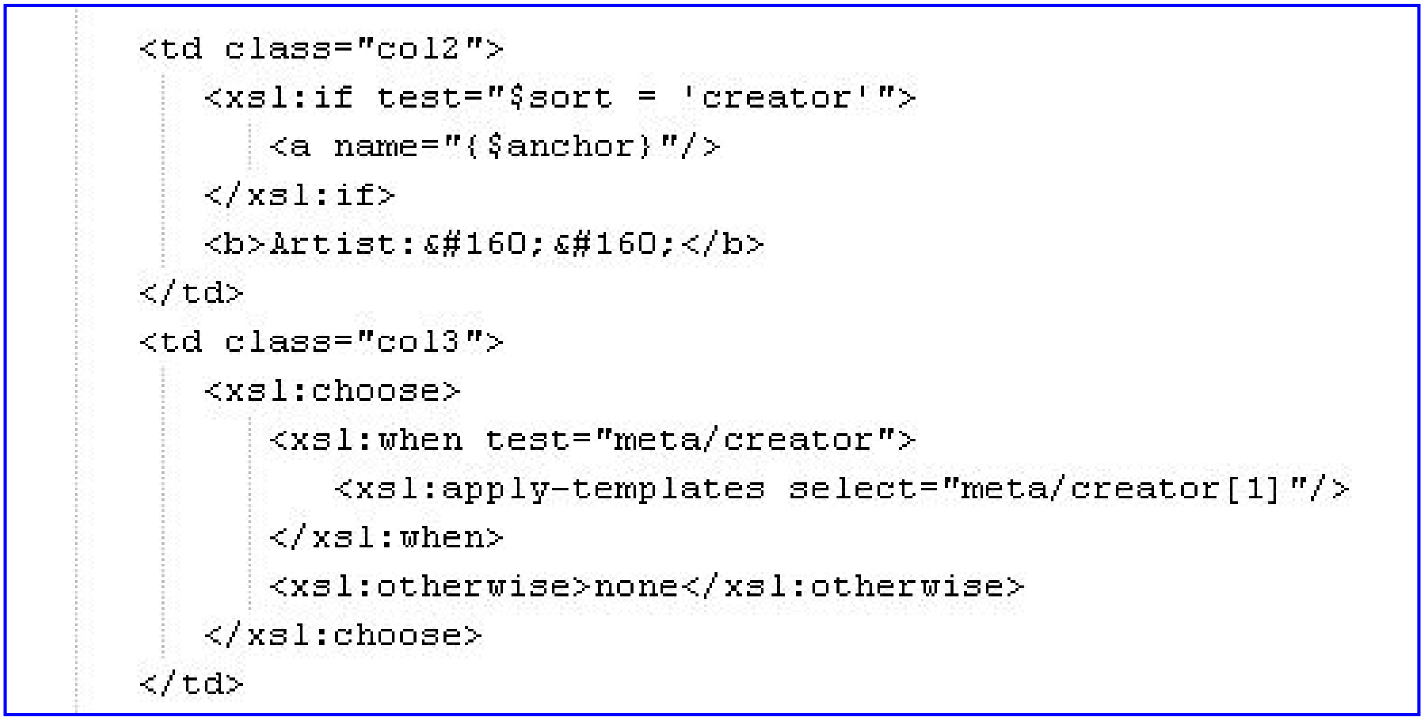


```
<!-- Thumbnail image row                -->
<td rowspan="20" width="250px" align = "top" valign="middle">
   <xsl:choose>
      <xsl:when test="meta/resource">
         <xsl:variable name="fileName" select="meta/resource[1]"></xsl:variable>
         <xsl:variable name="imageFile" select="concat($thumbsImageFilePath,lower-case(string($fileName)))"></xsl:variable>

         <!-- Make thumbnail images clickable   -->
         <a>
            <xsl:attribute name="href">
               <xsl:choose>
                  <xsl:when test="matches(meta/display, 'dynaxml')">
                     <xsl:call-template name="dynaxml.url">
                        <xsl:with-param name="path" select="$path"/>
                     </xsl:call-template>
                  </xsl:when>
                  <xsl:otherwise>
                     <xsl:call-template name="rawDisplay.url">
                        <xsl:with-param name="path" select="$path"/>
                     </xsl:call-template>
                  </xsl:otherwise>
               </xsl:choose>
            </xsl:attribute>
            <img border="0" src="{$imageFile}"></img>
         </a>
      </xsl:when>
      <xsl:otherwise>
      </xsl:otherwise>
   </xsl:choose>
</td>
```

For each thumbnail, its target page is a *full record* display. For more information, see displaying full records.

Altering metadata field display labels

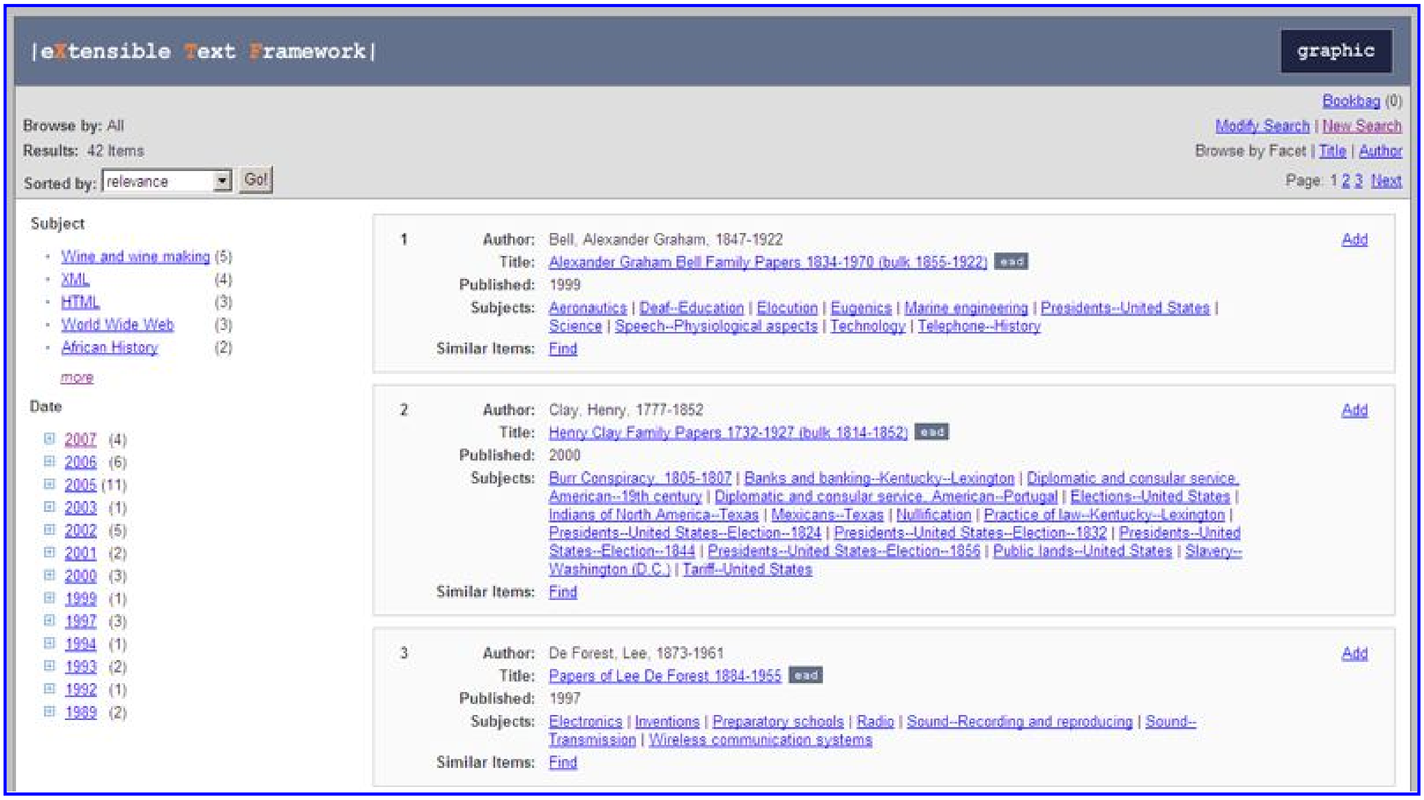
XTF's default brief record layout includes an *Author* metadata label. In the example below, taken from the *docHit* template, I change the label to *Artist*. Note that the metadata tag is named *creator (*reflecting the metadata element tag definition contained in *scaPreFilter.xsl*. See identifying metadata elements for indexing and faceted browsing*)*. Metadata display labels may be different than metadata tag names.



```
<td class="col2">
   <xsl:if test="$sort = 'creator'">
      <a name="{$anchor}"/>
   </xsl:if>
   <b>Artist:&#160;&#160;</b>
</td>
<td class="col3">
   <xsl:choose>
      <xsl:when test="meta/creator">
         <xsl:apply-templates select="meta/creator[1]"/>
      </xsl:when>
      <xsl:otherwise>none</xsl:otherwise>
   </xsl:choose>
</td>
```

Screenshot: Default brief record display

Default brief record display showing EAD records.



Browse by: All
Results: 42 Items
Sorted by: relevance

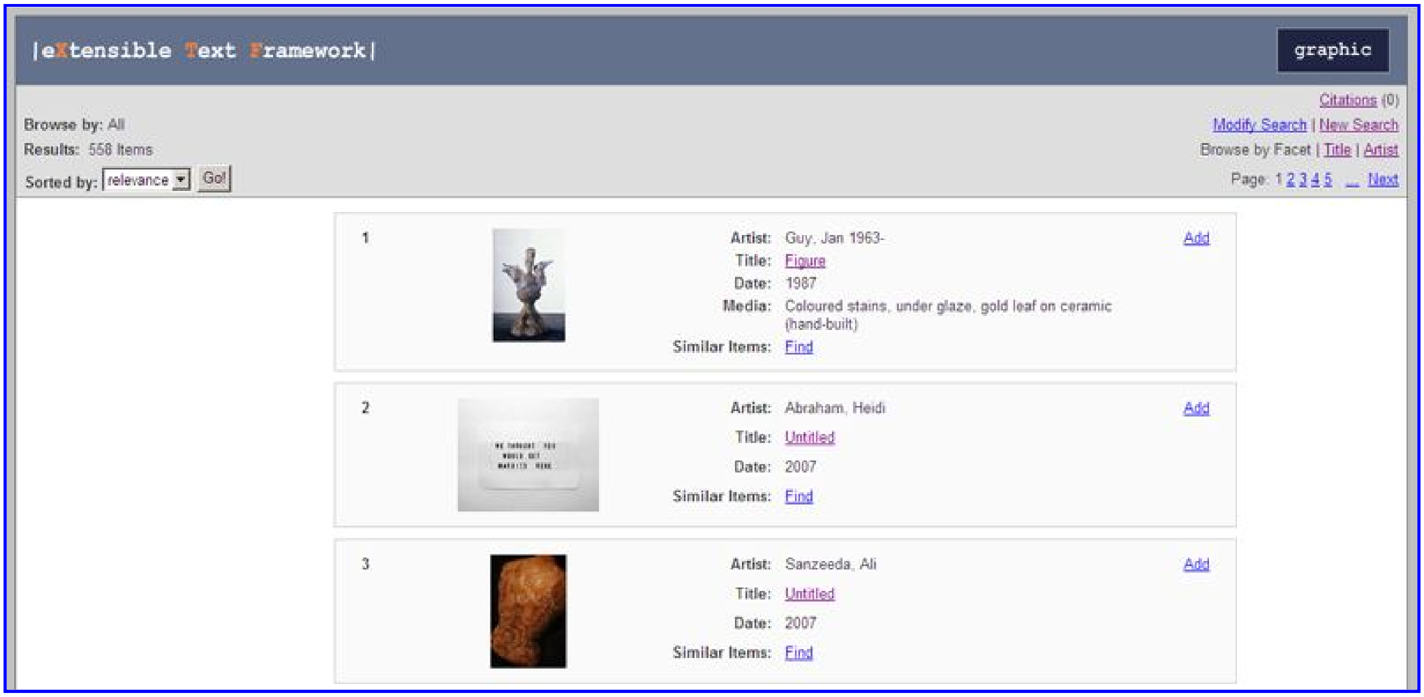
1
Author: Bell, Alexander Graham, 1847-1922
Title: Alexander Graham Bell Family Papers 1834-1970 (bulk 1855-1922)
Published: 1999
Subjects: Aeronautics | Deaf--Education | Elocution | Eugenics | Marine engineering | Presidents--United States | Science | Speech--Physiological aspects | Technology | Telephone--History
Similar Items: Find

2
Author: Clay, Henry, 1777-1852
Title: Henry Clay Family Papers 1732-1927 (bulk 1814-1852)
Published: 2000
Subjects: Burr Conspiracy, 1805-1807 | Banks and banking--Kentucky--Lexington | Diplomatic and consular service, American--19th century | Diplomatic and consular service, American--Portugal | Elections--United States | Indians of North America--Texas | Mexicans--Texas | Nullification | Practice of law--Kentucky--Lexington | Presidents--United States--Election--1824 | Presidents--United States--Election--1832 | Presidents--United States--Election--1844 | Presidents--United States--Election--1856 | Public lands--United States | Slavery--Washington (D.C.) | Tariff--United States
Similar Items: Find

3
Author: De Forest, Lee, 1873-1961
Title: Papers of Lee De Forest 1884-1955
Published: 1997
Subjects: Electronics | Inventions | Preparatory schools | Radio | Sound--Recording and reproducing | Sound--Transmission | Wireless communication systems
Similar Items: Find

Screenshot: Modified brief record display

Modified brief record display illustrating the addition of image thumbnails and SCA metadata elements. Note that facets are not yet displaying. Although metadata elements have been identified for faceted browsing (described in Identifying metadata elements for indexing and faceted browsing), additional work is required to produce facets (*see* sorting, browsing and facets).


|eXtensible Text Framework|
graphic
Citations (0)
Modify Search | New Search
Browse by Facet | Title | Artist
Page: 1 2 3 4 5 ... Next
Go!
Add
Add
Add

Browse by: All
Results: 558 Items
Sorted by: relevance

1
Artist: Guy, Jan 1963-
Title: Figure
Date: 1987
Media: Coloured stains, under glaze, gold leaf on ceramic (hand-built)
Similar Items: Find

2
Artist: Abraham, Heidi
Title: Untitled
Date: 2007
Similar Items: Find

3
Artist: Sanzeeda, Ali
Title: Untitled
Date: 2007
Similar Items: Find

In the above screenshot, *Bookbag* has changed to *Citations*. This is another simple change to display text, along the lines of altering a metadata field label.

Next… Displaying full records

Displaying full records

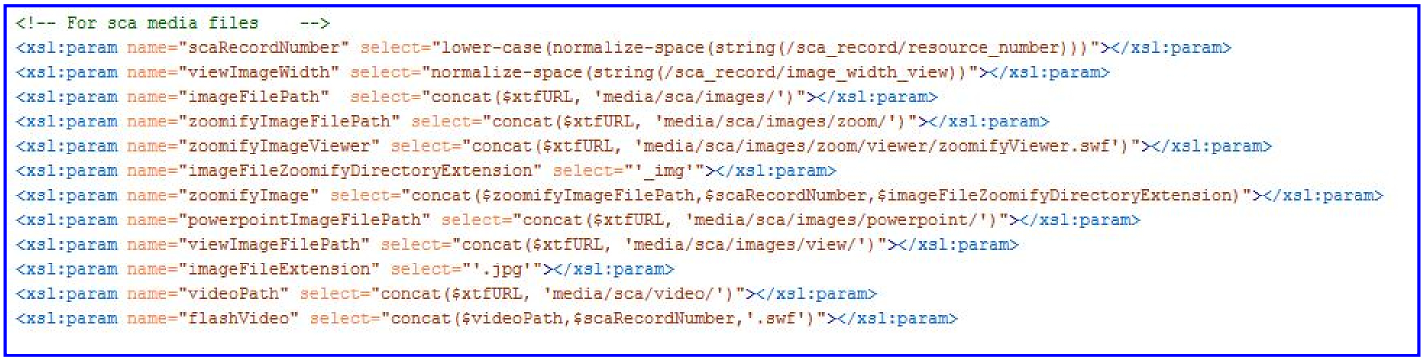
style/dynaXML/docFormatter/sca/scaDocFormatter.xsl

In associating stylesheets with metadata records, I describe how a relationship is signified between a particular stylesheet and its associated set of XML records. All SCA XML records are associated with the *scaDocFormatter* stylesheet. When a user clicks on a title hyperlink or image thumbnail within a brief record, XTF has sufficient information to identify the target XML file and its related stylesheet, applying the stylesheet to the XML to render html.

- Defining parameters to save time and ease maintenance
- Selecting a template matching requested layout
- Formatting a full record
- Screenshot: Full record display
- Incorporating additional navigation options on a full record page
- Screenshot: Full record with button-bar navigation options
- Generating hyperlinks to other types of images
- Generating hyperlinks to video
- Including header and footer information

Defining parameters to save time and ease maintenance

The *scaDocFormatter.xsl* stylesheet is based on *nlmDocFormatter.xsl* included in the default installation, with the addition of a number of parameters containing information such as the location of types of image and video files. Using parameters provides a means of maintaining in one place information that might be used in multiple locations within a stylesheet. As an example, if I wish to reference the path to the location of the powerpoint versions of the image files, it is easier to use the shorthand $powerpointImageFilePath than writing out the complete directory path each time. Later if I decide to change the location of the files, I need only alter the location information once within the definition of the parameter.

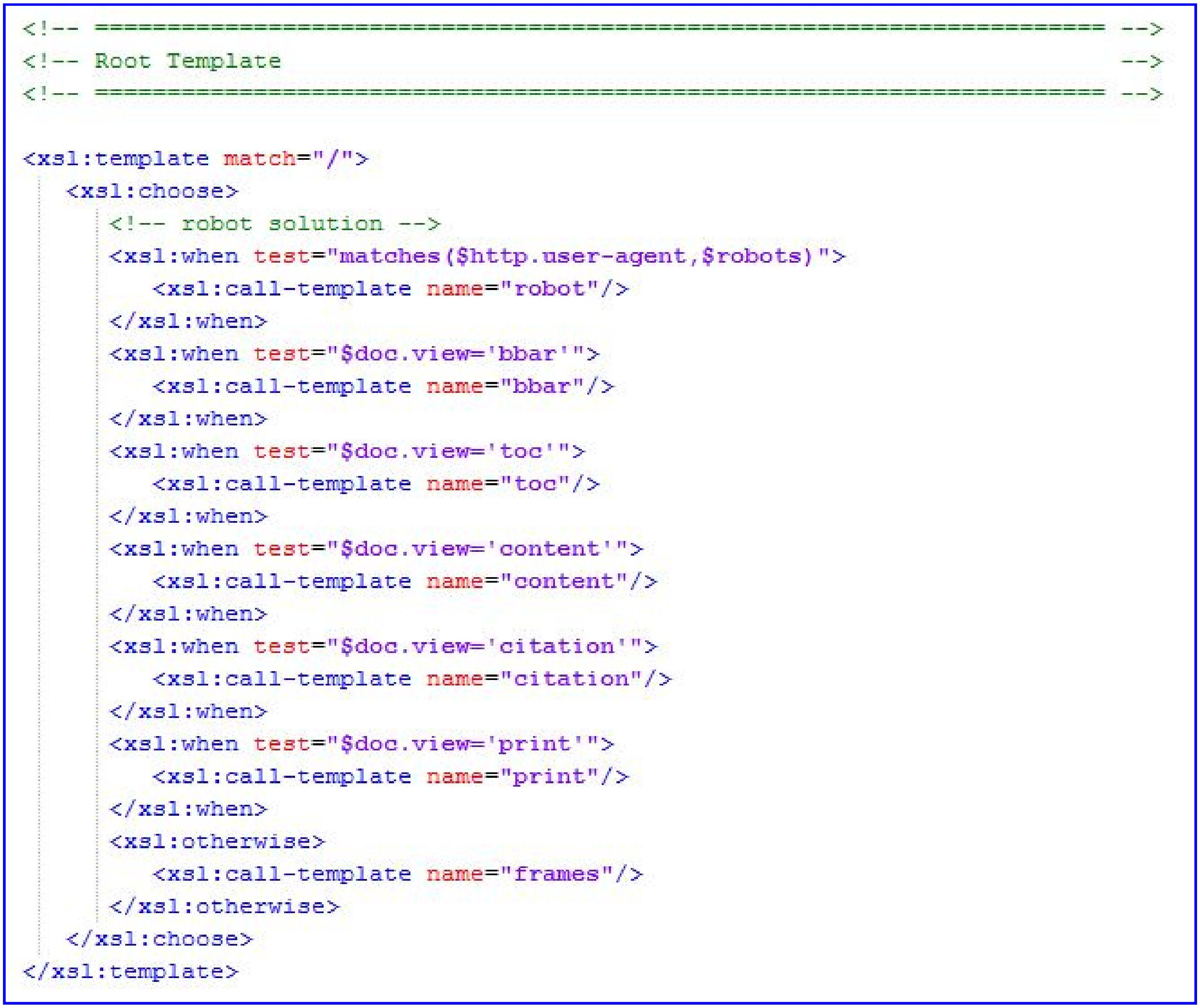


```
<!-- For sca media files    -->
<xsl:param name="scaRecordNumber" select="lower-case(normalize-space(string(/sca_record/resource_number)))"></xsl:param>
<xsl:param name="viewImageWidth" select="normalize-space(string(/sca_record/image_width_view))"></xsl:param>
<xsl:param name="imageFilePath"  select="concat($xtfURL, 'media/sca/images/')"></xsl:param>
<xsl:param name="zoomifyImageFilePath" select="concat($xtfURL, 'media/sca/images/zoom/')"></xsl:param>
<xsl:param name="zoomifyImageViewer" select="concat($xtfURL, 'media/sca/images/zoom/viewer/zoomifyViewer.swf')"></xsl:param>
<xsl:param name="imageFileZoomifyDirectoryExtension" select="'_img'"></xsl:param>
<xsl:param name="zoomifyImage" select="concat($zoomifyImageFilePath,$scaRecordNumber,$imageFileZoomifyDirectoryExtension)"></xsl:param>
<xsl:param name="powerpointImageFilePath" select="concat($xtfURL, 'media/sca/images/powerpoint/')"></xsl:param>
<xsl:param name="viewImageFilePath" select="concat($xtfURL, 'media/sca/images/view/')"></xsl:param>
<xsl:param name="imageFileExtension" select="'.jpg'"></xsl:param>
<xsl:param name="videoPath" select="concat($xtfURL, 'media/sca/video/')"></xsl:param>
<xsl:param name="flashVideo" select="concat($videoPath,$scaRecordNumber,'.swf')"></xsl:param>
```

Selecting a template matching requested layout

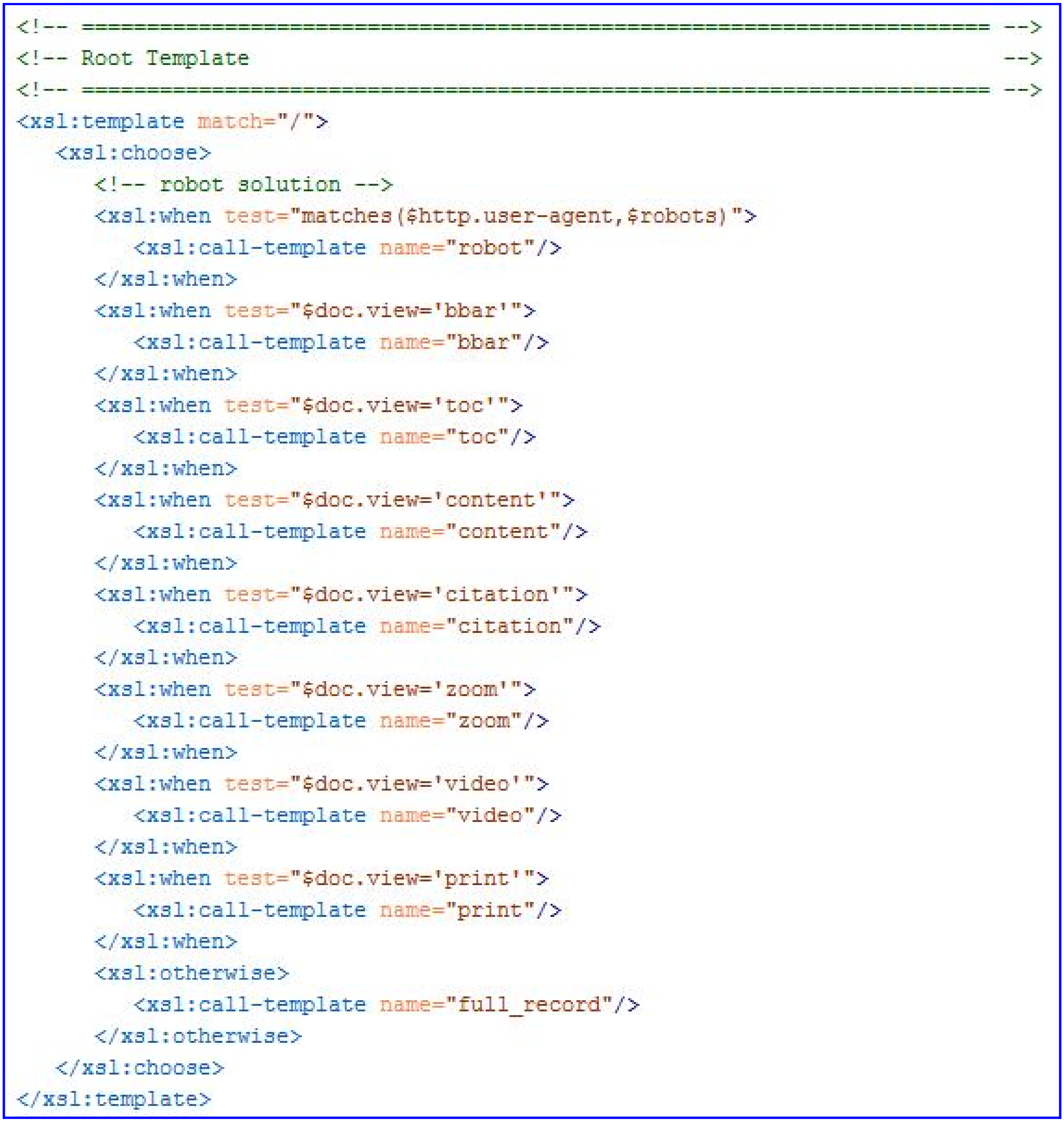
The root template in the default nlmDocFormatter stylesheet uses a choose element to provide options for dealing with requests to see various types of layout, such as table of contents, citation or print view. Each choice corresponds to a template which formats the content to match the desired view. At this stage I intend to include templates for print, citation, full record zoom (for display of zoomified versions of the images) and video (for presenting Flash files). I'm not using the frames template and the default view will be full record.

Default root template



```
<!-- ====================================================================== -->
<!-- Root Template                                                          -->
<!-- ====================================================================== -->

<xsl:template match="/">
   <xsl:choose>
      <!-- robot solution -->
      <xsl:when test="matches($http.user-agent,$robots)">
         <xsl:call-template name="robot"/>
      </xsl:when>
      <xsl:when test="$doc.view='bbar'">
         <xsl:call-template name="bbar"/>
      </xsl:when>
      <xsl:when test="$doc.view='toc'">
         <xsl:call-template name="toc"/>
      </xsl:when>
      <xsl:when test="$doc.view='content'">
         <xsl:call-template name="content"/>
      </xsl:when>
      <xsl:when test="$doc.view='citation'">
         <xsl:call-template name="citation"/>
      </xsl:when>
      <xsl:when test="$doc.view='print'">
         <xsl:call-template name="print"/>
      </xsl:when>
      <xsl:otherwise>
         <xsl:call-template name="frames"/>
      </xsl:otherwise>
   </xsl:choose>
</xsl:template>
```

Modified root template showing additional zoom and video options and default setting for full record.

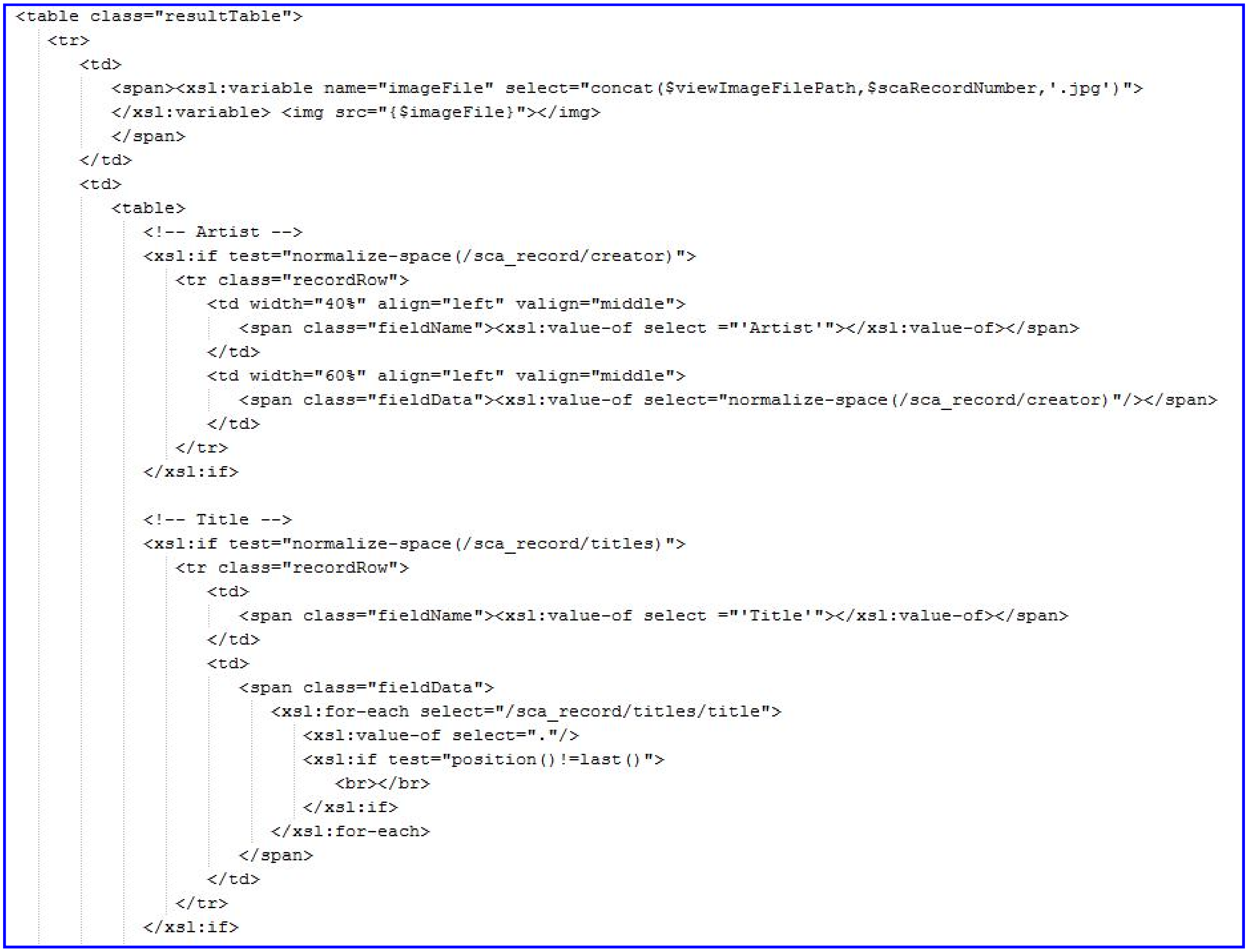


```
<!-- ====================================================================== -->
<!-- Root Template                                                          -->
<!-- ====================================================================== -->
<xsl:template match="/">
   <xsl:choose>
      <!-- robot solution -->
      <xsl:when test="matches($http.user-agent,$robots)">
         <xsl:call-template name="robot"/>
      </xsl:when>
      <xsl:when test="$doc.view='bbar'">
         <xsl:call-template name="bbar"/>
      </xsl:when>
      <xsl:when test="$doc.view='toc'">
         <xsl:call-template name="toc"/>
      </xsl:when>
      <xsl:when test="$doc.view='content'">
         <xsl:call-template name="content"/>
      </xsl:when>
      <xsl:when test="$doc.view='citation'">
         <xsl:call-template name="citation"/>
      </xsl:when>
      <xsl:when test="$doc.view='zoom'">
         <xsl:call-template name="zoom"/>
      </xsl:when>
      <xsl:when test="$doc.view='video'">
         <xsl:call-template name="video"/>
      </xsl:when>
      <xsl:when test="$doc.view='print'">
         <xsl:call-template name="print"/>
      </xsl:when>
      <xsl:otherwise>
         <xsl:call-template name="full_record"/>
      </xsl:otherwise>
   </xsl:choose>
</xsl:template>
```

Formatting a full record

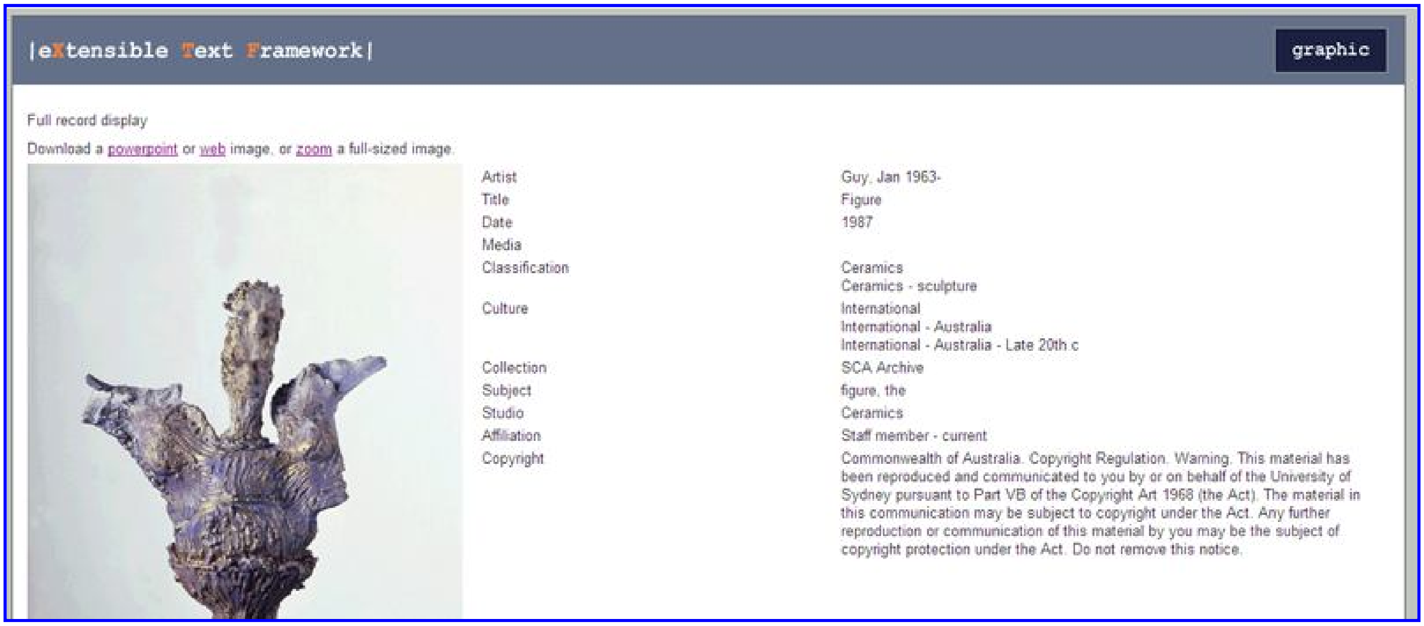
Whenever *scaDocFormatter.xsl* is applied to an SCA XML record (such as when a user clicks on an image thumbnail in a brief record or its accompanying brief record title), the content is rendered using the full record template. An SCA full record comprises a number of elements such as artist, title, classification, culture, subject and copyright. For each element, the full record template tests whether the requested XML record contains the required information. In each case, if the information exists a field label is displayed along with the accompanying metadata.

Extract from the full record template illustrating the display of metadata elements dependent on their occurrence within an SCA XML record



```
<table class="resultTable">
    <tr>
        <td>
            <span><xsl:variable name="imageFile" select="concat($viewImageFilePath,$scaRecordNumber,'.jpg')">
            </xsl:variable> <img src="{$imageFile}"></img>
            </span>
        </td>
        <td>
            <table>
                <!-- Artist -->
                <xsl:if test="normalize-space(/sca_record/creator)">
                    <tr class="recordRow">
                        <td width="40%" align="left" valign="middle">
                            <span class="fieldName"><xsl:value-of select ="'Artist'"></xsl:value-of></span>
                        </td>
                        <td width="60%" align="left" valign="middle">
                            <span class="fieldData"><xsl:value-of select="normalize-space(/sca_record/creator)"/></span>
                        </td>
                    </tr>
                </xsl:if>

                <!-- Title -->
                <xsl:if test="normalize-space(/sca_record/titles)">
                    <tr class="recordRow">
                        <td>
                            <span class="fieldName"><xsl:value-of select ="'Title'"></xsl:value-of></span>
                        </td>
                        <td>
                            <span class="fieldData">
                                <xsl:for-each select="/sca_record/titles/title">
                                    <xsl:value-of select="."/>
                                    <xsl:if test="position()!=last()">
                                        <br></br>
                                    </xsl:if>
                                </xsl:for-each>
                            </span>
                        </td>
                    </tr>
                </xsl:if>
```

Screenshot: Full record display



Full record display

Download a powerpoint or web image, or zoom a full-sized image.

Artist	Guy, Jan 1963-
Title	Figure
Date	1987
Media	
Classification	Ceramics Ceramics - sculpture
Culture	International International - Australia International - Australia - Late 20th c
Collection	SCA Archive
Subject	figure, the
Studio	Ceramics
Affiliation	Staff member - current
Copyright	Commonwealth of Australia. Copyright Regulation. Warning. This material has been reproduced and communicated to you by or on behalf of the University of Sydney pursuant to Part VB of the Copyright Art 1968 (the Act). The material in this communication may be subject to copyright under the Act. Any further reproduction or communication of this material by you may be the subject of copyright protection under the Act. Do not remove this notice.

In the example, despite using a test to display a *media* field label only on occasions when accompanying metadata exists, the label is displaying against an apparently empty field. It could be the case that the source metadata record contains characters in the media field which although not displaying, pass the test by the fact of their existence. A more effective test may involve a simple regular expression to make field label display dependent on the occurance of alphabetic or numeric characters.

Incorporating additional navigation options on a full record page

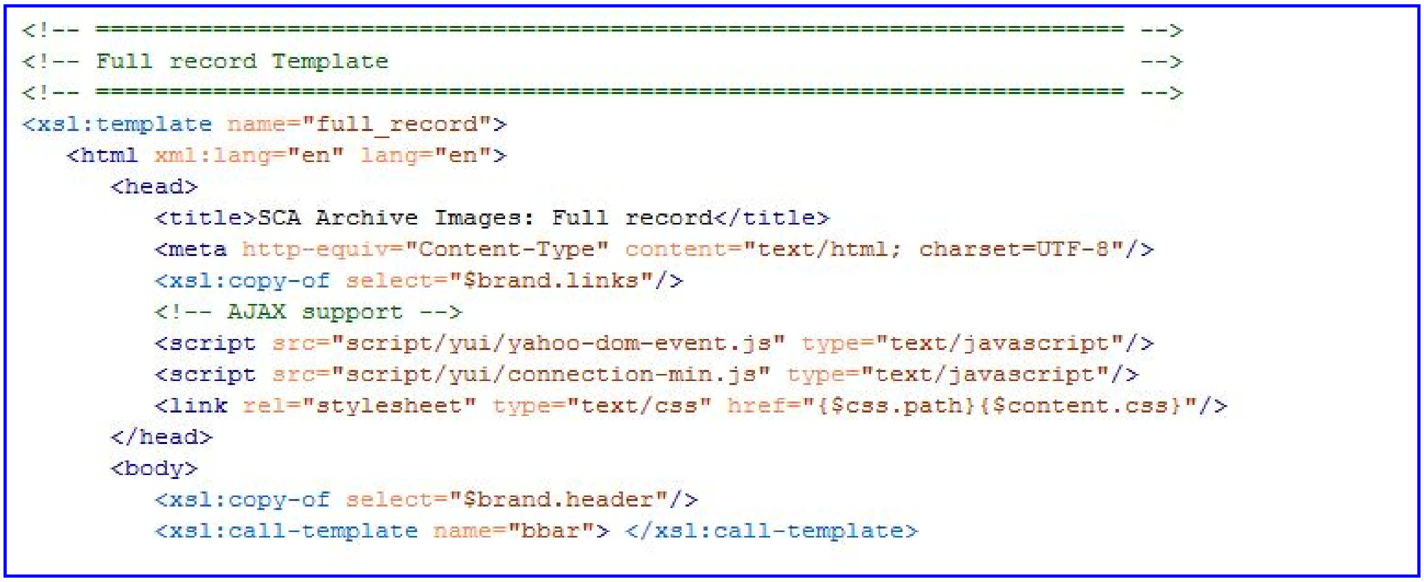
The above screenshot includes no option to return to the previous page. This can be remedied by including a reference to XTF's button bar. By default the button bar provides links to print and citation displays, return to search results and home. The template can be called using the following instruction.

```
<xsl:call-template name="bbar"> </xsl:call-template>
```

Although the template is included in a separate stylesheet, it can be called from within scaDocformatter.xsl, as it has been imported.

```
<xsl:import href="../common/docFormatterCommon.xsl"/>
```

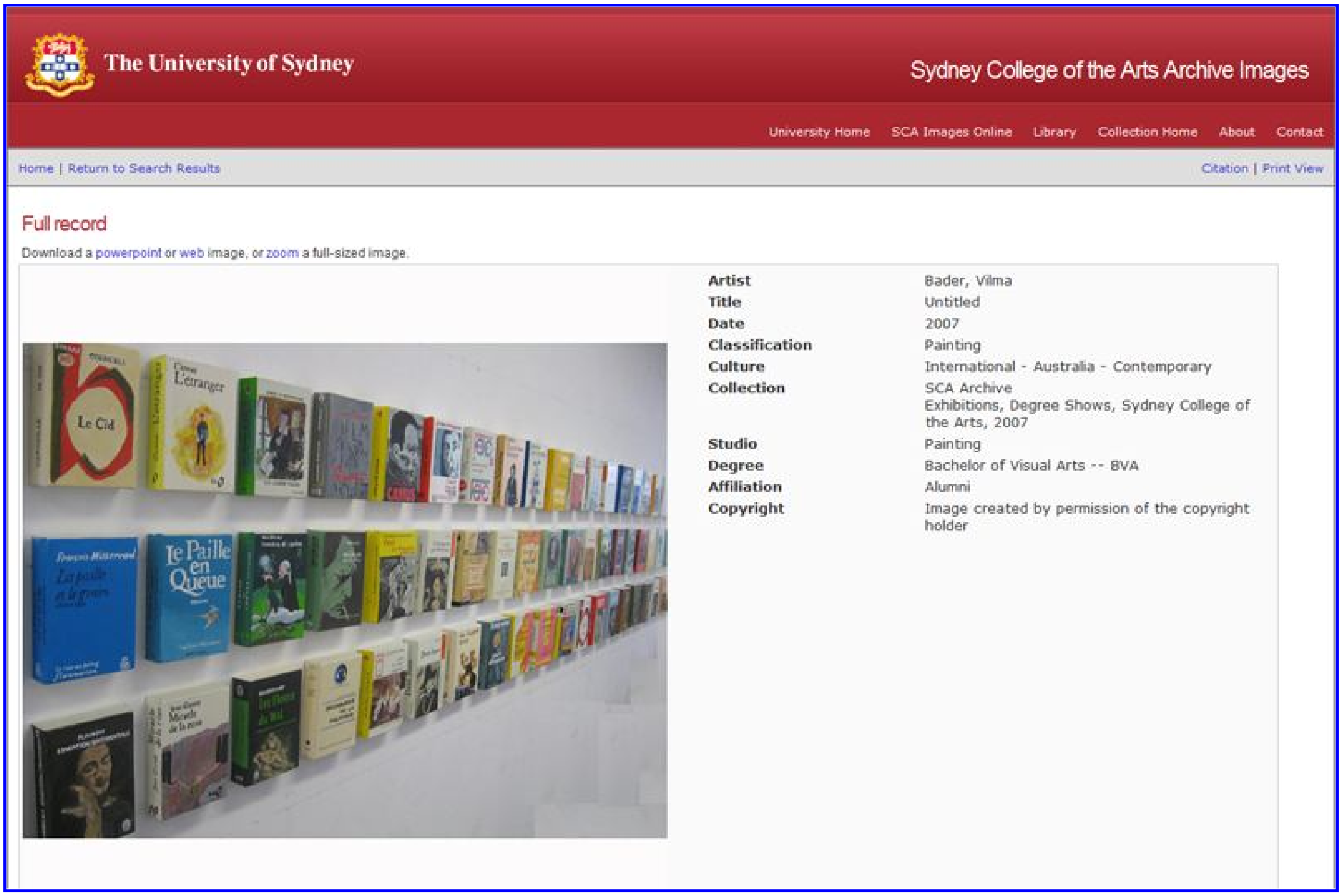
Screenshot including reference to the button bar template (bbar).



```
<!-- ====================================================================== -->
<!-- Full record Template                                                   -->
<!-- ====================================================================== -->
<xsl:template name="full_record">
   <html xml:lang="en" lang="en">
      <head>
         <title>SCA Archive Images: Full record</title>
         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
         <xsl:copy-of select="$brand.links"/>
         <!-- AJAX support -->
         <script src="script/yui/yahoo-dom-event.js" type="text/javascript"/>
         <script src="script/yui/connection-min.js" type="text/javascript"/>
         <link rel="stylesheet" type="text/css" href="{$css.path}{$content.css}"/>
      </head>
      <body>
         <xsl:copy-of select="$brand.header"/>
         <xsl:call-template name="bbar"> </xsl:call-template>
```

Screenshot: Full record with button-bar navigation options

The following image was captured after the site was styled



The University of Sydney

Sydney College of the Arts Archive Images

Full record

Download a powerpoint or web image, or zoom a full-sized image.

Artist	Bader, Vilma
Title	Untitled
Date	2007
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Painting
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

By default, XTF's button bar includes a search box (for searching within the currently displayed text). As this isn't required for the SCA collection, it is commented out from the template (/style/dynaXML/docFormatter/Common /docFormatterCommon.xsl).

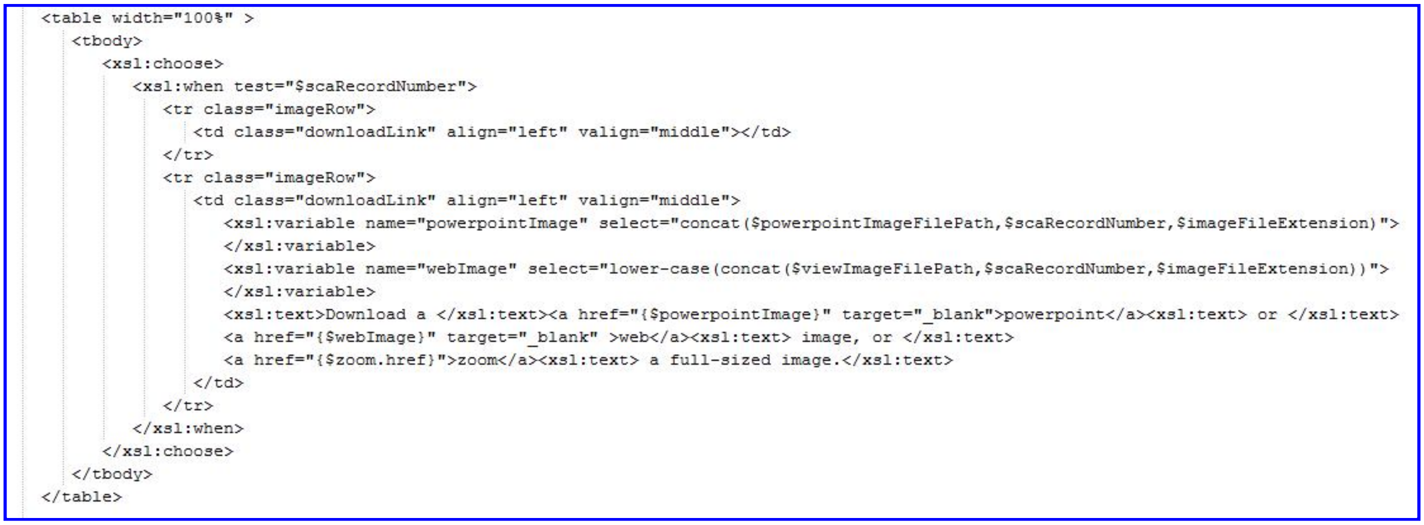
Generating hyperlinks to other types of images

The screenshot above includes hyperlinks to *powerpoint*, *web* and *zoomified* image displays. Each image filename comprises a record number and a file extension. (eg. sca3501-1.jpg). As described in installation and setup, each set of images is contained in a separate directory. To create the hyperlink targets, the full record template concatenates the directory path, record number and file extension (each of which is specified as a parameter, as described earlier in this section).

The hyperlink references a variable containing information identifying the record currently being displayed, and the desired view option.

```
<xsl:variable name="zoom.href">
  <xsl:value-of select="$xtfURL"/>view?<xsl:value-of select="$query.string"/>;doc.view=zoom
</xsl:variable>
<a href="{$zoom.href}">zoom</a><xsl:text> a full-sized image.</xsl:text>
```

Extract from full record template illustrating the creation of hyperlink targets for display of *powerpoint*, *web* and *zoomified* images.



```
<table width="100%" >
   <tbody>
      <xsl:choose>
         <xsl:when test="$scaRecordNumber">
            <tr class="imageRow">
               <td class="downloadLink" align="left" valign="middle"></td>
            </tr>
            <tr class="imageRow">
               <td class="downloadLink" align="left" valign="middle">
                  <xsl:variable name="powerpointImage" select="concat($powerpointImageFilePath,$scaRecordNumber,$imageFileExtension)">
                  </xsl:variable>
                  <xsl:variable name="webImage" select="lower-case(concat($viewImageFilePath,$scaRecordNumber,$imageFileExtension))">
                  </xsl:variable>
                  <xsl:text>Download a </xsl:text><a href="{$powerpointImage}" target="_blank">powerpoint</a><xsl:text> or </xsl:text>
                  <a href="{$webImage}" target="_blank" >web</a><xsl:text> image, or </xsl:text>
                  <a href="{$zoom.href}">zoom</a><xsl:text> a full-sized image.</xsl:text>
               </td>
            </tr>
         </xsl:when>
      </xsl:choose>
   </tbody>
</table>
```

Using the above technique, clicking on a hyperlink for a zoomified image displays a static html page pre-generated using photoshop's zoomify export. A more flexible approach is to use a template to generate the html wrapper for the zoomified image via XTF. This enables dynamic incorporation of image metadata within the page. (See zoomable image display).

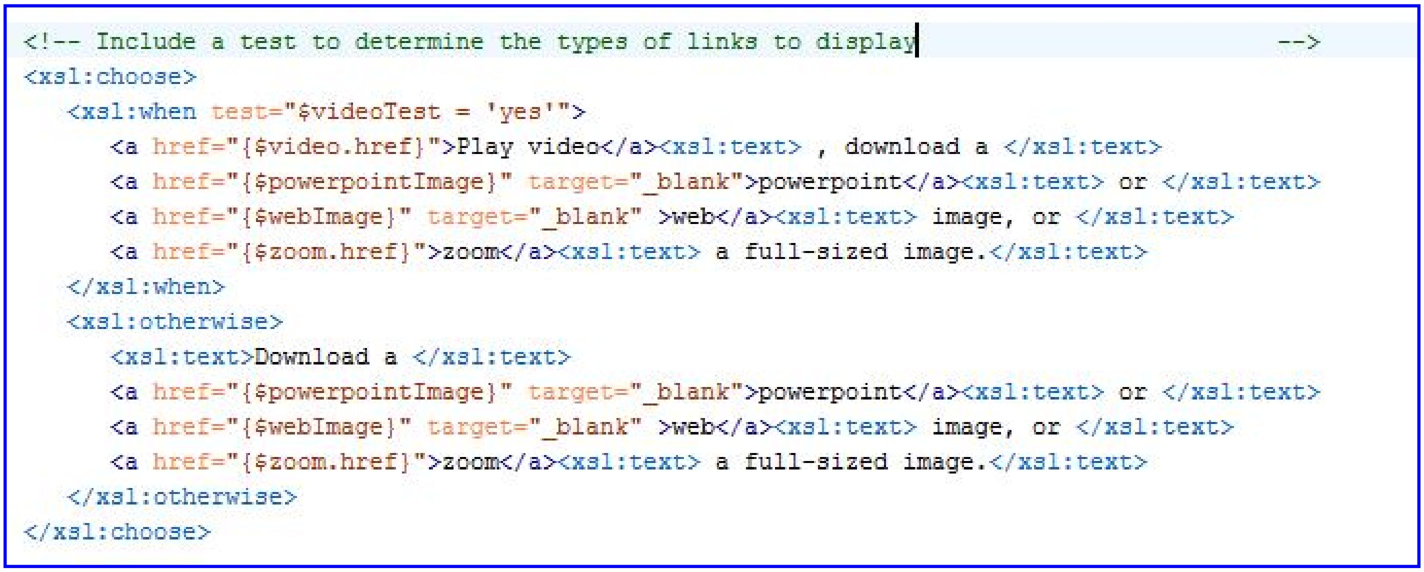
Generating hyperlinks to video

Although every record is associated with a common set of images (including thumb and zoom), only a small number describe video files. Metadata records contain information about available format types (such as digital image and video), and I use the occurrence of this information to control hyperlink display. Catering for conditional inclusion of video hyperlinks involves a slight modification of the instructions covered in the previous section.

Within the full record template, a hyperlink references a variable containing information identifying the record currently being displayed, and the desired view option.

```
<xsl:variable name="videoTest" select="/sca_record/video_available"></xsl:variable>

<xsl:variable name="video.href">
  <xsl:value-of select="$xtfURL"/>view?<xsl:value-of select="$query.string"/>;doc.view=video

</xsl:variable>
```

Extract from full record template illustrating the use of a test for conditional display of video playback hyperlinks.



```
<!-- Include a test to determine the types of links to display                    -->
<xsl:choose>
   <xsl:when test="$videoTest = 'yes'">
      <a href="{$video.href}">Play video</a><xsl:text> , download a </xsl:text>
      <a href="{$powerpointImage}" target="_blank">powerpoint</a><xsl:text> or </xsl:text>
      <a href="{$webImage}" target="_blank" >web</a><xsl:text> image, or </xsl:text>
      <a href="{$zoom.href}">zoom</a><xsl:text> a full-sized image.</xsl:text>
   </xsl:when>
   <xsl:otherwise>
      <xsl:text>Download a </xsl:text>
      <a href="{$powerpointImage}" target="_blank">powerpoint</a><xsl:text> or </xsl:text>
      <a href="{$webImage}" target="_blank" >web</a><xsl:text> image, or </xsl:text>
      <a href="{$zoom.href}">zoom</a><xsl:text> a full-sized image.</xsl:text>
   </xsl:otherwise>
</xsl:choose>
```

Including header and footer information

In the full record screenshot above, the header is the blue section along the top of the page. The header and footer are displayed by including within the full record template references to $brand.header and $brand.footer.

```
<xsl:copy-of select="$brand.header"/>
<xsl:copy-of select="$brand.footer"/>
```

The header and footer are defined within brand\default.xml. (For more information, see styling the site).

Next… Zoomable image display

Zoomable image display

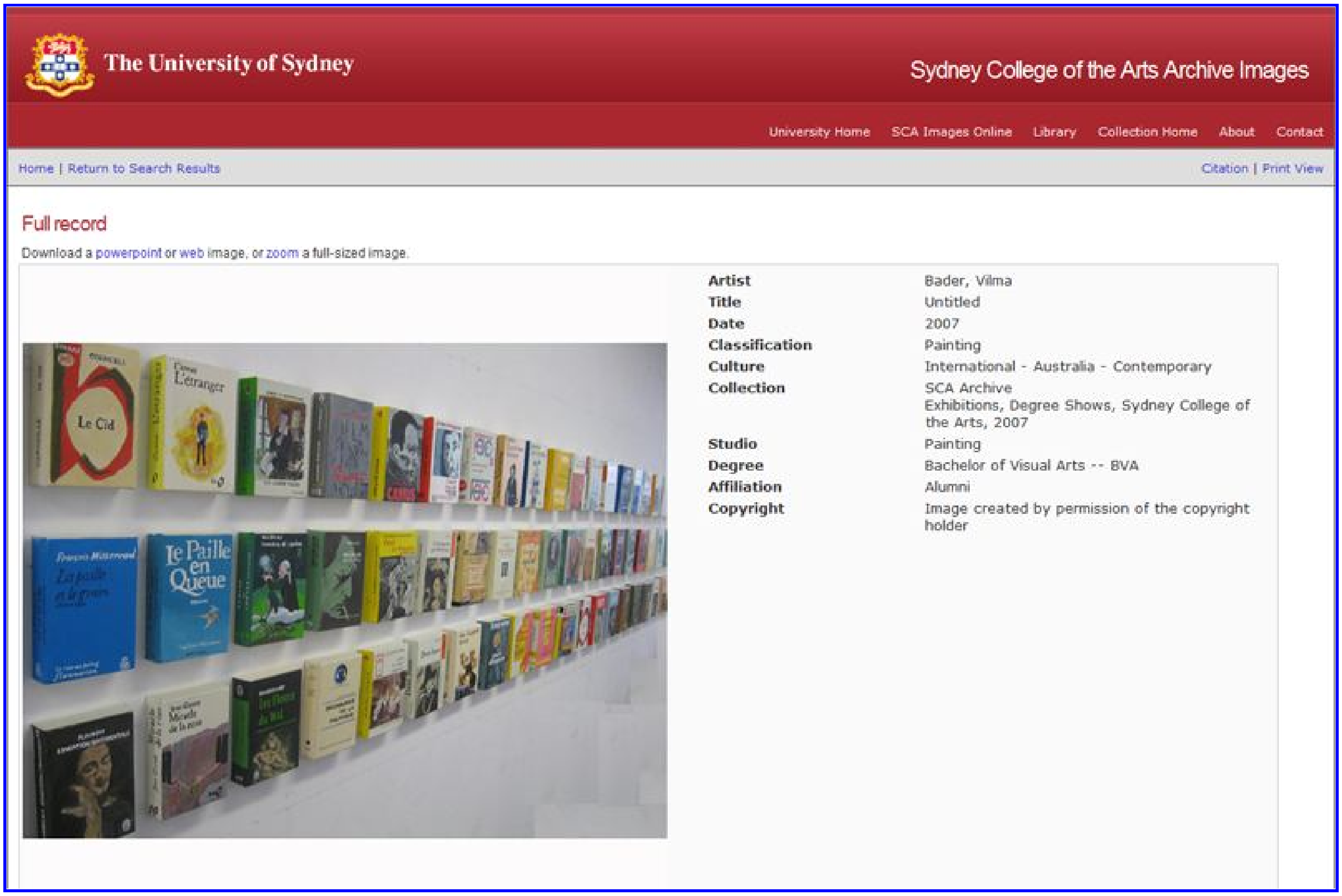
style/dynaXML/docFormatter/sca/scaDocFormatter.xsl

- Screenshot: Full record with zoom hyperlink
- Configuring zoom hyperlink
- Zoom template
- Screenshot: Full record with zoomified image

Layout for a page displaying a zoomable image is similar to that for a print view, with the addition of header and footer information. (For information about including branding and header/footer content, see styling the site).

Screenshot: Full record with zoom hyperlink

The option to zoom a full-sized image is included as a hyperlink on the full record display page (blue hyperlink just under the *Full record* heading).



The University of Sydney
Sydney College of the Arts Archive Images
University Home SCA Images Online Library Collection Home About Contact
Home | Return to Search Results
Citation | Print View

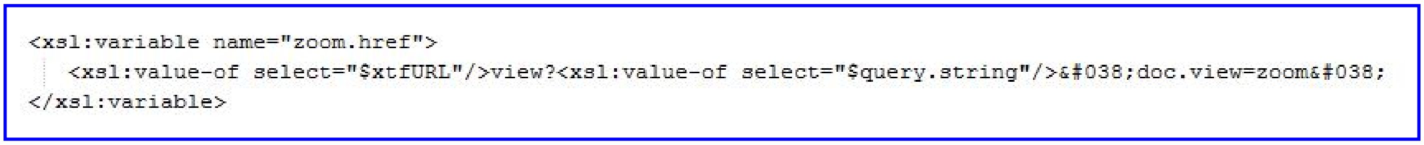
Full record

Download a powerpoint or web image, or zoom a full-sized image.

Artist	Bader, Vilma
Title	Untitled
Date	2007
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Painting
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Configuring zoom hyperlink

The hyperlink references a variable containing information identifying the record currently being displayed, and the desired view option.



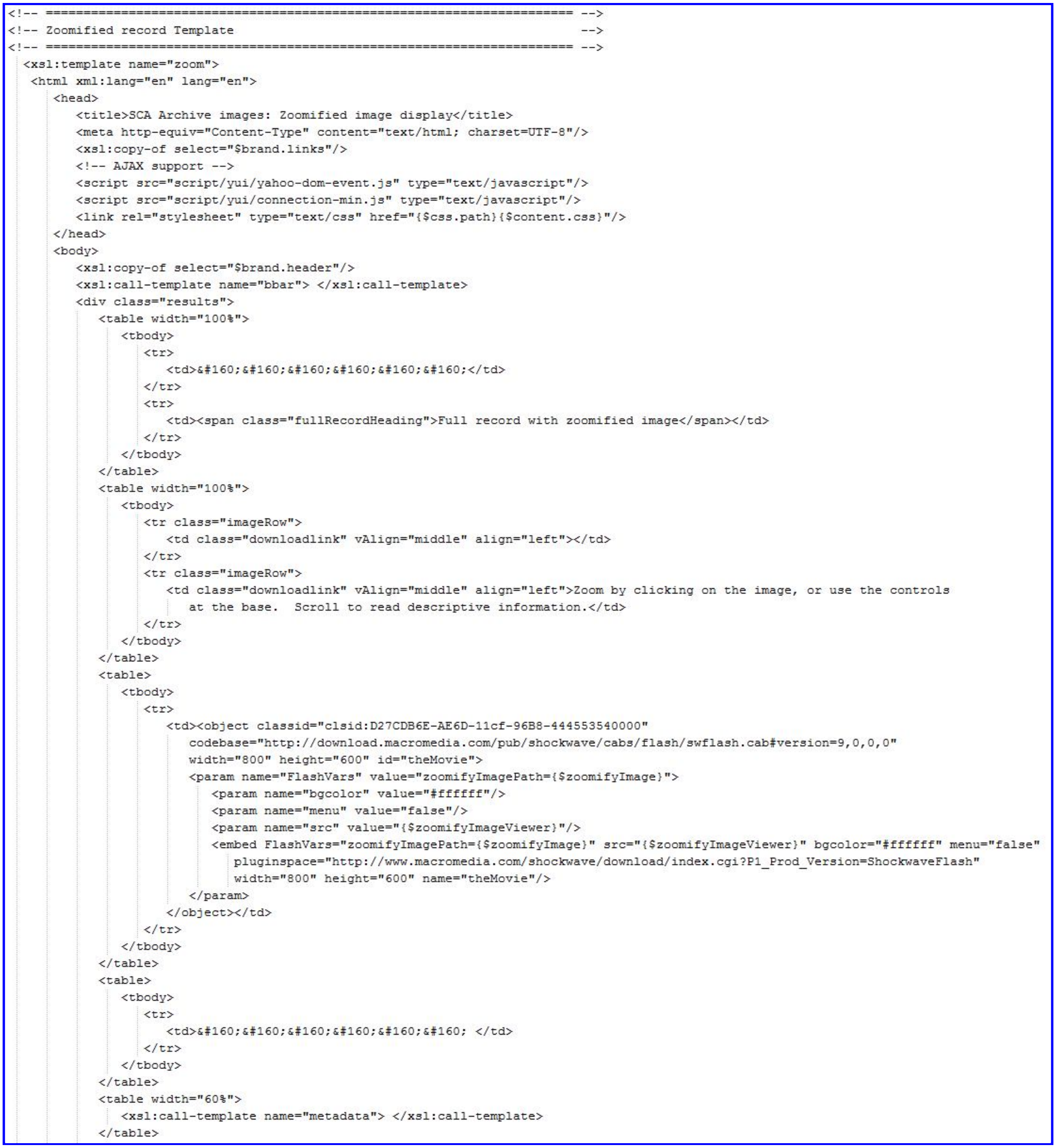
```
<xsl:variable name="zoom.href">
   <xsl:value-of select="$xtfURL"/>view?<xsl:value-of select="$query.string"/>&#038;doc.view=zoom&#038;
</xsl:variable>
```

<a href=”{$zoom.href}”>zoom</a><xsl:text> a full-sized image.</xsl:text>

Zoom template

Clicking on the link supplies this information to scaDocFormatter.xsl, which calls the specified document view template. The zoom template relates the current record with the path to the zoomified image, and holds this information in a variable called *zoomifyImage*. (XTF knows to use scaDocFormatter.xsl because of an association previously defined within docSelector.xsl).

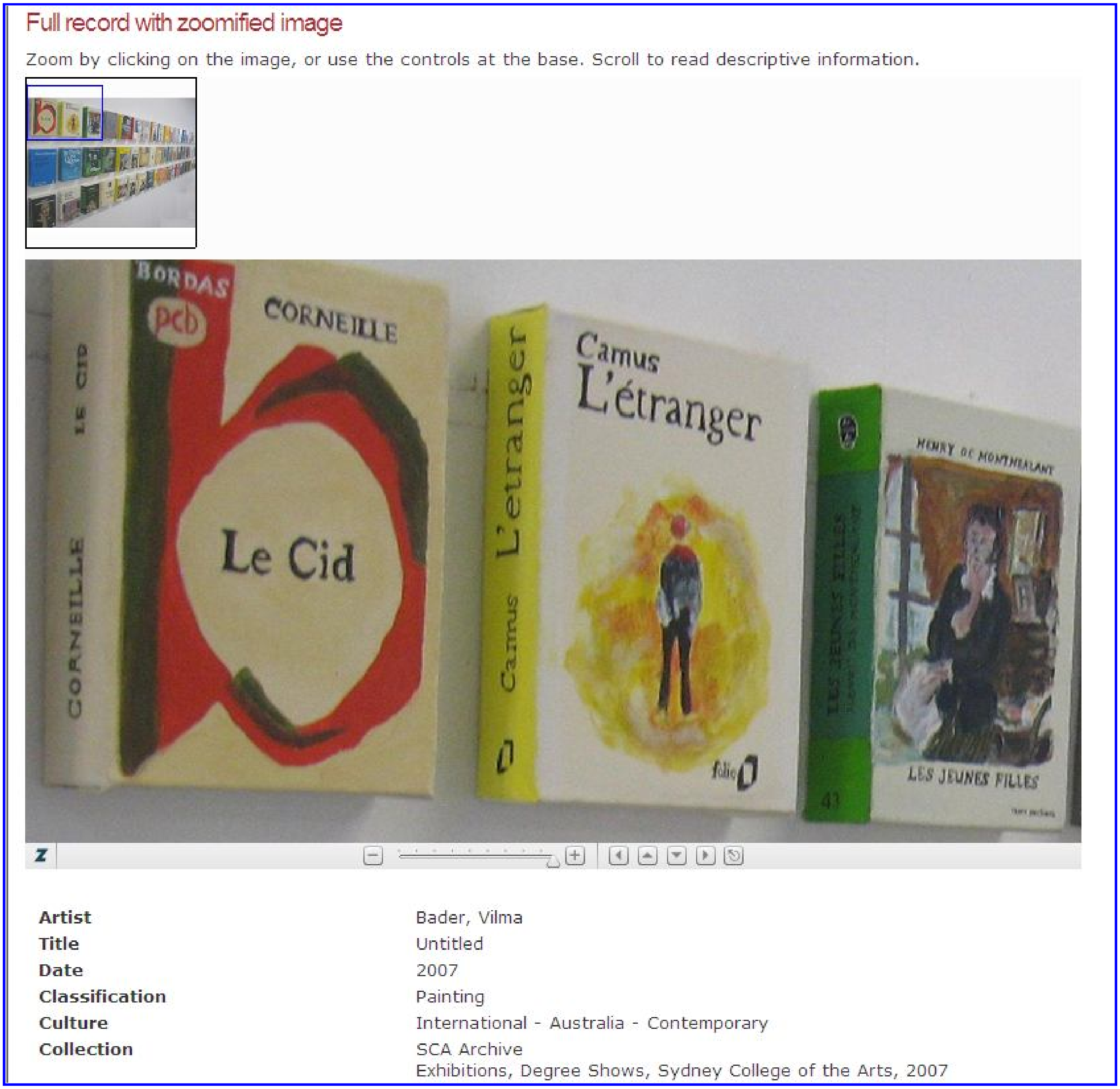
The parameter *zoomifyImage* (defined earlier) is referenced by the section of the template responsible for configuring and displaying the flash player used to render and navigate the zoomable image.



```
<!-- ====================================================================== -->
<!-- Zoomified record Template                                              -->
<!-- ====================================================================== -->
  <xsl:template name="zoom">
   <html xml:lang="en" lang="en">
      <head>
         <title>SCA Archive images: Zoomified image display</title>
         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
         <xsl:copy-of select="$brand.links"/>
         <!-- AJAX support -->
         <script src="script/yui/yahoo-dom-event.js" type="text/javascript"/>
         <script src="script/yui/connection-min.js" type="text/javascript"/>
         <link rel="stylesheet" type="text/css" href="{$css.path}{$content.css}"/>
      </head>
      <body>
         <xsl:copy-of select="$brand.header"/>
         <xsl:call-template name="bbar"> </xsl:call-template>
         <div class="results">
            <table width="100%">
               <tbody>
                  <tr>
                     <td>&#160;&#160;&#160;&#160;&#160;&#160;</td>
                  </tr>
                  <tr>
                     <td><span class="fullRecordHeading">Full record with zoomified image</span></td>
                  </tr>
               </tbody>
            </table>
            <table width="100%">
               <tbody>
                  <tr class="imageRow">
                     <td class="downloadlink" vAlign="middle" align="left"></td>
                  </tr>
                  <tr class="imageRow">
                     <td class="downloadlink" vAlign="middle" align="left">Zoom by clicking on the image, or use the controls
                        at the base.  Scroll to read descriptive information.</td>
                  </tr>
               </tbody>
            </table>
            <table>
               <tbody>
                  <tr>
                     <td><object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
                        codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=9,0,0,0"
                        width="800" height="600" id="theMovie">
                        <param name="FlashVars" value="zoomifyImagePath={$zoomifyImage}">
                           <param name="bgcolor" value="#ffffff"/>
                           <param name="menu" value="false"/>
                           <param name="src" value="{$zoomifyImageViewer}"/>
                           <embed FlashVars="zoomifyImagePath={$zoomifyImage}" src="{$zoomifyImageViewer}" bgcolor="#ffffff" menu="false"
                              pluginspace="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod_Version=ShockwaveFlash"
                              width="800" height="600" name="theMovie"/>
                        </param>
                     </object></td>
                  </tr>
               </tbody>
            </table>
            <table>
               <tbody>
                  <tr>
                     <td>&#160;&#160;&#160;&#160;&#160;&#160; </td>
                  </tr>
               </tbody>
            </table>
            <table width="60%">
               <xsl:call-template name="metadata"> </xsl:call-template>
            </table>
```

The zoom template acts as a dynamic and flexible option for presenting zoomable images wrapped in associated metadata. A customisable (though static) html presentation for zoomable images can alternately be generated through Photoshop.

Screenshot: Full record with zoomified image



Full record with zoomified image

Zoom by clicking on the image, or use the controls at the base. Scroll to read descriptive information.

Artist	Bader, Vilma
Title	Untitled
Date	2007
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007

Next… Video playback

Video playback

style/dynaXML/docFormatter/sca/scaDocFormatter.xsl

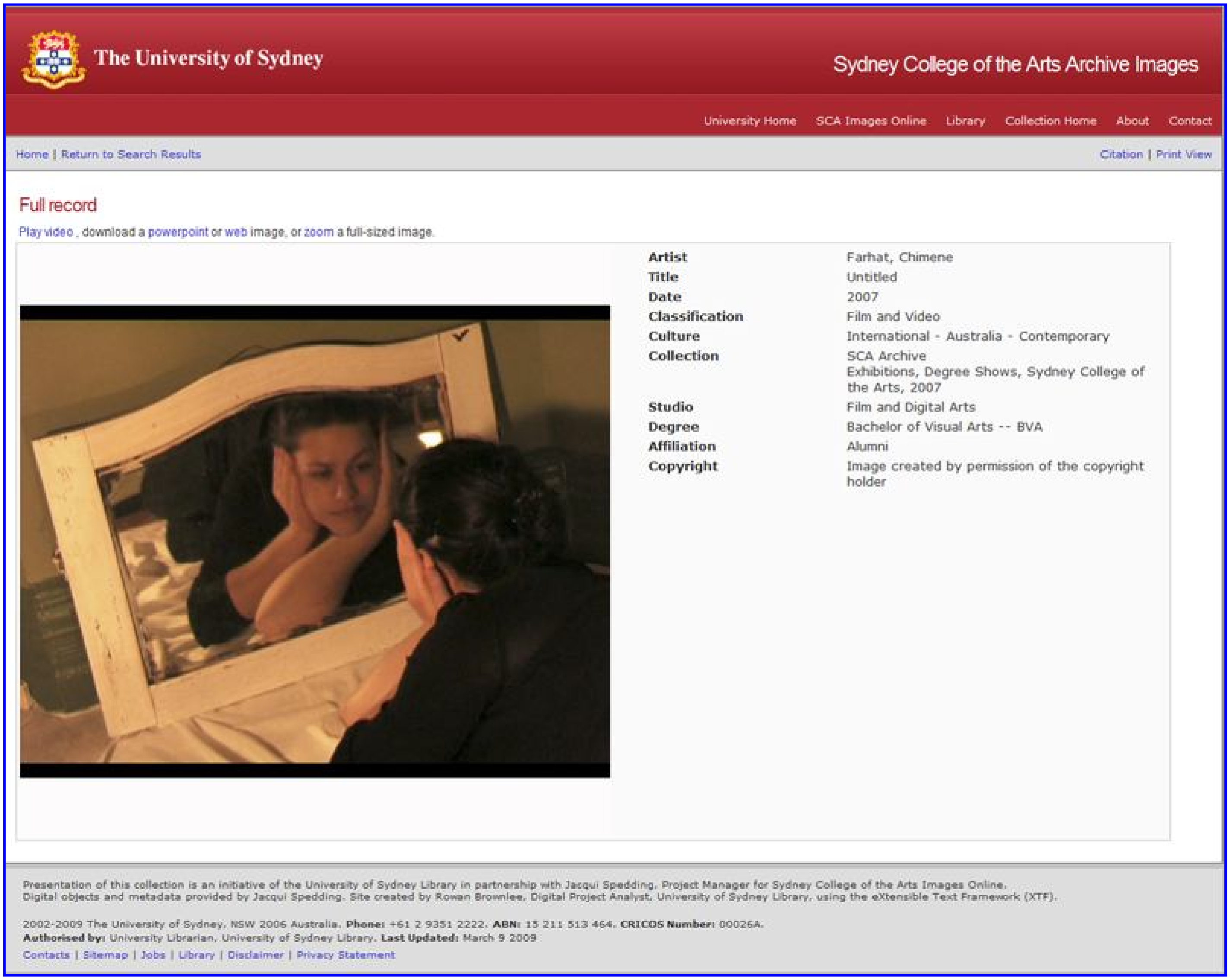
- Testing whether a metadata record describes a video
- Screenshot: Full record with video hyperlink
- Screenshot: Full record without video hyperlink
- Configuring video hyperlink
- Video template
- Screenshot: Video playback

Layout for a page including a video is similar to that for a zoomable image view, with both presenting content using Flash.

Testing whether a metadata record describes a video

Although every record is associated with a common set of images (including thumb and zoom), only a small number describe video files. Metadata records contain information about available format types (such as digital image and video), and I use the occurrence of this information to control hyperlink display. If a record is accompanied by a video, a playback hyperlink displays on the full record display page (blue hyperlink just under the *Full record* heading).

Screenshot: Full record with video hyperlink



The University of Sydney

Sydney College of the Arts Archive Images

Full record

Play video , download a powerpoint or web image, or zoom a full-sized image.

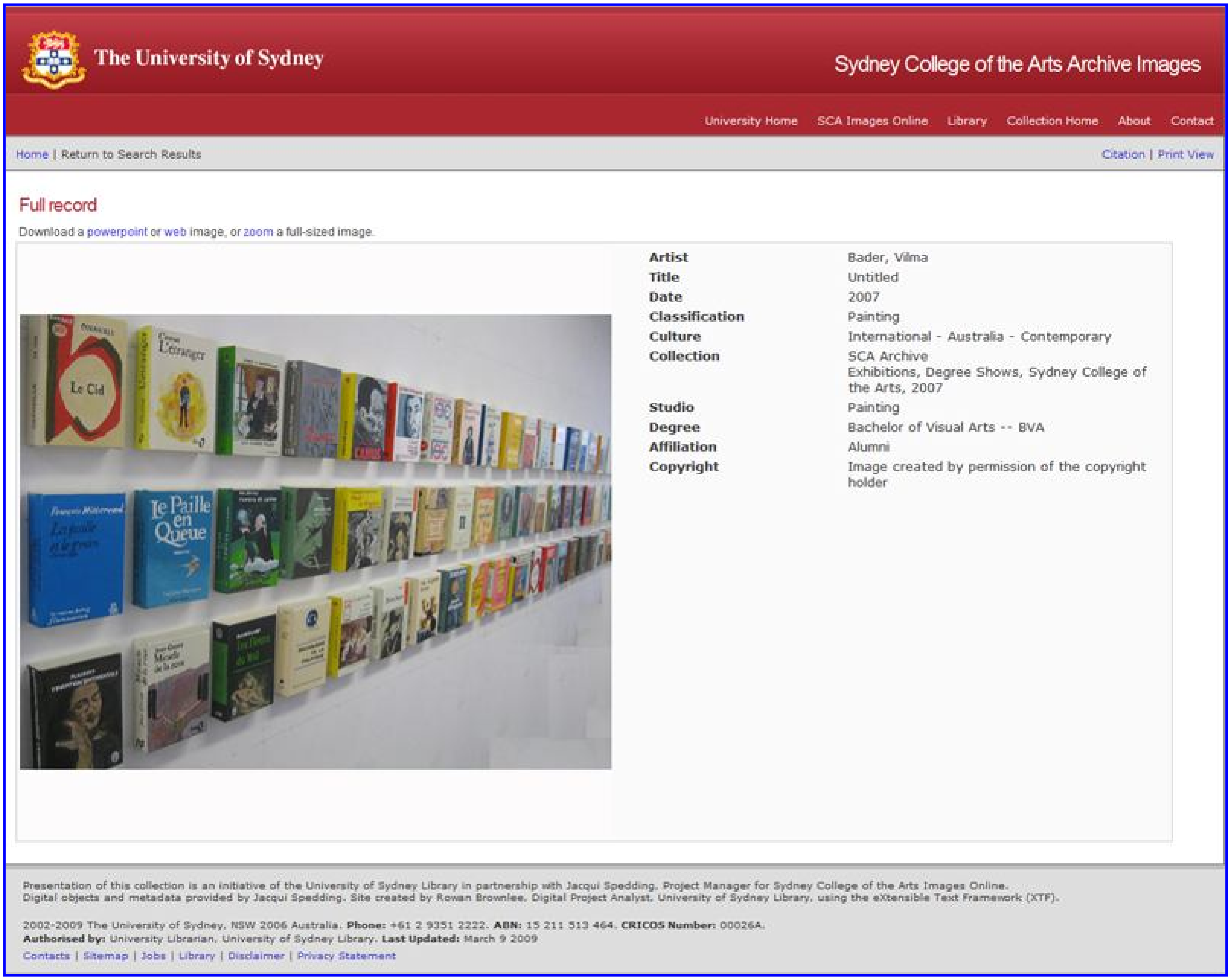
Artist	Farhat, Chimene
Title	Untitled
Date	2007
Classification	Film and Video
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Film and Digital Arts
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** March 9 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Screenshot: Full record without video hyperlink



Full record

Download a powerpoint or web image, or zoom a full-sized image.

Artist	Bader, Vilma
Title	Untitled
Date	2007
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Painting
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

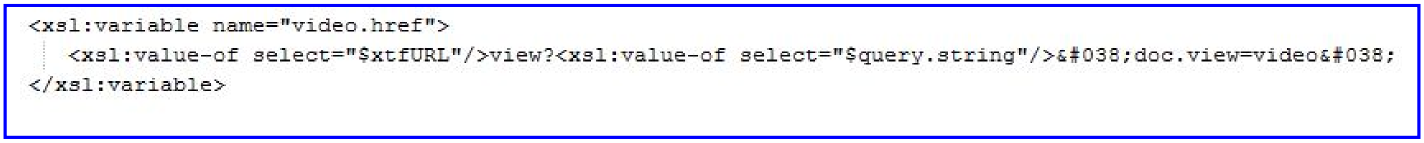
Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** March 9 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Configuring video hyperlink

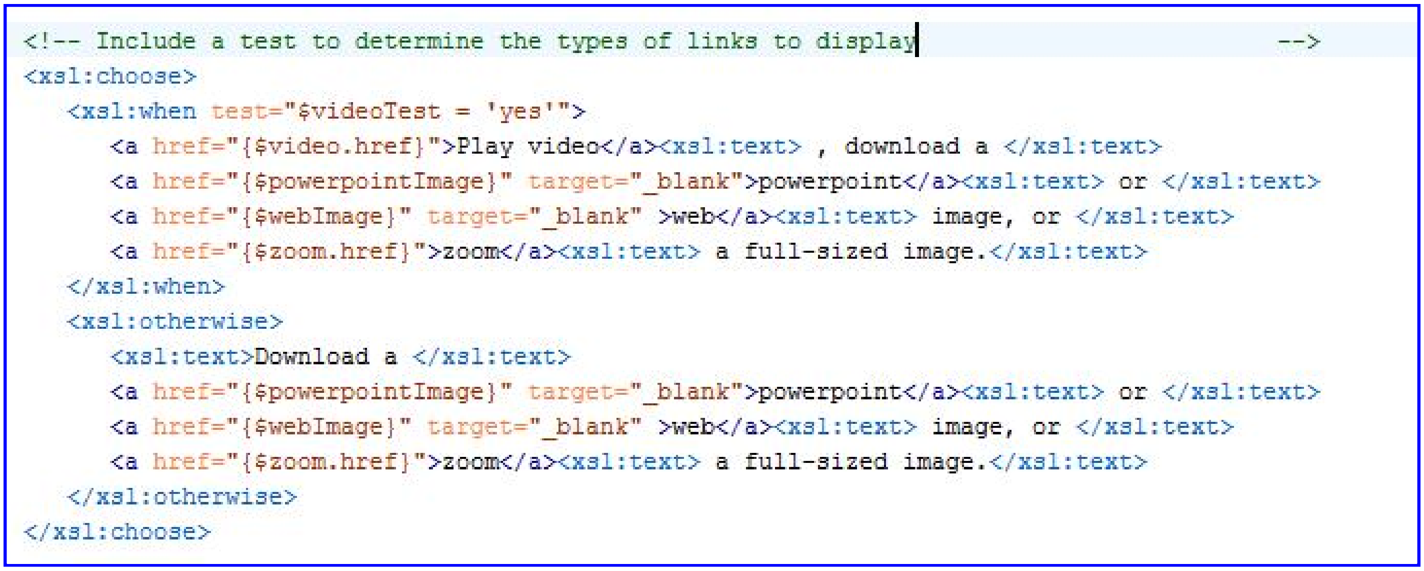
Within the full record template, a hyperlink references a variable containing information identifying the record currently being displayed, and the desired view option.

```
<xsl:variable name="videoTest" select="/sca_record/video_available"></xsl:variable>
```



```
<xsl:variable name="video.href">
    <xsl:value-of select="$xtfURL"/>view?<xsl:value-of select="$query.string"/>&#038;doc.view=video&#038;
</xsl:variable>
```

The following instructions are similar to those previously used to generate hyperlinks to enable access to various forms of an image (such as zoom or powerpoint). The only difference in this case is the addition of a conditional statement to ensure that a link for a video playback page will only display if the metadata indicates availability of a video file.

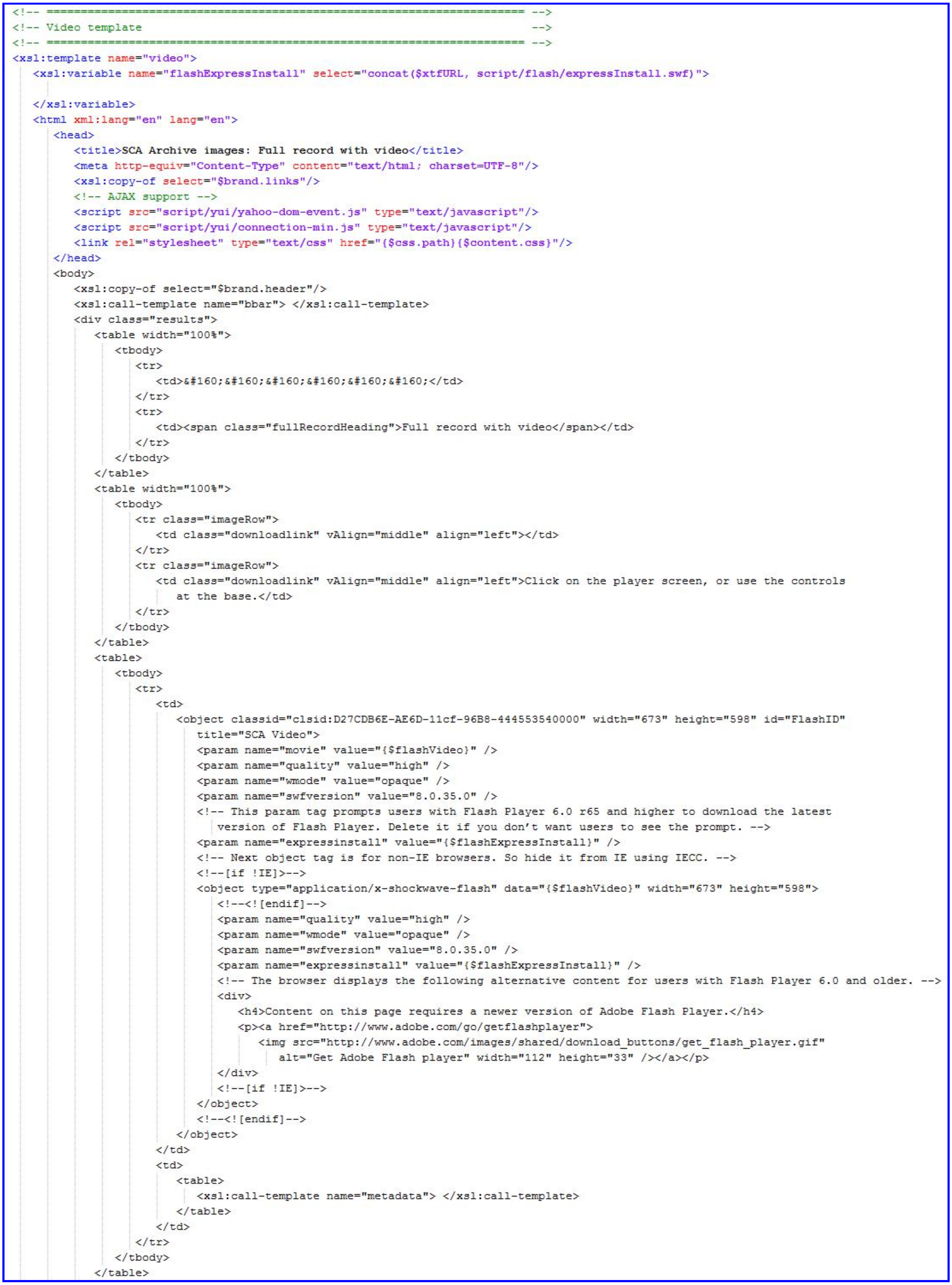


```
<!-- Include a test to determine the types of links to display                         -->
<xsl:choose>
   <xsl:when test="$videoTest = 'yes'">
      <a href="{$video.href}">Play video</a><xsl:text> , download a </xsl:text>
      <a href="{$powerpointImage}" target="_blank">powerpoint</a><xsl:text> or </xsl:text>
      <a href="{$webImage}" target="_blank" >web</a><xsl:text> image, or </xsl:text>
      <a href="{$zoom.href}">zoom</a><xsl:text> a full-sized image.</xsl:text>
   </xsl:when>
   <xsl:otherwise>
      <xsl:text>Download a </xsl:text>
      <a href="{$powerpointImage}" target="_blank">powerpoint</a><xsl:text> or </xsl:text>
      <a href="{$webImage}" target="_blank" >web</a><xsl:text> image, or </xsl:text>
      <a href="{$zoom.href}">zoom</a><xsl:text> a full-sized image.</xsl:text>
   </xsl:otherwise>
</xsl:choose>
```

Video template

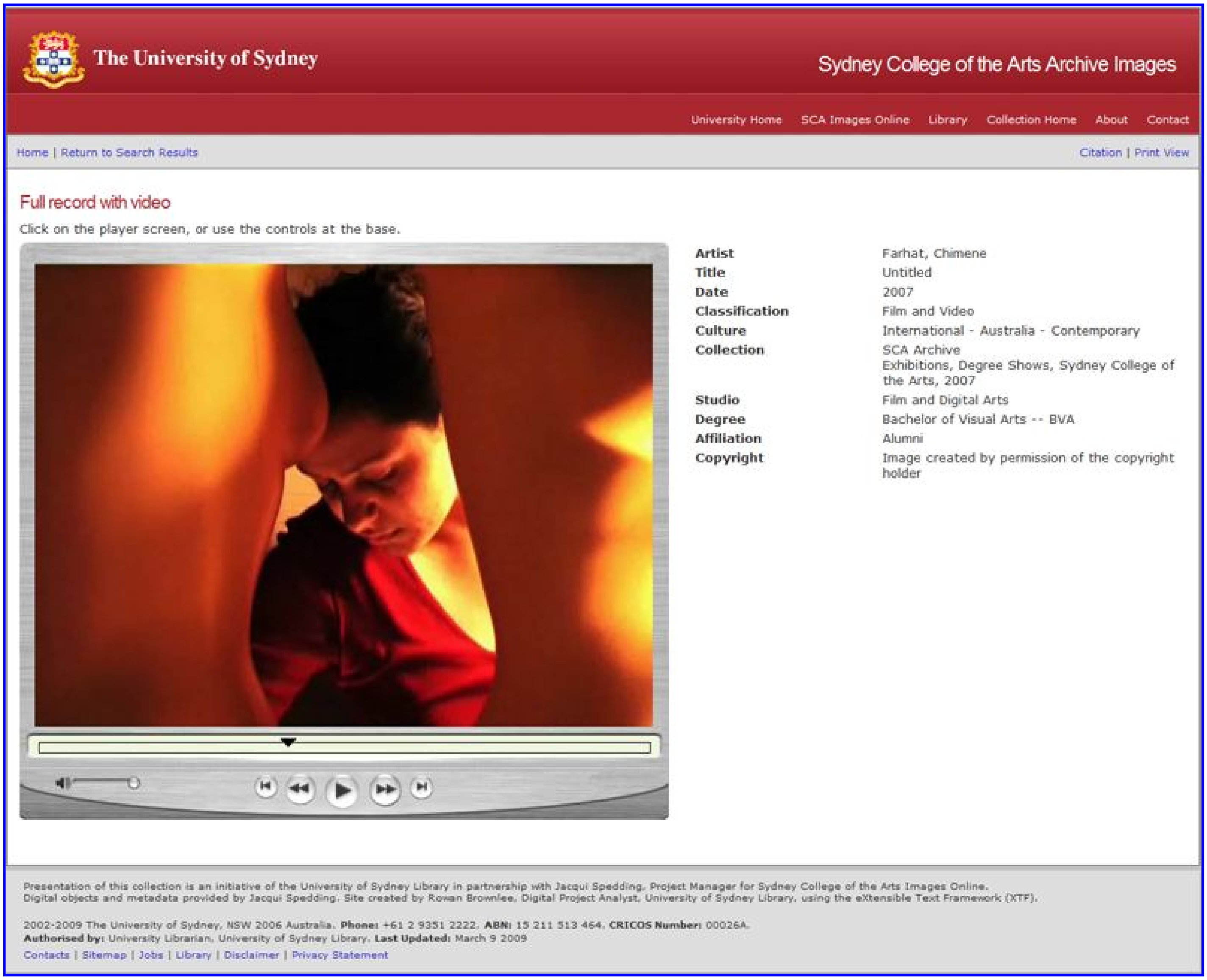
When the video hyperlink is clicked, *scaDocFormatter.xsl* calls the *video* document view template. The video template relates the current record with the path to the video, and holds this information in a parameter called *flashVideo*. (XTF knows to use scaDocFormatter.xsl because of an association previously defined within docSelector.xsl).

The parameter *flashVideo* (defined earlier) is referenced by the section of the template responsible for configuring and displaying the flash video player.



```
<!-- ====================================================================== -->
<!-- Video template                                                         -->
<!-- ====================================================================== -->
<xsl:template name="video">
   <xsl:variable name="flashExpressInstall" select="concat($xtfURL, script/flash/expressInstall.swf)">

   </xsl:variable>
   <html xml:lang="en" lang="en">
      <head>
         <title>SCA Archive images: Full record with video</title>
         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
         <xsl:copy-of select="$brand.links"/>
         <!-- AJAX support -->
         <script src="script/yui/yahoo-dom-event.js" type="text/javascript"/>
         <script src="script/yui/connection-min.js" type="text/javascript"/>
         <link rel="stylesheet" type="text/css" href="{$css.path}{$content.css}"/>
      </head>
      <body>
         <xsl:copy-of select="$brand.header"/>
         <xsl:call-template name="bbar"> </xsl:call-template>
         <div class="results">
            <table width="100%">
               <tbody>
                  <tr>
                     <td>&#160;&#160;&#160;&#160;&#160;&#160;</td>
                  </tr>
                  <tr>
                     <td><span class="fullRecordHeading">Full record with video</span></td>
                  </tr>
               </tbody>
            </table>
            <table width="100%">
               <tbody>
                  <tr class="imageRow">
                     <td class="downloadlink" vAlign="middle" align="left"></td>
                  </tr>
                  <tr class="imageRow">
                     <td class="downloadlink" vAlign="middle" align="left">Click on the player screen, or use the controls
                        at the base.</td>
                  </tr>
               </tbody>
            </table>
            <table>
               <tbody>
                  <tr>
                     <td>
                        <object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" width="673" height="598" id="FlashID"
                           title="SCA Video">
                           <param name="movie" value="{$flashVideo}" />
                           <param name="quality" value="high" />
                           <param name="wmode" value="opaque" />
                           <param name="swfversion" value="8.0.35.0" />
                           <!-- This param tag prompts users with Flash Player 6.0 r65 and higher to download the latest
                              version of Flash Player. Delete it if you don't want users to see the prompt. -->
                           <param name="expressinstall" value="{$flashExpressInstall}" />
                           <!-- Next object tag is for non-IE browsers. So hide it from IE using IECC. -->
                           <!--[if !IE]>-->
                           <object type="application/x-shockwave-flash" data="{$flashVideo}" width="673" height="598">
                              <!--<![endif]-->
                              <param name="quality" value="high" />
                              <param name="wmode" value="opaque" />
                              <param name="swfversion" value="8.0.35.0" />
                              <param name="expressinstall" value="{$flashExpressInstall}" />
                              <!-- The browser displays the following alternative content for users with Flash Player 6.0 and older. -->
                              <div>
                                 <h4>Content on this page requires a newer version of Adobe Flash Player.</h4>
                                 <p><a href="http://www.adobe.com/go/getflashplayer">
                                    <img src="http://www.adobe.com/images/shared/download_buttons/get_flash_player.gif"
                                       alt="Get Adobe Flash player" width="112" height="33" /></a></p>
                              </div>
                              <!--[if !IE]>-->
                           </object>
                           <!--<![endif]-->
                        </object>
                     </td>
                     <td>
                        <table>
                           <xsl:call-template name="metadata"> </xsl:call-template>
                        </table>
                     </td>
                  </tr>
               </tbody>
            </table>
```

Screenshot: Video playback



Full record with video

Click on the player screen, or use the controls at the base.

Artist	Farhat, Chimene
Title	Untitled
Date	2007
Classification	Film and Video
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Film and Digital Arts
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** March 9 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Next… Video playback part two: Offering video for high, medium and low-speed connections

Video playback part two: Offering video for high-, medium- and low-speed connections

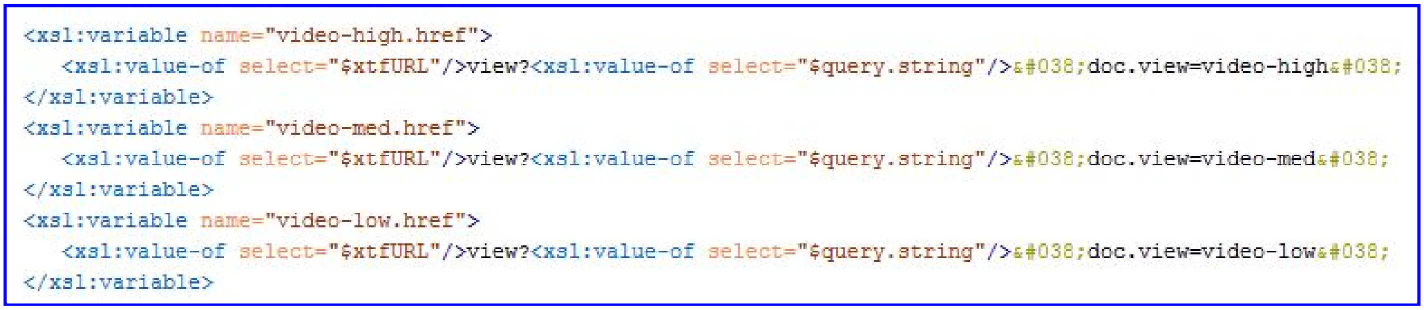
style/dynaXML/docFormatter/sca/scaDocFormatter.xsl

- Configuring video hyperlinks
- Video display options within the root template
- Video template
- Screenshot: Full record with video hyperlinks

I have Flash video suited for 1000k, 512k and 56k connections (via Flix). I use a named template with parameters to handle requests for the different video resolutions. By using parameters, I avoid the need to repeat almost identical templates, and this assists maintenance.

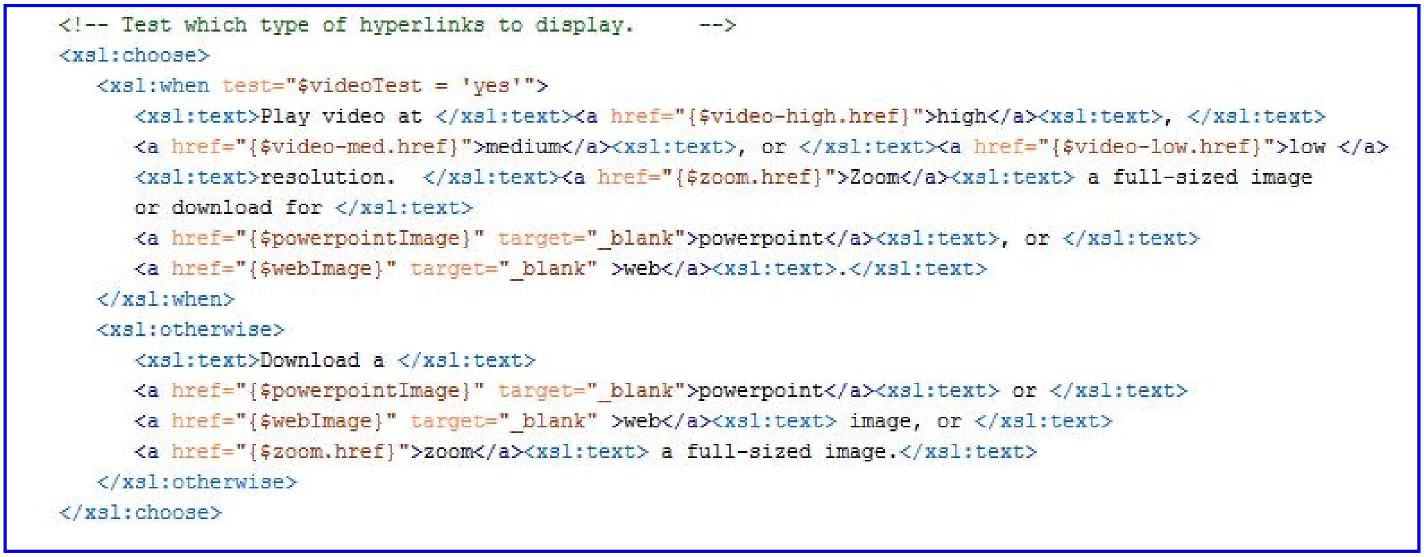
Configuring video hyperlinks

This is similar to the setup in the previous section, though now I have several variables.



```
<xsl:variable name="video-high.href">
   <xsl:value-of select="$xtfURL"/>view?<xsl:value-of select="$query.string"/>&#038;doc.view=video-high&#038;
</xsl:variable>
<xsl:variable name="video-med.href">
   <xsl:value-of select="$xtfURL"/>view?<xsl:value-of select="$query.string"/>&#038;doc.view=video-med&#038;
</xsl:variable>
<xsl:variable name="video-low.href">
   <xsl:value-of select="$xtfURL"/>view?<xsl:value-of select="$query.string"/>&#038;doc.view=video-low&#038;
</xsl:variable>
```

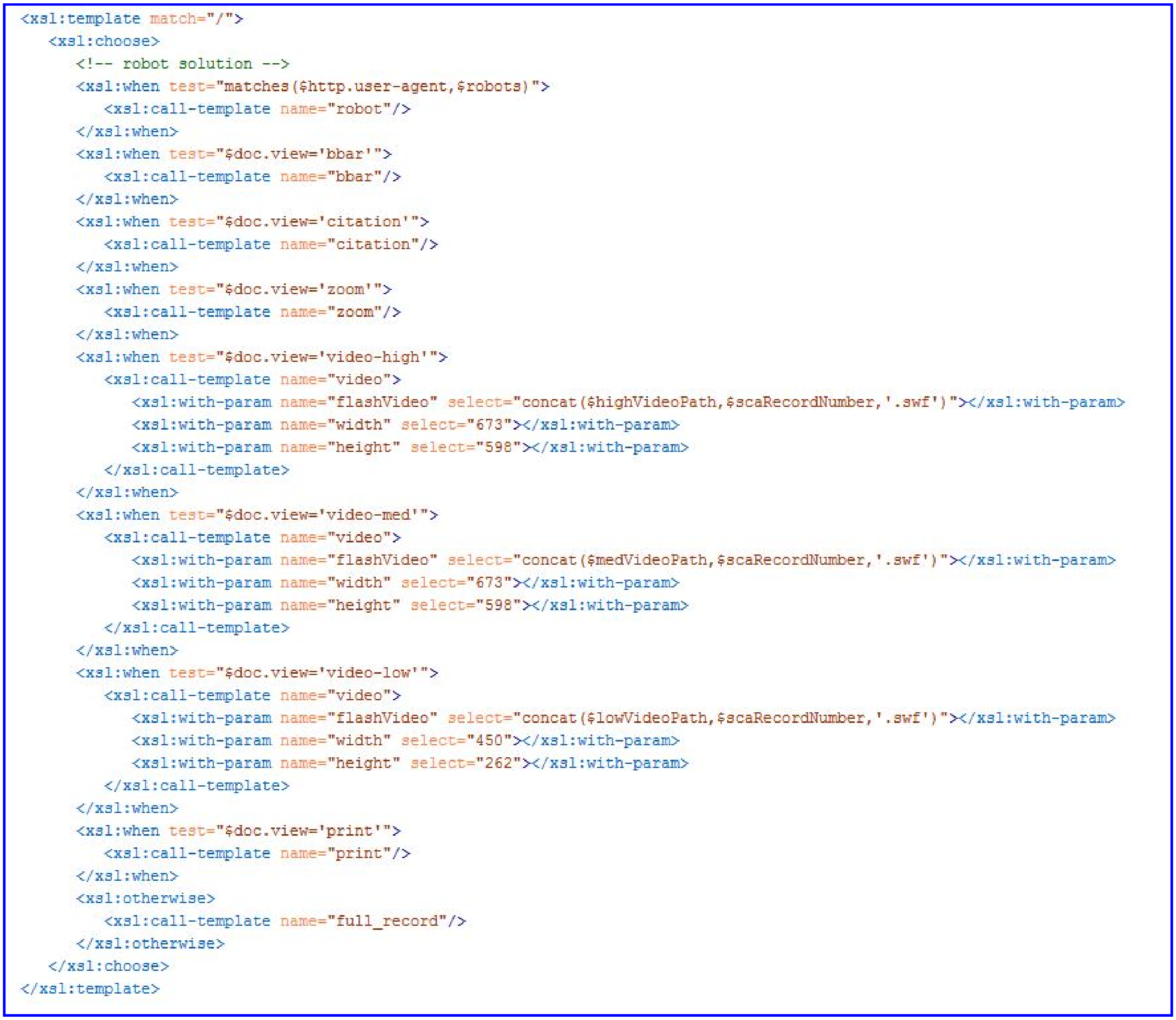
Slight change to the test used to determine which hyperlinks will display.



```
   <!-- Test which type of hyperlinks to display.     -->
   <xsl:choose>
      <xsl:when test="$videoTest = 'yes'">
         <xsl:text>Play video at </xsl:text><a href="{$video-high.href}">high</a><xsl:text>, </xsl:text>
         <a href="{$video-med.href}">medium</a><xsl:text>, or </xsl:text><a href="{$video-low.href}">low </a>
         <xsl:text>resolution.  </xsl:text><a href="{$zoom.href}">Zoom</a><xsl:text> a full-sized image
         or download for </xsl:text>
         <a href="{$powerpointImage}" target="_blank">powerpoint</a><xsl:text>, or </xsl:text>
         <a href="{$webImage}" target="_blank" >web</a><xsl:text>.</xsl:text>
      </xsl:when>
      <xsl:otherwise>
         <xsl:text>Download a </xsl:text>
         <a href="{$powerpointImage}" target="_blank">powerpoint</a><xsl:text> or </xsl:text>
         <a href="{$webImage}" target="_blank" >web</a><xsl:text> image, or </xsl:text>
         <a href="{$zoom.href}">zoom</a><xsl:text> a full-sized image.</xsl:text>
      </xsl:otherwise>
   </xsl:choose>
```

Video display options within the root template

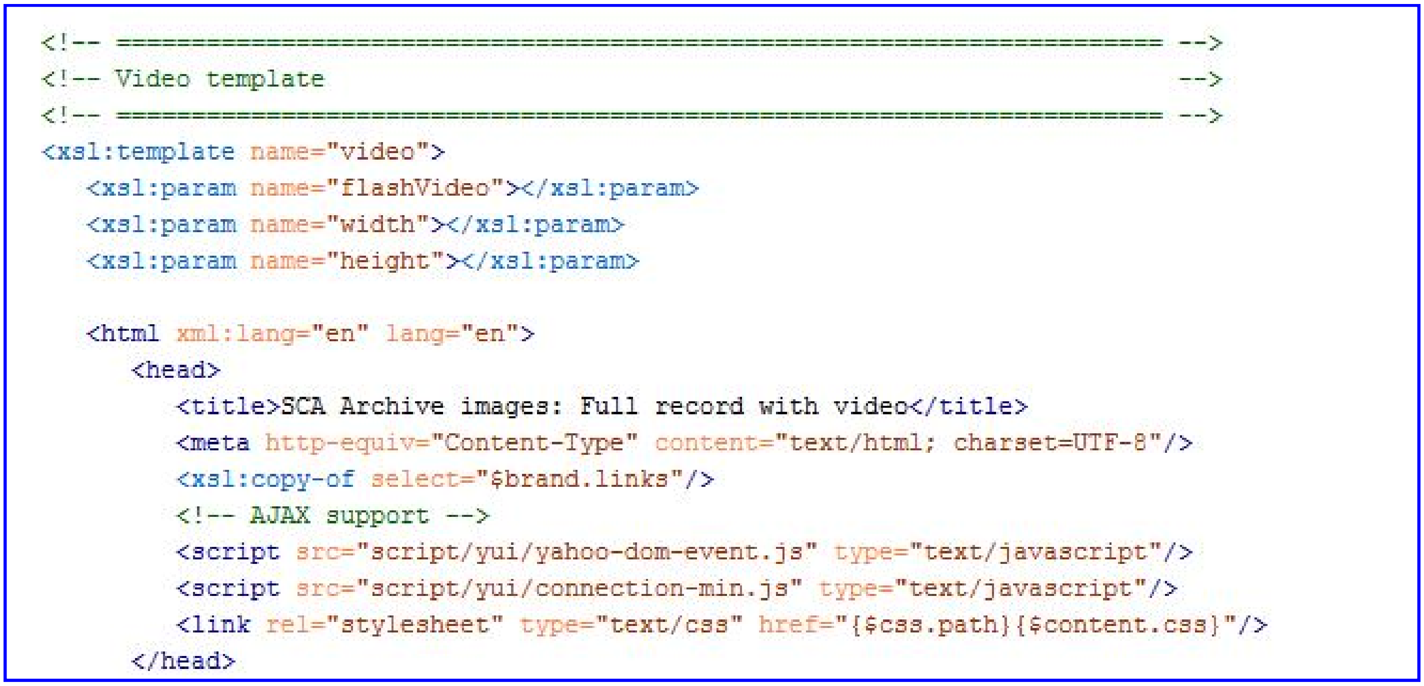
The root template now contains instructions for dealing with requests to view high-, medium- and low-resolution video files.



```
<xsl:template match="/">
   <xsl:choose>
      <!-- robot solution -->
      <xsl:when test="matches($http.user-agent,$robots)">
         <xsl:call-template name="robot"/>
      </xsl:when>
      <xsl:when test="$doc.view='bbar'">
         <xsl:call-template name="bbar"/>
      </xsl:when>
      <xsl:when test="$doc.view='citation'">
         <xsl:call-template name="citation"/>
      </xsl:when>
      <xsl:when test="$doc.view='zoom'">
         <xsl:call-template name="zoom"/>
      </xsl:when>
      <xsl:when test="$doc.view='video-high'">
         <xsl:call-template name="video">
            <xsl:with-param name="flashVideo" select="concat($highVideoPath,$scaRecordNumber,'.swf')"></xsl:with-param>
            <xsl:with-param name="width" select="673"></xsl:with-param>
            <xsl:with-param name="height" select="598"></xsl:with-param>
         </xsl:call-template>
      </xsl:when>
      <xsl:when test="$doc.view='video-med'">
         <xsl:call-template name="video">
            <xsl:with-param name="flashVideo" select="concat($medVideoPath,$scaRecordNumber,'.swf')"></xsl:with-param>
            <xsl:with-param name="width" select="673"></xsl:with-param>
            <xsl:with-param name="height" select="598"></xsl:with-param>
         </xsl:call-template>
      </xsl:when>
      <xsl:when test="$doc.view='video-low'">
         <xsl:call-template name="video">
            <xsl:with-param name="flashVideo" select="concat($lowVideoPath,$scaRecordNumber,'.swf')"></xsl:with-param>
            <xsl:with-param name="width" select="450"></xsl:with-param>
            <xsl:with-param name="height" select="262"></xsl:with-param>
         </xsl:call-template>
      </xsl:when>
      <xsl:when test="$doc.view='print'">
         <xsl:call-template name="print"/>
      </xsl:when>
      <xsl:otherwise>
         <xsl:call-template name="full_record"/>
      </xsl:otherwise>
   </xsl:choose>
</xsl:template>
```

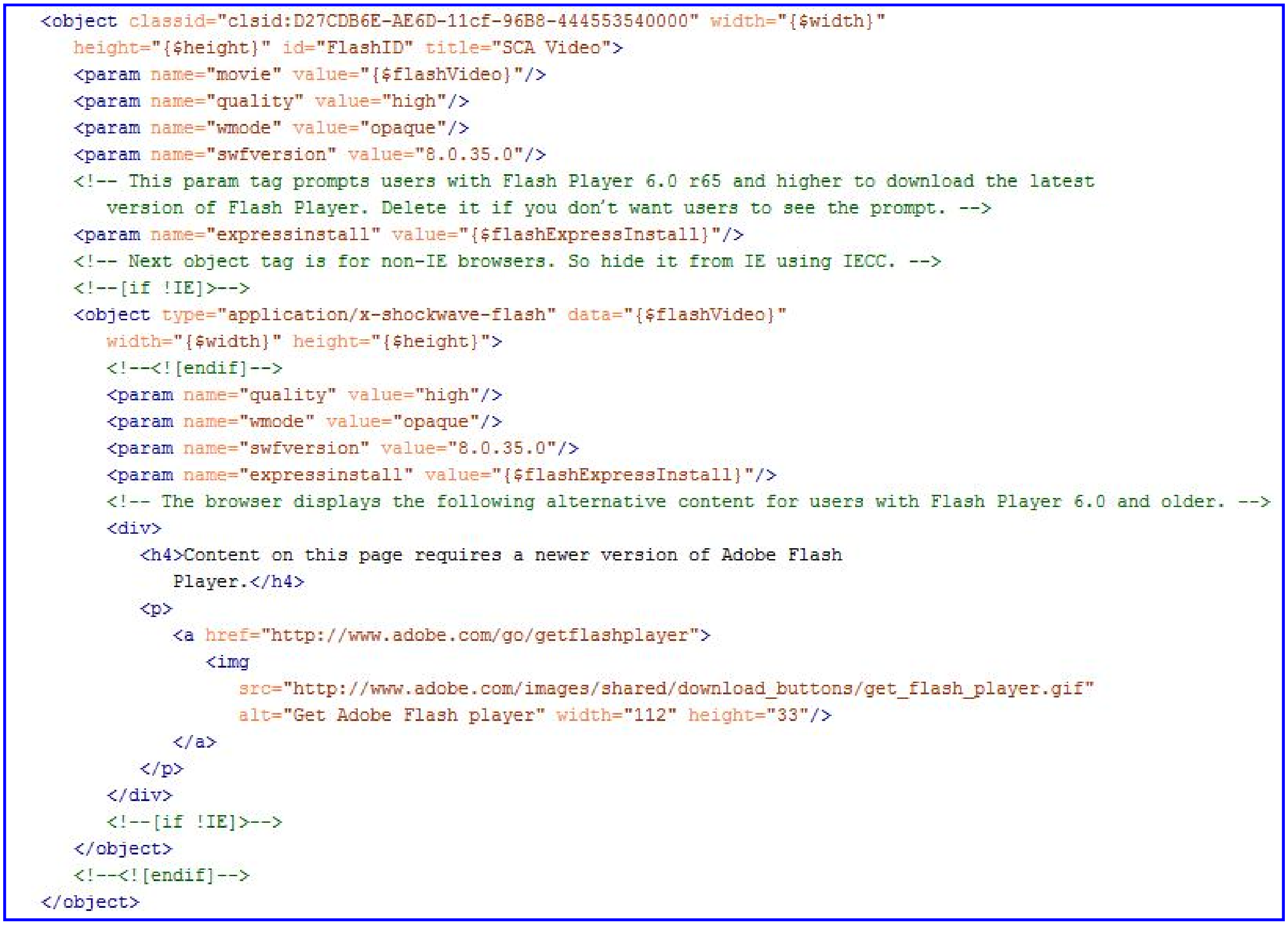
Video template

The video template includes parameters specifying the path to the video file, and player dimensions. This information is passed by the calling template.



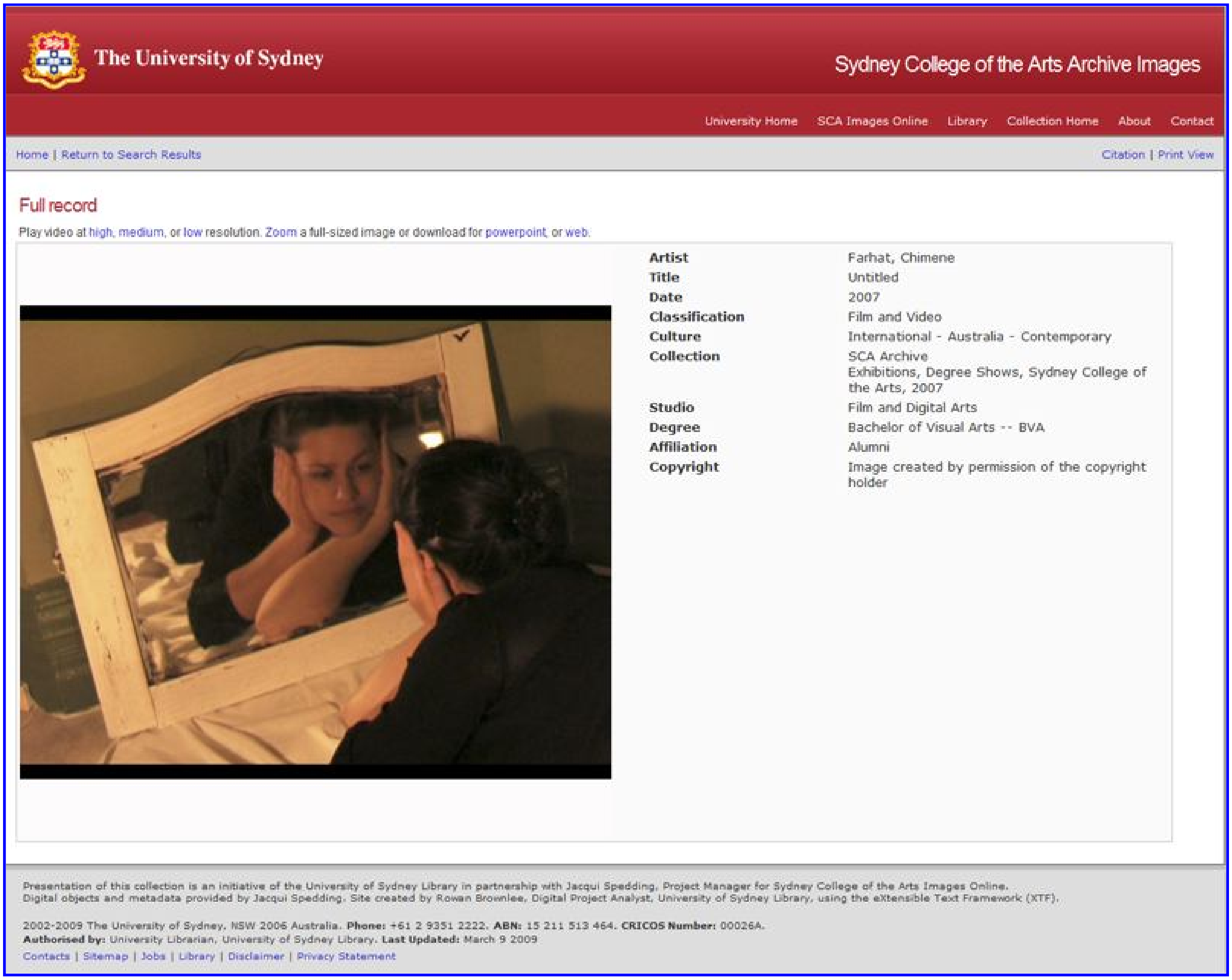
```
<!-- ====================================================================== -->
<!-- Video template                                                         -->
<!-- ====================================================================== -->
<xsl:template name="video">
   <xsl:param name="flashVideo"></xsl:param>
   <xsl:param name="width"></xsl:param>
   <xsl:param name="height"></xsl:param>

   <html xml:lang="en" lang="en">
      <head>
         <title>SCA Archive images: Full record with video</title>
         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
         <xsl:copy-of select="$brand.links"/>
         <!-- AJAX support -->
         <script src="script/yui/yahoo-dom-event.js" type="text/javascript"/>
         <script src="script/yui/connection-min.js" type="text/javascript"/>
         <link rel="stylesheet" type="text/css" href="{$css.path}{$content.css}"/>
      </head>
```

These parameters are used within the section of the video template responsible for setting up the Flash player



```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" width="{$width}"
   height="{$height}" id="FlashID" title="SCA Video">
   <param name="movie" value="{$flashVideo}"/>
   <param name="quality" value="high"/>
   <param name="wmode" value="opaque"/>
   <param name="swfversion" value="8.0.35.0"/>
   <!-- This param tag prompts users with Flash Player 6.0 r65 and higher to download the latest
      version of Flash Player. Delete it if you don’t want users to see the prompt. -->
   <param name="expressinstall" value="{$flashExpressInstall}"/>
   <!-- Next object tag is for non-IE browsers. So hide it from IE using IECC. -->
   <!--[if !IE]>-->
   <object type="application/x-shockwave-flash" data="{$flashVideo}"
      width="{$width}" height="{$height}">
      <!--<![endif]-->
      <param name="quality" value="high"/>
      <param name="wmode" value="opaque"/>
      <param name="swfversion" value="8.0.35.0"/>
      <param name="expressinstall" value="{$flashExpressInstall}"/>
      <!-- The browser displays the following alternative content for users with Flash Player 6.0 and older. -->
      <div>
         <h4>Content on this page requires a newer version of Adobe Flash
            Player.</h4>
         <p>
            <a href="http://www.adobe.com/go/getflashplayer">
               <img
                  src="http://www.adobe.com/images/shared/download_buttons/get_flash_player.gif"
                  alt="Get Adobe Flash player" width="112" height="33"/>
            </a>
         </p>
      </div>
      <!--[if !IE]>-->
   </object>
   <!--<![endif]-->
</object>
```

Screenshot: Full record with video hyperlinks



Full record

Play video at high, medium, or low resolution. Zoom a full-sized image or download for powerpoint, or web.

Artist	Farhat, Chimene
Title	Untitled
Date	2007
Classification	Film and Video
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Film and Digital Arts
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** March 9 2009

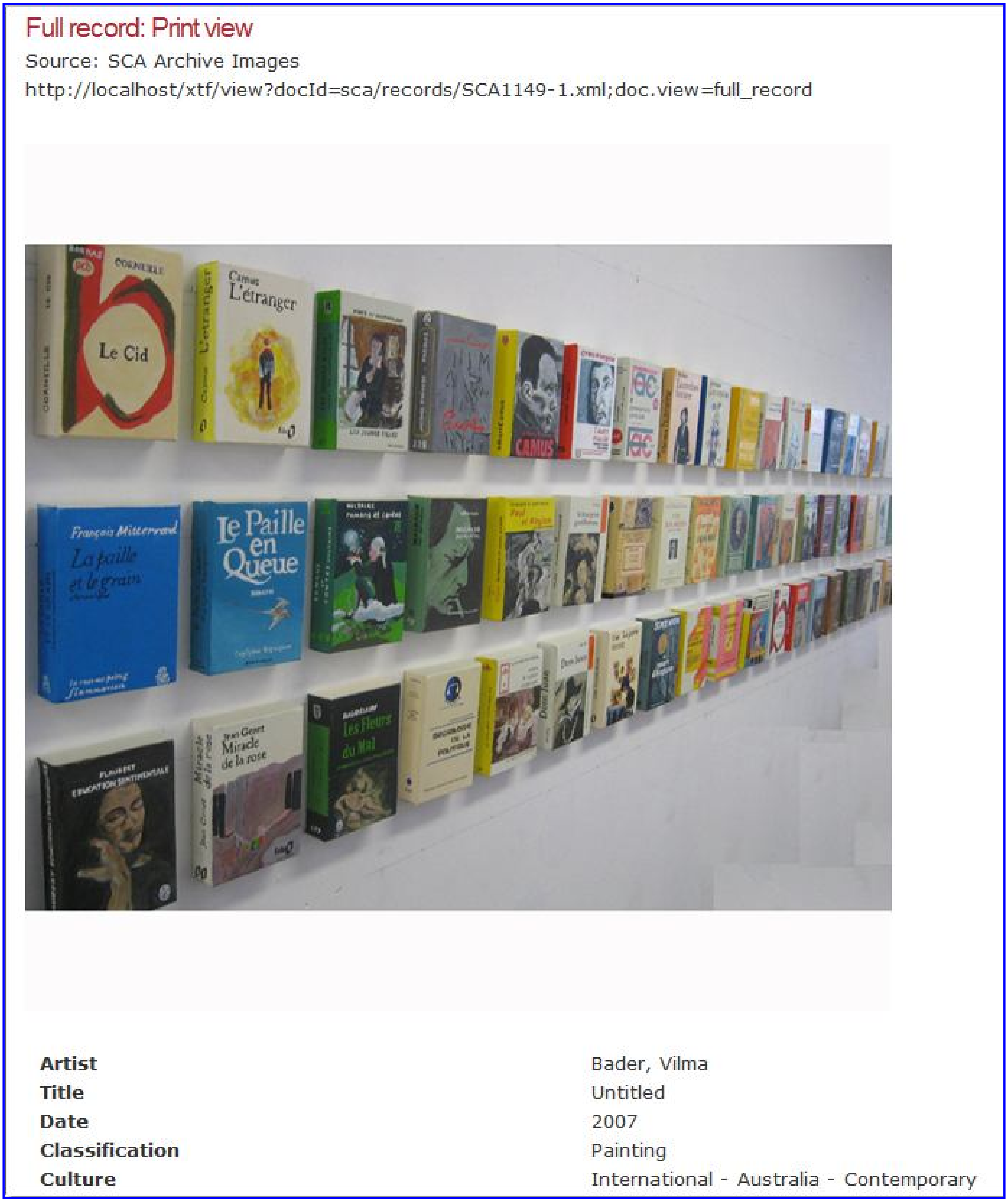
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Next… Displaying records formatted for printing

Displaying records formatted for printing

style/dynaXML/docFormatter/sca/scaDocFormatter.xsl

Apart from removing branding and altering the metadata display to follow below the image, the template responsible for producing a print view is very similar to a full record display. (For information about including branding and header/footer content, see styling the site).



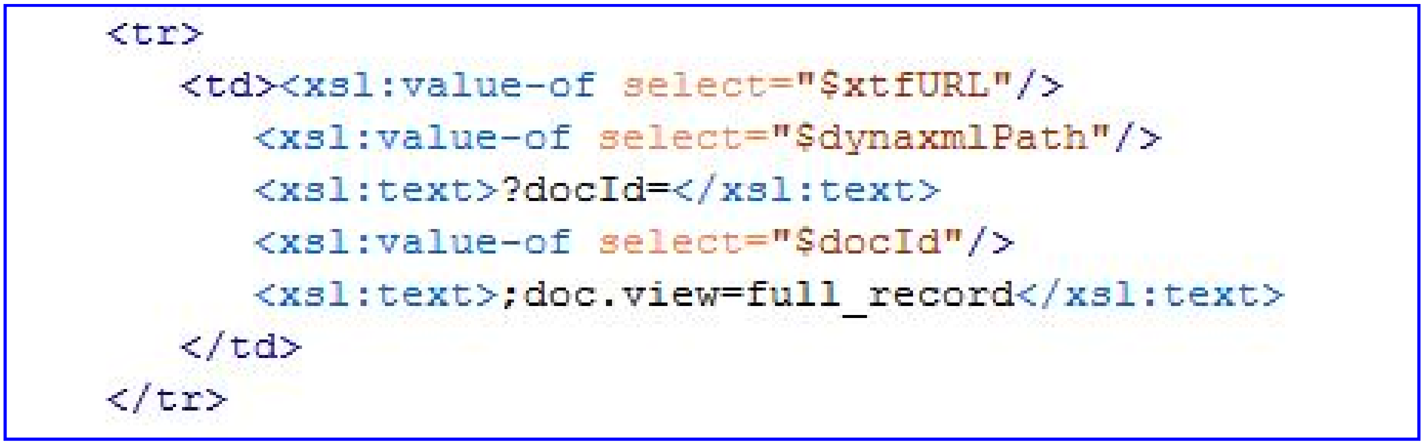
Full record: Print view

Source: SCA Archive Images

http://localhost/xtf/view?docId=sca/records/SCA1149-1.xml;doc.view=full_record

Artist	Bader, Vilma
Title	Untitled
Date	2007
Classification	Painting
Culture	International - Australia - Contemporary

As shown in the screenshot, the print view layout includes the URL for the full record. The URL is produced by adding the following instruction to the print template.



```
<tr>
   <td><xsl:value-of select="$xtfURL"/>
      <xsl:value-of select="$dynaxmlPath"/>
      <xsl:text>?docId=</xsl:text>
      <xsl:value-of select="$docId"/>
      <xsl:text>;doc.view=full_record</xsl:text>
   </td>
</tr>
```

Next… Sorting, browsing and facets

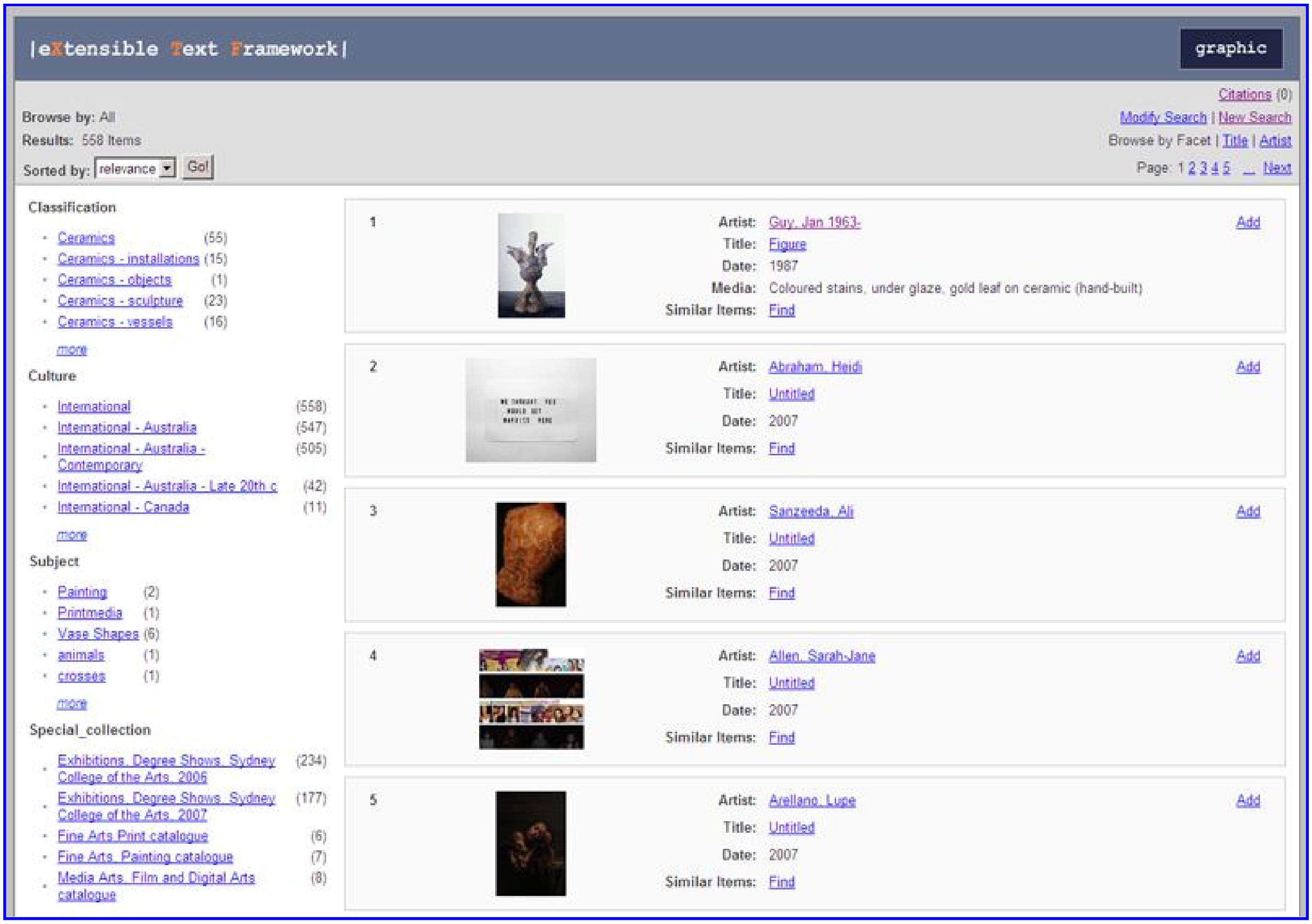
Sorting, browsing and facets

Following on from previous steps it is at this stage possible (if textIndexer has been run), to search metadata and retrieve and display result sets rendered as brief records with accompanying thumbnail images. The thumbnails are clickable through to full records, and from full-record display other types of media are viewable (such as powerpoint and zoomable images and Flash video files).

Although in a previous section I identified metadata for use in facets, other steps are required to produce facets. For the SCA collection, this involves working with preFilterCommon, queryParser, resultFormatterCommon, and resultFormatter.

- preFilterCommon
- queryParser
- resultFormatterCommon
- resultFormatter

After making the changes described in the hyperlinked pages above, a typical search result page displays as follows.



Browse by: All
Results: 558 Items
Sorted by: relevance Go!

Citations (0)
Modify Search | New Search
Browse by Facet | Title | Artist
Page: 1 2 3 4 5 ... Next

Classification

- Ceramics (55)
- Ceramics - installations (15)
- Ceramics - objects (1)
- Ceramics - sculpture (23)
- Ceramics - vessels (16)

more

Culture

- International (558)
- International - Australia (547)
- International - Australia - Contemporary (505)
- International - Australia - Late 20th c (42)
- International - Canada (11)

more

Subject

- Painting (2)
- Printmedia (1)
- Vase Shapes (6)
- animals (1)
- crosses (1)

more

Special_collection

- Exhibitions, Degree Shows, Sydney College of the Arts, 2006 (234)
- Exhibitions, Degree Shows, Sydney College of the Arts, 2007 (177)
- Fine Arts Print catalogue (6)
- Fine Arts, Painting catalogue (7)
- Media Arts, Film and Digital Arts catalogue (8)

1
Artist: Guy, Jan 1963-
Title: Figure
Date: 1987
Media: Coloured stains, under glaze, gold leaf on ceramic (hand-built)
Similar Items: Find
Add

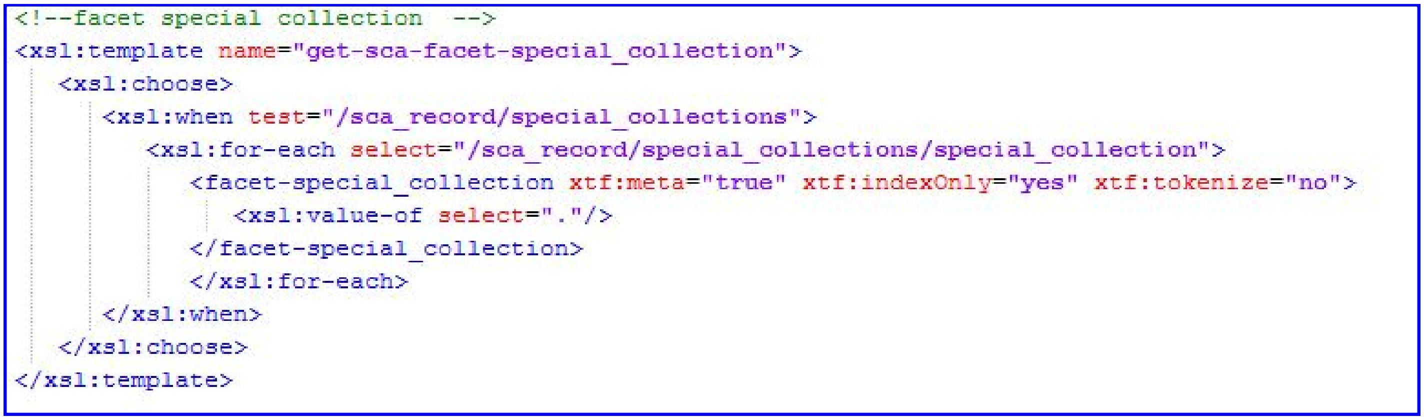
2
Artist: Abraham, Heidi
Title: Untitled
Date: 2007
Similar Items: Find
Add

3
Artist: Sanzeeda, Ali
Title: Untitled
Date: 2007
Similar Items: Find
Add

4
Artist: Allen, Sarah-Jane
Title: Untitled
Date: 2007
Similar Items: Find
Add

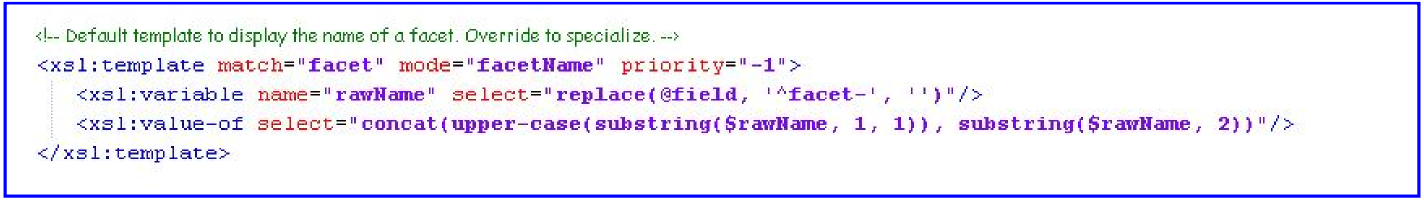
5
Artist: Arellano, Lupe
Title: Untitled
Date: 2007
Similar Items: Find
Add

In the above example, an underscore appears in the facet named *Special_collection*. Facet titles are defined in *scaPreFilter.xsl* (extract follows).



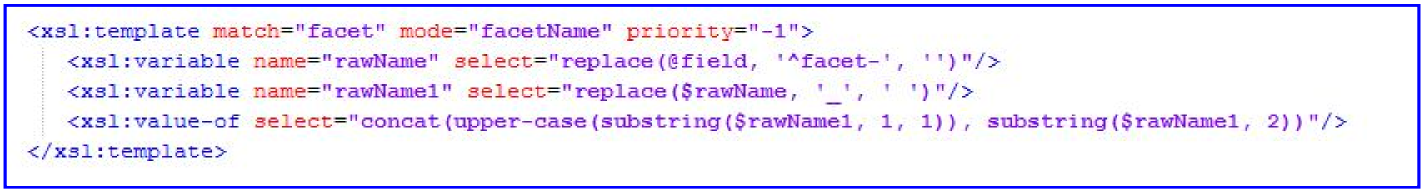
```
<!--facet special collection  -->
<xsl:template name="get-sca-facet-special_collection">
   <xsl:choose>
      <xsl:when test="/sca_record/special_collections">
         <xsl:for-each select="/sca_record/special_collections/special_collection">
            <facet-special_collection xtf:meta="true" xtf:indexOnly="yes" xtf:tokenize="no">
               <xsl:value-of select="."/>
            </facet-special_collection>
            </xsl:for-each>
      </xsl:when>
   </xsl:choose>
</xsl:template>
```

Facet title display may be altered by overriding a template within *resultFormatterCommon.xsl*.



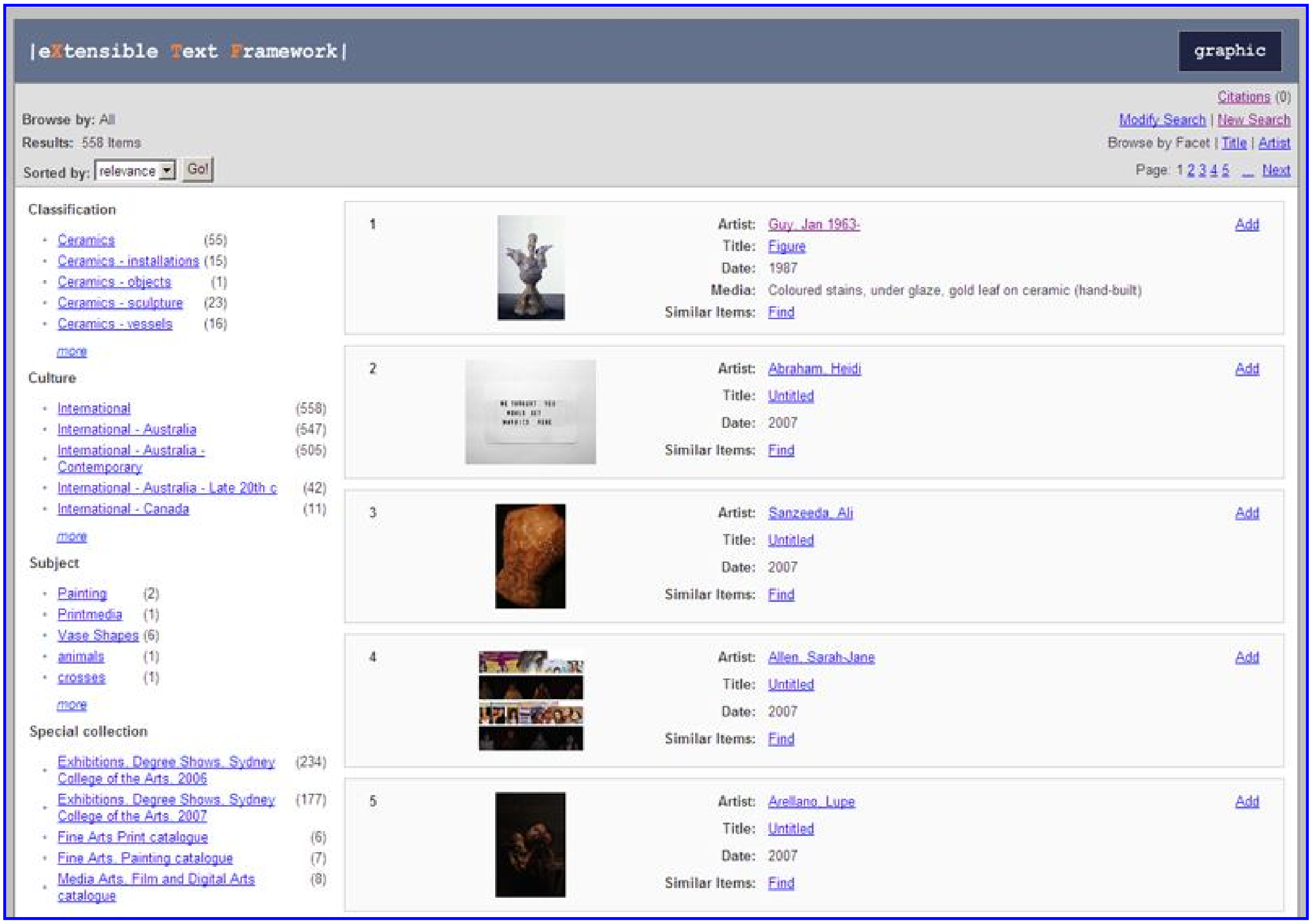
```
<!-- Default template to display the name of a facet. Override to specialize. -->
<xsl:template match="facet" mode="facetName" priority="-1">
   <xsl:variable name="rawName" select="replace(@field, '^facet-', '')"/>
   <xsl:value-of select="concat(upper-case(substring($rawName, 1, 1)), substring($rawName, 2))"/>
</xsl:template>
```

In this version of the template, an underscore is replaced by a single space.



```
<xsl:template match="facet" mode="facetName" priority="-1">
    <xsl:variable name="rawName" select="replace(@field, '^facet-', '')"/>
    <xsl:variable name="rawName1" select="replace($rawName, '_', ' ')"/>
    <xsl:value-of select="concat(upper-case(substring($rawName1, 1, 1)), substring($rawName1, 2))"/>
</xsl:template>
```

Example results page showing the Special collection facet title without an underscore.



Citations (0)
Modify Search | New Search
Browse by Facet | Title | Artist
Page: 1 2 3 4 5 ... Next

Browse by: All
Results: 558 Items
Sorted by: relevance Go!

Classification

- Ceramics (55)
- Ceramics - installations (15)
- Ceramics - objects (1)
- Ceramics - sculpture (23)
- Ceramics - vessels (16)

more

Culture

- International (558)
- International - Australia (547)
- International - Australia - Contemporary (505)
- International - Australia - Late 20th c (42)
- International - Canada (11)

more

Subject

- Painting (2)
- Printmedia (1)
- Vase Shapes (6)
- animals (1)
- crosses (1)

more

Special collection

- Exhibitions, Degree Shows, Sydney College of the Arts, 2006 (234)
- Exhibitions, Degree Shows, Sydney College of the Arts, 2007 (177)
- Fine Arts Print catalogue (6)
- Fine Arts, Painting catalogue (7)
- Media Arts, Film and Digital Arts catalogue (8)

1
Artist: Guy, Jan 1963-
Title: Figure
Date: 1987
Media: Coloured stains, under glaze, gold leaf on ceramic (hand-built)
Similar Items: Find
Add

2
Artist: Abraham, Heidi
Title: Untitled
Date: 2007
Similar Items: Find
Add

3
Artist: Sanzeeda, Ali
Title: Untitled
Date: 2007
Similar Items: Find
Add

4
Artist: Allen, Sarah-Jane
Title: Untitled
Date: 2007
Similar Items: Find
Add

5
Artist: Arellano, Lupe
Title: Untitled
Date: 2007
Similar Items: Find
Add

Next… Sorting, browsing and facets: preFilterCommon

Sorting, browsing and facets: preFilterCommon

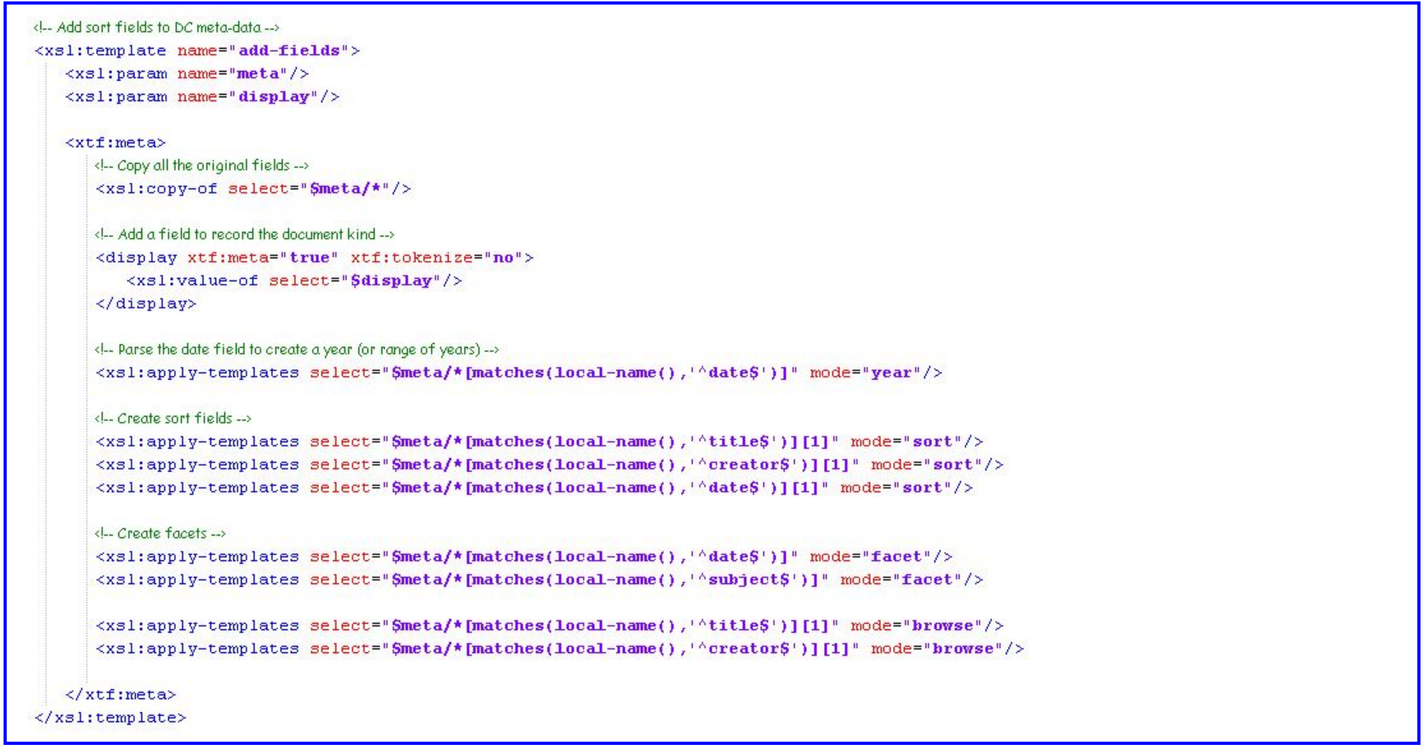
style/textIndexer/common/preFilterCommon.xsl

- Enabling facet, browse and sort fields
- Generating sorting
- Generating browsing
- Generating facets

Enabling facet, browse and sort fields

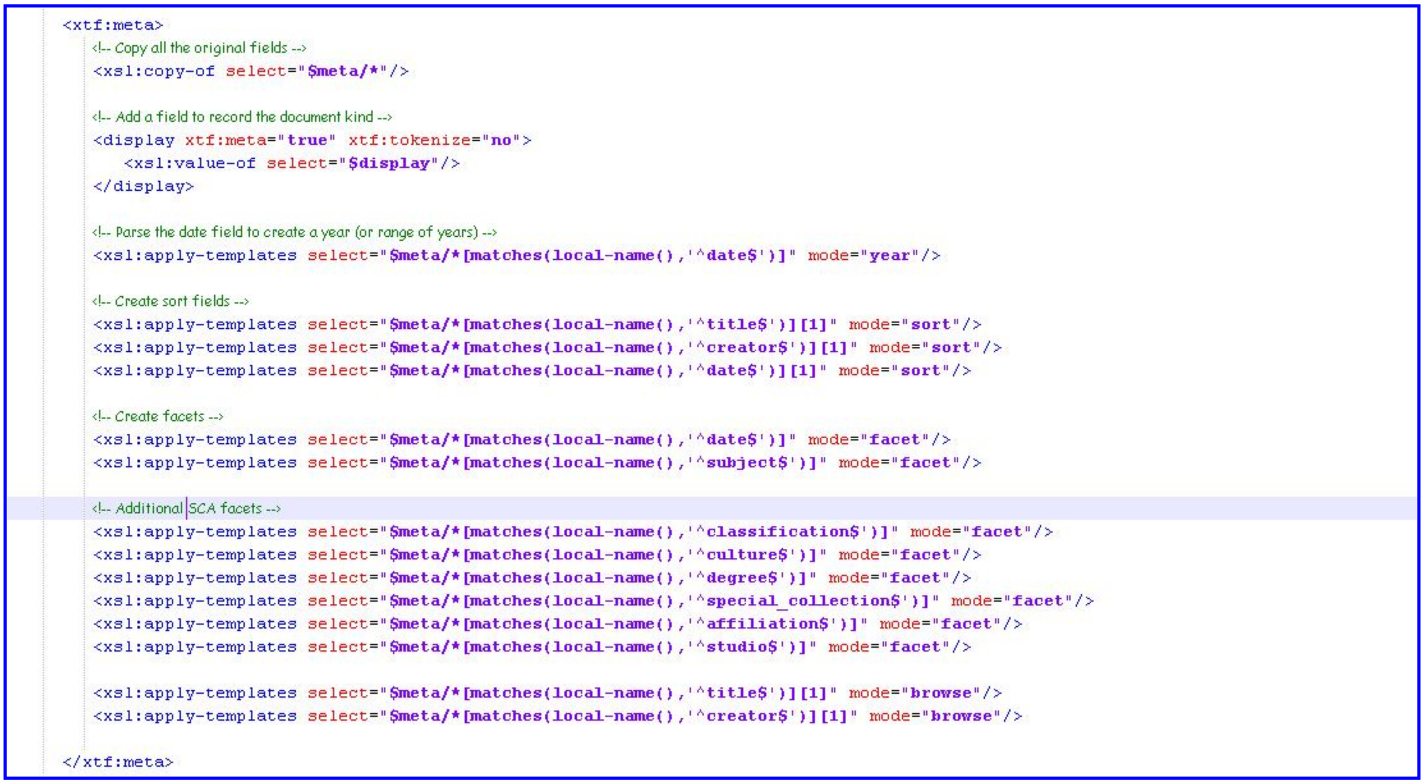
The XTF SCA site includes sort options for title and artist (creator), browse-by options for facet, title and artist, and additional facets for classification, culture, degree, special collection, affiliation and studio. The default version of preFilterCommon.xsl contains a number of sort, facet and browse options setup for elements including title, creator, and date, and these can be copied and adapted to support other facets based on other elements.

Default sort, browse and facet options



```
<!-- Add sort fields to DC meta-data -->
<xsl:template name="add-fields">
   <xsl:param name="meta"/>
   <xsl:param name="display"/>

   <xtf:meta>
      <!-- Copy all the original fields -->
      <xsl:copy-of select="$meta/*"/>

      <!-- Add a field to record the document kind -->
      <display xtf:meta="true" xtf:tokenize="no">
         <xsl:value-of select="$display"/>
      </display>

      <!-- Parse the date field to create a year (or range of years) -->
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^date$')]" mode="year"/>

      <!-- Create sort fields -->
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^title$')][1]" mode="sort"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^creator$')][1]" mode="sort"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^date$')][1]" mode="sort"/>

      <!-- Create facets -->
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^date$')]" mode="facet"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^subject$')]" mode="facet"/>

      <xsl:apply-templates select="$meta/*[matches(local-name(),'^title$')][1]" mode="browse"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^creator$')][1]" mode="browse"/>

   </xtf:meta>
</xsl:template>
```

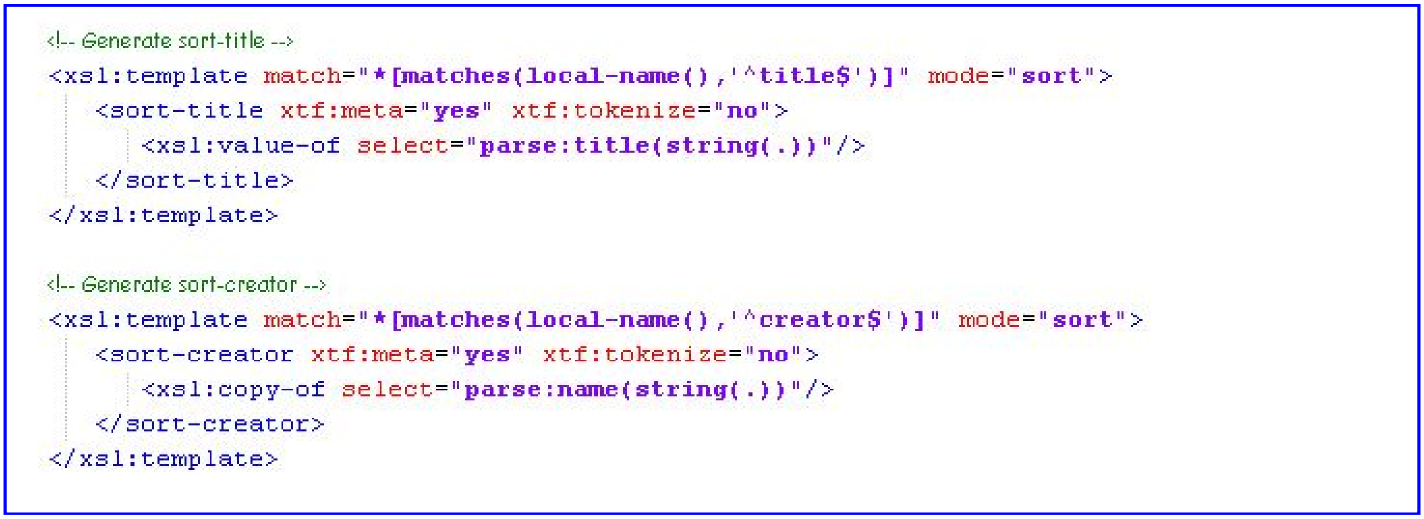
SCA sort, browse and facet options



```
   <xtf:meta>
      <!-- Copy all the original fields -->
      <xsl:copy-of select="$meta/*"/>

      <!-- Add a field to record the document kind -->
      <display xtf:meta="true" xtf:tokenize="no">
         <xsl:value-of select="$display"/>
      </display>

      <!-- Parse the date field to create a year (or range of years) -->
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^date$')]" mode="year"/>

      <!-- Create sort fields -->
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^title$')][1]" mode="sort"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^creator$')][1]" mode="sort"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^date$')][1]" mode="sort"/>

      <!-- Create facets -->
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^date$')]" mode="facet"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^subject$')]" mode="facet"/>

      <!-- Additional SCA facets -->
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^classification$')]" mode="facet"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^culture$')]" mode="facet"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^degree$')]" mode="facet"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^special_collection$')]" mode="facet"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^affiliation$')]" mode="facet"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^studio$')]" mode="facet"/>

      <xsl:apply-templates select="$meta/*[matches(local-name(),'^title$')][1]" mode="browse"/>
      <xsl:apply-templates select="$meta/*[matches(local-name(),'^creator$')][1]" mode="browse"/>

   </xtf:meta>
```

Generating sorting

Each sort option must be generated. The SCA collection sorts on title and creator (artist). The default preFilterCommon.xsl provides examples which may be copied and modified to enable sorting on other fields.

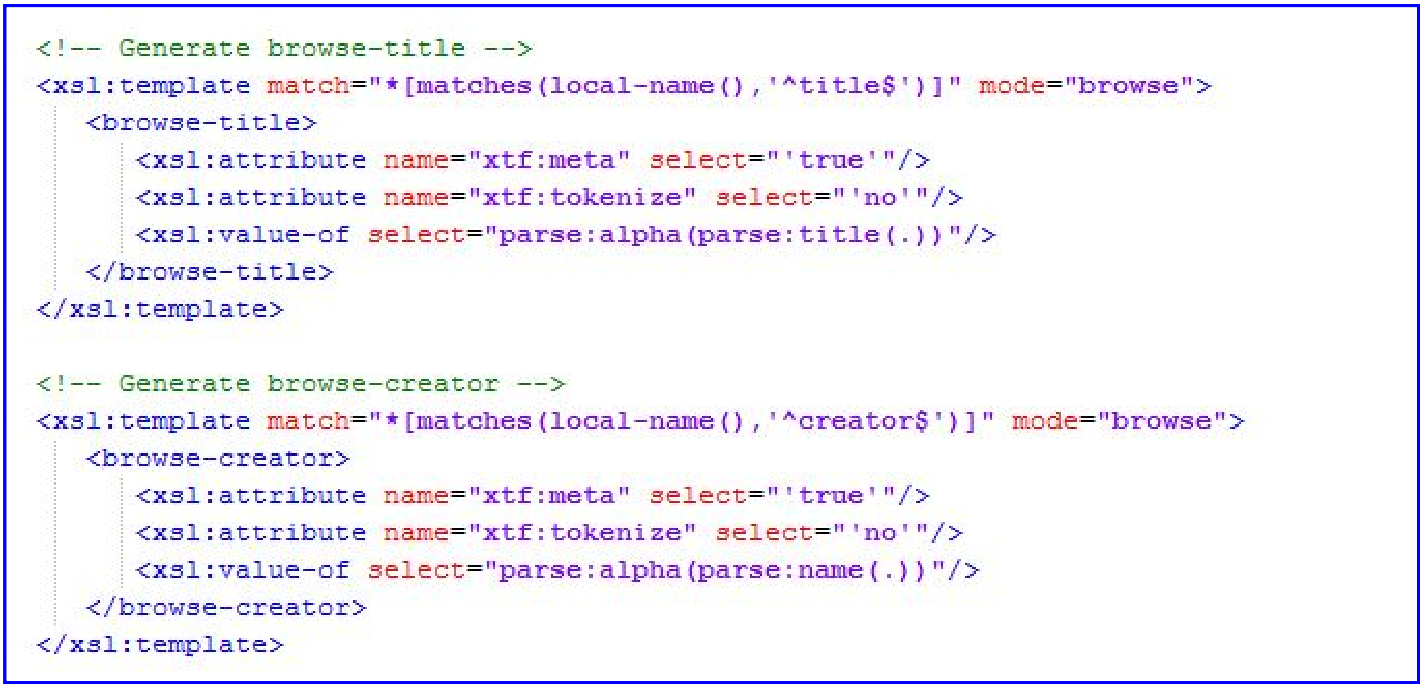


```
<!-- Generate sort-title -->
<xsl:template match="*[matches(local-name(),'^title$')]" mode="sort">
    <sort-title xtf:meta="yes" xtf:tokenize="no">
        <xsl:value-of select="parse:title(string(.))"/>
    </sort-title>
</xsl:template>

<!-- Generate sort-creator -->
<xsl:template match="*[matches(local-name(),'^creator$')]" mode="sort">
    <sort-creator xtf:meta="yes" xtf:tokenize="no">
        <xsl:copy-of select="parse:name(string(.))"/>
    </sort-creator>
</xsl:template>
```

In the extract above, parsing instructions for title and creator are different as they are processed quite differently.

Generating browsing

Each browse option must be generated. The SCA collection supports browsing on title and artist (creator). Examples available in the default preFilterCommon.xsl are adaptable to suit other fields.

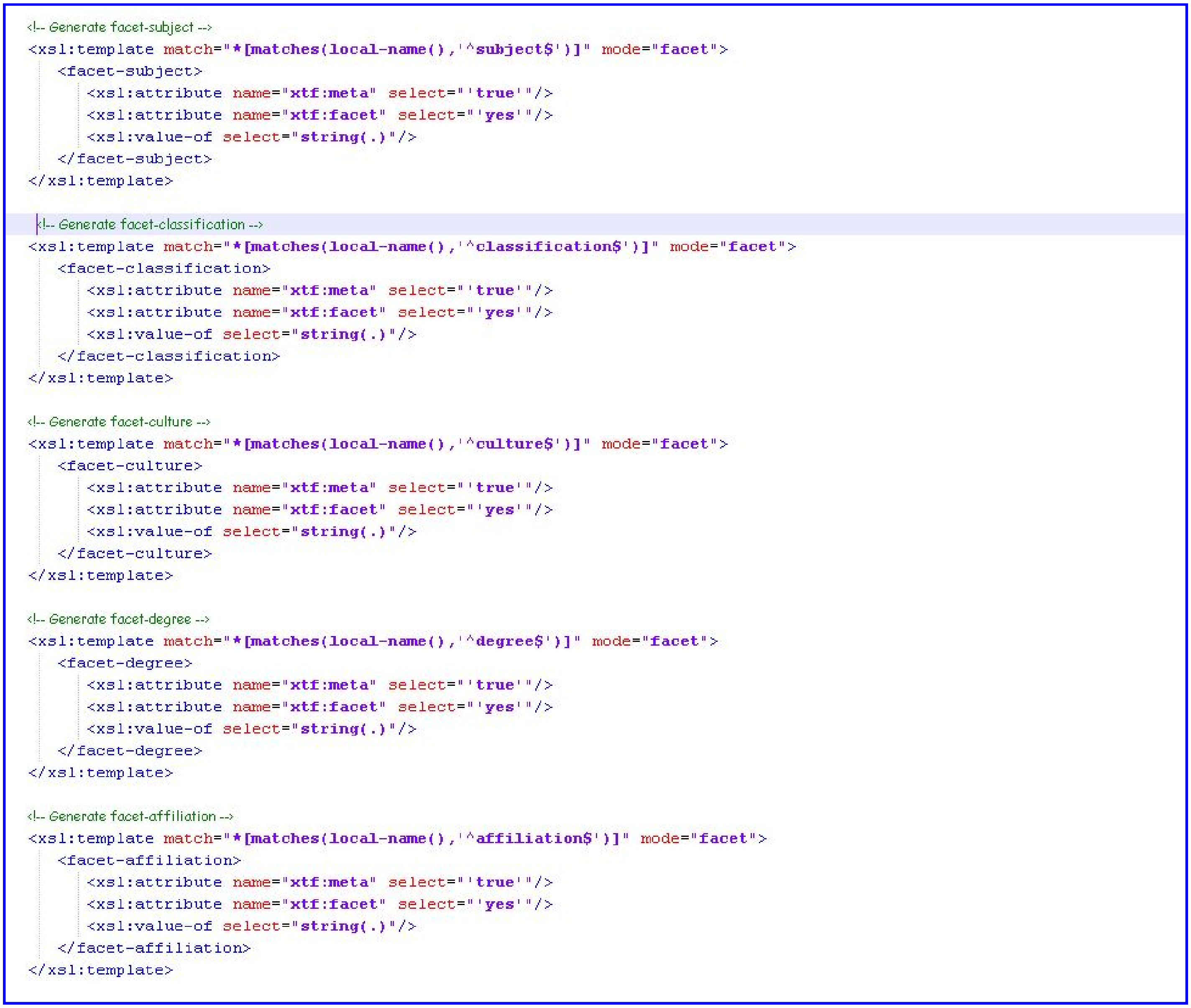


```
<!-- Generate browse-title -->
<xsl:template match="*[matches(local-name(),'^title$')]" mode="browse">
    <browse-title>
        <xsl:attribute name="xtf:meta" select="'true'"/>
        <xsl:attribute name="xtf:tokenize" select="'no'"/>
        <xsl:value-of select="parse:alpha(parse:title(.))"/>
    </browse-title>
</xsl:template>

<!-- Generate browse-creator -->
<xsl:template match="*[matches(local-name(),'^creator$')]" mode="browse">
    <browse-creator>
        <xsl:attribute name="xtf:meta" select="'true'"/>
        <xsl:attribute name="xtf:tokenize" select="'no'"/>
        <xsl:value-of select="parse:alpha(parse:name(.))"/>
    </browse-creator>
</xsl:template>
```

As with sorting, the parsing instructions are different for names and titles.

Generating facets

Each facet option must be generated. The following screen image shows facets for subject, classification, culture, degree and affiliation. Although most are specific to SCA, they are based on examples provided within the stylesheet.



```
<!-- Generate facet-subject -->
<xsl:template match="*[matches(local-name(),'^subject$')]" mode="facet">
   <facet-subject>
      <xsl:attribute name="xtf:meta" select="'true'"/>
      <xsl:attribute name="xtf:facet" select="'yes'"/>
      <xsl:value-of select="string(.)"/>
   </facet-subject>
</xsl:template>

<!-- Generate facet-classification -->
<xsl:template match="*[matches(local-name(),'^classification$')]" mode="facet">
   <facet-classification>
      <xsl:attribute name="xtf:meta" select="'true'"/>
      <xsl:attribute name="xtf:facet" select="'yes'"/>
      <xsl:value-of select="string(.)"/>
   </facet-classification>
</xsl:template>

<!-- Generate facet-culture -->
<xsl:template match="*[matches(local-name(),'^culture$')]" mode="facet">
   <facet-culture>
      <xsl:attribute name="xtf:meta" select="'true'"/>
      <xsl:attribute name="xtf:facet" select="'yes'"/>
      <xsl:value-of select="string(.)"/>
   </facet-culture>
</xsl:template>

<!-- Generate facet-degree -->
<xsl:template match="*[matches(local-name(),'^degree$')]" mode="facet">
   <facet-degree>
      <xsl:attribute name="xtf:meta" select="'true'"/>
      <xsl:attribute name="xtf:facet" select="'yes'"/>
      <xsl:value-of select="string(.)"/>
   </facet-degree>
</xsl:template>

<!-- Generate facet-affiliation -->
<xsl:template match="*[matches(local-name(),'^affiliation$')]" mode="facet">
   <facet-affiliation>
      <xsl:attribute name="xtf:meta" select="'true'"/>
      <xsl:attribute name="xtf:facet" select="'yes'"/>
      <xsl:value-of select="string(.)"/>
   </facet-affiliation>
</xsl:template>
```

Next… Sorting, browsing and facets: queryParser

Sorting, browsing and facets: queryParser

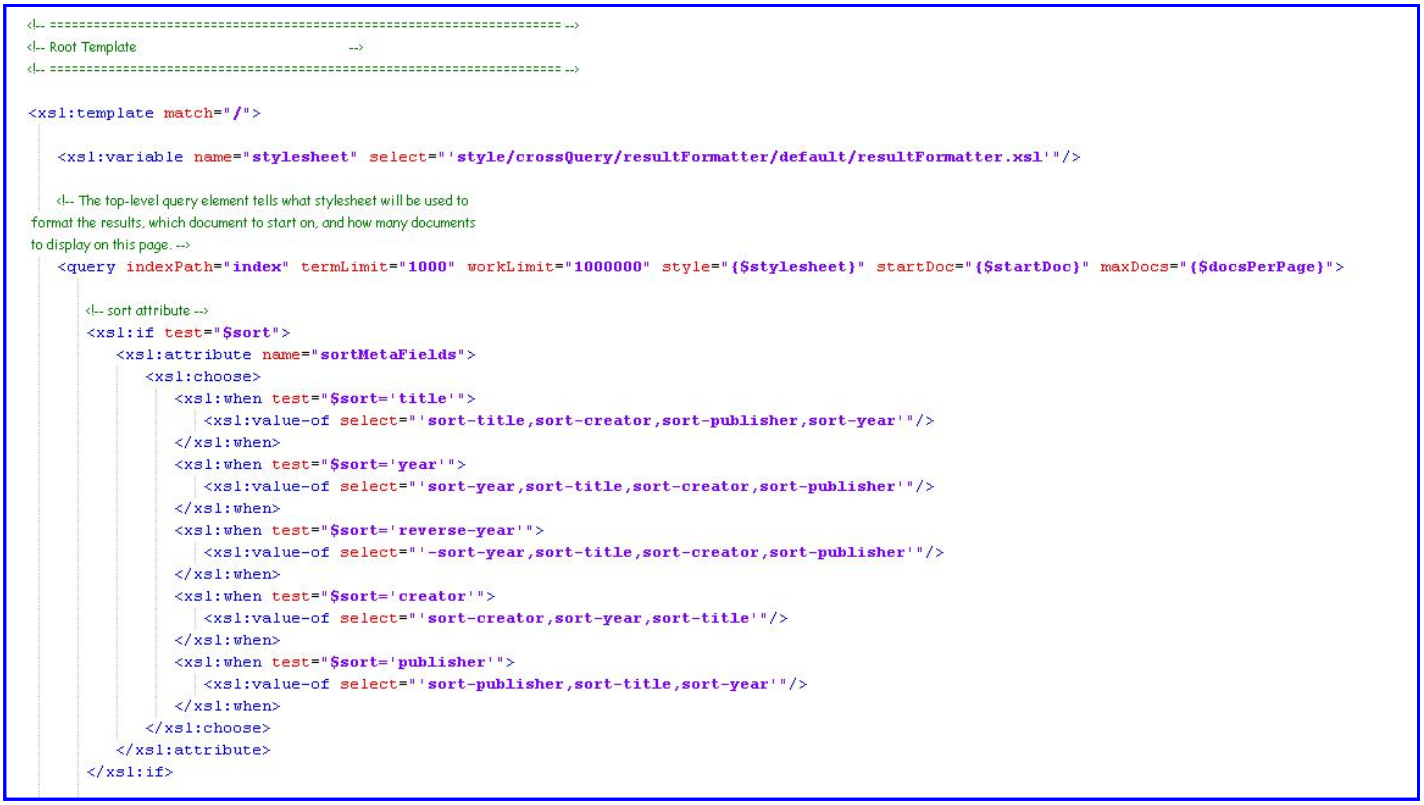
style/crossQuery/queryParser/default/queryParser.xsl

- Sorting
- Browsing
- Facets

Sorting

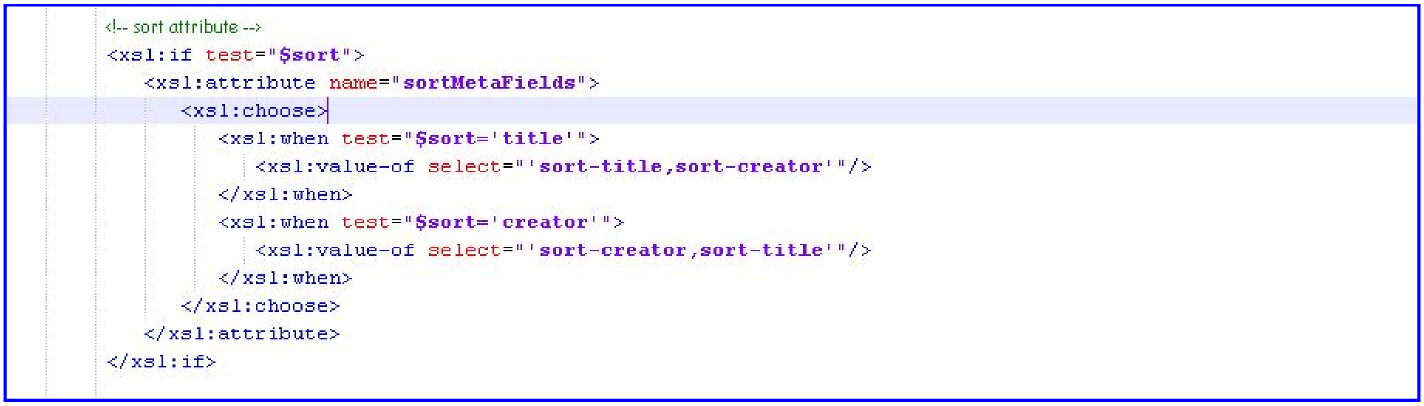
The default queryParser sorts by title, year, reverse year, creator and publisher. The SCA collection is sortable by creator (artist) and title, so the other options are removed, as illustrated below.

Default sort attribute



```
<!-- ====================================================================== -->
<!-- Root Template                                  -->
<!-- ====================================================================== -->

<xsl:template match="/">

   <xsl:variable name="stylesheet" select="'style/crossQuery/resultFormatter/default/resultFormatter.xsl'"/>

   <!-- The top-level query element tells what stylesheet will be used to
format the results, which document to start on, and how many documents
to display on this page. -->
   <query indexPath="index" termLimit="1000" workLimit="1000000" style="{$stylesheet}" startDoc="{$startDoc}" maxDocs="{$docsPerPage}">

      <!-- sort attribute -->
      <xsl:if test="$sort">
         <xsl:attribute name="sortMetaFields">
            <xsl:choose>
               <xsl:when test="$sort='title'">
                  <xsl:value-of select="'sort-title,sort-creator,sort-publisher,sort-year'"/>
               </xsl:when>
               <xsl:when test="$sort='year'">
                  <xsl:value-of select="'sort-year,sort-title,sort-creator,sort-publisher'"/>
               </xsl:when>
               <xsl:when test="$sort='reverse-year'">
                  <xsl:value-of select="'-sort-year,sort-title,sort-creator,sort-publisher'"/>
               </xsl:when>
               <xsl:when test="$sort='creator'">
                  <xsl:value-of select="'sort-creator,sort-year,sort-title'"/>
               </xsl:when>
               <xsl:when test="$sort='publisher'">
                  <xsl:value-of select="'sort-publisher,sort-title,sort-year'"/>
               </xsl:when>
            </xsl:choose>
         </xsl:attribute>
      </xsl:if>
```

Modified sort attribute

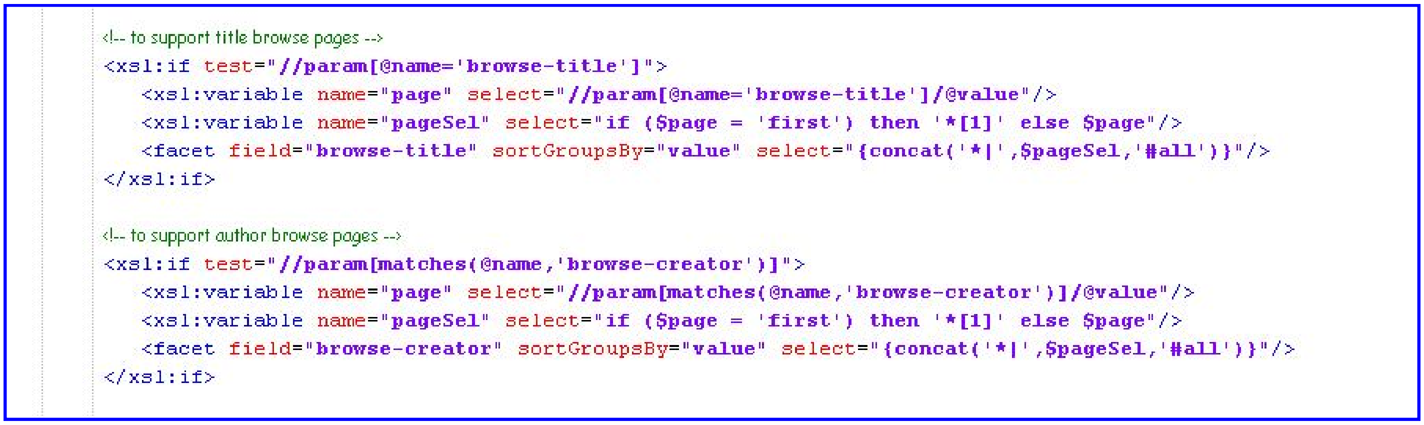


```
<!-- sort attribute -->
<xsl:if test="$sort">
   <xsl:attribute name="sortMetaFields">
      <xsl:choose>
         <xsl:when test="$sort='title'">
            <xsl:value-of select="'sort-title,sort-creator'"/>
         </xsl:when>
         <xsl:when test="$sort='creator'">
            <xsl:value-of select="'sort-creator,sort-title'"/>
         </xsl:when>
      </xsl:choose>
   </xsl:attribute>
</xsl:if>
```

Browsing

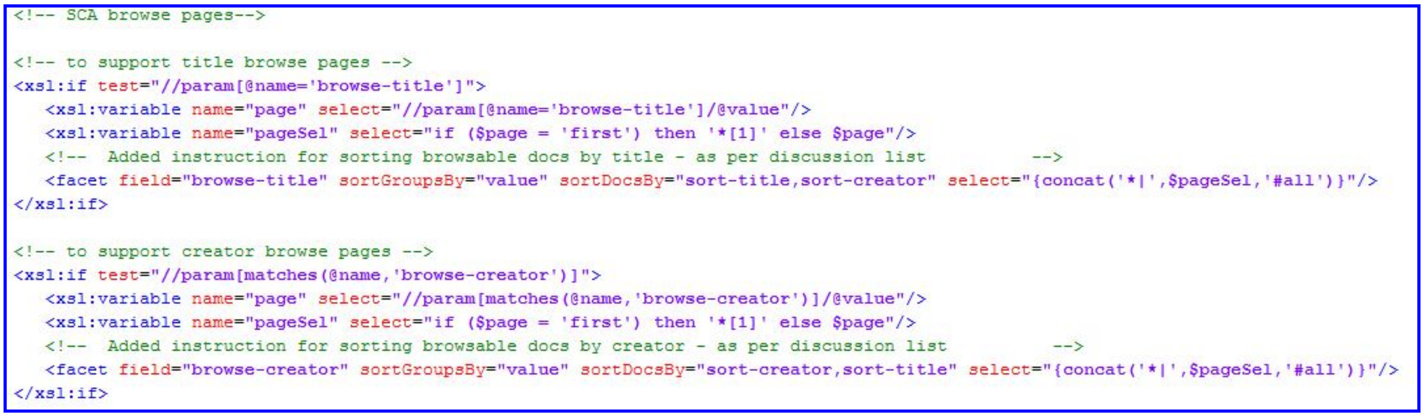
By default, queryParser is configured to support browsing by title and creator, and this meets the needs of the SCA collection. For XTF 2.1.1, an alteration to the default stylesheet is required to enable alphabetic sorting within browse sets. queryParser correctly sorts alphabetic divisions. All of the titles starting with *A* are gathered within an *A* group, and they are all displayed prior to those beginning with *B* (which are similarly correctly grouped within a *B* division). The default browse does not however include an instruction to ensure correct sorting within each of these divisions. The sequence of displayed titles within an alphabetic division corresponds to the order in which the XML records are indexed. A solution (provided via the XTF users list), is illustrated below.

Default browsing



```
<!-- to support title browse pages -->
<xsl:if test="//param[@name='browse-title']">
   <xsl:variable name="page" select="//param[@name='browse-title']/@value"/>
   <xsl:variable name="pageSel" select="if ($page = 'first') then '*[1]' else $page"/>
   <facet field="browse-title" sortGroupsBy="value" select="{concat('*|',$pageSel,'#all')}"/>
</xsl:if>

<!-- to support author browse pages -->
<xsl:if test="//param[matches(@name,'browse-creator')]">
   <xsl:variable name="page" select="//param[matches(@name,'browse-creator')]/@value"/>
   <xsl:variable name="pageSel" select="if ($page = 'first') then '*[1]' else $page"/>
   <facet field="browse-creator" sortGroupsBy="value" select="{concat('*|',$pageSel,'#all')}"/>
</xsl:if>
```

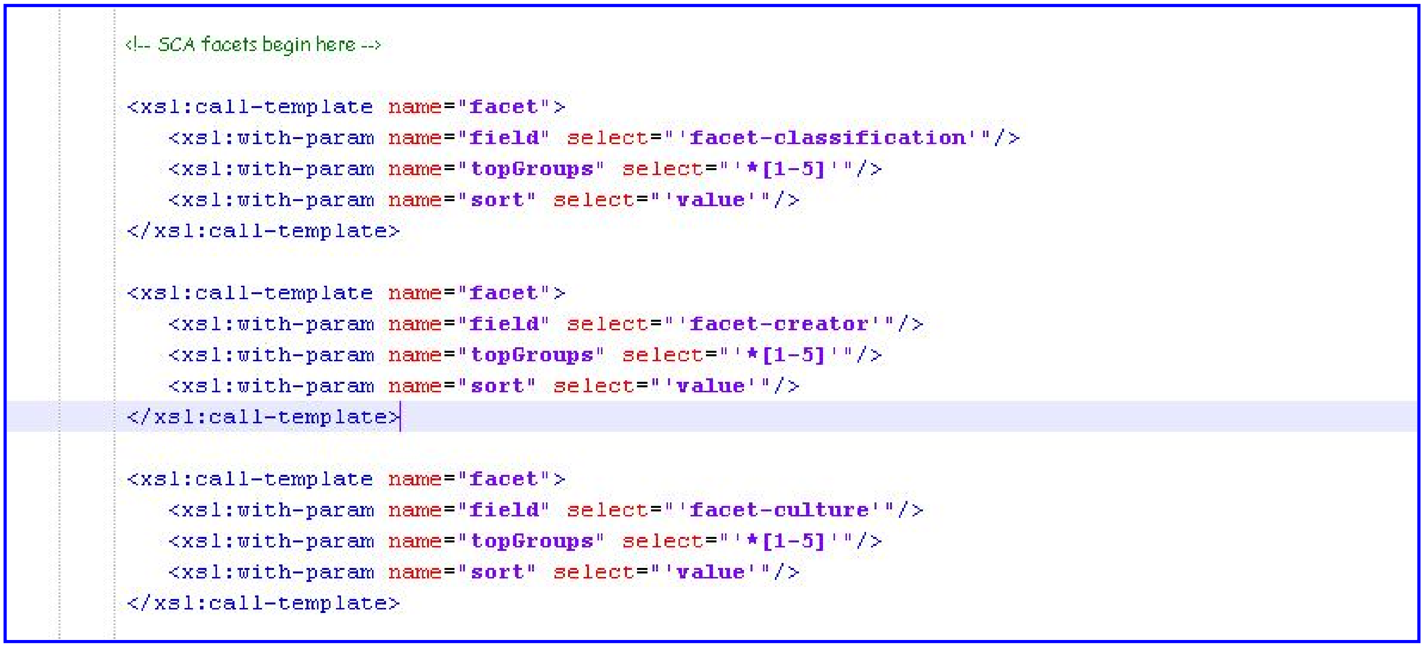
Modified browsing



```
<!-- SCA browse pages-->

<!-- to support title browse pages -->
<xsl:if test="//param[@name='browse-title']">
   <xsl:variable name="page" select="//param[@name='browse-title']/@value"/>
   <xsl:variable name="pageSel" select="if ($page = 'first') then '*[1]' else $page"/>
   <!--  Added instruction for sorting browsable docs by title - as per discussion list          -->
   <facet field="browse-title" sortGroupsBy="value" sortDocsBy="sort-title,sort-creator" select="{concat('*|',$pageSel,'#all')}"/>
</xsl:if>

<!-- to support creator browse pages -->
<xsl:if test="//param[matches(@name,'browse-creator')]">
   <xsl:variable name="page" select="//param[matches(@name,'browse-creator')]/@value"/>
   <xsl:variable name="pageSel" select="if ($page = 'first') then '*[1]' else $page"/>
   <!--  Added instruction for sorting browsable docs by creator - as per discussion list          -->
   <facet field="browse-creator" sortGroupsBy="value" sortDocsBy="sort-creator,sort-title" select="{concat('*|',$pageSel,'#all')}"/>
</xsl:if>
```

Facets

queryParser includes options for configuring how facets will be grouped at the time of page display. In the following example, I specify that no more than the first five facets in each category will display, and that they be alphabetically ordered (i.e. sorted by *value*).



```
<!-- SCA facets begin here -->

<xsl:call-template name="facet">
   <xsl:with-param name="field" select="'facet-classification'"/>
   <xsl:with-param name="topGroups" select="'*[1-5]'"/>
   <xsl:with-param name="sort" select="'value'"/>
</xsl:call-template>

<xsl:call-template name="facet">
   <xsl:with-param name="field" select="'facet-creator'"/>
   <xsl:with-param name="topGroups" select="'*[1-5]'"/>
   <xsl:with-param name="sort" select="'value'"/>
</xsl:call-template>

<xsl:call-template name="facet">
   <xsl:with-param name="field" select="'facet-culture'"/>
   <xsl:with-param name="topGroups" select="'*[1-5]'"/>
   <xsl:with-param name="sort" select="'value'"/>
</xsl:call-template>
```

In a previous section I describe how to identify metadata for faceted browsing by XTF and the associated steps involved in generating facets.

Next… Sorting, browsing and facets: resultFormatterCommon

Sorting, browsing and facets: resultFormatterCommon

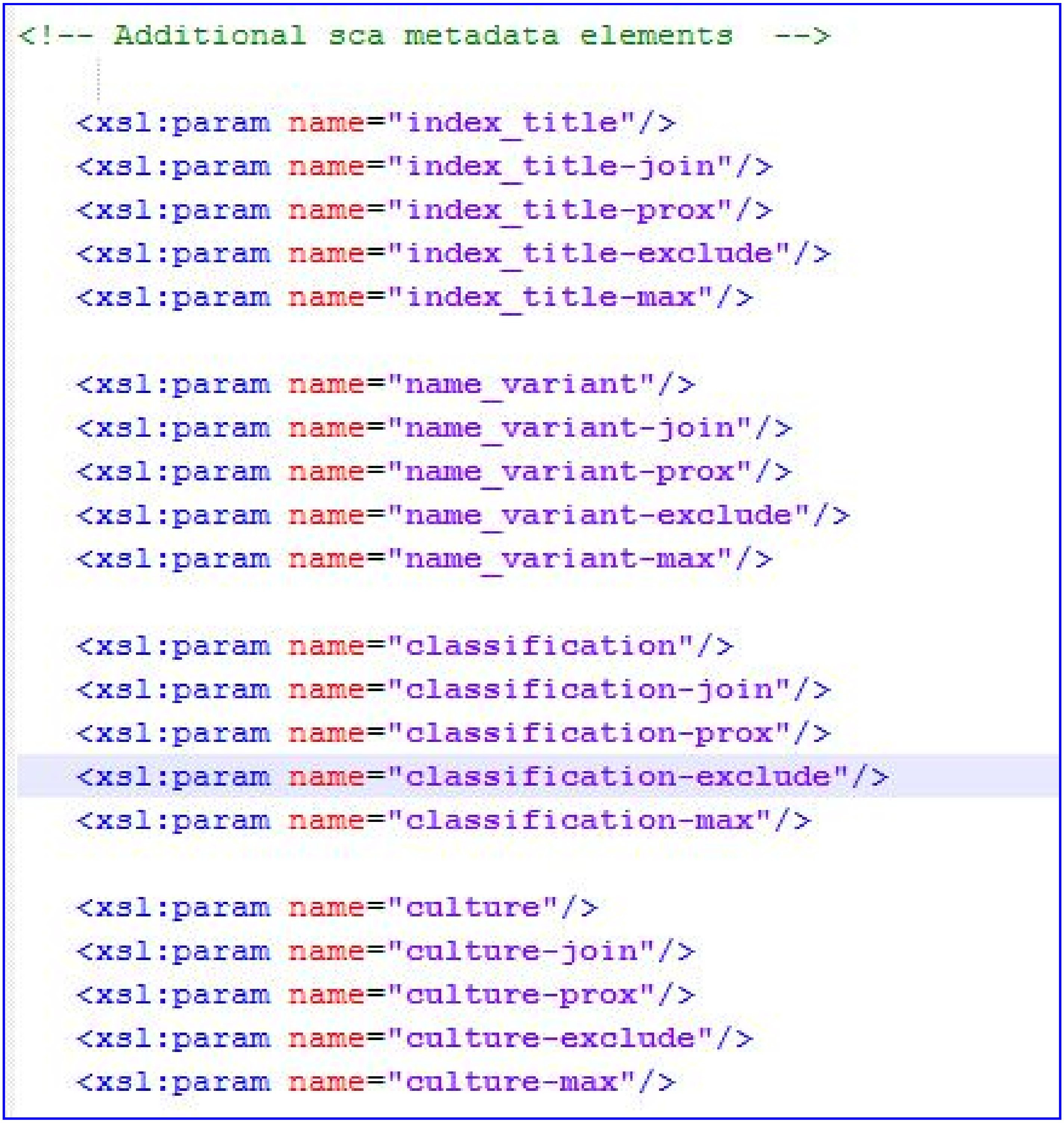
style/crossQuery/resultFormatter/common/resultFormatterCommon.xsl

- Metadata parameters
- Alpha browse parameters
- Sort options template
- Alpha list template

Metadata parameters

Within the parameters section, I include entries for SCA metadata elements, using the same format provided for Dublin Core elements.

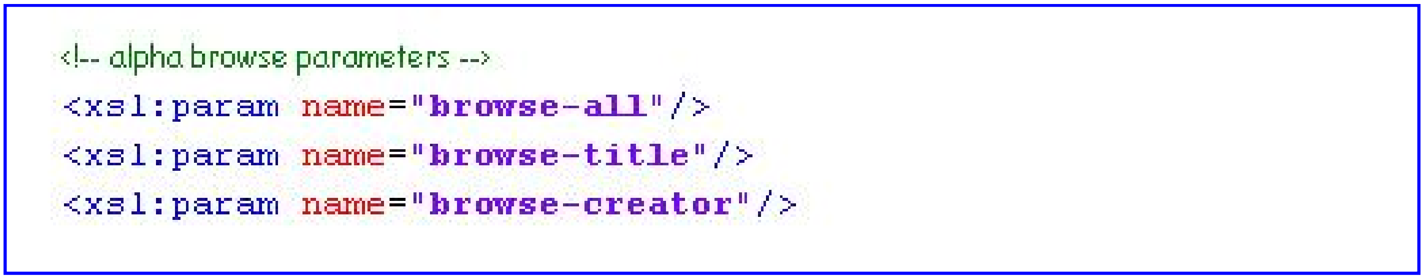
SCA metadata element parameters (extract)



```
<!-- Additional sca metadata elements  -->

   <xsl:param name="index_title"/>
   <xsl:param name="index_title-join"/>
   <xsl:param name="index_title-prox"/>
   <xsl:param name="index_title-exclude"/>
   <xsl:param name="index_title-max"/>

   <xsl:param name="name_variant"/>
   <xsl:param name="name_variant-join"/>
   <xsl:param name="name_variant-prox"/>
   <xsl:param name="name_variant-exclude"/>
   <xsl:param name="name_variant-max"/>

   <xsl:param name="classification"/>
   <xsl:param name="classification-join"/>
   <xsl:param name="classification-prox"/>
   <xsl:param name="classification-exclude"/>
   <xsl:param name="classification-max"/>

   <xsl:param name="culture"/>
   <xsl:param name="culture-join"/>
   <xsl:param name="culture-prox"/>
   <xsl:param name="culture-exclude"/>
   <xsl:param name="culture-max"/>
```

Alpha browse parameters

Within the alpha browse parameters section, there are already entries for *all*, *title* and *creator*. No additional browse options are needed for the SCA collection.

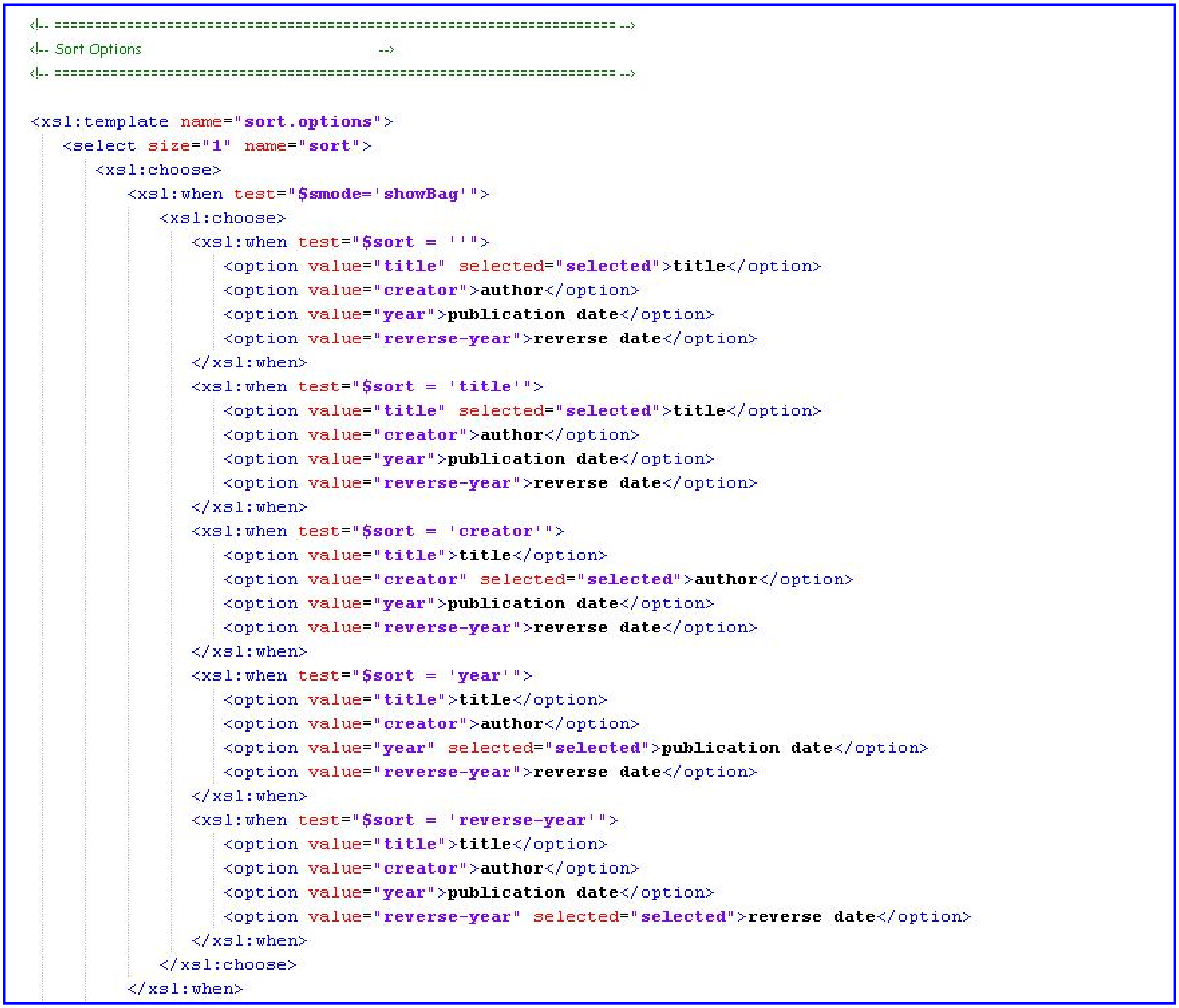


```
<!-- alpha browse parameters -->
<xsl:param name="browse-all"/>
<xsl:param name="browse-title"/>
<xsl:param name="browse-creator"/>
```

Sort options template

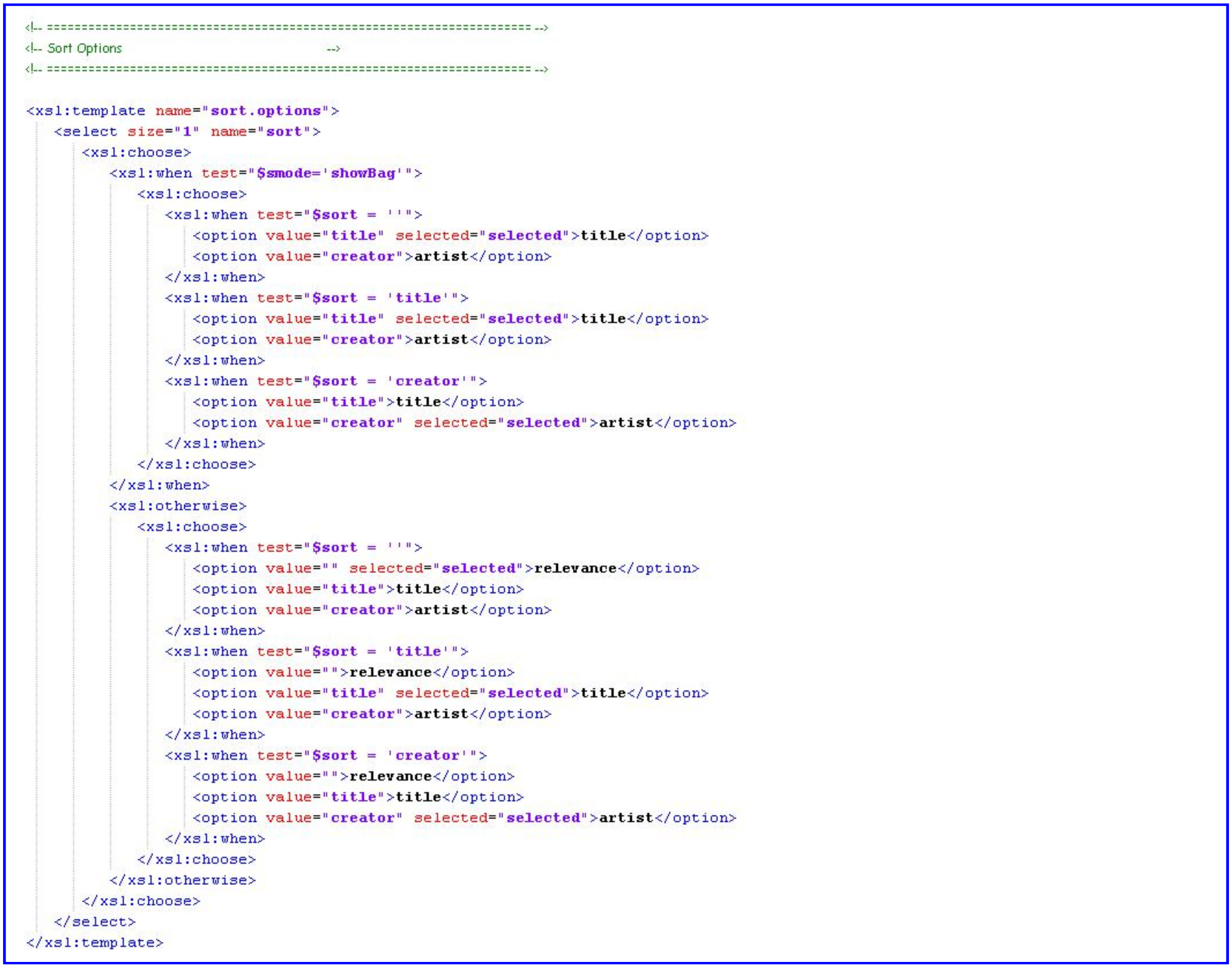
The *sort options* template controls display of the drop-down sort options which may be applied to retrieved records. SCA sorts on relevance, title and artist (which are available in the default stylesheet), so I remove the other options.

Default sort options template



```
<!-- ====================================================================== -->
<!-- Sort Options                                                           -->
<!-- ====================================================================== -->

<xsl:template name="sort.options">
   <select size="1" name="sort">
      <xsl:choose>
         <xsl:when test="$smode='showBag'">
            <xsl:choose>
               <xsl:when test="$sort = ''">
                  <option value="title" selected="selected">title</option>
                  <option value="creator">author</option>
                  <option value="year">publication date</option>
                  <option value="reverse-year">reverse date</option>
               </xsl:when>
               <xsl:when test="$sort = 'title'">
                  <option value="title" selected="selected">title</option>
                  <option value="creator">author</option>
                  <option value="year">publication date</option>
                  <option value="reverse-year">reverse date</option>
               </xsl:when>
               <xsl:when test="$sort = 'creator'">
                  <option value="title">title</option>
                  <option value="creator" selected="selected">author</option>
                  <option value="year">publication date</option>
                  <option value="reverse-year">reverse date</option>
               </xsl:when>
               <xsl:when test="$sort = 'year'">
                  <option value="title">title</option>
                  <option value="creator">author</option>
                  <option value="year" selected="selected">publication date</option>
                  <option value="reverse-year">reverse date</option>
               </xsl:when>
               <xsl:when test="$sort = 'reverse-year'">
                  <option value="title">title</option>
                  <option value="creator">author</option>
                  <option value="year">publication date</option>
                  <option value="reverse-year" selected="selected">reverse date</option>
               </xsl:when>
            </xsl:choose>
         </xsl:when>
```

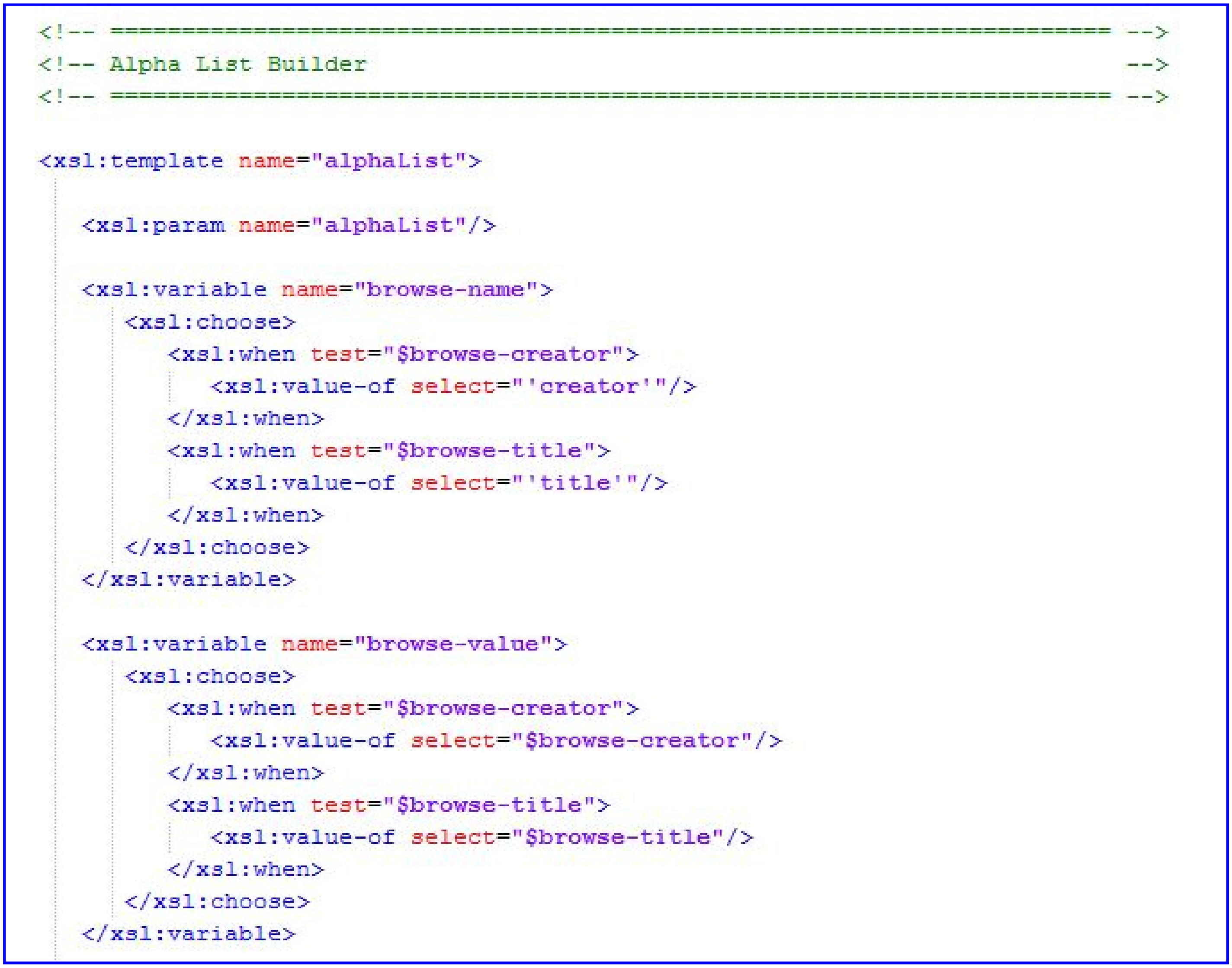
Modified sort options template



```
<!-- ====================================================================== -->
<!-- Sort Options                                                          -->
<!-- ====================================================================== -->

<xsl:template name="sort.options">
   <select size="1" name="sort">
      <xsl:choose>
         <xsl:when test="$smode='showBag'">
            <xsl:choose>
               <xsl:when test="$sort = ''">
                  <option value="title" selected="selected">title</option>
                  <option value="creator">artist</option>
               </xsl:when>
               <xsl:when test="$sort = 'title'">
                  <option value="title" selected="selected">title</option>
                  <option value="creator">artist</option>
               </xsl:when>
               <xsl:when test="$sort = 'creator'">
                  <option value="title">title</option>
                  <option value="creator" selected="selected">artist</option>
               </xsl:when>
            </xsl:choose>
         </xsl:when>
         <xsl:otherwise>
            <xsl:choose>
               <xsl:when test="$sort = ''">
                  <option value="" selected="selected">relevance</option>
                  <option value="title">title</option>
                  <option value="creator">artist</option>
               </xsl:when>
               <xsl:when test="$sort = 'title'">
                  <option value="">relevance</option>
                  <option value="title" selected="selected">title</option>
                  <option value="creator">artist</option>
               </xsl:when>
               <xsl:when test="$sort = 'creator'">
                  <option value="">relevance</option>
                  <option value="title">title</option>
                  <option value="creator" selected="selected">artist</option>
               </xsl:when>
            </xsl:choose>
         </xsl:otherwise>
      </xsl:choose>
   </select>
</xsl:template>
```

Alpha list template

Within the alphaList template, there are already *browse-name* and *browse-value* options for title and creator (artist). This suits the SCA collection.



```
<!-- ====================================================================== -->
<!-- Alpha List Builder                                                     -->
<!-- ====================================================================== -->

<xsl:template name="alphaList">

   <xsl:param name="alphaList"/>

   <xsl:variable name="browse-name">
      <xsl:choose>
         <xsl:when test="$browse-creator">
            <xsl:value-of select="'creator'"/>
         </xsl:when>
         <xsl:when test="$browse-title">
            <xsl:value-of select="'title'"/>
         </xsl:when>
      </xsl:choose>
   </xsl:variable>

   <xsl:variable name="browse-value">
      <xsl:choose>
         <xsl:when test="$browse-creator">
            <xsl:value-of select="$browse-creator"/>
         </xsl:when>
         <xsl:when test="$browse-title">
            <xsl:value-of select="$browse-title"/>
         </xsl:when>
      </xsl:choose>
   </xsl:variable>
```

Next… Sorting, browsing and facets: resultFormatter

Sorting, browsing and facets: resultFormatter

style/crossQuery/resultFormatter/default/resultFormatter.xsl

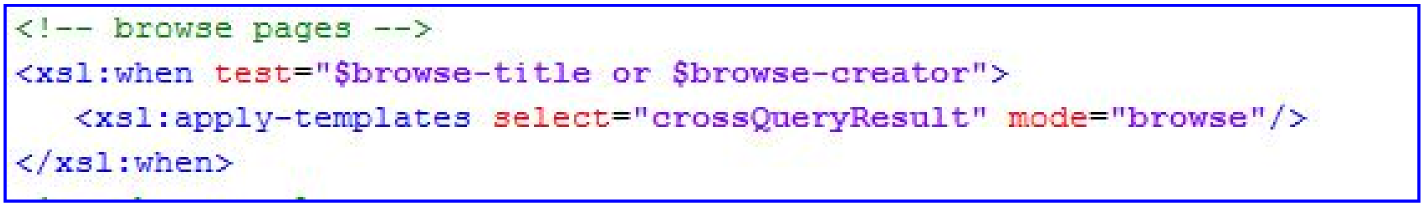
In an earlier section I describe changes to resultFormatter to enable display of brief records with accompanying clickable thumbnail images. The current section covers browsing by facets.

In the SCA example, very few changes are needed, as the required sort and browse options are catered for by the default version of resultFormatter. The examples are however readily adaptable for other cases. (Within a plant sciences image collection named eBot, I configured browse and sort options for title, creator, family, genus and species).

- Root template
- Browse template
- Browse links template
- Results template

Root template

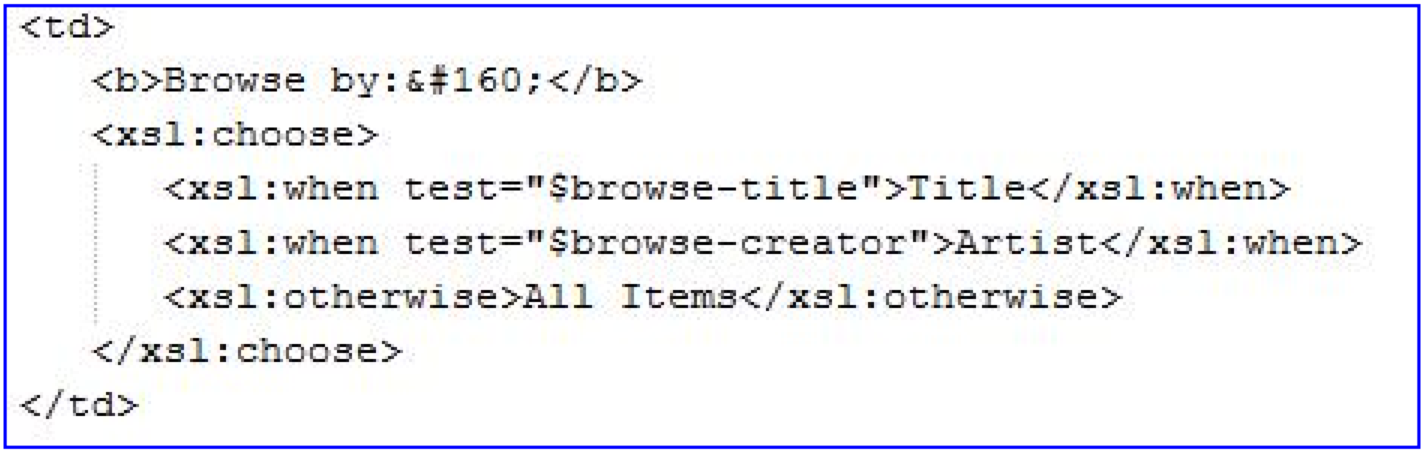
Within the *root* template, there are options to $browse-title and $browse-creator



```
<!-- browse pages -->
<xsl:when test="$browse-title or $browse-creator">
   <xsl:apply-templates select="crossQueryResult" mode="browse"/>
</xsl:when>
```

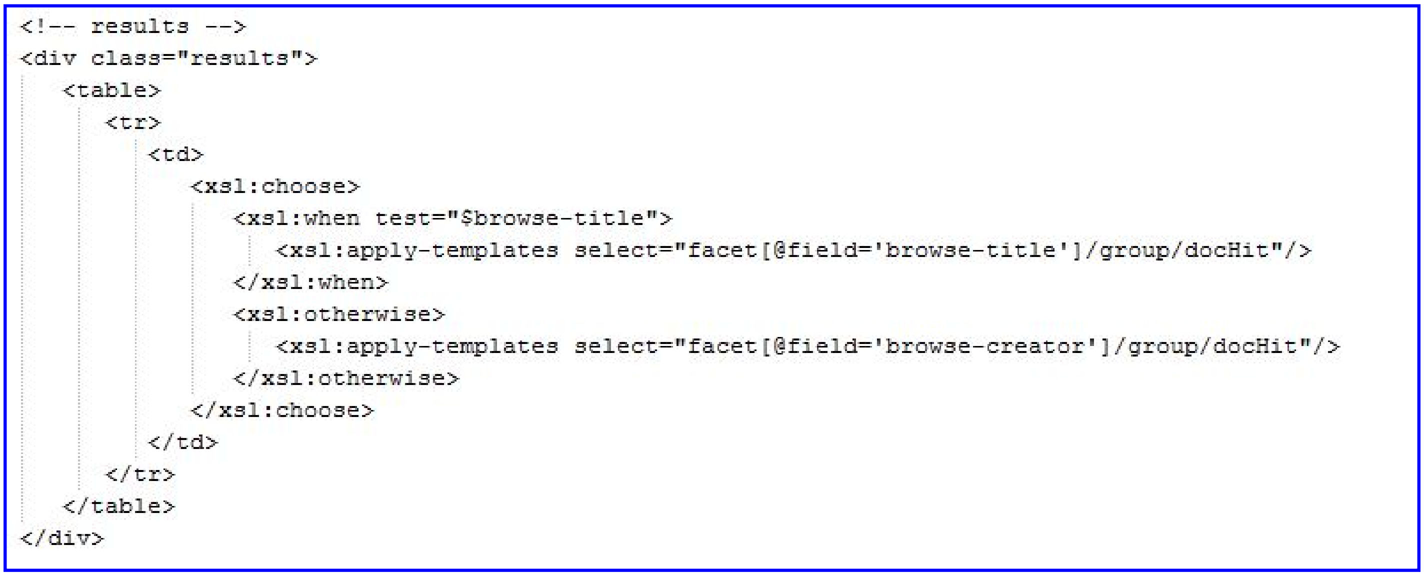
Browse template

Within the *browse by* section of the *browse template*, there are entries for $browse-title and $browse-creator



```
<td>
   <b>Browse by:&#160;</b>
   <xsl:choose>
      <xsl:when test="$browse-title">Title</xsl:when>
      <xsl:when test="$browse-creator">Artist</xsl:when>
      <xsl:otherwise>All Items</xsl:otherwise>
   </xsl:choose>
</td>
```

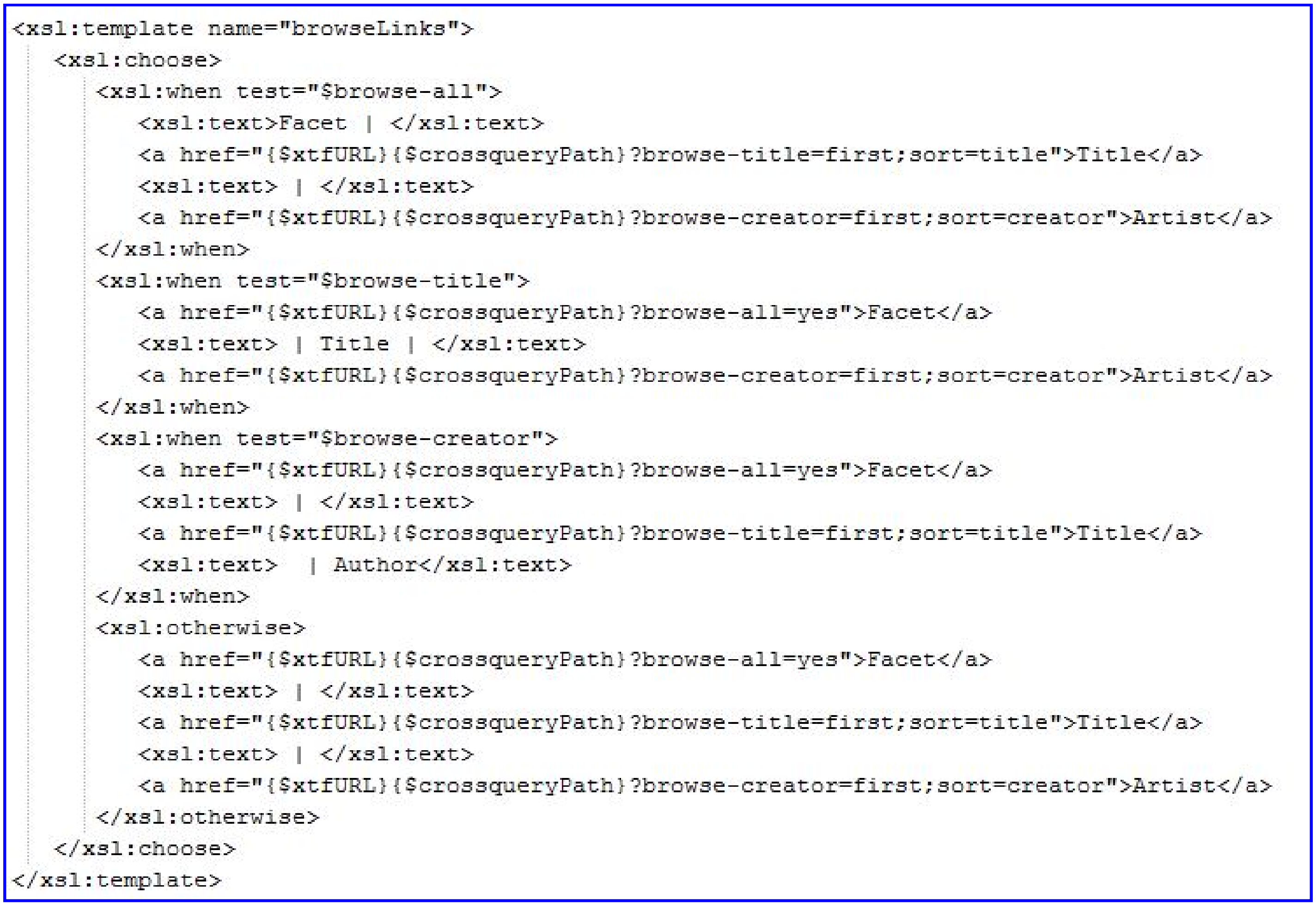
Within the *results* section of the *browse* template, there are entries for browse-title and browse-creator



```
<!-- results -->
<div class="results">
   <table>
      <tr>
         <td>
            <xsl:choose>
               <xsl:when test="$browse-title">
                  <xsl:apply-templates select="facet[@field='browse-title']/group/docHit"/>
               </xsl:when>
               <xsl:otherwise>
                  <xsl:apply-templates select="facet[@field='browse-creator']/group/docHit"/>
               </xsl:otherwise>
            </xsl:choose>
         </td>
      </tr>
   </table>
</div>
```

Browse links template

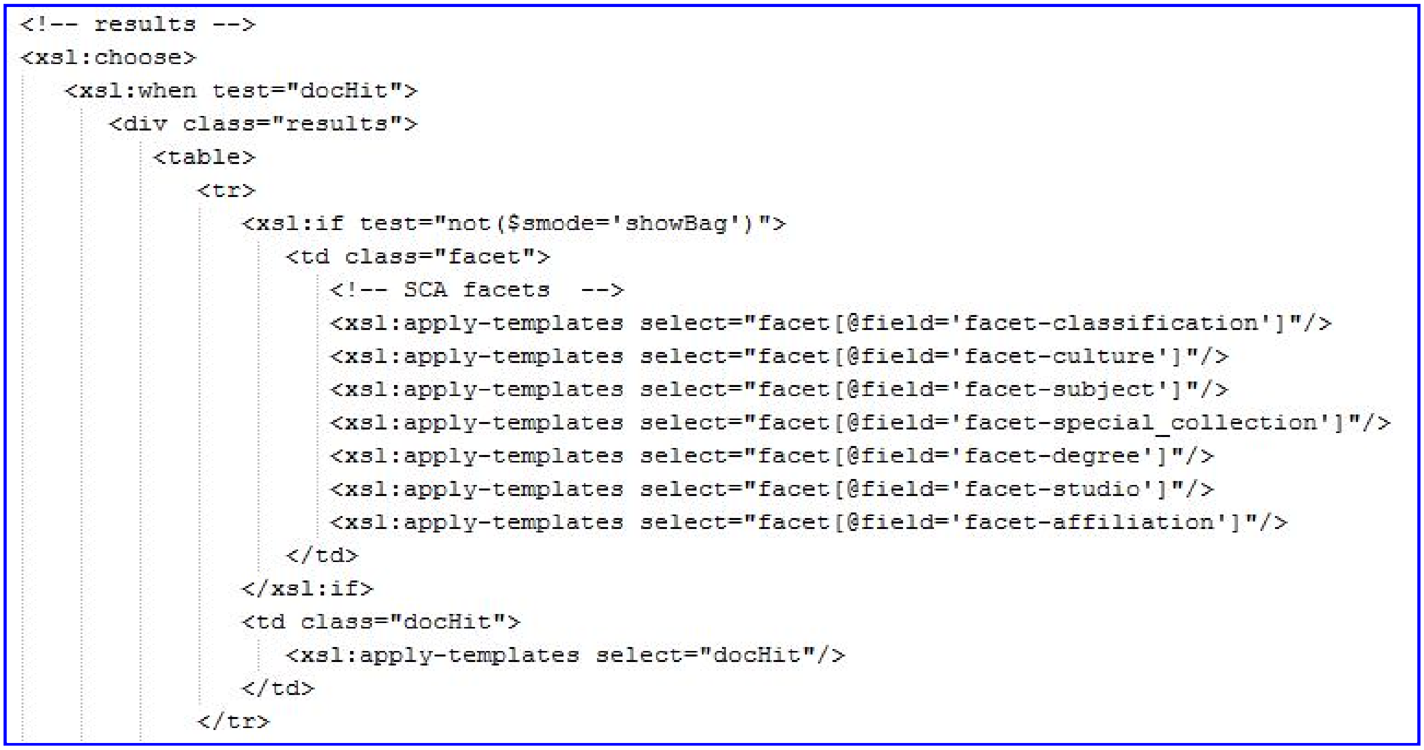
Within the *browseLinks* template, there are entries for browse-title and browse-creator. The only change involves altering display text to read *Artist* rather than *Author*.



```
<xsl:template name="browseLinks">
   <xsl:choose>
      <xsl:when test="$browse-all">
         <xsl:text>Facet | </xsl:text>
         <a href="{$xtfURL}{$crossqueryPath}?browse-title=first;sort=title">Title</a>
         <xsl:text> | </xsl:text>
         <a href="{$xtfURL}{$crossqueryPath}?browse-creator=first;sort=creator">Artist</a>
      </xsl:when>
      <xsl:when test="$browse-title">
         <a href="{$xtfURL}{$crossqueryPath}?browse-all=yes">Facet</a>
         <xsl:text> | Title | </xsl:text>
         <a href="{$xtfURL}{$crossqueryPath}?browse-creator=first;sort=creator">Artist</a>
      </xsl:when>
      <xsl:when test="$browse-creator">
         <a href="{$xtfURL}{$crossqueryPath}?browse-all=yes">Facet</a>
         <xsl:text> | </xsl:text>
         <a href="{$xtfURL}{$crossqueryPath}?browse-title=first;sort=title">Title</a>
         <xsl:text>  | Author</xsl:text>
      </xsl:when>
      <xsl:otherwise>
         <a href="{$xtfURL}{$crossqueryPath}?browse-all=yes">Facet</a>
         <xsl:text> | </xsl:text>
         <a href="{$xtfURL}{$crossqueryPath}?browse-title=first;sort=title">Title</a>
         <xsl:text> | </xsl:text>
         <a href="{$xtfURL}{$crossqueryPath}?browse-creator=first;sort=creator">Artist</a>
      </xsl:otherwise>
   </xsl:choose>
</xsl:template>
```

Results template

The results section of the results template lists the facets which will display. (Configuration and generation of facets is handled elsewhere)



```
<!-- results -->
<xsl:choose>
   <xsl:when test="docHit">
      <div class="results">
         <table>
            <tr>
               <xsl:if test="not($smode='showBag')">
                  <td class="facet">
                     <!-- SCA facets  -->
                     <xsl:apply-templates select="facet[@field='facet-classification']"/>
                     <xsl:apply-templates select="facet[@field='facet-culture']"/>
                     <xsl:apply-templates select="facet[@field='facet-subject']"/>
                     <xsl:apply-templates select="facet[@field='facet-special_collection']"/>
                     <xsl:apply-templates select="facet[@field='facet-degree']"/>
                     <xsl:apply-templates select="facet[@field='facet-studio']"/>
                     <xsl:apply-templates select="facet[@field='facet-affiliation']"/>
                  </td>
               </xsl:if>
               <td class="docHit">
                  <xsl:apply-templates select="docHit"/>
               </td>
            </tr>
```

Next… Hierarchical facets

Hierarchical facets

Hierarchical facets provide a means for expressing nested relationships within classification schemes and taxonomies and enable people to browse a taxonomic pathway through a collection. They also provide a way to efficiently use screen-space, as illustrated in the screenshots below. This is particularly the case with the eBot plant sciences collection which is underpinned by a deep taxonomy.

- Expressing hierarchical relationships
- Extract from an SCA XML record showing hierarchical facet notation
- Screenshots: SCA Archive, top-, first- second- and third-level facet navigation
- Screenshot: Fish-bone image collection, top level taxonomy facet
- Screenshot: eBot plant sciences collection, expanded taxonomy pathway

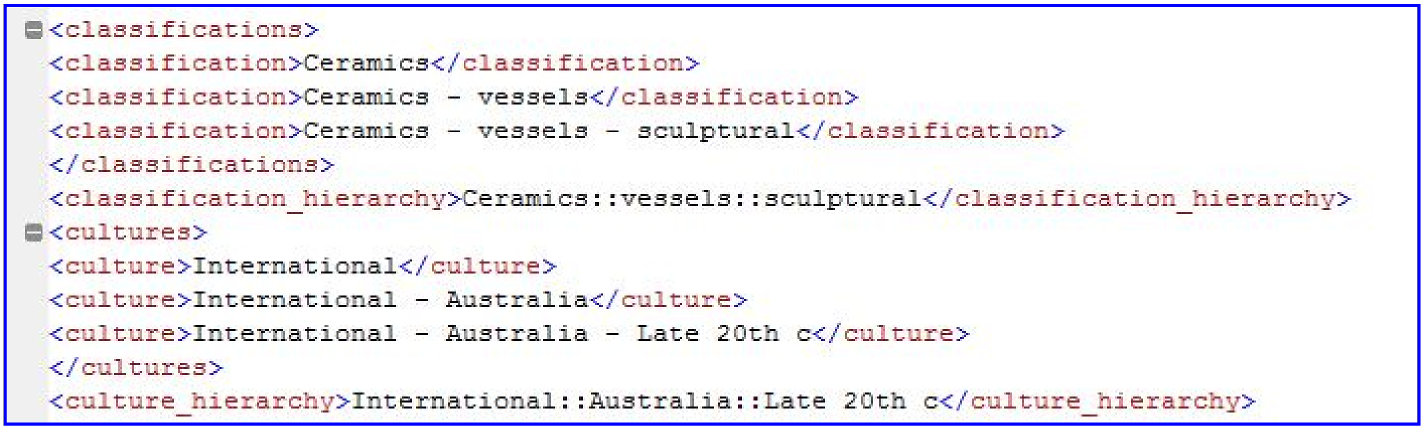
***Screenshots on this page were taken following site styling.*

Expressing hierarchical relationships

Apart from the steps outlined in the previous section, XTF needs to be provided with an expression of the relationships between the various levels within the hierarchical structure. Relationships are indicated using '::' notation, (as explained in the XTF documentation on the subject).

Extract from an SCA XML record showing hierarchical facet notation

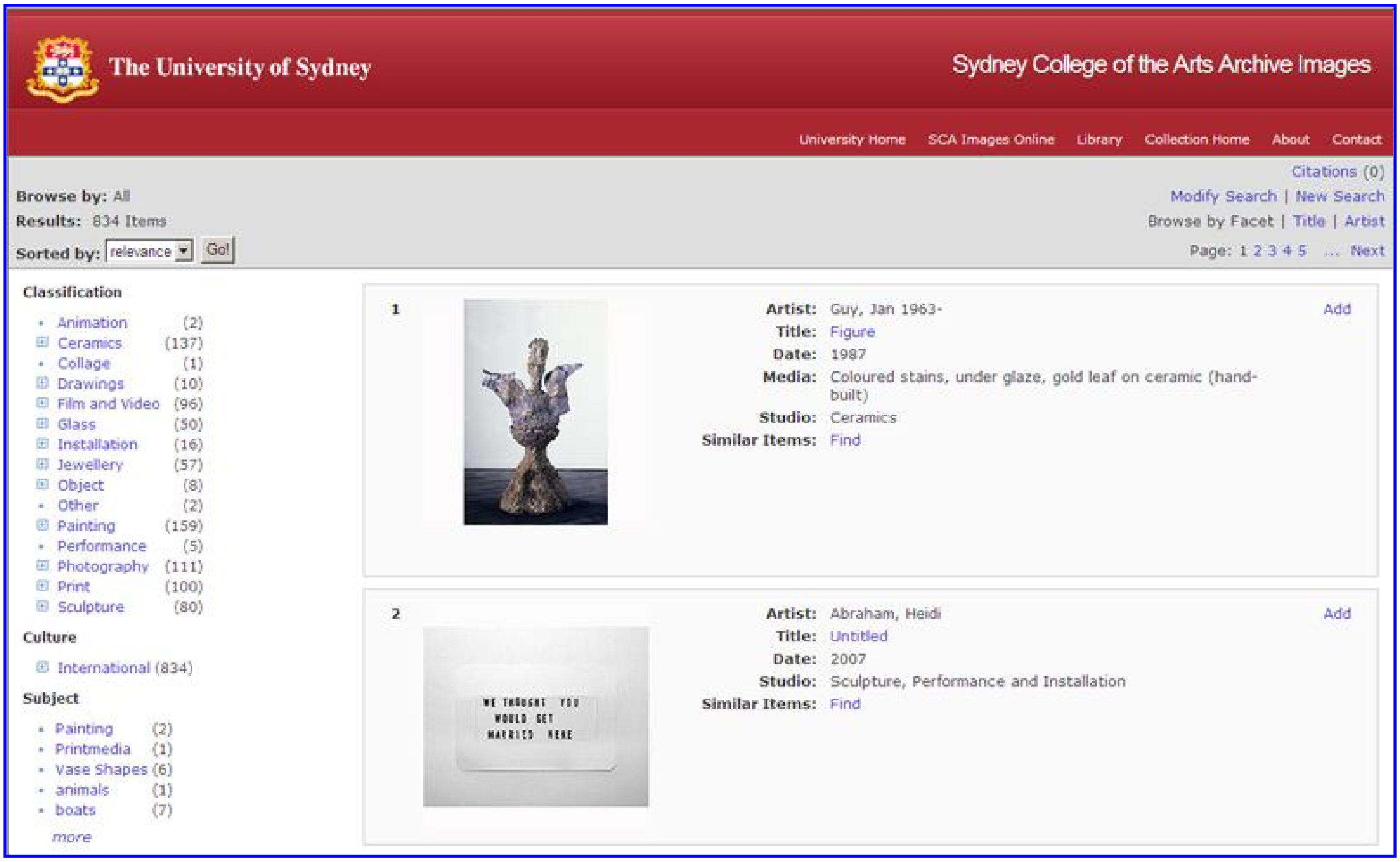
I followed the example provided in exercise 8 of Martin Haye's tutorial. As part of the pre-processing of metadata (not covered in this discussion), I use Python scripting to create an additional metadata field in each record.



```
<classifications>
<classification>Ceramics</classification>
<classification>Ceramics - vessels</classification>
<classification>Ceramics - vessels - sculptural</classification>
</classifications>
<classification_hierarchy>Ceramics::vessels::sculptural</classification_hierarchy>
<cultures>
<culture>International</culture>
<culture>International - Australia</culture>
<culture>International - Australia - Late 20th c</culture>
</cultures>
<culture_hierarchy>International::Australia::Late 20th c</culture_hierarchy>
```

Screenshots: SCA archive, top-, first-, second- and third-level facet navigation

Top level classification facet.


The University of Sydney
Sydney College of the Arts Archive Images
University Home SCA Images Online Library Collection Home About Contact
Citations (0)
Modify Search | New Search
Browse by Facet | Title | Artist
Go!
Page: 1 2 3 4 5 ... Next

Browse by: All
Results: 834 Items
Sorted by: relevance

Classification

- Animation (2)
- Ceramics (137)
- Collage (1)
- Drawings (10)
- Film and Video (96)
- Glass (50)
- Installation (16)
- Jewellery (57)
- Object (8)
- Other (2)
- Painting (159)
- Performance (5)
- Photography (111)
- Print (100)
- Sculpture (80)

Culture

- International (834)

Subject

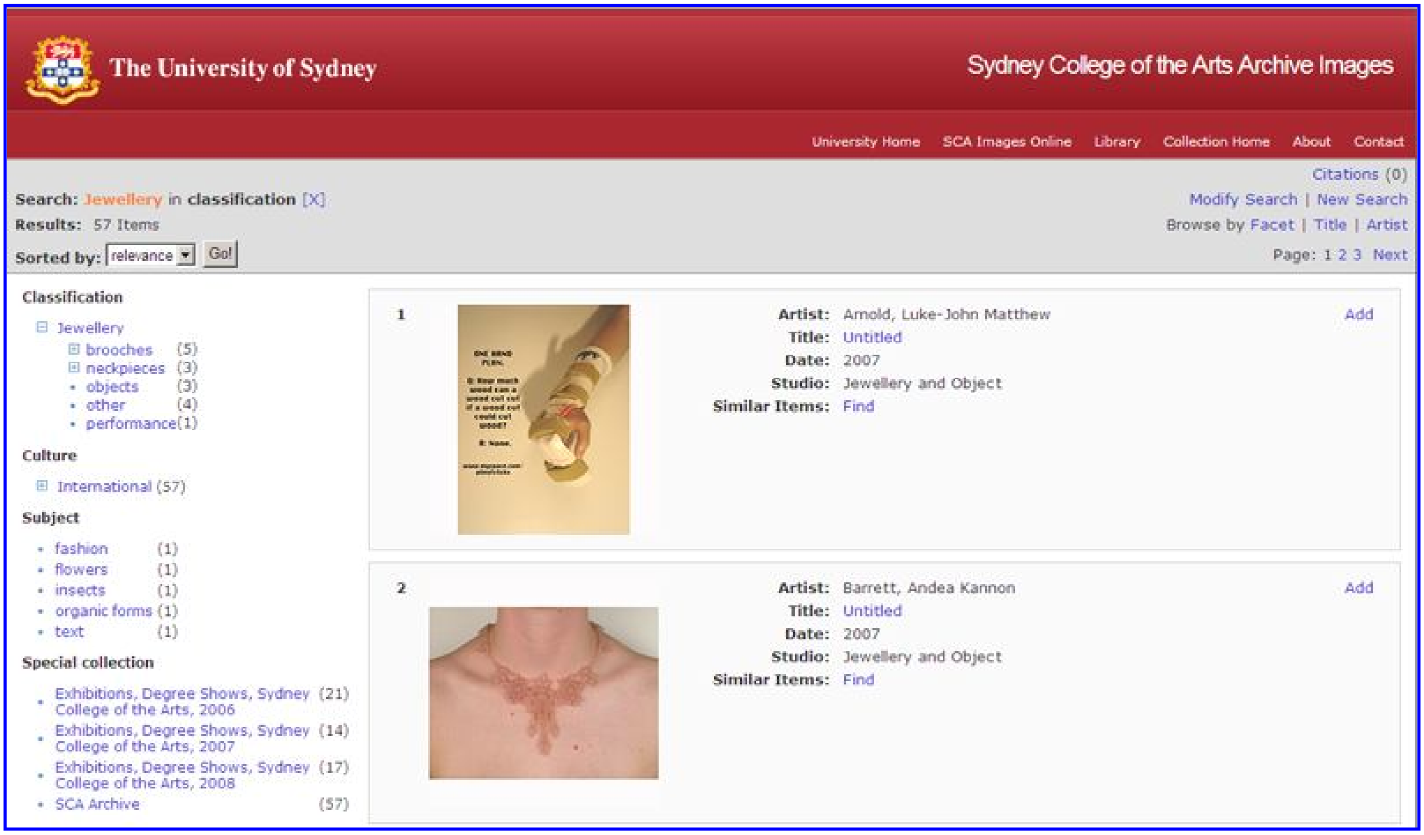
- Painting (2)
- Printmedia (1)
- Vase Shapes (6)
- animals (1)
- boats (7)

more

1
Artist: Guy, Jan 1963-
Title: Figure
Date: 1987
Media: Coloured stains, under glaze, gold leaf on ceramic (hand-built)
Studio: Ceramics
Similar Items: Find
Add

2
Artist: Abraham, Heidi
Title: Untitled
Date: 2007
Studio: Sculpture, Performance and Installation
Similar Items: Find
Add

First level


The University of Sydney
Sydney College of the Arts Archive Images
University Home SCA Images Online Library Collection Home About Contact
Citations (0)
Modify Search | New Search
Browse by Facet | Title | Artist
Go!
Page: 1 2 3 Next

Search: Jewellery in classification [X]
Results: 57 Items
Sorted by: relevance

Classification

- Jewellery
  - brooches (5)
  - neckpieces (3)
  - objects (3)
  - other (4)
  - performance (1)

Culture

- International (57)

Subject

- fashion (1)
- flowers (1)
- insects (1)
- organic forms (1)
- text (1)

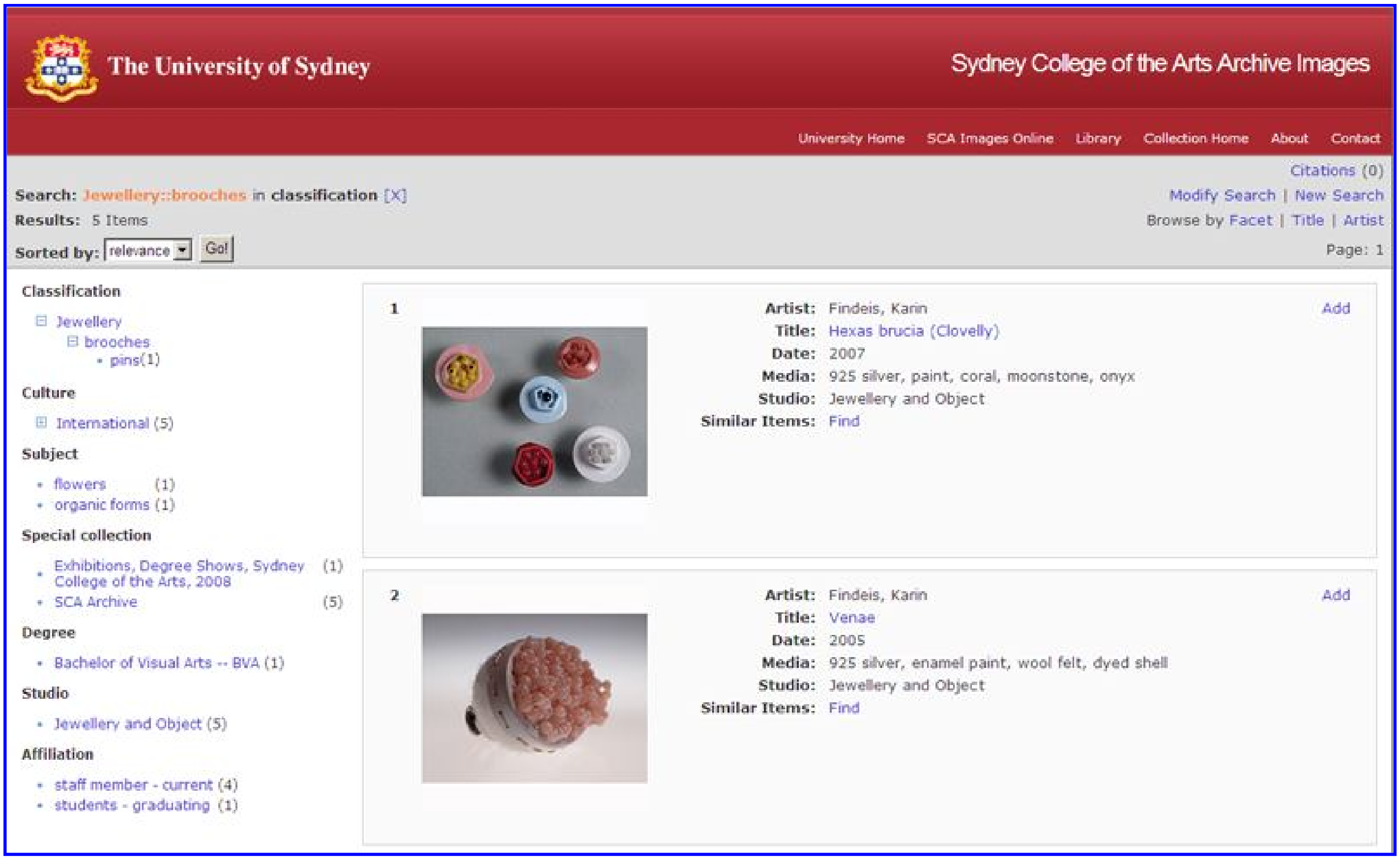
Special collection

- Exhibitions, Degree Shows, Sydney College of the Arts, 2006 (21)
- Exhibitions, Degree Shows, Sydney College of the Arts, 2007 (14)
- Exhibitions, Degree Shows, Sydney College of the Arts, 2008 (17)
- SCA Archive (57)

1
Artist: Arnold, Luke-John Matthew
Title: Untitled
Date: 2007
Studio: Jewellery and Object
Similar Items: Find
Add

2
Artist: Barrett, Andea Kannon
Title: Untitled
Date: 2007
Studio: Jewellery and Object
Similar Items: Find
Add

Second level


The University of Sydney
Sydney College of the Arts Archive Images
University Home | SCA Images Online | Library | Collection Home | About | Contact
Citations (0)
Modify Search | New Search
Browse by Facet | Title | Artist
relevance
Go!
Add

Search: Jewellery::brooches in **classification** [X]
Results: 5 Items
Sorted by: relevance
Page: 1

Classification
- Jewellery
  - brooches
    - pins(1)

Culture
- International (5)

Subject
- flowers (1)
- organic forms (1)

Special collection
- Exhibitions, Degree Shows, Sydney College of the Arts, 2008 (1)
- SCA Archive (5)

Degree
- Bachelor of Visual Arts -- BVA (1)

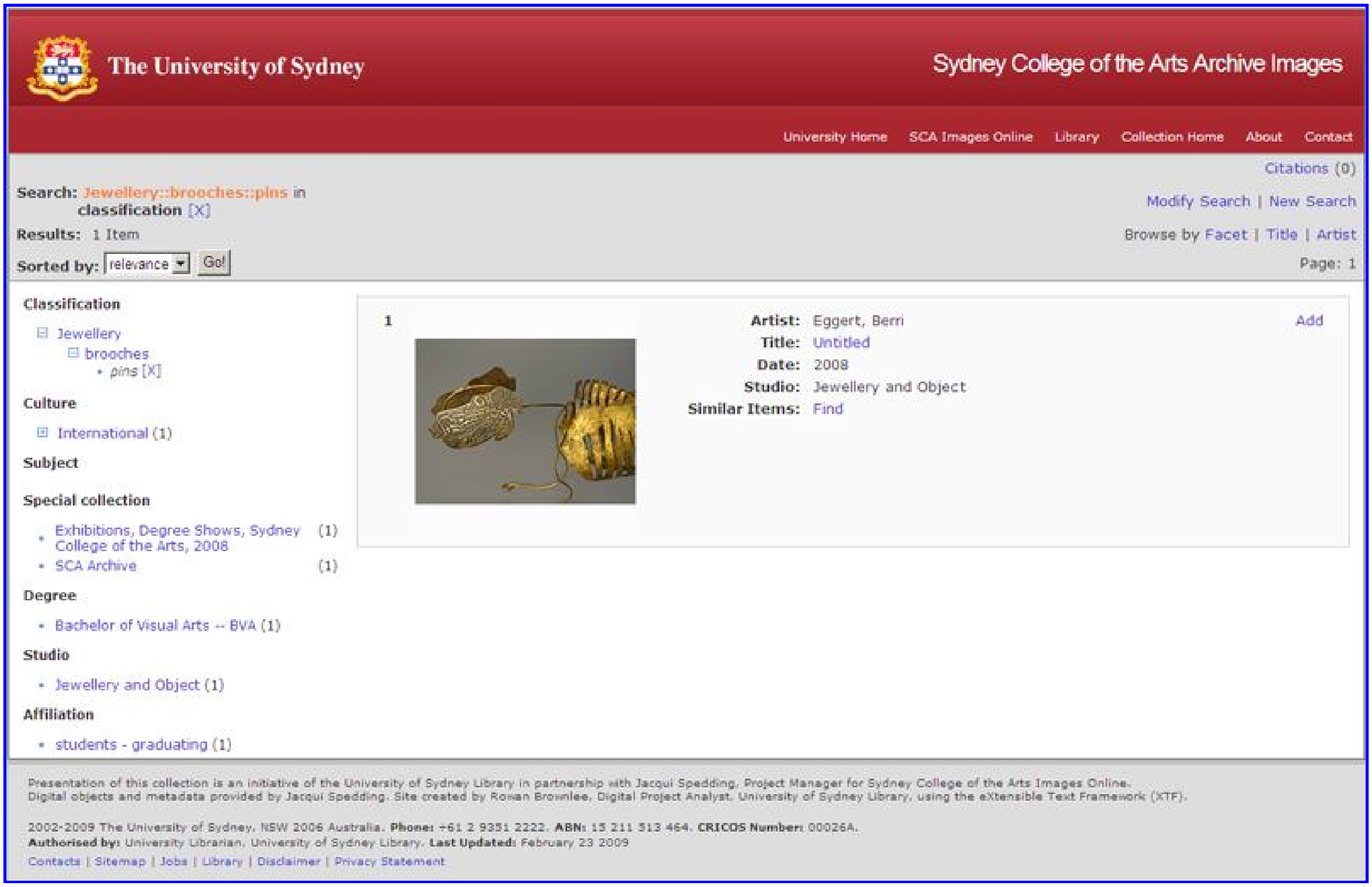
Studio
- Jewellery and Object (5)

Affiliation
- staff member - current (4)
- students - graduating (1)

1
Artist: Findeis, Karin
Title: Hexas brucia (Clovelly)
Date: 2007
Media: 925 silver, paint, coral, moonstone, onyx
Studio: Jewellery and Object
Similar Items: Find

2
Artist: Findeis, Karin
Title: Venae
Date: 2005
Media: 925 silver, enamel paint, wool felt, dyed shell
Studio: Jewellery and Object
Similar Items: Find

Third level


The University of Sydney
Sydney College of the Arts Archive Images
University Home | SCA Images Online | Library | Collection Home | About | Contact
Citations (0)
Modify Search | New Search
Browse by Facet | Title | Artist
relevance
Go!
Add
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Search: Jewellery::brooches::pins in **classification** [X]
Results: 1 Item
Sorted by: relevance
Page: 1

Classification
- Jewellery
  - brooches
    - *pins* [X]

Culture
- International (1)

Subject

Special collection
- Exhibitions, Degree Shows, Sydney College of the Arts, 2008 (1)
- SCA Archive (1)

Degree
- Bachelor of Visual Arts -- BVA (1)

Studio
- Jewellery and Object (1)

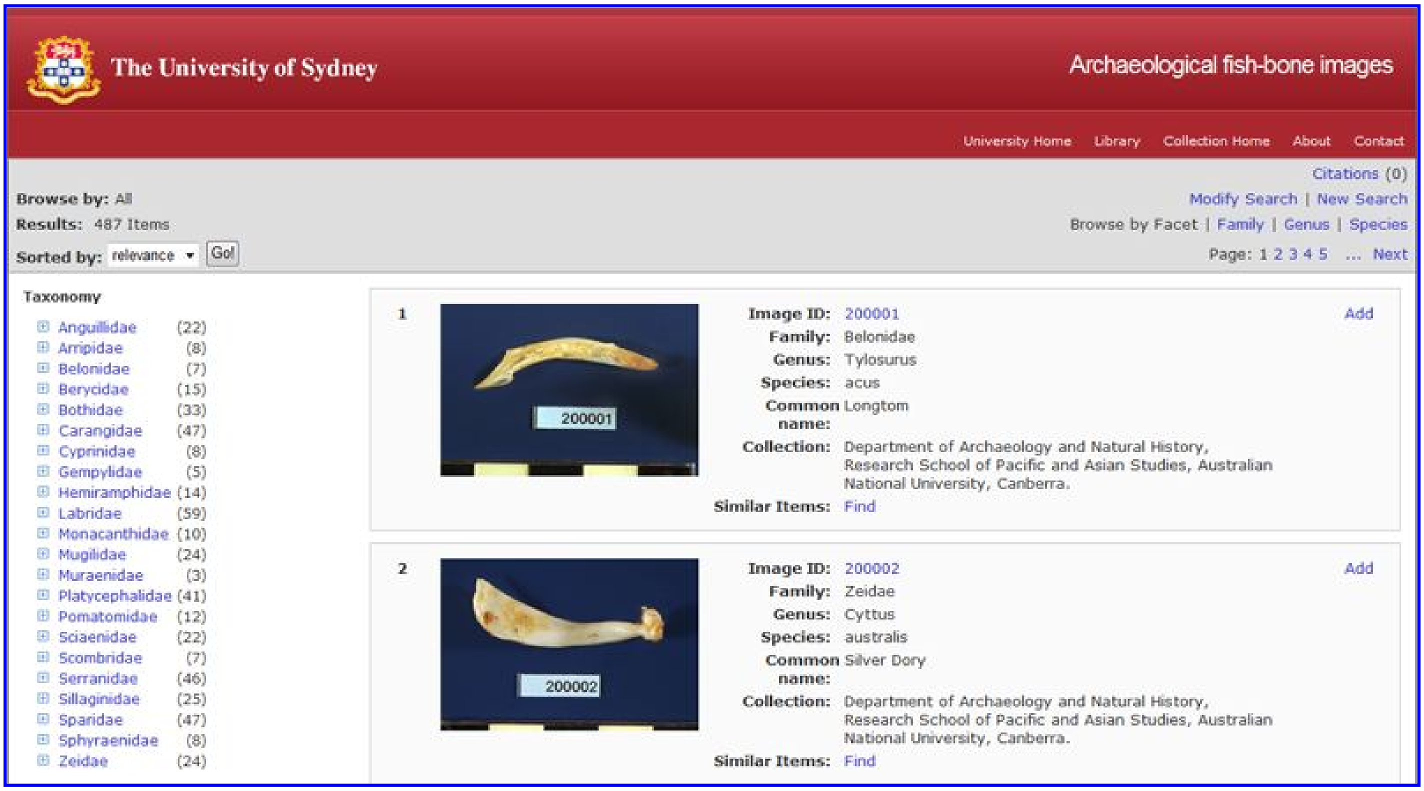
Affiliation
- students - graduating (1)

1
Artist: Eggert, Berri
Title: Untitled
Date: 2008
Studio: Jewellery and Object
Similar Items: Find

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009

Screenshot: Fish-bone image collection, top-level taxonomy facet



Browse by: All
Results: 487 Items
Sorted by: relevance

Browse by Facet | Family | Genus | Species

Page: 1 2 3 4 5 ... Next

Taxonomy

⊞ Anguillidae	(22)
⊞ Arripidae	(8)
⊞ Belonidae	(7)
⊞ Berycidae	(15)
⊞ Bothidae	(33)
⊞ Carangidae	(47)
⊞ Cyprinidae	(8)
⊞ Gempylidae	(5)
⊞ Hemiramphidae	(14)
⊞ Labridae	(59)
⊞ Monacanthidae	(10)
⊞ Mugilidae	(24)
⊞ Muraenidae	(3)
⊞ Platycephalidae	(41)
⊞ Pomatomidae	(12)
⊞ Sciaenidae	(22)
⊞ Scombridae	(7)
⊞ Serranidae	(46)
⊞ Sillaginidae	(25)
⊞ Sparidae	(47)
⊞ Sphyraenidae	(8)
⊞ Zeidae	(24)

1

200001

Image ID: 200001
Family: Belonidae
Genus: Tylosurus
Species: acus
Common name: Longtom
Collection: Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Similar Items: Find

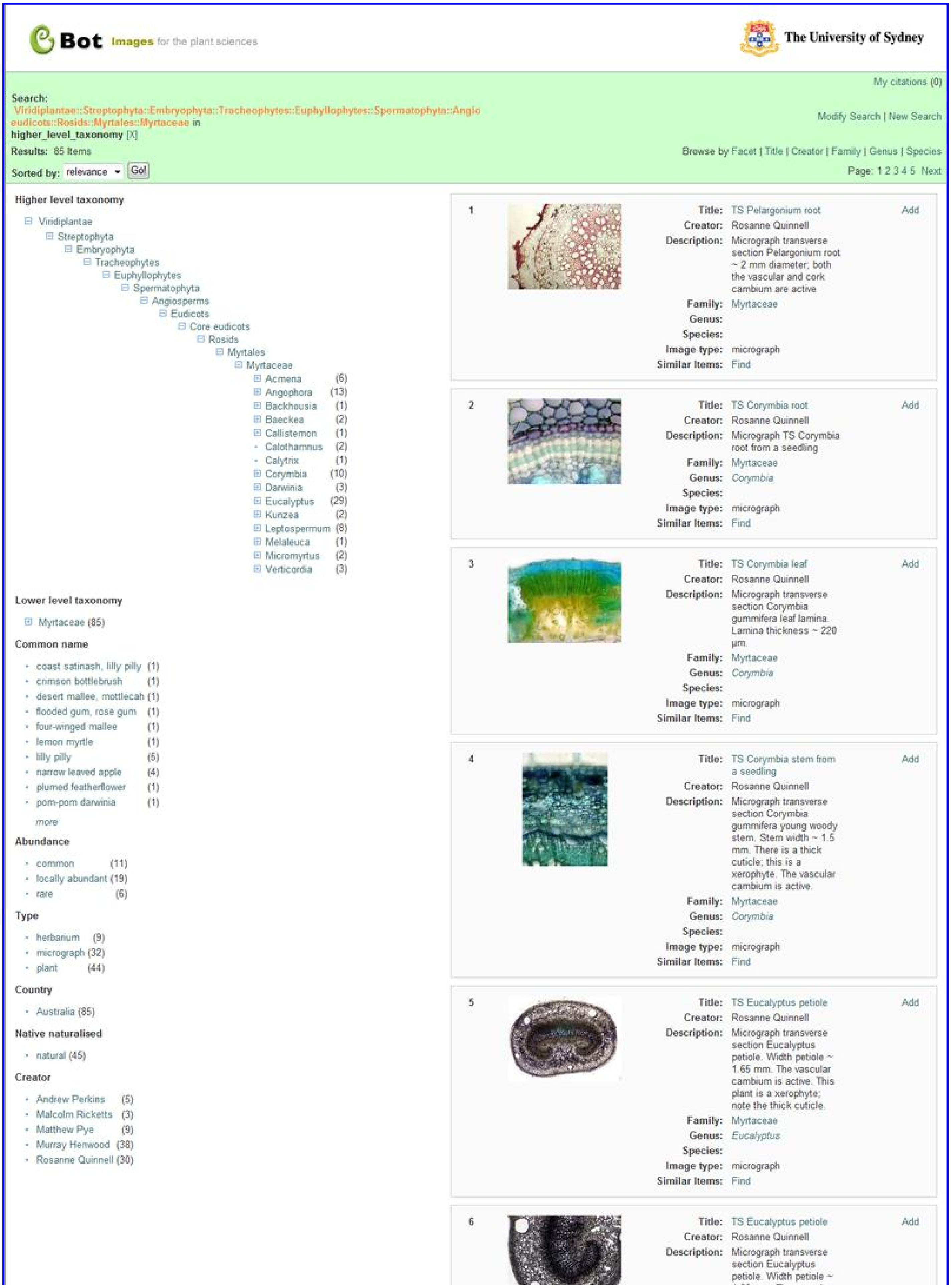
2

200002

Image ID: 200002
Family: Zeidae
Genus: Cyttus
Species: australis
Common name: Silver Dory
Collection: Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Similar Items: Find

Screenshot: eBot plant sciences collection, expanded taxonomy pathway

Example illustrating *partial* navigation of the higher level taxonomy of the eBot plant sciences collection.


eBot Images for the plant sciences
The University of Sydney
My citations (0)
Modify Search | New Search
Browse by Facet | Title | Creator | Family | Genus | Species
Page: 1 2 3 4 5 Next
Sorted by: relevance Go!

Search: Viridiplantae::Streptophyta::Embryophyta::Tracheophytes::Euphyllophytes::Spermatophyta::Angio eudicots::Rosids::Myrtales::Myrtaceae in higher_level_taxonomy [X]

Results: 85 Items

Higher level taxonomy

- ⊟ Viridiplantae
  - ⊟ Streptophyta
    - ⊟ Embryophyta
      - ⊟ Tracheophytes
        - ⊟ Euphyllophytes
          - ⊟ Spermatophyta
            - ⊟ Angiosperms
              - ⊟ Eudicots
                - ⊟ Core eudicots
                  - ⊟ Rosids
                    - ⊟ Myrtales
                      - ⊟ Myrtaceae
                        - ⊞ Acmena (6)
                        - ⊞ Angophora (13)
                        - ⊞ Backhousia (1)
                        - ⊞ Baeckea (2)
                        - ⊞ Callistemon (1)
                        - Calothamnus (2)
                        - Calytrix (1)
                        - ⊞ Corymbia (10)
                        - ⊞ Darwinia (3)
                        - ⊞ Eucalyptus (29)
                        - ⊞ Kunzea (2)
                        - ⊞ Leptospermum (8)
                        - ⊞ Melaleuca (1)
                        - ⊞ Micromyrtus (2)
                        - ⊞ Verticordia (3)

Lower level taxonomy

- ⊞ Myrtaceae (85)

Common name

- coast satinash, lilly pilly (1)
- crimson bottlebrush (1)
- desert mallee, mottlecah (1)
- flooded gum, rose gum (1)
- four-winged mallee (1)
- lemon myrtle (1)
- lilly pilly (5)
- narrow leaved apple (4)
- plumed featherflower (1)
- pom-pom darwinia (1)

more

Abundance

- common (11)
- locally abundant (19)
- rare (6)

Type

- herbarium (9)
- micrograph (32)
- plant (44)

Country

- Australia (85)

Native naturalised

- natural (45)

Creator

- Andrew Perkins (5)
- Malcolm Ricketts (3)
- Matthew Pye (9)
- Murray Henwood (38)
- Rosanne Quinnell (30)

1

Title: TS Pelargonium root — Add
Creator: Rosanne Quinnell
Description: Micrograph transverse section Pelargonium root ~ 2 mm diameter; both the vascular and cork cambium are active
Family: Myrtaceae
Genus:
Species:
Image type: micrograph
Similar Items: Find

2

Title: TS Corymbia root — Add
Creator: Rosanne Quinnell
Description: Micrograph TS Corymbia root from a seedling
Family: Myrtaceae
Genus: *Corymbia*
Species:
Image type: micrograph
Similar Items: Find

3

Title: TS Corymbia leaf — Add
Creator: Rosanne Quinnell
Description: Micrograph transverse section Corymbia gummifera leaf lamina. Lamina thickness ~ 220 µm.
Family: Myrtaceae
Genus: *Corymbia*
Species:
Image type: micrograph
Similar Items: Find

4

Title: TS Corymbia stem from a seedling — Add
Creator: Rosanne Quinnell
Description: Micrograph transverse section Corymbia gummifera young woody stem. Stem width ~ 1.5 mm. There is a thick cuticle; this is a xerophyte. The vascular cambium is active.
Family: Myrtaceae
Genus: *Corymbia*
Species:
Image type: micrograph
Similar Items: Find

5

Title: TS Eucalyptus petiole — Add
Creator: Rosanne Quinnell
Description: Micrograph transverse section Eucalyptus petiole. Width petiole ~ 1.65 mm. The vascular cambium is active. This plant is a xerophyte; note the thick cuticle.
Family: Myrtaceae
Genus: *Eucalyptus*
Species:
Image type: micrograph
Similar Items: Find

6

Title: TS Eucalyptus petiole — Add
Creator: Rosanne Quinnell
Description: Micrograph transverse section Eucalyptus petiole. Width petiole ~

Next… Queries and search forms

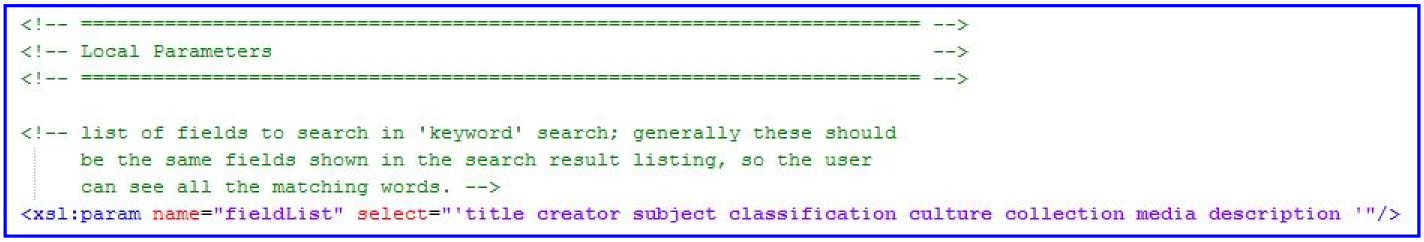
Queries and search forms

- Defining a simple keyword search
- Defining search forms
- Altering example search text to reflect SCA subject matter
- Screenshot: Modified search page

Defining a simple keyword search

style/crossQuery/queryParser/default/queryParser.xsl

XTF needs to know which fields to target for a simple keyword search. Field names are specified within a parameter.



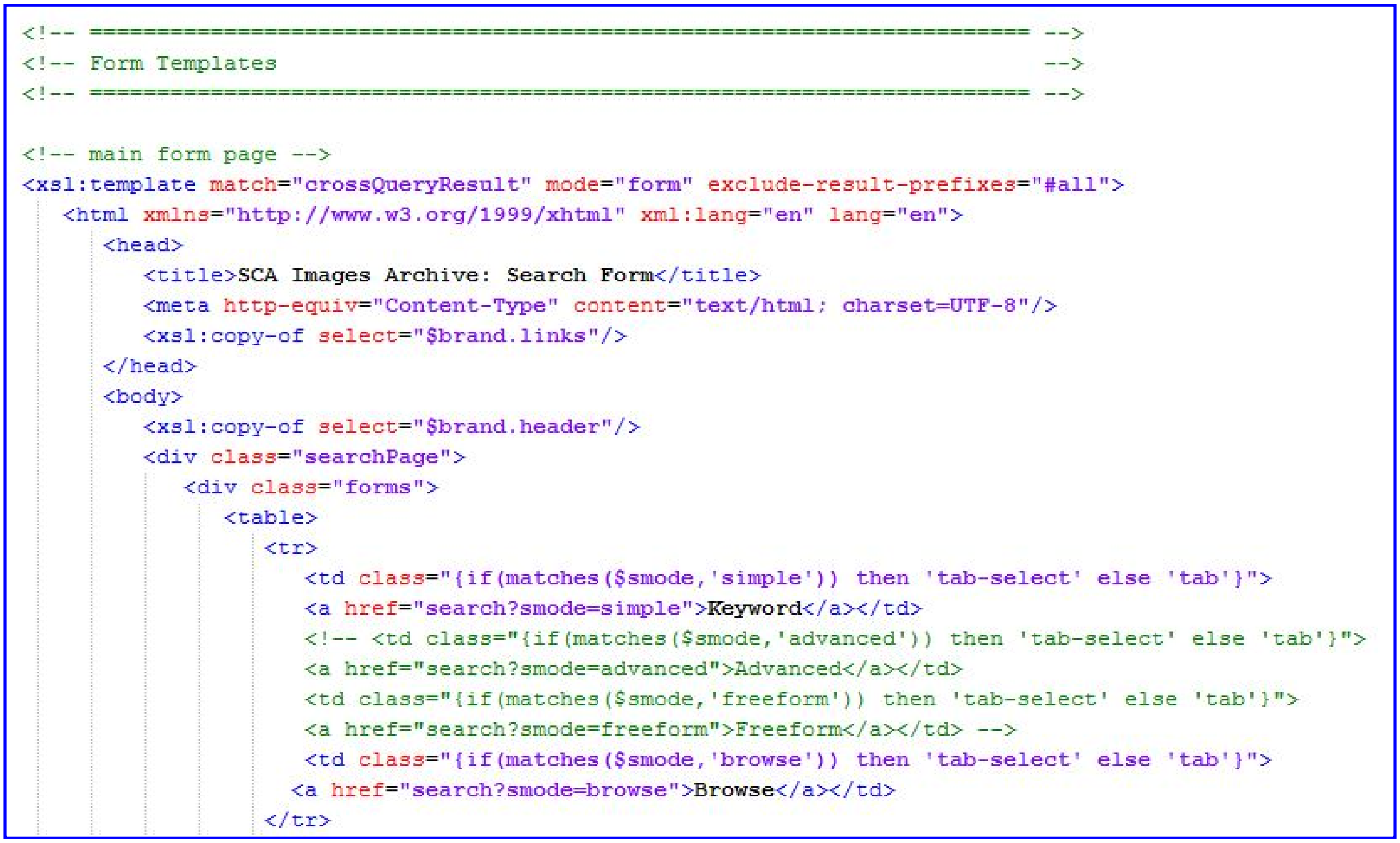
```
<!-- ====================================================================== -->
<!-- Local Parameters                                                       -->
<!-- ====================================================================== -->

<!-- list of fields to search in 'keyword' search; generally these should
     be the same fields shown in the search result listing, so the user
     can see all the matching words. -->
<xsl:param name="fieldList" select="'title creator subject classification culture collection media description '"/>
```

Defining search forms

style/crossQuery/resultFormatter/default/searchForms.xsl

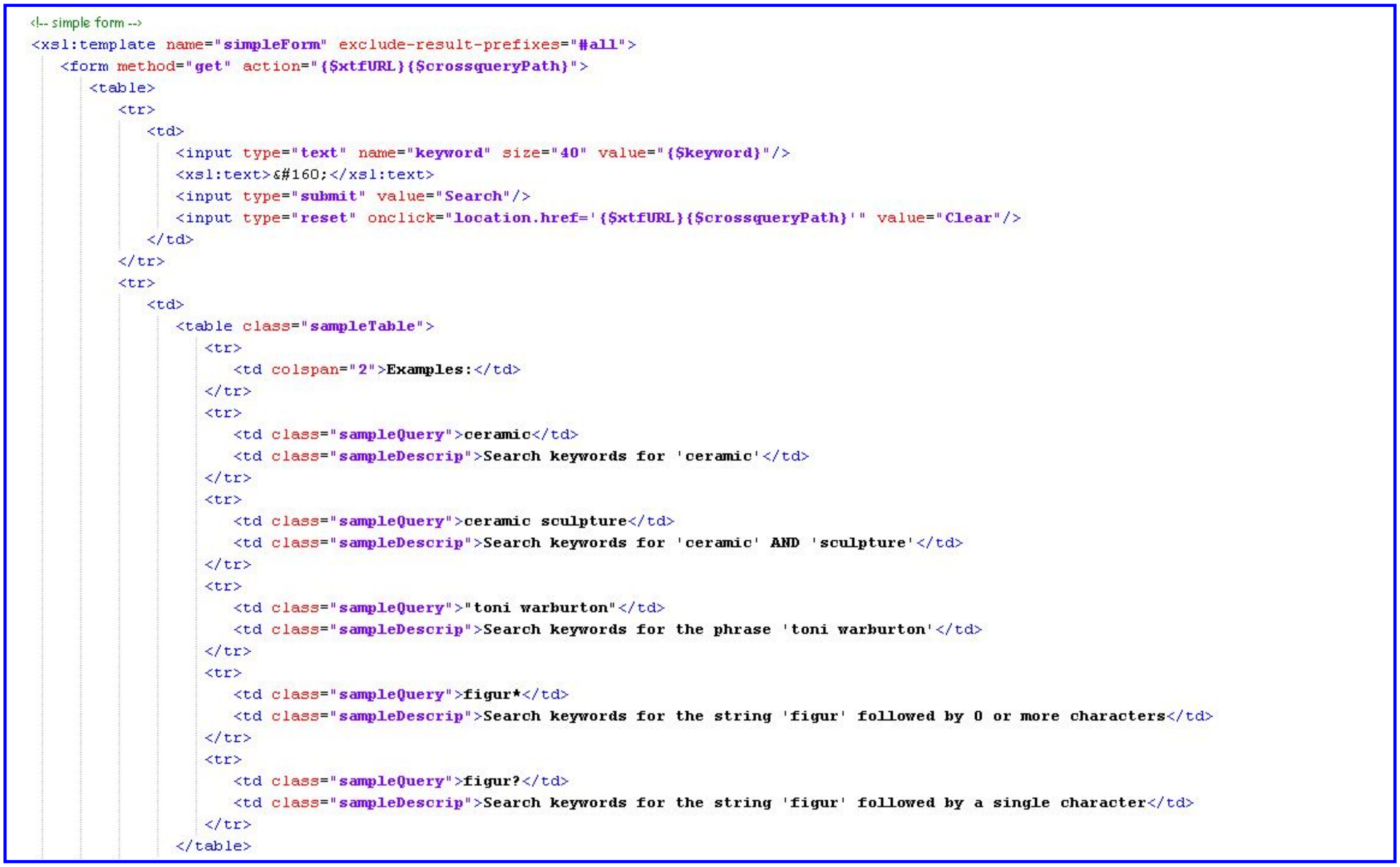
XTF's default user interface includes keyword, advanced, freeform and browse options. As Jacqui and I have not yet designed an advanced search, I remove it from display, along with the freeform option.

Search options for advanced and freeform are commented out.



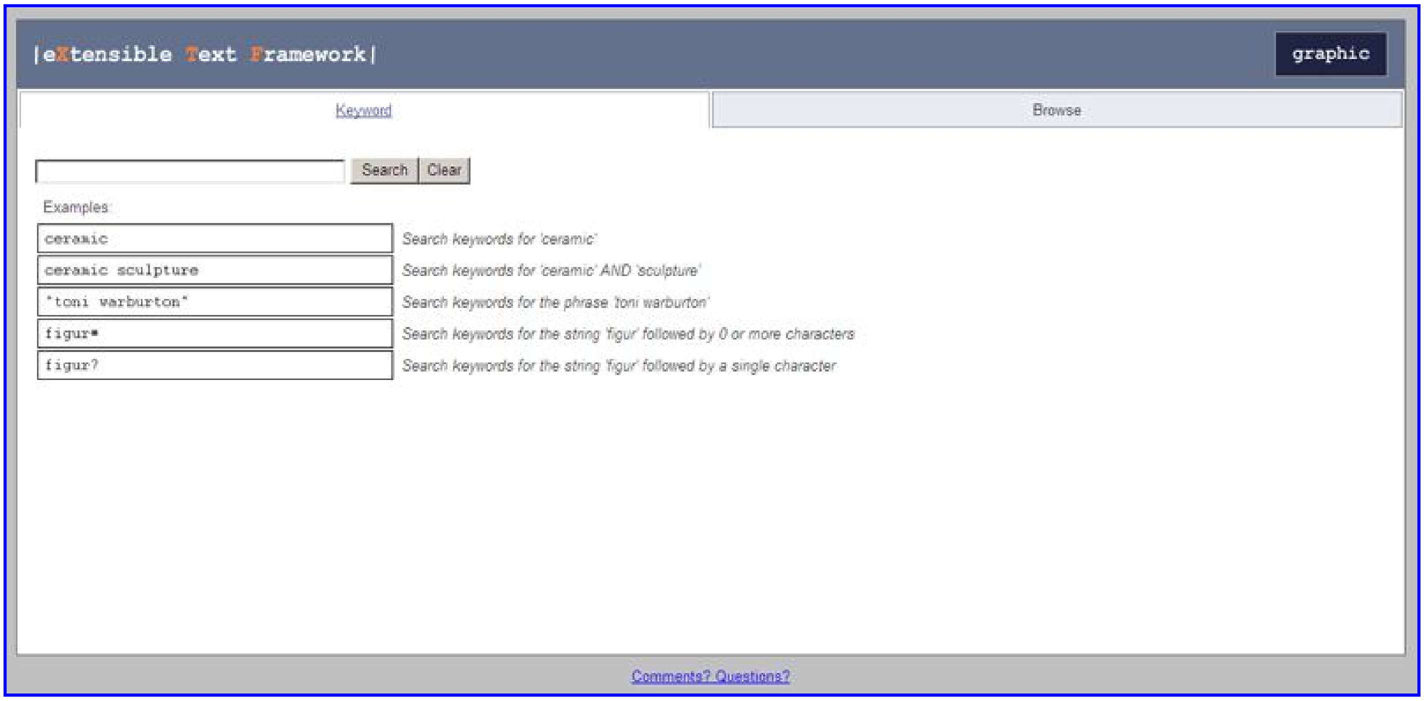
```
<!-- ====================================================================== -->
<!-- Form Templates                                                         -->
<!-- ====================================================================== -->

<!-- main form page -->
<xsl:template match="crossQueryResult" mode="form" exclude-result-prefixes="#all">
   <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
      <head>
         <title>SCA Images Archive: Search Form</title>
         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
         <xsl:copy-of select="$brand.links"/>
      </head>
      <body>
         <xsl:copy-of select="$brand.header"/>
         <div class="searchPage">
            <div class="forms">
               <table>
                  <tr>
                     <td class="{if(matches($smode,'simple')) then 'tab-select' else 'tab'}">
                     <a href="search?smode=simple">Keyword</a></td>
                     <!-- <td class="{if(matches($smode,'advanced')) then 'tab-select' else 'tab'}">
                     <a href="search?smode=advanced">Advanced</a></td>
                     <td class="{if(matches($smode,'freeform')) then 'tab-select' else 'tab'}">
                     <a href="search?smode=freeform">Freeform</a></td> -->
                     <td class="{if(matches($smode,'browse')) then 'tab-select' else 'tab'}">
                    <a href="search?smode=browse">Browse</a></td>
                  </tr>
```

Altering example search text to reflect SCA subject matter



```
<!-- simple form -->
<xsl:template name="simpleForm" exclude-result-prefixes="#all">
   <form method="get" action="{$xtfURL}{$crossqueryPath}">
      <table>
         <tr>
            <td>
               <input type="text" name="keyword" size="40" value="{$keyword}"/>
               <xsl:text>&#160;</xsl:text>
               <input type="submit" value="Search"/>
               <input type="reset" onclick="location.href='{$xtfURL}{$crossqueryPath}'" value="Clear"/>
            </td>
         </tr>
         <tr>
            <td>
               <table class="sampleTable">
                  <tr>
                     <td colspan="2">Examples:</td>
                  </tr>
                  <tr>
                     <td class="sampleQuery">ceramic</td>
                     <td class="sampleDescrip">Search keywords for 'ceramic'</td>
                  </tr>
                  <tr>
                     <td class="sampleQuery">ceramic sculpture</td>
                     <td class="sampleDescrip">Search keywords for 'ceramic' AND 'sculpture'</td>
                  </tr>
                  <tr>
                     <td class="sampleQuery">"toni warburton"</td>
                     <td class="sampleDescrip">Search keywords for the phrase 'toni warburton'</td>
                  </tr>
                  <tr>
                     <td class="sampleQuery">figur*</td>
                     <td class="sampleDescrip">Search keywords for the string 'figur' followed by 0 or more characters</td>
                  </tr>
                  <tr>
                     <td class="sampleQuery">figur?</td>
                     <td class="sampleDescrip">Search keywords for the string 'figur' followed by a single character</td>
                  </tr>
               </table>
```

Screenshot: Modified search page



Examples:

ceramic	Search keywords for 'ceramic'
ceramic sculpture	Search keywords for 'ceramic' AND 'sculpture'
"toni warburton"	Search keywords for the phrase 'toni warburton'
figur*	Search keywords for the string 'figur' followed by 0 or more characters
figur?	Search keywords for the string 'figur' followed by a single character

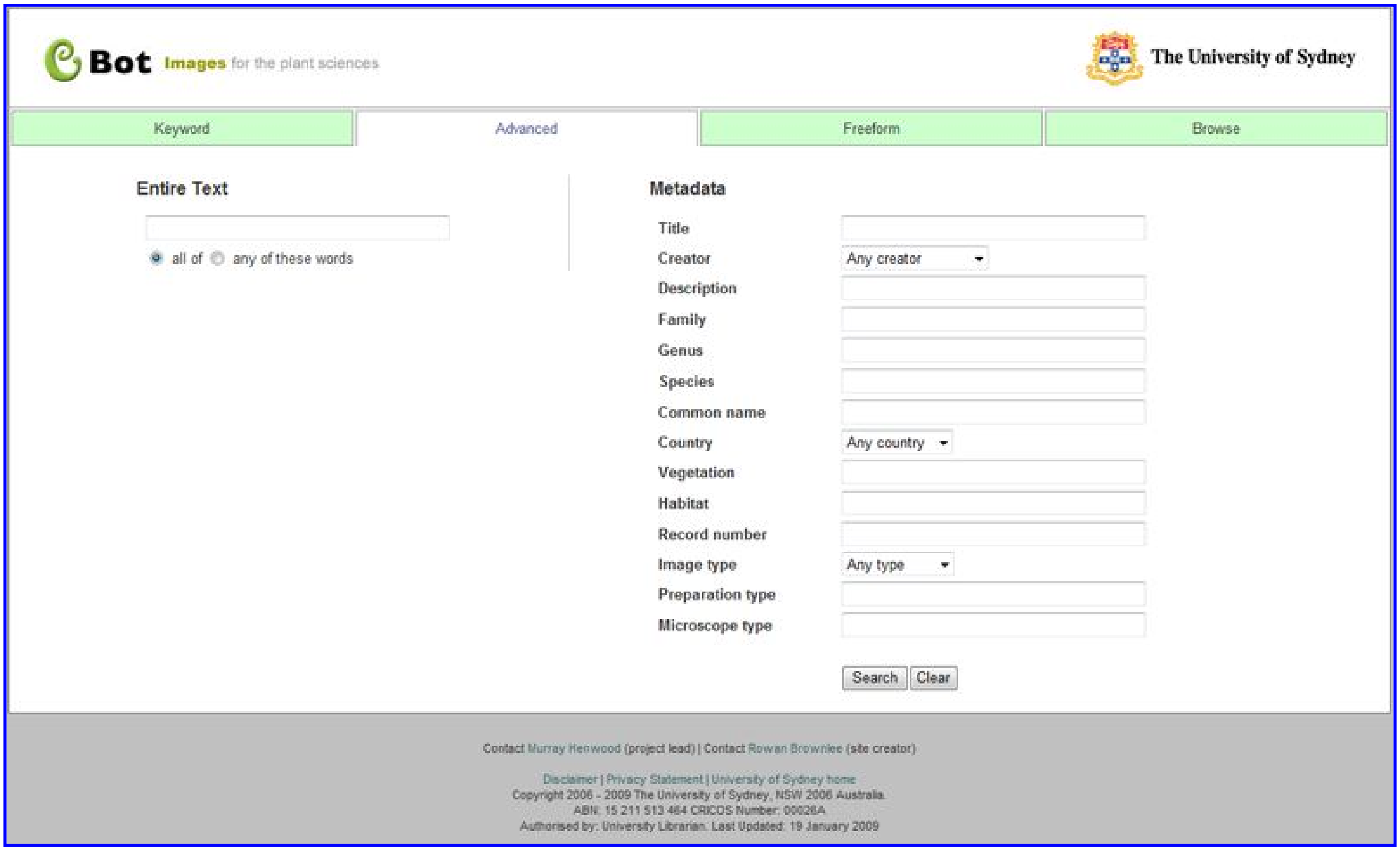
Next… Advanced search

Advanced search

style/crossQuery/resultFormatter/default/searchForms.xsl

- Input fields
- Dropdown lists
- Making dropdowns sticky
- Future enhancements

As mentioned in the previous section, Jacqui and I have not yet designed an advanced search for the SCA collection, so this section instead describes the advanced search page for the eBot plant sciences collection.

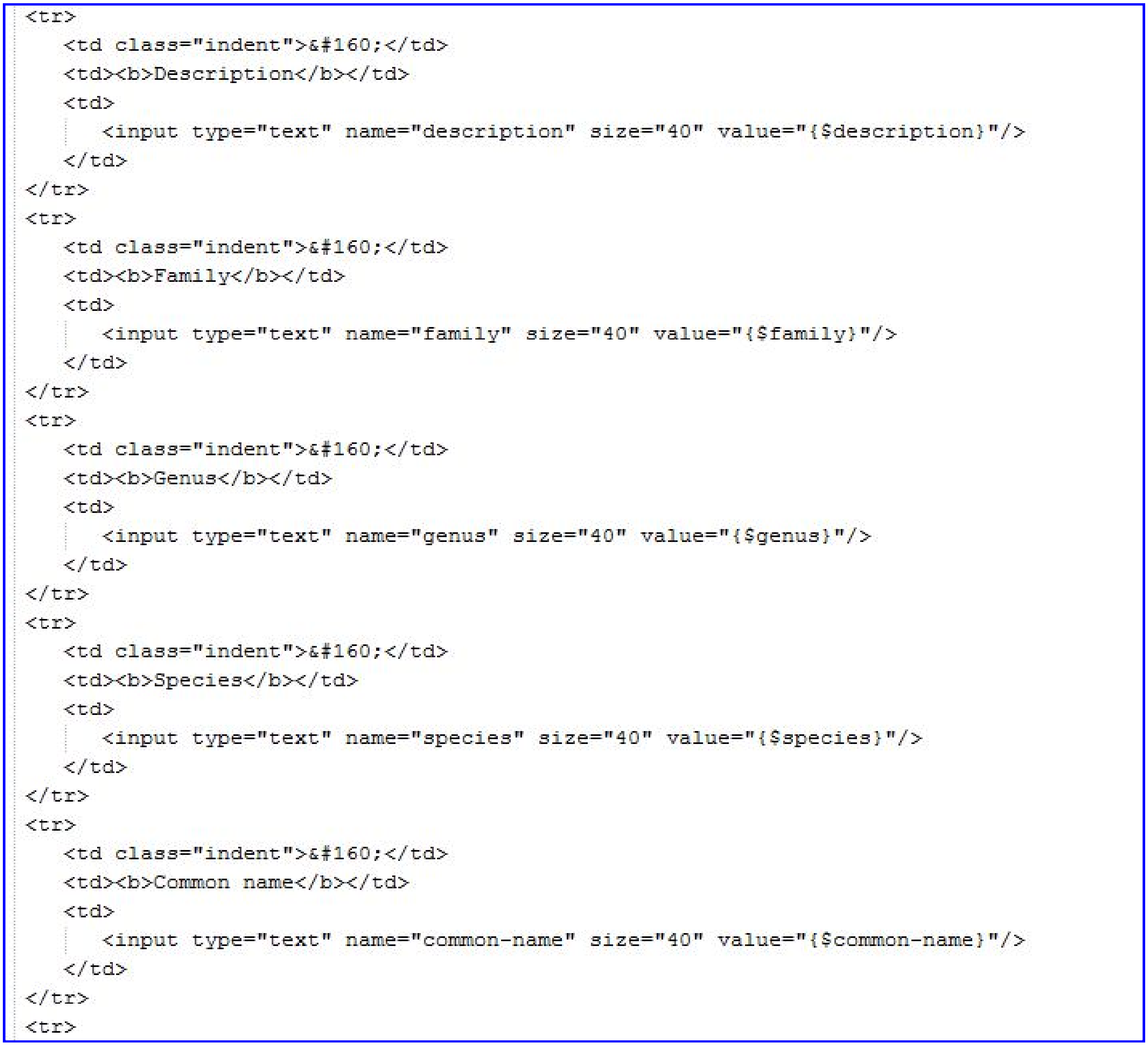


Contact Murray Henwood (project lead) | Contact Rowan Brownlee (site creator)

Disclaimer | Privacy Statement | University of Sydney home
Copyright 2006 - 2009 The University of Sydney, NSW 2006 Australia.
ABN: 15 211 513 464 CRICOS Number: 00026A
Authorised by: University Librarian. Last Updated: 19 January 2009

Input fields

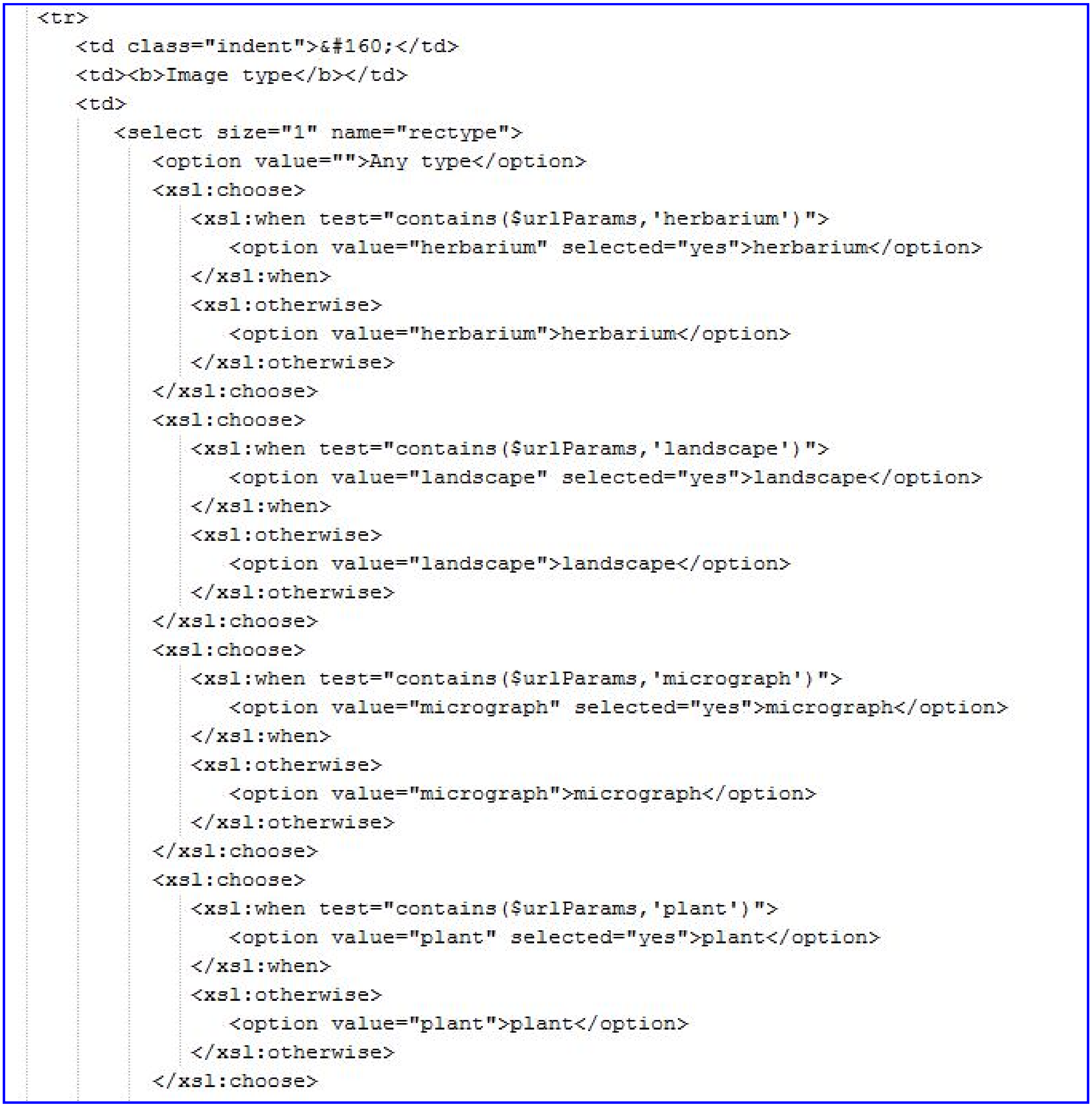
Apart from removing from display some of the default search options, I setup input fields to enable targeted search across particular metadata elements. The example below illustrates description, family, genus, species and common-name fields. In each case *value="{$abc}"* refers to the name of the target metadata element. Metadata element names are defined in an indexing preFilter. (For the SCA collection, *scaPrefilter.xsl* is described in an earlier section.)



```
<tr>
   <td class="indent">&#160;</td>
   <td><b>Description</b></td>
   <td>
      <input type="text" name="description" size="40" value="{$description}"/>
   </td>
</tr>
<tr>
   <td class="indent">&#160;</td>
   <td><b>Family</b></td>
   <td>
      <input type="text" name="family" size="40" value="{$family}"/>
   </td>
</tr>
<tr>
   <td class="indent">&#160;</td>
   <td><b>Genus</b></td>
   <td>
      <input type="text" name="genus" size="40" value="{$genus}"/>
   </td>
</tr>
<tr>
   <td class="indent">&#160;</td>
   <td><b>Species</b></td>
   <td>
      <input type="text" name="species" size="40" value="{$species}"/>
   </td>
</tr>
<tr>
   <td class="indent">&#160;</td>
   <td><b>Common name</b></td>
   <td>
      <input type="text" name="common-name" size="40" value="{$common-name}"/>
   </td>
</tr>
<tr>
```

Dropdown lists

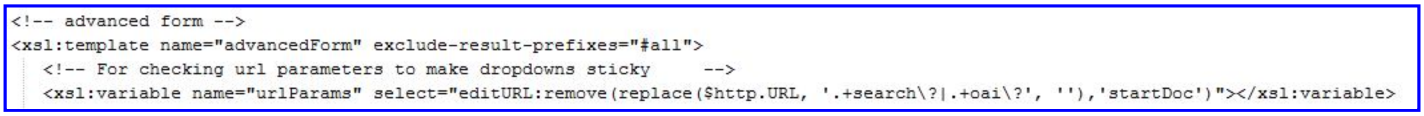
The eBot advanced search form uses several dropdown lists. The following example is an extract from *image type*. eBot contains a number of types including *herbarium*, *plant* and *micrograph*.



```
<tr>
   <td class="indent">&#160;</td>
   <td><b>Image type</b></td>
   <td>
      <select size="1" name="rectype">
         <option value="">Any type</option>
         <xsl:choose>
            <xsl:when test="contains($urlParams,'herbarium')">
               <option value="herbarium" selected="yes">herbarium</option>
            </xsl:when>
            <xsl:otherwise>
               <option value="herbarium">herbarium</option>
            </xsl:otherwise>
         </xsl:choose>
         <xsl:choose>
            <xsl:when test="contains($urlParams,'landscape')">
               <option value="landscape" selected="yes">landscape</option>
            </xsl:when>
            <xsl:otherwise>
               <option value="landscape">landscape</option>
            </xsl:otherwise>
         </xsl:choose>
         <xsl:choose>
            <xsl:when test="contains($urlParams,'micrograph')">
               <option value="micrograph" selected="yes">micrograph</option>
            </xsl:when>
            <xsl:otherwise>
               <option value="micrograph">micrograph</option>
            </xsl:otherwise>
         </xsl:choose>
         <xsl:choose>
            <xsl:when test="contains($urlParams,'plant')">
               <option value="plant" selected="yes">plant</option>
            </xsl:when>
            <xsl:otherwise>
               <option value="plant">plant</option>
            </xsl:otherwise>
         </xsl:choose>
```

Making dropdown selections sticky

By default, dropdown list selections are not sticky. If I choose an option from a dropdown list, perform a search and click 'modify search' from the results page to return to the advanced search page my previously selected dropdown choice will not display. Instead the default option at the head of the dropdown list displays. Selections can be made sticky by including an instruction provided elsewhere by XTF.



```
<!-- advanced form -->
<xsl:template name="advancedForm" exclude-result-prefixes="#all">
   <!-- For checking url parameters to make dropdowns sticky     -->
   <xsl:variable name="urlParams" select="editURL:remove(replace($http.URL, '.+search\?|.+oai\?', ''),'startDoc')"></xsl:variable>
```

Future enhancements

The current advanced search is quite simple and could be developed to be far more extensive and flexible. I’d like to provide dropdown lists or automatic completion for fields such as family, genus and species. These classification elements are already related to each other through an extensive taxonomy underpinning the eBot collection, and the relationships might be exploited to provide additional support for searching. As an example, if a user wishes to search on a particular species, selection from the species category should trigger auto-population of the the family and genus fields. It would also be useful to enable the boolean ‘OR’ within categories such as *image type*. This would enable multiple selections from the list, to suit cases where people are interested in results comprising several image types such as *herbarium OR micrograph*.

Next… Styling the site

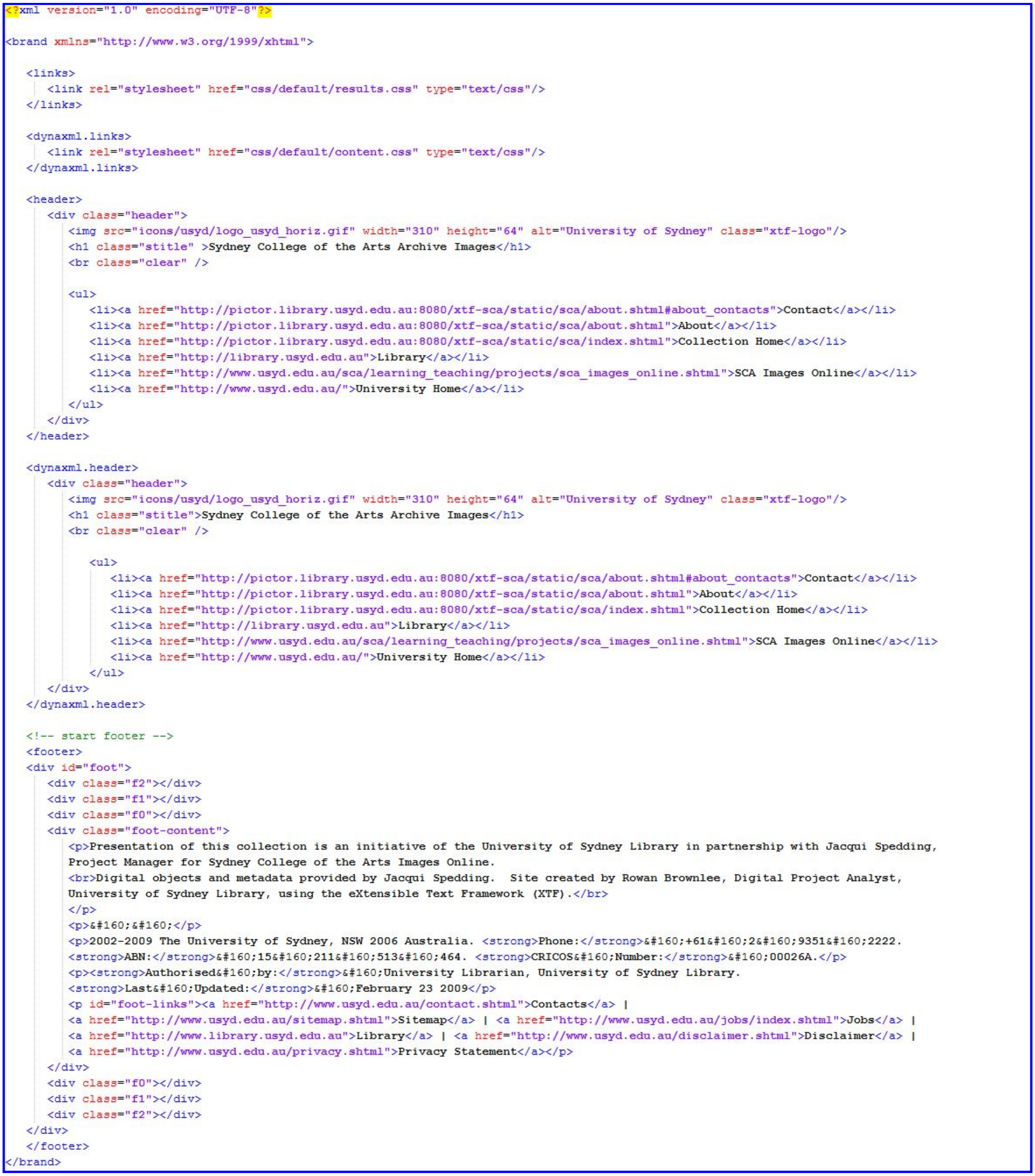
Styling the site

brand/default.xml

- Modified header and footer
- Changes to css
- Referencing header, footer and css within templates
- Screenshot: Styled search page
- Screenshot: Styled browse-all by facet page
- Screenshot: Styled full record page

Modified header and footer

Header and footer information is contained in *brand/default.xml* which I alter to provide pointers to styling elements more reflective of the University of Sydney corporate style.



```xml
<?xml version="1.0" encoding="UTF-8"?>

<brand xmlns="http://www.w3.org/1999/xhtml">

   <links>
      <link rel="stylesheet" href="css/default/results.css" type="text/css"/>
   </links>

   <dynaxml.links>
      <link rel="stylesheet" href="css/default/content.css" type="text/css"/>
   </dynaxml.links>

   <header>
      <div class="header">
         <img src="icons/usyd/logo_usyd_horiz.gif" width="310" height="64" alt="University of Sydney" class="xtf-logo"/>
         <h1 class="stitle" >Sydney College of the Arts Archive Images</h1>
         <br class="clear" />

         <ul>
            <li><a href="http://pictor.library.usyd.edu.au:8080/xtf-sca/static/sca/about.shtml#about_contacts">Contact</a></li>
            <li><a href="http://pictor.library.usyd.edu.au:8080/xtf-sca/static/sca/about.shtml">About</a></li>
            <li><a href="http://pictor.library.usyd.edu.au:8080/xtf-sca/static/sca/index.shtml">Collection Home</a></li>
            <li><a href="http://library.usyd.edu.au">Library</a></li>
            <li><a href="http://www.usyd.edu.au/sca/learning_teaching/projects/sca_images_online.shtml">SCA Images Online</a></li>
            <li><a href="http://www.usyd.edu.au/">University Home</a></li>
         </ul>
      </div>
   </header>

   <dynaxml.header>
      <div class="header">
         <img src="icons/usyd/logo_usyd_horiz.gif" width="310" height="64" alt="University of Sydney" class="xtf-logo"/>
         <h1 class="stitle">Sydney College of the Arts Archive Images</h1>
         <br class="clear" />

            <ul>
               <li><a href="http://pictor.library.usyd.edu.au:8080/xtf-sca/static/sca/about.shtml#about_contacts">Contact</a></li>
               <li><a href="http://pictor.library.usyd.edu.au:8080/xtf-sca/static/sca/about.shtml">About</a></li>
               <li><a href="http://pictor.library.usyd.edu.au:8080/xtf-sca/static/sca/index.shtml">Collection Home</a></li>
               <li><a href="http://library.usyd.edu.au">Library</a></li>
               <li><a href="http://www.usyd.edu.au/sca/learning_teaching/projects/sca_images_online.shtml">SCA Images Online</a></li>
               <li><a href="http://www.usyd.edu.au/">University Home</a></li>
            </ul>
      </div>
   </dynaxml.header>

   <!-- start footer -->
   <footer>
   <div id="foot">
      <div class="f2"></div>
      <div class="f1"></div>
      <div class="f0"></div>
      <div class="foot-content">
         <p>Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding,
         Project Manager for Sydney College of the Arts Images Online.
         <br>Digital objects and metadata provided by Jacqui Spedding.  Site created by Rowan Brownlee, Digital Project Analyst,
         University of Sydney Library, using the eXtensible Text Framework (XTF).</br>
         </p>
         <p>&#160;&#160;</p>
         <p>2002-2009 The University of Sydney, NSW 2006 Australia. <strong>Phone:</strong>&#160;+61&#160;2&#160;9351&#160;2222.
         <strong>ABN:</strong>&#160;15&#160;211&#160;513&#160;464. <strong>CRICOS&#160;Number:</strong>&#160;00026A.</p>
         <p><strong>Authorised&#160;by:</strong>&#160;University Librarian, University of Sydney Library.
         <strong>Last&#160;Updated:</strong>&#160;February 23 2009</p>
         <p id="foot-links"><a href="http://www.usyd.edu.au/contact.shtml">Contacts</a> |
         <a href="http://www.usyd.edu.au/sitemap.shtml">Sitemap</a> | <a href="http://www.usyd.edu.au/jobs/index.shtml">Jobs</a> |
         <a href="http://www.library.usyd.edu.au">Library</a> | <a href="http://www.usyd.edu.au/disclaimer.shtml">Disclaimer</a> |
         <a href="http://www.usyd.edu.au/privacy.shtml">Privacy Statement</a></p>
      </div>
      <div class="f0"></div>
      <div class="f1"></div>
      <div class="f2"></div>
   </div>
   </footer>
</brand>
```

Changes to css

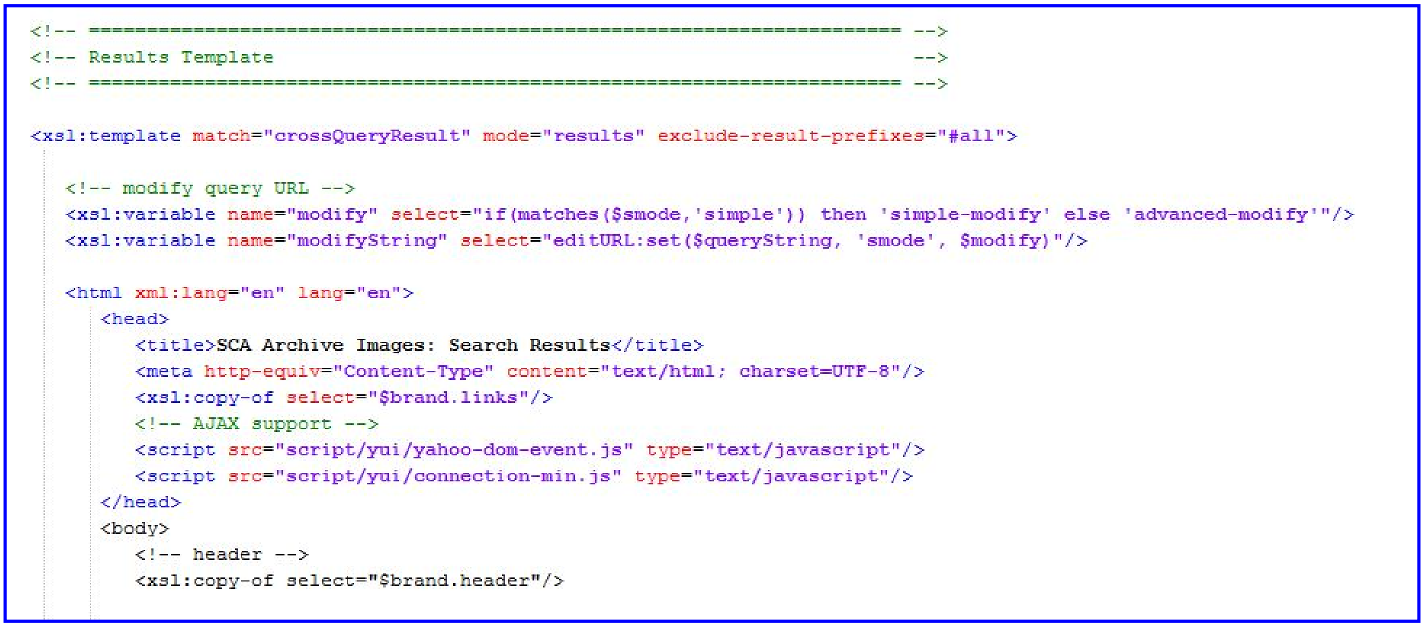
As css changes are specific to the University of Sydney context, I haven't included details.

Referencing header, footer and css within templates

Because css, header and footer information is contained within *brand/default.xml*, it is sufficient to provide references within templates, in the following format.

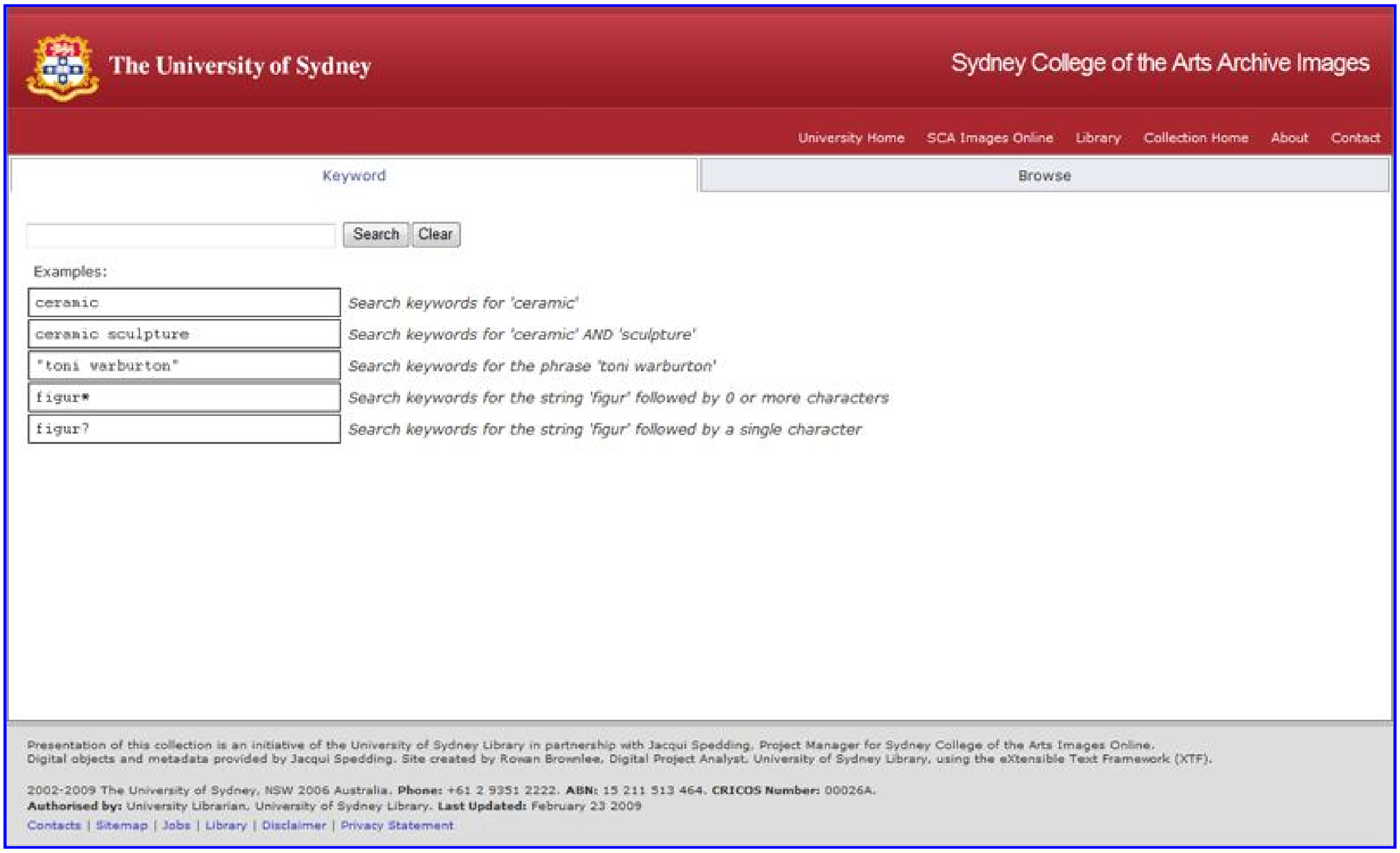
```
<xsl:copy-of select="$brand.links"/>
<xsl:copy-of select="$brand.header"/>
```

Results template including references to $brand.links (for css) and $brand.header:



```
<!-- ====================================================================== -->
<!-- Results Template                                                       -->
<!-- ====================================================================== -->

<xsl:template match="crossQueryResult" mode="results" exclude-result-prefixes="#all">

   <!-- modify query URL -->
   <xsl:variable name="modify" select="if(matches($smode,'simple')) then 'simple-modify' else 'advanced-modify'"/>
   <xsl:variable name="modifyString" select="editURL:set($queryString, 'smode', $modify)"/>

   <html xml:lang="en" lang="en">
      <head>
         <title>SCA Archive Images: Search Results</title>
         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
         <xsl:copy-of select="$brand.links"/>
         <!-- AJAX support -->
         <script src="script/yui/yahoo-dom-event.js" type="text/javascript"/>
         <script src="script/yui/connection-min.js" type="text/javascript"/>
      </head>
      <body>
         <!-- header -->
         <xsl:copy-of select="$brand.header"/>
```

Screenshot: Styled search page



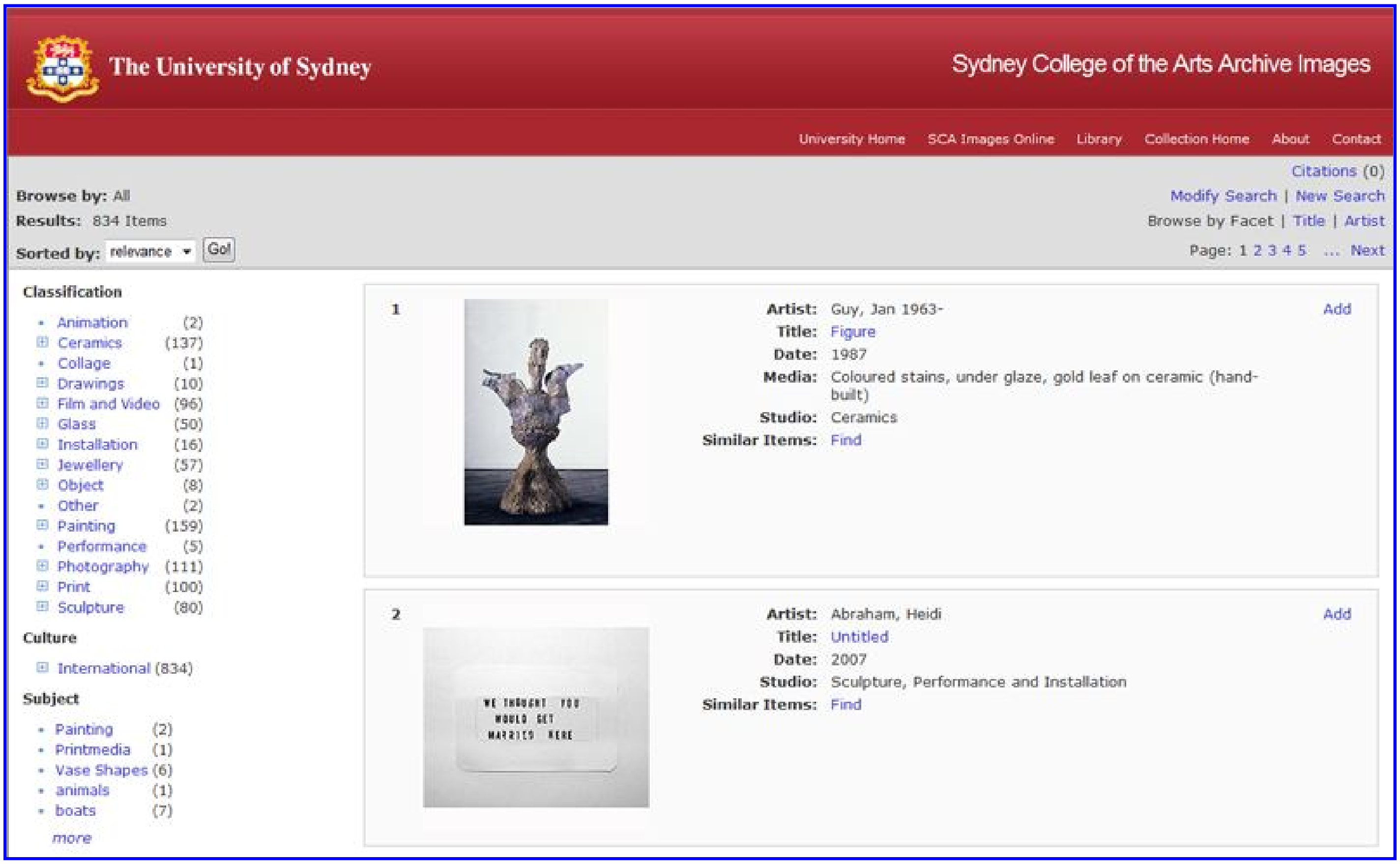
Examples:

ceramic	*Search keywords for 'ceramic'*
ceramic sculpture	*Search keywords for 'ceramic' AND 'sculpture'*
"toni warburton"	*Search keywords for the phrase 'toni warburton'*
figur*	*Search keywords for the string 'figur' followed by 0 or more characters*
figur?	*Search keywords for the string 'figur' followed by a single character*

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Screenshot: Styled browse-all by facet page



Browse by: All
Results: 834 Items
Sorted by: relevance

Classification

Animation	(2)
Ceramics	(137)
Collage	(1)
Drawings	(10)
Film and Video	(96)
Glass	(50)
Installation	(16)
Jewellery	(57)
Object	(8)
Other	(2)
Painting	(159)
Performance	(5)
Photography	(111)
Print	(100)
Sculpture	(80)

Culture

International	(834)

Subject

Painting	(2)
Printmedia	(1)
Vase Shapes	(6)
animals	(1)
boats	(7)

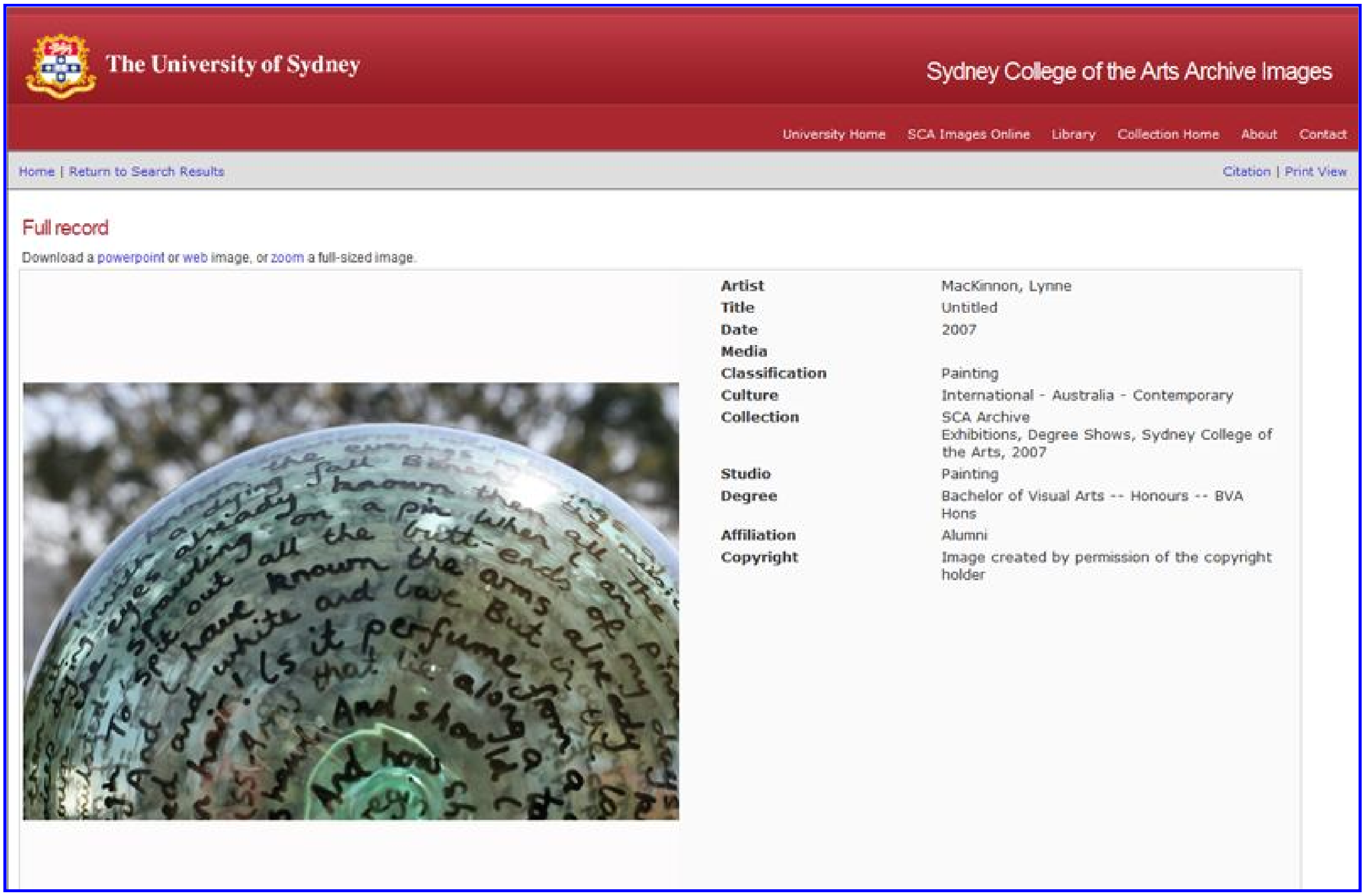
1

Artist: Guy, Jan 1963-
Title: Figure
Date: 1987
Media: Coloured stains, under glaze, gold leaf on ceramic (hand-built)
Studio: Ceramics
Similar Items: Find

2

Artist: Abraham, Heidi
Title: Untitled
Date: 2007
Studio: Sculpture, Performance and Installation
Similar Items: Find

Screenshot: Styled full-record page



Full record

Download a powerpoint or web image, or zoom a full-sized image.

Artist	MacKinnon, Lynne
Title	Untitled
Date	2007
Media	
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Painting
Degree	Bachelor of Visual Arts -- Honours -- BVA Hons
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Next… Displaying error messages

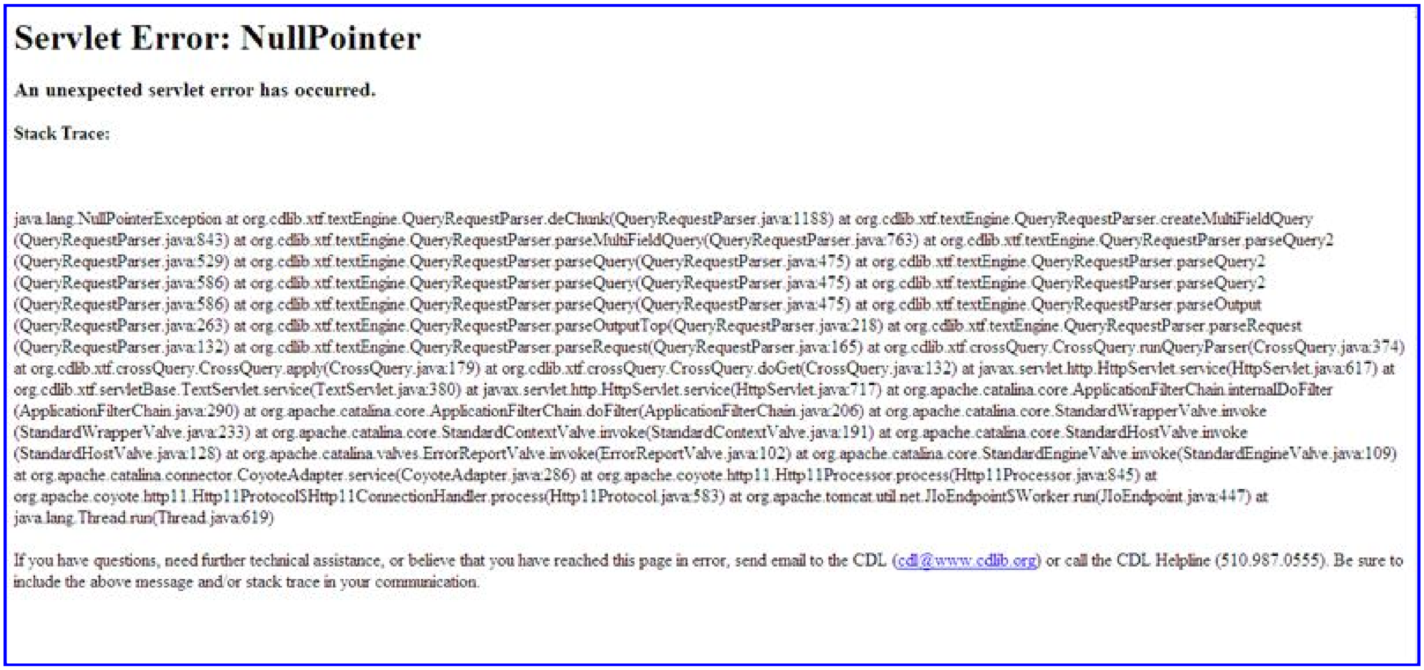
Displaying error messages

Presentation of error messages is handled by errorGen.xsl. There are two files, one each for index errors and query errors. They are located at

```
style\dynaXML\errorGen.xsl
style\crossQuery\errorGen.xsl
```

- Screenshot: Default error display
- Screenshot: Customised presentation of error messages
- Referencing site styling elements
- Modifying the error presentation template

Screenshot: Default error display



Servlet Error: NullPointer

An unexpected servlet error has occurred.

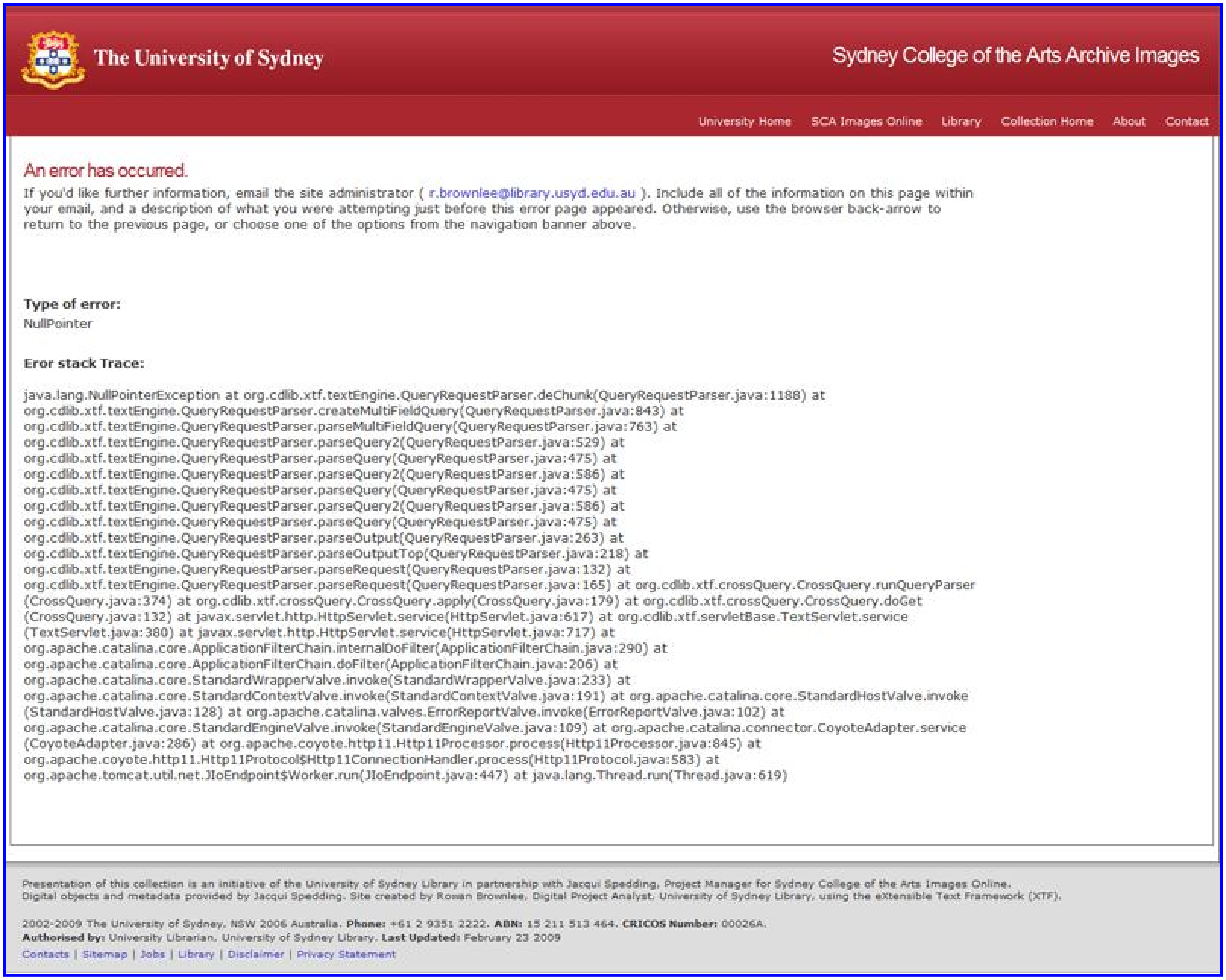
Stack Trace:

```
java.lang.NullPointerException at org.cdlib.xtf.textEngine.QueryRequestParser.deChunk(QueryRequestParser.java:1188) at org.cdlib.xtf.textEngine.QueryRequestParser.createMultiFieldQuery
(QueryRequestParser.java:843) at org.cdlib.xtf.textEngine.QueryRequestParser.parseMultiFieldQuery(QueryRequestParser.java:763) at org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery2
(QueryRequestParser.java:529) at org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery(QueryRequestParser.java:475) at org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery2
(QueryRequestParser.java:586) at org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery(QueryRequestParser.java:475) at org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery2
(QueryRequestParser.java:586) at org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery(QueryRequestParser.java:475) at org.cdlib.xtf.textEngine.QueryRequestParser.parseOutput
(QueryRequestParser.java:263) at org.cdlib.xtf.textEngine.QueryRequestParser.parseOutputTop(QueryRequestParser.java:218) at org.cdlib.xtf.textEngine.QueryRequestParser.parseRequest
(QueryRequestParser.java:132) at org.cdlib.xtf.textEngine.QueryRequestParser.parseRequest(QueryRequestParser.java:165) at org.cdlib.xtf.crossQuery.CrossQuery.runQueryParser(CrossQuery.java:374)
at org.cdlib.xtf.crossQuery.CrossQuery.apply(CrossQuery.java:179) at org.cdlib.xtf.crossQuery.CrossQuery.doGet(CrossQuery.java:132) at javax.servlet.http.HttpServlet.service(HttpServlet.java:617) at
org.cdlib.xtf.servletBase.TextServlet.service(TextServlet.java:380) at javax.servlet.http.HttpServlet.service(HttpServlet.java:717) at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter
(ApplicationFilterChain.java:290) at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206) at org.apache.catalina.core.StandardWrapperValve.invoke
(StandardWrapperValve.java:233) at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:191) at org.apache.catalina.core.StandardHostValve.invoke
(StandardHostValve.java:128) at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:102) at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:109)
at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:286) at org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:845) at
org.apache.coyote.http11.Http11Protocol$Http11ConnectionHandler.process(Http11Protocol.java:583) at org.apache.tomcat.util.net.JIoEndpoint$Worker.run(JIoEndpoint.java:447) at
java.lang.Thread.run(Thread.java:619)
```

If you have questions, need further technical assistance, or believe that you have reached this page in error, send email to the CDL (cdl@www.cdlib.org) or call the CDL Helpline (510.987.0555). Be sure to include the above message and/or stack trace in your communication.

Screenshot: Customised presentation of error messages

This example shows an error message within a navigation model used throughout the site.



An error has occurred.

If you'd like further information, email the site administrator (r.brownlee@library.usyd.edu.au). Include all of the information on this page within your email, and a description of what you were attempting just before this error page appeared. Otherwise, use the browser back-arrow to return to the previous page, or choose one of the options from the navigation banner above.

Type of error:

NullPointer

Eror stack Trace:

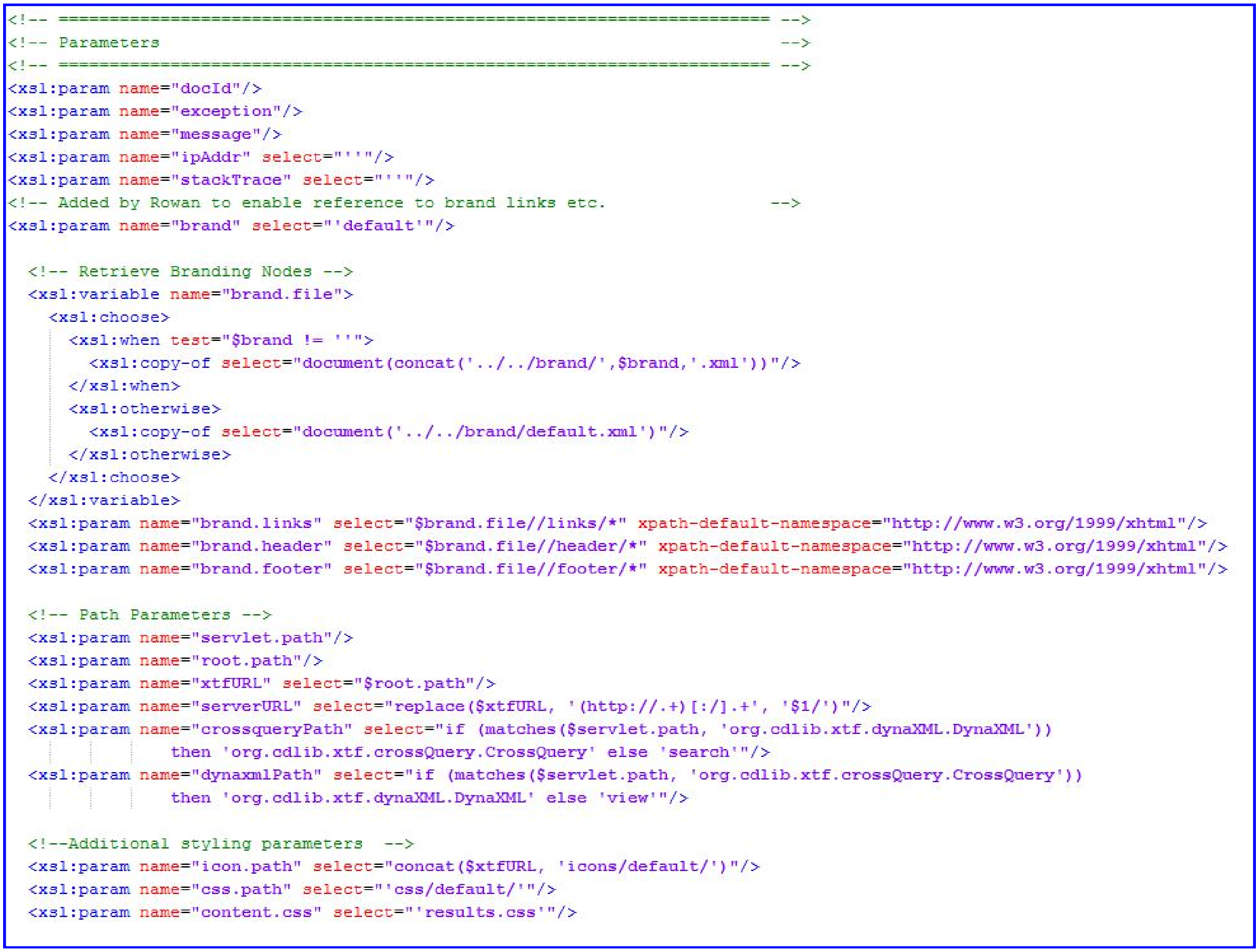
```
java.lang.NullPointerException at org.cdlib.xtf.textEngine.QueryRequestParser.deChunk(QueryRequestParser.java:1188) at
org.cdlib.xtf.textEngine.QueryRequestParser.createMultiFieldQuery(QueryRequestParser.java:843) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseMultiFieldQuery(QueryRequestParser.java:763) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery2(QueryRequestParser.java:529) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery(QueryRequestParser.java:475) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery2(QueryRequestParser.java:586) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery(QueryRequestParser.java:475) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery2(QueryRequestParser.java:586) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseQuery(QueryRequestParser.java:475) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseOutput(QueryRequestParser.java:263) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseOutputTop(QueryRequestParser.java:218) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseRequest(QueryRequestParser.java:132) at
org.cdlib.xtf.textEngine.QueryRequestParser.parseRequest(QueryRequestParser.java:165) at org.cdlib.xtf.crossQuery.CrossQuery.runQueryParser
(CrossQuery.java:374) at org.cdlib.xtf.crossQuery.CrossQuery.apply(CrossQuery.java:179) at org.cdlib.xtf.crossQuery.CrossQuery.doGet
(CrossQuery.java:132) at javax.servlet.http.HttpServlet.service(HttpServlet.java:617) at org.cdlib.xtf.servletBase.TextServlet.service
(TextServlet.java:380) at javax.servlet.http.HttpServlet.service(HttpServlet.java:717) at
org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:290) at
org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206) at
org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:233) at
org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:191) at org.apache.catalina.core.StandardHostValve.invoke
(StandardHostValve.java:128) at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:102) at
org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:109) at org.apache.catalina.connector.CoyoteAdapter.service
(CoyoteAdapter.java:286) at org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:845) at
org.apache.coyote.http11.Http11Protocol$Http11ConnectionHandler.process(Http11Protocol.java:583) at
org.apache.tomcat.util.net.JIoEndpoint$Worker.run(JIoEndpoint.java:447) at java.lang.Thread.run(Thread.java:619)
```

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Referencing site styling elements

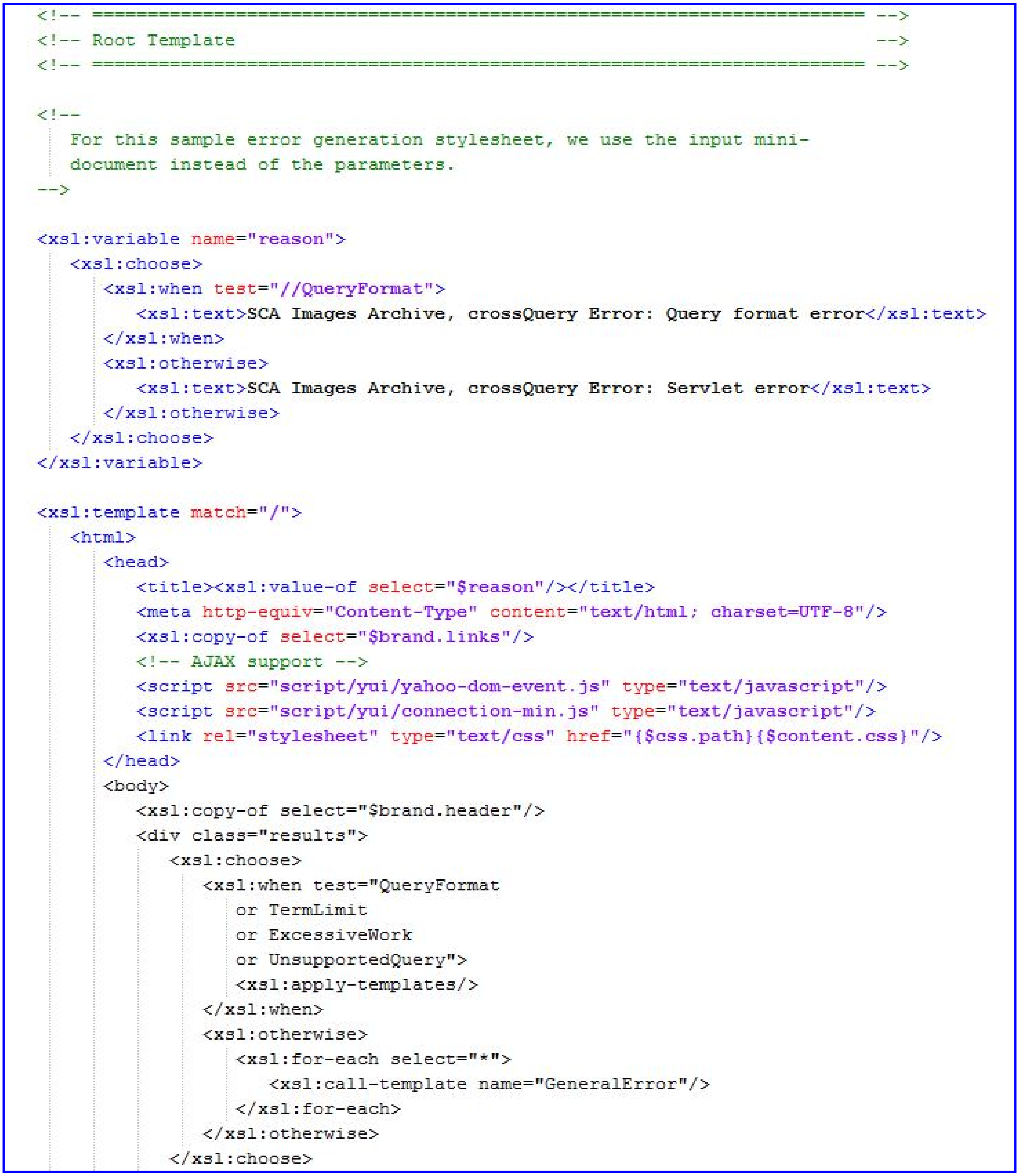
Apart from altering the order of displayed sections and editing contact details, I add references to styling elements.



```
<!-- ===================================================================== -->
<!-- Parameters                                                            -->
<!-- ===================================================================== -->
<xsl:param name="docId"/>
<xsl:param name="exception"/>
<xsl:param name="message"/>
<xsl:param name="ipAddr" select="''"/>
<xsl:param name="stackTrace" select="''"/>
<!-- Added by Rowan to enable reference to brand links etc.              -->
<xsl:param name="brand" select="'default'"/>

  <!-- Retrieve Branding Nodes -->
  <xsl:variable name="brand.file">
    <xsl:choose>
      <xsl:when test="$brand != ''">
        <xsl:copy-of select="document(concat('../../brand/',$brand,'.xml'))"/>
      </xsl:when>
      <xsl:otherwise>
        <xsl:copy-of select="document('../../brand/default.xml')"/>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:param name="brand.links" select="$brand.file//links/*" xpath-default-namespace="http://www.w3.org/1999/xhtml"/>
  <xsl:param name="brand.header" select="$brand.file//header/*" xpath-default-namespace="http://www.w3.org/1999/xhtml"/>
  <xsl:param name="brand.footer" select="$brand.file//footer/*" xpath-default-namespace="http://www.w3.org/1999/xhtml"/>

  <!-- Path Parameters -->
  <xsl:param name="servlet.path"/>
  <xsl:param name="root.path"/>
  <xsl:param name="xtfURL" select="$root.path"/>
  <xsl:param name="serverURL" select="replace($xtfURL, '(http://.+)[:/].+', '$1/')"/>
  <xsl:param name="crossqueryPath" select="if (matches($servlet.path, 'org.cdlib.xtf.dynaXML.DynaXML'))
              then 'org.cdlib.xtf.crossQuery.CrossQuery' else 'search'"/>
  <xsl:param name="dynaxmlPath" select="if (matches($servlet.path, 'org.cdlib.xtf.crossQuery.CrossQuery'))
              then 'org.cdlib.xtf.dynaXML.DynaXML' else 'view'"/>

  <!--Additional styling parameters  -->
  <xsl:param name="icon.path" select="concat($xtfURL, 'icons/default/')"/>
  <xsl:param name="css.path" select="'css/default/'"/>
  <xsl:param name="content.css" select="'results.css'"/>
```

Modifying the error presentation template

Styling elements are referenced within a modified template.



```
<!-- ====================================================================== -->
<!-- Root Template                                                          -->
<!-- ====================================================================== -->

<!--
   For this sample error generation stylesheet, we use the input mini-
   document instead of the parameters.
-->

<xsl:variable name="reason">
   <xsl:choose>
      <xsl:when test="//QueryFormat">
         <xsl:text>SCA Images Archive, crossQuery Error: Query format error</xsl:text>
      </xsl:when>
      <xsl:otherwise>
         <xsl:text>SCA Images Archive, crossQuery Error: Servlet error</xsl:text>
      </xsl:otherwise>
   </xsl:choose>
</xsl:variable>

<xsl:template match="/">
   <html>
      <head>
         <title><xsl:value-of select="$reason"/></title>
         <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
         <xsl:copy-of select="$brand.links"/>
         <!-- AJAX support -->
         <script src="script/yui/yahoo-dom-event.js" type="text/javascript"/>
         <script src="script/yui/connection-min.js" type="text/javascript"/>
         <link rel="stylesheet" type="text/css" href="{$css.path}{$content.css}"/>
      </head>
      <body>
         <xsl:copy-of select="$brand.header"/>
         <div class="results">
            <xsl:choose>
               <xsl:when test="QueryFormat
                  or TermLimit
                  or ExcessiveWork
                  or UnsupportedQuery">
                  <xsl:apply-templates/>
               </xsl:when>
               <xsl:otherwise>
                  <xsl:for-each select="*">
                     <xsl:call-template name="GeneralError"/>
                  </xsl:for-each>
               </xsl:otherwise>
            </xsl:choose>
```

Next… About the collections

About the collections

The University of Sydney Library is using an XTF presentation model to enable online access to several media collections in the fields of plant sciences, visual arts and archaeology. eBot was released for public access in October 2009, and the Archaeology collection followed in November. The SCA collection was launched in February 2010. This initiative results from partnerships between the Library, academics and associated project managers.

- **SCA Archive**
  The SCA Archive contains image and video reproduction of works created by visual arts academics and students at Sydney College of the Arts. The key project partner is Jacqui Spedding, project manager for SCA Images Online. The SCA Archive comprises 1509 items.
  Screenshots | Access the SCA Archive
- **eBot**
  eBot is a collection of plant sciences images representing the work or University of Sydney academics for use in research, learning and teaching programs. Key project partners include Murray Henwood and Rosanne Quinnell (School of Biological Sciences) and Su Hanfling (Library). More information about eBot, from the proceedings of the 2008 Uniserve Conference. eBot comprises 1856 images.

  Screenshots | Access eBot
- **Sarah Colley’s fish-bone image collection**
  Sarah Colley’s archaeological collection contains images of fish-bones and provides support for her research and that of her project partners. One of Sarah’s research interests concerns the archaeological study of fish and fishing in Sydney before and after 1788. The collection comprises 812 images.

  Screenshots | Acess Sarah Colley’s fish-bone image collection

Screenshots and Details for Each Collection

SCA Archive

The SCA Archive contains image and video reproduction of works created by visual arts academics and students at Sydney College of the Arts. The key project partner is Jacqui Spedding, project manager for SCA Images Online. The SCA Archive comprises 1509 items.

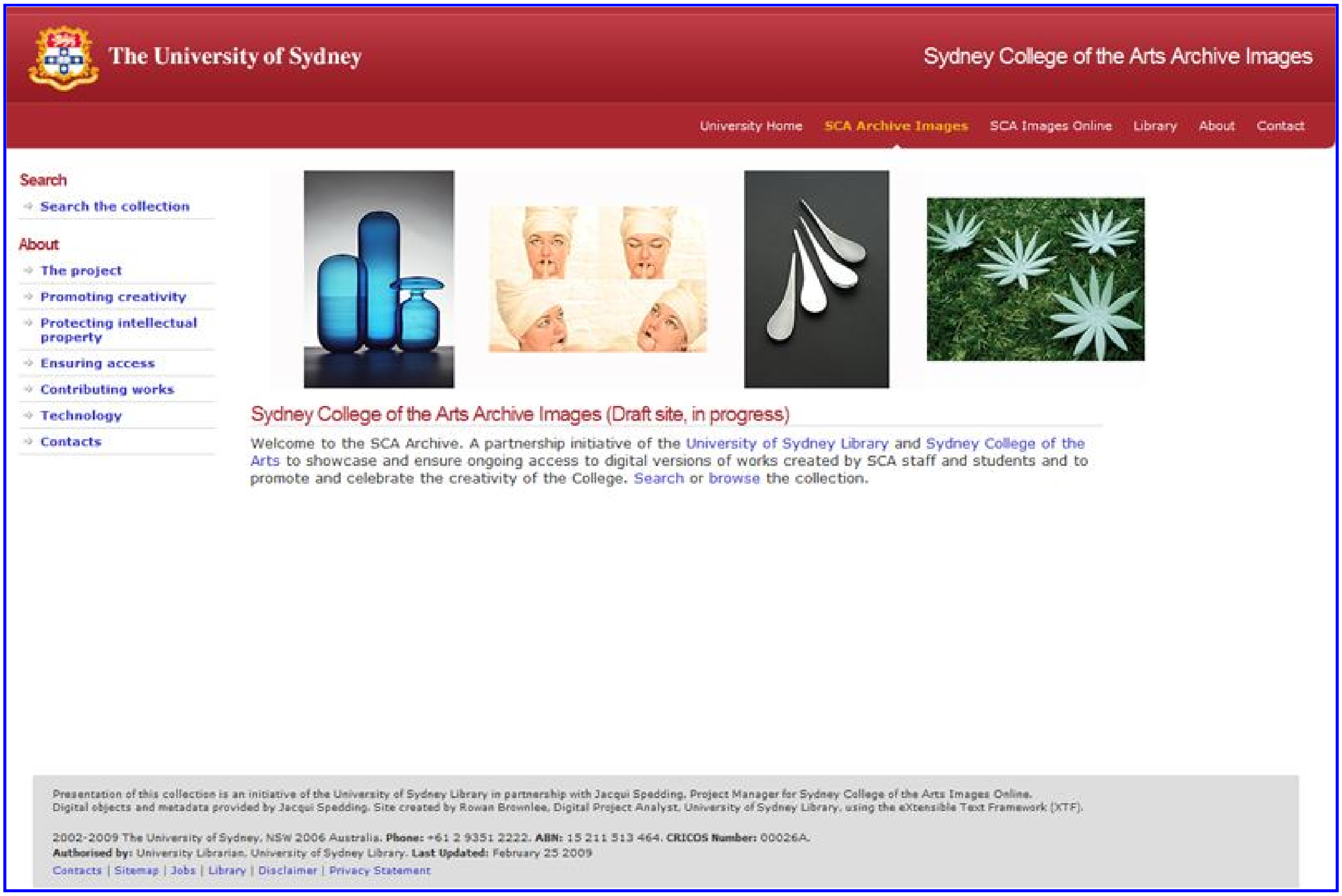
Screenshots

Screenshots from the SCA Archive

- Collection home page
- About the collection
- Search page
- Browse page
- Browse all by facet
- Full record with image display options
- Full record with zoomable image
- Full record with image and video display options
- Full record with video playback
- Full record print view

Collection home page

A static html page providing an entry point to the collection. Useful for promotion and marketing.



Sydney College of the Arts Archive Images (Draft site, in progress)

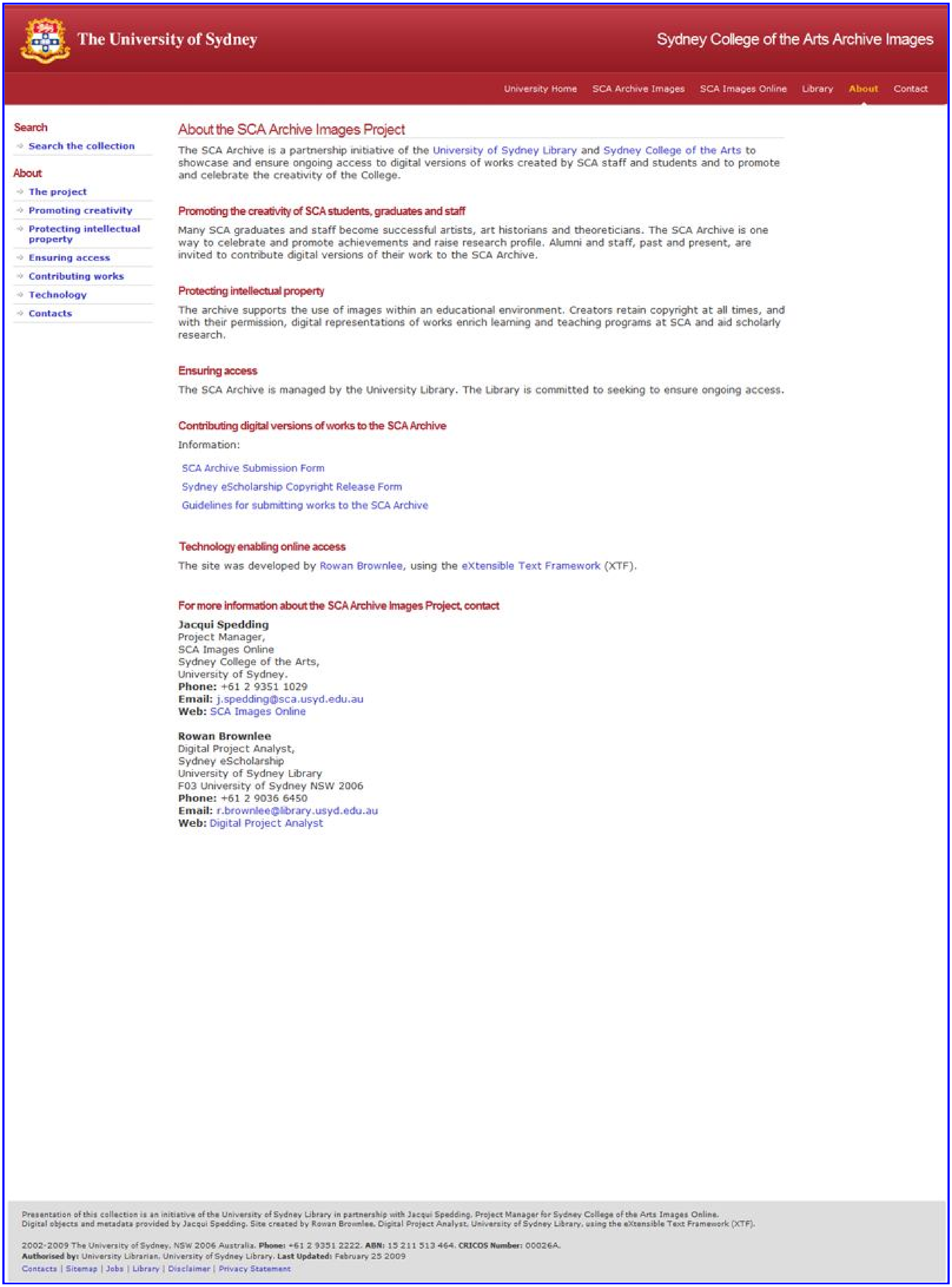
Welcome to the SCA Archive. A partnership initiative of the University of Sydney Library and Sydney College of the Arts to showcase and ensure ongoing access to digital versions of works created by SCA staff and students and to promote and celebrate the creativity of the College. Search or browse the collection.

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 25 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

About the collection

A static html page with background information.



About the SCA Archive Images Project

The SCA Archive is a partnership initiative of the University of Sydney Library and Sydney College of the Arts to showcase and ensure ongoing access to digital versions of works created by SCA staff and students and to promote and celebrate the creativity of the College.

Promoting the creativity of SCA students, graduates and staff

Many SCA graduates and staff become successful artists, art historians and theoreticians. The SCA Archive is one way to celebrate and promote achievements and raise research profile. Alumni and staff, past and present, are invited to contribute digital versions of their work to the SCA Archive.

Protecting intellectual property

The archive supports the use of images within an educational environment. Creators retain copyright at all times, and with their permission, digital representations of works enrich learning and teaching programs at SCA and aid scholarly research.

Ensuring access

The SCA Archive is managed by the University Library. The Library is committed to seeking to ensure ongoing access.

Contributing digital versions of works to the SCA Archive

Information:

SCA Archive Submission Form

Sydney eScholarship Copyright Release Form

Guidelines for submitting works to the SCA Archive

Technology enabling online access

The site was developed by Rowan Brownlee, using the eXtensible Text Framework (XTF).

For more information about the SCA Archive Images Project, contact

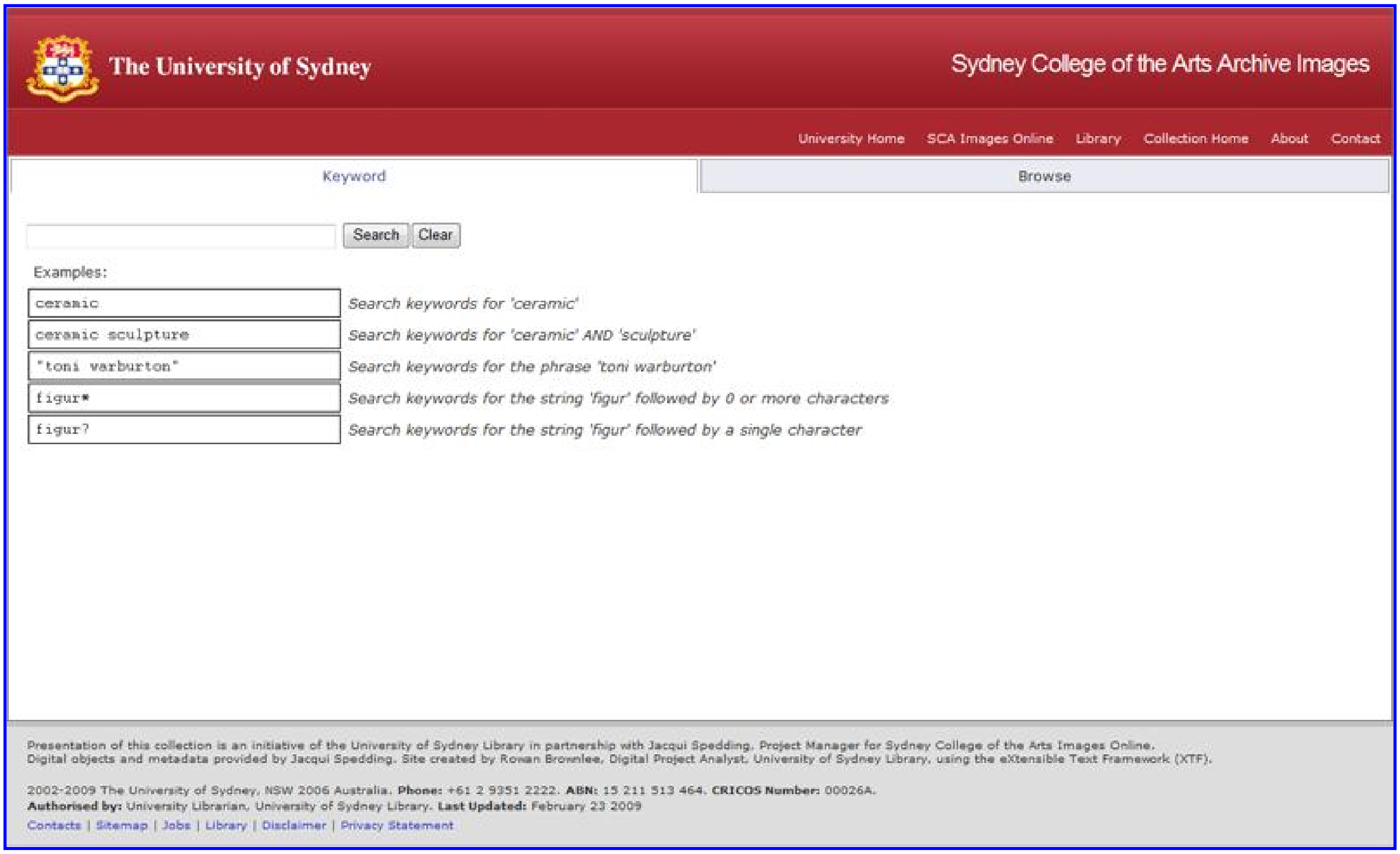
Jacqui Spedding
Project Manager,
SCA Images Online
Sydney College of the Arts,
University of Sydney.
Phone: +61 2 9351 1029
Email: j.spedding@sca.usyd.edu.au
Web: SCA Images Online

Rowan Brownlee
Digital Project Analyst,
Sydney eScholarship
University of Sydney Library
F03 University of Sydney NSW 2006
Phone: +61 2 9036 6450
Email: r.brownlee@library.usyd.edu.au
Web: Digital Project Analyst

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 25 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Search page



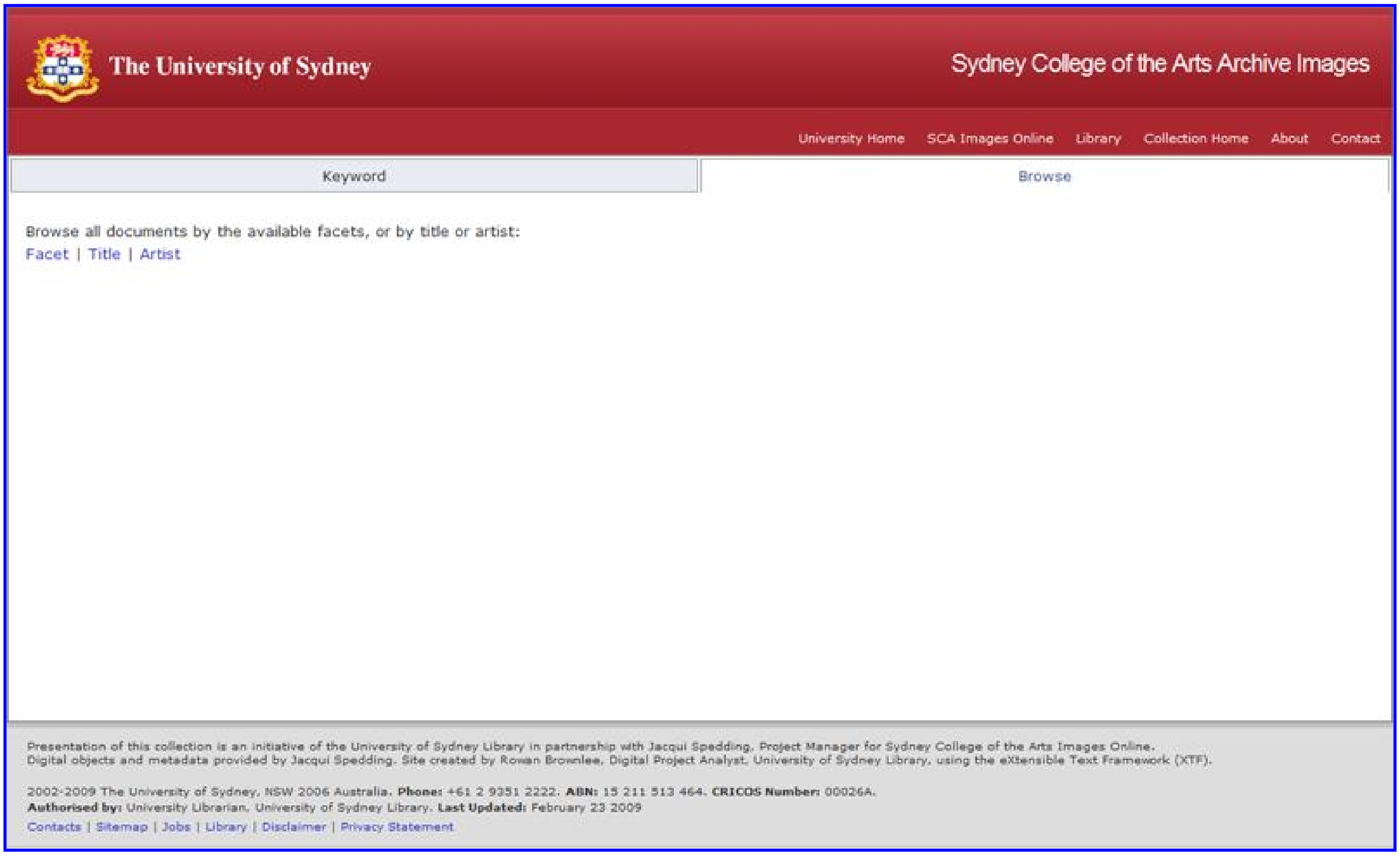
Examples:

`ceramic`	*Search keywords for 'ceramic'*
`ceramic sculpture`	*Search keywords for 'ceramic' AND 'sculpture'*
`"toni warburton"`	*Search keywords for the phrase 'toni warburton'*
`figur*`	*Search keywords for the string 'figur' followed by 0 or more characters*
`figur?`	*Search keywords for the string 'figur' followed by a single character*

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Browse page

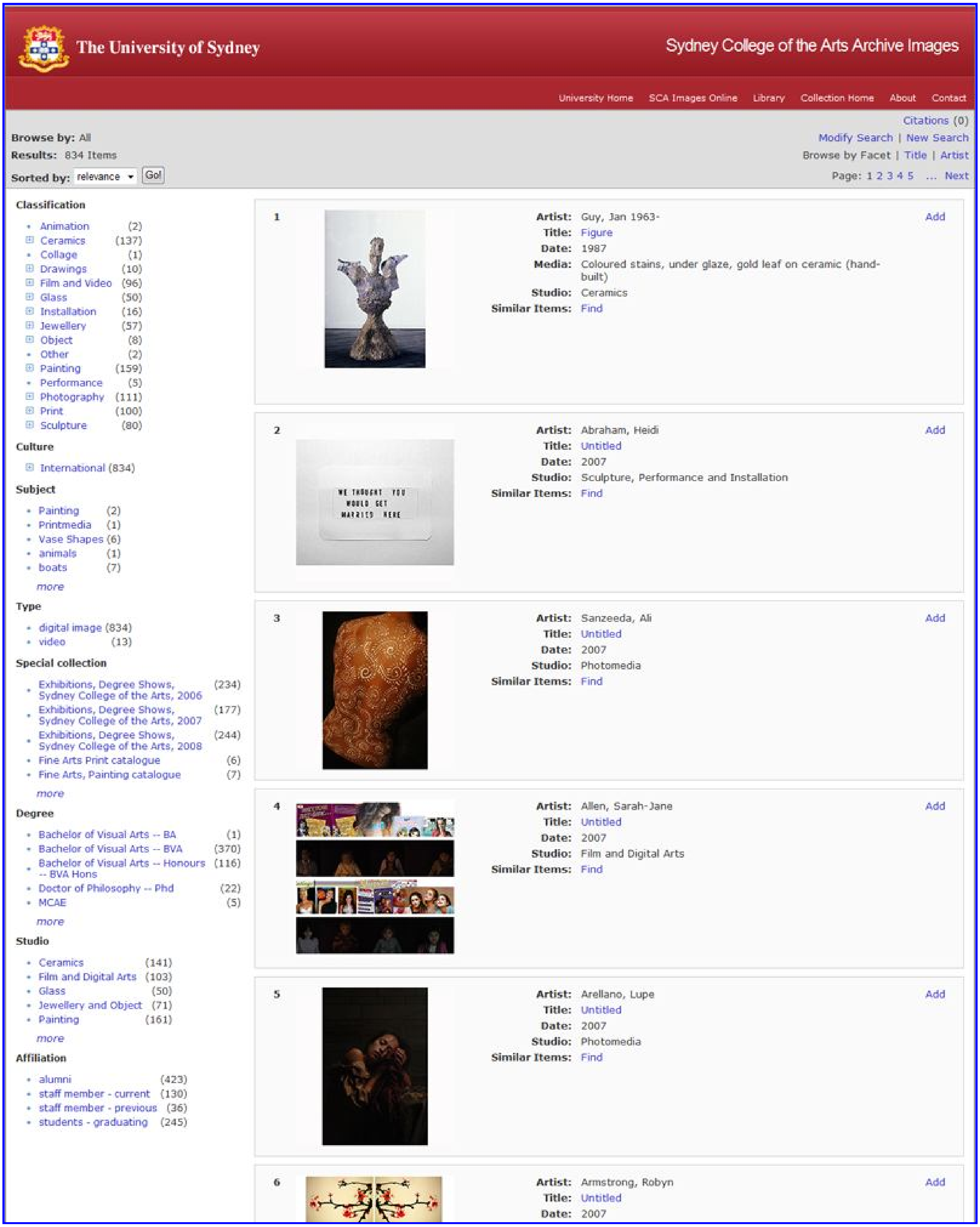


Browse all documents by the available facets, or by title or artist:
Facet | Title | Artist

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Browse all by facet



The University of Sydney

Sydney College of the Arts Archive Images

University Home | SCA Images Online | Library | Collection Home | About | Contact

Citations (0)

Browse by: All

Results: 834 Items

Sorted by: relevance Go!

Modify Search | New Search

Browse by Facet | Title | Artist

Page: 1 2 3 4 5 ... Next

Classification

- Animation (2)
- Ceramics (137)
- Collage (1)
- Drawings (10)
- Film and Video (96)
- Glass (50)
- Installation (16)
- Jewellery (57)
- Object (8)
- Other (2)
- Painting (159)
- Performance (5)
- Photography (111)
- Print (100)
- Sculpture (80)

Culture

- International (834)

Subject

- Painting (2)
- Printmedia (1)
- Vase Shapes (6)
- animals (1)
- boats (7)

more

Type

- digital image (834)
- video (13)

Special collection

- Exhibitions, Degree Shows, Sydney College of the Arts, 2006 (234)
- Exhibitions, Degree Shows, Sydney College of the Arts, 2007 (177)
- Exhibitions, Degree Shows, Sydney College of the Arts, 2008 (244)
- Fine Arts Print catalogue (6)
- Fine Arts, Painting catalogue (7)

more

Degree

- Bachelor of Visual Arts -- BA (1)
- Bachelor of Visual Arts -- BVA (370)
- Bachelor of Visual Arts -- Honours -- BVA Hons (116)
- Doctor of Philosophy -- Phd (22)
- MCAE (5)

more

Studio

- Ceramics (141)
- Film and Digital Arts (103)
- Glass (50)
- Jewellery and Object (71)
- Painting (161)

more

Affiliation

- alumni (423)
- staff member - current (130)
- staff member - previous (36)
- students - graduating (245)

1

Artist: Guy, Jan 1963-
Title: Figure
Date: 1987
Media: Coloured stains, under glaze, gold leaf on ceramic (hand-built)
Studio: Ceramics
Similar Items: Find
Add

2

Artist: Abraham, Heidi
Title: Untitled
Date: 2007
Studio: Sculpture, Performance and Installation
Similar Items: Find
Add

3

Artist: Sanzeeda, Ali
Title: Untitled
Date: 2007
Studio: Photomedia
Similar Items: Find
Add

4

Artist: Allen, Sarah-Jane
Title: Untitled
Date: 2007
Studio: Film and Digital Arts
Similar Items: Find
Add

5

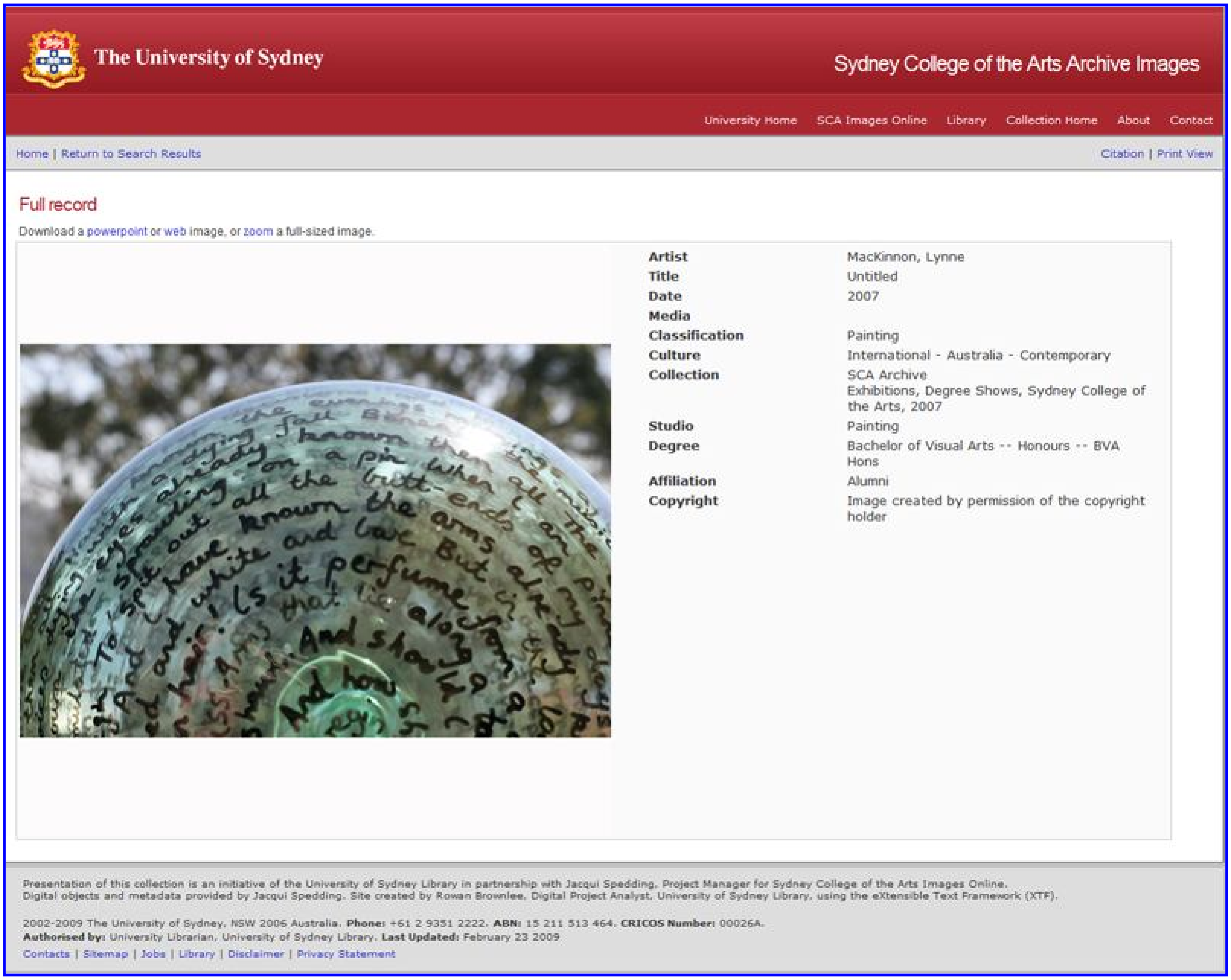
Artist: Arellano, Lupe
Title: Untitled
Date: 2007
Studio: Photomedia
Similar Items: Find
Add

6

Artist: Armstrong, Robyn
Title: Untitled
Date: 2007
Add

Full record with image display options

These display options are visible if the record describes an image. (Options appear below the red *Full record* heading.)



Full record

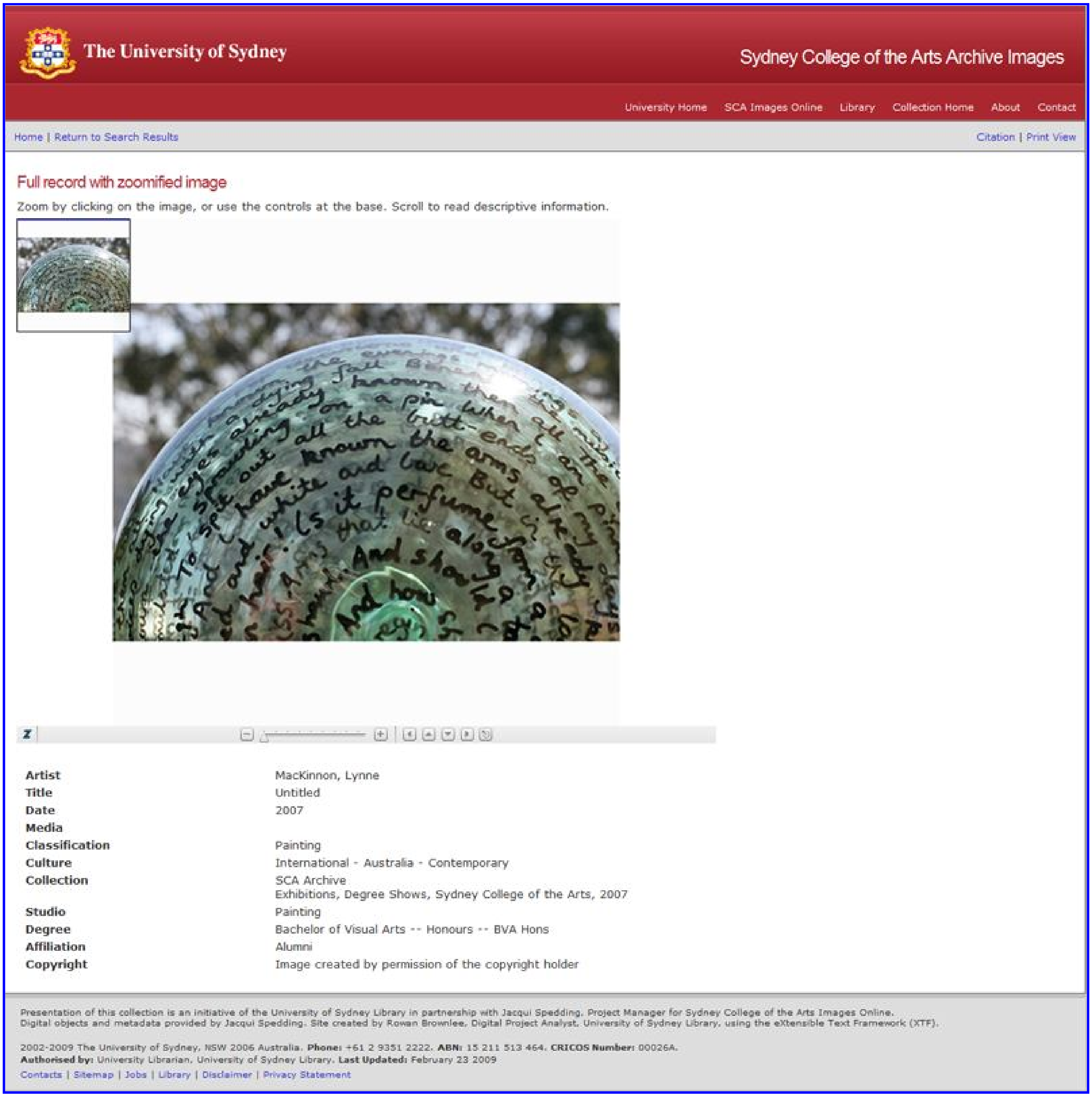
Download a powerpoint or web image, or zoom a full-sized image.

Artist	MacKinnon, Lynne
Title	Untitled
Date	2007
Media	
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Painting
Degree	Bachelor of Visual Arts -- Honours -- BVA Hons
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Full record with zoomable image



Full record with zoomified image

Zoom by clicking on the image, or use the controls at the base. Scroll to read descriptive information.

Artist	MacKinnon, Lynne
Title	Untitled
Date	2007
Media	
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Painting
Degree	Bachelor of Visual Arts -- Honours -- BVA Hons
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

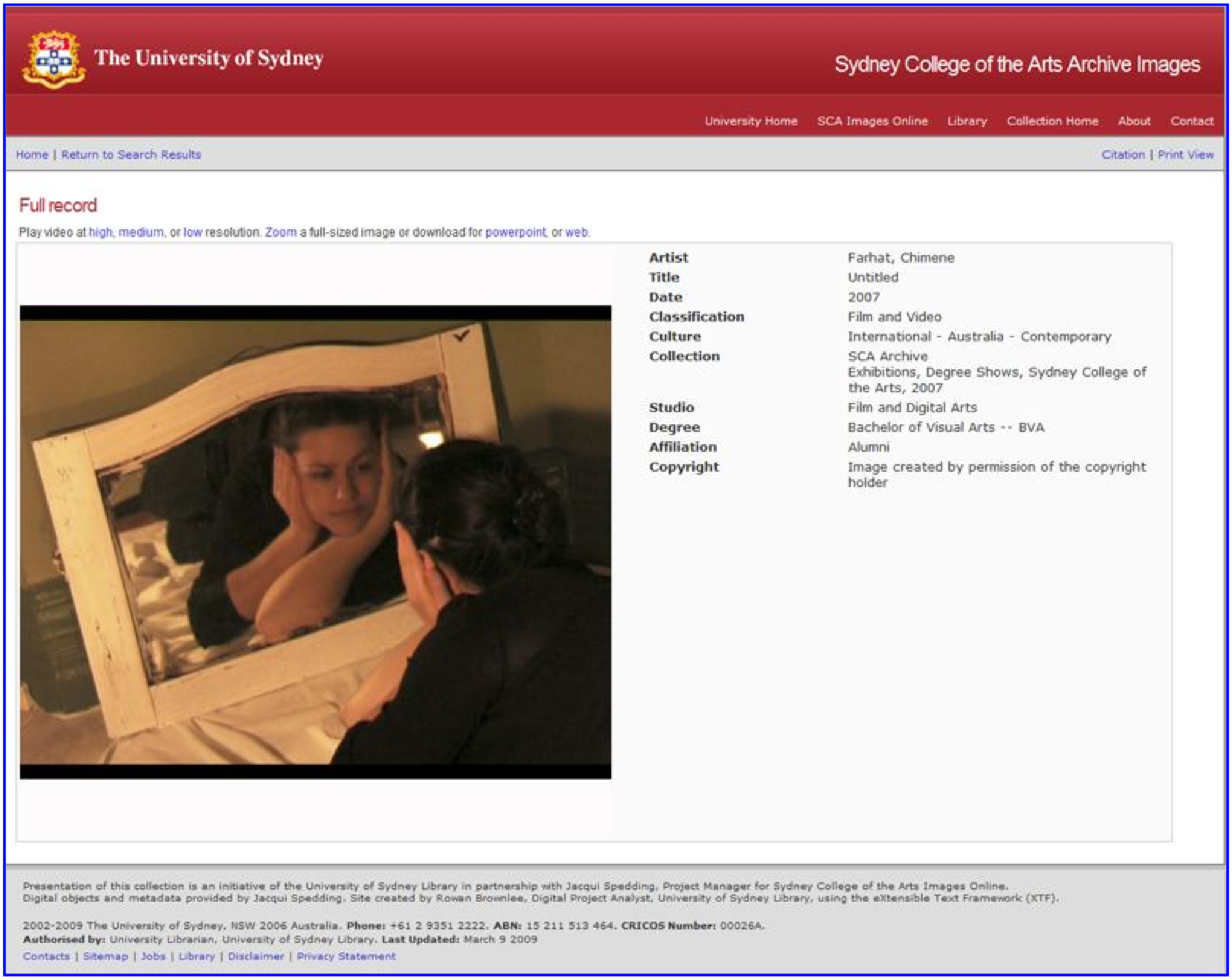
Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Full record with image and video display options

These display options are visible if the record describes a video. (Options appear below the red *Full record* heading.)



Full record

Play video at high, medium, or low resolution. Zoom a full-sized image or download for powerpoint, or web.

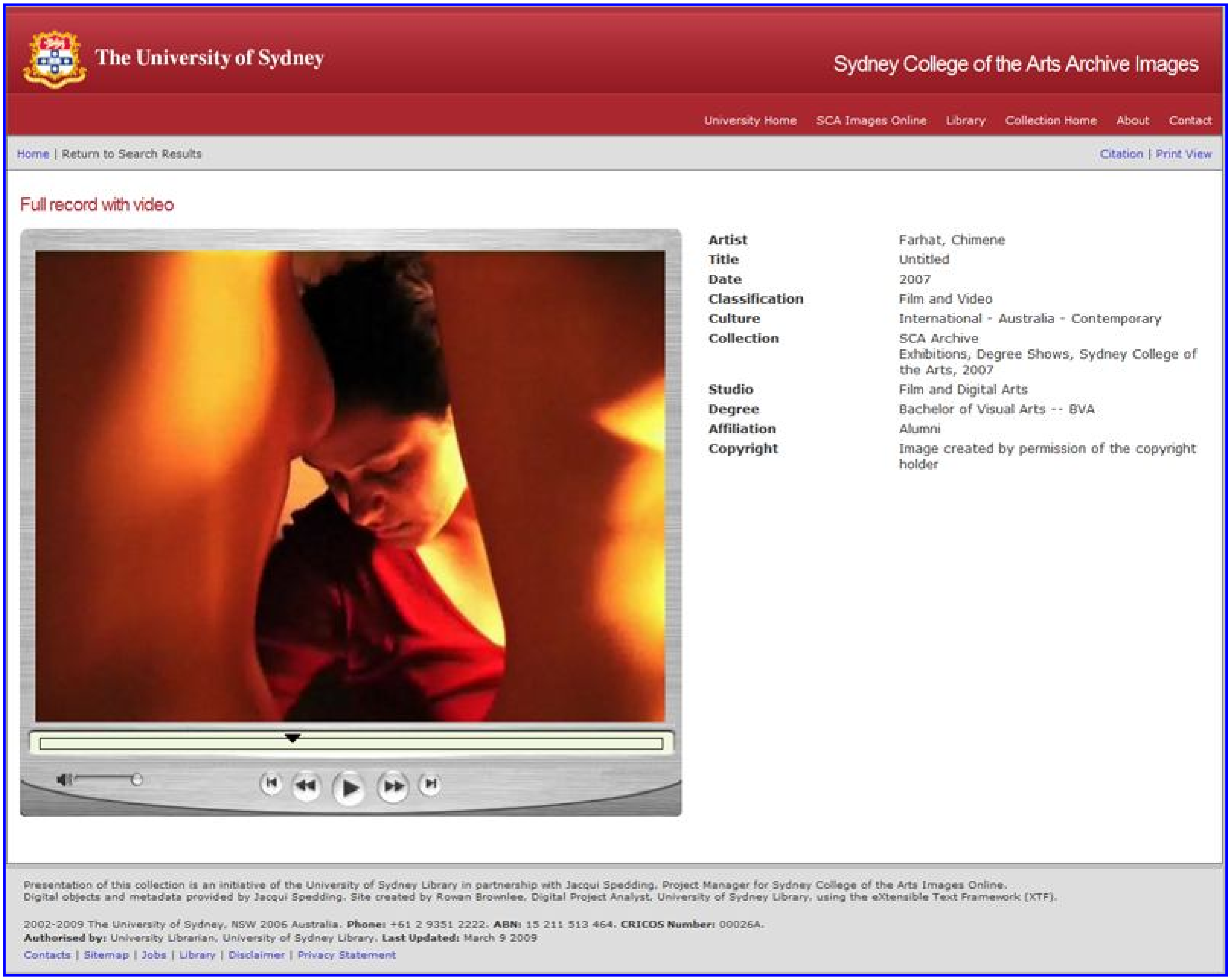
Artist	Farhat, Chimene
Title	Untitled
Date	2007
Classification	Film and Video
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Film and Digital Arts
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** March 9 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Full record with video playback



Full record with video

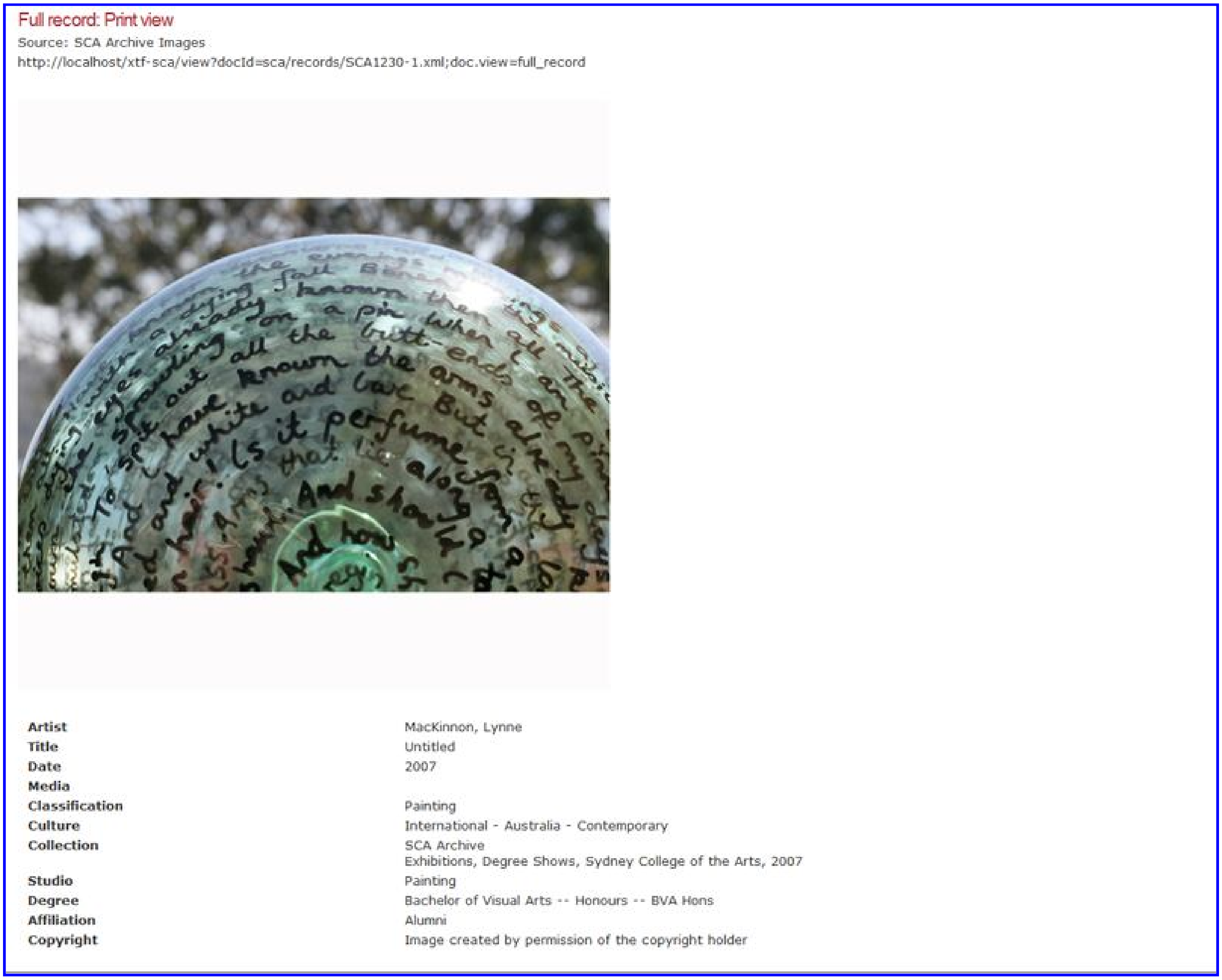
Artist	Farhat, Chimene
Title	Untitled
Date	2007
Classification	Film and Video
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Film and Digital Arts
Degree	Bachelor of Visual Arts -- BVA
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Jacqui Spedding, Project Manager for Sydney College of the Arts Images Online.
Digital objects and metadata provided by Jacqui Spedding. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** March 9 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Full record print view



Full record: Print view

Source: SCA Archive Images

http://localhost/xtf-sca/view?docId=sca/records/SCA1230-1.xml;doc.view=full_record

Artist	MacKinnon, Lynne
Title	Untitled
Date	2007
Media	
Classification	Painting
Culture	International - Australia - Contemporary
Collection	SCA Archive Exhibitions, Degree Shows, Sydney College of the Arts, 2007
Studio	Painting
Degree	Bachelor of Visual Arts -- Honours -- BVA Hons
Affiliation	Alumni
Copyright	Image created by permission of the copyright holder

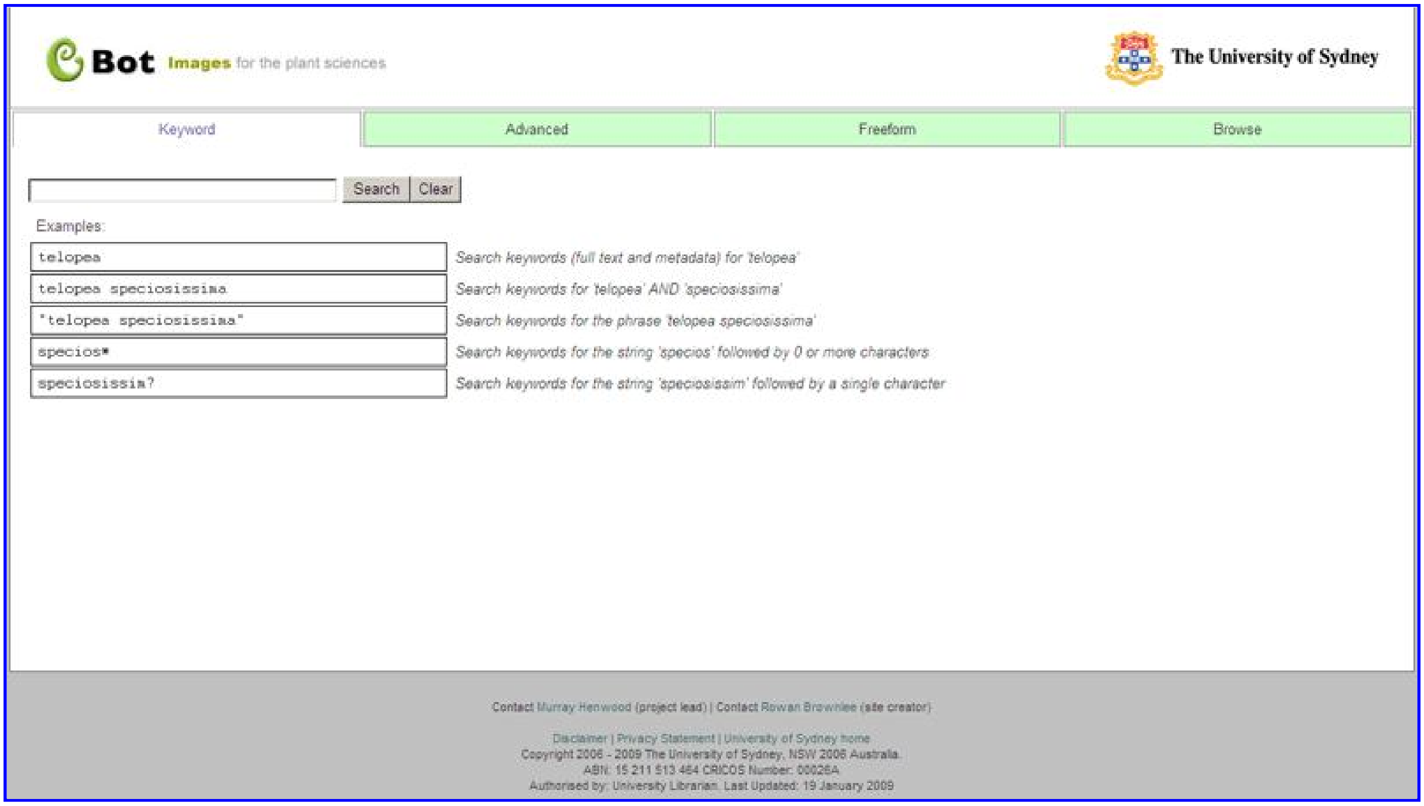
eBot Archive

eBot is a collection of plant sciences images representing the work or University of Sydney academics for use in research, learning and teaching programs. Key project partners include Murray Henwood and Rosanne Quinnell (School of Biological Sciences) and Su Hanfling (Library). More information about eBot, from the proceedings of the 2008 Uniserve Conference. eBot comprises 1856 images.

Screenshots from theeBot archive

- Search page
- Browse page
- Browse all by facet
- Full record
- Full record with zoomable image
- Full record print view

Search page

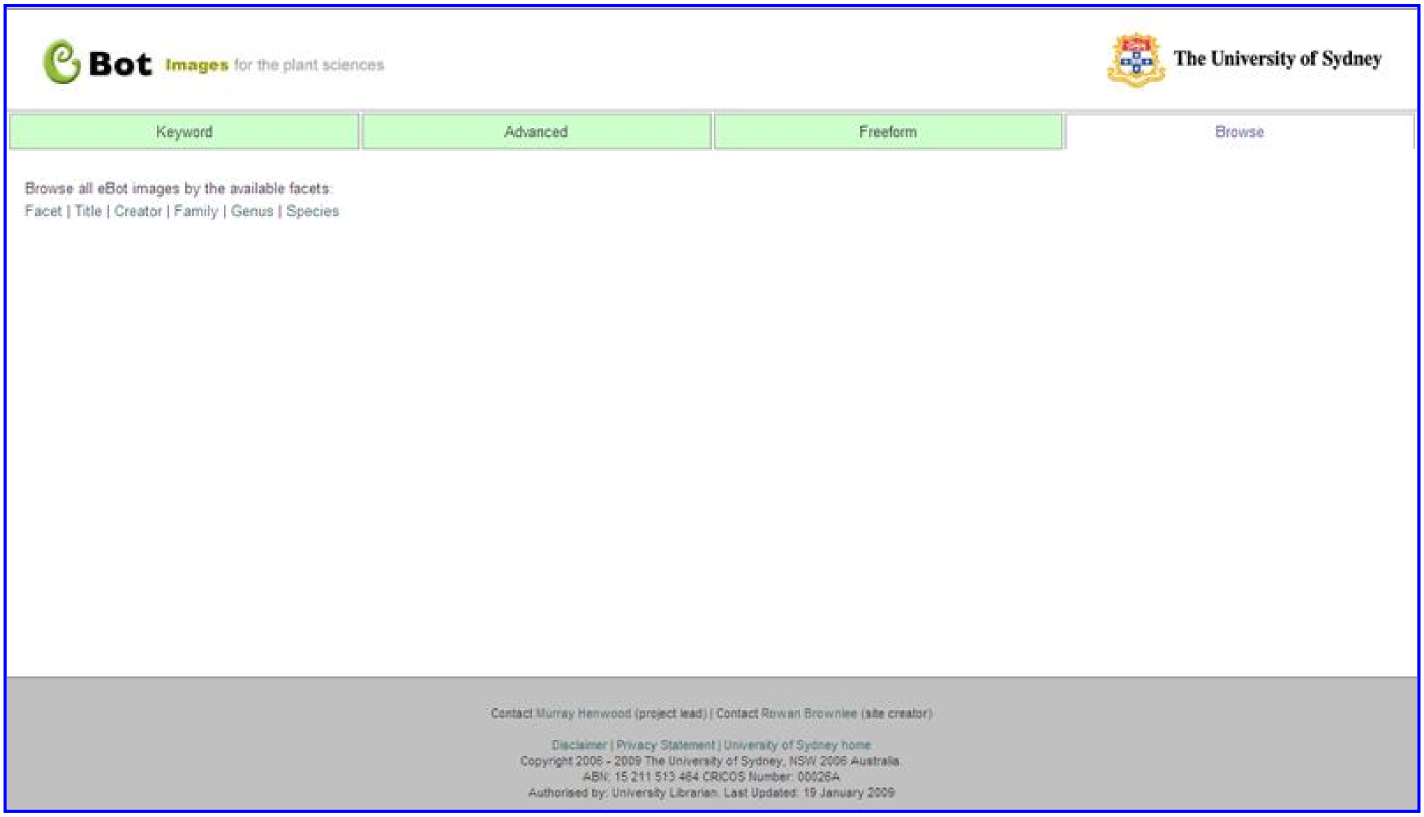


`telopea`	*Search keywords (full text and metadata) for 'telopea'*
`telopea speciosissima`	*Search keywords for 'telopea' AND 'speciosissima'*
`"telopea speciosissima"`	*Search keywords for the phrase 'telopea speciosissima'*
`specios*`	*Search keywords for the string 'specios' followed by 0 or more characters*
`speciosissim?`	*Search keywords for the string 'speciosissim' followed by a single character*

Contact Murray Henwood (project lead) | Contact Rowan Brownlee (site creator)

Disclaimer | Privacy Statement | University of Sydney home
Copyright 2006 - 2009 The University of Sydney, NSW 2006 Australia.
ABN: 15 211 513 464 CRICOS Number: 00026A
Authorised by: University Librarian. Last Updated: 19 January 2009

Browse page

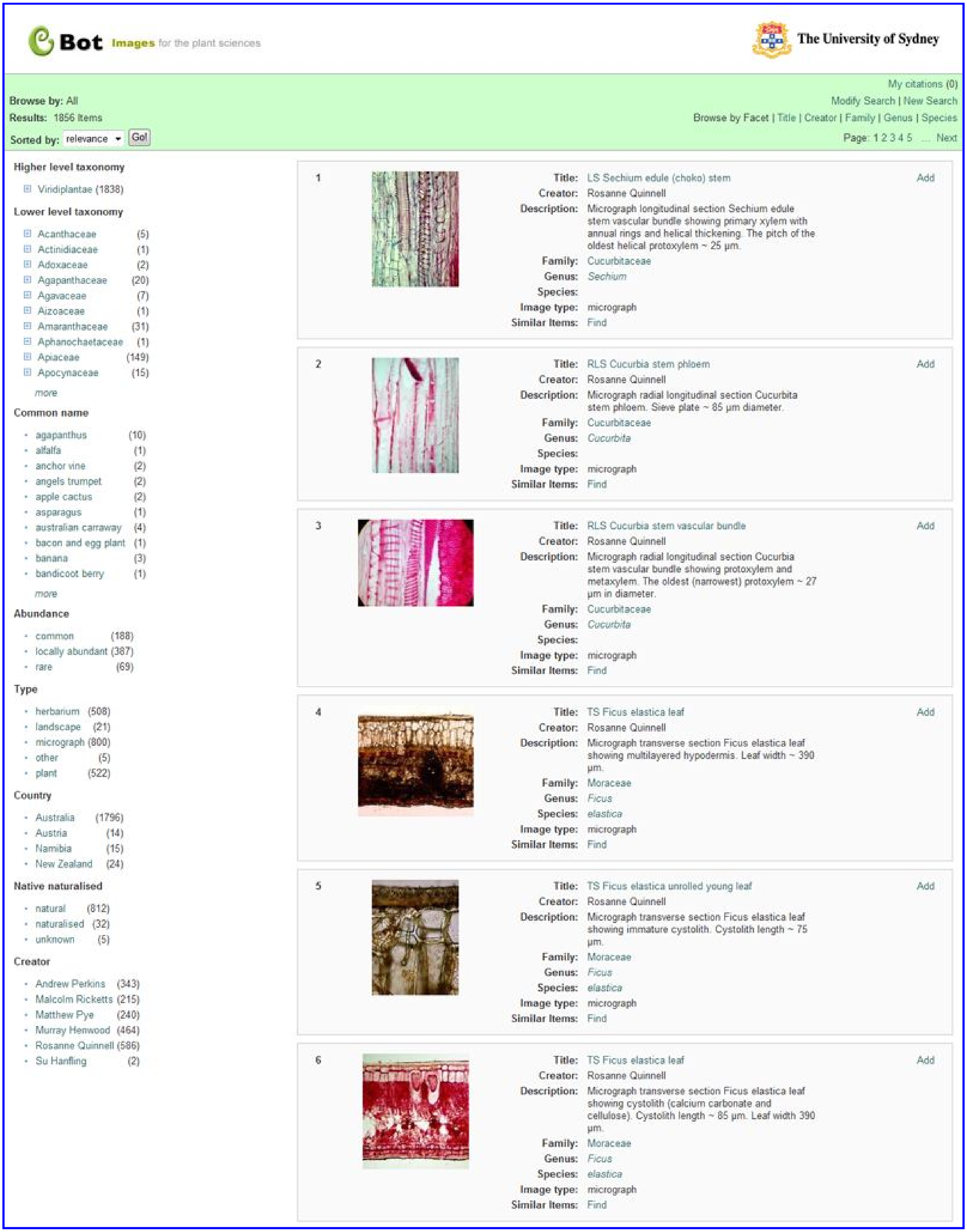


Browse all eBot images by the available facets:
Facet | Title | Creator | Family | Genus | Species

Contact Murray Henwood (project lead) | Contact Rowan Brownlee (site creator)

Disclaimer | Privacy Statement | University of Sydney home
Copyright 2006 - 2009 The University of Sydney, NSW 2006 Australia.
ABN: 15 211 513 464 CRICOS Number: 00026A
Authorised by: University Librarian. Last Updated: 19 January 2009

Browse all by facet



My citations (0)

Browse by: All
Results: 1856 Items
Sorted by: relevance Go!

Modify Search | New Search
Browse by Facet | Title | Creator | Family | Genus | Species
Page: 1 2 3 4 5 ... Next

Higher level taxonomy

⊞ Viridiplantae	(1838)

Lower level taxonomy

⊞ Acanthaceae	(5)
⊞ Actinidiaceae	(1)
⊞ Adoxaceae	(2)
⊞ Agapanthaceae	(20)
⊞ Agavaceae	(7)
⊞ Aizoaceae	(1)
⊞ Amaranthaceae	(31)
⊞ Aphanochaetaceae	(1)
⊞ Apiaceae	(149)
⊞ Apocynaceae	(15)

more

Common name

agapanthus	(10)
alfalfa	(1)
anchor vine	(2)
angels trumpet	(2)
apple cactus	(2)
asparagus	(1)
australian carraway	(4)
bacon and egg plant	(1)
banana	(3)
bandicoot berry	(1)

more

Abundance

common	(188)
locally abundant	(387)
rare	(69)

Type

herbarium	(508)
landscape	(21)
micrograph	(800)
other	(5)
plant	(522)

Country

Australia	(1796)
Austria	(14)
Namibia	(15)
New Zealand	(24)

Native naturalised

natural	(812)
naturalised	(32)
unknown	(5)

Creator

Andrew Perkins	(343)
Malcolm Ricketts	(215)
Matthew Pye	(240)
Murray Henwood	(464)
Rosanne Quinnell	(586)
Su Hanfling	(2)

1

Title: LS Sechium edule (choko) stem
Creator: Rosanne Quinnell
Description: Micrograph longitudinal section Sechium edule stem vascular bundle showing primary xylem with annual rings and helical thickening. The pitch of the oldest helical protoxylem ~ 25 µm.
Family: Cucurbitaceae
Genus: *Sechium*
Species:
Image type: micrograph
Similar Items: Find

Add

2

Title: RLS Cucurbia stem phloem
Creator: Rosanne Quinnell
Description: Micrograph radial longitudinal section Cucurbita stem phloem. Sieve plate ~ 85 µm diameter.
Family: Cucurbitaceae
Genus: *Cucurbita*
Species:
Image type: micrograph
Similar Items: Find

Add

3

Title: RLS Cucurbia stem vascular bundle
Creator: Rosanne Quinnell
Description: Micrograph radial longitudinal section Cucurbia stem vascular bundle showing protoxylem and metaxylem. The oldest (narrowest) protoxylem ~ 27 µm in diameter.
Family: Cucurbitaceae
Genus: *Cucurbita*
Species:
Image type: micrograph
Similar Items: Find

Add

4

Title: TS Ficus elastica leaf
Creator: Rosanne Quinnell
Description: Micrograph transverse section Ficus elastica leaf showing multilayered hypodermis. Leaf width ~ 390 µm.
Family: Moraceae
Genus: *Ficus*
Species: *elastica*
Image type: micrograph
Similar Items: Find

Add

5

Title: TS Ficus elastica unrolled young leaf
Creator: Rosanne Quinnell
Description: Micrograph transverse section Ficus elastica leaf showing immature cystolith. Cystolith length ~ 75 µm.
Family: Moraceae
Genus: *Ficus*
Species: *elastica*
Image type: micrograph
Similar Items: Find

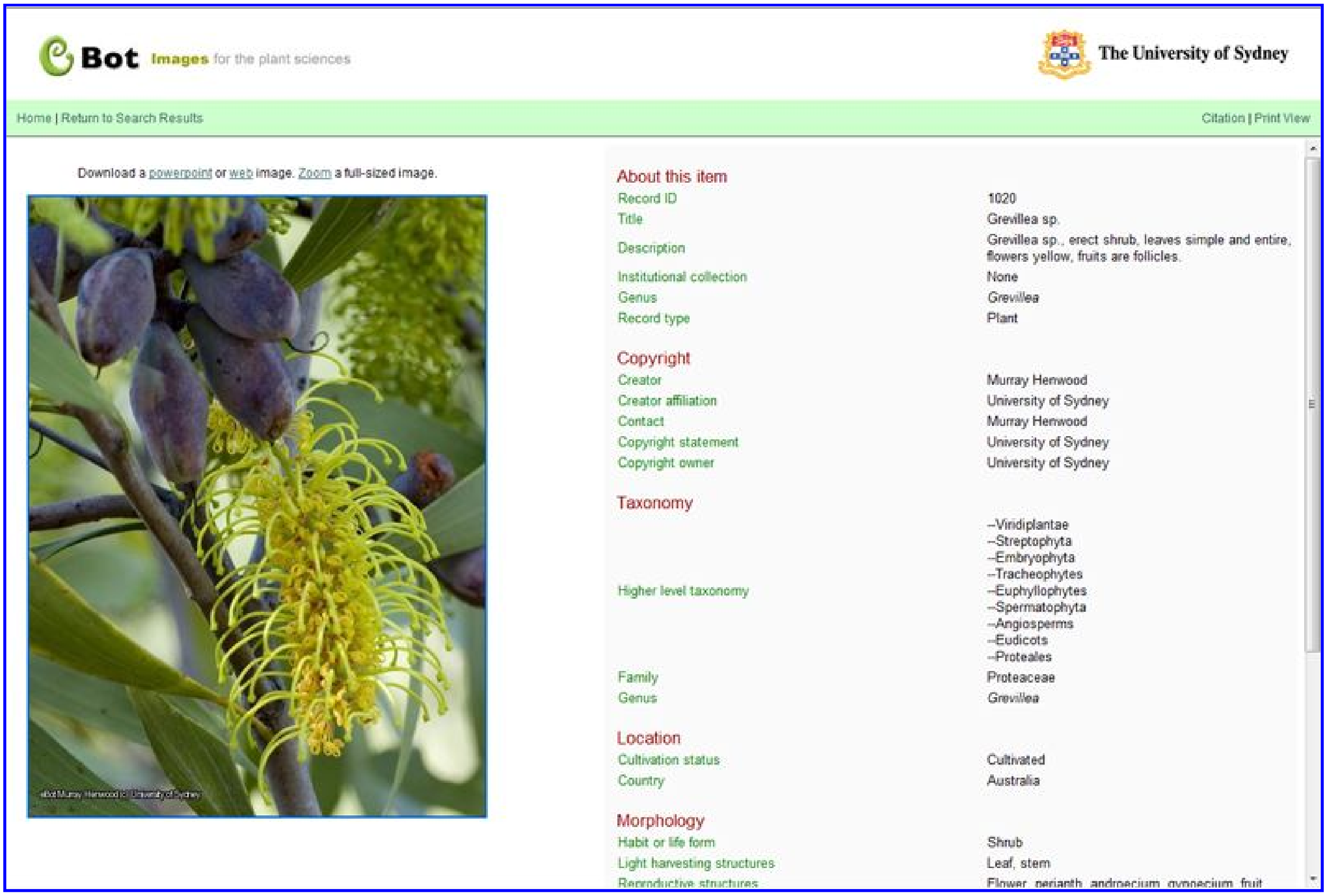
Add

6

Title: TS Ficus elastica leaf
Creator: Rosanne Quinnell
Description: Micrograph transverse section Ficus elastica leaf showing cystolith (calcium carbonate and cellulose). Cystolith length ~ 85 µm. Leaf width 390 µm.
Family: Moraceae
Genus: *Ficus*
Species: *elastica*
Image type: micrograph
Similar Items: Find

Add

Full record



eBot Images for the plant sciences
The University of Sydney
Home | Return to Search Results
Citation | Print View

Download a powerpoint or web image. Zoom a full-sized image.

eBot Murray Henwood (c) University of Sydney

About this item

Record ID	1020
Title	Grevillea sp.
Description	Grevillea sp., erect shrub, leaves simple and entire, flowers yellow, fruits are follicles.
Institutional collection	None
Genus	*Grevillea*
Record type	Plant

Copyright

Creator	Murray Henwood
Creator affiliation	University of Sydney
Contact	Murray Henwood
Copyright statement	University of Sydney
Copyright owner	University of Sydney

Taxonomy

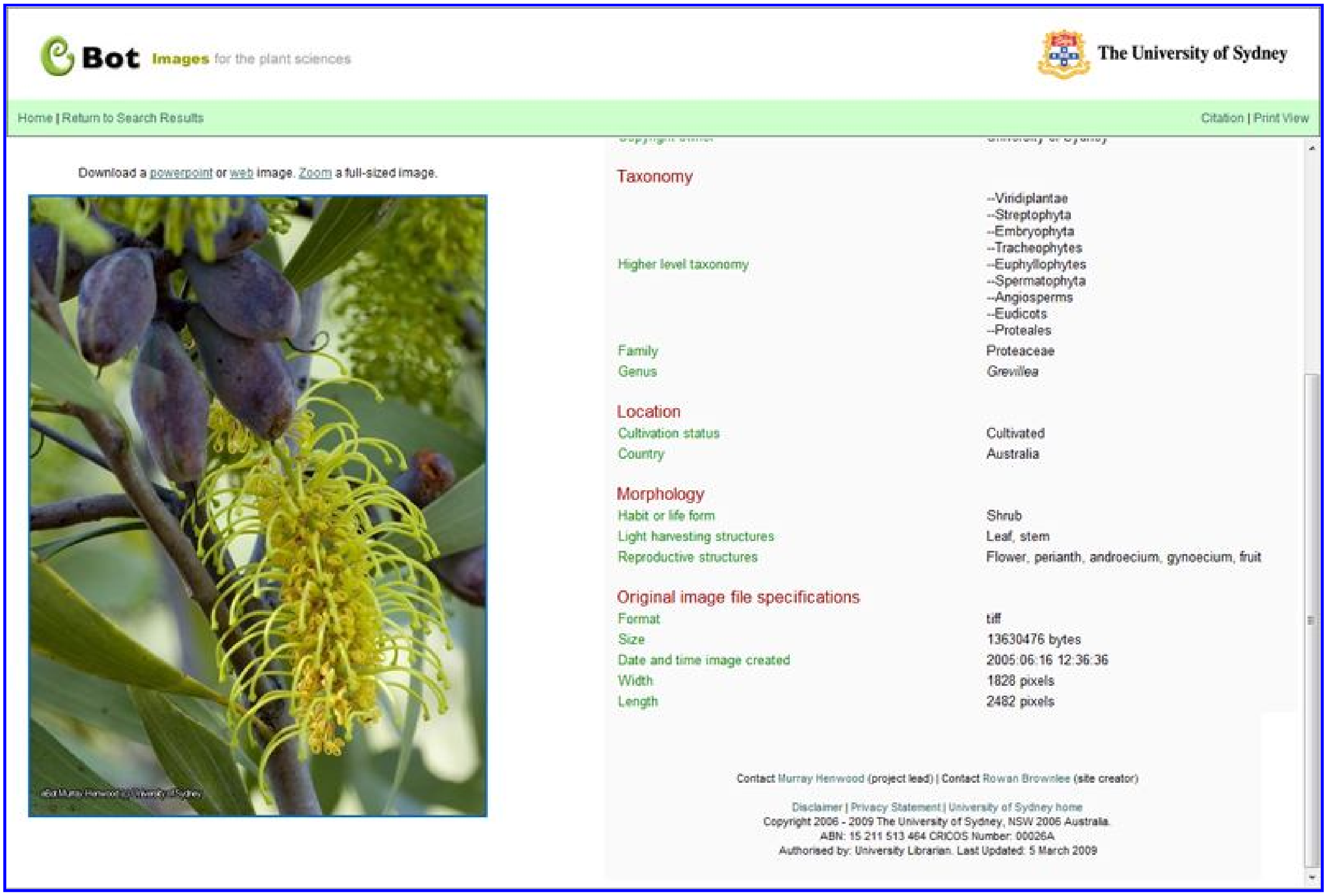
Higher level taxonomy	--Viridiplantae --Streptophyta --Embryophyta --Tracheophytes --Euphyllophytes --Spermatophyta --Angiosperms --Eudicots --Proteales
Family	Proteaceae
Genus	*Grevillea*

Location

Cultivation status	Cultivated
Country	Australia

Morphology

Habit or life form	Shrub
Light harvesting structures	Leaf, stem
Reproductive structures	Flower, perianth, androecium, gynoecium, fruit


eBot Images for the plant sciences
The University of Sydney
Home | Return to Search Results
Citation | Print View

Download a powerpoint or web image. Zoom a full-sized image.

Taxonomy

Higher level taxonomy	--Viridiplantae --Streptophyta --Embryophyta --Tracheophytes --Euphyllophytes --Spermatophyta --Angiosperms --Eudicots --Proteales
Family	Proteaceae
Genus	*Grevillea*

Location

Cultivation status	Cultivated
Country	Australia

Morphology

Habit or life form	Shrub
Light harvesting structures	Leaf, stem
Reproductive structures	Flower, perianth, androecium, gynoecium, fruit

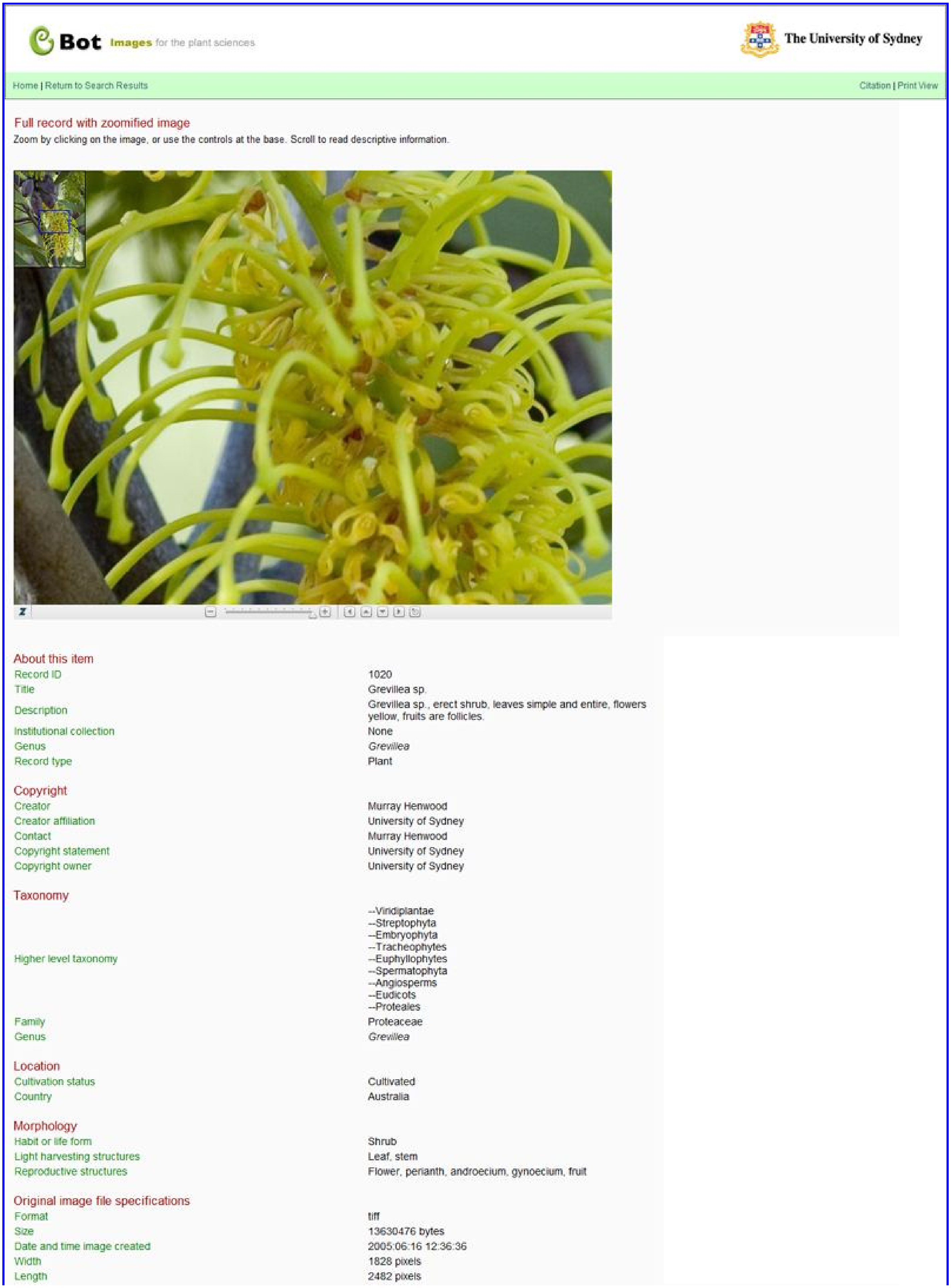
Original image file specifications

Format	tiff
Size	13630476 bytes
Date and time image created	2005:06:16 12:36:36
Width	1828 pixels
Length	2482 pixels

Contact Murray Henwood (project lead) | Contact Rowan Brownlee (site creator)

Disclaimer | Privacy Statement | University of Sydney home
Copyright 2006 - 2009 The University of Sydney, NSW 2006 Australia.
ABN: 15 211 513 464 CRICOS Number: 00026A
Authorised by: University Librarian. Last Updated: 5 March 2009

Full record with zoomable image


Bot Images for the plant sciences
The University of Sydney
Home | Return to Search Results
Citation | Print View

Full record with zoomified image

Zoom by clicking on the image, or use the controls at the base. Scroll to read descriptive information.

About this item

Record ID	1020
Title	Grevillea sp.
Description	Grevillea sp., erect shrub, leaves simple and entire, flowers yellow, fruits are follicles.
Institutional collection	None
Genus	*Grevillea*
Record type	Plant

Copyright

Creator	Murray Henwood
Creator affiliation	University of Sydney
Contact	Murray Henwood
Copyright statement	University of Sydney
Copyright owner	University of Sydney

Taxonomy

Higher level taxonomy	--Viridiplantae --Streptophyta --Embryophyta --Tracheophytes --Euphyllophytes --Spermatophyta --Angiosperms --Eudicots --Proteales
Family	Proteaceae
Genus	*Grevillea*

Location

Cultivation status	Cultivated
Country	Australia

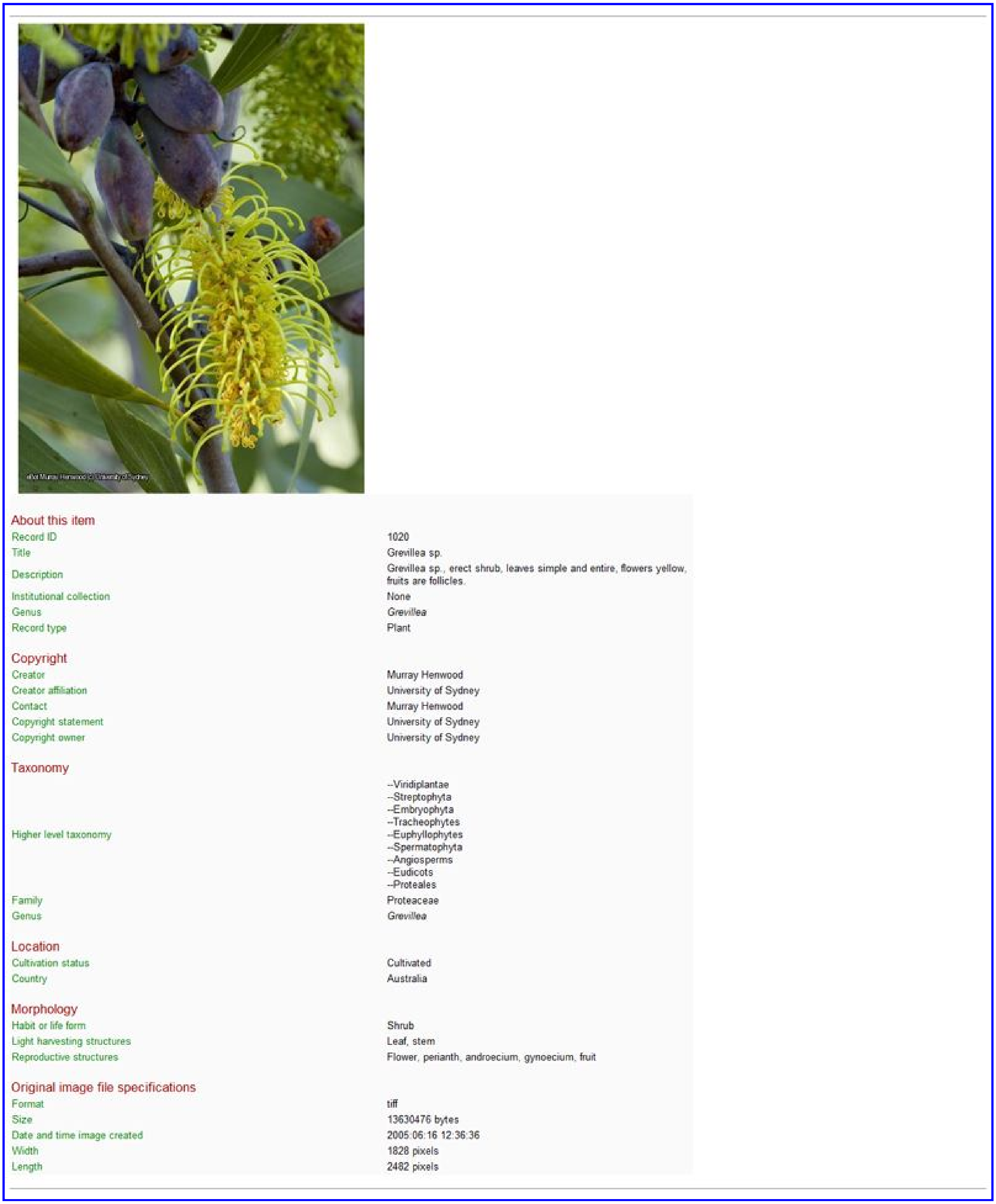
Morphology

Habit or life form	Shrub
Light harvesting structures	Leaf, stem
Reproductive structures	Flower, perianth, androecium, gynoecium, fruit

Original image file specifications

Format	tiff
Size	13630476 bytes
Date and time image created	2005:06:16 12:36:36
Width	1828 pixels
Length	2482 pixels

Full record print view



About this item

Record ID	1020
Title	Grevillea sp.
Description	Grevillea sp., erect shrub, leaves simple and entire, flowers yellow, fruits are follicles.
Institutional collection	None
Genus	Grevillea
Record type	Plant

Copyright

Creator	Murray Henwood
Creator affiliation	University of Sydney
Contact	Murray Henwood
Copyright statement	University of Sydney
Copyright owner	University of Sydney

Taxonomy

Higher level taxonomy	--Viridiplantae --Streptophyta --Embryophyta --Tracheophytes --Euphyllophytes --Spermatophyta --Angiosperms --Eudicots --Proteales
Family	Proteaceae
Genus	Grevillea

Location

Cultivation status	Cultivated
Country	Australia

Morphology

Habit or life form	Shrub
Light harvesting structures	Leaf, stem
Reproductive structures	Flower, perianth, androecium, gynoecium, fruit

Original image file specifications

Format	tiff
Size	13630476 bytes
Date and time image created	2005:06:16 12:36:36
Width	1828 pixels
Length	2482 pixels

Sarah Colley's fish-bone image collection

Sarah Colley's archaeological collection contains images of fish-bones and provides support for her research and that of her project partners. One of Sarah's research interests concerns the archaeological study of fish and fishing in Sydney before and after 1788. The collection comprises 812 images.

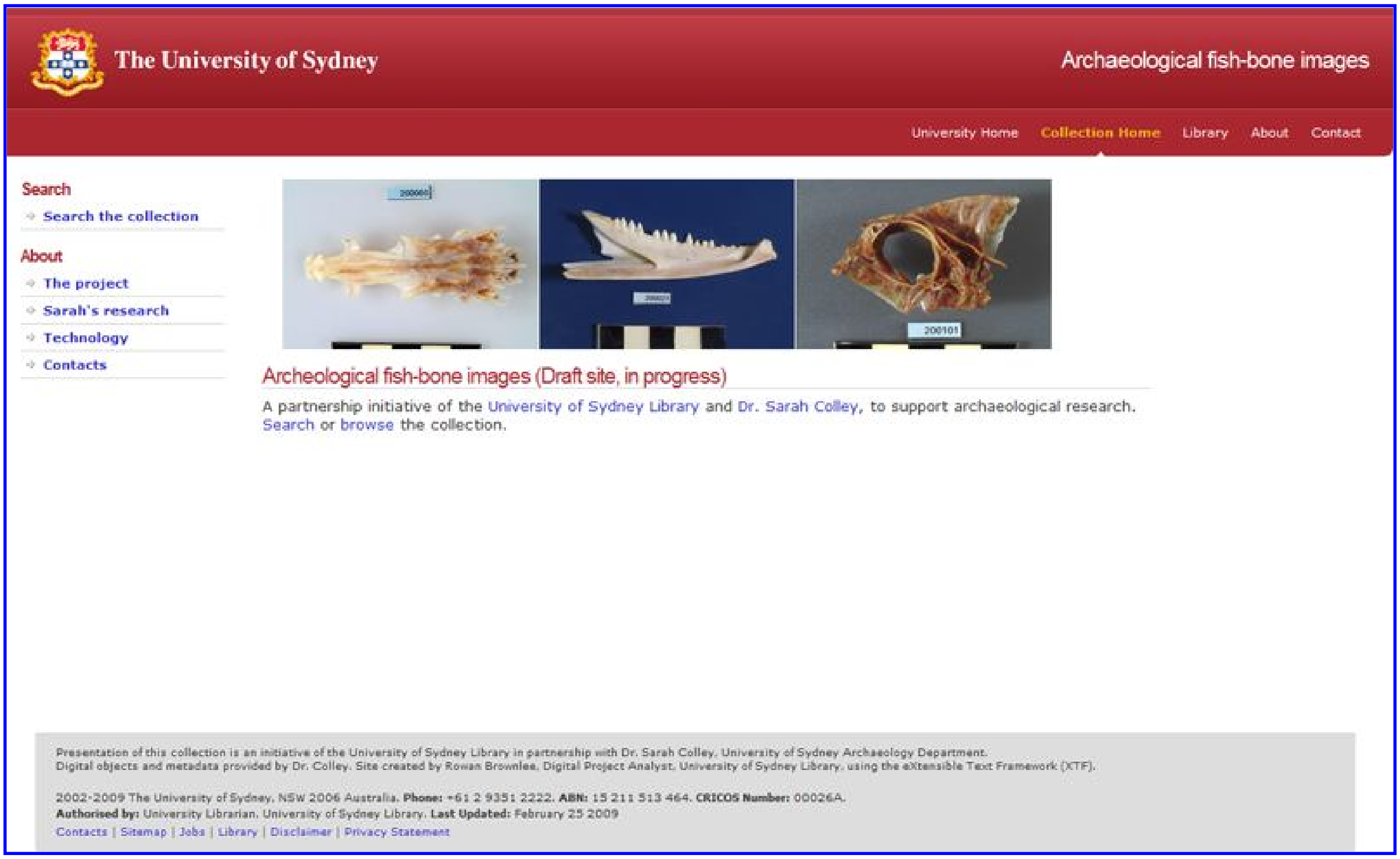
Screenshots from Sarah Colley's fish-bone collection

Sarah Colley's archaeological collection contains images of fish-bones and provides support for her research and that of her project partners. One of Sarah's research interests concerns the archaeological study of fish and fishing in Sydney before and after 1788. At this time the collection is available for use by Sarah's project partners, though in future it will become openly accessible. The collection comprises 487 images.

- Collection home page
- About the collection
- Search page
- Browse page
- Browse all by facet
- Full record
- Full record with zoomable image
- Full record print view

Collection home page

A static html page providing an entry point to the collection. Useful for promotion and marketing.



Archeological fish-bone images (Draft site, in progress)

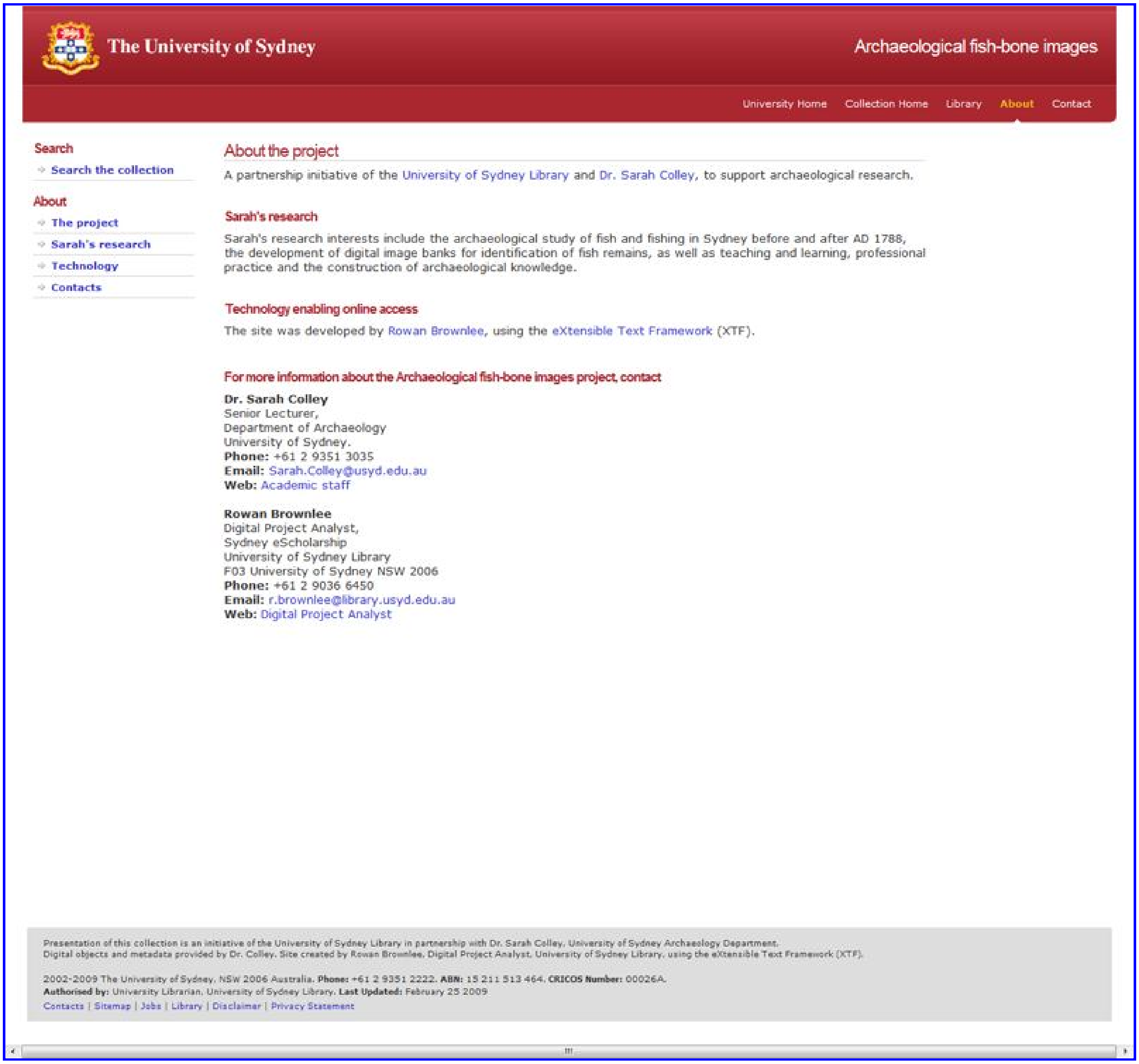
A partnership initiative of the University of Sydney Library and Dr. Sarah Colley, to support archaeological research.
Search or browse the collection.

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Dr. Sarah Colley, University of Sydney Archaeology Department.
Digital objects and metadata provided by Dr. Colley. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 25 2009
Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

About the collection

A static html page with background information.



About the project

A partnership initiative of the University of Sydney Library and Dr. Sarah Colley, to support archaeological research.

Sarah's research

Sarah's research interests include the archaeological study of fish and fishing in Sydney before and after AD 1788, the development of digital image banks for identification of fish remains, as well as teaching and learning, professional practice and the construction of archaeological knowledge.

Technology enabling online access

The site was developed by Rowan Brownlee, using the eXtensible Text Framework (XTF).

For more information about the Archaeological fish-bone images project, contact

Dr. Sarah Colley
Senior Lecturer,
Department of Archaeology
University of Sydney.
Phone: +61 2 9351 3035
Email: Sarah.Colley@usyd.edu.au
Web: Academic staff

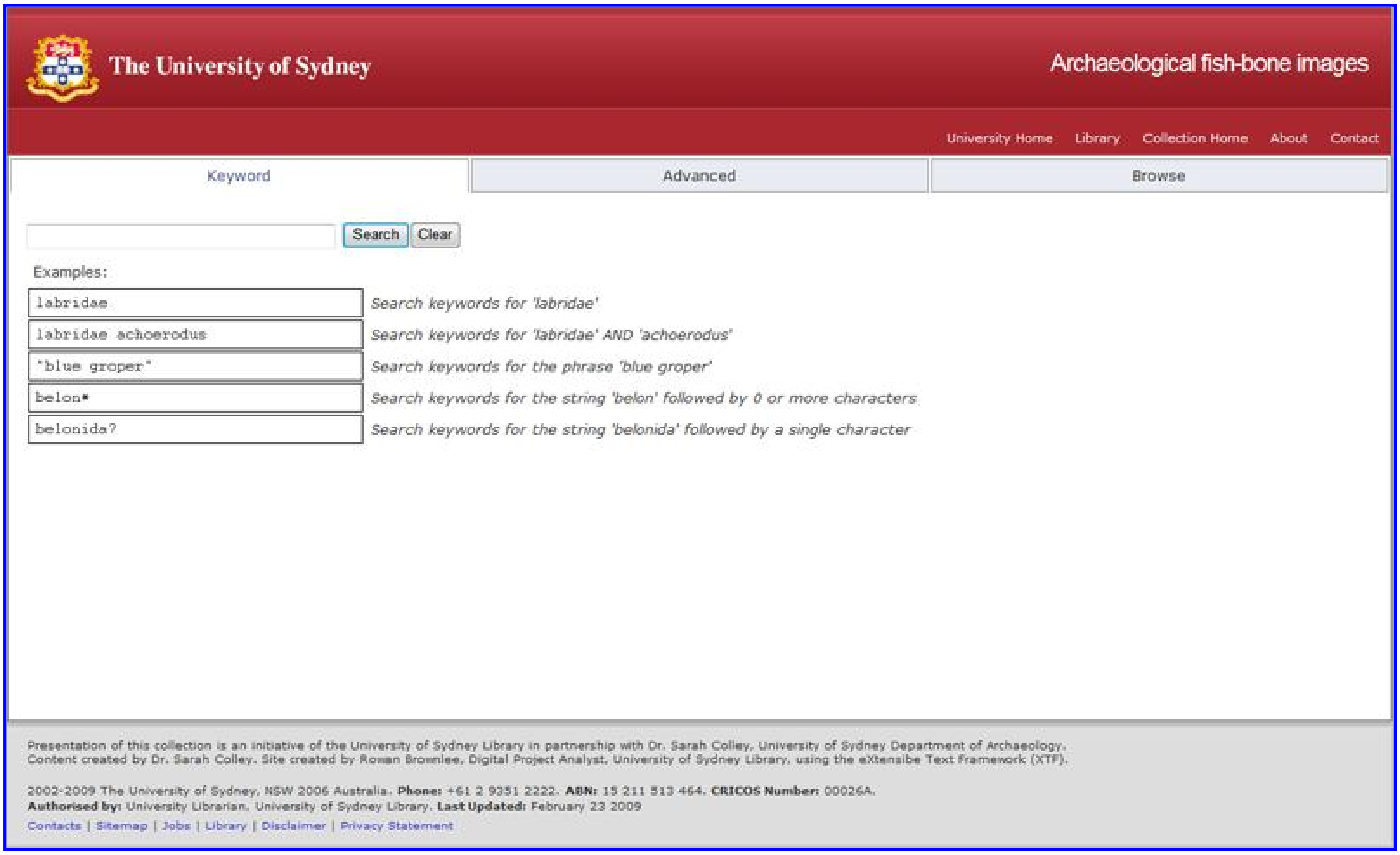
Rowan Brownlee
Digital Project Analyst,
Sydney eScholarship
University of Sydney Library
F03 University of Sydney NSW 2006
Phone: +61 2 9036 6450
Email: r.brownlee@library.usyd.edu.au
Web: Digital Project Analyst

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Dr. Sarah Colley, University of Sydney Archaeology Department.
Digital objects and metadata provided by Dr. Colley. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 25 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Search page



Examples:

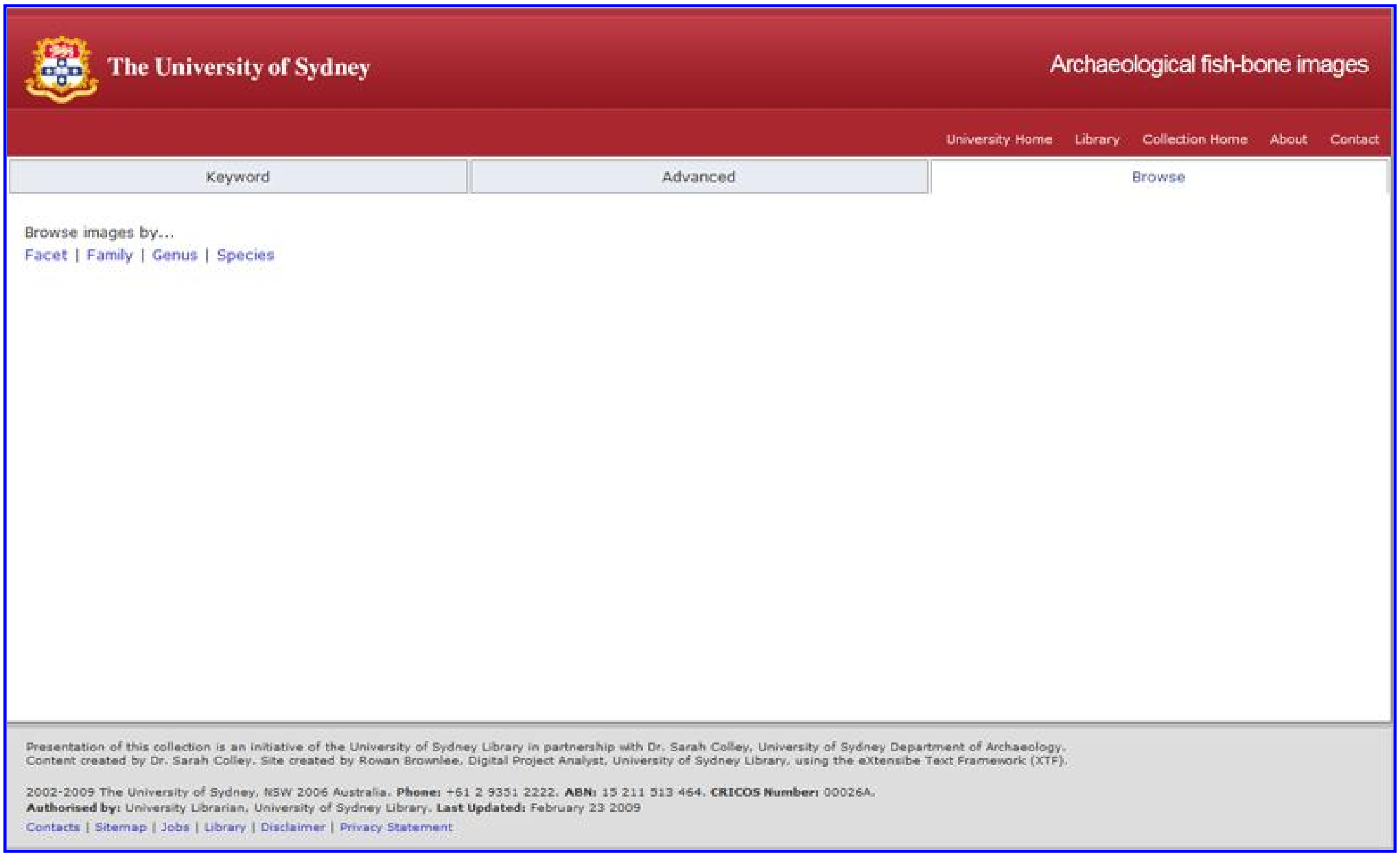
`labridae`	*Search keywords for 'labridae'*
`labridae achoerodus`	*Search keywords for 'labridae' AND 'achoerodus'*
`"blue groper"`	*Search keywords for the phrase 'blue groper'*
`belon*`	*Search keywords for the string 'belon' followed by 0 or more characters*
`belonida?`	*Search keywords for the string 'belonida' followed by a single character*

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Dr. Sarah Colley, University of Sydney Department of Archaeology. Content created by Dr. Sarah Colley. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensibe Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Browse page



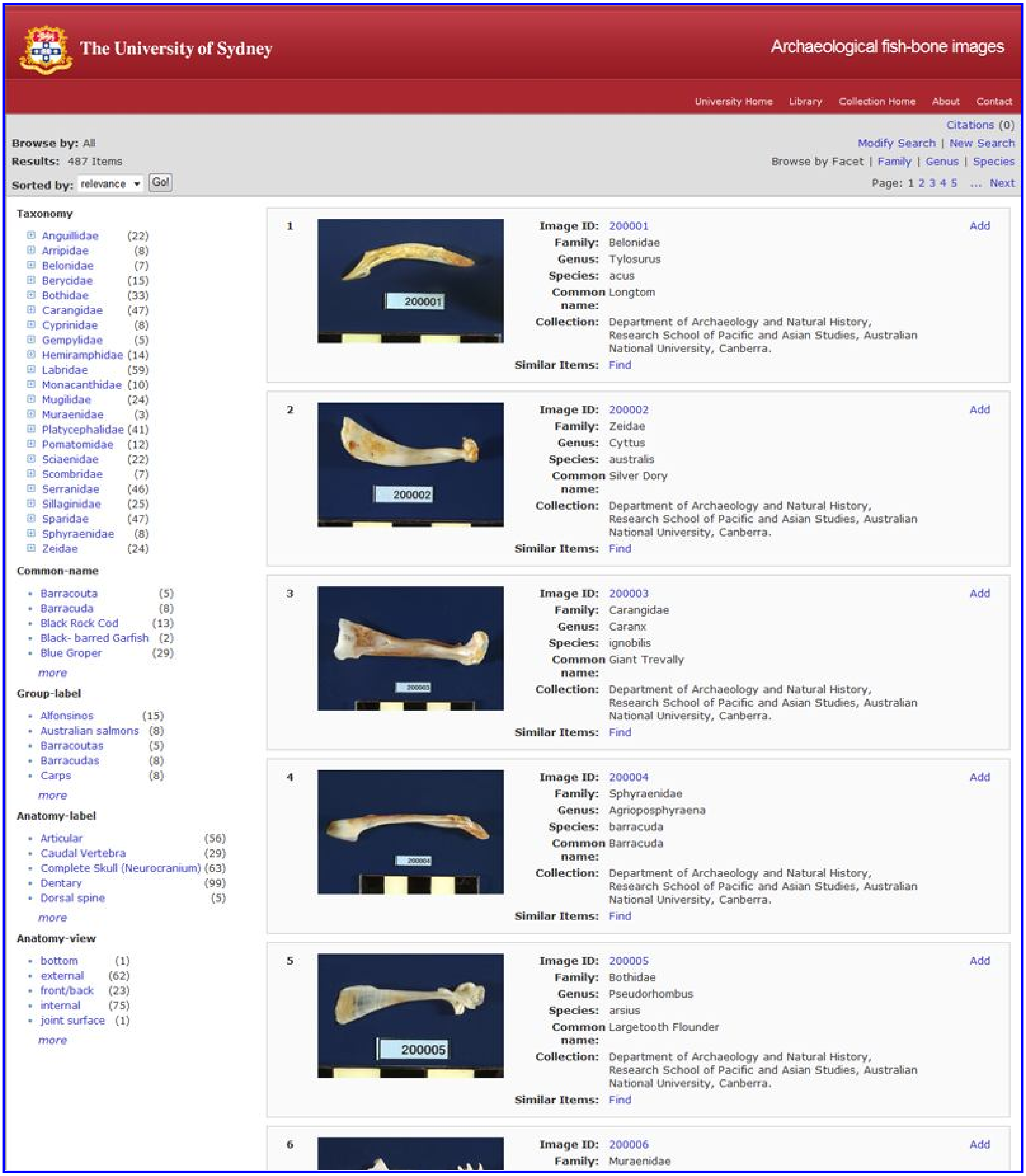
Browse images by...
Facet | Family | Genus | Species

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Dr. Sarah Colley, University of Sydney Department of Archaeology. Content created by Dr. Sarah Colley. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensibe Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Browse all by facet



The University of Sydney

Archaeological fish-bone images

Browse by: All

Results: 487 Items

Sorted by: relevance

Taxonomy

Anguillidae	(22)
Arripidae	(8)
Belonidae	(7)
Berycidae	(15)
Bothidae	(33)
Carangidae	(47)
Cyprinidae	(8)
Gempylidae	(5)
Hemiramphidae	(14)
Labridae	(59)
Monacanthidae	(10)
Mugilidae	(24)
Muraenidae	(3)
Platycephalidae	(41)
Pomatomidae	(12)
Sciaenidae	(22)
Scombridae	(7)
Serranidae	(46)
Sillaginidae	(25)
Sparidae	(47)
Sphyraenidae	(8)
Zeidae	(24)

Common-name

Barracouta	(5)
Barracuda	(8)
Black Rock Cod	(13)
Black- barred Garfish	(2)
Blue Groper	(29)

more

Group-label

Alfonsinos	(15)
Australian salmons	(8)
Barracoutas	(5)
Barracudas	(8)
Carps	(8)

more

Anatomy-label

Articular	(56)
Caudal Vertebra	(29)
Complete Skull (Neurocranium)	(63)
Dentary	(99)
Dorsal spine	(5)

more

Anatomy-view

bottom	(1)
external	(62)
front/back	(23)
internal	(75)
joint surface	(1)

more

1

Image ID: 200001
Family: Belonidae
Genus: Tylosurus
Species: acus
Common name: Longtom
Collection: Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Similar Items: Find

Add

2

Image ID: 200002
Family: Zeidae
Genus: Cyttus
Species: australis
Common name: Silver Dory
Collection: Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Similar Items: Find

Add

3

Image ID: 200003
Family: Carangidae
Genus: Caranx
Species: ignobilis
Common name: Giant Trevally
Collection: Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Similar Items: Find

Add

4

Image ID: 200004
Family: Sphyraenidae
Genus: Agrioposphyraena
Species: barracuda
Common name: Barracuda
Collection: Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Similar Items: Find

Add

5

Image ID: 200005
Family: Bothidae
Genus: Pseudorhombus
Species: arsius
Common name: Largetooth Flounder
Collection: Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Similar Items: Find

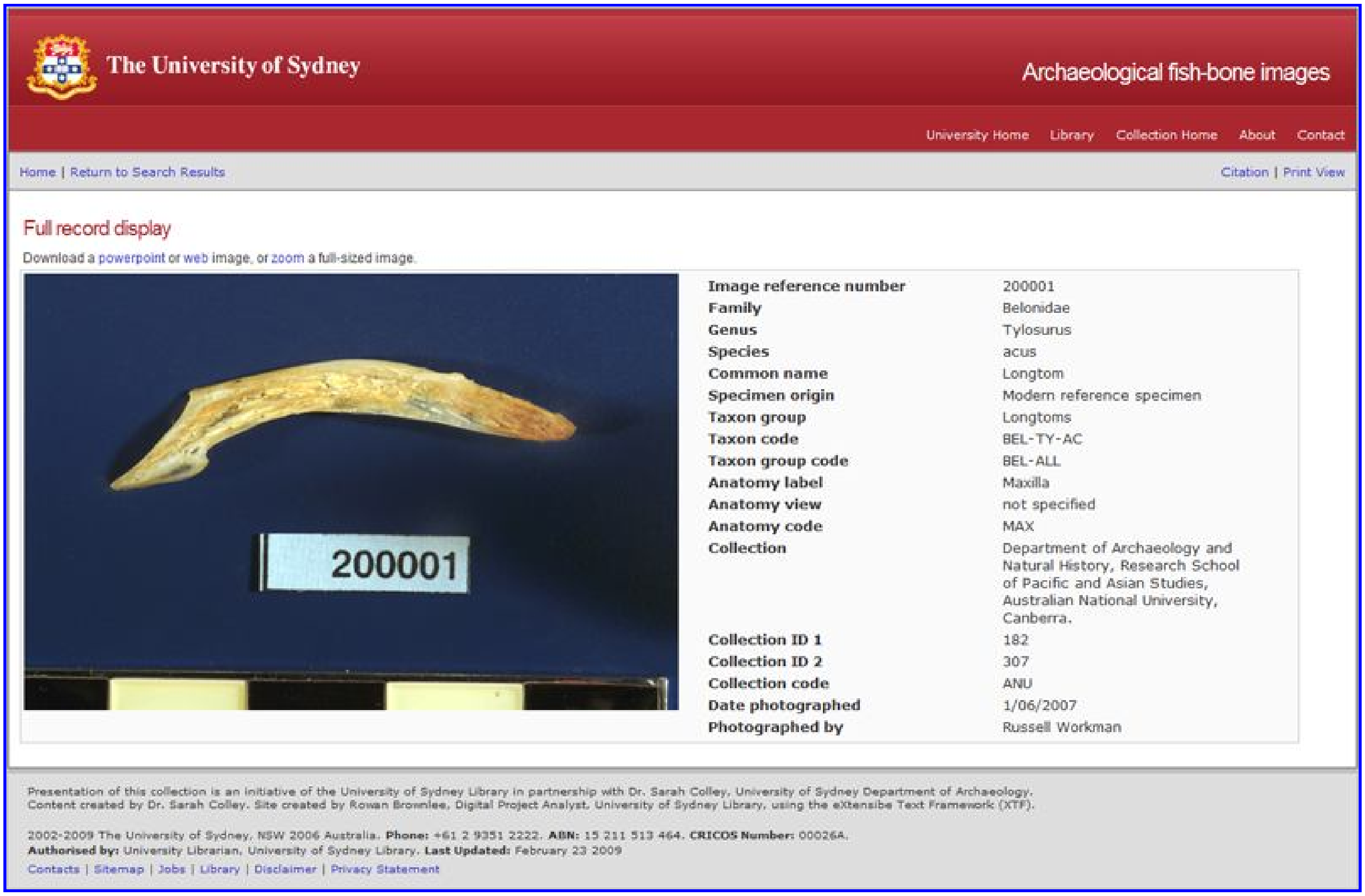
Add

6

Image ID: 200006
Family: Muraenidae

Add

Full record



Full record display

Download a powerpoint or web image, or zoom a full-sized image.

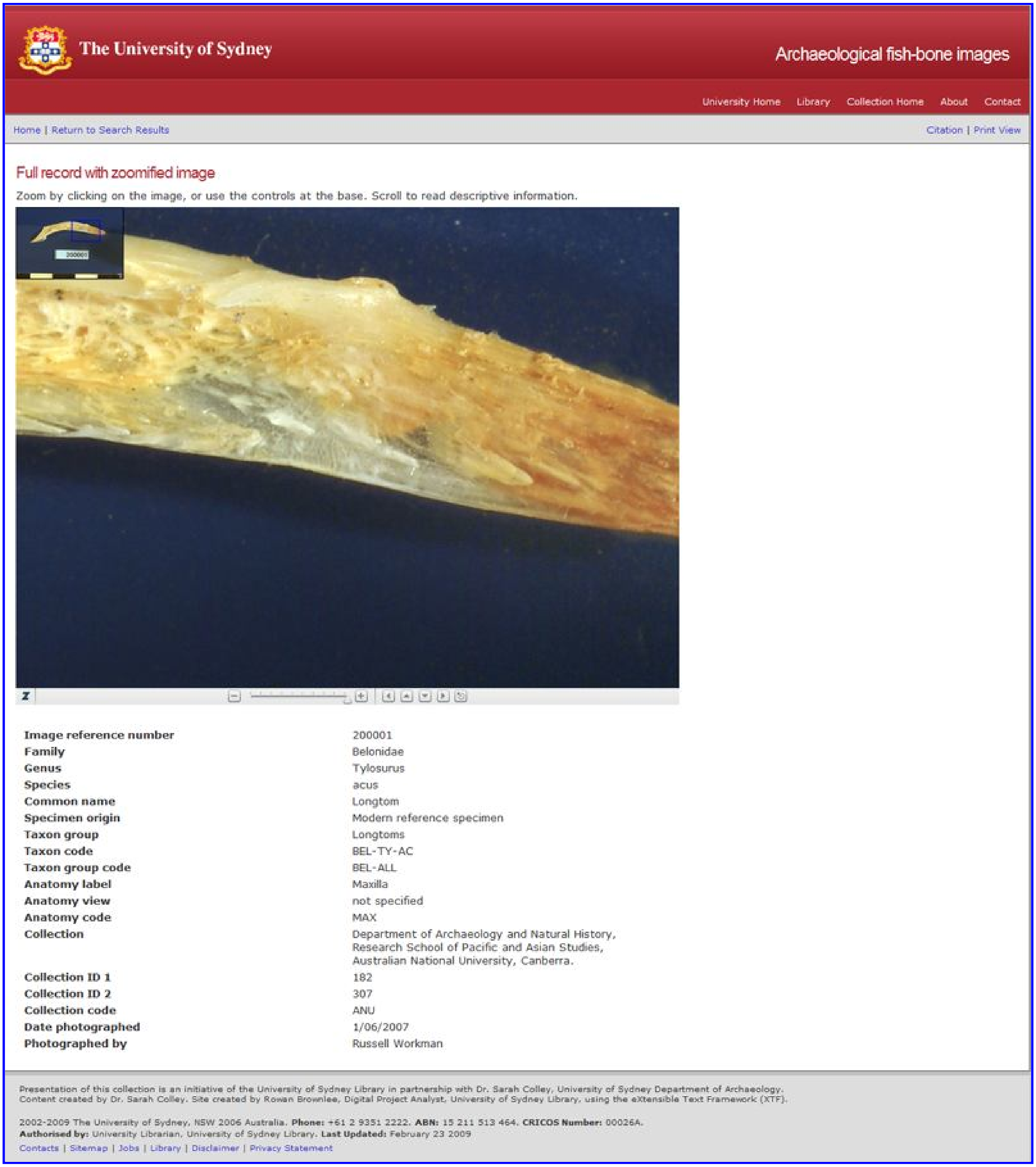
Image reference number	200001
Family	Belonidae
Genus	Tylosurus
Species	acus
Common name	Longtom
Specimen origin	Modern reference specimen
Taxon group	Longtoms
Taxon code	BEL-TY-AC
Taxon group code	BEL-ALL
Anatomy label	Maxilla
Anatomy view	not specified
Anatomy code	MAX
Collection	Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Collection ID 1	182
Collection ID 2	307
Collection code	ANU
Date photographed	1/06/2007
Photographed by	Russell Workman

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Dr. Sarah Colley, University of Sydney Department of Archaeology. Content created by Dr. Sarah Colley. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensibe Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Full record with zoomable image



Full record with zoomified image

Zoom by clicking on the image, or use the controls at the base. Scroll to read descriptive information.

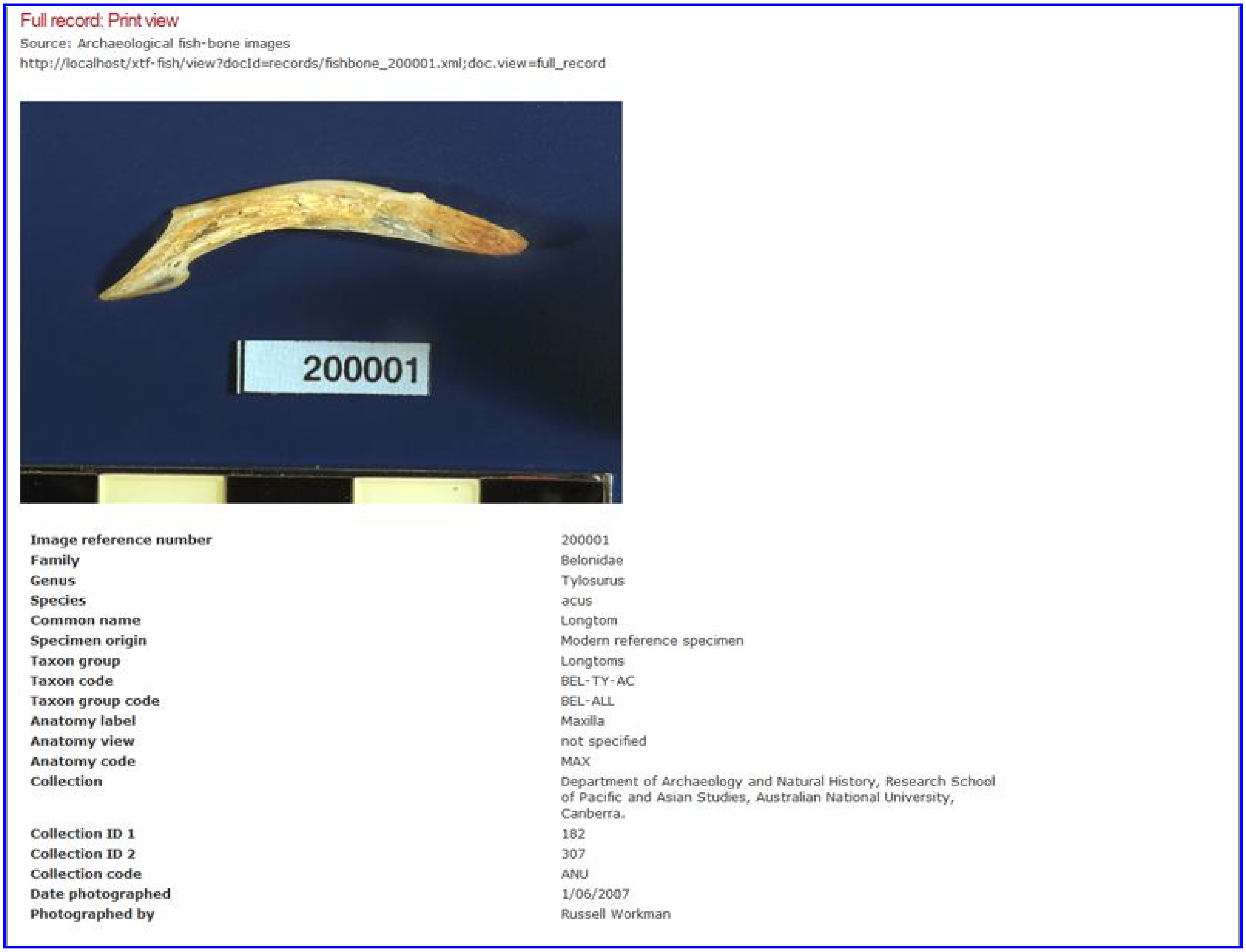
Image reference number	200001
Family	Belonidae
Genus	Tylosurus
Species	acus
Common name	Longtom
Specimen origin	Modern reference specimen
Taxon group	Longtoms
Taxon code	BEL-TY-AC
Taxon group code	BEL-ALL
Anatomy label	Maxilla
Anatomy view	not specified
Anatomy code	MAX
Collection	Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Collection ID 1	182
Collection ID 2	307
Collection code	ANU
Date photographed	1/06/2007
Photographed by	Russell Workman

Presentation of this collection is an initiative of the University of Sydney Library in partnership with Dr. Sarah Colley, University of Sydney Department of Archaeology. Content created by Dr. Sarah Colley. Site created by Rowan Brownlee, Digital Project Analyst, University of Sydney Library, using the eXtensible Text Framework (XTF).

2002-2009 The University of Sydney, NSW 2006 Australia. **Phone:** +61 2 9351 2222. **ABN:** 15 211 513 464. **CRICOS Number:** 00026A.
Authorised by: University Librarian, University of Sydney Library. **Last Updated:** February 23 2009

Contacts | Sitemap | Jobs | Library | Disclaimer | Privacy Statement

Full record print view



Full record: Print view

Source: Archaeological fish-bone images

http://localhost/xtf-fish/view?docId=records/fishbone_200001.xml;doc.view=full_record

Image reference number	200001
Family	Belonidae
Genus	Tylosurus
Species	acus
Common name	Longtom
Specimen origin	Modern reference specimen
Taxon group	Longtoms
Taxon code	BEL-TY-AC
Taxon group code	BEL-ALL
Anatomy label	Maxilla
Anatomy view	not specified
Anatomy code	MAX
Collection	Department of Archaeology and Natural History, Research School of Pacific and Asian Studies, Australian National University, Canberra.
Collection ID 1	182
Collection ID 2	307
Collection code	ANU
Date photographed	1/06/2007
Photographed by	Russell Workman

Next steps and future developments

- Harvesting and robots
- Advanced search
- Content management and relationships between content managment & presentation
- Facet paging
- PDFs and powerpoints
- Packaging and downloading images and related metadata
- Other …

Harvesting and robots

This isn't something I've investigated at all, though I understand that XTF provides support for robots and OAI harvesting. I'd like to enable harvesting and discovery by Google, while ensuring that media is viewed in the context of the XTF presentation.

Advanced search

In the advanced search section I mention a couple of eBot future enhancements which are applicable to presentation of the other collections. The current eBot advanced search is quite simple and could be developed to be far more extensive and flexible. I'd like to provide dropdown lists or automatic completion for fields that employ controlled vocabularies (such as family, genus and species). These classification elements are already related to each other through an extensive taxonomy underpinning the eBot collection, and the relationships might be exploited to provide additional support for searching. As an example, if a user wishes to search on a particular species, selection from the species category should trigger auto-population of the the family and genus fields. It would also be useful to enable the boolean 'OR' within categories such as *image type*. This would enable multiple selections from the list, to suit cases where people are interested in results comprising several image types such as *herbarium OR micrograph*.

Content management and relationships between content management & presentation

Following on from the advanced search comments above, I'd like to draw in content for dropdown lists and taxonomies from other sources. For each collection, metadata and taxonomies are maintained elsewhere by the content creators. At this stage I have a semi-automated approach to metadata and media import (for indexing and presentation), but I'm not yet importing controlled vocabularies for presentation on the search pages. For these three target collections, I'd like to automate as much as possible the process of content transfer.

Facet paging

On a brief record display screen, it's useful to be able to choose to display the first 'X' number of facets within a category (eg top 5). Having scanned the first 5 or 10, I'd like to be able to 'page' through successive sets, as an option to viewing all of the remaining facets in one hit. This may be useful in cases where there is an extensive facet list and where nesting of facets is not an option.

PDFs and powerpoints

It may be useful to enable generation of pdfs and powerpoints combining an image with its associated metadata.

Packaging and downloading images and related metadata

As an additional option to citation emailing (i.e. bookbag / my citations), it may be useful to enable downloading of packages of selected media and associated metadata.

Other…

Other activities will include investigation of alternate media presentation models and learning more about XML, XSL, text processing techniques and ways of ensuring common display across multiple browsers (eg via the Yahoo User Interface Library).

About the author

My name is Rowan Brownlee and I work in the eScholarship Division of the University of Sydney Library . As Digital Project Analyst I participate in activities supporting research, learning and teaching. Lead by Ross Coleman, eScholarship is a relatively new initiative which includes a digital archive and Sydney University Press. The Library also has considerable experience in text encoding through the work of Creagh Cole within SETIS (Sydney Electronic Text and Imaging Service).

I have been exploring the use of XTF for searching and presenting media collections in the subject areas of plant sciences, archaeology and visual arts. Although an XSLT beginner, the outcomes have been encouraging. I am interested in helping introduce others to XTF, so I have described my application of XTF to the SCA Archive, a collection of image and video reproductions of works created by staff and students at Sydney College of the Arts. My project partner is Jacqui Spedding, a ceramic artist and project manager for SCA Images Online.

Edit this entry.

Latest XTF News

- XTF 3.0 beta
- XTF Website Launched
- XTF Community Preview
- XTF 2.2 released

Subscribe to XTF News

- RSS

The **eXtensible Text Framework (XTF)** is supported by the California Digital Library