Documentation search functionality

These are the requirements for the live search function in the online documentation redesign. Following is a description of the search feature, and a set of implementation requirements.

The live search is based on an html textbox triggering an Ajax function in which searches an index for matching keywords. The results returned, will then populate a dynamic menu, displaying the most relevant hits.

Search box and dynamic menu

The search box will be a standard HTML text box located at the top of the menu. The rest of the menu will be a set of elements structuring the results into a set of categories.

ID:#DocHTML01	**Date:2/2/2010**
Requirement:	HTML element `<input type="text" name="docSearch" onkeyup="runSearch (this.value)" />`
Description:	This textbox is connected to an Ajax search feature triggered by the onkeyup attribute.
Depending on:	#DocAjax01
Changes:	

ID: #DocHTML02	**Date:2/2/2010**
Requirement:	HTML elements
Description:	These elements will be used to populate the menu with the search results. The menu will originally contain a set of default links when the search box in DocHTML01 is empty. The menu contains three main categories; • Id catAPIlookup (API lookup), which will contain links to overview pages like “All classes”, module/class reference page matching the search, etc. • Id catArticles (Articles), which will contain links to articles matching the search. This includes: best practices, core features, key technologies etc. • Id catExamples (Examples), which will link to examples matching the search. All elements populating the menu will contain title, URL and category. The categories will have a heading and nest the relevant links below the heading box. The categories will automatically provide a scrollbar in case of overpopulating.
Depending on:	#DocAjax01

Changes:	

Ajax search

The Ajax search will feed the search key-words to the server as they are typed into the search box. It will then receive the results and use these results to populate the menus.

ID:#DocAjax01	**Date:2/2/2010**
Requirement:	Ajax feature – search and populate
Description:	The search feature must start a HTTP Request and send the current search phrase to the server side search script. The script will return a result which the Ajax function inserts into the menu based on the result category. The link wrapping will be created on the client side to minimize the amount of data sent from the server to the client.
Depending on:	#DocPython01, #DocHTML02
Changes:	

Search script

The search script will traverse an XML-based index and look for nodes containing or matching the search phrases sent by the HTTP Request.

ID:#DocPython01	**Date:2/2/2010**
Requirement:	Python search script
Description:	The script will use an XML-based index and traverse this for keywords matching the query. On finding such matches, the script will return a result in xml format containing a link title, URL and category.
Depending on:	#DocAjax01, #DocIndex01
Changes:	

Search index

The search index is an XML structure generated by QDoc, used by the search script to locate results matching the search query.

ID:#DocIndex01	**Date:2/2/2010**
Requirement:	XML-based index mapping the documentation content
Description:	The XML structure is defined in the table below. Every keyword registered populates as nodes in the XML structure with: title, URL and category as sub-nodes.
Depending on:	DocQDoc01
Changes:	

XML structure and schema

XML structure

```
<?xml version="1.0" encoding="utf-8"?>
<qtKeys>
  <keyword id="0">
    <title>QWidget Class Reference</title>
    <url>qwidget.html</url>
    <category>classes</category>
  </keyword>
  <keyword id="1">
    <title>Animation Framework Examples</title>
    <url>examples-animation.html</url>
    <category>examples</category>
  </keyword>
  <keyword id="2">
    <title>The Animation Framework</title>
    <url>animation-overview.html</url>
    <category>overview</category>
  </keyword>
</qtKeys>
```

XML schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="qtKeys">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="keyword">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="title" type="xs:string" />
              <xs:element name="url" type="xs:string" />
              <xs:element name="category" type="xs:string" />
            </xs:sequence>
            <xs:attribute name="id" type="xs:unsignedByte" use="required" />
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

QDoc - requirements

QDoc will generate an XML file mapping keywords and matching information in the documentation.

ID:#DocQDoc01	Date:
Requirement:	Feature mapping keywords and generating an XML index
Description:	QDoc should traverse the documentation and add all keywords to an XML structure, formed as the one in #DocIndex01. Use of the QDoc command /keyword, should be used to map the documentation. This will have to be maintained manually in all documentation pages.
Depending on:	QDoc
Changes:	