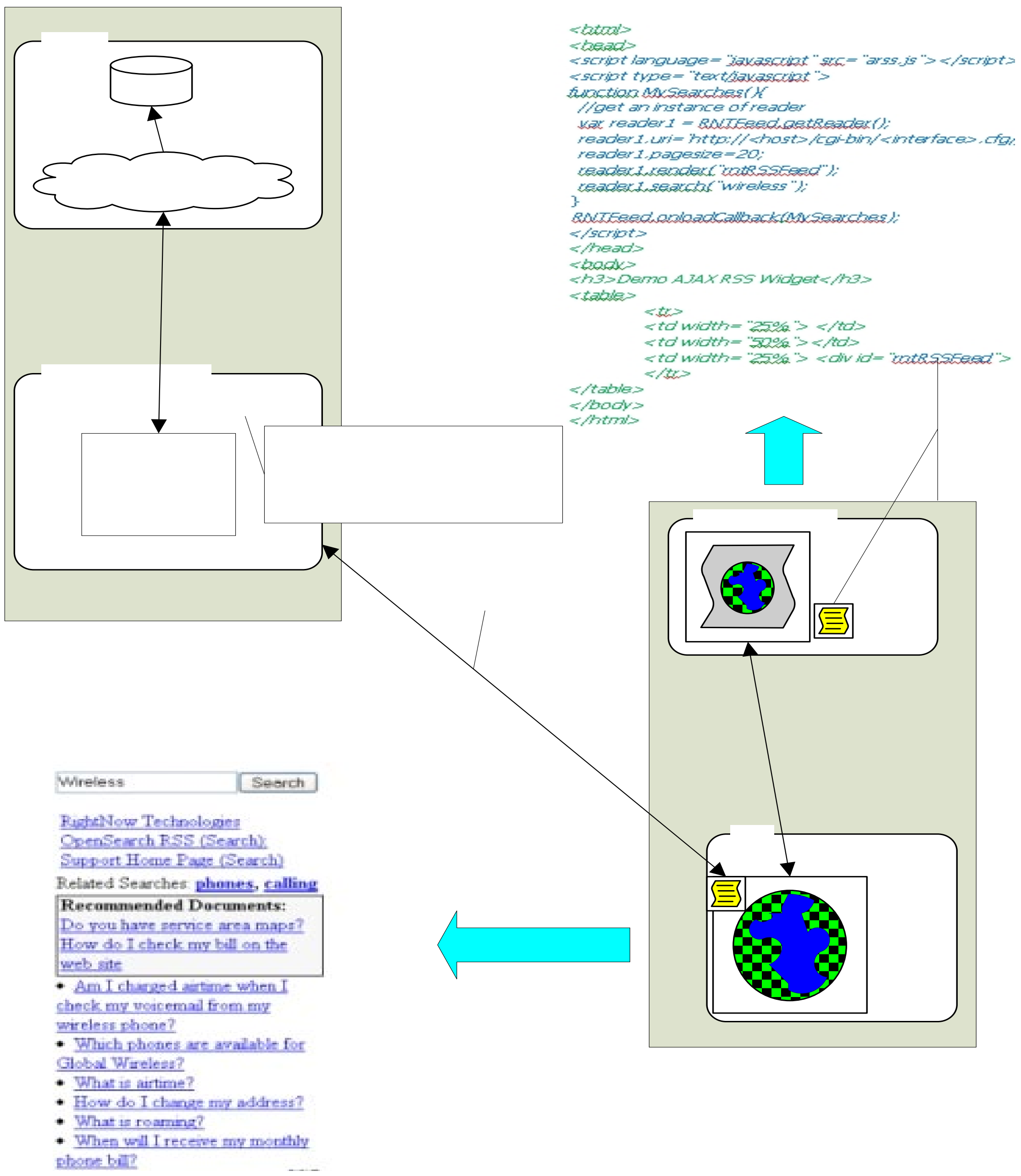
RightNow Knowledge Base Syndication Widget

This document describes how the knowledge syndication widget is setup and delivered. It uses a simple reference implementation to illustrate how the widget can be delivered, and discusses some of the options available to the web developer when deploying the widget. Note: Please contact your RightNow Account Executive about knowledge syndication widget pricing.



```
<html>
<head>
<script language="javascript" src="arss.js"></script>
<script type="text/javascript">
function MySearches(){
  //get an instance of reader
  var reader1 = RNTFeed.getReader();
  reader1.uri='http://<host>/cgi-bin/<interface>.cfg/
  reader1.pagesize=20;
  reader1.render("rntRSSFeed");
  reader1.search("wireless");
}
RNTFeed.onloadCallback(MySearches);
</script>
</head>
<body>
<h3>Demo AJAX RSS Widget</h3>
<table>
        <tr>
        <td width="25%"> </td>
        <td width="50%"></td>
        <td width="25%"> <div id="rntRSSFeed">
        </tr>
</table>
</body>
</html>
```

RightNow Technologies
OpenSearch RSS (Search)
Support Home Page (Search)
Related Searches: **phones**, **calling**

Recommended Documents:
Do you have service area maps?
How do I check my bill on the web site

- Am I charged airtime when I check my voicemail from my wireless phone?
- Which phones are available for Global Wireless?
- What is airtime?
- How do I change my address?
- What is roaming?
- When will I receive my monthly phone bill?

Figure 1: Reference Widget Deployment Architecture

How to install the widget:

To enable syndication of Enduser answers, the EU_SYNDICATION_ENABLE configuration verb needs to be set. The EU_SYNDICATION_ENABLE configuration verb is available in the “RightNow User Interface/End-User Interface” configuration settings section of the RightNow Management and Administration pages.

As seen in figure 1, the widget can be deployed to any target server to display syndication feeds in any web page. The following step by step instructions describe the process of copying the files from the RightNow development community and deployment and customization of the widget on your own web pages.

1) Copy all files (_arss.js, arss.js, arss.css, prototype.js, and a proxy file for your platform) to the target client machine’s web server. The proxy file is optional. See the Widget functionality section for details on where and when the proxy would need to be used.

2) Add the script reference to the page’s header section. For example, if the widget will be on a product page named product.html, the following reference needs to be in the header section of the page you will be using to display the widget.

   Example:

```
    <script language="javascript" src="arss.js"></script>
    <script type="text/javascript">

    function MySearches(){
                var reader1 = RNTFeed.getReader();
                reader1.uri='http://yellowstone.rightnowtech.com/cgi-
bin/htosun_current.cfg/php/enduser/opensearch.php?q=Wireless&startPage=1';
                //reader1.proxy='http://kaxta/wordpress/widget/opensearch_proxy.php';
                reader1.pagesize=20;
                reader1.render("rntRSSFeed");
                reader1.search("");


                var reader2 = RNTFeed.getReader();
                reader2.uri='http://yellowstone.rightnow.com/cgi-
bin/htosun_current.cfg/php/enduser/opensearch.php?';
                //reader2.proxy='http://kaxta/wordpress/widget/opensearch_proxy.php';
                reader2.pagesize=20;
                reader2.navigation="off";
                reader2.searchbox="off";
                reader2.render("rntRSSFeed2");
                reader2.search("phone");

       }
       RNTFeed.onloadCallback(MySearches);
  </script>
```

All properties of the script are defined in next section.

3) Add two div sections to your page. The above code has two searches; one will be displayed in the "rntRSSFeed" div section and the other will be displayed in the "rntRSSFeed2" div section. You can have as many instances of the widget on a page as you like – the reference implementation uses two widgets. In order to display the two widgets corresponding to each of the javascript functions referenced above, you web page needs to have these two div sections in the html body as shown in the appendix.

4) Update *arss.css* style sheet according to your page's style sheet.

Widget functionality (Refer to usage snippet in appendix):

1) **Field**: uri and proxy

   If the optional proxy is provided, the call will go through the proxy, and the format of the feed is in OpenSearch RSS 2.0 syndication format, *XML* . Otherwise, the feed response is in JSON (JavaScript Object Notation) format. For XML data, Ajax (Asynchronous Javascript And XML) model is used to request data from the server. Thus, it is subject to the browser's "same origin policy" where the widget can only request data originating from the same domain and over the same protocol. As such, if the XML format is chosen and enduser pages and the server that hosts the widget are on different domains, a proxy needs to be installed on the server where the widget is deployed. On the other hand, JSON methodology doesn't have such a limitation and it does not require a proxy to request data from a different domain.

   The uri must be a fully qualified url with all necessary parameters, such as product or category parameters. The uri will have the following structure:

   http://<rightnow site URL>/cgi-bin/<site config file>.cfg/php/enduser/opensearch.php?q=<optional keyword>&startPage=1&p_pv=<optional product constraint>&p_cv=<optional category constraint>'

   Consult with your RightNow administrator for the RightNow site URL and sit config file name. The most taxing part of the configuration URL is choosing the product and category value to constrain against, if a constraint is needed. When a product and/or category is selected from the RightNow enduser pages, the uri will include strings like p_pv=4.63&p_cv=2.75. p_pv specifies the product and p_cv specifies the category. The first number in the value of parameters indicates the hierarchy level. The number after the period is the id of the product or category. We strongly suggest that these values should be chosen on the enduser page when you are looking to add product and/or category as widget constraints. That is, the uri needs be constructed by selecting a product and category on the RightNow enduser search page and performing a search, where the values will be displayed in the resulting URL for you to copy and place in the Widget uri. The q parameter in the uri defines the search term (on a normal RightNow answer list page the search term is p_search_text). It will be automatically set in the uri when a search is done on the enduser page with a keyword entry.

2) **Field**: pagesize

   The widget provides pagination. The *pagesize* field can be set to change the default page size. The default page size is 20. Note: Setting this value to a value greater than the rows per page setting in your end user answers report within RightNow may have undesirable consequences on the behavior of the paging mechanisms handled by the navigation attribute (below).

3) **Field**: navigation
The user can turn off navigation buttons at the bottom of the widget by using the following JavaScript in the html header: “reader.navigation=’off’;”

4) **Field**: searchbox

The search box can be turned on and off. By default, the search box is on. The user can turn it off by using the following JavaScript in the html header: “reader.searchbox='off';”

5) **Method**: render

This method renders the search results. Its parameter is the name of the div section that will display the search results.

6) **Method**: search

This method carries out the searching. Its parameter is the default search term. If the search term is specified in a *uri*, the parameter can be an empty string (“”). You don’t need to add a search term at all (you may be constraining on product or category instead) – this method is optional. For more detail check the description of uri field.

Appendix

Usage snippet:

```
<html>
<head>
            <script language="javascript" src="arss.js"></script>
            <script type="text/javascript">

                        function MySearches(){

                        //get an instance of reader
                        var reader1 = RNTFeed.getReader();
                        reader1.uri='http://<host>/cgi-
bin/<interface>.cfg/php/enduser/opensearch.php?q=Wireless&startPage=1';
                        //reader1.proxy='http://kaxta/wordpress/widget/opensearch_proxy.php';
                        reader1.pagesize=20;
                        reader1.render("rntRSSFeed");
                        reader1.search("");

                        //second search: a new widget instance
                        var reader2 = RNTFeed.getReader();
                        reader2.uri='http://<host>/cgi-bin/h<interface>cfg/php/enduser/opensearch.php?';
                        //reader2.proxy='http://kaxta/wordpress/widget/opensearch_proxy.php';
                        reader2.pagesize=20;
                        reader2.navigation="off";
                        reader2.searchbox="off";
                        reader2.render("rntRSSFeed2");
                        reader2.search("phone");

                }
                RNTFeed.onloadCallback(MySearches);
  </script>

</head>
<body>
<h3>Demo AJAX RSS Widget</h3>
<table>
            <tr>
            <td width="25%"> <div id="rntRSSFeed"></div> </td>
            <td width="50%"></td>
            <td width="25%"> <div id="rntRSSFeed2"></div> </td>
            </tr>
</table>
</body>
</html>
```