EasyRDF

Mohammad Asif Jamaluddingari

Harshal Jhaveri

Name of the tool:	EasyRdf
Home page:	http://www.aelius.com/njh/easyrdf/
Programming Language(s) that can be used with the tool:	PHP
Relevant Semantic Web technologies:	RDF
Categories:	Programming Environment

Introduction:

EasyRdf is a PHP library designed to make it easy to consume and produce RDF. It was designed for use in mixed teams of experienced and inexperienced RDF developers. It is written in Object Oriented PHP.

During parsing EasyRdf builds up a graph of PHP objects that can then be walked around to get the data to be placed on the page.

Data is typically loaded into a EasyRdf_Graph object from source RDF documents. The source document could either be an RDF file on the web or the output of a Construct or Describe SPARQL query from a triplestore.

Requirements and Implementation:

EasyRdf contains all its features written in its php library. So it needs PHP 5.2.1.

In order to check out the features provided by the EasyRdf and to run the php programs from its library, we set up an environment by installing wamp server which already has the apache and php running in it, is made available at www.wampserver.com/en/download.php

Parsing Example:

The following example illustrated how the EasyRdf parses any kind of Rdf document.

Consider the RDF document present at http://www.bbc.co.uk/music/artists/70248960-cb53-4ea4-943a-edb18f7d336f.rdf

Suppose, we want to print the artist information from the above document. The following code is used to generate the artist info from the document. Each step of the code is described in the comments

```
set_include_path(get_include_path() . PATH_SEPARATOR . '../lib/');
```

//set_include_path sets the path for the rdf document which is to be parsed and get_include_path reads the file path and places it in the library.

```
 require_once "EasyRdf.php";
 require_once "html_tag_helpers.php";
```

//The above statement states that the program uses once the file EasyRdf.php and html_tag_helpers.php. The EasyRdf.php is described below

EasyRdf.php :

This file is used to load the core of EasyRdf if you don't have an autoloader. It includes many other files among which some of the important files are:

EasyRdf_Http_Response : Class that represents an HTTP 1.0 / 1.1 response message
EasyRdf_Graph : Container for collection of EasyRdf_Resources.
EasyRdf_Http_Client : This class is an implementation of an HTTP client in PHP.
EasyRdf_Literal : Class that represents an RDF Literal
EasyRdf_Parser_RdfPhp : Class to parse RDF with no external dependencies.
EasyRdf_Exception : All exceptions thrown by EasyRdf are an instance of this class.
EasyRdf_Namespace : A namespace registry and manipulation class.
EasyRdf_Parser : Parent class for the EasyRdf parsers.
EasyRdf_Parser_Ntriples : A pure-php class to parse N-Triples with no dependencies.
EasyRdf_Parser_Json : A pure-php class to parse RDF/JSON with no dependencies.
EasyRdf_Resource : Class that represents an RDF resource
EasyRdf_Serialiser : Parent class for the EasyRdf serialiser
EasyRdf_Serialiser_Json : Class to serialise an EasyRdf_Graph to RDF/JSON with no external dependencies.
EasyRdf_Serialiser_RdfXml : Class to serialise an EasyRdf_Graph to RDF/XML with no external dependencies.
EasyRdf_Serialiser_Turtle : Class to serialise an EasyRdf_Graph to Turtle with no external dependancies.EasyRdf_TypeMapper : Class to map between RDF Types and PHP Classes
EasyRdf_Utils: Class containing static utility functions.

```
 require_once "EasyRdf/Http/Client.php";
 EasyRdf_Graph::setHttpClient( new EasyRdf_Http_Client() );
```

//Configure the HTTP client to use

```
function birthEvent()
        {
            foreach ($this->all('bio:event') as $event) {
                if (in_array('bio:Birth', $event->types())) {
                    return $event;
                }
            }
            return null;
        }
```

//Function to iterate for each event in the document

```
function age()
        {
            $birth = $this->birthEvent();
            if ($birth) {
                $year = substr($birth->get('bio:date'), 0, 4);
                if ($year) {
                    return date('Y') - $year;
                }
            }
            return 'unknown';
```

//The above function reads the date of birth of the artist in the first step. And then takes the substring of the date of birth (i.e., only year is returned) and the returns the year.

```
    EasyRdf_Namespace::set('mo', 'http://purl.org/ontology/mo/');
    EasyRdf_Namespace::set('bio', 'http://purl.org/vocab/bio/0.1/');
    EasyRdf_TypeMapper::set('mo:MusicArtist', 'Model_MusicArtist');
```

//the above statements adds the namespaces. The detailed information of EasyRdf_Namespace is given below.

<u>EasyRdf_Namespace :</u>

This is a namespace registry and manipulation class. The methods included are:

(Note:- All the methods are static methods. For example, If there is delete, it means that *static method delete*)

Delete : Delete an existing namespace.
Expand : Expand a shortened URI (qname) back into a full URI.
Get : Return a namespace given its prefix.
Namespaces: Return all the namespaces registered
prefixofUri : Return the prefix namespace that a URI belongs to.
reset : Delete the anonymous namespaces and reset the counter to 0
set : Register a new namespace.
shorten : If $createNamespace is true, and the URI isn't part of an existing namespace, then EasyRdf will attempt to create a new namespace and use that namespace to shorten the URI (for example ns0:term).
If it isn't possible to shorten the URI, then null will be returned.

```
<?php echo form_tag(); ?>
<?php  echo text_field_tag('uri', 'http://www.bbc.co.uk/music/artists/70248960-cb53-4ea4-943a-edb18f7d336f.rdf', array('size'=>50)); ?>
<?php echo submit_tag();?>
<?php echo form_end_tag() ;?>
```

//The above code is normal php code. The first step opens a new form. Then a textfield is created with the uri. This is the place where the input rdf document is to be passed. The URI of the rdf documents is passed here. And as usual a sumit button and an end tag of the form is echoed.

```
   if (isset($_REQUEST['uri'])) {

       $graph = new EasyRdf_Graph( $_REQUEST['uri'] );
                   $graph ->load();
       if ($graph)
                   { $artist = $graph->primaryTopic();
          }
          }
```

//Here comes the major part. i.e., building the graph object for the input rdf document. The EasyRdf_Graph build the graph for the input rdf through which one can traverse to get the intended nodes or children.

EasyRdf_Graph:

This is the container for collection of EasyRdf_Resources. It has the following methods.

getHttpClient : Get the HTTP Client object used to fetch RDF data
setHttpClient : Set the HTTP Client object used to fetch RDF data
Constructor_construct ; If no URI is given then an empty graph is created.

If a URI is supplied, but no data then the data will be fetched from the URI.

The document type is optional and can be specified if it can't be guessed or got from the HTTP headers.
Add : The resource can either be a resource or the URI of a resource.
The properties can either be a single property name or an associate array of property names and values.
The value can either be a single value or an array of values.

Examples: $res = $graph->resource("http://www.example.com"); $graph->add($res, 'prefix:property', 'value'); $graph->add($res, 'prefix:property', array('value1',value2')); $graph->add($res, array('prefix:property' => 'value1')); $graph->add($res, 'foaf:knows', array('foaf:name' => 'Name')); $graph->add($res, array('foaf:knows' => array('foaf:name' => 'Name'));

allofType : Get all the resources in the graph of a certain type. If no resources of the type are available and empty array is returned.

dump : Return view of all the resources in the graph
This method is intended to be a debugging aid and will return a pretty-print view of all the resources and their properties.

getUri : Get the URI of the graph

load ; Load RDF data into the graph.
If a URI is supplied, but no data then the data will be fetched from the URI.
The document type is optional and can be specified if it can't be guessed or got from the HTTP headers.

newBnode : Create a new blank node in the graph and return it.
If you provide an RDF type and that type is registered with the EasyRdf_TypeMapper, then the resource will be an instance of the class registered.

primaryTopic : Get the primary topic of the graph

resource : Get or create a resource stored in a graph
If the resource did not previously exist, then a new resource will be created. If you provide an RDF type and that type is registered with the EasyRdf_TypeMapper, then the resource will be an instance of the class registered.

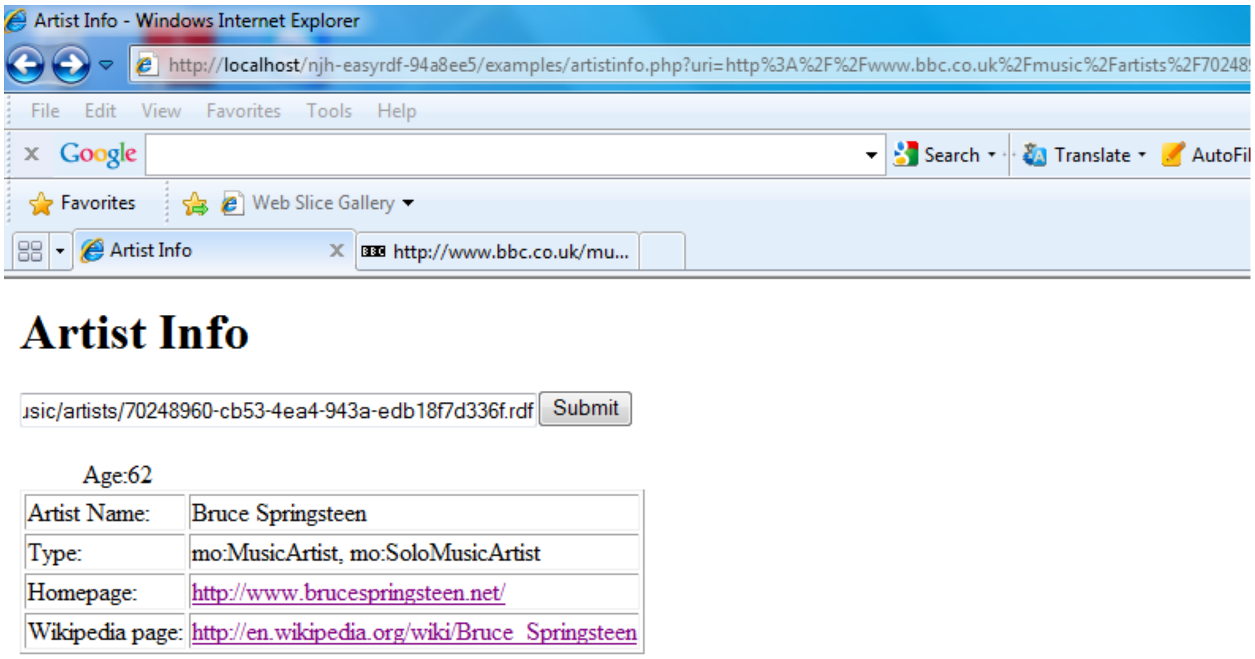
resources_matching : Get an arry of resources matching a certain property and value.
For example this routine could be used as a way of getting everyone who is male: $people = $graph->resourcesMatching('foaf:gender', 'male');

serialize : Serialise the graph into RDF

```
<table border="1">
          <tr><td>Artist Name:</td><td><?php echo $artist->get('foaf:name'); ?></td></tr>
   <tr><td>Type:</td><td><?php echo join(', ', $artist->types()) ;?></td></tr>
   <tr><td>Homepage:</td><td><?php echo link_to($artist->get('foaf:homepage'));
?></td></tr>
   <tr><td>Wikipedia page:</td><td><?php echo link_to($artist->get('mo:wikipedia'));
?></td></tr>
   <?php
      if ($artist->is_a('mo:SoloMusicArtist')) {
        // echo "  <dt>Age:</dt>";
         echo "  <dd>Age:" . $artist->age(). "</dd>\n";
      }
   ?></table>
```

//This is the normal php code in order to display the artist information in a tabular format.

Finally the output would be as in following screen


Artist Info - Windows Internet Explorer
http://localhost/njh-easyrdf-94a8ee5/examples/artistinfo.php?uri=http%3A%2F%2Fwww.bbc.co.uk%2Fmusic%2Fartists%2F70248
File Edit View Favorites Tools Help
Google
Search
Translate
AutoFil
Favorites
Web Slice Gallery
Artist Info
http://www.bbc.co.uk/mu...

Artist Info

usic/artists/70248960-cb53-4ea4-943a-edb18f7d336f.rdf Submit

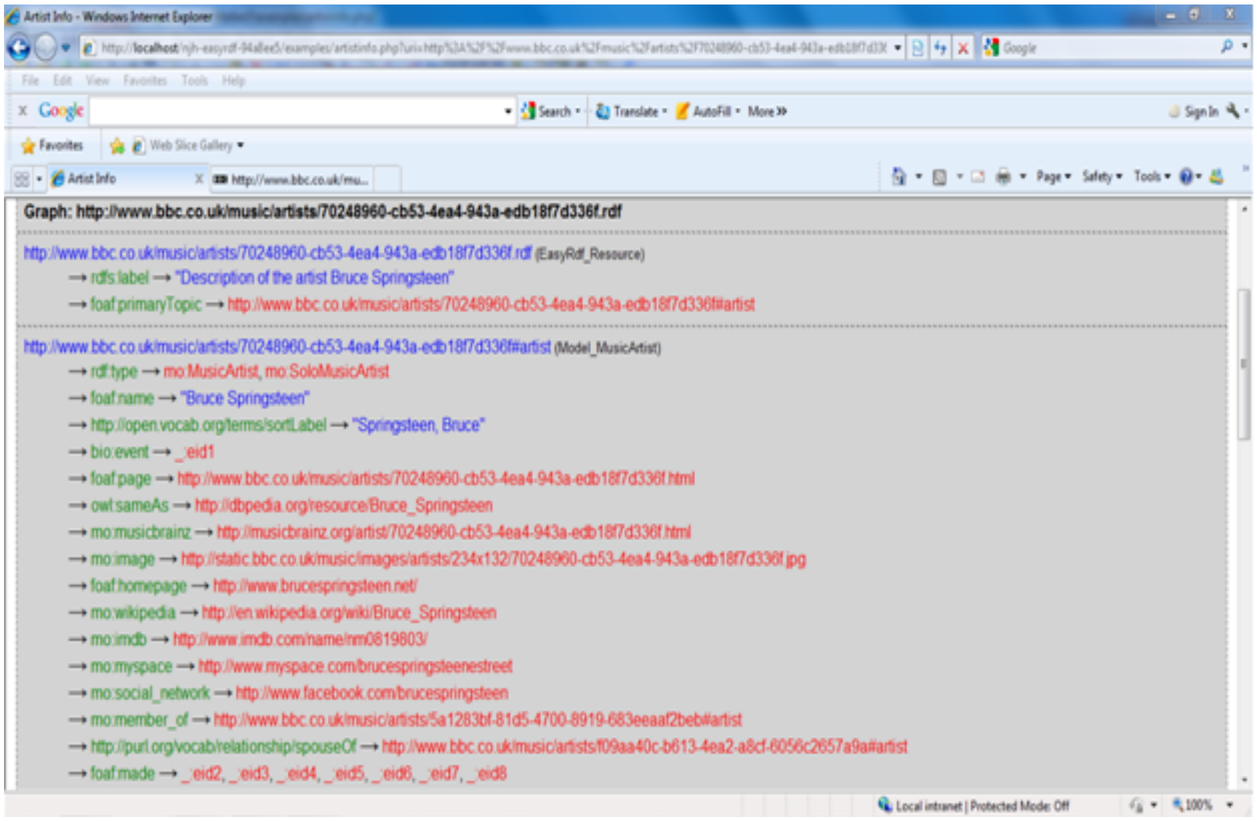
Age:62

Artist Name:	Bruce Springsteen
Type:	mo:MusicArtist, mo:SoloMusicArtist
Homepage:	http://www.brucespringsteen.net/
Wikipedia page:	http://en.wikipedia.org/wiki/Bruce_Springsteen

The graph created can be seen by just echoing the following line

echo $graph->dump();

//This will just print the dump of the graph as in following screenshot.


Artist Info - Windows Internet Explorer
File Edit View Favorites Tools Help
Search Translate AutoFill More
Favorites
Web Slice Gallery
Artist Info
Page Safety Tools
Local intranet | Protected Mode: Off
100%

```
Graph: http://www.bbc.co.uk/music/artists/70248960-cb53-4ea4-943a-edb18f7d336f.rdf

http://www.bbc.co.uk/music/artists/70248960-cb53-4ea4-943a-edb18f7d336f.rdf (EasyRdf_Resource)
        → rdfs:label → "Description of the artist Bruce Springsteen"
        → foaf:primaryTopic → http://www.bbc.co.uk/music/artists/70248960-cb53-4ea4-943a-edb18f7d336f#artist

http://www.bbc.co.uk/music/artists/70248960-cb53-4ea4-943a-edb18f7d336f#artist (Model_MusicArtist)
        → rdf:type → mo:MusicArtist, mo:SoloMusicArtist
        → foaf:name → "Bruce Springsteen"
        → http://open.vocab.org/terms/sortLabel → "Springsteen, Bruce"
        → bio:event → _:eid1
        → foaf:page → http://www.bbc.co.uk/music/artists/70248960-cb53-4ea4-943a-edb18f7d336f.html
        → owl:sameAs → http://dbpedia.org/resource/Bruce_Springsteen
        → mo:musicbrainz → http://musicbrainz.org/artist/70248960-cb53-4ea4-943a-edb18f7d336f.html
        → mo:image → http://static.bbc.co.uk/music/images/artists/234x132/70248960-cb53-4ea4-943a-edb18f7d336f.jpg
        → foaf:homepage → http://www.brucespringsteen.net/
        → mo:wikipedia → http://en.wikipedia.org/wiki/Bruce_Springsteen
        → mo:imdb → http://www.imdb.com/name/nm0819803/
        → mo:myspace → http://www.myspace.com/brucespringsteenestreet
        → mo:social_network → http://www.facebook.com/brucespringsteen
        → mo:member_of → http://www.bbc.co.uk/music/artists/5a1283bf-81d5-4700-8919-683eeaaf2beb#artist
        → http://purl.org/vocab/relationship/spouseOf → http://www.bbc.co.uk/music/artists/f09aa40c-b613-4ea2-a8cf-6056c2657a9a#artist
        → foaf:made → _:eid2, _:eid3, _:eid4, _:eid5, _:eid6, _:eid7, _:eid8
```

The example above we have seen is just a simple example of easyrdf to show the way of building up of the graph and parsing. It has lot of capabilities and functionalities which are defined in a number of methods in each class, inturn in number of classes defined in the php library.

The easyRDF is also capable of converting the file into different formats. For example, given a input RDF file, it can convert it into RDF/PHP, RDF/JSON Resource-centric, N-Triples, Turtle terse RDF triple language, RDF/XML, Notation3.

Though easyRDF has lot of advantages, the major disadvantage of this is that it is not flexible with the documents which have a large amount of data as it has to produce a big graph in which the traversing through such a big graph reduces the performance of the code. And hence, it is more flexible with small sized documents.