SDMX Sax Parser
An SDMX Parser Written in Java
Quick Start

SDMX Sax is client software for SDMX Data Services such as the ABS .Stat SDMX Data Service we will use the ABS Webservice as an example.

1. First you must obtain a DataProvider using

ServiceList.getDataProvider(int type, String agency,String serviceURL,String options,String attribution,String htmlAttribution)
Parameters;
Type: this is the 'driver code' used to connect to the service
DataProvider.TYPE_SDW = 0; // dotStat
DataProvider.TYPE_NSI = 1;   // Reference Infrastructure
DataProvider.TYPE_REST = 2; // SDMX 2.2 Rest Interface
DataProvider.TYPE_OPENSDMX = 3; // FAO SDMX Service
DataProvider.TYPE_ILO = 4; // International Labour Organisation Custom Service
DataProvider.TYPE_KNOEMA = 5; // Knoema SDMX Data Service

Agency: This is used as the SDMX Agency Code, some drivers use this parameter to look up resources (SDW, NSI,REST)

serviceURL: the Base URL of the data service
options: this is used only for SDW connections, and changes the SOAP Namespace used in the SOAP Message.

attribution: what should be displayed as an attribution message
htmlAttribution: same as attribution, possibility to include HTML codes in here.

once you have a DataProvider object.. you can obtain a 'Queryable' Object with
DataProvider.getQueryable();

with a sdmx.Queryable, you may obtain either a 'sdmx.Registry' or a 'sdmx.Repository' object with;
Queryable.getRegistry();
Queryable.getRepository();
Registry is an interface to structural data, Repository is an interface to the data itself.

The Registry interface has methods to list dataflows, find structural information such as codelists, concepts
The Repository interface has a single method;
Repository.query(DataQueryMessage q);
to query the repository of data.

Here is some example code to query the ABS SDMX Data Service;

```
public class Example {

   public static void main(String args[]) throws MalformedURLException {
      // Set this to true to dump the SOAP query to System.out that has been
      // created by the driver
      SdmxIO.setDumpQuery(false);
      DataProvider dp = ServiceList.getDataProvider(0, "ABS", "http://stat.abs.gov.au/sdmxws/sdmx.
asmx", "http://stats.oecd.org/OECDStatWS/SDMX/", "Based on Australian Bureau of Statistics data",
"Based on Australian Bureau of Statistics data");
      Queryable queryable = dp.getQueryable();
      Registry reg = queryable.getRegistry();
      Repository rep = queryable.getRepository();
      // List the dataflows on the server
      List<DataflowType> dataflows = reg.listDataflows();
      DataflowType flow = dataflows.get(0);
      // flow.getStructure() returns a DataStructureReference which is only a reference
      // a reference is just a set of IDs used to find the concrete instance of DataStructureType
      DataStructureType struct = reg.find(flow.getStructure());
      // struct contains information about this dataflow's structure

      DataQueryMessage query = new DataQueryMessage();
      // Ignore the Header, it will be filled in by the driver
      // You can set the Header fields yourself if you wish, and the driver
      // will not overwrite them
      DataQuery q = new DataQuery();
      DataParametersAndType dw = new DataParametersAndType();
      List<DataParametersOrType> ors = new ArrayList<DataParametersOrType>();
      // This sets which cube we want to query...
      // some queryables fudge the dataflow name (like SDW)
      // as SDW does not have dataflows, only a list of datastructures
      dw.setDataflow(Collections.singletonList(flow.asReference()));
      // year-month-day
      dw.setTimeDimensionValue(Collections.singletonList(new TimeDimensionValueType(new Time-
Value("2010-01-01"), new TimeValue("2014-01-01"))));
      for (int i = 0; i < struct.getDataStructureComponents().getDimensionList().size(); i++) {
         DataParametersOrType or = new DataParametersOrType();
         List<DimensionValueType> dims = new ArrayList<DimensionValueType>();
         CodelistType codes1 = reg.find(struct.getDataStructureComponents().getDimensionList().get-
Dimension(i).getLocalRepresentation().getEnumeration().asCodelistReference());
         dims.add(new DimensionValueType(struct.getDataStructureComponents().getDimensionList().
getDimension(i).getId().toString(), codes1.getItem(0).getId().toString()));
         // If there is more than 1 code in the codelist, include the second code too!
         if (codes1.size() > 1) {
            dims.add(new DimensionValueType(struct.getDataStructureComponents().getDimension-
List().getDimension(i).getId().toString(), codes1.getItem(1).getId().toString()));
         }
         or.setDimensionValue(dims);
         ors.add(or);
      }
      dw.setOr(ors);
      DataParametersType dpt = new DataParametersType();
      // Some Providers require another "AND" query element to be under the main DataParameters-
Type(which is an And)
      // so we put everything into dw and set it here
      dpt.setAnd(Collections.singletonList(dw));
      q.setDataWhere(dpt);
      query.setQuery(q);
      long t3 = System.currentTimeMillis();
      DataMessage dm = rep.query(query);
      long t4 = System.currentTimeMillis();
      System.out.println("Got CompactData " + dm.getDataSets().get(0).size() + " observations " + (t4 - t3) +
" ms");
// Dump the dataset to System.out
// At the moment, this is just the raw code ID's for the dataset
// if you know the codes, this is fine, but i like to use the
// the internationalised strings from the codelist, rather than ID's
      dm.dump();
      // This just returns a string...
      // this is what's in the data message... data with no structure..
      System.out.println("Value at 0,0 is:" + dm.getDataSets().get(0).getValue(0, 0));
      System.out.println("Column name at 0,0 is:" + dm.getDataSets().get(0).getColumnName(0));
      // Structured Data Message needs a registry to find Structural Data
      dm.setDataStructure(flow.getStructure(),null);
      StructuredDataMessage sdm = new StructuredDataMessage(dm, reg);
      StructuredDataSet sds = sdm.getStructuredDataSet(0);
      // Get the value in the top left hand corner of the dataset at 0,0
      StructuredValue value = sds.getStructuredValue(0, 0);
      // Here we have access to the Codelist from the particular item in the dataset..
      // this is where we can get internationalised strings from inside 'findName'
      System.out.println("Code is:" + value.getCode().getName());
      System.out.println("Column Concept is:"+value.getConcept().getName());
      System.out.println("Other valid codes;" + value.getCodelist().getItems());

   }
}
```