x12-Parser Documentation

Release 1.0.1

Prasad Balan, Ryan Colwell

January 31, 2016

Contents

ASC X12 standards define one of the widely used EDI formats. This library enables easy parsing and creation of X12 transactions.

Parse X12 transactions (With Loop identification introduced with version 0.7).

See example X12 Configuration for Loop Detection

Create X12 transactions.

Convert X12 to XML.

See Javadoc for more details.

CHAPTER 1

Developer

1.1 x12-Parser Overview

ASC X12 standards define one of the widely used EDI formats. This library enables easy parsing and creation of X12 transactions.

Parse X12 transactions (With Loop identification introduced with version 0.7).

See example X12 Configuration for Loop Detection

Create X12 transactions.

Convert X12 to XML.

See Javadoc for more details.

1.1.1 Explanation of Object Types

- **Cf** - class represents a configuration element. Each Cf instance represents items required to identify a Loop in a X12 transaction. Some Loops can be identified by only the segment id. Others require segment id and additional qualifiers to be able to identify the Loop.
- **Context** - The class represents an X12 context. A X12 context consists of a segment separator, element separator and a composite element separator.
- **Loop** - The Loop class is the representation of an Loop in a ANSI X12 transaction. Segments are grouped as loops. And a set of loops form an X12 transaction.
- **Parser**
- **Segment** - An aggregation of elements.
- **Element** - The building block of an X12 transaction is an element. Some elements may be made of sub elements.

1.1.2 Limitations

This library cannot validate a X12 transaction. When parsing, reads the whole X12 transaction into memory.

1.1.3 Current Version

- V1.0 - 21st May 2013

- V0.9 - 16th Apr 2011

1.2 Parse X12 file and loop over segments. (without loop identification)

1.2.1 Introduction

Example showing Parser reading a X12 file and looping over the segments.

1.2.2 Details

```
X12Simple x12 = (X12Simple) new X12SimpleParser().parse(new File("C:\\test\\835.txt"));
for (Segment s : x12)
{
        if (s.getElement(0).equals("CLP")) {
                System.out.println("Total Change Amount " + s.getElement(3));
        }
}
```

1.3 X12 Configuration for Loop Detection

1.3.1 Introduction

To enable the X12 parser to detect loops, a configuration object (Cf) needs to be created with the Loop details. Configuration object is created based on the X12 transaction to be parsed. Different variations of the configuration object can be created allowing Parser to parse the X12 file to meet the specific needs.

1.3.2 Simple/Flattened Hierarchy

Here is a sample 835 hierarchy which is flattened. Some may not be interested in the hierarchy of the loops.

```
+--X12
|  +--ISA    - ISA
|  +--GS     - GS
|  +--ST     - ST   - 835, - 1
|  +--1000A - N1   - PR,   - 1
|  +--1000B - N1   - PE,   - 1
|  +--2000   - LX
|  +--2100   - CLP
|  +--2110   - SVC
|  +--SE     - SE
|  +--GE     - GE
|  +--IEA    - IEA
```

Format: LoopName - SegmentId - SegmentQualifier - SegmentQualifierPosition Note: Separate multiple qualifiers with a COMMA.

```
Cf cfX12 = new Cf("X12");

cfX12.addChild("ISA", "ISA");
cfX12.addChild("GS", "GS");
cfX12.addChild("ST", "ST", "835", 1);
cfX12.addChild("1000A", "N1", "PR", 1);
cfX12.addChild("1000B", "N1", "PE", 1);
cfX12.addChild("2000", "LX");
cfX12.addChild("2100", "CLP");
cfX12.addChild("2110", "SVC");
cfX12.addChild("GE", "GE");
cfX12.addChild("IEA", "IEA");

System.out.println(cfX12);
```

1.3.3 Actual Hierarchy

Here is a sample 835 hierarchy with the actual loop hierarchy.

```
+--X12
|  +--ISA                - ISA
|  |  +--GS              - GS
|  |  |  +--ST           - ST   - 835, - 1
|  |  |  |  +--1000A     - N1   - PR,  - 1
|  |  |  |  +--1000B     - N1   - PE,  - 1
|  |  |  |  +--2000      - LX
|  |  |  |  |  +--2100   - CLP
|  |  |  |  |  |  +--2110 - SVC
|  |  |  +--SE           - SE
|  |  +--GE              - GE
|  +--IEA                - IEA
```

Format: LoopName - SegmentId - SegmentQualifier - SegmentQualifierPosition Note: Separate multiple qualifiers with a COMMA.

Example of how to create a configuration object for the above hierarchy.

```
Cf cfX12 = new Cf("X12");

Cf cfISA = cfX12.addChild("ISA", "ISA");
Cf cfGS = cfISA.addChild("GS", "GS");
Cf cfST = cfGS.addChild("ST", "ST", "835", 1);

cfST.addChild("1000A", "N1", "PR", 1);
cfST.addChild("1000B", "N1", "PE", 1);

Cf cf2000 = cfST.addChild("2000", "LX");

Cf cf2100 = cf2000.addChild("2100", "CLP");
cf2100.addChild("2110", "SVC");

cfISA.addChild("GE", "GE");
cfX12.addChild("IEA", "IEA");

System.out.println(cfX12);
```

1.4 X12 Parsing with Loop detection.

1.4.1 Introduction

Example of parsing a X12 file.

1.4.2 Details

There are two steps:

1. Load/Create the X12 configuration object (based on the X12 transaction) to be parsed.
2. Parse the X12 file.

1.4.3 Load/Create a configuration object

Example of creating an X12 configuration object for a 835 transaction.

```
private static Cf loadCf() {

        Cf cfX12 = new Cf("X12");

        Cf cfISA = cfX12.addChild("ISA", "ISA");
        Cf cfGS = cfISA.addChild("GS", "GS");
        Cf cfST = cfGS.addChild("ST", "ST", "835", 1);

        cfST.addChild("1000A", "N1", "PR", 1);
        cfST.addChild("1000B", "N1", "PE", 1);

        Cf cf2000 = cfST.addChild("2000", "LX");

        Cf cf2100 = cf2000.addChild("2100", "CLP");
        cf2100.addChild("2110", "SVC");

        cfISA.addChild("GE", "GE");
        cfX12.addChild("IEA", "IEA");

        //System.out.println(cfX12);
        return cfX12;
}
```

For more details on creating configuration object check Creating X12 configuration https://code.google.com/p/x12-parser/wiki/exampleConfigurationLoading.

1.4.4 Parse the X12 file

```
// The configuration Cf can be loaded using DI framework.
// Check the sample spring application context file provided.

Cf cf835 = loadCf();
Parser parser = new X12Parser(cf835);
X12 x12 = (X12) parser.parse(new File("C:\\test\\835.txt"));
```

```
Double totalChargeAmount = 0.0;

// Calculate the total charge amount
List<Loop> loops = x12.findLoop("2100");

for (Loop loop : loops) {
    for (Segment s : loop) {
        if (s.getElement(0).equals("CLP")) {
            totalChargeAmount = totalChargeAmount + Double.parseDouble(s.getElement(3));
        }
    }
}
System.out.println("Total Charged Amount = " + totalChargeAmount.toString());```


Don't need the loops. Alternate way.
```
Double totalChargeAmount = 0.0;

List segments = x12.findSegment("CLP");

for (Segment s : segments) {
    totalChargeAmount = totalChargeAmount + Double.parseDouble(s.getElement(3));
}
System.out.println("Total Charged Amount = " + totalChargeAmount.toString());
```
```

CHAPTER 2

API

2.1 Javadoc

2.1.1 com.yarsquidy.x12.example

exampleCreateX12One

public class **exampleCreateX12One**

Example showing how to create a X12 transaction from scratch.

Author Prasad Balan

```
Context c = new Context(`~', `*', `:');
X12 x12 = new X12(c);
Loop loop_isa = x12.addChild(``ISA'');

// Add ISA segment to the loop
loop_isa.addSegment(``ISA*00*          00          ZZ*SENDERID       *''
        + ``ZZ*RECEIVERID    *030409*0701*U*00401*0000000001*0*T:'');

// Add GS child loop to ISA loop
Loop loop_gs = loop_isa.addChild(``GS'');
// Add GS segment directly as a string
loop_gs.addSegment(``GS*1212*SENDERID*RECEIVERID*0701*000000001*X*00401'');

Loop loop_st = loop_gs.addChild(``ST'');
loop_st.addSegment(``ST*835*000000001'');
loop_st.addSegment(``BPR*DATA*NOT*VALID*RANDOM*TEXT'');
loop_st.addSegment(``TRN*1*0000000000*1999999999'');
loop_st.addSegment(``DTM*111*20090915'');

Loop loop_1000A = loop_st.addChild(``1000A'');
loop_1000A.addSegment(``N1*PR*ALWAYS INSURANCE COMPANY'');
loop_1000A.addSegment(``N7*AROUND THE CORNER'');
loop_1000A.addSegment(``N4*SHINE CITY*GREEN STATE*ZIP'');
loop_1000A.addSegment(``REF*DT*435864864'');

Loop loop_1000B = loop_st.addChild(``1000B'');
loop_1000B
        .addSegment(``N1*PE*FI*888888888*P.O.BOX 456*SHINE CITY*GREEN STATE*ZIP*EARTH'
```

```
Loop loop_2000 = loop_st.addChild(``2000'');
loop_2000.addSegment(``LX*1'');

Loop loop_2010_1 = loop_2000.addChild(``2010'');
loop_2010_1.addSegment(``CLP*PCN123456789**5555.55**CCN987654321'');
loop_2010_1.addSegment(``CAS*PR*909099*100.00'');
loop_2010_1.addSegment(``NM1*QC*1*PATIENT*TREATED*ONE***34*333333333'');
loop_2010_1.addSegment(``DTM*273*20020824'');
loop_2010_1.addSegment(``AMT*A1*10.10'');
loop_2010_1.addSegment(``AMT*A2*20.20'');

Loop loop_2010_2 = loop_2000.addChild(``2010'');
loop_2010_2.addSegment(``LX*2'');
loop_2010_2.addSegment(``CLP*PCN123456789**4444.44**CCN987654321'');
loop_2010_2.addSegment(``CAS*PR*909099*200.00'');
loop_2010_2.addSegment(``NM1*QC*1*PATIENT*TREATED*TWO***34*444444444'');
loop_2010_2.addSegment(``DTM*273*20020824'');
loop_2010_2.addSegment(``AMT*A1*30.30'');
loop_2010_2.addSegment(``AMT*A2*40.40'');

Loop loop_se = loop_gs.addChild(``SE'');
loop_se.addSegment(``SE*XX*000000001'');

Loop loop_ge = loop_isa.addChild(``GE'');
loop_ge.addSegment(``GE*1*000000001'');

Loop loop_iea = x12.addChild(``IEA'');
loop_iea.addSegment(``IEA*1*000000001'');

// Since the SE loop has the incorrect segment count let us fix that.
Integer count = loop_st.size();
count += 1; // In the loop hierarchy SE is not a child loop of ST. So
// when we get the rows in ST loop it does not have the count of SE.
// so add 1.

// We can set the count directly, like
// loop_se.getSegment(0).setElement(1, count.toString());
// this is just to show how to use the findLoop()
List<Loop> trailer = x12.findLoop(``SE'');
trailer.get(0).getSegment(0).setElement(1, count.toString());

//another way
List<Segment> se = x12.findSegment(``SE'');
se.get(0).setElement(1, count.toString());

//another way
loop_se.getSegment(0).setElement(1, count.toString());

System.out.println(loop_st.size());
System.out.println(x12.toString());
System.out.println(x12.toXML());
```

Methods

main
public static void **main** (String[] *args*)

exampleCreateX12SimpleOne

public class **exampleCreateX12SimpleOne**
Example showing how to create a X12 transaction from scratch.

Author Prasad Balan

```
Example of creating a X12 transaction

//Create a X12 context. Set the segment, element and sub-element separator
Context c = new Context('~','*',':');

//Create a X12 passing the context
X12Simple x = new X12Simple(c);

//Add a segment to the X12
Segment s = x.addSegment();

//Add elements to the segment
s.addElement("ISA");
s.addElement("00");
...
//Continue to add segments and then add elements
s = x.addSegment();
s.addElement("GS");
s.addElement("121");
...

//Convert X12Simple object to X12 string representation
String x12 = x.toString();

//Convert X12 object to XML string representation if that is what you need
String xml = x.toXML();
```

Methods

main
public static void **main** (String[] *args*)

exampleCreateX12SimpleTwo

public class **exampleCreateX12SimpleTwo**
Example showing how to create a X12 transaction from scratch.

Author Prasad Balan

```
Example of creating a X12 transaction

//Create a X12 context. Set the segment, element and sub-element separator
Context c = new Context(`~','*',':');
```

```
//Create a X12 transaction passing the context
X12Simple x = new X12Simple(c);

//Add a segments to the X12 transaction
Segment s = x.addSegment(``ISA*00*          00      .... T*:'');
s = x.addSegment(``GS*1212*SENDERID* ... X*00401'');
s = x.addSegment(``ST*835*000000000'');

//Modify one of the elements in ST segment just added
s.setElement(2,''000000001'');

//Add an empty segment and then add elements
s = x.addSegment();
s.addElement(`BPR');
s.addElement(`DATA');
s.addElement(`NOT');
...
...
//Convert X12 object to string representation
String x12 = x.toString();

//Convert X12 object to XML string representation if that is what you need
String xml = x.toXML();
```

Methods

main
public static void **main** (String[] *args*)

exampleParseX12FileOne

public class **exampleParseX12FileOne**
Example showing X12 Parser reading a X12 file and looping over the segments.

Author Prasad Balan

```
Example of parsing a X12 file

This is the loop hierarchy of a 835 transaction used here.

+--X12
|  +--ISA - ISA
|  |  +--GS - GS
|  |  |  +--ST - ST - 835, - 1
|  |  |  |  +--1000A - N1 - PR, - 1
|  |  |  |  +--1000B - N1 - PE, - 1
|  |  |  |  +--2000 - LX
|  |  |  |  |  +--2100 - CLP
|  |  |  |  |  |  +--2110 - SVC
|  |  |  +--SE - SE
|  |  +--GE - GE
|  +--IEA - IEA
```

```
Cf cf835 = loadCf();
Parser parser = new X12Parser(cf835);
// The configuration Cf can be loaded using DI framework.
// Check the sample spring application context file provided.

Double totalChargeAmount = 0.0;
X12 x12 = (X12) parser.parse(new File(``C:\test\835.txt''));
List<Segment> segments = x12.findSegment(``CLP'');
for (Segment s : segments) {
    totalChargeAmount = totalChargeAmount + Double.parseDouble(s.getElement(3));
}
System.out.println(``Total Change Amount '' + s.getElement(3));
```

```
Example of how to create a configuration object for the above hierarchy

private static Cf loadCf() {
    Cf cfX12 = new Cf("X12");
    Cf cfISA = cfX12.addChild("ISA", "ISA");
    Cf cfGS = cfISA.addChild("GS", "GS");
    Cf cfST = cfGS.addChild("ST", "ST", "835", 1);
    cfST.addChild("1000A", "N1", "PR", 1);
    cfST.addChild("1000B", "N1", "PE", 1);
    Cf cf2000 = cfST.addChild("2000", "LX");
    Cf cf2100 = cf2000.addChild("2100", "CLP");
    cf2100.addChild("2110", "SVC");
    cfISA.addChild("GE", "GE");
    cfX12.addChild("IEA", "IEA");
    //System.out.println(cfX12);
    return cfX12;
}
```

Methods

main
public static void **main** (String[] *args*)

exampleParseX12FileThree

public class **exampleParseX12FileThree**
Example showing X12 Parser reading a X12 file and looping over the segments.

Author Prasad Balan

```
Example of parsing a X12 InputStream.

This is the modified loop hierarchy of a 835 transaction used in this example.
The original/actual hierarchy is in example exampleParseX12FileTwo.
This just illustrates different ways you can setup the hierarchy to achieve
the desired results.
Note: Such a hierarchy change will work only when there is not more than
one Loop with the same identifiers. For e.g. in an 837 transaction both
Loop 2010BA (Subscriber) and 2330A (Other Subscriber) have the same identifiers,
which are segment id NM1 and IL at position NM102 (or index 1). This might
cause the parser to identify both elements as the same loop. In such cases
```

```
it is advisable to maintain the hierarchy.

 +--X12
 |  +--ISA - ISA
 |  +--GS - GS
 |  +--ST - ST - 835, - 1
 |  +--1000A - N1 - PR, - 1
 |  +--1000B - N1 - PE, - 1
 |  +--2000 - LX
 |  +--2100 - CLP
 |  +--2110 - SVC
 |  +--SE - SE
 |  +--GE - GE
 |  +--IEA - IEA

Cf cf835 = loadCf();
Parser parser = new X12Parser(cf835);
// The configuration Cf can be loaded using DI framework.
// Check the sample spring application context file provided.

Double totalChargeAmount = 0.0;
X12 x12 = (X12) parser.parse(new File(``C:\test\835.txt''));
List<Segment> segments = x12.findSegment(``CLP'');
for (Segment s : segments) {
    totalChargeAmount = totalChargeAmount + Double.parseDouble(s.getElement(3));
}
System.out.println(``Total Change Amount '' + s.getElement(3));
```

```
// Simple configuration, single level hierarchy
// Alternately can be loaded using Spring/DI
private static Cf loadCf() {
        Cf cfX12 = new Cf("X12");
        cfX12.addChild("ISA", "ISA");
        cfX12.addChild("GS", "GS");
        cfX12.addChild("ST", "ST", "835", 1);
        cfX12.addChild("1000A", "N1", "PR", 1);
        cfX12.addChild("1000B", "N1", "PE", 1);
        cfX12.addChild("2000", "LX");
        cfX12.addChild("2100", "CLP");
        cfX12.addChild("2110", "SVC");
        cfX12.addChild("GE", "GE");
        cfX12.addChild("IEA", "IEA");

        //System.out.println(cfX12);
        return cfX12;
}
```

Methods

main
public static void **main** (String[] *args*)

exampleParseX12FileTwo

public class **exampleParseX12FileTwo**
Example showing X12 Parser reading a X12 file and looping over the segments.

Author Prasad Balan

```
Example of parsing a X12 file

This is the modified loop hierarchy of a 835 transaction used in this example.
The original/actual hierarchy is in example exampleParseX12FileTwo.
This just illustrates different ways you can setup the hierarchy to achieve
the desired results.
Note: Such a hierarchy change will work only when there is not more than
one Loop with the same identifiers. For e.g. in an 837 transaction both
Loop 2010BA (Subscriber) and 2330A (Other Subscriber) have the same identifiers,
which are segment id NM1 and IL at position NM102 (or index 1). This might
cause the parser to identify both elements as the same loop. In such cases
it is advisable to maintain the hierarchy.

 +--X12
 |  +--ISA - ISA
 |  +--GS - GS
 |  +--ST - ST - 835, - 1
 |  +--1000A - N1 - PR, - 1
 |  +--1000B - N1 - PE, - 1
 |  +--2000 - LX
 |  +--2100 - CLP
 |  +--2110 - SVC
 |  +--SE - SE
 |  +--GE - GE
 |  +--IEA - IEA

Cf cf835 = loadCf();
Parser parser = new X12Parser(cf835);
// The configuration Cf can be loaded using DI framework.
// Check the sample spring application context file provided.

Double totalChargeAmount = 0.0;
X12 x12 = (X12) parser.parse(new File(``C:\test\835.txt''));
List segments = x12.findSegment(``CLP'');
for (Segment s : segments) {
    totalChargeAmount = totalChargeAmount + Double.parseDouble(s.getElement(3));
}
System.out.println(``Total Change Amount '' + s.getElement(3));
```

Methods

main
public static void **main** (String[] *args*)

exampleParseX12SimpleFileOne

public class **exampleParseX12SimpleFileOne**
Example showing X12Simple Parser reading a X12 file and looping over the segments.

Author Prasad Balan

```
Example of parsing a X12 file

X12Simple x12 = (X12Simple) new X12SimpleParser().parse(new File(``C:\test\835.txt''))
for (Segment s : x12) {
    if (s.getElement(0).equals(``CLP'')) {
        System.out.println(``Total Change Amount '' + s.getElement(3));
    }
}
```

Methods

main
public static void **main** (String[] *args*)

exampleParseX12SimpleStringOne

public class **exampleParseX12SimpleStringOne**
Example showing X12Simple Parser reading a X12 String and looping over the segments.

Author Prasad Balan

```
Example of parsing a X12 String

X12Simple x12 = new X12SimpleParser().parse(``ISA*00*          00          *ZZ*SENDERI
for (Segment s : x12) {
    if (s.getElement(0).equals(``CLP'')) {
        System.out.println(``Total Change Amount '' + s.getElement(3));
    }
}
```

Methods

main
public static void **main** (String[] *args*)

exampleSpringParseX12FileOne

public class **exampleSpringParseX12FileOne**
Example showing X12 Parser reading a X12 file and looping over the segments.

Author Prasad Balan

```
Example of using Spring to load the configuration

This is the loop hierarchy of a 835 transaction used here.

+--X12
```

```
|  +--ISA - ISA
|  |  +--GS - GS
|  |  |  +--ST - ST - 835, - 1
|  |  |  |  +--1000A - N1 - PR, - 1
|  |  |  |  +--1000B - N1 - PE, - 1
|  |  |  |  +--2000 - LX
|  |  |  |  |  +--2100 - CLP
|  |  |  |  |  |  +--2110 - SVC
|  |  |  +--SE - SE
|  |  +--GE - GE
|  +--IEA - IEA

Cf cf835 = loadCf();
Parser parser = new X12Parser(cf835);
// The configuration Cf can be loaded using DI framework.
// Check this example to see how to load configuration using Spring.

Double totalChargeAmount = 0.0;
X12 x12 = (X12) parser.parse(new File(``C:\test\835.txt''));
List segments = x12.findSegment(``CLP'');
for (Segment s : segments) {
    totalChargeAmount = totalChargeAmount + Double.parseDouble(s.getElement(3));
}
System.out.println(``Total Change Amount '' + s.getElement(3));
```

Methods

main
public static void **main** (String[] *args*)

CHAPTER 3

Indices and tables

- genindex
- search

C

E

M