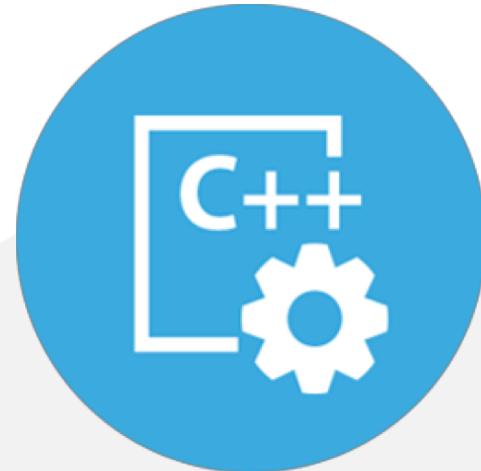


Music Xml to Braille Compiler

<Scanning & Parsing by JAVA CC>

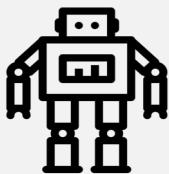


INDEX



1

**Jsoup
Token & Parse**



2

**JAVA CC
Output Braille**



3

**Compiler
Trial and Error**



4

**Learning in
This Course**



Jsoup Library – contain Parser & Token

- ✓ Jsoup.jar



jsoup-1.11.3.jar

jsoup-1.11.3-jav
adoc.jar

Open Source “Jsoup” – XML Controller

Finally, we found the Open source related with XML in JAVA

But, some MXML tags contains attributes, name of tags are changed

After tokenizing & parsing, deal with be used to j-soup library



Jsoup Library – contain Parser & Token

✓ Input XML file in JAVA

```

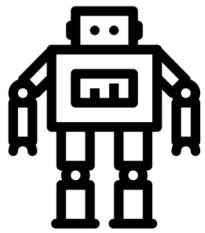
1 package Jsoup;
2
3 import org.jsoup.Jsoup;
4 import org.jsoup.nodes.Document;
5 import java.io.*;
6
7 public class Test{
8     public static void main(String[] args) throws Exception{
9         File input = new File(
10             "C:/Users/user/Desktop/다운로드/Debussy_-_Deux_arabesques/lg-20474636.
11         Document doc = Jsoup.parse(input, "UTF-8");
12
13         System.out.println(doc);
14     }
15 }
```

✓ Jsoup function – Detail Info

The screenshot shows a Java API documentation page for the `Parser` class from the `org.jsoup.parser` package. The `CLASS` tab is selected. The class extends `java.lang.Object`. The constructor summary lists the `Parser(TreeBuilder treeBuilder)` constructor, which creates a new `Parser` using the specified `TreeBuilder`. The method summary includes links for All Methods, Static Methods, Instance Methods, Concrete Methods, and Deprecated Methods.

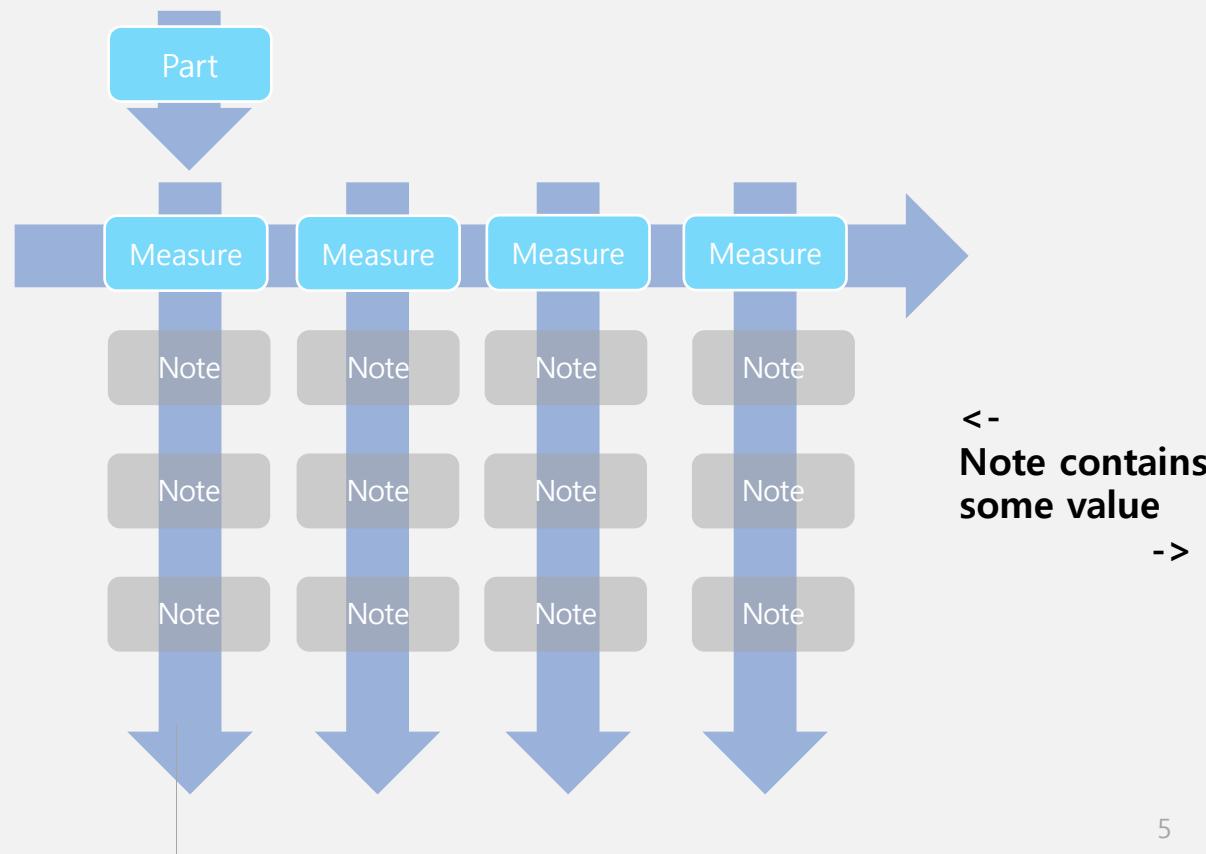
Make JAVA can read XML(UTF-8) File by upper code

<https://jsoup.org/apidocs/org/jsoup/parser/Parser.html>



Simple Output of Braille Music Code - Text

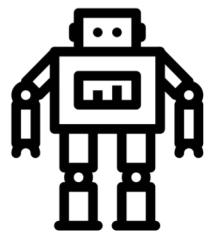
✓ Tree Structure



✓ Can Recognize Measure

```
<part>
  <measure>
    <note>
      <pitch> 4
      </pitch>
      <duration> 2
      </duration>
      <type> eighth
      </type>
    </note>
  </measure>
</part>
measure 1
  note 1
    step: D
    octave: 3
    duration: 12
    type: half
  note 2
    step: C
    octave: 4
    duration: 6
    type: quarter
  note 2
    step: A
    octave: 4
    duration: 2
    type: eighth
end
```

Output of our parser (Input: XML file)



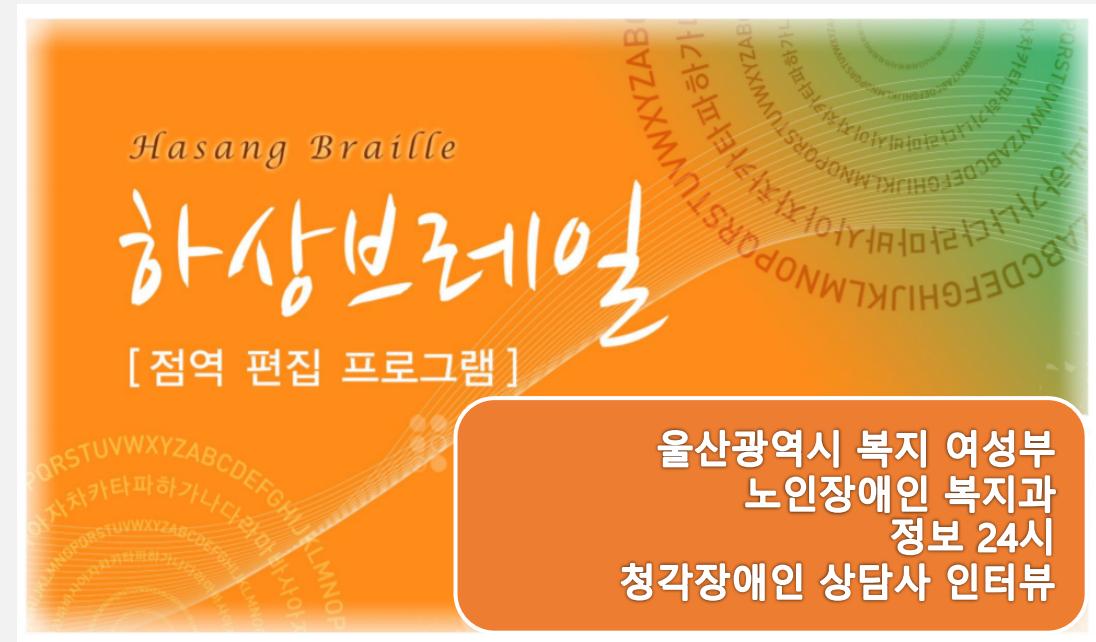
Simple Output of Braille Music Code - Text

✓ Output Braille code in JAVA

The screenshot shows the Eclipse IDE interface with the following details:

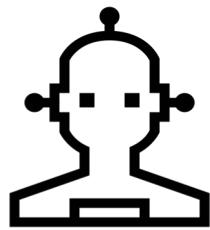
- Project Explorer:** Shows the project structure with files like CompilerCC, MyNewGrammar.jj, ProjectGrammarConstants.java, SimpleCharStream.java, TokenManager.java, TokenManagerError.java, Braille_character.java, and Test.txt.
- Editor:** Displays the Java code for `Braille_character.java`. The code handles the conversion of text into Braille, specifically for the line "what is the use of a book without pictures or conversations on my ears and whiskers how late its getting curioser and curioser". It uses nested loops and string manipulation to generate the Braille representation.
- Terminal:** Shows the output of the application, which is a long string of Braille characters representing the input text.

✓ Interview with Consultant



Input Character -> Output Braille Code

Interview Hearing impairment Consultant



The reference we read for Compiler project

✓ Google Scholar for Compiler

The screenshot shows a Google Scholar search interface. The search bar at the top contains the query 'Compiler'. Below the search bar, there are three suggested search terms: '자바 컴파일러', '컴파일러 개선', and '컴파일러 개발'. A quote from Alan Turing is visible: '거인의 어깨에 올라서서 더 넓은 세상을 바라보라 - 아이작 뉴턴'. Below the search bar, there's a link to 'Google Scholar in English'. At the bottom of the page, there's a navigation bar with icons for Home, Search, and other links, along with the date '2018-06-13'.

✓ Interest paper and so on

개방 시스템 응용을 위한 개선된 ASN. 1 컴파일러 설계 및 구현 (The Design and Implementation of an Enhanced ASN. 1 Compiler for Open System Application)

김홍렬, 임제탁 - 전자공학회논문지-A, 1996 - dbpia.co.kr

Abstract Abstract Syntax Notation One (ASN. 1), defined by ITU-T and ISO, is a formal abstract specification language which has been widely used in international standards specification to interconnect distributed open systems. It is necessary to have well defined ...

VHDL 컴파일러-컴파일러 시스템에 관한 연구 (A study on VHDL compiler-compiler system)

공진홍, 박기영 - 대한전자공학회 학술대회, 1994 - dbpia.co.kr

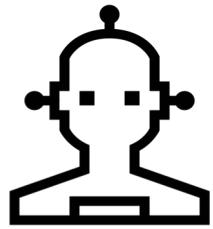
... 표준화된 VHDL 언어는 주기적으로 개선되는 데, 이에 따른 VHDL 지원 시스템의 신속하고도 효과적인 개선 필요성을 컴파일러 ... 실험은 VHDL 컴파일러의 전반부와 후반부를 나누어서 실행하였다. IW.1 VHDL, 컴파일러 전반부 실험 생성된 VHDL 컴파일러 전반부는 VHDL LRM"에 ...

☆ 99 1회 인용 관련 학술자료 전체 4개의 버전

C-언어를 사용한 CMOS 셀 컴파일러의 개발 (Development of CMOS Cell Compiler Using C-Language)

김용주, 오창준, 이철동 - 대한전자공학회 학술대회, 1985 - dbpia.co.kr

... 둘째, 각 실들의 고정된 사양으로 인해 필요에 따라 그 사양을 변경하거나 개선하는 대에 많은 ... 2. CMOS 셀 컴파일러 설계의 기본개념 및 과정 (1) 기본기념 셀 컴파일러는 컴퓨터 언어로 ... 종래의 표준 셈은 셀 라이브러리의 대이다. 베이스가 고정되어 있는 반면 셀 컴파일러는 컴퓨터 ...



Our team project started with 4 IDEATE

✓ Decompiler For Hack



Extracting Source Code from .exe or Android App

8

Obfuscating Code by add trash token at tokenizing, and trash node at Parsing level.

Make lots of tokenizing, parsing and address instruction algorithm.

3 tokenizing * 4 parsing * 3 address instruction = 36 way to decompiling

✓ Make PunchCard For Orgel



At Fortran, PunchCard is used for SourceCode.

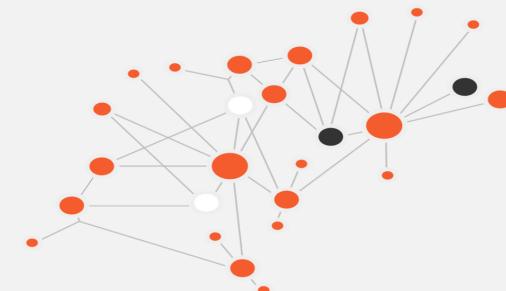
Punch card is used for source code + Some Orgels use punch card as sheet music

Then, lets compiling source code(sheet music) to punch card for orgel.

Lexical = note. Parsing = HashCode for implementing Harmony.

9

✓ SourceCode = Variables Interact



SourceCode seems like Society of many Variables.

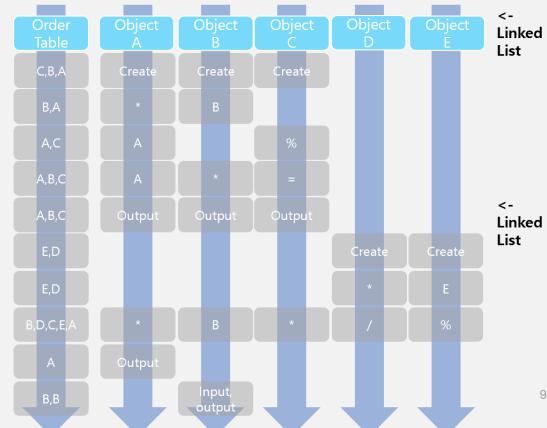
9

Point of programming is Variables. Method and function are just interactive.

Tokenizing string as Operand and Operator. Parsing by Tree Structure.

Starting Point = Design Variable by Class(object)

✓ Tree Structure

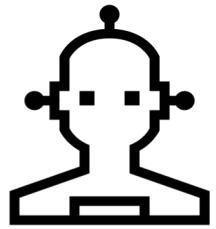


✓ Source Code

```
Public static main(string arg){
    1.     object A, B, C= new object (string );
    2.     A.*=(B);
    3.     C.%=(A);
    4.     C.=(B.*(A));
    5.     A.output(); B.output(); C.output();
    6.     object D, E = new object (string );
    7.     D.*=(E);
    8.     A.*=(E.%*(C.*(D./(B))));
    9.     A.output();
    10.    B.output(B.input());
}
```

8

9



Our team project started with 4 IDEATE

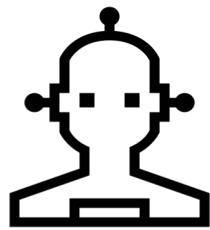
- ✓ Just For Translate One Melody



Not for Translate Harmony Music.

- ✓ Helpful for Conversation

Down to Upper code or Upper to Down code.



Draft Tokenizing by java cc & Revision of Tokenizing

✓ Attributes in Token

```
- <note default-y="-180.00" default-x="119.00">
  + <pitch>
    <duration>2</duration>
    <voice>1</voice>
    <type>eighth</type>
- <time-modification>
  <actual-notes>3</actual-notes>
  <normal-notes>2</normal-notes>
</time-modification>
<stem>up</stem>
<staff>2</staff>
<beam number="1">begin</beam>
- <notations>
  <tuplet type="start" show-number="none" bracket="no"/>
  <slur type="start" number="1"/>
</notations>
</note>
```

✓ Grouping Should be deleted

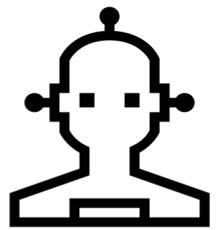
```
TOKEN :
{
  <_MEASURE : < PRINT >< ATTRIBUTES >< NOTE > >
  <_ATTRIBUTES : < DIVISIONS >< KEY >< TIME >< STAVES >< CLEF > >
  <_TIME : < BEATS >< BEATTYPEn > >
  <_CLEF : < SIGN >< LINE > >
}

TOKEN :
{
  < PART : "part" >
  < MEASURE : "measure" >
  < PRINT : "print" >
  < ATTRIBUTES : "attributes" >
  < DIVISIONS : "divisions" >
  < KEY : "key" >
  < FIFTHS : "fifths" >
  < TIME : "time" >
  < BEATS : "beats" >
  < BEATTYPEn : "beat-type" >
  < STAVES : "staves" >
  < CLEF : "clef" >
  < SIGN : "sign" >
  < LINE : "line" >
  < NOTE : "note" >
  < PITCH : "pitch" >
  < STEP : "step" >
  < OCTAVE : "octave" >
  < DURATION : "duration" >
  < TYPE : "type" >
}

TOKEN :
{
  < NUMBER : "number=\\"[1-3]\\\" >
  < NEWLINE : "new-system=\\"yes\\\" >
}
```

In token note - default-x, default-y is existence

Grouping evoke some Error



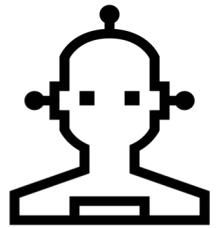
Draft Parsing by java cc

✓ Source Code of Parser.jj

```
101 }
102 {
103     "<"< PRINT >""
104     "</"< PRINT >">"
105     { }
106 }
107
108 void parseAttributes() :
109 {
110 }
111 {
112     "<"< ATTRIBUTES >""
113 //     ( parseDivisions() | parseKey() | parseTime() | parseStaves() | parseClef(
114     "</"< ATTRIBUTES >""
115     { }
116 }
117
118 void parseNote() :
119 {
120 }
121 {
122     "<"< NOTE >">"
123 //     ( parsePitch() | parseDuration() | parseType() )+
124     "</"< NOTE >">"
```

✓ Result of our parser

```
<part>
    <measure>
        <attributes>
        </attributes>
    </measure>
    <measure>
        <print>
        </print>
        <print>
        </print>
        <note>
        </note>
    </measure>
</part>
damn ! !
```



Updated Parsing by java cc

- ## ✓ Source Code of Updated Parser.jj

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer View:** Shows the project structure with files like MyNewGrammar.jj, ProjectGrammar.java, ParseException.java, and various Braille and Token classes.
- Code Editor View:** Displays the Java code for the `ProjectGrammar` class, which implements the `ProjectGrammarConstants` interface. The code handles the main method, parsing input from System.in, and printing results to System.out.
- Outline View:** Shows the class hierarchy and member variables for `note`, `measure`, and `ProjectGrammar`.
- Problems View:** Shows compilation errors and warnings.
- Console View:** Displays the output of the application, including "NOK." and error messages.

```
12 }
13
14 class measure {
15     LinkedList<note> Measure = new LinkedList<note>();
16 }
17
18 public class ProjectGrammar implements ProjectGrammarConstants {
19
20     public static void main(String args []) throws ParseException
21     {
22         ProjectGrammar parser = new ProjectGrammar(System.in);
23
24         LinkedList<measure> part = new LinkedList<measure>();
25
26         boolean end = false;
27
28         while (!end)
29         {
30             try
31             {
32                 end = ProjectGrammar.forNameXML(part);
33             } catch (Exception e)
34             {
35                 System.out.println("NOK.");
36                 System.out.println(e.getMessage());
37             }
38         }
39     }
40 }
```

- ## ✓ Can Recognize Measure

```
<octave> 4
</octave>
</pitch>
<duration> 2
</duration>
<type> eighth
</type>
</note>
</measure>
</part>
|measure 1
|  note 1
|    step: D
|    octave: 3
|    duration: 12
|    type: half
measure 2
  note 1
    step: C
    octave: 4
    duration: 6
    type: quarter
  note 2
    step: A
    octave: 4
    duration: 2
    type: eighth
end
```

Output of our parser (Input: XML file)



Learning through Compiler Course & Project

- ✓ **Ideate Power Project Design**



- ✓ **Project Planning Ability**



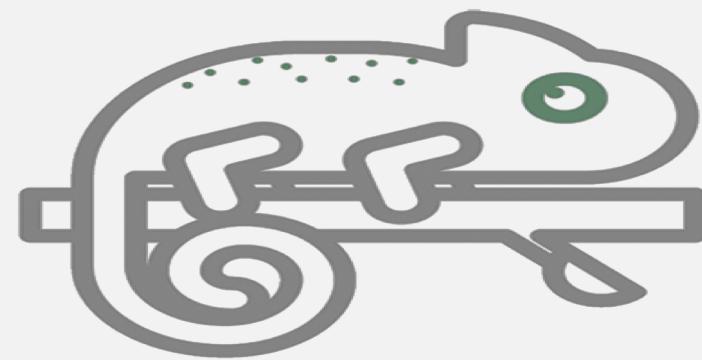
Study = Ideate (By reference paper, history, and present)

Project = Planning (Study + Design) + Programming (Work)



Learning through Compiler Course & Project

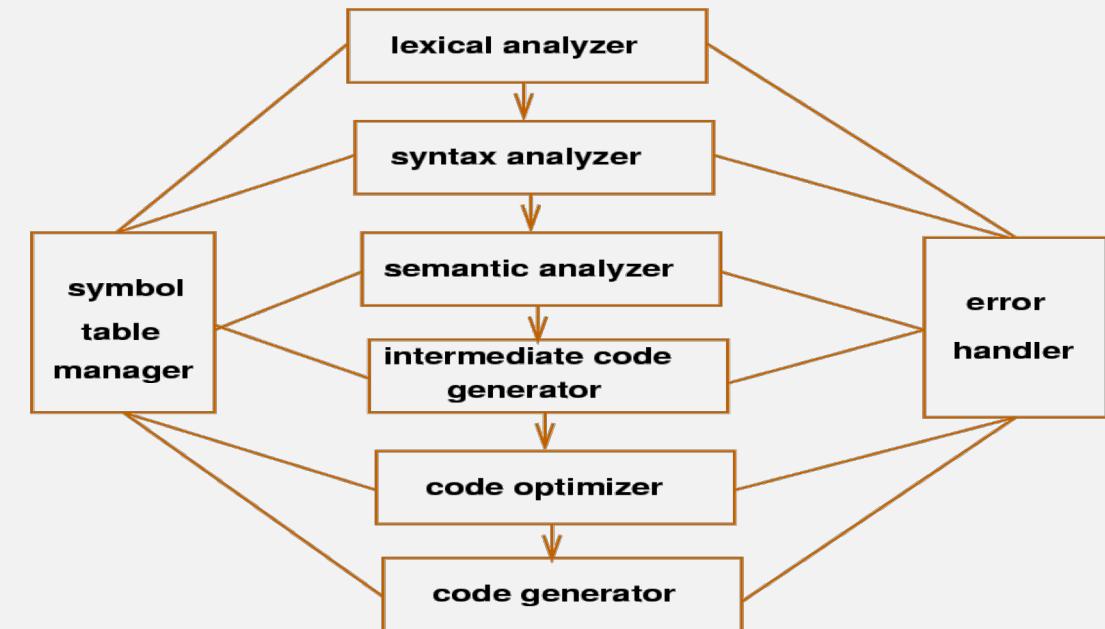
✓ Adaptability on Library



Adaptability

Not only Jsoup for XML. Git hub, Bit bucket, program creek...

✓ Conversation with Computer



Concept of translation between code and machine language

Thank you