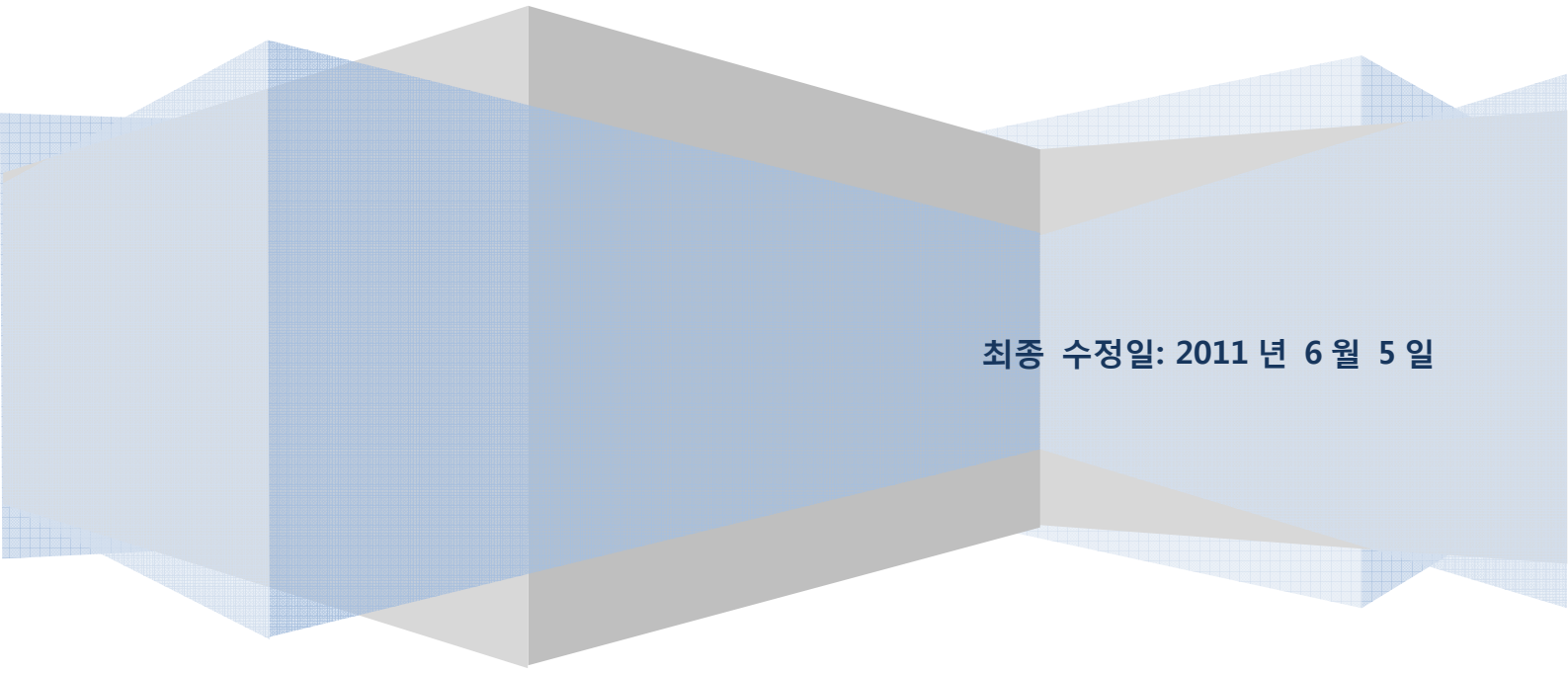
시맨틱 웹 첨단연구센터, 한국과학기술원

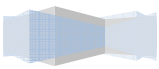
한나눔 한국어 형태소 분석기 사용자 매뉴얼

(jhannanum 0.8.3 기반)



최종 수정일: 2011 년 6 월 5 일

목차



1. 들어가는 말

한나눔 한국어 형태소 분석기는 KAIST Semantic Web Research Center 에서 개발되어 현재 오픈소스로 관리되고 있다. 한국어 형태소 분석기를 개발하기 위해서는 많은 노력과 시간이 요구되기 때문에 오픈소스로 공개된 프로그램은 높은 활용성을 지닌다. 이를 통해 더 많은 사람들이 한국어 분석 기술에 보다 쉽게 접근하여, 한국어 자연언어처리 기반 기술이 더욱 발전할 수 있기를 기대한다.

- 카이스트 시맨틱 웹 첨단 연구 센터: http://semanticweb.kaist.ac.kr
- KLDP 프로젝트 커뮤니티: http://kldp.net/projects/hannanum
- SourceForge.net 프로젝트 커뮤니티: http://sourceforge.net/projects/hannanum/

본 매뉴얼은 한나눔 자바 버전 0.8.3 을 기반으로 하고 있다. 목적에 맞게 다음과 같이 참조하기를 권장한다.

- 제일 먼저 프로그램의 동작을 확인하고 싶습니다. 4. 사용방법, 4.3.2 GUIDemo 실행하기
- 한나눔 라이브러리를 이용한 예제 코드를 보고 싶습니다. 4.3 한나눔 데모 프로그램 활용하기
- 분석 결과에 있는 ncn, jca 등은 무엇을 뜻하나요? 3.4 태그 집합
- 한나눔은 무엇을 하는 프로그램 인가요? 2. 형태소 분석기 개요
- 한나눔이 다른 한국어 형태소 분석기와 어떻게 다른가요? 3. 한나눔 형태소 분석기
- 한나눔은 어떤 기능들을 가지고 있나요? 3.1 한나눔 Workflow 3.2 한나눔 Plug-in 리스트
- 프로그램을 사용하는데 지켜야 할 사항이 있나요? 5. 라이센스
- 새로운 Plug-in 을 직접 개발할 수 있나요? 4.5 새로운 한나눔 Plug-in 작성하여 활용하기

2. 형태소 분석기 개요

자연언어 처리 과정에서 하나의 단어가 여러 품사를 갖는 모호성을 가질 수 있으며 이러한 품사의 모호성을 해소하는 과정을 품사 태깅(Part-Of-Speech Tagging)이라고 한다.[2] 이를 통해서 문장에 사용된 형태소들의 품사를 파악하고 문장의 구조를 파악할 수 있다. 한국어 형태소 분석기는 한국어 텍스트를 입력으로 하고 그것을 형태소 단위로 분석하여 이를 품사와 함께 출력해주는 소프트웨어이다. 형태소 분석을 거쳐 태깅된 데이터는 한국어 자연언어처리에서 기초적이면서도 중요한 역할을 한다.

자연언어처리는 대상언어의 특성에 따라서 분석하는 방법이 상당히 달라지게 된다. 형태소 분석의 경우 고립어에 속하는 영어는 공백 단위로 구분한 토큰을 자르면 쉽게 형태소를 구분할 수 있지만, 굴절어에 속하는 한국어는 1 개 이상의 형태소가 어절을 이루고, 이들 형태소들은 서로 다른 형태소에 대한 영향력을 가지고 있어 형태소 구분 방법이 보다 복잡하다.

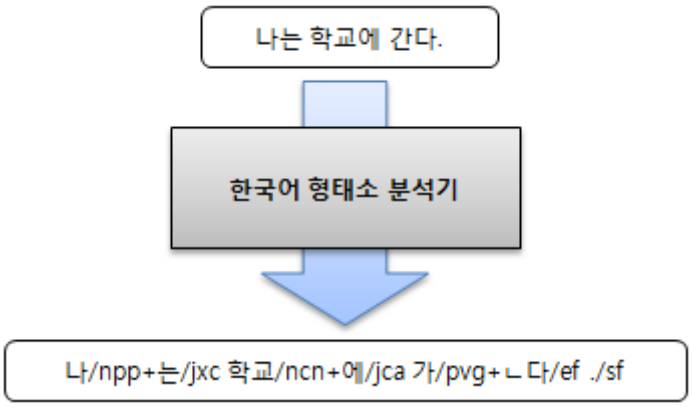


Figure 1 형태소 분석기의 입출력 예

지금까지 이용된 태깅 방법은 크게 규칙을 이용하는 방법과 말뭉치로부터 추출된 통계정보를 이용하는 방법으로 나눌 수 있다. 규칙 접근방법은 규칙을 기술하기가 어렵고 다른 영역으로의 적응성이 떨어지므로 일반적으로 통계정보를 이용하는 방법이 많이 사용되고 있다.[3]

자연언어처리 시스템은 하나의 목적을 위하여 만들어지기 때문에 각 시스템의 구성 요소들은 전체 시스템의 목표를 이루기 위한 가장 효율적인 방법으로 설계되는 것이 일반적이다. 하지만 범용적인 자연언어처리 도구를 개발하기 위해서는 다양한 요구를 유연하게 수용할 수 있는 형태의 설계가 필요하다.[1]

기존의 C 버전에서 발전한 Java 버전의 한나눔 형태소 분석기는 많은 사람들이 쉽고 간편하게, 그리고 다양한 분야에서 활용할 수 있도록 설계되었다.

3. 한나눔 형태소 분석기

한나눔 형태소 분석기는 플러그인 컴포넌트 아키텍처를 적용하여 보다 유연하게 사용될 수 있도록 개선되었다. 사용자는 한국어 처리 목적에 따라 기 개발된 Plug-in 들을 선택하여 Workflow 를 구성하여 사용하면 되고, 개발자는 새롭게 필요한 기능만을 Plug-in 으로 개발하여 기존의 Plug-in 들과 함께 활용할 수 있다.

3.1 한나눔 Workflow

플러그인 컴포넌트 아키텍처 기반의 한나눔의 구성은 다음과 그림과 같다. Workflow 는 분석 수준에 따라서 전처리 단계, 형태소 분석 단계, 품사 태깅 단계의 총 3 단계로 구성되며, 각 Plug-in 은 그 특성에 따라서 Major Plug-in 과 Supplement Plug-in 으로 분류된다.

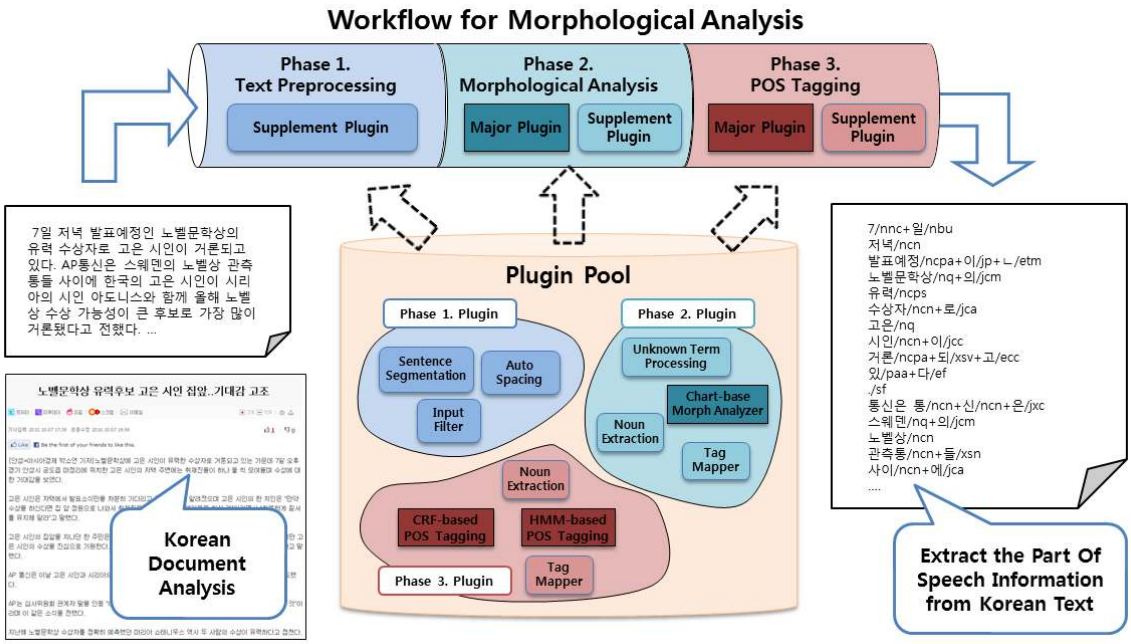


Figure 2 플러그인 컴포넌트 아키텍처 기반의 한나눔 형태소 분석기

한나눔 Workflow 에서 각 분석 단계의 역할과 Major Plug-in 과 Supplement Plug-in 의 역할은 다음과 같다.

- 각 분석 단계 별 역할:
  - **Phase 1. Text Preprocessing** : 문장 경계 인식, 필터링, 자동 띄어쓰기 등 형태소 분석 이전에 필요한 전처리 작업을 수행한다.
  - **Phase 2. Morphological Analysis** : 입력 문장에 대해서 어절 단위로 발생 가능한 모든 형태소 분석 결과를 생성한다.
  - **Phase 3. POS Tagging** : 가장 유망한 형태소 분석 결과들을 선택하여 입력 문장에 대한 최종 품사 태깅 결과를 반환한다.

- Major Plug-in 과 Supplement Plug-in 의 역할
  - **Major Plug-in** : 형태소 분석, 품사 태깅 등 각 분석 단계에서 핵심이 되는 기능을 수행한다. Major Plug-in 은 입력 형태와 출력 형태가 서로 다르기 때문에 각 단계에서는 단 하나의 Major Plug-in 만 배치 할 수 있다.
  - **Supplement Plug-in** : 문장 경계 인식, 필터링, 형태소 태그 변환, 명사 추출 등 형태소 분석과 품사 태깅 이외의 보조적인 기능을 수행한다. Supplement Plug-in 의 입력과 출력 형태는 서로 같으므로 각 단계에서 여러개의 Supplement Plug-in 을 선택하여 활용 할 수 있다.

다음 그림은 한나눔 Workflow 의 구성 예를 보여준다.

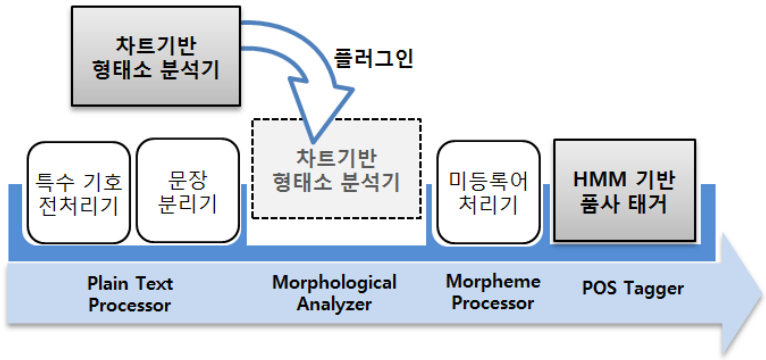


Figure 3 Work flow 구성 예

멀티 프로세서 환경에서 보다 효과적인 동작을 위해 한나눔 Plug-in 들은 Workflow 상에서 개별적인 Thread 위에서 동작 가능하다. 다음 그림은 Multi-thread mode 로 구성된 한나눔 Workflow 의 동작을 보여준다.

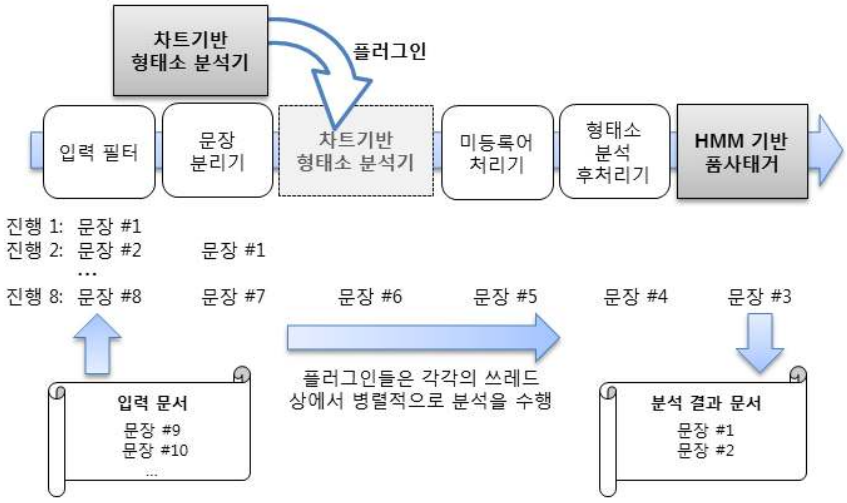


Figure 4 Multi-thread mode 에서의 work flow 동작

3.2 한나눔 Plug-in 리스트

3.2.1 Phase 1. Plain Text Processing

3.2.1.1 Supplement Plug-in

- InformalSentenceFilter

InformalSentenceFilter 는 약 9 만건의 인터텟 댓글 자료를 조사하여 발견한 비형식적 패턴을 기반으로 한다. 비형식적 패턴들은 반드시 세밀한 분석이 필요하지 않음에도 불구하고 많은 리소스를 소비하여 형태소 분석기의 성능을 떨어트리기 때문에 경우에 따라서는 전처리 과정이 필요하다. 대표적인 예는 아래와 같다. 이러한 입력들은 설정에 따라서 워크플로의 이후 단계에서 분석되지 않거나 짧은 단위로 나뉘어 분석된다.

- 특수기호의 반복적인 사용

  예) $$$$$$$$$$$$ 일시 $$$$$$$$$$$$$

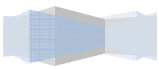
- 띄어쓰기 없는 짧은 패턴의 지속적인 반복

  예) 서울시장서울시장서울시장서울시장서울시장...

- SentenceSegmentor

SentenceSegmentor 는 문장의 구분자 역할을 할 수 있는 마침표, 물음표, 느낌표 문장 기호를 기준으로 전·후 조건에 따라 문장 구분을 결정하는 간단한 방식으로 구현되었다. 마침표에 대해서는 바로 뒤에 숫자가 있는 경우 소수점으로, 마침표 바로 앞에 영문자가 있는 경우 영문 약어로, 바로 뒤에 또 다른 마침표가 있는 경우 말줄임표로, 앞 단어의 길이가 2 글자 이하인 경우 말머리표로 인식하여 문장구분을 하지 않는다. 물음표와 느낌표는 다른 특수 기호와 함께 사용된 경우에 문장 구분자로 인식하지 않는다. 문장을 구분하는 방법은 위와 같은 단순한 방법 이외에도 구문 구조를 이용한 방법 등 보다 복잡한 방법이 활용될 수 있으므로 필요에 따라서는 새로운 플러그인의 개발이 필요하다.

- 3.14 소수점으로 인식
- Ltd. 영문 약어로 인식
- 가. 일정 말머리표로 인식
- !@#?$% 다른 특수기호와 사용된 경우 일반 특수 기호로 인식



3.2.2 Phase 2. Morphological Analysis

3.2.2.1 Major Plug-in

- ChartMorphAnalyzer

차트 기반 형태소 분석기는 형태소 분석을 위한 내부 저장 공간으로 Lattice 형태의 차트를 사용한다. 차트는 Morpheme Chart, Segment Position, Inverse Segment Position 으로 구성되며 [5],[6]의 CKY table 과 [4]의 격자구조를 발전시킨 형태이다. 사전 검색 모듈은 시스템 사전 검색과 사용자 사전 검색, 그리고 숫자 인식기로 구성된다. 형태소 분석기의 기본 사전 내용은 변경하지 않고 필요에 따라서 사용자 사전을 구축할 수 있고, 숫자 처리 또한 사전 검색과 동일한 관점에서 처리할 수 있는 장점이 있다. 사전은 형태소 분석기를 위해 고안된 사전 구조인 TDBM(Trie based DBM)의 형태로 이용된다. 음운 변화처리는 어미의 탈락, 어간의 탈락과 같은 자동적 변화와 불규칙 용언에 의한 불규칙 변화, 모음조화 및 축약과 같은 선택적 변화로 나누어 처리한다. 미등록어에 대한 처리를 위해서 형태소 분석 결과가 없는 경우에는 모든 분할 위치에 "unk" 품사를 할당하고 다시 형태소 분석을 실시한다. 내부적으로 사용하는 한글 인코딩 방식은 초성, 중성, 종성 단위의 3 Character 인코딩 방식으로 삼보 KSSM 조합형 코드와 유사한 형태이다. 유니코드와 내부 한글 인코딩의 상호 변환은 코드 변환 모듈을 통해 이루어진다. 형태소 사전과 태그셋, 결합 규칙은 쉽게 편집 가능한 독립적인 파일로 존재하여 형태소분석 컴포넌트 내에서도 유연한 변경이 가능하다.

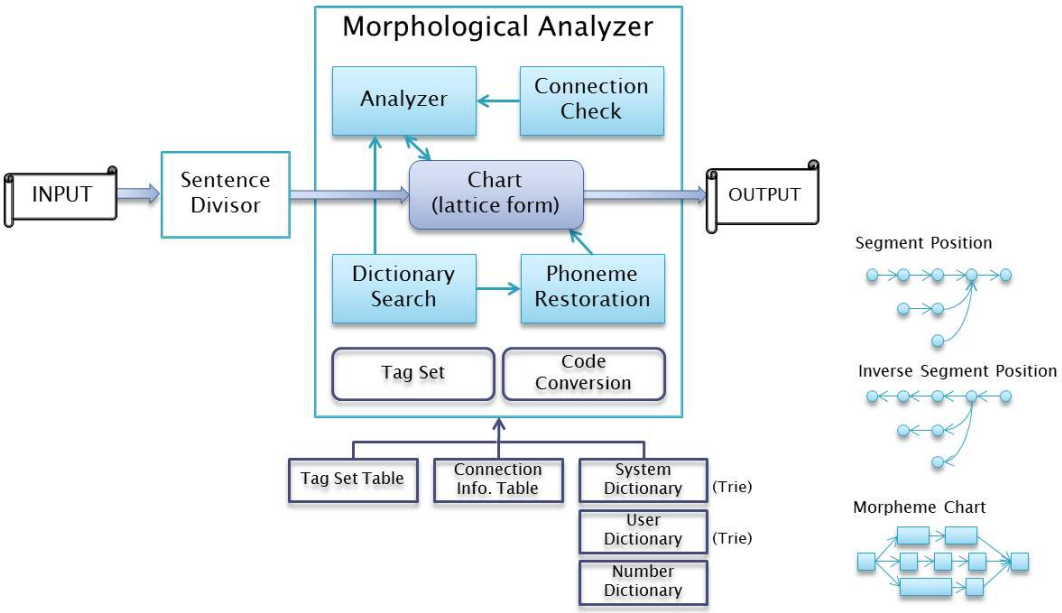


Figure 5 Chart-based Morphological Analyzer 구성

3.2.2.2 Supplement Plug-in

- UnknownMorphProcessor

현재 미등록 명사 처리기는 "unk"로 태깅된 결과 중 유력한 분석에 대해서 비 서술성 명사와 고유명사로 보정하는 단순하면서도 효과적인 방법을 사용하고 있다. 형태소 분석과정에서는 사전에 등록되어 있지 않은 단어에 대한 처리를 위해서 "unk" 태그를 이용하고 있다. "나는 지금 불닭을 먹고 있다." 라는 예제 문장에서 "불닭"은 형태소 사전에 등록되어 있지 않은 단어로 형태소 분석기는 다음과 같은 분석 후보를 생성한다.

```
불닭을
        불닭/unk+을/jco
        불닭을/unk
```

여기에 대해서 미등록어 처리기는 "불닭/unk"에 비서술성명사(ncn)와 고유명사(nq)를 태깅한 결과를 생성한다.

```
불닭을
        불닭/ncn+을/jco
        불닭/nq+을/jco
```

- SimpleMAResult22

ChartMorphAnalyzer 는 총 69 개의 품사 태그로 구성된 카이스트 태그셋을 기반으로 하고 있다. 세분화된 태그셋을 이용한 형태소 분석은 보다 상세한 정보를 제공하므로 일반적으로 유용하지만, 간단한 분석 결과를 원하는 사용자에게는 오히려 부담이 될 수 있다. SimpleMAResult22 plug-in 은 기본적인 형태소 분석 결과를 1 단계 낮은 단계의, 총 22 개의 품사 태그로 구성된 태그셋을 사용한 분석 결과로 변환시켜 제공한다.

입력 예)

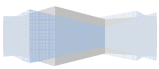
```
학교에서조차도 그 사실을 모르고 있었다.
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor

```
학교에서조차도
	학교/ncn+에서/jca+조차도/jxc
	학교/ncn+에서/jca+조차/jxc+도/jxc
그
	그/mmd
	그/npd
	그/npp
사실을
	사실/ncn+을/jco
	사/pvg+아/ecx+싣/px+을/etm
모르고
	모르/pvg+고/ecc
	모르/pvg+고/ecs
	모르/pvg+고/ecx
있었다
	있/paa+었/ep+다/ef
	있/px+었/ep+다/ef
.
	./sf
	./sy
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor + SimpleMAResult22

```
학교에서조차도
	학교/NC+에서/JC+조차도/JX
그
	그/NP
	그/MM
사실을
	사실/NC+을/JC
	사/PV+아/EC+싣/PX+을/ET
모르고
	모르/PV+고/EC
있었다
	있/PA+었/EP+다/EF
	있/PX+었/EP+다/EF
.
	./SF
	./SY
```

- SimpleMAResult09

ChartMorphAnalyzer 는 총 69 개의 품사 태그로 구성된 카이스트 태그셋을 기반으로 하고 있다. 세분화된 태그셋을 이용한 형태소 분석은 보다 상세한 정보를 제공하므로 일반적으로 유용하지만, 간단한 분석 결과를 원하는 사용자에게는 오히려 부담이 될 수 있다. SimpleMAResult09 plug-in 은 기본적인 형태소 분석 결과를 2 단계 낮은 단계의, 총 9 개의 품사 태그로 구성된 태그셋을 사용한 분석 결과로 변환시켜 제공한다.

입력 예)

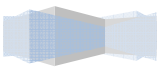
```
학교에서조차도 그 사실을 모르고 있었다.
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor

```
학교에서조차도
	학교/ncn+에서/jca+조차도/jxc
	학교/ncn+에서/jca+조차/jxc+도/jxc
그
	그/mmd
	그/npd
	그/npp
사실을
	사실/ncn+을/jco
	사/pvg+아/ecx+싣/px+을/etm
모르고
	모르/pvg+고/ecc
	모르/pvg+고/ecs
	모르/pvg+고/ecx
있었다
	있/paa+었/ep+다/ef
	있/px+었/ep+다/ef
.
	./sf
	./sy
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor + SimpleMAResult09

```
학교에서조차도
	학교/N+에서조차도/J
그
	그/N
	그/M
사실을
	사실/N+을/J
	사/P+아/E+싣/P+을/E
모르고
	모르/P+고/E
있었다
	있/P+었다/E
.
	./S
```

3.2.3 Phase 3. POS Tagging

3.2.3.1 Major Plug-in

- HmmPosTagger

한나눔 형태소 분석기에서 구현한 HMM 품사 태거는 어절간의 의존성과 형태소간의 의존성 모두를 반영하는 은닉 마르코프 모델을 기반으로 한다.[3] 영어의 경우는 하나의 문장을 단어의 열로 볼 수 있지만 한국어의 경우는 어절의 열로 보는 것이 합리적이다. 한국어의 어절은 그 순서가 자유롭지만 통계적으로 볼 때 규칙을 발견 할 수 있다. 어절간의 의존성은 구문 정보로서 태깅 과정에서 중요한 역할을 할 수 있다. 품사 태깅은 주어진 문장 $W=W_0W_1...W_n$ 에 대한 어절 태그열 $T=T_0T_1...T_n$ 을 찾는 문제로 정의할 수 있으며 어절 태그열을 구하는 함수 Φ는 다음과 같이 표현된다.

$$\Phi(\mathrm{T}) = \underset{T}{argmax}\ \ p(W|T)\,p(T)$$

위 확률식을 기반으로 마르코프 독립 가정을 적용하여 단순화 시키면 어절 태그 발생 확률은 (a)로 표현되고 어절 내의 형태소 태그 발생확률은 (b)로 표현된다.

$$\mathrm{P(T,W)} \cong \prod_{i=1}^{n} \frac{P(T_i, W_i)\,P(T_i|T_{i-1})}{P(T_i)} \quad (a)$$

$$\mathrm{P(T_i,W_i)} \cong \prod_{j=1}^{n} \frac{P(t_{i,j}, w_{i,j})\,P(t_{i,j}|t_{i,j-1})}{P(t_{i,j})} \quad (b)$$

$w_{i,j}$: i 번째 어절의 j 번째 형태소
$t_{i,j}$: i 번째 어절의 j 번째 품사 태그

3.2.3.2 Supplement Plug-in

- NounExtractor

입력 문장 중에서 명사만을 추출하기 위해 형태소 분석을 수행하는 경우가 있다. NounExtractor는 품사 태깅 결과 명사로 인식된 형태소만을 추출한다.

입력 예)

롯데마트가 판매하고 있는 흑마늘 양념 치킨이 논란이 되고 있다.

출력 예)

롯데마트/ncn, , 판매/ncpa, 흑마늘/ncn, 양념/ncn, 치킨/ncn, 논란/ncpa

- SimplePOSResult22

ChartMorphAnalyzer 는 총 69 개의 품사 태그로 구성된 카이스트 태그셋을 기반으로 하고 있다. 세분화된 태그셋을 이용한 품사태깅은 보다 상세한 정보를 제공하므로 일반적으로 유용하지만, 간단한 분석 결과를 원하는 사용자에게는 오히려 부담이 될 수 있다. SimplePOSResult22 plug-in 은 기본적인 품사태깅 결과를 1 단계 낮은 단계의, 총 22 개의 품사 태그로 구성된 태그셋을 사용한 태깅 결과로 변환시켜 제공한다.

입력 예)

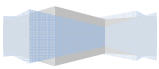
```
학교에서조차도 그 사실을 모르고 있었다.
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor + HmmPosTagger

```
학교에서조차도
	학교/ncn+에서/jca+조차/jxc+도/jxc
그
	그/mmd
사실을
	사실/ncn+을/jco
모르고
	모르/pvg+고/ecc
있었다
	있/px+었/ep+다/ef
.
	./sf
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor + HmmPosTagger + SimplePOSResult22

```
학교에서조차도
	학교/NC+에서/JC+조차도/JX
그
	그/MM
사실을
	사실/NC+을/JC
모르고
	모르/PV+고/EC
있었다
	있/PX+었/EP+다/EF
.
	./SF
```

- SimplePOSResult09

ChartMorphAnalyzer 는 총 69 개의 품사 태그로 구성된 카이스트 태그셋을 기반으로 하고 있다. 세분화된 태그셋을 이용한 품사태깅은 보다 상세한 정보를 제공하므로 일반적으로 유용하지만, 간단한 분석 결과를 원하는 사용자에게는 오히려 부담이 될 수 있다. SimplePOSResult09 plug-in 은 기본적인 품사태깅 결과를 2 단계 낮은 단계의, 총 9 개의 품사 태그로 구성된 태그셋을 사용한 태깅 결과로 변환시켜 제공한다.

입력 예)

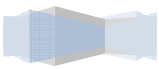
```
학교에서조차도 그 사실을 모르고 있었다.
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor + HmmPosTagger

```
학교에서조차도
	학교/ncn+에서/jca+조차/jxc+도/jxc
그
	그/mmd
사실을
	사실/ncn+을/jco
모르고
	모르/pvg+고/ecc
있었다
	있/px+었/ep+다/ef
.
	./sf
```

출력 예: ChartMorphAnalyzer + UnknownMorphProcessor + HmmPosTagger + SimplePOSResult09

```
학교에서조차도
	학교/N+에서조차도/J
그
	그/M
사실을
	사실/N+을/J
모르고
	모르/P+고/E
있었다
	있/P+었다/E
.
	./S
```

3.3 형태소 사전

잘 구축되어 있는 형태소 사전은 형태소 분석 과정에서 매우 중요한 역할을 한다. 한나눔 형태소 분석기에서는 세가지 형태의 사전을 활용한다. 사전의 종류는 다음과 같다.

- **시스템 사전**: 카이스트 코퍼스를 기반으로 구축된 사전으로 한나눔 형태소 분석기에서 기본적으로 활용되는 사전이다.
- **사용자 사전**: 시스템 사전은 사용자가 간단히 수정하기 어려운 문제가 있지만 사용자 사전은 각 사용자의 목적에 따라서 간단히 항목을 추가 / 수정 할 수 있기 때문에 유연하게 사용될 수 있다.
- **숫자 사전**: 오토마타를 이용한 프로그램이다. 숫자 인식 모듈을 사전의 형태로 구현함으로써 사전 검색과 동일한 관점에서 숫자를 처리할 수 있다.

```
75149명득 nqpb
75150명란 ncn
75151명란젓 ncn
75152명랑 ncps
75153명랑성 ncn
75154명랑해지 pvg
75155명래 nqpb
75156명량 nqq
75157명렬 nqpb
75158명령 ncpa
75159명령개정 ncn
75160명령계통 ncn
75161명령구조 ncn
75162명령문 ncn
75163명령문장 ncn
75164명령서 ncn
75165명령수령자 ncn
75166명령실행속도 ncn
75167명령어 ncn
75168명령어언어사양 ncn
75169명령어체계 ncn
75170명령어축약형컴퓨터 ncn
75171명령어캐시 ncn
75172명령어코드 ncn
75173명령어코드체계 ncn
```

Figure 6 시스템 사전의 일부

3.4 태그 집합

한나눔 형태소 분석기에서는 총 69 개의 확장된 카이스트 태그셋을 기본으로 사용하고 있다. 확장되기 이전의 카이스트 태그셋은 총 54 개의 태그로 구성되어 있지만 현재는 6 개 상위 태그에 대해서 20 개의 새로운 태그를 세분화하여 사용하고 있다. 새롭게 추가된 태그는 파란색으로 표시하였다.

.상위 분류			태그	
기호 S			sp 쉼표 sl 여는 따옴표 및 묶음표 sd 이음표 su 단위 기호	sf 마침표 sr 닫는 따옴표 및 묶음표 se 줄임표 sy 기타 기호
외국어 F			f 외국어	
체언 N	보통명사 NC	서술성명사 ncp	ncpa 동작성 명사	ncps 상태성 명사
		비서술성명사 ncn	ncn 비서술성 명사	ncr 비서술성 -- 직위 명사
	고유명사 NQ		nqpa 성 nqpc 성+이름	nqpb 이름 nqq 기타 - 일반
	의존명사 NB		nbu 단위성 의존명사 nbn 비단위성 의존명사	nbs 비단위성 의존명사 -- 하다 붙는 것
	대명사 NP		npp 인칭대명사	npd 지시대명사
	수사 NN		nnc 양수사	nno 서수사
용언 P	동사 PV		pvd 지시 동사	pvg 일반 동사
	형용사 PA		pad 지시형용사	paa 성상형용사
	보조용언 PX		px 보조용언	
수식언 M	관형사 MM		mmd 지시관형사	mma 성상관형사
	부사 MA		mad 지시부사 mag 일반부사	maj 접속부사
독립언 I	감탄사 Ⅱ		ii 감탄사	
관계언 J	격조사 JC		jcs 주격조사 jcc 보격조사 jcv 호격조사 jcj 접속격조사 jcr 인용격조사	Jco 목적격조사 jcm 관형격조사 jca 부사격조사 jct 공동격조사

	보조사 JX		jxc 통용보조사	jxf 종결보조사
	서술격조사 JP		jp 서술격조사	
어미 E	선어말어미 EP		ep 선어말어미	
	연결어미 EC		ecc 대등적 연결어미 ecx 보조적 연결어미	ecs 종속적 연결어미
	전성어미 ET		etn 명사형 전성어미	etm 관형사형 전성어미
	종결어미 EF		ef 종결어미	
접사 X	접두사 XP		xp 접두사	
	접미사 XS	명사파생 접미사 xsn	xsnu 단위뒤 xsnca 일반명사 뒤 xsncc 일반명사 뒤	xsna 동작성 뒤 xsns 상태성 뒤 xsnp 인명 1,3 뒤 xsnx 모든 명사 뒤
		동사파생 접미사 xsv	xsvv 동사뒤 xsvn 일반명사뒤	xsva 동작명사뒤
		형용사파생 접미사 xsm	xsms 상태명사뒤	xsmn 일반명사뒤
		부사파생 접미사 xsa	xsam 형용사뒤	xsas 상태명사뒤

표 1 한나눔에서 기본적으로 사용하는 카이스트 형태소 태그 집합

4. 사용방법

4.1 사용환경

한나눔 형태소 분석기는 자바 프로그래밍 언어로 개발되었다. 따라서 JDK 6 이상의 자바 플랫폼이 설치되어 있다면 어떤 환경에서든지 한나눔을 사용하는 것이 가능하다. JDK 는 ORACLE 의 자바 다운로드 페이지에서 내려 받을 수 있다.

- JDK 다운로드: http://www.oracle.com/technetwork/java/javase/downloads/index.html

Figure 7 JDK 다운로드 웹 페이지

4.2 다운로드

한나눔 형태소 분석기는 KLDP 에서 제공하는 공개 소프트웨어 프로젝트 사이트에 등록되어 있다. 이곳에서 다운로드 가능하며 문의사항이 생기거나 다른 의견이 있는 경우 게시판을 자유롭게 활용 할 수 있다. 릴리즈된 형태의 프로그램을 다운로드 받을 수 있으며, SVN Repository 를 통해 현재 개발중인 가장 최신 버전의 프로그램을 이용할 수 있다.

- 한나눔 프로젝트 페이지: http://kldp.net/projects/hannanum/

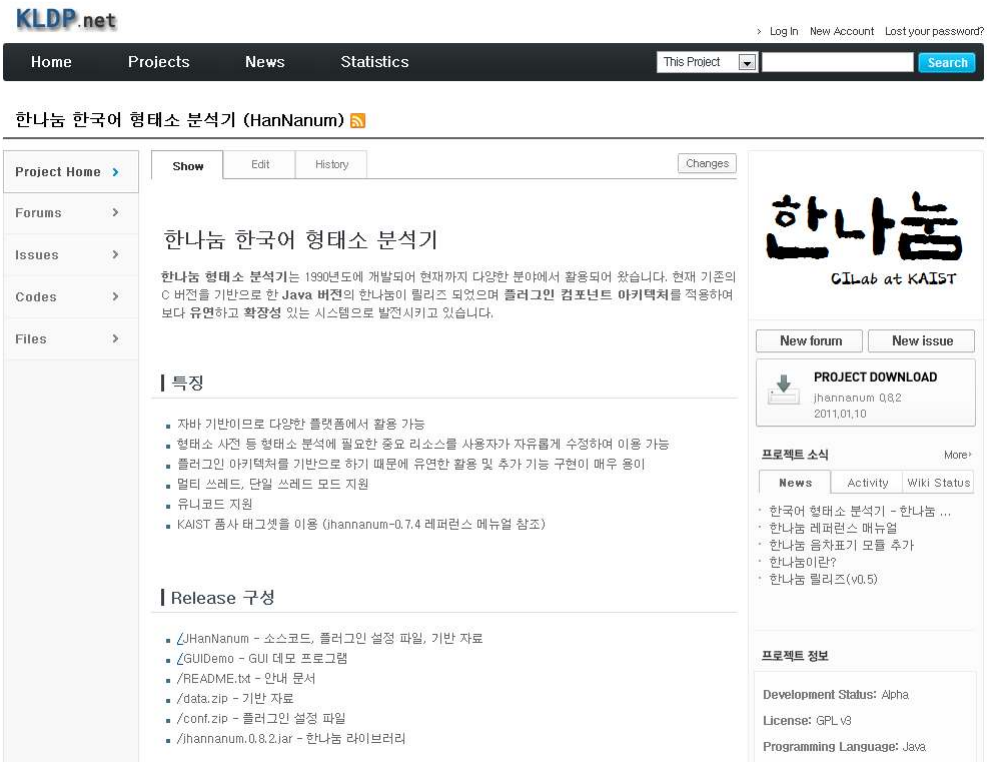


Figure 8 KLDP 한나눔 프로젝트 사이트

4.2.1 릴리즈 다운로드

KLDP 한나눔 프로젝트 사이트의 다운로드 페이지에서 릴리즈를 다운로드 받을 수 있다. 현재 매뉴얼은 jhannanum 0.8.3 (java version)을 기반으로 하고 있다.

- 한나눔 릴리즈 다운로드: http://kldp.net/projects/hannanum/download

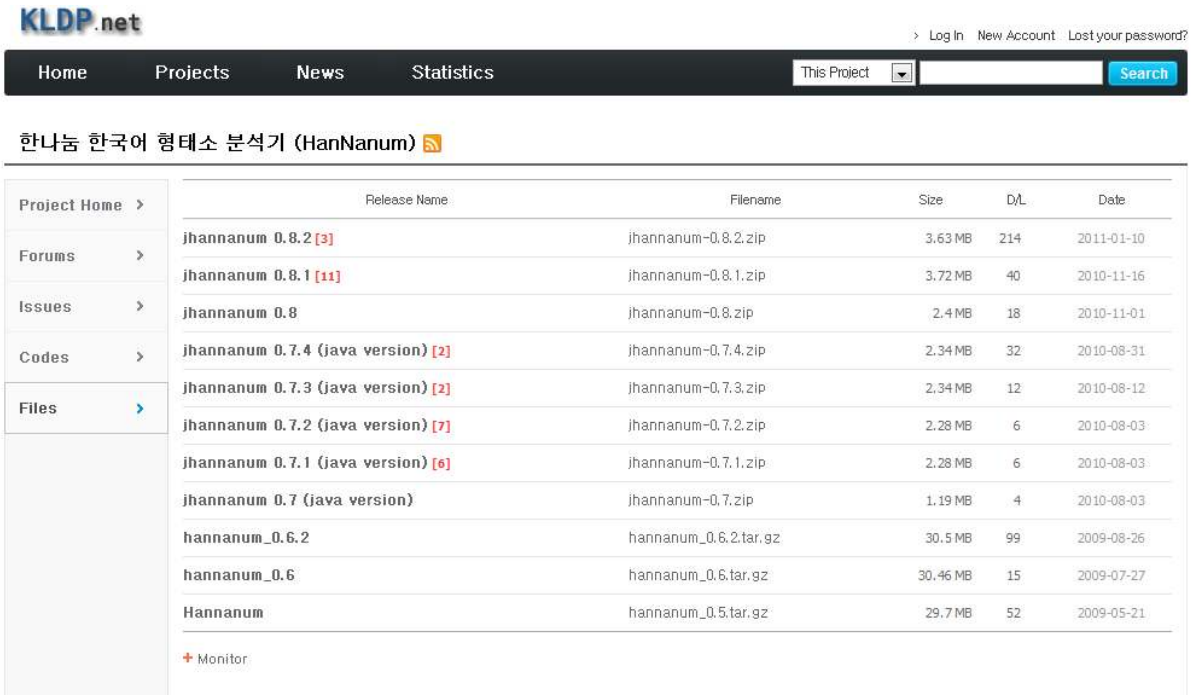


Figure 9 한나눔 릴리즈 다운로드

Jhannanum 0.8.3 릴리즈는 다음과 같이 구성되어 있다.

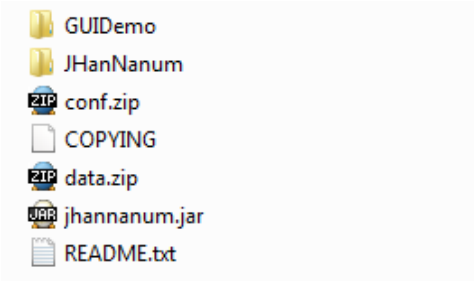


Figure 10 jhannanum release 0.8.3

- **GUIDemo**: GUI 기반의 한나눔 형태소 분석기 데모 프로그램으로 다양한 Workflow 를 간단히 구성하고 테스트하는 것이 가능하다. 플랫폼에 따라서 간단히 execute.bat 또는 execute.sh 를 실행하면 된다.
- **JHanNanum**: 자바 버전 한나눔의 프로젝트 디렉토리로 소스코드, JAVADOC, 데이터 등 모든 것을 포함하고 있다.

- **jhannanum-0.8.3.jar**: 자바 아카이브 형태의 라이브러리로 자바 응용프로그램에 링크하여 바로 활용될 수 있다. 각 플러그인의 설정 파일이 포함되어 있는 conf 디렉토리 및 형태소 사전 등의 데이터 파일이 포함된 data 디렉토리를 PROJECT_ROOT/ 경로에 위치시키지 않는다면 API 사용시 경로를 정확히 지정해줘야 한다.

- **conf.zip**: 각 플러그인의 환경 설정 파일이 포함되어 있다. 환경 설정 파일은 json 포멧을 사용하는 것을 원칙으로 하지만 그 내용은 제한적이지 않다.

- **data.zip**: 한나눔 라이브러리를 이용하여 프로그램을 개발하기 위해서는 data.zip 에 포함된 데이터가 필요하다. 사용자가 직접 형태소 사전을 수정하는 등의 편의성을 제공하기 위하여 자바 아카이브에 포함시키지 않았다. 압축 파일의 구성은 다음과 같다.
  - kE: 시스템 사전, 기분석 사전, 사용자 사전, 태그셋, 결합정보를 포함한다.
  - stat: HMMTagger 가 사용하는 어절 태그 기반 확률 정보를 포함한다.

- **README.txt**: 릴리즈 정보 및 한나눔 이용에 관한 간략한 설명이 포함되어 있다.

4.2.2 Check Out from SVN Repository

릴리즈 다운로드 이외에 SVN Repository 를 통해서 현재 개발 중인 가장 최신 버전의 한나눔을 이용할 수 있다. SVN/trunk/에는 C 버전의 한나눔과 Java 버전의 한나눔이 구분되어 있으므로 필요에 따라서는 한가지 버전만 체크아웃 할 수 있다.

- SVN 정보 페이지: http://kldp.net/projects/hannanum/src
- 한나눔 Java 버전 체크아웃:
  svn checkout --username anonsvn http://kldp.net/svn/hannanum/trunk/java
  password: anonsvn

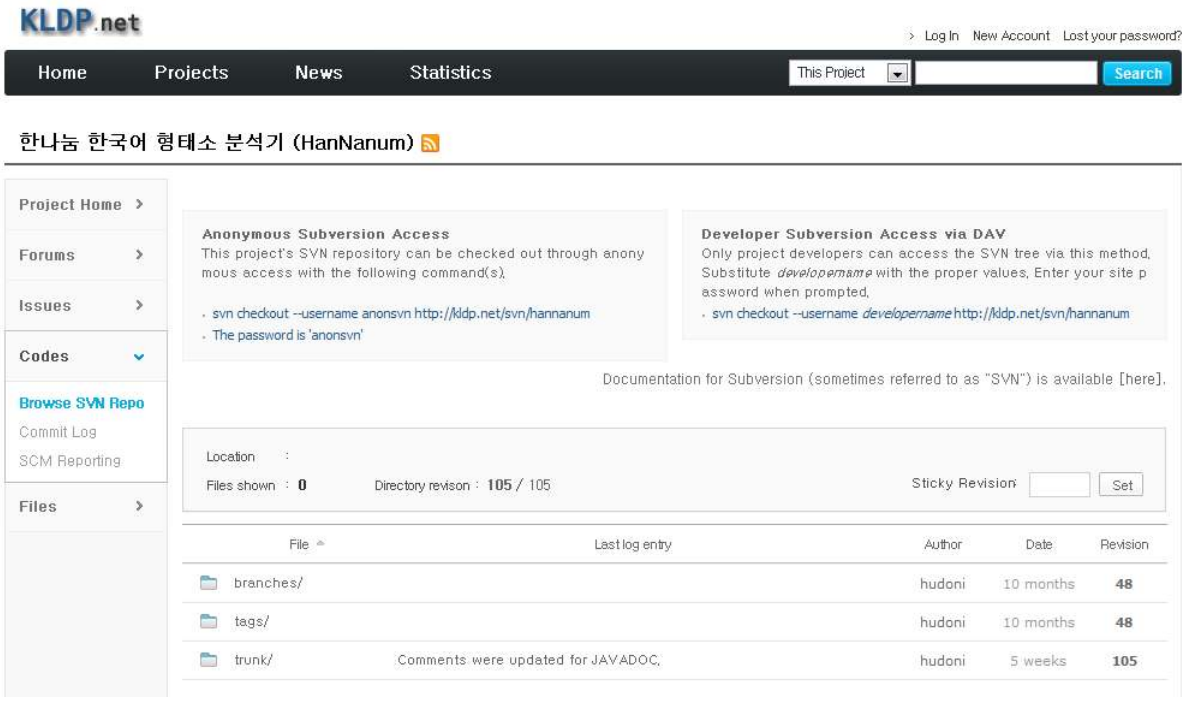


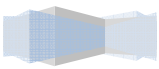
Figure 11 한나눔 SVN Repository

4.3 한나눔 데모 프로그램 활용하기

한나눔을 이용해서 자바 응용프로그램을 개발하기 위해서는 한나눔 릴리즈에 포함되어 있는 라이브러리를 사용하거나 소스코드를 직접 이용하면 된다. 현재 자바 버전의 한나눔에는 사용자의 편의를 위한 다양한 데모 프로그램들이 포함되어 있다. 이번 절에는 데모 프로그램을 동작시키는 방법이 기술되어 있다. 현재 다음 데모 프로그램들이 포함되어 있다.

Demo Package: kr.ac.kaist.swrc.jhannanum.demo.*

- **GUIDemo.java**: GUI 기반의 데모 프로그램으로 plug-in 들을 drag-and-drop 방식으로 work flow 배치시켜 테스트 할 수 있다. 다양한 work flow 를 간단히 구성하여 테스트 하는 것이 가능하다.
- **ManualWorkflowSetUp.java**: 한나눔 라이브러리의 가장 기본적인 활용 방법을 소개하는 데모 프로그램으로, plug-in 들을 work flow 에 배치하여 활성화 시키고 입력 문장을 분석하여 결과를 보여주는 API 를 소개한다.
- **WorkflowHmmPosTagger.java**: 형태소 분석 이후에 품사 태깅을 수행하는 예제 프로그램이다.
- **WorkflowMorphAnalyzer.java**: 품사 태깅은 하지 않고 형태소 분석까지만 수행하는 work flow 를 이용한다.
- **WorkflowNounExtractor.java**: 형태소 분석, 품사 태깅 이후에 명사만을 추출하여 보여준다.
- **WorkflowSimplePos09.java**: 69 개의 카이스트 형태소 태그를 기반으로 하는 기존의 품사 태깅 결과를 9 개의 형태소 태그 기반의 간단한 분석 결과로 변환하여 제공한다.
- **WorkflowSimplePos22.java**: 69 개의 카이스트 형태소 태그를 기반으로 하는 기존의 품사 태깅 결과를 22 개의 형태소 태그 기반의 간단한 분석 결과로 변환하여 제공한다.



4.3.1 Eclipse 를 이용한 한나눔 데모 프로그램 실행 방법

데모 프로그램을 동작 시키기 위한 방법으로 한나눔 프로젝트를 이클립스에 등록시키고 실행하는 방법을 소개한다. 이클립스는 자바 응용 프로그램을 개발하기 위한 IDE 로 가장 많이 사용되고 있다. 본 예제를 위해 사용된 이클립스 버전은 "Eclipse Helios SR2 Win32"이다. 이클립스는 다음 웹 페이지에서 다운로드 받을 수 있다.

- 이클립스 다운로드: http://www.eclipse.org/downloads/

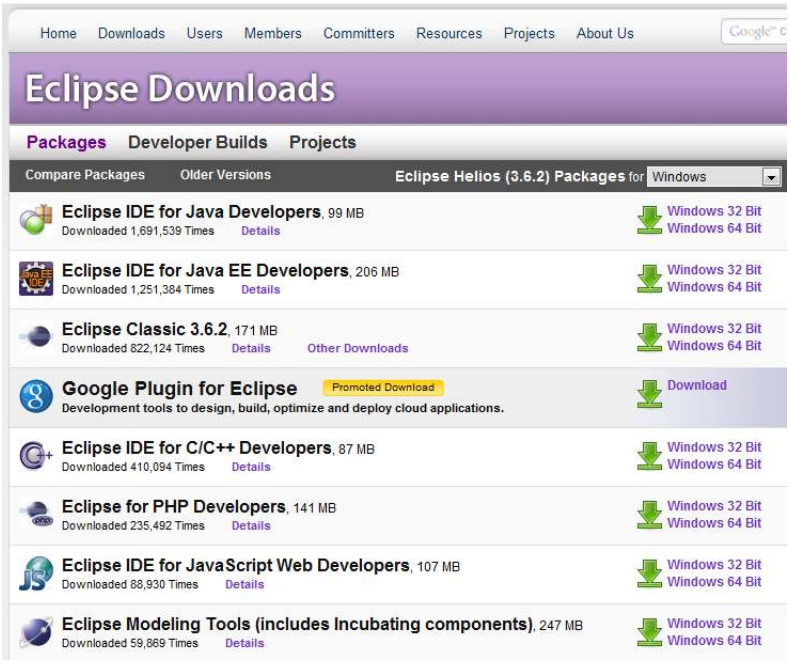


Figure 12 이클립스 다운로드 페이지

A. 이클립스 자바 프로젝트 생성

다운로드 받은 한나눔 릴리즈 (또는 SVN Repository 로부터 다운로드 받은 프로그램)을 이용하여 이클립스 자바 프로젝트를 생성한다.

- Eclipse Java Workspace 에 한나눔 자바 버전 프로그램을 위치시킨다.
  예) JAVA_WORKSPACE/JHanNanum-0.8.3
- Eclipse 상단 메뉴에서 File > New > Java Project 선택.
- Java Workspace 가 한나눔이 위치한 경로와 일치하는지 확인 후, Project Name 에 한나눔의 디렉토리 명을 입력한다
  예) JHanNanum-0.8.3
- Finish 버튼을 눌러 프로젝트 생성을 완료한다.

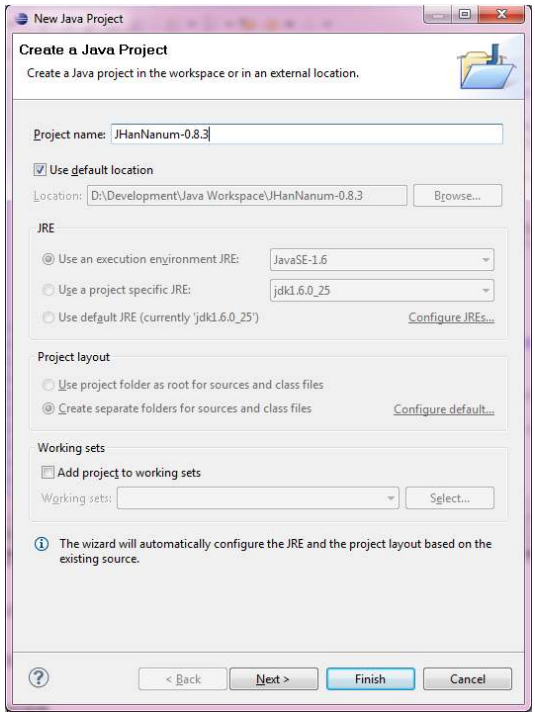


Figure 13 한나눔 프로젝트 생성

B. WorkflowHmmPosTagger 데모 프로그램 실행

- kr.ac.kaist.swrc.jhannanum.demo.WorkflowHmmPosTagger
- WorkflowHmmPosTagger.java 파일을 열고 "Menu > Run > Run" 을 클릭 또는 "Ctrl + F11"을 입력하여 프로그램을 실행

C. 결과 확인

이클립스 Console 창을 통해서 예제 문장에 대한 분석 결과 화면을 확인 할 수 있다.

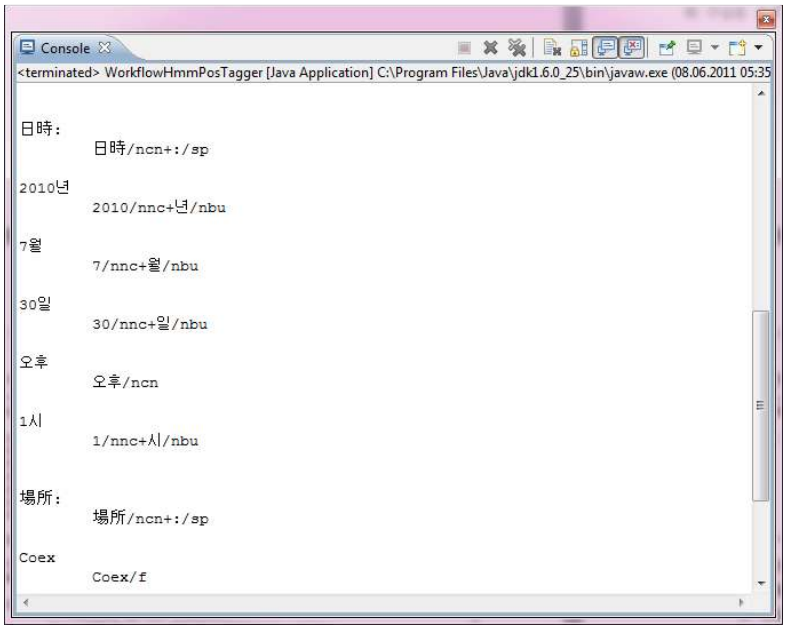


Figure 14 WorkflowHmmPosTagger 데모 실행 결과

4.3.2 GUIDemo 실행하기

KLDP 에 등록된 한나눔 릴리즈는 GUIDemo 를 포함하고 있다. GUIDemo 의 구성은 다음과 같다.

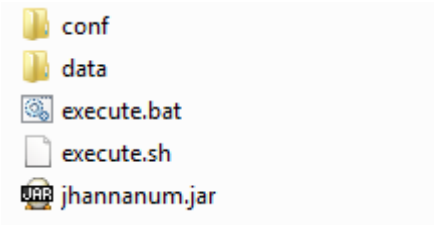


Figure 15 GUIDemo 구성

플랫폼에 따라 execute.bat 또는 execute.sh 를 실행하면 다음과 같이 GUI 기반 데모 프로그램이 실행된다.

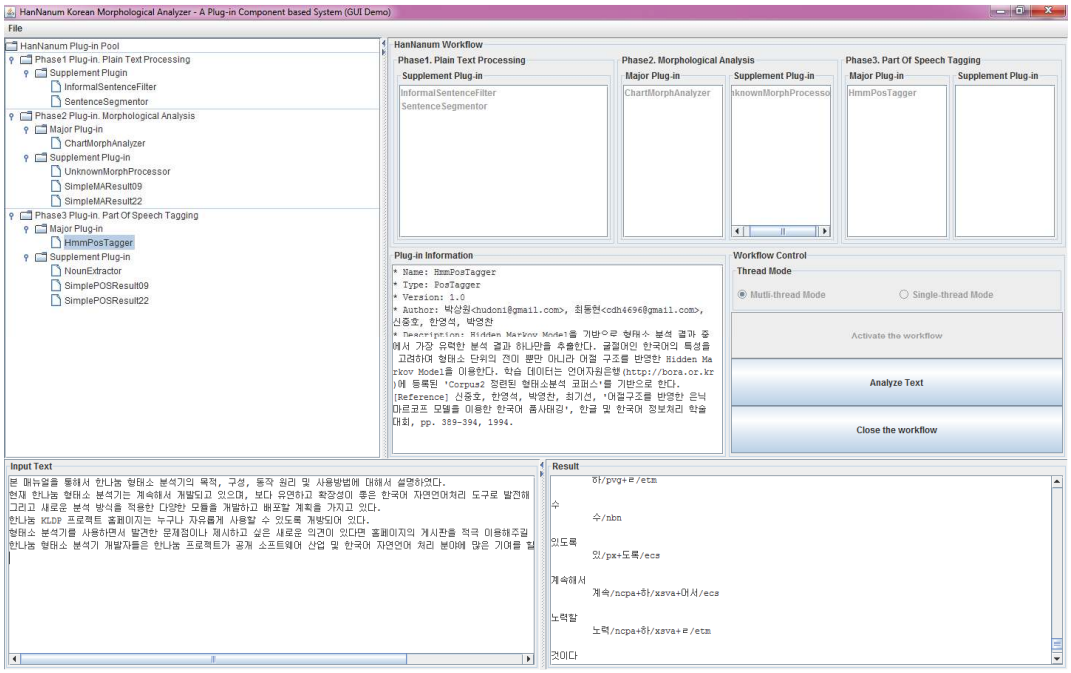


Figure 16 GUIDemo 실행 예

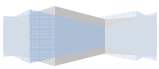
사용 방법은 다음과 같다.

1. Tree 형태의 Plug-in Pool 에 등록된 plug-in 들을 확인한다. Plug-in 을 click 하면 Plug-in Information 항목에 간략한 설명이 나타나므로 work flow 구성시 참고할 수 있다.
2. 선택한 Plug-in 을 drag-and-drop 방식으로 work flow 에 배치시킨다. 이 때 plug-in 의 phase 와 type 이 일치해야 하므로 주의해야 한다.
3. Work flow 설정이 끝나면 'Multi-thread mode' 또는 'Single-thread mode'를 선택한다.
4. 'Activates the work flow' 버튼을 누르면 설정한 work flow 가 활성화 된다.
5. 'Input Text' 부분에 분석하고자 하는 텍스트를 입력 또는 복사하여 붙여넣는다. File – Open 메뉴를 이용하는 것도 가능하다.
6. 'Analyze Text' 버튼을 눌러 활성화된 work flow 를 이용하여 분석을 수행한다.
7. 'Result' 영역에 나타난 분석 결과를 확인한다.
8. 5~7 단계를 반복하며 활성화된 work flow 를 반복적으로 이용한다.
   또는 'Close the work flow' 버튼을 누르고 1 단계로 돌아가 새로운 work flow 를 설정하여 테스트한다.

4.4 한나눔 라이브러리를 이용한 프로그램 작성

한나눔 릴리즈에 포함된 jhannanum-0.8.3.jar 파일을 라이브러리로 등록시키면 단 몇 줄의 코드만으로 형태소 분석 및 품사 태깅등의 한국어 분석 결과를 활용할 수 있다. 앞에서 소개된 데모 프로그램들을 확인한 이후에 JAVADOC 을 참고하면 어렵지 않게 라이브러리를 이용할 수 있을 것이다.

- ManualWorkflowSetUp.java 에는 plug-in 을 work flow 에 배치하여 이용하는 코드와 상세한 주석이 기술되어 있으니 라이브러리 이용시 참고하면 된다.
- WorkflowNounExtractor.java 에는 분석 결과를 문자열이 아닌 객체 형태로 결과를 받아오는 프로그램 코드가 작성되어 있다. 문자열로 반환된 결과를 다시 parsing 하여 이용하는 수고를 덜기 위해서는 이 예제 프로그램을 참고하면 된다.
- kr.ac.kaist.swrc.jhannanum.hannanum.WorkflowFactory 에는 미리 정의된 대표적인 work flow 들이 존재하므로 간편한 방법으로 work flow 를 이용할 수 있다. Workflow***.java 예제 프로그램들을 참고하면 된다.



4.5 새로운 한나눔 Plug-in 작성하여 활용하기

원하는 기능의 plug-in 이 존재하지 않는다면 직접 plug-in 을 개발하여 사용하는 것이 가능하다. 전체 시스템의 구성을 분석할 필요없이 개발하고자 하는 플러그인의 기능과 입출력 형태만 고려하면 되고, 또 기존의 플러그인들과 함께 work flow 를 구성하여 활용할 수 있으므로 효율적으로 새로운 기능을 추가 및 테스트 할 수 있다. 새로운 플러그인을 개발하기 위해서는 다음과 같은 단계를 거치면 된다.

1. 플러그인의 분석 단계와 타입을 결정한다. (3.1 한나눔 workflow 참조)
2. 분석 단계와 타입에 맞는 플러그인 자바 인터페이스를 확인하여 구현한다. (이미 구현된 플러그인 및 JAVADOC 참고)

   Package kr.ac.kaist.swrc.jhannanum.plugin.*

   Phase 1. Supplement Plug-in:
   SupplementPlugin.PlainTextProcessor.PlainTextProcessor.java

   Phase 2. Major Plug-in:
   MajorPlugin.MorphAnalyzer.MorphAnalyzer.java

   Phase 2. Supplement Plug-in:
   SupplementPlugin.MorphemeProcessor.MorphemeProcessor.java

   Phase 3. Major Plug-in:
   MajorPlugin.PosTagger.PosTagger.java

   Phase 3. Supplement Plug-in:
   SupplementPlugin.PosProcessor.PosProcessor.java

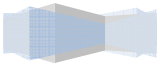
   예) ChartMorphAnalyzer (Phase 2. Major Plug-in – MorphAnalyzer 구현)

```
public class ChartMorphAnalyzer implements MorphAnalyzer {

   @Override
   public SetOfSentences morphAnalyze(PlainSentence ps) {
      …
   }

   …
}
```

3. JSON 포멧의 설정 파일을 작성한다. (conf/ 디렉토리 내 *.json 파일 참조)
4. 구현한 plug-in 을 다른 한나눔 사용자들과 공유하고 싶다면 hudoni@world.kaist.ac.kr 로 메일을 보낸다.



5. 라이센스

한나눔 형태소 분석기는 GPLv3 라이센스를 따른다.

- GPLv3 라이센스: http://www.gnu.org/licenses/gpl.html

6. 맺음말

본 매뉴얼을 통해서 한나눔 형태소 분석기의 목적, 구성, 동작 원리 및 사용방법에 대해서 설명하였다. 현재 한나눔 형태소 분석기는 계속해서 개발되고 있으며, 보다 유연하고 확장성이 좋은 한국어 자연언어처리 도구로 발전해 나가는 것을 목표로 하고 있다. 그리고 새로운 분석 방식을 적용한 다양한 plug-in 을 개발하고 배포할 계획을 가지고 있다. 한나눔 KLDP 프로젝트 홈페이지는 누구나 자유롭게 사용할 수 있도록 개방되어 있다. 형태소 분석기를 사용하면서 발견한 문제점이나 제시하고 싶은 새로운 의견이 있다면 홈페이지의 게시판을 적극 이용해주길 바란다. 한나눔 형태소 분석기 개발자들은 한나눔 프로젝트가 공개 소프트웨어 산업 및 한국어 자연언어 처리 분야에 많은 기여를 할 수 있도록 계속해서 노력할 것이다.

7 참고문헌

[1] 이운재, 김선배, 김길연, 최기선, "모듈화된 형태소 분석기의 구현", 한국정보과학회 언어공학연구회 학술발표 논문집, pp. 123-136, 1999.

[2] 신중호, 한영석, 박영찬, 최기선, "어절구조를 반영한 은닉 마르코프 모델을 이용한 한국어 품사태깅", 한글 및 한국어 정보처리 학술대회, pp. 389-394, 1994.

[3] 이하규, 김영택, "통계정보에 기반을 둔 한국어 어휘중의성해소", 한국통신학회논문지 '94-2 Vol.19 No.2, 1994

[4] 이상호, 김재훈, 조정미, 서정연, "부분 분석 결과를 공유하는 한국어 형태소 분석", 제 11회 음성통신 및 신호처리 워크샵 논문집, pp. 75-79, 1994.

[5] 이은철, 이종혁, "계층적 기호 접속정보를 이용한 한국어 형태소 분석기의 구현", 제 4 회 한글 및 한국어 정보처리 학술대회 논문집, pp. 95-104, 1992.

[6] 김성용, "TABULAR PARSING 방법과 접속 정보를 이용한 한국어 형태소 분석기", 석사학위논문, 한국과학기술원, 1987.

[7] 언어자원은행, http://www.bora.or.kr/