한글 처리

인천대학교 컴퓨터공학부
교수 홍 윤식

왜 한글 코드를 알아야 하는가?

- 세종대왕이 어여삐 여긴 백성의 후손
  - 나라 말씀이 없었다면
    - 필리핀이 우리의 모습일까? 홍콩이 우리의 모습일까?
- 지금보다 더 편하게, 더 쉽게 내 생각을 글자로 나타내기 위해
  - 모든 지식을 자유롭게 공유하기 위해
- 구전(口傳)은 소문으로 끝나지만,
- 활자로 된 소식은 역사(歷史)가 된다!

기억할 만한 한글 코드

- **N-bytes (multi-bytes) 한글 코드**
  - 자음과 모음에 대해 각각 1바이트씩 할당
    - 음절당 최대 5 바이트가 필요
      - 가 = ㄱ + ㅏ → 2 바이트
      - 꽥 = ㄲ + ㅗ + ㅏ + ㅣ + ㄱ → 5 바이트
- **KSC5601-1987 완성형 코드**
  - 2바이트 체계 : 94×94(=8,836) 문자 집합
    - 한글 2,350자, 한자 4,888자, 특수문자 1,128자 등
      - 한글 음절 11,172자 중 약 21%에 불과
- **상용 조합형 코드**
  - 2바이트 체계
    - 음절을 초성, 중성, 종성으로 구분하고 각 5비트씩 할당
    - 최상위 bit를 '1'로 설정 → 한글과 영문자 구분
- **KSC5601-1992 조합형 코드**

유니코드(ISO/IEC 10646)

- **개요**
  - http://www.unicode.org
  - 여러 나라의 기존 표준 문자 코드를 기반으로 하여 설계된 통합 다국어 문자 코드 체계 (32비트 character set)
  - 유니코드는 단지 정수를 문자에 할당하는 코드 테이블
    - ISO(국제 표준 기구)와 유니코드 컨소시엄에서 공동 개발
  - 특수문자를 제외한 각 국가 문자들은 하위 16 bits 영역(BMP : Basic Multilingual Plane)에 정의
- **코드 표현법**
  - 접두사 "U-"를 붙여서 표현(최대 U-7FFFFFFF까지 정의)
  - 하위 16 bits만 나타낼 경우 접두사 "U+" 사용
    - 영역: U+0000 ~ U+7FFF (ISO 8859-1(Latin-1)과 같음)
  - 알파벳 'A'는 "U+0x0041", 한글의 '가'는 "U+0xAC00"로 표현

IEC: International Electro-technical Commission

Encoding Forms

- 3가지 encoding form을 규정
- **UTF-8** : 8-bits per code unit
  - 가변 길이 문자 encoding 방식을 사용
  - ASCII 문자는 1-byte Unicode 문자와 호환

- **UTF-16** : 16-bits per code unit
  - 가장 많이 사용하는 형태
  - 메모리 크기도 중간이면서, 코드를 효과적으로 access.
    - 32비트 중 16비트만 사용해서도 해당 문자 체계의 코드를 access.

- **UTF-32** : 32bits per code unit
  - 메모리를 많이 차지하긴 하지만,
  - 32bit 로 encoding되기 때문에 크기가 고정.

utf-8: unicode transformation format(가변길이 문자 인코딩 방식)

한글 표준 코드

- 연혁
  - 1987: KS C 5601-1987 (완성형 한글)
    - 한글 2,350자, 한자 4,888자, 특수문자 987자
  - 1991: KS C 5657-1991 (완성형 확장 글자)
  - 1992: KS C 5601-1992 (조합형 한글)
  - 1995: KS C 5700-1995 (완성형 한글)
    - 한글 11,172자, 조합형 자모 334자, 한자 23,000자
  - 1998: KS X 1001 (KS C 5601-1987 신규격)
    - 한글 2,350자, 한자 4,888자, 특수문자 1,128자, 사용자정의 188자, 미지정 문자 282자
    - KS X 1001 기반 문자 인코딩으로는 EUC-KR(완성형), CP949, 2바이트 상용 조합형 등이 있음
    - 현재 "KS X 1001:2002" 이 최신 버전임
  - 2001: KS X 1002 (KS C 5657-1991 신규격)
    - 한글 11,172자 표현 가능(초성 19자, 중성 21자, 종성 27자 조합)
  - 2002: KS X 1005-1 (KS C 5700-1995의 신규격)
    - ISO 10646-1에 등록된 한글 코드를 그대로 표준코드로 정한 것

NIDA National Internet Development Agency of Korea

출처 : 한국인터넷진흥원

북한 한글 표준 코드

- 조선국규 KPS 9566-2003 (KS X 1001과 유사)
  - 총 16,776자(한글은 남한보다 329글자 정도 더 많음)
  - 유니코드와 왕복 변환(round trip) 변환 불가
    - 국규 9566-2003에는 “김, 일, 성, 김, 정, 일” 6글자가 특수문자 부분에 일렬 배치
    - 국규 9566-2003의 16,776자 중 22자가 유니코드에 미포함
    - 유니코드의 자모 배치 순서나 음절 배치 순서는 대한민국의 표준이 반영되어 있음
- 남북한 한글 표준 코드 차이점
  - 북한은 받침으로 가는 `o`과 모음 앞에 오는 음가 없는 `o'을 구분함
  - 정렬순서가 "ㄱㄴㄷㄹㅁㅂㅅㅈㅊㅋㅌㅍㅎㄲㄸㅃㅆㅉㅇ"으로 "ㅇ"이 맨 마지막임
- OS 및 응용프로그램 지원 현황
  - OS(MS 윈도우, Linux, Unix 등) 레벨에서는 북한 한글 코드 비지원

NIDA National Internet Development Agency of Korea

출처 : 한국인터넷진흥원

한글 Unicode

parts		Unicode specification
Hangul Jamo	**한글 자모** (조합형)	U+1100 ~ U+11F9
Hangul Compatibility Jamo	**한글 호환 자모** (자음 또는 모음)	U+3131 ~ U+318E
Hangul Jamo Extended A	한글 자모 확장 A	U+A960 ~ U+A97C
Hangul Syllables	**한글 음절**(완성형)	U+AC00 ~ U+D7A3
Hangul Jamo Extended B	한글 자모 확장 B	U+D7B0 ~ U+D7FF
Halfwidth Jamo	**반각 자모**	U+FF00 ~ U+FFEF

한글 Unicode 영역

- **한글 자모 (Hangul Jamo) : 첫가끝 한글**
  - 범위 : 한글 초성 ㄱ(U+1100) ~ 한글 종성, 여린 히읗 (U+11F9) 총 240자
  - 초성 자음/중성 모음/종성 자음으로 각각 나누어 한 글자씩 대응
  - 한글 한 글자를 표현하는데 많은 저장 공간이 필요
- **한글 호환 자모 (Hangul Compatibility Jamo)**
  - 범위 : ㄱ(U+3131) ~ 아래아 ㆍ(U+318E)
  - 초/중/종성을 구분하지 않고 사용되는 모든 자모들을 한데 묶어 놓음
    - keyboard로 한글 자모를 입력할 때 사용됨
- **한글 음절 (Hangul Syllables)**
  - 범위 : 가(U+AC00) ~ 힣(U+D7A3)
  - 현대 한글 자모로 표현 가능한 모든 한글 문자(11,172자)
    - 초성 19자, 중성 21자, 종성 27자로 조합 가능한 글자의 수
- **한글 반각 자모 (Halfwidth Jamo / Halfwidth Hangul variants)**
  - 범위 : (반각) ㄱ (FFA1) ~ (반각) ㅣ (FFDC)

한글 Unicode (1/2)

- 초성(19자)
  - ㄱ(0) ㄲ(1) ㄴ(2) ㄷ(3) ㄸ ㄹ ㅁ ㅂ ㅃ ㅅ ㅆ ㅇ ㅈ ㅉ ㅊ ㅋ (15) ㅌ(16) ㅍ(17) ㅎ(18)
- 중성(21자)
  - ㅏ(0) ㅐ(1) ㅑ(2) ㅒ(3) ㅓ ㅔ ㅕ ㅖ ㅗ ㅘ ㅙ ㅚ ㅛ ㅜ ㅝ ㅞ ㅟ ㅠ(17) ㅡ(18) ㅢ(19) ㅣ(20)
- 종성(28자)
  - 없음(0) ㄱ(1) ㄲ(2) ㄳ ㄴ ㄵ ㄶ ㄷ ㄹ ㄺ ㄻ ㄼ ㄽ ㄾ ㄿ ㅀ ㅁ ㅂ ㅄ ㅅ ㅆ ㅇ ㅈ ㅊ ㅋ ㅌ ㅍ(26) ㅎ(27)

한글 Unicode (2/2)

- 한글 **Unicode** 구하는 공식
  - 초성x21x28 + 중성x28 + 종성 + 44032
    - 44032 : 0xAC00 (‘가’)
      - 초성 :ㄱ(0), 중성:ㅏ(0), 종성: 없음(0)
    - 55203 : 0xD7A3(‘힣 ‘)
      - 초성 :ㅎ(18), 중성:ㅣ(20), 종성: ㅎ(27)
      - 18x21x28 + 20x28 + 27 + 44032 = 55203
  - 모두 11,172자의 한글 문자가 존재
    - 19x21x28 = 11172
    - 55203-44032+1 = 11172

Python에서 한글 코드 변환

- python에서는 utf-8 코드 체계를 사용
  - utf-8 : unicode를 완벽하게 지원하는 코드 체계

- 파일을 읽어오거나 처리 결과를 파일로 저장할 때 해당 문자열의 인코딩 방식에 맞는 변환이 필요.
  - decode : latin-2, utf-8 등으로 인코딩한 파일
    - 파이썬 내부로 가져올 때 unicode로 변환하는 과정.
  - encode : 파이썬 내부에서 처리한 unicode 문자열
    - utf-8, cp949 등으로 인코딩해서 파일로 저장하는 과정.
    - cp949는 완성형 코드
    - utf-8(unicode)은 조합형 코드

unicode에서 한글 음절

- unicode에서 한글 음절은
  - 19(초성)x21(중성)x28(종성) = 11,172자
  - 첫 음절은 '가', 마지막 음절은 '힣'.
    - 첫 음절 '가'의 코드: 44032(0xAC00)
    - 마지막 음절 '힣'의 코드: 55203(0xD7A3)
  - 초성 시작 위치: 0x1100(4352)
  - 중성 시작 위치: 0x1161(4449)
  - 종성 시작 위치: 0x11A7(4519)

한글 음절에서 자모 분리 (1/2)

- 이름의 자모를 분리 : '홍' ➔ ㅎ + ㅗ + ㅇ
- Algorithm : 변환 공식의 역순 적용
  - 한글 음절 유니코드 = 초성x21x28 + 중성x28 +종성+ 44032
  - 단계 1: 유니코드에서 offset (44032)를 뺀다
    - 54861 – 44032 = 초성x21x28 + 중성x28 +종성
  - 단계 2: 종성 구하기
    - (초성x21x28 + 중성x28 +종성) **% 28** = 종성
      - Modulus연산 결과 초성, 중성 항은 모두 0 (모두 28의 배수)
    - (초성x21x28 + 중성x28 +종성) **/ 28** = 초성x21 + 중성
      - **정수 나눗셈** ➔ 28의 배수가 아니면 0
  - 단계 3: 중성, 초성 구하기
    - (초성x21 + 중성) **% 21** = 중성
    - (초성x21 + 중성) **/ 21** = 초성

한글 음절에서 자모 분리 (2/2)

```
def separateJamo (name):
    result = []

    for syllable in name:
        code = ord(syllable)
        if code >= 44032 and code <= 55203:
        #if syllable >= '가' and syllable <= '힣':
        #if re.match('[ㄱ-ㅎㅏ-ㅣ가-힣]', syllable) is not None:
            #code = ord(syllable) - 44032
            code -= 44032
            jong = int(code % 28)

            code = int(code / 28)
            joong = int(code % 21)

            cho = int(code / 21)
            result.append(uni_choSung[cho])
            result.append(uni_joongSung[joong])
            result.append(uni_jongSung[jong])
        else:
            result.append(syllable)
    print(result)
```

한글 자판과 오토마타

- 2벌식, 3벌식 자판이란?
  - 한글 자판이 다르면 한글 오토마타도 다르다?

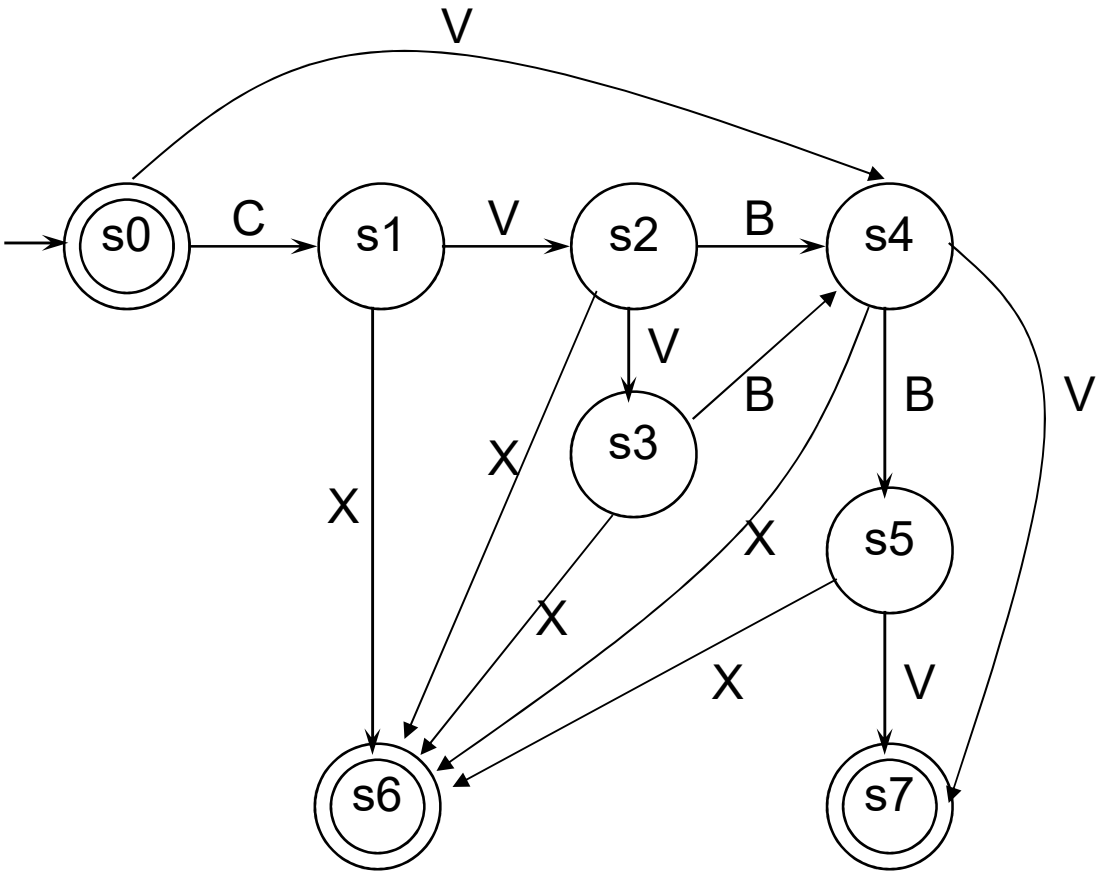
- 스마트 폰 한글 오토마타는 desktop PC 한글 오토마타와 같을까 다를까?
  - 천지인 한글 자판으로 입력해 보자.
    - soft keypad 자음은 왜 그렇게 배치했을까?
  - i-Phone의 QWERTY 자판으로 입력해 보자.

한글 자판 2벌식과 3벌식

현행 표준 자판. – 2벌식
한글 입력 시 자음과 모음만을 구분해서 입력.
초성과 종성을 구분하지 않기 때문에
초성의 "ㄱ"과 종성의 "ㄱ" 의 글꼴이 다르지
만, 입력할 방법은 없다.

공병우 박사가 개발
초성, 중성, 종성의 3벌로 나누어 한글을
입력. 2벌식 자판에 비해 사용하는 키가
많아서 키보드 상단의 숫자 키에까지 한글
자소가 배열되어 있음.

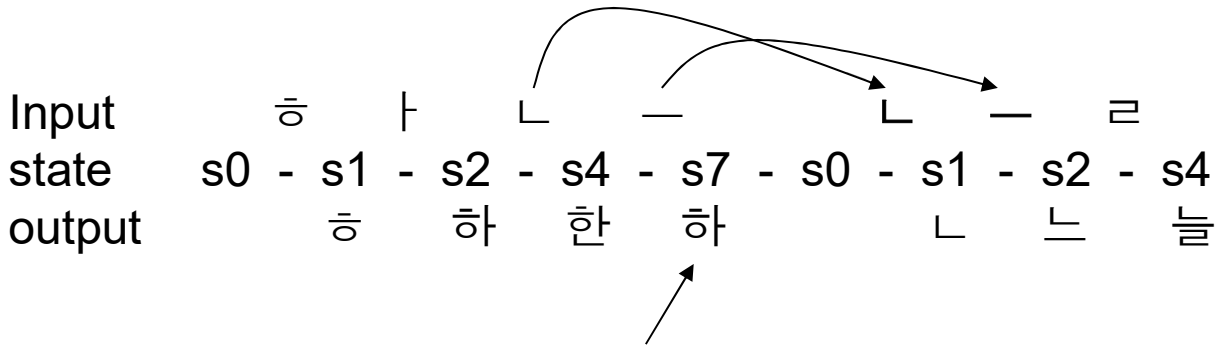
2벌식 한글 오토마타(1/2)



s0 : 초기 상태
s1 : 자음이 입력(초성)된 상태
s2 : 모음이 입력된 상태
s3 : s2에 이어 모음이 하나 더 입력(즉, *겹모음*)된 상태
s4 : s3에 자음이 입력되어 *받침*으로 처리된 상태
s5 : s4에 자음이 하나 더 입력되어 *겹받침*으로 처리된 상태
s6 : 한 글자 완성된 상태
s7 : 모음이 입력되어 앞 글자의 받침과 분리되면서 한 글자가 완성된 상태

2벌식 한글 오토마타(2/2)

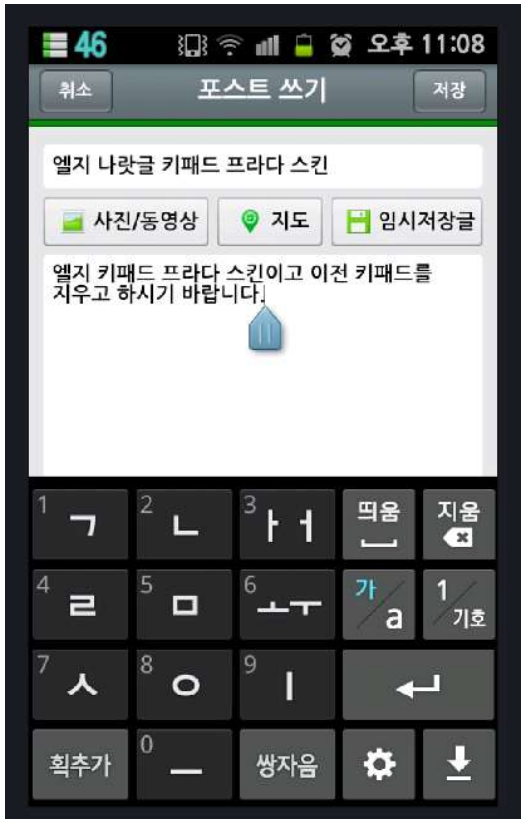
C(consonant): 초성이 될 수 있는 자음(단자음 **+** 쌍자음(ㄲ,ㄸ,ㅃ,ㅆ,ㅉ))
V(vowel) : 모음(단모음 **+** 겹모음(ㅐ,ㅒ, ㅔ,ㅖ)
B : 받침이 될 수 있는 자음(단자음 **+** 쌍자음(ㄲ,ㅆ))
X : **C,V,B**에 해당되지 않는 기호



받침이 분리되어 한 글자가 완성된 상태

스마트 폰에서의 한글 입력

천지인 키패드



나랏글 키패드

스카이 키패드