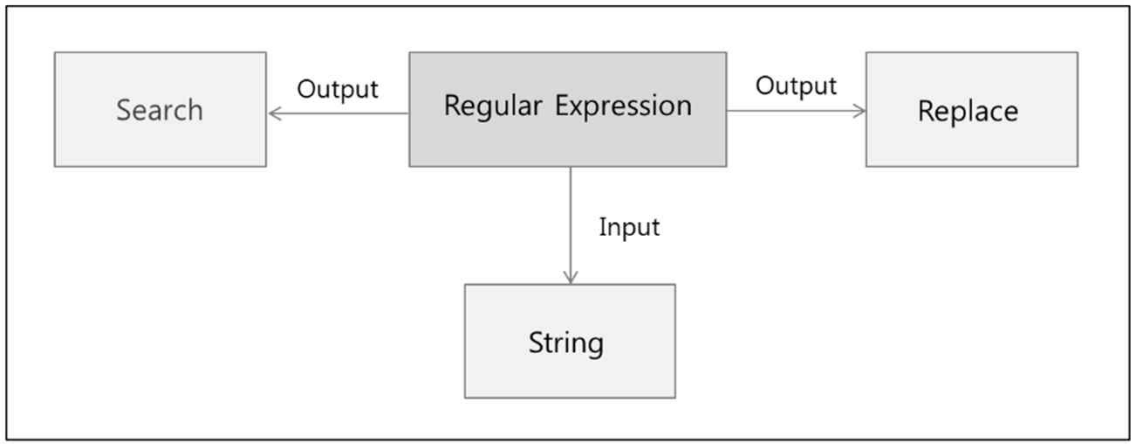
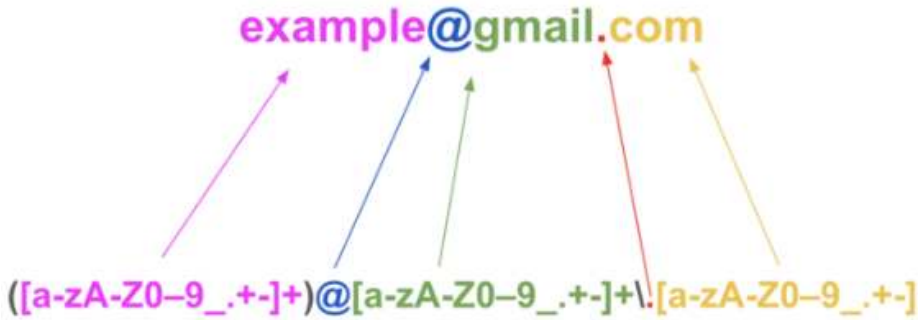
정규표현식
Regular Expression

정규표현식

- 특정한 규칙을 가진 문자열의 집합을 표현하는데 사용하는 형식 언어
- Programming Language나 Text Editor 등 에서 문자열의 검색과 치환을 위한 용도로 사용
  - 입력한 문자열에서 특정한 조건을 표현할 경우 일반적인 조건문으로는 다소 복잡할 수도 있지만, 정규표현식을 이용하면 매우 간단하게 표현 가능
  - 코드가 간단한 만큼 가독성이 떨어져서 표현식을 숙지하지 않으면 이해하기 어려움





정규식(Regular expressions, Regex 또는 Regexp)

- 특정 검색 패턴(ASCII 또는 유니코드 문자의 시퀀스)에 대한 하나 이상의 일치 항목을 검색하여 텍스트에서 정보를 추출하는데 매우 유용
- 응용분야
  - 유효성 검사
  - 문자열 파싱 및 대체
  - 데이터를 다른 형식으로 변환
  - 웹 스크래핑
- 가장 중요한 점은 정규식을 학습한 후에는 거의 모든 프로그래밍 언어(Javascript, Java, VB, C#, C, C++, Python, Perl, Ruby, Delphi, R 등등)에서 동일하게 사용할 수 있다는 것
  - 엔진이 지원하는 문법버전 이나 최신특징에 따라서 약간의 차이

기본 개념

- 패턴(pattern)
  - 정규 표현식은 특정 목적을 위해 필요한 문자열 집합을 지정하기 위해 쓰이는 식
  - 문자열의 유한 집합을 지정하는 단순한 방법은 문자열의 요소나 멤버를 나열하는 것
- 일치(match)
  - 패턴은 3개의 문자열을 각각 일치(match)시킨다고 이야기한다.
- 메타 문자
  - 본래 문자가 가진 뜻이 아닌 특별한 용도로 사용하는 문자
  - . ^ $ * + ? { } [ ] \ | ( )

정규식 기본 기호

기호	설명	예제
.	임의의 문자 1개를 의미	줄바꿈 문자인 ₩n을 제외한 모든 문자와 매치
^	시작을 의미한다 [] 괄호 안에 있다면 일치하지 않는 부정의 의미로로 쓰인다	^a : a로 시작하는 단어 [^a] : a가 아닌 철자인 문자 1개
$	$앞의 문자열로 문자가 끝나는지를 의미한다	a$: a로 끝나는 단어
[]	[] 괄호 안의 문자가 있는지를 확인한다 Bracket expressions	[ab][cd] : a,b중 한 문자와 c,d중 한 문자 → ac ad bc bd
[^]	[] 대괄호 안에 ^ 문자가 있으면, 제외를 뜻함 - 대괄호 안에 ^ 가 쓰이면 제외의 뜻 - 대괄호 밖에 ^ 가 쓰이면 시작점의 뜻	[^a-z] : 알파벳 소문자 a부터 z까지를 제외한 모든 문자
-	사이의 문자 혹은 숫자를 의미한다	[a-z] : 알파벳 소문자 a부터 z까지 하나 [a-z0-9] : 알파벳 소문자 전체,0~9 중 한 문자
|	또는	[a|b] : a 혹은 b
()	그룹	01(0|1) : 01뒤에 0 또는 1이 들어간다 → 010(o), 011(o), 012(x)
{}	개수	a{3}b : a가 3번 온 후 b가 온다 → aab(x), aaab(o), aaaab(o)

정규식 기본 기호

기호	설명	예제
₩b	공백, 탭, ",", "/" 등을 의미한다	apple₩b : apple뒤에 공백 탭등이 있다 → apple juice (o), apple.com (x)
₩B	₩b의 부정 공백 탭등이 아닌 문자인 경우 매치한다	apple₩B → apple.com (o)
₩d	0~9 사이의 숫자 [0-9]와 동일	
₩D	₩d의 부정 숫자가 아닌 어떤 문자, [^0-9]와 동일	
₩s	공백, 탭, whitespace 문자와 매치 [₩t₩n₩r₩f₩v] - 맨 앞의 빈 칸은 공백문자(space)를 의미	
₩S	공백, 탭이 아닌 문자	
₩w	알파벳 대소문자+숫자(alphanumeric)인 문자와 매치 [a-zA-Z_0-9]와 동일	
₩W	₩w의 부정 [^a-zA-Z_0-9]	
₩t	탭	

Anchor(앵커)

메타 문자	설명
^	문자열이나 행의 처음을 의미한다.
$	문자열이나 행의 끝을 의미한다.

- ^The
  - The 로 시작하는 모든 문자열을 매칭
- end$
  - end로 끝나는 문자열과 매칭합니다
- ^The end$
  - The end와 정확하게 일치하는 문자열을 매칭
- gray
  - gray가 들어있는 모든 문자열과 매칭

Quantifier(수량자)

메타 문자	설명
{}	반복
*	0개 이상의 c 를 포함한 문자열과 매칭
+	1개 이상의 c 를 포함한 문자열과 매칭
?	0개 또는 1개의 c 를 포함한 문자열과 매칭

수량자	설명
{n}	정확히 n개만을 찾습니다.
{n,}	n개 이상을 찾습니다.
{n,m}	최소 n개, 최대 m개의 경우를 찾습니다.

Quantifier(수량자)

기호	설명	예제
?	앞의 표현식이 없거나 or 최대 한개만	a1? : 1이 최대 한개만 있거나 없을수도 있다 → a(o), a1(o), a11(x), a111(x)
*	앞의 표현식이 없거나 or 있거나 (여러개)	a1* : 1이 있을수도 없을수도 있다 → a(o), a1(o), a11(o), a111(o)
+	앞의 표현식이 1개 이상 or 여러개	a1+ : 1이 1개 이상있다 → a(x), a1(o), a11(o), a111(o)
{n}	딱 n개 있다	a{3} : a가 딱 3개 있다 → aa(x), aaa(o), aaaa(x), aaaaaaa(x)
{n, m}	n개 이상 m개 이하	a{3,5} : a가 3개 이상 5개 이하 있다 → aa(x), aaa(o), aaaa(o), aaaaaaa(x)
{n,}	n개 이상	a{3,} : a가 3개 이상 있다 → aa(x), aaa(o), aaaa(o), aaaaaaa(o)

*?, +?, ??

Greedy and Lazy match

- Greedy = "최대한 많이"
- Lazy = "최대한 적게"
  - **?**

Grouping and capturing(그룹 캡쳐)

기호	설명
()	그룹 및 캡쳐
(?:)	찾지만 그룹에 포함 안됨
(?=)	=앞 문자를 기준으로 전방 탐색
(?<=)	=뒤 문자를 기준으로 후방 탐색

Look-ahead and Look-behind

- 전방 탐색
  - **(?=)**
  - d(?=r) third drone
- 후방 탐색
  - **(?<=)**
  - (?<=r)d third drone

Flags

- 정규식은 보통 /abc/와 같은 형식을 사용
- 두번째 슬래쉬 뒤에 플래그를 사용 가능
- g(global)
  - 문자열에서 첫번째 매칭 후, 끝나지 않고 매칭되는 모든 항목을 조회
- m(multi-line)
  - anchor(^ 또는 $)가 문자열 전체가 아닌, 줄 각각에 매칭하여 줄별로 정규식 패턴을 매칭
- i(insensitive)
  - 대소문자 구분을 무시하고 매칭

Boundaries

- ₩b and ₩B
- ₩b는 anchor 문자(^ 또는 $)와 유사하며 한쪽은 문자를 의미하며, 다른 쪽은 문자가 아닌 단어와 매칭
- ₩B는 ₩b와 매치되지 않는 문자열과 매칭
  - 단어로 싸여있는 패턴을 찾을 때 사용

- ₩babc₩b abc와 동일한 문자를 매칭

Back-references

- ([abc])₩1
  - ₩1은 첫번째 캡쳐그룹과 동일한 패턴을 의미
- ([abc])([de])₩2₩1
  - ₩2는 두번째 캡쳐그룹과 동일한 패턴을 의미
  - ([abc])([de])([de])([abc])와 동일

정규식 샘플

- 예문 검색
- Email
- 전화번호
- ~~우편번호~~
- ~~주민등록번호~~

자바의 정규식

- String class
- regex package
  - Pattern class
  - Matcher class

String 정규식 메서드

메서드	설명
boolean matches(String regex)	인자로 주어진 정규식에 매칭되는 값이 있는지 확인
String replaceAll(String regex, String replacement)	문자열내에 있는 정규식 regex와 매치되는 모든 문자열을 replacement 문자열로 바꾼 문자열을 반환
String[] split(String regex)	인자로 주어진 정규식과 매치되는 문자열을 구분자로 분할

regex package

- Pattern class
- 문자열을 정규표현식 패턴 객체로 변환해주는 역할

Pattern 클래스 메서드	**설명**
compile(String regex)	정규표현식의 패턴을 작성
matches(String regex, CharSequence input)	정규표현식의 패턴과 문자열이 일치하는지 체크 일치할 경우 true, 일치하지 않는 경우 false를 리턴 (일부 문자열이 아닌 전체 문자열과 완벽히 일치 해야한다)
asPredicate()	문자열을 일치시키는 데 사용할 수있는 술어를 작성
pattern()	컴파일된 정규표현식을 String 형태로 반환
split(CharSequence input)	문자열을 주어진 인자값 CharSequence 패턴에 따라 분리

regex package

- Matcher class
- 대상 문자열의 패턴을 해석하고 주어진 패턴과 일치하는지 판별하고 반환된 필터링된 결과값들을 관리
- Pattern.compile() 을 통해 정규식문자열을 패턴 객체로 변환

Matcher class method

메서드	설 명
find( )	패턴이 일치하는 경우 true를 반환, 불일치하는 경우 false반환 (여러 개가 매칭되는 경우 반복 실행하면 일치하는 부분 다음부터 이어서 매칭됨)
find(int start)	start 위치 이후부터 매칭검색
start( )	매칭되는 문자열의 시작위치 반환
start(int group)	지정된 그룹이 매칭되는 시작위치 반환
end( )	매칭되는 문자열 끝위치의 다음 문자위치 반환
end(int group)	지정된 그룹이 매칭되는 끝위치의 다음 문자위치 반환
group( )	매칭된 부분을 반환
group(int group)	그룹화되어 매칭된 패턴중 group 번째 부분 반환
groupCount( )	괄호로 지정해서 그룹핑한 패턴의 전체 개수 반환
matches( )	패턴이 전체 문자열과 일치할 경우 true반환 (일부 문자열이 아닌 전체 문자열과 완벽히 일치 해야한다)

- https://www.nextree.co.kr/p4327/
- https://ko.wikipedia.org/wiki/%EC%A0%95%EA%B7%9C_%ED%91%9C%ED%98%84%EC%8B%9D
- https://wikidocs.net/4308
- https://chrisjune-13837.medium.com/%EC%A0%95%EA%B7%9C%EC%8B%9D-%ED%8A%9C%ED%86%A0%EB%A6%AC%EC%96%BC-%EC%98%88%EC%A0%9C%EB%A5%BC-%ED%86%B5%ED%95%9C-cheatsheet-%EB%B2%88%EC%97%AD-61c3099cdca8
- https://regex101.com/r/cO8lqs/1
- https://crazykim2.tistory.com/602
- https://blog.hexabrain.net/197
- https://inpa.tistory.com/entry/JAVA-%E2%98%95-%EC%A0%95%EA%B7%9C%EC%8B%9DRegular-Expression-%EC%82%AC%EC%9A%A9%EB%B2%95-%EC%A0%95%EB%A6%AC