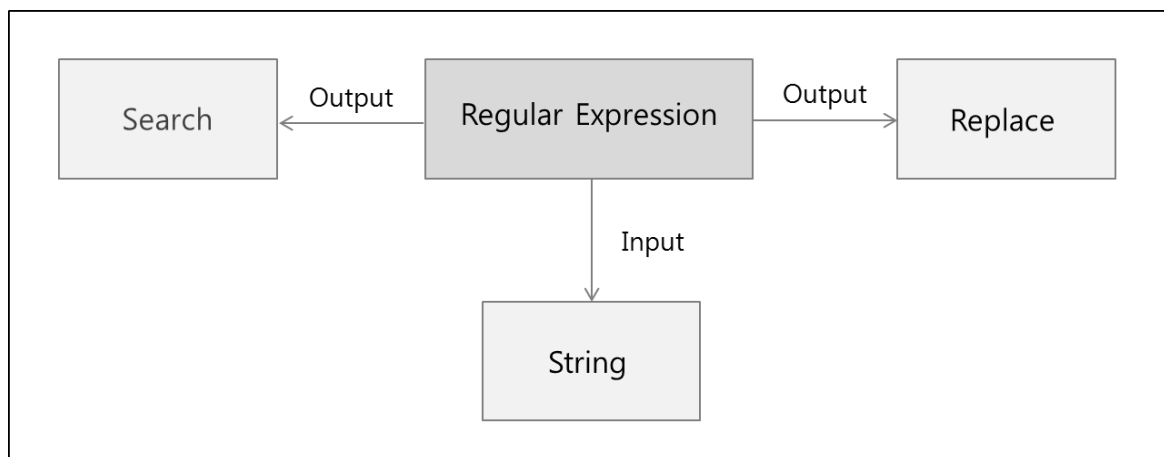


## 정규 표현식(Regular Expression)

날이 갈수록 개인정보 보호에 관련하여 보안정책을 점진적으로 강화하고 있습니다. 이에 따라 Web 에서 회원가입 시 Password 설정을 복잡해진 보안정책에 맞추다 보니 복잡하게 조합해야만 정상적으로 가입을 할 수 있습니다. 이러한 강화된 보안정책 때문에 기존에 사용하던 자신만의 Password 를 인위적으로 보안정책에 맞추는 경우가 많을 것입니다. 그러다 보니, 종종 Log-In 을 할 때 Password 를 잊어버려서 곤란한 상황이 발생하는 경우도 한번쯤은 있었을 것입니다. 일반적으로 이렇게 복잡한 조건이 필요한 경우 사용자에게 입력을 받을 때 여러 가지 조건을 주면서 정해진 규칙 안에서만 입력을 하도록 유도를 하고 있습니다.

### 정규표현식이란?

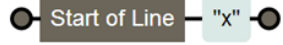
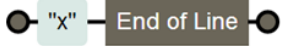
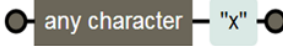
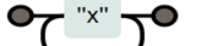
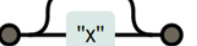



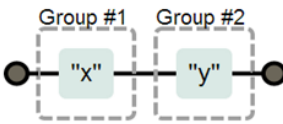
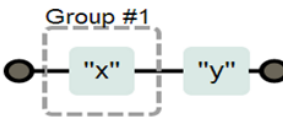
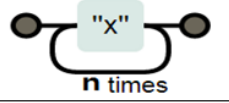
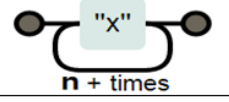
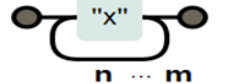
정규 표현식의 사전적인 의미로는 특정한 규칙을 가진 문자열의 집합을 표현하는데 사용하는 형식 언어입니다. 주로 Programming Language 나 Text Editor 등 에서 문자열의 검색과 치환을 위한 용도로 쓰이고 있습니다. 입력한 문자열에서 특정한 조건을 표현할 경우 일반적인 조건문으로는 다소 복잡할 수도 있지만, 정규 표현식을 이용하면 매우 간단하게 표현 할 수 있습니다. 하지만 코드가 간단한 만큼 가독성이 떨어져서 표현식을 숙지하지 않으면 이해하기 힘들다는 문제점이 있습니다.



Regular Expression UML

## 정규 표현식 표현방법

정규 표현식은 표준인 POSIX의 정규 표현식과 POSIX 정규 표현식에서 확장된 Perl 방식의 PCRE가 대표적이며, 이외에도 수많은 정규표현식이 존재하며 정규 표현식 간에는 약간의 차이점이 있으나 거의 비슷합니다. 정규 표현식에서 사용하는 기호를 Meta 문자라고 합니다. Meta 문자는 표현식 내부에서 특정한 의미를 갖는 문자를 말하며, 공통적인 기본 Meta 문자의 종류로는 다음과 같습니다.

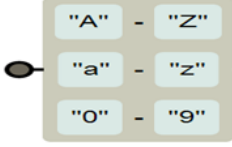
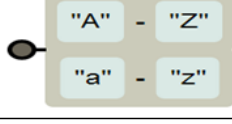
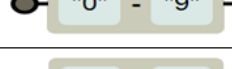

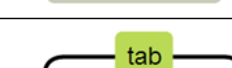

정규표현식	표현	설명
<code>^x</code>		문자열이 x로 시작합니다.
<code>x\$</code>		문자열이 x로 끝납니다.
<code>.x</code>		임의의 한 문자를 표현합니다. (x가 마지막으로 끝납니다.)
<code>x+</code>		x가 1번이상 반복합니다.
<code>x?</code>		x가 존재하거나 존재하지 않습니다.
<code>x*</code>		x가 0번이상 반복합니다.
<code>x y</code>		x 또는 y를 찾습니다. (or연산자를 의미합니다.)
<code>(x)</code>		( )안의 내용을 캡처하며, 그룹화 합니다.
<code>(x)(y)</code>		그룹화 할 때, 자동으로 앞에서부터 1번부터 그룹 번호를 부여해서 캡처합니다. 결과값에 그룹화한 Data가 배열 형식으로 그룹번호 순서대로 들어갑니다.
<code>(x)(?:y)</code>		캡처하지 않는 그룹을 생성할 경우 ?:를 사용합니다. 결과값 배열에 캡처하지 않는 그룹은 들어가지 않습니다.
<code>x{n}</code>		x를 n번 반복한 문자를 찾습니다.
<code>x{n,}</code>		x를 n번이상 반복한 문자를 찾습니다.
<code>x{n,m}</code>		x를 n번이상 m번이하 반복한 문자를 찾습니다.

Meta 문자 중에 독특한 성질을 지니고 있는 문자클래스 '[' ]'라는 문자가 있습니다. 문자클래스는 그 내부에 해당하는 문자열의 범위 중 한 문자만 선택한다는 의미이며, 문자클래스 내부에서는 Meta 문자를 사용할 수 없거나 의미가 다르게 사용됩니다.

정규표현식	표현	설명
[xy]		x,y중 하나를 찾습니다.
[^xy]		x,y를 제외하고 문자 하나를 찾습니다. (문자 클래스 내의 ^는 not을 의미합니다.)
[x-z]		x~z 사이의 문자중 하나를 찾습니다.
\w^		^(특수문자)를 식에 문자 자체로 포함합니다. (escape)
\wb		문자와 공백사이의 문자를 찾습니다.
\WB		문자와 공백사이가 아닌 값을 찾습니다.
\wd		숫자를 찾습니다.
\WD		숫자가 아닌 값을 찾습니다.
\ws		공백문자를 찾습니다.
\WS		공백이 아닌 문자를 찾습니다.
\wt		Tab 문자를 찾습니다.
\wv		Vertical Tab 문자를 찾습니다.
\ww		알파벳 + 숫자 + _ 를 찾습니다.
\WW		알파벳 + 숫자 + _ 을 제외한 모든 문자를 찾습니다.

POSIX 에서만 사용하는 문자클래스가 있는데, 단축키처럼 편리하게 사용할 수 있습니다. 대표적인 POSIX 문자클래스는 다음과 같으며 대괄호 '[' ]' 가 붙어있는 모양 자체가

표현식이므로 실제로 문자클래스로 사용할 때에는 대괄호를 씌워서 사용해야만 정상적인 결과를 얻을 수 있습니다.

정규표현식	표현	설명
[:\w:]		알파벳과 숫자를 찾습니다.
[:\alpha:]		알파벳을 찾습니다.
[:\d:]		0~9사이를 찾습니다.
[:\lower:]		알파벳 소문자를 찾습니다.
[:\upper:]		알파벳 대문자를 찾습니다.
[:\s:]		탭과 공백문자를 찾습니다.

이밖에도 [:\cntrl:] : 아스키 제어문자(0~31 번, 127 번), [:\print:] : 출력 가능한 모든 문자, [:\xdigit:] : 모든 16 진수 숫자 등이 있습니다.

정규표현식을 실제로 사용할 때 언어마다 사용방법이 각각 다릅니다. 진행했던 프로젝트에서는 정규표현식을 JavaScript 에서 사용했는데, JavaScript 에서 사용하는 방법에 대해서 설명 하겠습니다. 사용하는 JavaScript 버전이 1.1 이하 버전일 경우에는 정규표현식을 사용할 수 없습니다. 정규표현식을 사용하는 방법으로는 두 가지가 방법이 존재하며, 첫 번째로는 'RegExp'객체를 이용하는 방법이 있습니다. 주로 정규표현식이 자주 변경되는 경우 사용합니다.

```
// RegExp 객체를 이용하는 방법
var objectInitializer = new RegExp('정규표현식','[Flag]');
```

두 번째로는 객체초기화(Object Initializer)를 사용하는 방법입니다. 주로 입력된 표현식이 거의 바뀌지 않는 상수 형태의 표현식을 사용할 때 사용합니다.

```
// 객체초기화(Object initializer) 방법  
var regExp = /정규표현식/[Flag];
```

## Flag 의 종류

자주 사용하는 Flag 는 밑의 3 종류가 있으며 Flag 를 사용을 하지 않을 수도 있습니다.  
만약 Flag 를 설정 하지 않을 경우에는 문자열 내에서 검색대상이 많더라도 한번만 찾고 끝나게 됩니다.

Flag	설명
g	Global - 문자열 내의 모든패턴을 찾습니다.
i	Ignore Case -문자열의 대소문자를 구별하지 않습니다.
m	Multi Line - 문자열이 행이 바뀌어도 찾습니다.

이 외에도 공백을 무시하고 주석을 허용하는 x, 개행문자도 포함해서 찾는 s 등 다양한 Flag 들이 있습니다.

## 정규 표현식 실제 적용

사용자로부터 값을 입력 받는 부분에서 유효성 체크를 하기 위해 정규 표현식을 간단하게 적용한 경우가 있었습니다. 먼저 입력 받은 값은 반드시 한글이 포함되지 않도록 유효성 체크를 하는 부분이 있었습니다. 사용자가 입력한 데이터 중에서 유효하지 않는 데이터를 정규 표현식을 이용하여 검색한 뒤 Return 하는 방법을 사용하였습니다.

```
//사용자가 입력한 ID 가 한글이 포함되어 있는지 Check 합니다.  
function idCheck () {  
    // 입력한 ID 를 Check 하기 위해 가져옵니다.  
    var titleCheck = $("titleId").val;  
    // 정규표현식으로 한글만 선택하도록 만듭니다.  
    var languageCheck = /[ㄱ-ㅎ|ㅏ-ㅣ|가-힣]/;
```

```

// 입력한 ID 와 정규표현식을 비교하여 한글 여부를 판단합니다.
// test 외에도 search ,exec , match 등을 사용할 수 있습니다.
if (languageCheck.test(titleCheck)) {
    alert("ID 에 한글이 포함되어 있습니다.");
    return;
}

...

}

```

다음으로는 8 자리 이하 정수로 이루어진 x, y 좌표를 사용자로 부터 입력 받는 경우가 있었습니다. 사용자가 조건에 충족하지 않은 값을 입력할 경우 DB 에 적재 할 때나 좌표를 활용할 때 문제가 발생할 수 있기 때문에 유효성 체크가 필요했습니다. 사용자가 값을 입력할 때마다 유효한 값인지 체크를 하고, 잘못된 값을 입력하면 그 값을 Null 로 치환을 하는 방법을 사용했습니다. 사용자 입장에서는 유효하지 않은 값을 입력하면 값을 입력하는 순간 아무런 동작을 하지 않은 것처럼 보입니다.

```

// 8 자리 이하인 숫자인지 Check 하는 Function
// 사용자가 Key 를 입력할 때마다 Function 이 호출되도록 구현하였습니다.
function checkNumber (data) {
    // 사용자가 입력한 값을 Check 를 위해 변수에 넣습니다.
    var checkData = data.value;
    // 입력한 값이 8 자리가 넘어가는지 Check 를 합니다.
    if ( checkData.length > 8 ) {
        // 8 자리가 넘어가면 8 자리까지만 표현하고 나머지는 제외합니다.
        data.value = checkData.substring(0,8);
    } else {
        // 8 자리 이하일 경우
        // Number 형이 아닌값이 입력되면 입력값을 null 값으로 대체합니다.

```

```

data.value = checkData.replace(/^[^0-9]/g, "");

}

}

```

정규 표현식으로 조건을 구현하니 매우 간단하게 해결하였습니다. 이 밖에도 Email Check, File 확장자 Check, 주민등록번호 Check, 문자열 공백제거, 문자열 첫 글자 대문자로 치환 등등 정규 표현식을 이용하여 다양한 형태의 유효성검사를 구현할 수 있습니다. 정규 표현식을 구현하면서 유용한 Utility 들이 있습니다. 물론 이러한 Utility 들은 Web 에서 다양하게 찾아 볼 수 있지만 프로젝트를 진행하면서 유용하게 사용했던 Utility 두가지에 대해서 간단하게 소개하도록 하겠습니다. 먼저 사용자가 정규 표현식을 작성하고 직접 원하는 문자열을 Test 할 수도 있고, quality 높은 표현식을 구현하는데 도움을 주는 Utility 입니다. 정규 표현식에 대해서 지식이 부족한 사용자도 우측의 정규식 표현 Sample 과 그에 대한 설명이 자세하게 나와있어서 쉽게 구현할 수 있습니다. 프로그램을 다운받지 않고 Web 에서 직접 실행하므로 별다른 설치 없이도 즉시 사용할 수 있는 편리성이 있습니다. 하지만 Web 에서 실행하므로 Off-Line 에서는 지원이 안되며, 프로그램 내부에서 전체적으로 Font Size 가 작다는 단점이 있습니다.

<http://gskinner.com/RegExr/>

The screenshot shows the RegExr website interface. The main area is titled "정규표현식 입력" (Regex Input) and contains a text input field with the regex pattern `\d{6} \- [1-4]\d{6}`. Below the input field, there are checkboxes for `global`, `ignoreCase`, `extended`, `dotall`, and `multiline`. A "Share Link" button is also present. The text area below the input field contains the following text: "사용자가 문자열을 직접 입력해서 해당 정규표현식에 해당하는 문자열을 찾습니다. (Test)" and a list of test strings: "123456 - 2345678", "111111 - 2222222", "123 - 3444444", "860701 - 3333", and "1234561234567".

On the right side, there is a "Samples" tab with a "show all" button. Below it, a list of regex samples is shown, including `\w`, `\W`, `\d`, `\D`, `\s`, `\S`, `[ABC]`, `[^ABC]`, `[a-z]`, `[a-zA-Z]`, `[^f-m]`, `[0-9]`, `[\w-]`, `\t`, `\r`, `\n`, `\xFF`, `\\`, and `\.`. Below the list, there is a description: "Matches any character, except for line breaks if dotall is false." and a note: "각 정규표현식에 대한 설명을 확인할 수 있습니다." (You can check the explanation for each regular expression).

At the bottom left, there is a section titled "RegExr: /d{6} \- [1-4]\d{6}/g" showing the pattern and flags, and a note: "0 capturing groups: 해당 정규표현식의 정보를 확인할 수 있습니다." (You can check the information for this regular expression).

두번째 Utility 는 표현식을 쉽게 이해할 수 있도록 도식화 하는 Utility 입니다. 앞에서 정규 표현식 표현방법을 소개 할 때 쉽게 이해할 수 있도록 도식으로 처리한 부분도 이 Utility 를 이용하여 직접 구현하였습니다. 이 Utility 는 표현식을 구현하기 보다는 복잡한 표현식을 해석하고 이해하는 목적이 가장 알맞다고 생각합니다. 프로젝트를 진행하면서 직접 구현한 표현식이 도식으로 목적에 맞게 구현 되는지 Test 할 수 있습니다. 정규 표현식에 대해 어느 정도 지식을 갖추고 있는 사용자들에게 적합하다고 생각합니다. 이 Utility 도 앞선 Utility 와 마찬가지로 Web 에서 별다른 설치 없이 즉시 사용 가능합니다.

<http://www.regexper.com/>

# REGEXPER

You thought you only had two problems...

Fork me on GitHub

정규표현식을 입력하고 Display 버튼을 Click

Display

정규표현식을 구현하는 부분

One of:

digit

" - "

"1" - "4"

digit

6 times

6 times

정규표현식을 도식으로 표현한 부분

Created by [Jeff Avallone](#) | Source code licensed: 

Generated images licensed: 



## 참조 Site

정규표현식 - wiki 백과 <http://ko.wikipedia.org/wiki/정규표현식>

정규표현식의 기본 문법 정리표 <http://blog.daum.net/creazier/15309380>

정규표현식 사용하기 <http://icoon22.tistory.com/220>

정규식이란 무엇인가 <http://twinstarbox.tistory.com/entry/Java-정규식이란-무엇인가>

자바스크립트 정규 표현식 <http://yaku.tistory.com/75>

Perl 정규표현식,

메타데이터 <http://blog.naver.com/PostView.nhnblogId=turtle1006&logNo=60107758671>

## 정규표현식 관련 Utility Site

정규표현식 Test 및 생성 Util <http://gskinner.com/RegExr/>

정규표현식 도식화 표현 Util <http://www.regexper.com/>