정규식 체크 예제

모든 공백 체크 정규식
```
var regExp = /\s/g;
```

숫자만 체크 정규식
```
var regExp = /^[0-9]+$/;
```

이메일 체크 정규식
```
var regExp = /^[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*@[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*\.[a-zA-Z]{2,3}$/i;
```

핸드폰번호 정규식
```
var regExp = /^\d{3}-\d{3,4}-\d{4}$/;
```

일반 전화번호 정규식
```
var regExp = /^\d{2,3}-\d{3,4}-\d{4}$/;
```

아이디나 비밀번호 정규식
```
var regExp = /^[a-z0-9_]{4,20}$/;
```

휴대폰번호 체크 정규식

```
var regExp = /^01([0|1|6|7|8|9]?)-?([0-9]{3,4})-?([0-9]{4})$/;
```

정규표현식

1. 확장문자 (: backslash)
    - s : 공백 문자(스페이스, 탭, 폼 피드, 라인 피드)

    - b : 단어의 경계
    - B 이를 제외한 모든 문자 매칭

    - d : 숫자
    - D : 숫자가 아닌 문자 [^0-9] 와 동일

    - w : 알파벳, 숫자로 된 문자, 밑줄 기호(_) [A-Za-z0-9]
    - W : w의 반대 문자 [^A-Za-z0-9]

    - 특수문자 : 특수문자 자체를 의미 예) + (+ 기호 자체)

2. 특수문자
    - * : 0회 이상 반복
    - + : 1회 이상 반복
    - ? : 0 또는 1개의 문자 매칭
    - . : 정확히 1개 문자 매칭

3. 플래그
    - g : 전역매칭
    - i : 대소문자 무시
    - m : 여러 줄 매칭

4. 기타
    - () : 괄호로 묶인 패턴은 매칭된 다음, 그 부분을 기억한다.

- $1,...,$9 : 괄호로 갭처한 부분 문자열이 저장 됨.
- | : ~또는~
- {} : 반복 횟수

############## 간단한 정규 표현식
var re = /a/ --a 가 있는 문자열
var re = /a/i --a 가 있는 문자열, 대소문자 구분 안함
var re = /apple/ -- apple가 있는 문자열
var re = /[a-z]/ -- a~z 사이의 모든 문자
var re = /[a-zA-Z0-9]/ -- a~z, A~Z 0~9 사이의 모든 문자
var re = /[a-z]|[0-9]/ -- a~z 혹은 0~9사이의 문자
var re = /a|b|c/ -- a 혹은 b 혹은 c인 문자
var re = /[^a-z]/ -- a~z까지의 문자가 아닌 문자("^" 부정)
var re = /^[a-z]/ -- 문자의 처음이 a~z로 시작되는 문장
var re = /[a-z]$/ -- 문자가 a~z로 끝남

상기에 정의된 간단한 표현식을 아래에 넣어 직접 해 보시기 바랍니다.
var str = "sample string";
re.test(str)?"true":"false";

* 특수문자(''', ''^'', ''$'', ''*'', ''+'', ''?'', ''.'', ''('', '')'', ''|'', ''{'', ''}'', ''['', '']'')를 검색할 경우는 ''' 를 넣는다.

############## 간단한 응용예제

var re = /s$/; -- 공백체크
var re = /^ss*$/; -- 공백문자 개행문자만 입력 거절
var re = /^[-!#$%& amp;'*+./0-9=?A-Z^_a-z{|}~]+@[-!#$%&'*+/0-9=?A-Z^_a-z{|}~]+.[-!#$%& amp;'*+./0-9=?A-Z^_a-z{|}~]+$/; --이메일 체크
var re = /^[A-Za-z0-9]{4,10}$/ -- 비밀번호,아이디체크 영문,숫자만허용, 4~10자리
var re = new RegExp("(http|https|ftp|telnet|news|irc)://([-/.a-zA-Z0-9_~#%$?&=:200-377()]+)","gi") -- 홈페이지 체크

```
var re = "<[^<|>]*>";  -- 태그제거
var re = /[<][^>]*[>]/gi;-- 태그제거
str = str.replace(RegExpTag,"");

var RegExpJS = "<script[^>]*>(.*?)</script>";  -- 스크립트 제거
str = str.replace(RegExpJS,"");

var RegExpCSS = "<style[^>]*>(.*?)";  -- 스타일 제거
str = str.replace(RegExpCSS,"");

var RegExpHG = "(/[ㄱ-ㅎ|ㅏ-ㅣ|가-힣]/)";  -- 한글 제거
str = str.replace(RegExpHG,"");

var RegExpDS = /<!--[^>](.*?)-->/g;    -- 주석 제거
str6 = str.replace(RegExpDS,"");

var regExp = /[a-z0-9]{2,}@[a-z0-9-]{2,}.[a-z0-9]{2,}/i; --이메일 체크


## 기타 응용
re = new RegExp("^@[a-zA-Z0-9]+s+","i");//문장의 처음이 @이고 문자가 1나 이상 있으면 ok



기타 상기와 동일하나 약간씩 다른 샘픔
영숫자 조합책크
if ((new RegExp(/[^a-z|^0-9]/gi)).test(frm.loginid.value)) {
      alert("ID는 영숫자 조합만 사용하세요");
      frm.loginid.focus();
}

홈페이지 주소 책크
 function chk(v){
```

```
  str='';
  re = new RegExp("^http://","i");
  re.test(v)?str='y':str='n';
  alert(str);
 }
```

hanmail인지를 책크

```
 function chk(v){
  str='';
  re = new RegExp("hanmail.net","i");
  re.test(v)?str=true:str=false;
  return str
 }
```

//본문내에서 도메인 구하기

```
var patt = /(http(s)?://)?w+(.w+)+/gi;
       var result = (aa.value.match(patt));
```

//본문내에서 url구하기

상기와 유사 var patt = /(http(s)?://)?w+(.w+).S*-gi;

########## 정규식 메소드 및 사용법

참조 http://eknote.tistory.com/1251

참조 http://www.javascriptkit.com/javatutors/redev3.shtml

RegExp.exec(string)

RegExp.test(string)

String.match(pattern)

String.search(pattern)

String.replace(pattern,string)

String.split(pattern)