B 정규 표현식

모던 웹을 위한 Javascript jQuery 입문

한빛미디어 Hanbit Media, Inc.

B.1 정규 표현식 객체

❖ 정규 표현식

- 자바스크립트의 기본 내장 객체
- 문자열 패턴을 검사할 때 사용
- 원래는 펄Perl 프로그래밍 언어에서 사용 시작한 객체
- 정규 표현식 객체는 코드 B-1 처럼 두 가지 방법으로 생성 가능

코드 B-1 정규 표현식 객체의 생성

```
<script>
    var regExp1 = new RegExp('text');
    var regExp2 = /text/;
</script>
```

표 B-1 정규 표현식 객체의 메서드

메서드 이름	설명
test()	정규 표현식과 일치하는 문자열이 있으면 true를 아니면, false를 리턴합니다.
exec()	정규 표현식과 일치하는 문자열을 리턴합니다.

B.1 정규 표현식 객체

❖ 정규 표현식을 사용한 예제 (1)

- 정규 표현식은 문자열처럼 만들어 사용
- 출력 결과
  - 변수 string 내부에 정규 표현식과 일치하는 문자열 'script' 있음
    - **true 출력**

코드 B-2 정규 표현식의 메서드

```
<script>
    // 변수를 선언합니다.
    var regExp = /script/;
    var string = 'Javascript jQuery Ajax';

    // 메서드를 사용합니다.
    var output = regExp.test(string);

    // 출력합니다.
    alert(output);
</script>
```

❖ 정규 표현식을 사용한 예제 (2)

- 표 B-2의 문자열 객체 메서드와 함께 사용하는 것이 일반적

표 B-2 문자열 객체의 정규 표현식 사용 메서드

메서드 이름	설명
match(regExp)	정규 표현식과 일치하는 부분을 리턴합니다.
replace(regExp, replacement)	정규 표현식과 일치하는 부분을 새로운 문자열로 바꿉니다.
search(regExp)	정규 표현식과 일치하는 부분의 위치를 리턴합니다.
split(regExp)	정규 표현식을 기준으로 문자열을 잘라 배열을 리턴합니다.

- split() 메서드를 사용한 예제
  - 정규 표현식을 기준으로 문자열 잘라 배열로 만들어 리턴

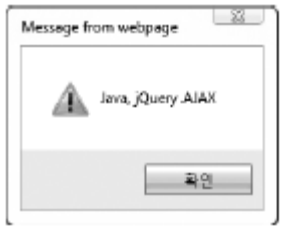
코드 B-3 정규 표현식을 사용하는 String 객체의 메서드

```
<script>
    // 변수를 선언합니다.
    var regExp = /script/;
    var string = 'Javascript jQuery Ajax';

    // 메서드를 사용합니다.
    var output = string.split(regExp);

    // 출력합니다.
    alert(output);
</script>
```

그림 B-1 실행 결과



B.2 대체 문자

❖ replace() 메서드

- 정규 표현식 사용하면 표 B-3의 대체 문자 사용 가능

표 B-3 대체 문자

정규 표현식 기호	설명
$&	일치하는 문자열
$`	일치하는 부분의 앞부분 문자열
$'	일치하는 부분의 뒷부분 문자열
$1, $2, $3	그룹

참고 `는 숫자 1키 왼쪽에 있는 기호이고 '는 엔터키 왼쪽에 있는 기호입니다.

B.2 대체 문자

❖ replace() 메서드 테스트 예제

- 문자열 객체의 replace() 메서드 사용
  - 정규 표현식에 일치하는 문자열을 '+$&+'로 변경

코드 B-4 대체 문자의 사용 (1)

```
<script>
    // 변수를 선언합니다.
    var regExp = /a/;
    var string = 'Javascript jQuery Ajax';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');

    // 출력합니다.
    alert(output);
</script>
```

그림 B-2 실행 결과

B.2 대체 문자

❖ replace() 메서드 테스트 예제 (2)

- replace() 메서드의 매개 변수에 함수를 넣음

코드 B-5 replace() 메서드의 함수 매개 변수

```
// 메서드를 사용합니다.
var output = string.replace(regExp, function (value) {
    return '+' + value + '+';
});
```

그림 B-2 실행 결과

❖ replace() 메서드 테스트 예제 (3)

- 정규 표현식을 코드 B-6처럼 괄호를 사용해서 묶으면 각각의 문자는 $1, $2, $3 문자로 대체
  - 각각의 문자가 그림 B-3처럼 a, v, a를 출력
  - 정규 표현식을 사용하면 가장 먼저 패턴이 일치하는 문자열만 찾음
  - 모든 문자를 대체하려면 플래그 문자 사용

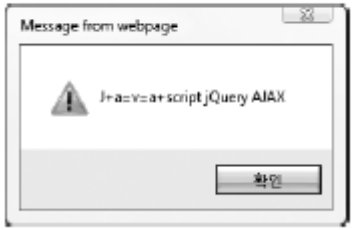
코드 B-6 대체 문자의 사용 (2)

```
<script>
    // 변수를 선언합니다.
    var regExp = /(a)(v)(a)/;
    var string = 'Javascript jQuery Ajax';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$1=$2=$3+');

    // 출력합니다.
    alert(output);
</script>
```

그림 B-3 실행 결과



B.3 플래그 문자

❖ 플래그 문자

- 간단한 방법으로 생성할 때에는 뒤에 붙여 사용
- 생성자를 사용할 때에는 두 번째 매개 변수에 입력
- 플래그 문자 위치에 들어가는 플래그 문자의 순서는 상관 없음

표 B-4 플래그 문자

정규 표현식 기호	설명
g	전역 비교를 수행합니다.
i	대소문자를 가리지 않고 비교합니다.
m	여러 줄의 검사를 수행합니다.

```
var regExp = /Expression/im
var regExp = new regExp('Expression', 'im');
```

B.3 플래그 문자

❖ 플래그 문자 테스트 예제 (1)

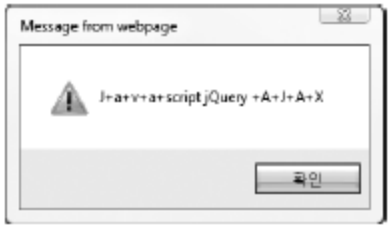
- 첫 번째 패턴 일치 부분뿐만 아니라 대소문자와 전역 비교 가능
  - 코드 B-7처럼 플래그 문자 i와 g 사용
  - i와 g의 순서는 변경돼도 상관 없음

코드 B-7 플래그 문자

```
<script>
    // 변수를 선언합니다.
    var regExp = /a/ig;
    var string = 'Javascript jQuery Ajax';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');
    // 출력합니다.
    alert(output);
</script>
```

그림 B-4 실행 결과



B.4 앵커 문자

❖ 앵커 문자란?

- 문자열의 앞과 뒤를 구분해주는 정규 표현식 기호
- 표 B-5의 앵커 문자 많이 사용

표 B-5 앵커 문자

정규 표현식 기호	설명
^ABC	맨 앞 문자가 ABC
ABC$	맨 뒤 문자가 ABC

B.4 앵커 문자

❖ 앵커 문자 테스트 코드 (1)

- j로 시작하는 부분을 찾으므로 문자열 Javascript의 첫 번째 문자 J 찾음

코드 B-8 앵커 문자

```
<script>
    // 변수를 선언합니다.
    var regExp = /^j/ig;
    var string = 'Javascript\njQuery\nAjax';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');

    // 출력합니다.
    alert(output);
</script>
```

그림 B-5 실행 결과

❖ 앵커 문자 테스트 코드 (2)

- 플래그 문자 m
  - 문자열이 여러 줄을 형성할 때, 각각의 줄을 개별적인 문자열로 인지
  - 여러 줄에 걸친 대소 문자를 무시한 전역 검사 실시
    - **두 번째 줄의 jQuery의 j도 시작하는 문자로 판단**

코드 B-9 앵커 문자와 m 플래그 문자

```
// 변수를 선언합니다.
var regExp = /^j/igm;
var string = 'Javascript\njQuery\nAjax';
```

그림 B-6 실행 결과

B.5 메타 문자

❖ 메타 문자란?

- 자바스크립트의 정규 표현식 객체가 갖는 가장 유용한 기능
  - [가-히] 형태로 사용해 한글도 구분 가능
  - [h-k]의 형태로 특정 알파벳 범위를 한정할 수도 있음

표 B-6 메타 문자 (1)

기호	설명
.	아무 글자
[abc]	괄호 안의 글자
[^abc]	괄호 안의 글자 제외
[a-z]	알파벳 a부터 z까지
[A-Z]	알파벳 A부터 Z까지
[0-9]	숫자 0부터 9까지

B.5 메타 문자

❖ 메타 문자 테스트 코드 (1)

- [aj] 메타 문자와 플래그 함께 사용
- a 또는 j, A 또는 J를 검사
  - 코드를 실행하면 각각의 a, j, A, J 문자가 대체

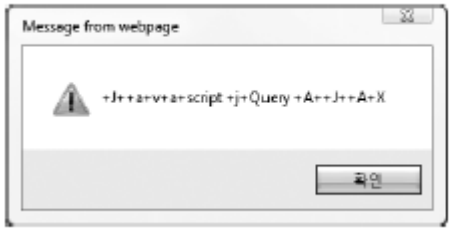
코드 B-10 메타 문자

```
<script>
    // 변수를 선언합니다.
    var regExp = /[aj]/ig;
    var string = 'Javascript jQuery Ajax';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');

    // 출력합니다.
    alert(output);
</script>
```

그림 B-7 실행 결과



❖ 메타 문자 테스트 코드 (2)

- 예제 코드 B-11
  - 주민등록번호를 검사하는 정규 표현식
  - 앞에 여섯 개의 글자가 오고 중간에 '-'
  - 뒤에 일곱 개의 글자가 나오면 OK
  - 실행 결과
    - **주민등록번호 형식을 넣으면 양 끝에 +를 추가해 출력**

코드 B-11 메타 문자를 사용한 주민등록번호 확인(1)

```
<script>
    // 변수를 선언합니다.
    var regExp = /......-...../;
    var string = '910209-2001211';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');

    // 출력합니다.
    alert(output);
</script>
```

❖ 메타 문자 테스트 코드 (3)

- 사용자가 주민등록번호 대신 문자를 입력하면?
  - 표 B-7의 메타 문자 참고

표 B-7 메타 문자 (2)

기호	설명
₩d	숫자
₩w	아무 단어(숫자 포함)
₩s	공백 문자(탭, 띄어쓰기, 줄바꿈)
₩D	숫자 아님
₩W	아무 단어 아님
₩S	공백 문자 아님

코드 B-12 메타 문자를 사용한 주민등록번호 확인 (2)

```
<script>
    // 변수를 선언합니다.
    var regExp = /\d\d\d\d\d\d-[1234]\d\d\d\d\d\d/;
    var string = '910209-2001211';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');

    // 출력합니다.
    alert(output);
</script>
```

B.6 수량 문자

❖ 수량 문자를 이용한 예제코드 (1)

표 B-8 수량 문자

정규 표현식 기호	설명
a+	a가 적어도 1개 이상
a*	a가 0개 또는 여러 개
a?	a가 0개 또는 1개
a{5}	a가 5개
a{2,5}	a가 2개~5개
a{2,}	a가 2개 이상
a{,2}	a가 2개 이하

코드 B-13 수량 문자를 사용한 주민등록번호 확인

```
<script>
    // 변수를 선언합니다.
    var regExp = /\d{6}-[1234]\d{6}/;
    var string = '910209-2001211';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');

    // 출력합니다.
    alert(output);
</script>
```

B.6 수량 문자

❖ 수량 문자를 이용한 예제코드 (2)

- 괄호를 수량 문자와 함께 사용
  - 특정 문자열의 반복을 찾아낼 수 있음
- 예제 코드 B-14]
  - 문자열 na가 한 번 이상 반복되는 패턴을 찾는 정규 표현식 사용

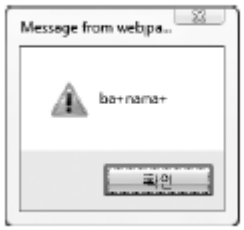
코드 B-14 수량 문자

```
<script>
    // 변수를 선언합니다.
    var regExp = /(na)+/;
    var string = 'banana';

    // 메서드를 사용합니다.
    var output = string.replace(regExp, '+$&+');

    // 출력합니다.
    alert(output);
</script>
```

그림 B-8 실행 결과



B.7 선택 문자

❖ 선택 문자란?

- '또는'의 역할을 수행하는 정규 표현식 기호
- 모든 정규 표현식 기호와 함께 활용 가능

❖ 예제 코드 B-15]

- 소문자 또는 숫자로 구성된 단어인지 확인하는 코드

표 B-9 선택 문자

정규 표현식 기호	설명
(abc|def)	abc 또는 def를 선택합니다.

코드 B-15 선택 문자

```
<script>
    // 변수를 선언합니다.
    var string = prompt('소문자 또는 숫자로만 구성된 단어를 입력하세요.', '단어');
    var regExp = /([0-9]|[a-z])/g;

    // 출력합니다.
    if (string.replace(regExp, '').length == 0) {
        alert('감사합니다.');
    } else {
        alert('ㅠㅜ');
    }
</script>
```

B.8 정규 표현식 사용 예제

❖ 한글 이름 확인하는 함수

- 정규 표현식을 사용해 한글을 모두 빈 문자열로 대체
- 모든 한글이 빈 문자열로 바뀌었다면?
  - 문자열 replacedString의 length 속성은 0

코드 B-16 한글 이름 확인

```
<script>
    function isKoreanName(string) {
        // 변수를 선언합니다.
        var regExp = /([가-힣])/g;

        // 메서드를 사용합니다.
        var replacedString = string.replace(regExp, '');

        // 확인합니다.
        if (replacedString.length == 0) {
            return true;
        } else {
            return false;
        }
    }

    alert(isKoreanName('윤인성'));
</script>
```

B.8 정규 표현식 사용 예제

❖ 이메일 구분하는 방법

그림 B-9 이메일의 패턴

`글자` @ `글자` . `글자`

코드 B-17 이메일

```
<script>
    function isEmail(string) {
        // 변수를 선언합니다.
        var regExp = /\w+@\w+\.\w+/;

        // 확인합니다.
        return regExp.test(string);
    }

    alert(isEmail('rintiantta@naver.com'));
</script>
```

B.8 정규 표현식 사용 예제

❖ 숫자의 천 단위마다 점을 찍는 예제

- 예제 코드 B-18의 addComma() 함수 사용

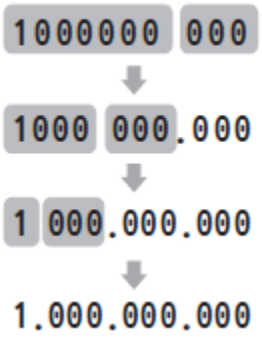
코드 B-18 숫자 천 단위 점 찍기

```
<script>
    function addComma(number) {
        var regExp = /(^[+-]?\d+)(\d{3})/;
        var string = String(number);

        while (regExp.test(string)) {
            string = string.replace(regExp, '$1,$2');
        }
        return string;
    }

    alert(addComma('1000000000'));
</script>
```

그림 B-10 천 단위 콤마 함수 작동 순서



Thank You !

모던 웹을 위한 Javascript jQuery 입문

한빛미디어 Hanbit Media, Inc.