**파이썬**

**String 메서드**

* isalpha() : 객체가 문자열이면 True 리턴
* isdigit() : 객체가 숫자면 True 리턴

#

* capitalize() :  텍스트 맨 첫 글자를 대문자로 바꾸어준다.(대문자면 변화 x) 출력값 type 은 str
* upper() : 문자열 중 소문자를 모두 대문자로 바꾼다.(대문자면 변화 x) 출력값 str
* lower() : 문자열 중 대문자를 모두 소문자로 바꾼다.(소문자면 변화 x) 출력값 stㄱ
* swapcase() : 대문자면 소문자로 소문자면 대문자로 바꾼다. 출력값 str
* isupper() : 문자열이 모두 대문자이면 True를 출력한다. 출력값  bool
* islower() : 문자열이 모두 소문자이면 True를 출력한다. 출력값 bool
* count("찾을문자") : 찾을 문자(굳이 한글자가 아니어도 된다.)의 개수를 리턴한다. 출력값은 int
* startswith("내용") : 해당 문자열이 "내용"으로 입력한 것으로 시작하는지 확인하는 함수 출력값은 bool(True, False)
* endswith("내용") : 해당 문자열이 "내용"으로 입력한 것으로 끝나는지 확인하는 함수 출력값은  bool(True , False)
* find("내용", 시작번호, 끝번호) :  문자열중 "내용"이 처음 나온 위치(파이썬은 처음이 0부터 시작)를 반환한다.

시작번호 ,끝번호는 생략할 수있다. 시작번호 생략하면 0으로 시작,  시작번호를 쓰고 끝번호 생략하면 맨 끝까지 찾는다.

만약 문자열중 "내용"이 존재 않는다면 -1을 반환한다.

출력값은 int

* index('내용',시작번호,끝번호) : find와 마찬가지로 문자열중 "내용"이 처음 나온 위치(파이썬은 처음이 0부터 시작)를 반환한다.

시작번호 ,끝번호는 생략할 수있다. 시작번호 생략하면 0으로 시작,  시작번호를 쓰고 끝번호 생략하면 맨 끝까지 찾는다.

find와 다른 점은 문자열 중 "내용"값이 존재하지 않으면 오류(ValueError)가 난다.

출력값은 int

* lstrip() : 문자열 중 가장 왼쪽에 있는 한 칸 이상의 공백을 모두 지운다. 출력값 str
* rstrip() : 문자열 중 가장 오른쪽에 있는 한 칸 이상의 공백을 모두 지운다. 출력값 str
* strip() : 문자열 중 양 쪽에 있는 한 칸 이상의 공백을 모두 지운다. 출력값 str

* join("내용") : "내용"안의 각각의 문자 사이에 문자열을 집어넣는다. 헷갈릴 수도 있는데 함수 앞에 나오는 문자가 삽입되는 문자열이다. 출력값은 str
* replace("기존의 것","새로운 것") : 기존에 있던 문자열을 새로운 문자열로 바꾸는 함수이다. 출력값 str
* split("내용") : "내용"값을 기준으로 문자열을 나누어 준다. 만약 "내용"값이 없으면 공백을 기준으로 나눈다.  나눈 항목들은 리스트의 요소로 저장이 된다. 출력값  list

**list 매서드**

* 수정

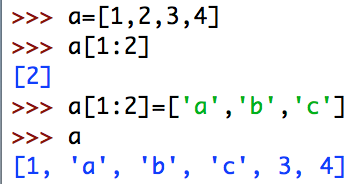
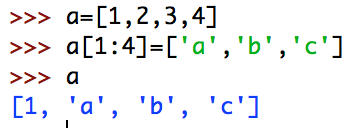
1. 리스트에서 하나의 값 수정하기

L[i]=x

리스트의 i번 째 항목이 x로 수정된다.

주의 : x=L[i]로 입력하면 변수 x에 리스트의 i번째 항목(객체)가 할당된다.

2. 연속된 범위의 값도 수정이 가능하다.

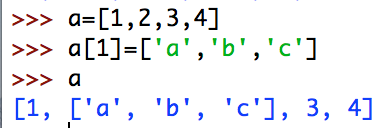
  

이렇게 [1:2] 라고 1부터 2전까지 라고 범위를 지정해두면 1은 포함 2는 포함하지않으므로 a[1]이라는 값을 슬라이싱해서 보여주지만

a[1:2]는  연속된 값으로 처리되므로 문자열 'a','b','c'가 2 대신에 들어 간 것을 확인 할 수 있고 (a[1]과 a[2]사이의 리스트를 ['a','b','c']로 수정)

오른쪽 예에서도 볼 수있듯이 a[1:4]는 1부터 3까지 범위 즉 2,3,4가 'a','b','c'로 바뀐 것을 확인 할 수 있다.(a[1]과 a[3](포함)사이의 리스트를 ['a','b','c']로 수정)

주의 :



이처럼 한 요소만 저렇게 리스트로 바꾸면 리스트 안의 a[1]의 요소가 리스트 자체가 되버린다.

a[1]의 요소를 ['a','b','c']로 수정한다는 뜻이므로 전혀 다른 결과 값이 나오는 것을 확인 할 수있다.

* 삭제

1.[]을 이용해 삭제하기

a[1:3]을 빈 공백으로 수정하라는 뜻인데 a[1], a[2]를 빈 공백으로 수정하니까 삭제 된 것을 확인 할 수있다.

하나의 값, 연속된 범위 둘 다 적용된다.

2. del 함수를 이용하여 삭제하기

del함수는 파이썬이 자체적으로 갖고있는 내장함수이다. 객체를 삭제하는 함수인데

위의 예에서는 a[1:3]을 삭제하라고 했으니까 a[1], a[2]가 삭제 된 것을 확인 할 수있다.

하나의 값, 연속된 범위 둘 다 적용된다.

리스트의 연산

len(L)(리스트 요소의 개수)

max(L)(리스트 중 최대값)

 min(L)(리스트 중 최소값)

sum(L)(리스트 요소의 합)

 sorted(L)(작은 것부터 큰 것 순으로 정렬)

이 5개가 가능하다. ( 함수를 쓸 때 입력값은 리스트이다.)

리스트 연산자

리스트는 +와 \*으로도 연산이 가능하다

+ : 두개의 리스트를 하나의 리스트로 합친다

\* : 한개의 리스트를 곱한 만 큼 반복한다.

Aliasing vs Copying

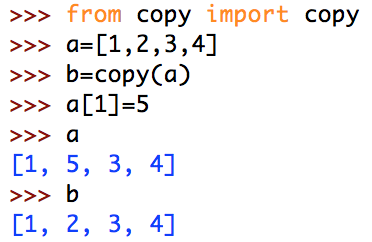
Aliasing : 두개의 변수가 하나의 리스트(객체)를 가르켜 한 변수의 리스트가 수정되면 다른 변수의 리스트도 수정이 된다.

Copying : 기존의 리스트에서 새로운 리스트를 만들어 그 리스트가 기존의 리스트 요소(객체)를 가르켜  리스트가 수정되어도 새로운 리스트는 수정되지 않는다.

copying 하는 방법은 두가지 이다.

1.[:] (리스트 전체)를 이용하는 방법

2.copy 모듈을 이용하는 방법



변수가 같은 객체를 가리키고 있는지 확인 하는 방법으로는 파이썬 내장 함수로 있는 is 를 쓰면 된다.

a is b 를 했을 때 True면 같은 객체를 가리키고 있는 것이고 False면 다른 객체를 가리키고 있는 것이다.

* 추가

* append(x) : 리스트의 맨 마지막 요소에 x 를 추가한다. (x는 리스트에 들어갈 수있는 어떠한 값이어도 상관없다.)(append함수는 요소 하나만 추가하는 함수)

a에 리스트 요소가 추가 되고 그냥 정수형 도 추가 된 것을 확인 할 수있다.(추가되는 위치는 항상 맨 마지막에 추가된다)

* insert(x,y) : 리스트의 x번째 위치에 y를 추가하는 함수 (append와 마찬가지로 요소 하나를 추가하는 것이지만 다른 점은 추가할 위치를 지정할 수 있다는 것이다.)
* extend(x) : 리스트의 맨 마지막에 리스트 x를 추가 하는 함수(a=a+[...] 와 같은 함수 이다.) extend로 리스트 요소를 추가하고 싶으면 [[...]]로 두번 씌우면 된다.
* 삭제

* clear() : 리스트에 있는 모든 요소를 삭제하는 함수

* remove(x) : 리스트에서 첫 번째로 나오는 x를 삭제하는 함수

* pop(x)  : x(인덱싱넘버)는 생략가능하고 생략하면 리스트의 맨 마지막 요소를 출력하고 그 요소는 삭제한다. x에 값을 넣으면 x번째 요소를 출력하고 그 요소는 삭제한다.

* 정렬

* sort() : 숫자가 작은 순부터 큰 순으로 , 알파벳 순서대로 리스트의 요소를 정렬한다.

* reverse() : 현재 리스트의 요소를 역으로 정렬한다. (sort한 상태에서 역정렬이 되는 것이 아니라 기존의 정렬이 역정렬이 되는 것임)

* sort(reverse=True) : sort와는 역방향으로  정렬한다.(sort()를 하고 reverse()를 쓴 상태와 같다)

* 그 외

* count(x) : 리스트의 x의 개수를 리턴한다 출력값 int

* index(x,시작번호,끝번호) : str매서드의 index와 똑같은 기능을 한다.

       리스트 요소 중 x가 처음 나온 위치(파이썬은 처음이 0부터 시작)를 반환한다.

       시작번호 ,끝번호는 생략할 수있다. 시작번호 생략하면 0으로 시작,  시작번호를 쓰고 끝번호 생략하면 맨 끝까지 찾는다.

      리스트 요소 중 x값이 존재하지 않으면 오류(ValueError)가 난다.

      출력값은 int

**set ( 순서 상관 x, 중복요소 없음)**

**set method**

* s.add(value) : set에 value값 추가한다.
* s1.update(iterable) : dict의 update, list의 extend와 같다 여러개의 값을 추가할 때 사용
* s.remove(value) : set에서 value값을 지운다
* s.clear() : set에 있는 모든요소를 제거한다.
* s1.union(s2) : s1과 s2의 합집합 (Operators  : s1|s2)
* s1.symmetric\_difference(s2) : s1과 s2의 교집합 을 제외한 합집합 (Operators  : s1^s2)
* s1.differecne(s2) : s1-s2 (s1의 차집합) (Operators  : s1-s2)
* s1.intersection(s2):  s1,s2교집합 (Operators  : s1&s2)
* s1.issubset(s2) : s1이 s2의 부분집합이냐 라고 물어보는 형식 (리턴값 boolean True or False) (s1<=s2)
* s1.issuperset(s2) : s2가 s1의 부분집합이냐라고 물어보는 형식 (리턴값 boolean) (s1>=s2)

빈 set은 {}이 아니다 그냥  {}을 쓰면 빈 딕셔너리를 만든것이라고 한다.

빈 set은 set()으로 인자를 넣지 않고 typecast함수를 쓰자.

**Tuple(순서는 있지만 값은 변할 수 없다.)**

* 하나의 요소만 갖는 튜플은 (a,)처럼 뒤에 ,를 붙여줘야한다.
* 튜플의 요소는 변할수 없다 근데 튜플의 요소가 리스트인경우 리스트의 값은 변할 수 있다. (즉 리스트인것은 변하지 않지만 리스트안의 요소는 변하게 할 수 있다.)
* 튜플은 한번에 여러가지 변수를 assign 할 때 쓰인다.

빈 튜플은 ()이라고 표현한다.

**Dictionary (map, (순서는 없고, key는 변하지 않고 value는 변할 수 있다.), key에 따라 value를 꺼내오는 형식)**

* key값은 딕셔너리 내 중복이 없어야하고 , 변하지 않는 값이어야 한다.
* Value값은 중복되어도 상관 없고, 리스트형식이어도 상관없다(즉 value값은 하나지만 하나안에 여러가지 데이터를 담는 리스트가 가능하다.)

**dict method**

* D[key]=value : key:value 값이 들어간다.
* del D[key] :  key값에 해당하는 value와 함께  key:value 쌍이 지워진다.
* D.clear() : 딕셔너리 안에 있는 모든 항목이 지워진다
* D.get(key) : key값에 해당하는 value값을 리턴한다.
* D.get(key,value) : 왜 사용하는지는 모르겠는데 기존에 key값이 있으면 기존 value값을 돌려주고 없으면 새로입력한 value를 리턴한다. 대신 새로운 key값을 넣어도

      기존 딕셔너리에 추가되지는 않는다.

* D.setdefault(key) : D.get(key)와 같다 key값에 해당하는 value값을 리턴한다.
* D.setdefault(key,value) : get 과는 다른데 기존에 key값이 있으면 기존 value값을 돌려주고 없으면 새로입력한 value를 리턴한다. 대신 새로운 key값을 넣으면 기존 딕셔너리에 추가된다.
* D1.update(D2) : D1에 D2에 있는 key: value를 모두 추가시킨다.( list 의 extend)
* D.pop(key) : key값을 안쓰면 에러가 나고(순서가 없으니 마지막이라는 것도 없다.) key값에 해당하는 value를 리턴하고 기존 딕셔너리에서 제거한다.
* D.pop(key,value) : 기존에 key값이 있으면 기존 value값을 돌려주고 제거한다. 없으면 위의 경우와 달리 에러가 나지 않고 입력한 value값을 리턴한다.
* D.keys() : 딕셔너리에 있는 key값들을 dict\_keys 리스트형태인 객체로  리턴한다.
* D.values() : 딕셔너리에 있는 value값들을 dict\_values 리스트형태인 객체로 리턴한다.
* D.items() : (key,value) 튜플 형태로 모든 key,value를 dict\_items 리스트형태인 객체로 리턴한다.[(key,value),(,)....]

~객체로 리턴한다고 하면 형태는 리스트지만 리스트로 타입을 변경하지않으면 리스트의 매서드를 적용할 수 없다.

빈 딕셔너리는 {}이다.

from collections import defaultdict를 이용해 defaultdict함수를 이용해

a=defaultdict(int) 를 하면 a의 딕셔너리는 key값이 없으면 자동으로 int의 초기값인 0을 배정해준다.

int대신 list도 가능한 것 같다.

in operator(반복가능한 자료형에 찾는 값이 있는지 알려주는 연산자(리턴값 boolean))

* set, tuple가능
* dict에서는 key값만 가능 value값은 안된다.

**모듈**

* math

영어 뜻 처럼 수학 기능을 갖고 있는 모듈이다 . 거의 공학용 계산기 인데 계산기가 없다면 잘 활용하면 계산기보다 쓰기편할 것 같다.

import math를 하고 help(math)를 하면 math모듈이 갖고있는 함수 종류를 볼 수있다. 정말 다양한 함수가 있으니 보고 잘 쓰면 좋을 것 같다.

많이 쓰이는게

sqrt(x) : x의 제곱근을 리턴한다

pi : 변수다. 파이 값을 갖고있다. 자료형은 float

sin(x) : 사인(x) 리턴한다

cos(x) :  코사인(x) 리턴한다

tan(x) :  .....

흔히 우리가 알고 있는 공학용 계산기와 비슷하다 .

* random

random  모듈은 난수(랜덤값)을 발생시키는 모듈이다

random() : [0,1) 사이에 소수점이 있는 난수를 리턴한다. -> 출력값은 float, 범위에 0은 포함,1은 포함하지 않는다.

uniform(start,stop) : [start,stop) 사이에 소수점이 있는 난수를 리턴한다. -> 출력값은 float , 범위에 start는 포함 stop는 포함하지 않는다.

randrange(start, stop, step) : [start, stop)  사이에 step값으로 건너뛰는 정수인 난수를 리턴한다.

-> 출력값은 int

     범위는 start는 포함 stop 은 포함하지 않는다

     start , step값은 입력을 안해도 된다

     start 값을 입력 안하면 [0,stop)으로 계산된다.

     step값을 입력하면 start, start+step, start+2\*step, start+3\*step, .... , stop-step 값 중에서 출력이 된다. 슬라이싱에 step 이랑 개념이 같다.

randint(start,stop) : [start, stop] 사이에서 정수인 난수를 리턴한다. -> 출력값은 int 범위는 start을 포함, stop도 포함한다.

choice(s) : 나열형 자료형(s)(list, tuple, str, ....) 의 요소 중 무작위로 하나를 선택하여 리턴한다. ->나열형 자료형이라고하는데 그냥 반복가능한 자료형에서 딕셔너리만 안 되는 것 같다.

shuffle(l) : list(l)의 요소들을 무작위로 섞은 후 섞인 리스트를 리턴한다. -> list만 되는 것 같다.

**클래스**

isinstance(변수명, 클래스명) - 객체가 클래스 의 인스턴스인지 확인 맞으면 True 리턴

\_\_init\_\_(self, …) 생성자 : 인스턴스 생성될때마다 실행

\_\_str\_\_(self, …) : 객체를 읽을 때 실행

\_\_ eq\_\_(self, …) : 객체 == 객체 를 했을 때 실행

매서드 부를때는 클래스이름.매서드이름()으로 호출

변수 부를때는 클래스이름.변수명 으로 호출

**HTML 태그 정리**

* <태그 이름 속성="값"></태그>

(모든 태그가 닫히는 태그가 있어야 하는 것은 아니다)

block 요소

* + 줄 바꿈 속성 을 갖고 있음
  + 기본 width 가 100%
  + 블록 요소를 포함할 수 없지만 예외도 있다.(예외 , div , li, dd, address(p태그 만 가능) , ...등 -> 이 요소들은 블록 요소도 포함할 수 있다. )
  + inline은 포함할 수 있다.

<h1></h1>~<h6></h6>

* + 제목 태그
  + 숫자가 작을 수록 글자의 크기가 크다

<p></p>

* + paragraph를 만드는 태그

<ul></ul>

* + <li>의 부모 태그
  + 비순차 목록을 만드는 태그 -> 옆에 동그라미 같은 걸 만들어줌

<ol></ol>

* + <li>의 부모 태그
  + 순차 목록을 만드는 태그 -> 옆에 1. 2. 3. ... 같은 걸 만들어줌

<li></li>

* + 목록을 만드는 태그

<div></div>

* + 범용 그룹화 요소
  + 기능이 없지만 기능이 없어서 정말 많이 쓰인다.

<form></form>

* + 입력값을 서버로 전송해주는 태그
  + 속성이 3가지가 있다.

속성 :

name = "내용" : 폼을 구분짓기위한 폼의 고유한 이름을 부여하는 속성

method = "내용" : 보낸 내용을 어떻게 보낼 것인지 전송 방식을 정해주는 속성( get , post, ...)

action ="주소" : 입력 값을 보낼 페이지를 정해주는 속성

Inline 요소

* + 줄 바꿈 속성이 없다
  + 내용만큼 width를 가짐
  + width, height의 속성을 사용할 수 없음
  + 블록요소를 포함할 수 없고 인라인 요소만 포함 가능

<br>

* + 줄 바꿈 태그
  + br태그 다음부터 줄이 바뀐다.

<b></b>, <em></em>, <strong></strong>

* + 강조 태그
  + 태그 안 내용이 진하게 표시된다.

<a></a>

* + 링크 태그, 링크를 걸어주는 태그이다.
  + 속성이 3가지가 있다.

href="주소" : 어디에 링크를 걸건지 주소를 입력하는 속성

target="속성" : 해당 주소를 어떻게 열 것인지 정하는 속성(예, \_blank -> 링크를 눌렀을 때 새로운 페이지에서 열린다.)

title="내용" : 태그 내용에 마우스를 올렸을 때 나오는 툴팁의 내용을 정하는 속성

<img>

* + 사진을 넣을 때 쓰는 태그
  + src 속성을 이용해 src="주소, 파일경로/이미지파일 이름' 형식으로 사진을 집어넣는다.
  + alt 속성은 사진을 로드할 수 없을 때 나오는 내용을 정해주는 속성이다.

<span>

* + 범용 인라인 요소 그룹화 태그
  + 아무 기능이 없지만 인라인 요소이고 아무 기능이 없어서 정말 많이 쓰이는 태그

부모태그 <table></table>

<table> : 표를 생성하는 태그

속성 :

width="가로 길이" : 표의 가로의 길이를 정하는 속성

border="선의 두께" : 선 두께를 정하는 속성

cellspacing = "셀 간격" : 셀과 셀의 간격을 정하는 속성

자식태그<tr></tr>

<tr> : 행을 생성하는 태그

<td></td>

<tr>로 생성된 행의 열을 생성한다

속성

rowspan="행 개수" : 지정한 행 개수 만큼 행 합치기

colspan="열 개수" : 지정한 열 개수 만큼 열 합치기

css 를 적용하는 방법은 3가지가 있다.

Inline : 태그 안에 style="내용;"으로 사용하는 방법 style 안에 내용은 css의 문법을 사용해야 한다.

Internal : head태그 안에 <style></style>태그를 넣고 스타일 태그안에 css를 작성하는 방법

External : <link href="css파일 주소">를 이용해 외부의 css파일을 불러와 적용시키는 방법

HTML (HyperTextMarkupLanguage) :

웹 상에서 정보를 입력하고 보여주는 문서를 작성할 때 사용하는 언어

HTTP(HyperTextTransferProtocol): 프로토콜 - 서버와 브라우저간의 통신적 약속?개념

브라우저와 서버의 요청(request)/응답(response) 과정을 약속한 양식과 규칙의 체계

주로 html문서를 주고받는데 사용한다. 현재 1.1 버젼까지 나왔다

80번 포트를 기본으로 사용한다.

HTTP를 통해 전달되는 자료는 http:로 시작하는 URL(Uniform Resource Locator)(인터넷 주소)(네트워크 상 리소스가 어디있는지 알려주기 위한 약속(일종의 집주소라고 생각하면 된다.)로 조회할 수 있다.)

인터넷 작동원리

브라우저(사용자) 파일이나 정보 요청(url) -> http 프로토콜로 요청 - > 서버 인식 후 해당 파일 조사해서 응답 -> http 프로토콜로 응답 -> 브라우저 수신

URL

http://example.com:80/toys?p=foo#hello

1. http: Protocol - 어떤 방법으로 파일을 전해주고 수신받는지에 대한 약속이다. 프로토콜은 https, ftp, ssh, ssl등 여러가지 가있다.

2. example.com : Hostname - 해당 주소의 주인의 이름이라는 뜻이다. 원래 주소는 IP address를 쓰는데 127.0.0.1(localhost) 처럼 숫자로 되어있다 이것을 기억하기 어려우므로

DNS(Domain Name Service)를 통해 example.com이라고 이름을 붙여 표시하는 것이다.

3.  :80 : Port -  http 는 위에서 기본적으로 80번 포트를 사용한다고 했다. 80xx로 x에 두자리를 더 붙일수 있다 00~99까지 이다.

4. /toys : path - 요구사항있는 곳으로 가기위한 길

5. ?p=foo :Query parmeter  -요구 사항이 있는 곳에서 해당 정보가 있는 번호 (정보의 아이디 값이라고 생각하면 된다)

6. #Hello : fragment - 많은 분량의 페이지 정보중 해당 지점을 볼수 있도록 옮김 (서버로 요청하지 않고 브라우저 상에서 동작함)

그럼 사용자가

<http://example.com/foo/logo.png?p=1#tricky>

라고 요청을 보내면 http 가 처리하는 방식을 보면

request

GET /foo/logo.png?p=1 HTTP/1.1

1. GET : method - 정보를 전송할 때 사용하는 방식 이다 . POST, HEAD,PUT  등 여러 전송방식 있다.

2. /foo/logo.png?p=1 : 요청한 파일의 위치

3. HTTP/1.1 : http 1.1 버전의 프로토콜 사용

그럼 서버가 요청을 받고 다시 브라우저로 넘길 때 response 하는 방식을 보면

response

HTTP/1.1 200 OK

1. HTTP/1.1 : http 1.1 버전의 프로토콜 사용

2. 200 : statust code - 응답 상태 표시

3. OK : reason phrase - 응답 상태에 대한 이유

200 OK : 해당 파일 이 있어서 응답 양호

302 Found : 지금 주소에는 없지만 해당 주소 다른곳에 있다.

404 Not Found : 해당 파일이 존재 하지 않아 페이지를 로드 할 수 없다.(응답할 수 없다)

500 Server Error : 요청 중 서버가 고장났다. (서버측 문제

**JavaScript**

* 이벤트 - 유저가 페이지 에 행하는 모든 동작

이벤트 종류

마우스

onmouseover - 마우스(커서)가 지정한 요소에 올라갔을 때 실행

onmouseout - 마우스가 지정한 요소에서 벗어났을 때 실행

onclick - 마우스가 지정한 요소를 클릭했을 때 실행

ondbclick  - 마우스가 지정한 요소를 더블클릭했을 때 실행

키보드

onkeypress - 지정한 요소에서 키보드를 눌렀을 때 실행(shift, ctrl, 등 기능 키 및 방향키 제외)

onkeydown - 지정한 요소에서 모든 키가 눌렸을 때 실행

onkeyup - 지정한 요소에서 키보들 눌렀다 떼었을 때 실행

기타

onfocus - 지정한 요소에 포커스가 갔을 때 실행

onblur - 지정한 요소에서 포커스가 생겼다 잃었을 때 실행

onload - 지정한 요소의 하위 요소를 모두 로딩 했을 때 실행

onunload - 문서를 닫거나 다른 페이지로 이동할 때 실행

onsubmit - 폼 요소에서 submit 버튼을 눌렀을 때 실행

onreset - 폼 요소에서 reset버튼을 눌렀을 때 실행

이벤트 핸들러 - 이벤트 발생시 자바스크립트 실행문을 실행하는 것

이벤트 등록 방식은 두가지이다

직접 요소 이벤트 등록 : 태그에 직접 등록 하는 방법

ex <input type='text' id='myevent' onblur='functionName'>

 DOM을 이용한 이벤트 등록  : script 나 js 파일에 등록하는 방법

ex)  <input type='text' id='myevent'> html에 작성

script 내부나 js 파일에

document.getElementById('myevent').onblur=function(){ code;}

window 객체는 브라우저의 객체 (BOM의 최상위 객체 이다.)

window 객체는 document, screen, location, history, navigator 하위 객체를 포함하고 있다. 이들을 통틀어 BOM이라고 한다.

window 객체의 메서드 (window는 브라우저에 내장된 객체의 최상위 객체이므로 따로 앞에 window.alert() 처럼 window를 붙이지 않고 alert()만 작성해도 된다.)

open('url 경로','창 이름', '옵션') - 새 창을 열 때 사용

alert('내용') - 경고 창을 띄움

prompt('내용', 기본default답변) - 질의응답 창을 띄움

comfirm('내용') - 확인/취소 창을 띄움 (확인이면 true 취소 면 false 값 리턴)

자바스크립트의 css 조작

document.선택자.style.~

[자바스크립트 css 스타일 조작](https://www.w3schools.com/jsref/dom_obj_style.asp)

함수

function 함수명(parameter){

    block

    return ~

}

다음 으로 나온것이 document.getElementById('id).innerHTML이라는 코드인데 이것을 알려면 DOM(Document Object Model) - 문서 객체 모델을 알아야한다.

파이썬에서 클래스라는 것을 배우면서 클래스명.프로퍼티 혹은 매서드 이런식으로 클래스의 프로퍼티, 매서드를 불러서 사용한 것을 기억해보면 document라는 클래스안에 getElementById라는 프로퍼티 혹은 매서드를 불러왔다고 생각하면 쉽다.

일단 기본적으로 파이썬에서도 object가 최상위 객체 였고 클래스면 모두 object를 상속 받은 클래스 이므로 쓸 때 생략해도 되는 것을 알고있다.

자바스크립트도 마찬가지다.

document객체는 window 객체의 속성이므로 사용할 때 window.document라고 사용해야하지만 window는 생략할 수 있다.

dom에 대해 자세하게 다루려면 끝이 없을 것 같다, 내가 느끼기에 일단 필요한 것만 정리를 해야겠다.

선택자에 대해서 일단 말하면 직접 선택자(절대경로라고 생각) 인접 선택자(상대 경로라고 생각) 이있다.

선택자

직접 선택자

* + document.getElementById('id명') - 해당 아이디를 가진 요소를 선택
  + document.getElementByClassName('class명') - 해당 클래스 명을 가진 첫번째 요소 선택
  + document.getElementByTagName('태그명') - 해당 태그명을 가진 첫번째 요소 선택

두번 째 세번 째 요소를 선택하고 싶을 때는 해당 매서드 뒤에 리스트 처럼 [0],[1],[2] ... 를 쓰면 된다. 첫번 째 요소는 0인데 생략 가능하고 두번째 요소를 선택하고 싶으면 [1]을 붙여주면 된다.

ex) getElementByTagName('p')[1] .~

인접선택자는 수업 시간에 다루지 않으니 나도 다루지 않겠다. 알려면 문서 객체 구조를 알아야되고 설명할게 너무 많아질 것 같다.

CSS 선택자 를 이용하여 선택할 수 있는 방법도 있다.

* + document.querySelector('#id, .class ,...') 해당 아이디, 클래스명 등을 가진 첫번 째 요소 선택
  + document.querySelectorAll('#id, .class ,...') 해당 아이디, 클래스 명 등을 가진 모든 요소 선택

속성 관련 매서드

document. 선택자. 속성명 - 선택한 요소의 지정한 속성값을 리턴

document.선택자.getAttribute('속성'); - 선택한 요소의 지정한 속성값을 리턴

document.선택자.속성명='value' - 선택한 요소의 지정한 속성값을  value값으로 수정 혹은 생성

document.선택자.setAttribute('속성','value'); - 선택한 요소의 지정한 속성값을  value값으로 수정 혹은 생성

innerHTML 속성  매서드

document.선택자.innerHTML;  - 선택한 요소의 모든 하위 요소를 문자 데이터로 리턴

document.선택자.innerHTML='value'; - 선택한 요소의 모든 하위 요소를 value값으로 수정 혹은 생성

document.write('내용'); - 내용을 출력한다

document.writeln(); - 내용 한줄을  출력한다

주의할 점 : document.write를 페이지가 로드가 끝난 후 핸들러나 콘솔을 통해서 실행 시키면 이전 페이지 내용은 모두 지워지고 document.write로 쓴 내용으로 대체된다.

document 객체 프로퍼티

document.title -  title 요소값을 반환

document.body - body 요소값을 반환

document.head - head  요소 값을 반환

document.forms - form 요소 목록을 반환

document.images - 이미지 요소 목록 반환

document.anchors - a 요소 목록 반환

document.links - 링크 요소 반환

document.scripts- script 요소 값 반환

이외에도 window , screen, location 등 다양한 객체들이 있다.

JavaScript 적용방법

1. inline :  inline은 많이 안쓰이는데 많이 쓰이는 부분이 event를 발생시킬 때 onclick, onblur, onmouseover, 등등 이벤트를 다룰 때 자세하게 설명하겠지만 이 이벤트 에 쓰는 문법은 자바스크립트 문법으로 써야한다.

2. inner :  <script type=text/javascript></script> script태그를 써서 그 안에 자바스크립트를 쓰는 것이다. 스크립트 태그 안은 자바스크립트 문법으로 써야하고 자바스크립트 문법으로 인식한다.

3.external : <script type=text/javascript src='~.js'></script>

이렇게 src 속성을 써서 외부의 js 파일을 불러와 쓰는 것이다. css처럼 마찬가지로 따로 외부로 파일을 불러 써야 좋다. 관리의 편리성과 재사용이 용이하기 때문이다. 또한 캐싱면에서도 좋은데 외부로 따로 관리를 하면 html위에 써있을 때와 달리 페이지를 리로드 할때마다 다시 읽어야하는 것이 아니라 한번 읽었으므로 페이지를 리로드할 때 미리 읽었던 파일을 꺼내기 때문에 전송할 때 경량화가 되어 좋다.

그리고 위치에 대해서도 말을 하자면 css는 헤드 태그 안에 js는 바디 태그 끝에 쓰는 것이 네트워크 체감 속도 으로 좋다.

html도 위에서 차례로 코드를 읽어가며 코드를 수행하는데 헤드에 무거운 js파일 이 있으면 페이지의 글자가 로드 되기 전에 js파일을 불러오면 js파일이 크기 때문에 페이지가 뜨는 시간이 길어져 사용자는 로드 되는 시간이 느리다고  인식하겠지만 맨 뒤 바디 태그 끝에 쓰면 페이지 가 보여지는 부분을 모두 로드 하고 js파일을 로드하면 사용자는 일단 기능과 시각적인 것들이 눈에 보이므로 페이지 로드 시간이 빠르다고 느끼고 편의성을 느낄 것이다. (css=눈에 보이는 파일, js= 사용자가 기능을 사용할 때 사용하는 파일)

그리고 웹 작업을 할 때 구글 크롬을 사용하는 것이 좋다. 그 이유는 크롬에서 제공하는 inspect(검사) 기능 때문인데 개발자 모드 인지 자세하게는 잘 모르겠다. 이 항목에 들어가면

css제작, console, network등 개발자가 개발 할 때 필요한 기능을 모두 갖춰져 있다.  그리고 문법 검사와 디버깅도 해주기 때문에 꼭 연습 할 때 자주 이용해야 한다.

* CSS 적용 방법

Inline : 태그 안에 style="내용;"으로 사용하는 방법 style 안에 내용은 css의 문법을 사용해야 한다.

Internal : head태그 안에 <style></style>태그를 넣고 스타일 태그안에 css를 작성하는 방법

External : <link rel='stylesheet' type='text/css' href="css파일 주소" media='all'>를 이용해 외부의 css파일을 불러와 적용시키는 방법(파일 위치는 href에 작성 , 나머지는 몰라도 현재 수업에 상관 없다.)

link방식외에도 import 방식이 있다.

<style type='text/css'>

@import : url('path')

</style >

이 방식으로도 불러 올 수 있다.

이 것의 사용용도는 스타일 태그안에 자신이 적용한  css와 함께 외부 css 파일도 적용 시킬 수 있다.

JS , CSS 파일 불러올 때 Tip

절대주소  : 파일이나 url(http:// 부터 시작)의 전체적 주소

상대주소 : 상위 디렉터리  -> ../

       하위 디렉터리 -> /

       현재 디렉터리 -> .(생략 가능)

상대주소를 쓰는 편이 이식성 부분과 수정할 때 매우 좋은 것 같다.

CSS 문법

선택자(selector) { property1 : value;  property2 : value; ...}

선택자는 선택된 태그, 클래스, 아이디 등 해당 선택자에 선택된 것들은 그 다음에 쓴 property(color, backgroud-color ,border...)에 쓴 value값이 적용된다.

inline으로 쓰면 한 컴포넌트? 밖에 css를 적용 못시키지만 inline, external처럼 css를 따로 쓰면 중복도 제거할 수 있고 관리하기도 더 편하다. 밑에서 쓴 것처럼 기본적 위치는 head태그이다.

CSS 선택자

Selector

Element : 태그 이름 으로 지정 a, p, div, span ,.....

요소명{속성: 값}

class : 클래스 이름으로 지정

.클래스명{속성:값}

id : 아이디 이름으로 지정

#아이디명{속성:값}

선택자의 우선순위는 id > class > element이다.  즉 선택자의 프로퍼티가 적용 될 때 같은 컴포넌트인데 클래스와 아이디의 프로퍼티가 중복되면 아이디의 프로퍼티를 따라가는 것이다.

참고로 id값은 중복되면 안된다. 그냥 해당 컴포넌트를 하나의 객체로 만들어버리는 것이다.  그래서 특히 css를 사용할 땐 컴포넌트들의 아이디와 클래스 값을 지정하는 일이 매우 중요하다.

전체 선택자

\*{속성 :값} : 전체 요소 선택

가상 선택자

 a : link{속성 :값} - 링크 걸린 텍스트 선택

a : visited {속성 :값} - 링크 방문을 했던 텍스트만 선택

a:hover {속성 :값} - 링크 걸린 텍스트에 마우스를 올렸을 때

a: focus {속성 :값} - 링크 걸린 텍스트에 포커스가 생성되었을 때

a: active{속성 :값} - 링크 걸린 텍스트에 마우스를 눌렀을 때

하위 선택자

ex)

div h2{속성 :값} -div 태그안에 있는 h2태그만 선택

span #id {속성 : 값} - span 태그 안에 있는 id값을 id를 가진 요소만 선택

그룹 선택자

h1,h2,p{속성:값} - h1, h2, p태그 선택

종속 선택자

요소명.클래스명{속성:값} - 선택한 요소 중 지정 클래스를 갖는 요소만 선택

요소명#아이디명{속성:값} - 선택한 요소 중 지정 아이디를 갖는 요소만 선택

css comment

/\*내용 \*/ 형식으로 주석을 단다.

css color

컬러를 들어가기전에 구글 크롬 앱스에 들어가 color picker와 아톰 패키지에서 color picker를 다운 받아서 사용하는 것을 추천한다.

크롬 앱스의 color picker는 웹 상에 있는 색깔을 클릭하면 해당 색깔의 정보를 주어서 개발 할 때 그대로 사용할 수 있다.(클론 코딩할 때 정말 필요)

아톰 패키지에서도 마찬가지로 색을 선택을 하면 자동으로 rgb나 hex코드로 바꾸어줘 개발 할 때 정말 편하고 또한 해당 색깔을 쓰면 그 색깔을 편집기 위에서 표시해준다.

https://khlug.org/image/1985b0ad1eb40713.pnghttps://khlug.org/image/9915b0ad1eb40575.pnghttps://khlug.org/image/1465b0ad1eb40506.png

css 컬러 값(value)은 기본적으로name, rgb, hex코드를 쓴다.

name은 lightblue, red, cyan, yellow, 등 색깔의 영어이름 을 써서 표현하는 것이다.

rgb(red{0~255},green{0~255},blue{0~255},transparent{0~1})가 기본적인 형태이다 transparent는 투명도를 정해준다.

hex  코드는 #XXXXXX 형태로 6자리를 16진수로 표현하는 방식이다.

컬러 표시하는 방법을 알았으니 다음으로 css에서 많이 사용되는 속성 몇가지만 정리하려고한다.

글자체

* + font-family : value; -모양 (글꼴 [구글 폰트 제공 API](https://fonts.google.com/) 에 들어가면 많은 폰트와 적용시키는 방법까지 알려준다. 꼭 들어가보는 것을 추천한다.)

* + font-size : value; - 글자크기 (px, %...단위)

* + font-weight : value; - 글자 두께 (bold , normal ...)

* + font-style : value; - 글자 기울기 (italic , normal...)

* + font : [두께 , 스타일] [사이즈] [글꼴] - 함축 방식 쓴 순서대로 value값 지정한다.

문단 속성

* + color : value; - 텍스트 색상 (name , rgb , hex)

* + text-align : value; - 블록 요소 내에 인라인 요소를 정렬(블록 요소에만 적용)(center, left, right ...)

* + vertical-align : value; - 인라인 요소간의 세로 정렬(인라인 요소에 직접 적용)(top , middle , bottom ...)

* + text-indent :value; - 블록 요소 내에 텍스트 내어/들여 쓰기(px)

* + text-decoration : value; - 텍스트 꾸미기(underline)

배경 속성

* + background-color : value; - 배경 색상(name, rgb, hex)

* + background-image : value; - 배경 이미지 (url('이미지경로'))

박스 모델 (중요)

* + width/height : value; - 가로(너비)/세로(높이) (px)

* + margin/padding : value; - 바깥쪽 여백 / 안쪽 여백 (px)(margin-top, margin-right, padding-bottom, padding-left처럼 상하좌우 한쪽만 선택해서 여백 크기를 조정할 수 있다.)

또한 위치를 지정해주지 않는 경우

2개를 쓰면 margin : apx bpx;( 상하 a픽셀 , 좌우 b픽셀 적용)

3개를 쓰면 margin : apx bpx cpx;(상 a픽셀 좌우 b픽셀 하 c픽셀 적용)

4개를 쓰면 margin : apx bpx cpx dpx;(상 apx 우 bpx 하 cpx 좌 dpx 적용)

padding 도 똑같이 적용

* + border-style : value; - 선 모양(solid, dotted, dashed....)

* + border-width : value; - 선 두께(px)

* + border-color : value; - 선 색깔(name, rgb, hex)

* + border : [두께][스타일][색깔] - border 함축 방식 쓴 수서대로 value적용

display속성

* + display : value; - 요소의 성격을 바꿀 때 사용 ( inline, block, inline-block(인라인 속성에 너비/높이 값 사용 가능), none)

float 속성(블록 요소를 왼쪽, 오른쪽 정렬할 때 사용 ,float를 사용했으면 clear를 통해 float해지하자)

* + float : value; - 블록 요소 좌우 정렬 (left, right)

* + clear : value; - float 해지 할 때 사용(left, right, both)

포지션 - 위치 지정

position : value; - 위치를 지정(relative(현재 위치 기준 위치 지정), fixed(스크린 기준 위치 고정))

요즘 css는 트위터가 jQuery를 이용해 만든 bootstrap을 대부분 사용한다. 위에 내용은 기본적으로 알고 있고 제대로 공부할 거면 bootstrap 공부를 추천한다.

bootstrap에 대해 더 공부하고 싶다면 jQuery 공부도 같이하는 것을 추천한다.

* <input type='submit'>과 <button></button>의 기능은 같다. 기본적으로 버튼 태그는 type을 명시하지 않으면 submit 기능을 한다. 즉 둘 다 폼 태그안에서 쓰이며 정보를 전송할 때 쓰이는 태그이다

button 태그의 속성

type='submit' - 폼의 정보 전송 기능

type='reset' - 폼 작성 내용 초기화

type='button' - div, span처럼 기능이 아무것도 없다(?) 하지만 자바스크립트 기능을 이용할 때 (event) 자주 사용하는 속성

formtarget -> form에서 할 수 있지만 따로 분류할 수 있기 때문에 더 다양하게 폼 정보 전송을 제어할 수 있다.

formtarget="\_self" 현재 창에서 이동

formtarget="\_blank" 새로운 창 생성

기능이 똑같은 태그가 중복되서 있는데 이렇게 구분 지은 이유가 있다.

input 에 비해 button 은 css를 적용하는데(꾸미는데) 자유롭다. bootstrap을 사용해 다양한 종류의 버튼을 꾸미고 제작하여 사용할 수 있다.

또한 태그인 만큼 버튼 태그 안에 다른 태그도 삽입할 수 있다.

input은 input 태그 답게 입력하는 형식이 자유롭다. 또한 html5로 개발이 되고 다양한 입력 필드를 만들어줘서 개발자가 개발하기 더 편해졌다.

Input 태그 분석

input : form 태그 안에서 사용자의 정보를 입력받는 정말 중요한 태그(파이썬 input매서드와 똑같다고 보면 된다.)

type, value, name 속성이 있는데 세가지 모두 중요하다.

* + type은 어떤 input을 받을 것인지 유형을 결정해주는 속성이다.
  + value는 입력태그의 디폴트값(사용자 변경 가능) , 폼에서 정보를 넘길 때 queryparameter의 값(인자?)
  + name은 폼에서 정보를 넘길 때 서버로 전달되는 queryparameter(변수)의 이름(사용자 변경 가능)
  + form기능을 배우는 week13때 더 자세하게 설명해야겠다.

type은 html5로 넘어가면서 정말 다양한 속성이 생겼다.

* + type='text' :

텍스트 필드로 입력을 받는 유형이다. 특징은 따로 없고(글을 입력하는 창이니까?) value값을 적으면 디폴트 값이 되어 value값에 지정한 내용이 기본으로 써있는 상태에서 출력되게 된다.

참고: (같이 자주 쓰이는 속성 정리)

value : 텍스트 필드에 기본적으로 입력되어있는 내용을 만들 수 있다.

placeholder :  이 속성은 value와 달리 미리 써있는 상태가 아니라 아무것도 안 쓰여 있으면 흐린 글씨로 써있어 클릭하면 내용이 사라지고 빈칸이 되는 기능이다. 사용자에게 입력양식을 알려줄 때 많이 쓰인다.

size : 텍스트 필드의 길이를 지정 (영어가 1글자 한글이 2글자)

maxlength : 텍스트 필드에 입력할 수 있는  최대 문자 개수 를 정해준다.

* + type='password'

기본적으로 text와 같지만 다른 점은 입력값이 ● 이런식으로 보여 사용자도 못보게 된다는 것이다.  단 text처럼 value값을 넣어 기본값을 설정하지는 못한다.

참고 :

size

maxlength

pattern : 허용하는 입력식을 regex(정규표현식)으로 표현해 개발자가 지정한 형식 대로 써야지만 제대로 처리한다.(ex 숫자 + 영어 대문자 + 영어 소문자 순서로 입력해야만 인정(?)을 해주게 만들 수 있다.

* + type='radio'

라디오 버튼이다. 라디오 버튼은 동그라미 체크 형식을 갖고 있고 중복 선택이 불가능 하다.  한 질문에 객관식으로 답을 선택하게 하는 유형일 때 (설문조사)  자주 쓰이는데 중요한 것은 한 질문에 여러가지 라디오 버튼을 만들 때 그 항목들의 name값을 같게 해야 같은 그룹으로 인식을 한다.

checked라는 속성을 이용하여 초기 상태일 때 어떤 값이 미리 선택되있게 할 수 있다.(디폴트 값을 지정할 수 있음!)

* + type='checkbox'

체크 박스다.  라디오 버튼과 똑같고 다른 점은 중복 선택이 가능 하다는 것이다.

* + type='hidden'

사용자에게 보이지 않게 하지만 서버로 전송해야할 경우 사용한다.

* + type='submit'

폼 태그 안에서 작동하는데 폼에서 action으로 지정한 페이지로 폼안에 입력된 정보들을 넘기는 기능을 한다.

* + type='reset'

폼 태그 안에서 작동하는데 폼 안에 입력한 정보를 초기화 시켜준다.

* + type='number', type='range'

number는 스핀박스, range는 슬라이드 막대를 이용하여 숫자를 입력 하게 하는 유형

참고 :

value : 초기값

min : 입력 가능한 최솟값을 지정

max : 입력 가능한 최댓값을 지정

step : 숫자의 간격을 지정 (생략하면 기본 1로 지정이 된다.)

* + type='date', type='month', type='week'

날짜를 입력하는 필드 생성

date - yyyy/mm/dd

month - yyyy/mm

week - 1월 첫째 주 기준 연도 와  주차 수 표시

* + type='datetime', type='datetime-local'

datetime - yyyy/mm/dd 와 시간까지 표시

datetime-local - 자신이 있는 지역의 날짜와 시간 표시

* + type='time'

 시간 지정하는 필드 생성

시간 00:00 ~23:59까지 지정가능 스핀박스로 입력

참고 :

value : 초기값 지정

min : 최솟값 지정

max : 최댓값 지정

step : 스핀박스를 이용해 입력할 때 증가/감소 값 지정

* + type='color'

색 선택하는 필드 생성

참고 :

list : 선택할 수 있는 색상리스트 지정했을 때 그 리스트의 id값 부여

value : 초기값 지정 hex(16진수)코드로 표시

* + type='button'

div, span처럼 아무 기능 없이 일반 버튼 형식 이 된다. 자바스크립트 이벤트와 연동해 많이 사용한다.

* + type='file'

파일을 첨부하려고 할 때 쓰이는 유형

파일선택 ,찾아보기, 선택 등으로 표시

* + type='search'

검색어를 입력하는 필드 생성

* + type='url

url을 입력하는 필드 생성

* + type='email'

이메일 주소를 입력하는 필드 생성

* + type='tel'

전화 번호를 입력하는 필드 생성

* + type='image' src='path' alt=' 대체 내용'

이미지를 전송 버튼으로 만든다.

참고할 만한 다른 속성

readonly : 해당 필드를 읽기 전용으로 바꿈 boolean 형식을 갖고있기때문에 true,false로도 지정 가능

autofocus : 페이지가 로드되면 폼의 요소중 원하는 필드에 마우스 커서가 표시되도록 함

autocomplete : 자동완성 기능 autocomplete='on/off' 로 자동완성 기능을 키고 끌수 있다.

required : 필수 입력 필드 지정하기 readonly와 마찬가지로 boolean형식을 갖고 있기 때문에 true,false로도 지정 가능

html  특수문자 처리

&nbsp 빈칸 (공백)

&ndash ~

&lt <

&gt >

&amp &

정규표현식

[regexr.com(정규표현식 사이트)](https://regexr.com/) (정규표현 테스트 해볼 수 도 있고 남들이 만든 정규표현식도 쓸 수 있다. 유용한 사이트)

정규표현(RegularExpression , RegExp, Regex )

**정규식**(正規式)은 특정한 규칙을 가진 문자열의 집합을 표현하는 데 사용하는 형식 언어(참조 : 위키백과)

자바스크립트에서 정규표현(RegExp) 객체 는 입력 요소에 데이터를 규칙에 맞게 작성했는지 판단해서 알려주는 객체 이다. (속성 pattern에서도 정규표현식을 써서 true, false를 리턴)

자바스크립트 뿐만 아니라 정규표현은 정말 유용하고 다양하게 쓰이기 때문에 웹개발 쪽에 관심이 있다면 꼭 알아야할 항목이다.

기본적으로 / 패턴 / 검색옵션  형태를 갖고 있다.

자바스크립트에서는 정규표현 객체를 생성하는 형태는 두가지이다(나중에 객체 생성할 때 배우는 형식)

var 변수  = new RegExp(패턴, 검색 옵션)

var 변수 = /패턴/검색 옵션

[] : 대괄호 안에 있는 한개의 문자의 의미

(범위를 지정할 때 는 - 를 쓴다 [a-c]이면 [abc]와 같고 [0-9]이면 [0123456789]와 같다. 만약 원하지 않는 문자열로 지정하고 싶다면 [^] 로 지정하면 된다.)

{} : {} 앞에 있는 문자(열)의 개수를 정한다. {시작 번호, 끝 번호}

(시작 번호를 생략하면 처음 부터고 끝 번호를  생략하면 끝까지 이다. 시작번호<= 문자 개수<=끝번호 처럼 시작번호 , 끝번호 모두 포함한다)

{n} - {n}앞에있는 문자 n번 반복되는 것을 의미

{n,m} - {n,m}앞에 있는 문자가 n번 부터 m번까지 반복되는 것을 의미

{n,} -{n,}앞에 있는 문자가 n번 이상 반복되는 것을 의미

() : 소괄호 안에 있는 문자를 그룹화 한다.

([ab]면 a와 b 중 한 문자만을 가리키지만 (ab)로 하면 ab가 그룹화 되어 ab를 가리킨다.)

그룹화는 중요한 사항이다.

재참조할 때 편히 쓸수 있는데

그룹내에서 \0은 매치된 전체 문자열을 가리키고 \1 은 첫 번 째 그룹에 해당하는 문자열 \2 는 두번 째 그룹에 해당하는 문자열이다. \n은 n번 째 해당하는 문자열이다.

/(\w+)+\1/을 하면 /(\w+)+(\w+)/와 같은 뜻이 된다.

그룹에 이름을 붙이는 것 도 가능하다 (?P<그룹명>..)을 통해서 가능 한데 재참조 하는 것도 가능해서 (?P=그룹명)으로 재참조 할 수 있다.

 | : or 의 기능을 한다.

(010|011|017) 로 하면 010, 011, 017 중 하나 와 일치하는 패턴이다

^ : 대괄호 안에서는 제외하는 의미로 쓰이지만 그 외에는 문자열을 시작하는 것을 의미

$ : 문자열이 끝나는 것을 의미

[a-z] : 영어 소문자 하나를 의미

[A-Z] : 영어 대문자 하나를 의미

[가-힣] : 한글 하나를 의미

. : \n(line escape)을 제외한 임의의 한글자를 의미

\.  : .을 의미

[.] : .을 의미

\w : 알파벳, 숫자, 밑줄(\_) 하나를 의미 [a-zA-Z0-9]와 동일

\W : 알파벳, 숫자, 밑줄(\_) 를 제외한 문자 하나를 의미 [^a-zA-Z0-9]와 동일

\d : 숫자 하나를 의미 [0-9]와 동일

\D :  숫자를 제외한 문자 하나를 의미 [^0-9]와 동일

\s : 공백(space, tab ...) 하나를 의미 [ \t\n\r\f\v]와 동일

\S : 공백을 제외한 문자 하나를 의미 [^ \t\n\r\f\v]와 동일

\* : \* 앞에 0회 이상 반복하는 문자 를 의미 {0,}와 동일

a\*b 면 b, ab, aab, aaab ... 와 일치한다

+ : +앞에 1회 이상 반복하는 문자를 의미 {1.}와 동일

a+b 면 ab,aab,aaab, ... 와 일치

? : ?앞에 문자가 있을 수 도 있고 없을 수 도 있는 것을 의미 {0,1}와 동일

as?b 면 ab, asb, adfdfdfdfdb와 일치

?는 \*, + {} 뒤에 붙어 검색 방식을 다르게 할수 있다. (greedy 방식을 lazy방식으로)

기본적으로 패턴을 검색할 때 greedy하게 검색을 한다. 최대한 많은 문자를 일치시키려고 하는데 한번 검색된 상태(검색결과를 리턴)를 소모되었다고 한다.

소모되었으면 후에 다시 검색을 했을 때 검색 하지 않고 넘기는 상황이 올 수 있다. 그래서 ?으로 lazy방식으로 바꾸어 검색 결과를 바꿀 수 있다.

?을 붙이면 가능한 최소한의 반복을 수행하도록 한다.

예를 들어 internationalization 이라는 단어에서

i\w+d를 했다면 internationalization을 검색 결과로 내놓을 것이다.

i\w+?d를 했다면 intern, ion, ization 을 검색 결과로 내놓는다.

/^[A-Za-z]{3}$/  : 시작부터 끝까지 영어 대문자 와 소문자 3글자 패턴

/^[A-Za-z]{3}/ : 시작에서 부터 3글자가 영어 대문자와 소문자

/[A-Za-z]{3}$/ : 끝에서부터 3글자가 영어 대문자와 소문자

/^[a-zA-Z\s\'\-]{2,15}$/ 의 뜻 : 시작에서 부터 끝까지 2~15글자이고 영어 대문자, 소문자, 공백, ', - 을 포함 한 패턴

\d : 숫자 하나 패턴 [0-9]와 같은 뜻

html form 설명

<form> 태그

속성

action ='경로' : 폼에서 받은 정보를 해당 경로에 있는 파일로 넘겨준다.(정보를 넘겨줄 대상 지정)

method = 'post/get' : 폼에서 받은 정보를 넘겨줄 때 쓰는 방법을 지정 (정보를 넘겨줄 방식 지정)

get : default로 지정  폼에서 받은 정보를 url에 queryparameter를 작성하여 정보를 보낸다. (민감한 데이터 의 경우 이 방식은 url에 정보가 모두 노출되므로 절대 사용하면 안된다.)

get방식으로 보내는 경우

GET /?say=Hi&to=Mom HTTP/1.1

Host: foo.com

post : url에 등록하지 않고 정보를 몰래 보낸다. (크롬 검사 기능에서 확인 가능)

POST / HTTP/1.1

Host: foo.com

Content-Type: application/x-www-form-urlencoded

Content-Length: 13

say=Hi&to=Mom

이런 식으로 url에 등록하지 않고 정보를 넘긴다

name='이름' : 폼에 이름을 붙인다.

<fieldset></fieldset> : HTML요소를 박스로 묶어준다. (주로 form태그와 같이 사용됨)

<legend></legend> : fieldset 박스의 제목을 달아준다.

<label></label> :

input 태그의 편의성, 사용성을 높여주기 위해 주로 사용된다. 라벨의 아이디를 for속성을 통해 input(예를들어 체크박스)과 묶어주면 체크박스가 가리키는 툴팁을 클릭했을 때도 체크가 된다.

for='id값' : 해당 위젯에 맞는 id를 참조한다

<textarea></textarea> :

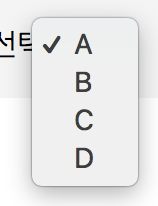
다중 텍스트 구문

속성

cols='숫자' : 박스 한 줄에 몇 글자 까지 들어가게 할 것인지 정하는 속성

rows='숫자' : 박스의 높이를 몇 줄 까지 할 것인지 정하는 속성

<select></select> : 선택할 옵션을 한가지만 보여준다



이런 식으로 저런 항목들을 만드는 태그가 option태그이다.

기본적으로 select의 name 을 쿼리 파라미터로 받고 option의 value값을 그 인자로 받아서 넘긴다.

속성

multiple : 항목을 여러개 선택할 수 있게 해준다.(여러개를 선택할 땐 ctrl or 마우스 드래그로 가능하다.)

size='숫자' : 기본적으로 보여주는 항목의 개수를 정하는 속성 default는 mutiple이 아니면 한 개,  mulitple이면 4 개가 기본값이다.

 <option></option> : select 태그 안에서 항목을 만드는 태그

속성

selected : 기본적으로 체크되어있는 항목 지정

<optgroup> </optgroup>  : 항목들을 그룹지어 준다.

