[11/15] 십이지로 알아보는 연애유형(2) 톺아보기

1) createElement()

```
var a = document.querySelector('.answerBox');
var answer = document.createElement('button');
    answer.classList.add('answerList','my-3','py-3','mx-auto','fadeIn');
    a.appendChild(answer);
    answer.innerHTML = answerText;
```

전체 구조

- 요소 생성 : HTML 문서에서 지정한 태그의 요소 생성, 위치 미지정 상태이므로 보이지 않음
  - 구조 : 변수 = 요소를 생성한다 '특정 태그를'

```
var answer = document.createElement('button');
```

- 요소 추가 : 새로운 요소를 자식 요소로 추가하는 것
  - 구조 : 부모 노드의 요소명.appendChild(자식요소)

```
a.appendChild(answer);
```

  - appendchild : 특정 부모 노드의 자식 노드 리스트 중 마지막 자식으로 추가
  - a 는 answerBox를 의미, qna 섹션의 div 로 answerBox가 존재
  - 즉, <qna> 하위인 <answerBox> 에 answer 를 추가

- 자식 요소값 설정 : 자식요소의 메서드를 수행하거나 속성 값 설정
  - 구조 : 자식요소.메서드 혹은 자식요소.속성 = 속성 값

```
answer.innerHTML = answerText;
```

- classList : 자바스크립트에서 HTML 요소의 Class 추가,제거등, element의 읽기전용 속성
  - 속성 : add, remove, contains(요소에 특정 클래스가 포함되어 있는지 테스트), toggle(미속성값시 추가, 추가 제거), replace(이름변경)
  - HTML 에서 class 를 여러개 늘려도 되지만 JS 에서 늘리는 방식을 채용
  - ('my-3 , py-3 , mx-auto') : 한 줄로 작성가능

```
function addAnswer(answerText){
    var a = document.querySelector('.answerBox');
    var answer = document.createElement('button');
        answer.classList.add('answerList');
        answer.classList.add('my-3');
        answer.classList.add('py-3');
        answer.classList.add('mx-auto');
        answer.classList.add('fadeIn');
        a.appendChild(answer);
        answer.innerHTML = answerText;
```

addAnswer에게 answerText 전달
a 는 answerBox 태그를 지칭
answer 는 button 태그를 지칭

answer의 class 추가

a의 자식요소로 answer 요소 추가 (answerBox 부모, answer 자식)
answer 를 answerText 로 지정

2) createElement : 삽입 메서드

- 삽입 메서드의 종류
  - node.append(노드나 문자열) : 노드나 문자열을 node 끝에 삽입
  - node.prepend(노드나 문자열) : 노드나 문자열을 node 맨 앞에 삽입
  - node.before(노드나 문자열) : 노드나 문자열을 node 이전에 삽입
  - node.after(노드나 문자열) : 노드나 문자열을 node 다음에 삽입
  - node.replaceWith(노드나 문자열) : node를 새로운 노드나 문자열로 대체

- 삽입 메서드의 예시(https://ko.javascript.info/modifying-document)

```
let div = document.createElement('div');
div.className = "alert";
div.innerHTML = "<strong>안녕하세요!</strong>";
document.body.append(div);
```

1. 요소 생성 :  요소를 만듬
2. 클래스 이름 지정 : 'alert'로 설정
3. statement : HTML에 내용 작성
4. 위치 지정 : div가 삽입될 곳

- 실습을 통해 확인된 예시 (하도 많이 봐서...)

```
var a = document.querySelector('.answerBox');
var answer = document.createElement('button');
    answer.classList.add('answerList');
    a.appendChild(answer);
    answer.innerHTML = answerText;
```

1. button 에 요소 생성
2. 읽기전용으로 answerList 를 추가
3. a 의 위치에 answer를 넣음
4. statement 로 행동할 기능

- 실습을 통해 확인된 예시2

```
const modalElem = document.createElement('div');
modalElem.classList.add('modal');
modalElem.appendChild(modalContentElem)
modalElem.appendChild(closeBtn)
modalElem.addEventListener('click', stopPropagation);
```

1. 요소생성
2. 읽기전용 속성 추가
3. 위치지정(1)
4. 위치지정(2)
5. 행동할 기능, 함수도 가능

- 정리하자면,
  - let answer = document.createElement('button')
    - who, where : 요소를 생성, 해당 요소가 어디를 지칭하는지 설정한다.
  - answer.classList.add('answerList')
    - what : 요소의 속성값을 부여한다.
    - 주로 읽기용도로 classList.add를 통해 찾기 용이하도록 설정한다.
  - a.appendChild(cl)
    - where : a 라는 곳에 (아래에) cl 을 넣는다.
    - 속성값인 answerList도 따라가는 것이다. 이를 통해 찾기 용이해진다.
    - 생성된 요소가 어디로 갈지를 의미한다.
  - answer.innerHTML = answerText
    - how : 변수 answer 가 어떤 행동을 할지, statement 로써 기능을 정의한다.

- classList.Methods
  - .add(string) : 지정한 클래스 값을 추가, 추가하려는 클래스가 존재시 무시
  - .remove(string) : 지정한 클래스 값을 제거, 존재하지 않는 클래스면 에러 발생
  - .contains(string) : 지정한 클래스 값이 존재하는지 확인, True or False 반환
  - .replace(old, new) : old class 를 new class 로 대체
  - .item(number) : 인덱스 값을 활용하여 클래스 값을 반환

3) 십이지로 알아보는 연예유형 : createElement 상세 정리

```
function addAnswer(answerText, qIdx){
    var a = document.querySelector('.answerBox');
    var answer = document.createElement('button');
        answer.classList.add('answerList','my-3','py-3','mx-auto','fadeIn');
        a.appendChild(answer);
        answer.innerHTML = answerText;

        answer.addEventListener("click", function(){
            var children = document.querySelectorAll('.answerList');
            for(let i=0; i<children.length; i++){
                children[i].disable=true;
                children[i].style.webkitAnimation="fadeOut 0.5s";
                children[i].style.Animation="fadeOut 0.5s";
            }

            setTimeout(()=>{
                for(let i=0; i<children.length; i++){
                    children[i].style.display='none';
                }
                goNext(++qIdx);},400)
        }, false);
}
```

전체 문단

- 요소 생성(class의 선언) 및 속성 부여
    - answerList my-3 py-3 mx-auto fadeIn 라는 속성을 주목하자

```
<button type="button" class="answerList my-3 py-3 mx-auto fadeIn"
onclick="js:begin()">시작하기</button>
```

html

```
var answer = document.createElement('button');
    answer.classList.add('answerList','my-3','py-3','mx-auto','fadeIn');
```

js

    - 둘의 기능적 차이는 없다.
    - 다만, HTML 의 코드를 js에서 구현하기 위해서이다.

- 요소의(선언된 class) 의 호출 : document.querySelectorAll('.answerList')

```
answer.addEventListener("click", function(){
    var children = document.querySelectorAll('.answerList');
    for(let i=0; i<children.length; i++){
        children[i].disable=true;
        children[i].style.webkitAnimation="fadeOut 0.5s";
        children[i].style.Animation="fadeOut 0.5s";}
```

js

    - 앞서 button 에 createElement 라는 요소를 생성했다.
    - querySelectorAll 은 answerList 이 포함된 모든 곳에 적용된다는 의미이다.
        - 그러나, button이 두개라면 첫번째 button 에만 적용되는 것에 주의
        - answerList 가 지정된 곳에 해당 for 문이 전부 적용된다.

- querySelectorAll 의 활용 (추후보충)
    - for 문 or 파라미터를 여러개 지정하는 용도로 사용

```js
const sections = document.querySelectorAll("#sections , #sections .section")
console.log(sections.constructor.name);
for (const i = 0; i < sections.length; i++) {
  const item = sections.item(i);
  item.style.border = "1px solid #ff0000"; }
```