

INTERACTION PROGRAMMING 2

인터랙션 프로그래밍 2

jQuery

2017. 9.14.

jQuery

\$ jQuery

JavaScript 라이브러리

가볍고 적은 코드로 많은 동작을 수행할 수 있다.

HTML / DOM / CSS 조작 기능

EVENT 제어

효과 및 애니메이션 기능

AJAX (비동기 통신)

유틸리티 메서드 제공

코어

- jQuery 함수
- 개체 접근자
- Data
- Plug-in

선택자

- 계층 구조
- 기본 필터
- 자식 필터

유틸리티

- 배열
- 문자열
- 브라우저

조작

- 내/외부 삽입
- 제거
- 복사

스타일

- CSS
- position
- width
- height

탐색

- 필터링
- 검색
- 체인

AJAX

- AJAX 요청
- AJAX 이벤트

이벤트

- 페이지 로드
- 핸들러
- 콜백

```
$( '#id' ).val();
```

```
$( '#id' ).val();
```

jQuery 메서드


```
$('#id').val();
```

Selector

```
$('#id').val();
```

Selector - id Dom 객체를 반환

```
$( '#id' ).val();
```

객체의 속성을 반환



jQuery

\$ = jQuery

`$ = jQuery`

`$ = jQuery = function`

`$(...);`

`jQuery(...);`

```
$.isArray(...);  
jQuery.isArray(...);
```


Selectors

```
$( 'div' );
```

```
$( 'div' );
```

```
$( '#header' );
```

```
$( 'div' );
```

```
$( '#header' );
```

```
$( '.content' );
```

Selectors 선택자

tag name

```
$( 'div' );
```

id

```
$( '#header' );
```

class name

```
$( '.content' );
```

Selectors 선택자

jQuery

```
$( 'div' );
```

Javascript

```
document.getElementsByTagName( 'div' );
```

Selectors 선택자

jQuery

```
$( '#header' );
```

Javascript

```
document.getElementById( 'header' );
```

Selectors 선택자

jQuery

```
$( '.content' );
```

Javascript

```
document.getElementsByClassName( 'content' );
```



```
$( 'div' );
```

```
$( '#header' );
```

```
$( '.header' );
```

```
$( 'a[href="#"]' );
```

```
$( 'li' ).length;
```

```
$( 'li' )[index];
```

```
$( 'li' ).get(index);
```

```
$( 'li' ).eq( index );
```

Pseudo Selectors

CSS 기본선택자가 아닌 Filter 를 이용해 선택하는 선택자

다양한 방식으로 선택이 가능하다.

[attribute] [attribute=value] [attribute!=value] [attribute^=value]

:header :animated :focus :has() :hidden :visible

:first :last :even :odd

:first-child :last-child nth-child() :nth-of-type() :only-child

parent > child element + next element ~ siblings

:eq() :gt() :lt() :not()

:input :text :password :radio :checkbox :submit :selected

[attribute] [attribute=value] [attribute!=value] [attribute^=value]

:header :animated :focus :has() :hidden :visible

:first :last :even :odd

:first-child :last-child nth-child() :nth-of-type() :only-child

parent > child element + next element ~ siblings

:eq() :gt() :lt() :not()

:input :text :password :radio :checkbox :submit :selected


```
$( 'li:first' );
```

Pseudo Selectors 유사 선택자

```
$( 'li:first' );  
$( 'li' ).first();
```

Pseudo Selectors 유사 선택자

```
$('li:first');
```

```
$('li').first();
```

```
$('li').eq(0);
```

Pseudo Selectors 유사 선택자

```
$( ':last' );  
last();
```

```
$( ' :has(span) ' );  
has( 'span' );
```

Pseudo Selectors 유사 선택자

```
$( ' :not(span) ' );  
not( 'span' );
```

```
$( 'li:visible' );
```


Quest 1.

li 태그들을 모두 선택하고 addClass 메서드를 이용해서
선택된 태그들에 모두 active 클래스를 추가합니다.

Quest 2.

문서내의 a 태그들 중 type-2 클래스를 가진 노드를 선택하고,
해당 태그에 current 클래스를 추가합니다.

Quest 3.

문서내의 li 태그들 중 0 index 에 위치한 노드를 선택하고,
해당 태그에 zero 클래스를 추가합니다.

Quest 4.

문서내의 a 태그들 중 span 태그를 포함하고 있는 노드를 선택하고,
해당 태그에 inner 클래스를 추가합니다.

Quest 5.

문서내의 li 태그들 중 data-role attribute 가 link 인 노드를 선택하고,
해당 태그에 role 클래스를 추가합니다.

Selectors

```
$( 'li' ).index();
```

```
$( 'li' ).index( 'li.last' );
```

```
$( 'ul' ).children();
```



```
$( 'ul' ).children( 'li' );
```

```
$( 'ul' ).children().first();  
$( 'ul' ).children().eq(0);  
$( 'ul' ).children().( ':last' );
```

```
$( 'ul' ).parent();
```

```
$( 'a:has(span) ' ).parents( 'ul' );
```

```
$( 'ul' ).prev();
```

```
$( 'li:last' ).prevAll();
```

```
$( 'li:last' ).prevAll( 'li.prev' );
```

```
$( 'ul' ).next();
```

```
$( 'li:last' ).nextAll();
```

```
$( 'li:last' ).nextAll( 'li.prev' );
```

```
$( 'ul' ).find( 'li' );  
$( 'ul' ).find( 'li a' );  
$( 'ul' ).find( 'li' ).find( 'a' );
```


Quest 6.

radio-list 아이디를 가지고 있는 노드의 자식 노드들의 길이를 구하고
console.log 메서드를 이용하여 출력합니다.

Quest 7.

type-2 클래스를 가지고 있는 노드의 부모 노드를 찾고,
그 index 값을 console.log 메서드를 이용하여 출력합니다.

Quest 8.

아이디가 radio-1 인 input radio 엘리먼트의 value 를 출력하는 여러 가지 방법을 찾아, console.log 메서드를 이용하여 출력합니다.

Quest 9.

menu 7 텍스트를 포함하고 있는 a 엘리먼트를 선택할 수 있는 여러 가지 방법을 찾아,
console.log 메서드를 이용하여 출력합니다.

Quest 10.

아이디가 radio-2 가 아닌 input radio 엘리먼트의 부모 노드를 찾고,
자식 노드 label 엘리먼트에 emphasis 클래스를 추가합니다.

Quest 11.

sub-last 클래스를 가진 li 엘리먼트를 찾고 last 클래스를 가진 부모 노드를 찾은 후,
자식노드 a 엘리먼트에 emphasis 클래스 / 모든 자식노드 a 엘리먼트들에 bold 클래스를 추가합니다.

과제입니다.

HOMEWORK

1. Quest

제출

- github에 업로드

2. 인터랙션 사이트 리서치

- 자료 수집
- 벤치마킹 자료
- 사이트 디자인 및 구조정리
- 마크업 작성

제출

1) 리서치 자료

- 파일명: 학번_이름_프로젝트 명칭_리서치.ppt/pdf ex) 20170000_김용원_헬로월드_리서치.ppt/pdf
- 카페 게시글에 답글로 제출

2) 마크업 작성

- 깃헙에 학번_이름 폴더 생성하여 커밋 후 푸시

기한

- 2017. 9. 20. 23:00 까지

