**웹 앱 개발**

**\* 자바스크립트 배열**

배열의 선언은 아래 두 가지 방법으로 할 수 있다.

|  |
| --- |
| var arr= [];  var arr= [1,2,3,4,”hello”,null,[]]; |

배열은 length, push, indexOf, join 등등의 메소드들이 존재한다. 배열은 for 문을 통해 방문할 수 있지만 forEach를 통해 방문할 수도 있다.

|  |
| --- |
| arr.forEach(fuction(value, index, object){  console.log(value);  }); |

배열에는 forEach 외에도 map 이라는 함수가 존재한다. map 함수는 배열에 새로운 값을 가진 배열로 만들어줄 수 있는 함수이다.

|  |
| --- |
| var mapped= arr.map(function(value, index, obj){  return value \* 2; <!-- arr 배열에 곱하기 2를 한 배열을 만든다. -->  }); |

배열에 map과 다르게 filter라는 함수가 존재한다. filter의 입력 파라미터 함수는 리턴 값이 bollean 타입으로 함수의 이름에서 알 수 있듯이 만들어지는 배열에 추가될 인자를 결정한다.

|  |
| --- |
| var filtered= arr.filter(function(value, index, obj) {  return value % 2 === 0 ? true : false;  }); |

**\* 자바스크립트 객체**

객체는 key, value를 가지고 있는 객체이다. 배열은 [] 표기법을 사용하고 객체는 {} 표기법을 사용한다. 아래와 같이 선언하고 불러올 수 있다.

|  |
| --- |
| var obj= {name : “crong”, age : 20};  console.log( obj.name );  console.log( obj[“name”] ); |

객체를 탐색할 때는 for-in 문을 많이 사용한다. 아래와 같이 사용할 수 있다.

|  |
| --- |
| for(key in obj) {  console.log(“key [“ + key + “], value [“ + obj[key] + “]”);  } |

객체의 key 값을 아래와 같이 배열 형태로 반환받을 수 있다. 이렇게 반환 받은 후 아래 예제처럼 forEach를 활용하여 탐색할 수 있다.

|  |
| --- |
| var keys= Object.keys(obj);  keys.forEach(function(value, index, obj){  console.log(obj[value]);  }); |

**\* DOM Node 조작하기**

DOM, 즉 HTML은 동적으로 다양하게 변경될 수 있다. DOM APIs에서는 다양한 메서드와 속성을 지원하고 있으니 이를 활용하는 방법을 배울 필요가 있다. DOM API는 라이브러리(React, Angular JS 등등)를 쓰는 것보다 대체로 더 빠르다. 물론 라이브러리보다 그 사용법이 불편하고 어려울 수는 있지만 low-level의 DOM API를 이해한다면, 라이브러리나 프레임워크를 사용하면서도 그 동작을 이해하고 유추하기가 쉽고 어떤 문제를 해결할때도 더 유리하다.

**\* Ajax 응답 처리와 비동기**

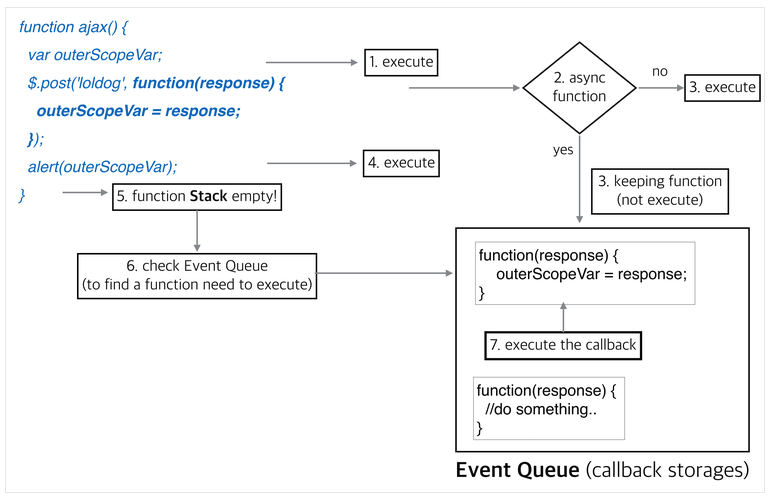
화면 전환 없이 서버에 데이터를 요청하여 서버측 데이터를 화면에 그려줄 수 있도록 해주는 기술이다. 이를 처리하기 위해 Ajax와 같은 요청이 있다면 Ajax 요청을 서버에 보내고 응답 전까지 다른 작업을 수행하고 응답이 오면 해당 데이터를 그리는 작업을 수행한다(비동기)

즉, 아래 코드를 예로 들면 addEventListener 함수의 콜백함수가 Event 큐에 들어가게 되고, 실제 서버에 요청을 보낸 후 ajax 함수는 수행을 끝내게 된다. 이 후 서버에 응답이 오면 콜백함수의 로직이 처리되게 된다.

|  |
| --- |
| function ajax() {  var req= new XMLHttpRequest();  req.addEventListener(“load”, function(){  var jsonobj= JSON.parse(this.responseText);  }  req.open(“GET”, “http://localhost:8080/study”);  req.send();  } |

위의 예제를 보면 Ajax를 통해 서버에 보낸 요청은 JSON 타입의 문자열 형태로 응답이 온다. 서버로부터의 응답이 load 이벤트와 함께 응답이 오면 콜백함수에서 this.responseText를 통해 응답 문자열을 가져와 JSON 함수를 통해 JSON 객체로 파싱할 수 있다.

실제 자바스크립트에서 콜백함수를 비동기로 수행하는 전체적인 로직은 아래와 같다.



지금 배운 Ajax 통신은 보안 문제로 인해 서로 다른 도메인 간 통신에 사용하기는 어려운 부분이 많다(예를 들어, 네이버 특정 앱에서 네이버의 검색 결과 데이터를 요청) 이를 해결하기 위해 요청 헤더에 특정 데이터가 있는 경우에만 응답을 해준다거나 하는 CORS 라는 표준이 생기게 된다. 또한 보안 문제를 해결하기 위핸 JSONP 기술을 제공하기도 한다.