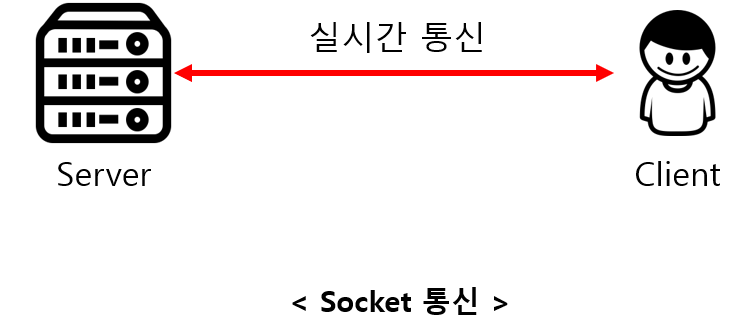
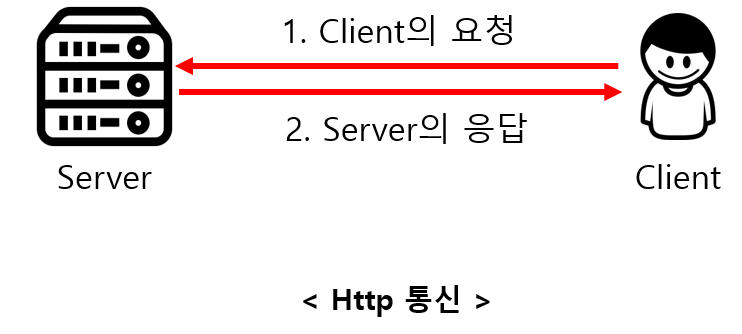
7월 4일 기술면접 문제

1. http통신과 소켓통신은 어떻게 다른가.

네트워크를 통해 서버로부터 데이터를 가져오기 위한 통신 방식은 크게 Http 통신과 Socket 통신 2가지가 있다.

Http통신은 Client의 요청이 있을 때만 Server가 응답하여 처리를 한 후에 연결을 끊는 방식이다. 이러한 연결 방식은 Client가 요청을 보내는 경우에만 Server가 응답하는 단방향적 통신으로, Server가 Client로 요청을 보낼 수는 없다. 그렇기 때문에 요청을 보낼 때, 내용을 기다리는 시간과 함께 연결하는 시간이 들어가게 된다. 이러한 Http 통신은 실시간 연결이 아닌, 필요한 경우에만 Server로 접근하는 콘텐츠 위주의 데이터를 사용할 때 용이하다. 일반적으로 모바일 어플리케이션은 필요한 경우에만 Server로 정보를 요청하는 경우가 많은데, 이러한 Web Server로 Http 통신을 주로 사용하며 비용 및 유지보수 등 대부분의 방면에서 좋다.

Socket통신은 Http 통신과 달리 Server와 Client가 특정 Port를 통해 연결을 성립하고 있어 실시간으로 양방향 통신을 하는 방식이다. Client만 필요한 경우에 요청을 보내는 Http 통신과 달리 Socket 통신은 Server 역시 Client로 요청을 보낼 수 있으며, 계속 연결을 유지하는 연결지향형 통신이기 때문에 실시간 통신이 필요한 경우에 자주 사용됩니다. 예를 들면, 실시간 Streaming 중계나 실시간 채팅과 같이 즉각적으로 정보를 주고받는 경우에 사용합니다. 예를 들어 실시간 동영상 Streaming 서비스를 Http 통신으로 구현하였다고 가정하겠습니다. 이러한 경우에 사용자가 서버로 동영상을 요청하기 위해서는 동영상이 종료되는 순간까지 계속해서 Http 통신을 보내야 하고 이러한 구조는 계속 연결을 요청하기 때문에 부하가 걸리게 됩니다. 그러므로 이러한 경우에는 Socket을 통해 구현하는 것이 적합합니다.



출처: <https://mangkyu.tistory.com/48>

2. 컴포넌트의 각 생명주기는 어떻게 되고 각 시점은 언제 호출되는가?

컴포넌트의 각 생명주기(Life Cycle)은 총 세가지 카테고리로 나눌 수 있다.

1. 마운트

DOM이 생성되고 브라우저상에 나타나는 것을 마운트라고 한다.

2. 업데이트

업데이트가 발생하는 경우는 다음과 같다.

- props가 바뀌는 경우

- state가 바뀌는 경우

- 부모 컴포넌트가 리렌더링 되는 경우

- this.forceUpdate로 강제 렌더링을 트리거하는 경우

3. 언마운트

DOM에서 컴포넌트를 제거하는 것을 언마운트라고 한다.

출처: <https://yzzzzun.tistory.com/32>

3. 자바스크립트에서 사용하는 자료구조에는 어떤 것들이 있는가?

일반적으로 자료구조를 구분할 때 선형과 비선형 구조로 구분한다. 각각 선의 형태를 띄는 것과 그렇지 않은 것으로 풀어 말할 수 있으며, 선형 구조에는 배열, 리스트, 큐, 스택이 있고 비선형 구조에는 그래프와 트리가 있다.

1. 배열 (Array)

특징:

1) 여러 데이터를 그룹핑해서 관리하기 위한 자료구조

2) 배열안에 여러 정보를 담을 수 있고, 반복문과 결합하면 정보 효율적 관리 가능

3) 크기가 정해져 있음

4) 인덱스로 식별 및 조회 가능

2. 리스트 (List)

특징:

1) 빈틈없는 데이터의 적재

2) 순서가 있는 데이터의 모임

3) 리스트의 인덱스는 몇 번째 데이터인가 정도만 의미함

4) 순차성을 보장하지 않음.

종류:

리스트에는 Array List와 Linked List가 존재한다. 둘의 큰 차이는 '데이터 연결구조'라 할 수 있는데, 그중 Array List는 배열을 이용해 Javascript의 배열 구조처럼 리스트를 구현한 것을 말한다. 두 List 구조는 장단점을 가지고 있는데, 일반적으로 고정된 데이터의 검색이 필요하면 Array List를 사용하며, 검색이 필요 없는 가변적인 데이터가 필요한 경우에는 Linked List를 사용한다.

1) Array List

-내부적으로 배열을 사용하기 때문에 인덱스를 이용해 데이터 접근 가능

- 데이터 조회 속도 증가(데이터를 추가 하거나 삭제하게 되면 각 순서가 일정하게 변경되어야 하기 때문에 데이터의 삽입과 삭제에 상대적으로 오랜 시간 소요)

2) Linked List

- 배열을 사용하지 않고, 하나의 데이터에 다음 엘리먼트의 위치정보를 포함

- 특정 데이터를 조회하는 인덱스가 존재하지 않기 때문에 조회하는 속도가 느림

- 데이터 추가 or 삭제 시 다른 데이터에 영향을 주지 않기 때문에 상대적으로 빠름

3. 큐 (Queue)

특징:

Queue는 '줄', '줄을 서서 기다리다'라는 뜻을 가지고 있다. 은행창구에서 번호표를 뽑아 기다리기', '신호를 기다리는 차들' 등의 일상생활에서의 예시가 자주 비유된다. 이 또한 일종의 큐 구조에 해당된다고 할 수 있다. 이러한 큐 구조에 공통적으로 적용되는 특징이 있는데, FIFO & LILO, 즉 먼저 들어간 사람이 먼저 나오고 나중에 들어간 사람이 나중에 나온다.

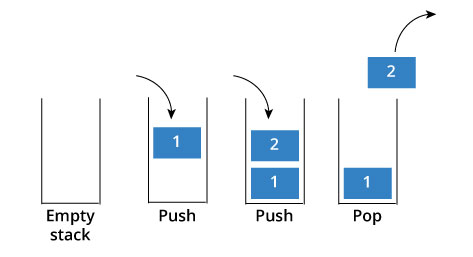
이러한 큐 구조는 컴퓨터 과학 전반에 자주 쓰이는 자료구조이다. 가장 대표적인 예로 '버퍼(Buffer)'를 들 수 있다.

일반적인 큐는 선형이지만 크기가 제한되어 있고 빈 공간을 사용하려면 모든 자료를 꺼내거나 자료를 한 칸 씩 옮겨야 한다는 단점이 있기 때문에 \*원형 큐(혹은 ‘순환 큐’라고도 한다)를 구현해 선형 큐의 문제점을 보완할 수도 있다.

\*원형 큐(Circular Queue): 원형 큐는 선형 큐의 문제점을 보완하기 위한 자료구조이다. 큐를 직선 형태로 놓는 것이 아닌 원형으로 생각해서 큐를 구현하는 것을 말한다.

4. 스택 (Stack)

흔히 스택을 쌓는다고 이야기하는 것처럼 스택은 하나의 바구니에 데이터들이 순차적으로 담겨져 있는 형태를 가진다. LIFO (Last In First Out)으로 데이터가 쌓이게 된다. ex) 웹 브라우저의 '앞으로 가기' '뒤로 가기'



\*pop(): 스택에서 가장 위에 있는 항목을 제거

\*Push(item): item 하나를 스택의 가장 윗부분에 추가

\*peek(): 스택의 가장 위에 있는 항목을 반환

\*isEmpty(): 스택이 비어 있을 때에 true를 반환

5. 그래프 (Graph)

그래프는 단순히 노드와 그 노드를 연결하는 엣지를 하나로 모아 놓은 자료구조를 말한다. 즉, 연결되어 있는 객체 간의 관계를 표현할 수 있는 자료구조로 대표적인 예는 지도, 지하철 노선도 최단경로, 전기 회로, 도로 등이 있다. 또한 그래프는 여러 개의 고립된 부분 그래프로 구성될 수 있다.

특징:

1) 방향성:

- 방향그래프 (Directed) : 간선을 통해 양 방향으로 갈 수 있다.ex) 양방향 도로

- 무방향 그래프(Undirected) : 간선에 방향성이 존재 ex) 일반 통행

2) 사이클:

- 사이클(Cycle)가능 : 단순 경로의 시작 정점과 종료 정점이 동일

- 자체 \*간선(self-loop)가능  
(\*간선(edge): 노드를 연결하는 선 (link, branch 라고도 부름))

- 순환 그래프(Cyclic)

- 비순환 그래프(Acyclic): 사이클이 없는 그래프

3) \*루트노드 개념 없음  
(\*루트 노드(root node): 부모가 없는 노드, 트리는 하나의 루트 노드만을 가짐)

4) 부모-자식의 개념 없음

5) 네트워크 모델

6. 트리 (Tree)

1) 방향 그래프(Directed Graph)

2) 비순환 그래프(Acyclic Graph)

3) 계층모델

4) 사이클불가능

5) 자체 간선 (self-loop) 불가능

6) 한 개의 루트 노드만이 존재 (모든 자식 노드는 한 개의 부모 노드 만을 가짐)

7) 부모-자식 관계 (카테고리 생각하면 좋음)

Ex) 카테고리 구분, 댓글, 이진트리, 이진탐색트리 등

출처:

<https://velog.io/@rockjeon/>자료구조

<https://lktprogrammer.tistory.com/59>