## **프론트엔드 신입 기술 면접 문제 은행 TOP20**

**(feat. 전 카카오 엔터프라이즈 개발자)**

### **1. 자신의 기술사용에 있어서 강점과 약점은 무엇인가요?**

예시답변:

저의 강점은 새로운 기술에 대한 열린 마음과 적극적인 학습 태도입니다. 또한, 문제 해결 능력과 논리적 사고 능력도 강점 중 하나입니다. 반면에 제 약점은 디테일한 부분에 대한 주의 부족입니다. 이를 보완하기 위해 더욱 신중하게 작업하고, 팀원들과 함께 코드 리뷰를 진행하는 등의 방법을 사용한다면 보완할 수 있을 것이라고 생각합니다!

**-> 자신의 약점에 대한 보완을 얼마나 성실하고 지속적으로 하고 있는지에 대한 내용이 중요합니다.**

### **2. 자신의 프로젝트에 어떤 프레임워크나 라이브러리를 사용하여 개발했나요? 왜 그것을 선택했나요?**

예시답변:

저는 React를 선호합니다. 이유는 React가 사용자 인터페이스 구축과 관리를 쉽게 할 수 있는 사용자 측 라이브러리이기 때문입니다. 또한, 리액트 컴포넌트는 모듈화를 통해서 개발 생산성을 높이고, 코드 유지 보수성을 높일 수 있습니다.  
  
**-> 여기서 React를 통해서 어떤 장점을 활용할 수 있는지 조금 더 준비하면 좋습니다.**

### **3. RESTful API에 대해 설명해주세요.**

예시답변:

RESTful API는 HTTP 프로토콜을 기반으로하는 웹 서비스 아키텍처입니다. 자원, 메소드, 메시지 등을 정의하여 클라이언트-서버 간의 통신을 가능하게 합니다. 또한, RESTful API는 표준 HTTP 메소드(GET, POST, PUT, DELETE)를 사용하여 서버와 통신합니다  
  
**-> 이 질문을 대답하면서, HTTP 메소드에 대한 사용 방법에 대해서도 꼬리물기 질문이 들어올 수 있으니 준비하시는 것이 중요합니다.**

### **4. 그렇다면 HTTP 메소드에 대한 설명도 해주실 수 있나요?**

예시답변:

**GET**: 서버에서 **리소스(데이터)를 요청**하는 메소드입니다. 요청한 데이터를 가져와 응답합니다.  
**POST**: 서버에 **데이터를 전송**하는 메소드입니다. 데이터를 전송하여 서버에서 처리하고, 처리 결과를 응답합니다.  
**PUT**: 서버에 **데이터를 업데이트**하는 메소드입니다. 요청한 데이터를 서버에 저장하고, 처리 결과를 응답합니다.  
**DELETE**: 서버에서 **데이터를 삭제**하는 메소드입니다. 요청한 데이터를 서버에서 삭제하고, 처리 결과를 응답합니다.  
  
**-> 간략하게 기능 별 포인트를 함축적으로 이야기하는 것이 중요합니다.**

### **5. CSRF나 XSS 공격을 막는 방법은?**

예시답변:

CSRF 공격을 막기 위해서는 서버에서 CSRF Token을 생성하여 세션에 저장하고, 프론트엔드에서 요청 시 해당 Token을 함께 전송하여 인증합니다. SameSite 속성을 쿠키에 설정하여 도메인이 다른 사이트에서는 쿠키를 사용할 수 없도록 제한하는 방법도 있습니다.

XSS 공격을 막기 위해서는 입력 값들을 유효성 검증하고, 특수문자들을 제외하는 정규식을 통해서 제거합니다.

또, 서버에서 CSP(Content-Security-Policy)정책을 설정하여, 허용된 스크립트만 실행되도록 제한 할 수도 있습니다. 마지막으로, HTTP 대신에 신뢰할 수 있는 HTTPS를 사용하여 통신 프로토콜을 암호화할 수 있습니다.  
  
**-> 아는 부분에 대해서만 말씀하시고, 부풀려서 말하지 않도록 주의하세요.**

### **6. async / await에 대해 설명해보세요.**

예시답변:

async/await는 비동기적인 작업을 처리할 수 있는 ES2017 문법 입니다.

async 함수를 정의하면 함수 내부에서 await 키워드를 이용하여 비동기적으로 처리되는 작업이 완료될 때까지 기다린 후, 결과값을 반환하는 처리를 할 수 있습니다.

async/await는 Promise를 기반으로 하며, 코드를 보다 간결하고 직관적으로 작성할 수 있도록 해줍니다. async 함수는 항상 Promise 객체를 반환하며, await 키워드를 이용하여 비동기 처리 결과를 기다립니다.  
  
**-> 비동기 함수이며, 비동기가 어떤 뜻인지도 설명할 준비가 되어 있으셔야 합니다.**

### **7. 브라우저 렌더링 원리에 대해서 설명해보세요.**

예시답변:

브라우저 렌더링은 HTML, CSS, JavaScript 등의 웹 페이지 자원을 브라우저가 화면에 그리는 과정을 말합니다. 브라우저 렌더링 원리와 순서는 크게 다음과 같은 단계로 구성됩니다.  
먼저 DOM을 생성합니다. 브라우저는 HTML 문서를 파싱하여 DOM 트리를 생성합니다. 이때, HTML 태그를 노드로 변환하고, 노드간의 계층 관계를 형성합니다.  
두 번째로 CSSOM을 생성합니다. 브라우저는 CSS 파일을 파싱하여 CSSOM 트리를 생성합니다. 이때, CSS 속성을 노드로 변환하고, 노드간의 계층 관계를 형성합니다.  
세 번째로 DOM트리와 CSSOM을 결합하여 렌더 트리를 생성 합니다. 이때, 실제 화면에 표시될 요소만을 선택하여 렌더 트리를 형성합니다.  
이제, 브라우저는 렌더 트리를 이용하여 각 요소의 크기와 위치를 계산하는 과정인 레이아웃을 거쳐 화면에 요소를 그리는 페인팅 과정을 거치게 됩니다. 이때, 요소의 배경, 테두리, 글자 등을 그리게 됩니다.  
  
**-> 전체적인 과정을 머릿 속에 그려놓고, 꼭 위와 같이 설명하지 않아도 흐름을 대략적으로 설명하실 수 있다면 대답하셔도 충분할 것이라 생각 됩니다.**

**(참고:**

CSS 객체 모델 (CSS Object Model, CSSOM)

CSS 객체 모델 (CSS Object Model, CSSOM)은 문서의 스타일 관련 (CSS) 정보를 읽고 수정하기 위한 API 세트입니다. 즉, DOM을 사용해 JavaScript에서 문서의 구조와 내용을 읽고 수정할 수 있는 방식과 비슷하게, CSSOM을 사용하여 JavaScript에서 문서의 스타일을 읽고 수정할 수 있습니다.

### **8. 가장 최근에 해결한 기술적인 문제는 무엇이었나요?**

예시답변:

최근에는 프로젝트에서 발생한 성능 이슈를 해결하는 문제를 해결했습니다. 문제 해결을 위해 코드를 리팩토링하고, 불필요한 부분들을 최적화하여 크롬 LightHouse를 통해 높은 점수를 받을 수 있었습니다.  
  
**-> 본인이 프로젝트에서 직접적으로 개선한 부분들을 말씀하시면 됩니다. 평소에 리팩토링 습관이 있으시다면 대답하시기 수월하실 거라 생각됩니다.**

### **9. 해당 프로젝트에서 자신의 역할은 무엇이었나요?**

프로젝트 초기에는 요구사항 분석과 설계 단계에 참여해서 와이어프레임을 팀원들과 함께 제작 하였습니다. 이후 개발 단계에서는 주로 지도 카테고리를 맡았으며, 프로젝트 종료 후에는 유지 보수와 최적화를 진행 하였습니다.  
  
**-> 본인이 협업 프로젝트를 한 경험이 무조건 있으면 대답할 수 있는 좋은 상황 입니다. 회사는 협업하는 공간이기 때문에 매우 중요한 부분이라고 할 수있습니다.**

### **10. 호이스팅에 대해 설명해보세요.**

호이스팅은 변수와 함수 선언이 스코프(scope)의 최상단으로 올려져 실행되는 것을 의미합니다. 이로 인해 선언 전에 사용하는 경우 오류가 발생할 수 있습니다.  
  
**-> 변수 선언의 3단계인 선언, 초기화, 할당 순서에 대해서 머릿 속에서 그리고 스스로 대답 해보세요.**

### **11. "==" 와 "==="의 차이는 무엇인가요?**

예시답변:

"=="는 동등 연산자로, 두 값을 비교할 때 형변환(type coercion)을 수행합니다. 즉, 비교하는 값의 데이터 타입이 다르더라도 자동으로 형변환을 수행한 후에 비교합니다. 이러한 형변환은 때로 예측하지 못한 결과를 초래할 수 있으므로, "=="를 사용할 때는 조심해야 합니다. "==="는 일치 연산자로, 두 값이 데이터 타입과 값이 모두 같은지 비교합니다. 따라서 "==="를 사용하면 형변환 없이 정확한 값을 비교할 수 있습니다. 이러한 일치 연산자를 사용하는 것이 더 안전하고 예측 가능한 결과를 얻을 수 있습니다.  
  
**-> 헷갈리지 않게 핵심만 파악해보세요. 동등! 일치!**

### **12. 클로저란 무엇인가요?**

예시답변:

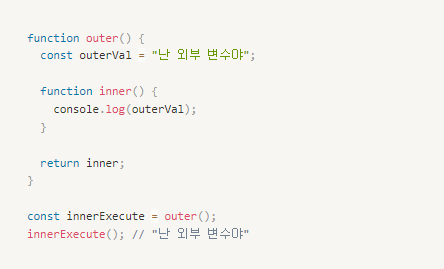
함수가 선언될 때 자동으로 생성되는 렉시컬 환경에 대한 설명입니다.

이러한 렉시컬 환경은 스코프 체인(scope chain)을 형성하게 되는데, 스코프 체인은 함수가 선언될 때의 모든 변수와 함수를 포함하는 렉시컬 스코프(lexical scope)를 형성합니다.

외부 함수가 실행 되고 반환된 후에도 외부 함수의 범위 내의 함수에 체이닝을 할 수 있는 함수 입니다.

정보를 은닉하기 위해서 주로 사용 합니다.

**-> 매우 중요한 개념이므로 꼭 이해하고 면접에 임하세요!**

****

**-> outer는 inner를 return 했습니다.innerExecute에는 outer에서 반환한 inner함수가 할당되게 됩니다. innerExecute를 호출할 때마다, outer에서 정의한 outerVal을 출력하게 됩니다. 이때 inner 함수는 outer 함수의 스코프 체인을 유지하게 되는데, 이러한 특성을 이용하여 클로저를 구현한 것입니다.**

**참고)**

**렉시컬(스코프) 환경은 함수를 "어디서 호출" 했는지가 아닌 "어디서 정의" 했는지에 따라 상위 스코프가 결정되는 것 입니다!**

### **13. 클래스형 컴포넌트와 함수형 컴포넌트의 차이는 무엇일까요?**

예시답변:

리액트에서 컴포넌트를 만드는 방법에는 클래스형 컴포넌트(class components)와 함수형 컴포넌트(functional components)가 있습니다. 클래스형 컴포넌트는 ES6의 클래스(class)를 사용하여 컴포넌트를 정의합니다. 이 방식은 React의 컴포넌트의 상태(state)와 생명주기(lifecycle)를 다룰 때 매우 유용합니다.  
반면에 함수형 컴포넌트는 ES6의 화살표 함수(arrow function)를 주로 사용하여 컴포넌트를 정의합니다. 이 방식은 훅(hook) API와 함께 많이 사용되며, 컴포넌트의 상태와 생명주기를 다룰 때도 유용합니다.

클래스형 컴포넌트와 함수형 컴포넌트의 가장 큰 차이점은 상태와 생명주기의 다루는 방식입니다. 클래스형 컴포넌트는 상태를 this.state로 정의하고, 생명주기 메서드를 오버라이드하여 다양한 작업을 수행합니다.

반면에 함수형 컴포넌트는 상태를 useState 훅을 사용하여 정의하고, useEffect 훅을 사용하여 생명주기를 다룹니다.

또한, 클래스형 컴포넌트에서는 this 키워드를 사용하여 상태나 메서드를 참조하며, 함수형 컴포넌트에서는 this 키워드를 사용하지 않습니다.

이러한 차이점 때문에, 함수형 컴포넌트는 더 간결하고 가독성이 좋아지며, 테스트와 리팩토링이 쉽게 이루어질 수 있습니다.  
  
**-> 차이점에 대해 인지하는 부분이 중요합니다.**

### 

### **14. 이벤트 버블링에 대해서 설명해주세요.**

예시답변:

이벤트 버블링(event bubbling)은 HTML에서 이벤트가 발생했을 때, 해당 요소에서 이벤트가 처리된 후, 상위 요소로 이벤트가 전파되는 현상을 말합니다. 이벤트 버블링은 이벤트가 전파되는 동안 부모 요소에서도 이벤트를 감지할 수 있다는 장점이 있습니다.

즉, 이벤트를 처리하는 핸들러 함수를 부모 요소에 등록해놓으면, 자식 요소에서 발생한 이벤트도 모두 처리할 수 있습니다.

이를 활용하면, 여러 개의 하위 요소에서 발생하는 이벤트를 하나의 이벤트 핸들러로 바인딩하는 이벤트 위임 처리를 할 수 있습니다.

버블링 및 캡쳐링을 방지하기 위해서는 이벤트 객체의 stopPropagation() 메서드를 사용하여 이벤트 전파를 중지시키거나, 이벤트 핸들러에서 이벤트가 발생한 요소를 확인하여 처리해야 합니다.  
  
**-> 이벤트 버블링과 캡쳐링의 차이점에 대해서 구분해서 인지하는 것이 중요합니다.**

**->이벤트 위임에 대해서도 인지하고 있어야 합니다.**

**참고)**

**이벤트 위임이란?**

**이벤트 버블링을 활용하여 상위 요소에만 이벤트 리스너를 등록하는 기법입니다.**

**이벤트 위임을 적용하면 메모리 사용량이 감소하며 코드를 더 간결하게 구현할 수 있습니다.**

### 

### **15. 스토리북에 대해서 아시나요?**

예시답변:

스토리북(Storybook)은 UI 컴포넌트 개발을 위한 도구로, 개별 컴포넌트의 기능과 상태를 다양한 시나리오로 테스트하고, 문서화하여 관리할 수 있는 환경을 제공합니다. 이를 통해 UI 개발자와 디자이너는 컴포넌트의 독립적인 테스트와 문서화를 통해 작업 효율성과 코드 품질을 향상시킬 수 있습니다.  
  
**-> 최근 트렌드를 넘어 스테디하게 사용되고 있는 부분들도 파악 해보세요.**

### **16. TDD란 무엇인가요?**

예시답변:

TDD(Test-Driven Development)는 테스트 주도 개발 방법론의 준말로, 개발자가 코드를 작성하기 전에 먼저 테스트 케이스를 작성하고 이를 통과시키는 것을 중심으로 개발을 진행하는 방법입니다. TDD는 코드 품질을 향상시키고 버그를 미리 발견하여 개선하는 데 도움이 됩니다. 또한, 코드 변경 시 이전에 작성된 테스트 케이스를 수행하여 기존 코드의 영향을 미치는 부분을 파악하고, 안정적인 코드 개발에 도움을 줍니다.  
  
**-> TDD 도구인 Jest, Cypress, testing-library 등에 대한 학습도 해보세요.**

### **17. SSG**(Static Site Generator)**에 대해서 아시나요?**

예시답변:

Static Site Generator(SSG)는 정적인 HTML, CSS, JavaScript 파일을 생성하는 소프트웨어입니다. SSG를 사용하면 동적인 서버 측 프로그래밍 없이 정적인 웹 페이지를 생성할 수 있습니다.

이에 대한 몇 가지 장점은 다음과 같습니다. 정적인 파일을 생성하기 때문에, 웹 서버의 보안에 대한 걱정이 줄어듭니다. 정적인 파일은 웹 브라우저에 의해 더 빠르게 다운로드됩니다.

따라서 로딩 속도가 향상되며 사용자 경험이 좋아집니다.

GatsbyJS, Next.js, Hugo, Jekyll 등과 같은 플랫폼들이 있습니다.  
  
**-> 각각의 플랫폼들의 특징들에 대해서도 간략히 파악하면 도움이 될 것입니다.**

### 

### **18. Flex와 Grid의 차이점에 대해서 설명해보세요.**

예시답변:

Flex는 주로 단일 축 방향의 레이아웃을 구성하는 데 사용되며, 주로 수평 방향으로 레이아웃을 정렬합니다. Grid는 2차원 그리드 시스템을 구성하는 데 사용되며, 수평과 수직 방향으로 모두 레이아웃을 정렬할 수 있습니다. Flex는 아이템들의 크기가 자유로워 유동적으로 변할 수 있습니다. 반면에 Grid는 아이템들의 크기를 미리 정의하여 각 셀의 크기를 일정하게 유지합니다. Flex는 유동적인 레이아웃 변화를 구성하기 적합합니다. 반면에 Grid는 미리 정의된 2차원 그리드 구조를 유지하기 때문에 레이아웃 변화가 적은 경우에 적합합니다.  
  
**-> 언제 Flex를 쓰고 언제 Grid를 쓸지에 대해 아는 것이 가장 중요합니다.**

### **19. MVVM패턴과 Flux패턴의 차이점에 대해서 설명해보세요.**

예시답변:

MVVM은 데이터 바인딩을 사용하여 View와 ViewModel 간에 양방향 데이터 흐름을 유지합니다.

반면 Flux는 단방향 데이터 흐름을 사용하여 Action -> Dispatcher -> Store -> View의 방향으로 데이터가 흐릅니다.

MVVM에서는 ViewModel이 Model의 역할을 수행하고, View와 ViewModel 사이에서 데이터를 중개합니다.

반면 Flux에서는 Store가 애플리케이션의 데이터와 비즈니스 로직을 담당하고, Dispatcher를 통해 Action을 처리합니다. Flux는 단방향 데이터 흐름과 단일 책임 원칙을 따르므로 테스트하기 쉬운 코드를 작성하기 쉽습니다. MVVM에서는 ViewModel과 View 간의 양방향 데이터 바인딩 때문에 테스트하기가 어려울 수 있습니다.

MVVM은 Angular, Vue.js 등의 프레임워크에서 사용되며,

Flux는 React에서 주로 사용됩니다.  
  
**-> MVVM과 Flux가 각각 어떤 프레임워크에서 사용되는 패턴인지 아는 것이 중요합니다.**

### 

### **20. 비동기 함수에 대해서 설명해 보세요.**

예시답변:

비동기 함수(Asynchronous function)는 함수의 실행 결과가 즉시 반환되지 않고, 특정 조건이 충족될 때까지 기다리는 함수입니다.

비동기 함수는 일반적으로 콜백 함수나 Promise 객체를 반환합니다.

콜백 함수는 비동기 작업이 완료되었을 때 호출되는 함수이며,

Promise 객체는 비동기 작업이 성공적으로 완료되었는지 또는 실패했는지를 나타내는 객체입니다.

비동기 함수는 애플리케이션의 성능을 향상시키고, 사용자 인터페이스의 반응성을 유지하는 데 유용합니다. 그러나 비동기 함수를 사용할 때는 콜백 지옥과 같은 문제를 피하기 위해 적절한 에러 처리와 코드 구성이 필요합니다. 여러개의 비동기 함수가 실행되면, 이벤트 루프는 비동기 함수 호출을 처리하고, 비동기 함수가 완료되었을 때 콜백 함수를 호출합니다. 이벤트 루프는 실행 대기 중인 비동기 함수가 있으면 해당 함수를 호출하고, 실행이 완료될 때까지 다음 비동기 함수를 호출합니다. 따라서, 여러 개의 비동기 함수가 실행될 때 이들 함수는 동시에 실행되며, 이벤트 루프에 의해 비동기 함수의 실행 순서가 제어됩니다.  
  
**-> 가장 중요한 질문이라고 생각되네요. 이벤트 루프에 대해서도 꼭 숙지하셔서 좋은 결과 있기를 바랍니다.**