Flutter 면접 질문)

1. **플러터의 빌드시 모드들의 차이는 무엇인가요.**

Flutter 도구는 앱을 컴파일 하기 위한 3가지 모드를 제공하고, 테스트를 위한 headless 모드를 제공합니다. 개발 진행 상황에서 어느 정도에 위치해 있느냐에 따라 컴파일 모드를 선택합니다. 코드를 디버깅 중인가요? 프로파일링 정보가 필요한가요? 앱을 배포할 준비가 됐나요?

언제 어떤 모드를 사용해야하는 지 간략하게 요약해봅시다:

* 개발하면서 [hot reload](https://flutter-ko.dev/docs/development/tools/hot-reload)를 사용하고 싶다면, [디버그](https://flutter-ko.dev/docs/testing/build-modes#%EB%94%94%EB%B2%84%EA%B7%B8) 모드를 사용하세요.
* 성능을 분석하고 싶다면, [프로파일](https://flutter-ko.dev/docs/testing/build-modes#%ED%94%84%EB%A1%9C%ED%8C%8C%EC%9D%BC) 모드를 사용하세요.
* 앱을 출시할 준비가 됐다면, [릴리즈](https://flutter-ko.dev/docs/testing/build-modes#%EB%A6%B4%EB%A6%AC%EC%A6%88) 모드를 사용하세요

디버그

*디버그 모드* 에서는 물리적 기기 혹은 에뮬레이터, 시뮬레이터에서 앱이 디버깅 할 수 있는 상태로 준비가 됩니다.

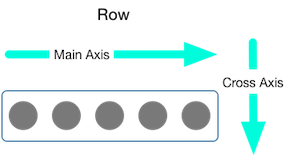
릴리즈

최대한 최적화하고 앱의 사이즈를 가장 작게 하기를 원한다면, 앱을 배포할 때 *릴리즈 모드* 를 사용하세요. 모바일의 경우 릴리즈 모드는 (시뮬레이터와 에뮬레이터에서 지원되지 않습니다)

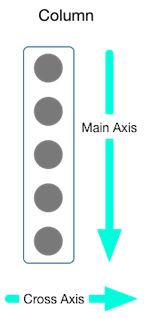
프로파일

*프로파일 모드* 에서는 앱의 성능을 측정 할 만한 일부 디버깅 기능이 유지됩니다. 에뮬레이터나 시뮬레이터는 실제 성능을 표현할 수 없기 때문에 프로파일 모드를 사용할 수 없습니다. 모바일에서 프로파일 모드는 아래 차이를 제외하면 릴리즈 모드와 유사합니다:

**Q) mainAxisAlignment 와 crossAxisAlignment는 언제 사용하나요?**



행(Row) 위젯 사용시, 주 축(mainAxis)는 수평이며, 횡축(crossAxis)는 세로이다.



 열(Column)위젯 사용시, 주축(mainAxis)는 수직이며, 횡축(crossAxis)는 가로이다.

## Q) 플러터 Inspector에 대해 설명하세요

앱의 Widget Tree를 한눈에 보기 좋게 정리되어 있습니다.

Select Widget Mode 를 눌러주면 해당 위젯의 영역을 보여줍니다.

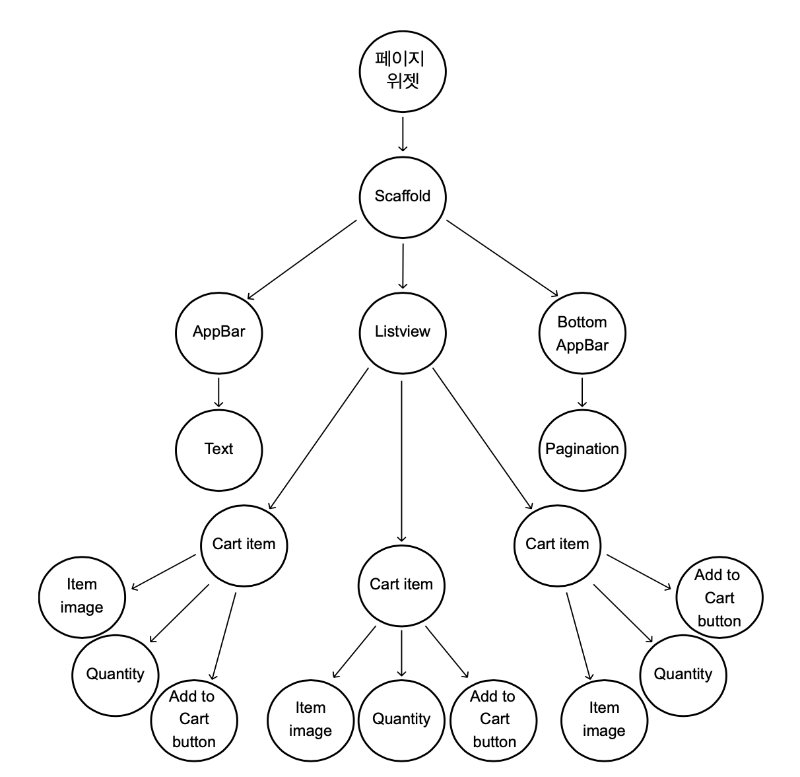
**Q) 플러터의 동작원리에 대해 설명하시오**

넓은 의미로 플러터는 웹의 리액트처럼 리액티브, 선언형declarative, 조합할 수 있는 뷰 계층 라이브러리다(하지만 플러터는 렌더링 엔진도 포함하므로 실제로는 리액트와 브라우저를 합한 것과 더 비슷하다). 즉 위젯이라는 작은 컴포넌트component를 조합해 모바일 UI를 만든다. 플러터의 모든 것은 위젯이며 위젯은 뷰를 묘사하는 다트 클래스다. 구조, 스타일, 애니메이션 그리고 그 밖에 UI를 구성하는 모든 것이 위젯이다.1

**플러터 렌더링: 내부 동작 원리**

플러터의 진정한 능력은 앱을 셀 수 없이 여러 번 리빌드할 때 발휘된다. 플러터는 눈 깜짝할 새에 거대한 위젯 트리widget tree를 빌드한다. 다음 그림은 한 페이지의 장바구니 위젯 트리가 어떻게 생겼는지를 가상으로 보여준다(현실의 트리는 이보다 더 거대하다).

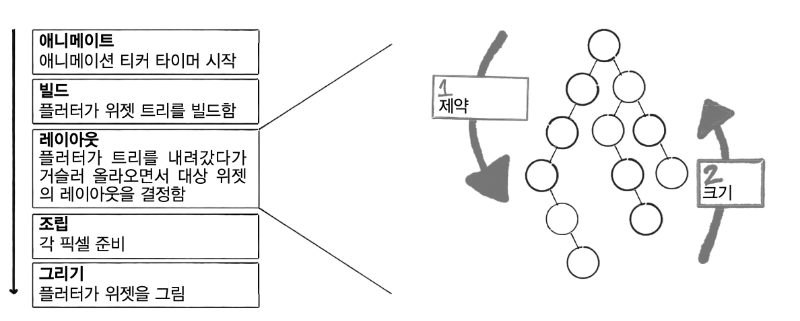
CartItem 위젯을 살펴보자. 이 위젯은 상태를 가지며 위젯의 자식은 위젯 상태에 의존한다. CartItem 위젯의 상태가 바뀌면 이 위젯을 포함한 모든 하위 위젯이 다시 그려진다.

그림\_ 위젯 트리 모습이다. 현실의 위젯 트리는 훨씬 많은 위젯을 포함한다

플러터 위젯은 리액티브다. 즉 외부(또는 setState)에서 새 정보를 얻으면 이에 반응하고, 필요하면 플러터가 위젯을 다시 그린다. 이를 간략히 설명하면 다음과 같다.

1 사용자가 버튼을 누름  
2 Button.onPressed 콜백에서 setState를 호출함  
3 Button의 상태가 dirty로 바뀌었으므로 플러터는 이 위젯을 리빌드함  
4 트리에서 기존 위젯을 새 위젯으로 바꿈  
5 플러터가 새 트리를 그림

위젯을 새로 바꿨으므로 이제 위젯을 새로 그릴 수 있다. 렌더링도 여러 과정을 거쳐 진행된다. 다음 그림은 전체적인 플러터의 렌더링 과정이며 세 번째 단계를 눈여겨보자.

전체적인 렌더링 과정

애니메이션 티커animation ticker가 동작하면서 그리기 작업을 시작한다. 예를 들어 리스트를 아래로 스크롤하는 등 위젯을 다시 그려야 하는 상황에서는 화면의 처음 요소 위치에서부터 최종 위치까지 조금씩 이동하면서 애니메이션이 부드럽게 일어난다. 이 과정은 요소가 움직여야 하는 시간을 결정하는 애니메이션 티커가 제어한다. 이렇게 애니메이션이 일어나는 동안 플러터는 프레임마다 위젯을 리빌드하고 그린다. 이 책의 뒷부분에서 플러터의 애니메이션 기능을 자세히 알아본다.

● 플러터는 모든 사람이 아름답고, 좋은 성능의 모바일 앱을 만들 수 있는 모바일 SDK이며 다트로 구현되었다.  
● 다트는 구글이 만든 언어로 자바스크립트로 컴파일할 수 있다. 다트는 빠르며, 엄격한 형식을 지원하고 배우기 쉽다.  
● 플러터는 네이티브 디바이스 코드로 컴파일되므로 다른 크로스 플랫폼 기술보다 성능이 뛰어나다. 또한 다트의 JIT, 플러터의 핫 리로드 덕분에 최상의 개발자 경험을 제공한다.  
● 플러터는 훌륭한 성능의 크로스 플랫폼 앱을 빨리 만들어야 하는 사람에게 적합하다. 하지만 두 개의 네이티브 팀을 이미 보유한 큰 회사에게는 플러터가 좋은 선택이 아닐 수 있다.  
● 플러터의 모든 것은 위젯이다. 위젯은 뷰를 묘사하는 단순한 다트 클래스다. 여러 작은 위젯을 조립해 위젯 트리를 완성하며 UI를 만든다.  
● 위젯은 크게 상태가 없는 위젯과 상태가 있는 위젯으로 분류된다.  
● 플러터는 위젯 생명주기 메서드, 특별한 State 객체 등 상태 관리도구를 제공한다.

**Q) Mixin이란?**

공식 문서에는 아래와 같이 나와있습니다.

"Mixin은 여러 클래스 계층에서 클래스 코드를 재사용하는 방법입니다."

덧붙여 Mixin을 사용하기 위해선 with 키워드를 사용해 이름을 적어주면 된다고 나와있습니다.