Jetpack Compose

Jetpack Compose 소개

Jetpack Compose 는 Android 네이티브 UI를 빌드하기 위한 현대적인 선언형 UI 툴킷으로, 뷰를 명령형으로 변형하지 않고도 앱 UI를 렌더링할 수 있게 하는 선언형 API를 제공하여 UI 개발을 간소화하고 가속화한다고 한다.

즉, 선언형 API 를 통해 Android UI 를 더 빠르고 쉽게 빌드할 수 있도록 도와주는 것이다.

선언형 프로그래밍이란?

- 사전적 의미 : 어떻게(How) UI 를 구성하는지가 아니라 무엇(What) 을 UI 에 구성하는지 나타내는 방식
- UI = f(state) 라고도 이야기 함
- React 와 같이 인터액티브한 UI 를 통해 데이터가 변결될 때 효율적으로 필요한 구성요소만 렌더링함.
- HTML 이나 기존 안드로이드 View 시스템은 선언형 UI 툴킷이라 말하기 어려움

Jetpack Compose 배경

15년 전 (2008년) 에 만들어진 레거시 UI 코드와 API 를 탈피하고자 Android 를 위한 현대적인 선언형 UI 도구 키트 Jetpack Compose 가 등장하게 된다. (2021년 7월 28일 런칭)

안드로이드 타임라인

1. 1.0 (2008년 9월 23일)
2. 1.1 Petit Four (2009년 2월 9일)
3. 1.5 Cup Cake (2009년 4월 27일)
4. 1.6 Donut (2009년 9월 15일)
5. 2.0 Eclair (2009년 10월 27일)
6. 2.2 Froyo (2010년 5월 20일)
7. 2.3 Gingerbread (2010년 12월 6일)
8. 3.0 Honeycomb (2011년 2월 22일) Holo
9. 4.0 Ice Cream Sandwich (2011년 10월 18일)
10.4.1 Jelly Bean (2012년 7월 9일)
11.4.4 Kitkat (2013년 10월 31일)
12.5.0 Lollipop (2014년 11월 4일) Material

13.6.0 Marshmallow (2015년 10월 2일)
14.7.0 Nougat (2016년 8월 22일)
15.8.0 Oreo (2017년 8월 21일)
16.9 Pie (2018년 8월 6일)
17.10 (Q) (2019년 9월 3일)
18.11 (R) (2020년 9월 8일)
19.12 (S) (2021년 10월 4일)
20. 13 Tiramisu (2022년 8월 15일)

안드로이드 레거시 View 시스템의 문제점

- UI를 업데이트하기 위해 View 객체를 가져와야함
    - findViewById()와 같은 함수로 뷰를 찾음.
- 뷰는 상태를 가지고 있어 함수로 상태를 가져옴
    - getText()같은 함수로 상태를 가져옴.
- 뷰의 상태를 업데이트하기 위해 여러 함수를 사용.
    - setText()로 업데이트
    - container.addChild()
    - img.setImageBitmap()
- 각 컴포넌트마다 상태를 가지고 있고 이를 관리하는 것이 쉽지 않음.

Jetpack Compose 장점

- 코드 감소
    - 상호작용
        - 컴포즈는 기존 앱과 상호작용 가능함.
        - View들을 컴포즈 UI 안에 넣을 수 있고 컴포즈를 뷰 안에 넣을 수 있음.
    - Jetpack 통합
        - Compose는 Jetpack과 통합되도록 설계됨.
        - Navigation, Paging, LiveData, Flow/RxJava, ViewModel, Hilt
    - Material
        - 머터리얼 디자인 컴포넌트와 테마를 제공.

        - 아름다운 앱을 만들고 브랜드를 표현.
        - 복잡한 XML을 다루지 않고 이해하기 쉽고 추적하기 쉬움.
- 직관적
    - Compose는 선언적 API를 통해 Compose가 나머지를 알아서 다 처리하므로 개발자는 UI를 설명하기만 하면 된다.
        - 테마 레이어가 기존 xml 에서는 여러 파일로 확장했어야 했는데 단일한 코틀린 파일 내에서 달성할 수 있다.
        - 직관적인 Stateless UI 구성요소를 새로운 세트로 제공하기가 쉬워졌다.
        - 추론하는 동안 기억해야 하는 사항이 줄어들고 통제를 벗어나거나 제대로 이해하지 못하는 행동도 적어졌다.
- 빠른 개발 과정
    - Compose는 기존의 모든 코드와 호환되어 Navigation, ViewModel, Kotlin 코루틴과 같은 대부분의 일반적인 라이브러리와 함께 작동한다.
        - 라이트 모드와 다크 모드 등 테마 관리에서 매우 쉬워졌다.
        - Preview 기능을 통해 UI 를 미리 볼 수 있어 오류 상태나 여러 가지 글꼴 크기 등을 바꿔가며 테스트하기 편해졌다.
- 강력한 성능
    - Compose는 Android 플랫폼 API에 직접 액세스하고 머터리얼 디자인, 다크모드, 애니메이션 등을 기본적으로 지원한다.
        - 접근성 API, 레이아웃 등 모든항목이 개선되어 만들고 싶은 것과 실제 만드는 것 사이의 차이가 줄어들었다.
        - 애니메이션을 Compose 에 쉽게 추가할 수 있으니 색상/크기/고도 변경 등 적용에 어려움이 사라졌다.

명령형 프로그래밍

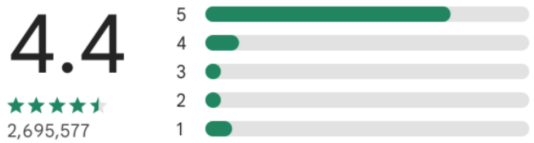
```
val layout = LinearLayout(context)
layout.setBackgroundColor(red)
layout.removeAllViews()
val childTextView = TextView(context)
childTextView.text = "child"
layout.addView(childTextView)
```

선언형 프로그래밍

```
Column(
        modifier = Modifier
                .background(red)
) {
        Text("child")
}
```

활용 사례

- **트위터**
    - Twitter 앱은 10년에 걸친 레거시
    - 확장 가능한 디자인 시스템 구축을 원함
    - 컴포즈 선택
        - 사용 및 유지관리가 용이한 stateless UI 콤포넌트
        - 확장과 커스터마이징이 편함
        - 컴포넌트 단위로 레거시 설정에 의존하지 않고 변경함
- **스퀘어**
    - 스퀘어는 POS, 결제 등을 만드는 회사
    - Retrofit, OkHttp, Dagger 1, Picasso, Moshi, Javapoet 오픈소스 라이브러리 강자
    - Square는 이미 선언형 도구 (Workflow)로 앱을 개발했었음. MVI에 가까운 Unidirectional Data Flow.
    - 하지만 UI는 Mosaic란 내부 툴을 사용했지만 Jetpack Compose로 변경.
    - Android 도구와의 통합의 장점이 있다.
- **Google Play**



With Views, this table consists of:

- 3 View classes total, with 2 requiring custom drawing for the rounded rects, and stars
- ~350 lines of Java, 55 lines of XML

With Compose, this table consists of:

- All `@Composable` functions contained in the same file and language!
- ~210 lines of Kotlin

View - 3개의 클래스. 350 라인의 Java, 55 라인의 XML

Compose - @Composable 함수로 같은 파일의 같은 언어. 210 라인의 코틀린

    - 10년 이상된 레거시 기술 부채
    - 네트워크 계층부터 픽셀 렌더링까지 다년간 마이그레이션 로드맵을 만듦.

    - 현대적이고 선언적인 대안을 찾다 알파 버전의 Compose를 도입하기로 결정.
    - 50%가 적은 코드로 UI를 작성.
    - 가장 크게 고려했던 부분
        - 개발자 생산성
            - 수백명의 엔지니어가 개발하기 편하고 재밌어야 함.
        - 성능
            - 지연시간과 버벅거림이 민감한 비즈니스
            - Android Go 디바이스 대응해야 함.

권장 사항

MVVM 등 단방향 데이터 흐름 아키텍처 권장

targetSdkVersion : 30

Gradle : 7.0

Kotlin Version : 1.5.21

minSdkVersion 21

Android Studio : Arctic Fox (4.2)