WaffleStudio Android Seminar - 5

이승민 (안드로이드 세미나장)

2021.11.06.(토) 11:30 ~

What we will learn in Seminar 5

- SharedPreferences
- Custom View
- Simple about backend
  - Postman
  - Token Authorization

SharedPreferences

- 배웠던 것
  - Room DB
  - Database table을 생성 & 관리
- 그런데…
  - 그렇게 대단한 것을 저장하려고 하는게 아닌데…
  - Database table 만드는거 귀찮은데…

SharedPreferences

- 그럴 때 필요한 SharedPreferences
- Hash Table과 같이 Key-Value 관계를 가지는 간단한 저장소

```
private val sharedPreferences by lazy {
    getSharedPreferences(BuildConfig.PREF_KEY, MODE_PRIVATE)
}
```

SharedPreferences

- 값 가져오는 법

```
count = sharedPreferences.getInt(COUNT_KEY, 0)
```

- 값 저장하는 법

```
sharedPreferences.edit {  this: SharedPreferences.Editor
    putInt(COUNT_KEY, count)
}
```

어딘가에 private const val COUNT_KEY = “count_key” 등…

Custom View

- 기본적인 View들…
  - TextView, EditTextView, Button, ImageView…
  - 보통 모으고 모아서 하나의 Activity(Fragment)를 구성
- 열심히 만든걸 다른 곳에서 쓰려면?
  - 복사 붙여넣기? 구현 코드까지?
- 구현 코드가 복잡하면?
  - Activity(Framgment) 파일 길어지는데…

Custom View

- 그럴 때 필요한 Custom View
- View들을 모은 구현체를 묶어서 저장해놓고 쓰기

```
<merge xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:orientation="horizontal"
    tools:parentTag="android.widget.LinearLayout">
```

- merge로 시작해서 parentTag로 Layout 명시

```
class RatingStarsView : LinearLayout
```

- 명시한 Layout 상속해서 구현

Custom View

- 필수적으로 만들어야할 constructor (attributeSet?)

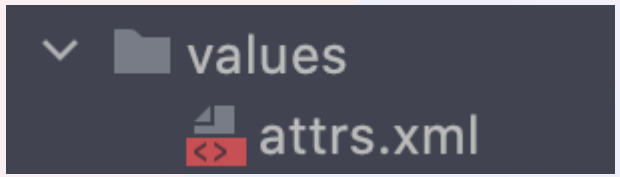
```
constructor(context: Context) : super(context) {
    init( attr: null)
}

constructor(context: Context, attributeSet: AttributeSet?) : super(context, attributeSet) {
    init(attributeSet)
}

constructor(context: Context, attributeSet: AttributeSet?, defStyle: Int) : super(
    context,
    attributeSet,
    defStyle
) {
    init(attributeSet)
}
```

Custom View

- TextView
  - android:text= ~~
  - android:textColor=~~
- 이런 요소들을 직접 만들어줄 수 있다!




```
<resources>
    <declare-styleable name="RatingStarsView">
        <attr name="starSize" format="enum">
            <enum name="small" value="0" />
            <enum name="medium" value="1" />
            <enum name="large" value="2" />
        </attr>
        <attr name="dragEnabled" format="boolean" />
    </declare-styleable>
</resources>
```

Custom View

- 코드에서 가져오기

```
context.theme.obtainStyledAttributes(
    attr,
    R.styleable.RatingStarsView,
    defStyleAttr: 0,
    defStyleRes: 0
).apply {  this: TypedArray
    try {
        dragEnabled = getBoolean(R.styleable.RatingStarsView_dragEnabled,  defValue: false)
    } finally {
        recycle()
    }
}
```

Simple about Backend - Postman

- Postman
  - 원래 request-response를 주고 받으려면 terminal을 이용
  - curl –d “~~” –H “~~:~~” –X GET http://~~~~
    - 너무 귀찮음
- 쉽게 request-response 를 주고 받고 결과를 저장하는 프로그램

Simple about Backend - Token

- 서버가 사용자를 인지하는 방식
- 사용자가 로그인 시 서버가 해석할 수 있는 Token을 발급
- 클라이언트는 request에 Token을 포함시켜서 전달
- 서버는 Token을 검증하여 사용자를 확인
- 과제에서는 JWT 토큰을 사용

QA