
Kotlin을 이용한 Android 프로그래밍

코틀린 기초

Contents

I. FireBase 개요

I. Android Studio에서 Firebase 연결

II. Kotlin - FireBase의 연동을 통한 앱 작성



kotlin-7f545

inputData








```
+ -LikXceNLG3kUOANju4-
+ -Lik_QeaqW_KNA236bK
+ -LikbWqhgm17lqMTmUxy
+ -Like3fCQpb-DYP6RiN7
+ -LilTpkPpKRK0MXWX6k-
+ -LiliFKSFmKVzDdW9SFy
- -LipfFcbSi7y6_K0lN2z
  id: "-LipfFcbSi7y6_K0lI
  name: "Kotlin Programm
  rating: 4
```

코틀린 기초



▶ 파이어베이스(Firebase) 개요

- ▶ 파이어베이스(Firebase)는 2011년 파이어베이스(Firebase, Inc) 개발하고, 구글이 2014년 인수
- ▶ 모바일 및 웹 어플리케이션 개발 플랫폼으로 안드로이드 스튜디오의 각 기능 연동 API를 제공
 - ▷ 실시간 데이터베이스, 클라우드 저장소, 호스팅, 성능 모니터링, 클라우드 메시징 등








더 멋진 앱 개발

-  **Cloud Firestore**
글로벌 규모의 앱 데이터 저장 및 동기화
-  **ML Kit ^{BETA}**
모바일 개발자를 위한 머신러닝
-  **Cloud Functions**
서버를 관리하지 않고 모바일 백엔드 코드 실행
-  **Authentication**
간편하고 안전한 사용자 인증
-  **Hosting**
빠르고 안전하게 웹 앱 애셋 전달
-  **Cloud Storage**
Google의 규모를 활용한 파일 저장 및 제공
-  **Realtime Database**
순식간에 앱 데이터 저장 및 동기화

앱 품질 향상

-  **Crashlytics**
강력한 실시간 오류 보고를 통해 문제 우선순위 지정 및 해결
-  **Performance Monitoring**
앱 성능에 대한 통계 파악
-  **Test Lab**
Google이 호스팅하는 기기에서 앱을 테스트합니다.

비즈니스 성장 도모

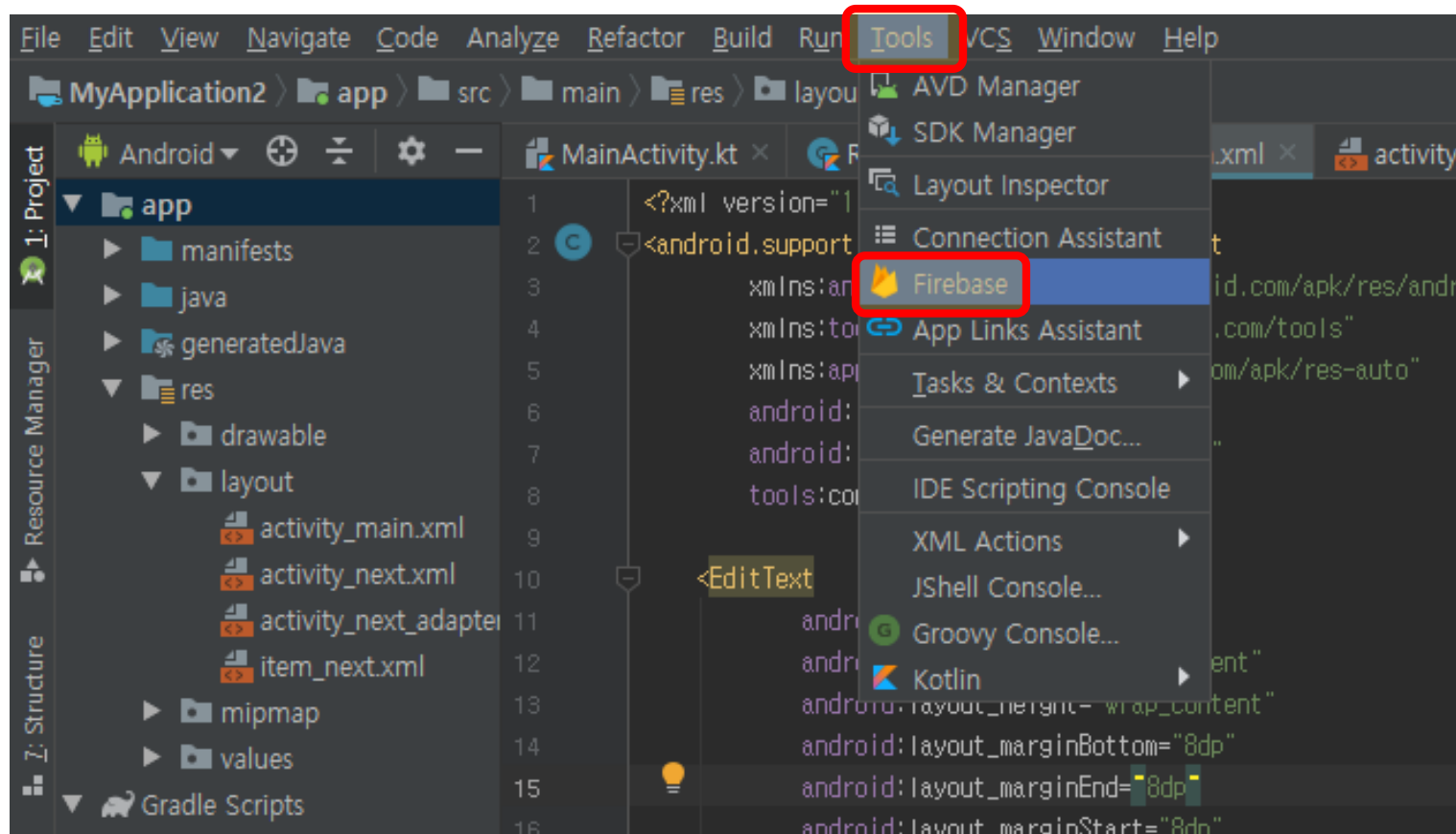
-  **In-App Messaging ^{BETA}**
상황별 메시지로 활성 앱 사용자의 참여를 유도합니다.
-  **Google Analytics**
제한 없는 무료 앱 분석 사용
-  **Predictions**
예측된 행동을 기반으로 한 스마트 사용자 세분화
-  **A/B Testing ^{BETA}**
실험을 통해 앱 환경 최적화
-  **Cloud Messaging**
타겟팅 메시지 및 알림 전송
-  **Remote Config**
새 버전을 배포하지 않고도 앱 수정
-  **Dynamic Links**
기여도가 추적되는 딥 링크로 성장 도모

코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ Tools - Firebase 클릭

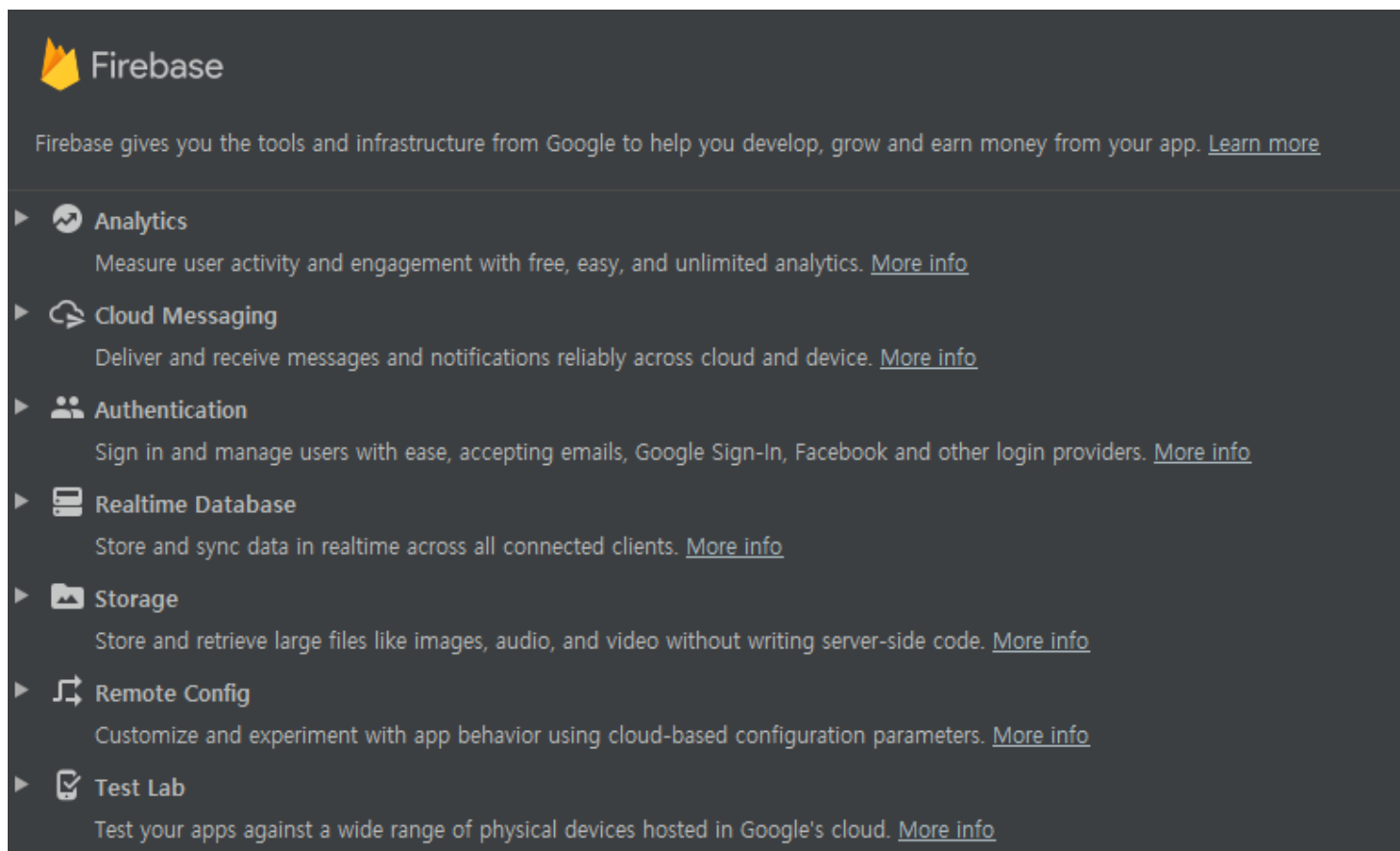


코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 클릭 시, 사진과 같은 Firebase 메뉴가 전시

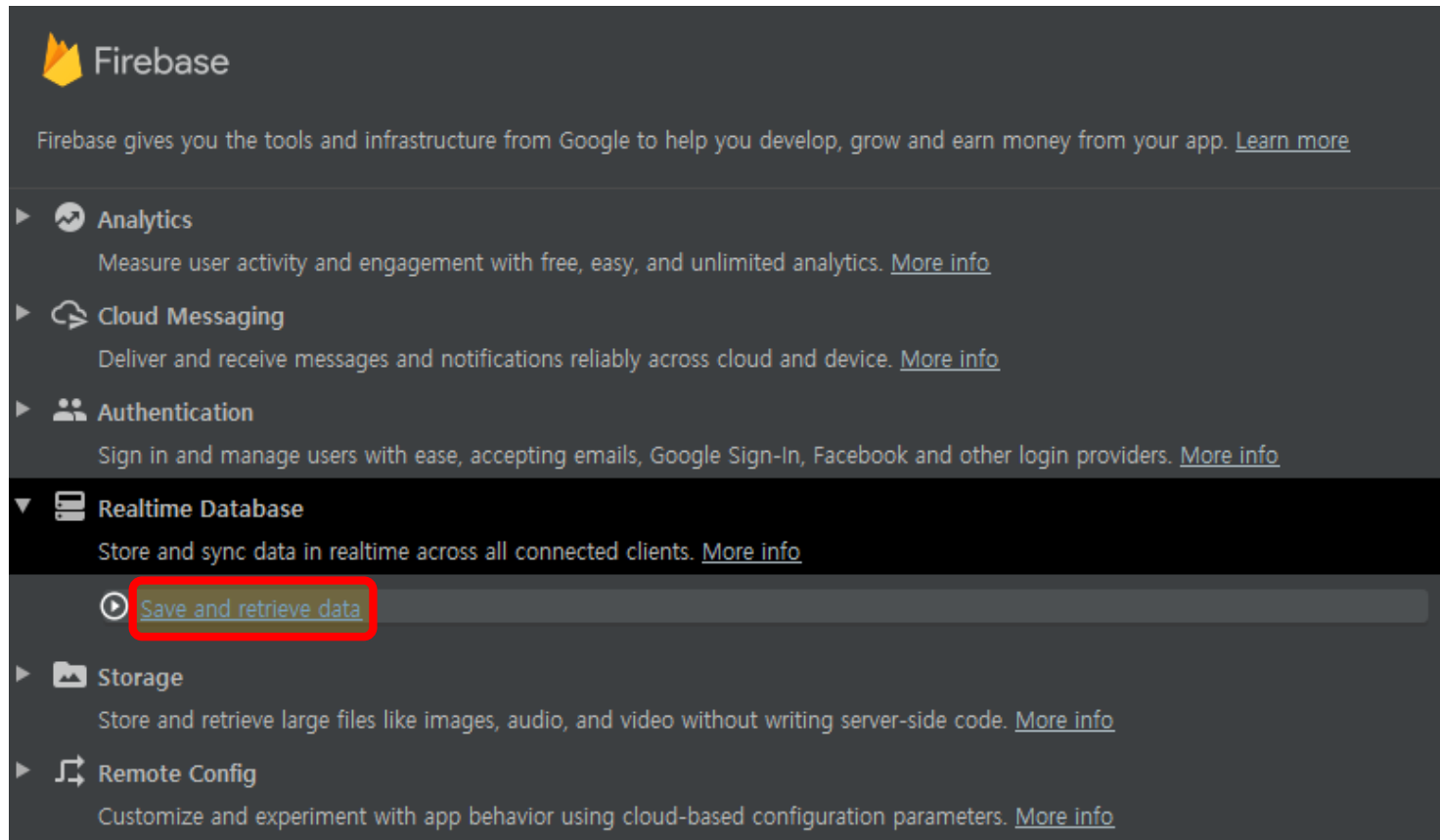


코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 우리가 사용할 것은, Realtime Database로 More Info가 아닌 Save and retrieve data 클릭



코틀린 기초

- ▶ Android Studio에서 Firebase 연결
 - ▶ 프로젝트와 Firebase 연결
 - ▶ Save and retrieve data 클릭 시 가이드 전시
 - ▶ Connect your App to Firebase 클릭

The screenshot shows the Firebase Realtime Database setup guide. The first step, 'Connect your app to Firebase', is highlighted with a red box and contains a 'Connect to Firebase' button. The subsequent steps are 'Add the Realtime Database to your app' (with an 'Add the Realtime Database to your app' button) and 'Configure Firebase Database Rules'. The final step, 'Write to your database', includes a code snippet for writing to the database.

1 Connect your app to Firebase

Connect to Firebase

2 Add the Realtime Database to your app

Add the Realtime Database to your app

3 Configure Firebase Database Rules

The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up [Authentication](#), you can [configure your rules for public access](#). This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.

4 Write to your database

Retrieve an instance of your database using `getInstance()` and reference the location you want to write to.

```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");

myRef.setValue("Hello, World!");
```

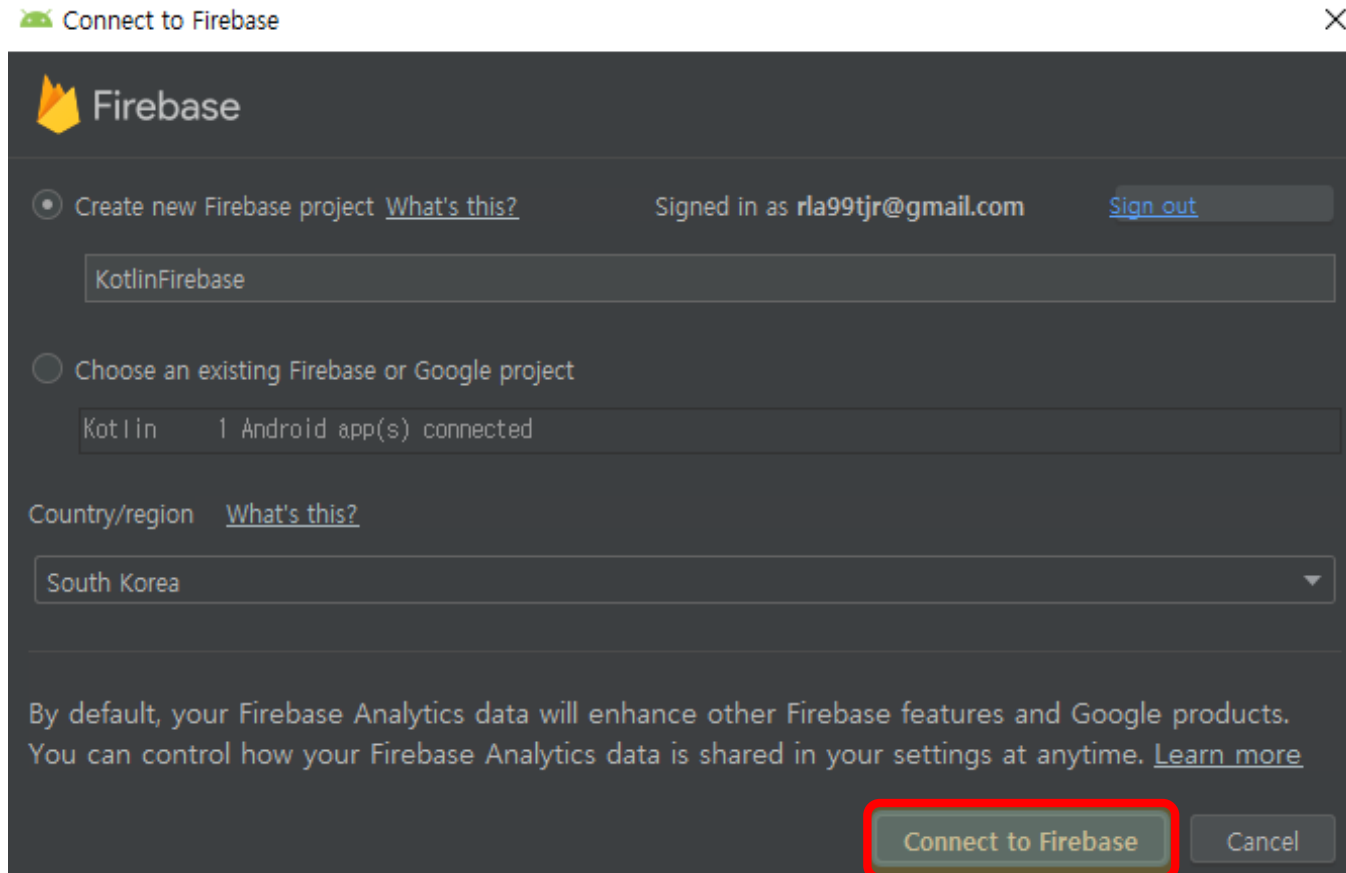
Gradle Console Event Log

코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 새로운 프로젝트 생성 혹은, 기존의 프로젝트 연결 후 Connect to Firebase 클릭

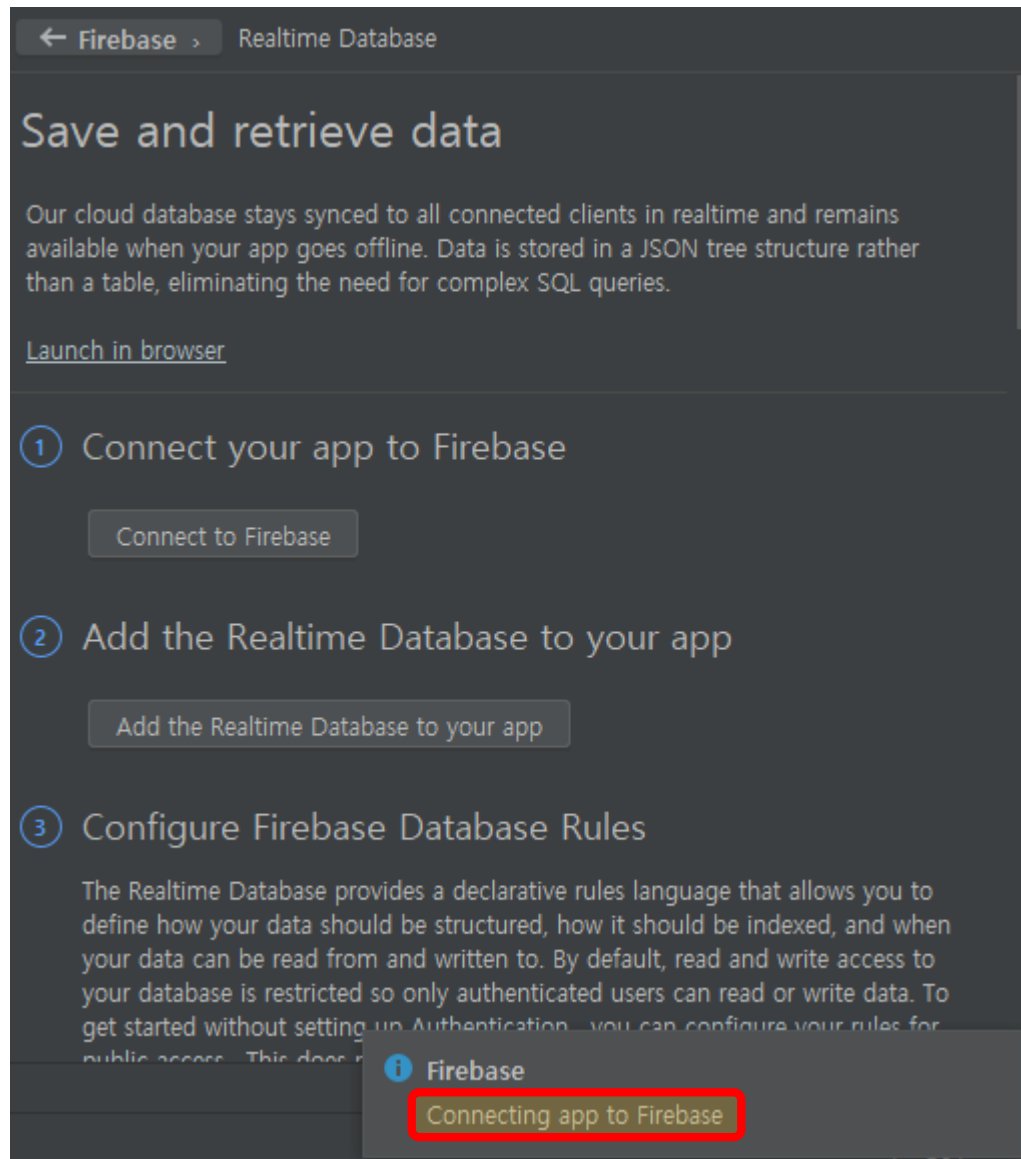


코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ Firebase 연결 중인 것 확인



코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ Connected로 연결 확인

The screenshot shows the Firebase Realtime Database interface. At the top, there's a navigation bar with a back arrow, the word 'Firebase', and 'Realtime Database'. Below this, the main heading is 'Save and retrieve data'. A paragraph explains that the cloud database is synced to all connected clients in realtime and remains available when the app goes offline. Data is stored in a JSON tree structure. A link 'Launch in browser' is present. Below this, there are three numbered steps: 1. 'Connect your app to Firebase' with a green 'Connected' status indicator in a red box; 2. 'Add the Realtime Database to your app' with a button labeled 'Add the Realtime Database to your app'; 3. 'Configure Firebase Database Rules' with a paragraph explaining the declarative rules language. At the bottom right, there's an information box titled 'Firebase' stating 'Firebase project created and connected locally to module: app.'

← Firebase > Realtime Database

Save and retrieve data

Our cloud database stays synced to all connected clients in realtime and remains available when your app goes offline. Data is stored in a JSON tree structure rather than a table, eliminating the need for complex SQL queries.

[Launch in browser](#)

- 1 Connect your app to Firebase
✓ Connected
- 2 Add the Realtime Database to your app
Add the Realtime Database to your app
- 3 Configure Firebase Database Rules
The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up [Authentication](#), you can [configure your rules for public access](#). This does not

i Firebase
Firebase project created and connected locally to module: app.

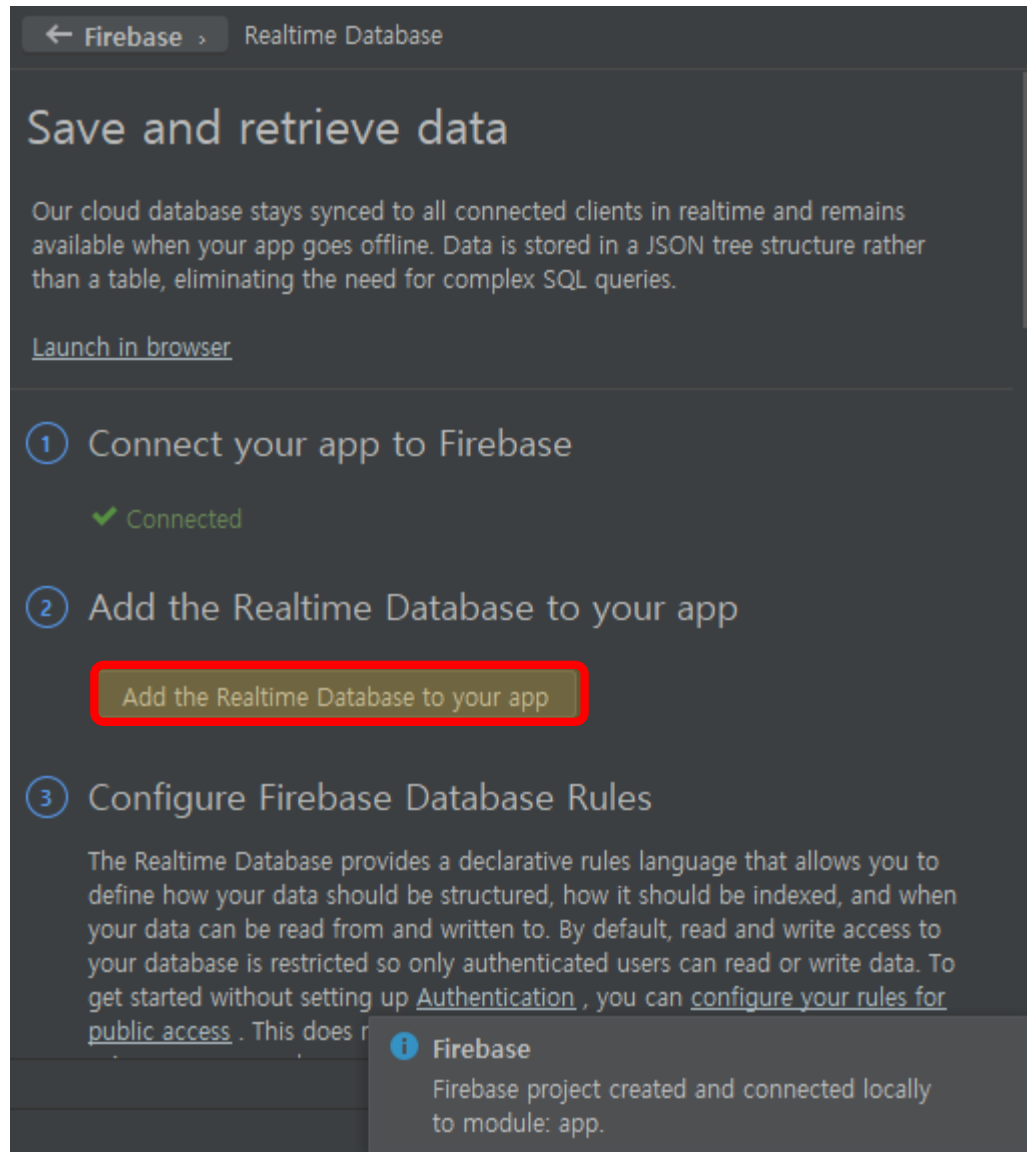
코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ Gradle에 추가를 위해서

Add the Realtime Database
to your App 클릭



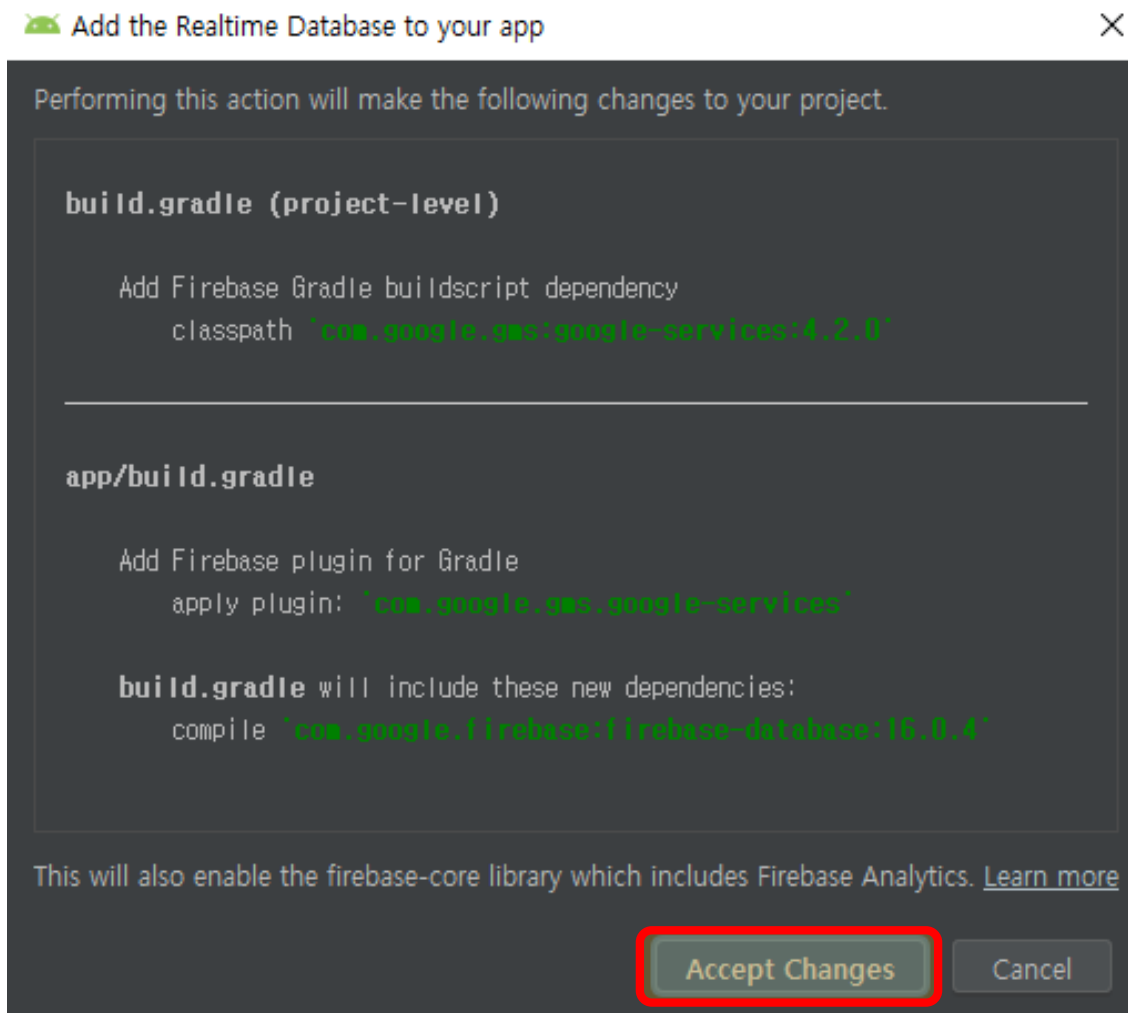
코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▷ Accept Changes 클릭하여

Gradle에 자동으로 추가

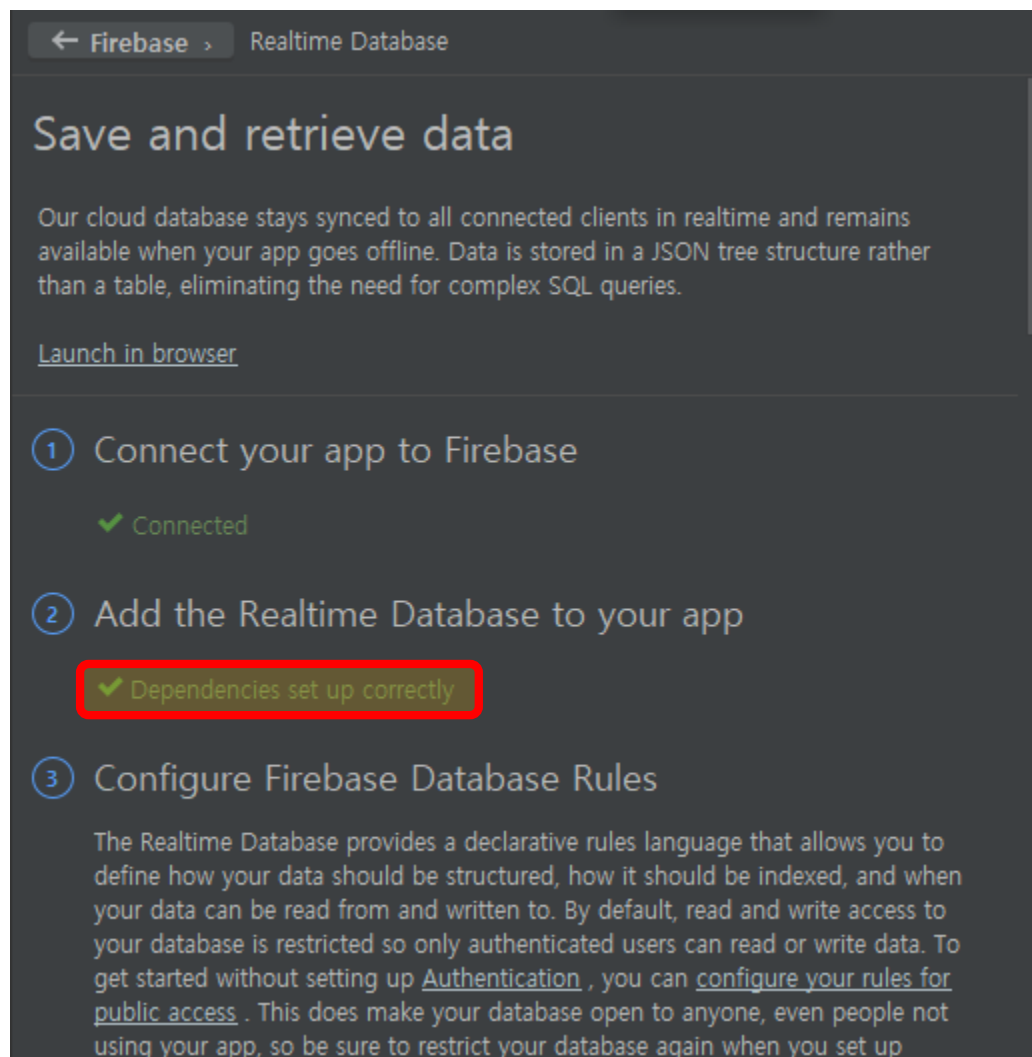


코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 정상적으로 연결 확인

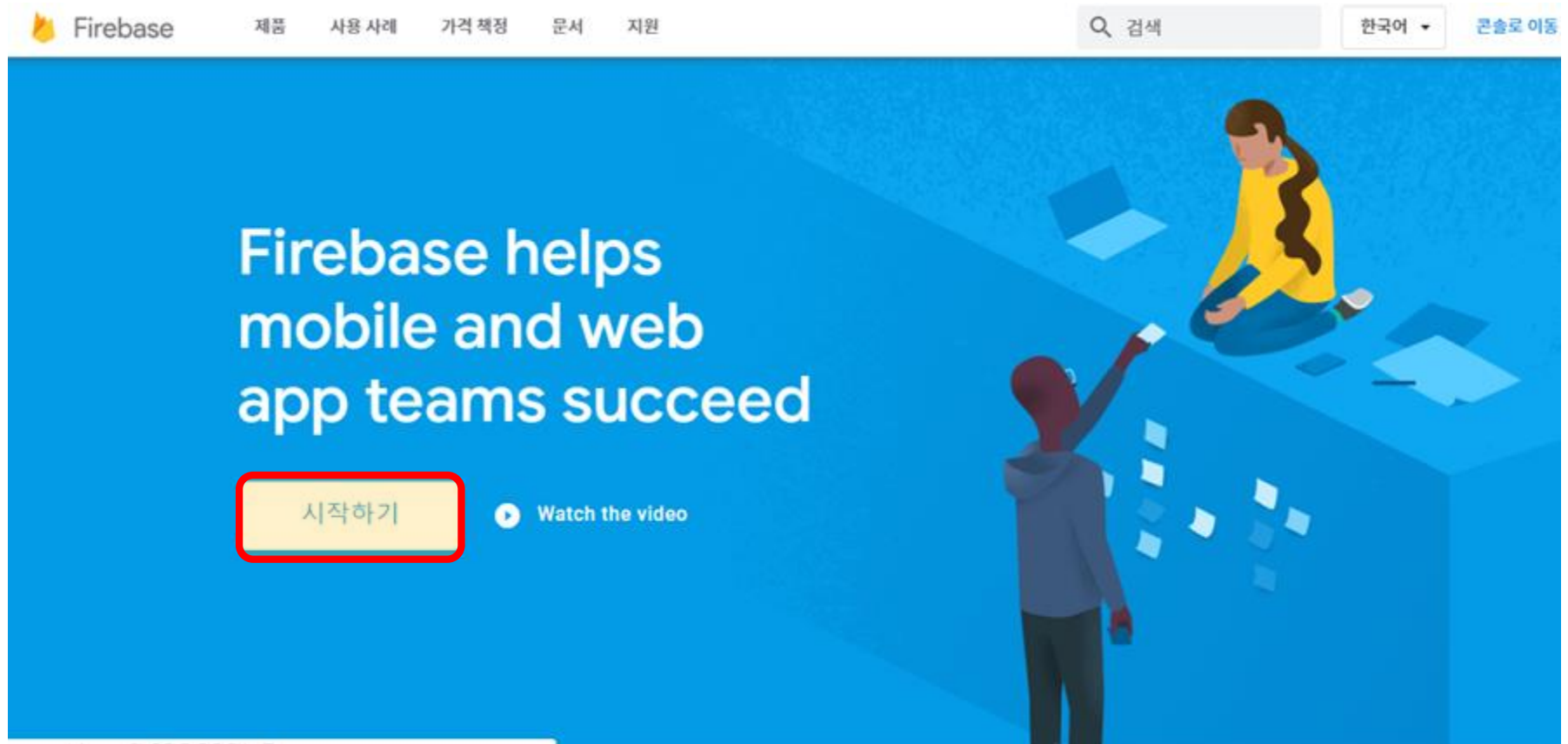


코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▷ <https://firebase.google.com/?hl=ko> 접속하여 Firebase 시작하기 클릭

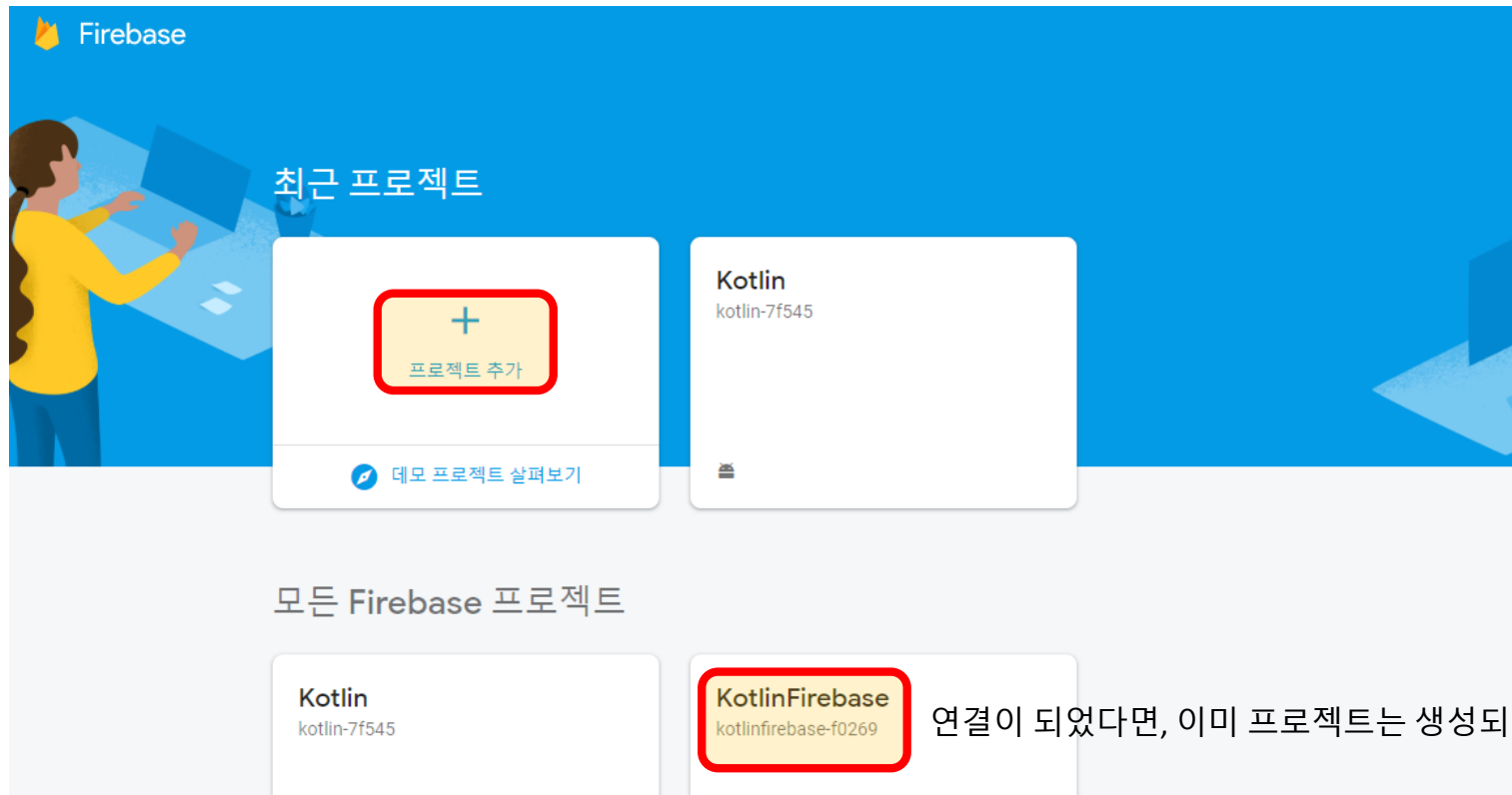


코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 기존의 프로젝트가 있는 경우 사용하여도 되고, 없을 경우 프로젝트 추가 클릭



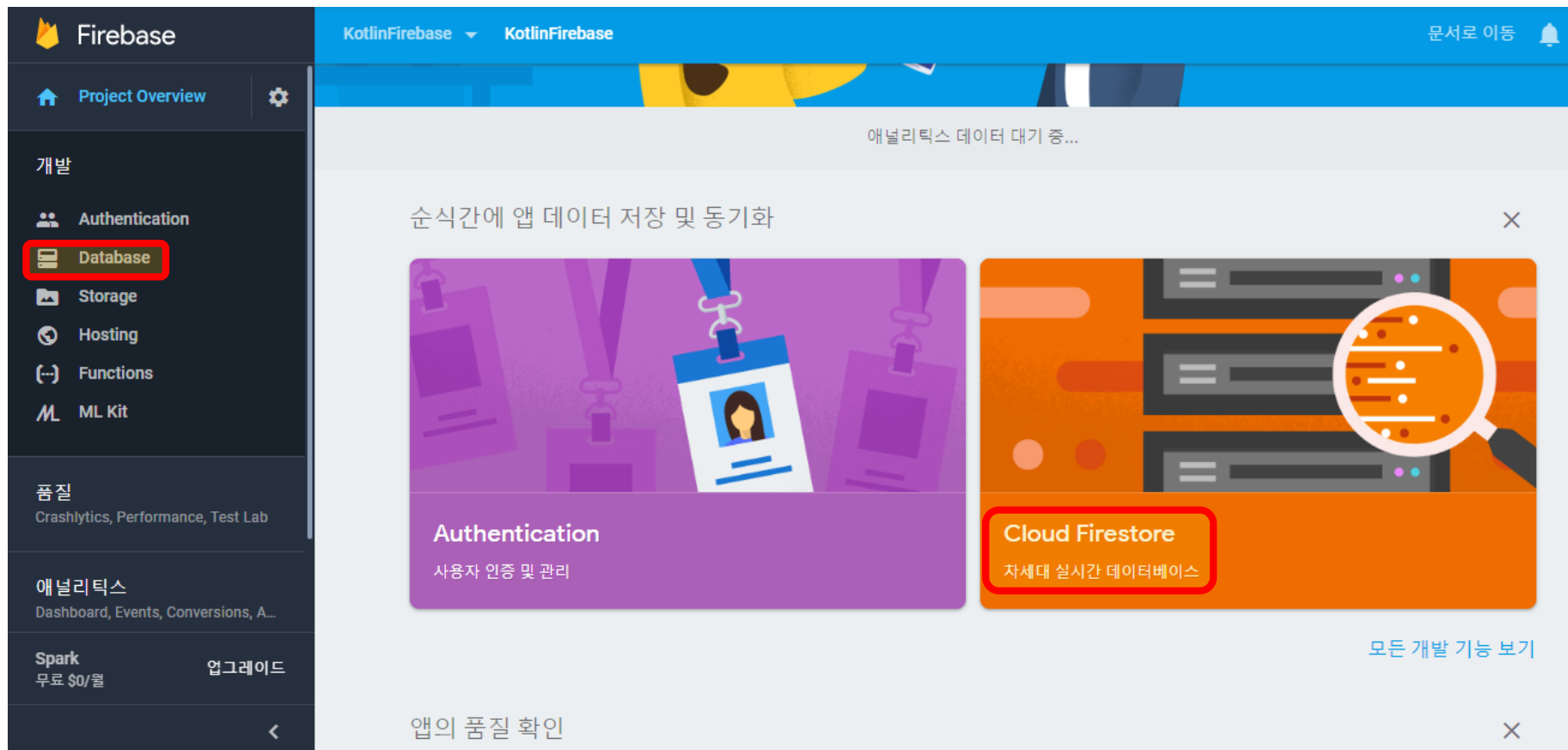
연결이 되었다면, 이미 프로젝트는 생성되어 있음

코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ Database 클릭



코틀린 기초

- ▶ Android Studio에서 Firebase 연결
 - ▶ 프로젝트와 Firebase 연결
 - ▶ 실시간 Realtime Database 생성하기

The screenshot shows the Firebase console interface. On the left is a dark sidebar with the 'Database' option highlighted. The main content area has an orange header with 'KotlinFirebase' and 'Database'. Below the header, the text '또는 Realtime Database 선택' (or select Realtime Database) is displayed. A large graphic shows three server racks connected by lines. To the right of the graphic, the 'Realtime Database' section explains it's the default Firebase database and includes links for documentation and a 'Create Database' button, which is highlighted with a red rectangle. At the bottom, there's a section for '개발자를 위한 추가 기능' (Additional features for developers).

Firebase

KotlinFirebase Database 문서로 이동

Project Overview

개발

- Authentication
- Database**
- Storage
- Hosting
- Functions
- ML Kit

품질

Crashlytics, Performance, Test Lab

애널리틱스

Dashboard, Events, Conversions, A...

또는 Realtime Database 선택

Realtime Database

Firebase의 기존 데이터베이스입니다. Cloud Firestore와 마찬가지로 실시간 데이터 동기화를 지원합니다.

[문서 보기](#) [자세히 알아보기](#)

데이터베이스 만들기

개발자를 위한 추가 기능

코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 잠금 모드 또는 테스트 모드로 데이터베이스 시작

실시간 데이터베이스 보안 규칙

데이터 구조를 정의한 후 규칙을 작성해 데이터를 보호해야 합니다.

[자세히 알아보기](#)

☒ 잠금 모드로 시작

모든 읽기 및 쓰기를 거부하여 데이터베이스를 비공개로 설정하세요.

☐ 테스트 모드로 시작

데이터베이스에 대한 모든 읽기 및 쓰기를 허용하여 빠르게 설정하세요.

```
{
  "rules": {
    ".read": false,
    ".write": false
  }
}
```

 모든 제3자 읽기 및 쓰기가 거부됩니다.

취소

사용 설정

실시간 데이터베이스 보안 규칙

데이터 구조를 정의한 후 규칙을 작성해 데이터를 보호해야 합니다.

[자세히 알아보기](#)


☐ 잠금 모드로 시작

모든 읽기 및 쓰기를 거부하여 데이터베이스를 비공개로 설정하세요.

☒ 테스트 모드로 시작

데이터베이스에 대한 모든 읽기 및 쓰기를 허용하여 빠르게 설정하세요.

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

 데이터베이스 참조를 사용하는 누구나 데이터베이스를 읽고 쓸 수 있습니다.

취소

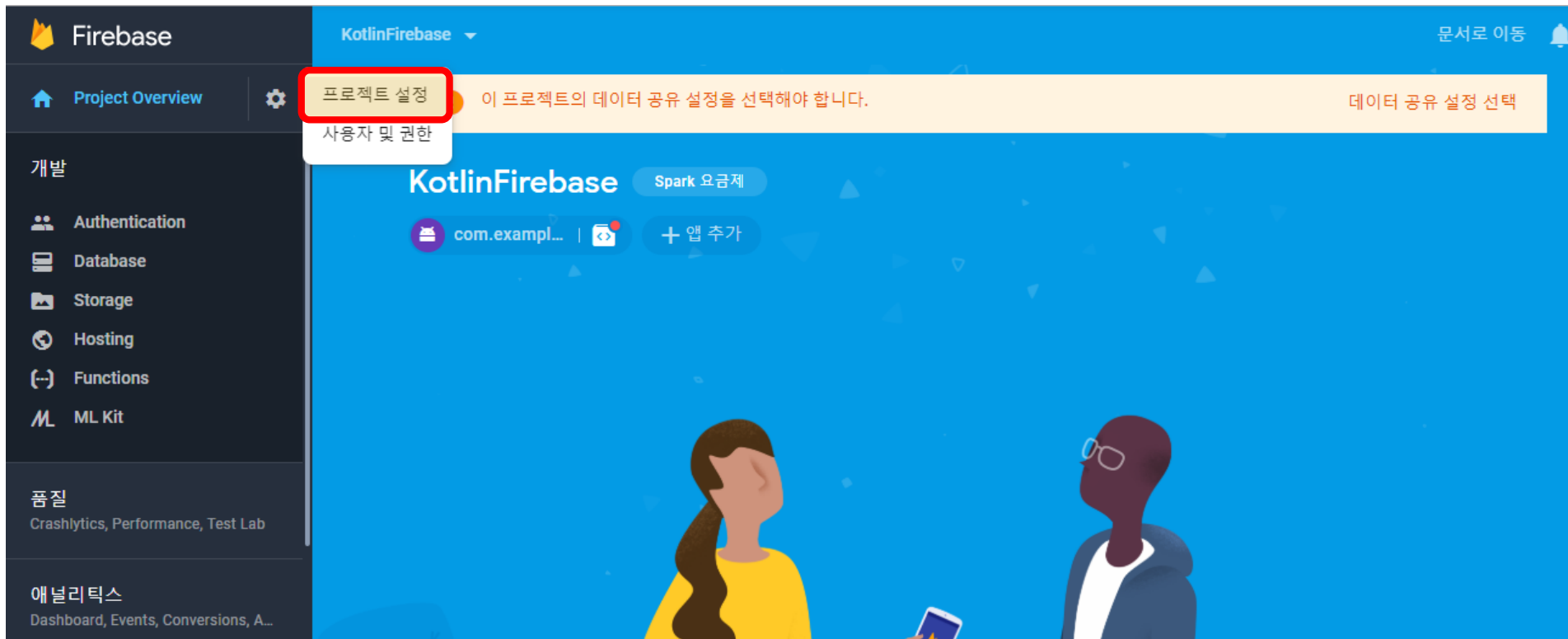
사용 설정

코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ Project 생성 후 Project Overview의 프로젝트 설정 클릭



코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 프로젝트 생성 후 google-services.json을 다운로드

The screenshot shows the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and navigation links: 'Project Overview' (selected), '개발' (Development) with sub-links for Authentication, Database, Storage, Hosting, Functions, and ML Kit; '품질' (Quality) with links for Crashlytics, Performance, and Test Lab; and '애널리틱스' (Analytics) with links for Dashboard, Events, Conversions, and A/B testing. The main area has a blue header with 'KotlinFirebase' and 'Settings'. Below this, a list of Android apps shows 'com.example.kotlinfirebase' selected. The right pane displays the app's configuration, including the 'google-services.json' download button (highlighted with a red box), the app ID '1:841779766227:android:b6e8b65a191e04ad', the app nickname 'com.example.kotlinfirebase', and the SHA-1 certificate fingerprint '9c:d0:a4:0f:d5:90:46:5b:d6:01:c1:8e:1b:86:2f:95:9d:d5:7c:fc'.

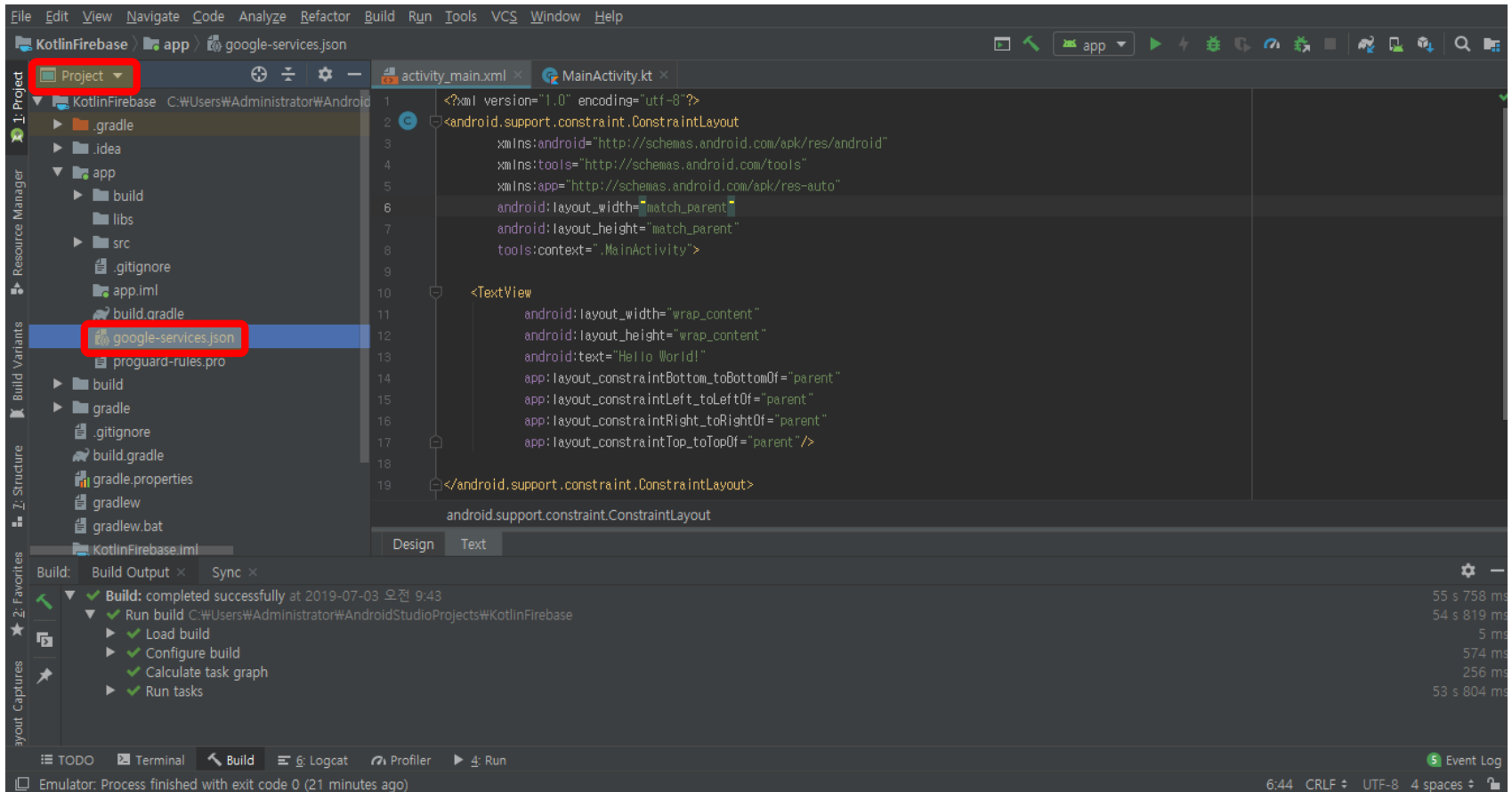
앱 ID	앱 닉네임	SHA-1 인증서 지문	유형
1:841779766227:android:b6e8b65a191e04ad	com.example.kotlinfirebase	9c:d0:a4:0f:d5:90:46:5b:d6:01:c1:8e:1b:86:2f:95:9d:d5:7c:fc	SHA-1

코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ 다운받은 google-services.json 파일을 Project 수준에서의 app에 추가

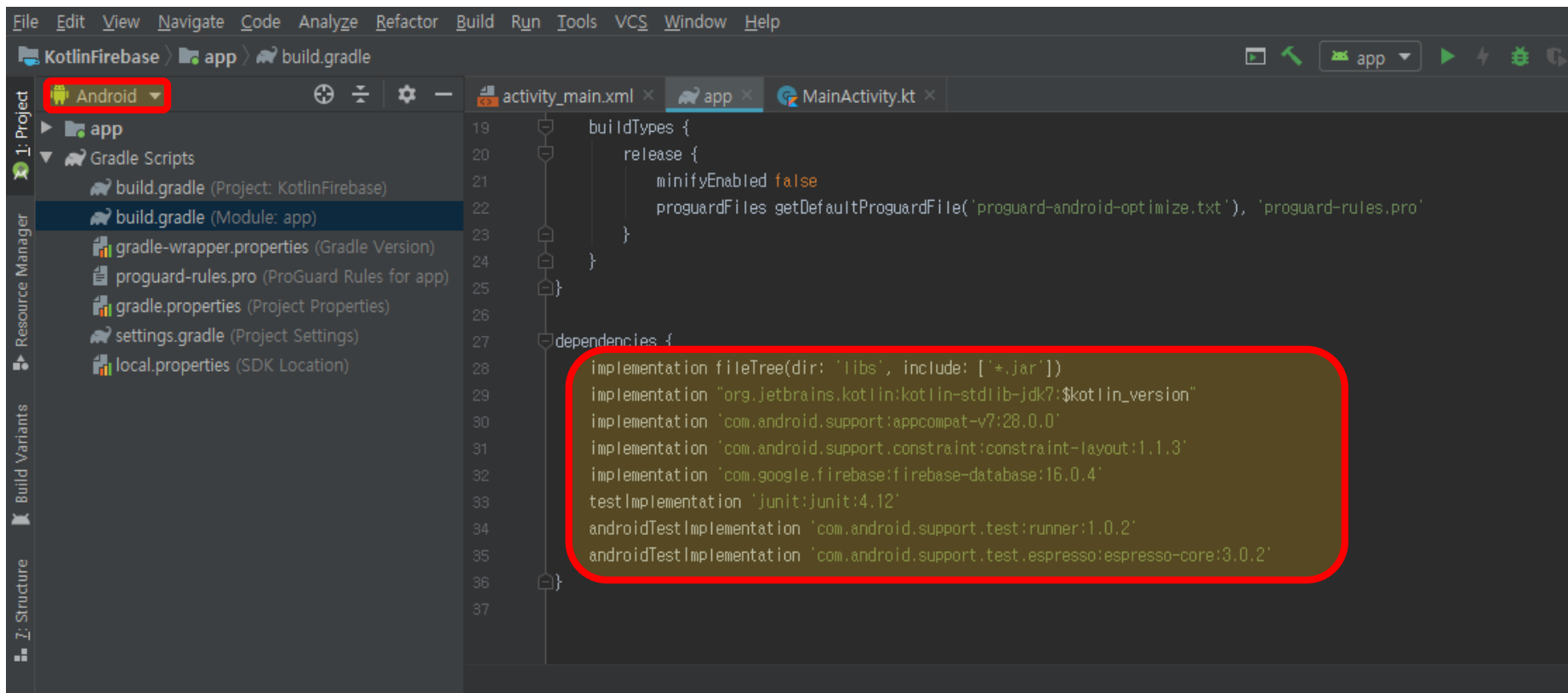


코틀린 기초

▶ Android Studio에서 Firebase 연결

▶ 프로젝트와 Firebase 연결

▶ Android의 build.gradle에서 Firebase 버전 및 implementation 확인



코틀린 기초

▶ Kotlin - FireBase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

- ▶ Android와 Firebase의 정보 저장을 위한 간단한 예제
- ▶ Android에서 정보 입력 시 Firebase에 정보 저장 확인



코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

▶ activity_main.xml에서 정보 입력을 위한 EditText와 저장 버튼을 위한 Button 생성

```
<EditText
    android:id="@+id/testEditText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
/>
<Button
    android:id="@+id/testSaveBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:background="@android:color/holo_red_dark"
    android:text="저장"
    app:layout_constraintEnd_toEndOf="@+id/testEditText"
    app:layout_constraintStart_toStartOf="@+id/testEditText"
    app:layout_constraintTop_toBottomOf="@+id/ratingBar" />
```


코틀린 기초

▶ Kotlin - FireBase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

▷ activity_main.xml에서 Button 작성, 이번 강의에서는 조회와 리스트 버튼의 기능은 구현하지 않음

```
<Button
    android:id="@+id/nextBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:background="@color/colorPrimary"
    android:text="조회"
    app:layout_constraintEnd_toEndOf="@+id/testEditText"
    app:layout_constraintStart_toStartOf="@+id/testEditText"
    app:layout_constraintTop_toBottomOf="@+id/testSaveBtn" />

<Button
    android:id="@+id/list"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="64dp"
    android:background="@android:color/holo_blue_dark"
    android:text="리스트"
    app:layout_constraintEnd_toEndOf="@+id/testEditText"
    app:layout_constraintStart_toStartOf="@+id/testEditText"
    app:layout_constraintTop_toBottomOf="@+id/testSaveBtn"
    app:layout_constraintHorizontal_bias="0.501" />
```

코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

▷ main.xml에 표시되는 RatingBar와 ImageView, TextView 생성

```
<RatingBar
    android:id="@+id/ratingBar"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:numStars="5"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/testEditText" />
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" app:srcCompat="@drawable/original_10"
    android:id="@+id/imageView"
    app:layout_constraintStart_toStartOf="parent" android:layout_marginStart="8dp"
    app:layout_constraintEnd_toEndOf="parent" android:layout_marginEnd="8dp" android:layout_marginTop="8dp"
    app:layout_constraintTop_toTopOf="parent" />
<TextView
    android:text="Kotlin Programming"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/textView" app:layout_constraintStart_toStartOf="parent" android:layout_marginStart="8dp"
    app:layout_constraintEnd_toEndOf="parent" android:layout_marginEnd="8dp"
    app:layout_constraintBottom_toTopOf="@+id/testEditText" android:layout_marginTop="8dp"
    app:layout_constraintTop_toBottomOf="@+id/imageView"
    android:textAppearance="@style/TextAppearance.AppCompat.Body1" android:textSize="30sp"
    android:textStyle="bold" />
```

코틀린 기초

▶ Kotlin - FireBase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

▶ Kotlin으로 MainActivity를 구성

```
lateinit var testEditText:EditText  
lateinit var testSaveBtn:Button  
lateinit var ratingBar:RatingBar
```

▶ View를 전역변수로 선언하고, lateinit 를 통해서 늦은 초기화를 통해 초기화 시 메모리 할당

코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

- ▷ MainActivity.kt 에서 onCreate 내부에서 View를 초기화
- ▷ Button에 해당하는 OnClickListener를 세팅해주고 saveText() 메소드 생성

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
        testEditText = findViewById(R.id.testEditText)  
        testSaveBtn = findViewById(R.id.testSaveBtn)  
        ratingBar = findViewById(R.id.ratingBar)  
        testSaveBtn.setOnClickListener {  
            saveText()  
        }  
    }  
}
```

코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

- ▶ RatingBar를 위한 Rating.kt 생성
- ▶ Rating.kt 에서 Rating 클래스를 생성하여 Parameter의 저장형식을 지정함
- ▶ Class를 생성하지 않고, String 및 int 단일 값으로 생성가능

```
class Rating(val id:String, val name:String, val rating:Int)
```

코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

- ▶ MainActivity.kt에서 저장을 위한 saveText() 메소드 내부 내용 작성
- ▶ EditText.text.toString() 으로 name의 값을 가져옴
- ▶ firebase는 FirebaseDatabase 객체를 저장하고, ref 는 참조 경로를 저장.

```
private fun saveText() {  
    val name = testEditText.text.toString().trim()  
    val firebase = FirebaseDatabase.getInstance()  
    val ref = firebase.getReference("inputData")  
  
    if (name.isEmpty()) {  
        testEditText.error = "영화 제목을 입력해주세요."  
        return  
    }  
    val rateId = ref.push().key  
    val rating = rateId.toString().let {  
        Rating(it, name, ratingBar.rating.toInt())  
    }  
    ref.child(rateId.toString()).setValue(rating).addOnCompleteListener {  
        Toast.makeText(applicationContext, "저장", Toast.LENGTH_SHORT).show()  
    }  
}
```

코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

- ▶ MainActivity.kt에서 저장을 위한 saveText() 메소드 내부 내용 작성
- ▶ EditText의 값이 비어 있을 경우, error 메시지를 표시
- ▶ 고유 Id를 가져오기 위해 ratelId에 ref.push().key를 저장
- ▶ 키 값, editText 값, ratingBar의 별점을 저장하고 있는 객체를 생성하여 rating에 저장
- ▶ 결과 값이 float 형이기 때문에 toInt()로 형 변환

```
private fun saveText() {  
    val name = testEditText.text.toString().trim()  
    val firebase = FirebaseDatabase.getInstance()  
    val ref = firebase.getReference("inputData")  
  
    if (name.isEmpty()) {  
        testEditText.error = "영화 제목을 입력해주세요."  
        return  
    }  
    val ratelId = ref.push().key  
    val rating = ratelId.toString().let {  
        Rating(it, name, ratingBar.rating.toInt())  
    }  
    ref.child(ratelId.toString()).setValue(rating).addOnCompleteListener {  
        Toast.makeText(applicationContext, "저장", Toast.LENGTH_SHORT).show()  
    }  
}
```

코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

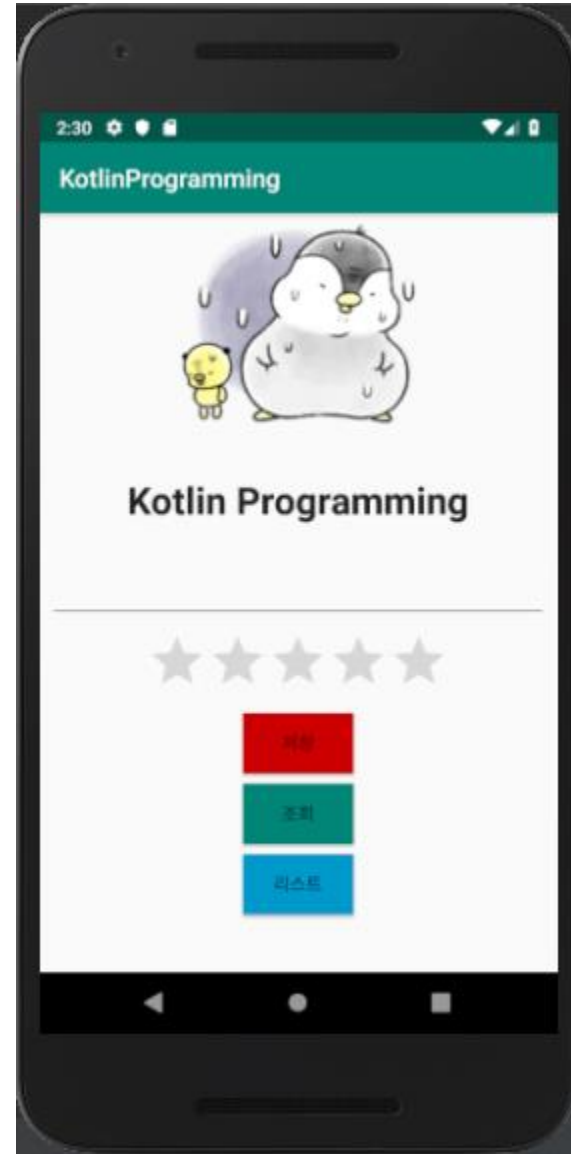
▶ Android와 Firebase의 연결

- ▶ MainActivity.kt에서 저장을 위한 saveText() 메소드 내부 내용 작성
- ▶ child() 메소드를 통해 경로 아래를 탐색하며 push()로 생성된 키 값에 바로 rating 객체를 setValue().
addOnCompleteListener에 저장 완료 후 이벤트를 설정 = “저장”

```
private fun saveText() {  
    val name = testEditText.text.toString().trim()  
    val firebase = FirebaseDatabase.getInstance()  
    val ref = firebase.getReference("inputData")  
  
    if (name.isEmpty()) {  
        testEditText.error = "영화 제목을 입력해주세요."  
        return  
    }  
    val rateld = ref.push().key  
    val rating = rateld.toString().let {  
        Rating(it, name, ratingBar.rating.toInt())  
    }  
    ref.child(rateld.toString()).setValue(rating).addOnCompleteListener {  
        Toast.makeText(applicationContext, "저장", Toast.LENGTH_SHORT).show()  
    }  
}
```


코틀린 기초

- ▶ Kotlin - FireBase의 연동을 통한 앱 작성
 - ▶ Android와 Firebase의 연결
 - ▶ 어플 실행 화면



코틀린 기초

▶ Kotlin - FireBase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

- ▶ 어플 실행 화면
- ▶ 영화 제목과 함께 평점 4점을 입력
- ▶ 정상적으로 저장 시 “저장” 메시지 출력



코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

▶ Firebase 데이터베이스 내의 name과 rating이 정상적으로 입력된 모습

The screenshot displays the Firebase console interface. On the left, the 'Database' tab is selected in the sidebar. The main area shows a list of database entries under the 'inputData' collection. The entry with ID '-LipfFcbSi7y6_K0IN2z' is highlighted with a red box, showing its details: id: '-LipfFcbSi7y6_K01l', name: 'Kotlin Programmi', and rating: 4. The URL bar shows 'https://kotlin-7f545.firebaseio.com/'.

```
https://kotlin-7f545.firebaseio.com/

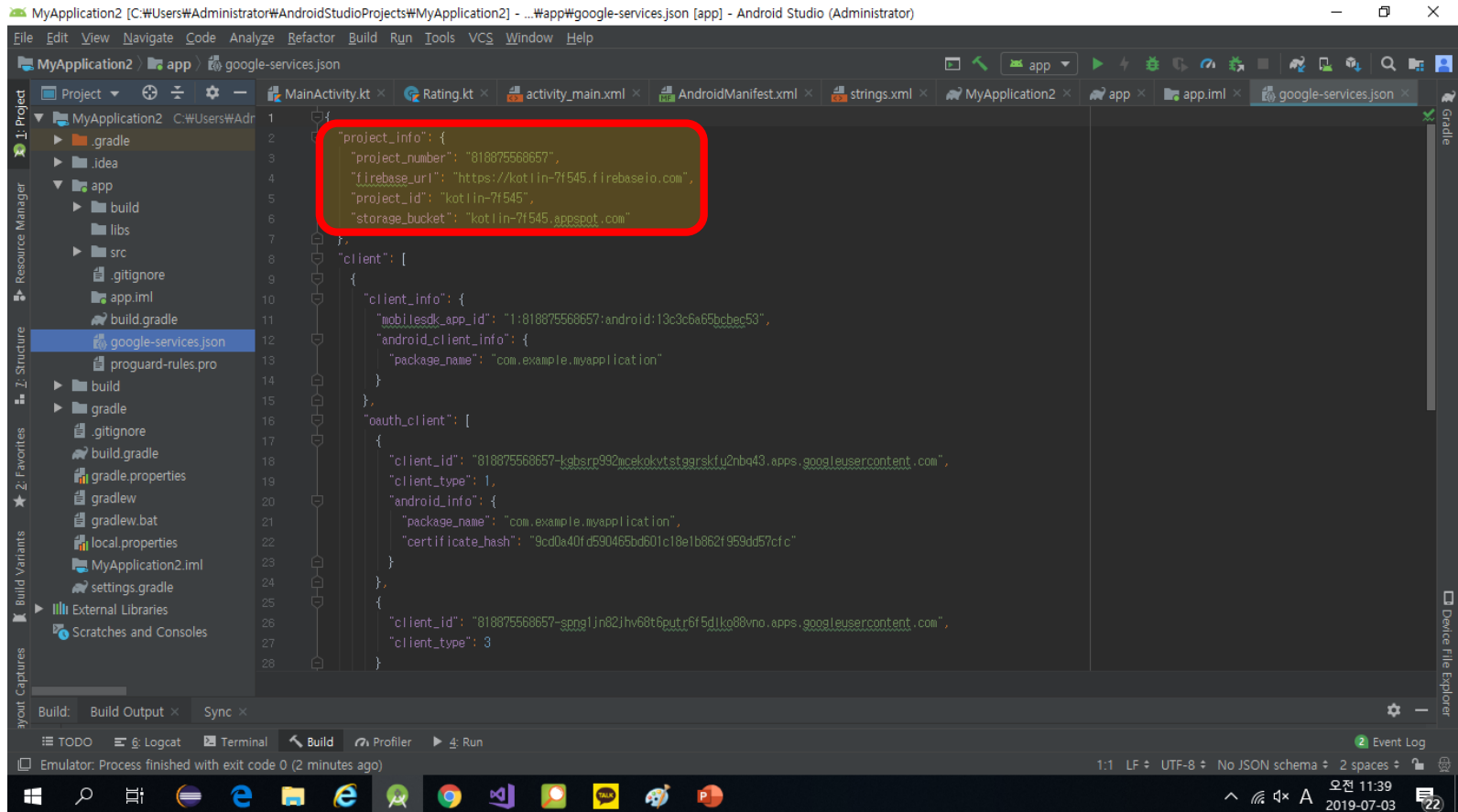
kotlin-7f545
├── inputData
│   ├── -LikXceNLG3kU0ANju4-
│   ├── -Lik_QeaqW_KNA236bK
│   ├── -LikbWqhgm17IqMTmUxy
│   ├── -Like3fCQpb-DYP6RiN7
│   ├── -LiITpkPpKRK0MXWX6k-
│   ├── -LiliFKSFmKVzDdW9SFy
│   └── -LipfFcbSi7y6_K0IN2z
│       ├── id: "-LipfFcbSi7y6_K01l"
│       ├── name: "Kotlin Programmi"
│       └── rating: 4
```

코틀린 기초

▶ Kotlin - Firebase의 연동을 통한 앱 작성

▶ Android와 Firebase의 연결

▶ Android의 프로젝트 수준의 json파일에서의 값과 데이터 베이스의 파일 값 일치하는지 확인



Q & A
