

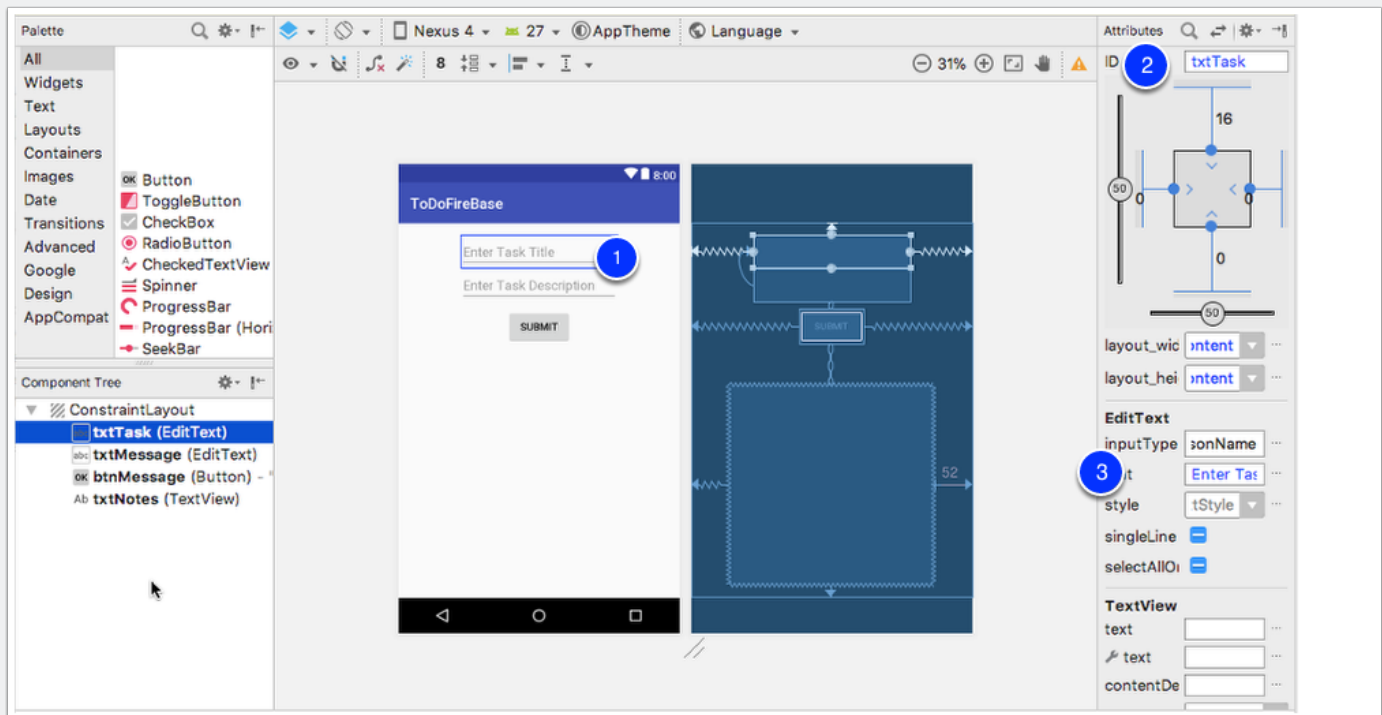
Android Emulator - ToDoList using Firebase

Github Link: <https://github.com/RVCAndroidClass/TaskFirebase>

1. Go to <http://gmail.com>
2. Create a NEW Gmail Account Just For App
3. Logoff of Google in all of your web browsers (go to <http://google.com> in all your browsers)
4. Create New Project Named MyToDoList

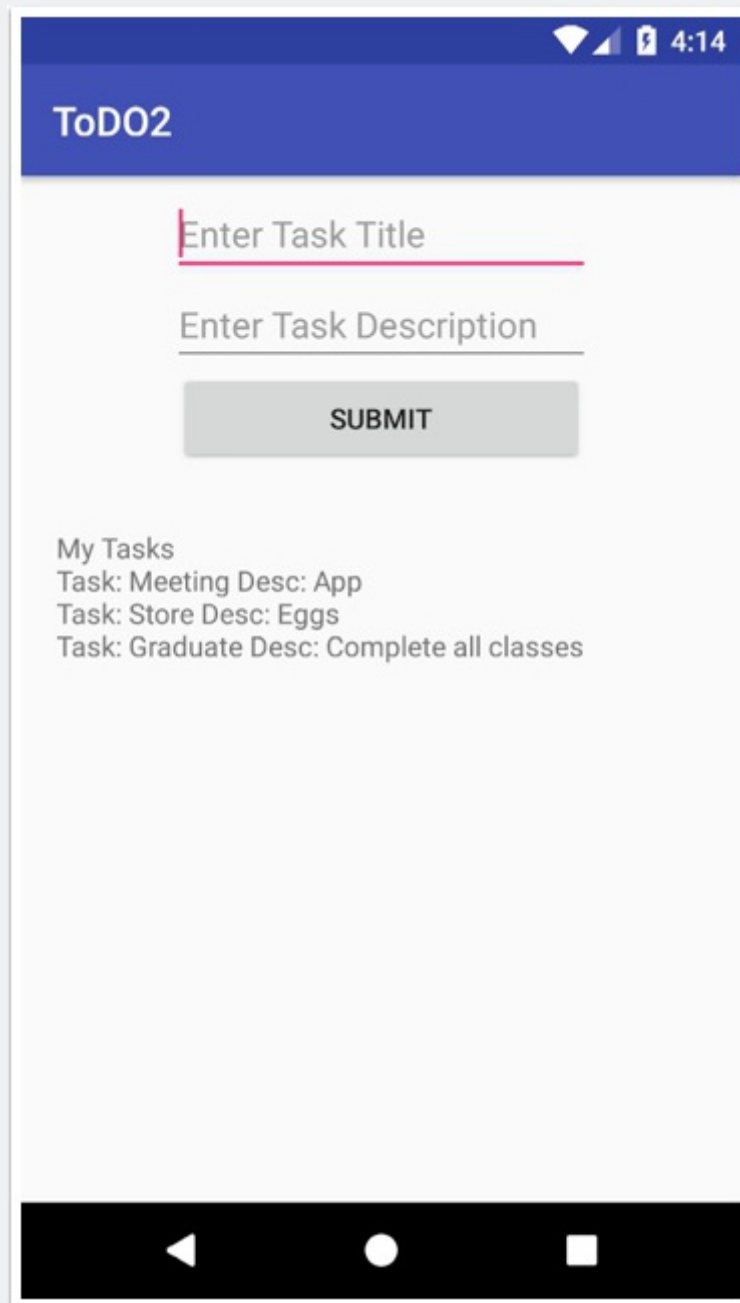
Create New Project Named MyToDoList

1. Remove Hello Label
2. Add EditText,
3. ID: txtTask
4. Hint: Enter Task Name



Android Emulator - ToDoList using Firebase

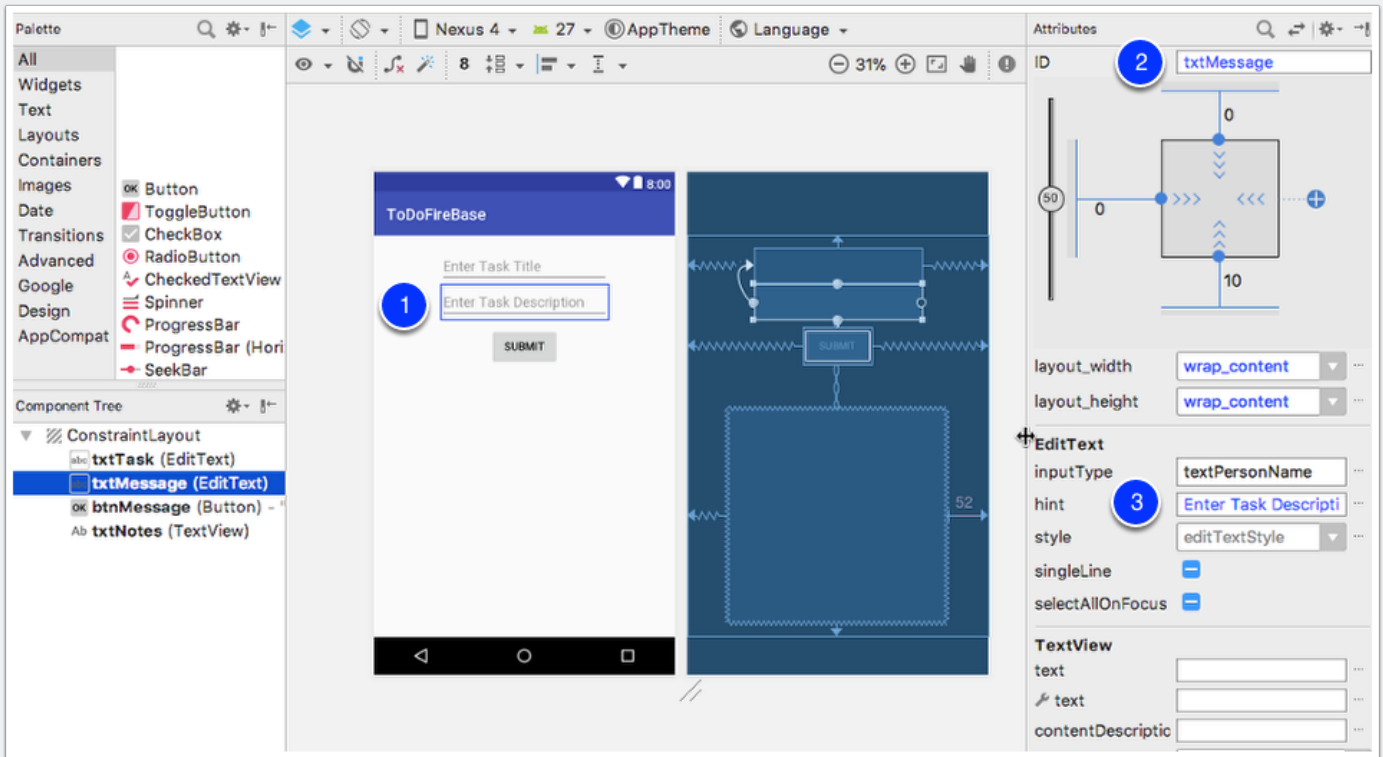
MyToDoList with FireBase



Android Emulator - ToDoList using Firebase

activity_main.xml - Add txtMessage

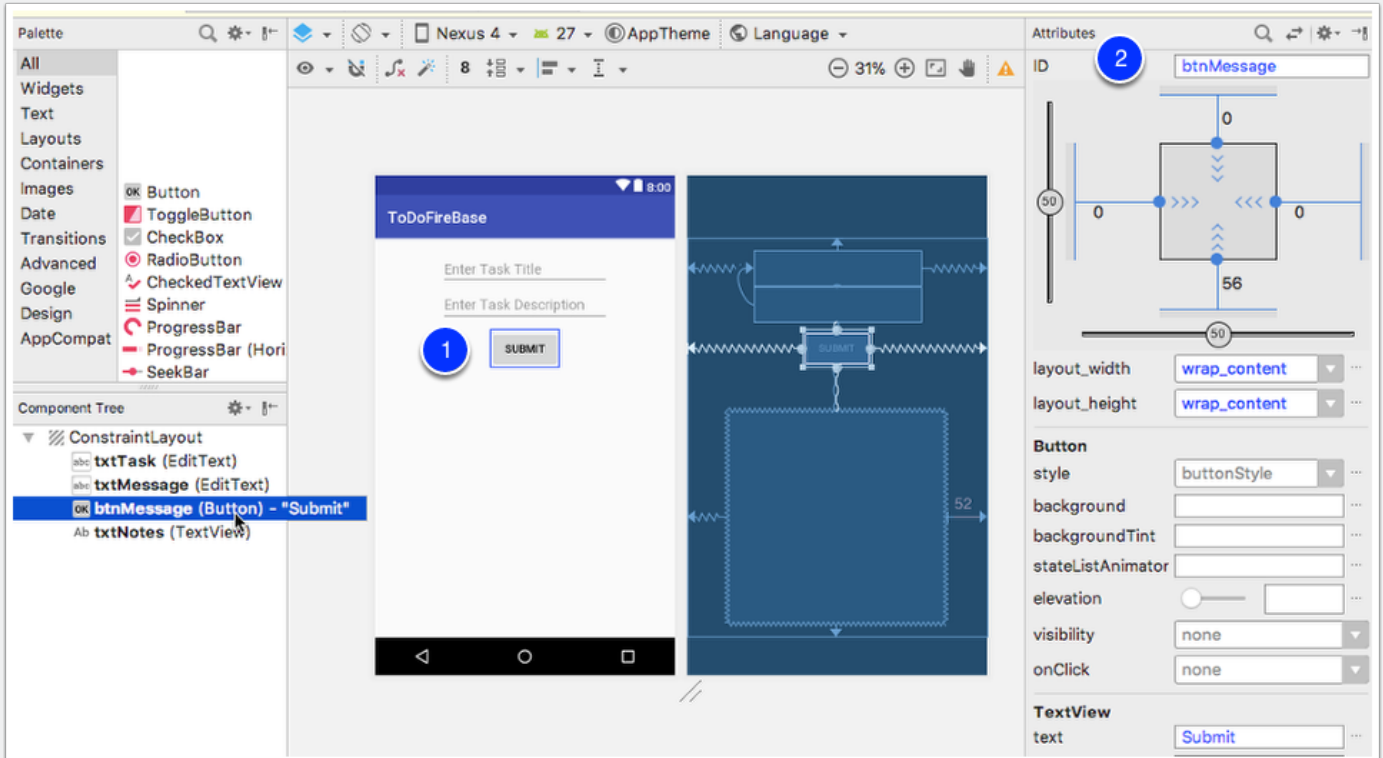
1. Add EditText
2. ID: txtMessage
3. Hint: Enter Task Description



Android Emulator - ToDoList using Firebase

Adding btnMessage

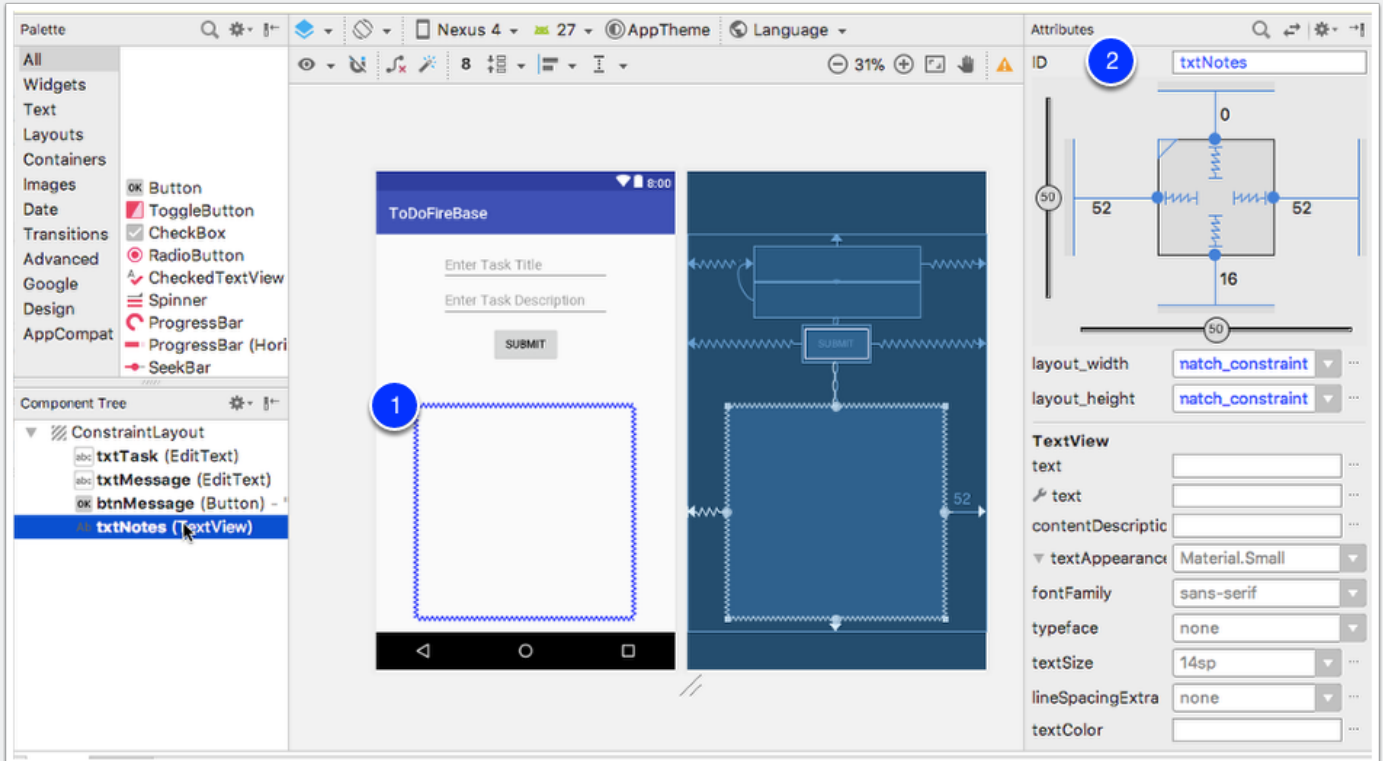
1. Add Button
2. ID: btnMessage



Android Emulator - ToDoList using Firebase

activity_main.xml - Add txtNotes

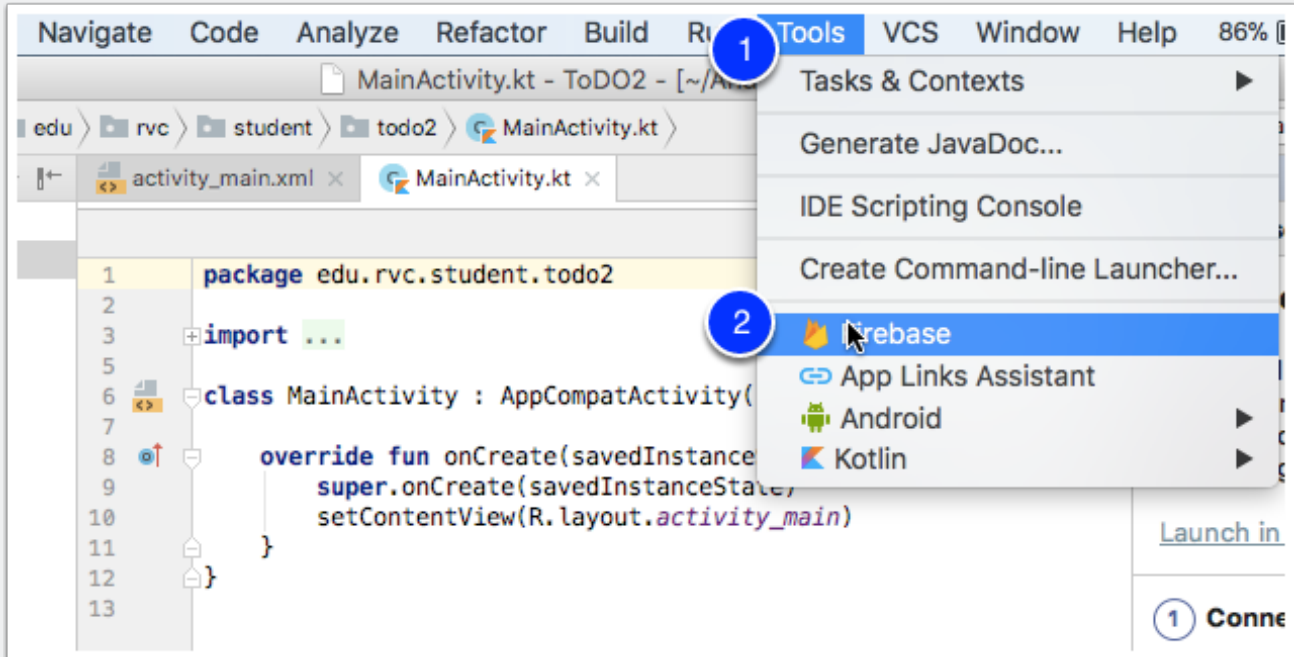
1. Add TextView
2. ID; txtNotes



Android Emulator - ToDoList using Firebase

Create a FireBase Database Connection

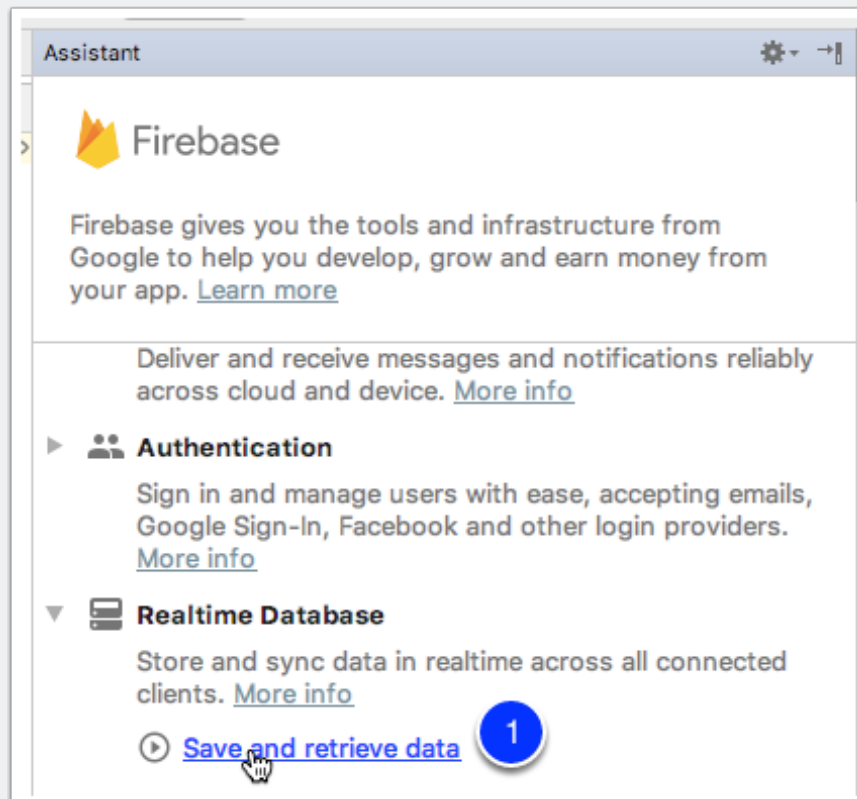
1. Click on Tools
2. Click on Firecase



Android Emulator - ToDoList using Firebase

Create FireBase Connection

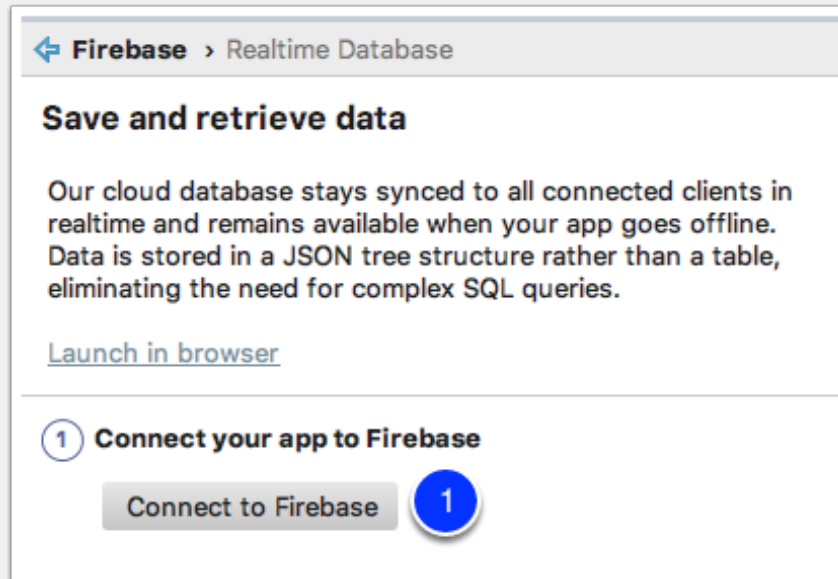
1. Click on Save and Retrieve data



Android Emulator - ToDoList using Firebase

Create FireBase Connection

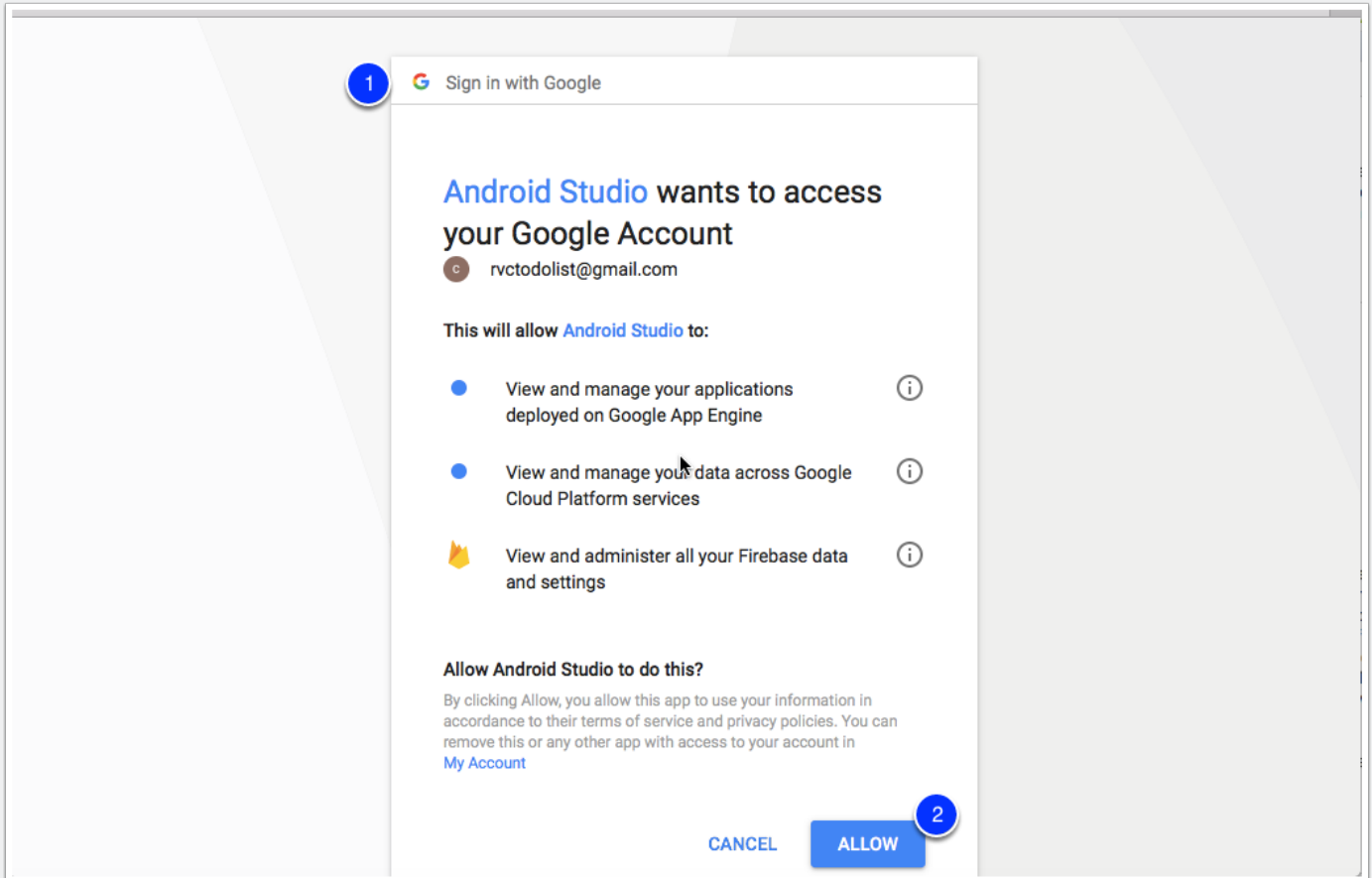
Click on **Connect to Firebase**



Android Emulator - ToDoList using Firebase

Sign in - Google Accounts

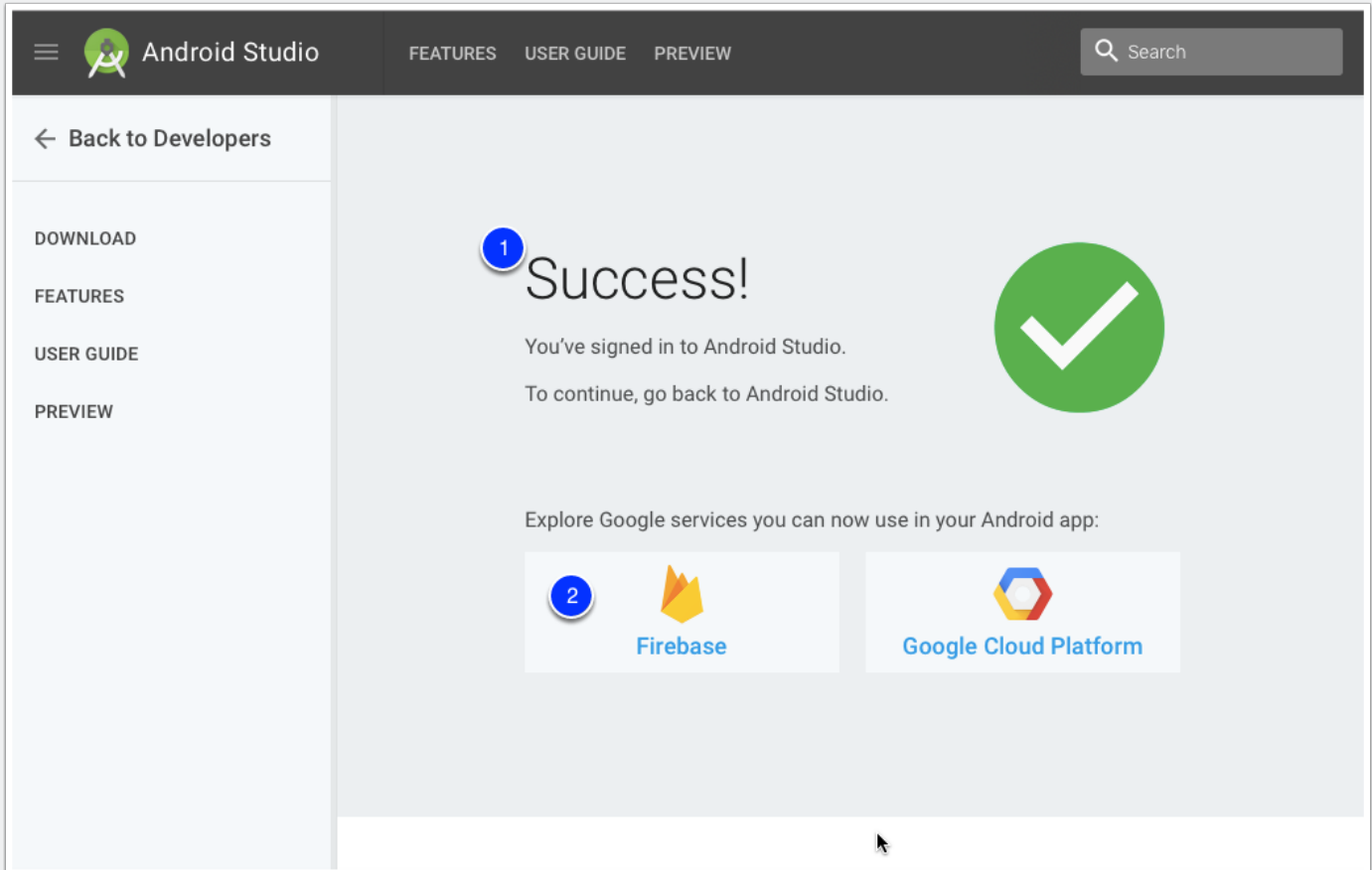
1. Sign-in to new Google account
2. Allow Android Studio to Access your new Google Account



Android Emulator - ToDoList using Firebase

Sign into Android Studio | Android Studio

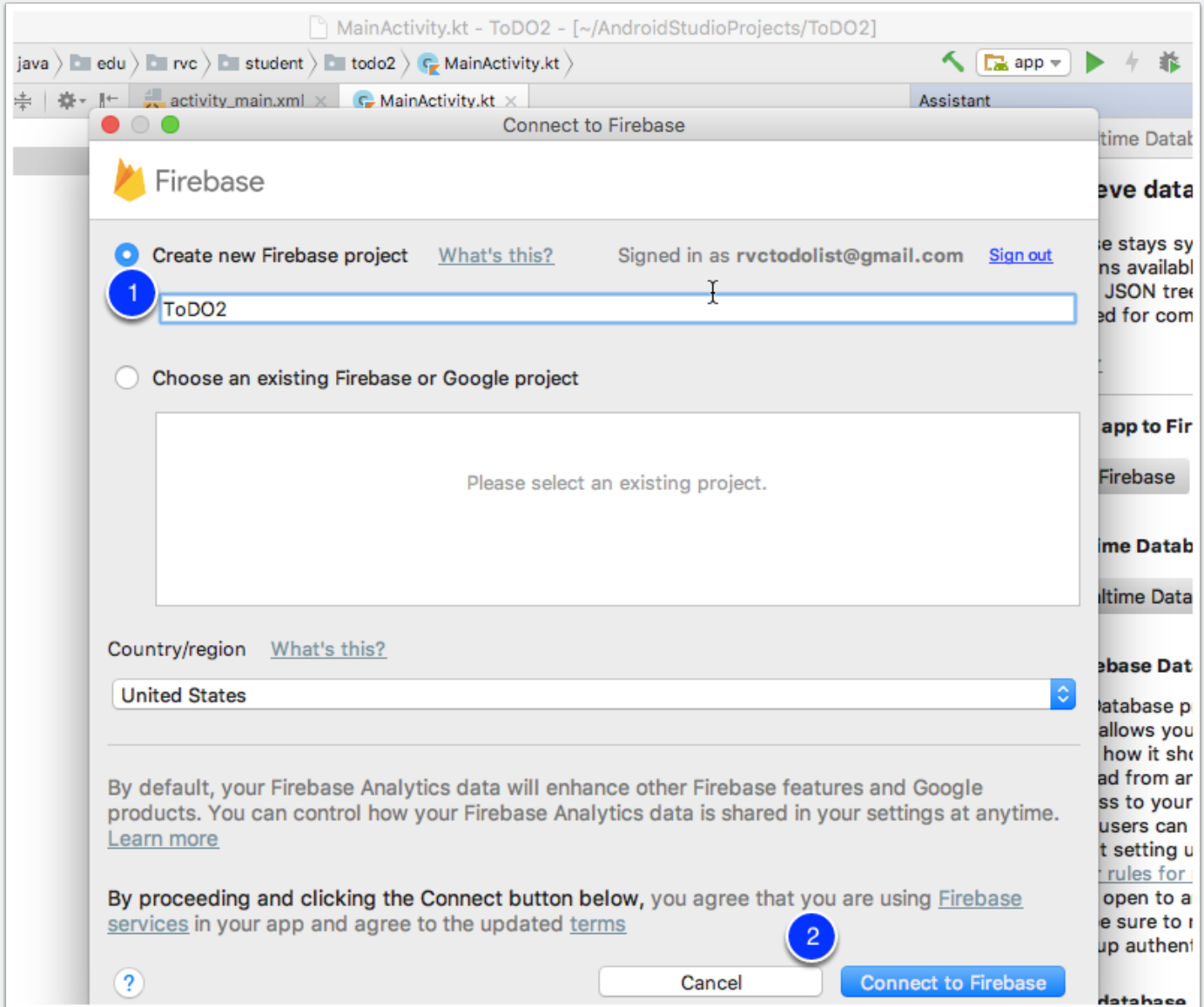
1. You Should See a Success Message!
2. Click on **Firebase**



Android Emulator - ToDoList using Firebase

Connect to Firebase

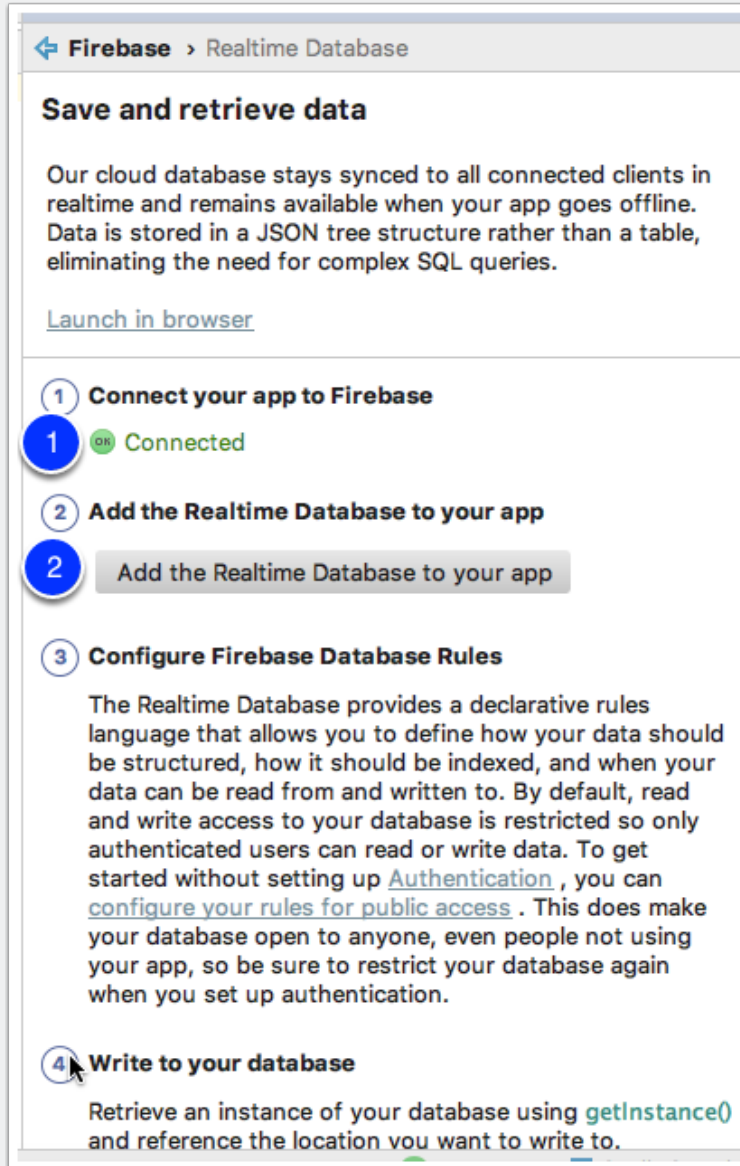
1. It should populate with your new Android Project Name
2. Click **Connect to FireBase**



Android Emulator - ToDoList using Firebase

Connect to Firebase

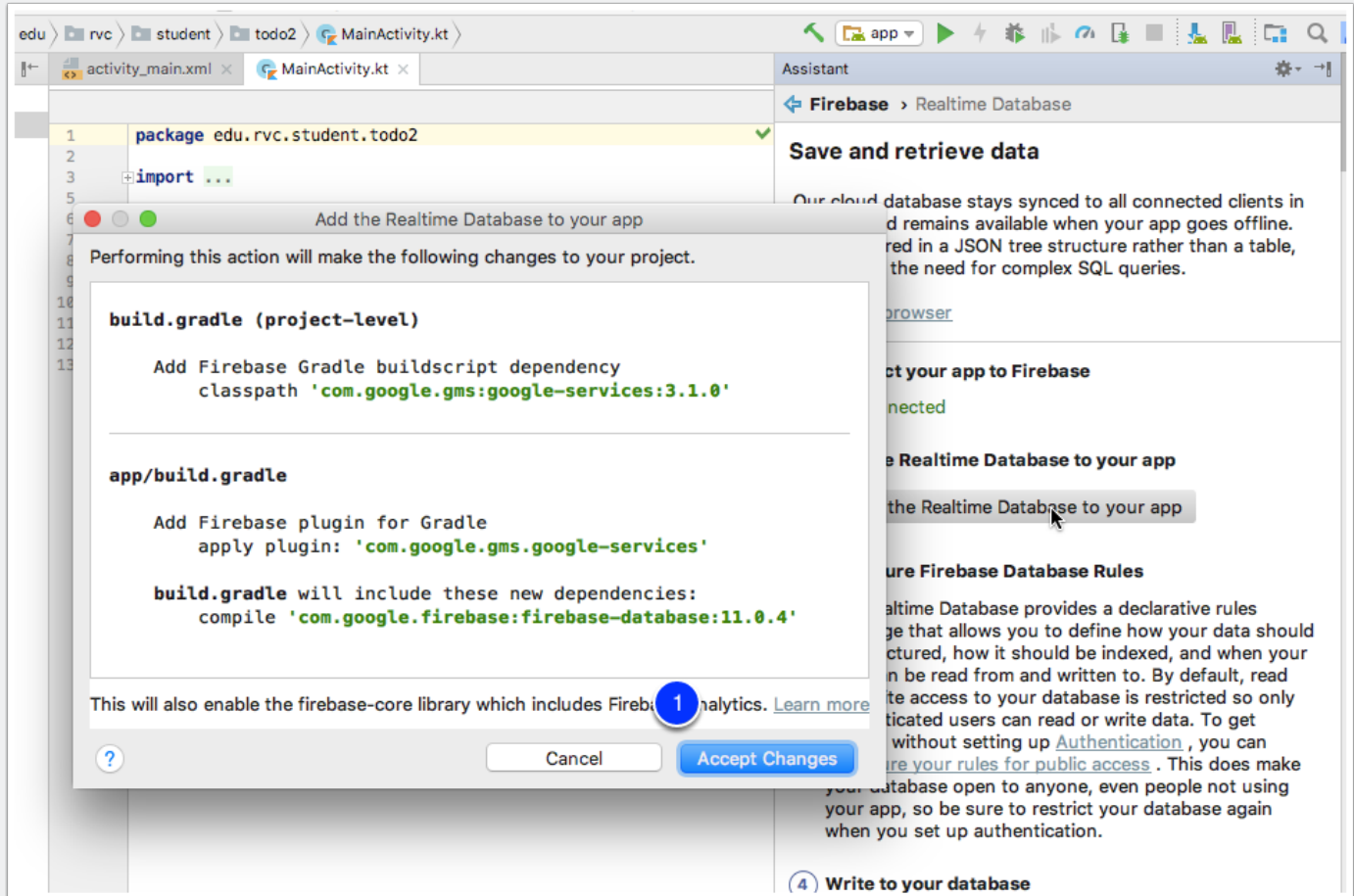
1. Should see **Connected**
2. Click on **Add the Realtime Database to your app**



Android Emulator - ToDoList using Firebase

Add the Realtime Database to your app

1. Click **Accept Changes**
2. Allow to Update Gradle (takes a bit)



Android Emulator - ToDoList using Firebase



Connect to Firebase

Should see success (green icon) once Gradle is done

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**
 - 1**  **Dependencies set up correctly**
- 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.

Android Emulator - ToDoList using Firebase

Connect to Firebase

1. Update Firebase Rules
2. Allow all to read and write
3. Click on **configure your rules for public access**

3 Configure Firebase Database Rules

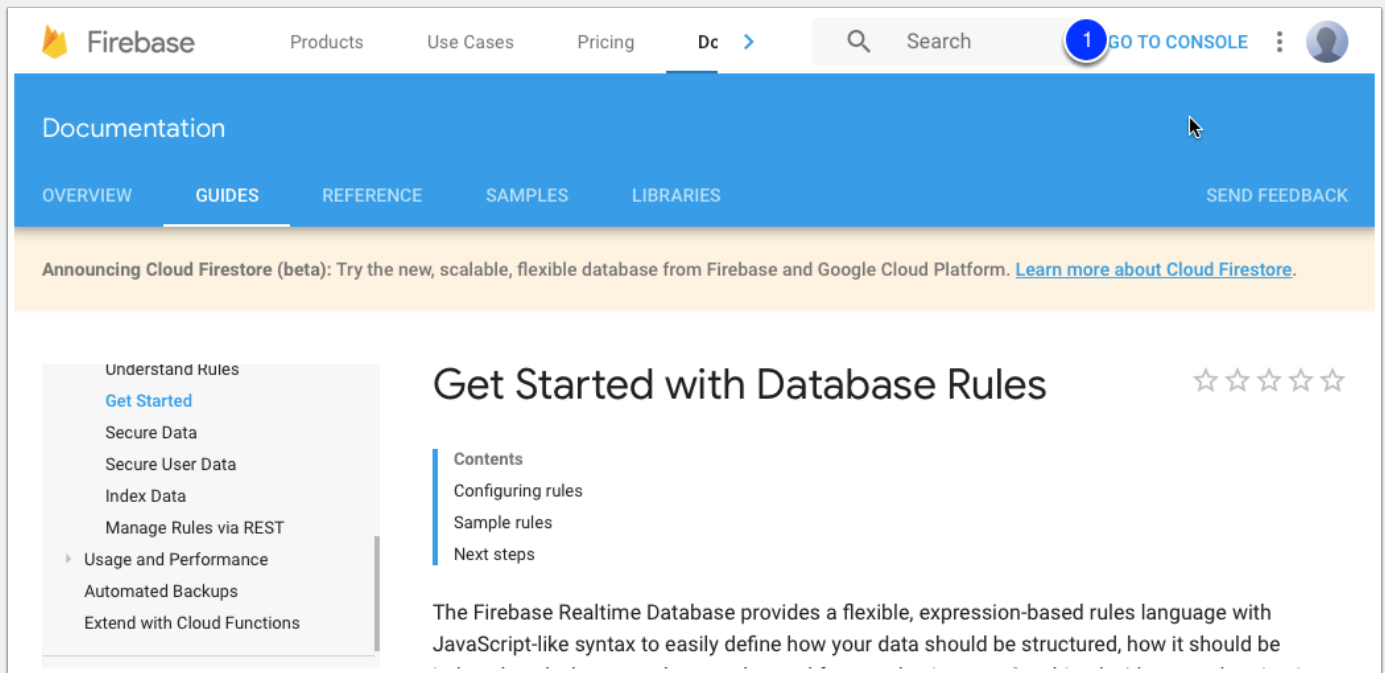
The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read 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.

1

Android Emulator - ToDoList using Firebase

Connect to Firebase

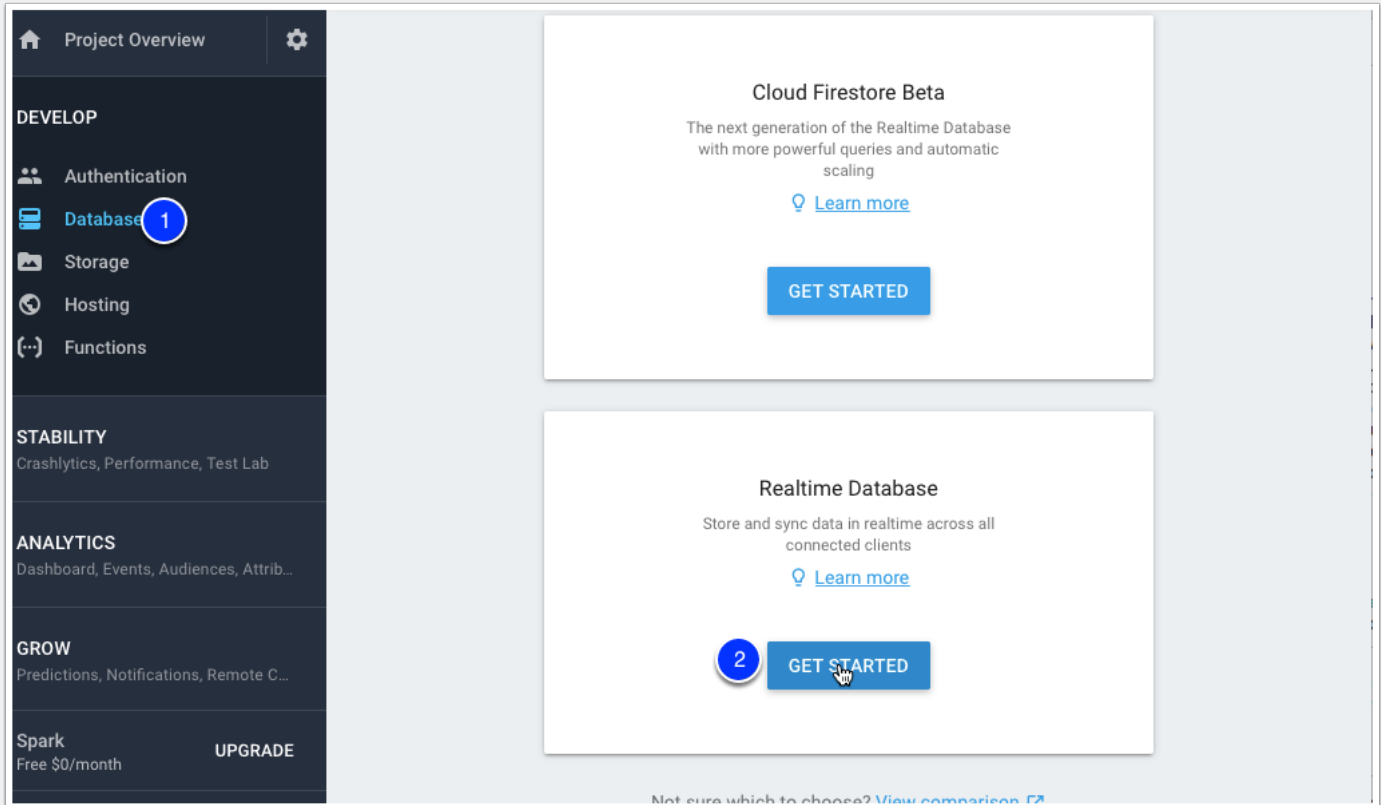
Click on **GO TO CONSOLE**



Android Emulator - ToDoList using Firebase

Connect to Firebase

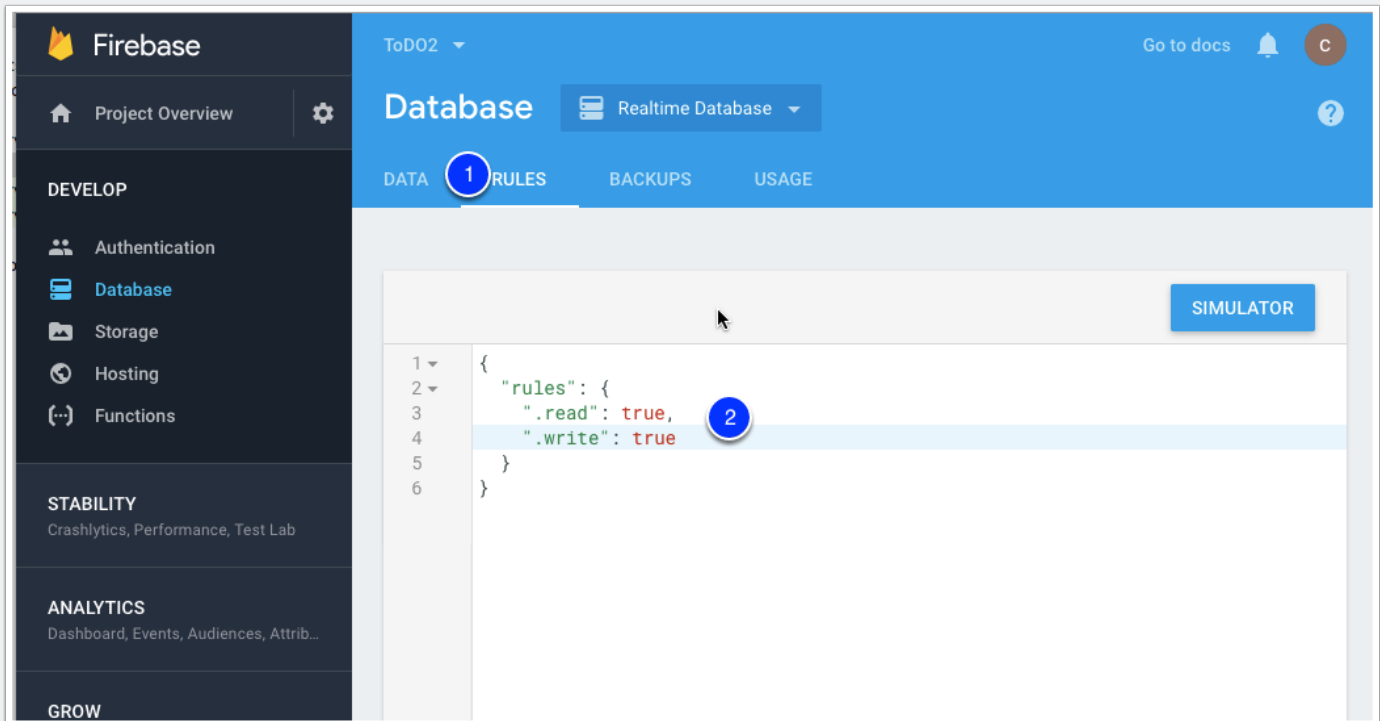
1. Click on **Database**
2. Click on **Get Started** under Realtime Database



Android Emulator - ToDoList using Firebase

Connect to Firebase

1. Click on **RULES** tab
2. Make sure your rule looks like this
3. It allows all who use app to read and write to database



Android Emulator - ToDoList using Firebase

Connect to Firebase

1. Write a test message to database
2. Copy code under #4
3. Paste below line 15 in MainActivity.kt (It should convert to Kotlin)
4. Run App

```
1 package edu.rvc.student.todo2
2
3 import android.support.v7.app.AppCompatActivity
4 import android.os.Bundle
5 import com.google.firebase.database.DatabaseReference
6 import com.google.firebase.database.FirebaseDatabase
7
8
9
10 class MainActivity : AppCompatActivity() {
11
12     override fun onCreate(savedInstanceState: Bundle?) {
13         super.onCreate(savedInstanceState)
14         setContentView(R.layout.activity_main)
15
16         // Write a message to the database
17         val database = FirebaseDatabase.getInstance()
18         val myRef = database.getReference("message")
19
20         myRef.setValue("Hello, World!")
21     }
22 }
23
```

Firebase > Realtime Database

be structured, now 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.

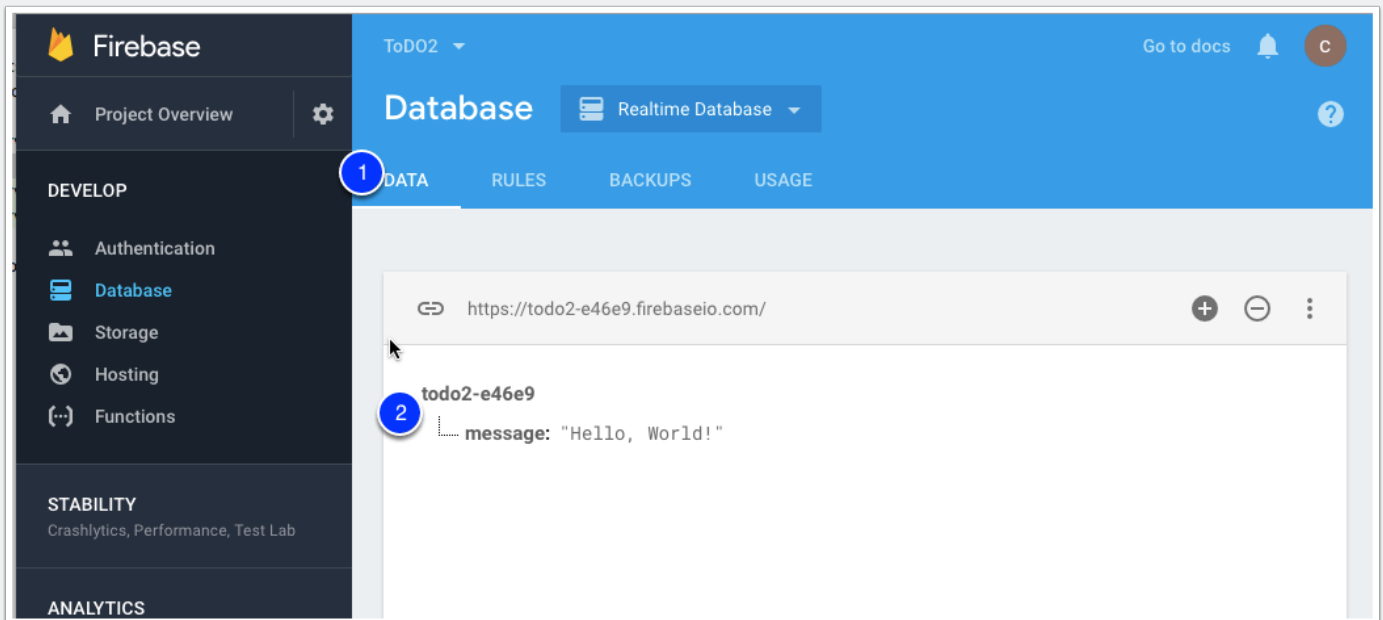
```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance()
DatabaseReference myRef = database.getReference("message")
myRef.setValue("Hello, World!");
```

You can save a range of data types to the database this way, including Java objects. When you save an object the responses from any getters will be saved as children of this location.

Android Emulator - ToDoList using Firebase

Connect to Firebase

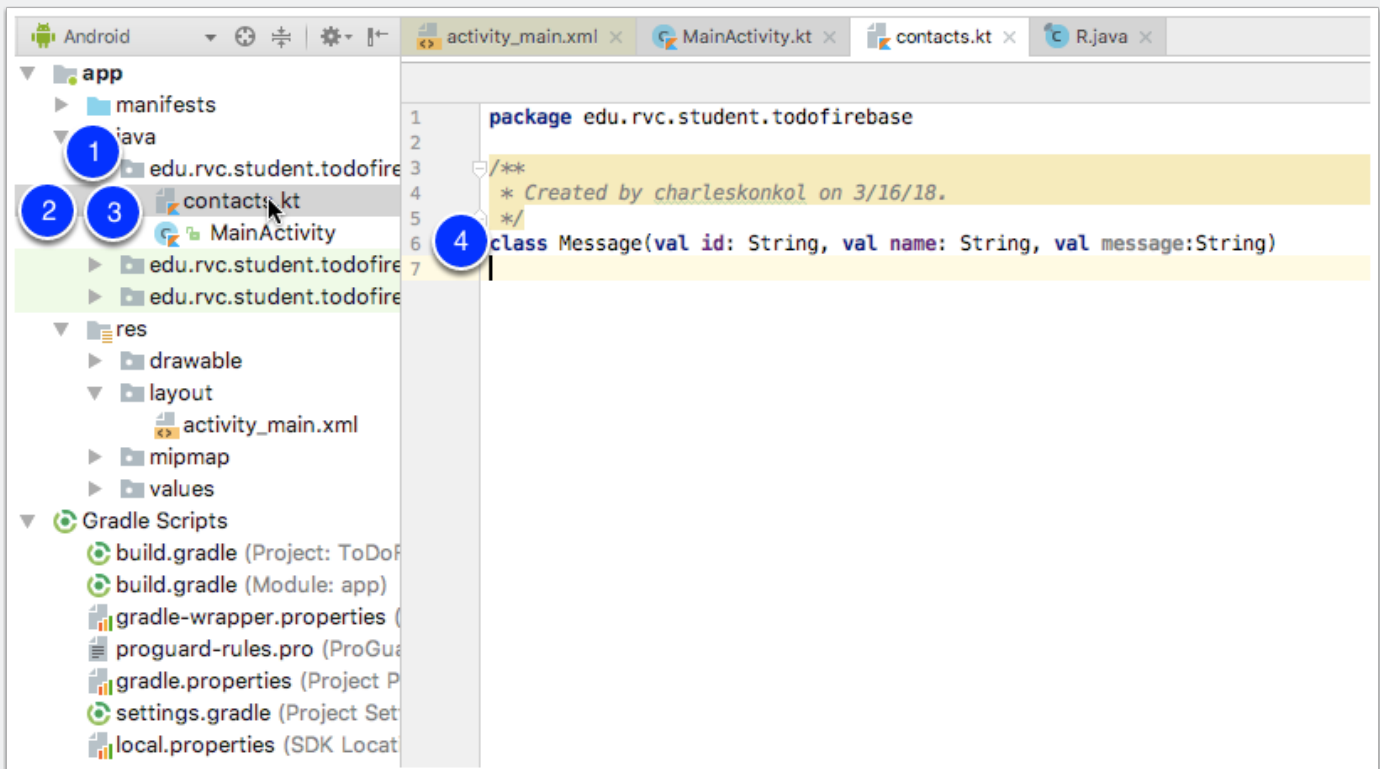
1. Go Back to Firebase in your web browser
2. Click on **Data**
3. View **message: "Hello World!"** that was just written



Android Emulator - ToDoList using Firebase

contacts.kt - Add file > contacts.kt

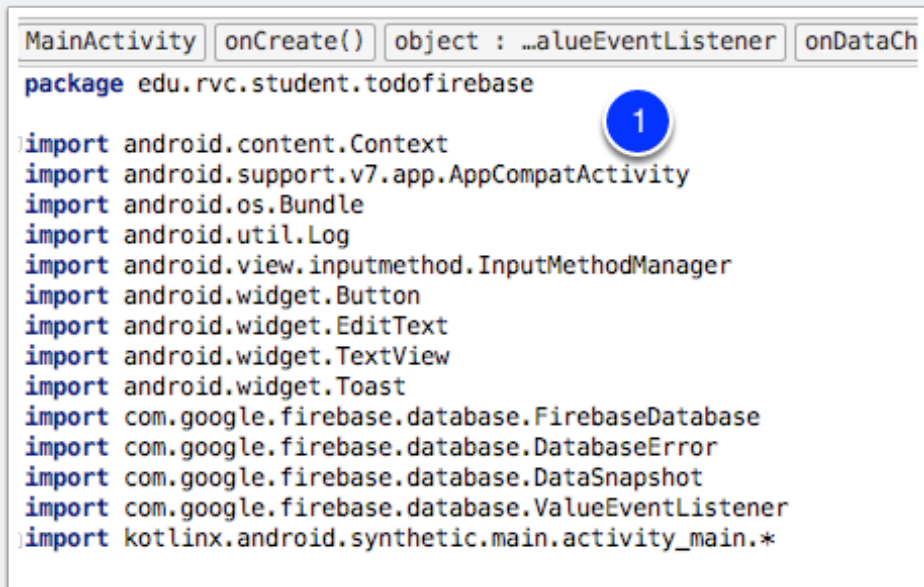
1. Right-Click on edu.rvc.stud... under java folder and select
2. New > Kotlin File Class
3. Add file > contacts.kt
4. Add Class Message on line 6 below



Android Emulator - ToDoList using Firebase

MainActivity.kt - Add Code

Make sure these imports are added to MainActivity.kt



```
MainActivity | onCreate() | object : ValueEventListener | onDataChange  
package edu.rvc.student.todofirebase  
  
import android.content.Context  
import android.support.v7.app.AppCompatActivity  
import android.os.Bundle  
import android.util.Log  
import android.view.inputmethod.InputMethodManager  
import android.widget.Button  
import android.widget.EditText  
import android.widget.TextView  
import android.widget.Toast  
import com.google.firebase.database.FirebaseDatabase  
import com.google.firebase.database.DatabaseError  
import com.google.firebase.database.DataSnapshot  
import com.google.firebase.database.ValueEventListener  
import kotlinx.android.synthetic.main.activity_main.*
```

Android Emulator - ToDoList using Firebase

MainActivity.kt - Add Code

1. Remove the test code earlier
2. Add Remaining Code

MainActivity.kt - Add Code

1. Add two functions **ref.addValueEventListener** & **hidekeyboard**

Android Emulator - ToDoList using Firebase

2. Test & confirm data is writing to database online
3. Submit to Github

```
// Listen and show data changes
1 ref.addValueEventListener(object : ValueEventListener {
    override fun onDataChange(dataSnapshot: DataSnapshot) {
        messages.text = ""
        val children = dataSnapshot.children
        children.forEach {
            println("data: " + it.toString())
            if (messages.text.toString() != "") {
                messages.text = messages.text.toString() + "\n" + "Task: " + it.child( p0: "name").value.toString() + " " + "Desc: " + it.child( p0: "message").value.toString()
            } else {
                messages.text = "My Tasks"
            }
            messages.text = messages.text.toString() + "\n" + "Task: " + it.child( p0: "name").value.toString() + " " + "Desc: " + it.child( p0: "message").value.toString()
        }
    }

    override fun onCancelled(error: DatabaseError) {
        // Failed to read value
        Log.w( tag: "Message", msg: "Failed to read value.", error.toException())
    }
})

// function to hide keyboard goes right before the last right bracket of Class MainActivity
//import android.content.Context
//import android.view.inputmethod.InputMethodManager
2 fun hideKeyboard() {
    try {
        val imm = getSystemService(Context.INPUT_METHOD_SERVICE) as InputMethodManager
        imm.hideSoftInputFromWindow(currentFocus!!.windowToken, flags: 0)
    } catch (e: Exception) {
        // TODO: handle exception
    }
}
```