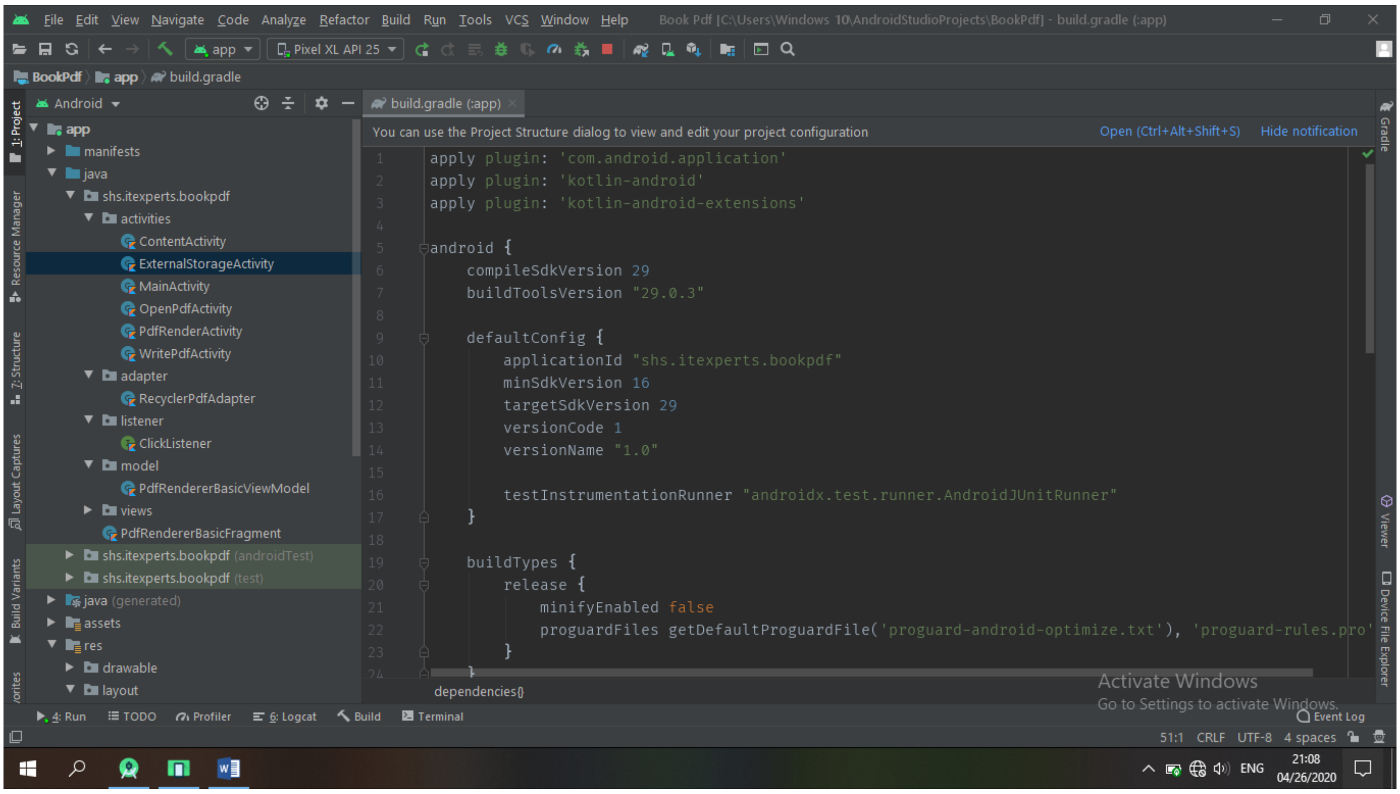
Androidda pdf fayllar ustida ishlash.

1.Loyihani yaratamiz.

2. Gradle ba’zi kutubxonalarni yuklab olamiz.

3. Ilovani code qismni boshlaymiz.



You can use the Project Structure dialog to view and edit your project configuration   Open (Ctrl+Alt+Shift+S)   Hide notification

```
apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply plugin: 'kotlin-android-extensions'

android {
    compileSdkVersion 29
    buildToolsVersion "29.0.3"

    defaultConfig {
        applicationId "shs.itexperts.bookpdf"
        minSdkVersion 16
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
dependencies{}
```

Build.gradle

```
apply plugin: 'com.android.application'
apply plugin: 'kotlin-android'
apply plugin: 'kotlin-android-extensions'

android {
    compileSdkVersion 29
    buildToolsVersion "29.0.3"

    defaultConfig {
        applicationId "shs.itexperts.bookpdf"
        minSdkVersion 16
        targetSdkVersion 29
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }

    kotlinOptions {
        jvmTarget = '1.8'
    }

}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'org.jetbrains.kotlinx:kotlinx-coroutines-android:1.3.3'

    implementation 'androidx.fragment:fragment-ktx:1.2.4'
    implementation 'androidx.appcompat:appcompat:1.1.0'


    implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:2.2.0"
    implementation "androidx.lifecycle:lifecycle-livedata-ktx:2.2.0"
    androidTestImplementation 'androidx.arch.core:core-testing:2.1.0'

    implementation 'androidx.core:core-ktx:1.2.0'
    implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'androidx.test.ext:junit:1.1.1'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'
    implementation 'androidx.recyclerview:recyclerview:1.1.0'

    // write pdf
    implementation 'com.itextpdf:itextg:5.5.10'
    // read pdf
    implementation 'com.github.barteksc:android-pdf-viewer:2.8.2'
    implementation 'com.google.android.material:material:1.1.0'
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="shs.itexperts.bookpdf">

    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="false"
        android:icon="@drawable/pdf_icon"
        android:label="@string/app_name"
        android:roundIcon="@drawable/pdf_icon"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".activities.OpenPdfActivity"
            android:label="Open Pdf"/>
        <activity android:name=".activities.ExternalStorageActivity"
            android:label="Pdf List"/>
        <activity android:name=".activities.WritePdfActivity"
            android:label="Write Pdf"/>
        <activity android:name=".activities.PdfRenderActivity" />
        <activity android:name=".activities.ContentActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".activities.MainActivity" />
    </application>

</manifest>
```

ContentActivity.kt

```
package shs.itexperts.bookpdf.activities

import android.Manifest
import android.content.Intent
import android.content.pm.PackageManager
import android.os.Bundle
import android.util.Log
import android.widget.Button
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import shs.itexperts.bookpdf.R


class ContentActivity : AppCompatActivity() {

    private lateinit var button: Button
    private lateinit var button1: Button
    private lateinit var button2: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_content)

        if (checkPermission()) {
            Toast.makeText(this,"Ruxsat berilgan", Toast.LENGTH_LONG).show()
        } else {
            requestPermission() // Code for permission
        }


        button = findViewById(R.id.main)
        button1 = findViewById(R.id.main1)
        button2 = findViewById(R.id.main2)

        button.setOnClickListener {
            val intent = Intent(this@ContentActivity, MainActivity::class.java)
```

```kotlin
            startActivity(intent)
        }
        button1.setOnClickListener {
            val intent = Intent(
                this@ContentActivity,
                ExternalStorageActivity::class.java
            )
            startActivity(intent)
        }
        button2.setOnClickListener {
            val intent = Intent(
                this@ContentActivity,
                WritePdfActivity::class.java
            )
            startActivity(intent)
        }

    }

    private fun checkPermission(): Boolean {
        val result = ContextCompat.checkSelfPermission(
            this@ContentActivity,
            Manifest.permission.WRITE_EXTERNAL_STORAGE
        )
        return result == PackageManager.PERMISSION_GRANTED
    }

    private fun requestPermission() {
        if (ActivityCompat.shouldShowRequestPermissionRationale(
                this@ContentActivity,
                Manifest.permission.WRITE_EXTERNAL_STORAGE
            )
        ) {
            Toast.makeText(
                this@ContentActivity,
                "Write External Storage permission allows us to do store images. Please allow this permission in App 
Settings.",
                Toast.LENGTH_LONG
            ).show()
        } else {
            ActivityCompat.requestPermissions(
                this@ContentActivity,
                arrayOf(Manifest.permission.WRITE_EXTERNAL_STORAGE),
                1
            )
        }
    }

    override fun onRequestPermissionsResult(
        requestCode: Int,
        permissions: Array<out String>,
        grantResults: IntArray
    ) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults)
        when (requestCode) {
            1 ->
            if (grantResults.isNotEmpty() && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                Log.e("value", "Permission Granted, Now you can use local drive .")
            } else {
                Log.e("value", "Permission Denied, You cannot use local drive .")
            }
        }
    }
}
```

WritePdfActivity.kt

```
package shs.itexperts.bookpdf.activities

import android.Manifest
import android.content.pm.PackageManager
import android.os.Build
import android.os.Bundle
import android.os.Environment
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import com.itextpdf.text.Document
import com.itextpdf.text.Paragraph
import com.itextpdf.text.pdf.PdfWriter
import shs.itexperts.bookpdf.R
import java.io.File
import java.io.FileOutputStream
import java.text.SimpleDateFormat
import java.util.*

class WritePdfActivity : AppCompatActivity() {

    private lateinit var mTextEt: EditText
    private lateinit var mSaveBtn: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_write_pdf)

        mTextEt = findViewById(R.id.textEt)
        mSaveBtn = findViewById(R.id.saveBtn)


        mSaveBtn.setOnClickListener {
            //we need to handle runtime permission for devices with marshmallow and above
            if (Build.VERSION.SDK_INT > Build.VERSION_CODES.M) {
                //system OS >= Marshmallow(6.0), check if permission is enabled or not
                if (checkSelfPermission(Manifest.permission.WRITE_EXTERNAL_STORAGE) ==
                    PackageManager.PERMISSION_DENIED
                ) {
                    //permission was not granted, request it
                    val permissions =
                        arrayOf(Manifest.permission.WRITE_EXTERNAL_STORAGE)
                    requestPermissions(permissions, 1)
```

```kotlin
            } else {
                //permission already granted, call save pdf method
                savePdf()
            }
        } else {
            //system OS < Marshmallow, call save pdf method
            savePdf()
        }
    }

}


private fun savePdf() {
    //create object of Document class
    val mDoc = Document()
    //pdf file name
    val mFileName: String = SimpleDateFormat(
        "yyyyMMdd_HHmmss",
        Locale.getDefault()
    ).format(System.currentTimeMillis())
    //pdf file path
    val file = File(Environment.getExternalStorageDirectory(), "Pdf App")
    if (!file.exists()){
        file.mkdirs()
    }
    val mFilePath: String = file.path+ "/"+mFileName+".pdf"
    try {
        //create instance of PdfWriter class
        PdfWriter.getInstance(mDoc, FileOutputStream(mFilePath))
        //open the document for writing
        mDoc.open()
        //get text from EditText i.e. mTextEt
        val mText : String = mTextEt.text.toString()

        //add author of the document (optional)
        mDoc.addAuthor("Atif Pervaiz")

        //add paragraph to the document
        mDoc.add(Paragraph(mText))

        //close the document
        mDoc.close()
        //show message that file is saved, it will show file name and file path too
        Toast.makeText(this, "$mFileName.pdf\nis saved to\n$mFilePath", Toast.LENGTH_SHORT)
            .show()
    } catch (e: Exception) {
        //if any thing goes wrong causing exception, get and show exception message
        Toast.makeText(this, e.message, Toast.LENGTH_SHORT).show()
    }
}

override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<out String>,
    grantResults: IntArray
    ) {
        when (requestCode) {
            1 -> {
                if (grantResults.isNotEmpty() && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                    //permission was granted from popup, call savepdf method
                    savePdf()
                } else {
                    //permission was denied from popup, show error message
                    Toast.makeText(this, "Permission denied...!", Toast.LENGTH_SHORT).show()
                }
            }
        }
    }
}
```

ExternalStorageActivity.kt

```
package shs.itexperts.bookpdf.activities

import android.content.Intent
import android.os.Bundle
import android.os.Environment
import android.util.Log
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import shs.itexperts.bookpdf.R
import shs.itexperts.bookpdf.adapter.RecyclerPdfAdapter
import shs.itexperts.bookpdf.listener.ClickListener
import java.io.File

class ExternalStorageActivity : AppCompatActivity() {

    private val fileList = ArrayList<File>()
    private lateinit var rv : RecyclerView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_external_storage)

        // Recycler View ni topib olamiz va unga layout beriladi
        rv = findViewById(R.id.rv_pdf)
        rv.setHasFixedSize(true)
        rv.layoutManager = LinearLayoutManager(this)

        val dir = File(Environment.getExternalStorageDirectory().absolutePath+"/Pdf App/")
        Log.d("absolutePath",dir.absolutePath)

        // searchDir() method orqali pdf fayllarni topib fileList ga ozlashtiramiz
        searchDir(dir)
        // adapter class chaqirilib file royxati beriladi
        val adapter = RecyclerPdfAdapter(this,fileList)
        rv.adapter = adapter

        adapter.setOnItemClickListener(object : ClickListener {
            override fun onItemClickListener(position: Int) {
                val intent = Intent(this@ExternalStorageActivity, OpenPdfActivity::class.java)
                intent.putExtra("pdf",fileList[position].absolutePath)
                startActivity(intent)
            }
        })
    }

    // xotira ichidagi ".pdf" ma'lumotlarni topadi.
    private fun searchDir(dir: File) {
        val pdfPattern = ".pdf"
        val arrayOfFiles = dir.listFiles()
        if (arrayOfFiles != null) {
            for (i in arrayOfFiles.indices) {
                if (arrayOfFiles[i].isDirectory) {
                    searchDir(arrayOfFiles[i])
                } else {
                    if (arrayOfFiles[i].name.endsWith(pdfPattern)) {
                        fileList.add(arrayOfFiles[i])
                    }
                }
            }
        }
    }

}
```

OpenPdfActivity.kt

```
package shs.itexperts.bookpdf.activities

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast
import com.github.barteksc.pdfviewer.PDFView
import shs.itexperts.bookpdf.R
import java.io.File

class OpenPdfActivity : AppCompatActivity() {

    private lateinit var pdfView : PDFView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_open_pdf)

        pdfView = findViewById(R.id.pdfOpen)

        val b   = intent.extras

        val uriPath  = b?.getString("pdf").toString()
        Toast.makeText(this,uriPath, Toast.LENGTH_LONG).show()
        val dir = File(uriPath)
        // pdfView ga fayl adresi berilmoqda
        pdfView.fromFile(dir).load()

    }
}
```