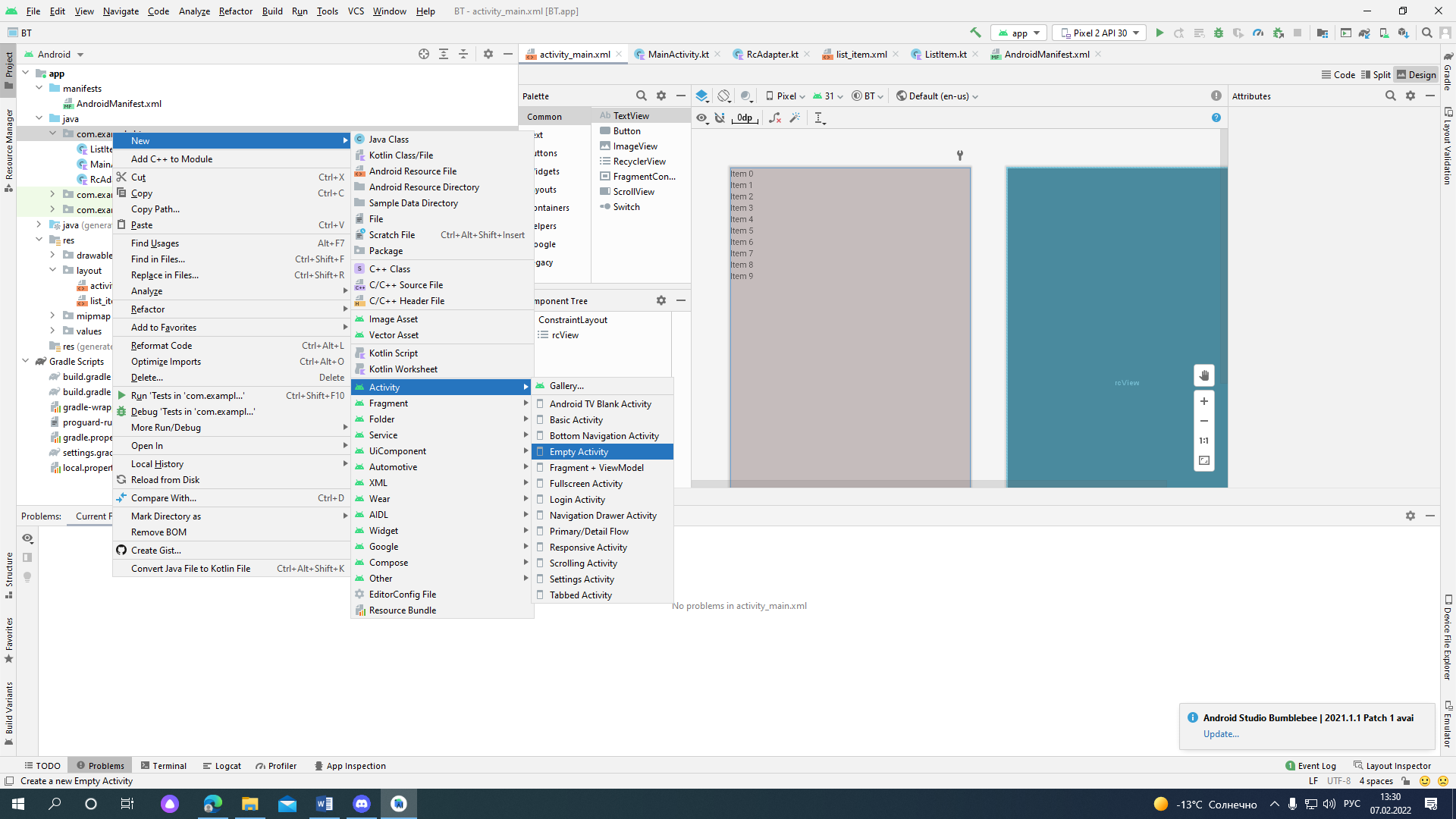
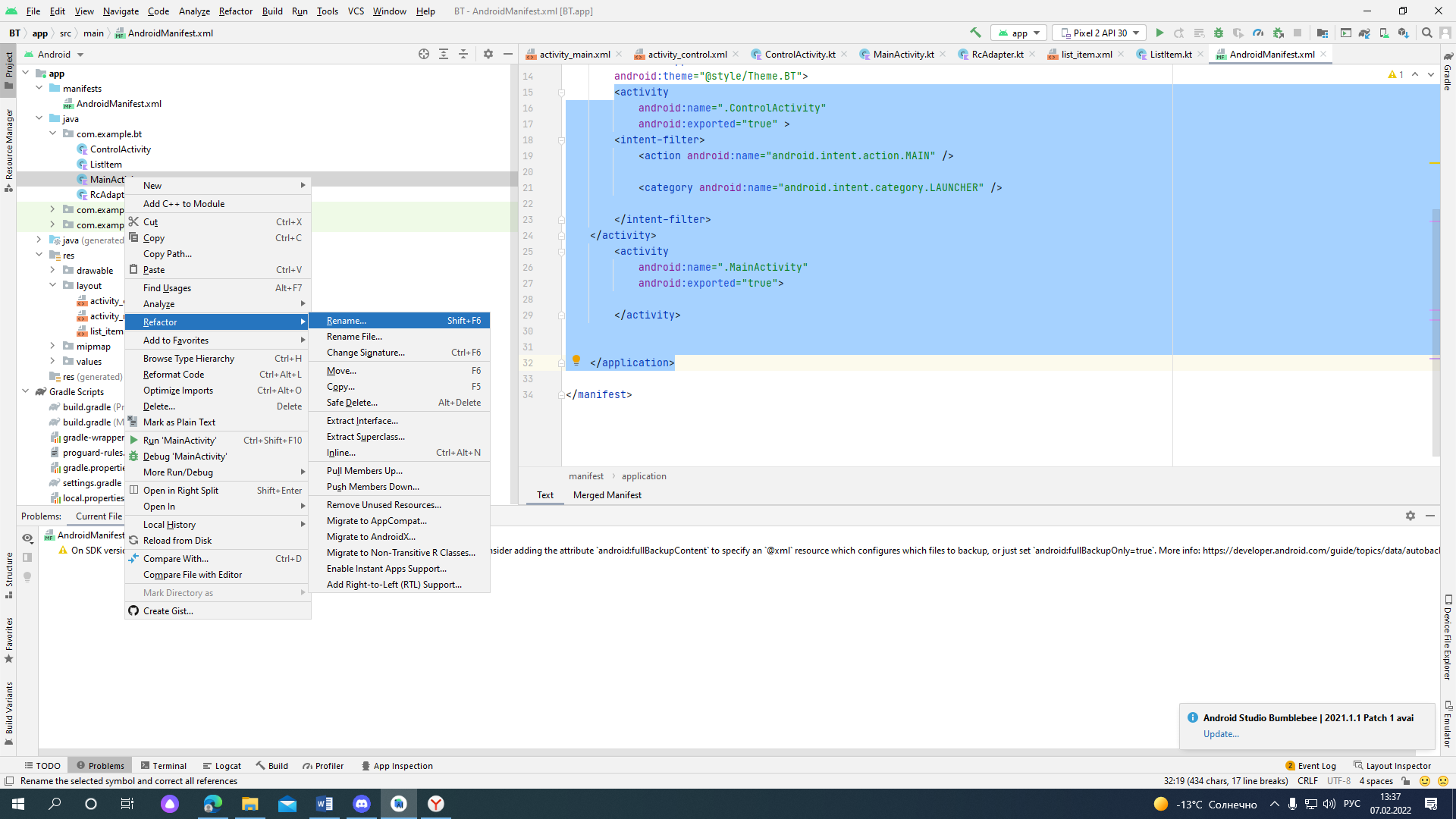
**Создание приложения по получения на телефон перечня устройств BlueTooth к которым можно подключиться и подключение к одному из них (часть 2)**

В этой части сделаем стартовое активити, где будет высвечиваться адрес, по которому будет происходить подключение. То активити, что делали в первой части мы будем вызывать из стартового.

1. Создаем новое активити с именем ControlActivity



1. Делаем его стартовым, для этого идем в AndroidManifest.
2. И переносим Intent-filter в первое активити, вот так (изменения выделены)
3. <activity  
    android:name=".ControlActivity"  
    android:exported="true" >//здесь убрали черту закрытия!!!  
    **<intent-filter>  
    <action android:name="android.intent.action.MAIN" />  
     
    <category android:name="android.intent.category.LAUNCHER" />  
     
    </intent-filter>  
   </activity>** <activity  
    android:name=".MainActivity"  
    android:exported="true">  
     
    </activity>  
     
     
   </application>
4. Назовем теперь MainActivity как BTList



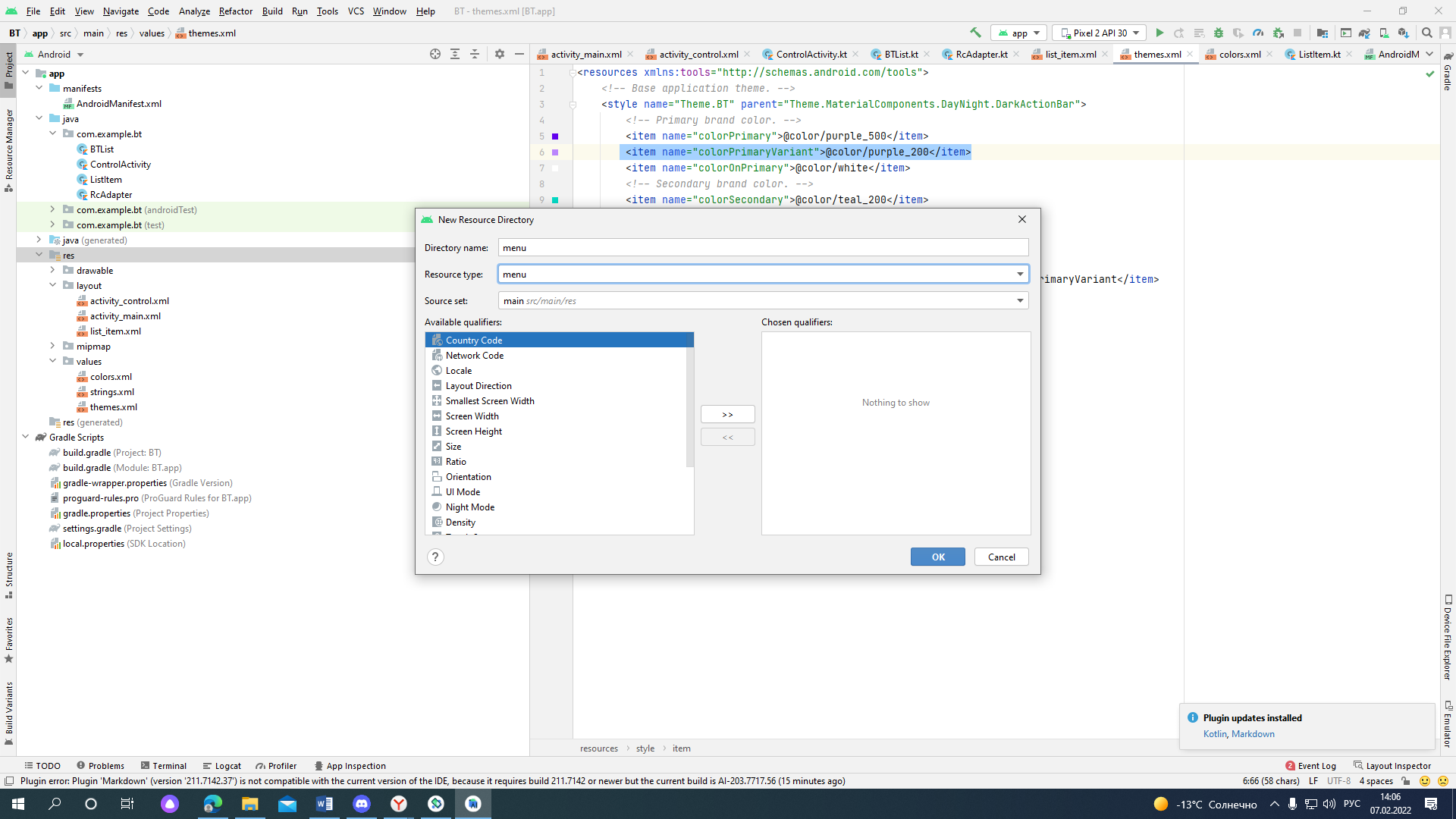
1. Немного изменим внешний вид стартового активити. В папке value проекта в папке themes (ночную тему файл удалим) в оставшийся файл темы добавим удаление со стартового экрана статусной строки

<item name="android:windowFullscreen">true</item>

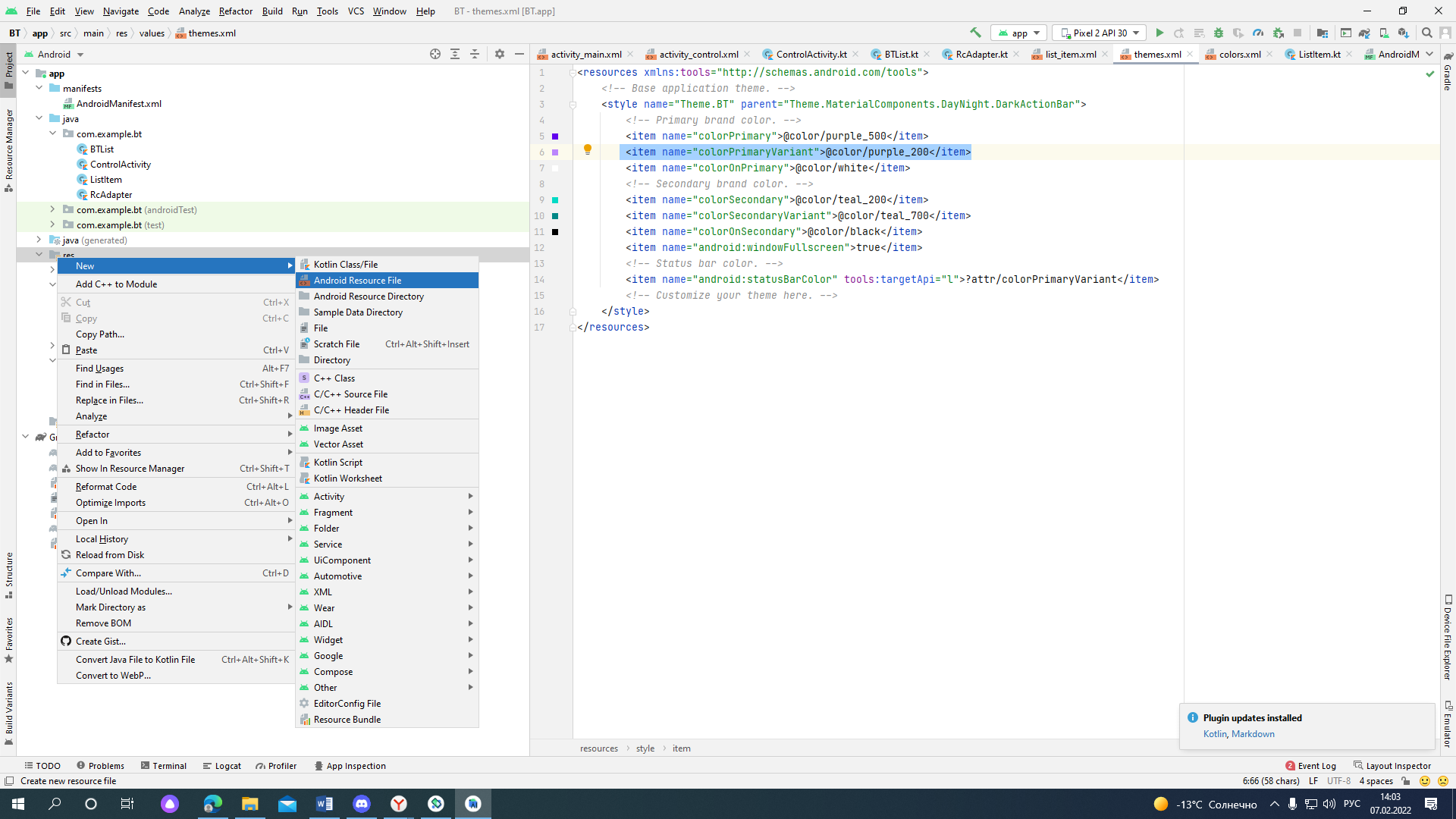
1. И заменим цвет заголовка изменив

<item name="colorPrimaryVariant">@color/**purple\_200**</item>

1. На строке заголовка изменим название , добавим туда кнопку подключения и меню Для этого идем создаем папку для ресурса



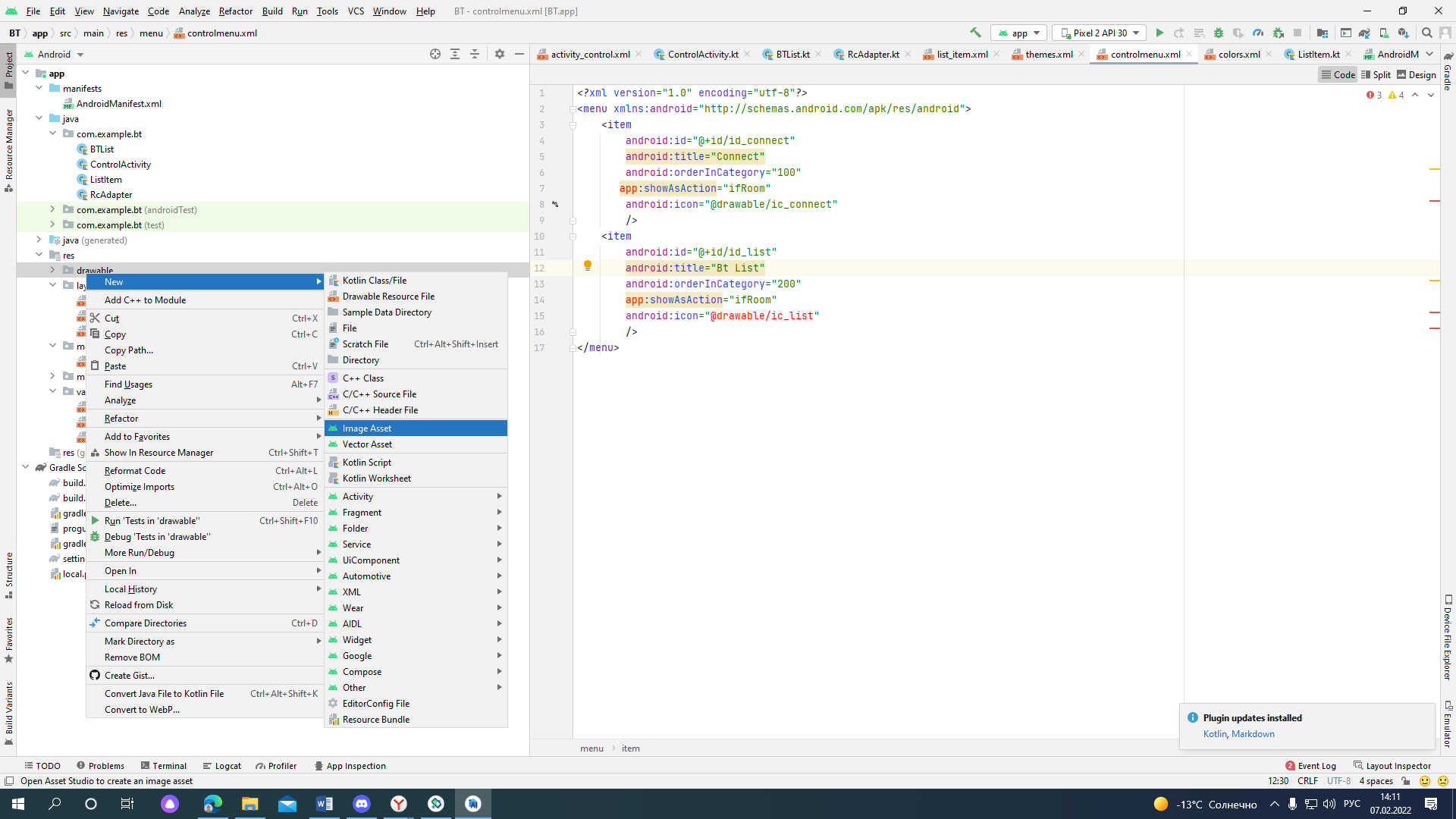
1. И затем в этой папке новый ресурс меню ControlMenu



И в этом файле мы формируем меню

<item  
 android:id="@+id/id\_connect"  
 android:title="Connect"  
 android:orderInCategory="100"  
 app:showAsAction="ifRoom"  
 android:icon="@drawable/ic\_connect"  
 />  
<item  
 android:id="@+id/id\_list"  
 android:title="Bt List"  
 android:orderInCategory="200"  
 app:showAsAction="ifRoom"  
 android:icon="@drawable/ic\_list"  
 />

При этом не забудьте создать графические иконки для пунктов меню с именами ic\_connect и ic\_list



1. Теперь наше меню нужно передать в стартовое активи. Для этого создадим в файле ControlActivity

override fun onCreateOptionsMenu(menu: Menu?): Boolean {  
 *menuInflater*.inflate(R.menu.*controlmenu*, menu)  
 return super.onCreateOptionsMenu(menu)  
 }

1. И затем прописываем что будет при нажатии кнопок меню в отдельной функции

override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 if(item.*itemId* == R.id.*id\_list*){  
  
 } else if(item.*itemId* == R.id.*id\_connect*){  
  
 }  
 return super.onOptionsItemSelected(item)  
 }

1. При этом нужно добавить в class ControlActivity

private lateinit var binding: ActivityControlBinding  
private lateinit var actListLauncher: ActivityResultLauncher<Intent>

1. и в функцию onCreate

binding = ActivityControlBinding.inflate(*layoutInflater*)  
setContentView(binding.*root*)

1. Для получения результата, какое устройство будет выбрано, создаем функцию также в файле ControlActivity(берем в нашей функции actListLauncher и его регистрируем и просмотри его только в Logcat, пока нет кода для получения устройств)

private fun onBtListResult(){  
 actListLauncher = registerForActivityResult(  
 ActivityResultContracts.StartActivityForResult())**{** if(**it**.*resultCode* == *RESULT\_OK*){  
 Log.d("MyLog","Name: ${(**it**.*data*?.getSerializableExtra(BTList.DEVICE\_KEY) as ListItem).name}")  
 }  
 **}** }

пока нет адреса устройства создадим обьект который будем принимать, создадим его в файле BTList

companion object{  
 const val DEVICE\_KEY = "device\_key"  
 }

1. Инициализируем функцию onBtListResult(), прописав ее в onCreate

override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 binding = ActivityControlBinding.inflate(*layoutInflater*)  
 setContentView(bin**ding.*root*)  
 onBtListResult**()  
}

1. И теперь мы ее можем прописать

override fun onOptionsItemSelected(item: MenuItem): Boolean {  
 if(item.*itemId* == R.id.*id\_list*){  
 **actListLauncher.launch(Intent(this, BTList::class.*java*))** } else if(item.*itemId* == R.id.*id\_connect*){  
  
 }  
 return super.onOptionsItemSelected(item)  
 }

1. Пропишем теперь слушатели событий при нажатии на кнопки меню в RCAdapter

interface Listener{  
 fun onClick(item: ListItem)  
 }

1. и подключаем его в класс BTList

class BTList : AppCompatActivity(), **RcAdapter.Listener** {

и имплементируем его функцию, которая создаст функцию его нажатия автоматически

override fun onClick(item: ListItem) {  
 *TODO*("Not yet implemented")  
}

1. для этого идем в RcAdapter и указываем что мы ходим передать

class RcAdapter**(private val listener: Listener)**: ListAdapter <ListItem,RcAdapter.ItemHolder>(ItemComparator()) {

1. и добавляем его в функцию setData

fun setData(item: ListItem, **listener: Listener**) = *with*(binding)**{**

1. и добавляем сюда слушатель нажатий

fun setData(item: ListItem, listener: Listener) = *with*(binding) **{** tvName.*text* = item.name  
 tvMac.*text* = item.mac  
 **itemView.setOnClickListener {  
 listener.onClick(item)  
 }  
}**

1. Чтобы обратно передать данные в BtList по событию OnClick пропишем

val i= Intent().apply **{** putExtra(DEVICE\_KEY, item)  
**}**setResult(*RESULT\_OK*, i)  
finish()

1. Чтобы наш item воспринимался как тип byte, заходим в класс ListItem и дописываем преобразрвание

data class ListItem(  
 var name: String,  
 var mac: String  
**): Serializable**

1. в функции надо init(){ дописать

private fun init(){  
 val btManager = getSystemService(Context.*BLUETOOTH\_SERVICE*) as BluetoothManager  
 btAdapter = btManager.*adapter* adapter = RcAdapter**(this)**

**Проверка возможна только если подключено реальное устройство у которого есть уже список подключенных устройств**

И при открытом Logcat