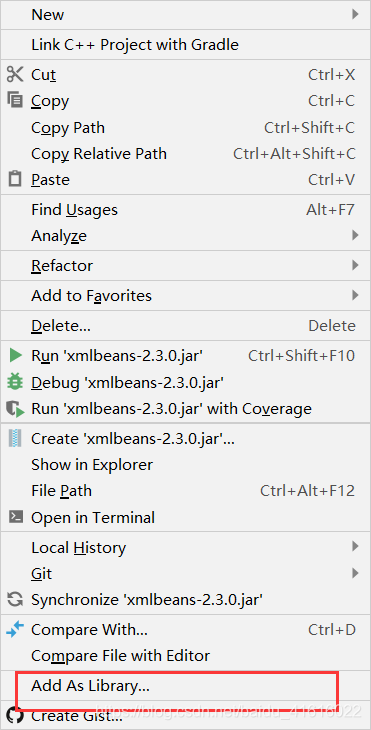
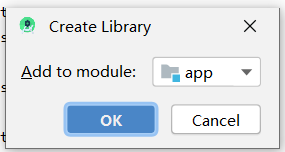
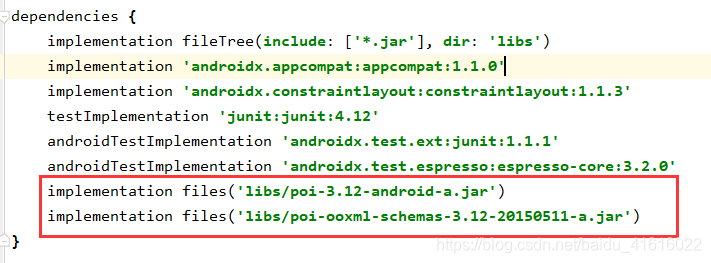
**Android stdio使用poi读取、创建、另存Excel，支持xlsx和部分xls格式**

**在这里，我使用的poi是3.12版本的，它不是Apache官方poi包**

下载jar包后，将其放进libs文件夹里面，选中两个jar包，鼠标右键点击add As library…添加为库，[下载jar点击这里](https://download.csdn.net/download/baidu_41616022/12222443)(此库收费,到github下载)  
  
点击OK  
  
打开build.gradle文件，添加完依赖后就可以开始 敲键盘  
  
[开发文档可以参考这里，需要注意的是我们使用的poi包不是Apache软件基金会的，部分方法及其功效会有所不同！！！](https://blog.csdn.net/qq_42651904/article/details/88221392)

**这里简要介绍我们用到的方法：**

**创建.xlsx格式文件对象**

workbook = new XSSFWorkbook(inputStream);

* 1

**创建.xls格式文件对象**

workbook = new HSSFWorkbook(inputStream);

* 1

**获取工作表的对象**

Sheet sheetAt = workbook.getSheetAt(0);

* 1

**获取工作表的行**

Row row = sheetAt.getRow(0);

* 1

**获取实际单元格数**

int physicalNumberOfCells = row.getPhysicalNumberOfCells();

* 1

**获取工作表的单元格**

Row.getCell(i)；

* 1

**获得单元格格式**

Cell.getCellType()；

* 1

**获取单元格类型**

Cell.getBooleanCellValue();//获取布尔类型的单元格

Cell.getNumericCellValue();//获取数字类型的单元格

Cell.getDateCellValue();//获取日期类型的单元格

Cell.getNumericCellValue();//获取数值类型的单元格

Cell.getStringCellValue();//获取字符串类型的单元格

* 1
* 2
* 3
* 4
* 5

**获取实际行数**

SheetAt.getPhysicalNumberOfRows();

* 1

**创建工作表的名字**

XSSFSheet sheet = workbook.createSheet(WorkbookUtil.createSafeSheetName("Sheet1"));

* 1



**创建行**

Row row = sheet.createRow(int i);

* 1

**创建列**

Cell cell = row.createCell(int i);

* 1

**将需要添加到Excel的文本添加到对应的Cell**

Cell.setCellValue((String) map.get(j));

* 1

**将数据写入文件并保存在指定文件夹**

OutputStream outputStream = context.getContentResolver().openOutputStream(Uri uri);

XSSFWorkbook.write(outputStream);

* 1
* 2

**读取Excel并将其写入数据库：**

public List<Map<Integer, Object>> readExcel(Context context, Uri fileUri, String strFileUri) {

mySQLHelp = new MySQLHelp(context, "mydb.db", null, 1);

SQLiteDatabase writableDatabase = mySQLHelp.getWritableDatabase();

excelStr = strFileUri.substring(strFileUri.lastIndexOf("."));

try {

inputStream = context.getContentResolver().openInputStream(fileUri);

if (excelStr.equals(".xlsx")) workbook = new XSSFWorkbook(inputStream);

else if (excelStr.equals(".xls")) workbook = new HSSFWorkbook(inputStream);

else workbook = null;

if (workbook != null) {

Sheet sheetAt = workbook.getSheetAt(0);

Row row = sheetAt.getRow(0);

int physicalNumberOfCells = row.getPhysicalNumberOfCells();//获取实际单元格数

Map<Integer, Object> map = new HashMap<>();

for (int i = 0; i < physicalNumberOfCells; i++) {//将标题存储到map

Object cellFormatValue = getCellFormatValue(row.getCell(i));

map.put(i, cellFormatValue);

}

dataList.add(map);

int physicalNumberOfRows = sheetAt.getPhysicalNumberOfRows();//获取最大行数

int size = map.size();//获取最大列数

contentValues = new ContentValues();

for (int i = 1; i < physicalNumberOfRows; i++) {

Map<Integer, Object> map1 = new HashMap<>();

Row row1 = sheetAt.getRow(i);

if (!row1.equals(null)) {

for (int j = 0; j < size; j++) {

Object cellFormatValue = getCellFormatValue(row1.getCell(j));

map1.put(j, cellFormatValue);

System.out.println(j);

}

contentValues.put("materialID", (String) map1.get(0));

contentValues.put("materialEncoding", (String) map1.get(1));

contentValues.put("materialName", (String) map1.get(2));

contentValues.put("materialModel", (String) map1.get(3));

contentValues.put("materialSize", (String) map1.get(4));

contentValues.put("unit", (String) map1.get(5));

contentValues.put("price", (String) map1.get(6));

contentValues.put("count", (String) map1.get(7));

contentValues.put("manufacturers", (String) map1.get(8));

contentValues.put("type", (String) map1.get(9));

contentValues.put("receiptor", (String) map1.get(10));

contentValues.put("storagelocation", (String) map1.get(11));

contentValues.put("materialState", (String) map1.get(12));

writableDatabase.insert("module", null, contentValues);

} else break;

dataList.add(map1);

}

contentValues.clear();

writableDatabase.close();

}

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

return dataList;

}

private static Object getCellFormatValue(Cell cell) {

Object cellValue;

if (cell != null) {

switch (cell.getCellType()) {

case Cell.CELL\_TYPE\_BOOLEAN:

cellValue = cell.getBooleanCellValue();

break;

case Cell.CELL\_TYPE\_NUMERIC:

cellValue = String.valueOf(cell.getNumericCellValue());

break;

case Cell.CELL\_TYPE\_FORMULA:

if (DateUtil.isCellDateFormatted(cell)) {

cellValue = cell.getDateCellValue();

} else {

cellValue = cell.getNumericCellValue();

}

break;

case Cell.CELL\_TYPE\_STRING:

cellValue = cell.getStringCellValue();

break;

default:

cellValue = "";

}

} else {

cellValue = "";

}

return cellValue;

}

**读取数据库数据将其写入Excel并保存到指定路径文件夹**

public void getDataAndSave(Context context,Uri uri) {

ArrayList<Map<Integer,Object>> arrayList = new ArrayList<>();

Map<Integer,Object> m = new HashMap<>();

m.put(0,"物料ID");

m.put(1,"物料编码");

m.put(2,"名称");

m.put(3,"编号");

m.put(4,"规格");

m.put(5,"单位");

m.put(6,"单价");

m.put(7,"数量");

m.put(8,"厂家");

m.put(9,"类别");

m.put(10,"经手人");

m.put(11,"存放地点");

m.put(12,"状态");

arrayList.add(m);

mySQLHelp = new MySQLHelp(context, "mydb.db", null, 1);

SQLiteDatabase readableDatabase = mySQLHelp.getReadableDatabase();

cursor = readableDatabase.rawQuery("select \* from module", null);

while (cursor.moveToNext()) {

Map<Integer,Object> map = new HashMap<>();

String materialID = cursor.getString(cursor.getColumnIndex("materialID"));

String materialEncoding = cursor.getString(cursor.getColumnIndex("materialEncoding"));

String materialName = cursor.getString(cursor.getColumnIndex("materialName"));

String materialModel = cursor.getString(cursor.getColumnIndex("materialModel"));

String materialSize = cursor.getString(cursor.getColumnIndex("materialSize"));

String unit = cursor.getString(cursor.getColumnIndex("unit"));

String price = cursor.getString(cursor.getColumnIndex("price"));

String count = cursor.getString(cursor.getColumnIndex("count"));

String manufacturers = cursor.getString(cursor.getColumnIndex("manufacturers"));

String type = cursor.getString(cursor.getColumnIndex("type"));

String receiptor = cursor.getString(cursor.getColumnIndex("receiptor"));

String storagelocation = cursor.getString(cursor.getColumnIndex("storagelocation"));

String materialState = cursor.getString(cursor.getColumnIndex("materialState"));

map.put(0,materialID);

map.put(1,materialEncoding);

map.put(2,materialName);

map.put(3,materialModel);

map.put(4,materialSize);

map.put(5,unit);

map.put(6,price);

map.put(7,count);

map.put(8,manufacturers);

map.put(9,type);

map.put(10,receiptor);

map.put(11,storagelocation);

map.put(12,materialState);

arrayList.add(map);

}

try {

XSSFWorkbook workbook = new XSSFWorkbook();

XSSFSheet sheet = workbook.createSheet(WorkbookUtil.createSafeSheetName("Sheet1"));

Cell cell;

int size = arrayList.get(0).size();

for (int i = 0;i < arrayList.size();i++){

Row row = sheet.createRow(i);

Map<Integer, Object> map = arrayList.get(i);

for (int j = 0;j < size;j++){

cell = row.createCell(j);

cell.setCellValue((String) map.get(j));

}

}

OutputStream outputStream = context.getContentResolver().openOutputStream(uri);

workbook.write(outputStream);

outputStream.flush();

outputStream.close();

Toast.makeText(context, "另存成功", Toast.LENGTH\_SHORT).show();

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}