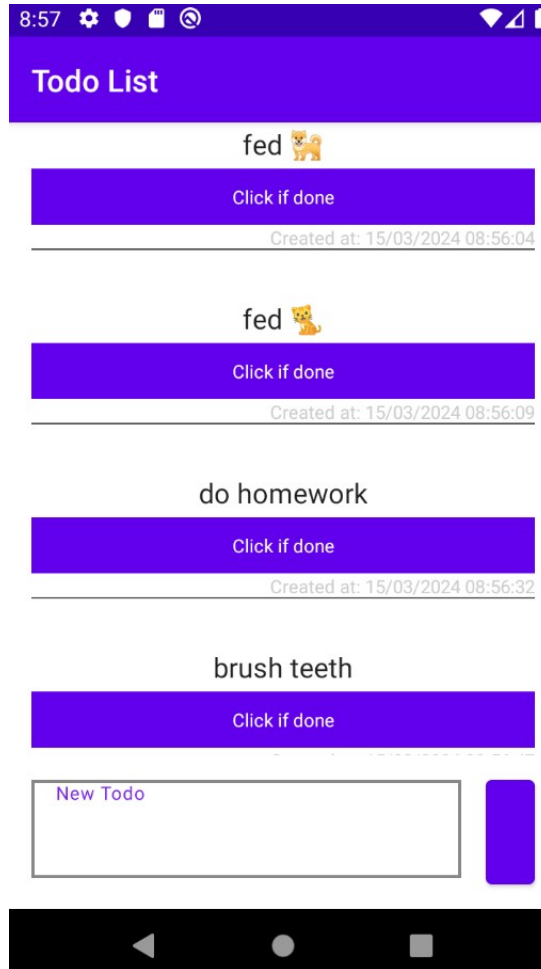


# Android приложение Список дел



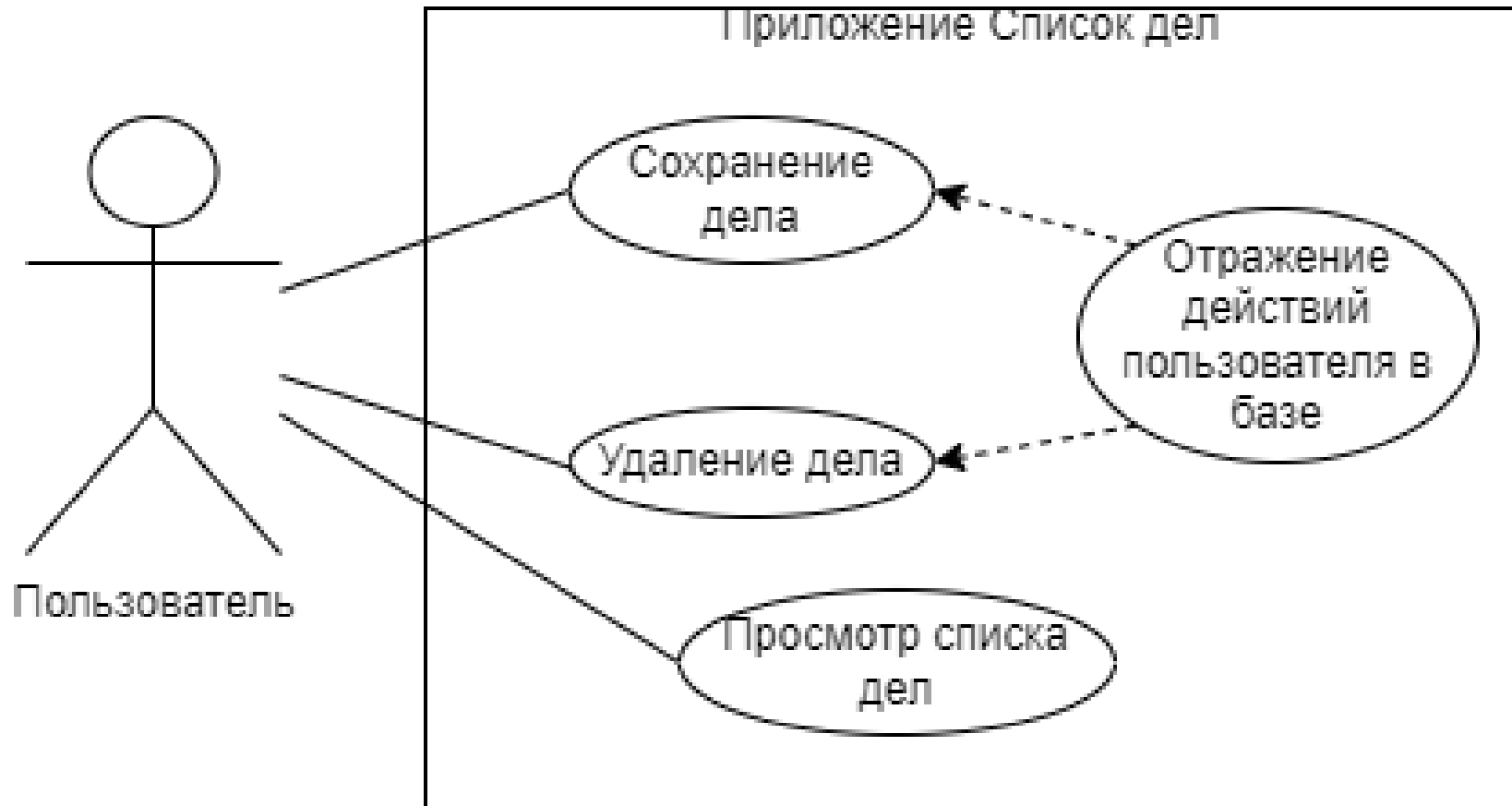
# Титульный лист

1. Титульный лист стр.2
2. Кратко о приложении стр.3
3. Диаграммы и схемы стр.4-6
4. База данных стр.7-10
5. Разметка базового элемента списка стр. 11-13
6. Главная активность стр. 14-18
7. Перспективы стр. 19

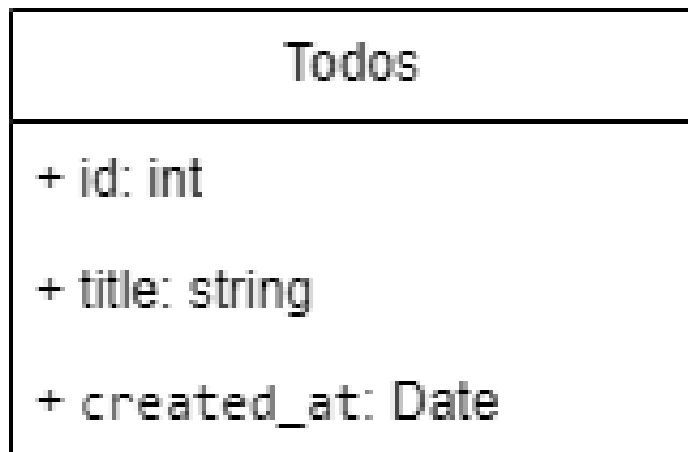
# Кратко о программе

Программа для ведения списка дел. Дела хранятся в базе данных. Также хранится дата добавления дела. Программа поддерживает добавление и удаление дел. Дело состоит из названия и/или краткого описания. Пользователь отдельный для каждого устройства, поэтому не имеет логина и пароля. Результат действий пользователя добавляется в базу. Приложение использует базу SQLite и функционирует на информации, записанной в ней.

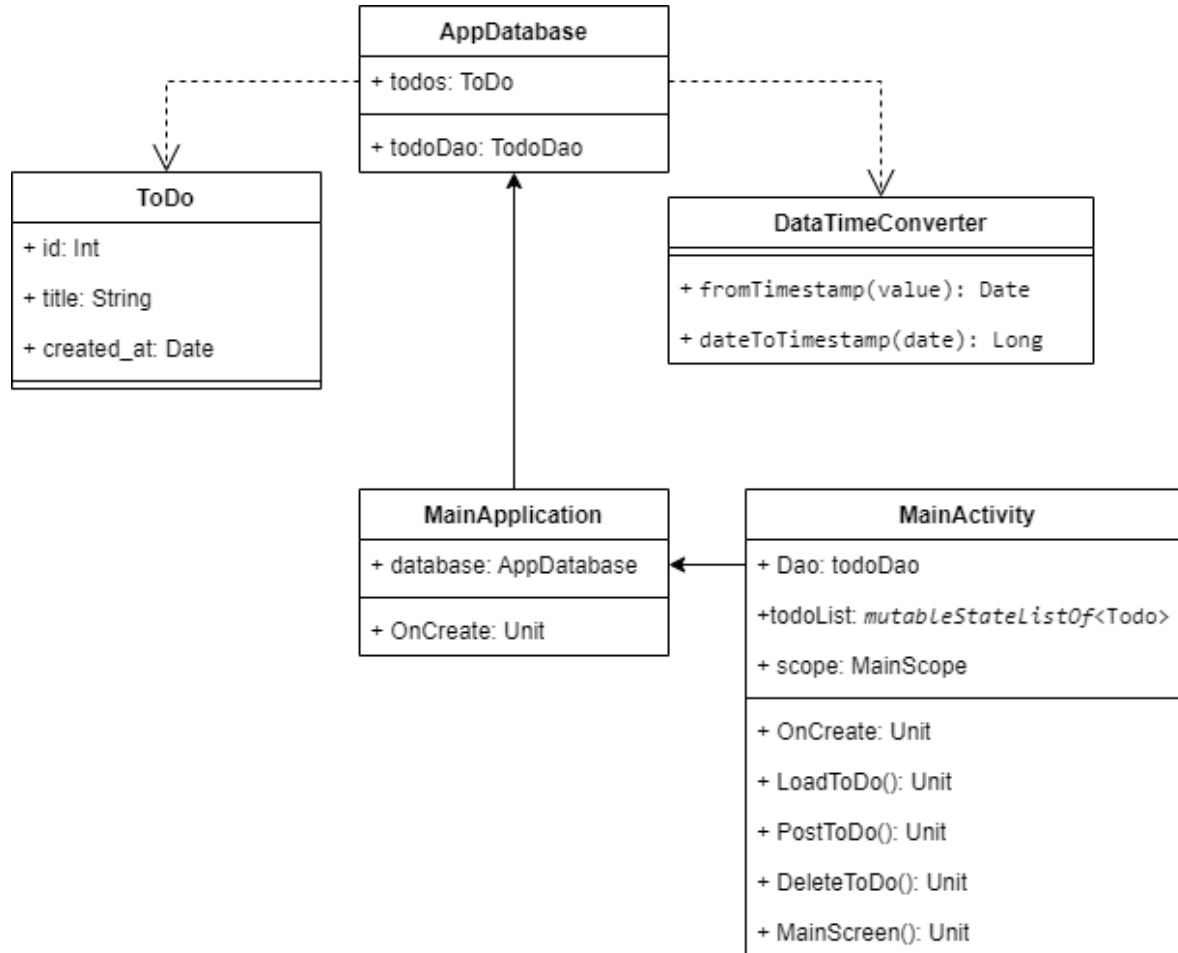
# Диаграмма использования



# Диаграмма базы данных



# Диаграмма классов



# База данных приложения

```
@Database(entities = [Todo::class], version = 1, exportSchema = false)
@TypeConverters(DateTimeConverter::class)
abstract class AppDatabase : RoomDatabase() {
    abstract fun todoDao(): TodoDao

    companion object {
        const val NAME = "todos"
    }
}
```

```
@Entity(tableName = "todos")
data class Todo(
    @PrimaryKey(autoGenerate = true)
    val id: Int = 0,
    val title: String,
    val created_at: Date = Date()
)
```

# База данных приложения

```
@Dao
interface TodoDao {
    @Query("select * from todos order by created_at asc")
    fun getAll(): MutableList<Todo>

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    fun post(todo: Todo)

    @Delete
    fun delete(todo: Todo)
}
```



# Конвертация даты между приложением и базой

```
class DateTimeConverter {  
    @TypeConverter  
    fun fromTimestamp(value: Long?): Date? {  
        return value?.let { Date(it) }  
    }  
  
    @TypeConverter  
    fun dateToTimestamp(date: Date?): Long? {  
        return date?.time?.toLong()  
    }  
}
```

# Инициализация базы данных

```
class MainApplication : Application() {  
    companion object {  
        lateinit var database: AppDatabase  
    }  
  
    override fun onCreate() {  
        super.onCreate()  
  
        database = Room.databaseBuilder(  
            applicationContext,  
            AppDatabase::class.java,  
            AppDatabase.NAME  
        ).build()  
    }  
}
```

## Разметка базового элемента списка

```
Column(  
    modifier = Modifier  
        .fillMaxWidth()  
        .padding(16.dp)  
        .clickable {  
            onClick(todo)  
        }  
) {  
    this: ColumnScope  
    Text(  
        text = todo.title,  
        fontSize = 18.sp,  
        textAlign = TextAlign.Center,  
        modifier = Modifier  
            .fillMaxWidth()  
            .padding(bottom = 4.dp)  
    )  
}
```

## Разметка базового элемента списка

```
Text(  
    text = "Click if done",  
    fontSize = 12.sp,  
    color = Color.White,  
    textAlign = TextAlign.Center,  
    modifier = Modifier.fillMaxWidth().background(Color(red: 98, green: 0, blue: 238)).padding(10.dp)  
)  
  
Text(  
    text = "${stringResource(id = "Created at:")} ${sdf.format(todo.created_at)}",  
    fontSize = 12.sp,  
    color = Color.LightGray,  
    textAlign = TextAlign.Right,  
    modifier = Modifier.fillMaxWidth().bottomBorder(1.dp, Color.DarkGray)  
)
```

## Вид базового элемента списка

fed 🐱

Click if done

Created at: 15/03/2024 04:56:09

---

# Функции главной активности

```
private fun loadToDo() {  
    scope.launch { this: CoroutineScope  
        withContext(Dispatchers.Default) { this: CoroutineScope  
            dao.getAll().forEach { todo ->  
                todoList.add(todo)  
            } } } }  
  
private fun postTodo(title: String) {  
    scope.launch { this: CoroutineScope  
        withContext(Dispatchers.Default) { this: CoroutineScope  
            dao.post(Todo(title = title))  
            todoList.clear()  
            loadToDo()  
        } } }  
  
private fun deleteTodo(todo: Todo) {  
    scope.launch { this: CoroutineScope  
        withContext(Dispatchers.Default) { this: CoroutineScope  
            dao.delete(todo)  
            todoList.clear()  
            loadToDo()  
        } } }
```

# Разметка главной активности

```
Column(  
    modifier = Modifier.clickable {  
        keyboardController?.hide()  
    }  
) { this: ColumnScope  
    TopAppBar(  
        title = { Text(text = stringResource(id = "Todo List")) }  
    )  
    LazyColumn(modifier = Modifier  
        .fillMaxWidth()  
        .weight(1f)) { this: LazyListScope  
  
        items(todoList) { this: LazyListItemScope todo ->  
            key(todo.id) {  
                TodoItem(  
                    todo = todo,  
                    onClick = { it: Todo  
                        deleteTodo(todo)  
                    }  
                }) } } }  
    }
```

# Разметка главной активности

```
Row(modifier = Modifier
    .fillMaxWidth()
    .padding(16.dp)) { this: RowScope

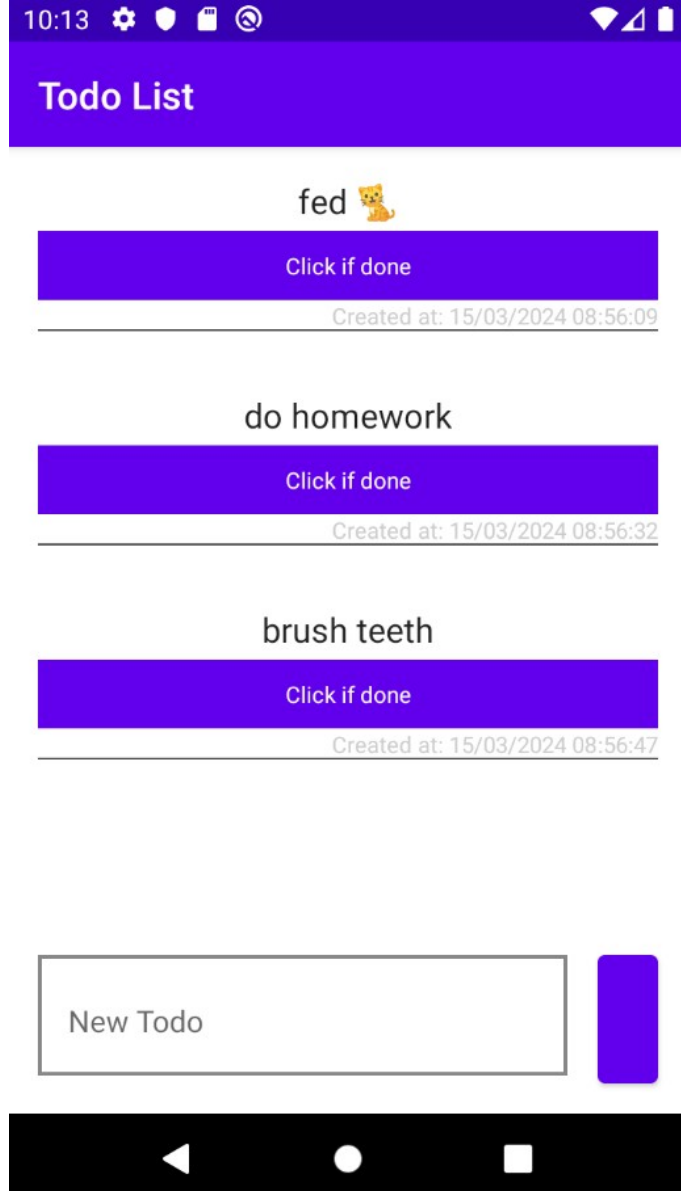
    OutlinedTextField(
        value = text,
        onValueChange = { text = it },
        colors = TextFieldDefaults.textFieldColors(
            backgroundColor = Color.Transparent,
            focusedIndicatorColor = Color.Transparent,
            unfocusedIndicatorColor = Color.Transparent
        ),
        modifier = Modifier.border(
            BorderStroke(2.dp, Color.Gray)
        ),
        label = { Text(text = stringResource(id = "New Todo")) }
    )

    Spacer(modifier = Modifier.size(16.dp))
```



# Разметка главной активности

```
Button(  
    onClick = {  
        if (text.isEmpty()) return@Button  
  
        postTodo(text)  
        text = ""  
    },  
    modifier = Modifier.align(Alignment.CenterVertically)  
) { this: RowScope  
    Text(text = stringResource(id = R.string.main_add_todo))  
}
```



Вид главной активности

# Перспективы

- 1)Расширение функционала (группировка дел, просрочка дел, сделанные дела)
- 2)Улучшение разметки
- 3)Расширение базы данных
- 4)Добавление аккаунтов
- 5)Экспорт/импорт между разными телефонами, но с одним аккаунтом

Спасибо за внимание