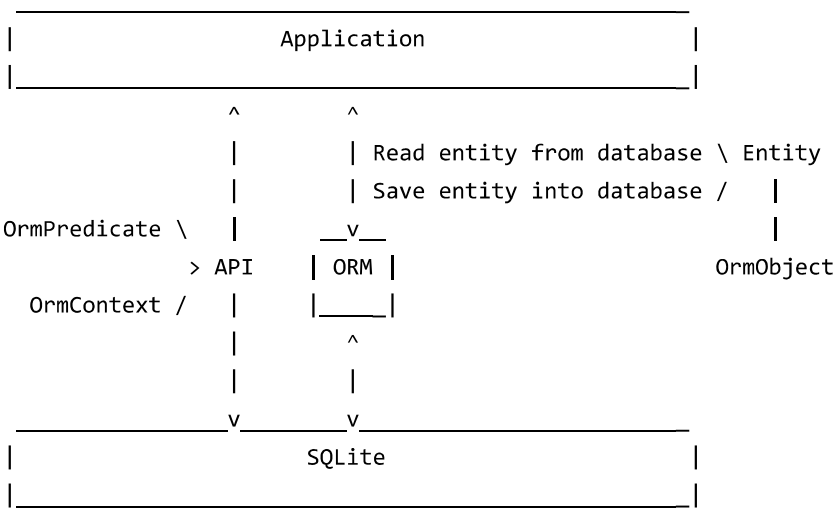
基于ORM数据库的记事本实训项目

1 ORM概述

HarmonyOS采用SQLite作为数据存储的标准配置。编写SQLite代码既繁琐又极易出错，因此HarmonyOS提供了一款基于SQLite数据库框架的对象关系映射(Object Relational Mapping, ORM)数据库作为访问SQLite数据库的高级封装。ORM数据库屏蔽了底层SQLite数据库的SQL操作，针对实体和关系提供了增删改查等一系列面向对象的操作接口，极大地降低了数据库应用程序的开发难度。此外，ORM数据库也提供包括数据库创建、升级等常用功能。



ORM包括三个重要组件：

- @Database：OrmDatabase的子类，表示数据库
- @Entity：OrmObject的子类，表示数据库中的表
- OrmContext(上下文)和OrmPredicate(谓词)：表示对数据库的操作

一个Entity对应数据库中的一张表。Entity类是SQLite表结构对Java类的映射，在Java中被看作是一个Model类，用于对数据库中的数据进行建模。其中：

- @PrimaryKey：主键
- @Column：字段
- @ForeignKey：外键
- @Index：索引

OrmContext用于表示对数据库中的表所做的操作，具体包括：

- insert：增，即插入
- delete：删，即删除
- update：改，即更新
- query：查，即查询
- where：设置谓词(OrmPredicate)的方法
  - equalTo
  - notEqualTo
  - greaterThan
  - lessThan
  - between
  - notBetween
  - or
  - and
  - limit
  - groupBy

@Database注解告诉系统这是一个ORM数据库对象，如：

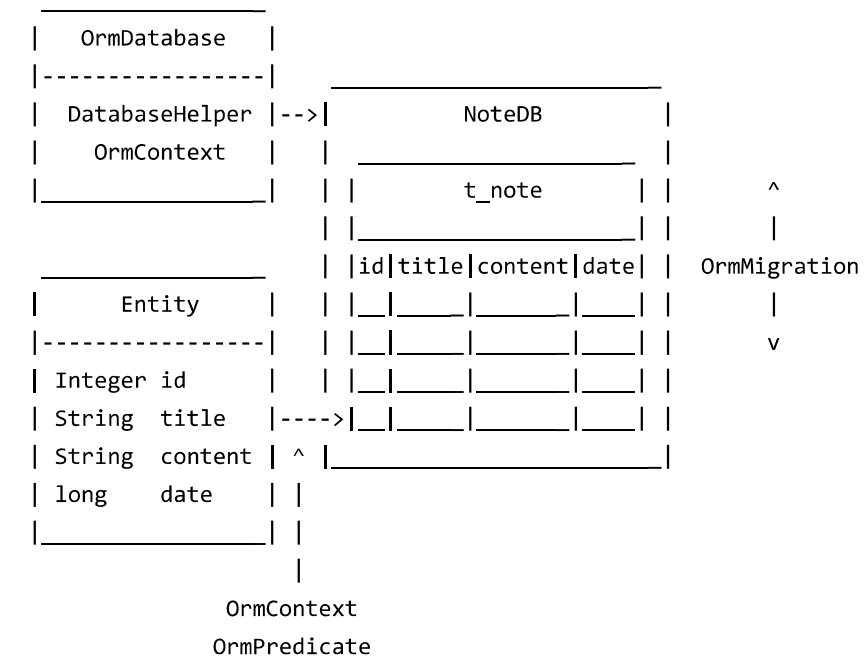
```
@Database(entities={NoteEntity.class},version=1)
public abstract class NoteDatabase extends OrmDatabase {...}
```

entities：指定该数据库包含哪些实体，多个实体以逗号隔开
version：指定数据库的版本号，作为以后数据库升级的依据

数据库类需要继承自OrmDatabase抽象类，并结合OrmContext接口完成创建：

```
DatabaseHelper databaseHelper = new DatabaseHelper(...);
OrmContext ormContext = databaseHelper.getOrmContext(
    "NoteDB", "note.db", NoteDatabase.class);
```

OrmMigration类用于数据库的升降级。



2 创建实体类、数据库类和数据访问类

2.1 启用注解

在entry\build.gradle中启用注解：

```
...
ohos {
    ...
    compileOptions {
        annotationEnabled true
    }
    ...
}
...
```

2.2 创建实体类

创建entity包：

com.hostest.orm

New

Package

com.hostest.orm.entity

创建NoteEntity类：

entity
New
Java Class
NoteEntity (Class)

在entry\src\main\java\com\hostest\orm\entity\NoteEntity.java中添加代码：

- NoteEntity类继承自OrmObject类
- 添加Entity注解，设置表名为“t_note”
- 添加变量，即表中的字段，只能使用包装类型，不能使用基本类型
- 右键——Generate...——Getter and Setter——全选——OK——Get/Set方法

```
@Entity(tableName = "t_note")
public class NoteEntity extends OrmObject {
    @PrimaryKey(autoGenerate = true)
    private Integer id;
    private String title;
    private String content;
    private long date;

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getContent() {
        return content;
    }

    public void setContent(String content) {
        this.content = content;
    }

    public long getDate() {
        return date;
    }

    public void setDate(long date) {
        this.date = date;
    }
}
```

2.3 创建数据库类

创建database包：

com.hostest.orm

New

Package

com.hostest.orm.database

创建NoteDatabase类：

database

New

Java Class

NoteDatabase (Class)

在entry\src\main\java\com\hostest\orm\database\NoteDatabase.java中添加代码：

- NoteDatabase类继承自OrmDatabase类，同时添加abstract关键字，表示其是一个抽象类
- 添加Database注解，设置实体为“NoteEntity.class”，版本为“1”

```
@Database(entities = NoteEntity.class, version = 1)
public abstract class NoteDatabase extends OrmDatabase {
}
```

2.4 创建数据访问接口

创建dao包：

com.hostest.orm

New

Package

com.hostest.orm.dao

创建NoteDao接口：

dao

New

Java Class

NoteDao (Interface)

在entry\src\main\java\com\hostest\orm\dao\NoteDao.java中添加代码：

- 添加增删改查的方法

```
public interface NoteDao {
    boolean insert(NoteEntity noteEntity);
    boolean delete(int id);
    boolean update(NoteEntity noteEntity);
    List<NoteEntity> query();
    NoteEntity query(int id);
}
```

2.5 实现数据访问接口

创建NoteDaoImpl类：

dao
    New
        Java Class
            NoteDaoImpl (Class)

在entry\src\main\java\com\hostest\orm\dao\NoteDaoImpl.java中添加代码：

- NoteDaoImpl类实现自NoteDao接口
- 右键——Generate...——Implement Methods——全选——OK——实现接口中的方法

```
public class NoteDaoImpl implements NoteDao {
    @Override
    public boolean insert(NoteEntity noteEntity) {
        return false;
    }

    @Override
    public boolean delete(int id) {
        return false;
    }

    @Override
    public boolean update(NoteEntity noteEntity) {
        return false;
    }

    @Override
    public List<NoteEntity> query() {
        return null;
    }

    @Override
    public NoteEntity query(int id) {
        return null;
    }
}
```

3 为数据访问类添加实际功能

3.1 连接数据库

为NoteDaoImpl类添加OrmContext成员，并在构造函数中完成初始化：

```
public class NoteDaoImpl implements NoteDao {
    private OrmContext ormContext;

    public NoteDaoImpl(Context contex) {
        ormContext = (new DatabaseHelper(contex)).getOrmContext(
            "NoteDB", "note.db", NoteDatabase.class);
    }
    ...
}
```

3.2 完成增删改查的具体操作

```
public class NoteDaoImpl implements NoteDao {
    private OrmContext ormContext;

    public NoteDaoImpl(Context contex) {
        ormContext = (new DatabaseHelper(contex)).getOrmContext(
            "NoteDB", "note.db", NoteDatabase.class);
    }

    @Override
    public boolean insert(NoteEntity noteEntity) {
        ormContext.insert(noteEntity);
        return ormContext.flush();
    }

    @Override
    public boolean delete(int id) {
        OrmPredicates ormPredicates = ormContext.where(
            NoteEntity.class).equalTo("id", id);
        List<NoteEntity> noteEntities = ormContext.query(ormPredicates);

        NoteEntity deletedNoteEntity = noteEntities.get(0);

        ormContext.delete(deletedNoteEntity);
        return ormContext.flush();
    }

    @Override
    public boolean update(NoteEntity noteEntity) {
        OrmPredicates ormPredicates = ormContext.where(
            NoteEntity.class).equalTo("id", noteEntity.getId());
        List<NoteEntity> noteEntities = ormContext.query(ormPredicates);

        NoteEntity updatedNoteEntity = noteEntities.get(0);
        updatedNoteEntity.setTitle(noteEntity.getTitle());
        updatedNoteEntity.setContent(noteEntity.getContent());
        updatedNoteEntity.setDate(noteEntity.getDate());

        ormContext.update(updatedNoteEntity);
        return ormContext.flush();
    }

    @Override
    public List<NoteEntity> query() {
        OrmPredicates ormPredicates = ormContext.where(
            NoteEntity.class).greaterThan("id", 0);
        List<NoteEntity> noteEntities = ormContext.query(ormPredicates);

        return noteEntities;
    }

    @Override
```

```
    public NoteEntity query(int id) {
        OrmPredicates ormPredicates = ormContext.where(
            NoteEntity.class).equalTo("id", id);
        List<NoteEntity> noteEntities = ormContext.query(ormPredicates);

        NoteEntity queriedNoteEntity = noteEntities.get(0);

        return queriedNoteEntity;
    }
}
```

4 添加界面

4.1 增加数据

添加编辑页面：

com.hostest.orm
    New
        Ability
            Empty Page Ability
                NoteInfoAbility

在entry\src\main\resources\base\graphic目录下创建background_save.xml文件，为“保存”按钮定义背景样式：

```
<shape
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:shape="rectangle">
    <corners ohos:radius="100"/>
    <solid ohos:color="#00a2e8"/>
</shape>
```

在entry\src\main\resources\base\layout\ability_note_info.xml中设计详情页面布局：

```
<DependentLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:height="match_parent"
    ohos:width="match_parent">

    <TextField
        ohos:id="$+id:tfTitle"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:margin="20vp"
        ohos:multiple_lines="true"
        ohos:basement="#808080"
        ohos:hint="填写标题"
        ohos:text_size="24fp"
        />

    <TextField
        ohos:id="$+id:tfContent"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:below="$id:tfTitle"
        ohos:left_margin="20vp"
        ohos:right_margin="20vp"
        ohos:multiple_lines="true"
        ohos:hint="编辑内容"
        ohos:text_size="22fp"
        />

    <Button
        ohos:id="$+id:btnSave"
        ohos:height="50vp"
        ohos:width="match_parent"
        ohos:padding="10vp"
        ohos:align_parent_bottom="true"
        ohos:margin="20vp"
        ohos:background_element="$graphic:background_save"
        ohos:text="保存"
        ohos:text_size="24fp"
        ohos:text_color="#ffffff"
        />

</DependentLayout>
```

在entry\src\main\java\com\hostest\orm\slice\NoteInfoAbilitySlice.java中为NoteInfoAbilitySlice类增加属性:

```
public class NoteInfoAbilitySlice extends AbilitySlice {
    private NoteDao noteDao;

    private TextField tfTitle;
    private TextField tfContent;
    private Button btnSave;
    ...
}
```

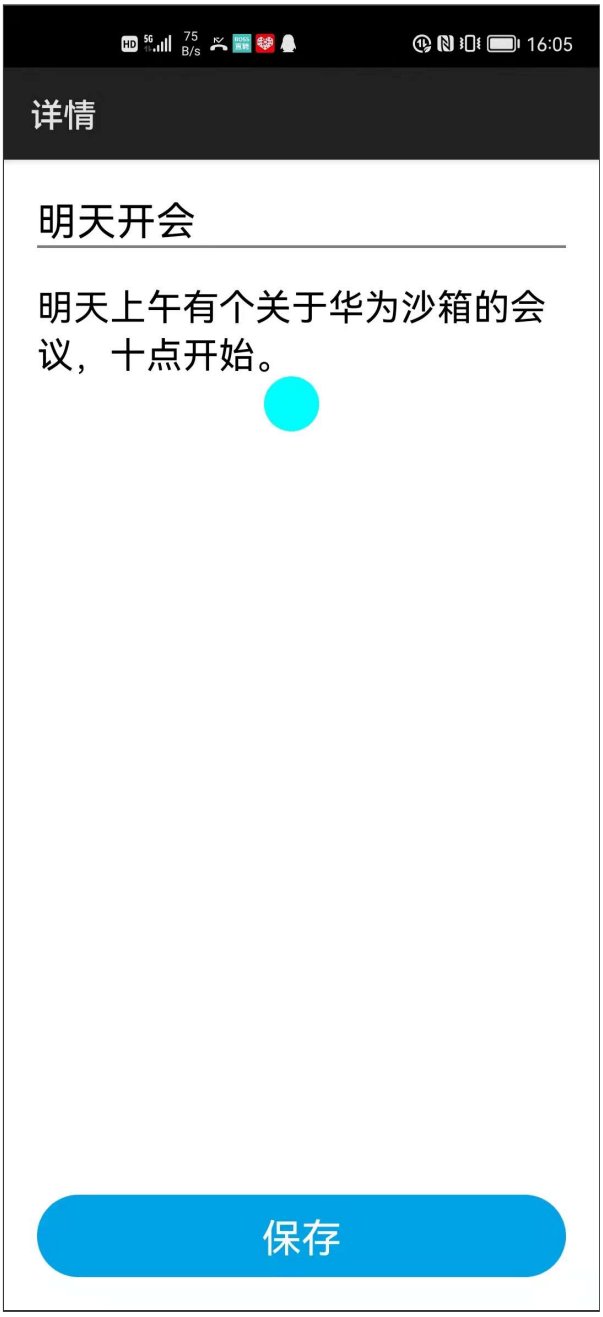
在NoteInfoAbilitySlice类中为“保存”按钮添加点击事件处理：

```
public class NoteInfoAbilitySlice extends AbilitySlice {
    ...
    @Override
    public void onStart(Intent intent) {
        super.onStart(intent);
        super.setUIContent(ResourceTable.Layout_ability_note_info);

        noteDao = new NoteDaoImpl(this);

        tfTitle = (TextField)findComponentById(ResourceTable.Id_tfTitle);
        tfContent = (TextField)findComponentById(ResourceTable.Id_tfContent);
        btnSave = (Button)findComponentById(ResourceTable.Id_btnSave);

        btnSave.setClickedListener(component -> {
            NoteEntity noteEntity = new NoteEntity();

            noteEntity.setTitle(tfTitle.getText());
            noteEntity.setContent(tfContent.getText());
            noteEntity.setDate(System.currentTimeMillis());

            if (noteDao.insert(noteEntity))
                new ToastDialog(this)
                    .setText("保存成功")
                    .setAlignment(LayoutAlignment.CENTER)
                    .show();
            else
                new ToastDialog(this)
                    .setText("保存失败")
                    .setAlignment(LayoutAlignment.CENTER)
                    .show();
        });
    }
    ...
}
```

在MainAbilitySlice类中为主页面文本添加点击事件处理，点击该文本跳转到详情页面：

```
public class MainAbilitySlice extends AbilitySlice {
    @Override
    public void onStart(Intent intent) {
        super.onStart(intent);
        super.setUIContent(ResourceTable.Layout_ability_main);

        findComponentById(ResourceTable.Id_text_helloworld)
            .setClickedListener(component -> {
                Intent intentNoteInfo = new Intent();
                intentNoteInfo.setOperation(new Intent.OperationBuilder()
                    .withDeviceId("")
                    .withBundleName("com.hostest.orm")
                    .withAbilityName("com.hostest.orm.NoteInfoAbility")
                    .build());
                startAbility(intentNoteInfo);
            });
    }
    ...
}
```

运行效果如下图所示：



4.2 查询数据

在entry\src\main\resources\base\graphic目录下创建background_button.xml文件，为“+”按钮定义背景样式：

```
<shape
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:shape="oval">
    <solid ohos:color="#00a2e8"/>
</shape>
```

在entry\src\main\resources\base\layout\ability_main.xml中设计列表页面布局：

```
<DependentLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:height="match_parent"
    ohos:width="match_parent">

    <ListContainer
        ohos:id="$+id:lc"
        ohos:height="match_parent"
        ohos:width="match_parent"
        ohos:margin="10vp"
        />

    <Button
        ohos:id="$+id:btn"
        ohos:height="50vp"
        ohos:width="50vp"
        ohos:align_parent_right="true"
        ohos:align_parent_bottom="true"
        ohos:margin="20vp"
        ohos:background_element="$graphic:background_button"
        ohos:text="+"
        ohos:text_size="30fp"
        ohos:text_color="#ffffff"
        />

</DependentLayout>
```

将删除图片delete.png放到entry\src\main\resources\base\media目录下：

在entry\src\main\resources\base\layout目录下创建note_item.xml文件，定义列表项布局：

```
<DependentLayout
    xmlns:ohos="http://schemas.huawei.com/res/ohos"
    ohos:height="match_content"
    ohos:width="match_parent"
    ohos:margin="10vp">

    <Text
        ohos:id="$+id:txtTitle"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:multiple_lines="true"
        ohos:text_size="26fp"
        />

    <Text
        ohos:id="$+id:txtContent"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:below="$id:txtTitle"
        ohos:top_margin="10vp"
        ohos:multiple_lines="true"
        ohos:text_size="20fp"
        ohos:text_color="#808080"
        />

    <Text
        ohos:id="$+id:txtDate"
        ohos:height="match_content"
        ohos:width="match_parent"
        ohos:below="$id:txtContent"
        ohos:top_margin="10vp"
        ohos:text_size="18fp"
        ohos:text_color="#c3c3c3"
        />

    <Text
        ohos:id="$+id:txtId"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:align_parent_right="true"
        ohos:align_bottom="$id:txtDate"
        ohos:text_size="18fp"
        ohos:text_color="#c3c3c3"
        />

    <Text
        ohos:height="2vp"
        ohos:width="match_parent"
        ohos:below="$id:txtDate"
        ohos:top_margin="5vp"
        ohos:background_element="#808080"
```

```
        />

    <Image
        ohos:id="$+id:img"
        ohos:height="match_content"
        ohos:width="match_content"
        ohos:align_parent_right="true"
        ohos:align_parent_top="true"
        ohos:image_src="$media:delete"
        />

</DependentLayout>
```

添加provider包和NoteItemProvider类，为列表容器提供表项组件：

```
public class NoteItemProvider extends BaseItemProvider {
    private List<NoteEntity> noteEntities;

    public NoteItemProvider(List<NoteEntity> noteEntities) {
        this.noteEntities = noteEntities;
    }

    @Override
    public int getCount() {
        return noteEntities.size();
    }

    @Override
    public Object getItem(int i) {
        return noteEntities.get(i);
    }

    @Override
    public long getItemId(int i) {
        return i;
    }

    @Override
    public Component getComponent(int i, Component component,
        ComponentContainer componentContainer) {
        if (component == null)
            component = LayoutScatter.getInstance(
                componentContainer.getContext()).parse(
                ResourceTable.Layout_note_item, null, false);

        ((Text)component.findComponentById(ResourceTable.Id_txtId))
            .setText(noteEntities.get(i).getId().toString());
        ((Text)component.findComponentById(ResourceTable.Id_txtTitle))
            .setText(noteEntities.get(i).getTitle());
        ((Text)component.findComponentById(ResourceTable.Id_txtContent))
            .setText(noteEntities.get(i).getContent());

        Calendar calendar = Calendar.getInstance();
        calendar.setTimeInMillis(noteEntities.get(i).getDate());
        ((Text)component.findComponentById(ResourceTable.Id_txtDate))
            .setText(new SimpleDateFormat("yyyy-MM-dd HH:mm:ss")
            .format(calendar.getTime()));

        return component;
    }
}
```

为MainAbilitySlice类增加显示查询列表的功能，并在点击“+”按钮后跳转到详情页面：

```
public class MainAbilitySlice extends AbilitySlice {
    private NoteDao noteDao;

    private ListContainer listContainer;
    private Button button;

    @Override
    public void onStart(Intent intent) {
        super.onStart(intent);
        super.setUIContent(ResourceTable.Layout_ability_main);

        noteDao = new NoteDaoImpl(this);

        listContainer = (ListContainer)findComponentById(
            ResourceTable.Id_lc);
        button = (Button)findComponentById(ResourceTable.Id_btn);

        button.setClickedListener(component -> {
            Intent intentNoteInfo = new Intent();
            intentNoteInfo.setOperation(new Intent.OperationBuilder()
                .withDeviceId("")
                .withBundleName("com.hostest.orm")
                .withAbilityName("com.hostest.orm.NoteInfoAbility")
                .build());
            startAbility(intentNoteInfo);
        });
    }

    @Override
    public void onActive() {
        super.onActive();

        listContainer.setItemProvider(
            new NoteItemProvider(noteDao.query()));
    }
    ...
}
```

运行效果如下图所示：

4.3 更新数据

点击查询列表中表项，跳转到详情页面，编辑被点击条目的内容：

```
public class MainAbilitySlice extends AbilitySlice {
    ...
    @Override
    public void onStart(Intent intent) {
        ...
        listContainer.setItemClickedListener(new ItemClickedListener() {
            @Override
            public void onItemClicked(ListContainer listContainer,
                Component component, int i, long l) {
                Intent intentNoteInfo = new Intent();
                intentNoteInfo.setOperation(new Intent.OperationBuilder()
                    .withDeviceId("")
                    .withBundleName("com.hostest.orm")
                    .withAbilityName("com.hostest.orm.NoteInfoAbility")
                    .build());
                intentNoteInfo.setParam("id", Integer.valueOf(
                    ((Text)component.findComponentById(
                    ResourceTable.Id_txtId)).getText()).intValue());
                startAbility(intentNoteInfo);
            }
        });
        ...
    }
    ...
}
```

在详情页面中添加更新数据库的分支：

```
public class NoteInfoAbilitySlice extends AbilitySlice {
    ...
    @Override
    public void onStart(Intent intent) {
        ...
        int id = intent.getIntParam("id", -1);
        if (id != -1) {
            NoteEntity noteEntity = noteDao.query(id);
            tfTitle.setText(noteEntity.getTitle());
            tfContent.setText(noteEntity.getContent());
        }

        btnSave.setClickedListener(component -> {
            if (id != -1) {
                NoteEntity noteEntity = noteDao.query(id);

                noteEntity.setTitle(tfTitle.getText());
                noteEntity.setContent(tfContent.getText());
                noteEntity.setDate(System.currentTimeMillis());

                if (noteDao.update(noteEntity))
                    new ToastDialog(this)
                        .setText("保存成功")
                        .setAlignment(LayoutAlignment.CENTER)
                        .show();
                else
                    new ToastDialog(this)
                        .setText("保存失败")
                        .setAlignment(LayoutAlignment.CENTER)
                        .show();
            }
            else {
                NoteEntity noteEntity = new NoteEntity();

                noteEntity.setTitle(tfTitle.getText());
                noteEntity.setContent(tfContent.getText());
                noteEntity.setDate(System.currentTimeMillis());

                if (noteDao.insert(noteEntity))
                    new ToastDialog(this)
                        .setText("保存成功")
                        .setAlignment(LayoutAlignment.CENTER)
                        .show();
                else
                    new ToastDialog(this)
                        .setText("保存失败")
                        .setAlignment(LayoutAlignment.CENTER)
                        .show();
            }
        });
    }
```

```
        ...
    }
```

运行效果如下图所示：

4.4 删除数据

为MainAbilitySlice类添加删除方法：

```
public class MainAbilitySlice extends AbilitySlice {
    ...
    @Override
    public void onActive() {
        super.onActive();

        listContainer.setItemProvider(
            new NoteItemProvider(noteDao.query(), this));
    }
    ...
    public void delete(int id) {
        if (noteDao.delete(id)) {
            new ToastDialog(this)
                .setText("删除成功")
                .setAlignment(LayoutAlignment.CENTER)
                .show();

            listContainer.setItemProvider(
                new NoteItemProvider(noteDao.query(), this));
        }
        else
            new ToastDialog(this)
                .setText("删除失败")
                .setAlignment(LayoutAlignment.CENTER)
                .show();
    }
}
```

在NoteItemProvider中添加对删除图像的点击处理：

```
public class NoteItemProvider extends BaseItemProvider {
    private List<NoteEntity> noteEntities;
    private MainAbilitySlice mainAbilitySlice;

    public NoteItemProvider(List<NoteEntity> noteEntities,
        MainAbilitySlice mainAbilitySlice) {
        this.noteEntities = noteEntities;
        this.mainAbilitySlice = mainAbilitySlice;
    }
    ...
    @Override
    public Component getComponent(int i, Component component,
        ComponentContainer componentContainer) {
        ...
        component.findComponentById(ResourceTable.Id_img)
            .setClickedListener(new ClickedListener() {
                @Override
                public void onClick(Component component) {
                    new CommonDialog(componentContainer.getContext())
                        .setTitleText("提示")
                        .setContentText("您真的要删除【" +
                            noteEntities.get(i).getTitle() + "】吗？")
                        .setButton(0, "确定", new IDialog.ClickedListener() {
                            @Override
                            public void onClick(IDialog iDialog, int j) {
                                iDialog.destroy();
                                mainAbilitySlice.delete(
                                    noteEntities.get(i).getId());
                            }
                        })
                        .setButton(1, "取消", new IDialog.ClickedListener() {
                            @Override
                            public void onClick(IDialog iDialog, int j) {
                                iDialog.destroy();
                            }
                        })
                        .setSize(LayoutConfig.MATCH_CONTENT,
                            LayoutConfig.MATCH_CONTENT)
                        .show();
                }
            });

        return component;
    }
}
```

运行效果如下图所示：

5 真机部署

登录
https://developer.huawei.com/consumer/cn/service/josp/agc/index.html#/
添加项目和应用，通过自动签名部署到真机，运行调试。

更多精彩，敬请期待.....

达内集团C++教学部

2021年8月10日