# 软件测试报告

162130117-傅锦龙

## 黑盒测试

对新建功能进行等价类方法的黑盒测试，新建功能包括新建书签，分类或卡片。

|  |  |  |  |
| --- | --- | --- | --- |
| 划分依据 | | 有效等价类 | 无效等价类 |
| 按输入条件进行划分 | 卡片必要信息输入个数 | （1）3个 | （6）小于3个 |
| 按输入书签是否与已有重复 | （2）不重复 | （7）重复 |
| 按输入分类是否与已有重复 | （3）不重复 | （8）重复 |
| 按输入是否为空 | （4）不为空 | （9）为空 |
| 按输出条件划分 | 按出错提示 | （5）生成成功 | （10）出错提示 |

对应的测试用例如下

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 目标等价类 | 测试输入 | 预期结果 | 实际覆盖的等价类 |
| 1 | （1） | 创建卡片 tit,book,cate | 创建成功 | （1）（5） |
| 2 | （2） | 创建书签 book（未有book) | 创建成功 | （2）（4）（5） |
| 3 | （3） | 创建分类 cate（未有cate) | 创建成功 | （3）（4）（5） |
| 4 | （6） | 创建卡片 tit,book, | 出错提示 | （6）（10） |
| 5 | （7） | 创建书签 book（已有book) | 出错提示 | （7）（10） |
| 6 | （8） | 创建分类 cate（已有cate) | 出错提示 | （8）（10） |
| 7 | （9） | 创建分类 | 出错提示 | （9）（10） |

## 白盒测试

对创建或编辑卡片功能进行白盒测试

  static editContent(Content *content*, BuildContext *context*,

      {bool *fixedBook* = false,

      bool *fixedCate* = false,

      bool *onlyEdit* = false}) async {

    Content? old;

    if (*onlyEdit*) {

      old = *content*.clone();

    }

    final ret = await \_newContent(

*content*: *content*,

*context*: *context*,

*fixedBook*: *fixedBook*,

*fixedCate*: *fixedCate*);

    if (ret == null) return;

*// ignore: use\_build\_context\_synchronously*

    final appData = Provider.of<AppData>(*context*, *listen*: false);

    int id;

    bool needPlus = true;

    if (*onlyEdit*) {

      needPlus = false;

      if (old!.book.name != ret.book.name) {

        final n = await Book.getByName(ret.book.name);

        ret.book.id = n == null ? Isar.autoIncrement : n.id;

      }

*// 检查旧的分类名称是否与新的分类名称不同*

      if (old.category.name != ret.category.name) {

*// 如果不同，那么获取新的分类*

        final n = await Category.getByName(ret.category.name);

*// 将旧的分类的数量减1*

        old.category.count -= 1;

*// 更新旧的分类的id为新的分类的id*

        old.category.id = ret.category.id;

*// 将旧的分类添加到数据库中，并获取新的id*

        id = await appData.addCategory(old.category);

*// 标记需要将新的分类的数量加1*

        needPlus = true;

*// 检查新的分类是否已经存在*

        if (n == null) {

*// 如果不存在，那么设置新的分类的id为自动增长的id，并将数量设置为0*

          ret.category.id = Isar.autoIncrement;

          ret.category.count = 0;

        } else {

*// 如果存在，那么设置新的分类的id和数量为已存在的分类的id和数量*

          ret.category.id = n.id;

          ret.category.count = n.count;

        }

      }

    }

    id = await appData.addContent(ret, *needPlus*: needPlus);

*// ignore: use\_build\_context\_synchronously*

    await displayInfoBar(*context*, *builder*: (*context*, *close*) {

      return InfoBar(

*title*: const Text('结果:'),

*content*: Text(id == *content*.id ? '成功编辑内容' : '成功创建内容'),

*action*: IconButton(

*icon*: const Icon(FluentIcons.clear),

*onPressed*: *close*,

        ),

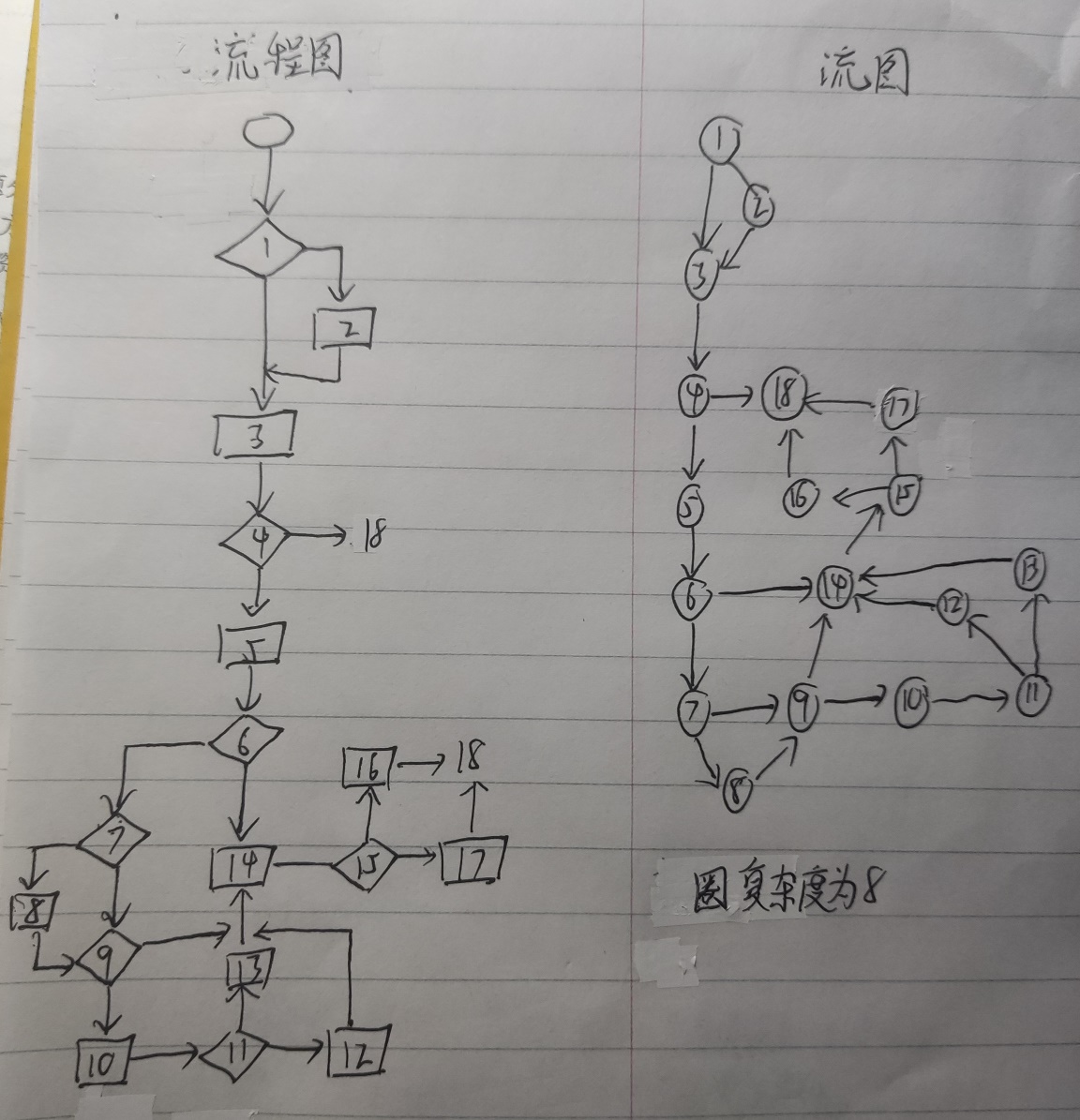
*severity*: InfoBarSeverity.info,

      );

    });

  }

对应的程序流图如下



基本路径如下

1. 1-3-4-18
2. 1-2-3-4-18
3. 1-3-4-5-6-14-15-16-18
4. 1-3-4-5-6-14-15-17-18
5. 1-2-3-4-5-6-7-9-14-15-16-18
6. 1-2-3-4-5-6-7-8-9-14-15-16-18
7. 1-2-3-4-5-6-7-9-10-11-12-14-15-16-18
8. 1-2-3-4-5-6-7-9-10-11-13-14-15-16-18

设计测试用例

1. onlyEdit=false,ret=null，结果：创建失败
2. onlyEdit=true,ret=null，结果：编辑失败
3. onlyEdit=false,ret!=null,创建卡片，结果：创建成功
4. onlyEdit=true,ret!=null,编辑卡片，结果：编辑成功
5. onlyEdit=true,ret!=null,不修改书签，编辑卡片，结果：编辑成功
6. onlyEdit=true,ret!=null,修改书签，编辑卡片，结果：编辑成功
7. onlyEdit=true,ret!=null,不修改书签，修改为新分类，编辑卡片，结果：编辑成功
8. onlyEdit=true,ret!=null,不修改书签，修改为已有分类，编辑卡片，结果：编辑成功