[质量保证文档]

[用于解释和说明关于寸草心项目

测试的相关注意事项]

[Group 4]

[2017/6/10]

目录

一、测试计划

1. 引言

1.1 编写目的

测试计划主要面向系统负责人、项目开发人员和项目测试人员，详细完备的测试计划有助于所有相关人员对项目测试内容、测试流程的了解，进而使工作更有针对性，提高团队效率。

1.2 项目背景

随着中国国人口老龄化的趋势以及目前智能手机普遍的现状，老年人们急需一款操作简单但功能多样的、可用于与子女沟通等方面的 APP；针对老年人视力相对较差、长时间使用手机易产生不适、倾向喜欢相对简单的操作流程等特征，我们设计了这款 APP。

2. 计划

2.1 测试方案 【说明确定测试方法和选取测试用例的原则】

2.1.1.功能测试

测试范围	验证数据的精确度、数据类型、业务功能等相关方面的正确性。

测试目标	核实所有功能均已正常实现，即是否与需求一致。
采用技术	主要采用黑盒测试、边界测试等测试方法。
工具与方法	手动测试、编码测试
开始标准	开发阶段对应的功能完成并且测试用例设计完成
完成标准	测试用例通过并且最高级缺陷全部解决

2.1.2.用户界面(UI)测试

测试范围	1.界面菜单、背景、颜色、字体、按钮名称、TITLE、提示信息的一致性； 2.友好性、可操作性（易用性）
测试目标	核实各个界面风格（包括颜色、字体、提示信息、图标、title 等）都与需求保持一致，或符合可接受标准，能够保证用户界面的友好性、易操作性，而且符合用户操作习惯。
采用技术	CI 自动集成测试，Robotium 框架
工具与方法	手动测试、自动测试、编码测试
开始标准	界面开发完成
完成标准	UI 符合可接受标准，能够保证用户界面的友好性、易操作性，而且符合用户操作习惯
测试重点与优先级	界面跳转，颜色搭配，信息一致性

2.1.3.安全性测试

测试范围	1.用户、管理员的密码安全 2.权限 3.非法攻击
测试目标	1. 用户、管理员的密码管理 2．应用程序级别的安全性：核实用户只能操作其所拥有权限能操作的功能。 3．系统级别的安全性：核实只有具备系统访问权限的用户才能访问系统。
采用技术	代码包或者非法攻击工具
工具与方法	百度云与 testin
开始标准	功能测试完成
完成标准	执行各种非法操作无安全漏洞且系统使用正常
测试重点与优先级	组件安全

2.1.4.兼容性测试

测试范围	1．使用不同厂家的不同机型进行测试。 2．不同机型和各种运行软件等各种条件的组合测试。
测试目标	核实系统在不同的软件和硬件配置中运行稳定
采用技术	黑盒测试
工具与方法	百度云与 testin
开始标准	项目组移交系统测试

完成标准	在不同机型下均能正常实现其功能(此测试根据开发提供依据决定测试范围)
测试重点与优先级	根据实际需求而定
需考虑的特殊事项	根据实际需求而定

2.1.5.回归测试

测试范围	所有功能、用户界面、兼容性、安全性等测试类型
测试目标	核实执行所有测试类型后功能、性能等均达到用户需求所要求的标准
采用技术	黑盒测试
工具与方法	手工测试和自动化测试
开始标准	每当被测试的软件或其环境改变时在每个合适的测试阶段上进行回归测试
完成标准	95%的测试用例执行通过并通过系统测试
测试重点与优先级	测试优先级以测试需求的优先级为参照
需考虑的特殊事项	软硬件设备问题

2.2．测试实施阶段

测试类型	测试阶段			
	单元测试	集成测试	系统测试	验收测试
功能测试	×	✓	✓	×
性能测试	×	✓	✓	×
安全性测试	×	✓	✓	×
兼容性测试	×	✓	✓	×
用户界面（UI）测试		×	✓	×
回归测试	每当被测试的软件或其环境改变时在每个合适的测试阶段上进行回归测试。			
备注：“✓”表示由测试组执行，“×”表示由项目组执行；				

2.3 测试人员

史彤： 测试负责人

李心睿： 安全性测试、兼容性测试、性能测试、手动测试负责人

董鹏程： 编写测试用例，实施自动化测试、手动测试

3、测试进度控制

测试项目	**时间安排**	**负责人员**
制定测试计划	2017.03.14-2017.03.20	史彤
熟悉被测试系统 搭建测	2017.03.21-2017.03.25	董鹏程

试环境		
进行测试前被测系统培训	2017.03.25-2017.03.30	史彤
设计测试用例	2017.04.05-2017.05.30	史彤 董鹏程 李心睿
测试用例评审	2017.04.05-2017.05.30	史彤 董鹏程 李心睿
执行测试用例	2017.04.05-2017.05.30	史彤 董鹏程 李心睿
整理测试结果，出软件问题清单和测试分析报告	2017.06.06-2017.06.20	史彤 李心睿

4、测试标准

4.1 测试接收标准

1）到测试合同（或项目计划）约定的时间；

2）软件测试所需的各种文档已经准备完毕；

3）所提交的被测软件受控；

4）软件源代码正确通过编译或汇编；

5）最好从一开始就介入到被测软件的开发周期。

4.2 测试停止标准

1）按要求完成了合同（或项目计划）所规定的软件测试任务；

2）实际测试过程遵循了原定的软件测试计划和软件测试说明；

3）客观、详细地记录了软件测试过程和软件测试中发现的所有问题；

4）软件测试的全过程自始至终在控制下进行；

5）软件测试中的问题或异常有合理解释或正确有效的处理；

6）软件测试工作通过了测试评审；

7）全部测试软件、被测软件、测试支持软件和评审结果已纳入配置管理。

4.3 非正常停止标准

1）项目需要暂停进行调整，测试应暂停并备份暂停点的数据；

2）软件在开发过程中出现重大偏差；

3）本轮提交的缺陷未得到开发反馈；

4）项目和需求中有 2 处不一致的情况出现；

5）项目经理有特殊情况，需发文档说明并停止测试。

4.4 通过标准

4.4.1 测试项的通过标准

测试项的通过标准：当此项的功能能够正确地完成，并且它的操作没有引起其他功能项或整个系统的错误，则认为此项测试通过。

4.4.2 系统测试通过标准

系统测试的通过标准：对于每一类测试，当没有发现致命性错误和严重性错误、一般性错误数量小于测试用例总数的 2%，告警性错误数量小于测试用例总数的 5%，则认为系统通过本次测试，但要以测试结果评审会的评审结果为最后标准。

5、其它

5.1 影响计划的潜在因素

在测试计划执行过程中，可能存在以下因素影响计划的按时完成：

1）测试人员对被测试产品的熟悉进度慢；

2）测试人员对测试工具的使用熟悉程序不够；

3）被测试产品存在重大错误，以致于测试无法继续，需要开发组进行额外的调试和修改才能继续；

4）硬件、软件或网络环境出现故障等。

5）第一点是影响测试进度的最大的因素。

5.2 应急措施

如果上述潜在的可能事件发生，则通过适当加班来保证计划的按时完成。如果是由于被测试产品存在重大错误而严重影响测试进度，则考虑按照测试暂停标准来暂停该测试。

5.3 测试的局限性

1）系统硬件配置存在不可预测的问题；

2）测试范围不能覆盖所有的可能情况；

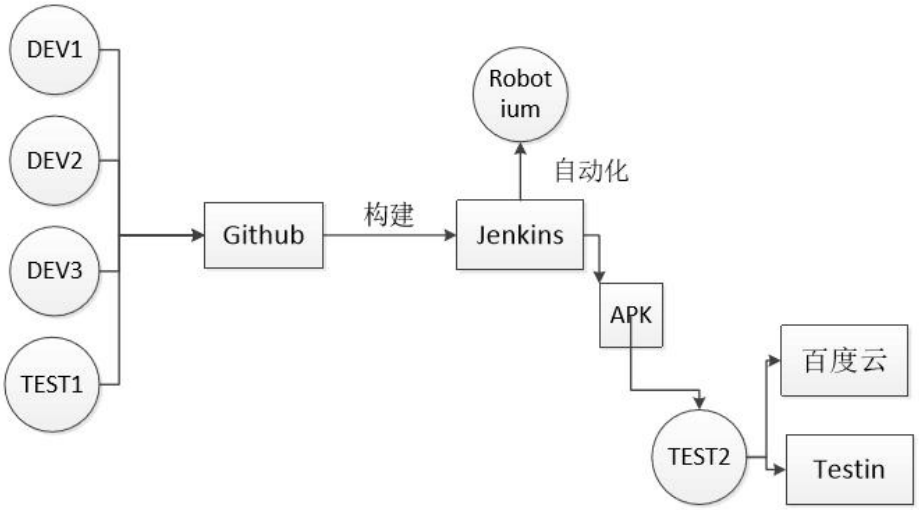
3）测试时间的限制；

4）测试数据可能不全面；

5）测试工具自身的缺陷；

6）测试人员的失误。

二、测试设计



三． 测试用例设计

1.UI 测试

1.1 注册

编制人	DongPC	审定人		时间	
软件名称	寸草心		编号/版本	V 1.2	
测试用例	注册的输入				
用例编号	2001				
参考信息（参考的文档及章节号或功能项）： 注册					

输入说明（列出选用的输入项，覆盖正常、异常情况）： 1、点击注册按钮 2、输入 11 位手机号，输入密码，二次输入密码，注册。 3、输入 10 位手机号，输入密码，二次输入密码，注册。 4、输入 9 位数手机号，输入密码，二次输入密码，注册。 5、输入含特殊符号（, 。空格汉字）的手机号，输入密码，二次输入密码，注册。 6、输入低于六位数的密码。 7、输入超过 25 位数的密码 8、密码第一个数包含空格。 9、密码中间包含空格 10、密码包含汉字 11、确认密码是 与第一次输入不一致 12、第二次输入密码时，复制第一次输入的，然后粘贴到第二次输入的。
输出说明（逐条与输入项对应，列出预期输出）： 1、跳转到注册页面 2、成功注册 3、提示手机号输入错误 4、提示手机号输入错误 5、提示手机号输入错误 6、提示密码不能低于 6 位 7、提示密码不能超过 20 位

8、密码中不应含有空格

9、密码中不应含有空格

10、提示密码格式错误

11、提示两次密码输入不一致

12、提示两次密码输入不一致

环境要求（测试要求的软、硬件、网络要求）：

Android 4.4 联网

特殊规程要求：

无

用例间的依赖关系

无：

编制人	DongPC	审定人		时间	
软件名称	记事本程序		编号/版本	V 1.2	
测试用例	注册时点击注册按钮				
用例编号	2002				

参考信息（参考的文档及章节号或功能项）：

注册

输入说明（列出选用的输入项，覆盖正常、异常情况）： 1、输入正确的注册信息后，点击注册按钮。 2、多次点击注册按钮。
输出说明（逐条与输入项对应，列出预期输出）： 1、提时注册成功。 2、提示已注册
环境要求（测试要求的软、硬件、网络要求）： Windows 2000 professional 中文版
特殊规程要求： 无
用例间的依赖关系： 2001

1.2 登录

编制人	DongPC	审定人		时间	
软件名称	寸草心		编号/版本	V1.2	
测试用例	登录				
用例编号	2003				

参考信息（参考的文档及章节号或功能项）： 登录
输入说明（列出选用的输入项，覆盖正常、异常情况）： 1、打开“寸草心”，输入正确用户名，密码，登录 2、输入不存在的用户名，登录 3、输入数据库中含有的用户名，错误的密码 （找回密码功能）
输出说明（逐条与输入项对应，列出预期输出）： 1、成功登录 2、提示用户名不存在，并可以重新输入 3、提示密码错误，并可以重新输入
环境要求（测试要求的软、硬件、网络要求）： Android 4.4
特殊规程要求： 无
用例间的依赖关系： 无

1.3 群聊相关

编制人	DongPC	审定人		时间	2017 5/20

软件名称	寸草心	编号/版本	V 1.1
测试用例	切换群聊，浏览信息		
用例编号	2007		
参考信息（参考的文档及章节号或功能项）： 群聊			
输入说明（列出选用的输入项，覆盖正常、异常情况）： 1、点击群聊按钮。 2、点击某一群聊。 3、如果已有过群聊信息，进入群聊后。 4、如果没有群聊信息，进去群聊后。 5、点击主界面“+”。 6、不进行任何操作，点击发表 7、仅添加图片，点击发表 8、不输入任何信息，选择群聊，点击发表。 9、填写完整的信息，并选择群聊，点击发表 10、填写群聊信息时，退出编辑。 11、基于 10，再次进入编辑界面 12、编辑过程中，按 home 键。 13、基于 12，再次打开 App 14、基于 12，一段时间（5 分钟+），再次打开 App 15、基于 14，打开发送群聊			

输出说明（逐条与输入项对应，列出预期输出）： 预期输出： 1、跳转至显示所有群聊界面 2、跳转至群聊的信息界面 3、可查看群聊信息 4、不含群聊信息 5、跳转至发送群聊界面 6、提示请选择群聊 7、提示请选择群聊 8、提示请输入信息 9、成功发送群聊信息 10、直接回到主界面（此处应该提示是否保存信息） 11、没有保存信息（此处应该回到上一次的编辑状态） 12、返回手机的主界面 13、返回到原来界面 14、重新的登陆界面。 15、回到退出前的编辑状态
环境要求（测试要求的软、硬件、网络要求）： Android 4.4 联网 群聊功能完成
特殊规程要求： 无
用例间的依赖关系： 2003（必须先完成注册登陆）

1.4 老人端登录

编制人	DongPC	审定人		时间	2017 5/20
软件名称	寸草心		编号/版本	V1.1	
测试用例	老人端的登陆				
用例编号	2008				

参考信息（参考的文档及章节号或功能项）： 登陆
输入说明（列出选用的输入项，覆盖正常、异常情况）： 1、打开 App。 2、输入正确的登陆账号密码，点击登陆。 3、输入账号（位数错误）和密码，点击登陆 4、输入正确的账号和错误的密码，点击登陆 5、点击退出登陆 6、基于 5 再次打开 App 7、强制关闭 App，再次打开
输出说明（逐条与输入项对应，列出预期输出）： 预期输出： 1、跳转至登陆的主界面 2、正常登陆，跳转至主界面。（是否需要进入填写具体信息界面待讨论） 3、提示手机号位数错误。（这里应该提示老人具体的错误信息） 4、提示密码错误。 5、退出 App 6、进入主界面 7、进入主界面
环境要求（测试要求的软、硬件、网络要求）： Android 4.4 联网

特殊规程要求： 无
用例间的依赖关系： 2002

1.5 发朋友圈

编制人	DongPC	审定人		时间	2017/6/2
软件名称	寸草心		编号/版本	V1.2	
测试用例	发朋友圈（老人）				
用例编号	2011				
参考信息（参考的文档及章节号或功能项）： 参考子女端朋友圈发送					
输入说明（列出选用的输入项，覆盖正常、异常情况）： 1、点击右上角“+” 2、填写文字信息，点击发送 3、填写文字信息，并选择群聊，点击发送 4、添加图片，点击发送 5、添加图片，选择群聊，点击发送					

输出说明（逐条与输入项对应，列出预期输出）： 预期输出： 1、跳转至编辑界面 2、本人的主界面无显示 3、本人及群聊好友的界面有显示 4、本人界面无显示 5、本人及群聊好友界面有显示
环境要求（测试要求的软、硬件、网络要求）： 安卓 4.4 版本以上
特殊规程要求：
用例间的依赖关系：

1.6 群组

编制人	DongPC	审定人		时间	2017/6/2
软件名称	寸草心		编号/版本	V1.2	
测试用例	群组				
用例编号	2012				
参考信息（参考的文档及章节号或功能项）： 2007					

输入说明（列出选用的输入项，覆盖正常、异常情况）： 1.无群组时，点击我的群组。 2.在我的好友界面，点击右上角 3.点击新加群组 4.基于 3，不进行任何输入，点击确定按钮 5.基于 3，填写文字信息，不选择好友，点击确定按钮 6.基于 3，正常填写，选择好友，点击确定 7.点击我的群组 8.基于 7，（事先没有好友）点击添加新成员 9.基于 7，（事先有好友）点击添加新成员 10.基于 8，点击添加新成员 11.基于 9，点击添加新成员 12.作为非群主成员，查看群聊，看一下是否有删除好友权限
输出说明（逐条与输入项对应，列出预期输出）： 1、进入我的群组界面，无任何群组 2、打开添加好友和创建群聊界面 3、进入编辑界面 4、禁止添加群组 5、提示请选择好友（单人无法创建群组） 6、成功创建群组 7、进入查看我的所有群组

8、进入添加好友界面，但无好友 9、进入添加好友界面，可以选择好友 10、提示添加失败 11、成功添加 12、无权限
环境要求（测试要求的软、硬件、网络要求）： Windows 2000 professional 中文版
特殊规程要求：
用例间的依赖关系：

2.接口测试用例

2.1 正文及评论

编制人	DongPC	审定人		时间	
用例名称	正文或者评论			用例接口	ShowType
项目名称	寸草心			版本号	V1.0
测试目	测试能否成功获取信息				

的				
接口方法名	无			
用例编号	输入	预期输出	实际输出	是否通过
1	获取 SHOW 的值	show	show	是
2	获取 COMMENT 的值	comment	comment	是

编制人	DongPC	审定人		时间	
用例名称	用户验证信息			用例接口	userBlackListener
项目名称	寸草心			版本号	V1.0
测试目的	测试用户发送的验证信息能否通过第三方服务器				
接口方法名	showResult(**boolean** result, String message)				
用例编号	输入	预期输出	实际输出	是否通过	
3	Message=null;	Result=false	Result=true	是	

4	Message= (空白);	Result=true	Result=true	是
5	Message="Aa"	Result =true	Result=true	是

2.2 用户好友列表

编制人	DongPC	审定人		时间	
用例名称	用户好友列表			用例接口	UserBackListListener
项目名称	寸草心			版本号	V1.0
测试目的	测试能否获取好友列表				
接口方法名	showResult(boolean result, ArrayList<String> message, List<UserInfo> userInfos)				
用例编号	输入	预期输出		实际输出	是否通过
6	Message=null; userInfo=null;	Result = false		Result=false	是
7	Message=null; userInfo!=null;	Result=false		Result=false	是
8	Message!=null; userInfo=null;	Result=false		Result=false	是

9	Message!=null; userInfo!=null;	Result=true	Result=true	是

2.3 监听返回的用户信息

编制人	DongPC	审定人		时间	
用例名称	监听返回的用户信息			用例接口	UserBackUserListInfo
项目名称	寸草心			版本号	V1.0
测试目的	测试能否监听用户的返回信息				
接口方法名	showResult（boolean result, List<UserInfo> message）				
用例编号	输入	预期输出		实际输出	是否通过
10	Message = null	Result=false		Result=false	是
11	Message !=null	Result=true		Result=true	是

2.4 获取从服务器中获得的信息

编制人	DongPC	审定人		时间	
用例名称	获取从第三方服务器获得的信息			用例接口	UserBackUserInfo
项目名称	寸草心			版本号	V1.0

测试目的	测试能否从第三方服务器获得的信息			
接口方法名	showResult(boolean result, String s, UserInfo message)			
用例编号	输入	预期输出	实际输出	是否通过
12	S!=null,message!=null	true	true	是
13	S=null,message=null	false	false	是
14	S=null,message!=null	true	true	是

四．测试代码

1.entity

1.1 CommentItemModelTest

```
package com.example.jasper.ccxapp.entitiy;

import org.junit.Test;
import static org.junit.Assert.*;

/**
 * Created by DPC on 2017/5/10.
 */
```

```
public class CommentItemModelTest {

	private String msgKey="a";
	private String commKey="b";
	private String commentUsername="c";
	private String commentVoice="d";
	private int commentLength=1;

	@Test
	public void testgetCommKey() throws Exception {
		assertEquals(commKey,"b");
	}

	@Test
	public void testsetCommKey() throws Exception {
	    int    commentLength = 2;
	     this.commentLength = commentLength;
		assertEquals(2,this.commentLength);
	}

	@Test
	public void testgetCommentLength() throws Exception {
		assertEquals(commentLength,1);
	}

	@Test
	public void testsetCommentLength() throws Exception {
		String commentUsername ="dd";
		this.commentUsername=commentUsername;
		assertEquals("dd",this.commentUsername);
	}

	@Test
	public void testgetMsgKey() throws Exception {
		assertEquals(msgKey,"a");
	}

	@Test
	public void setMsgKey() throws Exception {
	       String msgKey = "e";
		this.msgKey=msgKey;
		assertEquals("e",this.msgKey);
	}
```

```
    @Test
    public void testgetCommentUsername() throws Exception {
        assertEquals("c",commentUsername);
    }

    @Test
    public void testsetCommentUsername() throws Exception {
        String    commentUsername = "ee";
        this.commentUsername=commentUsername;
        assertEquals("ee",this.commentUsername);
    }

    @Test
    public void testgetCommentVoice() throws Exception {
        assertEquals("d",commentVoice);
    }

    @Test
    public void testsetCommentVoice() throws Exception {
        String commentVoice="ff";
        this.commentVoice=commentVoice;
        assertEquals("ff",this.commentVoice);
    }
}
```

1.2 NewFriendTest

```
package com.example.jasper.ccxapp.entitiy;

import org.junit.Test;
import static org.junit.Assert.*;
/**
 * Created by DPC on 2017/5/10.
 */
public class NewFriendTest {
    private String requestFriend="yes";
    private String responseFriend="get";
    private String message="hello";

    @Test
    public void testgetRequestFriend() throws Exception {
        assertEquals(requestFriend,"yes");
    }
```

```
    @Test
    public void testsetRequestFriend() throws Exception {
        String requestFriend = "no";
        this.requestFriend =requestFriend;
        assertEquals("no",this.requestFriend);
    }

    @Test
    public void testgetResponseFriend() throws Exception {
        assertEquals(responseFriend,"get");
    }

    @Test
    public void testsetResponseFriend() throws Exception {
        String responseFriend ="Noget";
        this.responseFriend = responseFriend;
        assertEquals("Noget",this.responseFriend);
    }

    @Test
    public void testgetMessage() throws Exception {
        assertEquals("hello",message);
    }

    @Test
    public void testsetMessage() throws Exception {
        String message = "what";
        this.message = message;
        assertEquals("what",this.message);

    }

}
```

1.3 ShowItemModelTest

```
package com.example.jasper.ccxapp.entitiy;

import org.junit.Test;
import java.io.File;
import java.util.ArrayList;
import java.util.List;

import static org.junit.Assert.*;
```

```
/**
 * Created by DPC on 2017/5/10.
 */
public class ShowItemModelTest {
    private String msgKey="a";
    private String showUsername="b";
    private String showText="c";
    private String showVideo="d";
    private File showAvatar=new File("F:/abc.txt");
    @Test
    public void testgetShowAvatar() throws Exception {
        assertEquals(showAvatar.getName(),"abc.txt");
    }

    @Test
    public void testsetShowAvatar() throws Exception {
        File s = new File("F:/bcd.txt");
        showAvatar=s;
        assertEquals(showAvatar.getName(),"bcd.txt");

    }

    @Test
    public void testgetMsgKey() throws Exception {
        assertEquals("a",msgKey);
    }

    @Test
    public void testrsetMsgKey() throws Exception {
        String msgkey ="aaa";
        this.msgKey=msgkey;
        assertEquals("aaa",this.msgKey);
    }

    @Test
    public void testgetShowUsername() throws Exception {
        assertEquals("b",showUsername);
    }

    @Test
    public void testsetShowUsername() throws Exception {
        String showusername= "bbb";
        this.showUsername= showusername;
```

```
        assertEquals("bbb",this.showUsername);
    }

    @Test
    public void testgetShowText() throws Exception {
        assertEquals("c",showText);
    }

    @Test
    public void testsetShowText() throws Exception {
        String showtext = "fff";
        this.showText= showtext;
        assertEquals("fff",this.showText);
    }

    @Test
    public void testgetShowVideo() throws Exception {
        assertEquals("d",showVideo);
    }

    @Test
    public void testsetShowVideo() throws Exception {
        String showvideo = "ddd";
        this.showVideo=showvideo;
        assertEquals("ddd",this.showVideo);
    }

}
```

2.widget

```
package com.example.jasper.ccxapp.widget;

import org.junit.Test;

import static org.junit.Assert.*;

/**
 * Created by DPC on 2017/5/10.
 */
public class CustomVideoViewTest {
    private int videoWidth=10;
    private int videoHeight=20;
    @Test
    public void testgetVideoWidth() throws Exception {
    assertEquals(videoWidth,10);
```

```
    }

    @Test
    public void setVideoWidth() throws Exception {
    this.videoWidth=20;
        assertEquals(20,this.videoWidth);
    }

    @Test
    public void getVideoHeight() throws Exception {
    assertEquals(20,videoHeight);
    }

    @Test
    public void setVideoHeight() throws Exception {
        this.videoHeight=30;
        assertEquals(30,this.videoHeight);
    }
}
```