Test and Integration Plans and Reports

Group 3

June 8, 2022

Contents

Revision History

Revision	**Date**	**Author(s)**	**Description**
0.1.0	14.03.22	SSP526	Doc created in GDocs
0.1.1	14.03.22	DM1306	Fit to LaTeXtemplate
0.2.0	14.03.22	DM1306	Add further information to all sections. Refocus on product goals. Change overall test strategy to bottom-up approach.
0.3.0	30.05.22	SSP526	Amend certain Unit tests and fill out available test records
0.3.1	07.06.22	SSP526	Fill out all test records

1 Introduction

1.1 Overview

This document describes our Testing Methodology that will be used through the development cycle of our product. It will define Functional Tests for User Stories as found in the Functional Specification document, and a broad overview of our methodology for generating and applying automated Unit Tests.

2 Test Plan

2.1 Overview

At the highest level, our codebase can be split into two super-modules: The UI Controller and the Engine. This is a practical distinction, with the two running on seperate threads to prevent heavy processing blocking the UI; the two super-modules may be divided further into several modules each.
The UI Controller may be seen to minimally consist of:

- Message Display
- 2D Graphics Display (Including Text and Tables)
- Image Display
- Video Display
- Audio Player Display
- Tool List Display
- Element Properties Display

The Engine may be seen to minimally consist of:

- Document Parse
- Document Output
- Event Handling
- Script Engine
- Remote Interface
- Tool Parse and Handling

These modules may even be split several times further into their component parts; we shall start our testing with these, employing the common JUnit Test framework to run localised, automated Unit Tests enusring that we have confidence in these parts as they become available, prior to further high-level Functional testing. In a word, our strategy is “bottom-up”.

2.2 Testing Format

1. Refer to the appropriate Test if available.
2. Ensure that all testable dependancies have passed Unit Testing.
3. Build the software with the module to-be-tested included.
4. Enter the required input for the Test.
5. Compare the expected outcome with the actual outcome.
6. Record the result.

2.3 Unit Testing

Automated Unit Testing shall be applied to “lower-level” modules. Every public method on an Object should be tested for correctnes of operation through the Unit Test suite using a combination of random, invalid, and valid inputs and a combination standard Unit Testing and “fuzzing” techniques.

2.4 Considerations

“Lower-level” modules have impacts on those above, and so flawed Unit Testing has the potential to invalidate further Functional Testing. This means that our Unit Test suite must be near-complete with high measured code coverage, to provide confidence in our semi-automatic and manual Functional Tests.

3 Functional Tests

3.1 Message Display

Description	Post individual blocking and non-blocking messages to the UI for display.
Purpose	Users require notification about certain events within the program. Test this function.
Inputs	Blocking message (Action-required message). Non-blocking message (Information message).
Expected Outcome	Messages containing the input text of the correct type shall be displayed.

3.2 2D Graphics Display

Description	Post valid and invalid 2D graphics objects to the UI for display.
Purpose	The User may require that a certain 2D Graphical element be displayed. Test this function.
Inputs	Random 2D graphical object.
Expected Outcome	Valid objects should be displayed correctly. Invalid objects should not be displayed.

3.3 Image Display

Description	Post valid and invalid images to the UI for display.
Purpose	The User may require that a certain image be displayed. Test this function.
Inputs	Random images.
Expected Outcome	Valid images should be displayed correctly. Invalid images should not be displayed.

3.4 Video Display

Description	Post valid and invalid videos to the UI for display.
Purpose	The User may require that a certain video be displayed. Test this function.
Inputs	Random images.
Expected Outcome	Valid video should be displayed correctly. Invalid video should not be displayed.

3.5 Audio Player

Description	Post valid and invalid audio to the UI for output.
Purpose	The User may require that certain audio is played. Test this function.
Inputs	Random audio files.
Expected Outcome	Valid audio should be played correctly. Invalid video should not be played.

3.6 Tool List Display

Description	Post valid and invalid tools to the UI for display.
Purpose	The User requires access to tools from the Tools menu. Test this function.
Inputs	Selection of Tools.
Expected Outcome	Available tools are displayed in the Tools menu.

3.7 Element Properties Display

Description	Click on a visual element to show its' properties. Click off to hide them.
Purpose	The User shall select a visual element, which should reveal its' properties in the Properties menu. Test this function.
Inputs	Mouse clicks.
Expected Outcome	An object's properties are displayed in the Properties menu on click on the object.

3.8 Document Parse

Description	Post valid and invalid documents to the engine to parse.
Purpose	The User shall open a document, and the engine will attempt to parse it. Test this function.
Inputs	Valid and Invalid presentation XML documents.
Expected Outcome	The parsed result of a valid document is returned. Invalid documents return nothing.

3.9 Document Output

Description	Try to save a presentation Stack Document.
Purpose	The User shall edit a document and then save it. Test this function.
Inputs	Graphically edited document.
Expected Outcome	Valid XML document is written to the User's specified location.

3.10 Event Handling

Description	Post valid and invalid events to the engine.
Purpose	The User shall perform an action and the engine should respond. Test this function.
Inputs	User actions.
Expected Outcome	Correct response to a valid event. No response to an invalid event.

3.11 Script Engine

Description	Ensure that the Script Engine can execute scripts and with correct access to program data.
Purpose	The User shall trigger an event associated with a script, which should execute as expected. Test this function.
Inputs	Test Script.
Expected Outcome	Script executes correctly.

3.12 Tool Parse

Description	Post valid and invalid tool documents to the engine to parse.
Purpose	The User shall open the application which shall attempt to load a tool file. Test this function.
Inputs	Valid and Invalid tool documents.
Expected Outcome	Valid documents are correctly parsed. Invalid documents return nothing.

3.13 Tool Handling

Description	Post valid and invalid actions for tools.
Purpose	The User shall select a tool from the UI and use it, triggering an action. Test this function.
Inputs	User tool input.
Expected Outcome	Tool handler is run correctly, posting the required items to the UI.

4 Unit Tests

Below states the functions to be tested, order by which folder they're in and then which class within that folder, and are given a corresponding testID. For example, 'setParent' is in the 'RecursiveBinding' class located within 'Core' and has the testID '1.1'.

4.1 Core

4.1.1 RecursiveBinding

TestID	Function tested	Description	Assertions
1.1	setParent	Two new instances of recursiveBindings with one of them being set as a parent. Assert this parent is present and correct.	assertEquals
1.2	localContainsKey	A new instance of recursiveBindings and a new random variable. Repeat making a new byte array using random and creating a new key using this. Assert this key is present and correct.	assertEquals
1.3	localGet	A new instance of recursiveBindings and a new random. Repeat making a new byte array using random and creating a new key using this. Call the get function and assert this returns the same as was set.	assertEquals
1.4	containsKey	Three new instances of recursiveBindings, one as an instance one as a parent and one as a grandparent. Test each by setting a new key using a random byte array and assert than calling containsKey is true for each.	assertEquals
1.5	get	Three new instances of recursiveBindings, one as an instance one as a parent and one as a grandparent. Test each by setting a new key using a random byte array and assert than calling get returns what the key should be.	assertEquals
1.6	getParent	Two new instances of recursiveBindings with one of them being set as a parent. Assert than the parent is null then set the parents and assert than it's value is as expected and then re-null the parent.	assertEquals

4.1.2 Scripting

TestID	Function tested	Description	Assertions
2.1	evalString	Create a new scripting instance sending the language as Python and two new recursiveBindings instances (one for python, one for rhino). Call evalString twice once with python code and one with rhino code (both describing a simple maths sum) and then assert than the two recursiveBinding instances return the sum of the python and rhino code through get.	assertEquals

4.1.3 Threaded

TestID	Function tested	Description	Assertions
3.1	getRunning	Call function and assert Boolean return is false	assertSame
3.2	getSuspended	Call function and assert Boolean return is false	assertSame

4.1.4 Tools

TestID	Function tested	Description	Assertions
4.1	getTools	Create a new tools instance and then call getTools and assert than the retrieved tools are as expected	assertEquals

4.1.5 ToolsFactory

TestID	Function tested	Description	Assertions
5.1	startMaking-Element	Call function with all valid String and some invalid strings and assert the outputs are as expected	assertSame

4.1.6 Tool

TestID	Function tested	Description	Assertions
6.1	getName	Using a new tool instance call getName and assert it returns the name that was created	assertEquals
6.2	getID	Using a new tool instance call getID and assert it returns the ID that was created	assertEquals
6.3	getParent-Element-Scripting-Binding	Call function and assert that the returned optional is as expected	assertSame
6.4	getRealType	Call function and assert that the returned string is the current classes name	assertSame
6.5	getScriptEL	Call function and assert that the outputted array is all the elements of a script as expected	assertSame
6.6	getScripting-Bindings	Call this function and assert that it returns elementScriptBindings	assertEquals
6.7	getEvalRequired	Calling this function with an expected true and an expected false req and assert the outputs are as expected	assertFalse assertTrue
6.8	addScriptFile	Call addScriptFile from a new tool instance and assert that the error message received when sending an invalid path is as expected	assertSame

4.2 Elements

4.2.1 DocElement

TestID	Function tested	Description	Assertions
7.1	getUniqueID	Create four empty strings, to store the 4 most previous ID's. In the current variable call getUniqueID and assert it's not equal to the other three strings. Iterate this ID into the next string, that string into the next string and so on. Repeat this as many times as possible for certainty.	assertNot-Equals

4.2.2 VisualElement

TestID	Function tested	Description	Assertions
8.1	setID	Create a new random ID and send it to setID. Call getID and assert this result is equal to the ID created.	assertSame
8.2	setZInd	Create a new random z and send it to setZIng. Call getZInd and assert this result is equal to the z created.	assertSame
8.3	setFillColour	Call setFillColour with an invalid fill colour and assert the correct error message is present. Also send lots of valid string colours and assert not error message is outputted.	assertEquals and assertNotEquals
8.4	testSetOriginXY	call setOriginXY and send it a random loc then assert than the return from getOrigin is the same as what was set	assertTrue
8.5	testMakeAttrWithNS	create an attribute and then send the parts that make that attribute to makeAttrWithNS and assert the two are the same	assertEquals

4.3 XMLIO

4.3.1 DocIO

TestID	Function tested	Description	Assertions
9.1	save	Call save with allowSave as false and assert the exception is as expected. Call save with origZip as null as false and assert the exception is as expected. set these to be true and not-null and assert saveAs was called	assertSame
9.2	saveAs	send saveAs a null path and a valid pass and assert that the messages outputted are correct and as expected	assertSame
9.3	isUriInternal	Call isUriInternal sending an internal path and two external paths (http://pathhere and file://pathere) and assert if isUriInternal is true as expected	assertTrue and assertFalse
9.4	removeResource	Send an external resources and assert false. Send an internal path and assert if an exception was outputted or not	assertFalse assertSame
9.5	retreiveDoc	Send an invalid FileSystem and assert the exception is as expected. Send a valid FileSystem and assert the returned optional is as expected.	assertSame

4.3.2 IO

TestID	Function tested	Description	Assertions
10.1	getDoc	call getDoc with a test file and assert that what is returned is what is within the file	assertSame
10.2	getResource	Send an invalid path and assert that the correct exception is outputted. With a valid path, assert that the optional of the array is returned	assertSame
10.3	getResource-TempPath	With a valid path, assert that the optional of the array is returned	assertSame assertSame
10.4	canSave	Call where allowSave will be false and where allowSave will be true.	assertTrue assertFalse
10.5	pathToUriString	Send pathToUriString paths beginning with '/', 'http:/' and 'http://' and assert each one gets dealt with as expected	assertSame
10.6	maybeURI	send maybeURI and invalid URI and assert that the function deals with it as expected	assertEquals
10.7	close	assert the exception message is as expected if ZipFs isn't null and zipFs couldn't close. Assert all temp files were deleted	assertSame
10.8	isUriInternal	Call isUriInternal sending an internal path and two external paths (http://pathhere and file://pathere) and assert if isUriInternal is true as expected	assertTrue and assert-False

4.3.3 ToolIO

TestID	Function tested	Description	Assertions
11.1	isUriInternal	Call isUriInternal sending an internal path and two external paths (http://pathhere and file://pathere) and assert if isUriInternal is true as expected	assertTrue and assert-False
11.2	canSave	Call the canSave function	assertFalse

4.3.4 Parse

TestID	Function tested	Description	Assertions
12.1	testParse-DocXML_File	A test XML file should be sent to parseDocXML and check if that file is present	assertTrue
12.2	testParse-DocXML_Input-Stream	A test XML file should be sent to parseDocXML and check if that file is present	assertTrue
12.3	testParseTool-XML	A test XML file should be sent to parseDocXML and check if that file is present	assertTrue

4.4 Graphics

4.4.1 ExtShapeFactory

TestID	Function tested	Description	Assertions
13.1	makeShape	For every shape type make a new shape and assert that that shape is present and correct	assertTrue
13.2	setTextClick-Handler	Send the handler a true mouse event and assert it responds true	assertTrue
13.3	setHrefClick-Handler	Send the handler a true mouse event and assert it responds true	assertTrue
13.4	setHrefHover-EnterHandler	Send the handler a true mouse event and assert it responds true	assertTrue
13.5	SetHrefHover-ExitHandler	Send the handler a true mouse event and assert it responds true	assertTrue

4.4.2 ExtShape

TestID	Function tested	Description	Assertions
14.1	getShapeType	Set a shapeType as rectangle and assert getShapeType returns rectangle.	assertEquals
14.2	setTextClick-Handler	Send the handler a true mouse event and assert it responds true	assertTrue
14.3	setHrefClick-Handler	Send the handler a true mouse event and assert it responds true	assertTrue
14.4	setHrefHover-EnterHandler	Send the handler a true mouse event and assert it responds true	assertTrue
14.5	setHrefHover-ExitHandler	Send the handler a true mouse event and assert it responds true	assertTrue
14.6	setSize	Create a SizeObj with x,y,rot attributes as random doubles. With a ExtShape instance set size using this SizeObj and assert that the instances width, height and rot equal x,y and rot respectively.	
14.7	setFill	Call setFill with a colour and assert that getShape().getFill is equal to that colour	assertTrue
14.8	getStack	Call get stack and assert what it returns an instance of StackPane	assertTrue
14.9	getShape	Call getShape and assert than what it returns is an instance of rectanle	assertTrue
14.10	getTextFlow	Add some text to a new arrayList and call SetText with this array and alignment	assertTrue
14.11	getTextVBox	Add some text to a new arrayList and call SetText with this array and alignment	assertTrue
14.12	getWidth	Create a SizeObj with x,y,rot attributes as random doubles. Create a ExtShape instance and call setSize with this SizeObj. Assert getWidth returns x	assertEquals
14.13	getHeight	Create a SizeObj with x,y,rot attributes as random doubles. Create a ExtShape instance and call setSize with this SizeObj. Assert getHeight returns y	assertEquals
14.14	getRot	Create a SizeObj with x,y,rot attributes as random doubles. Create a ExtShape instance and call setSize with this SizeObj. Assert getRot returns rot	assertEquals

14.15	setStroke	Create a new StrokeProps with some property details, and with an instance of ExtShape setStroke with this new StrokeProp.	assertEquals
14.16	setVisualProps	Create a new VisualProp with some property details, and with an instance of ExtShape setVisualProps with this new StrokeProp.	assertEquals
14.17	setText	With and instance of ExtShape set some text and assert getText returns the same text	assertEquals

4.4.3 LocObj

TestID	Function tested	Description	Assertions
15.1	getLoc	Repeat numerous times: Create a new 2D point wth random x and y values, create a new object with these points and assert that getLoc returns the same value as the object.	assertEquals
15.2	getZ	Repeat numerous times: Create a new 2D point at 0 and a new object with that point and a random value and assert that getZ returns the same as what the random point was equal to.	assertEquals

4.4.4 SizeObj

TestID	Function tested	Description	Assertions
16.1	getX	Repeat numerous times: Create random x, y and rotation. Create a new sizeObj, sending the x, y and rotation. assert That getX returns the same X	assertEquals
16.2	getY	Repeat numerous times: Create random x, y and rotation. Create a new sizeObj, sending the x, y and rotation. assert That getY returns the same Y	assertEquals
16.3	getRot	Repeat numerous times: Create random x, y and rotation. Create a new sizeObj, sending the x, y and rotation. assert That getRot returns the same rotation value	assertEquals

4.4.5 StyledTextSeg

TestID	Function tested	Description	Assertions
17.1	setHRef	Call setHRef with an instances of FontProps and send it a target string with it's type and assert the returned target is the same as what was sent.	assertEquals
17.2	IsHref	Call isHref from an instance of FontProps and assert that it returns false.	assertEquals
17.3	getrefTarget	Set a HRef sending it a target and the type of target both correct. Assert that when calling getRefTarget this returns the same target as was sent.	assertEquals
17.4	getRefType	Set a HRef sending it a target and the type of target both correct. Assert that when calling getRefType this returns the same type as was sent.	assertEquals
17.5	getStyle	Create a new FontProps with some property details then call getStyle one an instance of FontProps and assert the two are equal	assertEquals
17.6	getString	Create a new string and send it to an instance of StyledTextSeg and assert that .getString returns the same string	assertEquals

5 Test Records

5.1 Overview

All of the tests specified above, in sections 3 and 4, were carried out and the outcomes are listed below.

5.2 Functional Test Reports

Tests specified in section 3.

Test Name	Actual outcome	Comments
Message Display	As Expected	Messages are displayed
2D Graphics Display (Including Text and Tables)	As Expected	Valid objects are displayed correctly
Image Display	As Expected	Valid images displayed
Video Display	As Expected	Valid videos are displayed
Audio Player Display	As Expected	Valid audio is outputted
Tool List Display	As Expected	Tools are available in the menu
Element Properties Display	As Expected	An objects properties are displayed once clicked on
Document Parse	As Expected	Valid documents are parsed and outputted
Document Output	As Expected	XML documents are written correctly
Event Handling	As Expected	All valid events have some sort of response
Script Engine	As Expected	Scripts are executed
Tool Parse	As Expected	Valid documents are parsed correctly and load tool files
Tool Handling	As Expected	Tool handler runs correctly

5.3 Unit Test Records

Tests specified in section 4.

Test ID	Expected outcome	Actual outcome	Comments
	1 RecursiveBinding tests		
1.1	After setting the parent, calling getParent.isPresent() is true	As expected	None
1.2	Expected result is equal to the result from calling the function	As expected	None
1.3	Calling localGet returns the same results that was pushed into the array	As expected	None
1.4	After setting the key for an instance, parent and grandparent, calling containsKey for these three should be TRUE	As expected	None
1.5	After setting the key for an instance, parent and grandparent, calling get should return the key	As expected	None
1.6	getParent should be empty and then after setting a parent a parent being present should be TRUE and then nulling it again should mean a parent being empty is TRUE	As expected	None
	2 Scripting tests		
2.1	All assertions are TRUE	As expected	None
	3 Threaded tests		
3.1	The returned Boolean should be false	As expected	None
3.2	The returned Boolean should be false	As expected	None
	4 Tools tests		
4.1	A list of correct tools is received	As expected	None

Test ID	Expected outcome	Actual outcome	Comments
5 ToolsFactory tests			
5.1	The output for each 'name' should be the directory of where the array is stored, so when there's content hat will be outputted	As expected	None
6 Tool tests			
6.1	The name that was sent when creating a tool instances is returned	Null name	Test needs amending
6.2	The ID that was sent went creating a new tool is returned	Null ID	Test needs amending
6.3	ParentElement-ScriptingBindings should be returned as expected	Null	Test needs amending
6.4	The type Tool should be outputted	As expected	None
6.5	Script element shouldn't be present	As expected	None
6.6	Scripting bindings should be output correctly	Function removed	As expected
6.7	Eval required should be true	As expected	None
6.8	An outputted error message stating that editing tools is not supported	As expected	None
7 DocElement tests			
7.1	All of the retrieved ID's should be unique and so assertNotEquals should always be true	As expected	This test isn't fool proof and doesn't prove for all values

Test ID	Expected outcome	Actual outcome	Comments
	8 VisualElement tests		
8.1	The returned ID should be the same as the ID send	As Expected	None
8.2	The returned Z should be the same as the Z send	As Expected	None
8.3	The outputted exception should be "Bad Colour String"	As expected	Could test valid colours
8.4	The original returned from getOrigin should equal the one sent in setOrigin	As expected	None
8.5	The attribute returned from makeAttrWithNS should be the same one that was created with the same attributes	As excepted	The test uses the same algorithm to create the attributes as the method, so for more solidarity could change this.
	9 DocIO tests		
9.1	No output	Null	No proof of pass/failure
9.2	An IOException for an invalid path	As expected	Should test valid path too
9.3	Return should be false for paths beginning http:// and file://		
9.4	Calling function should remove a resource from a local loc and delete the files. No output	null	No outputs to test
9.5	A documents path should be used to return an optional document	Null	Test doesn't really show if passed/failed

Test ID	Expected outcome	Actual outcome	Comments
	10 IO tests		
10.1	The test file data should equal what getDoc retrieves	As expected	None
10.4	canSave should return false	As expected	None
10.5	The paths returned from pathToUriString should be the path sent with "file://" in front except for if a path begins with http:/	As expected	None
10.8	Should be false if the path begins http:// or file://	As expected	None
	11 ToolIO tests		
11.1	Should be false if the path begins http:// or file://	As expected	None
11.2	Should return false	As expected	None
	12 Parse tests		
12.1	sending parsing-DocXML a test file should create a text file which is present and the same as the test file	As expected	None
12.2	sending parsing-DocXML a test file should create a text file which is present and the same as the test file	As expected	None
12.3	sending parsing-DocXML a test file should create a text file which is present and the same as the test file	As expected	None

Test ID	Expected outcome	Actual outcome	Comments
	13 ExtShapeFactory tests		
13.1	For each shape it should be present and then when calling getShape it should return the same shape sent to makeShape	As expected	None
13.2	A mouse event excising should be true and it should be accepted by textClickHandlerConsumer	As expected	None
13.3	A mouse event excising should be true and it should be accepted by hrefClickHandlerConsumer	As expected	None
13.4	A mouse event excising should be true and it should be accepted by hrefHovEntHandlerConsumer	As expected	None
13.5	A mouse event excising should be true and it should be accepted by hrefHovExHandlerConsumer	As expected	None

Test ID	Expected outcome	Actual outcome	Comments
	14 ExtShape tests		
14.1	ShapeType should be rectangle	As Expected	None
14.2	A mouse event excising should be true and it should be accepted by textClickHandlerConsumer	As Expected	None
14.3	A mouse event excising should be true and it should be accepted by hrefClickHandlerConsumer	As Expected	None
14.4	A mouse event excising should be true and it should be accepted by hrefHovEntHandlerConsumer	As Expected	None
14.5	A mouse event excising should be true and it should be accepted by hrefHovExHandlerConsumer	As Expected	None
14.6	getWidth should return x, getHeight should return y and getRot should return rot	As Expected	None
14.7	getFill should return the fill that was set	As Expected	None
14.8	result instanceof StackPane should be true	As Expected	None
14.9	result instanceof Rectangle should be true	As Expected	None
14.10	getTextFlow should be not null	As Expected	None
14.11	getTextVBox should be not null	As Expected	None
14.12	getWidth should return x	As Expected	None
14.13	getHeight should return y	As Expected	None
14.14	getRot should return rot	As Expected	None

Test ID	Expected outcome	Actual outcome	Comments
14.15	The instances stroke properties should equal the property details set in the stroke	As Expected	None
14.16	The instances visual properties should equal the property details set in the visualProps	As Expected	None
14.17	getText shoudl return the testText	As Expected	None

15 LocObj tests

15.1	The outputted 2D point, from getLoc, should equal the 2D point created and set as a LocObj	As expected	None
15.2	The outputted 2D point, from getZ, should equal the 2D point created and set as a LocObj	As expected	None

16 SizeObj tests

16.1	The x returned from getX should be the same sent when creating a new SizeObj	As expected	None
16.2	The y returned from getY should be the same sent when creating a new SizeObj	As expected	None
16.3	The rot returned from getRot should be the same sent when creating a new SizeObj	As expected	None

Test ID	Expected outcome	Actual outcome	Comments
	17 StyledTextSeg tests		
17.1	getReftarget returns the target sent when setting HRef	As expected	None
17.2	isHRef is false	As expected	None
17.3	getRefTarget returns the same target that was sent in setting up HRef	As expected	None
17.4	getRefType returns the same type as what was sent when setting up HRef	As expected	None
17.5	getStyle returns the same FontProps as when setting the Font-Props with default	As expected	None
17.6	getString returns the string that is sent when setting an instance of StyledTextString	As expected	None