­­­G51FSE Assessed Lab 4

|  |  |  |
| --- | --- | --- |
| **The Assemblers with Flip-Flops** |  | 09/03/2018 |

# 3. Test Plans

This section of the software specification document highlights the details of the system through the use of Unified Modelling Language and prototypes.

We have added some additional columns to the bug table. These are notes and a description for the changes are given in this paragrah.We have decided to have additional columns in the test table so we can provide more detail about the tests carried out. This will increase the traceability of the tests and make the changes easier to see. Firstly, we added **a Test ID** so that we can have a unique id for each test carried out. This will make it easier to refer back to the tests later on in the document. In addition to this, we added a column for prerequisite which will store detail about what other functions need to be performed before this test is carried out. Next, we included a column for the **actual output**. This is necessary so that we can compare if the expected output matches the actual output. If it doesn’t, a reason must be provided as to why and most likely there is an error/flow in the program. In addition to this, we added a column called ‘**test created by’**. This will hold the name of the person who created the tests and carried it out. This will increase accountability. Next, we added a column called ‘**pass or fail’**. This will make it easier to see the number of tests that has passed without the person having to read the details of each individual test. We then created a column which will store the **date** that the test has been carried out on. This will make it easier to see which version of a test is the latest if a test had been carried out multiple times. Finally, we added a column for **notes** which can be used to store additional details about the tests that does not fall into any other category. For example, if a test has failed, the notes could include a description as to why the test has faile

Email Class Test Plan

The following tests use references to constants defined in the Junit test files at the start of the document. As such for easier comparison in the place the constants are defined below and are used in the Input, Expected Output and Actual Output columns to reduce repeating data and if the test inputs change in the future the document can be updated at a single point.

**kSENDER** = [joe.bloggs@gmail.com](mailto:joe.bloggs@gmail.com)  
**kBADEMAIL1 =** joe.bloggs@   
**kBADEMAIL2 =** [joe.@bloggs](mailto:joe.@bloggs)  
**kBADEMAIL3 =** [joe.bloggs@gmail@.com](mailto:joe.bloggs@gmail@.com)  
**kBADEMAIL4 =** @joe.bloggs@gmail.com   
**kRECIPIENT** = [max.power@live.com](mailto:max.power@live.com)  
**kSUBJECT** = “RE: Lorem Ipsum”  
**kBODY1** = “Lorem ipsum dolor sit amet, consecteutur adipiscing elit.”  
**kBODY2** = “This is a test email for unit testing”   
**kBODY3** = “”   
**kBODY4** = “Test Email”  
**nullEmail** (CompanyEmail Object)  
**populatedEmail** (CompanyEmail Object)  
**noRecipientEmail** (CompanyEmail Object)  
**noEmailBody** (CompanyEmail Object)

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test ID | Function Name | Test Aim (From perspective of method/how method handles this) | Source: Spec or code inspection | Test Created By | Pre-Requisites | Inputs (+ examples) | Expected Output(s) | Actual Output | Pass / Fail | Date | Tested By | Notes (Questions/Assumptions) |
| 101 | Default Constructor | To see how the method will react when no variables are passed to the constructor. | Class Document | Ram Raja | N/A | N/A | Initialises an email ready for populating. | Stack overflow error | FAIL | 26/04/2018 | Ram Raja | Error as “emailMessage()” method returns itself rather than the variable “emailMessage”. |
| Initialised object | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 102 | Main Constructor | To see how the method will react when receiving all four string parameters in correct form. | Class Document | Ram Raja | N/A | **Sender’s Email:**  **kSENDER**  **Recipient’s Email:**  **kRECIPIENT**  **Subject:**  **kSUBJECT**  **Email Body: kBOD1** | Initialises an email ready to be sent, with a sender’s email, recipient’s email, subject and body | Stack overflow error | FAIL | 26/04/2018 | Ram Raja | Error as “emailMessage()” method returns itself rather than the variable “emailMessage”. |
| Initialised object with correct values that were passed | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 103 | Main Constructor | To test interactivity between both the Default and Main Constructor when all parameters are not populated. In this case all but the recipient’s email has been given. | Code Inspection | Ram Raja | N/A | Recipient’s email is null. | Initialises an email with all but the recipient’s email given, ready to be populated. | Stack overflow error as | FAIL | 26/04/2018 | Ram Raja | Error as “emailMessage()” method returns itself rather than the variable “emailMessage”. |
| Initialised object without email | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 104 | Main Constructor | To test interactivity between both the Default and Main Constructor when only one parameter is given, in this case the sender’s email. | Code Inspection | Athullya Roy | N/A | **Sender’s Email:**  **kSENDER** | Initialises an email with the sender’s email; with recipient’s email, subject and body as null, ready to be populated. | Stack overflow error | FAIL | 26/04/2018 | Ram Raja | Error as “emailMessage()” method returns itself rather than the variable “emailMessage”. |
| Initialised object with just senders email set | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 105 | Getter for Sender’s email address | To ensure the getter outputs the sender’s email when specified. | Class Document | Ram Raja | N/A | **Sender’s Email:**  **kSENDER** | Returns the sender’s email as a string. | The given sender’s email is returned. | PASS | 26/04/2018 | Ram Raja |  |
| 106 | Getter for Sender’s email address | To test how the method will respond when the sender’s email is not set. | Class Document | Athullya Roy | N/A | N/A | Returns sender’s email as null. | The sender’s email is returned as null. | PASS | 26/04/2018 | Ram Raja |  |
| 107 | Getter for Recipient’s Email Address | To ensure the getter outputs the recipient’s email when specified. | Class Document | Ram Raja | N/A | **Recipient’s Email:** **kRECIPIENT** | Returns the recipient’s email as a string. | The given recipient’s email is returned. | PASS | 26/04/2018 | Ram Raja |  |
| 108 | Getter for Recipient’s Email Address | To test how the method will respond when the recipient’s email is not set. | Class Document | Athullya Roy & Ram Raja | N/A | N/A | Return’s recipient’s email as null. | The recipient’s email is returned as null. | PASS | 26/04/2018 | Ram Raja |  |
| 109 | Getter for Subject Line | How the method will respond when there is a subject line. | Class Document | Athullya Roy & Ram Raja | N/A | **Subject:**  **kSUBJECT** | **kSUBJECT** | **kSUBJECT** | PASS | 26/04/18 | Athullya Roy |  |
| 110 | Getter for subject Line | How the method will respond when there is no subject line | Class Document | Athullya Roy & Ram | N/A | null | null | null | PASS | 26/04/18 | Athullya Roy |  |
| 111 | Getter for Email body | How the method will respond when body text has been given. | Class Document | Athullya Roy & Ram | N/A | **Message:**  **kBODY1** | **kBODY1** | Stack over flow | FAIL | 26/04/18 | Athullya Roy | StackOverFlow error |
| **kBODY1** | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 112 | Getter for Email body | How the method will respond when there is no body text | Class Document | Athullya Roy & Ram | N/A | null | null | Stack overflow error | FAIL | 27/04/18 | Athullya Roy | StackOverFlow error |
| Null | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 113 | Setter for Senders Email | To see if it could take a string as a parameter, ensuring that it follows the email format | Class Document | Athullya Roy & Ram | N/A | **Sender:**  **kSENDER** | null | null | PASS | 27/04/18 | Athullya Roy |  |
| 114 | Setter for Senders Email | To check if class variable is set to null and it fails if invalid email address is entered. | Class Document | Athullya Roy & Ram | N/A | Joe.bloggs | Function returns false | Null | FAIL | 27/04/18 | Athullya Roy | Fails because it doesn’t return the correct type. Also assumes fails means returning error message |
| 115 | Setter for Receivers Email | To see if it could take a string as a parameter, ensuring that it follows the email format | Class Document | Athullya & Ram | N/A | Recipient:  kRECIPIENT | Null | Null | PASS | 27/04/18 | Athullya Roy |  |
| 116 | Setter for Receivers Email | To check if class variable is set to null and it fails if invalid email address is entered. | Class Document | Athullya & Ram | N/A | “Max.power” | Warning Message – Return false | Null | FAIL | 27/04/18 | Athullya Roy | Fails because it doesn’t return the correct type. Also assumes fails means returning error boolean value false. |
| 117 | Set method for Subject Line | To check if it takes a String as a parameter. | Class Document | Athullya & Ram | N/A | **Subject:**  **kSUBJECT** | Null | Null | PASS | 27/04/18 | Athullya Roy |  |
| 118 | Set method for Subject Line | To check if warning produced if null is passed. | Class Document | Athullya Roy & Ram | N/A | null | Warning Message – return false | null | FAIL | 27/04/18 | Athullya Roy | Fails because system doesn’t respond to null being passed to function. |
| 119 | Setter for Email Message Body  **(setMessage)** | Test the method correctly sets the email message and then compare that to what is retrieved from getMessage method | Class Document | Aidan Reed | Uses **noEmailBody** Object with Message set using setMessage method and **kBODY1** Constant | **Message:**  **kBODY1** | **kBODY1**  After calling the getMessage method once the message has been set | Stack overflow error | FAIL | 26.04.2018 | Aidan Reed | Unable to determine if message was set correctly as when trying to retrieve the message using emailMessage method received a stack overflow error. |
| **kBODY1** | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 120 | Setter for Email Message Body  **(setMessage)** | Test the method correctly sets the email message to an empty string “” and then compare that to what is retrieved from getMessage method | Class Document | Aidan Reed | Uses **noEmailBody** Object with Message set using setMessage method and **kBODY3** Constant | **Message:**  **kBODY3** | **kBODY3**  After calling the getMessage method once the message has been set | Stack overflow error | FAIL | 26.04.2018 | Aidan Reed | Unable to determine if message was set correctly as when trying to retrieve the message using emailMessage method received a stack overflow error. |
| **kBODY3** i.e. an empty string | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG001 |
| 121 | Check Validity of Email Function  **(isValid)** | Tests the email object with no attributes set in the constructor meaning they are null | Class Document | Aidan Reed | Uses **nullEmail** Object | N/A | False | False | PASS | 26.04.2018 | Aidan Reed | The worst case where no values are set |
| 122 | Check Validity of Email Function  **(isValid)** | Tests the email object with All attributes set in the constructor | Class Document | Aidan Reed | Uses **populatedEmail** object created with **kSENDER**, **kRECIPIENT**, **kSUBJECT**, and **kBODY1** constants | **Sender**:  **kSENDER**  **Recipient**:  **kRECIPIENT**  **Subject:**  **kSUBJECT**  **Message**:  **kBODY1** | True | True | PASS | 26.04.2018 | Aidan Reed | The best case where all values are set |
| 123 | Check Validity of Email Function  **(isValid)** | Tests the email object with one attribute set – the from address | Class Document | Aidan Reed | Uses **nullEmail** object and setFrom method passing **kSENDER** constant | **Sender:**  **kSENDER** | False | False | PASS | 26.04.2018 | Aidan Reed | The following 4 tests of similar nature test the individual cases for each attribute to ensure all attributes are checked in the isValid Function |
| 124 | Check Validity of Email Function  **(isValid)** | Tests the email object with one attribute set – the To address | Class Document | Aidan Reed | Uses **nullEmail** object and setTo method passing **kRECIPIENT** constant | **Recipient**  **kRECIPIENT** | False | False | PASS | 26.04.2018 | Aidan Reed |  |
| 125 | Check Validity of Email Function  **(isValid)** | Tests the email object with one attribute set – the Subject | Class Document | Aidan Reed | Uses **nullEmail** object and setSubject method passing **kSUBJECT** constant | **Subject:**  **kSUBJECT** | False | False | PASS | 26.04.2018 | Aidan Reed |  |
| 126 | Check Validity of Email Function  **(isValid)** | Tests the email object with one attribute set – the message body | Class Document | Aidan Reed | Uses **nullEmail** object and setMessage method passing **kBODY1** constant | **Message**:  **kBODY1** | False | False | PASS | 26.04.2018 | Aidan Reed |  |
| 127 | toString() override  **(toString)** | Checks the toString override method returns a value of type string | Class Document | Aidan Reed | Uses **populatedEmail** object created with **kSENDER**, **kRECIPIENT**, **kSUBJECT**, and **kBODY1** constants | N/A | (String) | (String) | PASS | 26.04.2018 | Aidan Reed |  |
| 128 | toString() override  **(toString)** | Checks the correct subject is returned when set during the test | Class Document | Aidan Reed | Uses **nullEmail** object and setSubject method passing **kSUBJECT** constant | **Subject:**  **kSUBJECT** | **kSUBJECT** | **kSUBJECT** | PASS | 26.04.2018 | Aidan Reed |  |
| 129 | toString() override  **(toString)** | Checks the correct subject is returned when set during the test to an empty string | Class Document | Aidan Reed | Uses **nullEmail** object and setSubject method passing “” empty string | Subject:  “” | “[no subject]” | “no subject” | PASS | 26.04.2018 | Aidan Reed |  |
| 130 | toString() override  **(toString)** | Checks the correct subject is returned when not set i.e null | Class Document | Aidan Reed | Uses **nullEmail** object | N/A | “[no subject]” | Null pointer exception | FAIL | 26.04.2018 | Aidan Reed | This fails as the method tries to return null as type string and causes a null pointer exception. Although the class description does not include this behavior the test was included to find potential bugs that cause the system to crass |
| “[no subject]” | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG002 |
| 131 | Check class member variables are private | Checks from address, to address, subject line and email message variables are all set to private | Code review | Ram Raja & Aidan Reed | N/A | N/A | True for all 4 variables | True | PASS | 27.04.2018 | Aidan Reed |  |
| 132 | Check validity of Full email in setFrom **(setFrom)** | Checks the validity of the email address instead of just checking for a @ | Code Review | Ram Raja | Uses **nullEmail** object | **Sender**  **kBADEMAIL1 kBADEMAIL2 kBADEMAIL3 kBADEMAIL4** | Null when calling fromAddress | **kBADEMAIL1 kBADEMAIL2 kBADEMAIL3 kBADEMAIL4** | FAIL | 27/04/2018 | Aidan Reed |  |
| **Null** | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG003 |
| 133 | Check validity of Full email in setTo **(setTo)** | Checks the validity of the email address instead of just checking for a @ | Code Review | Ram Raja | Uses **nullEmail** object | **To**  **kBADEMAIL1 kBADEMAIL2 kBADEMAIL3 kBADEMAIL4** | Null when calling toAddress | **kBADEMAIL1 kBADEMAIL2 kBADEMAIL3 kBADEMAIL4** | FAIL | 27/04/2018 | Aidan Reed |  |
| **Null** | PASS | 27/04/2018 | Aidan Reed | Test passed after change to code with ChangeID CHG003 |

Change Log

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Change ID | GIT Commit ID | Bug ID | Description of Change | Files Changed | Date Received | Date Changed | Initiator By (Who Changed) | Change Checked By | Notes |
| CHGE001 | D2c7cf7 | BUG001 | Return statement of emailMessage() getter method changed from “emailMessage()” method to “emailMessage” variable. | CompanyEmail (emailMessage) | 26/04/2018 | 27/04/2018 | Aidan Reed |  | Changed code removing method call and setting return value to emailMessage member variable |
| CHGE002 | 5aa2fcf | BUG003 | Null Pointer exception when using toString when the subject has not been set. Add a condition to the method to check if null and print “[no subject set]” if null | CompanyEmail toString() | 26/04/2018 | 27/04/2018 | Aidan Reed |  | Added a null check to toString method to prevent null pointer exception |
| CHGE003 | Edd9154 | BUG001 | SetTo and From methods check for @ in address but not position. Include regular expression to validate combinations of addresses. To do so I have added a new function called Email Parser which takes an address as input and returns true or false depending on whether it is valid or not. I have updated setTO and setFom methods to use the boolean result of email parser in the if statement | CompanyEmail | 26/04/018 | 27/04/2018 | Aidan Reed |  |  |

Bug Fix List

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Problem ID | Problem Description | Line of code - | Test ID  (if created) | Change ID | Proposed Fix | Priority  High, Med, Low | Date Problem identified | Fixed? (Yes/ No) | Date Problem fixed | Who identified/fixed the test? | Who Fixed it | Does the bugged code relate to other functions? | Notes |
| BUG001 | The setFrom() and setTo() email methods only check for a “@” being present in the entered string, not at any specific position. Therefore, an incorrect email address could be entered. i.e. “psyrr1nottinghamac.uk@” | Line 39 Line 45 | 132  133 | CHGE003 | Specify clearly where the @ should be. | Med | 09/03/2018 | Yes |  | Ram Raja | Aidan Reed |  | Added new method email parser to validate emails using regular expressions |
| BUG002 | emailMessage() method returns itself, not the value stored in the variable “emailMessage”. The method should be performing as a getter for the variable “emailMessage”. | Line 35 | 101 102 103 104 111 112 119 120 | CHGE001 | Change return statement to the variable “emailMessage”, instead of the method “emailMessage()”. | High | 26/04/2018 | Yes | 27/04/2017 | Ram Raja Athullya Roy Aidan Reed | Aidan Reed | Both main and default constructors within the CompanyEmail class. | Found whilst performing tests on default constructor – Ram Raja Found this when performing tests 119-120 -Aidan Reed |
| BUG003 | Null Pointer exception when trying to print the subject of a email that has not been set | Line 73 | 130 | CHGE02 | Perform a check to see if the member variable is null and return the no subject string | High | 26.04.2018 | Yes | 27/04/2018 | Aidan Reed | Aidan Reed |  | Although the class description does not include this behavior the test was included to find potential bugs that cause the system to crash  27/04/2018 – Added OR condition to toString if statement |