# Test Plan and Cases (TPC)

**QuickShip**

**DEV-T**

**Trinh Nguyen (developer)**

**Emmanuel Mendoza (developer)**

**Victor Fateh (developer)**

**David Navarro (developer)**

**03/12/17**

# Version History

| Date | Author | Version | Changes made | Rationale |
| --- | --- | --- | --- | --- |
| 03/13/17 | DN | 1.0 | * Converted original template to test document. | * Initial draft for use with QuickShip |

# Table of Contents

[Test Plan and Cases (TPC) i](#_Toc219626969)

[Version History ii](#_Toc219626970)

[Table of Contents iii](#_Toc219626971)

[Table of Tables iv](#_Toc219626972)

[Table of Figures v](#_Toc219626973)

[1. Introduction 6](#_Toc219626974)

[2. Test Strategy and Preparation 7](#_Toc219626975)

[2.1 Hardware preparation 7](#_Toc219626976)

[2.2 Software preparation 7](#_Toc219626977)

[2.3 Other pre-test preparations 7](#_Toc219626978)

[2.4 Requirements Traceability 7](#_Toc219626979)

[3. Test Identification 8](#_Toc219626980)

[3.1 Test Identifier 8](#_Toc219626981)

[3.2 Test Identifier 10](#_Toc219626982)

[4. Resources and schedule 11](#_Toc219626983)

[4.1 Resources 11](#_Toc219626984)

[4.2 Staffing and Training Needs 11](#_Toc219626985)

[4.3 Schedule 11](#_Toc219626986)

# Table of Tables

[Table 1: Requirements Traceability Matrix 8](#_Toc219626987)

[Table 2: TC-01-01 Check Bluetooth Adapter Exists 9](#_Toc219626988)

[Table 3-5: TC-02-01 – TC-02-03: Test Bluetooth Data Packet 9](#_Toc219626989)

[Table 6-11: TC-03-01 – TC-03-06: Test quickShip Model Board 11](#_Toc219626990)

# Table of Figures

[Figure 1: <Figure Title> 21](#_Toc55215008)

### Introduction

As the QuickShip project grows, it is important that all features function to expectation. The vision behind QuickShip is to connect people who are present with each other through a fast-paced strategy game. As the name entails, games should be quick, which inherently means the app is stable. The focus of the testing is to ensure that the app delivers the vision behind QuickShip. The scope of testers will mainly revolve around the coders themselves, due to lack of resources. The testers will utilize methods of testing such as Black Box Testing, Unit Testing, and Incremental Integration Testing.

### Test Strategy and Preparation

Utilizing Agile inherently structures the testing schedule along with scrum meetings. Scrum meetings provide the perfect time to demo the functionality of the recently implemented feature. At this time, we will also allow the other coders to intentionally find bugs and to see if the app delivers the desired functionality. By utilizing the story points fundamental in Agile, we can also use those points to determine the priority of testing new features. The logic is simple, if a story was deemed difficult to complete, it is reasonable to prioritize testing said story over others. In order to keep track of test history, there will be a shared document for coders to keep a list of bugs found during testing. The test environment will be either through Android emulators or an Android device itself. This is made possible by Android Studio.

#### Hardware preparation

Hardware for testing purposes will either be the development machine or Android devices. Within most of our development machines will be an emulator to run Android Nougat and test our app. Android devices will be the most useful for testing, as most devices include required functionality such as Bluetooth. We will just need to ensure that the Android devices are updated to an acceptable version in order for our application to run.

#### Software preparation

As stated before, emulation of Android Nougat can be a method of testing the QuickShip application. However, since emulation does not give us access to some hardware dependent functionality, such as Bluetooth, we will mostly utilize emulation for architectural and GUI testing rather than testing gameplay. This testing will rely on the debugging tools provided by Android Studio.

#### Other pre-test preparations

As expected, all coders and testers should have their environment set up and functioning to expectation. There should also be a document or another form of record to keep testing history such as result and new bugs to fix. Developers should get their hands on a tester Android device, as said device would be the most useful for testing purposes.

#### Requirements Traceability

*Table 1: Requirements Traceability Matrix*

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Verification Type** | **Test Case ID (if applicable)** |
| Check Bluetooth Adapter Exists | Testing | TC-01 |
| Test Bluetooth Data Packet | Testing | TC-02 |
| Test quickShip Model Board | Testing | TC-03 |

### Test Identification

#### Test Identifier

TC-01 Check Bluetooth Adapter Exists

##### Test Level

Hardware Level Test

##### Test Class

BluetoothInstrumentedTest.java

##### Test Completion Criteria

The test case asserts as true, detecting that the device does indeed have a Bluetooth adapter.

##### Test Cases

Table 2: TC-01-01 checkBluetoothAdapterExists\_test

|  |  |
| --- | --- |
| Test Case Number | TC-01 |
| Test Item | Android Bluetooth Adapter |
| Test Priority | M |
| Pre-conditions | Have Android device connected via usb cable. |
| Post-conditions | Test case successfully executes |
| Input Specifications | Connect Android device. Run test. |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert Equals |
| Assumptions and Constraints | Android device is used, USB cable is fully functional, Android Studio detects device. |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

#### Test Identifier

TC-02 Test Bluetooth Data Packet

##### Test Level

Software Level Test

##### Test Class

quickShipBluetoothPacketsToBeSent\_UnitTests.java

##### Test Completion Criteria

Bluetooth Data packets can be successfully created and retrieved.

##### Test Cases

Table 3: TC-02-01 quickShipBluetoothPacketsToBeSent\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-02-01 |
| Test Item | SetAndGet\_PacketType |
| Test Priority | M |
| Pre-conditions | Set data packet with TURN\_DONE with Boolean value true |
| Post-conditions | Retrieve data packet with TURN\_DONE |
| Input Specifications | TURN\_DONE enum and true Boolean to data packet |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert Equals |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

Table 4: TC-02-02 quickShipBluetoothPacketsToBeSent\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-02-02 |
| Test Item | SetAndGet\_ChatMessage |
| Test Priority | M |
| Pre-conditions | Set data packet with CHAT and String message. |
| Post-conditions | Retrieve String message and check equality |
| Input Specifications | CHAT enum and String message |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert True |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

Table 5: TC-02-03 quickShipBluetoothPacketsToBeSent\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-02-03 |
| Test Item | setAndGet\_QuickShipBoard |
| Test Priority | M |
| Pre-conditions | Set quickShip Model Board, convert to byte array, convert to data packet. |
| Post-conditions | Retrieve byte array, convert to model board, check equality |
| Input Specifications | quickShip Model Board |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert True |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

**3.3 Test Identifier**

TC-03 Test Model Board

**3.3.1 Test Level**

Software Level Test

**3.3.2 Test Class**

quickShipModelBoard\_UnitTests.java

**3.3.3 Test Completion Criteria**

quickShip Model Board can be properly set and retrieved.

**3.3.4 Test Cases**

Table 6: TC-03-01 quickShipModelBoard\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-03-01 |
| Test Item | EmptyBoardTest |
| Test Priority | M |
| Pre-conditions | Create quickship model board |
| Post-conditions | Test if ships are placed |
| Input Specifications | None |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert False |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

Table 7: TC-03-02 quickShipModelBoard\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-03-02 |
| Test Item | SetAllFiveShips |
| Test Priority | M |
| Pre-conditions | Create quickship model board and set all five ships |
| Post-conditions | Test if all five ships are test |
| Input Specifications | Set anchor index, ship, and positioning. |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert True |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

Table 8: TC-03-03 quickShipModelBoard\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-03-03 |
| Test Item | AddAndRemove\_Ships |
| Test Priority | M |
| Pre-conditions | Set all five ships and remove all five ships |
| Post-conditions | Check if all ships are not set. |
| Input Specifications | Set all five ships and remove all five ships |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert False |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

Table 9: TC-03-04 quickShipModelBoard\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-03-04 |
| Test Item | HorizontalAndVerticalShipCollisionTest |
| Test Priority | M |
| Pre-conditions | Set a horizontal ship, then set a vertical ship, then set horizontal ship |
| Post-conditions | Should not be able to place ship if collision exists. |
| Input Specifications | Setting three ships, one on top of the other. |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert True |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

Table 10: TC-03-05 quickShipModelBoard\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-03-05 |
| Test Item | checkGameOverTest |
| Test Priority | M |
| Pre-conditions | Set all five ships, check game over, then hit all five ships and check game over |
| Post-conditions | checkGameOver fails then checkGameOver succeeds. |
| Input Specifications | Set five ships, then set each ship as hit. |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert True |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

Table 11: TC-03-06 quickShipModelBoard\_UnitTests

|  |  |
| --- | --- |
| Test Case Number | TC-03-06 |
| Test Item | board2byteArray2boardTest |
| Test Priority | M |
| Pre-conditions | Set all five ships, convert to byte array, then convert back to board |
| Post-conditions | Original board should match restored board. |
| Input Specifications | Set five ships, then convert to byte array and restore to board. |
| Expected Output Specifications | Test Pass. |
| Pass/Fail Criteria | Assert True |
| Assumptions and Constraints | N/A |
| Dependencies | AndroidJUnit4 |
| Traceability | All results to be printed on Android Monitor console |

### Resources and schedule

Some immediate resources we have at our disposable include the coders themselves. It is likely, friends and acquaintances of said coders would be included as human resources by extension. Time varies between each person involved with testing, but we can assume a lower bound of 4 hours a week towards testing. Testing will usually take place on Fridays, since these are the days most available to majority of the coding team. Some testing may occur outside these hours. Budget will remain low since the development team is composed of college students.

#### Resources

The most needed resource for testing is people to test the product. This is where social connections of the coders will come into play, as it is easiest to recruit friends for beta testing. Preferably, these appointed testing duties have access to a physical Android device. In cases where we aim to test a specific function of the application, we will also include a list of instructions for the testers to follow with the intention of pushing the functionality through different test cases.

#### Staffing Needs

#### Schedule

Table 4: Testing Schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Test Identifier** | **Responsible person** | **Resources** | **Training needs** |
| *05/10.17* | *TC-01-01 to TC-03-06* | *Emmanuel Mendoza* | *Report test data sets,*  *JUnit* | *N/A* |
|  |  |  |  |  |
|  |  |  |  |  |