# Acceptance Test Plan

## Apologies

Thomas Cuevas

Benjamin Greenfield

Bruce Johnson

Rory O’Kane

Denisa Qori

February 21, 2013

Revision 0

## Revision History

| Name | Date | Reason for Change | Version |
|---|---|---|---|
| Rory O’Kane,<br>Benjamin Greenfield,<br>Denisa Qori,<br>Bruce Johnson,<br>Thomas Cuevas | February 20, 2014 | Initial Version | 0 |

## Table of Contents

# 1. Introduction

## 1.1. Background

This document provides the process for completing the acceptance testing of the Apologies software system. The processes and test cases were influenced by the Apologies Software Requirements Specifications (see § 1.2.), which details the functionality of the Apologies system.

The Apologies program is a virtual board game that mimics the popular board game Sorry!. Apologies is played with two to four players. Each player, using his or her pawns, tries to travel around the board faster than any other player. The game is regulated by instructions that each player gets by picking a card during his or her turn. These instructions specify the way each player is supposed to move his or her pawns during that round.

Sorry! has not been implemented for the home computer since 1998, and Apologies will provide a more modern version of that implementation.

### 1.2. References

- **Apologies Software Requirements Specification**
  Cuevas, Thomas, Benjamin Greenfield, Bruce Johnson, Rory O'Kane, and Denisa Qori. *Apologies SRS*. *Google Code*. N.p., 7 Feb. 2014. Web. 17 Feb. 2014. <https://code.google.com/p/apologies/source/browse/doc/Apologies_SRS.pdf>.
- **Apologies Issue Tracking System**
  Cuevas, Thomas, Benjamin Greenfield, Bruce Johnson, Rory O'Kane, and Denisa Qori. *Apologies Issue Tracking System*. *Apologies Issues*. N.p., 16 Jan. 2014. Web. 17 Feb. 2014. <https://code.google.com/p/apologies/issues/list>.

### 1.3. Glossary

| | |
|---|---|
| **SRS :** | Software Requirements Specification, a complete description of the behavior of the system to be developed. |
| **Test Case :** | A set of conditions and/or variables under which it can be determined if the application is working as it was originally established to. |
| **Unit Test :** | A procedure used to evaluate if individual units of the source code are working as expected. |
| **Player :** | Any user in the system who is playing the Apologies game. |
| **Test Team Leader:** | The person in charge of managing the test team members and tasks. |

**Project Leader:** The person in charge of managing the tasks and members of the entire project.

**Pawn:** A player's pieces. These pieces are moved around the board during a player's turn. Each player has four pawns.

**Safe Zone:** A series of tiles next to each player’s Home where pawns cannot be targeted by an 11 or a My Apologies card.

**Home:** The end goal of the game. A player wins the game when he or she moves all of his or her pawns to the location labeled Home on the game board.

**Start:** The start location for all pawns.
Square: A tile on the game board. A square can only be occupied by one pawn at a time.

**Slide:** A region on the game board designated by a triangle tile. If a pawn lands on the start of a slide that is not the same color as the pawn, then the pawn may move to the end of the slide.

## 2. Test Approach and Constraints

This section describes the objectives, structure, and constraints for the acceptance test plan of the Apologies system.

### 2.1. Test Objectives

The main objective of the acceptance test plan is to verify that the system follows all requirements and functionality specified in the Apologies SRS. The system will be ready for deployment when all test cases are passed.

### 2.2. Test Structure

Each test case in the acceptance test plan will be derived from the use cases specified in the Apologies SRS. Each test case will have three parts: a precondition, actions to complete the test case, and a post condition. The precondition specifies the state of the software before the test case begins. The actions to complete the test case is an itemized list of steps to be taken. The post condition is the expected state of the software after the actions are performed.

### 2.3. Test Constraints

The Acceptance Test Plan is focused only on testing that the Apologies system meets the requirements and functionality specified in the requirements document. System design and implementation details are not a concern of the Acceptance Test Plan.

# 3. Test Assumptions and Exclusions

This section provides more detail about which functionality of Apologies will be covered by the Acceptance Test Plan, and which functionality will not be covered by the Acceptance Test Plan.

## 3.1. Test Assumptions

It is assumed that all issues covered by the Acceptance Test Plan were also previously addressed by the unit tests, integration tests, and system tests of the Apologies system.

The Acceptance Test Plan will cover:

- The functional requirements of the system listed in the Software Requirements Specification.
- The consistency of user related system documentation.

## 3.2. Test Exclusions

It is assumed that all issues not covered by the Acceptance Test Plan were previously addressed by the unit tests, integration tests, and system tests of the Apologies system.

The Acceptance Test Plan will not cover:

- The non-functional requirements of the system listed in the Software Requirements Specification.
- Structural integrity of the source code.

# 4. Entry and Exit Criteria

This section lists the criteria that must be satisfied for the Acceptance Test Plan to begin, as well as the criteria that must be satisfied for the Acceptance Test Plan to end.

## 4.1. Entry Criteria

The Acceptance Test Plan can begin after the following criteria have been met:

- All other unit, system, and integration tests have been successfully completed.
- A proper testing environment that meets the hardware and software requirements specified in § 2.2. of the Apologies SRS is available.
- A copy of the latest version of the Apologies SRS is available.
- The most recent version of the Apologies system is available and installed.
- Consent from the project leader.
- Consent from the test team leader.

### 4.2. Exit Criteria

The Acceptance Test Plan should be stopped after either of the following:

- All Priority 1 requirements were tested without deviation from the specified behavior (Success).
- One or more Priority 1 requirements that were tested deviated from the specified behavior in the SRS (Failure).

## 5. Testing Participants

This section describes the roles and responsibilities of all parties involved in the Acceptance Test Plan, as well as the process for reporting the test results and any subsequent issues.

### 5.1. Roles and Responsibilities

- **Test Team Leader:** Bruce Johnson
- **Testers:** Thomas Cuevas, Benjamin Greenfield, Rory O'Kane, Denisa Qori

### 5.2. Training Requirements

All parties involved in the Acceptance Test Plan should be comfortable with standard Java application interfaces. All parties should be familiar with the Apologies user interface, as well as the Apologies SRS. Previous knowledge of the Hasbro game Sorry! would be helpful, but is not required.

### 5.3. Problem Reporting

Any problem found by a tester must be documented and reported to the test team leader via the team's issue tracking system. The test team leader will meet with the development team and testing teams at the conclusion of each sprint to review each reported issue and its severity.

### 5.4. Progress Reporting

The testing team will compile the Acceptance Test Plan report after the Acceptance Test Plan has been stopped due to either of the criteria listed in § 4.2. The report will be submitted to the Project Team Leader and will be reviewed at the end of each sprint.

## 6. Test Cases

These test cases are designed to test the functionality of use cases as specified in the SRS.
Their format is as following:

| | |
|---|---|
| ID: | the identifying number of the test case |
| Name: | the name of the test case |

| | |
|---|---|
| Requirement(s): | the requirements for this case as specified in the SRS |
| Description: | a description of the use case |
| Precondition(s): | conditions needed to exist before the test case is initiated |
| Actions(s): | the actions the tester is expected to perform |
| Post condition(s): | the expected effect of the test case |

## Test cases for the Player

## 6.1. Create New Game

### 6.1.1. Successful setup

| | |
|---|---|
| **ID** | TC 1 |
| **Name** | Successful Setup |
| **Requirement(s)** | 0100, 0200 |
| **Description** | The player sets up a new game with two or more players, one is selected to go first, and each player is given a name. |
| **Precondition(s)** | ● The player has the Apologies program running.<br>● The player is at the Main Menu. |
| **Action(s)** | 1. The player clicks the “New Game” button from the main menu, and is taken to the setup screen.<br>2. The player selects the checkbox next to at least two colors.<br>3. The radio button next to one of the checked colors is selected, indicating that color will go first.<br>4. The player enters a name in the text field next to each checked color.<br>5. The player clicks the “Start” button. |
| **Postcondition(s)** | ● The player is taken to the game board screen.<br>● The player who was selected to go first has their name displayed above the deck.<br>● The deck is highlighted indicating a card needs to be drawn |

### 6.1.2. Not enough players

| | |
|---|---|
| **ID** | TC 2 |
| **Name** | Not enough players |
| **Requirement(s)** | 0100, 0200 |

| | |
|---|---|
| **Description** | The player attempts to setup a new game with only one player and setup fails. |
| **Precondition(s)** | ● The player has the Apologies program running.<br>● The player is at the Main Menu. |
| **Action(s)** | 1. The player clicks the “New Game” button from the main menu, and is taken to the setup screen.<br>2. The player selects the check box next to one color.<br>3. The radio button next to the checked color is selected, indicating that player will go first.<br>4. The player enters a name in the text field next to the checked color.<br>5. The player clicks the “Start” button. |
| **Postcondition(s)** | ● A dialog box appears informing the player that at least two colors must be chosen.<br>● The player remains on the setup screen. |

### 6.1.3. No Names

| | |
|---|---|
| **ID** | TC 3 |
| **Name** | No Names |
| **Requirement(s)** | 0100, 0200 |
| **Description** | The player sets up a new game with two players, one is selected to go first, but no names are entered. |
| **Precondition(s)** | ● The player has the Apologies program running.<br>● The player is at the Main Menu. |
| **Action(s)** | 1. The player clicks the “New Game” button from the main menu, and is taken to the player setup screen.<br>2. The player selects the checkbox next to two colors.<br>3. The player selects the radio button next to one of the checked colors, indicating that player will go first.<br>4. The player clicks the “Start” button. |
| **Postcondition(s)** | ● A dialog box appears notifying the player that each player must be given a name in order to proceed. |

## 6.2. Quitting the Application

### 6.2.1. Quit from main menu

| | |
|---|---|
| **ID** | TC 4 |
| **Name** | Quit from main menu |
| **Requirement(s)** | 0700 |
| **Description** | While at the main menu, the player wishes to quit. |
| **Precondition(s)** | ● The player has the Apologies program running.<br>● The player is at the main menu. |
| **Action(s)** | 1. The player clicks “Quit” from the main menu. |
| **Postcondition(s)** | ● The program terminates |

### 6.2.2. Quit from setup screen

| | |
|---|---|
| **ID** | TC 5 |
| **Name** | Quit from setup screen |
| **Requirement(s)** | 0700 |
| **Description** | While at the setup screen, the player wishes to quit. |
| **Precondition(s)** | ● The player has the Apologies program running.<br>● The player is at the setup screen. |
| **Action(s)** | 1. The player clicks “Quit” from the setup screen |
| **Postcondition(s)** | ● The program terminates |

### 6.2.3. Quit game in progress

| | |
|---|---|
| **ID** | TC 6 |
| **Name** | Quit game in progress |
| **Requirement(s)** | 0700 |
| **Description** | While a game is in progress, the player wishes to quit |
| **Precondition(s)** | ● The player has the Apologies program running. |

| | |
|---|---|
| | ● A game is in progress. |
| **Action(s)** | 1. The player clicks “Quit” from the in-game menu. |
| **Postcondition(s)** | ● The program terminates. |

## 6.3. End Game

### 6.3.1. End Game

| | |
|---|---|
| **ID** | TC 7 |
| **Name** | End Game |
| **Requirement(s)** | 0700 |
| **Description** | During a game a player can choose the “End Game” option from the “Game” menu to end the game early. |
| **Precondition(s)** | ● A player is at the game board screen. |
| **Action(s)** | 1. A player clicks the “Game” drop down menu.<br>2. A player clicks the “End Game” option from the menu, |
| **Postcondition(s)** | ● The player is brought back to the main menu. |

## 6.4. Game Won

### 6.4.1. Restart Game

| | |
|---|---|
| **ID** | TC 8 |
| **Name** | Restart Game |
| **Requirement(s)** | 300, 400 |
| **Description** | After a winner has been declared, the player wishes to restart the game with the same players and colors. |
| **Precondition(s)** | ● One player has collected all of his pieces in his “Home” section.<br>● The screen shows the final position of all pieces. |
| **Action(s)** | 1. The system shows a dialog box, prompting the user to either click “Restart” or “End Game”. |

| | |
|---|---|
| | 2. The user clicks “Restart”. |
| **Postcondition(s)** | ● The game board screen appears with the previous players, names, colors, and positions. |

### 6.4.2. Return to Menu

| | |
|---|---|
| **ID** | TC 9 |
| **Name** | Return to Menu |
| **Requirement(s)** | 300, 400 |
| **Description** | After a winner has been declared, the player wishes to end the game. |
| **Precondition(s)** | ● One player has collected all his pieces in his “Home” section.<br>● The screen showing the final position of all pieces is showing. |
| **Action(s)** | 1. The system shows a dialog box, prompting the user to either click “Restart” or “End Game”.<br>2. The user clicks “End Game”. |
| **Postcondition(s)** | ● The player is brought back to the main menu. |

## 6.5. Take Turn

### 6.5.1. Draw card

| | |
|---|---|
| **ID** | TC 10 |
| **Name** | Draw card |
| **Requirement(s)** | 0500 |
| **Description** | It is the player’s turn and he or she draws a card to make a move. |
| **Precondition(s)** | ● It is the beginning of a player’s turn and a card has not been drawn yet, or the player has drawn a card that directed him or her to draw again. |
| **Action(s)** | 1. The player clicks the deck. |

| | |
|---|---|
| **Postcondition(s)** | ● A card is drawn. |

### 6.5.2. Move forward

| | |
|---|---|
| **ID** | TC 11 |
| **Name** | Move forward |
| **Requirement(s)** | 0600 |
| **Description** | When it is a player's turn, he or she is able to and wants to move a pawn forward. |
| **Precondition(s)** | ● The player draws a card that allows him or her to move forward<br>● The player has pawns that are able to move forward<br>● If the card drawn has multiple move options, then the "Move forward x spaces" option is selected |
| **Action(s)** | 1. The player clicks a pawn that is able to move forward |
| **Postcondition(s)** | ● The pawn moves forward the number of spaces specified by the drawn card |

### 6.5.3. Move backwards

| | |
|---|---|
| **ID** | TC 12 |
| **Name** | Move backwards |
| **Requirement(s)** | 0600 |
| **Description** | When it is a player's turn, he or she is able to and wants to move a pawn backwards. |
| **Precondition(s)** | ● The player draws a card that allows him or her to move backwards<br>● The player has pawns that are able to move backwards<br>● If the card drawn has multiple move options, then the "Move backwards x spaces" option is selected |
| **Action(s)** | 1. The player clicks a pawn that is able to move backwards |
| **Postcondition(s)** | ● The players moves backwards the number of spaces specified by the drawn card |

### 6.5.4. Skip turn

| | |
|---|---|
| **ID** | TC 13 |
| **Name** | Skip turn |
| **Requirement(s)** | 0500, 0510 |
| **Description** | A player skips his or her turn. |
| **Precondition(s)** | ● The player has no available moves during his or her turn. |
| **Action(s)** | 1. The player draws a card but is unable to make any of the moves specified by the card |
| **Postcondition(s)** | ● A message box appears telling the player that he or she has no available moves.<br>● The player clicks the “Skip Turn” button on the message box.<br>● The player’s turn is ended. |

### 6.5.5. Reset moves

| | |
|---|---|
| **ID** | TC 14 |
| **Name** | Reset moves |
| **Requirement(s)** | 0500, 0510 |
| **Description** | A player who has drawn a 7 card wishes to redo his or her moves since drawing the 7 card. |
| **Precondition(s)** | ● It is the player’s turn.<br>● The player has drawn a 7 card. |
| **Action(s)** | 1. The player clicks one of the pawns to move<br>2. The player clicks one square to move the piece to.<br>3. The player clicks “Reset” to undo the move. |
| **Postcondition(s)** | ● The player’s moves for the current turn are undone.<br>● The number of moves left for the player are reset to 7 |

## 6.6. Bumping and Sliding

### 6.6.1. Bumping opponent’s piece

| | |
|---|---|
| **ID** | TC 15 |

| | |
|---|---|
| **Name** | Bumping opponent's piece |
| **Requirement(s)** | 0500, 0510, 0600 |
| **Description** | When a player ends their turn on the same square as an opponent the opponent's pawn is sent back to start. |
| **Precondition(s)** | ● Player has at least one pawn out of start<br>● Player draws a card that allows him or her to move a pawn |
| **Action(s)** | 1. Player moves pawn as specified by the drawn card.<br>2. Player finishes their turn on a space occupied by an opponent's pawn. |
| **Postcondition(s)** | ● Opponent's pawn is sent back to start. |

### 6.6.2. Bumping player's own piece

| | |
|---|---|
| **ID** | TC 16 |
| **Name** | Bumping player's own piece |
| **Requirement(s)** | 0500, 0510, 0600 |
| **Description** | When a player draws a card that would cause him or her to bump his or her own piece, the player loses that turn |
| **Precondition(s)** | ● Player has at least two pawns out of start |
| **Action(s)** | 1. Player draws a card that would land two of his or her pawns on the same space. |
| **Postcondition(s)** | ● Player is notified he or she cannot make a move.<br>● Player loses that turn. |

### 6.6.3. Sliding

| | |
|---|---|
| **ID** | TC 17 |
| **Name** | Sliding |
| **Requirement(s)** | 0500, 0510, 0600 |
| **Description** | When a player lands on the start of an opponent's slide, the player's pawn moves to the end of the slide, bumping any pawns in the way. |

| | |
|---|---|
| **Precondition(s)** | ● Player has at least one pawn out of start.<br>● Player draws a card that allows him or her to move a pawn |
| **Action(s)** | 1. Player moves card as specified by the drawn card.<br>2. Player ends turn on the start of an opponent's slide. |
| **Postcondition(s)** | ● Player's pawn is moved to the end of the slide.<br>● Any other pawns on the slide are sent back to start. |

### 6.6.4. Landing on player's own slide

| | |
|---|---|
| **ID** | TC 18 |
| **Name** | Landing on player's own slide |
| **Requirement(s)** | 0500, 0510, 0600 |
| **Description** | When a player lands on the start of an opponent's slide, the player's pawn moves to the end of the slide, bumping any pawns in the way. |
| **Precondition(s)** | ● Player has at least one pawn out of start.<br>● Player draws a card that allows him or her to move a pawn |
| **Action(s)** | 3. Player moves card as specified by the drawn card.<br>4. Player ends turn on the start of their own slide. |
| **Postcondition(s)** | ● Player's pawn stays where it originally ended the turn.<br>● Player's slide is treated like a regular sequence of squares on the board. |

# 7. Traceability

## 7.1. Traceability Matrix

| **Requirement Identifiers** | **Reqs. Tested** | 0100 | 0200 | 0300 | 0400 | 0500 | 0510 | 0600 | 0700 |
|---|---|---|---|---|---|---|---|---|---|
| **Test Cases** | 33 | 3 | 3 | 2 | 2 | 7 | 6 | 6 | 4 |
| TC 1 | 2 | x | x | | | | | | |

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