Project Flip Jump

Group 17

Mikael Ahlberg
Daniel Ericsson
Axel Stenkula
Johannes Svensson
Fredrik Vretblad

6. Functional test cases

6.1. General

6.1.1. Tutorial text

Function: Display a tutorial text.
Description: Displays a tutorial text for the user to learn how to play the game and to make the user develop his or her skills.
Pre-condition: The user has chosen a difficulty level.
Input: None.
Testing procedure: Check that the tutorial text is shown.
Expected output: The tutorial text is shown.
Output destination: The game screen.
Reference: *Requirements Document 6.1.1.1 Tutorial text*

6.1.2. Durance of play

Function: Play time.
Description: Allow the user to play until the character falls to the bottom of the screen.
Pre-condition: The user has started a game session.
Input: None.
Expected output: Game session ends.
Output destination: -
Testing procedure: Check that the game session doesn't end until the character falls to the bottom of the screen.
Reference: *Requirements Document 6.1.1.2 Durance of play*

6.1.3. Screen size

Function: Adjusting the screen size.
Description: Changes the size of the screen to oblige the user preferences.
Pre-condition: The user has chosen the Options menu option.
Input: Chosen screen size.
Expected output: Change in screen size.
Output destination: The game screen.
Testing procedure: Check that the screen size changes when the user toggles between Fullscreen on and off.
Reference: *Requirements Document 6.1.1.3 Screen size*

6.1.4. Appearance of sound

Function: Option of sound.
Description: Changes the appearance of sound in the game, muted or full sound.
Pre-condition: The user has chosen the Options menu option.
Input: Chosen sound option.
Expected output: Sound is played or not played.
Output destination: Computer speakers.
Testing procedure: Check that the sound appears or disappears when the Sound effect option is toggled.

Reference: *Requirements Document 6.1.1.4 Appearance of sound*

6.2. Game

6.2.1. Sideways movement

Function: Move sideways.
Description: Moves the character sideways as the user commands.
Pre-condition: The user has started a game session.
Input: User input.
Expected output: The character moves sideways.
Output destination: The game screen.
Testing procedure: Check that the character moves in any sideways direction when the user clicks the right or left buttons.
Reference: *Requirements Document 6.1.2.1 Sideways movement*

6.2.2. Character jump

Function: Character jumping.
Description: The character jumps as the user commands.
Pre-condition: The user has started a game session.
Input: User input.
Expected output: The character jumps.
Output destination: The game screen.
Testing procedure: Check that the character jumps when the user presses the space bar.
Reference: *Requirements Document 6.1.2.2 Character jump*

6.2.3. Collect special item

Function: Pick up item.
Description: The character picks up a special item.
Pre-conditions: The user has started a game session. The character have jumped up until a special item appears on the screen. The character has at lease one free item slot.
Input: User input.
Expected output: Item disappears and reappears in the upper left corner where all the characters special items are shown.
Output destination: The game screen.
Testing procedure: Check that the item disappears and reappears in the upper left corner of the screen when the character touches the item.
Reference: *Requirements Document 6.1.2.3 Collect special item*

6.2.4. Use special item

Function: Use item.
Description: The character uses a special item and gets a special jump.
Pre-condition: The user has started a game session. The character has jumped up in the are and has at least on special item.
Input: User input.
Expected output: The character makes a special jump and one item is removed from the inventory.
Output destination: The game screen.
Testing procedure: Check that the character does a special jump when he presses the space bar

while still in the air.
Reference: *Requirements Document 6.1.2.4 Use special item*

6.2.5. Screen movement

Function: Move the screen.
Description: Moves the screen in a certain speed upwards, forcing the character to move upwards.
Pre-condition: The user has started a game session. The character jumps.
Input: None.
Expected output: The screen is moving.
Output destination: The game screen.
Testing procedure: Check that the screen starts to move upwards after the first character jump.
Reference: *Requirements Document 6.1.2.5 Screen movement*

6.2.6. Ability to jump through blocks

Function: Jump through blocks.
Description: The character jumps through blocks on it's way up.
Pre-condition: Choose Start game
Input: None.
Expected output: None.
Output destination: The game screen.
Testing procedure: Check that the character passes trough a block when jumping if the player isn't approaching the block from above.
Reference: *Requirements Document 6.1.2.6 Ability to jump through blocks*

6.2.7. Difficulty level

Function: Choose difficulty level.
Description: The user selects difficulty level for the game.
Pre-condition: Choose Start game
Input: Chosen difficulty level.
Expected output: The corresponding screen speed to the selected difficulty level.
Output destination: The game screen.
Testing procedure: Check that for each higher difficulty level the screen speed increases.
Reference: *Requirements Document 6.1.2.7 Difficulty level*

6.2.8. Screen speed

Function: Speed increment.
Description: The screen speed increases to higher rate.
Pre-condition: The game has started. Jump upwards until a screen speed indication appears.
Input: Current speed level.
Expected output: New speed level.
Output destination: The game screen.
Testing procedure: Check that the speed increases after the indication of a speed increase.
Reference: *Requirements Document 6.1.2.8 Screen speed*

6.2.9. Screen speed increase indication

Function: Indication of speed change.
Description: Indicates the user that the screen speed has changed to a
higher rate.
Pre-condition: The game has started. Jump upwards.
Input: Speed increment.
Expected output: Visual and audible indication.
Output destination: The game screen and computer speakers.
Testing procedure: Check that a screen speed indication appears (visual and audible) after a specified amount of time.
Reference: *Requirements Document 6.1.2.9 Screen speed increase indication*

6.2.10. Receiving points

Function: Receive points.
Description: The user receives points for every block passed.
Pre-condition:The game has started. Jump upwards.
Input: User input.
Expected output: The application.
Output destination: The game screen.
Testing procedure: Check that when the character passes a block the user receives corresponding points for the block.
Reference: *Requirements Document 6.1.2.10 Receiving points*

6.2.11. Showing score

Function: Show current score.
Description: Displays the user's current score in-game.
Pre-condition: The game has started.
Input: None.
Expected output: A presented score.
Output destination: The game screen.
Testing procedure: Check that there are a in-game score counter that corresponds to the current score of the user.
Reference: *Requirements Document 6.1.2.11 Showing score*

6.2.12. Flip function

Function: Flip the screen.
Description: The screen flips (rotates) in a certain direction.
Pre-condition: A new game has been started and the user is currently playing.
Input: None.
Expected output: Rotated screen.
Output destination: The computer screen.
Testing procedure: Play the game until the screen flips around. Check if the screen rotated 90 degrees in any direction.
Reference: *Requirements Document 6.1.2.12 Flip function*

6.2.13. Flip indication

Function: Indication of rotate direction.
Description: Indicates the flip direction that the screen will rotate too.
Pre-condition: A flip rotation will occur.
Input: Flip rotation.
Expected output: Flip indication.
Output destination: The computer screen.
Testing procedure: Play the game until the screen flips around. Notice if there was a flip indication before the rotation.
Reference: *Requirements Document 6.1.2.13 Flip indication*

6.2.14. Mirroring sideways movement

Function: Mirror movement.
Description: Moves the character from one side to the other when crossing the side of the screen.
Pre-condition: A new game has started and the user is currently playing.
Input: User input.
Expected output: A mirror movement.
Output destination: The computer screen.
Testing procedure: Walk or jump towards the side of the screen. Notice if the player appears on the other side.
Reference: *Requirements Document 6.1.2.14 Mirroring sideways movement*

6.3. High score

6.3.1. Storing of high score

Function: Store the high score list.
Description: Stores the high score list to the local computers hard drive.
Pre-condition: High score has been beaten.
Input: User input.
Expected output: Store high score list.
Output destination: The computer hard drive.
Testing procedure: Enter your name and confirm it. Restart the program. Check if the name is still in the high score list.
Reference: *Requirements Document 6.1.3.1 Storing of high score*

6.3.2 Enter high score list

Function: Enter the high score.
Description: Checks the high score list and lets the user enter a name if a score was beaten.
Pre-condition: A game has ended.
Input: A high score list.
Expected output: User name.
Output destination: The computer screen.
Testing procedure: Enter your name. Confirm it with the enter button. Check to see if the name was stored in the high score list.
Reference: *Requirements Document 6.1.3.2 Enter high score list*

6.3.3. Check high score

Function: Check the high score list.
Description: Presents the current locally stored high score list to the user.
Pre-condition: The system has started and the user is in the menu.
Input: High score list.
Expected output: The reset high score list.
Output destination: The current high score list.
Testing procedure: Select the high score option. Press the enter button. Check the high score list.
Reference: *Requirements Document 6.1.3.3 Check high score*

6.3.4. Reset high score

Function: Reset high score list.
Description: Resets all the scores on the high score list.
Pre-condition: The system has started and the user is in the menu.
Input: High score list.
Expected output: The reset high score list.
Output destination: The computer screen.
Testing procedure: Select the high score option. Press the enter button. Select the reset high score button. Press the enter button.
Reference: *Requirements Document 6.1.3.4 Reset high score*

6.3.5. High score sound

Function: Play high score sound.
Description: Plays a sound effect when a previous score on the high score list was beaten.
Pre-condition: A game has ended and a score on the high score list has been beaten.
Input: High score list.
Expected output: Special sound effect.
Output destination: The computer speakers.
Testing procedure: Check the speakers for sound.
Reference: *Requirements Document 6.1.3.5 High score sound*

6.4. Non-functional requirements

6.4.1. Usability requirements

6.4.1.1. Tutorial text

Function: Learn the game in one minute.
Description: The tutorial text shall provide the user with the knowledge of how to play the game. The text shall be readable in one minute.
Pre-condition: The tutorial text is loaded on the screen.
Input: A user.
Expected output: The user has learned how to play the game.
Output destination: The user.
Testing procedure: Read the tutorial text during a minute.
Reference: *Requirements Document 6.2.1.1 Tutorial text.*

6.4.1.2. Easy to install

Function: Install the game with ease.

Description: The game installed with minimum effort. It shall only require an installation command.
Pre-condition: The installation file is reachable.
Input: Installation command.
Expected output: The game is installed with only an installation command.
Output destination: The computer.
Testing procedure: Locate installation executable. Start the installation with the OS specific installation command.
Reference: *Requirements Document 6.2.1.2 Easy to install.*

6.4.2. Performance requirement

6.4.2.1. Low start up time

Function: Start a game with limited time required.
Description: The game starts in five seconds after the user has viewed the tutorial text and pressed the start game button.
Pre-condition: The tutorial text is shown on the screen.
Input: Enter/Select command to press the start game button.
Expected output: The game starts within five seconds.
Output destination: The computer.
Testing procedure: Select the start button. Press the button. Count the amount of time for the game to start. One Mississippi, two Mississippi...
Reference: *Requirements Document 6.2.2.1 Low start up time*

6.4.3. Space requirement

6.4.3.1. Game size

Function: Install the game with limited space requirement.
Description: The system size is at most 20MB, so the textures and audio files needs to be small.
Pre-condition: The installation file is reachable and you have at least 20MB of usable space.
Input: Installation command.
Expected output: The program has been installed, using less than or equal to 20MB of space.
Output destination: The computer.
Testing procedure: Locate installation executable. Start the installation with the OS specific installation command. Check the used space for the program.
Reference: *Requirements Document 6.2.3.1 Game size*

6.4.4. Portability requirement

6.4.4.1. Multi platform playability

Function: Play on different platforms.
Description: The system works on a set of different platforms supporting the system requirements, at least OSX, Windows and Linux.
Pre-condition: The program is located on the users computer.
Input: Start command.
Expected output: The program has started.
Output destination: The computer.
Testing procedure: Locate the game. Find the executable. Start the executable with the OS specific

start command.
Reference: *Requirements Document 6.2.4.1 Multi platform playability.*