Blockr: Use Cases

Group 05
Jesse Geens, Oberon Swings,
Aram Khachaturyan, Bert De Vleeschouwer

21 February 2020

```
# Use Cases

## Use Case 1: Move Block from Palette to Program Area (Add Block)

### Primary Actor
    User

### Stakeholders and interests
    The User wishes to expand his program by adding a new block to the Program Area

### Preconditions
    The block limit has not been reached

### Success guarantee
    No blocks are added if the block is not dragged in to the program area
    Otherwise the program has been expanded by one block and the block has re-appeared in the Palette

### Main Success Scenario
    1. The user moves the mouse cursor over a block in the Palette.
    2. Presses the left mouse key.
    3. The palette is reset so that all blocks are available there
    4. Moves the mouse cursor to the Program Area.
    5. Releases the left mouse key.
    6. The system adds the new block to the Program Area.

## Use Case 2: Add Last Block (Maximum Reached)

### Primary Actor
    User

### Stakeholders and interests
    The User wishes to expand his program by adding a new block to the Program Area

### Preconditions
    The block limit has not been reached

### Success guarantee
    No blocks are added if the block is not dragged in to the program area
    Otherwise the program has been expanded by one block and the block has re-appeared in the Palette

### Main Success Scenario
    1. The user moves the mouse cursor over a block in the Palette.
    2. Presses the left mouse key.
    3. Moves the mouse cursor to the Program Area.
    4. Releases the left mouse key.
    5. The system adds a new block of the same type to the Program Area.
    6. The palette blocks disappear because the maximum is reached.

## Use Case 3: Move Block to Palette (Delete Block)
```

```
### Primary Actor
    User

### Stakeholders and interests
    The User wishes to move a block to back to the palette (removing a block from the program area or just
    the palette and letting it go.

### Preconditions

### Success guarantee
    A block is clicked on and dragged to the palette.
    When the user lets go of the block it disappears and the palette will look like it's reset.

### Main Success Scenario
    1. The user moves the mouse cursor over a block in the Palette or Program Area,
    2. The user presses the left mouse key.
    3. The user moves the mouse cursor to the Palette.
    4. The user releases the left mouse key.
    5. The system removes the block from where it came from and makes sure the palette is reset.
    6. The palette blocks reappear if the where gone because the limit is not reached anymore.


## Use Case 4: Connecting Block to Another

### Primary actor
    User

### Stakeholders and interests
    User wants to be able to connect blocks within the Program Area to make a complete program.

### Preconditions
    At least one block is in the Program Area already.

### Succes guarantee
    The new block is placed within the Program Area in range of the already existing block.
    The new block connects correctly to the other block and they make a whole.

### Main Success Scenario
    1. The user lets go of the left mouse button.
        When releasing the mouse the process checks if one of following conditions are satisfied:
        1.1 The bottom socket of the block is near a top plug of another block.
        1.2 The top plug of the block is near a bottom socket of another block.
        1.3 The left plug of the block is near a right socket of another block.
        1.4 The right socket of the block is near a left plug socket of another block.
    2. When this happens the aforementioned plug(s) and socket(s) connect with each other
    3. This block gets placed in relation to the other block so that the plugs are in the sockets
    4. The connected elements are now considered as a group of blocks

  ## Use Case 5: Move Block Around (Disconnects blocks)

### Primary actor
    User

### Stakeholders and interests
    User wants to be able to move blocks to different positions.
```

```
### Preconditions
    At least one block is in the Program Area already.

### Succes guarantee
    The user clicks on a block in the Program Area and moves it around. When the user lets go of the block t
    set to the place where the user let go of the block.

### Main Success Scenario
    1. The user moves their cursor over a block.
    2. The user left clicks on the block.
    3. The block is disconnected form it's neighbours if it was connected.
    4. The user drags block in to the Program Area and holds the button down.
    5. The user lets go of the left mouse button.
    6. The block can be connected if the conditions are met otherwise it is just placed where the user let g

  ## Use Case 6: Move Block to Illegal Position

### Primary actor
    User

### Stakeholders and interests
    User doesn't want blocks to be placed on the grid or outside of the window.

### Preconditions

### Succes guarantee
    The user clicks on a block in the Program Area or Palette and moves it outside the Palette and Program A
    of the block the position of the block should be the last valid position within the window.

### Main Success Scenario
    1. The user moves their cursor over a block.
    2. The user left clicks on the block.
    3. The block is disconnected form it's neighbours if it was connected.
    4. The user drags block in to the Program Area and holds the button down.
    5. The user drags the block outside of the window.
    6. The user lets go of the left mouse button.
    7. The block is placed in it's last valid location within the Program Area and checked if it needs to be
       position was within the Palette.

#Use Case 7: Starting the Program

### Primary actor
    User

### Stakeholders and interests
    The User wishes to run the program succesfully and lead the robot to the goal cell.

### Preconditions
    There is only one group of blocks (can be a single block) in the Program Area,
    in this group only the top socket and bottom plug are not connected.
    The program is not running yet.

### Succes guarantee
    The program starts running, the first block is highlighted.

### Main Succes Scenario
    1. The user presses F5.
    2. The program checks if the blocks in the Program Area make up a whole.
    3. The first block that needs to be executed is highlighted.

#Use Case 8: Execute Next Step

### Primary actor
    User

### Stakeholders and interests
    The User wishes to run the program succesfully and lead the robot to the goal cell.

### Preconditions
    There is only one group of blocks (can be a single block) in the Program Area,
    in this group only the top socket and bottom plug are not connected.
    The program is running already.

### Succes guarantee
    The program executes the highlighted block and highlights the block after the previously executed block

### Main Succes Scenario
    1. The user presses F5.
    2. The highlighted block is not highlighted anymore
    3. The previous block is executed
    3.1 If the block is a turn block the robot is turned accordingly.
    3.2 If the block is a move block there can't be a wall in front of the robot. If there is a wall in fron
           If there is no wall in front of the robot the position of the robot is changed accordingly.
    3.3 If the block is a while or if block first the condition within this block is checked, if it is satis
           highlighted and execution goes on from there. If the condition is not satisfied the next block i
           block.
    4. The next block that needs to be executed is highlighted. If the next block doesn't exist because the
         one no new block is highlighted.

#Use Case 9: Stop Execution

### Primary actor
    User

### Stakeholders and interests
    The User wishes to reset the program or make changes to the Program Area after running the program.

### Preconditions
    There is only one group of blocks (can be a single block) in the Program Area,
    in this group only the top socket and bottom plug are not connected.
    The program is running already.

### Succes guarantee
    The program stop execution and the robot and grid are reset to the initial state.

### Main Succes Scenario
    1. The user presses ESC or moves a block within the Program Area or presses F5 after the last block is e
    2. The highlighted block is not highlighted anymore.
    3. The program stops execution.
    4. The grid is reset to it's initial state.

#Use Case 10: Undo action

### Primary actor
```

```
    User

### Stakeholders and interests
    The user wishes to undo a certain action from the grid or program area.

### Succes guarantee
    The grid state is set back to the previous one if the program was running.
    The last change made to the blocks in the program area was undone, the added block was
    removed or the moved block was set back to it's previous position or the removed block was added again.

### Main succes scenario
    1. User presses Shift + F5
    2. Check if the program is running.
    2.1 If the program was running the previous grid state is used again.
          Except if the current grid state is the initial grid state, then nothing will happen.
    2.2 If the program was not running the previous program area state is used again.
          If there is no previous program area state nothing will happen.

### Extensions


#Use Case 11: Redo action

### Primary actor
    User

### Stakeholders and interests
    The user wishes to undo a certain action from the grid or program area.

### Succes guarantee
    The grid state is set to the next on if the program was running and there is a next grid state
    (one or more undo's has been done before redo).
    The last undone change to the blocks in the program area was redone, the removed added block was readde
    the unmoved moved block was again moved or the added removed block was removed again.

### Main succes scenario
    1. User presses Shift + Alt + F5
    2. Check if the program is running
    2.1 If the program is running and there is an undone next grid state, use this next state again.
           If there is no next grid state, nothing will happen.
    2.2 If the program was not running and there is an undone next program area stat, use this next state ag
           If there is no next program area state nothing will happen.

## Extensions
```