Block field

Idea

The program area can now be zoomed in/out and by holding the mouse down and dragging an empty space (think of Google maps or Miro).

This means that blocks are not always visible and can differ in size depending on how zoomed in the user is (they should all be the same size but this size is larger when zoomed in and smaller when zoomed out).

Proposed solution/patterns

- Prototype pattern: clone de bestaande blocks met de nieuwe scale factor
- Translator (=adapter) pattern: translate new coordinates
- Observer pattern voor scaling

Scroll bars

Idea

Similar to the previous one but now the program area has a fixed size through which you can scroll.

Proposed solution/patterns

Same as block field

Decorator hoort hier technisch gezien ook bij maar aangezien de meeste mensen vanuit een MVC logica gewerkt hebben is de UI “dom” en dus in de praktijk niet van toepassing (lijkt wel belangrijk om te vermelden dat dit de reden is om decorator niet te kiezen).

Resize

Idea

You can resize all three subwindows by dragging the border. Scroll bars can be included in this idea as well.

Proposed solution/patterns

- In plaats van hardcoded width gebruik je een object met een observer/listener

Editor mode

Idea

When pressing “e”, the user can start editing the gameworld. When clicking on a coordinate in the game grid, it iterates between all possible block states in the game world that fits the game (e.g. empty->wall->goalcell->robot->empty etc.).

When the user is done they press “t” The new grid has to be valid, otherwise it reverts back to the way it was before.

Proposed solution/patterns

- API aanpassen: changeBlock(x,y)

- GameWorld.changeBlock(x,y) past de map aan zoals gewenst
- Memento pattern safed nu ook de map & kan deze resetten
- Update UI
- ???
- Profit

- State pattern for game edit mode/normal mode

Boolean/int variable blocks

Idea

You now have boolean and int variable blocks which have instantiation blocks, and edit blocks. Conditionals now contain equals, greater than/less than and boolean variable reference blocks.

You also have integer operation blocks. (uitleg is wat vaag atm maar het idee is van meer richting scratch te gaan)

Proposed solution/patterns

- Memento pattern for saving vars
- Set > x & Get > x
- X saved in blockarea
- Nul & plusEen(x) -> alle ints kunnen gevormd worden
- x > Greater than > y & x > equals > y

Scoring system

Idea

Scoring works like golf (less points is better) => the amount of required moves is your score. At the end of your game you need to input your name and there is a scoreboard which you can see by pressing “s”.

(if you really want to go nuts: score equals number of moves divided by minimal number of moves => which you determine by a shortest path algorithm like A* in advance or something)

Proposed solution/patterns

- Score in programrunner of via observer
- Op einde bij victory doorgeven aan apart object die scoreboard bijhoudt via notifier/observer
- Template method pattern voor scoreboard
- State pattern for s-mode

Free play mode

Idea

You can now use arrow buttons to control the robot instead of the program by pressing “f” and going to free play mode. You go back to default mode by pressing “q”.

Proposed solution/patterns

- State pattern

Programming with text

Idea

You can generate the blocks program using a text notation.

MAIN:MOVEFORWARD->WHILE(NOT WALLINFRONT)(TURNRIGHT->TURNLEFT)->FUNCTIONCALL1

FUNCTIONDEF1:MOVEFORWARD->TURNLEFT

Proposed solution/patterns

Is dat niet gewoon parsing

- Interpreter pattern -> ignore die shit
- Zware adapter

Import

Idea

You can import a game field representation with a text file.
You can import a program representation with a text file (see “Programming with text”).

Proposed solution/patterns

- Memento pattern
- Interpreter pattern

Multiple windows

Idea

You have a window containing the palette and programarea. When you start the game this window disappears and a new one containing the game appears. When you quit the game, you go back to the palette/programarea window.

Proposed solution/patterns

- gewoon niet tekenen

Menu bar / toolbar

Idea

Add a menu bar and toolbar with all functionalities for which you can use buttons.

Voeg een menubar en een toolbar toe. Menubar items tonen een kolom van acties wanneer er op geklikt wordt, deze acties worden dan uitgevoerd wanneer het item aangeklikt wordt (bv. switch diagram type, create new diagram, add party). Toolbar items bevatten een icon. Menubar en toolbar items kunnen enabled of disabled zijn afhangende van bepaalde voorwaarden, toolbar items tonen een tooltip wanneer de cursor 0.5 seconden op het item blijft. De toolbar bevat ook een textbox om labels te bewerken. De menubar en toolbar passen de weergave en positie van de items aan naarmate de canvas resized.

Proposed solution/patterns

- Nieuwe klikbare area die een beetje zoals de palet beslist waarop wordt geklikt en dit dan execute
- State met aparte render en event handling naargelang een menubar open staat (no menu open, menu 1 open, menu 2 open, …)
- En dan wat standaard methodes in de UI die de menu bar/toolbar renderen

Obstacle breaker

Idea

You have one “obstacle breaker” block which removes a wall in the direction the robot is facing (if there is one). This block is not allowed to be placed inside a while block.

(niet helemaal bruikbaar)

Proposed solution/patterns

- memento

Levels

Idea

The game has multiple levels. You start at “level 0” and whenever you finish the level (for the robot -> you step on the goal block), the next level is loaded, the robot is put in the starting position and the program area is cleared. Levels are described by text files with a certain format (for the size, location of walls etc.) and dynamically read in when needed. For instance, there is a folder /levels with files level01.txt, level02.txt ...

Proposed solution/patterns

- Memento met map
- Nieuw level laden?

Complex conditionals

Idea

The standard conditions are extended with “and” and “or”. This makes evaluating your conditions more complex. Possibly there also has to be an extra condition like “EnemyInFront” or something like that because with only one boolean the “WallInFront” this wouldn’t make much sense but it is the design that counts here not if it makes a lot of sense or not.

Proposed solution/patterns

Specification pattern voor evaluatie van conditionals + composite pattern for blocks hierarchy should do the trick. Wikipedia even has the perfect example of evaluating: https://en.wikipedia.org/wiki/Specification_pattern

Break statement

Idea

This is not very difficult to implement if your functions are already working great. Still it could be nice to think about. Either a button that breaks out of your current function call or a block that breaks out of the current function call.

Proposed solution/patterns

- Pop 1 element uit uw stack van function calls, of exit execution indien deze stack leeg is