# Tests

Contents

[Tests 1](#_Toc289674026)

[Scenario 1 1](#_Toc289674027)

[test1\_1MoveLeftPushedBackHorizontally01 1](#_Toc289674028)

[test1\_2MoveLeftPushedBackDownRight02 1](#_Toc289674029)

[test1\_3MoveInsertedVertically1 1](#_Toc289674030)

[test1\_4MovePushedVertically2 1](#_Toc289674031)

[Scenario 2 2](#_Toc289674032)

[Test2\_1InsertAndPushedRightHorizontally 2](#_Toc289674033)

[test2\_2PushedRightAndDownNicely 2](#_Toc289674034)

[Scenario 3 2](#_Toc289674035)

[test3\_1PushedBetweenLeftAndRight 2](#_Toc289674036)

[test3\_2PushedBetweenLeftAndRightRefused 2](#_Toc289674037)

[test3\_2aPushedRefusedButLeftMovedAnyway 3](#_Toc289674038)

[test3\_3InsertedAndTwoPushedRight 3](#_Toc289674039)

[test3\_4InsertedVerticallyNothingPushedRight 3](#_Toc289674040)

[test3\_5InsertedVerticallyTwoPushedDown 3](#_Toc289674041)

[Scenario 3a 3](#_Toc289674042)

[test3a\_1PushedLeft 3](#_Toc289674043)

[test3a\_2PushedLeftD97DoesntMove 4](#_Toc289674044)

[test3a\_3PushedLeftNotFlyingUpY 4](#_Toc289674045)

[Scenario 4 5](#_Toc289674046)

[test4\_1InsertedTwoPushedRightTwoPushedDown 5](#_Toc289674047)

[test4\_2InsertedTwoPushedRightThreePushedDown 5](#_Toc289674048)

[Scenario 5 – stress tests 6](#_Toc289674049)

[Scenario 6 – line crossing enhancements 6](#_Toc289674050)

[test6\_1LineCrossingNotNeeded 6](#_Toc289674051)

[test6\_2LineCrossingAvoided 6](#_Toc289674052)

[Scenario 7 – weird behaviour when edges involved 7](#_Toc289674053)

[test7\_1DontJumpTooFarY 7](#_Toc289674054)

[test7\_2DontJumpTooFarY 7](#_Toc289674055)

## Scenario 1

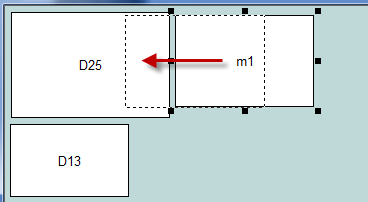
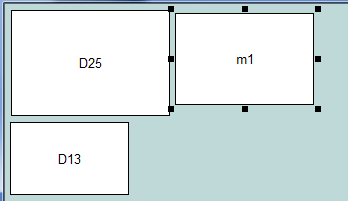
Basic tests. Note that the algorithm is tuned to move the minimum needed, so if moving Y is less distance than moving X then a vertical movement will be applied.

{'type':'node', 'id':'D25', 'x':6, 'y':7, 'width':159, 'height':106}

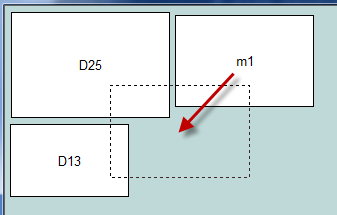
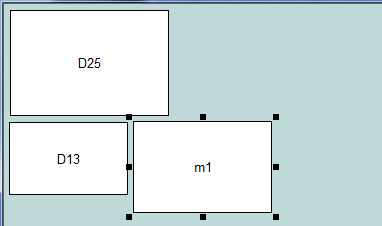
{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

{'type':'node', 'id':'m1', 'x':170, 'y':9, 'width':139, 'height':92}

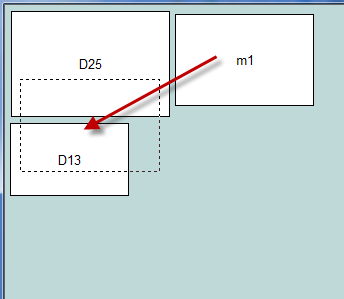
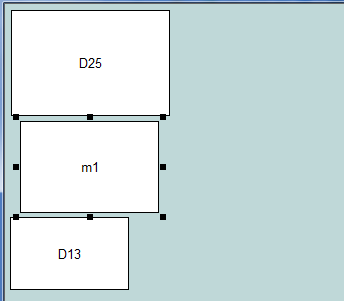
### test1\_1MoveLeftPushedBackHorizontally01

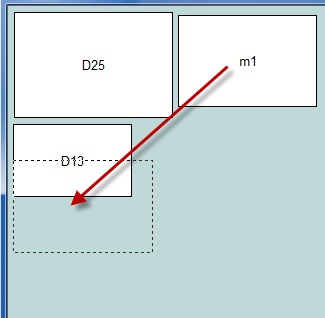
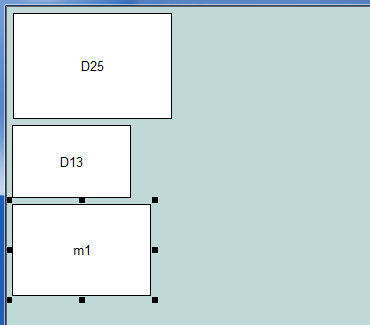
### test1\_2MoveLeftPushedBackDownRight02

### test1\_3MoveInsertedVertically1

### test1\_4MovePushedVertically2

## Scenario 2

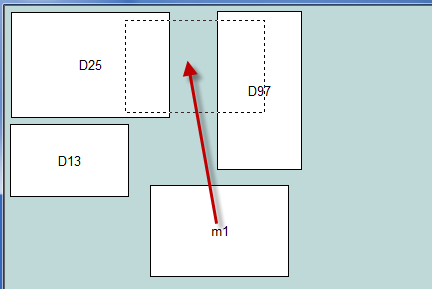
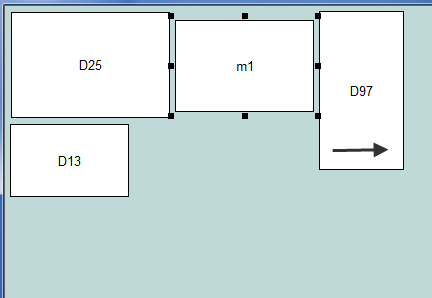
{'type':'node', 'id':'D25', 'x':7, 'y':6, 'width':159, 'height':106}

{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

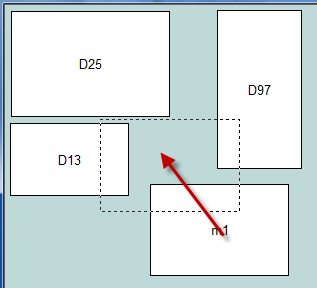
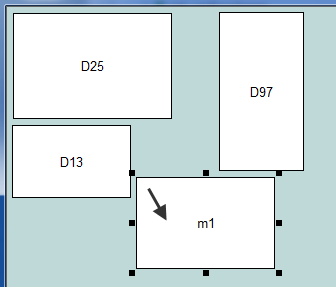
{'type':'node', 'id':'m1', 'x':146, 'y':179, 'width':139, 'height':92}

{'type':'node', 'id':'D97', 'x':213, 'y':6, 'width':85, 'height':159}

### Test2\_1InsertAndPushedRightHorizontally

### test2\_2PushedRightAndDownNicely

## Scenario 3

{'type':'node', 'id':'D25', 'x':7, 'y':6, 'width':159, 'height':106}

{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

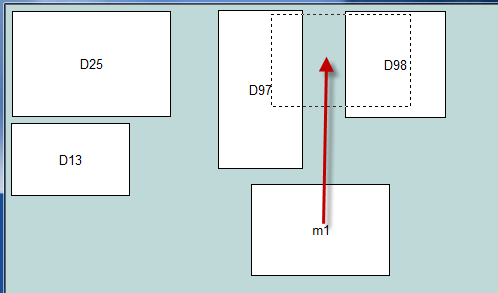
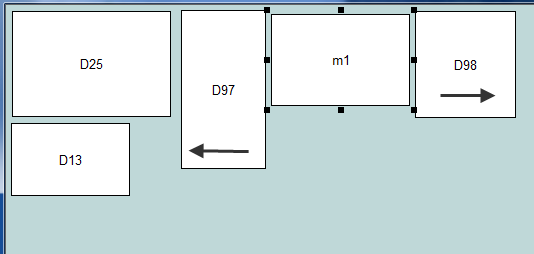
{'type':'node', 'id':'m1', 'x':246, 'y':179, 'width':139, 'height':92}

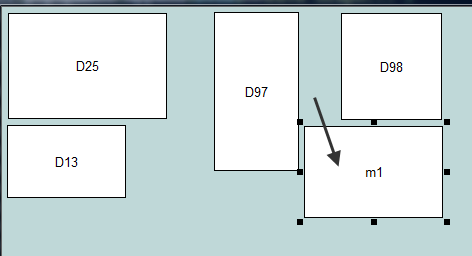
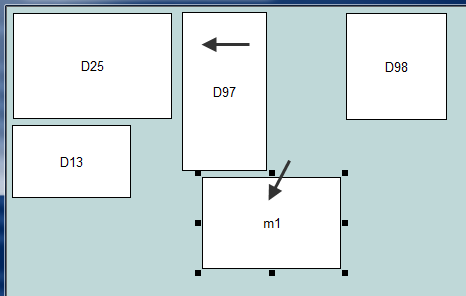
{'type':'node', 'id':'D97', 'x':213, 'y':6, 'width':85, 'height':159}

{'type':'node', 'id':'D98', 'x':340, 'y':7, 'width':101, 'height':107}

### test3\_1PushedBetweenLeftAndRight

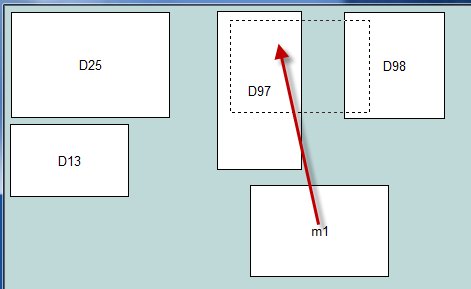
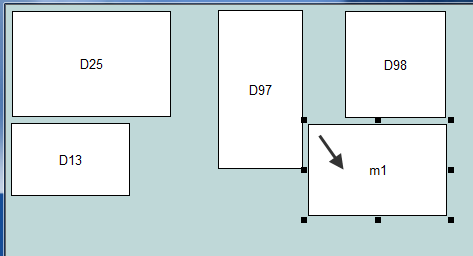
assert m1 has been pushed between two nodes, horizontally. Both left and right nodes moved left and right respectively.

Or if snug logic is on, then  or 

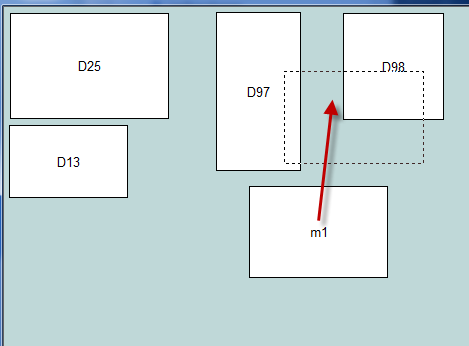
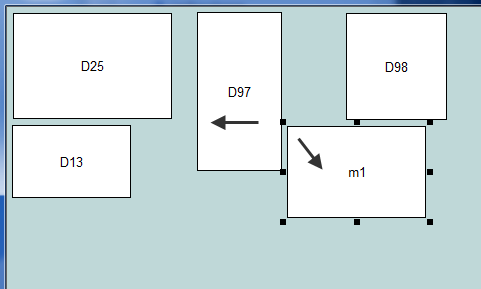
### test3\_2PushedBetweenLeftAndRightRefused

assert m1 has been not been inserted - refused and snuggled instead

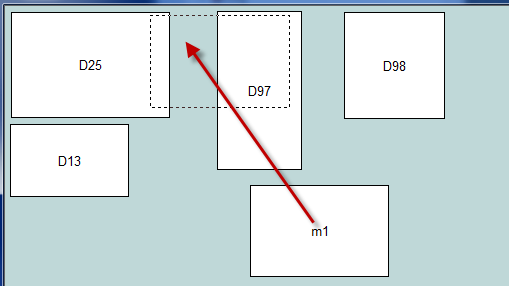
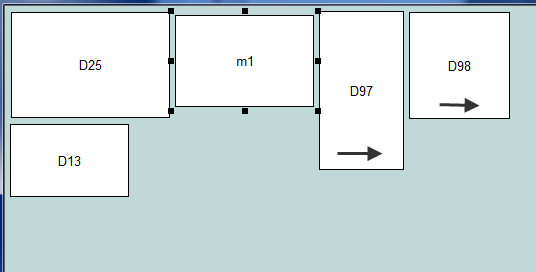
### test3\_2aPushedRefusedButLeftMovedAnyway

m1 has been refused insertion, but left (D97) moved leftwards cos there is room. m1 snuggled below and to the right.

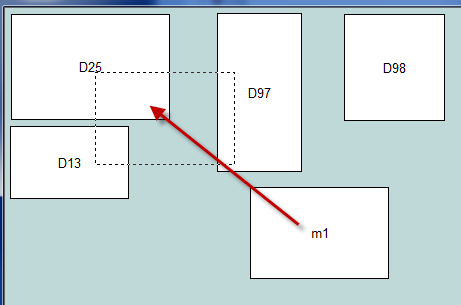
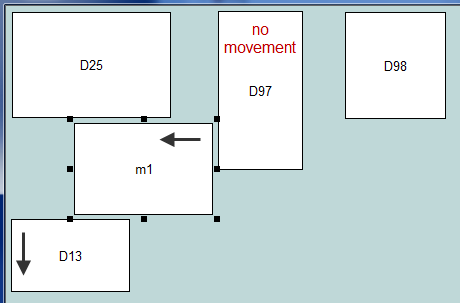
### test3\_3InsertedAndTwoPushedRight

assert m1 has been inserted - and two nodes pushed right

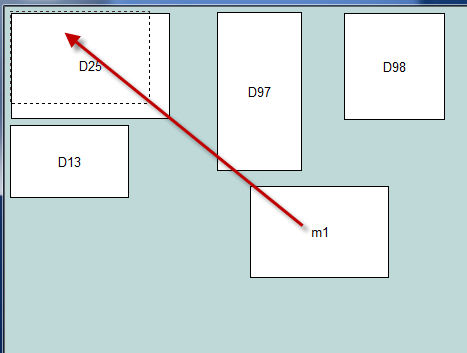
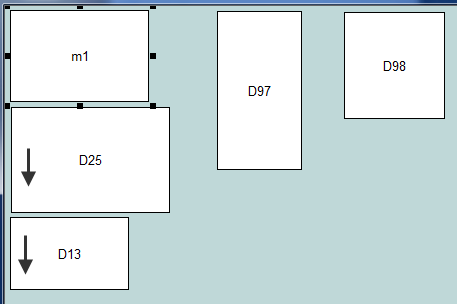
 

### test3\_4InsertedVerticallyNothingPushedRight

assert m1 has been inserted vertically - one node pushed down, NO nodes pushed right

### test3\_5InsertedVerticallyTwoPushedDown

## Scenario 3a

{'type':'node', 'id':'D25', 'x':7, 'y':6, 'width':159, 'height':106}

{'type':'node', 'id':'D13', 'x':6, 'y':116, 'width':119, 'height':73}

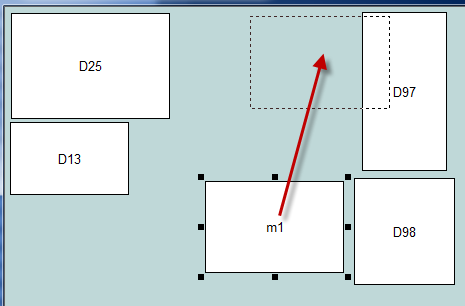
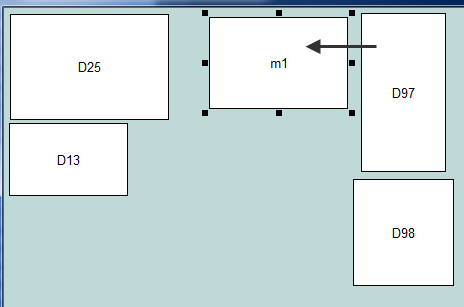
{'type':'node', 'id':'m1', 'x':201, 'y':174, 'width':139, 'height':92}

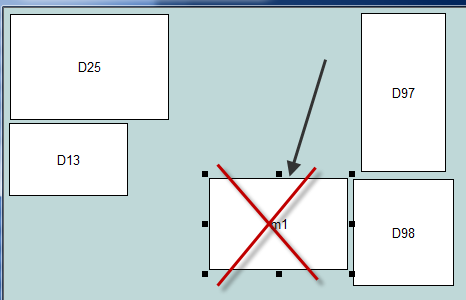
{'type':'node', 'id':'D97', 'x':358, 'y':6, 'width':85, 'height':159}

{'type':'node', 'id':'D98', 'x':350, 'y':172, 'width':101, 'height':107}

### test3A\_1PushedLeft

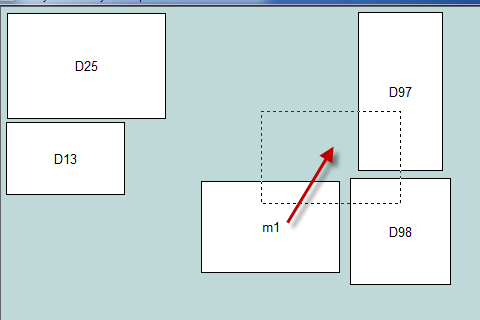
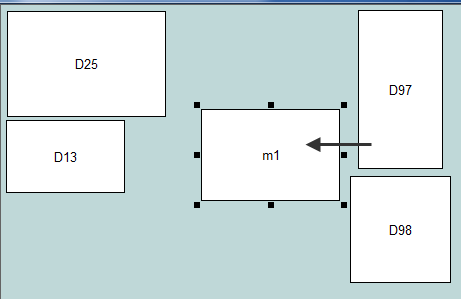
move m1 to the right, should be pushed back to the left

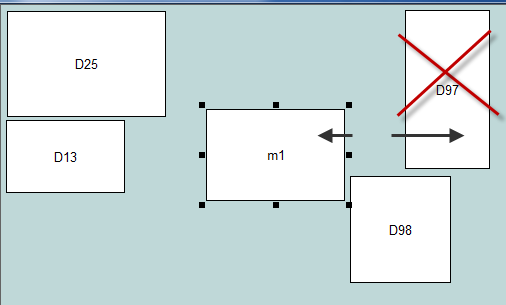
 

**not **

### test3A\_2PushedLeftD97DoesntMove

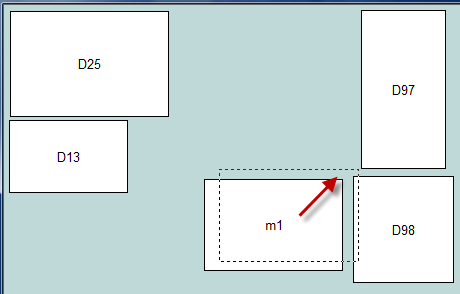
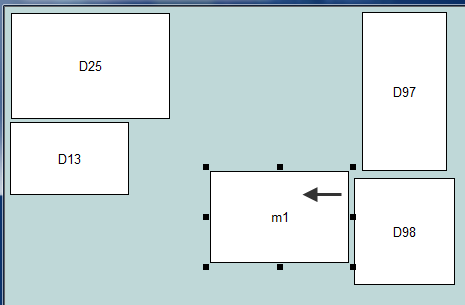
# move m1 to the right, should be pushed back to the left. D97 shouldn't move!

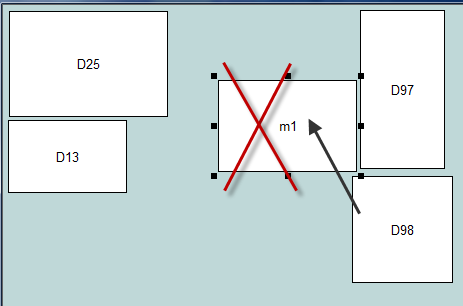
 

**not **

### test3A\_3PushedLeftNotFlyingUpY

move m1 to the right, should be pushed back to the left. Not flown up Y. **Note that the top of m1 is > top of D98 to trigger this test**.

**not **

## Scenario 4

{'type':'node', 'id':'D25', 'x':7, 'y':6, 'width':159, 'height':106}

{'type':'node', 'id':'D13', 'x':6, 'y':119, 'width':119, 'height':73}

{'type':'node', 'id':'m1', 'x':6, 'y':214, 'width':139, 'height':92}

{'type':'node', 'id':'D97', 'x':213, 'y':6, 'width':85, 'height':159}

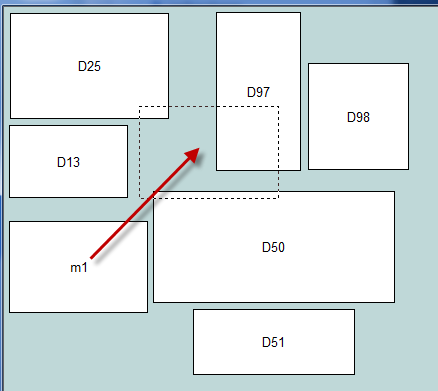
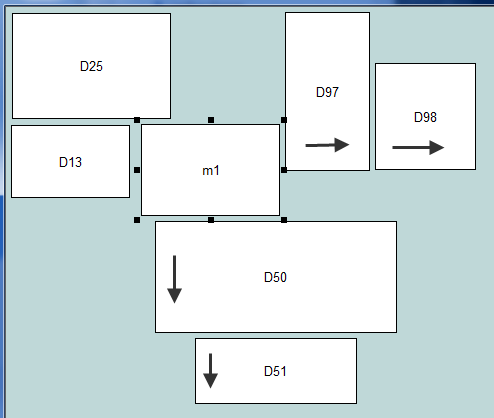
{'type':'node', 'id':'D98', 'x':305, 'y':57, 'width':101, 'height':107}

{'type':'node', 'id':'D50', 'x':149, 'y':184, 'width':242, 'height':112}

{'type':'node', 'id':'D51', 'x':189, 'y':302, 'width':162, 'height':66}

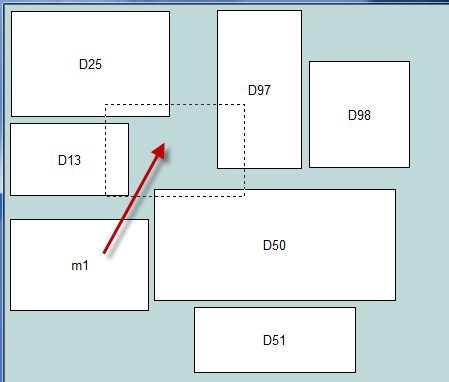
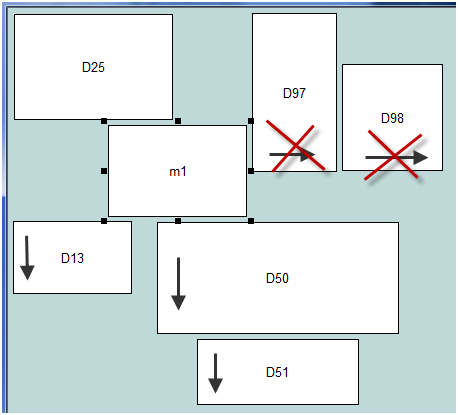
### test4\_1InsertedTwoPushedRightTwoPushedDown

assert m1 has been inserted - two pushed right, two pushed down

### test4\_2InsertedTwoNotPushedRightThreePushedDown

assert m1 has been inserted - ~~two pushed right~~, two pushed down, and extra D13 pushed down because m1 overlaps/attacks D13 (and there is room for D13 to move downwards I guess)

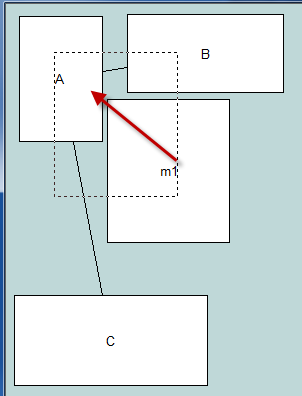
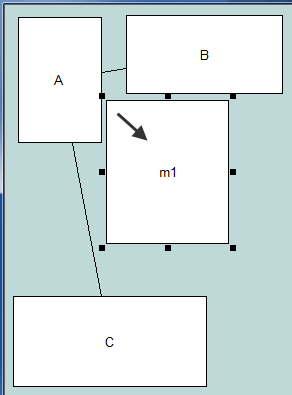
## Scenario 5 – stress tests

Run a complex jumble repeatedly till it fails. I increased MAX\_CYCLES to 20 to improve success.

## Scenario 6 – line crossing enhancements

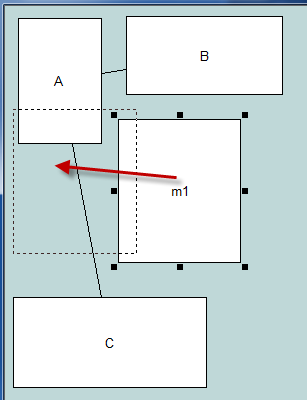
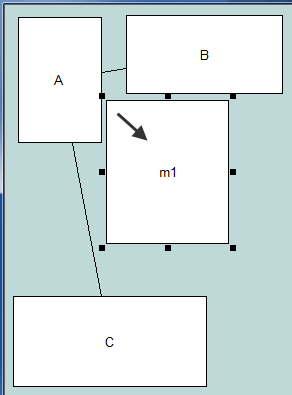
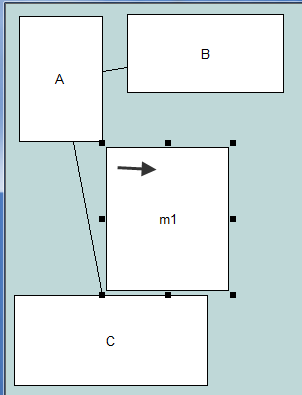
### test6\_1LineCrossingNotNeeded

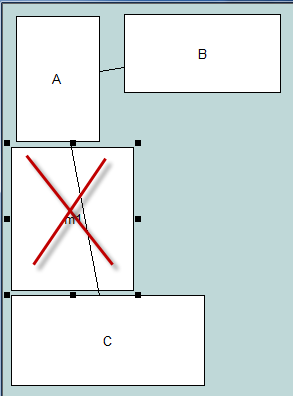
Usual repulsion and snuggling.

### test6\_2LineCrossingAvoided

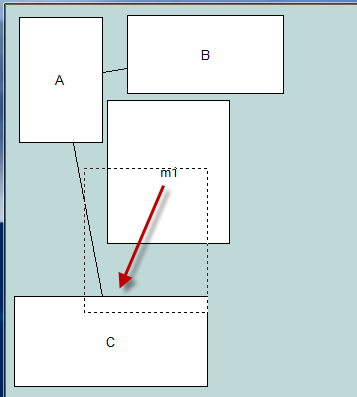
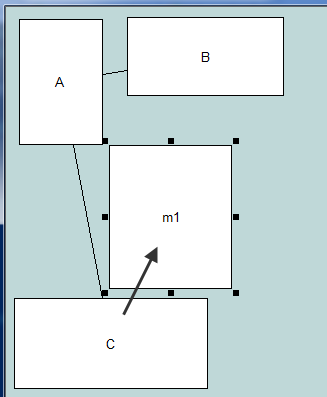
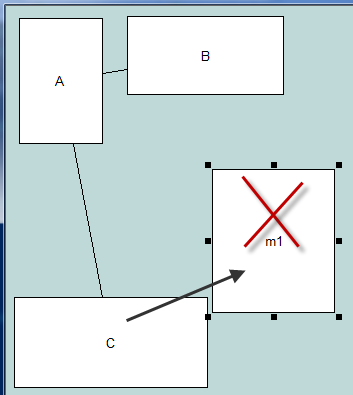
assert m1 has been repulsed and snuggeled, and line not crossed - same results as ABOVE

  **or** 

**not** 

### test6\_3LineCrossingAvoidedGoSnug

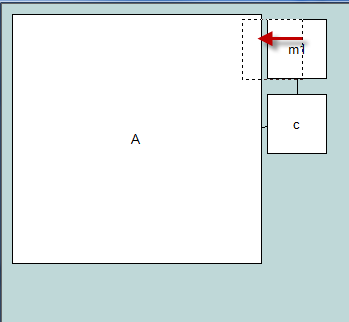
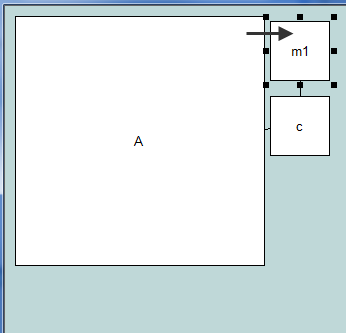
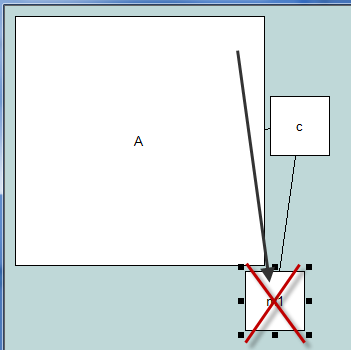
assert m1 has been repulsed up and to the right, snuggled, and line not crossed

  **not** 

## Scenario 7 – weird behaviour when edges involved

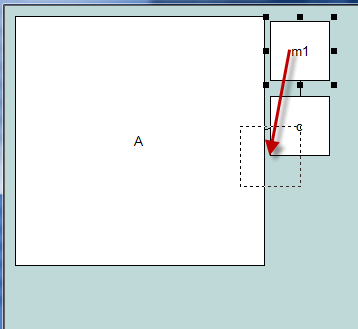
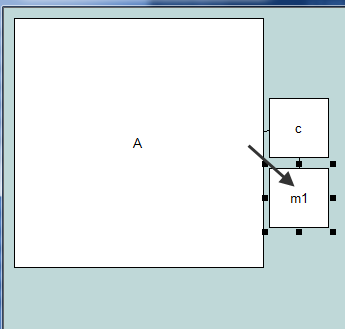
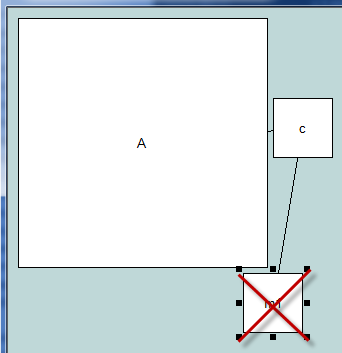
### test7\_1DontJumpTooFarY

assert m1 has been pushed to the right. don't see why edges should make any difference. initially found m1 was being pushed way too far down in the Y direction!

  **not **

### test7\_2DontJumpTooFarY

assert m1 has been pushed to the right and down – snugly – not pushed way too far down in the Y direction!

  **not **

## Scenario 8 – Back to normal node overlapping

{'type':'node', 'id':'B1', 'x':17, 'y':17, 'width':56, 'height':106}

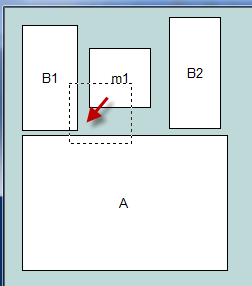
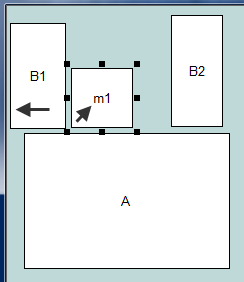
{'type':'node', 'id':'B2', 'x':164, 'y':9, 'width':52, 'height':112}

{'type':'node', 'id':'A', 'x':17, 'y':127, 'width':206, 'height':136}

{'type':'node', 'id':'m1', 'x':84, 'y':40, 'width':62, 'height':60}

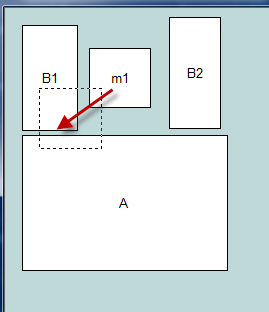
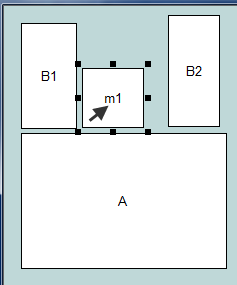
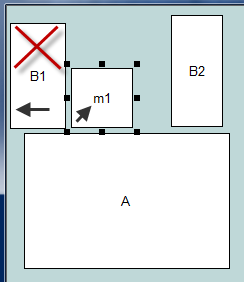
### test8\_1JumpUpAndSnuggle

assert m1 has been pushed up and to the right. Ok for B1 to be pushed a little left

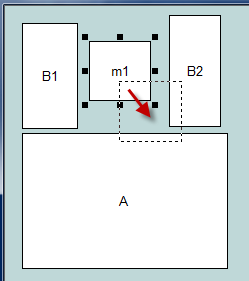
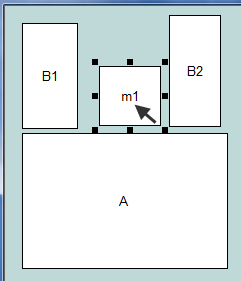
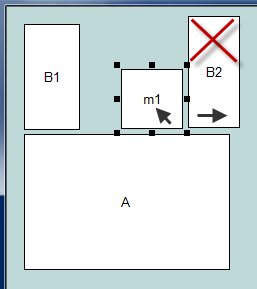
### test8\_1JumpUpAndSnuggleB1PushedOk

assert m1 has been pushed up and to the right. We used to have it so moving y up was not an option for m1 so A got pushed down instead.

  **not **

### test8\_2JumpUpAndSnuggle

assert m1 has been pushed up and to the left. We used to have it so moving y up was not an option for m1 so A got pushed down instead.

  **not **