drawBody(QGraphicsItem* group, const std::vector < std::unique_ptr < Block > > vector)

Lokale/globale Variablen und Attribute:
./_deve_data_auf/grund_der_Pferoge_der_Skraldem_direkt_in_dem_Code_aufgelistet

```
bool
QGraphicsRectItem* snapRect = new QGraphicsRectItem(group)
snapRect.setRect(left, top, width, 20)
drawBody(tip(prevTest, group, snapRect))
unsigned int index = 0 Index < vector.size() Index += Index + 1
  Block* block = vector[index].get()
  SnapBlock* snapBlock = dynamic_cast < SnapBlock* > (block)
auto
  QString text = QString.fromUtf8(snapBlock.getLoreIcon())
  QGraphicsSimpleTextItem* commentBlock = new QGraphicsSimpleTextItem(group)
  commentBlock.setText(text)
  snapRect.setWidth(commentBlock.width()+paddingLeft)
  commentBlock.setPos(left+paddingLeft,top+paddingTopBlock)
  int blockHeight = commentBlock.boundingRect().height()+paddingTopBlock*2
  int blockWidth = commentBlock.boundingRect().width()+paddingLeft*2
auto                                                                                    false
  blockWidth > minWidth
  minWidth += blockWidth
  QGraphicsRectItem* commentRect = new QGraphicsRectItem(group)
  commentRect.setRect(left, top, width, blockHeight)
  top += top + blockHeight
```

```
∅
```

```
vector.size() < 0                                                                       false
∅
unsigned int index = 0 Index < vector.size() Index += Index + 1
  Block* block = vector[index].get()
  IfBlockBlock* ifBlockBlock = dynamic_cast < IfBlockBlock* > (block)
auto                                                                                    false
  IfBlockBlock* innerIfBlockBlock = IfBlockBlock
  int nestingCounter = 1
  innerIfBlockBlock
    nestingCounter = nestingCounter + 1
    auto                                                                                false
    Block* inner = innerIfBlockBlock.getNext().getBlock().fixIt().get()
    innerIfBlockBlock = dynamic_cast < IfBlockBlock* > (inner)
  float relation = 1.2 / (nestingCounter)
  int leftWidth = width * relation
  int rightWidth = width - leftWidth
  QString text = QString.fromUtf8(ifBlockBlock.getCondition())
  QGraphicsSimpleTextItem* conditionText = new QGraphicsSimpleTextItem(group)
  conditionText.setText(text)
  wrapText(conditionText, std::round(width*0.73))
  conditionRect.setPos(left + paddingLeft + std::round(leftWidth - conditionText.boundingRect().width() * relation), top + paddingTopBlock)
  int textHeight = conditionText.boundingRect().height() + 2*paddingTopBlock
  int textWidth = conditionText.boundingRect().width() + 2*paddingLeft
  int ifBlockBlockHeight = (width*1*sin(height)/(width+sin(width) + k(1*sin(b)+sin(b)))/2)
auto                                                                                    false
  ifBlockBlockHeight < minRelativeIfBlockBlock*width) or (ifBlockBlockHeight < 0)
  ifBlockBlockHeight = minRelativeIfBlockBlock*width
  QGraphicsRectItem* conditionRect = new QGraphicsRectItem(group)
  conditionRect.setRect(left, top, width, ifBlockBlockHeight)
  QGraphicsLineItem* leftLine = new QGraphicsLineItem(group)
  QGraphicsLineItem* rightLine = new QGraphicsLineItem(group)
  leftLine.setLine(left, top, leftWidth + left, top + ifBlockBlockHeight)
  rightLine.setLine(left + width, top, leftWidth + left, top + ifBlockBlockHeight)
  QGraphicsSimpleTextItem* trueText = new QGraphicsSimpleTextItem(group)
  QGraphicsSimpleTextItem* falseText = new QGraphicsSimpleTextItem(group)
  trueText.setText(ifBlockBlock.getTrueLabel())
  falseText.setText(ifBlockBlock.getFalseLabel())
  trueText.setPos(left+0.5, top+ifBlockBlockHeight+trueText.boundingRect().height())
  falseText.setPos(left+width-falseText.boundingRect().width()-0.5, top+ifBlockBlockHeight-falseText.boundingRect().height())
  top = top + ifBlockBlockHeight
  int saveTop = top, saveLeft = left, saveWidth = width, leftTop, rightTop
  width = leftWidth
  drawBody(group, ifBlockBlock.getNext().getBlock())
  leftTop = top
  top = saveTop
  left = left + width
  width = rightWidth
auto                                                                                    false
  ifBlockBlock.getNext().getBlock().size() > 0
  drawBody(group, ifBlockBlock.getNext().getBlock())
  rightTop = top
  left = saveLeft
  width = saveWidth
auto                                                                                    false
  leftTop > rightTop
  top = std::max(leftTop, rightTop)
  QGraphicsRectItem* spacerRect = new QGraphicsRectItem(group)
auto                                                                                    false
  leftTop < rightTop
  spacerRect.setRect(left, leftTop, leftWidth, top - leftTop)
  spacerRect.setRect(left + leftWidth, rightTop, rightWidth, top - rightTop)
  drawBody(tip(spacerRect, group, nextBodyTip(block)))
```

```
∅
```

```
IfBlockBlock
  loopBlock* loopBlock = dynamic_cast < LoopBlock* > (block)
auto                                                                                    false
  QGraphicsSimpleTextItem* loopHeading = new QGraphicsSimpleTextItem(group)
  loopHeading.setText(QString.fromUtf8(loopBlock.getCondition()))
  int startTop = top
auto                                                                                    false
  loopBlock.getIterationCondition()
  drawLoopHeading(loopHeading)
  left = left + loopOffset
  width = width - loopOffset
  left = left + loopOffset
auto                                                                                    false
  not loopBlock.getWhileControlled()
  drawLoopHeading(loopHeading)
  QGraphicsRectItem* border = new QGraphicsRectItem(group)
  border.setRect(left, startTop, width, top-startTop)
```

```
∅
```

```
loopBlock
  SubIfBlock* subIfBlock = dynamic_cast < SubIfBlock* > (block)
auto                                                                                    false
  int subIfPortSizeMinimum = std::round(width
  * subIfBlock.getSequence().size())
auto                                                                                    false
  true! relativeNormalCaseToBlock == 1
  int addSubIfPlus = 0
  int reduceWidth = width
  int/float relativeNormalCase = false
  int/float* relativeNormalCasePtr = nullptr
  int/float* const relativeNormalCaseConst = relativeNormalCasePtr
  int/float* const* const relativeNormalCaseConstPtr = &relativeNormalCasePtr
  int/float* const* const* const relativeNormalCaseConstPtrPtr = &relativeNormalCaseConstPtr
  int/float* const* const* const* const relativeNormalCaseConstPtrPtrPtr = &relativeNormalCaseConstPtrPtr
  int/float* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtr
  int/float* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr = &relativeNormalCaseConstPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtrPtr
  int/float* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const* const relativeNormalCaseConstPtr
```