

Math

The image shows a screenshot of a programming environment's menu system. At the top, there is a toolbar with buttons for 'this.pig', 'move', 'FORWARD', '-1.0', and 'add detail'. Below the toolbar, a menu is open, showing the following options:

- amount:
 - 1.0 (current value)
 - 0.0
 - 0.25
 - 0.5
 - 1.0
 - 2.0
 - 10.0
- Random ▶
- Whole to Decimal Number ▶
- Math ▶**
- Custom DecimalNumber...

The 'Math' menu is expanded, showing the following options:

- 1.0 + ??? ▶
- 1.0 - ??? ▶
- 1.0 * ??? ▶
- 1.0 / ??? ▶
- ??? + ??? ▶
- ??? - ??? ▶
- ??? * ??? ▶
- ??? / ??? ▶
- min, max ▶
- absolute value, round, ceiling, floor ▶
- sqrt, pow ▶
- sin, cos, tan, asin, acos, atan, atan2, PI ▶
- exp, log, E ▶

Given below are the condition possibilities for an if statement

The image shows the 'if' block menu in Scratch, which is used to create conditional logic. The menu is organized into several sections:

- Boolean Logic:**
 - `if true is true then` (with a dropdown menu)
 - `if true (current value)`
 - `if true`
 - `if false`
 - `if nextRandomBoolean`
 - `if NOT true`
 - `if NOT ???`
 - `if BOTH true AND ???`
 - `if EITHER true OR ???`
 - `if BOTH ??? AND ???`
 - `if EITHER ??? OR ???`
- Relational Operators:**
 - Relational (DecimalNumber) { ==, !=, <, <=, >=, > }
 - Relational (WholeNumber) { ==, !=, <, <=, >=, > }
 - Relational (SThing) { ==, != }
 - Relational (MoveDirection) { ==, != }
 - Relational (TurnDirection) { ==, != }
 - Relational (RollDirection) { ==, != }
 - Relational (Key) { ==, != }
 - Relational (Color) { ==, != }
 - Relational (Paint) { ==, != }
 - TextString Comparison

Arrows point from the menu items to their corresponding Scratch code blocks:

- The `if true` section points to a list of comparison blocks: `???` < `???`, `???` <= `???`, `???` > `???`, `???` >= `???`, `???` == `???`, and `???` != `???`.
- The `if NOT true` and `if NOT ???` items point to `???` == `???` and `???` != `???` blocks.
- The `TextString Comparison` item points to a list of text comparison blocks: `???` contentEquals `???`, `???` equalsIgnoreCase `???`, `???` startsWith `???`, `???` endsWith `???`, and `???` contains `???`.

Below are the tiles at the bottom of a **procedure**

The bottom of a Scratch procedure block contains the following control flow tiles:

- `do in order`
- `count _`
- `while _`
- `for each in _`
- `if _`
- `do together`
- `each in _ together`
- `variable...`
- `assign`
- `//comment`

Below are the tiles at the bottom of a **function**

The bottom of a Scratch function block contains the following control flow tiles:

- `do in order`
- `count _`
- `while _`
- `for each in _`
- `if _`
- `do together`
- `each in _ together`
- `variable...`
- `assign`
- `//comment`
- `return _`

Given below are the panda procedures and panda Properties on the bottom right.

The screenshot shows a dropdown menu for 'this.panda' with a panda icon. Below it are two tabs: 'Procedures' and 'Functions'. A 'group by category' dropdown is visible. The 'Panda' category contains three 'Editable Procedures': 'standingPose', 'sleepingPose', and 'crawlingPose', each with an 'edit' button. The 'Biped' category is currently empty. Below these are sections for 'say, think', 'position', 'orientation', and 'position & orientation', each containing several procedural blocks with input fields for parameters like 'text', 'direction', 'amount', 'target', and 'spatialRelation'.

The screenshot shows a dropdown menu for 'this.panda' with a panda icon. Below it are several categories of properties, each with a corresponding block: 'size' (setWidth, setHeight, setDepth, resize, resizeWidth, resizeHeight, resizeDepth), 'appearance' (setPaint, setOpacity), 'vehicle' (setVehicle), 'audio' (playAudio), 'timing' (delay), and 'other' (straightenOutJoints). Each block contains a 'this.panda' icon and a parameter input field.

The screenshot shows the 'this.panda's Properties' panel. It features a 'one shots' dropdown, a 'Panda' label with a 'new (Panda)' button, and several property settings: 'Paint' set to 'WHITE', 'Opacity' set to '1.0', 'Vehicle' set to 'this', and 'Position' with x: -1.00, y: 1.73, and z: -0.10. Below these are 'Width' (0.75), 'Size' (Height: 1.15), and 'Depth' (0.53) settings, with a 'Reset' button. At the bottom, there is a 'Show Joints' checkbox.

Given below are the panda functions.

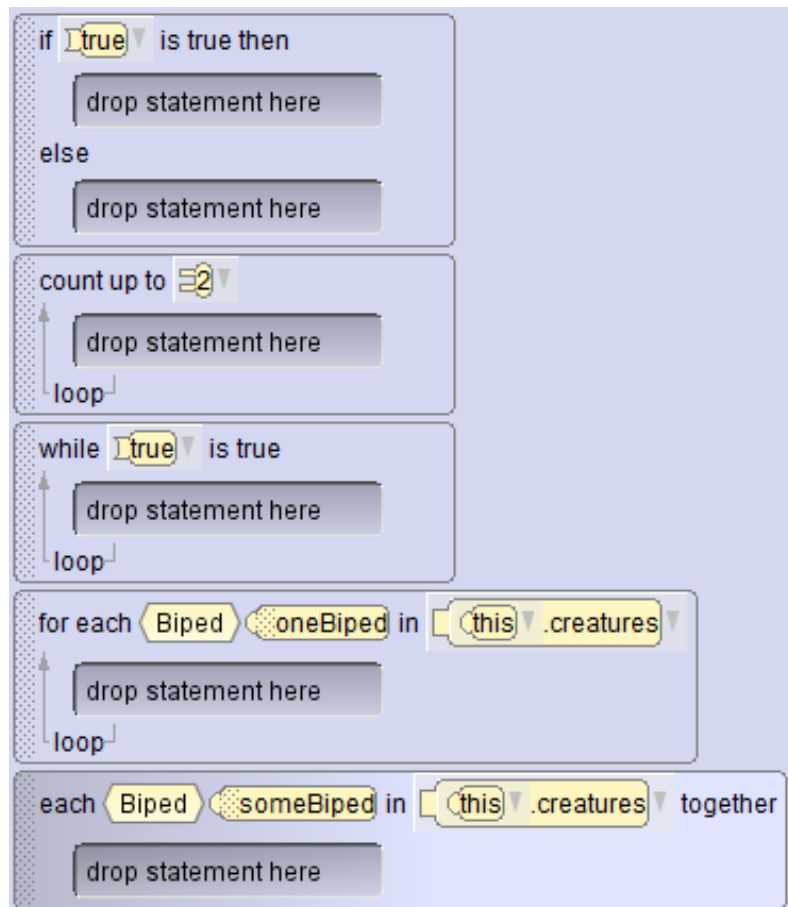
The screenshot shows a software interface for the 'this.panda' object. At the top, there is a search bar containing 'this.panda' and a dropdown arrow. Below this, there are two tabs: 'Procedures' and 'Functions', with 'Functions' being the active tab. A 'group by category' dropdown menu is visible. The main area is divided into several sections:

- Panda**: 's Editable Functions (3)'
 - edit this.panda getLeftEar
 - edit this.panda getRightEar
 - edit this.panda creatureAbove friend1: ???, friend2: ???
- Biped**: 's Editable Functions (0)'
- appearance**
 - this.panda getPaint
 - this.panda getOpacity
- size**
 - this.panda getWidth
 - this.panda getHeight
 - this.panda getDepth
- prompt user**
 - this.panda getBooleanFromUser message: \$\$\$
 - this.panda getStringFromUser message: \$\$\$
 - this.panda getDoubleFromUser message: \$\$\$
 - this.panda getIntegerFromUser message: \$\$\$
- other**
 - this.panda getDistanceAbove other: ???
 - this.panda getDistanceBehind other: ???
 - this.panda getDistanceBelow other: ???
 - this.panda getDistanceInFrontOf other: ???
 - this.panda getDistanceTo other: ???
 - this.panda getDistanceToTheLeftOf other: ???
 - this.panda getDistanceToTheRightOf other: ???
 - this.panda getVantagePoint entity: ???
 - this.panda getVehicle
 - this.panda isAbove other: ???
 - this.panda isBehind other: ???
 - this.panda isBelow other: ???
 - this.panda isCollidingWith other: ???
 - this.panda isFacing other: ???
 - this.panda isInFrontOf other: ???
 - this.panda isToTheLeftOf other: ???
 - this.panda isToTheRightOf other: ???
 - this.panda toString

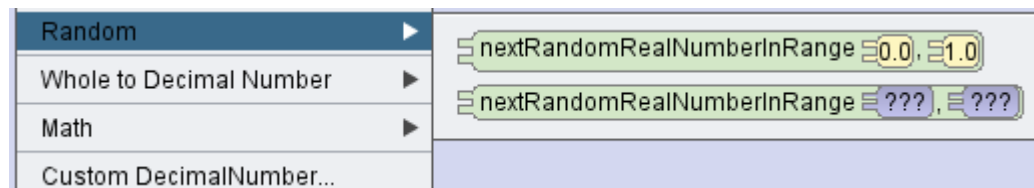
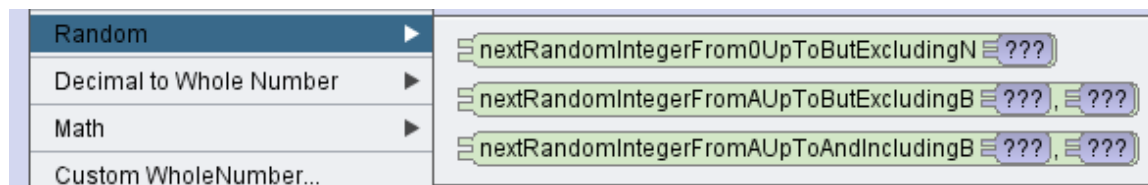
The screenshot shows a vertical list of functions categorized under 'joints'. The list includes the following methods:

- this.panda getHead
- this.panda getLeftAnkle
- this.panda getLeftClavicle
- this.panda getLeftElbow
- this.panda getLeftEye
- this.panda getLeftEyelid
- this.panda getLeftFoot
- this.panda getLeftHand
- this.panda getLeftHip
- this.panda getLeftIndexFinger
- this.panda getLeftIndexFingerKnuckle
- this.panda getLeftKnee
- this.panda getLeftMiddleFinger
- this.panda getLeftMiddleFingerKnuckle
- this.panda getLeftPinkyFinger
- this.panda getLeftPinkyFingerKnuckle
- this.panda getLeftShoulder
- this.panda getLeftThumb
- this.panda getLeftThumbKnuckle
- this.panda getLeftWrist
- this.panda getMouth
- this.panda getNeck
- this.panda getPelvis
- this.panda getRightAnkle
- this.panda getRightClavicle
- this.panda getRightElbow
- this.panda getRightEye
- this.panda getRightEyelid
- this.panda getRightFoot
- this.panda getRightHand
- this.panda getRightHip
- this.panda getRightIndexFinger
- this.panda getRightIndexFingerKnuckle
- this.panda getRightKnee
- this.panda getRightMiddleFinger
- this.panda getRightMiddleFingerKnuckle
- this.panda getRightPinkyFinger
- this.panda getRightPinkyFingerKnuckle
- this.panda getRightShoulder
- this.panda getRightThumb
- this.panda getRightThumbKnuckle
- this.panda getRightWrist
- this.panda getSpineBase
- this.panda getSpineMiddle
- this.panda getSpineUpper

If, loops, and creating an array element.



Random Integer and Decimal Numbers



Events

this addSceneActivationListener

declare procedure **sceneActivated**

do in order

this myFirstMethod

this addTimeListener 1.0 add detail

declare procedure **timeElapsed** event getTimeSinceLastFire

do in order

drop statement here

this addKeyPressListener add detail

declare procedure **keyPressed** event isLetter event isDigit event getKey event isKey key:

do in order

if event isKey S is true then

drop statement here

else

drop statement here

this addMouseClickedOnObjectListener, setOfVisuals new Visual[] { this.bunny, this.panda, this.panda2, this.panda3 } add detail

declare procedure **mouseClicked** event getScreenDistanceFromLeft event getScreenDistanceFromBottom event getModelAt

do in order

if event getModelAtMouseLocation == this.panda is true then

drop statement here

else

drop statement here

this addCollisionStartListener new SThing[] { this.bunny }, new SThing[] { this.panda, this.panda2, this.panda3 } add detail

declare procedure **collisionStarted** event getSThingFromSetA event getSThingFromSetB

do in order

drop statement here

this addDefaultModelManipulation