Whiteboard Software

# Outline

Software that allows easier creation of interactive on-screen resources. It should make it easier to create tables, move objects about, be able to accomplish more, more intuitively.

Like ActivInspire, without the limitations. But, to some extent, like a basic PowerPoint, in that it can easily show text and graphics, gradually revealing in sequence, but having options for more variability.

It won’t have the slide and text animation features of PowerPoint, but should have equal or better interactive animation capabilities.

Ideally, it will have features for both single touch and multiple touch screens.

## Issues with existing software

### Too many tools

ActivInspire has a cluttered toolbox. For example, its Shape tool gives the option for 21 different types of line and 26 different 2D shapes – most of which won’t be needed in 90% of cases.

The 80-20 rule suggests we should keep the basic tools to a minimum, with the option of creating (or importing) those tools a particular user might need.

The most common variations can be easily stored and accessed using the Palette idea.

# Terminology

## Tool

An element used to create, edit and interact with objects on screen.

Tools are not objects, but Button Objects can be created from tools to store settings for that tool.

At startup, tools revert to default settings (which can be edited).

These should be kept to a minimum, with prior collections of the results from using these tools (i.e. Objects) stored in a “palette” for future re-use.

### Button Objects

A tool can be added to a palette or the board, in which case it becomes a Button Object, which stores all the current tool settings for that tool. This allows, for example, a red thin pen or a wide highlighter pen to be included with a particular board.

## Board

The part of the screen showing the content (the collection of objects).

A board is a plane of fixed (and extendable) or infinite size, which has a background colour or style (Graph paper?). Think of the different black and white boards in classrooms and lecture halls – rolling, flipping, moving across each other.

A board’s scale allows the placement of objects to scale. E.g. an object can be placed “5 metres” above a “ground object”, where “5 metres” is determined by the board’s scale.

## Objects

An element, or group of elements, on a Board. Object Properties can be applied to objects.

Objects can be created by Tools, or can be imported images.

Videos and music files should also be catered for, as well as links.

## Properties

Properties allow objects to be interacted with and to interact with each other, triggering effects and giving some basic animation to the screen.

### Object Properties

Properties can be assigned to Objects, turning them into specialised tools. An Object assigned with “Deleter” and “Container” properties can turn it into a recycle bin, where objects are discarded and can be recovered.

### Board Properties

Properties affecting all objects on a particular board.

A board can have a grid, allowing snap to grid functions.

A board has a friction value, allowing objects to skate across it or stick rigidly.

Multiple boards can have their properties set simultaneously, doing away with the need for Global Properties.

## File

A collection of boards. The saved file.

## Palette

The collection of tools, colour picker and objects that can be applied to a board.

# Palettes

Dragging anything from a palette places a copy of the object or button (with the current settings) onto the board.

Clicking on a Button object (on the palette or the board) activates that tool with the stored settings.

There can be multiple palettes, and the Toolbox can be a specialised palette, accepting only Tool Button objects which are added on an application level.

## Toolbox Palette

Part of the toolbox is reserved for Button Objects – these can be added by the user to add buttons they expect to use from day to day.

## General Palettes

For tidier storage, multiple palettes are available for storing and organising a variety of objects and Button objects.

Ideally, these will be easy to search and locate different objects, perhaps using tags, categories, etc.

## File Palette

A file palette is stored with a particular file. This is available throughout the various boards. Objects and Buttons on here are expected to be of use while developing and delivering the series of boards in this file.

## Board Palette

Not strictly a palette – any Button Object dragged to a board acts as a mini customisable toolbox just for that board.

# Tools

A set of basic tools allows the creation of vector images, lines and importing of text, bitmap images, etc.

A tool can be customised and added to the Palette or the Screen. In this way, a particular pen type, etc., can be selected without having to mess with loads of settings during use.

E.g. set up a thick orange highlighter, or a thin red straight line tool. These are added to the screen as "buttons" or to the palette (where they can be selected or dragged to the screen).

## Pen Tool

Most likely will need to be different to a “freeform” line tool. The pen tool can produce “word” and “sentences” that will need to be treated differently to general shapes, I think??

Such creations need intelligent grouping. Ideally, words and sentences are easily joined and separated.

* ActivInspire tends to create objects for each continuous use of the pen tool, as long as the objects are in close proximity.
* SmartBoard software used to create objects for each continuous use of the pen tool, but this was easily “tricked” by manipulating the hardware pen tray that the pen had been returned and removed. In this way, e.g., an algebraic expression could be broken up into terms while being written.

Furthermore, annotations on top of such text should be intelligently “added” to the text (similar rules should apply to typed text). So a highlight remains with the word as the word is moved.

## Eraser

The eraser can erase pen annotations. Can it also erase parts of Shape objects? E.g. deleting lines?

## Line Tool

Creates a line object:

* Straight (inc. constrain to horizontal or vertical), arc, freehand
* Solid or dotted styles
* Thickness
* Colour (single colour only)
* Transparency (highlighter)
* Different ends, set for each end (e.g. arrow heads, arrow feathers, empty).
  + The ends are displayed left to right to show the order of the ends, start to finish. So an arrow with feathers on left and point on right will have feathers on the start point and arrow at the end point.
* Square or rounded corners (for high thickness levels)
* Stretchy or Fixed? – in ActivInspire, the Connector tool creates lines that are anchored between objects so that, instead of moving when an object is moved, they stretch.

Drawing a closed shape does what? Create a Shape object.

### Line tool versus pen tool

Is it possible for this to act as a pen, as well as for lines?? Or would we need a separate pen tool? ActivInspire: the freeform line tool draws a series of line segments. If the object forms a closed shape (with no “dangling” lines), it fills the shape automatically. The pen tool draws a series of overlapping filled circles to create a smoother annotation. The erase tool can erase pen tool circles, but not line segments. The line segments can be edited by adjusting the underlining points. When scaled, pen annotations change thickness too. Line shapes do not.

## Shape Tool

Related to Line tool Creates a shape object. Each shape object has:

* Line thickness, including zero, along with the same settings as the line tool.
* Fill colour, including gradient styles and textures.
* Transparency

### Polygonal

A series of points joined by straight lines. The points can be edited and constrained to create particular polygons.

* Triangle – three points
  + Force right angle, isosceles or equilateral.
* Square – one point and a dimension generate the square.
* Rectangle – two points defining the diagonal.
* Rhombus – two points defining one side.
* Parallelogram – 3 points defining two adjacent sides.
* Arrowhead – 3 points defining two adjacent sides.
* Trapezium – 4 points, defining two opposite sides.
* Regular Polygon – 2 points defining the length of side, plus the number of sides. (So can also create equilateral triangle and square.)

### Elliptical

Shapes defined by radii. Can be set to a circle or an ellipse. Can also be set to a part circle (quarter, semi, three-quarter, or an % in between).

* Circle – a radius
* Ellipse – two radii
* Part circle – a radius, 2 angles for each “arm”

## Fill Tool

The ol’ bucket o’ paint can fill in closed areas of lines, pen annotations and change existing colours.

Fills can be:

* Solid
* Gradient (2 or more colours)
* Patterned / images ???

As with colour, the transparency setting is available for this too.

## Slice/Glue Tool

The ability to slice apart an object into two or join two into one would be good.

## Text Tool

Creates a text object.

A text object is a block of text, which can include:

* Multiple paragraphs.
* Bullets or numbered points.
* Different styles, decorations and colours.
* Different fonts and sizes.
* A background colour (or none).
* A transparency level.
* An outline (with similar settings to a line object).

So a “text object” is probably a collection of text objects, on a low level.

However, it should also be able to contain annotation – see below.

### Text Edits

Editing text, such as highlights, should be automatically attached to the text, so that if the text moves the highlights move with it. In this way, a text object should grow to join with annotations.

This could be done with a (on by default) property that automatically groups with any object overlapping it. So, highlights and pen marks are automatically grouped.

### Text Frames

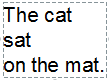
ActivInspire has a slightly irritating text frame. When first typing, the frame expands to contain the text. Subsequent edits stick to that frame size, so it must be manually expanded to accommodate if you want to avoid word wrapping.

Instead, make the “expand frame” drag point into a toggle that switches between locked and unlocked. When unlocked, the frame still expands when new content is added. When locked, it must be manually dragged to alter the text flow and avoid automatic word wrapping.

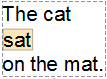
### Text Selection

In ActivInspire, double clicking existing text switches to edit mode. This also adds a toolbar to the top of the screen and pushes everything downwards in a jarring visual effect. Once in edit mode, double clicking will select a word and triple clicking selects a line. Click and drag selects multiple lines. Dragging on selected text produces a duplicate object of that text.

Possible alternative: Click to select a section of text. The entire frame is surrounded by a frame.



While selected, hovering over individual words will make them highlight in some way.



Clicking on a highlighted word selects that word.

Different tools might then be enacted upon the word – e.g. copying it, removing it from the frame into its own, etc.

## Table tool

To draw out tables really easily. Mainly visual – does not have Excel or Word-like properties.

The resulting cells can store an object, which will move with the table. If too large for a cell, the overhanging parts will be hidden (meaning the cells are draw from left to right, top to bottom).

When active:

Click and drag out a rectangle. The table sets up a particular number of rows and columns by default, which grow as the table grows.

Columns and rows can be added/removed in two ways – either adding/removing ones on/from the existing table, or changing the size of divisions on the existing table (size doesn’t change).

Individual rows / columns can be resized manually.

## Undo/Redo

It has to be possible to reverse changes and reapply changes.

## Other tools

There needs to be options for clearing the page, etc. too, as well as general colour and thickness settings.

# Object Properties

The main issue to resolve is the distinction between properties that act upon other objects, and those that act on themselves.

For example, an object might Replicate itself, akin to ActivInspire’s drag-a-copy. (This could be done by adding an object to a Palette, which automatically gives this ability.) Or it might replicate other objects passed to it.

An object with Bounce – does it cause other objects to bounce, or does it bounce itself?

Launchers & Motors – one gives velocity to other objects, one gives velocity to itself.

Position – one positions itself (??) one positions others.

## Default Properties

These properties are assigned to every object.

* X,Y – position on the board – this is where the ORIGIN of the object is displayed
* OffsetX, OffsetY – the different between the ORIGIN and the actual display position of the object.
* Z – Overlap order within the LAYER.
* Layer – each board has one or more LAYERS. Any layer can be LOCKED so that the items on it cannot be moved by the user.
* Visibility flag – whether the object is displayed
* Name / ID – identifier, internal and friendly name for interaction
* Scaling – how large they appear – for x and y directions
* Rotation – the angle to display them – rotate in all dimensions?
* A pointer to parent and child objects – objects can be chained/grouped together.

### Origin

Every object has an Origin – this is a point used for determining various calculations for the object. It’s the point about which the object rotates.

## Tool Object Properties

These properties are assigned to individual tool object properties

* Colour(s) – lines and fills have a colour.
* Thickness – how thick lines/outlines/pen strokes are.

## Group Object Properties

When objects are grouped, a Group Object is created which acts as the parent of all of them. This object has certain properties which either overrule or set the baseline for objects’ individual properties:

* X,Y Position – baseline, so other objects maintain their relative position.
* Rotation – baseline, so other objects maintain their current rotation.
* Layer & Z order – a Group object has its own layer and z order – or none?? – while child objects maintain their own layer and z-order (allowing similar “reveal” tricks to those in ActivInspire).
* Scale – baseline, so other object maintain their relative scale.
* Transparency – Overrule.
* Name/ID – has its own identifier
* Visibility – if invisible, this only overrules child objects set to visible.

## Tags

All objects can be assigned one or more keywords, which can be used for effect triggers.

## Position

Objects have this property by default. It allows the positioning of the object.

An object can be picked up by clicking on it, moving it with the mouse, and dropped by releasing the mouse button.

Or new values can be entered manually for X and Y.

### Position Details and Parameters

This is a property by DEFAULT and PERMANENT (cannot be removed).

* Position (x,y vector) – physical location on screen
* PagePosition (x,y vector) – physical location on BOARD (can be off board)
* Z Order - whether it appears above or below other objects on the SAME LAYER
* Layer – which layer the object appears on.
* Grabbed – whether it’s held by the mouse or not – possibly elevate this to Object?

## Movement

This is a property by DEFAULT (but can be deleted to prevent movement).

Need to add restrictions, so will only move horizontally or vertically.

* Velocity (x,y vector) – the velocity vector the object is currently moving in
* Direction – the direction the object faces. A normalised vector of the last non-zero Velocity vector.
* Max. speed – restriction on how fast the object can move (when not being dragged by the mouse).

## Rotation

## Solid

Objects with the Solid property cannot overlap if they are on the same layer, regardless of their Z value (order within the layer).

Both objects must be solid for this interaction to be valid. However, objects WITHOUT this property can still interact with the edges of the Board (e.g. Bouncing off them).

Partly equivalent to “Can block” in ActivInspire, although that only requires one object to have the Can Block setting – then all other objects are thus restricted.

Possible merge with Bounce?

Only if both objects are Solid (edge of the Board is solid or a Destructor, destroying any objects leaving it) will this feature come into play. In addition, the resultant interaction relies on a Reaction setting:

* Stop & Stop – both objects stop.
* Bounce & Bounce – both objects bounce off each other.
* Stop & Bounce – the bouncing object bounces.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **1st Object** | |
| **Stop** | **Bounce** |
| **2nd Object** | **Stop** | 1 & 2 stop | 1 bounces off 2 |
| **Bounce** | 2 bounces off 1 | 1 & 2 bounce |

Objects without Movement Property are automatically set to Stop. – Would it therefore make more sense to add this to the Movement Property? Something with movement by default will Stop, but can be set to Bounce. Anything without movement is automatically Stop. BUT still need the Solid setting…

Perhaps there should be an Elasticity setting too, which controls how much acceleration a bouncing object receives when it bounces. This could replace the Stop/Bounce setting. A high elasticity with no movement property would give a colliding object some increased velocity. Objects would only stop dead on impact if the friction setting of the object behind them was high. So all moving objects would potentially bounce.

## Bounce

Causes objects to bounce? Or bounces off objects? Or both?

The issue seems to be – does the property make the object behave in that way, or make other objects behave in some way? Do you create a rubber band wall? Or create objects that bounce off walls? Guess it depends which bit is rubber.

Unless the default for such “physics” is the bounce, and an object must be made “sticky” to hit an object, or be hit by one, and for the two to end up stopped dead at the point of impact.

Merge with Solid?

### On Collision

What if the Movement property has an On Collision setting? This requires the Collider and Collidee to be Solid. SFML book has a “Collidable” component, and uses a Collision System class, requiring Position and Collidable components from those that use it.

It could be:

* Stop – object stops dead
* Bounce – object bounces

## Hunt

An object with Hunt will, if able to Move, move towards a target object (specifically its Origin).

If it can only Rotate, it will turn to face its target (effectively treat the target as “magnetic north”).

## Gravity affected

Objects with this property are affected by Gravity, if it is active on the current Board.

## Gravity Field

Exerts a gravity field. A direction objects that are Gravity Affected will move in.

## Numerical Value Property

Objects with this property can have a number of values assigned to them, with different descriptions.

Example: Weight(kg) = 30; Height(cm) = 75

Objects can also be assigned to “measure” a particular value with the Accumulator property. Any objects placed on top of them will, if that value exists, be added together.

This is akin to the Custom field in database software like Music Collector. A bunch of objects can all be assigned a “Weight” value. Then, when those objects are placed on an object that can “measure” “Weight”, the total weight is returned (HOW?).

### Numerical Value Attributes

* Description – is displayed in the property, and used as an identifier (so all descriptions must be unique).
* Value – a numerical value (LIMITS?) is given. This can be positive, negative, and be a decimal.

### Numerical Value Manager

A Numerical Value Manager is used to coordinate the different value types. It stores a list of all descriptions used by all objects with the Numerical Value property, making this list available to other properties, as well as shortcuts to assign existing properties to other objects.

E.g. An object is assigned the Numerical Value property. The Description is set to “Weight (kg)”, and a value of 5 assigned.

The Numerical Value Manager checks that this description is not already in existence. It is not, so it stores this description and creates a shortcut property called “Weight (kg)”.

Another object is assigned the shortcut property “Weight (kg)”.

Another object is assigned the Numerical Value property, and the description is also set to “Weight (kg)” As the Numerical Value Manager recognises this description, it does not store it again.

### Uses

* Weight, Size, Cost

### Expansion

Should be a way to link different Numerical Value properties, such as “Weight (kg)”, “Weight (g)” and “Weight (lb)”. This would require a multiplier and constant to convert between them.

## Accumulators

An object with the Accumulator property will count up something.

It can use existing values (such as height, width, or the total number of objects – an option determines whether grouped objects are treated as one or multiple).

It can the number of times something happens (e.g. times it has been clicked; times an object has passed across it)

It can count a Numerical Value, via a list from the Numerical Value Manager.

### Accumulator attributes and tools

* Display total – whether and how to show the total
  + Show value description (“Weight (kg) :”) or a custom label (inc. blank).
  + Show value or not.
* Reset – set the counter back to zero.
* What to do with objects that don’t have that value
  + Either accept as zero, or report as a problem / automatically out of the range.
* What to do if the total hits a range or value
  + Continue processing further properties
  + Prevent further properties from being processed
  + Skip next X properties
  + The message to be displayed (“Too many objects in the container!”; “This object is too heavy!”; “This object will not fit!”)

## Containers

An object set as a container can “hold” other objects.

Objects dragged onto a container object are removed from the screen and “stored” in the container object. While contained, some components of objects are inert:

* Self-destruct will still work.
* Duplicators will not work.
* Motors will not work.

Dragging from the container will remove an item from the container.

* By default, this will remove the last object put in (LIFO – a Stack)
* Can be set to remove the first object (FIFO – a Queue)
* Can be set to remove an object at random.

Objects can be shuffled in the container.

The contents can be viewed – they will be shown in their current “order”, based on the current setting (LIFO or FIFO). Shuffling permanently changes the order.

Objects can be scattered from the container – the entire contents are emptied in random locations on the screen. IMPORTANT, the z value of each object would also need to be changed, as well as x & y, to disguise the order. Option – to keep the z order the same? Or to set it to the order the object is removed?

### Container Attributes and Tools

* Hide / Show objects – Hide by default, otherwise objects remain on the object wherever they were placed.
* View contents – displays the objects in the container as a list. Allows the changing of the order, by drag and drop, alphabetical, random and other options such as “size”.
* Empty – clears the contents of objects, placing them on the board.
* Randomise/shuffle – randomly orders the contents (without having to view the contents, so the order is unknown).
* LIFO/FIFO – determines whether objects are removed from the top or bottom of the list.

### Working in conjunction with Accumulator

If an object has Container and Accumulator, the Accumulator should be processed first (placed higher in the Properties list). This enables the container to have a limit:

* The accumulator measures a property such as Count, Weight or Size.
* When the limit is reached, the Accumulator does not process the Container object
  + # items – places a maximum number of items that can be contained
  + Item size – places a limit on the size of an object
  + Total size – places an overall limit on total size of all objects.
* Take no size as: (determines how to deal with an object with no set size).
  + Zero
  + Infinite

If a container can be placed inside another container. How would limits apply?

## Replicators

An object with the Replicator property will make a copy of any object placed on top of it.

1. Object intersects with the replicators “hotzone”.
2. Is this Object part of a group? If so, the entire group of objects should be replicated.
3. The object/object group is RECORDed by the replicator.
4. If the replicator is set to AUTO, the object is replicated as many times as the replicator is set for, and the object record is noted as being COMPLETE
5. Once the object moves away (is no longer intersecting with the hotzone), the RECORD is removed.
6. While intersecting, if the replicators REPLICATE button is pressed, as many copies as set are produced.
7. What about the copies?
   1. The replicator has a choice:
      1. Leave in situ, but add to the record as COMPLETE, so they don’t get auto duplicated.
      2. Pass the copies to another Property, such as a Positioner.
8. Timer – if a timer is set, all objects still in the zone are replicated when the timer reaches zero, regardless of their COMPLETE status.

## Positioner

A positioner property will place an object somewhere on the Board.

* Place at specific coordinates.
* Place within a zone
  + Randomly within the ZONE
  + Filling the zone from left to right/top to bottom

## Launchers & Motors

A Launcher applies a one-time force to an object. It can act upon multiple objects passed to it. This is like a "flick".

A Motor is fixed to an object and provides a particular acceleration up to a set maximum speed.

### Launcher

This will apply a velocity to an object passed to it.

* The vector can be specified or random, either in both axes or differently for each, meaning:
  + Random in both x and y directions.
  + Random in x and specified in y OR vice-versa.
  + Specified in both x and y.

### Motor – Sub-Property

A Motor alters a variable of the object it is assigned to over time.

Specifically, it continues “motion” in the current direction of that variable.

* A turning object will continue to turn.
* A moving object will continue to move.

Would be nice to get this working for any value (rather than have separate motor types for moving and turning) – attach it to the x and y part of an object’s position, and the object will move. Attach it to the angle property, and it will rotate about the object’s Origin.

* Direction vector – a normalised vector showing the current direction the object is travelling in. This would only apply to a vector value, with two parts (x and y), not to an angle. Could this apply to anything other than position??
* Acceleration – how slowly the object builds up to its speed. The opposite of friction, which decelerates.
* Max Speed – the object’s target speed. This cannot be exceeded. Should this be controlled by the Movement property rather than the motor?
* Speed – counter being added to by acceleration. This can also be set to Constant speed, which means the object just moves/rotates at that rate, which disables acceleration. For movement, the Direction vector is multiplied by this value and added to the object’s current position vector.
  + A speed of 5 applied to an angle might mean 5 degrees per second (so an anticlockwise spin). Negative values would reverse the direction (clockwise). This would ignore the direction vector.
  + A speed of 5 applied to a position (both x & y) would use the direction vector to calculate a new position.
* Always on or not:
  + Always on: The motor is always applied.
  + Pulse: "pushes" the object:
    - after a period of time, or
    - when the object falls below a particular speed.

#### PropulsionMotor class

In Prototype 1, the PropulsionMotor class has these variables:

* Type
  + Constant (default) – the motor is always on. The object can still be moved by the mouse and accelerated by a “flick”.
  + TimePulse – the motor turns at regular intervals
  + MinSpeedPulse – the motor turns on when the speed drops below a threshold.
* Magnitude – how much “oomph” the motor puts into the velocity OR the velocity the motor sets. This variable is shared between the two settings.
* Counter – an internal counter for the TimePulse
* PulseSetting – if TimePulse, this is the delay between pulses. If a MinSpeedPulse, this is the speed threshold that turns on the motor.
* Accelerate – a Boolean to determine whether Magnitude is an exact speed to set the Object or an acceleration amount.

Acceleration fix:

1. Calculate magnitude of current velocity.
2. Increase magnitude by acceleration.
3. Normalise velocity vector.
4. Multiply normalised vector by increased magnitude.

Possible revisions:

* A Duration setting determines how long the motor is switched on for, when a Pulse type is set. Once the duration is over, the Counter is set to zero. OR the counter can be used for on and off, with another setting controlling the motor’s state.
* On/Off setting, for use with Duration above, and also a simple switch for user-control.
* Need an initial direction, otherwise the object doesn’t move until given a “kick”.
* Are there 2 types of propulsion motor? One that keeps an object moving in the current direction, and one that propels along a specified vector? Or one that is applied to x and y directions separately?

### Spawner

Like a launcher, this also holds a copy of an object which it replicates and sends off into the world, possibly with some random elements (colour, size, rotation).

## Randomizer

Return 1..x values between LOW and HIGH, either integers or floats, positive, negative or either.

* Low
* High
* Allow zero or not
* Weighted towards positive / negative / specific subsets.

The Randomizer can be linked to the Positioner to provide random locations on the board, or a random vector for the Launcher.

### Positioner, Launcher and Randomizer – how they might work together

An object can have multiple components assigned, which act as a conveyor belt.

It applies the methods of each component in turn. So, if it needs to act upon another object, it will apply the effects of each component in turn. If, say, it had Duplicater-Positioner-Launcher, it would duplicate an object, place it, then launch it in a particular direction.

The positioner has a number of settings:

* Zone – the boundaries to place the object.
  + 4 values to be specified (top left and bottom right, or top left, width and height).
  + Default is the entire screen, but can be set to the whole "board" or a smaller subset.
* Type of placement
  + Specific coordinates – 2 values to be specified.
  + Filling up zone, from left to right and top to bottom (or top to bottom and left to right).
  + Randomly (any coordinate that keeps the whole object within the zone or, failing that, keeps its "origin" within the zone. This requires the Randomizer sub-component.

Once the positioner has placed the object, it is passed to the Launcher, which then "flicks" it based on its settings:

* X direction - randomizer or specific
* Y direction - randomizer or specific

## Tracer

A tracer is a point assigned to an object. When active, it draws a continual line with a specified pen (i.e. thickness, colour, transparency) to trace out the loci of that object.

The resulting drawing object is a separate object from its originator.

Can be used to simulate a compass.

## Deleter

Effectively, the recycling bin, but this means you can have custom bins and more than one.

Anything dragged into its "zone" is deleted (or moved into temporary storage).

Can be set to immediately delete, or act as a container with fixed capacity storage (to "undo" accidental deletions), with First In First Deleted. (If it was unlimited storage, it might as well be a normal container.)

A grouped item counts as "one" deleted object.

## Self-Destruct

Like the deleter, except this applies to the object itself.

An object can be set to self-destruct (i.e. delete itself) based on a number of parameters:

* Timer - will delete after a certain time has passed. - Randomizer allowed
* Position - will delete if it reaches a certain zone on the screen. - Randomizer allowed
* Probability - will delete if a probability calculation succeeds.

## Friction – Board Property

Any object on this board can “slide” and will come to rest dependent on this setting.

# Objects

All objects have:

* X,Y – position on the board – this is where the ORIGIN of the object is displayed
* Z – Overlap order
* Layer – more global overlap order – allows a background, middle and top level, or perhaps even more if required.
* Visibility flag – whether they are shown / printed.
* Name / ID – identifier, internal and friendly name for interaction
* Scaling – how large they appear – for x and y directions
* Rotation – the angle to display them – rotate in all dimensions?
* Colour(s)
* A list of Object properties
* A pointer to a child object – objects can be chained together.

Objects can be made up of several parts. When multiple objects are turned into a single object, their properties are combined. Their basic properties (listed above) can still be edited individually in edit mode, but otherwise retain their “group” settings.

## By Default

### Movement

New objects are added to the same “layer”, positioned above all other objects on that same layer.

All objects can be repositioned by dragging them around by their centre.

The default background has 100% inertia, so objects stop moving the moment they are released. OR objects by default do not use velocity / friction, etc.

Objects can be restricted to movement in horizontal or vertical directions. Objects restricted in both directions are effectively pinned and cannot be moved. How would this work for groups?

Shape and line objects can be partially pinned, to make hinges / pivot points. Pivot points are either relative (will keep their relative positions within the object as the object moves) or absolute (are fixed on the Board). [Are pivot points just the Origins described under Rotating?]

(Note: in AI, an object restricted from moving will instead rotate about its “centre” when attempting to move it. You can set a “rotate about” to any of the 9 points of a shape [top-left, top, top-right, left, centre, right, bottom-left, bottom, bottom-right] as well as a specified coordinate, the centre of another object, or the “first line of text”[??]

In addition, objects can be restricted to follow a path – ideally a line, but if applied to a shape, it will follow the outline of the shape. Applied to an image or group, such as annotation, it will follow the rectangular boundaries.)

Momentum allows an object to keep moving for a while after being dragged.

A Motor keeps an object moving, following the restrictions set for it.

A force can be automatically applied – like a one time motor effect, launching an object in a particular direction.

Objects on the same layer can block each other. One or more zones can be created for an object which are impenetrable by other objects. Zones can automatically be created from the shape of an existing object, or be set independently (e.g. on an image, allow part of the image to act as a barrier, but the rest is a background). Hitting a zone causes an object to either stop (if no momentum) or bounce.

### Rotating

Dragging a shape or image by an edge or vertex will rotate it about its current Origin. For line objects, the drag point must be one of the ends of the line.

The origin of an object can be changed, and fixed to relative or absolute placement.

As with Movement, the objects stop rotating as they are released. Momentum could be added to allow them to “spin” freely. A Motor option could allow them to spin automatically.

### Scaling/Reflecting

Scaling line objects makes them longer without changing the thickness of the line.

Scaling up a freehand collection (result of writing) probably should make it bigger and thicker.

Scaling shapes and images similarly should make them larger or smaller.

Grabbing any “corner” should scale keeping proportion.

Grabbing a side allows distortion in that dimension, but avoid shearing when the object is rotated? (See AI).

Images can be reset to their default size.

When scaling text, AI just changes the textbox to allow different flow of text, but it should also be possible to scale the text size similarly.

### Visibility

An object can be opaque, translucent or transparent.

For text, a component could exist that automatically allows a text box to appear or disappear when clicked on.

## Line Object

Lines and curves – open shapes rather than closed (a closed shape is a Shape object with an outline colour but no interior fill).

Arrow ends optional.

## Shape Object

Fill colour and line colour.

Line style. Square or rounded corners.

Fill style. Gradient

### Polygon

Shape made up of series of points joined by straight lines. The individual points can be edited / added / deleted.

### Elliptical

For ellipses and circles.

## Text Object

A line of text or multiple paragraphs. Includes styling such as bullets, bold, italics.

Colour (paper and ink), fonts, super and subscript.

## Table Object

The resultant object of the Table tool. A series of lines of variable thickness and style.

# Zones, States and Triggers

## Zones

A zone is an area, most likely rectangular. It can be an area on the screen/board. It can be an area on an object.

It can be displayed, as a dotted or shaded area, but is otherwise invisible.

When objects are inside a zone, the zone can trigger an effect. OR it passes a value to the object, which can be programmed to react to it. ?? This is important, and similar to the problem of the way object properties work; is the zone triggering an effect on an object, or is an object detecting it has entered a zone?

### Component Zones

An object on a zone linked to a Duplicator is duplicated when the component is activated.

An object on a Launcher zone is launched.

### Screen Zones

A zone on the screen is still linked to a component. So a zone might play a sound when “correct” or “wrong” objects are placed into it, and possible eject wrong objects while repositioning correct one.

The zone would have a list of trigger values which would match, or not, value lists attached to the objects interacting with it.

### Zone Shapes

The simplest shape is a rectangle, but would it be viable/useful to have circular/elliptical and triangular zones too?

#### Rectangular Zones

This is already catered for in SFML, and can easily be created bespoke if needed.

#### Circular Zones

For a point to be within a circular zone, the magnitude of the vector between point and the circle’s centre must be less than the circle’s radius.

#### Triangular Zones

## States

An object can be assigned different states, which redefine its appearance. These different states can be switched between depending on the zone they inhabit.

### Basic states

* Default – how the object usually looks when no zone is affecting it, or no other matching state has been set.
* Dragged – how it looks when being dragged

However, this wouldn’t be enough for some cool uses: e.g. an equation showing ‘7x = 35’ – when dragging the 7, it should change to ‘×7’, then switch to ÷7 as it passes over the = sign, then becoming, perhaps, ‘/7’ (vertically written) when released.

### Uses

Words being sorted into nouns and verbs might change colour depending on whether they are placed into the correct or wrong boxes.

Dragging part of an equation across the equals sign might change it from plus to minus.

## Triggers

This could be a powerful part of the system. A trigger is added to either a zone or an object (though a zone should probably be a type of object??).

When an object enters a zone, a trigger might be fired – either the zone notifying the object, or vice versa. Or even a third party being notified?? (But that would suggest the zone should belong to that third party.)

# Screen Layout

## Palette

The palette stores objects and tool buttons for dragging onto the board or (in some cases) selecting/using directly from the palette (so they don't get in the way of the board).

Each palette is unique to a file, and is shared across multiple "boards".

Objects and buttons can be added to the palette either from the screen and toolbox respectively, or from a library of previously stored objects and buttons.

## Selected Object's Component list

The selected object has its list of components displayed for editing.

Components can be added, removed and reordered, and their settings changed. (The order of components can change their behaviour.)

## Toolbox

The toolbox contains a list of simple tools and their settings. They are used for:

* Creation of (vector) shapes, lines and pen marks.
* Creating text objects.
* Creating tables.
* Loading in bitmap images / videos.
* Erasing / deleting things.
* Editing the settings of the selected tool - only the necessary settings for the selected tool are shown.
* A list of customised buttons for permanent storage.

## Container view

If an object is a container, its contents can be viewed in this panel.

(or should this be part of the Component list??)

# Sample Workflow

1. Create a number of objects.
2. Assign one of the objects the Container property.

There are a number of objects. One is given the property of “Container”.

When this object is selected, the Container property settings are available to edit:

* View contents – displays the list of objects in the container – also allows the changing of the order, by drag and drop, alphabetical, random and other options such as “size”.
* Empty – clears the contents of objects, placing them on the board.
* Randomise – randomly orders the contents
* LIFO/FIFO – determines whether objects are removed from the top or bottom of the list.
* Limits:
  + # items – places a maximum number of items that can be contained
  + Item size – places a limit on the size of an object
  + Total size – places an overall limit on total size of all objects.
* Take no size as: (determines how to deal with an object with no set size).
  + Zero
  + Infinite

Another object is dragged and dropped onto the Container object. As there are no other properties assigned to the object, the Container is automatically activated: a reference to the object is sent to the object’s container.

EVENT: OBJECT DROPPED ONTO ANOTHER OBJECT.

If the object comes within limits, the object is removed from the board and stored in a separate location.

Later, the Container object is clicked on. If there are multiple properties that react to clicking on, either the properties are worked through in turn, or a pop-up selection allows the Users to choose which property to use.

The first or last object (depending on the Container property) is removed from the container and becomes attached to the cursor. The container is updated accordingly.

# Ideas

Many of these come from ActivInspire and PowerPoint creations. How to combine into one application?

* Tracer – affix a “pen” to an object and it draws the locus as the object moves.
* Anagrams – a way of jumbling and unjumbling words, or at least a PowerPoint style way of swapping out text chunks. Good to be able to break up and reassemble individual words.
* Straight lines and angles
* Right angles
* Constructions
* Hinges (lines connected at a point) / rotating around arbitrary points (AI’s XY Origin)
* Fractions – including mixed numbers, breaking shapes into pieces.
* Copying part of the screen to another and overlaying / Tracing paper / without background
* Highlighting
* Drawing on top of a shape, automatically staying with that shape / autogrouping
* Tables
* Graph paper
* Number lines – easily attach objects to ride on “rails”
* Change size just by dragging – e.g. height of water in jug / thermometer bulb
* Value connected to dimension of an object – e.g. water volume connecting to “height” of blue rectangle
* Ability to “pour” jug of water from one to another somehow??
* Easy “spider” diagrams
* Tool selectors actually on the board, for quick selection – e.g. a thick red highlighter, a blue thin pen, a dashed green line with arrows.
* Rotating gears with interlocking teeth.
* Rearranging algebra “blocks”
* Weighing scales / balance
* Rubber bands / peg boards
* Spinners (like dice)
* Multiplying values between objects – imagine a circle landing on a target – the value of the circle multiplied by the value of the target adds to a running score.

## PowerPoint vs ActivInspire

PP gives a (mostly) linear and uninteractive display of information, with jumping points to web sites, videos, etc. Animation is key for demonstrations.

ActivInspire is interactive, with the opportunity for data gathering, exploring ideas, etc. Animation abilities are very limited.

# Prototype

The SFML book examples give the following structure:

* Entities (here, Board Objects) are fairly simple structures.
* They are linked with individual Components (here, Properties) in std::unordered\_maps.
* To move an entity, it is linked to a Position Component and a Movement Component.
* None of these things “own” each other – they are all linked through map containers.
* Another class, S\_Movement, is responsible for actual movement calculations. It is set up so that any entity using it must have both Position and Movement components.
* Each member function gets a pointer to each component (by using Entity designation) and performs the necessary calculations, affecting each component as required.
* To render, another system, S\_Renderer, uses the Position component for each entity (as well as others, such as Drawable / Spritesheet components) to draw the entity where required.

## Stage 1

All objects are circular shapes. DONE – though the physical shape is currently drawn by the Object class.

The Board is finite in size. It has the Friction Property set low, so that objects can “skate”. DONE – although the Board object itself doesn’t actually exist.

All objects have the Position & Movement Properties. – DONE – the interaction between these and the Mouse cursor is controlled by a managing class, SysMovement.

RED: One object has the Bounce property but no Motor, so can be manually “flicked” around the board. – DONE – currently, all objects are doing this.

YELLOW: One object has a Motor Sub-Property. This object should stop when it hits the Board edge. – might reevaluate the Bounce behaviour. TO TEST

GREEN: One object has a Motor Sub-Property and a Bounce Property. This object should rebound from the Board edge and continue moving. DONE – the Motor is an object that can be optionally created within the MovementProperty object.

BLUE: One object has a Motor Property set to Pulse, and a Bounce Property. – TO TEST

MOTOR – created a PropulsionMotor class, which is created within the MovementProperty class. The constant motor is working, along with Pulse set to a time factor. But it slows down too quickly, so might be nice to add a feature where, instead of a one-off pulse, it actually \*runs\* for a period of time too. Acceleration (along with min speed pulse – might be related) isn’t working – need to review how acceleration should be implemented.

## Stage 2

Which of these should be in stage 2?

* Rectangular objects as well as circles are possible.
* Two layers – objects can be moved between layers.
* Different frictions can be applied, and objects on layers above react to frictions on layers below.
* Objects can be grouped, but retain their separate layer setting.
* When grouped, some properties are combined, some are retained by their respective objects, and some are superseded by the “lead” object.
* An object can be replicated and, if a group, all objects are duplicated.
* Objects can bounce off each other (resolve the Bounce/Solid/Elasticity setting).

1. Some rectangular objects are created, but are placed on a lower layer.
2. The circular objects from stage 1 are on an upper layer.
3. The upper layer objects can bounce off each other, but not bounce off any rectangular object on the lower layer.
4. The rectangles have a Friction property, which affects objects on the upper layer - some cause an object to stop altogether.
5. Bounce in MovementProperty is expanded to other possible reactions to the board edge. Each edge has a Stop, Slide, Bounce, Wrap or NoReaction setting. Stop means the object stops dead. Slide means its velocity in that direction only stops (causing it to possibly slide along the boundary if it still has velocity in the other dimension). Bounce means it bounces (as it currently works). Wrap means the object appears on the opposite side (a la old arcade games). NoReaction means the object effectively keeps moving.