**Esse**

Simply put, this is a framework for building presentations of a new generation. The emphasis is on supporting thinking processes, saving, envolving, presenting and sharing ideas.

**How to install and run**

To install this system you have to:

* Download Pharo (for example from [http://www.pharo-project.org](http://www.pharo-project.org/) )
* Run Pharo
* Download sources of this project or clone them using Monticello from <http://ss3.gemstone.com/ss/Esse>
* Create Esse project (Example: Type “EssePr:=EsseProject new.” In a workspace and press Do It in a menu. )

**Screencast**

Here is short screencast <http://www.youtube.com/watch?v=OVjx9KMd3i8>

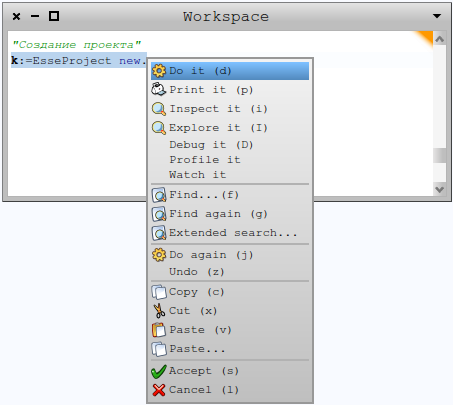
**Features and Capabilities**

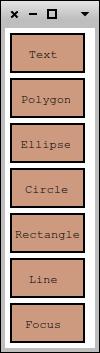
* Large workspace (scene)
* Adding any morph from Pharo to the scene
  + Interface with collection of generic components (morphs) commonly used in presentations
* Camera
  + Camera can move over the scene and show morphs from scene
  + Camera support scaling
* Scenario
  + Camera movement can be scripted
  + Presentations can be divided into several steps
* Managing of the presentation
  + Control panel to start/pause/continue/rewind/etc. presentations
  + Control panel for structuring presentations by tagging particular slides, parts, etc.
* Animation by signal

All the goals from proposal were achieved and considered functionality is implemented. But there is one thing I hadn’t enough time to do. It’s learning TDD and rewriting code using it.

**User Manual**

1. Getting started
   * Creating new Esse project
     1. Open Pharo
     2. Open Workspace
     3. Type “k:=EsseProject new.” and press “Do It” button in a menu



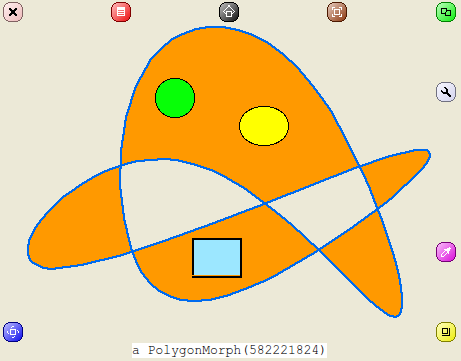
* + 1. After doing previous action you will see four windows
       - ProjectScrollWindow – main window, there you can create your presentation
       - EsseControlPanel – panel for managing presentations(/pause/continue/rewind/home)
       - EsseAddMenu – panel for adding standart components to presentations
       - PartMenu – panel for separation your presentation on the parts

1. Work with morphs
   * Adding morphs to presentation

To add morph into presentation you can use EsseAddMenu

* + - Button “Text” creates string morph that contents text “String Morph”
    - Button “Polygon” creates polygon Morph that can take custom forms
    - Buttons “Ellipse” and “Circle” create objects with round shape
    - Button “Rectangle” creates rectangle morph
    - Button ”Line” creates line morph that also can take custom form
    - Button “Focus” creates special morph using which you can create the scenario of your presentation
  + Manipulation morphs
    - Simple manipulations(move, resize, duplicate, rotate, change color, delete)

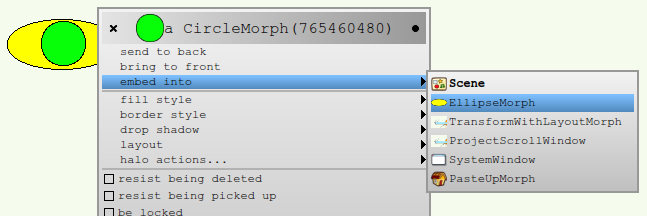
All these actions you can provide by using “halo menu”. Meta-click on morph, you want to change, will open “halo menu” of this morph. In Pharo, how you meta-click depends on your operating system: either you must hold SHIFT ctrl or SHIFT option while clicking scroll button. If you use a Macintosh without a second mouse button, you can simulate one by holding down the ⌘ key while clicking the mouse.



* Imbedding one morph into another

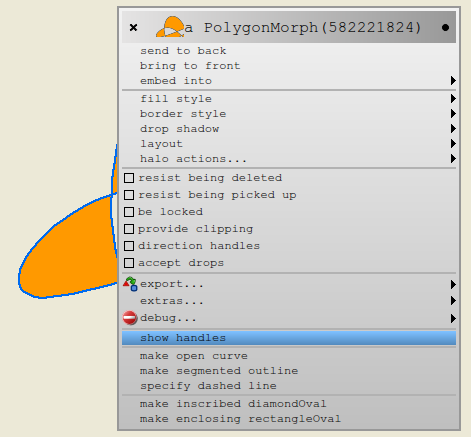
Put morph which you want to embed on another morph >> Open “halo menu” for upper morph >> Press button “Menu” >> Click “Embed into” >> Choose necessary morph.

Also you can embed whole system windows from Pharo to your presentation and show software with another level of interactivity.



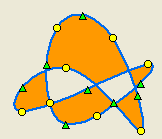
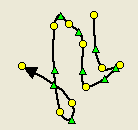
* + - Special manipulations

Every morph has its own list of special manipulations. Open “halo menu” and press in it button “Menu”. At the end of this menu is the list of special manipulations.



* + - * Action for changing shape(for polygon and line morphs)

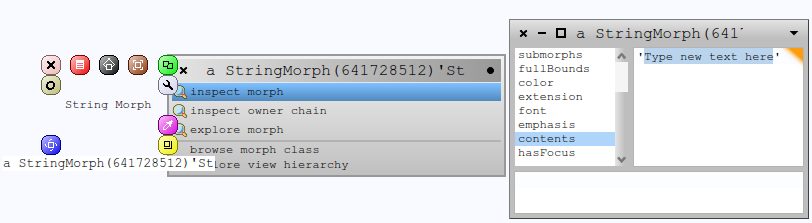
To open the change of shape you must hold SHIFT ctrl or SHIFT option while clicking right or left button.



By changing positions of yellow circles and green triangles you will change the shape of morph.

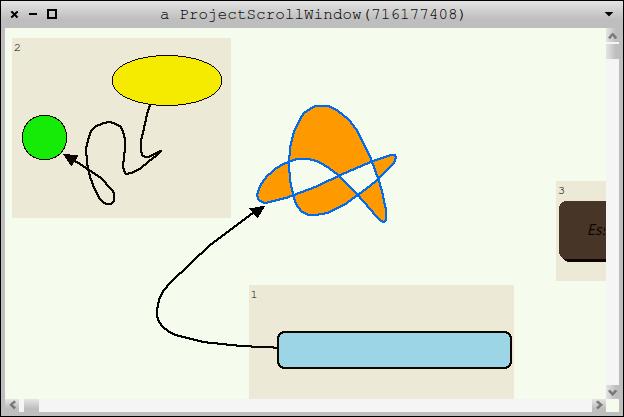
* + - * Action for changing content of string morph

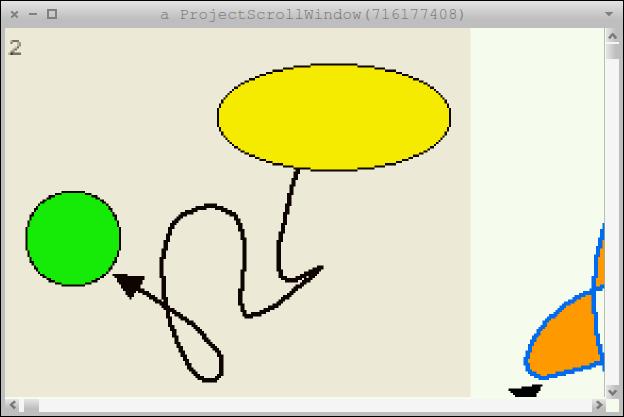
Open “halo menu” for string morph >> Press button “Debug” >> Click “Inspect It” >> In inspect window find “contents” >> Type new text >>Press “Ctrl+S” to save.



1. Scenario

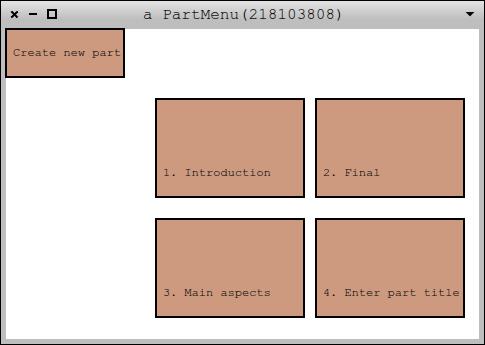
There are special objects in Esse that create the scenario of presentation that is called focus morph. With help of this morph you can mark necessary area and during presentation camera will adapt to the size of this marked area. Focus morph is created by clicking on button “Focus” from EsseAddMenu.





1. Dividing presentation on parts
   * Creating new part

In Esse there is a special panel for dividing presentations. This panel allows you to create parts and shows them.



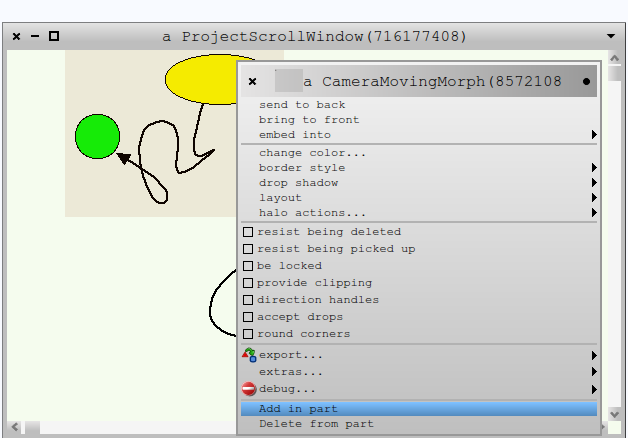
To create part you must:

Press button “Create new part” >> Type the title of new part.

* + Adding/removing focus morphs in/from part

To add/remove focus morph you must:

Choose focus morph and open its “halo menu” >> Press button “Menu” >> Click Add part/Delete from part >> Choose necessary part



* + Removing part

To remove part you must:

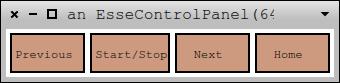
Choose part and open its “halo menu” >> Press button “Remove”

1. Animation by signal

This feature allows you simply add your ready-to-use animated morph into presentation. Animation starts when camera approaches this morph. All you need is to add your morph in presentation and mark area where the action of this morph will be. So if camera approaches focus morph with animated morph inside, the animation starts. Animation stops when camera starts moving to another focus morph.

1. Managing of the presentation

There is special control panel in Esse. You can simply manage your presentation with help of this panel.



“Previous” – button for moving to previous focus morph.

“Start/Stop” – this button starts presentation. Presentation will pause if you press this button at demonstration process. And if you press this button again the presentation will continue from “pause point”.

Button “Home” set scale at 1.

**Classes and methods**

* EsseProject – create Esse project.
* CameraMovingMorph – focus morph class.
* EsseAddMenu – add menu class.
* EsseCameraMovingMorphButton – button for creating focus morph.
* EsseCircleButton – button for creating circle.
* EsseControlPanel – control panel class.
* EsseEllipseButton – button for creating ellipse.
* EsseNewPartButton – button for creating part.
* EsseNextButton – button for moving to next focus morph in presentation.
* EssePolygonButton – button for creating polygon.
* EssePreviousButton – button for moving to previous focus morph in presentation.
* EsseRectangleButton – button for creating rectangle.
* EsseLineButton – button for creating line.
* EsseStartButton – button for start/stop presentation.
* EsseStringButton – button for creating text.
* EsseZoomButton – button to set scale 1.
* NextButtonInPart – button for moving to previous focus morph in part.
* Part – part class.
* PartMenu – part menu class.
* PreviousButtonInPart – button for moving to previous focus morph in part.
* ProjectScrollWindow – main window class.
* Scene – scene class.

**EsseProject instance**

* initialize – Esse project is being initialized in this method.

**EsseProject class**

* projectE – return variable projectE which contens ProjectScrollWindow(Esse main window)
* projectE: – set variable projectE

**CameraMovingMorph instance**

* addCustomMenuItems: aCustomMenu hand: aHandMorph – add actions “Add in part” and “Delete from part” to halo menu of this morph.
* addInPartMenuAction – creates special choosing form for “Add in part” action.
* afterStart – change color of CameraMovingMorph.
* delete – delete CameraMovingMorph.
* deleteFromAllParts – delete this CameraMovingMorph from all parts.
* deleteFromPart: aNumber – delete this morph from part with index=aNumber.
* deleteMenuAction – create special choosing form for “Delete from part” action.
* deletePartChoises – special collection for choosing form “Delete from part”.
* handleKeystroke: t1 – handler keystrokes for this morph.
* indexInCol/indexInCol: - return/set index in collection of CameraMovingMorphs.
* initialize – initialize of CameraMovingMorph.
* metka – return skip this morph or not.
* metka: - set skip this morph or not.
* partChoises – special collection for choosing form “Add in part”.
* partT/partT:aTitle - return/set part title.
* size – return size of this morph.
* textI/textI: - return/set textbox with index of this morph.
* updateCountMorphs – update position of camera.
* updatePartIndexes – update indexes of CameraMovingMorphs in parts.

**EsseAddMenu instance**

* initialize – initialize menu for add components.

**EsseCameraMovingMorphButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates CameraMovingMorph.
* initialize – initialize button.

**EsseCircleButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates CircleMorph.
* initialize – initialize button.

**EsseControlPanel instance**

* initialize – initialize control panel.

**EsseEllipseButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates EllipseMorph.
* initialize – initialize button.

**EsseNewPartButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates part.
* initialize – initialize button.

**EsseNextButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler change camera destination point to next in cameraMovingMorphsCollection.
* initialize – initialize button.

**EssePolygonButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates PolygonMorph.
* initialize – initialize button.

**EssePreviousButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler change camera destination point to previous in cameraMovingMorphsCollection.
* initialize – initialize button.

**EsseRectangleButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates RectangleMorph.
* initialize – initialize button.

**EsseLineButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates LineMorph.
* initialize – initialize button.

**EsseStartButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler starts/stops prezentation.
* initialize – initialize button.

**EsseStringButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler creates StringMorph.
* initialize – initialize button.

**EsseZoomButton instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler set scale 1.
* initialize – initialize button.

**NextButtonInPart instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler checks the condition of transition.
* initialize – initialize button.
* moveToNextInPart – change camera destination point to next in focusMorphsInPart.

**Part instance**

* addInPart: aMorph – add morph in part. aMorph – morph.
* buttonInitialize – initialize and add buttons to Part morph.
* delete – delete part.
* deleteButtons – delete buttons from Part morph.
* deleteButtonsBesideMe – delete buttons from all Part morphs beside this morph.
* focusMorphsInPart – return collection of CameraMovingMorphs in this part.
* handlesMouseDown: evt - turn on “mouse down” handler in this morph.
* initialize – initialize Part.
* initializeTextMorphs – initialize TextMorph of this morph.
* isButtonVisible – return visible or not buttons of this morph.
* isButtonVisible: - set visible or not buttons of this morph.
* mouseDown: evt – “mouse down” handler change camera destination point to the first in this Part.
* partNum/partNum:aNumber – return/set index of this part in partsCollection.
* partNumText/partNumText:aText – return/set text boxes with index of this part.
* partTitle/partTitle:aTitle – return/set title of this part.
* updatePartNumbers – update index of this part in partsCollection.

**PartMenu instance**

* initialize – initialize menu for managing parts.

**PartMenu class**

* createPartMenu – create menu for managing parts.
* getPartMenu – return part menu.
* updatePartMenu – update position of parts in part menu.

**PreviousButtonInPart instance**

* handlesMouseDown: t1 – turn on “mouse down” handler in this morph.
* mouseDown: t1 – “mouse down” handler checks the condition of transition.
* initialize – initialize button.
* moveToPreviustInPart – change camera destination point to previous in focusMorphsInPart.

**ProjectScrollWindow instance**

This class contents a lot of methods from system class GeneralScrollPane.

* adoptPaneColor: t1 – set color of window.
* afterStart – set color of CameraMovingMorph.
* cameraMovingX: aValueX – camera moving along X-axis. aValueX – current position.
* cameraMovingY: aValueY – camera moving along Y-axis. aValueY – current position.
* changeScrollerTableLayout – change scroller depending from window size.
* countStepH – return step along the X-axis.
* countStepH: – set step along the X-axis.
* countStepV – return step along the Y-axis.
* countStepV: – set step along the Y-axis.
* defaultColor – return default color.
* defaultScrollTarget – set scroll target.
* distanceBeetwenPositionsX – return distance between current position and destination point along the X-axis.
* distanceBeetwenPositionsY – return distance between current position and destination point along the Y-axis
* doLayoutIn: t1 – update scrollbars depending from window size.
* extent: t1 – update scrollbars when window is being changing.
* fitScrollTarget – fit scroll target under scroller.
* hHideScrollbar – hide scrollbar along X-axis.
* hPageDelta – return delta along X-axis. Delta is a ratio of the window width to the scroll target width.
* hResizeScrollbar – resize scrollbar along X-axis.
* hScrollbar/hScrollbar: t1 – return/set scrollbar along X-axis.
* hScrollbarInterval – return ratio of the window width to the scroll target width in float.
* hScrollbarNeeded – return is scrollbar along X-axes needed or not.
* hScrollbarShowing – return is showing scrollbar now or not.
* hScrollbarValue – return scrollbar inner value.
* hScrollbarValue: t1 – set scrollbar inner value.
* hSetScrollDelta – set scrollbar value. Scrollbar is the value of a single transition.
* hShowScrollbar – show scrollbar along X-axis.
* hUpdateScrollbar – update scrollbar along X-axis.
* handleKeystroke: t1, keyStroke: t1 – keyboard handlers.
* handlesKeyboard: t1 – turn on keyboard handler.
* handlesMouseWheel: t1 – turn on mouse wheel handler.
* initialize – initialize Esse main window.
* isNext – move camera to next CameraMovingMorph if current position must be skipped.
* magnification/magnification:aScale – return/set scale of the scroll target.
* mouseWheel: t1 – mouse wheel handler.
* moveTo: aMorph – move camera to aMorph.
* myAddMorph: t1 – add morph to scroll target.
* myAddMorph: t1 position: t2 – add morph to scroll target to position t1.
* myScene/myScene – return/set scroll target.
* scrollToShow: t1 – set scroller window.
* scrollbarThickness – return scrollbar thickness.
* scroller/ scroller: – return/set scroller.
* setScrollDeltas – set scroll delta for both scrollbars.
* startAnimation – start animation.
* startColor: t1 – set start color.
* step – provide moving and scaling between CameraMovingMorhs.
* stepTime – return step time.
* stopAnimation – stop animation.
* toPoint/toPoint: – return/set destination point for camera.
* toPointScrollbarX – return position of destination point along X-axis.
* toPointScrollbarY – return position of destination point along Y-axis.
* updateScrollbars – update both scrollbars.
* vLeftoverScrollRange – returns the entire scrolling range minus the currently viewed area.
* vHideScrollbar – hide scrollbar along Y-axis.
* vPageDelta – return delta along Y-axis. Delta is a ratio of the window width to the scroll target width.
* vResizeScrollbar – resize scrollbar along Y-axis.
* vScrollbar/hScrollbar: t1 – return/set scrollbar along Y-axis.
* vScrollbarInterval – return ratio of the window width to the scroll target width in float.
* vScrollbarNeeded – return is scrollbar along Y-axes needed or not.
* vScrollbarShowing – return is showing scrollbar now or not.
* vScrollbarValue – return scrollbar inner value.
* vScrollbarValue: t1 – set scrollbar inner value.
* vSetScrollDelta – set scrollbar value. Scrollbar is the value of a single transition.
* vShowScrollbar – show scrollbar along Y-axis.
* vUpdateScrollbar – update scrollbar along Y-axis.
* zoom – return scale ratio for camera scaling to the size of the CameraMovingMorph.
* zoom: – set scale ratio.
* zooming – check of conditions for camera zooming.
* zoomingAction – camera zooming.
* zoomingStepX/zoomingStepX: – return/set horizontal zooming step.
* zoomingStepY/zoomingStepY: – return/set vertical zooming step.

**ProjectScrollWindow class**

* startStOut/startStOut: bool – answer is presentation active(in demonstration process) or not.

**Scene instance**

* addInPart: aNumber1 focusMorph: aNumber2 – add CameraMovingMorph to part. Part with index aNumber1 from partsCollection and CameraMovingMorph with index aNumber2 from cameraMovingMorhpsCollection.
* cameramovingmorphscollection **–** return cameraMovingMorphsCollection.
* countInPart/countInPart: – return/set counter for the transition between CameraMovingMorphs in a part.
* countMorphs/countMorphs – return/set counter for the transition between CameraMovingMorphs in cameraMovingMorphsCollection.
* countParts/countParts – counter for parts.
* createCameraMovingMorph – create CameraMovingMorph.
* createCameraMovingMorphOnPosition – create CameraMovingMorph at origin of Esse main window.
* createPresentationPart – create part.
* forwardDirection/ forwardDirection: – return/set forward direction for morph Scene.
* heading – return heading.
* initialization – Scene initialization.
* partscollection – return partsCollection.
* prepareForRotating – prepare scene for rotating.
* rotationDegrees/ rotationDegrees – return/set rotation degrees.
* sizeOfScene – return size of Scene.
* startStopAction – stop/start presentation.

Other classes: Arrow, ArrowEnd, DroppedMorph, MoveMorph, MyMorphs are training examples of development in Morphic. MoveMorph is an example of animated morph.

**MoveMorph instance**

* countPoint/countPoint: – return/set counter for collection of points.
* finishPoint/finishPoint: – return/set destination point for move morph.
* initialize – initialize move morph.
* pointcollection – return collection of points.
* startPoint/startPoint – return/set start point for move point.
* stepTime – return time between steps.
* step – provide morph moving.