Drobs: Taking The Drag Out Of 'Drag 'N Drop' Web Pages

Todd Cochrane

Wellington Institute of Technology
Wellington New Zealand

Todd.Cochrane@weltec.ac.nz

Proceedings of the 15th Annual NACCQ, Hamilton New Zealand July, 2002 www.naccq.ac.nz

ABSTRACT

'Drag and drop' interaction on web pages can be achieved using a number of technologies, for example: Flash or Java embedded on the web page, and DHTML. In Drobs ('DRaggable' OBjectS), development of 'drag and drop' interaction is undertaken by providing the maximum "degrees of movement" for a draggable object. Adding constraints on the 'degrees of movement' then refines interaction. This approach is described using the development of 'drag and drop' into a scrolling window on a web page as an example. A version of a Drobs object library in DHTML is presented.

Keywords: 'drag and drop', web page development, user interface, interaction, programming, object libraries.

1. INTRODUCTION

'Drag and drop' interaction has become intrinsic to direct manipulation [7] interfaces. Interaction using "'drag and drop' implements 'transfer operations', as described in [4]. Dragging an item and dropping it into a given area transfers data from one place to another. The application and utility of interactive graphics user interfaces in hypertext systems has been understood prior to the existence of the World Wide Web [6]. The current trend towards 'drag and drop' on a web page reflects the need for a level of interaction expected in other user interfaces.

'Drag and drop' can be achieved on web pages with a number of technologies [1][2][3][5]. These use different mechanisms based on an underlying pattern; first an event is generated indicating that dragging has started, events are generated while dragging proceeds, finally an event is generated when dragging stops, at the drop. A transfer operation implemented by the 'drag and drop' usually happens at the drop. While dragging, position and layout adjustments can also be made to the interface. How and where the dragging happens depends on the technology. In [1], for example, drag and drop is implemented by deriving objects from at least three classes: DragGestureListener, DragSourceListener and DropTargetListener.

2. DROBS

In Drobs ('DRaggable' OBjectS), development of 'drag and drop' interaction is undertaken by providing the maximum 'degrees of movement' for a draggable object. The idea that an aspect of a user interface can be described in terms of degrees of movement of components or data is not necessarily a new idea [8].

In Drobs the goal is to provide maximum degrees of movement within layout constraints. See [9][10][11] for examples of work undertaken on the application of constraints to user interface implementation.

2.1 TYPES OF MOVEMENT WHEN DRAGGING

Dragging is implemented by creating an instance of a draggable object that, from the outset can be dragged anywhere in the interface. The types of movement that you expect in a 2D interface are: movement in horizontal and vertical directions, movement bounded by regions and movement into regions in the interface.

2.2 CONSTRAINTS ON MOVEMENT

Constraints on the movement available during dragging are expressed in vertical and horizontal and region terms, where regions include a drobs' regions. These are checked while an instance of a draggable object (drob) is dragged. At each 'drag' of an object all constraints on its movement are checked before moving the drob to its proposed location, see figure 1.

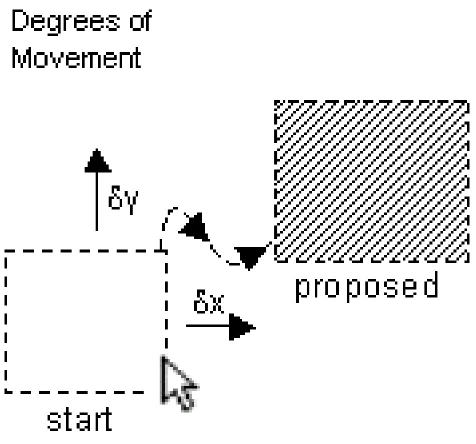


Figure1. Dragging from a start position to a proposed destination. Constraints are expressed in vertical and horizontal terms as well as regions. A constraint may prevent the proposed destination.

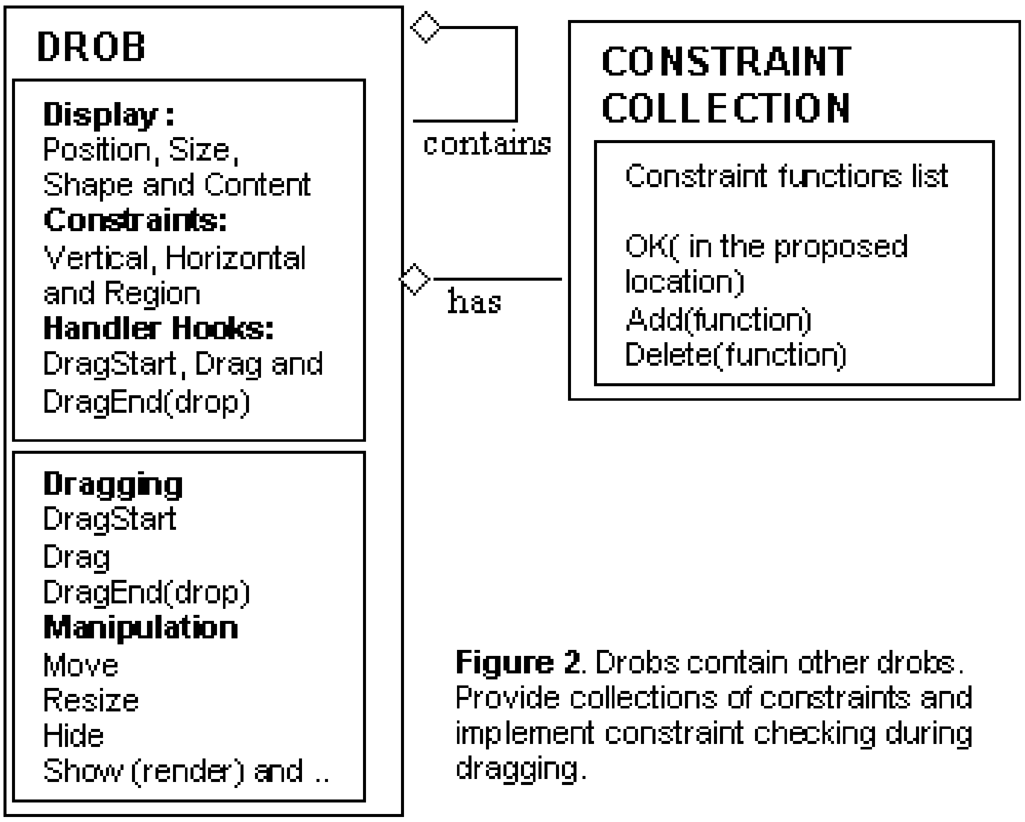

DROB
Display: Position, Size, Shape and Content
Constraints: Vertical, Horizontal and Region
Handler Hooks: DragStart, Drag and DragEnd(drop)
Dragging
DragStart
Drag
DragEnd(drop)
Manipulation
Move
Resize
Hide
Show (render) and ..
contains
has
CONSTRAINT COLLECTION
Constraint functions list
OK(in the proposed location)
Add(function)
Delete(function)

Figure 2. Drobs contain other drobs. Provide collections of constraints and implement constraint checking during dragging.

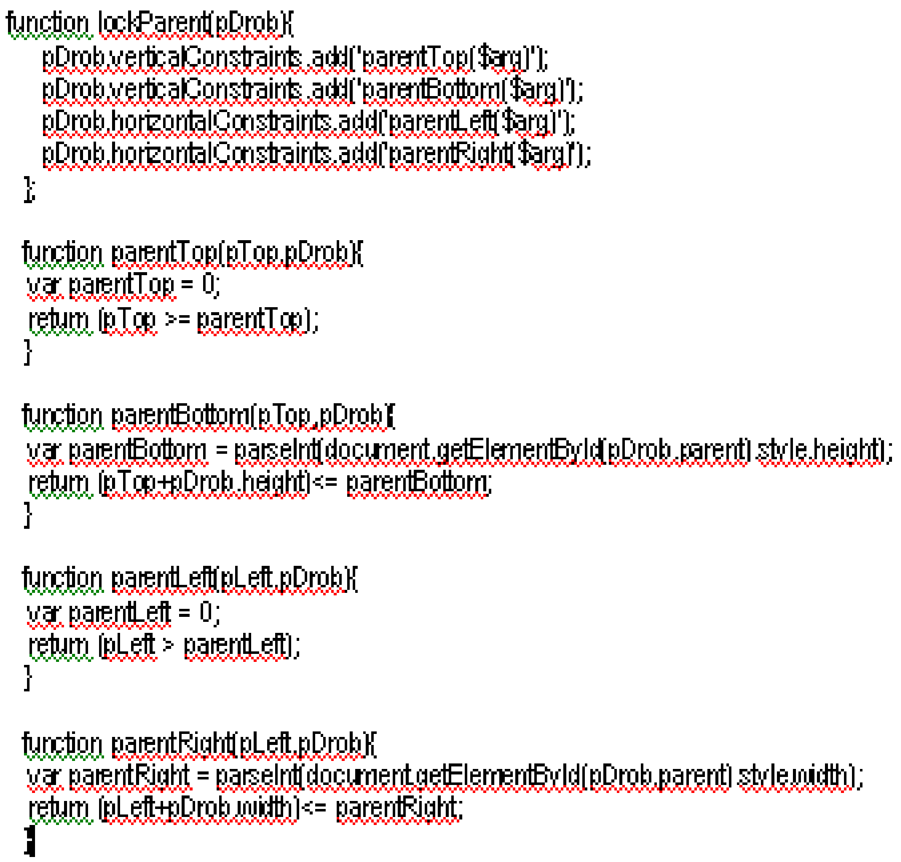
2.3 DROBS OBJECTS

Drobs are implemented based on existing objects that handle event messages from user interaction such as ‘mouse-up’, ‘mouse-down’ and ‘mouse-move’ or ‘drag-start’, ‘dragging’ and ‘drag-end’ (drop). Drob code is added to override in-built movement from dragging if constraints prevent the movement. A drob includes collections of constraints to be checked before movement is allowed. A drob can be created and then constraints added to the drob instance or a specialised drob can be defined as a sub-class of the base drob class. A specialised drob could be defined, for example for dragging that is constrained to the horizontal direction. This particular sub-class of drob is useful when developing scroll bars and sliders.

Figure 2 depicts a base drob class and its relationship with a constraint collection class. Attributes are listed under ‘Display’,’Constraint’ and ‘Handler’ categories. Display attributes are used during rendering and movemenet of the drob. Constraint attributes are checked before redrawing a drob at a proposed position. Handler attributes provide a mechanism for additional behaviours to be executed in addition to the built in drob behaviour. Methods or messages processed by a drob are listed in ‘Dragging’ and ‘Manipulation’ categories. ‘Dragging’ methods implement the standard ‘drag and drop’ pattern. ‘Manipulation’ methods are those that allow the programmer to implement layout adjustments during dragging and potentially cause adjustments as side effects of the evaluation of constraint rules.

3. DRAG AND DROP INTO A SCROLLING WINDOW USING DROBS

As mentioned above, constraints on freedom of movement of a drob can facilitate the implementation of common user interface components such as the thumb of a scroll bar. A scroll bar can be implemented in drobs by creating two drobs. One the containing scroll bar area and the other the thumb. The thumb is constrained to the parent by adding four constraints written in the current DHTML version of drobs as in figure 3.



```
function lockParent(pDrob){
    pDrob.verticalConstraints.add('parentTop($arg)');
    pDrob.verticalConstraints.add('parentBottom($arg)');
    pDrob.horizontalConstraints.add('parentLeft($arg)');
    pDrob.horizontalConstraints.add('parentRight($arg)');
  };

  function parentTop(pTop,pDrob){
   var parentTop = 0;
   return (pTop >= parentTop);
  }

  function parentBottom(pTop,pDrob){
   var parentBottom = parseInt(document.getElementById(pDrob.parent).style.height);
   return (pTop+pDrob.height)<= parentBottom;
  }

  function parentLeft(pLeft,pDrob){
   var parentLeft = 0;
   return (pLeft > parentLeft);
  }

  function parentRight(pLeft,pDrob){
   var parentRight = parseInt(document.getElementById(pDrob.parent).style.width);
   return (pLeft+pDrob.width)<= parentRight;
  }
```

Figure 3. Four constraint functions

Dragging and dropping into a scrolling window can be implemented by adding region constraints in the definition of a sub-class of drobs that can be accepted by the scrolling window. Additional behaviour needs to be added to the sub-class so that the behavior of instances changes when they are dropped into the scrolling window.

4. A DROB OBJECT LIBRARY IN DHTML

The implementation of a drob object library in DHTML starts with the development of a draggable object based on dragging of instances the HTML DIV element objects. The DIV element provides a mechanism for hooking into the event handling provided with the DOM. A base drob class is defined using JavaScript object prototyping. Each drob renders itself onto the web page as a DIV element. A top level DIV element is provided as a parent into which all other drobs render themselves either at the top level as immediate children or within the DIV element of graphic descendants.

4.1 DRAGGING

Dragging is implemented in a two-stage process. The drob's 'Start drag' handler handles a 'mouse down' event on a DIV tag rendered by a drob. At the 'Start drag' the drob also handles the document's move events, moving its element as the mouse moves over the document. As the mouse moves the drob checks the location of the mouse against its constraints.

4.2 DROP

On mouse up the current drob releases the document's mouse move handler and executes its drop behaviour.

5. SUMMARY

An approach to developing drag and drop on web pages using maximum degrees of movement is a useful and may be easier than more complex approaches.

REFERENCES

[1] **"Using Drag and Drop in JDK 1.2 Beta 4". Accessed May 10, 2002.** <http://developer.java.sun.com/developer/qow/archive/1/index.html>

[2] **"Flash Bible - Actionscript - Drag and Drop". Accessed May 10, 2002.** <http://www.flashbible.com/Members/Drag/Drag.htm>

[3] **"Drag & Drop in Explorer 4 - webreference.com". Accessed May 10, 2002.** <http://www.webreference.com/dhtml/column7/index.html>

[4] **"Transfer Operations", Accessed May 10, 2002.** <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwue/html/ch06f.asp>

[5] **"DoDragDrop Method", Accessed May 10, 2002.** <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpref/html/frlrfsystemwindowsformscontrolclassdodragdroptopic.asp>

[6] **Weiland, W. J., Shneiderman, B. (1989)** Interactive graphics interfaces in hypertext systems Proc. 28th Annual ACM DC Technical Symposium (Aug 1989), 23-28.

[7] **Shneiderman, B., (August 1983).** Direct manipulation: A step beyond programming languages, IEEE Computer 16, 8, 57-69.

[8] **Roth*,S. F., Chuah*,M.C., Kerpedjiev*,S., Kolojejchick*,J., Lucas,P., (1997).** "Towards an Information Visualization Workspace: Combining Multiple Means of Expression", Human Computer Interaction Journal Volume 12, Number 1 & 2, 1997 131-185. Accessed May 10, 2002. <http://www.maya.com/Visage/base/papers/HCI-journal.html>

[9] **Borning,A. ,(1986).** "Defining Constraints Graphically - Drawing and Animation Systems" Proceedings of ACM CHI'86 Conference on Human Factors in Computing Systems p.137-143

[10] **Karsenty,S. , Landay,J. A.,Weikart,C. (1992)**. "Inferring Graphical Constraints with Rockit" Graphics — Design and Techniques Proceedings of the HCI'92 Conference on People and Computers VII p.137-153

[11] **Duisberg,R.A., (1986).** "Animated Graphical Interfaces Using Temporal Constraints" Drawing and Animation Systems Proceedings of ACM CHI'86 Conference on Human Factors in Computing Systems p.131-136