JAWAA: Easy Web-Based Animation from CS 0 to Advanced CS Courses

Ayonike Akingbade[‡], Thomas Finley[*], Diana Jackson[†], Pretesh Patel[*], and Susan H. Rodger[*†]
Department of Computer Science
Duke University
Durham, NC 27708-0129
rodger@cs.duke.edu

Abstract

We present JAWAA 2.0, a scripting language for creating animations easily over the web. JAWAA includes primitives, easy creation of data structures and operations on these structures, and an editor for easy creation of complex objects. We show how to use JAWAA in a range of computer science courses including CS 0, CS 1, CS 2 and advanced courses. Instructors can quickly build animations for demos in lecture, and students can enhance their programming projects with an animation.

Categories and Subject Descriptors K.3.2 [**Computers & Education**]: Computer & Information Science Education — *Computer Science Education*; D.1.7 [**Programming Techniques**]: Visual Programming; I.3.3 [**Computer Graphics**]: Picture/Image Generation — *Viewing Algorithms*

General Terms Algorithms, Human Factors, Languages

Keywords JAWAA, Animation

1 Introduction

Animations of concepts are fun to watch, but how helpful are they in learning? A recent survey [3] and an analysis of twenty-one experiments [1] suggest that learner involvement such as student-built animations improves learning. However, building animations from scratch can be time consuming

We have developed the tool JAWAA [4] (Java And Web based Algorithm Animation) for easy creation of data-structures over the web. Students in CS 1 or CS 2 can output simple JAWAA script commands from their program, put them on a web page and have an instant animation of one or more of their data structures without installing software. Recently we have updated JAWAA with additional data structures, new features and developed a JAWAA Editor. The JAWAA Editor allows for easier layout of complex objects, making JAWAA now accessible for novices in CS 0 and easier to use for more advanced CS courses.

SIGCSE '03, February 19-23, Reno, Nevada, USA.

2 Related Work

There are many systems for generating animations from Xtango [8] in which animations are built from scratch to Samba [9], a scripting language in which students output commands from their program that can then be used to generate an animation. Other animation tools that use scripting include GAIGS [2] and AnimalScript [7]. JAWAA is most like Samba; however, in addition it has scripting commands for easy creation of data structures, runs over the web, and more recently has a JAWAA Editor for easy creation of complex objects.

3 JAWAA 2.0

JAWAA 2.0 is a scripting language for creating animations and running them easily over the web. A student can start with a program written in any programming language, output JAWAA commands to a *.anim* file that is in a web accessible directory, create a simple *.html* page with the same name, load the web page and run the animation. JAWAA interprets the commands one by one.

There are three types of JAWAA commands: primitive objects, general action commands and data structures. The data structures include special action commands for manipulating them.

3.1 Primitive Objects

Figure 1 shows the six types of primitive objects in JAWAA, which are circle, rectangle, text, line, polygon and oval. Each object has an identifier name, location, colors and size attributes. Shown below is an example of a rectangle command. The identifier is *box1*, the upper left corner is at (10,20), the width is 100, the height is 120, the outline is black and the background or main color is red.

[*]The work of these authors is supported by the National Science Foundation through grant NSF DUE-9752583.

[†]The work of these authors is supported in part by the Computing Research Association's Distributed Mentor Project which is funded in part by the National Science Foundation Grant EIA-0124641.

[‡]The work of this author is supported by Eli Lilly.

```
rectangle box1 10 20 100 120 black red
```

Figure 1: JAWAA Primitives

3.2 General Action Commands

JAWAA action commands control the number of simultaneous commands, allow movement and changes to an object, and the grouping of objects.

Several JAWAA commands can appear to execute simultaneously. Usually, as each JAWAA command is executed, the resulting action is shown in the window. By placing commands inside a *begin-end* block, the resulting actions for all the commands in the block are shown at the same time.

There are several ways to modify an object. With the *changeParam* command an object can change several features including its color, size, location, and associated data. Additional commands include moving an object, scaling the size of an object and removing an object.

3.3 Data Structure: Array

With JAWAA one can easily create an array with single or multiple data per cell and illustrate the movement through the array. Arrays can be oriented either horizontally or vertically.

The example code below specifies an array, swaps two elements in the array, changes the color of a cell and moves a cell. Figure 2 shows three snapshots of the same array as it changes. The leftmost array corresponds to the first line of the code below (shown on two lines). In the first line the array is given the name *run*, is placed at position (10,100), has three cells each with two data items, the outline and text are black and the background color is yellow. The middle array corresponds to the next two lines of code. The data in cells 1 and 2 are swapped and cell 1's background color is changed to white. The rightmost array corresponds to the last two lines of code. The second item in cell 0 is changed to 8 and cell 1 is moved to another location.

```
array run 10 100 3.2 owen 4 susan 12
    jeff 5 vert black yellow black
changeParam run[1] swap run[2]
changeParam run[1] bkgrd white
changeParam run[0].1 text 8
changeParam run[1] x 320
```

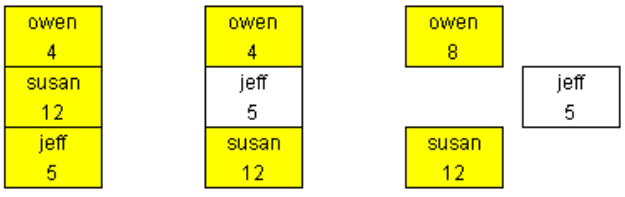

Figure 2: JAWAA Arrays

3.4 Data Structures: Stack and Queue

Stacks and queues can easily be implemented in JAWAA by specifying the initial stack or queue and then using pop or push for a stack and enqueue and dequeue for a queue. Figure 3 shows two snapshots of a queue and two snapshots of a stack. The left stack is the initial stack created by the stack command with three initial elements. The name of the stack is *st1*, the stack is positioned at (90,99), the *3* indicates there are three initial elements on the stack which follow the 3, and the color of the outline and the text are black. The right stack shows the stack after the next three lines have been executed. The left queue is the initial queue created by the queue command with two initial elements. The name of the queue is *q1*, the queue is positioned at (280,100), the *2* indicates there are two initial elements in the queue, and the color of the outline and text are black. The right queue shows the queue after the next three lines have been executed.

```
stack st1 90 99 3 ho markus erich black black
pop st1
push st1 jason
push st1 davis
queue q1 280 100 2 frog man black black
dequeue q1
enqueue q1 spider
enqueue q1 100.50
```

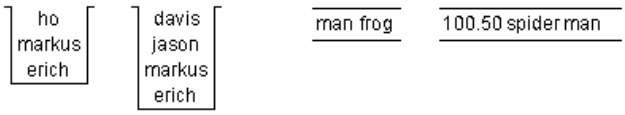

Figure 3: JAWAA Stack and Queue

3.5 Data Structures: Graph and Tree

Graphs and trees can be created by creating nodes and edges using the *node* and *connectNodes* commands. Edges can be specified to have 0-2 arrowheads allowing for directed or undirected graphs. Markers can be created using a marker command that move through the graph easily by specifying the next node to move to.

3.6 Data Structures: Linked List and Pointer

Linked lists are easy to create and manipulate. A complete linked list can be created in one line. Nodes can be created either with the *list* command or by creating separate nodes with *node* and then using the *list* command to link them together. Figure 4 shows the addition of an element to a linked list. Line 1 below (shown on two lines) shows the creation of a list with four elements. The name of the list is *names*, it is positioned at (50,110), the width and height of new nodes are 26 and 20, N is the name of the header pointer (shown in the figure), the *4* indicates there are 4 nodes in the list, the next 8 fields are the names and values of those nodes, *HORZ* indicates horizontal display, the outline is black, the background is yellow, the text color is black and the shape of the node is a rectangle. Line 2 (shown on two lines) shows the addition of a node. The next three lines reset pointers to include the node in the list by adding a pointer from node n5 to n3, deleting the pointer from n2 to n3 and adding a pointer from n2 to n5.

```
list names 50 110 26 20 N 4 n1 jo n2 flo n3 ed
    n4 mo HORZ black yellow black rect
node n5 160 70 26 20 1 gib black yellow
    black rect
connectNodes e2 n5 n3 black 1
delete n2_arrow
connectNodes e1 n2 n5 black 1
```

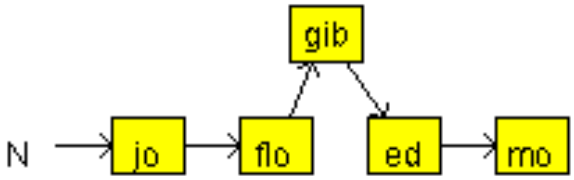

Figure 4: JAWAA Link List

4 JAWAA Editor

The JAWAA Editor allows one to create animations graphically by laying out primitive objects and modifying them across time. The editor then can export the animations created to a .anim file, which can be viewed by JAWAA. Figure 5 shows the JAWAA Editor currently in selector (or modify) mode. The middle balloon has been selected as indicated by the surrounding dots. The interface of the JAWAA Editor is similar to existing vector graphics applications.

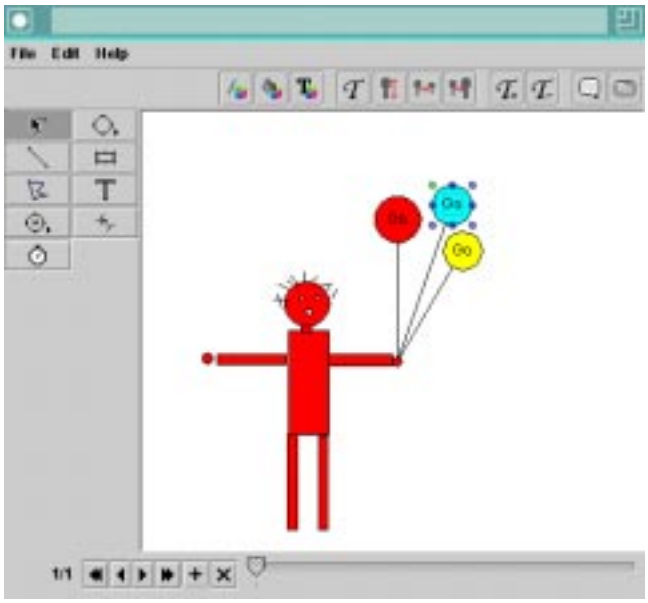

Figure 5: JAWAA Editor

The JAWAA Editor is useful for four reasons. First, its use requires no knowledge of the JAWAA language. Second, the WYSIWYG editing allows one to correctly build complex objects involving lots of primitive objects. This avoids the guesswork involved in getting objects to look right with textual commands. Third, one can define the animations by changing the state of objects across time. Upon export to a .anim file the editor will generate the appropriate commands to morph objects across time. Fourth, it's output can be combined with JAWAA output from a student's program. The editor does the layout of the complex objects and the student's program manipulates the objects.

5 JAWAA's Use in Courses

We show how we use JAWAA (which includes the JAWAA Editor) in a range of computer science courses at Duke University.

The instructor can use JAWAA in many different ways including creating an example animation of a concept to show for lecture, creating objects for a programming assignment that students can then move with output from their programs, and letting students use JAWAA to animate their programs.

5.1 JAWAA in CS 0

In CS 0 students do not have any background in computer science and are not planning on majoring in computer science. CS 0 courses typically are an overview of computer science and some teach programming using special languages. The paper [6] describes a non-majors freshmen seminar entitled CPS 49S Animation and Virtual Worlds that includes one section on using JAWAA for one unit of the course. This material could easily be used in CS 0. At the time, the JAWAA Editor was not complete and students did not know any programming language, so they used JAWAA by typing JAWAA commands into a file. Now students can use the JAWAA Editor to build their animation, or some combination of the JAWAA Editor and finish their animation by adding in additional JAWAA commands. Assignments given in CPS 49S include a simple assignment that uses all the primitives and action commands; a traffic assignment with cars, an intersection, and a stoplight; a sorting assignment to move bars around and sort them by their height, and projects of their choice.

5.2 JAWAA in CS 1

JAWAA can be used in CS 1 for illustrating concepts and by students in their programming assignments.

JAWAA animations for arrays can easily be constructed using a few lines of JAWAA code that illustrate the operations on an array. Stepping through an array can be illustrated by changing the color of a cell upon entering it and changing it's color back upon leaving it. Thus, searching for an element, modifying an element, an insertion, and a deletion are all easy to show. We mention two concrete examples. First, an animation illustrating which sorting algorithm takes longer (insertion sort or selection sort) can be shown by showing two arrays and one step in each algorithm at the same time. Second, consider the problem of finding the number of unique words in an array. One algorithm is to sort the words first, and then scan them counting. Another algorithm is to enter the words into an array only if they are not a duplicate. Again two arrays could be shown side by side showing one step in each algorithm at the same time.

JAWAA animations can be easily incorporated into programming assignments; however, we recommend that JAWAA commands are automatically generated (especially at the beginning of the semester) and hide the JAWAA code

from the students. In CS 1 students are learning a programming language and struggling with the syntax of that language. They do not need to be concerned with the syntax of an additional language at the same time.

We describe an example of three CS 1 assignments that use JAWAA in which students manipulate hot air balloons and eventually race them. All the JAWAA code is generated automatically into a separate file. Students call functions that generate the JAWAA output. In the first assignment students create a hot air balloon, make it go up, cruise and go back down. In the second assignment students create multiple balloons that move in specified movements. In the third assignment students know about loops, randomness and functions that return values. Here they create a hot air balloon race with three balloons, each randomly moving, and determine the winner. In the past our balloons were just circles or very simple pictures. Now with the use of the JAWAA Editor it is easy to make more realistic looking balloons and background. Figure 6 shows a balloon race for a CS 1 project. The sign, the balloons and the trees were all created with the JAWAA Editor in five minutes and can be used by the students in their programs. The students can add the output from their program to manipulate and race the balloons.

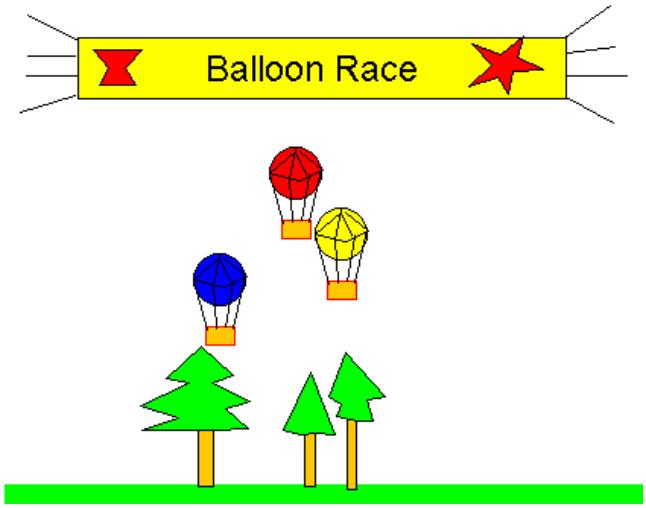

Figure 6: JAWAA CS 1 Balloon Race

5.3 JAWAA in CS 2

JAWAA works well with CS 2 as it has many data structures built in that can easily be animated either by the instructor for demos in class or by the students in their programming projects. We have used JAWAA in CPS 100 at Duke University in these ways and give examples of how the instructor can prepare animations for lecture and how students can animate projects.

The instructor can easily create simple animations to show in lecture. We give three examples. All three examples were created with the JAWAA Editor and each took less than ten minutes to create. First, Figure 7 shows an example of a recursive function executing. The line currently executing is in red (the first cout statement) and the rest of the lines are in black. With each recursive call, a new instance of Mystery with its current value of n appears on the stack. When a function finishes executing, the instance disappears off the stack. Second, Figure 8 shows an example of a max heap being created. All the nodes have the heap property except for the root. A downheap operation on the root is being performed. The 4 is being moved to the 8's position to swap with it. The array is also shown and swapping of elements in the array and tree are shown at the same time. The third example (not shown), is an animation of the difference between static and dynamic memory in a programming language. The animation shows 8 lines of code representing declarations and assignments on the left side of the window, showing one line of code at a time. For each line, a picture of the corresponding memory is shown on the right side of the window. Copies can be shown by a copy floating down to the assigned memory location. Memory that is not put back in the heap is left floating.

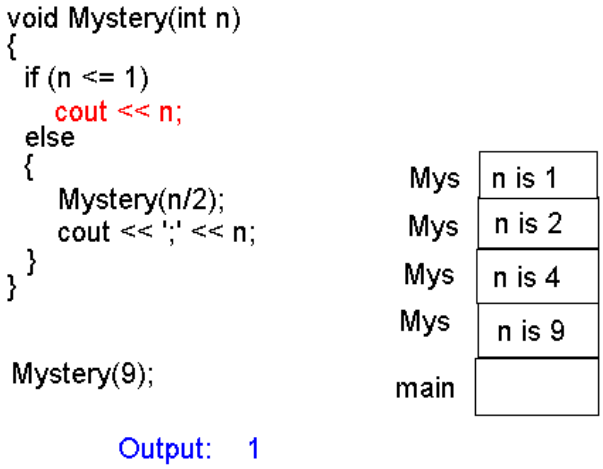

Figure 7: JAWAA CS 2 Recursion

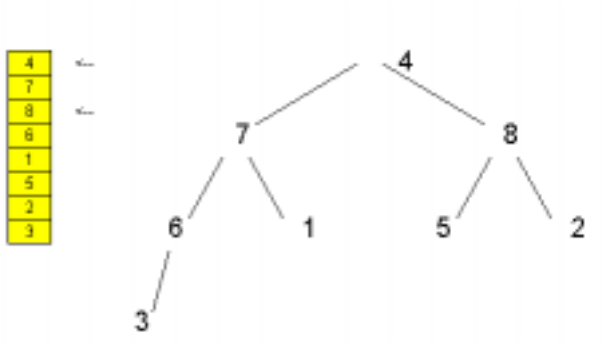

Figure 8: JAWAA CS 2 Heap

Students can use JAWAA for projects. Figure 9 shows a student project for depth first search. In this project the student creates a graph, then using a *marker* command, traverses the graph by moving the marker along the edges. In addition a stack is used, and the nodes in the graph are listed in preorder and postorder. Shown in the animation is the graph (coloring nodes as they are processed), the marker moving along an edge (from node 8 to 7), a stack showing the elements still to be output for postorder, and the preorder and postorder listings (shown as arrays with white outline.)

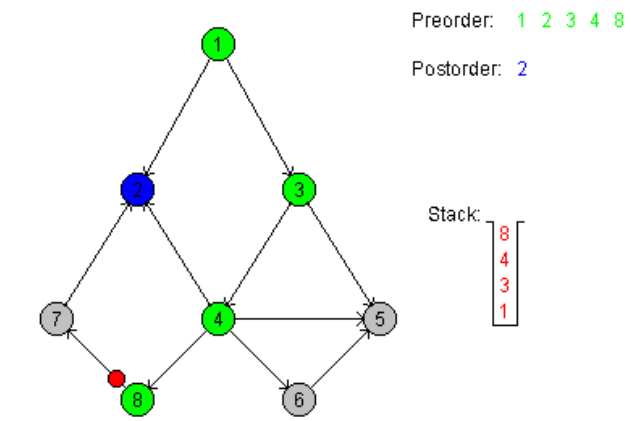

Figure 9: JAWAA CS 2 Depth First Search

Another project example is the Word Ladder problem. A word ladder is a list of words in which consecutive words differ by one letter. Students keep track of the words in an array and use a queue to process the words. An animation of the student's array and queue are easy to display.

5.4 JAWAA in Other CS Courses

JAWAA can be used easily in other courses as it can be output from a program written in any language. In CPS 140 at Duke University, an automata and formal languages course, students study a simple programming language and write an interpretor for the language. One project was a language with robots and barriers. In the last phase, JAWAA was added to animate the running of a robot program. Figure 10 shows a snapshot of a student's animation with 1 robot and 11 barriers.

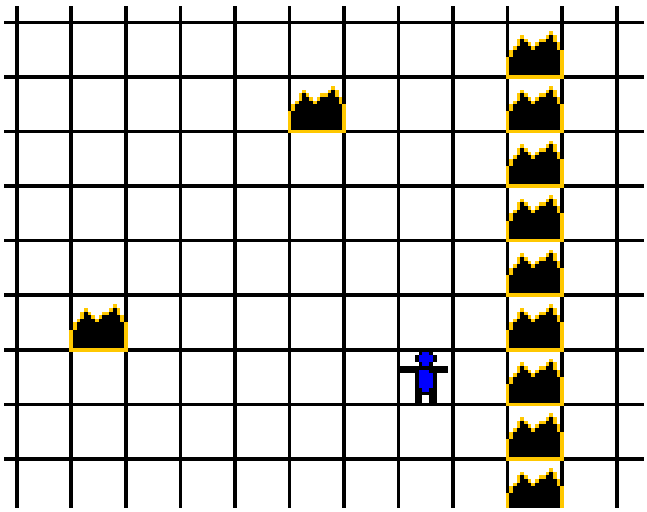

Figure 10: JAWAA Robot

6 Evaluation of JAWAA

An evaluation of JAWAA was given in the freshmen seminar course CPS 49S in the spring of 2001 and 50% of the students reported that they liked this unit. This course is for novices and the JAWAA Editor did not exist at the time. Students found it tedious to calculate the placement of objects by trial and error. We will reevaluate the use of JAWAA in this course in the fall of 2002 using the JAWAA Editor. In other computer science courses CPS 6, CPS 100 and CPS 140, students have given positive feedback on the demos we've given in class and especially the demoing of their projects to the whole class.

7 Future Work and Obtaining Software

We plan to add additional data structures to JAWAA, and implement automatic layouts for graphs and trees. We also plan to modify JAWAA and the JAWAA Editor based on its use in several courses. JAWAA is available free from [5].

References

[1] Hundhausen, C. D., Douglas, S. A., and Stasko, J. T. A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing* (2002), 259–290.

[2] Naps, T. L., and Bressler, E. A multi-windowed environment for simultaneous visualization of related algorithms on the world wide web. *Twenty-ninth SIGCSE Technical Symposium on Computer Science Education* (1998), 277–281.

[3] Naps, T. L., Rossling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., Korhonen, A., Malmi, L., McNally, M., Rodger, S., and Velazquez-Iturbide, J. A. Exploring the role of visualization and engagement in computer science education, report of the working group on "improving the educational impact of algorithm visualization". *The Seventh Annual Conference on Innovation and Technology in Computer Science Education* (2002), (to appear).

[4] Pierson, W., and Rodger, S. H. Web-based animation of data structures using jawaa. *Twenty-ninth SIGCSE Technical Symposium on Computer Science Education* (1998), 267–271.

[5] Rodger, S. Visual and interactive tools web site, www.cs.duke.edu/∼rodger/tools/

[6] Rodger, S. H. Introducing computer science through animation and virtual worlds. *Thirty-first SIGCSE Technical Symposium on Computer Science Education* (2002), 186–190.

[7] Rossling, G., and Freisleben, B. Animalscript: An extensible scripting language for algorithm animation. *Thirty-second SIGCSE Technical Symposium on Computer Science Education* (2001), 70–74.

[8] Stasko, J. Tango: A framework and system for algorithm animation. *IEEE Computer* (1990), 27–39.

[9] Stasko, J. T. Using student-built algorithm animations as learning aids. *Twenty-eighth SIGCSE Technical Symposium on Computer Science Education* (1997), 25–29.