Animator Commands
(modified Animator handout from John Stasko's Xtango)

Animations developed with this system will be carried out in a real-valued coordinate system that originally runs from 0.0 to 1.0 from left-to-right and from 0.0 to 1.0 from bottom-to-top. Note, however, that the coordinate system is infinite in all directions. You will create and place graphical objects within the coordinate system, and then move them, change their color, visibility, fill, etc., in order to depict the operations and actions of a computer algorithm.

Each command begins with a unique one word string. Make sure that you spell the strings correctly. Each graphical object that you create should be designated by a unique integer id. You will need to use that id in subsequent commands that move, color, alter, etc., the object. In essence, the id is a handle onto the object. Most of the commands and their parameters should be self-explanatory. Arguments named `steps` and `centered` are of integer type. Arguments named `xpos, ypos, xsize, ysize, radius, lx, by, rx, ty` are real or floating point numbers. Make sure that they include a decimal point (typing 0 instead of 0.0, for example, will cause an error). The argument `fillval` should be one of the following strings: `outline`, `half` or `solid`. The argument `widthval` should be one of the following strings: `thin`, `medthick`, or `thick`. The parameter `colorval` can be any basic color string name: white, black, red, blue, yellow, orange, brown, green, and purple (one word, no blanks allowed). Note that the `color` command only supports a subset of all these possible colors.

Note that any line with a per cent character (%) in column 1 is interpreted as an instructional comment—no command is carried out.

The animator will print warning messages if you try to do erroneous actions like reuse an integer ID, try to delete or move an invalid ID, and so on. You can turn these warnings off by using the -O command line flag.

- **bg** - change the background color

  format: `bg colorval`

  Change the background to the given color. The default starter is white. Replace colorval by a color such as: red, green, blue, yellow, black, orange.

  example: `bg green`

  The background color is changed to green.

- **coords** - declare the window size

  format: `coords lx by rx ty`

  Define the x values in the window to go from lx (left x value) to rx (right x value). Define y values in the window to go from by (bottom y) to ty (top y).

  example: `coords 0.0 0.0 100.0 100.0`

  The window is defined with x values from 0.0 to 100.0, and y values from 0.0 to 100.0.

  You can use repeated applications of this command to pan or zoom the animation view.

- **delay** - slowing down the animation

  format: `delay steps`

  Delay for "steps" units of time. So "steps" should be replaced by an integer.

  example: `delay 6`

  Delay for 6 units.

- **line**

  Create a line with one endpoint at the given position and of the given size.

  format: `line id xpos ypos xsize ysize colorval widthval`

  Write the word line. then "id" is replace with a unique object number (integer) (xpos,ypos) is the x-y value of one endpoint of the line (should be real numbers), xsize is the x distance between the endpoints (real number), ysize is the y distance between the points (real number), colorval is a color, and widthval is either thin, medthick or thick.

  example: `line 10 15.0 13.0 10.0 5.0 green thick`

  This draws a thick green line with endpoints (15.0,13.0) and (25.0,18.0). Its unique id is 10.

- **rectangle**

  format: `rectangle id xpos ypos xsize ysize colorval fillval`

  Create a rectangle with lower left corner at the given position and of the given size (size must be positive).

  Write the word rectangle. then "id" is replaced with a unique object number (integer), (xpos,ypos) is the x-y value of one endpoint of the line (should be real numbers) xsize is the x distance between the endpoints (real number), ysize is the y distance between the points (real number), colorval is a color, and fillval is either outline, half or solid.

  example: `rectangle 2 8.5 30.0 4.5 20.5 orange solid`

  This draws a solid orange rectangle with lower left corner (8.5,30.0), width 4.5, and height 20.5. Its unique id is 2.

- **circle**

  Create a circle centered at the given position

  format: `circle id xpos ypos radius colorval fillval`

  Write the word circle, then "id" is replaced with a unique object number (integer), (xpos,ypos) is the x-y value of the center of the circle (should be real numbers) radius is the radius of the circle (real number), colorval is a color, and fillval is either outline, half or solid.

  example: `circle 5 9.0 34.0 4.0 yellow half`

  This draws a yellow circle half-filled in with center at (9.0,34.0), and radius 4.0. Its unique id is 5.

- **triangle**

  Create a triangle whose three vertices are located at the given three coordinates. Note that triangles are moved (for move, jump, and exchange commands) relative to the center of their bounding box.

format: `triangle id v1x v1y v2x v2y v3x v3y colorval fillval`

Write the word triangle, then "id" is replaced with a unique object number (integer), (v1x,v1y), (v2x,v2y), and (v3x,v3y) are the three points of the triangle, colorval is a color, and fillval is either outline, half or solid.

example: `triangle 21 30.0 30.0 70.0 30.0 30.0 90.0 blue outline`

This creates a blue outline of a triangle with endpoints (30.0,30.0), (70.0,30.0), and (30.0,90.0). Its id is 21.

- **text**

  Create text with lower left corner at the given position if `centered` is 0. If `centered` is 1, the position arguments denote the place where the center of the text is put. The text string is allowed to have blank spaces included in it but you should make sure it includes at least one non-blank character.

  format: `text id xpos ypos centered colorval string`

  Write the word text, then "id" is replaced with a unique object number (integer), (xpos,ypos) is the x-y value of the position of the text (should be real numbers), centered is 0 if the x-y position is the lower left corner of the text and 1 if the x-y position is the center of the text, colorval is a color, and string is the text to be printed (can include blanks).

  example: `text 8 46.0 75.0 1 red Surfing the Internet is cool`

  This prints "Surfing the Internet is cool" in red, centered at (46.0,75.0). Its unique id is 8.

- **bigtext**

  Similar to text, but bigger font.

  format: `bigtext id xpos ypos centered colorval string`

- **move** - moving an object to a specific position

  format: `move id xpos ypos`

  Smoothly move, via a sequence of intermediate steps, the object with the given id to the specified position (xpos, ypos). The id is an integer and xpos and ypos are real numbers.

  example: `move 8 89.0 32.0`

  This moves object 8 to (89.0,32.0).

- **moverelative** - moving an item via relative distance

  Smoothly move, via a sequence of intermediate steps, the object with the given id by the given relative distance.

  format: `moverelative id xdelta ydelta`

  Write the word moverelative, then "id" is replaced with a unique object number (integer), if (x,y) is the current position before the move, then (x+xdelta,y+ydelta) is the x-y value of the position of the object after the move where xdelta and ydelta should be real numbers.

  example: `moverelative 8 5.0 31.0`

  This moves object 8 5.0 units in the right direction, and 31.0 units in the y direction.

- **moveto** - move an item to the position of another item

  format: `moveto id id`

  Smoothly move, via a sequence of intermediate steps, the object with the first id to the current position of the object with the second id. Both id's are integers.

  example: `moveto 6 10`

  This moves object 6 to object the same position as object 10.

- **jump** - move instantly to specific position

  format: `jump id xpos ypos`

  Move the object with the given id to the designated position in a one frame jump. Here, "id" is replaced with a unique object number (integer), and (xpos,ypos) is the position of the object after the move (both are real numbers).

  example: `jump 7 50.0 34.9`

  This moves object 7 instantly to position (50.0,34.9).

- **jumprelative** - move instantly a relative distance

  format: `jumprelative id xdelta ydelta`

  Move the object with the given id by the provided relative distance in one jump where id is an integer and xdelta and ydelta are real numbers.

  example: `jumprelative 10 5.4 9.5`

  This moves object 10 instantly 5.4 units in the x direction and 9.5 units in the y direction.

- **jumpto** move instantly to the position of another object

  format: `jumpto id id`

  Move the object with the given id to the current position of the object with the second id in a one frame jump. Both id's are integers.

  example: `jumpto 8 10`

  This moves object 8 instantly to the same position as object 10.

- **color** - change the color of an object

  format: `color id colorval`

  Change the color of the object with the given id to the specified color value. Only the colors white, black, red, green, blue, orange, maroon, and yellow are valid for this command.

  example: `color 6 green`

  This changes the object 6 to green.

- **delete** - permanently remove an object

  format: `delete id`

  Permanently remove the object with the given id (unique integer of an object) from the display, and remove any association of this id number with the object.

  example: `delete 8`

  This removes object 8. You can't refer to it anymore.

- **fill** - change the fill value of an object

  format: `fill id fillval`

  Change the object with the given id to the designated fill value. The id is the unique integer associated with an object. Valid fill values are: outline, half or solid. This has no effect on lines and text.

  example: `fill 6 outline`

  This changes the object 6 to an outline.

- **vis** - toggle visibility: flash

  format: `vis id`

  Toggle the visibility of the object with the given id (id is the unique integer associated with the object). If the object is visible, executing this command makes the object invisible. If the object is invisible, executing this command makes the object visible. Repeatedly executing this command makes the object appear to flash.

  example: `vis 6`

  This makes object 6 invisible if it was currently visible, and vice versa.

- **lower** - put object behind

  format: `lower id`

  Push the object with the given id backward to the viewing plane farthest from the viewer.

  example: `lower 8`

  This pushes object 8 behind other objects at the same position.

- **raise** - object to front

  format: `raise id`

  Raise the object with the given id forward to the viewing plane closest to the viewer.

  example: `raise 8`

  This raises object 8 in front of objects with the same position.

- **exchangepos** - smoothly exchange two objects

  format: `exchangepos id id`

  Make the two objects specified by the given ids smoothly exchange positions.

  example: `exchangepos 8 10`

  Smoothly exchange objects 8 and 10.

- **switchpos** - instantly exchange two objects

  format: `switchpos id id`

  Make the two objects specified by the given ids exchange positions in one instantaneous jump.

  example: `switchpos 9 34`

  Instantly exchange objects 9 and 34.

- **swapid** - swap id numbers

  format: `swapid id id`

  Exchange the ids used to designate the two given objects. This doesn't show anything on the screen.

  example: `swapid 9 23`

  If 9 is a rectangle and 23 is a circle, then after this command, 9 refers to the circle and 23 refers to the rectangle.