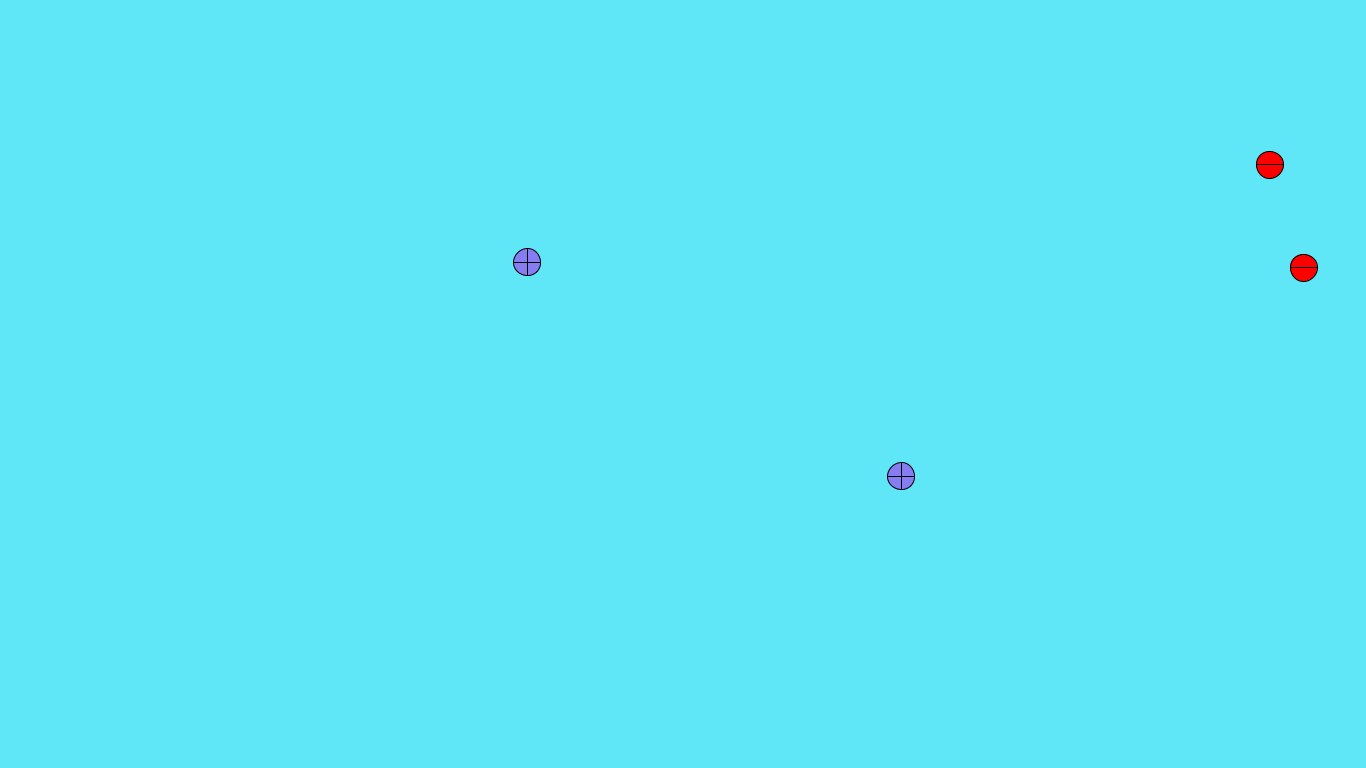
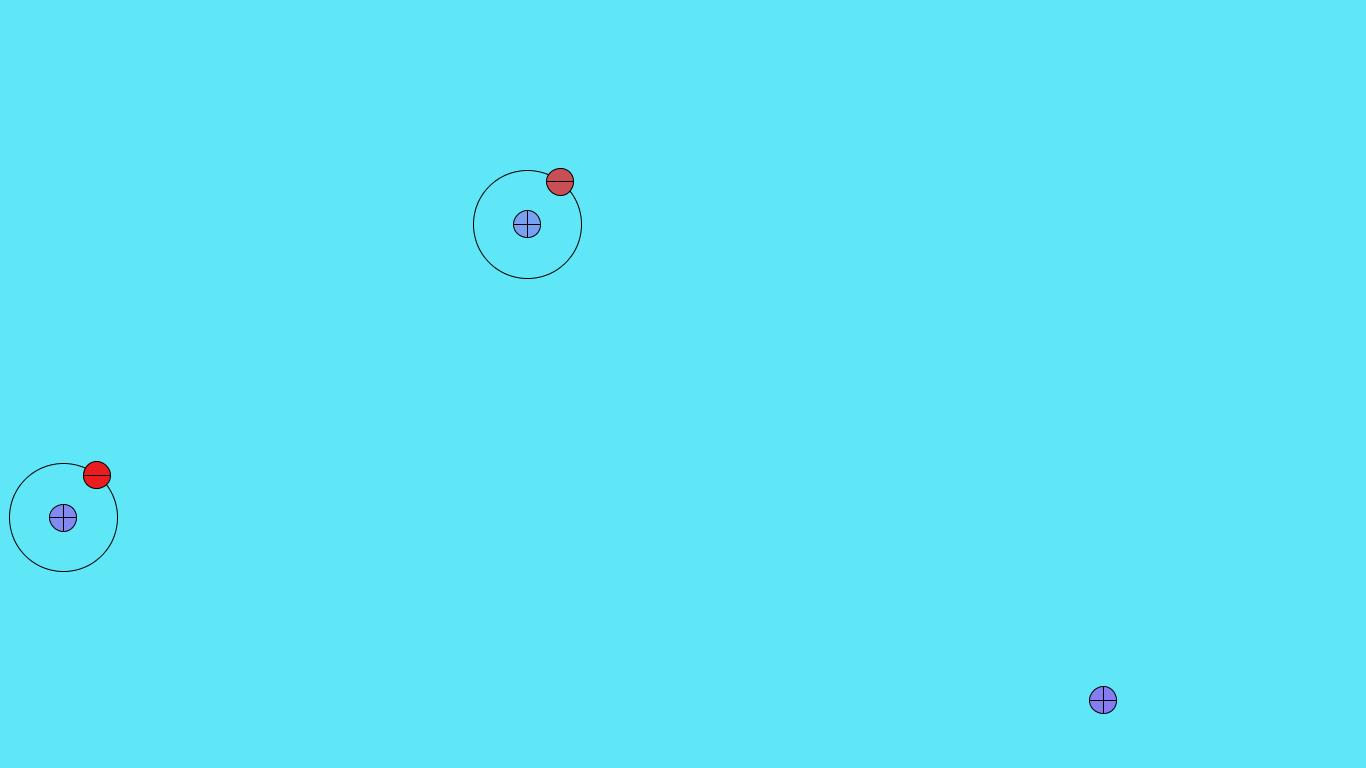
The Electrostatic Emulator

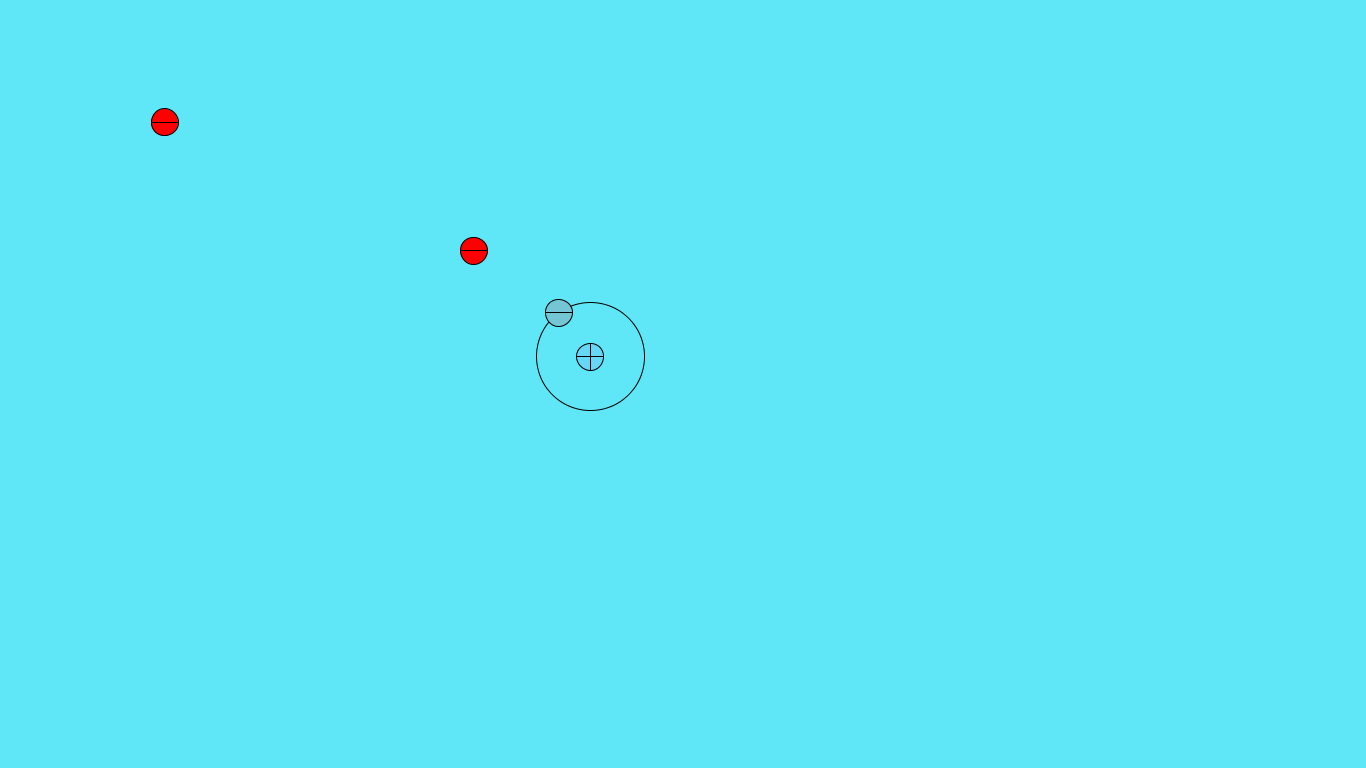
This program offers a simulation of the motion of spherically symmetrical, charged particles under the dynamics of electrostatic forces, by Coulomb’s Law; the motion is abridged to translational, and the dynamics thanks to the movement of the charges, magnetism, is foregone. As the primary interest lies in the patterns of motion under such circumstances, the situations of possible collisions between charges, as the particles are of finite radii, are rendered scenarios with plausible assumptions as to minimize the effect of such complex occasions on the motion of charges under standard conditions. The motion is confined two two dimensional, which, accordingly, restricts the dynamics thus.

The general process of the algorithm is the calculation of accelerations of all particles each frame and the increment of the velocities and positions of such particles accordingly. All such quantities, along with the mass and charge, are stored as lists with the same length as global fields. The computeAcceleration() method computes the acceleration of two particles based on their electrostatic force and store them. Performing such calculation between all charges yields the dynamic state of the system. Succeedingly, the velocities and accelerations are incremented in accord with the duration given by the frame rate. It is observed that a frame rate greater than 50 readily constitutes an accurate approximation. 

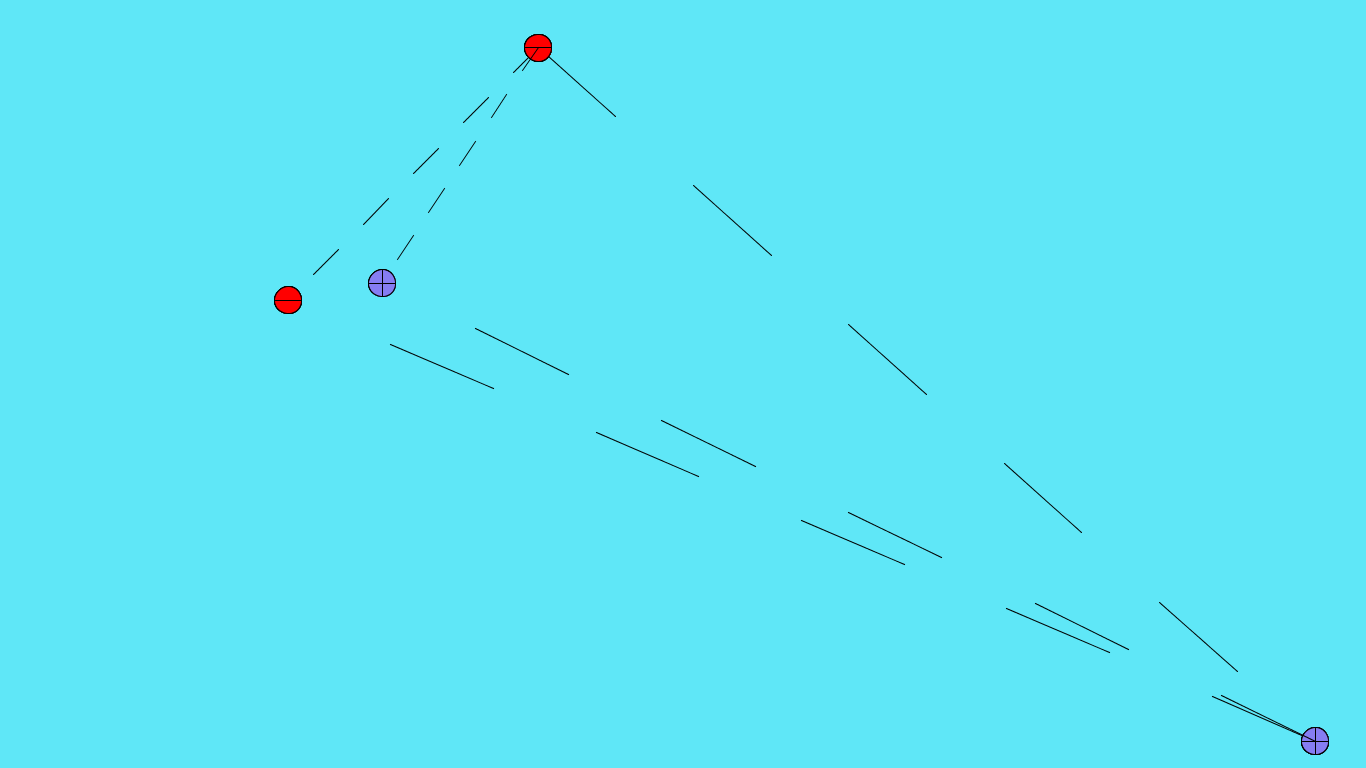
Upon the case a charge is exceeding the boundaries of the screen, the wall of the container, the screen, reverses the according velocity in transgression and accounts for the electrostatic work done in this process.

Upon the occasions in which two charges may collide such that interiors do overlap, two assumptions are employed—between charges of different signs, it is assumed that an atom is formed with the negative charge incorporate into the orbital of the positive, which would then gradually fade in color and ultimately ignored; between charges of the same sign, the two particles fused into one particle with the summed charge and mass, which proceeds in a random direction. Despite that the crudity of such assumptions are plainly perceived, as the tenor and purport of this simulation rests solely in the motion of charges under simple standard conditions, they are allowed.





A concise visualization of the internal calculation is provided by connecting particles with dashed lines indicating the computation of electrostatic forces between such particles.



This lines are drawn by a customized connector class which, by the boolean state control idiom in the draw() method, completes the connection between two charges or proceed to connect two new ones. Holding the enter key thus demonstrates.

Finally, fields are customized to certain quantities—charge, mass, velocities, and positions are random; a particle may be populated at a specific position at run time by clicking the screen; all particles are given a certain diameter, and all orbits are of a certain radius; the background color and that of charges are also customized. All the above fields can be customized and adjusted as they are listed as global variables, with suggestive names and sufficient documentations.

The overall runtime of such a program per frame is *O*(*n*^2). It should be noted that *n* accounts for charges that may have experienced collisions previously as well.