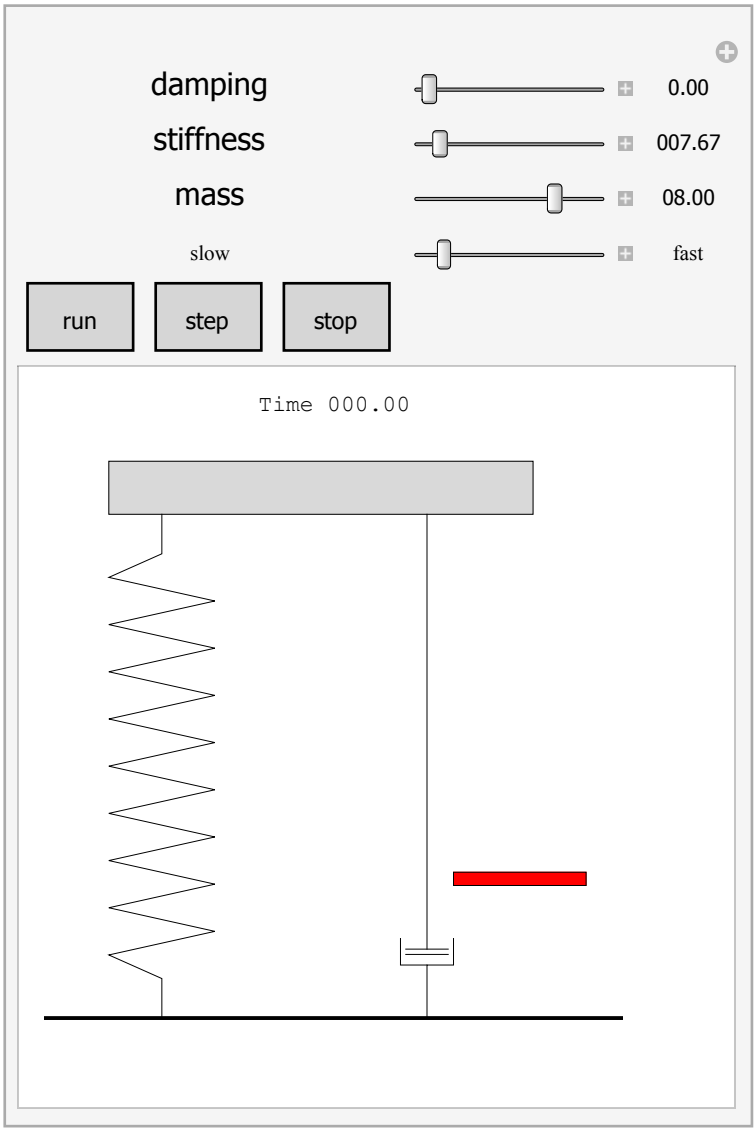
```
Manipulate[
 (*by Nasser M. Abbasi, June,17,2014*)
 tick;
 Module[{data, eq, y, t, pos, vel, sol, graph},
  data = rugo[-3, 0, -3, y0 - massThickness];
  graph = makeGraph[currentTime, y0, massThickness, L1, data, False];

  If[(runningState == "RUNNING" || runningState == "STEP"),

   eq = (m y''[t] + c y'[t] + k (y[t] - L0) == -m g);
   sol = NDSolve[{eq, y[0] == y0, y'[0] == v0}, {y, y'}, {t, 0, deltT}];
   {pos, vel} = {y, y'} /. First@sol;
   currentTime = currentTime + deltT;
   y0 = pos[deltT];
   v0 = vel[deltT];
   If[(y0 - massThickness) < (L1 + 0.25),
    v0 = -v0;
    graph = makeGraph[currentTime, y0, massThickness, L1, data, True]
   ]
  ];
  If[currentTime > 999.9, currentTime = 0];
  If[runningState == "RUNNING",
   tick = Not[tick]
  ];
  graph
 ],

 Grid[{
   {"damping",
    Manipulator[Dynamic[c, {c = #, currentTime = 0, v0 = 0, y0 = L0} &],
     {0, 1, 0.01}, ImageSize → Small], Style[Dynamic@padIt2[c, {3, 2}], 11]
   },
   {"stiffness",
    Manipulator[Dynamic[k, {k = #, currentTime = 0, v0 = 0, y0 = L0} &],
     {1, 100, 0.01}, ImageSize → Small], Style[Dynamic@padIt2[k, {5, 2}], 11]
   },
   {"mass",
    Manipulator[Dynamic[m, {m = #, currentTime = 0, v0 = 0, y0 = L0} &],
     {1, 10, 0.01}, ImageSize → Small], Style[Dynamic@padIt2[m, {4, 2}], 11]
   },
   {Text@Style["slow", 11],
    Manipulator[Dynamic[deltT, {deltT = #} &], {0.001, .1, 0.001},
     ImageSize → Small, ContinuousAction → False], Text@Style["fast", 11]
   },
   {Grid[{
      {
       Button[Style["run", 12], {runningState = "RUNNING";
         tick = Not[tick]}, ImageSize → {55, 35}],
       Button[Style["step", 12], {runningState = "STEP";
         tick = Not[tick]}, ImageSize → {55, 35}],
       Button[Style["stop", 12], {runningState = "STOP";
         currentTime = 0, v0 = 0, y0 = L0}, ImageSize → {55, 35}]
```

```
      }
     }
    ]
   }
  }
 ],
 {{wasHit, False}, None},
 {{y0, 10}, None},
 {{v0, 0}, None},
 {{m, 8}, None},
 {{k, 26}, None},
 {{c, 0}, None},
 {{runningState, "STOP"}, None},
 {{currentTime, 0}, None},
 {{deltT, 0.01}, None},
 {{tick, True}, None},
 TrackedSymbols :→ {tick},
 Initialization :→
  (
   g = 9.8; L0 = 10; massThickness = 0.5; L1 = 2.5;

   makeGraph[currentTime_, y0_, massThickness_, L1_, data_, hit_] := Module[{splash},

     splash = If[hit,
       {Red, Text[Style["Bang!!", 16], {5, L1 + 0.6}]}, Sequence @@ {}
      ];
     Grid[{
       { Row[{"Time ", padIt2[currentTime, {5, 2}]}]},
       {
        Graphics[
         {
          {EdgeForm[Black], LightGray,
           Rectangle[{-4, y0 - massThickness}, {4, y0 + massThickness}]},
          Line[{{2, 0}, {2, 1}, {2.5, 1}, {2.5, 1.5}}],
          Line[{{2, 1}, {1.5, 1}, {1.5, 1.5}}],
          Line[{{1.6, 1.2}, {2.4, 1.2}}],
          Line[{{1.6, 1.3}, {2.4, 1.3}}],
          Line[{{2, 1.3}, {2, y0 - massThickness}}],
          {EdgeForm[Black], Red, Rectangle[{2.5, L1}, {5, L1 + 0.25}]},
          splash,

          {Thick, Line[{{-6, 0}, {6, 0}}]},
          Line[data]
         }, PlotRange → {{-5, 5.5}, {-1, 11}},
         Axes → False, ImageSize → 300, ImagePadding → 5
        ]}
      }
     ]
    ];

   (*definitions used for parameter checking*)
   integerStrictPositive = (IntegerQ[#] && # > 0 &);
```

```
    integerPositive = (IntegerQ[#] && # ≥ 0 &);
    numericStrictPositive = (Element[#, Reals] && # > 0 &);
    numericPositive = (Element[#, Reals] && # ≥ 0 &);
    numericStrictNegative = (Element[#, Reals] && # < 0 &);
    numericNegative = (Element[#, Reals] && # ≤ 0 &);
    bool = (Element[#, Booleans] &);
    numeric = (Element[#, Reals] &);
    integer = (Element[#, Integers] &);
    padIt1[v_?numeric, f_List] := AccountingForm[Chop[v], f,
      NumberSigns → {"-", "+"}, NumberPadding → {"0", "0"}, SignPadding → True];
    padIt2[v_?numeric, f_List] := AccountingForm[Chop[v], f,
      NumberSigns → {"", ""}, NumberPadding → {"0", "0"}, SignPadding → True];
    padIt2[v_?numeric, f_Integer] := AccountingForm[Chop[v], f,
      NumberSigns → {"", ""}, NumberPadding → {"0", "0"}, SignPadding → True];

    rugo[xkezd_, ykezd_, xveg_, yveg_] := Module[{step = 20, szel = 1
        (*spring width*), hx, hy, veghossz = 0.3, hossz, dh, i}, {hx = xveg - xkezd;
         hy = yveg - ykezd;
         hossz = Sqrt[hx^2 + hy^2];
         dh = (hossz - 2 * veghossz) / step;
         {xkezd, ykezd}}~Join~
        {{xkezd + hx * (dh + veghossz) / hossz, ykezd + hy * (dh + veghossz) / hossz}}~Join~
        Table[If[OddQ[i], {xkezd + hx * (i * dh + veghossz) / hossz + hy * szel / hossz,
            ykezd + hy * (i * dh + veghossz) / hossz - hx * szel / hossz},
           {xkezd + hx * (i * dh + veghossz) / hossz - hy * szel / hossz,
            ykezd + hy * (i * dh + veghossz) / hossz + hx * szel / hossz}], {i, 2, (step - 2)}]~
        Join~{{xkezd + hx * ((step - 1) * dh + veghossz) / hossz,
          ykezd + hy * ((step - 1) * dh + veghossz) / hossz}}~Join~{{xveg, yveg}}
     ]
  )
]
```


damping
0.00
stiffness
007.67
mass
08.00
slow
fast
run
step
stop
Time 000.00