Среда моделирования двумерной школьной физики

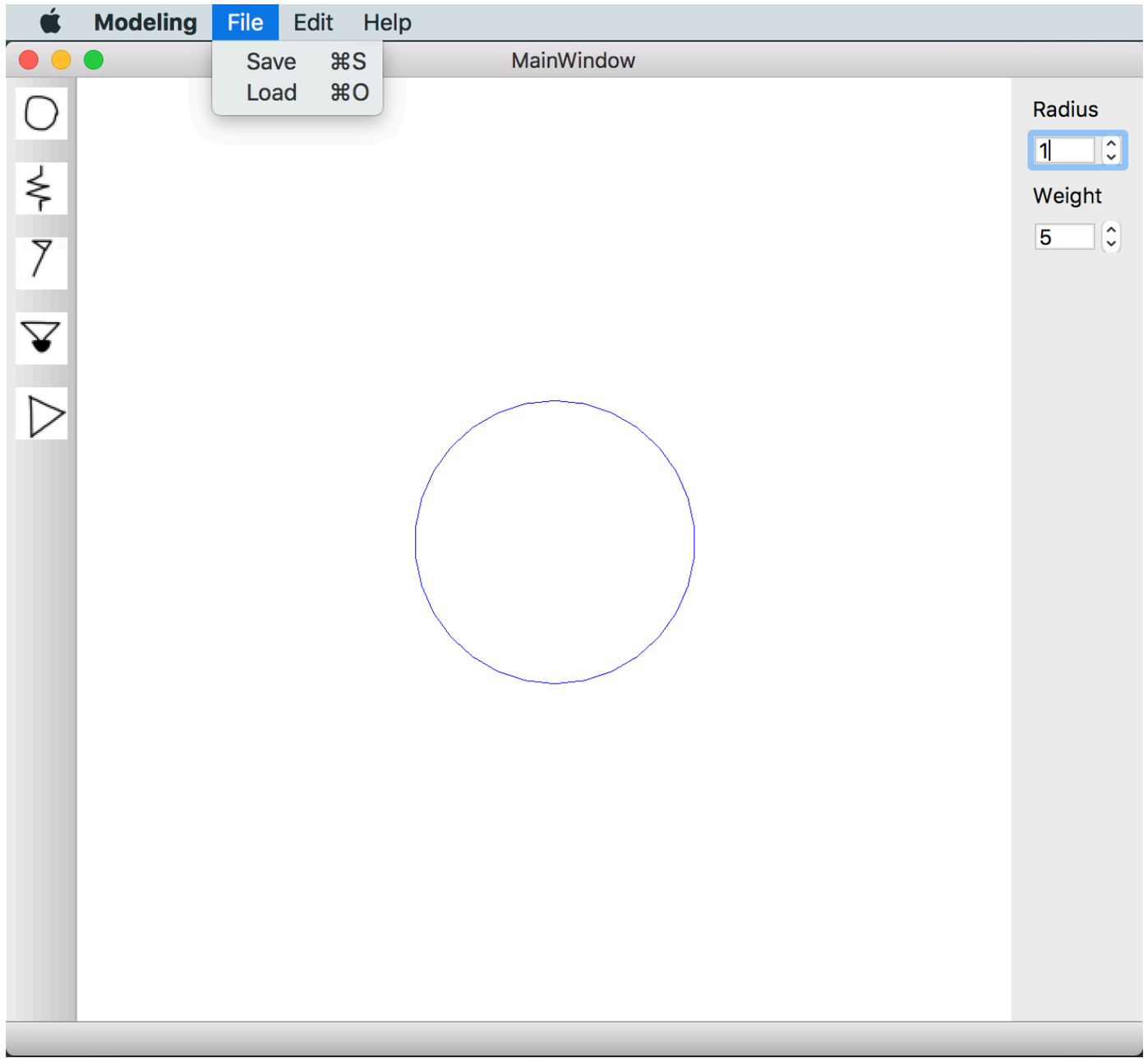
Лапатин Владимир, ИУ9-51

Москва, 2017

Постановка задачи

Изучение математического аппарата, способного смоделировать двумерную школьную физику и создание программы, выполняющей данное моделирование

Реализация интерфейса



Реализация интерфейса

Шаблон MVC

Модель — ModelingModel, RungeCutta, DrawableObject и все его наследники.

Представление — MainWindow, OpenGLWidget.

Контроллер — MainWindow.

Разработка математической модели

Формулы: из уравнения Лагранжа 2-го рода:

$$\frac{\partial}{\partial t}(\frac{\partial L}{\dot{q}_i}) - \frac{\partial L}{q_i} = 0$$

Итоговые формулы:

$$\ddot{x}_i = -\frac{1}{m_i}\sum_{j=1}^{n}\frac{k_i(\sqrt{(x_{i2}-x_{i1})^2+(y_{i2}-y_{i1})^2}-L_{i0})(x_{i2}-x_{i1})}{\sqrt{(x_{i2}-x_{i1})^2+(y_{i2}-y_{i1})^2}}$$

$$\ddot{y}_i = -\frac{1}{m_i}(m_i g + \sum_{j=1}^{n}\frac{k_i(\sqrt{(x_{i2}-x_{i1})^2+(y_{i2}-y_{i1})^2}-L_{i0})(y_{i2}-y_{i1})}{\sqrt{(x_{i2}-x_{i1})^2+(y_{i2}-y_{i1})^2}})$$

$$\ddot{\varphi}_i = -\frac{1}{m_i l_i^2}(m_g l_i sin\varphi_i + \sum_{j=0}^{n}\frac{k_i(\sqrt{(x_{stat}-l_i sin\varphi_i - x_{mat})^2+(y_{stat}-l_i cos\varphi_i - y_{mat})^2}-L_{i0})(-l_i cos\varphi_i(x_{stat}-l_i sin\varphi_i - x_{mat})+l_i sin\varphi_i(y_{stat}-l_i cos\varphi_i - y_{mat}))}{\sqrt{(x_{stat}-l_i sin\varphi_i - x_{mat})^2+(y_{stat}-l_i cos\varphi_i - y_{mat})^2}}$$

После чего — численное интегрирование методом Рунге-Кутты четвертого порядка

Реализация интерактивной модели

Изображения с сайта ru.wikipedia.org

Реализация интерактивной модели

Конечная сила, действующая на каждую точку, может быть произвольной. Поэтому силы хранятся в замыканиях языка C++.

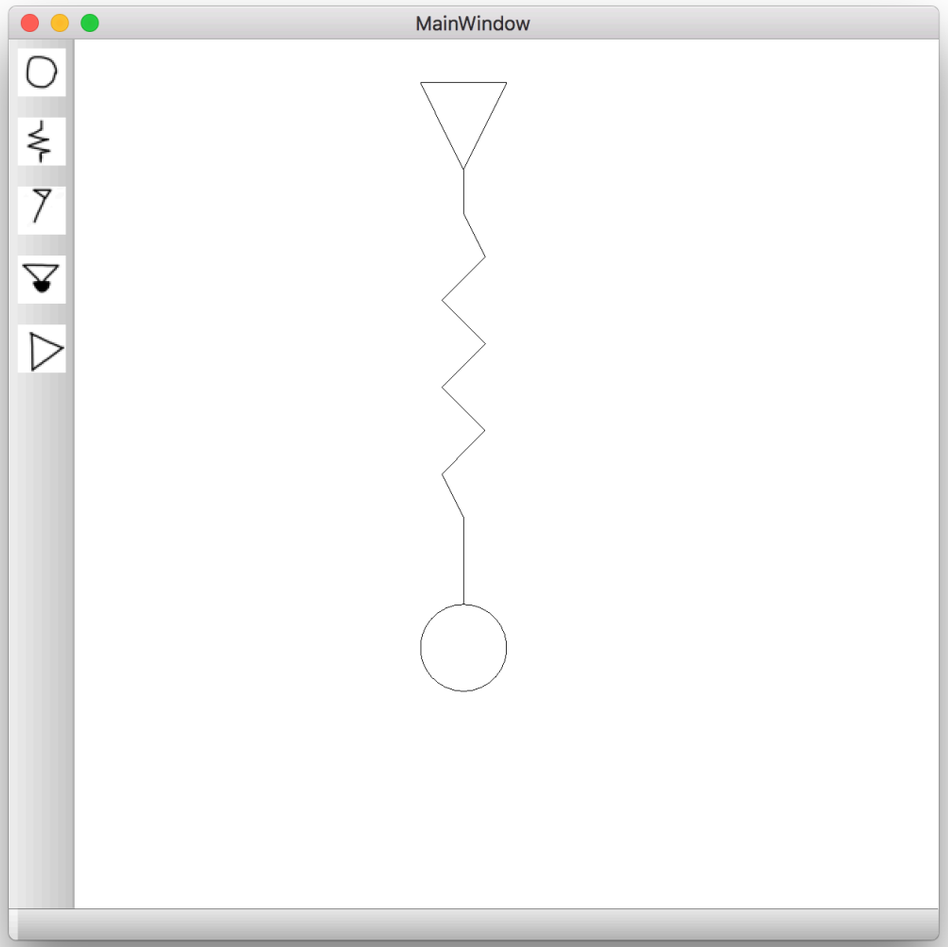
```
[capturingAccelerationX, k, m, L0, x2Index, x1Index](
      std::valarray<double> args)
{
      double x2 = args[6*x2Index];
      double y2 = args[6*x2Index + 2];
      double x1 = args[6*x1Index];
      double y1 = args[6*x1Index + 2];
      double square = std::hypot(x2 - x1, y2 - y1);
      return capturingAccelerationX(args) +
            (-1.0) / m * k * (square - L0) * (x2 - x1)  / square;
}
[capturingAccelerationY, k, m, L0, x2Index, x1Index](
      std::valarray<double> args)
{
      double x2 = args[6*x2Index];
      double y2 = args[6*x2Index + 2];
      double x1 = args[6*x1Index];
      double y1 = args[6*x1Index + 2];
      double square = std::hypot(x2 - x1, y2 - y1);
      return capturingAccelerationY(args) +
            (-1.0) / m * k * (square - L0) * (y2 - y1) / square;
}
```

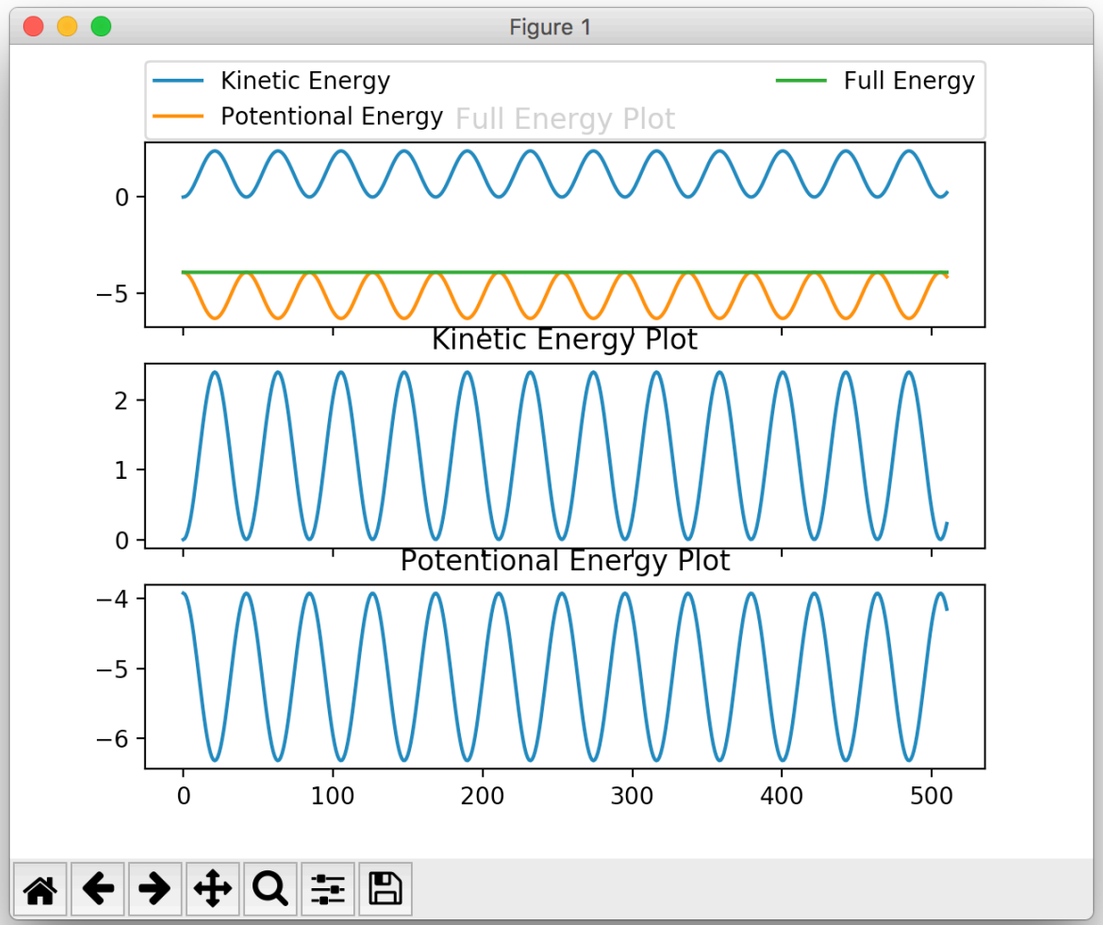
Тестирование

Критерии тестирования:
Визуальная корректность
Выполнение закона сохранения энергии

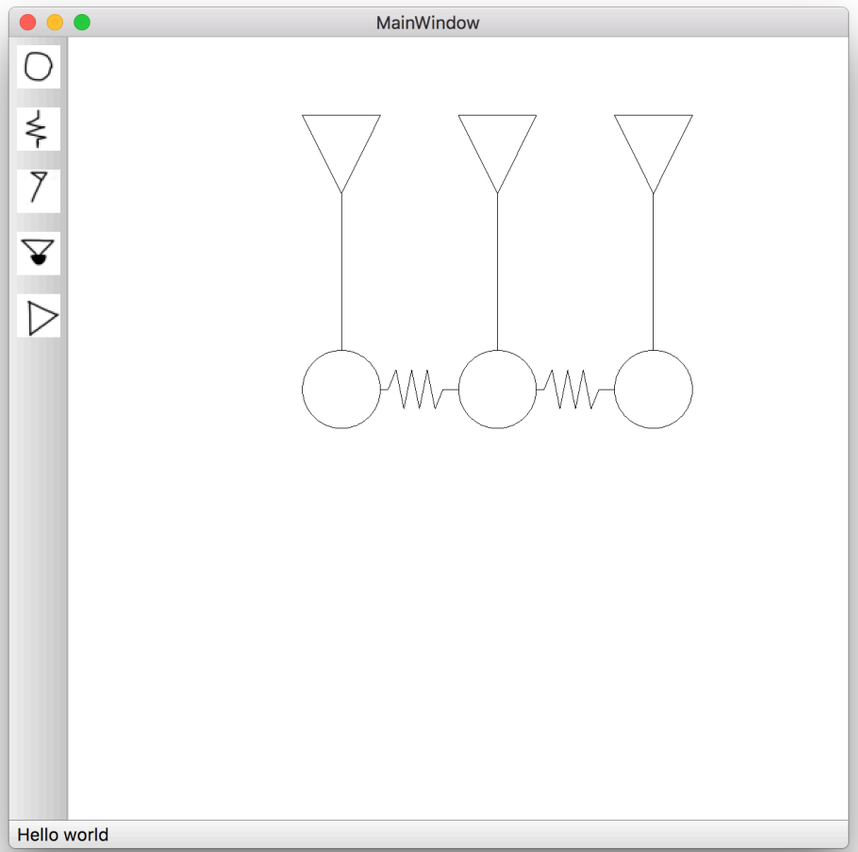
Для этого выполняется сохранение кинетической и потенциальной энергии на каждой итерации в CSV файл.
После чего на основе этих данных выполняется построение графиков кинетической, потенциальной энергий и их суммы

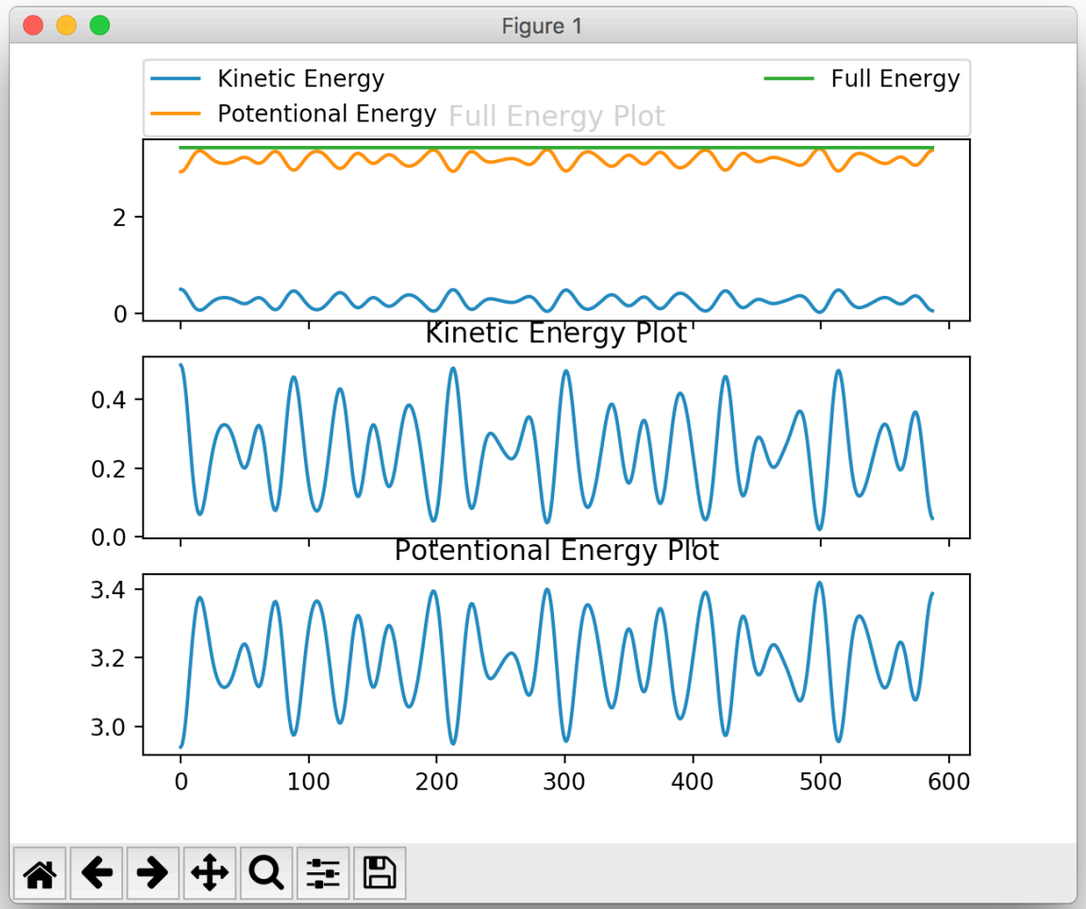
Тестирование





Тестирование





Тестирование