

Название проекта:

Румунк симуляция (падения, столкновения набора тел с разными параметрами)

Описание идеи:

Проект позволяет создавать наборы объектов, задавать параметры и запускать симуляцию для просмотра.

Физическая симуляция использует глобальные параметры ускорение свободного падения, коэффициент трения поверхности и для каждого объекта задается вектор силы (по x и по y отдельно).

Описание реализации:

Структура проекта

./ui содержит ui файлы форм и диалогов, созданные в qt designer

./db sqlite базу rumunk.db, init_db.sql содержит sql(для создания в базе таблиц)

Стартовая точка main.py загружает основное окно.

За запуск настроенной симуляции отвечает logic.py.

Параметры симуляции это PGParameters.py, а наборы объектов (PMStorage.py::PMObject)

Их можно редактировать в главном окне программы (main.py).

Поддерживается сохранение и загрузка в базу данных (DataProvider.py).

Экспорт и импорт текущего набора PMObject в csv (FileObjectsProvider.py)

За отображение симуляции отвечает библиотека PyGame, работа с ней вынесена в PyGameRender.py.

Обработка цикла событий окна PyGame и вызов расчета симуляции реализован с помощью выделенного таймера в том же классе.

В PyGame симуляция рассчитанная PyMunk отрисовывается за счет вызова rumunkSim.space.debug_draw(self.draw_options) поскольку это просто демонстрация использования.

За работу с библиотекой rumunk отвечает класс RumunkSimulation.

В класс загружается набор PMObject из PMStorage. Для каждого объекта создается обертка Square, который отвечает за применение параметров к конкретному телу для rumunk, создает связанные с телом: body(для расчета физической симуляции), shape(для отображения в pygame), содержит метод step, который вызывается при каждом пересчете симуляции в цикле отрисовки для применения силы.

Описание реализации интерфейса:

Для размещения виджетов используются разные типы Layout и Spacer. Для передачи параметров созданного в диалоге EditPMObjectQDialog используется сигнал onAcceptSignal.

В EditPMObjectQDialog для выбора и показа цвета используется QLabel и QPushButton, отображение цвета делается через применения стиля к Label, а выбор цвета с помощью показа стандартного QColorDialog.

Для отображения PMObject используется TableView и модель данных (PMTTableModel.py),

что позволяет менять только модель для отображения изменений в наборе, а не сами виджеты интерфейса.

Необходимые для запуска библиотеки находятся в requirements.txt



