ЛАБОРАТОРНАЯ РАБОТА 2
ДВИЖЕНИЕ ОБЪЕКТА ПО ЗАДАННОЙ ТРАЕКТОРИИ

Разработать модель, которая демонстрирует движение сферического объекта синего цвета по заданной траектории. Траектория объекта изменяется по синусоидальному закону. Когда траектория объекта приближается к значению амплитуды траектории с погрешностью равной 0,01 окрасить его в красный цвет, при удалении восстановить синий цвет.

Таблица 2.1 - Параметры траектории

№	Параметр	Значение
1	Амплитуда	3 метра
2	Угловая частота	0,5 рад/с
3	Модельное время	500с

Для создания модели нужно сформировать новый проект. Модель создается с нуля. Название модели AnimalBall2.

Используя контекстное меню узла модели AniamalBall2, создайте активный класс Ball (см рисунок 2.1).

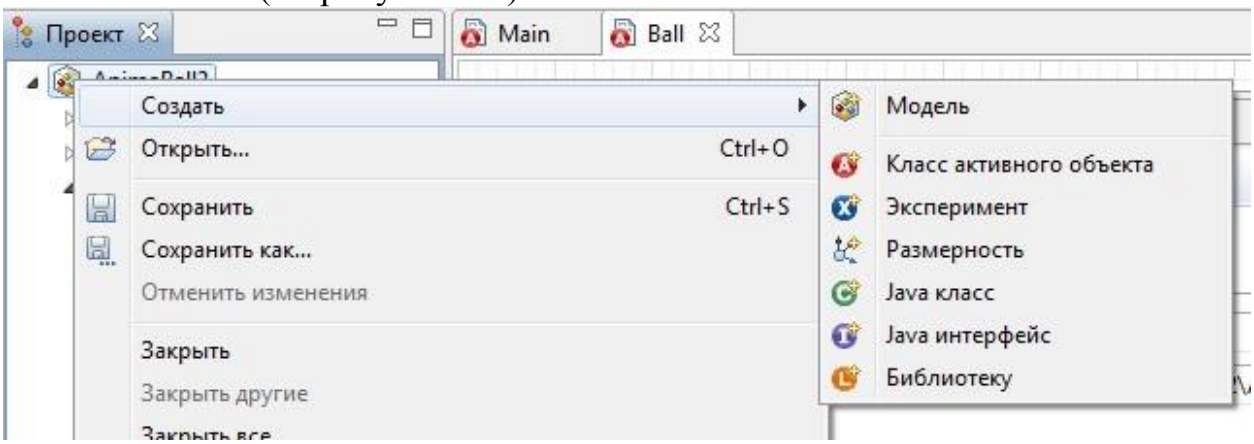

Рисунок 2.1 - Создание нового активного класса

На рисунке 2.2 приводятся элементы активного объекта Ball

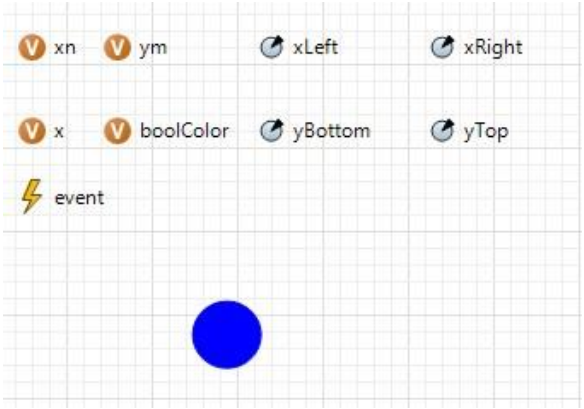

Рисунок 2.2 - Структура активного класса Ball

Для размещения закрашенного круга используйте палитру «Презентация» и элемент «Овал». Настройте это элемент так, как это показано на рисунке 2.3.

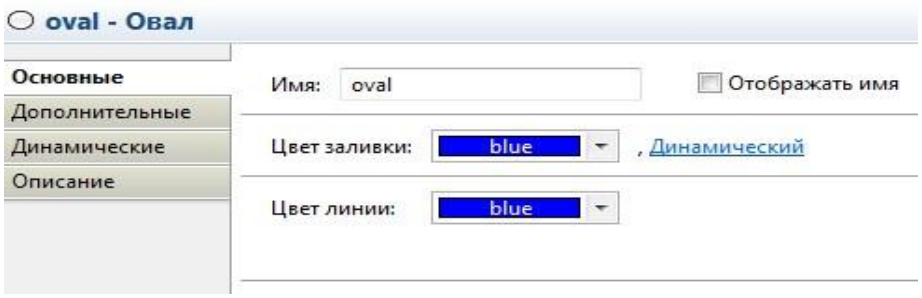

Рисунок 2.3 - Свойства активного элемента oval

Элементы активного класса должны соответствовать таблице 2.2.

Таблица 2.2 - Активный класс Ball

№	Элемент	Назначение
1	x_n	Положение шара по оси X – физическая координата
2	x	мировое - расчетное значение траектории
3	y_m	Положение шара по оси Y – физическая координата
4	boolColor	Логическая переменная, начальное значение «ЛОЖЬ»
5	xLeft	Левая координата физическая области вывода по оси X=0
6	yBottom	Левая координата физическая области вывода по оси Y=-3
7	xRight	Правая координата физическая области вывода по оси X=500
8	yTop	Правая координата физическая области вывода
		по оси X=+3
9	event	Событие

Настройка события должна соответствовать рисунку 2.4.

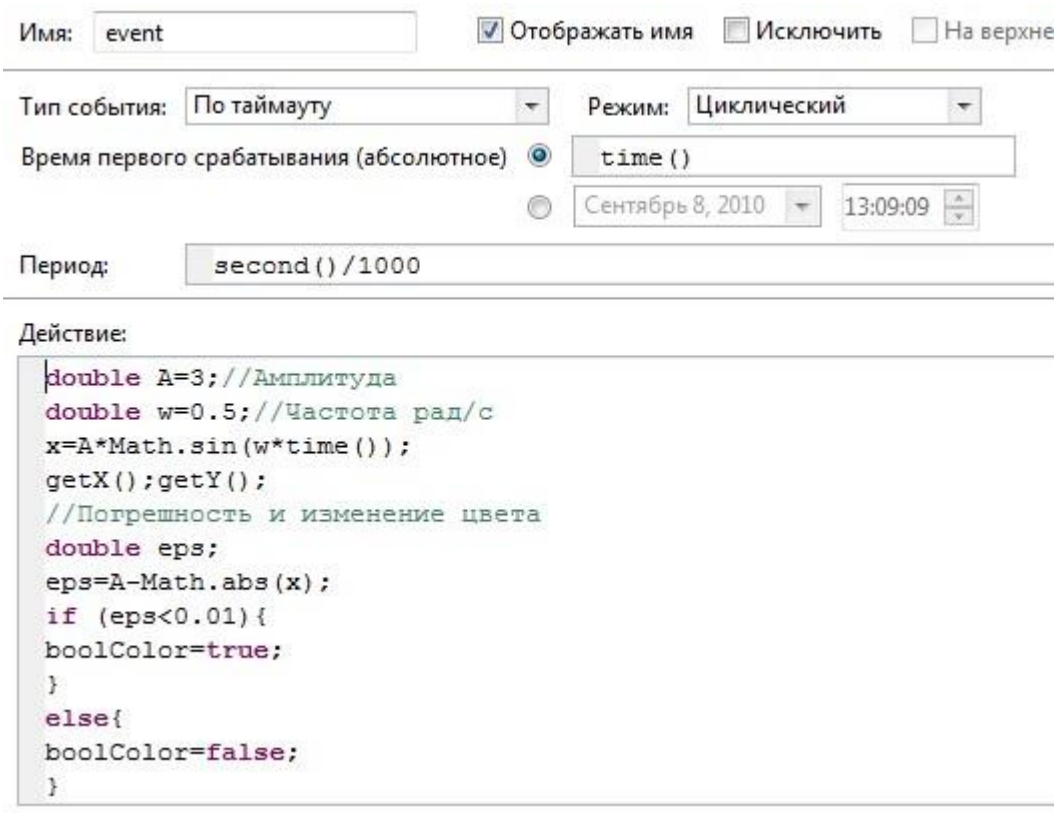

Рисунок 2.4 - Настройка элемента event

При достижении погрешности по отношению к амплитуде равной 0,01 переменная boolColor получает значение «ИСТИНА», что является сигналом для смены цвета шара.

Для того чтобы шар двигался и изменялся его цвет нужно настроить его свойства так как это показано в таблице 2.3.

Таблица 2.3 - Динамические свойства

№	Свойство	Значение
1	X	x_n
2	Y	y_m
3	Цвет заливки	boolColor ? new Color(255,0,0): new Color(0,0,255)

Чтобы вывод анимации осуществлялся в заданную область, следует выполнить пересчет мировых – расчетных координат объекта в физические координаты. Физические координаты это координаты в пикселах в области вывода объекта при его анимации.

Соответствие между физическими и мировыми координатами показано на рисунке 2.5.

В общем виде формулы пересчета имеют вид:

$$x_n = \frac{x - xLeft}{xRight - xLeft} \times (nRight - nLeft) + nLeft$$

$$y_m = \frac{y - yBottom}{yTop - yBottom} \times (mTop - mBottom) + mBottom$$

Где: x_n и y_m физические координаты

Откройте код настройки активного класса Ball и введите код для двух функций, осуществляющих пересчет физических координат в мировые, так как это показано на рисунке 2.6.

Движение сферического объекта будет моделироваться в области объекта Main. Для этого следует в поле этого объекта ввести прямоугольную область с помощью элемента палитры «Презентация» «Прямоугольник». Настройка прямоугольника должна соответствовать рисунку 2.6.

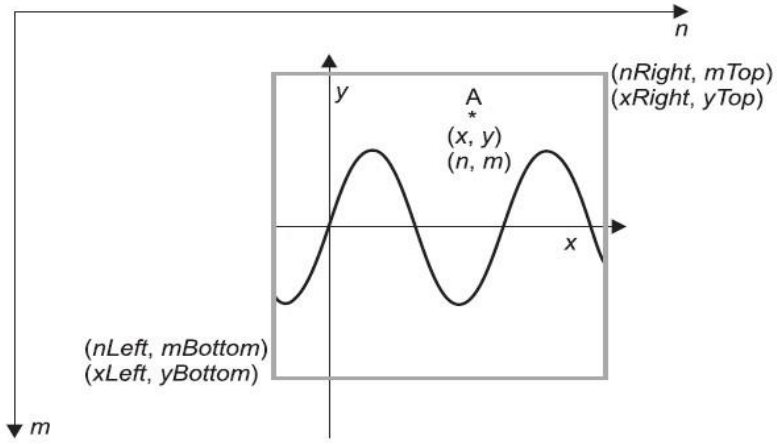

Рисунок 2.5- Соответствие физических и мировых координат.

На рисунке 2.5 Префиксы n и m соответствуют физическим координатам, x и y мировым координатам

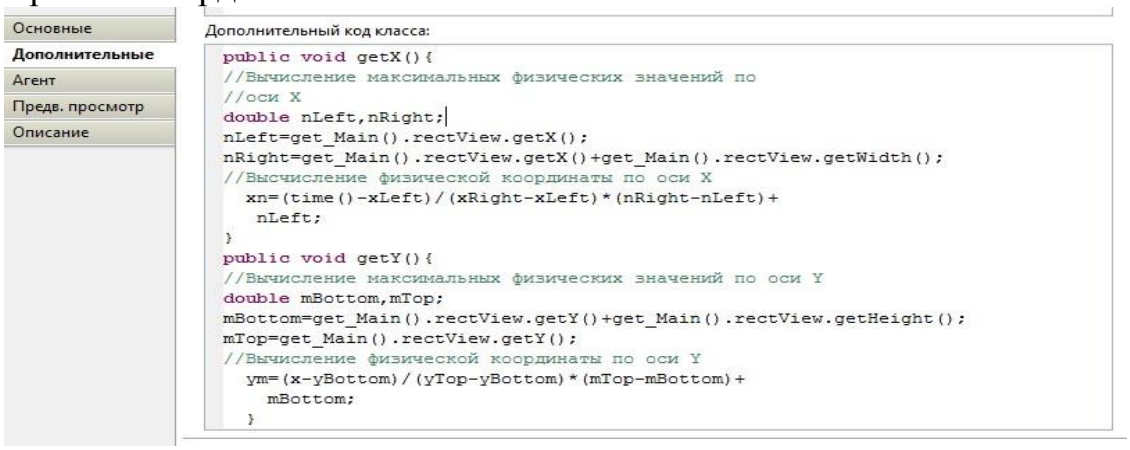

Рисунок 2.6 - Функции пересчета мировых координат в физические

При анимации объект Ball будет двигаться в прямоугольной области объекта Main. Для размещения этой области нужно сделать активным этот объект и используя палитру «Презентация» создать прямоугольную область с помощью элемента «Прямоугольник».

Настройки прямоугольной области показаны на рисунке 2.8.

При программировании пересчета мировых координат в физические координаты нужно автоматизировать определение краевых значений физической области вывода (см. рисунок 2.5).

Для обращения к объекту Main, с целью получения экземпляра прямоугольной области, используется оператор

Java get_Main().Имя_объекта для нашей модели именем объекта является идентификатор прямоугольной области rectView.

Как любая фигура в технологии Java прямоугольник характеризуется свойствами, показанными ни рисунке 2.7

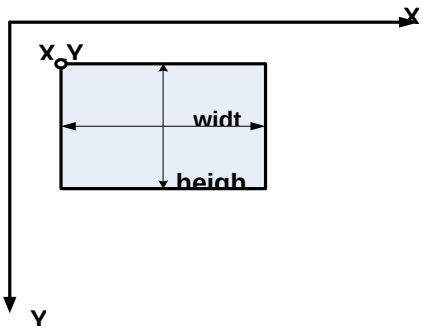

Рисунок 2.7 - Фигура Java и ее свойства

На рисунке 2.7 width и height – ширина и высота фигуры, Xn,Ym – координаты точки прорисовки фигуры

Для определения свойств фигуры используются ее методы, показанные в таблице 2.4.

Таблица 2.4 - Методы фигуры

№	Метод	Назначение
1	double getWidth()	Получение ширины
2	double getHeight()	Получение высоты
3	double getX()	Точка Xn
4	double getY()	Точка Ym

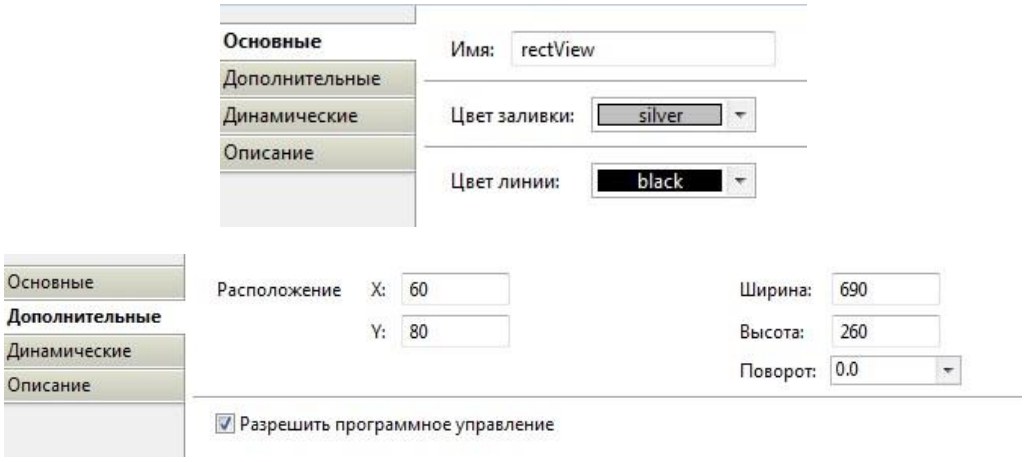

Рисунок 2.8 - Настройка прямоугольника класса Main

После выполнения настройки объектов нужно сделать объект Ball вложенным в объекте Main. Для этого нужно открыть дерево проекта. Оно находится слева. Если его нет, то его нужно вывести командой меню AnyLogic Панель > Проект.

Затем нужно выбрать объект Main двойным щелчком мыши, используя технологию Drag & Drop, перетащить объект Ball в поле объекта Main, так как это показано на рисунке 2.9.

Если вложенный объект скрывается за прямоугольной областью, то следует воспользоваться инструментами панели AnyLogic

 для изменения взаимного расположения элементов.

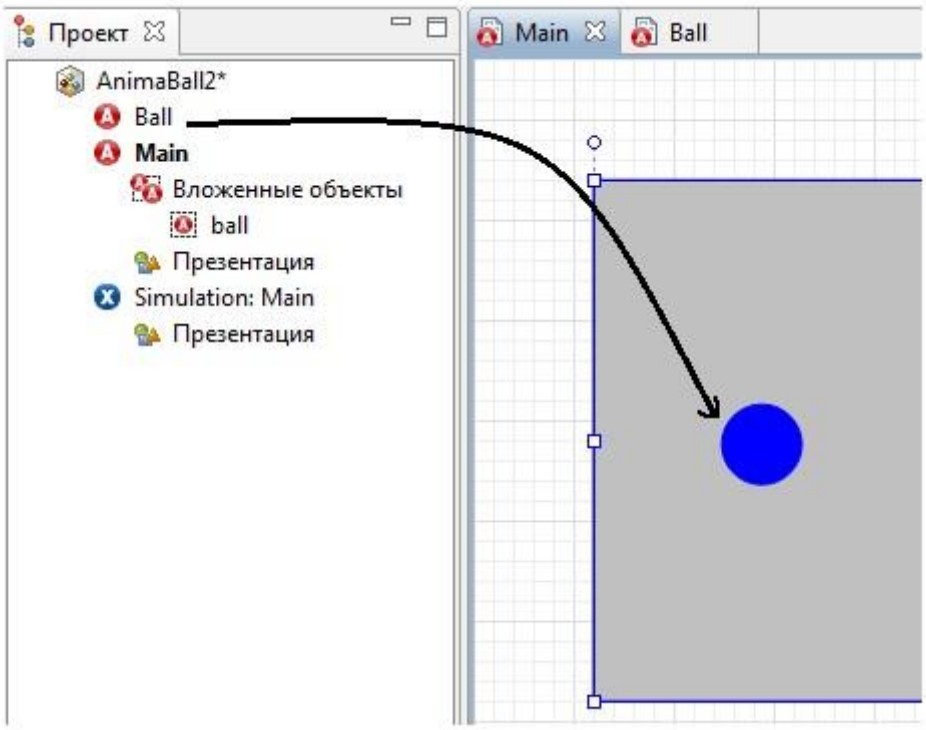

Рисунок 2.9 - Создание вложенного объекта в классе Main

Внимание: После размещения встроенного объекта его нельзя перемещать в поле вывода, так как это сделает невозможным правильный автоматический пересчет мировых координат в физические!

После получения вложенного объекта следует сохранить модель и настроить параметры эксперимента Simulation, так как это показано на рисунке 2.10.

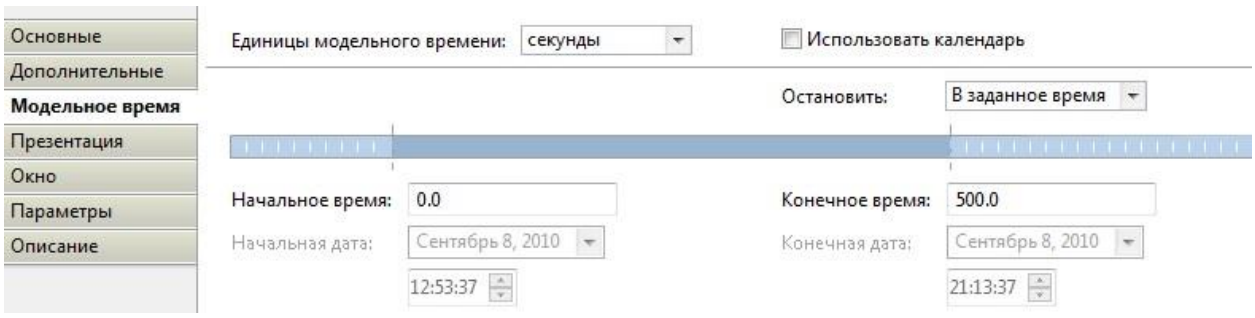

Рисунок 2.10 - Настройка эксперимента

На рисунке 2.11а показана модель в действии.

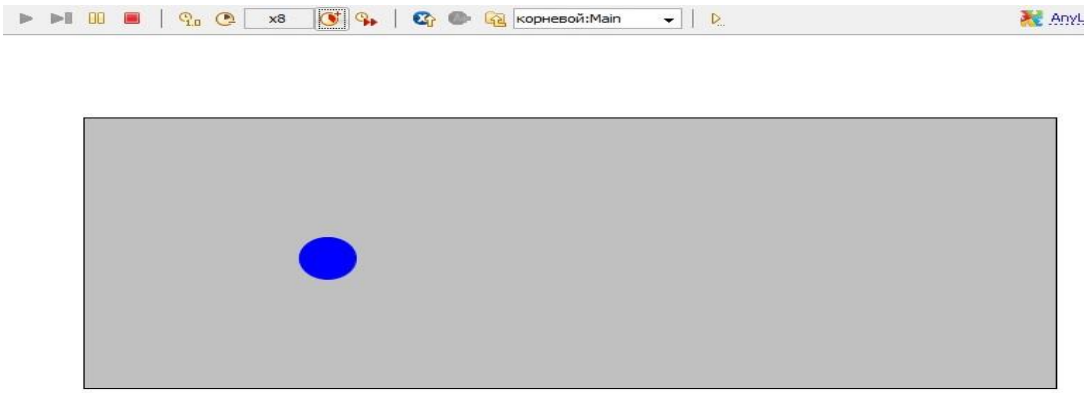

Рисунок 2.11а - Движение сферы в прямоугольной области. Коэффициент ускорения обработки модели восемь