ЗАДАНИЕ 1
Первая модель на Anylogic

1.1. ИНТЕРФЕЙС ПРОГРАММЫ

Когда Вы первый раз запустите AnyLogic, Вы увидите начальную страницу. Начальная страница содержит краткое описание основных возможностей программы , ссылки на примеры моделей, поставляемые вместе с AnyLogic, а также ссылки на веб-сайт компании XJ technologies (разработчика этого программного продукта) и на форму обратной связи с компанией.

Чтобы закрыть начальную страницу, щелкните мышью по кнопке 

в панели заголовка начальной страницы.

С помощью кнопки **Открыть модель** панели инструментов или команды **Файл** / **Открыть** в главном меню выберите файл Balls. Это несложная модель прыгающего мяча. На экране Вы увидите следующее окно - рис. 1.1.

AnyLogic при открытии проекта открывает несколько панелей: проекты, диаграмма, палитра, ошибки и свойства. Рисунок 1.1 показывает основные составляющие пользовательского интерфейса. Рассмотрим их поочередно.

Панель **Проекты** обеспечивает навигацию по элементам открытых моделей. Поскольку модель организована иерархически, то она отображается в виде дерева: сама модель образует верхний уровень дерева; классы активных объектов и эксперименты образуют следующий уровень и т.д.

Панель **Проекты** по умолчанию прикреплена к левой части рабочей области AnyLogic.

Полужирным шрифтом в дереве выделяется тот элемент, редактор которого активен в текущий момент.

Если Вы внесете в модель какие-то изменения и не сохраните их, то такая модель будет сразу же выделена в дереве - к имени модели будет добавлена звездочка (*).

Вы можете разворачивать и сворачивать ветви дерева элементов модели с помощью кнопок + и -.

У каждого класса активного объекта и эксперимента есть своя диаграмма, которая редактируется в графическом редакторе.

На диаграммах Вы можете:

- Нарисовать презентацию с помощью фигур и элементов управления.
- Задать поведение активного объекта с помощью событий и диаграмм действий.
- Задать структуру класса, добавив вложенные объекты.
- Добавить на презентацию визуализирующие графики, диаграммы.

Панель **Палитра** содержит элементы, которые могут быть добавлены на диаграмму класса активного объекта или эксперимента.

По умолчанию она прикреплена к правому краю окна приложения.

Панель **Палитры** состоит из нескольких вкладок (палитр), каждая из которых содержит элементы, относящиеся к определенной задаче:

- **Основная** содержит основные элементы, с помощью которых Вы можете задать динамику модели, ее структуру и данные.
- **Системная динамика** содержит: элементы диаграммы потоков и накопителей, а также параметр, соединитель и табличную функцию.
- **Диаграмма состояний** содержит блоки диаграмм, позволяющих графически задавать поведение объекта.

- **Диаграмма действий** содержит блоки структурированных блок-схем, позволяющих задавать алгоритмы визуально.
- **Статистика** содержит элементы, используемые для сбора, анализа и отображения результатов моделирования.
- **Презентация** содержит элементы для рисования презентаций: примитивные фигуры, а также элементы управления, для придания презентации интерактивности.
- **Внешние данные** содержит инструменты для работы с базами данных и текстовыми файлами.
- **Картинки** содержит набор картинок наиболее часто моделируемых объектов: человек, грузовик, фура, погрузчик, склад, завод и т. д.

Панель **Свойства** используется для просмотра и изменения свойств выбранного в данный момент элемента модели.

Панель **Свойства** содержит несколько вкладок. Каждая вкладка содержит элементы управления, такие как поля ввода , флажки, переключатели, кнопки и т.д., с помощью которых Вы можете просматривать и изменять свойства элементов модели. Число вкладок и их внешний вид зависит от типа выбранного элемента.

Вы можете, как Вам угодно перемещать панели в пределах окна AnyLogic. Для восстановления принятых по умолчанию настроек расположения панелей нужно в главном меню вызвать **Окно / Восстановить расположение панелей**.

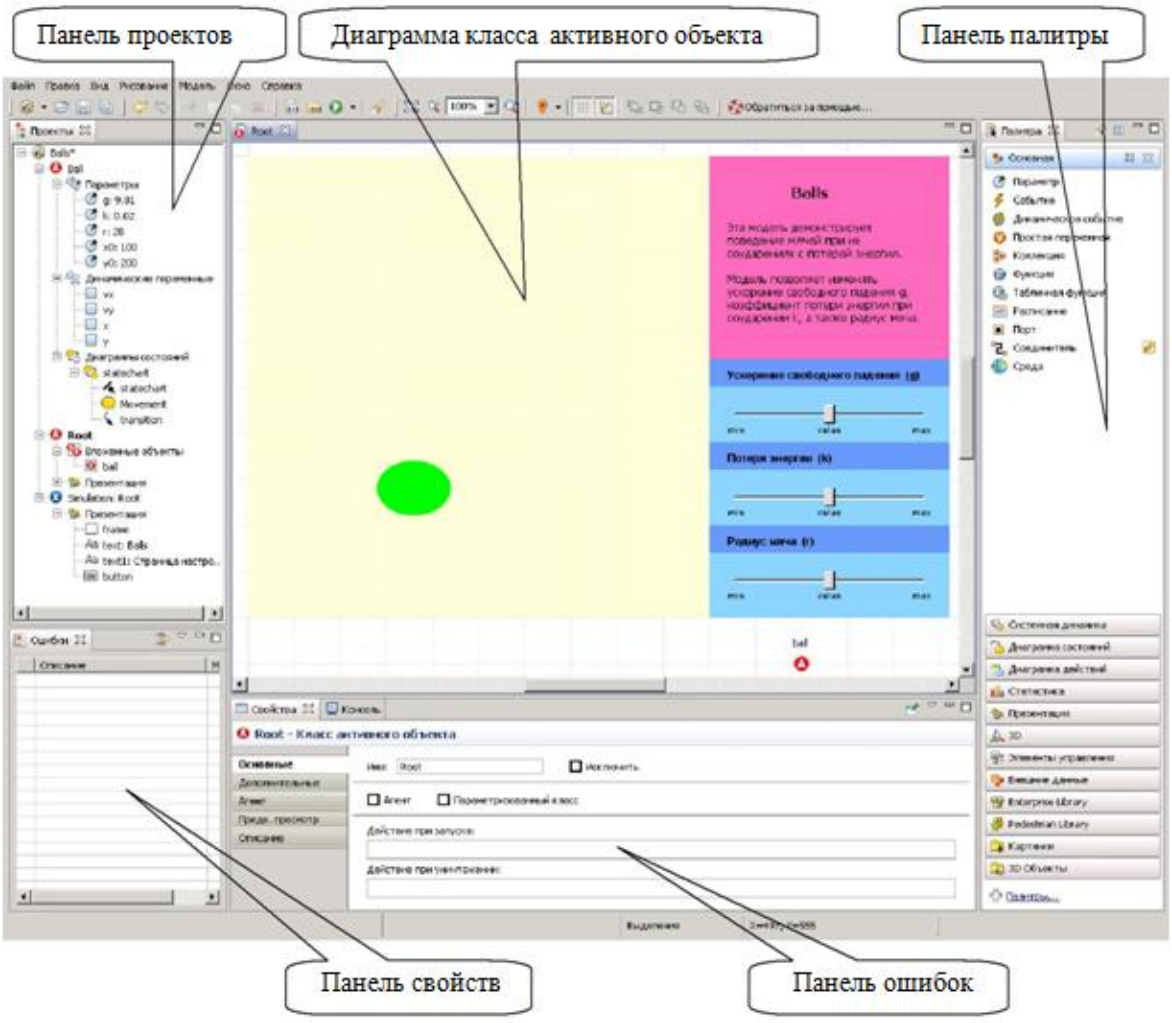

Рис. 1.1

На этапе компиляции модели AnyLogic производит проверку синтаксиса диаграмм, типов и параметров. Все обнаруженные на этапе компиляции и построения модели ошибки отображаются в панели **Ошибки**. Для каждой ошибки показывается ее описание и

местоположение - имя элемента модели, при задании которого эта ошибка была допущена.

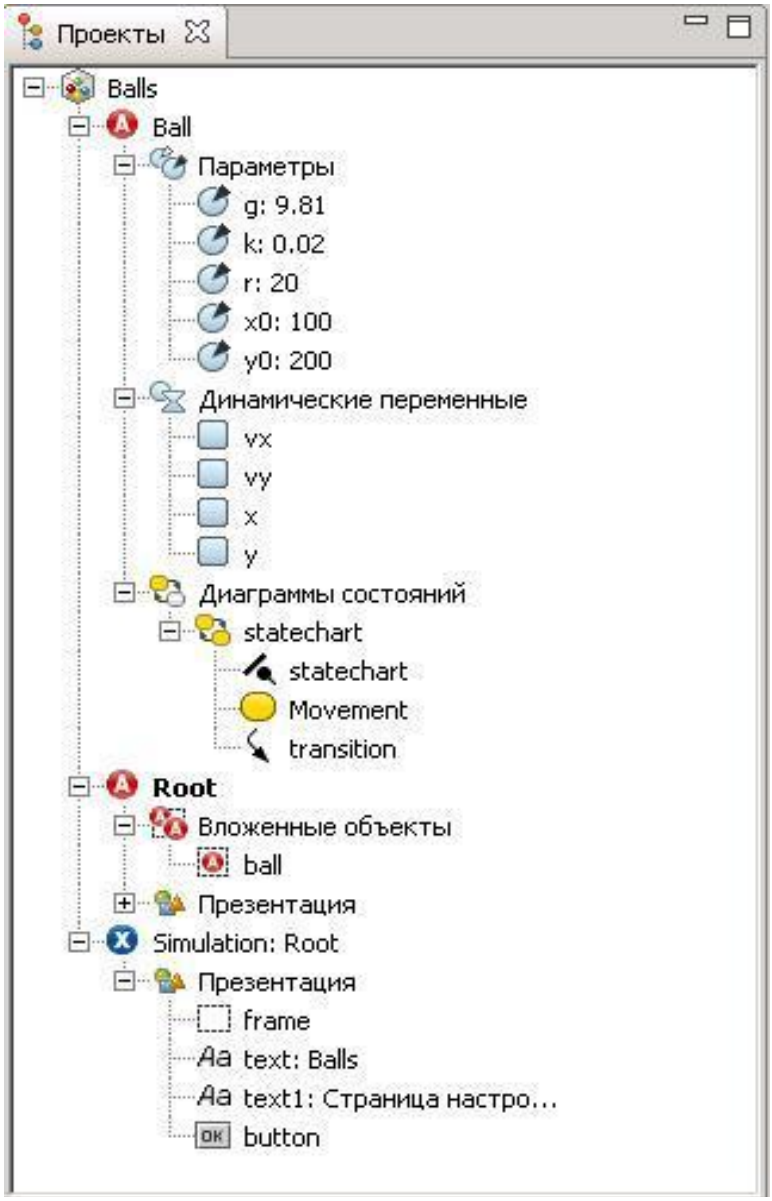

Рис. 1.2

Активный объект является основным структурным элементом модели в AnyLogic. Активным объектом называется сущность, которая включает в себя данные, функции и поведение как единое целое. Активный объект строится как класс, который может включать в качестве составных элементов экземпляры других классов активных объектов. Наш проект Balls включает два класса активных объектов: класс Ball и класс Root. На дереве проекта (рис. 1.2) как составные элементы класса Ball показаны Параметры, Динамические переменные и Диаграммы состояний, у класса *Root* составными его элементами показаны Вложенные объекты и Презентация.

Одна из ветвей в дереве проекта имеет название *Simulation*, это эксперимент, который может быть выполнен с моделью. С помощью экспериментов задаются конфигурационные настройки модели. AnyLogic поддерживает несколько типов экспериментов, каждый из которых соответствует своей задаче моделирования. AnyLogic поддерживает следующие типы экспериментов:

- Простой эксперимент
- Варьирование параметров
- Оптимизация
- Сравнение "прогонов"
- Монте-Карло
- Анализ чувствительности
- Калибровка
- Нестандартный

Эксперименты: **Сравнение "прогонов"**, **Монте-Карло**, **Анализ чувствительности, Калибровка** и **Нестандартный** доступны только в AnyLogic Professional.

1.1.1. ДИАГРАММА КЛАССА АКТИВНОГО ОБЪЕКТА

В нашей модели диаграмма класса активного объекта – мяча задается в окне с именем **Ball,** в котором содержатся его переменные (координаты мяча `x`, `y` и его скорости `Vx` и `Vy`), параметры (`g`, `r`, `k`, `x0` и `y0`) и диаграмма состояний с именем `statechart` (рис. 1.3). AnyLogic отображает получившиеся зависимости между переменными с помощью тонких голубых стрелок. Стрелка, направленная от переменной `Vx` к переменной `x` означает, что переменная `Vx` упоминается в формуле переменной `x`. Стрелки зависимостей рисуются автоматически, всегда синхронизированы с формулами переменных и автоматически появляются или исчезают на диаграмме, как только переменная появится в формуле или будет исключена. Для улучшения наглядности можно редактировать внешний вид стрелок зависимостей, а именно изменять их цвет и радиус закругления.

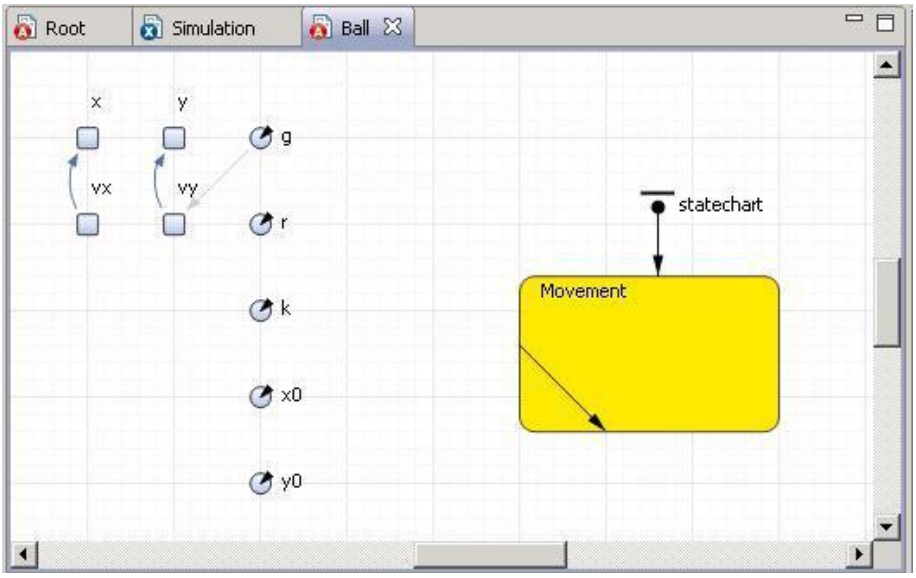

Рис. 1.3

Структура корневого активного объекта `Root` задана в окне с именем **Root**. В модели (рис. 1.1) активный объект `Root` содержит иконку с именем `ball` – один экземпляр активного объекта `Ball` и анимацию.

1.1.2. ПАНЕЛЬ СВОЙСТВ ОБЪЕКТОВ МОДЕЛИ

Каждый элемент модели обладает теми или иными свойствами или параметрами. При выделении какого-либо элемента в панели **Проекта** или **Диаграммы** внизу появляется окно свойств именно этого выделенного элемента. Окно **Свойства** (рис. 1.4) используется для просмотра и изменения свойств элементов.

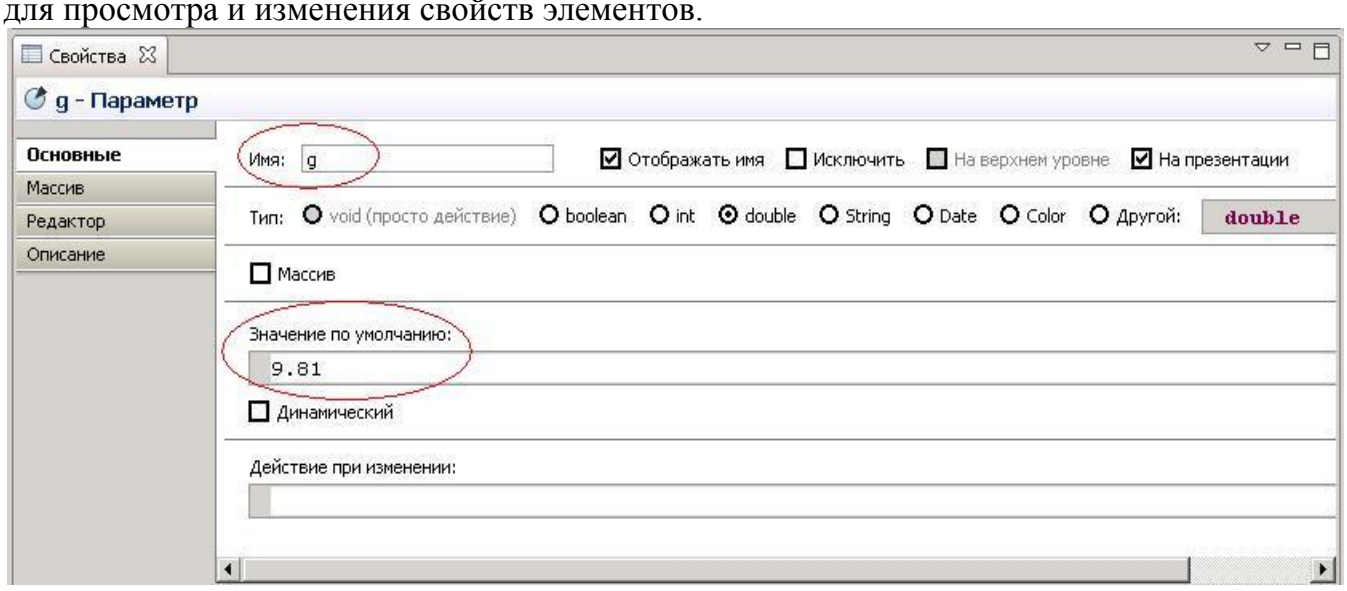

Рис. 1.4

Например, при выборе параметра `g` в окне редактора диаграммы объекта `Ball` внизу в панели свойств появятся 4 вкладки: **Общие, Массив, Редактор** и **Описание**. На вкладке **Общие** кроме имени этого объекта указаны его параметры: тип (`double`), значение по умолчанию (`9.81`) и отмечено, что следует показывать этот параметр на презентации и отображать его имя на диаграмме объекта `Ball`.

1.1.3. ПОВЕДЕНИЕ АКТИВНОГО ОБЪЕКТА

Поведение мяча представлено в стейтчарте, рис. 1.3, который состоит из Начала диаграммы состояний, одного состояния с именем `Movement` и одного перехода. Переход срабатывает при выполнении условия касания поверхности земли при движении мяча вниз, которое можно записать выражением:

```
y      <= r && Vy < 0
```

Когда вертикальная координата `y` центра мяча с радиусом `r` будет отстоять от поверхности на `r` и при этом скорость мяча `Vy` будет направлена вниз, переход стейтчарта сработает. Это выражение записано в поле **При выполнении условия** окна свойств перехода (рис. 1.5). При выполнении данного условия мяч отскакивает, то есть его скорость меняет свой знак на противоположный и уменьшается на долю `k`, моделирующую потерю энергии при отскоке. Это отражено в поле **Действие** окна свойств перехода.

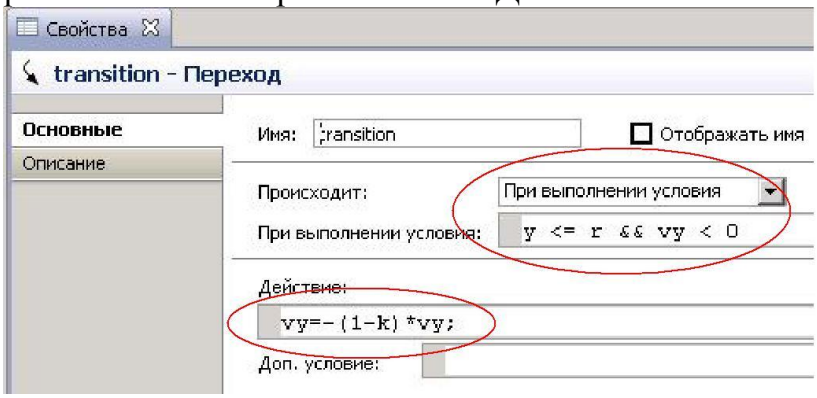

Рис. 1.5

Именно так, стейтчарты следят за событиями. При наступлении нужного события выполняется необходимое действие. И условие наступления события, и само действие, меняющее переменные модели, записываются операторами языка Ява.

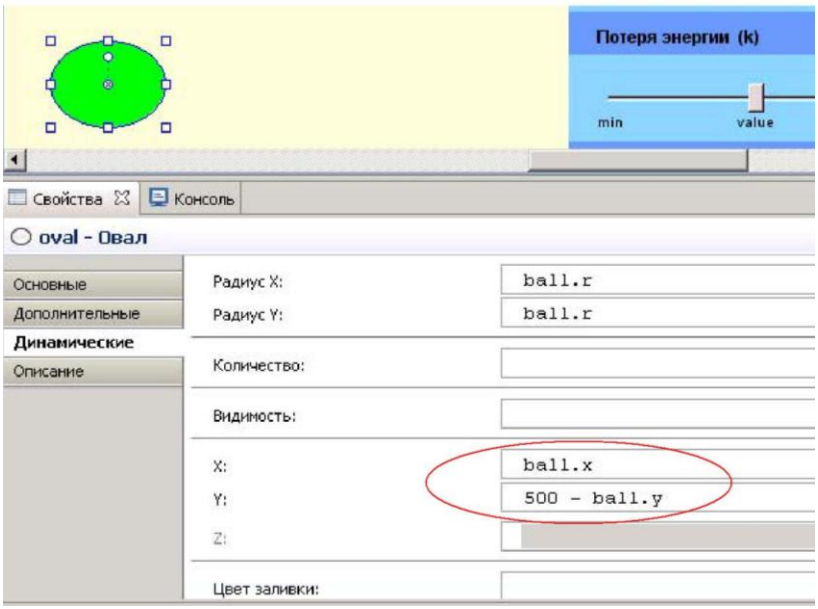

Рис. 1.6

1.1.4. ПРЕЗЕНТАЦИЯ

В нашей модели строится двумерное динамическое представление, которое показывает, что происходит с моделью с течением времени. Для модели `Balls` в диаграмме `Root` построено изображение мяча в виде закрашенной окружности.

Презентация позволяет более наглядно представить динамику моделируемой системы, т. е. поведение ее во времени. Для презентации геометрических фигур, например окружности, их параметры –координаты, радиус, цвет и т.п., связываются с переменными и параметрами модели.

Изменение переменных модели во времени приводит к изменению во времени внешнего вида геометрических фигур, что позволяет наглядно представить динамику моделируемой системы.

На рис. 1.6 показано, что в окне свойств зеленого круга координаты `X` и `Y` его центра и радиус `r` имеют динамические значения, которые связаны с переменными `x`, `y` и `r` экземпляра `ball` класса активного объекта `Ball`. Таким образом, изменение данных переменных будет вызывать перемещение центра и изменение радиуса окружности, моделирующей мяч, при работе модели.

Щелкните мышью на различных анимационных объектах в панели диаграммы `Root`. Вы увидите, что для различных элементов модели окно свойств содержит различные параметры, характеризующие именно данный элемент. Например, щелкните мышью в презентации на слайдере. В окне свойств на вкладке **Основные** будет представлена информация о связи слайдера с конкретным параметром модели и геометрическими параметрами самого слайдера.

Обратите внимание, что в качестве координаты Y мы взяли значение переменной `y` со знаком минус и смещением 500 единиц. Это сделано потому, что ось Y на презентации направлена вниз, а не вверх, и знак минус позволяет нам перевернуть ось Y и опустить ее до нужного уровня.

Итак, модель `Balls` построена и готова к запуску.

1.2. РЕЖИМ ВЫПОЛНЕНИЯ МОДЕЛИ

При запуске модели можно выполнять различные эксперименты с моделью. Рассмотрим основные средства управления экспериментом.

1.2.1. ЗАПУСК МОДЕЛИ

Запуск модели производится кнопкой **Запустить** на панели инструментов.

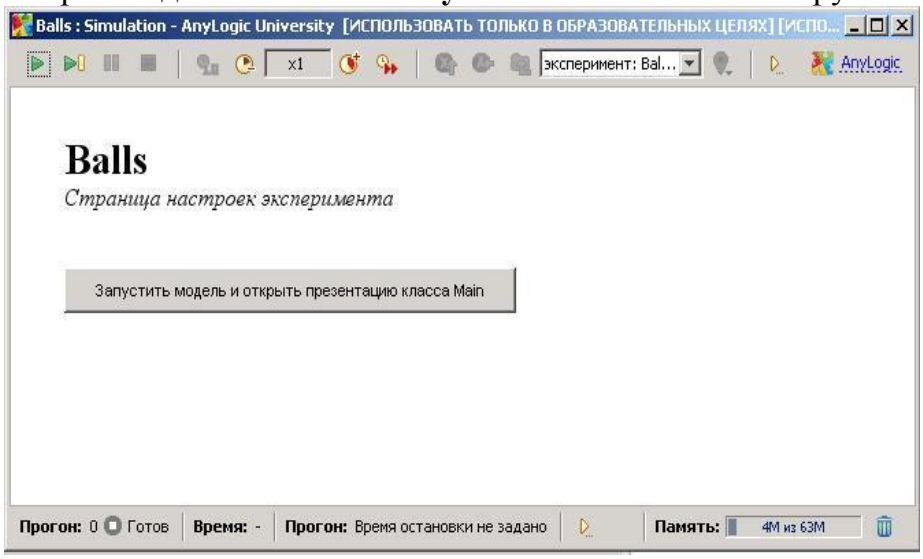

Рис. 1.7

При запуске эксперимента AnyLogic автоматически производит построение запускаемой модели. Поэтому в случае обнаружения ошибки Вам будет показано сообщение об ошибке,а более подробная информация будет выведена в панель Консоль.

При отсутствии ошибок откроется окно презентации эксперимента, рис. 1.7., которое содержит кнопку **Запустить модель и открыть презентацию класса Main**.

Когда Вы запустите модель с помощью этой кнопки, откроется окно презентации либо эксперимента, либо одного из активных объектов запущенной модели, рис. 1.8. На презентации будут видны все элементы, в свойствах которых были установлены флажки **На презентации**.

При проведении компьютерных экспериментов можно использовать все кнопки, показанные в верхней части окна рис. 1.8:

- запуск или продолжение моделирования
- запуск выполнения модели по шагам
- пауза
- останов модели и возврат в окно презентации эксперимента

В нижней части окна виден статус модели (пауза или выполнение, № прогона и др. информация).

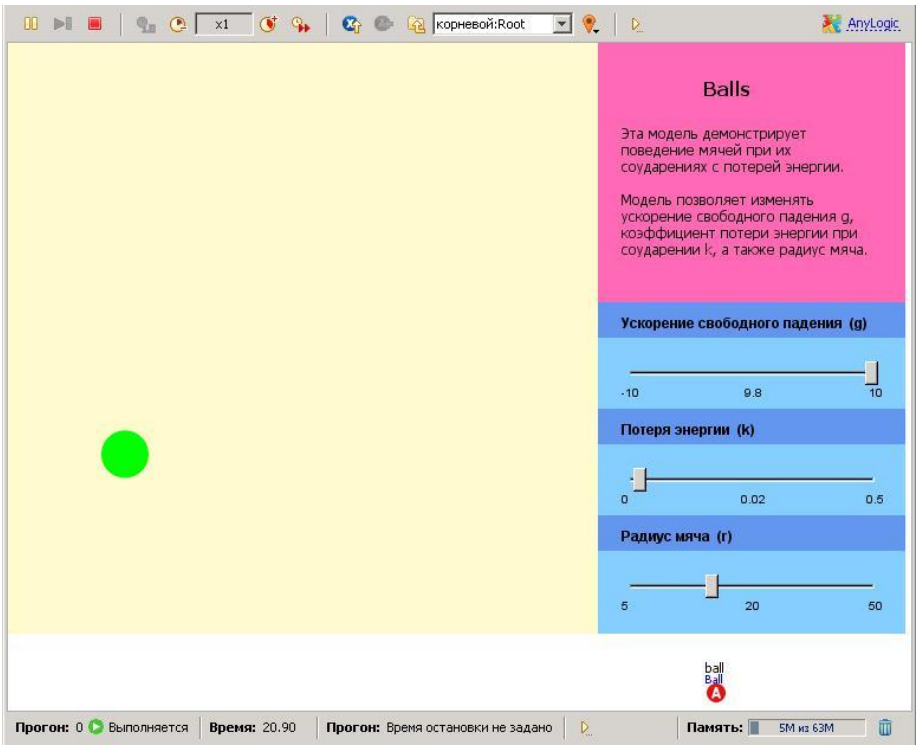

Рис. 1.8

1.2.2. ЭКСПЕРИМЕНТЫ С МОДЕЛЬЮ

На рис. 1.8. кроме движущегося изображения мяча видны текстовый комментарий и "бегунки" или "слайдеры" – подвижные указатели для изменения параметров модели во время ее выполнения . Перемещая бегунки слайдеров, можно менять три параметра – ускорение свободного падения , долю потери скорости мяча при каждом отскоке и радиус мяча. Изменение параметров позволяет исследовать поведение модели в различных условиях – это и есть компьютерный эксперимент.

Проведите несколько экспериментов с моделью, изменяя параметры модели.

1.2.3. УПРАВЛЕНИЕ СКОРОСТЬЮ ВЫПОЛНЕНИЯ МОДЕЛИ И ИЗОБРАЖЕНИЕМ

Модель AnyLogic может выполняться либо в режиме виртуального, либо в режиме реального времени.

В режиме виртуального времени модель выполняется без привязки к физическому времени– она просто выполняется настолько быстро, насколько это возможно. Этот режим лучше всего подходит в том случае, когда требуется моделировать работу системы в течение достаточно длительного периода времени.

В режиме реального времени задается связь модельного времени с физическим, то есть задается количество единиц модельного времени, выполняемых в одну секунду. Это часто требуется, когда Вы хотите, чтобы презентация модели отображалась с той же скоростью, что и в реальной жизни.

Переключение между виртуальным и реальным временем исполнения модели осуществляется кнопкой **Виртуальное/реальное время** панели управления окна презентации, а уменьшение масштаба времени выполняется с помощью двух кнопок **Замедлить**, **Ускорить** и расположенного между ними поля. Это поле указывает коэффициент ускорения модельного времени относительно физического (здесь 1x означает единичный коэффициент ускорения).

Выполните несколько экспериментов с различными скоростями выполнения модели, используя кнопки управления.

1.2.4. НАВИГАЦИЯ ПО МОДЕЛИ

Расположенный в панели управления окна презентации выпадающий список Навигация открывает организованный в виде дерева список объектов модели, обеспечивая простую навигацию по модели и быстрый доступ к любым ее объектам, рис.1.9. Корнем дерева объектов является корневой объект запущенного эксперимента. Если структура модели меняется во время выполнения модели, то эти изменения тут же отображаются в дереве объектов модели.

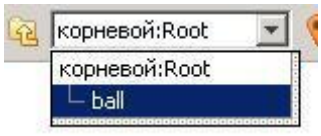

Рис. 1.9

Если выбрать объект `Ball`, то мы увидим его структурную диаграмму с динамично изменяющимися значениями переменных `Y` и `Vy`. AnyLogic поддерживает различные инструменты для сбора, отображения и анализа данных во время выполнения модели. Простейшим способом просмотра текущего значения и истории изменения значений переменной или параметра во время выполнения модели является использование окна **инспекта.** Щелкните мышью по значку переменной в окне презентации. Будет отображено небольшое желтое окно - это и есть окно инспекта, рис. 1.10.

Установите подходящий размер окна путем перетаскивания мышью нижнего правого угла окна инспекта. Если нужно, переместите окно, перетаскивая его мышью за панель названия окна.

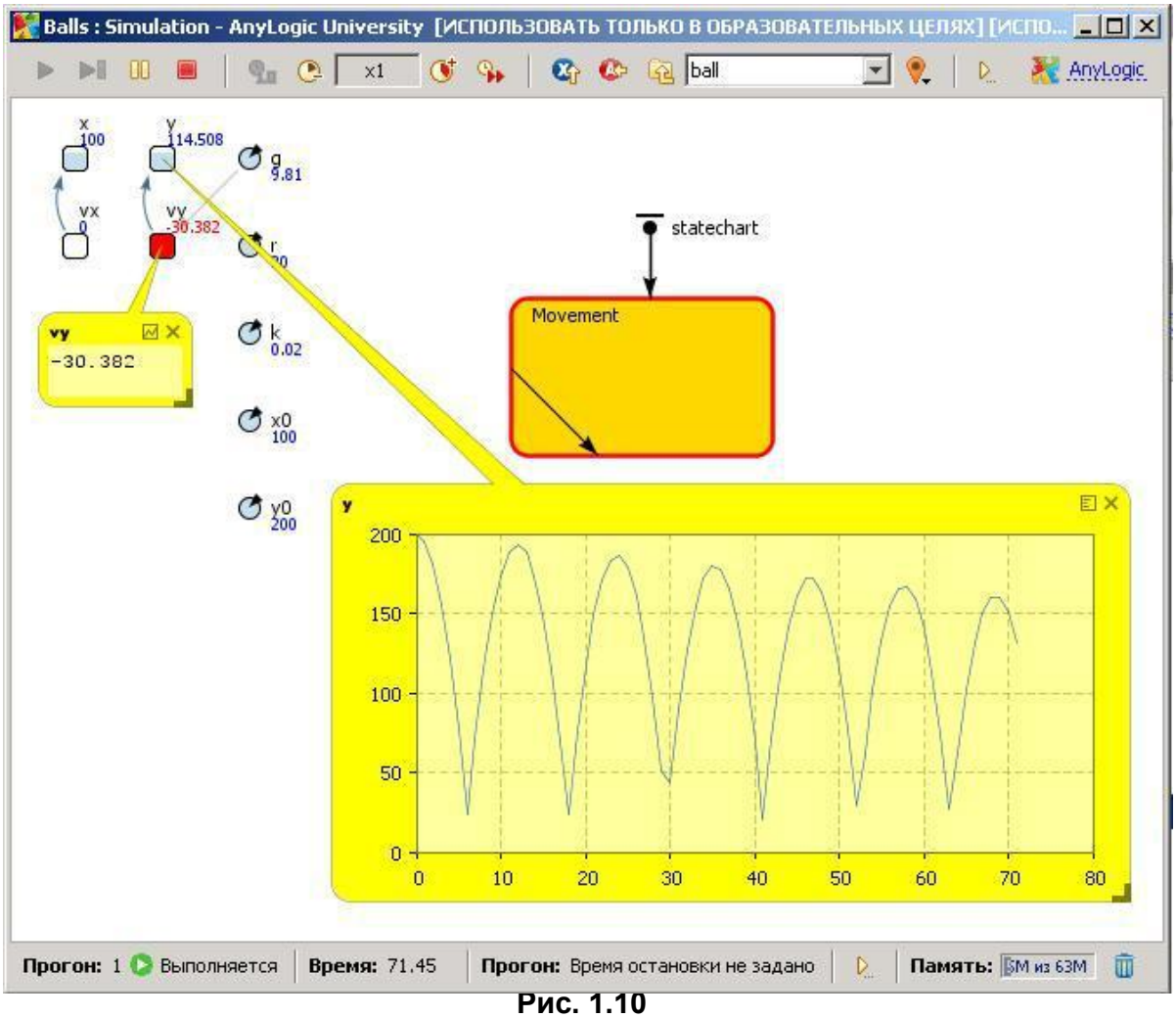

Рис. 1.10

Чтобы переключить окно инспекта в режим графика щелкните мышью по символу графика , находящегося в правом верхнем углу окна инспекта.

ЗАДАНИЕ 2
Доработка модели Balls

2.1. ИЗМЕНЕНИЕ ЦВЕТА МЯЧА ПРИ ОТСКОКЕ

Дополним анимационное представление мяча динамическим цветом, так, чтобы при отскоке его цвет на несколько секунд изменялся на красный. Для этого нужно запомнить момент отскока и установить красный цвет окружности в презентации на небольшой интервал времени, следующий за этим моментом.

Создайте переменную `t0`, которая будет фиксировать момент отскока. Для этого перейдите на диаграмму класса активного объекта `Ball`, затем в панели **Палитра** откройте вкладку **Системная динамика** и перенесите иконку (**Параметр**) на диаграмму. В поле **Имя** открывшегося окна свойств этого параметра введите `t0`, а в поле **Значение по умолчанию** введите `-1` (рис. 2.1). Для того чтобы параметр `t0` фиксировал момент отскока, нужно значение текущего времени в модели при выполнении условия "отскок" запомнить в этом параметре.

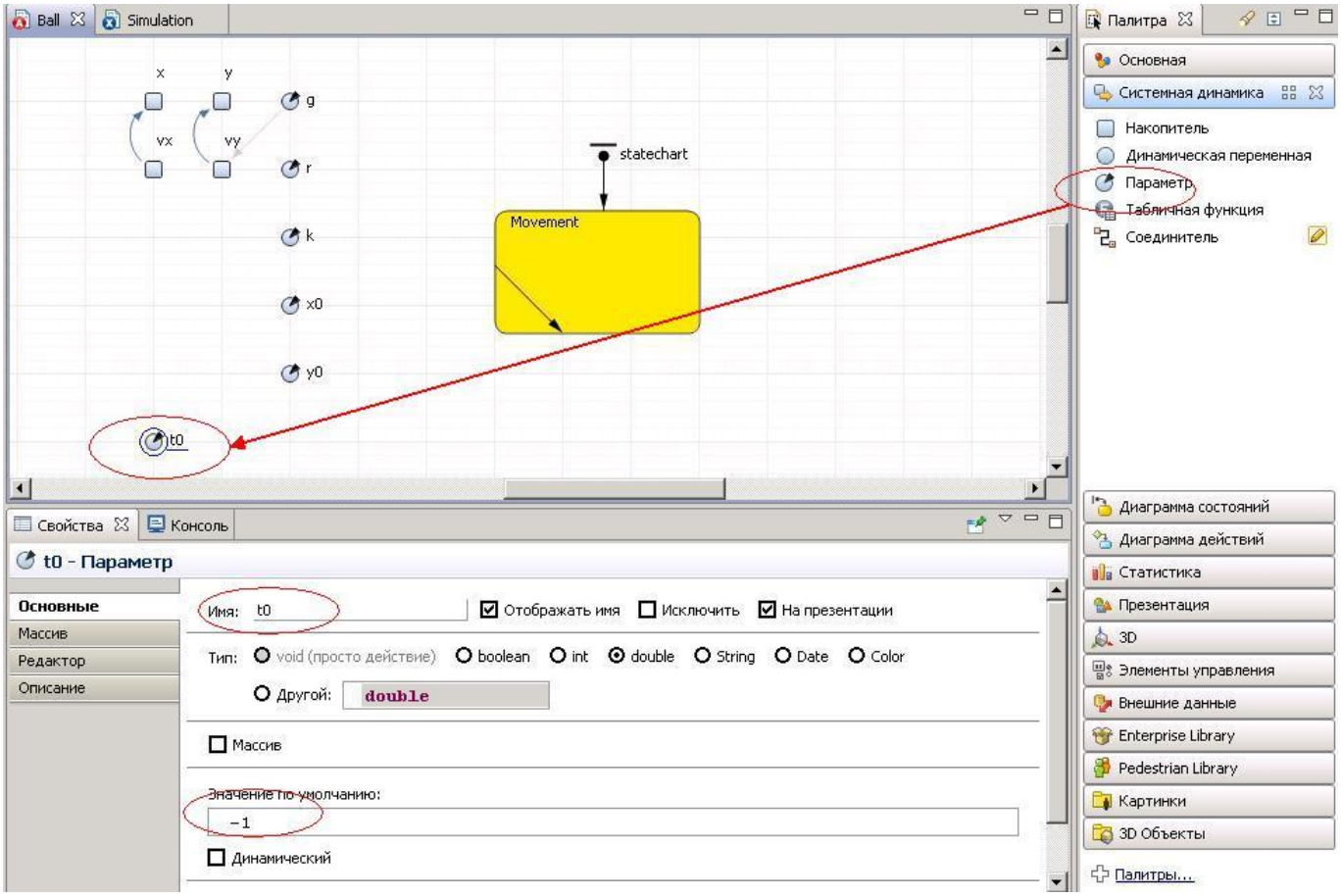

Рис. 2.1

За наступлением данного условия следит стейтчарт, поэтому выделите мышью переход стейтчарта (рис. 2.2), и в поле **Действие** добавьте выражение: `t0 = time();`

При каждом вызове функция `time()` возвращает текущее значение модельного времени.

Параметр `t0` имеет начальное значение *-1*, а при работе модели хранит значение момента времени последнего отскока. Для того чтобы каждый раз при отскоке мяча его цвет изменялся на красный (в течение 0.3 сек.), нужно перейти на диаграмму класса `Root`, выделить зеленый овал (мяч), в панели свойств этого овала открыть вкладку **Внешний вид** и установить в поле **Цвет заливки** динамическое значение цвета (рис.2.3):

```
time()<ball.t0+0.3? red: lime
```

Это условное выражение устанавливает цвет заливки изображения мяча `ball` красным в течение 0.3 сек. после каждого отскока.

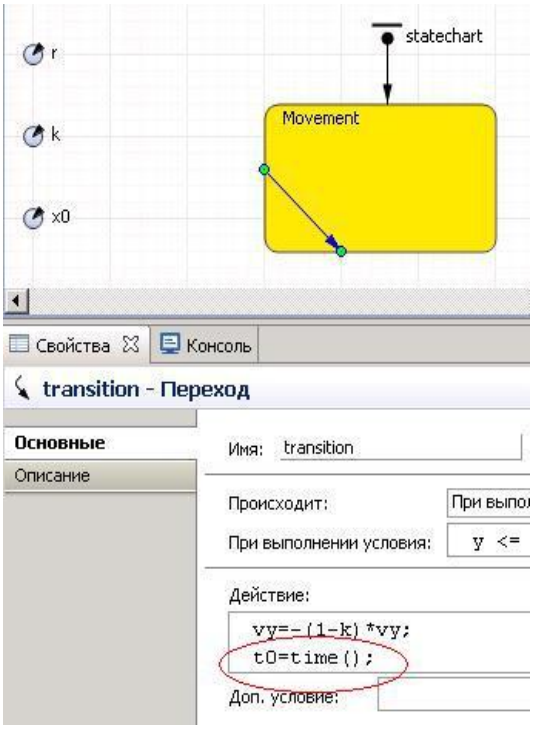

Рис. 2.2

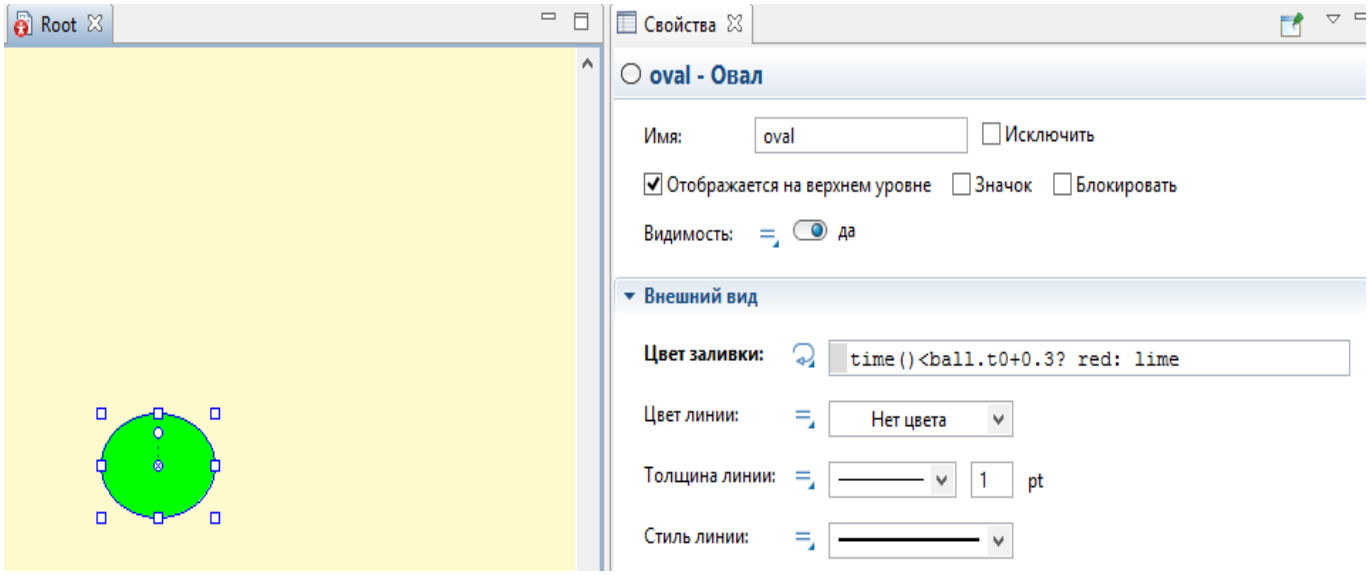

Рис. 2.3

2.2. МОДЕЛЬ С ДВУМЯ МЯЧАМИ

Добавим в модель второй мяч. Перейдите на диаграмму класса активного объекта `Root` и перенесите мышью на него еще один экземпляр мяча. Появившийся объект автоматически получит имя `ball1` (рис. 2.4). При этом в окне свойств нового экземпляра мяча мы увидим те же значения параметров мяча, которые были определены для активного объекта `Ball`.

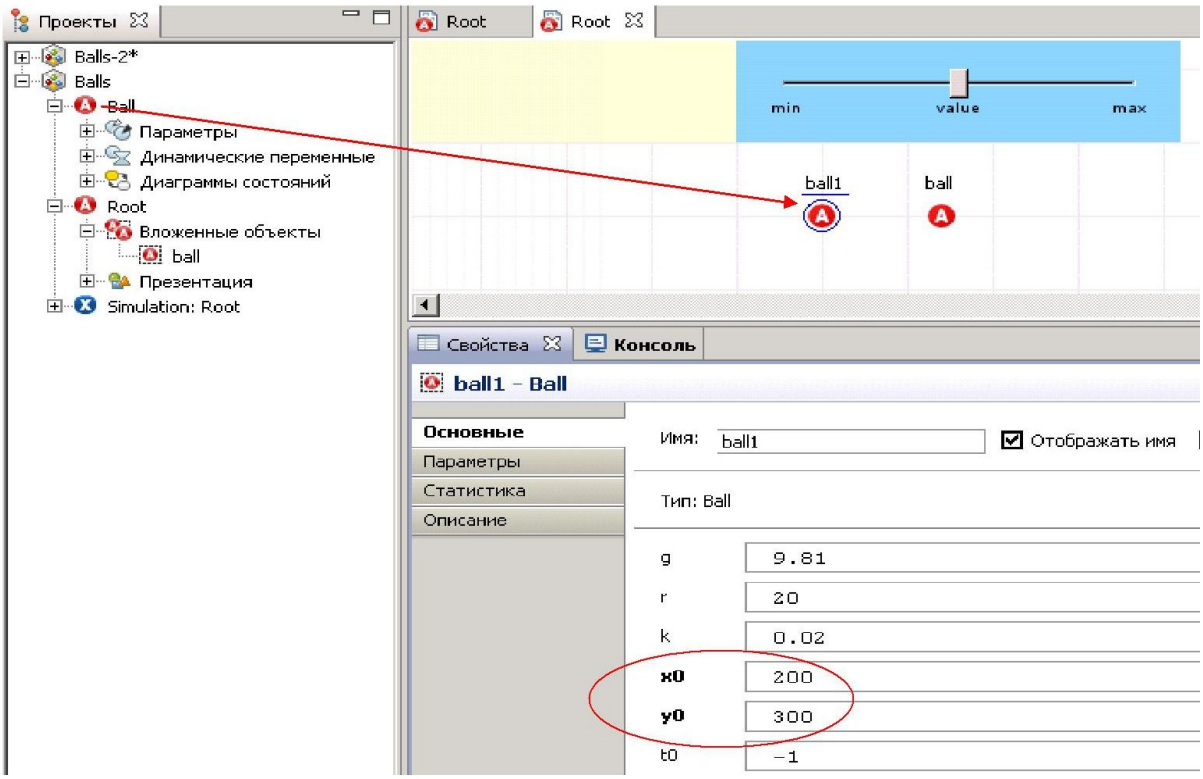

Рис. 2.4

Установите начальные значения `x0` и `y0` нового мяча равными 200 и 300 соответственно.

Чтобы на презентации показать движение второго мяча, продублируйте изображение первого мяча с помощью клавиш <Ctrl+C> и <Ctrl+V>. Параметры нового изображения овала (координаты и цвета) связаны с характеристиками объекта `ball`. Их нужно связать с новым объектом – шаром с именем `Ball1`. То есть, вместо `Ball.x, Ball.y` и `Ball.t0` в соответствующих полях нужно записать `Ball1.x, Ball1.y` и `Ball1.t0` (рис. 2.5). А для значения радиусов **Радиус X** и **Радиус Y** нужно установить `Ball1.r` вместо `Ball.r`.

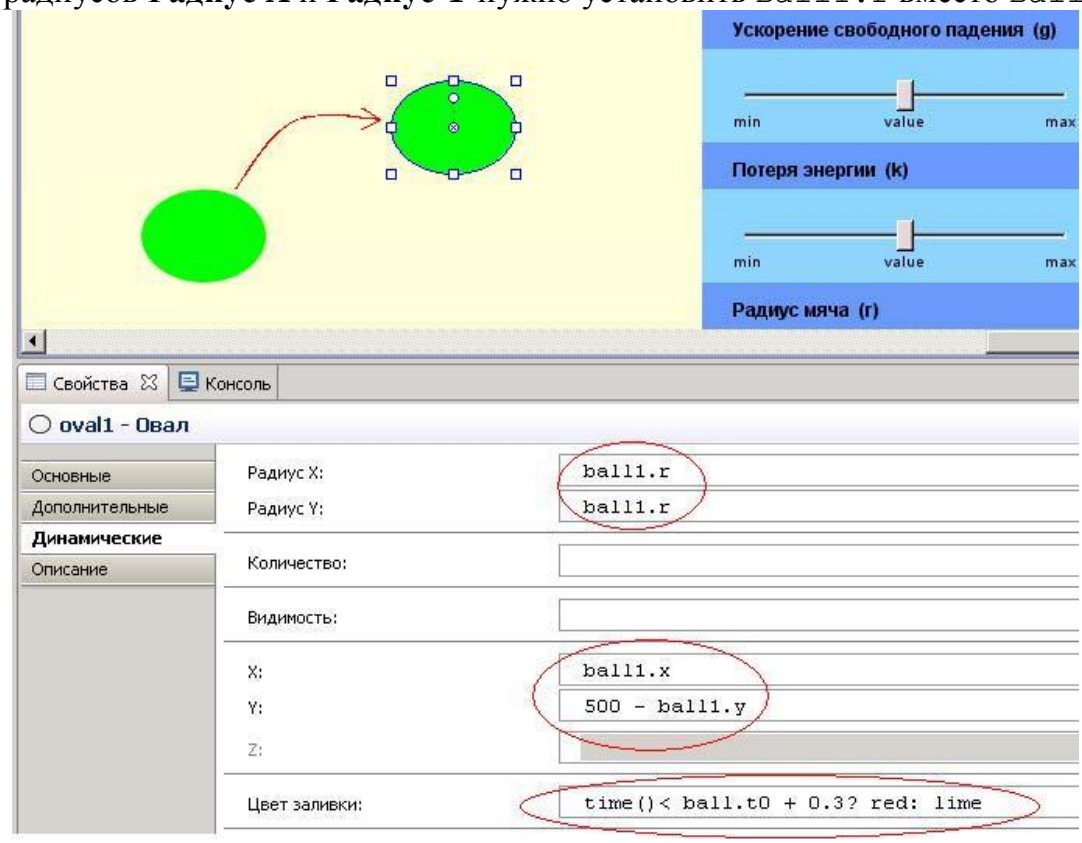

Рис. 2.5

Теперь при запуске модели будут имитироваться независимые движения двух шаров. Продемонстрируйте свою модель преподавателю.

2.3. ПРОИЗВОЛЬНЫЕ ПЕРЕМЕЩЕНИЯ МЯЧА

В нашей модели мячи движутся строго вертикально, отскакивая от горизонтальной поверхности. Это происходит потому, что начальная скорость мячей по координате `x` равна 0. Если мы изменим начальные скорости мячей по `x`, нам необходимо будет описать поведение мячей при столкновении с вертикальными стенками и потолком.

Зададим случайные начальные значения скоростей `Vx` и `Vy`. Для этого перейдите на диаграмму класса активного объекта `Ball`, выделите переменную `Vx` и в поле **Начальное значение** этой переменной замените значение 0 на значение `uniform(-100, 100)`. При этом у различных экземпляров активного объекта `Ball` начальная скорость по координате `x` будет задана случайно из диапазона (-100, +100) метров в секунду. То же самое сделайте для переменной `Vy`.

Для моделирования отскока мяча от потолка нужно на переходе стейтчарта изменить условие столкновения мяча с поверхностью. Мячи двигаются в пространстве, размером 500x500 метров. В поле **При выполнении условия** панели свойств перехода стейтчарта активного объекта `Ball` выражение:

```
y <= r && vy < 0
```

измените на:

```
y <= r && vy < 0 || y >= 500 - r && vy > 0
```

При этом, выполняемое действие должно остаться без изменения, а именно: смена направления скорости `Vy` с частичной ее потерей.

Для того чтобы мяч отскакивал от вертикальных стен, нужно записать это условие в стейтчарте добавлением дополнительного перехода. Откройте палитру **Диаграмма состояний** и сделайте двойной щелчок мышью по иконке **Переход** (рис. 2.6), включив, тем самым, режим рисования (изображение карандашика). Нарисуйте переход внутри состояния `Movement`, как показано на рис. 2.6.

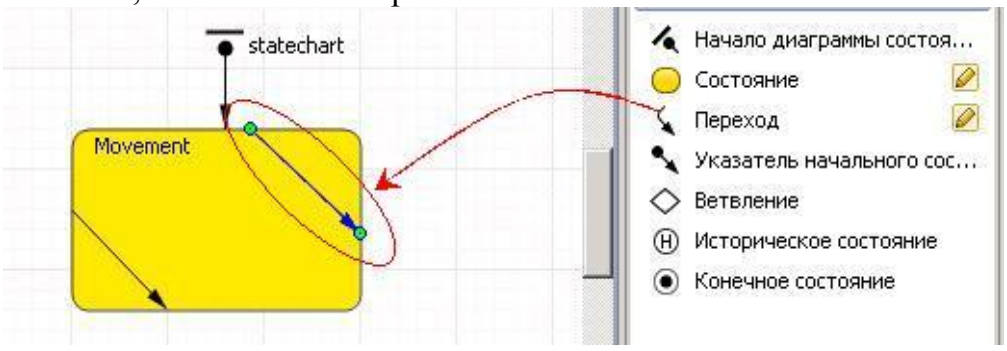

Рис. 2.6

В окне свойств этого перехода в поле **Происходит** нужно выбрать вариант `При выполнении условия`, в поле **При выполнении условия** следует записать условие касания мяча о вертикальную стенку:

```
x <= r && vx < 0 || x >= 500 - r && vx > 0
```

а в поле **Действие** записать изменение направления составляющей `Vx` скорости мяча и запомнить момент времени, когда произошло касание стенки для последующего изменения цвета мяча, рис. 2.7:

```
vx = -(1 - k) * vx;

t0 = time();
```

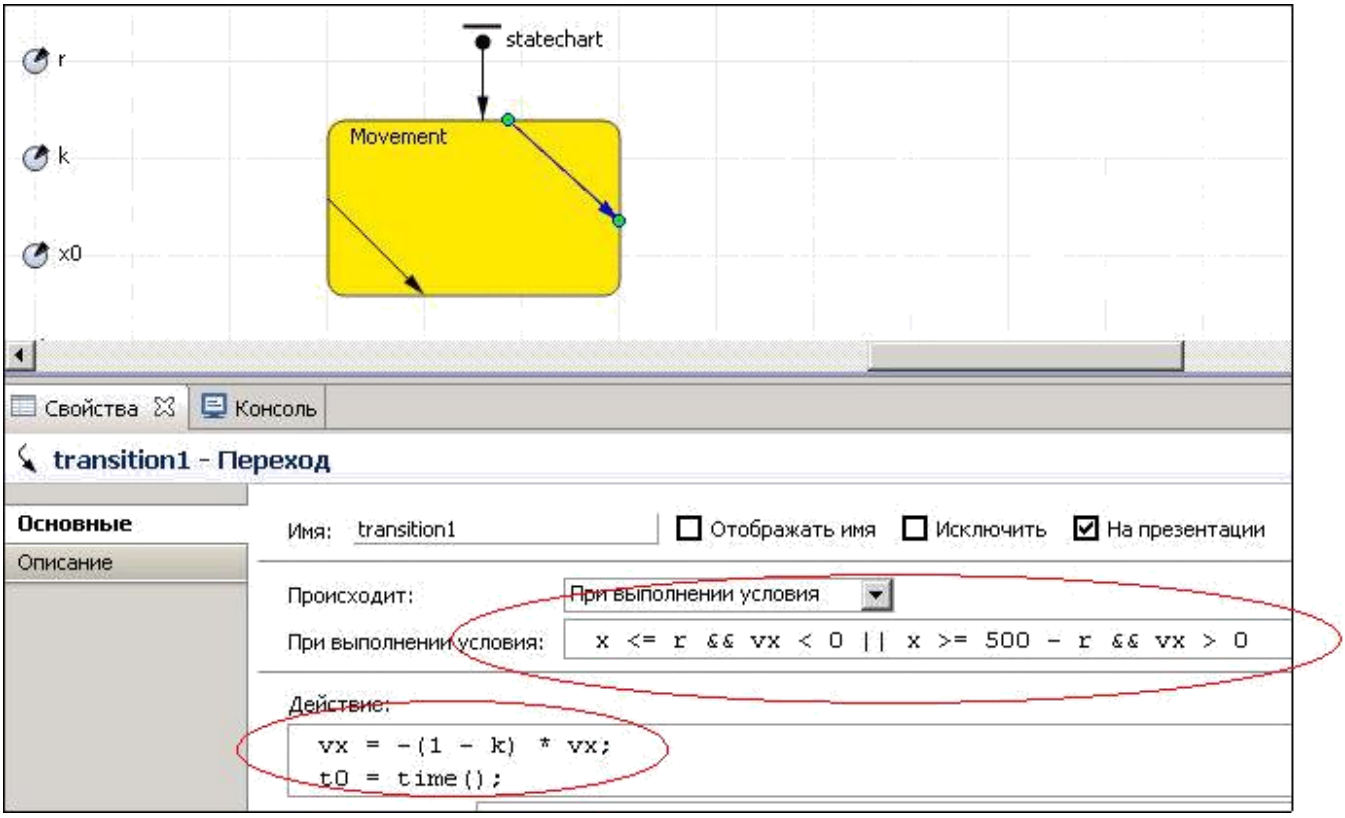

Рис. 2.7

Запустите модель. Поэкспериментируйте с ней, используя слайдеры. Продемонстрируйте модель преподавателю.