

Программирование виртуальной реальности с Alice и Java

Описание курса

- ▣ Обучение программированию в контексте анимации, симуляции, повествования и построения мини-игр.
- ▣ Обучение основам программирования в Alice
 - Быстрое средство прототипирования
 - Создание «черновиков» анимаций, игровых симуляций и фильмоподобных историй

Инструменты

- Программное обеспечение доступно онлайн, бесплатно
 - Alice 3.1
 - NetBeans 6.9.1 и позже
 - Java 1.6 и позже
 - Alice 3 плагин для NetBeans

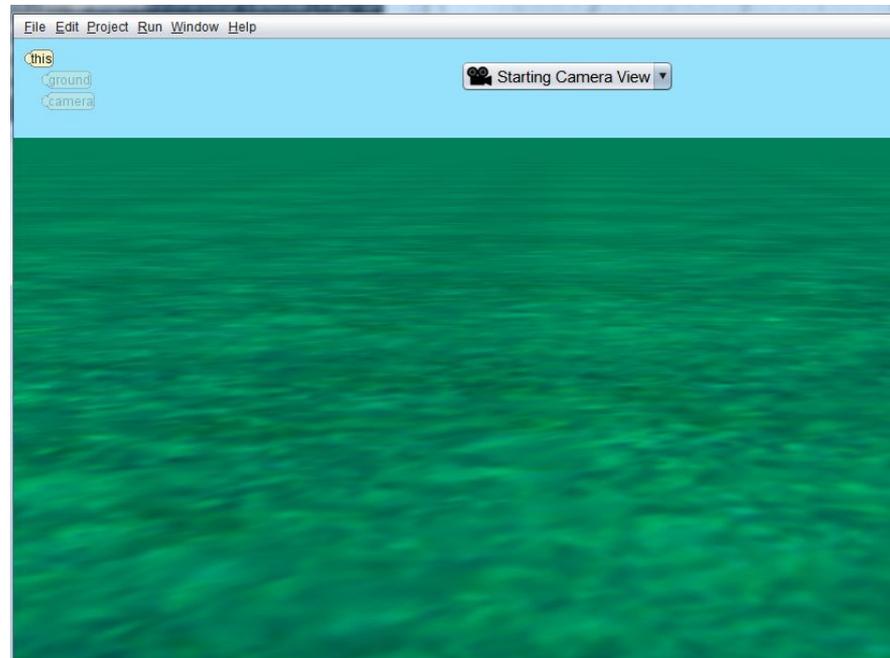
Online help/instructions

- Все необходимые материалы можно найти на сайте alice.org, в том числе, как
 - [Download and install Alice 3](#)
 - [Download and install Netbeans and Java](#)
 - [Download and install the Alice 3 plugin for Netbeans](#)

Классы и объекты

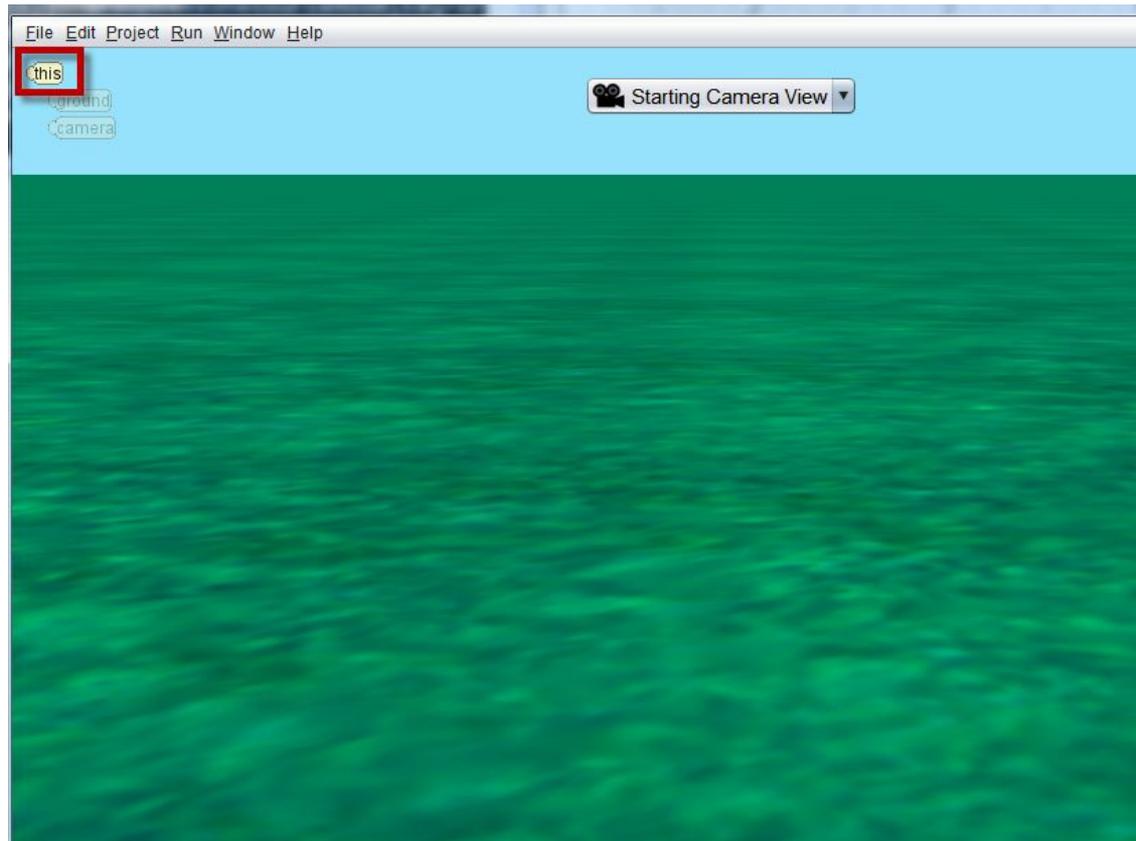
Alice World

Проект Alice создает виртуальный мир.
Первичный компонент мира – это сцена,
показанная в Редакторе Сцены (Scene Editor).



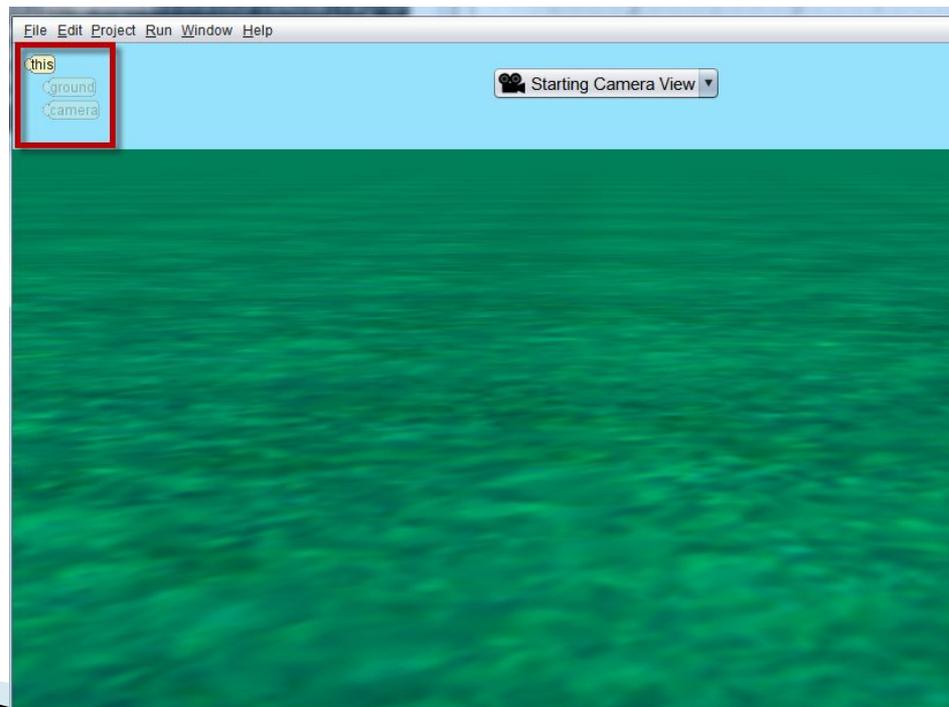
this

- Текущая сцена известна как “this” сцена



Компоненты сцены

- У сцены всегда есть камера и поверхность, которая может быть травой, песком, скалами или даже водой. Другие объекты могут быть добавлены к сцене.



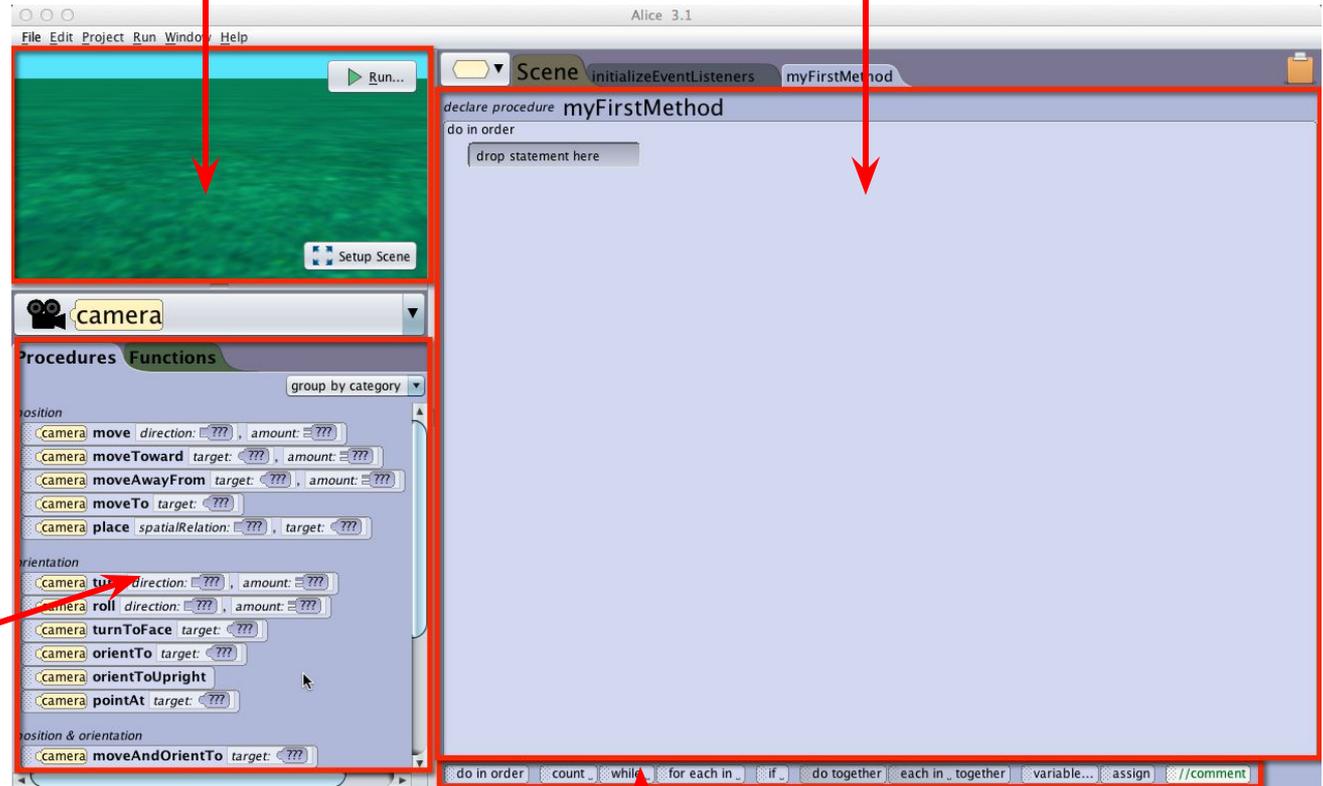
Редактор кода

Scene view

Code editor

*Где пишется
программа
анимации или
игры*

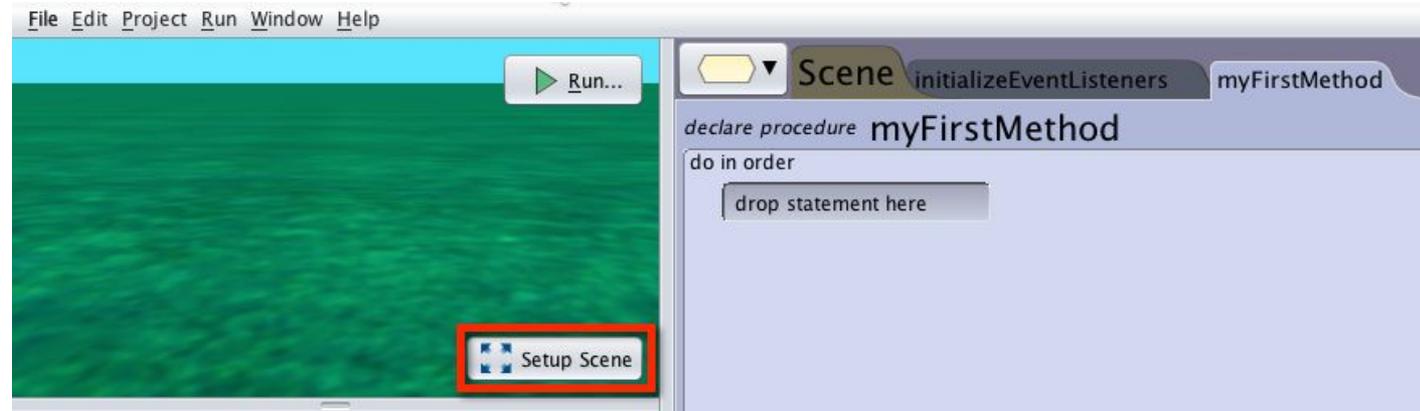
Methods panel



Control tiles

Как перейти к редактору сцены

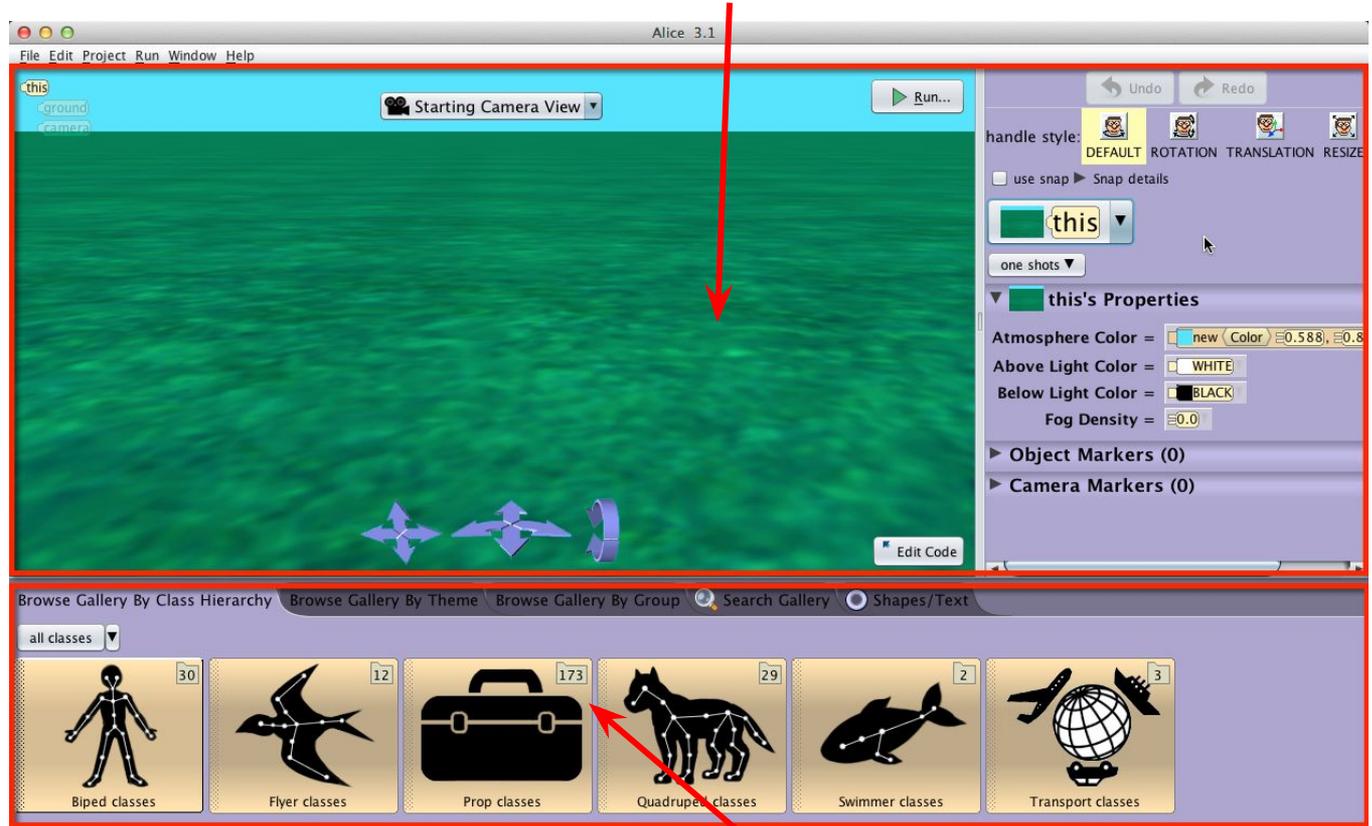
Нажать на
Setup Scene
чтобы
перейти к
**Scene
Editor**



Scene Editor

*Где создается
сцена для
анимации или
игры*

Scene setup panel



Gallery panel, организованная как иерархия классов

Классы

- Каждая 3D модель это заранее созданный класс
- В Alice класс определяет
 - План для создания нового объекта в сцене Alice
 - Действия, которые может выполнять объект этого класса



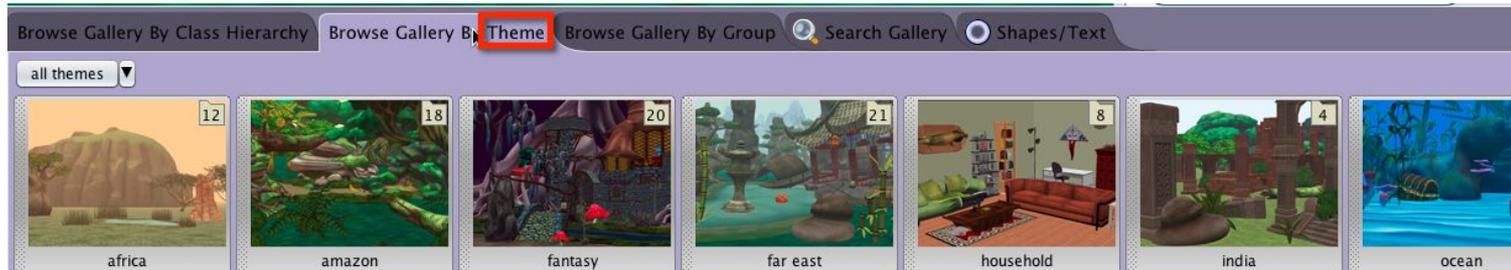
Объекты в Alice

- Экземпляр класса – это объект
- Следующие слайды иллюстрируют, как создать новый объект в сцене Alice.



Галерея

- Разделы галереи содержат классы, упорядоченные по разным признакам



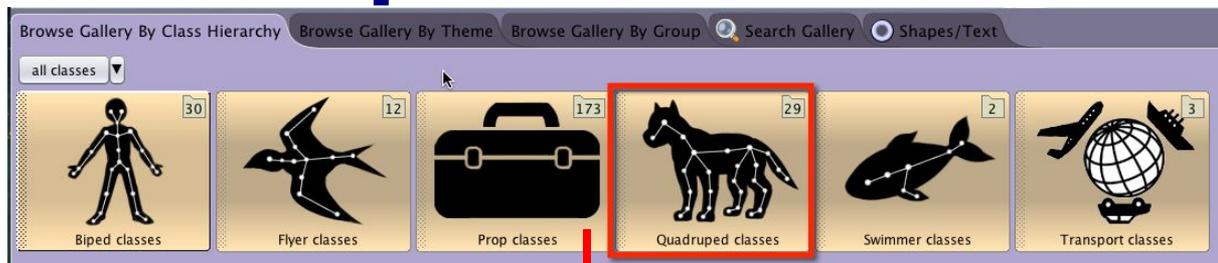
Иерархия классов



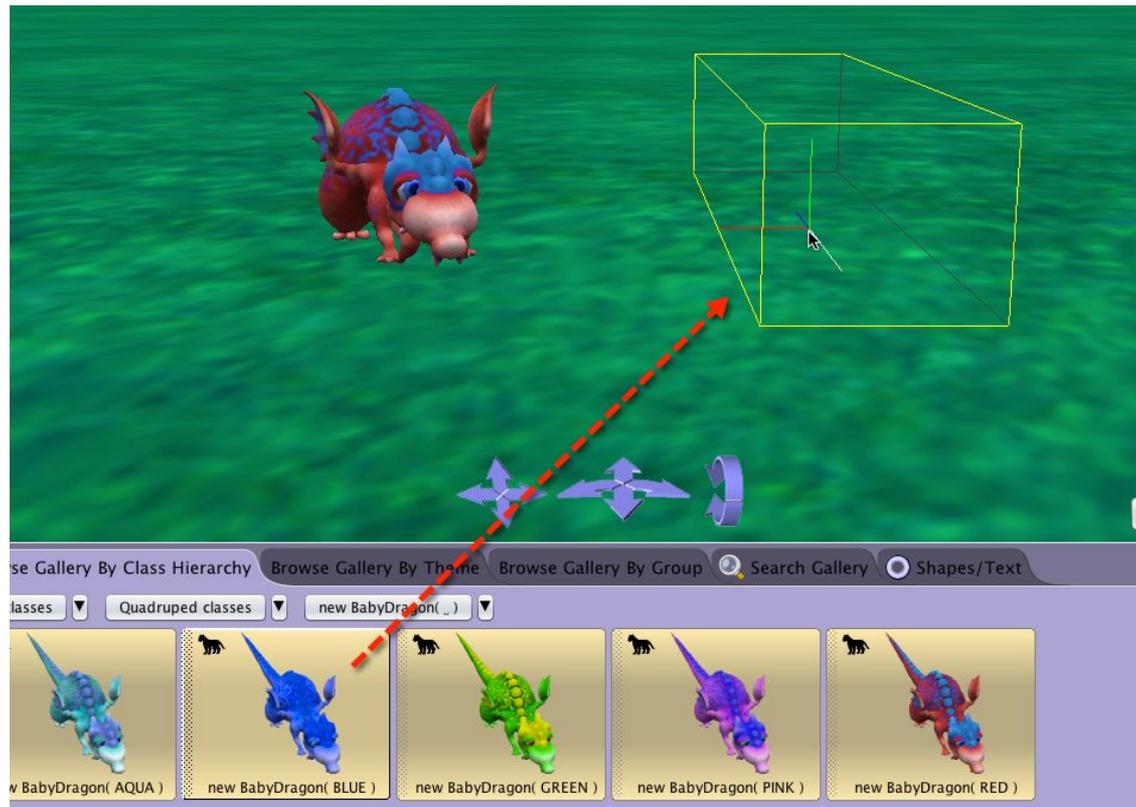
- Здесь классы организованы согласно способу передвижения их объектов.



Выбор класса



Добавление объекта



Именованіе об'єкта



Instance/object

- Мы создали новый экземпляр класса DragonBaby и идентифицировали его именем *fergie*.
- Говорят, что экземпляр класса это объект.



Объекты одного класса

- Сцена может содержать несколько объектов одного класса.
- Пример: *fergie* и *bert* это два различных объекта (экземпляра) одного и того же класса `BabyDragon`. У различных объектов должны быть разные имена.



Панель методов & Дерево объектов

Редактор Сцены – Дерево Объектов

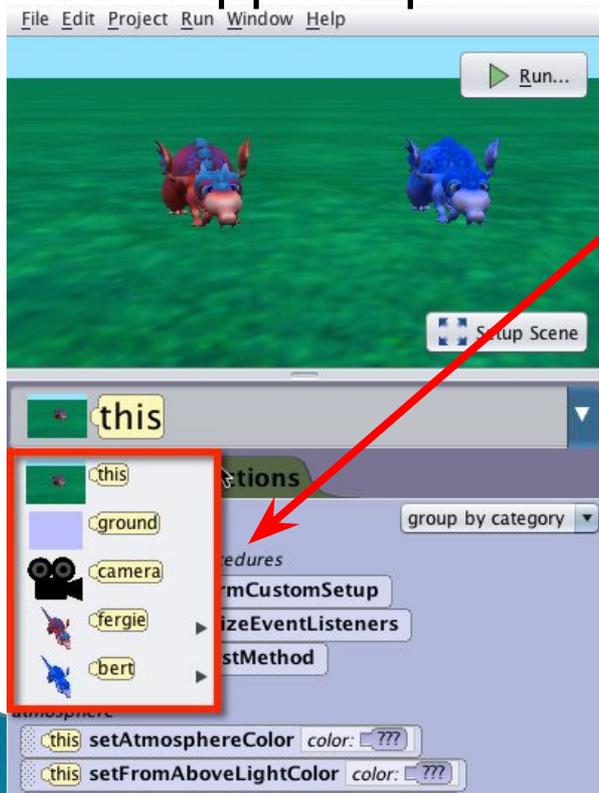
В редакторе сцены Alice генерирует дерево объектов, чтобы показывать список всех объектов в этой сцене.



Чтобы
вернуться в
Редактор
кода,
нажимаем
Edit Code

Редактор Кода – Список Объектов

В редакторе кода Alice генерирует соответствующий список объектов в выпадающем меню.



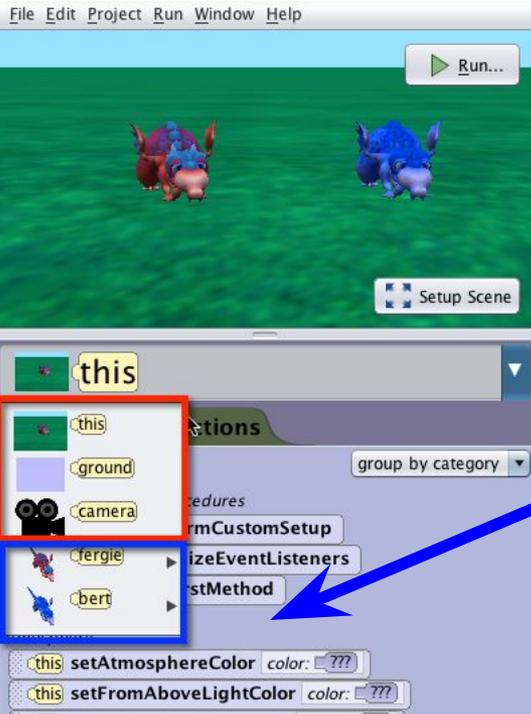
Object tree



Object menu

Alice World (Мир)

- Каждый мир Alice содержит сцену (this), поверхность и камеру.
- Объекты, создаваемые в редакторе сцены, добавляются в список.



Every scene

In this scene



Панель методов

- В редакторе кода панель методов содержит набор плиток (tiles).
- Каждая плитка описывает действие, которое выполняет объект или выполняется над объектом.

Methods panel



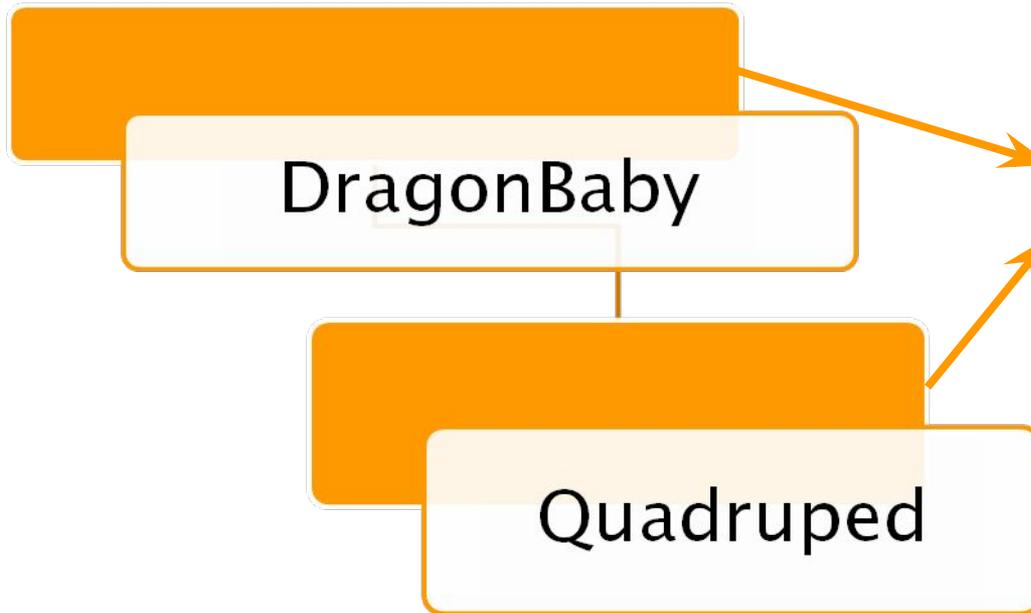
Предки

- Каждый объект в сцене это экземпляр класса, который принадлежит семейству классов.
- Предки класса показаны в панели методов.

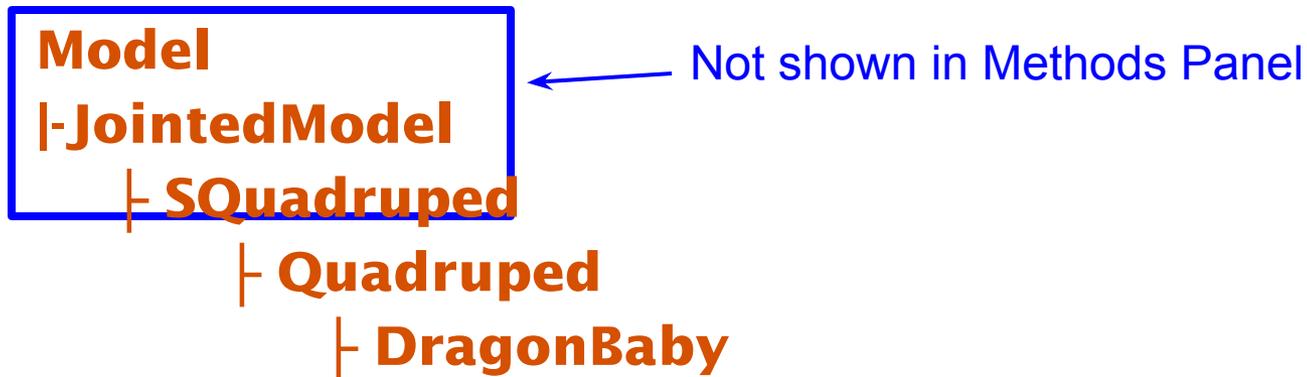


Наследование: Порядок

- Предки могут быть упорядочены от специфических до наиболее общих. Список предков говорит нам, что fergie это экземпляр класса DragonBaby, наследует все свойства и методы класса Quadruped.



Иерархия классов

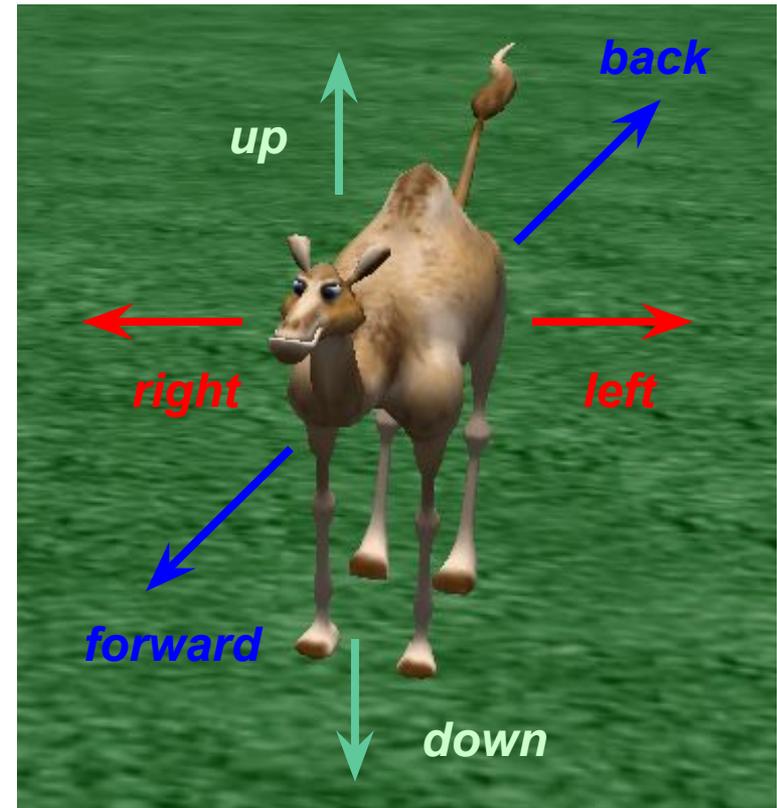


Движение и вращение

3-мерное пространство

□ Объект Alice

- Расположен в 3D мире в точке с координатами (x, y, z)
- Имеет 3 измерения:
 - высота, ширина, глубина
- Может двигаться в 6 направлениях

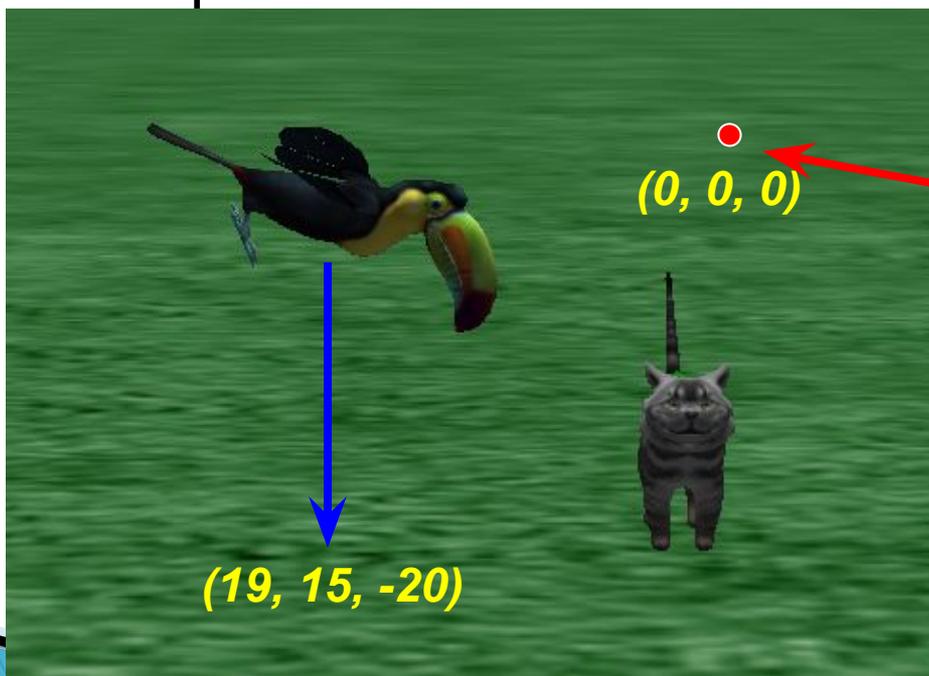


- Центр объекта находится в его опорной точке, вокруг которой он двигается и вращается
- Центр объекта обычно там, где он стоит на земле



Где «находится» объект

- Один из способов описать, где «находится» объект
 - Позиция – относительно начала координат поверхности



Где «находится» объект

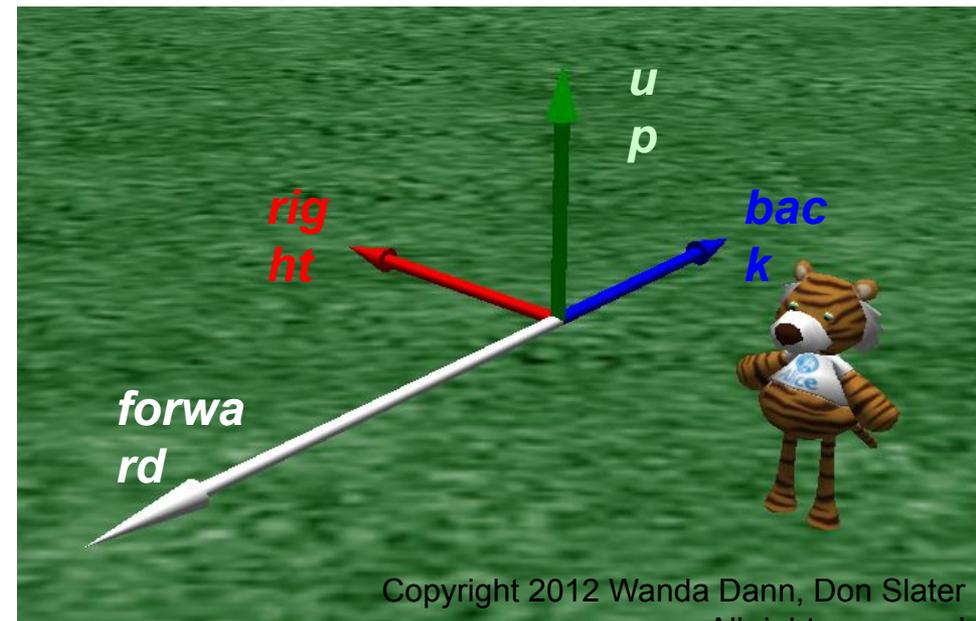
□ Второй способ

- Ориентация – знание объекта о его собственных направлениях «вверх» и «вперед»



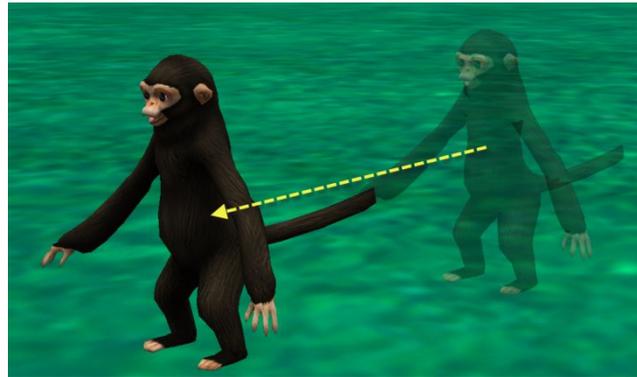
Поступательное движение

- Направление поступательного движения определяется относительно ориентации объекта.
 - Шесть возможных направлений движения
 - **Влево**
 - **вправо**
 - **вверх**
 - **вниз**
 - **вперед**
 - **назад**



Пример: движение вперед

Движение вперед – новое положение объекта «перед» его предыдущим положением относительно собственного направления «вперед» объекта. Координаты объекта изменяются.



Вращательное движение: целый объект

- Вращательное движение меняет ориентацию объекта в мире
- При этом не меняется позиция объекта
- Центр объекта служит опорной точкой для вращения.
- Два вида вращательного движения:
 - **поворот**
 - Вперед, назад
 - Влево, вправо
 - **вращение**
 - Влево, вправо

Пример: Вращение вправо

Вращение вправо – новая позиция объекта вращается вправо от его исходной позиции. Расположение объекта (x,y,z) не меняется. Его направление «вверх» меняется, а «вперед» не меняется.



Пример: поворот направо

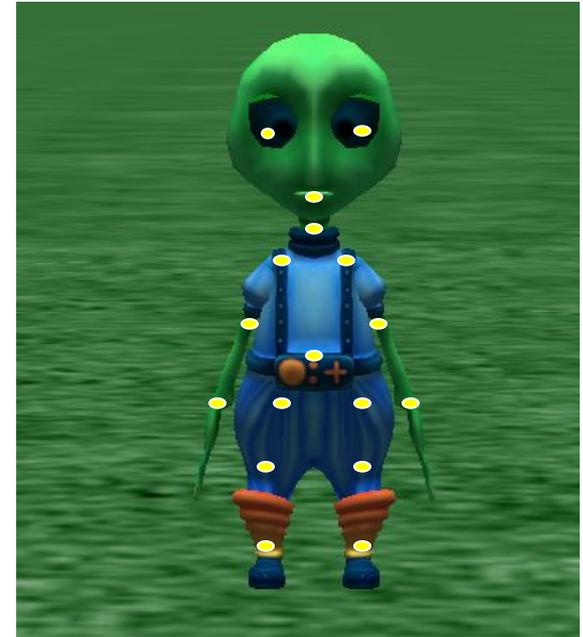
Поворот направо – меняется направление «вперед» объекта, «вверх» остается прежним. Позиция (x,y,z) не меняется.



Движение сочленения



- У объектов часто (не всегда) есть внутренние сочленения (шарниры), которые можно анимировать для управления частями модели.
- Сочленение соединяет часть тела с остальным телом.
- Сочленение можно поворачивать или вращать, но нельзя двигать.

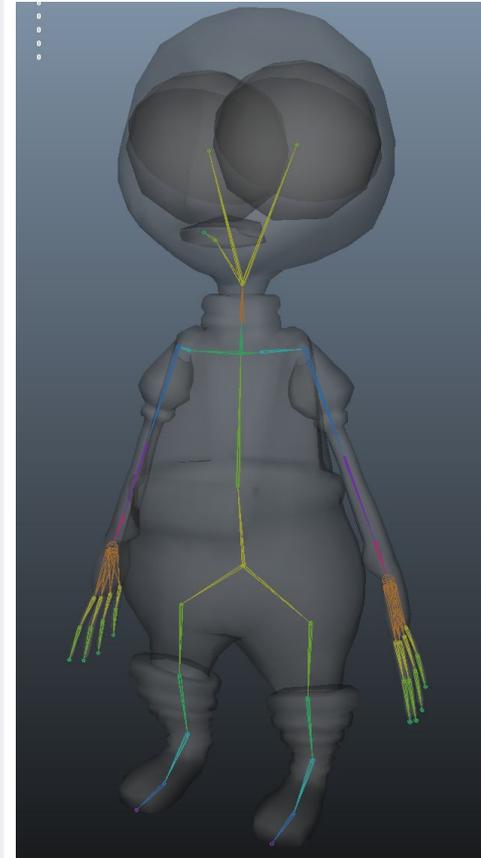


Рентген двуногого



Biped Joints:

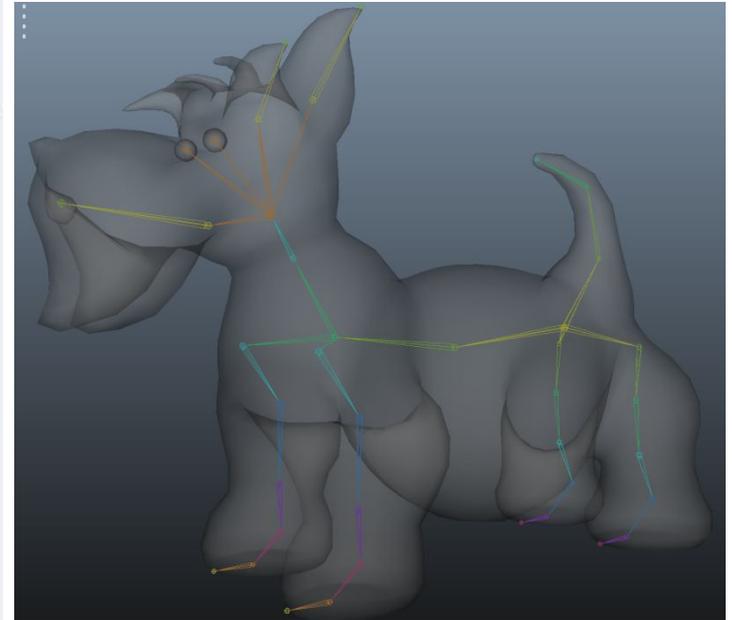
-  alien getPelvisForLowerBody
-  alien getPelvisForUpperBody
-  alien getSpineMiddle
-  alien getSpineUpper
-  alien getNeck
-  alien getHead
-  alien getMouth
-  alien getRightEye
-  alien getLeftEye
-  alien getRightHip
-  alien getRightKnee
-  alien getRightAnkle
-  alien getLeftHip
-  alien getLeftKnee
-  alien getLeftAnkle
-  alien getRightClavicle
-  alien getRightShoulder
-  alien getRightElbow
-  alien getRightWrist
-  alien getLeftClavicle
-  alien getLeftShoulder
-  alien getLeftElbow
-  alien getLeftWrist



Рентген четвероногого



-  scotty getFrontLeftShoulder
-  scotty getFrontLeftKnee
-  scotty getFrontLeftAnkle
-  scotty getFrontLeftBall
-  scotty getFrontLeftToe
-  scotty getFrontRightShoulder
-  scotty getFrontRightKnee
-  scotty getFrontRightAnkle
-  scotty getFrontRightBall
-  scotty getFrontRightToe
-  scotty getPelvisLowerBody
-  scotty getTail2
-  scotty getBackLeftHip
-  scotty getBackLeftKnee
-  scotty getBackLeftAnkle
-  scotty getBackLeftBall
-  scotty getBackLeftToe
-  scotty getBackRightHip
-  scotty getBackRightKnee
-  scotty getBackRightAnkle
-  scotty getBackRightBall
-  scotty getBackRightToe
-  scotty getTail



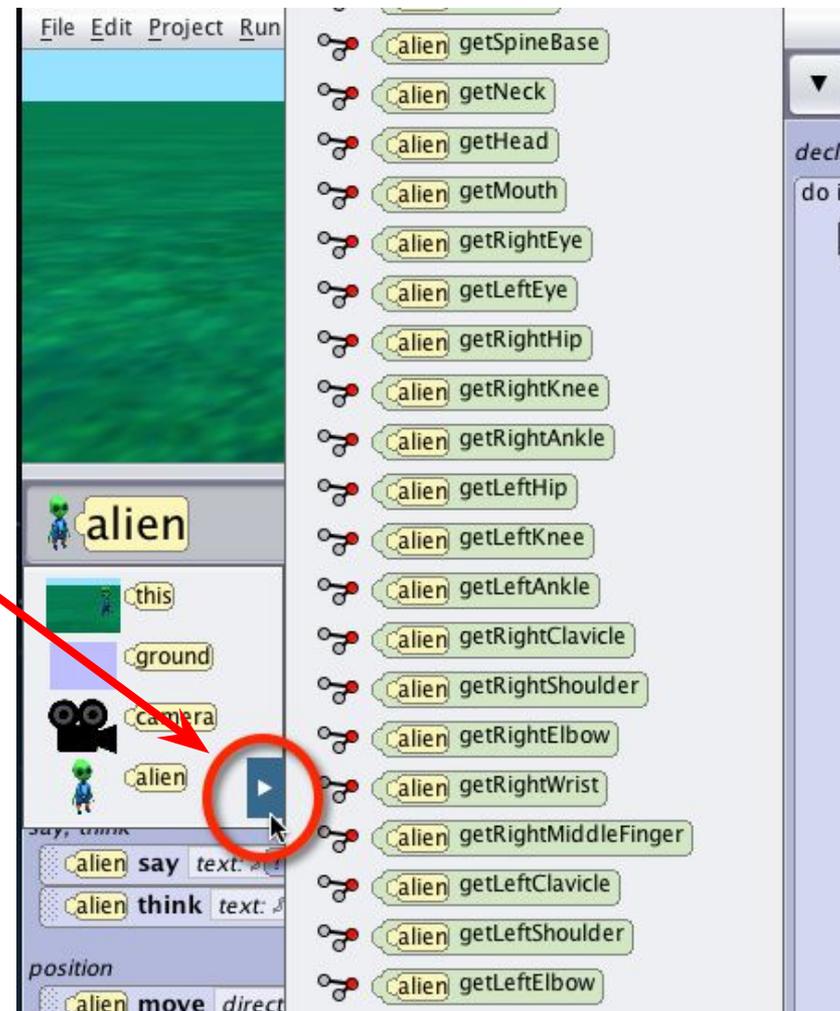
Соединенные части

- Часть может быть соединена с другой частью
- Вращение одной части может также приводить к повороту других частей
- Пример:
 - Поворот левого плечевого сочленения приведет к повороту всей руки инопланетянина, включая предплечье, запястье, кисть и т.д.



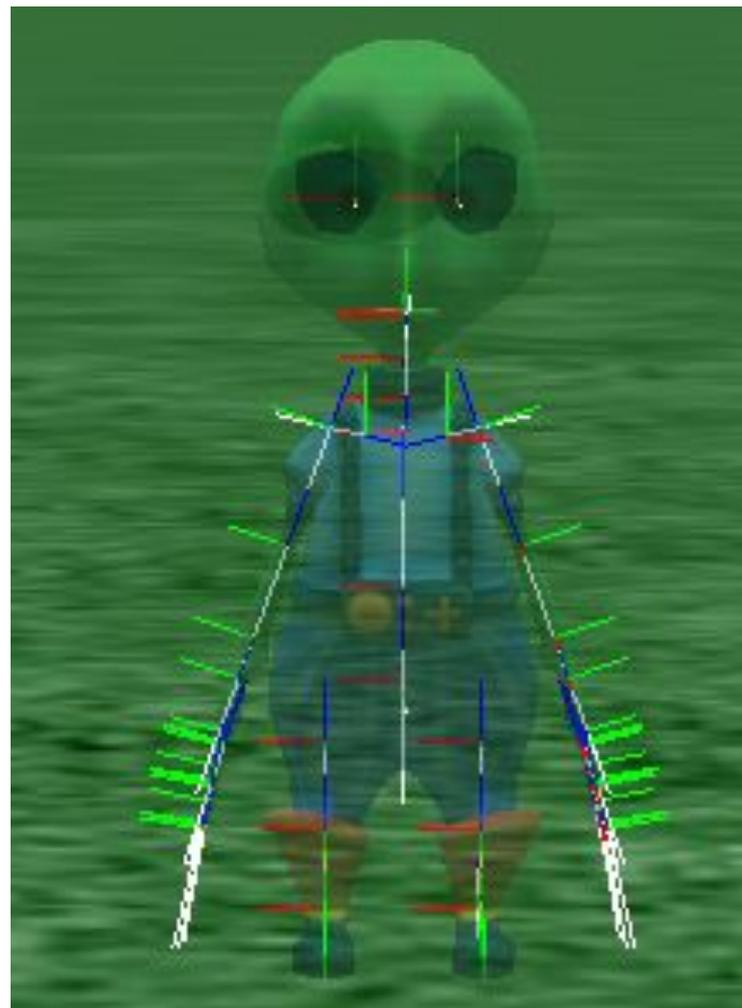
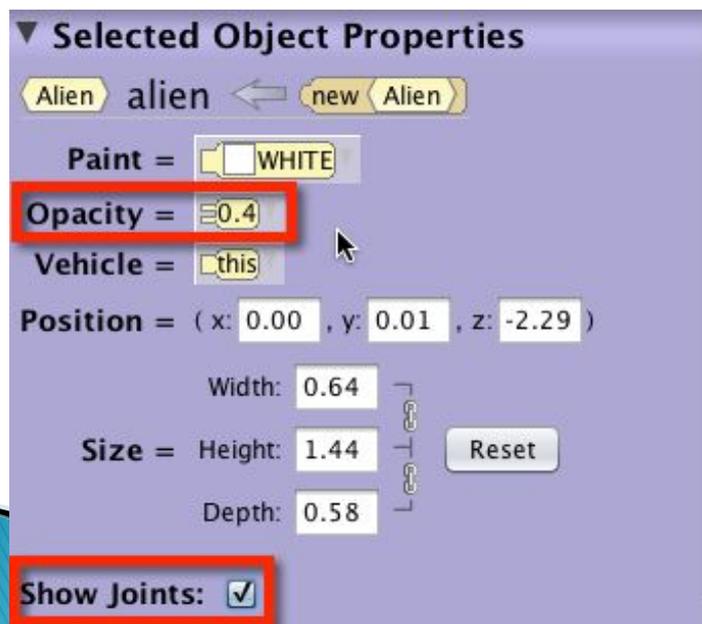
В редакторе кода

- Чтобы получить доступ к сочленению, нужно нажать на стрелку рядом с именем объекта в списке экземпляров.



In the Scene Editor

Если выбрать опцию **Show Joints** и подобрать *opacity* в Редакторе Сцены, можно увидеть все сочленения модели.



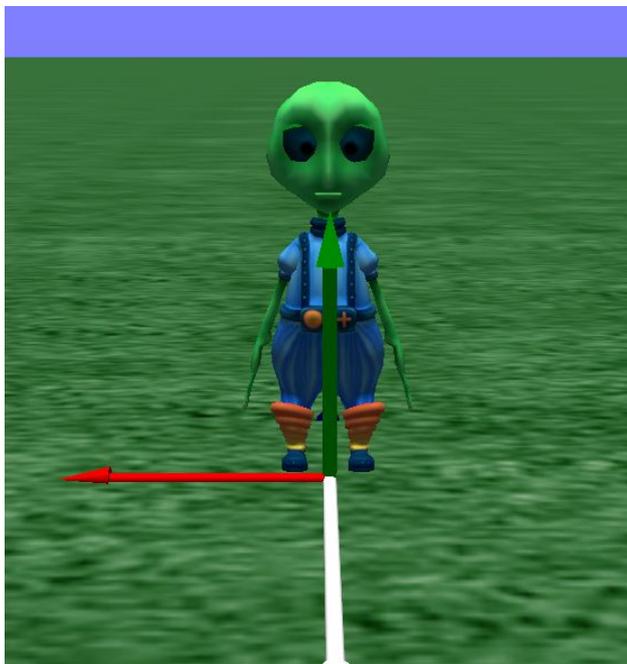
Ориентация сочленения

- Добавьте объект **axes** в сцену
- Подвиньте и сориентируйте этот объект с сочленением
- Белая ось указывает вперед, красная направо, зеленая вверх



Ориентация: целое vs. часть

Ориентация часть не всегда такая же как ориентация всего объекта.



Поступательное движение: сочленения

- Часть нельзя двигать
- Движение отсоединит сочленение (и присоединенную к нему часть) от остальной модели, поэтому действие *move* не разрешается для части объекта.



Вращательное движение

Сочленение/Часть

- Сочленение части служит опорной точкой для вращения части
- Два вида вращательного движения:
 - **поворот**
 - Вперед, назад
 - Влево, вправо
 - **вращение**
 - Влево, вправо

Вращение сочленения



Поворот вперед/назад



Красное колесо

*Поворачивает
сочленение вперед
и назад*

Поворот влево/вправо



Зеленое колесо:
*Поворачивает
сочленение влево и
вправо*

Roll left/right



Синее колесо:

*Вращает
сочленение влево и
вправо*

Панель методов

Панель методов содержит «плитки» для создания предложений, из которых состоит программа



Процедурные методы

Для всех объектов в Alice созданы встроенные процедуры

Процедуры – это методы, которые выполняют действия

Движение

Поворот

Вращение

Поворот к чему-либо

И т.д...



Built-in

The screenshot displays a software interface with two tabs: "Procedures" and "Functions". A dropdown menu is set to "group by category".

PlayingCard's Editable Procedures (6)

- edit `card2 jumpUp`
- edit `card2 jumpBack`
- edit `card2 jumpingJack`
- edit `card2 bendForward`
- edit `card2 bendBackward`
- edit `card2 touchToes`

Biped's Editable Procedures (0)

say, think

- `card2 say text: ???`
- `card2 think text: ???`

position

- `card2 move direction: ???, amount: ???`
- `card2 moveToward target: ???, amount: ???`
- `card2 moveAwayFrom target: ???, amount: ???`
- `card2 moveTo target: ???`
- `card2 place spatialRelation: ???, target: ???`

A red arrow points from the left side of the image to the "position" section of the code blocks, which is enclosed in a red rectangular border.

User-defined

user-defined



The screenshot displays a 3D environment with two card characters on a green field. Below the scene is a control panel for the 'card2' object, showing 'Procedures' and 'Functions' tabs. A red box highlights the 'PlayingCard' section, which lists six user-defined procedures: 'jumpUp', 'jumpBack', 'jumpingJack', 'bendForward', 'bendBackward', and 'touchToes'. Below this, the 'Biped' section shows 'say' and 'think' procedures.

card2

Procedures Functions

group by category

PlayingCard 's Editable Procedures

- edit card2 jumpUp
- edit card2 jumpBack
- edit card2 jumpingJack
- edit card2 bendForward
- edit card2 bendBackward
- edit card2 touchToes

Biped 's Editable Procedures

say, think

- card2 say text: [???]
- card2 think text: [???]