документация WASP\_SLAYER

Оглавление

[Загрузка ресурсов 3](#_Toc107757341)

[Текстуры 3](#_Toc107757342)

[Получение текстур 3](#_Toc107757343)

[Загрузка \*.obj файлов 3](#_Toc107757344)

[Получение загруженных \*.obj 3](#_Toc107757345)

[Загрузка аудио 4](#_Toc107757346)

[Получение аудио 4](#_Toc107757347)

[Удаление тестур аудио моделий 4](#_Toc107757348)

[Создание модели 4](#_Toc107757349)

[цвет 4](#_Toc107757350)

[Цвет + нормали 4](#_Toc107757351)

[Цвет + нормали + текстура 4](#_Toc107757352)

[Цвет + нормали + текстура + карта нормалей 4](#_Toc107757353)

[Небо 4](#_Toc107757354)

[Авто 4](#_Toc107757355)

[3Д объекты 5](#_Toc107757356)

[Создание 5](#_Toc107757357)

[добавление 5](#_Toc107757358)

[удаление 5](#_Toc107757359)

[методы 5](#_Toc107757360)

[2Д объекты 6](#_Toc107757361)

[Создание 6](#_Toc107757362)

[Добавление 6](#_Toc107757363)

[удаление 6](#_Toc107757364)

[Общие методы 7](#_Toc107757365)

[для текста 7](#_Toc107757366)

[Физика 7](#_Toc107757367)

[Твёрдое тело 7](#_Toc107757368)

[Методы 8](#_Toc107757369)

# Загрузка ресурсов

## Текстуры

Renderer renderer = core.getRenderer();

Путь ключ

renderer.loadTexture("textures/tests/floor.png", "floor");

renderer.loadCubemap(new String[]{

"textures/tests/skybox/r.png", // путь

"textures/tests/skybox/l.png",

"textures/tests/skybox/d.png",

"textures/tests/skybox/u.png",

"textures/tests/skybox/f.png",

"textures/tests/skybox/b.png"

}, "sky"); ключ

## Получение текстур

ключ

int texture = renderer.getTexture("floor");

## Загрузка \*.obj файлов

ModelLoader modelLoader = core.getModelLoader();

Путь ключ

modelLoader.loadModel("models/cube.obj", "cube");

## Получение загруженных \*.obj

VertexesData vertexesData = modelLoader.getVertexesData("cube");

## Загрузка аудио

AudioLoader audioLoader = core.getAudioLoader();

Путь ключ

audioLoader.addAudio("audio/bg\_m.mp3", "bg");

## Получение аудио

Audio audio = audioLoader.getAudio("bg");

## Удаление тестур аудио моделий

renderer.deleteTextur(“key”);

audioLoader.deleteAudio(“key”);

modelLoader.deleteVertexesData(“key”);

# Создание модели

Нужны для оптимизации

## цвет

вершины

Model model = new Model(new float[]{}, Core core);

## Цвет + нормали

Вершины нормали

Model model = new Model(new float[]{}, new float[]{}, core);

## Цвет + нормали + текстура

Вершины нормали текстур.коорд.

Model model = new Model(new float[]{}, new float[]{}, new float[]{} core);

## Цвет + нормали + текстура + карта нормалей

Вершины нормали текстур.коорд. текстур.коорд.

Model model = new Model(new float[]{}, new float[]{}, new float[]{}, new float[]{}, core);

## Небо

Model model = new Model("texture key",core);

## Авто

Model model = new Model(VertexesData,core);

model.minPoint – минимальная точка для отсечения невидимых объектов

model.maxPoint – максимальная точка для отсечения невидимых объектов

int c = model.getNumberPolygons(); - количество полигонов

# 3Д объекты

## Создание

RenderObject cubeRO = new RenderObject(model);

Текстура

RenderObject cubeRO = new RenderObject(model, “texture key”);

Текстура карта нормалей

RenderObject cubeRO = new RenderObject(model, “texture key” , “normal texture key”);

## добавление

Core.getRenderer().addRenderObject(cubeRO);

## удаление

Core.getRenderer().deleteRenderObject(cubeRO);

## методы

позиция

cubeRO.setPosition(new Vector3(0));

поворот

cubeRO.setRotation(new Vector3(0));

масштаб

cubeRO.setScale(new Vector3(0));

установка текстуры

cubeRO.setTexture(“texture key”);

установка карты нормалей

cubeRO.setNormalTexture(“texture key”);

получение модели

Model m = cubeRO.getModel();

Получение позиции

Vector3 pos = cubeRO.getPosition();

Получение поворота

Vector3 rot = cubeRO.getRotation();

Получение масштаба

Vector3 scale = cubeRO.getScale();

Проверка видимости

boolean v = cubeRO.inCamera(camera);

включить/отключить отрисовку

cubeRO.activity = true/false;

блики

cubeRO.specular = new Vector2(1, 32);

цвет

cubeRO.color = new Vector4(1);

# 2Д объекты

Из 2Д есть текст и изображения RenderText и RenderImg

## Создание

Изображение

RenderImg img = new RenderImg(“texture key”, uiModel);

текст

RenderText text = new RenderText(“font texture key”, “text”, uiModel);

RenderText text = new RenderText(“font texture key”, uiModel);

## Добавление

Core.getRenderer().addUI(img);

Core.getRenderer().addUI(text);

## удаление

Core.getRenderer().deleteUI(img);

## Общие методы

позиция

img.setPosition(new Vector3(0));

поворот

img.setRotation(new Vector3(0));

масштаб

img.setScale(new Vector3(0));

установка текстуры

img.setTexture(“texture key”);

получение модели

Model m = img.getModel();

Получение позиции

Vector3 pos = img.getPosition();

Получение поворота

Vector3 rot = img.getRotation();

Получение масштаба

Vector3 scale = img.getScale();

установка текстуры

img.setTexture(“texture key”);

включить/отключить отрисовку

img.activity = true/false;

цвет

img.color = new Vector4(1);

## для текста

text.text = “text”;

# Физика

## Твёрдое тело

RigidBody cubeRB = new RigidBody(new CubeCollider(cubeRO), cubeRO, core);

cubeRB.drag – трение float

cubeRB.mass – вес float

cubeRB.elasticity – что то вроде эластичности сколько будет передавать энергии при столкновении 0-всё float

cubeRB.usGravity – использование гравитации bool

cubeRB.activity - активность bool

установить скорость

cubeRB.setVelocity(new Vector3(1));

установить силу

cubeRB.setForce (new Vector3(1));

добавить скорость

cubeRB.addVelocity(new Vector3(1));

добавить силу

cubeRB.addForce (new Vector3(1));

получить скорость

Vector3 v = cubeRB.getVelocity();

получить силу

Vector3 f = cubeRB.getForce();

## Методы

Установить ускорение свободного падения

Physics physics = core.getPhysics();

physics.g = new Vector3(0,9.8f,0);

ray cast

Hit hit = new Hit();

Информация о попадание позиция направление мин расстояние и макс расстояние

Boolean f = physics.rayCast(hit, new Vector3(0), new Vector3(0,0,1),0,100);