Requirements

I. Introduction

1. Purpose

Система предназначена для создания трехмерных игр.

2. Scope

“CHDBS Engine” – это игровой движок, который позволяет разработчикам создавать трёхмерные игры с помощью языка программирования C++.

3. Product overview

3.1 Product perspective

Для работы движка требуются поддержка OpenGL.

Игровой движок не имеет GUI.

3.2 Product functions

CHDBS Engine предоставляет возможность отрисовки 3D объектов, простейшую симуляцию физики.

3.3 User characteristics

Пользователями системы являются разработчики видеоигр. Пользователи должны владеть базовыми знаниями C++ и иметь понимание работы системы. Зона ответственности пользователей - конфигурация системы, загрузка ресурсов и создание скриптов.

3.4 Limitations

Система не обеспечивает вывод звука, эффективное управление памятью, взаимодействия с сетью.

4. Definitions

* Transform – класс трансформаций в трехмерном пространстве. Transform описывает позицию, поворот и масштаб.
* Entity – некоторый класс, содержащий уникальный идентификатор, класс Transform и массив Component.
* Component – некоторый класс, прикрепляемый к Entity и определяющий поведение и функции этого Entity.
* Resource / Asset – ресурс, используемый игровым движком. В CHDBS Engine есть несколько видов ресурсов:
* XML – содержит пары ключ-значение
* Model – содержит трехмерную модель
* Texture – содержит изображение
* Scene – содержит список Entity, которые находятся в одном пространстве

II. Reference

Mike McShaffry, David Graham (2013). *Game Coding Complete.* Boston, MA: Course Technology.

Joey de Vries (2017). *Learn OpenGL.* (n.p.).

Jason Gregory (2009). *Game Engine Architecture.* Boca Raton, FL: Taylor and Francis Group, LLC.

Christer Ericson (2005). *Real-Time Collision Detection.* San Francisco, CA: Elsevier Inc.

III. Specific Requirements

1. External Interfaces

2. User Interface

1. Создание Component.

CHDBS Engine использует Component Based Architecture Pattern, следовательно, чтобы добавить функциональности некоторому Entity, пользователю нужно создать Component:

* создать класс, который должен наследоваться от интерфейса IComponent
* в заголовочном файле в описании класса добавить “CLASSDECLARATION(<название класса>)” и “CLASSDEFINITION(<название класса-родителя>, <название класса>)”

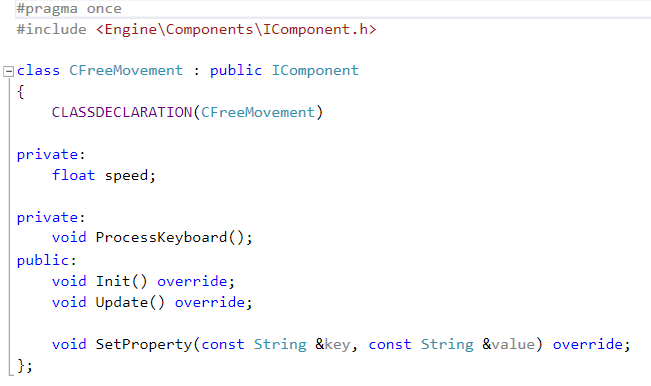


Figure 1 - пример описания Component



Figure 2 - пример описания Component

1. Абстрактные функции класса IComponent.

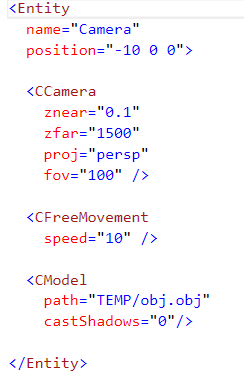


Figure 3 - пример XML файла для создания Entity "Camera"

IComponent содержит 3 абстрактные функции – Init(), Update() и SetProperty(String, String).

Функция SetProperty вызывается при создании Entity в EntityFactory во время парсинга XML файла, в котором хранятся пары ключ-значение. В *Figure 2* приводится пример реализации, при этом если в XML файле (*Figure 3*) есть пара с ключом *“speed”*, то полю *“speed”* в данном Component присваивается значение.

Функция Init вызывается после присваивания всех значений из XML файла.

Функция Update вызывается после каждой смены кадра.

1. Загрузка ресурсов.

Загрузка осуществляется через класс ResourceManager, который возвращает специальный объект ресурса по данному пути к требуемому файлу, а также в нём кэшируются уже загруженные ресурсы.

3. Hardware Interfaces

Нужна поддержка клавиатуры и мыши.

4. Software Interfaces

Для работы движка требуются:

* поддержка OpenGL версии 3.0 и выше
* операционная система Windows версии 7 и выше
* IDE Visual Studio версии 2015 и выше

5. Product Functions

6. Communication Requirements

7. Performance Requirements

8. Memory Requirements

9. Site Adaptation Requirements

10. Design Constraints

11. Software System Attributes

IV. Verification

V. Supporting Information

VI. Appendices

1. Assumptions & Dependencies

2. Acronyms & Abbreviations