Software Requirements Specification

For

CHDBS Engine

1. **Введение**
2. Цели

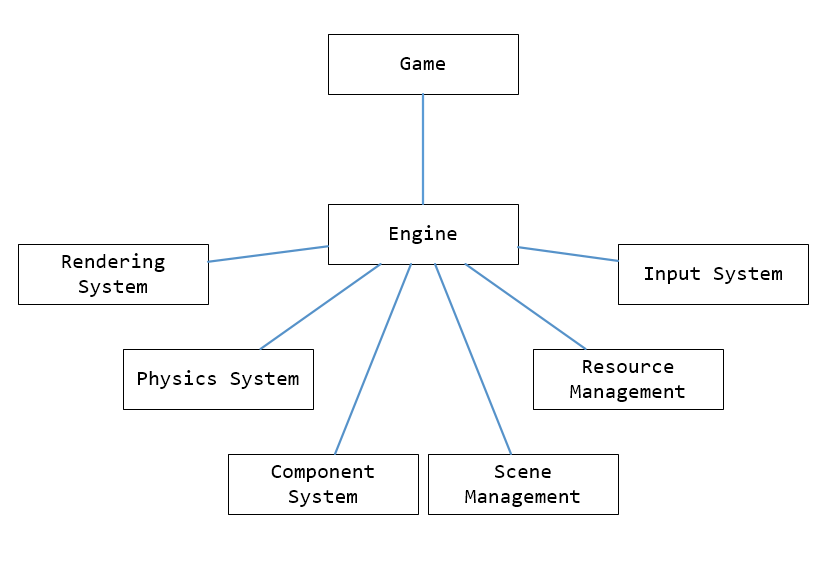
Система предназначена для создания трехмерных видеоигр.

1. Область действия

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

1. Общее описание
   1. *Видение продукта*

Цель игрового движка – возможность создания различных видеоигр без существенного изменения основных систем: отрисовка, симуляция физики и т.д.



* 1. *Функции продукта*

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

* 1. *Характеристики пользователя*

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.

* 1. *Ограничения*

Система не обеспечивает вывод звука, эффективное управление памятью, взаимодействия с сетью. Игровой движок не имеет GUI.

1. Определения

* Transform – класс трансформаций в трехмерном пространстве. Transform описывает позицию, поворот и масштаб.
* Entity – некоторый класс, содержащий уникальный идентификатор, класс Transform и массив Component.
* Component (компонент) – некоторый класс, прикрепляемый к Entity и определяющий поведение и функции этого Entity.
* Script (скрипт) – код определенного Component.
* Shader (шейдер) – программа, исполняемая с помощью GPU
* Texture (текстура) – изображение, используемое для отрисовки объектов
* Model (модель) – трехмерная модель
* Material (материал) – набор текстур и шейдера
* Skybox – специальный объект, используемый для рисования заднего фона
* Cubemap – специальная текстура, обычно используемая для отрисовки отражений. Является набором из 6 текстур
* Particle System (система частиц) – специальная техника, позволяющая рисовать большое количество спрайтов или 3D моделей
* Rigidbody (rigid body) – представляет абсолютно твердое тело
* Collider (коллайдер) – специальный компонент для проверки столкновений, представляет собой некоторый геометрический объект
* AABB (axis aligned bounding box) – представляет параллелепипед со сторонами, параллельными осям координат
* OBB (oriented bounding box) – представляет параллелепипед с произвольным вращением
* Scene (сцена) – содержит Entity, которые находятся в одном пространстве
* Resource / Asset – ресурс, используемый игровым движком. В CHDBS Engine есть несколько видов ресурсов:
* XMLResource – содержит пары ключ-значение
* ModelResource – содержит трехмерную модель
* TextureResource – содержит изображение
* SceneResource – содержит информацию о сцене
* GLFW – библиотека для создания и открытия окон, создания OpenGL контекста и управления вводом
* GLEW – кроссплатформенная библиотека для вызова функций OpenGL.

1. **Ссылки**

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.

1. **Детальные требования**
   1. Требования к внешним интерфейсам
      1. *Интерфейс системы*

Для интеграции с другими системами игровой движок должен обеспечивать:

* загрузку сцен из XML файла
* загрузку настроек для движка

Так как сцены содержат Entity, то игровой движок не обязан предоставлять доступ к созданию Entity внешним системам, это должно происходить исключительно в скриптах (которые прикреплены к Entity в некоторой сцене).

* + 1. *Интерфейс пользователя*

Нет интерфейса пользователя.

* + 1. *Интерфейсы аппаратного обеспечения*

Необходима поддержка клавиатуры и мыши.

* + 1. *Интерфейсы программного обеспечения*

Для работы движка требуются:

* операционная система Windows версии 7 и выше
* поддержка OpenGL версии 3.0 и выше – для отрисовки графики
  1. Функциональные требования

Функции игрового движка делятся на:

* отрисовку графики
* симуляцию физики
* обновление компонентов
* управление сценами
* управление ресурсами
* обработку устройств ввода
  + 1. *Отрисовка графики*

1. Создание материала с заданными текстурами и шейдером
2. Отрисовка статичных (неизменяющихся) трехмерных моделей с помощью материалов
3. Отрисовка динамических трехмерных моделей со скелетной анимацией
4. Отрисовка систем частиц
5. Отрисовка skybox
6. Frustum culling – отрисовка только видимых объектов
7. Отрисовка внутриигрового GUI
   * 1. *Симуляция физики*
8. Проверка столкновений между:

* сферами
* AABB
* сферой и AABB
* сферой и треугольником
* сферой и набором треугольников
* AABB и треугольником
* сферой и набором треугольников

Между треугольниками столкновения не проверяются.

1. Проверка пересечения луча и отрезка с AABB, сферой, треугольниками
2. Компонент Rigidbody представляет динамический физический объект
3. Компоненты для проверки столкновений:

* AABBCollider – содержит AABB
* SphereCollider – содержит сферу
* MeshCollider – содержит набор треугольников

1. Entity с компонентами Rigidbody и Collider должны взаимодействовать с другими физическими объектами
2. Entity с компонентом Collider, но без Rigidbody представляет статический физический объект
3. Collider может быть триггером, т.е. физические объекты с этим Collider сталкиваться не будут, но им будут посланы сообщения
4. Rigidbody могут не участвовать в симуляции физики: силы, столкновения не будут влиять на этот Rigidbody, но остальные физические объекты будут взаимодействовать с этим Rigidbody.
   * 1. *Управление компонентами*
5. Все компоненты должны обновляться последовательно у каждого активного Entity после смены кадра
6. Обновление компонентов физики происходит с фиксированной точностью (~50 кадров в секунду)
7. Каждый Entity должен предоставлять доступ к своим компонентам
   * 1. *Управление сценами*
8. Движок должен поддерживать загрузку сцен
   * 1. *Управление ресурсами*
9. Загрузка и выгрузка ресурсов из оперативной памяти
10. Предотвращение загрузки уже загруженного ресурса
    * 1. *Обработка устройств ввода*
11. Определение состояния клавиш, кнопок
12. Определение позиции мыши, смещения относительно предыдущего кадра
    1. Требования к связи

Нет.

* 1. Требования к производительности

Система должна обеспечивать хотя бы 60 кадров в секунду в простых сценах (в Release Mode) при конфигурации:

* 2-х ядерный процессор AMD A9 с тактовой частотой 2.0 GHz
* 8 GB оперативной памяти
* Операционная система Windows 10
  1. Требования к памяти

Система должна занимать не более 1 гигабайта в оперативной памяти и 1 гигабайта во внешней памяти.

* 1. Условия окружения

Нет.

* 1. Ограничения дизайна

1. Сцены не иерархичны, т.е. Entity не может содержать другие Entity
   1. Атрибуты системы
      1. *Надежность*

Никогда не должно происходить аварийное завершение системы.

* + 1. *Доступность*

Нет специальных требований по доступности.

* + 1. *Безопасность и конфиденциальность*

Нет специальных требований по безопасности и конфиденциальности.

* + 1. *Поддерживаемость*

Только основные функции, видимые пользователю, должны быть полностью задокументированы.

* + 1. *Портируемость*

Система должна быть портируемой на любую систему Windows версии 7 и новее.

* + 1. *Требования к безопасности*

Нет специальных требований по безопасности.

1. **Тестирование и проверка**

Тестированию должны подвергнуться:

* класс геометрических векторов
* класс кватернионов
* класс матриц
* класс трансформаций
* всевозможные пересечения геометрических объектов
* структуры данных
* класс строк

1. **Вспомогательная информация**
   1. Подключение

Чтобы использовать игровой движок для создания игры, необходимо создать проект в решении (solution) самого движка. Проверить: подключены ли статические библиотеки OpenGL, GLEW, GLFW и Assimp. Также следует отметить файл *«Engine\Base\Typedef.h»* как forced include, т.к. в нем содержатся forward declaration всех классов.

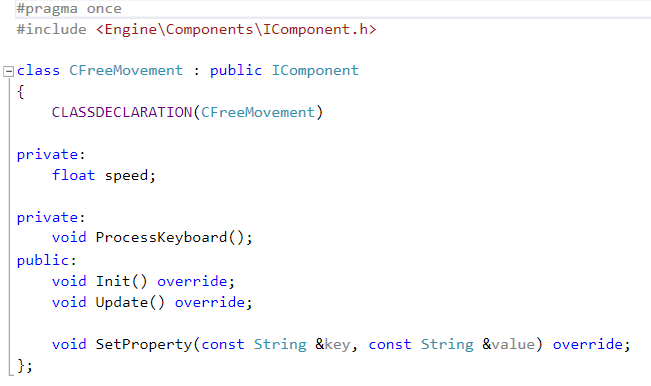
* 1. Использование
     1. *Запуск*

Для начала следует инициализировать игровой движок, далее загрузить все файлы, содержащие информацию о сценах, указать начальную сцену и запустить главный цикл движка.

* + 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
* зарегистрировать класс в EntityFactory для возможности загрузки данных из XML файлов
* 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*

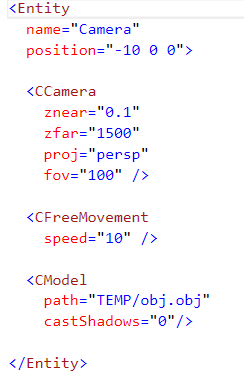


Figure 3 - пример XML файла для создания Entity "Camera"

* + 1. *Абстрактные функции класса IComponent*

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

Функция SetProperty должна вызываться 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 должна вызываться после присваивания всех значений из XML файла.

Функция Update должна вызываться после каждой смены кадра.

* + 1. *Загрузка ресурсов*

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

1. **Приложения**
   1. Предположения и зависимости

Нет.

* 1. Акронимы и аббревиатуры
* AABB – axis aligned bounding box
* OBB – oriented bounding box
* OpenGL – Open Graphics Library
* GLEW – OpenGL Extension Wrangler Library