

CMDK

Copyright © 2016-2017 by Marat Sungatullin

В этом документе даются сведения, необходимые для программирования игр на движке “CMDK”.

## Подсистема ввода-вывода (io)

### Структура для хранения путей к файлам и директориям

Для удобства оперирования путями к файлам и директориям в движке используется класс File. Чтобы создать объект типа File, описывающий путь к директории, нужно просто передать строку с путем к директории в конструктор. Чтобы составить путь к файлу, нужно передать в конструктор класса File строку с путем к директории в первом параметре и строку с именем файла – во втором. Метод getPath() возвращает строку с путем, который хранится в объекте типа File. Метод exists() возвращает true, если директория или файл, чей путь хранится в объекте типа File, существуют, иначе возвращается false. Существуют методы проверки пути – является ли путь путем к директории, путем к файлу, является ли путь абсолютным. Пример (предполагается, что файлы существуют):

io::File fname1("../data/characters/skin");

fname1.exists(); // Возвращает true

fname1.isDirectory();//Возвращает true

fname1.isFile();//Возвращает false

fname1.isAbsolute();// Возвращает false

io::File fname2("../data/characters/skin", "monster1.txt");

fname1.exists(); // Возвращает true

fname1.isDirectory();//Возвращает false

fname1.isFile();//Возвращает true

fname1.isAbsolute();// Возвращает false

Метод length() возвращает длину файла в байтах. Метод getAbsolutePath() возвращает полный путь на основе хранимого в объекте File относительного пути.

Класс DirectoryInputStreamArchive позволяет хранить список абсолютных путей директорий – регистрировать их в движке. Метод addPath() служит для добавления пути к директории, возможно заданной относительным путем или просто именем, в список. Метод addPaths() сохраняет во внутреннем списке класса не только директорию, заданную через параметр, но и все ее вложенные директории. Пример:

io::DirectoryInputStreamArchive arch;

arch.addPaths("../data"); // В arch будут лежать пути к самой директории data //и ко всем ее поддиректориям и поддиректориям директорий и т.д.

Метод getInputStream() автоматически находит заданный файл, если тот существует в любой из зарегистрированных в классе директорий, и открывает его для чтения.

### Формат хранения целых чисел в бинарных файлах

Целые числа в бинарных файлах хранятся в следующем формате:

Битовый шаблон 1-го байта:

msxx xxxx

0123 4567

m = оставшиеся биты содержатся в следующем байте = {0=нет, 1=да}

s = знак числа = {0=без знака, 1=со знаком}

x = первые биты числа (0-6), без знака

Битовый шаблон n-го байта:

mxxx xxxx

0123 4567

m = оставшиеся биты содержатся в следующем байте = {0=нет, 1=да}

x = следующие биты числа, без знака

Для загрузки целых чисел данного формата из файла предназначена статическая функция readVarInt, определение которой находится в файле “ChunkInputStream.cpp”.

Класс InputStream является базовым интерфейсом потока ввода байтов.

Класс FilterInputStream содержит внутри себя указатель на другой объект класса, производного от класса InputStream. Суть класса FilterInputStream состоит в определении интерфейса класса, который делегирует операции класса InputStream какому-то другому объекту класса, производного от InputStream, указатель на который передается как параметр в конструктор класса FilterInputStream.

Класс DataInputStream реализует интерфейс, заданный классом FilterInputStream, для чтения примитивных типов данных: булевых значений, целых, чисел с плавающей точкой, символов Юникода и т.п.