IgriZdes Engine

(izeng)

Описание функций движка

Оглавление

Предисловие

Движок IZEng является бесплатным open-source игровым движком двумерной графики с помощью DirectX с включенными простыми траекториями движения — по прямой, окружности, с/без ускорения, с трением/гравитацией или без таковых. На этом движке написана одна из моих игр — RockCarrier. Полноценной документации к движку нет. Ниже и в архиве с движком — мой набросок на такую документацию. Если у вас возникли вопросы:

doublevenom@gmail.com

в заголовке укажите «IgriZdes Engine»

IScreen интерфейс

Интерфейс управления экраном, камерой, загрузкой/выгрузкой графики

Ограничения

Во всех методах возвращаемый результат — HRESULT, используйте FAILED(hResult) для проверки успешности выполнения методов.

Методы

HRESULT ResetDisplay(

[in] HWND hDispWnd,

[in] int nWidth,

[in] int nHeight,

[in] int nRefRate,

[in] BOOL bFullscreen,

[in] int nFPS);

Настройка экрана

параметр hDispWnd - дескриптор окна Windows для отображения

параметр nWidth - ширина экрана

параметр nHeight - высота экрана

параметр nRefRate - частота (Гц)

параметр bFullscreen - отображение на весь экран или в окне; ЕСЛИ ВЫ ОТЛАЖИВАЕТЕ ПРОГРАММУ СОВЕТУЮ ВСЕГДА ЗАПУСКАТЬ В ОКНЕ

параметр nFPS - число кадров в секунду в интервале от 1 до 999, рекомендую 30

HRESULT UpdateScreen();

Обновление экрана — выводит на экран новый кадр, позаботьтесь о том, чтобы кадр содержал анимации – см. IDraw->Show()

HRESULT MoveCamera([in]POINTF* pvecD);

Управление передвижением камеры.

параметр pvecD — задает вектор сдвига камеры от текущего положения

HRESULT SetCamera([in]POINTF* pptCam);

Установка положения камеры в пространства

параметр pptCam — задает положение камеры в пространстве

HRESULT GetCameraPos([in,out]POINTF *pptCam);

Получение текущей позиции камеры

ВЫХОД pptCam — содержит возвращенную позицию

HRESULT ShutDown();

Освобождение ресурсов IZEngine, удаление всех объектов движка. ВЫЗЫВАТЬ ТОЛЬКО ПО ОКОНЧАНИИ РАБОТЫ С IZEngine — ВСЕ УКАЗАТЕЛИ БУДУТ БИТЫМИ ПОСЛЕ ВЫЗОВА.

HRESULT GetScreenRect([in,out]RECT *prcScreen);

Получение прямоугольника, ограничивающего экран

выход prcScreen — содержит возвращенный прямоугольник. left и top прямоугольника всегда равны 0, a right и bottom – заданному разрешению.

HRESULT GetScreenWH([in,out]POINTF *pptScreen);

Получение ширины и высоты экрана, т.е. заданного разрешения

выход pptScreen — содержит возвр. разрешение

HRESULT GetCameraRect([in,out]RECT *prcCam);

Получение прямоугольника, ограничивающего текущее положение камеры

выход prcCam — содержит возвр. прямоугольник; left & top – левый верхний угол камеры, right & bottom – нижний правый

HRESULT GetCameraRBPos([in,out]POINTF *pptCam);

Получение координат правого нижнего угла текущего положения камеры

выход pptCam — содержит возвр. координаты

HRESULT SetWorldBounds([in]RECT* prcBounds);

Установка границ мира — камера сможет двигаться только в этих границах, по умолчанию - -32000,-32000,32000,32000

параметр prcBounds — задает границы мира

HRESULT GetWorldBounds([in,out]RECT* prcBounds);

Получение установленных ранее границ мира

параметр prcBounds — содержит возвр. границы мира

HRESULT AddFrameTable(

[in,out] int* pnID,

[in] UCHAR* strPathToFile,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparent,

[in] DWORD dwTranspColor);

Создание таблицы кадров на основе графического файла. Использование таблицы кадров визуальным объектом осуществляется с помощью других методов. См. pAnim->UseExistingFrameTable(<индекс *pnID>) и его аналоги в том же интерфейсе.

параметр strPathToFile — путь к файлу. Поддерживаются JPEG,PNG,BMP. (другие — на свой страх и риск) Рекомендуется PNG при использовании прозрачности и JPEG при ее отсутствии.

параметр nNeededFrameW — нужная ширина кадра, 0 - исходная

параметр nNeededFrameH — нужная высота кадра, 0 - исходная

параметр nColumns — число колонок в таблице в файле. Если у вас одна колонка, можно установить nColumns =1 и nRows=0 и число рядов будет посчитано автоматически.

параметр nRows — число рядов в таблице в файле. Если у вас один ряд, можно установить nRows=1 и nColumns=0 и число колонок будет подсчитано автоматически.

параметр bTransparent — имеет ли кадр прозрачный фон, true – да.

параметр dwTranspColor — цвет, считающийся прозрачным. Не используется, если bTransparent=false

выход pnID — индекс созданной таблицы кадров, используется для связывания таблицы кадров с визуальным объектом

HRESULT AddBMPFrameTable(

[in,out] int* pnID,

[in] UCHAR* strPathToFile,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparent,

[in] DWORD dwTranspColor);

Аналог AddFrameTable(), но можно использовать только для BMP файлов. использует Win API GDI.

HRESULT AddEmptyFrameTable(

[in,out] int* pnID,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparentFill,

[in] DWORD dwFillColor);

Создание пустой таблицы кадров. Использование таблицы кадров визуальным объектом осуществляется с помощью других методов. См. pAnim->UseExistingFrameTable(<индекс *pnID>) и его аналоги в том же интерфейсе.

параметр nNeededFrameW — нужная ширина кадра

параметр nNeededFrameH — нужная высота кадра

параметр nColumns — число колонок в таблице в файле

параметр nRows — число рядов в таблице в файле

параметр bTransparentFill — см. dwFillColor. Определяет, будет ли dwFillColor считаться прозрачным

параметр dwFillColor — цвет, которым будут залиты кадры

выход pnID — индекс созданной таблицы кадров, используется для связывания таблицы кадров с визуальным объектом

HRESULT AddAngledTables(

[in,out] int* pnID,

[in] int* pidSrcFrameTable,

[in]float fTurnAngle,

[in]float fNeededAngleOfFirstTable);

Создание набора таблиц кадров поворота (НТКП) на основе существующей таблицы кадров. Чтобы подключить НТКП к виз. объекту вызовите из него UseExistingAngledTables(<индекс *pnID>)". Использование НТКП визуальным объектом осуществляется с помощью других методов. См. pAngledDraw->UseExistingAngledTables() и его аналоги того же интерфейса.

параметр pidSrcFrameTable — индекс исходной таблицы кадров. Ее создание - см. AddFrameTable().

параметр fTurnAngle — угол поворота в градусах. Если равен 30, то в наборе будет 360/30 = 12 таблиц кадров

параметр fNeededAngleOfFirstTable — угол для нулевой — старт-таблицы НТКП.

выход pnID - индекс созданного НТКП, используется для связывания с визуальным объектом

HRESULT AddOrGetExistingFrameTable(

[in,out] int* pnID,

[in] UCHAR* strPathToFile,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparent,

[in] DWORD dwTranspColor);

Если файл с таким именем (полным) уже использовался, то возвращает номер его таблицы кадров, иначе - аналог AddFrameTable().

IInput интерфейс

Интерфейс для работы со вводом пользователя — нажатиями клавиатуры, мыши.

Методы

HRESULT CheckKeyDown([in] int vKey,[in,out] BOOL* bRes);

Проверка нажатия заданной клавиши

параметр vKey — виртуальный код клавиши. Коды такие же, как и в Win API: VK_SPACE, VK_UP, 'W' – означает нажатие W (или w) и т.п.

выход bRes — true – клавиша нажата, false – нет

HRESULT CheckKeyUp([in] int vKey,[in,out] BOOL* bRes);

Проверка отжатия заданной клавиши

параметр vKey — виртуальный код клавиши. Коды такие же, как и в Win API: VK_SPACE, VK_UP, 'W' – означает нажатие W (или w) и т.п.

выход bRes — true – клавиша отжата, false – нет

IAnim интерфейс

Интерфейс для работы с анимацией — выбора таблицы кадров — базы для анимации, создания, изменения текущего кадра, заморозки изображения и т.п.

Ограничения

Анимаций у одного визуального объекта может быть не больше 64, индексация соответственно от 0 до 63.

В движке предполагается, что у любого визуального объекта, выводимого на экран есть анимация с индесом 0 — она по умолчанию. Рекомендуется придерживаться этого правила.

Перечисления

enum {PLAY_REPEAT, PLAY_ONCE};

Типы воспроизведения анимации. См. SetAnimType().

Методы

HRESULT UseExistingFrameTable([in] int* pnID);

Привязка существующей таблицы кадров к текущему визуальному объекту

параметр pnID — индекс таблицы кадров. См. pScreen->AddFrameTable().

HRESULT SetFrameTime([in]int nAnimInd, [in] int msTime);

Установка времени, которое проходит между сменой кадров. Должно быть больше 1 / FPS. По умолчанию = 30. (33 FPS) См. pScreen-ResetDisplay().

параметр nAnimInd — индекс анимации, для которой устанавливается время

параметр msTime — задает время между сменой кадров

HRESULT CreateAnim(

[in] int nAnimInd,

[in] int nFramesAmount,

[in] int *arrFrameInds);

Создание анимации. Визуальный объект должен быть связан с таблицей кадров до вызова этого метода.

параметр nAnimInd — индекс новой анимации. См. ограничения.

параметр nFramesAmount — число кадров в анимации (размерность массива arrFrameInds)

параметр arrFrameInds — массив индексов кадров из связанной таблицы кадров. Индексы считаются от 0 (левый верхний кадр из таблицы) до числа кадров в таблице слева-направо и сверху-вниз.

HRESULT SetAnimType([in] int nAnimInd,[in] int nPlayType);

Установка типа анимации.

параметр nAnimInd - индекс анимации, к которой применяется метод.

параметр nPlayType = тип анимации. Может быть PLAY_REPEAT или PLAY_ONCE. PLAY_REPEAT — по окончании проигрывания анимации анимация будет проигрываться с начала. PLAY_ONCE - по окончании проигрывания анимации анимация замрет на последнем кадре, метод SelectActiveFrame() может изменить выводимый кадр в данной ситуации.

HRESULT SelectActiveAnim([in] int nAnimInd);

Выбор анимации для проигрывания

параметр nAnimInd — индекс анимации.

HRESULT SelectActiveFrame([in] int nFrameInd);

Выбор кадра, который будет выведен на экран — текущего кадра

параметр nFrameInd — индекс кадра

HRESULT FreezeFrame([in] BOOL bFreeze);

Замораживание текущего кадра.

параметр bFreeze — true – заморозить, false – разморозить

HRESULT GetCurAnimInd([in,out] int *pnCurAnimInd);

Возврат индекса текущего кадра

выход pnCurAnimInd — индекс текущего кадра

HRESULT CheckEndAnim([in,out] BOOL *pbEndAnim);

Проверка, не закончилась ли анимация. Только для PLAY_ONCE анимаций. См. SetAnimType(). За счет SelectActiveFrame() можно выбрать другой кадр и анимация не будет считаться завершенной.

выход pbEndAnim — флаг окончания анимации

HRESULT CheckHalfAnimPlayed([in,out] BOOL *pbHalfAnim);

Проверка, проиграна ли половина анимации. Не зависит от типа анимации.

выход pbHalfAnim — флаг, означающий, проиграна ли половина анимации.

HRESULT GetFrameAmountInTable([in,out] int *pnFrames);

Возврат числа кадров в связанной с данным виз. объектом таблице кадров.

выход pnFrames — число кадров

HRESULT UseExistingFrameTableAsSimpleAnim([in] int* pnID);

Аналог UseExistingFrameTable(), но дополнительно создает анимацию по умолчанию. В данном случае анимация — простая.

параметр pnID — индекс таблицы кадров. См. pScreen->AddFrameTable().

Похожие методы

IAnim::UseExistingFrameTable(), CreateSimpleAnim()

HRESULT CreateSimpleAnim();

Создание анимации по умолчанию - простой анимации в данном случае. Объект должен быть связан с таблицей кадров до вызова этого метода. См. UseExistingFrameTable() интерфейса IAnim.

Похожие методы

IAnim::CreateLinearAnim().

HRESULT CreateLinearAnim([in] int nAnimInd,[in] int nStartFrame,[in] int nEndFrame);

Создание линейной анимации под номером nAnimInd. Объект должен быть связан с таблицей кадров до вызова этого метода. См. UseExistingFrameTable() интерфейса IAnim.

параметр nAnimInd — индекс, под которым будет сохранена новая анимация

параметр nStartFrame – стартовый кадр анимации, должен быть меньше nEndFrame

параметр nEndFrame — конечный кадр анимации, должен быть больше nStartFrame

Похожие методы

IAnim::CreateSimpleAnim()

HRESULT LoadSimpleAnim(

[in] UCHAR* strPathToFile,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparent,

[in] DWORD dwTranspColor);

Загрузка таблицы кадров, привязка ее к объекту и создание в качестве анимации по умолчанию простой анимации. Параметры - см. AddFrameTable() интерфейса IScreen.

Похожие методы

IScreen:: AddFrameTable()

IAnim::UseExistingFrameTable(), CreateSimpleAnim()

HRESULT LoadFrameTable(

[in] UCHAR* strPathToFile,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparent,

[in] DWORD dwTranspColor);

Загрузка таблицы кадров и ее привязка к объекту. Параметры — см. AddFrameTable() интерфейса IScreen.

Похожие методы

IScreen:: AddFrameTable()

IAnim::UseExistingFrameTable()

HRESULT LoadOrUseExistingSimpleAnim(

[in] UCHAR* strPathToFile,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparent,

[in] DWORD dwTranspColor);

Загружает или использует загруженную ранее простую анимацию (сравнивает имена файлов). Внимание: если файл strPathToFile уже был загружен одним из вызовов AddFrameTable(), LoadFrameTable() или других методов, то все остальные параметры метода будут проигнорированы и использованы значения из предыдущей загрузки strPathToFile. Если вас это не устраивает, используйте LoadSimpleAnim().

Параметры — см. AddFrameTable().

Похожие методы

LoadOrUseExistingFrameTable()

HRESULT LoadOrUseExistingFrameTable(

[in] UCHAR* strPathToFile,

[in] int nNeededFrameW,

[in] int nNeededFrameH,

[in] int nColumns,

[in] int nRows,

[in] BOOL bTransparent,

[in] DWORD dwTranspColor);

Загружает или использует загруженную ранее таблицу кадров (сравнивает имена файлов) и связывает ее с объектом. Внимание: если файл strPathToFile уже был загружен одним из вызовов AddFrameTable(), LoadFrameTable() или других методов, то все остальные параметры метода будут проигнорированы и использованы значения из предыдущей загрузки strPathToFile. Если вас это не устраивает, используйте LoadFrameTable().

Параметры — см. AddFrameTable().

Похожие методы

LoadOrUseExistingSimpleAnim()

IDraw интерфейс

Интерфейс для работы с экраном — добавления/удаление объекта в список отображения, рисование граф. примитивов, получение пикселя подготовленного кадра и т.п.

Ограничения

При работе движок выводит всю графику в формате 32 бит ARGB, т.е. цвет представляет собой шестнадцатиричное 0xAARRGGBB, где:

RR – оттенок красного, от 00 до FF

GG – зеленого аналогично,

BB – синего,

AA на данный момент не используется и равен 00.

Таким образом, если вы работаете с методами, использующими цвета (получение, установка — напр. прозрачности) используйте эти правила. Напр. вызов метода для загрузки анимации с прозрачным цветом зеленым макс. оттенка может выглядеть:

```
IAnim::LoadSimpleAnim(
    (UCHAR*)”C:\mygame\res\hero.png”,
```

0, 0, 0, 1,
TRUE, 0x00FF00); //цвет исходного изображения, считающийся прозрачным
На заметку: вы можете использовать встроенный метод DWORD rgb32bit(char a, char r,char g, char b) для формирования цвета и логические операторы | и & для работы с цветом. (чит. в интернете)

Методы

HRESULT Show();

Добавление объекта в список вывода на экран. Чем позже добавлен объект, тем «ближе» он к нам. Т.е., если добавить 2 объекта, перекрывающих друг друга, мы увидим полностью тот объект, который добавлен последним. Обратите внимание на то, что, с помощью этого метода нельзя задать глубину вывода объекта.

HRESULT Hide();

Скрытие объекта. Противоположно Show().

HRESULT ShowOn([in] int depth);

Добавление объекта в список вывода на экран с глубиной depth.

параметр depth – глубина вывода. 0 <= depth, чем больше, тем ближе объект к нам.

HRESULT CheckVisible([in,out]BOOL* pRes);

Проверка видимости данного объекта.

выход pRes — флаг видимости

HRESULT GetTranspColor([in,out]DWORD *pdwColor);

Получение цвета, считающегося прозрачным для вызывающего объекта. Об установке цвета прозрачности — см. напр. IScreen::AddFrameTable().

выход pdwColor — «прозрачный» цвет

HRESULT GetPixel([in] POINTF* pptWhere,[in,out]DWORD *pdwPixColor);

Получение пикселя. Вызову обязательно должен предшествовать Lock(), а по окончании работы — Unlock().

параметр pptWhere — координаты пикселя от (0,0) до (IParams::GetWidth()-1, IParams::GetHeight()-1).

выход pdwPixColor — пиксель в формате ARGB. (см. ограничения выше)

HRESULT Lock();

Блокирование поверхности вызывающего объекта. Работает в паре с Unclock(). Требуют некоторые другие методы для корректной работы. (см. GetPixel()) Операция долгая и, если можно обойтись без нее — лучше обойдитесь. Если же очень нужно, то за 1 кадр для 1 объекта рекоммендую вызывать не более одного раза. И по окончании вызывать Unlock().

Важно вызвать Unlock() до обновления экрана (вызова IScreen::UpdateScreen()), иначе программа вероятнее всего зависнет. Те, кто читал литературу по DirectX поймут, почему это так важно.

HRESULT Unlock();

Освобождение поверхности вызывающего объекта. Работает в паре с Lock(). Подробнее — чит. Lock().

IAngledDraw интерфейс

Интерфейс, дополняющий IDraw (!) функциональностью для работы с поворачиваемыми изображениями.

Методы

HRESULT UseExistingAngledTables([in] int* pnAngledTablesPack);

Связывание существующего набора таблиц кадров поворота (НТКП) с вызывающим объектом.

параметр pnAngledTablesPack — номер таблиц кадров поворота, может быть получен работой IScreen::AddAngledFrameTables()

HRESULT UseExistingAngledTablesAndSimpleAnim([in] int* pnAngledTablesPack, [in] int nNumStartTable);

Связывание существующего набора таблиц кадров поворота (НТКП) с вызывающим объектом и создание простой анимации (Simple Anim).

параметр pnAngledTablesPack — номер таблиц кадров поворота, может быть получен работой IScreen::AddAngledFrameTables()

параметр nNumStartTable — номер таблицы НТКП, которая будет считаться стартовой. Каждая таблица соответствует некоторому углу поворота, поэтому номер задает стартовый поворот

HRESULT GetNumTables([in,out] int* pnNumTables);

Получение числа таблиц НТКП вызывающего объекта. Т.к. само НТКП получается за счет работы IScreen::AddAngledTables(), то за подробностями см. этот метод.

выход pnNumTables — число таблиц

HRESULT SelectTable([in] int nNo);

Выбор текущей таблицы из НТКП по номеру таблицы. Иными словами, индекс определяет угол, под которым будет выведен на экран вызывающий объект.

параметр nNo — номер таблицы, в интервале [0;число таблиц всего-1]. См. GetNumTables()

HRESULT SelectTableByAngle([in] float fAngle);

Выбор текущей таблицы из НТКП на основе угла. Иными словами, вы вводите угол, движок выводит кадры под этим углом. Если же таковых нет — движок берет ближайшее соответствие.

Если, скажем, у нас 4 таблицы в НТКП — 0,90,180,270 градусов. То при вызове SelectTableByAngle(160) будет использована таблица 180 градусов. Если, SelectTableByAngle(360) - таблица 0 градусов и т.д.

параметр fAngle — угол поворота (в углах, не в радианах), лежит в (-inf;+inf). В данной версии ограничен значениями float.

HRESULT SelectTableByVector([in] POINTF *pvec);

Выбор текущей таблицы из НТКП на основе вектора. Иными словами, вы вводите угол, а движок пытается вывести этот объект так, чтобы он был сонаправлен с вектором. Как и в SelectTableByAngle() берется ближайшее соответствие.

параметр pvec — вектор на плоскости

IParams интерфейс

Интерфейс параметров. Предоставляет доступ к позиции и размерам вызывающего объекта.

Методы

HRESULT SetPos([in] POINTF *pptPos);

Установка позиции.

параметр pptPos — левая верхняя точка — координата для вызывающего объекта

HRESULT Move([in]POINTF *pvecShift);

Перемещение

параметр pvecShift — вектор перемещения

HRESULT GetSmallestRad([in,out] float *pfRad);

Получение радиуса вписанной в прямоугольник, ограничивающий вызывающий объект, окружности.

выход pfRad — радиус

HRESULT GetBiggestRad([in,out] float *pfRad);

Получение радиуса описанной вокруг прямоугольника, ограничивающего вызывающий объект, окружности.

выход pfRad — радиус

HRESULT GetCoordsRect([in,out] RECT* rcCoords);

Получение прямоугольника, ограничивающего вызывающий объект, в мировых координатах.

выход rcCoords — прямоугольник, right и bottom которого соответствуют координатам правого нижнего угла

Похожие методы

GetCoordsWHRect()

HRESULT GetCoordsWHRect([in,out] RECT* rcCoordsWH);

Получение прямоугольника, ограничивающего вызывающий объект, в мировых координатах.

выход rcCoordsWH — прямоугольник, right и bottom которого соответствуют ширине и высоте прямоугольника

Похожие методы

GetCoordsRect()

HRESULT GetCoordsPt([in,out] POINTF* ptCoords);

Получение левой верхней координаты вызывающего объекта

выход ptCoords — точка в мировых координатах

HRESULT GetCentreCoordsPt([in,out] POINTF* ptCentreCoords);

Получение координат центра вызывающего объекта

выход ptCentreCoords — точка в мировых координатах

HRESULT GetWidthAndHeight([in,out] POINTF *pwhFrame);

Получение ширины и высоты вызывающего объекта.

выход pwhFrame — ширина и высота

HRESULT GetHalfWidthAndHeight([in,out] POINTF *pwhFrame);

Получение половины ширины и половины высоты вызывающего объекта.

выход pwhFrame — половина ширины и половина высоты

HRESULT GetWidth([in,out] int *pnWidth);

Получение ширины вызывающего объекта

выход pnWidth - ширина

HRESULT GetHeight([in,out] int *pnHeight);

Получение высоты вызывающего объекта

выход pnHeight - высота

HRESULT GetLX([in,out] int *pnLX);

Получение абсциссы левой грани прямоугольника, ограничивающего вызывающий объект

выход pnLX — x координата

HRESULT GetCX([in,out] int *pnCX);

Получение абсциссы центра прямоугольника, ограничивающего вызывающий объект

выход pnLX — x координата

HRESULT GetRX([in,out] int *pnRX);

Получение абсциссы правой грани прямоугольника, ограничивающего вызывающий объект

выход pnRX — x координата

HRESULT GetTY([in,out] int *pnTY);

Получение ординаты верхней грани прямоугольника, ограничивающего вызывающий объект

выход pnTY — y координата

HRESULT GetCY([in,out] int *pnCY);

Получение ординаты центра прямоугольника, ограничивающего вызывающий объект

выход pnCY — y координата

HRESULT GetBY([in,out] int *pnBY);

Получение ординаты нижней грани прямоугольника, ограничивающего вызывающий объект

выход pnBY — y координата

HRESULT Resize([in]int nNewWidth, int nNewHeight);

Изменение размеров изображения вызывающего объекта. Можно изменять размеры уже выводимых на экран объектов во время игры. Если несколько объектов используют одну и ту же графику (таблицу кадров) и один из них меняет размеры, то на других объектах эти изменения не сказываются.

параметр nNewWidth — ширина

параметр nNewHeight — высота

Терминология

Линейная анимация — анимация на таблице кадров, представляющая собой последовательный перебор ее кадров слева-направо сверху-вниз от i до j где i & j – порядковые номера кадров.

Будильник — таймер, отсчитывающий опред. число милисекунд и извещающий об окончании отсчета. Таймер отсчитывает каждый кадр, а извещает об окончании в течение одного кадра. В следующем кадре он возвращен в изначальное состояние.

Таблица кадров
Кадр
Набор таблиц кадров поворота (НТКП) — последовательность таблиц, кадры каждой из которых повернуты на определенный угол, по отношению к другой (исходной) таблице, на основе которой НТКП создана. Первая таблица из набора — всегда исходная. Используется визуальными объектами, которые нужно отображать под некоторым углом.
Замораживание кадра — остановка анимации на текущем кадре.
Простая анимация (Simple Anim) — анимация на таблице кадров, представляющая собой последовательный перебор всех ее кадров слева-направо сверху-вниз.
Анимация по умолчанию — анимация номер 0. См. SelectActiveAnim() интерфейса IAnim.
Список вывода (на экран)
Глубина вывода объекта (на экран)

Примеры

HRESULT VisResize([in]int nNewWidth, int nNewHeight);

```
int dyn_w = 50;
int dyn_h = 50;
int min_w = 10;
int min_h = 10;
int max_w = 100;
int max_h = 100;
int dir_w = 1;
int dir_h = 1;

void makeRidiculus() {
        dyn_w+=dir_w;
        dyn_h+=dir_h;
        if(dyn_w>max_w || dyn_w<=min_w) dir_w = -dir_w;
        if(dyn_h>max_h || dyn_h<=min_h) dir_h = -dir_h;
        izShip->pParams->Resize(dyn_w,dyn_h);
}

int Game_Main(void *parms,int num_parms) …
makeRidiculus();
```

Заставляет объект izship динамически менять размеры от 10 до 100.