47
2017

BLENDER: новости EEVEE

Новый вьюпорт Blender

Blender для начинающих

Освещение

Bsurfaces

И снова - ретопология

Разработка для PlayStation

Часть 5

dlib: практикум

+ многое другое!

Независимый электронно-познавательный журнал.
Издается с 2008 г. Доступен по CC-BY-NC-SA

FPS

«FPS» – бесплатный, свободно распространяемый электронный журнал, посвященный компьютерному творчеству.

«FPS» охватывает широкий круг тем: на страницах журнала рассматриваются вопросы программирования игр с использованием разнообразных движков и графических библиотек, публикуются материалы по двумерной и трехмерной компьютерной графике, включая уроки по популярным графическим редакторам, игровые обзоры, а также статьи по игровой теории, геймдизайну и современным мультимедийным формам искусства.

Журнал издается с января 2008 г.
Периодичность выхода: раз в два месяца.
Главный редактор: Тимур Гафаров
Дизайн и верстка: Тимур Гафаров, Наталия Чумакова
Обложка: Тимур Гафаров

Наш сайт: **http://fps-magazine.cf**
Наш адрес электронной почты: **gecko0307@gmail.com**

Blender

::. Новости
::. Eevee – новый вьюпорт
::. Blender для начинающих. Освещение
::. Bsurfaces. И снова – ретопология
::. Обзор дополнений. Выпуск 25

2D-графика

::. Новости

Программирование

::. Язык D: новости
::. dlib: практикум. Выпуск 1
::. Идеальный язык – это миф

Геймдев

::. Game Maker Studio 2
::. Разработка для PlayStation, часть 5
::. Фан-игры по мотивам Spyro

Linux

::. Игровые новости из мира СПО

BLENDER

Н О В О С Т И

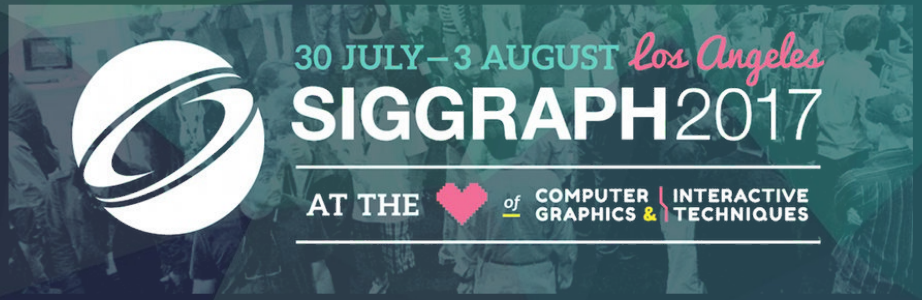
Вышли подряд два обновления Blender – **2.78b** и **2.78c**. Релизы, главным образом, нацелены на повышение производительности: внесено множество оптимизаций в Cycles, обеспечена поддержка многопоточной компиляции шейдеров, ускорены операции с мешами и текстурами, добавлена поддержка 3D-текстур для OpenCL, сокращено время построения графа зависимостей, ускорена многопоточная симуляция жидкостей.

Скачать Blender для всех платформ можно, как обычно, здесь: **https://www.blender.org/download**

Ежегодная инициативная программа **Google Summer of Code** одобрила семь проектов, касающихся Blender – в их числе такие интересные разработки, как интеграция Mantaflow, неявный скиннинг, улучшение режимов лепки и вершинного рисования, а также реализация пакетного менеджера для управления аддонами.

Кстати, уже существует проект Blender Add-on Updater от CGCookie – дополнение, позволяющее обновлять установленные аддоны напрямую из репозиториев, что можно считать первым шагом на пути к полноценному пакетному менеджеру. Найти его можно тут: **https://github.com/CGCookie/blender-addon-updater**.

Blender Foundation приглашает всех желающих на предстоящую конференцию SIGGRAPH 2017, которая пройдет с 30 июля по 3 августа в Лос-Анджелесе. На 30 июля запланирована встреча сообщества пользователей и разработчиков Blender, в ходе которой будут озвучены планы на ближайший год, а художники, аниматоры и программисты смогут выступить с докладами и презентациями. Зарегистрироваться в качестве участника можно здесь: **https://www.blender.org/events**



Blender Institute снимает свой первый полнометражный анимационный фильм – «Agent 327». Это экранизация одноименной серии комиксов голландского писателя Мартина Лодевейка, которая выпускается аж с 1966 года по нынешний день. Комикс повествует о приключениях Хендрика Айцербрута, известного также как Агент 327 – сотрудника нидерландской секретной спецслужбы. Автором сценария для фильма выступает сам Лодевейк, режиссером – исландский аниматор Хъялти Хъялмарссон, продюсером – как обычно, Тон Розендаль. Уже вышел тизер фильма – смотрим на YouTube-канале BI: **https://www.youtube.com/watch?v=mN0zPOpADL4**

Blend4Web, открытый фреймворк для создания браузерных 3D-приложений в Blender, обновился до версии 17.04. Из особенностей релиза – поддержка WebAssembly в физическом движке (Uranium.js), поддержка WebVR для контроллеров HTC Vive, сжатие ресурсов в формат gzip, улучшенный LOD.

https://www.blend4web.com

Вышлел первый стабильный релиз SceneTerrain, генератора ландшафтов для Blender. Появилась поддержка слоев популяций (population layers) для управления распределением растительности по ландшафту, добавлены два новых вида деревьев, введена рандомизация цвета листьев, улучшен GUI.

SceneTerrain по-прежнему поставляется только вместе с генератором городов SceneCity. Цена комплекта – $84.00.

http://cgchan.com/store/scenecity

Анонсирован интересный проект OD_CopyPasteExternal – инструмент для копипастинга 3D-геометрии между разными программами. Поддерживаются Blender, 3ds Max, Maya, Houdini, LightWave, Modo, SketchUp и Rhino. Теперь необязательно мучиться с FBX или OBJ, сталкиваясь с несовместимостями, потерями данных и багами в экспортерах – обмен геометрией между приложениями становится таким же простым, как копирование и вставка текста!

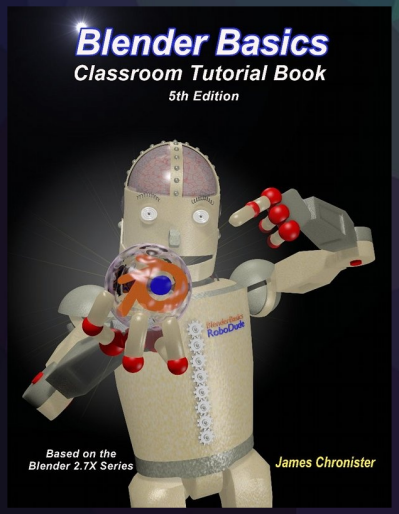
https://heimlich1024.github.io/OD_CopyPasteExternal

RENDERMAN

Вышло обновление знаменитого рендер-движка RenderMan от студии Pixar – версия 21.4. Релиз включает генератор текстур для волос (PxrHairColor), инструменты для работы со случайными текстурами, новые опции для источников света PxrRectLight, PxrDomeLight и PxrPortalLight, а также улучшенный интегратор PxrVCM.

Напомним, с 2015 года RenderMan доступен бесплатно для использования в некоммерческих целях. Бесплатная версия полностью функциональна, не имеет временных ограничений и не накладывает водяные знаки. Движок интегрируется в Maya, Katana, Houdini и Blender. Стоимость коммерческой лицензии – $495.

https://renderman.pixar.com/view/non-commercial-renderman



К книжным анонсам. Джеймс Кронистер выпустил пятое издание своей книги «Blender Basics Classroom Book», которая аж с 2004 года помогает начинающим пользователям сделать первые шаги в освоении Blender. В новом издании добавлены главы, посвященные Cycles, 3D-печати, отслеживанию движений. Как всегда, книгу можно скачать бесплатно, распространяется в формате PDF.

С нетерпением ждем русского перевода!

http://www.cdschools.org/blenderbasics

На BlenderMarket поступила в продажу книга «Blender Add-on Cookbook» Мишеля Андерса – практическое руководство и сборник рецептов для разработчиков дополнений Blender. Книга посвящена, главным образом, нетривиальным вопросам и различным тонкостям аддоностроительства. Книга доступна в форматах PDF, ePub и Mobi. Цена – $7.99.

https://blendermarket.com/products/blender-add-on-cookbook

Журнал «FPS» отслеживает все самые свежие новости из мира Blender, моделирования, анимации и рендеринга! В следующем номере ждите очередную подборку новостей – оставайтесь с нами и держите руку на пульсе последних событий!

Теперь, когда вьюпорт Blender использует OpenGL 2, открыта дорога для разного рода инноваций. Вьюпорт уже сейчас поддерживает интересные эффекты вроде SSAO, но разработчики на этом не остановились – в рамках ветки Blender 2.8x полным ходом идет реализация нового движка фотореалистического рендеринга в реальном времени Eevee, о котором хочется рассказать подробнее.

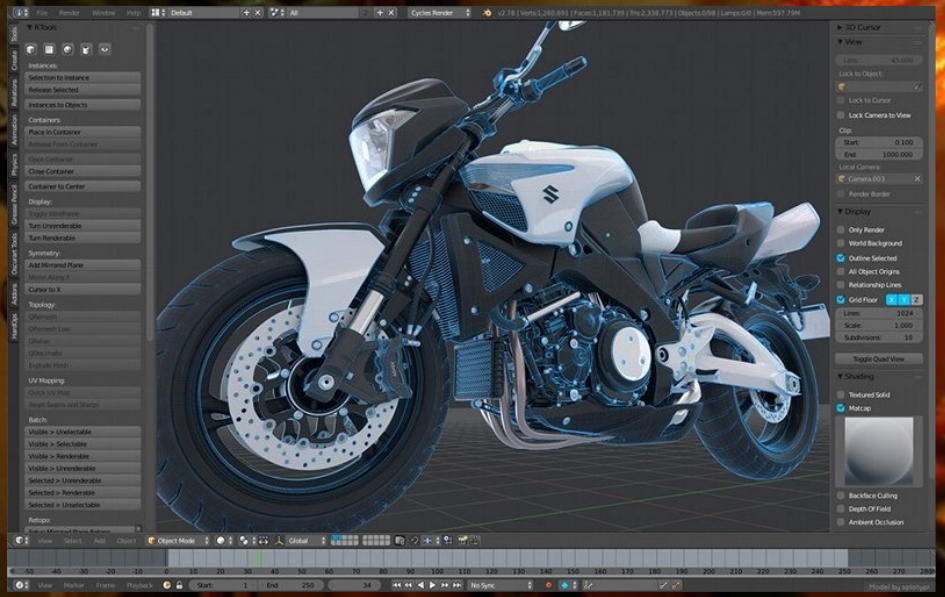
EEVEE

Название Eevee расшифровывается как «Extra Easy Virtual Environment Engine». Движок будет основан на физически обоснованном рендеринге (Physically Based Rendering, PBR) – мы уже неоднократно писали об этой технологии, которая сейчас является де-факто стандартом индустрии CG. Чего же ожидать от Eevee и какие возможности он принесет простым пользователям?

В первую очередь, будет пересмотрена система освещения – добавление на сцену новых источников света не будет приводить к тормозам. Информация об источниках света будет передаваться на GPU при помощи UBO (Uniform Buffer Object) – то есть, динамических массивов, что позволит избежать перекомпиляции шейдеров.

Будет реализована концепция убер-шейдеров – то есть, универсальных шейдеров, из которых можно выкрутить любой материал путем изменения параметров. В качестве образца разработчики взяли PBR-материалы из Unreal Engine 4. Предполагается, что убер-шейдер будет выходным узлом для материалов в композиторе, причем совместимым с Cycles – любую Cycles-сцену можно будет легко адаптировать под рендеринг в Eevee.

Что будет поддерживать убер-шейдер? Во-первых, освещение на основе изображений (Image Based Lighting) – то есть, HDR-карты окружения, как в Cycles. Загружаемые из внешних файлов световые зонды будут дополняться внутренними – для расчета непрямого освещения от объектов сцены. Для максимальной отзывчивости генерирование внутренних зондов будет выполняться в фоновом режиме, пока вьюпорт рендерит упрощенную картинку – например, только с рассеянным непрямым освещением.



Данные об освещении могут быть сохранены в blend-файл. Для рассеянного света будут использованы сферические гармоники, для отраженного – кубические карты.

Eevee будет поддерживать подповерхностное рассеивание (SSS), эффект многослойных материалов (Clear Coat), рендеринг объемов (Volumetric) – а также полный набор стандартных возможностей: размытие при движении, DoF, антиалиасинг и т.д. Не исключена поддержка отражений в экранном пространстве (Screen Space Reflection). В принципе, Eevee в будущем может заменить собой Blender Internal – будет реализована частичная совместимость с материалами BI.

А теперь – ложка дегтя. Все эти возможности будут требовать достаточно мощной видеокарты – для работы Eevee потребуется поддержка OpenGL 3.2.

Blender перестает быть подходящим решением для работы с 3D-графикой на слабых машинах, превращаясь в инструмент для профессиональных рабочих станций. С одной стороны, это, конечно, неплохо – это означает, что Blender используется для серьезных задач и востребован в коммерческой среде. Но с другой – уже не будет того былого кайфа, когда вы могли взять старенький нетбук, смоделировать какую-нибудь несложную сцену и тут же ее отрендерить. Если вам от 3D-пакета не нужны такие навороты, как PBR во вьюпорте или GPU-рендеринг, то вскоре придется искать альтернативу – например, взглянуть в сторону Wings 3D. Смешно сказать – когда-то многие из нас переходили с 3ds Max на Blender именно из-за его легковесности и низким системным требованиям, а теперь снова ищем что-то полегче...

Источники освещения

Создать реалистичную сцену невозможно без правильно настроенных источников света (ламп). Это особые объекты, которые учитываются рендер-движком во время расчета освещенности объектов. Лампы могут быть нескольких типов:

- Point (точечный свет) – светит одинаково во всех направлениях из одной точки, как, скажем, электрическая лампочка;

- Sun (солнечный свет) – имитирует удаленные источники света, например, солнце. Считается, что такая лампа бесконечно удалена от сцены – все световые лучи являются параллельными и направлены вдоль заданной оси;

- Spot (направленный свет) – похож на Point, но светит не во всех направлениях, а в пределах конуса. Освещаются только объекты, попадающие в этот конус, как в случае с карманным фонариком;

- Hemi (от «hemishpere» – «полусфера») – равномерный свет, идущий от поверхности бесконечно удаленной полусферы. Пример такого освещения в реальности – дневное небо в пасмурную погоду;

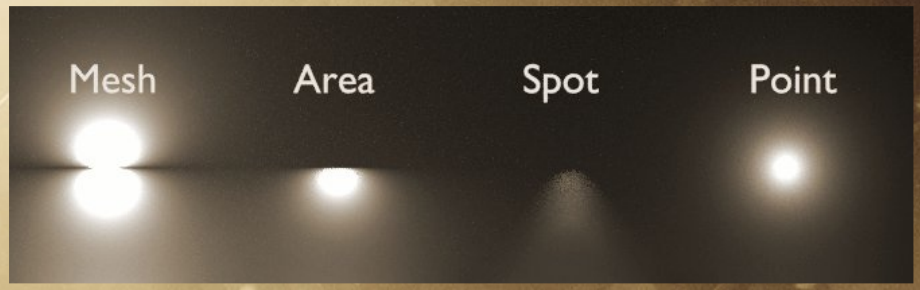
- Area (площадь) – свет, идущий от прямоугольной поверхности. Эту лампу можно использовать для имитации света из окна, от дисплеев, лайтбоксов и т.д.

У ламп есть параметр яркости – Energy: чем он выше, тем больше световой энергии излучает источник. Кроме того, любую лампу можно сделать «негативной» (опция Negative) – в этом случае она будет не излучать, а поглощать свет, создавая своеобразную искусственную тень.

Кстати, о тенях. Любой объект, для материала которого активирована тень, будет отбрасывать ее на другие объекты. Вы можете проверить это, создав плоскость и «поставив» на нее какой-нибудь примитив – скажем, куб.

По умолчанию тени резкие и черные, что не очень реалистично. Мы можем это изменить: перейдите в настройки лампы и на вкладке Shadow увеличьте количество сэмплов тени (Samples), например, на 4. Параметр Soft Size (величина размытия) выставьте равным 0.5. Цвет тени сделайте не черным, а серым. Теперь тень станет прозрачной и нечеткой по краям, как и в реальной жизни. Качество размытия повышается увеличением количества сэмплов.

Можно сделать тень цветной. В Blender даже есть возможность использования цвета материала для создания разноцветных теней! Это позволяет добиться любопытнейших эффектов. Один из них – «кинопроектор»: прозрачное изображение просвечивается, и полученная тень отбрасывается на экран напротив.



Создайте плоскость и поверните ее вертикально – это у нас будет экран. Создайте новую лампу и укажите ей тип Spot. Поместите лампу на нужном расстоянии от экрана и поверните на 90 градусов. Переключите ее режим тени на Ray Shadow. На вкладке Spot Shape активируйте опцию Halo и измените параметр Size на 30.

Создайте новую вертикальную плоскость, поменьше. Поместите ее на близком расстоянии от лампы. Если нужно – измените пропорции. Эта плоскость будет выступать в качестве слайда.

Создайте для слайда материал. Включите прозрачность (Transparency) и уменьшите значение Alpha до 0.2. Добавьте новую текстуру и загрузите картинку с альфа-каналом. Важно, чтобы картинка была полупрозрачной, как реальный слайд – это надо заранее сделать в графическом редакторе. В параметрах влияния укажите Color и Alpha.

Выберите плоскость-экран и создайте ей новый материал. На вкладке Shadow активируйте опцию Receive Transparent. Теперь экран будет принимать любые прозрачные тени. Вы можете создать несколько других объектов с этим же материалом и поставить их между экраном и слайдом – на них вы тоже увидите цветную проекцию!

«Blender. Настольная книга» доступна онлайн!

Мы долго думали, как быть с нашим многострадальным проектом «Blender. Настольная книга». Изначально мы собирались выпустить ее в виде PDF, с полноценным оформлением и обложкой, но сейчас стало очевидно, что над этим стоит задумываться только после написания текста – все-таки верстка PDF-книги дело само по себе достаточно серьезное и очень непростое. Сами тексты готовых статей когда-то были доступны на нашем старом сайте, но его уже давно нет, а нынешний плохо подходит для публикации подобного контента. В итоге мы решили опубликовать все, что есть, на GitHub Pages:

https://gecko0307.github.io/blender-handbook

Книга была переверстана в разметке Markdown и размещена в **git-репозитории**, чтобы к проекту могли присоединиться все желающие (Markdown-исходники автоматически публикуются в GitHub Pages). Вы можете написать новую статью или улучшить существующую – делайте pull request'ы.

Напомним, книга представляет собой сборник статей, охватывающих различные аспекты использования Blender, скомпонованных по принципу «от простого к сложному». В настоящий момент готовы вступительная часть, первая глава и частично вторая. Мы надеемся, что в дальнейшем статьи будут появляться регулярно – одновременно с выходом номеров журнала.

BSURFACES

И снова – ретопология

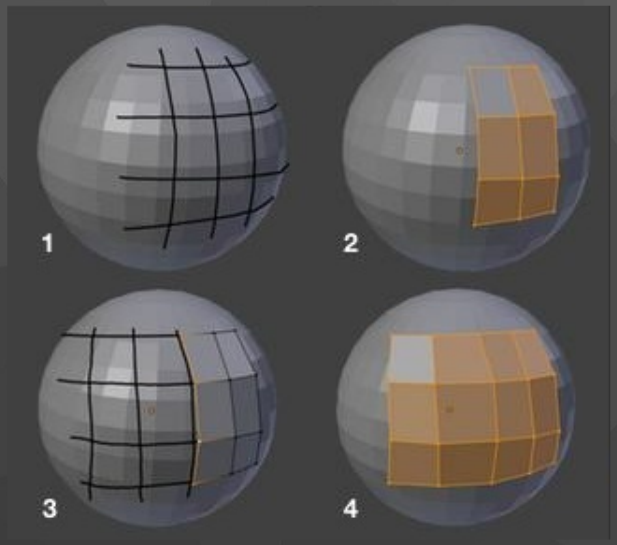
В «FPS» №44 '16 мы писали о замечательном аддоне RetopoFlow, сильно облегчающем процесс ручной ретопологии в Blender. Но мало кто знает, что есть еще один мощный аналогичный инструмент – встроенный аддон Bsurfaces.

Когда-то он был коммерческим, но в 2012 году обзавелся GPL-версией и в итоге стал частью дистрибутива Blender. На фоне RetopoFlow он оказался незаслуженно забыт, несмотря на то, что представляет собой очень простой и в то же время мощный инструмент, которым нельзя пренебрегать.

Bsurfaces работает на основе штрихов Воскового карандаша (Grease Pencil) и создает геометрию из квадов, соединяющую эти штрихи. Поскольку в Blender есть возможность наносить штрихи на поверхность модели, весь процесс ретопологии сводится к рисованию линий-направляющих, которыми вы даете системе понять, как ориентировать сетку. Фактически, Bsurfaces реализует удачный компромисс между ручной и автоматической ретопологией.

Работать с Bsurfaces очень просто. Загрузите в Blender ваш высокополигональный скульпт и создайте плоскость – она будет служить объектом для сохранения новой сетки, саму плоскость мы потом удалим. Выделите плоскость и создайте новый датаблок и слой для Grease Pencil. Во вкладке Grease Pencil на панели инструментов переключите Data Source на Object. Переключите Stroke Placement на Surface и начинайте рисовать (кнопка Draw). Можно включить режим Contunious Drawing, чтобы нарисовать много штрихов за раз. Основной принцип – штрихи нужно наносить строго в одном направлении: то есть, всегда слева направо или наоборот. Если наносить их как попало, Bsurfaces выдаст неправильную геометрию.

Закончив разметку, нажмите Enter и перейдите в режим редактирования меша (Tab). На вертикальной вкладке Tools вы найдете вкладку Bsurfaces. Нажмите кнопку Add Surface. Если все сделано правильно, Bsurfaces превратит штрихи в красивую правильную сетку. Можно наносить штрихи не только параллельно, но и «крест-накрест» – в данном случае вы дадите системе понять, на сколько квадов разбивать поверхность – сколько линий нарисуете, столько ребер и получите.



Bsurfaces – отличный инструмент для черновой ретопологии. С его помощью можно быстро наметить основные поверхности всей модели по отдельности, а затем «доводить» сетку вручную, соединяя все куски воедино. На мой взгляд, это гораздо проще, чем делать все Polypen'ом в RetopoFlow. Я уж не говорю о том, что оба аддона ничто не мешает использовать вместе – и ретопология из скучной тяжелой рутины превратится в сущее удовольствие!

Тимур Гафаров

P.S. Помимо Bsurfaces и RetopoFlow есть еще один интересный аддон, предоставляющий средства ретопологии – это Mira Tools. Это целый комплект разнообразных инструментов моделирования, среди которых выделяется CurveSurfaces – аналог Bsurfaces, отличающийся тем, что использует не Восковый карандаш, а кривые. Как следствие, пользователь имеет возможность быстро редактировать полученную сетку, просто меняя кривые-направляющие. В следующий раз мы обязательно напишем о Mira Tools поподробнее.

https://github.com/mifth/mifthtools

ОБЗОР ДОПОЛНЕНИЙ BLENDER

Выпуск 25

Благодаря удобному и мощному API для языка Python, Blender поддается практически неограниченному расширению. В этом выпуске мы представляем дополнения, связанные с моделированием твердых поверхностей (hard surfaces). Если вы разрабатываете собственное дополнение или просто нашли в Интернете чей-то интересный проект, будем очень рады, если сообщите нам об этом и поделитесь ссылкой. Пишите на наш ящик: gecko0307@gmail.com.

Ice Tools Pro

Более дешевый аналог Hard Ops. Предоставляет неразрушающий поток работы для моделирования твердых поверхностей, включает инструменты ретопологии.

Аддон коммерческий, цена – $12.

Разработчик: Blender Guppy
https://blendermarket.com/products/ice-tools-pro

Hard Ops

Мало кто не слышал о Hard Ops, однако в нашем журнале мы о нем до сих пор не упоминали – исправим это недоразумение.

Hardops – это набор инструментов для моделирования твердых поверхностей: всяческих деталей, механизмов и техники. Аддон облегчает и ускоряет работу с булевыми операциями, позволяет создавать фаски и гнезда под болты, умеет создавать UV-развертки и еще много чего.

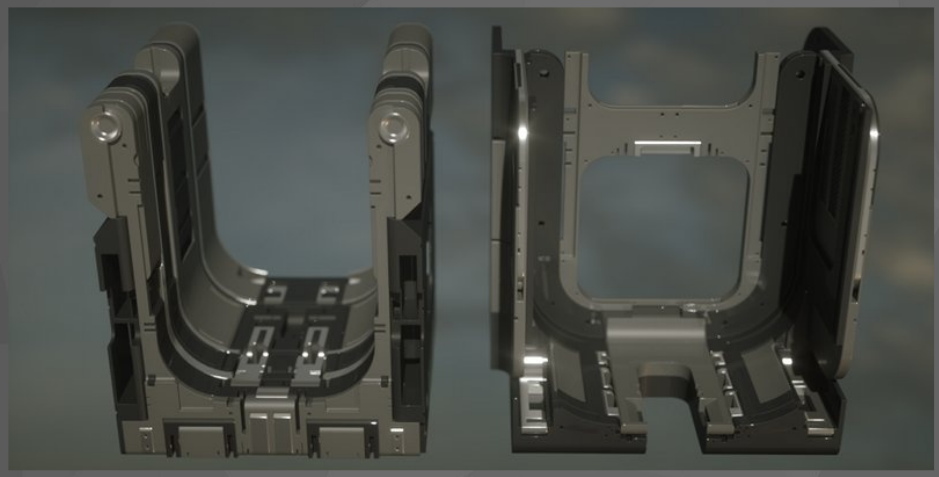
Дополнение платное, продается через Gumroad по произвольной цене от $15.

Разработчик: masterXeon1001
https://gumroad.com/l/hardops

Box Cutter

Упрощенный вариант Hard Ops от того же автора – облегчает булевые операции с мешами. Возможна совместная работа с Hard Ops. Дополнение платное, продается через Gumroad по произвольной цене от $15.

Разработчик: masterXeon1001
https://gumroad.com/l/BoxCutter



Decal Machine

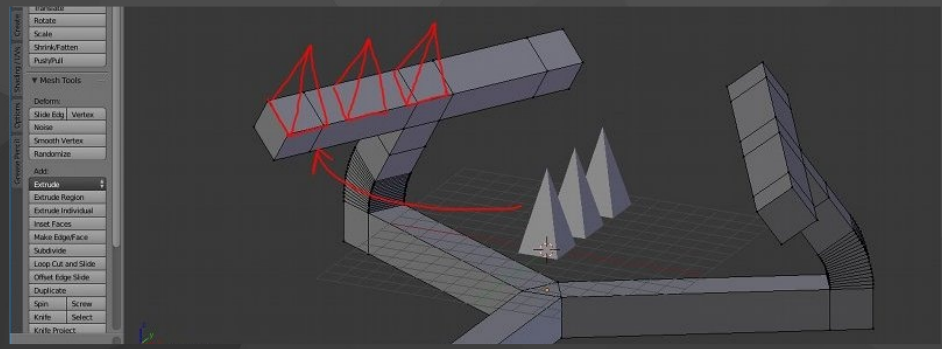
Дополнение, автоматизирующее работу с декалями – картинками, которые накладываются на поверхности. Декали используются, чтобы разнообразить поверхность мелкими деталями – это могут быть дырки от пуль, болты, царапины, надписи на стенах, плакаты и т.д. В данном случае основной упор сделан на работу с механическими деталями, разнообразными техническими выемками и отверстями. Дополнение платное, цена – $15.

Разработчик: MACHIN3
https://blendermarket.com/products/DECALmachine

Mesh Align Plus

Инструмент для точных выровненных операций с мешами – выручит, например, если вам нужно «приклеить» объект к грани или повернуть вокруг ребра. Сильно облегчает CAD-задачи, моделирование твердых поверхностей и интерьеров. Дополнение бесплатное.

Разработчик: Eric Gentry
https://github.com/egtwobits/mesh-align-plus



Новости графического СПО

GIMP 2.8.22

Вышел корректирующий релиз стабильной ветки свободного графического редактора GIMP – 2.8.22. В этой версии внесены изменения в организацию иерархии окон в однооконном режиме, что позволило добиться увеличения призводительности отрисовки при использовании некоторых видов тем оформления GTK+. Также устранено несколько багов, в том числе уязвимость, связанная с форматом ICO.

https://www.gimp.org

Krita 3.1.3

Увидела свет новая версия свободного графического редактора Krita – 3.1.3. Релиз исправляет несколько ошибок, включает возможность одновременного запуска нескольких экземпляров Krita и функцию для масштабирования вокруг точки привязки. Также представлен новый выпуск плагина для интеграции с Проводником Windows, отображающего эскизы для файлов в форматах .kra и .ora.

https://krita.org

Волшебница Пеппер и ее котенок Кэррот, придуманные французским художником-иллюстратором Давидом Ревуа, оживают в анимированном комиксе от Morevna Project, создателей российского открытого фильма «Моревна». В конце прошлого года была проведена кампания по сбору средств, и недавно этот проект был завершен!

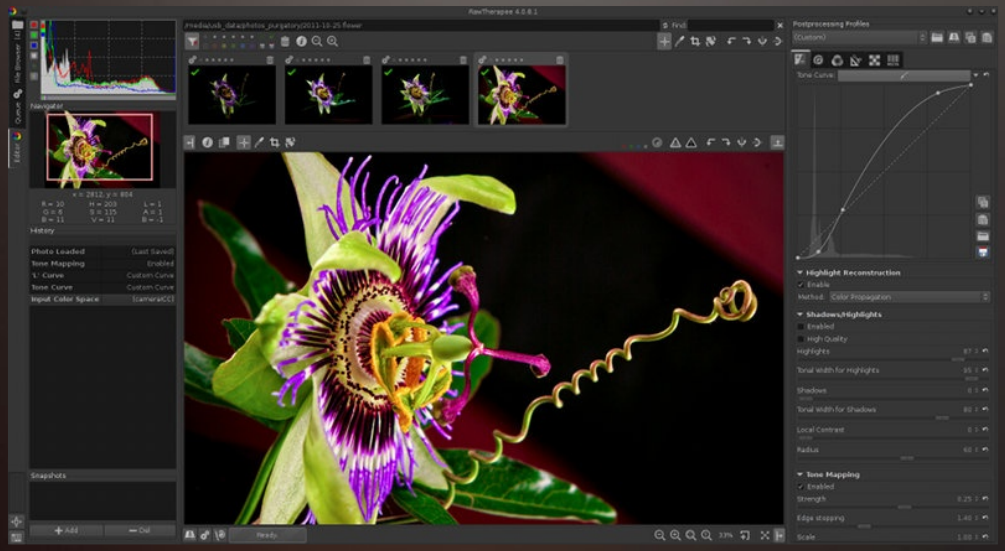
Сюжет основан на выпуске комикса №6 – «The Potion Contest». Есть два варианта озвучки – русская и английская. Вся творческая работа, как и полагается, выполнена при помощи СПО, а именно Krita, Blender, CoaTools, Papagayo-NG, RenderChan.

https://www.youtube.com/embed/iY2cuLJHAAA

RawTherapee 5.0

Вышла пятая версия свободного RAW-проявщика RawTherapee. Из особенностей релиза отметим доработку вейвлетных инструментов для повышения резкости и подавления шума. Также улучшены инструменты цветокоррекции, управление цветом и поддержка RAW-форматов.

http://rawtherapee.com



Natron 2.2

Natron, свободный пакет композитинга, обновился до версии 2.2. Из особенностей релиза – поддержка 32-битной macOS, новые форматы (HD_720, UHD_4K, 2K_DCP, 4K_DCP), множество новых OFX-плагинов.

http://natron.fr

Уважаемые читатели!

Наш журнал регулярно выходит уже 9 лет – с февраля 2008 года. Все эти годы он оставался бесплатным изданием, предлагая публике эксклюзивный контент с минимумом рекламы. Мы всегда работали на совесть – не ради денег, а на благо наших читателей. «FPS» был и остается проектом энтузиастов и полностью независимым изданием – мы не защищаем интересы корпораций или политиков, мы пишем о том, что считаем нужным и важным. Мы стоим за свободу слова и творчества, за обмен информацией и знаниями: все материалы журнала можно беспрепятственно копировать, распространять и использовать в любых производных работах.

И мы надеемся, что так будет продолжаться и дальше. Но у авторов на создание новых номеров уходит достаточно много сил и времени, которые никак материально не компенсируются. Авторы вынуждены зарабатывать деньги другими способами, в результате чего работа над журналом теряет приоритетность. При отсутствии финансовых вложений мы не можем гарантировать, что «FPS» продолжит свое существование. Поэтому, если вам нравится журнал, и вы хотели бы, чтобы он жил, развивался, становился больше и качественнее, просим **поддержать его электронной валютой** – при помощи WebMoney, PayPal или Яндекс.Денег, любой суммой на ваше усмотрение. Для нас важен любой, даже маленький вклад!

Наш WMR-кошелек: R120156543694

Адрес PayPal:
gecko0307@gmail.com

Яндекс.Деньги:
410012052560079

Заранее благодарны!

Язык D. Новости

Мы решили перейти на новый формат новостей, отказавшись от деления на тематические разделы – большинство библиотек на D имеют довольно широкую сферу применения и не очень-то поддаются строгой классификации. К тому же, в связи с растущей популярностью D, анонсов и обновлений в последнее время очень много, и отслеживать их все в рамках журнала просто нет возможности. Поэтому мы будем публиковать в этой рубрике только самые важные релизы, а также анонсы проектов с упором на геймдев.

DMD 2.074.0

Вышла новая версия референсного компилятора D – DMD 2.074.0. Спецификация языка не изменилась, однако релиз включает несколько важных улучшений в Phobos, а также множество багфиксов.

http://dlang.org/download.html

LDC 1.2.0

Увидела свет LDC 1.2.0 – новая версия компилятора D с LLVM в качестве бэкенда. Релиз основан на фронтенде и рантайме D 2.072.2, включает поддержку LLVM 3.5-4.0. Из основных изменений стоит отметить улучшенную обратную трассировку стека и полноценную поддержку типа real с учетом возможностей целевой платформы. Полным ходом идет работа над следующей версией компилятора, 1.3.0, в которой будет официальная полная поддержка Android!

https://github.com/ldc-developers/ldc

Moonshot – транслятор D в Lua

Очень интересный проект, нацеленный на превращение D в скриптовый язык путем трансляции в Lua AST. Moonshot основан на DMD 2.074 с модифицированным Phobos, и оддерживает большую часть спецификации языка, включая шаблоны, диапазоны и foreach.

https://github.com/Philpax/moonshot

SpaceD

Вышла новая игра, написанная на D – космические гонки под названием SpaceD. Игра была создана для конкурса Linux Game Jam 2017. Главная особенность проекта – генератор случайных трасс, а также встроенный редактор для создания своих трасс. SpaceD доступна для Windows и Linux.

https://webfreak.itch.io/spaced

dlib 0.11.0

Коллекция библиотек dlib обновилась до версии 0.11.0. Из основных улучшений – переход на аллокаторы dlib.memory для примитивов New и Delete, новый аллокатор GCallocator, основанный на сборщике мусора D, полноценная поддержка анимированных PNG (APNG) в dlib.image, а также новая функция обхода каталогов traverseDir, не задействующая сборщик мусора.

Напомним, dlib – это набор базовых библиотек для создания игровых движков, мультимедийных и научно-инженерных приложений. Включает пакеты линейной алгебры и вычислительной геометрии, средства управления памятью и абстрактного потокового ввода-вывода, инструменты обработки изображений и звука, быстрый лексический анализатор, XML-парсер и многое другое.

https://github.com/gecko0307/dlib

vibe.d 0.7.31

Вышло обновление ветки vibe.d 0.7, включающая обратные порты некоторых мелких изменений ветки 0.8. Напомним, что ветка 0.7.x будет поддерживаться одновременно с 0.8.x, однако новых возможностей в ней не будет – разработка фреймворка продолжается в ветке 0.8.

http://vibed.org

Coedit 3

Увидела свет третья версия Coedit, нативной IDE для D. Релиз включает визуальный интерфейс к отладчику GDB, поддержку групп проектов, а также возможность собирать проекты несколькими разными компиляторами.

https://github.com/BBasile/Coedit

Visual D 0.44

Райнер Шутце анонсировал новую версию Visual D – проекта по интеграции D в среду разработки MS Visual Studio. В новой версии обеспечена поддержка смешанных проектов на C/C++ и D.

http://rainers.github.io/visuald

Intellij D

Анонсирован плагин к Intellij IDEA, обеспечивающий поддержку D.

https://github.com/kingsleyh/DLanguage

DCompute

DCompute – это расширение компилятора LDC для параллельных вычислений на GPU, генерирующее код NVPTX для CUDA и SPIRV для OpenCL.

https://github.com/ldc-developers/ldc/tree/dcompute

Mir GLAS

Для D развивается нативная реализация BLAS/GLAS, де-факто стандартного набора функций линейной алгебры. В синтетических тестах Mir GLAS значительно обгоняет по производительности OpenBLAS и показывает сопоставимые результаты с пакетом Intel MKL. Проект доказывает, что на D вполне возможны высокоэффективные вычисления, и это не может не радовать.

https://github.com/libmir/mir-glas

DIOS

Анонсирован проект DIOS – минимальное ядро операционной системы для x86, написанное на D. Представляет собой программу уровнем чуть выше «Hello, World» – умеет печатать текст в VGA-режиме, включает поддержку Multiboot. Система оформлена как Live CD, в качестве загрузчика используется GRUB. Проект может пригодиться в качестве основы для построения полноценной ОС.

https://github.com/gecko0307/dios

dlib: практикум

Выпуск 1. Фокусы с Compound

Недавно я столкнулся с интересной задачей: описанием класса, который инкапсулирует неизвестный заранее набор объектов. Типы этих объектов задаются через variadic template – то есть, шаблон с переменным количеством параметров. Статические массивы для этой задачи, ясное дело, не годятся, но тут идеально подошел Compound из dlib.core.compound. Это своеобразный «гибрид» кортежа и структуры – составной тип данных, который можно создавать в шаблонах. Ему можно передавать кортеж параметров variadic-шаблона, и это позволяет проделывать замечательные вещи.

```
class Collection(C...)
{
    protected Compound!C _components;

    this(C comps)
    {
        _components = comps;
    }
}

auto inc = new Collection!(int, char, bool)(10, 'a', false);
```

Проблема с обычными кортежами (Tuple) в том, что их нельзя возвращать из функции. Но Compound – можно.

Таким образом, для ридонли-доступа к _components мы можем объявить следующий метод:

```
auto components() @property
{
    return _components;
}
```

А еще мы можем заполнять Compound в цикле, читая значения, например, из массива:

```
this(int[] arr)
{
    foreach(i, T; C)
    {
        _components[i] = cast(T)arr[i];
    }
}

auto inc = new Collection!(int, char, bool)([5, 40, 0]);
```

Пример бессмысленный, но наглядный – мы проходим по всем типам кортежа C и записываем входное значение в наш Compound, конвертируя в нужный тип. Без дополнительных проверок это небезопасно, но если вы знаете, что делаете – очень полезно.

Тимур Гафаров

Идеальный язык – это миф

Прошло семь лет с тех пор, как я опубликовал свою первую статью о языке D в журнале «FPS» (№12 за 2010 г.). Первоначальный восторг и энтузиазм за это время, конечно, поугас, но в целом мое отношение к языку не поменялось, я все еще остаюсь убежденным дишником – ничего лучше D я пока не вижу. Но я отнюдь не считаю D идеальным языком. За эти годы у меня накопилось много соображений по этому поводу, которыми сейчас и хочу поделиться.

Есть популярное мнение, что программисты СПО не должны распылять усилия на персональные проекты и разрабатывать только один, чтобы получился идеальный инструмент. Отчасти это мнение распространяется и на языки – кого-то раздражает «зоопарк» языков, среди которых, мол, нет однозначно лучшего. Так вот, идеальный язык программирования – это нонсенс, он не может существовать. По той простой причине, что языки – это инструменты, предназначенные для решения определенных задач. Немыслим такой инструмент, который решал бы все возможные задачи. Более того, по мере развития информационных технологий появляются новые задачи, для решения которых создаются новые инструменты. И это нормально, так было всегда. Чем больше языков, тем лучше – те, кто считает иначе, гонятся за иллюзией.

Можно выразить это через биологическую аналогию. Языки (и вообще все свободные программы) существуют в некоем подобии экосистемы, и к ним отчасти применимы законы естественного отбора. Выживают наиболее приспособленные – в данном случае, приспособленные к решению определенного набора задач. Схожие задачи объединяют языки в «ареалы обитания», где они конкурируют друг с другом.

Подобно живым организмам, языки в принципе не способны охватить все ареалы – нет существ, которые были бы одинаково хорошо приспособлены к жизни в пустыне, в тропиках, в арктических льдах и в океане.

Поэтому мне всегда казалась нелепой тенденция учить один-единственный язык, чтобы затем использовать его для любых целей. Если язык пригоден для решения вашей задачи – используйте, если нет – берите более подходящий.

Когда люди спрашивают что-то вроде: «Почему D непопулярен несмотря на то, что он такой идеальный?» – некорректна сама постановка вопроса. D – неидеальный язык. И C++ неидеальный, и Java, и Python, и все остальные. D спроектирован лучше, чем C++, Java или Python – но этот факт сам по себе не означает, что D решает все задачи однозначно лучше, чем C++, Java или Python.

Практическая ценность языка вообще мало коррелирует с его дизайнерскими достоинствами. D – очень красивый язык, вобравший в себя множество потрясающих архитектурных решений. На D можно писать образцово эффективный и читаемый код. Но это дизайнерское достоинство, которое сделало бы D идеальным языком только в идеальном мире, где нет необходимости линковаться с библиотеками на C++, взаимодействовать с морально устаревшими API типа Win32, писать клиентский код под браузеры, разрабатывать мобильные приложения под Android с его урезанным libc и отсутствием полной поддержки Posix. Список можно продолжать бесконечно: мир IT – это архитектурный хаос. Есть хорошо продуманные стандарты, ставшие популярными, но они теряются на фоне нагромождения разнородных поделок, «склеенных скотчем», наспех написанных по принципу «лишь бы работало».

Популярными становятся те инструменты, которые помогают уживаться со всем этим, не стремясь быть «вещью в себе». Есть инструменты, безупречные с точки зрения дизайна, но совершенно бесполезные на практике – например, язык Haskell или ОС Plan 9. Это классические примеры «вещей в себе», интересные лишь в эстетическом отношении. На практике никто не использует красивые языки и красивые операционные системы только из-за их красоты. В этом причина популярности Windows и C++. Никто не говорит, что они идеальны, но все ими пользуются – потому что они решают задачи большинства.

Но вернемся к D. Решает ли он какие-то задачи лучше, чем C++? Если да, то почему непопулярен? Это непростой вопрос, на который, наверное, нельзя ответить однозначно, но я попробую.

Я не раз сталкивался с необходимостью написать небольшую утилиту, эффективно выполняющую одну-единственную задачу. Например, сортировщик файлов, переименовывающий их в определенной последовательности. На D без дополнительных библиотек написать такое намного проще, чем на C++ без дополнительных библиотек. И работать оно будет значительно быстрее, чем, скажем, вариант на Python или Bash (и я бы еще поспорил, что проще – Bash или D).

D позволяет без особых телодвижений взять и реализовать с нуля любую идею – в этом его главная практическая ценность. Конечно, без библиотек это не всегда так просто, поэтому я и начал проект dlib – это, в первую очередь, именно библиотека для прототипирования. Суть в том, что D как язык располагает к быстрому воплощению идей, при этом предоставляя нужную производительность – с D не требуется переписывать прототип на язык с более низким уровнем абстракции, прототип с минимальными трудозатратами превращается в законченный продукт.

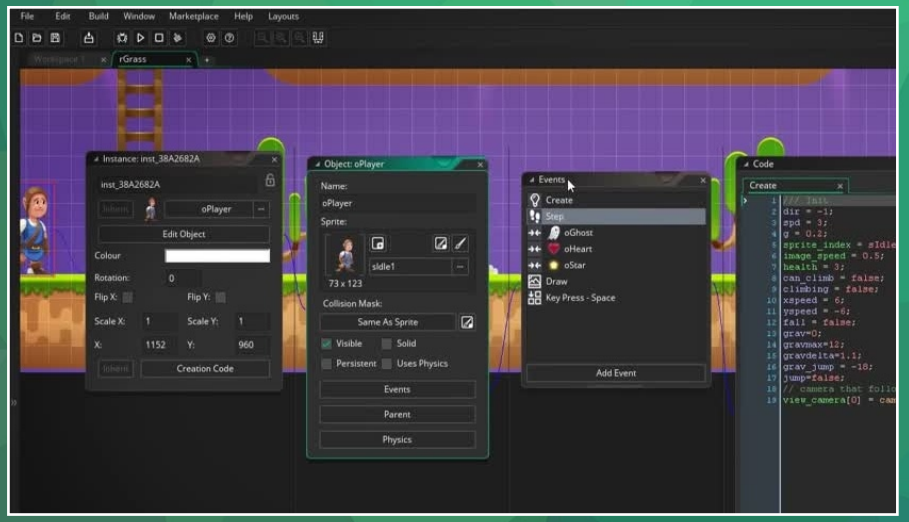
Может ли D конкурировать с C++ на вышеупомянутом хаотическом поприще? Да, но только при создании нового. Нет смысла писать на нем, если ваш код будет мешаниной из библиотечных вызовов, C-шных идиом и разных костылей типа toStringz. В таком случае вы просто усложните себе жизнь – намного проще использовать C++. D – это язык не для рутины, а для творчества. И в этом у него нет конкурентов.

Вопрос в том, насколько много программистов сейчас занимаются чистым творчеством. По моим впечатлениям, большинство pet-проектов на GitHub – это какие-то мелкие веб-приложения, разного рода «хеллоуворлды», бессмысленные порты с одного языка на другой, какие-то врапперы и биндинги. Серьезные открытые проекты давно переросли стадию чьего-то хобби – творчеством там, конечно, и не пахнет. Даже на D чего-то нового и интересного не так уж много, и это меня искренне расстраивает. Был легендарный h3r3tic с его потрясающими программными растеризаторами и трассировщиками лучей времени компиляции. Была Higgs, JavaScript-машина с JIT-компиляцией. Был игровой движок Dash. Была пара-тройка мини-игр. И это, по большому счету, все. Сейчас репозиторий DUB забит какими-то унылыми библиотеками для веба, драйверами СУБД и реализациями малоизвестных протоколов. Скучно, ребята, скучно. На D можно писать удивительные штуки – почему никто этого не делает?

Тимур Гафаров
http://dlanggamedev.blogspot.ru

Game Maker Studio 2

Состоялся релиз второй версии популярного конструктора игр Game Maker Studio. GMS2 отличается, в первую очередь, полностью переделанным интерфейсом – фактически, это совершенно новая программа. В частности, появилась возможность работы с несколькими рабочими пространствами, свободная навигация внутри рабочих пространств с поддержкой масштабирования, а также связывание нескольких окон-свойств в цепочки для более удобной работы с компонентами игры. Улучшен редактор кода - появились вкладки для одновременной работы с несколькими скриптами в одном окне. Также появился поиск по ресурсам проекта, а во встроенном редакторе спрайтов появились слои, пользовательские кисти, предпросмотр анимации, возможность быстро переключаться между кадрами, не закрывая окно редактора, а также разделение экрана, которое позволяет рассматривать редактируемое изображение с разным масштабом. Значительно улучшен редактор комнат – появились новые инструменты, облегчающие групповое создание и удаление объектов.



Множество нововведений появилось в скриптовом языке GML (Game Maker Language), drag'n'drop-ин-терфейс также подвергся изменениям – теперь он больше похож на Blueprints в Unreal Engine.

GMS2 полностью поддерживает импорт проектов предыдущей версии и позволяет создавать игры под Windows, Windows/UWP, Linux, OSX, HTML5, Android, iOS, PlayStation 4 и Xbox One. Цена конструктора начинается с $99,99 за версию для десктопов и заканчиваются $399 за девкит для мобильных устройств. Есть также бесплатная пробная версия с ограниченными возможностями.

http://www.yoyogames.com/gamemaker

Разработка для PlayStation

Часть 5. 3D-графика

Вот мы и добрались до святая святых: трехмерной графики на PlayStation! Во время появления консоли (а это был, напомню, 1994 год) аппаратное ускорение 3D-рендеринга было самой настоящей инновацией – на ПК оно повсеместно появилось только годы спустя стараниями 3dfx и NVIDIA. Последняя, кстати, называет «первым GPU в мире» свой GeForce 256, вышедший в 1999 году, хотя это не более чем типичная маркетинговая ложь – у PSX был полноценный GPU, и для своего времени очень неплохой (достаточно быстрый для рендеринга игр уровня первых двух Quake). Кстати, во все анналы как первая полностью трехмерная игра вошла именно Quake (1996), но на PSX до нее были, например, полностью полигональные гонки Destruction Derby (1995), а также Battle Arena Toshinden (1995), один из первых 3D-файтингов, и Jumping Flash! (1995), которая считается первым полностью трехмерным платформером. Конечно, они не достигли славы Quake, но среди фанатов PSX считаются бесспорными хитами.

Но вернемся к спецификациям. GPU PSX поддерживает отрисовку треугольников и четырехугольников, закрашенных сплошным цветом или заполненных текстурой. Также поддерживается освещение (плоское и интерполированное по Гуро), прозрачность и туман. Для выполнения геометрических преобразований PSX использует отдельный сопроцессор, GTE (Geometry Transformation Engine). Согласно заявленным спецификациям, PSX способен выводить до 180000 затекстуренных полигонов в секунду.

У 3D-режима PSX есть и несколько недостатков. Самый главный из них – отсутствие перспективной коррекции текстур, из-за чего происходит сильное искажение картинки на больших плоских поверхностях. К тому же, все вычисления на GTE делаются в числах с фиксированной запятой, а это зачастую приводит к заметному подергиванию 3D-объектов на экране.

Впрочем, подобные вещи стали уже своеобразной визитной карточкой олдскульных платформ, и старые игры без них смотрелись бы куда менее лампово...

Интересная особенность PSX – отсутствие аппаратной Z-буферизации. GPU умеет рисовать только 2D-примитивы и ничего не знает о координате Z – задача по сортировке примитивов в зависимости от удаленности от камеры ложится на CPU. Сортировка делается очень остроумным методом, о котором я уже упоминал и теперь расскажу поподробнее.

Перебирать все треугольники и сравнивать их друг с другом – слишком дорого. Поэтому для ускорения сортировки используется нечто вроде хэш-таблицы, где хэш-функцией выступает глубина примитива. Эта таблица называется таблицей порядка – Ordering Table (OT).

В основе OT лежит односвязный список, элементами которого выступают пакеты – то есть, команды видеопроцессору нарисовать тот или иной примитив. Пакет имеет 32-битный заголовок, в 24 нижних битах которого хранится адрес следующего пакета (или 0xffffff, если пакет последний), а в 8 верхних – размер. Для хранения пакетов в оперативной памяти создается буфер фиксированного размера под некоторое максимальное количество дискретных значений глубины – обозначим его как N. Каждый пакет в буфере ссылается на предыдущий – таким образом, цепочка команд отрисовки хранится задом наперед.

Когда программа хочет нарисовать какой-то примитив, она вычисляет его глубину и дискретизирует ее в диапазоне 0..N – получается индекс (i), по которому извлекается соответствующий элемент в буфере и вставляется новый пакет в список после этого элемента.

Иными словами, пакет i ссылается на новый пакет, а тот – на пакет i+1. Затем GPU линейно обрабатывает все пакеты, начиная с последнего – последний пакет рисуется первым, поэтому они и хранятся в списке задом наперед. Получается сортировка сложности O(1).

Если какие-то примитивы будут иметь одинаковую глубину, возникнет коллизия – в данном случае новый пакет всегда будет помещаться после старого, и отрисован будет первым. Это и приводит к глюкам сортировки, которые порой можно видеть в некоторых играх на PSX, когда один объект перекрывает другой, хотя вроде и не должен. Чем больше значение N, тем точнее сортировка.

Ну, довольно теории, переходим к практике. Перед тем, как рендерить 3D-модель, нужно ее создать и сохранить в особом формате – TMD. Под-готовка моделей для PSX – достаточно трудоемкий процесс, но он отчасти облегчается путем использования специально написанных экспортеров для 3D-редакторов. Я лично являюсь пользователем Blender, поэтому меня интересовало, можно ли использовать его. Оказалось, что можно – существует экспортер для Blender 2.69 и выше, написанный все тем же Lameguy64. Ищите его по следующей ссылке:

http://www.psxdev.net/forum/viewtopic.php?f=60&t=707&start=0

(если не сумели скачать, пишите мне на почту, я вам его отправлю).

Экспортер сохраняет модель в RSD – промежуточный формат, с которым работают утилиты Psy-Q. Это простой текстовый формат с человекочитаемым описанием 3D-данных. Сама геометрия (вершины, нормали и треугольники) сохраняется в файл *.ply, текстурные координаты и цвета вершин – в файл *.mat. А файл *.rsd просто ссылается на *.ply и *.mat. В этом же файле содержатся ссылки на TIM-текстуры, используемые моделью – имена файлов TIM соответствуют именам текстур в Blender, только с расширением *.tim (в Blender, естественно, TIM не поддерживается, поэтому придется использовать другие графические форматы). Одна модель может содержать несколько текстур. При этом текстуры должны быть назначены моделям не через материалы, а в редакторе UV/Изображений. Будьте с этим внимательны, иначе ничего не заработает.

Итак, на выходе у вас должны быть 4 файла, лежащие в одной папке - поскольку я для простоты смоделировал простой кубик, то назовем их cube.rsd, cube.ply, cube.mat и cube.tim. Теперь нам понадобится утилита RSDLINK, входящая в состав PsyQ. Она «компилирует» модель из текстового в бинарное представление – TMD, которое затем и загружается программой. Вызываем ее следующим образом:

```
rsdlink -s 32.0 -o cube.tmd cube.rsd
```

Параметр -s задает коэффициент масштабирования модели. Рекомендуется начать с 32 и проверить программу, а затем увеличить это значение, если модель кажется слишком маленькой.

Поздравляю, вы создали свою первую 3D-модель для PSX! Скопируйте ее вместе с текстурой в корень будущего диска и назовите CUBE.TMD и CUBE.TIM.

Переходим к коду. Объявляем переменные и инициализируем 3D-графику:

```
Data timData, tmdData;
GsCOORDINATE2 camCoord2;
VECTOR camPos = {0};
SVECTOR camRot = {0};
VECTOR objPos = {0};
SVECTOR objRot = {0};
GsF_LIGHT light;
GsDOBJ2 obj;
int objectCount = 0;

// Здесь у нас GsInitGraph
// и другая стандартная инициализация

GsInit3D();
GsSetProjection(SCREEN_WIDTH/2);
```

Вторая функция задает расстояние от вьюпорта до плоскости проекции – фактически, меняет угол обзора камеры, поскольку размер плоскости проекции постоянен и зависит от разрешения экрана. Чем меньше это значение, тем больше угол обзора. В данном случае мы сделали его зависимым от ширины экрана. Значение по умолчанию – 1000.

Инициализируем координатную систему для камеры – по сути, под этим скрывается всего лишь создание единичной матрицы в мировом пространстве:

```
GsInitCoordinate2(WORLD, &camCoord2);
```

Задаем начальные позицию и поворот камеры:

```
camPos.vx = 0;
camPos.vy = 10;
camPos.vz = 150;
camRot.vy = 0;
camRot.vx = 0;
```

Задаем цвет окружения для освещения:

```
GsSetAmbient(ONE/4, ONE/4, ONE/4);
```

Задаем режим освещения (0 - освещение без тумана, 1 - с туманом):

```
GsSetLightMode(0);
```

Инициализируем источник света:

```
light.vx = 100;
light.vy = 100;
light.vz = 100;
light.r = 0xff;
light.g = 0xff;
light.b = 0xff;
GsSetFlatLight(0, &light);
```

Загружаем текстуру и модель:

```
readFile("\\CUBE.TIM;1", &timData);
loadTexture(&timData, &texture);
freeData(&timData);

readFile("\\CUBE.TMD;1", &tmdData);
objectCount += linkModel((u_long*)(tmdData.ptr), &obj);
obj.attribute = 0;
```

Функция linkModel присоединяет данные TMD к объекту GsDOBJ2. Она выглядит следующим образом:

```
int linkModel(u_long* tmd, GsDOBJ2* obj)
{
  u_long* dop;
  int i, numObj;
  dop = tmd;
  dop++;
  GsMapModelingData(dop);
  dop++;
  numObj = *dop;
  dop++;
  for(i = 0; i < numObj; i++)
  {
    GsLinkObject4((u_long)dop, &obj[i], i);
    obj[i].attribute = (1 << 6);
  }
  return (numObj);
}
```

Теперь мы можем рендерить. Главный цикл программы будет выглядеть так:

```
while(1)
{
  currentBuffer = GsGetActiveBuff();
  GsSetWorkBase((PACKET*)gpuPacketArea[currentBuffer]);
  GsClearOt(0, 0, &myOT[currentBuffer]);

  calcCamera();

  GsSetFlatLight(0, &light);

  objRot.vy += 10;
  putObject(objPos, objRot, &obj);

  FntFlush(-1);

  DrawSync(0);
  VSync(0);
  GsSwapDispBuff();

  GsSortClear(50, 0, 50, &myOT[currentBuffer]);

  GsDrawOt(&myOT[currentBuffer]);
}
```

Осталось определить две функции – calcCamera и putObject. Функция calcCamera вычисляет видовую матрицу для заданных позиции и поворота камеры и передает ее в GTE. Ее нужно вызвать перед тем, как рисовать объекты.

```
void calcCamera()
{
  VECTOR vec;
  GsVIEW2 view;

  view.view = camCoord2;
  view.super = WORLD;

  RotMatrix(&camRot, &view.view);
  ApplyMatrixLV(&view.view, &camPos, &vec);
  TransMatrix(&view.view, &vec);

  GsSetView2(&view);
}
```

Функция putObject рендерит заданный объект GsDOBJ2 с заданными позицией и поворотом:

```
void putObject(VECTOR pos, SVECTOR rot, GsDOBJ2* obj)
{
  MATRIX lmtx, omtx;
  GsCOORDINATE2 coord;

  coord = camCoord2;

  RotMatrix(&rot, &omtx);
  TransMatrix(&omtx, &pos);
  CompMatrixLV(&Camera.coord2.coord, &omtx,
      &coord.coord);
  coord.flg = 0;
  obj->coord2 = &coord;

  GsGetLws(obj->coord2, &lmtx, &omtx);
  GsSetLightMatrix(&lmtx);
  GsSetLsMatrix(&omtx);
```

```
    GsSortObject4(obj, &myOT[currentBuffer],
        14-OT_LENGTH, getScratchAddr(0));
}
```

Если все сделано правильно, вы должны увидеть в окне эмулятора вот такой вращающийся кубик. Ура!

Напоследок напомню, что существует такой сайт, как ***http://www.psxdev.net****, там очень много полезной актуальной информации по разработке под PSX, есть активное сообщество. Если возникли вопросы, не стесняйтесь, пишите мне на почту:* ***gecko0307@gmail.com****, постараюсь помочь по мере своих возможностей.*

Тимур Гафаров
gecko0307@gmail.com

Наши проекты

Cook

Программа автоматизации сборки проектов на языке D. В отличие от аналогичных инструментов (Make, CMake, Scons, DUB и др.), Cook не требует конфигурационного файла: всю информацию о проекте она получает самостоятельно, сканируя модули (файлы *.d). При этом программа отслеживает прямые и обратные зависимости между модулями: если модуль был изменен, необходимо скомпилировать заново не только его, но и все модули, которые от него зависят (это важно, если был изменен внешний интерфейс модуля: объявления классов, семантика шаблонов и т.д.).

Для этого Cook производит лексический анализ модулей - но не всех, а только тех, которые были изменены со времени последнего анализа. Данные анализа кэшируются в файл для повторного использования (кэш автоматически обновляется при пересборке).

Cook работает в Windows и Linux.

http://github.com/gecko0307/cook2

dlib

Коллекция библиотек «на все случаи жизни» для D, основа для разработки игр и игровых движков, систем рендеринга, графических редакторов, мультимедийных и научно-инженерных приложений. Напи-сана на D2 с использованием Phobos, не имеет никаких других внешних зависимостей. dlib включает независимые от Phobos и сборщика мусора потоки ввода-вывода, средства управления памятью, кон-тейнеры, объекты и функции линейной алгебры, средства обработки изображений, импорта/экспорта популярных графических форматов (JPEG, PNG, TGA, BMP, HDR), средства обработки звука, сокеты, асин-хронный движок с обработчиком событий, инструменты обработки строк и текстов, парсер XML и многое другое.

http://github.com/gecko0307/dlib

Фан-игры по мотивам Spyro

В «FPS» №31 '14 мы в рамках обзора серии Spyro the Dragon упоминали главные фанатские инициативы по созданию новых игр о приключениях дракона Спайро по мотивам классической трилогии Spyro на PSX. Самой известной из них была Spyro Eclipse: игра разрабатывалась на Unity и предвещала современную картинку вкупе со старым дизайном локаций и стилем геймплея. Однако проект не выжил. Настало время еще раз вернуться к этой теме – оказывается, на сегодняшний день существует уже не одна подобная инициатива...

Spyro's Den

Не совсем ясно, является ли «Spyro's Den» названием игры, или так называется только блог автора. Во всяком случае, эта игра является на сегодняшний день самой качественной переработкой классической трилогии Spyro – оригинальный дизайн серии очень удачно сочетается с современными шейдерными спецэффектами. Как и Spyro Eclipse, эта игра разрабатывается на Unity. Надеемся, что этот перспективный проект все-таки будет доведен до конца.

http://spyrosdens.blogspot.ru

http://gamejolt.com/games/spyro-the-dragon-fan-game-super-alpha/84112

Spyro Project

2D-платформер в модной сейчас нарочито пиксельной графике, разрабатывающийся на Game Maker: Studio. Уже доступна альфа-версия проекта.

http://gamejolt.com/games/spyro-project-fan-game/130936

Spyro the Dragon: Gold Time

Еще один 3D-римейк. Тоже создается на Unity, однако графически сильно уступает Spyro's Den. Тем не менее, проект также представляет большой интерес – особенно учитывая тот факт, что уже доступна альфа-версия.

http://gamejolt.com/games/spyro-the-dragon-gold-time/122148

Мобильный FPS

Теперь любимый журнал всегда с вами!

Читайте FPS на мобильных устройствах: скачайте приложение для Android или iOS!

Available on the App Store

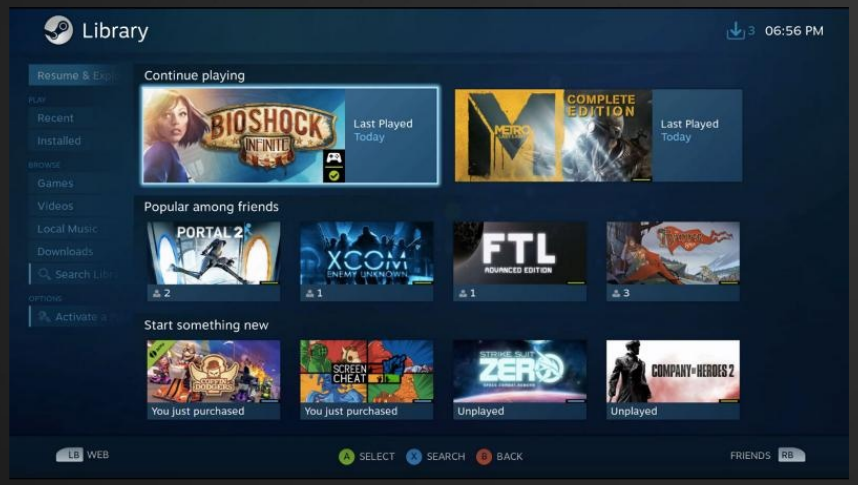
ANDROID APP ON Google play

Разработчик приложения: цифровое издательство St.Appler http://www.stappler.org/

Linux-гейминг

Игровые новости из мира СПО

Компания Valve объявила о прекращении работы Steam Greenlight, системы пользовательского голосования для допуска инди-игр в Steam. 13 июня начала работу платформа Steam Direct, которая позволит любому разработчику опубликовать свою игру в Steam, уплатив взнос в размере $100. Взнос будет возмещен после того, как продажи игры достигнут 1000 долларов.



Вышла новая версия игровой операционной системы SteamOS – 2.117. Новый выпуск основан на пакетной базе Debian 8.8 и ядре Linux 4.11. Наиболее примечательным изменением стала замена драйвера для графических карт AMD – вместо проприетарного драйвера AMDGPU-PRO пользователям теперь предлагается штатный открытый драйвер, предоставляемый Mesa. Сам пакет Mesa обновлен до версии 17.0.4.

После полутора лет разработки состоялся релиз Xonotic 0.8.2 – новой версии знаменитого свободного 3D-шутера. Проект является форком игры Nexuiz, созданным в результате конфликта ключевых разработчиков проекта и компании IllFonic. Из особенностей Xonotic можно отметить отличную графику, разнообразие карт, обилие режимов игры.

В новой версии реализованы быстрое меню и другие полезные элементы интерфейса, добавлены две новые deathmatch-карты Boil и Erbium, новые модели оружия и спецэффекты.

http://www.xonotic.org

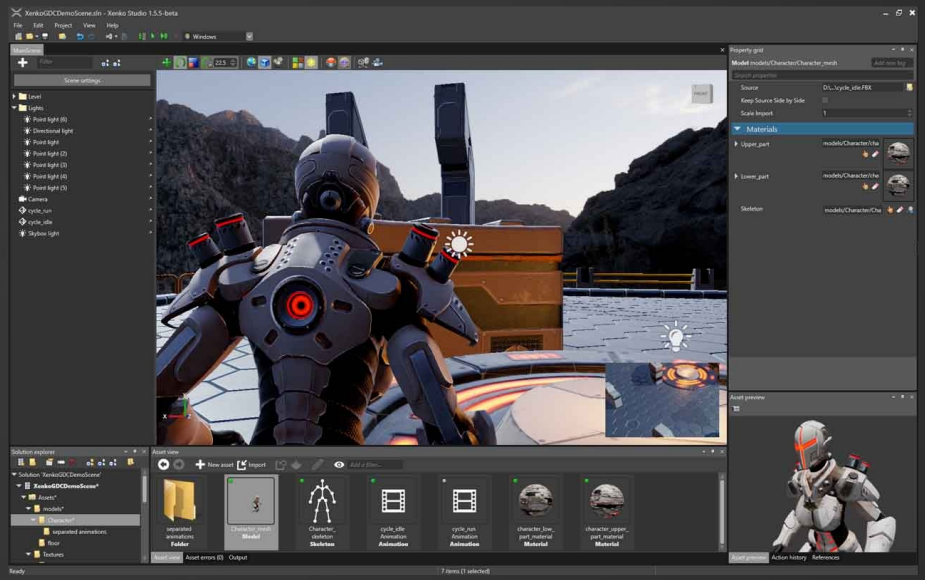
Увидела свет INSTEAD 3.0.0 – новая версия свободного движка для создания текстово-графических квестов. Из особенностей релиза – обновленный API, поддержка запуска игр в браузере, расширенные графические и звуковые возможности.

https://instead.syscall.ru/ru

После двух лет разработки вышла вторая версия игрового движка Xenco от Silicon Studio. Из нововведений отметим переработанный граф сцены, поддержку шлемов виртуальной реальности (в данный момент поддерживаются только Oculus и HTV Vive, разработчики обещают больше устройств в будущем), непрямое освещение (GI) при помощи световых зондов, а также новый графический композитор.

Начиная с версии 2.0 изменились условия коммерческой поставки движка. Теперь редактор Xenco Game Studio распространяется бесплатно только для персонального использования и маленьких студий с годовым доходом менее $200000 (Xenko Personal). Для крупных студий предлагается платная подписка на редакции Xenko Pro ($75 за одну лицензию в месяц) и Xenko Pro Plus ($150) с доступом к исходному коду редактора. Xenko Pro можно опробовать бесплатно до 31 июля.

Данные условия касаются только редактора – исходники рантайма все так же открыты для всех, но, к сожалению, теперь они распространяются под несвободной EULA, не разрешающей публиковать модифицированный код.



Ранее движок был доступен под GPLv3 – классический пример, когда в погоне за прибылью компания ужесточает лицензионные условия, забывая или игнорируя принципы свободного ПО. Похоже, что Silicon Studio берет пример с Epic Games и Crytek – Unreal Engine и CryEngine тоже открыты, но несвободны.

К счастью, исходники старой ветки Xenco (1.x) все еще доступны в репозитории на GitHub – ждем появления форков.

Напомним, что Xenco – это Unity-подобная среда для разработки игр, основанная на платформе .Net/Mono и работающая с языком C# (интегрируется с MS Visual Studio, но это необязательно). Позволяет создавать как 3D-, так и 2D-игры. Рендеринг осуществляется через Direct3D 12, OpenGL и Vulkan. Движок поддерживает Windows, Windows Phone, Android, iOS и Xbox One.

http://xenko.com

Компания AMD представила библиотеку Anvil – обертку над графическим API Vulkan для C++, предоставляющую разработчику полезные дополнительные возможности, как-то: менеджер памяти и автоматические указатели, функции для преобразования чисел с плавающей запятой разной точности, интеграция с компилятором шейдеров glslang и многое другое. Библиотека свободна, код доступен на GitHub по лицензии MIT.

https://github.com/GPUOpen-LibrariesAndSDKs/Anvil

Компания Google представила не менее интересный проект – библиотеку для сжатия 3D-моделей Draco. Она позволяет существенно сократить объем хранимых и передаваемых по сети 3D-данных, обеспечив при этом высокую скорость распаковки и упаковки. Например, использование Draco дает возможность существенно уменьшить размер игр и сократить время на загрузку сцен.

Код написан на языке C++ и распространяется под лицензией Apache 2.0. Для веб-разработчиков подготовлен декодер на JavaScript, позволяющий обрабатывать сжатый 3D-контент на стороне клиента.

В будущем планируется расширить Draco возможностью сжатия с потерей детализации, что может применяться в условиях низкой пропускной способности сети.

https://github.com/google/draco

Вышла новая версия физического движка Bullet – 2.86. В этой версии улучшен биндинг к Python, добавлена возможность остановить солвер ограничений при достижении определенного минимального порога ошибки, а также обеспечена базовая поддержка файлов формата MuJoCo MJCF.

http://bulletphysics.org

Где скачать «FPS»?

В Интернете часто можно встретить вопросы о том, где скачать старые номера нашего журнала. Отвечаем. Архив всех номеров «FPS» (с 2008 по 2017 гг.) можно найти сразу на нескольких сервисах:

На файловом хостинге DropBox:
https://www.dropbox.com/sh/b7lgxxh6nxbxre9/AADVDwrGSORvtEzm5V6O5d30a

В Документах Google (для скачивания файлов нужен аккаунт Google):
https://docs.google.com/folderview?id=0B1BlzRb1uMv-bnpHNDhwZTI4eHc

На сервисе Issuu.com:
http://issuu.com/tgafaroff/docs

Для тех, кто предпочитает скачивать с торрентов – некоторые номера журнала есть на РуТрекере:
http://rutracker.org/forum/viewtopic.php?t=4403193

Это все!

Надеемся, номер вышел интересным. Если вам нравится наш журнал, и вы хотели бы его поддержать – участвуйте в его создании! Отправляйте статьи, обзоры, интервью на любые темы, касающиеся компьютерных игр, графики, звука, программирования и т.д. на **gecko0307@gmail.com**.

http://fps-magazine.cf