



# Вестник *Kolibri OS*

#01(2)ноябрь 2007



**Мы Вам говорили, что выйдет второй выпуск Вестника Колибри, а Вы не верили!**

- ✓ **Новости проекта**
- ✓ **КолибриОС в лицах. Интервью с diamond**
- ✓ **Другие ОС: Polit OS**
- ✓ **Стиль и графика КолибриОС**
- ✓ **Взламываемая Editbox**
- ✓ **Самовар - все для чайников**





# Політ ОС

Сегодня мы представляем Polit OS (Політ ОС) Украинской мини ОС, написанную на Borland Pascal 7, Иваном Козаком (г.Киев), первая версия вышла 09.01.2001 размером 176 КБ, последняя версия вышла 30.08.2005 размером 1.4МБ. В Polit входит файл-менеджер, текстовый редактор, CD-плеер и проигрыватель wav-файлов, калькулятор, игры, есть возможность просмотра картинок - bmp, gif, jpg, программа настройки и т.д. Так же можно скачать для Polit скины, шрифты, курсоры. На данный момент проект заморожен, более подробно о PolitOS на сайте <http://bespin.org/~polit/> Сайт и вся ОС на Украинском языке. Минусы ОС - проблемы с металломом (не все железо распознает). Ну и прочие... Нам удалось связаться с Иваном, и взять у него интервью.

**Q:** Скажите, почему прекратил существовать проект ПОЛИТ ОС?

**A:** Приветствую, коллеги :) Да, несколько лет назад я сам работал над графической оболочкой DOS, традиционно называемой Полит-ОС :) С самого начала я поставил перед собой цель: создать систему для неопытного юзера (ну и притом открытую и бесплатную). Ориентация на неопытного юзера фактически означала две вещи:

1. Минимально достаточная для обычных дел функциональность.

2. Достаточно простой и привлекательный интерфейс.

Начинал ее писать я в далеком 1999-ом году; идея интерфейса была такова - традиционный оконный интерфейс, с нуля "сделанный правильно" :) И за несколько лет работы мне вполне удалось достичь обеих целей. Но вдруг оказалось :, что за эти годы многое успело измениться. И базовая функциональность, которую юзеры уже воспринимают как что-то само собой разумеющееся, стала куда шире (интернет, музыка, фильмы, файлы документов...), и интерфейсы ОС шагнули вперед. И Полит при всех его находках и элегантности уже выглядел устаревшим и ограниченным. Тогда-то я и понял с грустью, что паровоза мне не догнать :, что нужна функциональность и новый, современный интерфейс - это кусок работы на много лет... За которые все опять успеет измениться.

**Q:** Вы были один или была команда разработчиков?

**A:** Мне помогли многие люди - и советами, и тестированием, и написанием приложений, и даже с ядром, но все же большую часть работы делал я сам. Огромным плюсом такого подхода стало то, что я от начала до конца держал в голове четкую цель проекта; в результате весь Полит выдержан в едином духе, от алгоритмов и API до особенностей вида Элементов управления во всех скинах, время не тратилось ни на обсуждения, ни на реализацию не необходимых вещей. И настольно же огромным минусом, конечно же - скорость разработки, ограниченная моим свободным временем. Обычно работа над подобными проектами идет совсем другим образом - параллельно работает команда из нескольких человек, они все одновременно и увлеченно трудятся; но неизбежно (все мы

люди :) у каждого интересы и видение проекта немного отличаются. В итоге - два варианта. Один, самый распространенный - каждый делает что хочет, и мы получаем продукт без четкой цели, который пригоден для многого, но ни в чем не хорош по-настоящему му. Скажем, решили пятеро мастеров собрать супер-автомобиль. Один собрал компактный корпус, удобный для поездок по городу. Второй поставил туда мотор от гоночного болида. Третий соорудил внутри музыкальный центр. Четвертый пристроил на крыше маленькую башню с пушкой - на всякий случай :) И наконец пятый пристроил рядом с пушкой винт, как у вертолета. И вот машина готова. И когда довольные создатели сели в нее и поехали - оказалось, что в городе ездить очень мешает винт, и разогнаться как следует не дает, что рев мотора не дает слушать музыку, что винт мешает стрелять, тяжесть всей конструкции не дает взлететь выше чем на полметра, а чтобы все поместилось в корпусе пришлось сильно уменьшить бак с горючим. Типичная картина для open-source проектов, к сожалению :) Другой вариант - лидер проекта четко держит свою линию (и тогда неизбежны конфликты внутри команды - кому охота делать то, с чем ты не согласен, да еще за бесплатно? :) Как видим, идеального пути нет; а лучший же из них - пожалуй, третий, при условии что лидер проекта действительно хорош и умеет убеждать и избегать конфликтов. Чего я и желаю Колибри :)

**Q:** Ваше отношение к Колибри?

**A:** Во-первых я очень рад, что подобные проекты существуют до сих пор, и Колибри - один из самых перспективных среди них. Ее сильная сторона - это ее компактность и скорость, достижимые только благодаря ассемблеру. Для меня как бывшего программиста - это очень большие преимущества. И надо отметить - функциональность тоже на уровне лучших аналогичных проектов. А как дизайнер интерфейсов должен сказать вот что: с точки зрения юзера, выглядит ОС вполне неплохо, при этом местами она не сложнее в использовании чем Win98, местами же требует от юзера больших знаний. Такое странное сочетание тоже, к сожалению, очень распространено среди аналогичных систем. В результате, разработчики тратят много усилий, чтобы сделать свои программы красивыми, удобными и простыми, но из-за нескольких ключевых "узких мест" системой все равно может пользоваться только программист, что сильно ограничивает потенциальную базу пользователей ОС, и соответственно ее популярность. Сам знаю, что написать надежный и безопасный автодетект каково-нибудь параметра бывает в десятки раз труднее, чем просто спросить его у юзера. Но... В одном обзоре MacOS X я читал о том, как 13-летняя дочка автора сама настроила и подняла FTP-сервер на домашнем компьютере. Наводит на размышления, правда? :)

**Q:** При достаточном финансировании, возможно ли

создание полноценной мини ОС?

**A:** Для начала нам нужно определиться, что такое "мини-ОС"; как я уже говорил, базовая функциональность, которой ждут от компьютера даже начинающие юзеры (не говоря уже о профессионалах) - она в наши дни весьма высока; и с ней ОС вряд ли останется "мини", а без нее она не будет интересна широкой аудитории обычных юзеров (другими словами - не сможет конкурировать с Windows, MacOS X, Linux). А вообще, финансирование дает возможность программисту работать над проектом постоянно, а не только в свое свободное время; это, конечно же, сильно повышает его продуктивность в работе над проектом ОС, и соответственно темпы разработки этой самой ОС, что очень важно для ее успеха. Другой вопрос - будет ли такой разработчик получать удовольствие от процесса, ради которого мы все и занимаемся этим, или это просто станет для него работой? Это уже зависит от человека.

**Q:** Как вы считаете, на каком языке лучше программировать ОС?

**A:** На своем любимом языке :) При желании под это можно даже теорию развести, например, что ассемблер - это безграничная свобода, что он дает максимальную эффективность и компактность кода, какая только возможна, или там, Паскаль сильно упрощает и ускоряет работу программиста, помогает писать правильно и красиво, ну и так далее :) Ну а если серьезно - для современной ОС самой оптимальной я считаю связку ассемблера для ядра и самых критичных по скорости мест и что-то высокоуровневое для приложений и всего остального. В наши дни требования к функциональности ОС сильно выросли, а в ближайшие годы пойдут в гору и требования к интерфейсу (посмотрите на только что вышедшую MacOS X 10.5 Leopard и, частично, Windows Vista). При этом компьютеры становятся все быстрее, а объемы дисков и памяти - все больше. В этих условиях становится все важнее экономить ресурсы программиста, а не машины.

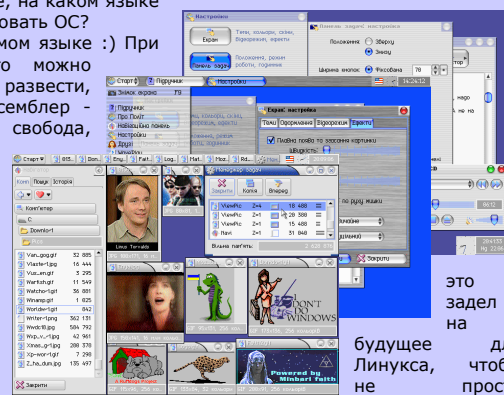
**Q:** И последний вопрос: Ваше мнение о больших ОС - Виста, Линукс, Мак? И когда мир изобретет монополию Виндос?

**A:** Сразу скажу: что будет с монополией Windows - я не знаю :) Скорее всего, она сохранится еще некоторое время - уж слишком многое завязано на этой ОС в нашем мире. И вместе с тем, как это уже бывало неоднократно, господство Windows буквально в любой момент может закончиться - или в силу изящного хода конкурентов, или из-за некоторых непредсказуемых обстоятельств. Итак, Windows... Когда-то я страстно ненавидел ее, как и многие в те времена :) Но когда я наконец смог заставить работать контроль "поле ввода" для Полита (со всей функциональностью, включая выделение текста, прокрутку, работу с клавиатурой, контекстное меню, горячие клавиши у куча других мелочей), я аж зауважал Windows :)

Я понял, что все-таки ее разработчики проделали огромную

работу, и проделали ее весьма хорошо. Как дизайнер интерфейсов скажу - она достаточно хороша для своей аудитории, и хотя действительно хороших решений в ней немного, но и больших ляпов тоже нет (я говорю про все версии). Меня это несколько удивляет - при размерах и ресурсах Microsoft они могли сделать куда лучший интерфейс. Я бы предположил, что "достаточно хороший" интерфейс - это их политика, у них нет большой нужды еще улучшить его. Ну, или у них просто не очень хорошие дизайнеры :) Линукс - он всегда имел мои симпатии, как символ движения open-source и его самое большое достижение. В последние годы Линукс по качеству интерфейса вполне сравним с Windows, хотя, к сожалению, до сих пор имеет некоторые узкие места. С другой стороны, на платформе Линукс сейчас разрабатывается несколько экспериментальных интерфейсов.

Некоторые из них действительно интересны (например Mezzo UI), и



это задел на

будущее для Линукса, чтобы не просто сравняться с

конкурентами по интерфейсу, но и кое в чем обогнать их. Наконец, как дизайнер интерфейсов, я просто люблю MacOS X. Ребята из Apple поставили перед собой ту же задачу - ОС для неопытного юзера. И им хватило ресурсов сделать это, а кроме того сделать ее красивой, эффективной и эффективной. Для графических интерфейсов сейчас зревает переломный этап - десктопно-окно-менюшно-контрольная модель исчерпывает себя, и приходит время для следующего шага. И именно дизайнеры Apple сейчас нащупывают этот путь, как я считаю, совершенно правильно. Будущее - за визуально - динамическими интерфейсами, наглядно показывающими юзеру все происходящее, берущими на себя много рутинных задач, к которым мы с вами давно привыкли, вообще делающие компьютеры куда ближе к жизни, чем они есть сейчас. Скажем, у нас есть задача - скачивание СМСок из телефона в компьютер. Процесс можно показать просто прогресс-баром с подписью, скажем: "Скачивание СМС: 15 из 37 передано". А можно изобразить телефон, СМСки в нем, и показать как они перелетают оттуда прямо в базу и тут же появляются там. При этом нам с одного взгляда на экран видно, что процесс идет, сколько скачано и сколько остается, перемещаются СМСки из телефона или копируются (остаются ли они в нем), куда попадают скачанные СМСки, и прямо на лету читать то что сейчас передается. Я считаю - через несколько лет мы все будем общаться с компьютерами подобным образом.

# Стиль и графика КолибриОС

Мы работаем с Колибри, как с Windows или Linux, входя в систему, открывая и закрывая десятки программ. Часто мы даже не задумываемся об их внешнем виде, а просто жмём и ожидаем результат. И всё же каждый из нас абсолютно чётко может сказать: «В этой системе мне приятно работать, а в этой



нет». А почему именно так часто бывает объяснить сложно.

Первое моё знакомство с Колибри произошло довольно давно. Скорее всего, это был 2004 год. Тогда это был просто перевод операционной системы

Менюэт на русский язык с некоторым количеством собственных программ и доработок ядра. Графика естественно была унаследована от неё же. Во второй раз я столкнулся с Колибри (0.6.3.0) в 2006 году. Как оказалось, многое с того времени изменилось: официальный сайт переехал, Колибри вырос в абсолютно самостоятельный проект, система стала значительно стабильней, появилось много нового софта.

В плане графики она тоже порадовала – симпатичный, не напрягающий глаза и не отвлекающий внимание валлпапер (обои рабочего стола). В будущем я узнал, что его автором является goglus. Ещё одним приятным моментом было то, что теперь практически все программы поддерживали оконные скины. Стиль по умолчанию тоже был довольно интересным. Но. Но иконки! Прошло несколько лет, а они совершенно не изменились, только добавилось пару новых. Имея желание помочь проекту и хобби иконки, я быстро сообразил, что надо делать.

Как-то я бывал на сайте <http://tango.freedesktop.org/> и прочитал там статью о создании иконок. Когда я начал делать собственные для Колибри, я

старался использовать цвета

проекта Tango, но, как оказалось, они совершенно не подошли. А подбор подходящих цветов играет немаловажную роль. Я стал использовать более яркие цвета, но всё равно, если присмотреться, можно увидеть схожесть. А вот что мне на самом деле помогло из той статьи – это метод рисования. Хотя большинство из прочитанного было и так интуитивно понятно и логично. В ходе рисования, я придерживался следующих правил:

1. в иконке должно быть не слишком много разных цветов (это не попугай) – несколько основных и их оттенки;
2. иконка должна точно отображать назначение программы; и одно из самых важных – это игра света. Заметьте светлую полосу сверху и слева и тёмную снизу и справа возле краёв изображения;
3. стараться не использовать острых углов;

Проблема, с которой я столкнулся в процессе разработки – это то, что количество цветов всех иконок (а их около 30!) должно быть не более 256. Поэтому я вначале экономил и старался использовать одинаковые цвета

в разных иконках. Если кому-то интересно, в каких программах я работал, то это были IconXP 2.12 и, может быть, временами AWiconsPro 9.4. Да, это не Photoshop и не Illustrator (который я сейчас пытаюсь освоить), но в данном случае мой выбор, как мне кажется, оказался оптимальным. Вернёмся же к Колибри. Мне потребовалось около двух недель чтоб сделать первые наработки и выложить их на форуме 21 февраля 2007 года. Иконки мной постепенно дорабатывались и совершенствовались. Колибри полностью отошла от своего предшественника – операционной системы Менюэт и сейчас является абсолютно самостоятельной системой. А по графике это, наверное, заметно больше всего. Менюэт использует тёмные цвета, Колибри же наоборот – яркие и жизнерадостные. Иконки я создавал, придерживаясь данного принципа, т.к. считаю этот путь верным. Колибри отличная ОС, вот только пока не всё она может. Именно в Ваших силах сделать её лучше. Такой, какой хотели бы Вы её видеть вместе с десятками тысяч людей по всему миру. Всё в Ваших руках. Удачи.

Липатов Кирилл, дизайнер.

Ваша реклама в Нашем журнале  
рекламный отдел : [re-zine@ya.ru](mailto:re-zine@ya.ru)

## Самовар - все для чайников

Многих новичков Колибри удручают командные строки Ассемблера, и у них сразу пропадает интерес к КОСе, но оказывается, ничего тут сложного нет, и на наглядном примере можно сделать что-то полезное для мини-ОСи. Давайте попробуем создать свой скин для Колибри! И так начнем, во первых скачайте исходники с официального сайта Колибри. Распакуйте их на жестком диске, напоминая: он должен работать под Fat32. Запускаем KFM, открываем исходники, находим папку Skin, открываем её, и выбираем на свой вкус скин. Например: Leetheme, открываем его и копируем на ram-диск /rd/1 файлы: default.asm, base.bmp, base\_1.bmp, left.bmp, left\_1.bmp, oper.bmp, oper\_1.bmp, oper\_1.bmp, myblue.dtp, me\_skin.inc. Запускаем приложение FASM (иконка на рабочем столе), нажимаем курсором на Infile, удаляем файл

- example.asm и на его месте набираем - default.asm; В строке Outfile, набираем файл - default.skn, в строке path – ничего не меняем, нажимаем Compile – так, мы создаем файл запуска, настройки окон – default.skn. Находим его на ram-диске /rd/1, запускаем, появляется окно настройки окон. Нажимаем применить, смотрим, что получилось. Теперь попробуем изменить параметры скина – сначала из исходников других скинов, копируем картинки bmp: base.bmp, base\_1.bmp, left.bmp, left\_1.bmp, oper.bmp, oper\_1.bmp на ram-диск /rd/1. Для примера можете взять blackskin. В приложении FASM нажимаем снова Compile – получаем новый скин! (если вы до этого закрыли FASM, тогда заполните его строки как было сказано выше). Запускаем default.skn на ram-диск /rd/1 смотрим что получилось.

В папке исходников - скинов есть заготовки шаблонов окон: папка dtp, открываем её и выбираем нужный файл, например black.dtp, копируем его на ram-диск /rd/1. Дальше открываем на ram-диск /rd/1 файл - default.asm, он откроется в приложении Tinurad, находим командную строку - dtp и меняем в ней файл - myblue.dtp на файл - default.asm, он откроется в Tinurad, сохраняем изменения в файле, Tinuradом. Компилируем FASMом, смотрим, что получилось. Для изменения цвета окантовки окон, делаем изменения в файле - default.asm. В строках colors active и colors inactive, например в подстроке: bframe набираем 0x363636 (код темно-серого

цвета), сохраняем изменения и опять запускаем компилятор FASM, смотрим, что получилось. После того как вы научились делать изменения в настройках скина, можно создавать свои кнопки (рисунки-bmp), где вы их сделаете в Photoshop или в другой программе - выбирайте сами, главное, не забудьте:

**Анекдот:** Встречаются два чайника, один чайник другому и говорит:

-Чайник! Давай напишем учебник по программированию на FASM для чайников!

-Ты что, чайник! Я в этом полный чайник, я в этом не разбираюсь!

-Ну и что, чайник! Чайники все равно ничего не понимают!

высота картинки от 7 до 10 пикселей, ширина на ваш вкус, сохраните рисунок в формате bmp 24-бита. Скопируйте их на ram-диск и наслаждайтесь. Порадуйте всех, новым нестандартным дизайном! Удачи!

сеса



```

data_of_code dd 0
mouse_flag dd 0x0
hed db 'EDITBOX optimization and retype <Lrz> date 20.07.2007',0
rb 256
ed_buffer:
;.1: rb 514;256
.2: rb 101
.3: rb 11
.4: rb 3
;два запасных байта необходимы для того что бы не переписать следующие байты, в конце буфера 0
buffer_end:
align 16
procinfo process_information
meos_app_end
udata

```

Рассмотрим простой каркас приложения. Итак, начнем по порядку:

Обратим внимание на то что во втором боксе располагается строка заголовка программы, курсор установлен в конец. Посмотрим внимательно на данные и разберем их, как что работает.

Нас интересует вот эта строка, обработав которую и можно увидеть данный результат:

```
edit2 edit_box 250,5,30,0xffffffff,0x6a9480,0,0xAABBCC,0,308,hed,ed_focus,53,53
```

Разберем и проанализируем каждый параметр:

edit2 — уникальный идентификатор данного бокса, обращаясь к нему можно получить 0 смещение к данным. После обработки компилятором вся данная запись превратится вот в такую структуру:

**edit2**

- .width dd 250** - ширина бокса
- .left dd 5** - левый отступ по x
- .top dd 30** - отступ сверху по y от начала приложения
- .color dd 0xffffffff** - цвет внутри рамки
- .shift\_color dd 0x6a9480** - цвет выделения при удерживании shift или мышкой
- .focus\_border\_color dd 0** - цвет рамки editbox
- .blur\_border\_color dd 0xAABBCC** - при неактивном боксе цвет рамки
- .text\_color dd 0** - цвет текста
- .max dd 308** - максимальная длина которую можно ввести
- .text dd hed** - смещение к буферу, куда будут помещены символы, вводимые с клавиатуры
- .flags dw ed\_focus** - флаги, в данном случае мы говорим, что editbox данный будет иметь фокус, флаг ed\_figure\_only говорит о том, что вводить можно только цифры.
- .size dd 53** - текущее кол-во символов, которые уже введены (как видно из строки head длина ее составляет 53 символа)
- .pos dd 53** - позиция курсора т.е. если она будет =53 мы будем в конце строчки

;Далее идут системные данные editbox, которые не предназначены для изменения т.е. сам editbox хранит в них свои данные.

- .offset dd 0** - смещение, т.е. кол-во шагов из расчета: кол-во символов в строке, поделенное на количество символов, уместяющихся в поле editbox
- .cl\_curs\_x dd 0** - хранит старое значение по x курсора (используется для очистки)
- .cl\_curs\_y dd 0** - хранит старое значение по y курсора (используется для очистки)
- .shift dd 0** - хранит положение курсора при нажатии shift
- .shift\_old dd 0** - хранит старое положение курсора

Итак надеюсь тут все понятно.

В самом начале программы происходит инициализация бокса

```
use_edit_box procinfo,22,5
```

где procinfo - указатель на эту структуру:

```

struct process_information
cpu_usage dd ? ; +0
window_stack_position dw ? ; +4
window_stack_value dw ? ; +6
dw ? ; +8
process_name rb 12 ; +10
memory_start dd ? ; +22
used_memory dd ? ; +26
PID dd ? ; +30

```

```

box          BOX      ; +34
slot_state   dw ?     ; +50
             dw ?     ; +52
client_box   BOX      ; +54
wnd_state    db ?     ; +70
rb (1024-71)

```

```

struct BOX
left         dd ?
top          dd ?
width        dd ?
height       dd ?
ends

```

Буфер, на который указывает ebx, содержит следующую информацию:

- +0: dword: использование процессора (сколько тактов в секунду уходит на исполнение именно этого потока)
- +4: word: позиция окна потока в оконном стеке
- +6: word: (не имеет отношения к запрошенному потоку) номер слота потока, окно которого находится в оконном стеке в позиции esx
- +8: word: зарезервировано
- +10 = +0xA: 11 байт: имя процесса (имя соответствующего исполняемого файла в формате 8+3)
- +21 = +0x15: byte: зарезервировано, этот байт не изменяется
- +22 = +0x16: dword: адрес процесса в памяти
- +26 = +0x1A: dword: размер используемой памяти - 1
- +30 = +0x1E: dword: идентификатор (PID/TID)
- +34 = +0x22: dword: координата окна потока по оси x
- +38 = +0x26: dword: координата окна потока по оси y
- +42 = +0x2A: dword: размер окна потока по оси x
- +46 = +0x2E: dword: размер окна потока по оси y
- +50 = +0x32: word: состояние слота потока:
- 0 = поток выполняется
- 1 = поток приостановлен
- 2 = поток приостановлен в момент ожидания события
- 3 = поток завершается в результате вызова функции -1 или насильственно как следствие вызова под функции 2 функции 18 или завершения работы системы
- 4 = поток завершается в результате исключения
- 5 = поток ожидает события
- 9 = запрошенный слот свободен, вся остальная информация о слоте не имеет смысла
- +52 = +0x34: word: зарезервировано, это слово не изменяется
- +54 = +0x36: dword: координата начала клиентской области по оси x
- +58 = +0x3A: dword: координата начала клиентской области по оси y
- +62 = +0x3E: dword: ширина клиентской области
- +66 = +0x42: dword: высота клиентской области
- +70 = +0x46: byte: состояние окна - битовое поле
- ✗ бит 0 (маска 1): окно максимизировано
- ✗ бит 1 (маска 2): окно минимизировано в панель задач
- ✗ бит 2 (маска 4): окно свёрнуто в заголовок

A 22 — это размер поля заголовки программы по y  
 5 — размер толщины оформления по x от окна до начала рабочей области.  
 Эти два параметра не передаются, если при инициализации окна в функции

```
mcall 0,(50*65536+390),(30*65536+200),0xb3AABBCC,0x805080DD,hed
```

- eax = 0 - номер функции
- ebx = [координата по оси x]\*65536 + [размер по оси x]
- ecx = [координата по оси y]\*65536 + [размер по оси y]
- edx = 0xXYRRGGBB, где:
- Y = стиль окна:
- Y=0 - тип I - окно фиксированных размеров
- Y=1 - только определить область окна, ничего не рисовать
- Y=2 - тип II - окно изменяемых размеров
- Y=3 - окно со скином
- Y=4 - окно со скином фиксированных размеров
- остальные возможные значения (от 5 до 15) зарезервированы, вызов функции с такими Y игнорируется
- RR, GG, BB = соответственно красная, зеленая, синяя составляющие цвета рабочей области окна (игнорируется для стиля Y=2)
- X = DCBA (биты)
- A = 1 - у окна есть заголовок; для стилей Y=3,4 адрес строки заголовка задаётся в edi, для прочих стилей используется под функция 1 функции 71
- B = 1 - координаты всех графических примитивов задаются относительно клиентской области окна
- C = 1 - не закрашивать рабочую область при отрисовке окна
- D = 0 - нормальная заливка рабочей области, 1 - градиентная

---

*Анекдот:) Ты слышал, новые дистрибутивы Колибри выходят быстрее, чем сама ОС загружается с дискеты!*

---

Следующие параметры предназначены для окон типа I и II и игнорируются для стилей Y=1,3:

- esi = 0xXYRRGGBB - цвет заголовка
- RR, GG, BB определяют сам цвет
- Y=0 - обычное окно, Y=1 - не перемещаемое окно
- X определяет градиент заголовка: X=0 - нет градиента, X=8 - обычный градиент, для окон типа II X=4 - негативный градиент
- прочие значения X и Y зарезервированы
- edi = 0x00RRGGBB - цвет рамки

в параметре 0xb3AABBCC указан

- b= 1011 = DCBA (биты)
- A = 1 - у окна есть заголовок; для стилей Y=3,4 адрес строки заголовка задаётся в edi, для прочих стилей используется под функция 1 функции 71
- B = 1 - координаты всех графических примитивов задаются относительно клиентской области окна
- C = 1 - не закрашивать рабочую область при отрисовке окна
- D = 0 - нормальная заливка рабочей области, 1 - градиентная

Нас интересует вот это: B = 1 - координаты всех графических примитивов задаются относительно клиентской области окна. Мы рисуем относительно клиентской области, а координаты мышки получаем относительно всего окна, и для учета этого мы вводим поправочные данные в наш код, чтобы он откомпилировался и учитывал эти значения. Если мы рисуем относительно всего окна, тогда эти коэффициенты не нужно вводить. Т.е. если бит B = 0 — то поправочные коэффициенты не нужно указывать при инициализации бокса.

Далее:

```
mouse_edit_boxes editboxes,editboxes_end
```

вставляет в это место код для обработки мышки. Где

```
editboxes      — указатель на структуру данных где располагаются боксы  
editboxes_end  — конец структуры.
```

Если нужно инициализировать один бокс, то достаточно указать только один параметр `mouse_edit_box editbox` т.е. только начала кода, т.к. общая структура известна.

```
key_edit_boxes editboxes,editboxes_end - работает аналогично  
key_edit_box editbox — для инициализации одного бокса.
```

Обязательным параметром включения в исходный код является `mouse_flag dd 0`: в нем хранится идентификатор активного бокса, которому требуется предоставление всех событий от мышки (это связано с выделением), пока он не сбросит его в 0.

Lrz

Мы вам говорили, что выйдет второй выпуск Вестника Колибри, а вы не верили! Может, третий номер напишут другие люди, время покажет. Ваше мнение и ваши отзывы присылайте на [re-zine@ya.ru](mailto:re-zine@ya.ru) и на форум КолибриОС.

### Состав группы, работающей над журналом:

**сеса**  
<[re-zine@ya.ru](mailto:re-zine@ya.ru)>

**Евгений Гречников aka diamond**  
<[diamondz@land.ru](mailto:diamondz@land.ru)>  
icq:402363612

**Константин Николенко aka Veliant**  
<[knikolenko@yandex.ru](mailto:knikolenko@yandex.ru)>  
icq:313257308

**Кирил Липатов aka Leency**  
icq:419114984

**Алексей Теплов aka Lrz**  
<[tay-ugur@chel.surnet.ru](mailto:tay-ugur@chel.surnet.ru)>  
icq:267994076

Вёрстка:

**Михайлов Илья aka Ghost**  
<[ghost.nsk@gmail.com](mailto:ghost.nsk@gmail.com)>  
icq:111114333

**Виктор Григорьев aka Vectoroc**  
<[vectoroc@gmail.com](mailto:vectoroc@gmail.com)>  
icq:8449540

*Если ты говоришь, что вернешься, то я знаю, так оно и будет...*

**Роберт Шекли.**