Создаем скин для библиотеки AlphaControls своими руками.

Вместо вступления.

Некоторое время назад на сайте библиотеки AlphaControls (http://www.alphaskins.com) появилась русскоязычная инструкция по созданию скинов. В данной статье хотелось бы с использованием данной инструкции пошагово пройти все этапы создания скина от самого начала и до финального его релиза.

Итак, оговорюсь сразу, что скин будем создавать простой, без особых наворотов и усложнений, используя базовые возможности библиотеки и редактора скинов. Если мы разберемся с основами, то в дальнейшем усложнить скин и задействовать весь функционал библиотеки будет проще. Да и мой опыт работы по созданию скинов не является поводом для чьей-либо зависти. Так что будем разбираться вместе, а в самых сложных местах нам будет помогать автор библиотеки AlphaControls

Более того, для одного из своих продуктов мне необходимо сделать скин в цветовой гамме и в стиле сайта заказчика. Скин будет без особых излишеств, но в работе над ним я постараюсь осветить наиболее важные моменты.

Итак, для работы нам понадобится:

- Графический редактор (Photoshop, Gimp, Paint.net или любой другой, который для вас наиболее доступен или удобен).
- Редактор скинов. Взять его можно тут: http://www.alphaskins.com/sfiles/askineditor.zip
- Время, терпение и желание сделать свой рабочий скин =)

Я не буду останавливаться на азах работы с графическим редактором, и предполагаю, что у вас есть некоторые навыки, чтоб самостоятельно суметь нарисовать элементы управление (*контролы*) в том виде, в котором вы хотите их видеть.

Так же я полагаю, что вы знакомы с разработкой приложений и хотя бы догадываетесь, что один и тот же элемент управления (например, кнопка) может в разное время быть в разных состояниях – активный, нажатый, не активный.

Предполагаю, что и опыт работы с AlphaControls у вас имеется, и очень надеюсь, что специфичные слова «контрол», «скин-секция», «градиент» и т.д. не повергнут вас в ступор и вы поймете о чем идет речь.

Итак, если вы уже готовы начать, то приступим!

Шаг 1. Установка, настройка, подготовка.

Пока пишу эту статью, сам одновременно делаю все действия, и их же описываю тут. Потому прошу прощения, если что-то написано не совсем понятным, благородным и литературным русским языком.

Итак, первым делом скачиваем редактор скинов и распаковываем его на жесткий диск. Я распаковал в папочку D:\SkinFactory. В дальнейшем вся работа будет вестись внутри этой папки.

Чтобы не попортить оригинальные скины, внутри D:\SkinFactory была создана папочка Out, чтоб получилось D:\SkinFactory\Out – это та папка, где будет лежать создаваемый нами скин. Если мы будем создавать свой скин на основе уже существующего, то имеет смысл в эту папочку переписать оригинальные скины из папки Skins: если мы загубим какой-то скин в папке Out – не страшно, оригинал останется неизменным.

Теперь можно запускать наш графический редактор и переходить к шагу 2.

Шаг 2. Общие настройки, начало разработки.

Теперь нам нужно хотя бы в первом приближении определиться с внешним видом контролов.

Так как мой скин должен быть похож на сайт заказчика, то обычным PrintScreen я слизываю картинку сайта в буфер обмена и вставляю ее в графический редактор. Часть контролов на скрине уже присутствует, но далеко не все. Впоследствии, при создании других скинов, возможно придется делать несколько дополнительных скриншотов, где будут отображены недостающие контролы. Ну а так как сайт относительно простенький, и многих контролов на нем нет и быть не может – недостающие будем придумывать и отрисовывать самостоятельно.

Так у нас получится разумный компромисс между копированием готового и чистым творчеством.

В графическом редакторе у скриншота отрезаем все лишнее (поля, остатки других окон или рабочего стола, если они есть), оставляя только изображение тех элементов интерфейса, которые нам интересны.

Теперь, когда все подготовительные этапы выполнены, ~~время пить Херши~~ запускать редактор скинов. Что и делаем.

Дла начала запустим мастер создания (кнопка Creation Wizard) и укажем все, что он спрашивает от нас. В появившемся окне укажем имя скина, и он у нас будет называться… э… ну скажем, просто Simple. Переименовать мы его и потом сумеем, а пока у меня нет на этот счет идей.

Далее в открывшемся диалоге укажем папку хранения скина – это будет как раз та самая папка D:\SkinFactory\Out. Редактор сам создаст папку для скина Simple. В итоге полный путь к папке скина получился таким: D:\SkinFactory\Out\Simple. Надеюсь, вы все еще пока не запутались.

Следующий шаг - выбор основных цветов для скина. В моем случае это бело-серебристо-голубая гамма и на текущем этапе я остановился на таком выборе цветов:

Основной цвет – белый. Да, просто белый.

Цвет текста я подсмотрел на скриншоте и он у нас простой черный.

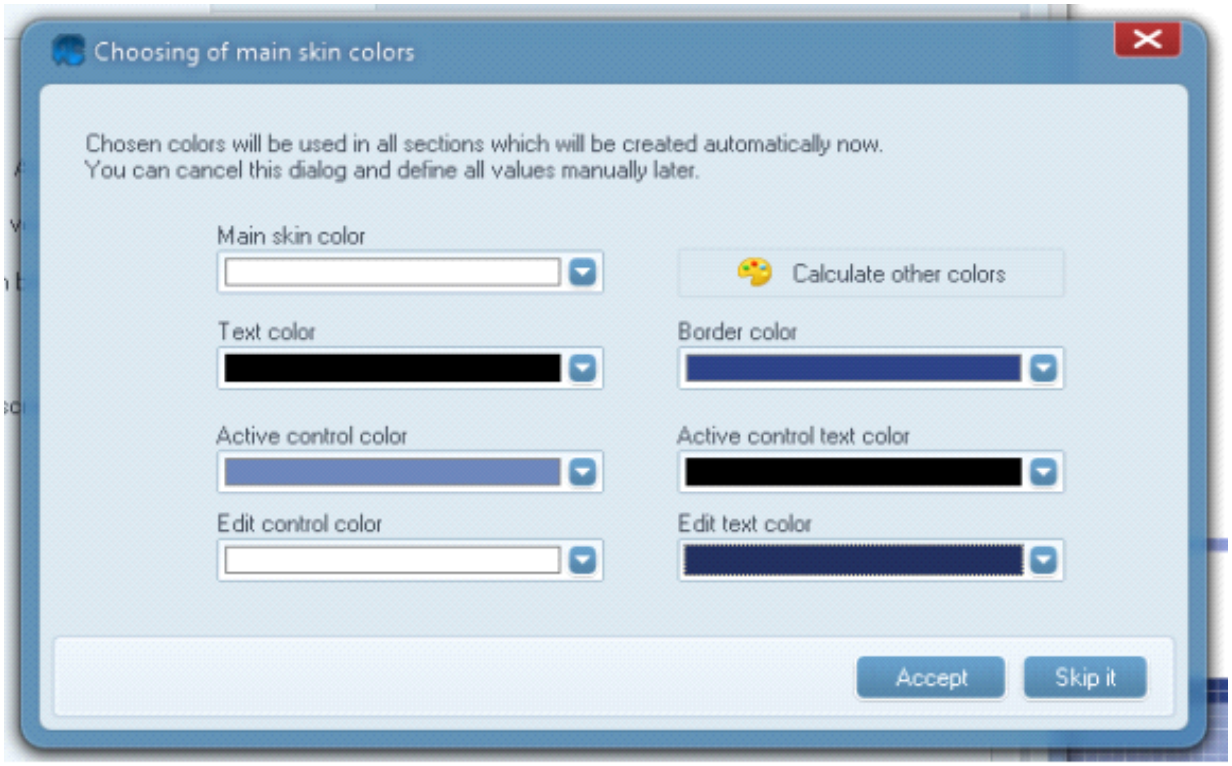
Цвет бордюра у нас темно- синий, вот такой в шестнадцатеричном счислении: #2d4286

Цвет активного контрола я пока еще не знаю как отразится на форме целиком, но пока поставлю его таким: #6b84b8 – холодный грязно-сине-серый Если что – потом посмотрим наглядно и исправим, если он нас не устроит.

Текст активного контрола пока выставим черненьким, хотя на предыдущем цвете и белый был бы не плох. Но пока экспериментируем, экспериментируем и еще раз экспериментируем. Творческий процесс зачастую сопряжен с экпериментаторством.

Цвет Edit'ов у нас будет беленьким. А текст в эдитах будет темно-темно синим, почти черным. Вот таким: #212F5E.

Вот что получилось:



Собственно, все, и с этим шагом пока покончено. Хотя впоследствии, когда будем в демонстрационной программе видеть , как у нас скин буквально вырисовывается, то не

исключаю, что будем менять эти цвета. Но это будет позже, а пока – продолжаем, нажав кнопку «Accept» в редакторе.

Теперь перед нами редактор скинов, в котором мы можем перемещаться по доступным нам вкладкам, которые весьма полно описаны в инструкции, которую я упоминал выше.

Всякие вкладки по типу General Info я оставляю за кадром, ибо сложностей с ними нет, а заполнить их можно в самый последний момент.

Теперь в редакторе можно нажать флажок «Preview Changes», запустится демка * и мы можем видеть, что у нас получается на текущий момент. Пока получается нормально, но все же результат пока далек от конечного.

** Редактор попросит указать местонахождение программы ASkinDemo.exe, но для предварительного просмотра редактируемого скина может использоваться любая программа с включенными AlphaSkins. Можно подключить свою программу и использовать ее постоянно для Preview. Главные требования к такой программе - наличие компонента TsSkinProvider На основной форме и наличие активного TsSkinManager.*

Шаг 3. Вкладка Main Colors.

Так как вкладочку General Info мы оставляем на потом, то переходим к следующему разделу – Main Colors.

Да, часть цветов мы уже задали при помощи мастера, но пока далеко не все. Потому сейчас установим те цвета, которые еще не заданы нами. Это цвета для меток (Labels), меток с тенью (FX Labels) и 2 состояния для WebLabels.

Мне хотелось бы, чтоб метки выглядели так же, как и основной текст, потому делаем их черными.

Метки с тенью – думаю что они тоже должны быть черными. По крайней мере - пока.

С WebLabels все просто – оставляем их цвета фуксия (clFuchsia) и они будут в дальнейшем иметь цвета по умолчанию (черный в обычном состоянии и красный при наведении мышью) ** .

Ну вот, на всякий случай я нажимаю кнопочку «Save»на панели редактора и мы готовы перейти дальше. Готовы ведь? Тогда вперед.

** *Если при выборе цветов остается clFuchsia, то используется цвет по умолчанию, но при отрисовке изображений этот цвет задаёт прозрачные пиксели в углах (прозрачные участки на сторонах элементов управления не поддерживаются для ускорения рендеринга).*

Шаг 4. Секция Title Bar и кнопки формы.

А вот с этого момента нам вполне может уже пригождаться графический редактор и навыки работы в нем. Ведь сейчас речь пойдет о кнопках, которые отображаются в заголовке формы

нашего будущего скина. Разумеется, их можно оставить и теми, что есть по-умолчанию, но это нам не совсем интересно, не так ли?

Итак, в графическом редакторе откроем файлик Master.bmp, который лежит тут : D:\SkinFactory\Out\Simple\Master.bmp. Только, на всякий случай, я сделал резервную копию этого файла, перед тем как открывать его графическим редактором. Так, мало ли что…

Как видим, тут уже есть нарисованные кнопки – это то самое, что используется по умолчанию.

Наша задача – добавить свои кнопочки и убрать ненужные. Для этого я создаю новый слой поверх существующего и все манипуляции буду делать там.

Новые кнопки я хотел бы сделать похожими на кнопки ОС Windows 7, потому в графическом редакторе создаю новый чистый документ размером 105 x 21 – именно в это пространство я попробую вставить 3 системные кнопки – минимизация, максимизация/восстановление и кнопку закрытия окна.

Более того, я скопировал эти кнопки PrintScreen'ом из проводника и вставил их в качестве подложки. Поверх этих стандартных кнопок я собираюсь нарисовать свои, и когда они будут готовы – перенесу их в Master.bmp.

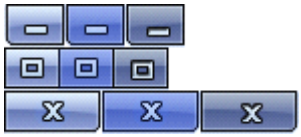
А пока это все выглядит вот таким образом:

Разумеется, мы не будем копировать кнопки системы полностью, но общий стиль сохраним. Вот, например, то, что получилось в итоге:

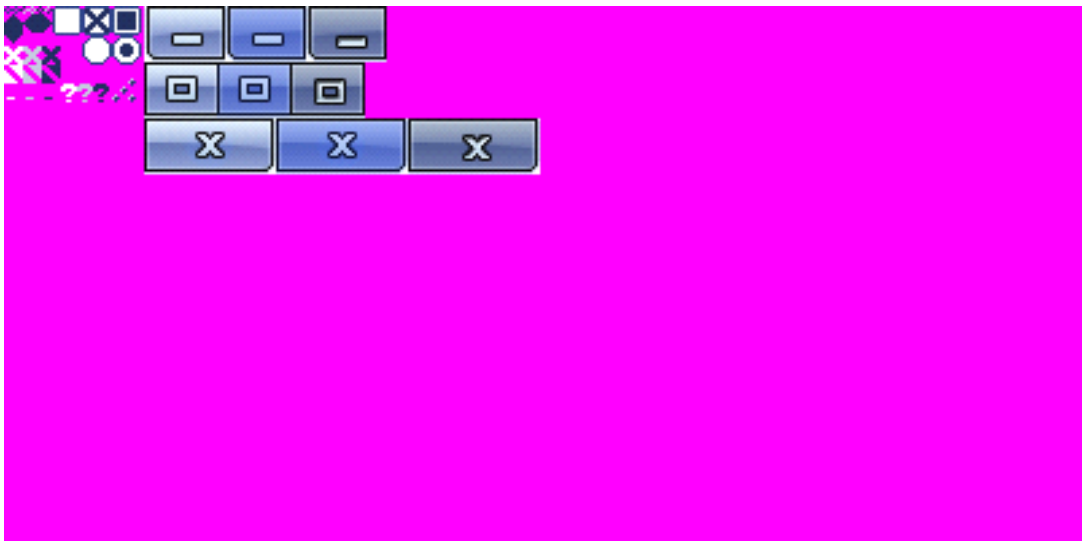
Да, это потребовало некоторого времени и умения работать с графическим редактором. И теперь. Аналогичным образом создаем другие состояния для этих кнопок – наведение и нажатие.

Так как мы пока не будем использовать сложные эффекты типа свечения вокруг кнопок, то копировать их поведение из системы так же не будем. Пока просто для удобства выделим нужное состояние цветами:

Единственное, теперь нужно поменять порядок кнопок – у нас различные их состояния расположены вертикально (сверху вниз), а нужно, чтоб это было горизонтально, слева на право:

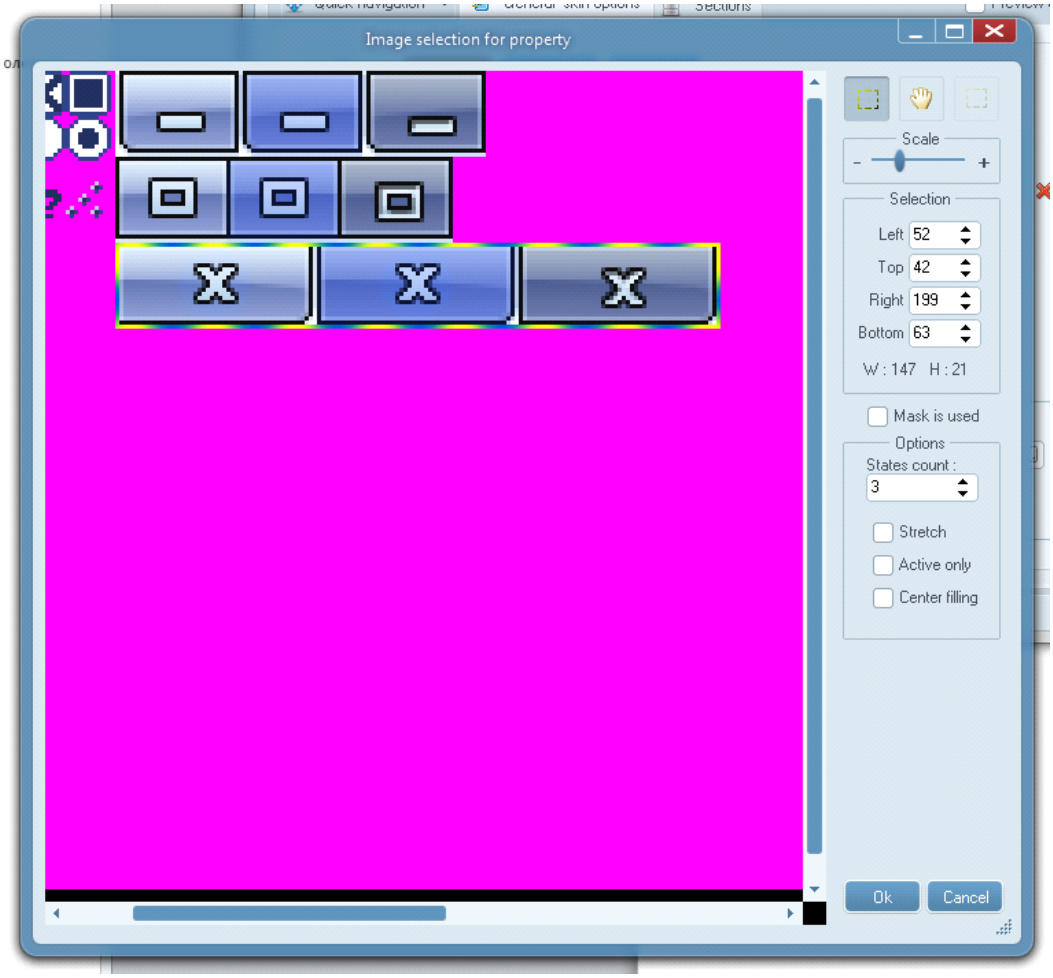


Теперь, готовые рисунки кнопок вставляем в Master.bmp отдельным слоем, не перекрывая то, что уже задано в нем по умолчанию. В итоге у меня получилось так:

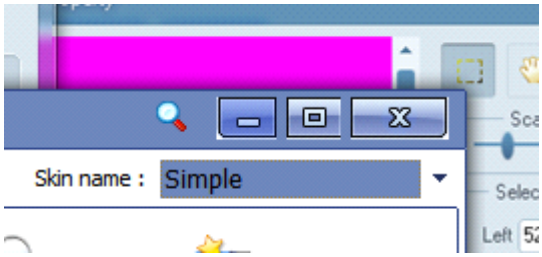


Если вы работаете с программой Photoshop, то bmp-файлы лучше сохранять в формате OS/2. Это связано с тем, что порой Photoshop как-то странно работает с некоторыми bmp-файлами, что впоследствии может привести к ошибке чтения этого файла из скина средствами Delphi VCL.

Итак, после того как кнопки помещены в Master.bmp, в редакторе скинов нужно определить для них новые координаты, что мы и делаем на закладке «TitleBar»:



Теперь наши кнопочки указаны, и можно посмотреть в программе Preview как это будет выглядеть. У меня это выглядит так:



Абсолютно не так, как я себе это представлял! Что же делать?

Во-первых, на все той же вкладке «TitleBar» зададим параметр «Space between buttons» равный нулю. Тут же можно задать и флаг «Buttons are aligned to top», что поднимет наши кнопки к самому верхнему краю заголовка формы. В итоге мы сможем увидеть такой вот результат:

Уже почти все хорошо, но вот красными стрелочками я выделил белесые поля, которые должны быть по идее полупрозрачными, но никак не такими!

Кроме показанных стрелками ляпов можно заметить еще следующие недостатки: во-первых, граница между кнопками минимизации и максимизации составляет 2 пиксела, а между кнопками максимизации и закрытия – всего один. Вроде и мелочь, а не красиво.

Еще один нюанс – это то, что мы не нарисовали кнопку восстановления окна в исходное состояние из максимизированного. Если вы откроете проводник в Windows 7 в не развернутом на весь экран окне, то увидите, что кнопка максимизации нарисована практически так же – т.е. прямоугольник:

А если развернете окно на весь экран, то рисунок на кнопке будет уже иным:

Т.е. эта кнопка помимо 3-х состояний имеет еще и разные рисунки. К счастью, это совсем не проблема, так как редактор скинов учитывает этот момент, и нам нужно лишь нарисовать соответствующую кнопку, разместив ее в Master.bmp и указав ее границы: кнопка Normal в редакторе скинов определяет рисунок, когда окно не развернуто на весь экран, а кнопка Max – когда окно максимизировано.

Итак, еще один шаг сделан, выявлены возможные недочеты и «подводные камни», которые мы теперь все знаем и уже можем исправить. Как это сделать, мы уже рассмотрели. Кроме одного недочета – это белые границы вокруг кнопок. Избежать этого весьма просто, достаточно не рисовать никаких белых границ после рамки кнопок и в уголках для их нормального скругления использовать прозрачный цвет clFuchsia.

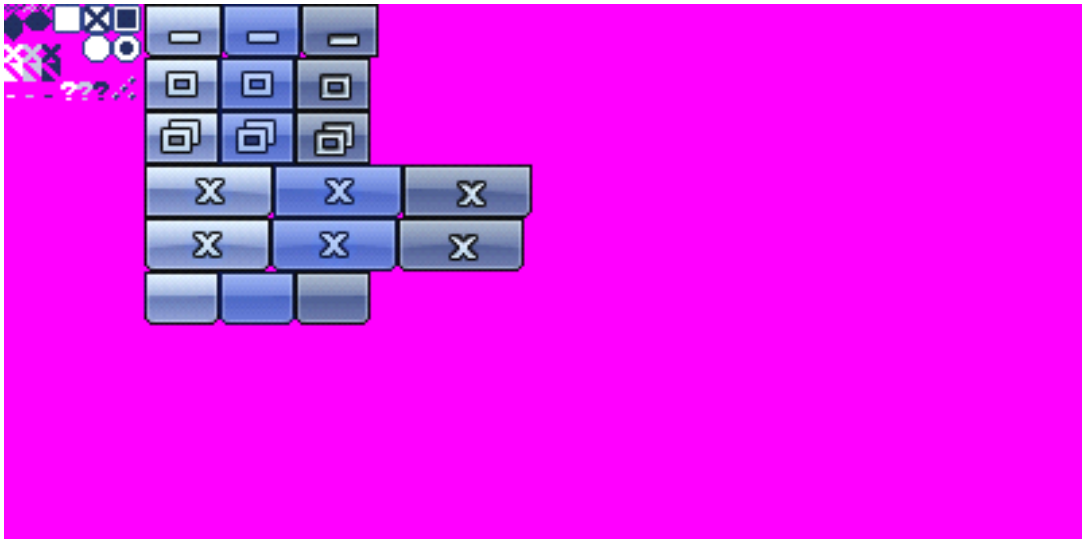
Либо впоследствии можно поменять цвет заголовка окна таким образом, чтоб он сочетался с такими светлыми бордюрами. В любом случае, все зависит исключительно от вашей фантазии.

Для примера можно посмотреть готовые скины с подобными кнопками (Air, Sapphire, Vienna, Vienna Ext и др.), чтоб наглядно увидеть, как это сделано там. Вообще, изучение готовых скинов вкупе с инструкцией на сайте – неплохое подспорье для того, чтоб понять, как реализован тот или иной момент в скине.

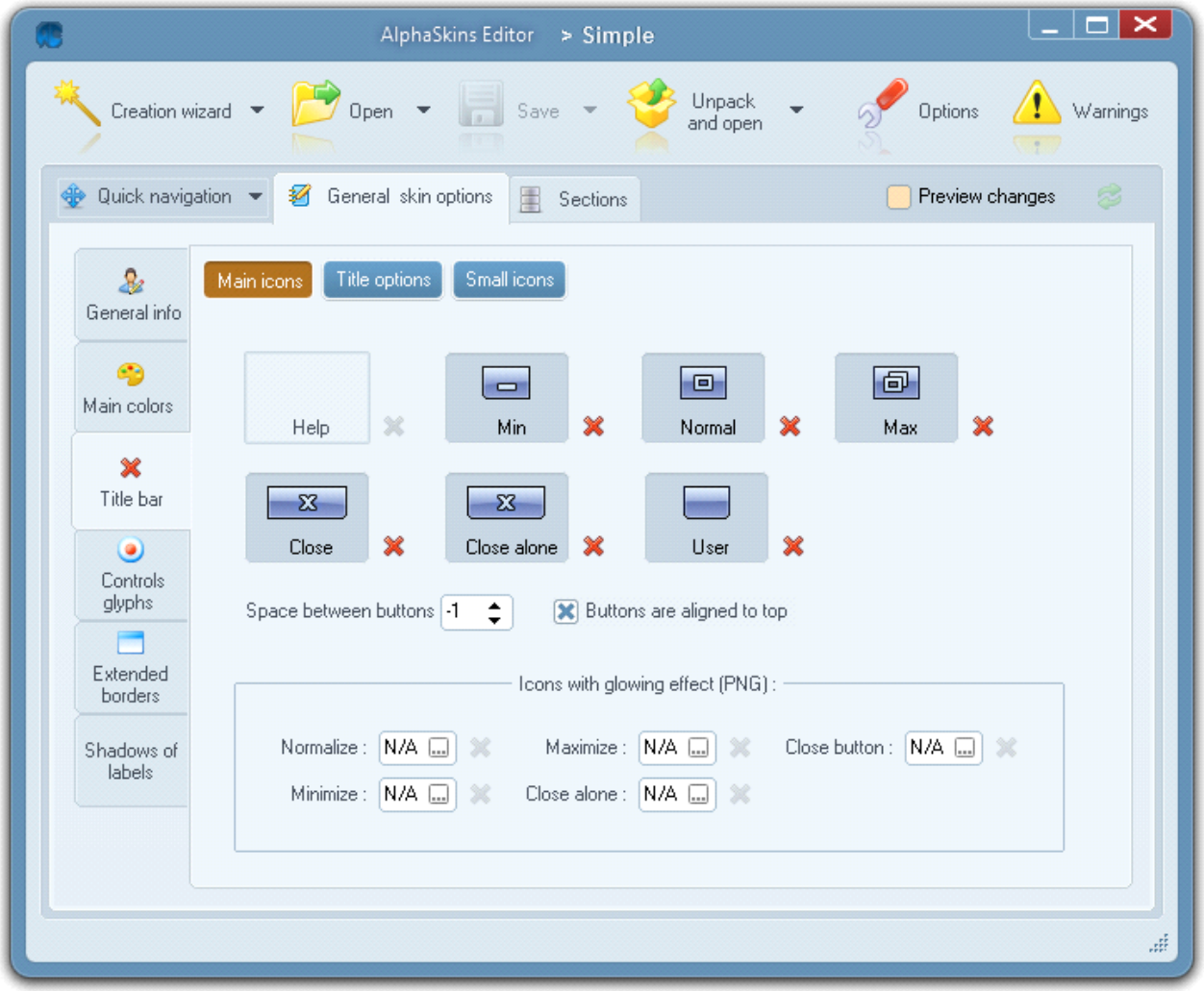
Выше я специально уделил немало времени возникающим ошибкам, описал их, и так же старался рассказать, как их избежать. Впоследствии, когда речь зайдет о других элементах, я надеюсь не затрагивать такие нюансы, и очень надеюсь что и вы, при создании скинов сможете не допускать таких досадных ляпов.

Ну а теперь перерисуем кнопки с учетом всех замечаний. Плюс еще я добавил кнопку пользователя (User) и кнопку закрытия окна, когда в заголовке формы только лишь она одна, например, в диалоговых окнах (кнопка Close alone).

Вот что в итоге у меня получилось в Master.bmp:



А так это выглядит в редакторе скинов:



Тут я немного поигрался со свойством «Space between buttons», и наилучшим значением мне показалось минус единица – в этом случае границы между кнопками смотрятся лучше всего, на мой взгляд. Вообще, используя это свойство можно порой добиться весьма интересных эффектов с кнопочками в заголовке формы.

И, наконец, то, что получилось непосредственно в скине:

По-моему, получилось весьма не плохо. Конечно, в системе кнопка закрытия окна выделена цветом, но здесь я ее не выделял умышленно, дабы градиент располагался равномерно сразу на всю группу кнопок. Так что все пока идет согласно творческому замыслу

Если посмотреть на окно редактора скинов, изображенное выше, то можно увидеть группу «Icons with glowing effect (PNG)». Эти инструменты позволят нам задать свечение для кнопок в тот момент, когда курсор мыши находится над ними.

Для этого нам потребуется несколько файлов в формате PNG, которые содержат изображение той или иной кнопки в состоянии, когда мышь находится над кнопкой.

Что же, кнопки уже нарисованы, и сделать из них PNG-файлик со свечением не составляет труда, чем и займемся (если, конечно, вам будут нужны светящиеся кнопки).

Так как кнопка светится у нас лишь в состоянии, когда на нее наведена мышь, то и делать будем свечение только для такого состояния кнопки, а получившееся файлы сохраним в папку нашего скина.

Так как гамма у нас сине-белая, то пусть и свечение будет темно синей тональности. При этом я не хотел бы сильно выделять подсветкой все кнопки, кроме кнопки закрытия. А ее хочу сделать чуть ярче, чем остальные. Посмотрим, что у нас получится.

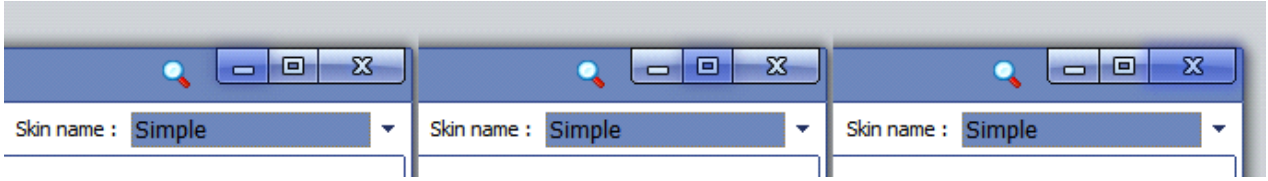
Кстати, таким образом можно назначить не только разную интенсивность свечения кнопкам, но и можно назначить свечение только одной какой-то кнопке, например, той же кнопке закрытия. Опять же – все зависит от нашей задумки. А в данном случае я попробую подсветить все кнопочки.

Когда кнопки готовы, то для каждой из них указываем свой PNG файл. При этом не забудьте верно установить параметр Internal Margin, который определяет внутреннюю границу внутри PNG-файла со свечением от его края, до кнопки (сколько пикселей от края до бордюра кнопки, т.е. ширина зоны свечения). В моем случае, при создании свечения, я дополнительно растягивал размеры картинки на 10 пикселей с каждой из сторон кнопки, т.е. если кнопка была размером 28 x 20, то итоговый размер картинки стал 48 x 40. Таким образом, Internal Margin у нас получился равен 10 пикселям. Если это свойство не настроить должным образом, то есть риск вместо свечения получить смещенную в самое неожиданное место кнопку. Так же это свойство можно использовать для создания некоторых спецэффектов. Например, если я бы выставил Internal Margin равный 11, то при наведении на кнопку курсора мыши получил бы слегка приподнимающуюся и как бы выпирающую над формой кнопку с подсветкой.

Так же для одной кнопки можно добавить сразу несколько файлов, но на данный момент работать будет только первый добавленный файл. Так что пока использовать несколько кадров для одной кнопки использовать не получится. Возможно, в будущем эта возможность будет доступна, и комбинации из нескольких файлов могут быть отображены самым причудливым образом . Или в виде анимации.

В любом случае – будьте готовы, что сразу желаемое может не получиться, и возможно придется потратить немало времени, прежде чем цель будет достигнута.

Вот что получилось в итоге с кнопками, если использовать разные PNG картинки:



Теперь создание кнопок практически завершено, но в закладке «Title Bar» есть еще разделы «Title options» и «Small Icons» помимо «Main Icons».

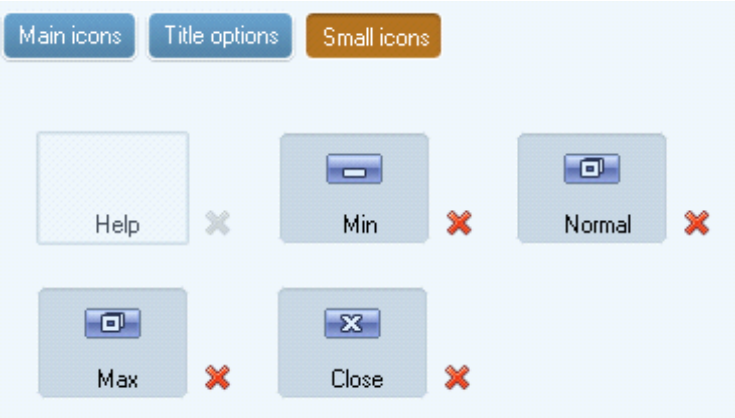
Опции заголовка, или «Title options» небольшой, но очень важный раздел. Тут мы можем задать поведение кнопок заголовка при изменении цветовой гаммы скина через HUE – можно сделать так, что бы кнопки заголовка меняли цвет вместе со скином, либо не меняли вовсе, либо кнопка закрытия окна никогда не меняла бы свою гамму.

Чуть ниже в этом же разделе можно задать смещения содержимого заголовка (в том числе и наших кнопок) внутри самого заголовка. Я оставил все значения выставленными в ноль.

Эти параметры тоже весьма интересны для экспериментирования. Например, сделав параметр Right отрицательным, скажем, -10, то можем увидеть, что кнопки в заголовке несколько смещены вправо и выпирают за пределы формы (при активном режиме Extended Borders). Конечно, в повседневной жизни это вряд ли пригодится, хотя с другой стороны и этой фиче можно найти применение при создании какого-нибудь, авангардного скина.

Раздел «Small Icons» служит для задания уменьшенных версий кнопок заголовка. Такие кнопки применяются, например, в компоненте sPageControl на табах или при вызове диалоговых окон sInputBox, sShowMessage и др. Подробно я на этом моменте останавливаться не буду, так как все вполне очевидно – нужно изготовить маленькие варианты системных кнопок, нарисовать их в Master.bmp и задать рабочие области для них в редакторе скинов.

В редакторе скинов это выглядит следующим образом:



Итак, с кнопками заголовка формы мы разобрались и переходим к следующему шагу.

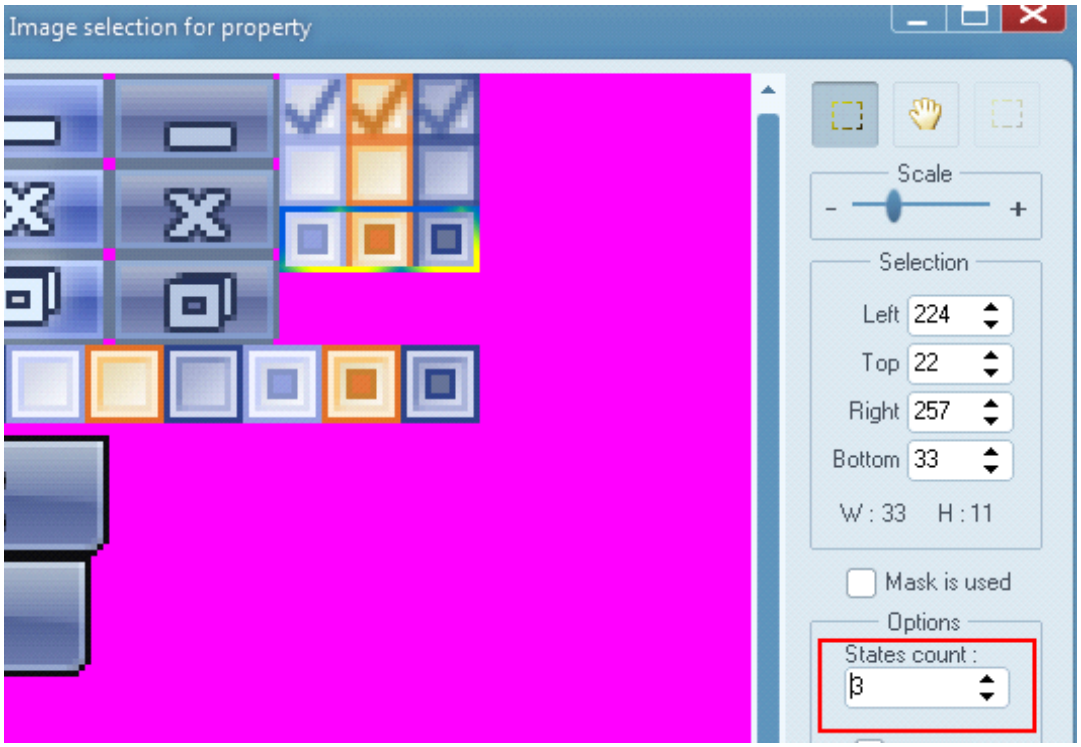
Шаг 5. Чек-боксы.

Работа с чек-боксами в редакторе скинов располагается в разделе «Controls glyphs». Значит, переходим туда и продолжаем.

Этот шаг скорее творческий, так как на данном этапе нам нужно отрисовать в графическом редакторе всевозможные состояния чек-боксов и малых чек-боксов. Потом остается определить необходимые области в Master.bmp через редактор скинов.

Итак, после недолгих терзаний графического редактора вот такие картинки получились для чек-боксов:

Кстати сказать, когда мы создаем скин заново (как сейчас), то по умолчанию, все элементы на этой страничке имеют одно состояние (нет разделения на нормальное, активное, нажатое состояния контрола), потому будет совершенно не лишним при задании области для элемента, установить параметр States Count равный трем (соответственно, для нормального, активного и нажатого состояний). Этот параметр находится вот тут (я его выделил рамочкой):



Да и впредь будет не лишним не забывать про этот параметр и активно его использовать в зависимости от того, сколько состояний для контрола вы хотите назначить.

Малые чек-боксы я сделал аналогичными, просто уменьшив картинку исходных чек-боксов на пару пикселей, что хорошо видно на рисунке выше. Часть деталей изменилась, но я посчитал это не особо критичным моментом.

Шаг 6. Маски.

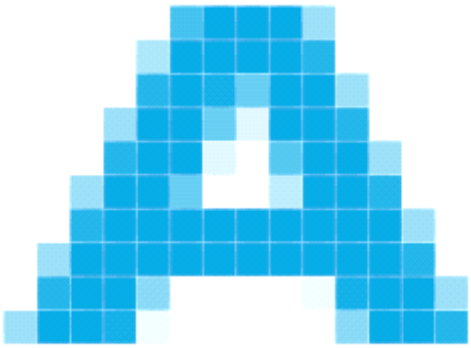
Следующим моментом в создании нашего скина стало рисование радио-кнопок. С ними пришлось повозиться чуть дольше, так как они имеют не прямоугольную форму, как те же чек-боксы, и нужно было сделать хорошо сглаженный круг на прозрачном фоне, и при этом использовать маски не хотелось. Но с другой стороны, почему бы и не попробовать, хотя изначально я этого делать и не хотел. Так что к радио-кнопкам мы вернемся несколько позже, а теперь попробуем разобраться с масками.

Про маски сказано весьма просто и хорошо в инструкции на сайте, и действительно, ничего сложного в них нет. Итоговое изображение формируется путем наложения маски на картинку с контролом, и получается весьма убедительный эффект полупрозрачности.

Попробую наглядно продемонстрировать этот процесс.

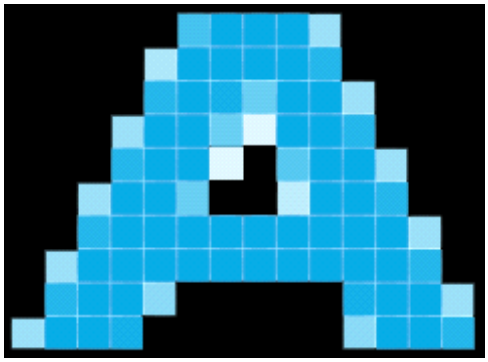
Пусть у нас есть картинка элемента, где нужно использовать полупрозрачность. Вот, например, буква «А»:

Очень хорошо видно, что буква прорисована четко, отдельных пикселей не видно (если, конечно, не сделать увеличение в несколько раз), т.е. наша буква выглядит *сглаженной*. Так же обратите внимание, что буква находится у нас на белом фоне. Это весьма важный аспект. Если эту самую картинку увеличить так, что бы мы смогли рассмотреть отдельные пиксели, то получится примерно такое изображение:



Отлично видно, что по краям символа пиксели имеют более светлые оттенки (и этих оттенков несколько), чем основной цвет.

Теперь представим, что фон, на котором будет располагаться наш элемент в виде буквы, не всегда будет белым. Просто зальем его другим цветом (например, черным) и с увеличением посмотрим, что получилось:



Замечательно видно, что те самые пиксели, которые отличаются от основного цвета и тут свой цвет не поменяли. Но если при белом фоне от этого нам была только польза, так как буква была сглаженной, то сейчас, уже без увеличения, наш символ выглядит так:

Согласитесь, не очень хорошо, не правда ли? Отчетливо видны те самые светлые пиксели, находящиеся по краям. Именно из-за этого символ выглядит не опрятно, с рваными краями. Т.е. в одном случае, с белым фоном, эти пиксели светлых оттенков нам помогли добиться эффекта сглаженности, то на черном фоне они только мешают.

Именно для того, чтобы избежать подобных неприятностей и используются маски, которые позволяют элементам выглядеть одинаково хорошо на любом фоне.

Давайте попробуем исправить ситуацию и задействуем маски.

В библиотеке AlphaControls принято, что маски могут быть исключительно в градациях серого, т.е. любой пиксель маски должен быть серым, причем его оттенок может варьироваться от белого до черного (считается, что черный – это очень-очень темный серый, а белый – он вовсе не белый, а тоже серый, но только очень-очень светлый).

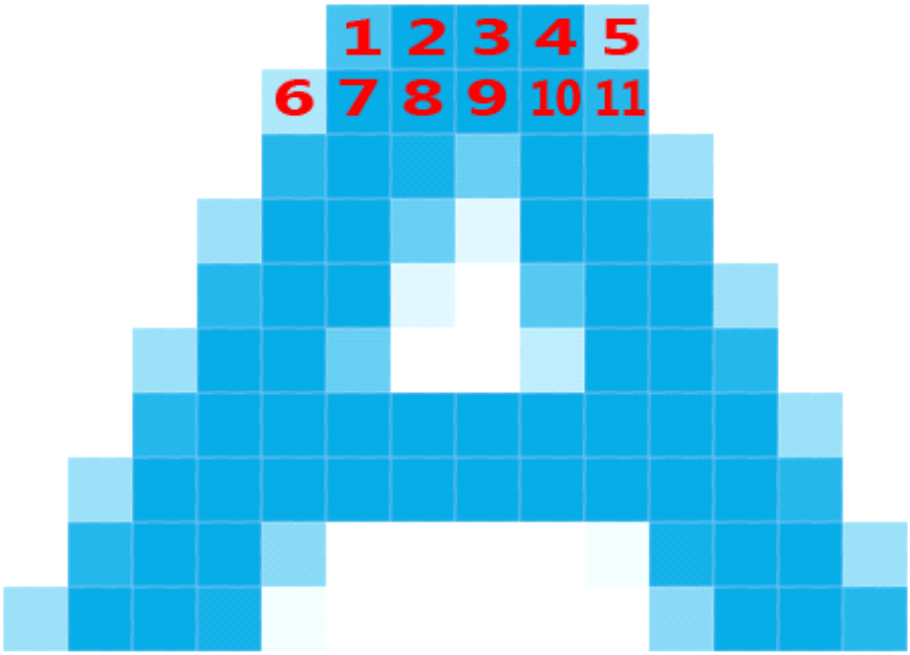
Сама маска должна полностью, один в один, повторять наш рисунок и совпадать с ним по размерам. Отличия будут лишь в цветах. Если наш исходный рисунок может быть цветным, то маска всегда будет в оттенках серого.

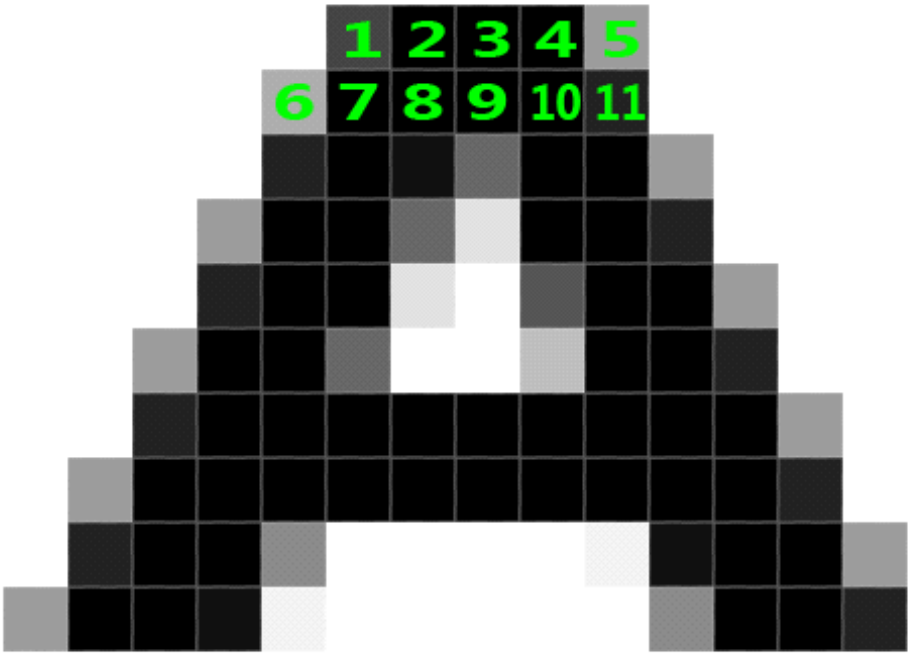
Важно понять, что сама по себе маска служит инструментом, который определяет прозрачность пикселей нашего рисунка. Для простоты понимания можно считать, что сама маска не рисуется, а при отрисовке нашего объекта каждый пиксель маски сравнивается с таким же пикселем в исходной картинке, и на основании оттенка серого цвета маски, пикселю нашего объекта назначается степень прозрачности.

Самый темный (черный) цвет пикселей в маске задает абсолютную непрозрачность оригинального изображения, самый светлый цвет (белый) – абсолютную прозрачность. Все остальные оттенки серого цвета, находящиеся между белым и черными цветами будут определять степень прозрачности и непрозрачности в зависимости от интенсивности.

Запомнить это очень легко, как мне кажется – черный – не прозрачный, белый – прозрачный.

Попробую проиллюстрировать сказанное рисунком, на котором я нарисовал наше изображение и ниже него – маску. При этом я пронумеровал некоторые пиксели на самом изображении и на маске:





При рисовании нашего элемента в виде буквы «А», библиотека возьмет пиксель №1 из исходного изображения (а он у нас должен быть чуть-чуть прозрачным), и возьмет пиксель №1 из маски. В нашей маске пиксель №1 темный, но все же не совсем черный. Именно поэтому

конечный пиксель, который отрисует библиотека – будет слегка прозрачным. Т.е. через него будет просвечивать фон, на котором лежит наша буква «А».

Пиксели 2,3,4 должны быть не прозрачными, потому и в маске они залиты черным цветом. Т.е. в финальном изображении будет нарисовано в точности то, что соответствует пикселям 2,3,4 в исходной картинке с нашей буквой.

5-й пиксель нашего изображения должен быть еще более прозрачным, чем первый, соответственно, в маске он обозначен умеренным серым. Такая же картина и с пикселем №6, только он еще чуточку прозрачнее и в маске так же 6-й пиксель более светлого серого оттенка. Далее, с 7 по 11 пиксели идут в маске черными, что скажет библиотеке, что в финальной картинке пиксели 7-11 нужно нарисовать так, как это задано в исходном изображении.

Думаю, что произойдет с остальными, не пронумерованными пикселями на картинке – и так понятно – будет использован тот же принцип, и в итоге наш символ будет иметь сглаженность по краям, и при этом цвет фона под ним будет не важен, чего мы и хотели добиться!

А теперь, думаю, самое время опробовать это на практике.

Для этого я оставил наш элемент в виде буквы «А», и подготовил итоговую картинку, которую собираюсь вставить в редактор скинов:

Обратите внимание, что полностью прозрачные куски рисунка все так же закрашены цветом clFuchsia, т.к. наша маска будет применена непосредственно к оригинальному изображению, а не к фону. Именно потому фон у нас прозрачный, т.е. цвета фуксия.

Сглаживающие пиксели у изображения и его маски выглядят на фоне clFuchsia не очень хорошо, все так же видны рваные края, как и в первых примерах, расположенных выше. Дело в том, что фон нашего скина в целом светлый, оттого и сглаживающие пиксели выглядят так, как они должны были выглядеть для белого фона.

Здесь так же есть место для экспериментов, и, будь наш скин темным, то сглаживающие пиксели для нашего рисунка вероятнее всего выглядели бы иначе, дабы сочетаться с темным фоном.

Вообще, при создании элементов, использующих маски в скине, не лишне иной раз запустить демонстрационную программу из редактора скинов, и посмотреть, что же получается, и при необходимости сразу внести коррективы.

Наш экспериментальный элемент я назначил радиокнопке, не забыв выставить флаг «Mask is used» в том же окне, где задаются границы элемента.

И вот что в итоге получилось (картинки из редактора скинов и демо-программы):

Как видим, наша экспериментальная радиокнопка в виде буквы «А» замечательно выглядит на разных фонах, что и требовалось получить!

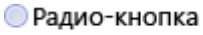
Надеюсь, мой рассказ о масках и их использовании в библиотеке AlphaControls вас не запутал, а напротив, помог лучше понять, как работать с масками, если до этого момента это было не очевидно. На самом деле, понимание масок будет тем лучше, чем больше будет практики работы с ними.

А сейчас вновь время применять наши знания на практике, так что предлагаю продолжить создание нашего скина.

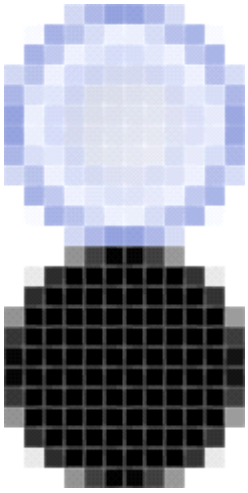
Шаг 7. Радиокнопки и прочее.

И вот мы снова возвращаемся к радио-кнопкам, которые, как я уже говорил в шаге про чек-боксы имеют круглую форму, и при этом могут располагаться на различных фонах. При этом очень хочется добиться эффекта, чтоб наши радио-кнопки выглядели красиво на любом фоне. Это значит, что сглаживание окружности радио-кнопки мы будем делать с использованием масок.

Первым делом в графическом редакторе я нарисовал обычную ради-кнопку, которая должна выглядеть так:



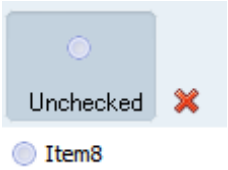
На белом фоне кнопка выглядит неплохо, но мы ведь помним, что цвет может быть не только белым? Добавляем маску, просто скопировав нашу кнопку ниже, и закрасив ее черным цветом, оставив серые пиксели по краям там, где нам нужно будет сглаживание. Вот что получилось (я специально увеличил изображение, чтобы лучше можно было увидеть маску):



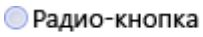
Теперь, прежде чем работать над другими состояниями нашей радиокнопки, я хотел бы посмотреть, как оно будет смотреться в скине, и, в случае, если мне что-то не понравиться, сразу это исправить. Так будет значительно проще и быстрее, чем потом переделывать все состояния этой кнопки.

Прежде чем добавить изображение кнопки и ее маску в Master.bmp, не забываем задать цвет прозрачности фону. При этом справа от нашего пробного рисунка я уже предусмотрел место для других состояний кнопочки, но пока потестируем ее в единственном состоянии:

Теперь можно добавить картинку с радио-кнопкой в Master.bmp. И вот что я увидел в редакторе скинов и демонстрационной программе:



Не плохо, кнопка смотрится одинаково и там, и там, но есть ощущение, что чего-то не хватает. Мне кажется, граница кнопки не достаточно четко прорисована, как минимум по сравнению с первоначальным рисунком кнопки (это тот, который был еще без маски). Вот он еще раз, для сравнения:



Разница хоть и не слишком велика, но она есть. Но хоть с маской, хоть без маски – в обоих случаях сам рисунок кнопки был один и тот же. Следовательно, влияние оказывает именно маска. Или же можем скорректировать рисунок кнопки, это тоже будет эффективно, но я решил все же поиграть именно с маской.

Возвращаемся в графический редактор, открываем Master.bmp и там доводим все до нужного состояния. Возможно, маска, созданная и примененная в этот раз и далека от идеала, но тем не менее, поставленная задача ею выполнена.

В принципе, использовать маску в подобных случаях не всегда обязательно. Если открыть уже готовые скины, и посмотреть их внутренее устройство, то можно увидеть, что во многих скинах маски вообще не используются, и при этом скины выглядят очень даже отлично. Так что использование масок, впрочем, как и других средств, всегда остается на усмотрение автора скина.

А вот что получилось в итоге с радио-кнопками:

Теперь в секции «Control Glyphs» редактора скинов дело осталось за малым – нарисовать группу контролов, расположенных в секции «Miscellaneous».

На этом моменте я не буду заострять особого внимания, все делается аналогично предыдущим шагам.

Шаг 8. Расширенные границы или Extended borders.

После того, как мы заполнили все в разделе «Control Glyps», переключаем редактор скинов в раздел «Extended borders» для того, чтоб рассмотреть возможности по созданию не стандартных границ формы.

Вообще необходимо помнить, что пользователь скина может выключить Extended borders в компоненте sSkinManager, потому эту часть можно пропустить, скин превосходно будет работать в таком режиме. Более того, по умолчанию, если все же режим Extended borders будет включен, то к форме по ее периметру добавится очень симпатичная тень, окружающая форму. Это очень просто наблюдать, если мы ничего не меняли в данном разделе и запустили демонстрационную программу с нашим пока еще не готовым скином, и уже в демонстрационной программе понаблюдать за поведением формы, нажимая кнопку «Native borders».

Тем не менее, будем последовательными и попробуем разобраться как создавать и использовать свои «расширенные границы».

Стоит отметить, что расширенные границы формы могут функционировать в двух режимах – это полное замещение существующих стандартных границ формы, и добавление к стандартным границам каких-либо спецэффектов, например, тени или свечения.

Тут же можно отдельным флажком «Allow Aero bluring» изменить режим смешивания (блюра) бордюров под Windows Vista и старше, если Aero будет включен в системе. Смешиваться будут пикселы рамки, у которых значения R, G и B не равны 0, то есть пикселы не черного цвета. Это сделано для того, чтобы можно было нарисовать тень черного цвета, которая не будет смешиваться с остальными пикселами.

И попробуем рассмотреть остальные параметры, которые можем задавать в этом разделе.

Прежде чем я начну перечислять исследуемые параметры, отдельно подчеркну, что они влияют на работу скинированной формы в целом, и для отображения эффектов, на которые влияют параметры, используются системные функции, потому при некоторых значениях может наблюдаться странное поведение приложения, вплоть до крэша программы. Потому при задании значений руководствуйтесь здравым смыслом, и не стоит делать, например, высоту заголовка формы -30 пикселей, например. И, конечно же, тестируйте и смотрите, что у вас получается в итоге, и при необходимости корректируйте.

Extended border (PNG) – это поле для указания внешнего PNG-файла (так же, как и с подсвечиваемыми кнопками в заголовке формы в шаге 4), в котором долен быть размещен рисунок границы формы, работающей в Extended borders режиме. К этому параметру мы еще вернемся несколько позже.

Border width, как следует из названия, задает ширину границы формы, работающей в режиме Extended borders. Иначе говоря, это расстояние между тенью и клиентской частью формы. Пока я установил это значение равное 15 пикселам. Будьте аккуратны с этим параметром, так как при некоторых значениях возможно появление артефактов или не правильная отрисовка формы.

Параметр **Shadow size** доступен если у нас расширенная граница *добавляется* к существующей, а не заменяет ее. Этот параметр указывает процедуре рисования форме размер тени, расстояние от начала тени до рамки.

Если в **Extended border (PNG)** был указан какой-то PNG-файл с границами, и у нас включен режим замещения границ формы, а не добавляения, то вместо параметра **Shadow size** будет доступен параметр **Content offset**, указывающий расстояние от начала тени бордюра. Практически это то же самое, что и Shadow size, но имеет другое название из-за другой технологии отрисовки.

Параметр **Title height** определяет высоту заголовка формы. При настройках по умолчанию у Windows 7 эта высота (при включенном Aero) составляет (включая рамки) 30 пикселей. Здесь я не стал отступать от этой традиции и установил высоту заголовка для нашего скина, так же, равной 30 пикселей.

Следующий параметр **Maximized title height** определяет высоту заголовка формы в случае, когда форма максимизирована (развернута на весь экран). Но в данном случае есть некоторые ограничения, накладываемые Windows 7. Технически невозможно изменять высоту заголовка максимизированной формы плавно, на каком-то шаге система изменяет его скачкообразно. Поэтому очень желательно проверять как ведет себя скин под Windows 7 при максимизированном окне.

Title center offset – параметр, который отвечает за вертикальное смещение содержимого заголовка формы (заголовок, дополнительный заголовок, пользовательские кнопки).

Параметр **Form align offset** позволяет откаллибровать смещение формы для эффекта прилипания к краям экрана. Это необходимо в некоторых скинах из-за особенной формы расширенных бордюров. В большинстве случаев этот параметр можно игнорировать.

Теперь вновь вернемся к полю **Extended border (PNG)**. В этом поле мы можем указать и настроить PNG-файл, который будет использован при создании расширенных границ нашей формы. Вообще тут довольно обширный простор для всяческого рода экспериментов, но при создании такого файла я отталкивался от следующих моментов:

- Состояний у формы будет 2 – активное и не активное, значит нам нужна картинка, на которой будут отображены эти 2 состояния.

- Как вы уже поняли, создаваемый скин в некоторых местах перекликается с интерфейсом Windows 7, и расширенные границы формы не стали исключением. Следовательно, на рисунке нам нужен заголовок формы, ее углы и центр, которые потом будут применены к настоящей. Живой форме.

- Я не против использовать существующий опыт, и потому взял файлик внешних границ из скина Vienna Ext в качестве основы, ориентируясь на которую создал свой вариант границ:

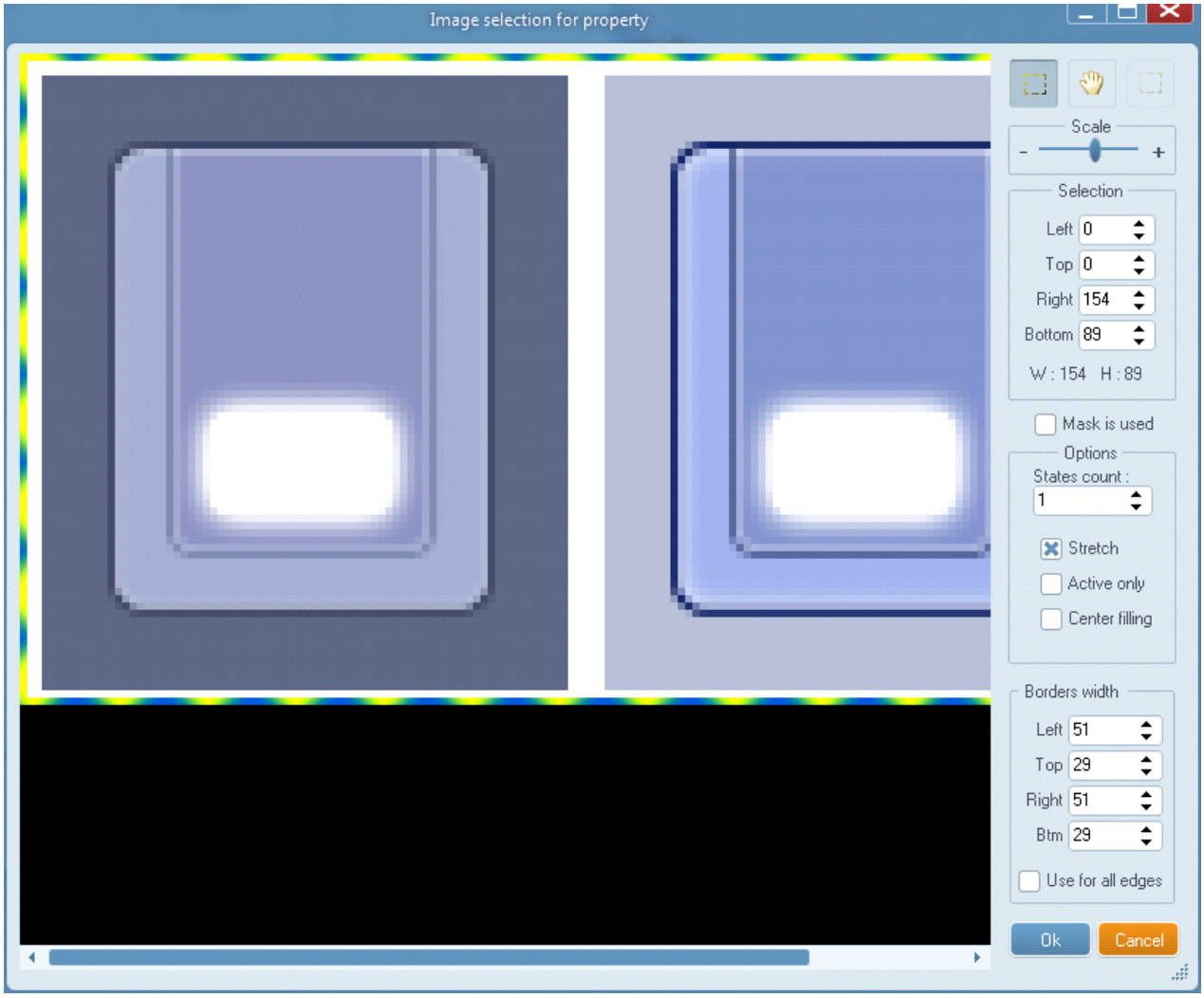
Разумеется, этот вариант получился далеко не сразу, и было несколько промежуточных вариантов, экспериментируя с которым я и получил этот итоговый результат.

Именно этот рисунок, похожий на 2 перевернутых номерка гардероба и станет основой наших внешних рубежей, эээ-м, границ. Левый – для неактивной формы, правый – для активной.

Здесь он выглядит несколько пересветленным, но это абсолютно нормально, так как эта картинка PNG частично прозрачна. Так мы решим еще одну задачу с небольшой полупрозрачностью не клиентской части формы.

Белое пятно в центре – это как раз область клиентской части и только она абсолютно не прозрачна.

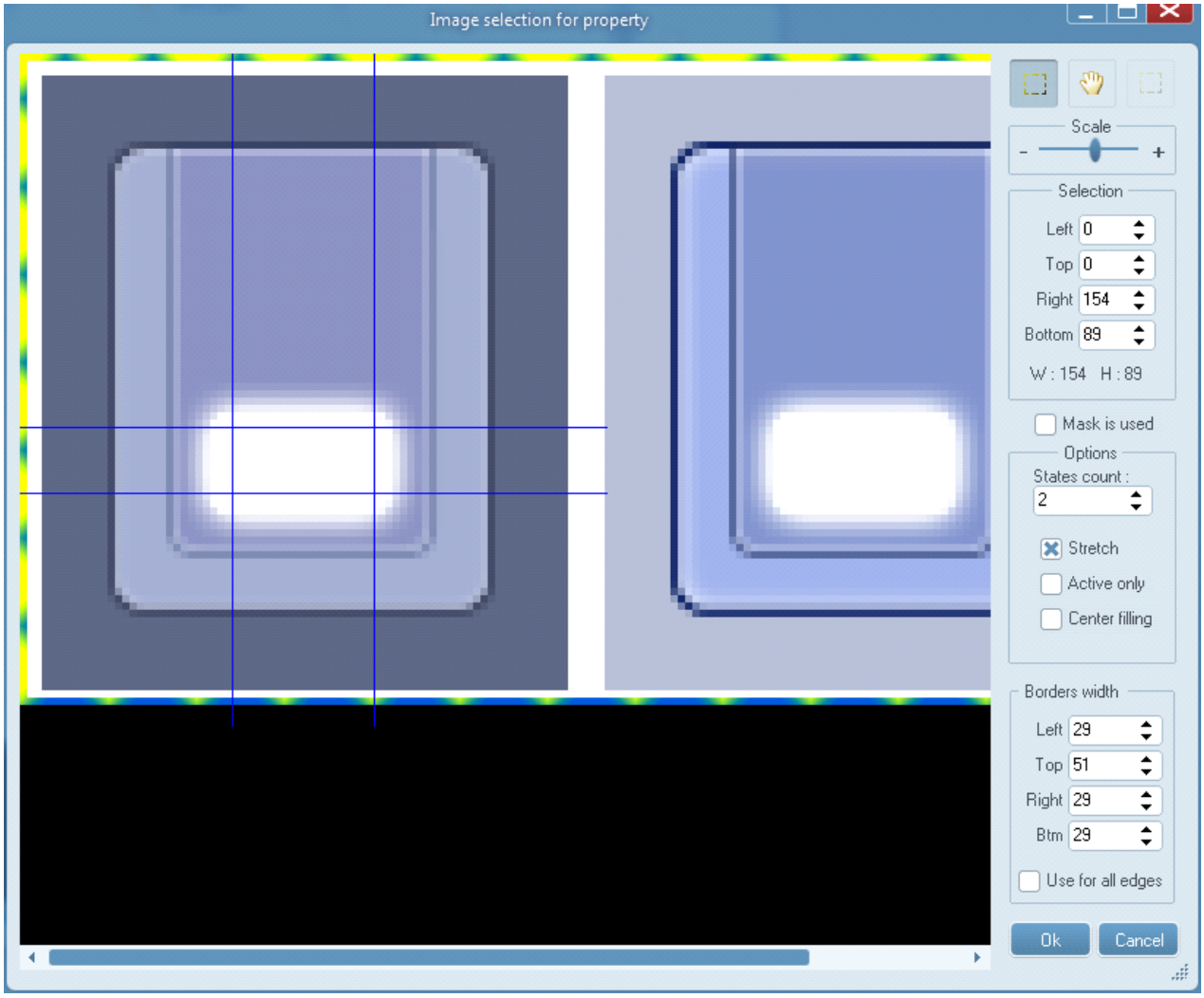
Теперь укажем наш файлик в пункте **Extended border (PNG)** и увидим примерно следующее:



На этом рисунке сразу же не забываем исправить параметр States Count на 2, и переключаемся на группу параметров Borders width, которые, судя по названию. Задают ширину нужных нам участков (заголовок, границы снизу, слева и справа, клиентская область и углы).

Важно заметить, что на рисунке расстояние от его краев до самого бордюра должно быть одинаково со всех сторон (на примеденном примере оно равно около десяти пикселей). Это означает, что место отведенное для тени должно быть одинаковым со всех сторон.

Как только фокус ввода попадет в одно из этих полей ввода, картинка изменится и мы сможем увидеть разграничительные линии, которые покажут нам, как будет разделен исходный рисунок:

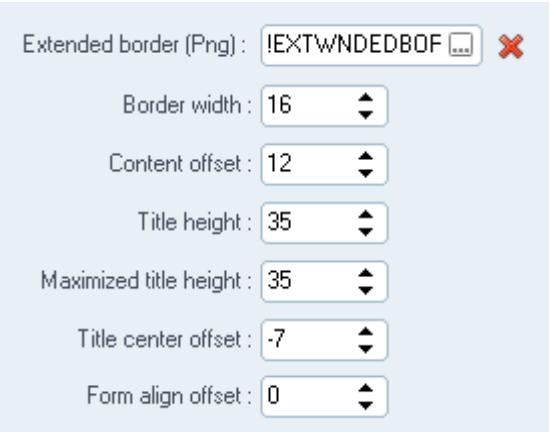


Меняя эти параметры, я добился такого разграничения частей границы, как показано на рисунке выше. Если бы наши границы основывались на другом рисунке, то и значения отступов были бы иными. Менять эти параметры нужно 1 раз – для неактивной формы. Эти же значения автоматически будут применены на соседнее изображение активной формы, так как подразумевается, что вторая часть имеет такие же размеры как и первая.

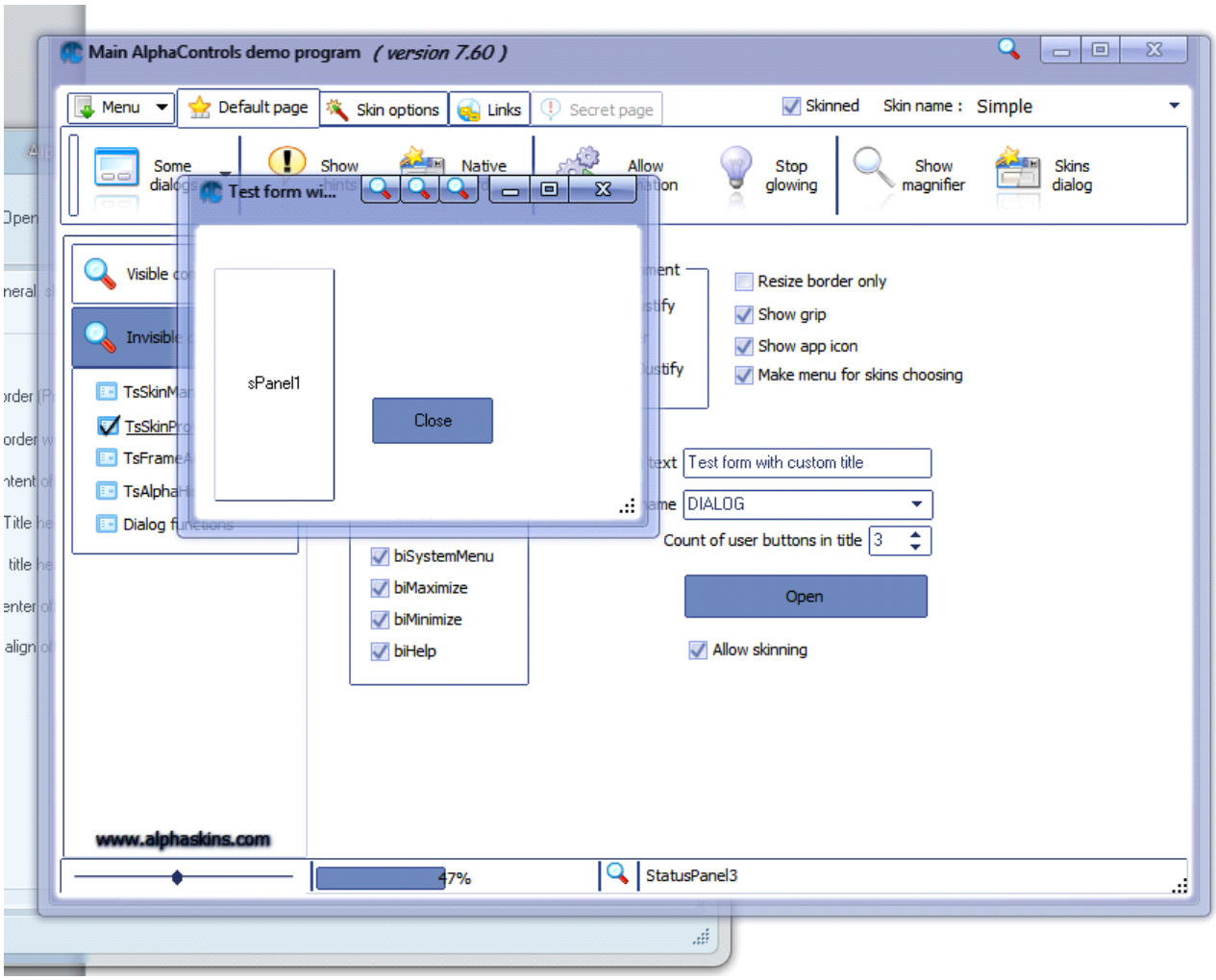
Небольшое замечание хотелось бы оставить для особо наблюдательных – если присмотреться, то видно (особенно по белой клиентской части), что левое поле несколько шире правого. На самом деле это иллюзия, оптический обман На самом деле все абсолютно нормально, просто редактор не совсем точно отрисовывает разграничительные линии. Жить и работать это абсолютно не мешает, так что паниковать и подгонять разграничительные линии не стоит – все решают не они, а цифровые значения, выставленные в полях справа.

Все, жмем кнопку **Ok**.

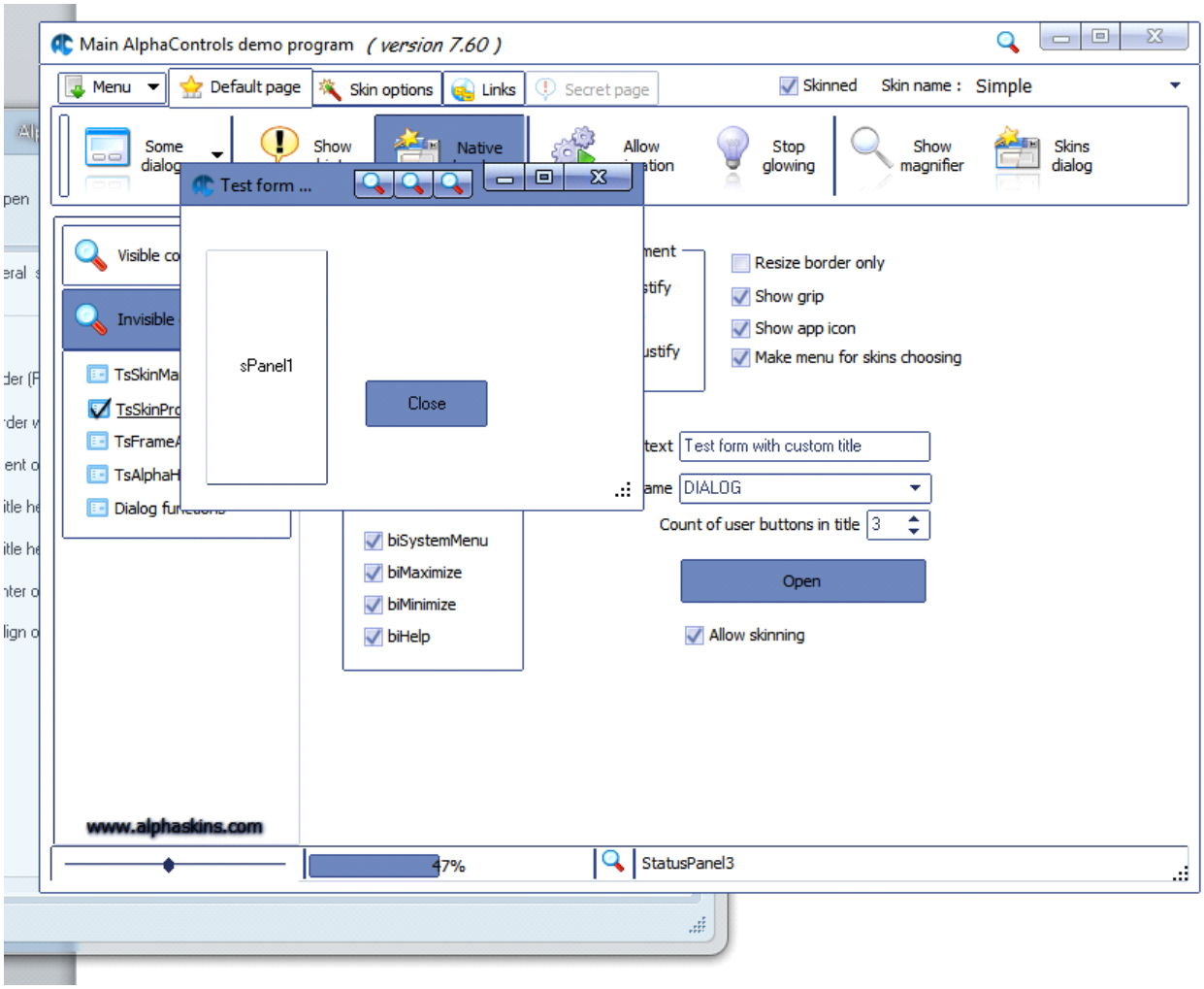
И теперь, путем тестирования выясняется, что некоторые параметры, которые мы выставили ранее, несколько не сочетаются с новыми внешними границами. Поэтому опытным путем я изменил их значения на следующие:



Теперь попробуем посмотреть, что же у нас получилось в итоге?



Наш скин, хотя его еще нельзя назвать готовым, но, тем не менее, заметно оживился, благодаря использованию Extended Borders. При этом и без них он смотрится ровно настолько, насколько мы его настроили :



Наш интерфейс далек от совершенства, впрочем, и до решения поставленной задачи еще далеко, так что продолжаем!

После написания этих строк возникла у меня некоторая блажь поэкспериментировать с расширенными границами еще немного (да, этот процесс затягивает) и в результате эти самые границы стали выглядеть несколько иначе, чем на скриншотах выше. Но, так как дело коснулось только лишь цветовой гаммы, и ничего принципиально не изменилось, скриншоты я решил оставить прежними.

Итак, теперь можно сказать, что с расширенными границами мы относительно разобрались (можно, конечно, экспериментировать еще, но я предоставлю эту возможность вам), и потому переходим к следующему шагу.

Шаг 9. Метки и их тени.

Это скорее даже не шаг, а маленький шажок, поскольку раздел «Shadows of labels» редактора скинов очень не велик и с ним все более чем просто: в единственной группе задаем цвет тени

для компонента sLabelFX, смещение (Offset) и степень размытия тени в пикселях (Blur count). Делать это лучше всего экспериментальным путем.

По умолчанию FX-метки выглядели так:

Invisible component

После экспериментов – так:

Invisible component

Я поменял цвет тени на светло-синий, смещение поставил равным нулю, а размер размытия выставил в 2. Таким образом, наш sLabelFX стал читаться не в пример лучше, и отличается он от обычного лэйбла наличием легкого, небольшого синего ореола-свечения вокруг букв.

Не смотря на всего 3 параметра, ответственных за вид компонента TsLabelFX, здесь вполне есть где развернуться экспериментам. Но это мы так же оставив на потом, остановившись на сделанном варианте, тем более что с этими настройками мы разобрались и пора переходить на следующий раздел редактора, который… которого нету!

Все, наконец-то мы закончили обзор разделов редактора скинов, попытались охватить многие из его возможностей, попутно создавая свой собственный скин. Но вот как раз скин еще и не закончен, несмотря на отсутствие в редакторе закладок на страничке General Skin Options.

Но, разумеется, это далеко не конец, и нас ждет, пожалуй, самое творческое и интересное – **создание скин-секций** и работа с ними. Но это уже тема следующего шага и следующей статьи.