Филиппович Анна Юрьевна

Лекционные материалы по дисциплине "Компьютерная графика" 2006-2007 уч. г.

Лекция 2 (4 часа)
Шрифты.

План занятия

1. Создание факсимильных шрифтов и орнаментов. Форматы шрифтов, кодировка. Особенности разметки шрифтов. Растровые шрифты. (4 часа).

Теоретическая часть

1. Программные средства создания шрифтов.
2. Конверторы шрифтов.
3. Переиздание «старых» книг.
4. Особенности разработки факсимильных шрифтов.
5. Этапы разработки факсимильных шрифтов.
6. Особенности использование сканированных изображений.
7. Оценка качества шрифта.
8. Понятие о формате и шрифтовой машине.
9. Структура шрифтового формата.
10. Кодировка символов.
11. Управление растеризацией символов.
12. Методы разметки символов.
13. Формат PostScript.
14. Формат TrueType.

Демонстрационная (практическая) часть

Разработка факсимильных шрифтов или орнаментов. Использование сканированных изображений символов для разработки шрифтов и орнаментов.

Программные средства создания шрифтов

С появлением персональных компьютеров и распространением на их основе настольных издательских систем разработка шрифтов не обходится без использования специальных программных средств. Такими средствами служат так называемые программы-конструкторы шрифтов. Одними из самых популярных и распространяемых программ-конструкторов шрифтов являются Macromedia Fontographer и FontLab.

FontLab

Программа создания шрифтов. TypeTool – облегченная версия FontLab. В настоящее время хорошо развитая и поэтому широко распространенная система создания шрифтов. Имеет следующие основные функции:

1. Glyph Editor - среда создания знаков;
2. TrueType & Type 1 Hinting - ручной и автоматический хинтинг;
3. VectorPaint Tools - инструменты для работы с векторными объектами;
4. FontAudit Technology - технология автоматического выявления и устранения проблем с контурами создаваемых знаков;
5. Font Metrics and Kerning - профессиональный редактор метрик и кернинга шрифтов с автоматическими функциями;
6. Transformations - различные трансформации, применяемые как к отдельным знакам, так и к их группам;
7. Font Header Editor - доступ к редактированию всех свойств шрифта от имени и поддерживаемых кодовых страниц до специфических TrueType-метрик.

В четвертой, версии FontLab появилась полная поддержка формата OpenType - импорт, создание, редактирование, экспорт и конвертирование. Также в более ранних версиях была введена подобная поддержка шрифтов в формате Multiple Master. Очень полезна новая функция Macro Language, которая позволяет на языке Python писать не только скрипты, но и создавать собственные инструменты и плагины (plugins), что значительно расширяет возможности программы. Также нельзя не отметить новые инструменты для работы с контуром - Knife, Magic Wand, 3D Rotate, Scale and Slant, Sketch-режим для создания нового контура Ikarus-подобными инструментами. В дополнение широкие возможности настройки интерфейса пользователя, позволяющие настроить клавиатурные сокращения и новые кнопоки на палитре инструментов.

Производитель *FontLab Ltd.* - http://www.fontlab.com/html/fontlab.html

Macromedia Fontographer

Профессиональная программа создания шрифтов. Несмотря на то, что программа давно не обновлялась, она до сих пор считается одним из наиболее профессиональных редакторов. Помимо присущих данным программам функций, Fontographer позволяет импортировать/экспортировать изображения в формате EPS, а также кернинговые пары. Авторы могут генерировать шрифты в форматах PostScript Type 1 и TrueType.

Производитель *Macromedia, Inc.* – http://www.macromedia.com/software/fontographer/ .

TypeDesigner

Профессиональный редактор. TypeDesigner имеет не только инструменты для создания и модификации шрифтов, но и целый ряд функций автоматизирующих рутинные операции. Среди возможностей программы можно отметить: поддержка редактирования шрифтов в форматах Type 1 и TrueType; одновременное редактирование до восьми шрифтов; тест-принтинг с кучей функций; глобальные преобразования (stretch, italicize, shift position, change boundaries и т.д.); автокернинг; глобальное изменение ширины стэмов; контекстно-зависимая помощь; автохинтинг с регулируемыми параметрами; редактор кернинга; любые повороты и зеркальные отображения; любые операции над контуром; импорт EPS, Calamus CFN fonts; 10 уровней отмены/повтора (Undo/Redo) и проч.

Производитель *DTP Software.* - http://www.dtpsoft.de/

GOTE - GNOME OpenType editor

Развивающаяся программа. Работает только со шрифтами в формате TrueType. В следующей версии создатели обещают поддержку Type1. Пока набор функций невелик, хотя создать шрифт "с нуля" все-таки возможно. В своей работе программа использует библиотеки Gnome - specifically, glib, gdk, gtk+, gnome, gnomeui, libglade. Эти библиотеки включены практически во все последние дистрибутивы Unix/Linux, включая FreeBSD, Solaris и Irix.

Разработчик - *Robert Brady (Department of Electronics&Computer Science, University of Southampton)* - http://gote.sourceforge.net/

Softy

Уникальный редактор для создания TrueType и bitmap шрифтов. Его автор - Дэвид Эммет (David Emmett). Этот редактор пользуется огромной популярностью во всем мире среди начинающих дизайнеров шрифта. В редакторе имеются все необходимые функции для создания и модификации шрифтов. На выходе поддерживаются форматы TrueType, FON, FNT, LaserJet SFP, SFL.

Разработчик *David Emmett* - http://users.iclway.co.uk/l.emmett/

Font Creator Program

Font Creator является программой довольно среднего уровня. Позволяет конвертировать заготовки из растрового формата (.bmp) в вектор, а также создавать и редактировать шрифты в формате TrueType. Из функциональных особенностей можно отметить: чтение и запись TTF-шрифтов, конвертирование "растр->вектор", примитивные средства работы с кривыми, комбинирование и разбиение контуров, неограниченное количество откатов/повторений (undo/redo), кернинг, PCL5 window, разбиение композитных глифов на простые, окно предварительного просмотра результатов (контрольный текст), Unicode-маппинг, автокернинг, автометрикс.

Производитель *High-Logic* - http://www.high-logic.com/fcp.html

PfaEdit

Очень динамично развивающийся и многообещающий редактор на базе UNIX для создания и редактирования шрифтов в форматах Type 1 и TrueType. По количеству функций и удобству использования его можно поставить между FontLab и Fontographer. А уж среди *nix-платформ он является бесспорным фаворитом. Очень большим плюсом можно считать возможность корректной конвертации шрифтов в разные форматы под разные же платформы - Type 1, TrueType (PC, UNIX и Mac). Советую разработчикам присмотреться к данной программе.

Разработчик *George Williams* - http://pfaedit.sourceforge.net/

Alphabet Synthesis Machine

Весьма примечательная программа, реализованная в виде Java-апплета. При желании, любой веб-серфер может попробовать себя на ниве создания шрифтов. Только создаваемый вами шрифт и близко не будет похож на кириллицу или латиницу. Дело в том, что механизм данного редактора построен так: предполагаемый автор заходит на страницу с загруженным апплетом Alphabet Synthesis Machine, рисует некий знак (совсем необязательно, чтобы он был похож на какую-либо букву кириллицы или латиницы), а ASM, опираясь на параметры этого знака, достраивает весь алфавит. Причем, для генерации остальных знаков, используется алгоритм построения генов. Получившийся продукт, более похожий на шрифт иноземной цивилизации, можно сохранить на своем компьютере в TTF-формате.

Подобных шрифтов создается примерно по 50 штук в день. При желании можно покопаться в архиве данного проекта.

Разработчики *Golan Levin, Jonathan Feinberg, Cassidy Curtis* - http://alphabet.tmema.org/entry.html

BDF Font Edito

Простенький редактор шрифтов написанный на Tcl/tk под *nix. Позволяет создавать и модифицировать шрифты в формате BDF.

Производитель *Thomas A. Fine* –

http://hea-www.harvard.edu/~fine/Tech/bdfedit.html

Pilot Font Editor

Интересный простенький редактор шрифтов под PalmOS. Включает в себя: fontedit (собственно сам редактор Pilot Font Editor), GetFonts (утилита загрузки системных шрифтов) и FontHack123 (утилита для замены системных шрифтов вашими разработками).

Разработчик *Sergey Menshikov* - http://www.sergem.net/fontedit/

LaserJet Bitmapped Font Editor

Редактор bitmapped-шрифтов под DOS. Максимальные размеры шрифта - 110 pt (VGA), 80 pt (EGA), 88 pt (Herc&AT&T), 44 pt (CGA). Имеется целый набор специальных эффектов. Возможен импорт черно-белых изображений в форматах .PCX и .TIF. К сожалению, поддерживаются не все модели мышей.

Производитель *Alexander Walter* - http://www.walterware.com/fe.html.

BitCopy

Довольно интересный редактор для создания, модификации и конвертирования шрифтов. BitCopy позволяет легко создавать bitmapped-шрифты из масштабируемых для PCL- и PostScript-принтеров. Работает со всеми стандартными шрифтовыми форматами, включая PostScript Type 1, TrueType и FastFont. В плане редактирования BitCopy позволяет: поворачивать символы, "накладывать" тень, инвертировать (white/black), генерировать жирное и тонкое начертания (относительно нормального), масштабировать, создавать новые знаки с помощью "аппликации" и проч.

Производитель *Lytrod Software* - http://www.lytrod.com/BitCopy.htm

Конверторы шрифтов

Кроме программ-конструкторов шрифтов существует большое количество конверторов шрифтов. Приведем примеры некоторые из них.

TransType

Программа работает под платформами Win и Mac. Позволяет конвертировать TrueType и Type 1 шрифты между обеими платформами, а также просто из одного формата в другой. Поддерживает также формат Multiple Master.

Производитель *FontLab Ltd.*- http://www.fontlab.com/

CrossFont

Программа работает в среде Windows 95, 98, NT, 2000, XP и конвертирует TrueType и PostScript Type1 шрифты между платформами Macintosh и PC. На входе и выходе поддерживаются форматы AFM, PFM, INF, PFA, .dfont. При конвертации сохраняются все метрики и хинтирование. Должен честно признаться, что не считаю результаты работы этой программы достаточно корректными и удовлетворительными.

Производитель *Acute Systems* - http://www.asy.com/scrcf.htm.

TrueBlue

TrueBlue - бесплатная утилита Mac для конвертации шрифтов из формата True Type (TTF) в PostScript Type 1 (PS). Причем конвертированные шрифты можно одним кликом, сразу, установить в систему. Поддерживается пакетный режим, т.е. все работы по конвертированию можно проводить как с отдельными файлами, так и с целыми папками, содержащими шрифты. Помимо стандартной процедуры конвертирования, есть возможность менять имена шрифтов и целых семейств, записывать их в форматах Binary/Ascii encoding, поддерживается трансляция в latin1, latin2, latin4, latin5, а также кириллица (русская, болгарская и т.д.).

Производитель *Stone Studio* - http://www.stone.com/TrueBlue/

Переиздание «старых» книг

С развитием информационных технологий способы хранения информации перешли на принципиально другой уровень. И если раньше таким способом была запись на бумажном носителе, что ограничивало объем и уменьшало надежность такого вида хранения, то сейчас использование информационных технологий позволяет нам значительно увеличить эти два показателя.

Многие книги, созданные в эпоху только зарождения печатного дела требуют более эффективного способа хранения. Конечно, с уверенностью нельзя утверждать, что хранение информации на электронном носителе более надежно, так как такой способ хранения еще слишком «молод». То есть, у нас нет тысячелетнего опыта хранения информации в электронном виде. При этом существуют книги, которым сотни лет и, не смотря на пожары и наводнения, до нас дошли пусть не все, но часть из книг прошлого. Но, не смотря на это, недолговечность материала очевидна.

Кроме этого, важно отметить, что электронный способ хранения обеспечит большую доступность информации. Очевидно, что например не каждый человек может позволить купить себе ту или иную книгу, не каждый может прочитать или посмотреть то или иное издание. «Старые» книги разбросаны по музеям и библиотекам всего мира, поэтому доступ к ним ограничен. В связи с этим можно говорить не только об эффективности хранения информации, а также об ее доступности. Поэтому вопрос переиздания книг прошлого и записи их в электронном виде является сейчас актуальным.

Существует несколько способов переиздания «старых» книг. Современные печатные технологии базируются на электронном тексте. И даже если печатным источником является оригинал-макет, представляемый в печатном виде, все равно редактура, корректура, верстка осуществляется на компьютере. Поэтому одним из способов переиздания «старых» книг является набор текста с последующей корректурой, редактурой и т.д. Но тут важно подчеркнуть некоторую особенность. Любая книга, не важно когда она создана, имеет свой зрительный образ. Этот зрительный образ отражает представления того времени. Поэтому он является вполне понятным и естественным. Если же мы нарушим этот зрительный образ, то, возможно, будет нарушен баланс между внутренним содержанием предмета и его внешним образом. Именно поэтому при переиздании некоторых «старых» книг стараются сохранить их первоначальный образ. И такие переиздания называются «факсимильными». Теперь вполне понятным является то, что факсимильные переиздания в точности воспроизводят оригинал.

Существует два способа факсимильного переиздания. В первом случае формируют изображение каждой страницы книги и печатают его в соответствующем формате. Для этого предварительно необходимо отсканировать каждую страницу книги, обработать ее.

Недостатки такого способа переиздания очевидны. Если книга находится в плохом состоянии, то изображения страниц требуют значительной обработки. Зачастую улучшить качество изображения невозможно. Кроме этого объем хранения графического файла изображения страницы значительно превышает объем текстового файла.

И наконец, для осуществления поиска по электронной книге, переизданной вышеуказанным способом необходимо будет фрагментировать изображение, осуществлять его индексирование и т.д. Но это касается печатных книг, если переиздается рукопись, то первый способ единственно правильный.

Другой же способ факсимильного переиздания позволяет исключить все вышеперечисленные недостатки. В этом случае создается оригинальная гарнитура, который был набран текст – это так называемая «Факсимильная гарнитура». Далее осуществляется набор, ввод текста с последующей его редактурой, корректурой. Потом осуществляется верстка, причем каждая сверстанная страница должна полностью повторять оригинал. К достоинствам такого переиздания можно отнести и возможность его хранения в электронном виде. Причем информация в данном случае будет храниться как текст, а не как изображение.

Особенности разработки факсимильных шрифтов

Факсимильный шрифт может совсем не отличаться от любого другого шрифта. Фактически многие существующие шрифты и целые гарнитуры были созданы на основе ранее существующих шрифтов и гарнитур. Сама эволюция печатного дела обусловила этот факт. Поэтому справедливо утверждать, что многие гарнитуры являются в том числе точной копией своих старинных собратьев, то есть являются факсимильными. Но важным является цель, которую преследуют создавая ту или иную гарнитуру. И если цель – факсимильное переиздание, и гарнитуру создают для этого, то безусловно речь идет о факсимильной гарнитуре. Но, не углубляясь в дебри определений, подчеркнем некоторые особенности разработки факсимильных шрифтов.

Этапы разработки факсимильных шрифтов

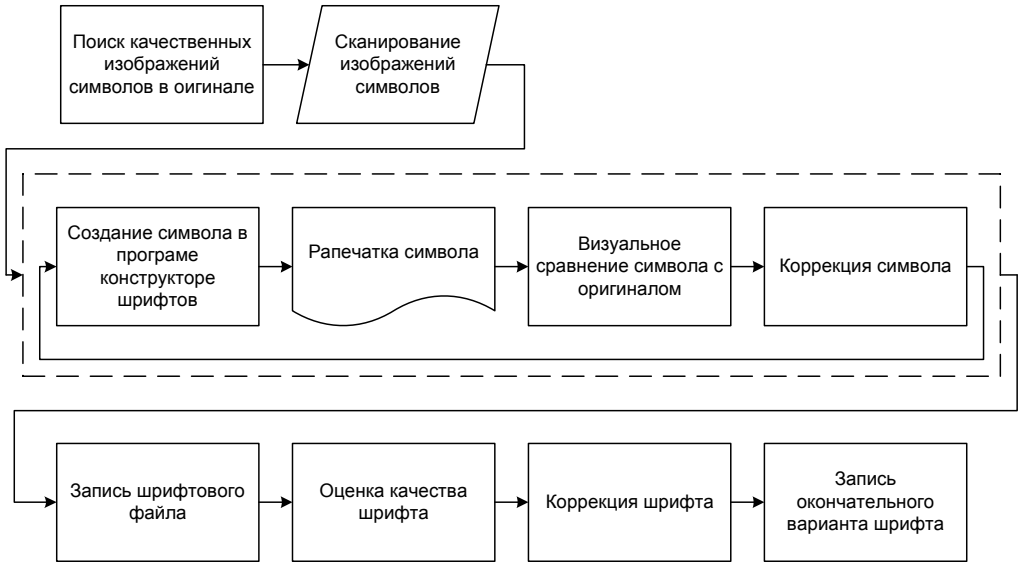


Рисунок 3.1. Схема создания факсимильного шрифта.

Процесс разработки факсимильного шрифта во многом не отличается от «классической схемы» разработки шрифта, однако некоторые существенные отличия существуют. И эти отличия продиктованы необходимостью точного повторения оригинала.

На первом этапе разработки в книге-источнике находятся все буквы алфавита создаваемого шрифта. Причем выбираются наиболее качественные изображения символов, так как возможно оригинал книги находится в плохом состоянии.

Особенностью создания факсимильного символа является возможность использования в качестве прообраза его отсканированного изображения. Поэтому следующим этапом все найденные символы сканируются с высоким разрешением и обрабатываются с помощью растрового редактора, например Adobe Photoshop.

Далее рисуются соответствующие символы в программе-конструкторе шрифтов, к примеру Fontographer. Следует отметить, что использование для этой цели других векторных редакторов, допустим Adobe Illustrator, нежелательно, так как они не содержат в себе всех функций необходимых для профессиональной разработки шрифтов.

На самом деле теоретически, можно создавать не векторный шрифт, а растровый, так как размер букв фиксируем и кроме этого известны параметры всех устройств вывода. То есть если разрешение принтера, на котором будет выводиться оригинал-макет известно и все его характеристики тоже, то можно использовать этот вариант. Однако этот вариант трудно реализуем. Дело в том, что растровые шрифты сейчас используются для отображения на экране, где разрешение значительно ниже, чем при печати. Разрешение экрана – 72 pixel/inch, а разрешение у стандартного лазерного принтера 600-1200 dpi.

Поэтому для вывода на печать используются векторные шрифты. Наиболее распространены такие форматы как True Type и Post Script. И этими стандартами оперируют все текстовые редакторы, они ориентированы на периферийное оборудование. Вообще такая постановка нерациональна по причине большого объема таких шрифтовых файлов.

Когда создан первый вариант рисуемого символа, тогда осуществляется проверка его на соответствие оригиналу. Для этого создается шрифтовой файл, и символ выводится на печать. Естественно, если оригинал макет будет сдаваться в печатном виде, то символ печатается на том принтере, на котором будет и сам оригинал макет.

Итак, созданный символ печатается на принтере и визуально сравнивается с оригиналом. Далее рисунок символа исправляется до тех пор, пока не будет достигнуто полного совпадения. И так с каждым создаваемым символом. При этом нельзя не учитывать некоторые особенности вывода текста. В зависимости от того, какое устройство вы используете, качество напечатанного символа будет разным. Так к примеру при типографском выводе краска немного расплывается, поэтому буквы становятся немного толще. И сравнивая с буквой, напечатанной на лазерном принтере, стоит штрихи символа сделать немного тоньше, тогда при печати в типографии в результате получится именно так как в оригинале.

У разработчика шрифта может возникнуть вопрос: «Какой символ стоит рисовать первым?» Вопрос конечно немаловажный, с чего начать? Можно конечно начать с первой буквы алфавита и так далее – до последней. Но думаю, что это не очень рационально. Начать лучше с того символа, который содержит в себе элементы других символов. Так, к примеру, буквы: «Н», «К», «И» все имеют одинаковую форму вертикального (основного) штриха, поэтому можно начать с этих букв, так как потом необходимо будет использовать эти элементы в других символах, чтобы достичь единства рисунка. Лучше начать с символов с более простыми рисунками.

Кроме этого можно начать со строчных букв, т.к. в тексте строчные буквы используются гораздо чаще прописных, поэтому то, как будет выглядеть текст в будущем, становится более понятным уже на первых этапах разработки шрифта.

Оценка качества шрифта

Чтобы оценить качество шрифта необходимо, во-первых, определить его характеристики. «Факсимильный шрифт должен обладать всеми теми характеристиками, которыми должен обладать любой другой шрифт: художественные достоинства, удобочитаемость, емкость» [Филиппович А.Ю., 2005, с.].

Кроме вышеперечисленных характеристик шрифта существуют внутренние критерии качества шрифта [2, с.232-240]:

1. **Качество разметки;**

Правильная разметка отдельных символов и шрифта в целом оказывает очень большое влияние на качество воспроизведения текста, особенно на выводных устройствах с невысокой разрешающей способностью — дисплеях и матричных принтерах.

2. **Полнота набора знаков;**

Для полноценного использования шрифта необходимо, чтобы он имел полный набор знаков в соответствии с некоторым стандартом. Если шрифт предполагается использовать в одной из программ, работающих под управлением MS Windows, то он должен быть выполнен в соответствии со стандартом 1251 фирмы Microsoft. Минимальный набор знаков, без которого работа со шрифтом будет весьма затруднена, включает все буквы (прописные и строчные), цифры, знаки препинания и некоторые специальные символы, например тире (его не стоит путать со знаками минус и дефис — это три разных символа!), символ номера или параграфа. Некоторые шрифты имеют только символы русского алфавита. Единственным допустимым исключением из этого правила можно

считать декоративные шрифты, в которых допустимо отсутствие строчных букв и цифр. Такие шрифты обычно применяются для выполнения акцентирующих надписей, состоящих всего из нескольких слов, так что полнота набора знаков для них не имеет особого значения.

3. **Правильность кодировки;**

Шрифт должен не только включать в себя все необходимые знаки. Важно также, чтобы все символы располагались строго на местах, определенным стандартом.

4. **Правильность оформления заголовка;**

Пожалуй, самые неприятные ошибки в шрифтах связаны с неправильным оформлением заголовка. В самом деле, шрифт вроде бы всем хорош, но пользоваться им нельзя, поскольку ни одна программа его не воспринимает.

Наиболее опасным является неправильное указание уникального идентификатора в Type 1–шрифтах (PostScript). Если два Type 1 -шрифта имеют одинаковое значение этого параметра и используются одновременно (например, в программе ATM), то растеризатор, как правило, полностью выходит из строя и требует перезагрузки. Во избежание подобных сбоев фирма Adobe производит регистрацию всех производителей шрифтов и присваивает им некоторый диапазон номеров.

Другая группа проблем связана с присвоением шрифтам неправильных имен. В результате шрифты могут неправильно регистрироваться растеризаторами и их использование окажется весьма затрудненным.

Некоторые трудности возникают в случае неправильного указания значений вертикальных размеров шрифта. Например, отсутствие информации о линиях прописных букв, строчных букв, верхних и нижних выносных элементов может привести к неправильному определению кегля шрифта при наборе текста. При этом одинаково заданный кегль для похожих шрифтов будет приводить к совершенно разным результатам.

5. **Соответствие требованиям формата;**

Наиболее неприятные ошибки возникают в том случае, если шрифт «совсем немного» не соответствует требованиям формата. Разница может заключаться буквально в одном бите шрифтового файла, но рано или поздно она скажется и возникнет ошибка. Такие ошибки трудно обнаружить, они возникают неожиданно и не всегда повторяются даже в случае точного копирования ситуации, в которой ошибка проявилась впервые. Наиболее надежным экспортом TrueType-шрифтов обладает программа Fontographer, а Type 1-шрифтов — программы FontLab (с точки зрения соответствия формату, Fontographer экспортирует Type 1 шрифты абсолютно корректно, но, в отличие от FontLab, он не поддерживает некоторые тонкости, что заметно ухудшает качество растеризации).

6. **Полнота описания метрических параметров.**

Как ни странно, но правильность определения метрических параметров шрифта (полей и ширины символов, кернинга и трекинга) оказывает на качество передачи текста большее влияние, чем качество прорисовки отдельных символов. Хорошо проработанный шрифт должен восприниматься гармонично, буквы не должны выпадать или налезать друг на друга.

Относительно поддержки кернинга рекомендуется отдавать предпочтение шрифтам, содержащим достаточно полную таблицу пар кернинга. Поддержка трекинга не имеет особого значения, так как пока только очень немногие программы могут использовать информацию о трекинге, содержащуюся в шрифте. Большинство издательских систем имеют встроенные редакторы трекинга, но, как это ни странно, не воспринимают трекинг,

определенный в шрифтах. Тем не менее, наличие правильно описанного трекинга можно считать полезным.

7. Качество контуров;

Признаками плохого качества контуров являются:

1. Нарушение гладкости в местах соединения графических примитивов.

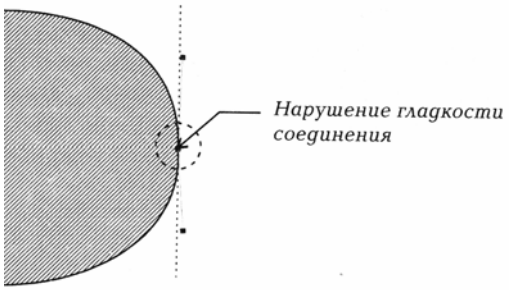


Рисунок 3.2. Пример нарушения гладкости соединения графических примитивов.

Данная ошибка может быть незаметна, однако сильно усложняет работу растеризатора.

2. Отсутствие выделенных точек экстремумов.

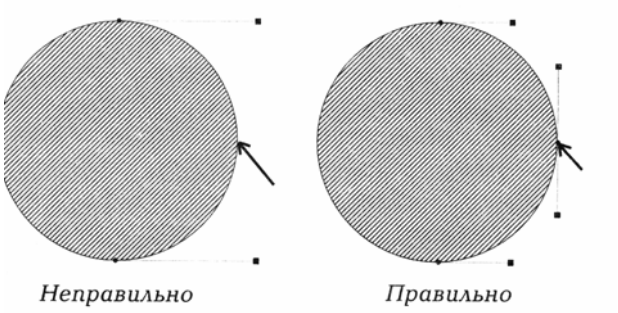


Рисунок 3.3. Отсутствие выделенных точек экстремумов в округлых элементах.

Для нормально работы растеризатора необходимо, чтобы все экстремумы контуров были выделены в качестве крайних точек кривых или векторов. В случае невыполнения этого правила растеризатор не может автоматически корректировать форму округлых элементов, из-за этого качество разметки резко уменьшается.

3. Наличие острых внутренних углов.

Рисунок 3.4. Наличие острых углов.

При описании острых углов (меньше 20°) для нормальной работы растеризатора необходимо включать короткий (1-3 единицы) вектор между примитивами, образующими угол. В противном случае форма контура вблизи примитива может сильно измениться.

4. Использование длинных кривых.

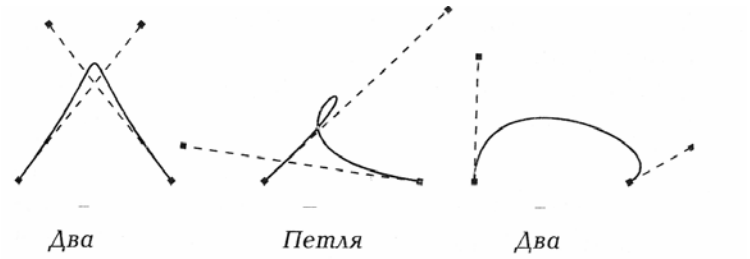


Рисунок 3.5. Примеры некорректных кривых.

При описании сложных элементов не рекомендуется использовать слишком длинные кривые. Причем некоторые кривые, имеющие две точки перегиба или слишком разные длины контрольных векторов, могут нарушить работу растеризатора.

При этом слишком большое количество кривых (большое количество точек, для описания формы кривой) тоже является нежелательным. Необходимо найти «золотую середину» между этими требованиями.

5. Нарушение вертикальности и горизонтальности штрихов.

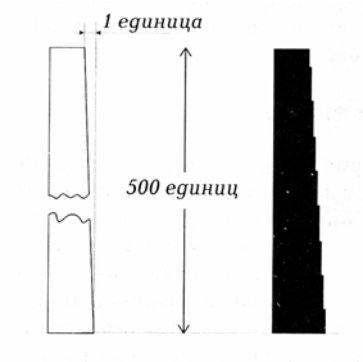


Рисунок 3.6. Нарушение вертикальности штрихов элементов символа.

Когда вектор, образующий вертикальный или горизонтальный штрих немного отстоит от строго вертикального или горизонтального направления, в некоторых случаях это может привести к появлению зубцов на линиях символа и изменить толщину штриха.

6. Нарушение размеров символов.

При смещении символов относительно базовой линии, разных размеров символов или различиях в величинах наплывов у округлых символов визуально нарушаются размеры символов, и в результате при выводе строки текста возникает лесенка внизу или вверху строки.

Особенности использования сканированных изображений

Так как Fontographer оперирует только двумя цветами – черный и белый, то отсканированные изображения должны быть битовом формате (Bitmap). Для хранения каждого пикселя битового изображения используется 1 бит. То есть все пиксели будут окрашены либо в белый, либо в черный цвет. Чтобы добавить изображение необходимо скопировать его в буфер обмена в любом графическом редакторе и вставить в окно

символа. При этом в Fontographer есть слой, предназначенный для таких изображений: Template. Важно отметить, что изображение в этом слое показывается с экранным разрешением, но, не смотря на это, качество исходного изображения и его размер является важным. Поэтому сканировать изображение необходимо с высоким разрешением не менее 300 dpi. Тогда при автоматической обводке в Fontographer результат будет достаточно точным.

Если исходное изображение было представлено в градациях серого (Grayscale), то Fontographer автоматически преобразует его в Bitmap. Но для достижения наилучшего результата желательно предварительно преобразовать его в Bitmap. Для этого можно использовать функции Adobe Photoshop.

Grayscale изображения еще могут называть 8 битовыми, так как пиксели могут иметь 256 оттенков серого цвета. Пиксели характеризуются величинами яркости в диапазоне от 0 (яркость отсутствует, черный цвет) до 255 (полная яркость, белый цвет) [О'Куин, 1998]. Когда Grayscale изображение преобразуется в битовое, необходимо решить, какие пиксели будут закрашены в черный, а какие в белый цвет.

В Adobe Photoshop существует несколько методов, которые позволяют преобразовать изображение в Bitmap. Рассмотрим подробно каждый из них.

Метод *50% Threshold* (Порог 50%) устанавливает границу яркости в диапазоне от 0 до 255 ровно посередине – 128. Все пиксели справа от границы окрашиваются в белый, а слева – в черный цвет. То есть все тона, содержащие более 50% черного цвета, становятся черными, а все тона, которые содержат менее 50% черного цвета, становятся белыми.

Именно этот же метод используется в Fontographer. Однако существует возможность точно задавать значение порога. Для этого нужно воспользоваться командой: Image (Изображение) → Adjust (Коррекция) → Threshold (Изогелия). При этом появится окно с гистограммой изображения (см. рисунок 3.10). Передвигая бегунок, можно изменять значение порога, при этом будет видно как изменяется изображение.

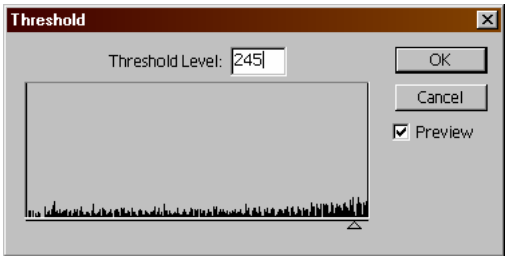


Рисунок 3.10 Окно настройки порога: Threshold.

Метод *Pattern Dither* (Регулярное распределение) «разбрасывает» пиксели, чтобы имитировать в изображении градации серого. При этом Photoshop использует заранее определенные геометрические шаблоны.

Метод *Diffusion Dither* (Случайное распределение) аналогичен Pattern Dither, но при этом пиксели распределяются случайным образом. Это метод неэффективен для разрешения, превышающего 300 dpi.

Метод *Halftone Screen* (Полутоновой растр) позволяет превращать тона серого в наборы полутоновых точек. При этом можно задать следующие установки:

- *Frequency (Линиатура)*
  Эта величина определяет размер смоделированных полутоновых точек.
- *Angle (Угол)*
  Эта опция позволяет задать угол наклона растра или угол поворота полутоновой сетки.
- *Shape (Форма)*
  Эта опция позволяет выбрать форму полутоновой точки (круг, ромб, эллипс, линия, квадрат, крест).

Полутоновой растр может быть использован только для создания специального эффекта.

Метод *Custom Pattern* (Заказной растр) позволяет использовать образец, заранее заданный с помощью команды: Edit (Редактирование) → Define Pattern (Определить

образец). Как и в случае применения метода 50% Threshold, цвета изображения преобразуются в черный и белый, но повторяющийся образец появляется только в тех областях, окрашенных в цвета, содержащие более 50% черного цвета.

Таким образом, для обработки изображений шрифтовых элементов или орнаментов наиболее подходит метод 50% Threshold или Threshold. Эти методы позволяет получать ровно окрашенные контрастные битовые изображения. Тем более, как показал опыт, при автоматической обводке в Fontographer таких изображений линии получаются более гладкими. Если исходное изображение слишком светлое и некоторые детали не видны, то можно отредактировать битовое изображение, изменяя порог с помощью метода Threshold, тем самым, увеличивая количество черных точек, то есть детальность и толщину линий.

Недостатки качества контуров при использовании функции автоматической обводки в Fontographer

При использовании функций автоматической обводки в Fontographer возникают недостатки связанные с качеством контуров. Данные ошибки могут плохо влиять на результат разметки символа, более того при определенных ошибках может быть нарушен вид символа настолько, что он станет нечитаемым. Т.о. при построении контуров стоит обратить внимание качество контуров. Основные признаки плохого качества контуров в Fontographer следующие:

1. **Наличие незамкнутых контуров.**

Для правильного отображения шрифтового знака необходимо, чтобы все контуры были замкнуты. Разрыв контура в Fontographer обозначается: . Аналогично обозначается начало контура, из-за этого может возникнуть путаница. Т.о. в замкнутом контуре может быть только одна точка с таким обозначением, иногда при неправильном отображении символа необходимо проверять начальные точки на разрыв (см. таблицу 3.6.).

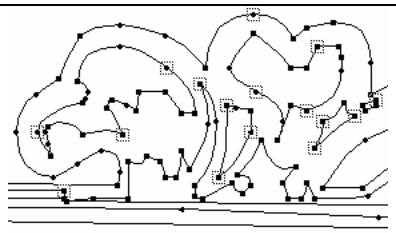
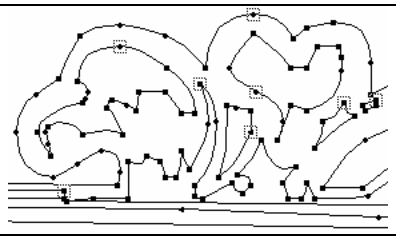
Неправильно	
	
Правильно	
	

Таблица 3.6. Наличие незамкнутых контуров.

2. **Наличие отдельных точек.**

Подобно тому, как плохо сказывается наличие разрывов при отображении символов, и их разметки наличие отдельных точек является недопустимым (см. Рисунок 3.7.). Точки используются для описания контуров, поэтому отдельные точки могут трактоваться программой как часть контура, искажая и изменяя его форму.

Неправильно	**Правильно**

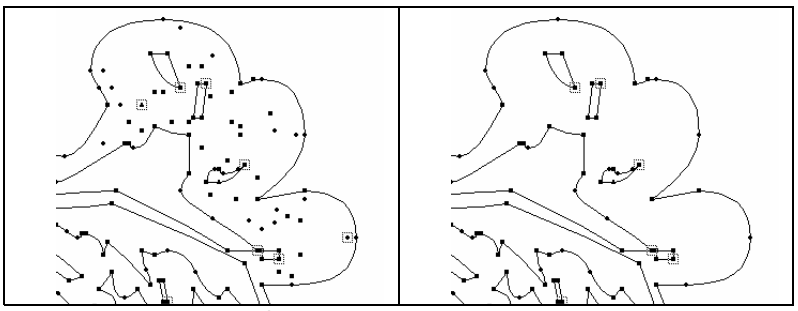


Рисунок 3.7. Наличие отдельных точек.

3. Наложение точек.

При наложении нескольких точек в Fontographer появляется следующее обозначение:  . При заполнении символа это никак не сказывается, однако эти точки являются лишними и ухудшают качество разметки (см. Рисунок 3.8.).

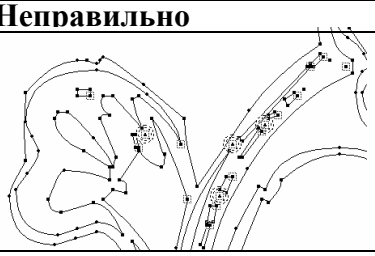
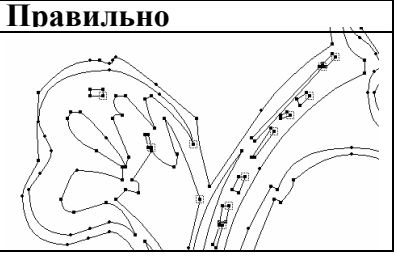
Неправильно	Правильно
	

Таблица 3.8. Наложение точек.

4. Ошибки в заполнении контуров.

Ошибки в заполнении контуров могут быть связаны с наличием разрывов или несоответствии типов внешних и внутренних контуров Так внутренний контур является внешним и при заполнении шрифтовой символ отображается некорректно (см. рисунок 3.9.)

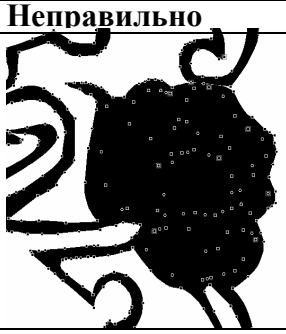
Неправильно	Правильно
	

Рисунок 3.9. Ошибки в заполнении контуров шрифтовых знаков.

5. Пересечение контуров

Пересечение контуров является недопустимым. Это может быть связано с наличием слишком длинных кривых, петель или с наложением контуров (см. рисунок 3.10.).

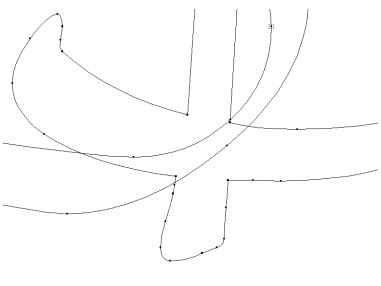
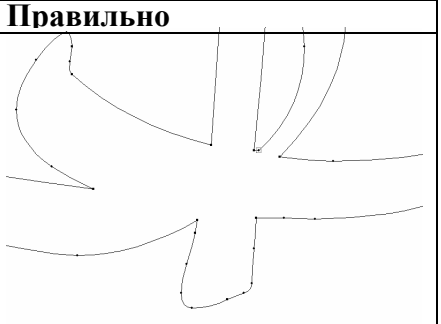
Неправильно	Правильно
	

Рисунок 3.10. Пересечение контуров.

Форматы шрифтов

В отношении компьютерного шрифта слово формат (format) используется в двух смыслах. Во-первых, формат определяется платформой, для которой шрифтовой файл создан. Например, два шрифтовых файла с одинаковыми данными для одних и тех же гарнитур могут иметь разные форматы в зависимости от того, предназначены они для платформ Apple Macintosh или Windows PC. Во-вторых, формат шрифтового файла отражает способ представления и организации собственно типографической информации. Рассмотрим три основных шрифтовых формата — *PostScript (PFM), TrueType (TTF) и OpenType (OTF)*.

Шрифты в формате **PostScript** основаны на языке описания страниц PostScript, и для их обработки и отображения требуется интерпретатор этого языка – RIP[1]. У принтеров с высоким разрешением и фотонаборных автоматов такой интерпретатор обычно встроен в устройство. Он представляет собой отдельный процессор, предназначенный для преобразования PostScript-кодов в управляющие коды устройства. Для устройств с низким разрешением, какими являются экран монитора и настольные офисные принтеры, PostScript-шрифты отображаются PostScript-интерпретатором, встроенным в операционную систему, или с помощью дополнительного приложения, которое называется Adobe Type Manager (ATM). PostScript-шрифты обычно снабжаются еще и комплектом растровых шрифтов для экранного отображения в системах без PostScript-интерпретатора.

В настоящее время PostScript-шрифты стали стандартом в издательской отрасли, поскольку они на нее ориентированы. Практически все устройства, используемые в издательствах снабжены растровыми процессорами (RIP). Естественно, такие процессоры лучше всего работают с PostScript-шрифтами.

Кроме этого PostScript-шрифты построены с помощью кривых 3-го порядка, так называемых кривых Безье (рис. 4.1). За счет этого достигается большая гладкость контуров и компактность шрифтовых файлов.

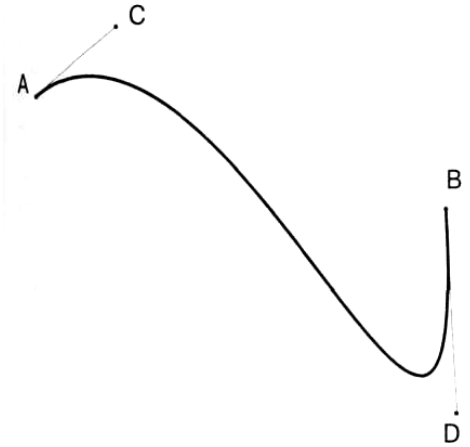


Рис. 4.1. Элементарная кривая в шрифтах PostScript (кривая Безье или кубическая парабола)

Когда мы говорим шрифты PostScript, мы обычно имеем ввиду шрифты PostScript Type 1. Это общепринятый стандарт для цифровых шрифтов (ISO 9541). Шрифт формата Type 1 — специальная форма программы PostScript и особый формат файла, который ориентирован на описание шрифта. В языке PostScript существуют и другие стандарты описания шрифтов — Type 0, Type 2, Type 3, ..., однако сейчас они почти не используются или используются для специальных целей. В последние годы язык PostScript был расширен, чтобы обеспечить поддержку шрифтовых возможностей стандартов TrueType и OpenType. Новые устройства с языком Adobe PostScript сейчас поддерживают все 3 шрифтовых стандарта.

[1] RIP – raster image processor, растровый процессор.

Формат Type1 распознается компьютерами и принтерами либо встроенными интерпретаторами языка PostScript, либо с помощью дополнительных утилит, таких как Adobe Type Manager (ATM). Технология ATM интегрирована в Microsoft Windows 2000 и XP, а так же в Macintosh OS X. Шрифты PostScript могут содержать до 220 печатаемых символов, т.е. не поддерживают стандарт Unicode.

Профессиональные шрифты PostScript Type 1 содержат специальные подсказки (хинты), которые помогают сохранить симметрию и другие эстетические параметры в процессе растеризации (рендеринга). Благодаря толково написанному растеризатору, относительно простой набор хинтов позволяет получать шрифты с приемлемым качеством. Однако разработчики шрифтов не имеют возможности полностью контролировать процесс растеризации, что не всегда позволяет достичь желаемого качества представления символов.

Шрифт PostScript состоит из нескольких файлов: Шрифт PostScript для Windows может состоять из 2-х, 3-х или 4-х файлов. Набор из 3 файлов состоит из файла с расширением PFB (Print Font Binary), который содержит информацию о контурах; файла с расширением AFM (Adobe Font Metrics) , содержащего информацию о ширинах символов и кернинге; INF файла, содержащего дополнительную информацию, которая требуется для инсталляции. В процессе инсталляции Windows генерирует PFM файл (Print Font Metrics), в основе которого лежит информация из AFM и INF файлов. Далее используется только PFB и PFM файлы. Некоторые производители генерируют PFM файлы самостоятельно и поставляют своим клиентам только два этих файла. Этого достаточно для нормального использования. Некоторые так же добавляют AFM файлы, а некоторые поставляют все 4 файла.

Шрифт PostScript для Macintosh состоит из файла-чемодана (suitcase) и принтерного файла. Если вы купили гарнитуру, а не одно начертание, то у вас может быть один чемодан на всю семью и несколько принтерных файлов, по одному для каждого начертания - Нормального (Regular), Курсивного (Italic), Жирного (Bold) и Жирного Курсивного (Bold Italic). Другой подход предполагает комплектацию каждого начертания отдельным файлом-чемоданом. В этом случае семья из 4 шрифтов будет состоять из 8 файлов. Гарнитуры могут поставляться с "совмещенными начертаниями" или с "разделенными начертаниями". Одна гарнитура может содержать до 4 начертаний, но иногда она содержит только Нормальное и Жирное, или Нормальное и Курсивное начертание. В случае "совмещенных начертаний" гарнитура будет представлена одним пунктом в шрифтовом меню и выбор начертания должен осуществляться при помощи меню Начертания (Styles), а так же с помощью кнопок B и I на панели инструментов. Если шрифты поставляются раздельными начертаниями - каждое начертание занимает свою строчку в шрифтовом меню. В этом случае не рекомендуется пользоваться кнопками B и I. Если Вы работаете в Windows 2000, XP или Mac OS X, шрифты инсталлируются с средствами операционной системы. В других версиях операционных систем нужно будет

В течение нескольких лет в конце 1980-х годов в области компьютерного шрифта и наборных процессов PostScript-шрифт являлся первым и единственным стандартом цифровых шрифтовых форматов (font format). Однако, в последствии фирмами Apple Computer и Microsoft был создан новый формат – **TrueType**, который дал возможность обеим компаниям встроить отображения шрифта в свои операционные системы, не будучи ничем обязанными компании Adobe.

Хотя предполагалось, что шрифты TrueType совместимы с PostScript-интерпретаторами, на фотонаборных автоматах возникали проблемы с выводом шрифтов этого формата. По этой причине PostScript-шрифты остались форматом, который предпочитают профессиональные издатели. Эти проблемы не утратили своей остроты, хотя популярность шрифтов TrueType в ОС Windows и новые коммерческие взаимоотношения компаний Adobe и Microsoft привели к более устойчивой работе PostScript-устройств.

В формате TrueType нашли свое воплощение несколько улучшений по сравнению с PostScript-шрифтами. TrueType-шрифты обычно распространяется без создаваемых вручную экранных (растровых) вариантов. Экранное представление шрифта генерируется

непосредственно из контура знака. Кроме этого формат TrueType допускает размещение более широкого комплекта знаков. В нем найдется место для альтернативных форм знаков и возможность контекстной замены знаков (contextual character switching). Это значит, что при определенных условиях один знак автоматически заменяется другим.

TrueType-шрифты в отличии от PostScript построены на кривых 2-го порядка. Каждый участок контура задается двумя точками – границами и направлением линии на каждой из границ (рис. 4.2).

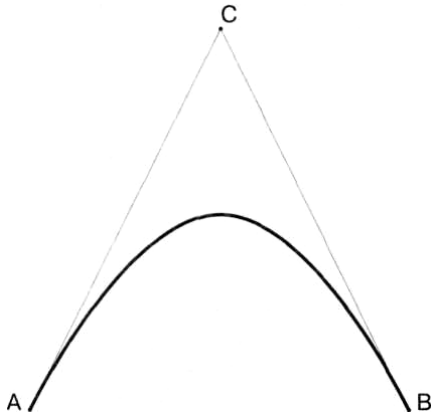


Рис. 4.2. Элементарная кривая в шрифтах TrueType (парабола второго порядка).

Т.о. для описания контура символа TrueType-шрифта требуется большее количество точек, по сравнению с PostScript (таблица 4.1), поэтому они объемнее. За счет большего числа степеней свободы PostScript-линия не имеет изломов в точках сопряжения фрагментов, тогда как для TrueType больший или меньший перелом линии в точке стыковки двух сегментов является почти неизбежным злом. Иначе говоря, символы PostScript-шрифта являются более гладкими, чем TrueType.

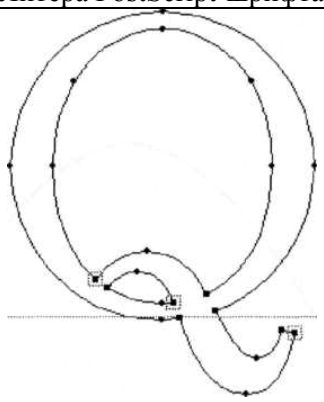
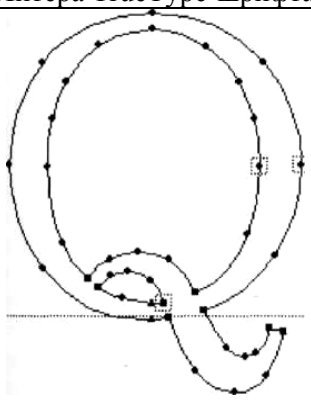
Литера PostScript-шрифта	Литера TrueType-шрифта
	

Таблица 4.1. Литеры PostScript и TrueType шрифтов.

Технология TrueType была разработана компанией Apple и сейчас используется компаниями Apple и Microsoft в своих операционных системах.

Шрифты TrueType могут содержать до 65000 символов, расположенных в порядке, определенном стандартом Unicode. В действительности, не все шрифты содержат расширенные наборы символов, большинство европейских изготовителей ограничиваются стандартной комплектацией западноевропейской кодировки (т.е. Latin 1). Исключение составляют компании Microsoft, которая большую часть европейских шрифтов выпускает в кодировке WGL4 и компания ParaType, выпускающая так называемые Multilingual (многоязычные) шрифты, которые кроме Западноевропейской, включают в себя Центральноевропейскую, Кириллическую, Балтийскую и Турецкую кодировки.

Формат TrueType использует специальную технику обеспечения качества представления шрифта при выводе на устройствах с низкой разрешающей способностью,

таких как экран компьютерного монитора. Он использует развитый набор инструкций, позволяющий добиться того же качества, что и у растровых шрифтов. Однако, с другой стороны, процесс инструктирования (хинтовки) шрифта оказывается настолько трудоемким, что многие производители не проводят ее в полном объеме и в результате средний шрифт в формате TrueType выглядит на экране менее качественно, чем аналогичный шрифт формата PostScript.

Шрифт в формате TrueType - это один файл. В системе Windows он имеет расширение TTF, а в Mac OS это файл-чемодан (suitcase) с ресурсом SFNT. В Mac OS X встроен шрифтовой процессор нового поколения, который кроме шрифтов с ресурсом SFNT, поддерживает и файлы TTF, созданные для Windows. Так что файл с расширением TTF можно использовать на обеих операционных платформах. Более подробная информация об истории TrueType, различных вариациях формата, тонкостях технологии и др. доступна на сайтах Microsoft Typography и TrueType Typography.

Формат **OpenType** является гибридным, он создан компаниями Adobe и Microsoft и сглаживает различия двух форматов, позволяя им сосуществовать в одном шрифтовом файле. Он также дает возможность один и тот же шрифтовой файл использовать в обеих операционных системах Macintosh и Windows.

Проще говоря, шрифт формата OpenType — это шрифт TrueType с «кармашком» для PostScript-данных. Шрифт OpenType может содержать данные формата TrueType, данные формата PostScript или (теоретически) обоих форматов. Таким образом, существует потенциальная возможность оптимальным образом объединить лучшие стороны обоих форматов. Операционная система сама сортирует данные шрифта OpenType и выбирает только те из них, которые ее устраивают.

Проблема шрифтовых файлов форматов OpenType и TrueType состоит в том, что будучи просто пользователем шрифта трудно узнать, что у них внутри. Формат PostScript-шрифтов обычно содержит только стандартный комплект знаков со стандартными параметрами. А формат TrueType, и в еще большей степени формат OpenType, предлагает широкий набор дополнительных параметров, которые могут включаться, а могут и не включаться в каждый конкретный шрифтовой файл. Например, формат OpenType может содержать от 256 до 65 536 знаков. И не существует способа узнать об этом, если только параметры шрифта не отражены в каком-либо сопроводительном документе.

Подобно формату TrueType, каждый шрифт OpenType использует один файл для хранения информации о контурах, метриках и служебных данных. Один и тот же файл можно устанавливать в операционных системах Windows и Macintosh.

Шрифты OpenType построены на основе мультибайтной кодировки Unicode, которая охватывает практически все мировые языки. Это важное преимущество формата TrueType теперь применимо к данным PostScript.

OpenType может иметь "цифровую подпись" производителя. Эта подпись позволяет операционной системе определить происхождение шрифта и выяснить, был ли он модифицирован.

Так же как TrueType, шрифты OpenType имеет так называемый "параметр уровня встраивания" ("embedding flag"). Этот механизм определяет какие ограничения накладываются на встраивание шрифта в документ для его распространения с этим документом.

Шрифты OpenType используют более эффективные методы сжатия данных: Compact Font Format (CFF) фирмы Adobe для данных PostScript и MicroType Express фирмы Agfa для данных TrueType. Благодаря сжатию файлы со шрифтами занимают меньше места на диске и быстрее пересылаются по сети.

Из-за ограничений предыдущих шрифтовых технологий в один шрифт часто невозможно было поместить все необходимые знаки. Поэтому в дополнение к основным шрифтам строили дополнительные варианты шрифтов в различных стандартизованных или чаще случайно придуманных кодировках. Использование шрифтов, для которых были, например, разработаны минускульные цифры и капитель, было чрезвычайно неудобно. Формат OpenType за счет поддержки Юникода и механизма включения альтернативных вариантов знаков снял эти проблемы. В один шрифтовой файл можно

поместить все национальные кодировки, экспертные комплекты символов и практически любое количество дополнительных и альтернативных знаков.

Одно из главных преимуществ новой технологии — поддержка расширенных типографских возможностей (т.н. OpenType features). Помимо собственно знаков шрифт OpenType может содержать правила использования этих знаков — позиционирование и подстановку одних знаков вместо других при определенных обстоятельствах.

Главное в реализации расширенной типографики OpenType лежит в разделении числового кода знака (character) и его графемы (glyph). Знак — это кодированная единица, упорядоченная в соответствии со стандартом Unicode, представляющая минимальную семантическую единицу языка, например букву. Глиф — это графический образ знака. Один знак может соответствовать нескольким глифам; строчная "a", капительная "a" и альтернативный вариант строчной "a" с росчерком являются одним и тем же знаком, но в то же время это три разных глифа (графемы). С другой стороны, один глиф также может соответствовать комбинации нескольких знаков, например лигатура "ffi", являясь единой графемой, соответствует последовательности трех знаков: f, f и i. Т.о. для программы проверки орфографии слово suffix будет состоять из 6 знаков, а графический процессор выдаст на экран 4 глифа.

Для любого знака по умолчанию определен базовый глиф и порядок размещения в тексте. Применение дополнительных правил к одному или нескольким знакам может изменить их взаимное расположение или заменить базовые глифы альтернативными. К примеру, применение правила КАПИТЕЛЬ к символу "a" заменит обычный знак "a" на капительный аналог "ᴀ". Для того чтобы воспользоваться новыми возможностями шрифтов необходимо, чтобы прикладные программы поддерживали эти возможности и имели соответствующий пользовательский интерфейс, однако это не означает, что шрифты OpenType не будут работать в старых программах. Программы, не поддерживающие Unicode и расширенные типографские функции OpenType, так же как и прежде смогут работать с основным набором глифов в OpenType шрифте, который аналогичен набору глифов шрифтов PostScript Type 1.

Adobe InDesign и Adobe Photoshop стали первыми приложениями, которые предоставляют поддержку типографических правил OpenType. Другие приложения фирмы Adobe также перейдут на эту технологию в ближайшем будущем. В InDesign и других программах, поддерживающих OpenType, можно включить типографические правила, которые будут определять подстановку глифов в тексте. Например, в InDesign к большинству из этих правил, таким как использование лигатур, капители, минускульных цифр и т.п., предоставлен прямой доступ через всплывающее меню на палитре Character. Кроме того, любой альтернативный глиф может быть вручную вставлен в документ через команду Insert Glyph.

К сожалению, не смотря на благие намерения унифицировать PostScript и TrueType в одном формате, OpenType с данными PostScript (OT/PS) и OpenType с данными TrueType (OT/TT) работают по-разному в различных системах и приложениях.

Кроме того, новые возможности формата могут поддерживаться не в полном объеме. Существуют 3 уровня поддержки:

Базовая поддержка — шрифты OpenType работают как обычные шрифты в кодировке Western.

Многоязычная поддержка — шрифты OpenType могут использоваться в соответствии с кодировкой Unicode. Замечание: Будьте внимательны, даже если в операционную систему встроена поддержка кодировки Unicode, это еще не означает, что все приложения автоматически пользуются средствами этой поддержки, и, наоборот, в ОС с базовой поддержкой, некоторые приложения могут работать со шрифтами напрямую, и иметь доступ к знакам за пределами однобайтного диапазона.

Полная поддержка — Unicode + поддержка расширенной типографики: шрифты OpenType могут использоваться в соответствии со всеми своими возможностями замены и позиционирования глифов.

Windows 95, 98, ME работает со шрифтами OT/TT так же как и со стандартными TrueType шрифтами, с ограниченной поддержкой Unicode. OT/PS не поддерживается

возможностями системы, для его поддержки требуется установить ATM 4.1.2 или более новые версии. В Windows 2000, XP встроена полная многоязычная поддержка для форматов OT/TT и OT/PS (ATM не требуется). Встроена также ограниченная поддержка расширенной типографики OpenType — автоматическая замена и позиционирование глифов для некоторых сложных письменностей (Арабской, Деванагари..) Mac OS Classic (7.x - 9.x) не поддерживает формат OT/TT и имеет базовый уровень поддержки для OT/PS при установке ATM 4.6.2 или более новой версии. В Mac OS X встроена многоязычная поддержка для обоих форматов OT/TT и OT/PS. ATM не требуется.

Полная поддержка возможностей форматов OT/TT и OT/PS версий сначала появилась в программах **Adobe InDesign** и **Adobe Photoshop 7.0**. Сейчас список программ Adobe расширился до пакета Adobe Creative Suite. Кроме вышеперечисленных программ, туда входит Adobe Illustrator CS и ImageReady CS. Многоязычная поддержка для OT/PS и OT/TT многоязычная поддержка реализована в CorelDRAW 10 и 11 версии, а так же в MS Word 2003 для OT/TT в MS Word 2000, 2002; Adobe Illustrator 10 для Mac и PC.

Другие популярные приложения, такие как Freehand, QuarkXPress все еще работают с OpenType шрифтами как с обычными шрифтами в кодировке Western.

Кодировка шрифта

Вся информация в компьютере, в том числе и текстовая, хранится в виде двоичных чисел (кодов).

Сейчас для кодировки шрифтов используется международный стандарт Unicode. Он расширяет кодовую схему, включая знаковые комплекты для нелатинских алфавитов. Большинство шрифтовых файлов стандарта Unicode двухбайтовые и могут содержать до 65000 знаков. Стандарт Unicode обеспечивает межплатформенную совместимость шрифтов, эту кодовую таблицу поддерживают Macintosh OS X и Windows NT 4, Windows 2000 и Windows XP.

У первых шрифтовых файлов имелись ячейки только для 256 знаков, и данный комплект знаков остается стандартом для большинства шрифтовых файлов. Иногда, даже если формат шрифта поддерживает 65000 знаков, он включает только 256 стандартных.

До Unicode единственным межплатформенным стандартом кодировки был американский стандартный код для обмена информацией (сокращенно ASCII или просто ASC-код), разработанный для телетайпа и других подобных систем связи. Код ASCII первоначально являлся семибитным и включал в себя символы с кодами от 32 до 128 (кодам от 0 до 31 соответствовали неотображаемые, служебные символы, типа — «звонок», 10 — «перевод строки», 13 — «возврат каретки»). Для отображения символов национальных алфавитов, символов псевдографики и некоторых математических символов таблица ASCII-кода была расширена до 16 бит, получившийся в результате код стали называть «расширенным ASCII-кодом».

До того как Macintosh и Windows стали поддерживать стандарт Unicode, они использовали разные кодовые схемы (таблицы). Таблицы совпадали в основной части комплекта ASCII, но различались в знаках, имеющих коды после 128, так называемые знаки старших разрядов (high-bit). Результатом стало то, что документы, кодировавшиеся на одной компьютерной платформе, на другой очень часто отображались некорректно.

И дело не только в том, что операционные системы до стандарта Unicode использовали разные таблицы кодирования, а в том, что они применяли разные подмножества комплекта Latin 1 в качестве своих стандартных комплектов знаков.

Комплект системы Macintosh (и кодовая таблица) называется MacRoman; а комплект системы для Windows (и кодовая таблица) называется Win ANSI. Хотя распространители шрифта могут продавать одноименные шрифтовые файлы для обеих платформ, пользователи системы Macintosh получат в шрифтовом файле одну группу знаков, а пользователи системы Windows — другую. Определенные знаки в комплекте MacRoman

заимствованы из шрифтового файла Symbol. Когда вы работаете на компьютере Macintosh, то кажется, что эти знаки являются частью каждого шрифта.

Понятие о формате и шрифтовой машине

Любой цифровой шрифт, как это сразу становится понятно из названия, представляет собой описание входящих в него символов, метрических и других параметров, определяющих особенности шрифта в цифровой форме. Форматом представления цифрового шрифта называется способ (стандарт) представления цифровой информации, образующей шрифт. Обычно он представляет собой один или несколько файлов, с которыми можно поступать так же, как и с любыми другими файлами: копировать, удалять, переименовывать и т.д.

Шрифт, представленный в определенном формате, можно использовать в любых программных и аппаратных средствах, которые могут воспринимать закодированную в формате информацию. Таким образом, создание определенного формата представления шрифтов не является достаточным для их использования. Необходимо иметь еще два компонента: средства преобразования информации в заданный формат и средства воспроизведения шрифтов, представленных в этом формате. Если средства кодирования используются в основном производителями шрифтов, то средства воспроизведения необходимы в первую очередь пользователям цифровых шрифтов. Совокупность определенного формата представления шрифтов и средств воспроизведения шрифтов, заданных в этом формате, мы будем называть *шрифтовой машиной* (рис. 4.3).

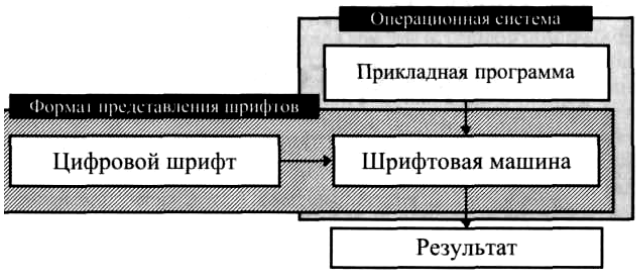


Рис. 4.3. Схема реализации шрифтовой машины

Очевидно, что без средств воспроизведения любой шрифтовой формат имеет только теоретический интерес, поскольку невозможно оценить качество воспроизведения шрифта и скорость работы. Поэтому бессмысленно говорить о преимуществах того или иного формата, оценивать можно только работу шрифтовой машины.

Структура шрифтового формата

Как и любой шрифт, имеющий определенный набор параметров, повторяющихся от шрифта к шрифту, любой шрифтовой формат имеет некоторые обязательные части. Перечислим их с краткими пояснениями.

Область заголовка. В этой части располагается следующая информация:

7. Информация о различных вариантах наименования шрифта (рабочее имя шрифта, имя гарнитуры и начертания, полное имя шрифта, имена и индексы, под которыми шрифт воспринимают прикладные программы).
8. Информация о создателях шрифта (знак принадлежности прав, ссылка на автора исходного рисунка шрифта, информация о торговой марке, история создания шрифта).
9. Регистрационная информация, предназначенная для автоматической классификации шрифта и обеспечения подстановки шрифтов. Обычно в этой области расположены описания насыщенности, угла наклона и пропорциональности шрифта, а также код шрифта по одной из систем описания шрифтов.

10. Статистическая информация о шрифте (минимальный охватывающий прямоугольник[2], количество символов и др.).

Область описания метрических параметров. В этой части описываются все измерения символов. Обычно к ним относят информацию о ширине символов, минимальные охватывающие прямоугольники для всех символов, информацию о кернинге и трекинге шрифта. В некоторых форматах (например, в формате PostScript) информация о трекинге и кернинге сохраняется в отдельном файле.

Область описания общих элементов. Некоторые символы имеют одинаковые элементы. Для сокращения объема шрифтового файла и для того, чтобы гарантировать действительную одинаковость этих элементов, они отделяются от символов. Символы содержат только ссылки на такие элементы. То же самое относится и к некоторым средствам разметки, общим для нескольких символов.

Область описания системы кодирования. В этой области располагаются кодовые таблицы, относящиеся к шрифту.

Область описания разметки символов. В этой области находится информация о разметке символов, необходимая для их качественного воспроизведения.

Область описания символов. Это — основная часть шрифтового файла. В ней находится описание самих символов. Для формирования контуров символов могут использоваться различные математические и логические методы. Обычно метод описания контуров и определяет эффективность работы, а также особенности растеризации шрифтов определенного формата.

Управление растеризацией символов

Как уже говорилось, фундаментальной особенностью контурных шрифтов является отделение информации о форме символов от процесса их воспроизведения на растровом выводном устройстве. Если контуры символов шрифта можно описывать самыми разными способами, то задача воспроизведения, в конечном итоге, сводится к активизации некоторых точек (высвечиванию на экране дисплея или заполнении краской при печати на принтере).

Алгоритм растеризации

Итак, при воспроизведении каждого символа на растровом устройстве (например, на лазерном принтере) необходимо решить две задачи:

1) масштабировать (уменьшить или увеличить) контур символа до необходимого размера. Например, при печати текста 10 кеглем на лазерном принтере с разрешением 300 точек на дюйм (12 точек на миллиметр) необходимо, чтобы контур символа H имел примерно 28 точек в высоту;
2) активизировать все точки, попавшие во внутренние области этого контура, заполнить контур.

---

[2] Минимальный охватывающий прямоугольник шрифта — это прямоугольник минимального размера, в который целиком помещаются все символы шрифта помещаются все символы шрифта (кегельная площадка).

Проблемы растеризации

В ходе решения этих простых, на первый взгляд, задач возникает немало проблем, связанных с масштабированием и заполнением контуров. Перечислим некоторые из них.

Нарушение пропорций символа. При воспроизведении символов на устройствах с малой разрешающей способностью (300 точек на дюйм и меньше), особенно при выводе текста небольшим кеглем (12 и меньше), сильно сказываются ошибки масштабирования. Масштабирование происходит в абсолютных координатах относительно некоторой произвольной точки и всегда приводит к получению целочисленного результата. При этом возникает проблема округления нецелых результатов. Например, если координаты некоторого элемента символа в системе координат описания контура равны (200; 100), то при уменьшении размера контура в 3 раза они трансформируются в (66. 666666; 33. 333333). Поскольку нам нужны целые значения, они превратятся в (67; 33), то есть значение горизонтальной координаты немного (на треть точки) увеличится, а горизонтальной — на столько же уменьшится. Если при этом специально не учитывать особенности формы символа, то он может сильно исказиться и даже стать нечитаемым. На рис. 4.4 приведен пример подобного масштабирования символа Н:

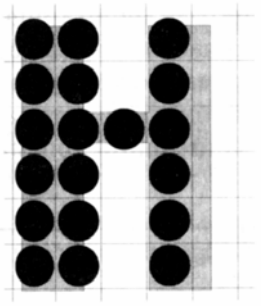


Рис. 4.4. Пример масштабирования символа Н

Нарушение симметричности некоторых символов. Прежде всего этот дефект относится к символам, обладающим симметрией, таким, как А, Ж, М, О, Т, Ф, Ш, и некоторым другим. Нарушение симметричности таких символов (например, возникновение разного расстояния между вертикальными штрихами буквы Ш) резко искажает их форму и затрудняет чтение текста.

Нарушение единства символов. Применяя некоторые приемы, мы можем избавиться от ошибок округления применительно к одному символу. Но при этом мы рискуем потерять единство символов в шрифте. Например, если в символе Н мы будем округлять толщину вертикальных штрихов в меньшую сторону, а в символе Ш — в большую, то некоторые слова станут трудно воспринимаемыми. Кроме того, при таком подходе нарушается ритмичность шрифта (характерный случай — разное округление расстояния между вертикальными штрихами в символах Ш и Щ).

Другой пример — масштабирование положения горизонтальных линий (например, средних линий символов в, е, ж, з, к) и величины оптических наплывов у округлых букв (таких, как а, б, е, з, о, с). В первом случае может возникнуть неприятный разнобой в некоторых словах, а во втором — искажение базовой линии текста и скачки букв в вертикальном направлении.

Смыкание штрихов. В некоторых случаях некачественного масштабирования штрихи и другие элементы символов смыкаются между собой. Наиболее часто это происходит с вертикальными штрихами в узких шрифтах. Ошибочное соединение штрихов (рис. 4.5) нарушает графему такой буквы, и человек теряет способность к ее распознаванию.

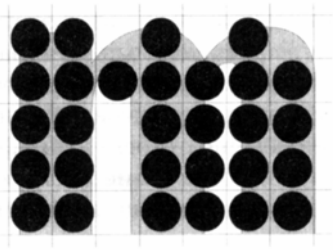


Рис. 4.5. Пример смыкания элементов символа.

Выпадение точек. Если не обращать внимания на прохождение линий при округлении координат опорных точек контура, то часто возникают ситуации, в которых программа заполнения масштабированного контура не может определить, какие именно растровые точки необходимо активизировать. Как правило, эта проблема возникает при заполнении тонких наклонных элементов (рис. 4.6).

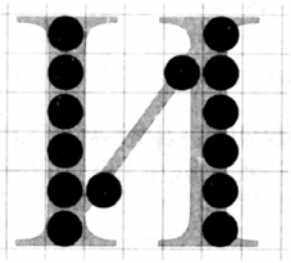


Рис. 4.6. Пример выпадения точек при заполнении контура.

Нарушение формы округлых букв. Этот дефект не так резко, как другие, влияет на удобство восприятия текста. Он «только» искажает форму символов, имеющих большие округлые элементы, например В, О, 3, Р, С, а, б и др. Вопрос о заполнении таких элементов можно решать разными способами, но лишь некоторые из них позволяют получить действительно качественное изображение буквы, а остальные приводят к подобным ошибкам, приведенным на рис. 4.7.

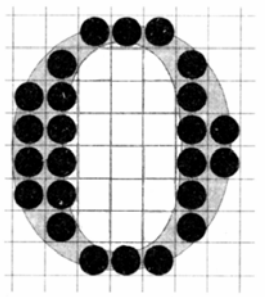


Рис. 4.7. Нарушение формы округлых букв.

Теперь, когда мы выявили некоторые проблемы, связанные с растеризацией символов, рассмотрим методы устранения этих проблем. Для этого прежде всего введем понятие разметки шрифта. *Разметкой* мы будем называть описание символов, их элементов и шрифта в целом, призванное улучшить качество растеризации символов. Иногда разметку называют хинтингом (от англ. *hint* — подсказка), но этот термин обычно относят к шрифтам в формате Type 1 (для TrueType шрифтов используют понятие инструкций), поэтому мы считаем необходимым ввести новый, более общий, термин.

Методы разметки символов

Существует два основных метода разметки символов контурных шрифтов: декларативный и программируемый. Первый применяется в формате Adobe Type 1, а второй — в TrueType шрифтах.

Декларативный метод разметки основан на описании особенностей символа при помощи их декларирования отдельно от описания контура (рис. 4.8). То есть описание символа при этом включает в себя две части: математическое описание контура символа и декларирование его особенностей:

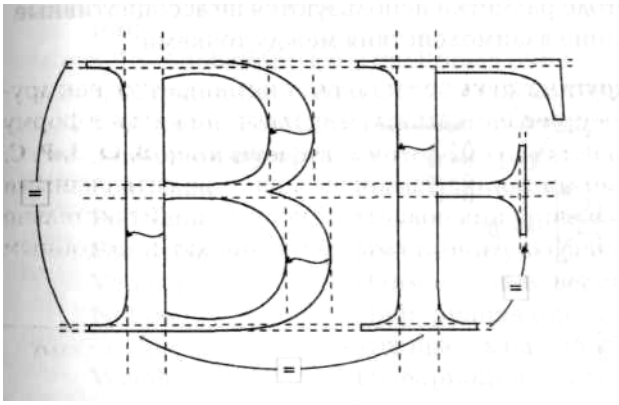


Рис. 4.8. Элементы разметки при декларативном методе.

Задачу связывания этих частей и построения правильных ассоциаций решает программа растеризации. Именно она анализирует форму символа, связывает ее с заданной разметкой и принимает решения об изменении контура в ходе его масштабирования и заполнения. Таким образом в шрифтовой машине, разметка символов в которой производится декларативным методом, основную часть работы по улучшению формы символов выполняет растеризатор. Обычно он представляет собой довольно сложную программу, содержащую множество высокоэффективных алгоритмов (ведь символы приходится воспроизводить очень быстро) и элементы искусственного интеллекта.

Огромное преимущество декларативной разметки — простота построения шрифтов. Так как производителей шрифтов намного больше, чем производителей растеризаторов, применение этого метода приводит к более быстрому появлению новых гарнитур.

Программируемый метод разметки основан на точном определении в шрифте всех действий, которые должен выполнять растеризатор. На долю растеризатора при этом остаются только интерпретация команд разметки и как можно более быстрое их выполнение. Растеризатор оказывается более простым, компактным и быстрым, но это происходит за счет резкого усложнения шрифтов и увеличения их в объеме. Программа разметки может быть очень сложной, имеющей циклы, условные переходы, описания переменных и массивов (рис. 4.8). Языки программирования разметки обычно имеют много команд модификации контуров символов, причем среди них есть как команды, работающие на этапе масштабирования контура, так и на этапе его заполнения.

В программируемом методе разметки используются не ассоциативные декларации, а точное указание взаимодействия между точками:

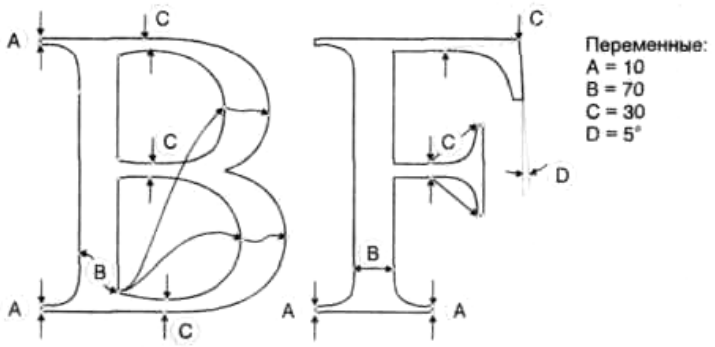


Рис. 4.9. Элементы разметки при программируемом методе.

Потенциально программируемая разметка может обеспечить намного лучшее качество, чем декларативная, но создание высококачественных шрифтов, использующих программы разметки, оказывается настолько трудоемким, что таких шрифтов появляется довольно мало. Обычно производители таких шрифтов (а это все TrueType-шрифты) применяют специальные системы, автоматически формирующие программы разметки символов и шрифта. Такой путь обычно приводит к невысокому качеству растеризации шрифтов, так что потенциальное преимущество программируемой разметки теряется.

Общая структура шрифта в формате PostScript

Любой Type 1-шрифт состоит из двух основных частей: открытой и закрытой (зашифрованной):

Открытая часть:

- Обозначение шрифта
- Заголовок шрифта
- Кодовая таблица шрифта
- Уникальный идентификатор шрифта

Закрытая часть:

- Область глобальной разметки
- Область глобальных подпрограмм
- Область подпрограмм разметки и контурных подпрограмм
- Область описаний символов

Открытая часть. В открытой части Type 1-шрифта содержится информация, доступная для любого текстового редактора. Эта часть может быть изменена при условии, что закрытая часть останется нетронутой. В открытой части можно выделить 4 области.

Обозначение шрифта показывает, что файл является именно шрифтом:

```
%! PS-AdobeFont-1.0: TimeRoman 001.1
%%CreationDate: Wed Oct 20 17:08:26 1993
%%Creator: FontLab(c) for Windows v2.5
```

Заголовок шрифта, в котором хранится следующая информация:

- Регистрационное имя рифта FontName;
- Полное имя шрифта FullName;
- Имя гарнитуры, в которую входит шрифт FamilyName;
- Наименование версии рифта Version;
- Информация о создателях шрифта и об авторских правах на шрифт Notice;
- Информация о насыщенности шрифта Weight;
- Угол наклона символов шрифта в градусах против часовой стрелки ItalicAngle;
- Информация о том, является ли шрифт моноширинным IsFixedPitch;
- Положение линии подчеркивания UnderlinePosition;
- Толщина линии подчеркивания UnderlineThikness;
- Вид шрифта PaintType: 0 — сплошной (заполняемый); 1 — контурный. Все Type 1-шрифты являются сплошными;
- Тип шрифта Font Type: 0 - Type 0; 1 - Type 1; 3 - Type 3;
- Стандартная матрица трансформирования символов FontMatrix. Ее более подробное описание приведено в разделе «Описание символов».
- Минимальный прямоугольник, охватывающий все символы шрифта. FontBBox

Приведем пример типичного заголовка Type 1-шрифта:

```
/Fontlnfo 9 dict dup begin
/FullName (Times New Roman) readonly def
/FamilyName (Times) readonly def
/version (001. 1) readonly def
/Weight (Normal) readonly def
/Notice ((c) Copyright Monotype, 1990) readonly def
/ItalicAngle 0 def
```

```
/isFixed Pitch false def
/UnderlinePosition -100 def
/UnderlineThickness 50 def end readonly def
/FontName /TimesNewRoman def
/PaintType 0 def
/FontType 1 def
/FontMatrix [ 0. 001 0 0 0. 001 0 0 ] readonly def
/FontBBox { -63 -231 1148 882 } readonly def
```

Кодовая таблица шрифта определяет связь между именами и кодами символов. В Type 1-шрифтах все символы имеют уникальные имена, которые однозначно их идентифицируют. Кодовая таблица позволяет установить некоторое соответствие между кодами символов, с которыми работают программы, использующие шрифт, и именами символов. Поскольку кодовая таблица находится в открытой части шрифта, ее можно изменять, тем самым меняя кодировку, в которой работает шрифт. Кодовая таблица представляет собой набор пар вида: <код> <имя>. Код — это 8-разрядный код символа (от 0 до 255), а имя — это строка, не имеющая пробелов. В формате Type 1 в именах символов различаются прописные и строчные буквы.

Хотя кодовая таблица Type 1 -шрифтов позволяет использовать только 8-битные значения для кодов, то есть с ее помощью можно определить не более 256 разных символов, Type 1 -шрифт может содержать любое их количество. В кодовой таблице символы, не попадающие в 256-знаковую область никак не отражаются, но они присутствуют в шрифте под своими именами, отличающимися от других. Изменяя кодовую таблицу (напомним, что это можно делать, не затрагивая остальной шрифт), можно получить доступ ко всем символам.

Уникальный идентификатор шрифта - 24-разрядное число (от 0 до 16777215). Идентификатор должен определять один и только один шрифт. В случае использования двух шрифтов с одинаковыми идентификаторами возможно возникновение серьезных ошибок. Идентификаторы в диапазоне 4000000 — 4999999 могут использоваться для внутренних целей любой организации. Для других шрифтов (например, ориентированных на продажу) необходима регистрация идентификаторов в фирме Adobe.

Закрытая часть — это основная часть любого Type 1-шрифта, в которой содержатся описания символов и информация об их разметке. Закрытая часть шрифта определяется его создателями, шифруется при помощи особого алгоритма и не может быть изменена после загрузки шрифта в принтер. Вообще говоря, шифрование этой части потеряло всякий смысл после того, как в 1990 году был опубликован алгоритм дешифровки, но для обеспечения совместимости со старыми устройствами шрифты продолжают зашифровывать. Кроме того, шифрование закрытой части Type 1-шрифтов немного ограничивает возможности тех, кто нелегально пытается их изменить и выдать за свои. Теоретически, сам акт дешифровки может в некоторых случаях считаться нарушением авторских прав.

В закрытой части есть области, зашифрованные дважды, — это описания подпрограмм и символов. При этом для дополнительной экономии места применяется специальный метод кодирования числовых значений и команд.

Зашифрованная часть начинается после слова eexec и, так же, как и открытая, состоит из четырех областей.

Область глобальной разметки, в которой содержатся описания параметров шрифта, которые используются для улучшения качества растеризации. Вот краткое описание некоторых из них.

BlueValues - массив пар чисел (до 7 пар в возрастающем порядке), определяющих зоны выравнивания сверху (кроме первой пары, которая определяет зону выравнивания базовой линии снизу).

OtherBlues - массив пар чисел (до 5 пар в возрастающем порядке), определяющих зоны выравнивания снизу, например для нижних выносных элементов.

BlueShift - определяет величину оптического наплыва (в точках выводного устройства), начиная с которой отключается его подавление.

StdHW и StdVW определяют наиболее распространенные толщины горизонтальных и вертикальных штрихов. В том случае, когда после масштабирования контура толщины штрихов мало отличаются от стандартных значений, используются эти значения, что улучшает внешний вид символов и скрадывает ошибки построения контуров.

Приведем пример описания этих значений в шрифте.

```
/BlueValues [ -16 0 488 504 712 728 752 752 ] ND
/OtherBlues [ -224 -221 ] ND
/BlueShift 7 def
/StdHW [ 48 ] ND
```

Область глобальных подпрограмм содержит несколько подпрограмм, написанных на языке PostScript. Обычно они используются для реализации наиболее сложных методов разметки.

Область подпрограмм разметки и контурных подпрограмм. Язык описания Type 1-шрифтов, как и PostScript, имеет встроенные возможности для структурной организации программы, реализованные в виде команд вызова глобальных (PostScript) и локальных (написанных на языке Type 1) подпрограмм. Локальные подпрограммы обычно применяются для организации сложной разметки символов, например для смены хинтов, и для описания повторяющихся элементов символов.

Область описания символов — основная область Type 1-шрифта, определяющая изображения всех символов шрифта. Описание каждого символа включает его имя, ширину левого поля, ширину символа (расстояние от линии левого поля до линии правого поля), описания разметки и контура.