СПРАВОЧНИК ПО VGCAST

Концепция 2

Настройки программы 2

Журнал событий 2

Протокол обмена и кодировка текстовых данных 2

Именование объектов 3

Выражения 3

Арифметические функции 3
Логические функции 4
Функции слоев и шаблонов 4
Функции работы со строками 5
Функции работы с временем и датой 6
Функции работы с файлами в формате XML 8
Примеры использования выражений: 8

Переменные 8

Команды управления 9

Команды работы с шаблонами 10

Клонирование шаблонов 10

Цикл loop 11

Условные операторы и циклы 12

Примеры 12

Управление прозрачностью объектов 14

Подстановки 14

Работа с опорными кадрами 15

Ключевые кадры объекта внутри слоя 16
Ключевые кадры масштабирования объекта 16
Ключевые кадры параметров слоя 17
Ключевые кадры параметров шаблона 17

Расположение слоев на экране 17

Расположение шаблонов на экране 18

Команды слоев 19

Работа со звуком 20

Работа с объектами 3D 20

Информационные команды 22

Сервисные команды 23

Изменение атрибутов объектов 23

Общие атрибуты 23

Специальный режим масштабирования текста....23
Атрибуты текстовых слоев....24

Описание шаблона....25
Общие операторы описания слоя....25
Операторы, специфичные для конкретных слоев....26
Примеры описания слоев в шаблоне....28
Операторы описания текстовых слоев....29
Примеры описания текстовых слоев в шаблоне....31
Описание эффектов....31

Пример организации связи....32

Концепция

Исполняющая система **vgcast** (движок) позволяет в реальном времени отображать графическую информацию для оформления эфира. Общая структура отображаемых данных задается шаблонами, в которых описана геометрия данных. В интерпретатор может быть загружено одновременно произвольное количество шаблонов. Шаблоны могут загружаться и удаляться динамически. Управление интерпретатором выполняется с помощью команд, которые задают конкретное поведение элементов шаблонов и позволяют динамически менять данные в шаблонах. Связь между движком и остальными приложениями выполняется по протоколу TCP/IP, собственно движок является сервером, которому по указанному порту приложения могут отправлять команды. Одновременно может работать сколько угодно приложений (клиентов). Когда команды приложения загружают шаблоны в движок, то фиксируется «владелец» шаблона, т.е. то приложение, которое инициировало создание шаблона. Когда приложение завершается, то движок удаляет все шаблоны, владельцем которых было приложение.

Исключением из этого правила являются команды с префиксом **#**, если шаблон был загружен командой с таким префиксом, то он остается «жить» даже после завершения приложения-владельца.

Настройки программы

Настройки движка хранятся в файле **vgcast.cfx**. Это файл в формате XML, и его теги можно исправлять с помощью любого редактора XML-файлов (или Notepad-ом).

Журнал событий

Программа vgcast ведет журнал событий. Папка и префикс файла для записи журнала задается в файле конфигурации программы **vgcast.cfx**. Если содержимое тега **<LogFile>** не пустое, то журнал записываются в указанную папку, например (это значение по умолчанию, журнал включен)

<LogFile>.\log\vgcast</LogFile>

Указывает, что журнал нужно записывать в папку log, расположенную в той же папке, из которой запускается исполняемый файл vgcast.exe.

<LogFile>c:\temp\vgcast</LogFile>

Журнал будет создаваться в папке temp на диске C:. Имена файлов журнала имеют следующий вид

vgcast-YYMMDD.txt

Файл имеет текстовый формат.

Протокол обмена и кодировка текстовых данных

Обмен данными между приложениями и движком выполняется по протоколу TCP/IP. Порт обмена указан в файле vgcast.cfx. Команды движка – это текстовые строки.

Каждая строка должна завершаться байтом со значением 0.

Все текстовые данные в системе хранятся в кодировке Unicode, что позволяет работать в любой языковой среде и использовать одновременно алфавиты нескольких языков.

При передаче данных между клиентом и сервером используется кодировка UTF-8.

Именование объектов

Все объекты движка – шаблоны и слои – имеют имена. Разделителями имен является точка. При использовании имен в командах можно употреблять символы-шаблоны (wildcards) «*****» и «**?**». В этом случае команда применяется ко всем объектам, имена которых совпадают с учетом символов-шаблонов.

Выражения

Во многих командах можно использовать не только числа, но и выражения. В выражениях можно использовать знаки арифметических операций

+	сложение
-	вычитание
*****	умножение
/	деление
%	остаток от деления

Логические операции

&&	логическое «и»
||	логическое «или»
!	логическое отрицание («нет»)

Операции сравнения

==	равно
!=	не равно
<=	меньше или равно
>=	больше или равно
>	больше
<	меньше

Строковые операции

!!	конкатенация строк

Операции с побочным эффектом

++	унарная операция увеличения на 1, может быть как префиксной, так и суффиксной
--	унарная операция уменьшения на 1, может быть как префиксной, так и суффиксной
=	присваивание

Для группирования операций можно использовать круглые скобки. Для задания последовательности вычислений в одном выражении можно использовать оператор «**,**» (символ запятой). Применение последовательности вычислений имеет смысл только при использовании операций и функций с побочным эффектом (см. xopen).

Арифметические функции

Позволяют вставлять в выражения координаты и размеры шаблонов и слоев

neg(number)
меняет знак параметра

ceil(number)
наименьшее целое число, превышающее значение параметра

floor(number)
наибольшее целое число, не превышающее значение параметра

round(number)
округленное значение параметра

rand()
случайное целое число

min(n1, n2)
минимум двух чисел

max(n1, n2)
максимум двух чисел

argb(a, r, g, b)
формирует 32-битное число, задающее цвет с альфа-каналом из отдельных компонент

Логические функции

Позволяют вставлять в выражения условия существования объектов

is_obj(name)
возвращает 1, если объект с именем name существует. Объектом может быть имя шаблона, имя слоя или имя переменной

is_tpl(name)
возвращает 1, если шаблон с именем name существует, т.е. загружен в движок

is_layer(name)
возвращает 1, если слой с именем name существует

is_var(name)
возвращает 1, если переменная с именем name существует

is_empty(name)
возвращает 1, если слой с именем name пустой, т.е если это видеослой, то видеофайл в нем не существует. Если текстовый слой, то в нем нет литер.

Функции слоев и шаблонов

Позволяют вставлять в выражения координаты и размеры шаблонов и слоев

bx(name)
возвращает горизонтальную координату начала слоя (если name задает имя слоя tpl.lay) или шаблона (если name задает имя шаблона)

by(name)
возвращает вертикальную координату начала слоя или шаблона

ex(name)
возвращает горизонтальную координату конца слоя или шаблона

ey(name)
возвращает вертикальную координату конца слоя или шаблона

wd(name)
возвращает ширину слоя или шаблона

ht(name)
возвращает высоту слоя или шаблона

wo(name)
возвращает ширину объекта в слое

ho(name)
возвращает высоту объекта в слое

xo(name)
возвращает горизонтальную координату начала объекта

yo(name)
возвращает вертикальную координату начала объекта

text(name)
возвращает значение (содержимое) текстового объекта как строку

val(name)
возвращает значение (содержимое) текстового объекта либо как целое число, либо как число с плавающей точкой, либо как текст. Выражения типа 01:23 интерпретируются как текст.

chars(name)
возвращает количество литер в текстовом слое.

lines(name)
возвращает количество строк в текстовом слое.

prop(name, req)
возвращает свойство слоя или шаблона. Имя свойства задается параметром req.

prop(name, "bx")	bx(name)	горизонтальная координата начала слоя
prop(name, "by")	by(name)	вертикальная координата начала слоя
prop(name, "ex")	ex(name)	горизонтальная координата конца слоя
prop(name, "ey")	ey(name)	вертикальная координата конца слоя
prop(name, "wd")	wd(name	ширина слоя
prop(name, "ht")	ht(name)	высота слоя
prop(name, "xo")	xo(name)	горизонтальная координата начала объекта
prop(name, "yo")	yo(name)	вертикальная координата начала объекта
prop(name, "wo")	wo(name)	ширина объекта
prop(name, "ho")	ho(name)	высота объекта
prop(name, "xzoom")	коэффициент масштабирования по горизонтали, 10000 – без масштаба	
prop(name, "yzoom")	коэффициент масштабирования по вертикали, 10000 – без масштаба	
prop(name, "zoom")	коэффициент масштабирования по обеим координатам, 1.0 – без масштаба	
prop(name, "wipe")	текущее положение шторки [-255 .. +255]	
prop(name, "soft")	текущее значение размытости края шторки [0 ..255]	
prop(name, "trsp")	текущее значение прозрачности слоя [0 .. 256]	
prop(name, "pri")	текущий приоритет слоя или шаблона	

Функции работы со строками

Позволяют вставлять в выражения координаты и размеры шаблонов и слоев

wildcmp(wild_str, test_str)
возвращает 1 если в строке test_str содержится хотя бы одно вхождение строки wild_str. В строке wild_str можно использовать метасимволы * и ?.

strcmp(str_1, str_2)
возвращает 0 если строки str_1 и str_2 совпадают, -1 если str_1 лексикографически меньше, чем str_2, и 1 если str_1 лексикографически больше, чем str_2.

stricmp(str_1, str_2)
возвращает 0 если строки str_1 и str_2 совпадают, -1 если str_1 лексикографически меньше, чем str_2, и 1 если str_1 лексикографически больше, чем str_2. При сравнении не учитывается регистр литер.

str2num(str)

преобразует строку в число. Строка может быть представлена в формате с плавающей точкой. Полученное число можно использовать в арифметических выражениях. Например, результат выражения

=5+str2num("4.2")

Равен 9.2

num2str(number)

преобразует число с плавающей точкой в строку.

int2str(number)

преобразует целое число в строку.

Функции работы с временем и датой

Позволяют выполнять преобразования строк в миллисекунды и кадры и наоборот. Время и дата представлены в миллисекундах в эпохе Windows, т.е. начиная с 1 января 1601 г.

chr2msec(format, str)

возвращает количество миллисекунд, вычисленное из строки-параметра str. Формат строки задается параметром format. Если, например, формат задан как "hh:mm" то строка "1:30" интерпретируется как 1 час 30 минут, если формат "mm:ss", то эта же строка интерпретируется как 1 минута 30 секунд.

str2frm(str)

возвращает количество кадров, вычисленное из строки-параметра. Строка-параметр должна быть в формате hh:mm:ss:ff.

str2msec(str)

возвращает количество миллисекунд, вычисленное из строки-параметра. Строка-параметр должна быть в формате hh:mm:ss:ff.

frm2str(format, frm)

форматирует время (в кадрах) заданное параметром frm в соответствии со строкой формата format. Строка форматирования времени должна иметь следующий вид:

hh:mm hh:mm:ss hh:mm:ss:ff hh:mm:ss:dd hh:mm:ss:ddd
M:ss:ff M:ss:dd M:ss:ddd
S:dd S:ddd

Перед первым символом форматирования можно указывать знаки 0, количество знаков задает обязательное количество знакомест старшего разряда после форматирования. В качестве разделителй можно указывать любые литеры, не совпадающие с литерами форматирования.

msec2str(format, ms)

форматирует время (в миллисекундах) заданное параметром ms в соответствии со строкой формата format.

date(format, day)

возвращает текущую дату + day как строку в формате, заданном параметром format.

time(utc)

возвращает текущее время дня в миллисекундах. Если параметр utc равен нулю, то возвращает местное время, в противном случае – время по Гринвичу.

Описанные ниже функции работы с датой и временем для форматирования срок используют понятие **locale**, список доступных локалей для Windows можно посмотреть тут https://msdn.microsoft.com/en-us/library/cc233982.aspx

today()

возвращает текущую дату в миллисекундах для значения времени 00:00:00.

now()

возвращает текущую дату и время в миллисекундах.

dt(str)
datetime(str)

возвращает указанную в строке дату и время в миллисекундах. Дата должна быть указана в формате yyyy-mm-dd, время в формате hh:mm:ss. Если указаны одновременно дата и время они должны разделяться знаком пробел или T. Можно сначала задавать дату, потом время и наоборот.

ftime(locale,format,ms)

Форматирует время ms в соответствии с указанными локалью и форматом. В строке формата можно использовать такие знаки форматирования

h	Hours with no leading zero for single-digit hours; 12-hour clock
hh	Hours with leading zero for single-digit hours; 12-hour clock
H	Hours with no leading zero for single-digit hours; 24-hour clock
HH	Hours with leading zero for single-digit hours; 24-hour clock
m	Minutes with no leading zero for single-digit minutes
mm	Minutes with leading zero for single-digit minutes
s	Seconds with no leading zero for single-digit seconds
ss	Seconds with leading zero for single-digit seconds
t	One character time marker string, such as A or P
tt	Multi-character time marker string, such as AM or PM

fdate(locale,format,ms)

Форматирует дату ms в соответствии с указанными локалью и форматом. В строке формата можно использовать такие знаки форматирования

d	День месяца как число без ведущего нуля для одноразрядных чисел.
dd	День месяца как число с ведущим нулем для одноразрядных чисел.
ddd	Сокращенное название дня недели в соответствии с указанной локалью.
dddd	Полное название дня недели в соответствии с указанной локалью.
M	Номер месяца как число без ведущего нуля для одноразрядных чисел.
MM	Номер месяца как число с ведущим нулем для одноразрядных чисел.
MMM	Сокращенное название месяца в соответствии с указанной локалью.
MMMM	Полное название месяца в соответствии с указанной локалью.
y	Последняя цифра номера года.
yy	Две последние цифры номера года.
yyyy	Четыре или пять цифр номера года в зависимости от используемого календаря..

fdur(locale,format,ms)

Форматирует интервал времени ms в соответствии с указанными локалью и форматом. В строке формата можно использовать такие знаки форматирования

d	Количество дней.
h или H	Количество часов.
hh или HH	Количество часов, если число менее 10, добавляется ведущий 0.
m	Количество минут.
mm	Количество минут, если число менее 10, добавляется ведущий 0.
s	Количество секунд.

ss	Количество секунд, если число менее 10, добавляется ведущий 0.
f	Количество долей секунд. Можно указывать до 9 знаков f подряд.

Примеры

Предположим, что текущее время и дата – это 12:34:56 11-06-2018.

fdate("uk","dddd, dd MMMM yyyy",now())	понеділок, 11 червня 2018
fdate("en","dddd, dd MMMM yyyy",now())	Monday, 11 June 2018
fdur("en","'days 'd, 'hours 'h", dt("2018-06-12")-now())	days 0, hours 11
fdur("en","'days 'd, 'hours 'h", dt("2018-06-13")-now())	days 1, hours 11
fdur("en","'hours 'h", dt("2018-06-13")-now())	hours 35
fdur("en","'hours 'h", dt("2018-06-13")-today())	hours 48
fdur("en","d", dt("2018-06-13")-today())	2

Функции работы с файлами в формате XML

Позволяют выполнять выборки из файлов.

xpath(file_name, xpath_str)

выбирает из файла с именем file_name значение, соответствующее строке xpath. Результат функции можно использовать либо как строковое значение, либо как численное (выполняется попытка преобразования в целое число).

xsum(file_name, xpath_str)

суммирует все значения тегов, выбранных в соответствии с шаблоном xpath_str. В шаблоне xpath_str можно использовать специальный знак #, который задает переменную перебора тегов. Например, xsum("c:/data/vars.xml", "//score[#]") вычисляет сумму значение всех тегов <score>.

xavg(file_name, xpath_str)

вычисляет среднее арифметическое всех значения тегов, выбранных в соответствии с шаблоном xpath_str. В шаблоне xpath_str можно использовать специальный знак #, который задает переменную перебора тегов.

При работе с функциями XML нужно учитывать тот факт, что XML-файлы, указанные в качестве первого параметра функции, кэшируются. Кэширование позволяет эффективно использовать множественные выборки из одного файла, поскольку не тратится время на открытие и разбор XML-файла.

Если же нужно принудительно «перечитать» файл, то можно использовать функции xopen и xclose.

xopen(file_name)

начинает кэширование файла. Если файл уже был в кэше, он перезачитывается. Использование этой функции позволяет «синхронизировать» файл, если его содержимое меняется каким-либо другим приложением.

xclose(file_name)

удаляет файл из кэша. Вообще говоря файлы из КЭШа удаляются автоматически, когда кэш переполняется. Функцию xclose имеет смысл использовать только для освобождения ресурсов КЭШа.

Примеры использования выражений:

```
xopen("c:/data/vars.xml"), xpath("c:/data/vars.xml", "//score")
key vote.bar n=0 x=-wd(vote.bar) n=25 x=0
key vote.bar n=0 x=-wd(vote.bar) n=1:00 x=-wd(vote.bar)/2
```

Переменные

В движке есть возможность задавать переменные – именованные объекты, которые можно использовать в выражениях.

var vname =expr
int vname =expr
float vname = expr
str vname =expr

Тип переменной определяется ключевым словом int, float или str, а в случае использования ключевого слова var – типом выражения, которое задает значение переменной.

Переменные могут быть глобальными и локальными. Имена локальных переменных начинаются с имени шаблона. Время жизни локальных переменных совпадает со временем жизни шаблона, при удалении шаблона все локальные переменные этого шаблона удаляются.

Команды управления

Если команда имеет префикс #, то она интерпретируется как глобальная команда, т.е. все объекты, которые были созданы вследствие выполнения команды, не удаляются, когда закрывается сеанс связи с движком. Это в первую очередь касается команд read и load.

exit

Завершить работу исполняющей системы

shot

Сделать слепок фрейм-буфера в файл в формате ***.png**. Папка для снимков задается в файле конфигурации программы **vgcast.cfx**. Если содержимое тега <ShotRoot> не пустое, то снимки записываются в указанную папку, например

<ShotRoot>.\shot\</ShotRoot>

Указывает, что снимки нужно записывать в папку shot, расположенную в той-же папке, из которой запускается исполняемый файл vgcast.exe.

<ShotRoot>c:\temp\</ShotRoot>

Снимки будут создаваться в папке temp на диске C:. Имена файлов-снимков имеют такой вид **shot-YYMMDD-HHMMSS.png**

Формат пикселов в файле – RGB32, т.е. хранится три цветовых компоненты и альфа-канал.

replay **PRO**

Эта команда позволяет включить/выключить запись всей видеоинформации, которую рендерит движок в файл в формате MOV с кодеком PNG, кодировка RGB+ALPHA.

Например, команда

replay c:\temp\demo.mov

Начинает запись в файл, а команда

replay

Завершает запись в файл.

Команда может иметь дополнительный параметр **dur=N** или **dur=HH:MM:SS:FF**, в этом случае в выходной файл записывается только указанное количество кадров, после чего запись прекращается и файл закрывается, например

replay dur=1:12 "c:\temp\demo.mov"
replay dur=37 "c:\temp\demo.mov"

запишет в файл только 37 кадров.

trace level

Эта команда позволяет устанавливает режим вывода сообщений в консольное окно движка. Параметр level может принимать значения от 0 (минимальный вывод сообщений) до 2 (максимальный). Эту команду на уровне 2 удобно использовать для отладки эффектов и связи с приложениями.

dev key N

Эта команда позволяет устанавливает режим микширования для устройства. Параметр N зависит от реализации. Эта команда реализована только для устройств семейства Decklink, и для них N=0 означает работать без микширования, N=1 – использовать режим внутреннего микширования, и N=2 – использовать режим внешнего микширования (FILL+KEY).

Команды работы с шаблонами

read file_name

Прочитать файл с описанием шаблона. Если для слоев программы не был указан оператор **show**, то после чтения файла все его слои остаются невидимыми (полностью прозрачными) до выполнения команд **fadein** или **show**.

load file_name

Прочитать файл с описанием шаблона. Отличие от **read** состоит в том, что все слои шаблона загружаются невидимыми, вне зависимости от описания в шаблоне.

del template_name
del *

Удалить шаблон. Все слои шаблона уничтожаются и убираются из системы отображения. Если указан параметр *, то удаляются все шаблоны.

efx tpl_name.efx_name param1 param2 … param9

Выполнить эффект efx_name из шаблона tpl_name. Параметры – это произвольные строки, которые передаются в эффект при его интерпретации. Ссылка на параметры в описании эффекта задается последовательностью \1, \2, …, \9.

call tpl_name.efx_name param1 param2 … param9

Выполнить эффект efx_name из шаблона tpl_name. В отличие от команды efx все команды тела эффекта выполняются синхронно, и не учитывается время старта команд.

swap layer_name_1 layer_name_2

Поменять имена слоев между собой.

tpl tpl_name global=(0|1) lock=(0|1) rename=new_name

Управление параметрами шаблонов.

global=(0|1) динамически меняет признак глобальности шаблона. Если шаблон глобальный, то когда клиент, который создал щаблон отсоединяется от сервера vgcast, шаблон не удаляется из сервера.

lock=(0|1) блокирует шаблон от удаления. Даже применение команды del не удаляет шаблон.

rename=new_name переименовывает шаблон.

Клонирование шаблонов

Иногда необходимо использовать в одной сцене несколько одинаковых шаблонов, которые отличаются только своим положением на экране. В этих случаях можно описать только один шаблон и «клонировать» его. Шаблон, который будет выступать образцом для клонирования (мастер-клон) должен иметь специальное имя, в состав которого входит знак «#». Команда клонирования имеет такой синтаксис:

```
clone    name[#] num    x-pos y-pos
clone    name[#] num    =expr =expr
```

где **num** – это номер клона, а **x-pos** и **y-pos** – координаты клона. Если координата клона начинается со знака =, то этот параметр интерпретируется как выражение. Если в выражении используются пробелы, то все выражение, включая знак = нужно взять в кавычки. Пример команды с использованием выражений.

```
clone   dog[#]   1   =rand()%300  =rand()%200
clone   dog[#]   2   "= rand() % 300" "= rand() % 200"
```

Например, пусть описан шаблон с такой структурой

Тогда команды

```
clone   dog[#]   1   100  200
clone   dog[#]   2   100  300
```

создадут два шаблона – dog[1] и dog[2] с координатами (100, 200) и (100, 300), соответственно, и к этим шаблонам можно применять любые команды, например

```
subst    dog[1].breed  колли
subst    dog[1].name   Тузик
fadein   dog[1].*
```

При клонировании шаблонов можно сразу задавать значения слоев в виде

```
layer_name=value
layer_name="value"
```

Кавычки нужно использовать, если значение слоя содержит пробелы, например

```
clone   dog[#]   1   100  200  breed="Ирландский сеттер" name=Том
clone   dog[#]   2   100  300  breed=Лабрадор name=Митч
```

Цикл loop

loop nrep delay cmd

nrep – количество повторений цикла. Если **nrep** – это отрицательное число, то цикл выполняется бесконечно, если число положительное – цикл выполняется указанное число раз. После каждого выполнения команды **cmd** выполняется задержка **delay** миллисекунд.

Цикл выполняется асинхронно, т.е. вне зависимости от завершения цикла начинает выполняться команды следующие за командой цикла. Чтобы остановить бесконечный цикл или «досрочно» прервать цикл нужно выполнить команду **loop** без параметров или с параметром 0.

В каждый момент времени может выполняться только одна команда **loop**. Если нужно использовать несколько циклов следует применять команды **for** или **while**.

Условные операторы и циклы

if (expr) cmd

Произвольная команда **cmd** движка выполняется в том случае, когда значение вычисленного выражения **expr** не равно 0.

```
if (expr)
  cmdlist-1
else
  cmdlist-2
end
```

```
if (expr-1)
  cmdlist-1
else if (expr-2)
  cmdlist-2
else if (expr-3)
  cmdlist-2
... ... ...
else
  cmdlist-N
end
```

```
if (expr)
  cmdlist-1
end
```

```
if (expr) {
  cmdlist-1
}
```

Список команд движка **cmdlist-1** выполняется в том случае, когда значение вычисленного выражения **expr** не равно **0**, а список команд **cmdlist-2** в противном случае. Часть **else** может отсутствовать.

```
while (expr)
  cmdlist
end
```

```
while (expr) {
  cmdlist
}
```

Список команд движка **cmdlist** выполняется до тех пор, пока значение вычисленного выражения **expr** не равно **0**.

```
for (init; cond; next)
  cmdlist
end
```

```
for (init; cond; next) {
  cmdlist
}
```

Список команд движка **cmdlist** выполняется до тех пор, пока значение вычисленного выражения **cond** не равно **0**. Перед началом выполнения цикла вычисляется выражение **init**, а после каждого выполнения списка команд вычисляется выражение **next**.

Для циклов за оператором **end** или **}** можно указывать лексему **async**, в этом случае цикл будет выполняться асинхронно, т.е. команды за циклом начнут выполняться до завершения цикла.

break

Прекращает выполнение цикла.

continue

Переходит на следующий шаг цикла.

cancel
cancel async

Прекращает выполнение асинхронного цикла **while** или **for**.

sleep expr
sleep expr async

Задерживает выполнение следующей команды на указанное количество миллисекунд. Если нужно выполнять задержку внутри асинхронного цикла, то необходимо использовать второй вариант команды с модификатором **async**.

Условные операторы и циклы могут быть вложенными

При использовании условных операторов и циклов нужно внимательно следить за тем, чтобы соблюдалось соответствие открывающих ключевых слов – if, while, for – и закрывающих – end.

Нужно учитывать тот факт, что система допускает использование бесконечных циклов.

Примеры

Простой условный оператор

```
int @.n =1
```

```
if (@.n==1) subst    @.bg[0]    TRUE
```

Условный оператор

```
int @.n =1
if (@.n==1) {
   subst    @.bg[0]    TRUE
else
   subst    @.bg[0]    FALSE
}
```

Вложенные условные операторы

```
int @.n =1
int @.m =0
if (@.n)
   subst @.bg[0] TRUE
   if (@.m)
      subst          @.bg[1] TRUE
   else
      subst          @.bg[1] FALSE
   end
else
   subst @.bg[0] FALSE
end
```

Условный оператор с несколькими ветвями выбора

```
if          (\1==0)
   subst @.bg[0] 0: TRUE
else if    (\1==1)
   subst @.bg[1] 1: TRUE
else if    (\1==2)
   subst @.bg[2] 2: TRUE
else if (\1==3)
   subst @.bg[3] 3: TRUE
else
   subst @.bg[4] 4: FALSE
end
```

Цикл типа **while**

```
int         @.n=0
while (@.n<4) {
   xsubst           @.bg[=@.n;]       =@.n
   if       (@.n==2)          break
   expr   @.n++
}
```

Цикл типа **for**

```
for         (int @.n=0; @.n<4; @.n++) {
   xsubst           @.bg[=@.n;]       field[=@.n;]
}
```

Управление прозрачностью объектов

fadein layer_name [speed]
fi layer_name [speed]
fadeout layer_name [speed]
fo layer_name [speed]

Ввести (убрать) слой микшером. Параметр speed задает количество кадров, за которое выполняется микшер. Если значение speed не указано, то эффект выполняется за один кадр (CUT).

trsp layer_name value

Задать для слоя прозрачность value от 0 (полностью прозрачный) до 256 (непрозрачный). Значение прозрачности, заданное в описании шаблона заменяется на value.

Если микшируется слой типа «видео со звуком», то при выполнении команд fi/fo изменяется и громкость соответствующей звуковой дорожки.

wipe layer_name [wipe=]WWW [soft=]SSS [file=]file_name

Задает параметры шторки.

Подстановки

subst layer_name param

Выполнить подстановку в слое с именем layer_name. Значение param зависит от типа слоя.

subst video "file" in=N out=M trim=N,M loop=N field=F map=A,...,Z scale=N adelay=N track=N par=N volume=N halign=N valign=N

Выполнить подстановку в видеослое с именем video. Имя файла контента желательно указывать в кавычках.

in, out, trim задают метки входа N и выхода M в клипе. Формат меток HH:MM:SS:FF.

loop задает количество повторений клипа (если указано loop=max, то повторяется бесконечно).

field задает порядок полей в клипе, 0 = первое поле верхнее, 1 = первое поле нижнее. Если значение = -1 – используется порядок полей, заданный в параметрах слоя.

map указывает переназначение звуковых каналов клипа. Через запятую должны быть указаны номера звуковых дорожек исходного клипа. Например, параметр map=A,B указывает что на выходную звуковую дорожку с номером 0 будет отображена дорожка A, а на дорожку 1 – дорожка B. Если в качестве номера дорожки указать -1, то на эту дорожку звук не отображается. Примеры: map=1,0 – переставляет левый и правый каналы звука; map=-1,0 – на левый канал ничего не выводится, на правый выводится левая дорожка клипа.

scale задает режим масштабирования, целое число (0 – без масштабирования, 1 – масштабирование с сохранением аспекта, 2 – анаморфное масштабирование, 3 – с аспектом многопотоковое, 4 – анаморфное многопотоковое, 5 – с сохранением аспекта и без эффекта letterbox/pillarbox, 6 – многопотоковая версия режим 5).

adelay задает задержку звука относительно видео (в кадрах).

track номер звуковой дорожки в исходном файле.

par пропорции пикселя, число с плавающей точкой.

volume громкость звука, 0 – масимальная, -10000 - минимальная.

halign, valign выравнивание видео внутри слоя, 0 – выравнивание по левому (верхнему) краю, 1 -. По правому (нижнему) краю, 2 – по центру.

fsubst layer_name file_name

Выполнить подстановку в слое с именем layer_name. Содержимое подстановки выбирается из файла с заданным именем.

xsubst layer_name string_with_expr

Выполнить подстановку в слое с именем layer_name. В отличие от команды subst сначала выполняется поиск и вычисление выражений в строке подстановки и названии слоя. Выражения начинаются со знака **=** и заканчиваются знаком **;**. Затем выполняется стандартная команда подстановки в зависимости от типа слоя.

queue layer_name param

Поставить в очередь roll/crawl объект в слое с именем layer_name. Значение param зависит от типа слоя. Когда текущий элемент проходит область вывода (по вертикали или по горизонтали, в зависимости от типа барабана), из очереди выбирается новый объект и начинает движение по области вывода. Когда объект полностью покидает область вывода, он удаляется и все ресурсы, связанные с ним освобождаются. В очередь можно ставить объекты к слоям типа **still**, **movie**, **mpeg**, **image**, **text**, (**clock**?).

При выполнении команды **queue** инициатору команды выдаются дополнительные сообщения (описание приведено в предположении, что в очередь ставятся бегущие строки, направление справа-налево):

!qi ID – (queue init) объект поставлен в очередь и ему присвоен идентификатор **ID**.

!qs ID – (queue start) объект, находящийся в очереди начал отображаться, его левый край появился в области вывода.

!qn ID – (queue next) правый край отображаемого объекта появился в области вывода. Фактически это событие означает, что из очереди будет выбран следующий объект.

!qe ID – (queue end) правый край отображаемого объекта покинул область вывода.

Где **ID** – это уникальный идентификатор объекта, беззнаковое целое число.

push layer_name param

Поставить в очередь roll/crawl объект в слое с именем layer_name. Значение param зависит от типа слоя. Когда текущий элемент проходит область вывода (по вертикали или по горизонтали, в зависимости от типа барабана), из очереди выбирается новый объект и начинает движение по области вывода. Если в момент, когда текущий объект выведен, в очереди нет новых элементов, то барабан останавливается.

fqueue layer_name file_name

Выполнить команду queue с выборкой содержимого подстановки из файла с именем file_name.

fpush layer_name file_name

Выполнить команду push с выборкой содержимого подстановки из файла с именем file_name.

qmode layer_name (play|pause|sts|info|next|prev) speed=N align=(top|bottom|left|right) block=N

Останавливает или продолжает прокрутку барабана.

Работа с опорными кадрами

Задает последовательность ключевых кадров. Каждый ключевой кадр может содержать такую информацию: время, координаты объекта, размеры и прозрачность. Интерполяция параметров между ключевыми кадрами может выполняться либо линейно, либо по B-сплайну, в зависимости от типа ключевого кадра. Описание ключей состоит из параметров, разделенных пробелами. Значение параметра (все, что следует за знаком =) может быть выражением.

n=NN – (**on=**NN) ключевой кадр, выполняется линейная интерполяция параметров. NN задает номер кадра, для которого указаны параметры. Если номер кадра задан в виде **n+**NN, то NN задает смещение относительно ***предыдущего*** ключевого файла.

f=NN – (**off=**NN, quad=NN) ключевой кадр, выполняется сплайновая интерполяция параметров. NN задает номер кадра, для которого указаны параметры. Если номер кадра задан в виде **f+**NN, то NN задает смещение относительно ***предыдущего*** ключевого файла.

F=x1,y1,x2,y2 – (**cubic=**x1,y1,x2,y2) ключевой кадр, который задает интерполяцию по формуле кубической кривой Безье. (x1,y1) и (x2,y2) – это вектора в нормированном на 1 пространстве, которые задают управляющий точки кривой. Первая (0,0) и последняя (1,1) точка кривой задаются неявно, из предыдущего и последующего кадра типа n=NN.

I (**in**, **easyin**) – сокращение для F= .42,0,1,1. Параметры интерполируются так, что они меняются с ускорением.

O (**out**, **easyout**) – сокращение для F= 0,0,.58,1. Параметры интерполируются так, что они меняются с замедлением.

E (inout, easy) – сокращение для F= .42,0,.58,1. Параметры интерполируются так, что они меняются медленно-быстро-медленно 

Ключевые кадры объекта внутри слоя

key layer_name key1 key2 … keyN

Задает изменение параметров объекта внутри слоя

x=NN – (bx=NN) координата X объекта внутри слоя

y=NN – (by=NN) координата Y объекта внутри слоя

h=NN – (hscale=NN) ширина слоя (зум по горизонтали), нормальная ширина = 1000

v=NN – (vscale=NN) высота слоя (зум по вертикали), нормальная высота = 1000. Если указано значение параметра большее 1000, то объект увеличивается, если меньшее – уменьшается. Отрицательное значение приводит к «переворачиванию» объекта по вертикали.

t=NN – (trsp=NN) прозрачность слоя

w=NN – (wipe=NN) положение шторки (если шторка была задана в описании слоя). Значение NN может меняться от 0 до 255.

s=NN – (soft=NN) размытость края шторки (если шторка была задана в описании слоя). Значение NN может меняться от 0 до 255.

z=NN – (zoom=NN) коэффициент изменения размеров объекта слоя. Выражение (или константа) с плавающей точкой. Объект увеличивается (NN > 1.0) или уменьшается (NN < 1.0) в соответствии с этим коэффициентом. При этом учитываются параметры выравнивания объекта, т.е. если задано выравнивание по центру, то увеличение или уменьшение объекта выполняется относительно центра слоя.

zx=NN – (xzoom=NN) ширина слоя (зум по горизонтали), нормальная ширина = 1.0. Выражение (или константа) с плавающей точкой. Объект увеличивается (NN > 1.0) или уменьшается (NN < 1.0) по горизонтали в соответствии с этим коэффициентом. Аналог параметра h, только в нормировке на 1.0.

zy=NN – (yzoom=NN) высота слоя (зум по вертикали), нормальная высота = 1.0. Выражение (или константа) с плавающей точкой. Объект увеличивается (NN > 1.0) или уменьшается (NN < 1.0) по вертикали в соответствии с этим коэффициентом. Аналог параметра v, только в нормировке на 1.0.

Цветокоррекция

H=NNN – (hue=NNN) цветность изменяется в пределах от -180° до +180°

S=NNN – (sat=NNN) насыщенность от -100% до +100%

L=NNN – (luma=NNN) светлота от -100% до +100%.

Colorization

Добавлена возможность выполнять динамически "подтягивать" цвет изображения к заданному цвету. Задается двумя параметрами.

C=NNN – (cc=NNN) цветность (hue) в пространстве HSL, меняется от 0 до 255, задает цвет, к которому будут "придвигаться" все компоненты hue изображения.

K=NNN – (k=NNN) коэффициент влияния этого цвета, меняется от 0 (все цвета остаются без изменений) до 255 (все цвета заменяются на указанный).

Ключевые кадры масштабирования объекта

zkey layer_name key1 key2 … keyN

Задает масштабирование объекта внутри слоя.

z=NN (zoom=NN) – коэффициент изменения размеров объекта. Выражение (или константа) с плавающей точкой. Объект увеличивается (NN > 1.0) или уменьшается (NN < 1.0) в соответствии с этим коэффициентом. При этом

учитываются параметры выравнивания объекта, т.е. если задано выравнивание по центру, то увеличение или уменьшение объекта выполняется относительно центра слоя.

Ключевые кадры параметров слоя

lkey layer_name key1 key2 ... keyN

Задает изменение параметров самого слоя

x=NN – (bx=NN) координата X слоя

y=NN – (by=NN) координата Y слоя

cx=NN – (X=NN) координата X центра слоя

cy=NN – (Y=NN) координата Y центра слоя

h=NN – (wd=NN, width=NN)) ширина слоя в пикселях

v=NN – (ht=NN, height=NN) высота слоя в пикселях.

t=NN – (trsp=NN)прозрачность слоя (то же, что и в команде **key**)

w=NN – (wipe=NN) положение шторки (если шторка была задана в описании слоя). Значение NN может меняться от 0 до 255 (то же, что и в команде **key**)

s=NN – (soft=NN)размытость края шторки (если шторка была задана в описании слоя). Значение NN может меняться от 0 до 255 (то же, что и в команде **key**)

z=NN – (zoom=NN) коэффициент изменения размеров объекта слоя. Выражение (или константа) с плавающей точкой. Объект увеличивается (NN > 1.0) или уменьшается (NN < 1.0) в соответствии с этим коэффициентом. При этом учитываются параметры выравнивания объекта, т.е. если задано выравнивание по центру, то увеличение или уменьшение объекта выполняется относительно центра слоя.

Ключевые кадры параметров шаблона

tkey tpl_name key1 key2 ... keyN

Задает изменение параметров шаблона, т.е. применяются ко всем слоям шаблона.

x=NN – (bx=NN) координата X шаблона

y=NN – (by=NN) координата Y шаблона

t=NN – (trsp=NN)прозрачность всех слоев шаблона (то же, что и в команде **key**)

z=NN – (zoom=NN) коэффициент изменения размеров всех объектов шаблона. Выражение (или константа) с плавающей точкой. Объект увеличивается (NN > 1.0) или уменьшается (NN < 1.0) в соответствии с этим коэффициентом. При этом учитываются параметры выравнивания объекта, т.е. если задано выравнивание по центру, то увеличение или уменьшение объекта выполняется относительно центра слоя.

Расположение слоев на экране

hslide layer_name speed from_x to_x
vslide layer_name speed from_y to_y

Выполнить эффект слайд над объектом слоя. Движение объекта выполняется со скоростью speed (количество пикселов за один кадр). Используется система координат относительно левого верхнего угла прямоугольника слоя, причем значение 0 обозначает положение объекта внутри прямоугольника, +1 – объект находится справа от прямоугольника, -1 – слева. Например,

```
hslide foo.bar 2 1 0
```

объект «приедет» по горизонтали справа в прямоугольник и останется в нем.

```
hslide foo.bar 2 -1 1
```

объект «приедет» по горизонтали слева в прямоугольник и «уедет» вправо от прямоугольника.

speed ::= n | speed=n | dur=n

Скорость движения задается в пикселях за один полукадр. Длительность задается в кадрах.

hmove layer_name speed from_x to_x
vmove layer_name speed from_y to_y

Выполнить эффект слайд над объектом слоя. Движение объекта выполняется со скоростью speed (количество пикселов за один кадр). Используется система координат относительно левого верхнего угла прямоугольника слоя, указываются относительные координаты начального и конечного положения объекта. Например,

```
hmove foo.bar 2 150 50
```

объект «проедет» по горизонтали справа от координаты 150 до координаты 50.

roll layer_name speed [rep]

Выполнить вертикальную прокрутку слоя (барабан). Движение объекта выполняется со скоростью speed (количество пикселов за один полукадр). Если значение speed положительное, выполняется прокрутка снизу вверх, если отрицательное – сверху вниз. Можно задать количество повторений барабана (rep).

crawl layer_name speed [rep]

Выполнить горизонтальную прокрутку слоя (бегущая строка). Движение объекта выполняется со скоростью speed (количество пикселов за один кадр). Если значение speed положительное, выполняется прокрутка слева направо, если отрицательное – справа налеко. Можно задать количество повторений бегущей строки (rep).

path layer_name speed x0 y0 x1 y1 … xN yN

Задает кривую, по которой будет перемещаться слой. Пар x, y должно быть столько же, сколько кадров в параметре speed.

Расположение шаблонов на экране

tmove template_name param=val xXs yYs xXe yYe PRO

Задает перемещение шаблона из точки Xs, Ys в точку Xe, Ye. Можно задавать такие параметры:

dur=NN

Общая длительность эффекта в кадрах. По умолчанию 25 кадров.

wave=N

Количество промежуточных точек интерполяции кривой. По умолчанию равно 0, т.е. движение происходит по прямой.

left=N.N

right=N.N

Коэффициент кривизны и направление выпуклости. Если задан параметр left, то кривая выпукла влево от направления движения, если right, то вправо. Значение параметра задает расстояние промежуточной точки интерполяции по нормали к вектору из начальной к конечной точке. В качестве исходной длины нормали используется длина вектора, которая умножается на заданный коэффициент. Т.е. если значение left=1.0, то точка к которой «подтягивается» кривая будет расположена достаточно далеко, и кривизна траектории будет довольно большой.

lefta=N.N

righta=N.N

Имеет смысл использовать если количество точек интерполяции более 1. Тогда направление кривизны кривой будет последовательно меняться, например, сначала влево, затем вправо и т.д. Таким образом, можно получить волнообразную траекторию.

conv=N.N

Коэффициент затухания волн в траектории. Если значение параметра < 1, то волны будут затухать, если > 1, то будут увеличиваться.

tpos template_name XX YY **PRO**

Задает новые координаты шаблона, на самом деле эквивалент команды

tkey template_name n1 xXX yYY

torder template_name param=val N1 N2 ... Nn **PRO**

Задает перестановку клонов в массиве клонов

Команды слоев

type layer_name dur string **PRO**

Вывести значение текстового поля эффектом «пишущей машинки». Т.е. литеры из строки string будут появляться одна за другой, с заданной задержкой (в кадрах).

text layer_name dur key value [format]

Довольно специфичная команда, позволяет реализовать часто используемый прием вывода информации в текстовые поля. С каждым текстовым полем связан числовой параметр. По этой команде выполняется интерполяция значения этого параметра от текущего к заданному, и непрерывный вывод этих значений в текстовое поле. Если поле формата не задано, используется формат %.0f.

text layer_name dur color value

Меняет цвет литер текста на указанный (value). Параметр dur игнорируется (но должен присутствовать, например указывайте 0).

solid layer_name dur color value [fore value]

Изменяет цвет слоя типа solid. Промежуточные цвета интерполируются от текущего значения к указанному. Параметр fore нужен для использования в градиентных заливках. Значение цвета можно указывать как шестнадцатеричное в виде 0xFFFFFFFF. Самые левые (старгие) шестнадцатеричные цифры задают прозрачность.

clock layer_name (play|stop|pause|resume|countdown msec|set msec)

Управление часами и таймерами.

video layer_name (play | pause | restart | out | volume | speed | fps | seek | playto | reply | delay | sts | dump)

Управление анимацией типа movie.

volume=expr

значение параметра volume задает уровень звука, expr = 0 – максимальный уровень, -10000 = -100dB – минимальный уровень (тишина).

speed=expr

установить скорость воспроизведения видео, если **expr=1** – нормальная скорость, если **expr<1** – замедленное воспроизведение, если **expr>1** – ускоренное воспроизведение

fps=N/M

альтернативный способ задания скорости воспроизведения, задается как рациональное число равное N кадров в M секунд. Если M не задано оно считается равным 1. Примеры: 25 – 25 кадров в секунду, 25000/1000 – 25 кадров в секунду, и т.д.

seek=expr

выполнить позиционирование видеофайла на кадр, являющийся значением выражения expr. Для кодеков с GOP структурой позиционирование может быть неточным. Желательно применять к файлам с внутрикадровым кодированием.

playto=expr

воспроизводить видеофайл с текущей позиции, до кадра являющимся значением выражения expr. Можно воспроизводить файлы как вперед, так и назад. Только для кодеков с внутрикадровым кодированием..

reply=expr

отправлять уведомления в виде !fr N, где N – номер текущего кадра

delay=expr

задержать видео на expr кадров

sts
выдать статистику заполнения буферов

dump
выдать развернутую статистику заполнения буферов

chart layer_name (reset|render|add num_point x1 y1 ...)
Управление диаграммами.

Работа со звуком

sound file_name | sound track_name file_name
Сыграть звуковой файл. Если не задан номер дорожки (track_name) то звук воспроизводится на дорожке 0. Звуки, воспроизводимые на разных дорожках, микшируются.

sound | sound track_name
Прекратить воспроизведение звука на заданной дорожке (или на умалчиваемой дорожке). Соответствующая звуковая дорожка освобождается.

si track_name dur
so track_name dur
Плавно микшировать звук за dur кадров.

audio track_name create "file_name" map=A,B,...
Загрузить аудиофайл на дорожку с именем track_name. Параметр map задает переотображение звуковых каналов файла в каналы выходного устройства. Имя файла file_name желательно указывать в двойных кавычках.

audio track_name delete
Удалить дорожку с именем track_name.

audio track_name play
Запустить воспроизведение звука на дорожке track_name.

audio track_name pause
Пауза на дорожке track_name.

audio track_name resume
Продолжить звук после паузы на дорожке track_name.

audio track_name limit NN
Установить верхний предел уровня звука на дорожке track_name. Диапазон уровня звука изменяется от -10000 (минимум, тишина) до 0 (максимальный уровень). NN = -dB*100.

audio track_name volume NN
Установить текущий уровень звука на дорожке track_name.

audio track_name loop NN
Установить количество повторений трека на дорожке track_name.

audio track_name map=A,B,...
Параметр map задает переотображение звуковых каналов файла в каналы выходного устройства. Желательно ремапинг задавать в команде create, поскольку сразу после создания дорожки звук начинает кэшироваться, и команда audio track map=A,B,... будет применяться ко всем аудиоданным, которые еще не попали в кэш. В кэшированных данных ремапинг звуковых каналов не меняется.

Работа с объектами 3D

mode name=val name=val ... PRO
Управление режимами работы интерпретатора

mode 3d=1 устанавливает режим преобразования всей сформированной сцены в трехмерном пространстве. Задается перспективная проекция.

dur=NN задает количество кадров, за которое будет выполняться интерполяция остальных параметров. Если значение dur не задано, то преобразования выполняются за 1 кадр.

xrot, yrot, zrot, xoff, yoff, zoff – задают повороты (в градусах) и сдвиги (в ???) по соответствующим осям. Поскольку используется перспективная проекция, то значение zoff должно быть отрицательным, причем значение около -12 соответствует примерно масштабу 1:1. Чем меньше значение zoff, тем меньший масштаб картинки. Плоскость отсечения проходит по значению оси Z = -5, поэтому объекты у которых смещение zoff больше -5 не выводятся. Значения этих параметров могут задаваться как значения с плавающей точкой, но в этом случае нужно все значение заключать в двойные кавычки, например zoff="-12.3"

pie3d layer_name dur pie_command PRO

Команда управления трехмерными секторными диаграммами. Параметр dur задает длительность эффекта в кадрах. Параметр pie_command задает либо поведение всего объекта, либо отдельных секторов. Интерполяция параметров выполняется за заданное количество кадров от текущего значения к указанному (линейно). Значения параметров (кроме углов) должно быть недалеко от 1.

zrot angle – повернуть диаграмму на угол angle вокруг оси Z. Угол задается в градусах.

xrot angle – повернуть диаграмму на угол angle вокруг оси X. Угол задается в градусах.

yrot angle – повернуть диаграмму на угол angle вокруг оси Y. Угол задается в градусах.

zoff dist – передвинуть диаграмму на заданное расстояние по оси Z.

pNN sector_command – команда конкретного сектора в диаграмме. NN задает номер сектора от 0 до максимального количества секторов в диаграмме. Если NN не указано, или задана команда p*, то команда применяется ко всем секторам диаграммы.

p0 ro NN – изменить внешний радиус сектора 0 до величины NN.

p0 ri NN – изменить внутренний радиус сектора 0 до величины NN (радиус дырки).

p0 ht NN – изменить высоту сектора 0 до величины NN.

p0 ex NN – изменить отступ сектора 0 от центра до величины NN.

p0 wa NN – изменить угол сектора 0 до величины NN.

p0 sa NN – изменить начальный угол сектора 0 до величины NN.

p0 ct 0xAARRGGBB – изменить цвет сектора 0 до указанного значения.

bar3d layer_name dur bar_command PRO

Команда управления трехмерными столбчатыми диаграммами. Параметр dur задает длительность эффекта в кадрах. Параметр bar_command задает либо поведение всего объекта, либо отдельных столбиков. Интерполяция параметров выполняется за заданное количество кадров от текущего значения к указанному (линейно).

p0 ro NN – изменить размер основания столбика по оси X до величины NN. NN – это числа с плавающей точкой, вообще говоря их значения должны быть недалеко от 1.0

p0 ht NN – изменить размер основания столбика по оси Y до величины NN.

p0 ri NN – изменить смещение столбика по оси X до величины NN. Может быть отрицательной или положительной величиной

p0 ex NN – изменить смещение метки столбика по оси Y до величины NN.

p0 sa NN – изменить смещение основания столбика по оси Y до величины NN.

p0 wa NN – изменить высоту столбика по оси Y до величины NN.

p0 ct 0xAARRGGBB – изменить цвет верхушки столбика.

p0 cb 0xAARRGGBB – изменить цвет основания столбика.

Информационные команды

help [command_name]
Выдать короткую справку по командам

sts display
Выдать состояние системы отображения

sts layer
Выдать состояние всех слоев шаблонов

sts comm
Выдать состояние коммуникационой подсистемы

sts template
Выдать состояние шаблонов

sts style
Выдать состояние стилей

sts mem
Выдать текущее распределение памяти

sts efx
Выдать очередь активных эффектов

sts time
Выдать тайминг видеоформата: количество кадров, количество полей.

sts dev
Выдать список используемых устройств

sts audio
Выдать звукового драйвера и состояние звуковых дорожек

sts video
Выдать состояние видеодрайверов и, если драйвер поддерживает, дополнительную информацию об опрном сигнале и пр.

Для информационных команд желательно указывать точное имя слоя, без *. Если указано расширенное имя слоя, то на один запрос будет выдано несколько строк с информацией о слоях, но определеть к какому слою относится какая информация невозможно.

info pos layer_name
Выдать текущие координаты слоя. Ответ передается в виде текстовой строки

pos XXX YYY

где XXX YYY – координаты левого верхнего угла объекта относительно области вывода слоя. Могут быть как положительные значения, так и отрицательные.

info size layer_name
Выдать текущие размеры слоя. Ответ передается в виде текстовой строки

size WWW HHH

где WWW HHH – текущие размеры объекта. Команда полезна для объектов, размер которых вычисляется во время выполнения, например для текстовых объектов.

info trsp layer_name
Выдать текущую прозрачность слоя. Ответ передается в виде текстовой строки

trsp TTT

где TTT – текущая прозрачность объекта. 0 = объект полностью прозрачный (невидимый), 256 = объект полностью непрозрачный.

info rect layer_name

Выдать область вывода слоя. Ответ передается в виде текстовой строки

rect BX BY EX EY

где BX – абсолютная координата левого угла области вывода, BY – абсолютная координата верхнего угла области вывода, EX – абсолютная координата правого края области вывода+1, EY – абсолютная координата нижнего края области вывода+1. Таким образом, ширина области вывода WD = EX – BX, высота области вывода HT = EY – BY.

Сервисные команды

runlog ? fmt="format" reg=name once=name

Включает логирование видеофайлов в указанном слое.

Изменение атрибутов объектов

attr layer_name param=val ...

Команда позволяет динамически изменять некоторые атрибуты объектов.

Общие атрибуты

onsubst=N

если 1, то включается режим вызова эффекта после выполнения подстановки. Если в шаблоне объявлен эффект с именем $имя_слоя, то он будет выполнен после операции подстановки.

Специальный режим масштабирования текста

Добавлен специальный режим масштабирования текста. Скорее всего, он наиболее полезен в тех случаях, когда задано выравнивание текста по центру. При использовании стандартной процедуры масштабирования, особенно когда эффект растянут во времени, возникает неприятное визуальное «дрожание» текста. Это связано с тем, что масштабирование (а в случае центрального выравнивания – еще и движение) выполняется не на субпиксельном уровне, и именно ошибки округления вызывают «дрожание» текста.

Чтобы активировать этот режим нужно предварительно выполнить команду

```
attr    @.txt zoom=1
```

Это есть смысл делать в эффекте **show**, сразу после загрузки шаблона.

В конфиге vgcast.cfx есть возможность точной настройки параметров масштабирования.

```
<SpecZoom hblur="0" vblur="0" alg="1"/>
```

Параметр **alg** задает алгоритм, используемый для масштабирования. По умолчанию равно 1.

1	Быстрая билинейная интерполяция.
2	Стандартная билинейная интерполяция, дает более точный результат, но требует больше времени.
4	Бикубическая интерполяция
16	По ближайшей точке
32	По области
128	Гаусс (Gauss)
256	Sinc
512	Ланцош (Lanczos)
1024	Сплайн

Параметры hblur и vblur задают степень размывки про горизонтали и вертикали. Целое число от 0 до 64. Существенно увеличивают время вычислений. Если число больше 0, то степень размывки меняется динамически, в зависимости от коэффициента масштабирования, чем ближе коэффициент масштабирования к 0, тем выше степень размывки. Если число меньше 0, то степень замывки остается постоянной, вне зависимости от коэффициента масштабирования.

Атрибуты текстовых слоев

face=N
изменяет начертание шрифта. N может принимать значения 0 – обычный шрифт, 1 – жирный, 2 – курсив, 3 – жирный курсив.

caps=N
устанавливает регистр литер. 0 – не менять регистр, 1 – малые заглавные буквы, 2 – первые буквы слов заглавные, 3 – все литеры заглавные

fsize=N
устанавливает размер (высоту литер) шрифта

cspace=N
меняет межлитерное расстояние. Значение 0 соответствует стандартному межлитерному расстоянию, заданному в шрифте. Отрицательные числа уменьшают его, а положительные – увеличивают.

wspace=N
меняет межсловное расстояние. Значение 0 соответствует стандартному межсловному расстоянию, заданному в шрифте. Отрицательные числа уменьшают его, а положительные – увеличивают.

lspace=N
меняет межстрочное расстояние. Значение 0 соответствует стандартному межстрочному расстоянию, заданному в шрифте. Отрицательные числа уменьшают его, а положительные – увеличивают.

hscale=N
задает горизонтальное масштабирование литер в процентах. 100 – нормальный масштаб.

halign=N
задает режим выравнивания текста по горизонтали. 0 – выравнивание по левому краю, 1 – выравнивание по правому краю, 2 – выравнивание по центру.

valign=N
задает режим выравнивания текста по горизонтали. 0 – выравнивание по верхнему краю, 1 – выравнивание по нижнему краю, 2 – выравнивание по центру.

base=N
задает сдвиг базовой линии текста. Значение 0 соответствует стандартному положению базовой линии, заданному в шрифте. Отрицательные числа сдвигают базовую линию вверх, а положительные –вниз.

wrap=N
задает режим переноса слов. 0 – не переносить слова, 1 – переносить слова, если текст не помещается в слой

fitext=N
режим масштабирования текста. 0 – не масштабировать текст, 1 – уменьшать размер литер текста до тех пор, пока весь текст не поместится в слой

kern=N
использование кернинга. Значение 0 – не использовать кернинг, 1 – использовать кернинг.

maxlines=N
задает максимальное количество строк при переносе слов. 0 – не ограничивает количество строк.

color=0xAARRGGBB
задает цвет текста.

Описание шаблона

Поскольку формирование шаблонов выполняется с помощью программы VgEdit, описание шаблонов приведено скорее для внутреннего использования.

Каждая команда описания шаблона должна располагаться на отдельной строке. Комментарий начинается со знаков // и продолжается до конца строки. Имена шаблонов и слоев могут состоять из произвольных знаков, включая пробелы.

Шаблон состоит из произвольного количества слоев. Порядок описания слоев задает их порядок на экране (z-order), самый первый слой является самым нижним, самый последний слой – самый верхний. В каждом слое задается его геометрия (размеры и положение на экране) и описывается тип слоя – заливка сплошным цветом, изображение, анимация, часы и т.д. Никаких ограничений на взаимное расположение слоев нет, они могут перекрываться.

Можно явно задавать приоритет слоя. Это принципиально важно для тех слоев, которые априори должны располагаться поверх других – часы, логотипы и т.д.

Каждый шаблон имеет имя, которое используется для ссылки на элементы шаблона. Имя шаблона указывается командой

tnam template_name
template template_name

где template_name произвольная последовательность литер (включая пробелы) до конца строки или до комментария. Если команда **tnam** отсутствует в шаблоне, то шаблон имеет такое же имя, как и файл шаблона.

Каждый слой описывается группой команд, начало слоя задается командой **layr (layer)**, завершает описание слоя команда **elay (endlayer)**.

layer

команда описания слоя

команда описания слоя

endlayer

Некоторые команды описания слоя являются общими для всех типов слоев, некоторые специфичны для конкретных типов слоев.

Общие операторы описания слоя.

name layer_name

Команда задает имя слоя, это произвольная последовательность литер (включая пробелы) до конца строки или до комментария. Вообще говоря имя слоя может отсутствовать, его необходимо задавать только если нужно менять параметры слоя в процессе работы (например, менять его прозрачность или положение на экране) или выполнять подстановки.

type layer_type

Команда задает тип слоя, где layer_type может принимать следующие значения

soli | **solid** – слой заполняется сплошным цветом

grad | **gradient** – слой заполняется градиентной заливкой

imag | **image** – слой содержит изображение. Допустимые форматы картинок TGA, BMP, VII, VIS, VIM.

still – слой содержит изображение, все операции с слоем такого типа совпадают с форматом **image**. Допустимые форматы картинок – все те, которые поддерживаются системой DirectX Microsoft Windows, например TGA, BMP, JPG, GIF, AVI, и т.д. Если указаны форматы файлов, содержащие видеоинформацию, то в качестве картинки используется первый кадр видеопотока. Для того, чтобы можно было читать и декодировать изображения, в DirectX должен быть установлен соответствующий кодек. Так, например, для чтения файлов в формате MPEG-2 нужно установить какой-либо из программных декодеров MPEG-2, например Elecard.

movi | **movie** – слой содержит анимацию с альфа-каналом

mpeg – слой содержит анимацию в формате MPEG-1/2

text –слой содержит форматированный текст

clck | **clock** – слой содержит часы или таймер

chart – слой содержит диаграмму

В зависимости от типа слоя в описании слоя могут присутствовать дополнительные команды. **rect** left top width height

Эта команда описывает геометрию слоя – его позицию на экране и размер области. Размер слоя может не совпадать с размером объекта, формирующего слой.

trsp number
transparency number

Прозрачность слоя, может изменяться от 0 до 256. Если эта команда не задана, то по умолчанию прозрачность слоя устанавливается в 256 – полностью непрозрачный слой. Если прозрачность слоя задана, то команды **fade** не могут установить ее больше чем number.

pri number | min | max
priority number | min | max

Приоритет слоя, может принимать любое целочисленное значение, как положительное, так и отрицательное. Значение **min** соответсвует минимальному приоритету (самый нижний слой), **max** – максимальному приоритету (самый верхний слой). Если эта команда не задана, то по умолчанию приоритет слоя устанавливается в 0. Слои с одинаковыми приоритетами располагаются в порядке загрузки слоев, самый последний загруженный слой становится самым верхним.

pos x-coord y-coord

Исходные координаты объекта, формирующего слой относительно прямоугольника слоя. Для использования команд **slide** или **roll/crawl**, чтобы исходное положение объекта было за пределами прямоугольника слоя.

show

Исходные состояние прозрачности слоя. По умолчанию все слои создаются полностью прозрачными, и становятся доступны для отображения только после команд **show** или **fadein**. Если же в описании слоя указан оператор **show**, то слой становится видимым сразу после создания (чтения файла с шаблоном).

Операторы, специфичные для конкретных слоев

file file_name

Команда **file** задает имя файла, содержащего объект описывающий слой. Для слоя типа **image** это графический файл в формате TGA, BMP, VII, VIS или VIC. Для слоя типа **movie** – файл содержащий анимацию в формате VIM. Для слоя типа **clock** – описание часов в формате CLK.

alpha number

Команда **alpha** принудительно задает прозрачность (альфа-канал) для изображений типа **still** (DirectX).

fit (0 | 1)

Команда **fit 1** указывает, что изображение нужно масштабировать до размеров области вывода слоя (в текущей реализации – только в сторону уменьшения).

hpos (0 | 1 | 2)

Команда **hpos** указывает, как изображение выравнивается относительно области вывода по горизонтали. Значение 0 указывает, что выравнивание происходит поо левому краю области вывода, значение 1 – по правому краю, значение 2 – по центру области. Применимо для слоев типа **image** и **still**.

vpos (0 | 1 | 2)

Команда **vpos** указывает, как изображение выравнивается относительно области вывода по вертикали. Значение 0 указывает, что выравнивание происходит поо верхнему краю области вывода, значение 1 – по нижнему краю, значение 2 – по центру области. Применимо для слоев типа **image** и **still**.

vflip

Команда **vflip** переворачивает изображение по вертикали. Связано с тем, что некоторые файлы в формате TGA используют загадочную привязку левого верхнего угла картинки.

orig template.layer

Команда **orig** указывает, что координаты прямоугольника текущего слоя заданы относительно левого верхнего угла прямоугольника того слоя, на который ссылается команда. Это оператор должен быть указан после команды **rect**.

fldo field_order
field field_order

Задает порядок полей для анимации. Значение 0 указывает, что в файле анимации первым идет четное поле, затем нечетное (синонимы -- Upper field first = Top field first = even = field order B). Значение 1 указывает обратный порядок полей -- Lower field first = Bottom field first = odd = field order A.

loop number | max

Количество повторений анимации. Значение 0 или –1 задает бесконечное количество повторений. **colr** (number | **rgb**=number | **yuv**=number)

color (number | rgb=number | yuv=number)

Цвет слоя для слоев типа **solid**. Если число начинается со знаков **0xNNNNNNNN** то цвет задан шестнадцатеричным числом, в противном случае -- десятичным.

play

Начальное состояние слоя часов – запущены (так, как будто была выполнена команда **clock * play**). Вообще говоря, не имеет смысла для таймеров с обратным отсчетом.

pause

Начальное состояние слоя movie – пауза (так, как будто была выполнена команда **movie * pause**).

Примеры описания слоев в шаблоне

```
template test // имя шаблона
layer   // заливка прямоугольника сплошным цветом
        name цвет   // имя слоя
        type solid   // тип слоя
        rect 50 100 200 50  // положение и размеры прямоугольника
        color 0xffff0000  // цвет прямоугольника (красный)
        trsp 128   // прозрачный на 50%
endlayer
```

Допустимы любые команды слоя: fadein, fadeout, show, hide, pos, hslide, vslide, crawl, roll. Команда subst позволяет изменить цвет прямоугольника. Например:

```
subst test.цвет 0xff00ff00
```

изменит цвет прямоугольника на зеленый.

```
layer    // изображение
        name    картинка  // имя слоя
        type    image   // тип слоя
        rect    50 50 400 250  // положение и размеры прямоугольника
        file    "c:\images\try.tga" // файл с картинкой
endlayer
```

Допустимы любые команды слоя: fadein, fadeout, show, hide, pos, hslide, vslide, crawl, roll. Команда subst позволяет изменить содержимое прямоугольника. Например:

```
subst test.картинка "c:\images\new.tga"
```

изменит содержимое прямоугольника на содержимое файла new.tga.

```
layer    // изображение в формате DirectX
        name    картинка  // имя слоя
        type    still   // тип слоя
        rect    50 50 400 250  // положение и размеры прямоугольника
        file    "c:\images\try.jpg" // файл с картинкой
endlayer
```

Статические изображения в могут содержать в себе альфа-канал (слой прозрачности). Если нужно слой прозрачности игнорировать, можно использовать оператор alpha = NN, где NN может принимать значения от 0 (полностью прозрачная картинка) до 255 (полностью непрозрачная картинка).

```
layer   // анимация с альфа-каналом
        name "аним лого"  // имя слоя
        type movie   // тип слоя
        rect 50 50 320 240  // положение и размеры прямоугольника
        file "c:\movie\alogo.vim" // файл с картинкой
        field 1    // порядок полей – нечет
        loop max    // повторять до бесконечности
        pri max    // самый верхний слой
endlayer
```

Допустимы любые команды слоя: fadein, fadeout, show, hide, pos, hslide, vslide, crawl, roll. Команда subst позволяет изменить выполняющуюся в данный момент анимацию. Например:

```
subst test."аним лого" "c:\movie\new_logo.tga"
```

изменит содержимое прямоугольника на содержимое файла new_logo.tga.

```
layer   // часы (таймер и т.д.)
        name clock   // имя слоя
```

```
        type mpeg   // тип слоя
        rect 50 30 100 50  // положение и размеры прямоугольника
        file "c:\clocks\dig.clk" // файл с описанием часов
        pri max   // самый верхний слой
endlayer
```

Допустимы любые команды слоя: **fadein**, **fadeout**, **show**, **hide**, **pos**, **hslide**, **vslide**, **crawl**, **roll**. Команда **subst** выполняет замену часов (таймера) в соответствии с новым описанием часов.

```
subst test.clock "c:\clocks\analog.clk"
```

Для часов есть дополнительная команда, управляющая состоянием часов/таймеров.

clock test.clock **play** -- включить часы (таймер)

clock test.clock **stop** -- выключить часы (таймер)

clock test.clock **pause** -- остановить отсчет таймера

clock test.clock **resume** -- возобновить отсчет таймера

clock test.clock **countdown** msec -- установить значение обратного осчета для таймера (в миллисекундах)

```
layer  // диаграмма
        name ch
        type chart
        rect 100 100 500 200
        color 0xff0000ff // цвет (синий)
        size 2   // толщина линии (0 – заливка)
        focus "dot.tga" // отмечать конечную точку кривой картинкой
        shape 0  // форма 0 – ломаная, 1 – кривая, 2 - столбики
endlayer
```

Допустимы любые команды слоя: **fadein**, **fadeout**, **show**, **hide**, **pos**, **hslide**, **vslide**, **crawl**, **roll**. Команда **subst** для диаграмм неприменима. Дополнительная команда chart управляет состоянием вывода диаграмм:

chart test.ch **reset** -- сбросить количество точек на диаграмме в 0

chart test.ch **render** -- отрисовать текущее состояние диаграммы

chart test.ch **add** 4 0 100 40 50 80 150 120 80

-- добавить к диаграмме 4 точки, первая точка X = 0, Y = 100, вторая X = 40, Y = 50 и т.д. Все координаты указываются относительно прямоугольника слоя. Начало системы координат левый верхний угол прямоугольника.

Например, такая последовательность команд:

```
        chart test.ch reset
        chart test.ch add 4 0 100 40 50 80 150 120 80
        chart test.ch render
        chart test.ch add 1 160 120
        chart test.ch render
```

сначала отрисует ломаную линию из четырех точек, затем добавит еще одну и отрисует результирующую ломаную из пяти точек.

Операторы описания текстовых слоев

Описание стилей могут располагаться в любом месте файла до использования ссылки на стиль.

```
style    style_name
        команда описания стиля
```

команда описания стиля

endstyle

paragraph style_name
Ссылка на стиль. Последующий текст приобретает все атрибуты стиля.

font font_name
Задает имя шрифта.

face num
Задает тип шрифта. 0 – обычный текст (plain), 1 – жирный (bold), 2 – наклонный (italic), 3 – жирный наклонный (bold italic)

caps num
Способ применения заглавных литер. 0 – обычные литеры, 1 – small caps, 2 – initial caps, 3 – all caps

size num
Задает размер шрифта.

chsoft num
Задает размытый край у литер.

chcolor num
Задает цвет литер.

shsoft num
Задает размытый край у тени.

shcolor num
Задает цвет тени.

shxoff num
Задает смещение тени по X.

shyoff num
Задает смещение тени по Y.

bdsoft num
Задает размытый край у бордюра.

bdcolor num
Задает цвет бордюра.

bdsize num
Задает толщину бордюра.

tracking num
Задает межлитерное расстояние.

scale num
Задает горизонтальный масштаб.

align num
Выравнивание по горизонтали. 0 – влево, 1 – вправо, 2 – центрировать, 3 – выровнять по обеим краям.

leading num
Межстрочное расстояние.

base num
Сдвиг базовой линии.

wrap num
Если 1 – выполнять перенос строк если текст не помещается в прямоугольник. Перенос выполняется по границе слова.

Примеры описания текстовых слоев в шаблоне

Вообще говоря, если не задано описание стиля или описание формата текста соответствующими командами, то для форматирования текста используется стиль «по умолчанию» с такими атрибутами: шрифт = Arial, размер = 24, цвет = белый непрозрачный, выравнивание = по левому краю.

```
layer  // текстовый слой
        name txt1  // имя слоя
        type text  // тип слоя
        rect 0 0 400 35  // положение и размеры прямоугольника
        text Тестовая строка
endlayer// формирует текст Arial, 24, белый, выравнивание влево
layer  // текстовый слой
        name txt2  // имя слоя
        type text  // тип слоя
        rect 0 0 400 35  // положение и размеры прямоугольника
        text Тестовая строка
        bdsize 2  // использовать окантовку шириной 2 пиксела
        bdsoft 2  // с «размытым» краем
        bdcolor 0xff000000  // цвет черный
endlayer
```

Пример использования описания стиля:

```
style my  // определяем стиль
        font Times New Roman // шрифт
        face 1   // начертание - жирное
        size 40  // размер 40 пикселов
        chcolor 0xffffff00  // цвет литер - желтый
        bdcolor 0xff000000  // окантовка - черный
        bdsize 2  // размер ркантовки 2 пиксела
        bdsoft 2  // «мягкий» край
endstyle
layer
        name txt
        type text
        rect 100 50 500 200
        paragraph my  // использовать для форматирования стиль
        align 2  // выравнивать по центру
        text Привет, world
endlayer
```

В командах подстановки для текстовых слоев необходимо указывать новый текст:

subst test.txt А это новый текст вместо старого

Описание эффектов

Описание эффектов используется для взаимодействия с редактором шаблонов. Интерпретатор эти операторы не использует. Описания эффектов должны следовать непосредственно за оператором задания имени шаблона.

$efx effect_name key modifier

Задает имя эффекта effect_name (если используются пробелы в имени, то в двойных кавычках). Параметр key задает десятичный код «горячей» клавиши, присвоенной эффекту, а параметр modifier десятичное значение модификатора CTRL, ALT или SHIFT.

$cmd delay command

Описание команд, формирующих эффект. Параметр delay задает задержку команды в кадрах относительно начала эффекта.

Пример организации связи

```
// start telnet client
//  if name == NULL -- connect to localhost
//  if name == hostname -- try to find host by name
//  if name == N.N.N.N -- try to find host by addr
SOCKET
tnc_start(char *name, int port) {
        WSADATA wsaData;
        struct hostent *hp;
        unsigned int addr;
        char *server_name;
        struct sockaddr_in server;
        SOCKET  conn_socket;
        int socket_type = SOCK_STREAM;

        if (name == NULL)
                server_name = "localhost";
        else
                server_name = name;

        if (WSAStartup(0x202,&wsaData) == SOCKET_ERROR) {
                WSACleanup();
                return NULL;
        }

        // Attempt to detect if we should call gethostbyname() or
        // gethostbyaddr()
       if (isalpha(server_name[0])) {   /* server address is a name */
                hp = gethostbyname(server_name);
        }
        else  { /* Convert nnn.nnn address to a usable one */
                addr = inet_addr(server_name);
                hp = gethostbyaddr((char *)&addr,4,AF_INET);
        }
        if (hp == NULL ) {
                WSACleanup();
                return NULL;
        }

        // Copy the resolved information into the sockaddr_in structure
        memset(&server,0,sizeof(server));
        memcpy(&(server.sin_addr),hp->h_addr,hp->h_length);
        server.sin_family = hp->h_addrtype;
        server.sin_port = htons(port);

        conn_socket = socket(AF_INET,socket_type,IPPROTO_TCP); // Open a socket
        if (conn_socket < 0) {
                WSACleanup();
                return NULL;
        }

       // Notice that nothing in this code is specific to whether we
        // are using UDP or TCP.
```

```
        // We achieve this by using a simple trick.
        //    When connect() is called on a datagram socket, it does not
        //    actually establish the connection as a stream (TCP) socket
        //    would. Instead, TCP/IP establishes the remote half of the
        //    ( LocalIPAddress, LocalPort, RemoteIP, RemotePort) mapping.
        //    This enables us to use send() and recv() on datagram sockets,
        //    instead of recvfrom() and sendto()
        if (connect(conn_socket,(struct sockaddr*)&server,sizeof(server)) == SOCKET_ERROR) {
                WSACleanup();
                return NULL;
        }
        return conn_socket;
}

// stop telnet client
void
tnc_stop(SOCKET socket) {
        closesocket(socket);
        WSACleanup();
}

// send data from client to server
int
tnc_send(SOCKET socket, char* buf) {
        int retval;
        char msgbuf[TELNET_MAXMSG];
        char* bp = buf;
        char prev = 0;
        int msglen = 0;

        if (tnc == NULL) return 0;
        if (bp == NULL) return 0;

        // translate buf into msgbuf
        while (*bp != 0) {
                if (*bp == '\n' && prev != '\r') {
                        msgbuf[msglen++] = '\r';
                }
                msgbuf[msglen++] = prev = *bp++;
        }
        if (prev != '\n') {
                msgbuf[msglen++] = '\r';
                msgbuf[msglen++] = '\n';
        }
        msgbuf[msglen++] = 0;

        retval = send(socket, msgbuf, msglen, 0);
        if (retval == SOCKET_ERROR)
                return -1;

        return retval;
}
```