

Vortex

Vortex

- Thunderstone's Taxis Database
- Metamorph Full-Text Engine
- Vortex Web Script Language
- Built-in Web Server

Technical Details

- Installation location
- Server Side
- Script location
- Auto-compilation

vhttpd

- Performance
 - integrated Vortex
 - leaner server
- entry/exit scripts
- secures scripts

Script Syntax

- HTML-like
- Functions
- Control statements
- Variables
- Database access
- Error handling

Script Example

```
<SCRIPT LANGUAGE=vortex>

<A NAME=main> <!-- Function declaration -->
  <BODY BGCOLOR=white> <!-- HTML output -->
    Hello world!
    My address is: $REMOTE_ADDR <!-- Var -->
  </BODY>
</A>

</SCRIPT>
```

URL Syntax

- CGI Basics
- SCRIPT_NAME, PATH_TRANSLATED
 - default script
- \$url
- EXPORT
- \$userpath

User Input

- HTML Forms
- CGI variables become Vortex variables
- GET or POST identical
- Cookies also available

Variables

- Assigning constants
- SQL Math expressions
- Displaying
 - HTML escaped, single vs. multiple value
- Assigned from functions or SQL
- Must exist in script
- State retention via EXPORT

Multi-value Variables

- All variables can hold multiple variables
- <options>, <checkbox>
- Vortex functions operate on lists of values
- Use <loop> to get at individual members
- Passed into SQL as Metamorph Set

Multi-value Form Variables

```
<$colors = red orange yellow green
           blue purple brown black>
<FORM METHOD=post ACTION=$url/display.html>
  Pick some colors:
  <SELECT NAME=user_colors MULTIPLE>
    <options $colors $user_colors $colors>
  </SELECT>
  <INPUT TYPE=submit>
</FORM>
```

<loop>

- Iterates over variables
- MAX= and SKIP=
- sets \$loop and \$next
- <sql>, <rex>, <import>, <fetch>, <xtree>

Variable Types

- Most functions return strings
- CGI and Literals are either String or Integer
- Variables from SQL keep their type
- Auto cast from left to right

Variable Scope

- Global and Local Scopes exist
- Default is Global
- Locals for function parameters
- Local if explicitly declared <LOCAL>
- Recursion
- Overlap

EXPORT Types

- TABLE for large data
- URL for small data that might change with each HREF generated
- QUERY for clear text export in query string
- USEROK to allow form variables to override state

Variable Initialization Order

- URL State
- State Table
- Command Line
- Environment
- HTTP Cookie
- Query String
- POST Content

Variable Information

- Taxis -dump
- <varinfo>
 - list
 - size
 - type
 - filename (MIME upload)
 - contenttype

Vortex Functions

- Script
- Library
- Builtin
- User

Function Parameters

- Declaring
- Default Values
- Modifying Parameters
- Calling Types, by-value and by-reference

Function Scope

- Defines Visibility of function
- Should always be specified
- PUBLIC, EXPORT, PRIVATE
- Default is PRIVATE

Common Tasks

- Common code
- Embedding Javascript and literal HTML
- Generating HTML Forms
- SQL Queries
- Pagination
- Hit highlighting

Common Code

- Declare Functions
 - <look> </look>
 - <verifyuser>
 - <resultfont>
 - should be <PRIVATE>
- Libraries
 - functions shared across applications
 - should be <EXPORT>

Literal HTML

- Use <VERB NOESC> </VERB>
- Useful for Javascript
- Body can be outside <VERB> for variable usage

Generating HTML Forms

- Special functions
 - checkbox, options, radiobutton
 - automatically select appropriate values
- \$url for ACTION= to maintain state
- Image Maps, client-side, server-side
- Vortex variables follow NAME= on form

SQL Queries

- Embedded variables parameters
- \$null to drop clauses from WHERE
- Variables assigned from column names
 - column aliasing
- ROW, SKIP, MAX, NOVARS
- \$loop, \$next, \$indexcount, \$rows.min, \$rows.max

Pagination

- Queries often return a lot of results
- <pagelinks> function designed to make it easy to generate page links
- default format
- can be customized

Pagelinks options

- SKIPVAR
- NUMROWS
- SUMFUNC
- ORDER
- PREVFUNC
- NEXTFUNC
- PGFUNC
- PGSZ
- MAXPGS

Hit Highlighting

- <fmt>, <strfmt>, <fmtcp>, <mm>
- bold or links
- automatic @0 w/.
- Respects query settings

Interesting Tasks

- Fetching pages
- Parsing data
- Search and Replace
- Geographic Searches
- User verification

Fetching Pages

- For proxy, metasearch, getting data
- <fetch> and <submit>
 - GET and POST
 - FTP put and get
- <fetch PARALLEL=n>
 - loopvars

Fetch continued

- \$ret contains the HTML source
- <urlinks>
- <urltext>
- <urlinfo>
 - title, contenttype, errmsg, metaname &c

Fetch continued

- <fetch> is controlled with <urlcp>
 - timeout
 - maxpgsize
 - user, pass
 - proxy
 - etc

Parsing Data

- Several useful functions
 - <rex> - extract a well defined piece of data
 - <split> - split the input into multiple values
 - <sandr> - search and replace
 - <timport> - general purpose data import
 - <read>, <readln> - reading a file

Search and Replace

- <sandr>
 - single variable
- <fmtcp SANDBLAST>
 - many variables
 - automatic
- <fmtcp SANDCALL>
 - dynamic replace strings
 - function callbacks

Geographic Searches

- Zip Code is a keyword search
 - fast for single zip code
 - unreliable for finding nearby
 - limited to US
- <geocode> provides for lon, lat searches
 - arbitrary sized area
 - more data stored

Geo Searches Zip Example

- Text\Zip like 'query 12345'
 - exact zip match
- Text\Zip like 'query 123*'
 - prefix match
- Text\Zip like 'query 12345' and Zip like '12345'
 - verifying hit location

Geo2code Example

```
<sql MAX=1 "SELECT lat, lon FROM city
WHERE Place\State LIKE $City
ORDER BY Pop DESC">
<geo2code $lat $lon 10000>
<sql "SELECT Name FROM docs
WHERE Text LIKE $query
AND Loc BETWEEN " $ret>
</sql>
</sql>
```

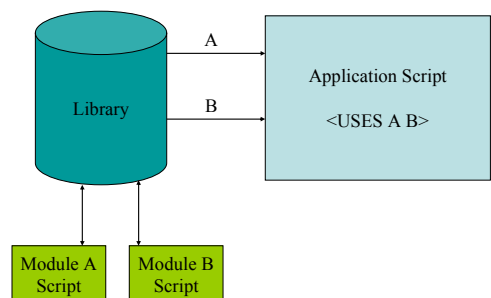
Controlled Access Application

- User/password login
 - once at session start
- Multiple users
- Multiple access levels
 - search, edit, admin
- Web-based user admin

Library Modules

- Modularity
- Code sharing
- Dependency checking
- Revision Control
- Advisory Locks

Libraries



Module Code

- Looks like any other Vortex Script
- <main> not required
- function scope EXPORT
- PRIVATE functions for utility functions

Library functions

- Options to Taxis
 - -ci -log -force
 - -co -date -rev
 - -lock -unlock
 - -module
 - -listlib -listrev -wipelib
 - -del -targets

Using Library Functions

- <USES [DATE=date] module [module...]>
- Only one USES allowed
- Module can use another module
- DATE argument includes time
- Can be relative, e.g. DATE="last Saturday"
- Name collisions except on PRIVATE

Common Mistakes 1

- Building SQL, use \$variable as literals, not parameters
- Disabling query protection as the easiest way to make the query work
- Using SQL logic when Metamorph logic is more appropriate

Common Mistakes 2

- Passing a multi-valued variable into SQL
- Complicated data parsing
- Inappropriate EXPORT Type
- Not having variables set during EXPORT
- Not clearing URL exported variables

Common Mistakes 3

- Too much, too little whitespace generated
 - Vortex strips leading whitespace on a line
 - Does not generate newline after function
- Misspell or wrong case on function names
- Not quoting \$'varname'.

Useful Tips

- If you are executing a lot of SQL statements in a loop, check the SQLCACHE setting
- Look at the HTML source or vortex.log if you have problems
- entry/exit scripts
- Vortex is designed for web sites. Many features are built in. If you don't see it, ask.

Useful Tips 2

- Declare function scope
 - Lowest possible
- Use common look and feel functions
- Can use <sum> for CSV type lists
- Append all values, then <sum>
- <sum> faster than SQL Math

Useful Tips 3

- Use <fmt %s \$binvar> to send binary data
- <capture></capture> lets you grab data for caching or processing
- <write> for writing to a file
- <exec> allows arbitrary execution
- <header> cookies, headers - browser control

Useful Tips 4

- Use ROW to save memory
 - sql, timport, xtree, rex
- UPDATE/DELETE/INSERT return ROWS
 - NOVARS to disable
- <LOCAL> for temp variables
 - protects/hides outer vars
- <xtree> in memory string index