

Pipeline Web CGI Techniques

VM 3C8

by Will Roden

A world map is visible in the background of the slide, rendered in a light blue color against the dark blue background.

Agenda

- Your Web and Pipelines Level?
- Demo of Web Site
- Intro to HTML, Pipelines, and CGI Scripts
- Pipeline Techniques
 - One to One
 - One to Many
 - Many to Many with REXX
 - PIPCMD
 - Many to Many with PIPCMD
- How to test/debug a CGI Script

Where are You?

- Who has NOT written a Web Page?
- Who has NOT written a Pipeline?
 - Who has written their own Pipeline Stage?
- Who has written JavaScript?
- Who has written a CGI Script?
- What web server are you using?

Web Basics

```
<!doctype html public "html2.0">
<html>
  <head>
    <title>.....</title>
  </head>
  <body>
    <center><h2>...heading...</h2></center>
    <form ....>
      Headings, Text, Fields, Tables, ...
    </form>
  </body>
</html>
```

Web Details

Text - your text

Break - `
` - place on the next line

Paragraph - `<p>` - skips a line

Forms

```
<form method=POST action="movie.cgi" name="aForm">
  <Input type="radio" name="TYPE" value="A">
  <Input type="text" name="FIND" value=""
    size=40 maxlength=99>
  <Input type="button" value=" Find " onClick="Find(this)">
  <Input type="hidden" name="PUSHED" value="find">
</form>
```

Web Tables

```
<table>
<tr>
  <th> table heading </th>
  <td> table detail </td>
</tr>
<tr>
  <td>...</td>
  <td>...</td>
</tr>
</table>
```

table heading	table detail
...	...

JavaScript

- NOT Java
- Written between `<script>` and `</script>` tags in an HTML file
- Adds Dynamic Function to HTML
- Sample Functions:
 - `if (...) {...} else {...}`
 - `document.form.obj_name.value="x"`
 - `document.form.submit()`
 - `alert("your msg")`
 - `onClick()`
 - `onSubmit()`

Pipeline Basics - Input/Output

Input Sample - Movie List

```
a  Indiana Jones and the Temple of Doom
h  Night of the Living Dead
h  Psycho
```

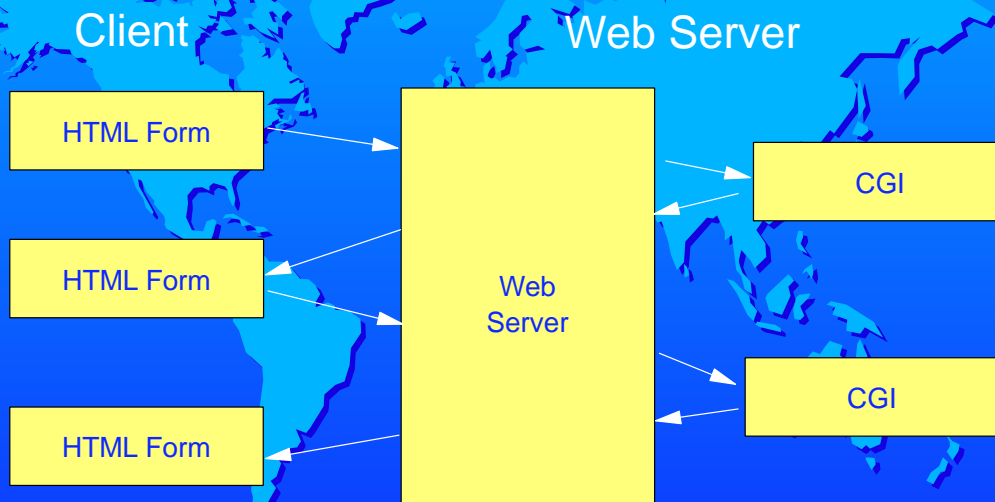
Output - HTML Portion

```
<select name="FILM" width=50 size=3>
<option>Night of the Living Dead
<option>Psycho
</select>
```

Pipeline Basics

```
'PIPE (endchar ? name Pipe1)',  
| < movie list',  
| locate 1-6 /h',  
| spec /<option>/ 1 7-49 n',  
| literal <select name="FILM" width=50 size=3>',  
| append literal </select>',  
| console'
```

CGI Basics



CGI Basics

- `<form>` tag specifies CGI to run
- `<input>` tags specify names and values
- names and values can be obtained by CGI
- CGI's output is an entire HTML page
- Strategies:
 - create entire HTML from scratch
 - read in a template and make changes
 - pass values from one CGI to the next using `<input type="hidden" ...>` statements

How CGI Works

- Shadow (EnterpriseWeb/VM) - Neon Sys (Beyond)
 - ✓ CGI is a pipeline stage
 - ✓ primary input stream - name=value
 - ✓ primary output stream - HTML page
- ESAWEB - Velocity Software
 - ✓ CGI is a program (EXEC, Assembler, PL1)
 - ✓ name=value is created in by a call
 - ✓ call Output with HTML

How CGI Works

- IBI - Information Builders
 - ✓ CGI is a program (EXEC, C++, ...)
 - ✓ name=value is created in EXEC
 - ✓ say output HTML information
- VM:Webgateway - CA (Sterling)
 - ✓ CGI is an EXEC
 - ✓ EXEC calls to get name=value
 - ✓ EXEC calls to provide HTML page

1:1 Pipeline Technique

HTML Target

```
...  
<!--SUB_Date -->  
...
```

CGI Code Sample

```
...
```

```
'change /<!-- SUB_Date -->/today/'
```

```
...
```

1:1 Pipeline Technique

HTML Target

```
...  
<!--SUB_Date -->  
...
```

CGI Code Sample

```
...  
'| change /<!-- SUB_Date -->/'today''  
...
```

1:M Pipeline Technique

HTML Target: <!option selected>

!:totarget locate /<!option /

!:totarget locate /<!option /

```
t:take 1  
specs /movie list/ 1  
getfile  
locate 1.6 /h/  
spec /<option>/ 1 7-49 n  
literal <select name="FILM" width=50  
size=3>  
append literal </select>
```

f:fanin

1:M Pipeline Technique

HTML Target: <!option selected>

Sample Pipeline Code:

```
| !:totarget locate /<!option /',
| f:fanin',
| '? !:',
| t:take 1',
| specs /movie list/ 1',
| getfile',
| locate 1.6 /h/',
| spec /<option>/ 1 7-49 n',
| literal <select name="FILM" width=50 size=3>',
| append literal </select>',
| f:?t:|f:'
```

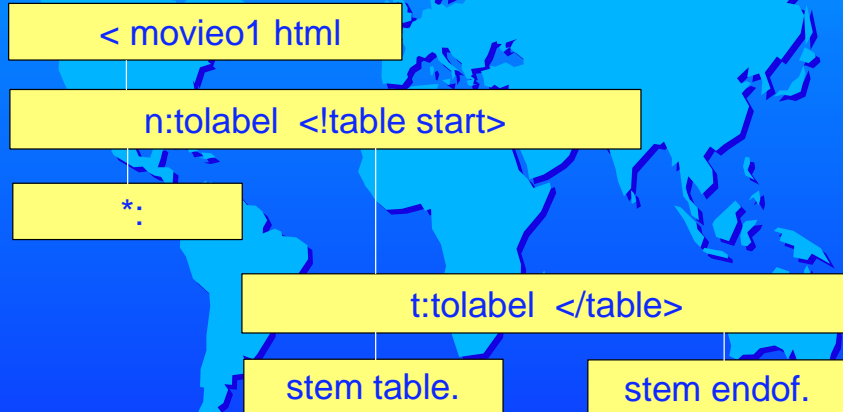
M:M Pipeline Technique - REXX

HTML Target:

```
<!table start>
<tr>
<td>SUB_1</td>
<td></td>
<td></td>
<td>SUB_2</td>
<td></td>
<td></td>
<td>SUB_3</td>
</tr>
</table>
```

M:M Pipeline Technique

Get the Template:



M:M Pipeline Technique

Get the Template:

```
'CALLPIPE (endchar ? name OrderAll)',  
  '< movieo1 html',  
  '| n:tolabel <!table start>||',  
  '| *:',  
  
  '? n:',  
  '| t:tolabel </table>||',  
  '| stem table.',  
  
  '? t:',  
  '| stem endof.'
```

M:M Pipeline Technique

Change one for each Movie

```
do while rc = 0
  'PEEKTO in '
  if (rc<>0) then leave
  name = substr(in,7,42)
  star = substr(in,50,15)
  by = substr(in,65)
  'callpipe stem table.',
    '| change /SUB_1/'name'',
    '| change /SUB_2/'star'',
    '| change /SUB_3/'by'',
    '|*.'
  'READTO'
end
```

M:M Pipeline Technique

Finish the Template:

```
'CALLPIPE (name EndOf)',
  '| stem endof.',
  '|*.'
Exit
```

PIPCMD and In/OutStore

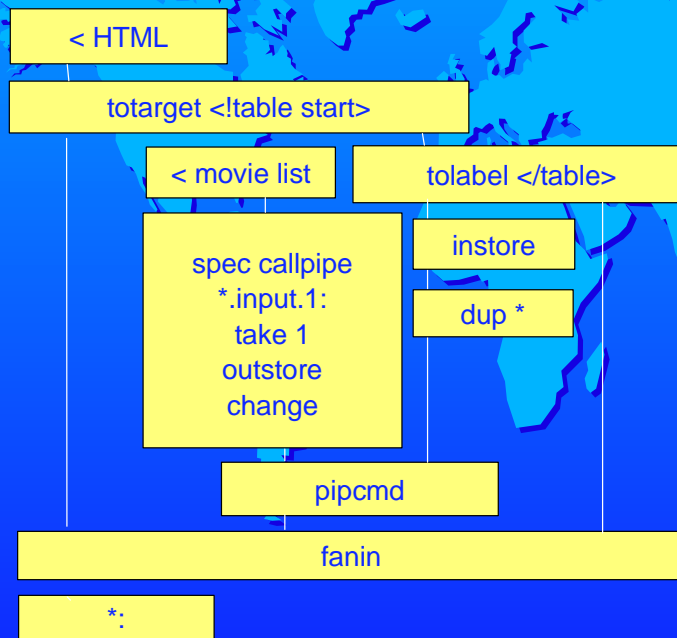
- PIPCMD Issues Pipeline Subcommands
 - i.e. callpipe
- InStore stores records for OutStore to retrieve
 - InStore write a record that tells OutStore where to find the treasure

M:M Pipeline Technique

HTML Target:

```
<!table start>
<tr>
<td>SUB_1</td>
<td></td>
<td></td>
<td>SUB_2</td>
<td></td>
<td></td>
<td>SUB_3</td>
</tr>
</table>
```

M:M Pipeline Technique - PIPCMD



M:M Pipeline Technique - PIPCMD

```
'CALLPIPE (endchar ? name OrderAll)',  
' < movieo1 html',  
'|n:todel <!table start>||',  
'|f:fanin',  
'|*.*',  
'|*.*',
```

M:M Pipeline Technique - PIPCMD

```
'? < movie list',  
'| locate 1-6 /"type"/',  
'| locate anycase 7-49 /"find"/',  
'| spec @callpipe *.input.1:||take 1||outstore@ 1',  
  '@||change /SUB_1/@ n 7-49 n @/@ n',  
  '@||change /SUB_2/@ n 50-64 n @/@ n',  
  '@||change /SUB_3/@ n 65-* n @/@ n',  
  '@||*:@ n',  
'|p:picmd',  
'|f:',
```

M:M Pipeline Technique - PIPCMD

```
'?n:',  
'|t:tolabel </table>'||,  
'| instore',  
'| dup *',  
'|p:',  
'?t:',  
'|f:'
```

Debugging Techniques

- Watch Web Server's console
- Write EXEC to run CGI
- Issue CMS TELL commands
- Write a file and use SendFile
- Add output to web page
- CP SP CONS

CGI Testing

Testing Find

```
'pipe',  
'| literal pushed=Find',  
'| literal type=a',  
'| literal find=',  
'| rexx (movie1 cgi)',  
'| > movie1 out a',
```

CGI Testing

Testing OrderAll

```
'pipe',  
'| literal pushed=OrderAll',  
'| literal type=a',  
'| literal find=Big',  
'| rexx (movie1 cgi)',  
'| frtarget locate /<!-- BEGIN CONTENT -->/',  
'| totarget locate /<!-- END CONTENT -->/',  
'| console'
```

How to Contact Will Roden

roden@us.ibm.com
(607) 752-5065
<http://www.vm.ibm.com/devpages/roden/>

The MOVIE Example:

<http://www.vm.ibm.com/devpages/roden/movie1.html>