Choices for Integrating

RPG with the web

Comparing PHP, Java and CGIDEV2

Presented by

Scott Klement

http://www.scottklement.com

"There are 10 types of people in the world.
Those who understand binary, and those who don't."

Objectives of the April 2008 Issue

Every issue of System iNEWS has an "in-depth" topic, a subject that's the focus of the issue. (A "cluster" topic as we call it, internally.)

- For those who want to get to the web, but don't know the right method. Focus on the most popular, free alternatives:
  - ✓Java (Don Denoncourt)
  - ✓PHP (Tony Cairns)
  - ✓ILE RPG with CGIDEV2 (Paul Tuohy)
  - ✓(Groovy, added later.)
- Authors were recruited, one for each of the above languages.
  - ✓Don Denoncourt (Java author) chose to show Groovy as well.
- Use same business logic module, written in RPG
  - ✓Implement as ILE service program
  - ✓Author was Scott Klement

M V C

One of the most important concept in engineering software for the future is Model-View-Controller (or MVC).

- M = Model

  The business rules and the program logic to implement them.

  Should be designed as a set of re-usable "services"

  In this cluster, the model was implement as an ILE RPG service program.

- V = View

  The user interface, and the program logic to implement it.

  Four of them, RPG, PHP, Java (JSF) and Groovy.

- C = Controller

  Logic that controls the flow of the application.

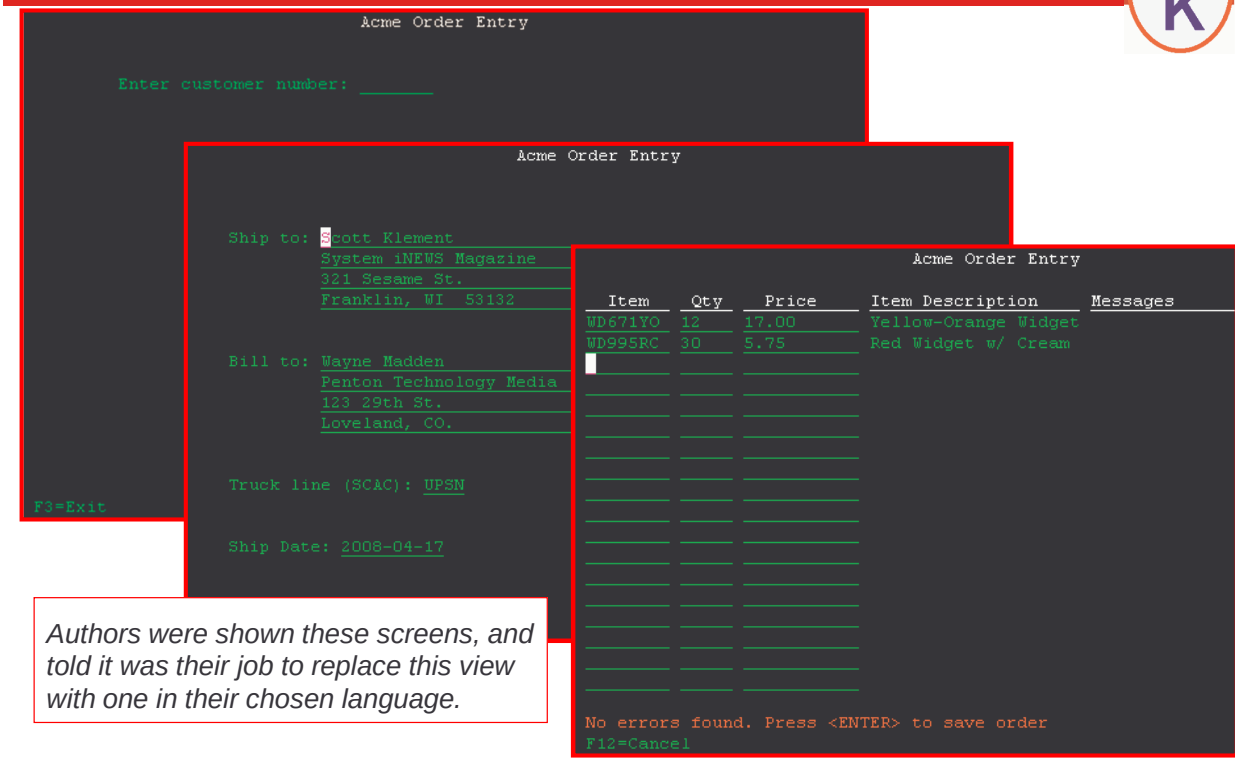
  Calls the routines in the Model & View

Model -- Business Logic

Provide an application sophisticated enough to resemble what we write in our day-to-day jobs, but simple enough to be explained in a magazine article.

- Implemented as an ILE RPG service program
- Important design factors
  - Callable from anywhere
  - Encapsulated.
  - Service-oriented
  - Upgradeable.
  - Maintainable / Agile
  - Stateless
- Use wrappers for stored procedures
  - Let ILE programs take advantage of ILE
  - Let other programs take advantage of SQL
  - Use result sets for all output parms.

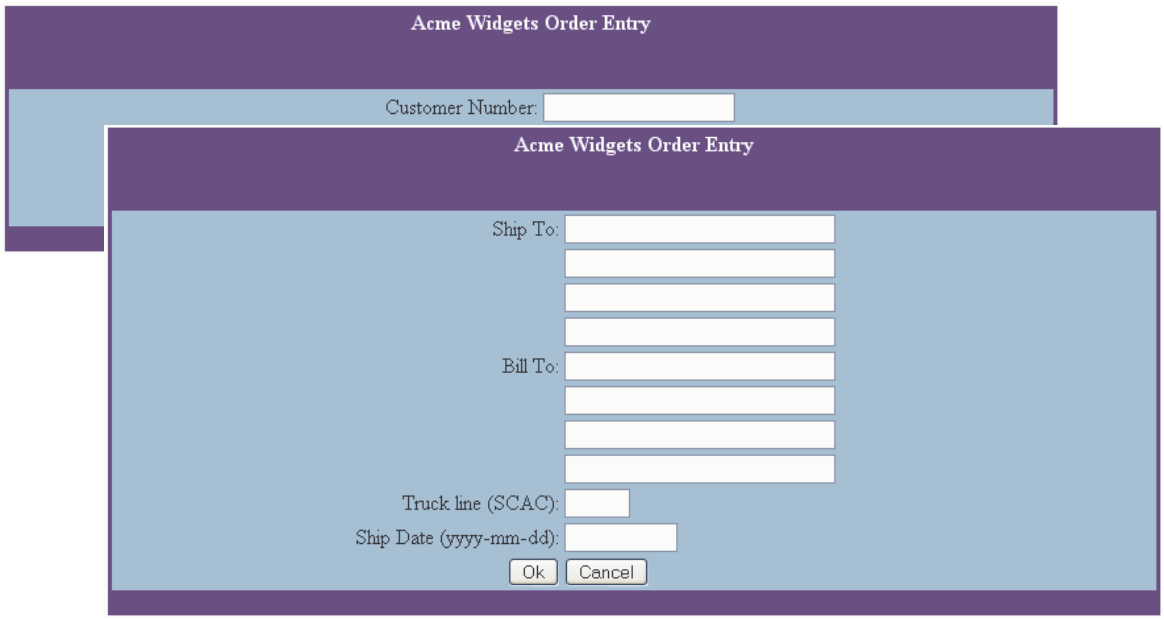
A Traditional View



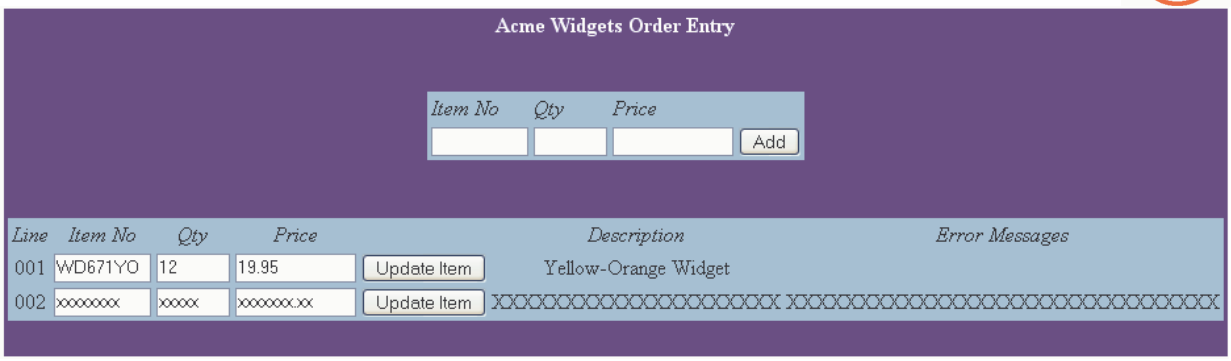
Authors were shown these screens, and told it was their job to replace this view with one in their chosen language.

The Web View (1 of 2)

The authors were all given the same HTML screens, designed by Scott Klement. (They are kept simple – I'm not a graphic artist!)



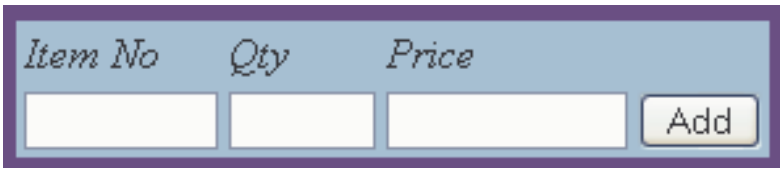
The Web View (2 of 2)



Each author was given the same RPG model (back-end business logic) and the same HTML files, so the languages could be compared on their own merits (not on which author could design a prettier screen)

- *Show how to implement the HTML in their language*
- *Show how to implement the controller logic in their language*
- *Show how to call the RPG business logic in their language.*

Statelessness of Web Apps



Each time you click a button, or follow a link, in an web application, the browser creates a new request to the HTTP server.

- In the case of a web app – that tells the HTTP server to run a program.
- It's basically a CALL PGM(MYWEBPGM) command.
- The program is passed all of the input (form) information.
- The program processes this input, and writes out a new web page.
- When the program ends, the page is displayed in the browser.

The only time your program runs is in-between the time a link is clicked, and the time it takes to display the next screen.

This means that your program cannot control the flow of screens like we are used to in traditional RPG. Instead, the browser controls them, and a new call is made with every button press, or click of a link.

Statelessness Challenges

Due to the "statelessness" of having a separate, individual call for each click, the following challenges have to be overcome:

- Your program doesn't know if the user will click the next click, or hit the "back button", or close the browser.
- Since you may have many users clicking links at the same time, you never know if the next call of your program will be from the same user running the same "job" as the last call.
- This means you can't remember variable values from call to call.
- In fact, you can't (reliably) set up anything in a previous call that will be used in subsequent calls.

To solve the problem, you issue a "session number" (a unique number generated by your program) and you pass it back to the browser.

- The browser passes it back on each new click.
- You can use it to look up "persistent" information in a file.

A Note About CGI

Web programming with CGI has gained a bad reputation in today's world.

Here's why:

- Interpreting the input from the browser is complex.
- The output HTML has to be coded into your program
- You must manually escape your output data to be valid in HTML.
- A new "process" (or "job" in OS/400 terminology) must be launched for each call to your program.

Unfortunately, this reputation has somehow spread to CGIDEV2 on i5/OS. CGIDEV2 is not CGI programming! Rather, CGIDEV2 prevents you from having to do the low-level work that would be required in traditional CGI.

- It takes care of interpreting the data from the browser, so you don't have to.
- It lets you keep your HTML outside of your program code
- It takes care of escaping output for you.
- And, on i5/OS, the HTTP server has NEVER required starting a new process for each call to the program! You can load your program into an ILE activation group so that it remains in memory. In that case, subsequent calls are INCREDIBLY fast, because all you're doing is calling a routine in memory.

Therefore, CGIDEV2 should not be thought of as "CGI programming."

CGIDEV2

CGIDEV2 is not a programming language. It's a toolkit from IBM to simplify the job of writing a web application in native ILE RPG.

- Runs in the native, free HTTP server provided with i5/OS (Apache or Original)

It provides routines:

- To make it easier to output HTML to a browser.
- To make it easier to read input from fields filled in on a web page.

Writing HTML Output:

- Divide your HTML into sections (like "record formats" in DDS)
- Insert replacement values (like "fields" in DDS)
- RPG code can insert replacement values, then write the sections
- Sections can be written repeatedly, to repeat elements that go out to the browser.

Paul Tuohy used `<!-- $SECTIONNAME$ -->` and `<!-- %REPVAL% -->`

CGIDEV2 Input From Browser

```
If ZhbGetInput(SavedQryStr: QUSEC) > 0;

     PageId = ZhbGetVar('PageId');
     Action = %trim(ZhbGetVarUpper('action'));
     NextPage = PageId;

     if (PageId <> 'PAGE1');
        Id = %TimeStamp(ZhbGetVar('Id'));
        IdNum = %Int(ZhbGetVar('IdNum'));
     EndIf;

     Select;
        When Action = 'CANCEL';
           PERSIST_ReleaseThisOrder( id : IdNum);
           PageId = 'PAGE0';

        When (PageId <> 'PAGE1');
           if PERSIST_GetThisOrder( Id : IdNum : p_this );
              scErrMsg = 'Session has timed out - select a new order';
              PageId = 'PAGE0';
           EndIf;
     EndSl;
 EndIf;
```

- ZhbGetInput() loads all input from the browser into arrays inside the CGIDEV2 srvpgm
- ZhbGetVar() locates one variable in the arrays, and returns it's value to the program.
- All values are character strings.
- The PERSIST_xxx() routines are for saving/retrieving data that's saved from call to call.

CGIDEV2 Output (HTML Template)

```
<!-- $Header$ -->
Content-type: text/html

<!-- Standard Header -->
<html>
  <head>
     <title>ACME Widgets Order Entry</title>
  </head>
  <body>

     <form method="post" action="/cgi-bin/acmecgir4.pgm">

     <input type="hidden" name="id" value="<!-- %id% -->" />
     <input type="hidden" name="idnum" value="<!-- %idnum% -->" />

     <table border="0" width="800" bgcolor="blue">
     <tr><td align="center">
       <p><b><font color="white">Acme Widgets Order Entry</font></b></p>
       <br>
     </td></tr>

<!-- $Select$ -->
<!-- Customer/Order No Selection -->
     <tr><td>
         <input type="hidden" name="pageid" value="PAGE1" />
```


CGIDEV2 Output (RPG code)

```
  GetHTMLIFS('/acmecgi/acmecgir4.html':
             '<!-- $':'$ -- >':'<!-- %':'% -->');

  updHTMLVar('id':%char(id));
  updHTMLVar('idNum':%char(idNum));

  wrtSection('header');

  if scErrMsg <> *Blanks;
      updHTMLVar('message':scErrMsg);
      wrtSection('MessageLine');
  EndIf;

  Select;
     When NextPage = 'PAGE1';
        DisplayPage1();
     When NextPage = 'PAGE2';
        DisplayPage2();
     When NextPage = 'PAGE3';
        DisplayPage3();
     When NextPage = 'PAGE4';
        DisplayPage4();
  EndSl;

  wrtsection('footer *fini');
```

RPG Calling RPG

Since the code is regular RPG code, it can call the business model logic in the service program directly. (PHP and Java call the RPG routines as SQL stored procedures.)

```
if (this.Order <> *blanks);
  rc = Order_loadHeader(this.Order: Hdr);
else;
  rc = Order_new(this.Cust: Hdr);
  this.Order = Hdr.OrderNo;
endif;

if (rc = *OFF);
   scErrMsg = Order_error();
   return *OFF;
endif;

if (not Order_LoadItems( this.Order: this.Count: Item));
   scErrMsg = Order_error();
   return *OFF;
endif;
```


PHP

PHP is a script language – very much like the others used for writing Unix shell scripts (Bourne Shell, Korn Shell, Perl, QShell, etc). However, it was designed with a strong focus on being particularly suited for web programming.

- An interpreted (non-compiled) script language.
- Runs under Apache, but in a separate (PASE) instance.

It provides:

- Arrays (automatically populated) containing input from the browser.
- A very easy means of writing out HTML code.
- Hundreds of built-in functions.
- Thousands of additional functions (added as an extension)

Syntax:

- Any data typed into a PHP document is assumed to be HTML output that will be sent, as-is, to the browser.
- Anything insider <?php and ?> delimiters is PHP program code, and will be run, and it's output will be sent to the browser.
- Anything that starts with $ (dollar sign) is considered a variable.

PHP "Hello World"

To write PHP, you can embed it into the middle of an HTML document. Anything outside of the <?php and ?> tags is written to the browser as-is. The stuff between those tags is your PHP program code.

```
<html>
<body>

<?php

// You can put any PHP code here.

echo "Hello World";
$info='Yes!';
?>

<p>Nice day, isn't it? <?php echo $info;?></p>

</body>
</html>
```


PHP Input from Browser

```
<?php
  include_once("config.php");

  isset($_GET['action']) ? $action=$_GET['action'] : $action="getOrder";

  switch ($action) {
  case "getOrder":        // step 0
    $go="order.php";
    break;

  case "loadOrder":       // step 1
    if (!loadOrder())
      $go="index.php?action=getOrder";     // goto step 0
    else
      $go="index.php?action=editHeader";   // goto step 2
    break;

  case "editHeader":      // step 2
    $go="ship.php";
    break;

  case "editItems":       // step 3
    $go="items.php";
    break;
  .
  .
```

PHP Output Considerations

In any web application (in any language) it's considered good form to keep the HTML data separated from the program code.

- Those with talent designing displays aren't always good at programming
- Those with talent writing code aren't always good at designing displays.

As you have seen, it's very easy to mix the two in PHP. However, it's not necessary to do so. PHP has the capability of using "include" statements (very similar in nature to copy books that you'd code with /COPY in RPG) that let you put the HTML in a separate file.

PHP also provides "session" functions let let you save a data structure containing data that needs to be saved from call to call (similar to the PERSIST routines provided in the CGIDEV2 example).

PHP implements it's sessions under the covers, so you don't have to understand how they work. You just save data into the session, and restore it when you need it.

Setting PHP Output Variables

```
<?php

if ($_POST['action']) $_POST['action']="display";
elseif ($_POST['Help']) order_help();
elseif ($_POST['custno'] && $_POST['orderno'])
  $_POST['msg']='Please provide one or the other!';
elseif (!($_POST['custno'] || $_POST['orderno']))
  $_POST['msg']='You must provide either a customer or order number';

  .
  .

// include this screen html
include_once("acmeords1.html");
?>
```

In the code above, the author (Tony Cairns) is reading and setting some variables (in the $POST array) to values that he wants to display in the HTML.

The syntax $POST['keyname'] works a lot like a database, except that it's an array in a program. PHP will store data in the array with an associated key. You can then recall the value by looking up the key.

You could say it's like using %LOOKUP in ILE RPG.

PHP Displaying Output

This is what the "include file" (acmeords1.html) contents look like:

```
<html>
<body>
     <table border="0" width="800" bgcolor="blue">
     <tr><td align="center">
       <p><b><font color="white">Acme Widgets Order Entry</font></b></p>
       <br>
     </td></tr>
     <tr><td>
          <form method="post" action="<?php echo($_SERVER['PHP_SELF']) ?>">

          <table border="0" width="100%" bgcolor="lightblue">
          <tr>
            <td align="right">Customer Number:</td>
            <td align="left"><input type="text" name="custno" maxlength="6" width="6"
            value="<?php echo($_POST['custno']);?>"/></td>
          </tr>
           .
           .
          <tr>
            <td align="center" colspan="2"><?php echo($_POST['msg']);?></td>
          </tr>
          </table>

          </form>
```

The "<?php echo" is used to fill in variable values in the HTML itself.

PHP Calling RPG

PHP calls the RPG subprocedures as stored procedures by connecting to the database and running the SQL CALL statement.

```
$host = "i.example.com";
$user = "klemscot";
$pass = "bigboy";
$opts = array("i5_naming"=>DB2_I5_NAMING_ON);

$conn = db2_pconnect( $host, $user, $pass, $opts );
if (!$conn) {
  set_ORDER_error("Failed: db2_pconnect($host,$user,$opts).");
  return FALSE;
}

$sql="CALL MYLIB/ORDER_NEW($custno)";
$stmt = db2_exec($conn, $sql);
$row=db2_fetch_assoc($stmt);

// results can now be found in fields like
//   $row['ORDERNO'], $row['CUSTNO'], $row['SHIPNAME'], etc.
```

Note: db2_pconnect() is for connections established once and re-used throughout all SQL code. Performs well, but might have security drawbacks.

There's also a db2_connect() that has to re-connect with every call to the script.

Java Server Pages (JSP)

There are many, many ways to write web code in Java. They can range from very simple, to extremely complicated.

The simplest manner, in my opinion is Java Server Pages (JSP), which provides the means to write a Java web application in a very similar manner to the PHP code I presented earlier. You embed the Java code within an HTML file.

<% and %> denote the start and end of the Java code. <%= %> can be used to display a variable.

```
<HTML>
<BODY>
<%
    // Other Java code here
    java.util.Date date = new java.util.Date();
%>
Hello!  The time is now <%= date %>
</BODY>
</HTML>
```


Java Server Faces (JSF)

Author Don Denoncourt wrote the Java article for the April issue. He chose to use Java Server Faces (JSF), which is a framework intended to simplify writing Java web applications.

JSF interfaces with JSP using "tag libraries" which are special HTML tags that you code into your JSP. When these tags appear, the underlying JSF code is called.

The following shows declarations you add (or a tool adds) to the top of your JSP when using JSF.

```
<%-- jsf:pagecode language="java" location="/src/pagecode/Rpgweb.java" --%>
<%-- /jsf:pagecode --%>
<%@page language="java" contentType="text/html; charset=ISO-8859-1"
        pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
<%@taglib uri="http://java.sun.com/jsf/html" prefix="h"%>

... more html follows …
```

The prefix="f" associates the f prefix with the JSF core routines
The prefix="h" associates the h prefix with the JSF html routines.
Rpgweb.java is the name of the Java code that will do the work.

Special Tags Embedded in HTML

```
   .
   .
<h:form id="acmeords1Form" >
   <table border="0" width="100%" bgcolor="lightblue">
   <tr>
     <td align="right">Customer Number:</td>
     <td align="left"><h:inputText id="custno"   size="6"/></td>
   </tr>
   <tr>
     <td align="right">-- OR --</td>
     <td align="left"> </td>
   </tr>
   <tr>
     <td align="right">Order number to change:</td>
     <td align="left"><h:inputText id="orderno" size="10"/></td>
   </tr>
   <tr>
     <td align="right">
       <h:commandButton value=" OK " action="#{pc_Rpgweb.getOrder}" /></td>
     <td> </td>
   </tr>
  </table>
</h:form>
```


Java Code Behind the JSF (1 of 2)

```
public class Rpgweb extends PageCodeBase {

   private DataSource dataSource = null;
   private Order header = new Order();

   private Connection getCon() throws SQLException, NamingException {
      if (dataSource == null) {
         dataSource = (DataSource) new InitialContext().lookup("jdbc/casdvl02");
      }
      return dataSource.getConnection();
   }

   public String getOrder() throws IllegalAccessException, InvocationTargetException,
                                   SQLException, NamingException {
      String custno = (String)getRequestParam().get("acmeords1Form:custno");
      String orderno = (String)getRequestParam().get("acmeords1Form:orderno");

      Object order = null;
      TreeMap items = new TreeMap();
      Connection con = null;

      try {
         con = getCon();

. . . Continued on next slide . . .
```

Java Code Behind the JSF (2 of 2)

```
. . . Continued from previous slide . . .

         if (custno != null && custno.trim().length() > 0) {
            order = new Order_new(con).call(new BigDecimal(custno.trim()))[0];
         } else if (orderno != null && orderno.trim().length() > 0) {
            order  = new Order_loadheader(con).call(orderno)[0];
            Order_loaditemsResult[] itemsResult
                  = new Order_loaditems(con).call(orderno);
            for (int i = 0; i < itemsResult.length; i++) {
               Item item = new Item();
               BeanUtils.copyProperties(item, itemsResult[i]);
               item = editItem(item);
               items.put(new Integer(item.getLineno()), item);
            }
         }
         header = new Order();
         BeanUtils.copyProperties(header, order);
         header.setItems(items);
      } catch (SQLException e) {
         throw e;
      } finally {
         try { con.close();} catch (SQLException e) {/* ignor */}
      }
      return "acmeords2";
}
```


Java Calling RPG

Author Don Denoncourt notes the following:

> *The Java code required to invoke an SQL procedure declares the SQL stored procedure (with specifics on the arguments), calls the procedure, handles errors, and if there are no errors, converts the returned result set into Java objects. But because the process of writing Java classes to call SQL stored procedures is both laborious and repetitive, I created an open-source tool that generates what I call RPG Call Beans.*

Indeed, the call to an RPG procedure was one line of code in RPG, but nearly 70 lines of code in Java. Fortunately, Don's tool generates that code for you, making it much easier.

```
      try {
         con = getCon();
         order = new Order_new(con).call(new BigDecimal(custno.trim()))[0];
      } catch (SQLException e) {
         throw e;
      } finally {
         try { con.close();} catch (SQLException e) {/* ignor */}
      }
```

Final Comparison

	RPG CGIDEV2	*PHP*	*JAVA*
Popularity (Whole industry)	Low (#25 at 0.297%)	High (#4 at 10.328%)	High (#1 at 20.529%)
Popularity (System i Industry)	High (#1 at 68%)	Low (#5 at 1%)	Medium (#3 at 11%)
Learning Curve	Very Easy **	Easy	Difficult
Security	Easy	Harder	Medium
Performance Small shops (< 200 users)	Excellent	Good	Poor
Performance (Larger shops)	Fair	Good	Excellent
Tooling Available	Minimal	Very Good	Excellent
Integration with i5/OS	Excellent	Good	Good
Cross-Platform	No	Yes	Yes
Supported	Community/IBM(??)	Zend	IBM

** For someone with existing RPG knowledge.

Links to April 2008 Articles

Articles from the April 2008 issue of System iNEWS magazine:

RPG and the Web: The CGIDEV2 Way (Paul Tuohy)
http://systeminetwork.com/article/rpg-web-cgidev2-way

RPG and the Web: The PHP Way (Tony Cairns)
http://systeminetwork.com/article/rpg-web-php-way

RPG and the Web: The Java or Groovy Way (Don Denoncourt)
http://systeminetwork.com/article/rpg-web-java-or-groovy-way

From System iNetwork Programming Tips e-newsletter:

Writing Reusable Service Programs (Scott Klement)
http://systeminetwork.com/article/writing-reusable-service-programs

This Presentation

You can download a PDF copy of this presentation from:

http://www.scottklement.com/presentations/

Thank you!