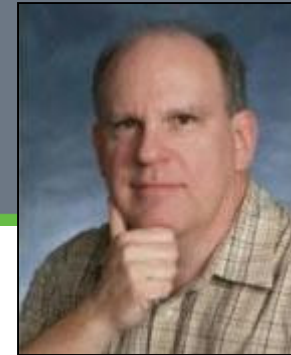


# Case Study: Zend Server on IBM i



Vermont Gas Systems  
Work Order Management System

# About John Valance



- **Independent consultant**
  - ▶ Specialty is helping iSeries shops develop web applications, and related skills
  - ▶ Training, mentoring, consultation and coding
- **25+ years iSeries/AS400 experience**
- **12+ years of web development experience**
  - ▶ Web scripting language of choice = PHP
- **Frequent presenter on web development topics**
- **Trainer for Zend Technologies**
  - ▶ Teaches Intro to PHP for RPG programmers

# Vermont Gas Systems

- **Natural Gas Utility in north western Vermont**
- **Regulated Business**
- **Serves Burlington and surrounding areas**
- **About 40,000 customers (small utility)**
- **Expanding service territory**

# IBM i (aka: iSeries, System i, i5, AS/400)

- ▶ IBM's legacy midrange platform
- ▶ Precursors date back to 1970s -80s
- ▶ Unique design, legendary reliability and longevity
  - High technology investment protection
- ▶ Proprietary, integrated, object-based operating system (i/OS)
  - Many built-in business capabilities
    - Database (DB2), Security, Communications,
  - Technology Independent Machine Interface
    - OS has survived many hardware technology changes
- ▶ Proprietary programming language = RPG
- ▶ Vast portfolio of 3rd party business applications, in all industries
  - Typically character-based terminal applications (aka green-screen)
- ▶ Runs enterprise applications, backbone for many medium to large businesses

# Screen-shot: IBM i Sign On Display

```
Sign On
System . . . . . : 00008047
Subsystem . . . . . : QINTER
Display . . . . . : QPADEV000R

User . . . . . : _____
Password . . . . . : _____
Program/procedure . . . . . : _____
Menu . . . . . : _____
Current library . . . . . : _____

(C) COPYRIGHT IBM CORP. 1980, 2005.
M a 06/053
```

# PHP on IBM i

- ▶ 2005: Zend/IBM partnership
  - Zend/PHP = Strategic technology for IBM i
  - IBM i = Strategic platform for Zend
- ▶ Simple Installation includes Zend Server CE, licenses for Zend Studio
- ▶ PHP has gained wide acceptance by IBM i community
- ▶ PHP is more accessible for RPG programmers than Java
  - Demand for education is growing
- ▶ Typically used to access legacy DB2 tables
- ▶ Toolkit for IBM i
  - Access native IBM i system objects
  - Call RPG programs

# Background - Reasons for a New System

- ▶ Project Scope: Rewrite VGS Work Order Mgmt System in PHP
  - Need to replace 15 y/o legacy, green-screen application, with numerous enhancements
  - Old system needed many enhancements, was difficult to maintain
  - Lots of redundant code, hard-coded value lists
- ▶ Technical goals of new system:
  - Modern, intuitive user interface
  - Solid, modular code base
  - Easily maintainable and extensible

# Why Custom PHP Solution?

- **Search for new system included several vendor offerings**
  - One vendor offering was \$1Million+ solution, plus services
  - Excellent solution, but...
  - Would have completely disrupted existing business processes
- **In the end, decision to use custom coded solution**
  - ▶ Browser-based interface
  - ▶ PHP / Zend Server running on IBM i
  - ▶ Consultant (me) to lead project and write code



# Benefits of Custom Approach

- Get exactly what they needed
- Low risk
- Low cost

## System functionality based on old system, with enhancements

- Meet pressing business requirements
- Greatly improved interface and code based
- Incremental improvements to business processes
- Allows for future enhancements

# Functional Features of the System

- **Maintain Information on Work Orders**
  - ▶ Anything related to construction or maintenance of gas lines
    - Gas Transmission lines (regional pipelines)
    - Gas Mains (local pipelines)
    - Gas Services (pipe to premises)
  - ▶ Repair Leaks (on mains / services)
  - ▶ Retire / Replace main or service
  - ▶ Work Order Type = describes type of line and work to be performed
- **Primary users are Engineering Department**
  - ▶ Accounting also researches W/O issues

# Ancillary Information

- Cleanup details
- Pipe Exposures
- Sewers on site
- **DIMP Data Collection (Key project goal)**
  - ▶ Distribution Integrity Management Plan

## Interfaces

- **Marketing - Sales Applications (New Installs)**
- **Accounting Activity / Project Costs**
  - ▶ Time Sheets / Payroll
  - ▶ Vendor Billing

# Life Cycle of Work Orders


- **Create Work Order**
- **Print Work Orders**
  - ▶ Several different formats for each WO type
- **Complete Work Orders**
  - ▶ Entry of data collected in field
- **Close Work Orders**
  - ▶ Post details to accounting

# Technical Features of the System

- **Hybrid Object Oriented / Procedural Design**
- **Model / View / Controller code organization**
- **Selective use of Zend Framework components**
- **Powerful custom helper classes**
  - ▶ Form generation
  - ▶ CRUD SQL generation
- **Highly consistent, easily maintainable code base**
  - ▶ Table searching/filtering/download
  - ▶ Single record CRUD screens
- **User maintainable list management (Drop-downs)**
- **Security**
  - ▶ Authentication with IBM i UID/PSWD
  - ▶ Robust access control at user or group level
- **Use of JOD Reports to generate PDF reports and printed forms**

# Demo of System





Database is PROD

## VGS Work Order Management System

### Main Menu

Script: menuMainCtrl.php  
User: JVALANCE [logout]  
Oct 17, 2012 @ 10:31:36 pm

---

#### Work Orders

- All Work Orders
- Leaks
- Create New Work Order
- Create Multiple SR/ST Orders
- Pipe Exposures
- Cleanups
- Sewers
- Plastic Pipe Failures
- Mechanical Fitting Failures

#### Tables

- Projects
- Pipe Types
- Drop Down Lists

#### Security

- Profiles
- User/Group Xref
- Authorities
- Profile Authorities
- Test with profile:

#### Reporting

- Cancelled Work Orders download
- W/O Inventory Reconciliation

#### System

- Log Out

©2010-2012 Vermont Gas Systems, Inc.  
P.O. Box 467, Burlington VT 05402. Phone: 802.863.4511.

# Use of Zend Framework

- **Began with desire to implement Zend Framework based architecture**
- **Not full implementation of Framework**
  - ▶ R & D period factored into schedule
  - ▶ After 2 months, could not get all features working
    - Zend\_DB\_Table, Zend\_DB\_Select, Zend\_Paginator
  - ▶ Created home-grown versions of these components
    - VGS\_DB\_Table, VGS\_DB\_Select, VGS\_Paginator
- **Used many of the concepts of Framework, but had to "roll our own" classes**
  - ▶ Very consistent design, using OO
- **No front controller, but using a common layout.php on all pages**
  - ▶ Handles authentication, error checking, page layout for every screen.
- **Extensions to Zend\_Form**
  - ▶ VGS\_Form extends Zend\_Form
- **VGS\_Form\_Helper**
  - ▶ Automatically sets many form attributes from DB2 metadata

# General Application Screens Structure

- **layout.php** - Included in each view script
  - ▶ Functions to show header and footer
  - ▶ Error handling: `set_error_handler()`
  - ▶ Session management: `session_start()`
    - Check `isset($_SESSION['userId'])`
    - If not, redirect to `loginCtrl.php`

## Two main types of applications:

- **Search Screens**
  - ▶ Multi-record, filtered, paginated record lists
- **Edit/Display Record screens**
  - ▶ Single-record screens, for CRUD operations

Cookie-cutter design for building list and record screens



# Search Screens - components

- **Nav Buttons bar**
  - ▶ Main menu (or close pop-up)
  - ▶ Download (optional)
  - ▶ Create record (optional)
  - ▶ + Custom buttons (app specific)
- **Filter bar**
- **Paginator bar**
- **Search Results (records matching search criteria)**
- **Download capability**
  - ▶ Uses filters entered for search

# Edit/Display Record screens - components

- **Nav Buttons bar**

- ▶ Main menu (or close pop-up)
- ▶ Save (not in display mode)
- ▶ Cancel
- ▶ Return to Search List
- ▶ + Custom buttons (app specific)

- **Form**

- ▶ Field Groups (defined in Form class)

# Record detail screens - CRUD

- **Modes:**
  - ▶ Create, Read (display), Update (edit), Delete
- **Create and Edit modes**
  - ▶ Same validations, by default
  - ▶ If form data changed, warning on cancel or navigate away
- **Display mode**
  - ▶ Same form layout as Create/Edit
  - ▶ Set all inputs to readonly, class="disabled", tabindex="-1", and clearValidators()
- **Delete mode**
  - ▶ Show record output only, format like display mode
  - ▶ Replace "Save button" with red "Delete button"
  - ▶ Confirm dialog before delete

# Building a Detail Screen

Three primary objects are involved, related to table being maintained:

- **Form object, extends Zend\_Form, e.g.:**
  - ▶ `class Project_Form extends Zend_Form { ... }`
- **Form Helper object (custom code, very helpful)**
  - ▶ `class VGS_FormHelper { ... }`
  - ▶ Retrieves DB2 metadata for table fields on form
  - ▶ Sets appropriate filters, validators, attributes based on metadata
- **Table object, extends VGS\_DB\_Table, e.g.:**
  - ▶ `class Project_Master extends VGS_DB_Table { ... }`

# Example -Sewer Update Form

Database is PROD

Vermont Gas

**VGS Work Order Management System**  
**Update W/O Sewer Information**

Script: wswEditCtrl.php  
User: JVALANCE [logout]  
Oct 19, 2012 @ 04:19:25 pm

Main Menu Save Cancel Sewer List for W/O 64901 Sewer List (all W/Os) Display W/O 64901

Enter changes and press ENTER to save. (\* = Required entry)

Work Order Details		Method of Construction	
W/O Number :	64901 19 Whisper Ln	MOC Trench :	<input checked="" type="checkbox"/>
W/O Type :	SI Service New Construction	MOC HDD :	<input checked="" type="checkbox"/>
W/O Status :	CMP Completed	MOC Hog :	<input type="checkbox"/>
Date Completed :	Oct 08, 2012	MOC Plowed :	<input type="checkbox"/>
		MOC Other :	

Sewer Information		Comments	
Sewer sequence# :	1	General Notes :	
* Address :	19 Whisper Ln		
	<input type="button" value="Copy from W/O"/>		
* City :	Milton Village		
Sewer Located Prior? :	<input type="checkbox"/>		
Sewer Size :			
Sewer Material :			
* Sewer Type :	Septic		
* Separation From New Gas Install :	Rear/Opposite Side		
Inspection Needed? :	<input type="checkbox"/>		
* Reason :	in rear		

Sewer Record Maintenance Info	
Created by User ID :	KIM
Date/Time Created :	Tue, Oct 09 2012 at 02:06:15 pm
Last Changed by User ID :	
Date/Time Last Changed :	

Save Changes Cancel

# Zend\_Form - base ZF class

- **HTML forms are unwieldy, and contain many co-dependent elements**
  - ▶ `<form>` tag
  - ▶ `<input>` tags of various types
  - ▶ Field labels
  - ▶ CSS and other rendering attributes
  - ▶ Error messages
  - ▶ Data values
- **Server side code must handle:**
  - ▶ Form loading (new record defaults/ existing record values)
  - ▶ Validations based on data types, business rules
  - ▶ Reloading form with values and messages, until valid input

# OO to the Rescue!

## Zend\_Form / Zend\_Form\_Element

- **Zend Framework provides classes to simplify/abstract form processing**
  - ▶ Handle all aspects of form definition and processing in PHP
  - ▶ Use `render()` method to display the form.
  - ▶ Largely avoids HTML, controls everything via structured program code.
- **Zend\_Form ~ = <form>**
- **Zend\_Form\_Element ~ = <input>**
  - ▶ Zend\_Form object contains multiple Zend\_Form\_Element objects

# Zend\_Form example

```
$form = new Zend_Form;
$form->setAction('/resource/process')
    ->setMethod('post');
$username = new Zend_Form_Element_Text('username');
$username->addValidator('alnum')
    ->addValidator('regex', false, array('/^[a-z]/'))
    ->setRequired(true)
    ->addFilter('StringToLower');
$form->addElement($username);
```



# Zend\_Form methods

- **reset()** - clear form or load defaults for new record in create mode
  - ▶ can override in derived class, with application appropriate values
- **populate(\$dataArray)** - load existing record in edit/display/delete mode
  - ▶ keys of \$dataArray are names of form elements
  - ▶ good idea to use DB field names
- **validate()** - perform custom validations on inputs
  - ▶ will run all validators added
    - you can create custom, reuseable validators
  - ▶ can override in derived class with custom validations
- **render()** - generate HTML `<form>` and `<input>` tags for all elements
  - ▶ handles all attributes, including `value="..."`
  - ▶ decorators handle positioning (HTML container tags)

```
abstract class VGS_Form extends Zend_Form { ... }
```

## <sup>A</sup> VGS\_Form

◇ \$conn

○ \$fh

○ \$inputs

○ \$mode

○ \$return\_point

○ \$screen\_title

○ \$valid

● <sup>C</sup> \_\_construct(\$conn)

● activate() : void

■ buildLookup(\$fieldName) : void

● <sup>S</sup> convertDateFormat(\$dateStr, \$fromFmt, \$toFmt, \$padLen=8)

● <sup>A</sup> createRecord() : void

● deleteRecord() : void

● <sup>S</sup> fixDateInput(string) : void

● <sup>S</sup> fixDateOutput(string, \$bInOutputOnly=false) : void

● <sup>S</sup> fixTimeInput(string) : void

● <sup>S</sup> fixTimeOutput(string) : void

● <sup>S</sup> getTimeStampOutputFormat(\$timestamp) : void

● isCreateMode() : void

● isDeleteMode() : void

● isInquiryMode() : void

● isUpdateMode() : void

● loadScreen() : void

■ preprocessFormInputs() : void

● processScreen() : void

● renderFieldGroup(\$fieldGroupName) : void

● renderFormButtons() : void

● renderFormHeaderMessage() : void

● renderFormHiddens() : void

● renderFormJS() : void

● renderFormTop() : void

● <sup>A</sup> retrieveRecord() : void

● returnToCaller() : void

● setDateOutputFormat(\$field) : void

■ setInputFormatsForDB2() : void

■ setOutputFormatsForScreen(\$data) : void

● <sup>A</sup> updateRecord() : void

● validate() : void

# VGS\_Form in a Nutshell

- Extends Zend\_Form (we get all that goodness!), plus...
- Basic data access methods (create, update, retrieve, delete)
- General DB2 data filtering (screen <-> database)
- Form rendering
  - ▶ Field groups, elements, buttons, messages, hidden fields, JavaScripts.
- Form processing
  - ▶ Initialization, loading from database, validation, database update, redirecting after success
- Boolean mode methods
  - ▶ isInquiryMode(), isUpdateMode(), etc...
- Attaching input helpers / popups
  - ▶ Lookups (foreign key table search popup)
  - ▶ Date pickers

# VGS\_Form\_Helper

- Loads metadata from DB2 for one or more tables
- Builds form elements from the metadata for selected fields
- Adds appropriate data type validations, filters and attributes, and labels to form elements
  - ▶ This saves a lot of tedious coding
- Allows definition of field groups (boxes of fields on screen)
- Fields can be added to form by passing comma-separated list of field names.
- Can define additional validations and attributes for lists of fields
  - ▶ Required entry, output only, override input type, attribs

# class VGS\_FormHelper

- 📁 VGS\_FormHelper
  - \$elements
  - \$fieldGroups
  - \$metaData
  - \$mode
  - \$view
  - <sup>C</sup>\_\_construct()
  - addCustomMetaDatum(\$name, \$text, \$type, \$length, \$precision="") : void
  - addElement(\$name, \$element) : void
  - addElementsFromMetaData() : Count
  - addFieldGroup(\$fieldList, \$fieldGroup, \$caption) : void
  - 📁 addMetaData(\$conn, \$table) : void
    - buildElementFromMetaData(\$elemMeta) : void
  - getElements() : void
  - getMetaData() : void
  - 📁 <sup>S</sup>getObjectLibrary(\$object, \$objType) : void
    - renderFieldGroup(\$fieldGroup, Zend\_Form) : void
    - setDescription(\$fieldName, \$description) : void
    - setElementDataTypeFilter(Zend\_Form\_Element, \$meta) : void
    - setElementProperty(\$name, \$property, \$value) : void
    - setElementsProperties(\$namesList, \$property, \$value) : void
    - setMultiOptions(\$fieldName, \$optionsList) : void
    - splitNames(string) : true

# VGS\_FormHelper - public attributes

```
class VGS_FormHelper
{
    /** The $metaData array will be used to generate labels,
    * filters and validators for the form elements automatically.
    * @var array
    */
    public $metaData = array();

    /** Contains an array of Zend_Form_Element to include on the form
    * @var array
    */
    private $elements = array();

    /**
    * Holds an array describing the field groupings for display
    * @var array
    */
    public $fieldGroups = array();
}
```

# Concrete form: WO\_Sewer\_Form - constructor

```
class WO_SewerForm extends VGS_Form
{
    private $swsRec; // Record array for existing w/o sewer record
    private $woRec; // Complete w/o record for the related w/o

    // Key fields for this sewer record
    private $woNum;
    private $swsSeqNo;

    public function __construct( $conn, $woNum ) {
        parent::__construct ( $conn );

        {constructor stuff}...

        $this->fh->addMetaData( $conn, "WO_SEWER" );
        $this->setDefaultElements( );
    }
}
```



## VGS\_Form\_Helper->addMetaData(\$conn, \$table)

- Retrieve column attributes from DB2 (qsys2/syscolumns)
- Store in VGS\_Form\_Helper->metaData array

```
public function addMetaData($conn, $table) {
    $schema = self::getObjectLibrary($table, '*FILE');

    $syscols = new VGS_DB_Table($conn);
    $query = "select * from qsys2/syscolumns
        where table_schema = '$schema'
        and system_table_name = '$table' ";
    $rs = $syscols->execListQuery($query);

    while ($sysColumn = db2_fetch_assoc($syscols->stmt)) {
        // Add each column's metadata to the master metadata array
        $colName = $sysColumn['COLUMN_NAME'];
        $this->metaData[$colName] = $sysColumn;
    }
}
```



# WSW\_ADDRESS - MetaData

- UPPER\_CASE = fields from QSYS2/SYSCOLUMNS
- **Green** = attributes used to generate form elements
- **Red** = attributes added by custom code

<pre>COLUMN_NAME = WSW_ADDRESS TABLE_NAME = WO_SEWER TABLE_OWNER = ORCOM ORDINAL_POSITION = 3 DATA_TYPE = VARCHAR LENGTH = 100 NUMERIC_SCALE = IS_NULLABLE = N IS_UPDATABLE = Y LONG_COMMENT = HAS_DEFAULT = Y COLUMN_HEADING = Address STORAGE = 102 NUMERIC_PRECISION = CCSID = 37 TABLE_SCHEMA = WORKORDT COLUMN_DEFAULT = '' CHARACTER_MAXIMUM_LENGTH = 100 CHARACTER_OCTET_LENGTH = 100 NUMERIC_PRECISION_RADIX =</pre>	<pre>DATETIME_PRECISION = COLUMN_TEXT = Address SYSTEM_COLUMN_NAME = WSWADDR SYSTEM_TABLE_NAME = WO_SEWER SYSTEM_TABLE_SCHEMA = WORKORDT USER_DEFINED_TYPE_SCHEMA = USER_DEFINED_TYPE_NAME = IS_IDENTITY = NO IDENTITY_GENERATION = IDENTITY_START = IDENTITY_INCREMENT = IDENTITY_MINIMUM = IDENTITY_MAXIMUM = IDENTITY_CYCLE = IDENTITY_CACHE = IDENTITY_ORDER = <b>group = sewer</b> <b>include = 1</b> <b>label-class = required</b> <b>required = 1</b></pre>
--	--

# WO\_Sewer\_Form->setDefaultElements( )

```
public function setDefaultElements( ) {

    $flWO = 'WSW_WO_NUM, WO_TYPE, WO_STATUS, WO_DATE_COMPLETED';
    $this->fh->addFieldGroup( $flWO, 'wo', 'Work Order Details');
    $this->fh->setElementsProperties( $flWO, 'output_only', true);

    $flSewer =
        'WSW_SEQNO, WSW_ADDRESS, WSW_CITY, WSW_LOCATED_PRIOR, WSW_SEWER_SIZE,
        WSW_SEWER_MATERIAL, WSW_SEWER_TYPE, WSW_SEPARATION_FROM_GAS,
        WSW_INSPECTION_NEEDED, WSW_INSPECT_REASON';

    $this->fh->addFieldGroup( $flSewer, 'sewer', 'Sewer Information');

    $this->fh->setElementsProperties(
        'WSW_SEQNO',
        'output_only', true);

    $this->fh->setElementsProperties(
        'WSW_ADDRESS, WSW_CITY, WSW_SEWER_TYPE, WSW_SEPARATION_FROM_GAS'
        'required', true);
}
```

## WO\_Sewer\_Form->setDefaultElements( ) - cont'd

```
$this->fh->setElementsProperties(  
    'WSW_LOCATED_PRIOR, WSW_INSPECTION_NEEDED',  
    'input_type', 'y/n');  
$this->fh->setElementsProperties(  
    'WSW_CITY, WSW_SEWER_TYPE, WSW_SEPARATION_FROM_GAS',  
    'input_type', 'select');  
$this->fh->setElementsProperties(  
    'WSW_NOTES', 'input_type', 'textarea');
```

*etc...*

```
// This creates Zend_Form_Elements out of the meta data  
$this->fh->addElementFromMetaData();  
$this->addElement ( $this->fh->getElements() );  
  
// Add a drop-down (<select>) list for Town  
$dd = new Code_Values_Master($this->conn);  
$ddList = $dd->getCodeValuesList('TOWN', ' ');  
$this->fh->setMultiOptions('WSW_CITY', $ddList);
```

*etc...*



Database is TEST

# VGS Work Order Management System Drop Down Lists Search

User: JVALANCE [logout]  
Oct 20, 2012 @ 10:16:42 pm

Main Menu


+ Create Drop Down List

[Clear] Filter on: List ID  Record Status **All** Description

Page 2 of 4.  << Previous Go to Page:  Go Next >>  99 matches.

	Drop Down ID	Description	Status	Codes	Last Update
Values	REPAIR_METHOD_EQUIP	Repair method/equipment for maintenance W/Os	Active	11	
Values	CONDITION_FOUND	Pipe condition found for maintenance W/Os	Active	10	
Values	CONS_TYPES_MI	Construction Types for Main Install	Active	4	
Values	CONS_TYPES_SI	Construction Types for Service Install	Active	5	
Values	PP_CONTACT	Default contact info for plastic pipe failure	Active	3	
Values	PP_MANUFACTURER	Plastic Pipe Failure - Manufacturer	Active	3	
Values	SEWER_TYPE	Sewer Types	Active	5	
Values	PREMISE_STS	Premise record status codes (UPRM)	Active	10	
Values	METHOD_OF_CONSTRUCTION	Method of Construction	Active	6	
Values	WCN_EMAIL_ADDR	Email address to receive W/O cancellation notifications	Active	3	
Values	WCN_REASON_CODE	W/O Cancellation Reasons	Active	9	
Values	PRJ_CUSTOMER_EXCAVATED	Customer Excavated	Active	4	Thu, Jan 12 2012 at 10:14:19 am
Values	RATECLASS	Rate Classes	Active	6	Mon, Nov 21 2011 at 11:14:32 am
Values	WPE_PIPE_COMPOSITION	Pipe Composition	Active	6	Fri, Oct 07 2011 at 04:19:28 pm
Values	LK_SURVEY_TYPE	Leak Survey Type	Active	5	Fri, Oct 07 2011 at 04:17:14 pm
Values	AP_PROFILE_TYPE	Authority Profile Type	Active	2	Fri, Aug 12 2011 at 04:12:56 pm
Values	AP_PERMISSION	Authority/Profile Permissions	Active	4	Fri, Aug 12 2011 at 12:57:28 pm
Values	AD_FUNCTIONAL_AREA	Grouping ID for Authority Definitions	Active	2	Fri, Aug 12 2011 at 12:50:20 pm
Values	WO_RETIRED_MAIN	Retired With or At Main	Active	3	Tue, Jul 26 2011 at 11:03:53 am
Values	LK_EVENTS	Leak Collateral Incidents	Active	3	Mon, Jul 25 2011 at 04:56:37 pm
Values	WO_METER_LOCATION	Meter Location	Active	4	Mon, Jul 18 2011 at 03:29:02 pm
Values	PRJ_STATUS	Project Status	Active	5	Tue, Jul 12 2011 at 02:29:24 pm
Values	WO_FLOW_LIMITER_SIZE	Valid Flow Limiter Sizes	Active	3	Fri, Jul 08 2011 at 12:32:24 pm
Values	MF_MECHANICAL_FITTING	Mech Fitting Failure Fitting	Active	4	Thu, Jul 07 2011 at 02:48:01 pm
Values	MF_SUPPLEMENTAL_REPORT	Mech Fitting Supplemental Report	Active	2	Wed, Jul 06 2011 at 03:41:47 pm
Values	MF_INITIAL_REPORT	Mech Fitting Initial Report - Y/N	Active	2	Wed, Jul 06 2011 at 03:39:45 pm

# User Maintainable Drop Down Lists (for <select>)

Database is TEST












## VGS Work Order Management System Drop Down Values Search

User: JVALANCE [logout]  
Oct 20, 2012 @ 10:18:24 pm

Main Menu Drop Down Lists + Create Drop Down Value

[Clear] Filter on: Group **SEWER\_TYPE** Status **\*All** Value  Description

Page 1 of 1. First Page << Previous Go to Page:  Go Next >> Last Page 5 matches.

	List	Seq#	Code	Value	Description	Status
  	SEWER_TYPE	1.0000	SEWER	Sewer		Active
  	SEWER_TYPE	2.0000	SEPTIC	Septic		Active
  	SEWER_TYPE	3.0000	BOTH	Both (sewer/septic)		Active
  	SEWER_TYPE	4.0000	NONE	None		Active
  	SEWER_TYPE	5.0000	UNKOWN	Unknown		Active

©2010-2012 Vermont Gas Systems, Inc.  
P.O. Box 467, Burlington VT 05402. Phone: 802.863.4511.

# Table Object (VGS\_DB\_Table)

Defines all table specific data access methods

- Parent class (VGS\_DB\_Table)

Encapsulates basic DB2 functionality:

- **Public methods used with Search lists and VGS\_Paginator:**
  - ▶ `execListQuery($queryString, $bindParam = array())`
  - ▶ `execScrollableListQuery(VGS_DB_Select $select)`
  - ▶ `getRowCount(VGS_DB_Select $select)`
- **Public methods used to retrieve and update single record (detail screens)**
  - ▶ `execUpdate($queryString, $bindParam = array())`
  - ▶ `fetchRow($queryString, $bindParam = array())`
- **Security:**
  - ▶ `checkPermissionByCategory( $category, $mode )`
  - ▶ Ensure user has authority to table for given mode

## VGS\_DB\_Table - SQL generator methods

`autoCreateRecord(array $inputs)`

`autoUpdateRecord(array $inputs)`

`autoDeleteRecord(array $inputs)`

- Automatically build SQL statements from form inputs
- If form fields change, never need to modify SQL statements
- Never have to align field names and values
- Uses bound parameters - no need to align parameter markers (?s)
- Huge time saver; ensures accurate updates without coding
- Bound parameters prevents SQL injection attacks

# VGS\_DB\_Table - public attributes

Name of the database table, specified in UPPERCASE.

```
public $tableName;
```

Field names prefix for this table (eg: 'WO\_');

Used to extract the update fields from form inputs.

```
public $tablePrefix;
```

Array of key field names for this table

```
public $keyFields;
```

Boolean = table includes audit fields (Default = true)

```
public $hasAuditFields;
```

Boolean = physical record delete is allowed. (Default = false)

```
public $isRecordDeletionAllowed;
```

- With above attributes, system can automatically create SQL for create, update, delete, and set audit fields appropriately.



# Example of VGS\_DB\_Table based class

```
class WO_Sewer extends VGS_DB_Table {
    public function __construct($conn) {
        parent::__construct($conn);
        $this->tableName = 'WO_SEWER';
        $this->tablePrefix = 'WSW_';
        $this->keyFields = array('WSW_WO_NUM', 'WSW_SEQNO');
        $this->hasAuditFields = true;
        $this->isRecordDeletionAllowed = true;
    }

    public function create( $rec ) {
        $this->checkPermissionByCategory('WO', 'CREATE');
        $rec['WSW_SEQNO'] = $this->getNextSewerNum($rec['WSW_WO_NUM']);
        $this->autoCreateRecord($rec);
    }

    public function update( $rec ) {
        $this->checkPermissionByCategory('WO', 'UPDATE');
        $this->autoUpdateRecord($rec);
    }
}
```

# Example - Update Sewer Details

Database is PROD

Vermont Gas

**VGS Work Order Management System**  
**Update W/O Sewer Information**

Script: wswEditCtrl.php  
User: JVALANCE [logout]  
Oct 19, 2012 @ 04:19:25 pm

Main Menu Save Cancel Sewer List for W/O 64901 Sewer List (all W/Os) Display W/O 64901

Enter changes and press ENTER to save. (\* = Required entry)

Work Order Details	
W/O Number :	64901 19 Whisper Ln
W/O Type :	SI Service New Construction
W/O Status :	CMP Completed
Date Completed :	Oct 08, 2012

Method of Construction	
MOC Trench :	<input checked="" type="checkbox"/>
MOC HDD :	<input checked="" type="checkbox"/>
MOC Hog :	<input type="checkbox"/>
MOC Plowed :	<input type="checkbox"/>
MOC Other :	

Sewer Information	
Sewer sequence# :	1
* Address :	19 Whisper Ln Copy from W/O
* City :	Milton Village
Sewer Located Prior? :	<input type="checkbox"/>
Sewer Size :	
Sewer Material :	
* Sewer Type :	Septic
* Separation From New Gas Install :	Rear/Opposite Side
Inspection Needed? :	<input type="checkbox"/>
* Reason :	in rear

Comments	
General Notes :	

Sewer Record Maintenance Info	
Created by User ID :	KIM
Date/Time Created :	Tue, Oct 09 2012 at 02:06:15 pm
Last Changed by User ID :	
Date/Time Last Changed :	

Save Changes Cancel

# Sewer Update - example input values

- Inputs array passed to VGS\_Form->autoUpdateRecord( \$inputs )
- **Blue elements** are ignored: prefix not = 'WSW\_'

```
WSW_WO_NUM = 64901
WSW_SEQNO = 1
popup =
mode = update
a = update
return_point = /wotest/controller/wswListCtrl.php
WO_TYPE = SI
WO_STATUS = CMP
WO_DATE_COMPLETED = Oct 08, 2012
WSW_ADDRESS = 19 Whisper Ln
WSW_CITY = MLV
WSW_LOCATED_PRIOR = N
WSW_SEWER_SIZE =
WSW_SEWER_MATERIAL =
WSW_SEWER_TYPE = SEPTIC
WSW_SEPARATION_FROM_GAS = REAR
WSW_INSPECTION_NEEDED = N
WSW_INSPECT_REASON = in rear
WSW_MOC_TRENCH = Y
WSW_MOC_HDD = Y
WSW_MOC_HOG = N
WSW_MOC_PLOWED = N
WSW_MOC_OTHER =
WSW_NOTES =
```

# Results of VGS\_DB\_Table->autoUpdateRecord(array \$inputs)

- Builds SQL update string, and array of values to bind, then...
- `$this->execUpdate($sql, $values);`

## \$sql:

```
update WO_SEWER set WSW_ADDRESS = ?, WSW_CITY = ?, WSW_LOCATED_PRIOR = ?, WSW_SEWER_SIZE = ?,  
WSW_SEWER_MATERIAL = ?, WSW_SEWER_TYPE = ?, WSW_SEPARATION_FROM_GAS = ?, WSW_INSPECTION_NEEDED = ?,  
WSW_INSPECT_REASON = ?, WSW_MOC_TRENCH = ?, WSW_MOC_HDD = ?, WSW_MOC_HOG = ?, WSW_MOC_PLOWED = ?,  
WSW_MOC_OTHER = ?, WSW_NOTES = ? , WSW_CHANGE_USER = ?, WSW_CHANGE_TIME = current timestamp  
where WSW_WO_NUM = ? AND WSW_SEQNO = ?
```

## \$values:

WSW_ADDRESS = 19 Whisper Ln	WSW_MOC_TRENCH = Y
WSW_CITY = MLV	WSW_MOC_HDD = Y
WSW_LOCATED_PRIOR = N	WSW_MOC_HOG = N
WSW_SEWER_SIZE =	WSW_MOC_PLOWED = N
WSW_SEWER_MATERIAL =	WSW_MOC_OTHER =
WSW_SEWER_TYPE = SEPTIC	WSW_NOTES =
WSW_SEPARATION_FROM_GAS = REAR	WSW_CHANGE_USER = JVALANCE
WSW_INSPECTION_NEEDED = N	WSW_WO_NUM = 64901
WSW_INSPECT_REASON = in rear	WSW_SEQNO = 1


- **Red** = audit fields - automatically inserted
- **Green** = key fields, put at end of \$values array

# JOD Reports

- **Java OpenDocument Reports**
- **<http://jodreports.sourceforge.net/>**
- **Open source, Java-based report template tool**
- **Create documents and reports in OpenDocument Text format from templates**
- **Templates can be visually composed using the OpenOffice.org Writer word processor**
- **These documents can then be converted to PDF, Word and RTF with JODConverter**

# JODReports XML request

```
<WO>
  <WO_NUM>65093</WO_NUM>
  <WO_ENTRY_DATE>Sat Oct 20, 2012</WO_ENTRY_DATE>
  <NEED_BY_DATE>Thu Oct 25, 2012</NEED_BY_DATE>
  <WO_DESCRIPTION>47 Barrett St</WO_DESCRIPTION>
  <WO_PREMISE_NUM>27590</WO_PREMISE_NUM>
  <OWNERS_NAME>Valance, John G</OWNERS_NAME>
  <OWNERS_PHONE>802-355-4024</OWNERS_PHONE>
  <METER_NO>25018</METER_NO>
  <WO_SPECIAL_INSTRUCTION />
  <WO_TYPE_DESC>Service New Construction</WO_TYPE_DESC>
  <WO_GL_COST>VGSBS-1071-0-65093</WO_GL_COST>
  <PT_DESCRIPTION>Plastic Service 1<span style="color: blue;">" ;</span></PT_DESCRIPTION>
  <ESTLEN>.00</ESTLEN>
  <ESTHRS>.00</ESTHRS>
  <CURBSTOP>N</CURBSTOP>
  <FLWLIM>800</FLWLIM>
  <WO_TOWN_NAME>So. Burlington</WO_TOWN_NAME>
  <MAIN_PIPE_TYPE />
</WO>
```

 **VERMONT GAS SYSTEMS, INC.** WO#: Input field  
Page Page numbers  
Work Order for Input field

Date Issued: Input field	Needed By: Input field	Charge Acct#: Input field
Location: Input field - Input field		ROW#: Input field
Type of Work: Input field		Pipe Type: Input field Input field
Owner's Name: Input field	Premise: Input field	Est. Length: Input field Est. Crew Hours: Input field
Owner's Phone#: Input field	Meter: Input field	Curb Stop: Input field Flow Limiter: Input field

Special Instructions: Input field  
 Dig Safe Auth. No.: \_\_\_\_\_ Excavation Permit #: \_\_\_\_\_  
 Date Called: \_\_\_\_\_ By: \_\_\_\_\_  
 Begin Dig Date: \_\_\_\_\_

Soil: Sand Clay Rocky Packing: Loose Me  
 Soil Moisture: Dry Moist Wet N/A

**Exposed MAIN Info**

Exposed MAIN Info Size: 3/4" 1" 1.25" 2" 4" 6"  
 Pipe Type: DPE8000 DPE8100 PPE8300 Endo  
 Coating Type: FlexClad Pritec Xtru Scotch  
 Coating External Cond.: Good Fair Poor Smooth  
 Internal Cond.: Good Fair Poor Smooth Pi  
 CP20 Reading: \_\_\_\_\_ Main Depth: \_\_\_\_\_  
 At Main | With Main | Other: \_\_\_\_\_

Clean Up Required: (Dimensions)	SisApp#	Premise#	Address	Apt#	Sis Pers.
<input type="checkbox"/> Topsoil <input type="checkbox"/> Blacktop <input type="checkbox"/> Concrete <input type="checkbox"/> Gravel/Stone	Input field	Input field	Input field	Input field	Input field

Comments: Input field

Input Field

Edit

jooscript

\$(WO.WO\_ENTRY\_DATE)

OK Cancel Help

# Use CURL to request JOD Report

```
public function retrievePDF($xml, $request) {  
    // urlencode and concatenate the POST arguments  
    $postargs = 'outputFormat=pdf&model=' . urlencode($xml);  
    // $request is JOD-compliant URL for appropriate report template  
    $session = curl_init ( $request );  
    curl_setopt($session, CURLOPT_POST, true); // use HTTP POST  
    curl_setopt($session, CURLOPT_POSTFIELDS, $postargs); // this is body of POST  
    curl_setopt($session, CURLOPT_HEADER, false); // return headers with response  
    curl_setopt($session, CURLOPT_RETURNTRANSFER, true); // return response  
    curl_setopt($session, CURLOPT_BINARYTRANSFER, true); // binary response  
  
    $response = curl_exec ( $session );  
    curl_close ( $session );  
    return $response;  
}
```



Contact Information:  
John Valance  
[johnv@jvalance.com](mailto:johnv@jvalance.com)  
802-355-4024