CASE STUDY:

PRODUCTION SCHEDULING USING JQUERY & PHP ON IBM i

John Valance

Division 1 Systems

johnv@div1sys.com

<div1>

division 1 systems

Company – Polar Beverages

- Largest independent soft-drink bottler in US
- Headquarters: Worcester, MA
- Fourth-generation, family-owned business
- Roots back to 1882.
- Manufacturer and distributor of Polar brand and numerous other soft drink brands
- Uses BPCS on IBM i for MRP
- http://www.polarbev.com/

Project

- Create an intuitive production scheduling application
- Replace a manual system done in MS Excel
- Allow drag/drop scheduling of manufacturing shop orders
- No stand alone solution that met their requirements available
- Application design and requirements done by IT Director and Master Scheduler
- Application progamming done by John Valace
- Other vendors considered, but:
  - much higher estimates
  - would not provide the intuitive interface they wanted

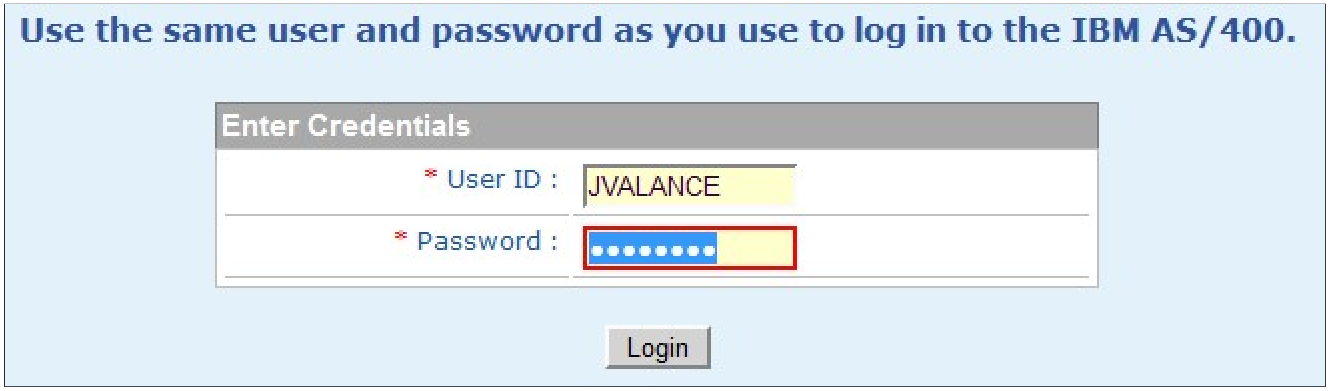
Application Screens

- Login
- Production Schedule Selection (parameter entry)
- Production Schedule Maintenance (main application screen)

- **DEMO**

Login screen

- Convert case on user id
- Use IBM toolkit to validate profile
- Store user ID in session
  - If page within application is requested, and user is not logged in, bounce to login screen

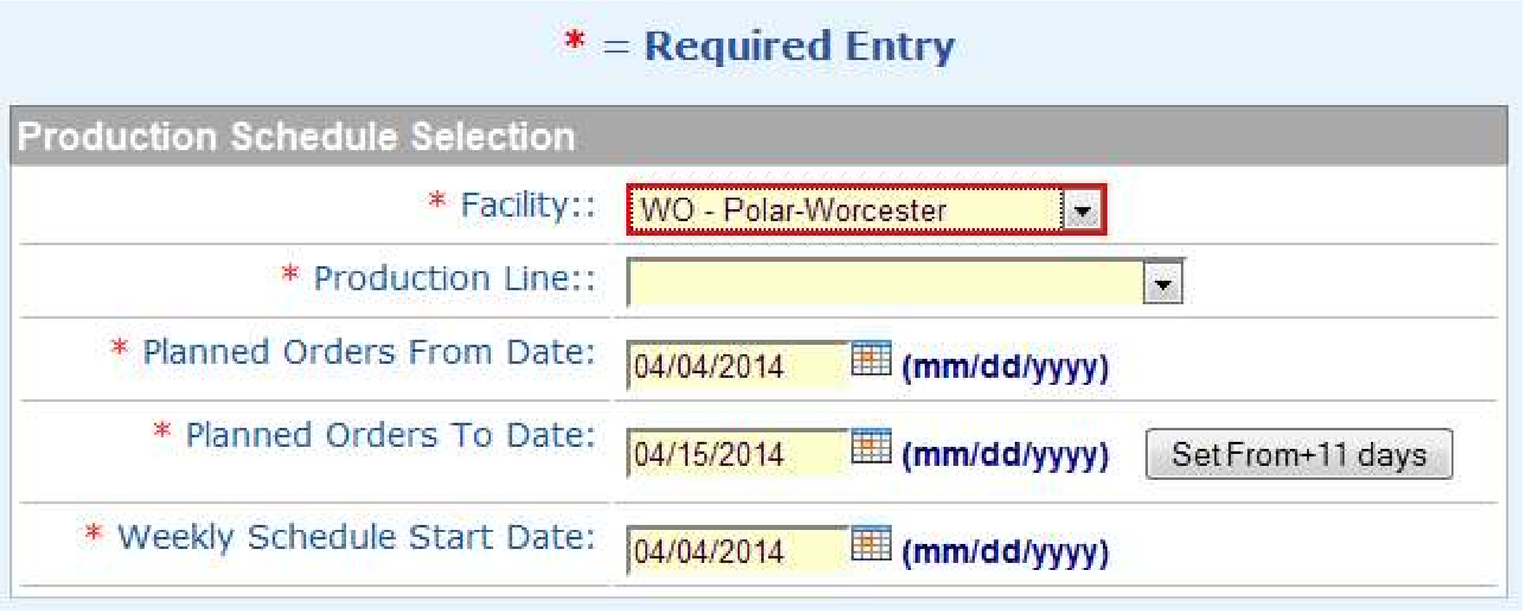


Use the same user and password as you use to log in to the IBM AS/400.

Enter Credentials

* User ID :	JVALANCE
* Password :	●●●●●●●●●

Production Schedule Selection screen



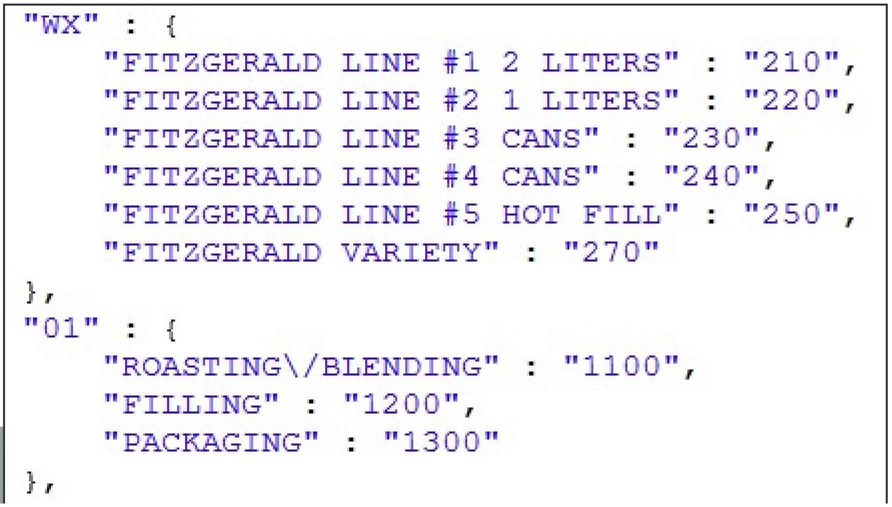
* = Required Entry

Production Schedule Selection

* Facility::	WO - Polar-Worcester		
* Production Line::			
* Planned Orders From Date:	04/04/2014	(mm/dd/yyyy)	
* Planned Orders To Date:	04/15/2014	(mm/dd/yyyy)	Set From+11 days
* Weekly Schedule Start Date:	04/04/2014	(mm/dd/yyyy)	

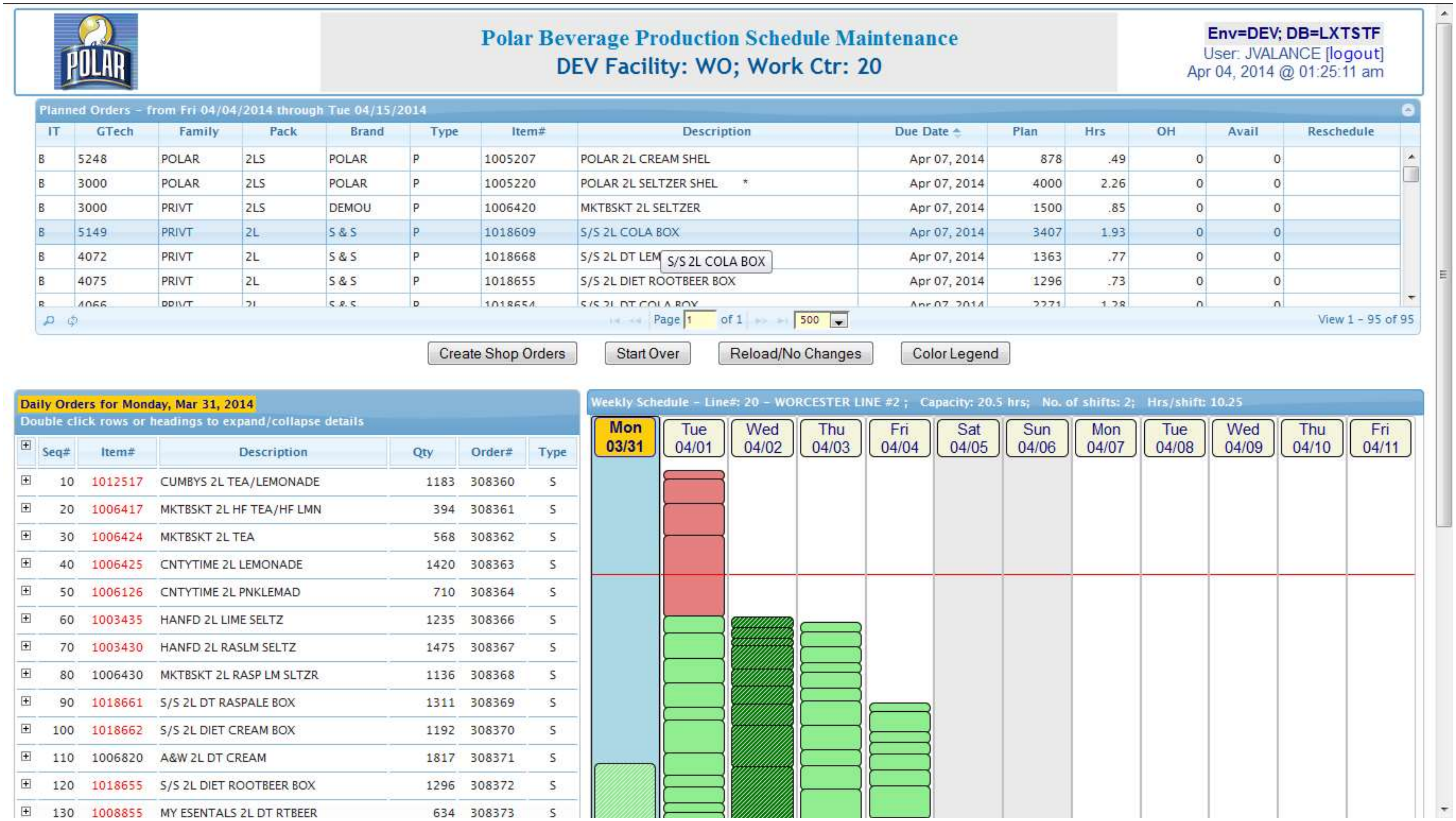
Production Schedule Selection screen

- Provide parameters for loading the scheduling screen
  - Facility (i.e. mfg plant)
  - Production line
  - Date range for planned orders search list
  - Start date for current schedule graph (12 days out)
- Uses jQuery date pickers
- jQuery used for loading Production Lines based on Facility
  - JSON returned from PHP builds an object structure



```
"WX" : {
    "FITZGERALD LINE #1 2 LITERS" : "210",
    "FITZGERALD LINE #2 1 LITERS" : "220",
    "FITZGERALD LINE #3 CANS" : "230",
    "FITZGERALD LINE #4 CANS" : "240",
    "FITZGERALD LINE #5 HOT FILL" : "250",
    "FITZGERALD VARIETY" : "270"
},
"01" : {
    "ROASTING\/BLENDING" : "1100",
    "FILLING" : "1200",
    "PACKAGING" : "1300"
},
```

Production Schedule Maintenance



Polar Beverage Production Schedule Maintenance
DEV Facility: WO; Work Ctr: 20

Env=DEV; DB=LXTSTF
User: JVALANCE [logout]
Apr 04, 2014 @ 01:25:11 am

Planned Orders – from Fri 04/04/2014 through Tue 04/15/2014

IT	GTech	Family	Pack	Brand	Type	Item#	Description	Due Date	Plan	Hrs	OH	Avail	Reschedule
B	5248	POLAR	2LS	POLAR	P	1005207	POLAR 2L CREAM SHEL	Apr 07, 2014	878	.49	0	0	
B	3000	POLAR	2LS	POLAR	P	1005220	POLAR 2L SELTZER SHEL *	Apr 07, 2014	4000	2.26	0	0	
B	3000	PRIVT	2LS	DEMOU	P	1006420	MKTBSKT 2L SELTZER	Apr 07, 2014	1500	.85	0	0	
B	5149	PRIVT	2L	S & S	P	1018609	S/S 2L COLA BOX	Apr 07, 2014	3407	1.93	0	0	
B	4072	PRIVT	2L	S & S	P	1018668	S/S 2L DT LEM	Apr 07, 2014	1363	.77	0	0	
B	4075	PRIVT	2L	S & S	P	1018655	S/S 2L DIET ROOTBEER BOX	Apr 07, 2014	1296	.73	0	0	
B	4066	PRIVT	2L	S & S	P	1018654	S/S 2L DT COLA BOX	Apr 07, 2014	2271	1.28	0	0	

Daily Orders for Monday, Mar 31, 2014
Double click rows or headings to expand/collapse details

⊞	Seq#	Item#	Description	Qty	Order#	Type
⊞	10	1012517	CUMBYS 2L TEA/LEMONADE	1183	308360	S
⊞	20	1006417	MKTBSKT 2L HF TEA/HF LMN	394	308361	S
⊞	30	1006424	MKTBSKT 2L TEA	568	308362	S
⊞	40	1006425	CNTYTIME 2L LEMONADE	1420	308363	S
⊞	50	1006126	CNTYTIME 2L PNKLEMAD	710	308364	S
⊞	60	1003435	HANFD 2L LIME SELTZ	1235	308366	S
⊞	70	1003430	HANFD 2L RASLM SELTZ	1475	308367	S
⊞	80	1006430	MKTBSKT 2L RASP LM SLTZR	1136	308368	S
⊞	90	1018661	S/S 2L DT RASPALE BOX	1311	308369	S
⊞	100	1018662	S/S 2L DIET CREAM BOX	1192	308370	S
⊞	110	1006820	A&W 2L DT CREAM	1817	308371	S
⊞	120	1018655	S/S 2L DIET ROOTBEER BOX	1296	308372	S
⊞	130	1008855	MY ESENTALS 2L DT RTBEER	634	308373	S

Weekly Schedule – Line#: 20 – WORCESTER LINE #2 ; Capacity: 20.5 hrs; No. of shifts: 2; Hrs/shift: 10.25

Layout - 3 panels

- Planned Orders
  - Orders not yet scheduled (across top)
- Weekly Schedule (actually 12 days)
  - Visual representation of manfacturing schedule
  - Orders are represented as columns of stacked blocks for each day
  - Height of each block proportional to quantity/time to complete
  - Horizontal lines show 1st & 2nd shift capacity
  - Orders over capacity are red
- Daily Panel
  - Shows details of one day's orders in tabular format
    - Click weekly schedule column heading to load into daily panel
  - Each order, or entire table can be expanded to show more detail

Update Features

- Planned Orders
  - Can be dragged / dropped into weekly schedule, on a specific day
    - Changes order from ‘planned’ to ‘firm planned’
- Weekly Schedule
  - Orders can be dragged to a different day
  - Allows leveling of schedule, to correct over capacity scheduling
- Daily Panel
  - Rearrange order priorities by drag/drop
  - Change quantities
  - Remove from schedule (firm planned back to planned)
  - Change work center (i.e. production line)

Database Interface

- All data loaded directly from BPCS tables onto screens
- All changes immediately reflected in the weekly schedule grid
  - Client-side only
  - Javascript object, not database
- Upon 1st change, “Save Changes” button appears
- No changes to database until Save button clicked
  - Then immediately updates BPCS tables
  - Reloads screen from BPCS database, showing changes applied
- If user leaves screen with changes pending, pop-up confirmation dialog

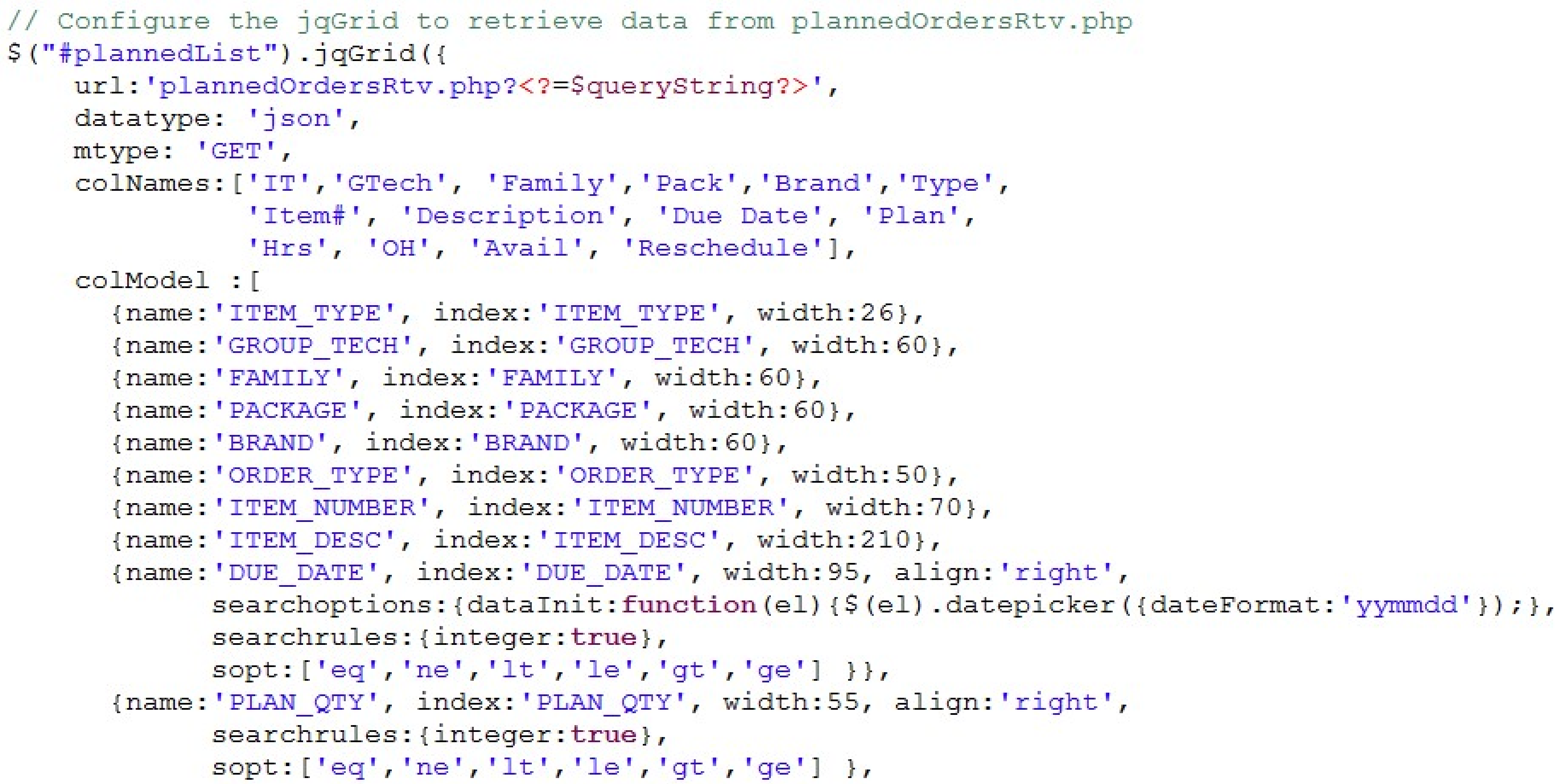
Technologies Employed

- Server side
  - PHP
  - Zend Server 6
  - Zend Framework 2 components
  - BPCS database
- Client side
  - JavaScript
  - jQuery
  - jQuery UI
  - jqGrid (planned orders list)
  - JSON (data transfer format)
  - Ajax (to retrieve data for planned orders)

Planned Orders Panel

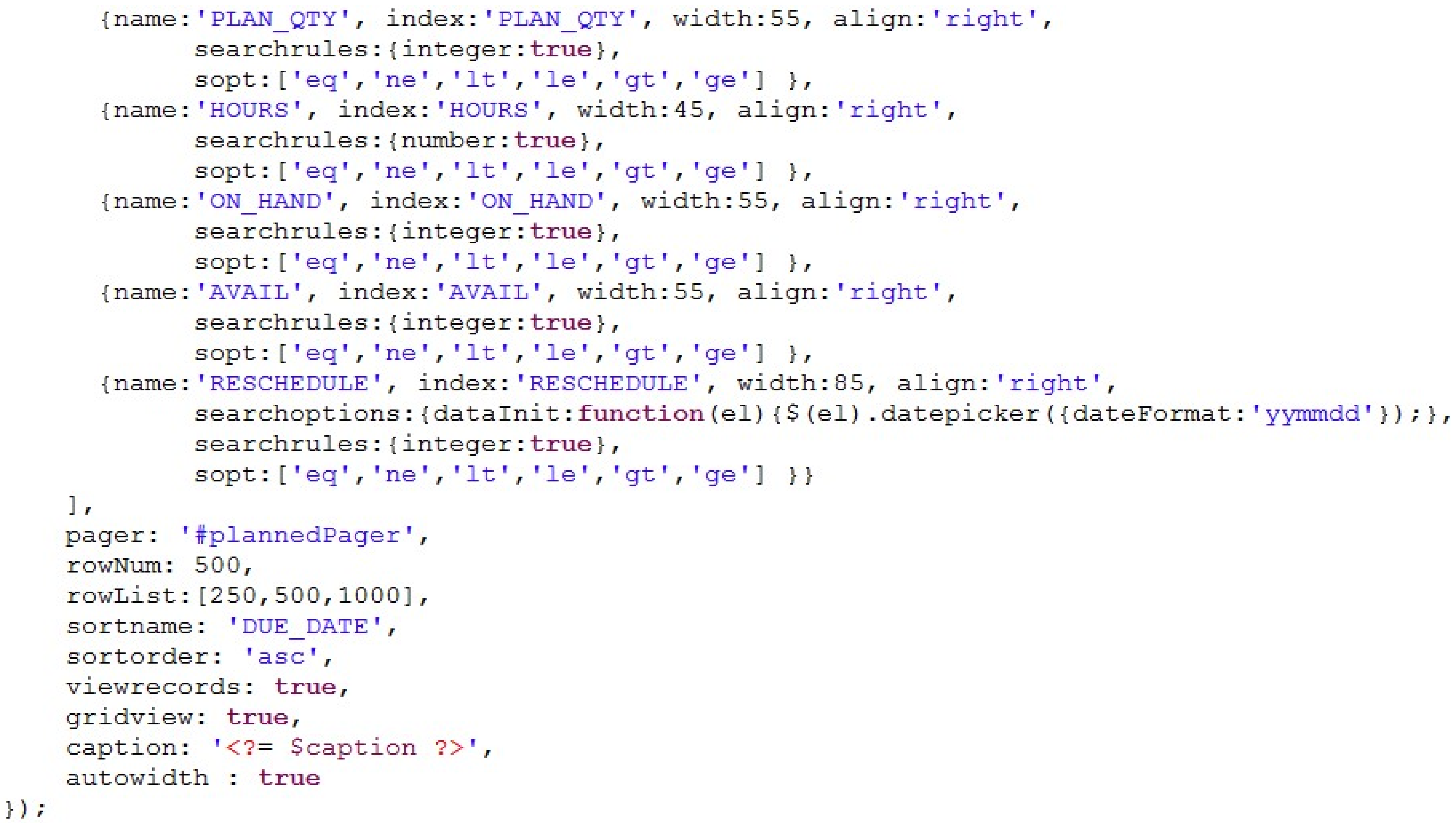
- jqGrid
  - Built on jQuery and jQuery UI
  - http://www.trirand.com/blog/jqgrid/jqgrid.html
  - Builds rich-UI HTML tables with **MANY** capabilities:
    - Pagination and/or scroll bar
    - Searching
    - Column Sorting
    - Row grouping
    - many more…
  - Simple interface, using jQuery syntax
  - Call a server script (e.g., PHP), to retrieve data, passing a list of parameters as JSON object
  - Accepts returned data as JSON, XML, JavaScript array
  - Can combine with other jQuery UI features, such as drag/drop

jqGrid usage



```
// Configure the jqGrid to retrieve data from plannedOrdersRtv.php
$("#plannedList").jqGrid({
    url:'plannedOrdersRtv.php?<?=$queryString?>',
    datatype: 'json',
    mtype: 'GET',
    colNames:['IT','GTech', 'Family','Pack','Brand','Type',
              'Item#', 'Description', 'Due Date', 'Plan',
              'Hrs', 'OH', 'Avail', 'Reschedule'],
    colModel :[
      {name:'ITEM_TYPE', index:'ITEM_TYPE', width:26},
      {name:'GROUP_TECH', index:'GROUP_TECH', width:60},
      {name:'FAMILY', index:'FAMILY', width:60},
      {name:'PACKAGE', index:'PACKAGE', width:60},
      {name:'BRAND', index:'BRAND', width:60},
      {name:'ORDER_TYPE', index:'ORDER_TYPE', width:50},
      {name:'ITEM_NUMBER', index:'ITEM_NUMBER', width:70},
      {name:'ITEM_DESC', index:'ITEM_DESC', width:210},
      {name:'DUE_DATE', index:'DUE_DATE', width:95, align:'right',
            searchoptions:{dataInit:function(el){$(el).datepicker({dateFormat:'yymmdd'});},
            searchrules:{integer:true},
            sopt:['eq','ne','lt','le','gt','ge'] }},
      {name:'PLAN_QTY', index:'PLAN_QTY', width:55, align:'right',
            searchrules:{integer:true},
            sopt:['eq','ne','lt','le','gt','ge'] },
```

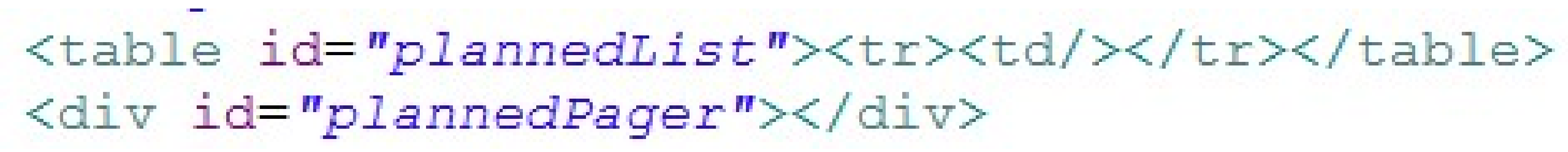
jqGrid usage - continued



```
      {name:'PLAN_QTY', index:'PLAN_QTY', width:55, align:'right',
            searchrules:{integer:true},
            sopt:['eq','ne','lt','le','gt','ge'] },
      {name:'HOURS', index:'HOURS', width:45, align:'right',
            searchrules:{number:true},
            sopt:['eq','ne','lt','le','gt','ge'] },
      {name:'ON_HAND', index:'ON_HAND', width:55, align:'right',
            searchrules:{integer:true},
            sopt:['eq','ne','lt','le','gt','ge'] },
      {name:'AVAIL', index:'AVAIL', width:55, align:'right',
            searchrules:{integer:true},
            sopt:['eq','ne','lt','le','gt','ge'] },
      {name:'RESCHEDULE', index:'RESCHEDULE', width:85, align:'right',
            searchoptions:{dataInit:function(el){$(el).datepicker({dateFormat:'yymmdd'});},
            searchrules:{integer:true},
            sopt:['eq','ne','lt','le','gt','ge'] }}
    ],
    pager: '#plannedPager',
    rowNum: 500,
    rowList:[250,500,1000],
    sortname: 'DUE_DATE',
    sortorder: 'asc',
    viewrecords: true,
    gridview: true,
    caption: '<?= $caption ?>',
    autowidth : true
});
```

jqGrid HTML

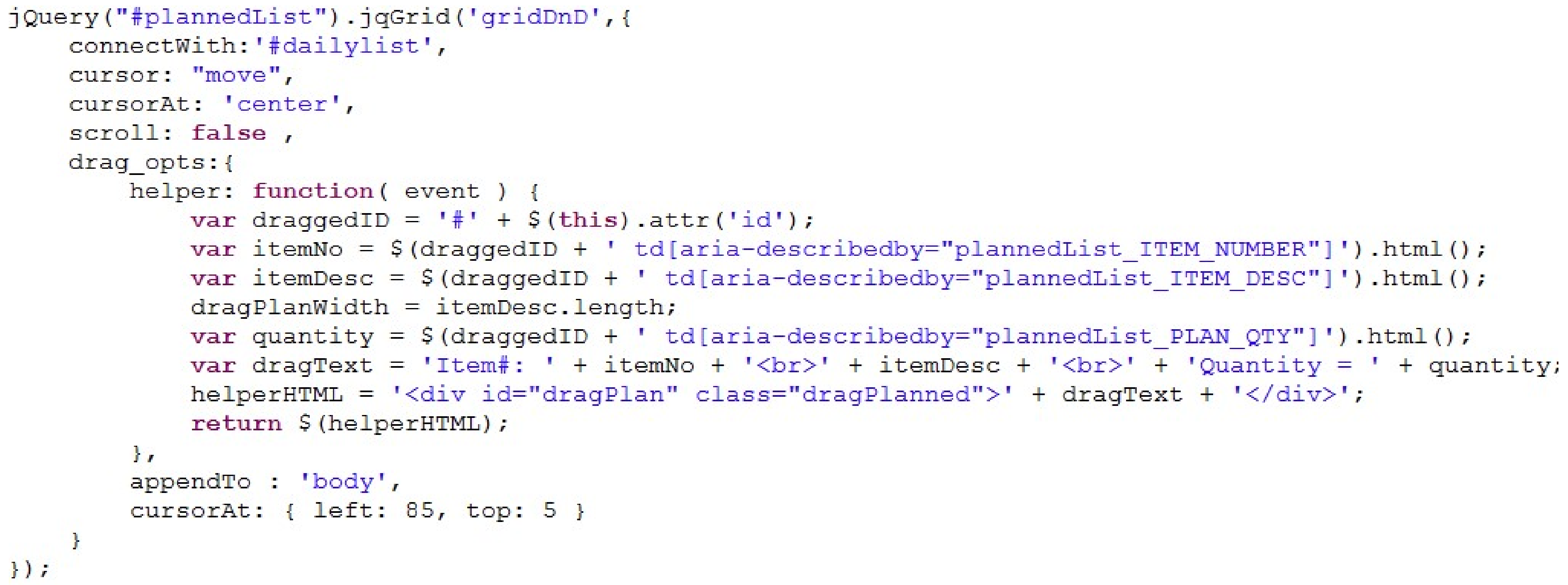
- Entire HTML for Planned Order table (id="plannedList") and paginator (id="plannedPager"):



```
<table id="plannedList"><tr><td/></tr></table>
<div id="plannedPager"></div>
```

Make jqGrid rows draggable onto schedule:

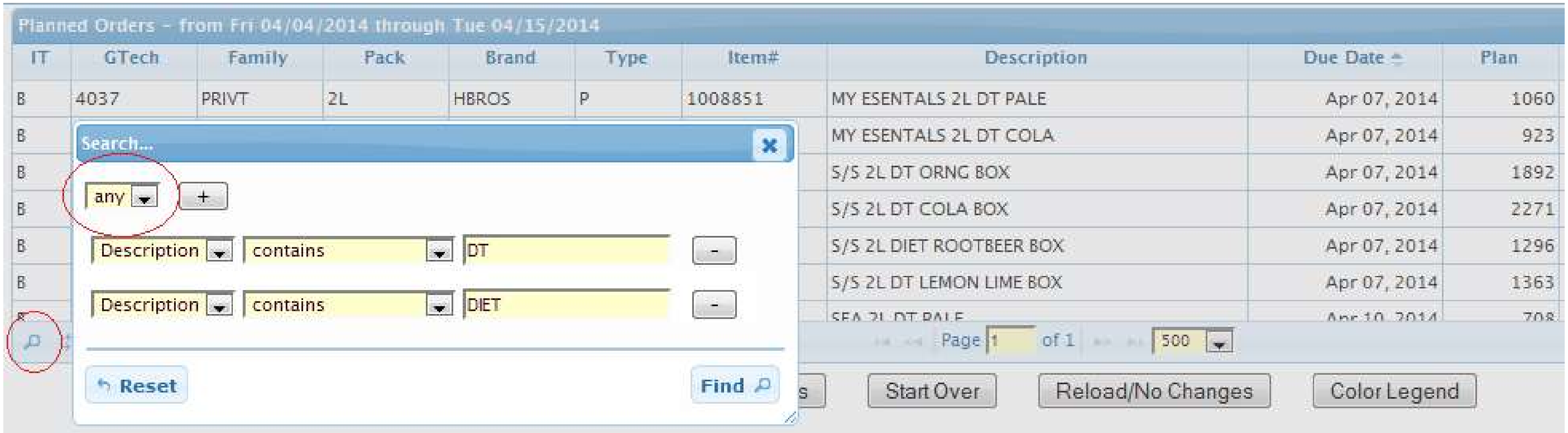
- jqGrid(‘gridDnD’, <options as json>)
- DnD = Drag and Drop



```
jQuery("#plannedList").jqGrid('gridDnD',{
    connectWith:'#dailylist',
    cursor: "move",
    cursorAt: 'center',
    scroll: false ,
    drag_opts:{
        helper: function( event ) {
            var draggedID = '#' + $(this).attr('id');
            var itemNo = $(draggedID + ' td[aria-describedby="plannedList_ITEM_NUMBER"]').html();
            var itemDesc = $(draggedID + ' td[aria-describedby="plannedList_ITEM_DESC"]').html();
            dragPlanWidth = itemDesc.length;
            var quantity = $(draggedID + ' td[aria-describedby="plannedList_PLAN_QTY"]').html();
            var dragText = 'Item#: ' + itemNo + '<br>' + itemDesc + '<br>' + 'Quantity = ' + quantity;
            helperHTML = '<div id="dragPlan" class="dragPlanned">' + dragText + '</div>';
            return $(helperHTML);
        },
        appendTo : 'body',
        cursorAt: { left: 85, top: 5 }
    }
});
```

jqGrid - Search

- Click magnifier
- Can do any (OR) or all (AND)



Planned Orders - from Fri 04/04/2014 through Tue 04/15/2014

IT	GTech	Family	Pack	Brand	Type	Item#	Description	Due Date	Plan
B	4037	PRIVT	2L	HBROS	P	1008851	MY ESENTALS 2L DT PALE	Apr 07, 2014	1060
B							MY ESENTALS 2L DT COLA	Apr 07, 2014	923
B							S/S 2L DT ORNG BOX	Apr 07, 2014	1892
B							S/S 2L DT COLA BOX	Apr 07, 2014	2271
B							S/S 2L DIET ROOTBEER BOX	Apr 07, 2014	1296
B							S/S 2L DT LEMON LIME BOX	Apr 07, 2014	1363
B							SEA 2L DT PALE	Apr 10, 2014	708

Search...

any

Description	contains	DT
Description	contains	DIET

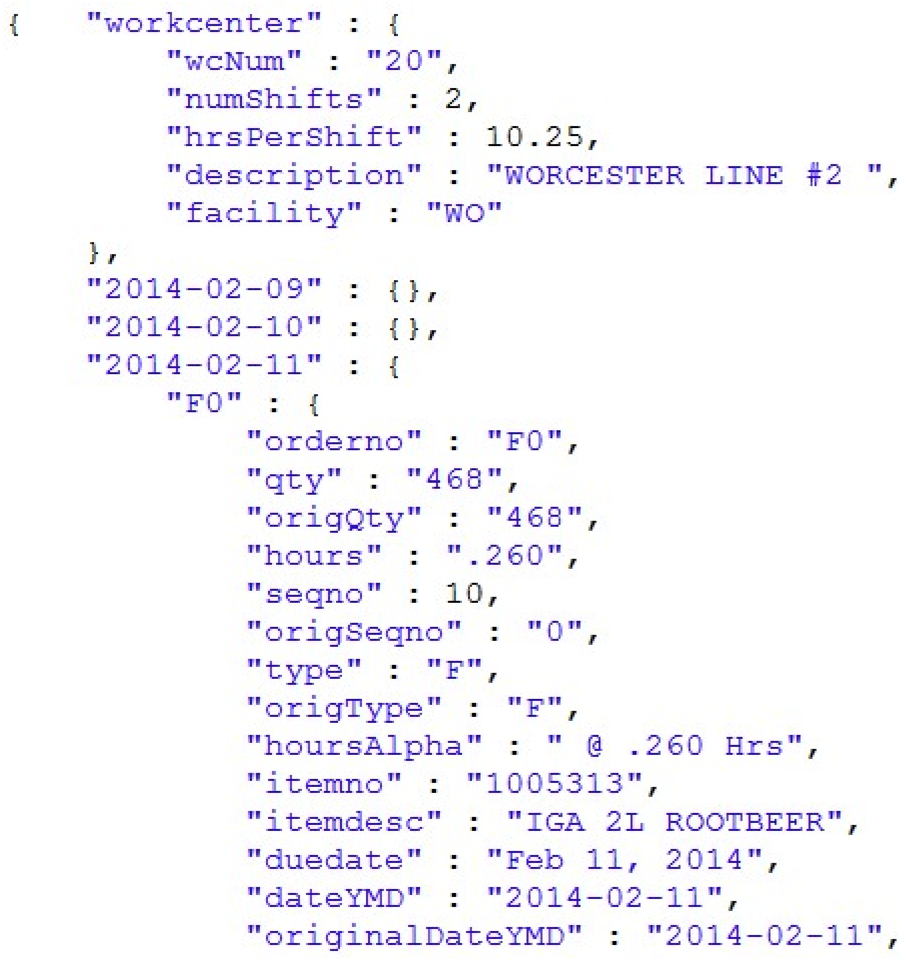
Weekly Schedule

- 12 columns showing schedule for the production line, 12 days out from specified date
- *n* order blocks, stacked. Height proportional to time.
- Entire panel built on nested <div> tags
  - 1 <div> for the entire schedule panel
  - 12 <div>s for the columns for each day
  - *n* <div>s for the orders in each day
  - Every <div> has unique id attribute
- CSS settings are computed in JavaScript and applied using jQuery to <div>s for height, width, position, color, etc.

Weekly Schedule - Data Retrieval

Data for schedule retrieved from PHP

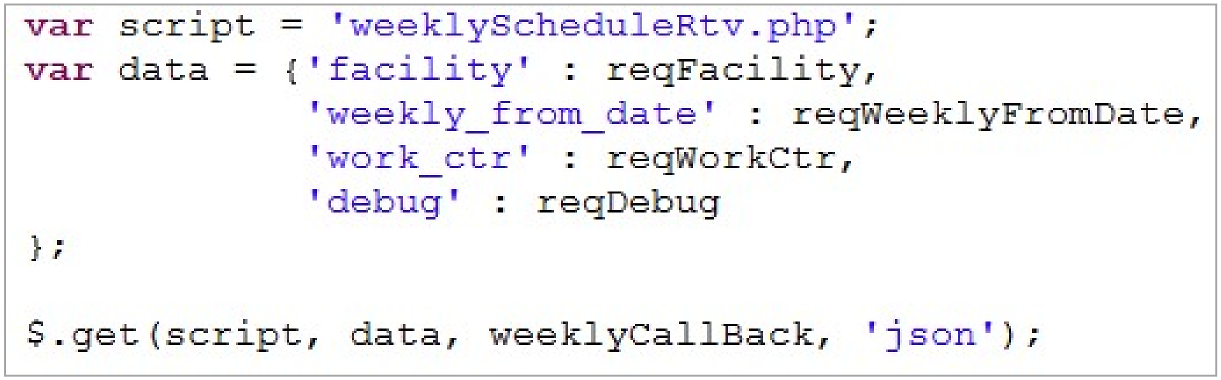
- PHP reads DB2, builds multi-dimensional array by date, order
- `echo` array using PHP built-in function `json_encode($mdArray)`
- JavaScript automatically builds nested object for schedule from the JSON returned by PHP



```
{   "workcenter" : {
        "wcNum" : "20",
        "numShifts" : 2,
        "hrsPerShift" : 10.25,
        "description" : "WORCESTER LINE #2 ",
        "facility" : "WO"
    },
    "2014-02-09" : {},
    "2014-02-10" : {},
    "2014-02-11" : {
        "F0" : {
            "orderno" : "F0",
            "qty" : "468",
            "origQty" : "468",
            "hours" : ".260",
            "seqno" : 10,
            "origSeqno" : "0",
            "type" : "F",
            "origType" : "F",
            "hoursAlpha" : " @ .260 Hrs",
            "itemno" : "1005313",
            "itemdesc" : "IGA 2L ROOTBEER",
            "duedate" : "Feb 11, 2014",
            "dateYMD" : "2014-02-11",
            "originalDateYMD" : "2014-02-11",
```

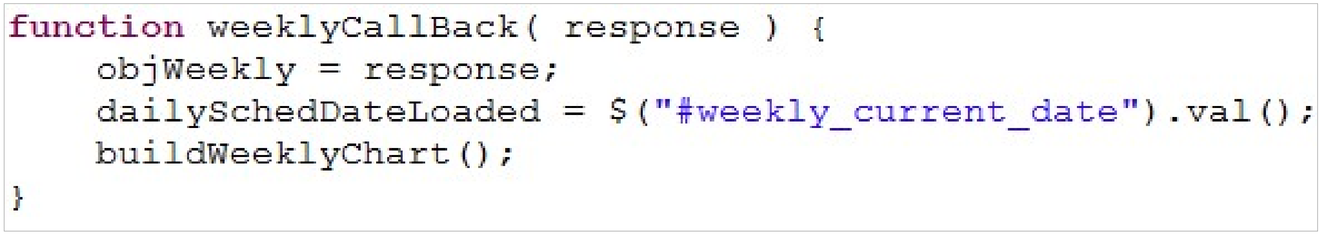
Weekly Schedule - Ajax

Ajax call to retrieve schedule:



```
var script = 'weeklyScheduleRtv.php';
var data = {'facility' : reqFacility,
            'weekly_from_date' : reqWeeklyFromDate,
            'work_ctr' : reqWorkCtr,
            'debug' : reqDebug
};

$.get(script, data, weeklyCallBack, 'json');
```

Callback function for Ajax response:



```
function weeklyCallBack( response ) {
    objWeekly = response;
    dailySchedDateLoaded = $("#weekly_current_date").val();
    buildWeeklyChart();
}
```

buildWeeklyChart() is called when anything changes, to re-paint the screen

weeklySchedule.js

- Big JavaScript file, with many functions (~1,000 lines of JS code)
- Handles most of the JS and jQuery magic
- Builds the weekly and daily panels

`objWeekly`

- Global variable which is a data model of the weekly grid
- Loaded initially by PHP Ajax call
  - automatically initialized from JSON returned by PHP
- `buildWeeklyChart()`
  - re-paints the weekly grid when data changes
    - initial page load from DB2/PHP
    - drag/drop orders on the schedule
    - change quantities, work centers in daily panel
    - add or remove planned orders from schedule

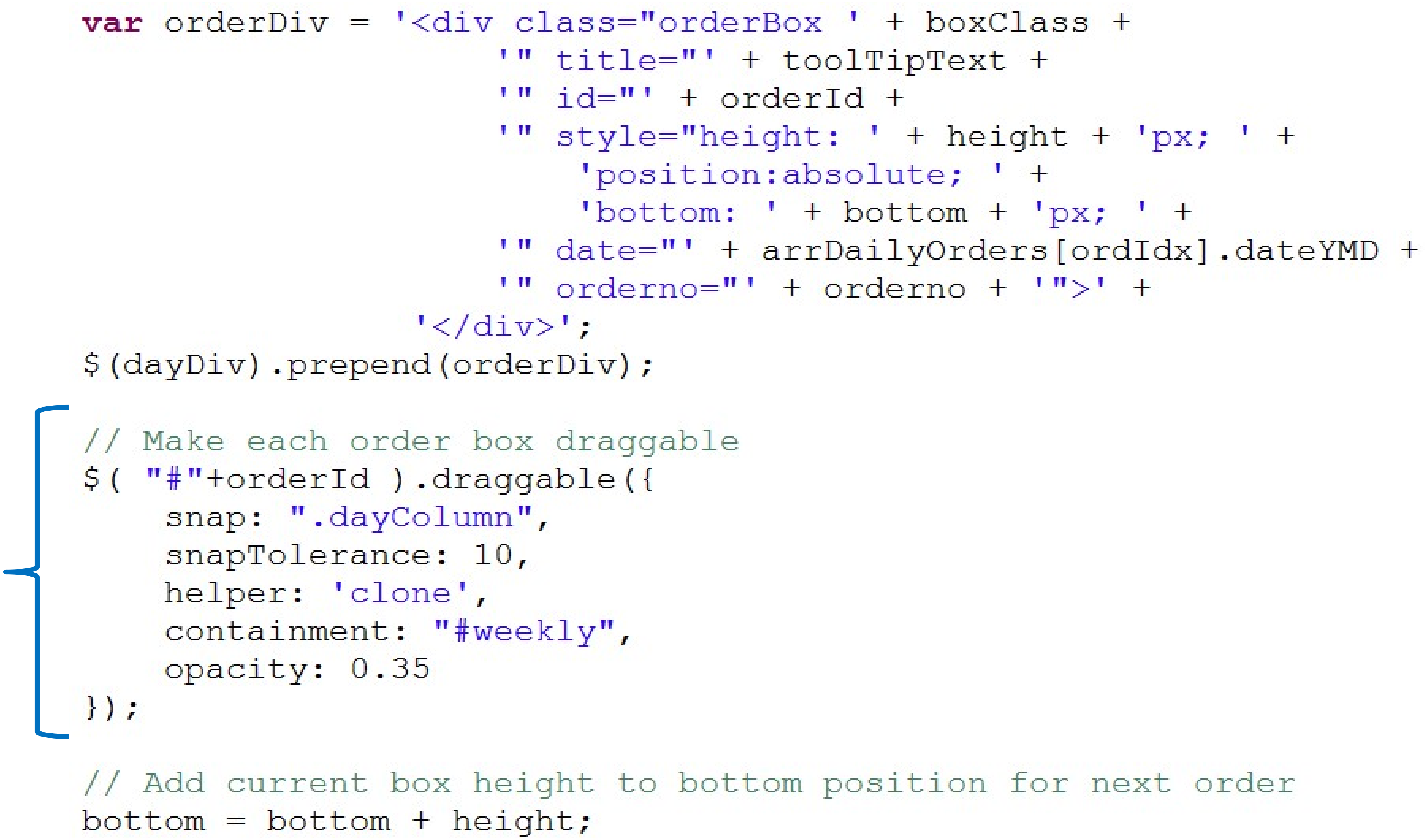
`buildWeeklyChart()`

- Empties the weekly panel and reset related variables
- Loop through the days in `objWeekly`, passing the day's orders into `addDayToWeek()`
- After weekly schedule is built, calls `loadDailySchedule()` to load the daily panel
- Add some behaviors to the panels
  - hover behavior
  - drag/drop
  - tooltips

addDayToWeek(dayNo,orders,oDate)

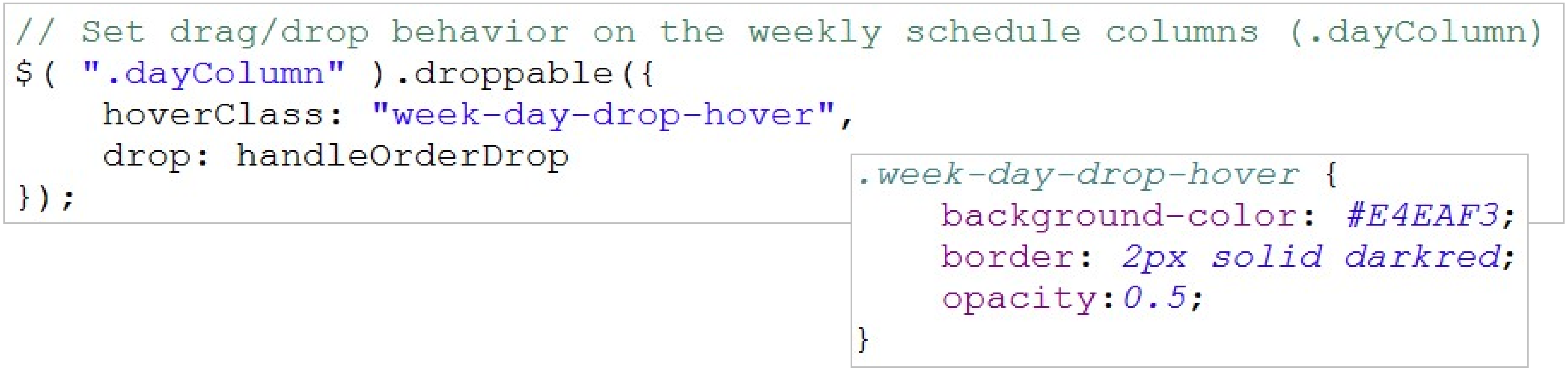
- Receives the orders for one day as an object
- Loops through each order for the day
- Builds the stack of blocks representing those orders
- Each order is visually rendered as a <div>
  - <div orderno="308802" id="ord308802" date="2014-04-07" style="**height: 42px**; position:absolute; **bottom: 23px**;" class="orderBox shop-under-cap ui-draggable"></div>
  - var **height** = Math.round(orderHours * hoursToPixelMult);
  - Next one: **bottom** = bottom + height;
- Use jQuery().prepend() function to add order div to schedule:
  - $(dayDiv).prepend(orderDiv);

Make orders draggable



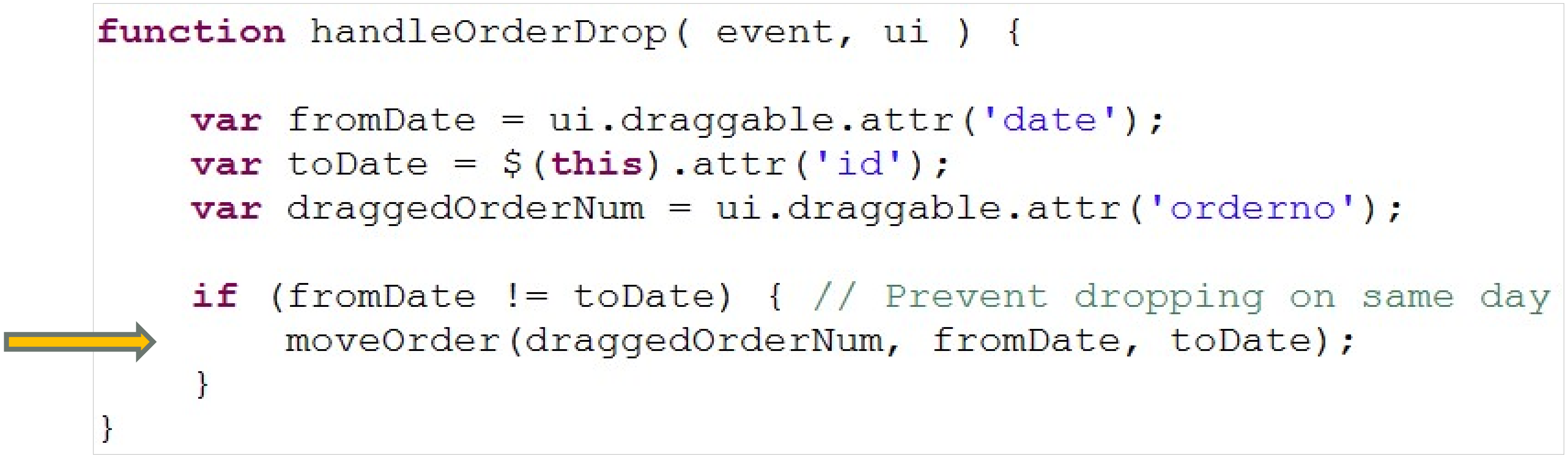
```
var orderDiv = '<div class="orderBox ' + boxClass +
                    '" title="' + toolTipText +
                    '" id="' + orderId +
                    '" style="height: ' + height + 'px; ' +
                        'position:absolute; ' +
                        'bottom: ' + bottom + 'px; ' +
                    '" date="' + arrDailyOrders[ordIdx].dateYMD +
                    '" orderno="' + orderno + '">' +
                '</div>';
$(dayDiv).prepend(orderDiv);

// Make each order box draggable
$( "#"+orderId ).draggable({
    snap: ".dayColumn",
    snapTolerance: 10,
    helper: 'clone',
    containment: "#weekly",
    opacity: 0.35
});

// Add current box height to bottom position for next order
bottom = bottom + height;
```

Order Drop Handler



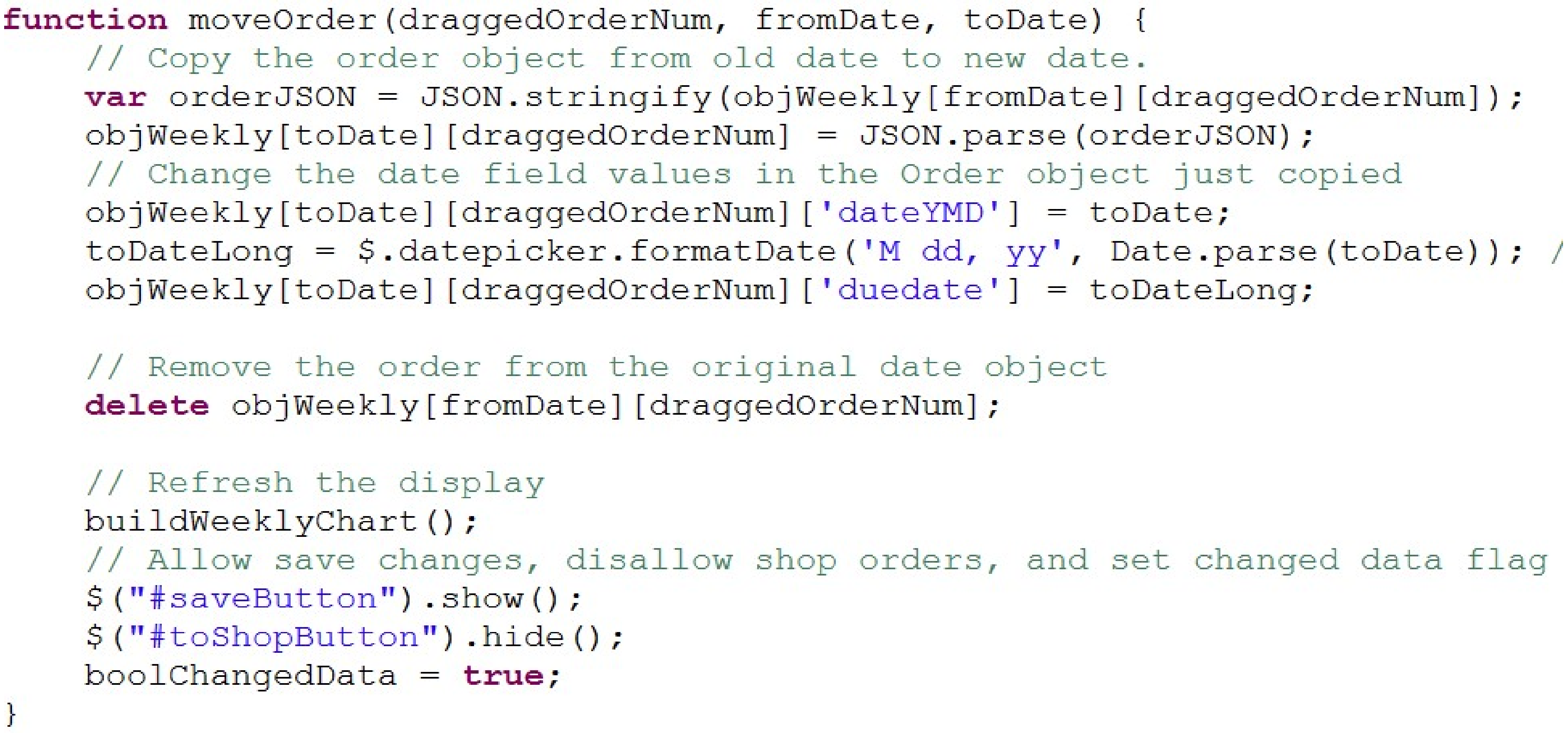
```
// Set drag/drop behavior on the weekly schedule columns (.dayColumn)
$( ".dayColumn" ).droppable({
    hoverClass: "week-day-drop-hover",
    drop: handleOrderDrop
});
```

```
.week-day-drop-hover {
    background-color: #E4EAF3;
    border: 2px solid darkred;
    opacity:0.5;
}
```



```
function handleOrderDrop( event, ui ) {

    var fromDate = ui.draggable.attr('date');
    var toDate = $(this).attr('id');
    var draggedOrderNum = ui.draggable.attr('orderno');

    if (fromDate != toDate) { // Prevent dropping on same day
        moveOrder(draggedOrderNum, fromDate, toDate);
    }
}
```

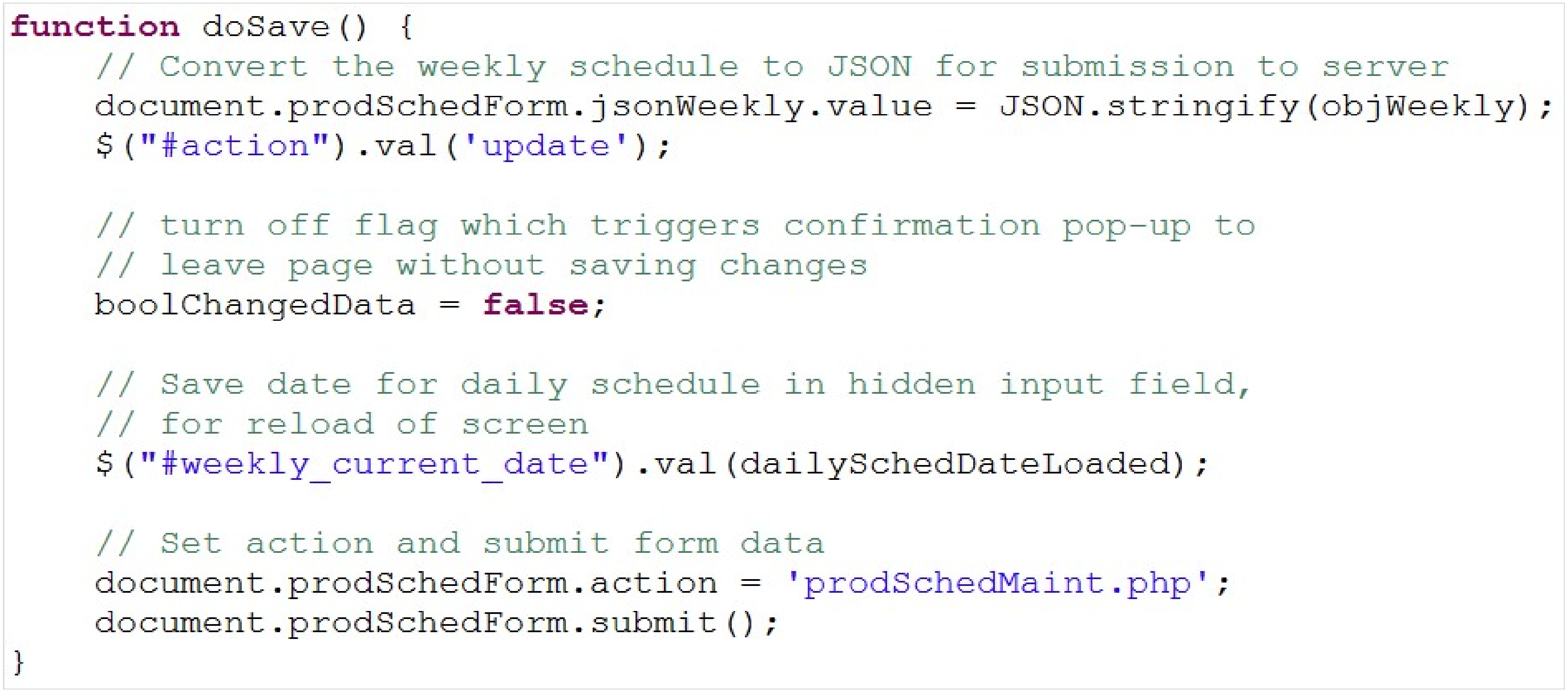
Moving an Order to Another Day



```
function moveOrder(draggedOrderNum, fromDate, toDate) {
    // Copy the order object from old date to new date.
    var orderJSON = JSON.stringify(objWeekly[fromDate][draggedOrderNum]);
    objWeekly[toDate][draggedOrderNum] = JSON.parse(orderJSON);
    // Change the date field values in the Order object just copied
    objWeekly[toDate][draggedOrderNum]['dateYMD'] = toDate;
    toDateLong = $.datepicker.formatDate('M dd, yy', Date.parse(toDate)); /
    objWeekly[toDate][draggedOrderNum]['duedate'] = toDateLong;

    // Remove the order from the original date object
    delete objWeekly[fromDate][draggedOrderNum];

    // Refresh the display
    buildWeeklyChart();
    // Allow save changes, disallow shop orders, and set changed data flag
    $("#saveButton").show();
    $("#toShopButton").hide();
    boolChangedData = true;
}
```

Save Changes button

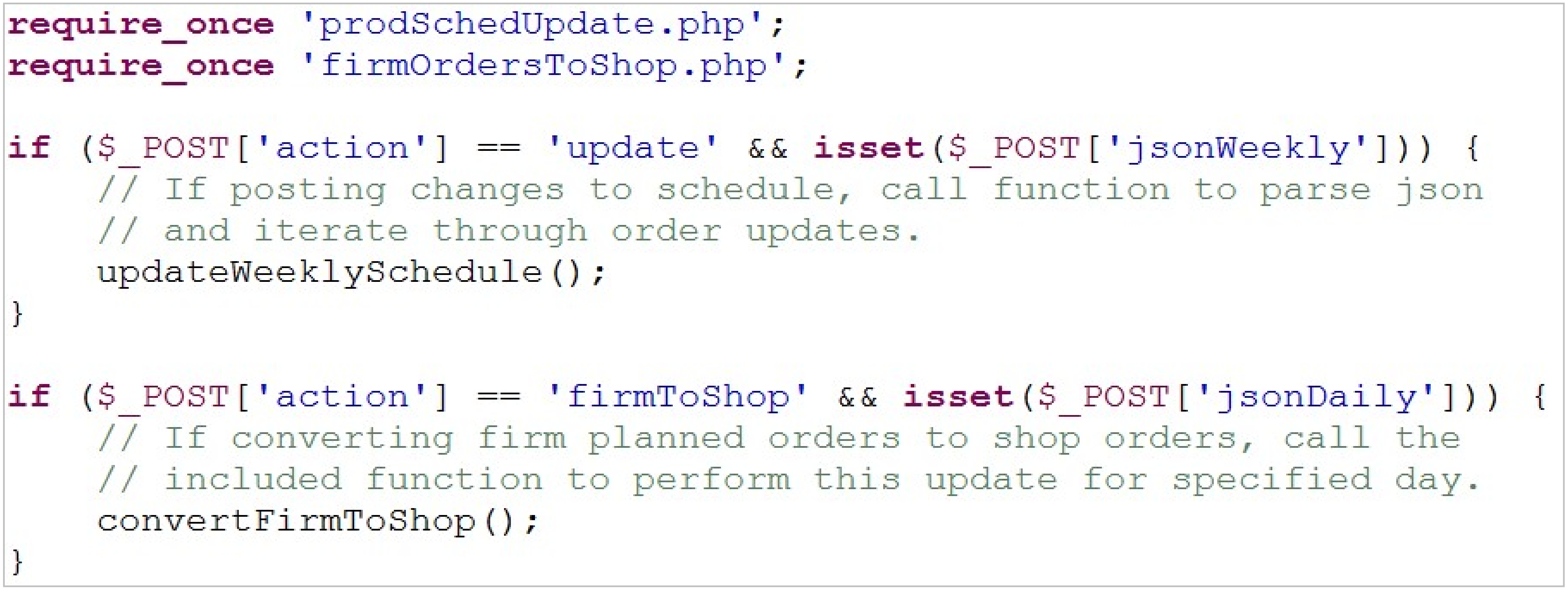
- onclick=“doSave()”



```
function doSave() {
    // Convert the weekly schedule to JSON for submission to server
    document.prodSchedForm.jsonWeekly.value = JSON.stringify(objWeekly);
    $("#action").val('update');

    // turn off flag which triggers confirmation pop-up to
    // leave page without saving changes
    boolChangedData = false;

    // Save date for daily schedule in hidden input field,
    // for reload of screen
    $("#weekly_current_date").val(dailySchedDateLoaded);

    // Set action and submit form data
    document.prodSchedForm.action = 'prodSchedMaint.php';
    document.prodSchedForm.submit();
}
```

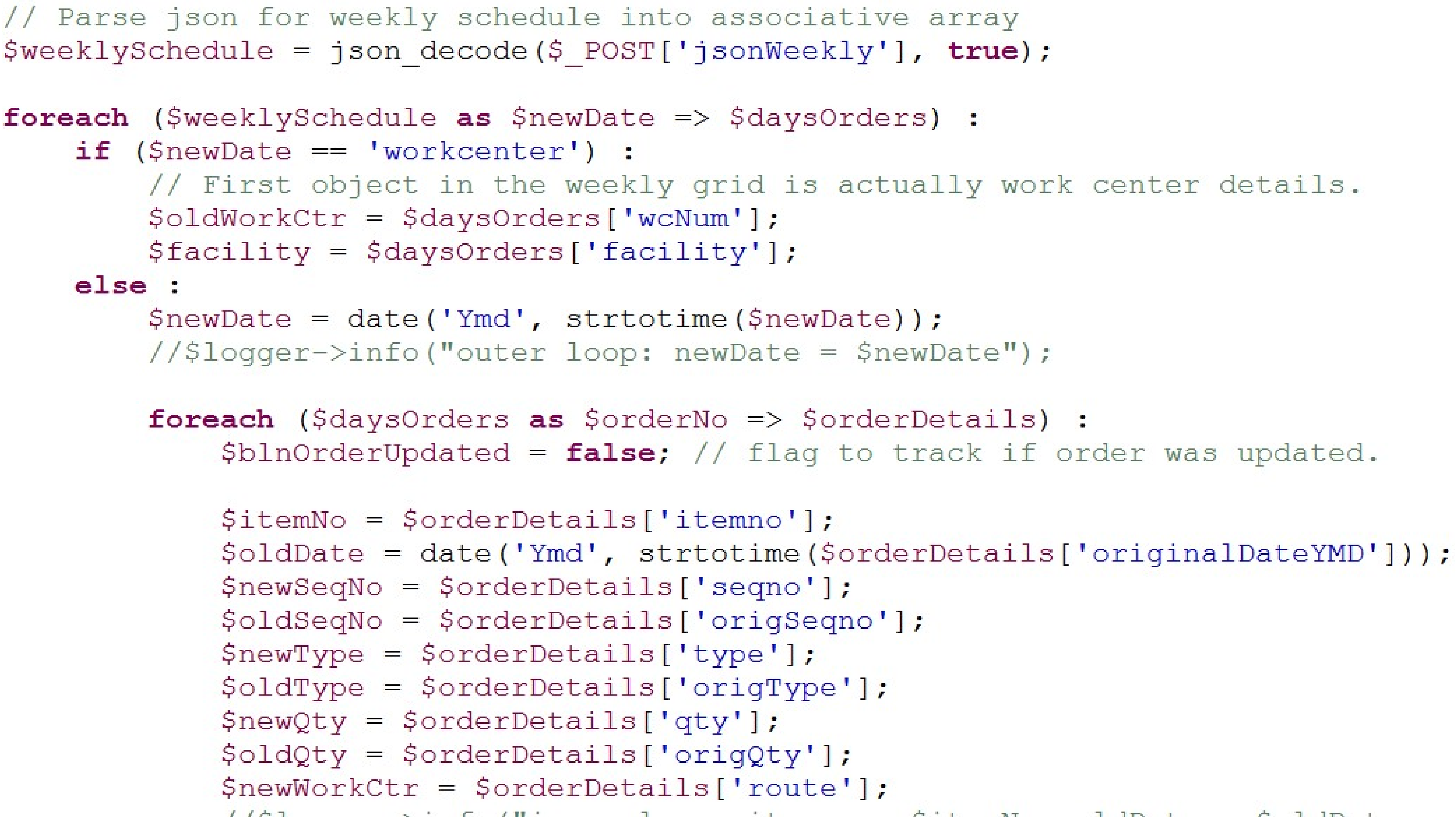
Saving changes - PHP side

- In prodSchedMaint.php:



```php
require_once 'prodSchedUpdate.php';
require_once 'firmOrdersToShop.php';

if ($_POST['action'] == 'update' && isset($_POST['jsonWeekly'])) {
    // If posting changes to schedule, call function to parse json
    // and iterate through order updates.
    updateWeeklySchedule();
}

if ($_POST['action'] == 'firmToShop' && isset($_POST['jsonDaily'])) {
    // If converting firm planned orders to shop orders, call the
    // included function to perform this update for specified day.
    convertFirmToShop();
}
```

Using json_decode()



```
// Parse json for weekly schedule into associative array
$weeklySchedule = json_decode($_POST['jsonWeekly'], true);

foreach ($weeklySchedule as $newDate => $daysOrders) :
    if ($newDate == 'workcenter') :
        // First object in the weekly grid is actually work center details.
        $oldWorkCtr = $daysOrders['wcNum'];
        $facility = $daysOrders['facility'];
    else :
        $newDate = date('Ymd', strtotime($newDate));
        //$logger->info("outer loop: newDate = $newDate");

        foreach ($daysOrders as $orderNo => $orderDetails) :
            $blnOrderUpdated = false; // flag to track if order was updated.

            $itemNo = $orderDetails['itemno'];
            $oldDate = date('Ymd', strtotime($orderDetails['originalDateYMD']));
            $newSeqNo = $orderDetails['seqno'];
            $oldSeqNo = $orderDetails['origSeqno'];
            $newType = $orderDetails['type'];
            $oldType = $orderDetails['origType'];
            $newQty = $orderDetails['qty'];
            $oldQty = $orderDetails['origQty'];
            $newWorkCtr = $orderDetails['route'];
```

Perform updates, etc…

THE END

More info…

References - More Information

- jQuery home:
  - https://jquery.org/
- jQuery UI:
  - http://jqueryui.com/
  - Check out the demos
- jqGrid:
  - http://www.trirand.com/blog/?page_id=6 = download
  - http://trirand.com/blog/jqgrid/jqgrid.html = demos
- JSON
  - http://en.wikipedia.org/wiki/JSON
- wsSchools
  - http://www.w3schools.com/

Contact Information

John Valance
Division 1 Systems

johnv@div1sys.com
802-355-4024

http://www.div1sys.com

<div1>

division 1 systems

Thanks for coming!