Paste the <html> to </html> script below into to Alter Response field on the Advanced Step when editing your service configuration. The pattern is: ^([\s\S]*?)$.

We also recommend disabling the response label under the “Enable custom display configuration” option using this structured text: {“response_label”:””}

```
<html>
   <head><meta name="viewport" content="initial-scale=1">
      <style>
         #kKitItems, #kBatchItems { margin-top:-15px; }
         .info { border:2px solid black; padding: 2px 15px; border-radius: 15px }
      </style>
   </head>
   <body>
   <!-- Use this Alter Response RegEx "^([\s\S]*?)$" with replacement set to "$1" -->
   <script type="text/codereadr" id="regex_replacement">$1</script>
   <!-- script type="text/codereadr" id="regex_replacementItemList">
      { "Apple": 3, "Banana": 1, "Orange": 2}
   </script>
   <script type="text/codereadr" id="regex_replacementExampleState">
      {
         "totalCnt": 0,
         "batch": {
            "packed": { "Apple": 0, "Banana": 0, "Orange": 0},
            "id": "BID1580202543058",
            "qty": 4
         },
         "session": {
            "orderId": "wo001",
            "kitId" : "pl001",
            "totalQty": 15,
            "batchQty": 4,
            "kit": { "Apple": 3, "Banana": 1, "Orange": 2}
         },
         "message": "Item Added"
      }
   </script -->

   <div id="itemListDiv" style="display:none"><ul id="itemList"></ul></div>

   <div id="kValidState">
```

```
    <h3 id="kMsg" style="padding-top:10px">PLACEHOLDER_MSG</h3>

    <div class="info">
        <h4><span id="kTotalKits">N of M</span> Kits completed</h4>
        <h4>
            <a href="#" onclick="toggleElement('kBatchItems')">
                <span id="kItemTotal">n of qty</span> Items added to batch of
                <span id="kBatchSize">N</span> Kits
            </a>
        </h4>
        <ul id="kBatchItems"></ul>
        <h4><a href="#" onclick="toggleElement('kKitItems')">Kit Contents</a></h4>
        <ul id="kKitItems">
            <li><i>Item (quantity)</i></li>
        </ul>
    </div>
    <p>
    Kit: <b id="kKitId">KIT_ID</b><br>
    Work Order: <b id="kOrderId">ORDER_ID</b><br>
    Current Batch: <b id="kBatchId">BATCH_ID</b>
    </p>
</div>

<script>
function toggleElement(elmId, forceDisplay) {
    var elm = document.getElementById(elmId);
    if (forceDisplay) {
        elm.style.display = forceDisplay;
    } else if (elm.style.display === "none") {
        elm.style.display = "block";
    } else {
        elm.style.display = "none";
    }
}
function isValidState(state) {
    return state.hasOwnProperty('totalCnt') && state.hasOwnProperty('session') &&
    state.session.hasOwnProperty('kit');
}
function getItemCounts(state) {
    // -1 to indicate there was no actualy quantity set
    var counts = {"kitItemCnt": 0, "batchItemQty": -1, "batchItemCnt": 0};
    if (isValidState(state)) {
        counts.kitItemCnt = sumValues(state.session.kit);
```

```
            counts.batchItemQty = counts.kitItemCnt * state.batch.qty;
            counts.batchItemCnt = sumValues(state.batch.packed);
        }
        return counts;
    }
    function sumValues(obj) {
        var sum = 0;
        for( var el in obj ) {
            if( obj.hasOwnProperty( el ) ) {
                sum += parseFloat( obj[el] );
            }
        }
        return sum;
    }

    // i.e. { "Apple": 3, "Banana": 1, "Orange": 2}
    function displayQuantiyList(listElmId, obj) {
        var list = document.getElementById(listElmId);
        for (const item in obj) {
            if (obj.hasOwnProperty(item)) {
                list.innerHTML = list.innerHTML +"<li>"+item+" (" + obj[item] + ")";
            }
        }
    }

    function displayFormattedResponse(data) {
        if (!isValidState(data)) {
            displayQuantiyList("itemList", data);
            toggleElement("itemListDiv", "block");
            toggleElement("kValidState", "none");
            return;
        }
        // See regex_replacementExampleState for valid state JSON
        document.getElementById("kTotalKits").innerHTML = data.totalCnt +" of
"+data.session.totalQty;
        document.getElementById("kMsg").innerHTML = data.message;
        document.getElementById("kOrderId").innerHTML = data.session.orderId;
        document.getElementById("kKitId").innerHTML = data.session.kitId;
        document.getElementById("kBatchId").innerHTML = data.batch.id;
        document.getElementById("kBatchSize").innerHTML = data.batch.qty;
        var sums = getItemCounts(data);
        document.getElementById("kItemTotal").innerHTML = sums.batchItemCnt+" of
"+sums.batchItemQty;
```

```
        displayQuantiyList("kKitItems", data.session.kit);

        var list = document.getElementById("kBatchItems");
        for (const item in data.batch.packed) {
            if (data.batch.packed.hasOwnProperty(item)) {
                var goal = data.session.kit[item] * data.batch.qty;
                var cnt = data.batch.packed[item];
                list.innerHTML = list.innerHTML +"<li>"+item+" (" + cnt + " of "+ goal + ")";
            }
        }
    }

    var responseText = document.getElementById("regex_replacement").innerHTML;
    try {
        var data = JSON.parse(responseText);
        displayFormattedResponse(data);
    } catch (e) {
        console.log("ERROR: "+e);
        document.getElementsByTagName("body")[0].innerHTML = responseText;
    }
    </script>
    </body>
</html>
```