WHITE PAPER

www.progress.com

ACCESSING THE PROGRESS® OPENEDGE® APPSERVER FROM PROGRESS® ROLLBASE® USING JSDO CODE

BY EDSEL GARCIA, PRINCIPAL SOFTWARE ENGINEER, PROGRESS OPENEDGE DEVELOPMENT

PROGRESS

TABLE OF CONTENTS

INTRODUCTION

Progress® Rollbase® provides a simple way to create an application that satisfies critical business needs without investing months in development. Progress® OpenEdge® users can take advantage of this rapid application development by accessing existing Progress OpenEdge data from Progress Rollbase web applications. Progress Rollbase uses JavaScript for HTML event handlers in its client-side development model. To provide access to Progress OpenEdge data, call Progress OpenEdge Mobile JavaScript Data Object (JSDO) code from the HTML event handlers to access the Progress OpenEdge Application Server (AppServer).

The JSDO code is similar to what you would use in an Progress OpenEdge Mobile application. However, when JSDO code is used in a Script Component in Progress Rollbase, the code can call the Rollbase AJAX API by using the `rbf _ *` set of functions, and refer via JavaScript to form components.

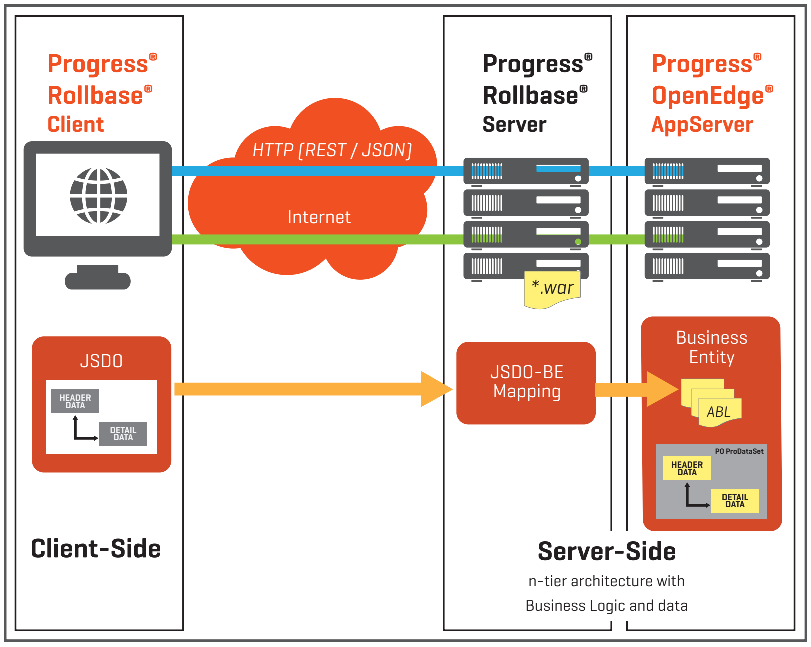


Figure 1. Components of a private cloud environment running Progress Rollbase.

This document describes how to access Progress OpenEdge data using JSDO to call from the client-side in Progress Rollbase.

COMPONENTS OF A PROGRESS ROLLBASE / OPENEDGE APPSERVER ENVIRONMENT

When calling JSDO code on the client-side, either Progress Rollbase Private Cloud or Progress Rollbase Hosted Cloud environments can be used. A Progress Rollbase Private Cloud environment requires a Java-based application server, and uses Apache Tomcat for development and runtime. Progress OpenEdge Mobile Services can be hosted in the same Tomcat instance. Figure 1 diagrams the components within a Progress Rollbase Private Cloud environment.

Components:

- **Client:** You must add code to your application that the JSDO executes on the client-side as part of a Script Component or an event handler in Progress Rollbase.
- **Web Server for REST adapter and Progress OpenEdge Mobile Service:** A JSDO connects to the Mobile Service hosted by the Web server.
- **Rollbase Server:** In a private cloud environment, the Web server for the REST adapter and the Progress OpenEdge Mobile service can be the Tomcat server used by the Progress Rollbase server.
- **Mobile Service:** A Java web application built using Progress Developer Studio for OpenEdge and deployed as a WAR (Web Application Archive) file to a Java-based application server, which provides access to the Progress OpenEdge AppServer via HTTP / HTTPS.
- **OpenEdge AppServer:** An application server which can access the Progress OpenEdge database anywhere on the network. The Progress OpenEdge Mobile service uses a REST adapter component to communicate with the OpenEdge AppServer and execute Progress OpenEdge Advanced Business Language (ABL) logic in a Business Entity class.
- **Mobile Business Entity:** An ABL class that implements the methods (CRUD—create, read, update and delete—operations and user-defined operations) available via the Mobile service. This class accesses the Progress OpenEdge database and executes the ABL logic.

CONFIGURATION

In order to use JSDO code from Rollbase, the following components must be present in your environment:

- JavaScript files for the JSDO code
- Mobile Service
- OpenEdge AppServer and Business Entity class

JAVASCRIPT FILES FOR THE JSDO

The JavaScript files for the JSDO code are `progress.js` and `progress.session.js`, and include both JSDO and Session Objects used to connect to the Progress OpenEdge AppServer via the REST adapter. These JavaScript files can found in the `$DLC/mobile/client/js` subdirectory of the Progress OpenEdge installation that includes the OpenEdge AppServer.

JavaScript files for the JSDO code need to be accessible by the HTML code (client-side) in your Progress Rollbase application. To use them, ensure you have hosted these files on a web server. You can use the **Hosted Files** functionality in Progress Rollbase to host the files as part of your Rollbase environment.

MOBILE SERVICE

To access the Progress OpenEdge AppServer, you must build a Mobile Service using Developer Studio for OpenEdge and deploy the WAR file for the Mobile Service to a Java-based application server. For a private cloud environment, you can use the same instance of Tomcat as Rollbase, or a separate instance. For a hosted cloud environment, the WAR file must be deployed to its own Tomcat instance. During development, the Tomcat instance included with Progress Developer Studio for OpenEdge can be used.

PROGRESS OPENEDGE APPSERVER AND BUSINESS ENTITY CLASS

The OpenEdge AppServer can be located anywhere on the network. The OpenEdge AppServer must be configured to connect to the OpenEdge database by default. The Business Entity class must be available to the OpenEdge AppServer to execute the ABL logic.

ABOUT CROSS-ORIGIN RESOURCE SHARING (CORS)

The HTML files for a Rollbase application run in a web browser, and are hosted by the Rollbase server. If the Mobile service used to access the OpenEdge AppServer is hosted on a different web server, the access to the data is considered a cross-domain access. Web browsers may block the request due to the web browser's same-origin policy.

Mobile services created from Progress Developer Studio for OpenEdge support cross-origin resource sharing (CORS) by default. For additional information on CORS and how to extend it, see the following references:

- "Cross-origin resource sharing (CORS)" in the Progress OpenEdge Mobile Applications book, available in the Progress OpenEdge documentation.
- "Extending CORS support" in the Progress OpenEdge Application Server book, available in the Progress OpenEdge documentation.
- Cross-Origin Resource Sharing Specification: http://enable-cors.org.

USING JSDO CODE IN JAVASCRIPT

You can add JavaScript that calls JSDO code directly from an HTML page to access the OpenEdge AppServer and retrieve specific data. Figure 2 demonstrates calling JSDO code from an HTML page. For a more extensive example, refer to Example 1 in Appendix A.

```
<!DOCTYPE html>
<html>
<head>
    <script src="http://oemobiledemo.progress.com/jsdo/progress.js">
    </script>
    <script src="http://oemobiledemo.progress.com/jsdo/progress.session.js">
    </script>
<script>
    session = new progress.data.Session();
    session.login("http://oemobiledemo.progress.com/MobilityDemoService", "", "");
    session.addCatalog("http://oemobiledemo.progress.com/MobilityDemoService" +
                       "/static/mobile/MobilityDemoService.json");
    jsdo = new progress.data.JSDO({ name: 'dsCustomer' });
    jsdo.subscribe('AfterFill', onAfterFillCustomers, this);
    jsdo.fill();
    function onAfterFillCustomers(jsdo, success, request) {
        jsdo.eCustomer.foreach(function(customer) {
        document.write(
            customer.data.CustNum + ' ' + customer.data.Name + '<br>');
        });
    }

</script>
</head>
<body>
</body>
</html>
```

Figure 2. Sample code to call JSDO code from an HTML page.

LINES 04 – 05:

The example demonstrates how to include `progress.session.js` and `progress.js` JavaScript files provided by Progress. These files provide the Session Object, and the JSDO code, respectively. The Session Object is used to establish a Session with the REST adapter to access the OpenEdge back-end. The example includes JavaScript files from the `/jsdo` subdirectory on the same server where the HTML is hosted.

LINES 08 – 09:

The parameters `<serviceURI>` and `<catalogURI>` specify the access to the Mobile Service, which handles communication with the OpenEdge AppServer. The parameter `<serviceURI>` is the base URI for the service. The parameter `<catalogURI>` corresponds to the path of the JSDO catalog, which is provided as a JSON file. The JSDO catalog defines the schema and operations, as well as REST URIs for the resources provided by the Mobile Service. You can use the JSDO API to call operations on the OpenEdge AppServer instead of calling REST URIs.

You can use Progress Developer Studio for OpenEdge to build the Mobile Service for your application. The code instantiates the Session and authenticates with the application server. This example uses anonymous access.

LINE 11:

The code instantiates the JSDO code as a resource specified in the JSDO catalog. The example uses `‘dsCustomer.’`

LINE 12:

The `subscribe()` method is used to define a callback function for the `AfterFill` event which is executed when the `fill()` method completes.

LINE 14:

The `fill()` method is called to read records from the OpenEdge AppServer. The `fill()` method is part of the JSDO API and performs an asynchronous READ operation on the Mobile Service.

The JSDO provides several methods that can be used to read and update data in the OpenEdge AppServer as well as the local storage for the JSDO code. In addition, developers can add their own methods to a Business Entity to call directly from JavaScript with custom parameters. These methods are also called INVOKE methods. Refer to the Progress OpenEdge Mobile Applications book in the Progress OpenEdge documentation for additional information on the JSDO, its methods, and properties.

LINE 16-21:

The function `onAfterFillCustomers()` is called when the `fill()` completes. The code scans all the customer records one at a time by using the JSDO API `foreach()`. For each customer record, the code calls the JavaScript `document.write()` function to update the content of the HTML page with the customer number and the customer name.

CALLING THE JSDO FROM ROLLBASE

Rollbase includes support for client-side development and customization in the form of HTML event handlers and the Rollbase AJAX API, accessible via the `rbf _ *` set of functions.

```
01 <select id=”cust _ select” size=”5” style=”font-family: monospace”>
02 </select>
03 <script src=”http://oemobiledemo.progress.com/jsdo/progress.js”>
04 </script>
05 <script src=”http://oemobiledemo.progress.com/jsdo/progress.session.js”>
06 </script>
07 <script>
08     var session = new progress.data.Session();
09     session.login(“http://oemobiledemo.progress.com/MobilityDemoService”, “”, “”);
10     session.addCatalog(“http://oemobiledemo.progress.com/MobilityDemoService” +
11                        “/static/mobile/MobilityDemoService.json” );
12
13     var jsdo = new progress.data.JSDO({ name: ‘dsCustomer’ });
14     jsdo.subscribe(‘AfterFill’, onAfterFillCustomers, this);
15
16     jsdo.fill();
17
18     function onAfterFillCustomers(jsdo, success, request) {
19         var htmltext = document.getElementById(‘cust _ select’);
20         jsdo.eCustomer.foreach(function(customer){
21             htmltext += ‘<option>’ + customer.data.Name
22                     + new Array(40 - customer.data.Name.length).join(‘&nbsp;’)
23                     + customer.data.Phone + ‘</option>’;
24         });
25     document.getElementById(‘cust _ select’).innerHTML = htmltext;
26     rbf _ showInfoMessage(‘Customer records have been loaded.’);
27 }
28
29 </script>
```

Figure 3. Sample code to call JSDO code from a Script Component.

Use **Page Editor** in Rollbase to add Script Component to a page, and write JavaScript code that calls the JSDO as well as the Rollbase AJAX API. Figure 3 provides an example that can be included as a Script Component. This example calls the JSDO code from a Script Component:

LINES 03 – 04:

The example includes the `progress.session.js` and `progress.js` JavaScript files using the <script> tag. These JavaScript files can be hosted on the same web server as Rollbase or on a different web server.

LINES 07 – 14:

The example connects to an OpenEdge Mobile service using the Session Object, and calls `fill()` in a JSDO instance for `'dsCustomer'`.

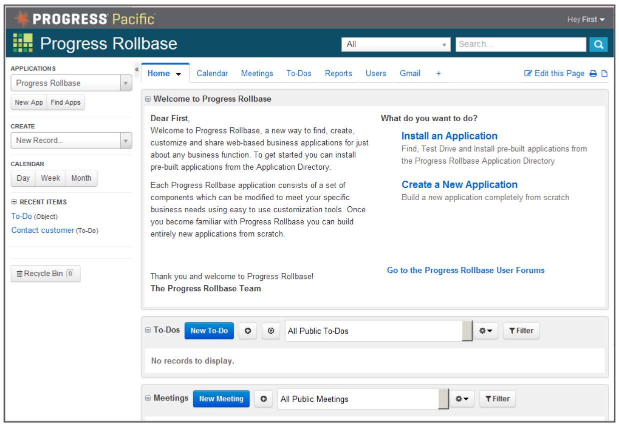
LINES 17 – 24:

The callback function for the `AfterFill` event populates an HTML select element and calls the Rollbase API `rbf _ showInfoMessage()` to indicate that records have been loaded.

STEPS TO CALL THE JSDO FROM A SCRIPT COMPONENT

Follow these steps to call the JSDO from a Script Component:

1. Select the desired Object from the Rollbase home page (Figure 4).



Dear First,
Welcome to Progress Rollbase, a new way to find, create, customize and share web-based business applications for just about any business function. To get started you can install pre-built applications from the Application Directory.

Each Progress Rollbase application consists of a set of components which can be modified to meet your specific business needs using easy to use customization tools. Once you become familiar with Progress Rollbase you can build entirely new applications from scratch.

Thank you and welcome to Progress Rollbase!
The Progress Rollbase Team

What do you want to do?

Install an Application
Find, Test Drive and Install pre-built applications from the Progress Rollbase Application Directory

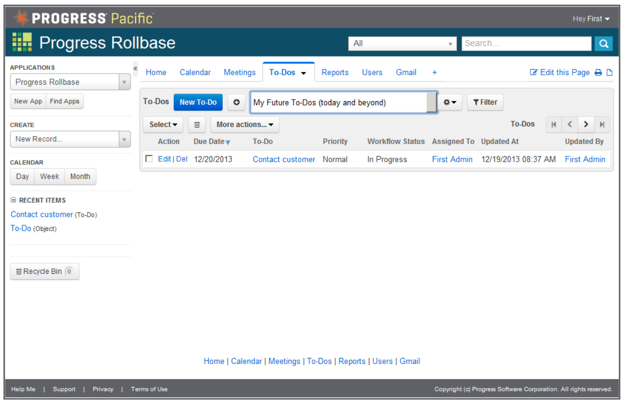
Create a New Application
Build a new application completely from scratch

Go to the Progress Rollbase User Forums

No records to display.

Figure 4. Rollbase home page.

2. Select **Edit this Page** from the Object page, from a **selected item** (as shown in Figure 5), or from the **Edit Page** of a selected item.
3. Drag **New <Script Component>** to the **Page Editor**.

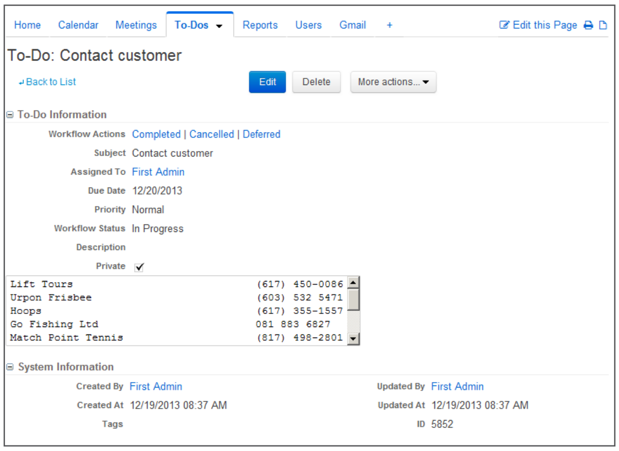


	Action	Due Date	To-Do	Priority	Workflow Status	Assigned To	Updated At	Updated By
☐	Edit | Del	12/20/2013	Contact customer	Normal	In Progress	First Admin	12/19/2013 08:37 AM	First Admin

Figure 5. To-Do Object selected from the Rollbase home page.

4. Choose **Edit** on the new Script Component and enter your JavaScript code. The JavaScript code to call the JSDO code would look similar to the one in Figure 3. Refer to Example 2 in Appendix A for a more extensive example.
5. Choose **Save** to save your changes. The Script Component executes after the page loads.

Note: Use the Web Inspector or debugging tools in your web browser to troubleshoot your work.



To-Do: Contact customer

To-Do Information

Workflow Actions	Completed | Cancelled | Deferred
Subject	Contact customer
Assigned To	First Admin
Due Date	12/20/2013
Priority	Normal
Workflow Status	In Progress
Description	
Private	✓

```
Lift Tours                         (617) 450-0086
Urpon Frisbee                      (603) 532 5471
Hoops                              (617) 355-1557
Go Fishing Ltd                     081 883 6827
Match Point Tennis                 (817) 498-2801
```

System Information

Created By	First Admin	Updated By	First Admin
Created At	12/19/2013 08:37 AM	Updated At	12/19/2013 08:37 AM
Tags		ID	5852

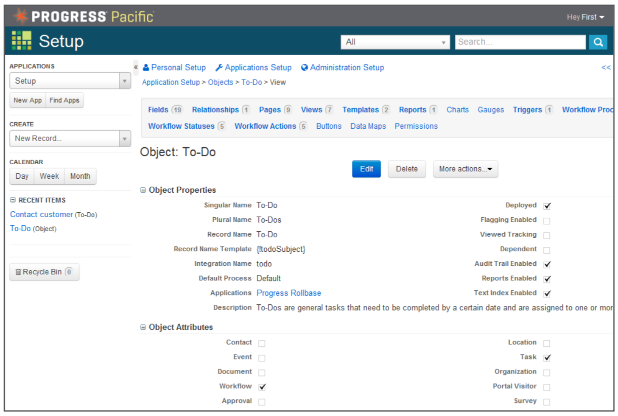
Figure 6. To-Do page showing records from the customer table in the OpenEdge database.

STEPS TO CALL THE JSDO FROM AN EVENT HANDLER

Follow these steps to call the JSDO from an event handler:

1. Add the JavaScript, using the JSDO as a Script Component, to the **Edit** page.
2. Select **Personal Setup | Application Setup | Administration Setup | Objects** to navigate to the **Definition Page** of the desired Object in Rollbase, as shown in Figure 7.

PROGRESS



Object: To-Do

Object Properties

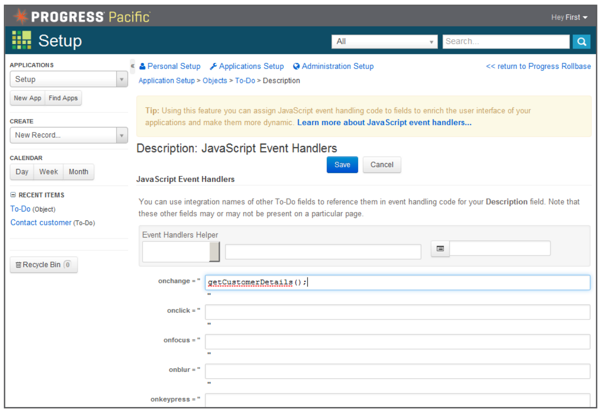
Singular Name	To-Do	Deployed	✓
Plural Name	To-Dos	Flagging Enabled	☐
Record Name	To-Do	Viewed Tracking	☐
Record Name Template	{!todoSubject}	Dependent	☐
Integration Name	todo	Audit Trail Enabled	✓
Default Process	Default	Reports Enabled	✓
Applications	Progress Rollbase	Text Index Enabled	✓
Description	To-Dos are general tasks that need to be completed by a certain date and are assigned to one or mor		

Object Attributes

Contact	☐	Location	☐
Event	☐	Task	✓
Document	☐	Organization	☐
Workflow	✓	Portal Visitor	☐
Approval	☐	Survey	☐

Figure 7. Definition page for the To-Do Object.

3. Select **Fields**.
4. Select **Events** on the desired field.
5. Enter your JavaScript code for the desired event handler, as shown in Figure 8.



Tip: Using this feature you can assign JavaScript event handling code to fields to enrich the user interface of your applications and make them more dynamic. Learn more about JavaScript event handlers...

Description: JavaScript Event Handlers

JavaScript Event Handlers

You can use integration names of other To-Do fields to reference them in event handling code for your **Description** field. Note that these other fields may or may not be present on a particular page.

```
onchange = " getCustomerDetails();
           "
onclick = "
           "
onfocus = "
           "
onblur = "
           "
onkeypress = "
```

Figure 8. Entering JavaScript code to selected Event Handlers.

BENEFITS

Using the Progress Rollbase platform, you can quickly create an application that calls Progress OpenEdge data and services. Your applications can then extend your customer's experience of working with your critical business applications outside of the office—boosting both your reach and your reputation without draining valuable resources.

PROGRESS SUPPORT SERVICES

Progress offers worldwide technical support and professional services around the clock and around the globe tailored to your business' individualized needs. Our self-service, global support centers, and tiered-level service agreements combine to protect the investments you have made in your technology. For more information, visit progress.com/support-and-services.

ABOUT THE AUTHOR

Edsel Garcia is a Principal Software Engineer in the Progress OpenEdge Engineering group. Edsel has a long history of first-hand experience using Progress Software products, starting as a customer and application developer about 23 years ago. During his 16-year tenure at Progress, Edsel has been a member of Customer Support, Solution Engineering development, the Tooling development team, the OpenEdge Architect product development team, the OpenEdge Management team, and the Core Client team as a member of the OpenEdge Mobile development initiative.

ABOUT PROGRESS ROLLBASE

Progress Rollbase is a cloud platform for development and delivery of software as a service (SaaS) business applications using point & click, drag & drop tools in a standard browser with a minimal amount of code. Progress Rollbase delivers on the promise of rapid application development (RAD), making application creation much faster than traditional software development methods. Progress Rollbase is offered as both a hosted service (Progress Rollbase Hosted Cloud) and as an installable product (Progress Rollbase Private Cloud) that can be deployed on any cloud infrastructure or on-premises. Progress Rollbase is designed for enterprises as well as independent software vendors (ISVs) with a complete system for tenant and subscriber management, provisioning, application development, publishing and deployment. Progress Rollbase provides ISVs and resellers with complete white label capabilities making it easy to deploy applications under any brand or identity. For more information about the Progress Rollbase platform, please visit: www.progress.com/rollbase.

APPENDIX A: JSDO EXAMPLES

EXAMPLE 1: USING THE JSDO FROM AN HTML PAGE JAVASCRIPT

This example demonstrates `progress.js` and `progress.session.js` from the oemobiledemo.progress.com and uses a Mobile service called MobilityDemoService hosted on the same machine.

```
<!DOCTYPE html>
<html>
<head>
    <script src="http://oemobiledemo.progress.com/jsdo/progress.js">
    </script>
    <script src="http://oemobiledemo.progress.com/jsdo/progress.session.js">
    </script>
<script>
    session = new progress.data.Session();
    session.login("http://oemobiledemo.progress.com/MobilityDemoService", "", "");
    session.addCatalog("http://oemobiledemo.progress.com/MobilityDemoService" +
                       "/static/mobile/MobilityDemoService.json");
    jsdo = new progress.data.JSDO({ name: 'dsCustomer' });
    jsdo.subscribe('AfterFill', onAfterFillCustomers, this);
    jsdo.fill();
    function onAfterFillCustomers(jsdo, success, request) {
        jsdo.eCustomer.foreach(function(customer) {
        document.write(
            customer.data.CustNum + ' ' + customer.data.Name + '<br>');
        });
    }
</script>
</head>
<body>
</body>
</html>
```

EXAMPLE 2: USING THE JSDO FROM A SCRIPT COMPONENT

This example demonstrates progress.js and progress.session.js from the /jsdo subdirectory of an Apache Tomcat server running Rollbase Private Cloud. It uses a Mobile Service called MobilityDemoService hosted on oemobiledemo.progress.com.

```
<select id="cust_select" size="5" style="font-family: monospace">
</select>
<script src="http://oemobiledemo.progress.com/jsdo/progress.js">
</script>
<script src="http://oemobiledemo.progress.com/jsdo/progress.session.js">
</script>
<script>
    var session = new progress.data.Session();
    session.login("http://oemobiledemo.progress.com/MobilityDemoService", "", "");
    session.addCatalog("http://oemobiledemo.progress.com/MobilityDemoService" +
                      "/static/mobile/MobilityDemoService.json" );

    var jsdo = new progress.data.JSDO({ name: 'dsCustomer' });
    jsdo.subscribe('AfterFill', onAfterFillCustomers, this);


    jsdo.fill();

    function onAfterFillCustomers(jsdo, success, request) {
        var htmltext = document.getElementById('cust_select');
        jsdo.eCustomer.foreach(function(customer){
            htmltext += '<option>' + customer.data.Name
                     + new Array(40 - customer.data.Name.length).join('&nbsp;')
                     + customer.data.Phone + '</option>';
        });
    document.getElementById('cust_select').innerHTML = htmltext;
    rbf_showInfoMessage('Customer records have been loaded.');
}


</script>
```

APPENDIX B: RELATED DOCUMENTS

REFER TO THE FOLLOWING DOCUMENTS FOR ADDITIONAL INFORMATION ON PROGRESS ROLLBASE AND JSDO CODE:

- Rollbase in Action: http://www.rollbase.com/download/Rollbase_in_Action.pdf
- Chapter 7: Client-Side Code: http://www.rollbase.com/master/docs/chapter7.pdf
- OpenEdge Mobile Applications: http://documentation.progress.com/output/OpenEdge112/pdfs/dvmad/dvmad.pdf