PUG CHALLENGE EXCHANGE AMERICAS

270: Working with OpenEdge Data and Business Logic in a Kendo UI Builder Application

June 6th 2017

Anil Kumar Kotha

Edsel Garcia

Disclaimer

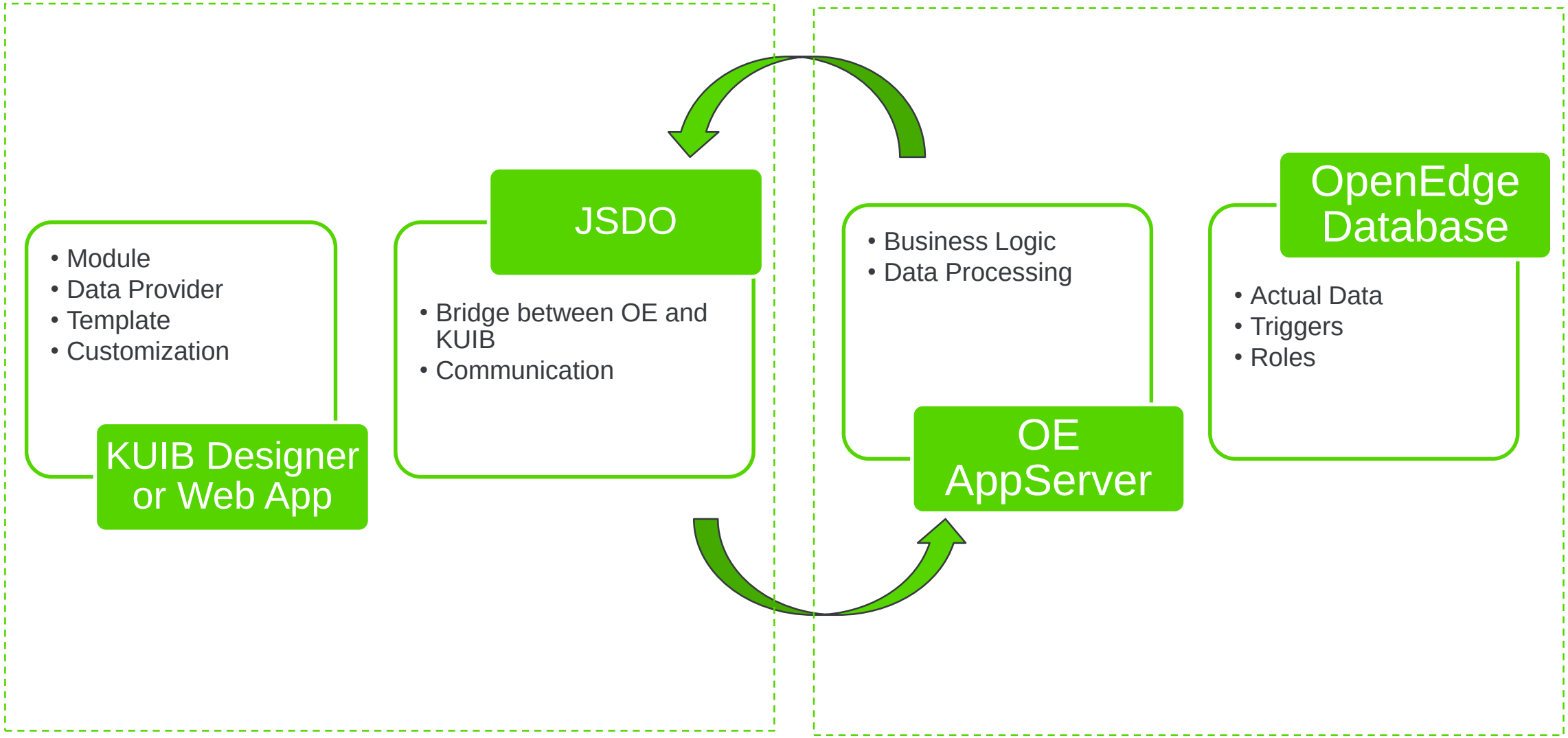
Agenda

- Introduction
- JSDO – JavaScript Data Object
- Business Entity and Extensions
- Working with relational and referential data
- Customizing Kendo UI Builder Web Apps
- OpenEdge Security and Kendo UI Builder

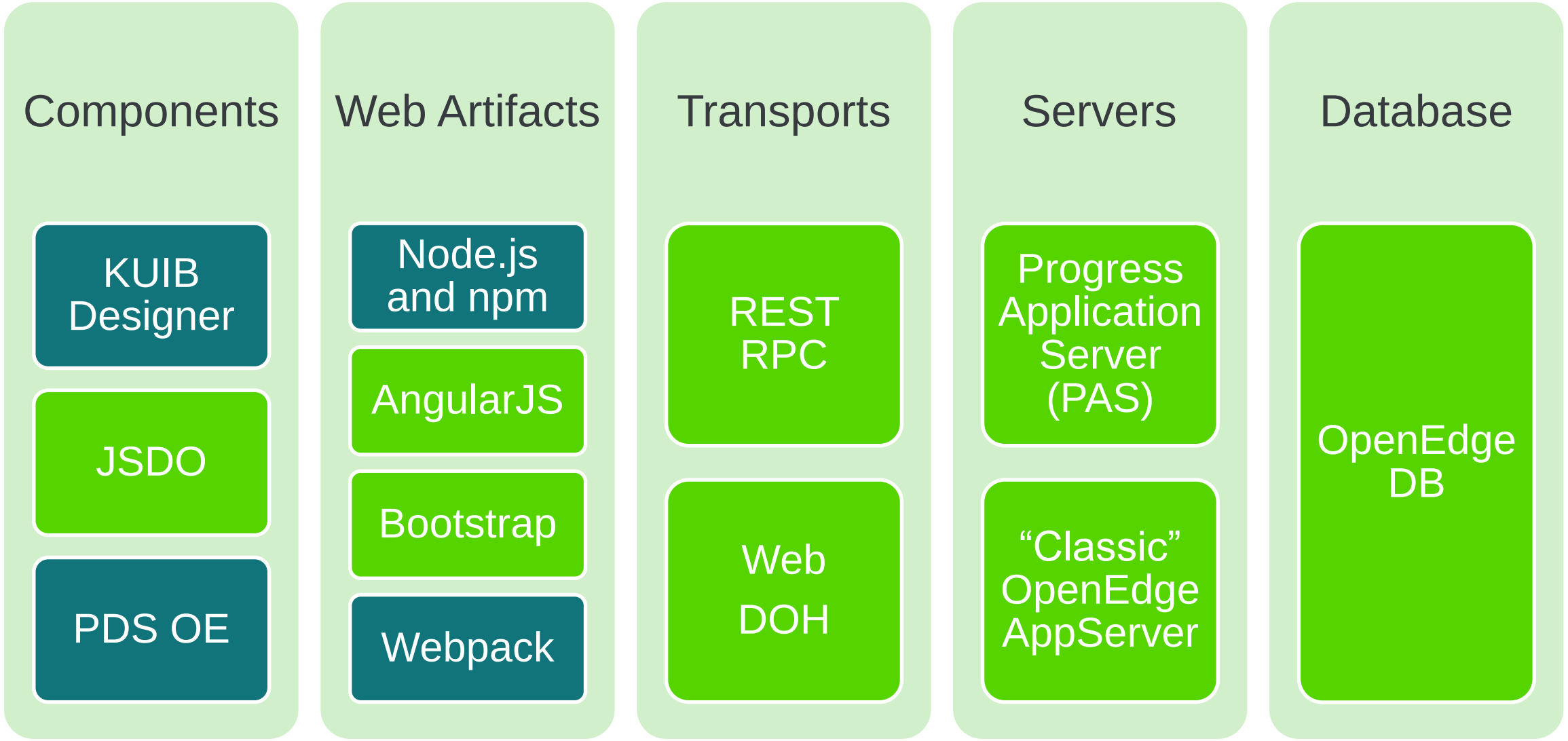
Introduction

- Single Page Application (SPA) Dev Environment
- Drag and Drop facility
- Predefined templates
- Rich set of controls
- Electron Shell container

KUIB App flow



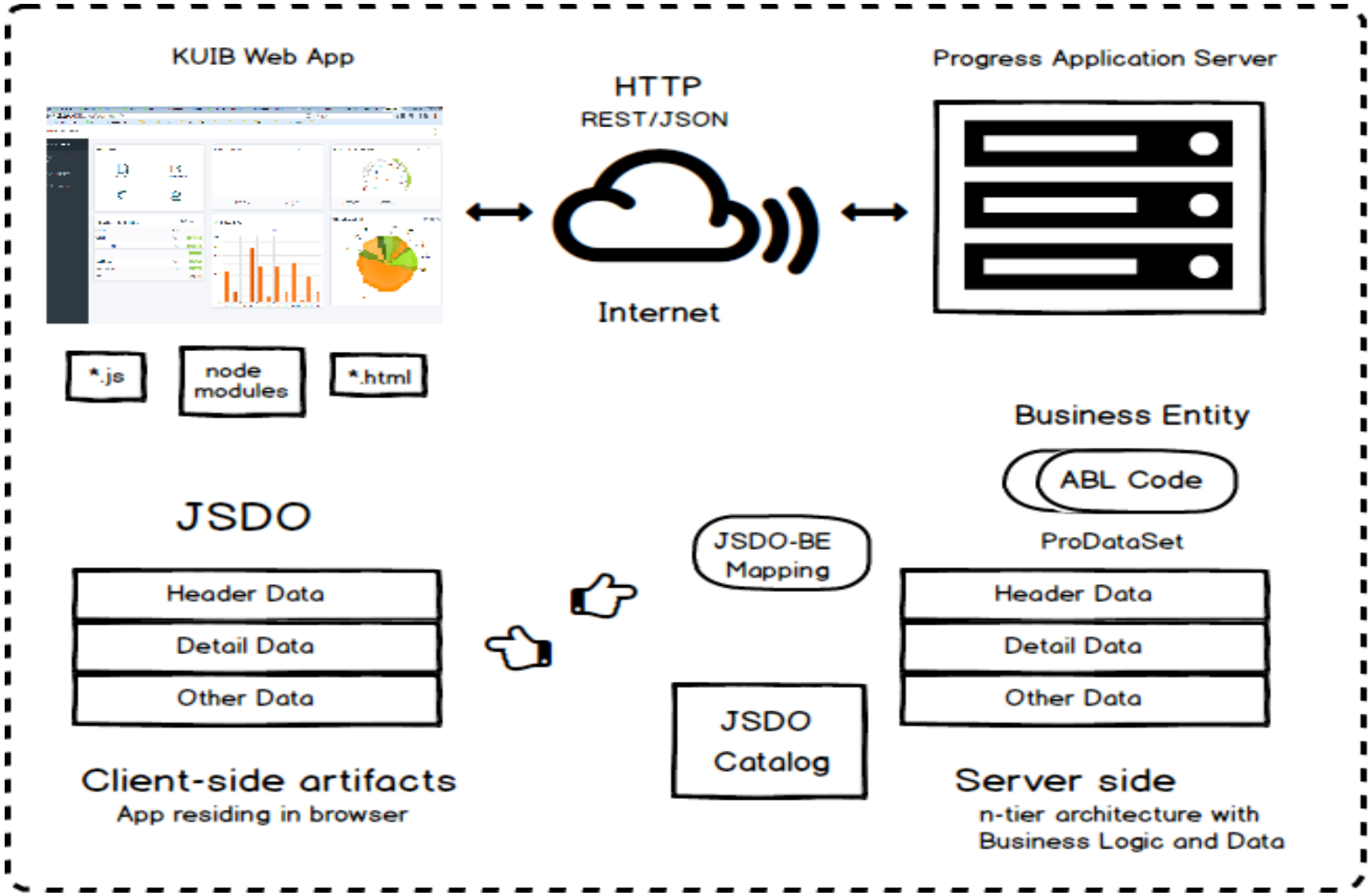
Different Components Involved – Runtime and Design



Agenda

- Introduction
- JSDO – JavaScript Data Object
- Business Entity and Extensions
- Working with relational and referential data
- Customizing Kendo UI Builder Web Apps
- OpenEdge Security and Kendo UI Builder

JSDO's role in KUIB Web App



JSDO APIs

JSDO Method	Business Entity
add() – Create assign() – Update remove() – Delete	
fill() – Read	READ
saveChanges(false)	CUD
saveChanges(true)	Submit
invoke(“myMethod”)	myMethod()

Error handling via JSDO

- Use getErrors() API
  - Allows us to access all AppServer errors seamlessly

```
jsdo = new progress.data.JSDO({ name: 'CustOrder' });

...

jsdoErrors = jsdo.eCustomer.getErrors();
```

- Error handler can be overridden in KUIB

Agenda

- Introduction
- JSDO – JavaScript Data Object
- Business Entity and Extensions
- Working with relational and referential data
- Customizing Kendo UI Builder Web Apps
- OpenEdge Security and Kendo UI Builder

Modernization
Working with Business Entities

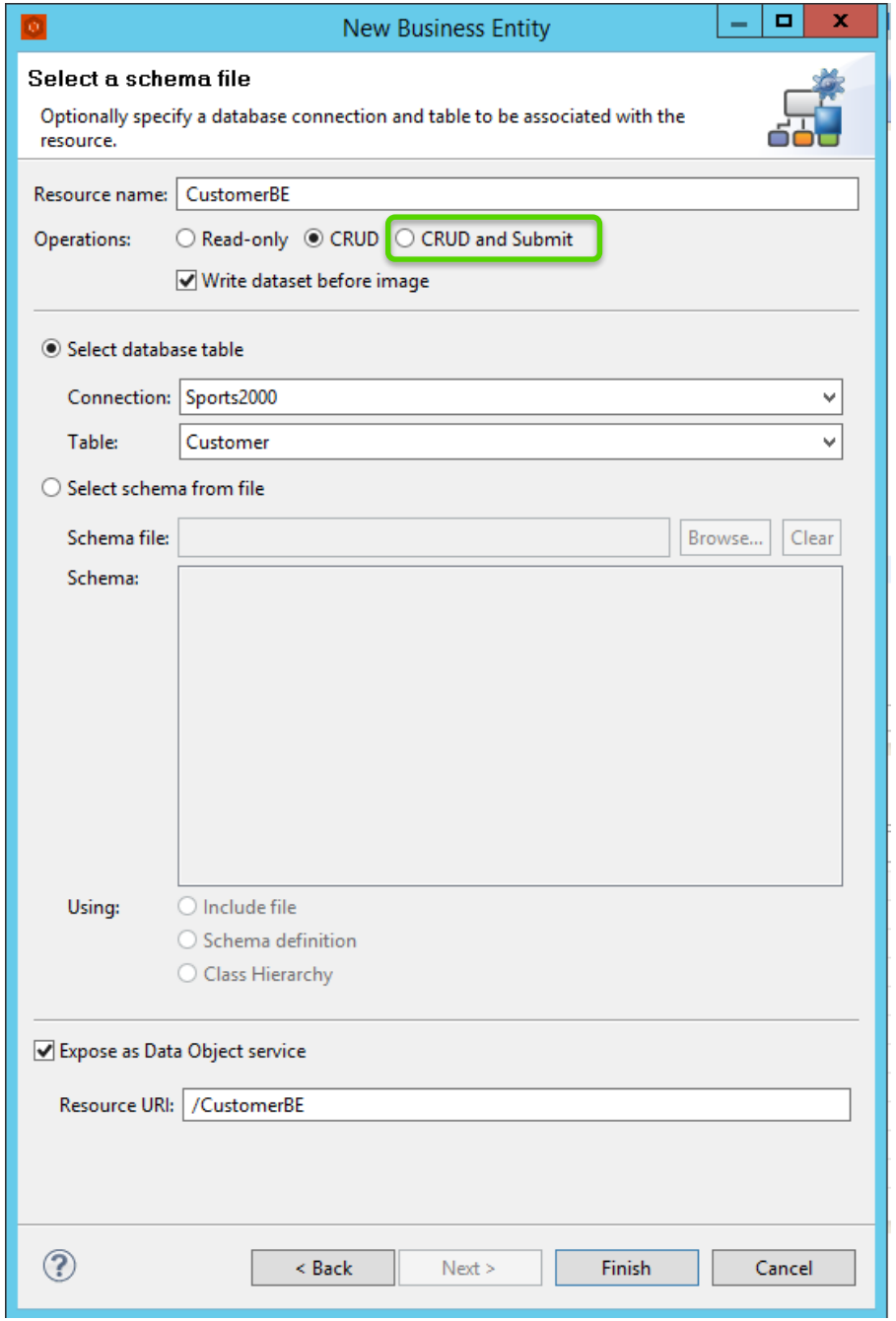
- Start from scratch

- Leverage existing code

- Convert existing code

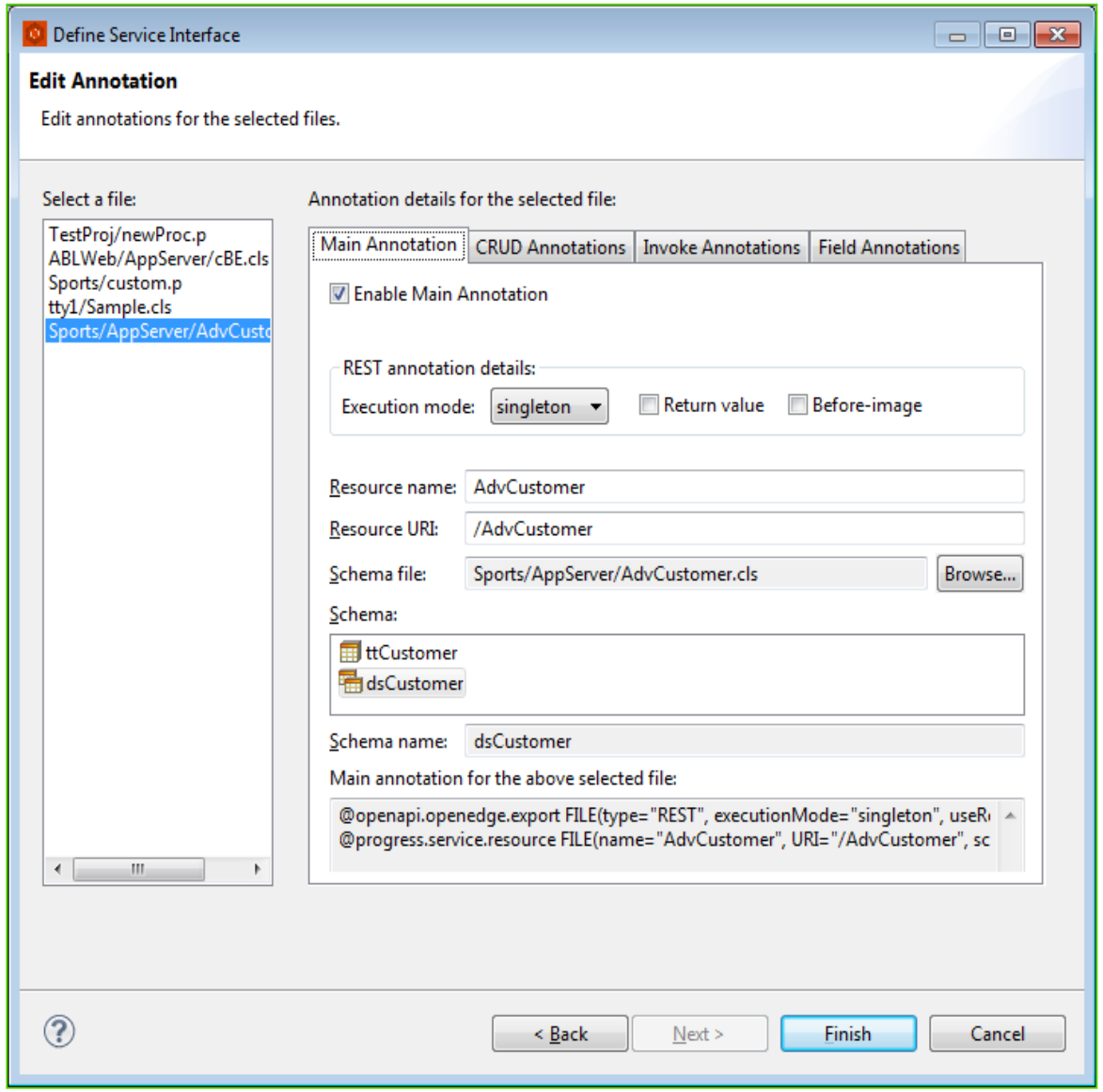
Business Entities

- CRUD Operations
- Significance of Submit
  - Transactional operation
- Abstract Business Entity
  - OpenEdge.BusinessLogic.BusinessEntity
  - Located in *DLC/tty/OpenEdge.BusinessLogic.pl*



Define Service Interface wizard

- Annotations
  - File level
  - Method level (CRUD + INVOKE)
  - Field level
    - Semantic types
    - Foreign Key

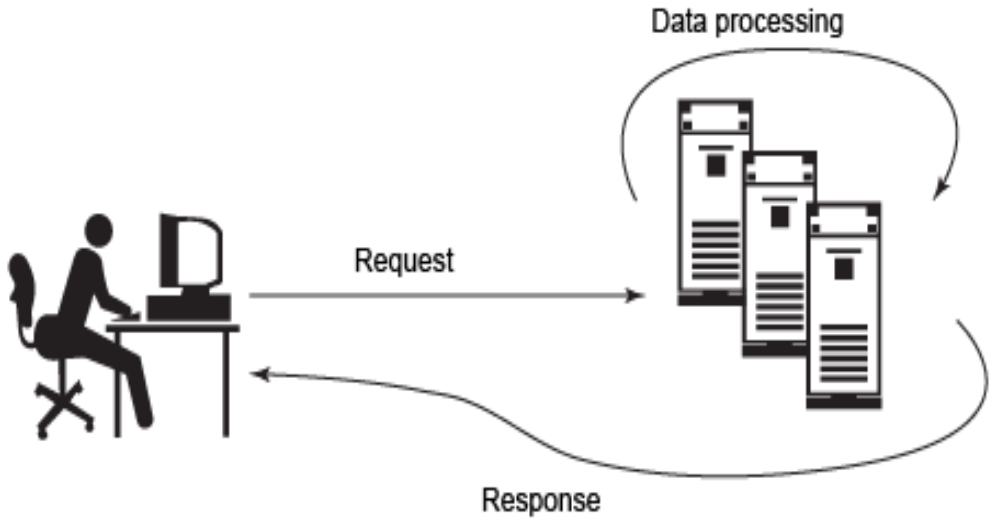


Extending a Business Entity

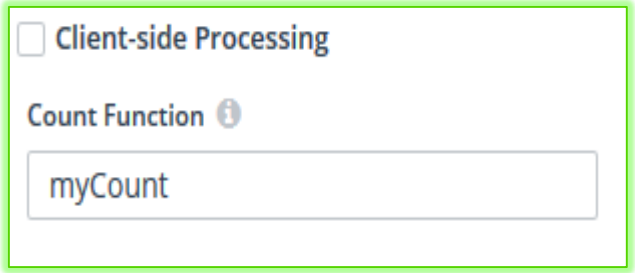
- Change temp-table / dataset definition
- Customizing auto-generated CRUD+S operations code
  - Abstract Business Entity is optional
- Server side processing
- Annotations:
  - Mapping Types
  - Semantic Types
  - Foreign Key
  - Count

Server side processing

- Business Entity should be configured with JFP
  - ‘Count’ operation is optional in KUIB
  - Auto-filled if BE already has a count method
- Every request is processed in AppServer layer
- Filtering, Paging, Sorting are dependent



- Simple config in KUIB’s data source



JFP – JSON Filter Pattern

- Allows data processing at server side
- Additional annotations to ‘Read’ method
- Default Kendo UI DataSource processing is at client side
- Accessed via JSDO’s mapping type - JFP

```
@openapi.openedge.export(type="REST", useReturnValue="false", writeDataSetBeforeImage="true").
@progress.service.resourceMapping(type="REST", operation="read", URI="?filter=~{filter~}",
    alias="", mediaType="application/json").
@openapi.openedge.method.property (name="mappingType", value="JFP").
@openapi.openedge.method.property (name="capabilities", value="ablFilter,top,skip,id,orderBy").
METHOD PUBLIC VOID ReadCustomer(INPUT filter AS CHARACTER, OUTPUT DATASET dsCustomer):
```

Count Operation

- Count operation [Required for Server side processing]
  - Fetch number of records in OE database
  - Additional operation similar to INVOKE
- Count operation annotation
  - JSDO is aware of ‘count’ by default

```
@openapi.openedge.export(type="REST", useReturnValue="false",
    writeDataSetBeforeImage="false").
@progress.service.resourceMapping(type="REST", operation="count",
    URI="/myCount?filter=~{filter~}", alias="", mediaType="application/json").
METHOD PUBLIC VOID myCount( INPUT filter AS CHARACTER, OUTPUT numRecs AS
    INTEGER):
```

```
"operations": [
    {
        "name": "myCount",
        "path": "\/myCount?filter={filter}",
        "useBeforeImage": false,
        "type": "count",
        "verb": "put",
        "params": []
    },
```

Mappings

- Request mapping

```
function registerPlugin() {
    var jfpPlugin = progress.data.PluginManager.getPlugin("JFP");
    progress.data.PluginManager.addPlugin("MYJFP", {
        requestMapping: function (jsdo, params, info) {
            var requestParams = {},

                object = {};
            params = jfpPlugin.requestMapping(jsdo, params, info);
            if (params && typeof params.filter === "string") {
                object = JSON.parse(params.filter);
            }
            object.mydata = jsdo.getProperty("mydata");
            requestParams.filter = JSON.stringify(object);
            return requestParams;
        }
    });
}
```

- Response mapping

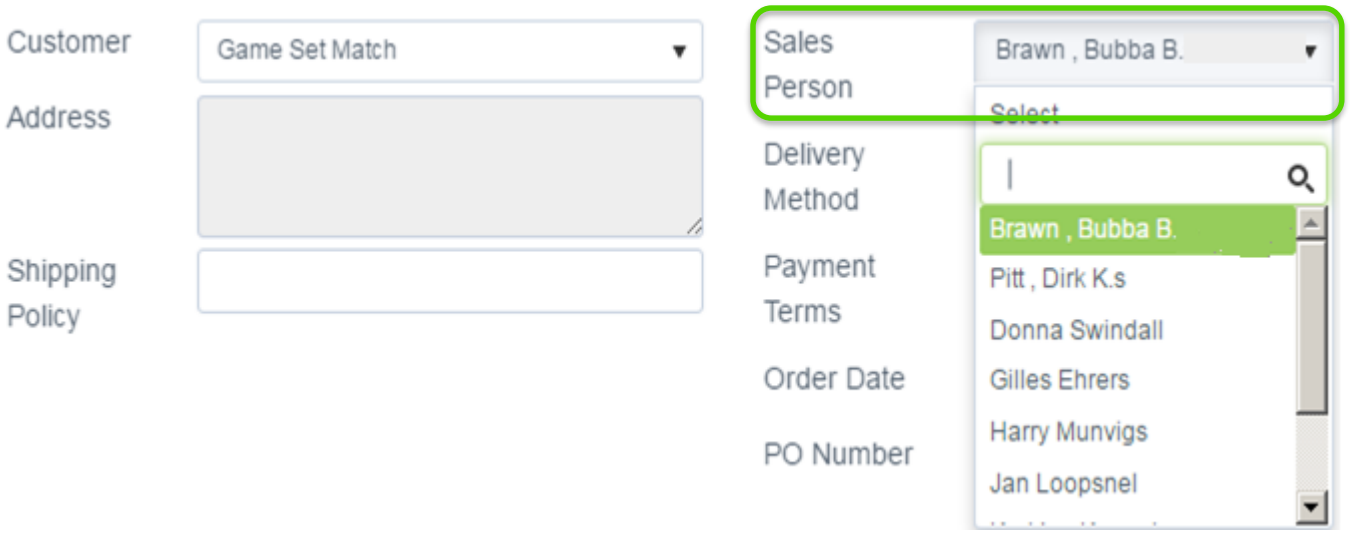
```
progress.data.PluginManager.addPlugin("myResponsePlugin", {

    responseMapping: function(jsdo, response, info) {

        var record;
        var newData = response.dsEmployee.ttEmployee;

        if (info.operation === "read") {

        for (var i = 0; i < newData.length; i++) {

            record = newData[i];
            record.VacDays = record.VacDays + 10;

        }

        jsdo.setProperty("server.count", response.myTotal);
        }

        // You must return the response
        return response;

    }

}) ;
```

DEMO

Agenda

- Introduction
- Business Entity and Extensions
- JSDO – JavaScript Data Object
- Working with relational and referential data
- Customizing Kendo UI Builder Web Apps
- OpenEdge Security and Kendo UI Builder

Foreign Key Support



- Placeholder field
- Semantic Type is ‘Lookup’
- Editor Types
  - Combo-box
  - Drop-down list
- Business Logic should be annotated as below:
  - 11.7.1 PDS OE supports tooling

```
@openapi.openedge.entity.foreignkey (name="ttSalesrepFK", fields="SalesRepID",
    parent="Salesrep.ttSalesrep", parentFields="SalesRep").
DEFINE TEMP-TABLE ttOrder NO-UNDO BEFORE-TABLE bttOrder
    FIELD Ordernum   AS INTEGER   INITIAL 0 LABEL "Order Num"
    FIELD SalesrepID AS CHARACTER LABEL "Sales Rep"
```

Hierarchical and Stacked Data Grids

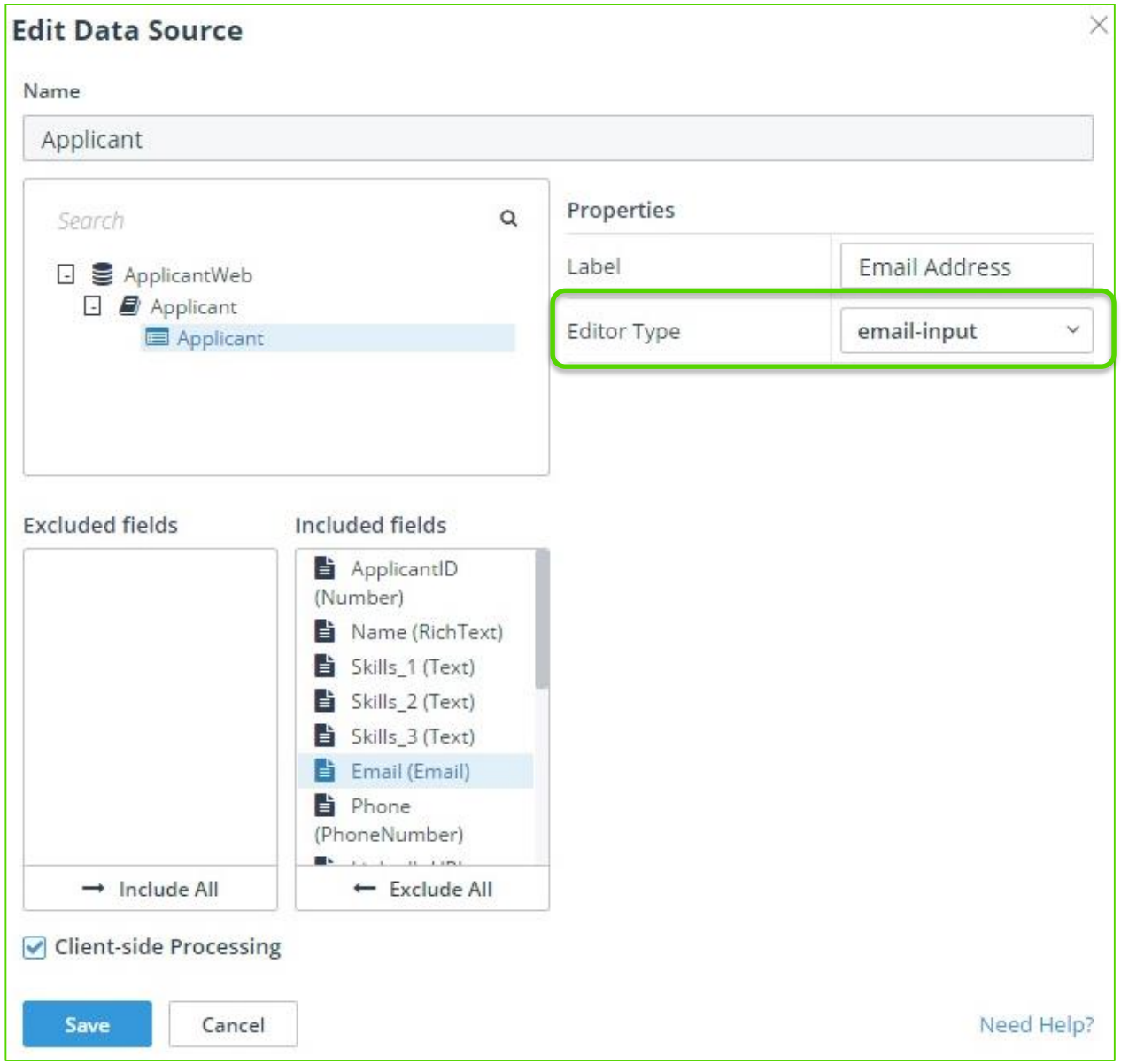
- Parent/Child data represented in different forms in KUIB webapp
- Supports Inline, Popup, Incell editing modes
- Allows CRUD operations on child table data
- Single relation or multi-relation(s) among tables
- Both parent and child tables should be in single resource

Agenda

- Introduction
- JSDO – JavaScript Data Object
- Business Entity and Extensions
- Working with relational and referential data
- Customizing Kendo UI Builder Web Apps
- OpenEdge Security and Kendo UI Builder

KUIB and JSDO code under the hood

- JSDO Catalog
  - Resources (DataSets and Temp-tables)
  - Operations
- Data Source definitions
- Arrays representation
- Metadata
- Generated code uses:
  - JSDO Dialect for Kendo UI DataSource



Customizing KUIB Code

- Custom Sections
- View Factory
- Public Controller
- Other Assets
- Custom Templates

Recommendations

- Encapsulate functionality into high level API methods
  - AngularJS code
  - Kendo UI components
  - Kendo UI DataSource
  - dsService

DEMO

Agenda

- Introduction
- Business Entity and Extensions
- JSDO – JavaScript Data Object
- Working with relational and referential data
- Customizing Kendo UI Builder Web Apps
- OpenEdge Security and Kendo UI Builder

Authentication

- Supported models
  - Anonymous
  - Basic
  - FORM
  - SSO (in pipeline)

- JSDO Specific:
  - Use **progress.data.JSDOSession**
  - progress.data.Session (plans to deprecate in future)

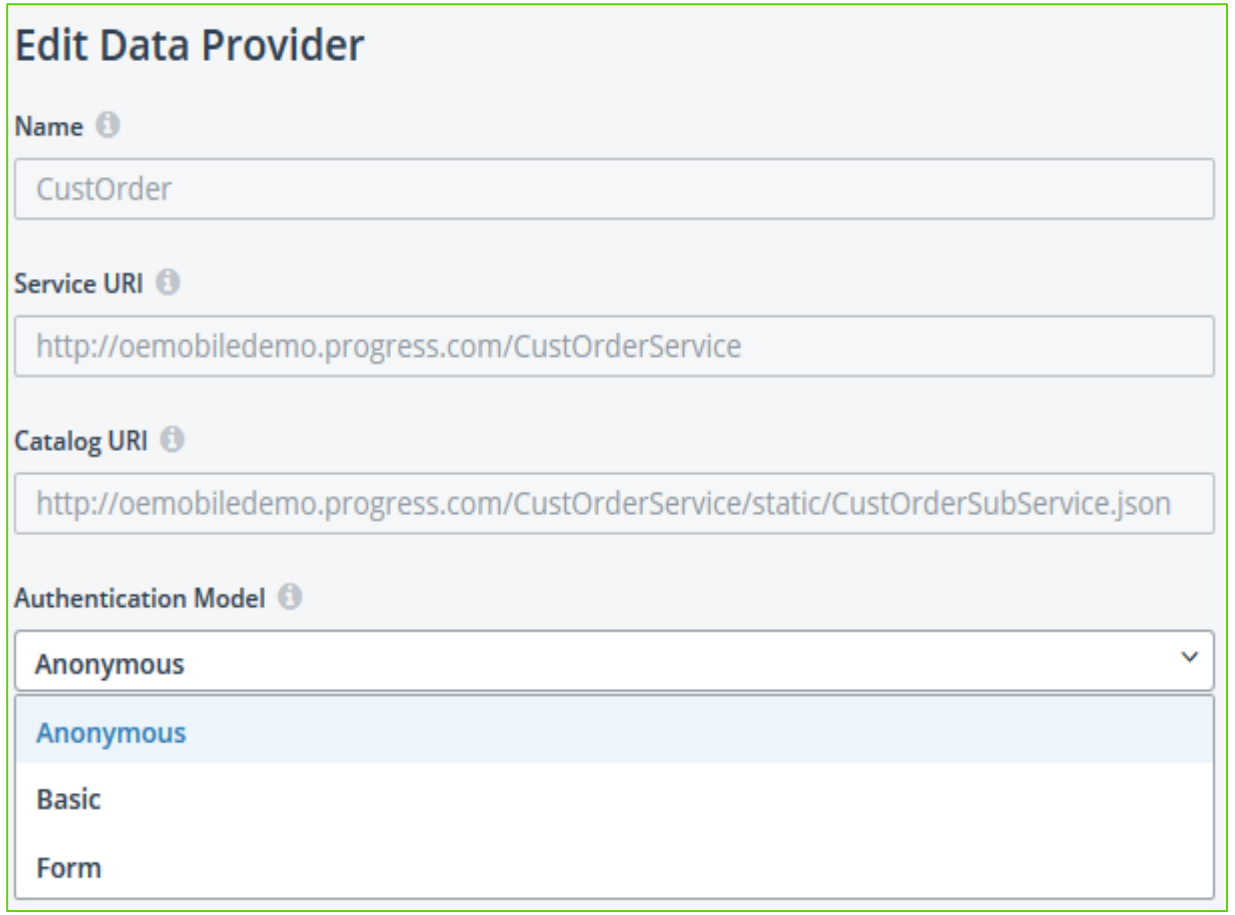
Enabling Authentication at PASOE layer

- Modifications to *oeablSecurity.properties* file **[new in OpenEdge 11.7]**
  - Located in {DLCWork}/<oepas_instance>\webapps\<webapp>\WEB-INF
- Change *client.login.model*
- User's information
  - users.properties
  - LDAP
  - OERealm

```
################## The HTTP client Authentication model to use ###############
## This property controls which HTTP client authentication model to use.  The
## allowed names are:
##
##      name                  Description
##    ========================================================================
##     anonymous              No user login - all clients have public access
##     basic                  Users authenticate using the HTTP BASIC standard
##     form                   Users authenticate using a HTTP POST message &
##                            form data
##     container              Users authenticate via Tomcat realm services and
##                            authorize URL access via Spring Security
##     sso                    OpenEdge Single Sign-on using ClientPrincipal
##                            access tokens
##

client.login.model=form

################## HTTP BASIC Realm name for All Transports #################
## Set the BASIC realm name used by browsers to prompt the user for a
## user-id/password.
##
http.all.realm=OpenEdge
```

Authentication – KUIB

- At Data Provider (resource) level
  - All Data sources (tables) will use same authentication
- Login screen is shown upon Preview
  - First module's resource loaded upon successful login



Summary

- Custom Business Entities to leverage business logic
- Flexibility with custom views in KUIB
  - Pre-defined views
  - User-defined (Blank) view
- Access to large set of Kendo UI components via KUIB
- Use API’s to improve maintainability of code

Happy Developing !!!

KUIB is the ‘Key’ to Modernization

PUGCHALLENGE EXCHANGE
AMERICAS