# Table of Contents

1. Introduction .......................................................................................................................... 1  
   1.1. Main systems .................................................................................................................. 1  
   1.2. Components ................................................................................................................... 1  
   1.3. Example workflow ......................................................................................................... 2  
      1.3.1. User enters URL ..................................................................................................... 3  
      1.3.2. Control servlet takes over ..................................................................................... 3  
      1.3.3. Widget rendered ..................................................................................................... 4  
2. Web Framework ..................................................................................................................... 6  
3. Web Applications .................................................................................................................. 7  
   3.1. Cross-domains Single Sign On (SSO) ........................................................................... 7  
   3.2. Control Servlet .............................................................................................................. 8  
      3.2.1. Requests ................................................................................................................ 8  
      3.2.2. Views ..................................................................................................................... 8  
4. Entity Engine ....................................................................................................................... 9  
   4.1. Entities ........................................................................................................................... 9  
      4.1.1. Standard Entities ................................................................................................... 9  
      4.1.2. View Entities ......................................................................................................... 9  
      4.1.3. Extended Entities ................................................................................................. 9  
      4.1.4. Dynamic View Entities ....................................................................................... 9  
   4.2. XML Data ....................................................................................................................... 9  
   4.3. Entity engine configuration ............................................................................................ 9  
   4.4. Supported databases ...................................................................................................... 9  
   4.5. Data Model Changes ..................................................................................................... 9  
      4.5.1. Changes with OFBiz Trunk (Upcoming Branch) .................................................... 9  
      4.5.2. Changes with OFBiz 17 ....................................................................................... 10  
      4.5.3. Changes between OFBiz 9 to OFBiz 16 ............................................................... 11  
5. Service Engine ..................................................................................................................... 24  
   5.1. Declaration and Implementation ..................................................................................... 24  
   5.2. Supported languages .................................................................................................... 24  
   5.3. Transaction management .............................................................................................. 24  
   5.4. Web services .................................................................................................................. 24  
6. Widget System ....................................................................................................................... 25  
   6.1. Screen Widget ............................................................................................................... 25  
      6.1.1. Decoration Pattern ............................................................................................... 25  
   6.2. Form Widget ................................................................................................................. 25  
   6.3. Menu Widget ................................................................................................................. 25  
   6.4. Tree Widget ................................................................................................................... 25  
   6.5. Portal Widget ................................................................................................................. 25
1. Introduction

Welcome to the Apache OFBiz developer manual. This manual provides information to help with customizing and developing OFBiz. If you are new to OFBiz and interested in learning how to use it, you may want to start with the "Apache OFBiz User Manual".

OFBiz is a large system composed of multiple subsystems. This manual attempts to introduce the overall architecture and high level concepts, followed by a detailed description of each subsystem. In addition, the manual will cover topics necessary for developers including the development environment, APIs, deployment, security, and so on.

1.1. Main systems

OFBiz at its core is a collection of systems:

- A web server (Apache Tomcat)
- A web MVC framework for routing and handling requests.
- An entity engine to define, load and manipulate data.
- A service engine to define and control business logic.
- A widget system to draw and interact with a user interface.

On top of the above mentioned core systems, OFBiz provides:

- A data model shared across most businesses defining things like orders, invoices, general ledgers, customers and so on.
- A library of services that operate on the above mentioned data model such as "createBillingAccount" or "updateInvoice" and so on.
- A collection of applications that provide a user interface to allow users to interact with the system. These applications usually operate on the existing data model and service library. Examples include the "Accounting Manager" and "Order Manager".
- A collection of optional applications called "plugins" that extend basic functionality and is the main way to add custom logic to OFBiz.

1.2. Components

The basic unit in OFBiz is called "component". A component is at a minimum a folder with a file inside of it called "ofbiz-component.xml"

Every application in OFBiz is a component. For example, the order manager is a component, the accounting manager is also a component, and so on.

By convention, OFBiz components have the following main directory structure:
It is apparent from the above directory structure that each OFBiz component is in fact a full application as it contains entities, data, services, user interface, routing, tests, and business logic.

Both core OFBiz applications as well as plugins are nothing more than components. The only difference is that core applications reside in the "applications" folder whereas plugins reside in the "plugins" folder; also OFBiz does not ship with plugins by default.

1.3. Example workflow

Many basic concepts were explained so far. An example would help in putting all of these concepts together to understand the bigger picture. Let us take an example where a user opens a web browser and enters a certain URL and hits the enter key. What happens? It turns out answering this question is not quite simple because lots of things occur the moment the user hits "enter".

To try to explain what happens, take a look at the below diagram. Do not worry if it is not fully understandable, we will go through most of it in our example.
1.3.1. User enters URL

In the first step in our example, the user enters the following URL:

https://localhost:8443/accounting/control/findInvoices

If we break down this URL, we identify the following parts:

- localhost: Name of the server in which OFBiz is running
- 8443: Default https port for OFBiz
- accounting: web application name. A web application is something which is defined inside a component
- control: Tells OFBiz to transfer routing to the control servlet
- findInvoices: request name inside the control servlet

1.3.2. Control servlet takes over

The Java Servlet Container (tomcat) re-routes incoming requests through web.xml to a special OFBiz servlet called the control servlet. The control servlet for each OFBiz component is defined in controller.xml under the webapp folder.

The main configuration for routing happens in controller.xml. The purpose of this file is to map requests to responses.
Request Map

A request in the control servlet might contain the following information:

- Define communication protocol (http or https) as well as whether authentication is required.
- Fire up an event which could be either a piece of code (like a script) or a service.
- Define a response to the request. A response could either be another request or a view map.

So in this example, the findInvoices request is mapped to a findInvoices view.

View Map

A view map maps a view name to a certain view-type and a certain location.

View types can be one of:

- screen: A screen widget which translates to normal HTML.
- screenfop: A PDF screen designed with Apache FOP based constructs.
- screencsv: A comma separated value output report.
- simple-content; A special MIME content type (like binary files).
- ftl: An HTML document generated directly from a FreeMarker template.
- screenxls: An Excel spreadsheet.

In the findInvoices example, the view-map type is a normal screen which is mapped to the screen: component://accounting/widget/InvoiceScreens.xml#FindInvoices

1.3.3. Widget rendered

Once the screen location is identified and retrieved from the previous step, the OFBiz widget system starts to translate the XML definition of the screen to actual HTML output.

A screen is a collection of many different things and can include:

- Other screens
- Decorator screens
- Conditional logic for hiding / showing parts of the screen
- Data preparation directives in the <action> tag
- Forms
- Menus
- Trees
- Platform specific code (like FreeMarker for HTML output)
- Others (portals, images labels etc ...)
Continuing the example, the FindInvoices screen contains many details including two forms. One form is for entering invoice search fields and the other form displays search results.
2. Web Framework
3. Web Applications

The OFBiz webapp is one of the core framework components. It is tightly integrated with other framework components.

3.1. Cross-domains Single Sign On (SSO)

In some cases you need to split the OFBiz applications on different servers, and possibly in production on different domains. This can happen for different reasons, most often for performance reason.

As it’s annoying to give each time a credential when changing from an OFBiz application to another on the same server, the same applies when changing from an OFBiz application to another on another domain.

To prevent that on the same server, the ExternalLoginKey mechanism is used. The cross-domains SSO feature allows to navigate from a domain to another with automated SSO.

It based on 3 technologies:

**JWT**
- JWT Official site - Wikipedia for JWT

**CORS**
- CORS (Mozilla doc) - Wikipedia for CORS

**Ajax**
- Ajax, now well known I guess, in OFBiz we use jQuery for that.

The mechanism is simple.

*On the source side:*

1. When an user log in in an application (webApp) a webappName.securedLoginId cookie is created. This cookie will be used by the mechanism to know the current logged in user. *Note that all webappName.securedLoginId cookies are deleted when the user session is closed or time out. Hence (apart also using an intrinsically secured cookie) the mechanism is secured, even on shared machines. Of course if people are sharing a machine during their sessions, things could get complicated. This unlikely later case is not taken in account.*

2. The user is given a JavaScript link which passes the URL to reach and the calling webapp name to the sendJWT() Ajax function.

3. The sendJWT() Ajax function calls the loadJWT() Ajax function which in turn calls the CommonEvents::loadJWT method through the common controller.

4. The CommonEvents::loadJWT method uses the calling webapp name to retrieve the userLoginId from the secured webappName.securedLoginId cookie, creates a JWT containing the userLoginId, and returns it to the loadJWT() Ajax function.

5. Then the sendJWT() Ajax function sends an Authorization header containing the JWT to the URL
to reach. At this stage, if all things are correct, the flow leaves the source side.

**On the server side:**

1. A CORS policy is needed. *Without it, the Authorization token containing the JWT will be rejected.*
   - It’s a simple policy but you need to strictly define the authorized domains. Never use the lazy "*" for domains (ie all domains), else the preflight request will not work. Here is an example for Apache HTTPD (domain value is "https://localhost:8443" for official OFBiz demo):

   ```
   Header set Access-Control-Allow-Origin domain
   Header set Access-Control-Allow-Headers "Authorization"
   Header set Access-Control-Allow-Credentials "true"
   ```

1. The checkJWTLogin preprocessor, similar to the checkExternalLoginKey, intercepts the JWT, checks it and if all is OK signs the user on. That’s it!

In the example component, the FormWidgetExamples screen contains 2 new fields in the LinksExampleForm which demonstrate the use from a local instance to the trunk demo instance.

If you are interested in more details you may refer to [https://issues.apache.org/jira/browse/OFBIZ-10307](https://issues.apache.org/jira/browse/OFBIZ-10307)

### 3.2. Control Servlet

#### 3.2.1. Requests

#### 3.2.2. Views
4. Entity Engine

4.1. Entities

4.1.1. Standard Entities

4.1.2. View Entities

4.1.3. Extended Entities

4.1.4. Dynamic View Entities

4.2. XML Data

4.3. Entity engine configuration

4.4. Supported databases

4.5. Data Model Changes

The Apache OFBiz® Project Release 17.12

Apache OFBiz follows The Universal Data Model by Len Silverston, with a grain of salt.

The following file contains information about the data model changes in the Apache OFBiz. The detailed description of migration scripts specified here can be found at Revisions Requiring Data Migration - upgrade ofbiz page.

4.5.1. Changes with OFBiz Trunk (Upcoming Branch)

Entity Changes

Added 1 new entity

1. ProdPromoCodeContactMech

Removed/Deprecate 1 entity

1. ProductPromoCodeEmail

Field Changes

<table>
<thead>
<tr>
<th>Entity</th>
<th>Field</th>
<th>Action</th>
<th>IsPK</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>GlXbrlClass</td>
<td>parentGlXbrlClassId</td>
<td>Added</td>
<td>No</td>
<td>36a123a</td>
</tr>
</tbody>
</table>
If in the list above fields are mentioned where 'IsPK' = yes, we advice to follow steps below for a successful upgrade:

1. Stop your OFBiz implementation
2. Go to the entity definition in the appropriate entitymodel.xml file
3. Disable temporarily the prim-key reference of the added field, and save the file
4. Restart the OFBiz implementation
5. Check with your RDBMS client that the new field is present in the table
6. Add values for the field in the table (if appropriate)
7. Stop the OFBiz implementation
8. Go back to the definition in the appropriate entitymodel.xml file
9. Enable the temporary disabled prim-key reference of the added field, and save the file
10. Check with your RDBMS client that the new field is part of the primary key definition of the table

**Migration Scripts**

1. Migration service migrateProductPromoCodeEmail is implemented to migrate the ProductPromoCodeEmail entity to ProductPromoCodeContactMech. (More detail at [OFBIZ-5426](#))

**4.5.2. Changes with OFBiz 17**

Field types *id-ne, id-long-ne & id-vlong-ne* has been removed. Use *id, id-long* and *id-vlong* instead (detailed description at [OFBIZ-9351](#)).

**Entity Changes**

No changes

**Field Changes**

<table>
<thead>
<tr>
<th>Entity</th>
<th>Field</th>
<th>Action</th>
<th>IsPK</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>MarketingCampaignPrice</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>R1805961</td>
</tr>
<tr>
<td>MarketingCampaignPrice</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>R1805961</td>
</tr>
<tr>
<td>MarketingCampaignPromo</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>R1805961</td>
</tr>
<tr>
<td>MarketingCampaignPromo</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>R1805961</td>
</tr>
<tr>
<td>MarketingCampaignRole</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>R1805961</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------</td>
<td>--------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>MarketingCampaignRole</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>R1805961</td>
</tr>
<tr>
<td>Product</td>
<td>manufacturerPartId</td>
<td>Removed</td>
<td>No</td>
<td>R1804408</td>
</tr>
<tr>
<td>SecurityGroupPermission</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>R1812383</td>
</tr>
<tr>
<td>SecurityGroupPermission</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>R1812383</td>
</tr>
</tbody>
</table>

**Migration Scripts**

1. Updated sql-type for date-time and time field in fieldtypemysql.xml file at commit R1793300

   *Update mysql sql-type for datetime field-type to support Fractional Seconds in Time Values Please upgrade mysql to at least 5.6.4 or higher.*

   After upgrade run `generateMySqlFileWithAlterTableForTimestamps` service, `groupName` is required field for this service.

   It will generate sql file with alter query statement for date-time and time field at location

   ```
   ${ofbiz.home}/runtime/tempfiles/.sql
   ```

   You can use execute sql statement from any of the mysql batch command.

**4.5.3. Changes between OFBiz 9 to OFBiz 16**

**Entity Changes**

**Added 77 new entities**

1. JobRequisition
2. ProductAverageCostType
3. WorkEffortSurveyAppl
4. WorkEffortIcalData
5. WebSiteContactList
6. WebAnalyticsType
7. WebAnalyticsConfig
8. UserLoginSecurityQuestion
9. UomGroup
10. TrainingRequest
11. ThirdPartyLogin
12. TestFieldType
13. TestingSubtype
14. TestingStatus
15. TestingRemoveAll
16. TestingItem
17. TestingCrypto
18. SystemProperty
19. ShipmentGatewayUsps
20. ShipmentGatewayUps
21. ShipmentGatewayFedex
22. ShipmentGatewayDhl
23. ShipmentGatewayConfig
24. ShipmentGatewayConfigType
25. ReturnContactMech
26. QuoteNote
27. ProductPromoContent
28. ProductPromoContentType
29. ProductGroupOrder
30. ProductCostComponentCalc
31. CostComponentCalc
32. PayPalPaymentMethod
33. PaymentGroupType
34. PaymentGroup
35. PaymentGroupMember
36. PaymentGatewayConfig
37. PaymentGatewayConfigType
38. PaymentGatewayWorldPay
39. PaymentGatewaySecurePay
40. PaymentGatewaySagePay
41. PaymentGatewayOrbital
42. PaymentGatewayEway
43. PaymentGatewayCyberSource
44. PaymentGatewayAuthorizeNet
45. PaymentGatewayIDEAL
46. PaymentContentType
47. PaymentContent
48. OAuth2LinkedIn
49. OAuth2GitHub
50. JobManagerLock
51. JobInterviewType
52. JobInterview
53. JavaResource
54. InvoiceNote
55. InvoiceItemAssocType
56. InvoiceItemAssoc
57. InvoiceContentType
58. InvoiceContent
59. GlAccountCategoryType
60. GlAccountCategoryMember
61. GlAccountCategory
62. GitHubUser
63. FixedAssetTypeGlAccount
64. FacilityContent
65. ExcelImportHistory
66. EmplLeaveReasonType
67. EbayShippingMethod
68. EbayConfig
69. CountryAddressFormat
70. ContentSearchResult
71. ContentSearchConstraint
72. ContentKeyword
73. CheckAccount
74. AgreementFacilityAppl
75. AgreementContentType
76. AgreementContent

**Removed 8 entities**

1. DepreciationMethod
2. FixedAssetMaintMeter
3. OagisMessageErrorInfo
4. OagisMessageInfo
5. SalesOpportunityTrackingCode
6. SimpleSalesTaxLookup
### Field Changes

<table>
<thead>
<tr>
<th>Entity</th>
<th>Field</th>
<th>Action</th>
<th>IsPK</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>AcctgTransAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>AcctgTransEntry</td>
<td>inventoryItemId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>AcctgTransTypeAttribute</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>BenefitType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>BenefitType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>BudgetAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>BudgetItemTypeAttribute</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>BudgetStatus</td>
<td>changeByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>BudgetTypeAttribute</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>CommunicationEventType</td>
<td>statusId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>CommunicationEventRole</td>
<td>contactMechTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ContactListCommunicationStatus</td>
<td>partyId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ContactListCommunicationStatus</td>
<td>messageId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ContactListCommunicationStatus</td>
<td>changeByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ContactMechAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ContactMechTypeAttribute</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>DeductionType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>DeductionType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>DocumentAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>---------------------------</td>
<td>----------------</td>
<td>------------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>DocumentTypeAttribute</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmploymentApp</td>
<td>approverPartyId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmploymentApp</td>
<td>jobRequisitionId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmploymentAppSourseType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmploymentAppSourseType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmplPositionClassType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmplPositionClassType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmplPositionType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmplPositionType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmplPositionType</td>
<td>partyId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>EmplPositionType</td>
<td>roleTypeId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FinAccountAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FinAccountTransAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FinAccountTrans</td>
<td>glReconciliationId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FinAccountTrans</td>
<td>statusId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FinAccountTransTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FinAccountTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FinAccountStatus</td>
<td>changeByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FixedAsset</td>
<td>acquireOrderId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FixedAsset</td>
<td>acquireOrderItemSeqId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FixedAssetAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>FixedAssetTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>GlAccount</td>
<td>externalId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>GlAccount</td>
<td>openingBalance</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>----------------------</td>
<td>------------------------</td>
<td>--------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>GlReconciliation</td>
<td>createdDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>GlReconciliation</td>
<td>lastModifiedDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>GlReconciliation</td>
<td>statusId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>GlReconciliation</td>
<td>openingBalance</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InventoryItemAttr</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InventoryItemStatus</td>
<td>changeByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InventoryItemTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InvoiceAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InvoiceItemAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InvoiceItemTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InvoiceStatus</td>
<td>changeByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InvoiceTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>InvoiceTermAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>JobSandbox</td>
<td>currentRetryCount</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>JobSandbox</td>
<td>tempExprId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>JobSandbox</td>
<td>currentRecurrenceCount</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>JobSandbox</td>
<td>maxRecurrenceCount</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>JobSandbox</td>
<td>jobResult</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAdjustment</td>
<td>amountAlreadyIncluded</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAdjustment</td>
<td>isManual</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAdjustment</td>
<td>oldPercentage</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAdjustment</td>
<td>oldAmountPerQuantity</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAdjustment</td>
<td>lastModifiedDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAdjustment</td>
<td>lastModifiedByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>------------------------</td>
<td>--------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>OrderAdjustment Attribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAdjustmentTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItem</td>
<td>supplierProductId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItem</td>
<td>cancelBackOrderDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItem</td>
<td>changByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItemAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItemShipGroup</td>
<td>facilityId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItemShipGroup</td>
<td>estimatedShipDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItemShipGroup</td>
<td>estimatedDeliveryDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItemShipGrpInvRes</td>
<td>priority</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItemShipGrpInvRes</td>
<td>oldPickStartDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderItemTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderTermAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderPaymentPreference</td>
<td>track2</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderPaymentPreference</td>
<td>swipedFlag</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderPaymentPreference</td>
<td>lastModifiedDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderPaymentPreference</td>
<td>lastModifiedByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderShipment</td>
<td>shipGroupSeqId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>OrderTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyAcctgPreference</td>
<td>orderSequenceEnumId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------</td>
<td>----------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>PartyAcctgPreference</td>
<td>quoteSequenceEnumId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyAcctgPreference</td>
<td>invoiceSequenceEnumId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyAcctgPreference</td>
<td>oldOrderSequenceEnumId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyAcctgPreference</td>
<td>oldQuoteSequenceEnumId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyAcctgPreference</td>
<td>oldInvoiceSequenceEnumId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyAcctgPreference</td>
<td>orderSeqCustMethId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyQual</td>
<td>infoString</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyQual</td>
<td>institutionInternalId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyQual</td>
<td>institutionPartyId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyQual</td>
<td>partyQualId</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyRate</td>
<td>percentageUsed</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyRate</td>
<td>rate</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PartyResume</td>
<td>contentId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PaymentAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PaymentGatewayResponse</td>
<td>gatewayCvResult</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PaymentMethod</td>
<td>finAccountId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PaymentTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfRatingType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfRatingType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfReview</td>
<td>payHistoryRoleTypeIdTo</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfReview</td>
<td>payHistoryRoleTypeIdFrom</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfReview</td>
<td>payHistoryPartyIdTo</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfReview</td>
<td>payHistoryPartyIdFrom</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfReview</td>
<td>payHistoryFromDate</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------</td>
<td>--------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>PerfReviewItemTy</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PerfReviewItemTy</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PersonTraining</td>
<td>trainingRequestId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PersonTraining</td>
<td>workEffortId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PersonTraining</td>
<td>approverId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PersonTraining</td>
<td>approvalStatus</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PersonTraining</td>
<td>reason</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PostalAddress</td>
<td>houseNumber</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PostalAddress</td>
<td>houseNumberExt</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PostalAddress</td>
<td>cityGeoId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PostalAddress</td>
<td>municipalityGeoId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PostalAddress</td>
<td>geoPointId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PosTerminal</td>
<td>terminalName</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>PosTerminalInter</td>
<td>reasonEnumId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>releaseDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>originalImageUrl</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>inventoryItemTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>shippingWeight</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>productWeight</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>diameterUomId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>productDiameter</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>virtualVariantMethodId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>defaultShipmentBoxType</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>lotIdFilledIn</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>orderDecimalQuantity</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>weight</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>taxCategory</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Product</td>
<td>taxVatCode</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------------</td>
<td>----------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>Product</td>
<td>taxDutyCode</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductAverageCost</td>
<td>productAverageCostTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductAverageCost</td>
<td>facilityId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductContent</td>
<td>sequenceNum</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductKeyword</td>
<td>keywordTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductKeyword</td>
<td>statusId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductRole</td>
<td>sequenceNum</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>balanceResOnOrderCreation</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>defaultTimeZoneString</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>oldStyleSheet</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>oldHeaderLogo</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>oldHeaderRightBackground</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>oldHeaderMiddleBackground</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>styleSheet</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>headerLogo</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>headerRightBackground</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStore</td>
<td>headerMiddleBackground</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStorePaymentSetting</td>
<td>paymentCustomMethodId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStorePaymentSetting</td>
<td>paymentGatewayConfigId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStoreShipmentMethod</td>
<td>shipmentCustomMethodId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStoreShipmentMethod</td>
<td>shipmentGatewayConfigId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductStoreShipmentMethod</td>
<td>allowancePercent</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------------</td>
<td>--------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>ProductStoreShipMentMeth</td>
<td>minimumPrice</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ProductTypeAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteAdjustment</td>
<td>lastModifiedDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteAdjustment</td>
<td>lastModifiedByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteItem</td>
<td>leadTimeDays</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteRole</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteRole</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteTerm</td>
<td>termDays</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteTerm</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteTerm</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteTerm</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>QuoteTerm</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>RequirementAttribute</td>
<td>changeByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>RequirementStatus</td>
<td>changeByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ResponsibilityType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ResponsibilityType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ReturnAdjustment</td>
<td>createdByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ReturnAdjustment</td>
<td>lastModifiedDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ReturnAdjustment</td>
<td>lastModifiedByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ReturnItemResponse</td>
<td>supplierRmaId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ReturnItemResponse</td>
<td>financialAccountTransId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ReturnStatus</td>
<td>changeByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>SalaryStep</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>NA</td>
</tr>
<tr>
<td>SalaryStep</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>------------------------</td>
<td>----------------------------</td>
<td>--------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>SalaryStep</td>
<td>createdByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>SalaryStep</td>
<td>lastModifiedByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>SalesOpportunity</td>
<td>nextStepDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ServiceSemaphore</td>
<td>lockedByInstanceId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ShoppingListItem</td>
<td>modifiedPrice</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>SkillType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>SkillType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>SupplierProduct</td>
<td>shippingPrice</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>SupplierProduct</td>
<td>supplierCommissionPercent</td>
<td>Removed</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TaxAuthorityRateProduct</td>
<td>isTaxInShippingPrice</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TerminationType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TerminationType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TestingNodeMember</td>
<td>extendFromDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TestingNodeMember</td>
<td>extendThruDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TimeEntry</td>
<td>planHour</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Timesheet</td>
<td>approvedByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TrainingClassType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>TrainingClassType</td>
<td>hasTable</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>UnemploymentClaim</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>UserLogin</td>
<td>externalAuthId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>UserLogin</td>
<td>userLdapDn</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>UserLogin</td>
<td>disabledBy</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>ValueLinkKey</td>
<td>createdByUserLogin</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WebSite</td>
<td>visualThemeSetId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WebSite</td>
<td>hostedPathAlias</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WebSite</td>
<td>isDefault</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>Entity</td>
<td>Field</td>
<td>Action</td>
<td>IsPK</td>
<td>Revision</td>
</tr>
<tr>
<td>----------------------------</td>
<td>------------------------------------</td>
<td>--------</td>
<td>------</td>
<td>----------</td>
</tr>
<tr>
<td>WebSite</td>
<td>displayMaintenancePage</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WebSitePathAlias</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>R1738588</td>
</tr>
<tr>
<td>WebSitePathAlias</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>R1738588</td>
</tr>
<tr>
<td>WorkEffort</td>
<td>tempExprId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffort</td>
<td>sequenceNum</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortAttribute</td>
<td>attrDescription</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortAttribute</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortContactMech</td>
<td>fromDate</td>
<td>Added</td>
<td>Yes</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortContactMech</td>
<td>thruDate</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortFixedAssetAssign</td>
<td>availabilityStatusId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortPartyAssignment</td>
<td>assignedByUserLoginId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortPurposeType</td>
<td>parentTypeId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortStatus</td>
<td>reason</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkEffortTypeAttr</td>
<td>description</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
<tr>
<td>WorkOrderItemFulfillment</td>
<td>shipGroupSeqId</td>
<td>Added</td>
<td>No</td>
<td>NA</td>
</tr>
</tbody>
</table>
5. Service Engine

5.1. Declaration and Implementation

5.2. Supported languages

5.3. Transaction management

5.4. Web services
6. Widget System

6.1. Screen Widget

6.1.1. Decoration Pattern

6.2. Form Widget

6.3. Menu Widget

6.4. Tree Widget

6.5. Portal Widget

6.6. Platform Specific Code

6.7. Apache OFBiz Themes

The Apache OFBiz Project Release 17.12

6.7.1. What is a theme

A Theme is an ofbiz component that defines all elements necessary to render all information generated by the screen engine through an embedded technology. Currently themes present in Apache OFBiz use html5/jquery/css to do that.

To realize that, a theme can define some properties, among them some can be necessary. It can define its own ftl macro to render the different modelScreen elements and can define its own screen decorator to prepare the final user screen structure for the technology used by the theme.

6.7.2. How to define it

A theme is a standard component, present in the directory themes or plugins with a file definition present on widget/Theme.xml and support the widget-theme.xsd

To offer the possibility for end users to select the theme through the "Select Theme Screen", the theme need to load the entity VisualTheme and one or more entries related to this theme.

6.7.3. Structure of Theme.xml

The theme definition file help OFBiz to know what specific rendering the Theme want to use. It's composed by two definition elements and four optional blocks.
**Main definition**

The First mandatory element is one or several visualThemeId related to the theme Defined like that

```xml
<visual-themes>
    <visual-theme id="MY_THEME"/>
</visual-themes>
```

You need to define in the database an entry in VisualTheme Entity for each visual-theme id defined.

```xml
<entity-engine-xml>
    <VisualTheme visualThemeId="MY_THEME" visualThemeSetId="BACKOFFICE"
                 description="My theme - Example (based on flatgrey")/>
</entity-engine-xml>
```

A theme component can load one or more visual theme id. Usually only one is present.

The second important (but not mandatory) element is `implements`

```xml
<extends location="component://common-theme/widget/Theme.xml"/>
```

This element indicates that your theme copies from the extend theme all information not present in its file definition. If this element isn't present in your theme, you will need to define all information present in common-theme to be sure that OFBiz misses nothing for a correct run. Otherwise some functionalities can be broken…

It's highly recommended to extend the common-theme to be sure that your theme works correctly and to surcharge only what you need.

The four following blocks are optionnal if you define an extends theme

**General properties**

This block contains all properties that the screen engine can use to prepare the rendering and that the theme can implement
<widget-properties>
  <!--Transversal properties relative to ofbiz widget component-->

  <default-view-size value="20"/>
  <autocompleter
    default-view-size="10"
    default-min-length="2"
    default-delay="300"
    display-return-field="true"/>
  <lookup
    position="topleft"
    width="640"
    height="500"/>
  <layered-modal
    width="800"
    height="600"/>

</widget-properties>

Theme's specific properties

This block contains all properties specific to this theme. In general these are some properties present on ftl template, that are initialized by the theme and can be surchaged by another theme through the extends elements.

<property name="jgrowlPosition" value="center" type="String"/>
<!--possible value: top-left, top-right, bottom-left, bottom-right, center-->
<property name="jgrowlWidth" value="800" type="Integer"/>
<property name="jgrowlHeight" value="" type="Integer"/>
<property name="jgrowlSpeed" value="100" type="Integer"/>

Ftl macro library

This block defines for each technology implemented by the screen engine where it can find the macro library for each model (Screen, Form, Tree, Menu)
If you want surcharge some macros, you can just create the desired macros and import the others from common-theme (at the top of file) like that:

```xml
<#include "component://common-theme/template/macro/HtmlFormMacroLibrary.ftl"/>
```

**Screen library**

This block defines where OFBiz can find all official screens definitions in framework/common

Normally, you don’t need to change this file except if you need to define a default screen style that doesn’t exist OOTB. If you need to extend an existing one, you have to do it in the theme directory.

To define a new default screen style, you have to add it in this file, and point to the screen decorator in common-theme to define your default screen style as the default one.
Screens are separated in three types:

- structural-decorator: contains all decorators that organise the screens structures
- embed-decorator: decorator used only on sub screens
- general-screen: list all generic inter applications screens

### 6.7.4. The common-theme

This is the root theme that contains all information to ensure a good basic theme for OFBiz. Currently it keeps all old themes system for backward compatibility with ftl template managed by the entity `VisualThemeResource`

### 6.7.5. Create your own theme

As a theme is a component, you can create a new theme like a plugin.

After creating a component, you can add the two minimal information:

- Theme.xml file in `plugins/my-theme/widget/` with minimal information:

```xml
<theme name="my-theme"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="http://ofbiz.apache.org/dtds/widget-theme.xsd">
  <visual-themes>
    <visual-theme id="MY_THEME" display-name="My Theme"/>
  </visual-themes>
</theme>
```

- your data file to add your visual theme in `plugins/my-theme/data/`

```xml
<entity-engine-xml>
  <VisualTheme visualThemeId="MY_THEME" visualThemeSetId="BACKOFFICE"/>
</entity-engine-xml>
```

The presence of `VisualTheme` entity helps to indicate which theme is available in your instance, specially helpful for tenant installations.

To display your theme in OFBiz theme library, you can complete the information on each visual theme like
display-name and description support the flexibleStringExpander syntax

extends common-theme

This is a first step to understand how the theme system works. With your new theme, you can try to surcharge different elements.

To start, extends the common-theme:

```xml
<theme name="my-theme"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://ofbiz.apache.org/dtds/widget-theme.xsd">
  <visual-themes>
    <visual-theme id="MY_THEME" display-name="My Theme">
      <description>My new funny theme under nice tecnno</description>
      <screenshot location="/mytheme/screenshot1.png"/>
      <screenshot location="/mytheme/screenshot2.png"/>
    </visual-theme>
  </visual-themes>
  <extends location="component://common-theme/widget/Theme.xml"/>
</theme>
```

Now your theme should be operational, but without particularity.

You can surcharge a ftl macro, to do this create your own ftl macro file in plugins/mytheme/templates/macro/HtmlFormMacroLibrary.ftl with
Now indicate to your theme that you want use this library

```xml
  <visual-themes>
    <visual-theme id="MY_THEME" display-name="My Theme">
      <description>My new funny theme under nice techno</description>
      <screenshot location="/mytheme/screenshot1.png"/>
    </visual-theme>
    <visual-themes>
  <extends location="component://common-theme/widget/Theme.xml"/>
  <templates>
    <template name="screen" type="html" content-type="UTF-8" encoding="none"
      encoder="html" compress="false">
      <template-file widget="form" location="component://my-theme/template/macro/HtmlFormMacroLibrary.ftl"/>
    </template>
    <templates>
  </theme>
```

and check the result when you select your theme. The result isn’t really interesting but it’s to understand how it works.

**create from scratch**

**TODO...**

### 6.7.6. Backware compatibility with OFBiz 16.11 and above

**How themes worked before**

Before the theme management by model definition, all configurations have been present in the database through entity VisualTheme and VisualThemeRessource. These ressources were loaded in a layoutProperties variable and used directly by decorator screens and ftl templates.
Now with the common-theme

All this logic is still present in the common-theme template to keep backward compatibility, but the VisualThemeRessource is now useless and properties have been migrated to the Theme definition in the part theme-properties

Example with BlueLight

The blue light theme has been these properties in VisualThemeRessource:

```xml
<VisualTheme visualThemeId="BLUELIGHT" visualThemeSetId="BACKOFFICE"
  description="Blue light Theme: breadcrumbs, drop-down menus and rounded corners"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_NAME"
  resourceValue="BLUELIGHT" sequenceId="01"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_IMAGE_URL"
  resourceValue="/images/ofbiz_logo.png" sequenceId="01"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_SHORTCUT_ICON"
  resourceValue="/images/ofbiz.ico" sequenceId="01"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_SCREENSHOT"
  resourceValue="/bluelight/screenshot.jpg" sequenceId="01"/>

<!-- CSS references -->
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_STYLESHEET"
  resourceValue="/bluelight/style.css" sequenceId="01"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HELPSTYLESHEET"
  resourceValue="/bluelight/help.css" sequenceId="01"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_DOCBOOKSTYLESHEET"
  resourceValue="/bluelight/webapp/bluelight/docbook.css" sequenceId="01"/>

<!-- Javascript references -->
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT"
  resourceValue="/common/js/jquery/jquery-1.11.0.min.js" sequenceId="01"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT"
  resourceValue="/common/js/jquery/jquery-migrate-1.2.1.js" sequenceId="02"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT"
  resourceValue="/common/js/jquery/ui/js/jquery-ui-1.10.3.min.js" sequenceId="03"/>
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT"
  resourceValue="/common/js/jquery/plugins/asmselect/jquery.asmselect-1.0.4a-beta.js" sequenceId="05"/>
<VisualThemeResource visualThemeId="BLUELIGHT"
```
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/plugins/datetimepicker/jquery-ui-timepicker-addon.min-1.4.3.js" sequenceId="07"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/plugins/fjTimer/jquerytimer-min.js" sequenceId="09"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/plugins/mask/jquery.mask-1.14.13.min.js" sequenceId="10"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/plugins/jeditable/jquery.jeditable-1.7.3.js" sequenceId="11"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/jquery/plugins/validate/jquery.validate.min.js" sequenceId="12"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/plugins/OpenLayers-2.13.1.js" sequenceId="13"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/plugins/OpenLayers-2.13.1.js" sequenceId="13"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/OfbizUtil.js" sequenceId="15"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/fieldlookup.js" sequenceId="16"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/plugins/date/date.format-1.2.3-min.js" sequenceId="17"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/plugins/date/date.timezone-min.js" sequenceId="18"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/miscAjaxFunctions.js" sequenceId="19"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/selectMultipleRelatedValues.js" sequenceId="20"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/util/util.js" sequenceId="21"/>

<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/common/js/plugins/date/FromThruDateCheck.js" sequenceId="22"/>

<!-- ftl references -->
<VisualThemeResource visualThemeId="BLUELIGHT" resourceTypeEnumId="VT_HDR_JAVASCRIPT" resourceValue="/bluelight/dropdown.js" sequenceId="30"/>
Now it’s just

```
<VisualTheme visualThemeId="BLUELIGHT" visualThemeSetId="BACKOFFICE"/>
```

And on theme definition

```
<theme-properties>
   <!--javascript lib-->
   <property name="VT_HDR_JAVASCRIPT["add"]" value="/bluelight/dropdown.js" sequenceId="30"/>
   <!--Css style-->
   <property name="VT_STYLESHEET["add"]" value="/bluelight/style.css"/>
   <property name="VT_HELPSTYLESHEET["add"]" value="/bluelight/help.css"/>
   <property name="VT_DOCBOOKSTYLESHEET["add"]" value="/bluelight/webapp/bluelight/docbook.css"/>
   <!--template location-->
   <property name="VT_HDR_TMPLT_LOC" value="component://bluelight/template/Header.ftl"/>
   <property name="VT_FTR_TMPLT_LOC" value="component://bluelight/template/Footer.ftl"/>
   <property name="VT_NAV_OPEN_TMPLT" value="component://bluelight/template/AppBarOpen.ftl"/>
   <property name="VT_NAV_CLOSE_TMPLT" value="component://bluelight/template/AppBarClose.ftl"/>
   <property name="VT_MSG_TMPLT_LOC" value="component://bluelight/template/Messages.ftl"/>
</theme-properties>
```

Values with /images/... have been moved to the common-theme that bluelight extends, the theme definition keeps only what the theme adds to the extended theme.
property name supports the FlexibleMapAccessor syntax, so you can continue to populate a list (VT_STYLESHEET['add']), reset a list (VT_STYLESHEET[]) or add an element on the top list (VT_STYLESHEET[+0]) because some time the order libraries loading is important

Migrate you own theme

Easily, create you Theme.xml and move your VisualThemeResource in theme-properties like in the BlueLight example above.
Maybe you will need to update your template because the modelTheme return ressources not always as list. So :
<property name="VT_HDR_TMPLT_LOC" value="component://bluelight/template/Header.ftl"/> → return a String with component://bluelight/template/Header.ftl
<property name="VT_STYLESHEET['add']" value="..." → return a List<String>
7. FrontJs Portal

frontJsPortal component included in frontJsPortal plugin aim to handle user interface with modern javascript framework (Vue.Js, React, Angular, etc...).

It use portlet/portal system defined by XML and a javascript application.

This component is actually at POC (Proof Of Concept) state, so it aim to show some concrete ideas in order to discuss about them.

It is produced in an "Agility" spirit which mean that it focus to result in despite of some technical debt.

This documentation aim to,

- Explain the current state of the component, how it work and what points had been choose or took away.
- Prepare future documentation by explaining element meant for future users.

The two main goal of using a javascript framework for user interface is

1. to have a "modern" look and feel
   - using recent GUI library
   - usable in multiple context (PC, tablet, smartphone)
2. to increase interactive elements in screen, even if theses screens are based on standard modules :
   - update some part of the screen by action or by data update;
   - modify forms field as function of other field values;
   - simpler screen configuration (portal page) without having to worry about portlets interaction.

7.1. POC Vuejs Renderer

Actually the first javascript application enabling to use portlet system - portal Apache OFBiz is written with the Vue.js framework.

Vue.js has been choose for its easy learning curve and its community driven development instead of a corporation.

In Vue.js, library vuetify has been choosen, because it's base on material design and have e very active community, so seem to be the most used vue.js GUI libraries.

7.1.1. POC Vuejs Renderer installation

1. From a standard trunk ofbiz-framework with 205914a1beb commit or superior (that can be downloaded at documentation standard SourceRepository using git).
2. Create the 'plugins' repository at ofbiz-framework root folder.
3. In the Apache OFBiz plugins repository, download the next repository using git:
   a. example plugin at http://svn.apache.org/repos/asf/ofbiz/ofbiz-plugins/trunk/example

4. In the ofbizextra plugins repository, download the 3 next repositories using git:
   a. vuejsportal plugin that can be downloaded at https://gitlab.ofbizextra.org/ofbizextra/ofbizplugins/vuejsPortal all vuejs components, a new ScreenViewHandler and 4 new renderers (screen, form, menu, tree) and all common files to all fjs component.
   b. examplefjs plugin that can be downloaded at https://gitlab.ofbizextra.org/ofbizextra/ofbizplugins/examplefjs.git specifics files for using vue.js with ofbiz example component
   c. flatgreyfjs plugin that can be downloaded at https://gitlab.ofbizextra.org/ofbizextra/ofbizplugins/flatgreyfjs.git dedicated theme for vue.js with vuetify library

5. In the ofbizextra git repository download ofbizJiraPatchAvailable project
   If there are some files, some patch may be necessary for this POC, for each patch you can goto JIRA to have more explanation for each.
   To apply them it is recommended to create a branch on your ofbiz, and apply the patches with git command `git am patchName`. Currently these patches are necessary:
   a. OFBIZ-11645_0001-simple-methods-optional-in-compound-file.patch ⇐ compound-widget files are used in POC
   b. OFBIZ-11676_0001-Fixed-drop-down-field-not-work-when-in-a-compound-wi.patch ⇐ compound-widget files are used in POC
   c. OFBIZ-11708_0001-fixed-sort-order-field-not-works-in-CompoundWidgets.patch ⇐ compound-widget files are used in POC

6. If there are some files in the ofbizCommit2add, it's some patch waiting jira creation or dedicated for this POC, so it's needed to apply them too with `git am fileName` at the ofbiz root directory (replace fileName by the patch file name and path)

7. After that you have to modify some files of ofbiz-framework (ex: model.java because some tag are added).
   For that, a (bash) script exist, it's located in the 'plugins/VuejsPortal':
   ```bash
   ./tools/applyOfbizFilesPatchs.sh
   ```
   And then copy some files
   ```bash
   rsync -a --exclude-from=./ofbizFiles/rsyncExcludes ./ofbizFiles/ ../../
   ```

8. To be able to build the vue.js application you must have Node.js installed on your system.
   a. To install Node.js on a debian based system, execute the following commands:
      ```bash
      curl -SL https://deb.nodesource.com/setup_8.x | sudo -E bash -
      sudo apt-get install -y nodejs
      ```
   b. Meanwhile an integrated gradle command
      i. Being located at 'plugins/vuejsPortal/webapp/vuejsPortal'
         - `npm i` ⇐ to install the project dependencies
         - `npm run build-prod` ⇐ to build the project (it's possible to use build-dev to be able to use browser dev-tools)
9. You can now launch OFBiz as regular:

   ./gradlew cleanAll loadAll ofbiz

   (being located at the root of ofbiz-framework)

10. You can now connect to the application portal at https://localhost:8443/examplefjs/control/main

### 7.1.2. POC choices

Apache OFBiz **trunk branch** usage.

Principle and architecture of Apache OFBiz **Portal Page and Porlet** usage.

The whole use cases are applied to **example component** and during the POC some new files used which should to be located in common have been moved to examplefjs aiming to simplify the installation and update process.

With examplefjs, there is an **additional webapp (examplefjs)**, it centralize allowed URIs for vuejsPortal component, currently this applicationName is listed in constantes.js file.

vuejsPortal component is not bound to a component, it just need a base URI for send its requests. This uri is temporary set in constantes.js but it will be more flexible later aiming to use vuejsPortal with multiple components.

Portal component (https://localhost:8443/examplefjs/control/login) use **ofbiz standard login by cookies** mechanism, it will later use ofbiz login by token (cf. [https://issues.apache.org/jira/browse/OFBIZ-9833](https://issues.apache.org/jira/browse/OFBIZ-9833)).

**screens, forms, menus** used for portlets are defined with **xml**, and dedicated files for better readability.

There is a **component (vuejs) for each ofbiz screen element** (SingleItemRow ⇒ vue.single-item-row), which are defined by renderer for screen, form and menu (and for html renderer that correspond globally to macro.ftl). Actually, for the POC, to list them, you can search in vuejsportal existing file for Vue*.vue in webapp/vuejs-portal/src/component/.

**XML tags used**

During the POC, few new properties or XML tags had been added but some of existing tags and properties had been distorted for an exclusive Vue.Js use.

- **screen**
  - in `<container` tag
    - container is the main piece of dynamics screen (update a part of screen on event)
      - `id` is used to identify the container and enable to inject content into it with `set-area` later.
      - When set-area is done for this id, its content will be replace by the content received from area-target.
      - So, do not resend the container!
    - `watcher-name` (new attribute) is a name in the local store (in web browser) which can contain some data (parameters). It’s possible to update watcher content with a `set-watcher`. Each time the watcher content changes, the `auto-update-target` is called using
watcher content as parameters to update the container. If refresh-area is call, the refresh is done by using auto-update-target and watcher content (if exist).

- **auto-update-target** is used to bind a watcher to the container.
  To use it with set-watcher, a new attribute watcher-name has been added, and when the content of this this watcher changes a set-area is done with auto-update-target as target and watcher content as parameters.
  auto-update-target support the REST uri format with some parameters in it include with {}, value is populate with content of watcher content. If uri contain, at least one {xxx} or have no parameters a GET method will be used otherwise a POST will be used.

- **screenlet**
  - collapsible, if collapsible = "true" then it's necessary to have a "unique" id to know what collapse when button is clicked. It's possible to have multiple screenlet with same id or each with a different one.
    So, it's important to give a id if collapsible ="true" otherwise collapse will collapse all screenlet without id!
  - id : is manatory if the screenlet should be collapse

- **in platform-specific-html tag a new tag**
  - vuejs is to call a specifics vuejs component named component-name.
    it's possible to give to it some parameters with sub tag parameters

- **form**
  - property entity-name is mandatory for grid or single form, vuejs use it to build entity store.
    When entity-name, exist data print to screen are associated to the web-browser local store, so when a data change in one screenlet, same data in an other screenlet will change too.
  - in <on-event-update-area tag (there can be many of them and they will be executed in xml apparition order. It will wait current task is completed before starting the next one. If any of these events is set, the submit button will prevent the form default behaviour, so if you need to post in addition to vuejsPortal feature you must add an event-type submit)

- **event-type**
  Define the action to be executed during the form submitting (click on submit button). It can take one of next values :
  - submit will submit the form to area-target with provided parameter elements. Like set-area if no parameter are defined it will use the whole form. Error and success messages will be displayed.
    Form target support the REST uri format with some parameters in it include with {}, value is populate with content of form - fields when submit.
    The http method used is POST, but you can add a field _method to choose the method received by the ofbiz controler (ex: PUT for an update)
  - set-area update area with id areaId with the post result at URI area-target with included parameter elements.
    If uri contain, at least one {xxx} or have no parameters a GET method will be used otherwise a POST will be used.
  - refresh-area similar with set-area but target used will be the auto-update-target
define at the container level and, if a watcher-name is define for this container, it's content will be used as parameters (for solving {} in target or as parameters if no {} in target.

- **set-watcher** update watcher **areaId** with included parameter. In case no parameter was passed to **set-watcher** it will use all form fields to update watcher content.

- **refresh-watcher** all container which watch a watcher-name are update when watcher content change, with refresh-watcher update is done without change watcher content.

- **collapse** do an action on collapse/expand screenlet.
  - **area-target** is the action to do: collapse or expand or toggle;
  - **area-id** is the screenlet name to change.

- **close-modal** close the current modal, currently **area-id** is necessary, when modal is open for a lookup areaId is {formName}_{fieldName}; area-target is not used, so can be left empty.

  - in **on-field-event-update-area** tag (there can be many of them and they will be executed in xml apparition order. It will wait current task is completed before starting the next one.)

  - **event-type** Define the action to be executed during the click on the associated hyperlink (<a link or button). It can take one of next values :

    - **post** will post at URI **area-target** with the parameter elements included in the tag. If there are no parameters, it will use the whole form. It will display error and success messages.

    - **put** will post at URI **area-target** with the http method PUT if target contain {xxxx} xxxx will be replace by value if xxxx is define in parameters included in the tag. If there are no parameters, it will use the whole form. It will display error and success messages.

    - **delete** will post at URI **area-target** with the http method DELETE if target contain {xxxx} xxxx will be replace by value if xxxx is define in parameters included in the tag. If there are no parameters, it will use the whole form. It will display error and success messages.

    - **set-area** update area with id **areaId** with the post result at URI **area-target** with included parameter elements.
      - If uri contain, at least one {xxx} or have no parameters a GET method will be used otherwise a POST will be used.

    - **refresh-area** similar with set-area but target used will be the auto-update-target define at the container level and, if a watcher-name is define for this container, it's content will be used as parameters (for solving {} in target or as parameters if no {} in target.

    - **set-watcher** update watcher **areaId** with included parameter. In case no parameter was passed to **set-watcher** it will use all form fields to update watcher content.

    - **refresh-watcher** all container which watch a watcher-name are update when watcher content change, with refresh-watcher update is done without change watcher content.
• set-field-in-form will set the value of property area-target from form area-id from the actual field value.

• collapse do an action on collapse/expand screenlet.
  area-target is the action to do: collapse or expand or toggle;
  area-id is the screenlet name to change.

• close-modal close the current modal, area-id is to know which modal close, when modal is open for a lookup areaId is {formName}_{fieldName};
  if area-id is empty all modal are closed;
  area-target is not used, so can be left empty.

○ in parameters tag
  • auto-parameters-portlet allow to automatically fill necessary portlet parameters:
    • portalPageId
    • portalPortletId
    • portletSeqId
    • currentAreaId

○ in <hyperlink tag
  • link-type="anchor" is used to make an internal call to FrontJs via set-area, target-window is used as id to update and target as area-target, parameters as parameters.
  • target-window is used to design the watcherName that have to be updated.
  • url-mode three new modes is added to be able to give the http method to use, POST is the default
    • intra-post : do a http POST request for the set-area, and so sending multiple parameters
    • intra-put : do the http request with a PUT method
    • intra-delete : do the http request with a DELETE method

• menu
  ○ in <link tag
    • link-type="set-area" is used to make an internal call to FrontJs via set-area, target-window is used as id to update and target as area-target, it's a http get so parameters should be in target and if there area some {xxx} in target they will be replace by correct value with fields list in parameters.
    • link-type="auto" (default value) and url-mode="intra-app" call the target-url to print a new page (using internal routing).
    • link-type="layered-modal" is used to open a modal windows and do a set-area in it
  • url-mode three new modes is added to be able to give the http method to use, POST is the default (except for set-area)
    • intra-post : do a http POST request for the set-area, and so sending multiple parameters
    • intra-put : do the http request with a PUT method
• intra-delete: do the http request with a DELETE method

**decorator, lookup**

Currently vuejs app is call in a ftl included in the classic mainDecorator, without application menu (which are directly manage by vuejsApp cf **Specifics URI**).

Currently a specifics theme has created, to be able to remove most of css to avoid "conflict" with vuetify lib. This theme is forced in the main decorator of the application.

**Lookup**

1. **LookupDecorator** is a specifics one for VueJs, but its usage is same as the standard one. It contain 2 main specificity
   - for autocomplete return, all data are in viewData
   - for search - result, when lookup receive parameters.lookupResult = "Y" only result form is sent to be injected in area lookup-results

2. **LookupForm - find**:
   - it's necessary to add a hidden field
     ```html
     <field name="lookupResult"><hidden value="Y"/></field>
     ```
   - it's necessary to add 2 on-event-update-area,
     first one to send result in the area lookup-result second one to collapse Find screenlet (if you want collapse it)
     ```html
     <on-event-update-area event-type="set-area" area-id="lookup-results" area-target="LookupExample"/>
     <on-event-update-area event-type="collapse" area-id="LookupFindScreenlet" area-target="collapse"/>
     ```

3. **LookupForm - list**:
   - for the link return field, 2 on-field-event-update-area must be added
     first one to set Field which call the lookup with the correct value
     second one to close the modal
     ```html
     <on-field-event-update-area event-type="set-field-in-form" area-id="SelectExample" area-target="exampleId" />
     <on-field-event-update-area event-type="close-modal" area-id="SelectExample_exampleId" area-target=""/>
     ```

4. **FindScreenDecorator**, is a specifics one for VueJs, but its usage is same as the standard one. Its specificity is about ListResult update. Some modifications should be done in the find form
   - add a hidden field onlyList
FrontJs Renderer

A new **viewHandler** and a new set of **renderer** had been created.
A new package had been created in org.apache.ofbiz.widget.renderer named frontJs which contain all new renderer.

In this package, there is a new FrontJsOutput class which allow to construct necessary elements whit the wished format. An object of this class is instantiate at the begin of viewHandler process, then is completed by rendered object call.

Uri **showPortletFj** of component examplefjs use the new viewHandler to return screen render result, showPortlet.

This URI is used by Vue.Js application for display information gathering.
New viewHandle return a json which contain 2 maps (viewScreen and viewEntities).
viewScreen contain all information about display. viewEntities contain all information about data set.

In future revision, it will probably be feasible to only receive the data map.

**Specifics URI**

there is a **showPortal** URI in vuejsPortal (CommonScreen), which is the Start point of the VueJs application.

**portalPageDetail** is a new URI (temporary for the POC), used by the POC to received data about the portalPage : list of column, list of portlet

**showPortlet** screen is redefine in this POC for simplification purpose, to only do what we want.

**applicationMenu** is a new uri which return with the FrontJsScreenViewHandler the data (ViewScreen & ViewData) from application xml menu.
In each fjs component, it's necessary to define a view-map entry applicationMenu pointing to a
screen.xml with only the correct menu.

REST URI

For OFBiz applications URI structure is: resourceName}{cover}{Pkvalue: .*)

So, for Example resource, it translate into:

<table>
<thead>
<tr>
<th>URI</th>
<th>method</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example/find</td>
<td>get</td>
<td>find form to select options for list</td>
</tr>
<tr>
<td>Example/list</td>
<td>get (via post)</td>
<td>Example list, ⇒ with a get (so without parameters) all the examples ⇒ with a post (parameters and _method=&quot;GET&quot;) examples selected by options</td>
</tr>
<tr>
<td>Example/create</td>
<td>get</td>
<td>create form</td>
</tr>
<tr>
<td>Example/edit/{exampleId}</td>
<td>get</td>
<td>edit form and data for the id sent</td>
</tr>
<tr>
<td>Example/show/{exampleId}</td>
<td>get</td>
<td>show form and data for the id sent</td>
</tr>
<tr>
<td>Example/summary/{exampleId}</td>
<td>get</td>
<td>summary form and data for the id sent</td>
</tr>
<tr>
<td>Example/data/{exampleId}</td>
<td>get</td>
<td>just data for the id sent</td>
</tr>
<tr>
<td>Example/change</td>
<td>post</td>
<td>create a example</td>
</tr>
<tr>
<td>Example/change/{exampleId}</td>
<td>put (via post)</td>
<td>update example with exampleId</td>
</tr>
<tr>
<td>Example/change/{exampleId}</td>
<td>delete</td>
<td>delete example with exampleId</td>
</tr>
</tbody>
</table>

All these actions([cover]) is about screenlet (part of a page), if cover should be for a page, its name is suffix by Pg.

HTTP request other than POST method cannot have parameters (other than in URI) so to be able to do a GET or a PUT with parameters a workaround is to used a POST with added `<field name="_method"><hidden value="GET"> (or "PUT") in the form (or as one of the parameters).

TOMCAT httpServlet.java not manage PATCH method, so it’s not usable

Added field

A new field had been added to PortalPagePortlet entity: **watcherName** which is the name of the watcher that fire portlet update.
ftl file can be mixed with Vue.js app, so it’s necessary to create a dedicated component by ftl migrated. Look at XML tags used on point platform-specific-html-vuejs.

7.1.3. POC current situation

Currently VueJs POC is only able to manage PortalPage, but it will be simple to manage all other screen types.

The Objective here is to make different use cases portal page to work.

At the menu, there are different pages, actually the name of portal pages name are MgmtPageFrontJs and RecapPageFrontJs.

The first page aim to find and edit examples. The second page aim to show all properties of a given example.

For each portlet present in theses pages, a portlet component is created in Vue.js with the portlet name which initialise itself with showPortletFj.

ShowPortletFj give the component all information about components it have to render (viewScreen) and data set it have to work with (viewEntities). Actually ShowPortletFj respond with a JSON.

ViewScreen and ViewEntities

Actually FrontJsViewHandler send all information to component who need update. In a client side
logic, ViewScreen would be send only the first time while the portlet call information then would only care about ViewEntity which update the data set it work on.

Actually "use-when" are handled by ofbiz in model model management whatever level they belong to (screen, form, field, menu), so we can’t get this information in the renderer. Later, we had to transmit to renderer all screen elements and if there was "use-when" for manage them in the frontend. This way next call could sent "use-when" correct value to properly update the final screen.

7.1.4. POC’s remaining todos

This list contain only points which are already meet in use case process.

Most of time, each new use case (or portlet) give new task (ex: new field type not yet meet, or field properties not yet process)

Minors

• review and change field-find to include title in the field to remove the "title" coloumn in find form
• Manage the ConfMod in the showPortalPage, to show (and manage) the portlet parameters page.
• do a uri to send label for login screen, to be able to have translation in login screen
• Include header and footer in the VueJs application to have a full page manage by VueJs, and so have a complete Vue.Js look.
• Include all Common and Commonext part, like systemInfoNote

Majeurs

• use VuejsRouter to be able to manage all screenType
• showPortletFj* must verify security about field SecurityServiceName and SecurityMainAction.

PortletWidget migration

Start of documentation for all portletwidget migration

1. create/copy the screen in {component}PortletScreen and remove <portalPortlet properties
   ◦ portletTypeI d
   ◦ component, subComponent, useScreen
     i. add screenName and screenLocation
   ◦ useScript, useMenu
2. copy the forms (and update screen if necessary)
3. update screen with portletUiLabelDecorator
4. update screen with screenlet and container if menu need it (id for setArea)
5. update screen with script, if necessary
6. update screen with menu, if necessary
7. copy from controller for uri edit and action
8. in Menu, replace

```
    <show-portlet port-id="\${portalPortletId}" target="EditPersonPt" area-id="\${subAreaDivId}"
        image-location="\${iconsPurpose.Edit}" image-title="\${uiLabelMap.IconsTooltips_Edit}"
        <parameter param-name="partyId" from-field="parameters.partyId"/>
    </show-portlet>
```

with

```
    <link text="Edit" target="editperson" link-type="anchor" target-window="viewPartyInfo"> <!-- usage du target-window pour donner l'id de destination -->
        <parameter param-name="partyId" from-field="parameters.partyId"/>
        <parameter param-name="DONE_PAGE" from-field="DONE_PAGE"/>
        <auto-parameters-portlet/>
        <!--                 TODO is it necessary to have 3 parameters not auto-parameters
        <!--                 <parameter param-name="portalPageId" from-field="parameters.portalPageId"/>
        <!--                 <parameter param-name="portalPortletId" from-field="parameters.portalPortletId"/>
        <!--                 <parameter param-name="portletSeqId" from-field="parameters.portletSeqId"/>
        <image src="\${iconsPurpose.Edit}" title="\${uiLabelMap.IconsTooltips_Edit}"/>
    </link>
```

7.1.5. POC’s use cases

**Operational**

All example management by the two portal page ExampleMgmt et ExampleRecap. On the find, the sub list are not yet manage.

**Next step**

Party component will the next step whith most of its functions

7.1.6. Vue.js
Vue.js general
todo

Components

A Vue.js component is defined by 3 distinct blocks (template, script, style) which are regrouped in a single `.vue` file.

1. Template:
   - Template must be contained in a single root element (div, span, table, etc...)
   - In template we can access computed properties and function with double curly bracket `{{}}`
   - Classic html attributes can be preceded by `:` to use script properties and js code instead of plain text
   - Vue.js give us some directives (v-for, v-if v-on:click, etc...) that are bind to script context and work like a classic attribute preceded by `:` and so can bind to script part and interpret js.

2. Script
   - Script section is wrapped into an `export default {}` that contain the whole script elements. TIP: import have to be made before this export
   - This export is a map that can contain predefined keys used by Vue.js (data, computed, methods, props, created, etc...)
     - `data()` is a function that contain variable used is the component context.
       All variable contained in `data()` are reactive, which mean that Vue.js will track any change on them and will reverberate these changes everywhere there are used to reevaluate render.
     - `computed` is a map that contain evaluation based on reactive data.
       They can be used like a reactive variable in the component and will reevaluate itself when one of its entry had changed.
     - `methods` is a map containing helper function (handleClick, doPost, etc...). WARNING: these functions don’t aim to be reactive.
     - `props` is an array of string who give attribute to the component for passing data from its parent to it.
     - This section can also contain hooks of the component life cycle (created(), mount(), beforeUpdate(), etc...). These block of code will be executed when the hook is fired.
       See below:
       [Component lifecycle]

3. Style
   - This part is dedicated to internal css of the component.
   - Style of this part is applied before other style of the project.

Vuex

Vuex is the centralised state system of the application. It allow us to create 'store' which contain
data that can be accessed/modified by any component of the application.

A store is composed of 4 elements:

1. **State**
   State is a map that contain all stored informations.
   State can only be modify by **mutation**. Default state’s data will be reactive.
   If Data had to be added in the state after its creation, they must be added with the method **Vue.set()** for them to be reactive.

2. **Mutations**
   Mutations is a map that contain function allowed to alter state. By convention, mutation’s key have to be upper-case. Mutations can’t be accessed through code, they must be called by actions.
   Mutations must be synchronous.

3. **Actions**
   Mutation first aim to fire mutation.
   Actions can be asynchronous, in which case there return a promise.

4. **Getters**
   Getters allow to access state data.
   Getters are reactive.
   Getters can use parameters, if they are they return a function instead of a reactive variable.

Store can be split into modules who can be described in an index.js file, stores can so be stored into a modules repository.

**Reactivity**

**todo**

### 7.2. modular and generic UI

Aiming to have an open and modular ERP, it's important that the user interface configuration allow to manage multiple screens from a set of screen "module" for manage **same business object** type in **different business context**.

Next chapters describe a user interface management system’s POC aiming to respond to theses modularity needs.

#### 7.2.1. Use cases for POC-UI

Theses use cases are to be used for new UI POC, documentation associated and selenium unit task test.

All these use cases should be done with existing entities and services, if it's necessary to develop one, simplify the use case, the goal is UI, not service or entity.

These use case description are done in a agile philosophy, only main point is present and during realization details choices will be discuss and done.
Preliminary remarks:

1. In this document, the term "application" corresponding to "plugin component" in the ofbiz terminology which is not same as a "applications/trunk component" in ofbiz terminology. An application is dedicated for a business purpose for a user type, it's build by assembling piece of ofbiz components, sometimes without any specifics entities, services and screen (ex: CRM-B2C, CRM-B2B, SFA are 3 applications uses by sales men).

2. Each use case is on an "application" and is part of one of the menu of this application.

   Of course, this document describe only menu-option needed by the use-case. As it's difficult to do a clear definition of "web-page" because it's depending of theme/template, use case is for me at two level:

   a. screen, which can be well define
   b. page, which depend on default theme

   Of course, some of use case (screen or page) will be done by a previous one (ex: sometime edit is done by the same screen as add). It's even, one of the re-usable important point (UI, Doc and Selenium)

3. Each time a line (or point) start by a "?" the question is "is this point is needed?"

7.2.2. Release and goal

Goal is the POC realization, not doing multiple super applications. POC realization should generate discussion and decision, each time be careful to mainly discuss on UI, Doc or Selenium and not on use case business justification. Use case are to be realistic but mainly as a support for a specifics UI, Doc or Selenium needed.

V1

Main Goal is to proof the technical points. Of course UI re-organization is the driver for this phase, documentation start to work in parallel but will wait first realization to test integration for help. For selenium unit task, it will be a simple usage of the new UI with the default theme, the sole purpose being to check that test with default theme is simple.

So never mind if all cases is not done, the list exist to give different classic case what we will need to know how to process it. Some case which seem to be duplicate from other can be use for beginner who want to help and check if the concept is understanding.

During V1 realization, UseCase list will be updated

V2

Main Goal is to check if the solution is useful for the different type of contributors

- experiment developer
- beginner developer
- functional consultant for parameters
• functional consultant for personalization
• ? end user - for parameters ? (if it’s still possible with the new architecture)

Use Case list will be use as a deployment plan. The list contains similar case to these which are realize in V1, so those on V2 can be achieve by all types of contributors.

**Documentation Goal**

- Apache OFBiz User documentation (asscidoc)
- Apache OFBiz web site wiki
- OFBiz Help

**Selenium test**

1. demo data for each use case and scenario
2. selenium scenario test for each page (or page group)
3. selenium unit test for each screen

**7.2.3. Simple Party Management**

**Which Sub-Application**

1. in HR application, simple employee management
2. in CRM B2C application, simple customer (person) management
3. in eCommerce, simple profile page
4. in HR application, simple organization/place (group) management
5. in CRM B2B application, simple customer (company) management
6. in Facility application, simple worker management

**Which Party sub-component**

1. Party - Person - PartyGroup
2. Contact Mech,
   - with postal address, phone and mail;
   - one or two fixes purpose (ex: phone fix number and mobile phone number)
3. Role
4. Party Identification
5. Party association
6. Not UserLogin because all Security entities should be use and it will generate a too large domain for this POC
Which Screen

1. Party
   - find, list
   - A person
     - add / edit, show
   - A group
     - add / edit, show
   - A company
     - add / edit, show
   - show a synthesis view (party/person/company, contact informations, roles, Identification)
     - Person
     - Company
     - PartyGroup

2. Contact information
   - all contact informations (for one party / facility)
     - with and without purpose
     - with and without history
     - deactivated
   - add / edit postal address
   - add / edit mail
   - add / edit phone

3. Role
   - list for a party
   - add a role (for a parent RoleType)
   - add a role in two step :
     4. select parent RoleType
     5. select the role
        - remove a role

6. Party Identifications
   - list, add, remove

7.2.4. HR Employee management

In HR Component, starting person management with the more complete form about person.
   - Menu option to manage employee
find, list, show, add, edit and manage his
  • contact information
  • identification (3 idTypes, one mandatory, two optionals)
• template page with a header (or sidebar or ...) to show on which employee we are

**Use Case Screen :**

1. find Person
   • simple form (only on party or person)
   • with an add button (which can be show or not depending on parameter or authorization)
2. Person list with an add button (which can be show or not depending on parameter or authorization)
3. add a Person
4. show a Person
5. show a Person with sub-menu with two options : contact informations and Identifications
6. edit a Person
7. List of all contact informations for a person, with an add button (which can be show or not depending on parameter or authorization)
8. add a postal address
9. add a mail
10. add a phone number (to go step by step, without purpose management, will be done in next Use Case group)
11. edit a postal address
12. edit a mail
13. edit a phone number
14. List of all identification number for a person, with an add button (which can be show or not depending on parameter or authorization)
15. add a identification number with choice of identification type
16. edit a identification number with choice of identification type
17. add a identification number with a fix identification type
18. edit a identification number with a fix identification type

**Use Case Page :**

1. create a person
2. search a person
3. visualize a person
4. manage informations about a person
5. template page with a header (or sidebar or ...) to show on which employee we are, (for example to show all his knowledges, or his skills, or his positions, or his ...)

6. manage informations about a person on one page, and with access at this page directly by a field (auto-completion on id, first, last name)

7.2.5. CRM B2C, customer mgnt

In a CRM B2C application, the customer (so, in this context, a person) management. The difference from previous use case group is :

1. person form is more simple than in HR
2. role will be used to characterize customer position (suspect, prospect, with_Quote, customer)

Menu option to manage employee

- find (with role field), list, show, add, edit and manage his
  - contact informations
  - identification (3idTypes, one mandatory, two optionals)
- template page with a header (or sidebar or ...) to show on which customer we are

Use Case Screen :

1. find Person with an add button (which can be show or not depending on parameter or authorization)
   - search field same as in HR find person
   - role field which can appear or not, when not appear a fix value has been put as parameters.
   - contact information field, phone, mail, town. These fields can be show or not by the user with a "deploy" button
2. Person list with an add button (which can be show or not depending on parameter or authorization)
   - role field appear or not, when not appear a fix value has been put as parameters, so only person with this role appear
3. add a Person, all main informations in the form
   - role
   - less field about person than in HR form
     - 1 postal address
     - 2 phone number
     - 1 identification number
4. show a Person, all main informations in the screen with indicator for contact information and identification when there are more data than what it's show.
5. show a Person with sub-menu with options :
   - contact informations
Identifications
role history
change role: a direct action button

6. edit a Person, only "Person" field

7. a button bar to change role (ex: for a suspect, there are the 3 options), this use case is for having a action bar, in this business process case it’s maybe not a need, but for more complex object like order or task, it’s a classical need.

8. List of all contact informations for a person, with one or multiple add buttons (which can be show or not depending on parameter or authorization) and purpose are show, it's the second step, with purpose management.

9. add a postal address (or just a purpose)

10. add a mail

11. add a phone number

12. edit a postal address

13. edit a mail

14. edit a phone number

15. List of all identification number for a person, with an add button (which can be show or not depending on parameter or authorization)

16. add a identification number with choice of identification type

17. edit a identification number with choice of identification type

Use Case Page

1. create a new entry in CRM (role is choose during creation)

2. search a "customer" (or suspect, prospect, ...)

3. visualize a "customer"

4. manage informations about a "customer"

5. template page with a header (or sidebar or ...) to show on which "customer" we are, (for example to show all his quotes, or his orders, or ...)

6. manage informations about a person on one page, and with access at this page directly by a field (auto-completion on id, first, last name)

7.2.6. eCommerce, profile page

A simple profile page.
The difference from previous use case will be mainly on Use Case Page because eCommerce theme could be more original and public user interface should be, most of the time, more simple.

Use Case Screen :

1. show the person, all main informations in the screen with indicator for contact information and
identification when there are more data than what it’s show.

2. show the Person with sub-menu with options:
   - contact informations
   - Identifications

3. edit a Person, only "Person" field

4. List of all contact informations for a person, with an add button and purpose are show, purpose is need for invoice or shipping.

5. add a postal address (or just a purpose)

6. add a mail

7. add a phone number

8. edit a postal address

9. edit a mail

10. edit a phone number

**Use Case Page :**

1. visualize the profile (the person) with edit button

2. manage his contact informations

3. manage his identifications

4. All in one page, which can be look as a long page.

**7.2.7. HR organization mgnt**

In HR component, a simple organization/place (group) management.
Now PartyGroup management (very simple), but with complex screen to manage hierarchy. In this use case group we will use the word "group" for service or department, or subsidiary.

- Menu option to manage the Company organization
  - manage group
  - associated employee in a group
  - manage a hierarchy of group

**Use Case Screen :**

1. find group (with a specific partyType)
   - simple form (only on party or partyGroup)
   - with an add button (which can be show or not depending on parameter or authorization)

2. PartyGroup list with an add button (which can be show or not depending on parameter or authorization)

3. add a group
4. show a Person, all informations in screen with sub-menu with two options : contact informations and Identifications

5. edit a Group

6. List all contact informations for a group, with an add button (which can be show or not depending on parameter or authorization)

7. add a postal address
8. add a phone number
9. edit a postal address
10. edit a phone number

11. List all identification number for a group, with an add button (which can be show or not depending on parameter or authorization)

12. add a identification number with choice of identification type
13. edit a identification number with choice of identification type
14. add a identification number with a fix identification type
15. edit a identification number with a fix identification type

16. List all person associated to the group with two add buttons (which can be, individually, show or not depending on parameter or authorization)
   ◦ add a manager
   ◦ add a member

17. List all group associated to the group (the child) with two add buttons (which can be, individually, show or not depending on parameter or authorization)
   ◦ add an existing group as a child
   ◦ create a new group and add it as a child
   ◦ in the list, each group is a link to this screen, to be able to navigate top-down
   ◦ a third button to go to the parent level, to be able to navigate bottom-up
   ◦ the name of the group manager appear above the list

18. List all parent group for a group or for a person ?

19. show group hierarchy as a tree with action or detail at each level, top-down
20. show group hierarchy as a tree with action or detail at each level, bottom-up

Use Case Page :

1. search a group
2. manage a group
3. manage its contact informations
4. manage hierarchy step by step (parent to child or child to parent)
5. manage hierarchy with a tree view
6. in HR employee, show the tree, top-down or bottom-up with the template "for an employee"

7.2.8. CRM B2B customer mgnt

In a CRM B2B application, the customer (so, in this context, a company) management. For clarification, in these Use Cases, B2B is an other application than B2C. The "CRM B2C & B2B" will be a third, but not in this list because it contains no specificity on screen-page definition

The main difference between B2C is:

1. company versus person,
2. contact management with PartyAssociation
3. ? customer organization management ?

Use Case Screen:

1. find customer (a company (specific partyType)) with an add button (which can be show or not depending on parameter or authorization)
   - search field are on multiple entities with some part deploy or not
   - role field which can appear or not, when not appear a fix value has been put as parameters.
   - contact information field, phone, mail, town. These fields can be show or not by the user with a "deploy" button
2. Company list with an add button (which can be show or not depending on parameter or authorization)
   - role field appear or not, when not appear a fix value has been put as parameters, so only company with this role appear
3. add a Company, all main informations in the form
   - role
   - field from PartyGroup
   - 1 postal address
   - 2 phone number
   - 2 identification number
4. show a Company, all main informations in the screen with indicator for contact informations and identification when there are more data that what it's show.
5. show a Company with sub-menu with options:
   - contact informations
   - Identifications
   - role history
   - change role: a direct action button
6. edit a Company, only "Company" field
7. A button bar to change role (ex: for a suspect, there are the 3 options), this use case is for having an action bar. In this business process case it’s maybe not a need, but for more complex object like order or task, it’s a classical need.

8. List of all contact informations for a company, with an add button (which can be show or not depending on parameter or authorization) and purpose are show, (so, with purpose management).

9. Add a postal address (or just a purpose)

10. Add a mail

11. Add a phone number with purpose

12. Edit a postal address

13. Edit a mail

14. Edit a phone number

15. List of all identification number for a company, with an add button (which can be show or not depending on parameter or authorization)

16. Add a identification number with choice of identification type

17. Edit a identification number with choice of identification type

18. List of contact (person) associated to this company with an add button (which can be show or not depending on parameter or authorization)
   - A contact is a person with contact information
   - List with only one line per contact
   - List of block with contact details for each

19. Edit a contact or his contact information

**Use Case Page:**

Exactly the same as the CRMB2C

1. Create a new entry in CRM (role is choose during creation)

2. Search a "customer" (or suspect, prospect, ...)

3. Visualize a "customer"

4. Manage informations about a "customer"

5. Template page with a header (or sidebar or ...) to show on which "customer" we are, (for example to show all his quotes, or his orders, or ...)

6. Manage informations about a company on one page, and with access at this page directly by a field (auto-completion on id, first, last name).

**7.2.9. Facility worker mgnt**

In Facility application, simple facility’s worker management. For this last use case group, it’s a simplification of the previous one.
Only a very simple and short process for adding people.

It’s the last one, because the goal is to check if it’s easy and rapid to create (or parametrize) a new small application from existing one.

In the Warehouse Management application (simple version OOTB)

- in the administration menu
  - the user menu to manage internal user per facility In the standard business process, it will be used mainly for login and authorization, in our case we will only manage person, his phone number and his facility (where he’s authorized)
  - the facility menu to manage contact informations and person authorized

Use Case Screen :

Already existing screen used

1. find Person
   - simple form (only on party or person)
   - with an add button
2. Person list with an add button
3. add a Person, simple form 3-6 fields
4. show a Person
5. show a Person with sub-menu with option to manage contact informations
6. edit a Person
7. List of all contact informations for a person, with one or multiple add button
8. add a mail
9. add a phone number
10. edit a mail
11. edit a phone number

New Screen

1. add a facility, simple form, if service exist, including some contact informations
2. List of all existing facility
3. List of all contact informations for a facility, with one or multiple add button
4. List of all persons associated to the facility, with two add button
   - add an existing person
   - create a new person and add it to the facility
5. List of all facility associated to a person, with one add button
   - add an existing facility
Use Case Page:

1. manage facilities
2. manage persons
3. visualize a facility details (info, contact informations, persons associated)

7.3. Portlet

Portlet is an autonomous part of the screen, which mean that there can be action that only alter this part of the screen, mainly fired by itself.

Portlet allow a huge modularity gain in user interface, that way an user action (click on a link, type in a field, etc...) must not precise the portlet itself but a logical name which can be subscribed one or more portlet.

This logical name which is subscribed by portlets is called watcherName, this field is in the association table between PortalPage and PortalPortlet.

7.3.1. Portlet update

It's the update of the data stored in js client side store who trigger portlet render update.

7.4. Portal Page

7.5. FrontJs Glossary

watchers

They are objects stored in js store that can be altered remotely. watcher’s subscribers update themselves with the new value when it change.

watcherName

It is the key of a watcher
8. Core APIs
9. Development environment

9.1. Setup your environment

9.1.1. Java SE

9.1.2. IDE

Eclipse

Intellij Idea

9.1.3. Database

9.2. Web tools
10. Testing

10.1. Unit Tests

10.2. Integration Tests
11. Deployment
12. Security

12.1. Passwords and JWT (JSON Web Tokens) usage

12.1.1. How are set and used passwords and JWT in Apache OFBiz

The Apache OFBiz Project Release 17.12

Passwords

Demo and seed passwords are stored in files loaded through security ofbiz-component.xml. To know more about that be sure to read:

- The technical production setup guide notably "Initial Data Loading" and "Security Settings" sections
- How to secure your deployment

These configuration steps are not to be neglected for the security of a production environment

JWT usage

As says Wikipedia:

JSON Web Token (JWT) is an Internet standard for creating JSON-based access tokens that assert some number of claims.

We currently use JWT in 2 places:

1. To let users safely recreate passwords (in backend and frontend)
2. To allow SSO (Single Sign-on) jumpings from an OFBiz instance to another on another domain, by also using CORS (Cross-origin resource sharing) on the target server

How to secure JWT

When you use JWT, in order to sign your tokens, you have the choice of using a sole so called secret key or a pair of public/private keys: https://jwt.io/introduction/.

You might prefer to use pair of public/private keys, for now by default OFBiz uses a simple secret key. Remains the way how to store this secret key. This is an interesting introduction about this question.

1. The first idea which comes to mind is to use a property in the security.properties file. It’s safe as long as your file system is not compromised.
2. You may also pick a SystemProperty entity (overrides the file property). It’s safe as long as your DB is not compromised.
3. We recommend to not use an environment variable as those can be considered weak:
   - http://movingfast.io/articles/environment-variables-considered-harmful
   - https://security.stackexchange.com/questions/49725/is-it-really-secure-to-store-api-keys-in-environment-variables

4. You may want to tie the encryption key to the logged in user. This is used by the password recreation feature. The JWT secret key is salted with a combination of the current logged in user and her/his password. This is a simple and effective safe way.

5. Use a JTI (JWT ID). A JTI prevents a JWT from being replayed. This auth0 blog article get deeper in that. The same is kinda achieved with the password recreation feature. When the user log in after the new password creation, the password has already been changed. So the link (in the sent email) containing the JWT for the creation of the new password can’t be reused.

6. Tie the encryption key to the hardware. You can refer to this Wikipedia page for more information.

7. If you want to get deeper in this get to this OWASP documentation

Note: if you want to use a pair of public/private keys you might want to consider leveraging the Java Key Store that is also used by the "catalina" component to store certificates. Then don't miss to read:


Also remember that like everything a JWT can be attacked and, though not used or tried in OFBiz yet, a good way is to mitigate an attack by using a KeyProvider. I have created OFBiz-11187 for that.

Properties

The security.properties file contains five related properties:

```properties
# -- If false, then no externalLoginKey parameters will be added to cross-webapp urls
security.login.externalLoginKey.enabled=true

# -- Security key used to encrypt and decrypt the autogenerated password in forgot password functionality.
#    Read Passwords and JWT (JSON Web Tokens) usage documentation to choose the way you want to store this key
login.secret_key_string=login.secret_key_string

# -- Time To Live of the token send to the external server in seconds
security.jwt.token.expireTime=1800
```
# -- Enables the internal Single Sign On feature which allows a token based login between OFBiz instances
# -- To make this work you also have to configure a secret key with security.token.key
security.internal.sso.enabled=false

# -- The secret key for the JWT token signature. Read Passwords and JWT (JSON Web Tokens) usage documentation to choose the way you want to store this key
security.token.key=security.token.key

There are also SSO related SystemProperties in SSOJWTDemoData.xml:

```xml
<SystemProperty systemResourceId="security" systemPropertyId="security.internal.sso.enabled" systemPropertyValue="false"/>
<SystemProperty systemResourceId="security" systemPropertyId="security.token.key" systemPropertyValue="security.token.key"/>
<SystemProperty systemResourceId="security" systemPropertyId="SameSiteCookieAttribute" systemPropertyValue="strict"/>
```

**Internal SSO**

The introduction of the same-site attribute set to 'strict' for all cookies prevents the internal Single Sign On feature. Why is clearly explained [here](#).

So same-site attribute set to 'none' is necessary for the internal SSO to work, *lax* is not enough. So if someone wants to use the internal SSO feature s/he also needs to use the CSRF token defense. If s/he wants to be safe from CSRF attacks. Unfortunately, due backporting difficulties, this option is currently (2020-04-15) only available in trunk.

**Fecth API**

An alternative would be to use the Fetch Javascript API with the

```text
credentials: "include"
```

option to enable CORS. Here is an example

For those interested, there are more information in [https://issues.apache.org/jira/browse/OFBIZ-11594](https://issues.apache.org/jira/browse/OFBIZ-11594)

**Last but not least**

Be sure to read *Keeping OFBiz secure*
12.2. Impersonation

12.2.1. What is Impersonation in Apache OFBiz

The Apache OFBiz Project Release 17.12

Introduction to User impersonation

User Impersonation is a feature that offer a way to select a user login and impersonate it, i.e. see what the user could see navigating through the application in his name.

How do this work?

An authorized user (see security and controls section for configuration), can select a user that will be impersonated.

The impersonation start, if everything is well configured, in current application (partymgr for the demo). Everything appears like if we were logged in with the userLoginId and the valid password (though we know nothing about it)

The only thing showing that we currently are impersonating a user is the little bottom-right image:

🧟

This icon indicates, when clicking on it, the user impersonated, and offer a way to depersonate.

The impersonate period is stored for audit purpose, and if the impersonator forgot to depersonate, the period is terminated one hour after impersonation start.

Security

This feature can draw some concerns about security aspect. This paragraph will introduce every controls and properties that have been implemented around the impersonation feature.

These configuration steps are not to be neglected for a production environment since this feature offer a way to act in place of another user.

Properties

The security.properties file introduce two properties that control impersonation feature:

```conf
security.disable.impersonation = true
```

This property, set by default to true, controls the activation of impersonation feature. If no configuration is done any user trying to use impersonation will face an error message, indicating that the feature is disabled.

To enable impersonation this property need to be set to false
This property controls the way impersonation occurred to the impersonated user:

In default configuration, the impersonated user see nothing and can use the application without knowing that he is currently impersonated. Several authorized user can impersonate a same login without any issue.

This configuration is intended for testing/QA environment allowing any authorized user to impersonate a login to validate its configuration, test the application etc.

Set to true, this configuration improve the control of the data generated by the impersonated user. Indeed, Only one authorized user can impersonate a login at the same time, and during the impersonation process, the impersonated user is unable to act within the application.

Since the impersonation period is stored in database, the actions done by the authorized user can be identified if there is the need to do so.

Controls

The permission

First, to be able to use impersonation, a user need to possess IMPERSONATE_ADMIN permissions. Demo data offer IMPERSONATION security group for this purpose. In demo data, FULLADMIN security group also possess the permission.

Permission based user restriction

An authorized user cannot impersonate any user. There are two main controls that will restrict the impersonation feature.

Cannot impersonate Admin user

It is impossible to impersonate a user that is granted any of the admin permission:

```
"IMPERSONATE_ADMIN"
"ARTIFACT_INFO_VIEW"
"SERVICE_MAINT"
"ENTITY_MAINT"
"UTIL_CACHE_VIEW"
"UTIL_DEBUG_VIEW"
```

Cannot impersonate more privileged user

It is impossible to impersonate a user that has more permission than your user. Even if the missing permission is a minor one.
12.3. CSRF defense

12.3.1. How is done the CSRF defense in Apache OFBiz and how to adapt it if needed

The Apache OFBiz Project Release 17.12

The same-Site attribute

The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.

— According to OWASP ZAP

By default OOTB the SameSiteFilter property sets the same-site attribute value to 'strict'. SameSiteFilter allows to change to 'lax' if needed. If you use 'lax' we recommend that you set the csrf.defense.strategy property to org.apache.ofbiz.security.CsrfDefenseStrategy in order to provide an effective defense against CSRF attacks.

Properties

The `security.properties` file contains related properties:

```properties
# -- By default the SameSite value in SameSiteFilter is 'strict'.
# -- This property allows to change to 'lax' if needed.
# -- If you use 'lax' we recommend that you set
# -- org.apache.ofbiz.security.CsrfDefenseStrategy
# -- for csrf.defense.strategy (see below)
SameSiteCookieAttribute=

# -- The cache size for the Tokens Maps that stores the CSRF tokens.
# -- RemoveEldestEntry is used when it's get above csrf.cache.size
# -- Default is 5000
# -- TODO: possibly separate tokenMap size from partyTokenMap size
csrf.cache.size=

crf.tokenName.nonAjax=
```
# -- The csrf.entity.request.limit is used to show how to avoid cluttering the Tokens Maps cache with URIs starting with "entity/
# -- It can be useful with large Database contents, ie with a large numbers of tuples, like "entity/edit/Agreement/10000, etc.
# -- The same principle can be extended to other cases similar to "entity/" URIs (harcoded or using similar properties).
# -- Default is 3
csrpf.entity.request.limit=

# -- CSRF defense strategy.
# -- Because OFBiz OOTB also sets the SameSite attribute to 'strict' for all cookies,
# -- which is an effective CSRF defense,
# -- default is org.apache.ofbiz.security.NoCsrfDefenseStrategy if not specified.
# -- Use org.apache.ofbiz.security.CsrfDefenseStrategy
# -- if you need to use a 'lax' for SameSiteCookieAttribute
csrpf.defense.strategy=

There is also a SystemProperty in SSOJWTDemoData.xml:

```xml
<SystemProperty systemResourceId="security" systemPropertyId="SameSiteCookieAttribute"
    systemPropertyValue="strict"/>
```
13. Appendices
14. From Mini Language to Groovy

This is a small guide for everybody involved in converting the Mini Language into Groovy.

**Why is this important?**
This tutorial is directly linked to the efforts of converting all scripts in Mini Language to newer Groovy Scripts. All of this is done, because Groovy is much more readable and easier to review, more up to date and many other reasons, which can be found here: [Proposal for deprecating Mini Language](#)

To contribute, or just be up to date with the current process, you can look at the existing [JIRA issue OFBIZ-9350 - Deprecate Mini Lang](#)

14.1. Groovy DSL (dynamic scripting library)

14.1.1. How to get Groovy support in your IDE

The following paragraph is for Eclipse users.

It is possible to get Groovy support in Eclipse by converting the loaded project to a Groovy Project. The project itself will work as before.

To do this just follow these few steps:

1. Right-click on the project that has to be converted
2. Click on "Configure"
3. Click on "Convert to Groovy Project"

Eclipse will automatically load the file OfbizDsldescriptorforeclipse.dslid, in which the known fields and methods used in Groovy Scripts are defined.

14.1.2. Known Fields

```java
property name: 'parameters'
type: 'java.util.Map'
```

These are the parameters given to the Groovy Script, when it is called as a service. It is equivalent to `Map<String, Object>` context in the Java-Service-Definition.

```java
property name: 'context'
type: 'java.util.Map'
```

More parameters, which are, for example, given through a screen or another Groovy Script. This is important when the script is called through an action segment of a screen.

```java
property name: 'delegator'
type: 'org.apache.ofbiz.entity.Delegator'
```

Normal instance of the Delegator, which is used for special database access.
property name: 'dispatcher'
type: 'org.apache.ofbiz.service.LocalDispatcher'
Normal instance of the LocalDispatcher, which is used to call services and other service-like operations.

property name: 'security'
type: 'org.apache.ofbiz.security.Security'
Normal instance of the Security-Interface with which permission checks are done.

### 14.2. Known Methods

**method name: 'runService'**
type: 'java.util.Map'
params: [serviceName: 'String', inputMap: 'java.util.Map']
Helping method to call services instead of dispatcher.runSync(serviceName, inputMap). Also possible: run service: serviceName, with: inputMap

**method name: 'makeValue'**
type: 'java.util.Map'
params: [entityName: 'String']
Helping method to make a GenericValue instead of delegator.makeValue(entityName). Creates an empty GenericValue of the specific entity.

**method name: 'findOne'**
type: 'java.util.Map'
params: [entityName: 'String', inputMap: 'java.util.Map']
Helping method to find one GenericValue in the database. Used instead of delegator.findOne(entityName, inputMap)

**method name: 'findList'**
type: 'java.util.List'
params: [entityName: 'String', inputMap: 'java.util.Map']
Helping method to find many GenericValue in the database. Used instead of delegator.findList(entityName, inputMap, null, null, null, false)

**method name: 'select'**
type: 'org.apache.ofbiz.entity.util.EntityQuery'
params: [entity: 'java.util.Set']
Helping method used instead of EntityQuery.use(delegator).select(…)

As above.

**method name: 'from'**
type: 'org.apache.ofbiz.entity.util.EntityQuery'
params: [entity: 'java.lang.Object']
Helping method used instead of EntityQuery.use(delegator).from(…)
method name: 'success'
type: 'def'
params: [message: 'String']
Helping method used instead of ServiceUtil.returnSuccess(message)

method name: 'failure'
type: 'java.util.Map'
params: [message: 'String']
Helping method used instead of ServiceUtil.returnFailure(message)

method name: 'error'
type: 'def'
params: [message: 'String']
Helping method used instead of ServiceUtil.returnError(message)

method name: 'logInfo'
type: 'void'
params: [message: 'String']
Helping method used instead of Debug.logInfo(message, fileName)

method name: 'logWarning'
type: 'void'
params: [message: 'String']
Helping method used instead of Debug.logWarning(message, fileName)

method name: 'logError'
type: 'void'
params: [message: 'String']
Helping method used instead of Debug.logError(message, fileName)

method name: 'logVerbose'
type: 'void'
params: [message: 'String']
Helping method used instead of Debug.logVerbose(message, fileName)

The actual definition of the methods can be found in '/framework/service/src/main/java/org/apache/ofbiz/service/engine/GroovyBaseScript.groovy, the
variables dctx, dispatcher and delegator are set in the file GroovyEngine.java which can be found in the
same location.

14.3. Services

14.3.1. From MiniLang to Groovy

To see additional examples and finished conversions, which may help with occurring questions, click: OFBiz-9350 - Deprecate Mini Lang There is a chance that a similar case has already been
converted.

⚠️ When a simple-method ends, it will automatically at least return a success-map.
All the Groovy Services have to return success at least, too.

```
return success()
```

### 14.3.2. Getting started

MiniLang files consist of services, which, in most cases, implement services.

The get converted to Groovy like the following:

```
<!-- This is MiniLang -->
<simple-method method-name="createProductCategory" short-description="Create an ProductCategory">
  <!-- Code -->
</simple-method>

// This is the converted Groovy equivalent
/**
 * Create an ProductCategory
 */
def createProductCategory() {
  // Code
}
```

It will be useful for future developers, and everybody who has to check something in the code, to put at least the short-description as the new Groovydoc. This will hopefully more or less explain, what the method should or shouldn’t do. If the short-description isn’t helpful enough, feel free complete it.

The structure of if and else in MiniLang is a little different than the one from Groovy or Java and can be a bit confusing when first seen, so here is an example:

```
<if-empty field="parameters.productCategoryId">
  <sequenced-id sequence-name="ProductCategory" field="newEntity.productCategoryId"/>
<else>
  <set field="newEntity.productCategoryId" from-field="parameters.productCategoryId"/>
  <check-id field="newEntity.productCategoryId"/>
  <check-errors/>
</else>
</if-empty>
```

Notice, that the else always starts before the if-tag is closed, but sometimes isn’t indented as one would expect it.
When navigating through bigger **if**-phrases, the navigation itself will be much easier through just clicking in the opening or closing **if**-tag: Eclipse will automatically mark the matching opening or closing **if**-tag for you.

There are two possibilities to initialize a field/variable in Groovy.

1. To define a field/variable with its correct typing
   
   ```java
   String fieldName = "value"
   ```

2. To just “define” a field/variable. The IDE you are working with may not recognize the typing, but OFBiz can work with it:
   
   ```java
   def fieldName = "value"
   ```

### 14.4. Checking Fields

<table>
<thead>
<tr>
<th>Minilang</th>
<th>Groovy</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;if-empty field=&quot;fieldName&quot;&gt;&lt;/if-empty&gt;</code></td>
<td>// checks if fieldName is existent and/or empty</td>
</tr>
<tr>
<td></td>
<td><code>if (!fieldName) {}</code></td>
</tr>
<tr>
<td>`&lt;if-empty field=&quot;fieldName.property&quot;&gt;</td>
<td>// fieldName has to be existent, property doesn't need to</td>
</tr>
<tr>
<td></td>
<td>// if known, that property does exist, the ? can be left out</td>
</tr>
<tr>
<td></td>
<td><code>if (!fieldName?.property) {}</code></td>
</tr>
<tr>
<td></td>
<td>// CAUTION: every query like this in Groovy evaluates to a Boolean type</td>
</tr>
<tr>
<td></td>
<td>// everything that is empty or false will turn into false</td>
</tr>
<tr>
<td></td>
<td>// null, [], [:], &quot;,&quot;, false -&gt; false</td>
</tr>
<tr>
<td></td>
<td><code>if (UtilValidate.isEmpty(fieldName)) {}</code></td>
</tr>
<tr>
<td>Minilang</td>
<td>Groovy</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
</tr>
</tbody>
</table>
|<if>
   <condition>
     <or>
       <if-empty field="field1"/>
       <if-empty field="field2"/>
     </or>
   </condition>
   <then>
     <!-- code in if -->
   </then>
   <else>
     <!-- code in else -->
   </else>
</if> | if (!field1 || !field2) {
   // code in if
} else {
   // code in else
} |

<table>
<thead>
<tr>
<th>Minilang</th>
<th>Groovy</th>
</tr>
</thead>
</table>
|<if-compare-field
    field="product.primaryProductCategoryId"
    to-field="parameters.productCategoryId"
    operator="equals"
> <!-- code --></if-compare-field> | // this will even work, if product is not existent or null
if (UtilValidate.areEqual(product?.primaryProductCategoryId, parameters .productCategoryId)) {
   // code
} |

<table>
<thead>
<tr>
<th>Minilang</th>
<th>Groovy</th>
</tr>
</thead>
</table>
|<if-instance-of
    field="parameters.categories"
    class="java.util.List"></if-instance-of> | if (parameters.categories instanceof java.util.List) {} |

### 14.5. Setting Fields

<table>
<thead>
<tr>
<th>Minilang</th>
<th>Groovy</th>
</tr>
</thead>
</table>
|<set field="fieldName" value="value"/> | // if fieldName is not initialized
String fieldName = "value"
// if fieldName is initialized
fieldName = "value" |
Minilang

```xml
<set field="otherFieldName.property"
value="value"/>
<set field="otherFieldName.otherProperty"
value="true" type="Boolean"/>
<set field="otherFieldName.otherProperty"
from-field="parameters.property"/>
```

// if otherFieldName is not yet initialized, you have to do it first
// MiniLang does that automatically
Map otherFieldName = [:]; // empty Map
// now put the values in
otherFieldName = [
  property: "value",
  otherProperty: true
];
// or the less efficient way
otherFieldName.property = "value"
otherFieldName.otherProperty = true

// it is possible to put different values in later:
otherFieldName.property = parameters.property

Groovy

```java
<set field="thisFieldName"
value="${groovy: []}" type="List"/>
```

// this is easier in Groovy
List thisFieldName = []

```java
String failMessage = UtilProperties.getMessage("CommonUiLabels", "CommonGenericPermissionError", parameters.locale);
// in Groovy there can be a difference for the second case
parameters.rateCurrencyUomId = UtilProperties.getPropertyValue('general.properties', 'currency.uom.id.default')
```

```java
<clear-field
field="product.primaryProductCategoryId"/>
```

product.primaryProductCategoryId = null

14.6. Starting Services
14.7. Preparing Service Results

```minilang
<set>
  field="relatedCategoryContext.parentProductCategoryId" from-
  field="defaultTopCategoryId"/>
<call-service service-
  name="getRelatedCategories" in-map-
  name="relatedCategoryContext">
  <result-to-field result-
    name="categories" field="resCategories"/>
</call-service>
</set-service-fields>
```

```groovy
def relatedCategoryContext = [
  parentProductCategoryId: defaultTopCategoryId]
def serviceResult = run service: 
  "getRelatedCategories", with: 
  relatedCategoryContext
def resCategories = serviceResult.
  categories
  // if it is not too confusing to read you can leave out the extra variable
run service: "getRelatedCategories", 
  with: [parentProductCategoryId: 
  defaultTopCategoryId]
```

14.8. Database Communication

```minilang
<field-to-result field="fieldBudgetId" result-name="budgetId"/>
```

```groovy
// MiniLang knows this implicitly
def result = success()
result.budgetId = fieldBudgetId
return result
```
### Minilang

```xml
<make-value entity-name="FinAccountTrans" value-field="newEntity"/>
<set-nonpk-fields map="parameters" value-field="newEntity"/>
<set-pk-fields map="parameters" value-field="newEntity"/>
```

### Groovy

```groovy
// this is the easy way
GenericValue newEntity = makeValue("FinAccountTrans", parameters)
// this is also possible
GenericValue newEntity = makeValue("FinAccountTrans")
newEntity.setPKFields(parameters)
newEntity.setNonPKFields(parameters)
```

```xml
<entity-and entity-name="BudgetStatus" list="budgetStatuses">
  <field-map field-name="budgetId" from-field="parameters.budgetId"/>
  <order-by field-name="-statusDate"/>
</entity-and>
```

### Groovy

```groovy
// this can also be done in one line, but it can easily become unreadable
def budgetStatuses = from("BudgetStatus")
  .where("budgetId", parameters.budgetId)
  .orderBy("-statusDate")
  .queryList()
```

```xml
<entity-one entity-name="StatusValidChange" value-field="statusValidChange">
  <field-map field-name="statusId" from-field="budgetStatus.statusId"/>
  <field-map field-name="statusIdTo" from-field="parameters.statusId"/>
</entity-one>
```

### Groovy

```groovy
// MiniLang has false set for useCache as the default value
statusValidChange = findOne("StatusValidChange", [statusId: budgetStatus.statusId, statusIdTo: parameters.statusId], false)
// this is also possible
statusValidChange = from("StatusValidChange")
  .where("statusId", budgetStatus.statusId, "statusIdTo", parameters.statusId)
  .queryOne()
```

```xml
<!-- entity-one can be called without child elements, too -->
<entity-one entity-name="Product" value-field="product" auto-field-map="true"/>
```

### Groovy

```groovy
// if there are no child elements, this can be used
GenericValue product = from("Product")
  .where(parameters).
```
### Minilang

```xml
<find-by-primary-key entity-name="ProductCategoryMember"
  map="lookupPKMap" value-field="lookedUpValue"/>
```

### Groovy

```groovy
GenericValue lookedUpValue = findOne
("ProductCategoryMember", lookupPKMap, false)
// this is also possible
lookedUpValue = from
("ProductCategoryRole")
  .where(lookupPKMap)
  .queryOne()
```

### Minilang

```xml
<entity-condition entity-name="ProductCategoryContentAndInfo"
  list="productCategoryContentAndInfoList"
  filter-by-date="true" use-cache="true">
  <condition-list combine="and">
    <condition-expr field-name="productCategoryList.productCategoryId"
                     from-field="productCategoryList.productCategoryId"/>
    <condition-expr field-name="prodCatContentTypeId"
                     value="ALTERNATIVE_URL"/>
  </condition-list>
  <order-by field-name="-fromDate"/>
</entity-condition>
<!-- entity-condition can also be used with the "or" operator -->

```xml
<entity-condition entity-name="ProdCatalogCategory"
  list="prodCatalogCategoryList" filter-by-date="true">
  <condition-list combine="and">
    <condition-expr field-name="parameters.productCategoryId"
                     from-field="parameters.productCategoryId"/>
    <condition-list combine="or">
      <condition-expr field-name="prodCatalogCategoryTypeId"
                       value="PCCT_VIEW_ALLW"/>
      <condition-expr field-name="prodCatalogCategoryTypeId"
                       value="PCCT_PURCH_ALLW"/>
    </condition-list>
  </condition-list>
</entity-condition>
```

### Groovy

```groovy
// the Groovy methods use the "and" and "equals" operator as default values
List productCategoryContentAndInfoList = from("ProductCategoryContentAndInfo")
  .where("productCategoryContentAndInfo")
  .filterByDate()
  .queryList()
// with the use of the "or" operator
you have to build your condition like this
EntityCondition condition =
EntityCondition.makeCondition([
  EntityCondition.makeCondition([
    EntityCondition.makeCondition("prodCatalogCategoryTypeId", "PCCT_VIEW_ALLW"),
    EntityCondition.makeCondition("prodCatalogCategoryTypeId", "PCCT_PURCH_ALLW"),
  ], EntityOperator.OR),
  EntityCondition.makeCondition("productCategoryId", parameters.productCategoryId)
])
List prodCatalogCategoryList = from("ProdCatalogCategory")
  .where(condition)
  .filterByDate().queryList()```

83
<make-value entity-name="FinAccountTrans" value-field="newEntity"/>
<set-nonpk-fields map="parameters" value-field="newEntity"/>
<!-- In this case multiple fields of the GenericValue are set -->
<make-value entity-name="ProductCategoryRollup" value-field="newLimitRollup"/>
<set field="newLimitRollup.productCategoryId" from-field="newEntity.productCategoryId"/>
<set field="newLimitRollup.parentProductCategoryId" from-field="productCategoryRole.productCategoryId"/>
<set field="newLimitRollup.fromDate" from-field="nowTimestamp"/>

<set field="statusValidChange.prop" value="value"/>

<create-value value-field="newEntity"/>

<store-value value-field="newEntity"/>
<store-list list="listToStore"/>

<clone-value value-field="productCategoryMember" new-value-field="newProductCategoryMember"/>

<remove-value value-field="lookedUpValue"/>
### 14.9. Permissions

To also check for admin-permissions, this method has to be used:

```groovy
hasEntityPermission(permission, action, userLogin)
```

If the method is used with wildcards, it is important to **not forget the underscore**, which comes before the parameter action!

```groovy
if (!security.hasEntityPermission("CATALOG", "_CREATE", parameters.userLogin)
  || security.hasEntityPermission("CATALOG_ROLE", "_CREATE", parameters.userLogin)) {
  return error(UtilProperties.getMessage("ProductUiLabels", "ProductCatalogCreatePermissionError", parameters.locale))
}
```
14.10. Timestamp And System Time

The first two simple-method are deprecated; the third method should have been used instead.

```
<set field="hasCreatePermission" value="false" type="Boolean"/>
<if-has-permission permission="${primaryPermission}" action="${mainAction}"
    <set field="hasCreatePermission" value="true" type="Boolean"/>
</if-has-permission>
```

```
// this will automatically be set to false if the user doesn't have the permission
def hasCreatePermission = security .hasEntityPermission (primaryPermission, "_${mainAction}", parameters.userLogin)
```

```
<now-timestamp field="nowTimestamp"/>
```

```
Timestamp nowTimestamp = UtilDateTime.nowTimestamp()
```

```
<now-date-to-env field="nowDate"/>
```

```
Timestamp nowDate = UtilDateTime.nowTimestamp()
```

```
<!-- this method also has the parameter "type", which is set to 'java.sql.timestamp' as default -->
<now field="fooNow"/>
```

```
Timestamp fooNow = UtilDateTime.nowTimestamp()
```

```
<if-compare-field field="productCategoryMember.thruDate" to-field="expireTimestamp" operator="less" type="Timestamp">
    <!-- code -->
</if-compare-field>
```

```
Timestamp thruDate = productCategoryMember.thruDate
if (thruDate && thruDate.before (expireTimestamp)) {
    // code
}
```

14.11. Logging

Since all of the log methods are know to the Groovy Language, it is possible to just nearly use them as they are in MiniLang.
For further explanation, here are some examples:
### 14.12. General

<table>
<thead>
<tr>
<th>Minilang</th>
<th>Groovy</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;log level=&quot;verbose&quot; message=&quot;Permission check failed, user does not have permission&quot;/&gt;</td>
<td>logVerbose(&quot;Permission check failed, user does not have the correct permission.&quot;)</td>
</tr>
<tr>
<td>&lt;log level=&quot;info&quot; message=&quot;Applying feature \[\{productFeatureId\}\] of type \[\{productFeatureTypeId\}\] to product \[\{productId\}\]&quot;/&gt;</td>
<td>logInfo(&quot;Applying feature [{productFeatureId}] of type [{productFeatureTypeId}] to product [{productId}]&quot;)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Minilang</th>
<th>Groovy</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;call-simple-method method-name=&quot;checkCategoryRelatedPermission&quot;/&gt; &lt;check-errors/&gt;</td>
<td>// simple-methods inside of classes, as long as they are not services, will be called like normal methods Map res = checkCategoryRelatedPermission(&quot;updateProductCategory&quot;, &quot;UPDATE&quot;, null, null) if (!ServiceUtil.isSuccess(res)) { return res }</td>
</tr>
<tr>
<td>&lt;iterate list=&quot;subCategories&quot; entry=&quot;subCategory&quot;&gt; &lt;!-- code --&gt; &lt;/iterate&gt;</td>
<td>for (def subCategory : subCategories) { // code } subCategories.each { subCategory -&gt; // code }</td>
</tr>
</tbody>
</table>
<iterate-map
map="parameters.productFeatureIdByType"
key="productFeatureTypeId"
value="productFeatureId">
  <!-- in here something should happen with value and key -->
</iterate-map>

for (Map entry : parameters.productFeatureIdByType.entrySet()) {
  def productFeatureTypeId = entry.getKey()
  def productFeatureId = entry.getValue()
  // in here something should happen with value and key
}

if (!security.hasEntityPermission("CATALOG", "_${checkAction}",
parameters.userLogin)) {
  // code
}

def query = from("ProductCategoryMember").where("productCategoryId",
parameters.productCategoryId)
if (parameters.validDate) {
  query.filterByDate()
}
List productCategoryMembers = query.queryList()
### 14.13. Where to find MiniLang implementation

If you find yourself in a position, where you don't know how to convert a certain tag from MiniLang to Groovy, you can always check the Java implementation of the MiniLang method. All of the methods have an existing Java implementation and you can find all of them in this folder: `/ofbiz/trunk/framework/minilang/src/main/java/org/apache/ofbiz/minilang/method`

The interesting part of this implementation is the method `exec()`, which actually runs the MiniLang tag. The tag `<remove-by-and>` for example is realized using this part of code here:

```java
@Override
public boolean exec(MethodContext methodContext) throws MiniLangException {
    @Deprecated
    String entityName = entityNameFse.expandString(methodContext.getEnvMap());
    if (entityName.isEmpty()) {
        throw new MiniLangRuntimeException("Entity name not found.", this);
    }
    try {
        Delegator delegator = getDelegator(methodContext);
        delegator.removeByAnd(entityName, mapFma.get(methodContext.getEnvMap()));
    } catch (GenericEntityException e) {
        String errMsg = "Exception thrown while removing entities: " + e.getMessage();
        Debug.logWarning(e, errMsg, module);
        simpleMethod.addErrorMessage(methodContext, errMsg);
        return false;
    }
    return true;
}
```

In this you can find one important part of code, which is:

```java
dealetor.removeByAnd(entityName, mapFma.get(methodContext.getEnvMap()));
```

This tells you, that, if you're trying to convert the tag `<remove-by-and>`, you can use `delegator.removeByAnd()` in Groovy.