

ИнтерТраст

Архитектура GUI Business Universe

Митавский Д. В.

Содержание

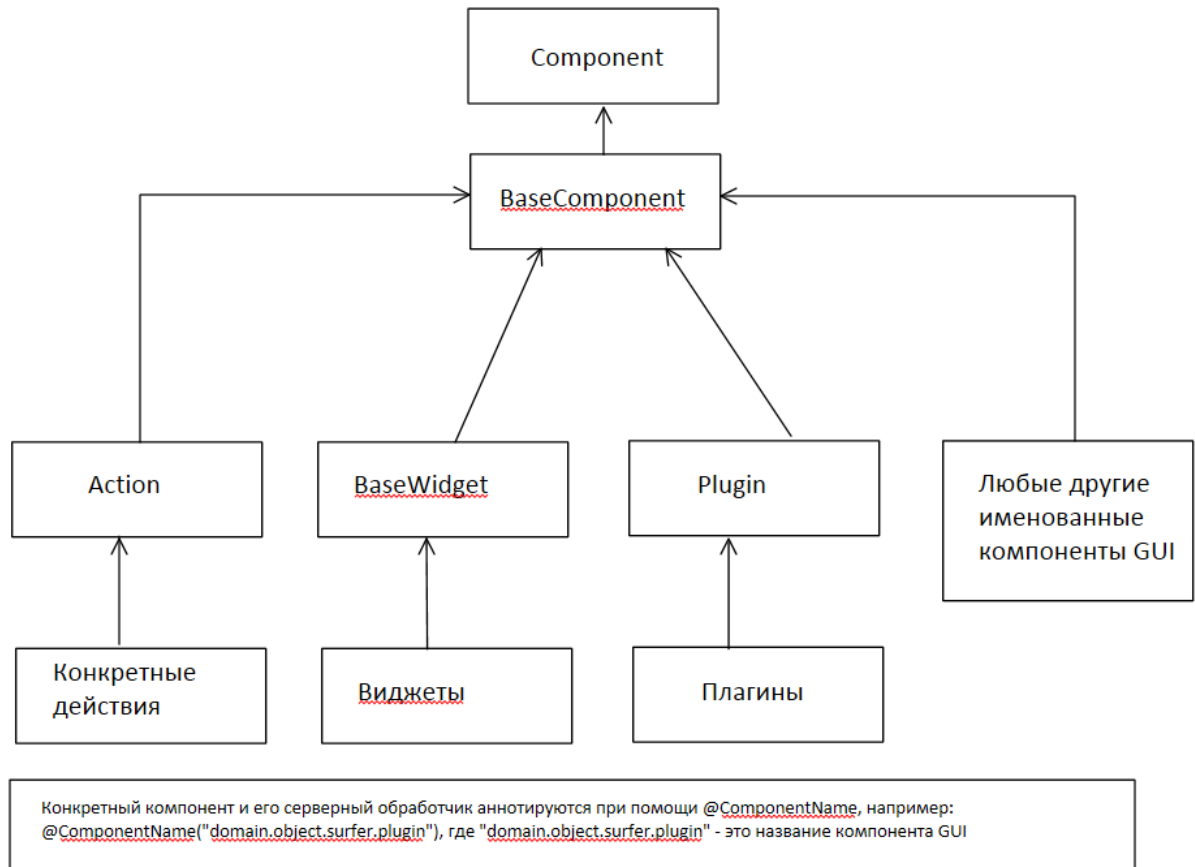
Архитектура GUI.....	3
Компоненты GUI	4
Плагины	6
Business Universe – разметка	7
Глобальная шина сообщений.....	8
Плагин «Domain Object Surfer».....	9
Локальная шина сообщений	10
Формы и виджеты	11
Путь к полю доменного объекта	11
Определение формы	14
Разметка формы	14
Описание виджетов, составляющих форму	22
Общие атрибуты для всех виджетов системы	22
Метка (Label)	22
Текстовое поле (Text Box)	24
Текстовая область (Text Area)	24
Целочисленное поле (Integer Box)	25
Десятичное поле (Decimal Box)	26
Поле даты/времени (Date Box).....	26
Список фиксированных значений (Enumeration Box).....	27
Выпадающий список (Combo Box)	28
Список значений (List Box)	29
Радио-кнопка (Radio Button).....	30
Флажок (Check Box)	30
Табличный обозреватель (Table Browser)	32
Обозреватель иерархии (Hierarchy Browser)	36
Выпадающий список с подсказками (Suggest Box).....	41
Набор вложений (Attachment Box)	44
Таблица связанных доменных объектов (Linked Domain Objects Table)	45
Редактируемая таблица связанных доменных объектов (Linked Domain Objects Editable Table)	47
Гиперссылка на доменный объект (Linked Domain Object Hyperlink)	48
Определение форм для ролей и конечных пользователей	49
Конфигурация	50
Панель навигации.....	50

Архитектура GUI

Документ описывает основные понятия, принципы построения и архитектуру веб-приложения «Business Universe». Приложение построено по принципам расширяемости и заменяемости (pluggability) и обладает очень широкими возможностями по конфигурируемости. Все основные свойства архитектуры останутся практически неизменными при реализации новой версии пользовательского интерфейса, таким образом, применение новых дизайнов к системе существенно упрощается.

Компоненты GUI

Пользовательский интерфейс построен при помощи именованных компонентов. Это обеспечивает возможность замены одного компонента другим и возможность ссылаться на компоненты из конфигурации, не затрагивая при этом исходный код. Любой компонент должен реализовывать интерфейс Component. Для удобства разработчиков и возложения некоторых утилитарных функций на архитектуру существует абстрактный класс BaseComponent (реализующий Component), наследниками которого все компоненты системы и являются.



Если компонент использует взаимодействие с серверной частью системы, то для него должен быть создан соответствующий обработчик на стороне сервера, реализующий интерфейс ComponentHandler. Интерфейс маркерный и обязательных к реализации методов не содержит. Серверный обработчик и клиентский класс компонента должны быть аннотированы следующим образом:

```
@ComponentName("domain.object.surfer.plugin")
```

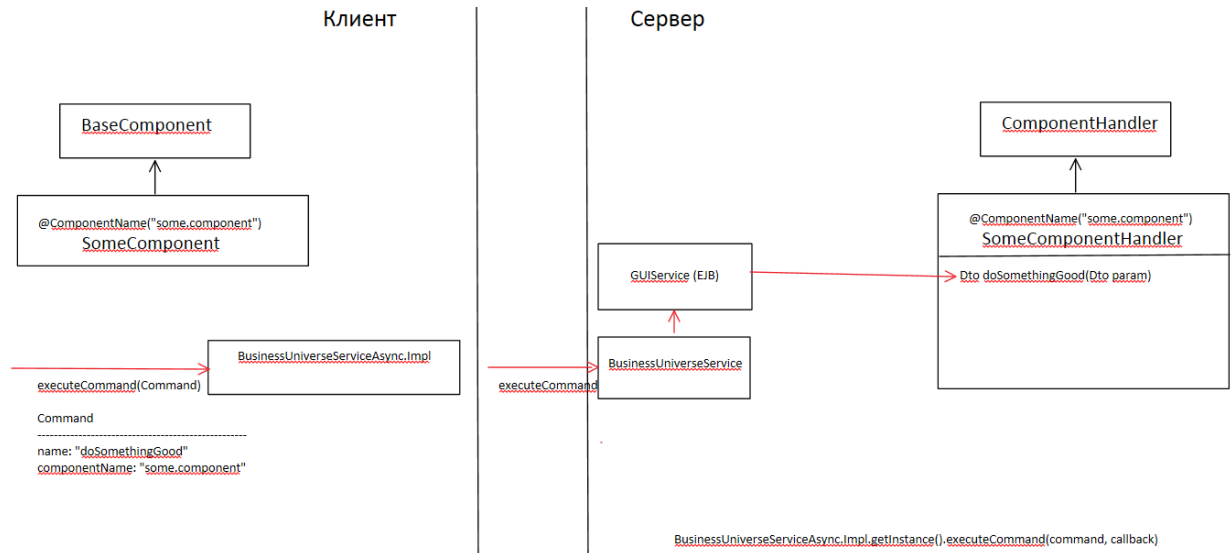
где в кавычках фигурирует название компонентов.

Схема взаимодействия серверной и клиентской частей компонента представлена на рисунке ниже.

Для выполнения команды достаточно получить экземпляр класса BusinessUniverseServiceAsyncImpl и вызвать асинхронный метод executeCommand с параметром, описывающим команду. Описание команды содержит :

- 1) Название компонента
- 2) Название команды. Оно совпадает с названием метода на стороне сервера в реализации ComponentHandler
- 3) Параметр команды (любой Dto)

Серверные команды также возвращают Dto, который доступен в методе обратного вызова (callback) асинхронного запроса.



Получить экземпляр именованного компонента пользовательского интерфейса можно используя так называемый «Реестр компонентов»: `ComponentRegistry.get()`.

Реализацию любого именованного компонента можно заменить. Для этого необходимо альтернативную реализацию компонента снабдить аннотацией, определяющей то же имя, как у компонента, который требуется заместить, и в файле `gui.properties` определить соответствие между названием компонента и классом, который будет при этом использоваться. Например, для того, чтобы использовать «окно логина для разработчиков», которое автоматически заполняется логином и паролем и авторизуется, достаточно поменять реализацию компонента «login.window»:

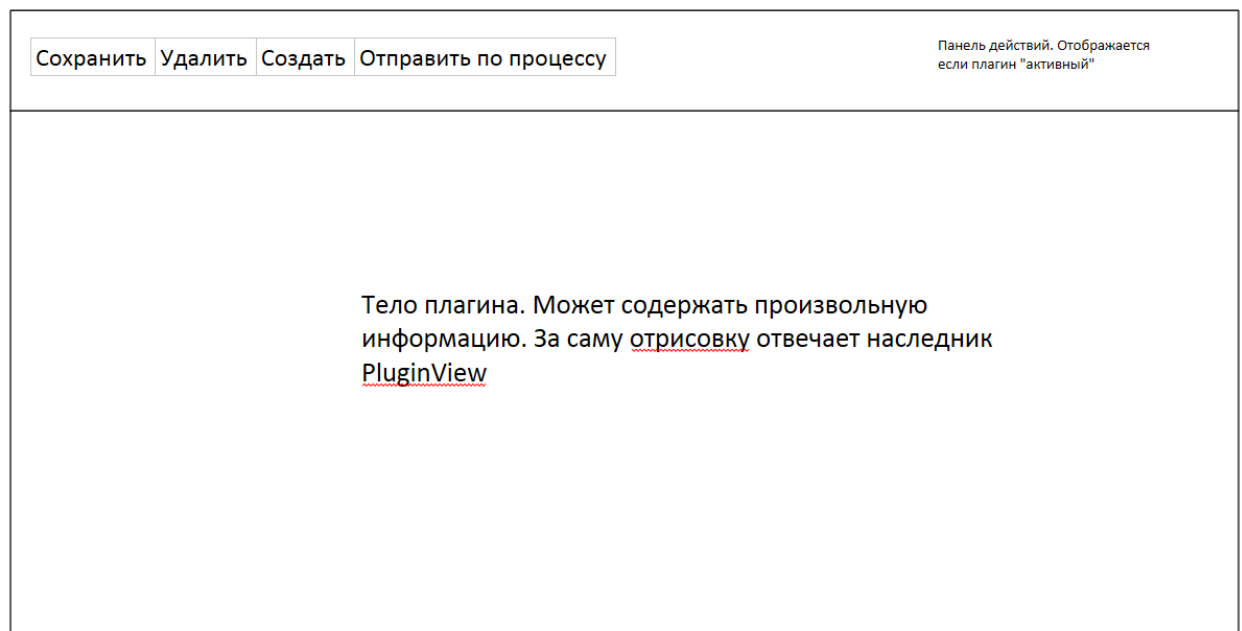
```
login.window.a = ru.intertrust.cm.core.gui.impl.authentication.tempdev.DevelopmentLoginWindow
```

Плагины

Плагин – это некоторая, часто, самостоятельная часть пользовательского интерфейса, которую можно отобразить в тех частях экрана, в которых это предусмотрено (в этих частях должна быть расположена панель плагина, которую реализует класс `PluginPanel`).

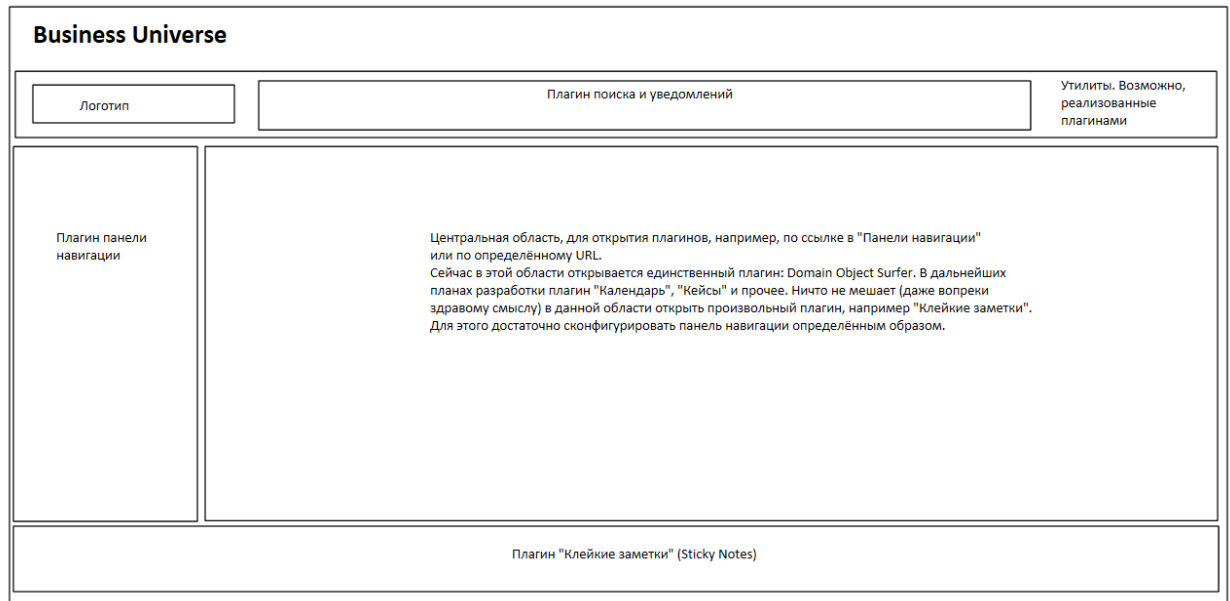
В качестве примера плагинов можно привести: Панель навигации, Форма, Коллекция, Календарь. Плагины являются именованными компонентами.

Плагин может быть как «активным» так и «неактивным». «Активный» плагин позволяет пользователю выполнять действия, расположенные в панели действий в верхней части плагина. Клиентский класс «активного» плагина должен реализовывать интерфейс `IsActive`, а серверный обработчик должен быть наследником `ActivePluginHandler`. Структура плагина показана на следующем рисунке.



Конфигурация действий, отображаемых «активным» плагином, формируется на стороне сервера при инициализации плагина в методе `ActivePluginHandler.initialize()`. Чтобы открыть плагин в GUI, необходимо создать его экземпляр, назначить конфигурацию и вызвать метод `PluginPanel.open(Plugin plugin)`.

Business Universe – разметка

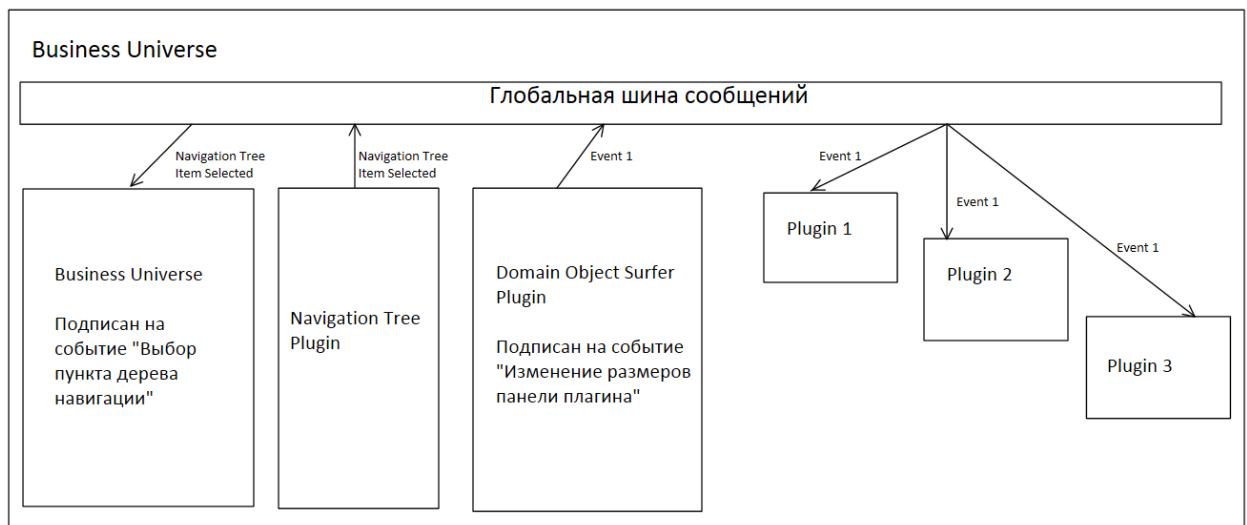


Business Universe состоит из следующих частей, показанных на рисунке.

1. Верхняя панель, которая в себя включает:
 - a. Логотип
 - b. Плагин поиска и уведомлений
 - c. Утилитарные области (настройки, «Выход», информация о пользователе. На данный момент они реализованы классическим способом, впоследствии, вполне вероятно, что они будут трансформированы в плагины (например, для того, чтобы в другом варианте пользовательского интерфейса иметь возможность быть размещёнными в произвольной (настраиваемой) области экрана.
2. Плагин панели навигации
3. Центральная область плагина, открываемого по требованию (например, по ссылке в "Панели навигации" или по определённому URL). Сейчас в этой области открывается единственный плагин: Domain Object Surfer. В дальнейших планах разработки плагины "Календарь", "Кейсы" и другие. Ничто не мешает (даже вопреки здравому смыслу) в данной области открыть произвольный плагин, например "Клейкие заметки". Для этого достаточно сконфигурировать панель навигации определённым образом.
4. Нижняя область, предназначенная в данной реализации исключительно для плагина «Клейкие заметки».

Глобальная шина сообщений

Для обмена сообщениями между компонентами в BusinessUniverse используется стандартный механизм GWT – Event Bus.



Глобальная шина сообщений доступна абсолютно всем компонентам. Получить её экземпляр можно следующим образом:

```
Application.getInstance().getEventBus();
```

В данный момент глобальная шина используется только плагинами, которые могут подписываться на интересующие сообщения. Для подписки на глобальные сообщения достаточно переопределить абстрактный метод `Plugin.getEventTypesToHandle()`.

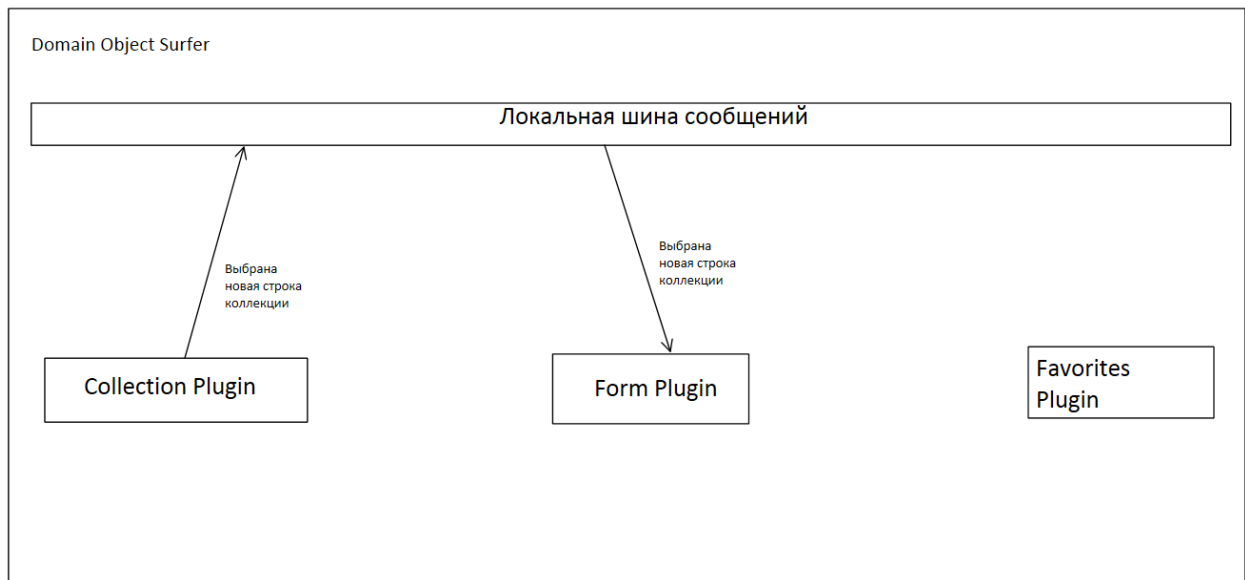
Плагин «Domain Object Surfer»

Самым часто используемым плагином в системах администрирования является «Domain Object Surfer» - плагин, совмещающий плагины «Collection Viewer» и «Form», а также плагин «Favorites». В верхней части этого плагина показывается коллекция, в нижней – форма, соответствующая выбранной строке, в правой части плагина присутствует панель плагина «Избранное», которая может быть свёрнута или развёрнута.

<div>Сохранить</div> <div>Удалить</div> <div>Создать</div> <div>Отправить по процессу</div>	
<div>Collection Plugin</div>	<div>Favorites Plugin</div>
<div>Form Plugin</div>	

Локальная шина сообщений

Плагины и другие компоненты системы часто используют в своём локальном контексте шину сообщений (GWT Event Bus) вместо классических наблюдателей и слушателей. Например, Domain Object Surfer, выступая в качестве «контроллера» между тремя плагинами, использует локальную шину сообщений для организации связывания компонентов.



Формы и виджеты

Одной из ключевых особенностей GUI является механизм «конфигурируемых форм». Данный механизм обеспечивает возможность определить современные, удобные в использовании формы и связать их с данными в хранилище декларативно – не написав ни строчки Java-кода.

Форма состоит из виджетов – элементов управления, при помощи которых пользователь вводит необходимые данные. Примерами виджетов являются: текстовое поле (Text Box), выпадающий список с авто-подсказками (Suggest Box), загрузчик вложений (Attachment Box) и т.д. Виджеты формы размещаются в определённой разметке, о которой будет рассказано далее.

Базовыми понятиями для формы являются корневой доменный объект и его тип и пути от корневого объекта к полю или полям, данные которого представляет конкретный виджет. Корневой доменный объект – это объект, который призвана форма создать или модифицировать средствами виджетов. Все поля формы являются либо частью корневого доменного объекта, либо они принадлежат другому доменному объекту, тем или иным способом связанным с корневым. Механизм форм осуществляет автоматическое «отображение» («mapping») данных, введённых пользователем, на доменные объекты. Данные автоматически считываются и конвертируются в определённое представление (виджеты) на экране. При сохранении осуществляется обратная операция отображения – данные виджетов трансформируются в данные конкретных доменных объекта хранилища.

Независимые доменные объекты в рамках одной формы не поддерживаются.

Путь к полю доменного объекта

Пути к полям доменных объектов определяются DOEL-выражениями. Они являются относительными по отношению к корневому доменному объекту. Предположим есть объекты «Страна» и «Город», описанные следующим образом:

```
<domain-object-type name="country" initial-status="Active">
  <fields>
    <string name="name" length="128"/>
    <dateTime name="independence_day"/>
    <long name="population"/>
    <boolean name = "is_country_rich"/>
    <decimal name="square" precision="15" scale="2"/>
    <reference name="capital" type="city"/>
    <reference name="most_famous_city" type="city"/>
    <string name="description" length="1024"/>
  </fields>
  <uniqueKey>
    <field name="name"/>
  </uniqueKey>
  <attachment-types>
```

```

        <attachment-type name="country_attachment"/>
    </attachment-types>
</domain-object-type>

<domain-object-type name="city" initial-status="Active">
    <fields>
        <reference name="country" type="country"/>
        <string name="name" length="128"/>
        <long name="population"/>
        <long name="year_of_foundation"/>
        <decimal name="latitude" precision="9" scale="2"/>
        <decimal name="longitude" precision="9" scale="2"/>
        <decimal name="square" precision="9" scale="2"/>
    </fields>
</domain-object-type>

<domain-object-type name="country_friend" initial-status="Active">
    <fields>
        <reference name="country" type="country"/>
        <reference name="friend" type="country"/>
    </fields>
</domain-object-type>

```

Город ссылается на страну. В свою очередь, объект «Страна» ссылается на объект «Город» своим полем «Столица». У страны есть «друзья» - другие страны. Друзья определяются в объекте «Друг страны» (country_friend).

На форме редактирования стран мы хотим разместить виджеты, позволяющие отредактировать:

- 1) Название (текстовое поле). Путь относительно корневого объекта: **name**
- 2) Столицу (выпадающий список). Путь: **capital** (прямая ссылка, связь 1:1)
- 3) Население столицы. Путь: **capital.population**
- 4) Загрузить вложения (например, фотографии). Путь: **country_attachment^country** (обратная ссылка, связь 1:N)
- 5) Список городов (выпадающий список с авто-подсказками). Путь: **city^country** (обратная ссылка, связь 1:N)
- 6) Список друзей страны (выпадающий список с авто-подсказками). Путь: **country_friend^country.friend** (обратная ссылка, связь N:M)

Пути к объекту трактуются следующим образом.

- 1) «Прямая ссылка» для типа связи 1:1. Отличается от обычного поля лишь своим типом (ссылка - Reference).
- 2) От ссылочного поля (тип reference) можно сослаться на поле объекта, на который указывает ссылка. Количество последовательных ссылок не ограничено. Например, от базового объекта «Страна» можно сослаться на название главной улицы столицы (если предусмотреть тип объектов «Улица» и поле «Главная улица» в объекте «Город»):
country.capital.main_street.name.
- 3) «Обратная ссылка» для типа связи 1:N указывает от связанного объекта на «родительский». Обратная ссылка состоит из двух частей – название объекта(ов), ссылающегося на родителя (до символа «^») и название поля, которое ссылается (после символа «^»). Например, **city^country** обозначает набор городов, которые ссылаются на страну (или список городов страны).

Так как обратная ссылка, в общем случае, обозначает множество объектов (N), то две обратные ссылки обозначают множество мощности пропорциональное N^2 , или $N \times M$ (декартово произведение). Это имеет очень мало практического смысла, поэтому обратная ссылка поддерживается только одна на весь путь к объекту.

- 4) Ещё один вид обратных ссылок предназначен для тех ситуаций, когда базовый объект и набор связанных с ним объектов имеют мощность связывания N:M и связываются через специальный промежуточный объект. В нашем примере «Друзья страны» (объект типа «Страна») и сами страны связаны между собой именно таким образом. У страны может быть много друзей, у каждого друга – может быть свой список друзей. Промежуточный связывающий объект – это «Друзья страны». Для описания пути к «другу страны» используется следующий вариант описания пути: **country_friend^country.friend**. Путь **country_friend^country** указывает на объекты типа country_friend, которые ссылаются на объект country полем country. **country_friend^country.friend** указывает на поля объектов country_friend с именем friend – таким образом описывается связь N:M (многие-ко-многим). Дальнейшие ссылки на этот вид обратных ссылок не поддерживаются, то есть нельзя написать **country_friend^country.friend.name**, подразумевая «множество названий друзей страны».

Во всех вариантах обратных ссылок, поля (в примерах выше – поле City.country, Country.friend, Country_Friend.country), участвующие в формировании ссылки, обязаны быть ссылками (тип reference).

Некоторые виджеты (например, «Иерархический обозреватель») могут менять связи сразу нескольких типов доменных объектов. В этом случае путь к полям доменных объектов определяется перечислением: **organization_addressee^letter.organization, department_addressee^letter.department, employee_addressee^letter.employee**. Виджеты, меняющие связи будут описаны далее.

Определение формы

Определение формы начинается тэгом **<form>**:

- **<form>** - головной тэг формы. Является тэгом верхнего уровня конфигурации. Атрибуты
 - name** – название (произвольная строка) формы, уникальное в пределах системы. Обязательный атрибут.
 - domain-object-type** – тип корневого доменного объекта формы (строка)
 - is-default** – определяет факт того, что данная форма является таковой «по умолчанию», то есть используется для тех пользователей и ролей, для которых форма данного типа объекта не переопределена. Допустимые значения: **true**, **false**.
 - debug** – определяет, показывать ли форму в режиме отладки (показывать ли явно разметку формы, например) . Допустимые значения: **true**, **false**.
 - min-width** – минимальная ширина, требующаяся для отрисовки формы в единицах CSS (http://www.w3schools.com/cssref/css_units.asp). Пример значения: **"900px"**

Разметка формы

Разметка формы определяется языком, похожим на HTML. Разметка может в себя включать следующие элементы:

- 1) Заголовок, содержащий элементы, расположенные в рамках собственной табличной разметки
- 2) Тело. Состоит из закладок
- 3) Закладки. Содержат «наборы групп».
- 4) «Наборы групп», из которых состоят закладки:
 - a. Группа с единственным содержимым. В этом специфическом «наборе групп», кроме табличной разметки ничего не может присутствовать.
 - b. Скрывающиеся группы. В этом наборе присутствуют группы, которые можно сворачивать и разворачивать.
 - c. Подзакладки. Эти группы выглядят как набор «подзакладок» в текущей закладке.
- 5) Табличная разметка, внутри которой располагаются виджеты. Каждая группа в наборе может содержать табличную разметку

Всё это продемонстрировано на рисунках. Исходная форма выглядит следующим образом:

Название:

Закладки Города и Друзья Attachments Table Browser

Население: Площадь:
 Столица:
 Известный город:
 День независимости:

Название:

Закладки Города и Друзья Attachments Table Browser

▶ **Друзья**
 ▼ **Города**

Города:

На следующих рисунках вышеназванные элементы разметки форм пронумерованы соответственно.

Название:

Закладки Города и Друзья Attachments Table Browser

Население: Площадь:
 Столица:
 Известный город:
 День независимости:

Виджеты

Название:

Закладки Города и Друзья Attachments Table Browser

▶ **Друзья**
 ▼ **Города**

Города:

Название:

Закладки Города и Друзья **Attachments** Table Browser

+

☐

4а) Группа с единственным содержимым. В данном случае её табличная разметка очень проста

Код разметки формы, представленной на данных рисунках выглядит следующим образом:

```
<markup>
  <header>
    <table>
      <tr>
        <td h-align="right">
          <widget id="1"/>
        </td>
        <!-- label (Название) -->
        <td width="100%" h-align="left">
          <widget id="2"/>
        </td>
        <!-- text-box (Название) -->
      </tr>
    </table>
  </header>
  <body display-single-tab="false">
    <tab name="Закладки">
      <bookmarks>
        <tab-group name="Главная">
          <table>
            <tr>
              <td h-align="center" v-align="center">
                <widget id="3"/>
              </td>
              <!-- label (Население) -->
              <td h-align="left">
                <widget id="4"/>
              </td>
              <!-- integer-box (Население) -->
            </tr>
          </table>
        </tab-group>
      </bookmarks>
    </tab>
  </body>
</markup>
```

```

        <td>
            <widget id="5"/>
        </td>
        <!-- label (Площадь) -->
        <td h-align="left">
            <widget id="6"/>
        </td>
        <!-- integer-box (Площадь) -->
    </tr>
    <tr>
        <td h-align="right">
            <widget id="7"/>
        </td>
        <!-- label (Столица) -->
        <td colspan="3" h-align="left">
            <widget id="8"/>
        </td>
        <!-- suggest-box (Столица) -->
    </tr>
    <tr>
        <td h-align="right">
            <widget id="7a"/>
        </td>
        <!-- label (Известный город) -->
        <td colspan="3" h-align="left">
            <widget id="8a"/>
        </td>
        <!-- suggest-box (Известный город) -->
    </tr>
    <tr>
        <td h-align="right">
            <widget id="9"/>
        </td>
        <!-- label (День независимости) -->
        <td colspan="3" h-align="left">
            <widget id="10"/>
        </td>
        <!-- date-box (День независимости) -->
    </tr>
</table>
</tab-group>
<tab-group name="Описание">

```

```

        <table>
            <tr>
                <td h-align="right">
                    <widget id="11"/>
                </td>
                <!-- label (Описание) -->
                <td h-align="left">
                    <widget id="12"/>
                </td>
                <!-- text-area (Описание) -->
            </tr>
        </table>
    </tab-group>
</bookmarks>
</tab>
<tab name="Города и Друзья">
    <hiding-groups>
        <tab-group name="Друзья">
            <table>
                <tr>
                    <td>
                        <widget id="best_friend_label"/>
                    </td>
                    <!-- label (Лучший друг) -->
                    <td h-align="left">
                        <widget id="best_friend"/>
                    </td>
                    <!-- text-box (Лучший друг (название)) -->
                </tr>
                <tr>
                    <td>
                        <widget id="best_desc_label"/>
                    </td>
                    <td h-align="left">
                        <widget id="best_friend_desc"/>
                    </td>
                </tr>
                <tr>
                    <td>
                        <widget id="best_desc_back_label"/>
                    </td>
                    <td h-align="left">

```

```

        <widget id="best_friend_desc_back"/>
    </td>
    <td h-align="left">
        <widget id="16"/>
    </td>
</tr>
</table>
</tab-group>
<tab-group name="Города">
    <table>
        <tr>
            <td h-align="right"> <widget id="13"/> </td>
            <td h-align="left"> <widget id="14"/> </td>
        </tr>
    </table>
</tab-group>
</hiding-groups>
</tab>
<tab name="Attachments">
    <single-entry-group>
        <tab-group>
            <table>
                <tr>
                    <td><widget id = "15" width = "600px"/></td>
                </tr>
                <tr>
                    <td><widget id = "16"/></td>
                </tr>
            </table>
        </tab-group>
    </single-entry-group>
</tab>
<tab name="Table Browser">
    <single-entry-group>
        <tab-group>
            <table>
                <tr>
                    <td><widget id = "17a" width="600px" /></td>
                </tr>
            </table>
        </tab-group>
    </single-entry-group>

```

```
</tab>
</body>
</markup>
```

Описание тэгов разметки (атрибуты и вложенные тэги, если не указано явно, не являются обязательными):

- **<markup>** - головной тэг разметки формы.
- **<header>** - определение заголовка формы. В заголовок можно поместить только тэг **<table>**, обозначающий начало табличной разметки заголовка и других составляющих формы, содержащих непосредственно виджеты. Тэг **<table>** будет описан далее
- **<body>** - определение тела формы. Тело состоит из закладок. Вложенным тэгом может являться только тэг **<tab>**. Атрибуты:

display-single-tab. Возможные значения: **true**, **false**. Определяет, показывать ли закладку (tab) на форме, если она всего одна.
- **<tab>** - закладка формы. Закладки состоят исключительно из «наборов групп» (виджетов). Допустимые вложенные тэги: **<single-entry-group>** (группа с единственным содержимым), **<bookmarks>** (подзакладки), **<hiding-groups>** (скрывающиеся группы). Атрибуты:

name – название закладки.
- **<single-entry-group>** - группа с единственным содержимым. Допустимые вложенные тэги: **<tab-group>**.
- **<bookmarks>** - подзакладки. Допустимые вложенные тэги: **<tab-group>**.
- **<hiding-groups>** - скрывающиеся группы. Допустимые вложенные тэги: **<tab-group>**.
- **<tab-group>** - определяет группу (виджетов) с определённой табличной разметкой. Допустимые вложенные тэги - **<table>**, описывающий разметку данной группы. Атрибуты:

name – название данной группы. В группе с единственным содержимым название не отображается, потому игнорируется.
- **<table>** - определяет табличную разметку. Допустимые вложенные тэги: **<tr>**. Атрибуты:

width – ширина таблицы в единицах CSS (http://www.w3schools.com/cssref/css_units.asp). Пример значения: **"500px"**.

height – высота таблицы в единицах CSS. Пример значения: **"200px"**.

row-height – высота строки таблицы по умолчанию в единицах CSS. Пример значения: **"50px"**.

col-width – ширина колонки таблицы по умолчанию в единицах CSS. Пример значения: **"20%"**

h-align="left" – горизонтальное выравнивание по умолчанию в ячейках таблицы.

Допустимые значения: **left** (по левому краю), **right** (по правому краю), **center** (по центру)

v-align="top" – вертикальное выравнивание по умолчанию в ячейках таблицы.

Допустимые значения: **top** (по верхнему краю), **bottom** (по нижнему краю), **middle** (по центру)

- **<tr>** - определяет строку таблицы. Допустимые вложенные тэги: **<td>**. Атрибуты:

height – высота строки в единицах CSS. Пример значения: **"200px"**.

v-align="top" – вертикальное выравнивание по умолчанию в ячейках строки. Допустимые значения: **top** (по верхнему краю), **bottom** (по нижнему краю), **middle** (по центру)

- **<td>** - определяет ячейку строки таблицы. Именно в ячейках располагаются виджеты, составляющие форму. Допустимые вложенные тэги: **<widget>**. Атрибуты:

colspan – количество ячеек, которые должны быть объединены по горизонтали.

Измеряется в количестве занимаемых ячеек. Пример значения: **"5"**

rowspan – количество ячеек, которые должны быть объединены по вертикали.

Измеряется в количестве занимаемых ячеек. Пример значения: **"3"**

h-align="left" – горизонтальное выравнивание в ячейке. Допустимые значения: **left** (по левому краю), **right** (по правому краю), **center** (по центру)

v-align="top" – вертикальное выравнивание в ячейке. Допустимые значения: **top** (по верхнему краю), **bottom** (по нижнему краю), **middle** (по центру)

- **<widget>** – определяет (ссылается на) виджет, размещённый в ячейке табличной разметки. Конфигурация непосредственно виджетов определяется в отдельном разделе конфигурации формы, описываемом тэгом (см. следующую главу). Атрибуты:

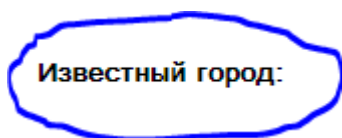
id – идентификатор виджета. Произвольное строковое значение. Примеры значений: **"2"**, **"some_id"**, **"2a"**

Описание виджетов, составляющих форму

Общие атрибуты для всех виджетов системы

- **Каждый** виджет (его тэг) имеет обязательный к определению атрибут **id** – идентификатор виджета – который в дальнейшем описании упоминаться не будет.
- **read-only** – признак того, что виджет доступен только для чтения, независимо от того, находится ли форма в режиме редактирования
- **max-tooltip-width, max-tooltip-height** – определяют максимальный размер всплывающего окна в случае, когда количество показанных элементов виджета превышает установленный лимит. Значение указывается в единицах CSS. Значения по умолчанию – 400 пикселей по ширине и 300 по высоте.

Метка (Label)



«Метка» отображается в виде статического текста на экране. Состояние данного виджета игнорируется при сохранении формы, даже если виджет привязан к каким-либо данным. Метка – единственный виджет, для которого параметр **field-path** является не обязательным. Если параметр **field-path** задан, то текст метки компонуется на основании значения поля, игнорируя, в случае наличия, настройку **<text>**.

Тэг: **<label>**. Тэги параметров:

- **<text>** - отображаемый текст
- **<field-path>** - путь к полю доменного объекта, который используется для генерации метки. Если он определён, то значение **<text>** игнорируется. Атрибуты:
value – значение параметра (обязательный)
- **<relates-to>** Определяет виджет, к которому относится данная метка. Атрибуты:
widget-id - ID виджета, к которому относится данная метка. Если такой виджет существует, то, в зависимости от его пути к полю объекта метка может декорируется «элементом обязательности». Параметр используется только, если метка сконфигурирована как статический текст (при помощи **<text>**).
- **<force-required-asterisk>** - для принудительной установки признака «элемента обязательности». Атрибуты:
value - true|false. В случае true метка декорируется, как обязательная (красная звездочка)
- **<font-weight>** – насыщенность шрифта. Атрибуты:
value – Допустимые значения: normal|bold|bolder|lighter
- **<font-style>** – начертание шрифта. Атрибуты:
value – Допустимые значения: normal | italic | oblique
- **<font-size>** – размер шрифта. Атрибуты:
value – значение в единицах CSS

- **<pattern>** - паттерн отображения значений. Позволяет скомпоновать метку из нескольких полей доменных объектов, на который ссылается виджет в элементе field-path. См. Пример 2. Атрибуты:
value – собственно сам паттерн
- **<all-values-empty-message>** - текст «по-умолчанию», если замена по pattern не удалась. Атрибуты:
value – текст метки
- **<renderer>** - ссылка на компонент, обеспечивающий пользовательский рендеринг виджета. Атрибуты:
component-name – имя спринг-бина
- **<formatting>** - позволяет форматировать числовые поля и даты. Дочерние элементы
<number-format> форматирование числа на базе Java NumberFormat
<date-format> форматирование дат на базе Java DateFormat
См. Пример 3.

Пример 1 конфигурации:

```
<label id="independence_day_label">
  <text>День независимости:</text>
</label>
```

Пример 2:

```
<label id="summary">
  <field-path value="name, population, independence_day,
capital.name"/>
  <font-weight value="bolder"/>
  <font-style value="italic"/>
  <font-size value="8px"/>
  <pattern value="{name}, {population}. День независимости:
{independence_day}, столица: {capital.name}"/>
  <all-values-empty-message value="День независимости:
столица:"/>
</label>
```


Пример 3:

```
<label id="label3">
  <field-path value="phone, certificate"/>
  <pattern value="{phone}, {certificate}, {age}, {date_of_birth}"/>
  <all-values-empty-message value="Номер телефона: сертификат:"/>
  <formatting>
    <number-format pattern="###.###">
      <field-paths>
        <field-path value="age"/>
      </field-paths>
    </number-format>
  </formatting>
</label>
```



```
</number-format>
<date-format pattern="dd-mm-yyyy HH:mm:ss" style="short">
  <time-zone id="Europe/Kiev"/>
  <field-paths>
    <field-path value="date_of_birth"/>
  </field-paths>
</date-format>
</formatting>
</label>
```

Текстовое поле (Text Box)



Виджет предназначен для ввода строк, относительно небольшого размера.

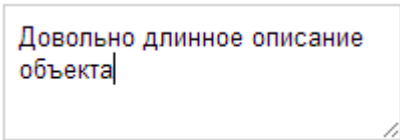
Тэг: **<text-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)

Пример конфигурации:

```
<text-box id="name">
  <field-path value="name"/>
</text-box>
```

Текстовая область (Text Area)



Виджет предназначен для ввода строк, относительно большого размера.

Тэг: **<text-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<text>** - пока не используется

Пример конфигурации:

```
<text-area id="name">
  <field-path value="name"/>
```

```
</text-area>
```

Целочисленное поле (Integer Box)

Виджет предназначен для заполнения целочисленных данных.

Тэг: **<integer-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)

Пример конфигурации:

```
<integer-box id="population">  
  <field-path value="population"/>  
</integer-box>
```

Десятичное поле (Decimal Box)

Виджет предназначен для заполнения десятичных данных с плавающей точкой.

Тэг: **<decimal-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)

Пример конфигурации:

```
<decimal-box id="square">
  <field-path value="square"/>
</decimal-box>
```

Поле даты/времени (Date Box)



Виджет предназначен для ввода даты/времени, в том числе с указанием часового пояса, к которому дата и время относятся.

Тэг: **<date-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<pattern>** - паттерн отображения даты. Значение: собственно паттерн (обязательный). Паттерны соответствуют паттернам дат, описанным в `java.text.SimpleDateFormat`
- **<time-zone-id>** - часовой пояс, в котором отображает дату и время виджет. Значение: идентификатор часового пояса, в котором нужно показывать и считывать дату с временем. Эта настройка не совместима с настройкой `display-time-zone-choice` – виджет должен содержать только одну из них. Соответствует идентификаторам из базы данных часовых поясов Олсона (см. http://en.wikipedia.org/wiki/List_of_tz_database_time_zones). Также

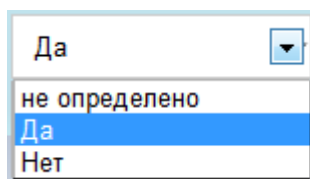
принимаются идентификаторы вида «GMT+03:30», соответствующие абсолютному смещению относительно UTC.

- **<display-time-zone-choice>** - настройка, определяющая, показывать ли список часовых поясов, позволяющий пользователю выбрать тот, которому соответствует вводимая дата. Если поле, за которое отвечает виджет (определяется field-path), типа "timelessDate", то часовые пояса никогда не показываются. Значение – "true", если отображать список часовых поясов, "false" – в противном случае
- **<display-time-box >** - определяет, нужно ли показывать отдельный элемент выбора времени. Значение: "true", если отображать элемент выбора времени, "false" – в противном случае
- **<range-start>** - если данный виджет является одной из составляющих интервала дат, то данная настройка определяет, является ли поле началом интервала. Если дата не является частью интервала, то данную настройку определять не надо. Атрибуты:
widget-id – идентификатор виджета, который определяет конец интервала.
- **<range-end>** - если данный виджет является одной из составляющих интервала дат, то данная настройка определяет, является ли поле концом интервала. Если дата не является частью интервала, то данную настройку определять не надо. Атрибуты:
widget-id – идентификатор виджета, который определяет начало интервала.
- **<range-error-messages>** - (пока не реализовано!) паттерны сообщений об ошибках

Пример конфигурации:

```
<date-box id="independence_day">
  <field-path value="independence_day"/>
  <pattern>dd.MM.yyyy hh:mm:ss"</pattern>
  <time-zone>Europe/Kiev</time-zone>
</date-box>
```

Список фиксированных значений (Enumeration Box)



Список позволяет выбрать единственный вариант из фиксированного списка значений. В отличие от виджетов combo-box и list-box, является виджетом, редактирующим значения, то есть он работает с нессылочными полями. Тип "строковых констант" автоматически определяется по типу поля в field-path. Поддерживаются строковые, числовые (целые и десятичные) и булевы типы.

Значения показываются в порядке описания. Выбранное определяется из значения в базе. Если в базе содержится значение, для которого не прописан mapping, то в enumeration-box добавится ещё одна запись. Если поле в базе содержит null, то в список будет добавлено пустое значение, но

только если поле в доменном объекте сконфигурировано как nullable (не not-null), и при этом не сконфигурировано ни одного null-value.

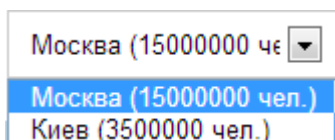
Тэг: **<enumeration-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<mapping>** - содержит один или более тегов **<map>**, которые устанавливают соответствие между фактическим значением поля ДО, и текстом, отображаемым в списке. Атрибуты тэга **<map>**:
value – фактическое значение. value="" в случае строк означает пустую строку, в случае остальных типов - null.
display-text - текст, отображаемый в списке. Если этот атрибут отсутствует, в списке будет отображено значение атрибута **value**.
null-value - true|false. По умолчанию false. null-value="true" указывает, что значение поля равно null. Если этот атрибут указан, value не должен присутствовать (для числовых и булевых типов допускается также указать value="").
- **<map-provider>** - определяет название серверного компонента, который будет возвращать данное отображение кодом. Компонент должен реализовать интерфейс ru.intertrust.cm.core.gui.api.server.widget.EnumerationMapProvider. Атрибуты:
component - имя серверного компонента.
Если задан **map-provider**, то тэг **mapping** не должен присутствовать.

Пример конфигурации:

```
<enumeration-box id="successEnum">
  <field-path value="success"/>
  <mapping>
    <map display-text="<не определено>" null-value="true"/>
    <map display-text="Да" value="true"/>
    <map display-text="Нет" value="false"/>
  </mapping>
</enumeration-box>
```

Выпадающий список (Combo Box)



Выпадающий список позволяет выбирать единственный вариант из фиксированного списка значений.

Тэг: **<combo-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:

value – значение параметра (обязательный)

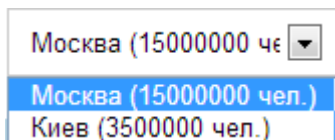
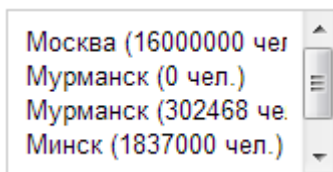
- **<pattern>** - паттерн отображения значений. Позволяет скомпоновать результаты выпадающего списка из полей того типа доменного объекта, на который ссылается виджет. В примере ниже приведён пример того, как выпадающий список формируется списком городов страны (field-path **city^country**). Паттерн включает в себя поля доменного объекта типа City, включая название (name) и население (population). Подобные поля должны быть заключены в фигурные скобки. Атрибуты:

value – собственно сам паттерн

Пример конфигурации:

```
<combo-box id="14">
  <field-path value="city^country"/>
  <pattern value="{name} ({population} чел.)"/>
</combo-box>
```

Список значений (List Box)



Список значений позволяет выбирать одно или несколько значений из фиксированного списка значений.

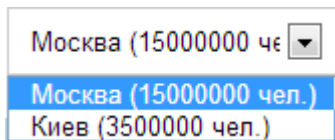
Тэг: **<list-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный). Виджет поддерживает множественные field-path, позволяя отображать/сохранять разнотипными объекты
- **<pattern>** - паттерн отображения значений. Позволяет скомпоновать результаты выпадающего списка из полей того типа доменного объекта, на который ссылается виджет. В примере ниже приведён пример того, как выпадающий список формируется списком городов страны (field-path **city^country**). Паттерн включает в себя поля доменного объекта типа City, включая название (name) и население (population). Подобные поля должны быть заключены в фигурные скобки. Атрибуты:
value – собственно сам паттерн
- **<single-choice>** - определяет, позволяет ли виджет выбирать единственное значение из списка. Атрибуты:
value - «true» если виджет поддерживает выбор единственного элемента, «false» в противном случае. Если выбор единичен, виджет «вырождается» в обычный выпадающий список (Combo Box)

Пример конфигурации:

```
<combo-box id="14">
  <field-path value="city^country"/>
  <pattern value="{name} ({population} чел.)"/>
</combo-box>
```

Радио-кнопка (Radio Button)



Выпадающий список позволяет выбирать из фиксированного списка значений.

Тэг: **<radio-button>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<pattern>** - паттерн отображения значений. Позволяет скомпоновать результаты выпадающего списка из полей того типа доменного объекта, на который ссылается виджет. В примере ниже приведён пример того, как выпадающий список формируется списком городов страны (field-path **city^country**). Паттерн включает в себя поля доменного объекта типа City, включая название (name) и население (population). Подобные поля должны быть заключены в фигурные скобки. Атрибуты:
value – собственно сам паттерн
- **<layout>** - разметка, в которой располагаются кнопки. Если параметр не указан, то кнопки располагаются по вертикали.
value – «horizontal» - если радио-кнопки должны располагаться по горизонтали, или «vertical» - если по вертикали

Пример конфигурации:

```
<radio-button id="cities">
  <field-path value="city^country"/>
  <pattern value="{name} ({population} чел.)"/>
  <layout value="horizontal"
</radio-button>
```

Флажок (Check Box)

Тэг: **<check-box>**. Тэги параметров:


- **<field-path>** - путь к полю доменного объекта. Поле должно иметь булевский тип.
Атрибуты:
value – значение параметра (обязательный)

Пример конфигурации:

```
<check-box id="15">  
  <field-path value="IsDeleted"/>  
</check-box>
```



Табличный обозреватель (Table Browser)

Москва 16000000 1247✕ Мурманск 0 1916✕ Мурманск 302468 1916✕

Поиск  ✕

	id	Имя	Население	Год основания	Широта	Долгота	Площадь
<input checked="" type="checkbox"/>	5035000000	Минск	1837000	1047			
<input type="checkbox"/>	5035000000	Магнитогор	411000	1929			

OK CANCEL

Поиск  ✕

id	Имя	Население	Год основания	Широта	Долгота	Площадь
503500000000	Мурманск	302468	1916			
503500000000	Минск	1837000	1047			
503500000000	Магнитогорос	411000	1929			

OK CANCEL

Выпадающий список позволяет выбирать из фиксированного списка значений.

Тэг: **<table-browser>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<collection-ref>** - ссылка на коллекцию, формирующую список. Атрибуты:
name – имя коллекции (обязательный)
use-default – (не используется)
- **<collection-view-ref>** - ссылка на представление коллекции. Атрибуты:
name – имя представления коллекции (обязательный)
- **<selection-pattern>** - паттерн отображения выбранных значений. списке (см. пример ниже). Название параметра в паттерне соответствует названию атрибута используемой коллекции (<collection-ref>). Атрибуты:
value – собственно сам паттерн
- **<input-text-filter>** - название фильтра по тексту, введённому пользователем. Атрибуты:
name – имя фильтра (обязательный)
- **<page-size>** - количество элементов, показываемых на одной странице выпадающего списка
- **<display-chosen-values>** - отображать уже выбранные значения в списке. Атрибуты:
value – true|false
- **<selection-style>** - стилистика показа выбранных значений.
name – название стиля выбранных значений. может принимать 2 значения inline – «встроенный» стиль (стиль Facebook, Студии Лебедева, Jira и пр.) и table – «табличный стиль». Если параметр не определён, то используется "inline"-вариант.
- **<single-choice>** - определяет, позволяет ли виджет выбирать единственное значение из списка. Атрибуты:
value - «true» если виджет поддерживает выбор единственного элемента, «false» в противном случае.
- **<clear-all-button>** - описание кнопки очистки содержимого. Если элемент отсутствует, кнопка не отображается. Атрибуты:
text – текст на кнопке
image - изображение
- **<add-button>** - описание кнопки открывающей диалог. Если элемент отсутствует, отображается кнопка с текстом «Добавить». Атрибуты:
text – текст на кнопке
image - изображение
- **<default-sort-criteria>** - сортировка по умолчанию. Атрибуты:
column-field – имя колонки
order – порядок сортировки
- **<dialog-window>** - параметры всплывающего окна. Атрибуты:
width – ширина в единицах CSS
height – высота в единицах CSS
- **<display-values-as-links>** - признак того, что значения в списке выбранных значений должны выступать в роли ссылок и открывать форму редактирования ДО. Атрибуты:
value – true|false
- **<initial-filters>** - первичные фильтры для выбора значений. Атрибуты:
panel-state – (open|closed) параметры отображения панели фильтров при 1м открытии, по умолчанию – closed

Тэги

<initial-filter> - описание одного или нескольких фильтров. Пример:

```
<initial-filters panel-state="open">
  <initial-filter name="byName">
    <param name="0" value="Япон*" />
  </initial-filter>
  <initial-filter name="byPopulation">
    <param name="0" value="388" />
    <param name="1" value="83838" />
  </initial-filter>
</initial-filters>
```

- **<formatting>** - позволяет форматировать числовые поля и даты. Дочерние элементы
<number-format> форматирование числа на базе Java NumberFormat
<date-format> форматирование дат на базе Java DateFormat
- **<selection-filters>** - описание фильтров, которые определяют, какие объекты (из реально связанных с корневым объектом) нужно отобразить виджету. Атрибуты:
row-limit – ограничение на количество строк
Тэги
<selection-filter> - описание одного или нескольких фильтров. Пример:

```
<selection-filters>
  <selection-filter name="withoutCountry" />
  <selection-filter name="byText">
    <param name="0" value="Дре%" />
  </selection-filter>
</selection-filters>
```

- **<create-new-button>** - описание кнопки создания нового связанного элемента.
Кнопка не рисуется если:
 - 1) отсутствует элемент **<create-objects>**, либо он пустой;
 - 2) у пользователя нет прав на создание объектов, указанных в **<create-objects>**;
 - 3) у пользователя нет прав на создание дочерних объектов ДО формы, на которой расположен виджет;

Если элемент отсутствует, но приведенные выше условия не соблюдаются, рисуется кнопка с рисунком “+”

Атрибуты:

text – текст на кнопке

image – изображение

- **<create-objects>** содержит список тэгов, нужных для создания нового связанного элемента
Дочерние элементы: **<create-object>**
- **<create-object>** Атрибуты:
domain-object-type – тип ДО, который можно создать
text – текст, который отображается при выборе типа ДО, который нужно создать

Пример конфигурации **<create-objects>**

```
<create-objects>
  <create-object domain-object-type="Employee" text="Сотрудник"/>
  <create-object domain-object-type="Organization" text="Компания"/>
</create-objects>
```

- **<linked-form-mappings>** - список описаний форм, которые могут открываться/создаваться виджетом. Дочерние элементы:
- **<linked-form>** - Описание формы. Атрибуты:
name – имя формы, которую нужно открыть
domain-object-type – тип ДО, для которого открывается форма
Дочерний элемент: **<title>**
- **<title>** - Описание заголовка формы
Дочерние элементы: **<existing-object>** и **<new-object>**
- **<existing-object>** - заголовок для сохраненного ДО.
Дочерние элементы: **<pattern>** и **<formatting>**
- **<pattern>** - шаблон заголовка
- **<formatting>** - позволяет форматировать числовые поля и даты.
Дочерние элементы: **<number-format>** и **<date-format>**
- **<number-format>** форматирование числа на базе Java NumberFormat
- **<date-format>** форматирование дат на базе Java DateFormat
- **<new-object>** заголовок для несохраненного ДО
Дочерние элементы: **<pattern>** и **<formatting>**
- **<pattern>** - шаблон заголовка
- **<formatting>** - позволяет форматировать числовые поля и даты.
Дочерние элементы: **<number-format>** и **<date-format>**
- **<number-format>** форматирование числа на базе Java NumberFormat
- **<date-format>** форматирование дат на базе Java DateFormat
- Пример конфигурации **<linked-form-mappings>**

```
<linked-form-mappings>
  <linked-form domain-object-type="Employee" name="employee_form"/>
  <linked-form domain-object-type="Organization" name="org_form"/>
</linked-form-mappings>
```

Пример конфигурации **<linked-form>**

```
<linked-form domain-object-type="Employee" name="employee_form"/>
  <title>
    <existing-object>
      <pattern value="Дата создания: {created_date}"/>
      <formatting>
        <date-format pattern="dd-mm-yyyy HH-mm-ss" style="short">
          <time-zone id="Europe/Kiev"/>
```

```

        <field-paths>
            <field-path value="created_date"/>
        </field-paths>
    </date-format>
</formatting>
</existing-object>
<new-object>
    <pattern value="Новая организация"/>
</new-object>
</title>
</linked-form>

```

Пример конфигурации:

```

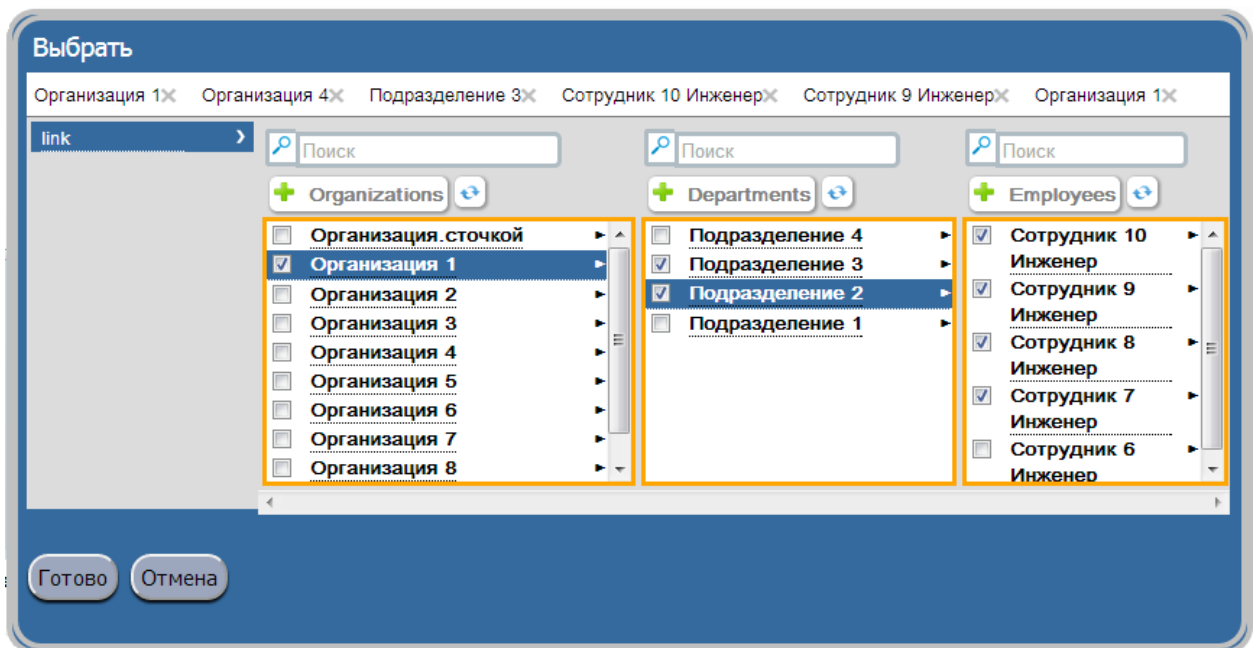
<table-browser id="cities">
    <field-path value="city^country"/>
    <collection-ref name="Cities"/>
    <collection-view-ref name="cities_view"/>
    <selection-pattern value="{name} {population} {year_of_foundation}"/>
    <input-text-filter name="byText"/>
    <page-size>30</page-size>
    <display-chosen-values value="false"/>
    <selection-style name="table"/>
    <single-choice value="true"/>
</table-browser>

```

Обозреватель иерархии (Hierarchy Browser)

Организация 1X Организация 4X Подразделение 3X Сотрудник 10 ИнженерX
 Сотрудник 9 ИнженерX

+ Выбрать



Выпадающий список позволяет выбирать из фиксированного списка значений.

Тэг: **<hierarchy-browser>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный). Виджет поддерживает множественные field-path, позволяя отображать/сохранять разнотипными объекты
- **<node-collection-def>** - Определение списка/таблицы, выпадающего при раскрытии узла. Атрибуты:
collection - название коллекции для отображения списка значений узла
parent-filter – фильтр, определяющий путь к узлу верхнего уровня (относительно пути этого узла) от узла данного уровня
selective – позволяет/запрещает (true/false) выбирать элементы в ноде (отображение check-box)
domain-object-type – тип доменного объекта
title – заголовок. Не поддерживается
display-create-button - (true/false) отображать/не отображать кнопку создания новых элементов
Дочерние тэги:
<input-text-filter> - название фильтра по тексту, введённому пользователем
<selection-pattern> - паттерн отображения выбранных значений. списке (см. пример ниже). Название параметра в паттерне соответствует названию атрибута используемой коллекции (<node-collection-def>)
<default-sort-criteria> - сортировка по умолчанию
<node-collection-def> - определение дочернего списка
<fill-parent-on-add> - если этот тэг указан, то ссылка на родителя будет заполняться автоматически
<root-node-link> -
<selection-filters> - описание фильтров, которые определяет, какие объекты (из реально связанных с корневым объектом) нужно отобразить виджету

<display-values-as-links> - признак того, что значения в списке выбранных значений должны выступать в роли ссылок и открывать форму редактирования ДО. Атрибуты: **value** – true|false. Приоритет выше, чем у идентичного тега внутри **<hierarchy-browser>**.

<single-choice> - определяет, позволяет ли виджет выбирать единственное значение из списка. Атрибуты:

value - «true» если виджет поддерживает выбор единственного элемента, «false» в противном случае.

Приоритет выше, чем у идентичного тега внутри **<hierarchy-browser>**.

<create-new-button> - описания кнопки создания нового связанного элемента.

Кнопка не рисуется если:

- 1) отсутствует элемент **<create-objects>**, либо он пустой;
- 2) у пользователя нет прав на создание объектов, указанных в **<create-objects>**;
- 3) у пользователя нет прав на создание дочерних объектов ДО формы, на которой расположен виджет;

Если элемент отсутствует, но приведенные выше условия не соблюдаются, рисуется кнопка с рисунком “+”

Атрибуты:

text – текст на кнопке

image – изображение

<create-objects> содержит список тэгов, нужных для создания нового связанного элемента

Дочерние элементы: **<create-object>**

<create-object> Атрибуты:

domain-object-type – тип ДО, который можно создать

text – текст, который отображается при выборе типа ДО, который нужно создать

Пример конфигурации **<create-objects>**

```
<create-objects>
  <create-object domain-object-type="Employee" text="Сотрудник"/>
  <create-object domain-object-type="Organization" text="Компания"/>
</create-objects >
```

<linked-form-mappings> - список описаний форм, которые могут открываться/создаваться виджетом. Дочерние элементы:

<linked-form> - Описание формы. Атрибуты:

name – имя формы, которую нужно открыть

domain-object-type – тип ДО, для которого открывается форма

Дочерний элемент: **<title>**

<title> - Описание заголовка формы

Дочерние элементы: **<existing-object>** и **<new-object>**

<existing-object> - заголовок для сохраненного ДО.

Дочерние элементы: **<pattern>** и **<formatting>**

<pattern> - шаблон заголовка

<formatting> - позволяет форматировать числовые поля и даты.

Дочерние элементы: **<number-format>** и **<date-format>**

<number-format> форматирование числа на базе Java NumberFormat

<date-format> форматирование дат на базе Java DateFormat

<new-object> заголовок для несохраненного ДО

Дочерние элементы: **<pattern>** и **<formatting>**

<pattern> - шаблон заголовка

<formatting> - позволяет форматировать числовые поля и даты.

Дочерние элементы: **<number-format>** и **<date-format>**

<number-format> форматирование числа на базе Java NumberFormat

<date-format> форматирование дат на базе Java DateFormat

Пример конфигурации **<linked-form-mappings>**

```
<linked-form-mappings>
  <linked-form domain-object-type="Employee" name="employee_form"/>
  <linked-form domain-object-type="Organization" name="org_form"/>
</linked-form-mappings>
```

Пример конфигурации **<linked-form>**

```
<linked-form domain-object-type="Employee" name="employee_form"/>
  <title>
    <existing-object>
      <pattern value="Дата создания: {created_date}"/>
      <formatting>
        <date-format pattern="dd-mm-yyyy HH-mm-ss" style="short">
          <time-zone id="Europe/Kiev"/>
          <field-paths>
            <field-path value="created_date"/>
          </field-paths>
        </date-format>
      </formatting>
    </existing-object>
    <new-object>
      <pattern value="Новая организация"/>
    </new-object>
  </title>
</linked-form>
```

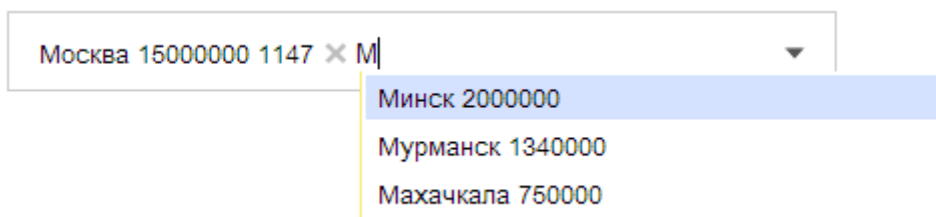
- **<page-size>** - количество элементов, показываемых в данном узле дерева изначально
- **<single-choice>** - определяет, позволяет ли виджет выбирать единственное значение из списка. Атрибуты:
value - «true» если виджет поддерживает выбор единственного элемента, «false» в противном случае.

- **<clear-all-button>** - описание кнопки очистки содержимого. Если элемент отсутствует, кнопка не отображается. Атрибуты:
text – текст на кнопке
image - изображение
- **<add-button>** - описание кнопки открывающей диалог. Если элемент отсутствует, отображается кнопка с текстом «Добавить». Атрибуты:
text – текст на кнопке
image - изображение
- **<selection-style>** - стилистика показа выбранных значений.
name – название стиля выбранных значений. может принимать 2 значения inline – «встроенный» стиль (стиль Facebook, Студии Лебедева, Jira и пр.) и table – «табличный стиль». Если параметр не определён, то используется "inline"-вариант.
- **<dialog-window>** - параметры всплывающего окна. Атрибуты:
width – ширина в единицах CSS
height – высота в единицах CSS
- **<display-values-as-links>** - признак того, что значения в списке выбранных значений должны выступать в роли ссылок и открывать форму редактирования ДО. Атрибуты:
value – true|false
- **<formatting>** - позволяет форматировать числовые поля и даты. Дочерние элементы
<number-format> форматирование числа на базе Java NumberFormat
- **<date-format>** форматирование дат на базе Java DateFormat

Пример конфигурации:

```
<hierarchy-browser id="2c">
  <field-path value="organization_addressee^letter.organization,
              department_addressee^letter.department,
              employee_addressee^letter.employee"/>
  <node-collection-def collection="Organizations" >
    <input-text-filter name="byName"/>
    <selection-pattern value="{name}"/>
    <node-collection-def collection="Departments"
                        parent-filter="byOrganization">
      <input-text-filter name="byName"/>
      <selection-pattern value="{name}"/>
      <node-collection-def collection="Employees"
                          parent-filter="byDepartment">
        <input-text-filter name="byNameAndPosition"/>
        <selection-pattern value="{name} {position}"/>
      </node-collection-def>
    </node-collection-def>
  </node-collection-def>
  <page-size>10</page-size>
</hierarchy-browser>
```

Выпадающий список с подсказками (Suggest Box)



Выпадающий список с подсказками позволяет выбирать из списка значений, при этом обеспечивая фильтрацию по тому тексту, который вводит пользователь.

Тэг: **<suggest-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<collection-ref>** - коллекция, необходимая для отображения записей в выпадающем списке и уже выбранных записей
name – название коллекции
- **<input-text-filter>** - фильтр коллекции, который используется для фильтрация по тексту, введённому пользователем
name – название фильтра
- **<drop-down-pattern>** - паттерн отображения значений в выпадающем списке (см. пример ниже). Название параметра в паттерне соответствует названию атрибута используемой коллекции (<collection-ref>).
value – сам паттерн отображения
- **<selection-pattern>** - паттерн отображения уже выбранных значений. Определяется аналогично <drop-down-pattern>
value – сам паттерн отображения
- **<page-size>** - размер страницы выпадающего списка значений
- **<selection-style>** - стилистика показа выбранных значений.
name – название стиля выбранных значений. может принимать 2 значения inline – «встроенный» стиль (стиль Facebook, Студии Лебедева, Jira и пр.) и table – «табличный стиль». Если параметр не определён, то используется "inline"-вариант.
- **<max-drop-down-width>** - устанавливает максимальную ширину (в пикселях) для выпадающего списка. Больше не поддерживается.
- **<max-drop-down-height>** - устанавливает максимальную высоту (в пикселях) для выпадающего списка. Больше не поддерживается.
- **<single-choice>** - определяет, позволяет ли виджет выбирать единственное значение из списка. Атрибуты:
value - «true» если виджет поддерживает выбор единственного элемента, «false» в противном случае.
- **<clear-all-button>** - описание кнопки очистки содержимого. Если элемент отсутствует, кнопка не отображается. Атрибуты:
text – текст на кнопке
image – изображение
- **<default-sort-criteria>** - сортировка по умолчанию

- **<display-values-as-links>** - признак того, что значения в списке выбранных значений должны выступать в роли ссылок и открывать форму редактирования ДО. Атрибуты:
value – true|false
- **<formatting>** - позволяет форматировать числовые поля и даты. Дочерние элементы
<number-format> форматирование числа на базе Java NumberFormat
<date-format> форматирование дат на базе Java DateFormat
- **<selection-filters>** - описание фильтров, которые определяет, какие объекты (из реально связанных с корневым объектом) нужно отобразить виджету
- **<create-new-button>** - описания кнопки создания нового связанного элемента.
Кнопка не рисуется если:
 - 4) отсутствует элемент **<create-objects>**, либо он пустой;
 - 5) у пользователя нет прав на создание объектов, указанных в **<create-objects>**;
 - 6) у пользователя нет прав на создание дочерних объектов ДО формы, на которой расположен виджет;

Если элемент отсутствует, но приведенные выше условия не соблюдаются, рисуется кнопка с рисунком “+”

Атрибуты:

text – текст на кнопке

image – изображение

- **<create-objects>** содержит список тэгов, нужных для создания нового связанного элемента
Дочерние элементы: **<create-object>**
- **<create-object>** Атрибуты:
domain-object-type – тип ДО, который можно создать
text – текст, который отображается при выборе типа ДО, который нужно создать
Пример конфигурации **<create-objects>**

```
<create-objects>
  <create-object domain-object-type="Employee" text="Сотрудник"/>
  <create-object domain-object-type="Organization" text="Компания"/>
</create-objects >
```

- **<linked-form-mappings>** - список описаний форм, которые могут открываться/создаваться виджетом. Дочерние элементы:
- **<linked-form>** - Описание формы. Атрибуты:
name – имя формы, которую нужно открыть
domain-object-type – тип ДО, для которого открывается форма
Дочерний элемент: **<title>**
- **<title>** - Описание заголовка формы
Дочерние элементы: **<existing-object>** и **<new-object>**
- **<existing-object>** - заголовок для сохраненного ДО.
Дочерние элементы: **<pattern>** и **<formatting>**
- **<pattern>** - шаблон заголовка
- **<formatting>** - позволяет форматировать числовые поля и даты.
Дочерние элементы: **<number-format>** и **<date-format>**

- **<number-format>** форматирование числа на базе Java NumberFormat
- **<date-format>** форматирование дат на базе Java DateFormat
- **<new-object>** заголовок для несохраненного ДО
Дочерние элементы: **<pattern>** и **<formatting>**
- **<pattern>** - шаблон заголовка
- **<formatting>** - позволяет форматировать числовые поля и даты.
Дочерние элементы: **<number-format>** и **<date-format>**
- **<number-format>** форматирование числа на базе Java NumberFormat
- **<date-format>** форматирование дат на базе Java DateFormat
- Пример конфигурации **<linked-form-mappings>**

```
<linked-form-mappings>
  <linked-form domain-object-type="Employee" name="employee_form"/>
  <linked-form domain-object-type="Organization" name="org_form"/>
</linked-form-mappings>
```

Пример конфигурации **<linked-form>**

```
<linked-form domain-object-type="Employee" name="employee_form"/>
  <title>
    <existing-object>
      <pattern value="Дата создания: {created_date}"/>
      <formatting>
        <date-format pattern="dd-mm-yyyy HH-mm-ss" style="short">
          <time-zone id="Europe/Kiev"/>
          <field-paths>
            <field-path value="created_date"/>
          </field-paths>
        </date-format>
      </formatting>
    </existing-object>
    <new-object>
      <pattern value="Новая организация"/>
    </new-object>
  </title>
</linked-form>
```

Пример конфигурации:

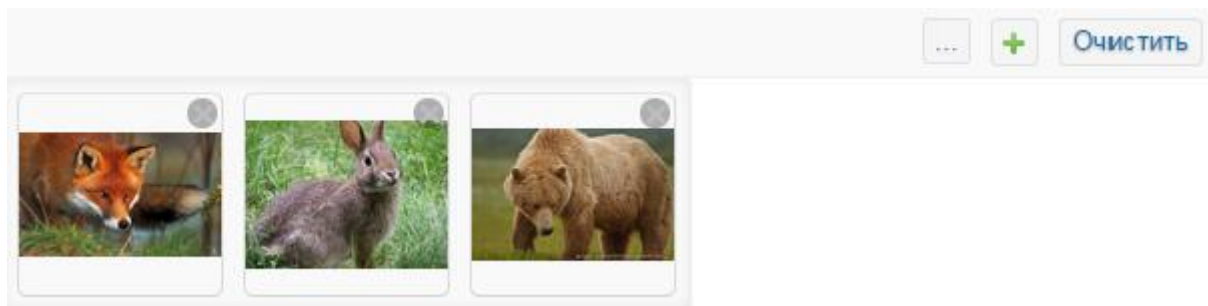
```
<suggest-box id="most_famout_cities">
  <field-path value="country_most_famous_city^country.city"/>
  <collection-ref name="Cities"/>
  <input-text-filter name="byText"/>
```

```
<drop-down-pattern value="{name} {population}"/>
<selection-pattern value="{name} {population} {year_of_foundation}"/>
<page-size>30</page-size>
<selection-style name="table"/>
</suggest-box>
```

Набор вложений (Attachment Box)



[documents-export-2013-11-18.zip \(884,1 KB\)](#) × [cmj-webgui_new.rar \(305,9 MB\)](#) ×
[gwt-2.5.1.zip \(105,9 MB\)](#) ×



Этот виджет загружает на сервер и с сервера вложения к доменным объектам.

Тэг: **<attachment-box>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<attachment-type-ref>** - тип доменного объекта вложений
name – название типа доменного объекта вложений
- **<scanner>** - поддержка использования сканнера для создания вложений
enabled – «true» если сканнер поддерживается, «false» в противном случае
- **<selection-style>** - стилистика показа выбранных значений.
name – название стиля выбранных значений. может принимать 2 значения inline – «встроенный» стиль (стиль Facebook, Студии Лебедева, Jira и пр.) и table – «табличный стиль». Если параметр не определён, то используется "inline"-вариант.
- **<single-choice>** - определяет, позволяет ли виджет выбирать единственное значение из списка. Атрибуты:
value - «true» если виджет поддерживает выбор единственного элемента, «false» в противном случае.
- **<action-link>** - позволяет выполнять некоторые действия из Attachment Box. Атрибуты:
text – текст ссылки.
action-name - название компонента-действия (наследника Action).
- **<accepted-types>** - перечисление допустимых MIME-типов для загрузки

- **<delete-button>** - конфигурация кнопки «Удалить» (крестика в углу картинки). Атрибут **display** true|false
- **<add-button>** - конфигурация кнопки «Добавить». Атрибуты: **display** true|false, **text**, **image**.
- **<clear-all-button>** - конфигурация кнопки «Очистить». Атрибуты: **display** true|false, **text**, **image**.
- **<images-only>** - указывает, что виджет позволяет загрузить исключительно изображения. При этом виджет будет отображать не названия вложений, а сами изображения в уменьшенном варианте. Вложенные тэги:
 - **<small-preview>** – конфигурация маленького превью изображения.
 - **<large-preview>** – конфигурация большого превью, появляющегося при клике на маленьком превью.
 - **<read-only-preview>** – конфигурация превью в read-only режиме.
 Эти тэги не являются обязательными. Атрибуты: **display** - true|false, показывать ли изображение, по умолчанию "true"; **width** – ширина; **height** – высота; **preserve-proportion** true|false - соблюдать пропорции картинок, по умолчанию "true".
- **<choice-style>** – определяет внешний вид виджета и способ выбора элемента. Атрибут: **name** "inline|popup". inline – вариант по умолчанию, когда отображаются все доступные вложения. popup - вариант, когда виджет отображает лишь выбранные элементы, а по кнопке "выбрать" отображается всплывающее окно, позволяющее выбрать вложения.

Пример конфигурации:

```
<attachment-box id="country_attachments">
  <field-path value="country_attachment^country"/>
  <attachment-type-ref name="country_attachment"/>
  <scanner enabled="true"/>
  <selection-style name="table"/>
</attachment-box>
```

Таблица связанных доменных объектов (Linked Domain Objects Table)

Данный виджет предназначен для создания, редактирования и удаления доменных объектов, связанных с базовым доменным объектом. Редактирование производится в открываемых дочерних формах связанных объектов.

Тэг: **<linked-domain-objects-table>**. Атрибуты:

- **modal-width** – ширина виджета в единицах CSS
- **modal-height** – высота виджета в единицах CSS
- **delete-linked-objects** – если определён как true, то связанные доменные объекты будут автоматически удалены:

Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты: **value** – значение параметра (обязательный)

- **<linked-form>** - форма, открывающаяся при нажатии на гиперссылку. Атрибуты:
name – название формы
inline - true|false. Если true, то на месте текущего плагина открывается дочерний. Если false (по умолчанию), то открывается в новом окне.
- **<summary-table>** - описание таблицы, отображающей сводную информацию о связанных доменных объектах. Вложенные тэги:
 - **<summary-table-column>** - Конфигурация колонки сводной таблицы. Атрибуты:
 - **header** – заголовок колонки
 - **widget-id** – идентификатор виджета связанной формы, данные которого используются при построении данной колонки
 Тэги:
<pattern> - паттерн отображения значений в колонках
<formatting> - позволяет форматировать числовые поля и даты
- **<single-choice>** - определяет, позволяет ли виджет выбирать единственное значение из списка. Атрибуты:
value - «true» если виджет поддерживает выбор единственного элемента, «false» в противном случае.
- **<selection-filters>** - описание фильтров, которые определяет, какие объекты (из реально связанных с корневым объектом) нужно отобразить виджету
- **<collection-ref>** - коллекция, определяющая содержимое виджета

Пример конфигурации:

```
<linked-domain-objects-table id="city_table" modal-width="500px" modal-
height="300px"
                                max-tooltip-width="300px">
  <field-path value="country_most_famous_city^country.city"/>
  <linked-form name="city_form" inline="true"/>
  <collection-ref name="Cities"/>
  <selection-filters row-limit="1"/>
  <summary-table>
    <summary-table-column header="Название" widget-id="2">
      <pattern value="{name}"/>
      <!-- пути задаются относительно базового field-path
      (correspondent) -->
      <formatting>
        <number-format pattern="###.###">
          <field-paths> <!-- описание путей к полям, к которым
          применяется форматирование -->
            <field-path value="age"/>
          </field-paths>
        </number-format>
        <date-format pattern="dd-mm-yyyy" style="short">
          <field-paths>
            <field-path value="date_of_birth"/>
          </field-paths>
        </date-format>
      </formatting>
    </summary-table-column>
  </summary-table>
</linked-domain-objects-table>
```

```

        </field-paths>
    </date-format>
</formatting>
</summary-table-column>
<summary-table-column header="Население" widget-id="4">
    <pattern value="{population}"/>
</summary-table-column>
</summary-table>
<single-choice value="false"/>
</linked-domain-objects-table>

```

Редактируемая таблица связанных доменных объектов (Linked Domain Objects Editable Table)

Виджет предназначен для создания, редактирования и удаления доменных объектов, связанных с базовым доменным объектом. Редактирование производится непосредственно в таблице, отображающей поля связанных объектов.

Тэг: **<linked-domain-objects-editable-table>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<linked-form>** - форма, открывающаяся при нажатии на гиперссылку. Атрибуты:
name – название формы
inline - true|false. Если true, то на месте текущего плагина открывается дочерний. Если false (по умолчанию), то открывается в новом окне.
- **<form-table>** - описание таблицы, отображающей сводную информацию о связанных доменных объектах. Вложенные тэги:
 - **<form-table-column>** - Конфигурация колонки сводной таблицы. Атрибуты:
 - **header** – заголовок колонки
 - **widget-id** – идентификатор виджета связанной формы, данные которого используются при построении данной колонки

Пример конфигурации:

```

<linked-domain-objects-editable-table id="5">
    <field-path value="correspondent"/>
    <linked-form name="organization_default_form" inline="true"/>
    <form-table form-name="some_form"> <!-- у этой формы базовый объект
    должен быть такого же типа, как correspondent -->
        <form-table-column header="Название" widget-id="name"/>
        <form-table-column header="Описание" widget-id="description"/>
    </form-table>
</linked-domain-objects-editable-table>

```


Гиперссылка на доменный объект (Linked Domain Object Hyperlink)

Этот виджет отображает ссылку на другие доменные объекты.

Тэг: **<linked-domain-object-hyperlink>**. Тэги параметров:

- **<field-path>** - путь к полю доменного объекта. Атрибуты:
value – значение параметра (обязательный)
- **<linked-form>** - форма, открываемая при нажатии на гиперссылку. Атрибуты:
name – название формы
inline - true|false. Если true, то на месте текущего плагина открывается дочерний. Если false (по умолчанию), то открывается в новом окне.
- **<pattern>** - паттерн отображения значений гиперлинках (см. пример ниже). Название параметра в паттерне соответствует названию поля связанного объекта.
value – сам паттерн отображения
- **<formatting>** - позволяет форматировать числовые поля и даты
- **<collection-ref>** - коллекция, определяющая содержимое виджета

Пример конфигурации:

```
<linked-domain-object-hyperlink id="hyperlink"> <!-- для единичного
связанного объекта -->
  <field-path value="department^organization"/>
  <!-- для связанных объектов field-path обязательно должен ссылаться на
поле типа reference -->
  <linked-form name="organization_default_form"/>
  <!-- форма, открываемая в модальном окне при нажатии на гиперссылку -->
  <pattern value="{name} {created_date}"/>
  <!-- в паттерне ссылки задаются относительно field-path -->
  <selection-filters row-limit="1"></selection-filters>
  <formatting>

    <date-format pattern="dd-mm-yyyy HH:mm:ss" style="short">
      <time-zone id="Europe/Kiev"/>
      <field-paths>
        <field-path value="created_date"/>
      </field-paths>
    </date-format>
  </formatting>
  <collection-ref name="Departments"/>
</linked-domain-object-hyperlink>
```

Определение форм для ролей и конечных пользователей

Конфигурация

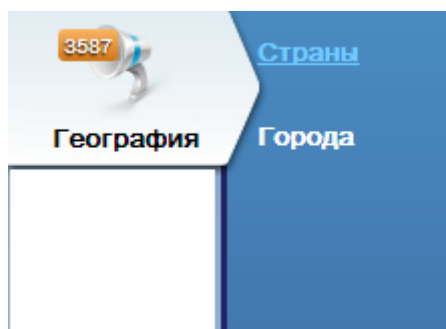
Данный раздел описывает конфигурирование пользовательского интерфейса, в частности следующие его составные части: панель навигации, плагин коллекций, формы и действия.

Панель навигации

Панель навигации настраивается конфигурационно. Пример конфигурации панели навигации:

```
<navigation name="panel" is-default="true">
  <link name="Geography" display-text="География"
    image="images/inbox.png" child-to-open="Countries">
    <child-links>
      <link name="Countries" display-text="Страны">
        <plugin>
          <domain-object-surfer
            domain-object-type-to-create="country">
            <collection-viewer>
              <collection-ref name="Countries"/>
              <sort-criterion field="name" order="asc"/>
            </collection-viewer>
          </domain-object-surfer>
        </plugin>
      </link>
      <link name="Cities" display-text="Города">
        <plugin>
          <domain-object-surfer
            domain-object-type-to-create="city">
            <collection-viewer>
              <collection-ref name="Cities"/>
              <sort-criterion field="name" order="asc"/>
            </collection-viewer>
          </domain-object-surfer>
        </plugin>
        <decorations>
          <collection-counter/>
        </decorations>
      </link>
    </child-links>
  </link>
</navigation>
```

Такая конфигурация сформирует следующую панель навигации:



Панель навигации можно определить для каждой роли и для конкретных пользователей (прим.: ещё не реализовано).