«HARDROLLER»

РУКОВОДСТВО ПО ИСПОЛЬЗОВАНИЮ

Содержание

1 Термины и определения

В настоящем документе используются следующие термины:

Термин	**Определение**
ПО	Программное обеспечение «HARDROLLER»
Платформа	ПО в своей основной функциональной роли – серверное ПО для исполнения программных модулей, непосредственно решающих прикладные задачи.
Бизнес-приложение	Прикладное приложение для решения отраслевых задач автоматизации, построенное на основе Платформы
Конфигурация	Бизнес-приложение в терминологии Платформы.
Объектная модель предметной области	Модель данных бизнес-приложения на основе Платформы, построенная на основе объектно-ориентированного подхода.
Организация	Общее рабочее пространстве для пользователей в рамках Платформы.
Проект	Конфигурация и соответствующий ей набор связанных данных (экземпляр конфигурации), обеспечивающие автоматизацию каких-либо бизнес-процессов.
БД	База данных
HTTP	HyperText Transfer Protocol – протокол передачи гипертекста
API	Application Programming Interface – набор компонентов, с помощью которых другая компьютерная программа может использовать платформу
JSON	JavaScript Object Notation – текстовый формат обмена данными
PDF	Portable Document Format – межплатформенный формат электронных документов
TypeScript	Язык программирования, средство разработки веб-приложений
React	Библиотека с открытым исходным кодом для разработки пользовательских интерфейсов

2 Введение

Настоящий документ содержит руководство по использованию платформы «HARDROLLER», даёт описание основных инструментов, предоставляемых платформой, и примеры их использования.

3 Использование платформы

3.1 Запуск в окружении разработчика

Инструкция по развертыванию ПО в окружении разработчика описан в документе «Руководство по установке» к настоящему ПО.

3.2 Публикация конфигурации в продуктивном окружении

Для публикации конфигурации в продуктивном окружении необходимо в папке hardroller в корневой директории конфигурации создать файл production.env.json и указать в нем сетевой адрес сервера HARDROLLER (например, https://reg1.hardroller.ru) и API-ключ суперпользователя, который можно получить по адресу **/Api/General/IssueApiKey** (например, https://reg1.hardroller.ru/Api/General/IssueApiKey).

После этого выполнить в корневой директории конфигурации при помощи терминала команду *npm run publish*.

HARDROLLER поддерживает возможность работы с несколькими продуктивными окружениями одновременно. Альтернативные окружения создаются при помощи файлов вида **environment_name**.env.json с параметрами, аналогичными файлу production.evn.json. Публикация в этом случае выполняется с помощью терминальной команды *npm run publish* ***environment_name***.

4 Возможности платформы и их использование

4.1 Описание сущностей предметной области

Все файлы с описанием сущности (само описание сущности, описание его печатных форм, форм списка, обработчики) размещаются в директориях внутри директории src/ конфигурации. Описание сущности производится в формате xml.

Имя файла имеет вид **%ИмяСущности%.Domain.xml**.

Сущность данных может содержать следующие свойства:

Тип свойства	**Свойство**	**Пояснение**
FileProperty	Attachment	Произвольный файл
StringProperty	String	Строка
DateTimeProperty	DateTime	Дата и время
BoolProperty	Boolean	Булево
IntegerProperty	Integer	Целое число
DoubleProperty	Double	Число с плавающей точкой
EnumProperty	Enum	Перечисление
ReferenceProperty	Reference	Ссылка
ProjectProperty	Project	Проект
UserProperty	User	Пользователь
ListProperty	List	Список
ExpressionProperty	Expression	Выражение
FileListProperty	FileList	Список файлов
GuidProperty	Guid	GUID

Для каждого свойства можно задать следующие ограничения на значение (правила валидации):

Вид ограничения	**Пояснение**
GreaterThen	Больше
LessThenOrEqualTo	Меньше или равно
NotNull	Непустое значение
Script	Произвольная проверка
After	Хронологически позже
Before	Хронологически раньше
NotAfter	Хронологически не позже
NotBefore	Хронологически не раньше

Пример описания сущности приведён ниже, описана простая сущность «Сотрудник» с двумя строковыми свойствами: «FullName» (есть проверка на не пустоту значения) и «LastNameAndInitials».

```
<!-- обязательный заголовок файла, в котором описывается сущность -->
<?xml version="1.0" encoding="utf-8"?>
<DomainFile>
   <!-- секция описания сущностей -->
   <Entities>
      <!-- определение сущности -->
      <Entity Name="Employee" SimpleBusinessObjectTypeId="9baf041a-e27b-4249-907c-b570482dc9ce">
         <Properties>
            <!-- секция описания свойства -->
            <String Name="FullName">
               <!-- пример проверки -->
               <Validations>
                  <!-- проверка на не пустое значение с указанием сообщения об ошибке -->
                  <NotNull>Please provide a value</NotNull>
               </Validations>
            </String>

            <!-- секция описания свойства (без проверок) -->
            <String Name="LastNameAndInitials"/>

         </Properties>

         <!-- секция обработчиков событий -->
         <EventHandlers>
         </EventHandlers>
      </Entity>
   </Entities>
</DomainFile>
```

В том же файле могут быть описаны сущности, которые использует описываемую (в данном контексте – корневой) сущность, в том числе – перечисления. Пример приведён ниже, описана сущность «Invoice».

```
<!-- обязательный заголовок файла, в котором описывается сущность -->
<?xml version="1.0" encoding="utf-8"?>
<DomainFile>
   <!-- секция описания сущностей -->
   <Entities>
      <!-- определение сущности -->
      <Entity Name="Invoice" SimpleBusinessObjectTypeId="cb193b05-d15c-4f51-bce6-a9bd95aee36f">
         <!-- секция описания свойства -->
         <Properties>
            <!-- секция описания свойства -->
            <!-- тип - описанный в другом каталоге сущность «BusinessEntity» -->
            <!-- есть проверка на не пустое значение -->
            <Reference Name="InvoiceBusinessEntity" ChildEntity="BusinessEntity">
               <Validations>
                  <NotNull>Please provide a value</NotNull>
               </Validations>
            </Reference>

            <!-- секция описания свойства -->
            <!-- тип - описанный в другом каталоге сущность "Employee" -->
            <!-- есть проверка на не пустое значение -->
            <Reference Name="ResponsiblePerson" ChildEntity="Employee">
               <Validations>
                  <NotNull>Please provide a value</NotNull>
               </Validations>
            </Reference>

            <!-- секция описания свойства -->
            <!-- тип - список значений, -->
            <!-- имеющих тип описанного ниже в этом файле сущности "InvoiceItem" -->
            <!-- InvoiceItem.Index декларировано как свойство, отвечающее за нумерацию списка -->
            <List Name="Items" ChildEntity="InvoiceItem" IndexProperty="Index"/>

            <!-- секция описания свойства -->
            <!-- тип - строка -->
```

```
            <String Name="InvoiceNumber"/>

            <!-- секция описания свойства -->
            <!-- тип - дата-время -->
            <DateTime Name="InvoiceDate"/>

            <!-- секция описания свойства -->
            <!-- тип - число с плавающей точкой -->
            <Double Name="CurrencyExchangeRate"/>

            <!-- секция описания свойства -->
            <!-- тип - список файлов -->
            <FileList Name="SignedActScan" />

            <!-- секция описания свойства -->
            <!-- тип - перечисление "SupportedCurrencies", -->
            <!-- описанное в другом сущностье -->
            <Enum Name="Currency" Enum="SupportedCurrencies" />

            <!-- секция описания свойства -->
            <!-- тип - перечисление "InvoiceStatus", -->
            <!-- описанное ниже в этом файле -->
            <Enum Name="InvoiceStatus" Enum="InvoiceStatus" />

            <!-- секция описания вычисляемого свойства -->
            <!-- алгоритм вычисления свойства описан в этой же секции -->
            <Expression Name="EditFormUrl">
               <StringExpression>Connector.ProjectBaseUrl + '/Forms/InvoiceForm/' +
model.Id</StringExpression>
            </Expression>

            <!-- секция описания вычисляемого свойства -->
            <!-- алгоритм вычисления свойства реализован вызовом операции "getInvoiceDueAmount" -->
            <!-- операция getInvoiceDueAmount описана в файле Invoices.ts этого каталога -->
            <Expression Name="DueAmount">
               <DoubleExpression>Invoices.getInvoiceDueAmount(model)</DoubleExpression>
            </Expression>

            <!-- секция описания свойства -->
            <!-- тип - GUID -->
            <Guid Name="ExternalId" />
         </Properties>

         <!-- секция описания обработчиков событий -->
         <EventHandlers>
            <!-- проверка при удалении -->
            <!-- операция invoiceDeleteValidation описана в файле Invoices.ts этого каталога -->
            <DeleteValidation>Invoices.invoiceDeleteValidation()</DeleteValidation>
         </EventHandlers>
      </Entity>

      <!-- секция описания сущности -->
      <!-- этот сущность используется в сущностье "Invoice" -->
      <Entity Name="InvoiceItem" SimpleBusinessObjectTypeId="6a1d3347-e752-45dc-b605-6c8ce7da2cf3"
IsListPropertyItem="true">
         <Properties>
            <!-- секция описания свойства -->
            <!-- тип - целое число -->
            <Integer Name="Index"/>

            <!-- секция описания свойства -->
            <!-- тип - строка -->
            <!-- есть проверка на не пустое значение -->
            <!-- SureNotNull означает, что это поле гарантированно not NULL -->
            <!-- для всех уже сохранённых объектов -->
            <String Name="ItemName" SureNotNull="true">
               <Validations>
                  <NotNull>Please provide a value</NotNull>
               </Validations>
            </String>

            <!-- секция описания свойства -->
            <!-- тип - строка -->
            <!-- есть проверка на не пустое значение -->
            <String Name="Unit" SureNotNull="true">
               <Validations>
                  <NotNull>Please provide a value</NotNull>
               </Validations>
            </String>
```

```
            <!-- секция описания свойства -->
            <!-- тип – число с плавающей точкой -->
            <!-- есть проверка на не пустое значение -->
            <Double Name="Quantity" SureNotNull="true">
               <Validations>
                  <NotNull>Please provide a value</NotNull>
               </Validations>
            </Double>

            <!-- секция описания свойства -->
            <!-- тип – число с плавающей точкой -->
            <!-- есть проверка на не пустое значение -->
            <Double Name="Price" SureNotNull="true">
               <Validations>
                  <NotNull>Please provide a value</NotNull>
               </Validations>
            </Double>

         </Properties>
      </Entity>
   </Entities>

   <!-- секция описания перечислений -->
   <Enumerations>
      <!-- определение перечисления -->
      <Enumeration Name="InvoiceStatus">
         <Members>
            <EnumerationMember Index="0" Name="Draft" />
            <EnumerationMember Index="1" Name="Billed" />
            <EnumerationMember Index="2" Name="Paid" />
            <EnumerationMember Index="3" Name="Cancelled" />
            <EnumerationMember Index="4" Name="Refunded" />
         </Members>
      </Enumeration>
   </Enumerations>
</DomainFile>
```

Операции сущности на языке TypeScript описываются в отдельном файле (рекомендуется располагать в той же директории). В данном случае – это файл Invoices.ts, содержащий функцию «getInvoiceDueAmount», которая используется в вычисляемом свойстве "DueAmount".

```
export function getInvoiceDueAmount(model: Models.Invoice): number | null {
   return model.Properties.Items.reduce((acc, item) => acc += (item.Properties.Price *
item.Properties.Quantity), 0);
}
```

4.2 Описание списков бизнес-объектов конфигурации

4.2.1 Списочное представление

Может содержать произвольные колонки.

Описание списочного представления рекомендуется размещать в том же каталоге, что и описание самого сущности, однако оно может быть размещено и в одном файле с сущностью.

Имя файла имеет вид %ИмяСущности%**List.Domain.xml**

Описание делается в файле формата XML. Пример для списочного описания для объектов сущности «EmployeeList» приведён ниже.

```
<!-- обязательный заголовок файла, в котором описывается список -->
<?xml version="1.0" encoding="utf-8"?>
<DomainFile>
   <!-- секция описания списков -->
   <Lists>
   <!-- секция описания списка -->
      <List Name="EmployeeList">
         <!-- заголовок списка -->
         <Title>Organization Employees</Title>

      <!-- секция колонок списка -->
         <Columns>
            <!-- колонка списка -->
            <!-- описана колонка "FullName", её заголовок, алгоритм вычисления -->
            <String Name="FullName" Title="Full Name">
               <StringExpression>model.Properties.FullName</StringExpression>
            </String>

            <!-- колонка списка -->
            <!-- описана колонка "Title", её заголовок, алгоритм вычисления -->
            <String Name="Title" Title="Title">
               <StringExpression>model.Properties.Title</StringExpression>
            </String>

         </Columns>

         <!-- это список для сущности "Employee" -->
         <Entities>
            <Entity>Employee</Entity>
         </Entities>

         <!-- по умолчанию сортировка будет по этой колонке -->
         <DefaultSortBy>
            <ColumnSort ColumnName="FullName" />
         </DefaultSortBy>

         <SortOptions />
      </List>
   </Lists>
</DomainFile>
```

4.2.2 Описание экранной формы просмотра и редактирования списка бизнес-объектов.

Форма списка автоматически строится Платформой по описанию и формируется в браузере Конечного пользователя бизнес-приложения.

Описание формы рекомендуется размещать в том же каталоге, что и описание самого сущности.

Имя файла имеет вид **%ИмяСущности%.ListView.xml**.

Описание делается в файле формата XML. Пример для формы сущности «EmployeeList» приведён ниже.

В описании формы списка указываются формы, в которых платформа будет открывать элементы списка на просмотр и редактирование (в том числе добавление). В приведённом ниже примере это форма «EmployeeForm». Правила описания формы просмотра и редактирования приведены в п. 4.3 («Описание экранной формы»).

```
<!-- обязательный заголовок файла, в котором описывается список -->
```

```
<?xml version="1.0" encoding="utf-8"?>
<ListView Name="EmployeeListView">
   <!-- в случае пустого списка будет отображаться этот текст -->
   <ListEmptyText>No employees yet</ListEmptyText>
   <!-- в случае неудачного поиска будет отображаться этот текст -->
   <NothingFoundText>Nothing found</NothingFoundText>
   <!-- секция форм просмотра и изменения объектов -->
   <!-- их может быть несколько, т.к. "список" может содержать объекты разных сущностей -->
   <!-- в этом случае при добавлении объекта в список пользователь должен будет выбрать  -->
   <!-- конкретную форму (или, что то же самое, объект какого сущности он хочет добавить  -->
   <ListItemForms>
      <!-- форма, в которой будет открываться объект списка на просмотр и изменение -->
      <ListItemForm EntityId="Employee" FormName="EmployeeForm">
         <Title>Organization Employee</Title>
      </ListItemForm>
   </ListItemForms>
</ListView>
```

4.3 Описание экранной формы редактирования бизнес-объекта

4.3.1 Общие сведения

Описание экранной формы сущности рекомендуется размещать в том же каталоге, что и описание сущности. Имя файла имеет вид **%ИмяСущности%.UxForm.xml**.

Описание делается в файле формата XML. В файле описываются элементы управления, отображение свойств сущности. Само построение формы в браузере Конечного пользователя производится платформой автоматически.

Формы делятся на «обычные» и «встроенные» («inline»). «Встроенные» формы располагаются внутри элементов управления, которые в свою очередь располагаются внутри «обычных» (полноэкранных) форм.

Сущность формы имеет следующие свойства:

Свойство	**Тип**	**Описание**
Name	string	Название формы
Title	string	Заголовок формы
ReadonlyExpression	string?	Вычисляемое выражение, которое возвращает, допускает ли форма редактирование
DeleteButton	bool	Есть ли на форме возможность удалить редактируемый объект
ProjectValidation	bool	Включены ли в форме перекрёстные проверки с другими объектами проекта (вывод результатов валидации объектов, связанных с редактируемым)

Свойство	Тип	Описание
FetchProjectValidationDataServerMethodName	string	Имя серверного метода, возвращающего результаты перекрёстной валидации (см. предыдущее свойство)
ControlsForm	ControlBase[]	Элементы управления формы
Свойства, отвечающие за согласование (устаревшее)		
EnableDocflowBinding	string	Выражение, вычисляемое на клиенте, определяющее, включать ли механизм согласования для объекта редактируемой формы (если это позволяет конфигурация)
Docflow	bool	Включен ли в принципе механизм согласования для объекта редактируемой формы
GetInstanceDocflowServerMethodName	string	Имя серверного метода, который возвращает состояние согласования
Свойства, отвечающие за согласование (используется конфигурацией)		
Approval	bool	аналогично
EnableApprovalBinding	string	аналогично
GetInstanceApprovalServerMethodName	string	аналогично

4.3.2 Элементы управления формы

Элементы управления делятся на «обычные» и «встроенные» («inline»). «Встроенные» элементы управления содержат в своём названии слово «Inline» и используются во «встроенных формах» (т. е. формах, располагающихся внутри «обычных» элементов управления, которые в свою очередь находятся внутри «обычных» форм), «обычные» элементы управления используются в «обычных» формах. Фактически, большинство элементов управления представлено двумя вариантами: «обычным» и «встроенным».

Элементы управления для отображения и редактирования простых типов данных:

Элемент управления	Тип данных для отображения и редактирования
CheckboxEditor, CheckboxEditorInline	Булево
ColoredEnumEditor	Перечисление
DateEditor, DateEditorInline	Дата
DateTimeEditor, DateTimeEditorInline	Дата-время
DoubleEditor, DoubleEditorInline	Двойное число с плавающей точкой
EnumEditor, EnumEditorInline	Перечисление
IntegerEditor, IntegerEditorInline	Целое
MultilineEditor	Многострочная строка
PhoneEditor, PhoneEditorInline	Номер телефона
QuantityEditor	Количество (числовое значение с указанной точностью и единица измерения)
ReadonlyText	Текст для отображения
TextEditor, TextEditorInline	Текст

Элементы управления для отображения и редактирования сложных типов данных:

Элемент управления	Для чего предназначен
FileAttachmentListEditor, FileAttachmentListEditorInline	Для управления списком список файлов, привязанных к объекту
FileAttachmentListViewInline	См. предыдущее, но без возможности добавить или удалить файл (только для просмотра и скачивания)
LinkedListView	Для отображения произвольных списков объектов сущностей
ListEditor	Для отображения значения поля, если поле имеет ссылочный тип. Позволяет редактировать объект сущности.

Элемент управления	**Для чего предназначен**
NumberAndDateEditor	Сдвоенный элемент управления для отображения и ввода значения двух полей: типа «Строка» и типа «Дата» (номер и дата документа).
OrganizationProjectPicker, OrganizationProjectPickerInline	Для отображения и выбора значения поля, если поле должно содержать имя проекта
OrganizationUserPicker	Для отображения и выбора значения поля, если поле должно содержать имя пользователя
PickAndEditListEditor	Для отображения значения поля, если поле имеет ссылочный тип. Позволяет редактировать объект сущности. Позволяет выбирать экземпляр сущности, в том числе с его предварительным добавлением
PickListEditor, PickListEditorInline	Для отображения значения поля, если поле имеет ссылочный тип. Позволяет просматривать и редактировать объект сущности. Позволяет выбирать экземпляр сущности
PickListView, PickListViewInline	Для отображения значения поля, если поле имеет ссылочный тип. Позволяет просматривать объект сущности
QuantityAndUnitEditor	Сдвоенный элемент управления для отображения и ввода значения двух полей, хранящих «количество» и «единицу измерения»
ReadOnlyTextExpression, ReadOnlyTextExpressionInline	Для отображения вычисляемого выражения
ReferencePickerEditor, InlineReferencePicker	Для отображения значения поля, если поле имеет тип ссылка на сущность справочника. Позволяет редактировать объект сущности. Позволяет выбирать экземпляр сущности, в том числе с его предварительным добавлением
StartDateEndDateEditor	Сдвоенный элемент управления для выбора интервала времени
TabularPropertyEditor	Элемент управления для отображения и редактирования табличных данных

4.3.3 Примеры описания экранной формы с простыми элементами управления

```
<!-- обязательный заголовок файла, в котором описывается форма -->
<?xml version="1.0" encoding="utf-8"?>
```

```
<EditForm>
   <!-- секция описания формы начинается с указания имени формы -->
   <Name>EmployeeForm</Name>
   <!-- заголовок формы -->
   <Title>Organization Employee</Title>
   <!-- отображать ли кнопку удаления -->
   <DeleteButton>true</DeleteButton>
   <!-- секция описания элементов управления формы -->
   <Controls>
      <!-- отображение строкового свойства сущности -->
      <TextEditor>
         <Label>Full Name</Label>
         <PropertyName>FullName</PropertyName>
      </TextEditor>

      <!-- отображение строкового поля -->
      <TextEditor>
         <Label>Signer Title</Label>
         <PropertyName>Title</PropertyName>
      </TextEditor>

      <!-- отображение строкового поля -->
      <TextEditor>
         <Label>Last Name and Initial</Label>
         <PropertyName>LastNameAndInitials</PropertyName>
      </TextEditor>

      <!-- отображение строкового поля -->
      <TextEditor>
         <Label>Comment</Label>
         <PropertyName>Comment</PropertyName>
      </TextEditor>

      <!-- отображение булевого поля -->
      <CheckboxEditor>
         <Label>Deactivated</Label>
         <PropertyName>Deactivated</PropertyName>
      </CheckboxEditor>
   </Controls>
</EditForm>
```

4.3.4 Пример описания формы со сложными элементами управления.

4.3.4.1 Форма «ProjectSettings»

Описана форма «ProjectSettings», которая содержит элемент управления, отображающий значение типа «BusinessEntity».

В описании этой формы есть ссылка на списочный сущность «BusinessEntityPickList» (откуда будет вестись выбор конкретного «BusinessEntity»). Списочный сущность «BusinessEntityPickList» описан в отдельном примере (см. 4.3.4.2, «Списочный сущность «BusinessEntityPickList»»).

В описании этой формы есть ссылка на форму просмотра и редактирования «BusinessEntityForm» (которая будет открываться для просмотра выбранного значения «BusinessEntity»). Форма «BusinessEntityForm» описана в отдельном примере (см. 4.3.4.3, «Форма «BusinessEntityForm»»).

Описание формы «ProjectSettings» (файл Project/ProjectSettings.UxForm.xml):

```
<!-- обязательный заголовок файла, в котором описывается форма -->
<?xml version="1.0" encoding="utf-8"?>
<EditForm xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance">
   <!-- секция описания формы начинается с указания имени формы -->
   <Name>ProjectSettings</Name>
   <!-- заголовок формы -->
   <Title>Organization Settings</Title>
   <!-- отображать ли кнопку удаления -->
   <DeleteButton>false</DeleteButton>
   <!-- секция описания элементов управления формы -->
   <Controls>
      <!-- отображение строкового поля -->
      <TextEditor>
         <Label>Brief Name</Label>
         <PropertyName>BriefName</PropertyName>
      </TextEditor>

      <!-- отображение и выбор значения, являющегося ссылкой на справочник -->
      <ReferencePickerEditor>
         <!-- текстовая надпись -->
         <Label>Own Business Entity</Label>
         <!-- свойство сущности "Project", которое нужно отобразить, -->
         <!-- и которое хранит значение (ссылку на справочник) -->
         <PropertyName>OwnBusinessEntity</PropertyName>

         <!-- выбор значения будет производиться -->
         <!-- среди объектов сущности-списка "BusinessEntityPickList" -->
         <!-- сущность-список "BusinessEntityPickList" описан в отдельном подпункте ниже -->
         <PickListViewName>BusinessEntityPickList</PickListViewName>
         <!-- полю формы "OwnBusinessEntity" -->
         <!-- соответствует объект "BusinessEntity" сущности-списка "BusinessEntityPickList" -->
         <PickListViewEntityId>BusinessEntity</PickListViewEntityId>

         <!-- просмотр значения будет производиться в этой «встроенной» форме -->
         <ViewForm>
            <Controls>
               <!-- во «встроенной» форме - и элементы управления «встроенные» -->
               <ReadOnlyTextExpressionInline>
                  <!-- this - это объект сущности "BusinessEntity" -->
                  <!-- отображается его свойство "Title" -->
                  <TextExpression>this.Properties.Title</TextExpression>
               </ReadOnlyTextExpressionInline>
            </Controls>
         </ViewForm>

         <!-- просмотр объекта "BusinessEntity" будет производиться
         <!-- в форме "BusinessEntityForm" сущности "BusinessEntity" -->
         <!-- Эта форма описана в отдельном подпункте ниже -->
         <ListItemForms>
            <ListItemForm EntityId="BusinessEntity" FormName="BusinessEntityForm">
               <Title>New business entity</Title>
            </ListItemForm>
         </ListItemForms>
         <!-- форма для добавления нового значения в справочник - та же, "BusinessEntityForm" -->
         <NewFromListFormName>BusinessEntityForm</NewFromListFormName>
      </ReferencePickerEditor>
   </Controls>
</EditForm>
```

4.3.4.2 Списочный сущность «BusinessEntityPickList»

Описание списочного сущности «BusinessEntityPickList» (файл «BusinessEntities/BusinessEntityPickList.Domain.xml»):

```
<!-- описание сущности списка "BusinessEntityPickList" -->
<?xml version="1.0" encoding="utf-8"?>
<DomainFile>
   <Lists>
      <!-- название сущности -->
      <List Name="BusinessEntityPickList">
```

```
         <!-- заголовок -->
         <Title>Please pick a business entity</Title>

         <!-- колонки списка -->
         <Columns>
            <!-- у этого списка одна колонка "Name" -->
            <String Name="Title" Title="Name">
               <StringExpression>model.Properties.Title</StringExpression>
            </String>
         </Columns>

         <!-- в списке отображаются объекты этого сущности -->
         <Entities>
            <Entity>BusinessEntity</Entity>
         </Entities>
         <!-- отображаются не все объекты, если Объект.Disabled == false, он не будет отображён -->
         <FilterExpression>!model.Properties.Disabled</FilterExpression>

         <!-- сортировка по умолчанию - по полю "Title" -->
         <DefaultSortBy>
            <ColumnSort ColumnName="Title" />
         </DefaultSortBy>
         <SortOptions />

      </List>
   </Lists>
</DomainFile>
```

4.3.4.3 Форма «BusinessEntityForm»

Описание формы «BusinessEntityForm» (файл «BusinessEntities/BusinessEntity.UxForm.xml»):

```
<!-- обязательный заголовок файла, в котором описывается форма -->
<?xml version="1.0" encoding="utf-8"?>
<EditForm>
   <!-- секция описания формы начинается с указания имени формы -->
   <Name>BusinessEntityForm</Name>
   <!-- заголовок формы -->
   <Title>Business Entity</Title>
   <!-- отображать ли кнопку удаления -->
   <DeleteButton>true</DeleteButton>
   <!-- секция описания элементов управления формы -->
   <Controls>
      <!-- отображение строкового свойства сущности -->
      <TextEditor>
         <Label>Name</Label>
         <PropertyName>Title</PropertyName>
      </TextEditor>

      <!-- отображение булева свойства сущности -->
      <CheckboxEditor>
         <Label>This is an individual</Label>
         <PropertyName>IsAnIndividual</PropertyName>
      </CheckboxEditor>

      <!-- отображение строкового свойства сущности -->
      <TextEditor>
         <PropertyName>FullName</PropertyName>
         <Label>Full Name</Label>
      </TextEditor>

      <!-- отображение булева свойства сущности -->
      <CheckboxEditor>
         <Label>No Longer Used</Label>
         <PropertyName>Disabled</PropertyName>
      </CheckboxEditor>
   </Controls>
</EditForm>
```

4.4 Описание обработчиков событий на языке TypeScript

Платформа позволяет написать обработчики на следующие события:

Событие	**Пояснение**
OnCreate	Код, который изменяет свойства нового объекта, который загружается в клиентское приложение для редактирования, но еще не был сохранен в хранилище
DeleteValidation	Код, вызываемый для проведения проверки допустимости удаления объекта, и позволяющий отказать в попытке удалить объект
BeforeCreate	Перед созданием объекта
AfterCreate	После создания объекта
BeforeUpdate	Перед обновлением объекта
AfterUpdate	После обновления объекта
BeforeDelete	Перед удалением объекта
AfterDelete	После удаления объекта

Обработчик на языке TypeScript может оперировать любыми бизнес-объектами внутри экземпляра конфигурации, используя объектную модель предметной области на языке TypeScript, генерируемую автоматически на основе описаний сущностей и функций на языке TypeScript, на которые ссылаются описания сущностей.

4.5 Описание шаблона печатной формы документа в формате платформы

Печатные формы документов описываются в формате XML и состоят из различных секций документа, которые в ходе процедуры генерации печатной формы документа образуют документ в формате Microsoft Word (.docx). Для формирования печатной формы можно использовать любую информацию, доступную в экземпляре конфигурации.

Свойства формы:

Свойство	**Тип**	**Описание**
Name	string	Имя формы
EntityName	string	Для какого сущности данных эта форма

Свойство	Тип	Описание
ModelExpression	ObjectExpression	Вычисляемое выражение, возвращающее объект произвольной структуры, содержащий все необходимые данные для формирования печатной формы документа
FileName	StringExpression	Алгоритм, по которому создаётся имя файла (результата работы печатной формы)
FontSettings	FontSettings	Шрифт формы по умолчанию
SpaceBeforeSectionExpression	DoubleExpression	Вертикальный отступ перед секцией печатной формы в сантиметрах
DefaultLineSpacingExpression	DoubleExpression	Межстрочный интервал текста внутри секции печатной формы в линиях
FontFamilyExpression	StringExpression	Выражение, возвращающее имя семейства шрифтов секции печатной формы
FontSizeExpression	DoubleExpression	Выражение, возвращающее размер шрифта формы
Margins	PageMargins	Отступы печатной формы от краев страницы
Sections	PrintFormSection[]	Секции формы
Header	PrintFormSection[]	Секции, являющиеся «шапкой» (PrintFormSection)
FirstHeader	PrintFormSection[]	Секции, являющиеся «шапкой» для первой секции (PrintFormSection)
Footer	PrintFormSection[]	Секции, являющиеся «подвалом» (PrintFormSection)
FirstFooter	PrintFormSection[]	Секции, являющиеся «подвалом» для первой секции (PrintFormSection)

Наследники сущности PrintFormSection (виды секций):

Сущность	Описание
CityAndDateSection	Секция, выводящая город составления и дату документа

Сущность	Описание
GridSection	Секция, позволяющая организовать (сверстать) содержимое документа в виде статической таблицы. Является контейнером для других секций.
GridTableSection	Секция, позволяющая организовать (сверстать) содержимое документа в виде таблицы со статическим заголовком и динамическим телом таблицы. Является контейнером для других секций.
HeadingSection	Секция, выводящая заголовок и подзаголовок документа
NumberAndDateSection	Секция, выводящая дату и номер документа
PageBlockSection	Секция, вставляющая в документ новый блок страниц с независимой настройкой ориентации, полей и колонтитулов
PageBreakSection	Секция, вставляющая принудительный разрыв страницы, но не начинающая новый блок страниц документа
RepeatSection	Секция, получающая в качестве параметра массив произвольных объектов и добавляющая в документ для каждого такого объекта копию вложенных в нее секций
StartDateEndDateSection	Секция, выводящая блок с датой начала и датой окончания периода
TableSection	Секция, выводящая таблицу со статическим заголовком и динамическим телом таблицы
TextSection	Секция, выводящая произвольный форматированный текст
TitleValueHintSection	Секция, выводящая в документ текст, визуально оформленный как машинописное заполнение формы, предназначенной для заполнения вручную, с возможностью добавления подстрочных подсказок

Свойства сущности PageMargins:

Свойство	Тип	Описание
Inside	DoubleExpression	Внутреннее поле
Top	DoubleExpression	Верхнее поле
Outside	DoubleExpression	Вешнее поле
Bottom	DoubleExpression	Внутреннее поле

Свойства сущности FontSettings

Свойство	Тип	Описание
FontFamily	string	Название шрифта
FontSize	double	Размер шрифта

Пример описания печатной формы приведён ниже.

```
<!-- обязательный заголовок файла, в котором описывается форма -->
<?xml version="1.0" encoding="utf-8"?>
<!-- имя формы, для какого сущности данных эта форма -->
<PrintForm Name="LayoutOfAxesActForm2018" EntityName="LayoutOfAxesAct">
   <!-- алгоритм формирования имени файла -->
   <FileName>'АРООКС' + (model.Properties.DocumentNumber ? ' №' + model.Properties.DocumentNumber :
'') + (model.Properties.DocumentDate ? ' от ' + formatDate(model.Properties.DocumentDate) :
'')</FileName>
   <!-- шрифт -->
   <FontSettings FontFamily="Arial" FontSize="9" />
   <FontFamilyExpression>Common.getProjectFontFamily()</FontFamilyExpression>
   <FontSizeExpression>Common.getProjectFontSize()</FontSizeExpression>
   <!-- поля -->
   <Margins>
      <Inside>Common.getProjectPageMargins().marginInside</Inside>
      <Top>Common.getProjectPageMargins().marginTop</Top>
      <Outside>Common.getProjectPageMargins().marginOutside</Outside>
      <Bottom>Common.getProjectPageMargins().marginBottom</Bottom>
   </Margins>
   <!-- секции -->
   <Sections>
      <!-- секция TitleValueHint -->
      <TitleValueHint AllowInlineTitle="!!Project.Properties.AllowInlineTitles" TitleOnly="false"
Visible="true" BoldTitle="Common.boldTitles()" BoldValue="Common.boldContent()">
         <Title>'Объект строительства'</Title>
         <Value>Project.Properties.FullName</Value>
         <Hint>'(Наименование объекта строительства)'</Hint>
         <TitlePostfix>null</TitlePostfix>
      </TitleValueHint>

      <!-- секция Heading -->
      <Heading Bold="Common.boldTitles()">
         <Heading>'АКТ'</Heading>
         <Subheading>'разбивки осей объекта капитального строительства на местности'</Subheading>
      </Heading>

      <!-- секция NumberAndDate -->
      <NumberAndDate>
         <Number>model.Properties.DocumentNumber</Number>
         <Date>model.Properties.DocumentDate</Date>
         <DateHint>'(дата составления акта)'</DateHint>
      </NumberAndDate>

      <!-- секция Repeat -->
      <Repeat>
         <ArrayExpression Array="model.Properties.ResponsiblePersonGroups">
            <TitleValueHint AllowInlineTitle="false" TitleOnly="false" Visible="true"
BoldTitle="Common.boldTitles()" BoldValue="Common.boldContent()">
               <Title>$item.Properties.RoleTitle</Title>
               <Value>PrintForms.responsiblePersonsTextExpression($item)</Value>
               <Hint>PrintForms.responsiblePersonsHintExpression($item)</Hint>
               <TitlePostfix>null</TitlePostfix>
            </TitleValueHint>
         </ArrayExpression>
      </Repeat>

      <!-- секция TitleValueHint -->
      <TitleValueHint AllowInlineTitle="true" TitleOnly="false" Visible="true"
BoldTitle="Common.boldTitles()" BoldValue="Common.boldContent()">
         <Title>'составили настоящий акт о том, что произведена разбивка в натуре осей'</Title>
         <Value>model.Properties.Axes</Value>
         <Hint>''</Hint>
         <TitlePostfix>null</TitlePostfix>
      </TitleValueHint>

      <!-- секция Text -->
      <Text Visible="Common.boldTitles()">
         <Chunk Bold="true">'Акт составлен в '</Chunk>
         <Chunk Bold="false" Underline="true">model.Properties.NumberOfCopies ?
formatInteger(model.Properties.NumberOfCopies) : '____'</Chunk>
         <Chunk Bold="true">' экземплярах'</Chunk>
```

```
      </Text>

      <!-- секция TitleValueHint -->
      <TitleValueHint AllowInlineTitle="true" TitleOnly="false" Visible="true"
BoldTitle="Common.boldTitles()" BoldValue="Common.boldContent()">
         <Title>'Приложения'</Title>
         <Value>model.Properties.Attachments</Value>
         <Hint>'(схема закрепления осей)'</Hint>
         <TitlePostfix>null</TitlePostfix>
      </TitleValueHint>

      <!-- секция Repeat -->
      <Repeat>
         <ArrayExpression Array="model.Properties.ResponsiblePersonGroups">
            <TitleValueHint AllowInlineTitle="false" TitleOnly="false" Visible="true"
BoldTitle="Common.boldTitles()" BoldValue="Common.boldContent()">
               <Title>$item.Properties.RoleTitle</Title>
               <Value>PrintForms.responsiblePersonSignaturesExpression2018($item)</Value>
               <Hint>'(фамилия, инициалы, подпись)'</Hint>
               <TitlePostfix>null</TitlePostfix>
            </TitleValueHint>
         </ArrayExpression>
      </Repeat>

   </Sections>
</PrintForm>
```

4.6 Описание шаблона печатной формы документа в формате «Word»

Шаблоном печатной формы в формате «Word» служит любой файл формата «Word» версии 2007 или новее (формат DOCX). В тексте формы допускается указание переменных (в двойных фигурных скобках), которые будут вычислены Платформой при формировании печатной формы.

Лицензиат:		ИНН {{Inn}} КПП {{Kpp}}
		{{BriefName}}
		{{LegalAddress}}

№	Наименование работ, услуг	Кол-во	Ед.	Цена, руб.	Сумма, руб.
1	{{LicenseDetails}}	1	шт.	{{LicenseFeeNumber}}	{{LicenseFeeNumber}}
				Итого:	{{LicenseFeeNumber}}
				НДС не предусмотрен	
Всего наименований 1, на сумму {{LicenseFeeNumber}} рублей.					

4.7 Описание меню проекта

Меню проекта описывается в файле src/Project/ProjectMenu.xml.

Элементы меню могут быть следующих типов:

Тип	**Описание**	**Свойства**
UrlProjectMenuItem	Название сущности, форма которого будет открыт при выборе пункта меню	string Url: название сущности (полный путь в каталоге проекта) bool External: если true, то это внешняя ссылка, а не ссылка внутри платформы
SubmenuProjectMenuItem	Подменю, содержит другие элементы меню	ProjectMenuItem[] Children: массив элементов меню
AsyncDownloadProjectMenuItem	Предназначен для запуска фоновой задачи, как правило, для формирования больших архивов. Задача, запущенная таким пунктом меню, ставится в очередь и будет выполнена в фоновом режиме	string MethodName: операция, которую нужно запустить

Ниже приведён пример описания меню.

```
<!-- обязательный заголовок файла, в котором описывается меню -->
<?xml version="1.0" encoding="utf-8"?>
```

```
<ProjectMenu>
   <!-- элементы меню -->
   <Items>
      <!-- элемент меню: вызов формы списка сущности "Invoice" -->
      <Url Title="Invoices" Url="/Lists/InvoiceListView/InvoiceList" />
      <!-- элемент меню: вызов формы списка сущности "BusinessEntity" -->
      <Url Title="Business Entities" Url="/Lists/BusinessEntityListView/BusinessEntityList" />
      <!-- элемент меню: вызов формы списка сущности "Payment" -->
      <Url Title="Payments" Url="/Lists/PaymentListView/PaymentList" />
      <!-- Подменю -->
      <Submenu Title="Settings">
         <Children>
            <!-- элемент меню: вызов формы списка сущности "Employee" -->
            <Url Title="Employees" Url="/Lists/EmployeeListView/EmployeeList" />
            <!-- элемент меню: вызов формы "ProjectSettings" -->
            <Url Title="My Organization" Url="/Forms/ProjectSettings"/>
         </Children>
      </Submenu>
   </Items>
</ProjectMenu>
```

4.8 Серверные методы

Платформа позволяет выполнять код не только на клиенте, но и в серверном окружении. Выполнить можно не произвольный код, а только заранее написанный метод. Они описываются в файле src/Project/Project.ServerMethods.xml.

Пример таких описаний:

```
<!-- обязательный заголовок файла, в котором описываются серверные методы -->
<?xml version="1.0" encoding="utf-8"?>
<ServerMethods>
   <!-- секция описания методов -->
   <Methods>
      <!-- этот метод является операцией конфигурации -->
      <!-- и описан в файле Project/Common.ts -->
      <ServerMethod Name="GetProjectStatus">Common.getProjectStatus()</ServerMethod>
      <!-- этот метод является операцией сущности Invoices -->
      <!-- и описан в файле Invoices/Invoices.ts -->
      <ServerMethod Name="GetInvoiceCopy">Invoices.getInvoiceCopy(model)</ServerMethod>
      <!-- этот метод является операцией сущности Payments -->
      <!-- и описан в файле Payments/Payments.ts -->
      <ServerMethod Name="GetPaymentCopy">Payments.getPaymentCopy(model)</ServerMethod>
   </Methods>
</ServerMethods>
```

4.9 Одностраничные приложения

Платформа позволяет формировать одностраничное браузерное web-приложение на платформе React. Это приложение описывается в каталоге файле client/src/.

Главный файл описания – исполняемый на клиенте client/src/index.tsx, в котором на языке TypeScript описаны web-страницы.

Каждое приложение размещается отдельно в своём подкаталоге и описывается файлом с расширением tsx. Например, «client/src/Start/» и файл

описания «StartPage.tsx», «client/src/Reports/» и файл описания «ReportsPage.tsx».

Пример приведён ниже:

```
/// <reference types="@hardroller/configuration-core/src/hardroller" />

import * as React from "react";

import { Guid } from "@hardroller/core";

import { ApplicationModelProject, AsyncDownloadConnector, ProjectConnector } from
"@hardroller/client-connector-core";

import StartPage from "./Start/StartPage";
import ReportsPage from "./Reports/ReportsPage";

class CustomPageFactory {
   createPage(
      pageName: string,
      instanceId: Guid | null,
      applicationModel: ApplicationModelProject,
      projectConnector: ProjectConnector,
      reloadApplicationModel: () => void,
      asyncDownloadConnector: AsyncDownloadConnector
   ): React.ReactNode {

      /* загрузка страницы "StartPage" */
      if (pageName === "Start") {
         /* чтобы мы попали сюда, в меню должна иметь вид ".../Application/.../Pages/Start"
         return () => <StartPage
            pageName={pageName}
            applicationModel={applicationModel}
            instanceId={instanceId}
            projectConnector={projectConnector}
            asyncDownloadConnector={asyncDownloadConnector}
            reloadApplicationModel={reloadApplicationModel}
         />;
      }

      /* загрузка страницы "ReportsPage" */
      else if (pageName === "Reports") {
         /* чтобы мы попали сюда, в меню должна иметь вид ".../Application/.../Pages/Reports"
         return () => <ReportsPage
            pageName={pageName}
            applicationModel={applicationModel}
            instanceId={instanceId}
            projectConnector={projectConnector}
            asyncDownloadConnector={asyncDownloadConnector}
            reloadApplicationModel={reloadApplicationModel}
         />;
      } else {
         return null;
      }
   }
}

export const customPageFactory = new CustomPageFactory();
```

Клиентский компонент конфигурации объявляет сущность CustomPageFactory с методом createPage, через который Платформа передает браузерному приложению следующие параметры:

pageName – имя страницы (приложения) которую запросил пользователь;

applicationModel – объект сущности ApplicationModelProject, содержащий необходимую информацию о текущем пользователе, организации и проекте;

instanceId – идентификатор экземпляра сущности, с которым будет работать страница (приложение) - не обязательно;

projectConnector – объект сущности ProjectConnector с методами для вызова API Платформы для клиентского приложения (он же используется стандартными формами редактирования);

asyncDownloadConnector – объект сущности asyncDownloadConnector с методами для запуска фоновых задач;

reloadApplicationModel – функция, вызывающая принудительную перезагрузку клиентским приложением платформы applicationModel, загруженной изначально (применяется, если страница (приложение) вызывала методы API, вносящие изменения в модель данных).

Метод должен вернуть корректно сформированный компонент React.

4.10 Параметры конфигурации

Параметры всей конфигурации в целом описываются в файле src/Project/Project.Domain.xml.

Конфигурация имеет следующие свойства:

Свойство	**Тип**	**Описание**
CompositeObjectTypeId	GUID	Уникальный идентификатор конфигурации
Name	string	Название (для разработчиков)
Title	string	Название (для пользователей)
StartUrl	string	Ссылка на стартовую страницу конфигурации
UserRoles	UserRole[]	Список ролей пользователей, определённых для конфигурации

Кроме этого, конфигурация может иметь произвольное количество свойств, описанных дополнительно.

Пример приведён ниже:

```
<!-- обязательный заголовок файла, в котором описывается конфигурация -->
<?xml version="1.0" encoding="utf-8"?>
<DomainFile CompositeObjectTypeId="a597f7e3-7620-42d2-846d-08e373c64507" Name="Ada" Title="AdaCRM"
StartUrl="/Lists/InvoiceListView/InvoiceList">
   <!-- дополнительные свойства -->
   <Properties>
      <String Name="BriefName">
         <Validations>
            <NotNull>Please provide a value</NotNull>
         </Validations>
      </String>
```

```
      <Reference Name="OwnBusinessEntity" ChildEntity="BusinessEntity">
         <Validations>
            <NotNull>Please provide a value</NotNull>
         </Validations>
      </Reference>

      <DateTime Name="DateTimeCreated" />
   </Properties>

   <Scripts>
      <ScriptReference FileName="bundle.js" />
   </Scripts>

   <!-- пользовательские роли -->
   <UserRoles>
      <UserRole Name="admin" Title="Administrator"/>
      <UserRole Name="manager" Title="Manager"/>
   </UserRoles>
</DomainFile>
```