Д.Э. Гаврилина

Дизайнер объектных онтологий Phlox

!!!TODO!!!
УДК 519.716

Д.Э. Гаврилина. Дизайнер объектных онтологий Phlox — Серия: Дискретная математика и информатика. Вып. 21. — Иркутск: Изд-во Ирк. гос. пед. ун-та., 2024. — 83 с.

Описывается дизайнер моделей Phlox, являющийся основным инструментом платформы Ontobox и служащий для визуализации построения *объектных онтологий*.

Содержание

Введение

В данном работе рассматривается дизайнер моделей Phlox, являющийся основным инструментом платформы Ontobox и служащий для визуализации построения *объектных онтологий*. Phlox позволяет создавать высокоуровневые модели объектов и их взаимодействия без применения традиционного программирования. Здесь представлены основные принципы работы с этим инструментом. Эта книга будет полезна как начинающим разработчикам, так и опытным специалистам, желающим расширить свой арсенал инструментов для моделирования объектов.

Объектные онтологии — простые логические системы, служащие для описания предметных областей и эквивалентные по выразительности объектным моделям в программировании. Отличие онтологий состоит в том, что они представляют собой логический формализм со строгой декларативной семантикой, поэтому могут использоваться не только для хранения данных и их использования в процедурном стиле, но и для логического анализа, бизнес-аналитики, автоматического решения интеллектуальных задач, то есть быть основой для искусственного интеллекта, основанного на логике. В объектных онтологиях базовые объектные описания также обогащены дополнительными возможностями, включая моделирование бизнес-процессов и управление жизненным циклом объектов. Концепция развития логико-вероятностного ИИ, основанного на объектных онтологиях, и построенная на этой базе платформа разработки интеллектуальных приложений представлены в white paper [4].

Теоретические основы объектных онтологий разработаны в исследованиях [6, 14–17] — как имплементация идей семантического моделирования [10, 11, 13].

Ontobox — платформа для разработки приложений, в которой реализован *декларативный low-code* — технология разработки через логические описания предметной области (ранее – платформа *bSystem*). Декларативный low-code увеличивает производительность разработчика на порядок по сравнению с традиционным программированием. Визуализация разработки таких описаний с помощью дизайнера моделей Phlox является ядром этой технологии.

Сами объектные онтологии формируются на платформе Ontobox как независимые (микро)сервисы, взаимодействующие друг с другом через строго специфицированный язык взаимодействия. Микро-

сервисный подход [21] обеспечивает масштабируемость технологии, позволяет строить приложения любой сложности и объемов.

Традиционно low-code определяется как метод проектирования и разработки приложений с помощью интуитивно понятных графических инструментов и встроенных функций, которые снижают традиционные (профессиональные) требования к написанию кода и повышают на порядок эффективность разработки несложных приложений.

Декларативный low-code платформы Ontobox принципиально отличается от традиционных low-code систем (см. Elma365 [5], Appian [7], OutSystems [20] и многие другие), основанных на процедурных диаграммных описаниях (в стиле спецификации [9]). Имея иную природу, декларативный low-code, в частности, позволяет избавиться от ключевых проблем традиционного low-code (перечисленных, например, в [22] и [8]).

Несмотря на проблемы, low-code привлекает пристальное внимание со стороны бизнеса и менеджмента крупных компаний [12], поскольку обеспечивает реакцию бизнеса на изменяющиеся обстоятельства и недостаток квалифицированного персонала. Анализ применения low-code в бизнес-практике, его плюсы и минусы представлены в аналитическом обзоре [18].

В отличие от традиционного подхода декларативный low-code:

- ориентирован на разработку масштабируемых приложений любой сложности, интегрирующих операционный уровень с бизнес-аналитикой и искусственным интеллектом, и
- ориентирован на поддержку работы профессиональных разработчиков информационных систем и приложений, обладающих системным мышлением и способных решать задачи высокой сложности.

В контексте декларативного low-code визуализация занимает центральное место, становясь заменой традиционному текстовому программированию. В качестве основы Phlox мы разработали метод визуализации, основанный на так называемых объектных электронных таблицах, что обеспечивает взаимодействие с объектными моделями в стиле Excel. Многолетний опыт использования Excel и его неувядающая популярность [19] свидетельствуют об исключительной точности технологических решений, на которых он основан. Это, в част-

ности, касается и табличного представления данных. При реализации Phlox было принято решение о табличном представлении логических описаний, представленных в онтологиях. Объектные онтологии являются структурно более сложной средой, чем арифметическая среда Excel. Тем не менее довольно большой на данный момент опыт эксплуатации редактора Phlox в рамках разработки приложений показывает, что и в случае онтологий визуализация данных через использование таблиц является чрезвычайно эффективным инструментом.

Ontobox обеспечивает пользователю доступ к визуальному интерфейсу объектной модели с момента её создания («приложение по умолчанию»), что экономит время на формирование интерфейсов, когда это не требуется, и в то же время значительно упрощает процесс разработки приложений. Он обеспечивает решение обычных задач, характерных для электронных таблиц, таких как ведение реестров, управление результатами экспериментов, управление кадрами, ресурсами, списками и так далее. В качестве front-end решения можно использовать встроенный интерфейс объектных электронных таблиц. Методы, определенные в модели, также можно активировать из таблиц, что позволяет настроить таблицу в соответствии с терминологией и функциональностью конкретной задачи (например, в методах можно объединить несколько операций в одной транзакции, добавить статистический анализ и бизнес-аналитику, генерировать печатные формы документов, привязку к мессенджерам, интернету вещей и другим источникам информации с автоматическим обновлением данных и т.д.).

При регистрации на платформе Ontobox пользователю предоставляется модель по умолчанию, с которой он может сразу работать. В случае работы с несколькими задачами или при построении сложного приложения с микросервисной архитектурой пользователь имеет техническую возможность создавать любое количество моделей. Каждая модель функционирует независимо от других.

Данная книга посвящена подробному описанию основных возможностей дизайнера объектных онтологий Phlox. Концептуальные основы построения дизайнера объектных онтологий Phlox представлены в работе [3].

1. Моделирование с помощью объектных онтологий

В центре методологии декларативного low-code находятся объектные модели. Именно модели интегрируют разные уровни разработки в единое пространство. Традиционные объектно-ориентированные модели основаны на сбалансированном сочетании декларативного (классы, объекты, свойства) и процедурного (методы) уровней. Модели Ontobox также сохраняют баланс между декларативностью и процедурностью. Но они имеют и важные дополнения в отличие от «программистских» ОО-моделей:

1. Объектные онтологии (модели) Ontobox — долгоживущие, автономные и реплицируемые. Кроме самих объектов, они хранят метаописания, а также определения методов и бизнес-процессов.

2. Объектные онтологии Ontobox, в отличие от традиционных ОО-моделей, умеют напрямую работать с процессами, которые разворачиваются во времени. Объектное моделирование бизнес-процессов превращает Ontobox в эффективный инструмент построения управленческих систем — CRM, ERP, HRM с возможностью быстрой реализации уникальных и нестандартных бизнес-процессов, их модернизации и развития с использованием визуальных средств. Расширенные модели Ontobox умеют напрямую моделировать диаграммы BPMN.

3. Декларативный уровень объектной модели Ontobox представлен в виде логического метаописания. Метаописание включает классы объектов, виды связей между объектами, поля, а также декларативную информацию о методах — имя метода, имена и типы параметров и т.д. Метаописание — открытая логическая система, доступная для ИИ и бизнес-аналитики. Оно включает почти всю содержательную информацию о модели. Процедурная составляющая модели состоит только из тел методов.

4. Объектные модели Ontobox доступны через Web API. Наличие API, автономность и реплицируемость моделей превращает их в микросервисы — с масштабируемостью и другими плюсами данной архитектуры.

Сочетание декларативного и процедурного уровней объектной модели позволяет строить декларативный low-code как совокупность

описательного и процедурного стилей разработки. Это и реализуется через редактор Phlox, действующий на основе объектных электронных таблиц.

Объектные электронные таблицы — декларативный графический интерфейс к объектной онтологии, наше ноу-хау. Таблицы поддерживают работу с классами, объектами, полями — в стиле «продвинутого Excel». Они позволяют как строить метаописания моделей, так и работать с конкретными данными — аналогично работе в Excel, Google Sheets. Но есть три отличия:

Во-первых, наши таблицы работают с объектными онтологиями — намного более выразительной системой данных, чем реляционные.

Во-вторых, над моделями можно строить веб-сервисы произвольной сложности, используя для этого собственный конструктор интерфейсов Ontobox или внешнюю front-end среду, которая взаимодействует с моделью через Web API.

Наконец, в третьих, над объектными онтологиями можно надстраивать системы искусственного интеллекта, использующие для своего обучения факты из онтологий. Особенно перспективным представляется использование логико-вероятностных методов ИИ [1, 2], поскольку и объектные онтологии, и логико-вероятностный вывод имеют общую теоретическую основу в виде семантического моделирования.

Отметим, что в Ontobox можно использовать и традиционные для low-code *BPM-диаграммы* в качестве процедурного графического интерфейса к объектной онтологии. В отличие от традиционных low-code платформ, в Ontobox диаграммы — это виртуальные сущности, которые автоматически генерируются в нужный момент по текущему состоянию модели.

На основе визуального моделирования в редакторе Phlox платформа Ontobox автоматически синтезирует («пишет») исполняемый код (high-code), непосредственно формирующий разрабатываемое приложение. Поскольку, как известно, средств low-code редко хватает для реализации сложных проектов, Ontobox предоставляет разработчику среду программирования, позволяя вручную дописывать то, что невозможно представить визуальными средствами (например, в виде high-code библиотек). Это обеспечивает бесшовную среду low-code и high-code разработки, что часто является проблемой для традиционных low-code систем.

Такая архитектура Ontobox предъявляет серьезные требования к high-code, то есть, к языку программирования. Он должен содержать все механизмы, необходимые для отображения low-code, уметь напрямую работать с объектными онтологиями Ontobox, служить языком запросов к объектным онтологиям, а также обеспечивать простое кодирование в стиле low-code. Эти требования привели к необходимости создания собственного языка программирования — Libretto, который был разработан А.А.Малых и А.В.Манциводой.

Единая среда разработки для low-code и high-code — одно из ключевых преимуществ нашего подхода. Эти два уровня могут свободно перемешиваться в одном приложении, задавая оптимальную стратегию разработки. Например, структурная часть работы может быть проведена аналитиками-не-программистами в объектных электронных таблицах, а затем доведена до практики профессиональными разработчиками на уровне high-code.

2. Объектная онтология (модель)

Обратимся теперь к подробному описанию работы редактора объектных онтологий Phlox. Для краткости объектные онтологии будем также называть моделями, частным случаем которых они являются.

Сам Phlox работает как веб-приложение, которое в рамках одной страницы управляет ровно одной моделью. Если пользователь взаимодействует с несколькими моделями, он имеет возможность переключаться между ними через меню моделей, расположенное в правой части основной панели (рис. 1).

Рис. 1: Меню моделей

2.1. Создание модели

Для создания новой модели нужно:

1. зайти в меню моделей в правой части основной панели интерфейса Phlox и выбрать последнюю опцию «All models»;

2. в открывшемся окне со списком моделей нужно нажать на «+» на панели справа вверху (рис. 3);

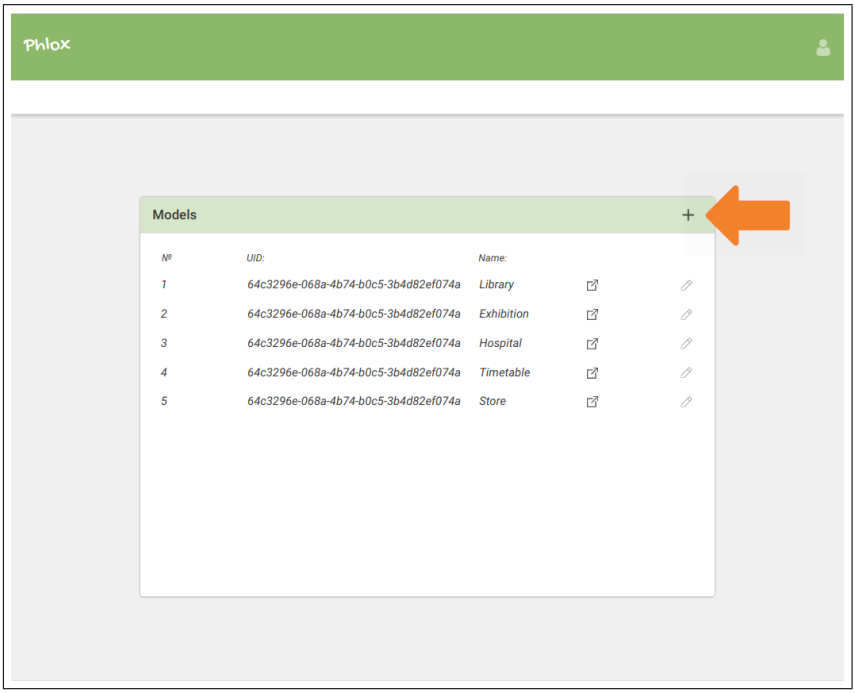

Рис. 2: Список моделей

3. в появившемся окне ввода ввести читаемое имя модели (используется только для удобства пользователя) и нажать на кнопку «Ok»;

4. созданная модель появится в списке моделей.

2.2. Редактирование модели

Для редактирования параметров модели необходимо:

1. зайти в меню моделей в правой части основной панели интерфейса Phlox и выбрать последнюю опцию «All models»;

2. в открывшемся окне со списком моделей нужно нажать на иконку карандашика у модели, которую нужно отредактировать (рис. 3);

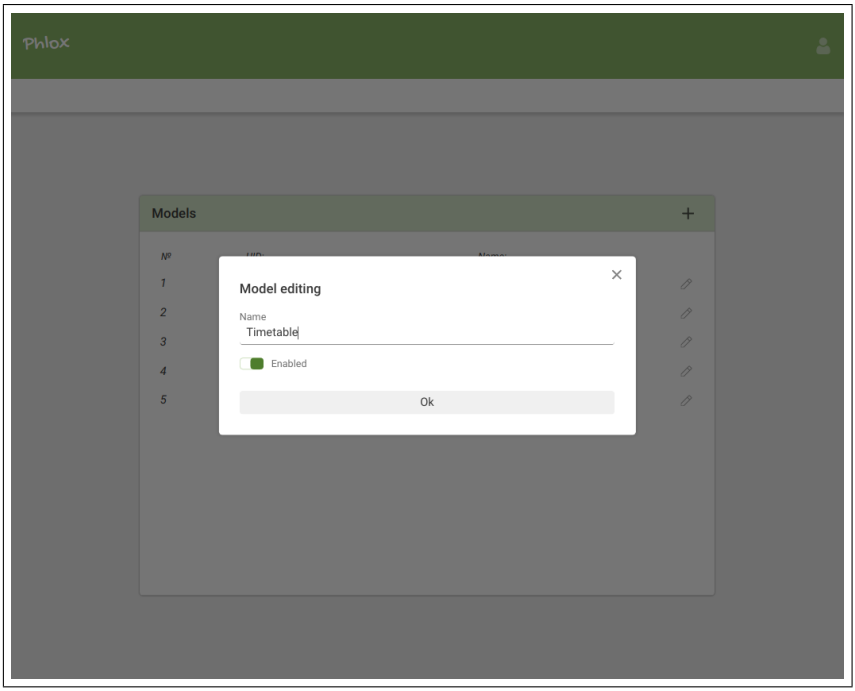

Рис. 3: Окно редактирования модели

3. в открывшемся окне можно сделать необходимые корректировки. В настоящее время это — изменение читабельного имени модели и активация/деактивация модели.

3. Управление пакетами

Основные операции управления пакетами модели проводятся в начальном окне управления моделью. Если во Phlox не открыто ни одной вкладки, то начальное окно управления моделью открывается автоматически.

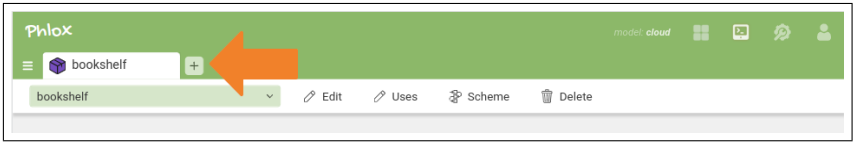

Рис. 4: Открытие начального окна управления моделью

Если в интерфейсе Phlox имеются открытые вкладки, то для открытия начального окна управления моделью нужно нажать на кнопку «+» в основной панели (рис. 4).

В начальном окне управления моделью осуществляются следующие операции:

1. создание пакета;
2. редактирование/удаление пакета;
3. открытие вкладки пакета;
4. выбор пакета и открытие списка форм документов/структур/трейтов/юнионов пакета;
5. создание новой формы/структуры/трейта/юниона (в зависимости от выбранного типа);
6. редактирование/удаление формы/структуры/трейта/юниона (в зависимости от выбранного типа);
7. открытие вкладки формы документов;
8. переключение отображения списка формы/структуры/трейта/юниона.

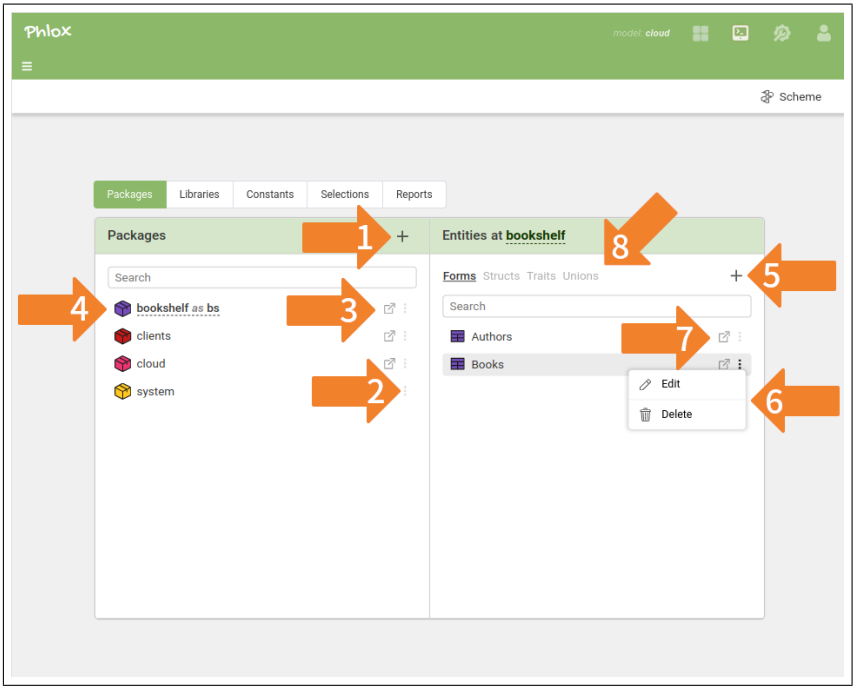

Рис. 5: Начальное окно управления моделью

3.1. Создание пакета

При первом вхождении во Phlox или при открытии новой модели открывается начальное окно управления моделью. Для создания нового пакета нужно:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);

2. в открывшемся окне управления пакетами нажать на кнопку «+» создания нового пакета в левом столбце;

3. в открывшихся полях ввода ввести идентификатор (name) и читаемое название (title), комментарии (comments), короткое имя (alias) пакета (рис. 6);

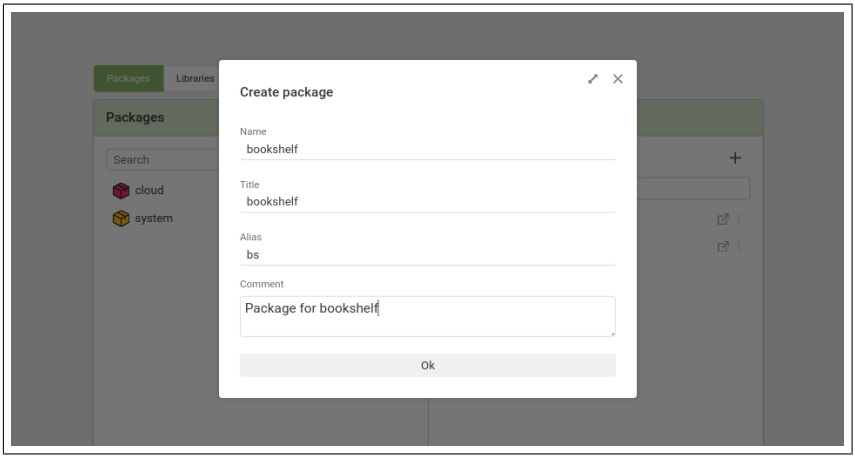

Рис. 6: Окно создания пакета

4. создать пакет, нажав кнопку «Ok».

Вновь созданный пакет появится в списке пакетов. Если пакет с данным идентификатором уже существует, то будет выдана ошибка.

3.2. Редактирование пакета

Для редактирования раннее созданного пакета необходимо:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);
2. в открывшемся окне управления пакетами выбрать нужный пакет, нажать на кнопку «⋮», затем выбрать функцию «Edit»;
3. в полях всплывающего окна «Package edititng» отредактировать идентификатор пакета (name) и/или его читаемое имя (title);
4. завершить редактирование пакета, нажав кнопку «Ok».

Измененный пакет отобразится в списке пакетов. Если при редактировании был указан идентификатор, который уже существует, то будет выдана ошибка.

3.3. Удаление пакета

Для удаления раннее созданного пакета требуется:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);
2. в открывшемся окне управления пакетами выбрать нужный пакет, нажать на кнопку «⋮», затем выбрать функцию «Delete»;

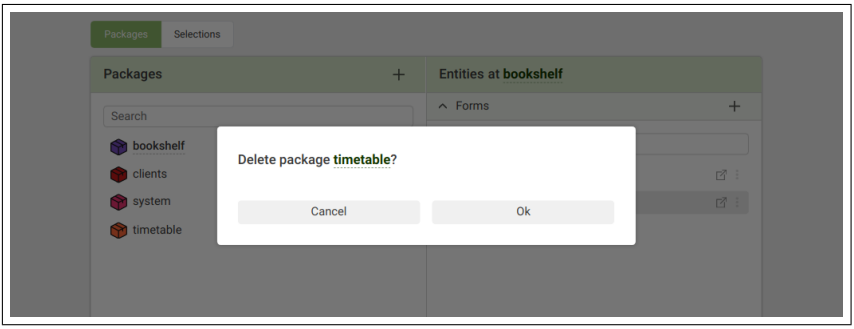

Рис. 7: Окно подтверждения удаления пакета

3. подтвердить удаление в модальном окне, нажав кнопку «Ok» (рис. 7).

После подтверждения удаления пакет будет удален. Во всех списках удаленный пакет перестанет отображаться. Если была открыта вкладка с данным пакетом, то она автоматически закроется.

3.4. Отображение пакета

Для детального анализа содержимого пакета доступна специальная вкладка, предназначенная для его просмотра.

Во вкладке пакета можно осуществлять следующие операции (рис. 8):

1. редактирование текущего пакета;

2. удаление текущего пакета;
3. открытие другого пакета;

Рис. 8: Операции на вкладке пакета

4. просмотр списка форм документов/структур/трейтов/юнионов пакета;
5. создание новой формы документов пакета;
6. открытие в новой вкладке/редактирование/удаление формы документов;
7. создание новой структуры пакета;
8. редактирование/удаление структуры пакета;
9. создание нового трейта пакета;
10. редактирование/удаление трейта пакета;
11. создание нового юниона пакета;
12. редактирование/удаление юниона пакета;
13. управление uses.

4. Управление формами

Основные операции управления формами модели проводятся в начальном окне управления моделью (рис. 5). Если во Phlox не открыто ни одной вкладки, то начальное окно управления моделью открывается автоматически.

4.1. Создание формы

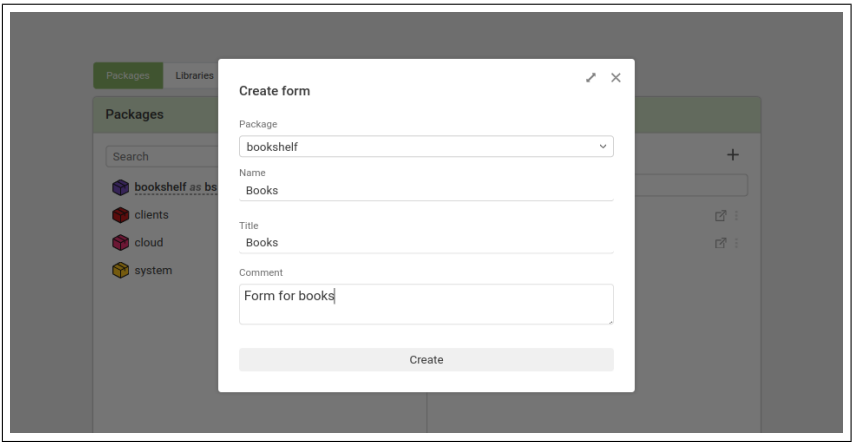

Рис. 9: Окно создания формы документов

Для создания новой формы в начальном окне управления моделью нужно:

1. нажать на кнопку «+» основной панели для открытия начального окна управления моделью;
2. в левом столбце открывшегося окна выбрать пакет, в котором планируется создание новой формы документа и кликнуть на него для открытия списка форм пакета;
3. в правом столбце окна нажать на «+» для создания новой формы;
4. в появившемся поле ввода набрать идентификатор (name) и читаемое название формы (title), комментарии (comments). Если

выбран не правильный пакет для создания формы, то на данном этапе его можно поменять на любой из ранее созданных;

5. для создания формы нажать кнопку «Create».

Вновь созданная форма появится в списке форм пакета. Если форма с данным идентификатором уже существует, то будет выдана ошибка.

4.2. Редактирование формы

Для редактирования формы документа в начальном окне управления моделью необходимо:

1. нажать на кнопку «+» основной панели для открытия начального окна управления моделью;

2. в левом столбце открывшегося окна выбрать пакет, в котором планируется редактирование формы документа и кликнуть на него для открытия списка форм пакета;

3. в правом столбце найти в списке форм нужную форму и нажать на кнопку «⋮», затем выбрать функцию «Edit»;

4. в полях всплывающего окна «Form edititng» (рис. 10) отредактировать выбранный пакет (package) и/или идентификатор формы (name) и/или ее читаемое название (title) и/или комментарии (comments) и/или последовательность полей формы (form field order);

5. завершить редактирование формы, нажав кнопку «Ok».

Отредактированная форма отобразятся в списке форм пакета. Если при редактировании был указан идентификатор, который уже существует, то будет выдана ошибка. После редактирования последовательности полей таблица документов автоматически перестраивается.

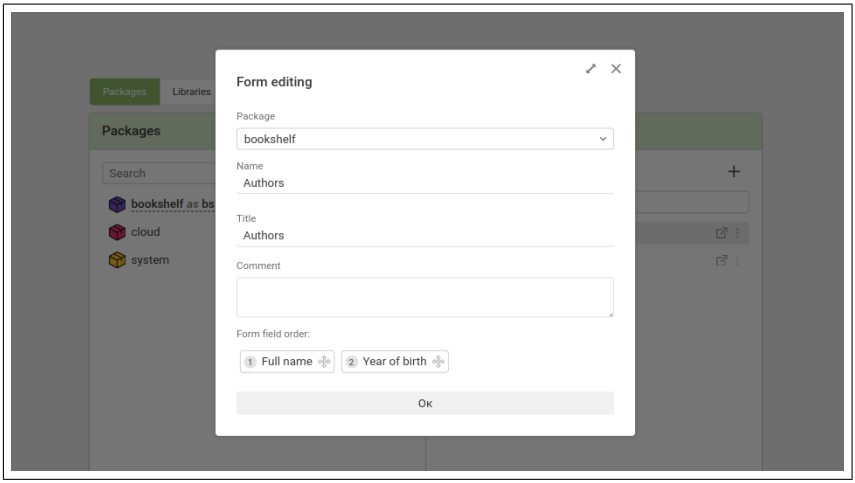

Рис. 10: Окно редактирования формы документов

4.3. Удаление формы

Для удаления формы в начальном окне управления моделью требуется:

1. нажать на кнопку «+» основной панели для открытия начального окна управления моделью;
2. в левом столбце открывшегося окна выбрать пакет, в котором планируется редактирование формы документа и кликнуть на него для открытия списка форм пакета;
3. в правом столбце найти в списке форм нужную форму и нажать на кнопку «⋮», затем выбрать функцию «Delete»;
4. подтвердить удаление в модальном окне, нажав кнопку «Ok».

После подтверждения удаления форма будет удалена. Во всех списках удаленная форма перестанет отображаться. Если была открыта вкладка с данной формой, то она автоматически закроется.

4.4. Отображение формы

Для детального отображения формы предусмотрена отдельная вкладка просмотра формы.

Во вкладке формы можно осуществлять следующие операции (рис. 11):

1. редактирование текущей формы;
2. удаление текущей формы;
3. открытие другой формы;

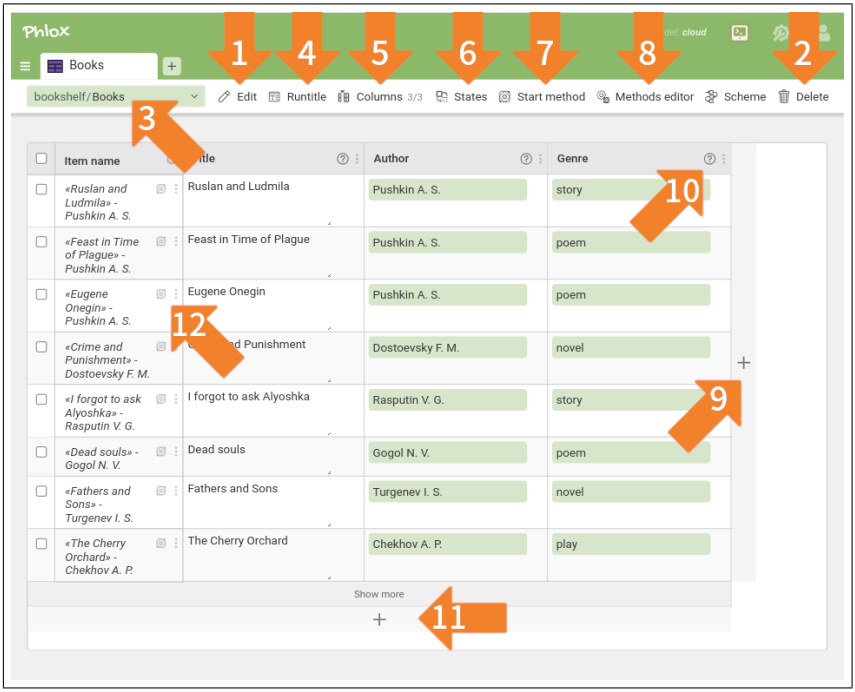

Рис. 11: Операции на вкладке формы

4. редактирование шаблона имени документа;
5. редактирование отображения колонок таблицы;

6. создание/редактирование/удаление статусов;
7. вызов методов в контексте формы;
8. редактирование методов;
9. создание полей формы;
10. создание документов формы;
11. редактирование полей документов формы;
12. вызов методов/удаление/открытие в новой вкладке документов формы;
13. редактирование/просмотр/удаление полей формы.

4.5. Шаблон имени документа

Шаблон имени документа — параметрическое выражение, вычисляемое в контексте документа данной формы, строковый результат которого интерпретируется как вычисляемое название данного документа. Вычисляемое название документа автоматически используется для идентификации пользователем документа. В частности, вычисляемое название (если определено) идентифицирует документ в первой колонке «Item name» таблицы документов формы.

Шаблон имени документа формируется в синтаксисе параметрической строки языка Libretto с вычисляемыми вставками в виде #{...}. Выражение внутри фигурных скобок вычисляется в контексте документа, имя которого генерируется.

Для определения/редактирования шаблона имени документа необходимо:

1. нажать кнопку «Runtitle» на основной панели интерфейса Phlox на вкладке формы (рис. 11, п. №4);
2. отредактировать шаблон в поле ввода шаблона;
3. нажать «Ok» для завершения операции.

4.6. Управление полями формы

4.6.1. Создание поля формы

Данная функция позволяет добавить в форму новое поле. Добавление поля в форму автоматически добавит это поле в каждый документ формы.

Для добавления в форму нового поля нужно:

1. нажать на «+» с правой стороны таблицы документов формы;
2. в появившемся окне «Create a new field» (рис. 12) ввести идентификатор поля (name), его читаемое имя (title) и комментарии (comment);

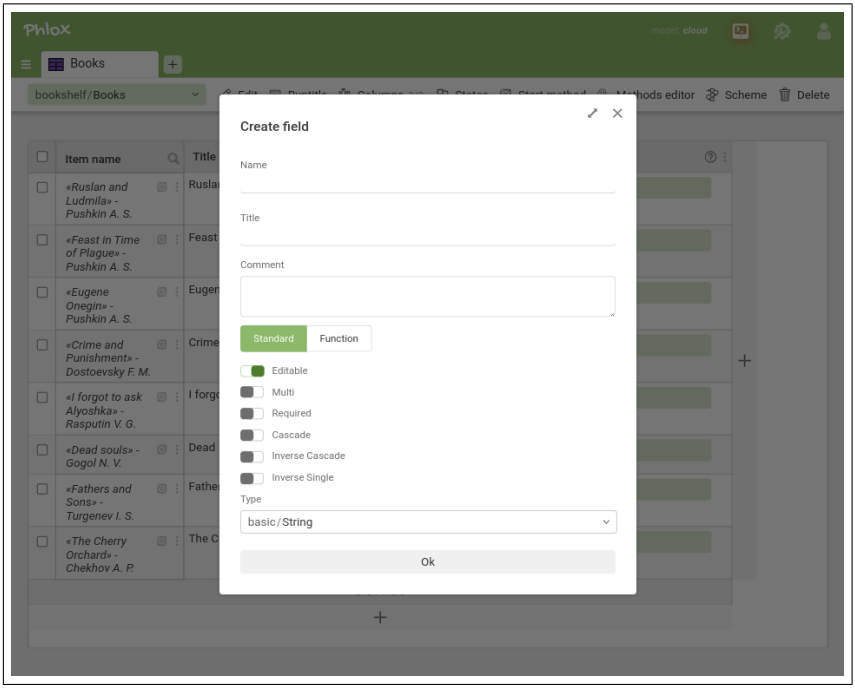

Рис. 12: Окно создания поля формы

3. выбрать формат поля «Standard» или «Function»;

- для поля «Standard»:
  - раскрыть выпадающее меню «Select type» (рис. 13) и выбрать типы допустимых значений поля (их может быть несколько);

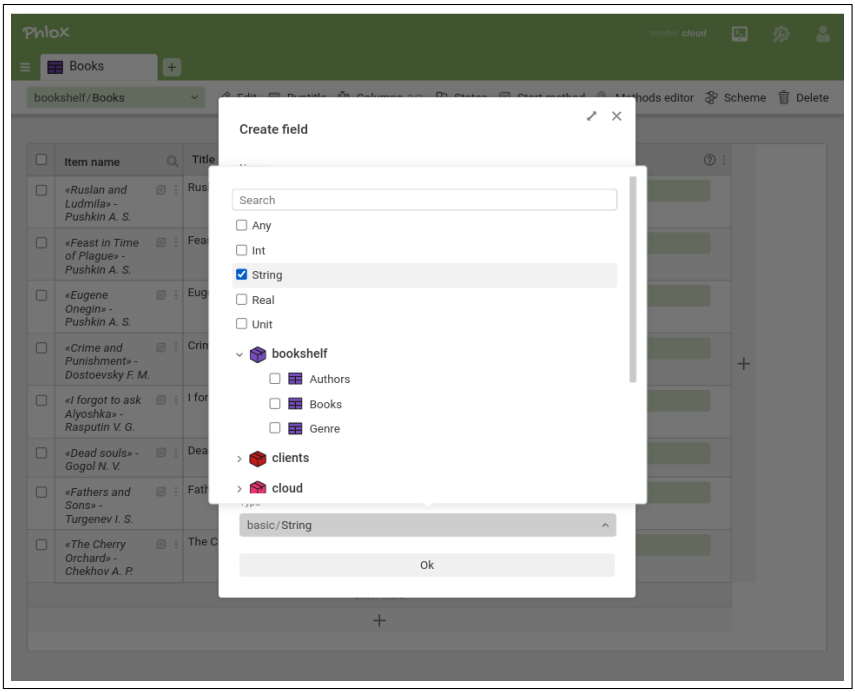

Рис. 13: Селектор типов

- для поля «Function»:
  - заполнить идентификатор параметра (name), представлен на рис. 14, п. №1;
  - заполнить тип (type), представлен на рис. 14, п. №2;
  - добавить при необходимости дополнительные параметры, представлено на рис. 14, п. №4;
  - при необходимости удалить параметры, представлено на рис. 14, п. №3;

– заполнить поле «Type value», раскрыв выпадающее меню «Select type» и выбрав типы допустимых значений поля (их может быть несколько), представлено на рис. 14, п. №5;

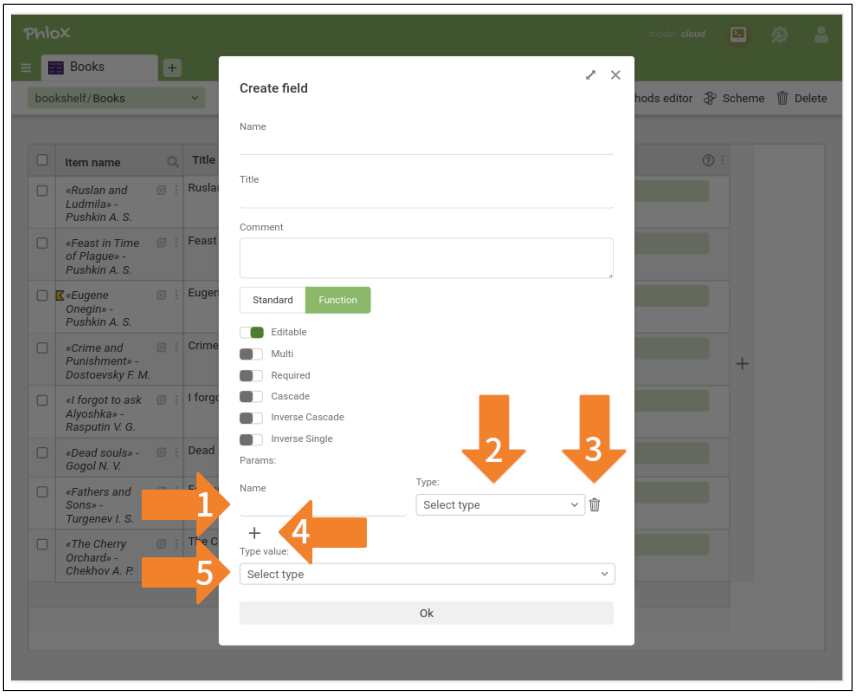

Рис. 14: Окно создания поля формы формата «Function»

4. если необходимо запретить редактирование поля, то нужно выключить опцию «Editable»;
5. если поле может содержать более одного значения, то нужно включить опцию «Multi»;
6. если значение поле является обязательным (не может быть пустым), то нужно включить опцию «Required»;
7. если значения поля являются зависимыми от данного документа и должны быть удалены при удалении документа, то нужно

включить опцию «Cascade».

8. опция «Inverse Cascade» позволяет удалить документы, у которых удаляемый документ проставлен в поле с включенной опцией «Inverse Cascade»;

9. опция «Inverse Single» включает контроль при выборе значения «документ», документ может быть прикреплён только к одному документу.

4.6.2. Редактирование поля формы

Для редактирования поля формы необходимо:

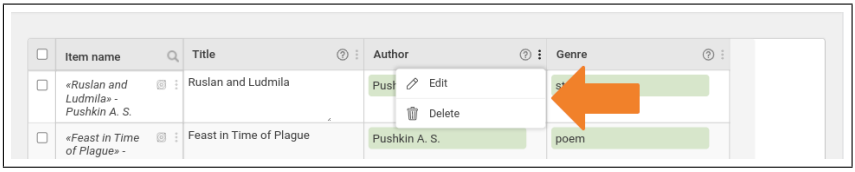

Рис. 15: Редактирование поля формы

1. нажать на кнопку «⋮» в правом углу столбца поля, затем выбрать функцию «Edit» (рис. 15);

2. в появившемся окне «Field editing» откорректировать необходимые параметры поля (идентификатор, читаемое имя, формат, параметры (для «Function»), тип, опции Editable, Multi, Required и Cascade);

3. завершить редактирование поля формы, нажав кнопку «Ok».

После редактирования поля формы изменения стразу отобразятся в таблице. Если тип поля был расширен, до заполненные ранее данные будут сохранены, если тип поля был кардинально изменен (например, тип Item заменили на тип structure/Date), то поля заполнятся пустыми значениями.

4.6.3. Удаление поля формы

Для удаления поля формы требуется:

1. нажать на кнопку «⋮» в правом углу столбца поля, затем выбрать функцию «Delete» (рис. 15);
2. подтвердить удаление в модальном окне, нажав кнопку «Ok».

4.6.4. Настройка отображения полей

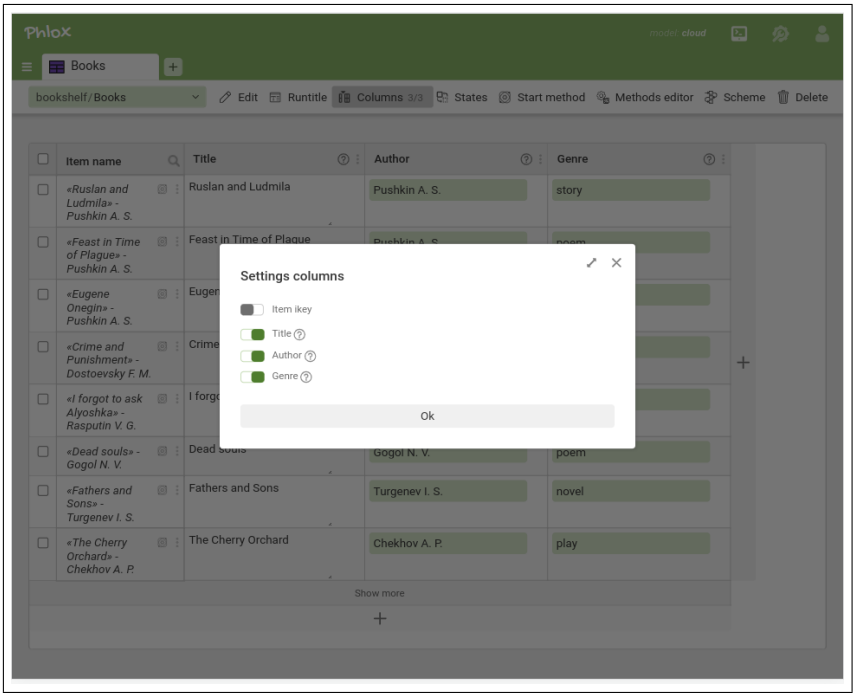

Рис. 16: Редактирование отображения полей формы

Каждое поле в таблице можно скрыть, для этого необходимо открыть окно настройки колонок, нажав на кнопку «Columns» на панели управления формой (рис. 11, п. №5).

Редактирование отображения полей формы представлено на рис. 16. Эта настройка сохраняется для каждого пользователя.

5. Управление статусами

Статусы используются для фиксации жизненного цикла документов данной формы. Окно управления статусами открывается с помощью кнопки «States» на основной панели интерфейса управления формой Phlox. Статус документа отображается в таблице документов (столбец «State», рис. 17), если эта опция включена в настройках статусов.

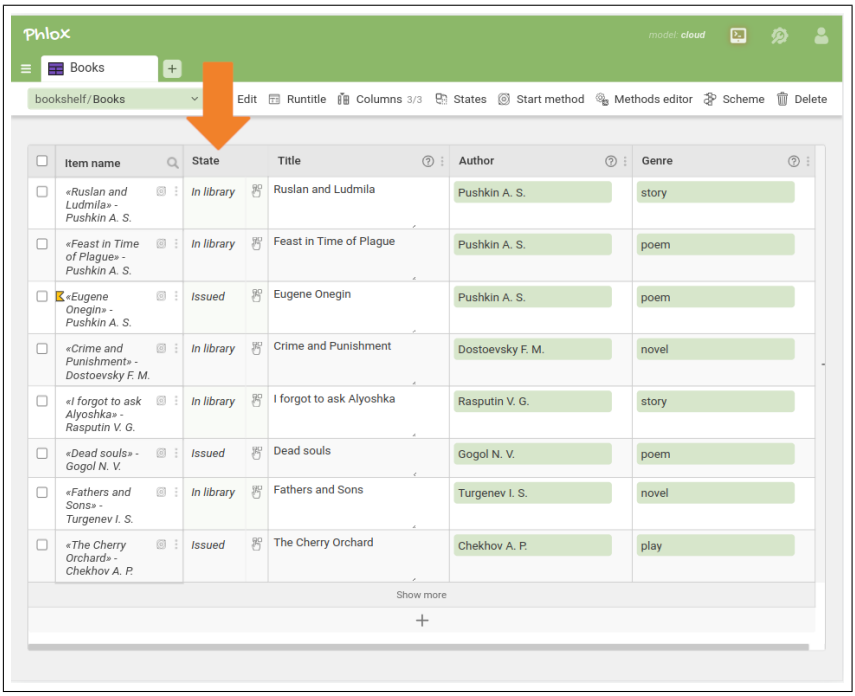

Рис. 17: Колонка статусов в таблице документов

5.1. Создание статуса

Для добавления нового статуса нужно:

1. нажать кнопку «States» на основной панели интерфейса Phlox на вкладке формы (рис. 11, п. №5);
2. в появившемся окне «States settings» (рис. 18) нажать кнопку «+»;
3. в полях всплывающего окна «Create state» заполнить идентификатор статуса (name) и его читаемое имя (title);
4. завершить создание статуса, нажав кнопку «Ok».

Описание дополнительных возможностей окна настроек статусов (рис. 18):

1. настройка отображения статусов в таблице;
2. установка статуса по умолчанию;
3. сортировка статусов по алфавиту;
4. ручная сортировка статусов (рис. 19).

5.2. Редактирование статуса

Для редактирования статуса нужно:

1. нажать кнопку «States» на основной панели интерфейса Phlox на вкладке формы (рис. 11, п. №5);
2. в появившемся окне «States settings» (рис. 18) нажать кнопку «карандаш»;
3. в полях всплывающего окна «State edititng» отредактировать идентификатор статуса (name) и/или его читаемое имя (title);
4. завершить редактирование статуса, нажав кнопку «Ok».

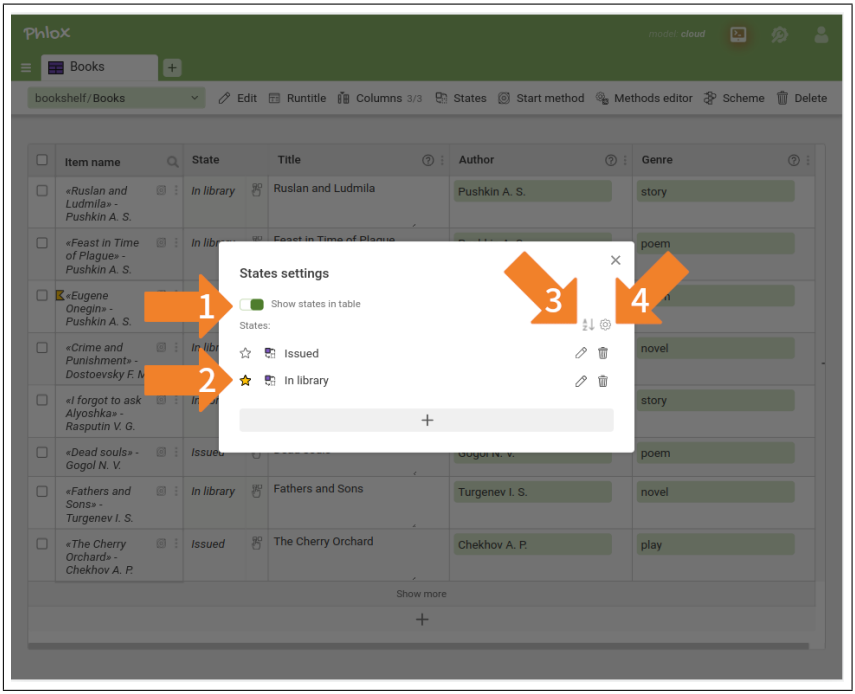

Рис. 18: Окно настроек статусов

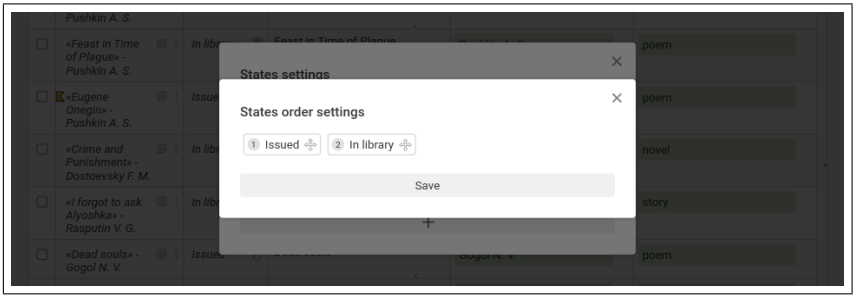

Рис. 19: Окно сортировки статусов

5.3. Удаление статуса

Для удаления статуса необходимо:

1. нажать кнопку «States» на основной панели интерфейса Phlox на вкладке формы (рис. 11, п. №5);
2. в появившемся окне «States settings» (рис. 18) нажать кнопку «корзина»;
3. подтвердить удаление в модальном окне, нажав кнопку «Ok».

5.4. Установить статус по умолчанию

Для того чтобы установить статус по умолчанию, требуется:

1. нажать кнопку «States» на основной панели интерфейса Phlox на вкладке формы (рис. 11, п. №5);
2. в появившемся окне «States settings» (рис. 18) нажать кнопку «звездочка»;
3. закрыть окно «States settings».

После проставления статуса по умолчанию, вновь создаваемым документам будет автоматически проставлен выбранный статус. Также документам созданным ранее, но на текущий момент без статуса, присвоится выбранный статус.

5.5. Включить/выключить отображение статуса в таблице

Для того чтобы включить/выключить отображение статуса в таблице нужно:

1. нажать кнопку «States» на основной панели интерфейса Phlox на вкладке формы (рис. 11, п. №5);
2. в появившемся окне «States settings» (рис. 18) включить / выключить опцию «Show states in table»;
3. закрыть окно «States settings».

5.6. Смена статуса документа

Для того чтобы «вручную» установить статус документу необходимо:

1. нажать кнопку «выбора» в колонке статуса выбранного документа на вкладке формы (рис. 20);

Рис. 20: Смена статуса документа

2. выбрать нужный статус и подтвердить изменения, нажав кнопку «Ok» (рис. 21).

6. Управление методами

Методы являются инструментом построения документных моделей в объектно-ориентированном стиле. Методы определяются на формах и применяются к документам этих форм для обработки данных модели.

Методы являются инструментом работы в стиле low-code, поскольку в методах, как правило, задаются минимальные инструкции, в то время как основные компоненты моделей визуализируются и определяются на декларативном уровне модели. При этом кодируется только тело метода, в то время как его сигнатура определяется декларативно. Декларативно задаются название метода, параметры метода, типы данных параметров и тип результата. Благодаря этому они доступны для логических и визуальных средств модели.

Также можно определить ограничения на область значений, которые могут принимать параметры метода, значения параметров по умолчанию и условия, при которых значения параметров задаются автоматически. Эти настройки важны для эффективного использования методов на уровне Phlox, а также быстрого построения веб-интерфейсов.

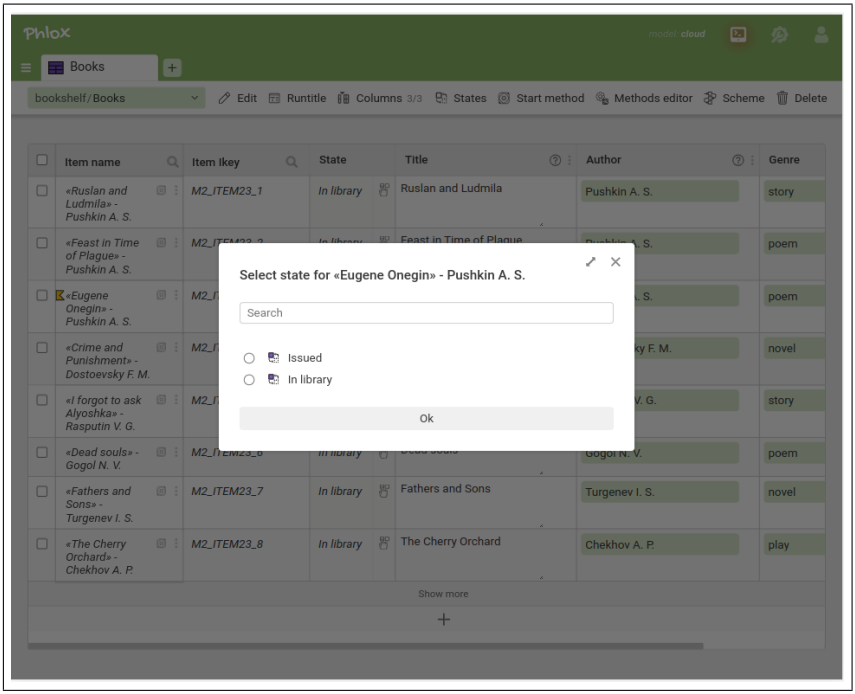

Рис. 21: Окно выбора статуса для конкретного документа

Ontobox обеспечивает разнообразные возможности применения методов в стиле no-code. В частности, вызов метода может осуществляться непосредственно из табличного интерфейса Phlox.

Можно сравнить действие методов с формулами в ячейках Excel — с тем отличием, что возможности методов в Ontobox намного шире. Они позволяют быстро разворачивать объектно-ориентированные модели любой вычислительной сложности, при необходимости варьируя уровень разработки от no-code, через low-code к full-code (например, когда нужно реализовать сложный вычислительный алгоритм). Выбор подходящего уровня разработки — влиятельный инструмент оптимизации разработки и поддержки моделей и приложений.

При компиляции моделей в код динамически генерируемой библиотеки Librun, методы становятся частью этого кода на языке

Libretto. Librun транслируется в byte-код виртуальной Java-машины, поэтому эффективность исполнения методов соответствует эффективности исполнения Java-программ.

6.1. Создание метода

Создание и редактирование методов формы осуществляется в редакторе методов (рис. 23), который открывается во Phlox как отдельное окно. Для открытия окна нужно на основной панели Phlox нажать на кнопку «Methods editor» (рис. 11, п. №7).

Для создания нового метода, определенного на форме нужно:

1. открыть редактор методов;
2. в левом поле редактора нажать на кнопку «+» (рис. 23, п. №1);
3. в поле Name и Title окна ввода ввести имя и читаемое имя определяемого метода, соответственно (рис. 22);
4. указать контекст метода: документ или форма документа;

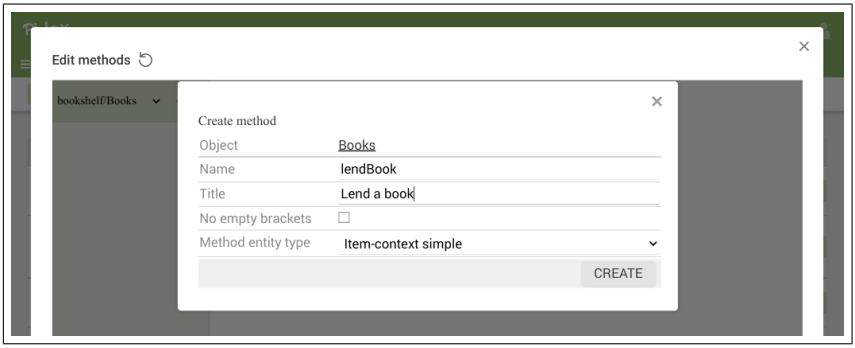

Рис. 22: Окно создания метода

5. нажать на кнопку «Create».

6.2. Редактирование метода

Для открытия окна редактирования метода необходимо:

1. открыть редактор методов;
2. выбрать метод для редактирования в левом поле редактора методов (рис. 23, п. №6). В правом поле редактора откроется вкладка редактирования метода «Code» (рис. 24).

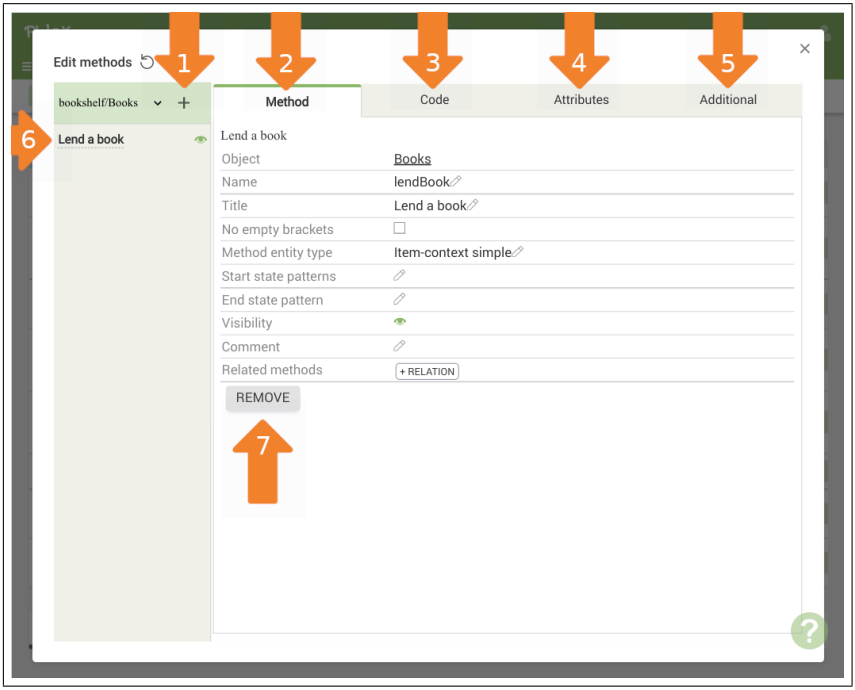

Рис. 23: Редактор методов, вкладка «Method»

Окно редактирования метода включает четыре вкладки для тонкой настройки метода:

- Method — основные характеристики метода (имя, читаемое имя, видимость, комментарии, связанные методы для определения бизнес-процессов и т.д.);

- Code — определение параметров (аргументов) и тела метода;
- Attributes — определение атрибутов/ключей метода (для управления методами в модели);
- Additional — дополнительные свойства метода.

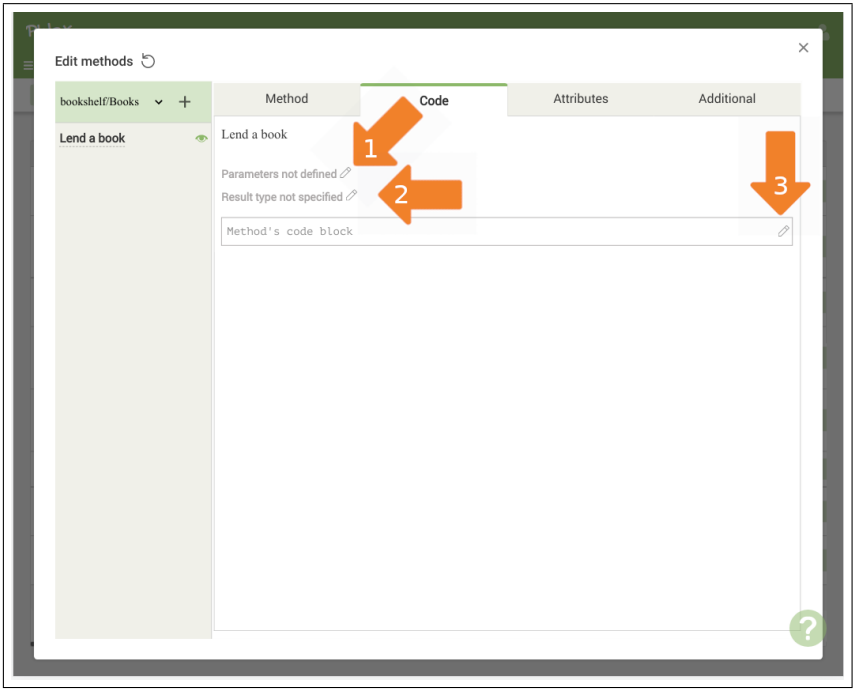

Рис. 24: Редактор методов, вкладка «Code»

6.3. Задание типа результата метода

Компилятор Libretto умеет автоматически вычислять тип результата метода. Тем не менее, рекомендуется явно типизировать результат, чтобы повысить уровень контроля над типами данных. Кроме того, задание типа на уровне модели позволяет использовать эту информацию при логической обработке.

Для задания типа результата метода необходимо:

1. во вкладке «Code» описания метода щелкнуть мышью на значок «карандаш» для задания типа результата метода (рис. 24, п. №2);

2. во всплывающем окне выбрать тип результата;

3. настроить кардинальность результата — включить опцию «Multi», если метод возвращает произвольное количество значений, и включить опцию «Required», если метод не может возвращать в качестве значения пустую последовательность;

4. закрыть окно, нажав на кнопку «Close».

6.4. Редактирование кода (тела) метода

В теле метода содержится код на Libretto, определяющий алгоритм вычисления метода. При разработке тела метода доступны как автоматически сгенерированные, так и ранее определенные пользователем методы из библиотеки управления документной моделью Librun. Методы, определенный в библиотеке Librun, могут быть получены с помощью подсказки в правом поле окна редактирования метода.

После завершения определения метода, код метода автоматически вставляется в библиотеку управления моделью Librun, после чего производится динамическая перекомпиляция Librun. После этого определяемый метод доступен для использования во Phlox.

Для кодирования тела метода нужно:

1. в окне редактирования метода (вкладка «Code») нажать на иконку редактирования тела метода (рис. 24, п. №3);

2. в открывшемся окне редактирования ввести код тела метода;

3. при необходимости, при редактировании тела метода можно использовать подсказку, позволяющую быстро находить и переносить в код вызовы нужных параметров и методов;

4. после завершения редактирования нажать кнопку «Save».

6.5. Редактирование имени и читаемого имени метода

Для редактирования имени и читаемого имени метода требуется:

1. открыть окно редактирования метода;
2. в окне редактирования метода перейти во вкладку «Method» (рис. 23, п. №2);
3. во вкладке отредактировать нужное поле, нажав на соответствующую иконку «карандаш»;
4. закрыть окно редактирования метода.

6.6. Удаление метода

Для удаления метода из модели необходимо:

1. открыть окно редактирования метода;
2. в окне редактирования метода перейти во вкладку «Method»;
3. нажать на кнопку «Remove» (рис. 23, п. №7).

6.7. Управление параметрами метода

6.7.1. Добавление параметра метода

Для добавления параметра метода нужно:

1. во вкладке «Code» описания метода щелкнуть мышью на значок «карандаш» для редактирования параметров метода (рис. 24, п. №1);
2. в появившемся окне редактирования параметров нажать на кнопку «Add parameter» (рис. 25, п. №9);
3. в появившейся новой строке таблицы параметров метода ввести имя параметра (рис. 25, п. №1), читаемое имя (используется для автоматически порождаемых интерфейсов), тип параметра (рис. 25, п. №2), провести настройку кардинальности (опции Multi и Required; рис. 25, п. №3);
4. завершить добавление параметра, нажав кнопку «Save».

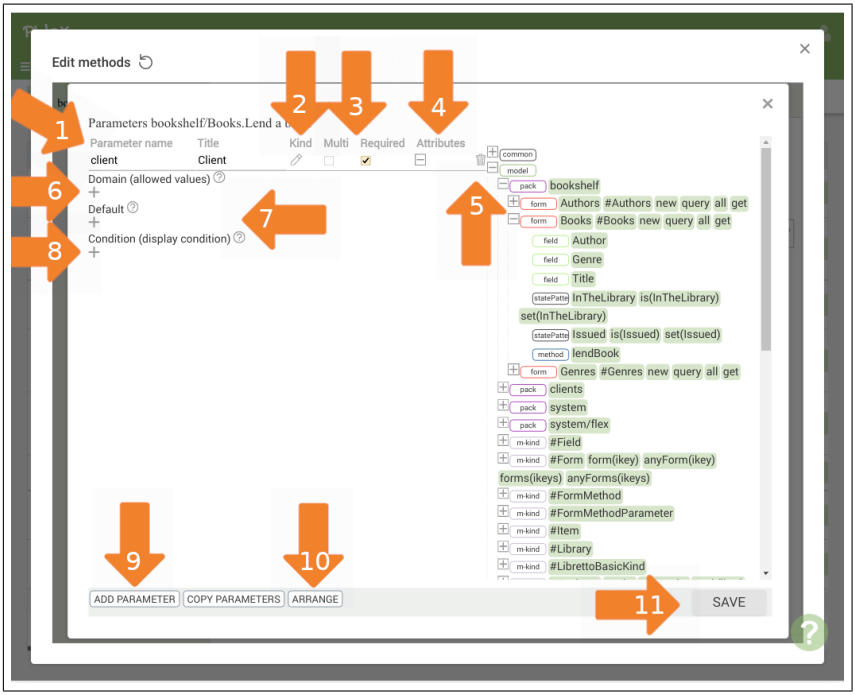

Рис. 25: Редактор параметров метода

6.7.2. Изменение порядка вхождения параметров

При вызове метода (в интерфейсе Phlox) значения параметров должны располагаться в том же порядке, как они расположены в таблице параметров. Для изменения порядка вхождения параметров нужно:

1. открыть окно редактирования параметров;
2. нажать на кнопку «Arrange» (рис. 25, п. №10);
3. в появившемся окне с помощью стрелок настроить порядок вхождения параметров;
4. нажать на кнопку «Ok» после завершения настройки порядка параметров.

6.7.3. Редактирование атрибутов параметра

Ontobox позволяет на декларативном уровне описать дополнительные семантические свойства параметров метода, которые не только полезны с точки зрения логического описания предметной области, но и приносят большую пользу при построении интерфейсов в стиле no-code и low-code в той его части, которая связана с применением методов. Речь идет о трех видах логических атрибутов параметра:

- Domain (allowed values) (рис. 25, п. №6). Определяется в виде выражения на Libretto, вычисляемого в контексте документа. Если значение определено, то возвращаемый результат вычислений интерпретируется как множество допустимых значений параметра.

  При построении приложений допустимые значения будут выдаваться пользователю в виде селектора значений, с помощью которого пользователь выбирает нужное ему значение параметра.

- Default (рис. 25, п. №7). Определяется в виде выражения на Libretto, вычисляемого в контексте документа. Если значение определено, то возвращаемый результат вычислений интерпретируется как значение параметра по умолчанию.

  При построении приложений значение по умолчанию выставляется в поле ввода параметра, а в случае запрета пользователю самостоятельно выбирать значения параметра (в зависимости от условия Display condition) значение параметра по умолчанию становится его единственным допустимым значением.

- Display condition (рис. 25, п. №8). Это условие говорит о том, показывать ли параметр пользователю в интерфейсе для ручного задания параметров, или этот параметр должен быть скрыт от пользователя и автоматически означен по умолчанию.

Для задания данных условий необходимо:

1. во вкладке «Code» описания метода нажать на значок «карандаш» для редактирования параметров метода;

2. в окне настройки параметров отыскать необходимый пункт в таблице и кликнуть на иконку в столбце «Attributes» (рис. 25, п. №4);

3. в появившихся атрибутах нажать на «+» напротив атрибута, который хотим отредактировать (рис. 25, п. №6, №7, №8), ввести в появившемся окне ввода нужное выражение;

4. при необходимости, при формулировании выражения можно использовать подсказку, позволяющую быстро находить и переносить в код вызовы нужных методов;

5. после завершения редактирования нажать на кнопку «Save».

6.7.4. Удаление параметра метода

1. открыть окно редактирования параметров метода;

2. нажать на иконку «корзина» в строке удаляемого параметра (рис. 25, п. №5).

7. Управление документами

Помимо возможности управления формой, во вкладке формы также предоставляется функционал для работы с документами этой формы, включая доступ к их содержимому и редактирование.

7.1. Создание документа

Для создания нового документа текущей формы нужно нажать на «+» внизу таблицы документов (рис. 11, п. №10).

В таблицу будет добавлена строка с новым документом, в которой будут присутствовать пустые поля, готовые для ввода информации.

7.2. Редактирование документа

1. Если поле имеет базовый тип данных (строка, целое или вещественное число и др.), то необходимо щелкнуть на ячейку поля и в текстовой области ввести базовое значение.

2. Если типом значения поля является форма документа, то необходимо навести на значок «+» в ячейке поля, нажать на него. В выпадающем окне появится список документов — допустимых значений поля. В этом списке нужно выбрать необходимый документ и щелкнуть по нему мышью. Выбранный документ станет значением поля.

3. В случае если поле допускает множественные значения, то в правой части ячейки поля доступен значок «+», нажав на который добавится новое пустое значение для заполнения в соответствии с п. №1 и №2.

Возможные типы полей (рис. 26):

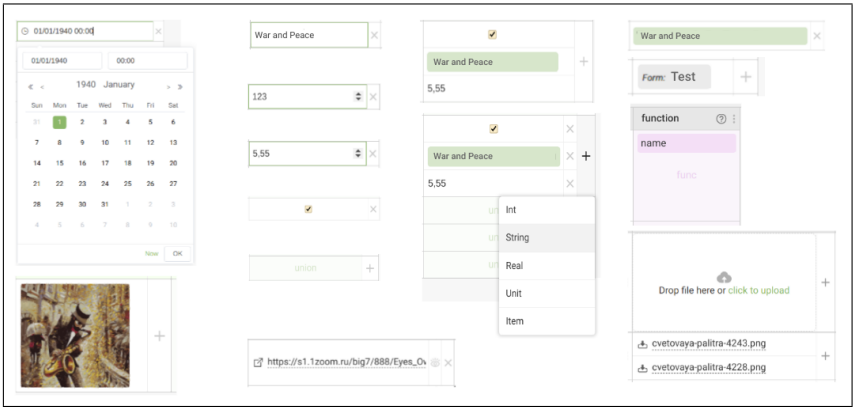

Рис. 26: Типы полей документов

- целое число;
- вещественное число;
- строка;
- unit;
- документ формы;
- мета-объект;

- структура;
- дата;
- wiki;
- картинка;
- ссылка;
- видео;
- файл;
- функция.

7.3. Удаление документа

Для удаления документа из модели нужно:

1. открыть меню удаляемого документа (в столбце «Item name») нажав кнопку «⋮» (рис. 11, п. №12),
2. выбрать функцию «Delete»;
3. подтвердить удаление в модальном окне, нажав кнопку «Ok».

После подтверждения удаления документ будет удален. Во всех списках удаленный документ перестанет отображаться. Если была открыта вкладка с данным документом, то она автоматически закроется.

7.4. Глобальный ключ документа

Задание глобального ключа документа (константы) — один из способов глобального доступа к документу в методах и ином коде, использующем библиотеку Librun.

Для привязки документа к глобальному ключу требуется:

1. открыть меню удаляемого документа (в столбце «Item name») нажав кнопку «⋮» (рис. 11, п. №12),
2. выбрать функцию «New constant»;

3. в появившемся окне нажать кнопку «Create constant», после этого ввести строковое значение ключа;

4. подтвердить создание, нажав кнопку «Ok»;

5. прикрепленный документ будет помечен специальным символом (рис. 27).

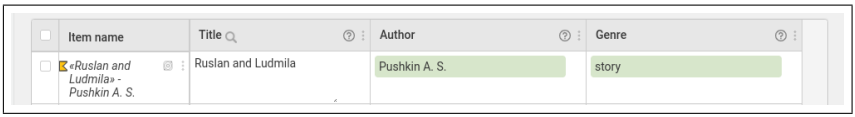

Рис. 27: Документ-константа

Поскольку имя ключа является глобальным на уровне модели, рекомендуется выбирать имена ключей, гарантирующих отсутствие конфликтов имен.

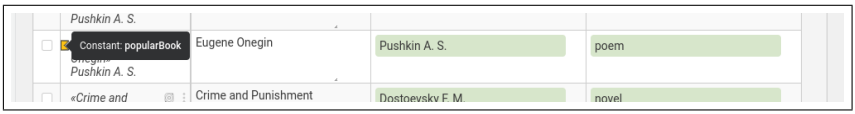

Рис. 28: Информация о константах документа

Для удаления глобального ключа документа необходимо:

1. открыть меню удаляемого документа (в столбце «Item name») нажав кнопку «⋮» (рис. 11, п. №12),

2. выбрать функцию «Edit constants»;

3. в появившемся окне со списком глобальных ключей документа найти удаляемый ключ и открыть его меню, нажав кнопку «⋮», затем выбрать функцию «Delete» (рис. 29);

4. документ будет отвязан от ключа, а сам ключ удален. Пока глобальный ключ используется в каком-либо методе, удаление будет невозможным.

При наведении на символ константы, появляется уведомление, в котором содержится информация о значение ключа (рис. 28).

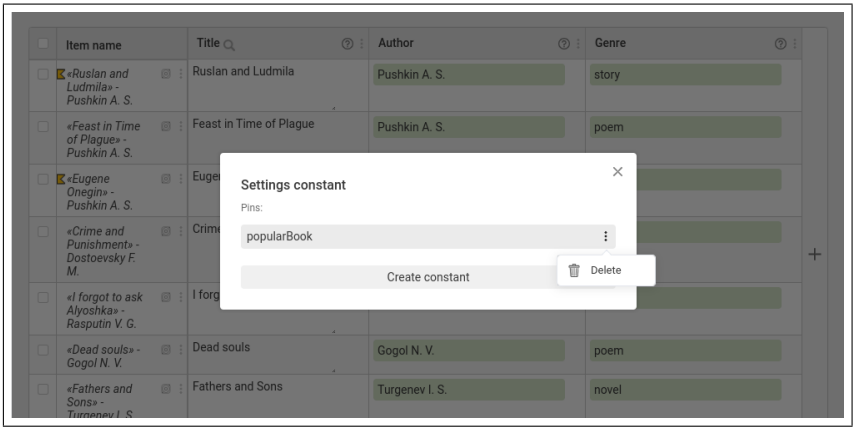

Рис. 29: Настройки констант документа

7.5. Поиск документов

Поиск документов можно осуществлять на вкладке форму как по имени документа (рис. 30), так и по идентификатору (рис. 31), для этого нужно открыть поле ввода, нажав кнопку «лупа» вверху колонки «Item name»/«Item ikey».

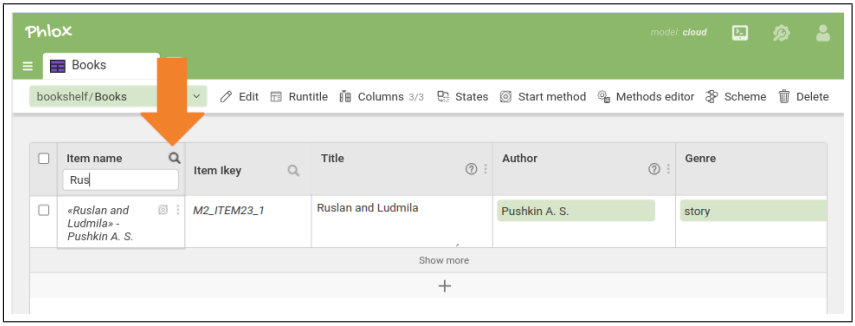

Рис. 30: Поиск документа по имени

Для того чтобы осуществить поиск по идентификатору, необходимо включить отображение колонки "Item ikey".

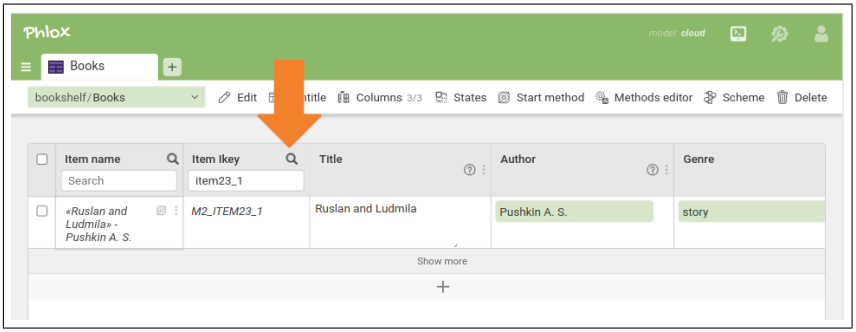

Рис. 31: Поиск документа по идентификатору

7.6. Вызов метода без параметров в контексте документа

Для вызова метода без параметров нужно:

1. открыть меню методов документа (в столбце «Item name») нажав кнопку «методы» (рис. 11, п. №12),

2. выбрать необходимый метод. После этого появится «loading», уведомляющий о начале выполнения метода. После выполнения метода появится всплывающее окно с результатом выполнения метода (рис. 32) и дополнительной информацией (время выполнения метода; логи выполнения, если они были прописаны в методе).

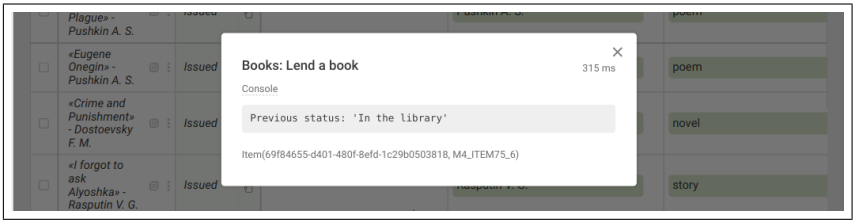

Рис. 32: Всплывающее окно с результатом выполнения метода

После выполнения метода данные документа будут обновлены.

7.7. Вызов метода с параметрами в контексте документа

Для вызова метода с параметрами необходимо:

1. открыть меню методов документа (в столбце «Item name») нажав кнопку «методы» (рис. 11, п. №12),

2. выбрать необходимый метод. После этого появится всплывающее окно для заполнения параметров метода (рис. 33);

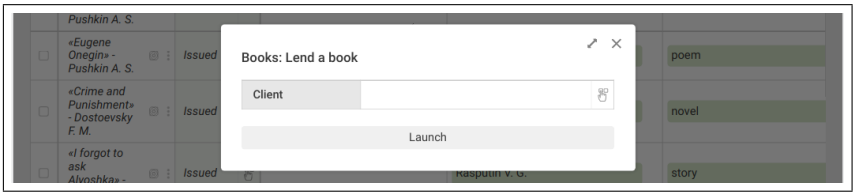

Рис. 33: Всплывающее окно с параметрами метода

3. заполнить параметры метода (рис. 34);

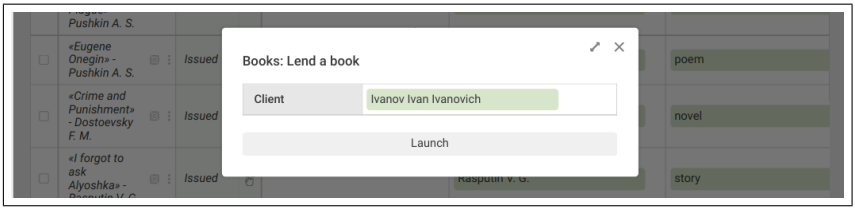

Рис. 34: Всплывающее окно с заполненными параметрами метода

4. нажать кнопку «Launch». Если все параметры были заполнены правильно, то появится «loading», уведомляющий о начале выполнения метода. Если при заполнении параметров были допущенные ошибки, то выполнение метода будет приостановлено и указано в каких полях допущены ошибки. После выполнения метода появится всплывающее окно с результатом выполнения метода и дополнительной информацией (время выполнения метода; логи выполнения, если они были прописаны в методе).

После выполнения метода данные документа будут обновлены.

7.8. Мультиредактор документов

Для ускоренного выполнения однотипных функций реализована возможность групповой обработки документов. Для этого необходимо выбрать нужные документы (первая колонка таблицы документов), после чего появится дополнительная строка функций (рис. 35) для обработки нескольких документов.

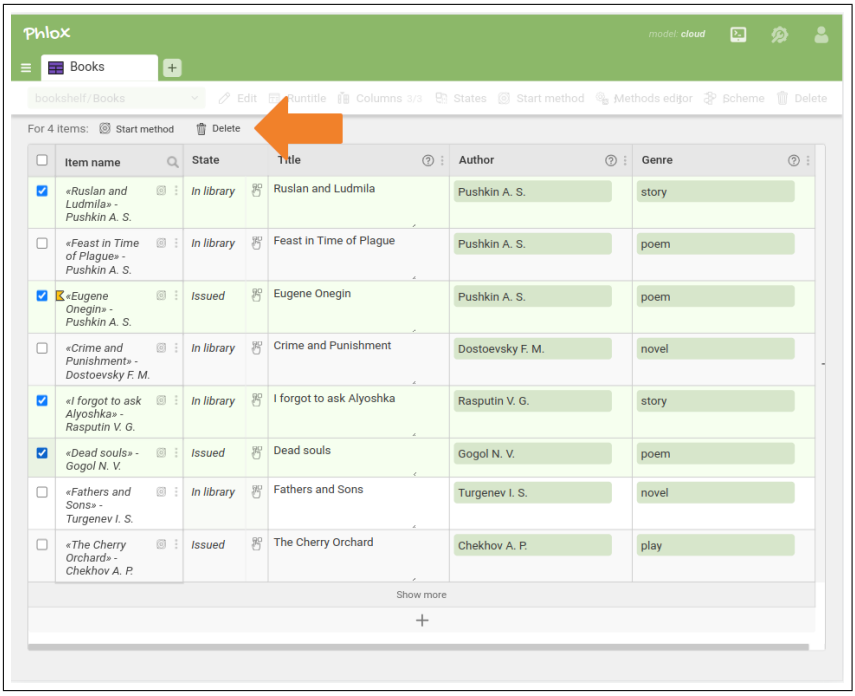

Рис. 35: Мультиредактор документов

Для групповой обработки документов доступны следующие функции:

- Удаление.
  После нажатия кнопки «Delete» (в дополнительной строке функций для групповой обработки) появится модальное окно для подтверждения удаления. Если подтверждение было получено, все выбранные документы будут удалены и строка функций для групповой обработки будет скрыта.

- Выполнение метода.
  После нажатия кнопки «Methods» (в дополнительной строке функций для групповой обработки) в выпадающем меню отобразится список методов доступных для выбранных документов. Если будет выбран метод без параметров, то он будет последовательно выполнен для каждого документа. Если будет выбран метод с параметрами, то появится всплывающее окно для заполнения параметров метода, атрибут «domain» для параметров будет рассчитан относительно всех выбранных документов. После правильного заполнения всех параметров, метод будет последовательно выполнен для каждого документа.

7.9. Взаимодействие с одним документом

7.9.1. Представление одного документа на отдельной вкладке

Для отображения документа также предусмотрена специальная вкладка (рис. 36).

Функционал вкладки «документ»:

1. вызвать метод в контексте документа;
2. удалить документ;
3. просмотр основных данных о документе: имя, идентификатор;
4. открыть форму текущего документа;
5. отредактировать статус текущего документа;
6. подгруздка инверсных документов — получение документов которые ссылаются на текущий документ;
7. редактирование полей документа.

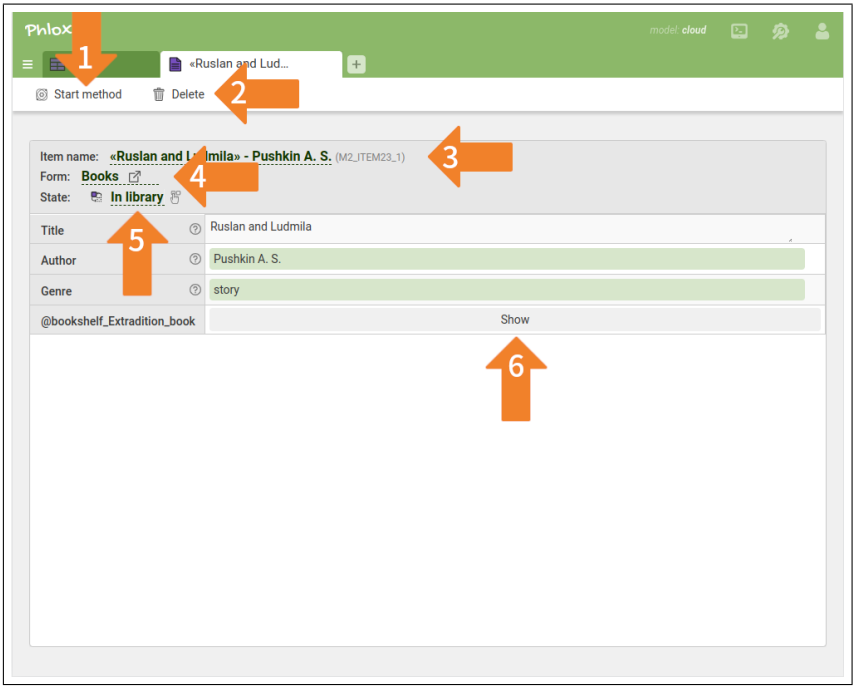

Рис. 36: Вкладка документа

7.9.2. Создание документа «по месту»

Периодически при работе в системе требуется создать документ «по месту», например, при заполнении полей документа или при заполнении параметров метода.

При заполнении какого-либо поля значением с типом «документ» появляется всплывающее окно с уже имеющимися в системе документами в виде дерева «пакет > форма > документы», у имени формы справа есть кнопка «+» (рис. 37, п. №1), которая отвечает за создание документа «по месту», по клику на которую открывается всплывающее окно для заполнения полей нового документа (рис. 38, п. №1). После заполнения всех полей, необходимо нажать кнопку «Create», всплывающее окно будет закрыто, а в поле для выбора документов добавится только что созданный документ (рис. 37, п. №2).

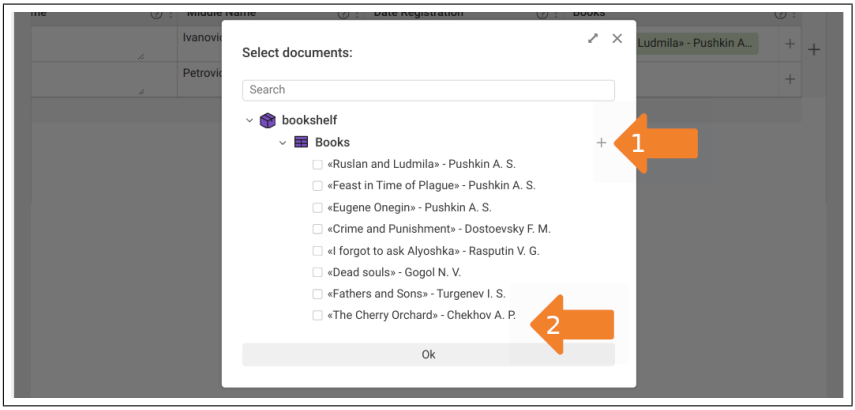

Рис. 37: Создание документа «по месту»

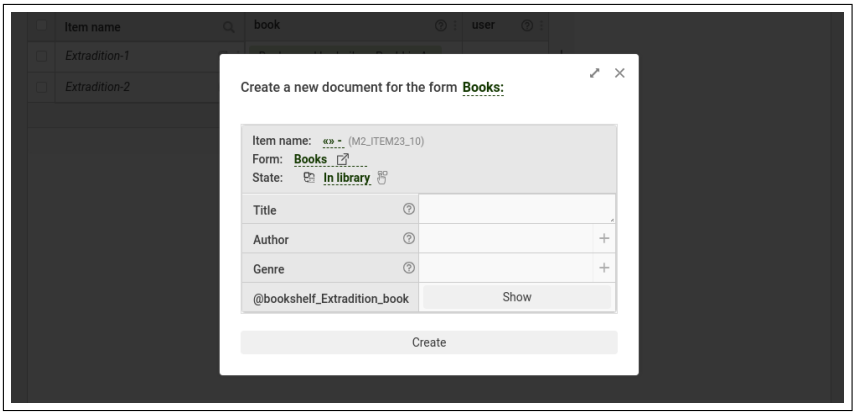

Рис. 38: Заполнение нового документа во всплывающем окне

8. Древовидная структура управления моделью

Для быстрого управления Meta-объектами реализован древовидный интерфейс, отображающий всю структуру модели — *древовидная структура управления (ДСУ)*. Для того чтобы открыть данный интерфейс, необходимо нажать кнопку «≡» на основной панели (рис. 39).

Рис. 39: Открытие древовидного интерфейса

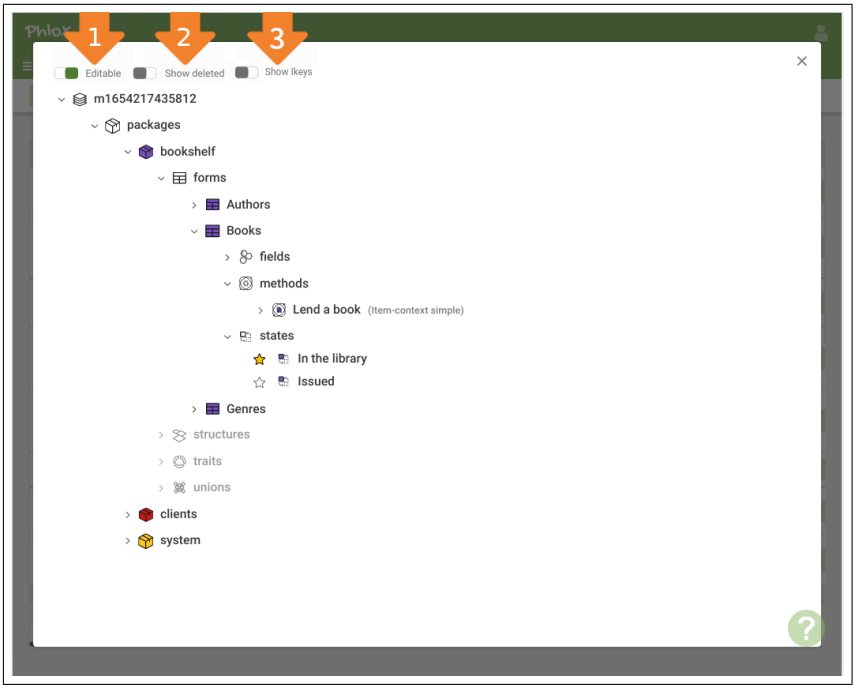

Рис. 40: Древовидная структура управления моделью (ДСУ)

Данный интерфейс содержит в себе три основные настройки (рис. 40):

1. Editable — разрешить редактирование объектов.
2. Show deleted — показывать удаленные объекты.
3. Show ikeys — показывать идентификаторы объектов.

Если включена настройка «Editable», то становится доступным следующий функционал древовидного интерфейса управления Meta-объектами:

- Пакеты.
  При наведении на «packages» появляется кнопка «+» (рис. 41), нажав на которую можно создать пакет.

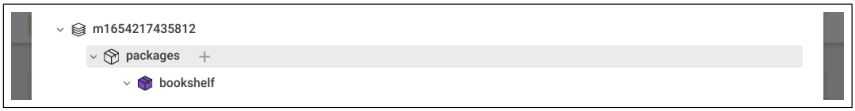

Рис. 41: ДСУ: Пакеты

- Пакет.
  При наведении на пакет появляются кнопки: открытия в новой вкладке, редактирования, удаления (рис. 42).

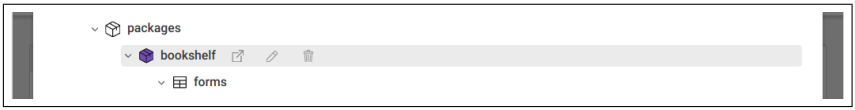

Рис. 42: ДСУ: Пакет

- Формы.
  При наведении на «forms» появляются кнопки: создания, копирования (рис. 43).

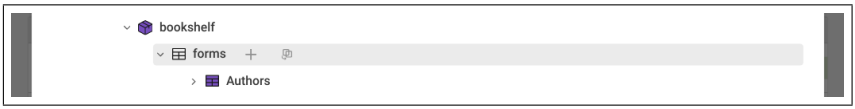

Рис. 43: ДСУ: Формы

- Форма.
  При наведении на форму появляются кнопки: открытия в новой вкладке, редактирования, редактирования шаблона имени документов, удаления (рис. 44).

Рис. 44: ДСУ: Форма

- Поля формы.
  При наведении на «fields» появляются кнопки: создания, копирования (рис. 45).

Рис. 45: ДСУ: Поля формы

- Поле формы.
  При наведении на поле формы появляются кнопки: редактирования, удаления (рис. 46).

Рис. 46: ДСУ: Поле формы

- Методы формы.
  При наведении на «methods» появляются кнопки: создания, копирования (рис. 47).

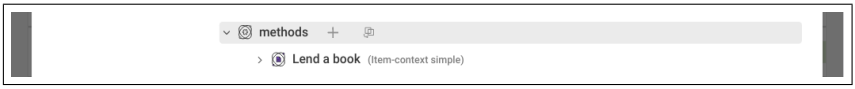

Рис. 47: ДСУ: Методы формы

- Метод формы.
  При наведении на метод формы появляется кнопка редактирования (рис. 48).

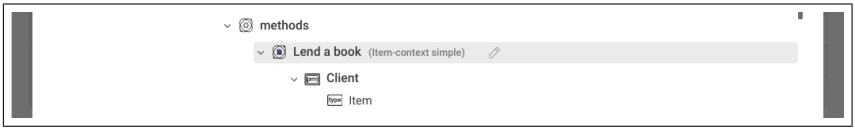

Рис. 48: ДСУ: Метод формы

- Статусы формы.
  При наведении на «states» появляются кнопки: создания, копирования (рис. 49).

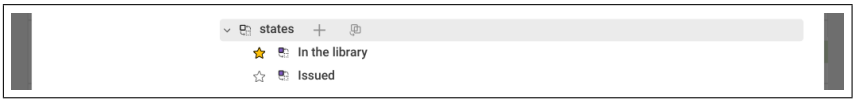

Рис. 49: ДСУ: Статусы формы

- Статус формы.
  При наведении на статус формы появляются кнопки: редактирования, удаления (рис. 50).

Рис. 50: ДСУ: Статус формы

- Структуры.
  При наведении на «structures» появляются кнопки: создания, копирования (рис. 51).

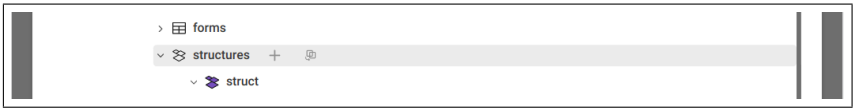

Рис. 51: ДСУ: Структуры формы

- Структура.
  При наведении на структуру появляются кнопки: редактирования, удаления (рис. 52).

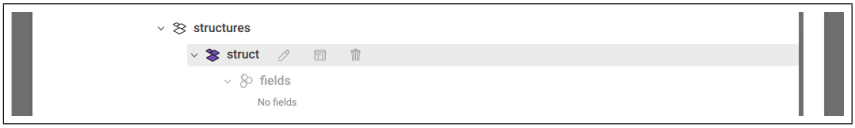

Рис. 52: ДСУ: Структура формы

- Трейты.
  При наведении на «traits» появляются кнопки: создания, копирования (рис. 53).

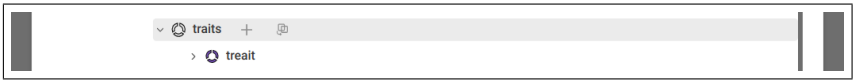

Рис. 53: ДСУ: Трейты формы

- Трейт.
  При наведении на трейт появляются кнопки: редактирования, удаления (рис. 54).

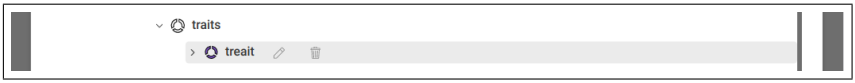

Рис. 54: ДСУ: Трейт формы

- Юнионы.
  При наведении на «unions» появляются кнопки: создания, копирования (рис. 55).

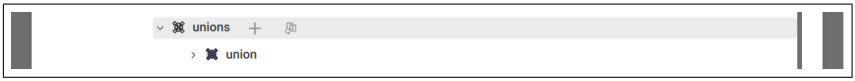

Рис. 55: ДСУ: Юнионы формы

- Юнион.
  При наведении на юнион появляются кнопки: редактирования, удаления (рис. 56).

Рис. 56: ДСУ: Юнион формы

Если включена настройка «Show deleted», то становится доступным следующий функционал древовидного интерфейса управления Meta-объектами:

- Восстановление удаленных объектов.
  При наведении на удаленный объект появляется кнопка восстановления (рис. 57). Для восстановления объекта необходимо подтвердить изменения в модальном окне представленном на рис. 58.

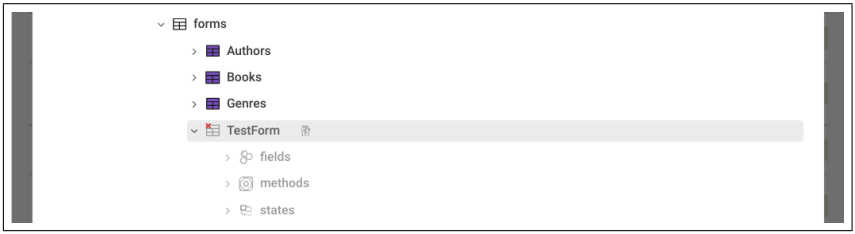

Рис. 57: Восстановление удаленных объектов

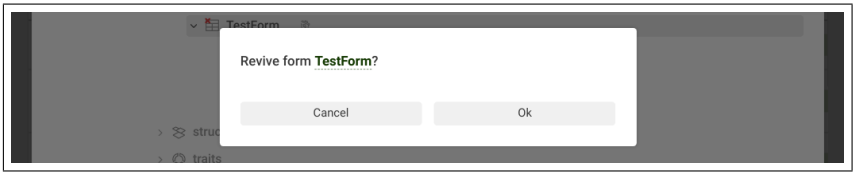

Рис. 58: Модальное окно подтверждения восстановления

Если включена настройка «Show ikeys», то у всех объектов отображаются идентификаторы (рис. 59).

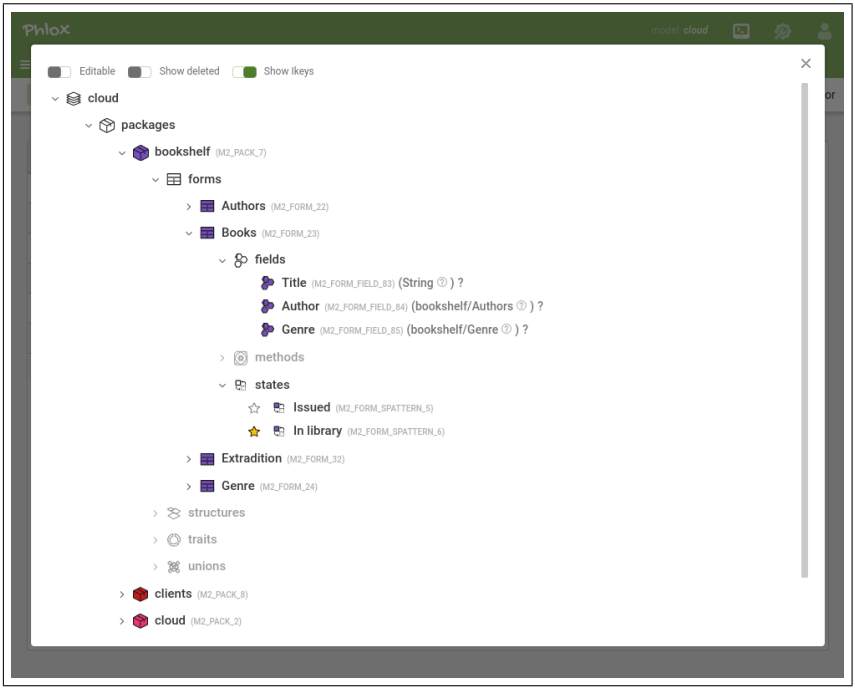

Рис. 59: Отображение идентификаторов у объектов

9. Управление запросами

Selection — это объект, который хранит в себе запрос к модели данных, написанный с помощью навигационного языка запросов к объектным моделям, при его выполнении возвращаются отобранные объекты.

9.1. Создание запроса

Для создания нового запроса нужно:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);
2. включить переключатель в позицию «Selections» (рис. 60, п. №1);
3. в открывшемся окне управления запросами нажать на кнопку «+» (рис. 60, п. №2);
4. в открывшихся полях ввода ввести читаемое название (title) запроса и код запроса;
5. создать запрос, нажав кнопку «Create».

9.2. Редактирование запроса

Для редактирования раннее созданного запроса необходимо:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);
2. включить переключатель в позицию «Selections» (рис. 60, п. №1);
3. в открывшемся окне управления запросами выбрать нужный запрос, нажать на кнопку «⋮», затем выбрать функцию «Edit» (рис. 60, п. №3);
4. в полях всплывающего окна «Selection edititng» отредактировать читаемое имя (title) и/или код запроса (рис. 61);

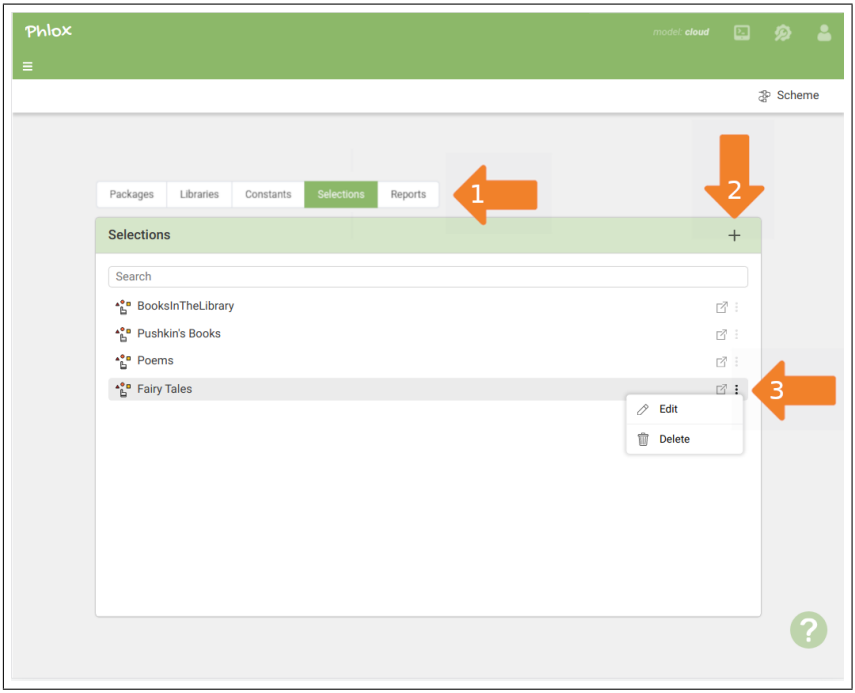

Рис. 60: Окно управления запросами

5. завершить редактирование запроса, нажав кнопку «Ok».

9.3. Удаление запроса

Для удаления запроса требуется:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);
2. включить переключатель в позицию «Selections» (рис. 60, п. №1);
3. в открывшемся окне управления запросами выбрать нужный запрос, нажать на кнопку «⋮», затем выбрать функцию «Delete»

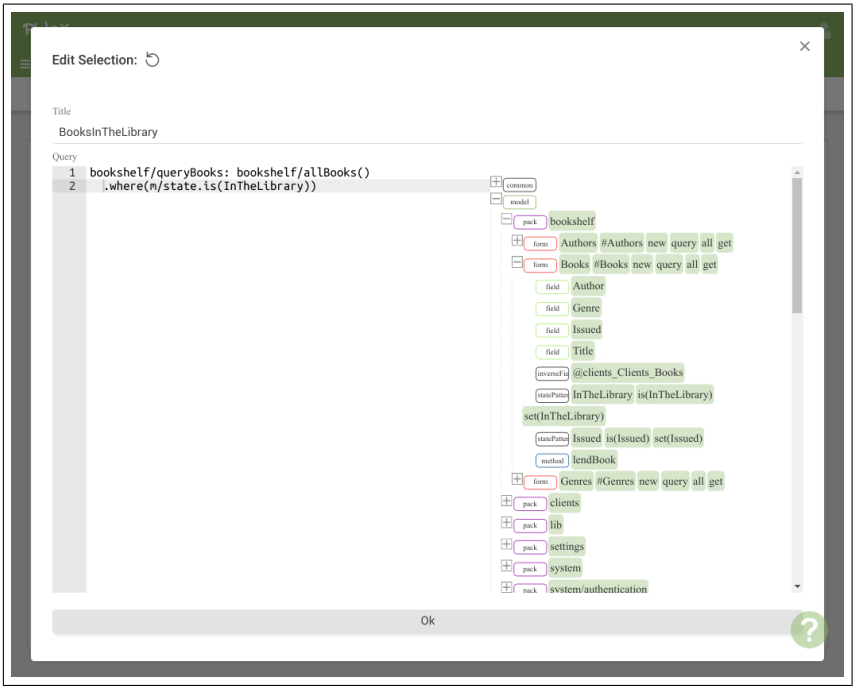

Рис. 61: Окно редактирования запроса

(рис. 60, п. №3);

4. подтвердить удаление в модальном окне, нажав кнопку «Ok».

После подтверждения удаления запрос будет удален. Во всех списках удаленный запрос перестанет отображаться. Если была открыта вкладка с данным запросом, то она автоматически закроется.

9.4. Результат запроса

Результат запроса представлен на отдельной вкладке. Функциональные возможности данной вкладки:

1. редактирование запроса;

2. редактирование колонок запроса;

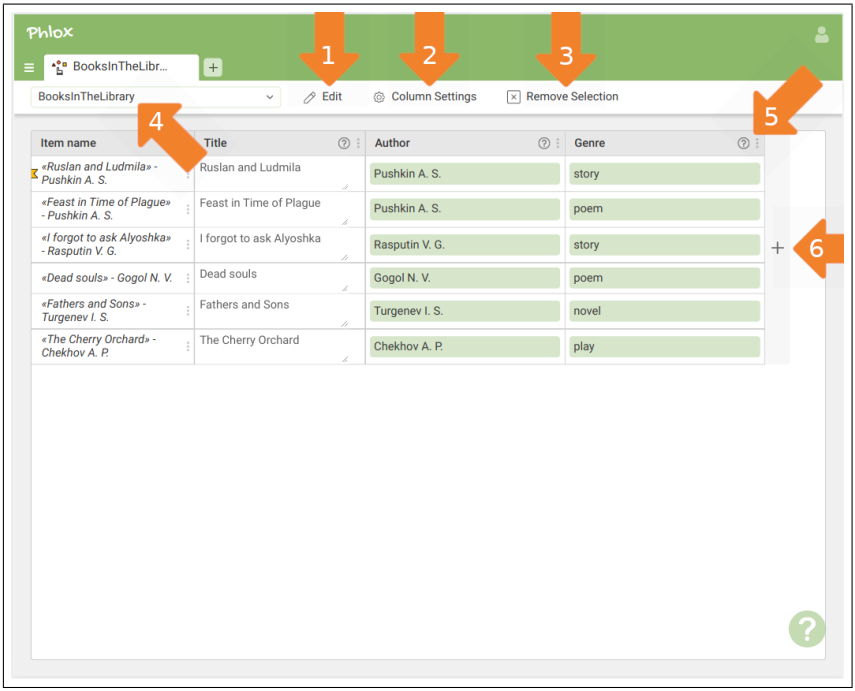

Рис. 62: Представление запроса на отдельной вкладке

3. удаление запроса;
4. выбор другого запроса для отображения на текущей вкладке;
5. редактирование, удаление и просмотр информации о колонке запроса;
6. добавление новой колонки;
7. редактирование полей документа.

9.5. Управление полями запроса

При создании запроса автоматически создаются и его колонки, по умолчанию колонки запроса идентичны полям формы.

9.5.1. Создание колонок запроса

Данная функция позволяет добавить в запрос новую колонки. Можно добавить колонку на основе уже существующего поля формы, либо создать псевдополе.

Для добавления в запрос новой колонки необходимо:

1. нажать на «+» с правой стороны таблицы результата запроса;
2. в появившемся окне «Create a new column» (рис. 63) ввести читаемое имя колонки (title to display) ;

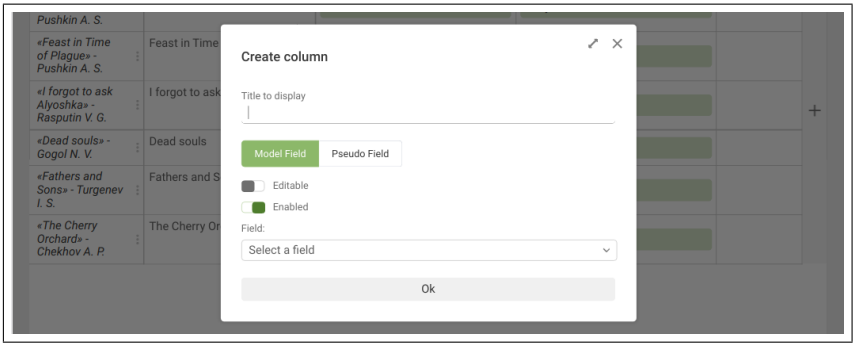

Рис. 63: Окно создания колонки запроса формата «Model Field»

3. выбрать формат колонки «Model Field» или «Pseudo Field»;
   - для колонки формата «Model Field»:
     - раскрыть выпадающее меню «Select a field» (рис. 64) и выбрать необходимое поле формы;
     - если необходимо запретить редактирование значений поля документа в данной колонке, то нужно выключить опцию «Editable»;
   - для колонки формата «Pseudo Field»:
     - заполнить поле «Evaluated Expression», представлено на рис. 65;
4. если необходимо чтобы данная колонка не отображалась в таблице результата, то нужно выключить опцию «Enabled».

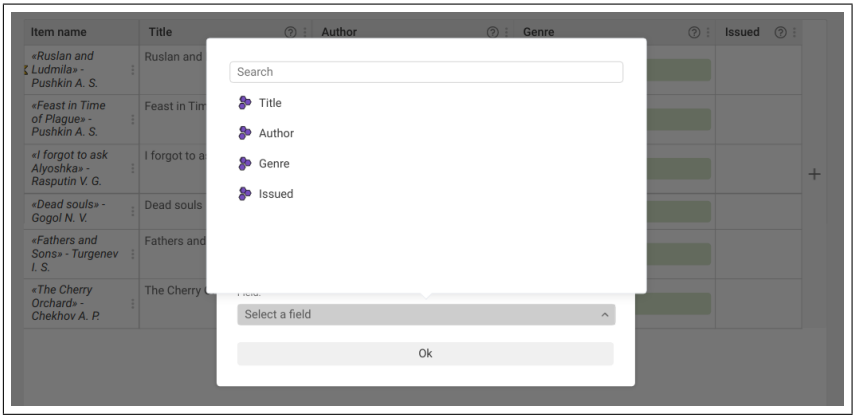

Рис. 64: Селектор полей

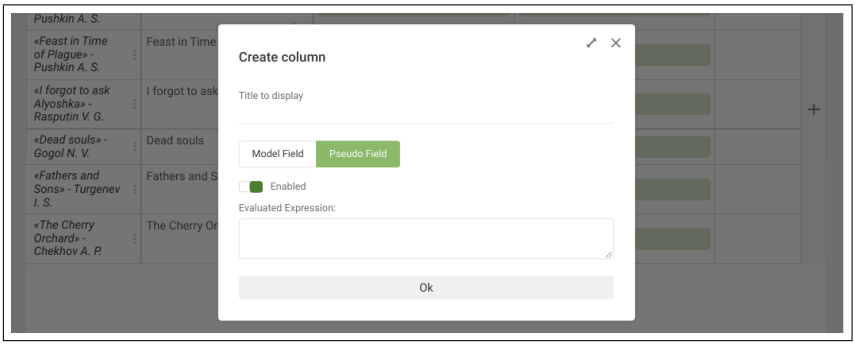

Рис. 65: Окно создания колонки формата «Pseudo Field»

9.5.2. Редактирование колонки запроса

Для редактирования колонки запроса нужно:

1. нажать на кнопку «⋮» (в правом углу наименования колонки, затем выбрать функцию «Edit» (рис. 66);
2. в появившемся окне «Column editing» откорректировать необходимые параметры колонки (читаемое имя, формат, код псевдополя, используемое поле формы, опции Editable, Enabled);

3. завершить редактирование колонки запроса, нажав кнопку «Ok».

После редактирования колонки запроса изменения стразу отобразятся в таблице.

9.5.3. Удаление колонок запроса

Для удаления колонки запроса требуется:

1. нажать на кнопку «⋮» в правом углу наименования колонки, затем выбрать функцию «Delete» (рис. 66);

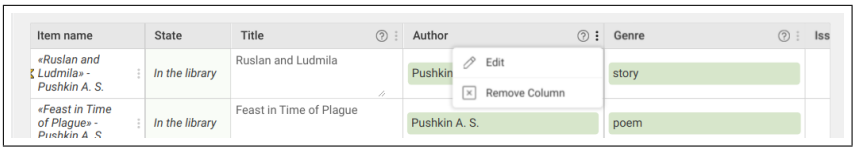

Рис. 66: Выпадающий список настроек колонки

2. подтвердить удаление в модальном окне, нажав кнопку «Ok».

9.5.4. Редактирование отображения колонок и их последовательности

Данная функция позволяет включить/выключить отображение колонок в результате запроса, поменять последовательность колонок и включить/выключить отображение статуса документов.

Для того чтобы открыть настройки колонок, нужно:

1. нажать кнопку «Edit» на панели управления запросом на вкладке запроса (рис. 62, п. №1);

2. во всплывающем окне «Column Settings» внести необходимые изменения;

3. для настройки отображения статуса документов включить / выключить поле «Show states in table» (рис. 67, п. №1);

4. для настройки отображения колонок включить/выключить переключатель напротив имени колонки (рис. 67, п. №2);

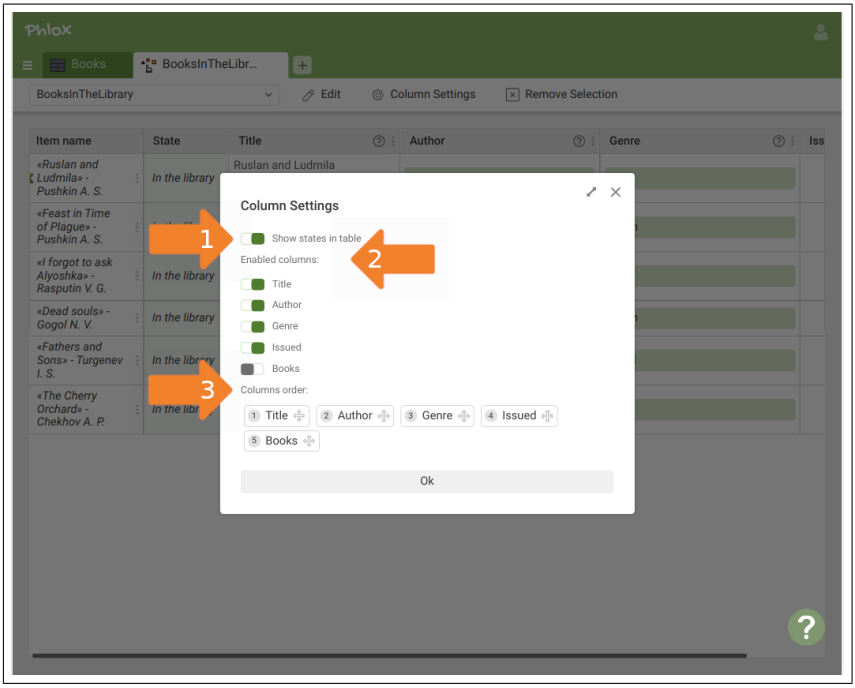

Рис. 67: Окно настроек отображения колонок

5. для того, чтобы поменять последовательность колонок, нужно перетащить выбранную колонку на другую позицию (рис. 67, п. №3);

6. завершить редактирование, нажав кнопку «Ok».

После редактирования таблица с результатом запроса автоматически перестраивается.

10. Отчеты

Важной составляющей платформы Ontobox является возможность эффективного построения отчетов, которые играют критическую роль в сборе, анализе и представлении данных.

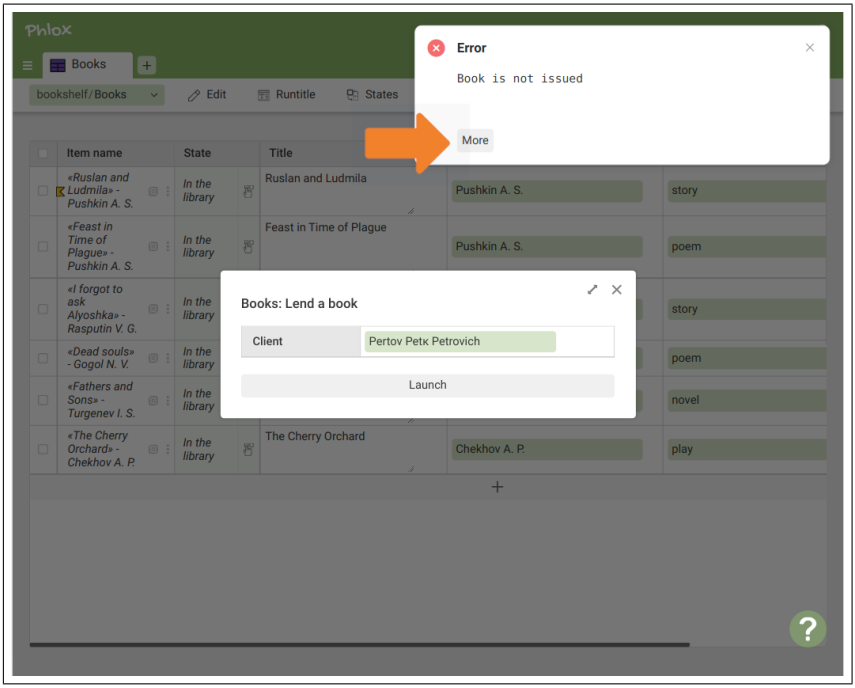

Рис. 68: Всплывающее окно с ошибкой

Построение отчетов с помощью визуализации Phlox имеет много преимуществ. Во-первых, это обеспечивает упрощение процесса создания отчетов, поскольку не требует подробного знания языков программирования. С помощью визуальных инструментов и шаблонов пользователи могут создавать сложные отчеты без необходимости написания сложного кода.

Во-вторых, визуальные средства позволяют легко манипулировать и изменять структуру отчета. Это означает, что пользователи могут быстро адаптировать отчеты к изменяющимся требованиям бизнеса или проекта.

При построении отчетов в системе основной акцент делается на использовании объектных моделей для представления данных. Это позволяет пользователям сфокусироваться на бизнес-логике и представлении данных, вместо того чтобы заниматься непосредственным

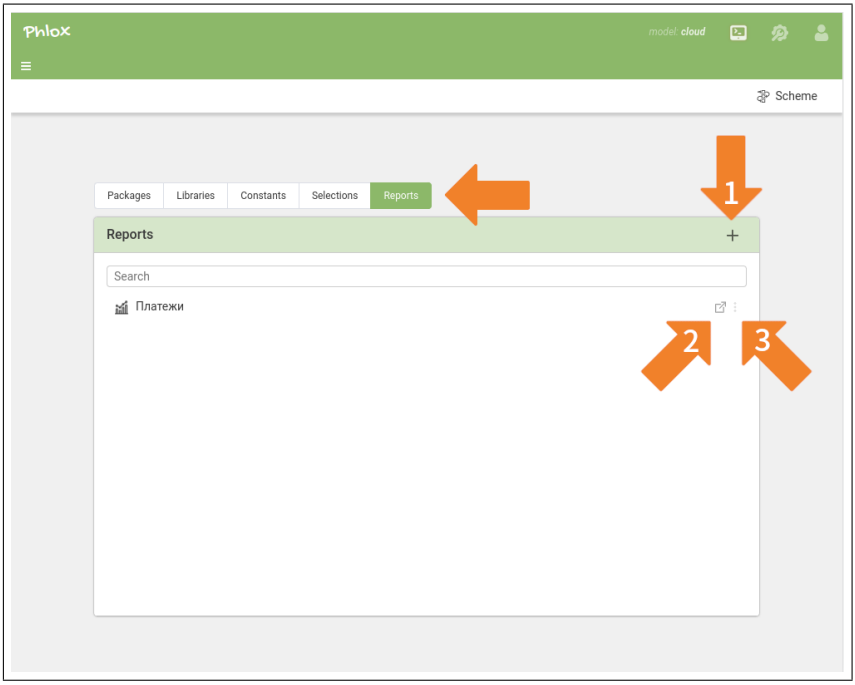

Рис. 69: Вкладка управления отчетами

написанием кода.

В этой главе мы подробно рассмотрим методы и техники построения отчетов.

10.1. Панель управления отчетами

Для того чтобы начать работу с отчётами необходимо открыть вкладку «Reports» (рис. 69) в начальном окне управления моделью. Данная панель управления отчетами предоставляет следующий функционал:

1. создать новый отчет;
2. открыть отчет в новой вкладке;
3. открыть настройки отчета.

10.2. Создание отчета

Для создания нового отчета требуется:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);
2. перейти на вкладку управления отчетами «Reports»;
3. в открывшемся окне управления отчетами нажать на кнопку «+» ;

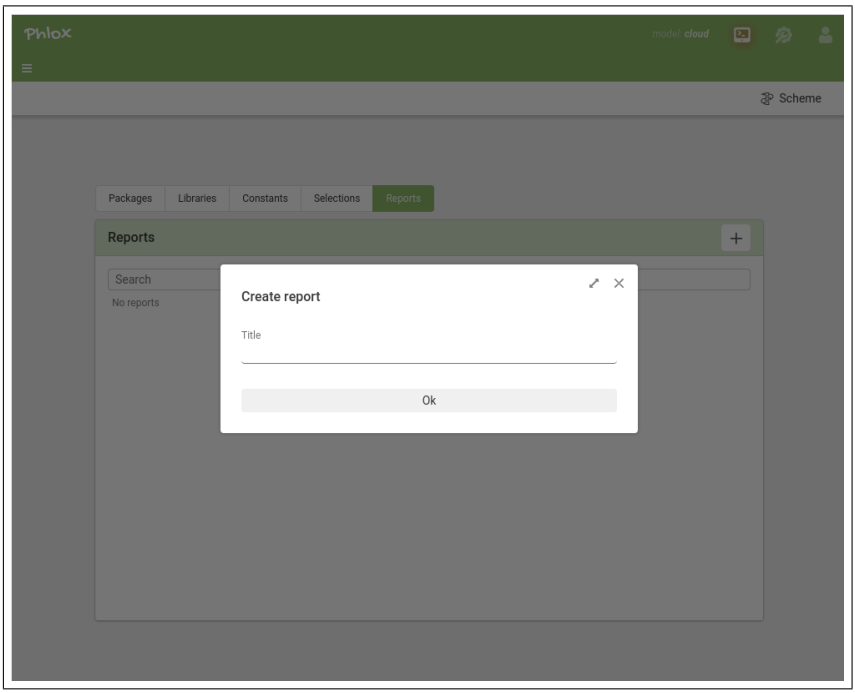

Рис. 70: Окно создания отчета

4. в открывшемся окне заполнить читаемое название (title) (рис. 70);
5. создать отчет, нажав кнопку «Ok».

Вновь созданный отчет появится в списке отчетов.

10.3. Редактирование отчета

Для отображения отчета также предусмотрена специальная вкладка (рис. 71).

Функционал вкладки «отчет»:

1. настройка аргументов отчета;
2. настройка запросов отчета;
3. настройка структуры отчета;
4. удалить отчет;
5. выбрать структуру и запустить выполнение отчета.

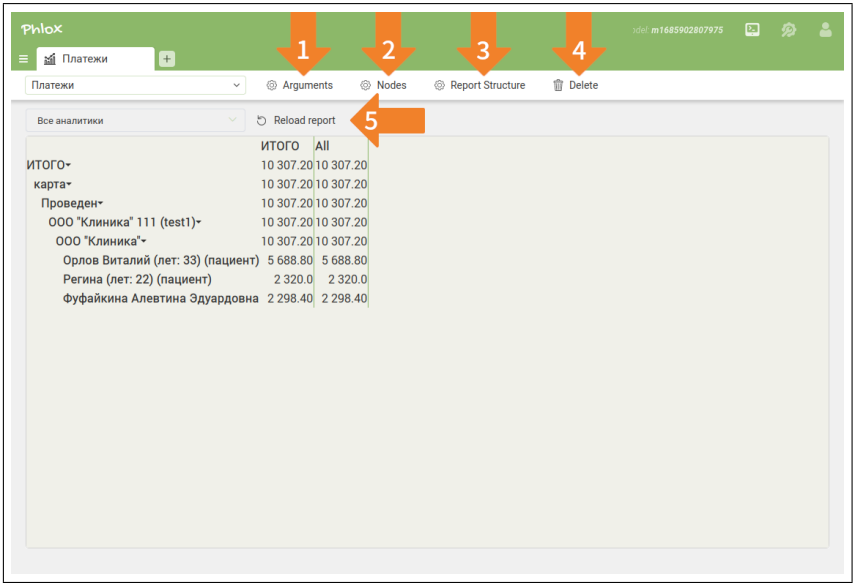

Рис. 71: Вкладка отчета

10.3.1. Настройка аргументов

Создание отчета требует определенного набора критериев, которые руководствуют его структурой и содержанием. В рамках данной платформы эти критерии называются аргументами. Аргументы в данном случае служат инструментом для определения специфики отчета.

Аргументы бывают двух типов:

- Analytics — конкретные значения по которым группируются данные;
- Indicators — накопители, вычисляющие различные суммы.

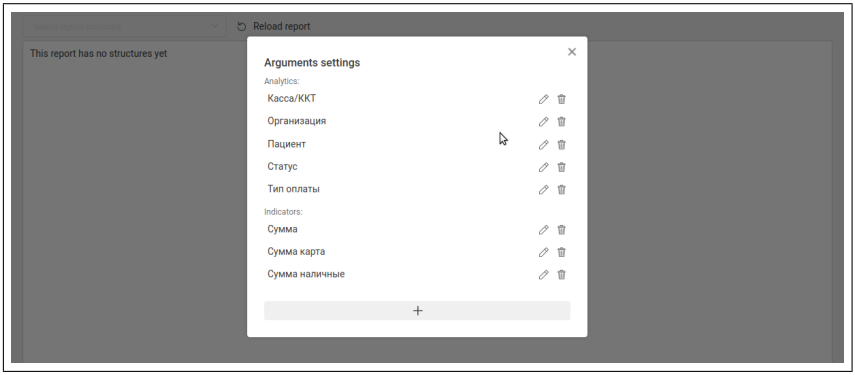

Рис. 72: Окно редактирования аргументов

Для добавления нового аргумента необходимо:

1. нажать на «+» в окне редактирования аргументов (рис. 72);
2. в появившемся окне «Create argument» (рис. 73) выбрать тип аргумента Analytics/Indicators (type) и ввести читаемое имя (title);
3. создать аргумент, нажав кнопку «Ok».

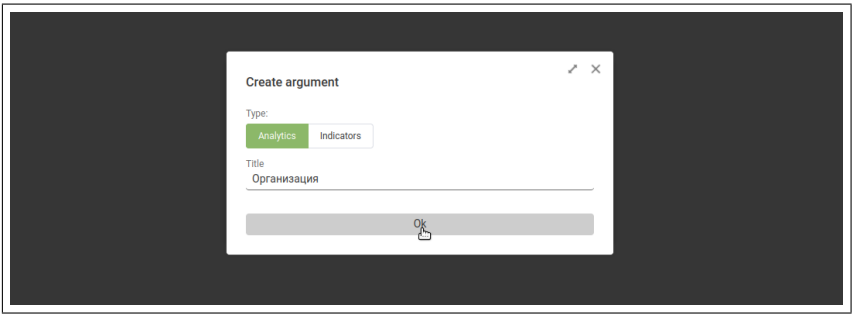

Рис. 73: Окно создания аргумента

10.3.2. Настройка узлов сборки данных

После того как определены аргументы для формирования отчета, следующим важным этапом является определение механизмов для сбора данных для каждого аргумента.

Каждый аргумент может иметь свой уникальный набор источников данных. Например, для аргумента, связанного с финансовыми показателями, источниками данных могут быть формы с финансовыми документами. В то время как для аргументов, связанных с производственными метриками, потребуются данные из форм управляющими производством или учетом рабочего времени.

Узел сборки данных включает в себя исходный запрос, который служит отправной точкой для дальнейшего извлечения данных, необходимых для аргументов.

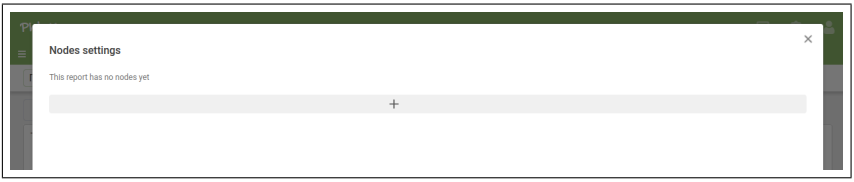

Рис. 74: Окно настройки узлов

Для добавления нового узла сборки данных нужно:

1. открыть окно настройки узлов, нажав кнопку «Nodes» на вкладке управления отчетом (рис. 71, п. №2).

2. нажать на «+» в окне настройки узлов (рис. 74);
3. в появившемся окне «Create nodes» (рис. 75) ввести запрос для получения корневых данных (query);
4. создать узел, нажав кнопку «Ok».

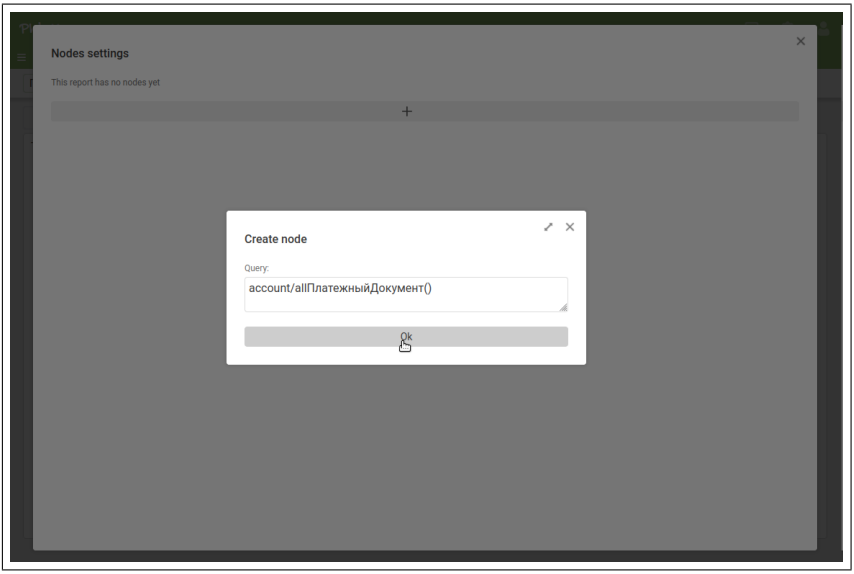

Рис. 75: Окно создания нового узла

После формирования узла сборки данных, следующим важным шагом является предоставление всей необходимой информации, которая требуется для сбора данных для аргументов.

Для аргументов с типом «Analytics» необходимо заполнить следующие значения (рис. 76, п. №1):

- Value — запрос для получения данных / значение, которое будет отображаться в теле отчета;
- Title — запрос для получения данных / значение, которое будет отображаться в названии колонки;
- Order — запрос для получения данных / значение по которому будет происходить сортировка значений в таблице отчета.

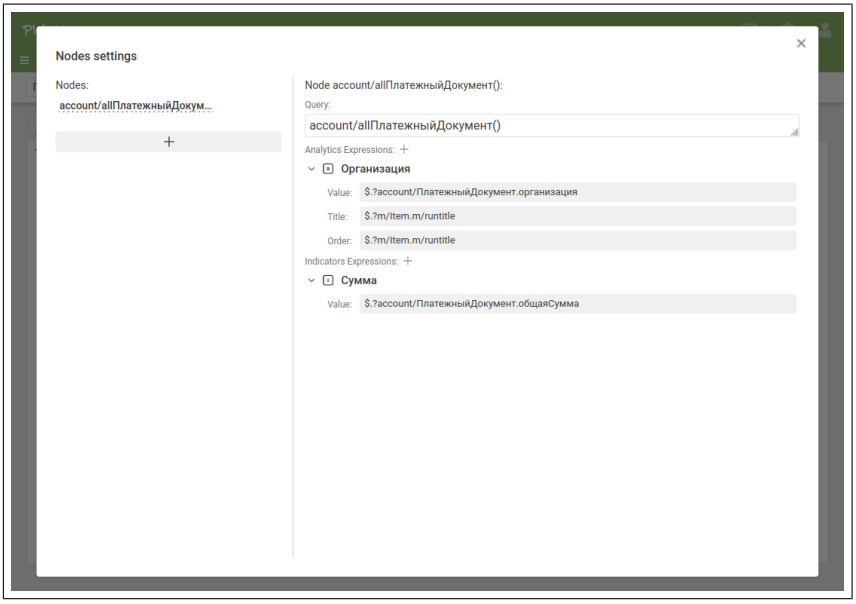

Рис. 76: Окно редактирования элементов узла

Для аргументов с типом «Indicators» необходимо заполнить следующие значения (рис. 76, п. №2):

- Value — запрос для получения данных / значение, которые будут суммироваться в отчете.

10.3.3. Настройка структуры отчета

Последний шаг в процессе формирования отчета — создание его структуры. Определение структуры отчета предполагает распределение аргументов по вертикальной и горизонтальной осям. Это определит, какие данные будут представлены по строкам, а какие — по столбцам отчета.

Не только выбор аргументов имеет значение, но также и их расположение, так как это влияет на восприятие и интерпретацию информации в отчете. Помимо этого, стоит учесть, что расположение аргументов может влиять на удобство проведения сравнительного анализа или на отслеживание определенных трендов.

Кроме аргументов, в структуре отчета должны быть указаны используемые индикаторы. Индикаторы представляют собой числовые значения, которые отражают определенный аспект или динамику исследуемой ситуации или процесса. Их выбор зависит от целей отчета и информации, которую нужно выразить.

Все эти элементы в совокупности образуют структуру отчета, которая помогает организовать данные таким образом, чтобы они были понятными, доступными и информативными для восприятия. Правильно сформированная структура облегчает восприятие информации и позволяет получить максимально объективное и полное представление о ситуации или процессе, которые анализируются в отчете.

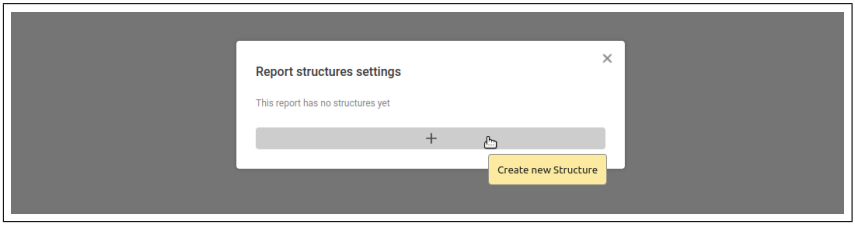

Рис. 77: Окно настройки структуры

Для добавления новой структуры отчета требуется:

1. открыть окно настройки структур отчета, нажав кнопку «Structure» на вкладке управления отчетом (рис. 71, п. №3).

2. нажать на «+» в окне настройки структур отчета (рис. 77);

3. в появившемся окне «Report Structure» (рис. 78) заполнить читаемое название (title), выбрать аналитики для отображения по вертикали (vertical analytics), выбрать аналитики для отображения по горизонтали (horizontal analytics), выбрать индикаторы (indicators);

4. создать структуру, нажав кнопку «Ok».

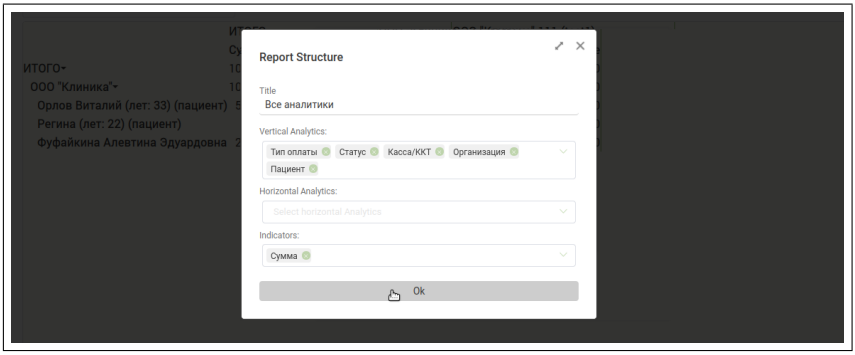

Рис. 78: Окно создания структуры

10.4. Удаление отчета

Для удаления раннее созданного отчета необходимо:

1. открыть начальное окно управления моделью, нажав на кнопку «+» основной панели (в случае отсутствия открытых вкладок окно открывается автоматически);
2. перейти на вкладку управления отчетами «Reports»;
3. в открывшемся окне управления отчетами выбрать нужный отчет, нажать на кнопку «⋮», затем выбрать функцию «Delete»;
4. подтвердить удаление в модальном окне, нажав кнопку «Ok».

После подтверждения удаления отчета, отчет будет удален. Во всех списках удаленный отчет перестанет отображаться. Если была открыта вкладка с данным отчетом, то она автоматически закроется.

11. Отображение ошибок

Если произошла какая-либо ошибка при выполнении метода, при получении документов, при расчете параметров и т.д., то появится всплывающее окно (рис. 68), уведомляющее о произошедшей ошибке. Всплывающее окно содержит краткое описание ошибки и кнопку «More», по клику на которую открывается дополнительное окно (рис. 79), содержащее StackTrace ошибки.

Определенные элементы StackTrace имеют интерактивные ссылки (рис. 79), по клику на которые осуществляется переход к указанному участку кода.

Рис. 79: StackTrase ошибки

Заключение

В данной работе описаны общие принципы построения объектных онтологий, их использования как ядра декларативного low-code и управления объектными онтологиями с помощью редактора Phlox в рамках платформы Ontobox. Сегодня редактор Phlox перешел в стадию промышленного применения. Например, на его основе разработана модель полной цифровизации медицинской клиники, включая расписание, управление ресурсами, рабочее место врача, CRM для взаимодействия с пациентами, финансовый блок, эквайринг и ряд других блоков. Суммарный объем кода приложения управления клиникой, автоматически генерируемый платформой Ontobox, составляет около 400 тысяч строк Libretto.

В настоящее время ведется работа по встраиванию в платформу Ontobox средств логико-вероятностного искусственного интеллекта. Это ставит интересные задачи использования визуальных low-code средств для работы в гибридной среде, соединяющей объектные онтологии и логико-вероятностный ИИ, к разработке которых мы недавно приступили.

Список литературы

1. Витяев Е. Е. Извлечение знаний из данных. Компьютерное познание. Моделирование когнитивных процессов / Е. Е. Витяев. — Новосибирск : НГУ, 2006. — 293 с.

2. Витяев Е. Е. Семантический вероятностный вывод предсказаний // Известия Иркутского государственного университета. Серия Математика. 2017. Т. 21. С. 33-50. https://doi.org/10.26516/1997-7670.2017.21.33

3. Гаврилина Д. Э., Манцивода А. В. Low-code и объектные электронные таблицы // Известия Иркутского государственного университета. Серия Математика. 2022. Т. 39. С. 93-103. https://doi.org/10.26516/1997-7670.2022.39.1

4. Гибридная технология построения интеллектуальных приложений. White Paper технологии.

5. Elma365: Low-code платформа для автоматизации внутренних бизнес-процессов и CRM // https://elma365.com/ru/ (accessed 1 April 2024).

6. Малых А.А., Манцивода А.В. Документное моделирование // Известия Иркутского. госуниверситета. Серия Математика, 2017, Т. 21, С.89–107. https://doi.org/10.26516/1997-7670.2017.21.89

7. Appian, a Low-Code Platform and Solutions. Available at: https://appian.com/ (accessed 1 April 2024).

8. Arul S. IT Professionals and DevOps Say No to Low-Code// AIM, January 2022. Available at: https://analyticsindiamag.com/it-professionals-and-devops-say-no-to-low-code/ (accessed 1 April 2024).

9. Business Process Model and Notation (BPMN). Version 2.0. Available at: https://www.omg.org/spec/BPMN/2.0/PDF (accessed 1 April 2024).

10. Ershov Yu. L., Goncharov S. S., Sviridenko D. I. Semantic Programming // Information processing 86: Proc. IFIP 10th

World Comput. Congress. Vol. 10. Elsevier Sci., Dublin, 1986. P. 1093–1100.

11. Ershov Yu. L., Goncharov S. S., Sviridenko D. I. Semantic Foundations of Programming // Fundamentals of Computation Theory: Proc. Intern. Conf. FCT 87, Lect. Notes Comp. Sci. Kazan, 1987. Vol. 278. P. 116–122. https://doi.org/10.1007/3-540-18740-5_28

12. Gartner Review: Magic Quadrant for Enterprise Low-Code Application Platforms. 20 Sep 2021. Available at: https://www.gartner.com/doc/reprints?id=1-27I04ZJT&ct=210921&st=sb (accessed 1 April 2024).

13. Goncharov S. S., Sviridenko D. I. Σ-programming, Transl. II // Amer. Math. Soc. 1989. N 142. P. 101–121.

14. Malykh A., Mantsivoda A. Query Language for Logic Architectures// Proceedings of 7th International Conference «Perspectives of System Informatics», Springer-Verlag, Lecture Notes in Computer Science 5947, 2010, P.294–305.

15. Mantsivoda A.V., Ponomaryov D.K. A Formalization of Document Models with Semantic Modelling // Известия Иркутского государственного университета. Серия Математика. 2019. Т. 27. С.36-54. https://doi.org/10.26516/1997-7670.2019.27.36

16. Mantsivoda A.V., Ponomaryov D.K. Towards Semantic Document Modelling of Business Processes // Известия Иркутского государственного университета. Серия Математика. 2019. Т. 29. С. 52-67. https://doi.org/10.26516/1997-7670.2019.29.52

17. Mantsivoda A.V., Ponomaryov D.K. On Termination of Transactions over Semantic Document Models // Известия Иркутского государственного университета. Серия Математика. 2020. Т. 31. С. 111-131. https://doi.org/10.26516/1997-7670.2020.31.111

18. Martin J., Gupta S., Kumar A. Scaling with Low Code to Accelerate Digital Transformation. HFS Research & Infosys. Published November 2021. Available at: https://www.infosys.com/services/digital-process-automation/insights/scaling-low-code.html (accessed 1 April 2024).

19. McCormick P. Excel Never Dies. The Spreadsheet That Launched A Million Companies. Not Boring, March 2021. Available at: https://www.notboring.co/p/excel-never-dies?s=r (accessed 1 April 2022).

20. OutSystems Low-Code Development Platform. Available at: https://www.outsystems.com/ (accessed 1 April 2024).

21. Wolff E. Microservices: Flexible Software Architecture. Addison-Wesley Professional, 2016, 432p.

22. Wayner P. Why Developers Hate Low-Code. 9 Reasons Programmers Grow Frustrated with the Tools That are Supposed to Save Them Time // InfoWorld, 2019, Sep. 16. Available at: https://www.infoworld.com/article/3438819/why-developers-hate-low-code.html (accessed 1 April 2024).

Серия

Дискретная математика и информатика

Гаврилина Д.Э.

Дизайнер объектных онтологий Phlox

Научное издание

Редакционно-издательский отдел
государственного образовательного учреждения
высшего профессионального образования
Иркутский государственный педагогический университет
Иркутск, Нижняя набережная, 6

!!!TODO!!!
Подписано в печать 15.05.2009г.
Формат бумаги 60×84 1/16. Объем 1,7 у. п.л.
Заказ 10. Тираж 100 экз.

Отпечатано на RISO
в ООП факультета МФИ ИГПУ