
Описание функций ПО "Платформа управления бизнес-процессами DocHub" (далее также - DocHub)

Дата актуализации: 2025-04-13 Версия: ДН3 (исходный код core)

[!info] Область описания Документ описывает базовое ядро DocHub. Прикладные решения, их метаданные и серверные модули могут расширять функции платформы и в состав ядра не входят.

1. Назначение

DocHub — серверная платформа для разработки и эксплуатации прикладных систем, которые работают со структурированными документами и реализуют бизнес-процессы. Поведение системы задаётся метаданными, а прикладная логика — серверными и клиентскими модулями на JavaScript.

Платформа разворачивается в инфраструктуре организации и подключается к одной или нескольким базам данных. Основные конфигурационные и системные сведения поставляются в виде YAML-файлов.

2. Модель платформы

2.1. Метаданные

В состав метамодели входят:

- типы объектов и описания их полей;
- пути хранения объектов;
- экранные формы и компоновки;
- действия и пункты меню;
- справочники;
- пользователи, группы и роли;
- источники данных, очереди, брокеры и задания;

- схемы workflow и сведения о шагах обработки;
- серверные и клиентские скрипты.

Метаданные описывают структуру объектов, правила проверки, расположение полей на формах, фильтры списков, доступные действия и права доступа. Исходный формат описаний — YAML, что позволяет хранить их в системе контроля версий.

2.2. Хранилища

Платформа поддерживает хранение объектов в PostgreSQL и Oracle. Для пути можно использовать таблицу или представление реляционной СУБД, локальное хранилище либо серверный модуль, реализующий виртуальное хранилище.

Для реляционных хранилищ предусмотрены:

- создание и обновление таблиц, индексов и последовательностей по метаданным;
- чтение, создание, изменение и удаление объектов;
- выборка списка по фильтру и подсчёт количества объектов;
- транзакции;
- блокировки документов;
- несколько источников данных.

3. Функциональные возможности

3.1. Работа с объектами

Пользовательский интерфейс и API поддерживают просмотр, создание, изменение и удаление объектов с учётом прав роли. Для списков доступны фильтрация, постраничный вывод, сортировка и настроенные действия.

Типы полей включают строки, даты и время, числа, суммы, логические значения, вложения, ссылки на объекты, таблицы и вложенные формы. Для полей могут быть заданы обязательность, значения по справочнику, длина, шаблон проверки, значение по умолчанию и сценарии обработки изменений.

3.2. Пользователи и права доступа

Система предоставляет:

- пользователей, группы и роли;
- разграничение доступа к путям и действиям;
- изменение пароля пользователя;
- аудит действий пользователей и системных операций;
- авторизацию через локальные данные и, при настройке, LDAP.

Меню и панель управления также являются настраиваемыми метаданными и могут быть ограничены требованиями роли.

3.3. Workflow, задания и блокировки

Workflow задаёт состояния объекта, события перехода и действия, выполняемые на отдельных шагах. Платформа сохраняет сведения о ходе обработки и поддерживает переходы по событиям, автоматизированные действия и бизнес-журнал.

В ядре также предусмотрены:

- внутренняя очередь заданий;
- планировщик периодических заданий;
- блокировка документа на время редактирования;
- хранение и обработка задач;
- уведомления и архивирование сообщений.

3.4. Файлы и документы

К объектам можно прикреплять файлы, загружать их через web-интерфейс и скачивать с учётом прав доступа. Для служебной диагностики предусмотрено сохранение аудита операций, включая загрузку и скачивание файлов.

3.5. Интеграция

Взаимодействие с внешними системами выполняется через:

- HTTP/API платформы;
- настраиваемые серверные web-сервисы;
- очереди и брокеры сообщений;
- AMQP и IBM MQ-совместимый обмен, если соответствующие компоненты и настройки включены в поставке;
- Oracle AQ для настроенного типа очереди.

Входящие сообщения передаются серверной функции, а отправка сообщений доступна из серверных модулей через API ядра.

4. Программные модули

Серверные модули размещаются в каталоге `server_scripts` и выполняются в среде Node.js. Клиентские модули размещаются в `client_scripts` и выполняются в браузере пользователя.

Серверным скриптам доступны загрузка других модулей, журналирование, получение описаний типов и путей, операции с документами, транзакции и отправка сообщений. Клиентским скриптам доступны модели интерфейса, формы, пути, справочники, уведомления, диалоги и

вызов серверных сервисов.

5. Пользовательский интерфейс

Web-интерфейс реализован как одностраничное React-приложение с компонентами Blueprint. В интерфейсе поддерживаются:

- вход пользователя;
- главное меню и панель управления;
- одновременная работа с несколькими формами во вкладках;
- разделение рабочей области;
- просмотр и редактирование документов;
- таблицы со списками и фильтрами;
- формы печати и просмотра исходного представления;
- работа с вложенными файлами;
- диаграммы и специализированные формы, если они описаны в метаданных.

Web-сервер отдаёт статические ресурсы интерфейса и JSON-RPC/API-запросы. Для масштабирования web-обработчиков допускается запуск нескольких экземпляров за балансировщиком.

6. Компоненты runtime

Компонент	Назначение
<code>main</code>	Управление процессами и внутренний обмен между модулями
<code>web</code>	Web-интерфейс и API
<code>worker</code>	Обработка внутренней очереди заданий
<code>queue</code>	Интеграция с внешними очередями и брокерами
<code>scheduler</code>	Запуск периодических заданий

В конфигурации по умолчанию используется один web-worker и один task-worker. Количество экземпляров задаётся параметрами `maxWebWorkers` и `maxTaskWorkers`.