Описание

Модуль создания обычных и динамических отчетов

Настройки отчета

Описание шаблона отчета, его параметров и источники получения данных для отчета

Структура настроек

Описание отчета `JPReport`

Свойство	Описание
description	Описание отчета
qName	QName
code	Код отчета
template	Шаблон отчета
jpPackage	Доступ к отчету
titleTemplate	Шаблон имени итогового отчета
groupCode	Код группы отчетов (общие настройки, в том числе место хранения итогового файла)
type	Тип привязки отчета (список/объект)
jpClasses	Список метаклассов, для которых доступен этот отчет
sqlSources	SQL источники данных
javaSources	Java источники данных
params	Список параметров отчета
storageTime	Время жизни итогового файла
sqlFilter	SQL маскирование отчета
javaFilter	Java маскирование отчета

Описание параметра отчета `JPReportParam`

Свойство	Описание
code	Название(Код) параметра
value	Значение параметра

Свойство	Описание
type	Тип параметра
length	Длина (для строковых полей)
description	Наименование
qName	QName
multiple	Признак множественного выбора
mandatory	Признак обязательности
external	Возможность внешнего переопределения параметра
refJpClass	Класс из меты
refJpAttr	Атрибут класса
refFilter	JSON фильтрации объектов
enums	Возможные перечислимые параметра

Описание перечислимого значения `JPReportEnum`

Свойство	Описание
value	Значение
description	Описание значения
qName	Полный код значения

Привязка отчета

Свойство	Описание
object	Обработка 1 объекта
list	Обработка списка объектов
uni	Обработка объекта и списка объектов
custom	Произвольная

Способы описания настроек отчета

XML

- xml файл, указанной структуры, должен быть размещен в src/main/resources/reports/settings

```
<JReports>
  <JReport>
    <description>Test report chart xlsx</description>
    <qName>testReport.chartXlsx</qName>
    <code>testReport_chartXlsx</code>
    <template>report.xlsx</template>
    <jpPackage>onlyReadonly</jpPackage>
    <titleTemplate>Test report chart xlsx</titleTemplate>
    <groupCode>common</groupCode>
    <type>custom</type>
    <jpClasses>class1,class2</jpClasses>
    <sqlSources>
      <sqlSource rsCode="Header"/>
      <sqlSource storage="main" sql="report.sql" rsCode="Incomes" />
      <sqlSource rsCode="Footer"/>
      <sqlSource rsCode="Chart"/>
    </sqlSources>
  <javaSources>
    <javaSource bean="mp.jprime.reports.services.DefaultReportDataGenerator" code="Main2" />
  </javaSources>
    <params>
      <param code="objectClassCode" type="string" multiple="false" mandatory="false" external="true"/>
      <param code="objectIds" type="string" multiple="true" mandatory="false" external="true"/>

      <param code="ext" value="xlsx" type="string" length="10" description="Расширение" qName="report.
             multiple="false" mandatory="true" external="true">
          <paramEnums>
            <paramEnum value="xlsx" description="XLSX" qName="report.ext.enum.xlsx"/>
            <paramEnum value="pdf" description="PDF" qName="report.ext.enum.pdf"/>
          </paramEnums>
      </param>

      <param code="param1" value="" type="integer" description="Параметр1" qName="report.param1"
             multiple="false" mandatory="false" external="true"/>
      <param code="param2" value="" type="date" description="Параметр2" qName="report.param2"
             multiple="false" mandatory="false" external="true"/>
      <param code="param3" value="" type="boolean" description="Параметр3"
             qName="report.param3" multiple="false" external="true"/>
      <param code="param4" value="" type="boolean" description="Параметр4"
             qName="report.param4" multiple="false" external="false"/>
    </params>
    <storageTime>day</storageTime>
  </JReport>
</JReports>
```

Параметры отчета

Особые параметры

`objectClassCode` и `objectIds` заполняются данными объектов, для которых вызывается отчет.

- objectClassCode - кодовое имя метакласса объекта/ов
- objectIds - идентификатор или идентификаторы объектов через запятую, в случае списка

Параметры с настройкой external=true

Значения таких параметров может быть указано из внешних источников. Пользователем или в прикладном коде
Значение прочих параметров не изменяются

Источники данных

Размещение ресурсов (sql, шаблоны)

Файлы шаблонов и sql-запросов следует размещать в `classpath:reports/storage/<reportCode>/` .

Допускается размещение файлов шаблонов без расширения или с расширением `.jrxml` в `classpath:reports/templates/` , а файлов sql-запросов в `classpath:reports/sql/ *_`(для старых отчетов)_.

sql source

```
 <sqlSources>
   <sqlSource storage="main" sql="report.sql" rsCode="Incomes" />
 </sqlSources>>
```

Использует данные, полученные запросом `sql` в хранилище `storage` , под именем `rsCode`

java source

```
<javaSources>
  <javaSource bean="mp.jprime.reports.services.DefaultReportDataGenerator" code="Main2" />
</javaSources>
```

Использует данные, генерируемые в наследнике от `mp.jprime.reports.ReportDataGenerator` , под именем `code`

Создание отчета

Для формирования отчета необходимо наличие настройки JPReport, содержащей шаблон отчета, запросы для получения данных и описание параметров

Формирование отчета в прикладном коде осуществляется, например, следующим кодом:

```
 JPReport s = reportSettingStorage.getReportSettingByCode("<код отчета>");

ReportStream rr = s != null ? reportService.getReportStream(s, Collections.emptyList()) : null;
if (rr != null) {
  try (InputStream is = rr.getIs()) {
    FileUtils.copyInputStreamToFile(is, new File("E:\\report_2.xlsx"));
  }
}
```

Генератор отчета

В JPrime осуществленна поддержка двух генераторов отчетов

- JasperReports
- yarg

При указании шаблона отчета без расширения или с расширением `.jrxml` создание отчета будет производиться с помощью библиотеки `JasperReports`

При наличии расширения, отличного от `.jrxml` и соответствующего общеупотребимым форматам (doc,docx,xls,xlsx,pfd,csv,html) за создания отчетов будет отвечать шаблонизатор `YARG`

JasperReports генератор отчетов

YARG генератор отчетов

Настройка шаблонов и разметка полей для корректной работы должны соответствовать [правилам] (https://doc.cuba-platform.com/reporting-7.0-ru/reporting.html#template)

При необходимости генерации pdf (конвертации прочих форматов в pdf) в настройках приложения необходимо указать настройку `jprime.openoffice.path`, содержащую путь до установленного LibreOffice или OpenOffice

```
jprime:
  openoffice:
    path: C:\Shells\LibreOffice\program
    ports: 2002,2003
```

Количество портов соответствует количеству потоков и требует корректировки при увеличении нагрузки

Отчет по умолчанию, если основной отчет не сформировался

Если при формировании основного отчета произошла ошибка, будет сформирован отчет `jpDefaultErrorReport` с указанием кода основного отчета и времени ошибки

Параметры по-умолчанию

В SQL запрос, помимо явно указанных параметров отчета, всегда передаются базовые значения

Код	Описание	Тип
authInfoUserId	идентификатор пользователя	Строка
authInfoOrgId	организация пользователя	Строка
authInfoDepId	подразделение пользователя	Строка
authInfoUsername	логин пользователя	Строка

Функции

Если часть данных лежит физически в другой БД и нет возможности достать их одним sql-запросом вместе с остальными данными, то можно воспользоваться Функциями, которые не зависят от расположения данных и могут обращаться к другим сервисам для их получения.

Принцип работы

`JPReportFunctionBaseService` проверяет резалт-сет на наличие в значениях полей шаблонов, вызывающих доступные функции и, если находит, то вызывает их с переданными в шаблоне аргументами. Функция возвращает данные в виде мапы, где ключ - составной ключ из шаблона функции и значений всех аргументов, а значение - результат функции. После этого `JPReportFunctionBaseService` ещё раз проходит по резалт-сету и подменяет шаблоны на результат функции.

Для вызова функции необходимо включить в sql-запрос константу вида:

```
'${<code>.<template>$<arg1>$<arg2>$...$<argN>}'
```

где `<code>` - это код функции,

`<template>` - один из шаблонов функции,

а `<arg1>-<argN>` - аргументы, передаваемые в функцию (порядок важен!). Могут быть переданы по ссылке (на поле из этого же запроса) или по значению (в запросе не должно быть полей совпадающих по имени с этим значением).

Пример:

- передача аргумента по ссылке:

```
select worker_id, '${staffTitle.position$worker_id}' as position from ...
```

в функцию `staffTitle`, у которой есть шаблон `position`, будет передан в качестве аргумента значение поля `worker_id`, после чего в поле `position` будет положен результат работы функции.

- передача аргумента по значению:

```
select worker_id, '${staffTitle.position$123}' as position from ...
```

в функцию `staffTitle`, у которой есть шаблон `position`, будет передано в качестве аргумента `123`, после чего в поле `position` будет положен результат работы функции.

Создание функций

Для создания функции необходимо унаследовать абстрактный класс `JPReportBaseFunction`.

- `getData(args, auth)` получает данные и возвращает их в виде мапы, где ключ - это строка, составленная из значений всех аргументов (порядок важен!), а значение - `JPObject`.
- `compute(jpObject, template)` вычисляет итоговую строку, которую необходимо будет подставить в резалт-сет, из переданного `jpObject` на основании переданного `template`.
- Имя аргумента `template` зарезервировано для передачи шаблона функции!