Библиотека по упрощенной работе с JavaCoreServer

Для упрощенной работы с Java-сервисами Saperion была разработана специальная библиотека, позволяющая работать с Saperion через Java приблизительно так же, как и через COM. В библиотеке были реализованы такие классы, как: Application, Cursor, Document, File, использование которых покрывает большинство потребностей в разработке под Saperion.

Для оценки упрощения работы ниже представлен пример кода для сохранения файлов документа, полученного по запросу из Сапериона:

|  |  |
| --- | --- |
| Сохранение через обычные методы  Saperion Java Classic Connector | Сохранение с использованием библиотеки  ru-saperion |
| SaClassicConnector connector = ru.elar.tools.saperion.SaClassicConnectorCreator.GetConnectorInstance("saperion.properties");  connector.logon("Администратор", "1", 3, "");  SaQueryInfo info = new SaQueryInfo("from inputtray");  List<SaDocumentInfo> docs = connector.searchHQL(info);  if (docs != null && docs.size() > 0)  {  for (SaDocumentInfo doc: docs)  {  SaPropertyValue valXHDOC = doc.getValue("$HDOC");  String xhdoc = valXHDOC.getStringValue();  SaDocInfo docInfo = connector.getDocumentInfo(xhdoc, true, true);  int elements = docInfo.getElementCount();  for (int iElement = 1; iElement <= elements; iElement++)  {  SaDocInfo.ElementInfo element = docInfo.getElementInfo(iElement);  String fileName = element.getName();  long fileSize = element.getSize();  InputStream readStream = connector.readDocument(xhdoc, true, iElement);  byte[] content = new byte[(int)fileSize];  readStream.read(content);  FileOutputStream out = new FileOutputStream("D:\\test\\" + xhdoc + "\\" + fileName);  out.write(content);  out.flush();  out.close();  out = null;  }  }  } | ru.saperion.com.Application oApp = new ru.saperion.com.Application(Application.ConnectorType.CLASSIC, "saperion.properties");  oApp.Login("Администратор", "1", 3);  ru.saperion.com.Cursor cur = oApp.SelectHQL("inputtray", "");  if (cur.First())  {  do  {  ru.saperion.com.Document doc = cur.Document();  if (doc.Load())  {  doc.SaveAs(0, Document.SaveType.CLASSIC, "D:\\test\\");  }  } while(cur.Next());  } |

Как видно из примера – использование библиотеки значительно проще воспринимается для специалистов, работавших с Saperion через COM (а так же, программировавших во встроенной среде).

В данном документе вы найдете описание существующих на данных момент классов и методов библиотеки, а так же общее описание как собирать свои библиотеки, а так же подключать их к Saperion.

Оглавление

[Реализация COM API в JAVA 3](#_Toc434574895)

[Application 3](#_Toc434574896)

[Используемые типы-перечисления 3](#_Toc434574897)

[Методы 3](#_Toc434574898)

[Cursor 5](#_Toc434574899)

[Методы 5](#_Toc434574900)

[Document 6](#_Toc434574901)

[Используемые типы-перечисления 6](#_Toc434574902)

[Методы 6](#_Toc434574903)

[File 8](#_Toc434574904)

[Методы 8](#_Toc434574905)

[Вспомогательные утилиты 9](#_Toc434574906)

[DateWorker 9](#_Toc434574907)

[Используемые типы-перечисления 9](#_Toc434574908)

[Методы 9](#_Toc434574909)

[FileWorker 10](#_Toc434574910)

[Методы 10](#_Toc434574911)

[WebWorker 11](#_Toc434574912)

[Методы 11](#_Toc434574913)

[Работа с WEB-клиентомВспомогательные утилиты 12](#_Toc434574914)

[Field 12](#_Toc434574915)

[Используемые типы-перечисления 12](#_Toc434574916)

[Методы 12](#_Toc434574917)

[Mask 14](#_Toc434574918)

[Отчеты 16](#_Toc434574919)

[IReportLine 16](#_Toc434574920)

[ReportGenerator 18](#_Toc434574921)

[Csv 19](#_Toc434574922)

[Пример кода генерации отчета 20](#_Toc434574923)

# Реализация COM API в JAVA

## Application

Пространство имен: ru.saperion.com

Предназначение: Отправная точка для работы с Саперион. Поддерживает работу напрямую с Saperion Classic Connector и с Saperion Web Client.

### Используемые типы-перечисления

#### ConnectorType

Данный тип используется для определения типа подключения к Saperion. Включает в себя два варианта:

|  |  |  |  |
| --- | --- | --- | --- |
| Программное название | Описание | Текстовое значение | Числовое значение |
| WEB | Подключение через WEB-клиент: Не требует настроек подключения, авторизации; Требует вызов только при исполнении кода WEB-клиента. | WebClient | 1 |
| CLASSIC | Прямое подключение к Classic Connector: Требует наличия файла настроек с адресом сервера; Требует обязательной авторизации. | Classic | 0 |

### Методы

#### Конструктор Application(ConnectorType type, String settingsFile)

При создании экземпляра объекта типа Application происходит автоматическое подключение к службам Saperion. Для подключения необходимо передать два параметра.

**ConnectorType type** – определяет метод подключения к Saperion

**String settingsFile** – путь к файлу настроек для подключения к Classic Connector. Пример файла можно найти по адресу ***%SAPERION%\scr\scr-classicconnector\config\saperion.properties***. В параметре brokerhost= необходимо прописать адрес и порт сервера Saperion, к которому выполняется подключение.   
\*В случае использования подключения к WEB – в данный параметр передать пустую строку.

При ошибке в подключении будет выдана ошибка типа Exception с описанием причины ошибки.

#### Конструктор Application(SaClassicConnector connector)

Данный вариант конструктора следует использовать только тогда ,когда есть необходимость использовать Connector вне библиотеки. В этом случае при передачи его в параметр происходит привязка объекта Application к данному коннектору.

#### Метод Login(String login, String password, int licenseType)

Данный метод производит авторизацию в Saperion. Принимает три параметра:

**String login** – Имя пользователя

**String password** – Пароль пользователя

**Int LicenseType** – тип лицении пользователя

В случае отказа в авторизации будет выдана ошибка типа Exception с соответствующим сообщением (в лучших традициях COM).

#### Метод Logoff()

Данный метод предназначен для выхода пользователем из системы.

#### Функция SelectHQL(String definition, String query), возвращает Cursor

Данная функция является аналогом функции Application.SelectSQL из COM-версии. Содержит два параметра:

**String definition** – таблица, из которой необходимо получить список результатов;

**String query** – запрос для получения фильтрованного списка

Важным отличием от COM-версии является то, что здесь используется не **SQL**, а **HQL**, который хоть и похож на первый, но, имеет свои особенности. Рекомендуется почитать про этот язык отдельно в случае, если ваш запрос не сработает.

Как и в COM-версии Сапериона имеется три варианта выполнения функции в зависимости от содержания параметра query:

Query = пусто: будет выполнен поиск без фильтрации

Query = условия: условия будут вставлены в запрос после where.

Query = @полный HQL запрос [начинается именно с символа @]: параметр definition будет проигнорирован и выполнится тот запрос, который стоит после символа @.

Функция возвращает объект типа Cursor, который описан ниже.

#### Функция SelectQuery (SaQueryInfo query) , возвращает Cursor

Выполняет запрос из текущего объекта Connector по уже составленному объекту запроса из ClassicConnector. Используется в крайне редких случаях.

#### Функция LoadDocumentStream (String xhdoc, boolean currentRevision, int iElement), возвращает InputStream

Данная функция служит для получения потока с контентом документа. Содержит три параметра:

**String xhdoc** – HexUID документа;

**Boolean currentRevision** – загрузка именно той ревизии, к которой относится HexUID;

**Int iElement** – номер элемента (файла) документа.

Примечание: Для получения контента в виде массива байт необходимо в первую очередь инициализировать массив байт по размеру контента и выполнить функцию потока read, в которую необходимо передать инициализированный массив.

#### Функция LoadDocInfo (String xhdoc), возвращает объект SaDocInfo

Данная функция возвращает объект с информацией о документе в формате Saperion Classic Connector. Содержит один параметр:

**String xhdoc** – HexUID документа.

## Cursor

Пространство имен: ru.saperion.com

Предназначение: Работа со списком документов, полученных в результате запроса.

### Методы

#### Конструктор Cursor(Application app, List<SaDocumentsInfo> docs)

Как правило, корректное использование библиотеки не подразумевает самостоятельное создание экземпляров данного класса. Экземпляр данного класса должен возвращаться функциями Application.SelectHQL и Application.SelectQuery.

Конструктор содержит два параметра:

**Application app** – экземпляр класса Application для реализации обратной связи и возможности выполнять загрузку документов из медии в дальнейшем;

**List<SaDocumentInfo> docs** – инициализирующий список, содержащий базовую информацию о документах, полученную из базы данных.

#### Конструктор Cursor(List<SaDocumentsInfo> docs)

Аналогично вышеописанному конструктору. Не использует экземпляр класса Application. Предполагается использовать в тех случаях, когда не нужно загружать документы из медии.

#### Функция First(), возвращает Boolean

Устанавливает в качестве текущего документа самый первый, полученный из базы данных.

Возвращает TRUE при успешной установке;

Возвращает FALSE при неудачной установке (нет документов).

#### Функция Last(), возвращает Boolean

Устанавливает в качестве текущего документа самый последний, полученный из базы данных.

Возвращает TRUE при успешной установке;

Возвращает FALSE при неудачной установке (нет документов).

#### Функция Next(), возвращает Boolean

Устанавливает в качестве текущего документа следующий после текущего.

Возвращает TRUE при успешной установке;

Возвращает FALSE при неудачной установке (нет документов, нету следующего документа).

#### Функция Count(), возвращает int

Возрващает количество документов в курсоре.

#### Функция Document(), возвращает Document

Возвращает текущий объект типа Document из курсора.

#### Функция getApplication(), возвращает Application

Возвращает объект типа Application (если он был задан), из которого был получен данный курсор.

## Document

Пространство имен: ru.saperion.com

Предназначение: Работа с документом.

### Используемые типы-перечисления

#### SaveType

Данный тип используется при сохранении файлов из документа. Включает в себя два варианта:

|  |  |  |  |
| --- | --- | --- | --- |
| Программное название | Описание | Текстовое значение | Числовое значение |
| WEB | При использовании данного параметра будет использовано сохранение через функции браузера. | Web | 1 |
| CLASSIC | При использовании данного параметра будет осуществлено сохранение напрямую в файловую систему. | Classic | 0 |

### Методы

#### Конструктор Document()

Данный конструктор создает пустой объект документа.

#### Конструктор Document(Application app, SaDocumentInfo docInfo)

Как правило, корректное использование библиотеки не подразумевает самостоятельное создание экземпляров данного класса. Экземпляр данного класса должен возвращаться функцией Cursor.Document()

Данный конструктор создает объект документа по аналогии с COM-объектом на основании информации из структуры SaDocumentInfo, полученной из Saperion Classic Connector.

Конструктор содержит два параметра:

**Application app** – экземпляр класса Application для реализации обратной связи и возможности выполнять загрузку документа из медии (метод Load);

**SaDocumentInfo docInfo** – структура, полученная из Saperion Classic Connector.

#### Конструктор Document (SaDocumentInfo docInfo)

Аналогично вышеописанному конструктору. Не использует экземпляр класса Application. Предполагается использовать в тех случаях, когда не нужно загружать документ из медии.

#### Функция Load(), возвращает Boolean

Данная функция загружает документ из медии. После загрузки становится доступна работа с файлами документа: известны названия файлов, размер, содержимое. Для удобства работы с файлами был пересмотрен классический концепт COM-объектов сапериона и был введен новый класс File.

Функция возвращает TRUE в случае успешной загрузки документа из медии;

Функция возвращает FALSE, если в Саперионе отсутствует информация о документе по XHDOC и если количество файлов в документе = 0. В случае ошибки будет выдана ошибка типа Exception с описанием.

#### Функция Load(String XHDOC), возвращает Boolean

Аналогична функции Load(), но выполняет загрузку документа по HexUID, указанного в качестве параметра.

#### Функция HexUID(), возвращает String

Данная функция возвращает HexUID (XHDOC) документа.

#### Функция getProperty(String name), возвращает String

Данная функция возвращает свойство документа (значение поля), переданное в параметре. В случае отсутствия такого свойства будет возвращена пустая строка.

#### Функция getFiles(), возвращает массив объектов File

Данная функция возвращает массив объектов типа File, которые содержат информацию о файлах в документе.

В случае, если документ не был загружен из медии – будет выдана ошибка типа Exception.

#### Функция NumElems(), возвращает int

Данная функция возвращает количество файлов в документе.

#### Функция SubDocument(int index), возвращает File

Данная функция принимает номер файла документа от 1 до количества файлов в документе (NumElems) и возвращает объект типа File.

В случае, если документ не был загружен из медии – будет выдана ошибка типа Exception

В случае, если был запрошен недействительный номер элемента – будет выдана ошибка типа Exception.

#### Функция SaveAs(int index, SaveType type, String path), возвращает Boolean

Данная функция используется для сохранения файлов документа. Используются следующие параметры:

**Int index** – номер файла в документе, указаывается от 1 до количества файлов в документе (NumElems). В случае ,если будет указан номер 0 – будут сохранены все файлы документа.

В случае, если документ не был загружен из медии – будет выдана ошибка типа Exception

В случае, если был запрошен недействительный номер элемента – будет выдана ошибка типа Exception.

**SaveType type** - Так как изначально с Saperion Classic Connector работа может вестись напрямую (Classic) и через веб клиент (Web), то, соответственно, и методов сохранения два (для работы напрямую и через браузер). При сохранении через браузер пользователю будет предложено выбрать место для сохранения файла.   
Категорически не рекомендуется использовать индекс 0 при сохранении через веб, так как при большом количестве файлов процесс сохранения может затянуться.

**String path** – адрес директории для сохранения файлов при работе через режим Classic. При использовании режима Web – игнорируется.

Функция возвращает TRUE в случае успешного сохранения.

#### Функция getApplication(), возвращает Application

Возвращает объект типа Application (если он был задан), из которого был получен данный курсор.

## File

Пространство имен: ru.saperion.com

Предназначение: Работа с файлом, полученным из документа.

### Методы

#### Конструктор File(Document doc, int elementNo)

Данный конструктор создает объект файла, который позволяет получить всю необходимую информацию о файле из архива. Принимает два параметра

**Document doc** – Объект документа

**Int elementNo** – номер файла в документе

#### Функция getFileName(), возвращает String

Возвращает имя файла

#### Функция getFileSize(), возвращает Long

Возвращает размер файла в байтах

#### Функция getElementType(), возвращает SaDocInfo.ElementType

Возвращает тип файла (в формате Saperion Classic Connector)

#### Функция getContent(), возвращает массив byte

Возвращает массив типа byte с содержимым документа

#### Функция getSaDocInfo(), возвращает SaDocInfo

Возвращает структуру с информацией о файле в формате Saperion Classic Connector.

# Вспомогательные утилиты

## DateWorker

Пространство имен: ru.saperion.tools

Предназначение: Работа с датами.

### Используемые типы-перечисления

#### DateFormat

Данный тип используется при преобразовании даты в строку и обратно:

|  |  |  |  |
| --- | --- | --- | --- |
| Программное название | Описание | Текстовое значение | Формат |
| SAPERION | Формат даты, как правило получаемый из свойств документа. | Saperion | EEE MMM dd HH:mm:ss Z yyyy |
| RUSSIAN | Формат даты, используемый для отображения в России | Russian | dd.MM.yyyy |
| ENGLISH | Формат даты, используемый для отображения в Англии\США. Так же используется в запросах. | English | yyyy-MM-dd |

### Методы

#### Статическая функция FromStringToDate(String sSource, DateFormat format), возвращает Date

Данная функция предназначена для преобразования строки в формат даты JAVA. Как правило, данная функция необходима для дальнейшем использовании даты в запросе. Используемые параметры:

**String sSource** – строка с записанной датой;

**DateFormat format**- тип формата даты, в котором она записана в строке.

#### Статическая функция FromDateToString(Date dateSource, DateFormat format), возвращает String

Данная функция предназначена для преобразования объекта даты в строку по нужному формату. Как правило, используется для отображения в графическом интерфейсе, или протоколах. Использует параметры:

**Date dateSource** – дата, которую нужно преобразовать;

**DateFormat format** – формат, в котором нужно записать дату в строке.

## FileWorker

Пространство имен: ru.saperion.tools

Предназначение: Работа с файлами.

### Методы

#### Статический метод SaveFile(String sFileName, byte[] content)

Данный метод предназначен для сохранения массива байт (содержимого файлов) в файловую систему. Принимает два параметра:

**String sFileName** – адрес для сохранения файла;

**byte[]content** – массив с содержимым файла.

Если во время сохранения возникла ошибка –будет выдана ошибка типа Exception с описанием причины ошибки.

#### Статический етод MkDir(String sDirName)

Данный метод предназначен для создания директории. Принимает один параметр – адрес директории.

Если во время создания возникла ошибка –будет выдана ошибка типа Exception с описанием причины ошибки.

## WebWorker

Пространство имен: ru.saperion.tools

Предназначение: Работа с WEB.

### Методы

#### Статический метод SaveFile(String sFileName, byte[] content)

Данный метод предназначен для сохранения массива байт (содержимого файлов) через браузер (при использовании веб-клиента). Принимает два параметра:

**String sFileName** – имя для файла;

**byte[] content** – массив с содержимым файла.

#### Статический метод SaveFile(String sFileName, String sContent, String sCodePage)

Данный метод предназначен для сохранения строки (содержимого файлов) через браузер (при использовании веб-клиента) с использованием конкретной кодировки. Принимает три параметра:

**String sFileName** – имя для файла;

**String sContent** – строка с содержанием файла;

**String sCodePage** – название кодировки, в которой необходимо сохранить файл.

# Работа с WEB-клиентомВспомогательные утилиты

## Field

Пространство имен: ru.saperion.webclient.forms

Предназначение: Работа с элементом формы - полем.

### Используемые типы-перечисления

#### Type

Данный тип используется при получении и занесении значений в поле:

|  |  |  |  |
| --- | --- | --- | --- |
| Программное название | Описание | Числовое значение | Текстовое значение |
| TEXT | Обычное поле | 1 | Text |
| COMBO | Поле с выпадающим списком | 2 | Combo |
| DATE | Поле дата (с календарем) | 3 | Date |

#### Limit

Данный тип используется при обработке поля в качестве условного параметра составляемого запроса:

|  |  |  |  |
| --- | --- | --- | --- |
| Программное название | Описание | Текстовое значение | Значение для запроса |
| BOTH | Параметр должен быть полностью равен содержимому поля | Both | = |
| LOWER | Параметр должен быть больше-равен содержимому поля (используется для дат и чисел) | Lower | >= |
| UPPER | Параметр должен быть меньше-равен содержимому поля (используется для дат и чисел) | Upper | <= |
| ANY | Параметр должен содержать содержимое поля (аналог LIKE) |  | Like |

### Методы

#### Конструктор Field(Type type, String name, String value)

Данный конструктор предназначен для создания объекта типа Field и хранения в нем оперативной информации по полю. Как правило, используется в функции ParseField и не предназначен для использования извне. Принимает параметры:

**Type type** – тип поля;

**String name** – название поля;

**String value** – значение поля.

#### Функция getType(), возвращает Type

Возвращает тип поля

#### Функция getValue(), возвращает String

Возвращает значение файла

#### Функция getName(), возвращает String

Возвращает название поля файла

#### Статическая функция ParseField(IntelligentField field), возвращает Field

Данная функция предназначена для получения актуальной информации о поле по объекту из Saperion WEB Client. Принимает в качестве параметра этот самый объект, выдает готовый заполненный объект для дальнейшего использования.

## Mask

Пространство имен: ru.saperion.webclient.forms

Предназначение: Работа с формой веб-клиента.

#### Конструктор Mask (IntelligentFormView form)

Данный конструктор инициализирует объект для дальнейшей работой с формой по привычному формату, как через COM. В качестве параметра конструктор принимает объект формы из Saperion WEB Client.

#### Функция Field(String sFieldName), возвращает String

Данная функция возвращает значение указанного в параметре поля. Если такого поля не существует – будет возвращено пустая строка.

#### Функция Field(String sFieldName, Field.Limit limit), возвращает String

Данная функция возвращает значение поля. Если такого поля не существует – будет возвращено пустая строка. Принимает два параметра:

**String sFieldName** – имя поля;

**Field.Limit limit** – ограничение\граница поля (используется для дат и чисел).

#### Метод SetField(String sFieldName , String sFieldValue)

Данная функция устанавливает значение в поле. Принимает два параметра:

**String sFieldName** – имя поля;

**String sFieldValue** – значение, которое нужно установить в поле.

#### Метод SetField (String sFieldName , String sFieldValue, Field.Limit limit)

Данная функция устанавливает значение в поле. Принимает два параметра:

**String sFieldName** – имя поля;

**String sFieldValue** – значение, которое нужно установить в поле;

**Field.Limit limit** – ограничение\граница поля (используется для дат и чисел).

#### Функция FieldToQuery(SaQueryInfo query, String sFieldName), возвращает SaQueryInfo

Данная функция предназначена для формирования корректного параметрического фильтра в запрос из Saperion Classic Connector.

**SaQueryInfo query** – запрос, который необходимо дополнить параметрическим фильтром;

**String sFieldName** – имя поля.

#### Функция FieldToQuery(SaQueryInfo query, String sFieldName, Field.Limit limit), возвращает SaQueryInfo

Данная функция предназначена для формирования корректного параметрического фильтра в запрос из Saperion Classic Connector.

**SaQueryInfo query** – запрос, который необходимо дополнить параметрическим фильтром;

**String sFieldName** – имя поля;

**Field.Limit limit** – ограничение\граница поля (используется для дат и чисел).

#### Статическая функция AddParamFromForm(SaQueryInfo query, IntelligentFormView formView, String fieldName, Field.Limit fieldLimit), возвращает SaQueryInfo

Данная функция аналогична функции FieldToQuery, отличается тем, что не требует создания объекта формы, но нуждается в передаче полного набора параметров:

**SaQueryInfo query** – запрос, который необходимо дополнить параметрическим фильтром;

**IntelligentFormView formView** – объект формы из Saperion WEB Client;

**String sFieldName** – имя поля;

**Field.Limit limit** – ограничение\граница поля (используется для дат и чисел).

#### Статическая функция FieldFromForm (IntelligentFormView formView, String fieldName, Field.Limit fieldLimit), возвращает String

Данная функция аналогична функции Field, отличается тем, что не требует создания объекта формы, но нуждается в передаче полного набора параметров:

**IntelligentFormView formView** – объект формы из Saperion WEB Client;

**String sFieldName** – имя поля;

**Field.Limit limit** – ограничение\граница поля (используется для дат и чисел).

#### Статическая функция SetFieldFromForm (IntelligentFormView formView, String fieldName, String value, Field.Limit fieldLimit), возвращает String

Данная функция аналогична функции SetField, отличается тем, что не требует создания объекта формы, но нуждается в передаче полного набора параметров:

**IntelligentFormView formView** – объект формы из Saperion WEB Client;

**String sFieldName** – имя поля;

**String value** – значение, которое нужно установить в поле;

**Field.Limit limit** – ограничение\граница поля (используется для дат и чисел).

# Отчеты

## IReportLine

Пространство имен: ru.saperion.reports

Предназначение: Интерфейс для описания состава строки отчета и ее обработки. Для работы с генератором отчетов (описан ниже) необходимо объявить класс, соответствующий данному интерфейсу.

#### Метод loadLine(Document oDoc)

Данный метод принимает объект типа ru.saperion.com.Document. При выполнении данного метода необходимо получить или высчитать все требуемые поля для строки отчета из объекта документа.

#### Метод combine(IReportLine line)

Данный метод используется в цикле в том случае, если найдена строка БД с ключем, который соответствует уже ранее обработанному ключу. Как правило, выполняется для сложения определенных значений (суммирования).

#### Функция getKey(), возвращает String

Данная функция должна возвращать содержимое поля-ключа (либо сумму нескольких полей). Именно по этой функции вычисляется повторяемость строки.

#### Функция getFirstLine(), возвращает String

Данная функция используется при генерации CSV-файла, должна возвращать полную строку шапки.

#### Функция getSaperionQuery(Mask oMask), возвращает SaQueryInfo

Данная функция должна создавать запрос для формирования отчета. В случае, если объект маски не передается – запрос должен выполнятся с заранее предопределенными условиями (либо без них). Если объект маски передан – критерии запроса должны быть собраны с полей формы через метод FieldToQuery маски.

#### Функция toString(), возвращает String

Данная функция используется при генерации CSV-файла, должна возвращать значение полей строки в качестве единой строчки с разделителями.

#### Пример реализации класса по интерфейсу

public class AreaInfo implements IReportLine {

public int iKiosk, iOperators, iHelpers, iJudges;

public String sAreaName;

@Override

public void combine(IReportLine other){

this.iKiosk += ((AreaInfo)other).iKiosk;

this.iOperators += ((AreaInfo)other).iOperators;

this.iHelpers += ((AreaInfo)other).iHelpers;

this.iJudges += ((AreaInfo)other).iJudges;

}

@Override

public String toString(){

return String.format("%s;%d;%d;%d;%d;", this.sAreaName, this.iKiosk, this.iOperators, this.iHelpers, this.iJudges);

}

@Override

public void loadLine(ru.saperion.com.Document oDoc){

this.sAreaName = oDoc.GetProperty("AREANAME");

this.iKiosk = (int)(Double.parseDouble(oDoc.GetProperty("KIOSK")));

this.iOperators = (int)(Double.parseDouble(oDoc.GetProperty("OPERATORS")));

this.iHelpers = (int)(Double.parseDouble(oDoc.GetProperty("HELPERS")));

this.iJudges = (int)(Double.parseDouble(oDoc.GetProperty("JUDGES")));

}

@Override

public String getKey(){

return sAreaName;

}

@Override

public String getFirstLine(){

String sOut = "Наименование судебного участка и отдела делопроизводства;Выдача дел в электронном виде для граждан;Выдача копий решений для граждан;Поиск прецедентов аппаратом судебных участков;Поиск прецедентов мировым судьей;Примечание";

return sOut;

}

@Override

public SaQueryInfo getSaperionQuery(ru.saperion.webclient.forms.Mask oMask) throws Exception{

try

{

SaQueryInfo query = new SaQueryInfo("");

String sHQL = "select audit.AREANAME, audit.KIOSK, audit.OPERATORS, audit.HELPERS, audit.JUDGES from case\_docs\_audit\_view audit where 1 = 1 ";

query.setQueryString(sHQL);

LOG.trace("Создана строка запроса без условий");

if (oMask != null)

{

query = oMask.FieldToQuery(query, "AREANUM", Field.Limit.ANY);

query = oMask.FieldToQuery(query, "PRTDATE", Field.Limit.LOWER);

query = oMask.FieldToQuery(query, "PRTDATE", Field.Limit.UPPER);

}

sHQL = query.getQueryString();

return query;

}

catch (Exception e)

{

throw new Exception(String.format("Во время формирования запроса к БД Саперион возникла ошибка:%n%s", e.getMessage()));

}

}

}

## ReportGenerator

Пространство имен: ru.saperion.reports

Предназначение: Класс для формирования объекта со строками отчета.

#### Конструктор ReportGenerator(Class<? extends IReportLine> reportType)

Данный конструктор инициализирует генератор отчетов по классу, реализованному согласно интерфейсу IReportLine.

#### Конструктор ReportGenerator (Class<? extends IReportLine> reportType, ru.saperion.com.Cursor oCur)

Данный конструктор инициализирует генератор отчетов, принимает два параметра:

**Class<? extends IReportLine> reportType** – класс, реализованный согласно интерфейсу IReportLine;

**ru.saperion.com.Cursor oСur** – курсор, в котором содержатся выбранные из БД данные.

#### Метод initDocs(ru.saperion.com.Cursor oCur)

Данный метод предназначен для поздней инициализации данных для генерации отчета (если по какой-то причине необходимо создать экземпляр класса без данных).

#### Функция getList(), возвращает ArrayList<IReportLine>

Данная функция возвращает созданный список-массив со сгенерированными строками отчета.

#### Функция generateList(ru.saperion.com.Cursor oCur), возвращает ArrayList<IReportLine>

Данная функция выполняет генерацию отчета по данным из переданного курсора.

#### Функция generateList(), возвращает ArrayList<IReportLine>

Данная функция выполняет генерацию отчета по данным из курсора, который был задан при инициализации, либо в функции initDocs.

## Csv

Пространство имен: ru.saperion.reports

Предназначение: Класс для формирования отчета в CSV.

#### Статическая функция generate(ru.saperion.com.Application oApp, Class<? extends IReportLine> reportType, IntelligentFormView form), возвращает String

Данная функция предназначена для генерации отчета в формате CSV. Принимает параметры:

**ru.saperion.com.Application oApp** – Объект-приложение Saperion;

**Class<? extends IReportLine> reportType** – Класс, реализованный согласно интерфейсу IReportLine;

**IntelligentFormView form** – Форма из Saperion WEB Client для генерации фильтра для запроса (реализуется через класс по интерфейсу IReportLine).

Возвращает готовое содержимое для файла в формате CSV в виде строки.

#### Статическая функция generate(SaClassicConnector connector, Class<? extends IReportLine> reportType, IntelligentFormView form), возвращает String

Данная функция предназначена для генерации отчета в формате CSV. Принимает параметры:

**SaClassicConnector connector** – Объект-коннектор к Saperion;

**Class<? extends IReportLine> reportType** – Класс, реализованный согласно интерфейсу IReportLine;

**IntelligentFormView form** – Форма из Saperion WEB Client для генерации фильтра для запроса (реализуется через класс по интерфейсу IReportLine).

Возвращает готовое содержимое для файла в формате CSV в виде строки.

## Пример кода генерации отчета

public class Helper {

private static final Logger LOG = Logger.getLogger(ru.elar.judges.Helper.class);

public static void GenerateCSV(IntelligentFormView parentForm)

{

LOG.info("GenerateCSV запущен. Версия 08.10.2015 17:22");

SaClassicConnector connector = null;

try

{

ru.saperion.com.Application oApp = new ru.saperion.com.Application(ru.saperion.com.Application.ConnectorType.WEB, "");

String sReport = ru.saperion.reports.Csv.generate(oApp, AreaInfo.class, parentForm);

ru.saperion.tools.WebWorker.SaveFile(sReport, "report.csv", "Cp1251");

connector = null;

}

catch ( Exception e)

{

LOG.error(String.format("Возникла ошибка при выполнении GenerateCSV:%n%s", e.toString()));

try {

Messagebox.show(e.getMessage(), "Ошибка", Messagebox.OK, Messagebox.ERROR);

} catch (InterruptedException ex) {

}

}

}

}