**QASreport ver.1.01**

**Библиотека C++ QT от вер.5.**

1. **Описание**
2. **Возможности**
3. **Инсталляция и использование**
4. **Описание класса: QASReport**
5. **Дизайнер отчетов**
   1. **Интерфейс**
   2. **Главное меню, команды клавиатуры, работа с мышью**
   3. **Главное меню**
   4. **Визуальные объекты дизайнера**
   5. **Бэнды**
6. **Общий порядок работы**

**1. Описание**

QASreport - это библиотека, содержащая набор классов для построения отчетов. Представляет собой сочетание дизайнера, генератора и средство вывода отчетов.

Автор не гарантирует правильную работу компонента и не несет ответственности за возможный ущерб в результате использования данного программного обеспечения. Все замечания и пожелания отправляйте по нижеприведенному адресу.

**2. Возможности**

* Генератор отчетов, основанный на бэндах;
* Встроенный дизайнер, доступный и в run-time, с возможностью работы как с обычным графическим редактором;
* Вывод результата отчета в Preview;
* Любое количество страниц отчета;
* Составные отчеты, возможность составления нескольких отчетов и форм в одном файле;
* Сохранение файла отчета в формате - XML;
* Master-detail отчеты;
* Полный контроль над процессом печати, поддержка любых типов бумаги и устройств вывода на печать;
* Простота конструирования отчета с помощью простых компонентов: текстовый прямоугольник, линия, изображение, фигура, и штрих-код;
* Вывод данных любой СУБД, драйвер которой установлен;
* Вывод данных из любой SQL модели, наследника QSqlQueryModel;
* Вывод данных из любой стандартной модели, наследнике QAbstractItemModel;
* Экспорт отчетов в PDF файлы;
* Встроенный математический инетрпретатор;
* Поддержка Qt Script (в дальнейшем);
* др.

**3. Инсталляция и использование**

Для компиляции библиотеки и демо программы, запустите на компиляцию файл группового проекта: asreport.pro.

После компиляции папка /build/debug-release будет содержать динамическую библиотеку: **asreportlib**, файл исполняемого файла демопрограммы: **asreportdemo,** файлы форм и остальные файлы.

Для использования, соответственно, к проекту линкуется **asreportlib.** В папку с проектом переписывается файладинамической библиотеки: asreportlib.dll (win).

К проекту также необходимо подключить модули: sql xml printsupport widgets.

Базовый класс для работы с библиотекой - **QASReport**.

Подключаем файл класса QASReport:

#include ".. /asreportlib/asreport.h "

//создаем экземпляр класса QASReport.

m\_Report = new QASReport(this);

//Добавляем к отчету источники данных - модели для вывода

m\_Report ->addModel("sqldatamodel ",m\_sqldatamodel);

m\_Report ->addModel("stringsmodel ", m\_stringsmodel);

m\_Report ->addModel("itemsmodel ", m\_itemsmodel);

//Загружаем файл отчета

QString filereport = QDir::currentPath() + "/forms/list.xml";

if(!QFile::exists(filereport)) return;

QFile file(filereport);

if (!file.open(QIODevice::ReadOnly)) return;

bool res = m\_Report->load(&file);

//Выводим отчет на предосмотр

m\_Report->reportView();

// печать

Report->reportPrint();

//или редактирование

m\_Report->edit(filereport);

//При необходимо, инициализируется обработка сигналов:

// начала печати

connect(m\_Report,SIGNAL(signalBeginReport()),this,SLOT(slotBeginPrint()));

// вывод пользовательских переменных отчетов

connect(m\_Report,SIGNAL(signalGetValue(QString,QVariant&)),this,

SLOT(slotGetValue(QString,QVariant&)));

// обработка вывода конца вывода печати

connect(m\_Report,SIGNAL(signalEndReport ()),this,SLOT(slotEndReport()));

// обработка вывода при переходе на следующую запись вывода табличных данных

// передается указатель на модель вывода

connect(m\_Report,SIGNAL(signalModelOnNextRec(QObject \*),this,SLOT(slotModelOnNextRec (QObject \*) ));

// обработка вывода при установке на первую запись таблицы

//передается указатель на модель вывода

connect(m\_Report,SIGNAL(signalModelOnFirstRec (QObject \*),this,SLOT(slotModelOnFirstRec (QObject \*) ));

// обработка формирования новой страницы

//передается номер текущей страницы

connect(m\_Report,SIGNAL(signalNewPage (int),this,SLOT(slotNewPage (int) ));

**4. Описание класса: QASReport.**

QASReport - является центральным, основным классом для работы с отчетами. C помощью его производится загрузка, вызов дизайнера отчетов, выполняется построение отчета, его печать и предварительный просмотр.

Публичные методы:

void **reportView**(); Вывод отчета на предварительный просмотр;

void **reportPrint**(bool show = true) ; Вывод на печать, переменная show – показывать диалог печати перед печатью;

QPrinter \* **printer**() ; возвращает текущий объект принтера (QPrinter) отчета;

QList<QString> **models**(); Возвращает список наименований таблиц, используемых QASReport для печати;

QObject \* **getModel**(QString name,bool addonabsent = false); Возвращает указатель на объект модели наследника от QAbstractListModel по наименованию: name. При отсутствии модели с таким наименованием и addonabsent = true создается и добавляется объект QSqlTableModel с наименованием: name.

void **addModel**(QString name,QObject\* object) ; Добавление модели в случае отсутствия модели с таким наименованием, name – наименование модели, object – любой объект наследник от QAbstractListModel.

void **clearModels**(); Очистка списка введенных моделей;

int **pageCount**() Возвращает количество страниц активного отчета;

bool **load**(QString xmltxt); Загрузка отчета из строки, содержащей XML текст отчета. Функция возвращает истину при удачной загрузке;

bool **load**(QIODevice \* dev); Загрузка отчета из файла на диске. Функция возвращает истину при удачной загрузке.

bool **load**(QByteArray & buffer); Загрузка отчета из байтового массива QByteArray. Функция возвращает истину при удачной загрузке.

void **runDesigner**(); Запуск дизайнера отчетов без загруженного отчета.

void **edit**(QString filename); Запуск дизайнера отчетов на редактирование отчета filename. Передается путь и наименование файла отчета.

void **edit**(QIODevice \* dev); Запуск дизайнера отчетов на редактирование отчета, загруженного устройством dev.

void **edit**(QByteArray & buffer); Запуск дизайнера отчетов на редактирование отчета, содержащемся в байтовом массиве buffer.

Signals:

void **signalGetValue**(QString paramName, QVariant &paramValue); Вызывается для передачи отчету значения пользовательской переменной. Пользовательская переменная это произвольная строка заключенная в скобки: []. Например: [наименование контрагента].

void **signalBeginReport**(); Вызывается перед началом процесса печати/просмотра отчета.

void **signalEndReport**(); Вызывается в конце процесса вывода отчета на печать/просмотр.

void **signalSaveToStream**(QDomDocument \*); Сигнал вызывается дизайнером отчетом в случае если файл отчета, передан на редактирование в виде байтового массива либо потока от устройства QIODevice. Для пользовательской обработки записи изменений отчета. Например, для записи файла отчета не в виде файла на диск, а в Blob поле таблицы отчетов. Передается указатель на объект QDomDocument, содержащий загруженный файл отчета.

void **signalModelOnNextRec**(QObject \*); Вызывается при переходе на следующую запись вывода табличных данных. Используется, например, для обновления вывода подчиненных данных, через бэнд: «Detail Data» в отчетах с подчиненными master-detail данными. Передается ссылка на модель, вывод которой обрабатывается. Вызывается при выводе через бэнды вывода потоковых табличных данных: Master Data, Detail Data.

Пример обработки:

void FormDetailList1::slotModelOnNextRec(QObject \*object)

{

if(object == m\_customerliter){ //если модель является родительской //(customerliter) производится переход на следующую запись в компоненте //QTableView ее вывода и соответствующая фильтрация данных подчиненной //таблицы

QModelIndex index = m\_customerliter->*index*(m\_customerliter->query().at(),0);

if(index.isValid())

ui->tablemaster->setCurrentIndex(index);

}

}

void **signalModelOnFirstRec**(QObject \*); Вызывается при установке вывода таблицы на первую запись. Передается ссылка на модель, вывод которой обрабатывается. Вызывается при выводе через бэнды вывода потоковых табличных данных: btMasterData, btDetailData.

void **signalNewPage**(int); Сигнал вызывается при формировании новой страницы. Передается номер текущей обрабатываемой страницы.

**5. Дизайнер отчетов**

Дизайнер отчетов открывается методом QASReport::**runDesigner**() - открытие пустого дизайнера. Либо командами:

void **edit**(QString filename) с передачей пути и наименования файла отчета;

void **edit**(QIODevice \* dev); с передачей указателя на устройство ввода/вывода с загруженным файлом отчетом;

void **edit**(QByteArray & buffer); с передачей указателя на байтовый массив, содержащий загруженный файл отчета;

* 1. **Интерфейс**

Дизайнер представляет удобные средства для открытия, редактирования, сохранения файла отчета, вывода его на просмотр либо печать. Редактирование формы отчета производится как в обычном графическом редакторе, используя те же приемы работы с мышью и клавиатурой.

Со следующими особенностями:

* свойства выделенных объектов можно одновременно менять как на панелях инструментов так и в инспекторе свойств;
* некоторые объекты имеют редакторы, которые вызываются двойным щелчком мыши на нем, либо с помощью кнопки – Enter на выделенном объекте, либо нажатием кнопки в инспекторе свойств на соответствующем свойстве, либо через правую кнопку мыши поп-меню - редактор;
* некоторые свойства: для объекта «Текст»: фонт, формат и маска, перенос и растяжения можно изменить с помощью выпадающего поп-меню через правую кнопку мыши;
* все объекты дизайнера можно перемещать и позиционировать как угодно и где угодно. Для некоторых объектов местоположение некритично, например для бэндов типа: Title, Page Hader, Page Footer, Column Title они будут выводить соответственно всегда в заголовке, в начале, конце страницы, независимо от их местоположения на странице дизайна. Для большинства объектов их местоположение критично, и на печать они будут выводиться именно там, где вы их видите на странице дизайна. Таким образом внешний вид печатной страницы определяется только вашей фантазией и воображением.
  1. **Главное меню, команды клавиатуры, работа с мышью**

Операции с файлами, главное меню – Файл:

*  Ctr + N – создание нового отчета;
*  Ctr + O – открытие файла отчета;
*  Ctr + S – запись текущего отчета;
* Сохранить как – запись отчета с переименованием;
* Список переменных - открытие редактора пользовательских переменных отчета. Пользовательские переменные - это строки, создаваемые пользователем для вывода через них данных. Обработка пользовательских переменных производится созданием слота сигнала void QASreport::**signalGetValue**(QString paramName, QVariant &paramValue);
* Параметры страниц – установка формата бумаги, ориентации вывода, отступов;
*  Ctr + P – вызов предварительного просмотра редактируемого отчета;
* Выход – закрытие и окончание редактирования отчета. В случае изменений в отчете, программа спросит о необходимости записи;

Команды редактирования дизайнера – Стандартные команды - главное меню - Редактирование:

*  Ctr + Z – восстановить любую предыдущую операцию редактирования (перемещение, изменение, свойств, удаление, т.д.);
*  Ctr – Y - вернуть любую отмененную операцию редактирования;
* Ctr + X – вырезать выделенные объекты;
* Ctr + C - копировать выделенные объекты в буфер;
* Ctr + V – вставка скопированных объектов из буфера;
* Del – удаление выделенных объектов;
*  Ctr + A - выделение всех объектов страницы дизайна;
*  - добавить страницу отчета;
*  - удалить текущую страницу отчета;
*  - переместить выделенные объекты выше в стеке отображения;
*  - переместить выделенные объекты ниже в стеке отображения;

Команды редактирования дизайнера, не имеющие ссылок на главное меню:

* Shift – клик по объекту – добавление объекта к списку выделения, повторный клик снимает выделение;
* Ctr + стрелки клавиши – попиксельно перемещение выделенных объектов;
* Ctr + Shift + стрелки клавиши – быстрое перемещение (на 5 пикселей) выделенных объектов;
* Shift + стрелки клавиши – попиксельно изменение размера выделенных объектов;
* Клик мышкой по фоновому экрану страницы дизайна - сброс выделения;

Сервисные команды – главное меню – Сервис:

* Панели – включение – выключение панелей: стандартная, текст, рамки, объектов, выравнивание;
  + Стандартная панель - содержит команды продублированные из меню – Файл и редактирования;
  + «Текст» панель – содержит команды редактирование текста: фонт, размер, толщина, цвет, вертикальное, горизонтальное выравнивание, ориентация текста;
  + «Рамка» панель - содержит команды настройки рамок вокруг объектов: наличие, толщина, цвет рамки, цвет фона заливки, стиль;
  + «Объекты» панель – содержит объекты дизайна: Текст, Бэнд, Картинка, Линия, Фигура, Бар-код;
  + «Выравнивание» панель - команды выравнивания объектов на странице дизайна. С помощью них можно аккуратно:
  + выровнять выделенные объекты по нижней, верхней, левой, правой, границе. Либо горизонтально/вертикально по центру страницы. Распределить равномерно объекты горизонтально, вертикально. Для команд выравнивания, источником границы выравнивания служит первый выделенный объект.
* Настройки дизайнера:
  + Сетка – отображение, шаг сетки страницы дизайна, выравнивание по ней;
  + Линейка страницы дизайна – отображение в миллиметрах либо в дюймах;
  + Редактирование после вставки – запуск редактора после создания нового объекта;
  + Отображение заголовка в бэндах отчета при дизайне;
* Помощь:
  + F1 – открытие файла помощи;
  + Информация о программе;

* 1. **Визуальные объекты дизайнера**

Все визуальные объекты отчета представлены шестью типами: Текст, бэнд, изображение, линия, фигура, бар-код. Общими свойствами всех объектов дизайнера являются:

Name - наименование;

Width - ширина;

Height - высота;

Left - отступ слева;

Top - отступ сверху;

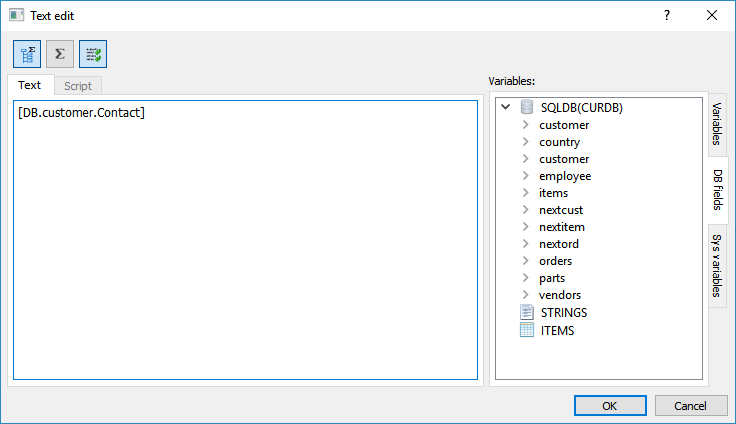
Border – тип обрамляющей рамки;

BorderStyle – стиль линии обрамляющей рамки;

Если компонент имеет редактор, нужно выделить, сделать двойной щелчок мыши на нем, либо нажать клавишу «Enter», либо вызвать редактор через инспектор свойств объекта, либо через выпадающее поп меню.

Объект **«Текст» -** является основным компонентом дизайна, базовым строительным кирпичиком отчета. Может выводить обычные строки, либо содержимое полей таблиц базы данных, либо математические формулы, выражения служебных функций и констант. Помимо вышеуказанных имеет следующие свойства:

* вертикальное, горизонтальное выравнивание,
* фонт;
* перенос слов;
* направление вывода горизонтальное, вертикально;
* растяжение (Strechable) – при установке данного свойства, бенд, на котором расположен объект, при печати будет менять высоту, в соответствии с содержимым объекта «Текст»;
* формат строки, применяющийся при выводе через объект чисел, даты/время, логических значений. Редактор формата можно вызвать либо через инспектор свойств, либо через выпадающее поп-меню;



Редактор объекта «Текст» вызывается двойным щелчком мыши на нем, либо нажатием кнопки – Enter, вызовом через поп-меню, либо инспектор свойств. Редактор объекта «Текст» представляет собой обычный текстовый редактор, панель инструментов, и броузер переменных, для автоматизированного ввода переменных. Броузер содержит три вкладки: пользовательские переменные, поля источников данных, системные переменные. Пользовательские переменные (Variables) - произвольные строки разделенные по группам, через которые производится вывод с помощью обрабоки сигнала **QASReport::signalGetValue(QString paramName, QVariant &paramValue) .** В paramName, соответственно, передаются пользовательские переменные, через paramValue возвращаются присвоенные им значения. Ввод пользовательских переменных производится через главное меню – файл – список переменных.

DB FIelds – переменные полей источников данных любого типа. Дерево групп QSQLDB(ConnecttionName) содержит список таблиц подключенных SQL любых баз данных, и полей таблиц. Группа STRINGS – модели наборы строк. Группа ITEMS - все другие типы моделей, наследников от [QAbstractItemModel](qabstractitemmodel.html). Поля источников данных SQL таблиц содержат наименования полей, поля всех остальных типов данных – цифровые индексы полей.

Sys variables – вкладка со встроенными системными переменные:

page# – номер страницы;

date# – текущая дата;

time# – текущее время;

linethrough# – номер обрабатываемой записи потоковых данных глобально, с учетом всех потоковых дата-бэндов;

current# – номер обрабатываемой записи потоковых данных текущего дата-бэнда;

totalpages# – количество страниц отчета;

Варианты ввода текста объекта «Текст».

* **обычная строка –** простая строка, на печать выводится в том виде, как вы ее ввели;
* **[строка переменная], [реквизиты - наименование] –** строка в квадратных скобках воспринимается как переменная. При выводе определяется, является ли она пользовательской, служебной переменной, либо полем источника данных;
* **[DB.customer.Contact] –** переменная – поле таблицы источника данных. Идентификатор источника данных: предлог DB, далее после точки, наименование таблицы: **customer,** и следующая подстрока после точки: **Contact** - наименование поля.

Для источников данных, не являющихся таблицами SQL баз данных (QStringListModel, QStandardItemModel, любые наследники [QAbstractItemModel](qabstractitemmodel.html), кроме QSqlQueryModel), поле обозначается цифровым индексом: **[DB.itemsmodel.0], [DB.itemsmodel.1]**, т.д.

* **[[сумма документа] + [налог]], [sum([сумма документа]) / 100 \* 20]**  - выражения, любые математические операции с переменными должны браться в квадратные скобки. Иначе при печати это будет восприниматься как обычная строка;
* **Наименование предприятия: [Company name] адрес компании: [Company address] -**  смешанный ввод обычного текста, переменных, выражений в одном объекте «Текст»;

**Пользовательские переменные** вводятся через главное меню - Список переменных.

|  |  |
| --- | --- |
|  | Форма отображает список переменных отчета. Команды панели формы позволяют добавлять, изменять, удалять переменные и их группы.  Для переменной необходимо наличие группы. Т.е. если список переменных пуст, сначала нажмите кнопку добавления  - группы.  Наименование переменной может содержать любой набор символов и пробелов, кроме квадратных скобок: [ ]. Количество переменных и групп не ограничено.  После добавления они будут доступны в редакторах объектов «Текст», «Картинка», «Штрих-код». |

Пример обработки пользовательских переменных отчета в тексте программы:

//--------------------------

void FormList::initPrint()

{

QObject::connect(MainForm->report(),SIGNAL(signalGetValue(QString,QVariant&)),this,

SLOT(slotGetValue(QString,QVariant&))); //init read m\_Report custom variables

}

void FormList::slotGetValue(QString paramName, QVariant &paramValue)

{

if(!paramName.compare("Company name"))

paramValue = m\_customer->query().value("Company").toString();

if(!paramName.compare("Company address"))

paramValue = m\_customer->query().value("Addr1").toString();

}

//--------------------------

**5.4.Бэнды**

**QASreport** является бэнд ориентированной библиотекой. Именно благодаря ним данные выводятся и выглядят так, как вам нужно. Все объекты выводятся в границах бэнда, на котором они расположены. Некоторые бэнды не зависят от того, где вы их расположите. Например, Report Title, Report Summar, Page Heade, Page Footer, Column Header.

Для всехостальныхтипов бэндов расположение их на странице дизайна имеет решающее значение.

Краткая характеристика бэндов:

* **Report Title** - выводит в начале отчета. С помощью это бэнда выводятся заголовочные данные отчета. В отчете может быть только один **Report Title;**
* **Report Summary –** выводв самом конце отчета. Используется для вывода суммирующих расчетов;
* **Page Header –** выводится в начале каждой страницы;
* **Page Footer –** выводится в конце каждой страницы;
* **Master Header –** выводится в начале печати потоковых данных из master data бэнд. Поэтому соответственно должен быть расположен на странице выше Master Data бэнда, для данных которого он является заголовком. Это требуется потому, что количество Master Data бэндов в отчете может быть любым. Удобно использовать для вывода заголовка таблицы.
* **Master Data –** базовый бэнд для вывода данных из любых источников данных. Имеет редактор, вызывается двойным щелчком мыши на нем, либо с помощью клавиши Enter. В открывшемся редакторе, из выпадающего списка наименований источников выберете необходимую модель.
* **Master Footer –** выводится в конце вывода потоковых данных Master Data. Удобно использовать для вывода суммарных расчетов для полей выводимого Master Data;
* **Detail Header -** выводится в начале печати потоковых данных из подчиненного Detail Data бэнда. Поэтому соответственно должен быть расположен на странице выше Detail Data бэнда, для данных которого он является заголовком. Количество Master Data бэндов в отчете может быть любым. Удобно использовать для вывода заголовка таблицы.
* **Detail Data -**  бэнд для вывода подчиненных потоковых данных. Вывод которых производится после вывода основных данных через Master Data бэнд. Имеет редактор аналогичный Master Data бэнду. Обработка Detail Data производится при каждом переходе на новую запись основной модели в родительском Master Data. Родительским для Detail Data является ближайший расположенный выше Master Data. Поэтому здесь важна обработка **signalModelOnNextRec**(QObject \* model). Сигнал передает ссылку на модель. После получения сигнала от родительской модели предпринимаются соответствующие действия для фильтрации дочерних данных, перехода на следующую запись, т.д. Количество Detail Data может быть любым.
* **Detail Footer -** выводится в конце вывода потоковых данных Detail Data. Удобно использовать для вывода суммарных расчетов для полей выводимого Detail Data;
* **Column Header -** выводится в начале вывода табличных данных через первый Master Data бэнд на странице. При переносе данных на следующую страницу, соответственно выводится в начале страницы. Может быть только один на страницу;
* **Column Footer –** выводится в конце вывода первого Master Data бэнд на странице дизайнера;
* **Group Header** –бэнд для вывода сгруппированных данных. В редакторе данного бэнда указывается источник-модель и поле, по которому производится группировка данных, выводимых через Data бэнд (Master либо Detail). Для корректного отображения данных, выводимые данные, соответственно должны быть сгруппированы. На дизайн странице Group Header бэнд должен располагаться выше дата бэнда, сгруппированные данные которого он выводит. Количество Group Header бэндов может быть любым, соответствующему количеству группирующих полей родительской таблицы.
* **Group Footer -** выводится в конце вывода сгруппированных данных. На дизайн странице располагается после Data бэнда, в том же порядке как его Group Header, для которого он служит финальным;

|  |  |
| --- | --- |
| Последовательность обработки бэндов при печати. | В таком же порядке бэнды должны располагаться на дизайн странице для корректного вывода печати. |

Агрегатные функции: SUM,COUNT,AVG,MAX,MIN с переменными можно применять только на суммирующих бэндах: Report Summary, Master Footer, Detail Footer, Group Footer. Соответственно, подсчет будет производиться уровню данных, охватываемым дата бэндами, для которых данные Footer служат финальными. В бэнде Report Summary – агрегаты производят расчет по всем источникам данных отчета.

1. **Общий порядок работы**

Создаем новый отчет.

Продолжение следует…

Автор: Жуков Сергей Анатольевич

Контактные данные:

email: serg\_@ukr.net

WWW: