

# Руководство разработчика | Отчёты

Данное руководство содержит теоретические сведения о расширении возможностей [Платформы НЕЙРОСС](#) в части построения отчётов, в частности добавления в Платформу новых шаблонов отчётов.

Пошаговая инструкция по созданию простого набора шаблонов приведена на странице [Как создать свой набор шаблонов](#).



## Требования к квалификации

Квалификация разработчика, необходимая для выполнения описываемых в руководстве действий, зависит от решаемых задач. Для создания простых отчётов выборки данных из базы данных (SQL) или модификации существующих шаблонов средствами визуального редактора достаточно квалификации опытного пользователя ПК и знаний SQL. Для создания собственных форм или полей ввода параметров, для выборки данных не из базы данных, а, например, XML-файла, потребуется опыт разработки программных средств и знание языка Scala (Java).

## Общие сведения

В составе Платформы НЕЙРОСС присутствует веб-приложение «Отчёты», которое позволяет:

1. загрузить шаблоны отчётов,
2. настроить подключение к источникам данных (SQL),
3. для выбранного источника данных и выбранного шаблона отчётов заполнить форму параметров формирования отчёта,
4. сгенерировать отчёт выбранного формата (PDF, XLS и пр.) по заданному источнику данных и шаблону отчёта.

*Шаблон отчёта* — совокупность файлов, содержащих необходимое для Платформы НЕЙРОСС описание того, как формировать отчёт.

Шаблоны отчётов загружаются в Платформу в форме *наборов*. Набор шаблонов — это zip-архив, в котором размещены файлы с определением набора и шаблонов отчётов.

Для добавления в Платформу новых «пользовательских» шаблонов отчётов необходимо создать соответствующий набор с такими шаблонами и загрузить его в программу. Средствами своего набора разработчик может:

1. Добавить в Платформу свои шаблоны отчётов.
2. Произвольным образом формировать отчёты (в рамках своих шаблонов):
  - a. формировать произвольные запросы к базе / базам данных,
  - b. задавать полностью произвольное визуальное оформление отчёта,
  - c. использовать источники данных, отличные от SQL баз данных,
  - d. поддержать новые форматы экспорта отчётов в дополнение к XLS, PDF и HTML.
3. Изменять внешний вид форм ввода параметров:
  - a. настраивать существующие поля ввода данных,
  - b. создавать собственные поля ввода данных — например, список-перечисление для выбора опций, подгружаемых из внешней базы данных,
  - c. использовать свои стили оформления для формы ввода входных параметров,
  - d. и даже формировать форму ввода параметров произвольным образом.

Основной сценарий формирования отчётов в Платформе — подключение к базе данных (SQL) и формирование отчётов с помощью программных компонентов [JasperReports Library](#). При этом разработка шаблонов отчётов преимущественно осуществляется в визуальном редакторе [Jaspersoft Studio](#).

Краткое руководство на русском языке по созданию отчётов с помощью JasperReports и Jaspersoft Studio [доступно по ссылке](#).

Подробная информация по Jaspersoft Studio на английском языке приведена на официальном сайте [jaspersoft.com](#) в [разделе Resources](#).



Настоящее руководство ориентировано только на использование JasperReports Library и Jaspersoft Studio. Формирование отчётов средствами НЕ JasperReports возможно, но обычно не востребовано. Соответствующий сценарий выходит за рамки данного руководства.

## Структура набора шаблонов

Набор шаблонов — это zip-архив, который файлы с определением набора и шаблонов отчётов. Zip-архив набора должен содержать следующие папки и файлы:

```
definitions/ //
  < 1>/
  ... //
  report.conf //
  < 2>/
  ...
libs/ // java-
*.jar // jar-
deployment.conf //
```

Файл `deployment.conf` содержит определение набора, этот файл обязательно должен присутствовать в составе архива. Если файла нет, то Платформа НЕЙРОСС не примет загружаемый zip-архив. Подробное описание данного файла приведено в разделе [Определение набора шаблонов](#).

Для каждого шаблона в наборе должна быть отдельная подпапка в директории `definitions`. В этой подпапке должен присутствовать файл определения шаблона — `reports.conf`. Подробное описание поддиректории шаблона отчёта приведено в разделе [Шаблон отчёта](#).

В директории `libs` могут размещаться jar-файлы, используемые в наборе. При использовании нестандартных компонентов ввода параметров или реализации собственной логики формирования отчёта на языке Scala / Java соответствующий код должен быть собран и в форме jar-файлов помещён в данную директорию. Подробная информация по использованию библиотек приведена в разделе [Программные компоненты](#).

## Определение набора шаблонов

Текстовый файл `deployment.conf` должен состоять из пар `=`, размещённых на отдельных строках. При создании набора необходимо корректно задать в файле значения для следующих ключей:

Ключ	Значение	Комментарий	Пример значения
<code>deployment.key</code>	Строковой идентификатор набора	Для каждого набора такой идентификатор должен быть уникальным. При обновлении набора (например, добавлении в набор новых шаблонов или изменении существующих) <i>не следует</i> менять этот идентификатор, достаточно изменять версию набора (см. ключ <code>deployment.version</code> ).  Идентификатор позволяет различить наборы разных видов. При загрузке набора выполняется проверка идентификатора. Если набор с таким идентификатором уже был загружен в Платформу НЕЙРОСС, то выполняется обновление набора. Если нет, то выполняется установка нового набора.  Следует выбирать идентификатор набора в соответствии с <a href="#">правилами именования пакетов в java</a> .	<code>ultima.reports.itrium</code>
<code>deployment.title</code>	Название набора	Данное название выводится в пользовательском интерфейсе в списке загруженных наборов.  Рекомендуется выбирать удобочитаемое название набора.	Типовые формы отчётов к Itrium
<code>deployment.version</code>	Версия набора	Строка с номером версии. Выводится в пользовательском интерфейсе в списке загруженных наборов.  Рекомендуется придерживаться общих <a href="#">правил нумерации версий программного обеспечения</a> .	<code>0.6.0</code>

Пример файла `deployment.conf` из набора [типовых шаблонов СКУД ITRIUM](#) приведён ниже:

```
deployment.key=ultima.reports.itrium
deployment.title= Itrium
deployment.version=0.6.0
```



Формат файла `deployment.conf` должен отвечать требованиям к файлу конфигурации [scala-библиотеки config](#). Учитывайте эти требования при использовании в файле специальных символов и т.д.

## Шаблон отчёта

Для каждого шаблона в наборе должна быть отдельная подпапка в директории `definitions`. В этой подпапке должен присутствовать файл определения шаблона — `reports.conf`. Текстовый файл `report.conf` должен состоять из пар `ключ=значение`, размещённых на отдельных строках. При создании шаблона необходимо корректно задать в файле значения для следующих ключей:

Ключ	Значение	Комментарий	Пример значения
<code>definition.class</code>	Тип реализации шаблона отчёта	Это имя программного компонента, который будет отвечать за формирование отчёта по данному шаблону.  Подробное описание данного параметра приведено ниже.	<code>extensions.reports.itrium.ReportDefinition</code>
<code>definition.htmlPaging</code>	ЗАРЕЗЕРВИРОВАНО	ЗАРЕЗЕРВИРОВАНО  Значение <u>всегда</u> должно быть равно <code>false</code> .	<code>false</code>
<code>definition.supportedFormats</code>	Список поддерживаемых форматов	Перечень форматов, в которых может быть подготовлен отчёт. Для каждого формата в интерфейсе пользователя будет доступна соответствующая кнопка.  За поддержку форматов отвечает класс реализации. Стандартная реализация поддерживает форматы PDF и XLS. Через данный параметр можно ограничить форматы экспорта, например, если шаблон не позволяет формировать корректный XLS-файл. В своей реализации разработчик может поддержать и другие форматы экспорта.	<code>[ "pdf" , "xls" ]</code>

Параметр `definition.class` является ключевым. В данном параметре необходимо указать имя программного компонента, который будет отвечать за подготовку и формирование отчёта по создаваемому шаблону. Разработчику доступно несколько готовых реализаций / компонентов, которые он может использовать в своём шаблоне. Разработчик может выбрать одно из следующих значений `definition.class`:

Значение	Описание
extensions. reports. GenericJasperReportDefinition	<p><b>Базовая реализация</b> шаблона отчёта с применением JasperReports Library со следующими возможностями:</p> <ol style="list-style-type: none"> <li>1. ключ, название, описание шаблона задаются в файле <code>report.conf</code> (см. далее);</li> <li>2. отчёт формируется в соответствии с <code>jasper</code>-шаблоном средствами JasperReports Library;</li> <li>3. можно формировать отчёты в форматах PDF, XLS (Excel) и HTML (для предпросмотра);</li> <li>4. можно использовать разные шаблоны для предпросмотра и экспорта в PDF / XLS;</li> <li>5. в форме ввода параметров поддерживаются параметры из <code>jasper</code>-шаблона общих типов: строка, число, выбор опций и др.;</li> <li>6. разработчик может добавить свои поля ввода в форме программного модуля.</li> </ol> <p>Все другие реализации шаблонов обычно унаследованы от базовой и лишь расширяют базовые возможности.</p> <p>Подробнее см. страницу <a href="#">Базовый шаблон отчёта с JasperReports</a>.</p>
extensions. reports.itrium. ReportDefinition	<p>Стандартная реализация шаблона отчёта к базе данных <b>платформы ITRIUM</b>. Используется в модуле <b>Типовые шаблоны СКУД к ITRIUM</b>, но также может использоваться и в других пользовательских модулях отчётов к ITRIUM.</p> <p>Обладает всеми возможностями базовой реализации (см. выше) и также позволяет использовать следующие поля ввода на форме ввода параметров:</p> <ol style="list-style-type: none"> <li>1. Выбор одного или нескольких элементов из конфигурации ITRIUM заданного типа.</li> <li>2. Выбор одного или нескольких значений перечислимого свойства ITRIUM заданного типа.</li> <li>3. Выбор одного или нескольких узлов / серверов сети НЕЙРОСС из конфигурации ITRIUM.</li> <li>4. Выбор одной или нескольких организаций и связанных / несвязанных подразделений из конфигурации ITRIUM.</li> <li>5. Выбор одного или нескольких пропусков с возможностью поиска по ФИО и номеру карты.</li> </ol> <p>Для использования данной реализации шаблона необходимо включить в модуль соответствующий <code>jar</code>-файл (см. <a href="#">Программные компоненты</a>).</p> <p>Подробнее см. страницу <a href="#">Шаблон отчёта к ITRIUM с JasperReports</a>.</p>
<Имя класса собственной разработки>	<p>При желании разработчик может создать свой класс и указать его имя в значении для ключа <code>definition.class</code>. В данном классе разработчик может либо расширить одну из существующих реализаций (стандартную, реализацию к ITRIUM), либо реализовать полностью свой алгоритм формирования отчёта.</p>

В зависимости от указанного значения `definition.class` в файле `report.conf` может потребоваться указать и другие параметры. См. описание конкретной реализации шаблонов.

Пример файла `report.conf` из [модуля типовых шаблонов СКУД ITRIUM](#) приведён ниже:

```
definition.class=extensions.reports.itrium.ReportDefinition
definition.htmlPaging=false
definition.supportedFormats=["pdf","xls"]

definition.generic.key="ultima.reports.itrium:byAccessPoints"
definition.generic.title=" "
definition.generic.description=" / "

definition.jasper.design = {
  main: "O5.jasper",
  html: "O5.jasper"
}
```



Формат файла `report.conf` должен отвечать требованиям к файлу конфигурации [scala-библиотеки config](#). Учитывайте эти требования при использовании в файле специальных символов и т.д.

Кроме `report.conf` в поддиректории шаблона отчёта также могут располагаться и другие файлы, необходимые для подготовки отчёта. Например `*.jasper`-файлы (в случае использования JasperReports Library). Состав файлов зависит от выбранной реализации шаблона отчёта.

## Программные компоненты

При загрузке набора шаблонов Платформа НЕЙРОСС распаковывает `zip`-архив и загружает все классы из всех `jar`-файлов (библиотек Java) из директории `libs` в модуле. Эти классы могут быть использованы:

1. для расширения или создания собственной реализации формирования отчёта;
2. для реализации новых полей ввода параметров;
3. для выборки данных, вычислений или преобразования / форматирования данных по вызову из шаблона JasperReports.



Для каждого модуля классы и jar-файлы загружаются независимо (для каждого модуля используется независимый [class loader](#)). Таким образом не может возникнуть коллизии между разными классами с одинаковыми именами или разными версиями одних и тех же классов в нескольких наборах шаблонов.

Разработчик может самостоятельно реализовать соответствующие Java-классы, собрать jar-файл (один или несколько) и включить его в состав набора, поместив все необходимые jar-файлы в директорию `libs` zip-архива набора.

Для пунктов 1 и 2 выше разработчику необходимы определения классов API (программного интерфейса), используемых в Платформе НЕЙРОСС. Такие классы поставляются в артефакте (библиотеке) `ultima-reports-api` вместе с исходным кодом. Каждая версия Платформы НЕЙРОСС использует свою версию API. Последнюю версию `ultima-reports-api` можно загрузить по ссылке ниже.

Версия программы	Версия API	Документация	Файлы
Платформа НЕЙРОСС 19.2	1.4.2	<a href="#">ultima-reports-api:1.4.2</a>	<a href="#">ultima-reports-api_1.4.2_180719.zip</a>

Список версий API для предыдущих версий программ приведён ниже.

Версия программы	Версия API	Документация	Файлы
Платформа НЕЙРОСС 18.3	1.2.2	<a href="#">ultima-reports-api:1.2.2</a>	<a href="#">ultima-reports-api_1.2.2_061218.zip</a>
НЕЙРОСС Отчёты 3.9	1.1.7	<a href="#">ultima-reports-api:1.1.7</a>	<a href="#">ultima-reports-api_1.1.7_270918.zip</a>
НЕЙРОСС Отчёты 1.3.1	1.0.15	—	По запросу



Библиотека API актуальной версии присутствует в составе Платформы НЕЙРОСС.

Включать её в модуль (в директорию `libs`) дополнительно не требуется.

Классы [базовой реализации шаблона отчёта с JasperReports Library](#) входят в API, поэтому для базовой реализации дополнительных jar-файлов включать в директорию `libs` не требуется. Но, например, реализация полей ввода, используемых в [типовых шаблонах СКУД к ITRIUM](#), не входит в API. Соответствующие классы реализованы и упакованы в [отдельный jar-файл](#), который при их использовании необходимо поместить в директорию `libs` в наборе шаблонов.