



# Advanced Excel Report for RAD Studio User's Manual



# **Advanced Excel Report for RAD Studio User's Manual**

© 1999-2024 EMS Software Development

All rights reserved.

This manual documents EMS Advanced Excel Report for RAD Studio, version 1.7.

No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Use of this documentation is subject to the following terms: you may create a printed copy of this documentation solely for your own personal use. Conversion to other formats is allowed as long as the actual content is not altered or edited in any way.

Document generated on: 19.01.2024

# Table of Contents

<b>Part I Welcome to Advanced Excel Report for RAD Studio!</b>	<b>6</b>
Overview .....	6
What's new .....	7
Installation .....	8
Registration .....	10
How to register Advanced Excel Report .....	11
Version history .....	12
Other EMS Products .....	15
<b>Part II Advanced Excel Report for RAD Studio</b>	<b>22</b>
<b>TEXLReport component</b> .....	<b>23</b>
TEXLReport component Reference .....	23
<b>Properties</b> .....	<b>24</b>
Bands.....	25
DataSet.....	26
Dictionary.....	27
FieldSign.....	28
Options.....	29
StoreInDFM.....	30
Template.....	31
TemplSheet.....	32
Title.....	33
<b>Methods</b> .....	<b>34</b>
EditTemplate.....	35
Show .....	36
<b>Events</b> .....	<b>37</b>
OnAfterBuild.....	38
OnBeforeBuild.....	39
OnDataEof.....	40
OnDataFirst.....	41
OnDataNext.....	42
OnDataPrior.....	43
OnFormatCell.....	44
OnGetFieldV value.....	45
OnGetReportFileName.....	46
OnSetCellV value.....	47
<b>Part III Units</b>	<b>49</b>
EXLReport unit .....	49
EXLReportBand unit .....	50
TEXLReportBand object .....	51

Properties.....	52
BandType .....	53
Range .....	54
<b>TEXLReportGroupHeaderBand object .....</b>	<b>55</b>
Properties.....	56
CaseInsensitiveCompare.....	57
FieldName .....	58
FooterBand .....	59
Events.....	60
OnCalcExpression.....	61
<b>TEXLReportBands object .....</b>	<b>62</b>
Properties.....	63
Items .....	64
Methods.....	65
AddBand .....	66
<b>EXLReportDictionary unit .....</b>	<b>67</b>
<b>TEXLReportField object .....</b>	<b>68</b>
Properties.....	69
FieldName .....	70
ValueAsString .....	71
<b>TEXLReportDictionary object .....</b>	<b>72</b>
Properties.....	73
Items .....	74
Methods.....	75
Add .....	76
FieldByName .....	77
FindField .....	78

## Part IV Appendix

**80**

<b>OnAfterBuild event - Example .....</b>	<b>80</b>
<b>OnBeforeBuild event - Example .....</b>	<b>81</b>
<b>User data events - Example .....</b>	<b>82</b>
<b>OnFormatCell event - Example .....</b>	<b>83</b>
<b>OnGetFieldValue event - Example .....</b>	<b>84</b>
<b>OnSetCellValue event - Example .....</b>	<b>85</b>
<b>OnCalcExpression event - Example .....</b>	<b>86</b>

**Part**



# 1 Welcome to Advanced Excel Report for RAD Studio!

## 1.1 Overview

**EMS Advanced Excel Report for RAD Studio** is a powerful band-oriented generator of template-based reports in MS Excel. Easy-to-use component property editors allow you to quickly create powerful reports in MS Excel. Now you can create reports, which can be edited, saved to file and viewed almost on any computer.

Visit our web-site for details: <https://www.sqlmanager.net/>

### Key features

- Creating report templates directly in MS Excel
- Saving template as external \*.xls file or as \*.dfm (.exe) file
- Band-oriented report generator
- Any data source can be used
- Creation of master-detail reports and reports with grouping
- Full integration with Delphi IDE
- High productivity even on slow computers
- Detailed help system and a demo application for quicker mastering the product
- Powerful component and property editors which allow you to set many report parameters at the design-time easily

### Product information

Homepage <https://www.sqlmanager.net/products/tools/excelreport>  
Support Ticket System <https://www.sqlmanager.net/support>  
Register on-line <https://www.sqlmanager.net/products/tools/excelreport/buy>

## 1.2 What's new

**Version****Advanced Excel Report for RAD Studio** 2.0.4**Release date**

December 4, 2023

**What's new in Advanced Excel Report for RAD Studio?**

- Support for RAD Studio 12 Athens implemented.
- Fixed paths for 32-bit Clang compiler in RAD Studio options.

---

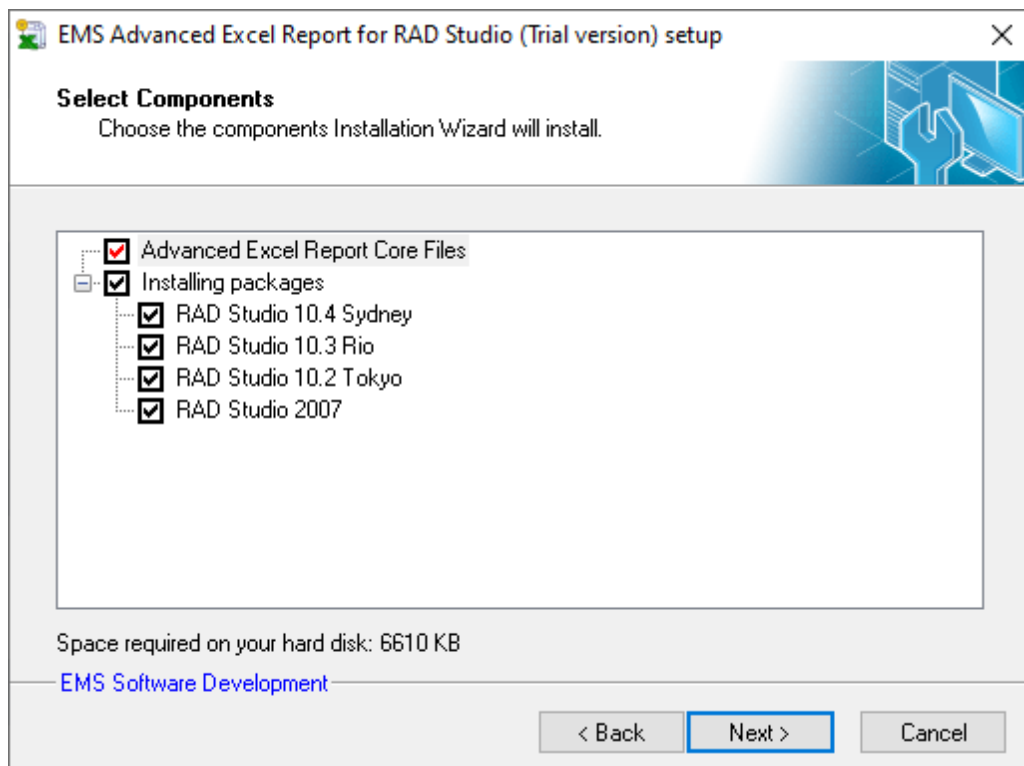
**See also:**[Version history](#)

## 1.3 Installation

To install the **trial version** of **Advanced Excel Report for RAD Studio** onto your system:

- download the distribution package of **Advanced Excel Report for RAD Studio** from the [download page](#) available at our website;
- unzip the downloaded file to any local directory, e.g. *C:\unzipped*;
- close all currently opened Delphi and/or C++ Builder IDEs, if any;
- run the executable setup file from the local directory and follow the instructions of the installation wizard.

During the installation you will need to select the packages to install.



When you are done, you can finish installation of the **trial version** of **Advanced Excel Report for RAD Studio**.

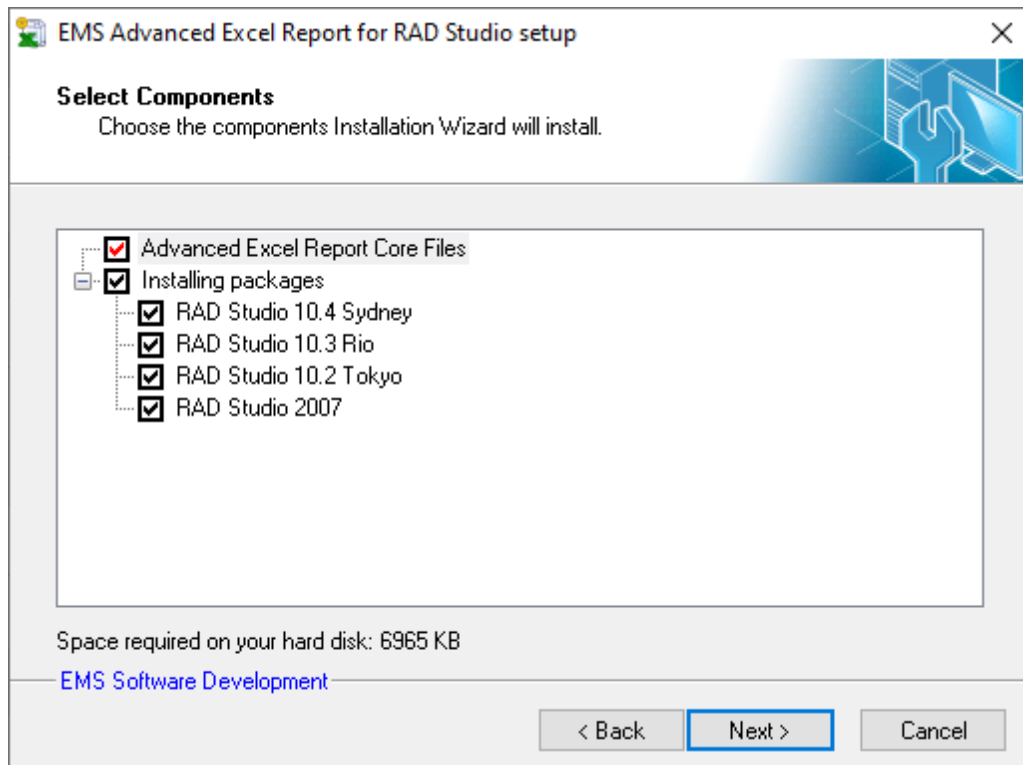
To install the **full version** of **Advanced Excel Report for RAD Studio** onto your system:

- download the distribution package of **Advanced Excel Report for RAD Studio** from the [download page](#) available at our website;
- unzip the downloaded file to any local directory, e.g. *C:\unzipped*;
- close all currently opened Delphi and/or C++ Builder IDEs, if any;
- run the executable setup file from the local directory and follow the instructions of the installation wizard.

Enter valid registration information in the appropriate boxes: **Registration name** and **Registration Key**. See [details](#) on getting this information.

During the installation you will need to select the packages to install.





When you are done, you can finish installation of the **full version** of **Advanced Excel Report for RAD Studio**.

**Note:** If the above given instructions have been insufficient for successful installation of the component, please refer to the *readme.1st* file distributed with the product.

## 1.4 Registration

All purchases are provided by **Digital River** registration service. The **Digital River** order process is protected via a secure connection and makes on-line ordering by credit/debit card quick and safe.

**Digital River** is a global e-commerce provider for software and shareware sales via the Internet. It accepts payments in US Dollars, Euros, Pounds Sterling, Japanese Yen, Australian Dollars, Canadian Dollars or Swiss Franks by Credit Card (Visa, MasterCard/ EuroCard, American Express, Diners Club), Bank/Wire Transfer, Check or Cash.

If you want to review your order information, or you have questions about ordering or payments please visit our [Customer Care Center](#), provided by **Digital River**.

Please note that all of our products are delivered via ESD (Electronic Software Delivery) only. After purchase you will be able to immediately download the registration keys or passwords. Also you will receive a copy of registration keys or passwords by email. Please make sure to enter a valid email address in your order. If you have not received the keys within 2 hours, please, contact us at [sales@sqlmanager.net](mailto:sales@sqlmanager.net).

Product distribution	MyCommerce/Digital River
<b>Advanced Excel Report for RAD Studio</b> Full version (with sources)*	<a href="#">Register Now!</a>
<b>Advanced Excel Report for RAD Studio</b> Trial version	<a href="#">Download Now!</a>

\* **EMS Maintenance Program** provides the following benefits:

- Free software bug fixes, enhancements, updates and upgrades during the maintenance period
- Free unlimited communications with technical staff for the purpose of reporting Software failures
- Free reasonable number of communications for the purpose of consultation on operational aspects of the software

After your maintenance expires you will not be able to update your software or get technical support. To protect your investments and have your software up-to-date, you need to renew your maintenance.

You can easily reinitiate/renew your maintenance with our on-line, speed-through Maintenance Reinstatement/Renewal Interface. After reinitiating/renewal you will receive a confirmation e-mail with all the necessary information.

## 1.5 How to register Advanced Excel Report

To register your newly purchased copy of **EMS Advanced Excel Report for RAD Studio**, perform the following steps:

- receive the notification letter from **Digital River** with the registration info;
- enter the **Registration Name** and the **Registration Key** from this letter while [installing](#) the **full version** of the product.

---

**See also:**

[Registration](#)

## 1.6 Version history

Product name	Version	Release date
Advanced Excel Report for RAD Studio	<a href="#">Version 2.0.3</a>	September 28, 2021
Advanced Excel Report for RAD Studio	<a href="#">Version 2.0.2</a>	June 10, 2020
Advanced Excel Report for RAD Studio	<a href="#">Version 2.0.1</a>	February 15, 2019
Advanced Excel Report for RAD Studio	<a href="#">Version 2.0</a>	June 30, 2017
Advanced Excel Report for RAD Studio	<a href="#">Version 1.9</a>	October 10, 2011
Advanced Excel Report for RAD Studio	<a href="#">Version 1.8</a>	February 20, 2011
Advanced Excel Report for RAD Studio	<a href="#">Version 1.7</a>	December 2, 2009
Advanced Excel Report for RAD Studio	<a href="#">Version 1.6</a>	September 11, 2008
Advanced Excel Report for RAD Studio	<a href="#">Version 1.5</a>	September 21, 2007
Advanced Excel Report for RAD Studio	<a href="#">Version 1.4</a>	April 1, 2006
ExcelReport	<a href="#">Version 1.3</a>	April 6, 2005
ExcelReport	<a href="#">Version 1.2</a>	January 21, 2004
ExcelReport	<a href="#">Version 1.1</a>	April 22, 2003
ExcelReport	<a href="#">Version 1.0</a>	September 16, 2002

Full version history is available at <http://www.sqlmanager.net/products/tools/excelreport/news>

### Version 2.0.3

- Support of RAD Studio 11 Alexandria implemented.
- End of support for RAD Studio 2009 and older versions.

### Version 2.0.2

- Support of RAD Studio 10.4 Sydney added

### Version 2.0.1

- Support of RAD Studio 10.3 Rio implemented

### Version 2.0

- Support of RAD Studio 10.2 Tokyo added.
- Support for 64-bit Windows target platform added.
- Support for C++Builder 2007 and higher has been added.
- Support of several subreports on one sheet in the report has been implemented.
- The Access Violation error occurred in reports with grouping bands. Fixed now.
- The error of processing data beginning with the "=" sign has been fixed.
- Some other minor improvements and bug-fixes.

### Version 1.9

- Added support of Embarcadero RAD Studio XE2

### Version 1.8

- Support of Embarcadero RAD Studio XE is added
- Minor improvements and bug-fixes

### Version 1.7

- Support of BDS 2010 is added
- Minor improvements and bug-fixes

**Version 1.6**

- Support of BDS 2009 is added
- Minor improvements and bug-fixes

**Version 1.5**

- Support of BDS 2007 is added
- Support of MS Office 2007 is added
- Minor improvements and bug-fixes

**Version 1.4**

- Support of BDS 2006 is added
- Minor improvements and bug-fixes

**Version 1.3**

- Quick creation of report templates. Added the corresponding item to the component's context menu
- Support of MS Office 2003 is added
- If a chart was present in a template, in the result report the chart was duplicated several times. The bug is fixed
- If a template file was specified as a network name like \\computer\folder\file.xls, the program worked incorrectly. Fixed now
- Minor improvements and bug-fixes

**Version 1.2**

- We have added a new event - OnGetReportFileName - which allows you to specify the name of the temporary file generated for display in Excel. Use it to make the names of generated files more readable for the user
- Sometimes some characters were interpreted as field sign if MS Excel template contained non-English text
- Automatic Band determination worked incorrectly in some cases
- Minor improvements and bug-fixes

**Version 1.1**

- Added the new SubReport property which allows you to create multisheet reports
- A possibility to use the Unicode strings in the template is implemented
- Now the rightmost report column is built correctly with more than A..Z columns in the template
- A nonvoid cell with an empty string does not appear in the top right corner any more
- An assertion 'Wrong "Num" parameter (EXLReportCopyMngr line 627)' appeared occasionally on building reports with groups. Now it doesn't
- Now you can use the TClientDataSet component linked to the master dataset through a DataSetField property as a detail dataset
- Progress window has changed the Position property from poDesktopCenter to poMainFormCenter as it was incorrectly displayed on computers with dual monitors
- Minor improvements and bug-fixes

**Version 1.0 (First public release)**

- Creating report templates directly in MS Excel
- Possibility of saving template as in the external \*.xls file, as in the \*.dfm(.exe) file
- Borland Delphi 5, 6, 7, and MS Office 97 SR-1, 2000, 2002 (XP) are supported
- Band-oriented report generator
- Any data source can be used

- Possibility of creating master-detail reports and reports with grouping
  - Full integration with Delphi IDE
  - High productiveness even on slow computers
  - Detailed help system and a demo application for quicker mastering the product
  - Powerful component and property editors, which allow you to set many report parameters at the design-time easily
  - Other useful features
- 

**See also:**[What's new](#)

## 1.7 Other EMS Products

### Quick navigation



[MySQL](#)



[Microsoft SQL Server](#)



[PostgreSQL](#)



[InterBase / FireBird](#)



[Oracle](#)



[IBM DB2](#)



[Tools & components](#)

### MySQL



#### [SQL Management Studio for MySQL](#)

EMS SQL Management Studio for MySQL is a complete solution for database administration and development. SQL Studio unites the must-have tools in one powerful and easy-to-use environment that will make you more productive than ever before!



#### [SQL Manager for MySQL](#)

Simplify and automate your database development process, design, explore and maintain existing databases, build compound SQL query statements, manage database user rights and manipulate data in different ways.



#### [Data Export for MySQL](#)

Export your data to any of 20 most popular data formats, including MS Access, MS Excel, MS Word, PDF, HTML and more.



#### [Data Import for MySQL](#)

Import your data from MS Access, MS Excel and other popular formats to database tables via user-friendly wizard interface.



#### [Data Pump for MySQL](#)

Migrate from most popular databases (MySQL, PostgreSQL, Oracle, DB2, InterBase/Firebird, etc.) to MySQL.



#### [Data Generator for MySQL](#)

Generate test data for database testing purposes in a simple and direct way. Wide range of data generation parameters.



#### [DB Comparer for MySQL](#)

Compare and synchronize the structure of your databases. Move changes on your development database to production with ease.



#### [DB Extract for MySQL](#)

Create database backups in the form of SQL scripts, save your database structure and table data as a whole or partially.



#### [SQL Query for MySQL](#)

Analyze and retrieve your data, build your queries visually, work with query plans, build charts based on retrieved data quickly and more.



#### [Data Comparer for MySQL](#)

Compare and synchronize the contents of your databases. Automate your data migrations from development to production database.

[Scroll to top](#)

## Microsoft SQL Server



### [SQL Management Studio for SQL Server](#)

EMS SQL Management Studio for SQL Server is a complete solution for database administration and development. SQL Studio unites the must-have tools in one powerful and easy-to-use environment that will make you more productive than ever before!



### [EMS SQL Backup for SQL Server](#)

Perform backup and restore, log shipping and many other regular maintenance tasks on the whole set of SQL Servers in your company.



### [SQL Administrator for SQL Server](#)

Perform administrative tasks in the fastest, easiest and most efficient way. Manage maintenance tasks, monitor their performance schedule, frequency and the last execution result.



### [SQL Manager for SQL Server](#)

Simplify and automate your database development process, design, explore and maintain existing databases, build compound SQL query statements, manage database user rights and manipulate data in different ways.



### [Data Export for SQL Server](#)

Export your data to any of 20 most popular data formats, including MS Access, MS Excel, MS Word, PDF, HTML and more



### [Data Import for SQL Server](#)

Import your data from MS Access, MS Excel and other popular formats to database tables via user-friendly wizard interface.



### [Data Pump for SQL Server](#)

Migrate from most popular databases (MySQL, PostgreSQL, Oracle, DB2, InterBase/Firebird, etc.) to Microsoft® SQL Server™.



### [Data Generator for SQL Server](#)

Generate test data for database testing purposes in a simple and direct way. Wide range of data generation parameters.



### [DB Comparer for SQL Server](#)

Compare and synchronize the structure of your databases. Move changes on your development database to production with ease.



### [DB Extract for SQL Server](#)

Create database backups in the form of SQL scripts, save your database structure and table data as a whole or partially.



### [SQL Query for SQL Server](#)

Analyze and retrieve your data, build your queries visually, work with query plans, build charts based on retrieved data quickly and more.



### [Data Comparer for SQL Server](#)

Compare and synchronize the contents of your databases. Automate your data migrations from development to production database.

[Scroll to top](#)

## PostgreSQL





### [SQL Management Studio for PostgreSQL](#)

EMS SQL Management Studio for PostgreSQL is a complete solution for database administration and development. SQL Studio unites the must-have tools in one powerful and easy-to-use environment that will make you more productive than ever before!



### [EMS SQL Backup for PostgreSQL](#)

Creates backups for multiple PostgreSQL servers from a single console. You can use automatic backup tasks with advanced schedules and store them in local or remote folders or cloud storages



### [SQL Manager for PostgreSQL](#)

Simplify and automate your database development process, design, explore and maintain existing databases, build compound SQL query statements, manage database user rights and manipulate data in different ways.



### [Data Export for PostgreSQL](#)

Export your data to any of 20 most popular data formats, including MS Access, MS Excel, MS Word, PDF, HTML and more



### [Data Import for PostgreSQL](#)

Import your data from MS Access, MS Excel and other popular formats to database tables via user-friendly wizard interface.



### [Data Pump for PostgreSQL](#)

Migrate from most popular databases (MySQL, SQL Server, Oracle, DB2, InterBase/Firebird, etc.) to PostgreSQL.



### [Data Generator for PostgreSQL](#)

Generate test data for database testing purposes in a simple and direct way. Wide range of data generation parameters.



### [DB Comparer for PostgreSQL](#)

Compare and synchronize the structure of your databases. Move changes on your development database to production with ease.



### [DB Extract for PostgreSQL](#)

Create database backups in the form of SQL scripts, save your database structure and table data as a whole or partially.



### [SQL Query for PostgreSQL](#)

Analyze and retrieve your data, build your queries visually, work with query plans, build charts based on retrieved data quickly and more.



### [Data Comparer for PostgreSQL](#)

Compare and synchronize the contents of your databases. Automate your data migrations from development to production database.

[Scroll to top](#)

## InterBase / Firebird



### [SQL Management Studio for InterBase/Firebird](#)

EMS SQL Management Studio for InterBase and Firebird is a complete solution for database administration and development. SQL Studio unites the must-have tools in one powerful and easy-to-use environment that will make you more productive than ever before!



### [SQL Manager for InterBase/Firebird](#)

Simplify and automate your database development process, design, explore and maintain existing databases, build compound SQL query statements, manage database user rights and manipulate data in different ways.



### [Data Export for InterBase/Firebird](#)

Export your data to any of 20 most popular data formats, including MS Access, MS Excel, MS Word, PDF, HTML and more



### [Data Import for InterBase/Firebird](#)

Import your data from MS Access, MS Excel and other popular formats to database tables via user-friendly wizard interface.



### [Data Pump for InterBase/Firebird](#)

Migrate from most popular databases (MySQL, SQL Server, Oracle, DB2, PostgreSQL, etc.) to InterBase/Firebird.



### [Data Generator for InterBase/Firebird](#)

Generate test data for database testing purposes in a simple and direct way. Wide range of data generation parameters.



### [DB Comparer for InterBase/Firebird](#)

Compare and synchronize the structure of your databases. Move changes on your development database to production with ease.



### [DB Extract for InterBase/Firebird](#)

Create database backups in the form of SQL scripts, save your database structure and table data as a whole or partially.



### [SQL Query for InterBase/Firebird](#)

Analyze and retrieve your data, build your queries visually, work with query plans, build charts based on retrieved data quickly and more.



### [Data Comparer for InterBase/Firebird](#)

Compare and synchronize the contents of your databases. Automate your data migrations from development to production database.

[Scroll to top](#)

## Oracle



### [SQL Management Studio for Oracle](#)

EMS SQL Management Studio for Oracle is a complete solution for database administration and development. SQL Studio unites the must-have tools in one powerful and easy-to-use environment that will make you more productive than ever before!



### [SQL Manager for Oracle](#)

Simplify and automate your database development process, design, explore and maintain existing databases, build compound SQL query statements, manage database user rights and manipulate data in different ways.



### [Data Export for Oracle](#)

Export your data to any of 20 most popular data formats, including MS Access, MS Excel, MS Word, PDF, HTML and more.



### [Data Import for Oracle](#)

Import your data from MS Access, MS Excel and other popular formats to database tables via

user-friendly wizard interface.



#### [Data Pump for Oracle](#)

Migrate from most popular databases (MySQL, PostgreSQL, MySQL, DB2, InterBase/Firebird, etc.) to Oracle



#### [Data Generator for Oracle](#)

Generate test data for database testing purposes in a simple and direct way. Wide range of data generation parameters.



#### [DB Comparer for Oracle](#)

Compare and synchronize the structure of your databases. Move changes on your development database to production with ease.



#### [DB Extract for Oracle](#)

Create database backups in the form of SQL scripts, save your database structure and table data as a whole or partially.



#### [SQL Query for Oracle](#)

Analyze and retrieve your data, build your queries visually, work with query plans, build charts based on retrieved data quickly and more.



#### [Data Comparer for Oracle](#)

Compare and synchronize the contents of your databases. Automate your data migrations from development to production database.

[Scroll to top](#)

## IBM DB2



#### [SQL Manager for DB2](#)

Simplify and automate your database development process, design, explore and maintain existing databases, build compound SQL query statements, manage database user rights and manipulate data in different ways.



#### [Data Export for DB2](#)

Export your data to any of 20 most popular data formats, including MS Access, MS Excel, MS Word, PDF, HTML and more.



#### [Data Import for DB2](#)

Import your data from MS Access, MS Excel and other popular formats to database tables via user-friendly wizard interface.



#### [Data Pump for DB2](#)

Migrate from most popular databases (MySQL, PostgreSQL, Oracle, MySQL, InterBase/Firebird, etc.) to DB2



#### [Data Generator for DB2](#)

Generate test data for database testing purposes in a simple and direct way. Wide range of data generation parameters.



#### [DB Extract for DB2](#)

Create database backups in the form of SQL scripts, save your database structure and table data as a whole or partially.



#### [SQL Query for DB2](#)

Analyze and retrieve your data, build your queries visually, work with query plans, build charts

based on retrieved data quickly and more.

[Scroll to top](#)

## Tools & components



### [Advanced Data Export for RAD Studio VCL](#)

Advanced Data Export for RAD Studio VCL allows you to save your data in the most popular office programs formats.



### [Advanced Data Export .NET](#)

Advanced Data Export .NET is a component for Microsoft Visual Studio .NET that will allow you to save your data in the most popular data formats for the future viewing, modification, printing or web publication. You can export data into MS Access, MS Excel, MS Word (RTF), PDF, TXT, DBF, CSV and more! There will be no need to waste your time on tiresome data conversion - Advanced Data Export will do the task quickly and will give the result in the desired format.



### [Advanced Data Import for RAD Studio VCL](#)

Advanced Data Import for RAD Studio VCL will allow you to import your data to the database from files in the most popular data formats.



### [Advanced PDF Generator for RAD Studio](#)

Advanced PDF Generator for RAD Studio gives you an opportunity to create PDF documents with your applications written on Delphi or C++ Builder.



### [Advanced Query Builder for RAD Studio VCL](#)

Advanced Query Builder for RAD Studio VCL is a powerful component for Delphi and C++ Builder intended for visual building SQL statements for the SELECT, INSERT, UPDATE and DELETE clauses.



### [Advanced Excel Report for RAD Studio](#)

Advanced Excel Report for RAD Studio is a powerful band-oriented generator of template-based reports in MS Excel.



### [Advanced Localizer for RAD Studio VCL](#)

Advanced Localizer for RAD Studio VCL is an indispensable component for Delphi for adding multilingual support to your applications.

[Scroll to top](#)

**Part**



## 2 Advanced Excel Report for RAD Studio

**EMS Advanced Excel Report for RAD Studio** represents a set of tools for efficient reporting.

**Advanced Excel Report for RAD Studio** provides a collection of the following components:

<b>Component</b>	<b>Brief description</b>
<a href="#">TEXLReport</a>	Intended for creating reports in Microsoft Excel

## 2.1 TEXLReport component

### 2.1.1 TEXLReport component Reference

#### Unit

[EXLReport](#)

#### Description

The *TEXLReport* component is intended for creating powerful reports in MS Excel.

The basic method of the component is [Show](#). To define the source of the report data use property [DataSet](#). The [Template](#) property sets the template Excel file, defining the appearance of the result Excel report. For creating complicate master - detail reports or reports with grouping you should define the [Bands](#) property (unnecessary for simple reports). You may also set constant values for some report fields in the [Dictionary](#) property.

## 2.1.2 Properties

▶ Run-time only

🔑 Key properties

- 🔑 [Bands](#)
- 🔑 [DataSet](#)
- 🔑 [Dictionary](#)
- 🔑 [FieldSign](#)
- 🔑 [Options](#)
- 🔑 [StoreInDFM](#)
- 🔑 [Template](#)
- 🔑 [TemplSheet](#)
- 🔑 [Title](#)



### 2.1.2.1 Bands

**Applies to**

[TEXLReport](#) component

**Declaration**

`property` Bands: [TEXLReportBands](#);

**Description**

The *Bands* property is the collection of the [TEXLReportBand](#) and [TEXLReportGroupHeaderBand](#) objects.

You can define the following bands: *Title*, *Group Header*, *Master Data*, *Detail Header*, *Detail Data*, *Detail Footer*, *Group Footer*, and *Summary*.

---

**See also:**

[TEXLReportBands object](#)

[TEXLReportBand object](#)

### 2.1.2.2 DataSet

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property DataSet: TDataSet;
```

**Description**

The *DataSet* property defines the source dataset component (any TDataSet descendant) which contains data for report. You can use data source different from the TDataSet, in this case you should leave the DataSet property empty, and write handler for events [OnDataEof](#), [OnDataFirst](#), [OnDataNext](#), and [OnDataPrior](#).

### 2.1.2.3 Dictionary

**Applies to**

[TEXLReport](#) component

**Declaration**

`property Dictionary: TEXLReportDictionary;`

**Description**

The *Dictionary* property is the collection of [TEXLReportField](#) objects using which you can define constant values for some of the report fields.

---

**See also:**

[TEXLReportDictionary component](#)

[TEXLReportField component](#)

#### 2.1.2.4 FieldSign

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property FieldSign: Char;
```

**Description**

The *FieldSign* property defines the character which denotes the field name in the template Excel file. This can be either [DataSet](#) field name or an arbitrary field name which value you define in the [OnGetFieldValue](#) event handler or which contain constant value defined by the [Dictionary](#) property.

The default property value is '#'. Note that if you want to use a field sign as a simple symbol, you can set two signs one after another, and they will be treated as one symbol not field sign (just as apostrophes in Delphi). The field sign is also ignored if it precedes the equality symbol ('='). This may be necessary if an error occurs on entering an Excel formula to the template cell.

---

**See also:**

[Template property](#)

### 2.1.2.5 Options

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property Options: TEXLReportOptions;  
TXQLOption = (xlroShowProgress, xlroTotalsAsFormula, xlroVisibleWhenError, xlroHourCur  
TXQLOptions = set of TXQLOption;
```

**Description**

Property *Options* is a set of boolean suboptions which define the common settings of the report building.

If *xlroShowProgress* is true, then the progress window is visible during the report building.

If *xlroTotalsAsFormula* is false, then the result of calculating Advanced Excel Report for RAD Studio function (SUM, AVG, etc.) is only a number, otherwise it is the appropriate Excel formula.

If *xlroVisibleWhenError* is true, then the result file becomes visible even if an error occurs during the report building.

If *xlroHourCursor* is true, then the cursor has hour appearance during the report building.

---

**See also:**

[Title property](#)

### 2.1.2.6 StoreInDFM

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property StoreInDFM: Boolean;
```

**Description**

You can store your Excel template not only in external *\*.xls* file, but in the form containing the Advanced Excel Report for RAD Studio component itself. In this case you don't have to define the [Template](#) property, just set *StoreInDFM* to true, and then the template you edit is saved to the *\*.dfm* file of the parent component form.

---

**See also:**

[Template property](#)

[EditTemplate method](#)

### 2.1.2.7 Template

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property Template: String;
```

**Description**

The *Template* property defines the Excel template filename. You can set either full or relative paths to the file. This property is obligatory, except if the [StoreInDFM](#) property is true (in this case the template is stored in the \*.dfm file of the parent component form). The template defines cell appearance for different bands and more.

**Hint:** To learn more about template files, see the demo application.

---

**See also:**

[FieldSign property](#)

[TemplSheet property](#)

### 2.1.2.8 TempSheet

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property TempSheet: String;
```

**Description**

The *TempSheet* property defines the name of the Excel template sheet, according to the template filename, set in the [Template](#) property of the template stored in \*.dfm file (if the [StoreInDFM](#) property is true). If this property is not defined, then the first sheet is taken from file as a template.

---

**See also:**

[Template property](#)



### 2.1.2.9 Title

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property Title: String;
```

**Description**


The *Title* property defines the header of the progress window displayed during the export process.


---

**See also:**

[Options property](#)

## 2.1.3 Methods

 Key methods

 [EditTemplate](#)

 [Show](#)

### 2.1.3.1 EditTemplate

**Applies to**

[TEXLReport](#) component

**Declaration**

```
procedure EditTemplate;
```

**Description**

The *EditTemplate* method opens the template file specified by the [Template](#) property, in MS Excel available for editing.

---

**See also:**

[Template property](#)

### 2.1.3.2 Show

**Applies to**

[TEXLReport](#) component

**Declaration**

```
procedure Show(const FileName: String = '');
```

**Description**

The *Show* method is the basic method of the component. It creates report and saves it to the file specified by *FileName* basing on the template specified by the [Template](#) property, or stored in the parent component form, if property [StoreInDFM](#) is true. If *FileName* is not specified, the result report is opened in the MS Excel after the report is built.

---











**See also:**

[DataSet property](#)

[Template property](#)

## 2.1.4 Events

### Key events

-  [OnAfterBuild](#)
-  [OnBeforeBuild](#)
-  [OnDataEof](#)
-  [OnDataFirst](#)
-  [OnDataNext](#)
-  [OnDataPrior](#)
-  [OnFormatCell](#)
-  [OnGetFieldValue](#)
-  [OnSetCellValue](#)
-  [OnGetReportFileName](#)

#### 2.1.4.1 OnAfterBuild

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnAfterBuild: TEXLReportAfterBuildEvent;
```

```
TEXLReportAfterBuildEvent = procedure(Sender: TObject; Report: ExcelWorksheet; RowCount: Integer; ColCount: Integer);
```

**Description**

The *OnAfterBuild* event takes place when the report is already built. Parameter Report contains properties of the result Excel worksheet. RowCount and ColCount indicate the number of rows and columns in the result file.

---

**See also:**

[OnBeforeBuild event](#)

#### 2.1.4.2 OnBeforeBuild

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnBeforeBuild: TNotifyEvent;
```

**Description**

The *OnBeforeBuild* event takes place before the beginning of building the report.

---

**See also:**

[OnAfterBuild event](#)

### 2.1.4.3 OnDataEof

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnDataEof: TEXLReportCheckEofEvent;  
TEXLReportCheckEofEvent = procedure(Sender: TObject; var Eof: Boolean) of object;
```

**Description**

Use the *OnDataEof* event if you build report from the data source different from the TDataSet descendant to define the condition of finishing the report building. The OnDataEof event takes place before writing each data row to the report, and checks the value of the Eof parameter. If it is true, the report building terminates.

---

**See also:**

[OnDataFirst event](#)

[OnDataNext event](#)

[OnDataPrior event](#)



#### 2.1.4.4 OnDataFirst

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnDataFirst: TNotifyEvent;
```

**Description**

Use the *OnDataFirst* event if you build report from the data source different from the *TDataSet* descendant. This event takes place before the report building starts for you to initialize the data source and set the pointer to the first record.

---

**See also:**

[OnDataEof event](#)

[OnDataNext event](#)

[OnDataPrior event](#)

#### 2.1.4.5 OnDataNext

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnDataNext: TNotifyEvent;
```

**Description**

Use the *OnDataNext* event if you build report from the data source different from the *TDataSet* descendant. This event takes place when the the pointer should be moved to the next record of the data source.

---

**See also:**

[OnDataEof event](#)

[OnDataFirst event](#)

[OnDataPrior event](#)

#### 2.1.4.6 OnDataPrior

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnDataPrior: TNotifyEvent;
```

**Description**

Use the *OnDataPrior* event if you build report from the data source different from the *TDataSet* descendant. This event takes place when the the pointer should be moved to the previous record of the data source.

---

**See also:**

[OnDataEof event](#)

[OnDataFirst event](#)

[OnDataNext event](#)

#### 2.1.4.7 OnFormatCell

##### Applies to

[TEXLReport](#) component

##### Declaration

```
property OnFormatCell: TEXLReportFormatCellEvent;  
TEXLReportFormatCellEvent = procedure(Sender: TObject; Band: TEXLReportBand; RowInTemp  
ColumnInReport: Integer; Report: ExcelWorksheet; const CellValue: OleVariant) of objec
```

##### Description

The *OnFormatCell* event takes place on building a certain cell in the report.

Parameter *Band* is the current Advanced Excel Report for RAD Studio band. *RowInTemplate* shows the template row applied to the current cell. *RowInReport* and *ColumnInReport* indicate the position of the cell in the result file. Parameter *Report* contains properties of the result Excel worksheet, and *CellValue* is the value of the current cell.

It is necessary to consider that in contrast to the [OnGetFieldValue](#) and [OnSetCellValue](#) events, at the moment of the *OnFormatCell* rise the position of the current record in Master and Detail datasets is not defined. That means, you can't use the field values of these datasets in the event handler.

---

##### See also:

[OnSetCellValue event](#)

#### 2.1.4.8 OnGetFieldValue

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnGetFieldValue: TEXLReportGetFieldValueEvent;  
TEXLReportGetFieldValueEvent = procedure(Sender: TObject; const Field: String; var Value: String);
```

**Description**

The *OnGetFieldValue* event takes place when the field name is read from the template, and not found neither in [Dictionary](#), nor in datasets. According to the field name (*Field* parameter) you should set the cell value (output *Value* parameter).

---

**See also:**

[OnSetCellValue event](#)

#### 2.1.4.9 OnGetReportFileName

##### Applies to

[TEXLReport](#) component

##### Declaration

```
property OnGetReportFileName: TEXLReportGetReportFileName;
```

```
TEXLReportGetReportFileName = procedure(Sender: TObject; var FileName: string) of obj
```

##### Description

The *OnGetReportFileName* property allows you to set the name of the temporary file generated for displaying in MS Excel. Use it to make names of the generated files more readable for user.

#### 2.1.4.10 OnSetCellValue

**Applies to**

[TEXLReport](#) component

**Declaration**

```
property OnSetCellValue: TEXLReportSetCellValueEvent;  
TEXLReportSetCellValueEvent = procedure(Sender: TObject; Band: TEXLReportBand; RowInT  
ColumnInReport: Integer; var CellValue: OleVariant) of object;
```

**Description**

The *OnSetCellValue* event takes place when the value of the result cell is set. According to the position of the cell you can change the cell value.

Parameter *Band* is the current Advanced Excel Report for RAD Studio band.

*RowInTemplate* shows the template row applied to the current cell. *RowInReport* and *ColumnInReport* indicate the position of the cell in the result file. Output parameter *CellValue* is the value of the current cell.

---

**See also:**

[OnGetFieldValue event](#)

**Part**





## 3 Units

### 3.1 EXLReport unit

Components

[TEXLReport](#)

## 3.2 EXLReportBand unit

### Objects

[TEXLReportBand](#)

[TEXLReportGroupHeaderBand](#)

[TEXLReportBands](#)

### 3.2.1 **TEXLReportBand** object

**Unit**[EXLReportBand](#)**Description**

The *TEXLReportBand* object contains properties which are common for all the bands created by Advanced Excel Report for RAD Studio.

### 3.2.1.1 Properties

- ▶ Run-time only
- ▶  Key properties
- ▶
  -  [BandType](#)
  -  [Range](#)

## 3.2.1.1.1 BandType

**Applies to**

[TEXLReportBand](#) object

**Declaration**

```
property BandType: TEXLReportBandType;  
TEXLReportBandType = (xlrbtTitle, xlrbtGroupHeader, xlrbtMasterData, xlrbtDetailHeader,  
xlrbtDetailFooter, xlrbtGroupFooter, xlrbtSummary);
```

**Description**

The *BandType* property indicates the type of the current band. The following band types are available: *Title*, *Group Header*, *MasterData*, *Detail Header*, *Detail Data*, *Detail Footer*, *Group Footer*, and *Summary*. This property is read-only.

---

**See also:**

[Range property](#)

## 3.2.1.1.2 Range

**Applies to**

[TEXLReportBand](#) object

**Declaration**

```
property Range: String;
```

**Description**

The *Range* property sets the cell range of the band in the template, e.g. B2, or A1:D4, etc.

---

**See also:**

[BandType property](#)

### 3.2.2 TEXLReportGroupHeaderBand object

**Unit**


[EXLReportBand](#)

**Description**

The TEXLReportGroupHeaderBand object is the [TEXLReportBand](#) descendant. It contains properties, methods and events which allow you to work with data groups.

### 3.2.2.1 Properties

▶ Run-time only

 Key properties

 [CaseInsensitiveCompare](#)

 [FieldName](#)

 [FooterBand](#)



### 3.2.2.1.1 CaseInsensitiveCompare

**Applies to**

[TEXLReportGroupHeaderBand](#) object

**Declaration**

```
property CaseInsensitiveCompare: Boolean;
```

**Description**

If the *CaseInsensitiveCompare* property is true, then when the field values are compared for grouping records by this field, the character case is not taken into consideration.

3.2.2.1.2 `FieldName`**Applies to**

[TEXLReportGroupHeaderBand](#) object

**Declaration**

```
property FieldName: String;
```

**Description**

The *FieldName* property defines the name of the field by which the records are grouped in the result Excel file. If you want to group records not by a single field value, but by the value of some expression, you can leave this property empty, and write handler for the [OnCalcExpression](#) event.

---

**See also:**

[OnCalcExpression event](#)

## 3.2.2.1.3 FooterBand

**Applies to**

[TEXLReportGroupHeaderBand](#) object


**Declaration**

```
property FooterBand: String;
```

**Description**

The *FooterBand* property defines the corresponding group footer by its display name.

### 3.2.2.2 Events

 Key events

 [OnCalcExpression](#)

## 3.2.2.2.1 OnCalcExpression

**Applies to**

[TEXLReportGroupHeaderBand](#) object

**Declaration**

```
property OnCalcExpression: TEXLReportCalcExpressionEvent;
```

```
TEXLReportCalcExpressionEvent = procedure(Sender: TEXLReportBand; var GroupValue: Str...
```

**Description**

Use the *OnCalcExpression* event to set grouping by some expression value. This event takes place before writing each data row to the report. Data are grouped by the output parameter *GroupValue*.

---

**See also:**

[FieldName property](#)

### 3.2.3 **TEXLReportBands** object

**Unit**[EXLReportBand](#)**Description**

The *TEXLReportBands* object is the collection of [TEXLReportBand](#) and [TEXLReportGroupHeaderBand](#) objects each of which defines a data band for the result file. Use property [Items](#) to access certain bands, and use method [AddBand](#) to add a band to the collection.

### 3.2.3.1 Properties

- ▶ Run-time only
- ▶  Key properties
- ▶  [Items](#)

## 3.2.3.1.1 Items

**Applies to**

[TEXLReportBands](#) object

**Declaration**

```
property Items[Index: Integer]: TEXLReportBand;
```

**Description**

Use the *Items* property to access the instances of the collection by Index.

---


**See also:**

[TEXLReportBand](#) object

[TEXLReportGroupHeaderBand](#) object



### 3.2.3.2 Methods

 Key methods

 [AddBand](#)

## 3.2.3.2.1 AddBand

**Applies to**  
[TEXLReportBands](#) object

**Declaration**

```
function AddBand(BandType: TEXLReportBandType): TEXLReportBand;  
TEXLReportBandType = (xlrbtTitle, xlrbtGroupHeader, xlrbtMasterData, xlrbtDetailHeader,  
xlrbtDetailFooter, xlrbtGroupFooter, xlrbtSummary);
```

**Description**

Use the *AddBand* method to add a band to the collection. Parameter *BandType* defines the type of the band.

The following band types are available: *Title*, *Group Header*, *MasterData*, *Detail Header*, *Detail Data*, *Detail Footer*, *Group Footer*, and *Summary*.

---

**See also:**

[TEXLReportBand object](#)

### 3.3 EXLReportDictionary unit

#### Objects

[TEXLReportField](#)

[TEXLReportDictionary](#)

### 3.3.1 TEXLReportField object

**Unit**


[EXLReportDictionary](#)


**Description**

The *TEXLReportField* object contains properties for defining a constant value for the certain report field.

### 3.3.1.1 Properties

▶ Run-time only

 Key properties

 [FieldName](#)

 [ValueAsString](#)

3.3.1.1.1 `FieldName`**Applies to**

[TEXLReportField](#) object

**Declaration**

```
property FieldName: String;
```

**Description**

Property *FieldName* defines the report field name for which the constant value is set ( [ValueAsString](#) property).

---

**See also:**

[ValueAsString property](#)

## 3.3.1.1.2 ValueAsString

**Applies to**

[TEXLReportField](#) object

**Declaration**

```
property ValueAsString: String;
```

**Description**

The *ValueAsString* property defines the constant value of the field specified by the [FieldName](#) property.

---

**See also:**

[FieldName property](#)

### 3.3.2 **TEXLReportDictionary** object

**Unit**

[EXLReportDictionary](#)

**Description**

The *TEXLReportDictionary* object is the collection of [TEXLReportField](#) objects each of which defines a constant value for the certain report field. To access the instances of the collection, use property [Items](#); to add a field with a constant value, use method [Add](#).



### 3.3.2.1 Properties

- ▶ Run-time only
- ▶  Key properties
- ▶  [Items](#)

## 3.3.2.1.1 Items

**Applies to**

[TEXLReportDictionary](#) object

**Declaration**

```
property Items[Index: Integer]: TEXLReportField;
```

**Description**


Use *Items* to access the instances of the collection by Index.

---

**See also:**

[Add method](#)

### 3.3.2.2 Methods

 Key methods

-  [Add](#)
-  [FieldByName](#)
-  [FindField](#)

## 3.3.2.2.1 Add

**Applies to**

[TEXLReportDictionary](#) object

**Declaration**

`function Add: TEXLReportField;`

**Description**

Use method *Add* to add a [TEXLReportField](#) object to the collection.

---

**See also:**

[Items property](#)

## 3.3.2.2.2 FieldByName

**Applies to**

[TEXLReportDictionary](#) object

**Declaration**

```
function FieldByName(const FieldName: String): TEXLReportField;
```

**Description**

As each [TEXLReportField](#) object has [FieldName](#) property, you can access this object by its *FieldName*, using the *FieldByName* method.

---

**See also:**

[FindField method](#)

## 3.3.2.2.3 FindField

**Applies to**

[TEXLReportDictionary](#) object

**Declaration**

```
function FindField(const FieldName: String): TEXLReportField;
```

**Description**

The *FindField* method allows you to find a field in the collection by its [FieldName](#). It works just like the [FieldByName](#) method, but it returns nil in case of unsuccessful execution, when [FieldByName](#) raises exception.

---

**See also:**

[FieldByName method](#)

**Part**



## 4 Appendix

### 4.1 OnAfterBuild event - Example

```
procedure TReportContainer.SheetPropAfterBuild(Sender: TObject;  
    Report: _Worksheet; RowCount, ColCount: Integer);  
var  
    XLApp: Variant;  
begin  
    Report.Name := 'Sheet name';  
    XLApp := Report.Application;  
    XLApp.Run('UserMacro'); // Not to set all the procedure parameters.  
end;
```



## 4.2 OnBeforeBuild event - Example

```
procedure TReportContainer.GroupingBeforeBuild(Sender: TObject);  
begin  
    // Add an index for the grouping  
    try  
        MasterTable.Close;  
        MasterTable.Exclusive := True;  
        MasterTable.Open;  
        MasterTable.AddIndex('ByDate', 'Event_date;Event_name', [ixCaseInsensitive]);  
    except  
    end;  
    MasterTable.Close;  
    MasterTable.Exclusive := False;  
    MasterTable.Open;  
    try  
        MasterTable.IndexName := 'ByDate';  
    except  
    end;  
end;
```

### 4.3 User data events - Example

```
procedure TReportContainer.UserDataDataEof(Sender: TObject);  
    var Eof: Boolean);  
begin  
    Eof := FCounter > 20;  
end;  
  
procedure TReportContainer.UserDataDataFirst(Sender: TObject);  
begin  
    Randomize;  
    FCounter := 1;  
end;  
  
procedure TReportContainer.UserDataDataNext(Sender: TObject);  
begin  
    Inc(FCounter);  
end;  
  
procedure TReportContainer.UserDataDataPrior(Sender: TObject);  
begin  
    Dec(FCounter);  
end;
```

## 4.4 OnFormatCell event - Example

```
procedure TReportContainer.FormatProFormatCell(Sender: TObject;
  Band: TEXLReportBand; RowInTemplate, RowInReport, ColumnInReport: Integer;
  Report: _Worksheet; const CellValue: OleVariant);
var
  R1, R2: String;
begin
  if (Band.BandType = xlrbtMasterData) and (ColumnInReport = 4) then
  begin
    R1 := 'A' + IntToStr(RowInReport);
    R2 := 'D' + IntToStr(RowInReport);
    if CellValue >= 45000 then
    begin
      Report.Range[R2, R2].Font.Color := clRed;
      Report.Range[R2, R2].Font.Bold := True;
    end
    else if CellValue <= 20000 then
    begin
      Report.Range[R1, R2].Font.Color := clWhite;
      Report.Range[R1, R2].Interior.Color := clRed;
    end;
  end;
end;
```

## 4.5 OnGetFieldValue event - Example

```
procedure TReportContainer.UserDataGetFieldValue(Sender: TObject;  
  const Field: String; var Value: OleVariant);  
begin  
  if AnsiCompareText(Field, 'No') = 0 then  
    Value := FCounter  
  else if AnsiCompareText(Field, 'String') = 0 then  
    Value := Format('This is the string %d.', [FCounter])  
  else if AnsiCompareText(Field, 'Number') = 0 then  
    Value := Random(100);  
end;
```

## 4.6 OnSetCellValue event - Example

```
procedure TReportContainer.CellDataSetCellValue(Sender: TObject;  
    Band: TEXLReportBand; RowInTemplate, RowInReport, ColumnInReport: Integer;  
    var CellValue: OleVariant);  
begin  
    if Band.BandType = xlrbtMasterData then  
        CellValue := AnsiUpperCase(CellValue);  
end;
```

## 4.7 OnCalcExpression event - Example

```
procedure TReportContainer.GroupingBandsGroupHeader1CalcExpression(  
    Sender: TExLReportBand; var GroupValue: String);  
begin  
    GroupValue := FormatDateTime('yyyymm', MasterTable.FieldByName('Event_date').AsDate);  
end;
```

# Credits

**Software Developers:**

*Alexander Zhiltsov*

*Alexey Butalov*

*Igor Petrov*

*Dmitry Ziborov*

**Technical Writers:**

*Dmitry Doni*

*Semyon Slobodenyuk*

*Olga Ryabova*

*Natalia Borovaya*

**Cover Designer:**

*Tatyana Makurova*

**Translators:**

*Anna Shulkina*

*Sergey Fominykh*

**Team Coordinators:**

*Alexey Butalov*

*Alexander Chelyadin*

*Roman Tkachenko*