

Record page

Timeline

Version 8.0



This documentation is provided under restrictions on use and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this documentation, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

Table of Contents

Timeline	4
The database tables	4
Adding the Timeline tab to the section	6
Example of using base tile	7
Example implementation algorithm	7
Create the [Timeline] tab tiles bound to custom section	9
Tile setup	9
The algorithm for creating a pre-configured tile	10
Example implementation algorithm	11

Timeline



Advanced

Starting from version 7.12.0 you can use the [*Timeline*] tab for quick analysis of customer cooperation, opportunity, case, etc. history in Creatio. This tab is available by default in the [*Contacts*], [*Accounts*], [*Leads*], [*Opportunities*] and [*Cases*] sections.

The database tables

The following [tables](#) are provided in the database for setting up the timeline:

- `TimelinePageSetting` – for setting up sections and their tiles).
- `TimelineTileSetting` – for setting up all existing and custom timeline tiles.
- `SysTimelineTileSettingLcz` – for localizing tile names.

`TimelinePageSetting` table primary columns

Column	Details
<code>Id</code>	Record identifier.
<code>Key</code>	Key – the name of section page schema. For example, <code>AccountPageV2</code> , <code>ContactPageV2</code> , etc.
<code>Data</code>	Section timeline setup in JSON format.

`TimelineTileSetting` table primary columns

Column	Details
<code>Id</code>	Record identifier.
<code>Name</code>	Tile caption that will be displayed in the filter menu. It must have plural form, for example, “Tasks”. Localization is performed via the <code>SysTimelineTileSettingLcz</code> table. If this field is not populated, the tile caption will be derived from the entity or type schema name.
<code>Data</code>	Section timeline setup in JSON format.
<code>Image</code>	The tile icon that will be displayed in the filter menu and on the left side of the tile on the [<i>Timeline</i>] tab.

Timeline tile configuration parameters in JSON format

Column	Details	If required	Example
--------	---------	-------------	---------

Column	Details	If required	Example
<code>entityConfigKey</code>	Tile key. It should match the <code>id</code> in the <code>TimelineTileSetting</code> table of the corresponding existing tile that should be displayed for the entity.	No	706f803d-6a30-4bcd
<code>entitySchemaName</code>	Name of the entity object schema.	Yes	Activity
<code>referenceColumnName</code>	Name of the object column that will be used for selecting records.	Yes	Account
<code>masterRecordColumnName</code>	Name of the parent record column that will be used for selecting records.	Yes	Id
<code>typeColumnName</code>	Name of the type column.	No	Type
<code>typeColumnValue</code>	Value of the type column.	Should only be applied when <code>typeColumnName</code> is indicated.	fbe0acdc-cfc0-df11
<code>viewModelClassName</code>	The view model class name of the existing tile.	No. If the value is not populated, the <code>BaseTimelineItemViewModel</code> base class will apply.	Terrasoft.Activity
<code>viewClassName</code>	Name of the existing tile view class.	No. If the value is not populated, the <code>BaseTimelineItemView1</code> base class will apply.	Terrasoft.Activity
<code>orderColumn</code>	Column for sorting.	Yes	StartDate
<code>authorColumnName</code>	Column for the author.	Yes	Owner
<code>captionColumnName</code>	Column for the caption.	Yes, if the <code>messageColumnName</code> column is not indicated	Title

Column	Details	IS NOT INDICATED. If required	Example
<code>messageColumnName</code>	Column for messages.	Yes, if the <code>captionColumnName</code> column is not indicated.	DetailedResult
<code>caption</code>	Tile caption that will be displayed in the filter menu. It must have plural form, for example, "Tasks". It is used for setting a tile caption that would differ from the one indicated in the <code>Name</code> field of the corresponding tile setting in <code>TimelinePageSetting</code> .	No	My Activity
<code>columns</code>	Setup array for additional tile columns.	No	
<code>columnName</code>	Path to the entity object column.	Yes	Result
<code>columnAlias</code>	Column alias in the tile model view.	Yes	ResultMessage
<code>isSearchEnabled</code>	Indicates the capability of text search according to the column value (for text columns only).	No	true

Adding the [*Timeline*] tab to the section

To [add the \[*Timeline* \] tab](#) to the section page and display records thereon:

1. Add a new record to the `TimelinePageSetting` table.
2. Populate the corresponding columns. Indicate the section page schema name in the `key` column. For example, if you need to add a tab to the [*Accounts*] section, the `key` column value will be "AccountPageV2". The `Data` column contains the configuration of timeline tiles that are displayed on the indicated section tab in JSON format.

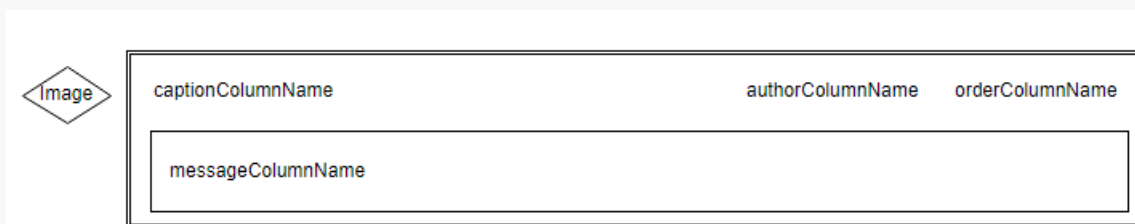
Attention. The [*Timeline*] tab will not be displayed on the section record edit page if the tile configuration in the `Data` column is not available or if there exist errors (for example, syntax error) in the configuration.

Example of using base tile

Advanced

To start using the timeline in a section, perform base tile configuration. The base tile compound elements:

- icon;
- caption;
- author;
- date (sorting);
- message.



Example. Add the [*Contract*] tile to the [*Accounts*] section page. Sorting should be performed according to the `StartDate` column; the caption values, author and tile messages should be derived from the `Number`, `Owner` and `Notes` columns correspondingly.

Example implementation algorithm

1. Add a new record (or update an existing record) in the `TimelinePageSetting` table.
2. Set the "AccountPageV2" value for the `key` column and populate the `Data` column with the following JSON object:

JSON object

```
[
  {
    "entityConfigKey": "0ef5bd15-f3d3-4673-8af7-f2e61bc44cf0",
    "entitySchemaName": "Contract",
    "referenceColumnName": "Order",
    "orderColumnName": "StartDate",
```

```

    "authorColumnName": "Owner",
    "captionColumnName": "Number",
    "messageColumnName": "Notes",
    "caption": "My Contracts",
    "masterRecordColumnName": "Id"
  }
]

```

The [*Orders*] base tile is used in the following case. This tile has a record in the [*TimelineTileSettings*] table with the 0ef5bd15-f3d3-4673-8af7-f2e61bc44cf0 Id.

Attention. As the data in the [*Data*] column are stored in the `varbinary(max)` form, use specific editor (such as dbForge Studio Express for SQL Server) to modify them. To do this:

1. Select a table.
2. Select the necessary column of the record and click the edit button.
3. Enter the text data display mode in the data editor.
4. Add necessary data.
5. Save the changes in the data editor.

The screenshot shows the dbForge Studio Express for SQL Server interface. The Database Explorer on the left shows the 'dbo' schema with various tables. The 'TimelinePageSetting' table is selected and highlighted with a red circle labeled '1'. The Data Viewer shows the table structure and data. The 'Data' column is selected, and the edit button is clicked, opening the Data Editor. The Data Editor shows the JSON data for the selected record, with the 'Data' column highlighted and the edit button clicked, opening the Data Editor. The Data Editor shows the JSON data for the selected record, with the 'Data' column highlighted and the edit button clicked, opening the Data Editor. The Data Editor shows the JSON data for the selected record, with the 'Data' column highlighted and the edit button clicked, opening the Data Editor.

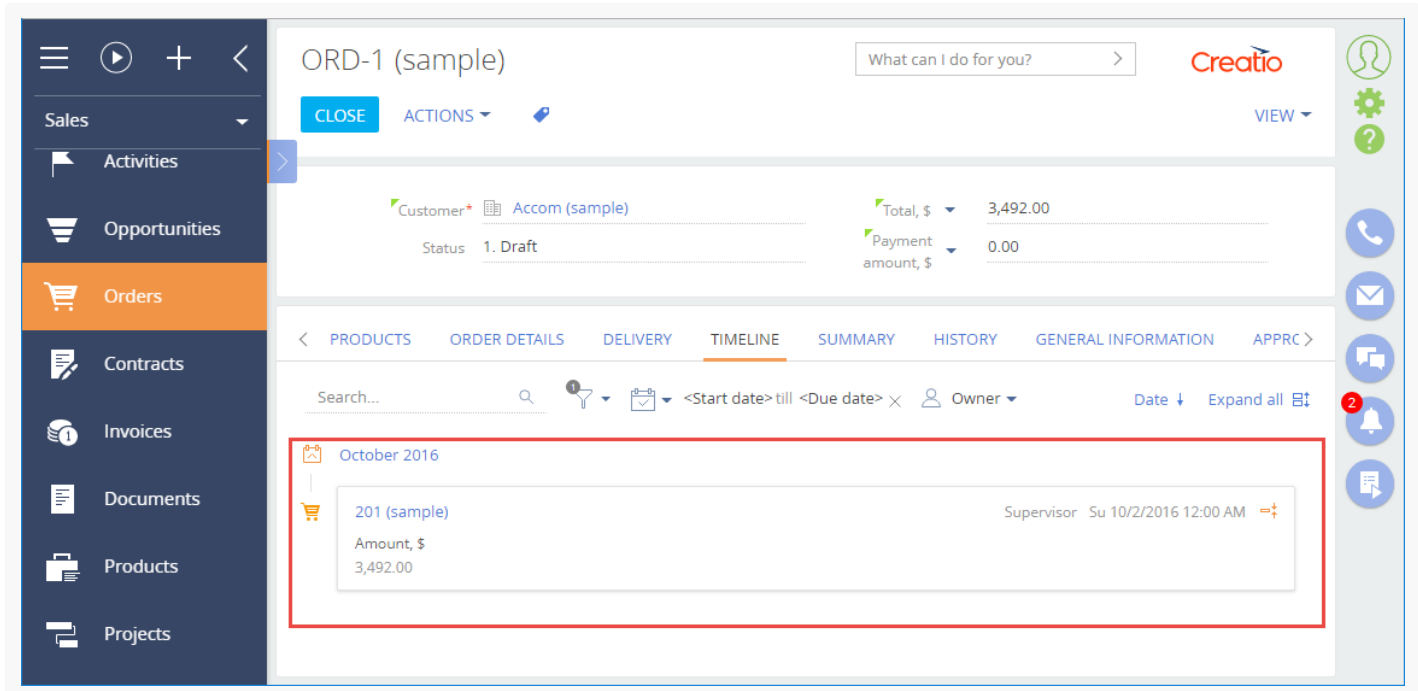
CreatedById	ModifiedOn	ModifiedById	ProcessListeners	Key	Data
302	28.03.2018 23:18:17.1421302	410006e1-ca4e-4502-a9ec-e54d922d2c00	0	ContactPageV2	
384	28.03.2018 23:18:17.1545684	410006e1-ca4e-4502-a9ec-e54d922d2c00	0	LeadPageV2	
700	16.04.2018 11:07:33.8370000	(null)	0	OrderPageV2	
707	28.03.2018 23:18:17.1615707	410006e1-ca4e-4502-a9ec-e54d922d2c00	0	OpportunityPageV2	

```

{
  "entityConfigKey": "0ef5bd15-f3d3-4673-8af7-f2e61bc44cf0",
  "entitySchemaName": "Contract",
  "referenceColumnName": "Order",
  "orderColumnName": "StartDate",
  "authorColumnName": "Owner",
  "captionColumnName": "Number",
  "messageColumnName": "Notes",
  "caption": "My Contracts",
  "masterRecordColumnName": "Id"
}

```


The result of the base tile usage on the [*Timeline*] tab in the [*Accounts*] section:



Create the [Timeline] tab tiles bound to custom section

Advanced

Starting from version 7.12.0 you can use the [*Timeline*] tab for quick analysis of customer cooperation, opportunity, case, etc. history in Creatio. This tab is available by default in the [*Contacts*], [*Accounts*], [*Leads*], [*Opportunities*] and [*Cases*] sections.

Tile setup

You can set up a tile using the [`TimelinePageSetting`] table settings as shown in the [example with the base tile](#). In such a case you will use the following for your tile:

- the default icon;
- the [`BaseTimelineItemView`] and [`BaseTimelineItemViewModel`] base view and view model modules;
- author field;
- tile caption field;
- message field.

You can use one and the same tile for different sections if needed. However, we recommend to use the [`TimelineTileSetting`] table and set up your tiles for different sections.

The [`TimelineTileSetting`] table contains tile configurations that already exist in Creatio. The section, however,

will only display the tiles indicated in the [TimelinePageSetting] table for this particular section.

For example, the [TimelinePageSetting] contains three pre-configured tiles: Tasks, Leads and Calls. The [TimelinePageSetting] table contains the Tasks and Calls tiles that are pre-configured for usage in the [Accounts] section, and only the Calls tile that is pre-configured for usage in the [Contacts] section. The Leads tile in this case will not be displayed in any section.

Note. It is considered a good practice to differentiate the tile settings. We recommend to use the [TimelineTileSetting] > table for setting up tiles, and the TimelinePageSetting - for adding the tiles to section timeline.

Attention. If you need to add the existing Files tile to a section, the "entitySchemaName" property of the [TimelinePageSetting] table should contain the entity schema name for files in the corresponding section configuration (for example, AccountFile, ContactFile, etc.). The object schema name (the "entitySchemaName" property) of the [TimelineTileSetting] table should always look as follows: "##ReferenceSchemaName##File".

The algorithm for creating a pre-configured tile

1. Add a new section (if needed).
2. Add a module schema to your custom package and determine the tile view class, bound to the new section. The class should be the inheritor of BaseTimelineItemView.
3. Add a module schema to your custom package and determine the tile view model class, bound to the new section. The class should be the inheritor of BaseTimelineItemView.
4. Add a record with the tile view settings bound to the new section into the [TimelineTileSetting] database table.
5. In the [TimelinePageSetting] table add or edit the record enabling the tile display on the [Timeline] tab in the necessary section.

Example. Display the tiles bound to the Books custom section on the Timeline tab of the Accounts section. The tiles should contain:

- icon;
- name;
- author;
- book record date;
- price;
- ISBN number.

A short book description should also be displayed when you deploy the tile.

Attention. You should also edit the database tables to implement the case (see steps 4 and 5).

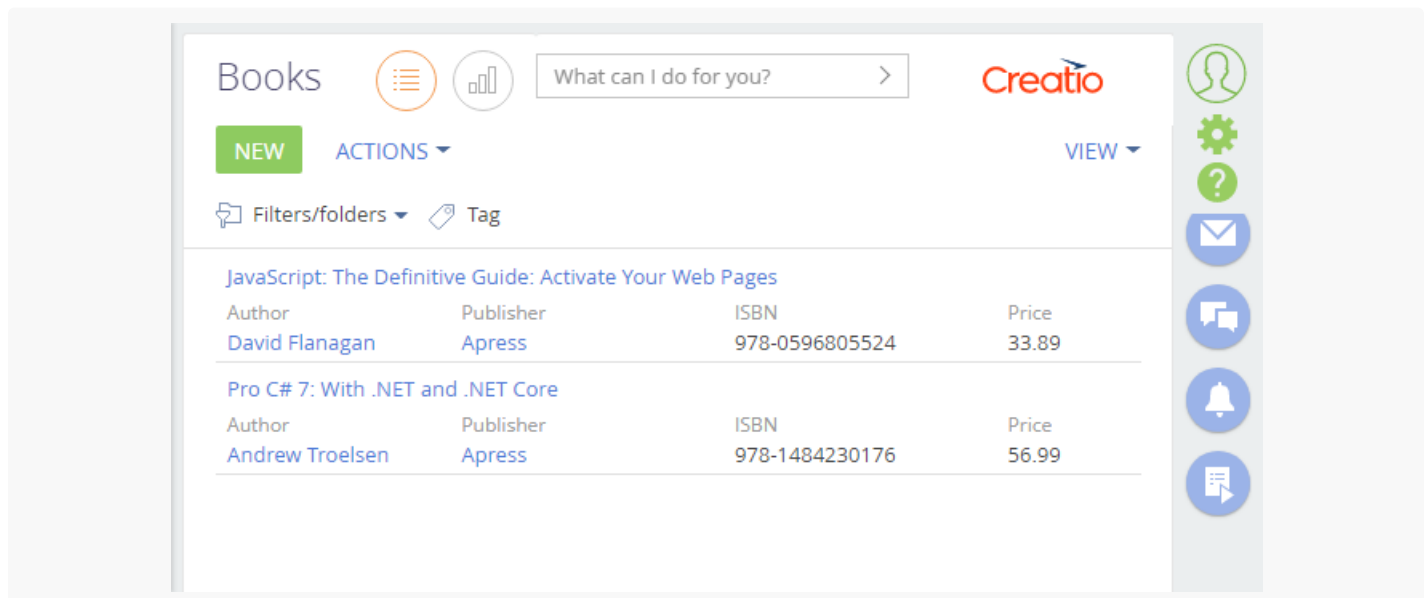
Example implementation algorithm

1. Adding a new [Books] section

Use the archive containing the needed function package to add the new [Books] section. [Install the package](#) via the marketplace application installation function from the *.zip-archive.

Note. You can also add the section via [section wizard](#).

The [Books] section will be available in the [General] workplace after you install the package.



You will also see a detail displaying the linked records from the [Books] section on the [Books] tab of the [Accounts] section record edit page.

The screenshot shows the Apress application interface. On the left, there is a sidebar with a chart showing a 10% increase and an 'Enrich data' button. The main content area has a navigation bar with tabs: ACCOUNT INFO, BOOKS (selected), CONTACTS AND STRUCTURE, TIMELINE, and CONNECT. Below the tabs, there is a 'Books' section with a table of books.

Name	ISBN	Author	Publisher
JavaScript: The Definitive Guide: Activate Your Web Pages	978-0596805524	David Flanagan	Apress
Pro C# 7: With .NET and .NET Core	978-1484230176	Andrew Troelsen	Apress

2. Adding a tile view module

[Add a client module schema](#) containing dependencies from the `Timeline` package to the custom package.

For the created module schema specify:

- [*Name*] - "UsrBookTimelineItemView";
- [*Title*] - "UsrBook Timeline Item View".

The screenshot shows the Properties window in a development tool. The 'General' section is expanded, showing the following properties:

- Title:** UsrBook Timeline Item View
- Name:** UsrBookTimelineItemView
- Package:** sdkTimelineExample

The 'Inheritance' section is also visible, with the 'Parent object' dropdown set to an empty value. There are checkboxes for 'Forbid substitution' and 'Replace parent', both of which are currently unchecked.

Add the following module source code to the [*Source Code*] tab of the schema:

UsrBookTimelineItemView

```
// Defining the module and its dependencies.
define("UsrBookTimelineItemView", ["UsrBookTimelineItemViewResources", "BaseTimelineItemView"],
    // Defining the tile view class.
```

```

Ext.define("Terrasoft.configuration.UsrBookTimelineItemView", {
    extend: "Terrasoft.BaseTimelineItemView",
    alternateClassName: "Terrasoft.UsrBookTimelineItemView",
    // Method returning the [UsrISBN] additional tile field configuration.
    getUsrISBNViewConfig: function() {
        return {
            // Field name.
            "name": "UsrISBN",
            // Field type – label.
            "itemType": Terrasoft.ViewItemType.LABEL,
            // Caption.
            "caption": {
                "bindTo": "UsrISBN"
            },
            // Visibility.
            "visible": {
                // Binding to the tile linked entity column.
                "bindTo": "UsrISBN",
                // Visibility setup.
                "bindConfig": {
                    // A field is visible if its value is populated.
                    "converter": "checkIsEmpty"
                }
            },
            // CSS field styles.
            "classes": {
                "labelClass": ["timeline-text-light"]
            }
        };
    },
    // Method returning the [UsrPrice] additional tile field configuration.
    getUsrPriceViewConfig: function() {
        return {
            "name": "UsrPrice",
            "itemType": Terrasoft.ViewItemType.LABEL,
            "caption": {
                "bindTo": "UsrPrice"
            },
            "visible": {
                "bindTo": "UsrPrice",
                "bindConfig": {
                    "converter": "checkIsEmpty"
                }
            },
            "classes": {
                "labelClass": ["timeline-item-subject-label"]
            }
        };
    }
});

```

```

    },
    // Redefined method returning the [Message] tile field configuration.
    getMessageViewConfig: function() {
        // Receiving standard settings.
        var config = this.callParent(arguments);
        // Visibility setup. Visible if the tile is deployed.
        config.visible = {
            "bindTo": "IsExpanded"
        };
        return config;
    },
    // Redefined method returning general tile configuration.
    getBodyViewConfig: function() {
        // Receiving standard settings.
        var bodyConfig = this.callParent(arguments);
        // Adding additional field configurations.
        bodyConfig.items.unshift(this.getUsrISBNViewConfig());
        bodyConfig.items.unshift(this.getUsrPriceViewConfig());
        return bodyConfig;
    }
});
});

```

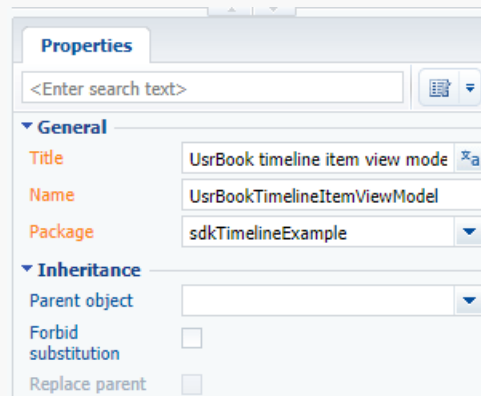
Here you can define the configuration of the [*UsrISBN*] and [*UsrPrice*] fields that are additionally displayed on the tab. The standard configuration is defined in the `BaseTimelineItemView` module.

3. Adding a tile view model module

Add a client module schema containing dependencies from the Timeline package to the custom package.

For the created module schema specify:

- [*Name*] - "UsrBookTimelineItemViewModel";
- [*Title*] - "UsrBook timeline item view model".



Add the following module source code to the [*Source Code*] tab of the schema:

UsrBookTimelineItemViewModel

```
define("UsrBookTimelineItemViewModel", ["UsrBookTimelineItemViewModelResources", "BaseTimelineItemViewModel"], function() {
    Ext.define("Terrasoft.configuration.UsrBookTimelineItemViewModel", {
        alternateClassName: "Terrasoft.UsrBookTimelineItemViewModel",
        extend: "Terrasoft.BaseTimelineItemViewModel"
    });
});
```

You define the `Terrasoft.configuration.UsrBookTimelineItemViewModel` class here. Since this class is defined as the inheritor of `Terrasoft.BaseTimelineItemViewModel`, it enables using all functions of the base class.

4. Adding the record with tile view settings to the [TimelineTileSetting] table

The [TimelineTileSetting] table is used to [set up the timeline tile properties](#).

Add a new record to the [TimelineTileSetting] table. You can add a new record via the following SQL query:

SQL query

```
INSERT INTO TimelineTileSetting (CreatedOn, CreatedById, ModifiedOn, ModifiedById, Name, Data, Image)
VALUES (GETUTCDATE(), NULL, GETUTCDATE(), NULL, 'UsrBooks', NULL, NULL);
```

Since data in the `Data` and `Image` columns are stored in the `varbinary(max)` format, use specific editors (such as dbForge Studio Express for SQL Server) to modify them. To do this:

1. Select a table.
2. Select the necessary record column and click the edit button.
3. Enter the text data display mode in the data editor.
4. Add necessary data.
5. Save the changes in the data editor.
6. Click the data update button.
7. Click OK in the popped up checkout window to apply the modifications.

Attention. This method is only good for the development environments deployed on-site. Since the modifications are implemented directly in the database, they are not bound to any package. That is why the modifications will not be implemented in the database if the package with the view models and the tile view models is installed into another application. For correct transfer of the developed functions you need to bind the SQL-scripts that implement the corresponding modifications in the database when installing the

package.

The screenshot shows the dbForge Studio Express for SQL Server interface. The Database Explorer on the left shows the 'dbo.TimelineTileSetting' table selected (1). The main window displays the table data, with the 'Data' column header selected (2). The toolbar at the bottom shows the 'Data' button (3) and the 'Data' button in the context menu (4). The 'Data' button in the toolbar is also highlighted (5). The 'Data' button in the toolbar is highlighted (6). A dialog box is open, asking to apply changes, with the 'Yes' button highlighted (7).

Id	CreatedOn	CreatedById	ModifiedOn	ModifiedById	Proce...	Name	Data	Image
35b5c45f-36e7-450f-a282-81c56624d29e	28.03.2018 2...	410006e1-...	28.03.2018 ...	410006e1-ca4...	0	ESN Feed	[Data]	[Image]
59de07a7-28dd-4dc9-a106-a07cb9981423	28.03.2018 2...	410006e1-...	28.03.2018 ...	410006e1-ca4...	0	Files	[Data]	[Image]
c57d375e-4ffa-4d65-a59e-d88e53f25803	24.04.2018 1...	(null)	24.04.2018 ...	(null)	0	UsrBooks	[Data]	[Image]
09a6dad5-036b-4075-a813-e8278a5360ea	28.03.2018 2...	410006e1-...	28.03.2018 ...	410006e1-ca4...	0	Links	[Data]	[Image]
aeca6df0-5c89-4066-bdfa-ef486ae8fed	28.03.2018 2...	410006e1-...	28.03.2018 ...	410006e1-ca4...	0	Calls	[Data]	[Image]

```

{
  "entitySchemaName": "UsrBook",
  "viewModelClassName": "Terrasoft.UsrBookTimelineItemViewModel",
  "viewClassName": "Terrasoft.UsrBookTimelineItemView",
  "orderColumnName": "CreatedOn",
  "authorColumnName": "UsrAuthor",
  "captionColumnName": "UsrName",
  "messageColumnName": "UsrDescription",
  "columns": [
    {
      "columnName": "UsrISBN",
      "columnAlias": "UsrISBN"
    },
    {
      "columnName": "UsrPrice",
      "columnAlias": "UsrPrice"
    }
  ]
}

```

Add the following configuration object to the `Data` column using the above mentioned algorithm:

UsrBook

```

{
  "entitySchemaName": "UsrBook",
  "viewModelClassName": "Terrasoft.UsrBookTimelineItemViewModel",
  "viewClassName": "Terrasoft.UsrBookTimelineItemView",
  "orderColumnName": "CreatedOn",
  "authorColumnName": "UsrAuthor",
  "captionColumnName": "UsrName",
  "messageColumnName": "UsrDescription",
  "columns": [
    {
      "columnName": "UsrISBN",
      "columnAlias": "UsrISBN"
    },
    {
      "columnName": "UsrPrice",
      "columnAlias": "UsrPrice"
    }
  ]
}

```



```
]
}
```

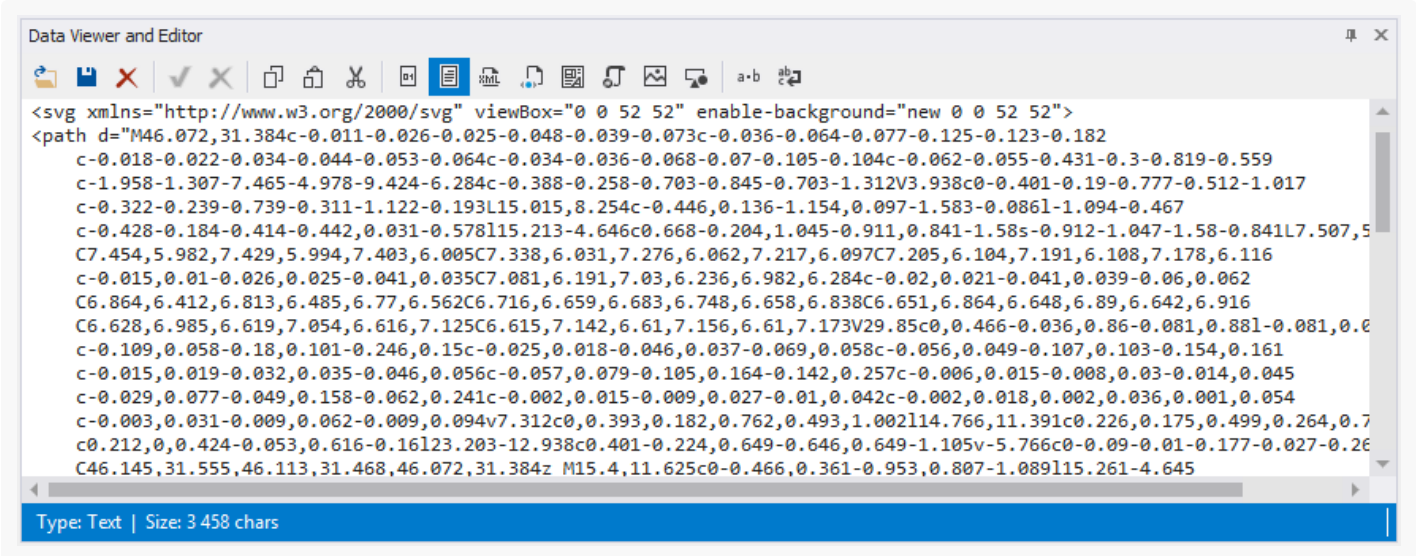
You need to indicate the additional field array whose display is configured in the `UsrBookTimelineItemView` view model in addition to the primary fields inherited from the base tile (see step 2).

Add SVG-format data to the `Image` column to display the icon that corresponds to the section icon.

SVG-format data

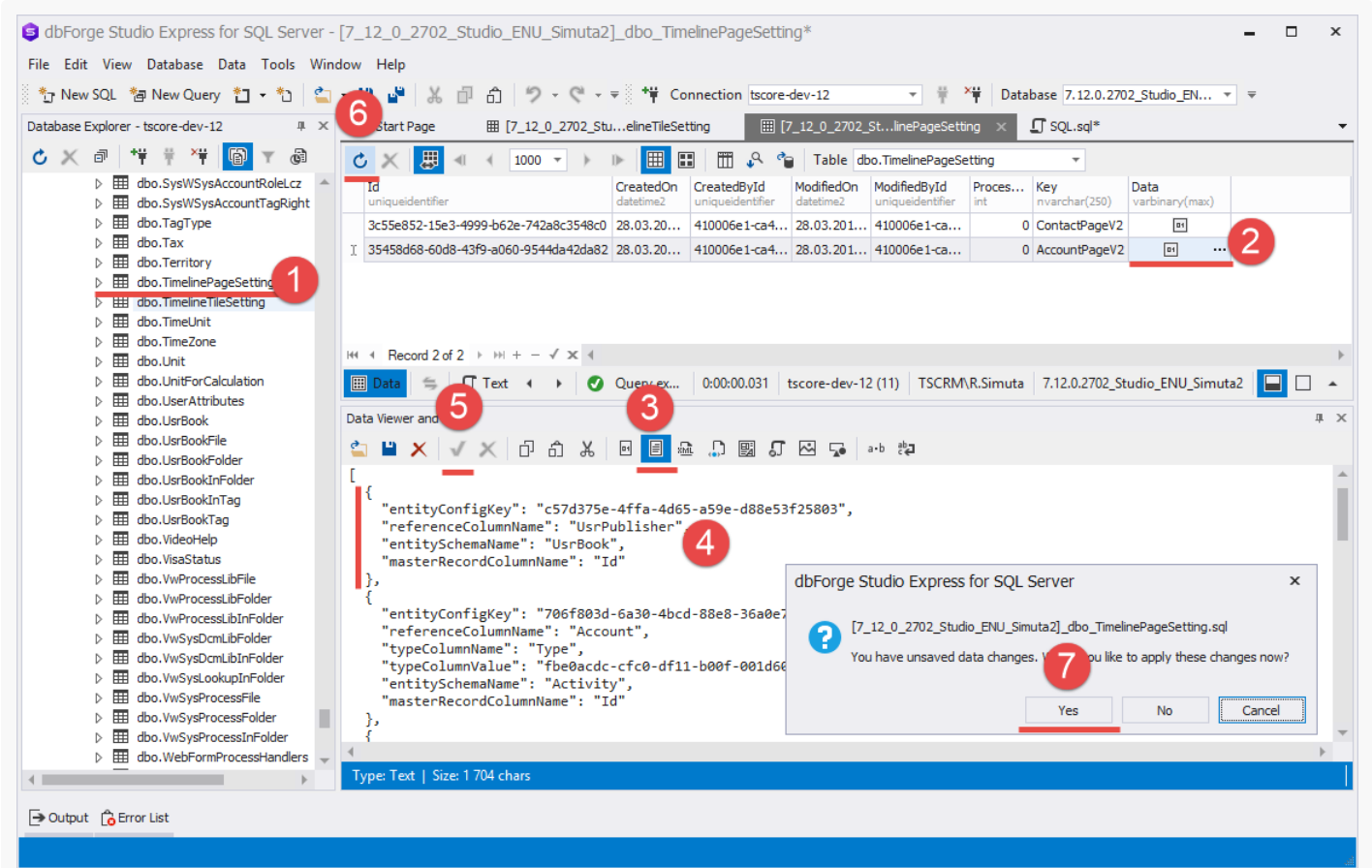
```
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 52 52" enable-background="new 0 0 52 52">
<path d="M46.072,31.384c-0.011-0.026-0.025-0.048-0.039-0.073c-0.036-0.064-0.077-0.125-0.123-0.18
c-0.018-0.022-0.034-0.044-0.053-0.064c-0.034-0.036-0.068-0.07-0.105-0.104c-0.062-0.055-0.431
c-1.958-1.307-7.465-4.978-9.424-6.284c-0.388-0.258-0.703-0.845-0.703-1.312V3.938c0-0.401-0.1
c-0.322-0.239-0.739-0.311-1.122-0.193L15.015,8.254c-0.446,0.136-1.154,0.097-1.583-0.0861-1.0
c-0.428-0.184-0.414-0.442,0.031-0.578L15.213-4.646c0.668-0.204,1.045-0.911,0.841-1.58s-0.912
C7.454,5.982,7.429,5.994,7.403,6.005C7.338,6.031,7.276,6.062,7.217,6.097C7.205,6.104,7.191,6
c-0.015,0.01-0.026,0.025-0.041,0.035C7.081,6.191,7.03,6.236,6.982,6.284c-0.02,0.021-0.041,0.
C6.864,6.412,6.813,6.485,6.77,6.562C6.716,6.659,6.683,6.748,6.658,6.838C6.651,6.864,6.648,6.
C6.628,6.985,6.619,7.054,6.616,7.125C6.615,7.142,6.61,7.156,6.61,7.173V29.85c0,0.466-0.036,0
c-0.109,0.058-0.18,0.101-0.246,0.15c-0.025,0.018-0.046,0.037-0.069,0.058c-0.056,0.049-0.107,
c-0.015,0.019-0.032,0.035-0.046,0.056c-0.057,0.079-0.105,0.164-0.142,0.257c-0.006,0.015-0.00
c-0.029,0.077-0.049,0.158-0.062,0.241c-0.002,0.015-0.009,0.027-0.01,0.042c-0.002,0.018,0.002
c-0.003,0.031-0.009,0.062-0.009,0.094v7.312c0,0.393,0.182,0.762,0.493,1.002L14.766,11.391c0.
c0.212,0,0.424-0.053,0.616-0.161L23.203-12.938c0.401-0.224,0.649-0.646,0.649-1.105v-5.766c0-0
C46.145,31.555,46.113,31.468,46.072,31.384z M15.4,11.625c0-0.466,0.361-0.953,0.807-1.089L15.
c0.446-0.136,0.807,0.132,0.807,0.598v14.63c0,0.467-0.314,0.635-0.702,0.3761-1.127-0.752c-0.3
l-13.059,5.805c-0.426,0.189-0.771-0.034-0.771-0.501C15.4,25.943,15.4,11.625,15.4,11.625z M28
c0.425-0.189,1.085-0.134,1.473,0.125L11.43,7.62c0.388,0.259,0.368,0.644-0.045,0.861-18.404,9
c-0.412,0.216-1.047,0.163-1.418-0.121L-11.789-9.001c-0.371-0.283-0.326-0.665,0.1-0.854L28.85
c0-0.466,0.348-0.695,0.776-0.512L2.174,0.929c0.429,0.183,0.776,0.708,0.776,1.175v2.158c-1.57
L9.142,9.932L9.142,9.932z M9.142,13.152c0.931,0.671,2.22,1.323,3.727,1.372v7.633c-1.57-0.066
C9.142,20.548,9.142,13.152,9.142,13.152z M9.142,21.627c0.931,0.671,2.22,1.323,3.727,1.372v3.
l-2.163,0.876c-0.432,0.175-0.782-0.061-0.782-0.527V21.627z M43.666,36.101c0,0.467-0.33,1.027
c-0.407,0.228-1.036,0.18-1.405-0.104L8.897,39.127c-0.369-0.284-0.668-0.893-0.668-1.358v-2.44
L12.764,9.748c0.225,0.171,0.496,0.26,0.768,0.26c0.201,0,0.403-0.048,0.588-0.146L19.899-10.44
c0.413-0.217,0.747-0.015,0.747,0.452V36.101z" style="fill:#6c91de;"/>
<path d="M33.81,34.064c0.072,0.049,0.155,0.073,0.239,0.073c0.072,0,0.145-0.018,0.209-0.055L4.505
c0.126-0.072,0.207-0.204,0.212-0.349c0.006-0.146-0.063-0.283-0.183-0.365L-9.011-6.192c-0.118
l-5.157,2.123c-0.143,0.059-0.243,0.191-0.259,0.346c-0.017,0.154,0.053,0.304,0.181,0.392L33.8
18.269,5.682L-3.692,2.111-8.803-6.052L29.492,25.426z" style="fill:#6c91de;"/>
</svg>
```

Editing the Image column data via the dbForge Studio Express for SQL Server



5. Editing the record that enables the tile display on the [Timeline] tab of the account page in the [TimelinePageSetting] table

For the [Accounts] section there already exists a record in the [TimelinePageSetting] table with settings of tiles bound to other sections. This is the record containing the "AccountPageV2" value in the Key column.



Attention. Since there are several tiles used on the [*Accounts*] section page timeline, the array of configuration objects enabling the corresponding tile is stored in the [*Data*] column.

Using the algorithm mentioned in step 4, change the configuration object array by adding a new record to it.

Record for configuration object array

```
[
  {
    "entityConfigKey": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "referenceColumnName": "UsrPublisher",
    "entitySchemaName": "UsrBook",
    "masterRecordColumnName": "Id"
  },
  ...
]
```

Here the "entityConfigKey" property: "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" should contain the [TimelineTileSetting] table record identifier created on step 4. In our case it is the "c57d375e-4ffa-4d65-a59e-d88e53f25803" value.

Attention. The "entityConfigKey" should be obligatory indicated. It should match the [*Id*] column value of the record containing settings of the necessary tile in the [TimelineTileSetting] table.

Since the identifiers of the added records are generated at random, the generated identifier in your database will be different from the one we have in our case, when you repeat step 4.

Attention. Be careful when modifying the [*Data*] column value. Incorrect modifications can disrupt the operation of all existing timeline tiles in a section.

As a result of case implementation you will have the tiles bound to the [*Books*] custom section displayed on the [*Timeline*] tab of the [*Accounts*] section page. These tiles contain all the fields we described in our case conditions. The short book description will only be displayed when you deploy the tile.

Apress

What can I do for you? >

Creatio

CLOSE
ACTIONS ▾
📌

PRINT ▾
VIEW ▾

Enrich data

Name*

Apress

Type

Owner

Supervisor

Web

Primary phone

Category

Industry

< ACCOUNT INFO
BOOKS
CONTACTS AND STRUCTURE
TIMELINE
CONNECT >

Search... 🔍

<Start date> till <Due date> ×

Date ↓ Collapse all ☰

Owner ▾

📅 April 2018

📖 Pro C# 7: With .NET and .NET Core Andrew Troelsen Tu 4/24/2018 10:53 AM 🗑️

56.99

978-1484230176

📖 JavaScript: The Definitive Guide: A... David Flanagan... Tu 4/24/2018 10:53 AM 🗑️

33.89

978-0596805524

Since 1996, JavaScript: The Definitive Guide has been the bible for JavaScript programmers—a programmer's guide and comprehensive reference to the core language and to the client-side JavaScript APIs defined by web browsers