

Локализуемые ресурсы

Версия 8.0



Эта документация предоставляется с ограничениями на использование и защищена законами об интеллектуальной собственности. За исключением случаев, прямо разрешенных в вашем лицензионном соглашении или разрешенных законом, вы не можете использовать, копировать, воспроизводить, переводить, транслировать, изменять, лицензировать, передавать, распространять, демонстрировать, выполнять, публиковать или отображать любую часть в любой форме или посредством любые значения. Обратный инжиниринг, дизассемблирование или декомпиляция этой документации, если это не требуется по закону для взаимодействия, запрещены.

Информация, содержащаяся в данном документе, может быть изменена без предварительного уведомления и не может гарантировать отсутствие ошибок. Если вы обнаружите какие-либо ошибки, сообщите нам о них в письменной форме.

Содержание

Понятие локализуемых ресурсов	4
Отобразить локализуемые ресурсы	4
Хранить локализуемые ресурсы	7
Установить привязанные локализуемые ресурсы	10
Операции с локализуемыми ресурсами	11
Использовать Creatio IDE для выполнения операций с локализуемыми ресурсами	11
Использовать базу данных для выполнения операций с локализуемыми ресурсами	14
Использовать систему контроля версий SVN для выполнения операций с локализуемыми ресурсами	20
Использовать файловую систему для выполнения операций с локализуемыми ресурсами	22
Получить локализуемые ресурсы из базы данных	22
Пример 1	23
Пример 2	24
Добавить локализуемую колонку	26
1. Создать объект	26
2. Добавить локализуемую колонку	26
Результат выполнения примера	27
Локализовать представление в базе данных	28
1. Создать схему объекта для представления	29
2. Добавить колонки	29
3. Создать представления в базе данных	31
Результат выполнения примера	32
Класс LocalizableValue<T>	36
Свойства	37
Методы	37

Понятие локализуемых ресурсов



Локализуемые ресурсы — ресурсы, которые используются для отображения интерфейса приложения Creatio в соответствии с языком, установленным в профиле пользователя. К локализуемым ресурсам относятся изображения и локализуемые строки.

Локализуемые ресурсы являются частью конфигурационных ресурсов приложения. Ресурсы приложения размещены в [пакетах](#) и привязаны к базовой схеме. При запросе ресурсов определенной схемы выполняется их сбор по иерархии с учетом уровней и позиций пакетов.

Для локализуемых ресурсов используются понятия основного и дополнительного языка.

Основной язык (основная языковая культура) — язык, который по умолчанию используется для отображения интерфейса приложения Creatio.

Дополнительный язык (дополнительная языковая культура) — язык, который используется для отображения интерфейса приложения Creatio, был изменен в профиле пользователя и отличается от основного языка.

Чтобы **изменить основной язык**, необходимо выполнить его активацию. Активация языка интерфейса описана в статье [Настроить язык интерфейса](#)

При настройке системы в мастерах разделов и деталей, дизайнерах процессов и кейсов, а также в разделе [*Переводы*] ([*Translation*]) используются все языки, которые установлены в системе. В других случаях используются только активированные языки (т. е. языки, для которых установлен признак [*Активен*] ([*Active*])). Такое разделение позволяет уменьшить время выполнения задач, например, вход в систему, открытие страницы записи и т. д. Подробнее читайте в статьях [Настроить язык интерфейса](#) и [Перевести элементы интерфейса в разделе \[*Переводы* \]](#).

Типы локализуемых ресурсов, которые реализованы в Creatio:

- Простые локализуемые ресурсы.
- Привязанные локализуемые ресурсы.

Отобразить локализуемые ресурсы

При отображении локализуемых ресурсов учитываются:

- Режим отображения.
- Способ отображения.

Режимы отображения локализуемых ресурсов

Режимы отображения локализуемых ресурсов, которые доступны в Creatio:

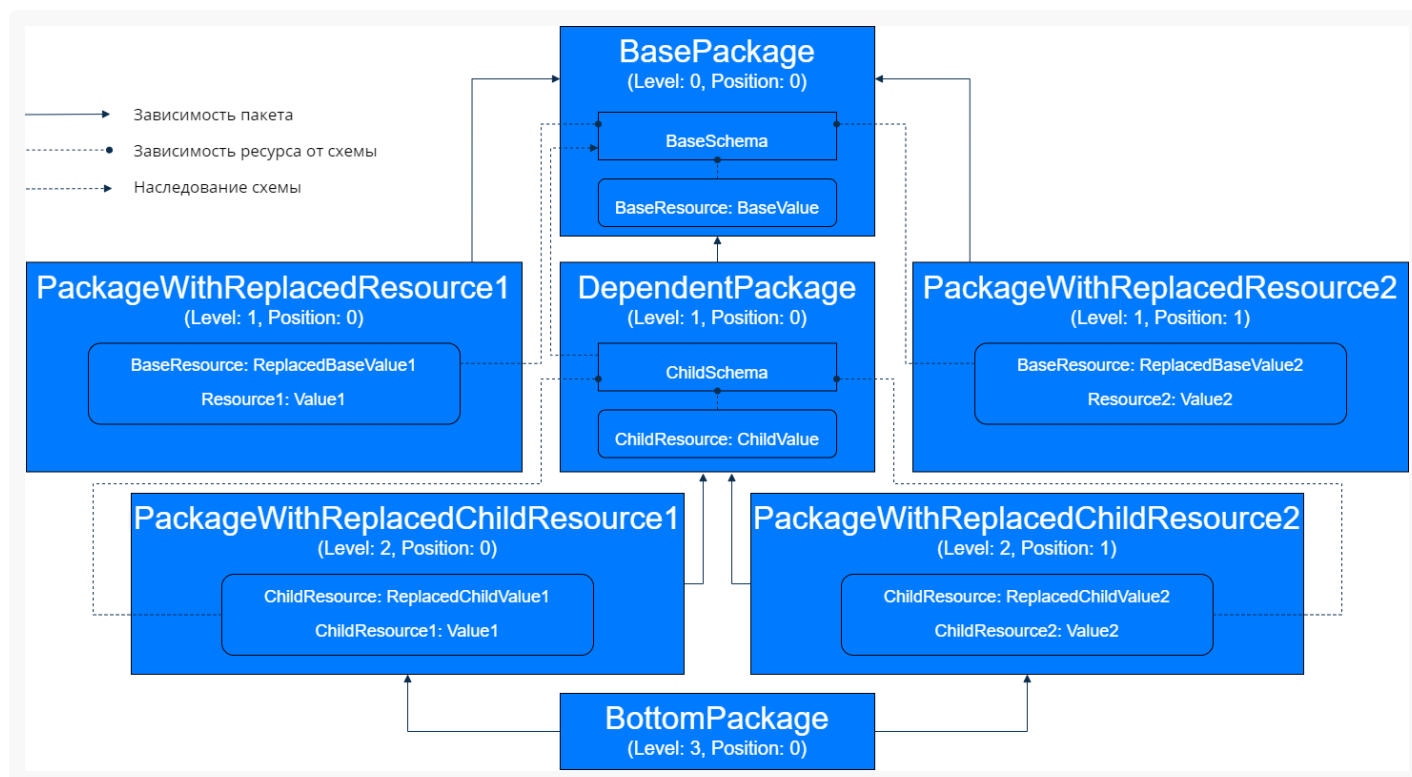
- Режим дизайна (Design-time).
- Режим выполнения приложения (Run-time).

При отображении локализуемых ресурсов учитывается иерархия пакетов.

Режим дизайна

Назначение режима дизайна — отображение локализуемых ресурсов в дизайнерах и мастерах. В этом режиме иерархия локализуемых ресурсов [схем](#) конфигурационных элементов строится только до уровня пакета, в котором содержится схема с запрашиваемыми ресурсами. При построении иерархии учитываются локализуемые ресурсы пакетов, которые попадают в иерархию [по прямым связям](#).

Рассмотрим отображение локализуемых ресурсов на примере ресурса `ChildResource: ChildValue` схемы `ChildSchema` пакета `DependentPackage`. Иерархия пакетов представлена на рисунке ниже.



Результирующие ресурсы для схемы `ChildSchema` в режиме дизайна:

- `BaseResource: BaseValue`.
- `ChildResource: ChildValue`.

В режиме дизайна в ресурсах не учитываются:

- Ресурсы пакетов `PackageWithReplacedResource1` и `PackageWithReplacedResource2`, поскольку эти пакеты не участвуют в построении иерархии.
- Ресурсы пакетов `PackageWithReplacedChildResource1` и `PackageWithReplacedChildResource2`, поскольку эти пакеты относительно запрашиваемой схемы находятся ниже в иерархии.

Если запрашивается схема с указанием стартового пакета (пакета, с которого необходимо начать сбор ресурсов), то результирующий набор ресурсов формируется до уровня запрашиваемого пакета.

Результирующие ресурсы для схемы `ChildSchema` до уровня пакета `BottomPackage` в режиме дизайна:

- `BaseResource: BaseValue .`
- `ChildResource: ReplacedChildValue2 .`
- `ChildResource1: Value1 .`
- `ChildResource2: Value2 .`

Здесь значение ресурса `ChildResource` изменилось на `ReplacedChildValue2`, поскольку в пакетах, которые по иерархии находятся на уровень ниже (`Level 2`), было выполнено замещение исходного ресурса. При этом учитывается позиция и имя пакета — преимущество отдается пакету с большим значением позиции. Для сортировки пакетов с одинаковым значением позиции используется алфавитный порядок.

Режим выполнения приложения

Назначение режима выполнения приложения — отображение локализуемых ресурсов в разделах приложения, за исключением дизайнеров и мастеров. **Отличие** режима выполнения приложения от режима дизайна — при запросе схемы в результирующем списке ресурсов учитываются ресурсы пакетов, которые не входят в иерархию.

Результирующие ресурсы для схемы `ChildSchema` в режиме выполнения приложения:

- `BaseResource: ReplacedBaseValue2 .`
- `Resource1: Value1 .`
- `Resource2: Value2 .`
- `ChildResource: ReplacedChildValue2 .`
- `ChildResource1: Value1 .`
- `ChildResource2: Value2 .`

Способы отображения локализуемых ресурсов

Способы отображения локализуемых ресурсов, которые реализованы в Creatio:

- Если в профиле пользователя **выбран основной язык**, то для отображения используется значение локализуемого ресурса на основном языке.
- Если в профиле пользователя **выбран дополнительный язык и присутствует значение локализуемого ресурса на дополнительном языке**, то для отображения используется значение локализуемого ресурса на дополнительном языке.
- Если в профиле пользователя **выбран дополнительный язык и отсутствует значение локализуемого ресурса на дополнительном языке**, то для отображения используется значение локализуемого ресурса на основном языке.

Классы, которые реализуют логику отображения локализуемых ресурсов:

- `Terrasoft.Common.LocalizableString` — работа с локализуемыми строками.
- `Terrasoft.Common.LocalizableImage` — работа с локализуемыми изображениями.

Свойства и методы, которые используются для получения значения локализуемого ресурса:

- `Value` — свойство, которое возвращает значение локализуемого объекта для текущего языка. Если не найдено значение локализуемого объекта для текущего языка, то будет возвращено значение для основного языка.
- `HasValue` — свойство, которое возвращает признак наличия значения локализуемого объекта для текущего языка. Если не найдено значение локализуемого объекта для текущего языка, то будет возвращено значение для основного языка.
- `GetCultureValue()` — метод, который возвращает значение локализуемого объекта для текущего языка. Если не найдено значение локализуемого объекта для текущего языка, то будет возвращено значение для основного языка.
- `HasCultureValue()` — метод, который возвращает признак наличия значения локализуемого объекта для запрашиваемого языка без учета основного языка.

Класс `Terrasoft.Common.LocalizableString` описан в [Библиотеке .NET классов](#). Класс `Terrasoft.Common.LocalizableImage` описан в [Библиотеке .NET классов](#).

Хранить локализуемые ресурсы

В зависимости от типа локализуемого ресурса, хранение имеет свои особенности.

Хранить простые локализуемые ресурсы

В качестве хранилищ простых локализуемых ресурсов используются:

- **База данных** — хранит ресурсы, которые необходимы для работы приложения. Основное хранилище локализуемых ресурсов.
- **Хранилище SVN** — хранит ресурсы, которые необходимо установить в приложение или перенести между [средами разработки](#). Предварительно необходимо выполнить экспорт локализуемых ресурсов в хранилище SVN.

Хранить простые локализуемые ресурсы в базе данных

Для хранения локализуемых ресурсов используется таблица `[SysLocalizableValue]` базы данных, в которой локализуемые ресурсы хранятся в текстовом формате в виде пар "ключ-значение". Каждая запись привязана к пакету и базовому идентификатору схемы.

Описание основных полей таблицы `[SysLocalizableValue]` базы данных приведено ниже.

Основные поля таблицы [SysLocalizableValue]

Колонка	Описание
[Id]	Идентификатор записи.
[SysPackageId]	Идентификатор пакета.
[SysSchemaId]	Идентификатор базовой схемы. Заполняется только для ресурсов конфигурации.
[ResourceManager]	Название менеджера ресурсов. Заполняется только для ресурсов ядра.
[SysCultureId]	Идентификатор языковой культуры.
[ResourceType]	Тип ресурса.
[IsChanged]	Признак изменения локализуемого ресурса пользователем.
[Key]	Ключ локализуемого ресурса.
[Value]	Значение строкового ресурса.
[ImageData]	Значение графического ресурса.

Если в профиле пользователя активной является дополнительная языковая культура, то при добавлении локализуемого ресурса в таблице [SysLocalizableValue] создается соответствующая запись. Пользователи других языковых культур имеют доступ к добавленному локализуемому ресурсу путем использования **механизма дублирования в основную языковую культуру**. Назначение механизма — создание аналогичной записи локализуемого ресурса со ссылкой на основную языковую культуру. Если в других языковых культурах не задано значение для текущего локализуемого ресурса, то будет отображаться значение, установленное в основной языковой культуре.

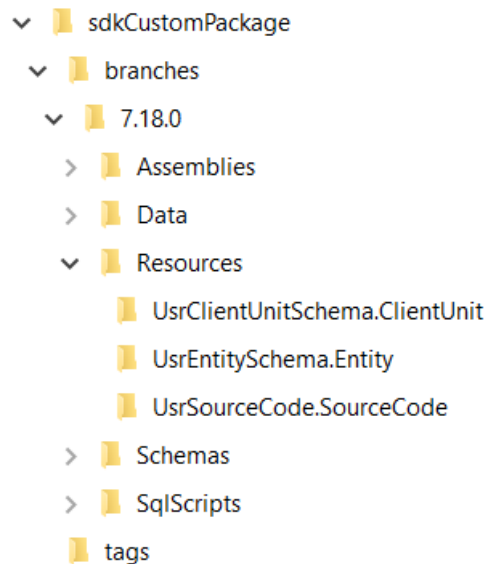
Для каждого набора ресурсов схемы в связке пакет — схема — культура в таблице [SysPackageResourceChecksum] базы данных хранится **контрольная сумма**, которая при обновлении пакета позволяет определить наличие изменений в его ресурсах. Благодаря использованию контрольной суммы ресурсы отделены от схем, что позволяет создавать пакеты переводов.

Хранить простые локализуемые ресурсы в хранилище SVN

Для хранения локализуемых ресурсов пакета в хранилище SVN предназначен каталог [Resources], который используется для создания пакета переводов. **Пакет переводов** — пакет, который содержит только локализуемые ресурсы и не содержит схем конфигурационных элементов. Пакет переводов может содержать локализуемые ресурсы для схемы, которая находится в другом пакете.

Для хранения локализуемых ресурсов схем с одинаковыми именами, но с разными менеджерами (например, [Entity] и [ClientUnit]), в имена локализуемых ресурсов пакета добавляются имена менеджеров схем без префикса [SchemaManager]. В экспортированных схемах локализуемые ресурсы хранятся в

формате *.xml.



Хранить привязанные локализованные ресурсы

Хранилища привязанных локализуемых ресурсов идентичны [хранилищам простых локализуемых ресурсов](#).

Хранить привязанные локализуемые ресурсы в базе данных

Для хранения привязанных локализуемых ресурсов используется таблица `[SysPackageDataLcz]` базы данных.

Описание основных полей таблицы `[SysPackageDataLcz]` базы данных приведено ниже.

Основные поля таблицы `[SysPackageDataLcz]`

Колонка	Описание
<code>[Id]</code>	Идентификатор записи.
<code>[SysPackageSchemaDataId]</code>	Идентификатор привязки в таблице <code>[SysPackageSchemaData]</code> .
<code>[SysCultureId]</code>	Идентификатор языковой культуры.
<code>[Data]</code>	Привязанные локализуемые данные.

Особенности сохранения привязанных локализуемых ресурсов:

- Если схема **содержит привязанные локализуемые ресурсы**, то данные сохраняются в таблицу `[SysPackageDataLcz]` базы данных.
- Если схема **не содержит привязанные локализуемые ресурсы**, то данные сохраняются в

таблицу [SysPackageSchemaData] базы данных.

Запись таблицы [SysPackageDataLcz] содержит:

- Ссылку на соответствующий идентификатор записи в таблице [SysPackageSchemaData] базы данных.
- Ссылку на идентификатор соответствующей языковой культуры в таблице [SysCulture] базы данных.

Например, если в системе используются английская и русская языковые культуры, то каждой записи таблицы [SysPackageSchemaData] соответствуют две записи в таблице [SysPackageDataLcz].

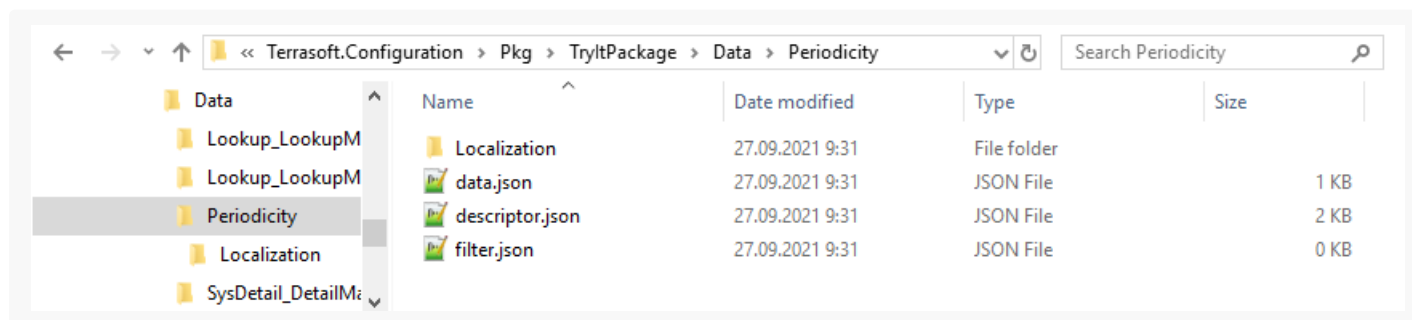
Хранить привязанные локализуемые ресурсы в хранилище SVN

Для хранения привязанных локализуемых ресурсов пакета в хранилище SVN предназначен каталог Data .

Структура каталога Data :

- Файл data.json — нелокализуемые ресурсы.
- Каталог Localization — привязанные локализуемые ресурсы. Каталог содержит соответствующие файлы для языковых культур. Название файла: data.КодЯзыковойКультуры.json (например, data.en-US.json).

Структура хранения привязанных локализуемых ресурсов рассмотрена на примере схемы Periodicity пакета TryItPackage и отображена на рисунке ниже.



Установить привязанные локализуемые ресурсы

Особенности установки привязанных локализуемых ресурсов:

- Если схема **не содержит привязанные локализуемые ресурсы**, то установка выполняется в основную таблицу объекта Entity .
- Если схема **содержит привязанные локализуемые ресурсы** (т. е. в таблице [SysPackageDataLcz] присутствуют соответствующие записи), то установка выполняется в основную таблицу базы данных соответствующей схемы и в таблицу локализации.

Шаблон формирования имени таблицы локализации: [SysИмяОсновнойТаблицыLcz] .

Например, при установке привязанных локализуемых ресурсов для схемы ContactType :

- Нелокализуемые данные будут установлены в таблицу `[ContactType]` базы данных.
- Локализуемые данные будут установлены в таблицу `[ContactType]` и в таблицу `[SysContactTypeLcz]` базы данных.

На заметку. Для таблицы локализации, которая соответствует системной таблице (т. е. название начинается с префикса `sys`), префикс `sys` повторно не добавляется. Например, для таблицы `[SysTestSchema]` базы данных таблица локализации называется `[SysTestSchemaLcz]`, а не `[SysSysTestSchemaLcz]`.

Операции с локализуемыми ресурсами



Инструменты, которые позволяют выполнять операции с локализуемыми ресурсами:

- Creatio IDE.
- База данных.
- Система контроля версий SVN.
- Файловая система.

Использовать Creatio IDE для выполнения операций с локализуемыми ресурсами

Использование Creatio IDE позволяет выполнять следующие **операции с локализуемыми ресурсами**:

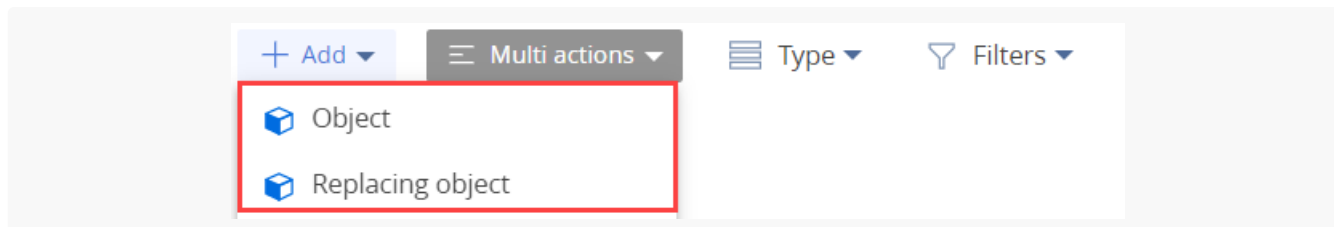
- Добавить локализуемую колонку.
- Добавить локализуемую строку.

Добавить локализуемую колонку

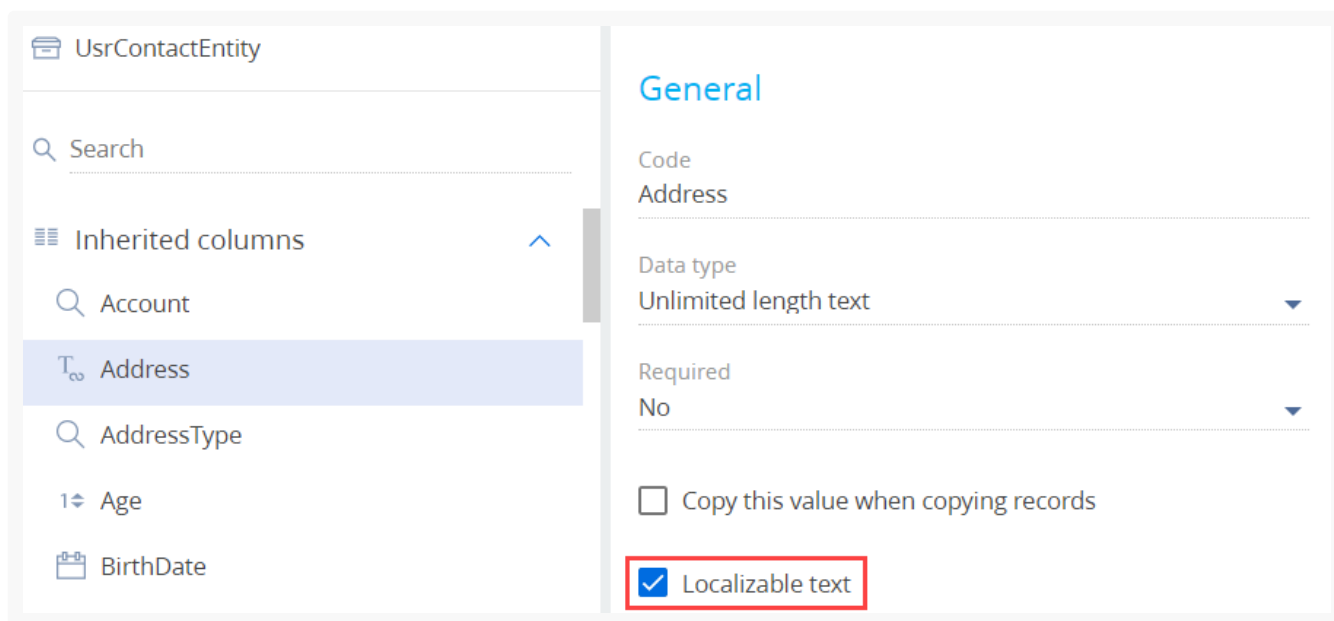
Назначение локализуемой колонки — возможность отображения в интерфейсе приложения данных объекта на нескольких языках (т. н. мультязычие). Значение локализуемой колонки зависит от выбранного в профиле пользователя языка. Локализуемая колонка настраивается в дизайнера объекта, а ее значение хранится в базе данных.

Чтобы **добавить локализуемую колонку**:

1. Добавьте колонку, которую необходимо локализовать.
 - a. [Перейдите в раздел \[Конфигурация \]](#) ([*Configuration*]) и выберите пользовательский [пакет](#), в который будет добавлена схема.
 - b. На панели инструментов реестра раздела нажмите [*Добавить*] ([*Add*]) и выберите вид схемы ([*Объект*] ([*Object*]) или [*Замещающий объект*] ([*Replacing object*])).



- c. В дизайнере объектов заполните свойства схемы.
- d. При необходимости добавьте колонки, которые требуется локализовать, или выберите существующую колонку объекта.
- e. В блоке свойств [*Общие*] ([*General*]) соответствующей колонки установите признак [*Локализуемый текст*] ([*Localizable text*]).



На заметку. Типы колонок, которые можно локализовать:

- [*Строка (50 символов)*] ([*Text (50 characters)*]).
- [*Строка (250 символов)*] ([*Text (250 characters)*]).
- [*Строка (500 символов)*] ([*Text (500 characters)*]).
- [*Строка неограниченной длины*] ([*Unlimited length text*]).

- j. На панели инструментов дизайнера объектов нажмите [*Сохранить*] ([*Save*]), а затем [*Опубликовать*] ([*Publish*]).
2. Выполните перевод значения локализуемой колонки. Для этого воспользуйтесь инструкцией статьи [Перевести элементы интерфейса в разделе \[Переводы \]](#).

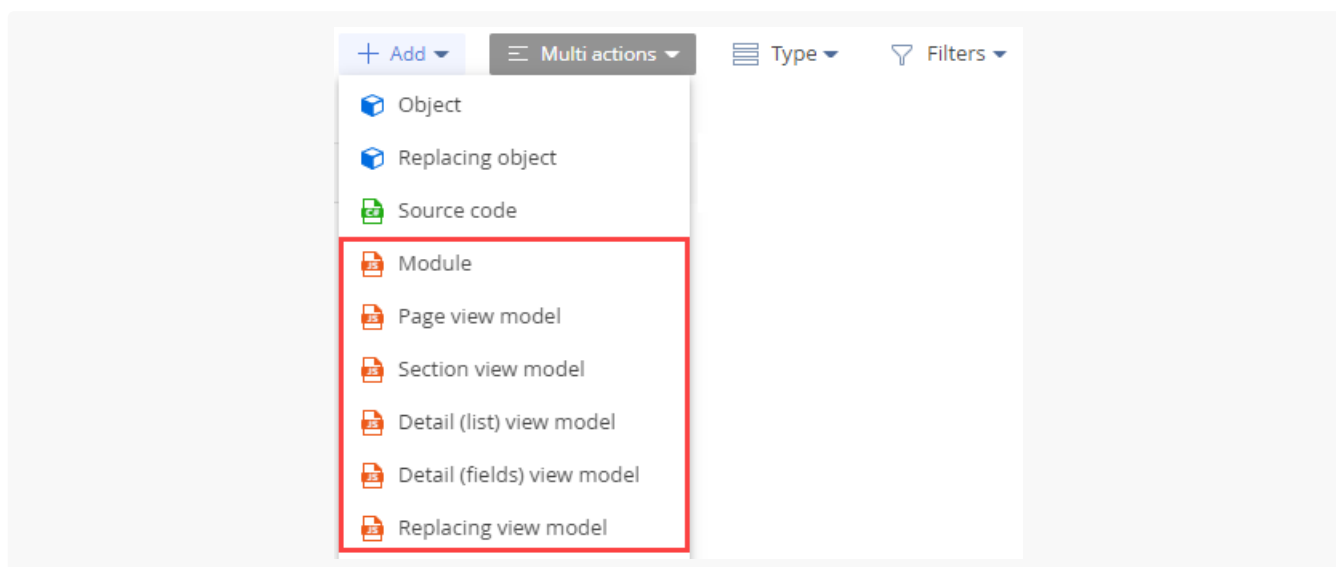
В результате колонка будет содержать значения на разных языках. В интерфейсе приложения будет использоваться значение, которое зависит от выбранного в профиле пользователя языка.

Добавить локализуемую строку

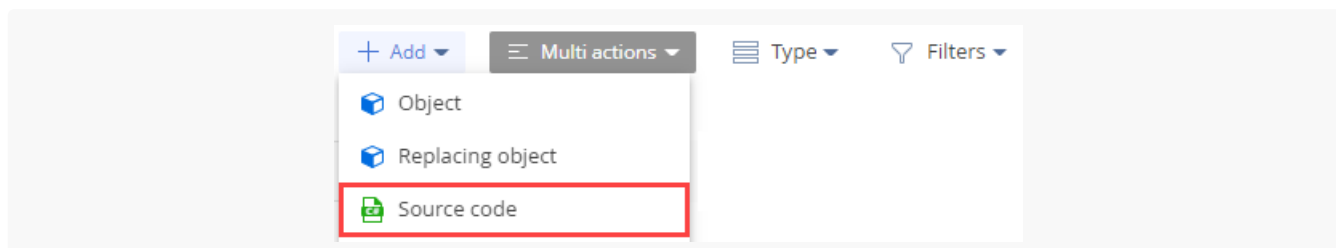
Назначение локализуемой строки — возможность отображения в интерфейсе приложения данных модуля на нескольких языках (т. н. мультязычие). Значение локализуемой строки зависит от выбранного в профиле пользователя языка. Локализуемая строка настраивается в дизайнера модуля и в дизайнера исходного кода, а ее значение хранится в базе данных.

Чтобы **добавить локализуемую строку**:

1. [Перейдите в раздел \[Конфигурация \]](#) ([*Configuration*]) и выберите пользовательский [пакет](#), в который будет добавлена схема.
2. На панели инструментов реестра раздела нажмите [*Добавить*] ([*Add*]) и выберите вид схемы.
 - Если необходимо **локализовать строку клиентского модуля**, то выберите один из вариантов:
 - [*Модуль*] ([*Module*]).
 - [*Модель представления страницы*] ([*Page view model*]).
 - [*Модель представления раздела*] ([*Section view model*]).
 - [*Модель представления детали с реестром*] ([*Detail (list) view model*]).
 - [*Модель представления детали с полями*] ([*Detail (fields) view model*]).
 - [*Замещающая модель представления*] ([*Replacing view model*]).



- Если необходимо **локализовать строку исходного кода**, то выберите [*Исходный код*] ([*Source code*]).



3. В дизайнера заполните свойства схемы.

4. Добавьте строку, которую требуется локализовать.

- a. В контекстном меню узла [*Локализуемые строки*] ([*Localizable strings*]) нажмите кнопку **+**.
- b. Заполните свойства локализуемой строки:
 - [*Код*] ([*Code*]) — название локализуемой строки (обязательное свойство). Должно содержать префикс (по умолчанию `Usr`), указанный в системной настройке [*Префикс названия объекта*] (код [*SchemaNamePrefix*]).
 - [*Значение*] ([*Value*]) — значение локализуемой строки. По умолчанию введенное значение сохраняется для выбранного в профиле пользователя языка. Чтобы задать значения локализуемой строки на других языках, нажмите кнопку **⌘A**.

- e. Для добавления локализуемой строки нажмите [*Добавить*] ([*Add*]).

5. На панели инструментов дизайнера нажмите [*Сохранить*] ([*Save*]), а затем [*Опубликовать*] ([*Publish*]).

В результате строка будет содержать значения на разных языках. В интерфейсе приложения будет использоваться значение, которое зависит от выбранного в профиле пользователя языка.

Использовать базу данных для выполнения операций с локализуемыми ресурсами

Использование базы данных позволяет выполнять следующие **операции с локализуемыми ресурсами**:

- Получить локализуемые ресурсы из базы данных.
- Обновить локализуемые ресурсы в базе данных.
- Сохранить локализуемые ресурсы в базе данных.
- Отключить локализуемые ресурсы в базе данных.

Получить локализуемые ресурсы из базы данных

Классы, которые реализуют логику получения локализуемых ресурсов из базы данных:

- `Terrasoft.Core.Entities.EntitySchemaQuery` — получение данных из базы данных через ORM-модель. Класс по умолчанию поддерживает работу с мультиязычными данными.
- `Terrasoft.Core.Entities.Entity` — работа с сущностью базы данных.

Подробнее о получении данных из базы данных с помощью классов

`Terrasoft.Core.Entities.EntitySchemaQuery` и `Terrasoft.Core.Entities.Entity` читайте в статье [Доступ к данным через ORM](#).

Правила формирования выборки мультиязычных данных:

- Если в профиле пользователя **выбран основной язык**, то выборка данных формируется из основной таблицы базы данных.
- Если в профиле пользователя **выбран дополнительный язык** и в таблице `[SysLocalizableValue]` **присутствует значение локализуемого ресурса**, то выборка данных формируется из таблицы `[SysLocalizableValue]` базы данных.
- Если в профиле пользователя **выбран дополнительный язык** и в таблице `[SysLocalizableValue]` **отсутствует значение локализуемого ресурса**, то выборка данных формируется из основной таблицы базы данных.

Для выборки данных можно использовать **представления** (`View`), которые позволяют получить данные из локализуемых колонок. Для формирования выборки локализуемых данных через представление необходимо настроить локализуемые представления.

Чтобы **настроить локализуемые представления**:

1. Создайте схему объекта для представления. Шаблон названия схемы: `UsrVwИмяСхемыОбъекта`.

Обязательные составляющие названия схемы:

- Префикс (по умолчанию `Usr`), указанный в системной настройке [*Префикс названия объекта*] (код [`SchemaNamePrefix`]).
 - Префикс `vw` (сокращение от `view`) который указывает, что схема является представлением в базе данных.
2. Настройте локализуемую колонку. Настройка локализуемых колонок описана в пункте [Добавить локализуемую колонку](#).
 3. Добавьте новое представление локализации в базе данных.

Обновить локализуемые ресурсы в базе данных

При изменении значения локализуемого ресурса в таблице `[SysLocalizableValue]` необходимо выполнить обновление локализуемых ресурсов в базе данных.

Чтобы для соответствующей схемы **обновить локализуемые ресурсы в базе данных**, необходимо с помощью SQL-запроса изменить значение колонки `[IsChanged]` таблицы `[SysPackageResourceChecksum]`. В другом случае при обновлении пакета в приложении возникнет конфликт локализуемых ресурсов. Он не будет обнаружен, что приведет к потере значения локализуемого ресурса.

Важно. Запрещено добавлять данные в таблицу `[SysLocalizableValue]` с использованием прямого SQL-запроса, поскольку в метаданных соответствующей схемы будет отсутствовать информация о

добавленных локализуемых ресурсах. Чтобы **добавить локализуемые ресурсы**, необходимо использовать Creatio IDE, систему контроля версий SVN или файловую систему.

Сохранить локализуемые ресурсы в базе данных

Чтобы **сохранить локализуемые ресурсы**, необходимо использовать метод `Entity.SetColumnValue()`, который может принимать параметры типа `string` и параметры типа `LocalizableString`.

Сохранить локализуемые ресурсы с использованием параметров типа `string`

Особенности сохранения локализуемых ресурсов при использовании параметров типа `string`:

- Если **запись добавляется** пользователем с дополнительным языком, то данные сохраняются и в основную таблицу объекта `Entity`, и в таблицу локализации объекта `Entity`.
- Если **запись изменяется** пользователем с дополнительным языком, то данные сохраняются в таблицу локализации объекта `Entity`.
- Если **запись добавляется или изменяется** пользователем с основным языком, то данные сохраняются в основную таблицу объекта `Entity`.

Шаблон формирования имени таблицы локализации: `[SysИмяОсновнойТаблицыLcz]`.

Ниже приведен пример сохранения локализуемых ресурсов с использованием параметра типа `string` для пользователя с основным языком (русский).

Пример сохранения локализуемых ресурсов с использованием параметра типа `string`

```
var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];
EntitySchema schema = userConnection.EntitySchemaManager.FindInstanceByName("AccountType");
Entity entity = schema.CreateEntity(userConnection);

/* Установка для колонок значений по умолчанию. */
entity.SetDefColumnValues();

/* Установка значения для колонки [Название]. */
entity.SetColumnValue("Name", "Новый клиент");

/* Сохранение. */
entity.Save();
var name = String.Format("Name: {0}", entity.GetTypedColumnValue<string>("Name"));
return name;
```

При выполнении этого кода пользователем с основным языком (русский) будет выполнен SQL-запрос к основной таблице `[AccountType]` базы данных. В параметре `@P2` указано значение "Новый клиент".

SQL-запрос


```
exec sp_executesql N'
INSERT INTO [dbo].[AccountType]([Id], [Name], [CreatedOn], [CreatedById], [ModifiedOn], [Modifie
VALUES(@P1, @P2, @P3, @P4, @P5, @P6, @P7, @P8)',N'@P1 uniqueidentifier,@P2 nvarchar(12),@P3 date
```

Ниже приведен пример сохранения локализуемых ресурсов с использованием параметра типа `string` для пользователя с дополнительным языком (например, английским).

Пример сохранения локализуемых ресурсов с использованием параметра типа `string`

```
var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];
EntitySchema schema = userConnection.EntitySchemaManager.FindInstanceByName("AccountType");
Entity entity = schema.CreateEntity(userConnection);
entity.SetDefColumnValues();
entity.SetColumnValue("Name", "New Customer");
entity.Save();
var name = String.Format("Name: {0}", entity.GetTypedColumnValue<string>("Name"));
return name;
```

При выполнении этого кода пользователем с дополнительным языком (например, английским) будут выполнены:

1. SQL-запрос к основной таблице `[AccountType]` базы данных. SQL-запрос такой же, как и для основной локализации, но в параметре `@P2` указано значение "New Customer".

SQL-запрос

```
exec sp_executesql N'
INSERT INTO [dbo].[AccountType]([Id], [Name], [CreatedOn], [CreatedById], [ModifiedOn], [Modi
VALUES(@P1, @P2, @P3, @P4, @P5, @P6, @P7, @P8)',N'@P1 uniqueidentifier,@P2 nvarchar(12),@P3 d
```

2. SQL-запрос в таблицу локализации `[SysAccountTypeLcz]`. В параметре `@P5` указано значение "New Customer".

SQL-запрос

```
exec sp_executesql N'
INSERT INTO [dbo].[SysAccountTypeLcz]([Id], [ModifiedOn], [RecordId], [SysCultureId], [Name])
VALUES(@P1, @P2, @P3, @P4, @P5)',N'@P1 uniqueidentifier,@P2 datetime2(7),@P3 uniqueidentifier
```

Таким образом, в основную таблицу `[AccountType]` будет помещено значение, которое не соответствует основной локализации. Чтобы этого избежать, необходимо выполнять сохранение локализуемых ресурсов с использованием параметров типа `LocalizableString`.

Сохранить локализуемые ресурсы с использованием параметров типа `LocalizableString`

Ниже приведен пример сохранения данных с использованием параметра типа `LocalizableString`.

Пример сохранения локализуемых ресурсов с использованием параметра типа `LocalizableString`

```
var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];
EntitySchema schema = userConnection.EntitySchemaManager.FindInstanceByName("AccountType");
Entity entity = schema.CreateEntity(userConnection);
entity.SetDefColumnValues();

/* Создание локализуемой строки с локализованными значениями для разных языковых культур. */
var localizableString = new LocalizableString();
localizableString.SetCultureValue(new CultureInfo("ru-RU"), "Новый клиент ru-RU");
localizableString.SetCultureValue(new CultureInfo("en-US"), "New Customer en-US");

/* Установка значения колонки с помощью локализуемой строки. */
entity.SetColumnValue("Name", localizableString);
entity.Save();

/* Результат будет выведен в текущей языковой культуре пользователя. */
var name = String.Format("Name: {0}", entity.GetTypedColumnValue<string>("Name"));
return name;
```

При выполнении этого кода, независимо от выбранного в профиле пользователя языка, будут выполнены:

1. SQL-запрос в основную таблицу `[AccountType]`. В параметре `@P2` указано значение "Новый клиент ru-RU".

SQL-запрос

```
exec sp_executesql N'
INSERT INTO [dbo].[AccountType]([Id], [Name], [CreatedOn], [CreatedById], [ModifiedOn], [ModifiedById], [RecordId], [SysCultureId], [Name])
VALUES(@P1, @P2, @P3, @P4, @P5, @P6, @P7, @P8)',N'@P1 uniqueidentifier,@P2 nvarchar(18),@P3 d
```

2. SQL-запрос в таблицу локализации `[SysAccountTypeLcz]`. В параметре `@P5` указано значение "New Customer en-US".

SQL-запрос

```
exec sp_executesql N'
INSERT INTO [dbo].[SysAccountTypeLcz]([Id], [ModifiedOn], [RecordId], [SysCultureId], [Name])
VALUES(@P1, @P2, @P3, @P4, @P5)',N'@P1 uniqueidentifier,@P2 datetime2(7),@P3 uniqueidentifier
```

При выполнении этого кода пользователем с дополнительным языком и отсутствии значения локализуемой строки на основном языке, в основную таблицу [AccountType] будет добавлена запись со значением для дополнительного языка.

Отключить локализуемые ресурсы в базе данных

Чтобы **отключить локализуемые ресурсы**, необходимо для свойства UseLocalization экземпляра класса EntitySchemaQuery установить значение false. Отключение локализуемых ресурсов не приводит к удалению локализуемых ресурсов из таблиц базы данных и не зависит от выбранного в профиле пользователя языка.

Пример отключения получения локализуемых ресурсов

```
/* Пользовательское подключение. */
var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];

/* Формирование запроса. */
var esqResult = new EntitySchemaQuery(userConnection.EntitySchemaManager, "City");

/* Добавление колонки в запрос. */
esqResult.AddColumn("Name");

/* Отключение механизма выборки локализованных данных. */
esqResult.UseLocalization = false;

/* Выполнение запроса к базе данных и получение всей результирующей коллекции объектов. */
var entities = esqResult.GetEntityCollection(userConnection);

/* Получение текста запроса. */
var s = esqResult.GetSelectQuery(userConnection).GetSqlText();

/* Возврат результата. */
return s;
```

При выполнении этого кода, независимо от выбранного в профиле пользователя языка, будет выполнен SQL-запрос к основной таблице [City] базы данных.

SQL-запрос

```
SELECT
    [City].[Name] [Name]
FROM
    [dbo].[City] [City] WITH(NOLOCK)
```

Использовать систему контроля версий SVN для выполнения операций с локализуемыми ресурсами

Использование системы контроля версий SVN позволяет выполнять следующие **операции с локализуемыми ресурсами**:

- Обновить локализуемые ресурсы из хранилища SVN.
- Фиксировать локализуемые ресурсы в хранилище SVN.

Описание работы с SVN содержится в статье [Контроль версий в Subversion](#).

Обновить локализуемые ресурсы из хранилища SVN

Чтобы **обновить локализуемые ресурсы из хранилища SVN**, необходимо выполнить обновление пакета из хранилища SVN. Обновление пакета подробно описано в статье [Контроль версий в Creatio IDE](#).

Changes

Name	Type	Status
▼ sdkCustomPackage	Package	Modified
UsrClientUnitSchema	Schema Resource	Modified
UsrClientUnitSchema	Schema Resource	Modified
UsrClientUnitSchema	Schema Resource	Modified
UsrClientUnitSchema	Schema Resource	Modified
UsrEntitySchema	Schema Resource	Modified
UsrEntitySchema	Schema Resource	Modified

CLOSE

Возможные **состояния локализуемых ресурсов**:

- [Изменено] ([Modified]) — локализуемый ресурс изменен.
- [Добавлено] ([Added]) — локализуемый ресурс добавлен.
- [Удалено] ([Deleted]) — локализуемый ресурс удален.
- [Конфликт] ([Conflicted]) — работа с локализуемым ресурсом выполнялась одновременно двумя разработчиками, один с которых зафиксировал изменения локализуемого ресурса в хранилище SVN.

Фиксировать локализуемые ресурсы в хранилище SVN

Чтобы **фиксировать локализуемые ресурсы в хранилище SVN**, необходимо выполнить фиксацию пакета в хранилище SVN. Фиксация пакета подробно описана в статье [Контроль версий в Creatio IDE](#).

Commit package to repository

Description *

Package changed

Name	Type	Status
▼ sdkCustomPackage (6)	Package	Modified
UsrClientUnitSchema	Schema	Modified
UsrSourceCode	Schema	Added
ExternalAssembly.dll	External Assembly	Deleted
UsrClientUnitSchema	Schema Resource	Modified
UsrSourceCode	Schema Resource	Added
UsrClientUnitSchema	Schema Resource	Modified

CANCEL
COMMIT CHANGES

Используя Creatio IDE, можно заблокировать пакет в хранилище SVN. Блокирование пакета позволяет избежать возникновения состояния [Конфликт] ([*Conflicted*]) локализуемого ресурса, поскольку отключена опция одновременной работы с пакетом. Например, один разработчик вносит изменения в локализуемые ресурсы пакета без его предварительной блокировки. При фиксации пакета в хранилище SVN может возникнуть ситуация, когда в текущем пакете другим разработчиком были изменены и зафиксированы в хранилище SVN эти же локализуемые ресурсы. В этом случае при обновлении пакета в Creatio IDE будет отображен перечень локализуемых ресурсов в состоянии [Конфликт] ([*Conflicted*]). Это означает, что версия и содержимое измененных локализуемых ресурсов не совпадают с версией и содержимым локализуемых ресурсов, зафиксированных в хранилище SVN. При выполнении фиксации изменений локализуемых ресурсов будут утеряны изменения, зафиксированные в хранилище SVN. Такие конфликтные ситуации необходимо решать вручную, используя SVN-клиенты, например, [TortoiseSVN](#).

Changes

Name	Type	Status
▼ sdkCustomPackage (1)	Package	Modified
UsrClientUnitSchema	Schema Resource	Conflicted

CLOSE

Использовать файловую систему для выполнения операций с локализуемыми ресурсами

Использование файловой системы позволяет редактировать локализуемые ресурсы.

Чтобы **редактировать локализуемые ресурсы из файловой системы**:

1. Настройте Creatio для работы в файловой системе. Подробнее читайте в статье [Внешние IDE](#).
2. Используя SVN-клиент, экспортируйте локализуемые ресурсы в файловую систему.
3. Измените локализуемые ресурсы.
4. Зафиксируйте изменения в хранилище SVN.

Важно. Значению локализуемого ресурса соответствует одна запись в таблице [SysLocalizableValue] базы данных. Эта запись содержит ссылки на соответствующие идентификаторы ([SysPackageId] , [SysSchemaId] , [SysCultureId]) и ключ ([key]). Поэтому при фиксации ресурса с состоянием [Конфликт] ([Conflicted]) в таблицу будет записано последнее значение.

Получить локализуемые ресурсы из

базы данных



Пример 1

Пример. С помощью класса `Terrasoft.Core.Entities.EntitySchemaQuery` получить локализуемые ресурсы для произвольной языковой культуры.

Класс `Terrasoft.Core.Entities.EntitySchemaQuery` позволяет выполнить выборку данных для произвольной языковой культуры (языковой культуры, которая отличается от дополнительной языковой культуры пользователя и основной языковой культуры приложения).

Чтобы **получить локализуемые ресурсы для произвольной языковой культуры**:

1. Перед получением данных в экземпляре класса `Terrasoft.Core.Entities.EntitySchemaQuery` вызовите метод `SetLocalizationCultureId(Guid cultureId)`.
2. В метод `SetLocalizationCultureId(Guid cultureId)` передайте идентификатор языковой культуры, данные которой необходимо получить.

Выполнение выборки данных для произвольной языковой культуры

```
/* Пользовательское подключение. */
var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];

/* Получение Id необходимой языковой культуры, например, итальянской. */
var sysCulture = new SysCulture(userConnection);
if (!sysCulture.FetchPrimaryInfoFromDB("Name", "it-IT"))
{
    /* Ошибка, запись не найдена. */
    return "Культура не найдена";
}
Guid italianCultureId = sysCulture.Id;

/* Формирование запроса. */
var esqResult = new EntitySchemaQuery(userConnection.EntitySchemaManager, "City");

/* Добавление колонки в запрос. */
esqResult.AddColumn("Name");

/* Установка необходимой локализации. */
esqResult.SetLocalizationCultureId(italianCultureId);

/* Выполнение запроса к базе данных и получение всей результирующей коллекции объектов. */
var entities = esqResult.GetEntityCollection(userConnection);
```

```

/* Получение текста запроса. */
var s = esqResult.GetSelectQuery(userConnection).GetSqlText();

/* Возврат результата. */
return s;

```

При выполнении этого кода будет выполнен SQL-запрос. Параметр `@P1` принимает значение идентификатора записи (`[Id]`), которое хранится в переменной `italianCultureId`.

SQL-запрос

```

SELECT
    ISNULL([SysCityLcz].[Name], [City].[Name])[Name]
FROM
    [dbo].[City] [City] WITH(NOLOCK)
    LEFT OUTER JOIN [dbo].[SysCityLcz] [SysCityLcz] WITH(NOLOCK) ON ([SysCityLcz].[RecordId] = [
    AND [SysCityLcz].[SysCultureId] = @P1)

```

Пример 2

Пример. С помощью метода `FetchFromDB()` класса `Terrasoft.Core.Entities.Entity` получить значение локализуемой колонки `[Название] ([Name])` объекта схемы `[Тип контрагента] ([AccountType])`.

Методы `FetchFromDB()` класса `Terrasoft.Core.Entities.Entity` позволяют получить мультиязычные данные. Ниже приведен пример использования одной из перегрузок методов `FetchFromDB()` для пользователя с основной (русской) и дополнительной (например, английской) языковыми культурами. Эти методы можно использовать, например, в методах [пользовательского веб-сервиса](#).

Пример использования метода `FetchFromDB()`

```

/* Пользовательское подключение. */
var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];

/* Получение схемы [Тип контрагента]. */
EntitySchema schema = userConnection.EntitySchemaManager.FindInstanceByName("AccountType");

/* Создание экземпляра Entity (объекта). */
Entity entity = schema.CreateEntity(userConnection);

/* Коллекция имен колонок для выборки. */
List<string> columnNamesToFetch = new List<string> {
    "Name",

```



```

        "Description"
    };
    /* Получение данных для объекта, у которого значение колонки [Название] равно "Клиент". */
    entity.FetchFromDB("Name", "Клиент", columnNamesToFetch);

    /* Формирование и отправка ответа. */
    var name = String.Format("Name: {0}", entity.GetTypedColumnValue<string>("Name"));
    return name;

```

При выполнении этого кода пользователем с основной языковой культурой (русской) будет выполнен SQL-запрос к основной таблице `[AccountType]` базы данных. В параметре `@P1` указано значение "Клиент", которое определяет соответствующую запись основной таблицы `[AccountType]` базы данных.

SQL-запрос

```

exec sp_executesql N'
SELECT
    [AccountType].[Name] [Name],
    [AccountType].[Description] [Description]
FROM
    [dbo].[AccountType] [AccountType] WITH(NOLOCK)
WHERE
    [AccountType].[Name] = @P1', N'@P1 nvarchar(6)', @P1=N'Клиент'

```

При выполнении этого кода пользователем с дополнительной языковой культурой (например, английской) будет выполнен SQL-запрос к таблице локализации `[SysAccountTypeLcz]` базы данных. В параметре `@P1` указано значение "Клиент", которое определяет соответствующую запись основной таблицы `[AccountType]` базы данных. В параметре `@P2` указано значение идентификатора (`[SysCultureId]`) дополнительной языковой культуры из таблицы `[SysCulture]`, которое определяет соответствующую запись таблицы локализации `[SysAccountTypeLcz]` базы данных.

SQL-запрос

```

exec sp_executesql N'
SELECT
    ISNULL([SysAccountTypeLcz].[Name], [AccountType].[Name]) [Name],
    ISNULL([SysAccountTypeLcz].[Description], [AccountType].[Description]) [Description]
FROM
    [dbo].[AccountType] [AccountType] WITH(NOLOCK)
    LEFT OUTER JOIN [dbo].[SysAccountTypeLcz] [SysAccountTypeLcz] WITH(NOLOCK) ON ([SysAccountTypeLcz].[AccountType] = [AccountType].[Name]
    AND [SysAccountTypeLcz].[SysCultureId] = @P2)
WHERE
    [AccountType].[Name] = @P1', N'@P1 nvarchar(6),@P2 uniqueidentifier', @P1=N'Клиент', @P2='A5426

```

В результате значению переменной `Name` для пользователя с русской языковой культурой соответствует значение "Клиент", а для пользователя с английской языковой культурой — "Customer".

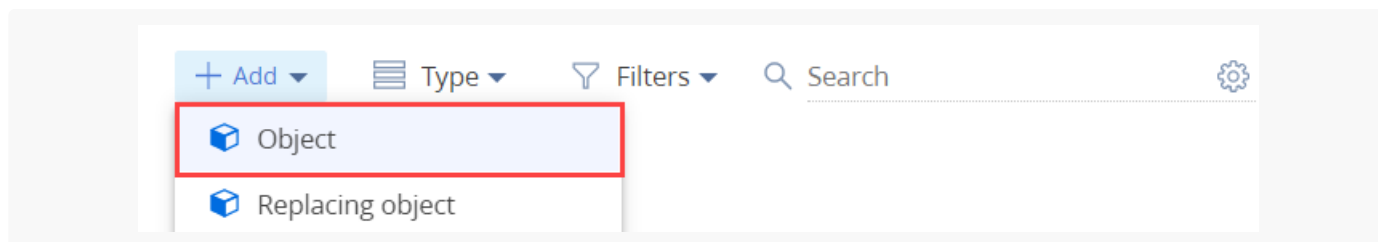
Добавить локализуемую колонку

 Средний

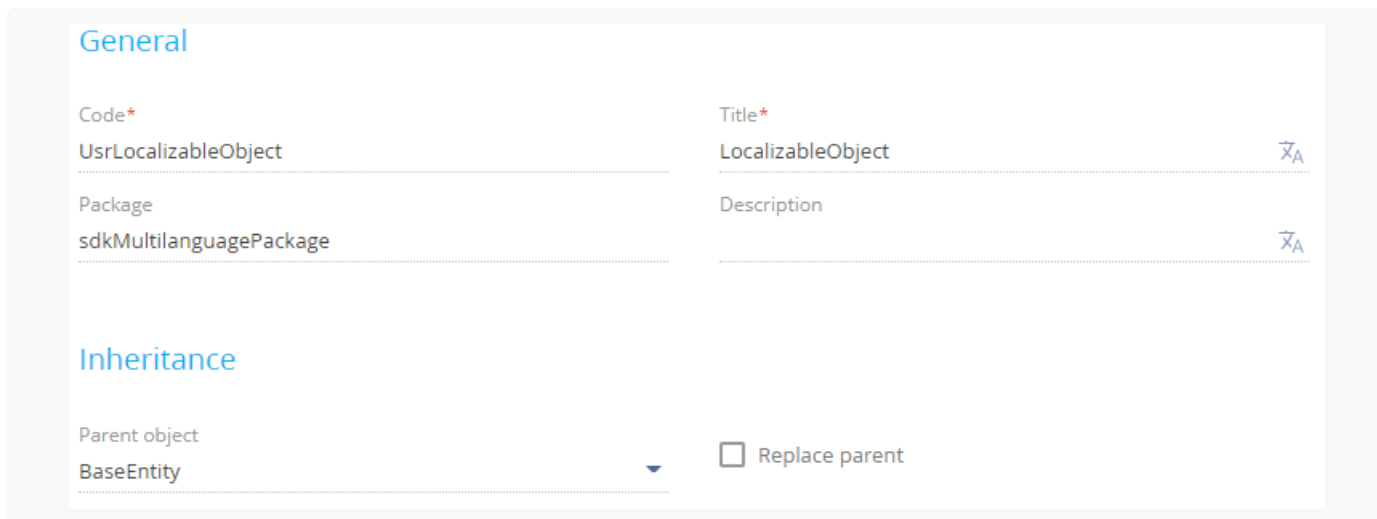
Пример. Добавить локализуемую колонку.

1. Создать объект

1. [Перейдите в раздел \[Конфигурация \]](#) ([*Configuration*]) и выберите пользовательский [пакет](#), в который будет добавлена схема.
2. На панели инструментов реестра раздела нажмите [*Добавить*] —> [*Объект*] ([*Add*] —> [*Object*]).

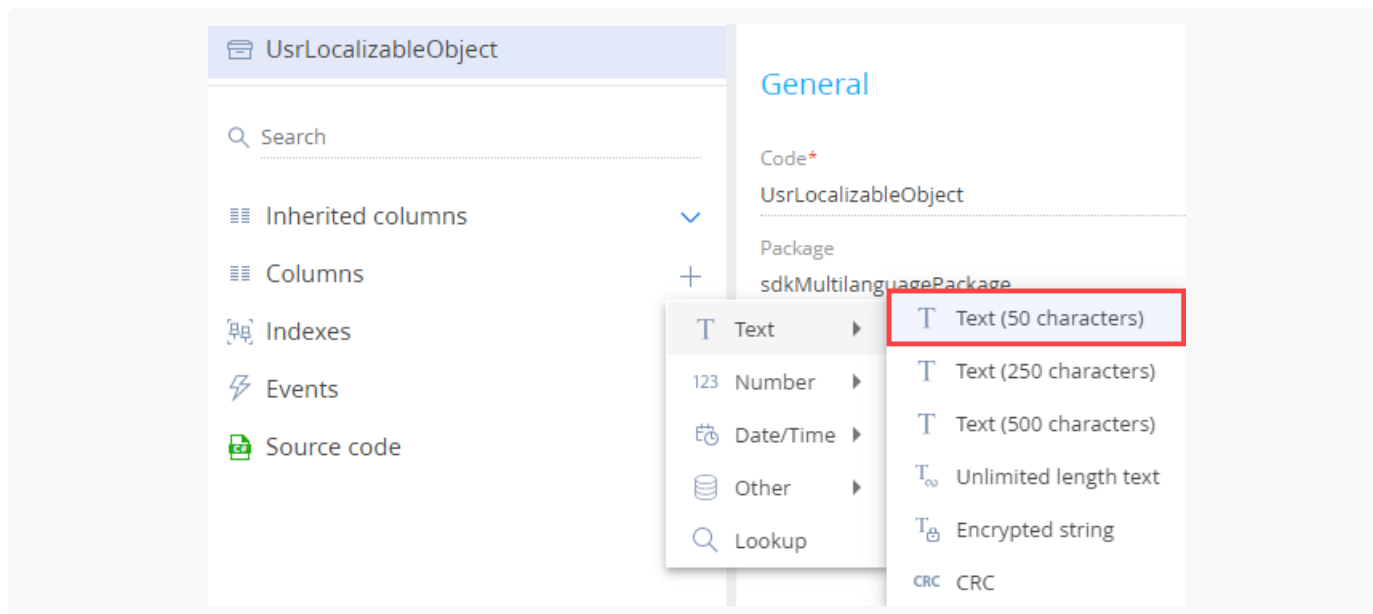


3. В дизайнера объекта заполните свойства схемы:
 - [*Код*] ([*Code*]) — "UsrLocalizableObject".
 - [*Заголовок*] ([*Title*]) — "LocalizableObject".
 - [*Родительский объект*] ([*Parent object*]) — выберите "BaseEntity".

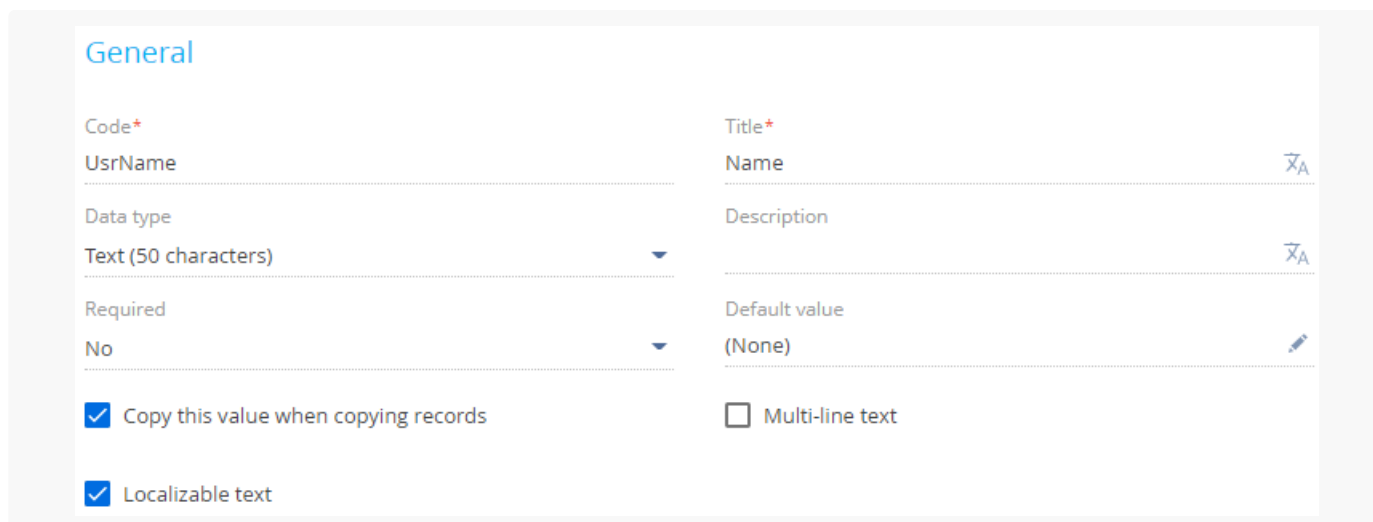

 A screenshot of the 'Object Designer' interface. It shows two tabs: 'General' and 'Inheritance'.
 In the 'General' tab, there are two input fields: 'Code*' with the value 'UsrLocalizableObject' and 'Title*' with the value 'LocalizableObject'. Below them are 'Package' (value: 'sdkMultilanguagePackage') and 'Description' (empty).
 In the 'Inheritance' tab, there is a 'Parent object' dropdown menu with 'BaseEntity' selected, and a checkbox labeled 'Replace parent' which is currently unchecked.

2. Добавить локализуемую колонку

1. В контекстном меню узла [Колонки] ([Columns]) структуры объекта нажмите **+**.
2. В выпадающем меню нажмите [Строка] —> [Строка (50 символов)] ([Text] —> [Text (50 characters)]).



3. В дизайнера объекта заполните свойства добавляемой колонки:
 - [Код] ([Code]) — "UsrName".
 - [Заголовок] ([Title]) — "Name".
 - Установите признак [Локализуемый текст] ([Localizable text]).



4. На панели инструментов дизайнера объектов нажмите [Сохранить] ([Save]), а затем [Опубликовать] ([Publish]).

Результат выполнения примера

В результате выполнения примера в базе данных создана таблица локализации

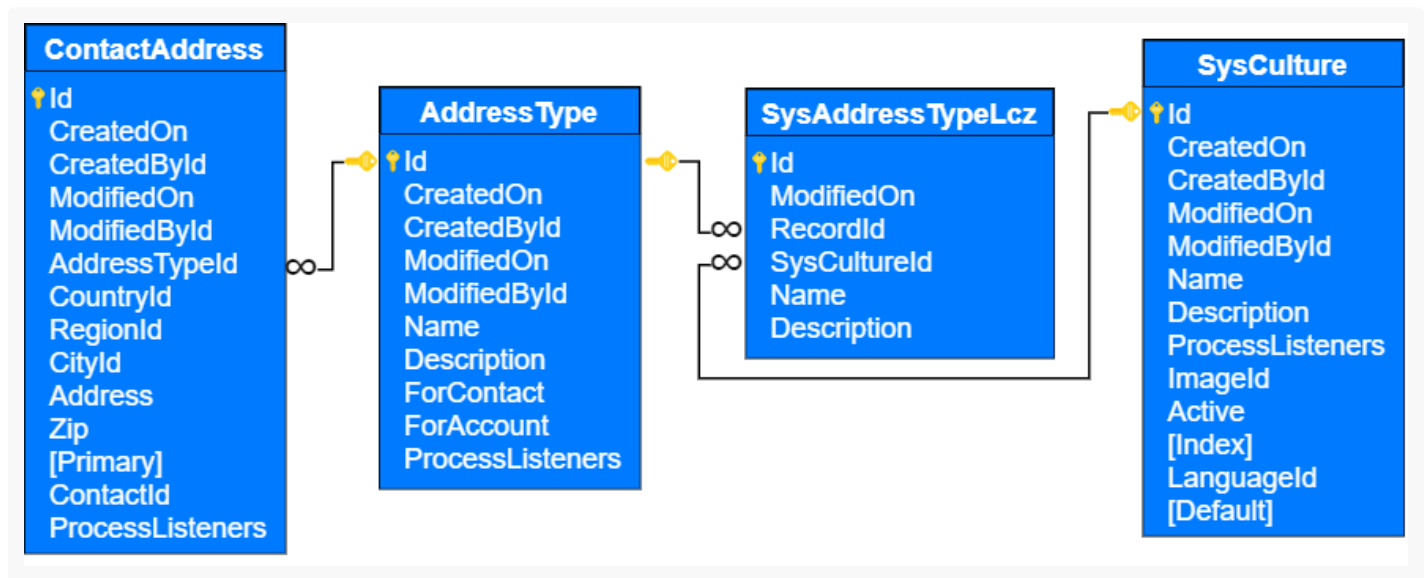
[SysUsrLocalizableObjectLcz] , в которой будут храниться локализованные данные для всех локализуемых колонок.

Локализовать представление в базе данных

 Сложный

Пример. Создать представление, которое формирует выборку, состоящую из локализованных значений адресов контактов (колонок [Name] таблицы [AddressType] базы данных) и значений адресов контактов (колонок [Address] таблицы [ContactAddress] базы данных).

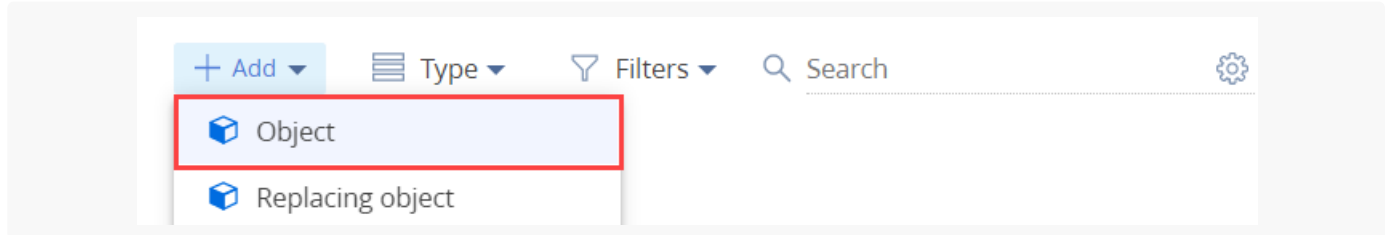
В Creatio реализована схема объекта [Адрес контакта] ([ContactAddress]). Колонка схемы ссылается на справочник [Тип адреса] ([AddressType]), который содержит локализуемую колонку [Название] ([Name]). Структура и связи таблиц представлены на рисунке ниже.



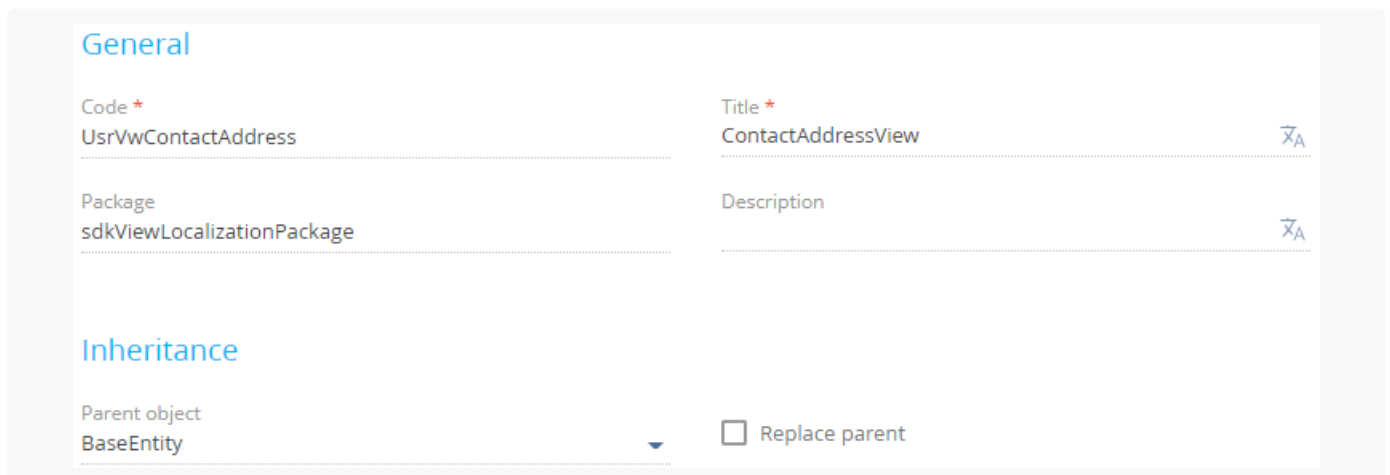
- [ContactAddress] — таблица базы данных, которая содержит перечень значений адресов контактов. Связана с таблицей [AddressType] по колонке [AddressTypeId] .
- [AddressType] — таблица базы данных, которая содержит перечень типов адресов контактов. Колонка [Name] таблицы содержит перечень значений типов адресов на основном языке. Значения на дополнительных языках содержатся в таблице [SysAddressTypeLcz] .
- [SysAddressTypeLcz] — автоматически генерируемая системная таблица базы данных, которая содержит перечень локализованных значений типов адресов контактов. Связана с таблицей [AddressType] по колонке [RecordId] и с таблицей [SysCulture] по колонке [SysCultureId] . Колонка [Name] таблицы содержит перечень локализованных значений типов адресов контактов для языковой культуры, которая указана в колонке [SysCultureId] текущей таблицы.
- [SysCulture] — системная таблица базы данных, которая содержит перечень языковых культур.

1. Создать схему объекта для представления

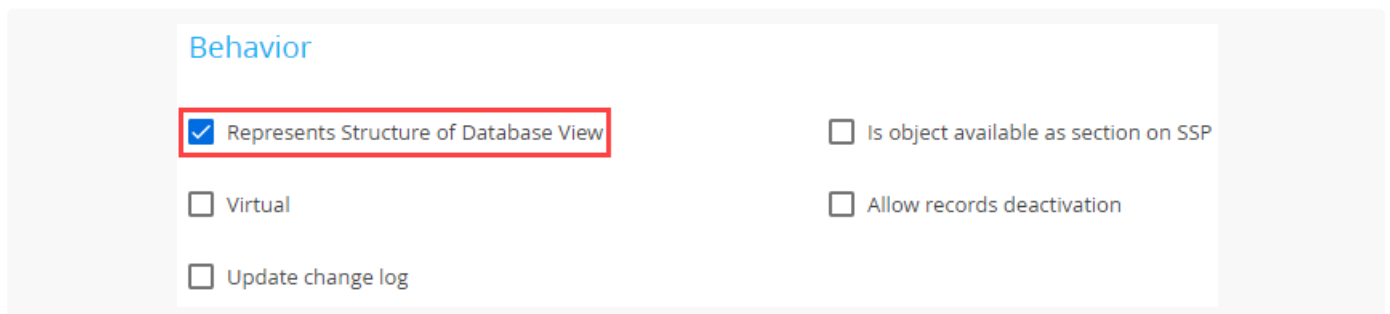
1. [Перейдите в раздел \[Конфигурация \]](#) ([Configuration]) и выберите пользовательский [пакет](#), в который будет добавлена схема.
2. На панели инструментов реестра раздела нажмите [Добавить] —> [Объект] ([Add] —> [Object]).



3. В дизайнере объекта заполните свойства схемы:
 - [Код] ([Code]) — "UsrVwContactAddress".
 - [Заголовок] ([Title]) — "ContactAddressView".
 - [Родительский объект] ([Parent object]) — выберите "BaseEntity".

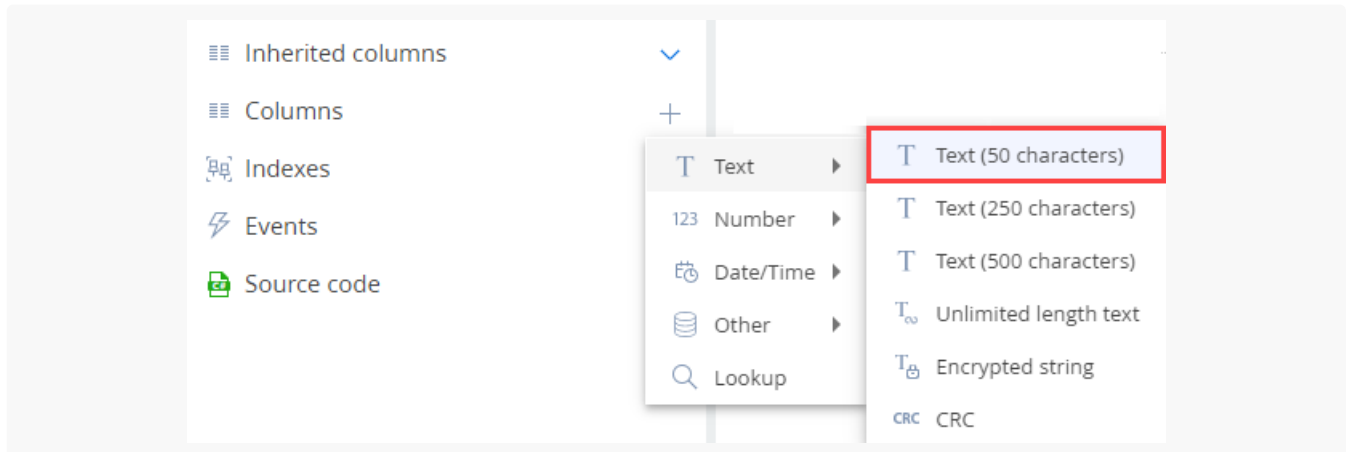


4. В блоке свойств [Поведение] ([Behavior]) установите признак [Представление в базе данных] ([Represent Structure of Database View]).



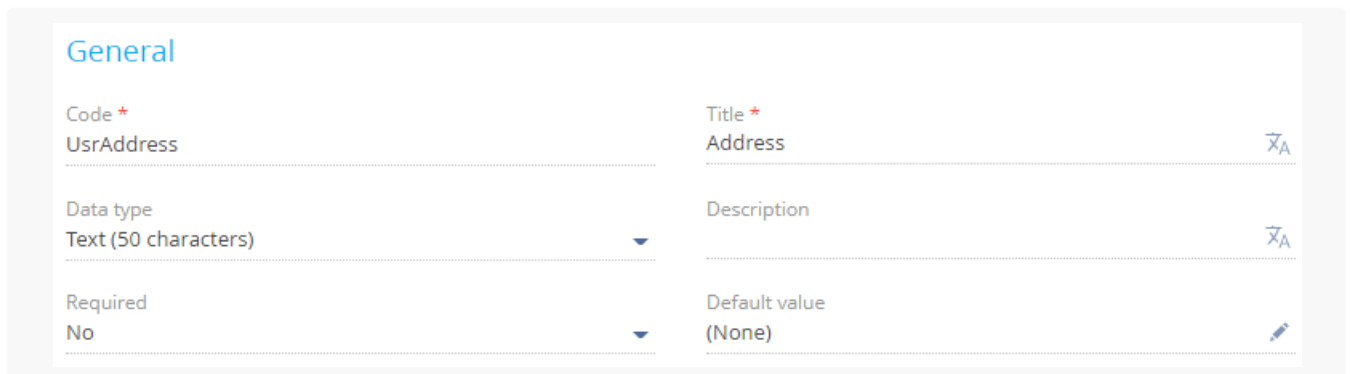
2. Добавить колонки

1. Добавьте колонку, которая будет содержать перечень значений адресов контактов на основном языке.
 - a. В контекстном меню узла [Колонки] ([Columns]) структуры объекта нажмите **+**.
 - b. В выпадающем меню нажмите [Строка] —> [Строка (50 символов)] ([Text] —> [Text (50 characters)]).

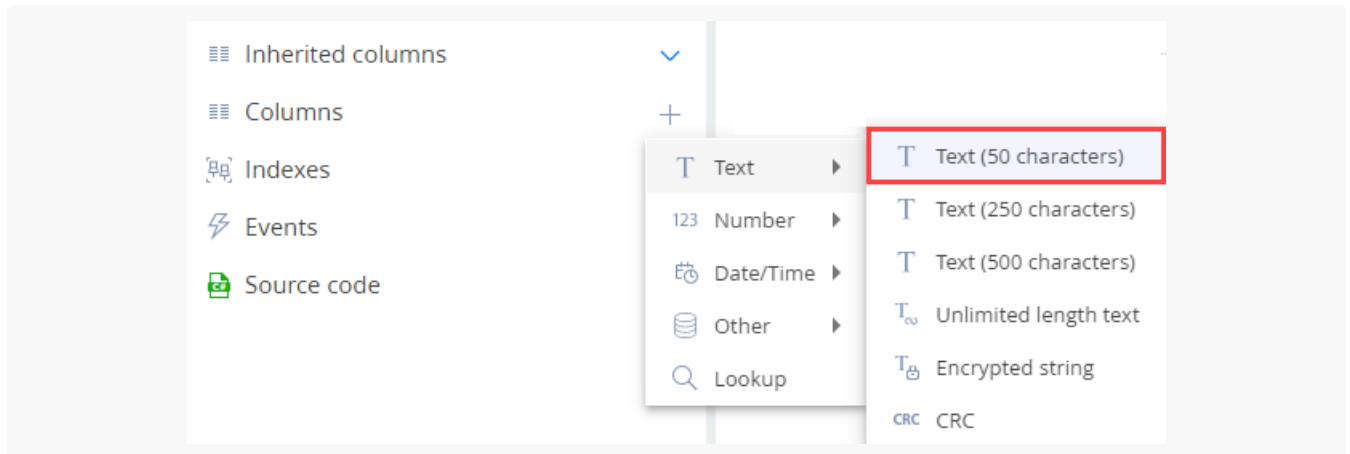


- c. В дизайнере объекта заполните свойства добавляемой колонки:

- [Код] ([Code]) — "UsrAddress".
- [Заголовок] ([Title]) — "Address".



2. Добавьте колонку, которая будет содержать перечень типов адресов контактов на дополнительном языке.
 - a. В контекстном меню узла [Колонки] ([Columns]) структуры объекта нажмите **+**.
 - b. В выпадающем меню нажмите [Строка] —> [Строка (50 символов)] ([Text] —> [Text (50 characters)]).



с. В дизайнера объекта заполните свойства добавляемой колонки:

- [Код] ([Code]) — "UsrAddressType".
- [Заголовок] ([Title]) — "AddressType".
- Установите признак [Локализуемый текст] ([Localizable text]).

г. На панели инструментов дизайнера объектов нажмите [Сохранить] ([Save]), а затем [Опубликовать] ([Publish]).

3. Создать представления в базе данных

1. Создайте представление [UsrVwContactAddress] в базе данных. Для этого выполните SQL-запрос.

SQL-запрос

```
-- Название представления должно соответствовать названию таблицы.
CREATE VIEW dbo.UsrVwContactAddress
AS
SELECT
    ContactAddress.Id,
```

```
-- Колонки представления должны соответствовать названиям колонок схемы.
ContactAddress.Address AS UsrAddress,
AddressType.Name AS UsrAddressType
FROM ContactAddress
INNER JOIN AddressType ON ContactAddress.AddressTypeId = AddressType.Id;
```

2. Создайте локализуемое представление `[SysUsrVwContactAddressLcz]` в базе данных. Для этого выполните SQL-запрос.

SQL-запрос

```
-- Название представления должно соответствовать названию таблицы локализации.
CREATE VIEW dbo.SysUsrVwContactAddressLcz
AS
SELECT
    SysAddressTypeLcz.Id,
    ContactAddress.id AS RecordId,
    SysAddressTypeLcz.SysCultureId,
    -- Колонки представления должны соответствовать названиям колонок схемы.
    SysAddressTypeLcz.Name AS UsrAddressType
FROM ContactAddress
INNER JOIN AddressType ON ContactAddress.AddressTypeId = AddressType.Id
INNER JOIN SysAddressTypeLcz ON AddressType.Id = SysAddressTypeLcz.RecordId;
```

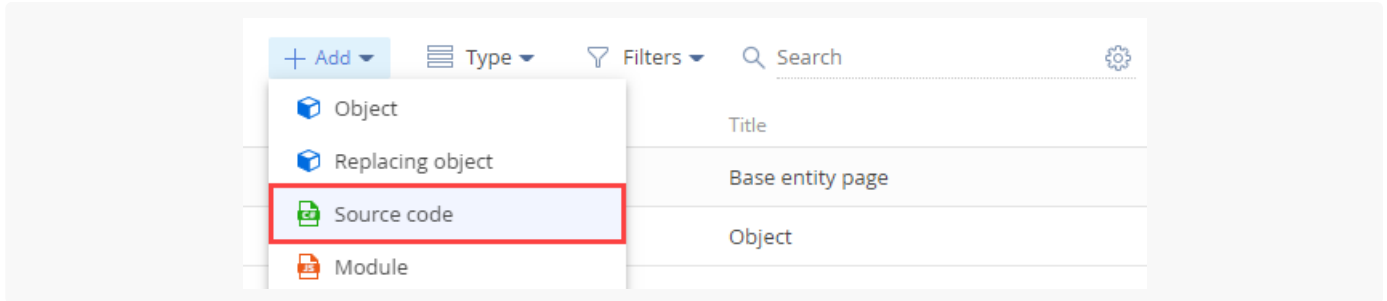
В результате при чтении данных с помощью `EntitySchemaQuery` из колонки `[UsrAddressType]` представления `[UsrVwContactAddress]` будут отображены корректные значения для разных языков.

Результат выполнения примера

Для проверки результата выполнения примера создайте пользовательский веб-сервис с аутентификацией на основе cookies.

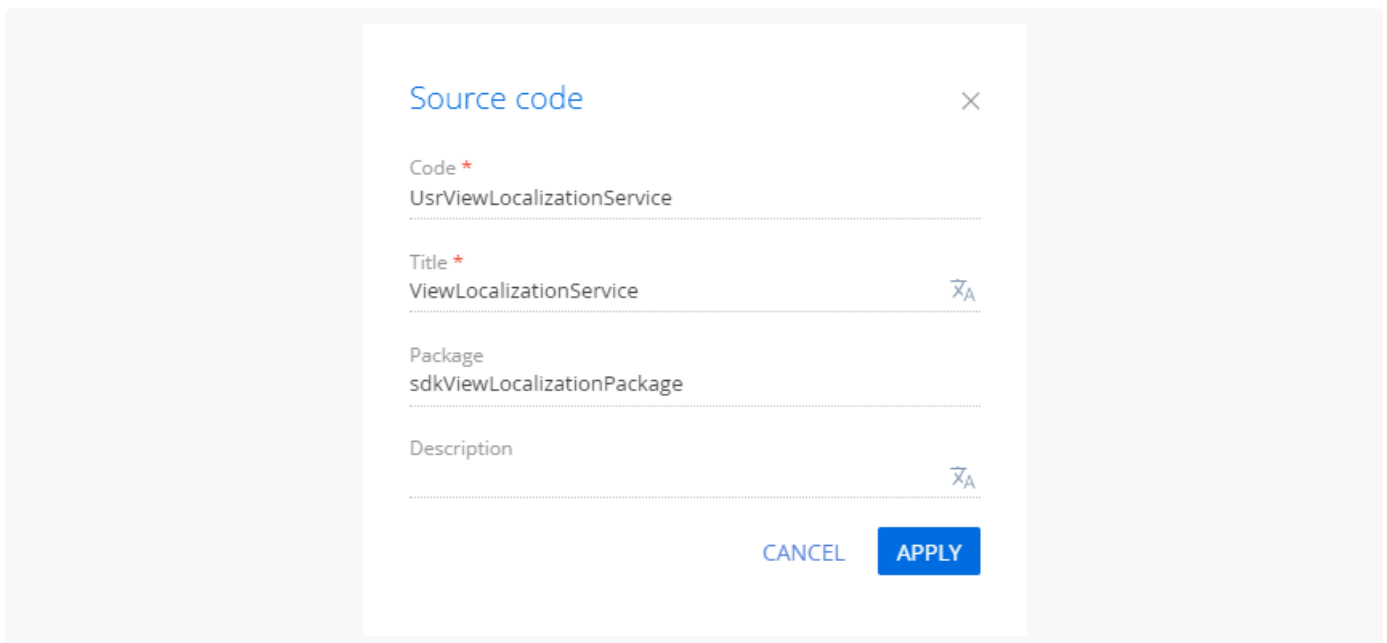
1. Создать схему [*Исходный код*]

1. [Перейдите в раздел \[Конфигурация \]](#) ([*Configuration*]) и выберите пользовательский [пакет](#), в который будет добавлена схема.
2. На панели инструментов реестра раздела нажмите [*Добавить*] —> [*Исходный код*] ([*Add*] —> [*Source code*]).



3. В дизайнере схем заполните свойства схемы:

- [Код] ([Code]) — "UsrViewLocalizationService".
- [Заголовок] ([Title]) — "UsrViewLocalizationService".



Для применения заданных свойств нажмите [Применить] ([Apply]).

2. Создать класс сервиса

1. В дизайнере схем добавьте пространство имен `Terrasoft.Configuration`.
2. С помощью директивы `using` добавьте пространства имен, типы данных которых будут задействованы в классе.
3. Добавьте название класса, которое соответствует названию схемы (свойство [Код] ([Code])).
4. В качестве родительского класса укажите `System.Web.SessionState.IReadOnlySessionState`.
5. Для класса добавьте атрибуты `[ServiceContract]` и `[AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.Required)]`.

3. Реализовать методы класса

1. Реализуйте метод, который вернет перечень типов адресов контактов и значений адресов контактов из созданного нелокализуемого представления `[UsrVwContactAddress]`. В дизайнера схем добавьте в класс метод `public string GetNonLocalizableView()`, который реализует конечную точку пользовательского веб-сервиса. С помощью `EntitySchemaQuery` метод отправит запрос к базе данных.
2. Реализуйте метод, который вернет перечень типов адресов контактов и значений адресов контактов из созданного локализуемого представления `[UsrVwContactAddress]`. В дизайнера схем добавьте в класс метод `public string GetLocalizableView()`, который реализует конечную точку пользовательского веб-сервиса. С помощью `EntitySchemaQuery` метод отправит запрос к базе данных.

Исходный код пользовательского веб-сервиса `UsrViewLocalizationService` представлен ниже.

UsrViewLocalizationService

```
namespace Terrasoft.Configuration
{
    using System.ServiceModel;
    using System.ServiceModel.Web;
    using System.ServiceModel.Activation;
    using System.Web;
    using Terrasoft.Core;
    using Terrasoft.Core.Entities;
    using System;
    using System.Collections.Generic;

    [ServiceContract]
    [AspNetCompatibilityRequirements(RequirementsMode = AspNetCompatibilityRequirementsMode.Required)]
    public class UsrViewLocalizationService : System.Web.SessionState.IReadOnlySessionState
    {
        [OperationContract]
        [WebInvoke(Method = "GET", UriTemplate = "GetNonLocalizableView")]
        public string GetNonLocalizableView()
        {
            var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];
            var esqResult = new EntitySchemaQuery(userConnection.EntitySchemaManager, "UsrVwContactAddress");
            esqResult.AddColumn("UsrAddress");
            esqResult.AddColumn("UsrAddressType");
            var entities = esqResult.GetEntityCollection(userConnection);
            var s = "";
            foreach (var item in entities)
            {
                s += item.GetTypedColumnValue<string>("UsrAddressType") + ": ";
                s += item.GetTypedColumnValue<string>("UsrAddress") + "; ";
            }
            return s;
        }
    }
}
```

```

[OperationContract]
[WebInvoke(Method = "GET", UriTemplate = "GetLocalizableView")]
public string GetLocalizableView()
{
    var userConnection = (UserConnection)HttpContext.Current.Session["UserConnection"];
    var sysCulture = new SysCulture(userConnection);
    if (!sysCulture.FetchPrimaryInfoFromDB("Name", "ru-ru"))
    {
        return "Культура не найдена";
    }
    Guid CultureId = sysCulture.Id;

    var esqResult = new EntitySchemaQuery(userConnection.EntitySchemaManager, "UsrVwCont");
    esqResult.AddColumn("UsrAddress");
    esqResult.AddColumn("UsrAddressType");
    esqResult.SetLocalizationCultureId(CultureId);

    var entities = esqResult.GetEntityCollection(userConnection);
    var s = "";
    foreach (var item in entities)
    {
        s += item.GetTypedColumnValue<string>("UsrAddressType") + ": ";
        s += item.GetTypedColumnValue<string>("UsrAddress") + "; ";
    }
    return s;
}
}
}

```

На панели инструментов дизайнера нажмите [*Сохранить*] ([*Save*]), а затем [*Опубликовать*] ([*Publish*]).

Результат работы пользовательского веб-сервиса

В результате выполнения примера в Creatio появится пользовательский веб-сервис

`UsrViewLocalizationService` с конечными точками `GetLocalizableView` и `GetNonLocalizableView`.

Авторизуйтесь в приложении и из браузера обратитесь к конечной точке `GetLocalizableView` веб-сервиса.

Строка запроса

`http://mycreatio.com/0/rest/UsrViewLocalizationService/GetLocalizableView`

В результате будет получена выборка, которая состоит из локализуемых значений типов адресов контактов и значений адресов контактов.

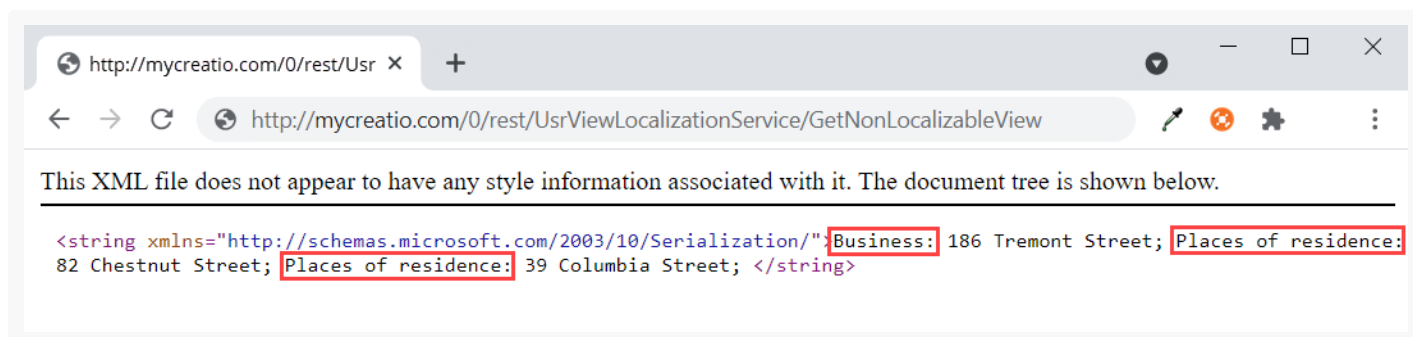


Из браузера обратитесь к конечной точке `GetNonLocalizableView` веб-сервиса.

Строка запроса

`http://mycreatio.com/0/rest/UsrViewLocalizationService/GetNonLocalizableView`

В результате будет получена выборка, которая состоит из нелокализуемых значений типов адресов контактов и значений адресов контактов.



Класс LocalizableValue<T> C#



Пространство имен `Terrasoft.Common`.

Класс `Terrasoft.Common.LocalizableValue<T>` является базовым классом для классов `Terrasoft.Common.LocalizableString` (отвечает за отображение локализуемых строк) и `Terrasoft.Common.LocalizableImage` (отвечает за отображение локализуемых изображений), которые используются для работы с локализуемыми ресурсами.

Класс `Terrasoft.Common.LocalizableValue<T>` является шаблоном для локализуемых значений различных типов и предоставляет методы для работы с ними.

Note. Полный перечень свойств и методов класса `LocalizableValue<T>`, его родительских классов, а также реализуемых им интерфейсов, можно найти в документации [Библиотеки .NET классов](#).

Свойства

Value T

Возвращает и устанавливает локализуемое значение с учетом текущей языковой культуры.

HasValue bool

Возвращает признак, который определяет наличие локализуемого значения данного типа для текущей языковой культуры.

CultureValues IDictionary<CultureInfo,T>

Возвращает справочник локализуемых значений текущего экземпляра для поддерживаемых языковых культур.

Методы

void ClearCultureValue(CultureInfo culture)

Удаляет локализуемое значение для поддерживаемых языковых культур.

Параметры

System.Globalization.CultureInfo culture	Языковая культура.
--	--------------------

T GetCultureValue(CultureInfo culture, bool throwIfNoManager)

Получает локализуемое значение указанного типа для поддерживаемых языковых культур. В зависимости от значения параметра `throwIfNoManager`, метод может сгенерировать исключение типа `ItemNotFoundException`, если для этого локализуемого значения не задан диспетчер ресурсов.

Параметры

System.Globalization.CultureInfo culture	Языковая культура.
System.Boolean throwIfNoManager	Флаг, который указывает необходимость вызова исключения <code>ItemNotFoundException</code> .

T GetCultureValueWithFallback(CultureInfo culture, bool throwIfNoManager)

Получает локализуемое значение указанного типа для поддерживаемых языковых культур. Если значение локализуемого ресурса для указанной языковой культуры не найдено, то возвращается

значение на основной языковой культуре. В зависимости от значения параметра `throwIfNoManager`, метод может сгенерировать исключение типа `ItemNotFoundException`, если для этого локализуемого значения не задан диспетчер ресурсов.

Параметры

<code>System.Globalization.CultureInfo culture</code>	Языковая культура.
<code>System.Boolean throwIfNoManager</code>	Флаг, который указывает необходимость вызова исключения <code>ItemNotFoundException</code> .

```
bool HasCultureValue(CultureInfo culture)
```

Определяет, существует ли локализуемое значение для заданной языковой культуры.

Параметры

<code>System.Globalization.CultureInfo culture</code>	Языковая культура.
---	--------------------

```
void LoadCultureValues()
```

Загружает перечень локализуемых значений данного типа для всех языковых культур, которые определены в глобальном хранилище ресурсов.

```
void SetCultureValue(CultureInfo culture, T value)
```

Устанавливает заданное локализуемое значение для заданной языковой культуры.

Параметры

<code>System.Globalization.CultureInfo culture</code>	Языковая культура.
<code>T value</code>	Локализуемое значение.