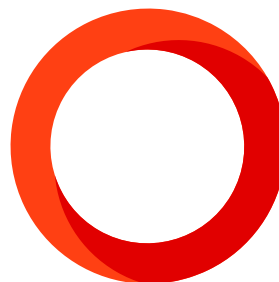
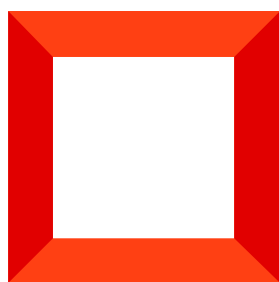
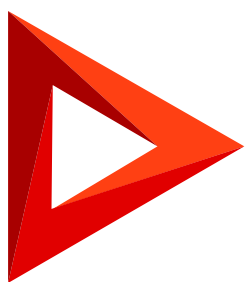


Лендинги

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

Web-to-Case	4
Логика автоматического заполнения полей для обращения	4
Рекомендации для выполнения проектных решений	5
Создать Web-to-Case лендинг	6
1. Создать новую запись лендинга в Creatio	6
2. Создать посадочную страницу	7
3. Добавить Web-страницу на сайт	9
Результат выполнения примера	10
Web-To-Object	11
Реализация сервиса Web-to-Object	11
Внешний API сервиса Web-to-Object	12
Интеграция с внешними системами	13
Настроить web-форму для создания пользовательского объекта	13
1. Зарегистрируйте новый тип лендинга	13
2. Добавьте страницу web-формы	14
3. Свяжите новый тип лендинга с созданной страницей записи	16
4. Актуализируйте наполнение скриптами для страницы web-формы	18
5. Создайте и настройте лендинг раздела [Лендинги и web-формы] ([Landing pages and web forms])	20
6. Разверните и настройте посадочную страницу, содержащую web-форму	21
7. Зарегистрируйте пользовательский объект в справочнике	25
Реализовать обработчик для создания сущности с помощью web-формы	27
1. Реализовать пользовательский обработчик	28
2. Зарегистрировать пользовательский обработчик в базе данных	30
Результат выполнения примера	32

Web-to-Case



Функциональность Web-to-Case реализует возможность создания обращений в Creatio посредством заполнения необходимых полей формы, встроенной на сторонний сайт — лендинга.

Пакет `ProductCore` зависит от пакета `WebForms`, в котором и содержится функциональность Web-to-Case. Это означает, что лендинги могут использоваться во всех продуктах. Преднастроенная "коробочная" функциональность реализована в продуктах service enterprise, customer center, marketing и всех бандлах, включающих эти продукты.

Подробнее о лендингах можно узнать из документации соответствующих продуктов, например, из блока статей "[Раздел \[Лендинги и web-формы \]](#)" документации Marketing Creatio.

Настройку Web-to-Case можно произвести, воспользовавшись интерфейсом системы, но для внедрения полученного JavaScript-скрипта на сторонний сайт необходимы общие базовые навыки Web-разработчика.

Базовая функциональность Web-to-Case без использования программирования в Creatio (но используя небольшие доработки на стороннем сайте) позволяет настроить следующее:

- Внешний вид формы и ее стили.
- Список дополнительно передаваемых полей.
- Список подстановочных значений по умолчанию для полей, не выведенных в форму.
- Список доменов, из которых будет возможна регистрация обращения по каждому лендингу.
- Адрес перехода, на который будет переадресован пользователь после отправки формы.
- JavaScript-обработчики событий успешной и неуспешной регистрации обращения.
- Дополнительные лендинги, каждый из которых может быть сконфигурирован по-своему, что дает возможность различить обращения, созданные с разных сайтов.

При помощи минимальных проектных доработок можно настроить предварительный обработчик регистрации обращения через Web-to-Case, в котором можно выполнить проверку данных, их коррекцию, создание связанных сущностей и т. п. В базовой поставке Creatio в обработчике регистрации обращения через Web-форму настроено автоматическое создание контакта для регистрируемого обращения.

Логика автоматического заполнения полей для обращения

При регистрации обращения через Web-форму предлагаются для заполнения следующие поля: [*ФИО*], [*Email*], [*Телефон*], [*Тема обращения*]. Значение поля [*Тема обращения*] будет передаваться в новое обращение.

По сочетанию полей [*ФИО*], [*Email*] и [*Телефон*] в Creatio идентифицируется контакт. Поиск происходит следующим образом:

1. Если существует контакт, у которого поля [ФИО], [Email] и [Телефон] совпадают с заполненными данными формы, то он подставляется в создаваемое обращение.
2. Иначе, если есть контакт, у которого совпадают только поля [ФИО] и [Email], то он подставляется в создаваемое обращение.
3. Иначе, если есть контакт, у которого совпадает только поле [Email], то он подставляется в создаваемое обращение.
4. Иначе создается новый контакт, и у него заполняются поля [ФИО], [Email] и [Телефон]. Созданный контакт подставляется в регистрируемое обращение.

Если по одному правилу найдено несколько контактов, то в качестве контакта обращения будет взят первый найденный. Также автоматически заполнится дата регистрации обращения (колонок `RegisteredOn`) текущей датой и временем.

Рекомендации для выполнения проектных решений

Если необходима кастомизация функциональности Web-to-Case, проще всего руководствоваться существующим базовым решением как примером.

Для выполнения проектного решения рекомендуется выполнить следующие действия:

1. Создать схему страницы, унаследованную от `CaseGeneratedWebFormPageV2`. Страница не должна быть замещающей.
2. Добавить запись нового типа лендинга в таблицу `LandingType` и локализацию в таблицу `SysLandingTypeLcz`.
3. Зарегистрировать стандартным образом типизированную страницу, созданную на первом шаге (значение типа — новое созданное).
4. Если необходима предварительная обработка данных формы до сохранения записи в базе данных, то нужно создать класс, реализующий интерфейс `IGeneratedWebFormPreProcessHandler`. Этот класс представляет собой предварительный обработчик регистрации обращения. Также нужно реализовать метод `Execute()`. Этот метод — точка входа в обработчик. В нем реализуются все вспомогательные действия. В качестве примера можно взять схему `WebFormCasePreProcessHandler`.
5. Если необходимо выполнить действия после сохранения обращения в базу данных, то нужно создать класс, реализующий интерфейс `IGeneratedWebFormPostProcessHandler`. Этот класс представляет собой предварительный обработчик регистрации обращения. Далее нужно реализовать метод `Execute()`, в котором выполнить необходимые действия.
6. Если созданы обработчики регистрации обращения, то нужно зарегистрировать их в таблице `WebFormProcessHandlers`. В качестве примера регистрации можно взять уже существующую запись.
7. Отредактировать шаблон скрипта, формирующего конфигурационный JavaScript-объект лендинга, и поместить в локализуемую строку `ScriptTemplate` созданной страницы. Аналогичный скрипт указать для всех используемых локализаций. Пример скрипта можно найти в схеме `CaseGeneratedWebFormPageV2`.
8. Привязать все созданные данные к пакету.

Алгоритм создания Web-to-Case лендинга:

1. Создать новую запись лендинга в Creatio.

2. Создать посадочную страницу, в которой будет содержаться сгенерированный в системе код, связывающий форму лендинга и запись лендинга.
3. Добавить посадочную страницу на сайт.

Создать Web-to-Case лендинг



Пример. Создать Web-to-Case лендинг.

1. Создать новую запись лендинга в Creatio

Чтобы создать новую запись лендинга, необходимо в разделе [Лендинги и web-формы] ([Landing pages and web forms]) выполнить действие [Добавить] ([Add]). В открывшейся странице нужно заполнить следующие поля:

- [Название] ([Name]) — заголовок лендинга в Creatio.
- [Домены сайта] ([Website domains]) — URL посадочной страницы.
- [Состояние] ([Status]) — состояние лендинга.
- [Адрес перехода] ([Redirection URL]) — URL страницы, на которую переходит клиент после регистрации на посадочной странице.

The screenshot shows the 'MyLanding' form in the Creatio system. The left sidebar contains navigation options: Service, Queues, Queues settings, Landing pages and web forms (highlighted), Feed, and Dashboards. The main form area has a header with 'MyLanding', a search bar, and buttons for 'SAVE', 'CANCEL', and 'ACTIONS'. The form fields are as follows:

- Name***: MyLanding
- Website domains***: localhost
- Description**: (empty)
- Status***: Active

Below the form fields is the 'LANDING SETUP' section, which includes two steps:

- STEP 1. Set up a redirection URL (optional)**: The 'Redirection URL' field is set to <http://bpmonline.com>.
- STEP 2. Copy the code and configure and map the**: A code block containing the following script:


```
<script src="http://ajax.googleapis.com/ajax/libs/j
<script src="https://webtracking-v01.bpmonline.com/
<script src="https://webtracking-v01.bpmonline.com/
<script>
/**
```

Поскольку из посадочной страницы при создании обращения можно получить только четыре поля ("Subject", "Email", "Name" и "Phone"), то для новой записи лендинга необходимо установить значения по умолчанию.

Field	Value
Account	Axiom
Category	Incident

Для применения изменений страницу нужно сохранить.

2. Создать посадочную страницу

Чтобы создать Web-страницу для лендинга нужно в любом текстовом редакторе при помощи HTML-разметки создать обычную HTML-страницу, содержащую Web-форму.

Для регистрации в Creatio данных, отправляемых через web-форму, в ее код необходимо добавить четыре поля (HTML-элемент `<input>`), определяющие обращение:

- тема обращения;
- Email контакта;
- имя контакта;
- телефон контакта

Для каждого поля нужно указать атрибуты `name` и `id`.

Чтобы при отправке данных формы в Creatio создавался новый объект [*Обращение*], в HTML-страницу нужно добавить скрипт на языке JavaScript. Исходный код скрипта необходимо скопировать из поля [*ШАГ 2. Скопируйте код и настройте в нем соответствие полей*] ([*STEP 2. Copy the code and configure and map the fields*]) страницы редактирования лендинга.

Скрипт необходимо скопировать из уже сохраненного лендинга.

Скрипт содержит конфигурационный объект `config`, в котором определены следующие свойства:

- `fields` — содержит объект со свойствами "Subject", "Email", "Name" и "Phone", значения которых должны совпадать с селекторами атрибутов `id` соответствующих полей формы.
- `landingId` — содержит идентификатор лендинга в базе данных.
- `serviceUrl` — содержит URL службы, по которому будут отправляться данные формы.
- `redirectUrl` — содержит URL адреса перехода, указанного в поле [Адрес перехода] лендинга.

- `onSuccess` — содержит функцию-обработчик успешного создания обращения. Необязательное СВОЙСТВО.
- `onError` — содержит функцию-обработчик ошибки создания обращения. Необязательное свойство.

Конфигурационный объект `config` передается в качестве аргумента функции `createObject()`, которая должна выполняться при отправке формы.

Чтобы функция `createObject()` была вызвана при отправке формы, в тег формы HTML-страницы лендинга необходимо добавить атрибут `onSubmit="createObject(); return false"`.

Пример полного исходного кода посадочной страницы для регистрации обращений

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <!--ШАГ 2-->
  <!--Эту часть необходимо скопировать из поля ШАГ 2 страницы редактирования лендинга-->
  <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
  <script src="https://webtracking-v01.creatio.com/JS/track-cookies.js"></script>
  <script src="https://webtracking-v01.creatio.com/JS/create-object.js"></script>
  <script>
    /**
     * Replace the "css-selector" placeholders in the code below with the element selectors
     * You can use #id or any other CSS selector that will define the input field explicitly
     * Example: "Email": "#MyEmailField".
     * If you don't have a field from the list below placed on your landing, leave the place
     */
    var config = {
      fields: {
        "Subject": "#subject-field", // Case subject
        "Email": "#email-field", // Visitor's email
        "Name": "#name-field", // Visitor's name code
        "Phone": "#phone-field", // Visitor's phone number
      },
      landingId: "8ab71187-0428-4372-b81c-fd05b141a2e7",
      serviceUrl: "http://localhost/creatioservice710/0/ServiceModel/GeneratedObjectWebFor",
      redirectUrl: "http://creatio.com",
      onSuccess: function(response) {
        window.alert(response.resultMessage);
      },
      onError: function(response) {
        window.alert(response.resultMessage);
      }
    };
  /**
   * The function below creates a object from the submitted data.
```



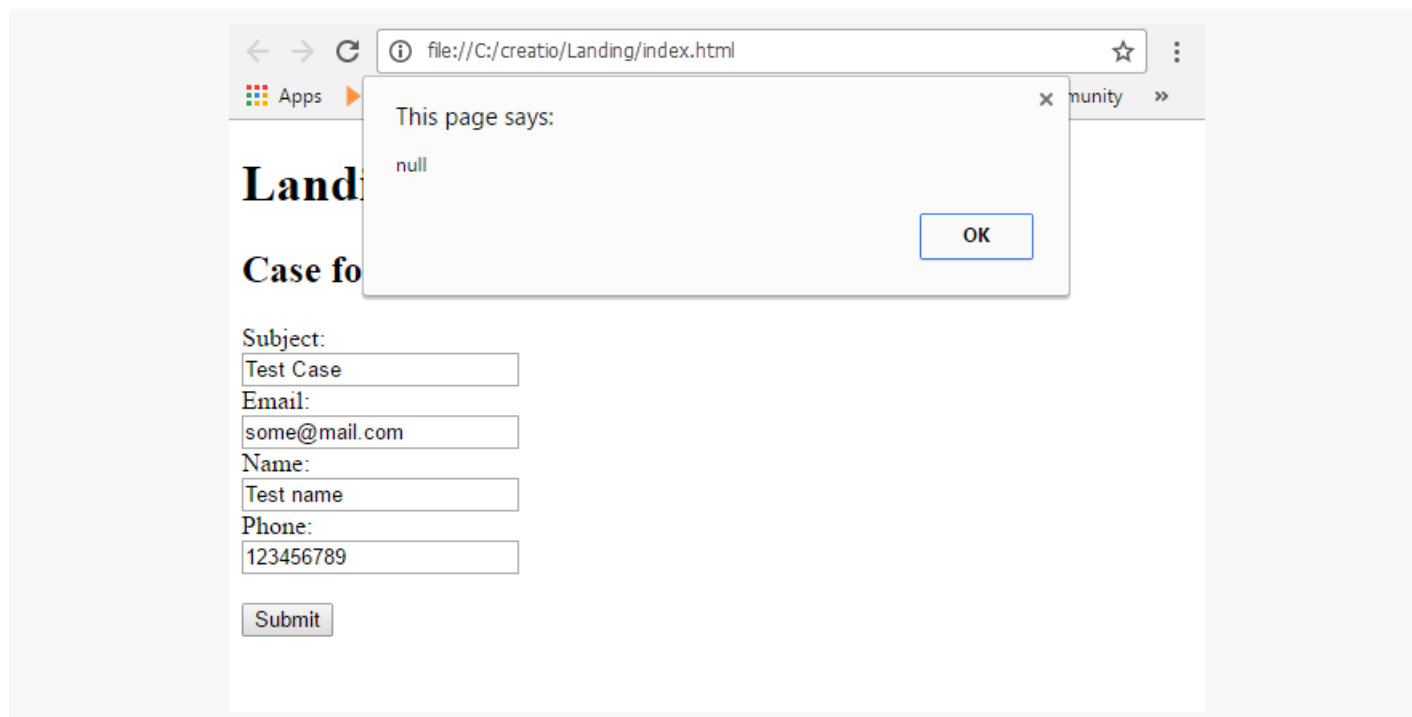
```

        * Bind this function call to the "onSubmit" event of the form or any other elements eve
        * Example: <form class="mainForm" name="landingForm" onSubmit="createObject(); return f
        */
    function createObject() {
        landing.createObjectFromLanding(config)
    }
</script>
<!--ШАГ 2-->
</head>
<body>
<h1>Landing web-page</h1>
<div>
    <h2>Case form</h2>
    <form class="mainForm" name="landingForm" onSubmit="createObject(); return false">
        Subject:<br>
        <input type="text" name="subject" id="subject-field"><br>
        Email:<br>
        <input type="text" name="Email" id="email-field"><br>
        Name:<br>
        <input type="text" name="Name" id="name-field"><br>
        Phone:<br>
        <input type="text" name="Phone" id="phone-field"><br><br>
        <input type="submit" value="Submit">
    </font>
    </form>
</div>
</body>
</html>

```

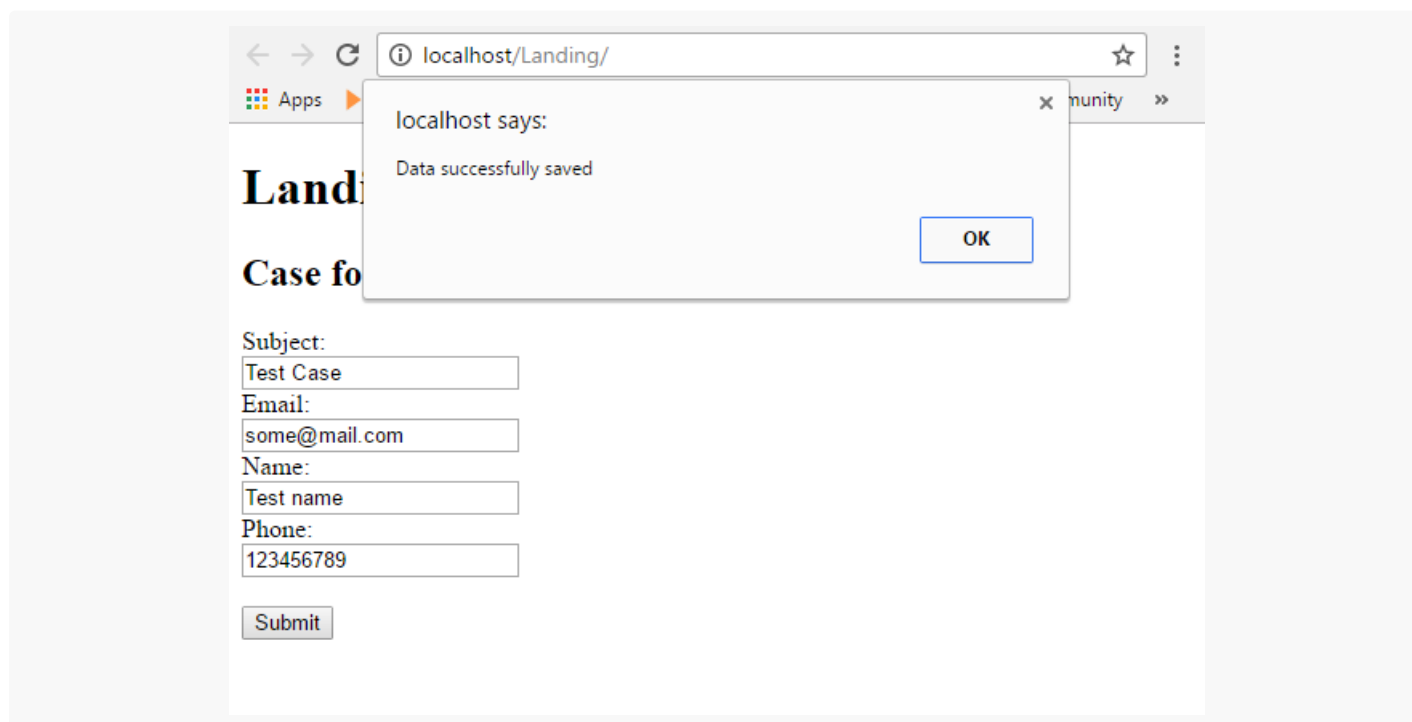
3. Добавить Web-страницу на сайт

Обращение с посадочной страницы будет добавлено в Creatio только в том случае, если страница размещена на сайте, имя которого указано в поле [*Домены сайта*] лендинга. Если открыть страницу в браузере локально, то при создании обращения будет выведено пустое сообщение.



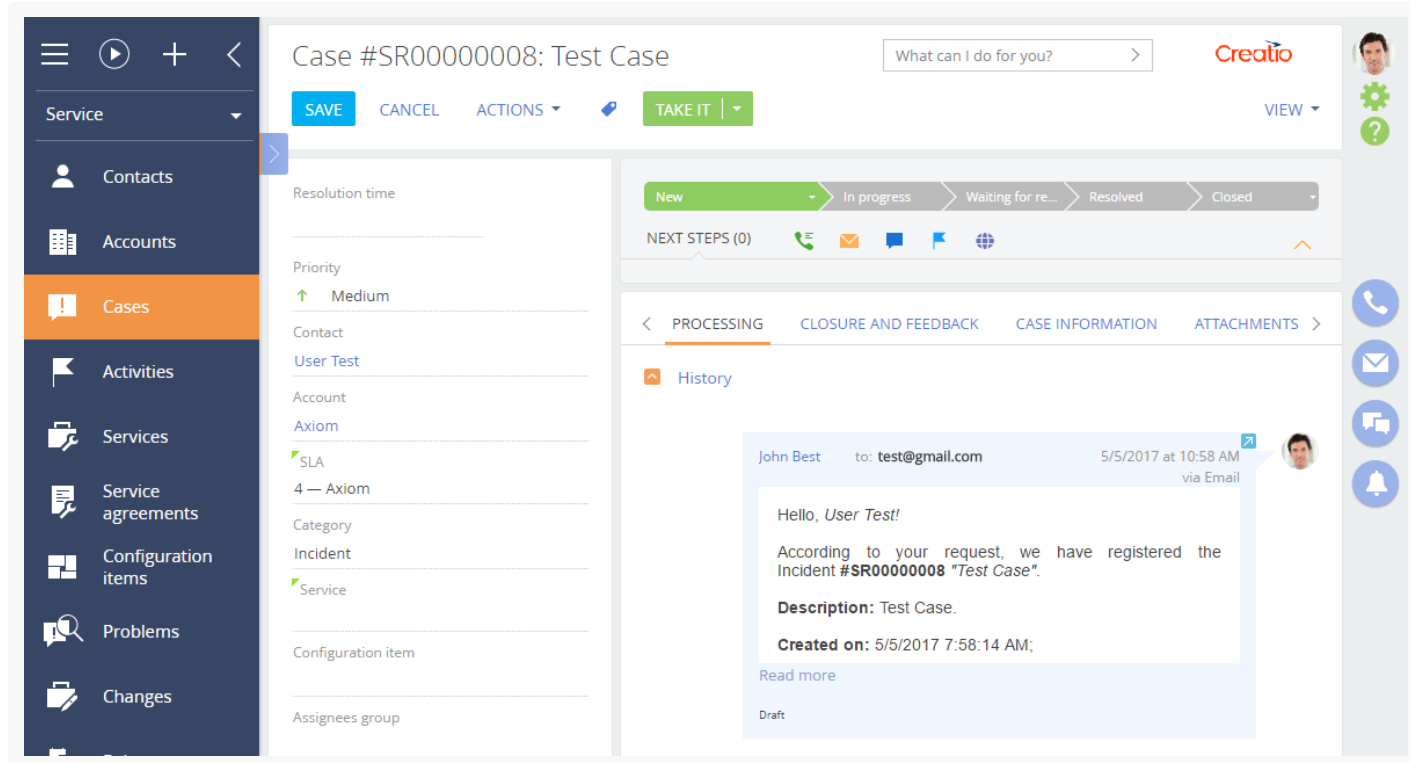
Вывод пустого сообщения настроен в методе-обработчике `onError()` конфигурационного объекта.

Если разместить страницу на локальном сервере компьютера, обслуживающий зарезервированное доменное имя `localhost` (как указано в настройке лендинга), то скрипт добавления обращения с Web-страницы лендинга отработает корректно.



Результат выполнения примера

В результате в системе будет автоматически создано обращение с указанными параметрами.



Web-To-Object



Web-to-Object — это механизм реализации простых односторонних интеграций с Creatio. С его помощью можно создавать записи в разделах Creatio (лиды, обращения, заказы и т. д.), просто отправив необходимые данные сервису Web-to-Object.

Наиболее распространенные **случаи использования** сервиса Web-to-Object:

- Интеграция Creatio с пользовательскими лендингами и веб-формами. Вызов сервиса выполняется из лендинга (особым образом настроенная пользовательская страница со специальной web-формой), после отправки заполненной web-формы посетителем.
- Интеграция с внешними системами, в результате работы которых должны создаваться объекты Creatio.

С помощью функциональности Web-to-Object можно настроить регистрацию в Creatio практически любых объектов. Например, в Creatio может быть зарегистрирован лид, контакт, заказ, обращение.

Для работы с лендингами в Creatio предусмотрен раздел [*Лендинги и web-формы*]. Этот раздел входит во все продукты Creatio, однако он может быть не включен по умолчанию в рабочие места некоторых продуктов (например, данный раздел не включен в рабочие места линейки продуктов Sales Creatio). Каждая запись раздела [*Лендинги и web-формы*] соответствует определенному лендингу. На странице записи предусмотрена вкладка [*Настройка лендинга*].

Реализация сервиса Web-to-Object

Основная функциональность механизма Web-To-Object содержится в пакете `WebForm` и является общей для всех продуктов. В зависимости от продукта Creatio эта функциональность расширена механизмами Web-to-Lead (пакет `WebLeadForm`), Web-to-Order (пакет `WebOrderForm`) и [Web-to-Case](#) (пакет `WebCaseForm`).

Для обработки данных, отправленных web-формой лендинга, в пакете `WebForm` реализован конфигурационный сервис `GeneratedObjectWebFormService` (класс

`Terrasoft.Configuration.GeneratedWebFormService`). Данные веб-формы лендинга принимаются в качестве аргумента метода `public string SaveWebFormObjectData(FormData formData)`. Затем они передаются в метод `public void HandleForm(FormData formData)` класса `Terrasoft.Configuration.WebFormHandler`, в котором и выполняется создание соответствующего объекта системы.

Внешний API сервиса Web-to-Object

Для использования сервиса необходимо отправить POST-запрос по адресу

[Путь к приложению Creatio]/0/ServiceModel/GeneratedObjectWebFormService.svc/SaveWebFormObjectData

Например

`http://mycreatio.com/0/ServiceModel/GeneratedObjectWebFormService.svc/SaveWebFormObjectData`

Тип содержимого запроса — `application/json`. Кроме необходимых cookies, в содержимое запроса нужно добавить JSON-объект, содержащий данные веб-формы. Пример JSON-объекта:

```
{
  "formData":{
    "formId":"d66ebbf6-f588-4130-9f0b-0ba3414dafb8",
    "formFieldsData":[
      {"name":"Name","value":"John Smith"},
      {"name":"Email","value":"j.smith@creatio.com"},
      {"name":"Zip","value":"00000"},
      {"name":"MobilePhone","value":"0123456789"},
      {"name":"Company","value":"Creatio"},
      {"name":"Industry","value":""},
      {"name":"FullJobTitle","value":"Sales Manager"},
      {"name":"UseEmail","value":""},
      {"name":"City","value":"Boston"},
      {"name":"Country","value":"USA"},
      {"name":"Commentary","value":""},
      {"name":"BpmHref","value":"http://localhost/Landing/"},
      {"name":"BpmSessionId","value":"0ca32d6d-5d60-9444-ec34-5591514b27a3"}
    ]
  }
}
```

Интеграция с внешними системами

Для интеграции с внешними системами необходимо:

1. Добавить запись в разделе [*Лендинги и web-формы*].
2. Получить из конфигурационного объекта созданной записи адрес к сервису (свойство `serviceUrl`) и идентификатор (свойство `landingId`).
3. Реализовать во внешней системе отправку POST-запроса к сервису Web-to-Object по полученному адресу. В запрос добавить необходимые данные в виде JSON-объекта. Свойству `formId` отправляемого JSON-объекта установить значение полученного идентификатора.

Настроить web-форму для создания пользовательского объекта



Используя web-форму посадочной страницы (лендинга) стороннего сайта можно создать пользовательский объект в приложении Creatio. Подробнее о лендингах можно узнать из блока статей "[Раздел \[*Лендинги и web-формы* \]](#)".

Общий порядок действий при создании пользовательского объекта через web-форму:


1. Зарегистрировать новый тип лендинга.
2. Добавить страницу записи web-формы.
3. Связать новый тип лендинга с созданной страницей записи.
4. Актуализировать наполнение скриптами для страницы web-формы.
5. Создать и настроить лендинг раздела **[Лендинги и web-формы] ([Landing pages and web forms])**.
6. Развернуть и настроить посадочную страницу, содержащую web-форму.
7. Зарегистрировать пользовательский объект в справочнике **[Настройки объекта для элемента кампании "Лендинг" ([Entity settings for campaign landing element])]**.

Важно. Трекинг событий сайта работает только для лидов. Для пользовательских объектов трекинг событий сайта не работает.

Пример. Через web-форму посадочной страницы создайте пользовательский объект [*Контакт*] ([*Contact*]).

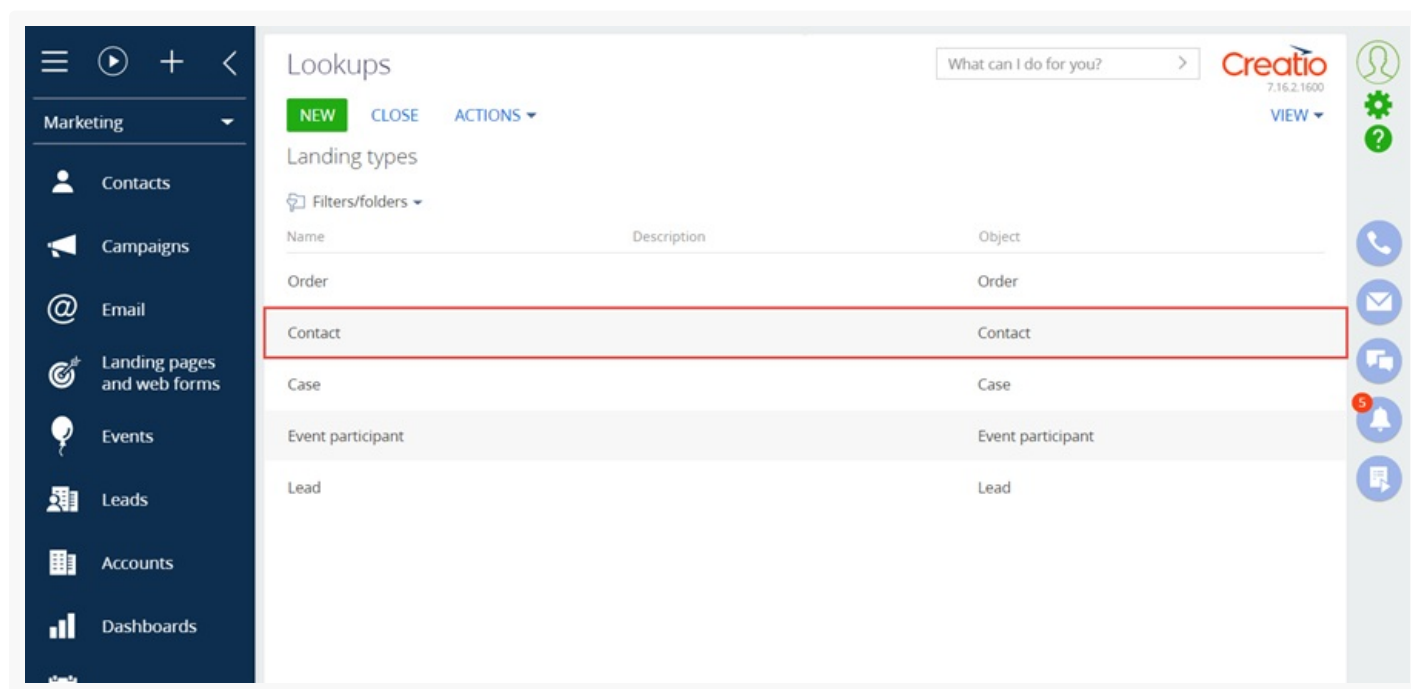
1. Зарегистрируйте новый тип лендинга

Для регистрации нового типа лендинга выполните следующие действия:

1. Перейдите в дизайнер системы по кнопке . В блоке **[Настройка системы] ([System setup])** перейдите по ссылке **[Справочники] ([Lookups])**.
2. Выберите справочник **[Типы лендингов] ([Landing types])**.
3. Создайте новую запись.

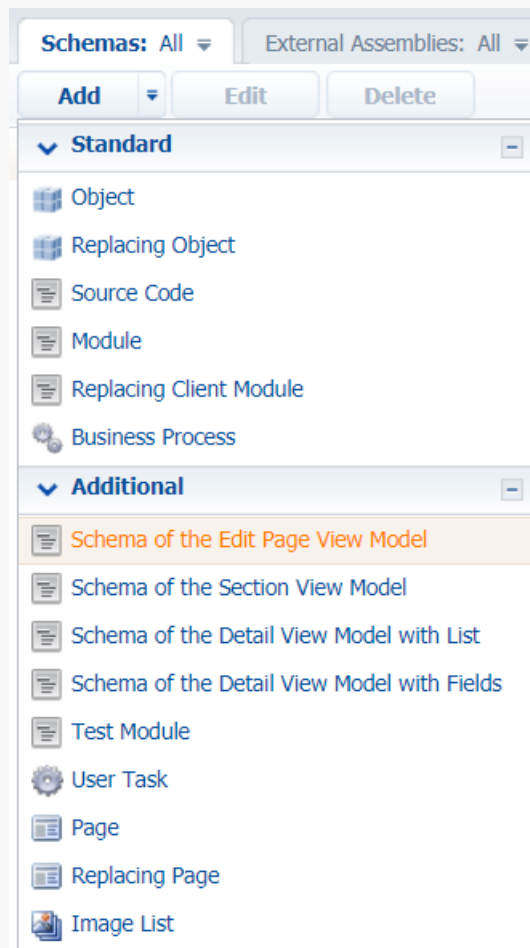
Для создаваемой записи установите:

- **[Название] ([Name])** — "Contact";
- **[Объект] ([Object])** — "Contact".



2. Добавьте страницу web-формы

В разделе **[Конфигурация] ([Configuration])** пользовательского пакета на вкладке **[Схемы] ([Schemas])** выполните действие **[Добавить] —> [Схема модели представления карточки] ([Add] —> [Schema of the Edit Page View Module])**. Процесс создания схемы модели представления карточки описан в статье "[Создать клиентскую схему](#)".



Для создаваемой схемы модели представления карточки установите:

- **[Заголовок] ([Title])** — "ContactGeneratedWebFormPage";
- **[Название] ([Name])** — "UsrContactGeneratedWebFormPage";
- **[Родительский объект] ([Parent object])** — "Edit page, landing".

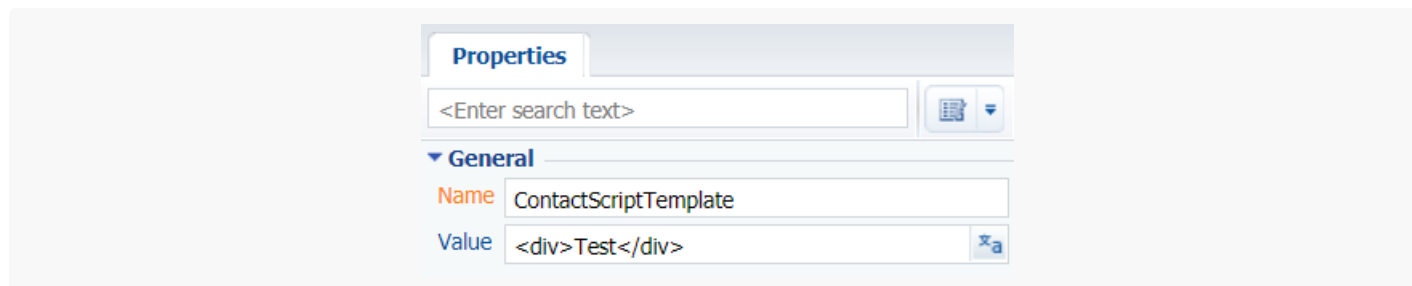
Properties	
<Enter search text>	
▼ General	
Title	ContactGeneratedWebFormPage
Name	UsrContactGeneratedWebFormPage
Package	sdkWebFormPackage
▼ Inheritance	
Parent object	Edit page, landing (WebForm)
Replace parent	<input checked="" type="checkbox"/>

UsrContactGeneratedWebFormPage.js

```
// UsrContactGeneratedWebFormPage – уникальное название схемы.
define("UsrContactGeneratedWebFormPage", ["UsrContactGeneratedWebFormPageResources"],
function() {
    return {
        details: /**SCHEMA_DETAILS*/{}/**SCHEMA_DETAILS*/,
        methods: {
            /**
             * @inheritdoc BaseGeneratedWebFormPageV2#getScriptTemplateFromResources
             * @overridden
             */
            getScriptTemplateFromResources: function() {
                var scriptTemplate;
                if (this.getIsFeatureEnabled("OutboundCampaign")) {
                    // ContactScriptTemplate – имя локализуемой строки.
                    scriptTemplate = this.get("Resources.Strings.ContactScriptTemplate");
                } else {
                    scriptTemplate = this.get("Resources.Strings.ScriptTemplate");
                }
                return scriptTemplate;
            },
            diff: /**SCHEMA_DIFF*/[/**SCHEMA_DIFF*/
        ];
    });
});
```

После внесения изменений сохраните схему.

Добавьте локализуемую строку `ContactScriptTemplate`. В значение строки добавьте `<div>Test</div>`. Больше информации о работе с локализуемыми строками содержится в статье "[Работа с локализуемыми ресурсами](#)".



После внесения изменений сохраните схему.

3. Свяжите новый тип лендинга с созданной страницей записи

Чтобы связать новый тип лендинга с созданной страницей записи, добавьте запись в таблицу **[dbo.SysModuleEdit]** базы данных. Для этого выполните следующий SQL-запрос:

Запрос для добавления записи в таблицу dbo.SysModuleEdit

```
-- Parameters of new landing page
DECLARE @editPageName nvarchar(250) = N'UsrContactGeneratedWebFormPage'; -- название созданной с
DECLARE @landingTypeName NVARCHAR(250) = N'Contact'; -- название типа лендинга
DECLARE @actionCaption NVARCHAR(250) = N'Contact form'; -- название типа лендинга в разделе при

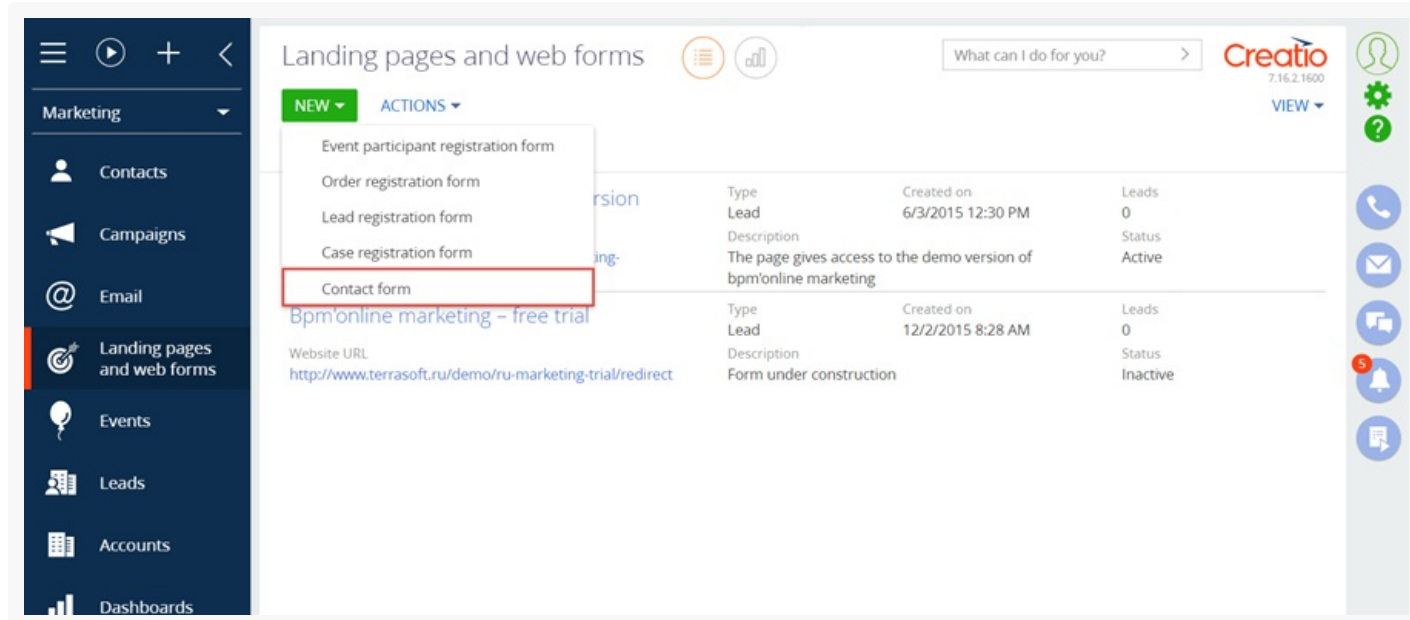
-- Set system parameters based on new landing page
DECLARE @generatedWebFormEntityUid uniqueidentifier = '41AE7D8D-BEC3-41DF-A6F0-2AB0D08B3967';
DECLARE @cardSchemaUid uniqueidentifier = (select top 1 Uid from SysSchema where Name = @editPage
DECLARE @pageCaption nvarchar(250) = (select top 1 Caption from SysSchema where Name = @editPage
DECLARE @sysModuleEntityId uniqueidentifier = (select top 1 Id from SysModuleEntity where SysEnt
DECLARE @landingTypeId uniqueidentifier = (SELECT TOP 1 Id FROM LandingType WHERE Name = @landir

-- Adding new Landing page variant to application interface
INSERT INTO SysModuleEdit
(Id, SysModuleEntityId, TypeColumnValue, UseModuleDetails, CardSchemaUid, ActionKindCaption, Act
VALUES
(NEWID(), @sysModuleEntityId, @landingTypeId, 1, @cardSchemaUid, @actionCaption, @editPageName,
```

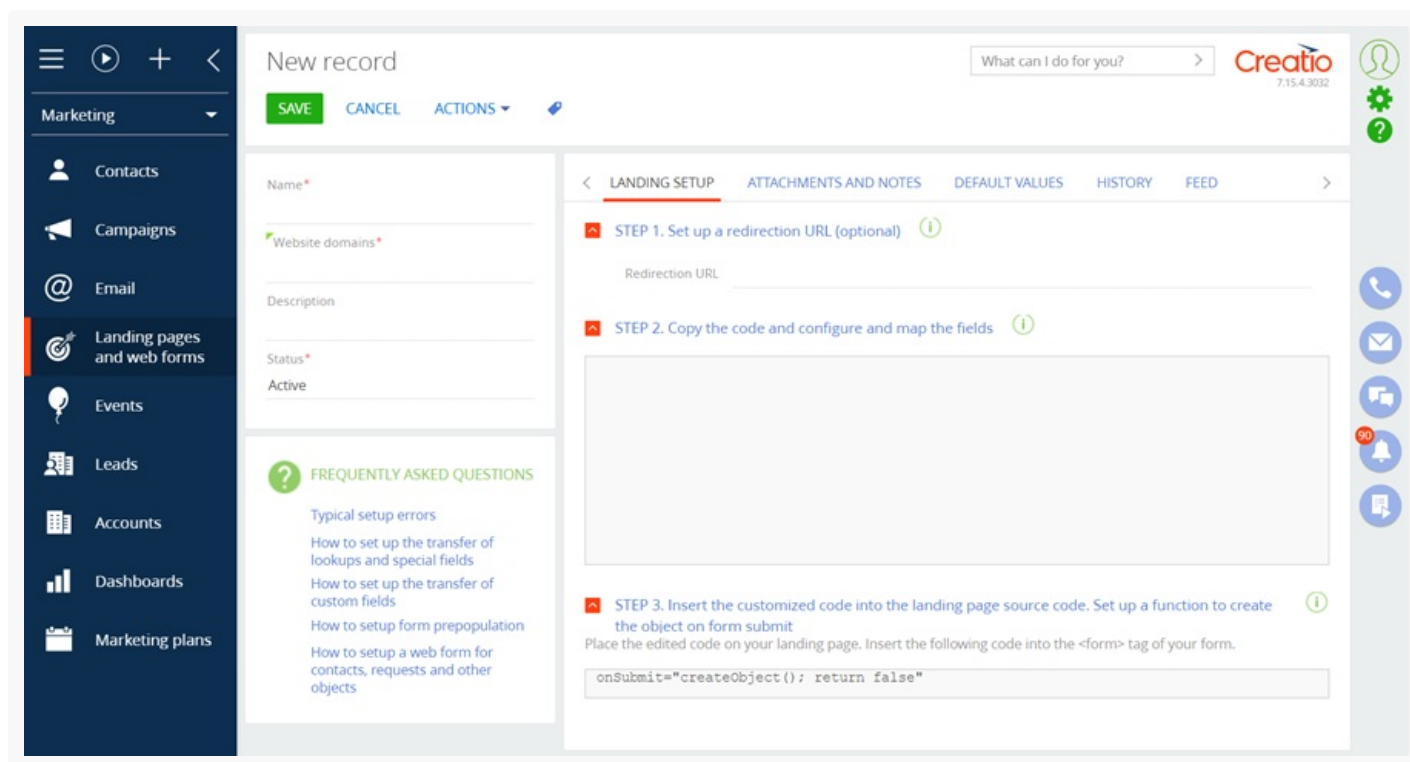
Важно. 41AE7D8D-BEC3-41DF-A6F0-2AB0D08B3967 — неизменный идентификатор схемы сущности GeneratedWebForm в таблице [**dbo.SysSchema**] базы данных для любого кейса добавления посадочной страницы для пользовательской сущности.

После выполнения скрипта очистите кэш браузера. В результате в разделе [**Лендинги и web-формы**] ([**Landing pages and web forms**]) появится возможность добавить новый тип лендинга [**Contact form**]. Но при открытии страницы лендинга будет отсутствовать скрипт, который необходимо добавить в код посадочной страницы.

Реестр раздела [**Лендинги и web-формы**] ([**Landing pages and web forms**])



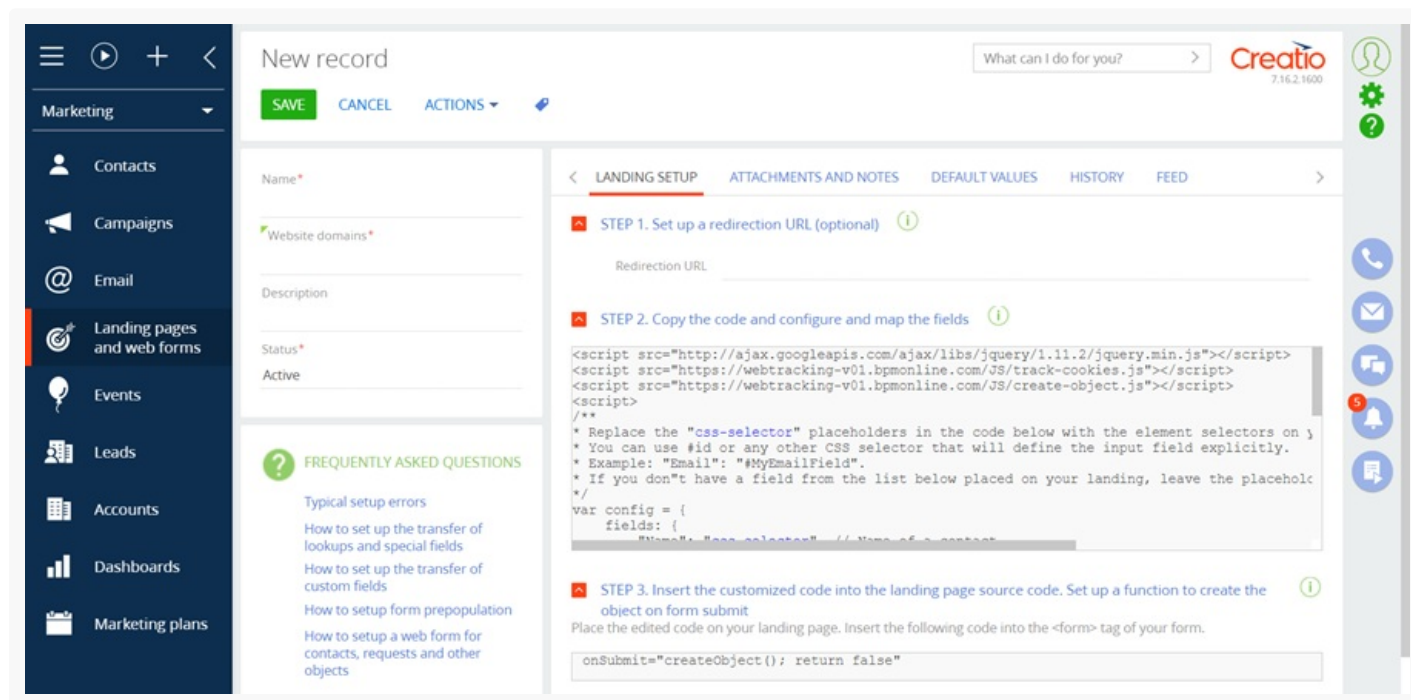
Страница лендинга



4. Актуализируйте наполнение скриптами для страницы web-формы

Значение переменной содержит экранированный html с тегами `<script>` и другую информацию по настройке связи полей web-формы — колонок создаваемой сущности. Это значение должно быть локализуемым. Для этого выполните следующий SQL-запрос:

который необходимо разместить в коде посадочной страницы.



Скрипт содержит конфигурационный объект `config`, в котором определены следующие свойства:

- `fields` — содержит объект со свойствами `Name` и `Email`, значения которых должны совпадать с селекторами атрибутов `id` соответствующих полей web-формы.
- `landingId` — содержит идентификатор лендинга в базе данных.
- `serviceUrl` — содержит URL службы, по которому будут отправляться данные web-формы.
- `onSuccess` — содержит функцию-обработчик успешного создания контакта. Необязательное свойство.
- `onError` — содержит функцию-обработчик ошибки создания контакта. Необязательное свойство.

Конфиг, который будет формироваться, представлен ниже.

```
var config = {
  fields: {
    "Name": "css-selector", // Имя контакта
    "Email": "css-selector", // Email контакта
  },
  landingId: "b73790ab-acb1-4806-baea-4342a1f3b2a8",
  serviceUrl: "http://localhost:85/0/ServiceModel/GeneratedObjectWebFormService.svc/SaveWebForm",
  redirectUrl: ""
};
```

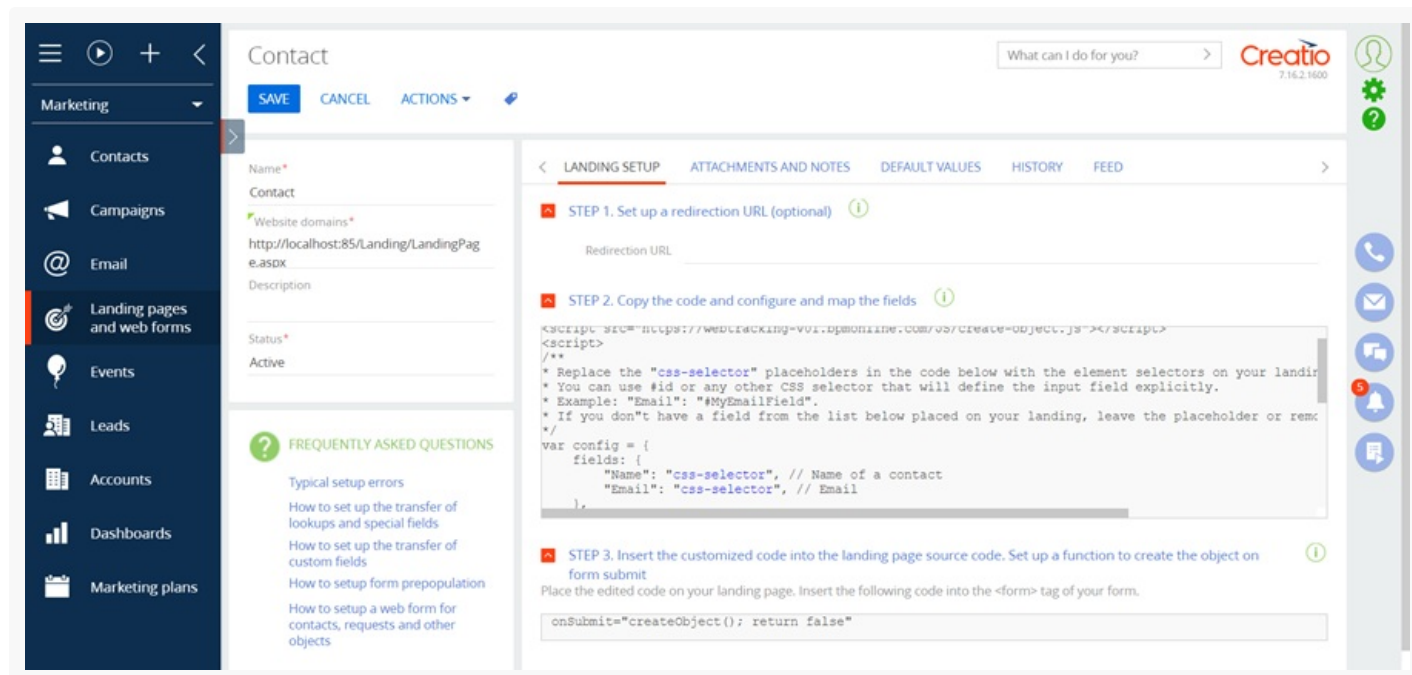
5. Создайте и настройте лендинг раздела [Лендинги и

web-формы] ([Landing pages and web forms])

Чтобы создать новую запись лендинга необходимо в разделе [**Лендинги и web-формы**] ([**Landing pages and web forms**]) выбрать [**Contact form**]. Добавление записи в раздел [**Лендинги и web-формы**] ([**Landing pages and web forms**]) описано в статье "[Как добавить новую запись в разделе \[Лендинги и web-формы \]](#)".

Для создаваемой записи установите:

- [**Название**] ([**Name**]) — "Contact";
- [**Домены сайта**] ([**Website domains**]) — "http://localhost:85/Landing/LandingPage.aspx";
- [**Состояние**] ([**Status**]) — "Active".



Для применения изменений сохраните страницу.

6. Разверните и настройте посадочную страницу, содержащую web-форму

Чтобы создать посадочную страницу для лендинга необходимо в любом текстовом редакторе при помощи html-разметки создать обычную посадочную страницу, содержащую web-форму. Создание посадочной страницы и добавление скрипта созданного лендинга в код посадочной страницы описаны в статье "[Как связать лендинг на сайте с Creatio](#)".

Для регистрации в Creatio данных контакта, отправляемых через web-форму, в код посадочной страницы необходимо добавить следующие поля (html-элемент `<input>`):

- Имя контакта.
- Email контакта.

Для каждого поля необходимо указать атрибуты `name` и `id`.

Чтобы при отправке данных web-формы в Creatio создавался новый объект [**Контакт**] ([**Contact**]), в посадочную страницу добавьте скрипт на языке JavaScript. Исходный код скрипта скопируйте из поля [**ШАГ 2. Скопируйте код и настройте в нем соответствие полей**] ([**STEP 2. Copy the code and configure and map the fields**]) страницы лендинга (рис. 8).

Конфигурационный объект `config` передается в качестве аргумента функции `createObject()`, которая должна выполняться при отправке web-формы.

Чтобы функция `createObject()` была вызвана при отправке web-формы, в тег `<form>` web-формы посадочной страницы добавьте атрибут `onSubmit="createObject(); return false"` из поля [**ШАГ 3. Вставьте настроенный код на страницу лендинга. Настройте запуск функции создания объекта по submit формы**] ([**STEP 3. Insert the customized code into the landing page source code. Set up a function to create the object on form submit**]) страницы лендинга.

Исходный код посадочной страницы

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <!--ШАГ 2-->
  <!--Эту часть необходимо скопировать из поля ШАГ 2 страницы редактирования лендинга-->
  <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
  <script src="https://webtracking-v01.bpmonline.com/JS/track-cookies.js"></script>
  <script src="https://webtracking-v01.bpmonline.com/JS/create-object.js"></script>
  <script>

  /**
   * Замените выражение в кавычках "css-selector" в коде ниже значением селектора элемента на Б
   * Вы можете использовать #id или любой другой CSS селектор, который будет точно определять г
   * Пример: "Email": "#MyEmailField".
   * Если Ваша лендинговая страница не содержит одного или нескольких полей из приведенных ниже
   */
  var config = {
    fields: {
      "Name": "#name-field", // Имя контакта
      "Email": "#email-field", // Email контакта
    },
    landingId: "b73790ab-acb1-4806-baea-4342a1f3b2a8",
    serviceUrl: "http://localhost:85/0/ServiceModel/GeneratedObjectWebFormService.svc/SaveWe
    redirectUrl: "",
    onSuccess: function(response) {
      window.alert(response.resultMessage);
    },
    onError: function(response) {
      window.alert(response.resultMessage);
    }
  };
  /**
```

```

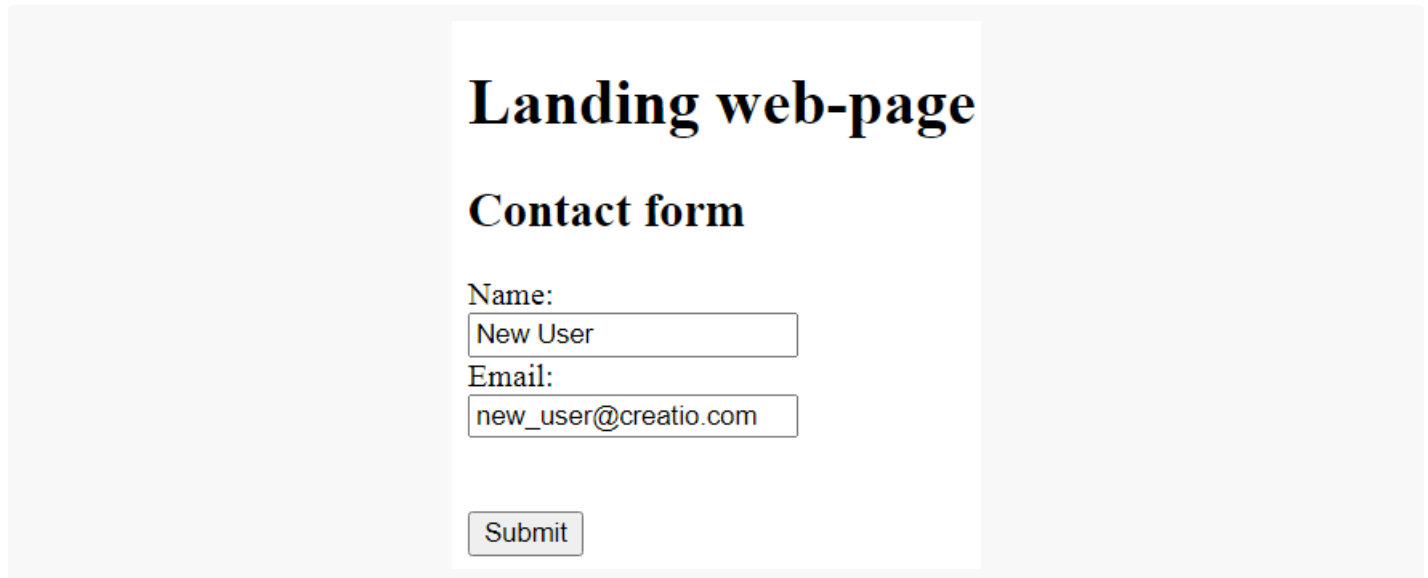
* Функция ниже создает объект из введенных данных.
* Привяжите вызов этой функции к событию "onSubmit" формы или любому другому элементу события
* Пример: <form class="mainForm" name="landingForm" onSubmit="createObject(); return false">
*/
function createObject() {
    landing.createObjectFromLanding(config)
}
/**
* Функция ниже инициализирует лендинг из параметров URL.
*/
function initLanding() {
    landing.initLanding(config)
}
jQuery(document).ready(initLanding)
</script>
<!--ШАГ 2-->

</head>
<body>
<h1>Landing web-page</h1>
<div>
    <h2>Contact form</h2>
    <form method="POST" class="mainForm" name="landingForm" onSubmit="createObject(); return false">
        Name:<br>
        <input type="text" name="Name" id="name-field"><br>
        Email:<br>
        <input type="text" name="Email" id="email-field"><br><br><br>
        <input type="submit" value="Submit">
    </font>
    </form>
</div>
</body>
</html>

```

Откройте посадочную страницу. Для создаваемого контакта установите:

- **[Name]** — "New User";
- **[Email]** — "new_user@creatio.com".

A screenshot of a web form titled "Landing web-page" and "Contact form". It contains two input fields: "Name:" with the value "New User" and "Email:" with the value "new_user@creatio.com". Below the fields is a "Submit" button.

Landing web-page

Contact form

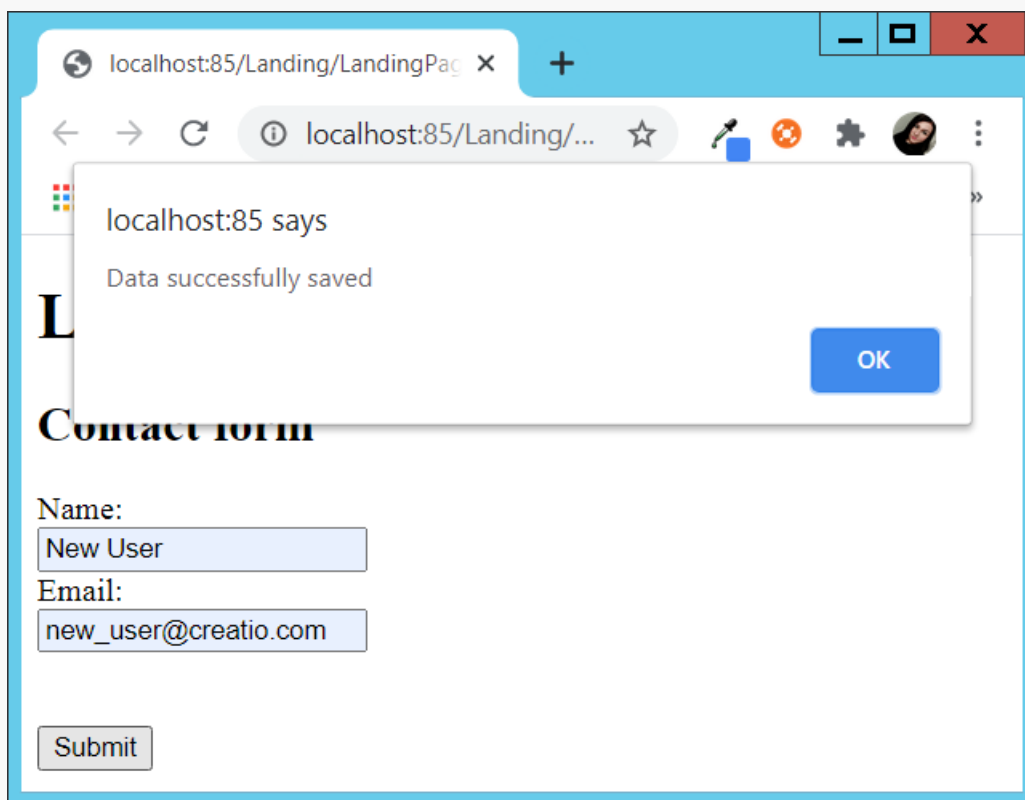
Name:

Email:

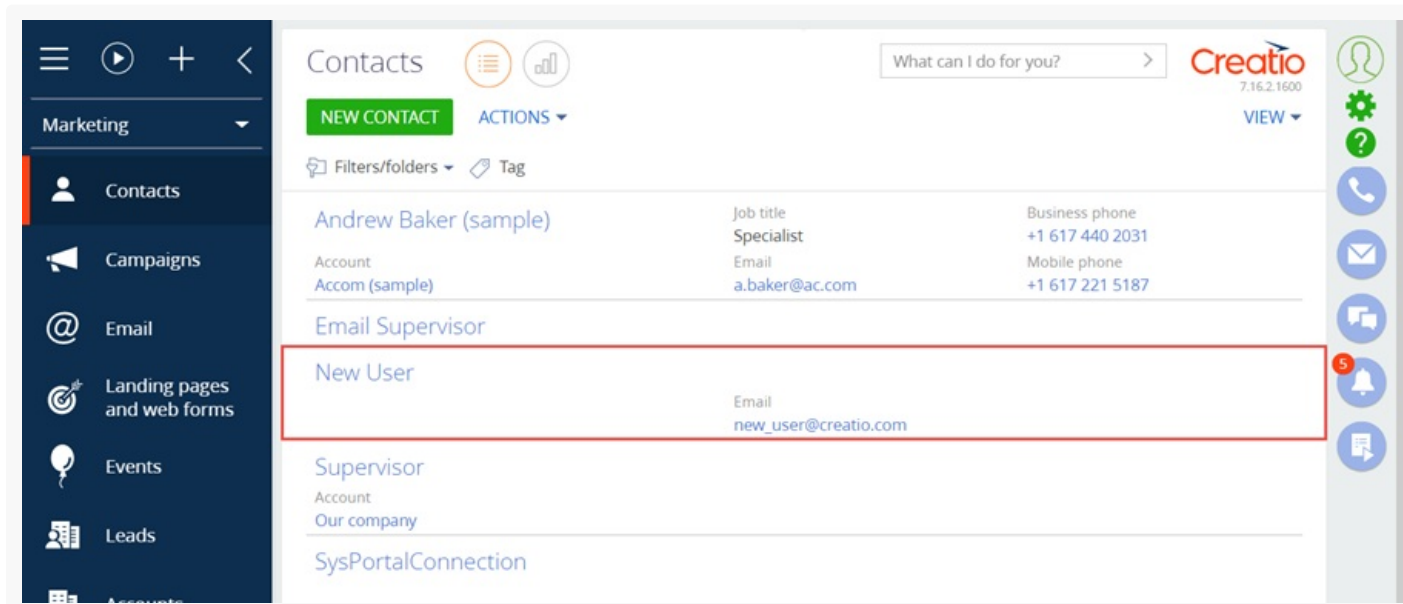
Для создания контакта нажмите [**Submit**].

Важно. Контакт с посадочной страницы будет добавлен в Creatio только в том случае, если страница размещена на сайте, имя которого указано в поле [**Домены сайта**] ([**Website domains**]) лендинга.

Если разместить страницу на локальном сервере компьютера, обслуживающего [зарезервированное доменное имя localhost](#) (как указано в настройке лендинга), то скрипт для создания контакта с посадочной страницы лендинга отработает корректно.



В результате в системе будет автоматически создан контакт с указанными параметрами.




7. Зарегистрируйте пользовательский объект в справочнике

Для использования пользовательского лендинга в маркетинговых кампаниях (элемент **[Добавить из лендинга] ([Landing page])**), нужно зарегистрировать его в справочнике.

Регистрация объекта в справочнике **[Настройки объекта для элемента кампании "Лендинг"]**

([Entity settings for campaign landing element]):

1. Перейдите в дизайнер системы по кнопке .
2. В блоке **[Настройка системы]** перейдите по ссылке **[Справочники] ([Lookups])**.
3. Откройте справочник **[Настройки объекта для элемента кампании "Лендинг"] ([Entity settings for campaign landing element])**.
4. Нажмите **[Добавить] ([Add])**. Для создаваемого объекта заполните поля:
 - **[Название] ([Caption])** — пользовательское название объекта.
 - **[Объект] ([Entity object])** — создаваемый посадочной страницей объект системы.
 - **[Путь к Контакту] ([Path to Contact])** — путь к колонке, которая связывает запись с контактом.
 - **[Путь к идентификатору веб-формы] ([Path to WebForm])** — путь к колонке, которая связывает запись с посадочной страницей.
5. Сохраните объект.

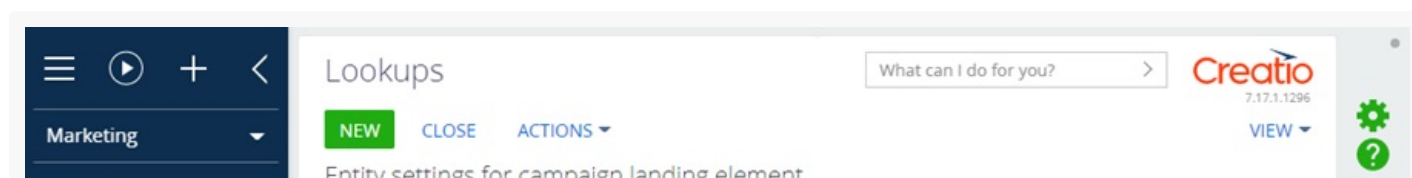
Пример заполнения полей для объектов **[Лид] ([Lead])** и **[Участник мероприятия] ([Event participant])** представлен в таблице.

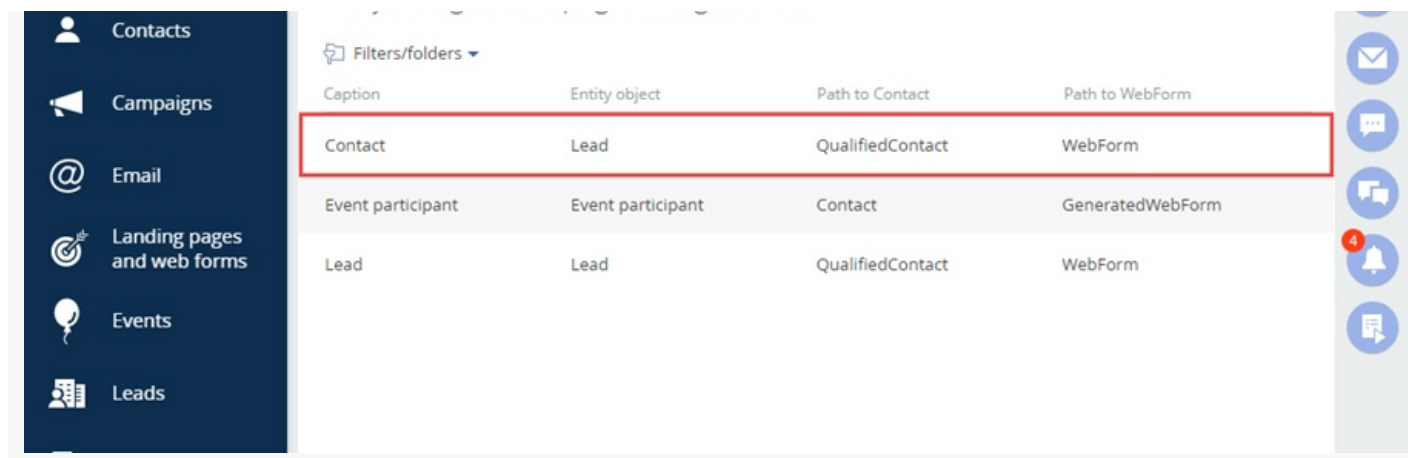
Полей объектов **[Лид] ([Lead])** и **[Участник мероприятия] ([Event participant])**

Имя поля	Значение поля объекта	
	[Лид] ([Lead])	[Участник мероприятия] ([Event participant])
[Название] ([Caption])	Lead	Event participant
[Объект] ([Entity object])	Lead	Event participant
[Путь к Контакту] ([Path to Contact])	QualifiedContact	Contact
[Путь к идентификатору веб-формы] ([Path to WebForm])	WebForm	GeneratedWebForm

Для создаваемой записи установите:

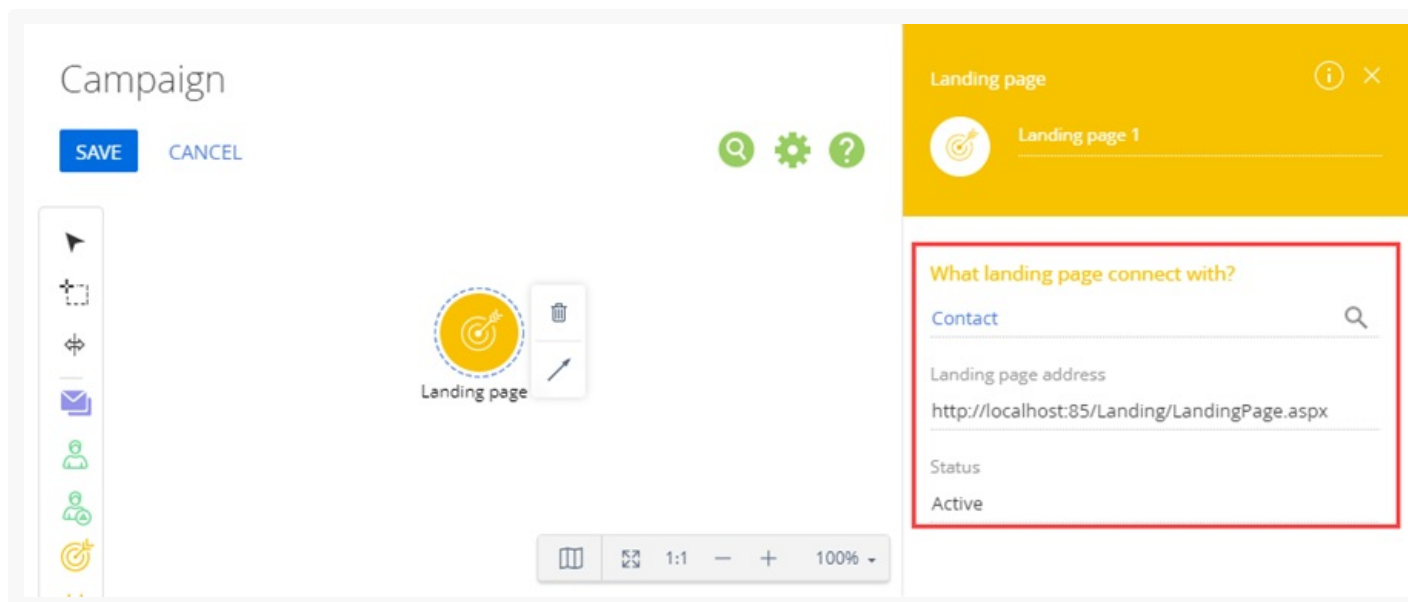
- **[Название] (Caption)** — "Contact";
- **[Объект] ([Entity object])** — "Lead";
- **[Путь к Контакту] ([Path to Contact])** — "QualifiedContact";
- **[Путь к идентификатору веб-формы] ([Path to WebForm])** — "WebForm".





Filters/folders	Caption	Entity object	Path to Contact	Path to WebForm
	Contact	Lead	QualifiedContact	WebForm
	Event participant	Event participant	Contact	GeneratedWebForm
	Lead	Lead	QualifiedContact	WebForm

Пользовательский лендинг "Contact" доступен к выбору в элементе **[Добавить из лендинга] ([Landing page])** маркетинговых кампаний.



Campaign

SAVE CANCEL

Landing page

Landing page 1

What landing page connect with?

Contact

Landing page address

http://localhost:85/Landing/LandingPage.aspx

Status

Active

Реализовать обработчик для создания сущности с помощью web-формы

 Сложный

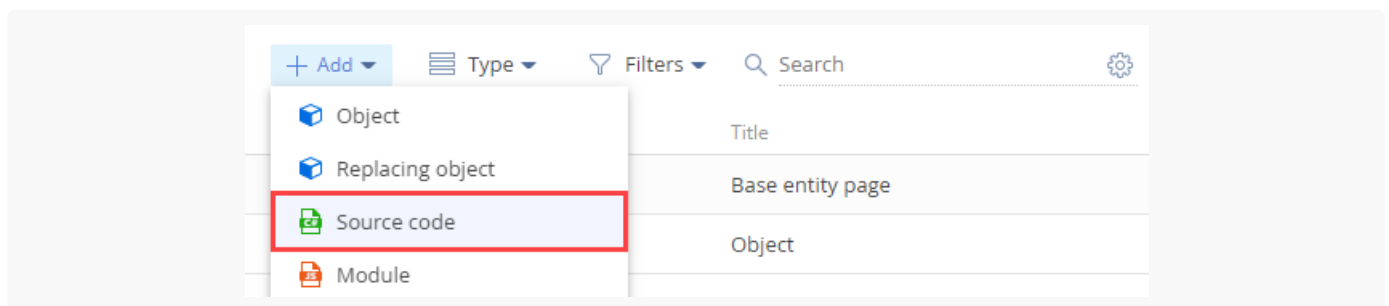
В сущности `Contact` присутствует обязательная текстовая колонка `CustomRequiredTextColumn`. При создании участника мероприятия (`EventTarget`) при помощи web-формы в приложении выполняется поиск соответствующего контакта. Если контакт не найден, то по умолчанию создается новый контакт. При сохранении контакта возникает ошибка, поскольку не заполнено обязательное поле `CustomRequiredTextColumn`. Чтобы успешно **сохранить контакт**, необходимо реализовать пользовательский обработчик перед созданием участника мероприятия.

Пример. Реализовать пользовательский обработчик перед созданием участника мероприятия.

Перед выполнением примера настройте web-форму для создания пользовательского объекта. В web-форму добавьте обязательное пользовательское поле `CustomRequiredTextColumn`. Для этого воспользуйтесь инструкцией, которая приведена в статье [Настроить web-форму для создания пользовательского объекта](#).

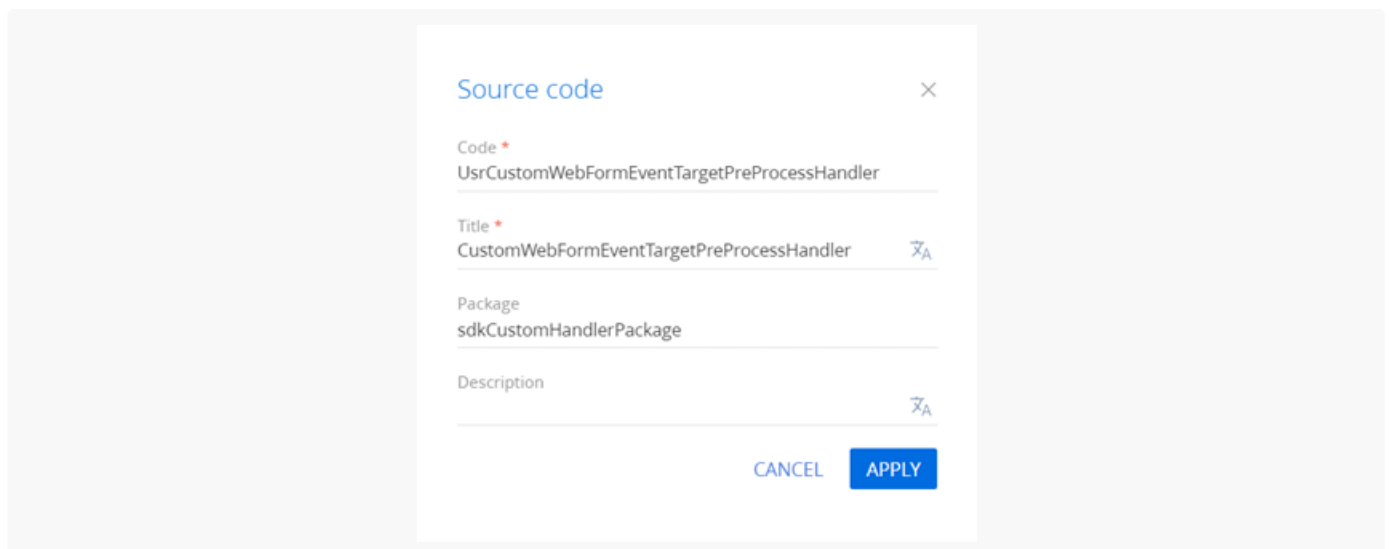
1. Реализовать пользовательский обработчик

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 дизайнера схем заполните свойства схемы:

- [*Код*] ([*Code*]) — "UsrCustomWebFormEventTargetPreProcessHandler".
- [*Заголовок*] ([*Title*]) — "CustomWebFormEventTargetPreProcessHandler".



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

4. Реализуйте **пользовательский обработчик перед созданием участника мероприятия**.
 - a. В дизайнера схем добавьте пространство имен `Terrasoft.Configuration`.

- b. С помощью директивы `using` добавьте пространства имен, типы данных которых будут задействованы в классе.
- c. Добавьте название класса, которое соответствует названию схемы (свойство [Код] ([Code])).
- d. В качестве родительского класса укажите класс `WebFormEventTargetPreProcessHandler`.

Исходный код схемы `UsrCustomWebFormEventTargetPreProcessHandler` типа [Исходный код] ([Source code]) представлен ниже.

`UsrCustomWebFormEventTargetPreProcessHandler`

```
namespace Terrasoft.Configuration
{
    using System;
    using System.Linq;
    using Core.Entities;
    using Core;
    using GeneratedWebFormService;

    #region Class: UsrCustomWebFormEventTargetPreProcessHandler
    /// <summary>
    /// Executes custom pre event target saving processing.
    /// </summary>
    /// <seealso cref="Terrasoft.Configuration.IGeneratedWebFormPreProcessHandler" />
    public class CustomWebFormEventTargetPreProcessHandler: WebFormEventTargetPreProcessHandler
    {

        #region Properties: Private
        private UserConnection _userConnection { get; set; }
        private FormData _formData { get; set; }
        #endregion

        #region Methods: Private
        private string GetCustomRequiredColumnValue(string customColumnName) {
            var customFormField = this._formData.formFieldsData
                .FirstOrDefault(x => x.name == customColumnName);
            if (customFormField == null) {
                throw new Exception($"There is no required form field {customColumnName}");
            }
            if (string.IsNullOrEmpty(customFormField?.value)) {
                throw new Exception($"Required value is empty for field {customColumnName}");
            }
            return customFormField.value;
        }
        #endregion

        #region Methods: Protected
        /// <summary>
```

```

    /// Creates contact entity with custom required text column filled with form value.
    /// </summary>
    /// <param name="contactId">Unique identifier of contact.</param>
    /// <param name="contactNameField">Required contact name form field.</param>
    protected override void CreateContactEntity(Guid contactId, FormFieldsData contactNameField,
        EntitySchema contactSchema = _userConnection.EntitySchemaManager.GetInstanceByName(
            "Contact", _userConnection);
        Entity contact = contactSchema.CreateEntity(_userConnection);
        contact.SetDefColumnValues();
        contact.SetColumnValue("Id", contactId);
        contact.SetColumnValue("Name", contactNameField.value);

        // set value for custom required column.
        var customRequiredColumnName = nameof(Contact.CustomRequiredTextColumn);
        var customRequiredColumnValue = GetCustomRequiredColumnValue(customRequiredColumnName, contactNameField.value);
        contact.SetColumnValue(customRequiredColumnName, customRequiredColumnValue);

        contact.Save(false);
    }
    #endregion

    #region Methods: Public
    /// <inheritdoc/>
    /// Override for base method implementation to init <see cref="UserConnection"/> instance
    public new FormData Execute(UserConnection userConnection, FormData formData,
        IWebFormImportParamsGenerator paramsGenerator) {
        _userConnection = userConnection;
        _formData = formData;
        return base.Execute(userConnection, formData, paramsGenerator);
    }
    #endregion
}
#endregion
}

```

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


2. Зарегистрировать пользовательский обработчик в базе данных

Реализация пользовательского обработчика предполагает его обязательную регистрацию в таблице [WebFormProcessHandlers] базы данных.

Способы регистрации пользовательского обработчика в базе данных:

- С помощью справочника.

Чтобы зарегистрировать пользовательский обработчик в базе данных **с помощью справочника**:

- Перейдите в дизайнер системы по кнопке .
- В блоке [*Настройка системы*] ([*System setup*]) перейдите по ссылке [*Справочники*] ([*Lookups*]).
- Добавьте новую запись для обработчика в справочник для сущности [*Обработчики веб-форм*] ([*Web form process handlers*]). По умолчанию этот справочник не добавлен в перечень справочников приложения. Чтобы добавить справочник [*Обработчики веб-форм*] ([*Web form process handlers*]) в приложение, создайте справочник и в качестве объекта выберите объект [*Обработчики веб-форм*] ([*Web form process handlers*]).
- Заполните **поля справочника**:
 - [*Имя объекта*] — "EventTarget".
 - [*FullName*] — "Terrasoft.Configuration.CustomWebFormEventTargetPreProcessHandler, Terrasoft.Configuration".
 - Установите признак [*Активный*] ([*isActive*]).
- С помощью SQL-запроса.


Чтобы зарегистрировать пользовательский обработчик в базе данных **с помощью SQL-запроса**, выполните следующий SQL-запрос.

SQL-запрос

```
INSERT INTO WebFormProcessHandlers (Id, EntityName, FullClassName, IsActive)
VALUES (NEWID(), N'EventTarget', 'Terrasoft.Configuration.CustomWebFormEventTargetPreProc
```

Поскольку схема наследуется от коробочного обработчика и базовая логика вызывается в пользовательской схеме, то необходимо отключить коробочный обработчик.

Способы отключения коробочного обработчика:

- С помощью справочника.
Чтобы отключить коробочный обработчик **с помощью справочника**:
 - Перейдите в дизайнер системы по кнопке .
 - В блоке [*Настройка системы*] ([*System setup*]) перейдите по ссылке [*Справочники*] ([*Lookups*]).
 - Откройте справочник и для сущности [*Участник мероприятия*] ([*EventTarget*]), у которого в поле [*FullName*] установлено значение "Terrasoft.Configuration.CustomWebFormEventTargetPreProcessHandler, Terrasoft.Configuration", отключите признак [*Активный*] ([*isActive*]).
- С помощью SQL-запроса.
Чтобы отключить коробочный обработчик **с помощью SQL-запроса**, выполните следующий SQL-запрос.

SQL-запрос

```
UPDATE WebFormProcessHandlers
SET IsActive = 0
```

```
WHERE FullClassName = 'Terrasoft.Configuration.WebFormEventTargetPreProcessHandler, Terrasoft
```

Результат выполнения примера

Чтобы **посмотреть результат выполнения примера**, перезапустите пул приложения.

В результате новый контакт создается при отправке формы с заполненными обязательными полями, среди которых присутствует поле `CustomRequiredTextColumn`.