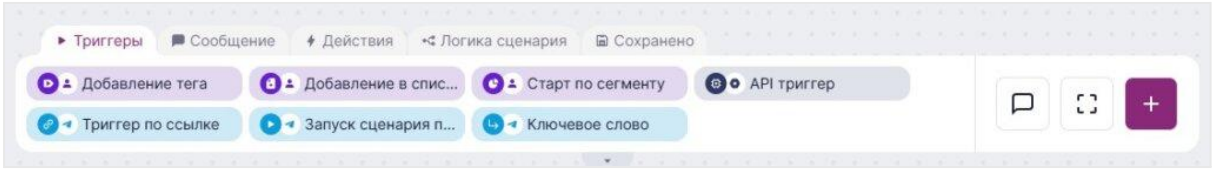
Использовать API-триггер в омниканальных сценариях

Блоки являются основой омниканальных сценариев. Они определяют, какие действия и в каком порядке будут происходить.

В редакторе доступно пять категорий блоков:

- Триггеры.
- Сообщение.
- Действия.
- Логика сценария.
- Сохранено.



Alt: Блоки триггеров.

В этой статье подробно разберем блок триггер «API-триггер» из категории «Триггеры».

Данный триггер позволяет запускать сценарии из внешней системы, например сайта или приложения, и отправлять клиентам письма сразу после оформления заказа, оплаты тарифа или записи на услугу.

В омниканальных сценариях Unisender можно быстро реагировать на действия клиента на сайте, в приложении или в CRM и автоматически отправлять нужные письма. Например:

- **после регистрации** — отправить welcome-цепочку или запустить онбординг;
- **после покупки** — подтвердить заказ, указать детали доставки или запросить отзыв;
- **если клиент оставил брошенную корзину** — напомнить о товарах и предложить промокод;
- **при изменениях в заказе** — уведомить о доставке или технической ошибке.

Такие рассылки помогают интернет-магазинам, онлайн-школам и другим бизнесам поддерживать связь с клиентами на каждом этапе воронки — от первого касания до повторных продаж.

Чтобы запускать рассылки по событиям из внешних систем, используйте API-триггер. В статье расскажем, как с ним работать.

Принцип работы API-триггера

Когда вы добавите в сценарий API-триггер, Unisender сгенерирует для него уникальный URL. Этот адрес нужно указать в настройках внешней системы, из которой хотите получать данные.

После запуска сценария все будет происходить автоматически. Как только клиент заполнит форму или оформит заказ, внешняя система отправит в Unisender его данные. Это может быть email, телефон, Telegram ID, список товаров, сумма заказа и другие параметры в формате JSON.

Если контакт с таким email или телефоном уже есть в базе, Unisender добавит его в сценарий. Если нет — создаст новый и запустит коммуникацию.

Внутри сценария вы сможете использовать переданную информацию и подставлять данные клиента в письма с помощью полей из сценария или шаблонизатора Liquid.

Как настроить API-триггер

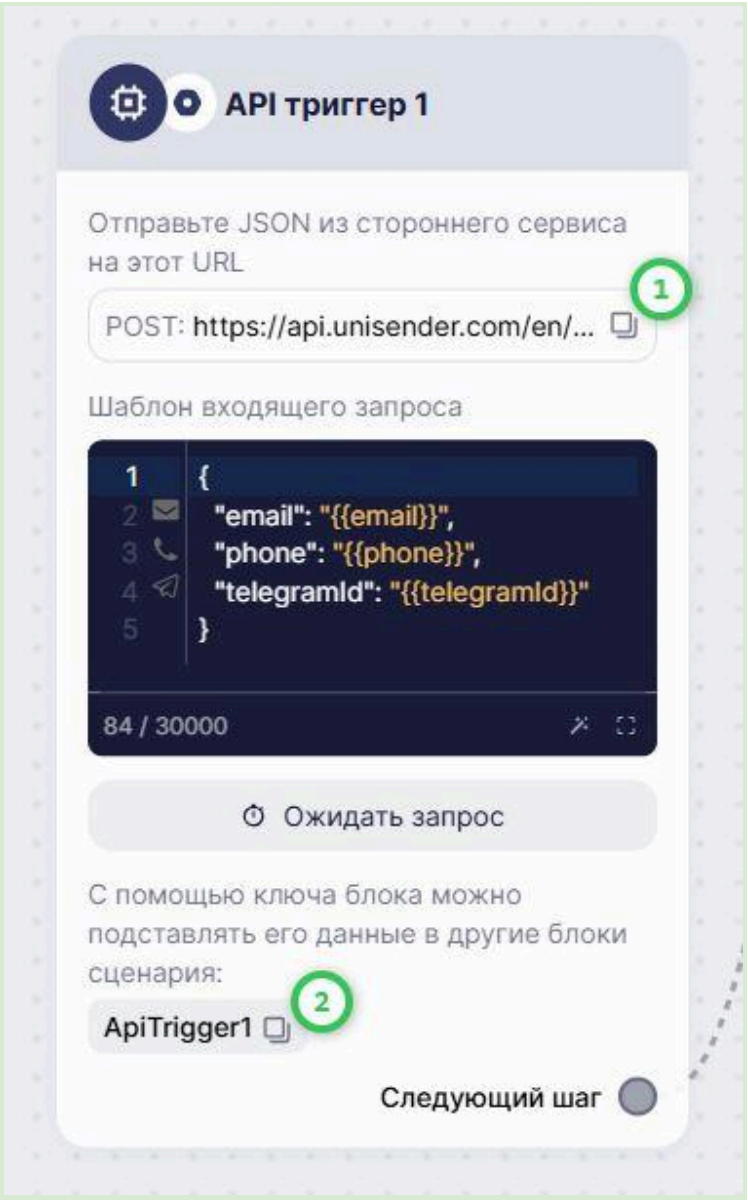
Чтобы запустить омниканальный сценарий по внешнему событию, настройте API-триггер в редакторе сценариев и подключите интеграцию с внешним сервисом.

Создание сценария

Перейдите в раздел «Сценарии», создайте новый сценарий и добавьте на рабочую область редактора блок «API триггер» из категории «Триггеры».

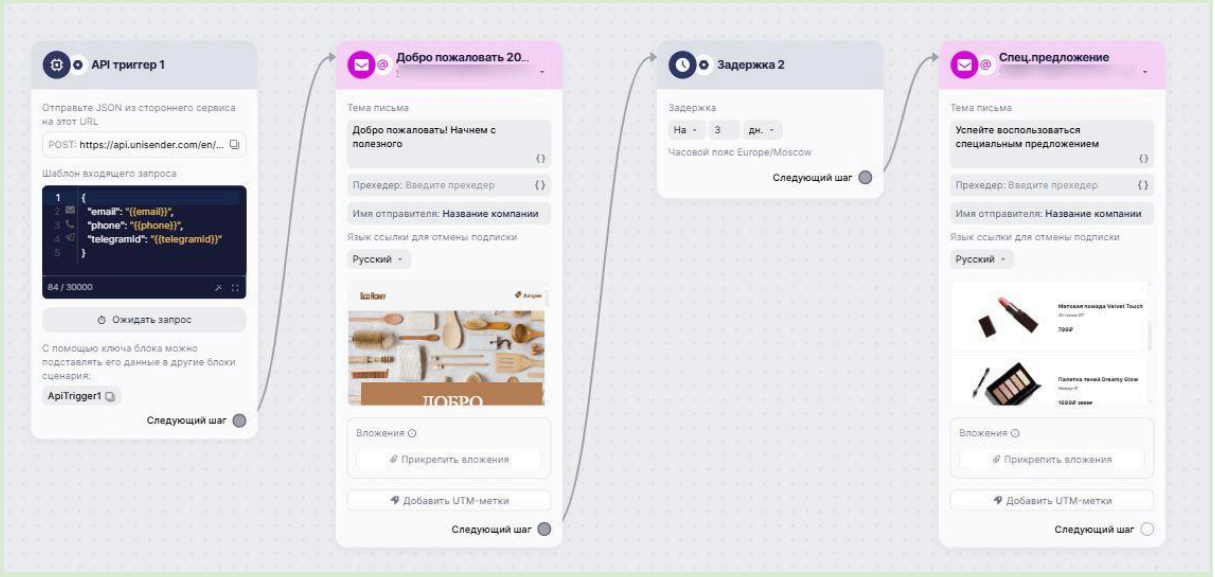
В настройках блока вы можете:

1. **Скопировать URL-адрес.** Он понадобится для настройки интеграции с внешним сервисом.
2. **Скопировать или изменить уникальный ключ блока.** Он потребуется, если вы планируете использовать передаваемые данные в письмах.



Alt: Настройки блока «API триггер».

Далее соберите остальную часть сценария с помощью готовых блоков и добавьте нужные письма, чтобы выстроить логику коммуникации. Например, можно настроить приветственную цепочку для новых клиентов. Сначала сценарий отправит приветственное письмо, а через сутки — второе письмо со специальным предложением.



Alt: Пример welcome-сценария.

Когда все будет готово, запустите сценарий. Без этого данные, передаваемые по API, не будут обрабатываться.

Настройка интеграции

Если вы используете конструктор сайтов или платформу с поддержкой вебхуков, интеграцию можно настроить самостоятельно через интерфейс личного кабинета. Обычно для этого нужно прописать URL, сгенерированный в блоке «API-триггер», и задать, какие данные передавать. В запросе обязательно должен присутствовать email или телефон, иначе Unisender не сможет определить, кому отправлять сообщения.

На примере Tilda это можно сделать через настройки сайта, следуя инструкции. После создания интеграции данные клиентов будут автоматически передаваться в Unisender и запускать сценарий.

Если вы используете собственный сайт или приложение, потребуется помощь разработчиков. Нужно настроить интеграцию по API, чтобы отправлять POST-запросы на URL, сгенерированный в блоке «API-триггер».

Тело запроса должно быть в формате JSON и содержать данные, которые вы планируете использовать в сценарии (например, email, имя клиента, список товаров или сумму заказа).

После настройки интеграции отправьте тестовый запрос (вебхук) из вашего сайта, приложения или CRM. Это поможет убедиться, что:

- URL блока корректен;
- данные принимаются в правильном формате.

Важно! В теле запроса обязательно должен присутствовать email или телефон. Эти данные Unisender будет использовать для идентификации получателей.

Пример запроса:

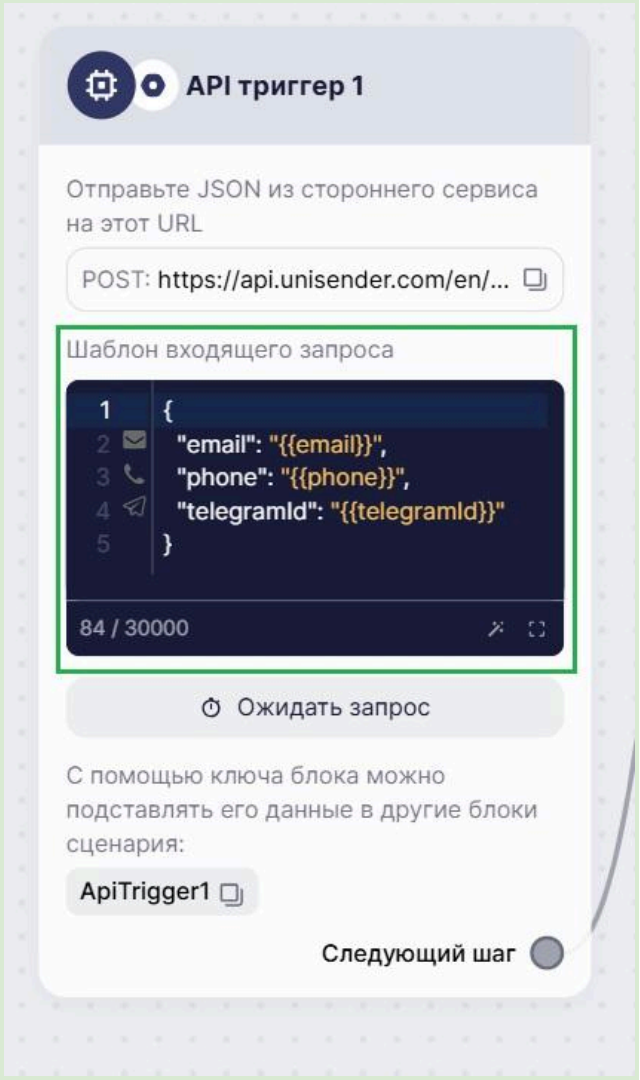
```
POST https://api.unisender.ru/ru/api/triggerBlock/{blockId}
Content-Type: application/json


{
  "email": "hello@example.com",
  "products": [
    {
      "name": "Кроссовки",
      "link": "https://example.com/link1"
    },
    {
      "name": "Футболка",
      "link": "https://example.com/link2"
    }
  ]
}
```

Сопоставление полей контакта

В блоке «API-триггер» с помощью модуля «Шаблон входящего запроса», можно указать, в каком именно поле JSON-запроса находится email, телефон или Telegram ID контакта — без изменения формата запроса на стороне внешней системы.

Модуль представляет собой редактор-кода. По умолчанию в нем уже указан заготовленный шаблон с полями для email, телефона и Telegram ID.

~~Если вы уже используете блок «API-триггер» без настройки сопоставления, он будет работать и после обновления. Новая версия сохраняет совместимость с ранее настроенными интеграциями.~~

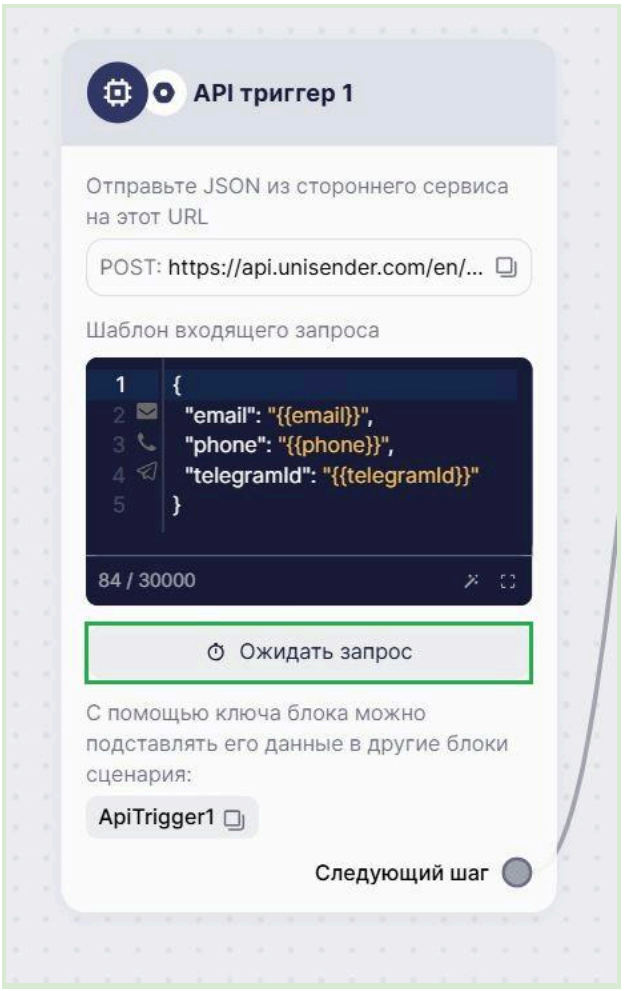


Alt: Модуль «Шаблон входящего запроса» в блоке «API триггер».

Чтобы упростить настройку, вы можете автоматически получить пример реального запроса из вашей внешней системы и сопоставить его поля. ~~и разметить его.~~

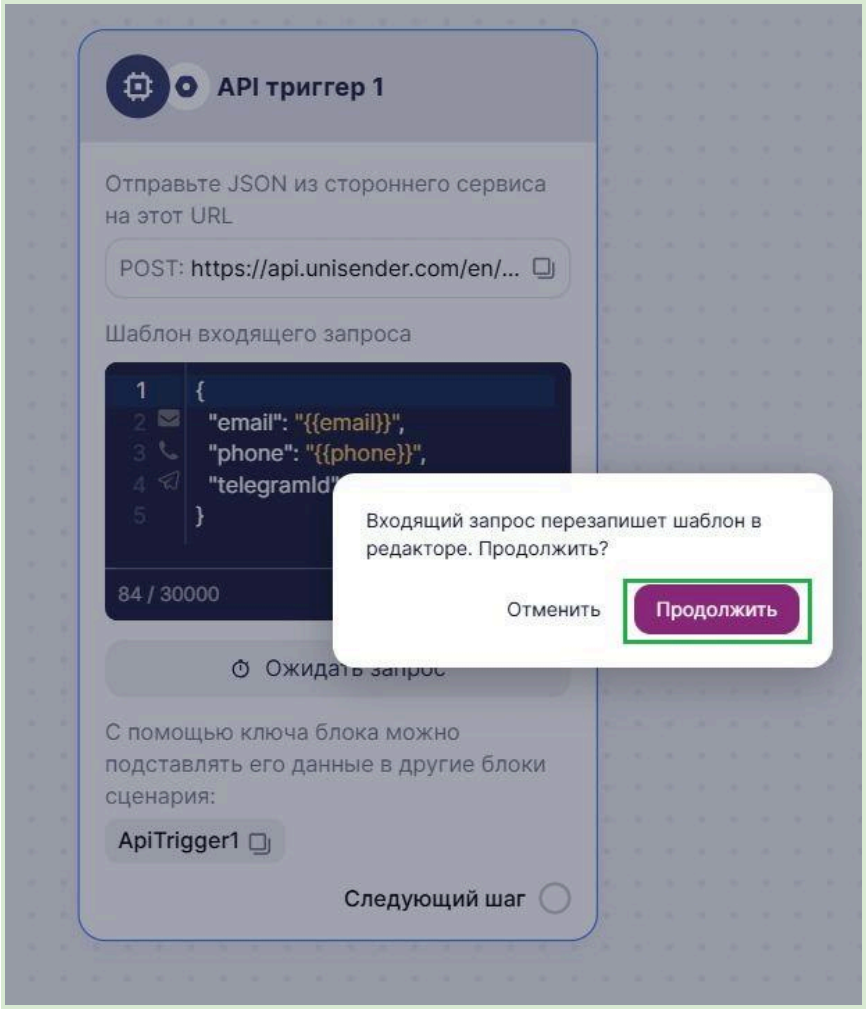
Получение примера запроса:

- Нажмите на кнопку «Ожидать запрос».



Alt: Как получить пример запроса в блоке «API триггер».

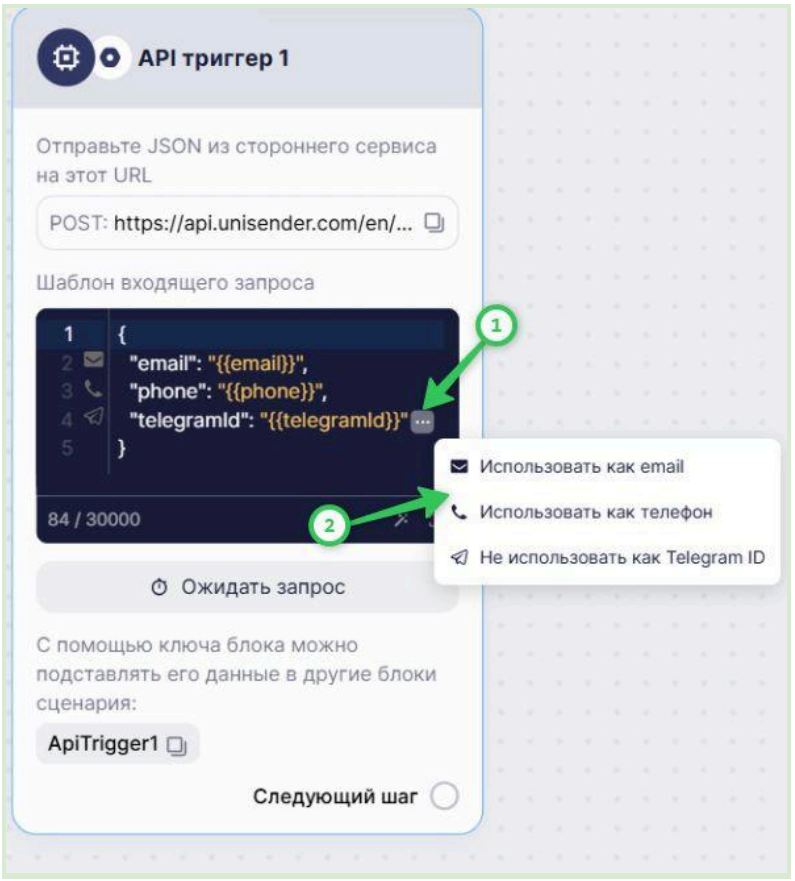
- Нажмите «Продолжить», чтобы подтвердить, что при поступлении запроса шаблон в редакторе будет перезаписан входящими данными.



Alt: Как подтвердить отправку тестового запроса.

Сопоставление полей:

- После того, как данные отобразятся в редакторе, отметьте нужные строки как идентификаторы контакта.
- Поставьте курсор в строку со значением, которое хотите использовать как email, телефон или Telegram ID. Строка подсветится, а справа появится кнопка с тремя точками.
- Нажмите на эту кнопку и выберите нужное действие из меню: «Использовать как email», «Использовать как телефон» или «Использовать (Не использовать) как Telegram ID».



Alt: Как сопоставить поля в блоке «API триггер».

После выбора значение в строке заменится на системную подстановку: {{email}}, {{phone}} или {{telegramId}}.

Важно:

Чтобы система могла определить контакт, в шаблоне обязательно должно быть отмечено хотя бы одно поле: email, телефон или Telegram ID. Указать можно несколько полей, но не более одного поля каждого типа (например, нельзя отметить два разных поля как {{email}}).

Отменить сопоставление можно тем же способом. Нажмите на три точки рядом с подстановкой и отмените выбор.

Ответы сервера

При успешной отправке запроса вы получите следующий ответ:

```
{
```

```
  "success": true
}
```

Если в теле запроса не указан ни email, ни телефон, сервер вернет ошибку:

```
{
  "error": "Error ID:... Can't define contact",
  "code": "invalid_arg"
}
```

Если в URL указан некорректный идентификатор блока, сервер вернет ошибку:

```
{
  "error": "Error ID:... Invalid UUID string",
  "code": "invalid_arg"
}
```

Как подставлять данные в письма

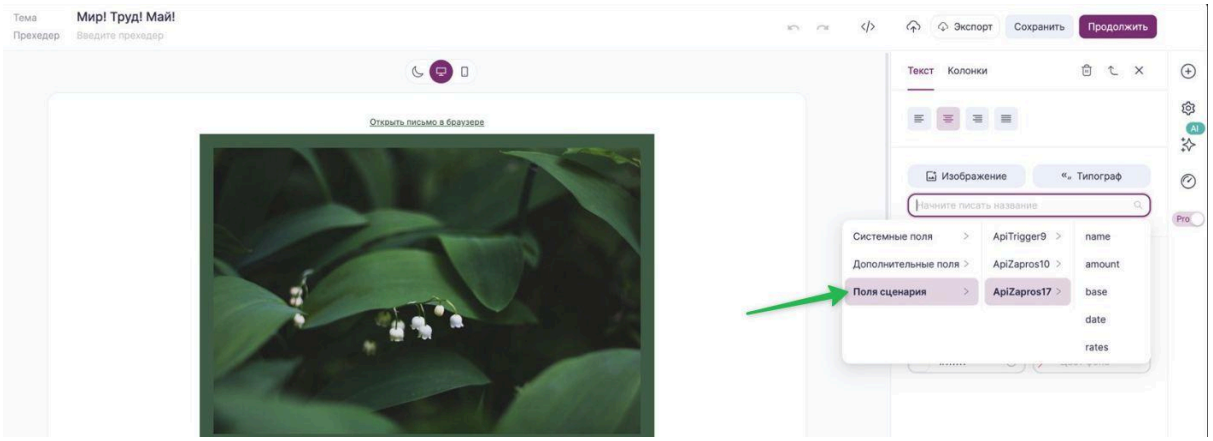
Если вы хотите персонализировать письмо, например обратиться к клиенту по имени, добавить список заказанных товаров или дату доставки, используйте данные из JSON-запроса.

Подстановка переменных в редакторе

Переменную можно подставить в:

- теме письма;
- прехедере;
- теле письма (в текстовых блоках);
- email-адресе получателя.

Чтобы добавить переменную, поставьте курсор в то место письма, куда хотите добавить подстановку. В настройках текста нажмите «Переменные», выберите тип поля сценария и найдите нужную переменную в списке.



Alt: Как добавить переменную из поля сценария.

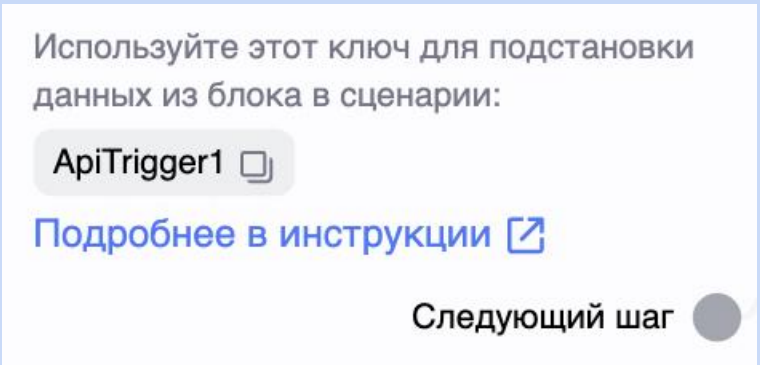
Шаблонизатор Liquid

Для сложной персонализации используйте шаблонизатор Liquid. Это язык шаблонов, с помощью которого можно подставлять значения из JSON-запроса в тело или тему письма.

Liquid использует три основные конструкции:

1. Переменные. Это значения, которые вы передаете в JSON-запросе. Чтобы вывести их в письме, используйте двойные фигурные скобки {{ }}. Также можно выбрать поле в редакторе способом выше, и система автоматически добавит переменную.

Важно! Обязательно указывайте уникальный ключ блока перед переменной, чтобы система понимала, откуда брать данные.



Alt: Пример ключа в настройках блока «API триггер».

Пример JSON-запроса	Шаблон в HTML-письме	Что увидит получатель

{
"email": "hello@example.com",
"name": "Анна"
}
	Здравствуйте, {{ ApiTrigger1.name }}!
	Ваш email: {{ ApiTrigger1.email }}
	
	Посмотреть, как это работает
		Здравствуйте, Анна!
		Ваш email: hello@example.com

2. Фильтры. С их помощью можно форматировать входящие данные. Например, преобразовать имя с маленькой буквы в заглавную.

Фильтры пишутся после переменной через вертикальную черту |.

Фильтр	Пример JSON-запроса	Шаблон в HTML-письме	Что увидит получатель
capitalize — делает первую букву строки заглавной	{ "name": "анна" }	Привет, {{ ApiTrigger1.name | capitalize }}!	Привет, Анна!
upcase — переводит строку в верхний регистр	{ "promo": "лето2025" }	Ваш промокод: {{ ApiTrigger1.promo | upcase }}	Ваш промокод: ЛЕТО2025
downcase — переводит строку в нижний регистр	{ "promo": "ЛЕТО2025" }	Ваш промокод:{{ ApiTrigger1.promo | downcase }}	Ваш промокод: лето2025
replace — заменяет часть переменной на другую	{ "phone": "+7 (999) 123-45-67" }	Телефон: {{ ApiTrigger1.phone | replace: "+7", "8" }}	Телефон: 8 (999) 123-45-67
default — подставляет значение по умолчанию, если переменная пустая	{ "name": "" }	Привет, {{ ApiTrigger1.name | default: "клиент" }}!	Привет, клиент!
date — форматирует дату	{ "delivery_date": "2025-05-01" }	Дата доставки: {{ ApiTrigger1.delivery_date | date: "%d.%m.%Y" }}	Дата доставки: 01.05.2025

3. Операторы. Позволяют выполнять условия и работать с массивами. Например, выводить списки товаров из брошенной корзины с помощью цикла for или проверять условия через if.

Обратите внимание:

Операторы указываются в одинарных фигурных скобках, а передаваемые поля — в двойных.

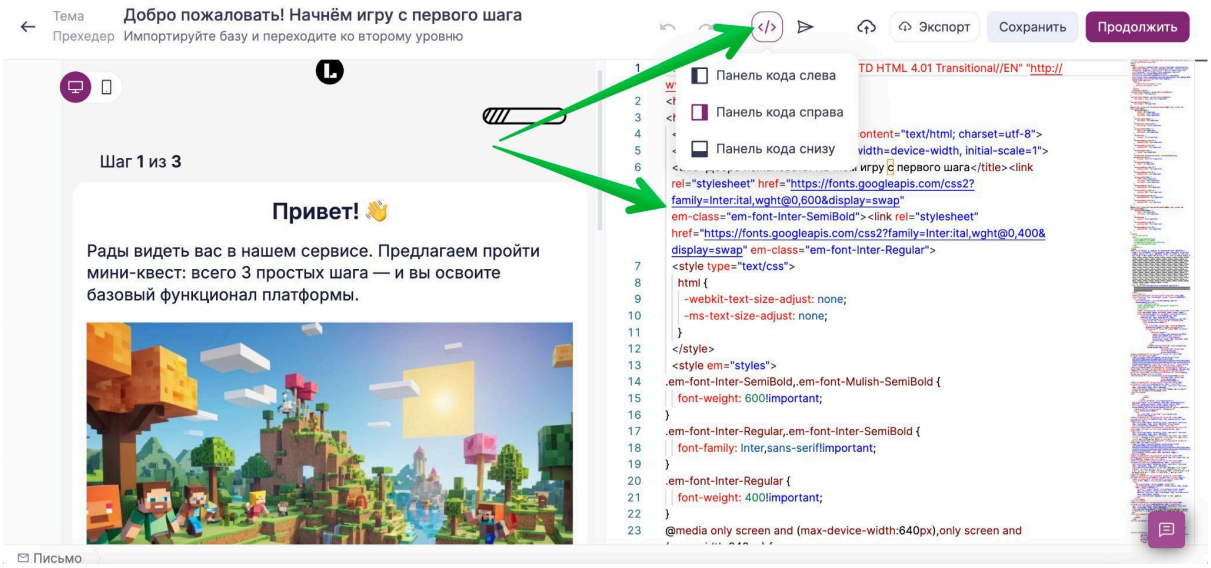
Пример JSON-запроса	Шаблон в HTML-письме	Что увидит получатель

{
  "email":
"anna@example.com",
  "products": [
  {
  "name": "Кроссовки",
  "link":
"https://example.com/shoes"
  },
  {
  "name": "Футболка",
  "link":
"https://example.com/tshirt"
  }
  ]
}
	Привет, {{ ApiTrigger1.email }}!
	Вы забыли в корзине:
	{% for product in ApiTrigger1.products %}
	- {{ product.name }} — {{ product.link }}
	{% endfor %}
	
	Посмотреть, как это работает
		Привет, anna@example.com!
		Вы забыли в корзине:
		- Кроссовки — https://example.com/shoes
		- Футболка — https://example.com/tshirt

Подробно про шаблонизатор Liquid можно прочитать в документации (на русском языке).

Ознакомиться с использованием JSON-данных в шаблонизации и протестировать функционал можно по этой ссылке.

Чтобы использовать значения из JSON-запроса, откройте письмо в блоке «Email» и переключитесь на режим кода.



Alt: Как переключиться на режим кода.

Вставьте шаблон подстановки в нужное место письма. Обязательно укажите уникальный ключ блока, иначе подстановка не сработает. Ключ можно скопировать или изменить в настройках блока «API-триггер».

Важно! Если вы измените уникальный ключ в настройках блока, не забудьте заменить его во всех подстановках, иначе нужные данные не будут попадать в письмо.

Alt: Как добавить подстановку в письмо.

Если возникли ошибки

Если сценарий не работает или данные не подставляются, проверьте, на каком этапе возникла проблема.

Ниже разберем основные причины и что с ними делать:

1. Ошибки интеграции. Если данные не приходят в Unisender или приходят с ошибками, нужно проверить интеграцию на стороне внешнего сервиса и убедиться, что:

- используется корректный URL из блока «API-триггер»;
- тело запроса содержит email или телефон;
- формат данных соответствует JSON.

Важно! У блока «API-триггер» нет собственных ошибок на стороне Unisender. Он просто принимает данные и запускает сценарий.

2. Ошибки сопоставления полей. Если в шаблоне есть ошибки, сценарий не сможет корректно определять контакты по входящим запросам.

Возможные ошибки:

Сообщение об ошибке	Когда возникает
Неверный формат JSON	Нарушен синтаксис JSON, не удалось распознать структуру
Для запуска сценария нужно задать в JSON поле с email или телефоном	Ни одно поле не отмечено как email, телефон или Telegram ID
Выбрано больше одного поля email. Пожалуйста, оставьте только одно	В шаблоне указано больше одной подстановки {{email}} (аналогичная ошибка выводится и для телефона, и для Telegram ID)
Пожалуйста, назначайте как телефон или email только начальные элементы массивов	Полем email, телефон или Telegram ID отмечен элемент массива, который не является первым на своем уровне вложенности

3. Ошибки в шаблоне письма. Если данные не подставляются в письмо или отображаются некорректно, необходимо убедиться, что:

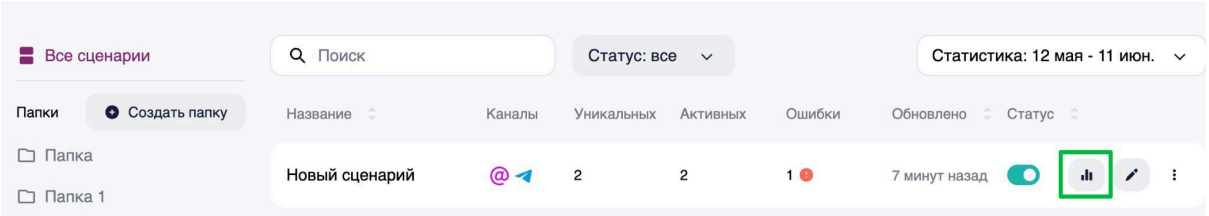
- правильно указаны переменные и ключ API-триггера;
- отсутствуют опечатки и лишние пробелы;
- корректно расставлены фигурные скобки и фильтры.

При необходимости можно свериться со справкой по Liquid.

4. Ошибки в сценарии. Если данные не приходят в Unisender или письма не отправляются, проверьте настройки сценария и убедитесь, что он запущен.

Откройте страницу с подробной статистикой. На ней отображается текущее состояние сценария, ключевые метрики (например, сколько контактов попало в сценарий и сколько писем отправлено), а также возможные ошибки.

Если в сценарии отсутствуют какие-то блоки или контент, скорее всего, вы не сохранили последние изменения. Вернитесь в редактор и обновите сценарий.



Alt: Как посмотреть подробную статистику сценария.

Если не удается разобраться самостоятельно, напишите в поддержку — поможем.

Полезные ссылки

Введение в омниканальную автоматизацию
Как создать и запустить омниканальный сценарий
Как использовать готовые блоки для создания сценария
Документация по Liquid