# События сервера UIDM

Оглавление

# Общая информация
  # Атрибуты
  # Генерация события
# Публикация события
# Категории событий
  # Событие auth-success
  # Событие auth-success.impersonate
  # Событие com.rooxteam.sso.provisioning.principal.post
  # Событие com.rooxteam.sso.provisioning.blocks.post
  # Событие com.rooxteam.sso.provisioning.blocks.delete
  # Событие com.rooxteam.sso.oauth.token-expired
  # Событие com.rooxteam.sso.oauth1.token-revoked
  # Событие com.rooxteam.sso.oauth2.token-revoked
  # Событие com.rooxteam.sso.oauth2.invalidate-token

---

## # Общая информация

Объект типа com.rooxteam.webapi.event.Event генерируется при наступлении определенных событий в SSO-server и публикуется используя стандартные механизмы Spring.

В дальнейшем для использования этих событий можно создавать бины-слушатели, которые будут отлавливать сообщения нужной категории и выполнять заданную бизнес-логику.

### Атрибуты

Класс `com.rooxteam.webapi.event.Event` имеет ряд базовых атрибутов, а также `Map<String, Object> parameters` для кастомных атрибутов.

### Базовые атрибуты

- String category — категория (тип) события
- String clientId — OAuth ClientId
- String principalId — идентификатор принципала
- String publishUri
- boolean async — флаг асинхронной обработки

- boolean forwardable

### Генерация события

Для создания событий используется экземпляр класса с интерфейсом `com.rooxteam.webapi.event.EventFactory`. Обычно это инициированный как бин `com.rooxteam.webapi.event.impl.EventFactoryImpl`.

Используя метод `builder` класса `EventFactoryImpl` создаётся `Event.EventBuilder`, в который уже добавлены несколько атрибутов по-умолчанию.

Таблица 1. Атрибуты по умолчанию, инициируемые `EventFactoryImpl`

| Атрибут | Описание | Откуда данные |
|---|---|---|
| ip | IP адрес из запроса | Из MDC |
| userAgent | userAgent из запроса | Из MDC |

## # Публикация события

События публикуются через стандартный интерфейс Spring Framework `org.springframework.context.ApplicationEventPublisher`.

Пример генерации и публикации события

```
@Autowired
private EventFactory eventFactory;

@Autowired
private ApplicationEventPublisher applicationEventPublisher;

...

Event.EventBuilder builder = eventFactory.builder().category(eventCategory)
        .publishUri("urn:rooxteam:event:auth/success")
        .clientId(clientId)
        .principalId(userId)
        .put("method", authMethod);
applicationEventPublisher.publishEvent(builder.build());
```

## # Категории событий

### Событие `auth-success`

Генерируется после успешной аутентификации пользователя.

Метод в котором генерируется событие: `com.rooxteam.sso.webflow.CommonLoginFlow#notifyAuthSuccess`.

Вызов метода происходит из webflow по окончании аутентификации пользователя, как правило рядом с аналогичным методом аудита.

Таблица 2. Дополнительные атрибуты события auth-success

| Атрибут | Описание | Откуда данные |
|---|---|---|
| method | Метод аутентификации | Из FlowScope |
| module | Модуль аутентификации | Из authState |
| msisdn | Номер телефона пользователя | передаётся как параметр метода |
| user_id | Идентификатор пользователя | передаётся как параметр метода |
| realm | realm | передаётся как параметр метода |
| claims | claims | Из authState |
| customer_id | Customer Id | Из Claims |
| executionId | Идентификатор execution, используется как идентификатор сессии | Из Claims |
| geoLocation | Данные геолокации пользователя | Получают из параметров запроса |
| applicationInstallID | Идентификатор версии установленного приложения | из параметров запроса |
| fingerprint | Хеш отпечатка браузера пользователя | из параметров запроса |
| fingerprintData | Структура данных по которым собирался fingerprint, urlencoded JSON string | из параметров запроса |
| x-client-type | Содержимое заголовка x-client-type | из параметров запроса |

## Событие `auth-success.impersonate`

Генерируется после успешной аутентификации пользователя через флоу impersonate-auth.

Метод в котором генерируется событие: `com.rooxteam.sso.webflow.CommonLoginFlow#notifyAuthSuccess`.

Вызов метода происходит из webflow impersonate-auth.xml по окончании аутентификации.

## Событие `com.rooxteam.sso.provisioning.principal.post`

Генерируется после успешной регистрации пользователя через provisioning API.

Метод в котором генерируется событие:
`com.rooxteam.sso.provisioning.service.SsoProvisionPrincipalService#buildCreatePrincipalEvent`

Событие отправляется при вызове POST запроса на API `/sso/provisioning/principals`

### Событие `com.rooxteam.sso.provisioning.blocks.post`

Генерируется после блокировки пользователя через provisioning API.

Метод в котором генерируется событие:
`com.rooxteam.webapi.services.impl.persistence.PersistenceBlockService#buildPostBlockEvent`

Событие отправляется при вызове POST запроса на API `/provisioning/block`

### Событие `com.rooxteam.sso.provisioning.blocks.delete`

Генерируется после снятия блокировки пользователя через provisioning API.

Метод в котором генерируется событие:
`com.rooxteam.webapi.services.impl.persistence.PersistenceBlockService#buildDeleteBlockEvent`

Событие отправляется при вызове DELETE запроса на API `/provisioning/block`

### Событие `com.rooxteam.sso.oauth.token-expired`

Генерируется после истечения срока токена пользователя.

Метод в котором генерируется событие:
`org.forgerock.openam.cts.reaper.CTSReaper#buildPostLogoutEvent`

### Событие `com.rooxteam.sso.oauth1.token-revoked`

Генерируется при запросе на отзыв oauth1 токена пользователя.

Метод в котором генерируется событие:
`com.rooxteam.sso.servlet.RevocationServlet#invalidateOAuth1AccessToken`

Событие отправляется при запросе на отзыв токена через POST запрос на `/oauth2/revoke`

### Событие `com.rooxteam.sso.oauth2.token-revoked`

Генерируется при запросе на отзыв oauth2 токена пользователя.

Метод в котором генерируется событие:
`com.rooxteam.sso.servlet.RevocationServlet#notifyOauth2TokenRevocation`

Событие отправляется при запросе на отзыв токена через POST запрос на `/oauth2/revoke`

### Событие `com.rooxteam.sso.oauth2.invalidate-token`

Генерируется при инвалидации токена при логауте.

Метод в котором генерируется событие: `com.rooxteam.sso.filter.LogoutFilter#doFilter`

Событие отправляется при запросе на `/UI/Logout`