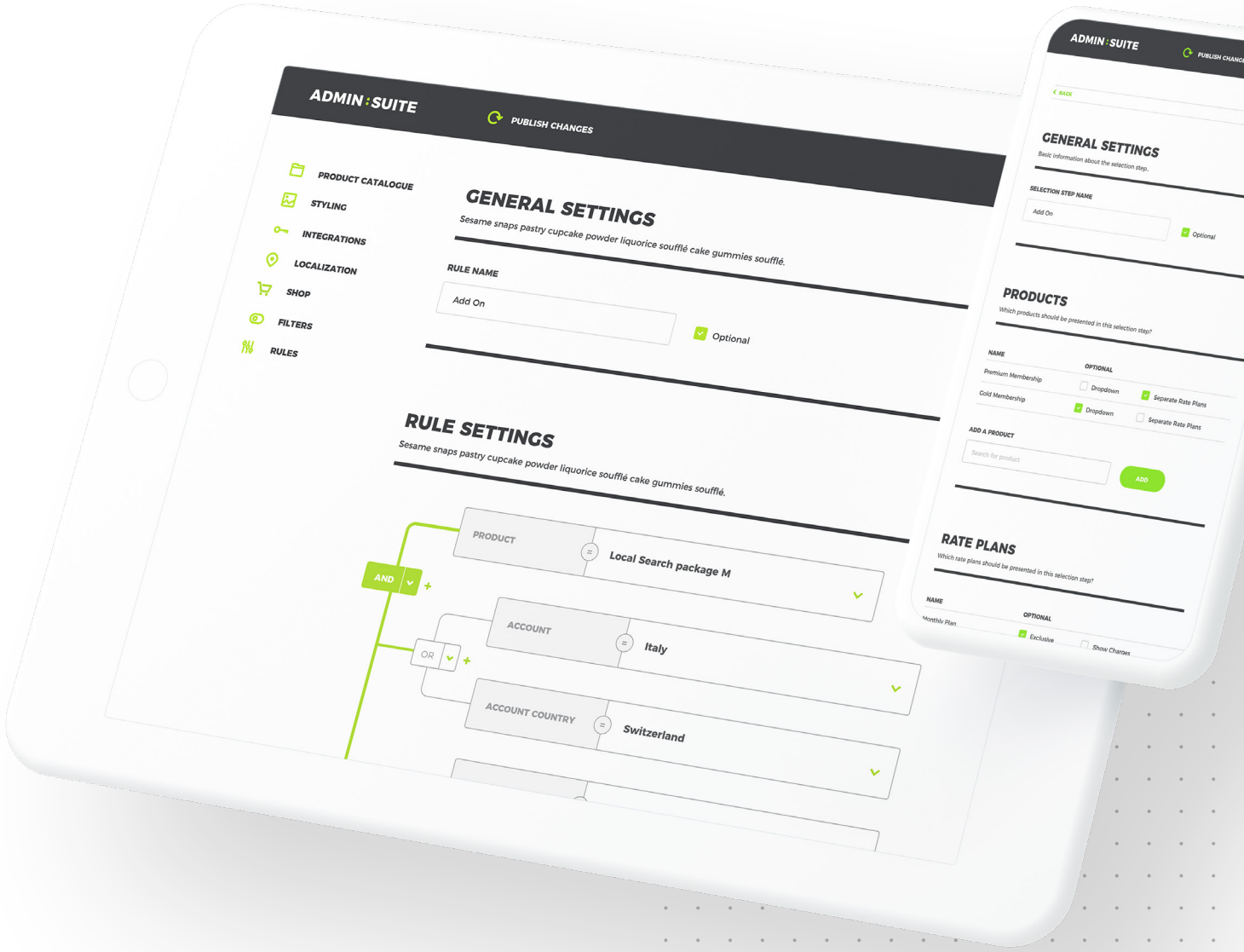
>keylight/
subscription suite



Catalog

Subscription Suite 2.0 | Version 1.1.0

Table of **contents/**

1. Introduction
1.1 In a nutshell

2. Data Model
2.1 Structure
2.2 Product Hierarchy
2.3 AddOns
2.4 Upgrades & Downgrades

3. Catalog & Rules API
3.1 Overview
3.2 Authentication & Authorization
3.3 Resources
3.4 Rules endpoints
3.5 Overview technical fields

4. Contact
4.1 General Inquiries

1.1 In a nutshell

Chapter 1.

Introduction

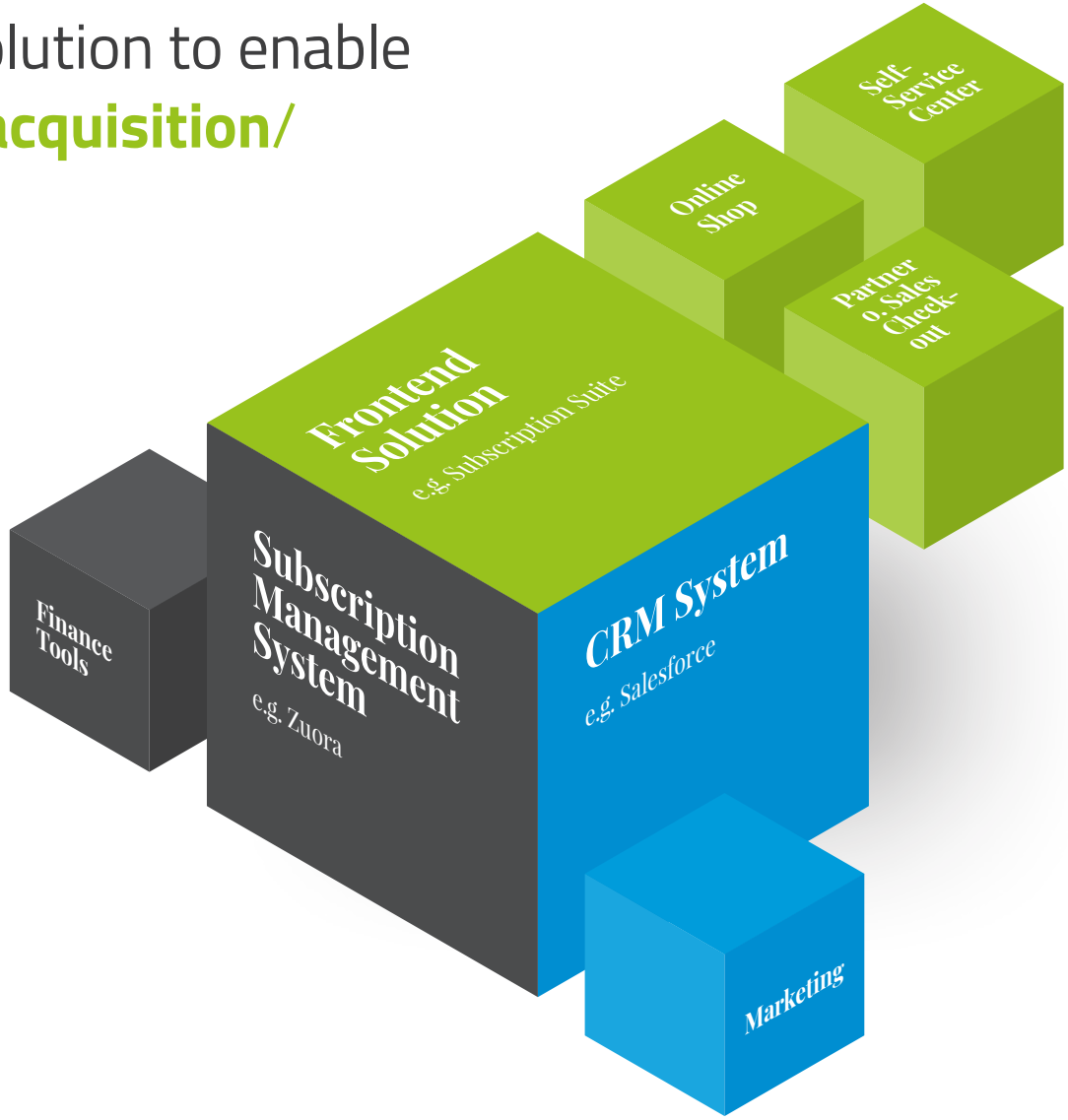
1.1 In a nutshell

A new class of software solution to enable and automate **subscriber acquisition/**

The subscription suite is a flexible self-service solution for subscription enterprises. It is well-integrated into the Zuora and Salesforce ecosystem.

The subscription suite utilizes the Zuora product catalog as the basis for subscription offerings. It extends the Zuora catalog in order to overcome its lack of workflows, rules and validation options.

This document describes the subscription suite catalog data model and related API endpoints.



2.1 Overview
2.2 Structure
2.3 Product Hierarchy
2.4 AddOns
2.5 Upgrades & Downgrades

Chapter 2.

Data Model

2.1 Structure

Our catalog data model is **independent** from the Zuora catalog/

The subscription suite catalog data model is independent from the Zuora catalog in order to provide additional functionality and to model dependencies between products. The structure described below enables the subscription suite to flexibly adjust functionality and to provide a high-level data model above all the technical details Zuora covers in its model.

At its core the suite catalog model maintains Suite Products. These Suite Products specify products a customer can buy, or a reseller can book for his customers. It specifies subscription terms, upgrade / downgrade behavior and many other topics. The Zuora rate plans are connected to the subscription suite catalog through the Suite Product Rate Plan model which allows to specify a list of Zuora RatePlans as a dependency. This flexible dependency allows e.g. to specify product bundles. The following list describes the main objects:

Suite Product

The suite product object represents the subscription suite object for representing subscribable packages. It's extending the Zuora product and product rate plan with further configuration options and constraints.

Suite Product Rate Plan

The suite product rate plan connects the subscription suite product with the Zuora catalog objects.

Suite Product Category

The suite product category object represents the subscription suite object for building product hierarchies. It supports multi inheritance and specifies a complex graph which is used for upgrade / downgrade handlings and replacements.

2.1 Structure

Suite Product Change Group

The suite product change group object represents the subscription suite object for a list of interchangeable products. Therefore, it defines valid upgrade and downgrade paths.The following chapter will go into details and provide examples on how to use

Suite Product Replacement Group

The suite product replacement group object represents the subscription suite object for a list of replacable products. In case of an up- / downgrade of a higher level product these groups are checked for e.g. replacing addons accordingly.

The following chapter will go into details and provide examples on how to use these entities.

2.2 Product Hierarchy

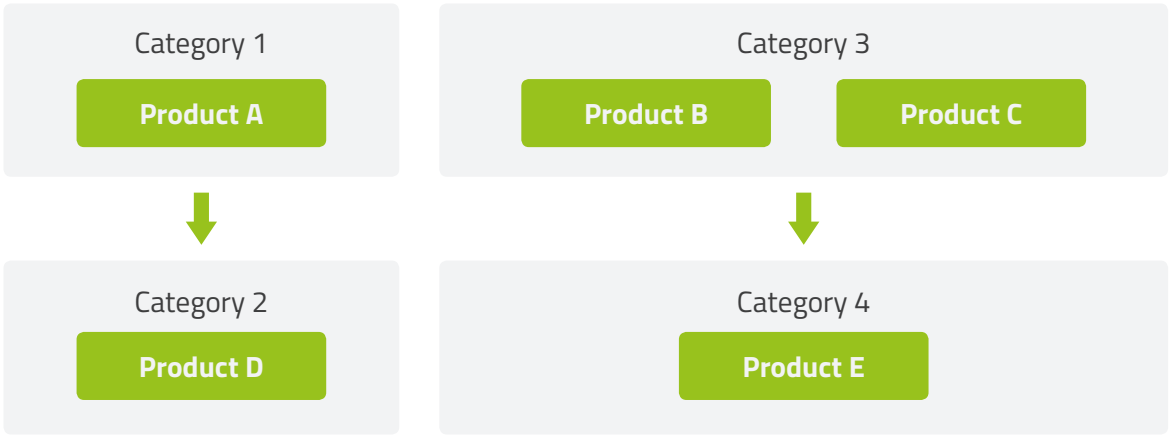
The product catalog makes use of **categories/**

Please note: *The described flexibility offers a lot of potential to break the graph dependencies. You can specify for example circular dependencies. We will validate the graph integrity in the future. For now, you need to take care.*

The subscription suite product catalog makes use of categories. The categories consist of labels and specify relations to other categories. Based on this, the subscription suite maintains a complex category graph which can get resolved in order to identify dependencies between categories.

Each SuiteProduct is connected to N categories. Through this link the product catalog specifies a dependency graph. These connections are later on used for managing upgrades / downgrades and product suggestions. If for example a base product is cancelled, all depending products get cancelled as well.

In this example have four categories. There are two top level categories (Cat 1 + 3) and two categories which depend on them. Within these categories there are products. All products in Cat 2 depend on a product in Cat 1. That means that you can only subscribe to a Cat 2 product in case you have already a Cat 1 product or you are subscribing to it as well in the same step. The same applies to the relation between Cat 3 and Cat 4 products. These category graphs provide a lot of flexibility – especially as you can place suite products in N categories.



2.3 AddOns

AddOns are **attached** to other suite products/

Addons are normal suite products which are attached to other suite products as an AddOn. The AddOn concept is used for specifying additional products which customers could buy as an addition.

As these additions are normal suite products, the whole catalog structure and feature set applies to them as well. This means that dependencies are automatically checked in order to only offer AddOns a customer can actually buy based on his current setup. Based on this, all side effects on subscription adjustments are resolved as well.

Each suite product can have N AddOns. The subscription suite AddOn concept is not related to Zuora add-on functionality.

Within the subscription suite frontend these AddOns are used to specify cross and upselling path to customers. A subscriber can use them in order to adjust his subscriptions. The AddOns functionality offers a flexible way for specifying valid adjustment paths.

2.4 Upgrades & Downgrades

Upgrades and downgrades **require relation** between different products or services/

Offering upgrades and downgrades require initially a relation between different products / services. This relation needs to specify which products would be counted as an upgrade and which as a downgrade. On top of this, downgrades are normally not allowed or only under special circumstances. When looking at product hierarchies (e.g. add-ons) these adjustments need to cascade in order to ensure that the overall subscription status remains valid. In order to specify these upgrade and downgrade paths the subscription suite catalog offers a couple of concepts:

- **Change Groups**
- **Replacement Groups**
- **Product hierarchies (see 2.1)**
- **AddOns (see 2.2)**

Change Groups

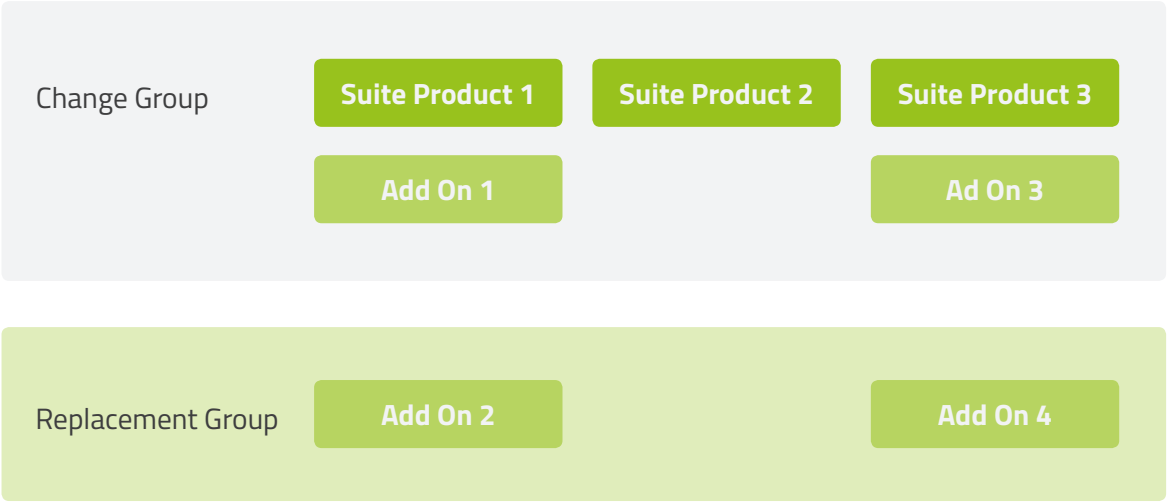
They specify groups of suite products which are interchangeable. The change group itself specifies further rules. It's possible for example to restrict downgrades or to specify how same level suite product changes are handled (downgrade or upgrade). Each change group can contain N suite proudcts.

2.4 Upgrades & Downgrades

Each suite product within this change group comes with a priority value in order to sort and compare them. In case the priority of the current suite product is lower than the new one - the change - is counted as an upgrade.

Replacement groups

They specify which suite products can be used as replacements. The general structure is the same as for change group, but in this case it's about handling cascading adjustments. Replacement groups work in combination with add-ons which you specify for each suite product. In case of an upgrade the subscription suite checks for each depending suite product if there is an add-on available which is in the same replacement group as the current add-on and if one of these replacements is specified in the new suite product's add-ons list:



2.4 Upgrades & Downgrades

In the example above a change from suite product 1 to suite product 2 would be counted as an upgrade. The dependent add-ons (AddOn1 and AddOn2) would need to change as well. For AddOn2 is a replacement group specified.

Therefore, the subscription suite would offer an upgrade to AddOn 4. For AddOn 1 is no replacement group specified. The subscription suite would require a cancellation. The final action would look the following:

- **Upgrade “suite product 1” to “suite product 3”**
- **Cancel AddOn 1**
- **Replace AddOn 2 with AddOn 4**

After applying these changes, the overall subscription would be in a valid state again.

3.1 Overview
3.2 Authentication / Authorization
3.3 Resources
3.4 Rules endpoints
3.5 Overview technical fields

Chapter 3.

Catalog & Rules API

3.1 Overview

Our **server to server** API can be used directly for different use cases/

The subscription suite offers a server to server API which can get used directly for different use cases. It makes use of the authentication protocol OAuth2 (**https://oauth.net/2/**) for authenticated access. There are monitoring and rate limits in place.

The API follows the RESTful API pattern. It's organized around resources and offers operations which are derived from the underlying HTTP protocol:

- **Get** ..... for fetching resources
- **Post** ..... for creating resources
- **Put** ..... for updating resources
- **Delete** ..... for deleting resources

Due to this the API is easy to implement and to understand.

3.2 Authentication & Authorization

We use industry standard **OAuth2 protocol** for authentication/

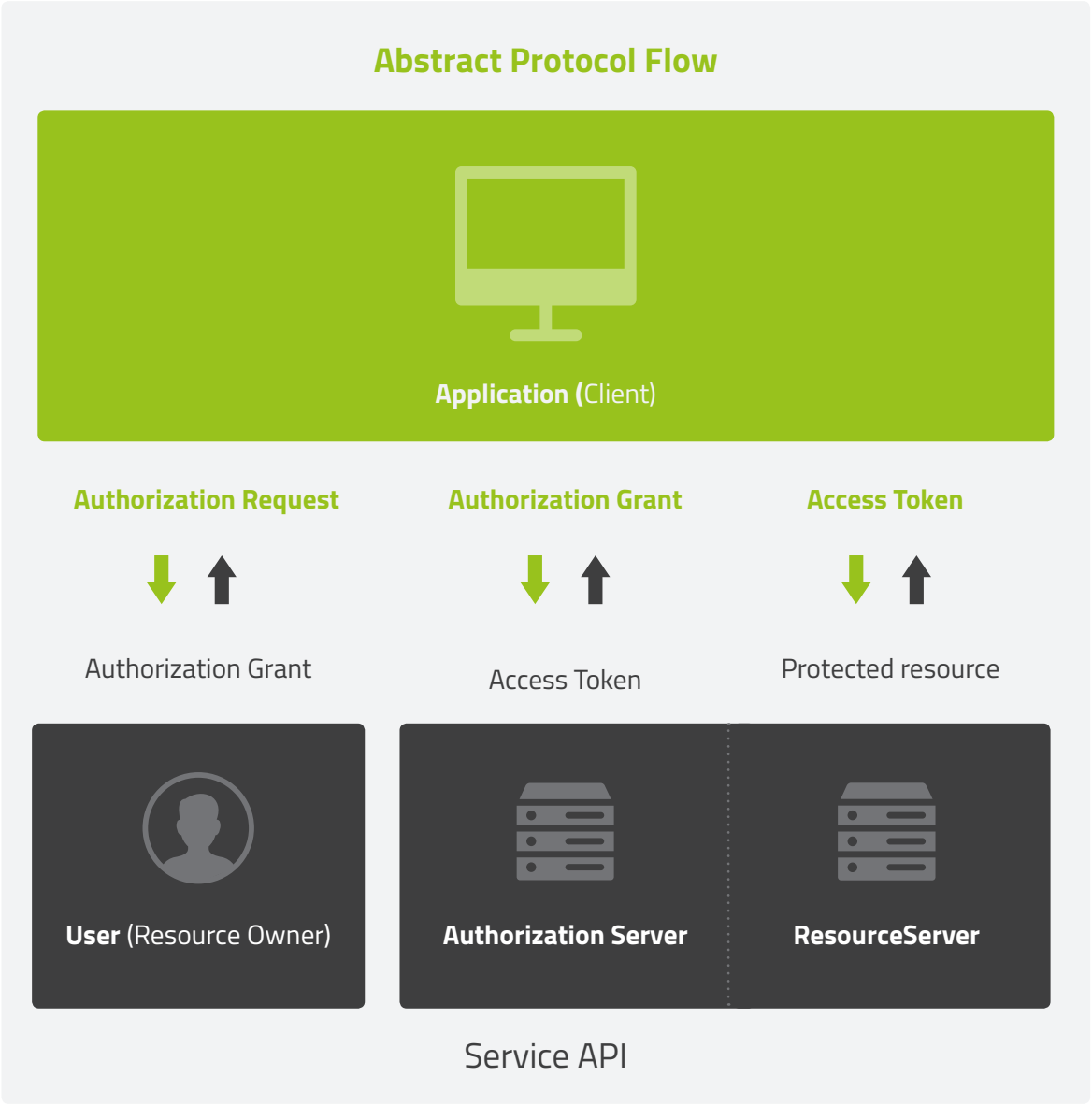
The subscription suite API utilizes the OAuth2 protocol which is an industry standard for authentication. It supports multiple authentication grant types and can there be used in different use cases.

The following grant types are supported:

- **Authorization Code**
- **Password**
- **Client Credentials**

Especially the client credentials flow is relevant for the server to server communication. The subscription suite API provides credentials per tenant. They are scoped on an organizational level. OAuth2 flow look the following on an abstract level:

3.2 Authentication & Authorization



https://www.digitalocean.com/community/tutorials/an-introduction-to-oauth-2

3.2 Authentication & Authorization

An authentication request could look like this:

```
curl -X POST -d "client_id=xxx&client_secret=xxx&grant_type=client_
credentials" https://sso.subscription-suite.io/oauth/token
```

The response would look like this:

```
{
  "access_token": "b4dc1410-6b15-4a7c-bd22-1abea1f0fa3a",
  "token_type": "bearer",
  "expires_in": 42921
}
```

keylight is planning to provide SDK moving forward in order to make the API integration even easier.

3.3 Resources

These resources are **available** as part of the product catalog API/

Account

The account object represents the Zuora account which owns e.g. subscriptions and has contacts connected to it.

Suite Product Category

The suite product category object represents the subscription suite object for building product hierarchies. It supports multi inheritance and specifies a complex graph which is used for upgrade / downgrade handlings and replacements.

Suite Product

The suite product object represents the subscription suite object for representing subscribable packages. It's extending the Zuora product and product rate plan with further configuration options and constraints.

Suite Product Rate Plan

The suite product rate plan connects the subscription suite product with the Zuora catalog objects.

Contact

The contact object represents the Zuora contact which could either be a bill to contact or sold to contact for an account.

Subscription

The subscription object represents the Zuora subscription object. It represents the contract between the customer and client.

Suite Product Change Group

The suite product change group object represents the subscription suite object for a list of interchangeable products. Therefore, it defines valid upgrade and downgrade paths.

Suite Product Replacement Group

The suite product replacement group object represents the subscription suite object for a list of replacable products. In case of an up- / downgrade of a higher-level product, these groups are checked for e.g. replacing add-ons accordingly.

3.4 Rules Endpoints

In the end the rules endpoints derives **Zuora API** interactions/

The catalog rule endpoints make use of the described catalog model. The subscription suite identifies dependencies between user actions and subscribed products and checks applicable terms and conditions. In the end it derives Zuora API interactions. Currently the following catalog rules endpoints (+ an extra preview endpoint for each of them) are available:

Changing products

Changing products can have side effects, too. The subscription suite checks which other products need to get cancelled or changed. The preview endpoint provides an overview about all changes which will happen.

Changing Quantitiy

The change quantity endpoint allows changing the quantity of one of the subscribed suite products. As described before changing the quantity of one suite product might have side effects and might be even not allowed. The subscription suite identifies all other products which need to change as well and if the change is actually allowed. Additionally, it sets the correct effect dates.

Cancelling products

Product cancellation have similar effects as 2). The subscription suite checks which other products need to get cancelled or changed. The preview endpoint provides an overview about all changes which will happen.

Overview and definitions of all technical fields/

name

The name that will be displayed in the frontend (maybe translated)

description

A description that might be displayed in the frontend (maybe translated)

feature_list

A list of features, separated by new line, that might be displayed in the frontend (maybe translated)

highlight

A highlighted label like "our recommendation" that might be displayed in the frontend (maybe translated)

effective_end_date

If set, the product cannot be subscribed afterwards

show_charges

Flag that defines if all included charges are explicitly listed

sort_priority

Number that determines the sort order when displayed in the customer center. Products with higher number will be displayed first.

purchasable_by_customer

Flag that defines if an end-customer can subscribe via the self-service frontend

purchasable_by_partner

Flag that defines if a partner can subscribe a customer via the self-service frontend

purchasable_by_sales

Flag that defines if the internal sales can subscribe a customer via the self-service frontend

effective_start_date

If set, the product cannot be subscribed before

show_rate_plans

Flag that defines if the included rate plans are shown as a list (might make sense in case of bundles)

3.5 Overview Technical Fields

quantity_level

- **0** Product level: There is only one quantity used on the Suite Product and all charges of all rate plans inherit this quantity
- **1** Rate Plan level: For each rate plan a separate quantity can be defined, and all charges will inherit the quantity of their associated rate plan
- **2** Charge level: Each charge that is based on a quantity will have its own quantity that should be defined separately

price_level

- **0** Product level: There is only one quantity used on the Suite Product and all charges of all rate plans inherit this quantity
- **1** Rate Plan level: For each rate plan a separate quantity can be defined, and all charges will inherit the quantity of their associated rate plan
- **2** Charge level: Each charge that is based on a quantity will have its own quantity that should be defined separately

subscription_type

termed or evergreen

renewal_type

termed or evergreen

term

in months, will only be used if subscription_type = termed

renewal_term

in months, will only be used if renewal_type = termed

is_auto_renew

Flag that defines if a subscription auto-renews, will only be used if subscription_type = termed

contract_effective_mode

- **0** use today's date
- **1** use start of next month
- **2** use start of next quarter
- **3** use start of next half-year
- **4** use start of next year

3.5 Overview Technical Fields

service_activation_mode

- **0** use today's date
- **1** use start of next month
- **2** use start of next quarter
- **3** use start of next half-year
- **4** use start of next year
- **5** leave blank

customer_acceptance_mode

- **0** use today's date
- **1** use start of next month
- **2** use start of next quarter
- **3** use start of next half-year
- **4** use start of next year
- **5** leave blank

term_start_mode

- **0** use contract effective date
- **1** use service activation date
- **2** use customer acceptance date

contract_effective_offset

number of days to offset from the regular start date (see mode above)

service_activation_offset

number of days to offset from the regular start date (see mode above)

customer_acceptance_offset

number of days to offset from the regular start date (see mode above)

cancel_mode

- **0** use global default
- **1** cancellation is not allowed
- **2** cancellation is possible and will be effective immediately (might give a refund)
- **3** cancellation is possible and will be scheduled at a target date, so that no refund happens (termed subscription: end of term / evergreen: next billing date)

cancel_period

number of days that the cancellation has to be made before the next target date (end of term or next billing date)

3.5 Overview Technical Fields

quantity_upgrade_mode

- **0** use global default
- **1** quantity upgrade is not allowed
- **2** quantity upgrade is possible and will be effective immediately
- **3** quantity upgrade is possible and will be scheduled at a target date, so that no refund happens (termed subscription: end of term / evergreen: next billing date)

add_on_mode

When the product is subscribed via the customer center as an add-on with respect to an existing product, this setting defines how the product will be processed.

- **0** use global default
- **1** create a new subscription
- **2** add to the subscription of parent product if terms are equal
- **3** add to the subscription of parent product in any case

upgrade_mode

- **0** use global default
- **1** replacing the product by another product that is considered an upgrade is not allowed
- **2** replacing the product by another product that is considered an upgrade is not allowed is possible and will be effective immediately
- **3** replacing the product by another product that is considered an upgrade is not allowed is possible and will be scheduled at a target date (termed subscription: end of term / evergreen: next billing date)

3.5 Overview Technical Fields

downgrade_mode

0 use global default
1 replacing the product by another pro duct that is considered a downgrade is not allowed
2 replacing the product by another product that is considered a downgrade is not allowed is possible and will be effective immediately
3 replacing the product by another product that is considered a downgrade is not allowed is possible and will be scheduled at a target date (termed subscription: end of term / evergreen: next billing date)

quantity_downgrade_mode

0 use global default
1 quantity downgrade is not allowed
2 quantity downgrade is possible and will be effective immediately (might give a refund)
3 quantity downgrade is possible and will be scheduled at a target date, so that no refund happens (termed subscription: end of term / evergreen: next billing date)

new_subscribe_mode

When the product is subscribed via the regular shop without specifying that it is an add-on with respect to an existing product, this setting defines how the product will be processed. The settings 2-5 will prefer products that are purchases together with the product in the same checkout process.

0 use global default
1 create a new subscription
2 if possible, add to an existing subscription of a parent product with equal term settings
3 if possible, add to an existing subscription of any parent product
4 f possible, add to an existing subscription of a product within the same category with equal term settings
5 if possible, add to an existing subscription of a product within the same category

Write us an email suite@keylight.de

Give us a call +49.30.814 760 14

Need help?

We're here!

4.1 General Inquiries

Get in touch with us **today**

Write us
suite@keylight.de

Call us
+49.30.814 760 14

Inquiries
Dr. Marco Sarich
ms@keylight.de

Jens Wübbenhorst
jw@keylight.de

Germany
keylight GmbH
Kantstrasse 24
10623 Berlin

berlin@keylight.de
www.keylight.de

Switzerland
keylight GmbH
Militärstrasse 52
8004 Zurich

zurich@keylight.ch
www.keylight.ch

www.subscription-suite.com

Kantstrasse 24, 10623 Berlin, Germany / Militaerstr. 52, 8004 Zurich, Switzerland