Keyclic Documentation

Version master

oct. 02, 2017

Table des matières

1 Sommaire **3**
1.1 Aperçu . . . 3
1.2 Considérations techniques . . . 5
1.3 Authentification et connexion . . . 10
1.4 Membres d'organisation et rôles . . . 12
1.5 Organisations . . . 15
1.6 Observations . . . 18
1.7 Rapports . . . 23
1.8 Applications . . . 26
1.9 Notifications . . . 27

Keyclic est une application de remontée et de traitement d'observations. Elle permet à ses utilisateurs de signaler des dysfonctionnements, des problèmes techniques ou tout autre type d'information, et de remonter ces signalements aux organisations concernées.

CHAPITRE 1

Sommaire

Aperçu

Fonctionnement général

L'application Keyclic est librement ouverte aux inscriptions. Une fois inscrit, tout utilisateur peut créer des observations. Une observation est toujours liée à une position géographique et comporte éventuellement une ou plusieurs photos.

Certains utilisateurs peuvent également être regroupés au sein d'organisations. Une organisation est donc un groupe d'utilisateurs membres de la même entreprise, association, corporation, école, etc. Tout utilisateur est libre de créer sa propre organisation et d'inviter d'autres utilisateurs à en devenir membres.

Les administrateurs d'organisation définissent des zones de responsabilité sur lesquelles leur organisation peut intervenir, et des catégories d'intervention (exemples : voirie, animal perdu, propreté, etc...) Ainsi, quand un utilisateur crée une observation, la position géographique de cette observation permet de lui proposer les catégories et organisations qui sont en mesure de traiter son signalement et de le laisser choisir celle qui lui semble la plus adaptée.

Quand un utilisateur crée une observation, celui-ci a le choix de la rendre publique ou privée. Si publique, l'observation est visible par la communauté : ainsi, les autres utilisateurs peuvent la commenter et/ou la soutenir.

De plus, au moment où l'observation est créée, celle-ci doit tout d'abord être modérée par un administrateur de l'application, sauf dans certains cas où la modération est automatique (voir : *Modération et cycle de vie d'une observation*). L'observation validée, un rapport est généré et transmis à l'organisation concernée.

Un administrateur d'organisation a donc accès à l'ensemble des rapports concernant son organisation, chaque rapport correspondant à une observation. Pour chaque rapport, il peut créer une ou plusieurs interventions, et affecter ces interventions aux membres de son organisation. Une autre possibilité est de déléguer ce rapport à une organisation partenaire, le patenaire aura alors la charge de créer des intervent. Une fois que toutes les interventions ont été accomplies, il pourra considérer que le traitement est terminé et clôturer le rapport.

Liens utiles

— Spécification Swagger de l'API

— Documentation technique de l'API
— Code source du SDK client javascript

Vocabulaire

Observation

Remarque effectuée sur le terrain par un utilisateur, pouvant porter sur un dysfonctionnement, un problème technique, une nuisance, etc. Toute observation est forcément faite en une position géographique donnée. Elle peut éventuellement comporter des photos, et être commentée et/ou soutenue par les autres utilisateurs.

Terme technique : feedback.

Voir la page *Observations*.

Rapport

Toute observation, une fois validée par un administrateur de l'application, entraîne la génération d'un rapport. Un rapport reprend donc toutes les informations de l'observation dont il est issu, et n'est visible et manipulable que par l'organisation concernée. Le rapport peut donc être vu comme le pendant « professionnel » de l'observation. C'est sur ce rapport que l'organisation travaille : création d'interventions, suivi, délégation, etc.

Terme technique : report.

Voir la page *Rapports*

Organisation

Groupe d'utilisateurs pouvant être une entreprise, une école, une association, un groupe de pesonnel d'un site, d'un chantier, etc.

Terme technique : organization.

Voir la page *Organisations*.

Zone de responsabilité

Aire géographique sur laquelle une organisation peut intervenir.

Terme technique : place.

Voir la section *Gestion des zones de responsabilité*.

Catégorie

Secteur d'activité d'une organisation.

Terme technique : category.

Voir la section *Gestion des catégories*.

Soutien

Une observation peut être soutenue par les utilisateurs de la communauté, afin de leur donner plus de poids.

Terme technique : contribution.

Voir la section *Soutiens*.

Opération

Une opération est une tâche créée par un administrateur d'organisation sur un rapport donné. Cette tâche est assignée à un membre de l'organisation. Un rapport ne peut être clôturé que si toutes les opérations qui lui sont liées ont été accomplies (ou refusées).

Terme technique : operation.

Voir la section reports-operations.

Partenaires

Un administrateur d'organisation peut définir des organisations partenaires, qui sont d'autres organisations auxquelles il pourra déléguer des rapports.

Terme technique : relationship.

Voir la section *Gestion des partenariats*.

Considérations techniques

L'API Keyclic est une **API REST**. Toutes les opérations sont effectuées via le protocole **https** et sécurisées par JSON Web Tokens. Les données sont échangées exclusivement au format **JSON**.

Le nom de domaine de l'API Keyclic est :

```
api.keyclic.com
```

Liens utiles

— Spécification Swagger de l'API
— Documentation technique Restlet de l'API

Applications et clés d'applications

Tout client de l'API doit transmettre dans les headers de chaque requête une clé définissant l'application sur laquelle il travaille.

Si vous développez un client pour travailler sur une application existante de Keyclic, vous devez connaître la clé de cette application. Si au contraire vous développez un client pour une nouvelle application, merci de vous adresser à la société Keyclic pour que celle-ci crée l'application et sa configuration et vous fournisse la clé correspondante.

Exemples de clés d'applications :
— com.keyclic.app
— com.keyclic.city

— com.keyclic.highway

Chaque requête doit donc préciser, dans ses headers, la valeur du paramètre X-Keyclic-App. Voir ci-dessous le paragraphe *Requêtes* pour la mise en œuvre.

Notez cependant que la base utilisateurs est commune à toutes les applications Keyclic. De fait, les endpoints d'inscription et de connexion (voir : *Authentification et connexion*) font exception à la règle ci-dessus : ces deux endpoints n'exigent pas qu'une clé d'application leur soit fournie.

Requêtes

Dans cette documentation, chaque endpoint de l'API sera décrit par le chemin d'accès à la ressource précédé du verbe HTTP.

Exemple :

```
GET /feedbacks
```

Le endpoint ci-dessus retourne toutes les observations. Son url véritable est

```
https://api.keyclic.com/feedbacks
```

mais pour des raisons de concision, dans cette documentation, nous ne préciserons jamais le protocole ni le nom de domaine.

Paramètres d'url

Dans cette documentation, les variables d'URI (exemples : identifiant d'une ressource, numéro de page, etc) seront exprimés entre accolades. Par exemple, pour récupérer une observation (feedback) donnée :

```
GET /feedbacks/{feedback}
```

Dans l'API Keyclic, conformément aux principes d'architecture REST, les paramètres de filtrage sont toujours passés en "query string". Exemple :

```
GET /feedbacks?page={page}
```

Par ailleurs, pour une meilleure lisibilité, les paramètres d'uri seront écrits tels quels dans cette documentation, et non sous leur forme url encodée :

```
GET /feedbacks?before=2018-04-22T01:00:00+05:00
```

Headers

En plus des headers conventionnels de HTTP/1.1, l'API Keyclic accepte, et même exige dans la plupart des cas, le header **X-Keyclic-App**, correspondant à l'application utilisée (voir ci-dessus : *Applications et clés d'applications*). Par exemple, pour récupérer toutes les observations sur l'application com.keyclic.app, la requête comportera le header :

```
X-Keyclic-App : com.keyclic.app
```

Tous les endpoints exigent que ce header soit fourni, à l'exception des endpoints de login et de changement de mot de passe. (voir : *Authentification et connexion*)

Toutes les requêtes (à l'exception du login, du register et du changement de mot de passe) doivent aussi comporter le header Authorization afin d'authentifier l'utilisateur. (voir : *Authentification et connexion*)

Format des requêtes et réponses

Le seul type de contenu accepté par l'API Keyclic est JSON. Vos requêtes devront donc comporter le header :

```
Content-type: application/json
```

et le corps des requêtes devra toujours être formaté en JSON. Les réponses sont également toujours retournées au format JSON.

Envoi de fichiers

Tous les fichiers sont envoyés en base 64 à l'API. Voici par exemple l'ajout d'une image représentant un carré rouge d'1 pixel sur 1 sur une observation :

```
POST /feedbacks/{feedback}/images
```

```
{
    "image":"
↪iVBORw0KGgoAAAANSUhEUgAAAAUAAAAFCAIAAAACDbGyAAAACXBIWXMAAAsTAAALEwEAmpwYAAAAB3RJTUUH4QIVDRUfvq7u+AA
↪"
}
```

Pagination

Tous les endpoints permettant de récupérer une collection de ressources peuvent être paginés avec les filtres **page** et **limit**. Par exemple, pour récupérer la deuxième page des observations à raison de 5 observations par page :

```
POST /feedbacks?page=2&limit=5
```

Par défaut, *page* a la valeur 1 et *limit* a la valeur 10. Ainsi le endpoint

```
POST /feedbacks
```

retourne les 10 premières observations.

Le retour d'une collection contient les informations et liens nécessaires pour naviguer entre les pages de cette collection. Exemple de retour (partiel) de la liste des observations :

```
{
  "page": 2,
  "limit": 10,
  "pages": 8,
  "total": 72,
  "_links": {
    "self": {
      "href": "/feedbacks?page=2&limit=10"
    },
    "first": {
      "href": "/feedbacks?page=1&limit=10"
    },
    "last": {
      "href": "/feedbacks?page=8&limit=10"
    },
    "next": {
      "href": "/feedbacks?page=3&limit=10"
```

```
    },
    "previous": {
      "href": "/feedbacks?page=1&limit=10"
    }
  }
}
```

Dans cette documentation, nous ne rappellerons pas systématiquement qu'il est possible de paginer avec les filtres *page* et *limit*, ceux-ci étant communs à tous les endpoints retournant une collection.

Modification de ressources avec la méthode PATCH

Dans l'API Keyclic, la modification des ressources s'effectue avec la méthode PATCH. Contrairement à la méthode PUT, la méthode PATCH permet de modifier une seule propriété, ou une partie seulement des propriétés, d'une ressource, sans qu'il soit nécessaire d'en envoyer une représentation complète. Le format utilisé pour la description du patch est JSON Patch. La seule opération acceptée par l'API lors d'un PATCH est l'opération *replace*.

À titre d'exemple, voici la modification de la popriété *billingEmailAddress* d'une organisation :

```
PATCH /organizations/{organization}
```

```
[
        {
                "op":"replace",
                "path":"/billingEmailAddress",
                "value":"test@test.com"
        }
]
```

Retours d'erreurs

Toute erreur entraîne une réponse de code 4xx reflétant le type d'erreur.

Quand il s'agit d'une erreur de type 400 (Bad Request), les raisons de l'erreur sont retournées.

Les erreurs sont décrites au format vdn.error.

L'exemple suivant montre un retour d'erreur de validation. Le champ *path* indique la propriété sur laquelle porte l'erreur (ici : reporter), et le champ *message* indique la nature de l'erreur.

```
{
   "@context":"https:\/\/github.com\/blongden\/vnd.error",
   "@type":"ValidationError",
   "message":"Validation failed.",
   "total":1,
   "_embedded":{
      "errors":[
         {
            "@context":"https:\/\/github.com\/blongden\/vnd.error",
            "@type":"Error",
            "message":"Cette valeur ne doit pas \u00eatre vide.",
            "path":"reporter"
         }
      ]
   }
}
```

Changements de statut

Plusieurs ressources manipulées par l'API ont un cycle de vie et possèdent un certain statut à un instant donné. C'est le cas des observations, des rapports et des opérations.

Pour ces ressources, l'état est toujours indiqué dans la réponse avec le paramètre *state*, et les actions possibles pour faire évoluer ce statut sont toujours indiquées sous le paramètre *stateTransitions*. Exemple :

```
GET reports/{report}
```

Réponse (partielle) :

```
{
  "type": "Report",
  "id": "cb7118b5-a821-4cf2-9475-0c0d0efdb8d0",
  "state": "NEW",
  "_embedded": {
    "stateTransitions": [
      "accept",
      "refuse"
    ]
  }
}
```

Dans l'exemple ci-dessus, le rapport est en statut NEW et les actions possibles sur son statut sont *accept* et *refuse*.

Tout changement de statut est effectué avec la méthode PATCH et l'opération *replace*, en précisant *transition* pour le path, et l'action à effectuer pour la valeur.

Par exemple, pour accepter le rapport ci-dessus :

```
PATCH /reports/{report}/state
```

```
[
        {
                "op":"replace",
                "path":"transition",
                "value":"accept"

        }
]
```

La réponse nous informe que le rapport possède désormais le statut ACCEPTED, et que les actions possibes sont désormais *refuse*, *hold* et *progress* :

```
{
  "type": "Report",
  "id": "32219286-528a-4f97-b81e-fe7a8cb85707",
  "state": "ACCEPTED",
  "_embedded": {
    "stateTransitions": [
      "refuse",
      "hold",
      "progress"
    ]
```

```
  }
}
```

Les actions et status possibles pour chaque type de ressources sont décrits dans les sections idoines de cette documentation.

Authentification et connexion

L'API Keyclic se base sur le standard JSON Web Tokens pour la sécurisation de l'échange des données. Toute requête à l'API est nécessairement effectuée en fournissant un jeton d'accès (accessToken) permettant au serveur de vérifier l'identité de l'utilisateur. Un endpoint permet à l'utilisateur d'obtenir ce jeton d'accès en fournissant son login et son mot de passe. Cette section détaille la création d'un compte, l'obtention et l'utilisation d'un accessToken, et la modification d'un compte utilisateur.

Pour plus d'informations sur le standard JWT, voir : le site officiel de JSON Web Tokens.

Création d'un compte utilisateur

Un nouveau compte utilisateur est créé avec la requête :

```
POST /security/register
```

Exemple :

```
{
    "email":"test@test.com",
    "password":"test"
}
```

Le nouvel utilisateur se voit attribuer un identifiant unique et le rôle ROLE_USER, lui permettant d'utiliser les fonctionnalités de base de l'API.

Connexion

La connexion consiste à envoyer ses credentials au serveur afin d'obtenir en retour un accessToken qui sera utilisé pour les requêtes ultérieures.

La connexion s'effectue avec la requête :

```
POST /security/login
```

Exemple :

```
{
    "login":"test@test.com",
    "password":"test"
}
```

Si les credentials sont reconnus par le serveur, celui-ci retourne un accessToken qui sera utilisé par l'utilisateur pour ses futures requêtes.

Utilisation du token

La quasi-totalité des requêtes à l'API nécessitent que l'utilisateur soit authentifié, c'est-à-dire qu'il fournisse son accessToken. Le token est envoyé dans l'en-tête Authorization de la requête, préfixé par Bearer.

Par exemple, pour récupérer la liste des feedbacks de l'application com.keyclic.app, un utilisateur utilise le endpoint :

```
GET /feedbacks
```

```
Request headers :
    X-Keyclic-App : com.keyclic.app
    Authorization : Bearer eyJhbGciOiJSUzI1NiJ9.
↪eyJyb2xlcyI6WyJPUkdBTklaQVRJT046QURNSU4iLCJST0xFX1VTRVIiXSwidXNlcm5hbWUiOiJ0ZXN0MjJAdGVzdC5jb20iLCJ
↪ZIqbBVSgJaXKj73IPYbFeEfie6FUIflv-ausUO-AAzVjPg8-jdhFv3nqsdOVJvE_
↪AB4bXjME1CRVFI7xD2SYCA8V6E6H2-y0XZE8SN_XTpHGMxDvOP27C2VUNQfPwgeWxjXzlDopo_
↪U9ybAEX4QdFhW14aeRgb9YWMDlzSD6VLgJO-LuprxX668Ajq5X9c8YND4D_p4WRDSQr8pqb3rTY9NQ6O34F-
↪OpDJlAUYj0pwMehYWpywVKJHRMv9xCRRoI8HrU6H3J3wo-K2OtQVJi9XFZ8g8sohw_ZaasG7dohxrO-
↪NtYSrOPXIXPI6kCDRuMi7sce06wfno1bC3jBoc83EhiBSBpDbWL98DSjPbF1SaCeE05aATfM5cMEXbnp8Iwh-
↪QLxglE4M-ZISJ8VooxzJxa7cWLlFW-iu0XWVFWrMbYgmSoU0PKRQB47w_
↪IOPxjWzDeMUTSA3esDwkxsYlNdS9SZe201EvI6zur5Ayot0PEGfAgex6Ew-
↪eKOHAfnuDiqeLQLbWs4Y69FO2DooWUhkfVGdl-IGglDPgk2AOs3w19e7mx-Gmm8DlUUr-
↪bK61NPPQ8dy7HPjXnU63-jbA17MAjHaRTO4eKopcZMWbpL-jgQjJltV3R5_0qNODaHCS_
↪auZs2cyqFN0HL9Rred5g7t6Fxyk-8MyyX0GiTyHsp3c
```

Toutes les requêtes à l'API Keyclic nécessitent l'envoi d'un accessToken dans les headers, à l'exception évidente des endpoints suivants :

- création d'un compte (POST /security/register)
- login (POST /security/login)
- demande de changement de mot de passe
- changement de mot de passe (POST /security/password/change/{changePasswordToken})

Modification du mot de passe

Un utilisateur qui souhaite modifier son mot de passe procède en deux étapes.

Il effectue d'abord une demande de changement de mot passe :

```
POST /security/password/change-request
```

Exemple :

```
{
    "email":"test@test.com"
}
```

Cette requête envoie un email à l'utilisateur contenant un lien se terminant par un token de vérification. Exemple de lien :

```
https://domain.com/#/password-reset/jrtVqBLxxoSA0c2hpsOBN-JQGQHGN3YXsKPMG1PWWWA
```

Dans le lien ci-dessus, jrtVqBLxxoSA0c2hpsOBN-JQGQHGN3YXsKPMG1PWWWA est le jeton de changement de mot de passe. La portion d'url *https ://domain.com/#/password-reset/* dépend de la configuration de l'application.

L'utilisateur peut ensuite changer son mot de passe avec :

```
POST /security/password/change/-VtYMG0VnU8vHJdKUC_AqA_XpypI9kd8OmOvWj4NYMw
```

Exemple :

```
{
    "password":"password"
}
```

Modification des données utilisateur

Pour les données autres que le mot de passe, l'utilisateur requêtera sur le endpoint :

```
PATCH /people/{user}
```

Pour plus d'informations sur les requêtes PATCH, voir la section *Modification de ressources avec la méthode PATCH*.

Par exemple, pour modifier son nom :

```
[
  {
    "op":"replace",
    "path":"/familyName",
    "value":"Nom de famille"
  }
]
```

Les champs qu'un utilisateur peut modifier sont : son nom (familyName), son prénom (givenName), sa photo (image), son travail (job), son adresse email (email).

Membres d'organisation et rôles

Quand un utilisateur est membre d'une organisation, il peut bénéficier de aucun, un ou plusieurs rôles définissant un ensemble de permissions et de possibilités d'actions.

Définissant un ensemble de droits, ces rôles peuvent être cumulés pour un même utilisateur et sont au nombre de cinq :

Les utilisateurs de l'application Keyclic possèdent un ou plusieurs rôles, définissant un ensemble de permissions et de possibilités d'action. Ces rôles, qui peuvent être cumulés pour un même utilisateur, sont au nombre de trois :

- Aucun rôle - Membre d'organisation
- Administrateur d'organisation
- Agent
- Statistiques
- Export

Un utilisateur pouvant être membre de plusieurs organisations, les rôles qu'il possède sont indépendants et peuvent être différent d'une organisation à une autre.

Aucun rôle - Membre d'organisation

Tout utilisateur nouvellement attaché à une organisation devient « Membre d'organisation ». Un « Membre d'organisation » possède les mêmes droits qu'un utilisateur de base sans attachement à une organisation.

Ce rôle permet en plus de pouvoir :

- Être listé par les administrateurs de l'organisation
- Recevoir une assignation à une intervention
- Éditer une intervention (changement des libellés, ajout de photos de constatation, etc)
- Changer le statut d'une intervention

Note : Un utilisateur peut être membre d'organisation de plusieurs organisations et une organisation peut avoir plusieurs membres d'organisation.

Administrateur d'organisation

Tout utilisateur a la possibilité de créer une nouvelle organisation.

L'utilisateur qui créé une nouvelle organisation en devient automatiquement membre et administrateur, c'est-à-dire qu'il se voit attribuer le rôle « Administrateur d'organisation ».

Aux permissions et aux droits des « Membres d'organisation » s'ajoute les suivants :
- Ajouter de nouveaux membres à une organisation
- Gérer les catégories d'une organisation
- Gérer les zones de responsabilité d'une organisation
- Gérer les partenaires d'une organisation
- Consulter et gérer les rapports reçus d'une et les opérations liées à ces rapports

Voir : *Organisations*

Voir : *Rapports*

Note : Un utilisateur peut être « Administrateur d'organisation » de plusieurs organisations et une organisation peut avoir plusieurs « Administrateurs d'organisation ».

Agent

« Agent » est un rôle particulier adapté aux personnels de terrain qui effectuent des remontées d'observation uniquement dédiées à leur organisation.

Aux permissions et aux droits des « Membres d'organisation » s'ajoute les suivants :
- Activer le Mode Pro (cf infra : Observation postée en Mode Pro)
- De remonter par défaut des observations privées

Note : Un utilisateur ne peut être « Agent » que d'une seule organisation et une organisation peut avoir plusieurs « Agents ».

Statistiques

Le rôle « Statistique » permet d'accéder aux statistiques d'une organisation.

Note : Un utilisateur peut avoir le rôle « Statistique » pour plusieurs organisations et une organisation peut avoir plusieurs utilisateurs avec le rôle « Statistique ».

Export

Le rôle « Export » permet d'accéder aux fonctionnalités d'exportation excel des rapports qu'a reçu une organisation.

Note : Un utilisateur peut avoir le rôle « Export » pour plusieurs organisations et une organisation peut avoir plusieurs utilisateurs avec le rôle « Export ».

Exemple de ressource utilisateur

Déprécié : La section suivante doit être mise à jour pour prendre en considération la nouvelle gestion matricielle des rôles.

La lecture d'une ressource Utilisateur permet de découvrir les rôles de cet utilisateur, et éventuellement l'organisation dont il est administrateur et/ou l'application dont il est administrateur.

```
GET /people/{user}
```


```
{
  "username": "test@gmail.com",
  "email": "test@gmail.com",
  "type": "Person",
  "roles": [
    "APPLICATION:ADMIN",
    "ORGANIZATION:ADMIN",
    "ROLE_USER"
  ],
  "id": "5020c6ea-ca07-42d1-994f-d90b86703b1a",
  "createdAt": "2017-02-20T17:52:39+01:00",
  "updatedAt": "2017-02-27T14:48:39+01:00",
  "_links": {
    "self": {
      "href": "/people/5020c6ea-ca07-42d1-994f-d90b86703b1a",
      "iriTemplate": {
        "mapping": {
          "person": "5020c6ea-ca07-42d1-994f-d90b86703b1a"
        }
      }
    },
    "memberOf": {
      "href": "https://api.sandbox.keyclic.com/organizations/84d36093-b8bc-47ad-bc8a-
↪a043b3e301a9",
      "iriTemplate": {
        "mapping": {
          "organization": "84d36093-b8bc-47ad-bc8a-a043b3e301a9"
        }
      }
    }
  }
}
```


Ce retour indique que :

1. Cet utilisateur possède le rôle ROLE_USER, comme tous les utilisateurs.
2. Il est membre de l'organisation 84d36093-b8bc-47ad-bc8a-a043b3e301a9
3. Il possède le rôle ORGANIZATION :ADMIN, il est donc administrateur de l'organisation 84d36093-b8bc-47ad-bc8a-a043b3e301a9
4. Il possède le rôle APPLICATION :ADMIN, il est donc administrateur de l'application à laquelle est rattachée l'organisation 84d36093-b8bc-47ad-bc8a-a043b3e301a9

Récupération des utilisateurs

Pour récupérer l'ensemble des utilisateurs de l'application :

```
GET /people
```

Pour récupérer un utilisateur :

```
GET /people/{user}
```

Pour rechercher les membres dont l'adresse email match un mot donné :

```
GET /people?search[email]=martin
```

Pour filtrer les membres d'une organisation :

```
GET /people?organization={organization}
```

Organisations

Dans l'application Keyclic, une organisation est une entité telle que corporation, entreprise, département d'entreprise, association, école, institution, etc à laquelle peuvent être rattachées les observations faites par les utilisateurs.

Les *Aucun rôle - Membre d'organisation* sont des utilisateurs du service Keyclic rattachés à une organisation. Un ou plusieurs membres d'une organisation peuvent en être les administrateurs (voir : *Administrateur d'organisation*). Une organisation possède au minimum un administrateur d'organisation.

Les *Administrateur d'organisation* peuvent définir les champs d'intervention de leur organisation en créant des catégories (exemple : voirie, transports, etc) et des zones de responsabilité. Quand un utilisateur crée une nouvelle observation, les coordonnées géographiques de cette observation sont toujours automatiquement précisées. Ainsi, l'application est en mesure de lui retourner l'ensemble des organisations et catégories qui ont une zone de responsabilité sur cette position. Ce qui lui permet de choisir la catégorie la plus adéquate à son observation.

Création d'une organisation

Tout utilisateur peut créer une nouvelle organisation :

```
POST /organizations
```

Exemple :

```
{
    "name":"Nom de l'organisation",
    "billingEmailAddress":"test@test.com",
    "notificationEmailAddress":"test@test.com"
}
```

L'utilisateur devient automatiquement membre et administrateur de cette nouvelle organisation.

Pour récupérer toutes les organisations de l'application :

```
GET /organizations
```

Il est possible de filtrer la requête ci-dessus sur un point géographique (voir ci-dessous : *Gestion des zones de responsabilité*) :

```
GET /organizations?geo_coordinates=+44.851404209987386-0.5762618780136108
```

Gestion des membres

Pour ajouter un nouveau membre à une organisation :

```
POST /organizations/{organization}/members
```

Exemple :

```
{
    "person":"63d07fc5-f4e6-471c-a8cc-3c3f227c8c2d"
}
```

Ce endpoint est réservé à un utilisateur possédant le rôle ORGANIZATION :ADMIN et membre de l'organisation {organization}.

Pour récupérer les membres d'une organisation :

```
GET /people?organization={organization}
```

Pour retirer un membre d'une organisation, un administrateur de cette organisation exécutera la requête :

```
DELETE /organizations/{organization}/members/{member}
```

Pour plus d'informations sur le rôle ORGANIZATION :ADMIN et ses privilèges, voir users-organization-admin.

Gestion des zones de responsabilité

Un administrateur d'organisation peut créer des zones de responsabilité, correspondant aux lieux sur lesquels cette organisation intervient :

```
POST /organizations/{organization}/places
```

body :

```
{
    "name": "Test",
    "polygon":
    {
        "rings":
        [
            {
                "points":
                [
                    {
                        "longitude": 2.373991012573242,
                        "latitude": 48.84088179130599
                    },
                    {
                        "longitude": 2.3763084411621094,
                        "latitude": 48.84205393836751
                    },
                    {
                        "longitude": 2.376694679260254,
                        "latitude": 48.84189859515306
                    },
                    {
                        "longitude": 2.3787975311279297,
                        "latitude": 48.84041574931067
                    },
                    {
                        "longitude": 2.376115322113037,
```

```
                        "latitude": 48.839031720249054
                    },
                    {
                        "longitude": 2.373991012573242,
                        "latitude": 48.84088179130599
                    }
                ]
            }
        ],
        "srid": 5555
    },
    "elevation": 1
}
```

Pour récupérer toutes les zones de responsabilité de l'application :

```
GET /places
```

La requête ci-dessus peut-être filtrée sur une organisation donnée et/ou sur un point géographique donné :

```
GET /places?geo_coordinates=+44.851404209987386-0.5762618780136108&organization=
↪{organization}
```

Gestion des catégories

Les catégories sont les secteurs d'activité d'une organisation. Un administrateur d'organisation peut créer une nouvelle catégorie en lui donnant un nom, une couleur et une icône. L'icône sera choisie dans le jeu d'icônes de Font Awesome.

```
POST /organizations/{organization}/categories
```

Exemple :

```
{
    "name":"Nom de la catégorie",
    "color":"#ff0000",
    "icon":"fa-bug"
}
```

Les 3 propriétés name, color et icon peuvent être éditées par une requête PATCH (voir : *Modification de ressources avec la méthode PATCH*).

Pour récupérer l'ensemble des catégories de l'application :

```
GET /categories
```

La requête ci-dessus peut-être filtrée sur une organisation donnée et/ou sur un point géographique donné :

```
GET /categories?geo_coordinates=+44.851404209987386-0.5762618780136108&organization=
↪{organization}
```

Gestion des partenariats

Une organisation peut avoir des partenaires, c'est-à-dire des organisations qui lui sont rattachées et à qui l'administrateur de l'organisation pourra déléguer des rapports. La relation de partenariat est unilatérale : si une organisation A est partenaire d'une organisation B, B n'est pas forcément partenaire de A.

Pour ajouter un nouveau partenaire à l'organisation, un administrateur de l'organisation exécutera le endpoint :

```
POST /organizations/{organization}/relationships
```

Exemple :

```
{
    "organization":"84d36093-b8bc-47ad-bc8a-a043b3e301a9"
}
```

Pour récupérer les partenaires d'une organisation :

```
GET /organizations/{organization}/relationships
```

Cette requête ne peut être exécutée que par un administrateur de l'organisation.

Observations

Une observation est toujours faite en une position géographique donnée. La position géographique est la composante la plus importante, et la seule obligatoire, d'une observation. Les paramètres optionnels étant la description, la catégorie, et éventuellement une ou plusieurs photos.

Tous les utilisateurs peuvent créer des observations.

Création d'une observation

```
POST /feedbacks/issues
```

Exemple minimaliste : dans l'exemple suivant, une observation est créée sans catégorie et sans description.

```
{
    "geo": {
        "elevation":1,
        "point": {
            "latitude":44.851343361295214,
            "longitude":-0.5763262510299683
        }
    },
    "reporter":"6dbbd601-267f-46ea-be90-8c9742f7180b"
}
```

Ce endpoint se présente sous la forme **/feedbacks/issue** et non pas simplement **/feedbacks**, car à terme, il sera possible de créer différents types d'observation. Actuellement, seul le type "issue" est disponible.

Exemple plus complet, l'utilisateur précise une catégorie et une description :

```
{
    "geo": {
        "elevation":1,
        "point": {
            "latitude":44.851343361295214,
            "longitude":-0.5763262510299683
        }
    },
    "category":"b0d007d5-e6ad-4113-b2b5-d8a1858a2fb1",
```

```
    "description":"Mon feedback 5",
    "reporter":"6dbbd601-267f-46ea-be90-8c9742f7180b"
}
```

L'utilisateur peut ensuite ajouter une ou plusieurs images à son observation :

```
POST /feedbacks/{feedback}/images
```

Exemple :

```
{
    "image":"
↪iVBORw0KGgoAAAANSUhEUgAAAAUAAAAFCAIAAAACDbGyAAAACXBIWXMAAAsTAAALEwEAmpwYAAAAB3RJTUUH4QIVDRUfvq7u+AA
↪"
}
```

Pour plus d'informations sur l'envoi d'images, voir *Envoi de fichiers*.

Rattachement d'une observation à une organisation

L'application Keyclic ne se contente pas de recueillir des observations : elle les fait ensuite remonter, sous la forme de *Rapports*, aux organisations concernées, qui en assureront le traitement. Toute observation doit donc être, dans la mesure du possible, remontée à une organisation sous la forme d'un rapport. Pour cela, quatre cas de figure peuvent se présenter :

- Si la position géographique de l'observation ne correspond à aucune zone de responsabilité, alors l'API retournera une erreur 409 et aucune organisation ne recevra de rapport sur cette observation.
- Si la position géographique de l'observation se trouve dans une zone de responsabilité définie par une organisation, alors le rapport de l'observation est automatiquement remonté à l'organisation en question.
- Si la position géographique de l'observation se trouve sur deux (ou plus) zones de responsabilité appartenant à deux (ou plus) organisations différentes, et que l'utilisateur a précisé une catégorie, alors le rapport de l'observation est remonté à l'organisation propriétaire de la catégorie en question.
- Si la position géographique de l'observation se trouve sur deux (ou plus) zones de responsabilité appartenant à deux (ou plus) organisations différentes, mais que l'utilisateur n'a pas précisé de catégorie, alors plusieurs rapports sont générés et remontés à toutes les organisations concernées. La première organisation qui acceptera le rapport pourra en effectuer le traitement.

Modération et cycle de vie d'une observation

Après qu'un utilisateur a créé une nouvelle observation, celle-ci possède le statut PENDING_REVIEW : en attente de modération. Elle devra être validée par un administrateur de l'application (sauf cas particulier d'une *Observation postée par un membre d'organisation*).

Voir : *Changements de statut*

Un administrateur d'application valide une observation avec le endpoint :

```
POST /feedbacks/{feedback}/state
```

Exemple :

```
[
    {
        "op":"replace",
        "path":"transition",
```

```
        "value":"accept"
    }
]
```

L'observation prend alors le statut DELIVERED et un rapport est créé sur cette observation.

Voir : *Rapports*

Pour refuser une observation :

```
[
    {
        "op":"replace",
        "path":"transition",
        "value":"refuse"
    }
]
```

L'observation prend alors le statut REFUSED.

Observation postée par un membre d'organisation

Les membres (*Membres d'organisation et rôles*) peuvent poster des observations de la même façon que tous les utilisateurs. Cependant, si un membre d'organisation fournit, dans sa requête, l'identifiant de son organisation, il entre dans le mode de fonctionnement que nous avons appelé le "mode pro", et son observation pourra être traitée différemment :

- Si son observation est positionnée dans une zone de responsabilité régie par son organisation, alors cette observation est automatiquement validée (sans passer par l'étape de modération) et le rapport créé qui en découle est automatiquement accepté.
- Si son observation n'est pas positionnée dans une zone de responsabilité régie par son organisation, alors son observation est refusée et une erreur 409 est retournée.

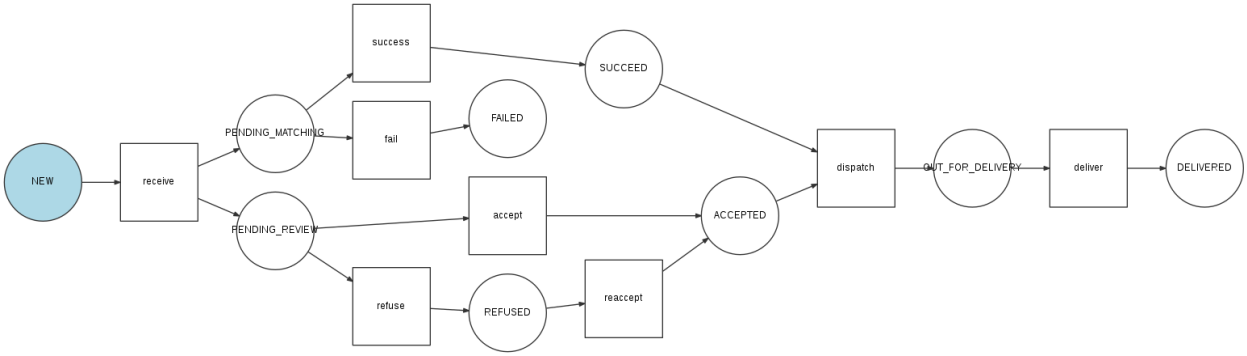
Mode normal vs "Mode pro"

Sur la figure ci-dessous, le rectangle A représente une zone de responsabilité appartenant à une organisation A, et le rectangle B représente une zone de responsabilité appartenant à une organisation B.

Chaque point représente une observation effectuée **par un utilisateur membre de l'organisation B**.

En bleu : observations effectuées en passant l'identifiant de son organisation (correspond au "mode pro"). En rouge : observations effectuées sans passer l'identifiant de son organisation. Ces observations sont donc identiques à celle d'un utilisateur lambda.

Résumé du cycle de vie d'une observation



Récupération des observations

Pour récupérer les observations :

```
GET /feedbacks
```

Cette requête retourne uniquement les observations dont le statut est DELIVERED.

Plusieurs critères permettent de filtrer les observations.

Par statut : paramètre state

Par exemple, pour filtrer les observations en attente de validation, un administrateur d'application effectuera la requête :

```
GET /feedbacks?state=PENDING_REVIEW
```

Autour d'un point : paramètre geo_near

Exemple :

```
GET /feedbacks?geo_near[radius]=1000&geo_near[geo_coordinates]=+44.8-0.5
```

retournera les observations situées dans un rayon de 1000 mètres autour du point de latitude +44.8 et de longitude 0.5.

Dans un GeoHash : paramètre geo_hash

GeoHash est un système de géocodage [...] basé sur une fonction de hachage qui subdivise la surface terrestre selon une grille hiérarchique. (Source : Wikipedia)

Pour plus d'informations sur GeoHash, voir :
- Site officiel de GeoHash
- GeoHash explorer

Les observations peuvent être filtrées par GeoHash de la façon suivante :

```
GET /feedbacks?geo_hash[]=ezzx&geo_hash[]=ezzz
```

retournera les observations comprises dans les geo hash ezzx et ezzz.

Sur une période donnée : paramètres before et after

Exemple :

```
GET /feedbacks?after=2017-01-10T00:00:00+05:00&before=2017-02-22T23:59:59+05:00
```

retournera les observations effectuées entre le 10/01/2017 et le 22/02/2017.

Les dates sont écrites au format : ISO 8601.

Par organisation

```
GET /feedbacks?organization={organization}
```

Commentaires

Les utilisateurs de la communauté peuvent commenter une observation :

```
POST /feedbacks/{feedback}/comments
```

Exemple :

```
{
    "text":"Mon commentaire"
}
```

Pour récupérer les commentaires d'une observation :

```
GET /feedbacks/{feedback}/comments
```

Soutiens

Un utilisateur peut soutenir une contribution en effectuant la requête suivante, sans paramètres :

```
POST /feedbacks/{feedback}/contributions
```

Pour récupérer tous les soutiens effectués sur une observation :

```
GET /feedbacks/{feedback}/contributions
```

Rapports

Chaque fois qu'une observation est acceptée, après validation par un administrateur de l'application sauf dans certains cas où la modération est automatique (voir : *Modération et cycle de vie d'une observation*), un rapport est créé.

Un administrateur d'organisation récupère les rapports concernant son organisation avec :

```
GET /organizations/{organization}/reports
```

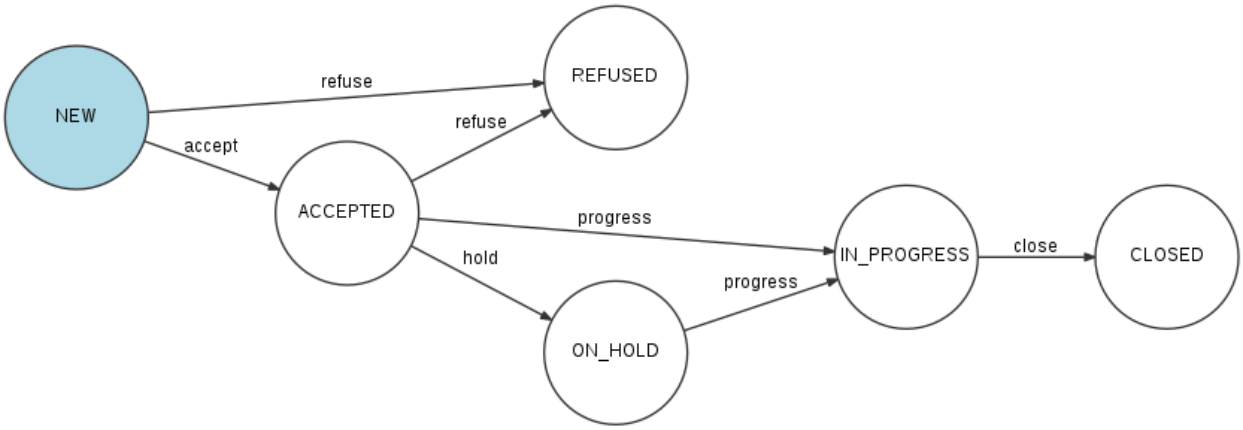
Et un rapport donné est récupéré avec :

```
GET /reports/{report}
```

Cycle de vie d'un rapport

Quand un nouveau rapport est généré à partir d'une observation, il possède le statut NEW.

Le schéma ci-dessous montre l'évolution du statut d'un rapport en fonction des actions qui sont effectuées sur ce rapport.



Un endpoint unique permet de changer le statut du rapport :

```
PATCH /reports/{report}/state
```

Par exemple, pour passer du statut NEW au statut ACCEPTED, l'administrateur de l'organisation effectuera un "accept" en passant dans le corps de la requête :

```
[
  {
    "op":"replace",
    "path":"/transition",
    "value":"accept"
  }
]
```

Un rapport ne peut être clôturé (statut CLOSED) que si :

— Toutes les interventions associées à ce rapport ont été clôturées ou refusées (voir ci-dessous le paragraphe *Interventions*).
— Tous les rapports délégués à d'autres organisations à partir de ce rapport ont été clôturés (voir ci-dessous le paragraphe *Délégation de rapports*).

Interventions

Une intervention est une action à réaliser associée à un rapport et assignée à un membre de l'organisation.

Pour récupérer l'ensemble des interventions associées à un rapport :

```
GET /reports/{report}/operations
```

Création et modification d'une intervention

Un administrateur d'organisation crée une intervention sur un rapport en effectuant la requête :

```
POST /operations
```

Exemple :

```
{
    "description":"Description de l'intervention",
    "name":"Nom de l'intervention",
    "report":"cb7118b5-a821-4cf2-9475-0c0d0efdb8d0"
}
```

Une intervention nouvellement créée possède le statut NEW.

Une ou plusieurs images peuvent être ajoutées à l'intervention :

```
POST /operations/{operation}/images
```

Exemple :

```
{
    "image":"
↪iVBORw0KGgoAAAANSUhEUgAAAAUAAAAFCAIAAAACDbGyAAAACXBIWXMAAAsTAAALEwEAmpwYAAAAB3RJTUUH4QIVDRUfvq7u+AA
↪"
}
```

La description d'une intervention peut être modifiée avec la requête :

```
PATCH /operations/{operation}
```

dont le body est :

```
[
  {
    "op":"replace",
    "path":"/description",
    "value":"Nouvelle description"
  }
]
```

Assignation

Pour assigner une intervention à un membre de l'organisation, l'administrateur de l'organisation effectue la requête :

```
POST /operations/{operation}/assign
```

dont le body est :

```
{
  "member":"{member}",
}
```

où {member} est l'identifiant du membre à qui est assignée l'intervention.

Acceptation ou refus

Une fois assignée, l'intervention peut être acceptée ou refusée, soit par la personne à qui l'intervention a été assignée, soit par un administrateur de l'organisation. Pour accepter l'intervention :

```
PATCH /operations/{operation}/state
```

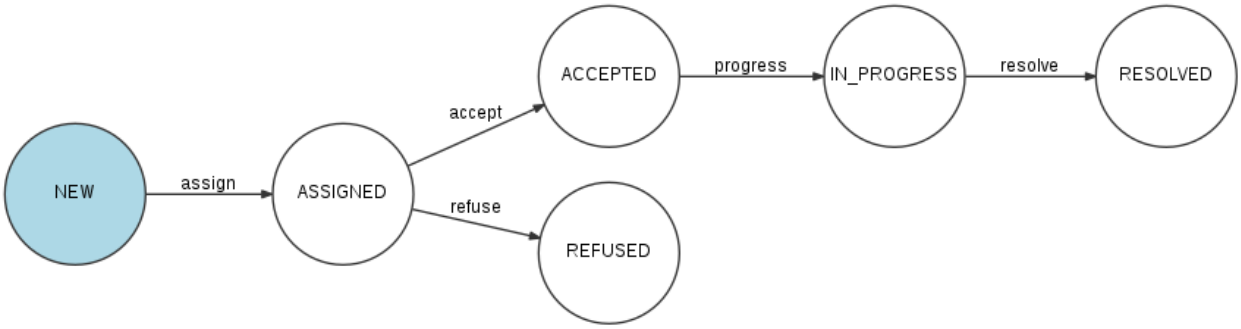
dont le body est :

```
[
  {
    "op":"replace",
    "path":"/transition",
    "value":"accept"
  }
]
```

Intervention en cours et clôture

Une fois acceptée, l'intervention peut-être passée "en cours" puis "clôturée", soit par la personne à qui l'intervention a été assignée, soit par un administrateur de l'organisation.

Résumé du cycle de vie d'une intervention



Commentaires

Il est possible de commenter une intervention :

```
POST /operations/{operation}/comments
```

dont le body est :

```
{
    "text":"Mon commentaire"
}
```

Pour récupérer tous les commentaires d'une intervention :

```
GET /operations/{operation}/comments
```

Logs d'une intervention

Un administrateur d'organisation peut consulter l'historique d'une intervention avec :

```
GET /operations/{operation}/logs
```

Délégation de rapports

Un administrateur d'une organisation peut déléguer un rapport à l'une des organisations partenaires.

Voir : *Gestion des partenariats*

Pour déléguer un rapport, un administrateur de l'organisation effectue la requête :

```
POST /organizations/{organization}/delegates
```

où {organization} est l'identifiant de l'organisation **courante** (dont le membre est administrateur).

Exemple :

```
{
  "report":"cb7118b5-a821-4cf2-9475-0c0d0efdb8d0",
  "organization":"a31d9ab7-9476-45f2-8cc7-033bf40bbcfa"
}
```

où a31d9ab7-9476-45f2-8cc7-033bf40bbcfa est l'identifiant de l'organisation à laquelle le rapport est délégué.

Déléguer un rapport ne signifie pas que ce rapport est simplement transmis. En effet, le rapport initial n'est pas modifié ni transféré, mais un nouveau rapport "enfant" est créé et attribué à l'organisation partenaire. Ce rapport enfant sera traité par l'organisation partenaire de la même façon que le rapport initial : changements de statuts, interventions, assignations des interventions, etc, jusqu'à sa clôture.

L'organisation partenaire peut elle-même déléguer le rapport à l'une de ses partenaires et ainsi de suite. Pour qu'un rapport puisse être clôturé, il est obligatoire que le rapport enfant, s'il existe, ait été préalablement clôturé par l'organisation partenaire.

Export des rapports

Un administrateur d'organisation peut exporter tous les rapports de son organisation au format Excel :

```
POST /organizations/{organization}/reports/exports
```

Une archive contenant le fichier Excel listant tous les rapports et les images associées à ces rapports est alors envoyé par email à l'administrateur authentifié.

Applications

Une application au coeur du service Keyclic permet de cloisonner les remontées d'observation suivant des domaines applicatifs. Cela se traduit par l'utilisation d'applications ou de sites internet spécifiques à certains métiers. Pour l'application éponyme du service, les applications sont « Keyclic » pour les smartphones et le site internet https://app.keyclic.com pour les navigateurs.

Exemple : Il existe d'autres applications utilisant le service Keyclic, notamment l'application « Jaidemaville ». Depuis l'application « Jaidemaville », il est impossible de remonter une observation à l'application « Keyclic » (et inversement) et il est impossible de lister les observations dédiées à l'application « Keyclic » (et inversement).

Administrateur d'application

L' « Administrateur d'application » est un statut particulier donné à un utilisateur du service Keyclic qui a la possibilité de modérer les observations d'une application avant que celles-ci soient transmises, sous forme de rapports, aux organisations concernées. Cette modération permet de filtrer toutes observations avec du contenu enfreignant les règles d'utilisation d'une application ou enfreignant la législation d'un pays (contenu pédopornographique, incitation à la haine, etc).

Note : Un « Administrateur d'application » n'est pas obligatoirement membre d'une organisation et ne peut modérer les observations que d'une seule et unique application, une application peut avoir plusieurs « Administrateur d'application ».

Notifications

Le service Keyclic peut notifier des utilisateurs après que certaines actions sont réalisées.

Type des notifications émises suivant les actions

Action	Destinataire	Type de notification
Création de rapport	Administrateurs de l'organisation	Email + Push smartphone
Clôture du rapport	Émetteur de l'observation (dont le rapport est issu)	Push smartphone
Assignation d'une opération	Membre assigné à l'opération	Push smartphone
Exportation des rapports	Administrateur courant de la session	Email
Modération d'une observation	Administrateurs de l'application	Push smartphone