Documentation Easy2Pilot V8

API

API Version 4.0.0

Dark mode (index.php?dark=1)

Historique du document

Version : 4.0.0

Date : 11/03/2020

Auteur : Julien Bolle

Commentaire :

Création du document

1. Introduction

Le lien de l'API en production (remplacé par xxxx dans la suite du document) : **https://middleware-production.easy2pilot-v8.com/api/UUID_AGENCE/ (https://middleware-production.easy2pilot-v8.com/api/)**

L'**uuid agence** correspond à l'id unique de l'agence, il vous sera transmis par Easysolutions.

Le webservice est une API RESTful avec authentification, de ce fait il réceptionne des données selon les méthodes GET / POST / DELETE / PUT qui sont encodées au format JSON et retourne une réponse au même format.

L'URL de l'API ainsi que vos identifiants pour vous y connecter vous seront transmis par Easysolutions, si vous ne les avez pas reçus veuillez prendre contact directement avec nous.

Scénario courant :

Le site internet doit passer par l'API pour récupérer ou modifier des informations.

Si l'api renvoie une erreur 401, le site internet refait une demande d'authentification et reçoit un nouveau token d'accès.

Retour :

La structure de retour du WebService est toujours un tableau sous la même forme :

Status : code de retour de la requête

Error : message d'erreur s'il y en a un

Data : Données récupérées pour cette requête (absent en cas d'erreur)

Codes de retour :

200 : Ok.

400 : Erreurs diverses.

401 : Problème d'autorisation.

404 : Ressource inconnue / route inconnue.

500 : Problème de serveur ou de base de données.

2. Authentification

La cinématique d'authentification s'appuie sur le type d'autorisation Client credentials grant.

Les informations principales dont vous avez besoin pour vous authentifier : l'url d'appel contenant l'UUID de l'agence, le login et le mot de passe associé.

Une fois ces éléments en votre possession, il vous faudra faire une demande de token afin de bénéficier d'un accès temporaire à l'api. Le token a une validité de 30 minutes, si celui-ci expire il vous faudra à nouveau faire une requête d'identification pour continuer à utiliser le WebService.

L'application s'authentifie auprès de Easysolutions et demande un access token. Si l'application est autorisée, Easysolutions génère et transmet un access token.

2.1. Récupérer un token :

Point d'accès : **xxxxx/token**

Méthode : **POST**

Les identifiants sont renseignés dans la passerelle ApiV4 de votre logiciel.

Détail des paramètres :

Paramètre(s)	**Valeur**
En-tête(s)	**Valeur**
Content-Type	application/json;charset=UTF-8
Corps de la requête	**Valeur**
login	Votre identifiant
password	Votre clé secrète

Exemple d'appel :

```
POST /api/UUID_AGENCE/token
Content-Type: application/json;charset=UTF-8

{
  "login": "[identifiant]"
  "password": "[clé secrète]",
}
```

Description de la réponse :

En-tête(s)	**Valeur**
Content-Type	application/json;charset=UTF-8
Cache-Control	no-store
Pragma	no-cache
Corps de la réponse	**Valeur**
token	Valeur de l'access token généré

Exemple de retour :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache

{
  "data": {
      "token": "5432103eedb233ba17906a109123456"
  },
  "status": 200,
  "message": ""
}
```

Cas d'erreur possibles :

Identifiant client et/ou mot de passe erroné ou absent :

```
HTTP 200 Ok
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache

{
    "status": 500,
    "message": "Identifiants de passerelle incorrects"
}
```

3. Requêtes Api Portail

3.1. Annonce - Liste des annonces publiées sur la passerelle :

Point d'accès : **xxxxx/annonces**

Méthode : **GET**

Détail des paramètres :

Champ	Type	Description
token	String	Votre access token
urlphotos	Boolean	Récupérer l'url des photos de l'annonce
filters	Array of objects	Recherche d'annonces.
key	String	Champ dans lequel rechercher
value	String	Valeur à rechercher pour le champ
compare	String	Opérateur à utiliser **=, LIKE, >, <, >=, <=, IN, IS NOT NULL**

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "token":"5432103eedb233ba17906a109123456",
    "urlphotos":true,
    "filters":[{
        "key":"id","value":"520","compare":"="
    }]
}
```

Description de la réponse :

En-tête(s)	Valeur	Description
Content-Type	application/json;charset=UTF-8	
Corps de la réponse	**Valeur**	**Description**
data	Array	Liste des annonces.

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": [
       {...}
   ],
   "status": 200,
   "message": ""
}
```

3.2. Annonce - Structure de retour de l'annonce :

Point d'accès : **xxxxx/structure**

Méthode : **GET**

Détail des paramètres :

Champ	Type	Description

token	String	Votre access token

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "token":"5432103eedb233ba17906a109123456"
}
```

Description de la réponse :

En-tête(s)	**Valeur**	**Description**
Content-Type	application/json;charset=UTF-8	
Corps de la réponse	**Valeur**	**Description**
data	Object	Données
hash	String	Hash md5 de la structure
structure	Object	Structure complète d'une annonce.
type	String	Type de la valeur renvoyée (string, boolean, integer, ...).

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": {
         "hash":"846a06edab52bc177a69f3077d65fb2e",
         "structure":{
             "chauffage":{
                 "climatisation":{
                     "type":"boolean"
                 }
             },
             {...}
         }
   },
   "status": 200,
   "message": ""
}
```

3.3. Annonce - Ajout d'une annonce sur la passerelle :

Point d'accès : **xxxxx/addAnnonce**

Méthode : **POST**

Détail des paramètres :

Champ	Type	Description
passerelle	Object	Vos identifiants (les mêmes que pour le token).
login	String	Votre identifiant
password	String	Votre clé secrète
annonces	Array of objects	Liste des annonces à insérer.
...	Object	La structure de l'annonce est récupérable (3.2)

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "passerelle":{
        "login":"[identifiant]",
        "password":"[clé secrète]"
    },
    "annonces":[
        {...},
        {...}
    ]
}
```

Description de la réponse :

En-tête(s)	Valeur	Description
Content-Type	application/json;charset=UTF-8	
Corps de la réponse	**Valeur**	**Description**
data	Array of object	Annonces créées (dans le même ordre que la requête).
origine_id	String	Id de l'annonce créée.

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": [
       {
         "origine_id":"1234"
       },
       {
         "origine_id":"1235"
       }
   ],
   "status": 200,
   "message": ""
}
```

3.4. Annonce - Suppression d'une annonce sur la passerelle :

Point d'accès : **xxxxx/removeAnnonce**

Méthode : **POST**

Détail des paramètres :

Champ	Type	Description
passerelle	Object	Vos identifiants (les mêmes que pour le token).
login	String	Votre identifiant
password	String	Votre clé secrète
annonces	Array of objects	Liste des annonces à insérer.
type_annonce	String	Type de l'annonce (lots ou batiments)
id	String	Id de l'annonce

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "passerelle":{
        "login":"[identifiant]",
        "password":"[clé secrète]"
    },
    "annonces":[
        {
            "type_annonce":"lots"
            "id":"1234"
        },
        {
            "type_annonce":"lots"
            "id":"1235"
        }
    ]
}
```

Description de la réponse :

En-tête(s)	Valeur	Description

Content-Type	application/json;charset=UTF-8

Corps de la réponse	Valeur	Description
data	Array of object	Annonces supprimées.
origine_id	String	Id de l'annonce supprimée.

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": [
       {
         "origine_id":"1234"
       },
       {
         "origine_id":"1235"
       }
   ],
   "status": 200,
   "message": ""
}
```

3.5. Nombre intéressés - Nombre des personnes intéressées par un ensemble de critères :

Point d'accès : **xxxxx/nbInteresses**

Méthode : **GET**

Détail des paramètres :

Champ	Type	Description
token	String	Votre access token
type_transaction	String	Type de transaction (vente, location)
criteres	Array of objects	Critères de recherche.
key	String	Critère à rechercher

value	String	Valeur à rechercher (parfois non présent, dépend du critère, voir exemple ci-dssous)
value_min	String	Valeur minimum à rechercher (parfois non présent, dépend du critère, voir exemple ci-dssous)
value_max	String	Valeur maximum à rechercher (parfois non présent, dépend du critère, voir exemple ci-dssous)

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "token":"5432103eedb233ba17906a109123456",
    "type_transaction":"vente",
    "criteres":[
        {
            "key":"nature",
            "value":"appartement"
        },
        {
            "key":"zone",
            "value":"Bonnevoie"
        },
        {
            "key":"budget",
            "value_min":"0",
            "value_max":"999999"
        },
        {
            "key":"nb_chambres",
            "value_min":"0",
            "value_max":"5"
        },
        {
            "key":"surface",
            "value_min":"0",
            "value_max":"200"
        }
    ]
}
```

Description de la réponse :

En-tête(s)	Valeur	Description
Content-Type	application/json;charset=UTF-8	

Corps de la réponse	Valeur	Description
data	Object	Données.
total	Integer	Nombre d'intéressés
bureaux	Array of Objects	Détail par bureau
computed_name	String	Nom du bureau
computed_address	String	Adresse du bureau
street_number	String	Numéro de rue du bureau
route	String	Rue du bureau
postal_code	String	Code postal du bureau
locality	String	Localité du bureau
country	String	Pays du bureau
lat	String	Latitude du bureau
lng	String	Longitude du bureau
nb_interesses	Integer	Nombre d'intéressés dans ce bureau

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": {
         "total": 17,
         "bureaux": [
             {
                 "computed_name": "Agence de la Gare",
                 "computed_address": "24, Rue de la Gare L-1234 Luxembourg, Luxembc
                 "street_number": "24",
                 "route": "Rue de la Gare",
                 "postal_code": "L-1234",
                 "locality": "Luxembourg",
                 "country": "Luxembourg",
                 "lat": "49.0000000",
                 "lng": "2.0000000",
                 "nb_interesses": 15,
             },
             {
                 "computed_name": "Agence du Centre",
                 "computed_address": "42, Rue du Centre-Ville L-1234 Luxembourg, Lu
                 "street_number": "42",
                 "route": "Rue du Centre-Ville",
                 "postal_code": "L-1234",
                 "locality": "Luxembourg",
                 "country": "Luxembourg",
                 "lat": "49.0000000",
                 "lng": "2.0000000",
                 "nb_interesses": 2,
             }
         ]
   },
   "status": "200",
   "message": ""
 }
```

3.6. Callback - Ajouter un callback concernant la passerelle :

Point d'accès : **xxxxx/addCallback**

Méthode : **POST**

Description : **Les callbacks sont les messages laissés par les clients à destination de l'agence. Ils permettent de signaler l'intérêt du client pour un bien.**

Détail des paramètres :

Champ	Type	Description

token	String	Votre access token
bien	Object	Les infos du bien.
id_e2p	String	Id Z16Pro du bien concerné (envoyé avec l'annonce)
client	Object	Les infos du client.
nom	String	Nom du client
prenom	String	Prénom du client
tel	String	Téléphone du client
email	String	Email du client
adresse	String	Adresse du client
code_postal	String	Code postal du client
ville	String	Ville du client
recherche	Object	**Facultatif**. Permet d'indiquer la recherche faite par le client pour trouver le bien.
type	String	Type de transaction (vente, location)
nature	String	Nature du bien
surface	String	Surface du bien (indiquer un entier dans une stirng)
chambres	String	Nb de chambres du bien (indiquer un entier dans une stirng)
budget_max	String	Budget max du bien (indiquer un entier dans une stirng)
budget_min	String	Budget min du bien (indiquer un entier dans une stirng)
message	String	Message du client.

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "token":"5432103eedb233ba17906a109123456",
    "bien": {
        "id_e2p": "3244"
    },
    "client": {
        "nom": "Doe",
        "prenom": "John",
        "tel": "0000",
        "email": "john@doe.com",
        "adresse": "22, rue du Soleil",
        "code_postal": "57000",
        "ville": "Metz"
    },
    "recherche": {
        "type": "vente",
        "nature": "Appartement",
        "surface": "35",
        "chambres": "1",
        "budget_max": "450000",
        "budget_min": "0"
    },
    "message": "Merci de me recontacter rapidement."
}
```

Description de la réponse :

En-tête(s)	Valeur	Description
Content-Type	application/json;charset=UTF-8	
Corps de la réponse	**Valeur**	**Description**
data	Array	Tableau vide.

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": [],
   "status": 200,
   "message": ""
}
```

(en construction) 3.7. Annonce - Retour de publication sur la passerelle :

Point d'accès : **xxxxx/addAnnonceFeedback**

Méthode : **POST**

Description : **C'est avec cette route que la passerelle peut renvoyer les messages d'erreur concernant une publication. Ils seront utilisés pour informer plus précisémment l'utilisateur sur la nature de l'erreur.**

Détail des paramètres :

Champ	Type	Description
token	String	Votre access token
id_e2p	String	Id easy2pilot du bien concerné (envoyé avec l'annonce)
code	Integer	Code d'erreur (voir la liste)
description	String	Description de l'erreur pour aider l'utilisateur à corriger. Par exemple le(s) champ(s) posant problème ainsi que la valeur envoyée et ce qui était attendu.
datetime	String	Optionnel. Date et heure de l'erreur (YYYY-MM-DD HH:mm:ss)

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "token":"5432103eedb233ba17906a109123456",
    "id_e2p": "3244",
    "code": 203,
    "description": "La nature 'agence immobilière' n'existe pas dans notre base de
    "datetime": "2021-01-31 10:00:00"
}
```

Description de la réponse :

En-tête(s)	Valeur	Description
Content-Type	application/json;charset=UTF-8	
Corps de la réponse	**Valeur**	**Description**
data	Array	Tableau vide.

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": [],
   "status": 200,
   "message": ""
}
```

Codes d'erreur possibles :

Code Description

100: Malformé

101: Structure de l'annonce incorrecte

102: Format de l'annonce incorrect (json, xml, ...)

103: Un champ n'a pas le format attendu

200: Données incorrectes

201: Champ obligatoire manquant

202: Correspondance de ville incorrecte ou manquante

203: Correspondance de nature incorrecte ou manquante

300: Indisponibilité serveur

301: Le serveur distant ne répond pas

302: Réponse du serveur distant vide ou inattendue

303: Page introuvable

3.8. Rdv - Liste des disponibilités :

Point d'accès : **xxxxx/listRdvClientDisponible**

Méthode : **GET**

Détail des paramètres :

Champ	Type	Description
token	String	Votre access token
date_debut	Datetime	Date début recherche (YYYY-MM-DD HH:mm:ss)
date_fin	Datetime	Date fin recherche (YYYY-MM-DD HH:mm:ss)
commercial_id	String	Identifiant du commercial

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "token":"5432103eedb233ba17906a109123456",
    "date_debut": "2021-01-25 08:00:00",
    "date_fin": "2021-01-29 17:00:00"
}
```

Description de la réponse :

En-tête(s)	Valeur	Description
Content-Type	application/json;charset=UTF-8	

Corps de la réponse	Valeur	Description
data	Array	Liste des disponibilités.
date_debut	Datetime	Date début dispo (YYYY-MM-DD HH:mm:ss)
date_fin	Datetime	Date fin dispo (YYYY-MM-DD HH:mm:ss)
commercial	Object	Commercial disponible
id	String	Identifiant du commercial
nom	String	Nom du commercial

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": [
       "date_debut": "2021-01-25 08:00:00",
       "date_fin": "2021-01-25 10:00:00",
       "commercial": {
           "id": "127",
           "nom": "Jean Dupont"
       }
   ],
   "status": 200,
   "message": ""
}
```

3.9. Rdv - Ajout d'un rdv client :

Point d'accès : **xxxxx/addRdvClient**

Méthode : **POST**

Détail des paramètres :

Champ	Type	Description
token	String	Votre access token
date_debut	Datetime	Date début (YYYY-MM-DD HH:mm:ss)
date_fin	Datetime	Date fin (YYYY-MM-DD HH:mm:ss)
remarque	String	Remarque du client

commercial_id	String	Identifiant du commercial
bien_id	String	Id easy2pilot du bien concerné (envoyé avec l'annonce) dans info/id
client	Object	Les infos du client.
nom	String	Nom du client
prenom	String	Prénom du client
tel	String	Téléphone du client
email	String	Email du client
adresse	String	Adresse du client
code_postal	String	Code postal du client
ville	String	Ville du client

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
{
    "token":"5432103eedb233ba17906a109123456",
    "date_debut": "2021-01-25 08:00:00",
    "date_fin": "2021-01-25 10:00:00",
    "remarque": "J'apporterai les documents dont nous avons parlé.",
    "commercial_id": "127",
    "bien_id": "24",
    "client": {
        "nom": "Doe",
        "prenom": "John",
        "tel": "0000",
        "email": "john@doe.com",
        "adresse": "22, rue du Soleil",
        "code_postal": "57000",
        "ville": "Metz"
    },
}
```

Description de la réponse :

En-tête(s)	Valeur	Description
Content-Type	application/json;charset=UTF-8	

Corps de la réponse	Valeur	Description
data	Array	Tableau vide.

Success-Response :

```
HTTP 200 OK
 Content-Type: application/json;charset=UTF-8
 Cache-Control: no-store
 Pragma: no-cache
 {
   "data": [],
   "status": 200,
   "message": ""
}
```

4. Webhooks

Nous utilisons un système de webhook permettant de signaler une modification d'annonce sur l'API.
Si vous souhaitez être averti lors d'un changement, il faut au préalable que le client utilisant le logiciel Z16Pro ait renseigné une **URL callback** dans la configuration de la passerelle API.
L'API appelera cette URL à chaque fois qu'un changement d'annonce sera effectué.
Vous avez également la possibilité d'ajouter une clé d'API afin d'identifier l'agence ayant émis ou retiré une annonce.
Pour ce faire, veuillez suivre les étapes ci-dessous ou demander à votre client de les suivre :

1. Dans le logiciel Z16Pro du client, rendez-vous dans **Configuration / Passerelles.**
2. Identifiez la passerelle vers votre plateforme, le format doit être **V4 API**.
3. Renseignez le champ URL callback avec l'URL de votre plateforme.

4. Vous pouvez en plus renseigner une clé d'API en remplissant le champ **API KEY**, un header "X-API-TOKEN" sera ajouté à la requête.

4.1. Webhooks - Publication/retrait d'une annonce :

Point d'accès : **L'URL callback concernant votre plateforme définie dans le logiciel Z16Pro**

Méthode : **POST**

Header : **Si une API Key est définie, un header "X-API-TOKEN" contenant la clé sera ajouté à la requête**

Description : **Cette requête est envoyée à chaque changement d'une annonce (publication ou retrait)**

Détail des paramètres :

Champ	Type	Description
multiple	Integer	Valeur toujours envoyée : 1
data	Array of objects	Les id des annonces impactées et la modification faite.
id	Integer	Id de l'annonce concernée
action	String	"add" ou "delete", signifiant respectivement que l'annonce a été ajoutée ou retirée

Exemple d'appel :

```
HTTP 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
X-API-TOKEN: AGC0001
{
    "multiple": 1,
    "data": [
        {"id": 1, "action": "add"},
        {"id": 2, "action": "add"},
        {"id": 3, "action": "add"},
        {"id": 4, "action": "delete"},
    ]
}
```

Dans l'exemple, la requête signifie : les annonces 1 à 3 viennent d'être publiées, l'annonce 4 vient d'être dépubliée.
Vous pouvez donc délencher à la suite un appel à la route 3.1. Annonces par exemple.