UTILISATION DE L API MARKETPLACE

PRÉSENTATION

Cet article a pour but de présenter l'API permettant d'automatiser la publication d'add-ons pour BlueMind et de leur versions sur le MarketPlace.

Cette API est elle-même téléchargeable sur le MarketPlace BlueMind.

> **Clé d'API**
>
> Pour utiliser l'API, vous aurez besoin de votre clé d'API disponible dans l'interface d'administration : https://marketplace.blue-mind.net/admin/. Cette clé d'API n'est modifiable que par l'administrateur du site web.

MES PLUGINS ET LEURS VERSIONS

Pour savoir quels sont les plugins que l'on a publié, leur statut (publié ou non), quelle est leur version courante, et de nombreuses autres informations, il faut faire une requête GET sur l'URL : https://marketplace.blue-mind.net/addons/api/plugins/. Cette requête renvoie les informations au format JSON.
La **clé d'API** associée à l'utilisateur faisant la demande **doit être incluse en tant que *header*** de la requête sous la forme : "api-key: la_cle_d_api".

Voici un exemple de code en Python qui permet de faire cette requête :

GET /api/plugins/

```
import requests

headers = {
    'api-key': 'your_api_key'
}

URL = 'https://marketplace.blue-mind.net/addons/api/plugins/'

req = requests.get(URL, headers=headers, verify=False)
print "Status code : " + str(req.status_code)
if req.text:
    print "Contents : " + str(json.dumps(req.json(), indent=4, sort_keys=True))
```

> N'oubliez pas de remplacer la clé d'API par la clé associée à votre compte sur le MarketPlace.

Des exemples d'implémentation dans d'autres langages sont fournis en bas de cette page. Si vous souhaitez construire la requête dans un langage qui n'y serait pas, vous pouvez développer un exemple de trame générée :

GET /addons/api/plugins/ HTTP/1.1

Host: marketplace.blue-mind.net

Content-Length: 51

Content-Type: application/x-www-form-urlencoded

Accept-Encoding: gzip, deflate

Accept: */*

User-Agent: python-requests/2.3.0 CPython/2.7.5+ Linux/3.11.0-12-generic

api-key: your_api_key

UN PLUGIN SPÉCIFIQUE

Pour avoir des informations sur un plugin spécifique (dont on connait l'ID), on peut faire une requête GET sur l'url : https://marketplace.blue-mind.net/addons/api/plugin/{id}/ où {id} est l'ID du plugin considéré. Cette requête renverra des informations sur le plugin désiré au format JSON si l'utilisateur est propriétaire de ce plugin, sinon elle renverra une erreur 403.
Cette requête demande aussi d'**inclure la clé d'API en tant que *header*** de la requête sous la forme : "api-key: la_cle_d_api".

Voici un exemple de code en Python qui permet de faire cette requête :

GET /api/plugin/1

```
import requests

headers = {
    'api-key': 'your_api_key'
}

URL = 'https://marketplace.blue-mind.net/addons/api/plugin/1/'

req = requests.get(URL, headers=headers, verify=False)
print "Status code : " + str(req.status_code)
if req.text:
    print "Contents : " + str(json.dumps(req.json(), indent=4, sort_keys=True))
```

Des exemples d'implémentation dans d'autres langages sont fournis en bas de cette page. Si vous souhaitez construire la requête dans un langage qui n'y serait pas, vous pouvez développer un exemple de trame générée :

GET /addons/api/plugin/1/ HTTP/1.1

Host: marketplace.blue-mind.net

Content-Length: 51

Content-Type: application/x-www-form-urlencoded

Accept-Encoding: gzip, deflate

Accept: */*

User-Agent: python-requests/2.3.0 CPython/2.7.5+ Linux/3.11.0-12-generic

api-key: your_api_key

AJOUTER UN PLUGIN

Pour ajouter un plugin, il faut construire une requête POST contenant les informations pour sa création au format JSON, et envoyer cette requête à l'URL https://marketplace.blue-mind.net/addons/api/plugins/. Ces informations sont :

Nom de la clé	Obligatoire	Description de la clé
api_key	oui	Identifiant de l'utilisateur qui va créer un plugin
name	oui	Nom du plugin à créer
shortdesc	oui	Description courte du plugin à créer (affichée sous le titre)
description	oui	Description longue du plugin à créer (affichée sous les captures d'écran)
license	oui	Licence de publication du plugin à créer
installation_instructions	oui	Démarche à suivre pour installer le plugin
home_url	non	URL du site web associé au plugin

La requête étant de type "*multipart/form-data*", il a été choisi que les données JSON soient associées au nom "json". Si vous ne respectez pas cette convention, la requête n'aboutira pas.

Vous pouvez aussi ajouter des images liées au plugin en les joignant en tant que fichiers associés à la requête POST. Le nom des images doit respecter la convention suivante :

Nom	Description
thumbnail	L'image associée au logo du plugin.
*	Toute image avec un autre nom sera importée en tant que capture d'écran pour le plugin.

Le nombre de screenshots n'est pas limité.

Voici un exemple de code en Python qui crée un plugin avec un logo et deux captures d'écran (*screenshots*) :

POST /api/plugins/

```
import requests, json

data = {
  'api_key': 'your_api_key',
  'name': 'Super plugin',
  'shortdesc': 'A short description',
  'description': 'A long description',
  'license': 'A license',
  'installation_instructions': 'Some instructions',
  'home_url': "http://www.blue-mind.net"
}

files = {
    'thumbnail': open('/home/user/Images/my_thumbnail.jpg', 'rb'),
    'screen1': open('/home/user/Images/my_screenshot1.png', 'rb'),
    'screen2': open('/home/user/Images/my_screenshot2.png', 'rb')
}

URL = 'https://marketplace.blue-mind.net/addons/api/plugins/'

req = requests.post(URL, files=files, data={'json': json.dumps(data)}, verify=False)
print "Status code : " + str(req.status_code)
```

Des exemples d'implémentation dans d'autres langages sont fournis en bas de cette page. Si vous souhaitez construire la requête dans un langage qui n'y serait pas, vous pouvez développer un exemple de trame générée :

POST /addons/api/plugins/ HTTP/1.1
Host: marketplace.blue-mind.net
Content-Length: 279271
Content-Type: multipart/form-data; boundary=1824dd5f14f14e57bb2b2a27424628db
Accept-Encoding: gzip, deflate
Accept: */*
User-Agent: python-requests/2.3.0 CPython/2.7.5+ Linux/3.11.0-12-generic

--1824dd5f14f14e57bb2b2a27424628db
Content-Disposition: form-data; name="json"
{"home_url": "http://www.blue-mind.net", "name": "Super plugin", "license": "A license", "installation_instructions": "Some instructions", "shortdesc": "A short description", "api_key": "your_api_key", "description": "A long description"}

--1824dd5f14f14e57bb2b2a27424628db
Content-Disposition: form-data; name="screen2"; filename="my_screenshot2.png"
[... DATA ...]
--1824dd5f14f14e57bb2b2a27424628db--
--1824dd5f14f14e57bb2b2a27424628db
Content-Disposition: form-data; name="screen1"; filename="my_screenshot1.png"
[... DATA ...]
--1824dd5f14f14e57bb2b2a27424628db--
--1824dd5f14f14e57bb2b2a27424628db
Content-Disposition: form-data; name="thumbnail"; filename="my_thumbnail.jpg"
[... DATA ...]
--1824dd5f14f14e57bb2b2a27424628db--

AJOUTER UNE VERSION DE PLUGIN

Pour ajouter une version de plugin à un plugin qui vous appartient, il faut construire une requête POST contenant les informations pour la création de la version au format JSON, et envoyer cette requête à l'URL : https://marketplace.blue-mind.net/addons/api/plugin_version/ . Ces informations sont :

Nom de la clé	Obligatoire	Description de la clé
api_key	oui	Identifiant de l'utilisateur qui va créer le plugin

plugin_name	oui	Nom du plugin associé à la version
version	oui	Numéro de la nouvelle version du plugin
target_bm_versions	oui	Versions de BlueMind concernées par cette nouvelle version
release_notes	oui	Informations sur la raison de cette nouvelle version

La requête étant de type "*multipart/form-data*", il a été choisi que les données JSON soient associées au nom "json". Si vous ne respectez pas cette convention, la requête n'aboutira pas.

Vous **devez associer un fichier** à cette nouvelle version (le fichier à télécharger) en le joignant à la requête POST sous le nom de "package".

Voici un exemple de code en Python qui permet d'ajouter la version 0.2 pour le plugin "Super plugin" :

POST /api/plugin_version/

```
import requests, json

data = {
  'api_key': 'your_api_key',
  'plugin_name': 'Super plugin',
  'version': '0.2',
  'target_bm_versions': '3.x',
  'release_notes': 'Brand new release'
}

files = {
    'package': open('/home/user/Files/super_plugin.jar', 'rb')
}

URL = 'https://marketplace.blue-mind.net/addons/api/plugin_version/'

req = requests.post(URL, files=files, data={'json': json.dumps(data)}, verify=False)
print "Status code : " + str(req.status_code)
```

La nouvelle version sera refusée si elle ne comporte pas de fichier, ainsi que si le plugin ne vous appartient pas (sauf pour un administrateur du MarketPlace).

Des exemples d'implémentation dans d'autres langages sont fournis en bas de cette page. Si vous souhaitez construire la requête dans un langage qui n'y serait pas, vous pouvez développer un exemple de trame générée :

POST /addons/api/plugin_version/ HTTP/1.1

Host: marketplace.blue-mind.net

Content-Length: 1191498

Content-Type: multipart/form-data; boundary=947a3f6396a244a99118fd5a3faa5204

Accept-Encoding: gzip, deflate

Accept: */*

User-Agent: python-requests/2.3.0 CPython/2.7.5+ Linux/3.11.0-12-generic

--947a3f6396a244a99118fd5a3faa5204

Content-Disposition: form-data; name="json"

{"api_key": "your_api_key", "release_notes": "Ceci est une nouvelle release", "version": "0.2", "plugin_name": "Super plugin", "target_bm_versions": "3.x"}

--947a3f6396a244a99118fd5a3faa5204

Content-Disposition: form-data; name="package"; filename="super_plugin.jar"

[... DATA ...]

--947a3f6396a244a99118fd5a3faa5204--

UTILISATION DE L'API DANS DIVERS LANGAGES

En Python

Voici un script complet qui reprend les scripts précédents et permet de faire du POST ou du GET (en fonction de ce qui est commenté) sur le MarketPlace :

Script complet

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import requests, json

WEBSITE = "https://marketplace.blue-mind.net/"
API_KEY = "your_api_key"

#### POST PLUGIN
data = {
  'api_key': API_KEY,
  'name': 'Super plugin',
  'shortdesc': 'A short description',
  'description': 'A long description',
  'license': 'A license',
  'installation_instructions': 'Several instructions !',
  'home_url': "http://www.blue-mind.net"
}

files = {
    'thumbnail': open('/home/user/Images/thumbnail.jpg', 'rb'),
    'screen1': open('/home/user/Images/screenshot1.png', 'rb'),
    'screen2': open('/home/user/Images/screenshot1.png', 'rb')
}

URL = WEBSITE + 'addons/api/plugins/'

#### POST PLUGIN VERSION
# data = {
#   'api_key': API_KEY,
#   'plugin_name': 'Super plugin',
#   'version': '0.2',
#   'target_bm_versions': '3.x',
#   'release_notes': 'On vient de faire une nouvelle release'
# }
#
# files = {
#     'package': open('/home/user/Files/super_plugin.jar', 'rb')
# }
#
# URL = WEBSITE + 'addons/api/plugin_version/'

#### POST REQUEST
req = requests.post(URL, files=files, data={'json': json.dumps(data)}, verify=False)


#### GET PLUGINS
# headers = {
#     'api-key': API_KEY
# }
#
# URL = WEBSITE + 'addons/api/plugins/'
#
# #### GET REQUEST
# req = requests.get(URL, headers=headers, verify=False)


#### POST OR GET RESPONSE
print "Status code : " + str(req.status_code)
if req.text:
    print "Contents : " + str(json.dumps(req.json(), indent=4, sort_keys=True))
```

Téléchargez le code source : api_python.py

En Java

Les extraits de code suivants donnent un aperçu de ce qu'il faut faire en Java pour avoir des requêtes GET et POST fonctionnelles. Ils utilisent les API **Apache HTTP** et **JSON-Simple.**

GET /addons/api/plugins

```
private static HttpClient httpClient = StaticTools.getHttpClient(true);
private static String API_KEY = "your_api_key";
private static String WEBSITE = "https://marketplace.blue-mind.net/";

private static void getPlugins() throws IOException {
    HttpGet httpGet = new HttpGet(WEBSITE + "/addons/api/plugins/");
    httpGet.addHeader("api-key", API_KEY);
    ResponseHandler<String> handler = new BasicResponseHandler();
    HttpResponse rep = httpClient.execute(httpGet);
    if (rep != null) {
        int statusCode = rep.getStatusLine().getStatusCode();
        System.out.println("Status code : " + statusCode);
        if (statusCode == 200) {
            System.out.println("Contents : " + handler.handleResponse(rep));
        }
    }
}
```

GET /addons/api/plugin/id

```
private static HttpClient httpClient = StaticTools.getHttpClient(true);
private static String API_KEY = "your_api_key";
private static String WEBSITE = "https://marketplace.blue-mind.net/";

private static void getPlugin(int id) throws IOException {
    HttpGet httpGet = new HttpGet(WEBSITE + "/addons/api/plugin/" + id);
    httpGet.addHeader("api-key", API_KEY);
    ResponseHandler<String> handler = new BasicResponseHandler();
    HttpResponse rep = httpClient.execute(httpGet);
    if (rep != null) {
        int statusCode = rep.getStatusLine().getStatusCode();
        System.out.println("Status code : " + statusCode);
        if (statusCode == 200) {
            System.out.println("Contents : " + handler.handleResponse(rep));
        }
    }
}
```

POST /addons/api/plugins

```
private static HttpClient httpClient = StaticTools.getHttpClient(true);
private static String API_KEY = "your_api_key";
private static String WEBSITE = "https://marketplace.blue-mind.net/";

@SuppressWarnings("unchecked")
private static void postPlugin() throws IOException {
    HttpPost httpPost = new HttpPost(WEBSITE + "/addons/api/plugins/");
    MultipartEntityBuilder builder = MultipartEntityBuilder.create();
    ContentType contentType = ContentType.getOrDefault(null);

    // Json part
    JSONObject j = new JSONObject();
    j.put("api_key", API_KEY);
    j.put("name", "Super plugin");
    j.put("shortdesc", "A short description");
    j.put("description", "A long description");
    j.put("license", "A license");
    j.put("installation_instructions", "Several instructions");
    j.put("home_url", "http://www.blue-mind.net");
    StringBody json = new StringBody(j.toString(), contentType);

    // Files part
    FileBody thumbnail = new FileBody(new File("/home/user/Images/thumbnail.jpg"));
    FileBody screen1 = new FileBody(new File("/home/user/Images/screenshot1.png"));
    FileBody screen2 = new FileBody(new File("/home/user/Images/screenshot2.png"));

    // Grouping
    builder.addPart("json", json);
    builder.addPart("thumbnail", thumbnail);
    builder.addPart("screen1", screen1);
    builder.addPart("screen2", screen2);

    httpPost.setEntity(builder.build());
    HttpResponse rep = httpClient.execute(httpPost);
    if (rep != null) {
        int statusCode = rep.getStatusLine().getStatusCode();
        System.out.println("Status code : " + statusCode);
    }
}
```

POST /addons/api/plugin_version

```
private static HttpClient httpClient = StaticTools.getHttpClient(true);
private static String API_KEY = "your_api_key";
private static String WEBSITE = "https://marketplace.blue-mind.net/";

@SuppressWarnings("unchecked")
private static void postPluginVersion() throws IOException {
    HttpPost httpPost = new HttpPost(WEBSITE + "/addons/api/plugin_version/");
    MultipartEntityBuilder builder = MultipartEntityBuilder.create();
    ContentType contentType = ContentType.getOrDefault(null);

    // Json part
    JSONObject j = new JSONObject();
    j.put("api_key", API_KEY);
    j.put("plugin_name", "Super plugin");
    j.put("version", "0.2");
    j.put("target_bm_versions", "3.x");
    j.put("release_notes", "On vient de faire une nouvelle release");
    StringBody json = new StringBody(j.toString(), contentType);

    // Files part
    FileBody pack = new FileBody(new File("/home/user/Files/super_plugin.jar"));

    // Grouping
    builder.addPart("json", json);
    builder.addPart("package", pack);

    httpPost.setEntity(builder.build());
    HttpResponse rep = httpClient.execute(httpPost);
    if (rep != null) {
        int statusCode = rep.getStatusLine().getStatusCode();
        System.out.println("Status code : " + statusCode);
    }
}
```

Téléchargez le code source : api_java.zip