ekipedia®

えきペディア MAP&バリアフリー施設情報

えきペディア®API

ご利用ガイド

Ver 1.23

目　次

1. えきペディア API とは？ 1
2. えきペディア API の利用方法 1
3. JavaScript 用 SDK ライブラリの利用方法 2
   3.1 SDK ライブラリの組み込み 2
   3.2 関数の説明 2
      ■init 関数 2
      ■api 関数 2
   3.2 リソースとパラメータ、レスポンス 2
      3.2.1 cities（地下鉄都市一覧） 4
      3.2.2 lines（都市別路線一覧） 4
      3.2.3 stations（路線別駅一覧） 5
      3.2.4 station（駅情報） 6
      3.2.5 exits（バリアフリー出入口情報） 7
      3.2.6 toilets（バリアフリー・トイレ情報） 9
4. えきペディア MAP の描画方法 11
   4.1 描画領域の確保 11
   4.2 関数の説明 12
      ■drawMap 関数 12
      パラメータ 12
      えきペディア MAP の描画サンプル 12
5. サムネイル画像の取得方法 14
   5.1 サムネイル画像の URL 14
6. えきペディア API を直接呼び出す方法 16
   6.1 API の URL 16
   6.2 リソースとパラメータ 16
   6.3 フォーマット 17
   6.4 API のレスポンス 17
   6.5 API の呼び出し例 19

1. えきペディア API とは？

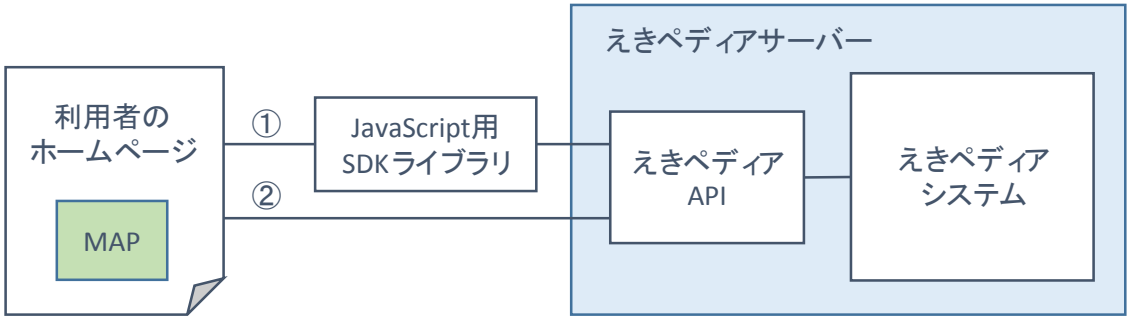
えきペディア API は、外部のプログラムから、えきペディアが提供する情報を取得するためのインターフェースです。この API を用いることによって、既存のホームページ上に、えきペディア MAP などを埋め込むことができます。
えきペディア API では、以下のような情報を取得することができます。

- 地下鉄都市一覧
- 都市別路線一覧
- 路線別駅一覧
- 駅情報（えきペディア MAP を含む）
- バリアフリー出入口情報
- バリアフリー・トイレ情報

2. えきペディア API の利用方法

えきペディア API を利用するには、以下の2つの方法があります。

① JavaScript 用 SDK ライブラリを介して利用する。
② えきペディア API を直接呼び出して利用する。



①の JavaScript 用 SDK ライブラリは、えきペディア MAP などを既存のホームページに簡単に組み込むために用意されています。
②のえきペディア API を直接呼び出す方法は、PHP などのサーバーサイドのプログラムからの利用に適しています。
ここでは、①の JavaScript 用 SDK ライブラリを用いる方法から説明します。②のえきペディア API を直接呼び出す方法は 6.で説明します。

3. JavaScript 用 SDK ライブラリの利用方法

3.1　SDK ライブラリの組み込み

以下のように、えきペディアの JavaScript 用 SDK ライブラリを、HTML ファイルに組み込みます。えきペディアサイトからダウンロードし、それをご自分のサイトにアップロードして組み込んでもかまいません。

```
<script src="http://ekipedia.jp/sdk/sdk_1.0.js"></script>
```

3.2 関数の説明

SDK ライブラリには以下の関数があります。

■init 関数

```
Ekipedia.init(コールバック関数);
```

ライブラリの初期化のために、必ず最初に呼び出してください。初期化が完了すると、コールバック関数が呼ばれます。通常、駅情報の取得や駅 MAP の描画は、このコールバック関数内で行います。

■api 関数

```
Ekipedia.api(リソース, パラメータ, コールバック関数);
```

えきペディアの API を呼び出し、リソースで指定した情報を取得します。要求するリソースによって、指定できるパラメータが異なります。

3.2 リソースとパラメータ、レスポンス

api 関数に指定できるリソースとパラメータを以下に示します。パラメータは

```
{ キー: 値 }
```

の形式で指定します。

リソース	意　味	指定できるパラメータ
cities	えきペディアが提供する地下鉄都市一覧。	指定できるパラメータはありません。
lines	指定された都市コードの路線一覧。	{city: 都市コード}
stations	指定された路線の駅一覧。	{line: 路線 ID}
station	指定された駅の情報。	{station: 駅 ID }
exits	指定された条件の出入口一覧。	{station: 駅 ID } または {type: 出入口仕様} 出入口仕様を指定した場合 {line: 路線 ID} または {city: 都市コード} がオプションとして指定できます。
toilets	指定された条件のバリアフリー・トイレ一覧。	{station: 駅 ID } または {equip: 設備コード} 設備コードを指定した場合 {line: 路線 ID} または {city: 都市コード} がオプションとして指定できます。

- **レスポンス**

Ekipedia.api 関数の呼び出しが終了するとコールバック関数が呼ばれ、パラメータとして以下のオブジェクトが渡されます。

response

プロパティ	意味	説　明
error	エラーメッセージ	呼び出しに失敗したとき、エラーメッセージが設定されます。成功した場合は設定されません。
data	取得したデータ	リソースごとに異なります。リソースごとのレスポンス.data を下記で説明します。

したがって、典型的な api 関数の呼び出し方は以下のようになります。(station リソースを取得する例)

```
<script>
    Ekipedia.init(function() {
        Ekipedia.api("station", "{id: 100}", function(response) {
            if (response.error) {
                console.log("エラーが発生しました。" + response.error);
            }
            else {
                // response.data を使う処理
                // ...
            }
        });
    });
</script>
```

以下に、リソースごとのリソース名、パラメータおよびレスポンス(レスポンス.data)を説明します。

3.2.1 cities(地下鉄都市一覧)

- **リソース**: cities
- **パラメータ**: 指定できるパラメータはありません。
- **レスポンス.data**:

えきペディアが提供する札幌から福岡までの9都市の情報が配列で渡されます。配列の1つの要素は、下記のプロパティを持つオブジェクトです。

response.data

プロパティ	意 味	値の例
code	都市コード	3
name	都市名(漢字)	東京
nameEn	都市名(ローマ字)	Tokyo

3.2.2 lines(都市別路線一覧)

- **リソース名**: lines

- **パラメータ：** 以下の形式で都市コードを指定します。

  {city: 都市コード}

- **都市コード：** 指定できる都市コードは以下のとおりです。

都市コード	都　市
1	札幌
2	仙台
3	東京
4	横浜
5	名古屋
6	京都
7	大阪
8	神戸
9	福岡

- **レスポンス.data：**

  指定された都市の路線一覧が配列で渡されます。配列の1つの要素は、下記のプロパティを持つオブジェクトです。

response.data

プロパティ	意　味	値の例
id	路線 ID	50
name	路線名（漢字）	銀座線
nameEn	路線名（ローマ字）	Ginza Line
orgName	事業体名（漢字）	東京メトロ
orgNameEn	事業体名（英語）	Tokyo Metro

3.2.3 stations（路線別駅一覧）

- **リソース名：** stations

- **パラメータ：** 以下の形式で路線 ID を指定します。

  {line: 路線 ID}

路線 ID は以下のページで調べることができます。

```
http://ekipedia.jp/aboutapi/eki_id
```

- **レスポンス.data**:

  指定された路線の駅一覧が配列で渡されます。配列の1つの要素は、下記のプロパティを持つオブジェクトです。

response.data

プロパティ	意 味	値の例
id	駅 ID	1340
name	駅名（漢字）	渋谷
nameEn	駅名（ローマ字）	Shibuya

3.2.4 station（駅情報）

各駅の駅情報を取得

- **リソース**: station

- **パラメータ**: 以下の形式で駅 ID を指定します。

  {station: 駅 ID}

- **駅 ID**

駅 ID は以下のページで調べることができます。

```
http://ekipedia.jp/aboutapi/eki_id
```

- **レスポンス.data**:

  指定された駅の情報が、下記のプロパティを持つオブジェクトとして渡されます。

response.data

プロパティ	意 味	値の例
id	駅 ID	1690
name	駅名（漢字）	東京
nameEn	駅名（ローマ字）	Tokyo
kana	駅名（ひらがな）	とうきょう
lineName	路線名（漢字）	丸ノ内線
lineNameEn	路線名（英語）	Marunouchi Line
orgName	事業体名（漢字）	東京メトロ
orgNameEn	事業体名（英語）	Tokyo Metro
lat	緯度	35.681753
lng	経度	139.764708
mapUrl	MAP の URL（注）	http://ekipedia.jp/ekipedia_map/…
mapWidth	MAP の幅（ピクセル単位）	1062
mapHeight	MAP の高さ（ピクセル単位）	1319

注）この MAP の URL は、API 呼び出し専用に生成された URL です。この URL を保存しておいて後で使用することはできません。
この URL にアクセスして得られるえきペディア MAP を描画する方法は、以下の 4.で説明します。

3.2.5 exits（バリアフリー出入口情報）

- **リソース**： exits

- **パラメータ**:

  {id: 駅 ID }
  または
  {type: 出入口仕様}

- **出入口仕様**

  出入口仕様は、駅の出入口がどのようにバリアフリーに対応しているかを表したもので、以下の番号を指定します。

出入口仕様	意　味
2	階段昇降機
3	エスカレータ
4	車イス対応エスカレータ
5	エレベータ
6	スロープ
7	段差なし

出入口仕様を指定した場合、オプションとして、さらに以下の条件を指定することができます。

{line: 路線 ID}
または
{city: 都市コード}

※路線 ID および都市コードは、3.2.3 および 3.2.2 で説明したものと同じです。
※路線 ID と都市コードの両方を指定した場合、都市コードは無視されます。

【例】大阪市営地下鉄御堂筋線で車イス対応エスカレーターの出入口情報を取得する。

```
Ekipedia.api("exits", { line: 280, type: 4 },
       function(response) { … }
);
```

- **レスポンス.data:**

指定された条件に合致する出入口情報の一覧が配列で渡されます。配列の1つの要素は、下記のプロパティを持つオブジェクトです。

プロパティ	意　味	値の例
cityName	都市名（漢字）	大阪
cityNameKana	都市名（ひらがな）	おおさか
cityNameEn	都市名（ローマ字）	Osaka
lineName	路線名（漢字）	御堂筋線
lineNameEn	路線名（ローマ字）	Midosuji Line
staId	駅 ID	120
staName	駅名（漢字）	動物園前

staNameKana	駅名（ひらがな）	どうぶつえんまえ
staNameEn	駅名（ローマ字）	Dobutsuen-mae
exitName	出入口名	2
exitType	出入口仕様	車イス対応エスカレータ
facilityName	施設名	なし
photos	トイレの写真配列	※下記に説明

photos 配列の各要素は、下記のプロパティを持つオブジェクトです。出入口ごとに、返される配列のサイズ（写真の枚数）は異なります（最大6枚）。

プロパティ	意　味	値の例
photoUrl	出入口写真の URL	http://ekipedia.jp/ekipediaimg/...
photoWidth	出入口写真の幅（ピクセル単位）	600
photoHeight	出入口写真の高さ（ピクセル単位）	450

3.2.6 toilets（バリアフリー・トイレ情報）

- **リソース**: toilets

- **パラメータ**:

  {id: 駅 ID }
  または
  {equip: 設備コード}

- **設備コード**

  設備コードは、トイレに備え付けられている設備を表したもので、以下のコードを指定します。

設備コード	意　味
1	オストメイト
2	介助ベッド
3	ベビーベッド
4	チャイルドシート

設備コードは1つだけではなく、複数個をカンマ(,)で区切って指定することができます。その場合は、カ

ンマで区切った設備コード全体を二重引用符で囲んでください。

【例】オストメイトと介助ベッドの設備があるトイレを検索する。

```
Ekipedia.api("toilets", { equip: "1,2" },
      function(response) { … }
);
```

設備コードを指定した場合、オプションとして、さらに以下の条件を指定することができます。

{line: 路線 ID}
または
{city: 都市コード}

※路線 ID および都市コードは、3.2.3 および 3.2.2 で説明したものと同じです。
※路線 ID と都市コードの両方を指定した場合、都市コードは無視されます。

【例】東京メトロ銀座線でベビーベッドのあるトイレを検索します。

```
Ekipedia.api("toilets", { line: 50, equip: 3 },
      function(response) { … }
);
```

- **レスポンス.data：**

指定された条件に合致するバリアフリー・トイレ情報の一覧が配列で渡されます。配列の1つの要素は、下記のプロパティを持つオブジェクトです。

プロパティ	意 味	値の例
cityName	都市名（漢字）	大阪
cityNameKana	都市名（ひらがな）	おおさか
cityNameEn	都市名（ローマ字）	Osaka
lineName	路線名（漢字）	御堂筋線
lineNameEn	路線名（ローマ字）	Midousuji Line
staId	駅 ID	60
staName	駅名（漢字）	梅田
staNameKana	駅名（ひらがな）	うめだ

staNameEn	駅名（ローマ字）	Umeda
toiletName	トイレ名	c
facilityName	施設名	阪神梅田
doorType	扉仕様	「手動式扉」「ボタン扉」「センサー扉」など
hasOstomate	オストメイト設備あり	true または false
hasCarebed	介助ベッドあり	true または false
hasBabybed	ベビーベッドあり	true または false
hasChildseat	チャイルドシートあり	true または false
photos	トイレの写真配列	※下記に説明

photos 配列の各要素は、下記のプロパティを持つオブジェクトです。トイレごとに、返される配列のサイズ（写真の枚数）は異なります（最大6枚）。

プロパティ	意 味	値の例
photoUrl	トイレの写真の URL	http://ekipedia.jp/ekipediaimg/…
photoWidth	トイレの写真の幅（ピクセル単位）	600
photoHeight	トイレの写真の高さ（ピクセル単位）	450

4. えきペディア MAP の描画方法

えきペディア MAP を描画するには、api 関数で当該駅の情報を取得し、そのレスポンスを以下に述べる drawMap 関数に指定して描画します。

4.1 描画領域の確保

えきペディアMAPを表示したい領域を タグで確保し、id 属性で独自の名前をつけます。またスタイルシート等で、幅と高さを与えてください。MAP はこの領域内に描かれますので、これは MAP のラッパーとなります。

```
<div id="station-map-area"></div>
```

4.2 関数の説明

■drawMap 関数

```
Ekipedia.drawMap(api 関数からのレスポンス.data, 描画要素, [パラメータ]);
```

api 関数で station リソースを指定して取得した情報に基づいて、指定された要素にえきペディア MAP を描画します。
「api 関数からのレスポンス.data」は、コールバック関数に渡されたレスポンスのデータ部分です。レスポンスが response の場合、response.data を渡します。

「描画要素」は 4.1 で確保した描画領域の ID を指定します。ID 名の前には"#"をつけてください。

パラメータ

パラメータはオプションで、指定できるパラメータには以下のものがあります。

パラメータ	値	意　味
scale	"auto"（規定値）	えきペディア MAP を、描画領域のサイズに自動的に合わせて描画する。
	"width"	えきペディア MAP を、描画領域の幅に合わせて描画する。
	"height"	えきペディア MAP を、描画領域の高さに合わせて描画する。
	数値（倍率）	えきペディア MAP を、指定された倍率で描画する。
offsetX	数値（ピクセル数）	指定されたピクセル数だけ MAP を左にずらす。（※）
offsetY	数値（ピクセル数）	指定されたピクセル数だけ MAP を上にずらす。（※）
mapCtrl	true（規定値）	MAP をマウスで移動、拡大縮小する MAP コントローラーを表示する。
	false	MAP コントローラーを表示しない。

※ここで指定するピクセル数は、もとの MAP に対するピクセル数ではなく、指定された scale（倍率）で拡大縮小された MAP に対するピクセル数です。したがってまずオフセットなしで描画し、その画面からオフセット値を割り出してください。

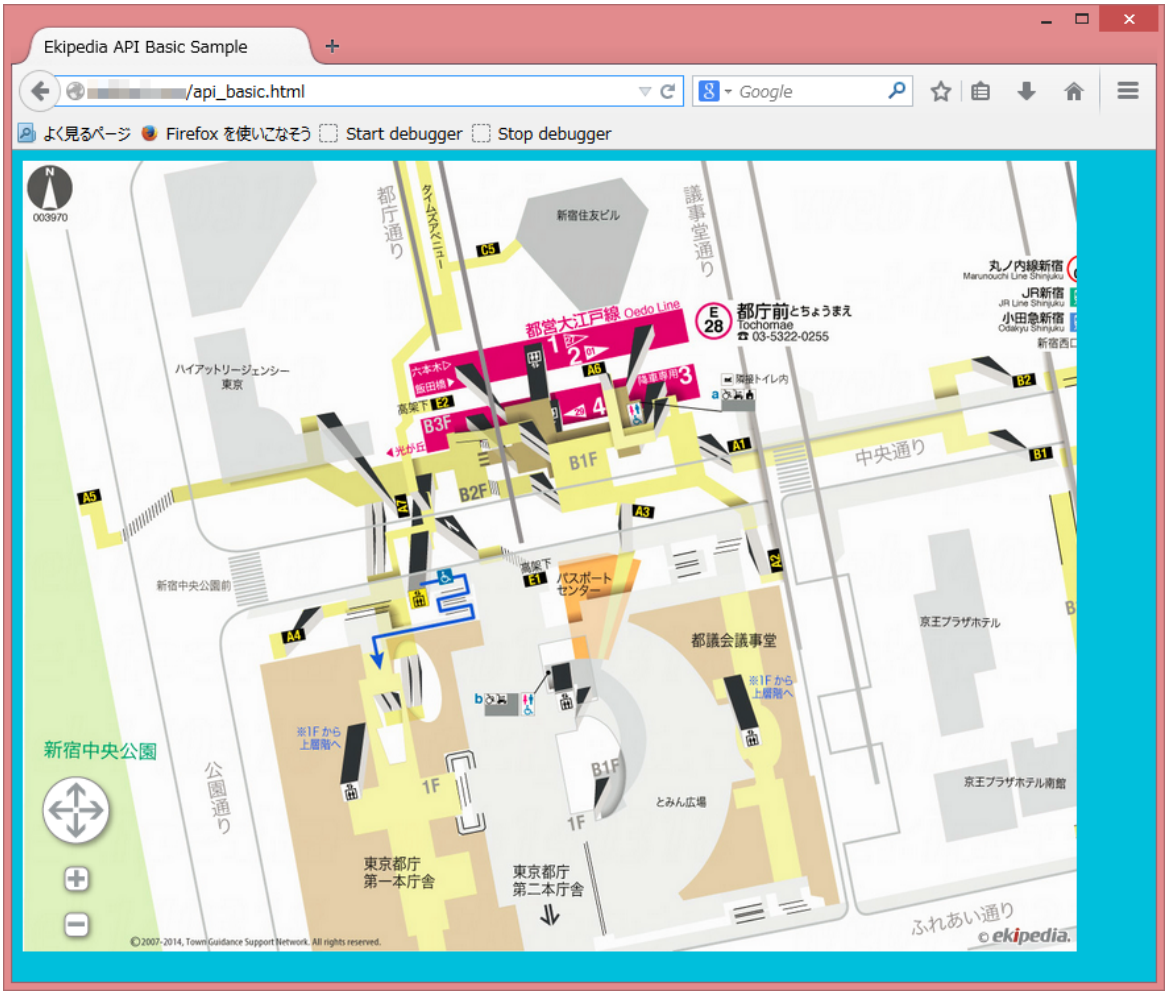
えきペディア MAP の描画サンプル

以下に幅 800 ピクセル、高さ 600 ピクセルの領域にえきペディア MAP を描画するサンプルを示します。

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

```
<title>Ekipedia API Basic Sample</title>
<script src="http://ekipedia.jp/sdk/sdk_1.0.js"></script>
<script>
    Ekipedia.init(function() {
        Ekipedia.api("station", {id: 3970},
            function(response) {
                Ekipedia.drawMap(response.data, '#station-map');
            }
        );
    });
</script>
<style type="text/css">
body {
    background-color: #00bfdc;
}
#station-map {
    width: 800px;
    height: 600px;
}
</style>
</head>

<body>
<div id="station-map"></div>
</body>
</html>
```



5. サムネイル画像の取得方法

最初 Web ページにえきペディア MAP の小さなサムネイル画像を配置しておき、それをクリックすると元の大きい MAP が表示されるという方法は、既存のページデザインを損なうことなく、かつ詳細な MAP を表示できる方法として検討に値します。

5.1 サムネイル画像の URL

えきペディア MAP のサムネイル画像は API を利用せず、下記 URL から直接得ることができます。

```
http://ekipedia.jp/ekipedia_map/thumb/駅 ID
```

駅 ID は、3.2　関数の説明の「リソースとパラメータ」で説明したものと同じで、各路線の駅に一意につけられた ID です。

えきペディアMAPのサムネイル画像は、幅300ピクセルで作成されています（高さはMAPによって異なります）。さらに縮小したい場合は、スタイルシート等でスタイルを与えてください。

以下に、サムネイル画像を幅200ピクセルで表示するサンプルを示します。

```
<style type="text/css">
.thumb-img {
    width: 200px;
    margin: 10px 0 0 10px;
    border: solid 1px #00bfdc;
}
.thumb-img img {
    max-width: 200px;
}
.caption {
    margin: 0 0 10px 10px;
    font-size: small;
}
</style>
…省略…
<body>
    <div class="thumb-img">
        <a href="popup.html?id=100" target="_blank">
            <img src="http://ekipedia.jp/ekipedia_map/thumb/100" alt="なんば"/>
        </a>
    </div>
    <p class="caption">なんば</p>
…省略…
```

なんば

6. えきペディア API を直接呼び出す方法

PHP などのサーバーサイド・プログラムからえきペディア API を利用するには、えきペディア API を直接呼び出す方法が適しています。

6.1 API の URL

えきペディア API は、下記 URL から呼び出すことができます。

```
http://ekipedia.jp/api/リソース/パラメータ/フォーマット
```

6.2 リソースとパラメータ

リソースは取得したいデータの種類を指定します。リソースとパラメータは、JavaScript SDK ライブラリの「3.2 関数の説明」で述べたものとほぼ同じですが、パラメータの形式が異なります。

リソース	意　味	指定できるパラメータ
cities	えきペディアが提供する地下鉄都市一覧。	指定できるパラメータはありません。
lines	指定された都市コードの路線一覧。	city/都市コード
stations	指定された路線の駅一覧。	line/路線 ID
station	指定された駅の情報。	id/駅 ID
exiits	指定された条件の出入口一覧。	station/駅 ID または type/出入口仕様 出入口仕様を指定した場合 line/路線 ID または city/都市コード がオプションとして指定できます。
toilets	指定された条件のバリアフリートイレ一覧	station/駅 ID または equip/設備コード 設備コードを指定した場合 line/路線 ID

		または city/都市コード がオプションとして指定できます。

6.3 フォーマット

指定したリソースをどの形式で返すかを指定します。フォーマットの指定はオプションで、規定値は JSON です。以下のフォーマットが指定できます。

フォーマット	意　味
format/json	指定されたリソースを JSON 形式で返す。(規定値)
format/xml	指定されたリソースを XML 形式で返す。

【例】
札幌市営地下鉄さっぽろ駅の情報を XML 形式で取得する。

```
http://ekipedia.jp/api/station/id/6640/format/xml
```

【例】
東京メトロ銀座線の全駅の一覧を XML 形式で取得する。

```
http://ekipedia.jp/api/stations/line/50/format/xml
```

【例】
大阪でオストメイトと介助ベッドとの設備があるトイレを調べ、JSON 形式で取得する。

```
http://ekipedia.jp/api/toilets/equip/1,2/city/7
```

6.4 API のレスポンス

「3.2 関数の説明」の「レスポンス」で説明した response.data と同様のデータが、指定した形式で返されます。データの詳細については「3.2 関数の説明」の「レスポンス」を参照してください。

以下に地下鉄都市一覧を XML 形式で取得した場合のレスポンスを示します。

```
http://ekipedia.jp/api/cities/format/xml
```

```
-<xml>
    -<item>
        <code>1</code>
        <name>札幌</name>
        <nameEn>Sapporo</nameEn>
    </item>
    -<item>
        <code>2</code>
        <name>仙台</name>
        <nameEn>Sendai</nameEn>
    </item>
    -<item>
        <code>3</code>
        <name>東京</name>
        <nameEn>Tokyo</nameEn>
    </item>
        …省略…
</xml>
```

6.5 API の呼び出し例

API の呼出には、えきペディアが発行したユーザーID とパスワードが必要です。
以下、PHP から cURL 関数を使用して API を呼び出す例を示します。

```
<?php
$userid = 'user1234';
$password = '1234';

$curl_handle = curl_init();
curl_setopt($curl_handle, CURLOPT_URL, 'http://ekipedia.dev/api/cities/');
curl_setopt($curl_handle, CURLOPT_RETURNTRANSFER, 1);
// ユーザーID、パスワードを設定する。
curl_setopt($curl_handle, CURLOPT_USERPWD, $userid . ':' . $password);

$buffer = curl_exec($curl_handle);
curl_close($curl_handle);

$result = json_decode($buffer);
// $result を使う処理
// …
?>
```