日日是Oracle APEX

Oracle APEXを使った作業をしていて、気の付いたところを忘れないようにメモをとります。

2020年3月27日金曜日

Oracle Spatialの機能をOracle APEXで使用する

海外では、Autonomous Database for APEX Developersというプログラム名で、Oracle Databaseが持つ機能をOracle APEXによるアプリケーションから活用する、というワークショップが展開されています。資料や手順は一般に公開されていますので、自分で試してみました。

Autonomous Database for APEX Developersの資料は、GitHubで公開されています。
https://github.com/oracle/cloudtestdrive/blob/master/ATP/APEX/readme.md
ここに含まれるLab 200, Add Spatial to your APEX appについて、どのような作業になったか記載していきます。

ライセンスやマニュアルではOracle Spatial and Graphとして、ひとつのまとまりになっています。この演習ではその中の、Oracle Spatialとして提供される地理空間を扱う機能を使用します。

準備

すでにOracle Cloudのアカウントは取得済みであり、Always FreeのAutonomous Databaseのインスタンスとと Oracle APEXのワークスペースも作成済みであれば、準備は完了しています。そのまま、次のステップに進むことができます。

このワークショップはAutonomous Database上で実行することが想定されています。そのための、Oracle Cloudのアカウント取得の手順についてPrerequisite for the workshop として説明されています。まだOracle Cloudのアカウントを持っていない場合は、こちらのリンクからサインアップを行います。それぞれの手順については、次のマニュアル、Oracle Cloudスタート・ガイドの無料Oracle Cloudプロモーションへのサインアップに記載されています。Oracle Cloudのアカウント取得が最初のステップです。

次に、ワークショップで使用するデータベースを用意します。Lab 100, First steps with ATPとして、Autonomous Transaction Processingのデータベースの作成について手順が示されています。このワークショップ全体として、無料トライアルの範疇で、かつ、Oracle APEXのワークスペース名を"WORKSHOPATP"と固定して行うことが想定されています。

とはいえ、気軽に作業を行ってみたいので、今回はAlways Freeのインスタンスを使うことにしました。準備の手順は、以前に書いたこちらのブログ記事「APEXからOCIのオブジェクト・ストレージを操作する - その３」の、Autonomous Databaseの作成で記載されている内容を実施します。ワークスペース名は自由に決めて構いません。

Oracle Spatialの機能をOracle APEXのアプリケーションで使用する

この演習では、あらかじめ提供されているOracle APEXのアプリケーションをインポートし、そのアプリケーションにOracle Spatial and Graphが提供する機能を使った処理を追加します。

作成するアプリケーションは、ドローンの飛行許可を与えて良いかどうかを判定する、というものです。ドローンを飛行させる位置をリクエストとして受け取って、データベースにあらかじめ保存している市街地の情報を参照し、リクエストされた位置が市街地に含まれていれば、不許可、含まれていなければ許可する、といった判定を行います。

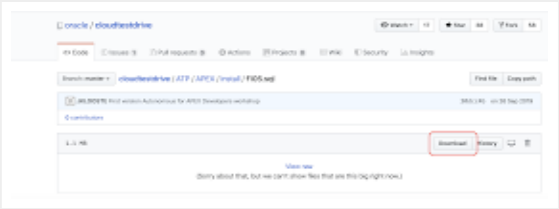
すでに作成済みのアプリケーションに、以下のふたつの機能を実装します。

- 住所から緯度経度を取得する - ジオコーディングを実装する
- Oracle Spatialが提供する空間演算子を使用して、上記の座標が市街地(ポリゴン)に含まれるかどうかを判定する

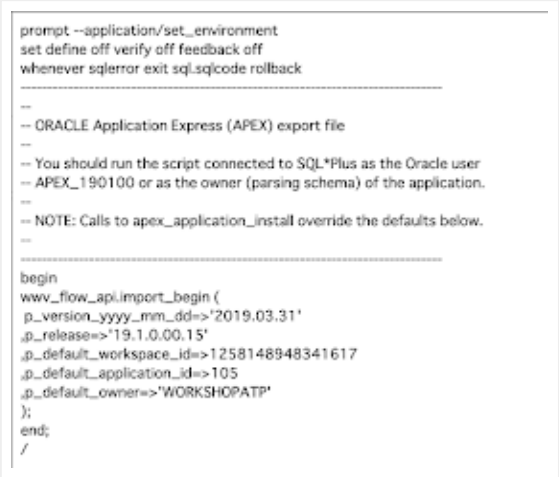
Oracle Spatial and Graphの開発者ガイドはこちらのリンクから参照できます。

APEXアプリケーションをインポートする

機能追加を行う元となるAPEXアプリケーションをインポートします。GitHubのこちらから、f105.sqlをダウンロードします。
GitHubの画面上で**Download**をクリックします。

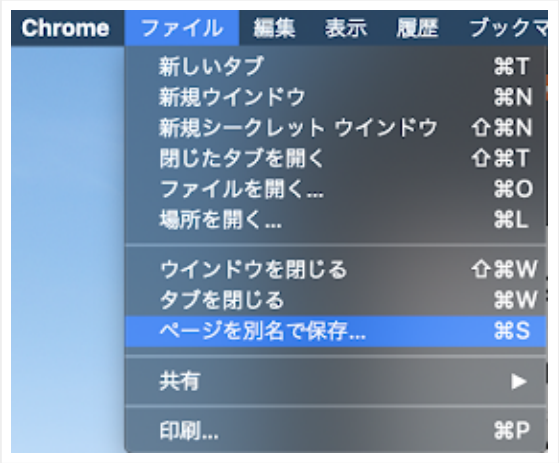


ブラウザの設定に依存するとは思うのですが、画面にPL/SQLのコードが表示されます。



```
prompt --application/set_environment
set define off verify off feedback off
whenever sqlerror exit sql.sqlcode rollback
--------------------------------------------------------------------------------
--
-- ORACLE Application Express (APEX) export file
--
-- You should run the script connected to SQL*Plus as the Oracle user
-- APEX_190100 or as the owner (parsing schema) of the application.
--
-- NOTE: Calls to apex_application_install override the defaults below.
--
--------------------------------------------------------------------------------
begin
wwv_flow_api.import_begin (
 p_version_yyyy_mm_dd=>'2019.03.31'
,p_release=>'19.1.0.00.15'
,p_default_workspace_id=>1258148948341617
,p_default_application_id=>105
,p_default_owner=>'WORKSHOPATP'
);
end;
/
```

これがインポートされるアプリケーションのコードですので、ファイルに保存します。以下はMacOS上のChromeを例にしていますが、大抵のブラウザのファイル・メニューに、**ページを別名で保存...**もしくは**別名でページを保存...**というコマンドとして、表示されているページをファイルに保存する機能はあるかと思います。そして、**f105.sql.txt**もしくは**f105.sql**というファイル名で、この内容をファイルに保存します。



ファイルに保存した後、ワークスペース名がWORKSHOPATP以外でもエラーなくアプリケーションのインポートが終了できるように、内容を変更します。ファイルに含まれる"Insert into WORKSHOPATP."という記載を"Insert into "へ置き換え、データの挿入の対象となっている表の指定からスキーマ名WORKSHOPATPを除きます。sedを使って実施した例は以下になります。

```
sed -e 's/Insert into WORKSHOPATP./Insert into /' f105.sql.txt > f105.sql
```

Oracle APEXのワークスペースにログインし、アプリケーション・ビルダーを開いて、インポートを実行します。

インポートの**ファイルのインポート**として、編集済みのファイル**f105.sql**を指定します。**ファイル・タイプ**はデフォルトで指定される、**データベース・アプリケーション、ページまたはコンポーネントのエクスポート**のままにし、**次**へ進みます。

これ以降のインポート・ウィザードからの質問はすべてデフォルトを選択して進みます。

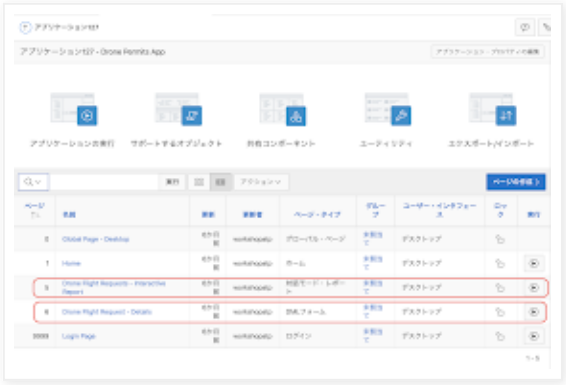
以下の画面が表示されるまで、かなりの時間がかかります。

インポートが正常に完了すると、緑のチェックの表示を含む画面になります。

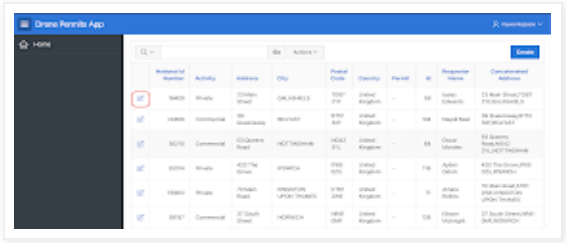
作成されたアプリケーションを確認する

作成されたアプリケーションは、ページ５の対話レポートと、そこから呼び出されるページ６のフ

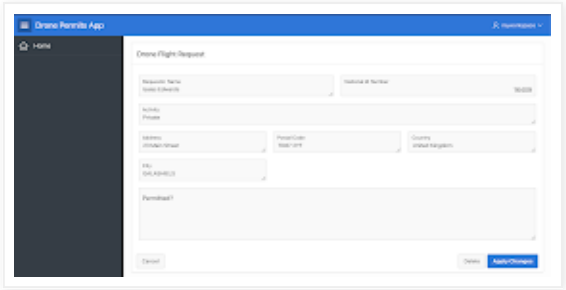
ォームを含みます。ホーム・ページには対話レポートを開くナビゲーションがないため、アプリケーション・ビルダーから直接ページ５の対話レポートを実行します。



対話レポートには、ドローンの飛行許可リクエストの一覧が表示され、そこから、それぞれのリクエストの編集画面を開くことができます。



編集画面に許可、不許可を判定する機能を追加していきます。



ジオコーディングを追加する

最終目標は、ドローンの飛行をリクエストする住所が市街地に含まれるかどうかを確認することですが、まず、リクエストされた住所から緯度(latitude)と緯度(longitude)を求める必要があります。この住所から座標を得る処理をジオコーディング(Geo-coding)と呼びます。

このジオコーディングを行う公開されたWebサービスはいくつかありますが、この演習ではオラクルが提供しているeLocationサービスを使用しています。ライセンスや利用許諾について、こちらに記載があります。詳しくはリンク先を確認していただきたいと思いますが、最初に"Oracle Maps Cloud is a cloud-based map data service for use with licensed Oracle applications and services."という記載があることより、**オラクルが提供しているアプリケーションおよびサービスからの利用に限定**されています。今回はAutonomous Databaseから呼び出すため、特に問題なく利用できます。住所と座標のデータはHERE Technologiesから提供されています。日経の記事によると日本の対応も行っており、サービスの説明にも"Geocoding provides comprehensive coverage in 196 countries"とあり日本が入っていないはずはないのですが、オラクルが提供しているeLocationサービスは残念なことに、日本に対応していないようです。ですので、実際にアプリケーションを作成する際には、別のジオコーダーを使う必要があります。

GoogleのジオコーディングAPIをOracle APEXにて使う方法について、いくつかの記事をネット上で見つけることができます。こちらの料金表からは一ヶ月$200(最大70,000リクエスト)までは無料で

利用できるとのことです。無料枠があるとはいえ、演習で有料サービスは扱いにくいため、Googleのジオコーディング**API**は使用しません。また別途、紹介できればと思います。

作成済みのアプリケーションは、Oracle Elocation Geocoderという動的アクションとして利用できるプラグインが組み込まれています。

このプラグインを使って、eLocationサービスのジオコーディング**API**の呼び出しを行います。このプラグインへ与える入力は以下になります。

- カンマ区切りで繋げた、通り(street)、郵便番号(postal code)、市(city)の情報
- 国名

戻り値は、**APEX**コレクションとして返されます。この中で重要なのは以下のふたつのカラムです。

- N002: 与えられた住所の経度
- N003: 与えられた住所の緯度

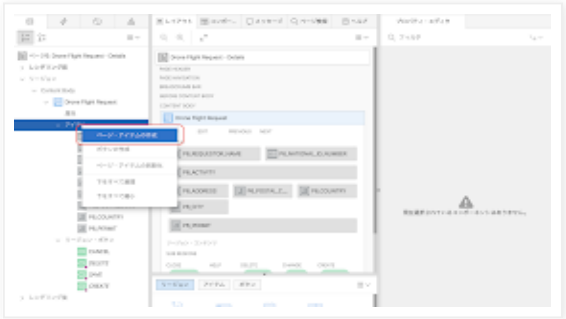
APEXコレクションについては、マニュアルのこちらが参考になります。端的にいえば、通常のデータベースの表となんら変わるところはないのですが、データ保存(こちらは**INSERT**文ではなく、**API**呼び出しによる)および取り出し時(こちらは**SELECT**文の実行)に**APEX**のセッションに紐付けられるという特徴があり、名前を付けたコレクションへは、セッションが継続している限りページ・リクエストをまたがってアクセスすることが可能になります。保存したデータを取り出すには**APEX_COLLECTIONS**ビューを対象とした**SELECT**文を実行します。このビュー定義についてはこちらに記載があります。実際に**eLocation**サービスを呼び出した結果を取り出す際には、以下の**SQL**を発行します。セッション**ID**は指定しませんが、現在のセッションで保存されたデータのみが検索対象になります。

```
SELECT N002, N003
from APEX_COLLECTIONS where collection_name = 'GEOCODER_RESULTS'
and n002 is not null and n003 is not null
order by seq_id;
```

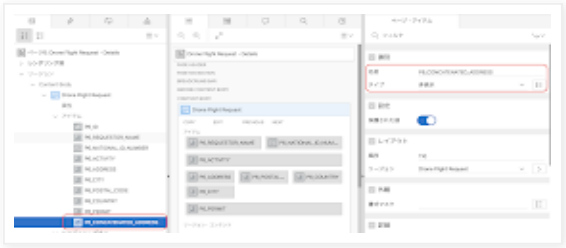
ジオコーディング**API**の結果は、与えられた住所から複数の地点が対象になることにより、複数の緯度経度のペアが返る場合があります。今回の演習では、実装を単純にするため、最初に返された値を判定に使用し、それ以外は無視しています。

ページ・アイテムP6_CONCATENETED_ADDRESSを追加する

ページ６のフォーム・リージョンDrone Fright Requestに、ページ・アイテム**P6_CONCATENATED_ADDRESS**を追加します。リージョンDrone Fright Requestを開き、アイテム上でコンテキスト・メニューを開いて、**ページ・アイテムの作成**を実行します。

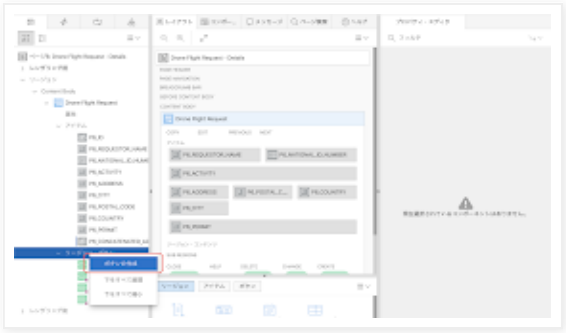


名前として**P6_CONCATENATED_ADDRESS**、**タイプ**を**非表示**とします。

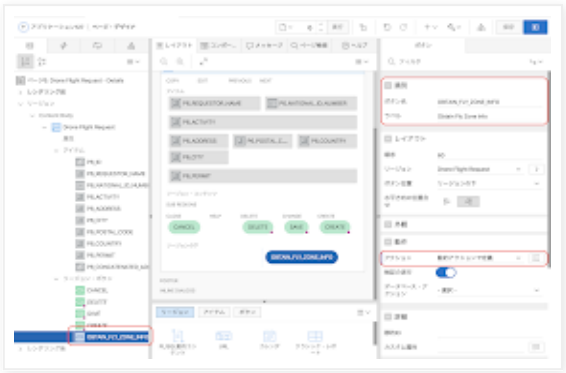


ボタンOBTAIN_FLY_ZONE_INFOを追加する

リージョン・ボタン上でコンテキスト・メニューを開いて、**ボタンの作成**を実行します。

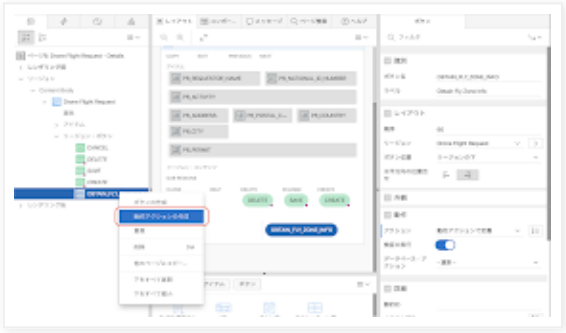


ボタン名として**OBTAIN_FLY_ZONE_INFO**、ラベルは**Obtain Fly Zone Info**を指定します。そして**動作**の**アクション**として、**動的アクションで定義**を選択します。



動的アクションを追加する

はじめにジオコーディングAPIに渡す住所を、P6_CONCATENATED_ADDRESSに設定するアクションを定義します。ボタンOBTAIN_FLY_ZON_INFO上でコンテキスト・メニューを開き、**動的アクションの作成**を実行します。

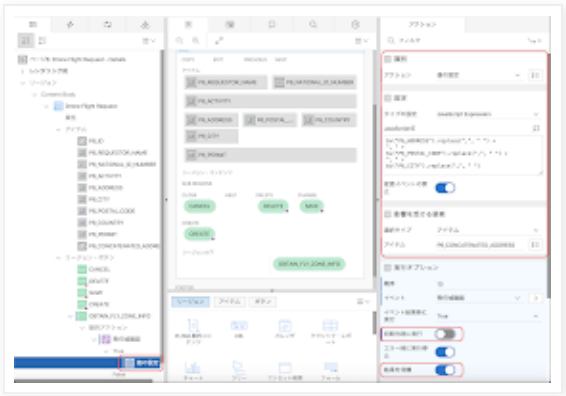


作成された動的アクションは名前が**新規**になっています。分かりやすくなるように**飛行域確認**に変更します。タイミングとして設定されている**イベント、選択タイプ、ボタン**はそれぞれ、**クリック、ボタン、OBTAIN_FLY_ZONE_INFO**となっているはずですが、確認しておきます。

Trueアクションの設定を行います。**表示**となっているTrueアクションを選択し、**アクション**を**値の設定**、**タイプの設定**を**JavaScript Expression**、**JavaScript式**として以下を入力します。
P6_ADDRESS, P6_POSTAL_CODE, P6_CITYをカンマで区切って連結しています。

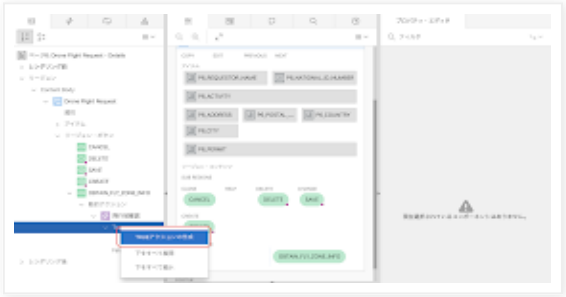
```
$v("P6_ADDRESS").replace(",", " ") +
", " +
$v("P6_POSTAL_CODE").replace(",", " ") +
", " +
$v("P6_CITY").replace(",", " ")
```

この**値の変更**という動作によって、別の動的アクションが起動されないよう、**変更イベントの禁止**を**ON**にします(今回は別のアクションは無いので、ONでもOFFでも動作に違いは発生しません)。そして、上記のJavaScript式の結果が設定される項目として、**影響を受ける要素**の**選択タイプ**を**アイテム**、**アイテム**を**P6_CONCATENATED_ADDRESS**とします。さらに**実行オプション**として、**初期化時に実行**を**OFF**（ページを開いたときに実行される必要がないため)、**結果を待機**を**ON**(後続となるアクションにて、設定された結果を参照できるようにするため）に指定します。

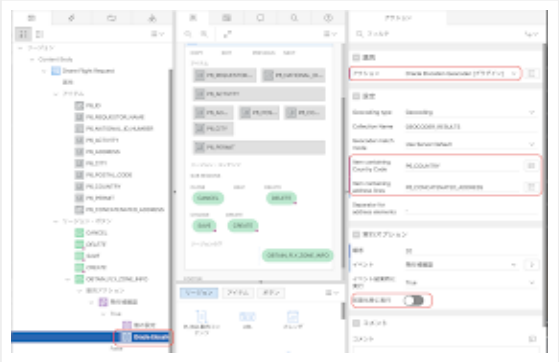


このアクションにより、P6_CONCATENATED_ADDRESSにジオコーディングAPIの引数となる住所が作成されます。

続いて、この住所を引数としてジオコーディングAPIを呼び出すアクションを作成します。動的アクションのTrue上でコンテキスト・メニューを開き、**TRUEアクションの作成**を実行します。



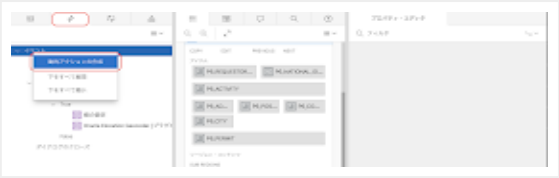
値の設定に続くTrue**アクション**として、**Oracle Elocation Geocoder [プラグイン]**を選択します。**設定**の**Item containing Country Code**は**P6_COUNTRY**、**Item containing address lines**は**P6_CONCATENATED_ADDRESS**を指定します。**実行オプション**の**初期化時に実行**は**OFF**とします。



設定のGeocoding type、Collection Nameはデフォルトで、GeocodingおよびGEOCODER_RESULTSとなっています。Geocoding typeは、これ以外にReverse Geocodingも指定可能です。Reverse Geocodingとは、座標から住所を求める操作です。今回は使用しません。

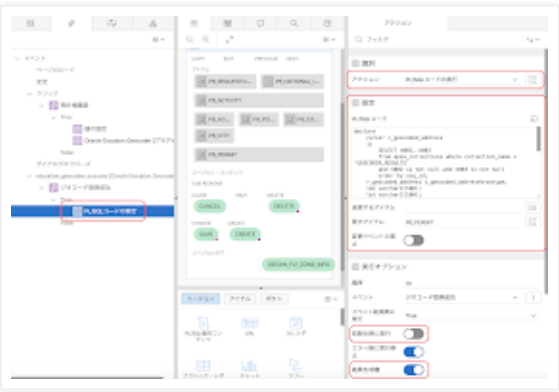
このアクションは値の取得を待たずに終了します(そのため、実行オプションに**結果を待機**がありません)。その代わりに、値が取得できるとイベントが通知されます。そのイベントによって起動される動的アクションを設定することにより、ジオコーディングAPIの戻り値を画面に表示します。

左ペインを**動的アクション・ビュー**の表示に切り替え、イベント上でコンテキスト・メニューを開き、**動的アクションの作成**を実行します。



作成された動的アクションの**名前**を**ジオコード取得成功**とし、**タイミング**として、**イベント**に**elocation_geocoder_success [Oracle Elocation Geocoder]**を選択、**選択タイプ**をJavaScript式、**JavaScript式**にdocumentを指定します。

続けて、**表示**となっているTrueアクションを選択し、**アクション**を**PL/SQLコードの実行**に変更します。そして、**PL/SQLコード**として以下に示すコードを入力します。**戻すアイテム**として**P6_PERMIT**、**変更イベントの禁止**は**OFF**(値の入力によってラベルの大きさが変わるようになっているため、イベントを禁止してはいけません)、**初期化時に実行**を**OFF**、**結果を待機**は**ON**とします。

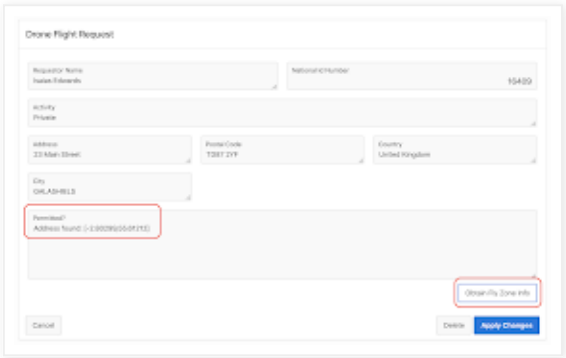


```
declare
   cursor c_geocoded_address
   is
      SELECT n002, n003
      from apex_collections where collection_name = 'GEOCODER_RESULTS'
      and n002 is not null and n003 is not null
      order by seq_id;
   r_geocoded_address c_geocoded_address%rowtype;
   lon varchar2(100);
   lat varchar2(100);
   coordinates MDSYS.SDO_GEOMETRY;
   overlap_count number;
   result varchar2(100) := '';
begin
   open c_geocoded_address;
   fetch c_geocoded_address into r_geocoded_address;
   lon := r_geocoded_address.n002;
   lat := r_geocoded_address.n003;
   close c_geocoded_address;
   if lon is not null and lat is not null then
      result := result || 'Address found: ' || '[' || lon || ',' || lat || ']';
   else
     result := result || 'Address not found';
   end if;
   :P6_PERMIT := result;
exception
   when others then
      :P6_PERMIT := sqlerrm;
end;
```

APEXコレクションGEOCODER_RESULTSより、先頭行のN002(経度)、N003(緯度)を取り出し、ページ・アイテムP6_PERMITへ設定しています。

ジオコーデイングAPIの呼び出しの実装はこれで完了です。動作を確認してみましょう。

ページ５の対話レポートを開き、適当なリクエストの編集画面を開きます。Obtain Fly Zone Infoボタンをクリックし、Permitted?の項目に座標値が表示されることを確認します。



ドローンの飛行リクエスト地点が市街地に入らないことを検証する

飛行リクエストに記載されている住所から、座標(緯度 - 経度)を求めることができました。この座標が市街地に入らないことを検証するには、市街地の定義情報が必要です。市街地のデータは、アプリケーションのインポートに含まれるサポート・スクリプトが実行されることにより、表URBAN_AREA_UKが作成され、その表にロードされています。

表URBAN_AREA_UKの定義は以下です。この中の"GEOM"列がOracle Spatialが提供するSDO_GEOMETRY型として定義されていて、それぞれの市街地の領域が登録されています。

```
CREATE TABLE  "URBAN_AREA_UK"
  ( "SCALERANK"      NUMBER(38,0),
 "FEATURECLA"    VARCHAR2(50 CHAR) COLLATE "USING_NLS_COMP",
 "AREA_SQKM"     NUMBER,
 "MIN_ZOOM"      NUMBER,
 "GEOM"          "MDSYS"."SDO_GEOMETRY" ,
 "STUDIO_ID"     NUMBER
  ) DEFAULT COLLATION "USING_NLS_COMP"
 VARRAY "GEOM"."SDO_ELEM_INFO" STORE AS SECUREFILE LOB
 VARRAY "GEOM"."SDO_ORDINATES" STORE AS SECUREFILE LOB
/
```

GEOM列を含む、市街地を定義する表URBAN_AREA_UKに保持される行は以下のようなINSERT文によって入力されています。

```
Insert into URBAN_AREA_UK (SCALERANK,FEATURECLA,AREA_SQKM,MIN_ZOOM,GEOM,STUDIO_ID)
values (9,'Urban area',11.659,7.7,MDSYS.SDO_GEOMETRY(2003, 4326, NULL,
MDSYS.SDO_ELEM_INFO_ARRAY(1, 1003, 1),
MDSYS.SDO_ORDINATE_ARRAY(-3.62936273541405, 57.656023725336, -3.62936823910738,
57.6560284078993, -3.62937442179765, 57.6560321484047, -3.62947207804757,
57.6560809765297, -3.62961856242254, 57.6561542187173, -3.62962329256217,
57.6561562541272, -3.62962776599323, 57.6561576233396, -3.62972374689068,
57.6561816185639, -3.62986859296981, 57.6562299005905, -3.62987190661818,
57.6562308655271, -3.62996956286821, 57.6562552795897, -3.6299730763682,
57.6562560103527, -3.63011778940148, 57.6562801291917, -3.63021370349318,
57.6563041077145, -3.63021680109313, 57.6563047671537, -3.63022460937489,
57.65630545029, -3.63037109374983, 57.65630545029, -3.63051757812485,
57.65630545029, -3.63052538640662, 57.6563047671537, -3.63052639670317,
57.6563045770786, -3.63064846701559, 57.6562801630162, -3.63076982633862,
57.6562558911515, -3.63091559550656, 57.6562315962903, -3.630922422655,
57.6562299005905, -3.63106726873372, 57.6561816185641, -3.63116324963161,
57.6561576233396, -3.63116772306265, 57.6561562541272, -3.63117245320223,
57.6561542187173, -3.63131893757708, 57.6560809765297, -3.63141659382717,
57.6560321484047, -3.6314189674261, 57.6560308712871, -3.63142346403629,
57.6560279023819, -3.63152112028632, 57.6559546601945, -3.63161877653624,
57.6558814180069, -3.63171643278636, 57.6558081758195, -3.63172124896073,
57.6558039987736, -3.63181890521076, 57.6557063425235, -3.63191656146077,
57.6556086862735, -3.63201421771072, 57.6555110300235, -3.63201890027403,
57.6555055263302, -3.63202264077944, 57.6554993436399, -3.63207010412043,
57.6554044169579, -3.63411645017348, 57.6524079816659, -3.63456912555174,
57.6519791313074, -3.64234706604447, 57.6496140522234, -3.64234936368763,
57.6496132853771, -3.6423549038706, 57.6496108346436, -3.65094865387056,
57.6450942330811, -3.65095021742607, 57.6450933712871, -3.65095672677717,
57.6450888006725, -3.66042947177598, 57.6370564992496, -3.6604348366576,
57.6370511065912, -3.66043672357409, 57.6370486957131, -3.66556367669915,
57.6300174457131, -3.66556710690691, 57.630011949168, -3.66556971157965,
57.6300060167137, -3.6655714366409, 57.6299997715159, -3.66620620226587,
57.6268259433909, -3.66620706140031, 57.6268182498759, -3.66620658820841,
57.6268105230155, -3.66620479671508, 57.6268029918247, -3.66620174001804,
```

57.6267958795193, -3.66493220876805, 57.624427715457, -3.66492811590572,
57.6244214104963, -3.66492302992876, 57.6244158754362, -3.66491709296636,
57.6244112649558, -3.66491047092887, 57.6244077078966, -3.66288875241895,
57.6235309779987, -3.66288406300589, 57.6235288890039, -3.66288159025021,
57.6235278001856, -3.66287501881783, 57.6235260015337, -3.66287402037145,
57.6235257657724, -3.66056031150012, 57.6230630180061, -3.6581321621951,
57.622018913805, -3.64139066211811, 57.6036423109506, -3.64138632552834,
57.6036381475299, -3.64138000656868, 57.6036337106362, -3.63730280730977,
57.6012655171435, -3.63729565275014, 57.6012621751857, -3.6372921730507,
57.6012610665026, -3.6267941261757, 57.5983802071276, -3.62679003484413,
57.5983792871591, -3.62678422604128, 57.5983786484997, -3.61855668697879,
57.5980124375622, -3.61855468749991, 57.5980123930854, -3.61854742598842,
57.5980129832842, -3.61586160572836, 57.5984524828779, -3.61585376131205,
57.5984545579981, -3.61584837818913, 57.5984569265807, -3.61406615162662,
57.5993846609557, -3.61406019446767, 57.5993883802168, -3.61405488906942,
57.5993929816726, -3.61405036484433, 57.5993983530814, -3.61300056015679,
57.6008631968314, -3.61299775325639, 57.6008676408101, -3.61299548411242,
57.6008723819725, -3.61299378373008, 57.6008773555363, -3.61250550248008,
57.6026351680363, -3.61250454515874, 57.6026393946559, -3.61250387062665,
57.6026463233343, -3.61184468663706, 57.6363381693861, -3.6118453654712,
57.6363464174693, -3.61184606344506, 57.6363496680663, -3.61226077292213,
57.6379841112998, -3.61961829299983, 57.636216539, -3.63247636599985,
57.6356468770001, -3.63249432321282, 57.6356547715024, -3.63255987063195,
57.6356516144041, -3.64042505760611, 57.6391413482644, -3.64561926999983,
57.641424872, -3.63768469999985, 57.646226304, -3.63255774599983, 57.650946356,
-3.62919778925394, 57.6559309078049, -3.62920209499975, 57.6559337456834,
-3.62920893455916, 57.6559367692701, -3.62921618063231, 57.6559386175324,
-3.62922363281245, 57.6559392393525, -3.6292294213055, 57.6559392393525,
-3.62926507916404, 57.655974897211, -3.62927044458806, 57.6559794797161,
-3.62927646081656, 57.6559831664657, -3.62928297971007, 57.6559858666798,
-3.62928984075165, 57.6559875138701, -3.62929687499988, 57.6559880674775,
-3.62932707755544, 57.6559880674775, -3.62936273541405, 57.656023725336))**,3199);

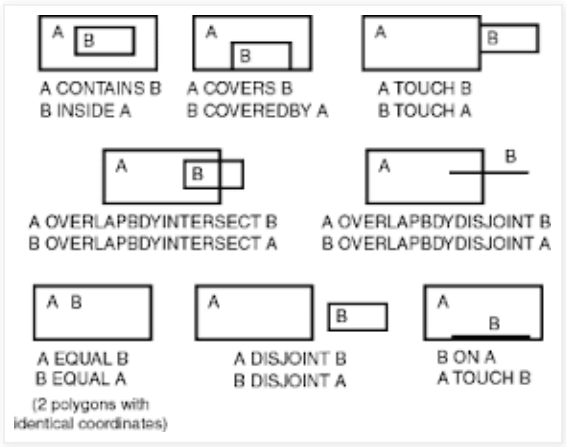
SDO_GEOMETRY型は以下で定義されています。

```
CREATE TYPE sdo_geometry AS OBJECT (
 SDO_GTYPE NUMBER,
 SDO_SRID NUMBER,
 SDO_POINT SDO_POINT_TYPE,
 SDO_ELEM_INFO SDO_ELEM_INFO_ARRAY,
 SDO_ORDINATES SDO_ORDINATE_ARRAY);
```

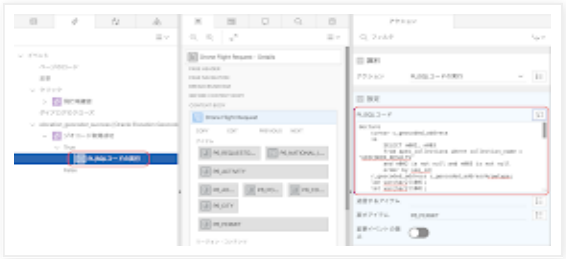
SDO_GTYPE属性は、ジオメトリのタイプを表しています。前出のINSERT文では2003が指定されています。これは２次元ポリゴンを示しています。SDO_SRID属性は座標系(空間参照システム)を表します。4326は地理座標系です。SDO_ELEM_INFOによって、後続のSDO_ORDINATESに格納される座標の解釈方法が決まります。SDO_ELEM_INFO_ARRAY(1, 1003, 1)は、先頭の値から始まる - オフセットの無い（最初の１）、反時計回りで、頂点が直線セグメントに連結している単純なポリゴンであり、かつ、始点と終点が同じ点であることを要求しています(続く1003,1)。SDO_ORDINATES属性は、空間オブジェクトの境界を形成する座標の値を格納するNUMBER型の可変長配列です。

このように定義された領域に、ジオコーディングの結果として取得した座標が入らないことを検証するために、Oracle Spatialが提供している空間演算子SDO_ANYINTERACTを使用します。これは位

相関係として以下に定義されているどれかに当てはまる場合にTRUEを返します。今回の場合はTRUEを返す行が存在する場合はドローンの飛行をリクエストしている住所が市街地に入るため、飛行を許可しない、ということになります。



以上の処理を実装するため、先ほどの動的アクション、ジオコード取得成功がTrueになったときに実行されるPL/SQLコードを以下に示すコードで置き換えます。



```
declare
    cursor c_geocoded_address
    is
        SELECT n002, n003
        from apex_collections where collection_name = 'GEOCODER_RESULTS'
        and n002 is not null and n003 is not null
        order by seq_id;
    r_geocoded_address c_geocoded_address%rowtype;
    lon varchar2(100);
    lat varchar2(100);
    result varchar2(100) := '';
    coordinates MDSYS.SDO_GEOMETRY; --Add
    overlap_count number; --Add
begin
    open c_geocoded_address;
    fetch c_geocoded_address into r_geocoded_address;
    lon := r_geocoded_address.n002;
    lat := r_geocoded_address.n003;
    close c_geocoded_address;
    if lon is not null and lat is not null then
        result := result || 'Address found: ' || '[' || lon || ',' || lat || ']';

        --Added code for urban area check
        coordinates := MDSYS.SDO_GEOMETRY(2001, 4326, MDSYS.SDO_POINT_TYPE(lon, lat, NULL), NULL, NULL);
        SELECT COUNT(*) INTO overlap_count
        FROM URBAN_AREA_UK "t2"
        WHERE SDO_ANYINTERACT("t2"."GEOM", coordinates) = 'TRUE';
        if overlap_count = 0 then
            result := result || ' - ALLOWED';
        else
            result := result || ' - NO FLY ZONE';
```

```
        end if;
        --End of added code for urban area check
    else
      result := result || 'Address not found';
    end if;
    :P6_PERMIT := result;
exception
    when others then
        :P6_PERMIT := sqlerrm;
end;
```

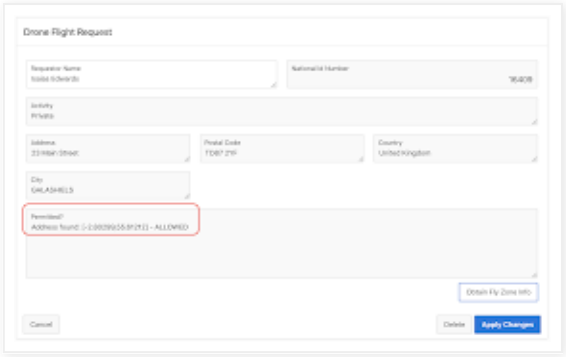
ここで判定に使われているSQLは以下になります。

```
SELECT COUNT(*) INTO overlap_count
 FROM URBAN_AREA_UK "t2"
 WHERE SDO_ANYINTERACT("t2"."GEOM", coordinates) = 'TRUE';
```

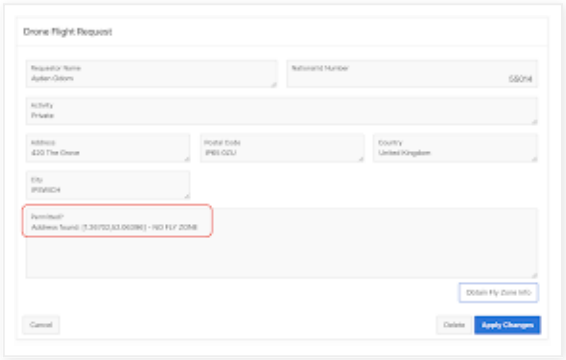
overlap_countが0であれば（つまりSDO_ANYINTERACTの結果がTRUEになる行が存在しなければ)、ALLOWEDになります。

最後にいくつかフォームを開いて、許可(ALLOWED)または不許可(NO FLY ZONE)となる結果を確認してみましょう。

Requestor NameがIsaias Edwardsのリクエストは許可されました。



Ayden Odomのリクエストは許可されませんでした。



Oracle Spatialの機能をOracle APEXから使用するアプリケーションを作成するワークショップの内容は以上でした。Oracle Spatialが提供する機能の、ほんのサワリを使っただけですが、どのようにAPEXアプリケーションからOracle Spatialの機能を利用するか、感じがつかめたでしょうか。 加えて、Oracle APEXの方針声明に、新しいマップ・コンポーネントの提供が表明されていることから、今後はより地理空間情報の扱いが容易になることが期待されます。

Oracle Spatial and Graphとして、本当に多彩な機能が提供されていますし、すでにOracle Databaseを使用している場合、追加ライセンスの購入なしで利用可能になっています(こちらのブログ、またはライセンスガイドを参照のこと)。

最後にOracle Spatial概要説明資料を紹介します。とても興味深い内容ですよ。

Oracle Spatial 概要説明資料 from **オラクルエンジニア通信**

完

Yuji N. 時刻: 17:15

ウェブ バージョンを表示

自己紹介

Yuji N.

日本オラクル株式会社に勤務していて、Oracle APEXのGroundbreaker Advocateを拝命しました。
こちらの記事につきましては、免責事項の参照をお願いいたします。

詳細プロフィールを表示