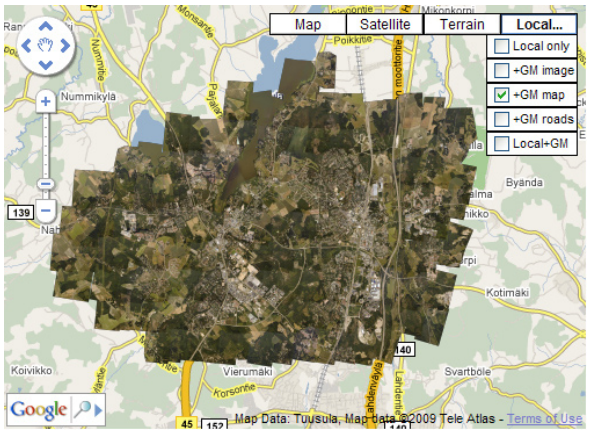
Web サイトを使ったカスタム Google マップの公開

TNTmips の自動モザイク処理で自分の画像や地図データから Google マップのタイルオーバレイを作成すれば、あなたのウェブサイトで簡単に公開できます。モザイク処理はカスタム地図セットを定義するサンプル HTML ファイル (マッシュアップ) を生成します。それは自分のタイルオーバレイを Google マップの画像や地図レイヤと組み合わせます。このファイルは地図選択のためのコントロールを設定しますので、カスタム地図や Google の標準地図の選択表示が可能になります。この HTML ファイルをタイルオーバレイと一緒に直接あなたのウェブサイトで公開すれば、誰でもあなたのカスタム地図をブラウザで見ることができます。HTML ファイルを編集して、表示するカスタム地図を変えたり、地図に含めるコントロールを変えたり、ガジェットを追加したり、ウェブページに独自のコンテンツを追加することもできます。ここでは構造理解の手助けのために、サンプル HTML ファイルに注釈をつけたものを掲載しました (抜粋)。コンテンツ編集のスタートラインにして下さい。

サンプル HTML ファイルには、Google マップの API を使ういくつかの JavaScript セクションがあり、そこで様々なカスタム地図の組み合わせで使われるレイヤを定義し、Google マップにロードします。あらかじめ定義されている Google マップ API エレメントの使用箇所全てと Google に必須の他のコードセクションが、赤で強調表示されています。Google はオンラインの Google マップ開発者向けガイドと API リファレンスを http://code.google.com/apis/maps で提供しています。

あなたのウェブサイトで Google マップ API を使うには Google マップ API キーも必要です。自動モザイク処理にある [Google マップキー (Google Maps Key)] ボタン (テクニカルガイドの「モザイク処理：Google マップタイルオーバレイへのモザイク (Mosaic:Mosaic to Google Maps Tile Overlay)」を参照) を使うと、Google のウェブページに飛び、そこでサインアップし、1 つのウェブドメイン内でのみ有効なキーを入手できます。モザイク処理で API キーを入力すれば、自動的にキーがサンプル HTML ファイルに組み込まれ、すぐ使えるようになります。



Google マップ上にローカルのフィンランド、Tuusula の正射画像タイルオーバレイを重ねた Google マップのカスタム地図 (マッシュアップ)。TNTmips のモザイク処理でタイルオーバレイとともに作られたサンプル HTML には、一連のカスタム地図の選択肢を持つ Local メニューボタンが作られ、ローカルタイルオーバレイと Google の地図や画像レイヤとの様々な組み合わせを表示できます。

Google マップ API のいくつかの主要なクラス

GTileLayer: カスタム地図で使う 1 つの多重 (マルチ) 解像度タイルレイヤを定義します。あなたのローカルのタイルオーバレイでも、標準の Google マップタイプのレイヤでもかまいません。

GMapType: 1 個または複数のタイルレイヤを持つカスタム地図タイプを定義します。標準の Google マップの地図タイプやタイルレイヤへのアクセスを提供します。

GMap2: ウェブページにおいて特定の HTML 内で " 地図 " の定義 (通常、標準とカスタム地図タイプで 1 セット) に使うコアクラス。

フィンランド、Tuusula のタイルオーバレイを使用したカスタム Google マップを定義している HTML ファイル

```
<html>
   <head>
      <title>Google Maps Tiles</title>
      <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
<script src="http://www.google.com/jsapi?key=ABQIAAAA" type="text/javascript">
   </script>

<script type="text/javascript">
var map;
var cmap=[];
var bounds;
var suppressedSearch=false;
function mapload(){

   if (GBrowserIsCompatible()){

      map = new GMap2(document.getElementById("mapDiv"),
            {googleBarOptions:{showOnLoad:true, onSearchCompleteCallback:function(){suppressedSearch=true;}}});
      bounds = new GLatLngBounds(new GLatLng(60.357634,24.916117), new GLatLng(60.455370,25.178711));

var myCopyright = new GCopyright(1, bounds, 10, "Tuusula");
var myCopyrightCollection = new GCopyrightCollection("Map Data:");
myCopyrightCollection.addCopyright(myCopyright);
```

Google の全地図と共に JavaScript をロードし、Google のウェブサイトから API を探し、Google API キーを指定します。HTML の body で呼ばれる google.load メソッドとセットで使用します。

カスタム地図をセットアップするためのローカル JavaScript

生成されるカスタム地図のリストを格納する変数の配列

地図セットをロードする関数

使用しているブラウザで Google マップ API が使用可能かをチェックします

HTML の "mapDiv"DIV エレメントにおいて新しい地図セットを生成し、Google バー検索コントロールを設定するオプションをセットします

ローカルのタイルオーバレイを緯経度座標の矩形の範囲で定義します

ローカルタイルオーバレイの地図の領域やズームレベルの範囲の表示に必要なコピーライトのメッセージをセットします

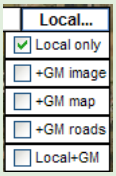
(次項へ続く)

"Local only" のカスタム地図に対するコード

ローカルタイルオーバレイではどのタイル位置、どのズームレベルでも、JPEG もしくは PNG フォーマットが使用できます。加えて、データがない位置には自動的に「データがありません」のメッセージの " ブランク "PNG タイルが表示されます。これらのタイルの各タイプは別々のタイルレイヤとしてロードされます (Google マップ API の GTileLayer エレメント)。これらの各レイヤを Google マップにロードするには、TileLayer クラスメソッドのコードを 3 個実装する必要があります。それはタイルそれぞれのインデックスポジションとズームレベルについて実行しなければなりません。getTileUrl は、タイルの URL またはローカルのディレクトリパスを返します。isPNG はそのタイルが PNG フォーマットかどうか、よって透明化できるかどうかを Google マップに伝えます。GetOpacity はタイルの画像フォーマットにかかわらず不透明度の値をセットします。

```
var myLayer1 = [new GTileLayer(myCopyrightCollection, 10, 20),
      new GTileLayer(myCopyrightCollection, 10, 20),
      new GTileLayer(myCopyrightCollection, 10, 20)];
```

"Local only(ローカルタイルオーバレイのみ)" のカスタム地図に対する 3 つのローカルタイルレイヤの配列を定義します。それぞれ、10 から 20 のズームレベルおよび前に定義したコピーライト情報が必要です。レイヤはこの順番で Google マップにロードされます。




```
myLayer1[0].getTileUrl = function (p,z) {
   var f;
   f = "Tuusula_Tiles/blank.png";
   return f;
   };
   myLayer1[0].isPng = function() { return true;};
   myLayer1[0].getOpacity = function() { return 1.0; }
```

注意 : JavaScript の変数配列のインデックスは 0 から始まります。

0 番目のレイヤ : タイル位置 *p*、ズームレベル *z* の各タイルに対して blank.png ブランクタイル 1 つをロードするコード。ブランクエリアに「データがありません」のメッセージが表示されます。

```
   myLayer1[1].getTileUrl = function (p,z) {
      var f;
      f = "Tuusula_Tiles" + "/" + z + "/" + p.y + "/" + p.x + ".png";
      return f;
      };
   myLayer1[1].isPng = function() { return true;};
   myLayer1[1].getOpacity = function() { return 1.0; }
```

1 番目のレイヤ : タイル位置 *p*、ズームレベル z の各タイルに対して PNG フォーマットのタイルを (存在すれば) 得るためのコード。各ズームレベルのローカルタイルはズームレベル z 用の名前のディレクトリに格納されています。各ズームレベルには、Google マップのタイル行番号 p.y の名前がついたサブディレクトリがあります。各行番号のディレクトリの中に、Google マップのタイル列の番号 p.x の名前がついたファイルがあります。ファイルの拡張子は ".png" または ".jpg" です。ここで実装している getTileUrl メソッドのコードは、タイルの位置とズームレベルから使用可能な PNG タイルファイルへのローカルパスを組立てて、タイルをロードできるように Google マップ API にパスを与えます。

```
   myLayer1[2].getTileUrl = function (p,z) {
      var f;
      f = "Tuusula_Tiles" + "/" + z + "/" + p.y + "/" + p.x + ".jpg";
      return f;
      };
   myLayer1[2].isPng = function() { return false;};
   myLayer1[2].getOpacity = function() { return 1.0; }
```

2 番目のレイヤ : タイル位置 *p*、ズームレベル *z* の各タイルに対して JPEG フォーマットのタイルを (存在すれば) 得るためのコード

```
cmap[0] = new GMapType(myLayer1, G_SATELLITE_MAP.getProjection(), "Local only");
```

Google の航空写真地図レイヤから得た投影法を使い、myLayer 1 を元にして "Local only" という名前のカスタム地図タイプを定義しそれを *cmap* 配列の 0 番目にストアします。

[カスタム地図の myLayer2(+GM image)、myLayer3(+GM map)、myLayer5(+GM roads) のコードは省略します]

"Local+GM" のカスタム地図用コード : Google マップの航空写真とローカルの Tuusula 画像、Google マップのラベルと道路

```
var myLayer5 = [new GTileLayer(myCopyrightCollection, 10, 20),
                G_HYBRID_MAP.getTileLayers()[0],
          new GTileLayer(myCopyrightCollection, 10, 20),
          new GTileLayer(myCopyrightCollection, 10, 20),
                G_HYBRID_MAP.getTileLayers()[1]];
```

ローカルと Google マップの " ハイブリッド " マップレイヤを組み合わせるカスタム地図用の 5 つのタイルレイヤ (下記に説明) の配列を定義します。

0 番目のレイヤ (一番下に表示):「データがありません」のメッセージのローカルのブランク PNG タイル

1 番目のレイヤ : Google" ハイブリッド " マップの中の航空写真レイヤ

2 番目のレイヤ : ローカルの PNG タイル (あれば)

3 番目のレイヤ : ローカルの JPG タイル (あれば)

4 番目のレイヤ (一番上に表示): Google" ハイブリッド " マップの中のラベルと道路のレイヤ

```
myLayer5[0].getTileUrl = function (p,z) {
   var f;
   f = "Tuusula_Tiles/blank.png";
   return f;
   };
myLayer5[0].isPng = function() { return true;};
myLayer5[0].getOpacity = function() { return 1.0; }
```

0 番目のレイヤとしてローカルブランクタイルをロードするコード

```
myLayer5[2].getTileUrl = function (p,z) {
   var f;
   f = "Tuusula_Tiles" + "/" + z + "/" + p.y + "/" + p.x + ".png";
   return f;
};
myLayer5[2].isPng = function() { return true;};
myLayer5[2].getOpacity = function() { return 1.0; }
```

2 番目のレイヤとしてローカルの PNG 画像タイルをロードするコード


```
myLayer5[3].getTileUrl = function (p,z) {
   var f;
   f = "Tuusula_Tiles" + "/" + z + "/" + p.y + "/" + p.x + ".jpg";
   return f;
   };
myLayer5[3].isPng = function() { return false;};
myLayer5[3].getOpacity = function() { return 1.0; }
```

3 番目のレイヤとしてローカルの JPG 画像タイルをロードするコード

```
cmap[4] = new GMapType(myLayer5, G_HYBRID_MAP.getProjection(), "Local+GM");
```

Google のハイブリッドマップレイヤから得た投影法を使い、myLayer 5 を元にして "Local+GM" という名前のカスタム地図タイプを定義し、それを *cmap* 配列の 4 番目にストアします。

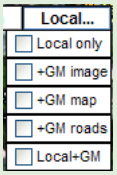
(次項へ続く)

```
    var myLayer6 = [new GTileLayer(myCopyrightCollection, 10, 20),G_HYBRID_MAP.getTileLayers()[0], G_HYBRID_MAP.getTileLayers()[1]];

    myLayer6[0].getTileUrl = function (p,z) {
       var f;
       f = "Tuusula_Tiles/blank.png";
       return f;
       };
    myLayer6[0].isPng = function() { return true;};
    myLayer6[0].getOpacity = function() { return 1.0; }

    cmap[5] = new GMapType(myLayer6, G_HYBRID_MAP.getProjection(), "Local...");
```

もし Local メニューのチェックボックスオプションが全部オフになっていた場合、" ローカル " 地図の選択メニューボタンに使うデフォルトレイヤを定義します。Google マップのハイブリッドマップを表示します。また他の (このレイヤと子の関係の)Local メニューオプションに対して親の役割を提供します。




```
    map.enableGoogleBar();
```

地図セットで Google バーの検索コントロールを使用可能にします

```
    var ui = new GMapUIOptions(new GSize(400,400));
    ui.controls.menumaptypecontrol = false;
    ui.controls.maptypecontrol = false;
    ui.controls.scalecontrol = false;
    map.setUI(ui);
```

その地図セット用の一連のユーザインタフェースコントロールオプションを作成し設定します。デフォルトマップタイプおよびメニューマップタイプの制御をオフにして (従って、別のマップ選択コントロールが可能となります)、デフォルトのスケールバーの制御をオフにします。

```
    for (var i =0; i < cmap.length; i++ ){
       cmap[i].getMinimumResolution = function() {return (10);}
       cmap[i].getMaximumResolution = function() {return (18);}
       cmap[i].isCustom = true;
       map.addMapType(cmap[i]);
       }

    var mapTypesControl = new GHierarchicalMapTypeControl();
```

作成したカスタム地図の配列を使った処理を繰り返します。それぞれ、最小解像度 (ズームレベル) ・最大解像度を定義してカスタム地図タイプとして宣言し、マップセットへ加えます。

" 階層的 " 地図選択ボタンとチェックボックスの入れ子式メニューを構築します。Google マップタイプの初期状態ではこの制御用のボタンが自動的に追加されます。

```
  for (var i = 0; i < cmap.length - 1; i++){
       mapTypesControl.addRelationship(cmap[cmap.length-1], cmap[i]);
       };
    map.setCenter(new GLatLng(60.406502,25.047414), 12, cmap[2]);
    map.addControl(mapTypesControl);
    map.enableContinuousZoom();
    map.enableScrollWheelZoom();
```

カスタム地図の配列を使った処理を繰り返します。最後のカスタムマップタイプを親として使い、残りのカスタム地図を子として制御に加えます (メニューチェックボックスアイテム)。

地図の中心を設定します。先に定義したユーザインタフェースコントロールオプションを加え、連続したスムーズなズーミングとマウスのスクロールホイールを使ったズーミングをできるようにします。

```
    GEvent.addListener(map, "move", function() {forceBounds();});
    }
```

地図の移動の Google マップイベントハンドラを登録し、常に地図の中心を表示するためのカスタム関数を呼びます。

```
  function forceBounds() {
    if (map.getCurrentMapType().isCustom != true) {
    return;
    }
    if (bounds.contains(map.getCenter())) {
    return;
    }
    var C = map.getCenter();
    var X = C.lng();
    var Y = C.lat();
    var maxX = bounds.getNorthEast().lng();
    var maxY = bounds.getNorthEast().lat();
    var minX = bounds.getSouthWest().lng();
    var minY = bounds.getSouthWest().lat();
    if (X < minX) {X = minX;}
    if (X > maxX) {X = maxX;}
    if (Y < minY) {Y = minY;}
    if (Y > maxY) {Y = maxY;}
    map.setCenter(new GLatLng(Y,X));
    if (suppressedSearch){
  alert("Your search results appear to be beyond the extents of this map data");
       suppressedSearch = false;}
       }
    }
</script>
</head>
```

地図が手動や Google バー検索によって移動されるときに呼ばれる関数。ブラウザにカスタム地図の中心を強制的に表示し続けます。

新しい地図表示の中心がまだカスタム地図の範囲内であれば何もしません。

新しい地図表示の中心の緯度と経度、そしてカスタム地図の境界の隅の緯度や経度を取得します。新しい地図の中心がこの境界の外側にあれば、中心の経度や緯度をカスタム地図の一番近い境界にリセットします。

Google バーの検索で再センタリングが抑止されていれば警告メッセージを表示します。

forceBounds 関数終わり

mapload 関数終わり

ページをロードしたときに呼ばれる mapload 関数および、そのページをアンロードしたときに Google マップのイベントハンドラと構造体を開放する Google の関数を登録します。

```
<body onload="mapload()" onunload="GUnload()">

    <div id="mapDiv" style="width:100%;height:100%" ></div>
```

地図を含む DIV エレメント。ここではウェブページいっぱいに地図を表示するように設定されていますが、そのページに独自の内容を付け加えたいのであれば地図のサイズを小さくできます。

```
    <script type="text/javascript">
       google.load("maps", "2");
    </script>
</body>
</html>
```

先に登録した API セットから Google マップ API のバージョン 2 をロードします。