



---

# CMP実習2

Ajax, XML, JSON, Web API

---

中村, 宮下, 齊藤, 菊池



# まずは

- Chrome を使いましょう Firefox でも良いですが、Internet Explorer と Safari は色々トラブルが発生することが多いので、避けましょう！
- Chrome を使う場合は、F12を押して、エラーが出ていないか確認しましょう！



エラーの場所  
クリックしよう！

# Web 2.0 (Tim O'reilly) (死語)

1. **Folksonomy:** Flickr, delicious, ...

2005～2007年くらい

2. **Rich User Experiences:** Google Map, Gmail, ...

3. **User as contributor:** Reviews in Amazon.com, ...

4. **The Long Tail:** Amazon.com, Google Adsense, ...

5. **Participation:** SNS (myspace, mixi, ...)

6. **Radical Trust:** Wikipedia, OSS, CC, ...

7. **Radical Decentralization:** Mashup, P2P, ...

# Rich User Experiences

- Web上での豊かな体験 (Ajaxなどによる)



The image shows two overlapping web browser windows. The background window is Google Maps, displaying a map of Kyoto University with several location markers (A, B, C, D) and their details. The foreground window is Gmail, showing the inbox and navigation menu. The Gmail interface includes a search bar, a list of mail folders (Inbox, Priority Inbox, Buzz, etc.), and a list of email messages with various icons and actions.

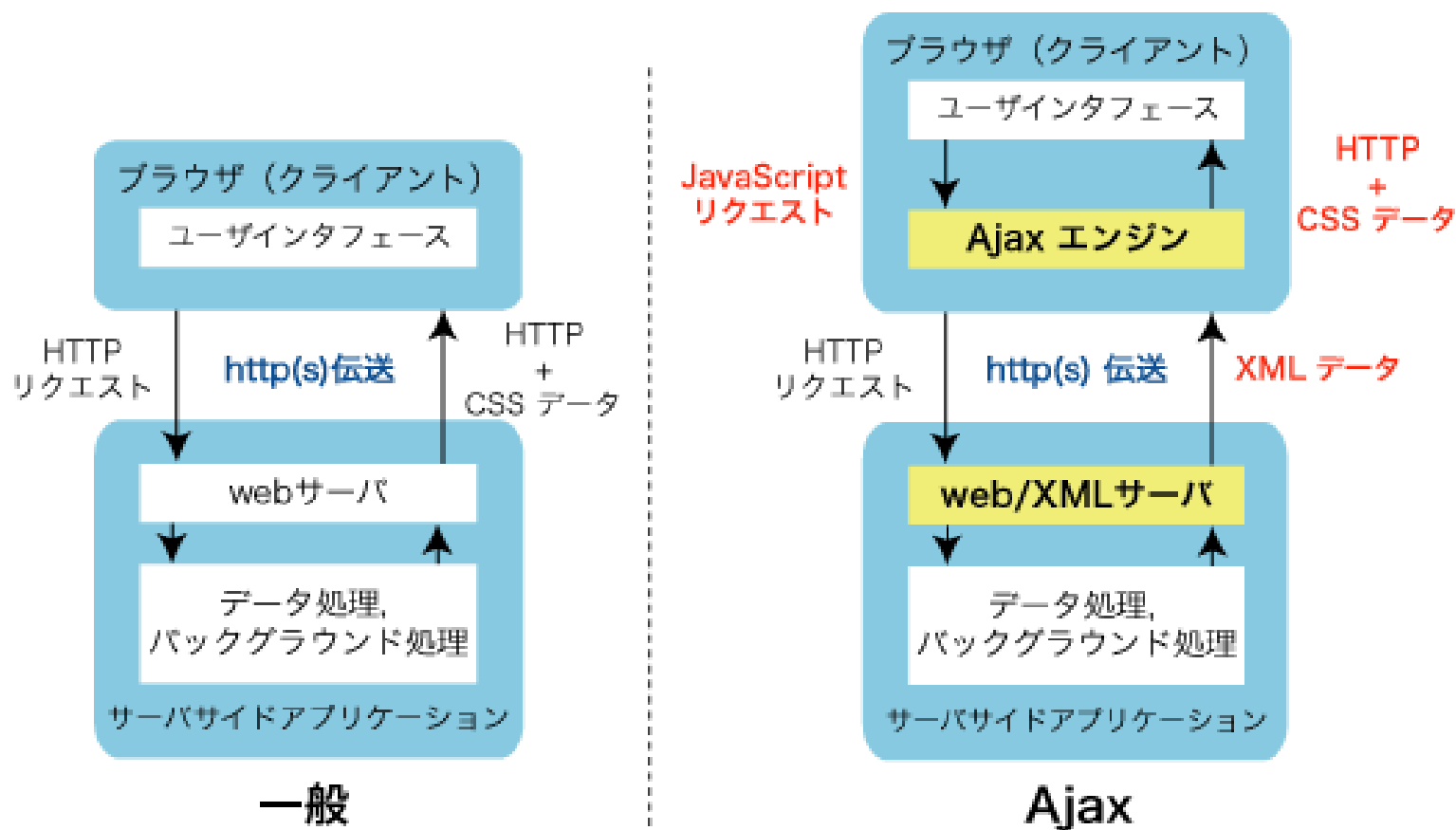
<http://maps.google.com/>

<http://mail.google.com/>



# Ajax のインパクト

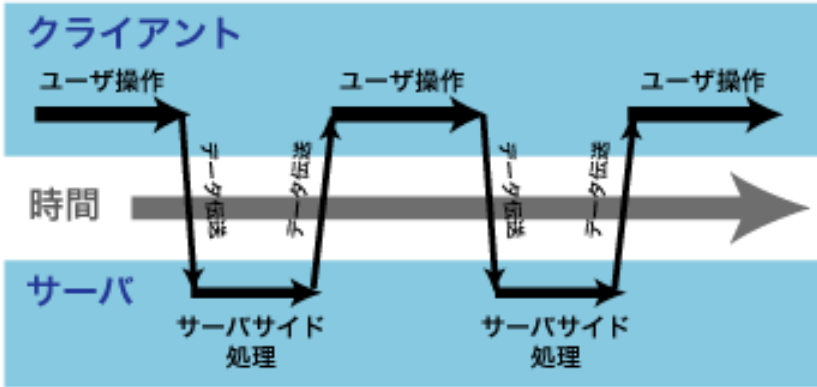
- サーバとやり取りしながら動的に変化



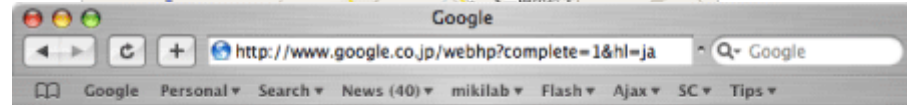
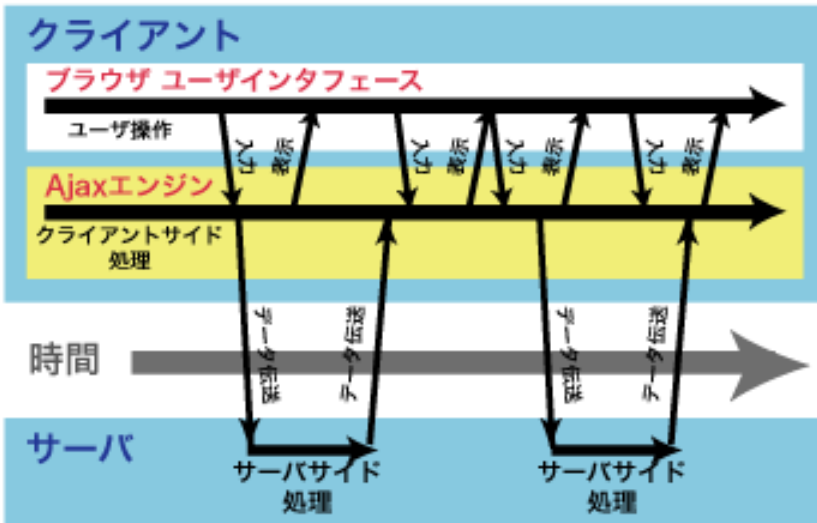


# これまでと Ajax 以降

## 一般のWebアプリケーション



## Ajax Webアプリケーション





# さて, Ajax

- Asynchronous JavaScript + XML
- XMLHttpRequestというJavaScriptのクラスを利用してページ遷移無く情報を変更することが可能
  - 動的にページを変更することが出来るため, ストレス無くユーザは使うことが出来るように!



# 試しに...

- uranai.html と uranai\_server.php を作成しよう！
  - <http://nkmr.io/lecture/> から開いてコピーして作成
  - uranai.txt ファイルから uranai.html を作成
  - uranai\_server.txt ファイルから uranai\_server.php を作成





<!-- 参考 <http://itpro.nikkeibp.co.jp/article/COLUMN/20060607/240192/> -->

```
<html><head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>うらないシステム</title>
<script>
window.onload = function(){
// [送信]ボタンをクリック時の処理を定義
var sendbutton = document.getElementById("sendbutton");
sendbutton.onclick = function() {
// 非同期通信を行うためのXMLHttpRequestオブジェクトを生成
try {
xmlReq = new XMLHttpRequest("Microsoft.XMLHTTP");
} catch(e) {
xmlReq = new XMLHttpRequest();
}
// サーバーからの応答時の処理を定義(結果のページへの反映)
xmlReq.onreadystatechange = function() {
var msg = document.getElementById("result");
if (xmlReq.readyState == 4) {
if (xmlReq.status == 200) {
msg.innerHTML = xmlReq.responseText;
} else {
msg.innerHTML = "通信に失敗しました。";
}
} else {
msg.innerHTML = "通信中...";
}
}
// サーバーとの通信を開始
xmlReq.open("GET","uranai_server.php?number=" + encodeURIComponent(document.fm.number.value),true);
xmlReq.send(null);
}
}
</script>
</head>
<body>
<form name="fm">
占いを行います。0~10までの数字を入力して下さい。 <br>
<input type="text" name="number" size="30">
<input type="button" value="Uranau" id="sendbutton">
<div id="result"></div>
</form>
</body>
</html>
```

# uranai.html

ボタンを押したら  
なにか送信！

受信したら結果を表示



# if文の中を変更してみよう！

```
<?php
// 参考: http://itpro.nikkeibp.co.jp/article/COLUMN/20060525/239029/?ST=develop
// 出力／内部文字コードをUTF-8に設定
mb_http_output('UTF-8');
mb_internal_encoding('UTF-8');
// 入力されたnumをキーに占いの結果を取得する
$num = $_GET['number'];
if( $num == 0 ){
    $result='今日の運勢はかなり良いです！ 幸せな一日が待っていることでしょう';
} else if( $num == 1 ){
    $result='今日の運勢はかなり良いでしょう～';
} else if( $num == 2 ){
    $result='だめだめですね！';
} else {
    $result='今日は最悪です！';
}
sleep(1); // 1秒休止(待ち時間を体感するためのダミー)
print($result); // 取得した結果を出力
?>
```

uranai\_server.php

PHPわからないと思うけど  
ここをProcessingのように  
書き換えてみるだけ！



# jQueryを使うと簡単に！

- \$.ajax( ... ) で実行可能 (jQuery.ajax( ... ) )

```
$.ajax({  
  url: "取得対象のURL",  
  dataType: "フォーマット (text, json, jsonp, xmlなど)",  
  success: function(data){  
    // 成功した時の処理 data は取得したデータ  
  },  
  error: function(xhr, status, err){  
    // 失敗した時の処理  
  }  
});
```

<!-- 参考 <http://itpro.nikkeibp.co.jp/article/COLUMN/20060607/240192/> -->

```
<html><head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>うらないシステム</title>
<script src="jquery-2.1.1.min.js"></script>
<script>
$(function(){
  $("#sendbutton").on("click",function(){
    console.log("click");
    $.ajax({ url:"uranai_server.php?number="+encodeURIComponent(document.fm.number.value),
      dataType: "text",
      success: function(data){
        console.log( data );
        $("#result").html( data );
      },
      error: function(xhr, status, err){
        $("#result").html( "通信に失敗しました" );
      }
    });
  });
});
</script>
</head>
<body>
<form name="fm">
  占いをを行います。0～10までの数字を入力して下さい。 <br>
  <input type="text" name="number" size="30">
  <input type="button" value="Uranau" id="sendbutton">
  <div id="result"></div>
</form>
</body>
</html>
```

uranai\_jquery.html



# Web API とは？

- API
  - Application Program Interface (何らかの機能をプログラミングするための仕組み)
  - メソッド名 + 引数で何らかの動作を実現する！
- Web API
  - Web上でアクセス可能なAPI
  - 様々な情報にアクセスすることが可能
    - 何かの緯度経度, キーワード検索結果, 画像検索結果, 商品検索, 書籍検索, ブックマーク数, 地図, 形態素解析, アニメ検索, Facebook, Twitter, メールなどなど
  - 一般的なWeb APIではURLで情報を取得

# Web API

- 例えばこんな感じ
  - <http://ma9.mashupaward.jp/apis>
  - <http://www.find-job.net/startup/api-2013>





# URI

<http://snakamura.org/software/index.html>



- 使える文字は英数字と一部の記号
  - -.~:@!\$&'()
  - 日本語を入力する場合は%エンコーディング
- URI は URL と URN を総称したもの
  - URL は Uniform Resource Locator
  - URN は Uniform Resource Name



# リクエストURL

`http://example.jp/search?query=test&area=10&...`

ベースURL

query=test

area=10

`http://example.jp/search`

ベースURL

query=test

query=test

area=10

area=10

ベースのURLのあと「?」が入り以降はオプション  
複数のオプションは「&」でつなぐ  
オプションは「=」で繋ぎ変数名と変数の値を指定





# 返り値はJSON/XML

- 返り値はあるデータフォーマット
  - JSONやXMLなどの形式

```
<staffs>
  <staff>
    <name>宮下芳明</name>
    <position>教授</position>
    <room>1018</room>
  </staff>
  <staff>
    <name>中村聡史</name>
    <position>准教授</position>
    <room>1007</room>
  </staff>
</staffs>
```

XML

```
{
  "staffs": {
    "staff": [
      {
        "name": "宮下芳明",
        "position": "教授",
        "room": "1018"
      },
      {
        "name": "中村聡史",
        "position": "准教授",
        "room": "1007"
      }
    ]
  }
}
```

JSON

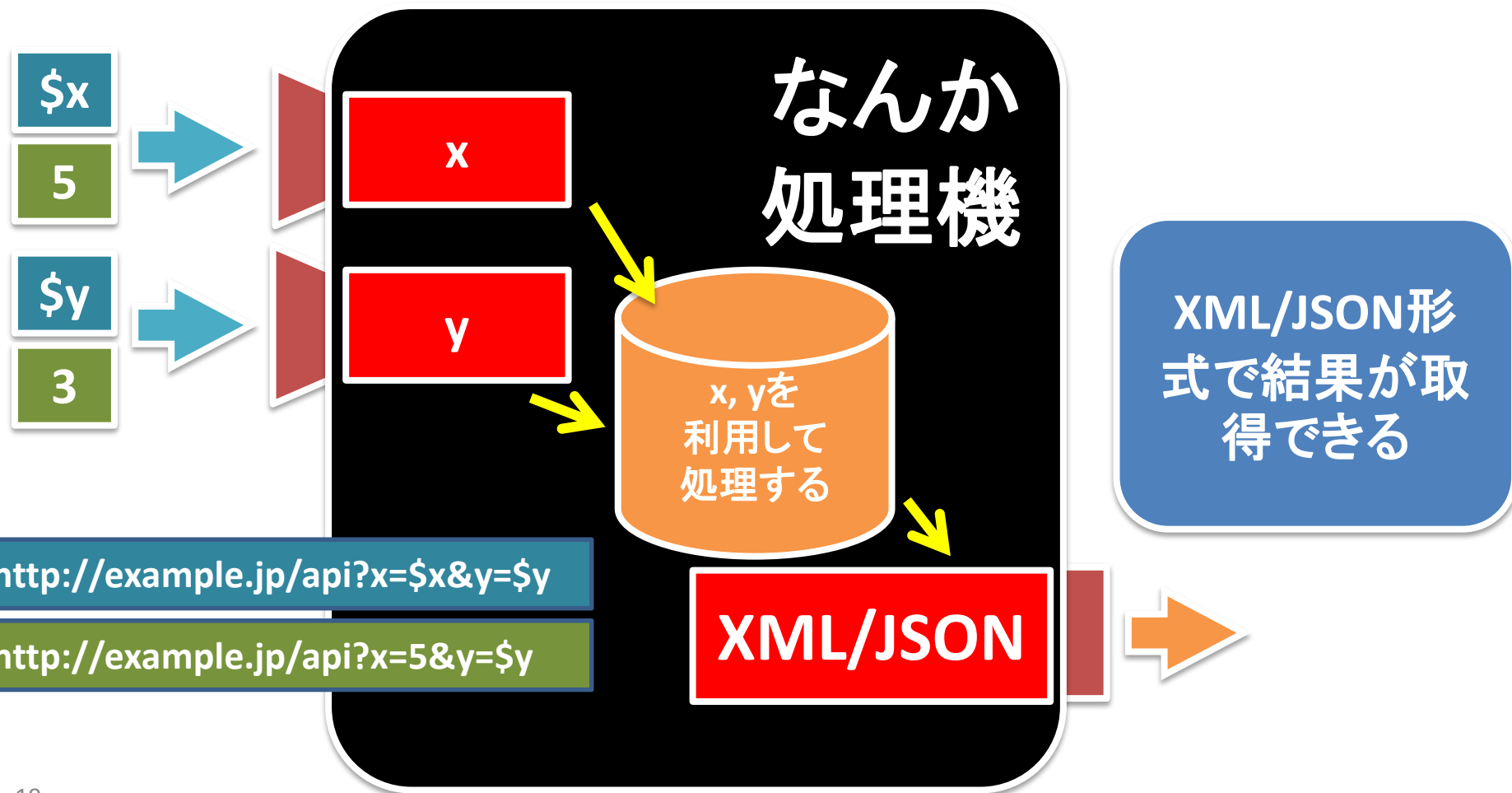


# 何ができるか？

- 一般的なAPIはメソッドとして用意されており, そこに引数を渡すことで何かの動作を実現する
  - `ellipse( 200, 200, 50, 50 );`
  - `dist( mouseX, mouseY, 200, 200 );`
- Web APIはGETリクエストであるURLに必要な情報を渡すことで何らかの結果を得る
  - `http://nkmr.io/api.php?person=homei`
  - `http://nkmr.io/api.php?x=50&y=30`

# Web APIの内部処理

- 複数の引数を受け取ることが可能





# JSONをよく見ると...

- [] で囲まれた部分が配列の定義となる

```
var a = [ 1, 2, 3, 4, 5 ];
```

```
a[1] = 5;
```

```
console.log( a[1] );
```

配列とオブジェクトの  
定義と全く同じ！

- {} で囲まれた部分がオブジェクトの定義となる

```
var human = { name: "宮下", age: 38,
```

```
              position: "准教授" };
```

```
human.position = "教授";
```

```
console.log( human.name + human.position );
```



# [演習] Panoramio APIを使おう

- Panoramio DATA APIを使ってみよう！

<http://www.panoramio.com/api/data/api.html>

– ある場所の周辺画像を集めてくるWeb API

– [http://www.panoramio.com/map/get\\_panoramas.php?set=public&from=0&to=20&minx=139.66&miny=35.70&maxx=139.67&maxy=35.71](http://www.panoramio.com/map/get_panoramas.php?set=public&from=0&to=20&minx=139.66&miny=35.70&maxx=139.67&maxy=35.71)

- from と to は表示する画像の番号
- minx と maxx は経度の範囲 (-180.0 ~ 180.0)
- miny と maxy は緯度の範囲 (-90.0 ~ 90.0)



# URLを分解してみる

- リクエストURLは下記のような感じ

[http://www.panoramio.com/map/get\\_panoramas.php  
?set=public&from=0&to=20&minx=139.66&miny=35.7  
0&maxx=139.67&maxy=35.71](http://www.panoramio.com/map/get_panoramas.php?set=public&from=0&to=20&minx=139.66&miny=35.70&maxx=139.67&maxy=35.71)

問い合わせ先 (APIの基本的なURL)

[http://www.panoramio.com/map/get\\_panoramas.php?](http://www.panoramio.com/map/get_panoramas.php?)

オプション (APIに送信する色々な情報)

set=public

from=0

to=20

:

maxy=35.71



# 返ってくるデータ一式

- JSON形式のデータになっている

```
* panoramio.json +
Save Find JSON Soft Tabs: 2 Please
1 {"count":216,"has_more":true,"map_location":
{"lat":35.705623000000003,"lon":139.66579458144909,"panoramio_zoom":2},"photos":
[{"height":375,"latitude":35.704909000000001,"longitude":139.66534000000001,"owner_id":78856,"owner_name":"chrisjongk
\u2022 archive","owner_url":"http://www.panoramio.com/user/78856","photo_file_url":"http://mw2.google.com/mw-
panoramio/photos/medium/548979.jpg","photo_id":548979,"photo_title":"Skyline looking west from
Nakano","photo_url":"http://www.panoramio.com/photo/548979","upload_date":"23 January 2007","width":500},
{"height":345,"latitude":35.709657,"longitude":139.663422999999999,"owner_id":66666,"owner_name":"Jeff
Metal","owner_url":"http://www.panoramio.com/user/66666","photo_file_url":"http://mw2.google.com/mw-
F count = 216
has_more = true
map_location = {
  lat = 35.705623000000003,
  long = 139.66579458144909,
  panoramio_zoom = 2
}
photos = {
  [写真の定義], {写真の定義} ... ]
}
45 height = 375
9 latitude = 35.704909000000001
US longitude = 139.66534000000001
:7 owner_id = 78856
\ owner_name = chrisjongking
0:
26 photo_file_url = http://mw2.google.com/mw-
01 panoramio/photos/medium/548979.jpg
39 upload_date = 23 January 2007
11 width = 500
5\u308a\u53e3","photo_url":"http://www.panoramio.com/photo/10234354","upload_date":"14 May 2008","width":375},
```

写真の定義

```
<html> <head> <script src="jquery-2.1.1.min.js"></script>
```

```
<script>
```

```
$(function(){
```

```
  $("#getdata").on("click",function(){
```

```
    var reqURL = "http://www.panoramio.com/map/get_panoramas.php?";
```

```
    var option = "set=public&from=0&to=20";
```

```
    option += "&minx=" + (139.6 - 0.1);
```

```
    option += "&miny=" + (35.7 - 0.1);
```

```
    option += "&maxx=" + (139.6 + 0.1);
```

```
    option += "&maxy=" + (35.7 + 0.1);
```

```
    $.ajax({
```

```
      url: reqURL + option,
```

```
      dataType: "jsonp",
```

```
      success: function(data){
```

```
        console.log(data);
```

```
      }
```

```
    });
```

```
  })
```

```
});
```

```
</script>
```

```
</head>
```

```
<body>
```

```
  <input type=button id="getdata" value="Get Data!">
```

```
  <div id="results"></div>
```

```
</body></html>
```

jqueryは各自適切なパスに変更

(139.6, 35.7)の周囲を取得

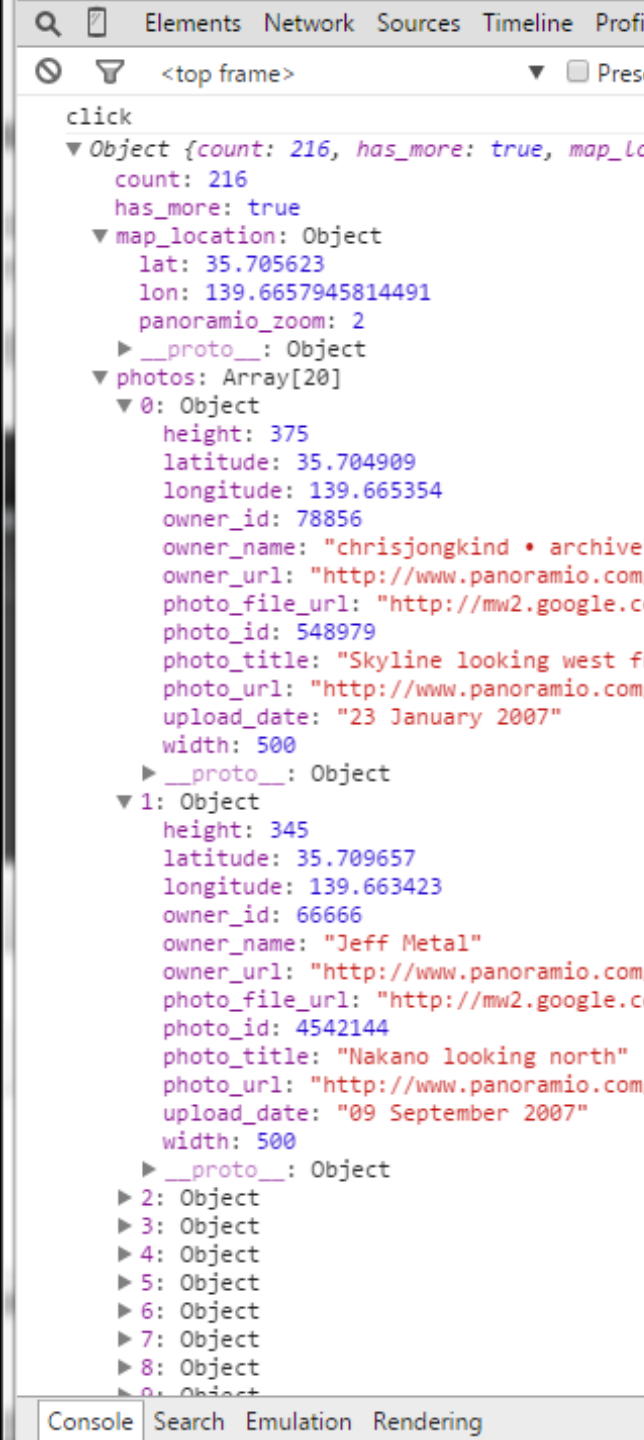
jsonpはjsonのクロスドメイン対応

console.log( 変数 ) で構造ごと表示

panoramio.html



```
success: function( data ){  
    // 全体の構造情報を表示  
    console.log( data );  
  
    // 数を表示  
    console.log( data.count );  
    // 緯度経度を表示  
    console.log( data.map_location.lat );  
    console.log( data.map_location.lon );  
  
    // 1枚目の写真の情報  
    console.log( data.photos[0].photo_title );  
    console.log( data.photos[0].photo_file_url );  
  
    // 2枚目の写真の情報  
    console.log( data.photos[1].photo_title );  
    console.log( data.photos[1].photo_file_url );  
}
```





# jQueryでのHTML操作

- `$("#要素の指定").html( "hogehoge" );`
  - 要素の中身をhogehogeに差し替え
- `$("#要素の指定").append( "hogehoge" );`
  - 要素の中身の最後にhogehogeを追加
- `$("#要素の指定").prepend( "hogehoge" );`
  - 要素の中身の最初にhogehogeを追加
- `$("#要素の指定").after( "hogehoge" );`
  - 要素の兄弟として後にhogehogeを追加
- `$("#要素の指定").before( "hogehoge" );`
  - 要素の兄弟として前にhogehogeを追加

```
<html> <head>
<script src="jquery-2.1.1.min.js"></script>
<script>
$(function(){
  $("#getdata").on("click",function(){
    var reqURL = "http://www.panoramio.com/map/get_panoramas.php?";
    var option = "set=public&from=0&to=20";
    option += "&minx=" + (139.6 - 0.1);
    option += "&miny=" + (35.7 - 0.1);
    option += "&maxx=" + (139.6 + 0.1);
    option += "&maxy=" + (35.7 + 0.1);
    $.ajax({
      url: reqURL + option,
      dataType: "jsonp",
      success: function(data){
        var len = data.length;
        console.log(data);
        for( var i=0; i<20; i++){
          $("#results").append( "" );
        }
      }
    });
  });
});
</script>
</head>
<body>
  <input type=button id="getdata" value="Get Data!">
  <div id="results"></div>
</body>
</html>
```

画像を表示する！

# 演習

- 位置を出身校の近辺や自宅近辺, 実家近辺, 今までに行ったことのある場所近辺に指定し, 周辺の写真を表示してみよう!

緯度経度は下記URLを参考に  
<http://www.geocoding.jp/>

```
Elements Network Sources Timeline Profiles Resources Audits Console
<top frame> Preserve log
click
▼ Object {count: 216, has_more: true, map_location: Object, photos: Array[20]}
  count: 216
  has_more: true
  ▼ map_location: Object
    lat: 35.705623
    lon: 139.6657945814491
    panoramio_zoom: 2
    ► __proto__: Object
  ▼ photos: Array[20]
    ▼ 0: Object
      height: 375
      latitude: 35.704909
      longitude: 139.665354
      owner_id: 78856
      owner_name: "chrisjongkind • archive"
      owner_url: "http://www.panoramio.com/user/78856"
      photo_file_url: "http://mw2.google.com/mw-panoramio/photos/medium/54899"
      photo_id: 548979
      photo_title: "Skyline looking west from Nakano"
      photo_url: "http://www.panoramio.com/photo/548979"
      upload_date: "23 January 2007"
      width: 500
      ► __proto__: Object
    ▼ 1: Object
      height: 345
      latitude: 35.709657
      longitude: 139.663423
      owner_id: 66666
      owner_name: "Jeff Metal"
      owner_url: "http://www.panoramio.com/user/66666"
      photo_file_url: "http://mw2.google.com/mw-panoramio/photos/medium/4542144"
      photo_id: 4542144
      photo_title: "Skyline looking west from Nakano"
      photo_url: "http://www.panoramio.com/photo/4542144"
    ► 7: Object
    ► 8: Object
    ► 9: Object
```



Geocoding

明治大学中野キャンパス

TOKYO097

検索

明治大学中野キャンパス, 〒164-8525 東京都中野区 4丁目 21-1(2) の座標(WGS84)

緯度:35.706962 経度:139.659547

[Google Earthを開く](#)

地図中央値(WGS84): 緯度 35度42分25.063秒(35.706962), 経度 139度39分34.3

この情報を利用する



minx と maxx が経度  
miny と maxy が緯度  
に対応



minx = 139.659547 - 0.1;  
maxx = 139.659547 + 0.1;  
miny = 35.706962 - 0.1;  
maxy = 35.706962 + 0.1;

Mobile & PC



Powered by imacoco.org

```
<html> <head> <script src="jquery-2.1.1.min.js"></script>
```

```
<script>
```

```
$(function(){
```

```
  $("#getdata").on("click",function(){
```

```
    var reqURL = "http://www.panoramio.com/map/get_panoramas.php?";
```

```
    var option = "set=public&from=0&to=20";
```

```
    option += "&minx=" + (139.659547 - 0.1);
```

```
    option += "&miny=" + (35.706962 - 0.1);
```

```
    option += "&maxx=" + (139.659547 + 0.1);
```

```
    option += "&maxy=" + (35.706962 + 0.1);
```

```
    $.ajax({
```

```
      url: reqURL + option,
```

```
      dataType: "jsonp",
```

```
      success: function(data){
```

```
        console.log(data);
```

```
        for( var i=0; i<data.photos.length; i++ ){
```

```
          $("#results").append( "" );
```

```
        }
```

```
      }
```

```
    });
```

```
  })
```

```
});
```

```
</script>
```

```
</head>
```

```
<body>
```

```
  <input type=button id="getdata" value="Get Data!">
```

```
  <div id="results"></div>
```

```
</body></html>
```

各自値を入れてみよう！

data.photos.lengths で  
配列の数を取得できるので利用



# 現在位置を取得しよう

- navigator.geolocation.getCurrentPosition を使うだけで、手軽に座標を取得することが可能！
  - ブラウザに機能として備わっている！
  - (古いブラウザだと未対応)
- アクセス時に、場所情報にアクセスしてよいかという質問が表示されるのでOKを押す！



# 現在位置を取得(クリック後に)

```
navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
```

```
function successCallback(position) {
```

```
  console.log( "緯度:" + position.coords.latitude );
```

```
  console.log( "経度:" + position.coords.longitude );
```

```
}
```

```
function errorCallback(error) {
```

```
  console.log( "エラーが発生しました" + error.code );
```

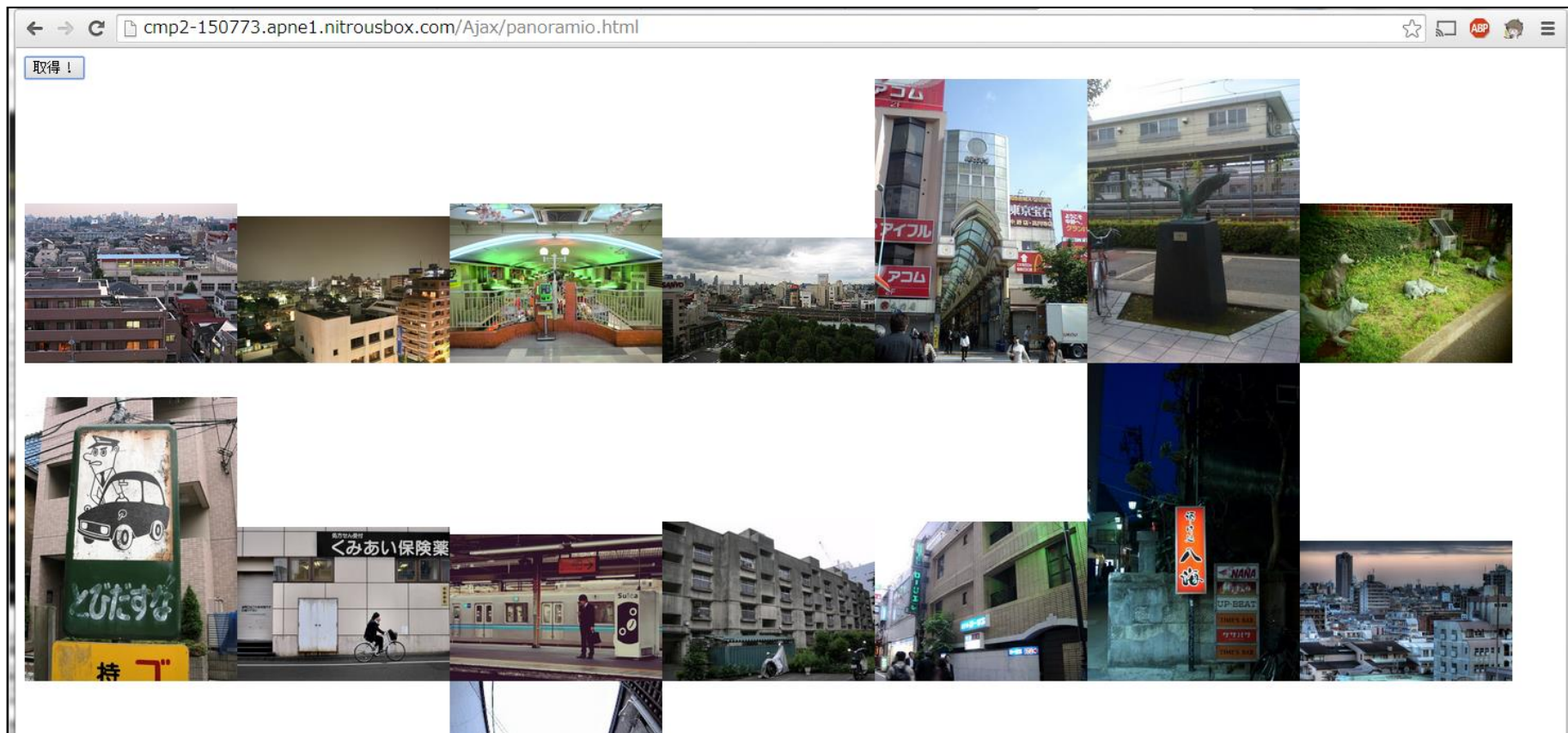
```
}
```

コンソールを確認してみよう！



# 演習

- 取得した現在位置をもとに，周辺の画像を表示してみましょう！



# ヒント

1. ボタンがクリックされたという情報を取得
2. ボタンがクリックされたら現在位置を取得
3. 現在位置の取得に成功したらその周辺の画像をPanoramio DATA APIに問い合わせ
4. Panoramio DATA APIから得られた結果を利用して画像をHTMLに埋め込み表示

# 骨組み

```
<script>
$(function(){
  $("#getdata").on("click",function(){
    navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
    function successCallback( position ){
      // 成功したらPanoramioから画像を取得する
      var reqURL = "http://www.panoramio.com/map/get_panoramas.php?";
      var option = "オプションにpositionを利用して色々指定";
      $.ajax({
        url: reqURL + option,
        dataType: "jsonp",
        success: function(data){
          console.log(data);
          // 画像を表示するプログラムを書く
        }
      });
    }
    function errorCallback( error ){ }
  })
});
</script>
```

# 例

```
<script>
$(function(){
  $("#getdata").on("click",function(){
    navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
    function successCallback( position ){
      var reqURL = "http://www.panoramio.com/map/get_panoramas.php?";
      var option = "set=public&from=0&to=20";
      option += "&minx=" + (position.coords.longitude - 0.1);
      option += "&miny=" + (position.coords.latitude - 0.1);
      option += "&maxx=" + (position.coords.longitude + 0.1);
      option += "&maxy=" + (position.coords.latitude + 0.1);
      $.ajax({
        url: reqURL + option,
        dataType: "jsonp",
        success: function(data){
          console.log(data);
          $("#results").html(""); // htmlを空っぽにする
          for( var i=0; i<data.photos.length; i++ ){
            $("#results").append( "" );
          }
        }
      });
    }
    function errorCallback( error ){ }
  })
});
</script>
```

panoramio\_geo.html



# 地図を使ってみよう！

- 現在位置をもとに地図を表示しよう！
- 地図APIはGoogle MapsやYahoo! 地図APIなど様々なものがある
- 基本的にWeb APIは開発者用のサイトでIDを発行する必要あり(次ページ以降手順参照)



# 取得した緯度経度を表示！

- Yahoo! 地図APIを利用してみよう！

developer.yahoo.co.jp

**YAHOO! JAPAN** デベロッパーネットワーク IDでもっと便利に新規取得 ログイン

Yahoo! JAPAN ヘルプ

子ども社会の「大人不在」が、あらゆる子どもの問題を生む

アプリケーションの管理 サービス一覧 サポート ブログ

お知らせ ショッピング ボックス オークション YConnect アプリ販売

**50本のAPIをフル公開!**

出品APIをつかって、ビジネスをはじめよう

Yahoo! ショッピングAPI連携デベロッパー募集!

詳細はこちら >

いいね! シェア 263人が「いいね!」と言っています。「いいね!」をクリックして、友達に知らせましょう。

ツイート 134

**Yahoo!デベロッパーネットワークを活用しよう**

- 探そう** 使いたいAPIやサービスを探そう
- 作ろう** アプリケーションIDを登録して開発しよう
- つながろう** 開発者同士のコミュニティを活用しよう

**法人の方へ**

WebAPIの商用利用について

**Yahoo!サービス**

- Yahoo!プロモーション広告の検索結果に広告掲載
- Yahoo!ウェブオーナーセンターを掲載してサイトの収益アップ!
- Yahoo!ビジネスセンター あなたのサイトビジネスをサポート

**2014年5月26日**  
Yahoo!ショッピング 注文APIをテスト環境にてリリースしました! **NEW!**

**2014年5月19日**  
Bashoジャパン主催「Riak Meetup #4」を開催します

**2014年5月13日**  
【内容変更】ヤフオク!APIのサーバーメンテナンスについて

**http://developer.yahoo.co.jp/**



# まずYahoo! Japan IDでログイン

**YAHOO! JAPAN** デベロッパーネットワーク IDでもっと便利に[新規取得](#)  
[ログイン](#)

[Yahoo! JAPAN - ヘルプ](#)

[デベロッパーネットワークトップ](#) > [アプリケーションの管理](#)

アプリケーションの管理

ログイン

Yahoo! JAPAN Web APIのご利用には、アプリケーションを登録し[アプリケーションID](#)を発行する必要があります。

アプリケーションIDは、ア  
Yahoo! JAPAN IDに対し

アプリケーションIDは、ユ  
為を防ぐために使用され

で、複数のアプリケーションに固有のIDを付与し、アプリケーションごとに登録されています。

アプリケーションIDは、第三者に開示したり、第三者が参照できる場所に記載しないでください。

**必要に応じて  
アカウントを取得しよう！**

続けるには、Yahoo! JAPAN IDでログインする必要があります。

[ログイン](#)

Yahoo! JAPAN IDをお持ちでない方はこちら

[>>Yahoo! JAPAN IDを取得\(無料\)](#)

※ Yahoo! JAPANにおける個人情報の取り扱いについては、[プライバシーポリシー](#)をご覧ください。

 [ご利用ガイド](#)

Web APIとその他のサービス

API

[▶ ボックス](#)

[▶ ショッピング](#)

[▶ オークション](#)

[▶ YOLP\(地図\)](#)

[▶ テキスト解析](#)

[▶ 知恵袋](#)

[▶ ニュース](#)

[▶ カテゴリ](#)

[▶ 震災関連情報](#)

[▶ Social API](#)

その他

[▶ YConnect](#)

[▶ YUI](#)

[▶ マーケット](#)



# 新しいアプリケーションを開発

**YAHOO!** JAPAN デベロッパーネットワーク ようこそ、todoviewer さん  
[ログアウト](#)

Yahoo! JAPAN - [ヘルプ](#)

[デベロッパーネットワークトップ](#) > [アプリケーションの管理](#)

## アプリケーションの管理

[ご利用ガイド](#)

アプリケーション一覧

[新しいアプリケーションを開発](#)

[Show Bar\(サーバーサイド\)](#)

作成日:2014年5月28日

[編集](#) [削除](#)

[利用状況](#)

[test](#)

作成日:2012年7月24日

[編集](#) [削除](#)

[利用状況](#)

[SyncRerank](#)

作成日:2009年10月22日

[編集](#) [削除](#)

[利用状況](#)

作成日:2007年9月27日

[編集](#) [削除](#)

[利用状況](#)

## Web APIとその他のサービス

### API

- [ボックス](#)
- [ショッピング](#)
- [オークション](#)
- [YOLP\(地図\)](#)
- [テキスト解析](#)
- [知恵袋](#)
- [ニュース](#)
- [カテゴリ](#)
- [震災関連情報](#)
- [Social API](#)

### その他

- [YConnect](#)
- [YUI](#)
- [マーケット](#)
- [メール](#)
- [OAuth1.0](#)



# 必要な情報を入力！

YAHOO! JAPAN デベロッパーネットワーク ようこそ、todoviewer さん  
[ログアウト](#)

[Yahoo! JAPAN](#) - [ヘルプ](#)

[デベロッパーネットワークトップ](#) > [アプリケーションの管理](#) > [新しいアプリケーションを開発](#)

新しいアプリケーションを開発

アプリケーション情報の入力

Yahoo! JAPANが提供するWeb APIを利用したアプリケーション開発を行っていただくにあたって必要な情報をご登録ください。  
 ご登録いただく情報は、アプリケーションの利用状況の把握や、アプリケーションの不正利用を防ぐ目的で使用いたします。

\* 入力必須項目

Yahoo!ショッピングのストア運営をサポートするAPIをご利用の場合は[こちら](#)より登録してください。

Web APIを利用する場所

- \* アプリケーションの種類
    - サーバーサイド  
 サーバー上など、秘密鍵を安全に保管できる場所からWeb APIを使いたい場合に選択してください。  
 OAuth 2.0 Authorization Codeフローによるユーザー属性情報が利用できます。
    - クライアントサイド  
 スマートフォンのネイティブアプリなど、クライアントアプリから直接Web APIを使いたい場合に選択してください。  
 OAuth 2.0 Implicitフローによるユーザー属性情報が利用できます。
- ※ [テキスト解析API](#)や[地図API](#)など、公開情報を扱うWeb APIはどちらを選択しても利用できます。

アプリケーションの基本情報

Yahoo! JAPAN ID todoviewer

\* 連絡先メールアドレス  [メールアドレスを追加・編集する](#)

Yahoo! JAPAN IDの登録情報で確認済みの登録メールアドレスと、Yahoo!メールアドレスから選択できるメールアドレスをお送りする

アプリケーション名はお好きにどうぞ

\* アプリケーション名  [?](#)

(50文字まで)

サイトURLはnitrous.ioのURL [?](#)

\* サイトURL  [?](#)

(半角英数字200文字まで)  
 利用者同意画面に表示されます。


 YAHOO! JAPAN! デベロッパーネットワーク ようこそ、todoviewer さん  
[ログアウト](#)
[Yahoo! JAPAN - ヘルプ](#)
[デベロッパーネットワークトップ](#) > [アプリケーションの管理](#) > 入力内容の確認

新しいアプリケーションを開発

## 入力内容の確認

入力内容を確認し、正しければ「登録」ボタンを押してください。

内容を訂正する場合は、「戻る」ボタンを押してください。

\* 入力必須項目

## Web APIを利用する場所

\* アプリケーションの種類    サーバーサイド

## アプリケーションの基本情報

Yahoo! JAPAN ID	todoviewer
* 連絡先メールアドレス	nakamura.satoshi@gmail.com
* アプリケーション名	Show Bar
サイト URL	http://nkmr.io/
アプリケーションの説明	< 未記入 >
利用するスコープ	< なし >

## ガイドラインを確認する

 \* [ガイドライン](#)に同意しますか?    同意する

ご注意

- ・複数のアプリケーションを開発される場合は、その都度ご登録ください。
- ・アプリケーションID、シークレットは第三者に開示しないようにしてください。

登録

戻る

# アプリケーションIDを取得

**YAHOO!** JAPAN デベロッパーネットワーク ようこそ、todoviewer さん  
[ログアウト](#)

[Yahoo! JAPAN](#)

[デベロッパーネットワークトップ](#) > [アプリケーションの管理](#) > 登録完了

新しいアプリケーションを開発

登録完了

登録が完了し、アプリケーションIDおよびシークレットが発行されました。

アプリケーションID:

`dj0zaizpPU45R29mZjYidUhTNIzZpWNvbnN1bWYyc2YjcmY0Jng9NDI-`

シークレット:

**このアプリケーションIDをコピーして利用！**

さっそくアプリケーションを作ってみましょう！

## 1. ブラウザーから直接リクエストする

[ショッピング商品検索API](#)で「テレビ」カテゴリ内の売れ筋商品を検索するには、以下のURLにアクセスします。

[http://shopping.yahooapis.jp/ShoppingWebService/Y1/itemSearch?appid=dj0zaizpPU45R29mZjYidUhTNIzZpWNvbnN1bWYyc2YjcmY0Jng9NDI-&category\\_id=635&sort=](http://shopping.yahooapis.jp/ShoppingWebService/Y1/itemSearch?appid=dj0zaizpPU45R29mZjYidUhTNIzZpWNvbnN1bWYyc2YjcmY0Jng9NDI-&category_id=635&sort=)

## 2. プログパーツにしてWebサイトに埋め込めるようにする



# 地図を表示するには？

**YAHOO! JAPAN** デベロッパーネットワーク ようこそ、[todoviewer](#) さん  
ログアウト

Yahoo! JAPAN ヘルプ

デザイナーズ住宅も夢じゃない？ ふり返る家、とは。

アプリケーションの管理	サービス一覧▼	サポート▼	ブログ
-------------	---------	-------	-----

デベロッパーネットワークトップ > YOLP(地図)

## YOLP(地図)

### Yahoo! Open Local Platformで、地図や地域情報サイトを開発

[Yahoo! Open Local Platform\(YOLP\)](#)は、Yahoo!JAPANがデベロッパー向けに提供する地図・地域情報のAPI・SDKです。ウェブページの制作はもちろん、スマートフォン向けアプリケーションの開発においても、多彩な地図表示、店舗・施設検索、ジオコーディング、ルート検索、標高地取得などの豊富な機能をご活用いただけます。

#### お知らせ

2014年4月8日 [Yahoo!ローカルサーチAPIのレスポンス変更のお知らせ](#)

2014年4月2日 [【重要なお知らせ】「YOLPカセットギャラリー」サービス終了のお知らせ](#)

2014年3月17日 [【重要なお知らせ】YOLP\(地図\)旧APIおよびSilverlightマップAPI,SDK サービス終了について](#)

## ウェブサービスで地図を使う

### Yahoo! JavaScriptマップAPI NEW!

マウスでドラッグできる地図をウェブページに自由に貼り付けて利用できます。雨雲レーダー情報を重ねて表示できます。

### Yahoo! スタティックマップAPI NEW!

地図の画像(PNG形式やJPEG形式など)をウェブページに自由に貼り付けて利用できます。

#### はじめよう

アプリケーションを開発

#### 法人の方へ

WebAPIの商用利用について

ツイート 104

いいね! 148

#### 開発のヒント

- Web APIの使い方(GET)
- サンプルコード集
- Web APIライブラリ集
- 爆速JSONP
- ドキュメント
- 開発者コミュニティ
- Web APIを利用したサービス
- Yahoo! JAPANが提供するRSS一覧
- クレジットの表示
- 用語集
- エラーメッセージおよびコード
- 利用制限について

▼開発のヒント



# 地図を表示してみよう

```
<html>
<head>
<script src="jquery-2.1.1.min.js"></script>
<script src="http://js.api.olv.yahooapis.jp/OpenLocalPlatform/V1/jsapi?appid=XXXXXXXXXX"></script>
<script>
$(function(){
  $("#sendbutton").on("click",function(){
    var ymap = new Y.Map("div_map");
    ymap.drawMap(new Y.LatLng( 35.706962, 139.659547 ), 17, Y.LayerSetId.NORMAL);
  })
});
</script>
</head>
<body>
<input type=button value="Get Data!" id="sendbutton">
<div id="geo_result" ></div>
<div id="div_map" style="width:400px; height:300px"></div>
</body>
</html>
```

アプリケーションID ↑↑

yahoo\_map.html

緯度経度は  
各自設定してみよう！



# じゃあ，これを現在位置から

- 緯度経度は
  - `position.coords.longitude`
  - `position.coords.latitude`
  - で取得できる！じゃあ，その取得した緯度経度を適用して地図画像を表示してみよう！

```
<html>
<script src="jquery-2.1.1.min.js"></script>
<script src="http://js.api.olv.yahooapis.jp/OpenLocalPlatform/V1/jsapi?appid=XXXXXXXXXX"></script>
<script>
$(function(){
  $("#sendbutton").on("click",function(){
    navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
    function successCallback(position) {
      $("#geo_result").append( "緯度:" + position.coords.latitude );
      $("#geo_result").append( "<br>" );
      $("#geo_result").append( "経度:" + position.coords.longitude );

      var ymap = new Y.Map("div_map");
      ymap.drawMap(new Y.LatLng( position.coords.latitude, position.coords.longitude ),
        17, Y.LayerSetId.NORMAL);
    }
    function errorCallback(error) {
      // 省略
    }
  })
});
</script>
<body>
<input type="button" value="Get Data!" id="sendbutton">
<div id="geo_result" ></div>
<div id="div_map" style="width:400px; height:300px"></div>
</body>
</html>
```

アプリケーションID ↑↑

yahoo\_map\_geo.html



# 地図が動いたことをどう検知？

- Y.Mapにはbindというメソッドがあり, これを利用してイベントに機能を割り当てる！
  - click: クリックされた位置の緯度経度座標を取得
  - dblclick: ダブルクリックされた位置の緯度経度を取得
  - movestart: マウスによるドラッグが開始された時
  - moveend: マウスによるドラッグが終了した直後
  - zoomstart: 縮尺が変更される直前
  - zoomend: 縮尺が変更された後
  - load: 地図が最初に描画された時



```
<html>
<script src="jquery-2.1.1.min.js"></script>
<script src="http://js.api.olp.yahooapis.jp/OpenLocalPlatform/V1/jsapi?appid=XXXXXXXXXX"></script>
<script>
$(function(){
  $("#sendbutton").on("click",function(){
    navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
    function successCallback(position) {
      $("#geo_result").append( "緯度:" + position.coords.latitude );
      $("#geo_result").append( "<br>" );
      $("#geo_result").append( "経度:" + position.coords.longitude );
      var ymap = new Y.Map("div_map");
      ymap.drawMap(new Y.LatLng( position.coords.latitude, position.coords.longitude ),
        15, Y.LayerSetId.NORMAL);
      ymap.bind( "moveend", function(){
        console.log( "地図の中心が動いたよ！" );
      });
    }
    function errorCallback(error) { }
  })
});
</script>
<body>
<input type="button" value="取得！" id="sendbutton">
<div id="geo_result" ></div>
<div id="div_map" style="width:400px; height:300px"></div>
</body>
</html>
```



# 移動した時の中心座標取得

```
ymap.bind( "moveend", function(){  
    console.log( "地図の中心が動いたよ！" );  
  
    // 中心座標(緯度経度)を取得  
    var latlng = ymap.getCenter();  
    console.log( latlng );  
    console.log( "緯度:" + latlng.lat() );  
    console.log( "経度:" + latlng.lon() );  
});
```

地図の中心座標を取得するための  
getCenter()で返されるlatlngには  
緯度経度を取得するための  
lat(), lon()というメソッドがある！

# [演習] 場所移動で何か表示

- 地図を動かしていき、東京駅の近くまで来たら「東京駅です！」と表示するプログラムを作成しよう！
- ヒント
  1. 東京駅の緯度経度を[www.geocoding.jp](http://www.geocoding.jp)から取得
  2. 地図が動いたら中心座標を取得
  3. 地図の中心座標から東京駅の緯度経度の距離が近い時に、alertで東京駅です！と表示する！



# 仕組みとしてはこんな感じ

```
ymap.bind( "moveend", function(){
  console.log( "地図の中心が動いたよ！" );

  // 中心座標(緯度経度)を取得
  var latlng = ymap.getCenter();
  console.log( latlng );
  console.log( "緯度:" + latlng.lat() );
  console.log( "経度:" + latlng.lon() );
  if( latlng.lat() > ??? && latlng.lat() < ??? &&
      latlng.lon() > ??? && latlng.lon() < ??? ){
    alert( "東京駅です！" );
  }
});
```



# [演習] 組み合わせてみよう！

- 現在位置を取得し，その地図を表示するとともに，周辺の画像を表示してみましよう！
- ヒント
  1. 地図が動いたら中心座標を取得
  2. 中心座標の取得に成功したらその周辺の画像をPanoramio DATA APIに問い合わせ
  3. Panoramio DATA APIから得られた結果を利用して画像をHTMLに埋め込み表示



# showPanoramio(x, y)を作る

- (x, y)周辺の写真をPanoramioから取得して表示

```
function showPanoramio( x, y ){
  var reqURL = "http://www.panoramio.com/map/get_panoramas.php?";
  var option = "set=public&from=0&to=20";
  option += "&minx=" + (x - 0.1);
  option += "&miny=" + (y - 0.1);
  option += "&maxx=" + (x + 0.1);
  option += "&maxy=" + (y + 0.1);
  $.ajax({
    url: reqURL + option,
    dataType: "jsonp",
    success: function(data){
      console.log(data);
      $("#results").html(""); // htmlを空っぽにする
      for( var i=0; i<data.photos.length; i++ ){
        $("#results").append( "" );
      }
    }
  });
}
```



```
$(function(){
  function showPanoramio(){
    // ここに前ページのように定義
  }

  $("#sendbutton").on("click",function(){
    navigator.geolocation.getCurrentPosition(successCallback, errorCallback);
    function successCallback(position) {
      $("#geo_result").append( "緯度:" + position.coords.latitude );
      $("#geo_result").append( "<br>" );
      $("#geo_result").append( "経度:" + position.coords.longitude );
      var ymap = new Y.Map("div_map");
      ymap.drawMap(new Y.LatLng( position.coords.latitude, position.coords.longitude ),
        15, Y.LayerSetId.NORMAL);
      ymap.bind( "moveend", function(){
        var latlng = ymap.getCenter();
        console.log( latlng );
        showPanoramio( latlng.lon(), latlng.lat() );
      });
    }
    function errorCallback(error) { }
  })
});
```

# 領域をもっとしぼってみる

- 方針

- 現在位置を取得して、その位置に応じて地図表示
- function `showPanoramioEx( x1, y1, x2, y2 ){ ... }`を定義して、 $(x1, y1) - (x2, y2)$ で囲まれた領域の写真をPanoramioから取得して表示
- 地図の中心位置が動く度に、地図の左上、右下の緯度経度を取得して、その区域内的の画像を`showPanoramioEx`を利用して表示





# 左上右下の緯度経度の取得

地図を400x300で表示している時, 左上の緯度経度は

```
var lt = ymap.fromContainerPixelToLatLng(new Y.Point(0, 0));
```

右下の緯度経度は

```
var rb = ymap.fromContainerPixelToLatLng(new Y.Point(400,300));
```

で取得することが可能. これを利用すると, 地図で囲まれている領域を取得及び利用することができる!

minx maxx がそれぞれどちらになるか計算

miny maxy がそれぞれどちらになるか計算

showPanoramioExで表示!



# XMLも取得しよう！

- Asynchronous JavaScript + XML なので、もともとはXMLの情報を取得するもの
  - JSON は JavaScript Object Notationという名前で、JavaScriptから使うことを前提とした軽量フォーマット



# XMLとは

- eXtensible Markup Language
- W3C (World Wide Web Consortium) で採択されたWeb上でのデータのやりとりに注目した構造化文書記述のためのデータフォーマット
- XMLの特徴
  - 新しいタグを定義することが可能
  - 構造は任意の形でネスト可能(繰り返し)
  - XMLはデータ記述言語であり, 表示能力は持っていない (HTMLとの違い. 表示にはCSSなどを使用)

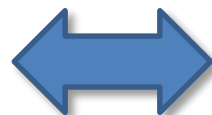


# XMLのメリット

- データを機械可読な形に記述可能
  - HTMLは機械での認識が難しい
- 関係データベースで表現できない半構造データを扱うことが可能
  - 生物学のデータ
  - Web上の各種データ

# HTMLとXMLの違い

```
<h1> 日本代表</h1>
<h2> GK </h2>
<ul> <li> 川島 </li> </ul>
<h2> DF </h2>
<ul>
<li> 中澤 </li> <li> 田中 </li>
<li> 長友 </li> <li> 駒野 </li>
</ul>
<h2> MF </h2>
<ul>
<li>遠藤 </li> <li> 阿部 </li>
<li>長谷部 </li> <li> 大久保 </li>
<li>松井 </li>
</ul>
<h2> FW </h2>
<ul> <li> 本田 </li> </ul>
<h2> 監督 </h2>
<ul> <li> 岡田 </li> </ul>
```



```
<Team>
<Name> 日本代表 </Name>
<GK>
<Player> 川島 </Player>
</GK>
<DF>
<Player> 中澤 </Player>
<Player> 田中 </Player>
<Player> 長友 </Player>
<Player> 駒野 </Player>
:
<Player> 大久保 </Player>
<Player> 松井 </Player>
</MF>
<FW>
<Player> 本田 </Player>
</FW>
<Director> 岡田 </Director>
</Team>
```

# XML の構成要素

- 要素：XMLの1単位
  - `<team> ... </team>`, `<player> ... </player>`
  - 空要素にもなりうる：`<director></director>`
- タグ：team, GK, MF, ..., player, director など
- 開始タグ：`<team>`
- 終了タグ：`</team>`
- 属性：要素の中で指定する属性
  - `<player position="GK" number="1" ...>`
  - 属性は開始タグの中で指定

# 様々な表現形式

```
<player position="GK" number="1" >
```

榎崎正剛

```
</player>
```

```
<player>
```

```
<name>榎崎正剛</name>
```

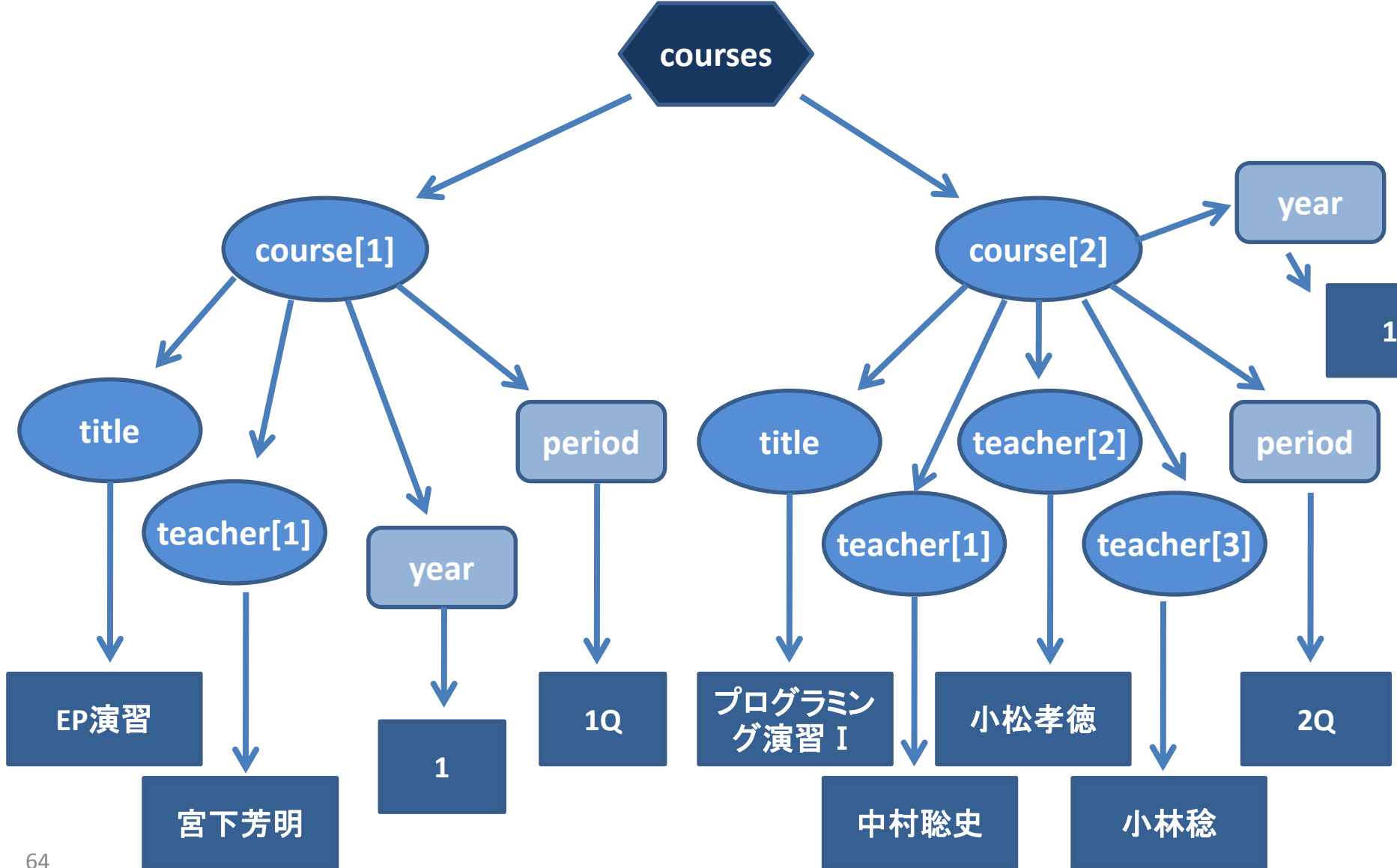
```
<position>GK</position>
```

```
<number>1</number>
```

```
</player>
```



# XMLの構造







# 演習の準備

- 下記のURLから該当するファイルをダウンロードして作業フォルダにアップロードせよ(プログラムと同じフォルダでよい)
  - <http://nkmr.io/lecture/2014/fms.xml>
- \$.ajax を利用して中身を読み込んでみよう！

# 開くと →

- どうやって取得する？
  - 教員リスト
  - 専門リスト
  - URLリスト
  - 講義名リスト

```
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <name>先端メディアサイエンス学科</name>
  <teachers>
    <teacher>
      <name>宮下 芳明</name>
      <link>http://homei.com/</link>
      <major>インタラクション</major>
      <room></room>
      <lectures>
        <lecture>
          <name>エンタテイメントプログラミング演習</name>
          <year>1</year>
          <term>春</term>
        </lecture>
        <lecture>
          <name>コンテンツエンタテイメント概論</name>
          <year>1</year>
          <term>秋</term>
        </lecture>
        <lecture>
          <name>コンテンツメディアプログラミング実習2</name>
          <year>2</year>
          <term>秋</term>
        </lecture>
      </lectures>
    </teacher>
    <teacher>
      <name>小松 孝徳</name>
      <link>http://www.tkomat-lab.com/</link>
      <major>認知心理学</major>
      <lecture>
        <year>1</year>
        <term>春</term>
      </lecture>
    </teacher>
  </teachers>
</data>
```



# XMLの値の取り方

- jQuery でXMLの要素の値を取得する方法
  - `$( 要素 ).text();`
  - `$( 要素 ).find( 探したい要素 ).text();`
- jQuery でXMLの要素の集合を取得する方法
  - `$( 要素 ).find( 探したい要素 ).each( 各々の処理 );`
  - 「each」というのは英語の示す通り1つずつ
  - 各々の処理は関数として書く
    - 関数の中では find の結果(要素)を `$(this)` というもので取得することが可能



# やってみよう

- URLリストの取得

```
$(function(){
  $.ajax({
    url: "fms.xml",
    dataType: "xml",
    success: function(data){
      $(data).find( "link" ).each(
        function(){
          console.log( $(this).text() );
        }
      );
    },
    error: function(xhr, status, err){
    }
  });
});
```



# やってみよう

- 教員名リストの取得

```
$(function(){
  $.ajax({
    url: "fms.xml",
    dataType: "xml",
    success: function(data){
      $(data).find( "teacher" ).each(
        function(){
          console.log( $(this).find("name").text() );
        });
    },
    error: function(xhr, status, err){
    }
  });
});
```



# 演習

- 教員名リストを作成しよう
- 講義名リストを作成しよう(重複してよい)
- 教員名一覧の各教員名にはリンクURLを設定しクリックするとそのページに飛ぶようにせよ

# XSS クロスサイトスクリプティング

- XML を読み込みたいんだけど許可されていないって出てくるのは何故？

```
✖ XMLHttpRequest cannot load http://api.gnavi.co.jp/ver1/RestSearchAPI/?  
keyid=6506d35c800a5506879c7d0632d7f1a9&latitude=35.7076195&longitude=139.6594432&range=2. No 'Access-Control-Allow-Origin'  
header is present on the requested resource. Origin 'http://cmp2-150773.apne1.nitrousbox.com' is therefore not allowed  
access. gnavi.html:1
```

- Ajax は基本的に他のサーバから情報を持って  
くることが出来ない！（自サーバからもってくる）
  - 悪意のあるコードを埋め込まれると困るため
  - JSONPというのは、アクセス可能とする仕組み



# ただ、方法はある

- PHPなどサーバのシステムからは情報を取得することが可能なので、PHPを経由する
- 下記のコードをコピーして利用しよう
  - [http://nkmr.io/lecture/2014/get\\_xml.php.txt](http://nkmr.io/lecture/2014/get_xml.php.txt)
  - 利用方法

```
$.ajax({  
  url: "get_xml.php?url=" + "取得対象のURL",  
  dataType: "xml",  
  success: function(data){  
    // 成功した時の処理 data は取得したデータ  
  },  
  error: function(xhr, status, err){  
  }  
});
```

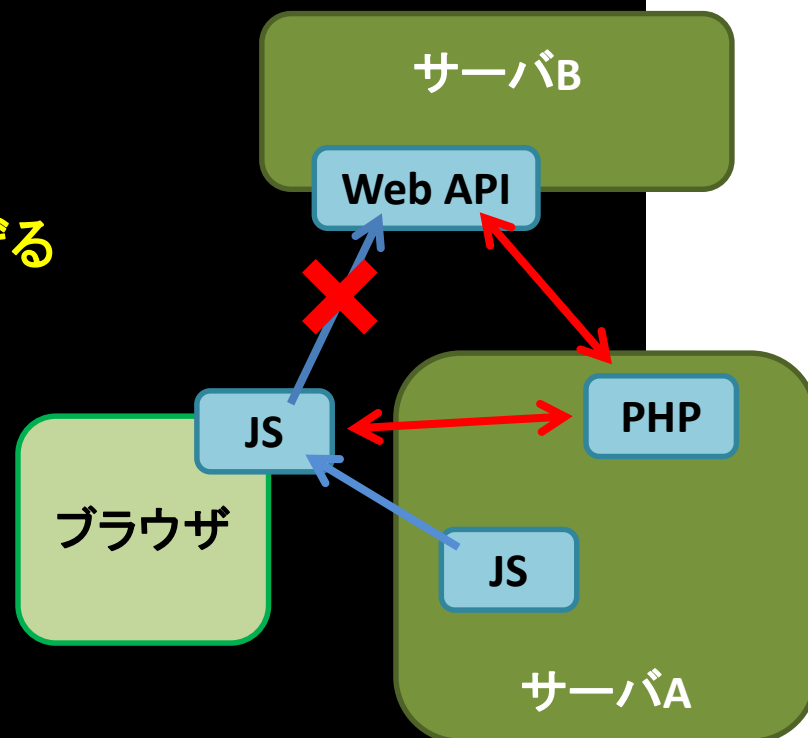




# ただ、方法はある

- PHP経由で情報取得

```
<?php
if(isset($_GET["url"]) && preg_match("/^https?:/", $_GET["url"])){
    header('Content-type: application/xml');
    $req_url = $_GET['url'];
    foreach ( $_GET as $key => $value){
        if( strcmp( $key, "url" ) ){
            // url でない場合はオプションをつなげる
            $req_url .= ("&" . $key . "=" . $value);
        }
    }
    echo file_get_contents($req_url);
} else {
    echo "error";
}
?>
```





# 次週までの課題

- 課題5-1 (a)
  - JavaScript (jQuery)+現在位置+地図+何らかのWeb APIを使ったアプリケーションを作る
- 課題5-1 (b)
  - JavaScript (jQuery)+Web APIで面白いアプリケーションを作る
  - (a), (b)はどちらか取り組む
- 課題5-2
  - FMSにまつわる何かのデータを格納するお役立ちXMLファイルを作成せよ. また, そのXMLファイルから状況に応じてデータを読み込み結果を出力せよ(ユーザの操作で少なくとも2つ以上の振る舞いをするようにせよ)



# 課題の提出方法

- すべて nitrous.io 上で動作するようにすること
  - 第5回課題として、ウェブページを作成し、そこから説明を付与して課題へのリンクを作成すること
  - nitrous.ioの課題のトップページのURLをテキストに保存して提出

## コンテンツプログラミング実習 第5回課題成果

### 課題5-1 (a)

JavaScript (jQuery)+現在位置+地図+何らかのWeb APIを使ったアプリケーションを作る

- 地図とレストランサイトと天気予報を使った超絶便利なサイト！

### 課題5-2

FMSIにまつわる何かのデータを格納するXMLファイルを作成せよ。また、そのXMLファイルから状況に応じてデータを読み込み結果を出力せよ

- 作成したXML(シラバスのXML)
- 作成したウェブページ(受講に関するお手軽サイト)