AJAX

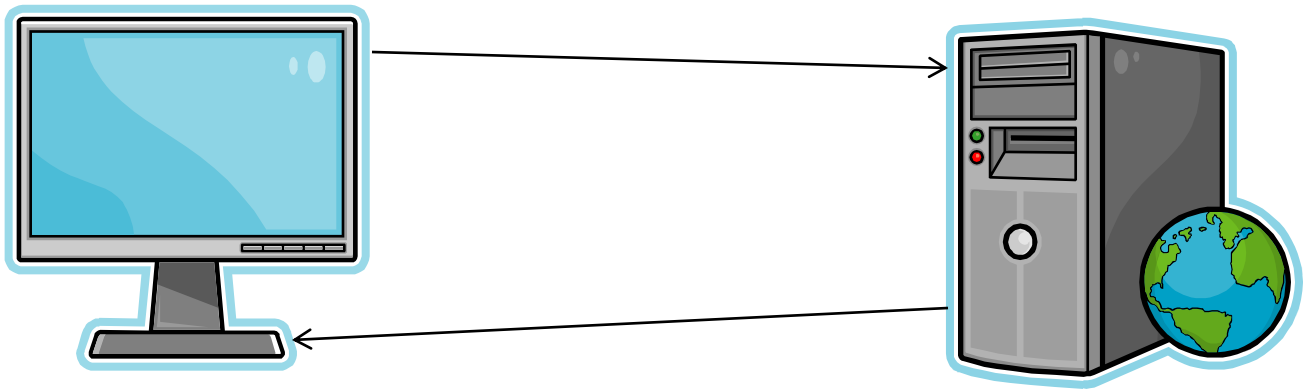
IF3038 Pemrograman Internet

Semester 2/2009-2010

AJAX

Asynchronous Javascript and XML

- Asynchronous: Client dapat beroperasi tanpa harus menunggu hasil lengkap dari server
- interaksi synchronous: client menunggu server selesai



AJAX

diperkenalkan oleh Jesse James Garrett, 2005 (**Ajax: A New Approach to Web Applications)**
http://www.adaptivepath.com/ideas/essays/archives/000385.php

terdiri atas elemen:

- Asynchronous: XMLHttpRequest / XMLHTTP
- Javascript
- XML & XHTML

XMLHttpRequest

perintah utama:

- open(method, url [, async, username, passwd)
- setRequestHeader(label, value)
- send(content)
- getAllResponseHeader( )
- getResponseHeader(label)
- abort

properties

- onreadystatechange
- readyState: 0 (uninitialized), 1 (open request), 2 (request sent), 3 (response received), 4 (finish loading)
- responseText, responseXML
- status, statusText

Menggunakan XMLHttpRequest

Masalah kompatibilitas

- browser IE lama menggunakan objek dengan nama berbeda: MSXML2.XMLHttp, MSXML2.XMLHttp.3.0, MSXML2.XMLHttp.4.0

solusi: membuat fungsi untuk mengambil objek XmlHttpRequest

```
function getXmlHttpRequest( ) {

  var xmlHttpObj;

  if (window.XMLHttpRequest) {
    xmlHttpObj = new XMLHttpRequest( );
  } else {
    try {
      xmlHttpObj = new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e) {
      try {
        xmlHttpObj = new ActiveXObject("Microsoft.XMLHTTP");
      } catch (e) {
        xmlHttpObj = false;
      }
    }
  }
  return xmlHttpObj;
}
```

Mengirimkan perintah

open: menentukan http method (GET/POST)

parameter:

- untuk POST, dikirim sebagai xmlhttp.send(parameter)
- untuk POST, harus mendefinisikan content-type (untuk POST)

```
xmlhttp.setRequestHeader('Content-Type','application/x-www-form-urlencoded');
```

- untuk GET, parameter dikirim sebagai bagian dari url

```
http://somewebserver.com/someapp.php?param=value&param2=value2
```

```
function getRecipe( ) {
  if (!xmlhttp)
    xmlhttp = getXmlHttpRequest( );
  if (!xmlhttp)
    return;
  var drink =
    encodeURIComponent(document.getElementById('drink').value);
  var qry = 'drink=' + drink;
  var url = 'drink.php?' + qry;
  xmlhttp.open('GET', url, true);
  xmlhttp.onreadystatechange = printRecipe;
  xmlhttp.send(null);
}
```

```
function getRecipe( ) {
  if (!xmlhttp)
    return;
  var drink = encodeURIComponent(
    document.getElementById('drink').value);
  var qry = 'drink=' + drink;
  var url = 'drink.php';
  xmlhttp.open('POST', url, true);
  xmlhttp.onreadystatechange = printRecipe;
  xmlhttp.setRequestHeader(
    'Content-Type', 'application/x-www-form-urlencoded');
  xmlhttp.send(qry);
}
```

menangani hasil

Perintah send berjalan asynchronous, saat hasil diperoleh, fungsi yang didefinisikan pada properties onreadystatechange akan dijalankan

contoh:

```
xmlhttp.onreadystatechange = printRecipe;
```

```
function printRecipe( ) {
  if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
    alert(xmlhttp.getAllResponseHeaders( ));
    alert(xmlhttp.responseText);
  }
}
```

menampilkan hasil

saat response telah diterima lengkap, hasil dapat ditambahkan ke page secara dinamis, misalnya dengan membuat elemen HTML baru untuk menampilkan hasil

alternatif: membuat elemen HTML khusus sebelumnya untuk menampilkan hasil

```
function printRecipe( ) {
  if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
    var body = document.getElementsByTagName('body');

    if (document.getElementById('recipe')) {
    body[0].removeChild(
      document.getElementById('recipe'));
    }

    var recipe = document.createElement('div');
    recipe.id = 'recipe';
    recipe.className = 'recipe';
    recipe.innerHTML = xmlhttp.responseText;
    body[0].appendChild(recipe);
  }
}
```

XMLHttpRequest properties

- responseText: data hasil server dalam format text
- responseXML: data hasil server dalam bentuk DOM
- status: status code response (e.g. 200, 404)
- statusText: teks yang dikirim bersama status code

XMLHttpRequest method

- abort: membatalkan request
- getAllResponseHeaders(): mengembalikan semua header response berupa string
- getResponseHeader(header): mengembalikan nilai header tertentu
- open(method, URL [, async flag [, username [,password]]])
- send(content)
- setRequestHeader(header, value)

contoh

```
if (request) {
  request.open('GET', URL, true);
  request.onreadystatechange = parseResponse;
  request.send('');
}
```

Format parameter: `param1=data1&param2=data2`

contoh

```
function parseResponse( ) {
  /* Is the /readyState/ 4? */
  if (request.readyState == 4) {
  /* Is the /status/ 200? */
    if (request.status == 200) {
    /* Grab the /responseText/ from the request */
      var response = request.responseText;
      alert(response);
      // here is where the parsing would begin.
    } else
       alert('There was a problem retrieving the data: \n'
       + request.statusText);
    request = null;
  }
}
```

Menggunakan XML untuk data

Contoh sebelumnya: data dikirim dengan format HTML

- jika server mengirimkan data dengan format salah, seluruh tampilan akan rusak
- client dapat mengolah dan mengatur hasil secara fleksibel

Menggunakan XML untuk data

Pada javascript, parsing XML dari server dapat menggunakan method getElementById() dan getElementsByTagName()

```
<parameters>
  <param id="1">data1</param>
  <param id="2">data2</param>
  <param id="3">data3</param>
</parameters>
```


```
function parseResponse( ) {
  if (request.readyState == 4) {
    if (request.status == 200) {
      var response = request.responseXML;
      var paramList = response.getElementsByTagName('param');
      var out = '<ul>';
      for (i = 0, il = paramList.length; i < il;)
        out += '<li>' + paramList[i++].firstChild.nodeValue +
          '</li>';
      out += '</ul>';
      document.getElementById('list').innerHTML = out;
    } else alert('There was a problem retrieving the data: \n'
   + request.statusText); request = null;
  }
}
```

Server XML processing

Client dapat mengirimkan data/parameter sebagai XML (menggunakan POST method)

Parsing XML pada PHP dapat dilakukan menggunakan method simplexml_load_string()

```
$raw_xml = file_get_contents("php://input");
$data = simplexml_load_string($raw_xml);
foreach ($data->param as $param)
   switch ($param['id']) {
     case 1: $value1 = $param; break;
     case 2: $value2 = $param; break;
     case 3: $value3 = $param; break; }
```

```
<?php
  $drink = $_REQUEST['drink'];
  if(empty($drink)) {
    echo "<title>No drink name sent</title>";
  } else {
    $search = trim($drink);
    switch($search) {
      case "TEA" : $result = "<title>Hot Tea</title>" .
"<ingredient>tea leaves</ingredient>" . "<instruction>Boil
water. Pour over tea leaves. " . "Steep five minutes.
Strain and serve.</instruction>";
      break;

...
```

```
function printRecipe( ) {
  if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
    var body = document.getElementsByTagName('body');
    if (document.getElementById('recipe')) {
      body[0].removeChild(document.getElementById('recipe'));
    }
    var recipe = document.createElement('div');
    recipe.id = 'recipe';
    recipe.className='recipe';
    var title =   xmlhttp.responseXML.
      getElementsByTagName('title')[0].firstChild.nodeValue;
    var titleNode = document.createElement('h3');
    titleNode.appendChild(document.createTextNode(title));
    recipe.appendChild(titleNode);
    var ul = document.createElement("ul");
    var ingredients =
      xmlhttp.responseXML.getElementsByTagName('ingredient');
```

```
    for (var i = 0; i < ingredients.length; i++) {
      var x = document.createElement('li');
      x.appendChild(document.createTextNode(ingredients[i].
        firstChild.nodeValue));
      ul.appendChild(x);
    }
    recipe.appendChild(ul);
    var instr = xmlhttp.responseXML.
      getElementsByTagName('instruction')[0].firstChild.nodeValue;
    var instrNode = document.createElement('p');
    instrNode.appendChild(document.createTextNode(instr));
    recipe.appendChild(instrNode);
    body[0].appendChild(recipe);
  }
}
```

Menggunakan JSON untuk data

format XML memiliki kelemahan: kompleksitas pengolahan

JSON: JavaScript Object Notation

- menggunakan tipe Javascript sebagai format pertukaran data
- lebih sederhana, karena data langsung dikirim berupa tipe javascript

format:

- object { *keyword* : *value*}

proses:

eval(xmlhttp.responseText);

JSON.parse () -> menggunakan library JSON

JSON

- JSON memiliki 2 struktur dasar
  - Object: menggunakan { }
  - Array: menggunakan [ ]

```
{'parameters': {
   'param': [
     {'id': 1, 'value': 'data1'},
     {'id': 2, 'value': 'data2'},
     {'id': 3, 'value': 'data3'}
] } }
```

```
{books:[
  {title:"AJAX and PHP: Building Modern Web Applications, 2nd Ed",
    isbn:"978-1904817726"},
  {title:"Beginning PHP and MySQL E-Commerce, 2nd Edition",
    isbn:"978-1590598641"},
  {title:"Professional Search Engine Optimization with PHP",
    isbn:"978-0470100929"}
]}
```

Penanganan PHP

PHP menyediakan JSON library:

- json_decode() : mengubah json menjadi PHP object
- json_encode() : mengubah PHP object menjadi JSON

Pengolahan JSON dapat pula dilakukan manual, menggunakan notasi [ ] dan { } langsung

Penanganan PHP

```
$raw_json = file_get_contents("php://input");
$data = json_decode($raw_json);
for ($i = 0, $il = count($data['parameters']['param']);
  $i < $il;) {
  $d = $data['parameters']['param'][$i++];
  switch ($d['id']) {
    case 1: $value1 = $d['value']; break;
    case 2: $value2 = $d['value']; break;
    case 3: $value3 = $d['value'];
  }
}
```

Penanganan PHP

```
$sql = 'SELECT * FROM table1 WHERE condition1 = $value1
   AND condition2 = $value2' .' AND condition3 = $value3';
$result = $conn->query($sql);
if ($rows = $result->fetchAll( )) {
  $value = array( );
  $value['results'] = array( );
   $value['results']['result'] = array( );
  foreach($rows in $row)
    $value['results']['result'][$i] =
      array('column1' => $row['column1'],
            'column2' => $row['column2']);
  $output = json_encode($value);
}
```

```
<?php
  $drink = $_REQUEST['drink'];
  if(empty($drink)) {
    echo "{'title': 'No drink name sent'}";
  } else {
    $search = trim($drink);
    switch($search) {
      case "TEA" : $result = "{ 'title' : 'Hot Tea'," .
       " 'ingredients' : [ { 'ingredient' : 'tea leaves' },".
       " {'ingredient' : 'water'}]," .
       " 'instruction' : 'Boil water. Pour over tea leaves." .
       " Steep five minutes. Strain and serve.'}";
      break;
```

```
function printRecipe( ) {
  if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
    var body = document.getElementsByTagName('body');
    if (document.getElementById('recipe')) {
     body[0].removeChild(
       document.getElementById('recipe'));
    }
    var recipe = document.createElement('div');
    recipe.id = 'recipe';
    recipe.className='recipe';
    var recipeObj = eval("(" + xmlhttp.responseText + ")");
    var title = recipeObj['title'];
    var titleNode = document.createElement('div');
    titleNode.className='title';
    titleNode.appendChild(document.createTextNode(title));
    recipe.appendChild(titleNode);
    var ingredients = recipeObj.ingredients;
```

```
    for (var i = 0; i < ingredients.length; i++) {
      var x = document.createElement('div');
      x.className = 'ingredient';
      x.appendChild(
        document.createTextNode(ingredients[i].ingredient));
      recipe.appendChild(x);
    }
    var instr = recipeObj.instruction;
    var instrNode = document.createElement('div');
    instrNode.className='instruction';
    instrNode.appendChild(document.createTextNode(instr));
    recipe.appendChild(instrNode);
    body[0].appendChild(recipe);
}
```

Parsing JSON

Pada Javascript, menggunakan eval() memiliki resiko dapat mengeksekusi kode javascript apa pun

Json.org menyediakan library untuk parsing JSON:

http://www.json.org/json2.js

JSON.parse(json document);

eval() memiliki kelebihan: cepat

JSON vs XML

Parsing JSON pada javascript lebih cepat

Parsing JSON menggunakan eval() memiliki resiko keamanan

Ukuran data JSON lebih kecil

XML dapat langsung diproses browser, atau diubah ke format HTML

XML dapat menangani data biner

AJAX tools

Beberapa library AJAX yang populer

- prototype
- script.aculo.us
- jQuery
- rico
- Dojo

prototype

Website: http://www.prototypejs.org

Fitur:

- memudahkan akses DOM via $(), $F() etc
- Ajax object
- Element object

Penggunaan:

```
<script type="text/javascript"
    src="/path/to/prototype.js"></script>
```

prototype

menyederhanakan akses elemen html

elemen HTML diakses dengan sintaks $( )

- HTML: ...
- diakses dengan: var elem = $('*some_element*');
- cara biasa: var elem = document.getElementById('*some_element*');
- $(‘id1’, ‘id2’, ‘id3’) akan mengembalikan array yang berisi elemen dengan id1, id2 dan id3

nilai atribut form diakses dengan $F( )

- HTML: <input type="text" id="*form_field*" />
- diakses dengan: var value = $F('*form_field*');
- cara biasa: var value = document.getElementById('*form_field*').value

prototype

Prototype meng-enhance document object dengan menyediakan document.getElementsByClassName()

Contoh:

untuk mengambil semua elemen dengan class='myclass'

```
var myEl = document.getElementsByClassName('myclass');
```

Untuk mengambil semua elemen dengan class='myclass' pada elemen dengan id='myid'

```
var myEl = document.getElementsByClassName('myclass', $('myid'));
```

$$() mengembalikan elemen menggunakan notasi selector CSS

$$('#menuitem div'): mengembalikan elemen div yang berada di dalam menuitem

prototype

menyediakan objek pembungkus XMLHttpRequest, yaitu Ajax

var myAjax = new Ajax.Request(url, { method: 'get', parameters: qry, onComplete: printRecipe });

printRecipe hanya dipanggil saat response selesai, bukan pada setiap perubahan state

constructor Ajax.Request menerima parameter berupa url dan sebuah object yang dapat berisi atribut-atribut untuk request

prototype

Atribut object pada Ajax.Request constructor:

- parameters: parameter untuk query (URL encoded string)
- method: post atau get
- asynchronous: true atau false
- requestHeaders: array yang berisi pasangan nama value untuk request header
- postBody: isi body jika request menggunakan post
- onInteractive, onLoad, onComplete: callback method yang dipanggil saat terjadi perubahan state pada request
- on404, onXXX: callback method yang dipanggil server mengembalikan status kode tertentu
- onSuccess: callback method jika request sukses dieksekusi
- onFailure, onException: callback method jika terjadi kegagalan

```
/* Create an Ajax call to the server */
new Ajax.Request(URL, {
    method: 'post',
    parameters: 'param1=data1&param2=data2&param3=data3',
    onSuccess: parseResponse,
    onFailure: function(xhrResponse) {
        alert('There was a problem retrieving the data: \n' +
           xhrResponse.statusText);
    }
});
var parseResponse = function(xhrResponse) {
    var response = xhrResponse.responseXML;
    var paramList = response.getElementsByTagName('param');
    var out = '<ul>';

    /* Loop through the /param/ elements in the response to create the list
   items */
    for (i = 0, il = paramList.length; i < il;) {
        out += '<li>' + paramList[i++].firstChild.nodeValue + '</li>';
    }
    out += '</ul>';
    $('list').innerHTML = out;
}
```

Prototype

Menyediakan objek Ajax.Updater yang dapat memanggil Ajax request dan mengupdate elemen tertentu

```
new Ajax.Updater('myDiv', URL, {
  method: 'get', parameters:'param=data1'} );
```

script.aculo.us

menggunakan prototype

menyediakan fasilitas berbagai efek untuk tampilan elemen HTML

fungsionalitas yang disediakan:

- builder
- effects
- dragdrop
- controls
- slider

script.aculo.us effect

Nama	Keterangan
Effect.Appear	menampilkan elemen
Effect.Fade	menghapus elemen dengan pudar
Effect.BlindDown	scroll elemen ke bawah
Effect.BlindUp	scroll elemen ke atas
Effect.SwitchOff	menghapus elemen
Effect.SlideDown	menggeser elemen ke bawah
Effect.SlideUp	menggeser elemen ke atas
Effect.DropOut	drop elemen
Effect.Shake	goncang/goyang
Effect.Grow	membesar
Effect.Shrink	mengecil

penggunaan script.aculo.us

```
<script src="javascripts/prototype.js" type="text/javascript"></script>
    <script src="javascripts/scriptaculous.js"
    type="text/javascript"></script>
```

```
new Effect.<nama efek> ( <nama elemen> );
```

```
new Effect.Shake( recipe );
```

jQuery

Library Javascript, website: http://www.jquery.com
Memiliki ukuran kecil (15K)

Fitur

- Memudahkan akses DOM $()
- Ajax

Penggunaan

```
<script type="text/javascript" src="jquery.js"></script>
```

jQuery: $()

$(): mengembalikan elemen atau array elemen, dengan menggunakan string selector dengan format CSS

$(‘a’): mengambil elemen-elemen dengan tag a

$(‘#myid’): mengambil elemen dengan id myid

jQuery: $.get()

$.get(): memanggil Ajax request dengan get

```
$.get("path/to/data.php?name=value",
      function (sData, sStatus) {
         alert(sStatus + ":" + sData);
      });
$.get("path/to/data.php", { name: "value" },
      function (sData, sStatus) {
          alert(sStatus + ":" + sData);
});
```

Callback parameter: Data & Status

Varian:

- `$.getIfModified()`, `$.getJSON()`, `$.getScript()`

```
<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Transitional//EN”
“http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd”>
<html xmlns=http://www.w3.org/1999/xhtml>  <head>
<title>jQuery GET Example</title>
<script type=”text/javascript”src=”jquery.js”></script>
<script type=”text/javascript”>
//<![CDATA[
function requestCustomerInfo() {
var sId = $(“input#txtCustomerId”).val();
$.get(“GetCustomerData.php?id=” + sId, displayCustomerInfo);
}
function displayCustomerInfo(sText, sStatus) {
if (sStatus == “success”) {
$(“div#divCustomerInfo”).html(sText);
} else { $(“div#divCustomerInfo”).html(“An error occurred.”); }
}
//]]> </script>
</head>
<body>
<p>Enter customer ID number to retrieve information:</p>
<p>Customer ID: <input type=”text” id=”txtCustomerId” value=”” /></p>
<p><input type=”button” value=”Get Customer Info”
onclick=”requestCustomerInfo()” /></p>
<div id=”divCustomerInfo”></div>
</body>
</html>
```

jQuery: $.post()

$.post() serupa dengan $.get()

```
$.post("path/to/data.php", { name: "value" },
  function (sData, sStatus) {
alert(sStatus + ":" + sData);
});
```

jQuery: load()

Serupa dengan Ajax.updater() pada prototype, melakukan Ajax request dan mengganti elemen DOM dengan hasilnya

```
var sId = $("input#txtCustomerId").val();
$("div#divCustomerInfo").load("GetCustomerData.php?id="
  + sId);
```

Contoh ini mengambil id dari elemen input dengan id txtCustomerId, memanggil Ajax untuk mendapatkan data customer, dan hasilnya disimpan pada elemen div dengan id divCustomerInfo

Pemanggilan load dengan 2 parameter menghasilkan eksekusi post

jQuery: $.ajax()

$.ajax(): low level method untuk Ajax request

Menerima 1 parameter berupa associative array

- type: get atau post
- url: url request
- data: url encoded data
- dataType: tipe data response: script, xml, html, json
- success(): callback untuk success
- Error(): callback error
- Complete(): callback untuk complete

```
$.ajax({
   type : “GET”,
   url : “GetCustomerData.php?id=” + sId,
   success : function (oXHR, status) {
    $(“div#divCustomerInfo”).html(oXHR.responseText);
    },
   error : function (oXHR, status) {
    $(“div#divCustomerInfo”).html(“An error occurred.”);
    }
   });
```

jQuery: ajax event

jQuery menyediakan fasilitas event handler untuk ajax request: ajaxStart() dan ajaxStop()

Jika pada sebuah elemen method ini didefinisikan, ajaxStart() akan dipanggil saat ajax request mulai dijalankan, dan ajaxStop dipanggil saat request selesai dieksekusi

```
$("div#divStatus").ajaxStart(function () {
   $(this).html("Contacting the server...");
}).ajaxStop(function () {
   $(this).html("Response received.");
});
```

Elemen div dengan id divStatus akan menampilkan: “Contacting the server...”, saat ajax request mulai dieksekusi, dan “Response received.” saat ajax request selesai dieksekusi

rico

menggunakan prototype

menyediakan fasilitas efek

fungsionalitas LiveGrid: halaman dapat dihubungkan ke sebuah sumber data, dan setiap ada data baru akan ditampilkan secara scroll ke bawah

- dinamakan juga live scrolling atau ajax pagination
- contoh: untuk menampilkan informasi cuaca, google map

dojo

kumpulan berbagai library javascript

- pengolahan string
- manipulasi DOM
- manipulasi CSS
- HTML
- dictionary
- animasi
- kriptografi
- library javascript umum

Penerapan AJAX

JIT Help system: menampilkan informasi/help saat pengisian elemen form

tooltips

live preview: menampilkan preview hasil editing

accordion: menampilkan informasi secara selektif

tabbed pages

overlays

Autocompletion

Menampilkan data tabular

Ajax Pattern

Predictive Fetch: Ajax app guesses what the user is going to do next, and retrieves the appropriate data

- Next page (tabular data, multipage article)
- Address book pada email, suggestion
- Gunakan timeout untuk meload data berdasarkan predictive fetch (misal: current page = 1, maka setelah 5 detik, panggil ajax request untuk me-load 2, dan setelah 5 detik lagi, load 3 dst

Ajax Pattern

Submission Throttling: untuk pengiriman data ke server dalam jumlah besar, sehingga dilakukan secara bertahap

- Saat entri data, data di buffer di client terlebih dahulu, kemudian data dikirimkan pada periode waktu tertentu
- Contoh: validasi form via Ajax: validasi request tidak dilakukan setiap entri field, tapi berdasarkan periode tertentu

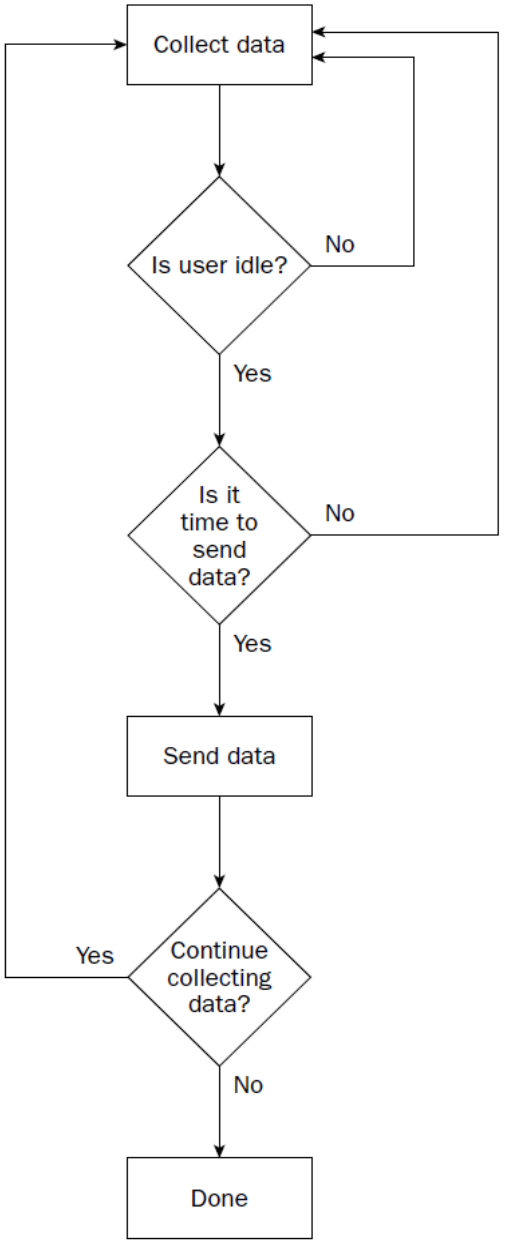


Figure 3-2

Ajax Pattern

Periodic Refresh: Secara periodic dilakukan ajax request untuk memeriksa apakah ada data baru

- Mencek email baru
- Mencek status/komentar/posting baru pada social networking

Multistage download: halaman tidak di-load sekaligus, namun bertahap.

- Hasil pada Google search
- Web portal

Ajax Pattern

Cancel Pending Request: membatalkan request yang mungkin akan dieksekusi

- Jika status result 404, tidak perlu lagi mencek server untuk memeriksa email baru

Priority Queue: HTTP hanya membolehkan 1 request aktif ke server domain yang sama. Client harus melakukan request queuing jika ingin mengirimkan beberapa request sekaligus