Aplicación al Web-IO digital:

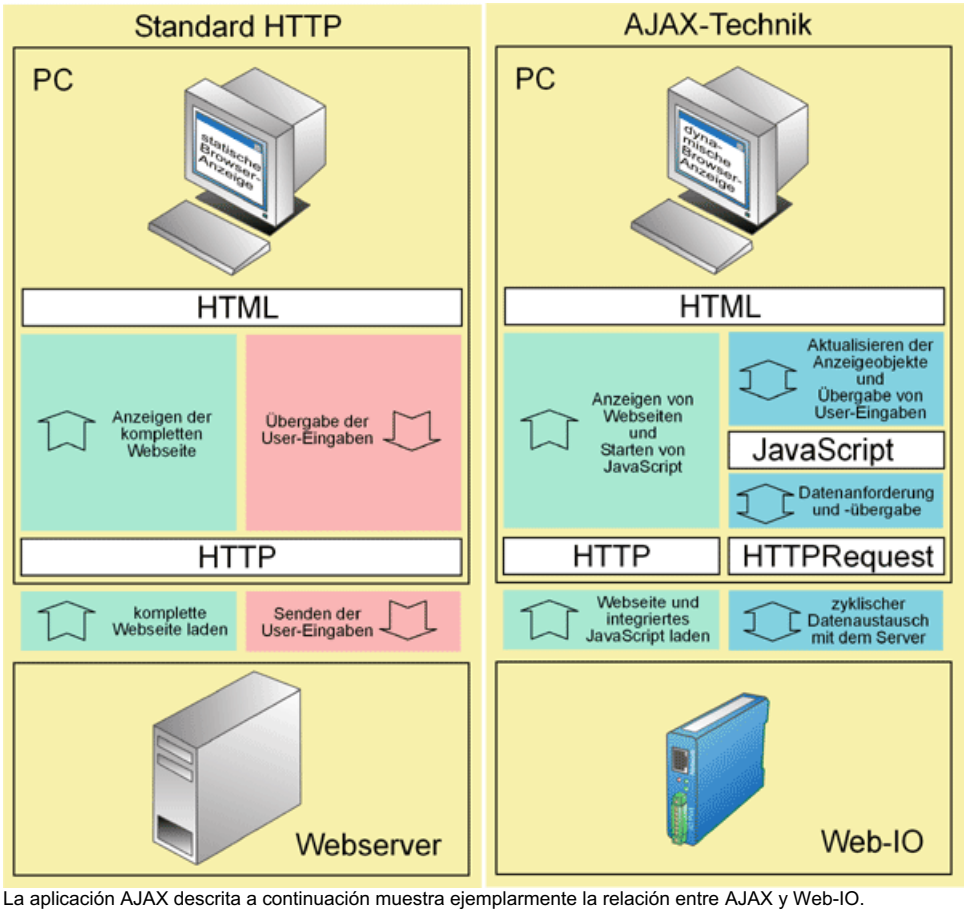
Web-IO digital - visualizar con AJAX en el Browser

Resumen de productos
Sinopsis de aplicaciones

El Browser de Internet es hoy día parte componente de todos los modernos sistemas operativos. Tanto si se trata de Internet Explorer, Firefox, Opera, Netscape o Safari - al navegar por Internet se estima el Browser como un instrumento versátil de indicación.

Con AJAX y los W&T Web-IOs el Browser se puede utilizar ahora también como elemento de indicación y de mando para aplicaciones dinámicas, técnicas.

AJAX significa Asynchronus JavaScript and XML, siendo la funcionalidad clave de AJAX seguir comunicando con el servidor después de cargar una página web en el Browser. Páginas web, que se forman en el estándar HTML, para la actualización sólo pueden cargarse completamente de nuevo. JavaScripte basadas en AJAX por el contrario pueden cambiar o modificar posteriormente elementos separador de indicación.

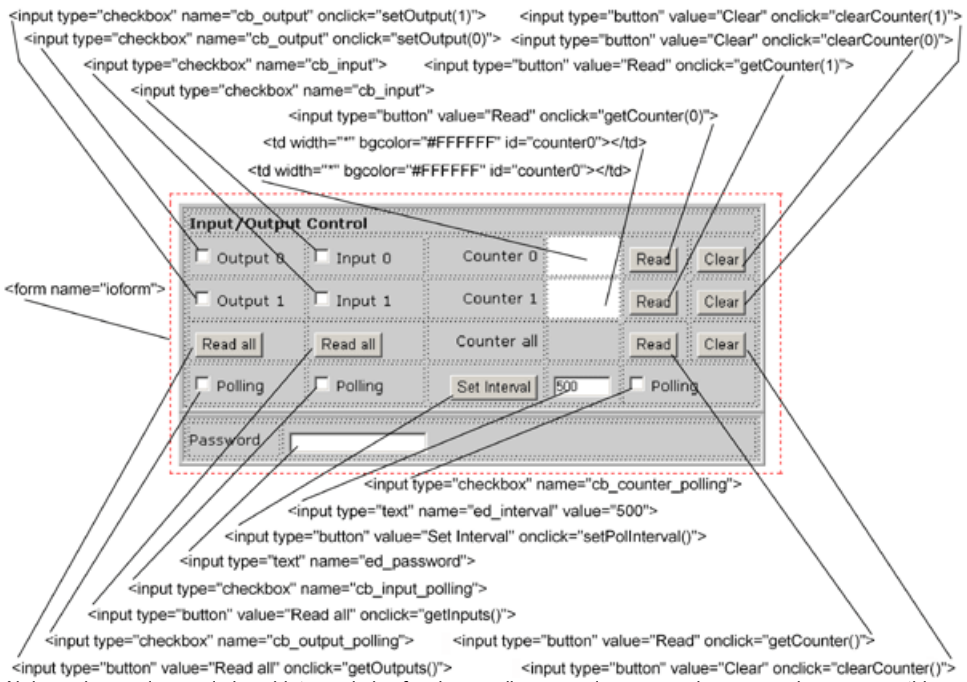


La aplicación AJAX descrita a continuación muestra ejemplarmente la relación entre AJAX y Web-IO.

Preparativos

- con corriente,
- entradas y salidas conectadas
- conectado a su red
- dotado con una dirección IP - conWuTility no hay problemas
- El Web-IO vía Web-based Management para el servicio AJAX configura (activar HTTP-Header)

Recopilación de los diferentes elementos de manejo y objetos de visualización de la página web



Al denominar cada uno de los objetos o de las funciones a llamar es de gran ayuda usar nombres con sentido.

1. Estructura base HTML de la página web Colocar los elementos de mando y de indicación
Para una mejor estructuración se mete cada uno de los elementos en tablas y se declara todo como formulario. El Tag <user.htm> no pertenece a la sintaxis estándar HTML y sirve al Web-IO para identificar la página web. El Tag se filtra antes de cargar en el Browser.

```
<user.htm>
 <html>
  <head>
   <title>Web-IO AJAX-Client</title>
   <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   <STYLE type=text/css>
    TD {
    COLOR: #000000;
    FONT-FAMILY: Verdana,Arial,Helvetica;
    FONT-SIZE: 10pt;
     }
   </STYLE>
  </head>
  <body bgcolor="#FFFFFF" text="#000000" link="#000000">
   <form name="ioform">
    <table width="500" border="1" height="144" bgcolor="#CCCCCC">
     <tr>
      <td>
       <table width="500">
        <tr>
         <td colspan="6">
          <b>Input/Output Control </b>
         </td>
        </tr>
        <tr>
         <td width="100">
          <input type="checkbox" name="cb_output" onclick="setOutput(0)">Output 0
         </td>
         <td width="100">
          <input type="checkbox" name="cb_input">Input 0
         </td>
         <td width="100">
          <div align="right">Counter 0 </div>
         </td>
         <td width="*" bgcolor="#FFFFFF" id="counter0">
         </td>
         <td width="55">
          <input type="button" value="Read" onclick="getCounter(0)">
         </td>
         <td width="55">
          <input type="button" value="Clear" onclick="clearCounter(0)">
         </td>
        </tr>
        <tr>
         <td width="100">
```

```
        <input type="checkbox" name="cb_output" onclick="setOutput(1)"> Output 1
       </td>
       <td width="100">
        <input type="checkbox" name="cb_input"> Input 1
       </td>
       <td width="100">
        <div align="right">Counter 1 </div>
       </td>
       <td width="*" bgcolor="#FFFFFF" id="counter1">
       </td>
       <td width="55">
        <input type="button" value="Read" onclick="getCounter(1)">
       </td>
       <td width="55">
        <input type="button" value="Clear" onclick="clearCounter(1)">
       </td>
      </tr>
      <tr>
       <td width="100" height="29">
        <input type="button" value="Read all" onclick="getOutputs()">
       </td>
       <td width="100" height="29">
        <input type="button" value="Read all" onclick="getInputs()">
       </td>
       <td width="100" height="29">
        <div align="right">Counter all </div>
       </td>
       <td width="*" height="29">
       </td>
       <td width="55" height="29">
        <input type="button" value="Read" onclick="getCounter()">
       </td>
       <td width="55" height="29">
        <input type="button" value="Clear" onclick="clearCounter()">
       </td>
      </tr>
      <tr>
       <td width="100">
        <input type="checkbox" name="cb_output_polling">Polling
       </td>
       <td width="100">
        <input type="checkbox" name="cb_input_polling"> Polling
       </td>
       <td width="100">
        <div align="right">
        <input type="button" value="Set Interval" onclick="setPolInterval()">
        </div>
       </td>
       <td width="*" >
        <input type="text" name="ed_interval" value="500" maxlength="6" size="9">
       </td>
       <td colspan="2">
        <input type="checkbox" name="cb_counter_polling"> Polling
       </td>
      </tr>
     </table>
    </td>
   </tr>
  </table>
  <table width="500" border="1" bgcolor="#CCCCCC" >
   <tr>
    <td height="35">
     <table width="500">
      <tr>
       <td width="78">Password</td>
       <td width="410">
        <input type="text" name="ed_password">
       </td>
      </tr>
     </table>
    </td>
   </tr>
  </table>
 </form>
```

```
 ....
```

2. Declaraciones globales JavaScript Variables y funciones generales
Aunque la página web está creada para el Web-IO 2xEntradas Digitales, 2xSalidas Digitales, las JavaScripte están preparadas para Web-IOs con más IOs. Por ello también la función HexToInt, que convierte los Strings hexadecimales en números enteros.

La variable SendString se utilizará posteriormente para el envío de datos al Web-IO.

```
...
var MAXIO=2;
var SendString;

function HexToInt(HexStr)
 {
 var TempVal;
 var HexVal=0;
 for( i=0; i<HexStr.length;i++)
  {
  if (HexStr.charCodeAt(i) > 57)
   {
   TempVal = HexStr.charCodeAt(i) - 55;
   }
  else
   {
   TempVal = HexStr.charCodeAt(i) - 48;
   }
  HexVal=HexVal+TempVal*Math.pow(16, HexStr.length-i-1);
  }
 return HexVal;
 }
```

3. Procesar el manejo por el usuario
Dependiendo de qué elemento de manejo de la página web ha chasqueado o cambiado el usuario, la variable SendString se llena con el comando correspondiente.

Poner Outputs
El usuario puede poner los Outputs a través de dos casillas de verificación cb_output. Al llamar la función se entrega el n° de los Outputs.

La función DataRequest que se llama a continuación, sirve para el intercambio de datos con el Web-IO y está descrita más detalladamente en el transcurso.

```
   function setOutput(OutputNr)
    {
    if (ioform.cb_output[OutputNr].checked==true)
     {
     SendString='outputaccess'+OutputNr+'?PW='+ioform.ed_password.value+'&State=ON&';
     }
    else
     {
     SendString='outputaccess'+OutputNr+'?PW='+ioform.ed_password.value+'&State=OFF&';
     }
    DataRequest(SendString);
    }
```

Solicitar estado de Output/Input
El usuario puede solicitar el estado de los Outputs e Inputs chasqueando el botón correspondiente.

```
   function getOutputs()
    {
    DataRequest('output?PW='+ioform.ed_password.value+'&');
    }

   function getInputs()
    {
    DataRequest('Input?PW='+ioform.ed_password.value+'&');
    }
```

Preguntar/borrar contadores
También se pueden preguntar o borrar los estados de contador de los contadores Input. Como parámetro se entrega el n° del contador que se debe leer o borrar. Si no se entrega ningún parámetro, el Web-IO lee o borra todos los contadores.

```
   function getCounter(CounterNr)
    {
    if (CounterNr==undefined)
     {
     DataRequest('counter?PW='+ioform.ed_password.value+'&');
     }
     else
     {
     DataRequest('counter'+CounterNr+'?PW='+ioform.ed_password.value+'&');
     }
    }

   function clearCounter(CounterNr)
    {
    if (CounterNr==undefined)
     {
     DataRequest('counterclear?PW='+ioform.ed_password.value+'&');
     }
    else
     {
     DataRequest('counterclear'+CounterNr+'?PW='+ioform.ed_password.value+'&');
     }
    }
```

4. Comunicación con el Web-IO
Intercambio de datos con el Web-IO y actualización de la página web después de que ya se cargó ésta.

- La función aquí presentada contiene todo lo que constituye AJAX.
- La función DataRequest envía en el trasfondo, invisible para el usuario, los comandos seleccionados y entregados al SendString al Web-IO. La función integrada DataReceived recoge las respuestas del Web-IO.
- Las respuestas del Web-IO presentan una estructura específica según el tipo.
- Para los Outputs: output;<valor binario del estado de salida en formato hexadecimal>
- Para los Inputs: input;<valor binario del estado de salida en formato hexadecimal>
- Para los contadores: counterx;<estado decimal de contador >
- o counter;<estado decimal de contador 0 >; <estado decimal de contador 0 >; ..... si todos los contadores se deben leer de una sola vez.
- Según la respuesta recibida la función de recepción desvía a la parte correspondiente y actualiza la indicación de los objetos en la ventana del Browser.

```
function DataRequest(SendString)
 {
 var xmlHttp;
 try
  {// Internet Explorer
  if( window.ActiveXObject )
   {
   xmlHttp = new ActiveXObject("Microsoft.XMLHTTP" );
   }
  // Mozilla, Opera und Safari
  else if(window.XMLHttpRequest)
   {
   xmlHttp = new XMLHttpRequest();
   }
  }
 // loading of xmlhttp object failed
 catch( excNotLoadable )
  {
  xmlHttp = false;
  alert("no knowen browser");
  }
 if (xmlHttp)
  { xmlHttp.onreadystatechange = DataReceived;
  xmlHttp.open("GET", SendString, true);
  xmlHttp.setRequestHeader("Cache-Control", "no-store, no-cache, must-revalidate");
  xmlHttp.setRequestHeader("Expires", "Sat, 05 Nov 2005 00:00:00 GMT");
  xmlHttp.setRequestHeader("Pragma","no-cache");
  xmlHttp.send(null);
  }
function DataReceived()
 {
 var HexVal;
 var ReceiveStr;
 if (xmlHttp.readyState == 4)
  {
  if (xmlHttp.status == 200)
   {
   ReceiveStr = xmlHttp.responseText;
   //Input handling
   if (ReceiveStr.substring(0,5)=='input')
    {
    HexVal=HexToInt(ReceiveStr.substring(6,10));
    for (i=0;i<MAXIO;i++)
     {
     if ((HexVal & Math.pow(2,i)) == Math.pow(2,i))
      {
      ioform.cb_input[i].checked = true;
      }
     else
      {
      ioform.cb_input[i].checked = false;
      }
     }
    }
   //Output handling
   if (ReceiveStr.substring(0,6)=='output')
    {
    HexVal=HexToInt(ReceiveStr.substring(7,11));
    for (i=0;i<MAXIO;i++)
     {
     if ((HexVal & Math.pow(2,i)) == Math.pow(2,i))
      {
      ioform.cb_output[i].checked = true;
      }
      else
      {
      ioform.cb_output[i].checked = false;
      }
     }
    }
   //Counter handling
   if (ReceiveStr.substring(0,7)=='counter')
    {
    var slength=ReceiveStr.length;
    if (ReceiveStr.substring(7,8)==';')
     {
     countervalue=ReceiveStr.split(';');
     for (i=0;i<MAXIO;i++)
      {
```

```
        document.getElementById('counter'+i).innerHTML = '<a>'+countervalue[i+1]+'<\/a>';
        }
       }
      else
       {
       if (ReceiveStr.substring(9,10)==';')
        {
        i=(ReceiveStr.substring(7,9));
        document.getElementById('counter'+i).innerHTML = '<a>'+ReceiveStr.substring(10,slength)+'<\
        }
        else
        {
        i=(ReceiveStr.substring(7,8));
        document.getElementById('counter'+i).innerHTML = '<a>'+ReceiveStr.substring(9,slength)+'<\/a
        }
       }
      }
     }
    }
   }
}
```

Aunque el ejemplo mostrado está recortado para el Web-IO 2x Entradas Digitales, 2x Salidas Digitales, la función DataRequest también está preparada para Web-IO con más IOs.

5. Polling
Solicitud cíclica de determinados valores - A fin de permitir una actualización completamente automática de la indicación, se utiliza la función JavaScript Interval. Dependiendo de las cajas de chequeo para el Polling de Otput, Input y Counter se llaman las informaciones correspondientes a un intervalo ajustado del Web-IO.

```
var pollingtimer = window.setInterval("Polling()", 500);
function Polling()
 {
 if (ioform.cb_output_polling.checked==true)
  {
  DataRequest('output?PW='+ioform.ed_password.value+'&');
  }
 if (ioform.cb_input_polling.checked==true)
  {
  DataRequest('input?PW='+ioform.ed_password.value+'&');
  }
 if (ioform.cb_counter_polling.checked==true)
  {
  DataRequest('counter?PW='+ioform.ed_password.value+'&');
  }
 }
```

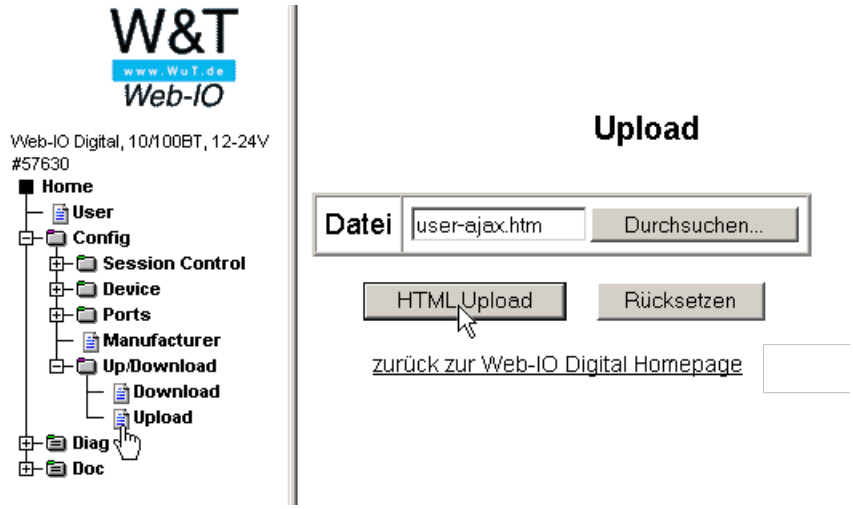
El intervalo deseado puede entrarse en el campo correspondiente de texto y adaptarse chasqueando el botón Set Interval.

```
function setPolInterval()
 {
 var intervaltime=parseInt(ioform.ed_interval.value)
 clearInterval(pollingtimer);
 pollingtimer = window.setInterval("Polling()", intervaltime);
 }
```

Copiar la página web en el Web-IO

Las páginas web basadas en AJAX sólo funcionan, si los datos para la actualización vienen del mismo servidor que la página web misma. Es imposible cargar la página web por el servidor A y llamar los datos por el servidor B.

Si se ha finalizado la programación de la página web, ésta tiene que cargarse en el Web-IO. Esto se hace más cómodamente con unos pocos clics de ratón vía interfaz Web del Web-IO:



El ejemplo asiste todas las funciones corrientes del Web-IO optimado para elWeb-IO 2x Digital Input, 2x Digital Output PoE. Para los otros modelos Web-IO tienen que realizarse en caso necesario adaptaciones en la página web ejemplo. Otros ejemplos de programa para la programación del zócalo los encontrarán en las páginas de herramientas al Web-IO. Una descripción detallada sobre los comandos de los modelos Web-IO digitales la encontrará en el manual de referencia.

Descargar el programa ejemplo

¿No tiene todavía un Web-IO y quiere probar el ejemplo presentado?
No hay problema: Le ponemos a disposición el Web-IO Digital 2xInput, 2xOutput gratis durante 30 días. Rellene sencillamente un pedido muestra y le enviaremos el Web-IO para probar a cuenta abierta. Si nos devuelve el aparato dentro de los 30 días, le abonamos la factura completa.

Al pedido muestra

W&T
www.WuT.de

Le atendemos personalmente:

Wiesemann & Theis
GmbH
Porschestr. 12
42279 Wuppertal
Tel: +49 202/2680-110 (lu-vi de 8-17
horas)
Fax: +49-202/2680-265
info@wut.de