HTTPSVR 가이드

Last update: 2023/11/16

목차

HTTPSVR 가이드

이 화면은 HTTP서버 컴포넌트에 대한 샘플 화면이다.

HTTP서버 컴포넌트는 HTTP 프로토콜에 대한 서버 기능을 제공하는 컴포넌트이다.

예시

템플릿 위치: /HTML5/COMPONENT/HTTPSVR/httpsvr_basic

템플릿 파일

- httpsvr_basic.xml
- httpsvr_basic.js
- 새창으로 실행

화면 스크립트

```
function btnInitXSvrComm_on_mouseup(objInst)
{
  // 초기화
    httpsvr01.init();
}

function btnStartXSvrComm_on_mouseup(objInst)
{
  // 시작
    httpsvr01.open();
}

function btnStopXSvrComm_on_mouseup(objInst)
{
  // 종료
    httpsvr01.close();

    grdSessionList.deleteall();
}

function btnCloseSession_on_mouseup(objInst)
{
    var nCheckedRowCount = grdSessionList.getcheckedrowcount();
    if(nCheckedRowCount == 0) {
        screen.alert("연결 해제할 세션을 체크하세요.");
        return;
    }
```

```
    var nCheckedRow = grdSessionList.getcheckedrow(0);
    var strRemoteIpAddr = grdSessionList.getitemtext(nCheckedRow, 0);
    var strRemotePortNo = grdSessionList.getitemtext(nCheckedRow, 1);

  // 특정 세션 해제
    httpsvr01.closesession(strRemoteIpAddr, strRemotePortNo);
}

function btnSendData_on_mouseup(objInst)
{
    var nCheckedRowCount = grdSessionList.getcheckedrowcount();
    if(nCheckedRowCount == 0) {
        screen.alert("데이터를 송신할 세션을 체크하세요.");
        return;
    }

    var nCheckedRow = grdSessionList.getcheckedrow(0);
    var strRemoteIpAddr = grdSessionList.getitemtext(nCheckedRow, 0);
    var nRemotePortNo = grdSessionList.getitemtext(nCheckedRow, 1);
    var nSocketKey = grdSessionList.getitemtext(nCheckedRow, 2);
    var strSendData = fldSendData.gettext();
    var nRet;

  // 데이터 송신
    SendResponseData(strRemoteIpAddr, nRemotePortNo, nSocketKey,
strSendData, strSendData);
}

function SendResponseData(strRemoteIpAddr, nRemotePortNo, nSocketKey,
strData)
{
    var strSendDataArr = [];

    if (strData === undefined || strData.length == 0) {
        strData = factory.getsystemtime("%Y-%M-%D %h:%m:%s %ms");
    }

    // HTTP Header 데이터는 맨 뒤에 \r\n 데이터를 붙여야 함(예:
"aaa:bbb\r\nccc:ddd\r\n");
    var strHeader = "";

    httpsvr01.send(strRemoteIpAddr, nRemotePortNo, strHeader, strData);
}

function btnClearData_on_mouseup(objInst)
{
  // 수신 데이터 정보 초기화
    fldRecvIpAddr.settext("");
    fldRecvPortNo.settext("");
    fldRecvDataLength.settext("");
    fldRecvData.settext("");
```

```
}

function btnGetSessionCount_on_mouseup(objInst)
{
  // 세션 갯수 조회
    screen.alert(httpsvr01.getsessioncount());
}

////////////////////////////////////////////////////////////////////////////////
/////////////
// EVENT
////////////////////////////////////////////////////////////////////////////////
/////////////

/**
 * 모듈 로딩 완료 및 컴포넌트 초기화 성공시 이벤트
 * @param objInst HTTP서버 컴포넌트 오브젝트
 */
function httpsvr01_on_init(objInst)
{
    factory.consoleprint(objInst.getname() + "_on_init>");
    console.log(objInst.getname() + "_on_init>");
}

/**
 * HTTP 세션이 연결되면 발생하는 이벤트 처리
 * @param objInst HTTP서버 컴포넌트 오브젝트
 * @param strRemoteIP 원격지 시스템의 IP 주소
 * @param nRemotePort 원격지 시스템의 TCP 포트 번호
 * @param strSocketKey 소켓 키 값
 */
function httpsvr01_on_open(objInst, strRemoteIP, nRemotePort, strSocketKey)
{
    factory.consoleprint("OnConnect> strRemoteIP = " + strRemoteIP);
    factory.consoleprint("OnConnect> nRemotePort = " + nRemotePort);
    factory.consoleprint("OnConnect> strSocketKey = " + strSocketKey);

    var nRow = grdSessionList.additem();
    grdSessionList.setitemtext(nRow, 0, strRemoteIP);
    grdSessionList.setitemtext(nRow, 1, nRemotePort);
    grdSessionList.setitemtext(nRow, 2, strSocketKey);
}

/**
 * 데이터 수신시 이벤트
 * @param objInst HTTP서버 컴포넌트 오브젝트
 * @param strRemoteIP 데이터를 송신한 시스템의 IP 주소
 * @param nRemotePort 데이터를 송신한 시스템의 TCP 포트 번호
 * @param strSocketKey 소켓 키 값
 * @param strHttpHeader HTTP 헤더 문자열(맨 마지막 \r\n은 제거되어 있음)
 * @param nDataLen 수신한 데이터 길이
```

```
 * @param strData 수신한 데이터
 */
function httpsvr01_on_recv(objInst, strRemoteIP, nRemotePort, strSocketKey,
strHttpHeader, nDataLen, strData)
{
    factory.consoleprint("OnReceive> strRemoteIP = " + strRemoteIP);
    factory.consoleprint("OnReceive> nRemotePort = " + nRemotePort);
    factory.consoleprint("OnReceive> strSocketKey = " + strSocketKey);
    factory.consoleprint("OnReceive> strHttpHeader = " + strHttpHeader);
    factory.consoleprint("OnReceive> nDataLen = " + nDataLen);
    factory.consoleprint("OnReceive> strData = " + strData);
    console.log(objInst.getname() + "_on_recv : ", arguments);

    fldRecvIpAddr.settext(strRemoteIP);
    fldRecvPortNo.settext(nRemotePort);
    fldRecvDataLength.settext(nDataLen);
    fldRecvData.settext(strData);

    if (chkAutoReply.getcheck()) {
        SendResponseData(strRemoteIP, nRemotePort, nSocketKey);
    }
}

/**
 * HTTP 세션 해제시 이벤트
 * @param objInst HTTP서버 컴포넌트 오브젝트
 * @param strRemoteIP 세션이 해제된 시스템의 IP 주소
 * @param nRemotePort 세션이 해제된 시스템의 TCP 포트 번호
 * @param strSocketKey 소켓 키 값
 */
function httpsvr01_on_close(objInst, strRemoteIP, nRemotePort, strSocketKey)
{
    factory.consoleprint("OnClose> strRemoteIP = " + strRemoteIP);
    factory.consoleprint("OnClose> nRemotePort = " + nRemotePort);
    factory.consoleprint("OnClose> strSocketKey = " + strSocketKey);

    var nRowCount = grdSessionList.getrowcount();
    var i;

    for(i = 0; i < nRowCount; i++) {
        if(grdSessionList.getitemtext(i, 0) == strRemoteIP) {
            if(grdSessionList.getitemtext(i, 1) == nRemotePort) {
                grdSessionList.deleterow(i);
                break;
            }
        }
    }
}

/**
 * 에러 발생시 이벤트
```

```
 * @param objInst HTTP서버 컴포넌트 오브젝트
 * @param nErrorCode 에러코드
 * @param strErrorMsg 에러메시지
 */
function httpsvr01_on_error(objInst, nErrorCode, strErrorMsg)
{
    screen.alert(objInst.getname() + "_on_error : " + nErrorCode + ", " +
strErrorMsg);
}
```