webdyn
flexitron group

TITAN

Application Note 62

Sending data to DEXMA platform

Sending data to DEXMA platform

1. Detalles del esceniario

Los router Titan disponen de todas las funcionalidades típicas de un router 4G/3G/2G pero además cuentan con una serie de prestaciones adicionales que lo convierten en uno de los routers con más prestaciones del mercado.

Una de las prestaciones adicionales es la capacidad de datalogger donde el router Titan almacena en su memoria no volátil distintos tipos de registros en un formato JSON. Estos registros almacenados pueden provenir de lecturas Modbus, o de capturas de datos SERIE por sus puertos RS232 / RS485, de posiciones GPS, etc Estos registros de tipo JSON son almacenados por el router Titan en su memoria interna no volátil para ser posteriormente enviados a una plataforma remota mediante comunicaciones tipo HTTP, HTTPS, MQTT, MQTTS, FTP, FTPS

Como se indica, el router Titan almacena en su memoria interna dichos registros JSON en un formato predeterminado y propietario. En ocasiones ésto puede ser un problema para la comunicación con aquellas plataformas que esperan recibir la información en un determinado formato (es decir, con un formato distinto de JSON al utilizado por defecto por el router Titan).

Mediante la funcionalidad "**JSON Transformer Function Script**" el router Titan permite realizar un formateo de cualquier objeto JSON antes de su almacenamiento en su memoria interna. Esta característica permite por tanto acomodar los objetos JSON con el formato adecuado para cada plataforma. Realmente no es necesario tranformar los datos en un formato JSON, puede transformarse en cualquier formato String.

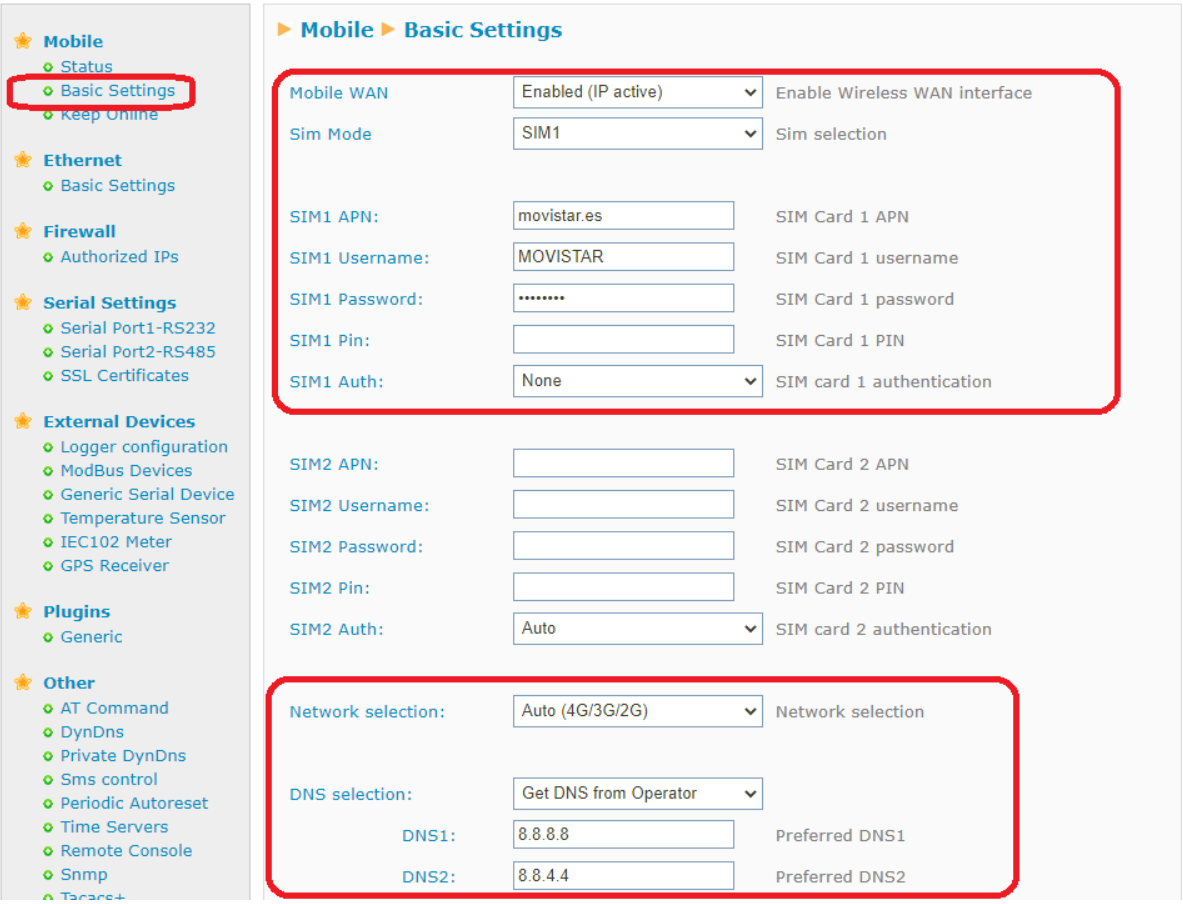
En esta nota de aplicación se desarrollará el ejemplo completo para el envío de datos a la conocida plataforma de Dexma (https://www.dexma.com/what-is-dexma-platform/) la cual requiere de un formato especial de los JSON enviados y de una serie de cabeceras.

En esta nota de aplicación en concreto vamos a suponer que se deben leer 2 registros de 2 PLCs con comunicaciones Modbus conectados a un Webdyn EasyTunnel por su puerto RS485.

En concreto se pretende lo siguiente: cada 1 minuto, el dispositivo WebDyn-Easytunnel debe leer los registros Modbus con dirección 40000 y 40001 del PLC1 y los registros 40000 y 40001 del PLC2. En ambos PLCs el registro 40000 corresponde con la temperatura medida y el registro 40001 con la humedad. Las lecturas realizadas deben almacenarse en la memoria interna no volátil del Webdyn EasyTunnel (en su datalogger) y éste debe enviar dichos datos leídos a la plataforma DEXMA siempre que sea posible (haya cobertura, conectividad IP, ...) La comunicación con los PLCs es a través de un bus RS485 con una configuración 9600,8,N,1

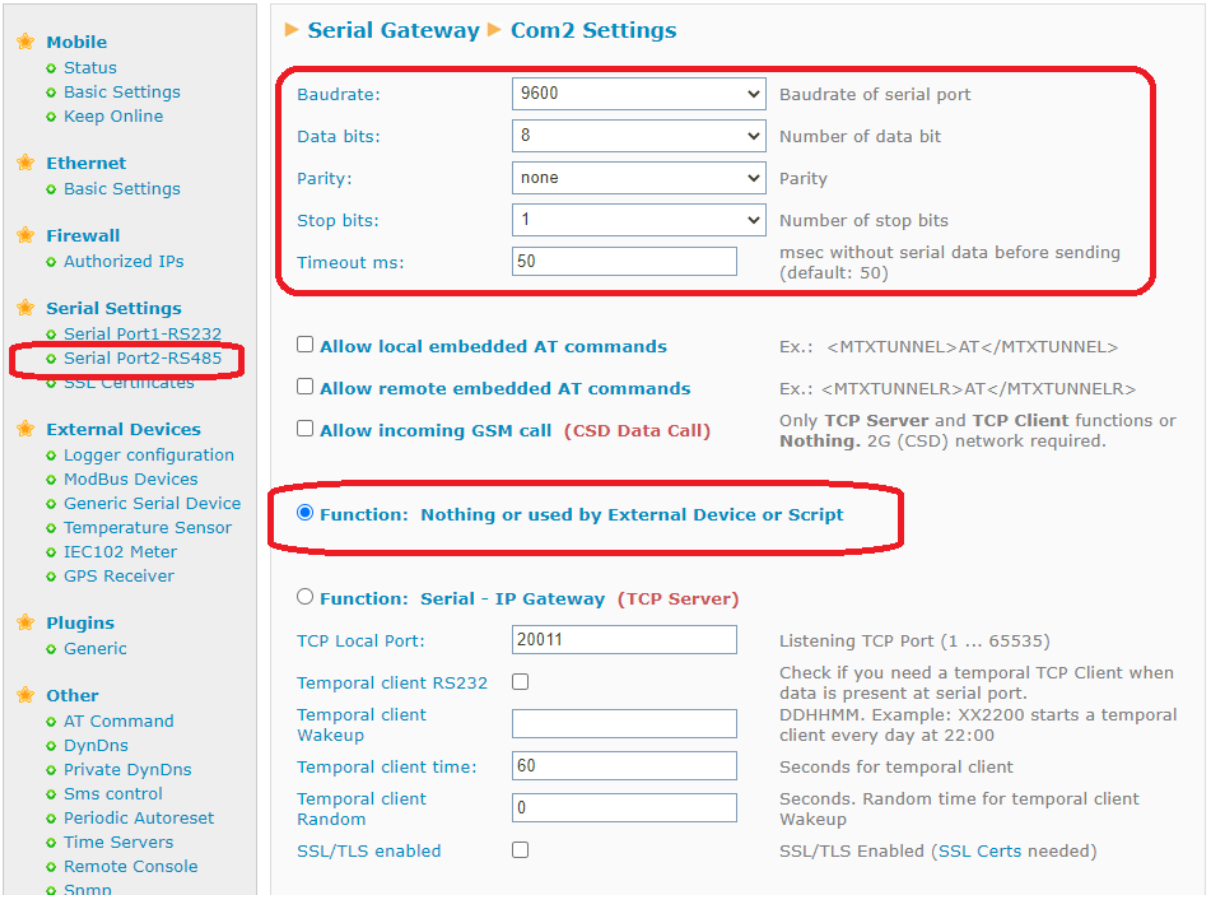
2. Configuración de la Mobile WAN

El Webdyn EasyTunnel debe comunicarse con la plataforma DEXMA a través de comunicaciones 4G/3G/2G, por lo que la sección "**Mobile=>Basic Settings**" debe configurarse apropiadamente en función de la tarjeta SIM utilizada.



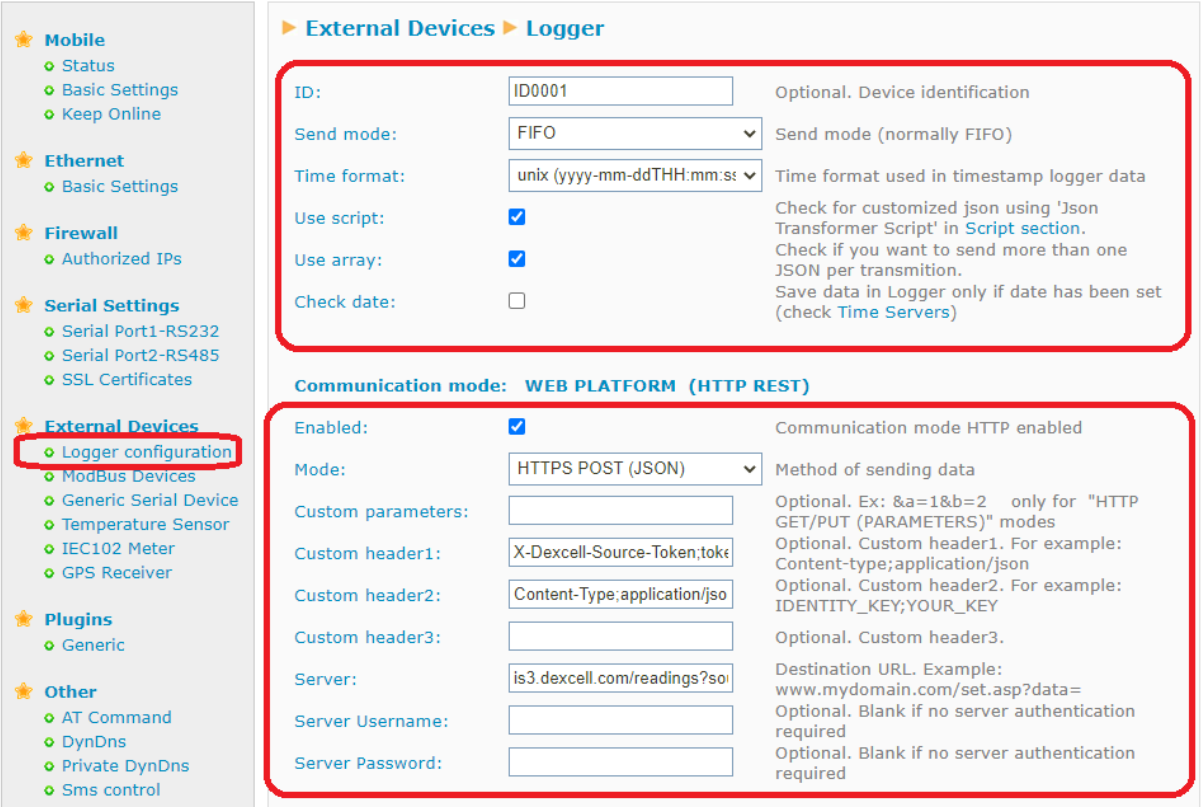
3. Configuración del puerto serie RS485

La comunicación con los dos PLCs se realizará a través de puerto serie RS485 a 9600,8,N,1, por lo que debe configurarse la sección "**Serial Settings => Serial Port2-RS485**" ajustando los parámetros como se indica a continuación.



4. Configuración del Logger

El siguiente paso es configurar el datalogger interno del WebDyn EasyTunnel. Para ello debe acudirse al menú "**External Devices => Logger configuration**". La configurción debería ser similar a la que se muestra en la siguiente figura:



Cabe destacar los siguientes parámetros:

- "**Time format**" es necesario seleccionar formato "unix", que es la soportada por la plataforma DEXMA.
- "**Use script**" es muy importante seleccionar esta casilla, pues al activarla se ejecutará el script que incorporaremos más adelante y que transformará el JSON estándar creado por el WebDyn-Easytunnel en un JSON con el formato necesario soportado por la plataforma DEXMA.
- "**Use array**" DEXMA permite el envío de datos en array, pues activamos también la casilla,
- "**Custom header1**" debe introducirse, tal y como se indica en https://support.dexma.com/hc/en-gb/articles/360013772759-Using-the-insertion-API-to-introduce-data-in-a-gateway el header "x-dexcell-sorce-token" que es distinto para cada dispositivo. En el caso de este ejemplo tiene el valor:

X-Dexcell-Source-Token;token123456789 (campos separados por ";", no por ":")

- "**Custom header2**" debe introducirse:

Content-Type;application/json (campos separados por ";", no por ":")

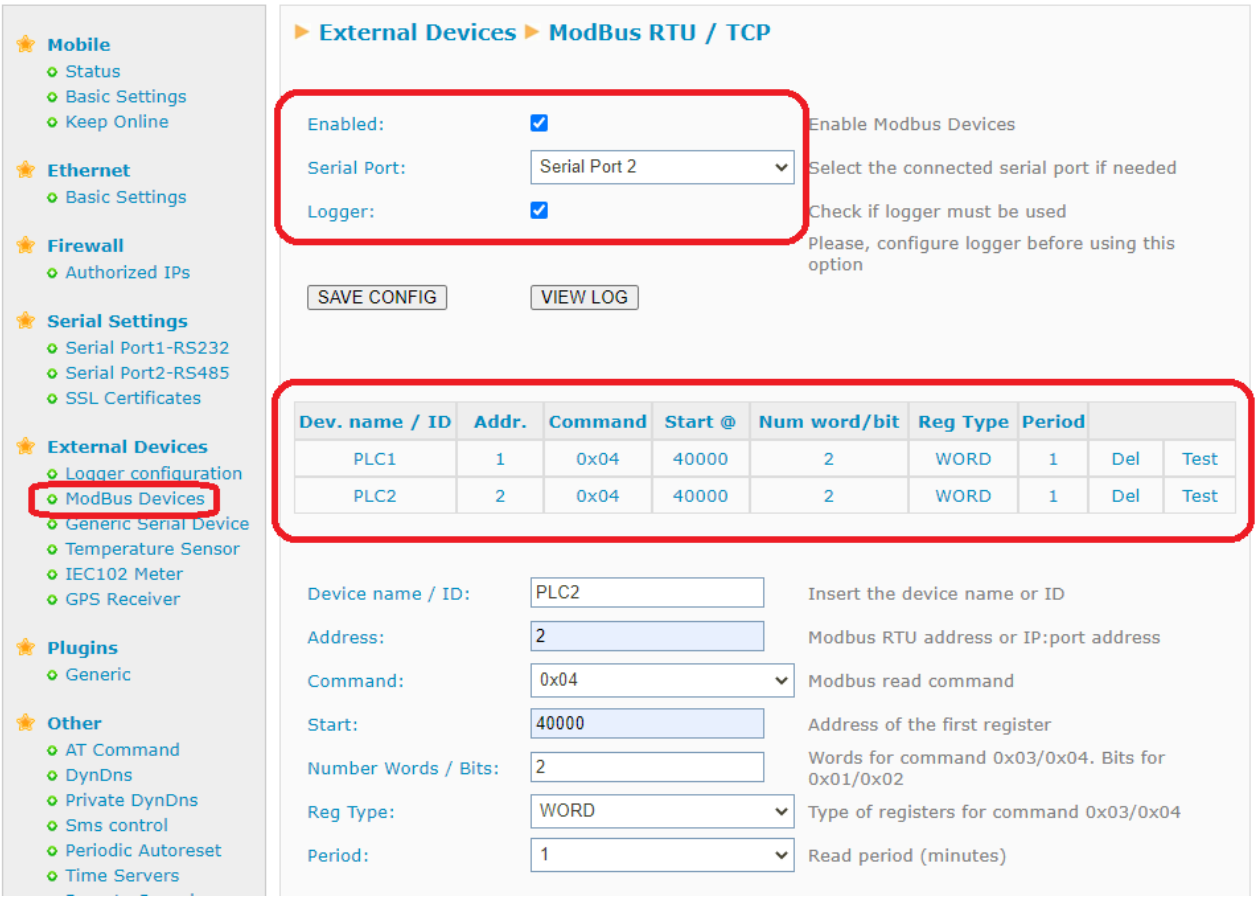
- "**Server**" aquí introduciremos la URL del servidor, que tendrá este formato:

is3.dexcell.com/readings?source_key=mac-123456789

donde mac-123456789 deberá ser substituido adecuadamente con el KEY obtenido en su plataforma DEXMA.

5. Configuración de la seccion Modbus

En esta sección de configuración "**External Devices => Modbus Devices**" configuraremos las lecturas Modbus que deben realizarse sobre los 2 PLCs



Debe habilitarse el servicio Modbus activando la casilla "**Enabled**". Debe seleccionar "**Serial Port 2**", pues la lectura se realizará a través del puerto RS485. Y también debe activarse la casilla "**Logger**", pues los registros Modbus leídos de los PLCs deben almacenarse en el datalogger interno del Webdyn EasyTunnel.

También deben crearse dos dispositivos que en este ejemplo hemos llamado PLC1 y PLC2.

En "**address**" debe indicarse la dirección Modbus de cada uno de los PLCs, por lo que en PLC1 pondremos la dirección 1 y en PLC2 la dirección 2. El comando Modbus que utilizaremos para la lectura de los registros 40000 y 40001 será el 0x04, por lo que seleccionaremos 0x04 en el campo "**Command**". El registro iniciar a leer en ambos PLCs es el registro 40000, pues indicaremos en el campo "**Start**" el valor 40000. Pretendemos leer 2 registros Modbus de cada PLC (el 40000 y 40001) por lo que en el campo "**Number Words**" indicaremos 2.

El tipo de registro “**Reg Type**” lo seleccionaremos com “WORD” y en “**Period**” seleccionaremos 1, pues queremos lecturas de los registros Modbus cada minuto.

6. Configuración del SCRIPT para adaptar el JSON generado por el Webdyn EasyTunnel con formato estándar al formation necesario para la correcta comunicación con la plataforma DEXAM

El formato de JSON estándar generado por el Webdyn EasyTunnel con los datos leídos y con el que los enviaría a una plataforma tendrá un formato como el que sigue:

[

{“IMEI”:”869101054286683”,”TYPE”:”MODB”,”TS”:”2022-11-11T12:17:00Z”,”ID”:”PLC1”,”A”:”1”,”ST”:”40000”,”N”:”2”,”V”:[225,62],”P”:”ID0001”},

{“IMEI”:”869101054286683”,”TYPE”:”MODB”,”TS”:”2022-11-11T12:17:01Z”,”ID”:”PLC2”,”A”:”2”,”ST”:”40000”,”N”:”2”,”V”:[225,62],”P”:”ID0001”}

]

Pero el formato requerido por la plataforma DEXMA, tal y como se indica en sus manuales, (https://support.dexma.com/hc/en-gb/articles/360013772759-Using-the-insertion-API-to-introduce-data-in-a-gateway) debe ser:

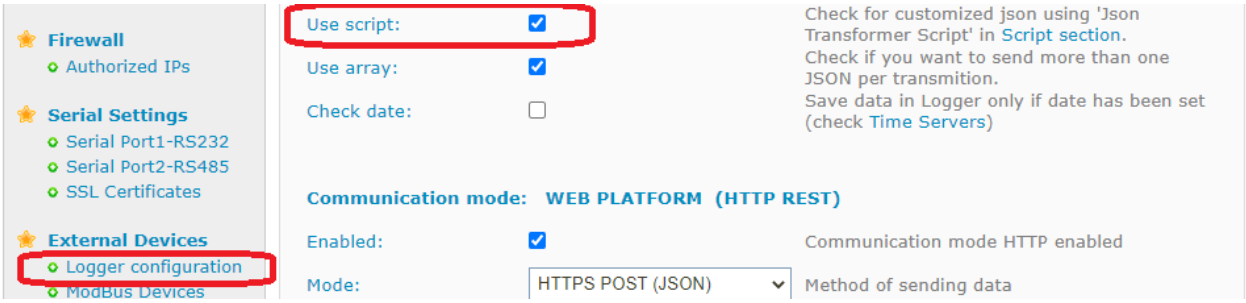
[

{“did”:”PLC1”,”sqn”:1,”ts”:”2022-11-11T12:20:01Z”,”values”:[{“p”:301,”v”:22.5},{“p”:401,”v”:62}]},

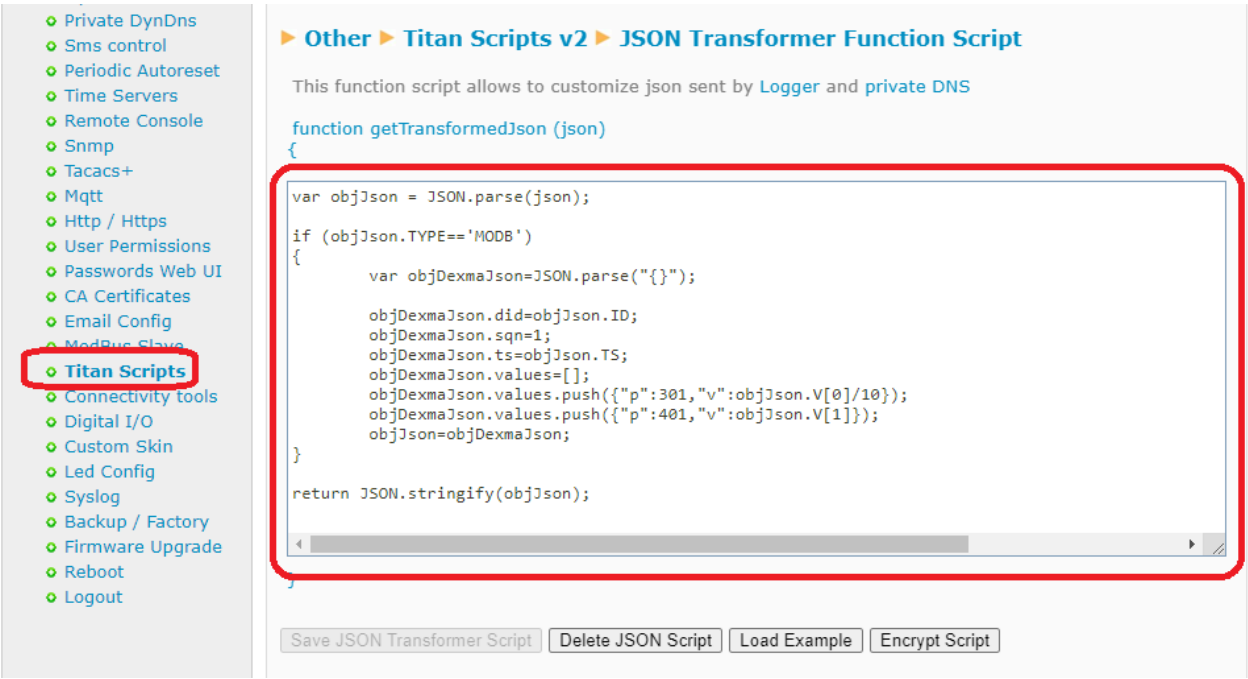
{“did”:”PLC2”,”sqn”:1,”ts”:”2022-11-11T12:20:01Z”,”values”:[{“p”:301,”v”:22.5},{“p”:401,”v”:62}]}

]

Por lo tanto hay que crear un pequeño SCRIPT para adaptar los datos. Recuerde que este script de transformación se ejecutará porque anteriormente en la sección “**External Devices => Logger Configuration**” seleccionamos la casilla “**Use script**”



Bien, para introducir el código del script de tansformación de datos debe acudirse al menú "**Other => Titan Scripts**". Concretamente el script de transformación debe introducirse en la sección "**JSON Transforme Function Script**"



Como se indica en el manual, esta función para formatear el JSON es ejecutada ANTES de almacenar los datos de 1 lectura en el datalogger interno y se pasa como parámetro el JSON con el firmato estándar del dispositivo, es decir, por ejemplo:

{"IMEI":"869101054286683","TYPE":"MODB","TS":"2022-11-11T12:17:00Z","ID":"PLC1","A":"1","ST":"40000","N":"2","V":[225,62],"P":"ID0001"}

Bien, pues el script de transformación debe transformar ese anterior JSON en el siguiente para ser compatible con la plataforma DEXMA.

{"did":"PLC1","sqn":1,"ts":"2022-11-11T12:20:01Z","values":[{"p":301,"v":22.5},{"p":401,"v":62}]}

El código del SCRIPT sería el siguente (con comentarios en añadidos).

```
//Creamos una variable objeto de tipo JSON, pues la variable "json" que recibe la //función
como argumento es de tipo "String".

var objJson = JSON.parse(json);
```

```
//Si en el json original recibido, se indica que es un JSON de datos Modbus ...
if (objJson.TYPE=='MODB')
{
        //Creamos un nuevo objeto JSON, donde almacenaremos los datos
        //transformados
        var objDexmaJson=JSON.parse("{}");

        //Como campo did de la plataforma DEXMA, utilizaremos el campo ID del JSON
        //original (es decir, PLC1 y PLC2)
        objDexmaJson.did=objJson.ID;

        //El campo sqn, siempre a 1.
        objDexmaJson.sqn=1;

        //Utilizamos el mismo timestamp que el JSON original
        objDexmaJson.ts=objJson.TS;

        //Añadimos al JSON un array donde almacenaremos los valores de
        //temperatura y humedad.
        objDexmaJson.values=[];

        //Añadimos al array el parámetro 301 junto con el valor de la temperatura, que //corresponde
        con el primer registro leído, el 40000, almacenado en la posición //0 del array del json original
        (resaltado en verde en la página anterior) //Dividimos el valor por 10 para que tenga 1 //decimal.
        objDexmaJson.values.push({"p":301,"v":objJson.V[0]/10});

        //Añadimos al array el parámetro 401 junto con la humedad, que corresponde con el segundo
        registro leído, el 40001, almacenado en la posición 1 del array del json original (resaltado en
        rojo en la página anterior)
        objDexmaJson.values.push({"p":401,"v":objJson.V[1]});
```

```
        //Finalmente copiamos el JSON transformado, para devolverlo como resultado
        //de la función de transformación.
        objJson=objDexmaJson;
}
//Devolvemos el JSON en formato String como resultado de la funciín de //transformación.
return JSON.stringify(objJson);
```

Una vez introducido el script y guardado basta con reiniciar el equipo. ya debería funcionar todo correctamente. En caso de problemas consulte los LOG del equipo y revise si necesita introducir algún certificado digital en la sección “Other => Ca Certificates”.

¿Más dudas?

Escríbanos sus consultas a iotsupport@mtxm2m.com