mdWebSockets Component Documentation v0.1
(Not complete yet… Updated 4/7/2020)

Table of Contents

System Requirements

Mac or Windows version 15 and up

Setting Up

- Step 1: Place the mdWebSocket_vXX.4dbase file into the “Components” folder next to your host database files

- Step 2: Create the “hook” methods in your database.
  - mdWebSocket_allowConnection
  - mdWebSocket_onConnect
  - mdWebSocket_onData
  - mdWebSocket_onDisconnect

  (Be sure to check the attribute “Shared by components and host database” for these methods)

- Step 3: Add the following in the Database Method “On Server Startup”

```
  // mdWS_EnterSerialNumber ("XXXX-XXXX-XXXX-XXXX")
mdWS_Start(8081)
mdWS_OpenConnectionsWindow
```

  (The Serial Number and OpenConnectionsWindow are optional…)

- Step 4: Add the following in the Database Method “On Server Shutdown”

```
mdWS_Stop
```

- Step 5: Restart the database or run the “On Server Startup” method manually

Hook Methods

- **mdWebSocket_allowConnection**
  Parameters/Return:
    1 return:
      $0: BOOLEAN: $connectionAllowed
    1 parameter:
      $1: OBJECT: $connectionDetails
  Description:

This is the first hook method called when a connection is made. This method is called synchronously and the expectation is to return a Boolean. Be careful of TRACES here, they are okay but put them in a condition that only applies to your connection. The browser will try again if it times-out connecting and you would get a never ending loop of traces. True to allow the connection or false to block the connection.

- **mdWebSocket_onConnect**
  Parameters:
    No return variable.
    1 parameter:
      $1: TEXT: $webSocketReference
  Description:

This method is called after the "_allowConnection" hook has APPROVED of the websocket request. It is called asynchronously in a new process.

[Tip] Use this method to send the just logged in user "welcome" data or to alert others there is a new active websocket connection.

- **mdWebSocket_onData**
  Parameters:
    No return variable.
    2 parameters:
      $1: TEXT: $webSocketReference
      $2: BLOB: $incomingDataBlob
  Description:

This method is called when the web browser sends data to the server. It is called asynchronously in a new process. I highly recommend sending json objects over the websocket and then in this method you would convert the blob to text and then JSON Parse the text into a 4D object.

[Tip] If you have used "mdWs_ChangeRegisteredName" to name the websocket reference properly, at the top of this method, split apart the name you made into variables that will help you keep track of the userId or userName or IPAddress.

- **mdWebSocket_onDisconnect**
  Parameters:
    No return variable.
    1 parameter:
      $1: TEXT: $webSocketReference
  Description:

This method is called when the web browser closes and the websocket disconnects. It is called asynchronously in a new process.

[Tip] Use this method to alert others this user has left the websocket application.

Commands Available List

- **mdWS_ChangeRegisteredName**

Example:

```
mdWS_changeRegisteredName ($currentReference;$newReference)
```

Parameters:

2 parameters:

$1: TEXT: $currentReference
$2: TEXT: $newReference

Description:

Use this method to rename a websocket reference. By default it is an incrementing number with the mdWebSocket prefix (ie: “mdWebSocket1” or “mdWebSocket2”). I usually use this method in the “_allowConnection” hook method to rename to a valid sessionId and IPAddress. Then I call it again in the “_onConnect” hook method to something more useful like “[33]USERNAME_10.0.0.22.”

[Tip] The websocket reference is passed to every hook. So if your reference is named properly, it is easy to parse and know exactly who the request came from without having to query the database against a sessionId everytime.

- **mdWS_Close**

Example:

```
mdWS_Close ($referenceToClose)
```

Parameters:

1 parameter:

$1: TEXT: $referenceToClose

Description:

This is used to manually close a websocket connection from the server. I don’t use this very much. Usually if I need to disconnect the websockets, I would stop the server which in turn would close all the connections. I wanted this ability though if there was a user you needed to kick off the websocket for various reasons (maintenance, user changed job codes, user should’ve logged off earlier in the day)

- **mdWS_EnterSerialNumber**

Example:

```
mdWS_EnterSerialNumber (“XXXX-XXXX-XXXX-XXXX-XXXX”)
```

Parameters:

1 parameter:

$1: TEXT: Serial Number if registered.

Description:

This method needs to be called BEFORE starting the server with mdWS_Start. If successful, then more than 3 websocket connections will be allowed and spawn up as people connect. If false, it will act like this method was not called and you will be limited to the (free forever) 3 connection limit.

- **mdWS_GetAllProcesses**

Example:

```
mdWS_GetAllProcesses ($webSocketReference;->$objectArray)
```

Parameters:

1 parameter:

$1: TEXT: Serial Number if registered.

$2: POINTER: Pointer to an Array Object to receive websocket details.

Description:

- **mdWS_InitHooks**

  Example:

  ```
  mdWS_InitHooks
  ($allowConnectionMethodName;$onConnectMethodName;$onDataMethodName;$onDisconnectMethodName)
  ```

  Parameters:

  4 parameters:

  $1: TEXT: Allow Connection Hook Method Name

  $2: TEXT: On Connect Hook Method Name

  $3: TEXT: On Data Hook Method Name

  $4: TEXT: On Disconnect Hook Method Name

  Description:

  This method allows you to rename the default "hook" methods so you can use whatever naming convention you would like.

  [TIP] Make sure the methods have the *"Shared by components and host database"* attribute checked.

- **mdWS_OpenConnectionsWindow**

  Example:

  ```
  mdWS_OpenConnectionsWindow
  ```

  Parameters:

  0 parameters

  Description:

  Use this method to open a connections window ON THE SERVER. I will write a client version at a later date.

- **mdWS_SendObject**

  Example:

  ```
  mdWS_SendObject ($webSocketReference;$objectToSend)
  ```

  Parameters:

  2 parameters:

  $1: TEXT: The reference of the websocket connection (wild character @ allowed!)

  $2: OBJECT: 4D Object that is sent as JSON text to the browser.

  Description:

- **mdWS_SendText**

  Example:

  ```
  mdWS_SendText ($webSocketReference;$textToSend)
  ```

  Parameters:

  2 parameters:

  $1: TEXT: The reference of the websocket connection (wild character @ allowed!)

  $2: TEXT: Text that is sent to the browser.

Description:

- **mdWS_Start**

  Example:

  `mdWS_Start ($portNumber)`

  Parameters:

  1 parameter:

  $1: LONGINT: Port number to listen to connections, default is 8081.

  Description:

  This is the method that starts the websocket server and spawns some listening processes.

- **mdWS_Stop**

  Example:

  `mdWS_Stop ($closeSilently)`

  Parameters:

  1 parameter:

  $1: BOOLEAN: $closeSilently

  Description:

  Use this method to stop the websocket server from listening. If you pass True then it will skip the 4D Progress Bar and just close. It takes a few seconds to close all the connections depending on how many are there.

Extra/Helpful Tips

- ToDo: Clientside examples…
- ToDo: nGinx Reverse Proxy setup for SSL support
  - Mac
  - Windows

Credits

Made by Matt Davis
https://the4dmd.com