TECHNODATA INTERWARE GMBH

MATRIX LOGON

HIGH-QUALITY WEB AUTHENTICATION

Benutzerhandbuch

Benutzerhandbuch

Die in diesen Unterlagen enthaltenen Angaben und Daten können ohne vorherige Ankündigung geändert werden. Ohne ausdrückliche schriftliche Erlaubnis der TDi GmbH darf kein Teil dieser Unterlagen für irgendwelche Zwecke vervielfältigt oder übertragen werden, unabhängig davon, auf welche Art und Weise oder mit welchen Mitteln - elektronisch oder mechanisch - dies geschieht.
Es gelten die AGB der TDi GmbH. Hiervon abweichende Vereinbarungen bedürfen der Schriftform.

Matrix Logon ist eine patentierte Lösung der TDi GmbH.

Software Lizenz
Die Software und die mitgelieferte Dokumentation sind urheberrechtlich geschützt. Durch die Installation erklären Sie sich mit den Vertragsbedingungen des Lizenzvertrages einverstanden.

Lizenzvertrag
TDi GmbH (kurz TDi) gewährt dem Käufer das einfache nicht ausschließliche und nicht übertragbare Lizenz-Recht, die Software auf einem einzelnen Computer, bzw. vernetzten Computersystem (LAN) zu benutzen. Das Kopieren oder jede anderweitige Vervielfältigung von Teilen oder der gesamten Software sowie das Mischen und Verbinden mit anderer Software ist ausdrücklich untersagt. Zu Sicherungszwecken darf der Käufer eine einzelne Kopie der Software für sich anfertigen (Back-up). TDi behält sich vor, die Software zu ändern, weiterzuentwickeln, zu verbessern oder durch eine neue Entwicklung zu ersetzen. Es besteht keine Verpflichtung für TDi, den Käufer über Änderungen, Neu- und Weiterentwicklungen sowie Verbesserungen zu informieren oder ihm diese zur Verfügung zu stellen. Eine rechtlich verbindliche Zusicherung bestimmter Eigenschaften wird nicht gegeben. TDi haftet nicht für Schäden, es sei denn, ein Schaden ist durch Vorsatz oder grobe Fahrlässigkeit auf Seiten der TDi oder deren Erfüllungs- und Verrichtungsgehilfen verursacht worden. Jede Haftung für indirekte sowie für Begleit- und Folgeschäden ist ausgeschlossen.

Einhaltung der CE/FCC-Bestimmungen

CE Dieses Gerät wurde auf Einhaltung der Grenzwerte für Digitalgeräte der Klasse B geprüft und zugelassen.

Der Betrieb unterliegt folgenden Bedingungen:
1. Das Gerät darf keine schädlichen Störstrahlungen verursachen
2. Das Gerät muß Störstrahlungen verarbeiten können, einschließlich solcher Strahlungen, die zu einem unerwünschten Betrieb führen könnten.

Das Produkt erfüllt die Grenzwerte gemäß EN55022 Class B, EN50081-1, EN50082-1 and EN55024.
Eine Änderung des Produktes ohne die ausdrückliche Zustimmung von TDi GmbH, kann dazu führen, daß die CE/FCC-Bestimmungen nicht mehr erfüllt sind. In diesem Fall erlischt das Nutzungsrecht des Anwenders für dieses Produkt.

Vorwort

Danke, dass Sie sich für Produkte aus dem Hause TechnoData Interware interessieren. Mit diesem Benutzerhandbuch möchten wir Sie so gut wie möglich im Umgang mit Matrix unterstützen. Sollen Sie noch Fragen oder Anregungen haben, stehen wir Ihnen gerne auch direkt zur Verfügung.

Unsere USB-Sticks werden als „Dongle“ bezeichnet, wenn sie als Software-Kopierschutz verwendet werden. Im Einsatz mit Matrix Logon werden sie als „Token“ bezeichnet.

Auf unserer Webseite *www.tdi-matrix.de* finden Sie stets die aktuelle Software und über dieses Handbuch hinausgehende Beispiele und Readme-Dateien für Windows, Mac OS X, Linux sowie zusätzliche Tools zur Optimierung Ihrer Arbeit mit Matrix. Für Verbesserungsvorschläge und Tipps sind wir stets offen.

Sie erreichen uns unter *support@tdi-matrix.de*

Matrix Logon

Wissen Sie, wer sich alles auf Ihrem Web-Portal einloggt?

Das Problem mit Passworten ist, dass man sehr schnell keine Kontrolle mehr darüber hat. Passworte werden einfach an jemand anderen weitergegeben. Oder sie werden aufgeschrieben und von anderen gelesen. Oder durch Pishing abgefangen. Passworte hinterlassen Spuren auf dem Rechner, werden belauscht, verloren usw.

Passwort-Ersatz und das Logon-Konzept

Matrix Logon ist optimiert als sehr gute Alternative der noch verbreiteten User-ID + Password Methode als Zugang fürs Internet oder Intranet. Es handelt sich hier um ein neuartiges patentgeschütztes Verfahren. Sicherheitskonzept und Benutzerfreundlichkeit sind direkt aus dem Kopierschutz übernommen.

Unsere Lösung verhindert die unbefugte Benutzung und das Pishing von Nutzerdaten.

Auch hier steht der User im Mittelpunkt:

Matrix Logon ist konfliktfrei mit beliebigen Browsern ohne Token-Treiber und Cross-Platform anwendbar. Die einwandfreie Funktion in jedem Systemumfeld und in jeder Infrastruktur ist gewährleistet.

Matrix Logon basiert auf Hardware und gewährleistet eine Two-Factor-Authentication. Ein Matrix USB-Token ersetzt Login-Daten wie User-ID und/oder Passwort.

Das Verfahren ist sehr sicher, weil die gesamte Kommunikation im Token verschlüsselt wird. Jedes Datenpaket ist mit einem Zufallswert verschlüsselt. Somit sind alle übertragenen Datenpakete verschieden. Das und weitere Verfahren machen die Übertragung sehr sicher.

Wie funktioniert es?

Matrix Logon basiert auf zwei Komponenten:

1. Der Matrix-Token und die Web-Client Application auf der Client-Seite
2. Der Web Server auf der Betreiber-Seite

Wenn sich der Nutzer einloggt, sendet der Server eine auf Zufallszahlen basierende Anfrage. Der Nutzer muss den Matrix USB-Token eingesteckt haben und ein ggf. eine PIN-Nr. eingeben.

Die Server-Anfrage, die PIN und einige andere Datenblöcke werden im Token verschlüsselt. Dieser verschlüsselte Datenblock wird zurück zum Server geschickt.

Wichtig: Der maßgebliche 128-Bit-Schlüssel wird im Token und auf dem Server des Betreibers gespeichert, aber niemals übertragen.

Matrix Logon bietet Entwicklern eine sichere Lösung zur Nutzer-Authentifizierung im Internet. Nur Nutzer mit einem gültigen Token + PIN dürfen auf die Server-Anwendung zugreifen.

Matrix Logon funktioniert mit allen Web-Browsern in beliebiger Konfiguration, ohne Cookies, PlugIns oder Applets. Die gesamte Kommunikation durch eine normale EXE-Datei (Web-Client Application) geregelt.

Die Komponenten der Server-Seite werden dem Betreiber im Quelltext zur Verfügung gestellt. Diese können dann einfach in beliebige Scripte, Web-Anwendungen oder Datenbanken eingebunden werden.

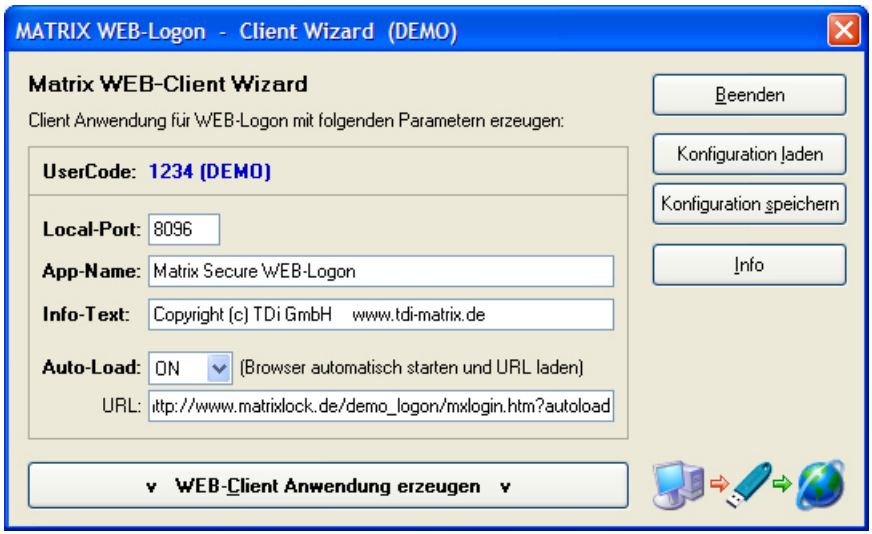
Der folgende Abschnitt beschreibt das Request/Response Interface für beide Seiten: Client und Server.

WEB-Client Application

Die Web-Client Application wird zum Endkunden geschickt und repräsentiert dort einen Teil Ihrer Anwendung. Um für Ihre Nutzer personalisierte und flexible Web-Client Applications zu generieren, bieten wir ein Standard-Tool: Matrix Web-Client Wizard.

Matrix WEB-Client Wizard enthält ein Tool namens **MxGenWeb**, mit dem die Datei WEB-Client Application (MxWeb32.exe) erzeugt wird, die dann zum Nutzer geschickt wird.

MxGenWeb hat folgende Grundeinstellungen:



Local-Port

Hier wir ein lokaler Port bestimmt. Dieser wird von der Web-Client Application für die HTTP-Anfragen benutzt. Derselbe Port muss ebenfalls im Server-Script verwendet werden.

Achtung: Es dürfen nur freie Ports benutzt werden.

App-Name/Info-Text

Hier können Sie den Namen Ihrer Anwendung und die Anzeige für den Info-Dialog festlegen.

Auto-Load

Hier kann eingestellt werden, ob beim Einstecken des Matrix-Tokens automatisch Ihre Website geladen wird oder nicht.

Diese Standardeinstellung kann vom User modifiziert werden.

Konfiguration laden / speichern

Um die Verwaltung verschiedener Anwendungen zu unterstützen, können die Einstellungen in einem Projekt gespeichert werden. Dies ermöglicht eine effiziente spätere Zuordnung.

v WEB-Client Anwendung erzeugen v

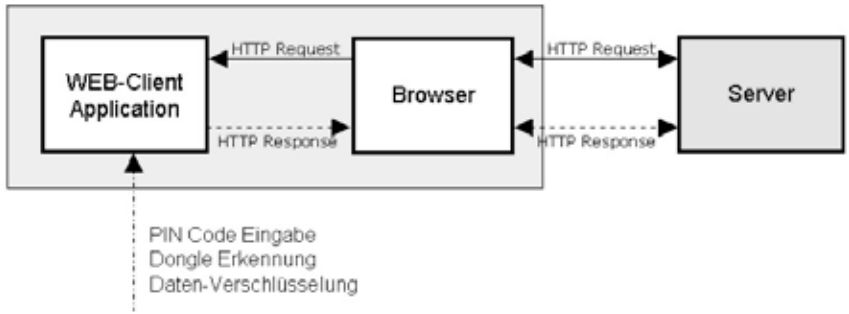
Generiert die personalisierte Web-Client Application (standard Name MxWeb32.exe), die dann zum Nutzer (Client) geschickt wird.

Request/Response Interface

Auf der Server-Seite werden Requests (Anfragen) erzeugt (HTTP-Request), zum Web-Browser geschickt und von der Web-Client Application verarbeitet.

Die Web-Client Application erhält den HTTP-Request, erwartet die PIN, führt die Token-Prüfung durch und leitet die Verschlüsselung ein.

Das Verschlüsselungsergebnis wird als HTTP-Response zum Server zurückgeschickt.



Die Web-Client Application erhält die HTTP-Anfrage und schickt die ermittelte HTTP-Antwort verschlüsselt an den Server zurück.

Die Anfragen (Requests) sehen etwa so aus:

```
GET
/?site=www.test.com/
logon.php&ssid=4f6de45a15a68059a5eb4639fafe1f06&action=clientOk HTTP/1.1
Host: localhost:8096
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; ja-JP; rv:1.7.8) Gecko/
20050511 Firefox/1.0.4
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9, text/
plain;q=0.8,image/png
.....
```

Die Web-Client Application muss die Anfrage verstehen. Dafür wird nur die 2. Zeile benutzt:

```
/?site=www.test.com/logon.php&ssid=4f6de45a15a68059a5eb4639fafe1f06&
action=clientOk HTTP/1.1
```

Dies ist ein Beispiel mit **3** Parametern: site, ssid, action.

site ... dies ist die Website, auf die eingeloggt werden soll

ssid ... ist die Session-ID des Server-Scripts

action ... die Aktion, die von der WEB-Client Application ausgeführt werden soll

Die Web-Client Application unterstützt **4** Anfrage-Aktionen (Requests):

„clientOk", „enterPin", „readData" und „writeData".

- Für die „clientOK" Anfrage-Aktion ist die Client-Antwort „clientOk".
- Für die „enterPin" Anfrage-Aktion ist die Client-Antwort „pinEntered".
- Für die „readData" Anfrage-Aktion ist die Client-Antwort „dataRead".
- Für die „writeData" Anfrage-Aktion ist die Client-Antwort „dataWrite".

Die Server-Anfrage muss alle von der Web-Client Application benötigten Informationen enthalten, damit eine plausible Antwort an den Server zurückgeschickt werden kann.

Request «clientOk»

Diese Anfrage ist die 1. Aktion und zeigt, dass der Client mit der Arbeit begonnen hat. Sollte die Web-Client Application nicht funktionieren, erhält der Browser keine Antwort.

In dieser Anfrage sendet das Server-Script **4** Parameter:

`site` - dies ist die Site, auf die eingeloggt werden soll.

`ssid` - ist die Session-ID *„ssid“* des Server-Scripts.

`data1` - muss ein hexadezimaler 16-Byte Block sein (z. B. Zufallszahlen), die vom Server erzeugt wurde.

(16 Bytes hexadezimale Daten = 32 Bytes Text).

`action` - ist der Aktionsname *„clientOk“*.

Anfrage (Request) Beispiel:

```
/?site=www.test.com/logon.php&ssid=4f6de45a15a68059a5eb4639fafe1f06&
data1=223344&action=clientOk
```

Response «clientOk»

Die Anfrage *„clientOk“* wird von der Web-Client Application mit dem gleichen Aktionsnamen *„clientOk“* beantwortet.

In dieser Antwort sendet die Web-Client Application **4** Parameter zurück:

`PHPSESSID` - ist die Session-ID *„ssid“* aus der Anfrage-Aktion *„clientOk“*.

`data1` - die Web-Client Application gibt den unveränderten Datenblock *„data1“* zurück, der vom Server in der Anfrage-Aktion empfangen wurde.

`action` - ist der gleiche Aktions-Name *„clientOk“* wie in der Anfrage-Aktion.

`status` - ist immer **0** in dieser Antwort-Aktion.

Antwort (Response) Beispiel:

```
http://www.test.com/logon.php?PHPSESSID=4f6de45a15a68059a5eb4639f afe1f06&d
ata1=223344&action=clientOk&status=0
```

Bemerkung: Das Server-Script prüft, ob *„data1“* dieselben Werte enthält wie die gesendeten Werte. Dadurch wird vom Server erkannt, ob die WEB-Client Application arbeitet, oder Fehlermeldungen ausgegeben werden sollen.

Request «enterPin»

Dies ist die 2. Anfrage-Aktion des Servers. Mit dieser Aktion wird mitgeteilt, dass die PIN-Code Abfrage, sowie die Erzeugung von verschlüsselten Daten über den Dongle durchgeführt werden soll.

In dieser Anfrage sendet das Server-Script **2** Parameter:

`action` - ist der Aktionsname *„enterPin“*.

`timeout` - ist der Timeout-Wert in Millisekunden für die PIN-Code Anforderung. Erfolgt die PIN-Eingabe nicht in diesem Zeitintervall, dann wird der PIN-Dialog automatisch beendet und keine Antwort zum Server zurückgeschickt.

Anfrage (Request) Beispiel:

```
/?action=enterPin&timeout=25000
```

In diesem Beispiel wartet die WEB-Client Application 25 Sekunden auf eine PIN-Eingabe.

Bemerkung: Das Timeout muss auch auf der Serverseite behandelt werden, da bei einem Timeout keine Antwort von der Web-Client Application kommt.

Response «pinEntered»

Auf die Anfrage *„enterPin“* antwortet die Web-Client Application mit dem Aktions-Namen *„pinEntered“*.

In dieser Antwort sendet die Web-Client Application **5** Parameter zurück:

`PHPSESSID` - ist die Session-ID *„ssid“*, die in der Anfrage-Aktion *„clientOk“* vom Server gesendet wurde.

`data1` - die Web-Client Application gibt den unveränderten Datenblock *„data1“* zurück, der vom Server in der Anfrage-Aktion *„clientOk“* empfangen wurde.

`data2` - die Web-Client Application gibt einen verschlüsselten Datenblock zurück. Dieser Datenblock wurde vom Matrix-Token erzeugt, der am Rechner des Nutzers steckt und repräsentiert den Authentifizierungs-Token des Nutzers.

Die Struktur dieses verschlüsselten Datenblocks ist weiter unten erklärt.

`action` - der Aktionsname dieser Antwort ist *„pinEntered“*.

`status` - der Status-Parameter ist der Return-Code dieser Client-Antwort und kann folgende Werte haben:

- **>0** Seriennummer des angeschlossenen Matrix-Tokens
- **0** PIN-Code Eingabe wurde durch den User abgebrochen
- **-1** Ein Hardware-Kommunikationsfehler ist aufgetreten oder der Token ist nicht vorhanden

Antwort (Response) Beispiel:

```
http://www.test.com/logon.php?PHPSESSID=4f6de45a15a68059a5eb
4639fa fe1f06&data1=223344&data2=0011-2233....4455&action=pinEntered&
status=1234567890
```

Beschreibung des verschlüsselten Datenblocks *„data2“*, der von der Web-Client Application in der Antwort *«pinEntered»* zurückgegeben wird

Der verschlüsselte Datenblock ***„data2“*** besteht aus 6 Segmenten, je 4 Byte Hexadezimal:

Die Segmente 1, 2, 3 und 4 bilden das verschlüsselte Ergebnis von ***"data1"***, das zuvor vom Server gesendet wurde.

Die hexadezimalen 16 Bytes (***„data1“*** aus der ***„clientOk“***-Anfrage) werden in der Web-Client Application in 4 hexadezimale Segmente von je 4 Bytes geteilt und durch den Matrix-Token verschlüsselt.

Diese 4 verschlüsselten Segmente werden von der WEB-Client Application in die Segmente 1, 3, 5, 6 von ***„data2“*** zurückgegeben.

Die Segmente 2 und 4 repräsentieren ein Verschlüsselungsergebnis aus "Token Seriennummer" und „User-PIN“.

Die Server-Anwendung muss den ***„data2“*** Block entschlüsseln und überprüfen. Die Entschlüsselung von ***„data2“*** muss jeweils über die Segment-Paare 1-2, 3-4 und 5-6 durchgeführt werden.

Nach der Entschlüsselung muss folgender Inhalt in den 6 Segmenten enthalten sein:

- Segmente 1, 3, 5, 6 müssen identisch sein mit dem Inhalt von "data1".
- Segment 2 muss die gleiche Seriennummer enthalten, die auch im „status“ emfangen wurde.
- Segment 4 muss den korrekten PIN-Code des Users enthalten.

Um *„data2“* zu entschlüsseln, muss die Server-Anwendung den gleichen Schlüssel besitzen wie der im Matrix-Token.

Dafür wird die Seriennummer des Matrix-Tokens auch in den *„status“* Parameter zurückgegeben. Die Seriennummer ermöglicht die Anbindung an Ihre Kunden-Datenbank in welcher Sie auf dem Server die jeweiligen Schlüssel und PIN-Codes für jeden Matrix-Token/User verwalten.

Mit dieser Seriennummer kann dann der korrespondierende Schlüssel aus der Datenbank ausgelesen werden, um die Entschlüsselung von *„data2“* durchzuführen.

Die Authentizität wird durch den Vergleich der entschlüsselten Daten ermittelt.

Außerhalb der Serveranwendung und außerhalb des Matrix-Tokens ist dieser wichtige Schlüssel (128-Bit-Key) nicht zu sehen und kann deswegen nicht abgefangen und missbraucht werden.

Request «readData»

Diese Anfrage kann benutzt werden, um Daten aus dem Speicher des Matrix-Tokens auszulesen.

In dieser Anfrage sendet das Server-Script **4** Parameter:

`data1` - Der Server sendet in **„data1“** den verschlüsselten UserCode des Matrix-Tokens. Serverseitig muss natürlich derselbe Schlüssel zur Verschlüsselung benutzt werden wie der im Token. Dieser Buffer ist ein hexadezimaler String mit folgendem Format:

R1-UC-R2-UC (Trennzeichen muss „-“ sein)

‚R1‘ und ‚R2‘ sind Zufallszahlen und ‚UC‘ ist der UserCode des Tokens. Die Web-Client Application entschlüsselt den Buffer **„data1“** und überprüft den UserCode.

`fpos` - die Position, ab der aus dem Token-Speicher gelesen werden soll (Nummer der Variable).

`fcnt` - ist die Anzahl von Variablen, die aus dem Token-Speicher gelesen werden sollen. Die maximale Anzahl von Variablen, die in einem Request übertragen werden können, ist 79.

`action` - ist der Aktionsname **„readData“**.

Anfrage (Request) Beispiel:

```
/?fpos=1&data1=6B2AB0D7-BC03A6F0-17F07A20-3C1A0D43&fcnt=3&
action=readData
```

Response «dataRead»

Auf die Anfrage *„readData“* antwortet die Web-Client Application mit dem Aktions-Namen *„dataRead“*.

In dieser Antwort sendet die Web-Client Application **5** Parameter zurück:

`PHPSESSID` - ist die Session-ID ***„ssid“***, die in der Anfrage-Aktion ***„clientOk“*** vom Server gesendet wurde.

`data1` - die Web-Client Application gibt in ***„data1“*** die Variablen zurück, die aus dem Matrix-Dongle gelesen wurden. Die Variablen werden zurückgegeben in einem hexadezimalen String mit dem Format:

oooooooo-oooooooo-....-oooooooo (Trennzeichen muss "-" sein)

`data2` - enthält die Seriennummer des Matrix-Dongle.

`action` - der Aktionsname dieser Antwort ist ***„dataRead“***.

`status` - der Status-Parameter ist der Return-Code dieser Client-Antwort und kann folgende Werte haben:

- **>0** Die Anzahl der Variablen, die aus dem Dongle-Speicher gelesen wurden
- **0** Es wurden keine Daten aus dem Dongle gelesen
- **-1** Ein Hardware-Kommunikationsfehler ist aufgetreten oder der Dongle ist nicht vorhanden
- **-2** Der angegebene UserCode entspricht nicht dem UserCode aus dem Dongle
- **-4** Der Dongle ist gesperrt; die Anti-Hacker-Sperre ist aktiv
- **-5** Der LPT/USB-Port kann nicht belegt werden, da dieser bereits von anderen Geräten belegt ist
- **-6** Beim Zugriff auf den LPT/USB-Port ist ein Fehler aufgetreten

Antwort (Response) Beispiel:

```
http://www.test.com/logon.php?PHPSESSID=4f6de45a15a68059a5eb4639fafe1f06&data1=0A0B0C0D-1A1B1C1D-2A2B2C2D&data2=1234567890&action=dataRead&status=3
```

Request «writeData»

Diese Anfrage kann benutzt werden, um Daten in den Speicher des Matrix-Dongle zu schreiben.

In dieser Anfrage sendet das Server-Script **5** Parameter:

`data1` - der Server sendet in ***„data1"*** den verschlüsselten UserCode des Matrix-Dongle. Serverseitig muss natürlich derselbe Schlüssel zur Verschlüsselung benutzt werden wie der im Dongle. Dieser Buffer ist ein hexadezimaler String mit folgendem Format:

R1-UC-R2-UC (Trennzeichen muss „-" sein)

‚R1' und ‚R2' sind Zufallszahlen und 'UC' ist der UserCode des Dongle. Die Web-Client Application entschlüsselt den Buffer ***„data1"*** und überprüft den UserCode.

`data2` - der Server sendet die in ***„data2"*** die Variablen, die in den Matrix-Dongle gespeichert werden sollen. Die Variablen werden in einem hexadezimalen String gesendet mit dem Format:

00000000-00000000-....-00000000 (Trennzeichen muss „-" sein)

`fpos` - die Position, ab der in den Dongle-Speicher geschrieben werden soll (Nummer der Variable).

`fcnt` - ist die Anzahl von Variablen, die in den Dongle-Speicher geschrieben werden sollen. Die maximale Anzahl von Variablen, die in einem Request übertragen werden können, ist 79.

`action` - ist der Aktionsname ***„writeData"***.

Anfrage (Request) Beispiel:

```
/?data1=6B2AB0D7-BC03A6F0-17F07A20-3C1A0D43&data2=0A0B0C0D-1A1B1C1D-
2A2B2C2D&fpos=1& fcnt=3&action=writeData
```

Response «dataWrite»

Auf die Anfrage ***„writeData“*** antwortet die Web-Client Application mit dem Aktions-Namen ***„dataWrite“***.

In dieser Antwort sendet die Web-Client Application **3** Parameter zurück:

`PHPSESSID` - ist die Session-ID ***„ssid“***, die in der Anfrage-Aktion ***„clientOk“*** vom Server gesendet wurde.

`action` - der Aktionsname dieser Antwort ist ***„dataWrite“***.

`status` - der Status-Parameter ist der Return-Code dieser Client-Antwort und kann folgende Werte haben:

- **>0** Die Anzahl der Variablen, die in den Dongle-Speicher geschrieben wurden
- **0** Es wurden keine Daten in den Dongle gespeichert
- **-1** Ein Hardware-Kommunikationsfehler ist aufgetreten oder der Dongle ist nicht vorhanden
- **-2** Der angegebene UserCode entspricht nicht dem UserCode aus dem Dongle
- **-4** Der Dongle ist gesperrt; die Anti-Hacker-Sperre ist aktiv
- **-5** Der LPT/USB-Port kann nicht belegt werden, da dieser bereits von anderen Geräten belegt ist
- **-6** Beim Zugriff auf den LPT/USB-Port ist ein Fehler aufgetreten

Antwort (Response) Beispiel:

```
http://www.test.com/logon.php?PHPSESSID=4f6de45a15a68059a5eb4639fafe1f06&action=dataWrite&status=3
```

WWW.TDI-MATRIX.DE