Anwenderdoku Dynamic Screenlibrary mit Template Applikation für DC/DS 14/16/24:

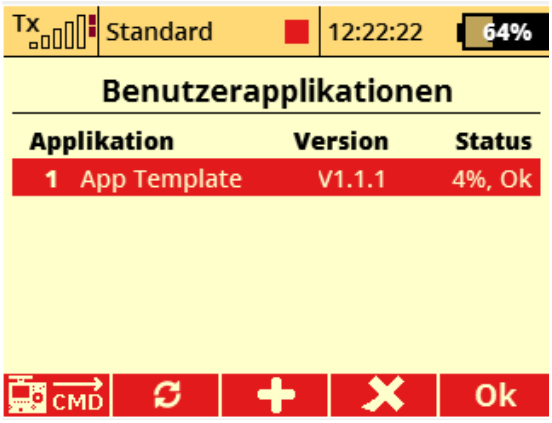
Die Applikation ermöglicht dem Anwender ohne Kenntnisse der Skriptsprache Lua eigene Full - Screen- Telemetrieseiten zu entwerfen. Auf bis zwei Telemetrieseiten verteilt werden unterschiedliche Anzeigeelemente zur Darstellung von Sensordaten angeordnet. Eine Bindung aller von Jeti unterstützter Sensoren ist möglich. Für Elektro – Verbrenner – und Turbinenmodelle sind unterschiedliche Anzeigen zu Akkukapazität, Tankfüllstand und Turbinenstatus vorgesehen. Für alle Anzeigen können optional Limits definiert werden, die entsprechend optisch hervorgehoben und zusätzlich akustisch ausgegeben werden. Weiterhin sind 4 freie Softwaretimer implementiert, die im count up \ count down und Anzeigemodi Stunde, Minute, Sekunde sowie Minute, Sekunde und Zehntelsekunde konfigurierbar sind. Alle Timerwerte werden bei Modellwechsel und Abschaltung des Senders modellspezifisch gespeichert.

1. Installation:

Zur Installation das vom GitHub geladene Zipfile in ein temporäres Verzeichnis auf dem PC entpacken, den Sender mittels USB Kabel verbinden und die AppTemplate_Setup.bat ausführen. Das Installations- batch AppTemplate_Setup startet zunächst mit einer Eingabeaufforderung für das USB Laufwerk des Senders. Hier nur den Laufwerksbuchstaben eingeben, z.B E oder F je nach verbundenem Laufwerk des Senders und bestätigen. Danach folgt eine weitere Eingabeaufforderung mit Frage, ob ein teilweises Backup des Senders angelegt werden soll. Dies wird empfohlen mit J bzw. Y und Bestätigung auszuführen. Das teilweise Backup wird im Installationsverzeichnis auf dem PC in ein Verzeichnis generiert aus Datum_partial kopiert und enthält den aktuellen Stand der Verzeichnisse Apps, Audio und Model vor Installation der AppTemplate Screenlibrary. Nach erfolgtem Backup wird die Screenlibrary App auf dem Sender installiert. Mit ausgegebener Komandozeile ‚installation successful finished' ist die Installation abgeschlossen. Bricht die Installation mit Fehler ab, wurde der Laufwerksbuchstabe des Senders falsch angegeben. Eine Installation über Appmanager ist auch möglich.

2. Registrierung:

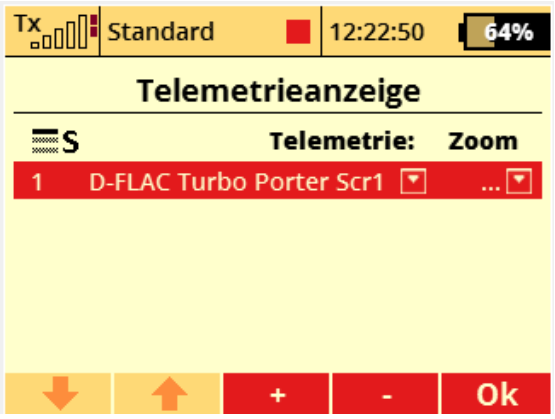
Unter *Hauptmenü/Zusatzfunktionen/Benutzerapplikationen* mittels *Key3 (+)* die App Template wählen und mit *OK* bestätigen. Bei den 14 –er und 16 –er Sendern sind alle bereits registrierten Apps (auch leere Apps) aus der Liste zu entfernen. Es darf aufgrund von Speicherlimit bei diesen Sendern nur die App Template aktiv sein!!!



3. Aktivierung der Telemetrieseite:

Dies ist erst möglich, wenn das Modell bereits Telemetriesensoren registriert hat. Dazu Sender und Empfänger des Modells mit allen vorgesehenen Sensoren einschalten. Nach einmaligem Einschalten des Empfängers mit Sensoren, sind diese in den Modelldaten gespeichert. Es genügt dann zur weiteren Bindung nur den Sender einzuschalten.

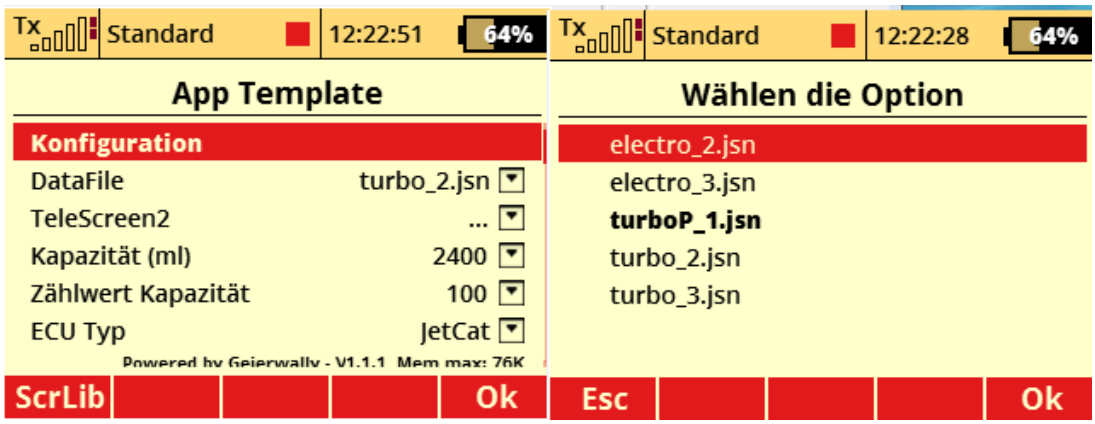
Unter *Hauptmenü/Stoppuhren/Sensoren/Telemetrieanzeige* mittels *Key3 (+)* das Telemetriefenster der App wählen. Name des Fensters setzt sich zusammen aus Modellname + SCR1. Auch hier darf aufgrund Speicherlimits bei den 14 –er und 16 –er Sendern nur diese eine Seite aktiviert sein.



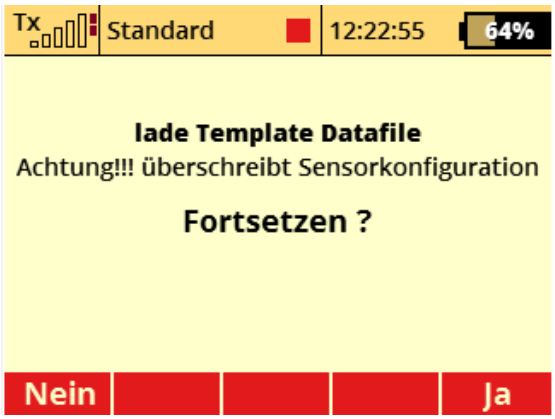
4. Datafile laden:

Die Anordnung der Anzeigeelemente wird in jsn – files gespeichert, die vom Anwender mittels Notepad++ (bitte nur diesen Editor verwenden) zu bearbeiten sind. Die Zuordnung der Anzeigeelemente im Datafile ist im Kapitel *Aufbau Datafile* dieser Anleitung beschrieben. Als Referenz für eigene Telemetrieseiten stehen dem Anwender 5 Datafile – Templates zur Verfügung, die ohne Änderung verwendet, bzw. anwenderspezifisch angepasst werden können. Sensorzuordnungen und Datafiles werden modellspezifisch gespeichert.

Zur Auswahl des Datafiles ins *Hauptmenü /AppTemplate* wechseln und das entsprechende *DataFile* wählen.



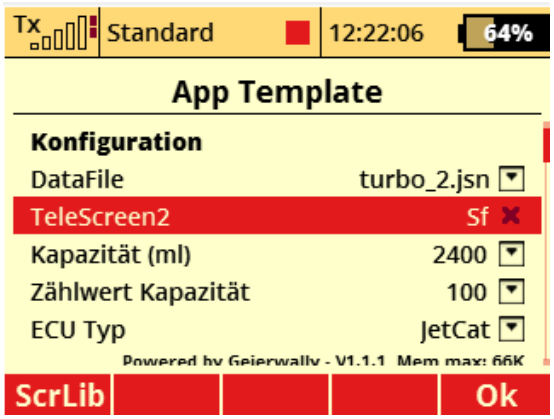
turboP_1.jsn ist ein Template für Turbinenmodelle mit nur einer Telemetrieseite
turboP_2.jsn ist ein Template für Turbinenmodelle aufgeteilt auf zwei Telemetrieseiten mit einem kleinen Modellfoto. (Zuordnung von Modellfotos funktioniert nur auf den 24-er Sendern)
turboP_3.jsn ist ein Template für Turbinenmodelle aufgeteilt auf zwei Telemetrieseiten mit einem großen Modellfoto. (Zuordnung von Modellfotos funktioniert nur auf den 24-er Sendern)*electro_2.jsn* sowie *electro_3.jsn* sind zweiseitige Templates für Elektromodelle.
Nach Auswahl des Datafiles folgt die Optionsbox mit Warnung, dass die aktuelle Sensorzuweisung mit Laden des Datafiles überschrieben wird. Dies bitte mit *Ja* bestätigen.



5. Schalter zur Umschaltung der Telemetrieseite konfigurieren (steht nur bei Doppelseiten zur Konfiguration):

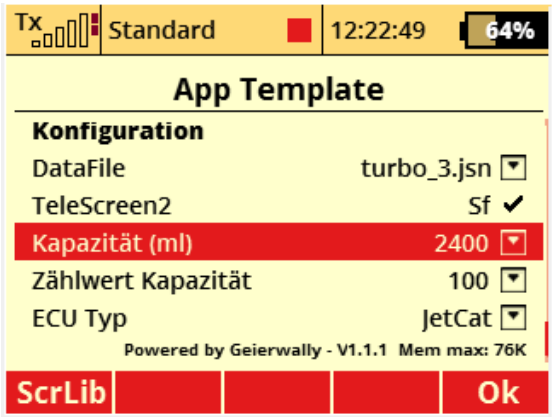
Eine Konfiguration von zwei getrennten Telemetrieseiten ist aus Speichergründen auf den 14/16-er Sendern nicht möglich. Aus diesem Grund wurde das Konzept über Schalter implementiert. Das hat auch den Vorteil, dass man z.B. zwei Seiten in

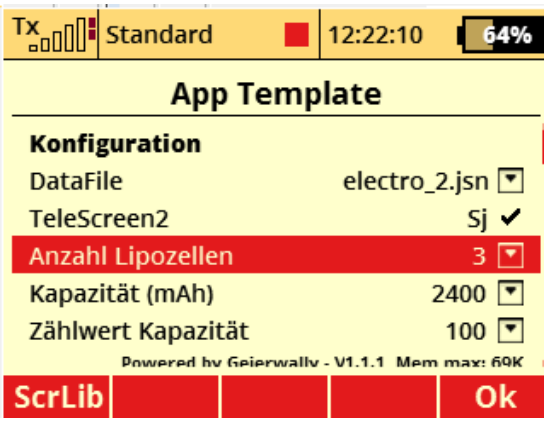
Abhängigkeit des Flugzustandes schalten kann. Zum Wechsel zwischen den Telemetrieseiten kann auch ein logischer Schalter konfiguriert werden.



6. Typspezifische Konfiguration:

Die folgenden Konfigurationen sind abhängig vom im Datafile konfigurierten Modelltyp Elektro, Verbrenner oder Turbine. Beim Elektromodell folgt die *Anzahl der Lipozellen*. Diese kann je nach Wunsch ob man die umgerechnete Einzelzellspannung auswerten möchte oder die Gesamtspannung auf *Zellenzahl* bzw. für Gesamtspannung auf *1* konfiguriert. Wird Einzelzellüberwachung als Telemetriesensor gebunden, welcher die niedrigste Zellspannung zurückgibt, ist die *Anzahl der Lipozellen* ebenfalls mit *1* zu konfigurieren.
Kapazität entspricht der *Tankfüllmenge* bei Verbrenner - bzw. der *Akkukapazität* bei Elektromodellen.
Zählwert Kapazität definiert den Zählwert womit *Kapazität* bei Auswahl inkrementiert oder dekrementiert wird und ist als Eingabeerleichterung für große Tank- bzw. Akkukapazitäten gedacht.
ECU Typ steht nur für Turbinenmodelle zur Konfiguration. Hier wählt man die entsprechende Turbine für die Statusanzeige aus.

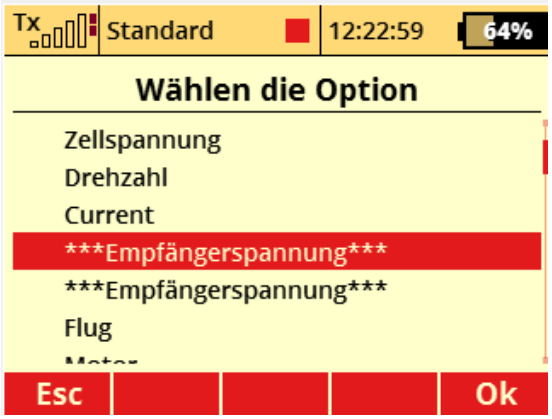




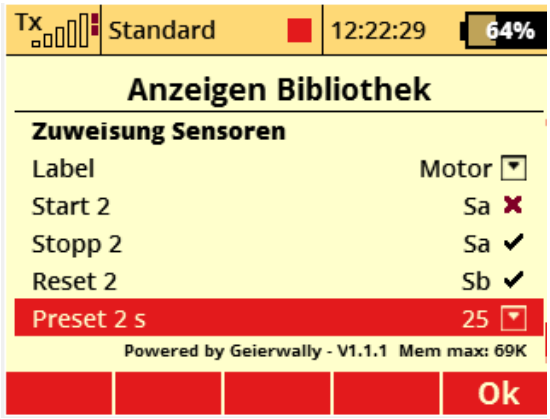
7. Bindung der Rahmen an Sensoren, Timer oder Modellfotos :

Dazu in mittels *Key1* in die *ScrLib* Seite wechseln und jedem Rahmenlabel den entsprechenden Sensor und Sensorparameter zuweisen. Bei Erstinbetriebnahme ist zum Einlesen der Sensoren erforderlich, den Empfänger mit verbundenen Sensoren einzuschalten.

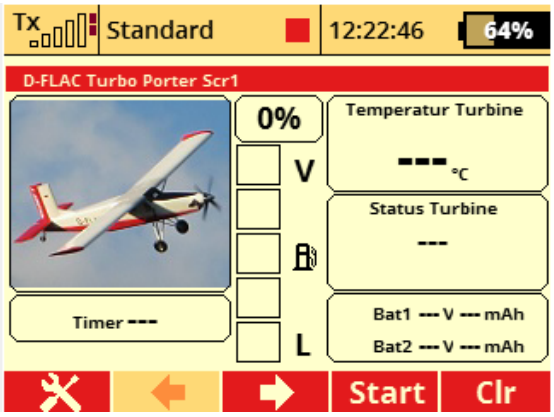
Für Rahmenlabel mit folgender Definition ****Name**** entfällt die Sensorzuweisung, da diese Rahmen bereits in der Anwendung gebunden wurden.



Wurden *Timer* im Datafile als Rahmenlabel definiert, wechselt die Konfiguration entsprechend von Sensor auf Timerparameter. Hier sind jeweils ein Start-, Stopp- und Resetschalter sowie der Timerwert je nach Datafile - Definition in Sekunden oder Minuten zu konfigurieren. Die Einheit wird nach Preset und der TimerID in *s* bzw. *min* angezeigt. Definition des Timertyps *count up / count down* sowie *Minuten* oder *Sekunden* sind nur im Datafile möglich und werden im Kapitel *Aufbau Datafile*beschrieben. Die Konfiguration eines *Stopp* Schalters kann wegfallen. In dem Fall läuft der Timer nur wenn Schalter *Start* aktiv ist. Bis zu 4 Timer können im Datafile definiert werden. Timerwerte werden bei Abschaltung des Senders bzw. Modellwechsel modellspezifisch gespeichert. Der *Preset Wert* entspricht bei Typ *count down* dem Startwert des Timers bzw. bei *count up* dem Endwert. Erreichen des Endwertes *count up* bzw. 0 bei *count down* werden optisch sowie akustisch ausgegeben.



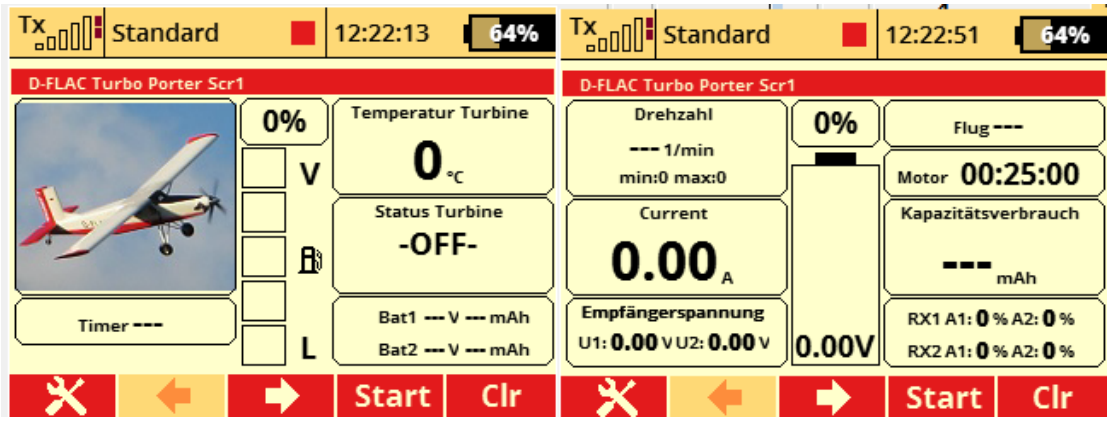
Zuweisung Modellfoto: Dieses Feature steht nur für 24-er Sender zur Verfügung. Bei Definition Modellfoto im Datafile, wechselt Die Parameterauswahl auf Auswahl des Modellfotos. Weitere Beschreibung zu diesem Feature erfolgt im Kapitel *Aufbau Datafile*



8. Prüfung der Telemetriehauptseiten:

Wurden alle Sensoren, Timer und Modellfotos gebunden, erfolgt die Prüfung der Telemetrieseiten.

Bei allen Rahmen mit fehlender Zuweisung von Sensoren, Timern bzw. Fotos wird der Wert --- angezeigt. Diese Rahmen sind in der Konfiguration noch entsprechend Typs zu binden. Fehlt die Bindung der Kapazitäts und Spannungs – Rahmen, wird das Akkusymbol bei Elektro sowie die Füllstandsanzeige bei Typ Verbrenner nicht angezeigt.



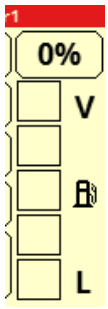
9. Aufbau Datafile:

Die verwendeten Frames sind wie folgt definiert:

Mittelteil Tankanzeige:

```
Main Turbine:
[1] 3                                   Modelltyp 3    (1 Elektro  2 Verbrenner 3 Turbine , 4 Segler)
[2] "Tankinhalt                         Framelabel, Text wird in der Konfig der Sensorzuweisung aufgelistet
[3] "ml"                                Einheit ist ml , wird im Frame dargestellt
[4] 1                                   Wert wird in Templateapp in % zur Füllstandsanzeige umgerechnet
[5] 30                                  Ab 30% erfolgt Fehlermeldung
[6] 100                                 Bei 100 % (voll getankt) wird Fehler wieder gelöscht
[7] 0                                   Platzhalter Anzahl der Nachkommastellen  (wird hier nicht verwendet)
[8] 0                                   von Screenlib gesetzt aktueller Wert
[9] 0                                   von Screenlib gesetzt wenn 1, ist Alarm aktiv
[10]1                                   von Screenlib gesetzt zugewiesene SensorID
[11]1                                   von Screenlib gesetzt zugewiesener Sensor Parameter
[12]0                                   von Screenlib gesetzt (Platzhalter für spätere Anwendung)
Beispiel nach obiger Beschreibung:
[3,"Tankinhalt","ml",1,30,100,0,0,0,0,0,0]
```

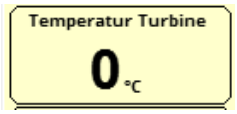
[3,"Tankinhalt","ml",1,30,100,0,0,0,0,0,0]



Frametype 1 Normalfenster :

```
Normalfenster
[1] 1                                   Frametype 1 (1 Normalfenster, 2 MinMax, 3 geteiltes Fenster, 4 Absc
[2] "Temperatur Turbine"                Framelabel, Text wird in der Konfig der Sensorzuweisung aufgelistet
[3] "°C"                                Einheit ist °C , wird im Frame dargestellt
[4] 0                                   Wert kommt direkt vom Telemetriesensor, wenn > 0 dann von der App
[5] 0                                   Alarmwert in dem Fall kein Alarm
[6] 0                                   Alarmwert gelöscht, in dem Fall ist kein Alarm konfiguriert
[7] 0                                   Anzahl der Nachkommastellen
[8] 0                                   von Screenlib gesetzt aktueller Wert
[9] 0                                   von Screenlib gesetzt wenn 1, ist Alarm aktiv
[10]0                                   von Screenlib gesetzt zugewiesene SensorID
[11]0                                   von Screenlib gesetzt zugewiesener Sensor Parameter
[12]0                                   von Screenlib gesetzt (Platzhalter für spätere Anwendung)
Beispiel nach obiger Beschreibung:
[1,"Temperatur Turbine","°C",0,0,0,0,0,0,0,0,0],
```

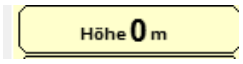
[1,"Temperatur Turbine","°C",0,0,0,0,0,0,0,0,0],



Frametype 3 geteiltes Fenster :

```
Geteilte Fenster
[1] 3                                   Frametype 3 (1 Normalfenster, 2 MinMax, 3 geteiltes Fenster, 4 Ab
[2] "Höhe"                              Framelabel, Text wird in der Konfig der Sensorzuweisung aufgelist
[3] "m"                                 Einheit ist m , wird im Frame dargestellt
[4] 0                                   Wert kommt direkt vom Telemetriesensor, wenn > 0 dann von der App
[5] 0                                   Alarmwert in dem Fall kein Alarm
[6] 0                                   Alarmwert gelöscht, in dem Fall ist kein Alarm konfiguriert
[7] 0                                   Anzahl der Nachkommastellen
[8] 0                                   von Screenlib gesetzt aktueller Wert
[9] 0                                   von Screenlib gesetzt wenn 1, ist Alarm aktiv
[10]0                                   von Screenlib gesetzt zugewiesene SensorID
[11]0                                   von Screenlib gesetzt zugewiesener Sensor Parameter
[12]0                                   von Screenlib gesetzt (Platzhalter für spätere Anwendung)
Beispiel nach obiger Beschreibung:
[3,"Höhe","m",0,0,0,0,0,0,0,0,0],
```

[3,"Höhe","m",0,0,0,0,0,0,0,0,0],



Frametype 5, Abschluss mit 4 Anzeigewerten:

Jedem Frame ist in der scrlib config ein Sensorparameter zuzuordnen!!!

```
Abschlussfenster 5 Werte es sind vier Zeilen zu konfigurieren, welche sich nur im Fensterwert unterscheiden
[1] 5                                   Frametype 5 , 4 Anzeigewerte
[2] "Bat1"                              Framelabel Zeile 1 ist Bat1
[3] "V"                                 Einheit ist Volt
[4] 0                                   Wert kommt direkt vom Telemetriesensor
[5] 0                                   Alarmwert kein Alarm konfiguriert
[6] 0                                   Alarm zurückgesetzt , in dem Fall kein Alarm
[7] 2                                   Anzahl der Nachkommastellen für Spannung 2 , für Strom 0
[8] 0                                   von Screenlib gesetzt aktueller Sensorwert
[9] 0                                   von Screenlib gesetzt wenn 1 Alarm aktiv
[10]1                                   von Screenlib gesetzt SensorID
[11]1                                   von Screenlib gesetzt Sensor Parameter
[12]0                                   vorbereitet für Outputcontrol
[13]1                                   Framewert 1   (bei diesem Frametyp von 1 - 4)
[14]""                                  optional Label für jeden Wert , hier aus Platzgründen nicht definiert

Beispiel nach obiger Beschreibung:
[5,"Bat1","V",0,0,0,2,0,0,1,1,0,1,""],
[5,"Bat1","mAh",0,0,0,0,0,0,1,1,0,2,""],
[5,"Bat2","V",0,0,0,2,0,0,1,1,0,3,""],
[5,"Bat2","mAh",0,0,0,0,0,0,1,1,0,4,""],
```

[5,"Bat1","V",0,0,0,2,0,0,1,1,0,1,""],
[5,"Bat1","mAh",0,0,0,0,0,0,1,1,0,2,""],
[5,"Bat2","V",0,0,0,2,0,0,1,1,0,3,""],
[5,"Bat2","mAh",0,0,0,0,0,0,1,1,0,4,""],