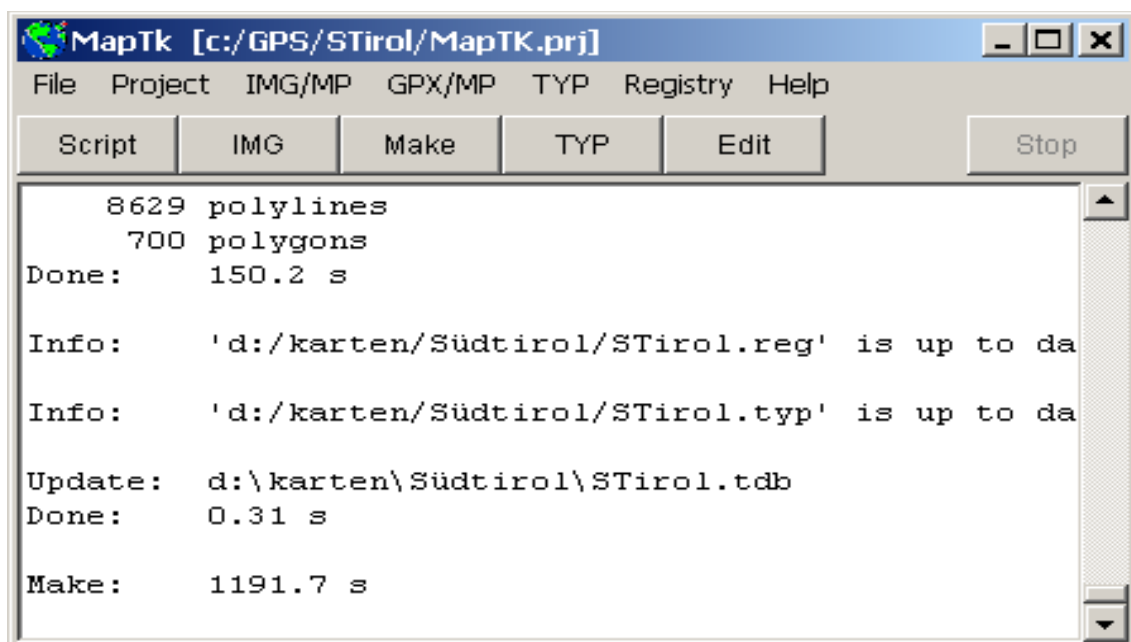


MapTk

Map Toolkit

Version 3.2



2012-03-26

Inhaltsverzeichnis

1	Übersicht.....	4
1.1	Installation und Aufruf	4
1.2	Startparameter.....	5
1.3	Funktionen.....	5
1.4	Kompatibilität der Karten.....	6
2	Menü und Buttons.....	8
2.1	Menü.....	8
2.2	Buttons.....	16
3	Funktionen.....	18
3.1	'Script'.....	18
3.2	'IMG'.....	20
3.3	'Make'.....	21
3.4	'TYP'.....	22
3.5	Übersichtskarte.....	23
3.6	Export in GPSMapEdit.....	23
4	Automatische Routen-Berechnung.....	25
4.1	Straßen und Wege.....	25
4.2	Karte bearbeiten.....	26
4.3	Routen-Probleme beheben.....	27
5	Editor.....	29
5.1	Neue PRJ-Datei.....	29
5.2	Header.....	30
5.3	Maps.....	31
5.4	Auswahlfenster.....	31
5.5	POI.....	33
5.6	Polyline.....	35
5.7	Polygon.....	37
5.8	Script.....	39
6	Dateien.....	42
6.1	MapTk.dat.....	42
6.2	PRJ-Datei.....	42
6.3	PRJ-Datei in den Funktionen.....	46
6.4	MP-Datei.....	48
6.5	IMG-Datei.....	50
6.6	TYP-Datei.....	51
6.7	TDB-Datei.....	51
6.8	Index.....	52
6.9	LOG-Datei.....	52
7	Python-Script.....	53
7.1	Übersicht.....	53
7.2	Daten des Headers.....	54
7.3	Daten der Objekte.....	54
7.4	Etwas Python.....	55
8	Tipps & Tricks.....	58
8.1	Zusammenhang TYP und IMG.....	58
8.2	Verwendung der Level.....	58
8.3	Sehr große Karten.....	60
8.4	Komplizierte Linien und Polygone.....	60
8.5	Verzeichnisse.....	60
8.6	Arbeiten mit vielen Dateien.....	61

8.7	Alle Dateien in einem Verzeichnis.....	61
8.8	Script.....	61
9	Anhang.....	65
9.1	Beispiel-Karte.....	65
9.2	TYP / Symbol.....	65
9.3	Benutzer definierte Typen.....	68
9.4	'Skin' für GPSMapEdit.....	71
9.5	Farbpalette (256 Farben).....	73
9.6	Sprach-Codes.....	73

1 Übersicht

Dieses Dokument beschreibt die Funktionen des Programms MapTk (Map Toolkit). Diese Beschreibung ist keine Einführung in das Erstellen und Bearbeiten von Kartensätzen. Es wird vorausgesetzt, dass der Benutzer des Programm mit allen Schritten der Kartenerstellung vertraut ist. Ebenso erforderlich sind fundierte PC-Kenntnisse.

Der Autor übernimmt keinerlei Haftung für Schäden, die durch die Benutzung des Programms entstehen könnten. Das gilt auch für mögliche Verletzungen des Urheberrechts durch die Verwendung des Programms.

Die Sprache des Programms ist Englisch. Das Englisch ist sehr einfach gehalten - solange nicht Meldungen vom Laufzeitsystem Python durchschlagen – und sollte auch Ungeübten keine Probleme bereiten.

Download bei <http://maptk.dnsalias.com>. Für die Bearbeitung des Kartenmaterials wird dringend [GPSMapEdit](#) empfohlen.

1.1 Installation und Aufruf

1.1.1 Windows

Es sind für die Ausführung von MapTk keine externen Programme erforderlich. Das Programm wird mit einem Inno-Setup-Script installiert durch Start der Datei 'MapTk <Version> Windows.exe'. Nur die Installation in das Verzeichnis 'Programme' erfordert die Rechte eines Administrators (unter Vista muss das Programm als Administrator gestartet werden). Es gibt keine Registry-Einträge für das Programm. Entwickelt wird das Programm mit Python 2.5 unter Windows 7 64-Bit. Die Verwendung von Python sollte sicherstellen, dass andere Windows-Versionen keine Probleme bereiten. Bei Bedarf wird eine leere Icon-Bibliothek 'MapTk.dat' mit vorgegebenen Einstellungen erzeugt.

Wenigstens zwei Arten zu starten:

1. Die Endung '.prj' kann mit MapTk.exe verknüpft werden. MapTk startet dann mit Doppelklick auf eine PRJ-Datei mit dieser Datei als aktuelles Projekt. Diese muss im Verzeichnis der MP-Quelldateien stehen.
2. Alternativ wird MapTk aus dem Verzeichnis der Quelldateien der Karten aufgerufen. Die PRJ-Datei ist dann MapTk.prj. Dazu legt man optimal im Verzeichnis der Quelldateien einen Link auf das Programm an. 'Ausführen in' enthält entweder den aktuellen Pfad, '.' oder bleibt leer.

Die für einen Kartensatz zentrale Projektdatei '*.prj' steuert das Programm. Diese Datei **muss** im Verzeichnis der MP-Quelldateien des Kartensatzes stehen. Die Daten des Projektes stehen in diesem Verzeichnis, die von MapSource verwendeten Dateien (IMG, TDB, TYP) meist in einem anderen.

Wenn PDF-Dateien keinem Viewer zugeordnet sind kann in der Konfiguration 'File → Preferences' ein Viewers bestimmt werden.

1.1.2 Andere Plattformen

Unter z.B. Ubuntu kann MapTk benutzt werden. In diesem Fall muss Python in der Version 2.5, 2.6 oder 2.7 und Tkinter (tk8.5 / tcl8.5) installiert sein. MapTk mit Python 3 ist nicht möglich wegen inkompatibler Sourcen. Wohl alle Linux-Distributionen haben Python 2 installiert. Um Bitmap-Icons importieren zu können muss die Python-Imaging-Library (PIL) installiert sein.

Alle Dateien des Programms ('MapTk.pyw', '*.pyc', die Hilfe-Dateien und 'globus.ico') werden in ein frei wählbares Verzeichnis kopiert durch Entpacken der ZIP-Datei. Die Zip-Datei muss natürlich zur installierten Python-Version passen. Wenn die Iconlibrary 'MapTk.dat' fehlt wird eine leere Bibliothek mit Voreinstellungen erzeugt.

Eine für ein Projekt zentrale Datei 'MapTk.prj' steuert das Programm. Diese Datei muss im Verzeichnis der Quelldateien (*.mp) des Kartensatzes stehen. Das Programm wird aus diesem Verzeichnis heraus gestartet:

```
python <Pfad zu den Python-Dateien>/MapTk.pyw
```

Die normalerweise von MapSource oder Sendmap benutzten Dateien stehen meist in einem anderen Verzeichnis.

Wenn PDF-Dateien keinem Viewer zugeordnet sind kann in der Konfiguration 'File → Preferences' ein Viewers bestimmt werden.

1.2 Startparameter

Parameter beim Aufruf:

- '-make <file>': MapTk wird mit <file> als aktuelle Projektdatei ausgeführt und aktualisiert sofort das Projekt (Make). Eine reduziertes Protokoll wird nach 'MapTk.log' geschrieben. MapTk beendet sich automatisch.
- '-script <file>': MapTk wird mit <file> als aktuelle Projektdatei ausgeführt und bearbeitet alle MP-Dateien des Projektes mit der Funktion 'Script'. Eine reduziertes Protokoll wird nach 'MapTk.log' geschrieben. MapTk beendet sich automatisch.
- '*<project>.prj': MapTk wird mit dieser Datei als die aktuelle Projektdatei gestartet.
- '<map>.mp': Diese MP-Datei wird in eine IMG-Datei compiliert.
- '<file1> <file2>': Die MP-Datei file1 wird in die IMG-Datei file2 compiliert.
- 4 Parameter, die von GPSMapEdit zum Export in eine IMG-Datei benutzt werden.

1.3 Funktionen

Die wichtigsten Funktionen des Programms im Überblick:

1. Mit der Funktion 'Script' können an Karten und Sätzen von Karten globale Änderungen vorgenommen werden wie z.B. Umbenennen oder entfernen von POIs nach bestimmten Kriterien, Ändern des Zoom-Level, ...
'Script' ist ein Pre-Prozessor für die 'IMG'-Funktion.

2. Kompilieren von einzelnen oder mehrerer Quelldateien zu Garmin-Image Dateien ('IMG'), einschließlich der 3-Byte-Typen. Die ist der traditionelle Stil. Karten in diesem Stil können mit Garmins MapConverter in das GMAP-Format übertragen werden.
3. Daten für die automatische Berechnung von Routen können optional erzeugt werden. MapTk setzt auf die Aufbereitung des Straßennetzes von GPSMapEdit auf. Siehe Seite 26.
4. Aus den IMG-Dateien wird mit 'Make' automatisch ein Index für POIs (MDR-Datei *_mdr.img) erzeugt. Nach Adressen kann nicht gesucht werden.
5. Erstellen eines Projektes, d.h. Kompilieren aller geänderten Quellen, falls notwendig Erzeugen der TDB-, REG-, MDX-, MDR- und TYP-Dateien ('Make').
6. Bearbeiten von TYP-Dateien mit einem grafisch orientierten Editor. Siehe Seite 30. Listen von benutzten Typen in MP-Dateien oder installierten Kartensätzen.
7. TYP-Datei aus Projektdatei erzeugen ('TYP').
8. Erzeugen einer Übersichtskarte aus einer oder mehreren Detailkarten. Siehe Seite 24.
9. Registry-Eintrag als REG-Datei zur Anmeldung in MapSource erzeugen ('Make'). Datensicherung von Registry-Einträgen und entfernen einzelner Kartensätze aus der Registry.
10. Erstellen und Bearbeiten einer TYP-Datei ('TYP'), auch für fertige Kartensätze wie z.B. 'City Select' oder 'Topo Deutschland V2'.
11. Analyse von IMG-, TDB- und TYP-Dateien
12. 'XOR' mit 0x96 einer IMG-Datei und zurück ('XOR')
13. Transparent-Flag in IMG-Dateien setzen
14. Lesen und Schreiben des 'DrawPriority'-Bytes von IMG-Dateien.
15. Funktionen zur Bearbeitung von GPX-Dateien. Siehe Seite 11.
16. Kompatibel mit GPSMapEdit. Export aus GPSMapEdit (als Ersatz für cgpsmapper.exe) und Erzeugen eine 'Skin'-Datei.

Namenskonventionen: Namen für Detailkarten sind numerisch (z.B. '01234567.img'); für die Übersichtskarte mit Buchstaben beginnend bis zu 8 Zeichen, sonst beliebig (z.B. 'abc2.img'). Die Namen der MP-Dateien sind beliebig. Der Name der Übersichtskarte sollte keine Leerzeichen enthalten. Dieses Verhalten kann im Kopf der MP-Datei überschrieben werden. Andere Dateien werden automatisch benannt.

Es ist unbedingt zu empfehlen für jeden Kartensatz ein eigenes Verzeichnis für die Quellen anzulegen und MapTk aus diesen Verzeichnis heraus zu starten (Link anlegen). Die für MapSource erzeugten Dateien müssen ebenfalls pro Kartensatz ein eigenes Verzeichnis bekommen oder im Verzeichnis der Quelldateien stehen.

1.4 Kompatibilität der Karten

Die erzeugten IMG-Dateien sind kompatibel mit Garmin-Geräten. Inkompatibilitäten sind nicht bekannt geworden. 3-Byte-Typen werden nur mit neueren Geräten angezeigt (ab ca. Ende 2005).

MapSource zeigt Produkte mit TYP-Datei schon auf dem PC richtig an. TYP-Dateien für z.B. 'CitySelect' können erzeugt werden. Die mit MapTk generierten Kartensätze und TYP-Dateien werden mit MapSource ganz normal auf das GPS-Gerät übertragen, auch gemischt z.B. 'CitySelect' mit einer topografischen Karte.

Die TYP-Dateien sind geeignet für 'CitySelect', 'MetroGuide', 'CityNavigator' und topografische Karten.

Marinekarten werden nicht unterstützt.

Karten mit Kopierschutz ('locked') werden nicht unterstützt.

2 Menü und Buttons

2.1 Menü

2.1.1 'File'

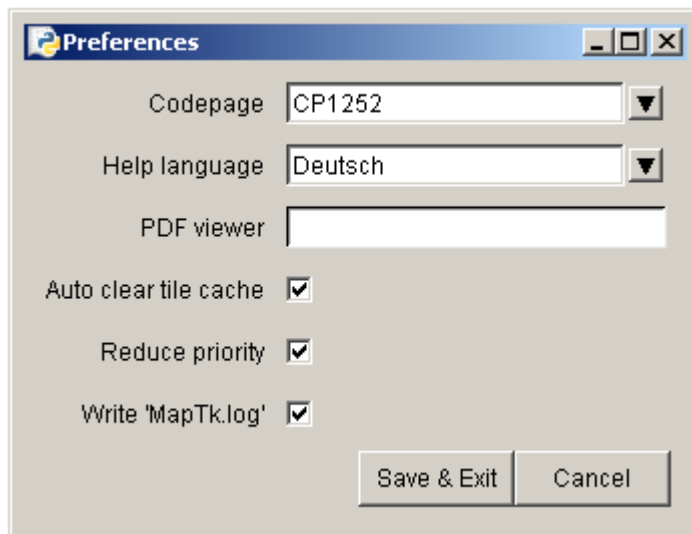
- 'Edit text file' bearbeitet eine wählbare Textdatei. Schlüsselworte werden nicht hervorgehoben.
- 'Clear cache': Löscht den Speicher für Kacheln von MapSource, Basecamp und MapInstall
- 'Rename GPX to time stamp': GPX-Dateien aus unterschiedlichen Quellen haben unterschiedliche Namen die meist Datum und Zeit enthalten. Sie lassen sich nicht ordentlich sortieren. Diese Funktion vereinheitlicht die Dateinamen auf das Format 'YYYY-MM-DD hh:mm:ss' und erlaubt damit das Sortieren Dateien mit folgenden

Formaten werden umbenannt:

```
YYYY-MM-DD hh.mm.ss Day
Day DD-MMM-YY hh.mm.ss
Track_DD-MMM-YY hh.mm.ss
Track_DD-MMM-YY hhmmss
Track_YYMMDD hhmmss
YYYY-MM-DD <digits>
MMM-DD-YY hhmmss
DD-MMM-YY hhmmss
DD-MMM-YY hh mm ss
YYYYMMDD hhmmss
YYYYMMDD-<digits>
```

Das Jahr 'YYYY' muss ≥ 2000 sein. 'MMM' ist die Abkürzung eines Monats (deutsch, englisch). Bei Dateien nach diesem Schema wird in der Datei der älteste Zeitstempel <time> gesucht. Zeitstempel in GPX-Dateien sind bezogen auf UTC (früher GMT). Die Zeit wird auf die Zeitzone des PC umgerechnet und als Dateiname verwendet. Dateien ohne Zeitstempel werden auf Basis des alten Dateinamen umbenannt. Der Inhalt der Datei bleibt unverändert. Dateien mit Namen die nicht in dieses Schema passen werden nicht umbenannt.

- 'Preferences':



- Definition der Codepage (CP1251, CP1252).
- Sprache der Hilfe (deutsch, englisch).
- Pfad zu einem PDF-Viewer. Wenn ein PDF-Viewer hier nicht definiert wird benutzt das Programm unter Windows den Standard-Viewer und unter Linux 'kpdf' oder 'evince'.
- Nach Erzeugen einer TYP-Datei kann der Cache von MapSource (ab Version 6.15), Basecamp und MapInstall automatisch gelöscht werden um die neu erzeugte TYP-Datei darzustellen (nur Windows).
- Für länger dauernde Funktionen kann die Priorität des Prozesses reduziert werden (nur Windows).
- Eine kompakte Log-Funktion wird durch Aktivieren von „Write 'MapTk.log'“ eingeschaltet (siehe Seite 53).
- 'Exit' beendet das Programm. Wenn noch Fenster offen sind oder Prozesse bearbeitet werden erfolgt eine Nachfrage. Bei Abbruch werden Änderungen an einer Datei im Editor werden verworfen.

2.1.2 'Project'

- 'Change working directory' wechselt das Projektverzeichnis (Das aktuelle Projektverzeichnis wird in der Titelleiste des Programms angezeigt [...]).
- 'Change directory + PRJ file' wechselt das Projektverzeichnis und die Projektdatei (Wird in der Titelleiste angezeigt [...]).
- 'New project file' erstellt eine Vorlage der Projektdatei 'MapTk.prj'. Bevor der Editor gestartet wird erscheint eine Dialogbox in der die Kopfdaten in [Project] eingegeben werden und bestimmt wird und ob Vorschläge für TYP-Dateien und die Scripte in die PRJ-Datei kopiert werden sollen. Details siehe Seite 30.

- 'Edit project file' bearbeitet eine wählbare PRJ-Datei. Schlüsselworte im Script-Teil werden farbig hervorgehoben.
- 'Build project' (= Button 'Make' ohne Abfrage der Projektdatei) erstellt anhand einer PRJ-Datei einen neuen Kartensatz. Siehe Seite 22.

2.1.3 'IMG/MP'

- 'Reorganize' reorganisiert eine MP-Datei. Linien und Flächen mit mehr als 256 Punkten werden automatisch geteilt. Ausnahme: Straßen müssen manuell geteilt werden nach Fehlermeldung im Compiler. Die beim Teilen oder Trimmen einer Karte entstehenden doppelten Data-Definitionen werden aufgelöst. Data in Ebenen > 0 wird entfernt. Hinzu kommen Funktionen zur Kompatibilität mit älterer Syntax von MP-Dateien.
- 'Script' (= Button 'Script' ohne Abfrage der Projektdatei) reorganisiert MP-Dateien wie in der Funktion 'Reorganize'. Zusätzlich können Veränderungen der Karte vorgenommen werden durch 4 Python-Skripte, die in einer PRJ-Projektdatei gespeichert sind. Siehe Button 'Script' Seite 19.
- 'Compile map' (= Button 'IMG') erzeugt aus MP-Dateien IMG-Dateien. Siehe Seite 21.
- 'IMG analysis': erzeugt aus einer IMG-Datei eine MP-Datei. Diese Datei enthält die Objekte aller Level.
Der Name der IMG-Datei wird als ID in die MP-Datei übernommen. Der Name der erzeugten MP-Datei wird aus der Beschreibung der Karte abgeleitet (in der MP-Datei: 'Name=...') oder aus der gegebenenfalls vorhandenen Datei 'tdb.dict'.
Eine IMG-Datei kann auch sehr gut mit GPSMapEdit in das MP-Format umgewandelt werden.
Karten die 'locked' sind können nicht analysiert werden. Kopierschutz !
- 'TDB analysis': schreibt Informationen aus einer TDB-Datei in eine Textdatei. Das sind z.B. Copyrights, Namen der Detailkarten, ID, Position, Größen, ... Außerdem wird eine Datei 'tdb.dict' angelegt. Diese Datei enthält eine Liste aller Kacheln im Format <ID> : <Name>. Beim Decompilieren von IMG-Datei wird die Liste zur Benennung der MP-Datei verwendet – sofern vorhanden.
- 'XOR IMG file': kodiert eine IMG-Datei mit 0x96 oder macht eine Kodierung rückgängig.
- 'IMG set file transparent': setzt das Transparent-Bit im TRE-Subfile (auch bei mit XOR kodierten Dateien). Die mit IMG erzeugten Karten lassen sich hier undurchsichtig machen (alternativ: in der PRJ-Datei). 'IMG reset file transparent' löscht das Transparent-Bit.

- 'IMG get 'DrawPriority' zeigt die Zeichenreihenfolge der Karten an. Mit 'IMG set 'DrawPriority' kann die Reihenfolge im Bereich 1 ... 31 gesetzt werden. Die Funktion ist anwendbar auf alle Karten, auch 'gelockte'.
- 'Create overview map': Siehe Seite 24.

2.1.4 'GPX/MP'

Dieser Menüpunkt bietet einige Funktionen zur Bearbeitung von GPX-Dateien. Manche Funktionen benötigen viel Rechenzeit. Z.B. 4 Mbytes Tracks in einer oder mehreren GPX-Dateien zu kombinieren kann leicht eine Stunde dauern.

- 'MP to GPX':
Alle Linien ausgewählter Polyline Typen werden als Tracks in eine GPX-Dateien konvertiert. Die Typen sind als Liste anzugeben. Beispiel: '1-12,0x16' steht für alle Straßen und Wege. Die Typen und Gruppen sind durch Komma zu trennen. Die Schreibweise ist dezimal oder hexadezimal. Der Label in der MP-Datei wird der Name des Tracks¹. Im Tag 'cmt' (Kommentar) der GPX-Datei² wird der Typ der Linie in der Form 'Type=0x1234' kodiert.

Für POIs wird der Eintrag um die Angabe eines Symbols erweitert. Das Symbol wird aus dem Typ des POI abgeleitet (siehe Seite 67). Routen werden durchgereicht.

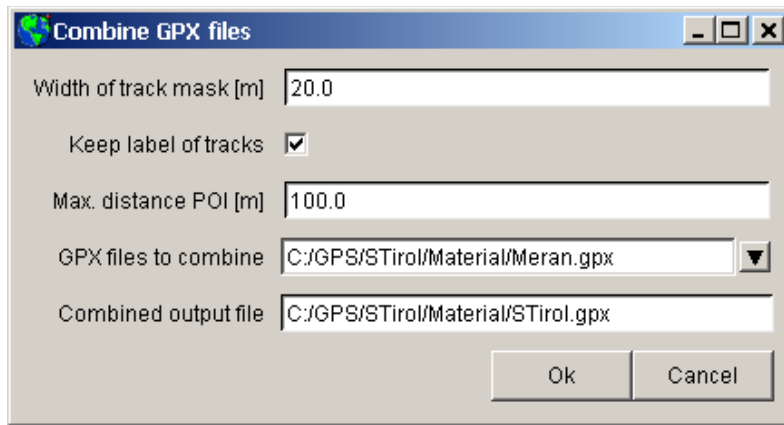
Der Name der MP-Datei(en) wird im Feld 'MP files to convert' angegeben. Die GPX-Datei wird benannt nach der MP-Datei.

- 'Split GPX':
Eine Tracks und Wegpunkte enthaltende GPX-Datei wird geteilt in Dateien, die jeweils nur Tracks, die nur Wegpunkte und nur die Routen enthalten. Trackpunkte enthalten nur die Position, Höhe und einen Zeitstempel. Die Funktion kann benutzt werden um zunächst nicht von BaseCamp oder MapSource lesbare GPX-Dateien kompatibel zu machen.
- 'Combine GPX files':
Diese Funktion kombiniert eine oder mehrere GPX-Dateien ('GPX files to combine') und deren Inhalt zu einer neuen Datei ('Combined output file').

Tracks innerhalb einer Maske mit der Breite 'Width of track mask [m]' werden durch Mittelwertbildung zu einer neuen Linie zusammengefasst.

1 Hat die Linie keinen Label, wird der Track Track_n genannt. n ist dabei eine automatisch erzeugte Nummer.

2 Vorsicht: MapSource zeigt das Kommentarfeld nicht an und schreibt es beim Speichern auch nicht zurück in die Datei. Linien ohne Typangabe bekommen bei weiterer Verarbeitung mit MapTk den Typ der Wege 0x16.



Das funktioniert nur gut bei Tracks guter Qualität. Die Tracks sollten zuvor von groben Fehlern befreit werden, als da sind Ausreißer oder 'Knäuel' während einer Pause. Die Breite der Maske liegt etwa im Bereich von 10 bis 30 m und hängt vom Inhalt der Dateien ab. Eine zu enge Maske fasst ggf. zusammengehörige Tracks nicht zusammen. Eine zu weite Maske fasst Tracks zusammen, die nicht zusammengehören. Je besser die Qualität der Tracks ist, desto enger darf die Maske sein und die Qualität des Ergebnisses steigt. Sehr gute Tracks sollten nicht mit sehr schlechten Tracks kombiniert werden, das Ergebnis wird nur mittelmäßig sein. Ist 'Keep label of tracks' gesetzt werden die Namen der zusammengefassten Tracks auch im Ergebnis kombiniert. Nur Tracks mit gleichem Typ (im Tag 'cmt') werden zusammengefasst.

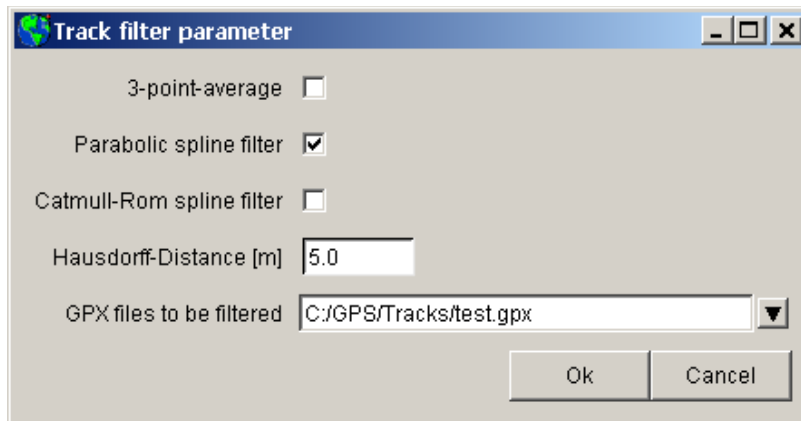
Höhe und Zeitstempel der Trackpunkte gehen wegen möglicher Zusammenfassung von Tracks verloren.

POIs können ebenfalls zusammengefasst wenn sie innerhalb eines Radius von 'Max. distancePOI [m]' liegen. Zusätzlich ist es erforderlich, dass die Symbole der POI gleich sind und der Name ähnlich ist. Die Namen sind ähnlich genug nach phonetischem Vergleich (Soundex) bei einer Levenshtein-Distance von 0.

Routen werden durchgereicht.

- 'GPX track filter':

Alle Tracks in der Datei zur Verschönerung werden gefiltert bevor sie in eine Karte eingebaut werden. Anfangs- und Endpunkte werden niemals verändert. Punkte, die mit anderen Tracks einer Datei verbunden sind werden ebenfalls nicht verändert. Der Name der erzeugten Datei wird aus dem Namen der zu filternden Datei abgeleitet durch den die Namenserweiterung '_filter'. Es gibt 4 verschiedene Filter:



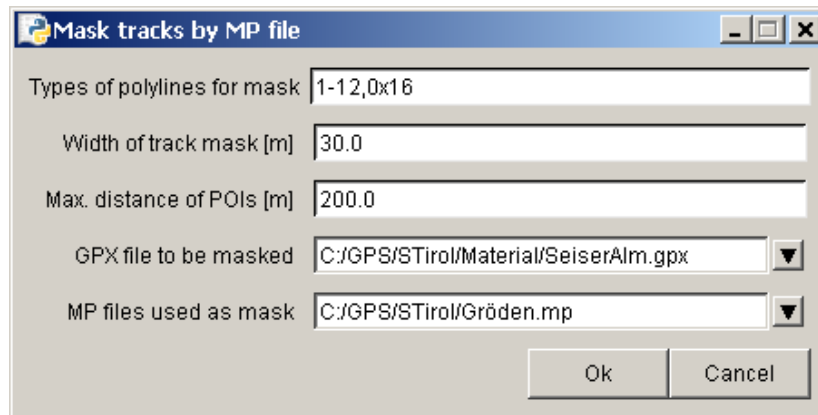
- '3-point-average': Aus einem Punkt und den zwei benachbarten Punkten wird das geometrisch Mittel berechnet. Dieses Filter wird empfohlen für geradere Tracks mit schlechter Qualität.
- 'Parabolic spline filter': Es wird eine Linie berechnet, die glatter ist als der ursprüngliche Track. Die berechnete Linie wird nicht durch alle Punkte des Tracks gehen. Dieses Filter ist geeignet zum Verschönern von Tracks ohne scharfe Änderungen der Richtung. Die Zahl der Punkte wird verzehnfacht.
- 'Catmull-Rom spline filter': Die berechnete Linie geht durch alle Punkte eines Tracks. Dieses Filter sollte benutzt werden wenn der Track scharf seine Richtung ändert, wie bei Serpentinaen. Die Zahl der Punkte wird verzehnfacht.
- Die 'Hausdorff-Distance' wird benutzt für ein Douglas-Peucker-Filter. Dieses Filter reduziert die Anzahl der Punkte. Alle Punkte liegen innerhalb des spezifizierten Abstands, genannt Hausdorff-Distance. Ein typischer Abstand ist 3.0 m. Hausdorff-Distance = 0.0 m schaltet das Filter ab.

Die Spline-Filter erzeugen 10 mal mehr Punkte als im ursprünglichen Track vorhanden. Sie sollten deshalb unbedingt nur zusammen mit dem Douglas-Peucker-Filter benutzt werden. Nur das als letztes berechnete Douglas-Peucker-Filter kann zusammen mit den anderen Filtern kombiniert werden.

Welches Filter günstig ist für einen Track sollte man ausprobieren.

Wegpunkte und Routen werden durchgereicht.

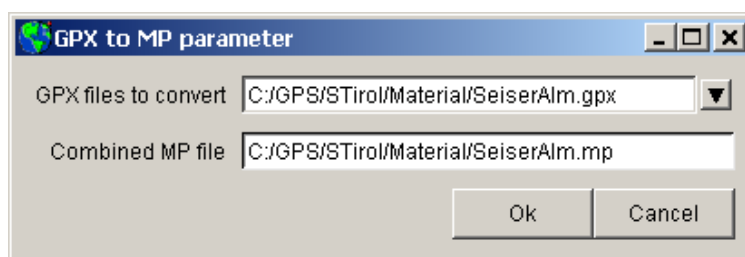
- 'Mask GPX tracks by MP':
Diese Funktion bereitet Tracks in einer GPX-Datei zur Übernahmen in eine Karte vor. Um diese Funktion sinnvoll nutzen zu können sind Tracks guter Qualität erforderlich.



Aus den in 'Types of polylines for mask' angegebenen Typen wird eine Maske mit der Breite 'Width of track mask [m]' angelegt. Die Maske wird erzeugt aus den Polylinien der Dateien in 'MP files used as mask'. Alle Tracks der GPS-Dateien werden gelöscht sofern sie innerhalb dieser Maske liegen. Trifft ein Track auf eine der Polylinien wird der Schnittpunkt auf die Polylinie verschoben. Das Ergebnis wird in GPX-Dateien gespeichert, deren Name in 'GPX file to be masked' angegeben ist und um '_mask' ergänzt wird.

POIs und Routen werden durchgereicht.

- 'GPX track to MP': Alle Tracks werden zu Linien des in den GPX-Dateien hinterlegtem Typ konvertiert. Der Label der Polylinie wird vom Track übernommen. Kein Label wird in die MP-Datei geschrieben wenn der Name des Tracks automatisch erzeugt wurde. Die Definition der Level im Kopf der MP-Datei bleibt leer damit die Datei ohne Probleme in jede andere MP-Datei integriert werden kann.

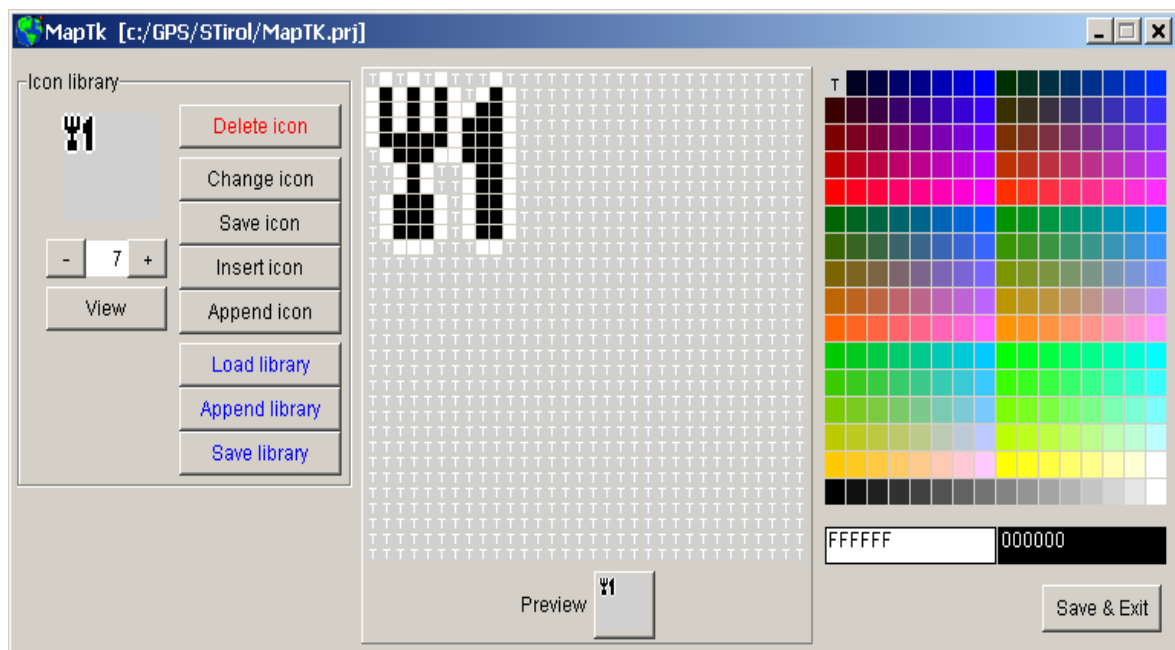


- 'Find not connected near roads'
In den ausgewählten Dateien werden alle unverbundenen Knotenpunkte in der Nähe einer Straße / Weg gesucht. Die Suchentfernung wird in 'Search for roads around nodes about [m]' in Metern eingegeben. Das Suchergebnis wird nach '#<source>.mp' geschrieben. Die Datei enthält POIs 'Type=0x00' für alle Knotenpunkte in der Nähe einer Straße / Weg. Code für die für GPSSMapEdit-Skin-Datei siehe Seite 73.
Optional können innerhalb der Suchentfernung liegende Endpunkte und Knotenpunkte automatisch miteinander verbunden werden.

- 'Find not matching external nodes'
Diese Funktion überprüft das ganze Projekt auf fehlende oder falsch positionierte Verbindungen zwischen den einzelnen Kacheln. Es wird eine MP-Datei mit dem Namen '#<FID>.mp' geschrieben nachdem alle IMG-Dateien des compilierten Projektes gelesen wurden. Für jeden nicht passenden externen Knotenpunkt enthält die Datei ein POI 'Type=0x00'. Code für die für GPSMapEdit-Skin-Datei siehe Seite 73.

2.1.5 'Typ'

- 'Icon library': Der Editor für die Icon-Bibliothek ist dem Editor für POI ähnlich (Seite 34).



Das Konzept der Benutzung ist entweder eine einzige Bibliothek (laden und speichern zusammen mit MapTk) oder eine projektspezifische Bibliothek ('Load library' and 'Save library' verwenden). Als Name für spezifische Bibliotheken wird 'MapTk.dat' empfohlen

Die importierte Datei kann eine Icon-Bibliothek oder ein Rasterbild in einem der folgenden Formate enthalten: BMP, GIF, ICO, JPG, PNG oder TIF. 'Import' hängt alle noch nicht vorhandenen Icons an die aktuelle Bibliothek an. Mehrere Datei können ausgewählt werden. Die aktuelle Bibliothek im Speicher wird in 'MapTk.dat' beim beenden von MapTk gespeichert.

- 'Compile TYP' (= Button 'TYP' ohne Abfrage der Projektdatei) erstellt aus deiner PRJ-Datei eine TYP-Datei. Siehe Seite 23.
- 'TYP analysis': erzeugt eine PRJ-Datei, die verändert und wieder in eine TYP-Datei konvertiert werden kann.
Einschränkungen: Dateien mit 2-Zeichen-Farbdefinitionen und Nachtfarben werden nicht unterstützt.

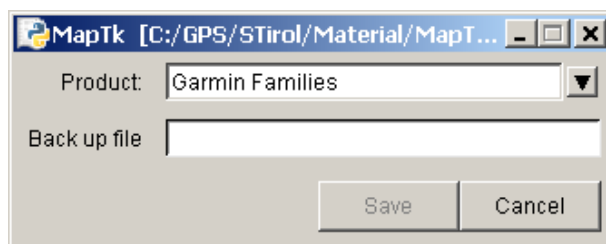
- 'List of used types': erzeugt eine Liste aller Typen der POIs, Linien und Flächen eines Kartensatzes. Die betreffenden MP-Dateien werden aus dem Block [IMG] der PRJ-Datei entnommen.
- 'PRJ file to GPSMapEdit skin' erstellt aus einer wählbaren PRJ-Datei eine Textdatei die in GPSMapEdit die Anzeige entsprechend der TYP-Datei verändert. Das funktioniert ab GPSMapEdit Version 1.0.48.0³. Die erzeugte Datei kann in GPSMapEdit unter 'View → Manage Map Skins...' eingebunden und unter 'View → MapSkin' aktiviert werden. Die Datei ist unabhängig von ProductID und FamilyID. Nur die 'Day'-Version sollte verwendet werden.

2.1.6 Registry

Nur für Windows:

Bei doppelten Family-IDs in der Registry wird immer gewarnt wenn die Registry ausgelesen wird.

- 'Backup registry':
Der Zweig 'Families' oder einzelne Kartensätze werden in einer REG-Datei gesichert.



³ Linien mit Mustern werden nicht korrekt dargestellt.

- 'Delete from registry':
Einzelne Registry-Einträge können gelöscht werden. Die Dateien des Kartensatzes werden nicht gelöscht.



2.1.7 'Help'

- 'PDF-Help' zeigt diese Hilfe-Datei (MapTk_*.pdf) im PDF-Format in einem eigenen Prozess an. Die Datei muss im Verzeichnis des Programms MapTk stehen.
- 'Homepage' zeigt die Homepage von MapTk im Standard-Browser an.
- 'Info' zeigt die Version von MapTk.

2.2 Buttons

2.2.1 Script

Ausführen eines Scripts, basierend auf der PRJ-Datei im Arbeitsverzeichnis. Siehe Seite 19.

2.2.2 IMG

Compilieren von einer oder mehreren MP-Dateien zu IMG-Dateien. Siehe Seite 21.

2.2.3 Make

Alles notwendige ausführen um eine vollständigen Kartensatz zu bekommen, inklusive TYP-, Index- und REG-Dateien. Basiert auf der PRJ-Datei im Arbeitsverzeichnis. Siehe Seite 22.

2.2.4 TYP

Erzeugen ein TYP-Datei aus den Daten der PRJ-Datei. Siehe Seite 23.

2.2.5 Edit

Erzeugen oder Bearbeiten einer PRJ-Datei mit speziellen Editoren für die verschiedenen Teile der Datei. Siehe Seite 30.

2.2.6 Stop

Wenn der Button aktiviert ist kann eine Bearbeitung abgebrochen werden falls mehrere Operationen erforderlich sind (z.B. 'Make'). Die Warteschlange für Aufträge wird gelöscht. Die Zahl neben 'Stop' zeigt die Anzahl Prozesse:

- 0 kein Prozess aktiv
- 1 ein Auftrag wird gerade bearbeitet
- 2 + ein oder mehrere Aufträge warten auf Bearbeitung

3 Funktionen

Die Funktionen werden durch Klick auf den entsprechenden Button oder über das Menü gestartet. Bei direktem Aufruf der Funktion über den Button wird die Projektdatei im aktuellen Projektverzeichnis herangezogen. Bei Start über das Menü kann in vielen Fällen die Projektdatei ausgewählt werden.

3.1 'Script'

Da Karten für einen Kartensatz oft aus unterschiedlichen Quellen stammen, Fehler und überflüssige Informationen enthalten ist es sinnvoll neue Karten zu reorganisieren. Mit 'Script' erhalten die Karten ein einheitliches Erscheinungsbild, z.B. Wege (Type=0x16) nur im Level 0, Straßen (Type=0x06) bis Level 1, ...). Das Erscheinungsbild einzelner Objekte ist durch Python-Skripte (in einer PRJ-Datei) an die persönlichen Wünsche anzupassen. Siehe Seite 54. Wegen der großen Datenmengen ist ein Script eine wesentliche Arbeitserleichterung.

Der Compiler ('IMG') wertet nur Daten im Level 0 aus und erwartet zum Zoomen die 'EndLevel=' bzw. 'Levels=' Zeile für alle Objekte die nicht nur im Level 0 sichtbar sein sollen. Wenn MP-Dateien alle Objekte einzeln in den verschiedenen Ebenen enthalten **muss** deshalb ein Script verwendet werden (oder mit GPSMapEdit 'Join per-level elements'). Da die Funktion 'IMG analysis' ab Version 1.3 mit 'IMG' kompatible MP-Dateien erzeugt, ist dieser Schritt nicht mehr erforderlich wenn eine MP-Datei mit 'IMG analysis' erzeugt wurde. Eine PRJ-Datei für vier Level kann im Menü 'File → New project file' erzeugt werden (MapTk.prj). Als Muster für fünf Level steht die PRJ-Datei 'Level5.prj' zum Download bereit. Beide Dateien sind auf die persönlichen Bedürfnisse und Gegebenheiten anzupassen.

Die Zeilen 'ID=' und 'Name=' müssen in der zu bearbeitenden MP-Datei angegeben sein. Sonst:

```
Error:      'ID'           oder
Error:      'Name'
```

und Abbruch.

Ist die ID selbst nicht angegeben wird bei Detailkarten '00000001' eingesetzt, bei Übersichtskarten der Name der MP-Datei. 'Name' darf leer bleiben. Es wird eine Warnmeldung angezeigt. Der Name der Übersichtskarte sollte keine Leerzeichen enthalten.

Zu den Funktionen siehe auch Beispiel PRJ-Datei auf Seite Fehler: Referenz nicht gefunden.

Die von 'Script' vorgenommene Reorganisation betrifft folgende Punkte:

- Die Ausgabedatei hat im Block [IMG ID] immer neben 'ID=' und 'Name=':
 - 'Elevation=M'
 - 'LblCoding=9'
 - 'CodePage=...' (wenn die Codepage nicht CP1252 ist)
 - 'DrawPriority=...' (optional, Vorgabe: 24)
 - 'Transparent=...' (optional)
 - 'Preview=' (optional)

- 'Routing=' (optional)
- 'Copyright=...' (optional)
- 'Levels=...'
- 'Levelx=...'
- 'Zoomx=...' (optional)

Es sind genau so viele Zeilen 'Levelx=' erforderlich wie in 'Levels=..' angegeben wurde (x steht für den Level 0 ... Levels-1). Bis auf 'ID', 'Name', 'Transparent', 'Copyright', 'Codepage', 'Levels', 'Levelx' und 'Zoomx' sind alle anderen Angaben, wie sie von z.B. GPSMapEdit hinzugefügt werden, ohne Bedeutung für die spätere Konversion in eine IMG-Datei und werden nicht übernommen. Die Einträge 'Transparent', 'Copyright', 'Codepage', 'DrawPriority', 'Levels' und alle 'Levelx' / 'Zoomx' können im Script-Teil der PRJ-Datei [CUSTOM_HEADER] gelesen und verändert werden. Die Werte für 'Zoom' werden optional aus der MP-Datei übernommen. Sonst: 'Zoom0=0', 'Zoom1=1', - aber erst beim Kompilieren in eine IMG-Datei.

- Orte in [RGN20] werden nach [POI] verschoben und 'City=Y' wird gesetzt wenn der Ort einen Namen hat (label=ps['cityname']).
- Alle Orte in [RGN10] bekommen die Zeile 'City=Y'. Falls vorhanden werden aus CityIdx und [Countries] die Zeilen CityName, RegionName und CountryName erstellt.
- Aus bestehenden Index-Blöcken [Countries], [Regions] und [Cities] werden für POIs mit CityIdx die entsprechenden Adressdaten erzeugt. Die Index-Blöcke werden nicht mehr in die MP-Datei geschrieben.
- Nur Koordinaten in 'Origin0' oder 'Data0' werden übernommen. Objekte, die kein 'Origin0' oder 'Data0' haben werden ignoriert. Zeilen mit 'Data1' und größeren Ebenen werden ignoriert. Die Koordinaten für Level 1 und höher werden bei der Erzeugung der IMG-Datei später aus dem Level 0 erzeugt.
- Einige Kennzeichnungen (z.B. Autobahn) für Straßen werden auf die in GPSMapEdit genannten und von MapTk akzeptierten Steuerzeichen (z.B. '~[0x04]') umkodiert.
- Die Namen der Blöcke werden übersetzt:
[RGN10] bzw. [RGN20] nach [POI], [RGN40] nach [POLYLINE] und [RGN80] nach [POLYGON].
- Kommentare (';' ... ') werden nicht übernommen, Anhänge (';@ ... ') werden übernommen jedoch nicht ausgewertet.
- Linien und Flächen mit mehreren Zeilen 'Data0' bzw. 'Origin0' werden in mehrere Objekte mit sonst gleichen Daten geteilt (1 pro 'Data0' / 'Origin0').
- Linien und Flächen mit mehreren mehr als 255 Knoten werden in mehrere Objekte mit sonst gleichen Daten geteilt.
- Beim Bearbeiten von Karten bleiben oft 'Routing nodes' zurück, die keine Funktion haben. Die Routing-Information wird entfernt wenn der Punkt keine Wege verbindet und auch keine Abbiegerestriktionen trägt.
- Höhen- oder Tiefenangaben werden ggf. von Feet nach Meter umgerechnet.

- Objekte mit 'Levels' < 0 werden ignoriert, d.h. nicht in die Ausgabedatei übernommen (im Script auf -1 gesetzt entfernt diese Objekte).
- Alle Änderungen aus dem Script-Teil der PRJ-Datei werden übernommen.

Mehrfache Bearbeitung einer Karte mit 'Script' ändert den Inhalt nicht. Beim Speichern wird die ursprüngliche Datei nicht überschrieben sondern in *.BAK umbenannt.

Die PRJ-Datei darf die Blöcke [...] auch nur teilweise enthalten oder sogar leer sein. 'Script' kann sogar ohne eine PRJ-Datei ausgeführt werden. In diesem Fall werden die folgenden Schritte ausgeführt:

- Alle Daten in Level >= 1 werden entfernt.
- Orte in [RGN20] werden nach [POI] verschoben wenn der Typ <= 0x1100 ist.
- Alle Orte in [RGN10] bekommen die Zeile 'City=Y'. Falls vorhanden werden aus CityIdx und [Countries] die Zeilen CityName, RegionName und CountryName erstellt.

3.2 'IMG'

Diese Funktion erzeugt aus einer MP-Datei direkt eine IMG-Datei. Umkodierungen sind nicht erforderlich. Weitere Dateien wie die PRJ-Datei sind nicht erforderlich. Die IMG-Datei kann in einen Kartensatz übernommen werden oder eine bestehende Kachel ersetzen. 'ID=' und 'Name=' müssen in der MP-Datei angegeben sein. Der Name der erzeugten IMG-Datei ist immer die ID + '.img'.

Sowohl Detailkarten als auch Übersichtskarten können erzeugt werden. Die Unterscheidung erfolgt durch die ID in der MP-Datei. Die Anzahl der Level ist 0 bis 8. Eine Übersichtskarte hat im Level 0 sinnvollerweise für Bits / Koordinate den Wert des höchsten Level der Detailkarten. Ein zusätzlicher Level vergrößert die IMG-Dateien deutlich.

Karten dürfen im Subfile RGN bis 4 MByte groß sein. Wird diese Grenze überschritten gibt es eine Fehlermeldung:

Abort: map is too complex (RGN)!

und die Funktion wird abgebrochen. Das ergibt IMG-Dateien bis etwas über 4 Mbyte die aus MP-Dateien von etwa 20 MByte erzeugt werden ('topologischer' Inhalt). Größere Karten müssen geteilt werden, z.B. mit GPSMapEdit.

Nur die Koordinaten mit dem Schlüssel 'Data0' / 'Origin0' werden in die IMG-Datei übernommen. Maximal 8 Level stehen zur Verfügung. Der höchste Level ist durch 'Levels=' definiert. Die Koordinaten der Level 1 bis Levels-1 werden aus 'Data0' durch Reduktion der Bits / Koordinate abgeleitet. Polygone und Linien werden mit der Douglas-Peucker-Funktion vereinfacht um die Dateigröße zu reduzieren.

Ein Hintergrund (Type=0x4b) ist nicht in jedem Fall erforderlich, für routingfähige Karten aber zwingend erforderlich ! Bei Detailkarten sollte ein Hintergrund vorhanden sein um in MapSource die Grenzen der Kacheln anzuzeigen. Der Hintergrund darf geteilt sein und muss auch nicht mehr rechteckig sein.⁴

⁴ Bekommt der Hintergrund (Polygon 0x4b) bei transparenten Karten eine Farbe, deckt er darunter liegende Karten vollständig ab. Es gibt dann keine doppelten Straßen ! Ist die abgedeckte Karte routing-fähig kann mit der Anzeige einer Topo-Karte die Routing-Funktion

Eine bereits vorhandene IMG-Datei wird überschrieben.

Die IMG-Dateien sind mindestens mit eMap, GPSMap60C(x), Colorado und Oregon kompatibel. Gründe für mögliche Inkompatibilität mit anderen Geräten sind nicht bekannt.

Die erzeugten IMG-Datei sind im Vergleich zu MapDekode etwa gleich groß. Die Geschwindigkeit ist etwa 60 kByte/s (bezogen auf die MP-Datei, 3 GHz CPU, 1 Gbyte Speicher).

3.3 'Make'

Mit dieser Funktion wird die Aktualisierung eines Kartensatzes automatisiert. Die Steuerung wird durch die PRJ-Datei 'MapTk.prj' vorgenommen. 'Make' erledigt folgende Aufgaben nacheinander:

- Falls eine MP-Datei aus dem Block [IMG] älter ist als die zugehörige IMG-Datei - oder wenn die IMG-Datei nicht existiert - wird kompiliert. Um eine neue Kompilation zu erzwingen können die IMG-Dateien im Projektverzeichnis einfach gelöscht werden.
- Nach erfolgreicher Kompilation wird die IMG-Datei in das Verzeichnis 'IMGpath' kopiert. Schlägt das Kopieren fehl, weil z.B. die Kachel in MapSource geöffnet ist, kann später durch erneutes 'Make' das Kopieren erneut versucht werden.
- Nachdem alle Kacheln aktualisiert sind werden die TYP- und REG-Dateien erzeugt. Siehe Seite 23. Nur Windows: Die Einträge der REG-Datei und die Registry werden verglichen und ggf. ein Information angezeigt. Wenn der Name des Kartensatzes ('Product name') fehlt wird keine REG-Datei erzeugt.
- Es wird die TDB-Datei erzeugt wenn die Projektdatei oder eine der IMG-Dateien im Block [IMG] der Projektdatei jünger ist als die vorhandene TDB-Datei oder keine TDB-Datei gefunden wird. Zusammen mit der TDB-Datei wird eine MDX-Datei für den Index erzeugt.
- Zum Schluss wird eine Index-Datei (*_mdr.img) aus allen Detailkarten erzeugt. Die Detailkarten müssen mit MapTk ab Version 2.7 compiliert worden sein. Die Erzeugung eines Index ist optional.

Wenn der Kartensatz bereits in die Registry eingetragen ist, kann er sofort in MapSource verwendet werden. Ansonsten kann mit der gerade erzeugten REG-Datei die Eintragung in die Registry erfolgen: Doppelklick auf die REG-Datei.

Scheinbar werden die Einträge unter Windows-32-Bit und Windows-64-Bit in zwei verschiedenen Zweigen der Registry gespeichert. Das ist aber nur die Sichtweise von 32- oder 64-Bit-Programmen.

Ein 32-Bit-Programm (wie MapTk) liest und schreibt

HKEY_LOCAL_MACHINE\SOFTWARE\Garmin\MapSource\Families*

verwendet werden. Allerdings wird die berechnete Route bei eingeschalteter Topo-Karte in älteren Geräten (z.B. GPSMap60C) nicht hervorgehoben. Zudem wird selbst die Übersichtskarte immer abgedeckt.

Ein 64-Bit-Programm liest und schreibt

\\HKEY_LOCAL_MACHINE\\SOFTWARE\\Wow6432Node\\Garmin\\MapSource\\Families*

3.4 'TYP'

Eine TYP-Datei wird auf Basis einer PRJ-Datei erzeugt (siehe Seite 43). Farben für ein Nacht-Design und Farbkodierung mit 2 Zeichen werden nicht unterstützt. Wird die TYP-Datei nicht in die Registry eingetragen werden Karten mit den Vorgaben von MapSource bzw. des Gerätes dargestellt. Die Darstellung in GPSMapEdit kann mit einer 'Skin'-Datei entsprechend der TYP-Datei angepasst werden (siehe Seite 16).

Eine Registry-Datei mit den Daten des Projektes wird in das Verzeichnis der TYP-Datei geschrieben. Ist die PRJ-Datei nur zur Erzeugung der TYP-Datei angelegt, enthält die REG-Datei nur den Verweis auf die TYP-Datei. In diesem Fall kann die Datei z.B. 'CS.prj' heißen und ist es unbedingt erforderlich, dass die Eintragungen 'ProductID' und 'FamilyID' mit dem zugehörigen Kartensatz übereinstimmen. Im Fehlerfall gibt MapSource kryptische Fehlermeldungen aus oder verabschiedet sich ohne Kommentar.

Die PRJ-Datei kann mit einem Texteditor bearbeitet werden Es ist aber unbedingt empfehlenswert den grafischen Editor von MapTk zu verwenden. Das erspart das Lernen der Syntax und Tippfehler.

Besonderheiten:

- Das Programm unterstützt Farben die mit einem 24-Bit-RGB-Wert dargestellt werden können. Entsprechend der Spezifikation einiger Geräte ist die Darstellung aber auf eine Farbpalette von max. 256 Farben (siehe Seite 75) beschränkt. Nicht in der Palette vorkommende Farben werden gerundet. Farbzeichen in Mustern, die nicht in der Liste der Farben 'Color=...' vorhanden sind machen das entsprechende Pixel transparent.
- Punkte (POI) dürfen eine beliebige, rechteckige Kombination aus 1 * 1 bis 32 * 32 Pixel-Spalten und -Zeilen haben. Größere Bilder werden beschnitten. Ein gutes Maß ist z.B. 10 * 10 bis 12 * 12. Aber auch kleiner, z.B. 4 Zeilen mit 7 Pixeln für ein kleines Dreieck (Berg). Die Farbe 'transparent' ist erlaubt. Eine Warnung wird angezeigt für POIs ohne Bild.
- Linien haben in der TYP-Datei immer 32 Pixel. In der Definition in der PRJ-Datei kann die Länge kleiner oder größer sein. Alles über 32 wird abgeschnitten. Sind unter 32 Pixel definiert werden sie solange wiederholt bis 32 erreicht sind. Um saubere Muster zu bekommen sollte deshalb die Zahl der Pixel 4, 8, 16 oder 32 betragen. Die Zahl der Zeilen sollte nicht zu groß sein. Mehr als 5 Zeilen gibt Balken, die viel abdecken. Eine obere Grenze wurde nicht getestet.
- Linien dürfen mit maximal 2 Farben definiert werden. Diese Linien sind dann zweifarbig. Bei einer Farbe wird die Linie an Pixeln der fehlenden Farbe transparent, wie beim Symbol für Punkte. Durch 'LineWidth' und 'BorderWidth' definierte Linien können die Farbe 'transparent' nicht verwenden.
- Wird 'LineWidth' und 'BorderWidth' auf 0 gesetzt entsteht eine unsichtbare Linie.
- Muster in Linien werden immer in die Richtung der Linie gedreht (Orientations-Bit ist gesetzt).

- Flächen mit Muster haben maximal 2 Farben. Bei nur einer Farbe wird die Fläche an den Stellen der 2. Farbe transparent. Flächen ohne Muster können nicht transparent sein.
- Die Textgröße kann verändert werden 'TextSize=...'.
- Die Farbe der Beschriftung kann mit 'TextColor=...' bestimmt werden.

3.5 Übersichtskarte

Übersichtskarten werden nur in MapSource verwendet um Detailkarten zu selektieren die an das GPS-Gerät übertragen werden sollen. Diese Übersichtskarte ist für MapSource / Basecamp erforderlich ! Eine Übersichtskarte kann aus einer oder mehreren Detailkarten erzeugt werden. Sie wird aus den in der PRJ-Datei gelisteten MP-Dateien im Arbeitsverzeichnis gebildet. Das Aussehen der Karte ist unkritisch, Sie wird nicht an das GPS-Gerät übertragen sondern dient nur der Übersicht in MapSource

- Das Programm setzt voraus, dass es fehlerfreie Kacheln sind die im höchsten Level nur den Hintergrund Type 0x4b enthalten. Das ist Level 4 wenn 'Levels=5' im Kopf angegeben ist. Alle Detailkarten müssen die gleiche Level-Definition haben.
- Alle Objekte aus dem höchsten aktiven Level der Detailkarten werden in die Übersichtskarte übertragen. Das ist der Level 3 wenn 'Levels=5' definiert wurde. Die Kopien gehen in der Übersichtskarte in den Level 0. So hat die Übersichtskarte nur 2 Level. Das ist ausreichend für MapSource. Das Ergebnis kann mittels Script verbessert werden.
- Zusätzlich werden alle Orte mit Typen < 0x0b in die Übersichtskarte übernommen.
- Aus der ID, dem Namen und der Abmessung der Detailkarten wird das Definitions-Polygon 0x4a in der Übersicht erzeugt. Die Abmessungen des Definitionspolygons werden aus der maximalen Ausdehnung aller Elemente der Detailkarte abgeleitet, mit Ausnahme des Hintergrundes 0x4b.
- 'Preview=Y' wird in den Kopf geschrieben.
- Die erste Detailkarte in der Liste bestimmt die Parameter für 'Codepage' und 'Elevation'. Die Übersichtskarte ist nicht transparent.
- ID und Name der Übersicht werden nach der folgenden Priorität bestimmt:
 1. aus der PRJ-Datei von der Definition der Übersichtskarte
 2. aus der Übersichtskarte in der Liste der MP-Dateien oder
 3. wird auf 'basemap' gesetzt.
 Der Name wird ebenfalls der Name der Datei. Der Name der Übersichtskarte sollte keine Leerzeichen enthalten
- Eine bestehende Übersichtskarte wird überschrieben.

3.6 Export in GPSMapEdit

In GPSMapEdit muss der Pfad für cgpsmapper vor dem ersten Export eingegeben werden. Gibt man stattdessen den Pfad zu MapTK.exe ein, kann aus GPSMapEdit direkt in eine IMG-Datei exportiert werden.

- Der vollständige Name der IMG-Datei ist anzugeben.
- Export reiner POI-Dateien wird nicht unterstützt (wird ignoriert). IMG-Dateien, nur aus POIs bestehend, sind möglich.

4 Automatische Routen-Berechnung

Die die Funktionen für automatisches Routing sind entworfen für MP-Dateien die mit GPSMapEdit bearbeitet und überprüft wurden. Die Verwendung von GPSMapEdit erfordert keine Kenntnisse von Syntax und Details einer MP-Datei. Für automatisches Routing verantwortliche Teile der MP-Datei sollten niemals mit einem Texteditor verändert werden. Die Script-Funktion von MapTk kennt einige Variable um Parameter für Straßen und Wege zu setzen.

4.1 Straßen und Wege

Nur wenige Polyline-Typen sind für eine automatische Routen-Berechnung verwendbar, Die folgende Tabelle enthält eine Liste der dieser Linien, die Beschreibung von Standard-Linien und Vorschläge für den Rest.

Code	Level	Beschreibung	Speed	Class	Restriktion									
					0	1	2	3	4	5	6	7	8	
01	3	Autobahn	6	4	0	0	0	0	0	0	0	1	1	0
02	3	Bundesstraße	5	3	0	0	0	0	0	0	0	0	0	0
03	3	Landesstraße	4	3	0	0	0	0	0	0	0	0	0	0
04	3	Kreisstraße	4	2	0	0	0	0	0	0	0	0	0	0
05	3	Nebenstraße	4	2	0	0	0	0	0	0	0	0	0	0
06	2	Straße	3	1	0	0	0	0	0	0	0	0	0	0
07	2	Wirtschaftsweg	2	0	0	0	1	1	1	1	1	0	0	1
08	2	Auffahrt	3	1	0	0	0	0	0	0	0	0	0	0
09	1	Auffahrt	3	1	0	0	0	0	0	0	0	0	0	0
0A	1	Wirtschaftsweg, unbefestigt	2	0	0	0	1	1	1	1	1	0	0	1
0B	3	Autobahnzubringer	3	2	0	0	0	0	0	0	0	1	1	0
0C	2	Kreisverkehr	2	1	0	0	0	0	0	0	0	0	0	0
0D		Pfad / Weg ⁵	1	0	0	1	1	1	1	1	1	0	0	1
0E	1	Unsichtbar, z.B. unter Seilbahn	1	0	0	1	1	1	1	1	1	0	0	1
0F	1	markierter Weg	1	0	0	1	1	1	1	1	1	0	0	1
10	1	Radweg	1	0	0	1	1	1	1	1	1	1	0	1
11	1	Reitweg	1	0	0	1	1	1	1	1	1	0	0	1
12	1	Steig / schwieriger Weg	1	0	0	1	1	1	1	1	1	0	1	1
13	2	Fußgängerzone	1	0	0	1	1	1	1	1	1	0	1	1
16	1	Pfad / Weg	1	0	0	1	1	1	1	1	1	0	0	1
1A	2	Fähre	1	2	0	0	0	0	0	0	0	0	0	0
1B	2	Fähre	1	2	0	0	0	0	0	0	0	0	0	0

gelb: Polyline für automatisches Routing.

blau: Beispiel für die Verwendung von nicht standardisierten Linien.

Die Spalten 'Level', 'Restriktionen' und 'Class' sind Beispiele für eine topografische Karte.

⁵ 'Endlevel' wird von Geräten ignoriert

Speed class	ø km/h
0	5
1	20
2	40
3	60
4	80
5	90
6	110
7	

Index	Restriktion
0	Mautstraße
1	keine Ambulanz
2	kein Lieferverkehr
3	kein Auto / Motorrad
4	kein Bus
5	kein Taxi
6	kein Fußgänger
7	kein Fahrrad
8	kein LKW

4.2 Karte bearbeiten

Die Karte muss ein Hintergrundpolygon Type=0x4b haben !

Um beste Ergebnisse zu erhalten sollte in GPSMapEdit 'Snap to grid' und 'Stick to neighbors' im Menü 'Tools → Options → Edit' aktiviert sein.

Die Bearbeitung beginnt mit einer fertigen Karte die schon alle Straßen und Wege enthält. Um die Datei für die automatische Routen-Berechnung verwenden zu können, müssen einige Informationen hinzugefügt werden:

- 'Routing=Y' in im Kopf der Date
- 'RoadID=...' für jede Straße / Weg (→ GPSMapEdit)
- 'Nodx=...' für jede Verbindung von Straßen und Wegen, sowie an allen unverbundenen Enden (→ GPSMapEdit).

Die MP-Datei kann optional Informationen enthalten, die das Verhalten beim automatischen Routen bestimmen:

- 'RouteParam=...' steuert das Verhalten von Straßen / Wegen (siehe Tabelle oben, → GPSMapEdit)
- Der Block [Restrict] ... [End] bestimmt Abbiege-Restriktionen für Knotenpunkte (→ GPSMapEdit). Die hier definierten Einschränkungen gelten nicht für Fußgänger. Zeitabhängige Abbiege-Restriktionen werden nicht unterstützt.

GPSMapEdit schreibt diese Informationen als korrekte MP-Statements während des Bearbeitens der Karte. Das Manual und die Hilfe von GPSMapEdit helfen die Details herauszufinden.

Vorausgesetzt sei eine vollständige Karte die mit den folgenden Schritten in GPSMapEdit für automatische Routen-Berechnung vorbereitet wird:

1. Erzeuge einen Straßennetzwerk in GPSMapEdit ('Tools → Generate Routing Graph → Using Coinciding Points of Polylines'). Verbindungen werden nur an punkten mit den selben Koordinaten gesetzt.

2. Das Netzwerk muss überprüft werden: 'Tools → Verify Map ...' mit allen Optionen 'Routing Graph' aktiviert. Diese Funktion sollte nach allen Änderungen am Straßennetz aufgerufen werden.
3. Die Fehler müssen beseitigt werden bis 'No invalid objects found' berichtet wird. 'invalid objects' sind zum Beispiel Straßen, die sich selbst kreuzen oder doppelte Knotenpunkte. Ein Knotenpunkt ist an allen, nicht weiter verbundenen Enden zu setzen. Eine Straße kann nicht auf der Start-Koordinate enden. Fehlende Verbindungen werden nicht erkannt.
4. Alle am Rand der Kachel endenden Straßen / Wege werden als 'external' markiert. Das sind die Punkte an denen die Route auf einer benachbarten Kachel fortgesetzt werden kann.

Während des Erstellens oder Ergänzens einer Karte sollten die Straßen / Wege gleich mit 'Connect to Nearest Nodes' verbunden werden. Übereinander liegende Punkte von Straßen die Brücken oder Tunnel kreuzen müssen vermieden werden.

4.3 Routen-Probleme beheben

Angenommen alle Klassifikationen und Einschränkungen sind wie erwartet, bleiben 2 Gruppen von Problemen bestehen:

- Viele Straßen / Wege sind nicht verbunden. Die Koordinaten passen nicht genau zusammen.
- Externe Knoten fehlen oder passen nicht zueinander.

MapTk hilft mit 2 Funktionen die fehlenden Verbindungen zu finden.

Zuvor müssen die Kacheln eines Projektes compiliert werden. Normalerweise wird ohne Fehlermeldung compiliert, vorausgesetzt alle von GPSMapEdit angezeigten Probleme wurden zuvor behoben.

Die 2 Funktionen zum aufspüren fehlender Verbindungen in MapTk sind im Menü 'GPX/MP':

- 'Find not connected near roads': Alle Straßen / Wege die sich in innerhalb eines definierten Abstands von Knotenpunkten befinden werden gesucht und markiert. Normalerweise können Abstände bis 3 m ohne weiteres verbunden werden. Bei größerem Abstand sollte die Umgebung beachtet werden um nicht falsche Verbindungen herzustellen. Siehe auch Seite 14.
- "Find not matching external nodes": Diese Funktionen untersucht alle IMG-Dateien des Projektes auf fehlende Verbindungen der Kacheln untereinander. Das Projekt muss vorher natürlich compiliert worden sein ('Make'). Siehe Seite 15.

Das Ergebnis beider Funktionen ist eine Datei mit POIs vom 'Type=0x00', die die fehlenden Verbindungen markieren. Der Skin von GPSMapEdit sollte ein entsprechender POI hinzugefügt werden: Icon mit 32 * 32 Pixel, ein roter Rande von 2 Pixeln um ein transparentes Zentrum. Code auf Seite 73.

Diese Dateien können der Karte hinzugefügt werden um die Probleme manuell zu korrigieren (automatisch erscheint mir zu gefährlich). Wenn die Straße / Weg zuvor selektiert wurden kann der Knotenpunkt im transparenten Zentrum des Marker-POI verschoben werden. Vor dem Speichern der Datei können die Marker entfernt werden. Es ist kein Problem wenn die Marker in der Datei bleiben. Der Compiler ignoriert alle Objekte 'Type=0x00'.

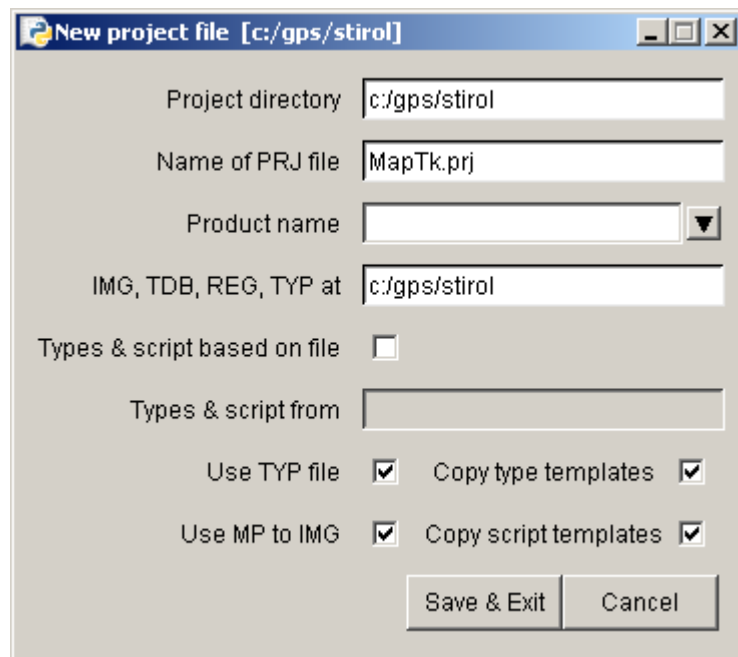
Änderungen an den Projektparametern oder der Windows-Registry sind nicht erforderlich. Wenn kein Problem übrig geblieben ist ('invalid objects', nicht verbundene Straßen / Weg oder externe Verbindungen) sollte die Routen-Berechnung problemlos funktionieren.

5 Editor

Der Editor besteht aus 6 spezialisierten Teilen. Jeder Teil wird durch einen Tab des obersten Editor-Fensters ausgewählt. 'Exit' schreibt die Änderungen in die PRJ-Datei und beendet den Editor. POIs, Linien oder Polygone werden nach Type sortiert in die Datei geschrieben.

5.1 Neue PRJ-Datei

Bei Aufruf des Editors bevor eine PRJ-Datei im Arbeitsverzeichnis angelegt wurde (oder Dateimenü 'New project file') fragt MapTk welche Art von Projekt angelegt werden soll.



The screenshot shows a Windows-style dialog box titled "New project file [c:/gps/stirol]". It contains the following fields and options:

- Project directory:** A text box containing "c:/gps/stirol".
- Name of PRJ file:** A text box containing "MapTk.prj".
- Product name:** A dropdown menu that is currently empty.
- IMG, TDB, REG, TYP at:** A text box containing "c:/gps/stirol".
- Types & script based on file:** An unchecked checkbox.
- Types & script from:** An empty text box.
- Use TYP file:** A checked checkbox.
- Copy type templates:** A checked checkbox.
- Use MP to IMG:** A checked checkbox.
- Copy script templates:** A checked checkbox.
- Buttons:** "Save & Exit" and "Cancel" at the bottom right.

Für ein ganz neues Projekt muss ein Name angegeben werden.

Die Combobox 'Product name' enthält alle für MapSource registrierten Dateien (schwarzes Dreieck rechts). In diesem Fall können die Daten für das Projekt aus der Registry entnommen werden.

Das Feld 'IMG, TDB, REG, TYP in' bestimmt den Ordner für diese Dateien. Wird der Ordner nicht verändert, kommen alle Dateien in einen Ordner, den Arbeitsordner.

Ist 'Types & script based on file' aktiviert können die Daten für TYP-Dateien und die Scripte aus einer bestehenden PRJ-Datei entnommen werden – anzugeben unter 'Types and scripts from'. In diesem Fall sind die folgenden Eingaben ausgeblendet. 'Use TYP file' bildet Pfad und Namen für die TYP-Datei. Um Layout-Beispiele zu kopieren kann 'Copy TYP templates' aktiviert werden. 'Use MP to IMG' bildet den Namen für die Übersichtskarte, den Ort der IMG-Dateien sowie die Maske für die IMG-Dateien aus den Inhalt des Ordners zu bilden. 'Use scripting templates' muss aktiviert werden um Script-Beispiele in die neue PRJ-Datei zu kopieren.

Bis auf den Namen der PRJ-Datei können alle Daten im Projekt-Editor geändert werden (nächstes Kapitel).

5.2 Header

Unter dem Header Tab werden die Projektparameter definiert. Die Daten werden aus der Windows Registry übernommen wenn 'Product name' aus der Liste übernommen wurde.

Header Maps POI Polyline Polygon Script

Name of PRJ file C:/GPS/Südtirol/MapTk.prj

Product name Südtirol ▼

Family ID 202

Binary files at d:/karten/Südtirol/

Overview map basemap.img

IMG files mask 002021*.img

Version 310

Copyright Mit Genehmigung von AUTONOME PROVINZ BOZEN-SÜDTIROL (Amt für...)

Compile (IMG, ...) ☒ Index file ☒

TYP file ☒ Garmin colors ☒ Binary files: IMG, MDX, TYP, REG

Save & Exit Cancel

Der Name des Produktes unter 'Product name' kann geändert oder alle Daten eines bestehenden Produktes übernommen werden (Pfeil-Knopf rechts).

Die 'Family ID' ist immer anzugeben (1 ... 65535) und darf mit keiner anderen ID in MapSource kollidieren.

'Binary files at' gibt den Speicher Ort der IMG-, Index-, MDX-, MDR-, TYP- und REG-Dateien an. Das Feld darf leer bleiben oder './' enthalten um den Arbeitsorder zu verwenden ('alles in einem Ordner'). Der Name der Index-, MDX-, MDR-, TYP- und REG-Dateien wird seit Version MapTk 2.7 aus der Family-ID gebildet. Das Beispiel FID = 202 ergibt folgende Dateien:

```
M00202_mdr.img  
M00202.TDB  
M00202.MDX  
M00202.TYP  
M00202.REG
```

'Overview map' bestimmt den Namen der Übersichtskarte. Der Speicherort entspricht den binären Dateien.

'Version' und 'Copyright' werden in die erzeugten Dateien übernommen.

'Compile (IMG, ...)' ist zu aktivieren wenn ein Kartensatz zu erzeugen ist.

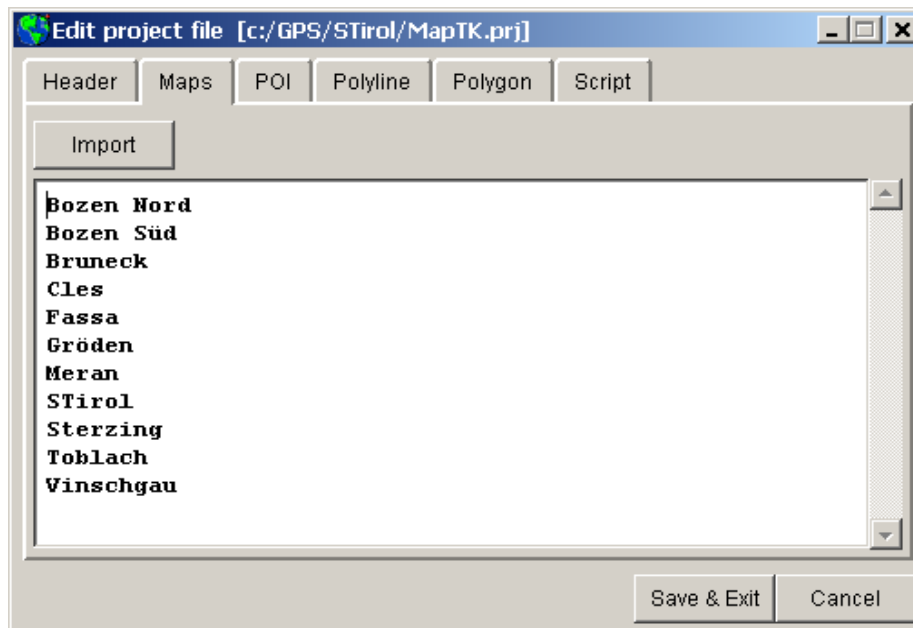
'Index file' wird zur Generierung eines Index (*_mdr.img) aktiviert. Ist diese Funktion nicht aktiviert aber eine Indexdatei schon vorhanden, so wird ein leerer Index erzeugt um die Registry nicht ändern zu müssen.

Passt der Index nicht zur TDB-Datei und den Detailkarten sind unvorhersehbare Probleme bei der Darstellung des Kartensatzes in MapSource oder dem GPS-Gerät zu erwarten.

Um eine TYP-Datei zu erhalten ist 'TYP file' zu aktivieren.

'Garmin colors' wählt eine von zwei 2 Paletten aus. Entweder die Palette bis Version 3.1.* (Checkbox aktiviert, Vorgabe nach Tabelle auf Seite 75) oder eine neue, hellere Palette. Siehe auch Seite 75.

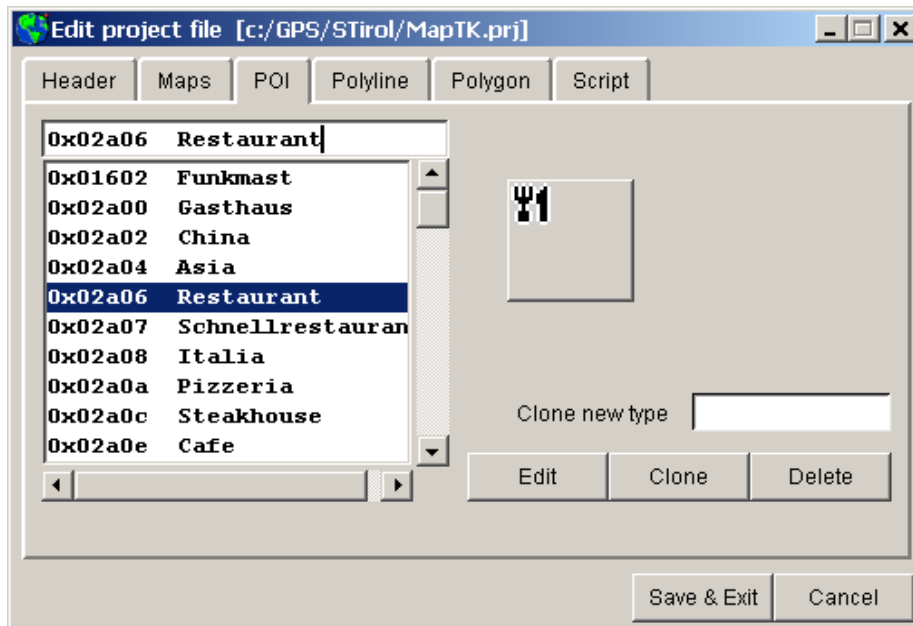
5.3 Maps



'Import' schreibt alle Namen von MP-Dateien im Arbeitsverzeichnis in eine sortierte Liste. Dateien, beginnend mit '#' werden nicht in die Liste übernommen. Import überschreibt eine vorhandene Liste. Die Liste wird in einem einfachen Texteditor dargestellt und kann bearbeitet werden.

5.4 Auswahlfenster

Für POIs, Polylines und Polygone wird ein gemeinsames Layout verwendet. Die Liste der Objekte wird aus Typ und dem ersten String gebildet. Das Symbol des ausgewählten Symbols wird angezeigt.



Ändern eines Objektes: Alle Objekte des betreffenden Types in der PRJ-Datei werden gelistet. Zur Bearbeitung wird eine Zeile markiert und 'Enter' gedrückt oder mit einem Doppelklick auf die Liste.

Neues Objekt erstellen: Der neue Typ wird in das Eingabefeld oberhalb der Liste eingegeben und die Return-Taste gedrückt oder Klick auf 'Edit'.

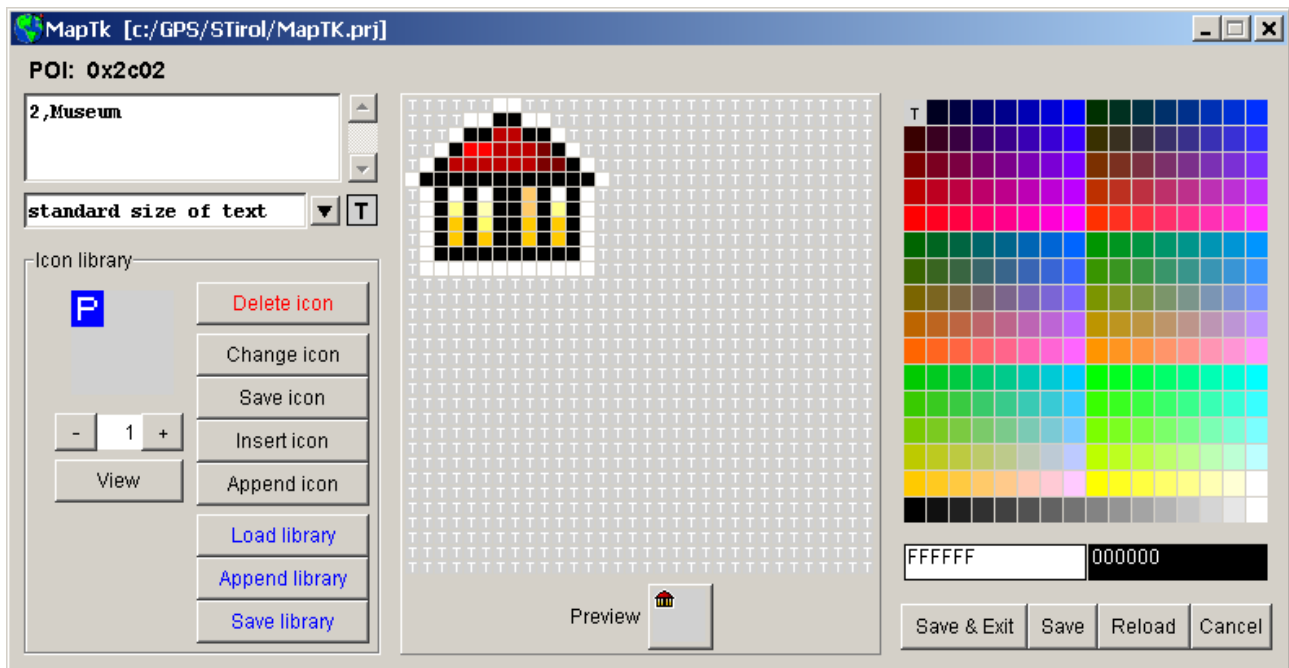
Kopie eines Objektes: Die Daten eines Objektes können kopiert werden (der Typ muss geändert werden) durch Eingabe des neuen Typs in das Feld 'Clone new type' und Klick auf 'Clone'.

Objekt löschen: Ein Objekt wird gelöscht durch Klick auf 'Delete'.

Starten des Editors öffnet ein Fenster mit 4 Bereichen. Generell ist die Funktion für POI, Polyline und Polygon ähnlich.

5.5 POI

Siehe auch Seite 15.



POI Editor

5.5.1 Links oben

Ein einfacher Texteditor für die Bearbeitung der Strings (Funktionen wie Ctrl-C, Ctrl-Z, ... werden unterstützt). Eine Zeile pro Sprache besteht aus dem Sprachcode (siehe Seite 76) und dem beschreibenden Text, getrennt durch ein Komma ','. Die gesamte Länge des Textes kann 125 Zeilen bei einer Sprache nicht überschreiten, 123 bei 2 Sprachen, 121 bei 3 Sprachen, ...

Die Größe des angezeigten Text kann in der Combobox darunter gewählt werden. 'standard size of text' ist die Vorgabe, die Größe wird dann von Garmin bestimmt.

Das Feld rechts bestimmt die Textfarben. Mit Klick rechts oder links wird die entsprechende Farbe übertragen. Die Standardfarben von Garmin werden benutzt wenn 'transparent' gewählt wird, das ist die Vorgabe.

5.5.2 Links unten

Zugang zum Inhalt Icon-Bibliothek:

- 'Change icon' kopiert das gewählte Icon in den Zeichenbereich. Das vorherige Bild wird überschrieben.
- 'Insert icon' fügt das Bild aus dem Zeichenbereich unter der angezeigten Nummer in die Icon-Bibliothek ein.
- 'Append icon' hängt das Bild des Zeichenbereichs an die Icon-Bibliothek an.

- 'Delete icon' löscht das angezeigte Icon aus der Bibliothek.
- 'View' erzeugt eine Seite mit allen Icons. Diese Seite kann als Postscript-Datei zum Drucken exportiert werden.
- '+' / '-' erhöht / verringert die Nummer des Icons (Shift: 10 mal). Das Icon kann durch Eingabe der Nummer direkt ausgewählt werden.

Eine Icon-Bibliothek wird zusammen mit dem Programm in den Speicher geladen und vor Ende des Programms in die Datei zurückgeschrieben. Die Bibliothek kann bearbeitet werden:

- 'Load library' lädt eine Bibliothek. Die aktuelle Bibliothek im Speicher wird überschrieben.
- 'Append library' hängt eine weitere Bibliothek oder eine Pixelgrafik (BMP, GIF, ICO, JPG, PNG, TIF) an die Icons im Speicher an. Bereits geladene Icons werden nicht angehängt.
- 'Save library' schreibt die Icons im Speicher in eine Datei.

Löschen oder Überschreiben von Icons muss bestätigt werden.

5.5.3 Mitte

Zeichenbereich für das Icon. Klick auf eines der 32 * 32 Pixel ändert die Farbe dieses Pixels. Alle 255 Farben können benutzt werden. Die Farbe wird aus der Palette entnommen. Mit 'T' markierte, graue Pixel sind transparent. Ein POI ist vollständig transparent wenn alle Pixel des Musters transparent sind.

- Linke Maustaste: Farbe des linken Farbfeldes (aktuelle Farbe).
- Rechte Maustaste: Farbe des rechten Farbfeldes (aktuelle Farbe).
- Maustaste halten und ziehen: Zeichnen einer Freihand-Linie.
- Control und eine Maustaste: Die Farbe aus dem Zeichenbereich wird die aktuelle Farbe.
- Shift und eine Maustaste: Die Farbe im Zeichenbereich wird durch die aktuelle Farbe ersetzt.
- Control + Cursor-Tasten schiebt das Bild Pixel für Pixel.

Das Feld 'Preview' zeigt das Icon in Originalgröße.

5.5.4 Rechts

Farbauswahl. Klick auf eine Farbe mit der linken oder rechten Maustaste ändert die aktuelle Farbe. Die zwei Felder zeigen die aktuellen Farben zusammen mit dem RGB-Wert (Hex) oder das Wort 'transparent'. Die transparente 'Farbe' (RGB = D0D0D0) befindet sich in der linken oberen Ecke.

5.5.5 4 Buttons

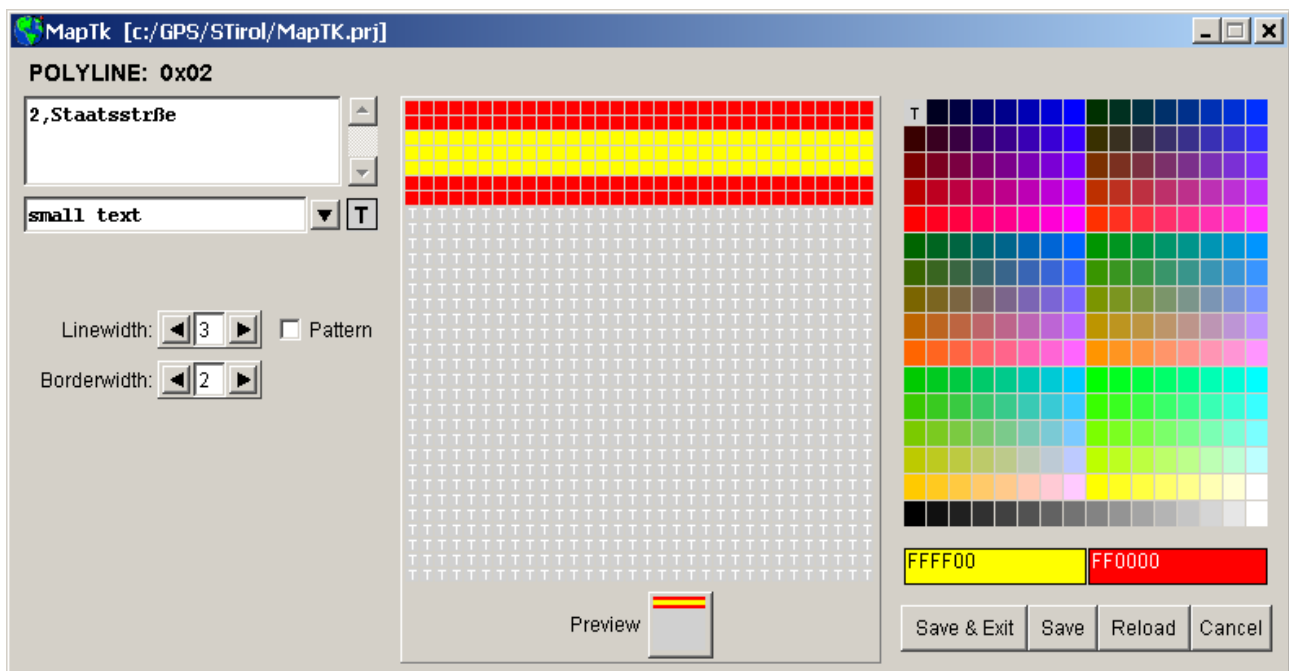
'Exit' sichert alle Änderungen und beendet den Editor.

'Save' sichert alle Änderungen ohne den Editor zu beenden.

'Reload' holt die ursprünglichen Daten zurück,

'Cancel' beendet den Editor, Änderungen werden verworfen.

5.6 Polyline



Polyline editor

5.6.1 Links oben

Ein einfacher Texteditor für die Bearbeitung der Strings (Funktionen wie Ctrl-C, Ctrl-Z, ... werden unterstützt). Eine Zeile pro Sprache besteht aus dem Sprachcode (siehe Seite 76) und dem beschreibenden Text, getrennt durch ein Komma ','. Die gesamte Länge des Textes kann 125 Zeilen bei einer Sprache nicht überschreiten, 123 bei 2 Sprachen, 121 bei 3 Sprachen, ...

Die Größe des angezeigten Text kann in der Combobox darunter gewählt werden. 'standard size of text' ist die Vorgabe, die Größe wird dann von Garmin bestimmt.

Das Feld rechts bestimmt die Textfarben. Mit Klick rechts oder links wird die entsprechende Farbe übertragen. Die Standardfarben von Garmin werden benutzt wenn 'transparent' gewählt wird, das ist die Vorgabe.

5.6.2 Links unten

'Pattern' nicht aktiviert: LineWidth (0 ... 12) und BorderWidth (0 ... 10) werden hier bestimmt. LineWidth und BorderWidth auf 0 gesetzt ergibt eine unsichtbare Linie. Das Aussehen der Polylinie ist im Zeichenbereich sichtbar.

'Pattern' ist aktiviert: Breite und Höhe des weiß abgegrenzten Zeichenfeldes werden hier bestimmt. Die Pixel im Zeichenfeld werden automatisch wiederholt bis eine Länge von 32 Pixeln erreicht ist. Es sind maximal 2 Farben (einschließlich transparent) möglich.

5.6.3 Mitte

'Pattern' nicht aktiviert: Darstellung der Linie. Klick mit einer Maustaste auf die Linie oder den Rand ändert die Farbe direkt.

- Linke oder rechte Maustaste: Farbe des linken/rechten Farbfeldes (aktuelle Farbe) wird auf die Linie oder den Rand übertragen. 'transparent' ist nicht möglich.
- Control und eine Maustaste: Die Farbe aus dem Zeichenbereich wird die aktuelle Farbe.

'Pattern' ist aktiviert: Breite und Höhe des weiß abgegrenzten Zeichenfeldes werden hier bestimmt. Sollten bereits 2 Farben verwendet werden, wird die Farbe des angeklickten Pixel für die gesamte Linie durch die aktuelle Farbe ersetzt. Ähnlich zum Bearbeiten von Polygonen.

5.6.4 Rechts

Farbauswahl. Klick auf eine Farbe mit der linken oder rechten Maustaste ändert die aktuelle Farbe. Die zwei Felder zeigen die aktuellen Farben zusammen mit dem RGB-Wert (Hex) oder das Wort 'transparent'. Die transparente 'Farbe' befindet sich in der linken oberen Ecke.

5.6.5 4 Buttons

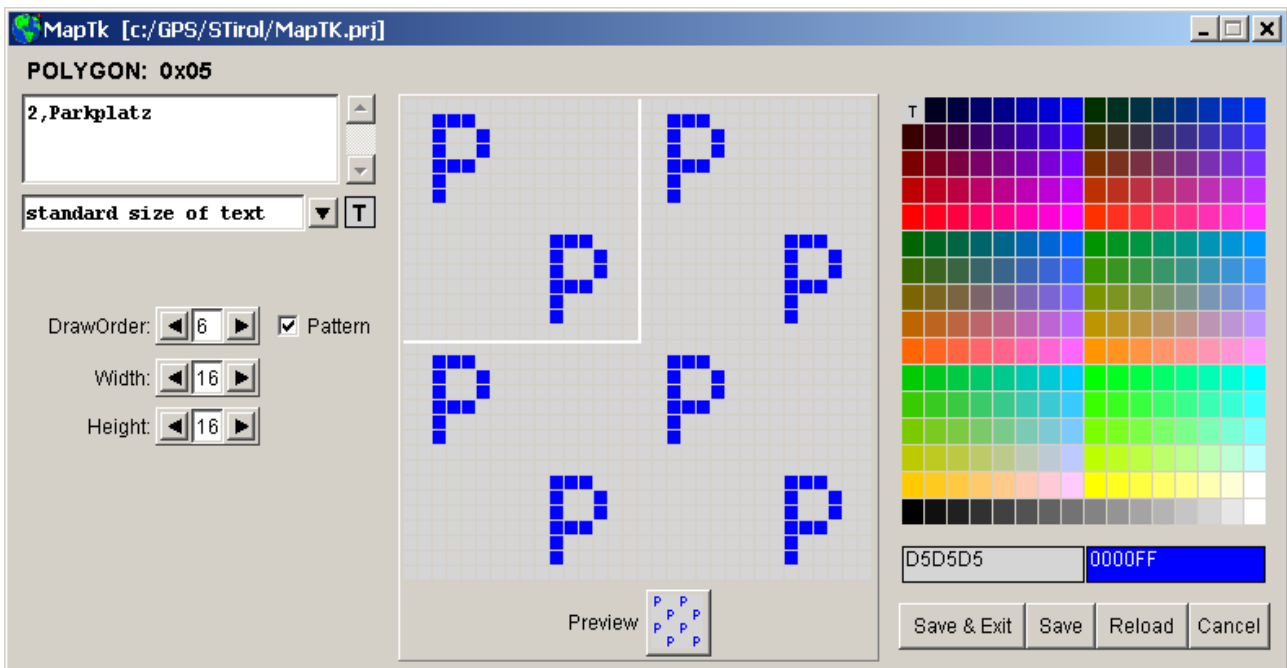
'Exit' sichert alle Änderungen und beendet den Editor.

'Save' sichert alle Änderungen ohne den Editor zu beenden.

'Reload' holt die ursprünglichen Daten zurück,

'Cancel' beendet den Editor, Änderungen werden verworfen.

5.7 Polygon



Polygon editor

5.7.1 Links oben

Ein einfacher Texteditor für die Bearbeitung der Strings (Funktionen wie Ctrl-C, Ctrl-Z, ... werden unterstützt). Eine Zeile pro Sprache besteht aus dem Sprachcode (siehe Seite 76) und dem beschreibenden Text, getrennt durch ein Komma ','. Die gesamte Länge des Textes kann 125 Zeilen bei einer Sprache nicht überschreiten, 123 bei 2 Sprachen, 121 bei 3 Sprachen, ...

Die Größe des angezeigten Text kann in der Combobox darunter gewählt werden. 'standard size of text' ist die Vorgabe, die Größe wird dann von Garmin bestimmt.

Das Feld rechts bestimmt die Textfarben. Mit Klick rechts oder links wird die entsprechende Farbe übertragen. Die Standardfarben von Garmin werden benutzt wenn 'transparent' gewählt wird, das ist die Vorgabe.

5.7.2 Links unten

Die Reihenfolge in der Darstellung des Polygons wird hier im Bereich 0 ... 7 bestimmt.

'Pattern' ist nicht aktiviert: keine weiteren Daten einzugeben.

'Pattern' ist aktiviert: Breite und Höhe des weiß abgegrenzten Zeichenfeldes wird hier bestimmt. Die Pixel im Zeichenfeld werden automatisch wiederholt bis eine Fläche von 32 * 32 Pixel gefüllt ist. Es sind maximal 2 Farben (einschließlich transparent) möglich.

Ein Polygon ist vollständig transparent wenn alle Pixel des Musters transparent sind.

5.7.3 Mitte

'Pattern' ist nicht aktiviert: Anzeige nur einer Farbe. Keine Bearbeitung. Klick mit einer Maustaste auf das Feld ändert die Farbe der Fläche direkt.

'Pattern' ist aktiviert: Zeichnen des Musters im weiß abgegrenzten Zeichenfeld mit der Maus wie bei POIs. Wenn das Zeichenfeld bereits 2 Farben enthält wird die Farbe des angeklickten Pixel für das gesamte Polygon durch die aktuelle Farbe ersetzt.

Ein Polygon ist vollständig transparent wenn alle Pixel des Musters transparent sind.

5.7.4 Rechts

Farbauswahl. Klick auf eine Farbe mit der linken oder rechten Maustaste ändert die aktuelle Farbe. Die zwei Felder zeigen die aktuellen Farben zusammen mit dem RGB-Wert (Hex) oder das Wort 'transparent'. Die transparente 'Farbe' befindet sich in der linken oberen Ecke.

5.7.5 4 buttons

'Exit' sichert alle Änderungen und beendet den Editor.

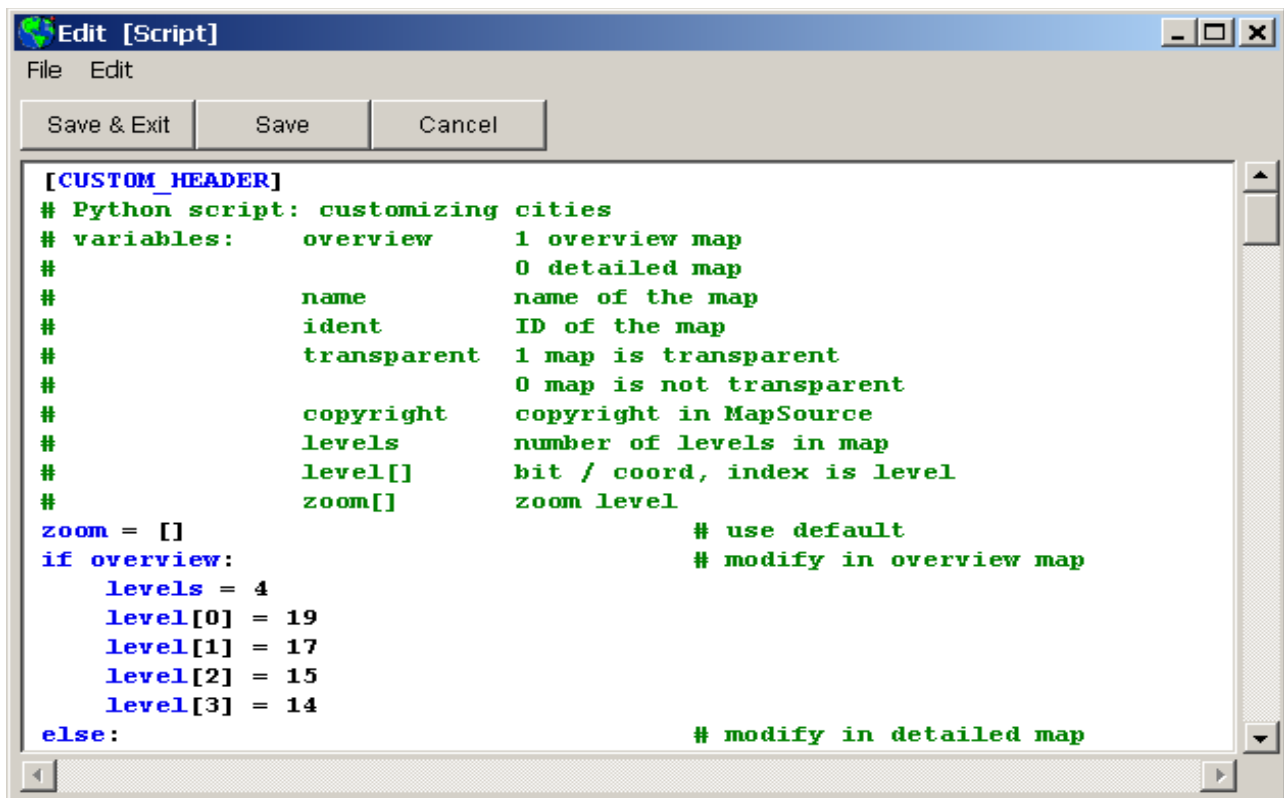
'Save' sichert alle Änderungen ohne den Editor zu beenden.

'Reload' holt die ursprünglichen Daten zurück,

'Cancel' beendet den Editor, Änderungen werden verworfen.

5.8 Script

'Exit' sichert das Script im Speicher. Die PRJ-Datei wird aktualisiert wenn die Editor-Funktion verlassen wird. 'Save' sichert die Änderungen im Speicher ohne den Script-Editor zu verlassen. 'Cancel' beendet den Script-Editor, Änderungen werden verworfen.



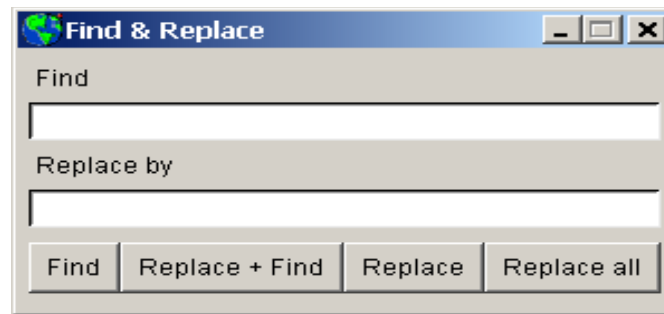
```
[CUSTOM_HEADER]
# Python script: customizing cities
# variables:      overview      1 overview map
#                  0 detailed map
#
#                  name         name of the map
#                  ident        ID of the map
#                  transparent   1 map is transparent
#                               0 map is not transparent
#                  copyright    copyright in MapSource
#                  levels       number of levels in map
#                  level[]      bit / coord, index is level
#                  zoom[]       zoom level
zoom = []
# use default
if overview:
# modify in overview map
    levels = 4
    level[0] = 19
    level[1] = 17
    level[2] = 15
    level[3] = 14
else:
# modify in detailed map
```

5.8.1 Funktionen

- Editieren von ASCII-Dateien
- Beim Bearbeiten werden Schlüsselworte farbig dargestellt.
- Find- und Replace-Funktion
- siehe 'Menü' und 'Tasten'

5.8.2 Menü

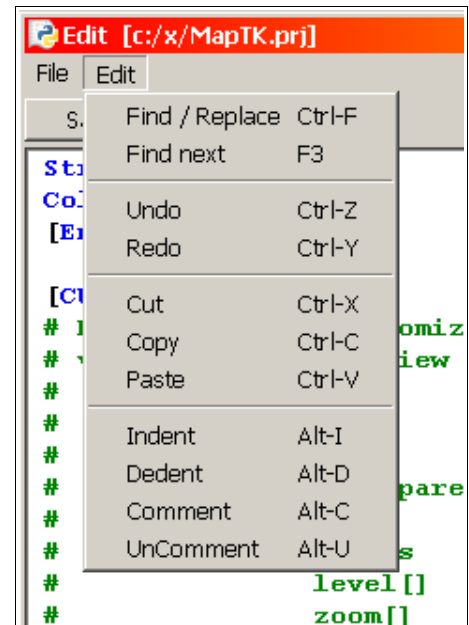
- 'File'
 - 'Save' speichert die Datei. Die vorherige Version wird überschrieben. Der Editor bleibt geöffnet. Der Button 'Save' hat die selbe Funktion.
 - 'Exit' beendet den Editor. Sollte der geladene Text verändert worden sein, erfolgt eine Rückfrage: Speichern ja/nein.
- 'Edit'
 - 'Find / Replace'



- 'Find': Es wird nach einem Text ab Cursor-Position gesucht.
 - 'Replace + Find': Text ersetzen und zur nächsten Position gehen.
 - 'Replace': Text ersetzen.
 - 'Replace all': Im gesamten Text ersetzen. Cursor wird nicht bewegt.
- 'Find next' wiederholt die Suche mit dem selben Text.
- 'Undo' macht die letzte Änderung am Text rückgängig.
- 'Redo' stellt rückgängig gemachte Änderung wieder her.
- 'Cut' schneidet markierten Text in die Zwischenablage aus.
- 'Copy' kopiert markierten Text in die Zwischenablage.
- 'Paste' fügt Text aus der Zwischenablage ein.
- 'Indent' rückt den Text ein.
- 'Dedent' rückt den Text aus.
- 'Comment' kommentiert eine Zeile aus.
- 'Uncomment' aktiviert eine auskommentierte Zeile.

5.8.3 Tasten

- DerCursor-Block hat die übliche Funktion.
- Markieren mit der Maus oder 'Shift-Cursor'.
- 'Control-C' wie Menü 'Copy'
- 'Control-F' wie Menü 'Find / Replace'
- 'F3' wie Menü 'Find next'
- 'Alt-I' wie Menü 'Indent'
- 'Alt-D' wie Menü 'Dedent'
- 'Control-V' wie Menü 'Paste'
- 'Control-X' wie Menü 'Cut'
- 'Control-Y' wie Menü 'Redo'
- 'Control-Z' wie Menü 'Undo'
- 'Control-Ende' zum Ende der Datei
- 'Control-Pos1' zum Anfang der Datei
- 'Alt-C' wie Menü 'Comment'
- 'Alt-U' wie Menü 'Uncomment'
- 'Maus-rechts' das 'Edit'-Menü als PopUp.



6 Dateien

Das Programm benötigt, bzw. erzeugt verschiedene Dateien.

6.1 MapTk.dat

Diese Datei ist neu in Version 2. 'MapTk.dat' teilt das Programmverzeichnis mit 'Maptk.exe' und 'MapTk_*.pdf'. Die Datei enthält die Icon-Bibliothek und einige Konfigurationsdaten. Die Datei wird nach dem Start von MapTk gelesen und vor dem Beenden auf die Platte zurückgeschrieben. Es ist eine komprimierte Binärdatei die nicht vom Benutzer bearbeitet werden kann. MapTk zeigt eine Warnung wenn die Datei nicht existiert und erzeugt eine leere Icon-Bibliothek und vorgegebene Konfigurationsdaten. Nur das Archiv für die Version 2.0 enthält diese Datei. Für spätere Versionen kann die Bibliothek zusätzlich heruntergeladen werden und in die bestehende Bibliothek importiert werden. Die Datei sollte in einem Backup-Konzept enthalten sein.

6.2 PRJ-Datei

Mit der Version 2 wurde das Dateiformat verändert. Ab Version 2.7 ist der Kopf der Datei anders gestaltet. Mit Aufruf des Editors erfolgt automatisch eine Konversion in das neue Format. Alte Versionen werden akzeptiert aber nicht geschrieben.

Obwohl die Datei mit einem ASCII-Editor bearbeitet werden kann, wird empfohlen die Datei nur mit den eingebauten Editoren zu bearbeiten. Eine neue PRJ-Datei wird mit dem eingebauten Editor erstellt. Das erspart das Lernen der Syntax und Tippfehler.

In der Projektdatei *.PRJ sind alle zur Erstellung eines Kartensatz erforderlichen Informationen abgelegt (Beispiel zum Download auf [MapTk](#)). Die Struktur der PRJ-Dateien ist an das MP-Format angelehnt. Die Datei ist in Blöcke unterteilt [...]. Der Name eines Blockes kann in Klein- oder Großbuchstaben geschrieben sein. Ein Block wird mit [End] abgeschlossen.

Die Informationen sind mit Schlüsselworten gespeichert. Ein Schlüssel kann in Klein- oder Großbuchstaben geschrieben sein. Nicht unterstützte Schlüssel werden als Warnung angezeigt und ignoriert.

Bei Texten dürfen die Begrenzungszeichen (' ") weggelassen werden. In diesem Fall kann das Leerzeichen () nicht verwendet werden. Die Begrenzungszeichen, Komma (,) und das Kommentarzeichen (#) sind innerhalb von Texten nicht erlaubt, alle Zeichen hinter # werden ignoriert. Die eingebauten schreiben – mit Ausnahme der Scripte – keine Kommentare in die Datei.

Sollten Fehler im Script auftreten werden die Zeilennummern im Block gezählt.

In Datei- und Pfadangaben kann anstelle '\' auch '/' geschrieben werden. Die REG-Datei wird später vorzugsweise im Verzeichnis der TDB-Datei oder im Verzeichnis der TYP-Datei als '<Product name>.reg' gespeichert.

[Project] Definition des Projektes, Die Daten werden schon im Dialogfenster eingegeben.
In Klammern { ... } der Name des Wertes

Name Name der PRJ-Datei, die im Arbeitsverzeichnis angelegt wird.

Product name { Product } erlaubt es ein neues Projekt anzulegen oder aus den bereits in die Registry eingetragenen Kartensätzen auszuwählen (die Daten der Registry⁶ werden in die Eingabefelder übernommen). Im Falle eines neuen Kartensatzes sind die freigegebenen Felder auszufüllen. Mit der später erzeugten REG-Datei kann der neue Kartensatz bei MapSource mit diesen Daten angemeldet werden.

Family ID	{ FamilyID } Wird in die TDB-, TYP-Datei und in die Windows Registry unter ID eingetragen. Beispiel: 200. Auf Konflikt mit anderen Kartensätzen achten ! Die Product ID ist fest auf 1 gesetzt.
Compile	= 1 bestimmt, dass ein Kartensatz erstellt wird.
Index	= 1 bestimmt, dass eine Indexdatei erzeugt werden soll.
Style	= 1 bestimmt, dass eine TYP-Datei erzeugt werden soll.
Overview map	{ OverviewMap } Name der Übersichtskarte ohne Verzeichnisangabe.
IMG files at	{ IMGPath } Verzeichnis in dem die kompilierten Karten und andere für MapSource benötigte Dateien gespeichert sind. Das Feld kann leer sein oder ':' enthalten um das Arbeitsverzeichnis zu benutzen.
IMG files mask	{ IMGfiles } Name der IMG-Dateien in 'IMGPath' (Beispiel: 0*.img für alle IMG-Dateien, die mit '0' beginnen, Die Übersichtskarte darf nicht auf dieses Suchmuster passen)
Version	{ Version } Ohne wichtige Funktion, kennzeichnet den Änderungsstand des Kartensatzes. Wird bei MapSource unter 'Kartenproduktinformationen' angezeigt.
Copyright	{ Copyright } Text wird optional in die TDB-Datei eingetragen und in MapSource in 'Help → Produktinformationen' angezeigt.

[IMG] Liste der MP-Dateien, die in IMG-Dateien konvertiert werden sollen. Die Namen sind pro Zeile ohne Schlüssel, '=' und Dateierweiterung aufzulisten (Beispiel: Lüneburg). Auch die Übersichtskarte wird hier eingetragen. Zeilen mit Semikolon (';') am Anfang sind Kommentar.

6 \\Registry\HKEY_LOCAL_MACHINE\SOFTWARE\Garmin\MapSource\Families\

[POI]

Definition von Icons zu einem Punkt in auf der Karte. Der Block wird für alle neu zu gestaltenden POIs wiederholt. POIs vom Typ > 0x10000 **müssen** in der PRJ-Datei deklariert werden um überhaupt dargestellt werden zu können ! Liste dieser POIs ab Seite 70.

- Type= Typ und Subtyp als 16-Bit-Wert des Punktes (Beispiel: *Type=0x2f0b* für Parkplatz). Der Wert darf dezimal oder hexadezimal angegeben werden. Es sind 32 Subtypen von 0x00 bis 0x1f möglich (für 'Services' also 0x2f00 bis 0x2f1f). Wenn Type < 0x100 wird das als Type mit Subtyp=0x00 behandelt. Zusätzlich führendes Byte 0x01 kennzeichnet vom Benutzer definieren Typ.
- String= Optionale Beschreibung des POI wenn keine explizite Beschreibung angegeben wurde (Beispiel: *String=2,Parken*. '2' ist der Code für die Sprache). Die Beschreibung wird dann anstelle der Vorgaben des Gerätes / MapSource angezeigt.
String=2," " oder
String=2,' '
ist möglich um vorgegebene Label zu überschreiben, nicht aber die in der IMG-Datei explizit vergebene Label.
- TextSize= 0: Standard (durch Garmin definiert, Vorgabe), 1: kein Text, 2: klein, 3: normal, 4: groß.
- TextColor= Farbe für die Darstellung der Beschriftung (z.B. *TextColor=0x0000ff* für blau). Wenn diese Zeile fehlt wird die Vorgabe benutzt.
- Color= Einer Farbe wird ein Zeichen für ein Pixel des Bildes zugewiesen (Beispiel: *Color=+,r0xff0000* für rot).
- Line= Definition einer Zeile des Bildes (Beispiel: *Line=---+--*). Dem Zeichen wird die Farbe aus der Farbtabelle [Colors] zugewiesen. Ist für ein Zeichen keine Farbe definiert, wird das Pixel transparent dargestellt. Die Größe eines Symbols ist 1 * 1 bis 32 * 32 Pixel.

[Polyline]

Definitionen für Linien. Der Block wird für alle neu zu gestaltenden Linien wiederholt. Linien vom mit Typ >= 0x10000 **müssen** in der PRJ-Datei deklariert werden um überhaupt dargestellt werden zu können ! Liste dieser Linien ab Seite 70.

- Type= Typ als 8-Bit-Wert der Linie (Beispiel: *Type=0x01* für Autobahn). Der Wert darf dezimal oder hexadezimal angegeben werden. Zusätzlich führendes Byte 0x01 kennzeichnet vom Benutzer definieren Typ.

String=	Optionale Beschreibung des POI wenn keine explizite Beschreibung angegeben wurde (Beispiel: <i>String=2,Autobahn</i>). Die Beschreibung wird dann anstelle der Vorgaben des Gerätes / MapSource angezeigt. Die Zahl ist der Code für eine Sprache. <i>String=2," "</i> oder <i>String=2,' '</i> ist möglich um vorgegebene Label zu überschreiben, nicht aber die in der IMG-Datei explizit vergebene Label.
TextSize=	0: Standard (durch Garmin definiert, Vorgabe), 1: kein Text, 2: klein, 3: normal, 4: groß.
TextColor=	Farbe für die Darstellung der Beschriftung (z.B. <i>TextColor=0x0000ff</i> für blau). Wenn diese Zeile fehlt wird die Vorgabe benutzt.
LineWidth=	Breite der Linie in Pixeln (Beispiel: <i>LineWidth=2</i>)
BorderWidth=	Breite des Randes in Pixeln (Beispiel: <i>BorderWidth=1</i>). Keine Angabe bedeutet kein Rand. Wird ein Rand definiert aber keine Farbe dazu, ist der Rand schwarz.
Color=	Es dürfen bis zu 2 Farben definiert werden (Beispiel: <i>Color=1,0x0000ff</i>). Ist für die Linie kein Muster angegeben, ist die erste Farbe die Linie selbst, die 2. Farbe der Rand. Ist für die Linie ein Muster angegeben werden die Farben entsprechen dem Zeichen gesetzt.
Line=	Definition einer Zeile jeder Linie (<i>Beispiel: Line=++++----</i>). Dem Zeichen wird die Farbe aus der Farbtabelle [Colors] zugewiesen. Ist für ein Zeichen keine Farbe definiert, wird das Pixel transparent dargestellt. Sind weniger als 32 Pixel definiert wird das Muster solange wiederholt bis 32 Pixel vorhanden sind. Es dürfen mehrere parallele Zeilen für eine Linie definiert werden.
[Polygon]	Definitionen für Flächen. Der Block wird für alle neu zu gestaltenden Flächen wiederholt. Polygone vom Typ $\geq 0x10000$ müssen in der PRJ-Datei deklariert werden um überhaupt dargestellt werden zu können ! Liste dieser Polygone ab Seite 70.
Type=	Typ als 8-Bit-Wert der Fläche (Beispiel: <i>Type=0x0c</i>). Der Wert darf dezimal oder hexadezimal angegeben werden. Zusätzlich führendes Byte 0x01 kennzeichnet vom Benutzer definierten Typ.
DrawOrder=	Der Wert (0 ... 7) gibt die Reihenfolge an in der die Flächen gezeichnet werden. Ein höherer Wert überschreibt darunter liegende Polygone mit niedrigerem Wert. Alle Polygone müssen mit dieser Zeile in die Darstellungsreihenfolge eingeordnet werden. Ausnahme: für die Definitionsfläche 0x4a wird DrawOrder ignoriert.

String=	<p>Optionale Beschreibung des POI wenn keine explizite Beschreibung angegeben wurde Beispiel: <i>String=2,Industrie</i>. '2' ist der Code für eine Sprache. Die Beschreibung wird dann anstelle der Vorgaben des Gerätes / MapSource angezeigt.</p> <p><i>String=2," "</i> oder <i>String=2,' '</i></p> <p>ist möglich um vorgegebene Label zu überschreiben, nicht aber die in der IMG-Datei explizit vergebene Label.</p>
TextSize=	0: Standard (durch Garmin definiert, Vorgabe), 1: kein Text, 2: klein, 3: normal, 4: groß.
TextColor=	Farbe für die Darstellung der Beschriftung (z.B. <i>TextColor=0x0000ff</i> für blau). Wenn diese Zeile fehlt wird die Vorgabe benutzt.
Color=	Es dürfen bis zu 2 Farben definiert werden (Beispiel: <i>Color=1,0x00ff00</i>). Ist für die Fläche kein Muster angegeben, wird die Fläche mit der ersten Farbe gefüllt, die zweite Farbe wird ignoriert. Ist ein Muster angegeben, werden die Farben entsprechend dem Muster eingesetzt.
Line=	Definition einer Zeile des Musters (Beispiel: <i>Line=++--</i>). Jedes Zeichen steht für die entsprechende Farbe. Das Muster wird solange wiederholt bis 32 Zeichen definiert sind. Die Zahl der Zeilen eines Muster beträgt immer 32. Die definierten Zeilen werden solange wiederholt bis 32 Zeilen definiert wurden. Dieses Muster sollte deshalb 2, 4, 8, 16 oder 32 Pixel / Zeile oder Zeilen haben.
[Custom_Header]	Python-Code zur Anpassung der Karteneigenschaften
[Custom_POI]	Python-Code zur Anpassung der Daten in RGN10.
[Custom_Polyline]	Python-Code zur Anpassung der Daten in RGN40.
[Custom_Polygon]	Python-Code zur Anpassung der Daten in RGN80.
[End]	steht als Abschluss eines Blockes.

6.3 PRJ-Datei in den Funktionen

Block / Zeile	Funktion			
	Script	IMG	Make	TYP
[Project]	nein	nein	ja	ja
Product=			ja	ja
FamilyID=			ja	ja
Overview=			ja	nein
IMGpath=			ja	nein
IMGfiles=			ja	nein
Compile			ja	nein

Block / Zeile	Funktion			
	Script	IMG	Make	TYP
Index			ja	nein
Style=			optional	ja
Version=			ja	nein
Copyright=			optional	nein
[IMG]	nein	nein	ja	nein
<Kartenname>			1 Zeile / Karte	1 Zeile / Karte
[POI]	nein	nein	optional	optional
Type=			ja	ja
String=			optional	optional
TextSize=			optional	optional
TextColor=			optional	optional
Color=			>= 1	>= 1
Line=			>= 1	>= 1
[Polyline]	nein	nein	optional	optional
Type=			ja	ja
String=			optional	optional
TextSize=			optional	optional
TextColor=			optional	optional
LineWidth=			optional	optional
BorderWidth=			optional	optional
Color=			1 oder 2	1 oder 2
Line=			optional	optional
[Polygon]	nein	nein	optional	optional
Type=			ja	ja
DrawOrder=			0 ...79	0 ... 7
String=			optional	optional
TextSize=			optional	optional
TextColor=			optional	optional
Color=			1 oder 2	1 oder 2
Line=			optional	optional
[Custom_Header]	optional	nein	nein	nein
<Phyton-code>	optional			
[Custom_POI]	optional	nein	nein	nein
<Phyton-code>	optional			
[Custom_Polyline]	optional	nein	nein	nein
<Phyton-code>	optional			
[Custom_Polygon]	optional	nein	nein	nein

Block / Zeile	Funktion			
	Script	IMG	Make	TYP
<Python-code>	optional			

ja	muss vorhanden sein
optional	kann verwendet werden
nein	wird ignoriert

6.4 MP-Datei

Die Quellfiles haben die Erweiterung *.MP. Sie werden vorzugsweise mit GPSMapEdit bearbeitet. Versionen ab 1.0.56 von GPSMapEdit können die 'user defined' 3-Byte-Typen bearbeiten. Nicht alle Blöcke und Schlüsselwörter, die GPSMapEdit generiert werden unterstützt. Nicht unterstützte Blöcke und Schlüssel werden ignoriert.

Die unterstützten Blöcke und Schlüssel sind (Übersicht der unterstützten Elemente):

[IMG ID]	Kopfinformation
ID=	Identifikation der Kachel, die als Dateiname für die IMG-Datei übernommen wird
Name=	Name der Kachel wie er in MapSource angezeigt wird
OverView=	Y: Dies ist eine Übersichtskarte.
Transparent=Y:	Die Kachel wird als transparent generiert (kein Hintergrund-Polygon und Transparent-Bit gesetzt) S: Die Kachel wird als semi-transparent generiert (kein Hintergrund-Polygon, Transparent-Bit nicht gesetzt) Transparente Karten können nicht routingfähig sein !
Drawpriority=	Reihenfolge der Darstellung bei mehreren Kartensätzen. Vorgabe: 24 (Topo-Karten)
Routing= Y:	MapTk fügt Routing-Informationen der IMG-Datei hinzu. N: Keine Routing-Informationen in der IMG-Datei, auch wenn ein Routing-Netzwerk generiert wurde.
Preview= Y:	bestimmt eine Übersichtskarte entgegen den Namenskonventionen. N: bestimmt eine Detailkarte entgegen den Namenskonventionen.
Copyright=	Der Text wird in die IMG-Datei übernommen. Anzeige in GPSMapEdit (optional)
CodePage=	Codepage für Label. Z.B. 'CodePage=1252' für Texte mit lateinischen Zeichen (wird in die IMG-Datei übernommen, optional)
Levels=	ist die maximale Anzahl der Zoom-Ebenen
Levelx=	Definition der Bit / Koordinate im Levelx. In Levelx = Levels-1 liegt nur der Hintergrund Type=0x4b.

Zoomx=	(optional)
[RESTRICT] ⁷	Abbiege-Restriktionen für Routing
Nod=	Knotenpunkt mit Abbiege-Restriktionen
TraffPoints=	Liste der beteiligten Knoten für automatisches Routen.
TraffRoads=	Liste der beteiligten Straßen für automatisches Routen.
RestrParam=	Liste der beteiligten Fahrzeuge
[POI]	'Points of Interest'
Type=	Typ des Punktes (16 Bit) und selbst definierte 3-Byte-Typen (Type > 0xffff) ⁸
Label=	Beschriftung des Punktes (optional)
EndLevel=	ist der höchste Level in dem der POI zu sehen ist
HouseNumber=	(darf alle druckbaren Zeichen enthalten)
StreetDesc=	(darf alle druckbaren Zeichen enthalten)
City=Y	Kennzeichen für Orte
CountryName=	Name des Landes. In GPSMapEdit kann dieses Format zum Schreiben gewählt werden: Tools → Options... → Load & Save → Settings for Polish Format.
RegionName=	Name der Region
CityName=	Name der Stadt
Zip=	Postleitzahl
Phone=	Telefonnummer (darf alle druckbaren Zeichen enthalten)
Data0=	Geografische Koordinate
[POLYLINE]	alle Linien und Straßen / Wege
Type=	Typ der Linie (1 Byte) und selbst definierte 3-Byte-Typen (Type > 0xffff)
Label=	Beschriftung der Linie (optional)
EndLevel=	ist der höchste Level in dem der POI zu sehen ist (oder Levels=)
RoadID=	einheitliche Nummerierung aller Straßen und Wege für automatische Routen-Berechnung (wird von GPSMapEdit bestimmt). Nicht mit einem Texteditor verändern !

⁷ Der Block wird vollständig von GPSMapEdit ausgefüllt. Nicht mit einem Texteditor bearbeiten !

⁸ 3-Byte-Typen werden nicht auf allen GPS-Empfängern angezeigt.

RouteParam=	Liste von 12 Werten, die das Verhalten der Straße beim automatischen Routen bestimmen. Wird in GPSMapEdit gesetzt oder in einem Script (Ausnahme: Einbahnstraße) mittels spezieller Variablen.
Data0=	Liste geografischer Koordinaten
Nod?=	Kennzeichnung aller Knotenpunkte für automatisches Routen (wird von GPSMapEdit bestimmt)
[POLYGON]	alle Flächen
Type=	Typ des Polygons (1 Byte) und selbst definierte 3-Byte-Typen (Type > 0xffff)
Label=	Beschriftung des Polygons (optional)
EndLevel=	ist der höchste Level in dem der POI zu sehen ist (oder Levels=). Der Hintergrund (Type=0x4b) wird vom Compiler automatisch auf höchstmöglichen Wert gesetzt.
Data0=	Liste geografischer Koordinaten

6.5 IMG-Datei

Die IMG-Datei wird aus der MP-Datei erzeugt und enthält eine Kachel eines Kartensatzes oder die Übersichtskarte. Ein Kartensatz kann beliebig viele Kacheln enthalten. Die Kacheln werden in der Übersichts-IMG-Datei und in der TDB-Datei referenziert. Welche IMG-Dateien aus einer MP-Datei zu erzeugen sind steht in der PRJ-Datei. Die Übersichtskarte muss alle Detailkarten in ihrem Umriss als Polygon vom Typ 0x4a enthalten. Der Name dieser Polygone ist der Name der Kachel mit dem Dateinamen der IMG-Datei (ohne '.img'), getrennt durch '~[0x1d]'. Beispiel: 'Bozen~[0x1d]00001006'. Die Begrenzung auf 256 Punkte von Linien und Polygonen gilt nicht für Übersichtskarten.

Eine topografische Karte enthält Höhenlinien. Die Linien haben die Typen 0x20, 0x21 und 0x22. Wenn die Linien in einer IMG-Datei gefunden werden und die TDB-Datei entsprechen vorbereitet ist kann das Profil von Routen in MapSource und BaseCamp angezeigt werden.

Detail-Karten enthalten keine Objekte 'Type=0x00' und Polygon 'Type=0x4a'.

Alle IMG-Dateien eines Kartensatzes sind in einem Verzeichnis zu speichern. Alle IMG-Dateien, die in diesem Verzeichnis stehen können in einen Kartensatz übernommen werden. Welche Dateien das sind ist in der PRJ-Datei unter 'IMGfiles=' als Maske angegeben. Beispiel einer Maske: '0*.img' für alle Dateien deren Name mit '0' beginnt. Die Übersichtskarte darf nicht zur Maske passen.

Die maximale Größe für topologische Karten liegt bei etwa 4 MByte (Standard-Typen), erzeugt aus einer etwa 20 MByte großen MP-Datei. Der Einsatz benutzerdefinierter Typen erlaubt wesentlich größere Dateien. Mit MapTk erzeugte Karten sind optional transparent. Die Kodierung von Namen und anderen Texten in der Karte ist bei MapTk immer 8 Bit. Das erlaubt z.B. Umlaute und 'ß' (Codepage 1252) oder kyrillische Schriftzeichen (Codepage 1251).

Ab Version 2.7 enthalten die IMG-Dateien zusätzliche Informationen für einen Index (MDR-Datei), ab Version 3.00 wird optional Information für automatische Berechnung von Routen hinzugefügt.

6.6 TYP-Datei

Die erforderlichen Angaben werden aus einer PRJ-Datei entnommen. TYP-Dateien bremsen den Bildaufbau in MapSource. Die Mindestanforderungen an die PRJ-Datei sind:

- Im Block [Project] müssen die Schlüssel 'ProductID', 'FamilyID' und 'Typ' vorhanden sein. Sind diese Schlüssel vorhanden wird zusätzlich eine REG-Datei erzeugt.
- Die Blöcke [POI], [Polyline] und [Polygon] werden zur Erstellung herangezogen. [DrawOrder] ist veraltet, wird aber ebenfalls akzeptiert.

Andere Blöcke werden ignoriert bei der Generierung von TYP-Dateien.

Ist im Block [Project] ein vollständiges Produkt definiert, wird eine vollständige REG-Datei erzeugt, mit der alle erforderlichen Produktparameter in die Registry unter 'HKEY_LOCAL_MACHINE\SOFTWARE\Garmin\MapSource\Families\' eingetragen werden können. Ein vollständiges Produkt ist definiert durch die Schlüssel 'FamilyID', 'IMGpath', 'TYP', 'TDB' und 'OverviewMap'.

Doppelklick auf die REG-Datei meldet den Kartensatz in der Windows-Registry an oder aktualisiert den Eintrag.

Die Namen für TYP-Dateien müssen für alle Kartensätze auf dem GPS-Gerät unterschiedlich sein !

6.7 TDB-Datei

Die TDB-Datei enthält wichtige Informationen über die einzelnen Kacheln. Unter anderen Daten ist die Größe der Subfiles enthalten. Das Kompilieren einer Kachel kann die Größe einer Kachel so verändern, dass MapSource damit Probleme bekommt. Es ist deshalb sinnvoll diese Datei - bei Änderung an Kacheln - neu zu erzeugen, zumal der Vorgang selbst bei einigen hundert Kacheln in einigen Sekunden abgeschlossen werden kann.

Durch die Maske in 'IMGfiles=' in der PRJ-Datei werden die Karten eines Satzes ausgewählt. Beispiel: 'IMGfiles=0*.img' wählt alle Dateien aus die mit '0' beginnen. Also '00000030.img' aber nicht 'topo.img' oder '12345678.img'.

Außerdem enthält die TDB-Datei die ProductID und FamilyID die möglicherweise in der PRJ-Datei verändert wurde. Stimmen ProductID und FamilyID in TDB- und TYP-Datei sowie der Registry nicht überein, verweigert MapSource den Dienst. Weicht die Größe einer Kachel von dem Eintrag in der TDB-Datei ab meldet MapSource beim Laden in das GPS-Gerät, dass ein kleiner Kartensatz nicht in einen reichlich großen Speicher passt !

Die TDB-Datei enthält zusätzliche Informationen für einen Index (MDR-Datei) und automatische Routen-Berechnung.

Wenn **alle** Kacheln Höhenlinien enthalten wird die TDB-Datei für die Anzeige von Profilen von Routen in MapSource und BaseCamp vorbereitet.

6.8 Index

Ab Version 2.7 kann innerhalb der Make-Funktion ein Suchindex für MapSource und GPS-Geräte erzeugt werden. Die Funktion ist nur über 'Make' erreichbar um die Konsistenz von Karten und Indexdatei möglichst sicherzustellen. Der Index benötigt spezielle Sektionen der IMG-Dateien. Dazu ist es erforderlich ältere IMG-Dateien neu zu compilieren. Eine Inkompatibilität wird angezeigt und die Indexerzeugung abgebrochen. Es ist unbedingt auf Konsistenz zwischen Kartensatz, der Registry und der Indexdatei zu achten !

In den Index werden Orte und POIs aufgenommen der Gruppen 0x0100 – 0x1100, 0x2800, 0x2a00 – 0x3000 und 0x6400 – 0x6600. 'City', 'Region' und 'Country' sollten angegeben werden um in MapSource die Suchen danach eingrenzen zu können.

In MapSource sucht über den ganzen Kartensatz und sortiert die Ausgabe alphabetisch. GPS-Geräte suchen nur im Umkreis (variiert je nach Karte und Gegend) und Sortiert nach Entfernung vom aktuellen Standort.

6.9 LOG-Datei

Die Log-Funktion muss in 'File → Preferences' aktiviert werden. MapTk schreibt Ergebnisse in die Testdatei 'MapTk.log' Projektverzeichnis. Diese Datei wird immer wieder neu angelegt. Unmittelbar hintereinander eingegebene Aufträge werden in einer LOG-Datei zusammengefasst. Sie enthält ausgewählte Zeilen aus dem Fortschrittfenster. Das sind die Zeilen, die beginnen mit 'Input:', 'Script:', 'Info:', 'New:', 'Update:', 'Warning:', 'Error:', 'Stopped:', und 'Make:'. Damit ist insbesondere nach 'Make' sehr großer Projekte eine schnell Übersicht über die durchgeführten Operationen gegeben. Die Meldungen wurden in Version 3.1 für diese Funktion modifiziert.

7 Python-Script

Vier Python-Skripte werden von der Funktion 'Script' verwendet um MP-Dateien zu manipulieren. Die Skripte sind in einer PRJ-Datei gespeichert. Um die Skripte verändern zu können, sollte man sich mit Python etwas vertraut machen (<http://www.python.org>).

7.1 Übersicht

Die Bearbeitung der MP-Datei erfolgt über Variable:

Variable	Header	POI	Polyline	Polygon
overview	r	r	r	r
type	-	rw	rw	rw
drawpriority	rw	-	-	-
transparent	rw	-	-	-
copyright	rw	-	-	-
levels	rw	-	-	-
codepage	rw	-	-	-
level	rw indexed	rw	rw	rw
zoom	rw indexed	-	-	-
label	-	rw	rw	rw
name	rw	-	-	-
ID	rw	-	-	-
domain	w indexed	r	-	-
index	-	r	-	-
routing	w	-	r	-
roadclass	-	-	rw	-
speed	-	-	rw	-
restrict	-	-	rw indexed	-
dirindicator	-	-	r	-
oneway	-	-	rw	-

r nur lesen
 rw lesen und schreiben
 - nicht verfügbar

Variable können für wiederholte Aufrufe gespeichert werden.

7.2 Daten des Headers

Der Block [CUSTOM_HEADER] wird einmal pro MP-Datei aufgerufen. Globale Variable sollten hier initialisiert werden. Ausgewertet und verändert werden können:

'transparent'	schaltet mit 'S' oder 'Y' die Karte transparent.
'copyright'	erlaubt die Angabe eines Textes
'codepage'	ist die Nummer der Codepage (z.B. lateinisch: 1252, kyrillisch: 1251).
'levels'	gibt die Anzahl der Levels für die Karte im Bereich 2 bis 8 an.
'level[]'	ist die Anzahl Bits / Koordinate. Index ist der Level 0 ... levels-1. Der Bereich ist 8 ... 24 und muss absteigend sein.
'zoom[]'	ist die Zoom-Stufe für einen Level. Index ist der Level 0 ... levels-1. Der Bereich ist 0 ... 8 und muss aufsteigend sein.
'name'	ist der Name der Karte. Darf leer bleiben.
'ID'	ist die Identifikation der Karte (8 numerische Zeichen bei Detailkarten, bis zu 8 Zeichen mit mit Buchstaben, beginnend bei Übersichtskarten).
'domain'	ist eine Liste von Polygon-Typen. Der Label solches Polygons wird in die Variable 'domain' geschrieben wenn die Position eines POI sich innerhalb des Polygons befindet. Das kann benutzt werden um z.B. die 'City=' für eine Gemeinde zu setzen.
'routing'	kann auf 'Y' oder 'N' gesetzt werden um die Roten-Berechnung für die Karte ein- / auszuschalten.

Für mit Python erfahrene Benutzer:

'ps'	ist ein Dictionary des zu bearbeitenden Objekt. Die Schlüssel stammen aus der MP-Datei. Die Schreibweise ist zu beachten.
'mp'	ist ein String-Array das in die MP-Datei ausgegeben wird.

'status.append(text)' gibt den Text-Parameter im Status-Fenster des Programms aus.

Variable, die nicht im Block vorkommen bleiben unverändert.

7.3 Daten der Objekte

Die Blöcke [CUSTOM_POI], [CUSTOM_POLYLINE] und [CUSTOM_POLYGON] werden für jedes, dem Block entsprechende Objekt einmal aufgerufen. Ausgewertet und verändert werden können:

'overview'	ist 'True' wenn eine Übersichtskarte bearbeitet wird.
'type'	ist der Code für das Objekt
'level',	bis zu dem ein Objekt in der Karte dargestellt wird.
'label'	ist die Beschriftung des Objektes.

Nur für POIs:

'index'	ist 'True' wenn der Typ des POI in den Index aufgenommen wird.
'domain'	ist der Label einer Fläche (die im Header definiert wurde und in der der POI liegt).

Für Karten mit Routen-Berechnung stehen für Straßen / Wege zusätzliche Variable zur Verfügung:

'speed'	entspricht dem 1. Parameter in 'Routeparam' der MP-Datei.
'roadclass'	entspricht dem 2. Parameter in 'Routeparam' der MP-Datei.
'restrict'	entspricht den Parametern 4 bis 12 in 'Routeparam' der MP-Datei.
'dirindicator'	zeigt nur die Pfeile in GPSMapEdit. 'dirindicator' wird vom Compiler nicht ausgewertet.
'oneway'	entspricht dem Parameter 3 in 'Routeparam' der MP-Datei. Ist 'True' bei Einbahnstraßen. Sollte auf den Wert von 'dirindicator' gesetzt werden ('oneway=dirindicator' im Script).

7.4 Etwas Python

Beispiel siehe Seite 62.

Im Prinzip sind alle Eigenschaften eines Objektes hier mit Python auszuwerten oder zu verändern, nicht nur die beschriebenen Variablen. Das erfordert aber Kenntnis der internen Strukturen. Objekte sind in einem Array von Dictionaries gespeichert. Das an das Script übergebene Dictionary eines Objekts ist `ps[]`. Das Dictionary kann mit

```
status.append(ps)
```

im Status-Fenster des Programms angezeigt werden. Die Schreibweise ist (Schlüssel : Wert, Schlüssel : Wert, ...)

7.4.1 Allgemein

Blockbildung erfolgt durch Einrückung mit z.B. 4 Leerzeichen.

Kommentare beginnen mit '#'. Der Rest der Zeile wird ignoriert.

'return' darf im Script nicht verwendet werden.

7.4.2 Variable

Variable müssen nicht deklariert werden. Globale Variable werden aber mit

```
global xyz                # globale Variable xyz
```

angelegt. Dieses Statement muss in jedem Block stehen in dem die globale Variable benutzt wird. Um Konflikte mit den Variablen des Programms zu vermeiden sollten selbst definierte Variable z.B. mit 'my_...' beginnen. Diese Konvention gilt auch für lokale Variable.

Der Index für indizierte Variable steht in Klammern (z.B. level[1]). Der Index für Dictionaries kann auch ein Text sein (z.B. ps['Label']).

7.4.3 Funktionen

Funktionen sind auf den aktuellen [CUSTOM_...]-Block beschränkt. Z.B.

```
def test(s, val):  
    s.append('Test: %s' % (val))
```

Aufruf dann mit

```
test(status, '...')        # gibt 3 Punkte aus
```

Formatierung von Strings ähnlich wie in C.

Die Standard-Funktionen upper() und lower() konvertieren keine String-Konstanten mit Umlauten. Die String-Konstante muß mit decode('cp1252') umgewandelt werden. Beispiel:

```
s = lower('ABCÄÖÜ'.decode('cp1252'))
```

Ergebnis: 'abcäöü'

7.4.4 'status.append()'

Meldungen können mit 'status.append('.....')' im Fenster des Programms angezeigt werden. Beispiel in [CUSTOM_HEADER]:

```
status.append('Copyright:' + copyright) # zeigt das Copyright an
```

'+' fasst Strings zusammen, hier den Text und eine String-Variable.

7.4.5 Bedingungen, Schleifen

Für bedingte Anweisung siehe Beispiele in den Muster-PRJ-Dateien. Vergleiche sind '==', '!=', '>', '>=', ... 'in' prüft ob ein Wert in einer Liste, einem Array oder Dictionary enthalten ist. Die folgende Anweisung gibt Typ und die Beschriftung aller Objekte aus für die ein Text definiert ist:

```
if 'Label' in ps:
    status.append('%2x: %s' % (type, label))
```

'elif' prüft alternative alternative Bedingungen, der 'else'-Zweig wird ausgeführt wenn keine der vorherigen Bedingungen wahr wurde.

Die folgende Schleife gibt in [CUSTOM_HEADER] die Bits / Koordinate aller Level aus:

```
for i in range(levels-1):
    status.append('Level%d=%d' % (i, level[i]))
```

Die folgende Schleife gibt alle Einträge eines Dictionaries mit allen Werten aus:

```
for s in ps:
    status.append(s + '=' + ps[s])
```

'continue' beginnt sofort mit dem nächsten Durchgang, 'break' beendet die Schleife vorzeitig.

8 Tipps & Tricks

8.1 Zusammenhang TYP und IMG

TYP-Dateien sind

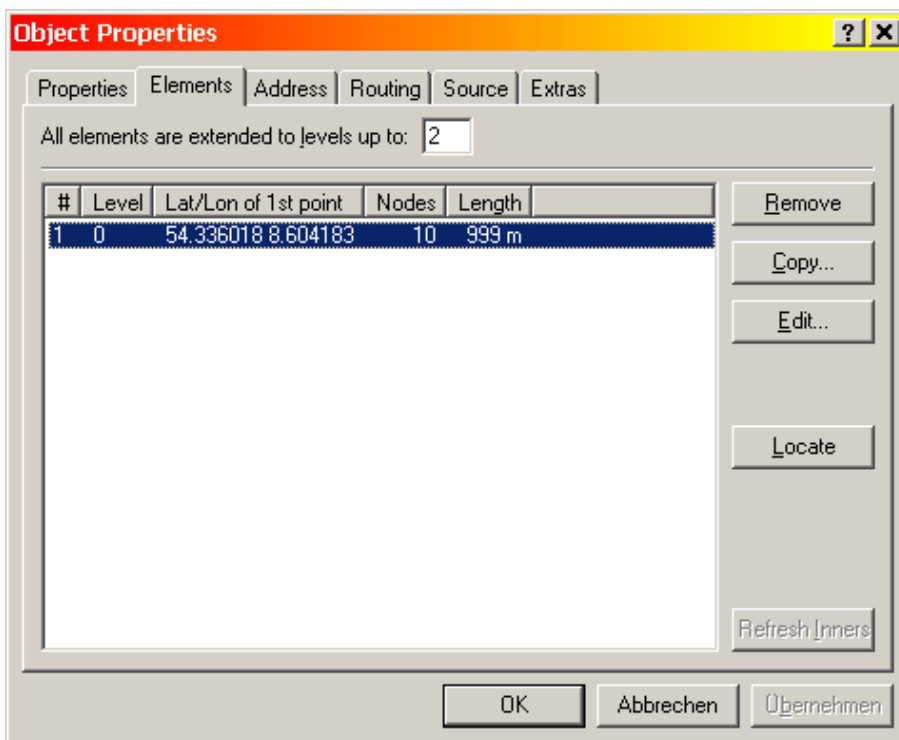
1. eine Erweiterung für Karten wie CS, CN oder Topo D V1
2. obligatorisch für Karten wie z.B. Topo D V2 um die erweiterten Typen überhaupt sichtbar zu machen

Um das Erscheinungsbild einer Karte zu verändern wird an den IMG-Dateien nichts geändert. Das ist bei geschützten Karten auch nicht möglich. Selbstverständlich ändert eine TYP-Datei nur die darin beschriebenen Elemente. Eintragungen in die TYP-Datei ohne Gegenstück in den IMG-Dateien schaden nicht. Jeder Kartensatz hat eine eigene (oder auch keine) TYP-Datei. Der Name der Datei ist egal, die FamilyID muss aber stimmen. Jede TYP-Datei muss in die Windows-Registry eingetragen sein. Damit ist die Darstellung am PC verändert. Alle gewünschten Karten werden in MapSource aus allen Kartensätzen ausgewählt. Beim Kopieren dieser Karten auf das Navigationsgerät werden alle Karten, und automatisch alle zugehörigen TYP-Dateien, in eine Datei gepackt (gmapsupp.img) und in das Gerät transferiert (oder auf eine Speicherkarte). Damit ist die Geräteanzeige verändert. Bei Verwendung anderer Tools ist die Vorgehensweise anders, meist umständlicher.

8.2 Verwendung der Level

Ideal ist eine Karte in der alle Information in einer Ebene dargestellt wird, die auf Papier gedruckte Karte. GPS-Geräte und sogar der PC-Bildschirm erlauben eine derartige Darstellung nicht. Mal abgesehen von den Abmessungen würden aufgrund der beschränkten Auflösung alle Details zu einem unleserlichen Brei verschmelzen. Also muss gerade soviel Information weggelassen werden, dass es beim Herauszoomen zu einer übersichtlichen Darstellung zu kommt. Was weggelassen wird bestimmt der Autor der Karte durch Angabe des Levels. Die Linien sind kritisch weil sie POIs, Flächen und andere Linien verdecken. Bei großen Karten bleibt nur eine schematische Vorgehensweise. Der Kreativität sind dabei enge Grenzen gesetzt. Ein Beispiel: Ein Wanderer möchte seine Wege (0x16) sehen und sie im Level 1 noch darstellen. Anstelle der Wege die kleinen Straßen (z.B. 0x06) auszublenden ergibt eine Karte, die unübersichtlich und nicht mehr brauchbar ist.

Die grobe Aufteilung Wege und Höhenlinien als erstes wegzulassen, dann die kleinen Straßen hat sich bewährt. Die Steuerung erfolgt bei jedem Objekt über die Einträge 'Levels=...' oder 'Endlevel=...' in der MP-Datei. Ohne diese Angabe ist das Objekt nur im Level 0 sichtbar. In GPSMapEdit durch Angabe von 'All elements are extended to levels up to:' :



Die Straße im Beispiel wird angezeigt bis zum Level 2.

Um die Übersicht zu verbessern dürfen nur wenige Objekte in den nächst höheren Level übernommen werden. Karten, wie die Topo Deutschland haben 3 aktive Level, d.h. mit anzuzeigenden Objekten. Das ist eine durchaus sinnvolle Aufteilung. In steilem Gelände mit 20m-Höhenlinienabstand kann es sinnvoll sein den Level 0 aufzuteilen. Die 20m Höhenlinien, Hecken, ... und andere weniger wichtige Objekte werden ab Level 1 ausgeblendet. In befestigte Straßen (0x06 ... 0x09) und unbefestigte Straßen (0x0a) aufzuteilen bringt dagegen sehr wenig.

Der Level 0 enthält immer alle Informationen der Karte. Getrennte Objekte in den verschiedenen Levels würde beim Bearbeiten unnötigen Aufwand erzeugen. Verfolgt man das Konzept aller Information im Level 0 mit Angabe bis zu welchem Level das Objekt dargestellt werden soll erzielt man optimale Ergebnisse und vermeidet Fehler bei der Bearbeitung. Die Reduktion der Information im Level 1 und höher erfolgt automatisch durch die Reduktion der verwendeten Bits / Koordinate. Zusätzlich wird beim Compilieren eine Vereinfachung von Linien und Polygonen vorgenommen, die sich an den Bits / Koordinate des betreffenden Levels orientiert. Linien und Polygone die durch die Kompression auf einen Punkt zusammenschrumpfen werden ignoriert.

8.3 Sehr große Karten

Karten sollten klein gehalten werden. 15 MByte sind für eine MP-Datei genug. Größere Dateien verlängern nur die Bearbeitungszeit unnötig. Es gibt einige Beschränkungen in der Größe von Karten, in jedem Fall aber eine Lösung für das Problem. Der Abstand zwischen zwei Punkten einer Linie oder eines Polygons ist auf einen 16-Bit-Wert beschränkt. Bei sehr großen Kacheln kann der Hintergrund Type=0x4b diesen Wert überschreiten. Wird eine Fehlermeldung

```
Error: map too large for 'Level0=xx' !
```

angezeigt, muss das Hintergrund-Polygon geteilt werden oder die Bits / Koordinate verkleinert werden.

8.4 Komplizierte Linien und Polygone

Haben Linien oder Polygone mehr als 255 Punkte werden sie in MapSource richtig dargestellt. Lädt man die Kachel aber in z.B. GPSMap60C fehlen diese Objekte. Derartige Objekte sollten beim Bearbeiten fertiger Karten nur durch eingefügte Tracks oder Höhenlinien vorkommen. Beim Generieren der IMG-Datei wird eine Warnung unter Angabe der Koordinaten des ersten Punktes der betreffenden Objekte ausgegeben.

```
Warning: polyline type=0x21 '120 M'  
        at 47.34039,10.207110 has > 255 nodes !
```

Linien und Polygone mit mehr als 255 Punkten in Detailkarten werden mit 'Reorganize' oder 'Script' automatisch zerteilt. Straßen / Wege müssen manuell geteilt werden.

Die Funktionen in GPSMapEdit 'Tool → Generalize', 'Remove Object Duplicates' und 'Merge Inner Polygons' vereinfachen die Karte ebenfalls. 'Generalize' sollte erst benutzt werden wenn alle Knotenpunkte für die Routen-Berechnung gesetzt wurden.

8.5 Verzeichnisse

MapTk speichert Daten vorzugsweise an zwei Orten:

1. Verzeichnis von dem aus das Programm aufgerufen wurde (Quellen):

- MP-Dateien,
- BAK-Dateien von 'Script' und die
- PRJ-Datei

MapTk soll aus diesem Verzeichnis heraus aufgerufen werden. Dieses Verzeichnis sollte regelmäßig gesichert werden.

2. Verzeichnis der Karten wie sie MapSource verwendet:

- IMG-Dateien,
- TDB-Datei
- TYP-Datei
- MDX- und MDR-Dateien und
- REG-Datei (nicht benutzt aber für MapSource)

Der Speicherort dieser Dateien kann in der PRJ-Datei an von den Quelldateien unterschiedlichen Ort definiert werden ('Binary files at'). Übersichtlicher ist jedoch ein gemeinsames Verzeichnis. Die Dateien können jederzeit aus den Quellen neu erzeugt werden. Die Sicherung dieses Verzeichnisses ist deshalb nicht unbedingt erforderlich.

Diese Trennung in zwei Verzeichnisse ist nicht notwendig, fördert aber die Übersichtlichkeit bei vielen Dateien. Es ist erlaubt alle Dateien in einem Verzeichnis zu halten.

8.6 Arbeiten mit vielen Dateien

Die Funktionen der Menüs 'Functions' und 'Tools' erlauben den Batch-Betrieb. Bei der Auswahl von Dateien können mit den Windows-Üblichen Tasten / Maus-Klick-Kombinationen mehrere Datei ausgewählt werden. Sollen mehreren hundert Dateien bearbeitet werden könnten Probleme auftreten. Es wird eine Fehlermeldung ausgegeben.

Für die 'Make'-Funktion stehen alle Dateien in einer Liste. Die mögliche Zahl der Dateien ist hier nicht beschränkt.

8.7 Alle Dateien in einem Verzeichnis

Alle Dateien können für 'Make' im selben Verzeichnis stehen. [Project] in der PRJ-Datei sieht dann wie folgt aus:

```
# alles im selben Verzeichnis

[
[Project]
Product=STirol
FamilyID=202
Version=201
Compile=1
Index=1
Style=1
Overview=basemap.img
IMGfiles=002021*.img
Copyright=Mit Genehmigung von ...
[END]

# IMGs

[IMG]
# Liste aller MP-Files
[End]
```

Der Eintrag 'IMGpath=' entfällt oder enthält './' für das aktuelle Verzeichnis.

8.8 Script

Beispiel-Script für 5 Level. Level 0 und 1 unterscheiden sich nur durch die Höhenlinien Type=0x20, die im Level 1 nicht mehr dargestellt werden.

```
[CUSTOM_HEADER]
# Python script: customizing cities
# variables:      overview      1 overview map
#                  0 detailed map
#                  transparent   1 map is transparent
#                  0 map is not transparent
#                  copyright     copyright in MapSource
#                  levels        number of levels in map
#                  level[]       bit / coord, index is level
if overview:      # modify in overview map
```

```

    levels = 4
    level[0] = 17
    level[1] = 15
    level[2] = 13
    level[3] = 11
else:                                     # modify in detailed map
    levels = 5
    level[0] = 23
    level[1] = 22
    level[2] = 21
    level[3] = 19
    level[4] = 17                         # for polygon 0x4b only
    transparent = 1
    routing= Y                           # map for automatic routing
    domain=(1,2,3)                       # polygons type=1,2 or 3 for POI-city
copyright = 'MapTk'                     # in MapSource
[END]

[CUSTOM_POI]
# Python script: customizing points
# variables:      overview    1 overview map
#                  0 detailed map
#                  type       code of object ( 16 bit )
#                  level      visibility up to level, -1: remove
#                  label      text for object
if overview:
    grp = type >> 8                     # modify in overview map
    if type == 0x2f04:                  # group of types
        type = 0x2d0b                 # airport (invisible)
        # -> visible
    if type < 0x0b00:
        level = 2
    elif type < 0x1200:
        level = 1
    else:
        level = 0
else:                                     # modify in detailed map
    grp = type >> 8                     # group of types
    if type == 0x2f04:                  # airport (invisible)
        type = 0x2d0b                 # -> visible
    if type < 0x0b00:
        level = 3
    elif type < 0x1200:
        level = 2
    elif type == (0x2f0b, 0x6616):      # parking, summit
        level = 2
    elif grp in (0x2c, 0x62, 0x63):    # spots
        level = 2
    else:
        level = 1
    # add first time a city name for a POI inside a city polygon
    if index and label and domain:
        if 'CityName' not in ps:
            ps['CityName'] = domain
[END]

[CUSTOM_POLYLINE]
# Python script: customizing polylines
# variables:      overview    1 overview map
#                  0 detailed map
#                  type       code of object ( 8 bit )
#                  level      visibility up to level, -1: remove

```

```

#          label      text for object
if overview:                                     # modify in overview map
    if type in (1, 0x1e):
        level = 2
    elif type in (2, 0x1c):
        level = 1
    elif type in (3, 0x18, 0x1a, 0x1b, 0x1f, 0x26):
        level = 0
    else:
        level = -1                                # remove
else:                                             # modify in detailed map
    if type == 0x18 and label <> '':             # with name -> Level 2
        type = 0x1f
    if type in (0x20, 0x21, 0x22):               # assign land contour
        if label.find('000 ') != -1:
            type = 0x22
        elif label.find('00 ') != -1:
            type = 0x21
        else:
            type = 0x20
    # change routing parameter for Italian highways to toll / limited speed
    if routing:
        if type == 1:
            speed = 5                             # average of 90 km/h
            restrict[0] = 1                       # is toll
    if type in (1, 2, 3, 4, 5, 0x14, 0x1c, 0x1e):
        level = 3
    elif type in (6, 7, 8, 9, 0x0a, 0x0b, 0x0c, 0x13, 0x1a, 0x1b,
        0x1f, 0x27, 0x28):
        level = 2
    elif type in (0x16, 0x18, 0x1d, 0x21, 0x22, 0x24, 0x25, 0x26,
        0x29):
        level = 1
    else:
        level = 0
[END]

[CUSTOM_POLYGON]
# Python script: customizing polygons
# variables:      overview      1 overview map
#                  0 detailed map
#                  type         code of object ( 8 bit )
#                  level        visibility level, -1: invisible
#                  label        text for object
if overview:                                     # modify in overview map
    if type == 0x4b:
        level = 3
    elif type in (1, 2, 0x28, 0x3c, 0x3f, 0x40, 0x42, 0x43, 0x44,
        0x46, 0x47, 0x48, 0x4a):
        level = 2
    elif type in (3,):
        level = 1
    else:
        level = 0
else:                                             # modify in detailed map
    if type == 0x4b:                             # background
        level = 4
    elif type in (1, 0x28, 0x3c, 0x3d, 0x3e, 0x3f, 0x40, 0x42,
        0x43, 0x44, 0x46, 0x47, 0x48):
        level = 3
    elif type in (2, 3, 4, 5, 6, 7, 8, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e,

```



```
        0x13, 0x14, 0x15, 0x18, 0x19, 0x50):  
    level = 2  
elif type in (0x16, 0x17, 0x1a, 0x41, 0x49, 0x4d, 0x4e, 0x4f,  
              0x51, 0x51, 0x52):  
    level = 1  
else:  
    level = 0                # only level 0  
[END]
```

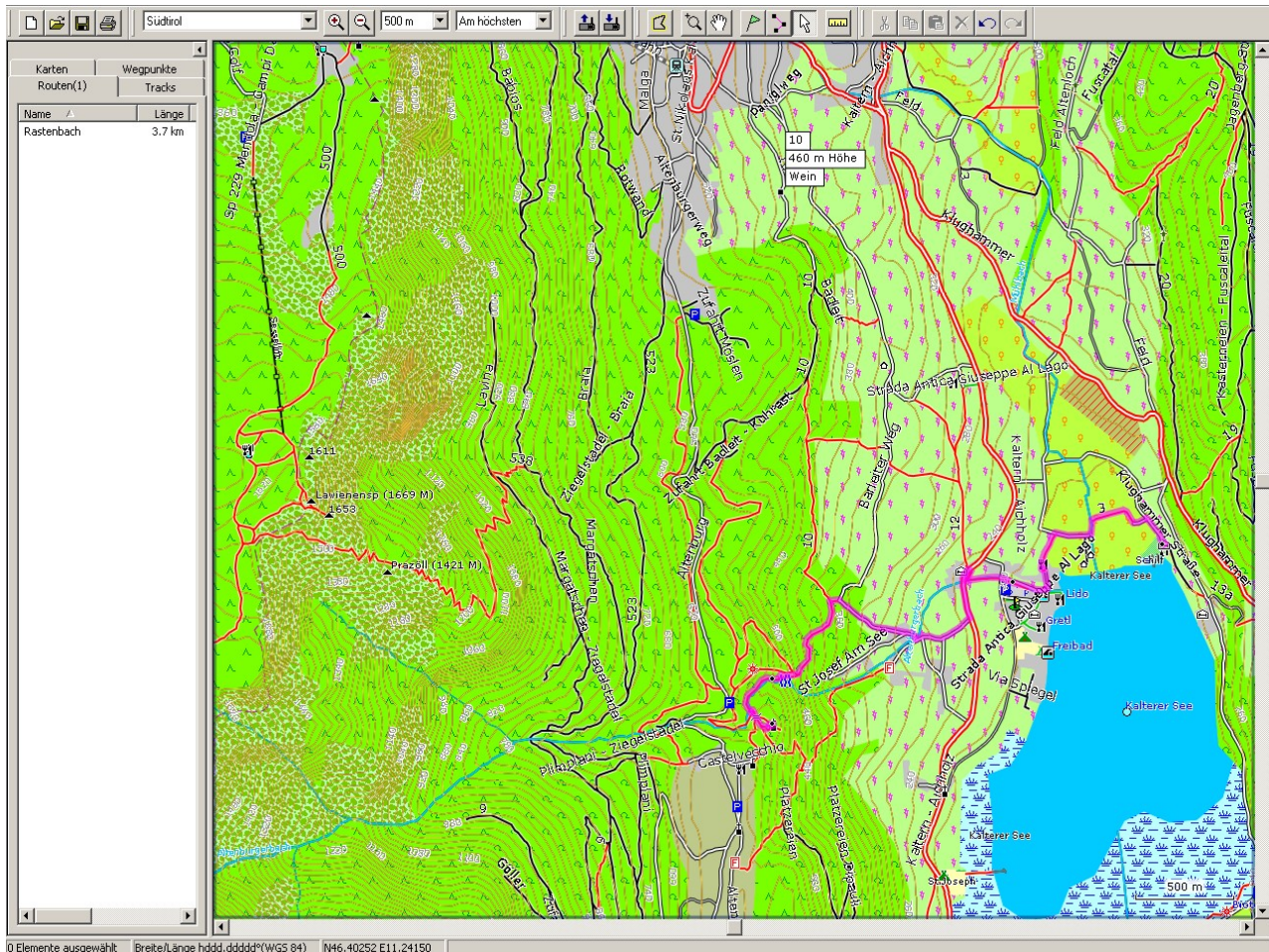
9 Anhang

9.1 Wissenswertes

Wissenswerte Fakten, die in der Zusammenfassung beim Einstieg in die Erstellung von Kartensätzen hilfreich sein könnten. Das sind Notwendigkeiten oder auch nur Hinweise die der Übersichtlichkeit oder Bequemlichkeit dienen.

- In MapTk ist ein Kartensatz ein Projekt, egal ob eine oder 123 Kachel vorhanden sind. Quelle und Daten für MapSource / BaseCamp können in unterschiedlichen Verzeichnissen stehen, müssen aber nicht. Mit mehreren PJ-Dateien können mehrere Projekte auch in einem Verzeichnis stehen. Vorzug: 1 Projekt = 1 Verzeichnis. MapTk soll immer mit der PRJ-Datei aufgerufen werden.
- Kacheln eines Kartensatzes haben eine ID (ID=...). Detailkarten: Dezimalzahl mit genau 8 Stellen (z.B. 01234567), Übersichtskarte: bis 8 Zeichen beginnend mit einem Buchstaben (vorzugsweise 'basemap'). Der Name (Name=...) einer Kachel oder Übersicht ist frei und kann zur besseren Übersicht gleich dem Namen der MP-Datei sein.
- Die eindeutige FamilyID (FID) ist nur im Kopf des Projektes definiert. Sie steht im fertigen Kartensatz (nach 'Make') in der TDB, MDX, der TYP-Datei und in der Registry. Diese FID darf auf dem PC oder GPS-Gerät nur einmal vorkommen.
- Kartensätze für MapSource oder Basecamp brauchen immer eine Übersichtskarte. Sie kann einfach aus den MP-Dateien des Projektes erzeugt werden. Sie wird spätestens nach Erzeugung in die Liste 'Maps' eingetragen, zu allen anderen Dateien des Projektes. Die Übersichtskarte muss kompiliert werden (z.B. mit 'Make').
- 'Make' aktualisiert nach jeglicher Änderung am Projekt die betroffenen Dateien unter Berücksichtigung der Abhängigkeiten. Nach Änderung nur der FID werden z.B. TYP, TDB, MDX, Index und REG-Datei aktualisiert oder neu erzeugt.
- Die Erfahrung sagt, dass die Begutachtung während der Entwicklung der Karte mittels MapSource / Basecamp völlig ausreichend ist. Eine Übertragung an das GPS-Gerät zum abschließenden Test sollte dennoch stattfinden.

9.2 Beispiel-Karte



Die Karte wurde mit MapTk erzeugt, mit einer TYP-Datei verschönert und eine automatische Route wurde berechnet.

9.3 TYP / Symbol

0x0000 : 'None',	0x0a00 : 'City (Medium)',
0x0100 : 'City (Capitol)',	0x0b00 : 'City (Medium)',
0x0200 : 'City (Capitol)',	0x0c00 : 'City (Medium)',
0x0300 : 'City (Capitol)',	0x0d00 : 'City (Small)',
0x0400 : 'City (Large)',	0x0e00 : 'City (Small)',
0x0500 : 'City (Large)',	0x0f00 : 'City (Small)',
0x0600 : 'City (Large)',	0x1000 : 'City (Small)',
0x0700 : 'City (Large)',	0x1100 : 'City (Small)',
0x0800 : 'City (Medium)',	0x1602 : 'Short Tower',
0x0900 : 'City (Medium)',	0x1610 : 'Flag, Red',

0x1610 : 'Pin, Red',	0x2c05 : 'School',
0x1611 : 'Flag, Green',	0x2c06 : 'Park',
0x1611 : 'Pin, Blue',	0x2c07 : 'Zoo',
0x1611 : 'Pin, Green',	0x2c08 : 'Fitness Center',
0x1615 : 'Flag, Blue',	0x2c09 : 'Building',
0x2700 : 'Exit',	0x2c0a : 'Winery',
0x2a00 : 'Restaurant',	0x2c0b : 'Church',
0x2a01 : 'Restaurant',	0x2d01 : 'Live Theater',
0x2a02 : 'Restaurant',	0x2d02 : 'Bar',
0x2a04 : 'Restaurant',	0x2d03 : 'Movie Theater',
0x2a06 : 'Restaurant',	0x2d05 : 'Golf Course',
0x2a07 : 'Fast Food',	0x2d06 : 'Skiing Area',
0x2a08 : 'Restaurant',	0x2d07 : 'Bowling',
0x2a0a : 'Pizza',	0x2d08 : 'Ice Skating',
0x2a0b : 'Restaurant',	0x2d09 : 'Swimming Area',
0x2a0c : 'Restaurant',	0x2d0a : 'Ball Park',
0x2a0f : 'Restaurant',	0x2d0a : 'Stadium',
0x2a10 : 'Restaurant',	0x2d0b : 'Airport',
0x2a11 : 'Restaurant',	0x2e00 : 'Shopping Center',
0x2a12 : 'Restaurant',	0x2e01 : 'Department Store',
0x2b00 : 'Lodging',	0x2e04 : 'Restroom',
0x2b01 : 'Lodging',	0x2e05 : 'Pharmacy',
0x2b02 : 'Lodging',	0x2e06 : 'Convenience Store',
0x2b03 : 'Campground',	0x2f00 : 'Car',
0x2b04 : 'Lodge',	0x2f01 : 'Gas Station',
0x2b0f : 'Tunnel',	0x2f02 : 'Car',
0x2c00 : 'Scenic Area',	0x2f03 : 'Car Repair',
0x2c01 : 'Amusement Park',	0x2f05 : 'Post Office',
0x2c01 : 'RV Park',	0x2f06 : 'Bank',
0x2c02 : 'Museum',	0x2f07 : 'Car Rental',
0x2c04 : 'Dot, White',	0x2f08 : 'Ground Transportation',

0x2f09 : 'Anchor',
0x2f0a : 'Wrecker',
0x2f0b : 'Parking Area',
0x2f0c : 'Restroom',
0x2f16 : 'Truck Stop',
0x2f18 : 'Ground Transportation',
0x3000 : 'City Hall',
0x3001 : 'Police Station',
0x3002 : 'Medical Facility',
0x3003 : 'City Hall',
0x3006 : 'Toll Booth',
0x3008 : 'Civil',
0x3009 : 'Church',
0x4500 : 'Restaurant',
0x4600 : 'Bar',
0x4900 : 'Park',
0x4a00 : 'Picnic Area',
0x4c00 : 'Information',
0x4f00 : 'Shower',
0x5000 : 'Drinking Water',
0x5100 : 'Bell',
0x5200 : 'Scenic Area',
0x5300 : 'Skiing Area',
0x5400 : 'Swimming Area',
0x5903 : 'Airport',
0x5904 : 'Heliport',
0x5a00 : 'Mile Marker',

0x5b00 : 'Bell',
0x6300 : 'Summit',
0x6400 : 'Triangle, Red',
0x6401 : 'Crossing',
0x6402 : 'Building',
0x6402 : 'Residence',
0x6403 : 'Cemetery',
0x6404 : 'Church',
0x6405 : 'Civil',
0x6406 : 'Crossing',
0x6408 : 'Medical Facility',
0x640b : 'Military',
0x640c : 'Mine',
0x640d : 'Oil Field',
0x6411 : 'Tall Tower',
0x6412 : 'Bike Trail',
0x6413 : 'Tunnel',
0x6416 : 'Scenic Area',
0x6419 : 'Trail Head',
0x6500 : 'Triangle, Blue',
0x6508 : 'Triangle, Blue',
0x6600 : 'Triangle, Green',
0x660a : 'Forest',
0x660b : 'Triangle, Green',
0x6611 : 'Summit',
0x6616 : 'Summit',
0x6619 : 'Scenic Area',

9.4 Benutzer definierte Typen

Die Typen sind nicht vordefiniert und werden in verschiedenen Kartensätzen unterschiedlich verwendet. Die folgenden Objekte sind in der 'Topo Deutschland V2' neu oder anders belegt als in der Version 1. Die neuen Objekte sind in den IMG-Dateien an einer anderen Stelle gespeichert als die bisher. Zur Unterscheidung ist im MapTk zur Unterscheidung das Bit 24 gesetzt (z.B. 0x1101d für Nadelwald). Besonderheiten dieser Typen:

- sie **müssen** in der TYP-Datei deklariert werden um überhaupt dargestellt zu werden.
- sie können mit GPSMapEdit seit Version 1.0.56 dargestellt und bearbeitet werden.
- sie können ab Version 2.6 mit MapTk in eine IMG-Datei übernommen werden
- sie werden nicht in den Index übernommen

Für die Bearbeitung in der PRJ-Datei gibt es sonst nichts zu beachten.

GPSMapEdit zeigt diese Typen mit Hilfe einer Skin-Datei in der Objektauswahl mit Code, Text und Icon an. Vor der Bearbeitung einer Karte sollten deshalb zuerst neue Typen für das Projekt erzeugt, und daraus eine Skin-Datei erzeugt werden.

9.4.1 POI

11500	SCHLEUSE
11501	GEBÄUDE
11502	HÖHLE
11503	STOLLENMUNDLOCH/SCHACHTÖFFNUNG
11504	KRAN
11505	PUMPE
11506	WINDRAD
11507	ARCHÄOLOGISCHE FUNDSTÄTTE
11508	DENKMAL
11509	KREUZ/BILDSTOCK
1150a	MEILENSTEIN
1150b	BAUM
1150c	PLATZ
1150d	HUBSCHRAUBERLANDEPLATZ
1150e	ANLEGER
1150f	BAKE
11510	LEUCHTFEUER
11511	FURT
11512	KILOMETRIERUNG
11513	ANTENNENMAST
11514	HALTESTELLE
11515	BERGBAUWANDERWEG

9.4.2 Polyline

00d00	FUSSGÄNGERZONE
00e00	WEG
00f00	WIRTSCHAFTSWEG
01300	STEIG
10e00	UNTERIRDISCHER WASSERLAUF
10e01	SICKERSTRECKE
10e02	STAUMAUER

10e03	WEHR
10e04	SCHLEUSE
10e05	SCHLEUSENKAMMER
10e06	UFERBEFESTIGUNG
10e07	KAIMAUER
10e08	BUHNE
10e09	LAHNUNG
10e0a	MOLE
10e0b	SPUNDWAND
10e0c	WASSERFALL
10e0d	SCHMALSPURBAHN
10e0e	S-BAHN
10e0f	STRASSENBAHN
10e10	U-BAHN
10e11	BERGBAHN
10e12	FREIZEITBAHN
10e13	MAGNETSCHWEBEBAHN
10e14	SEILBAHN
10e15	SESSELLIFT
10e16	SKILIFT
10e17	MATERIALSEILBAHN
10e18	TUNNEL
10e19	TUNNEL
10e1a	DURCHLASS
10e1b	DURCHLASS
10e1c	ANLEGESTELLE
10e1d	BRÜCKE
10e1e	FURT
10e1f	FÖRDERBAND
10f00	DAMM/DEICH
10f01	MAUER
10f02	STÜTZMAUER
10f03	ZAUN
10f04	FELS
10f05	GRADIERWERK
10f06	KRAN
10f07	ARCHÄOLOGISCHE FUNDSTÄTTE
10f08	DENKMAL
10f09	RENNBAHN
10f0a	SPRUNGSCHANZE
10f0b	BAUMREIHE
10f0c	HECKE
10f0d	HALDE
10f0e	NATIONALPARK
10f0f	SCHUTZGEBIET
10f10	GEBIETE
10f11	MARKIERTER WANDERWEG

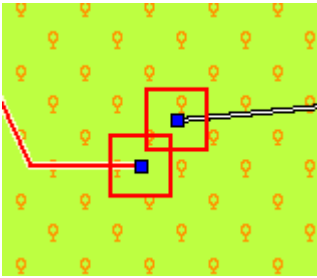
9.4.3 Polygon

10f00	DAMM/DEICH
10f01	UFERBEFESTIGUNG
10f02	SPERRWERK
10f03	BRÜCKE/ÜBERFÜHRUNG
10f04	GEBÄUDE
10f05	GEBÄUDE
10f06	ÖFFENTLICHES GEBÄUDE
10f07	SPUNDWAND
10f08	INDUSTRIE- UND GEWERBEFLÄCHE
10f09	CAMPINGPLATZ
10f0a	STROMSCHNELLE

10f0b	SONDERKULTUR
10f0c	BAUMSCHULE
10f0d	HOPFEN
10f0e	WEINGARTEN
10f0f	OBSTBÄUME
10f10	TALSPERRE/WEHR
10f11	GERÖLL
10f12	TROCKENDOCK
10f13	FESTPLATZ
10f14	LAHNUNG
10f15	SPIELFELD/SPIELFLÄCHE
10f16	WIESE
10f17	SCHOTTER
10f18	GEHÖLZ
10f19	GEHÖLZ (NADELHOLZ)
10f1a	GEHÖLZ (LAUBHOLZ)
10f1b	WELLENBRECHER/BUHNE
10f1c	HEIDE
10f1d	HUBSCHRAUBERLANDEPLATZ
10f1e	SCHLEUSE
10f1f	SCHLEUSENKAMMER
11000	MISCHBEBAUUNG
11001	MODELLFLUGGELÄNDE
11002	HAFENDAMM
11003	TAGEBAU/GRUBE/STEINBRUCH
11004	FUSSGÄNGERZONE
11005	ANLEGESTELLE
11006	HAFEN
11007	HAFENBECKEN
11008	KAIMAUER
11009	FREIZEITANLAGE
1100a	WOHNGEBIET
1100b	RASTPLATZ
1100c	FELS
1100d	SAND
1100e	SANDBANK
1100f	ABSETZBECKEN/SCHLAMMTEICH
11010	RIESELFELD
11011	SCHIFFSHEBEWERK
11013	SOLARZELLEN
11014	SONDERFLÄCHE
11015	STADION/SPORTPLATZ
11016	BAHNHOFSANLAGE
11017	SCHWIMMBAD
11018	WATT
11019	PRIEL
1101a	SCHÖPFWERK
1101b	WASSERFALL
1101c	NASSER BODEN
1101d	NADELWALD
1101e	LAUBWALD
1101f	MISCHWALD

9.5 'Skin' für GPSTMapEdit

Definition für ein POI 'Typ=0x0000' als Marker zum Anzeigen von fehlenden Verbindungen oder nicht passenden externen Knoten für automatisches Routen:

[illegible]

[end]

[POI]

```
String=0,Marker
```

[illegible]

```
Line=00000000000000000000000000000000
```

Line=00.....00

```
Line=00.....00
```

```
Line=00.....00
```

Line=00.....00

Line=00.....00

```
Line=00.....00
```

```
Line=00.....00
```

```
Line=00.....00
```

```
Line=00.....00
```

```
Line=00.....00
```

```
Line=00.....00
```

```
Line=00.....00
```

```
Line=00.....00
```

```
Line=00.....00
```

Line=00.....00

Line=00.....00

Line=00.....00

Line=00.....00

Line=00.....00

Time=00.....00

```
Time=00.....00
```

```
Time=00.....00
```

```
Time=00.....00
```

```
Time=00.....00
```

```
Time=00.....00
```

```
Time=00.....00
Time=00.....00
```

```

Line 99.....99
Time=00                                00

```

```

Line 99.....99
Time=00                                00

```

```

LINE 00.....00
Time=0000000000000000000000000000000000

```

```
LINE=0000000000000000000000000000000000000000000000000000000
Time=0000000000000000000000000000000000000000000000000000000
```

[End]

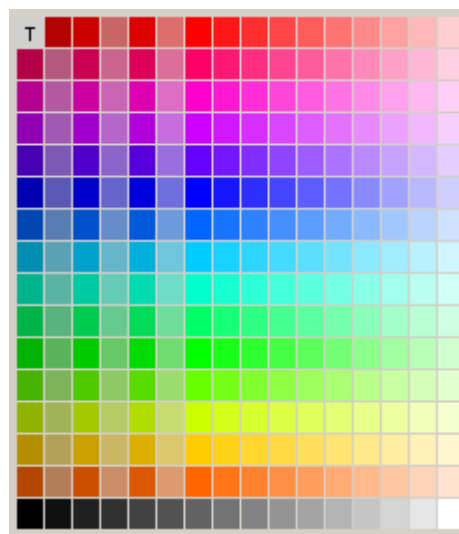
9.6 Farbpalette (256 Farben)

Kombination (240 Farben)			Grauwerte	
rot	grün	blau		
00	00	00	000000	schwarz
39	30	20	101010	
7B	65	41	202020	
BD	95	6A	313131	
FF	CA	8B	414141	
	FF	B4	525252	
		D5	626262	
		FF	737373	
			838383	
			949494	
			A4A4A4	
			B4B4B4	
			C5C5C5	
			D5D5D5	
			E6E6E6	
			FFFFFF	weiß
			D0D0D0	transparent

Diese Farben werden von Garmin für die selbst definierten POIs verwendet. Damit sollte es sicher sein, dass auch ältere Gräte damit funktionieren. Eine alternative Palette mit helleren Farben ist im 'Header' des Projektes auswählbar.



Garmin-Farben



MapTk-Farben

Nur Farben der aktuellen Palette werden mit dem graphischen Editor ausgewählt. Falls eine PRJ-Datei andere RGB-Werte enthält, werden diese bei der Auswahl des Objektes, auf die nächste vergleichbare Farbe aus aktuellen Palette gerundet. Mischen von Farben der 2 Paletten in einer PRJ-Datei sollte vermieden werden um ungewollte Farbveränderungen zu vermeiden. 'Save' speichert die momentane Farbe in der Projektdatei, 'Cancel' schließt das Fenster ohne zu speichern.

9.7 Sprach-Codes

0 keine	9 baskisch	18 tschechisch	27 lettisch
1 französisch	10 katalanisch	19 kroatisch	28 rumänisch
2 deutsch	11 galizisch	20 ungarisch	29 albanisch
3 holländisch	12 walisisch	21 polnisch	30 bosnisch
4 englisch	13 gälisch	22 türkisch	31 litauisch
5 italienisch	14 dänisch	23 griechisch	32 serbisch
6 finnisch	15 norwegisch	24 slowenisch	33 makedonisch
7 schwedisch	16 portugiesisch	25 russisch	34 bulgarisch
8 spanisch	17 slowakisch	26 estnisch	

Stichwortverzeichnis

255 Punkte.....	60	Exit.....	8
3-Byte-Typen.....	5f., 48ff.	externe Knotenpunkte.....	27
3-point-average.....	12	externer Knotenpunkt.....	14
Analyse.....	6	FamilyID.....	43, 51
Anhänge.....	19	Farbdefinition.....	14
Arbeitsverzeichnis.....	29	Farben.....	73
Aufrufparameter.....	5	Farbpalette.....	22
Backup registry.....	15	Feet.....	19
Batch.....	61	for.....	57
bedingte Anweisung.....	56	Funktionen.....	18, 56
Berechnung von Routen.....	5	gedruckte Karte.....	58
BorderWidth.....	22, 45	Generalize.....	60
Build project.....	9	geometrisches Mittel.....	12
Catmull-Rom spline filter.....	12	global.....	56
City.....	19	GMAP.....	5
CityIdx.....	19	gmapsupp.img.....	58
CityName.....	19	GPSMap60C.....	21
Codepage.....	8	GPSMapEdit.....	4, 9, 23, 48
CodePage.....	18, 48	GPSMapEdit skin.....	15
Color.....	44ff.	GPX.....	10
Colorado.....	21	GPX track filter.....	12
Combine.....	10	GPX track to MP.....	13
Combine GPX files.....	10	Grauwerte.....	73
Compile map.....	9	große Karten.....	60
Compile TYP.....	14	Hausdorf-Distance.....	12
Copyright.....	19, 43, 48	Help.....	16
CountryName.....	19	Hintergrund.....	20
CP1251.....	8	Homepage.....	16
CP1252.....	8	Icon library.....	14
Darstellungsreihenfolge.....	45	if.....	57
Data0.....	19	IMG.....	9, 58
Delete from registry.....	16	IMG analysis.....	9
Detaillkarte.....	20	IMG reset file transparent.....	10
Dictionary.....	55	IMG set file transparent.....	10
dirindicator.....	55	IMG-Datei.....	9, 20, 50
domain.....	54f., 62	IMGfiles.....	43
Douglas-Peuker-Filter.....	12	IMGPath.....	43
DrawOrder.....	45	Import.....	31
Drawpriority.....	48	in.....	57
DrawPriority.....	10, 18	index.....	55, 62
Edit project file.....	9	Index.....	6, 16, 21, 30, 50, 52, 68
Edit text file.....	8	indizierte Variable.....	56
Editor.....	29	Info.....	16
Elevation.....	18	Installation.....	4
elif.....	57	Kartensatz.....	6
else.....	57	Kommentare.....	19
eMap.....	21	Kopierschutz.....	7
Erscheinungsbild.....	18	label.....	55

LblCoding.....	18	Projektdatei.....	42
level.....	55	Projektverzeichnis.....	9, 18
Level.....	58	Prozess.....	17
Levels.....	19	ps.....	54
Levelx.....	19	Punkte.....	22
Line.....	44ff.	Python.....	55
LineWidth.....	22, 45	Python in der.....	5
Linien.....	22	Python-Imaging-Library.....	5
Liste aller Typen.....	15	Python-Script.....	18, 53
locked.....	7, 9	Rasterbild.....	14
Log-Funktion.....	8, 52	REG-Datei.....	21f., 51
Make.....	9, 21	RegionName.....	19
MapTk.....	16	Registry.....	16, 21f., 51
MapTk_*.pdf.....	16, 42	Reorganize.....	9
MapTk.dat.....	42	restrict.....	55, 63
Maptk.exe.....	42	Restrict.....	26
MapTk.prj.....	9, 21	Restriktion.....	25
Marinekarten.....	7	return.....	56
Marker.....	71	RGN10.....	19
Mask GPX tracks by MP.....	13	RGN20.....	19
MDR.....	60	RGN40.....	19
MDX.....	60	RGN80.....	19
Meter.....	19	roadclass.....	55
mp.....	54	RoadID.....	26, 49
MP to GPX.....	10	Routen-Berechnung.....	25
MP-Datei.....	20, 48	Routen-Probleme.....	27
Nachtfarbe.....	14	RouteParam.....	26, 50
Namenskonventionen.....	6	routing.....	54, 62f.
New project file.....	9	Routing.....	19, 26
No invalid objects.....	27	Script.....	9, 39
oneway.....	55	Skin.....	27, 71
Oregon.....	21	speed.....	55, 63
Origin0.....	19	Spline-Filter.....	12
orking directory.....	9	Split GPX.....	10
Orte.....	52	Sprach-Codes.....	73
overview.....	55	Sprache.....	8
overviewmap.....	10	Standard-Funktion.....	56
OverviewMap.....	43	status.append.....	56
Parabolic spline filter.....	12	Steuerzeichen.....	19
PDF-Viewer.....	8	Straßen und Wege.....	25
POI.....	22, 33, 52	Straßennetzwerk.....	26
Polygon.....	37	String.....	44ff.
Polyline.....	35	Subtyp.....	44
Pre-Prozessor.....	5	Tag 'cmt'.....	10
Preferences.....	8	Tasten.....	41
Preview.....	18	tcl8.5.....	5
PRJ-Datei.....	20, 22, 42, 51	TDB analysis.....	9
Product.....	43	TDB-Datei.....	51
Product ID.....	43	TextColor.....	23
ProductID.....	51	TextColor=.....	44ff.

TextSize.....	23, 44ff.
tk8.5.....	5
Track-Filter.....	12
Transparent.....	10, 18
typ.....	55
Typ.....	51
TYP.....	14, 58
Typ >= 0x10000.....	44f.
TYP-Datei.....	22, 51
Type.....	44f.
Type=0x00.....	28, 50
Type=0x4a.....	50
Typen > 0x10000.....	44
Übersichtskarte.....	20, 50
unsichtbar.....	22, 35
unverbundener Knotenpunkt.....	13
Variable.....	56
Verify Map.....	27
Version.....	43
Verzeichnis.....	6
Verzeichnisse.....	60
Vista.....	4
Windows.....	4
Wow6432Node.....	21
XOR IMG file.....	9
Zoomx.....	19
'TYP analysis'.....	14

[Cities].....	19
[Countries].....	19
[Custom_Header].....	46
[CUSTOM_HEADER].....	54
[Custom_POI].....	46
[CUSTOM_POI].....	54
[Custom_Polygon].....	46
[CUSTOM_POLYGON].....	54
[Custom_Polyline].....	46
[CUSTOM_POLYLINE].....	54
[End].....	46
[IMG ID].....	18, 48
[IMG].....	43
[POI].....	19, 44, 49, 51
[Polygon].....	51
[POLYGON].....	50
[Polygon].....	
.....	
.....	45
[Polyline].....	44, 51
[POLYLINE].....	49
[Project].....	9, 43, 51
[Regions].....	19
[RESTRICT].....	49
[RGN20].....	19