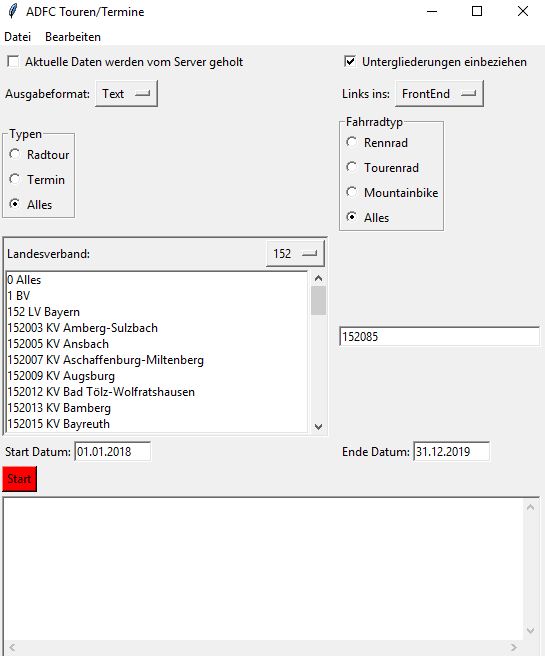
Programmdokumentation

Das Programm adfc\_gui.exe liest Daten zu Touren und Terminen (im folgenden zusammenfassend *Ereignisse* genannt) aus dem ADFC-Tourenportal aus, und faßt sie zu Text zusammen. Dieser Text kann dann weiterverarbeitet werden. Es werden verschiedene Ausgangsformate unterstützt. Die Beschreibung eines Ereignisses kann im Backend formatiert werden, und kann dann Text in verschiedenen Größen, mit Unterstrich oder Fettdruck, Titel, Aufzählungen oder Blocktext enthalten. Diese Beschreibung wird entweder unformatiert in das Ausgangsformat übernommen, oder, im Falle der Ausgabe ins Word-Format, mit möglichst der gleichen Formatierung.

Das Programm kann von <https://github.com/michaelu123/adfc2/releases> bezogen werden. Dazu muß die neueste Datei adfc\_gui.zip runtergeladen und entpackt werden. Das Programm muß nicht installiert werden. Nach Doppelklick auf die Datei adfc\_gui.exe öffnet sich nach kurzer Zeit die Programm-Oberfläche.

# Die Programm-Oberfläche



Von links oben nach rechts unten sehen wir diese Bedienelemente:

## Datei

Hat die Einträge *Speichern*, *Speichern unter*, und *Word Template*. Mit *Speichern* und *Speichern unter* wird der Text im unteren Textfenster in eine Datei geschrieben. Mit *Word Template* wird eine Word-Datei spezifiziert, die die Vorlage (das Template) für die zu erstellende Word-Datei enthält.

## Bearbeiten

Hat die Einträge *Ausschneiden*, *Kopieren*, *Einfügen*, *Suchen* und *Erneut suchen*. Damit kann der Text im unteren Textfenster bearbeitet und durchsucht werden. Man kann den Text natürlich auch erst speichern, und dann in einem anderen, mächtigeren Programm, weiterbearbeiten.

## Aktuelle Daten werden vom Server geholt

Wenn das Häkchen gesetzt ist, werden die Daten immer frisch aus dem Tourenportal geholt, und lokal gespeichert. Ist das Häkchen nicht gesetzt, wird erst geschaut, ob eine lokale Datei existiert, und wenn ja, diese gelesen, wenn nein, werden die Daten wieder aus dem Tourenportal geholt und gespeichert. Dazu unten im Absatz *Cache* mehr.

## Untergliederungen einbeziehen

Wenn das Häkchen gesetzt ist, werden auch die Untergliederungen (z.B. Ortsgruppen des Kreisverbands) der Gliederung(en) in die Suche mit einbezogen.

## Ausgabeformat:

Hat die Einträge *München*, *Starnberg*, *CSV*, *Text*, *Word*. Die ersten vier geben die Ereignisse als Text in das untere Textfenster aus. Die Unterschiede von München, Starnberg und Text sind nicht gravierend. Z.B. gibt *München* die Tour-Schwierigkeit als Zahl 1 bis 5 an, und *Starnberg* mit 1 bis 5 Sternchen. *Text* gibt die Schwierigkeit so wie im Tourenportal an: von sehr einfach bis sehr schwer. Bei *Text* erscheint vor jedem Ereignis das dazugehörige Vorschaubild, dieses wird aber beim Speichern nicht mitgespeichert. Gibt man *Word* an, und hat vorher, mit *Datei*/*Word Template* noch keine Vorlagendatei spezifiziert, muß man das jetzt nachholen.

## Links ins:

Hat die Einträge *Frontend*, *Backend* und leer. Im Ausgabeformat Word enthalten die Titel Links ins Tourenportal-Frontend oder -Backend, wenn der Eintrag nicht leer ist.

## Typen

Hiermit kann man festlegen, ob man nur Radtouren, nur Termine oder alles sucht.

## Fahrradtyp

Hiermit legt man fest, ob man nach Rennrad-, Tourenrad- oder Mountainbike-Touren sucht, oder nach allen. Wenn man nur nach Terminen sucht, ist dieses Menü inaktiv.

## Landesverband und Untergliederungen

Man kann oben einen Landesverband wählen und dann unten eine oder mehrere Gliederungen auswählen. Auch möglich ist das Auswählen des Bundesverbands (BV), oder *Alles*. Mehrere Gliederungen wählt man aus, indem man die Strg-Taste gedrückt hält, während man die Gliederungen anklickt. Rechts sieht man dann die dazugehörigen Gliederungsnummern. Diese kann man auch direkt eingeben. Die Liste links dient nur der einfacheren Auswahl der Gliederungsnummern in das rechte Gliederungsnummern-Feld.

## Start-Datum: und Ende-Datum:

Damit wird der Zeitraum eingegrenzt, in dem die Ereignisse stattfinden. Das Ende-Datum ist einschließlich. Sind beide Daten gleich, werden die Ereignisse dieses einen Tages gesucht. Das Format der Datumseingabe ist TT.MM.JJJJ.

## Start

Der rote Startknopf bewirkt, daß die Ereignisse der ausgewählten Gliederung(en) vom Tourenportal eingelesen, zeitlich aufsteigend sortiert, nach Typ und Fahrradtyp gefiltert und dann formatiert werden. Die Ausgabe erfolgt dann in das untere Textfenster, oder in eine Word-Datei. In letzterem Fall wird gleich das Programm gestartet, in dem Word-Programme auf Ihrem Rechner standardmäßig bearbeitet werden.

Sie können jetzt die Einstellungen ändern, z.B. das Ausgabeformat oder die Gliederungen, und Start erneut drücken.

# Scribus

Scribus ist ein freies, kostenloses Desktop-Publishing-Programm. Es ist über die Skript-Sprache Python, in der auch adfc\_gui programmiert ist, steuerbar. Als Scribus-script kann man die Datei adfc-rest2.py benutzen, das nach Gliederungsnummern, Start- und Ende-Datum fragt, und dann die Daten in das vorab selektierte Textfeld schreibt. Dieses Feature ist noch ausbaufähig, außerdem werden bisher nur Radtouren, keine Termine unterstützt. Das Ausgabeformat „Starnberg“ gibt die Daten, wenn auch unformatiert, so aus, wie sie das Skript adfc-rest2.py ausgeben würde.

# CSV

Das Ausgabeformat CSV speichert die Daten als eine CSV-Datei mit den Feldern Typ;Titel;Nummer;Radtyp;Tourtyp;Datum;Endedatum;Tourlänge;Schwierigkeit;Höhenmeter;Charakter;Abfahrten;Kurzbeschreibung;Beschreibung;ZusatzInfo;Tourleiter. Sie können die Datei mit Excel öffnen, oder in Word mit der Serienbrief-Funktion einlesen und formatieren. Ein Problem dabei ist, daß z.B. die Felder Abfahrten, Zusatzinfo und Tourleiter mehrere Werte haben können. Diese werden in der CSV-Datei durch newline getrennt.

# Cache

Die Daten, die das Programm vom Tourenportal bekommt, werden jedesmal in eine Datei in das Verzeichnis C:\temp\tpjson geschrieben. In diesem Verzeichnis sehen Sie, nachdem Sie mit Start experimentiert haben, einige bis viele Dateien:

##### Ereignisdaten Deren Namen fangen mit 6 hexadezimalen Zeichen (also 0-9 und a-f) an, dann folgt der Titel, z.B. 00f6a0\_Starnberger\_See\_Umrundung.json.

* Suchergebnisse  
  Deren Namen fangen mit search an, dann folgt die Gliederungsnummer, ggfs. ein I, falls Untergliederungen inkludiert waren, dann Start- und Ende-Datum, z.B. search-152085\_I\_2018-01-01-2019-12-31.json
* units.json  
  Diese Datei enthält die Daten der ADFC-Gliederungen. Daraus wird der Inhalt des Landesverband-Menüs (s.o.) erzeugt.
* Benutzerdaten  
  Deren Namen fangen mit user\_ an. Sie enthalten die Daten eines Benutzers, i.a. eines Tourenleiters. Das Programm interessiert sich nur für den Namen und die Telefonnummer.

Alle diese Dateien können gelöscht werden. Sie werden dann bei Bedarf neu vom Tourenportal geholt. Sie werden auch dann neu geholt, wenn das Häkchen bei *Aktuelle…geholt* gesetzt ist. Das kostet natürlich Zeit und Datenvolumen, und beansprucht den Tourenportal-Server. Am besten ist es, wenn sie gezielt nur die Dateien löschen, von denen Sie wissen, daß sie im Tourenportal geändert wurden. Wenn z.B. die Tour „Starnberger See Umrundung“ geändert wurde, gehen Sie mit dem Datei-Explorer in das Verzeichnis C:\temp\tpjson, suchen z.B. nach Starnberger oder Umrundung, und löschen die Datei. Löschen Sie ein paar zuviel, macht das auch nichts, sie werden dann eben neu geholt.

# Die Word-Vorlagendatei

Einige Vorlagendateien mit den Namen template…docx sehen Sie im Verzeichnis  
<https://github.com/michaelu123/adfc2/tree/master/venv/src/doc-templates> .  
Runterladen durch Anklicken der Datei, dann rechts mitte „Download“ anklicken. Die Dateien bestehen aus mehreren Teilen:

* Ein Kommentar am Anfang.
* Der Parameter *Linktyp*
* Der Parameter *Ausgabedatei*
* Der Parameter *Selektion* mit den Unterparametern für *Gliederungen*, *MitUntergliederungen*, *Beginn* und *Ende*.
* Der Parameter *Terminselektion* mit verschiedenen Unterselektionen
* Der Parameter *Tourselektion* mit verschiedenen Unterselektionen
* Eine Zeile mit mindestens 5 – (minus)-Zeichen
* Danach beliebiger Text, nach Bedarf formatiert
* Darin enthalten templates, beginnend mit /*template* und endend mit /*endtemplate*
* Evtl. noch Parameter im Format *${parameter}.fmt(format)*

Was Sie nicht sehen, ist, daß die Vorlagendatei noch Informationen zu Dokumenten-Styles und – Fonts enthält. Wenn Sie mit einem leeren Dokument anfangen würden, fehlen diese Informationen. Wenn dann in der Tourbeschreibung z.B. ein „header1“ definiert ist, und der Style „Header 1“ nicht vorhanden ist, gibt es eine Fehlermeldung. Sie sollten eine neue Vorlagendatei deshalb immer durch Ändern einer vorhandenen erstellen. Sie können fehlende Styles aber auch erzwingen, indem Sie in die Vorlagendatei irgendeinen Text mit diesem Style reinschreiben, speichern, und dann den Text wieder löschen. Der Style bleibt dabei erhalten.

Wozu das alles?

Wir können in der Vorlagendatei erst einmal Parameter spezifizieren, die von der Programmoberfläche übernommen werden. Die Parameter Linktyp, Gliederungen, MitUntergliederungen, Beginn, Ende werden Sie in der Oberfläche bei *Links ins:*, dem Gliederungen-Feld, dem Feld *Untergliederungen einbeziehen*, *Start Datum* und *Ende Datum* wiederfinden. Wenn sie die Ausgabedatei **nicht** spezifizieren, wird der Name der Ausgabedatei aus diesen Parametern gebildet.

Jedes template wird durch die Daten der Ereignisse aufgefüllt, die der zum template gehörenden Selektion entsprechen. Wie das Ereignis dargestellt wird, entscheidet der Text zwischen /template und /endtemplate.

Wir wollen in der Ausgabedatei vielleicht erst die Tourenrad-Touren, dann die Rennrad-Touren, dann die Mountainbike-Touren, und dann die Stammtisch-Termine haben. Dafür brauchen wir jeweils eine Selektion, die diese Ereignisse gezielt herausfiltert. Vielleicht wollen wir auch nur ganz spezielle Ereignisse an einer bestimmten Stelle sehen oder nicht sehen. Auch das läßt sich erreichen. Und dann unterscheidet sich vielleicht das Format, in dem wir Touren ausgeben. Bei Tagestouren interessiert uns z.B. der Tag, die Beginn- und vielleicht die Ende-Zeit. Bei Mehrtagestouren interessiert uns der erste und der letzte Tag, weniger die Uhrzeit.

Oder nehmen Sie als Beispiel ein in ihrem Zeitraum öfter vorkommendes Ereignis, das über eine etwas längliche Beschreibung verfügt. Um nicht zig-mal den gleichen Text in Ihrem Dokument zu haben, tun Sie das erste Ereignis mit Beschreibung in ein template, und alle anderen in ein anderes template ohne Beschreibung.

## Parameter in der Programm-Oberfläche

Nach dem Einlesen einer Vorlagendatei entsprechen die Parameter in der Oberfläche denen in der Vorlage, sofern dort gesetzt. Vor Anklicken von Start lassen sich diese Parameter aber in der Oberfläche noch ändern und beeinflussen dann das Ergebnis. Das betrifft den Link-Typ, die Gliederungen, das „Untergliederungen“-Häkchen, und das Start- und Ende-Datum. So können Sie z.B. kurz hintereinander zwei Dateien erzeugen mit gleichem Inhalt, aber die Links mal ins Frontend, mal ins Backend.

## Paragraphen und Newlines

In der Vorlagendatei sind Paragraphen und Newlines oft bedeutsam. Öffnen Sie die Datei template.docx und aktivieren Sie die Absatzmarken-Sicht, indem Sie im Reiter *Start* auf ¶ klicken. Sie sehen Zeilen, die mit ¶ oder mit einem Pfeil nach unten und links enden. Erstere werden mit Enter erzeugt, und stehen zwischen Paragraphen, letzere werden mit Shift-Enter erzeugt, und bedeuten einfach „neue Zeile“. Eine Zeile mit /template muß immer am Anfang, eine Zeile mit /endtemplate am Ende eines Paragraphen stehen. Der Parameter ${beschreibung} (Erklärung später, produziert die Event-Beschreibung) muß in einem eigenen Paragraphen stehen. Der Zeilenabstand im Absatzformat und andere Einstellungen werden auch in die Ausgabe übernommen.

## Symbolzeichen

Diese werden nur dann in die Ausgabe übernommen, wenn sie über das Keyboard eingegeben wurden, nicht aber, wenn sie über ein Symbol-Eingabe-Fenster mit dem Button „Einfügen“ erzeugt wurden.

## Der Kommentar

Am Anfang kann ein Kommentar stehen. Paragraphen, die mit dem Wort Kommentar beginnen, sowie Listen und eingerückte Paragraphen werden ignoriert. Der Text bis zur Zeile mit mindestens 5 Minus-Zeichen wird im resultierenden Dokument gelöscht.

## Linktyp

Analog zur Programm-Oberfläche. Voreingestellt ist *Frontend*. Wenn Sie keine Links wollen, geben Sie *Linktyp: Keine* an. Der Zwischenraum nach dem : ist wichtig.

## Ausgabedatei

Hier können Sie eine Ausgabedatei angeben. Sie sollte sinnvollerweise auf .docx enden. Geben Sie keine Datei an, wird die Ausgabedatei in das gleiche Verzeichnis geschrieben, in dem die Vorlagendatei steht. Der Name der Ausgabedatei ist dann ADFC\_gliederung\_[I\_]start-end\_[f|b].docx. I\_ wenn Untergliederungen einbezogen sind, und f oder b je nach Links ins Front- oder Backend.

Der Zwischenraum nach dem : ist wichtig.

## Selektion

Die Unterparameter müssen mit Zwischenraum oder Tabulator eingerückt sein. Der Zwischenraum nach dem : ist wichtig.

Für alle Selektionen gilt: bei den Namen der Parameter wie auch bei den Werten sind Groß- und Kleinschreibung egal (case-insensitive).

## Terminselektion, Tourenselektion

Wie bei Selektion. Eine neue Selektion beginnt mit *Name:* . Der Name wird später in den templates benutzt, um die Selektion zu bestimmen. Der Wert der Parameter kann ein einzelnes Wort sein, oder eine durch Komma getrennte Liste. Bei der Selektion werden die einzelnen Parameter durch UND verknüpft, die Werte in den Listen durch ODER. Eine Selektion  
 Titelenthält: A,B  
 Titelenthältnicht: C,D  
selektiert ein Ereignis also dann, wenn der Titel (A ODER B) UND NICHT (C ODER D) enthält.

Die möglichen Parameter sind

* TitelEnthält, TitelEnthältNicht:  
  Der Ereignis-Titel enthält das Wort, oder er enthält es nicht.
* Terminnr, NichtTerminNr, Tournr, NichtTourNr:  
  Die Tour- bzw. Terminnr (zu sehen z.B. im Ausgabeformat Text oder München) ist oder ist nicht eine der angegebenen Nummern.
* RadTyp:  
  Für Touren: kann die Werte Tourenrad, Rennrad, Mountainbike haben.
* Kategorie:

Für Touren: kann die Werte Tagestour, Halbtagestour, Feierabendtour, Mehrtagestour, Rad-Reise haben.

* MerkmalEnthält, MerkmalEnthältNicht:

Für Termine: kann z.B. die Werte Infoladen, Stammtisch, Öffentliche Arbeitsgruppe, Aktiventreff, Fahrradmesse annehmen, oder Teile davon, z.B.  
 MerkmalEnthält: messe,arbeit

Die Werte entsprechen denen, die man im Backend, Termin/Merkmale definieren kann.

Enthält die Selektion nur einen Namen, also keine weiteren Parameter, selektiert sie alle Touren bzw. Termine (nur durch Gliederung, Start, Ende eingeschränkt).

#### Tour/Terminnummer

Die Ereignisse werden innerhalb eines Jahres durchnumeriert. Tourenradtouren ab 100, Rennradtouren ab 300, Mountainbike-Touren ab 400, Mehrtagestouren ab 600, Termine ab 700. Sie können mit der Nummer selektieren, oder sie mit ${nummer} ins Ergebnis schreiben. Ändert sich das Start-Datum, ändern sich auch die Nummern.

## Das eigentliche Dokument

Nach der Zeile mit mindestens 5 minus-Zeichen beginnt das eigentliche Dokument. Alles, was hier steht, landet genauso im Ausgabedokument. Man schreibt es wie jedes andere Word-Dokument. Es gibt nur zwei Ausnahmen: Man kann in den Text den Parameter ${heute} schreiben, der durch das aktuelle Datum in einem bestimmten Format ersetzt wird, und es gibt Abschnitte zwischen /template und /endtemplate. Bis auf den Parameter ${beschreibung} gilt für alle Parameter: der Text wird ersetzt durch das, wofür der Parameter steht, aber der Font, die Zeichengröße, die Textfarbe, die Ausrichtung usw. bleiben erhalten. In template.docx steht ${titel} z.B. in einem Gelb/Orange, das auch im ADFC-Logo verwendet wird. Also steht später der Tour-Titel (mitsamt dem Link) in dem gleichen Gelb/Orange.

## Datums-Parameter mit Format

Im Dokument können Sie außerhalb der templates den Parameter ${heute}.fmt(…) in das Dokument hineinschreiben. Innerhalb der templates können Sie die Parameter ${start}.fmt(…) und ${end}.fmt(…) verwenden. ${heute} steht für das aktuelle Datum, ${start} und ${end} für Beginn und Ende des Ereignisses. Die Kürzel innerhalb der fmt()-Klammern bestimmen, in welchem Format das Datum ausgegeben wird. Eine exakte Beschreibung des Formats sehen Sie auf den Seiten <https://docs.python.org/3/library/datetime.html> oder <http://www.gnu.org/software/libc/manual/html_node/Formatting-Calendar-Time.html> . Für uns von Bedeutung sind:

%H: Stunde von 00 bis 23  
%M: Minute von 00 bis 59  
%d: Tag im Monat von 01 bis 31  
%B: Name des Monats ausgeschrieben: Januar, Februar, …  
%b: Name des Monats abgekürzt: Jan, Feb, …  
%A: Name des Tags ausgeschrieben: Montag, Dienstag, …  
%a: Name des Tags abgekürzt: Mo, Di, …  
%Y: Jahr vierstellig: 2018  
%y: Jahr zweistellig: 18

Falls Sie nichtdeutsche Namen sehen, müssen Sie ihre Ländereinstellung, das *Locale*, ändern, und das Programm noch einmal neu starten.

Z.B. produziert ${heute}.fmt(%A, den %d. %B %Y, um %H:%M Uhr) heute, jetzt   
 Montag, den 10. Dezember 2018, um 20:29 Uhr

## Die templates

Ein template beginnt mit der Zeile

/template /termin (oder /tour) /selektion=Name

Jedes template erzeugt Ausgabe für jedes Ereignis, das der dazugehörigen Selektion entspricht. Der Name hinter /selection= muß einem der Namen der Termin- oder Tourselektionen am Anfang der Vorlagendatei entsprechen. Das template endet mit /endtemplate. /template muß am Beginn, /endtemplate am Ende eines Paragraphen stehen. Innerhalb des templates stehen Platzhalter in der Schreibweise ${parameter}. Der nächste Abschnitt behandelt diese Platzhalter.

# Template-Parameter

Der Zeichensatz, die Zeichensatzgröße, die Farbe des Parameters usw. werden in der Regel in der Ausgabe beibehalten.

## ${heute}.fmt()

Das aktuelle Datum mit Uhrzeit, Tour/Termin-unspezifisch.

## ${start}.fmt(), ${end}.fmt()

Beginn und Ende des Ereignisses

## ${nummer}

Die Konkatenation von T, M oder R (für Tourenrad/Mountainbike/Rennrad) gefolgt von der Ereignisnummer (s.o.) gefolgt von G, H, oder F für Ganztags/Halbtags/Feierabend-Tour.

## ${titel}

Der Titel des Ereignisses.

## ${beschreibung}

Die (lange) Beschreibung des Ereignisses. Dieser Parameter muß in einem eigenen Paragraphen stehen. Der formatierte Text der Beschreibung wird übernommen / sollte übernommen werden.

## ${kurz}

Die (kurze) Beschreibung des Ereignisses. Dieser Parameter muß in einem eigenen Paragraphen stehen.

## ${tourleiter}

Produziert bei fehlendem Tourleiter nichts, sonst die Ausgabe **Tourleiter:** Vorname Nachname (Telefonnummer), sofern die Telefonnummer bekannt ist. Bei zwei Tourleitern folgen ein Komma und der zweite Name. Mehr als zwei Tourleiter erlaubt das Tourenportal derzeit nicht.

## ${tourleiterm}

Produziert die Ausgabe Vorname Nachname (Telefonnummer), sofern die Telefonnummer bekannt ist. Bei zwei Tourleitern folgen ein Komma und der zweite Name.

## ${betreuer}

Wie tourleiter, nur **Betreuer** anstatt **Tourleiter**.

## ${city}, ${street}

Der Ort, der im Backend bei Hauptinformationen/Startpunkt als „Straße/Ort“ eingetragen ist, und bei Radtour/Stationen an erster Stelle steht.

## ${kategorie}

Bei Touren das erste Merkmal, daß im Backend unter Radtour/Merkmale/Typen (nach Dauer und Tageslage) aufgeführt ist, also Tagestour, Mehrtagestour, Halbtagestour, Feierabendtour, Rad-Reise.  
Bei Terminen das erste Merkmal, daß im Backend unter Termin/Merkmale gefunden wird, z.B. Infoladen oder Stammtisch.

## ${schwierigkeit}

Das Portal gibt die Schwierigkeit als eine Zahl x.y, z.B. 1.4 oder 3.7 zurück. Im Frontend wird aber nur sehr einfach, einfach, mittel, schwer, sehr schwer angezeigt. Das Programm addiert 0.5 auf die Schwierigkeit, nimmt dann den Ganzzahl-Anteil und gibt dann die entsprechende Stufe aus. Z.B. 1.4 + 0.5 = 1.9, Ganzzahlanteil 1, also sehr einfach. 3.7 + 0.5 = 4.2, Ganzzahlanteil 4, also schwer. Genauso macht es anscheinend das Frontend. Anders ausgedrückt: sehr einfach: 0.5-1.4, einfach: 1.5-2.4, usw.

## ${schwierigkeitm}

Die Schwierigkeit als eine Folge von 5 schwarzen oder weißen Quadraten. 5x weiß=sehr leicht, 5\* schwarz = sehr schwer.

## ${tourlänge}

Gibt die Tourlänge als „nn km“ aus.

## ${höhenmeter}

Gibt die Höhenmeter aus (ohne m danach).

## ${character}

Gibt die Wegbeschaffenheit (im Frontend „Oberflächenqualität“ genannt) aus.

## ${abfahrten}

Im Backend wird die erste Station in Veranstaltung/Hauptinformationen mit Beginn, Ende, Startpunkt definiert, letzterer durch Straße, Ort. Unter Radtour/Stationen kann man eine oder mehrere zusätzliche Stationen mit einem Typ, einem Namen, einem Ort, einem Datum, evtl. einer Zeit, und evtl. einem Ende eingeben. Das Programm macht aus all dem mit gewisser Willkürlichkeit eine Ausgabe **Ort** oder **Orte** gefolgt von den Stationen. In künftigen Versionen läßt sich die Ausgabe vielleicht anpassen…

## ${abfahrtenm}

Eine erste solche Anpassung für ein spezielles „Münchner“ template.

## ${zusatzinfo}

Im Backend kann man unter Radtour/Merkmale Angaben zu „Besondere Zielgruppe“, „Besondere Charakteristik/Thema“, und „Weitere Eigenschaften“ machen. Diese Informationen werden hier ausgegeben.

# Debugging

Das Programm schreibt eine Datei adfc.log in Ihr aktuelles Verzeichnis. Wenn alles glatt läuft, ist die Datei leer, sonst enthält sie Fehlermeldungen. Wenn Sie vor Aufruf des Programms aus einer command line den environment-Parameter DEBUG=1 setzen, werden zahlreiche weitere Meldungen ausgegeben.

# Rest-Calls

Das Programm liest zuerst die Gliederungs-Info:  
 <https://api-touren-termine.adfc.de/api/units>  
Dann erfolgt die Suche nach Ereignissen:  
 [https://api-touren-termine.adfc.de/api/eventItems/search?unitKey=152085&beginning=2019-01-01&end=2019-12-31&includeSub=true](https://api-touren-termine.adfc.de/api/eventItems/search?unitKey=152085&beginning=2018-07-01&end=2018-07-31&includeSub=true)  
Für jedes Ereignis wird ein Request mit der eventItemId ausgeführt:  
 <https://api-touren-termine.adfc.de/api/eventItems/955229bd-79ef-4b7f-bab7-f909118f92c5>  
Falls in dem Ereignis eine cOrganizingUserId oder cSecondOrganizingUserId (vulgo Tourleiter/Betreuer) vorkommen, wird dieser mit  
 <https://api-touren-termine.adfc.de/api/users/ca503a39-3695-49eb-8bc1-0600694adbc4>  
gesucht.