Bingo – Export GTFS

This document explains what we need to export DIVA data to a valid GTFS format (non-realtime data only for a first milestone) to be shown on Google maps, or other tools that support GTFS. It is a minimal setup for now, separated into six chapters for each required CSV file that must exist, to fulfill the GTFS specification[1].

NB: We put either GTFS: or DIVA: as a source prefix to identify the primary source. Secondly, we add the corresponding file name or table name. The last field contains the attribute name, i.e., the CSV header column, or database table attribute. For instance, DIVA:REC_ORT.X10:ORT_NR means that we look at the source DIVA, the file REC_ORT.X10, and the attribute ORT_NR.

GTFS:stops.txt

Source data:

- DIVA:REC_ORT.X10

Minimal requirements:

- **GTFS:stops.txt:stop_id**
  The **stop_id** field contains an ID that uniquely identifies a stop, station, or station entrance. Multiple routes may use the same stop. The **stop_id** is used by systems as an internal identifier of this record (e.g., primary key in database), and therefore the **stop_id** must be dataset unique.
  - → DIVA:ORT_NR is the only possible field for that, DIVA:ORT_NAME, DIVA:ORT_REF_ORT_KUERZEL, or DIVA:ORT_REF_ORT_NAME are not uniquely identifiable
  - → Stops are not specified for a driving direction, hence the latter attributes mentioned in (1.a) are not applicable
- **GTFS:stops.txt:stop_name**
  The **stop_name** field contains the name of a stop, station, or station entrance. Please use a name that people will understand in the local and tourist vernacular.
  - → DIVA:REC_ORT.X10:ORT_NAME is possible, but has some issues:
    - Mixture of languages
    - Abbreviations due to a too short char field in the original source (40 characters)
    - Sometimes second language missing due to the too short field
    - Non standardized separation between languages (sometimes “-”, sometimes with “-” and spaces)
    - Misinterpretation possible due to too general descriptions (ex, “Schulen – Scuole”, without city/town names)
    - Uppercase/lowercase for names is random
    - Missing translations (names like “- Grumser” with no text before “-”)

1 https://developers.google.com/transit/gtfs/reference

→ We propose to model translations with GTFS:translations.txt and not within a single description field. However, that must be investigated further. See GTFS extensions[2] for details.

➢ **GTFS:stops.txt:stop_lat and GTFS:stops.txt:stop_lon**
The **stop_lat** field contains the latitude of a stop, station, or station entrance. The field value must be a valid WGS 84 latitude. The **stop_lon** field contains the longitude of a stop, station, or station entrance. The field value must be a valid WGS 84 longitude value from -180 to 180.

→ DIVA:REC_ORT.X10:ORT_POS_LAENGE (=longitude) and ORT_POS_BREITE (=latitude) which are of the form dd°mm'ss.sss, but written as a 9-digit long integer.

GTFS:agency.txt

Source data:

- DIVA:MENGE_UNTERNEHMER.X10
- Missing: A list of all companies and their homepages

Minimal requirements:

The minimal requirements depend on whether we want to use a single agency exposed to GTFS, or give additional details to the customers explaining which company takes responsibility for the actual ride. As a first prototype also a single company is possible, that means agency_id would not be necessary, and the rest is a single hard-coded CSV file. If we want to give all information the following setup is necessary:

➢ **GTFS:agency.txt:agency_id**
The **agency_id** field is an ID that uniquely identifies a transit agency. A transit feed may represent data from more than one agency. The agency_id is dataset unique. This field is optional for transit feeds that only contain data for a single agency.

→ DIVA:MENGE_UNTERNEHMER.X10:UNTERNEHMER_NR

➢ **GTFS:agency.txt:agency_name**
The **agency_name** field contains the full name of the transit agency. Google Maps will display this name.

→ DIVA:MENGE_UNTERNEHMER.X10:NAME with some issues:

- Uppercase/lowercase inconsistent
- Abbreviations inconsistent (some with company description, some without)
- Possible translations possible (Not sure, this needs some additional clarification)

→ In addition, add a description to the "default" agencies, and define when to use those.

➢ **GTFS:agency.txt:agency_url**
The **agency_url** field contains the URL of the transit agency. The value must be a fully qualified URL that includes http:// or https://, and any special characters in the URL must be correctly escaped. See URI-Recommendations[3] for a description of how to create fully qualified URL values.

→ Missing in DIVA, no source found for that information

2 https://developers.google.com/transit/gtfs/reference/gtfs-extensions#translations

3 http://www.w3.org/Addressing/URL/4_URI_Recommentations.html

- **GTFS:agency.txt:agency_timezone**
  The **agency_timezone** field contains the timezone where the transit agency is located. Timezone names never contain the space character but may contain an underscore. Please refer to http://en.wikipedia.org/wiki/List_of_tz_zones for a list of valid values. If multiple agencies are specified in the feed, each must have the same agency_timezone.
  → This is the same for all companies so far

GTFS:routes.txt

Source data:

- DIVA:REC_LID.X10

Minimal requirements:

- **GTFS:routes.txt:route_id**
  The **route_id** field contains an ID that uniquely identifies a route. The route_id is dataset unique.
  → DIVA:REC_LID.X10:LI_NR
  → Alternative: take a random unique identifier.
- **GTFS:routes.txt:agency_id**
  The **agency_id** field defines an agency for the specified route. This value is referenced from the agency.txt file. Use this field when you are providing data for routes from more than one agency.
  → NB: This field is needed, if we choose a multi-agency approach (see GTFS:agency.txt)
  → DIVA:REC_LID.X10:FREMDUNTERNEHMER_NR
    - if empty, take the default agency ID
    - otherwise, choose the contained ID
- **GTFS:routes.txt:route_short_name**
  The **route_short_name** contains the short name of a route. This will often be a short, abstract identifier like "32", "100X", or "Green" that riders use to identify a route, but which doesn't give any indication of what places the route serves. At least one of *route_short_name* or *route_long_name* must be specified, or potentially both if appropriate. If the route does not have a short name, please specify a *route_long_name* and use an empty string as the value for this field.
  → DIVA:REC_LID.X10:LI_KUERZEL (we have short names for all routes)
- **GTFS:routes.txt:route_long_name**
  The **route_long_name** contains the full name of a route. This name is generally more descriptive than the *route_short_name* and will often include the route's destination or stop. At least one of *route_short_name* or *route_long_name* must be specified, or potentially both if appropriate. If the route does not have a long name, please specify a *route_short_nam*e and use an empty string as the value for this field.
  → Idea: We could combine the start- and end-stop’s name to form a long name
  → Another solution is to take DIVA:REC_LID.X10:LI_KUERZEL as long name, and DIVA:REC_LID.X10:LIDNAME as short name
- **GTFS:routes.txt:route_type**
  The **route_type** field describes the type of transportation used on a route. Valid values for this field

are:
0 - Tram, Streetcar, Light rail. Any light rail or street level system within a metropolitan area.
1 - Subway, Metro. Any underground rail system within a metropolitan area.
2 - Rail. Used for intercity or long-distance travel.
3 - Bus. Used for short- and long-distance bus routes.
4 - Ferry. Used for short- and long-distance boat service.
5 - Cable car. Used for street-level cable cars where the cable runs beneath the car.
6 - Gondola, Suspended cable car. Typically used for aerial cable cars where the car is suspended from the cable.
7 - Funicular. Any rail system designed for steep inclines.

→ For now, we only have route_type = 3

GTFS:trips.txt

Please note, that a trip is unidirectional. This means, that you must specify at least 2 trips for each route. For instance, route 11 goes from Postal to Merano, that means we must specify one trip for Postal to Merano, and one for Merano to Postal.

Source data:

- GTFS:calendar.txt
- GTFS:routes.txt

Minimal requirements:

- **GTFS:trips.txt:trip_id**
  The **trip_id** field contains an ID that identifies a trip. The **trip_id** is dataset unique.
  → We did not find any unique trip identifier in DIVA, hence we suggest to simply take a random unique identifier.
- **GTFS:trips.txt:route_id**
  The **route_id** field contains an ID that uniquely identifies a route. This value is referenced from the routes.txt file.
  → From GTFS:routes.txt:route_id
- **GTFS:trips.txt:service_id**
  The **service_id** contains an ID that uniquely identifies a set of dates when service is available for one or more routes. This value is referenced from the calendar.txt or calendar_dates.txt file.
  → From GTFS:calendar.txt:service_id

GTFS:stop_times.txt

Source data:

- GTFS:trips.txt
- GTFS:stops.txt

Minimal requirements:

- **GTFS:stop_times.txt:trip_id**
  The trip_id field contains an ID that identifies a trip. This value is referenced from the trips.txt file.

  → Take it from GTFS:trips.txt:trip_id

- **GTFS:stop_times.txt:arrival_time**
  The arrival_time specifies the arrival time at a specific stop for a specific trip on a route. The time is measured from "noon minus 12h" (effectively midnight, except for days on which daylight savings time changes occur) at the beginning of the service day. For times occurring after midnight on the service day, enter the time as a value greater than 24:00:00 in HH:MM:SS local time for the day on which the trip schedule begins. If you don't have separate times for arrival and departure at a stop, enter the same value for arrival_time and departure_time.

  Scheduled stops where the vehicle strictly adheres to the specified arrival and departure times are timepoints. For example, if a transit vehicle arrives at a stop before the scheduled departure time, it will hold until the departure time. If this stop is not a timepoint, use either an empty string value for the arrival_time field or provide an interpolated time. Further, indicate that interpolated times are provided via the timepoint field with a value of zero. If interpolated times are indicated with timepoint=0, then time points must be indicated with a value of 1 for the timepoint field. Provide arrival times for all stops that are time points.

  An arrival time must be specified for the first and the last stop in a trip. Times must be eight digits in HH:MM:SS format (H:MM:SS is also accepted, if the hour begins with 0). Do not pad times with spaces. The following columns list stop times for a trip and the proper way to express those times in the arrival_time field:

Time	arrival_time value
08:10:00 A.M.	08:10:00 or 8:10:00
01:05:00 P.M.	13:05:00
07:40:00 P.M.	19:40:00
01:55:00 A.M.	25:55:00

  Note: Trips that span multiple dates will have stop times greater than 24:00:00. For example, if a trip begins at 10:30:00 p.m. and ends at 2:15:00 a.m. on the following day, the stop times would be 22:30:00 and 26:15:00. Entering those stop times as 22:30:00 and 02:15:00 would not produce the desired results.

  → I could not find any data about schedules within the export data

- **GTFS:stop_times.txt:departure_time**
  The **departure_time** specifies the departure time from a specific stop for a specific trip on a route. The time is measured from "noon minus 12h" (effectively midnight, except for days on which daylight savings time changes occur) at the beginning of the service day. For times occurring after midnight on the service day, enter the time as a value greater than 24:00:00 in HH:MM:SS local time for the day on which the trip schedule begins. If you don't have separate times for arrival and departure at a stop, enter the same value for **arrival_time** and **departure_time**.

  Scheduled stops where the vehicle strictly adheres to the specified arrival and departure times are timepoints. For example, if a transit vehicle arrives at a stop before the scheduled departure time, it

will hold until the departure time. If this stop is not a timepoint, use either an empty string value for the **departure_time** field or provide an interpolated time (further, indicate that interpolated times are provided via the **timepoint** field with a value of zero). If interpolated times are indicated with **timepoint**=0, then time points must be indicated with a value of 1 for the **timepoint** field. Provide departure times for all stops that are time points.

A departure time must be specified for the first and the last stop in a trip even if the vehicle does not allow boarding at the last stop. Times must be eight digits in HH:MM:SS format (H:MM:SS is also accepted, if the hour begins with 0). Do not pad times with spaces. The following columns list stop times for a trip and the proper way to express those times in the **departure_time** field:

Time	departure_time value
08:10:00 A.M.	08:10:00 or 8:10:00
01:05:00 P.M.	13:05:00
07:40:00 P.M.	19:40:00
01:55:00 A.M.	25:55:00

Note: Trips that span multiple dates will have stop times greater than 24:00:00. For example, if a trip begins at 10:30:00 p.m. and ends at 2:15:00 a.m. on the following day, the stop times would be 22:30:00 and 26:15:00. Entering those stop times as 22:30:00 and 02:15:00 would not produce the desired results.

→ I could not find any data about schedules within the export data

- **GTFS:stop_times.txt:stop_id**
  The **stop_id** field contains an ID that uniquely identifies a stop. Multiple routes may use the same stop. The stop_id is referenced from the stops.txt file. If location_type is used in stops.txt, all stops referenced in stop_times.txt must have location_type of 0. Where possible, stop_id values should remain consistent between feed updates. In other words, stop A with stop_id 1 should have stop_id 1 in all subsequent data updates. If a stop is not a time point, enter blank values for arrival_time and departure_time.

  → From GTFS:stops.txt:stop_id

- **GTFS:stop_times.txt:stop_sequence**
  The **stop_sequence** field identifies the order of the stops for a particular trip. The values for **stop_sequence** must be non-negative integers, and they must increase along the trip. For example, the first stop on the trip could have a **stop_sequence** of 1, the second stop on the trip could have a **stop_sequence** of 23, the third stop could have a **stop_sequence** of 40, and so on.

  → Enumeration of stops in increasing order for a single ride

GTFS:calendar.txt

Calendar entries should be discussed with Ivan Bernardi to understand how to model general schedules and exceptions. For the latter, we suggest to use GTFS:calendar_dates.txt (see next chapter).

Source data:

- DIVA:???

Minimal requirements:

- **GTFS:calendar.txt:service_id**
  The **service_id** contains an ID that uniquely identifies a set of dates when service is available for one or more routes. Each service_id value can appear at most once in a calendar.txt file. This value is dataset unique. It is referenced by the trips.txt file.

  → ?

- **GTFS:calendar.txt:{monday,tuesday,…,sunday}**
  The monday field contains a binary value that indicates whether the service is valid for all Mondays. A value of 1 indicates that service is available for all Mondays in the date range. (The date range is specified using the start_date and end_date fields.)
  A value of 0 indicates that service is not available on Mondays in the date range.
  Note: You may list exceptions for particular dates, such as holidays, in the calendar_dates.txt file.

  → ?

- **GTFS:calendar.txt:start_date**
  The **start_date** field contains the start date for the service. The start_date field's value should be in YYYYMMDD format.

  → ?

- **GTFS:calendar.txt:end_date**
  The **end_date** field contains the end date for the service. This date is included in the service interval. The end_date field's value should be in YYYYMMDD format.

  → ?

GTFS:calendar_dates.txt

This files defines exceptions from the regular scheduling. That is, holidays, non-school days, etc. We need more information on how to model holidays and other exceptions.

Source data:

- GTFS:calendar.txt
- DIVA:???

Minimal requirements:

- **GTFS:calendar_dates.txt:service_id**
  The **service_id** contains an ID that uniquely identifies a set of dates when a service exception is available for one or more routes. Each (service_id, date) pair can only appear once in calendar_dates.txt. If the a service_id value appears in both the calendar.txt and calendar_dates.txt files, the information in calendar_dates.txt modifies the service information specified in calendar.txt. This field is referenced by the trips.txt file.

  → Take it from GTFS:calendar.txt:service_id

- **GTFS:calendar_dates.txt:date**
  The **date** field specifies a particular date when service availability is different than the norm. You can use the **exception_type** field to indicate whether service is available on the specified date. The **date** field's value should be in YYYYMMDD format.

→ ?

- **GTFS:calendar_dates.txt:exception_type**
  The exception_type indicates whether service is available on the date specified in the date field.
  A value of 1 indicates that service has been added for the specified date.
  A value of 2 indicates that service has been removed for the specified date.
  For example, suppose a route has one set of trips available on holidays and another set of trips available on all other days. You could have one service_id that corresponds to the regular service schedule and another service_id that corresponds to the holiday schedule. For a particular holiday, you would use the calendar_dates.txt file to add the holiday to the holiday service_id and to remove the holiday from the regular service_id schedule.

  → ?