Datenschnittstelle MFPL-API

Spezifizierung der öffentlichen Onlinedienste

Inhaltsverzeichnis

Versionshinweise

Version	Datum	Autor	Beschreibung
1.0	01.08.18	Knopf, S.	Initialversion
1.1.0	25.09.18	Knopf, S.	- Unterstützung von `initial_input` - Wegfall von `signature` im *Container*-Objekt - Einbringung von `instance` im *Container*-Objekt - Anpassung von Beschreibungen
1.1.1	30.09.18	Knopf, S.	- Unterstützung der GET-Anfrage `query` - Anpassung der `include_*` - Parameter - Einbringung von mögl. Nutzungsszenarien
1.2.0	01.11.18	Knopf, S.	- Einbindung des StopEventService - Änderung in Frequency-Objekt: * `distance` -> `headway` * Einbindung von `exact_times` - Anpassung der Fehlercodierungen
1.2.1	15.11.18	Knopf, S.	- `StopEventService` -> `StopInformationService` - `include_stop_times` -> `include_full_stop_times`
1.2.2	18.11.18	Knopf, S.	- `include_stops` im StopInformationService - `ONLINE_SERVICE_REQUEST` -> `REQUEST` - `route_types` aus dem Filter des TripInformationService entfernt
1.3.0	20.11.18	Knopf, S.	- Unterstützung für Echtzeitdaten I (Servicemeldungen)
1.3.1/2/3	05.12.18	Knopf, S.	- Zeiteingabe im `filter`-Feld `time` als Ergebnisfilter - Beschränkung der Ergebnismenge durch das `options`-Feld `limit`
1.3.4	31.01.19	Knopf, S.	- Einbindung der Felder `wheelchair_accessible` und `bikes_allowed` im Feld `filter`
1.3.5	15.02.19	Knopf, S.	- Wegfall des Geocoding-Mode im `LocationInformationService`
1.4.0	28.02.19	Knopf, S.	- Einbindung des `CalendarService`
1.4.1	27.03.19	Knopf, S.	- `route_id` und `stop_id` als Request-Parameter im `CalendarService` nachträglich eingefügt

Einleitung

Die MFPL-Datenschnittstelle stellt eine REST-API zur einheitlichen Verteilung von Daten aus dem Hintergrundsystem bereit. Der Datenaustausch erfolgt grundsätzlich auf Basis von http(s)-Anfragen.

Die verarbeiteten Daten basieren auf dem Standard GTFS zur Beschreibung von Fahrplandaten im öffentlichen Personenverkehr.

Zur besseren Übersicht ist die MFPL-API in insgesamt vier öffentliche Onlinedienste

- LocationInformationService
- TripInformationService
- StopInformationService
- CalendarService

unterteilt.

Diese Onlinedienste dienen der allgemeinen Information sowie dem Datenaustausch zwischen dem Hintergrundsystem und anderen Auskunftssystemen. Sie sind in Aufbau und Funktion grundlegend an Onlinediensten orientiert, wie sie auch in EFA-Systemen zum Einsatz kommen.

Alle Onlinedienste können im Produktivsystem über die Adresse

`https://api.museumsfahrplan.de/w/{OnlineService}`

bzw. im Testsystem über die Adresse

`https://gtfs.svtest02.info/w/{OnlineService}`

angesprochen werden. Der Teil `{OnlineService}` ist dabei durch den Namen des gewünschten Onlinedienstes zu ersetzen. Die Anfrage muss in Form eines JSON codierten *Container*-Objekts in das Feld `REQUEST` eingetragen werden. Als Anfragemethoden sind `GET` und `POST` zulässig.

Die max. Länge einer `GET`-Anfrage ist beschränkt! Aus diesem Grund wird ausdrücklich empfohlen, als Anfragemethode `POST` zu wählen.

Die MFPL-API sendet bei jeder Antwort die Option `Access-Control-Allow-Origin` mit dem Wert `*` in den Headerinformationen, sodass der Zugriff von jeder Domain aus möglich ist.

In den folgenden Abschnitten werden die Onlinedienste und ihre Nutzung näher erläutert.

Abkürzungsverzeichnis

MFPL	- Museumsfahrplan
REST-API	- Serverbasierte Programmschnittstelle
JSON	- JavaScript Object Notation
HTTP(S)	- Hyper Text Transfer Protocol
GET	- HTTP-Protokoll zum Datenaustausch über die Adresszeile
POST	- HTTP-Protokoll zum Austausch unbegrenzter Datenmengen
EFA	- Elektronische Fahrplanauskunft
GTFS	- General Transfer Feed Specification
GTFS-RT	- General Transfer Feed Specification (Real Time)

1. Datenobjekte

Der Datenaustausch mit der MFPL-API erfolgt auf Basis von JSON – Objekten. Die genauen Strukturen dieser Objekte werden in diesem Abschnitt erläutert.

Bestimmte Objekte und Felder sind dabei zwingend erforderlich, andere wiederrum nur optional. Diese Objekte und Felder sind durch einen entsprechenden Eintrag in der Beschreibungstabelle

- **R** - required / zwingend erforderlich
- **CR** - conditional required / ggf. erforderlich
- **O** - optional / nicht zwingend erforderlich

gekennzeichnet.

Die Datentypen innerhalb der Objekte sind an Hochsprachen wie Java orientiert und haben in dieser Dokumentation dieselbe Bedeutung.

1.1 Container-Objekt

Alle Anfrage – und Antwortobjekte werden in einem Container-Objekt gekapselt.

Bezeichnung	Typ		Beschreibung
app_name	String	R	Zugangsname der Anwendung
api_key	String	R	API-Key der Anwendung
timestamp	String	R	UNIX-Zeitstempel
instance	String	CR	Eindeutige Instanznummer der Anfrage
request	Object	CR	Anfragestruktur für den entspr. Onlinedienst
delivery	Object	CR	Antwortstruktur der MFPL-API

Tabelle 1: Container-Objekt

Die ersten beiden Felder dienen der Anmeldung bei einer Anfrage an die MFPL-API. Für eine Anfrage an die MFPL-API werden hier die Zugangsdaten eingetragen, die von der Administration zugeteilt wurden.
Bei einer Antwort ist das Feld hingegen `app_name` grundsätzlich mit „de.mfpl.api“ gefüllt, während das Feld `api_key` den Wert null hat.

Im Feld `timestamp` wird der Zeitstempel im UNIX-Format eingetragen, zu dem das Objekt erstellt wurde. Wird kein Zeitstempel angegeben oder ist dieser älter als 2,5 Minuten, so wird die Anfrage mit einer Fehlermeldung abgelehnt.

Der im Feld `instance` eingetragene Wert ist eine Referenz auf die vom Server gespeicherte Log-Datei zu dieser Anfrage. Mit Hilfe dieser Log-Datei kann von der Administration ein Fehler nachvollzogen werden.

Abhängig davon, ob es sich um eine Anfrage oder eine Antwort handelt, ist dem *Container*-Objekt ein *Request*-Objekt oder ein *Delivery*-Objekt beigefügt. Hierzu dienen die Felder `request` und `delivery`.

Der genaue Inhalt der Felder ist abhängig vom angefragten Onlinedienst und wird in dessen Abschnitt jeweils näher erläutert.

1.2 Date-Objekt

Mit Hilfe des *Date*-Objekts kann entweder ein einzelnes Datum oder ein Zeitraum abgebildet werden. Die Uhrzeit wird hierbei nicht berücksichtigt.

Bezeichnung	Typ		Beschreibung
single	String	CR	Einzeldatum im Format *yyyyMMdd*
range	Object	CR	Zeitraum

Tabelle 2: Date-Objekt (Single)

Durch das Feld `single` wird ein Einzeldatum beschrieben. Das Feld `range` enthält ein Objekt, das einen Zeitraum beschreibt.

Bezeichnung	Typ		Beschreibung
start_date	String	R	Einzeldatum im Format *yyyyMMdd*
end_date	String	R	Einzeldatum im Format *yyyyMMdd*

Tabelle 3: Date-Objekt (Range)

Die Felder `start_date` und `end_date` enthalten jeweils ein Einzeldatum. Beide Felder zusammen beschreiben den gewünschten Zeitraum.

Werden die Felder `single` und `range` beide angegeben, so wird das Feld `single` vorrangig bei der Verarbeitung ausgewertet.

1.3 Position-Objekt

Das *Position*-Objekt beschreibt einen Standort auf Basis von GPS-Koordinaten im Format WGS 84.

Bezeichnung	Typ		Beschreibung
lat	Double	R	Breitengrad im Format WGS 84
lng	Double	R	Längengrad im Format WGS 84
distance	Int	O	Distanz zum aktuellen Standard in [m]

Tabelle 4: Position-Objekt

Das Feld `distance` wird nur vom *LocationInformationService* verwendet. In diesem Feld wird die Distanz zum aktuellen Standort angegeben.

1.4 Error-Objekt

Mit Hilfe des *Error*-Objekts werden Fehler innerhalb von *Delivery*-Objekten der MFPL-API durch einen Fehlercode[1] und eine lesbare Fehlermeldung dargestellt.

Bezeichnung	Typ		Beschreibung
error_code	String	R	Eindeutiger Fehlercode
error_message	String	R	Fehlermeldung im Klartext
error_details	String	O	Details zur vorliegenden Fehlermeldung

Tabelle 5: Error-Objekt

[1] Übersicht über alle Fehlercodes: Siehe Abschnitt

1.5 Stop-Objekt

Das *Stop*-Objekt dient zur Beschreibung einer Haltestelle.

Bezeichnung	Typ		Beschreibung
stop_id	String	R	Eindeutige Stop-ID
stop_name	String	R	Bezeichnung
stop_desc	String	O	Beschreibung
position	Object	R	GPS-Position als *Position*-Objekt
realtime	Object	O	*Realtime*-Objekt mit Echtzeitinformationen

Tabelle 6: Stop-Objekt

1.6 Route-Objekt

Mit Hilfe des *Route*-Objektes lassen sich Informationen rund um eine Linienroute darstellen.

Bezeichnung	Typ		Beschreibung
route_id	String	R	Eindeutige Route-ID
route_type	Int	R	Linientyp gem. der GTFS-Spezifizierung
route_short_name	String	CR	Kurzbezeichnung der Route
route_long_name	String	CR	Langbezeichnung der Route
route_desc	String	O	Beschreibung der Route
route_url	String	O	Informations-URL zu der Route
route_color	String	O	Hintergrundfarbe der Route
route_text_color	String	O	Farbe zur Textdarstellung der Route
position	Object	O	Referenzpunkt (Zentrum) der Route als *Position*-Objekt
agency_id	String	CR	Eindeutige ID des Verkehrsunternehmens *Nur eingetragen, wenn kein Agency-Objekt enthalten ist!*
agency	Object	CR	Informationen über das Verkehrsunternehmen als *Agency*-Objekt *Nur eingetragen, wenn keine Agency-ID enthalten ist!*
realtime	Object	O	*Realtime*-Objekt mit Echtzeitinformationen

Tabelle 7: Route-Objekt

Das Feld `position` dient hierbei als Referenz, um die Linienroute z.B. in einem größeren Kartenausschnitt an der richtigen Stelle anzeigen zu können.

Wird das Feld `include_agency` in den `options` des *Request*-Objekts auf `true` gesetzt so wird im Feld `agency` des *Route*-Objektes ein vollständiges *Agency*-Objekt eingetragen. Andernfalls wird im *Route*-Objekt ausschließlich die eindeutige ID des betreibenden Verkehrsunternehmens eingetragen.

1.7 Agency-Objekt

Ein *Agency*-Objekt kapselt Informationen über ein Verkehrsunternehmen.

Bezeichnung	Typ		Beschreibung
agency_id	String	R	Eindeutige ID des Verkehrsunternehmens
agency_name	String	R	Bezeichnung des Verkehrsunternehmens
agency_url	String	R	Informations-URL über das Verkehrsunternehmen
agency_phone	String	O	Service-Nummer des Verkehrsunternehmens
agency_fare_url	String	O	Informations-URL zu Beförderungstarifen des Verkehrsunternehmens
realtime	Object	O	*Realtime*-Objekt mit Echtzeitinformationen

Tabelle 8: Agency-Objekt

1.8 Trip-Objekt

Das *Trip*-Objekt kapselt Informationen über eine einzelne Fahrt.

Bezeichnung	Typ		Beschreibung
trip_id	String	R	Eindeutige Trip-ID
route_id	String	CR	Eindeutige Route-ID der zugeordneten Route *Nur eingetragen, wenn kein Route-Objekt enthalten ist!*
route	Object	CR	Informationen über die Route als Route-Object *Nur eingetragen, wenn keine Route-ID enthalten ist!*
trip_headsign	String	O	Angezeigter Zieltext der Fahrt
trip_short_name	String	O	Externe Fahrtnummer
direction_id	String	O	Fahrtrichtung (0=Hinfahrt, 1=Rückfahrt)
block_id	String	O	Umlaufnummer
wheelchair_accessible	String	O	Information zur Barrierefreiheit
bikes_allowed	String	O	Information zur Fahrradmitnahme
stop_times	Array<Object>	O	Array mit *StopTime*-Objekten
frequency	Object	O	*Frequency*-Objekt zur Abbildung von Taktzeiten
shape	Object	CR	Fahrweg als *Shape*-Objekt
realtime	Object	O	*Realtime*-Objekt mit Echtzeitinformationen

Tabelle 9: Trip-Objekt

Das Feld `stop_times` enthält alle Abfahrtszeiten der Fahrt als Array mit *StopTime*-Objekten.

Wird das Feld `include_route` in den `options` des *Request*-Objekts auf `true` gesetzt so wird im Feld `route` des *Trip*-Objektes ein vollständiges *Route*-Objekt eingetragen. Andernfalls wird im *Trip*-Objekt ausschließlich die eindeutige ID der zugehörigen Linienroute eingetragen.

Mit Hilfe des Feldes `frequency` lässt sich die Taktzeit einer Fahrt durch ein *Frequency*-Objekt abbilden. Aus dieser Taktzeit können dann einzelne Fahrten interpoliert werden.

Wird in den `options` des *Request*-Objektes das Feld `include_shape` auf `true` gesetzt, so wird dem *Trip*-Objekt ein vollständiges *Shape*-Objekt beigefügt, sofern für die Fahrt ein graphischer Fahrweg festgelegt ist.

Die Option `include_shape` sollte nur verwendet werden, wenn ein einzelnes *Trip*-Objekt von der Schnittstelle angefragt wird, da die Datenmenge ansonsten signifikant anwachsen kann!

1.9 StopTime-Objekt

Durch das *StopTime*-Objekt wird die Abfahrt einer Fahrt an einer bestimmten Haltestelle dargestellt. Die enthaltenen Felder entsprechen der GTFS-Spezifizierung.

Bezeichnung	Typ		Beschreibung
trip_id	String	R	Eindeutige ID der Fahrt
stop_id	String	CR	Eindeutige ID der Haltestelle
stop	Object	CR	Haltestelle als *Stop*-Objekt
arrival_time	String	R	Ankunftszeit im Format hh:mm:ss
departure_time	String	R	Abfahrtszeit im Format hh:mm:ss
pickup_type	String	R	Einstiegstyp (0=Standard, 1=Kein Einstieg, 3=Bedarfshalt)
drop_off_type	String	R	Ausstiegstyp (0=Standard, 1=Kein Ausstieg, 3=Bedarfshalt)
realtime	Objekt	O	*Realtime*-Objekt mit Echtzeitinformationen

Tabelle 10: StopTime-Objekt

Das *StopTime*-Objekt enthält entweder ein vollständiges *Stop*-Objekt zur Darstellung der entsprechenden Haltestelle oder nur die zugeordnete Stop-ID der Abfahrtszeit.
Das erstgenannte Verhalten kann erzwungen werden, indem in den `options` des *Request*-Objekts das Feld `include_stops` auf `true` gesetzt wird.

1.10 Frequency-Objekt

Taktzeiten einer Fahrt werden mit Hilfe des *Frequency*-Objekts beschrieben. Die enthaltenen Felder entsprechen mit Ausnahme des Feldes `distance` der GTFS-Spezifizierung.
Im Feld `distance` wird der Taktabstand abweichend in Minuten angegeben.

Bezeichnung	Typ		Beschreibung
start_time	String	R	Beginn der Taktzeit im Format hh:mm:ss
end_time	String	R	Ende der Taktzeit im Format hh:mm:ss
headway	Int	R	Taktabstand der Fahrten in [min]
exact_times	Int	R	Fahrzeiten sind variierbar / fest angegeben

Tabelle 11: Frequency-Objekt

1.11 Shape-Objekt

Das *Shape*-Objekt dient zur Abbildung eines graphischen Fahrweges auf einer Karte oder in einem Geoinformationssystem. Es enthält ausschließlich ein Array mit *Position*-Objekten,
die die einzelnen Punkte des Fahrweges auf der Karte abbilden.

Bezeichnung	Typ		Beschreibung
points	Array<Object>	R	Array mit *Position*-Objekten

Tabelle 12: Shape-Objekt

1.12 Day-Objekt

Das Day-Objekt dient zur Abbildung eines Kalendertages und zeigt ein Datum, sowie eine an diesem Datum verkehrende Linie.

Bezeichnung	Typ		Beschreibung
date	String	R	Datum im Format yyyyMMdd
route	Object	R	Verkehrende Linie als *Route*-Objekt

Tabelle 13: Day-Objekt

1.13 Realtime-Objekt

Mit Hilfe des *Realtime*-Objektes werden Echtzeitinformationen innerhalb von anderen Datenobjekten gekapselt.

In der momentanen Ausbaustufe enthält das *Realtime*-Objekt nur ein Array mit *Alert*-Objekten, welche mit Verkehrsmeldungen bekannter EFA-Systeme verglichen werden können.
Nach aktuellem Stand können diese Meldungen mit Objekte vom Typ *Stop*, *Agency*, *Route* und *Trip* verknüpft und zusammen mit diesen von der MFPL-API übermittelt werden.

Um Echtzeitdaten in die Anfrage einzubinden, muss in den `options` der Anfrage das Feld `include_realtime` auf `true` gesetzt werden.

Bezeichnung	Typ		Beschreibung
alerts	Array<Object>	R	Array mit *Alert*-Objekten
trip_update	Objekt	O	*TripUpdate*-Objekt mit Echtzeitdaten zur Fahrt
stop_time_update	Objekt	O	*StopTimeUpdate*-Object mit Echtzeitdaten zur Fahrplanlage

Tabelle 14: Realtime-Objekt

Die Felder `trip_update` und `stop_time_update` sind nur enthalten, wenn das *Realtime*-Objekt einem entsprechenden *Trip*-Objekt oder einem *StopTime*-Objekt untergeordnet ist.

1.14 Alert-Objekt

Das Alert-Objekt enthält jeweils eine Verkehrsmeldung zur zusätzlichen Information über die statischen Solldaten hinaus. Als Beispiel können beliebige Infotexte zur Aufhebung einer Haltestelle oder Informationen zu geänderten Fahrplänen heran gezogen werden.

Bezeichnung	Typ		Beschreibung
alert_id	Int	R	Eindeutige ID der Verkehrsmeldung
alert_header	String	R	Titel der Verkehrsmeldung
alert_description	String	CR	Freitext
alert_url	String	CR	URL für weiterführende Informationen
alert_cause	Int	R	Grund
alert_effect	Int	R	Auswirkungen

Tabelle 15: Alert-Objekt

In einem *Alert*-Objekt muss mindestens der Freitext oder eine URL, unter der weitere Informationen eingesehen werden können, enthalten sein.

Die Felder `alert_cause` und `alert_effect` enthalten codierte Informationen[2] zum Grund und der Auswirkung der Verkehrsmeldung. Diese Informationen haben keine betrieblichen Auswirkungen, können jedoch von einem Informationssystem ausgewertet und zur Optimierung der Darstellung genutzt werden.

[2] Codierung gem. GTFS-RT Spezifizierung

1.15 TripUpdate-Objekt

Das TripUpdate-Objekt dient zur Abbildung der Fahrplanlage und von Echtzeitinformationen einer Fahrt. Dem TripUpdate-Objekt sind dabei Informationen, die die gesamte Fahrt betreffen. Im Feld `schedule_relationship` wird die Fahrplanlage[3] (Planverkehr/Ausfall/Zusatzfahrt) codiert angegeben.

Die Felder `vehicle_label` und `vehicle_position` beziehen sich dabei auf das tatsächlich eingesetzte Fahrzeug und geben eine Fahrzeugnummer und die aktuelle Fahrzeugposition an.

Bezeichnung	Typ		Beschreibung
update_id	String	R	Eindeutige ID des Echtzeitdatensatzes
timestamp	String	R	Zeitpunkt der Erstellung des Echtzeitdatensatz
schedule_relationship	String	R	Fahrplanlage / Fahrtstatus
vehicle_label	String	O	Fahrzeugnummer der Fahrt
vehicle_position	Objekt	O	Aktuelle Fahrzeugposition als *Position*-Objekt

Tabelle 16: TripUpdate-Objekt

1.16 StopTimeUpdate-Objekt

Mit Hilfe des *StopTimeUpdate*-Objektes werden Echtzeitinformationen zur Fahrplanlage und zum Status einzelner Fahrplanhalte übermittelt. Um die Datenmenge geringer zu halten, enthält das Objekt ausschließlich die Ankunfts - / Verspätungsminuten, sowie den Status des Fahrplanhalts. Die Sollfahrzeiten können aus dem übergeordneten *StopTime*-Objekt entnommen werden.

Das Feld `update_id` enthält dabei dieselbe ID wie das *TripUpdate*-Objekt, d.h. zu jedem *StopTimeUpdate*-Objekt besitzt die entsprechende Fahrt auch ein *TripUpdate*-Objekt.

Im Feld `schedule_relationship` wird der Status des Fahrplanhalts[4] (Planhalt/Ausfall) codiert übermittelt.

Bezeichnung	Typ		Beschreibung
update_id	String	R	Eindeutige ID des übergeordneten Echtzeitdatensatzes
arrival_delay	Int	R	Ankunftsverspätung in [min]
departure_delay	Int	R	Abfahrtsverspätung in [min]
schedule_relationship	String	O	Fahrplanlage / Haltestatus

Tabelle 17: StopTimeUpdate-Objekt

[3] Codierung gem. GTFS-RT Spezifizierung

[4] Codierung gem. GTFS-RT Spezifizierung

2. LocationInformationService

Der Onlinedienst *LocationInformationService* dient der standortbasierten Ermittlung von Haltestellen und Linienrouten. Das hierzu erforderliche *Request*-Objekt enthält dazu die Felder `location`, `options` und `filter`.

2.1 Anfragestruktur

Als Basis für die Anfrage dienen ein genauer GPS-Standort und ein Suchradius. Diese Informationen werden in einem *Location*-Objekt gekapselt.

Bezeichnung	Typ		Beschreibung
position	Object	CR	Aktueller Standort als *Position*-Objekt
radius	Int	O	Suchradius in [m], in dessen Bereich nach Ergebnissen gesucht wird *Default = 50.000*

Tabelle 18: Location-Objekt

Das Feld `position` gibt dabei den aktuellen Standort als Bezugspunkt in Form eines *Position*-Objektes an.

Der Suchradius lässt sich optional durch das Feld `radius` spezifizieren. Wir dieses Feld nicht angegeben, so geht die MFPL-API von einem Standardsuchradius von 50.000m (50km) um den aktuellen Standort aus.

Optional können die gewünschten Ergebnisse durch entsprechende Einträge im Feld `options` genauer spezifiziert werden.

Bezeichnung	Typ		Beschreibung
include_stops	Boolean	O	Gibt an, ob nach Haltestellen gesucht werden soll *Default = false*
include_routes	Boolean	O	Gibt an, ob nach Linienrouten gesucht werden soll *Default = false*
include_agency	Boolean	O	Gibt an, ob in einer Linienroute statt einer agency_id ein *Agency*-Objekt in einer Route enthalten sein soll *Default = false*
include_realtime	Boolean	O	Gibt an, ob dem Ergebnis zusätzl. Echtzeitdaten beigefügt werden sollen *Default = false*
limit	Int	O	Limitiert die Anzahl der Ergebnisse pro Typ auf die angegebene Zahl *Default = 10*

Tabelle 19: Suchoptionen im LocationInformationService

Durch einfügen entsprechender Einträge in das Feld `filter` kann das Suchergebnis im Linientyp und im Datum eingegrenzt werden. Wie beim `options`-Feld sind alle Einträge und das Feld selbst optional bei der Anfrage anzugeben.

Bezeichnung	Typ		Beschreibung
route_types	Array<Int>	O	Schränkt das Suchergebnis auf die angegebenen Linientypen ein[5]
date	Object	O	Schränkt das Suchergebnis auf Haltestellen und Linienrouten ein, die in dem durch das *Date*-Objekt referenzierten Bezugszeitraum mindestens einmal bedient werden
wheelchair_accessible	Boolean	O	Schränkt das Suchergebnis auf auf Linien und Haltestellen ein, die barrierefrei verfügbar sind / über die keine genaue Information vorliegt
bikes_allowed	Boolean	O	Schränkt das Suchergebnis auf Linien und Haltestellen ein, bei denen Fahrradmitnahme möglich ist / über die keine genaue Information vorliegt

Tabelle 20: Filtermöglichkeiten im LocationInformationService

2.2 Antwortstruktur

Die Suchergebnisse aus dem *LocationInformationService* werden als Objekte in Arrays geliefert.
Hierzu dienen im *Delivery*-Objekt die Felder `stops`, `routes` und `places`, welche abhängig von den Anfrageparametern gefüllt werden.
Wurden keine Ergebnisse gefunden, so sind die Felder leer. Die Felder der in den Arrays enthaltenen Objekte entsprechen in Inhalt und Struktur weitestgehend der GTFS-Spezifizierung. Abweichungen hiervon werden in der Beschreibung gesondert beschrieben. Weitere Informationen sind der GTFS-Spezifikation zu entnehmen.

Bezeichnung	Typ		Beschreibung
stops	Array<Object>	CR	Array mit *Stop*-Objekten
routes	Array<Object>	CR	Array mit *Route*-Objekten
error	Object	O	*Error*-Objekt im Falle eines Fehlers

Tabelle 21: Delivery-Objekt des LocationInformationService

[5] Linientypen gem. GTFS-Spezifizierung / Benutzerdefinierte Typen

2.3 Einsatzmöglichkeiten

Der *LocationInformationService* stellt mit den Informationen zu Haltestellen und Linien eine der wichtigsten Informationsquellen für Auskunftssysteme dar.

Die Suche erfolgt grundsätzlich auf Basis eines Standorts. Ob hierzu der aktuelle Gerätestandort oder eine andere Quelle (z.B. Geodaten) herangezogen wird, spielt dabei keine Rolle. Zusätzlich kann zum Standort ein Suchradius angegeben werden. Alle Suchergebnisse außerhalb dieses Suchradius werden dann ignoriert.

Ein mögliches Nutzungsszenario wäre die Darstellung von Haltestellen und Linienrouten abhängig vom aktuellen Zoomlevel der Karte einer Smartphone-App:

- Bei einem hohen Zoomlevel werden die Haltestellen an ihrer Position in der Karte angezeigt
- Bei einem niedrigen Zoomlevel werden statt der Haltestellen die Linienrouten an ihrem Referenzpunkt in der Karte angezeigt

Die Liste mit gesuchten Objekten wird jeweils beim Verschieben der Karte und beim Ändern des Zoomlevels neu geladen.

3. TripInformationService

Der Onlinedienst *TripInformationService* liefert Informationen über Fahrten aus Basis einer gegebenen Trip –, Route – oder Block – ID.

Zur Suche dienen die Felder `trip_id`, `route_id` und `block_id`. Diese Felder bilden untereinander die Rangfolge

- `trip_id`
- `route_id`
- `block_id`

ab. Während die Felder `route_id` und `block_id` ein Array mit *Trip*-Objekten liefern, erhält man durch eine Anfrage mittels `trip_id` nur ein einzelnes *Trip*-Objekt.

Die Suche und ihre Detailgenauigkeit lassen sich über die Felder `filter` und `options` genauer festlegen.

Wird zwei oder mehr der Felder trip_id, route_id oder block_id gleichzeitig gefüllt, so wird nur das ranghöchste Feld ausgewertet!

3.1 Anfragestruktur

Als Basis für dienen die o.g. Felder `trip_id`, `route_id` und `block_id`.

Bezeichnung	Typ		Beschreibung
trip_id	String	CR	Trip-ID
route_id	String	CR	Route-ID
block_id	String	CR	Umlaufnummer

Tabelle 22: Suchfelder im TripInformationService

Die Suchergebnisse lassen sich über das Feld `options` genauer spezifizieren. Alle Einträge sind hierbei genau wie das gesamte `options`–Feld nicht zwingend erforderlich.

Bezeichnung	Typ		Beschreibung
include_full_stop_times	Boolean	O	Gibt an, ob die Abfahrtszeiten als *StopTime*-Objekt enthalten sein sollen *Default = true*
include_stops	Boolean	O	Gibt an, ob Haltestellen als vollständiges *Stop*-Objekt, oder nur deren eindeutige ID enthalten sein soll *Default = false*
include_shape	Boolean	O	Gibt an, ob der graphische Fahrweg enthalten sein soll *Default = false*
include_route	Boolean	O	Gibt an, ob Informationen zur Linienroute als *Route*-Objekt enthalten sein sollen *Default = false*
include_agency	Boolean	O	Gibt an, ob Informationen zum Verkehrsunternehmen als *Agency*-Objekt enthalten sein sollen *Default = false*
include_realtime	Boolean	O	Gibt an, ob dem Ergebnis zusätzl. Echtzeitdaten beigefügt werden sollen *Default = false*
limit	Int	O	Limitiert die Anzahl der Ergebnisse auf die angegebene Zahl *Default = 10*

Tabelle 23: Suchoptionen im TripInformationService

Durch einfügen entsprechender Einträge in das Feld `filter` kann das Suchergebnis im Linientyp eingegrenzt werden. Wie beim `options`-Feld sind alle Einträge und das Feld selbst optional bei der Anfrage anzugeben.

Bezeichnung	Typ		Beschreibung
date	Object	O	Schränkt das Suchergebnis auf Fahrten ein, die in dem durch das *Date*-Objekt referenzierten Bezugszeitraum mindestens einmal verkehren
time	String	O	Schränkt das Suchergebnis auf Fahrten ein, deren Startzeit nach dem angegebenen Zeitpunkt im Format *hh:mm:ss* liegt
wheelchair_accessible	Boolean	O	Schränkt das Suchergebnis auf auf Fahrten ein, die barrierefrei verfügbar sind oder über die keine genaue Information vorliegt
bikes_allowed	Boolean	O	Schränkt das Suchergebnis auf Fahrten ein, bei denen Fahrradmitnahme möglich ist oder über die keine genauen Informationen vorliegen

Tabelle 24: Filtermöglichkeiten im TripInformationService

3.2 Antwortstruktur

Der *TripInformationService* liefert ein Array mit *Trip*-Objekten, die mit den gegebenen Suchkriterien übereinstimmen. Im *Delivery*-Objekt dient hierzu der Eintrag `trips`. Wurden keine passenden Einträge gefunden, so ist das Feld leer.

Bezeichnung	Typ		Beschreibung
trips	Array<Object>	R	Array mit *Trip*-Objekten

Tabelle 25: Delivery-Objekt des TripInformationService

3.3 Einsatzmöglichkeiten

Mit Hilfe des *TripInformationService* lassen sich Fahrten nach Linienrouten oder Block-IDs sowie Details zu einzelnen Fahrten ermitteln.

Ein mögliches Nutzungsszenario hierzu sähe wie folgt aus: Auf Basis einer gegebenen Linienroute sollen alle Fahrten ermittelt werden, welche an einem bestimmten Datum und nach einer festgelegten Abfahrtszeit innerhalb dieser Linienroute noch verkehren.

Im Anschluss daran sollen alle möglichen Informationen über eine ausgewählte Fahrt ermittelt werden. Auf Basis der so ermittelten Daten können alle für den Benutzer relevanten Daten auf einer graphischen Benutzeroberfläche zur Anzeige gebracht werden.

4. StopInformationService

Mit Hilfe des *StopInformationService* können für eine gegebene Haltestelle Abfahrtstafeln erzeugt werden. Der Onlinedienst liefert hierzu alle Fahrten, welche an einer bestimmten Haltestelle oder einer ihr untergeordneten Halteposition mindestens einmal verkehren.

Die Suche und ihre Detailgenauigkeit lassen sich über die Felder `filter` und `options` genauer festlegen.

4.1 Anfragestruktur

Für die Anfrage ist lediglich eine gültige Stop-ID erforderlich, welche im Feld `stop_id` beim *Request*-Objekt eingetragen wird.

Bezeichnung	Typ		Beschreibung
stop_id	String	R	Stop-ID der Haltestelle, für die Fahrten gesucht werden sollen

Tabelle 26: Suchfelder im StopInformationService

Die Suchergebnisse lassen sich über das Feld `options` genauer spezifizieren. Alle Einträge sind hierbei genau wie das gesamte `options`-Feld nicht zwingend erforderlich.

Bezeichnung	Typ		Beschreibung
include_stops	Boolean	O	Gibt an, ob in den Fahrzeiten zusätzliche Informationen zur Haltestelle als *Stop*-Objekt enthalten sein sollen Default = false
include_route	Boolean	O	Gibt an, ob Informationen zur Linienroute als *Route*-Objekt enthalten sein sollen *Default = false*
include_agency	Boolean	O	Gibt an, ob Informationen zum Verkehrsunternehmen als *Agency*-Objekt enthalten sein sollen *Default = false*
include_realtime	Boolean	O	Gibt an, ob dem Ergebnis zusätzl. Echtzeitdaten beigefügt werden sollen *Default = false*
limit	Int	O	Limitiert die Anzahl der Ergebnisse auf die angegebene Zahl *Default = 10*

Tabelle 27: Suchoptionen im StopInformationService

Bei den erhaltenen Antwortobjekten sind nur die *Abfahrtszeit* an der referenzierten Haltestelle und die *Ankunftszeit* an der Zielhaltestelle der Fahrt enthalten.

Durch einfügen entsprechender Einträge in das Feld `filter` kann das Suchergebnis im Linientyp eingegrenzt werden. Wie beim `options`-Feld sind alle Einträge und das Feld selbst optional bei der Anfrage anzugeben.

Bezeichnung	Typ		Beschreibung
route_types	Array<Int>	O	Schränkt das Suchergebnis auf die angegebenen Linientypen ein[6]
date	Object	O	Schränkt das Suchergebnis auf Fahrten ein, die in dem durch das *Date*-Objekt referenzierten Bezugszeitraum mindestens einmal verkehren
time	String	O	Schränkt das Suchergebnis auf Fahrten ein, deren Abfahrtszeit an der referenzierten Haltestelle nach dem angegebenen Zeitpunkt im Abfahrtszeit an der referenzierten Haltestelle Format *hh:mm:ss* liegt
wheelchair_accessible	Boolean	O	Schränkt das Suchergebnis auf Fahrten ein, die barrierefrei verfügbar sind oder über die keine genaue Information vorliegt
bikes_allowed	Boolean	O	Schränkt das Suchergebnis auf Fahrten ein, bei denen Fahrradmitnahme möglich ist oder über die keine genauen Informationen vorliegen

Tabelle 28: Filtermöglichkeiten im StopInformationService

4.2 Antwortstruktur

Der *StopEventService* liefert ein Array mit *Trip*-Objekten, die mit den gegebenen Suchkriterien übereinstimmen. Im *Delivery*-Objekt dient hierzu der Eintrag `trips`. Wurden keine passenden Einträge gefunden, so ist das Feld leer.

Bezeichnung	Typ		Beschreibung
trips	Array<Object>	R	Array mit *Trip*-Objekten

Tabelle 29: Delivery-Objekt des StopInformationService

4.3 Einsatzmöglichkeiten

Der *StopInformationService* dient ist ein haltestellenbezogener Informationsdienst. Auf Basis einer vorhandenen Stop-ID lassen sich alle Abfahrten an dieser Haltestelle anzeigen.

Um die zu übertragende Datenmenge gering zu halten wird die Anfrage sinnvollerweise mit einem Bezugsdatum und einer Zeitangabe gesendet. Somit werden alle Abfahrten an einem bestimmten Datum nach einer definierten Uhrzeit geliefert.

[6] Linientypen gem. GTFS-Spezifizierung / Benutzerdefinierte Typen

5. CalendarService

Der CalendarService ermöglicht die Abfrage von verkehrenden Linien in einem gewissen Zeitraum in Form eines Kalenders. Die Anfrage kann entweder allgemein gehalten oder durch zusätzliche Eingabe der Fehler `route_id` auf eine Linie, oder eine `stop_id` auf eine Haltestelle festgelegt werden.

Werden die Fehler `route_id` und `stop_id` beide ausgefüllt, so wird ausschließlich der Wert inm Feld `route_id` berücksichtigt!

5.1 Anfragestruktur

Für die Anfrage sind grundsätzlich keine Parameter erforderlich. In diesem Fall liefert der Onlinedienst einen Kalender für das folgende Jahr ab dem Anfragedatum.
Alternativ kann dem *Request*-Objekt im Feld `date` ein Zeitraum in Form eines *Date*-Objektes mitgegeben werden. Der Kalender wird dann für diesen Zeitraum berechnet.

Bezeichnung	Typ		Beschreibung
date	Object	R	Zeitraum für den der Kalender berechnet werden soll als *Date*-Objekt
route_id	String	O	Route-ID einer Linie, für die der Kalender berechnet werden soll
stop_id	String	O	Stop-ID einer Haltestelle, für die der Kalender berechnet werden soll

Tabelle 30: Suchfelder des CalendarService

Die Suchergebnisse lassen sich über das Feld `options` genauer spezifizieren. Alle Einträge sind hierbei genau wie das gesamte `options`–Feld nicht zwingend erforderlich.

Bezeichnung	Typ		Beschreibung
include_agency	Boolean	O	Gibt an, ob Informationen zum Verkehrsunternehmen als *Agency*-Objekt enthalten sein sollen *Default = false*
include_realtime	Boolean	O	Gibt an, ob dem Ergebnis zusätzl. Echtzeitdaten beigefügt werden sollen *Default = false*

Tabelle 31: Suchoptionen im CalendarService

Durch einfügen entsprechender Einträge in das Feld `filter` kann das Suchergebnis im Linientyp eingegrenzt werden. Wie beim `options`–Feld sind alle Einträge und das Feld selbst optional bei der Anfrage anzugeben.

Bezeichnung	Typ		Beschreibung
route_types	Array<Int>	O	Schränkt das Suchergebnis auf die angegebenen Linientypen ein[7]
wheelchair_accessible	Boolean	O	Schränkt das Suchergebnis auf Linien ein, die barrierefrei verfügbar sind / über die keine genaue Information vorliegt
bikes_allowed	Boolean	O	Schränkt das Suchergebnis auf Linien ein, bei denen Fahrradmitnahme möglich ist / über die keine genaue Information vorliegt

Tabelle 32: Filtermöglichkeiten im CalendarService

[7] Linientypen gem. GTFS-Spezifizierung / Benutzerdefinierte Typen

5.2 Antwortstruktur

Der *CalendarService* liefert ein Array mit *Day*-Objekten, die mit den gegebenen Suchkriterien übereinstimmen. Im *Delivery*-Objekt dient hierzu der Eintrag `days`. Wurden keine passenden Einträge gefunden, so ist das Feld leer.

Bezeichnung	Typ		Beschreibung
days	Array<Object>	R	Array mit *Day*-Objekten

Tabelle 33: Delivery-Objekt des CalendarService

5.3 Nutzungsszenarien

Der CalendarService liefert einen Kalender mit allen Tagen, für die Fahrplandaten hinterlegt sind. Zusätzlich zu jedem Verkehrstag wird die Linie als *Route*-Objekt geliefert, die dann zu Informationszwecken ausgewertet werden kann.

Spezifische Anfragen für eine bestimmte Haltestelle lassen sich durch Einbeziehung des Feldes stop_id im Request-Objekt erreichen. In diesem Fall werden die Verkehrstage für alle Linien, welche an der spezifizierten Haltestelle mindestens einmal verkehren, errechnet.

6. Fehlercodierung

Fehlermeldungen werden innerhalb der MFPL-API einheitlich durch *Error*-Objekte dargestellt. Zusätzlich zu einer Fehlerbeschreibung enthält ein solches *Error*-Objekt auch eine eindeutige Fehlercodierung, der Rückschlüsse auf die Ursache ermöglicht.

Die folgende Tabelle gibt einen Überblick über alle möglichen Fehlercodierungen und ihre möglichen Ursachen.

Fehlercodierung	Quelle	Beschreibung
100	Allgemein	Interner Fehler – Dieser kann nur durch die Systemadministration nachvollzogen und behoben werden
101	Allgemein	In der gesendeten Anfragestruktur ist ein Fehler vorhanden! Möglicherweise ist ein Objekt unvollständig, oder es fehlen benötigte Felder
102	Allgemein	In der gesendeten Anfrage ist kein *Request*-Objekt enthalten
103	Allgemein	Bei der Authentifizierung ist ein Fehler aufgetreten. Entweder sind die gesendeten Zugangsdaten ungültig oder der API-Key ist bereits abgelaufen
104	Allgemein	Der Anfragezeitraum wurde überschritten. Entweder wurde bei der Anfrage kein Zeitstempel angegeben oder der angegebene Zeitstempel ist älter als die maximale Anfragedauer
105	Allgemein	Reserviert
106	Allgemein	Reserviert
107	Allgemein	Reserviert
108	Allgemein	Reserviert
109	Allgemein	Reserviert
201	LocationInformationService	Ungültige Location-Referenz Die übergebene GPS-Position ist ungültig
301	TripInformationService	Ungültige Trip-Referenz Entweder ist keines der erforderlichen Felder enthalten, oder die übergebene Trip-ID, Route-ID oder Umlaufnummer ist ungültig
401	StopInformationService	Ungültige Stop-Referenz Die übergebene Stop-ID ist ungültig
501	CalendarService	Ungültige Date-Referenz Das übergebene Date-Objekt ist ungültig