EFA JSON-Schnittstelle

HTTP-Parameter Dokumentation

Mentz GmbH
Grillparzerstraße 18
81675 München
Tel: +49 89 41868-0
Fax: +49 89 41868-160
info@mentz.net
www.mentz.net

Versionsgeschichte				
Dokument Version	Software Version	Datum	Name	Änderungsgrund
1.0		05.11.2014	SM	Initiale Version
1.1		11.02.2015	SM	Korrekturen
1.2		20.04.2017	SM	Parameter SpEncId entfernt, Parameter UTFMacro ergänzt
2.0		18.06.2021	SM	Aktualisierung auf JSON-Schnittstelle

Freigabe			
	Datum	Name	Unterschrift
geprüft:			
freigegeben:			

Inhaltsverzeichnis

1 Allgemeines

Dieses Dokument beschreibt die Nutzung der EFA JSON-Schnittstelle. Diese ermöglicht zustandsfreie Anfragen an die EFA und ist modular aufgebaut. Jeder Request entspricht einer Funktionalität der EFA.

Zunächst werden im Kapitel Anwendungsfälle die typischen Abläufe in einer elektronischen Fahrplanauskunft vorgestellt und deren Abbildung durch die Kombination verschiedener Requests aufgezeigt. Es folgen einige Kapitel über die Grundfunktionalität des Interfaces sowie über Funktionalität, die von verschiedenen Requests gemeinsam genutzt werden.

Anschließend werden die HTTP-Parameter zur Nutzung der folgende Requests beschrieben:

- StopFinder-Request: Punkt- bzw. Haltestellen-Suche
- ServingLines-Request: Anfrage der Bedienenden Linien
- Trip-Request: Anfrage von Fahrtauskünften
- PS-Request: Anfrage eines Pendlerfahrplans
- DM-Request: Anfrage eine Abfahrtsmonitors
- DMTTP-Request: Anfrage des Soll-Abfahrtsmonitors
- TripStopTimes-Request: Anfrage der durchfahrenen Haltestellen
- StopSeqCoord-Request: Anfrage der Koordinaten einer Haltestellensequenz
- LineStop-Request: Anfrage der durchfahrenen Haltestellen einer Linie
- StopTimetable-Request: Anfrage eines Aushangfahrplans für eine Linie
- CoordInfo-Request: Anfrage von Objekten und ihrer Koordinaten
- Geoobject-Request: Anfrage der Koordinaten und Haltestellen einer Linie
- ParkObject-Request: Anfrage von Detailinformation eines Parkobjekts
- OpArea-Request: Anfrage des Bediengebiets eines Sharers
- StopList-Request: Anfrage aller Haltestellen eines Verbundgebiets
- AddInfo-Request: Anfrage aktueller Meldungen
- SystemInfo-Request: Anfrage von Systeminformation

2 Anwendungsfälle

Ein erster Schritt bei der Nutzung des JSON-Interfaces muss die genaue Planung der Anwendungsfälle und deren Abbildung durch die verschiedenen modular aufgebauten Requests sein. Dieses Kapitel soll bei der Planung unterstützen, indem typische Anwendungsfälle vorgestellt werden.

2.1 Fahrplanauskunft

Eine Übersicht über die für die Fahrtauskunft relevanten Requests ist in Abbildung 1 zu sehen. Es werden mindestens der StopFinder-Request (1) zur Punktsuche und der Trip-Request (2) zur Fahrtanfrage benötigt.

Durch verschiedene optionale Requests kann Zusatzfunktionalität angeboten werden. Die Durchfahrenen Haltestellen eines Teilweges mit Abfahrts-, Ankunfts- und sofern vorhanden einschließlich Echtzeitinformation, werden mit dem TripStopTimes-Request (3) angefragt. Zum Einzeichnen des Fahrtverlaufs einschließlich Start- und Endpunkt sowie Umstiegshaltestellen, dient der StopSeqCoord-Request (4).

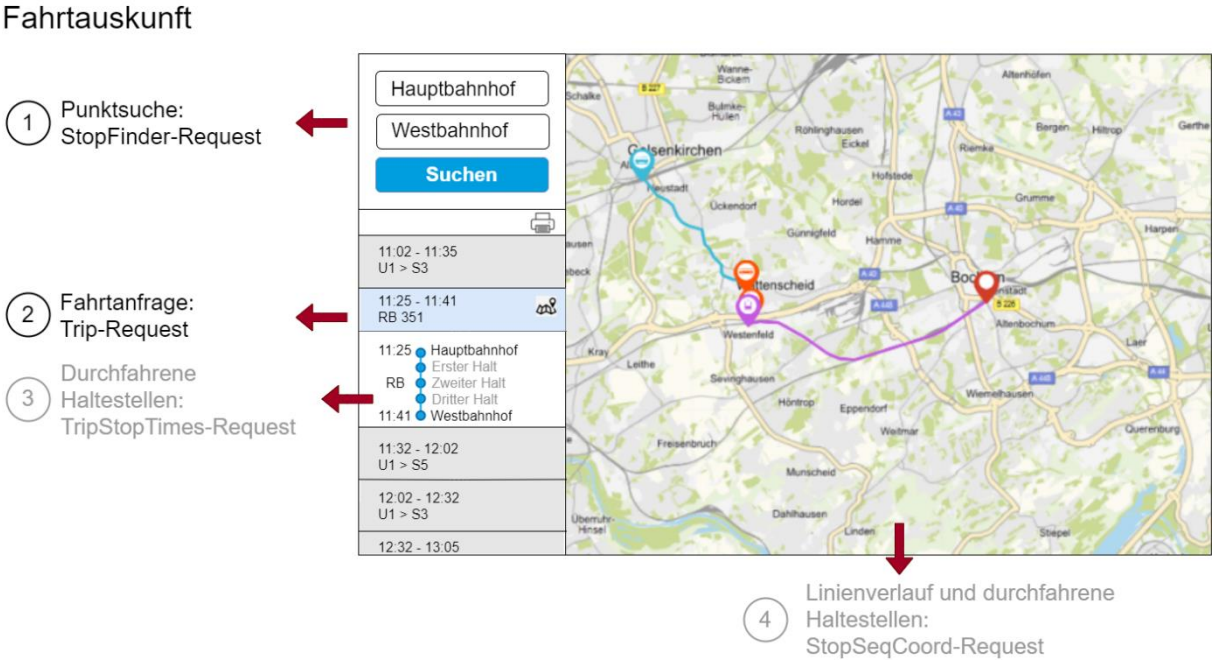

Abbildung 1 - Übersicht über die Requests, die für die Fahrtauskunft benötigt werden

2.2 Abfahrtstafel

Eine Übersicht über die für die Abfahrtstafel relevanten Requests ist in Abbildung 2 zu sehen. Es werden mindestens der StopFinder-Request (1) zur Punktsuche und der DM-Request (3) zur Anfrage der Abfahrten benötigt.

Optional können für die Implementierung eines Linienfilters die Bedienenden Linien mit dem ServingLines-Request (2) angefragt werden. Sollen für eine Abfahrt die durchfahrenen Haltestellen angezeigt werden, werden diese mit dem TripStopTime-

Request (4) angefragt. Die Anzeige von Uhrzeit und Echtzeitinformation ist möglich. Die Koordinatensequenz sowie die Haltestellen inklusive ihrer Koordinaten werden mit dem StopSeqCoord-Request angefragt.

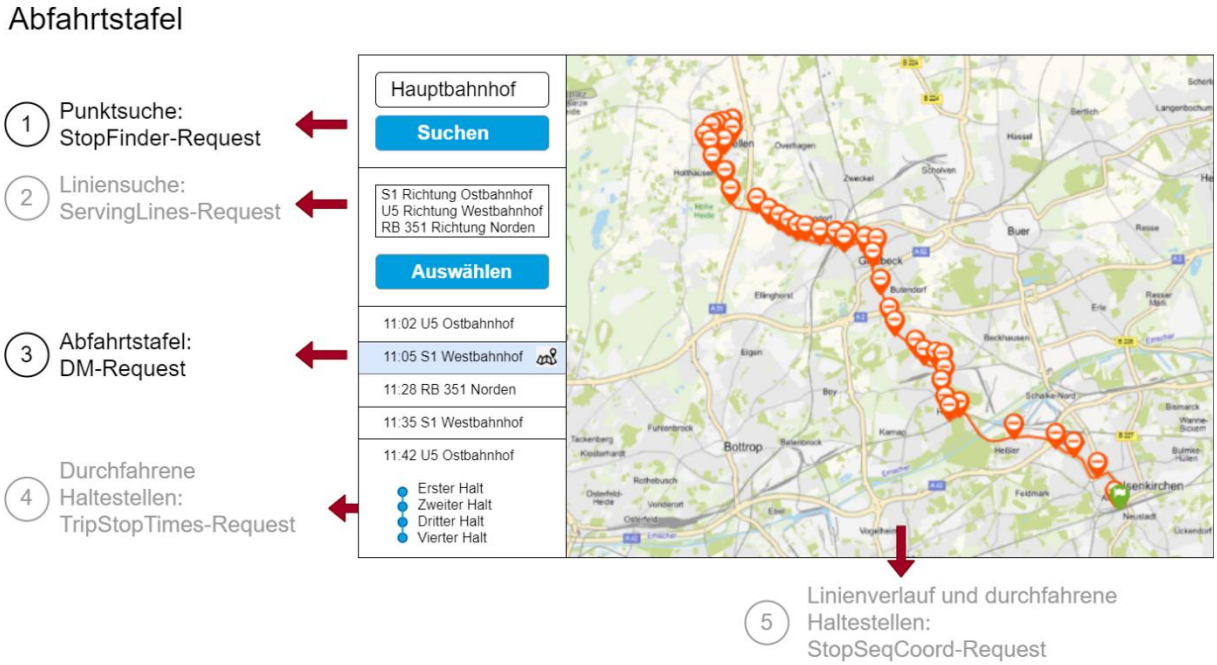

Abbildung 2 - Übersicht über die Requests, die für die Abfahrtstafel benötigt werden

2.3 Printprodukte

Verschiedene Requests erzeugen base 64 codierte Printrpodukte. Dazu gehört auch der StopTimetable-Request.

Printprodukte werden entweder für eine Haltestelle, eine Linie oder die Haltestelle einer Linie erstellt. Je nach dem was benötigt wird, werden die folgenden Requests miteinander kombiniert. Oft sind auch verschiedene Wege, ausgehend von einer Haltestellen- oder einer Liniensuche, möglich.

2.3.1 Möglichkeit 1: von der Haltestellensuche ausgehend

1. Haltestellensuche mit dem StopFinder-Request
2. Falls zusätzlich die Wahl einer bedienenden Linie notwendig ist: Bedienende Linien der Haltestelle mit dem ServingLines-Request und `mode=odv` anfragen und eine davon auswählen

2.3.2 Möglichkeit 2: von der Liniensuche ausgehend

1. Liniensuche mit dem ServingLines-Request und `mode=line`
2. Falls die Wahl einer durchfahrenen Haltestelle notwendig ist: Durchfahrene Haltestellen mit dem LineStopSeq-Request anfragen und eine auswählen

2.4 Interaktive Karte

Eine Übersicht über die für interaktive Karte relevanten Requests ist in Abbildung 3 zu sehen. Mit dem CoordInfo-Request (1) werden Kartenobjekte wie Haltestellen, wichtige Punkte inklusive ihrer Hierarchie, Sharer und Parkobjekte sowie ihre Koordinaten angefragt.

Der CoordInfo-Request dient ebenfalls dazu Detailinformationen von Sharern (5) anzufragen. Detailinformation für Parkobjekte erhält man hingegen mit dem ParkObject-Request (7). Soll beim Anklicken der Haltestellen eine Abfahrtstafel angezeigt werden, erfolgt die Anfrage ganz klassisch über den DM-Request (4).

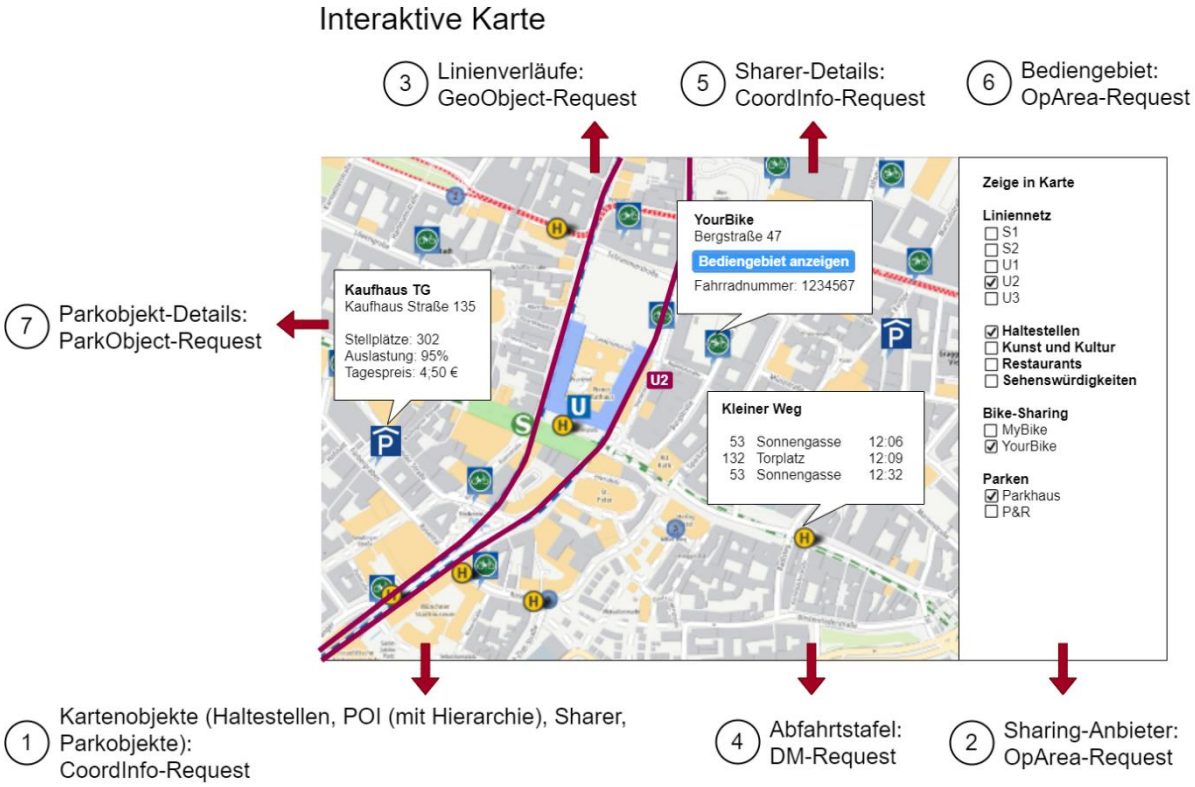

Abbildung 3 - Übersicht über die Requests, die für eine interaktive Karte benötigt werden

Eine Liste aller Sharing-Anbieter erhält man mit dem OpArea-Request (2). Dieser Request dient ebenfalls dazu das Bediengebiet eines Sharers (6) anzufragen.

Die Koordinaten und Haltestellen von Linien werden mit dem GeoObject-Request (3) angefragt. Auf diese Weise kann das Liniennetz interaktiv auf der Karte angezeigt werden.

3 Grundfunktionalität

3.1 Request-unabhängige Funktionalität

Unabhängig von den einzelnen Requests stellt das Interface folgende Grundfunktionalität zur Verfügung:

3.1.1 hideBannerInfo = 1

Potentiell kann jeder Request um aktuelle Meldungen angereichert sein. Werden diese nicht benötigt, ist es aus Performancegründen sinnvoll, diese mit diesem Parameter zu unterdrücken.

3.1.2 version

Soll eine andere als die aktuelle Version des JSON-Interfaces verwendet werden, kann diese per HTTP-Parameter angegeben werden. Wert des Parameters ist die gewünschte Versionsnummer.

Beispiel: `version=10.2.8.6`

3.2 Eingabe von Datum und Uhrzeit

Die Eingabe von Datum und Uhrzeit wird von vielen verschiedenen Requests verwendet. Zum Beispiel vom Trip-Request und DM-Request. Sie kann auf viele verschiedene Arten erfolgen, um unterschiedliche Datums- und Zeitformate abbilden zu können. Die Angabe der Parameter ist immer optional. Fehlende HTTP-Parameter werden so ergänzt, dass sie den aktuellen Datums- bzw. Uhrzeiteinstellungen des Fahrplanauskunft-Servers entsprechen.

Die Eingabe des Datums und der Zeit entspricht immer der Serverzeit. Ausgegeben wird die Zeit im UTC-Format (ISO 8601).

Der PS-Request verwendet eine abweichende Datums- und Uhrzeiteingabe, da hier Stichtage und ein Zeitintervall benötigt werden. Siehe Datum und Uhrzeit für den Pendlerfahrplan.

3.2.1 itdDate

Eingabe eines vollständigen Datums. Der Wert setzt sich dabei aus der vier- oder zweistelligen Jahreszahl, gefolgt von der zweistelligen Monatsangabe und der zweistelligen Angabe des Tages, zusammen.

3.2.2 itdDateDay

Eingabe des Tages. Der Tag wird als ein- oder zweistelliger Wert angegeben.

3.2.3 itdDateMonth

Eingabe des Monats. Der Monat wird als ein- oder zweistelliger Wert angegeben.

3.2.4 itdDateYear

Eingabe des Jahres. Das Jahr wird als zwei- oder vierstelliger Wert angegeben.

3.2.5 itdDateYearMonth

Eingabe von Jahr und Monat. Der Wert setzt sich dabei aus der vierstelligen Jahreszahl, gefolgt von der zweistelligen Monatszahl, zusammen.

3.2.6 itdDateDayMonthYear

Eingabe von Tag, Monat und Jahr. Der Wert setzt sich aus der zweistelligen Tagsangabe, gefolgt von der zweistelligen Monatsangabe, gefolgt von der zwei- oder vierstelligen Jahreszahl, zusammen. Auf Wunsch kann ein beliebiges Trennzeichen verwendet werden. In diesem Fall muss die Angabe der Jahreszahl jedoch immer vierstellig sein.

3.2.7 itdTime

Eingabe der Uhrzeit. Der Wert kann in einem der folgenden Formate angegeben werden:

- `HHMM`
- `HH:MM`
- `HH.MM`
- `MM`
- `HHMMa` (anglo-amerikanisches Format „am“)
- `HHMMh` (24-Stunden Format)
- `HHMMp` (anglo-amerikanisches Format „pm“)

3.2.8 itdTimeHour

Eingabe der Stunden. Der Wert gibt in ein- oder zweistelliger Form die Stunden an.

3.2.9 itdTimeMinute

Eingabe der Minuten. Der Wert gibt in ein- oder zweistelliger Form die Minuten an.

3.2.10 itdTimeAMPM

Bei der anglo-amerikanische Zeitangabe im 12-Stunden-Format unterscheidet der Wert mit *am* (Default) oder *pm* den Vor- und Nachmittag.

3.2.11 timeOffset

Der Wert gibt einen Zeit-Offset an, der auf die aktuelle Zeit des Fahrplanauskunft-Servers bzw. die Anfragezeit aufgerechnet wird. Die Angabe erfolgt in Minuten.

3.3 Eingabe eines Punktes

Die Eingabe von Punkten wird von vielen Requests benötigt. Der Trip-Request benötigt beispielsweise zwei Punkte: einen Start- und einen Zielpunkt. Der DM-Request hingegen nur einen. Wichtig ist, dass ein Punkt eindeutig identifiziert werden kann. Das ist auf zwei Arten möglich: Punkte wie Haltestellen, Adressen oder wichtige Punkte verfügen über eine eindeutige ID, alternativ kann eine Koordinate verwendet werden.

Im Folgenden werden die HTTP-Parameter vorgestellt, durch die ein Punkt an einen Request übergeben werden kann. Ist der Punkt unbekannt, kann er mithilfe des StopFinder-Requests gesucht werden. Dies ist der einzige Request, der keinen Volltreffer erwartet. Er wird verwendet, wenn der Anwender einen Punkt über eine freie Texteingabe suchen können soll.

Die im Zusammenhang der Punktverifikation häufig gebrauchte Abkürzung ODV steht für Origin-Destination-Via. Sie bezieht sich auf die Funktion, die der Punkt innerhalb eines Requests innehat. Die Funktion eines Punktes wird durch die Ergänzung `_<usage>` am Ende eines Parameters bestimmt. Parameter, die eine solche Ergänzung benötigen, werden für jeden Punkt separat definiert. Beim Trip-Request kann die Ergänzung `<usage>` folgende Werte annehmen:

- `origin` – Startpunkt
- `destination` – Zielpunkt
- `via` – Zwischenhalt

Welche Endungen für einen Request gültig sind, kann der Dokumentation des jeweiligen Requests entnommen werden.

3.3.1 Obligatorische Parameter

3.3.1.1 name_<usage>

Dieser Parameter enthält den Namen des Punkts. D.h. die ID oder Koordinate (siehe Eingabe einer ID oder Koordinate). Im Fall der Punktsuche mit dem StopFinder-Request kann er auch einen Freitext zur Punktsuche enthalten. Seine Funktion ist im Attribut `usage` angegeben.

3.3.1.2 type_<usage>

Dieser Parameter beschreibt den Typ des Punktes. Mögliche Werte sind:

- `any`
- `coord`

Außer bei der Koordinateneingabe ist der Wert `any`.

3.3.2 Nahegelegene Haltestellen löschen

Bei der Verwendung des JSON-Interfaces sollte die Anfrage der nahegelegenen Haltestellen bei allen Requests, die eine Punkteinabe verwenden, deaktiviert werden. Dies spart Rechenzeit und -aufwand.

Eine Ausnahme stellt der StopFinder-Request dar. Dieser übernimmt im JSON-Interface die Suche der nahegelegenen Haltestellen, wenn eine solche benötigt wird (siehe Wahl einer nahegelegenen Haltestelle). Soll keine nahegelegene Haltestelle gewählt werden, ist auch beim StopFinder-Request die Deaktivierung sinnvoll. Dies ist beispielsweise bei der Anzeige einer Vorschlagsliste der Fall.

3.3.2.1 deleteAssignedStops_<usage> = 1

Unterdrückt die Suche nach nahegelegenen Haltestellen.

3.3.3 Verschiedene Eingabemöglichkeiten ermöglichen

3.3.3.1 nameInfo_<usage>

Dieser Parameter überschreibt den Parameter `name_<usage>`. Muss der Parameter übergeben werden, soll jedoch keine Wirkung zeigen, kann er deaktiviert werden, indem der Wert auf *`invalid`* gesetzt wird.

3.3.3.2 typeInfo_<usage>

Dieser Parameter überschreibt den Parameter `type_<usage>`. Muss der Parameter übergeben werden, soll jedoch keine Wirkung zeigen, kann er deaktiviert werden, indem der Wert auf *`invalid`* gesetzt wird.

3.3.4 Eingabe einer ID oder Koordinate

Ein Punkt wird über seine ID oder über eine Koordinate identifiziert.

3.3.4.1 ID

Die ID einer Haltestelle, eines Punktes oder einer Adresse, kann beispielsweise mit dem StopFinder-Request ermittelt werden. Der Punkt enthält die Property `id`, deren Wert dem Parameter `name_<usasge>` übergeben wird.

Beispiel: `type_origin=any&name_origin=de:05911:5494`

3.3.4.2 Koordinaten

Eine Koordinate wird über die beiden Parameter `type_<usage>=coord` und `name_<usage>=<x>:<y>:<Koordinatenformat>:<Freitext>` eingegeben. Der Wert des Parameters `name_<usage>` setzt sich aus drei obligatorischen Werten und einem optionalen Wert, getrennt durch Doppelpunkte, zusammen. `<x>` und `<y>` stellen die x- und die y-Koordinate da. Der für die Wahl des Koordinatenformat ent-

scheidende Parameter ist `<Koordinatenformat>`. Dieser ist gewöhnlich `WGS84[dd.ddddd]`. Der letzte Wert, `<Freitext>` ist optional. Wird kein Freitext gewählt, versucht das System auf die nächstgelegene Straße zu schnappen.

Beispiel: `name_origin=6787835:51231000:WGS84[dd.ddddd]` oder `name_origin=6787835:51231000:WGS84[dd.ddddd]:ein schöner Ort`

3.4 Eingabe einer Linie

Einige Requests benötigen die Eingabe einer Linie. Beispielsweise beim DM-Request können Linien als optionales Filterkriterium verwendet werden.

Im Folgenden werden die HTTP-Parameter vorgestellt, durch die eine Linie an einen Request übergeben werden kann. Ist die Linie unbekannt, kann sie mithilfe des ServingLines-Request gesucht werden. Dies ist der einzige Request, der keinen Volltreffer bei der Linieneingabe erwartet. Er wird verwendet, wenn der Anwender eine Linie über eine freie Texteingabe oder über die Eingabe einer Haltestellen suchen können soll.

3.4.1 Obligatorische Parameter

3.4.1.1 line

Eine konkrete Linie wird durch ihre ID eingegeben. Falls die ID unbekannt ist, kann sie mit dem ServingLines-Request ermittelt werden.

Die ID setzt sich aus einer durch Doppelpunkt getrennte Liste von Teilnetz, DIVA-Liniennummer, Ergänzung und Richtung zusammen. Soll dabei einer der Parameter nicht berücksichtig werden, wird der Wert leer gelassen. Zur Wahl mehrerer Linien kann der Parameter mehrfach verwendet werden.

Beispiel: `apb:05136:12b:R` oder `apb:05136::`

3.4.2 Linienversion

Einige Requests, wie der Aushangfahrplan (siehe StopTimetable-Request) benötigen zusätzlich die Linienversion bzw. einen Stichtag für die Linienauswahl. Diese kann Bestandteil der ID sein bzw. an den Parameter `line` angehängt werden. Alternativ können folgende Parameter verwendet werden:

3.4.2.1 lineVer

Wählt die Linienversion aus. Ist die Linienversion nicht bekannt, kann alternativ das Datum mit dem Parameter `dateDay` übergeben werden. Die Linienversion wird dann anhand dessen intern ermittelt.

3.4.2.2 dateDay

Mit diesem Parameter kann der das Datum, für den der Aushangfahrplan, die Fahrplanbuchseite oder der Linienverlaufsplan erstellt werden soll, angegeben werden. Der Wert ist die vierstellige Jahreszahl, gefolgt von dem zweistelligen Monat und dem zweistelligen Tag. Die Angabe des Datums ist nur notwendig, wenn die Linienversion nicht bekannt ist.

3.4.3 Optionale Parameter

3.4.3.1 lsShowTrainsExplicit = 1

Standardmäßig werden keine Züge bei den bedienenden Linien ausgegeben. Die Anzeige der Züge in der Linienauswahl muss explizit durch diesen Parameter aktiviert werden.

3.5 Fehlerbehandlung

Fehler warden als ein Array `systemMessages` ausgegeben.

- `code` - ein nicht eindeutiger, vom Kontext abhängiger Fehlercode
- `error` - Beschreibung des Fehlers
- `type` - hat den Wert `message` oder `error`
- `module` - weist auf das EFA-Moodul hin, in dem der Fehler aufgetreten ist

Die Fehlercodes sind dem Dokument EFA9-10_Errorcodes zu entnehmen.

4 StopFinder-Request

Im Folgenden wird die Punktsuche mittels des StopFinder-Requests erläutert. Die hierfür verwendete EFA-Komponente ist der EFALocationServer. Über HTTP-Parameter wird ein Punkt, zum Beispiel eine Adresse, ein wichtiger Punkt oder eine Haltestelle, angefragt.

Die Punktsuche kann dazu verwendet werden, dem Anwender eine Freitexteingabe mit Vorschlagsliste zur Verfügung zu stellen (siehe Abbildung 4). Der so ermittelte „Volltreffer“ kann als eindeutige identifizierter Punkt anhand seiner ID in einen anderen Request übernommen werden.

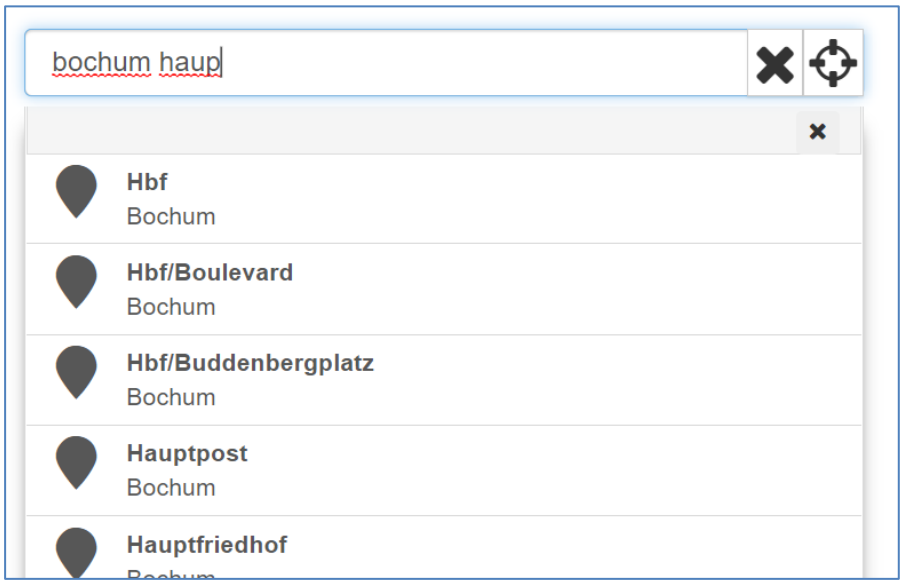

Abbildung 4 - Freitexteingabe mit Vorschlagliste zu Punktsuche

4.1 Ansteuerung des StopFinder-Requests

Der StopFinder-Request wird mit HTTP-Parametern gesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_STOPFINDER_REQUEST?HTTP-Parameter
```

4.2 Hinweis zur Eingabe von Punkten

Bei der Punktsuche werden die unter Eingabe eines Punktes beschriebenen Parameter verwendet. Als Erweiterung `<usage>` wird der Wert `sf` angegeben.

4.3 Obligatorische Parameter

Der StopFinder-Request erwartet mindestens die Eingabe eines Punktes.

4.4 Hilfetext für Eingabefelder

4.4.1 nameDefaultText_<usage>

Der durch den Parameter übergebene Text wird bei der Punktsuche nicht berücksichtigt. Dadurch ist es möglich im Eingabefeld einen Hilfetext anzuzeigen.

4.5 Optionale Parameter zur Optimierung der Punktsuche

4.5.1 anyMaxSizeHitList

Gibt die maximale Länge der Trefferliste an. Übersteigt die die Anzahl der gefundenen Treffer den angegebenen Wert, wird die Liste reduziert. Die Reduktion erfolgt anhand der Trefferqualität.

Falls die Trefferliste vor der Anwendung der definierten Bewertungskriterien reduziert wird, sollte der Wert nicht zu niedrig gewählt werden.

Hinweis: Ob die Liste vor oder nach der Anwendung der Bewertungskriterien reduziert wird, hängt von der Konfiguration des EFALocationServers ab. Eine weitere Möglichkeit zur Reduktion der Trefferliste wird unter Beeinflussung der Trefferqualität und Reduktion der Trefferliste beschrieben.

4.5.2 anyResSort_<usage>

Normalerweise gibt der EFALocationServer die Trefferliste unsortiert zurück. Die Sortierung wird von der Benutzeroberfläche durchgeführt. Das Standardverhalten kann mit diesem Parameter geändert werden. Sein Wert entspricht dem Namen eines Sortierkriterien-Sets `ResultSorter<ID>`, das in der Konfigurationsdatei des EFALocationServers festgelegt wird.

Hinweis: Die Verlagerung der Sortierung in den EFALocationServer hat den Vorteil, dass sie eine konstante Sortierreihenfolge in allen Benutzeroberflächen garantiert ist.

4.5.3 anySigWhenPerfectNoOtherMatches = 1

Wenn der gesuchte Punkt als perfekter Treffer erkannt wird, wird nur dieser berücksichtigt. Ansonsten werden auch Treffer, die der Eingabe ähnlich sind gefunden. Auf diese Weise kann entweder eine kurze Trefferlisten oder eine tolerante Suche ermöglicht werden.

4.5.4 locationInfoActive = 1

Aktiviert das Ermitteln aller Hausnummern zu einer verifizierten Straße. Die Ermittlung der Hausnummern erfolgt schließlich durch den Parameter `useHouseNumberList_<usage>`.

4.5.5 useHouseNumberList_<usage> = 1

Wird dieser Parameter eingeschaltet, wird im Rahmen der Adress-Verifikation eine Liste möglicher Hausnummern ausgegeben.

4.5.6 useLocalityMainStop = 1

Ein verifizierter Ort entspricht einer Koordinate, es müssen umliegende Haltestellen gesucht werden. Dieser Parameter ändert dieses Standardverhalten. Statt der Suche nach umliegenden Haltestellen werden die dem Ort zugeordneten Haltestellen verwendet. Auf diese Weise kann ein Ort mit einer Haupthaltestelle assoziiert werden.

4.5.7 doNotSearchForStops_<usage>=1

Bewirkt, dass nach der Verifizierung einer Koordinate nicht zu umliegenden Haltestellen geroutet wird. Der Parameter sollte eingesetzt werden, wenn ausschließlich Fahrten ohne Öffentliche Verkehrsmittel berechnet werden.

Hinweis: Die Verwendung dieses Parameters ist zum Beispiel sinnvoll, wenn eine Vorschlagsliste angezeigt werden soll.

4.5.8 prMinQu

Mit diesem Parameter kann eine Trefferqualität angegeben werden, die ein Treffer mindestens haben muss, um in die Trefferliste aufgenommen zu werden.

4.6 Suchkriterien

Mit den Suchkriterien kann der Raum für die Punktsuche eingeschränkt werden. Um die Anzahl der Parameter zu reduzieren und um Konsistenz über verschiedene Anwenderinterfaces hinweg zu erreichen, wird empfohlen die Suchkriterien in einem HTTP-Parameter Makro zusammen zu fassen. Das Konzept der Makros ist im Glossar HTTP-Parameter Makros erläutert.

Die Kriterien für die Punktsuche, das heißt die passende Parameterkombination, sollten vor der Implementation mit dem EFALocationServer Monitor getestet werden. In der Regel dauert es eine Weile bis die ideale Kombination ermittelt ist, da durch die Konzentration auf Sonderfälle schnell die Gesamtübersicht verloren geht. In der Regel ist das Testen mit dem Monitor weniger aufwändig als in der Benutzeroberfläche.

4.6.1 anyObjFilter_<usage>

Die Suche kann auf bestimmte Objekttypen eingegrenzt werden. Der Wert des Parameters ist eine Bitmaske. Die einzelnen Objekttypen lassen sich beliebig kombinieren. Die Objekttypen sind dem Anhang Objekttypen zu entnehmen.

Beispiel: `anyObjFilter_origin=34` (2 + 32 ➔ Haltestellen und POIs)

Hinweis: Möglicherweise sind nicht alle Objekttypen in den Daten verfügbar. Postleitzahlen und Kreuzungen sind bei deutschen Verbünden selten vorhanden.

4.6.2 anyOmc_<usage>

Bei der Verifikation werden nur Punkte innerhalb des Gemeindegebietes mit der durch den Parameter übergebenen Kennziffer berücksichtigt. Als Rückfallebene kann der Parameter `tryFurtherAnyWithoutLoc` aktiviert werden.

4.6.3 regionID_<usage>

Einschränkung des Suchraums auf die Region mit der durch den Parameter angegebenen ID. Die Regionen und ihre IDs müssen in einer separaten Datei definiert und die Funktionalität in der Konfigurationsdatei des EFALocationServers aktiviert werden.

4.6.4 tryFurtherAnyWithoutLoc = 1

Bei der Punktverifikation kann eine Gemeindekennziffer durch den Parameter `anyOmc_<usage>` angegeben werden. Wird dabei kein Treffer gefunden, wird eine erneute Suche ohne die vorgegebene Gemeindekennziffer gestartet, wenn dieser Parameter aktiv ist.

4.7 Beeinflussung der Trefferqualität und Reduktion der Trefferliste

Analog zu den Suchkriterien gilt auch für die folgenden Parameter, dass diese vor der Implementation mit dem EFALocationServer Monitor getestet werden sollten. Sie sollten auch dem Makro (siehe Glossar HTTP-Parameter Makros) für die Suchkriterien hinzugefügt werden.

Kann ein Punkt nicht eindeutig verifiziert werden, wird in der JSON-Ausgabe eine Liste von Punkten zurückgegeben (siehe auch Listenauswahl). Der beste Treffer ist durch das Attribut `isBest=1` markiert. Standardmäßig ist der beste Treffer derjenige, dessen Name die größte Ähnlichkeit zum Anfragestring `name_<usage>` hat. Das Standardverhalten kann durch die Parameter `w_<Kriterium><Modus>` geändert werden, indem Einfluss auf die Bewertung der Treffer genommen wird. Die Bewertung des Treffers, d.h. die Trefferqualität, lässt sich in der JSON-Ausgabe am Attribut `matchQualitiy` ablesen.

Außer der Gewichtung, die sich durch den Parameter Präfix `w_` für „weightning“ auszeichnet, ist eine Reduktion der Liste durch die Parameter `r_<Kriterium><Modus>` möglich. In diesem Fall wird das Präfix `r_` für „relevance“ verwendet. Die Reduktion entspricht einer Gewichtung und der anschließenden Reduktion durch den Parameter `prMinQu`.

Sowohl für Gewichtung als auch für Reduktion stehen zwei unterschiedliche Modi `<Modus>` zur Verfügung:

- `Al` („always“) - Das Bewertungskriterium `<Kriterium>` wird auf alle Treffer angewendet. Dabei wird die Trefferqualität aller Treffer, die nicht dem durch den Parameter angegebenen Kriterium entsprechen, auf 0 gesetzt. Allerdings nur, wenn es mindestens einen Treffer gibt, der dem Kriterium entspricht.

- `Am` („among best results“) - Es kommt häufig vor, dass mehrere Treffer mit der gleichen Trefferqualität bewertet werden. So kann es auch vorkommen, dass theoretisch mehrere Elemente beste Treffer im Sinne der Trefferqualität sind. Beim Standardbewertungsverhalten wird der als bester Treffer markierte Punkt zufällig unter diesen ausgewählt. Durch die Parameter `w_<Kriterium>Am` ist die Möglichkeit gegeben, die Auswahl durch Modifikation der Trefferqualität zu beeinflussen. Dazu wird die Trefferqualität für alle Treffer, die dem durch den Parameter angegebenen Kriterium entsprechen, um eins erhöht.

Der Modus wird dem Parameter als Suffix angehängt.

Folgende Kriterien `<Kriterium>` stehen zur Verfügung:

4.7.1 coord

Mit diesem Parameter wird die Koordinate des Zentums `<x>:<y>:<Koordinatenformat>` für die Methode zur Reduzierung bzw. Gewichtung `rMethod DISTANCEFROMCENTER` angegeben. Der Wert des Parameters setzt sich zusammen aus dem x, dem y und dem Koordinatenformat der Koordinate.

Hinweis: An diesen Prameter kann kein Suffix angehängt werden.

4.7.2 objPref

Der Wert des Parameters stellt eine Bitmaske dar. Werden Treffer mit dem durch die Bitmaske angegebenen Objekttypen gefunden, wird der beste Treffer unter diesen gesucht. Die Objekttypen sind dem Anhang Objekttypen zu entnehmen.

Hinweis: Beim Modus `<Modus> Al` werden zunächst Objekte mit den durch den Parameter angegebenen Typen gesucht. Sind keine Objekte in der Trefferliste vorhanden, wird auch unter den anderen Objekttypen nach Treffern gesucht

Beispiel: Sollen die besten Treffer auf Haltestellen und POIs beschränkt werden, wird der Parameter folgendermaßen angegeben: `w_objPref<Modus>=34` (2 + 32).

4.7.3 prefStModes

Befinden sich unter den besten Treffern Haltestellen, wird der zu selektierende Treffer unter den Haltestellen gesucht, die von dem durch den Parameter angegebenen Verkehrsmitteln bedient werden. Gibt es keine Haltestellen, die von den angegebenen Verkehrsmitteln bedient werden, werden auch andere Haltestellen ausgegeben.

Der Wert des Parameters ist eine Bitmaske, die sich aus den Identifikationsnummern der Verkehrsmittel herleitet. Um eine Eindeutigkeit der Verkehrsmittel zu erzielen, wird die Verkehrsmittel-ID als Exponent zur Zweierbasis verwendet. Eine Übersicht

über die Verkehrsmittel und ihrer Identifikationsnummern ist dem Anhang Verkehrsmitteltypen zu entnehmen.

Beispiel: Haltestellen, die von Zügen und S-Bahnen bedient werden. ID der Züge: 0 (2^0 = 1). ID der S-Bahnen: 1 (2^1 = 2). Daraus folgt: `w_prefStModesAm=3` (entspricht 1 + 2).

Hinweis: Diesen Parameter gibt es nur im Modus `<Modus> Am`. Er Parameter kommt nur dann zum Einsatz, wenn sich unter den besten Treffern Haltestellen befinden.

4.7.4 regPref

Der Wert des Parameters gibt die ID einer Region an. Werden Treffer in der angegebenen Region gefunden, wird der beste Treffer unter diesen gesucht. Die Angabe der Region erfolgt analog zum Parameter `regionID_<usage>` (siehe Suchkriterien).

Hinweis: Ist der Modus `<Modus> Al`, werden andere Regionen nur berücksichtigt, wenn in der angegebenen Region keine Treffer vorhanden sind.

4.7.5 relInf

Durch diesen Parameter fließt die Relevanz des Treffers in die Berechnung seiner Trefferqualität mit ein. Sein Wert ist ganzzahlig und liegt zwischen 0 und 100. Er beschreibt, mit wie viel Prozent die Relevanz in die Trefferqualität einfließt. Standardmäßig wird die Relevanz nicht berücksichtigt, was einem Wert 0 entspricht. Soll ausschließlich die Relevanz für die Trefferqualität berücksichtigt werden, muss der Wert 100 sein.

Die Relevanz der Treffer wird aus einer externen Datei, die automatisch oder händisch erstellt sein kann, eingelesen. Der Pfad zu dieser Datei muss in der Konfigurationsdatei des EFALocationServers angegeben werden.

Hinweis: Der Standardwert dieses Parameters muss nicht unbedingt 0 sein. Er kann in der Sektion in der Konfigurationsdatei des EFALocationServers abweichend definiert sein.

4.7.6 rMethod

Mit diesem Parameter wird die für die Reduzierung bzw. Gewichtung verwendete Methode angegeben. Folgende Methoden stehen zur Verfügung:

- `DISTANCEFROMCENTER` - Distanz zum Zentrum. Die Distanz wir mit `coord` angegeben.
- `BESTCOVER` - Größtmögliche Übereinstimmung von Wörterbuchtext und Benutzereingabe.

- `PREFIXCOUNT` - Anzahl gleicher Präfixbuchstaben von Wörterbuchtext und Benutzereingabe.
- `PREFIXRATIO` - Anteil gleicher Präfixbuchstaben von Wörterbuchtext und Benutzereingabe.

4.7.7 Anwendungsbeispiele

Das Präfix `w_` steht für „weighting“, da durch Kombination verschiedener Parameter eine Gewichtung nach verschiedenen Kriterien vorgenommen werden kann. Das heißt, dass eine beliebige Anzahl dieser Parameter verwendet werden kann; inklusive doppelter Verwendung gleicher Parameter mit unterschiedlichem Wert. Gleiches gilt für die Reduzierung der Trefferlisten mittels der Parameter mit Präfix `r_`. Die Parameter werden gemäß dem Kriterium `<Kriterium>` in folgender Reihenfolge ausgeführt:

1. Objekttypen (`objPref`)
2. Regionen (`regPref`)
3. Verkehrsmittel (`prefStModes`)

Innerhalb eines Kriteriums `<Kriterium>` werden die Parameter gemäß ihrem Modus in folgender Reihenfolge ausgeführt:

1. always (`Al`)
2. among best results (`Am`)

Werden Parameter mehrfach verwendet, erfolgt die Abarbeitung hierarchisch. Der erste Parameter hat die höchste Priorität, der letzte die niedrigste.

Beispiel zur Hierarchie: Werden die Parameter `w_regPrefAl=1` und `w_regPrefAl=2` (in dieser Reihenfolge) verwendet, so wird zunächst geprüft, ob sich unter den Treffern welche befinden, die in der Region 1 liegen. Ist dies der Fall kommt der Parameter `w_regPrefAl2` nicht mehr zum Einsatz. Die Treffer aus der Region 1 werden hoch bewertet und für alle anderen Treffer wird die Trefferqualität auf 0 gesetzt. Ist jedoch kein Treffer aus der Region 1 vorhanden, wird geprüft, ob Treffer aus der Region 2 vorhanden sind. Ist dies der Fall, werden die Treffer aus der Region 2 hoch bewertet und für alle anderen Treffer die Trefferqualität mit 0 angegeben. Siehe hierzu auch das Beispiel zur Kombination von Treffern weiter unten.

Anmerkung: Die Anzahl der Treffer wird durch die Gewichtung nicht beeinflusst. Es wird lediglich die Trefferqualität modifiziert, wodurch die Selektion des besten Treffers beeinflusst werden kann. Eine Reduktion der Liste kann durch die Parameter `anyMaxSizeHitList` oder `prMinQu=1` erreicht werden (siehe Optionale Parameter zur Optimierung der Punktsuche).

Beispiel zur Kombination von Kriterien: Es werden folgende Parameter verwendet:

```
w_regPrefAm=1    // 3. Kriterium
w_objPrefAl=2    // 1. Kriterium
w_objPrefAl=12   // 2. Kritierum (12=8+4)
```

Die Haltestellenliste bestehe aus:

- Haltestelle "A" mit Trefferqualität 101 in der Region mit der ID 1
- Haltestelle "B" mit Trefferqualität 102 in der Region mit der ID 1
- POI "C" mit Trefferqualität 103 in der Region mit der ID 1
- Haltestelle "D" mit Trefferqualität 102 in der Region mit der ID 2

Als bester Treffer wird die Haltestelle "B" durch die Property `isBest=1` gekennzeichnet, da:

1. Kriterium: Die Trefferqualität von POI "C" wird auf 0 gesetzt, da es sich nicht um eine Haltestelle (Objekttyp 2) handelt. Die modifizierten Trefferqualitäten sind wie folgt:
    - Haltestelle "A" mit Trefferqualität 101 in der Region mit der ID 1
    - Haltestelle "B" mit Trefferqualität 102 in der Region mit der ID 1
    - POI "C" mit Trefferqualität **0** in der Region mit der ID 1
    - Haltestelle "D" mit Trefferqualität 102 in der Region mit der ID 2
2. Kriterium: Kommt nicht zum Einsatz, da bereits ein anderes Kriterium für Objekttypen (1. Kriterium) zuvor angewendet wurde. Die Trefferqualitäten ändern sich somit nicht.
3. Kriterium: Bezieht aufgrund des Modus `<Modus> Am` ausschließlich auf die besten Treffer. Dies sind Haltestelle "B" und "D" mit der Trefferqualität 102. Da sich Haltestelle B in der Region mit der ID 1 befindet, wird die Trefferqualität auf 103 erhöht. Die Trefferqualitäten sind nun wie folgt:
    - Haltestelle "A" mit Trefferqualität 101 in der Region mit der ID 1
    - Haltestelle "B" mit Trefferqualität **103** in der Region mit der ID 1
    - POI "C" mit Trefferqualität 0 in der Region mit der ID 1
    - Haltestelle "D" mit Trefferqualität 102 in der Region mit der ID 2

    Damit kann Haltestelle "B" eindeutig als bester Treffer gekennzeichnet werden.

4.8 Listenauswahl

Ergibt ein Punkt kein Volltreffer, wird ein Array `locations` ausgegeben. Der beste Treffer ist durch die Property `isBest=1` markiert (siehe hierzu auch Beeinflussung der Trefferqualität und Reduktion der Trefferliste). Zur Auswahl eines Punktes wird der Wert der Property `id` durch den Parameter `name_<usage>` übergeben.

4.9 Wahl einer nahegelegenen Haltestelle

Einige Anfragearten, wie zum Beispiel der StopTimetable-Request, benötigen eine eindeutig identifizierte Haltestelle. Diese kann mittels des StopFinder-Requests ermittelt werden. Wird jedoch nach einer Adresse, einem wichtigen Punkt oder einer Koordinate gesucht, muss anschließend eine nahegelegene Haltestelle ausgewählt werden (siehe Abbildung 5). Auch komplexere Haltestellen wie ein Hauptbahnhof können durch mehrere Haltestellen modelliert sein. Es werden maximal zehn nahegelegene Haltestellen ermittelt. Dies sind die am nächsten gelegenen Haltestellen.

Sind nahegelegene Haltestellen vorhanden, werden diese als `assignedStops` JSON-Antwort ausgegeben. Die Wahl einer nahegelegenen Haltstelle erfolgt über den Parameter `name_<usage>`. Als Wert wird der Wert der Property `id` übergeben.

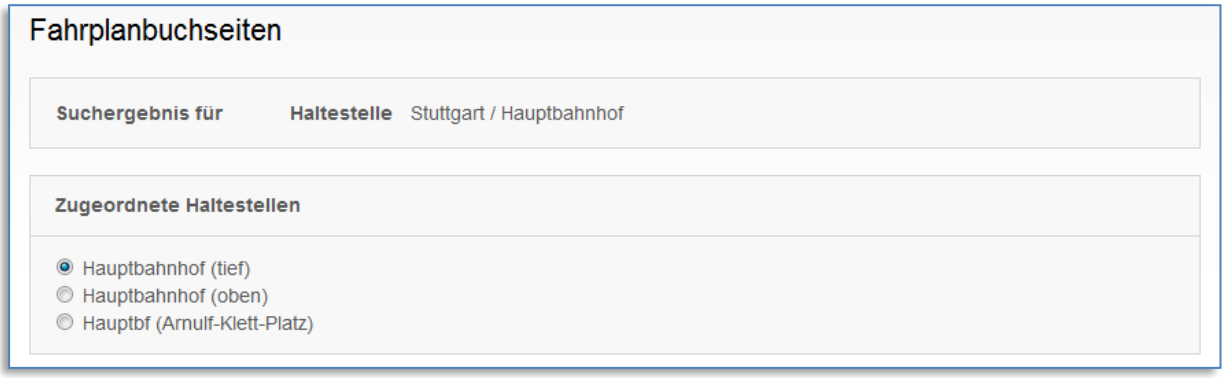

Abbildung 5 - Wahl einer nahegelegenen Haltestelle am Beispiel Fahrplanbuchseiten

4.10 Verifizierten Punkt in einen anderen Request übernehmen

Wie ein mit dem StopFinder-Request ermittelter Punkt an andere Requests als Volltreffer übergeben werden kann, wird im Kapitel Eingabe einer ID oder Koordinate beschrieben.

5 ServingLines-Request

Im Folgenden wird die Liniensuche (ServingLines-Request) erläutert. Die Liniensuche ermöglicht die Suche einer Linie über die Haltestellensuche oder die direkte Liniensuche.

5.1 Ansteuerung der Liniensuche

Die Liniensuche wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_SERVINGLINES_REQUEST?HTTP-Parameter
```

5.2 Hinweis zur Eingabe von Punkten

Bei der Eingabe eines Punktes wird als Erweiterung `<usage>` der Wert `sl` angegeben.

5.3 Obligatorische Parameter

Neben den hier beschriebenen Parametern sind entweder die Parameter für die Suche einer Linie über die Haltestellensuche oder die Parameter für die direkte Liniensuche notwendig.

5.3.1 mode

Suchmodus. Der Wert `odv` aktiviert die Liniensuche über die Haltestellensuche; der Wert `line` aktiviert die direkte Liniensuche.

5.3.2 lineReqType

Typ der angefragten Linien. Folgende Werte sind möglich:

- 0 - alle Linien
- 2 - nur Linien mit Aushangfahrplan (STT)
- 4 - nur Linien mit Fahrplanbuchseiten (TTB)
- 8 - nur Linien mit Routen-Karten (ROB)
- 16 - nur Linien mit ZOB Haltestellen-Fahrplan (StationTT)

Dabei handelt es sich um eine Bitmaske. Zur Wahl mehrere Typen können die Werte addiert werden.

5.4 Liniesuche über Haltestellensuche

Soll eine bedienende Linie einer Haltestelle gewählt werden, muss der Parameter `mode` den Wert `odv` haben. Außerdem sind die obligatorischen Parameter zur Eingabe eines Punktes notwendig.

5.5 Direkte Liniensuche

Soll die Linie direkt anhand eines Freitextes gesucht werden, muss der Parameter `mode` den Wert `line` haben. Außerdem ist folgender Parameter notwendig:

5.5.1 lineName

Dieser Parameter enthält den Namen der Linie der als Suchtext eingegeben wird.

6 Trip-Request

Im Folgenden wird die Ansteuerung der Fahrtanfrage (Trip-Request) erläutert. Die Fahrtanfrage berechnet Fahrten zu einem angegebenen Datum und einer angegebenen Uhrzeit. Mit An-/Abkennung, Start- und Zielpunkt sowie Verbindungsoptionen. Die als optionalen Verbindungsoptionen bieten umfassende Möglichkeiten die Fahrtergebnisse nach eigenen Bedürfnissen zu optimieren.

6.1 Ansteuerung der Fahrtanfrage

Die Fahrtanfrage wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_TRIP_REQUEST2?HTTP-Parameter
```

6.2 Hinweis zur Eingabe von Punkten

Bei der Eingabe eines Punktes werden als Erweiterung `<usage>` folgende Werte angegeben:

- `origin` – Startpunkt
- `destination` – Zielpunkt
- `via` – Zwischenhalt

6.3 Obligatorische Parameter

Für die Fahrtanfrage ist mindestens die Angabe eines Start- und eines Zielpunktes durch die im Kapitel Eingabe eines Punktes beschriebenen HTTP-Parameter notwendig. Außerdem wird folgender Parameter benötigt:

6.3.1 tripReductionMacro=1

Bei diesem Parameter handelt sich um ein HTTP-Parameter-Makro. Es fasst verschiedene HTTP-Parameter zusammen, die für die Verwendung der JSON-Schnittstelle notwendig sind.

6.4 Optionale Parameter

6.4.1 calcNumberOfTrips

Gibt die Anzahl der Fahrten mit Öffentlichen Verkehrsmitteln an. Standardmäßig werden vier Fahrten berechnet. Gibt es alternative Fahrten, kann die angegebene Anzahl überschritten werden. Alternative Fahrten sind an der Property `isAlternative=true` zu erkennen.

6.4.2 calcOneDirection = 1

Standardmäßig wird eine der Fahrten vor der gewählten Abfahrtszeit bzw. nach der gewählten Ankunftszeit ausgegeben. Dieser Parameter unterdrückt das Standardverhalten.

6.4.3 itdTripDateTimeDepArr

Der Wert des Parameters bestimmt, ob sich Datum und Uhrzeit auf die Abfahrt (`dep`) oder die Ankunft (`arr`) beziehen. Default ist `dep`.

6.4.4 useRealtime = 1

Aktiviert die Berücksichtigung der Echtzeitüberwachung (siehe Abbildung 6).

Abbildung 6 - Fahrtauskunft mit Echtzeitüberwachung

Dies bewirkt, dass in der JSON-Ausgabe (siehe Abbildung 7) für jeden echtzeitüberwachten Teilweg ein zusätzliches Element `isRealtimeControlled=true`, sowie der Echtzeitstatus `realtimeStatus` ausgegeben werden. Für einen Punkt (zum Beispiel `origin`) wird dann jeweils die Soll- und die Ist-Zeit als `departureTimePlanned` und `departureTimeEstimated` ausgegeben.

```
- journeys: [
    - {
          rating: 0,
          isAdditional: false,
          interchanges: 1,
        - legs: [
            - {
                  duration: 540,
                  isRealtimeControlled: true,
                - realtimeStatus: [
                      "MONITORED"
                  ],
                - origin: {
                      isGlobalId: true,
                      id: "de:08111:6052:1:1",
                      name: "Schwabstraße",
                      disassembledName: "Gleis 1",
                      type: "platform",
                      pointType: "TRACK",
                    - coord: [
                          48.77019,
                          9.15647
                      ],
                      niveau: -300,
                    - parent: {
                          isGlobalId: true,
                          id: "de:08111:6052",
                          name: "Schwabstraße",
                          type: "stop",
                        - parent: {
                              id: "placeID:8111000:51",
                              name: "Stuttgart",
                              type: "locality"
                          },
                        - properties: {
                              stopId: "5006052"
                          }
                      },
                    - productClasses: [
                          1,
                          5
                      ],
                      departureTimePlanned: "2021-06-01T12:00:00Z",
                      departureTimeEstimated: "2021-06-01T12:01:00Z",
                    - properties: {
                        - downloads: [
```


Abbildung 7 - Echtzeitüberwachung in der JSON-Antwort des Trip-Requests

6.5 Optionaler Zwischenhalt

Der Zwischenhalt hat die Parameterergänzung `via` und kann ausschließlich beim Trip-Request oder PS-Request verwendet werden. Ausschließlich auf den Zwischenhalt beziehen sich folgende optionale Parameter:

6.5.1 dwellTime_<usage>

Aufenthaltsdauer an einem Zwischenhalt (Via-Punkt). Das Präfix `<usage>` bezieht sich in diesem Fall ausschließlich auf Zwischenhalte; nicht auf den Start- oder

Zielpunkt. Die Aufenthaltsdauer wird in Minuten angegeben.

Hinweis: Wird eine Aufenthaltsdauer von 0 Minuten gewählt, werden Adressen und wichtige Punkte als Zwischenhalt nicht berücksichtigt. Dieses Verhalten kann durch den Parameter `AlwaysDwellTimeJourneCalculation=1` in der Konfigurationsdatei des EFAControllers geändert werden.

6.6 Verkehrsmittelausschluss und Verkehrsmitteleinschluss

Eine Übersicht über die Standard-Verkehrsmittelbelegung ist dem Anhang Verkehrsmitteltypen zu entnehmen.

6.6.1 excludedMeans

Dieser Parameter kann auf zwei Arten verwendet werden:

Der Wert `checkbox` aktiviert das Ausschließen von Verkehrsmitteltypen. Die auszuschließenden Verkehrsmittel werden dann mit dem Parameter `exclMOT_<ID>` angegeben. Standardmäßig sind alle Verkehrsmittel aktiviert.

Alternativ kann die ID des auszuschließenden Verkehrsmittels als Wert angegeben werden. Der Parameter kann mehrfach verwendet werden.

6.6.2 exclMOT_<ID> = 1

Dieser Parameter bewirkt, dass der Verkehrsmitteltyp mit der als Wert übergebenen ID ausgeschlossen wird. Zum Ausschließen mehrerer Verkehrsmittel kann der Parameter mehrfach verwendet werden.

Achtung: Das gewählte Verkehrsmittel wird ausgeschlossen, sobald der entsprechende Parameter übergeben wird. Auch wenn dieser keinen Wert oder die Werte 0 bzw. `false` hat.

6.6.3 includedMeans

Dieser Parameter kann auf zwei Arten verwendet werden:

Der Wert `checkbox` aktiviert das Einschließen von Verkehrsmitteltypen. Die einzuschließenden Verkehrsmittel werden dann mit dem Parameter `inclMOT_<ID>` angegeben. Standardmäßig sind alle Verkehrsmittel deaktiviert.

Alternativ kann die ID des einzuschließenden Verkehrsmittels als Wert angegeben werden. Der Parameter kann mehrfach verwendet werden.

6.6.4 inclMOT_<ID> = 1

Dieser Parameter bewirkt dass der Verkehrsmitteltyp mit der als Wert übergebenen ID eingeschlossen wird. Zum Einschließen mehrerer Verkehrsmittel kann der Parameter mehrfach verwendet werden.

Achtung: Das gewählte Verkehrsmittel wird eingeschlossen, sobald der entsprechende Parameter übergeben wird. Auch wenn dieser keinen Wert oder die Werte 0 bzw. `false` hat.

6.7 Mobilitätseinschränkungen

Die unter den Mobilitätseinschränkungen zusammengefassten Parameter ermöglichen es auf die Bedürfnisse von Menschen mit eingeschränkter Mobilität einzugehen. Diese Parameter können nur verwendet werden, wenn Treppen, Rolltreppen, etc. in den Daten erfasst sind.

6.7.1 imparedOptionsActive=1

Aktiviert die Parameter für eingeschränkte Mobilität.

6.7.2 lowPlatformVhcl

Ist dieser Parameter aktiviert, werden ausschließlich Niederflurfahrzeuge berücksichtigt. Es ist nicht notwendig dem Parameter einen bestimmten Wert zuzuweisen. Sobald der Parameter übergeben wird, ist diese Option aktiv.

6.7.3 noElevators

Wird dieser Parameter übergeben, werden ausschließlich Fahrten mit Umstiegen ohne Verwendung von Aufzügen beauskunftet. Es ist nicht notwendig dem Parameter einen bestimmten Wert zuzuweisen. Sobald der Parameter übergeben wird, ist diese Option aktiv.

6.7.4 noEscalators

Wird dieser Parameter übergeben, werden ausschließlich Fahrten mit Umstiegen, die nicht über Rolltreppen erfolgen, beauskunftet. Es ist nicht notwendig dem Parameter einen bestimmten Wert zuzuweisen. Sobald der Parameter übergeben wird, ist diese Option aktiv.

6.7.5 noSolidStairs

Wird dieser Parameter übergeben, werden ausschließlich Fahrten mit Umstiegen, die nicht über normale Treppen erfolgen, beauskunftet. Es ist nicht notwendig dem Parameter einen bestimmten Wert zuzuweisen. Sobald der Parameter übergeben wird, ist diese Option aktiv.

6.7.6 wheelchair

Wird dieser Parameter übergeben, werden ausschließlich rollstuhlgerechte Fahrzeuge bei der Auskunft berücksichtigt. Rollstuhlgerechte Fahrzeuge haben einen Hublift bzw. eine Rampe oder ermöglichen den niveaugleichen Einstieg. Es ist nicht notwendig dem Parameter einen bestimmten Wert zuzuweisen. Sobald der Parameter übergeben wird, ist diese Option aktiv.

6.8 Weitere Optionen für den öffentlichen Verkehr

6.8.1 ptOptionsActive = 1

Sollen die Optionen für den öffentlichen Verkehr verwendet werden, müssen diese zunächst mit diesem Parameter aktiviert werden. Standardmäßig sind die Optionen für den öffentlichen Verkehr deaktiviert.

6.8.2 changeSpeed

Modifiziert die für Umstiegsfußwege benötigte Zeit. Folgende Werte sind möglich:

- `fast` (entspricht 50)
- `normal` (default, entspricht 100)
- `slow` (entspricht 200)

Der Wert stellt einen Faktor dar, mit dem die in der Umstiegsmatrix definierte Zeit nach folgender Formel modifiziert wird:

```
Umstiegszeit [min] = (Zeit aus der Umstiegsmatrix [min] * Wert
des Parameters) / 100
```

Falls sich dieser Parameter auch auf den Weg vom Startpunkt zur ersten Haltestelle und von der letzten Haltestelle zum Zielpunkt beziehen soll, muss sichergestellt sein, dass die Optionen für den Individualtransport durch den Parameter `itOptionsActive=1` aktiviert sind (siehe Kapitel Optionen für den Individual-transport).

Für den Individualtransport sind die Alias-Namen wie folgt definiert:

- `fast` (entspricht 120)
- `normal` (default, entspricht 100)
- `slow` (entspricht 80)

Der Wert stellt einen Faktor dar, mit dem die für den EFAITKernel konfigurierte Standardgeschwindigkeit nach folgender Formel modifiziert wird:

```
Geschwindigkeit [km/h] = (Standardgeschwindigkeit [km/h] *
Wert des Parameters) / 100
```

Die Standardgeschwindigkeit beträgt, falls nicht anders konfiguriert, 4 km/h.

In der Regel wird für einen Umstieg mehr Zeit benötigt als für den Individualtransport, da der Passagier Treppen überwinden und sich seinen Weg durch andere Passa-giere bahnen muss. Daher sind die Werte für die Aliasnamen *fast* und *slow* für den Individualtransport nicht nur gegenüber dem Umstiegsfußweg vertauscht, sondern auch erhöht.

6.8.3 lineRestriction

Mit diesem Parameter können bestimmte Linien ausgeschlossen werden. Er kann folgende Werte annehmen:

- `400` (alle Linien, Default)
- `401` (alle Linien außer ICE)
- `402` (Verbund ohne Aufschlag)
- `403` (Verbund und Nahverkehr)

6.8.4 maxChanges

Maximale Anzahl von Umstiegen einer Fahrt. Fahrten mit mehr als den angegebenen Umstiegen werden bei der Fahrtberechnung verworfen. Standardmäßig sind 9 Umstiege als maximale Anzahl definiert, dieser Wert kann jedoch in der Konfigurationsdatei des EFA Controllers angepasst werden. Mögliche Werte sind:

- `0` (Direktverbindung)
- `1`
- `2`
- `9` (bzw. das in der Konfigurationdatei des EFA Controllers definierte Maximum, Default)

6.8.5 routeType

Gibt an, nach welchem Kriterium die Fahrtauskünfte optimiert werden sollen:

- `leastinterchange` (Verbindungen mit möglichst wenigen Umstiegen)
- `lasttime` (Möglichst schnelle Verbindungen, Default)
- `leastwalking` (Verbindungen mit möglichst wenigen Fußwegen)

6.8.6 useProxFootSearch = 1

Berücksichtigung von nahegelegenen Haltestellen.
Zur Unterscheidung der Start- und Zielhaltestelle kann der Parametername durch die Erweiterung `Orig` bzw. `Dest` ergänzt werden.

6.9 Optionen für den Individualtransport

6.9.1 itOptionsActive = 1

Sollen die Optionen für den Individualtransport verwendet werden, müssen diese zunächst mit diesem Parameter aktiviert werden. Standardmäßig sind die Optionen für den Individualtransport ausgeschaltet.

6.9.2 changeSpeed

Dieser Parameter gilt nicht nur für den Individualtransport, sondern auch für den öffentlichen Verkehr. Er ist im Kapitel Weitere Optionen für den öffentlichen Verkehr beschrieben.

6.9.3 trITMOT

Mit diesem Parameter wird der Transportmitteltyp für den Weg vom Startpunkt zur Starthaltestelle und von der Zielhaltestelle zum Zielpunkt gewählt. Folgende Werte sind möglich:

- `100` (Fußweg)
- `101` (Bike & Ride)
- `102` (Fahrradmitnahme)
- `103` (Kiss & Ride)
- `104` (Park & Ride)
- `105` (Taxi)

Alternativ können die Parameter `trITArrMOT` und `trITDepMOT` verwendet werden. Mit ihnen kann man den Transportmitteltyp für Start- und Zielpunkt separat setzen.

6.9.4 trITMOTvalue

Der Wert des Parameters gibt die maximale Zeit an, die der Weg vom Startpunkt zur Starthaltestelle sowie der Weg von der Zielhaltestelle zum Zielpunkt dauern soll. Die Zeit wird in Minuten angegeben. Voreingestellt sind 10 Minuten, falls nichts anderes in der Konfigurationsdatei des EFA Controllers bzw. des EFA Servers definiert ist.

Alternativ können die Parameter `trITArrMOTvalue` und `trITDepMOTvalue` verwendet werden. Mit ihnen kann man die maximale Zeit für Start- und Zielpunkt separat setzen. Sollen unterschiedliche maximale Zeiten für die einzelnen Transportmitteltypen angegeben werden, kommen die Parameter `trITMOTvalue<Transporttmittelyp>` bzw. `trITArrMOTvalue<Transportmitteltyp>` und `trITDepMOTvalue<Transportmitteltyp>` zum Einsatz. Die verfügbaren Transportmitteltypen sind für den Parameter `trITMOT` aufgelistet.

6.10 Fahrrad

Erfolgt ein Teil oder die komplette Fahrtstrecke mit dem Fahrrad, sind einige zusätzliche Parameter notwendig. Optional kann das Fahrrad geparkt werden (siehe Parken des Fahrrads).

Beispiel:

ÖPNV-Verbindungen und reine Fahrradfahrt als Alternative:

```
http://server:port/virtuellesVerzeichnis/XML_TRIP_REQUEST2?
name_origin=de:05513:5613&type_origin=any&name_destination=poi
```

```
ID:61032:5513000:-1:Schalke-Museum:Gelsenkirchen:Schalke-
Museum:ANY:POI:786740:5280177:MRCV:nrw&type_destination=any&ca
lcBicycleMacro=on&maxTimeBicycle=120
```

Fahrradfahrt vom Startpunkt zur ersten Haltestelle:

```
http://server:port/virtuellesVerzeichnis/XML_TRIP_REQUEST2?
name_origin=poiID:61032:5513000:-1:Schalke-
Museum:Gelsenkirchen:Schalke-
Museum:ANY:POI:786740:5280177:MRCV:nrw&type_origin=any&name_de
stination=de:05513:5613&type_destination=any&calcBicycleMacro=
on&maxTimeBicycle=20&std3_bikeSettings=toFirstStop&parkingMaxD
istance=500
```

6.10.1 Obligatorische Parameter

6.10.1.1.1 calcBicycleMacro = on

HTTP-Parameter-Makro zur Anfrage von Fahrradrouten.

6.10.1.2 Makro

In Abhängigkeit davon, ob eine Bike & Ride-Verbindung, eine Fahrt vom Startpunkt zur ersten Haltestelle, eine Fahrt von der letzten Haltestelle zum Zielpunkt oder eine Fahrt mit dem ÖPNV, bei der das Fahrrad mitgenommen werden soll, angefragt werden soll, muss einer der folgenden HTTP-Parameter-Makros gewählt werden. In den Makros sind alle obligatorischen Parameter für die gewählte Art der Anfrage zusammengefasst. Für eine reine Fahrradfahrt ist kein weiteres Makro notwendig

6.10.1.2.1 brRoutingMacro = true

Makro zur Anfrage von Bike & Ride-Verbindungen.

6.10.1.2.2 std3_bikeSettings = takealong

Makro zur Anfrage von ÖPNV-Verbindungen, die eine Fahrradmitnahme ermöglichen.

6.10.1.2.3 std3_bikeSettings = toFirstStop

Makro zur Anfrage einer ÖPNV-Verbindung, bei der für den Weg vom Start zur ersten Haltestelle das Fahrrad verwendet wird.

6.10.1.2.4 std3_bikeSettings = fromLastStop

Makro zur Anfrage einer ÖPNV-Verbindung, bei der für den Weg von der letzten Haltestelle zum Ziel das Fahrrad verwendet wird.

6.10.1.3 cycleSpeed

Mit dem Wert dieses Parameters wird die Geschwindigkeit, die bei der Fahrtberechnung zugrunde gelegt wird, angegeben. Die Einheit ist km/h.

6.10.1.4 maxTimeBicycle

Dieser Parameter gibt die maximale Zeit an, die die Fahrradfahrt dauern soll. Die Zeit wird in Minuten angegeben.

6.10.2 Parken des Fahrrads

Soll das Fahrrad geparkt werden, muss das HTTP-Parameter-Makro `brRoutingMacro=true` mitgeschickt werden; unabhängig davon, welche Rolle die Fahrradfahrt spielt. Das Parken von Fahrrädern ist bei Bike & Ride sowie bei einer Fahrradfahrt vom Start zur ersten Haltestelle oder bei einer Fahrradfahrt von der letzten Haltestelle zum Ziel möglich.

6.10.2.1 parkingMaxDistance

Suchradius in Metern, für eine Umkreissuche nach Parkobjekten für Fahrräder, zum Beispiel Fahrradboxen.

6.11 Park & Ride

Für die Anfrage von Fahrtoptionen mit Park & Ride werden zusätzliche Parameter benötigt.

Beispiel:

```
http://server:port/virtuellesVerzeichnis/XML_TRIP_REQUEST2?
name_origin=poiID:61032:5513000:-1:Schalke-
Museum:Gelsenkirchen:Schalke-
Museum:ANY:POI:786740:5280177:MRCV:nrw&type_origin=any&name_de
stination=de:05513:5613&type_destination=any&prRoutingMacro=tr
ue
```

6.11.1 Obligatorische Parameter

6.11.1.1 prRoutingMacro=true

In diesem HTTP-Parameter-Makro sind alle obligatorischen Parameter für die gewählte Art der Anfrage zusammengefasst.

6.11.2 Optionale Parameter

6.11.2.1 maxLengthCar

Maximale Wegstrecke in Metern, die mit dem PKW zurückgelegt werden soll.

6.11.2.2 maxTimeCar

Maximale Zeit in Minuten, die im PKW verbracht werden soll.

7 PS-Request

Im Folgenden wird die Ansteuerung zum Erstellen eines Pendlerfahrplans (PS-Request), auch „Persönlicher Fahrplan“ genannt, erläutert. Die Abkürzung „PS“ steht für „Personal Schedule“. Der PS-Request hat große Ähnlichkeit zum Trip-Request. Im Gegensatz zu diesem werden jedoch Fahrten innerhalb eines Zeitintervalls ermittelt. Daher unterscheidet sich die Eingabe von Datum und Uhrzeit. Auf Wunsch können Zeitintervalle für eine einfache Fahrt oder für eine Hin- und Rückfahrt angegeben werden.

7.1 Ansteuerung der Pendlerfahrplans

Der Pendlerfahrplan wird über HTTP-Parameter angesteuert. Die Antwort liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_PS_REQUEST2?HTTP-Parameter
```

Zusätzlich kann ein base64 kodierter Stream angefordert werden.

7.2 Hinweis zur Eingabe von Punkten

Bei der Eingabe eines Punktes werden als Erweiterung `<usage>` folgende Werte angegeben:

- `origin` – Startpunkt
- `destination` – Zielpunkt
- `via` – Zwischenhalt (standardmäßig deaktiviert)

7.3 Obligatoriche Parameter

Der Pendlerfahrplan benötigt mindestens die Angabe eines Start- und eines Zielpunktes durch die im Kapitel Eingabe eines Punktes beschriebenen HTTP-Parameter sowie mindestens ein Zeitintervall, für das die Fahrten berechnet werden sollen. Die Angabe des Zeitintervalls ist in Kapitel Datum und Uhrzeit für den Pendlerfahrplan beschrieben. Außerdem:

7.3.1 psParamWeekday

Der Parameter bestimmt für welchen Wochentagtagstypen Fahrten berechnet werden sollen. Mögliche Werte sind:

- *6:*SAMSTAG - Es werden Fahrten berechnet, die samstags gültig sind.
- *8:MO-FR* - Es werden Fahrten berechnet, die montags bis freitags gültig sind.
- *10:*SONNTAG - Es werden Fahrten berechnet, die sonn- und feiertags gültig sind.

Alternativ kann ein Referenzdatum durch den Parameter `psParamSampleDate` angegeben werden.

7.3.2 psParamSampleDate

Der Wert gibt einen Stichtag an, der die in den Daten vorhandenen übersteuert. Er setzt sich aus der vierstelligen Jahreszahl, gefolgt von der zweistelligen Angabe des Monats und der zweistelligen Angabe des Tages, zusammen.

In der Regel wird dieser Parameter nicht benötigt. Er ist eine Alternative zur Angabe des Wochentagstypes mit dem Parameter `psParamWeekday`.

7.3.3 useAllSampleDates = 1

Standardmäßig werden für die Berechnung eines Persönlichen Fahrplans nur die gewählten Stichtage verwendet. Dieses Verhalten wird durch diesen Parameter geändert. Er bewirkt, dass alle Stichtage berücksichtigt werden, unabhängig davon, ob sie zum gewählten Wochentagstypen passen.

7.4 Datum und Uhrzeit für den Pendlerfahrplan

Die Berechnung des Pendlerfahrplans erfolgt aufgrund von Stichtagen. Diese sind vordefiniert, so dass statt einem Datum ein Wochentagstyp gewählt werden muss. Zusätzlich wird ein Uhrzeitintervall benötigt. Start und Ende des Intervalls für die Hinfahrt und des optionalen Intervalls für die Rückfahrt werden anhand des Suffixes `_<usagePS>` unterschieden. Folgende Werte sind möglich:

- `arrFrom` - Der Wert des Parameters bezieht sich auf den Beginn des Intervalls für die Rückfahrt, d.h. die frühstmögliche Hinfahrt.
- `arrTo` - Der Wert des Parameters bezieht sich auf das Ende des Intervalls für die Rückfahrt, d.h. die spätestmögliche Hinfahrt.
- `depFrom` - Der Wert des Parameters bezieht sich auf den Beginn des Intervalls für die Rückfahrt, d.h. die frühstmögliche Hinfahrt.
- `depTo` - Der Wert des Parameters bezieht sich auf das Ende des Intervalls für die Rückfahrt, d.h. die spätestmögliche Hinfahrt.

Hinweis: Die maximale Größe des Intervalls und die Vorbelegung der Zeiten sind in der Konfigurationsdatei des EFAControllers festgelegt. Die Intervallgröße kann durch den HTTP-Parameter `psParamMaxTimeHours` überschrieben werden.

7.4.1 itdTime_<usagePS>

Eingabe der Uhrzeit. Der Wert kann in einem der folgenden Formate angegeben werden:

- `HHMM`
- `HH:MM`
- `HH.MM`
- `MM`
- `HHMMa` (anglo-amerikanisches Format „am“)
- `HHMMh` (24-Stunden Format)
- `HHMMp` (anglo-amerikanisches Format „pm“)

7.4.2 itdTimeAMPM_<usagePS>

Bei der anglo-amerikanische Zeitangabe im 12-Stunden-Format unterscheidet der Wert mit *am* (Default) oder *pm* den Vor- und Nachmittag.

7.4.3 itdTimeHour_<usagePS>

Eingabe der Stunden. Der Wert gibt in ein- oder zweistelliger Form die Stunden an.

7.4.4 itdTimeMinute_<usagePS>

Eingabe der Minuten. Der Wert gibt in ein- oder zweistelliger Form die Minuten an.

7.5 Optionale Parameter

Es können die meisten der für den Trip-Request beschriebenen optionalen Parameter verwendet werden. Außerdem:

7.5.1 base64 = 1

Fordert das base64 kodierte Printprodukt an.

7.5.2 psParamOneWay = 1

Standardmäßig werden sowohl Hin- als auch Rückfahrten berechnet. Wird dieser Parameter aktiviert wird der Fahrplan nur für eine Richtung berechnet.

7.5.3 psParamGroupID

Gelegentlich sind verschiedene Fahrplandaten mit unterschiedlichen Gültigkeitsperioden verfügbar. Dies ist insbesondere in der Zeit um den Fahrplanwechsel herum der Fall. Da der Persönliche Fahrplan sich nicht auf ein konkretes Datum bezieht, kann im Fall von mehreren Fahrplandaten der Satz mit der gewünschten Fahrplanperiode durch diesen Parameter ausgewählt werden.

Die verfügbaren Fahrplanperioden können mit dem SystemInfo-Request angefragt werden. Die Property `id` wird zur Auswahl der gewünschten Fahrplanperiode als Wert des Parameters übergeben. Als Beschreibungstext zur Anwendergesteuerten Auswahl steht die Beschreibung `description` zur Verfügung.

7.5.4 psParamMaxTimeHours

Der Parameter gibt die maximal zulässige Größe für das Intervall an, in dem Fahrten für den Persönlichen Fahrplan berechnet werden sollen. Wertebereich: [0..24]. Liegen die Uhrzeiten für den Start und das Ende des Intervalls weiter auseinander als durch den Parameter angegeben, wird eine Fehlermeldung ausgegeben.

7.5.5 useAltOdv = 1

Aktiviert die Verwendung eines Zwischenhalts, d.h. einer Haltestelle, an der bei den ausgegebenen Fahrten gehalten wird. Zwischenhalte mit `name_via` und `type_via`

angegeben. Außerdem können die in Kapitel Optionaler Zwischenhalt für den Trip-Request beschriebenen optionalen Parameter für den Zwischenhalt verwendet werden.

7.6 Zusatzinformation in der Print-Datei

Bei der Verwendung der Parameter ist zu beachten, dass die Ausgabe der Print-Datei durch den Parameter `base64`=1 aktiviert sein muss.

7.6.1 psParamAddressExt1

Ermöglicht das Einblenden eines Zusatztextes unterhalb des Namens im Kopf der PDF-Version des Persönlichen Fahrplans.

7.6.2 psParamAddressExt2

Ermöglicht das Einblenden eines zweiten Zusatztextes unterhalb des ersten Zusatztexts im Kopf.

7.6.3 psParamAddressName

Ermöglicht das Einblenden des Namens im Kopf.

7.6.4 psParamAddressPlace

Ermöglicht das Einblenden von PLZ und Ort im.

7.6.5 psParamAddressStreet

Ermöglicht das Einblenden der Straße und Hausnummer.

8 DM-Request

Im Folgenden wird die Anfrage einer Abfahrtstafel (DM-Request) erläutert. Grundsätzlich gilt dabei zu beachten, dass es zwei unterschiedliche Typen von Abfahrtstafeln gibt: die Abfahrtstafel einer konkreten Haltestelle (siehe Abbildung 8) und eine kombinierte Abfahrtstafel für mehrere Haltestellen (siehe Abbildung 9 bzw. Abbildung 10).

Für die Abfahrtstafel einer Haltestelle muss zunächst die Haltestelle mit dem StopFinder-Request ermittelt werden. Die Haltestelle kann dabei auch über die Anfrage einer Adresse, eines POIs oder einer Koordinate ermittelt werden. In diesem Fall muss in einem Zwischenschritt eine der vom System ermittelten nahegelegenen Haltestellen ausgewählt werden (siehe Kapitel Wahl einer nahegelegenen Haltestelle).

Abfahrtsmonitor

Datum: 13.03.2013 **von:** München, Hauptbahnhof
Abfahrt: 09:00 Uhr

Datum/Zeit	Verkehrsmittel	Linie	Richtung
13.03. - 09:00 Uhr	U	U2	Feldmoching
13.03. - 09:00 Uhr	U	U4	Westendstraße
13.03. - 09:00 Uhr	Tram	20	Moosach Bahnhof
13.03. - 09:00 Uhr	Tram	19	Willibaldplatz
13.03. - 09:00 Uhr	U	U2	Messestadt Ost
13.03. - 09:01 Uhr	S	S4	Geltendorf
13.03. - 09:01 Uhr	Tram	16	Romanplatz
13.03. - 09:01 Uhr	S	S2	Ostbahnhof
13.03. - 09:01 Uhr	R	RB 57016 Regionalbahn	Donauwörth
13.03. - 09:01 Uhr	R	ALX 84124 alex - Länderbahn und Vogtlandbahn GmbH	Hof Hbf
13.03. - 09:01	R	RB 57212 Regionalbahn	Dinkelscherben

Abbildung 8 - Abfahrtsmonitor einer Haltestelle

Wird ein kombinierter Abfahrtsmonitor für mehrere Haltestellen angezeigt, kann der eingegebene Punkt auch ein wichtiger Punkt, eine Adresse oder Koordinate sein. Allerdings sollte im Abfahrtsmonitor darauf hingewiesen werden, an welcher Haltestelle abgefahren wird bzw. sollen die Abfahrten pro Haltestelle gruppiert werden (siehe Abbildung 9) oder für jede Abfahrt der Haltestellenname angegeben werden (siehe Abbildung 10). Dies ist dem Objet `location` eines `stopEvent`-Elements in der JSON-Antwort zu entnehmen.

Abbildung 9 - Kombinierter Abfahrtsmonitor I

Abbildung 10 - Kombinierter Abfahrtsmonitor II

8.1 Ansteuerung der Anfrage einer Abfahrtstafel

Die Anfrage der Abfahrtstafel wird über HTTP-Parameter gesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_DM_REQUEST?HTTP-Parameter
```

8.2 Hinweis zur Eingabe von Punkten

Bei der Eingabe eines Punktes wird als Erweiterung `<usage>` der Wert `dm` angegeben.

8.3 Obligatorische Parameter

Die Abfahrtstafel benötigt die Eingabe eines Punktes. Für die korrekte Funktionalität des JSON-Interfaces ist außerdem die Verwendung der folgenden Parameter unbedingt notwendig:

8.3.1 mode = direct

Bei der Verwendung des JSON-Interfaces muss Direkt-Modus des Abfahrtsmonitors verwendet werden.

8.3.2 useProxFootSearch = 0

Beim Direkt-Modus werden alternative Haltestellen gesucht, sofern diese per Konfiguration aktiviert wurden. Um zu gewährleisten, dass der Abfahrtsmonitor nur für die konkret angefragte Haltestelle gilt, muss die Suche von alternativen Haltestellen in diesem Fall mittels des Parameters `useProxFootSearch=0` (siehe Weitere Optionen für den öffentlichen Verkehr) ausgeschaltet werden.

8.4 Optionale Parameter

Es können die im Kapitel Eingabe von Datum und Uhrzeit beschriebenen HTTP-Parameter verwendet werden. Ebenso gilt der für den Trip-Request beschriebene Verkehrsmittelausschluss und Verkehrsmitteleinschluss. Außerdem folgende Parameter:

8.4.1 itdDateTimeDepArr

Schaltet zwischen einem Abfahrts- (`dep`) und Ankunftsmonitor (`arr`) um.

Mit den Werten `firstService` bzw. `lastService` werden die ersten bzw. letzten Fahrten ausgegeben. Der Uhrzeit-Parameter `itdTime` wird in diesem Fall nicht berücksichtigt. Standardmäßig findet die erste Fahrt um 4 Uhr morgens und die letzte Fahrt um 2 Uhr des Folgetages statt. Diese Zeiten können in der Konfigurationsdatei des EFA Controllers bzw. des EFA Servers in der geändert werden.

8.4.2 limit

Dieser Parameter legt die maximale Anzahl an Abfahrten fest, die ausgegeben werden. Standardmäßig werden bis zu 40 Fahrten innerhalb eines Zeitraums von maximal zwei Tagen ausgegeben.

8.4.3 useRealtime = 1

Aktiviert die Berücksichtigung der Echtzeitüberwachung (siehe Abbildung 11).

	Zeit	heute	Linie	in Richtung	Gleis / Bussteig	Hinweise
	08:48	08:50	BUS 44	Killesberg		Niederflurbus
	08:49	08:50	U9	Hedelfingen	2	
	08:50		U12	Killesberg	3	

Abbildung 11 - Abfahrtsmonitor mit Echtzeit

Die Ausgabe der Echtzeitinformation erfolgt analog zum Trip-Request (siehe Parameter useRealtime = 1)

8.5 Linienfilter

Optional kann in einem Zwischenschritt oder auf der Ergebnisseite ein Linienfilter angeboten werden, bei dem eine oder mehrere bedienenden Linien ausgewählt werden (siehe Abbildung 12). Die Abfahrtstafel zeigt dann nur die Abfahrten der ausgewählten Linien an. Die für diesen Filter benötigten Parameter werden im Kapitel Eingabe einer Linie beschrieben.

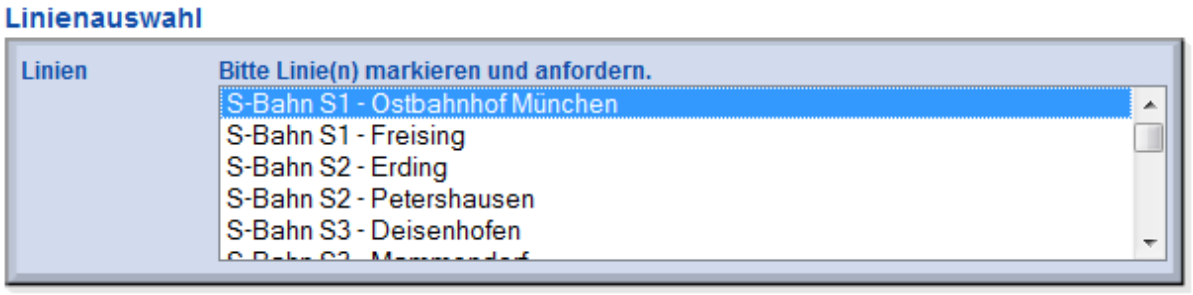

Abbildung 12 – Linienfilter bei der Abfahrtstafel

9 DMTTP-Request

Im Folgenden wird die Ansteuerung der Soll-Abfahrtstafel (DMTTP-Request) erläutert. Sie beinhaltet die Abfahrten ausgewählter Linien einer Haltestelle für die komplette aktuelle Fahrplanperiode. Die Abkürzung „TTP“ steht in diesem Sinne für „Time Table Period“.

9.1 Ansteuerung der Anfrage einer Soll-Abfahrtstafel

Die Anfrage der Soll-Abfahrtstafel wird über HTTP-Parameter gesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_DMTTP_REQUEST?HTTP-Parameter
```

9.2 Hinweis zur Eingabe von Punkten

Bei der Eingabe eines Punktes wird als Erweiterung `<usage>` der Wert `ss` angegeben.

9.3 Allgemeine Funktionalität

Die Soll-Abfahrtstafel funktioniert analog zur Abfahrtstafel und verwendet die gleichen HTTP-Parameter. Lediglich die Parametererweiterung `_<usage>` für die Eingabe eines Punktes unterscheidet sich.

10 TripStopTimes-Request

Im Folgenden wird die Anfrage der Haltestellensequenz inklusive An- und Abfahrtszeiten (TripStopTimes-Request) erläutert. Dieser Request dient dazu beispielsweise die Ergebnisse einer Fahrtanfrage oder einer Abfahrtstafel um durchfahrene Haltestellen anzureichern (siehe Abbildung 13 - Zwischenhalte mit Ankunftszeiten in der Detailansicht einer Fahrtauskunft).

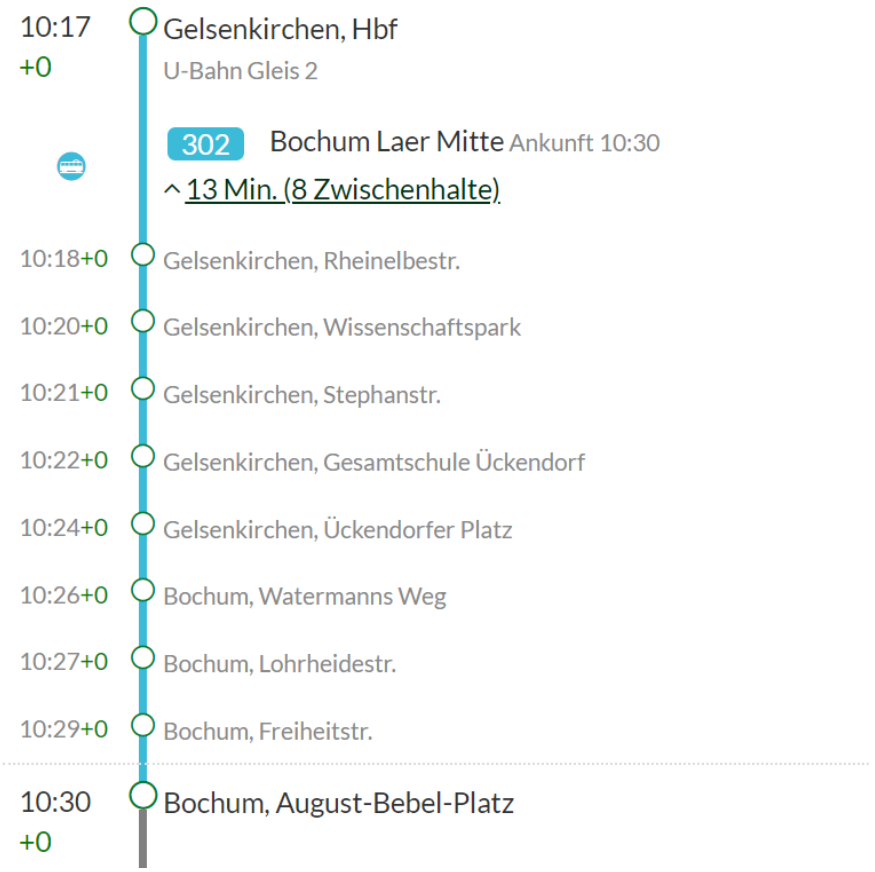

Abbildung 13 - Zwischenhalte mit Ankunftszeiten in der Detailansicht einer Fahrtauskunft

10.1 Ansteuerung des TripStopTimes-Requests

Der TripStopTimes-Request wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_TRIPSTOPTIMES_REQUEST?HTTP-Parameter
```

10.2 Obligatorische Parameter

Der TripStopTimes-Request benötigt Parameter, um die Linie zu identifizieren, für die die durchfahrenen Haltestellen und ihre Zeiten ermittelt werden sollen. Die Werte der Parameter können beispielweise der JSON-Antwort eines Trip-Requests entnommen werden (siehe Abbildung 14).

```
legs: [
  - {
        duration: 2640,
      - origin: {
            isGlobalId: true,
            id: "18020301",
            name: "King Fahd_01",
            type: "platform",
          + coord: [2],
            niveau: 0,
          + parent: {4},
            departureTimePlanned: "2020-08-26T04:45:00Z",
            departureTimeEstimated: "2020-08-26T04:45:00Z",
          + properties: {2},
        },
      - destination: {
            isGlobalId: true,
            id: "12420302",
            name: "Al Batha_01",
            type: "platform",
          + coord: [2],
            niveau: 0,
          + parent: {4},
            arrivalTimePlanned: "2020-08-26T05:29:00Z",
            arrivalTimeEstimated: "2020-08-26T05:29:00Z",
          + properties: {2},
        },
      - transportation: {
            id: "rda:01008: :R:a20",
            name: "Bus 8",
            disassembledName: "8",
            number: "8",
            description: "King Fahd_01-Al Batha_01",
          + product: {4},
          + operator: {3},
          + destination: {3},
          - properties: {
                tripCode: 53,
                timetablePeriod: "import",
                mtSubcode: "0",
                lineDisplay: "LINE",
            },
        },
```


Abbildung 14 - Ausschnitt der JSON-Ausgabe einer Fahrtanfrage

10.2.1 **line**

ID der Linie. Siehe auch Eingabe einer Linie.

10.2.2 **stopID**

ID der Starthaltestelle.

10.2.3 **tripCode**

Fahrtenschlüssel.

10.2.4 **date**

Datum der Abfahrt im Format `<YYYYMMDD>`.

10.2.5 **time**

Zeit der Abfahrt im Format `<HHMM>`.

10.3 **Optionale Parameter**

10.3.1 **tStOTType**

Filtern der Haltestellensequenz nach:

- **`ALL`** - alle Haltestellen
- `NEXT` - alle Haltestellen, die auf die angegebene folgen (siehe Parameter `stopID`)
- `PREVIOUS` - alle Haltestellen, die der angegebenen vorausgegangen sind (siehe Parameter `stopID`)

10.3.2 **useRealtime = 1**

Aktiviert die Berücksichtigung der Echtzeitüberwachung (siehe auch useRealtime = 1).

11 StopSeqCoord-Request

Im Folgenden wird die Anfrage der Haltestellen- und Koordinatensequenz einer Linie (StopSeqCoord-Request) erläutert. Dieser Request dient dazu beispielsweise die Ergebnisse einer Fahrtanfrage oder einer Abfahrtstafel auf einer interaktiven Karte einzuzeichnen (siehe Abbildung 15 - Visualisierung der Linie und der durchfahrenen Haltestellen auf einer interaktiven Karte).

Abbildung 15 - Visualisierung der Linie und der durchfahrenen Haltestellen auf einer interaktiven Karte

11.1 Ansteuerung des StopSeqCoord-Requests

Der TripStopTimes-Request wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_TRIPSTOPTIMES_REQUEST?HTTP-Parameter
```

11.2 Obligatorische Parameter

Der StopSeqCoord-Request benötigt Parameter, um die Linie zu identifizieren, für die die durchfahrenen Haltestellen und die Koordinatensequenz ermittelt werden sollen. Die Werte der Parameter können beispielweise der JSON-Antwort eines Trip-Requests entnommen werden (siehe Abbildung 14). Die Parameter sind identisch zu den Obligatorischen Parametern und den Optionalen Parametern des TripStopTimes-Requests.

12 LineStop-Request

Im Folgenden wird die Anfrage der durchfahrenen Haltestellen einer Linie (LineStop-Request) erläutert. Eine Kombination aus Linie und durchfahrener Haltestelle wird häufig für die Print-Produkte wie beispielsweise den StopTimetable-Request benötigt.

12.1 Ansteuerung des LineStop-Requests

Der LineStop-Request wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_LINESTOP_REQUEST?
HTTP-Parameter
```

12.2 Obligatorische Parameter

Der Linestop-Request benötigt einen Parameter, um die Linie zu identifizieren, für die die durchfahrenen ermittelt werden sollen.

12.2.1 line

ID der Linie. Siehe auch Eingabe einer Linie.

12.3 Optionale Parameter

12.3.1 allStopInfo = 1

Aktiviert die Ausgabe von zusätzlicher Information zu den Haltestellen, wie Bereiche und Steige.

13 StopTimetable-Request

Im Folgenden wird die Ansteuerung des Aushangfahrplans (StopTimetable-Request) erläutert. Aushangfahrpläne werden für bestimmte Haltestellen und Linien erstellt.

Im Kapitel Anwendungsfälle wird im Abschnitt Printprodukte beschrieben, wie eine Haltestelle und eine bedienende Linie oder eine Linie und eine durchfahrene Haltestelle ermittelt werden.

13.1 Ansteuerung des Aushangfahrplans

Der Aushangfahrplan wird über HTTP-Parameter angesteuert. Als Antwort wird ein base64 codiertes Printprodukt erstellt. Zusätzlich liegt die Antwort in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_STT_REQUEST?HTTP-Parameter
```

13.2 Hinweis zur Punktverifikation

Bei der Eingabe eines Punktes wird als Erweiterung `<usage>` der Wert `stt` angegeben.

13.3 Obligatorische Parameter

Zur Generierung eines Aushangfahrplans wird eine Haltestelle und eine bedienende Linie benötigt. Die Eingabe dieser ist in den Kapiteln Eingabe eines Punktes und Eingabe einer Linie beschrieben. Es muss allerdings sichergestellt sein, dass es sich bei der Linie um eine bedienende Linie der Haltestelle handelt, bzw. bei der Haltestelle, dass es sich um eine durchfahrene Haltestelle der Linie handelt.

Außerdem ist folgender Parameter notwendig:

13.3.1 mode = direct

Die der Verwendung des EFA JSON-Interfaces muss der Direkt-Modus aktiviert sein.

13.4 Optionale Parameter

Es können die im Kapitel Eingabe einer Linie beschriebenen HTTP-Parameter verwendet werden. Außerdem:

13.4.1 base64 = 1

Fordert das base64 kodierte Printprodukt an.

13.4.2 mrgSt = 1

Die Aushangfahrpläne werden standardmäßig steiggenau ausgegeben. Das kann zu doppelten Aushangfahrplänen führen, wenn Aushangfahrpläne für mehrere Steige gelten (siehe Abbildung 16). Ist dieser Parameter aktiv, werden doppelt vorhandene Aushangfahrpläne nur einfach ausgegeben.

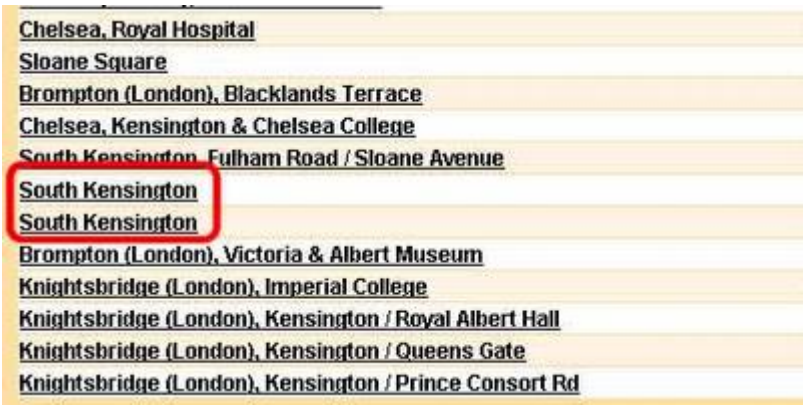

Abbildung 16 - Aushangfahrplan der Linie 360 von Traveline South East mit mrgSt=0

14 CoordInfo-Request

Im Folgenden wird die Ansteuerung der Koordinatenanfrage (CoordInfo-Request) erläutert. Diese dient dazu Koordinaten verschiedener Objekte anzufragen. Dabei kann es sich um Haltestellen, wichtige Punkte, Sharer oder Parkobjekte handeln. Diese Punkte können anschließend auf einer interaktiven Karte dargestellt werden (siehe Abbildung 17).

Die Auswahl der Punkt-Typen und die Einschränkung des Suchraums erfolgen über die weiter unten beschriebenen HTTP-Parameter.

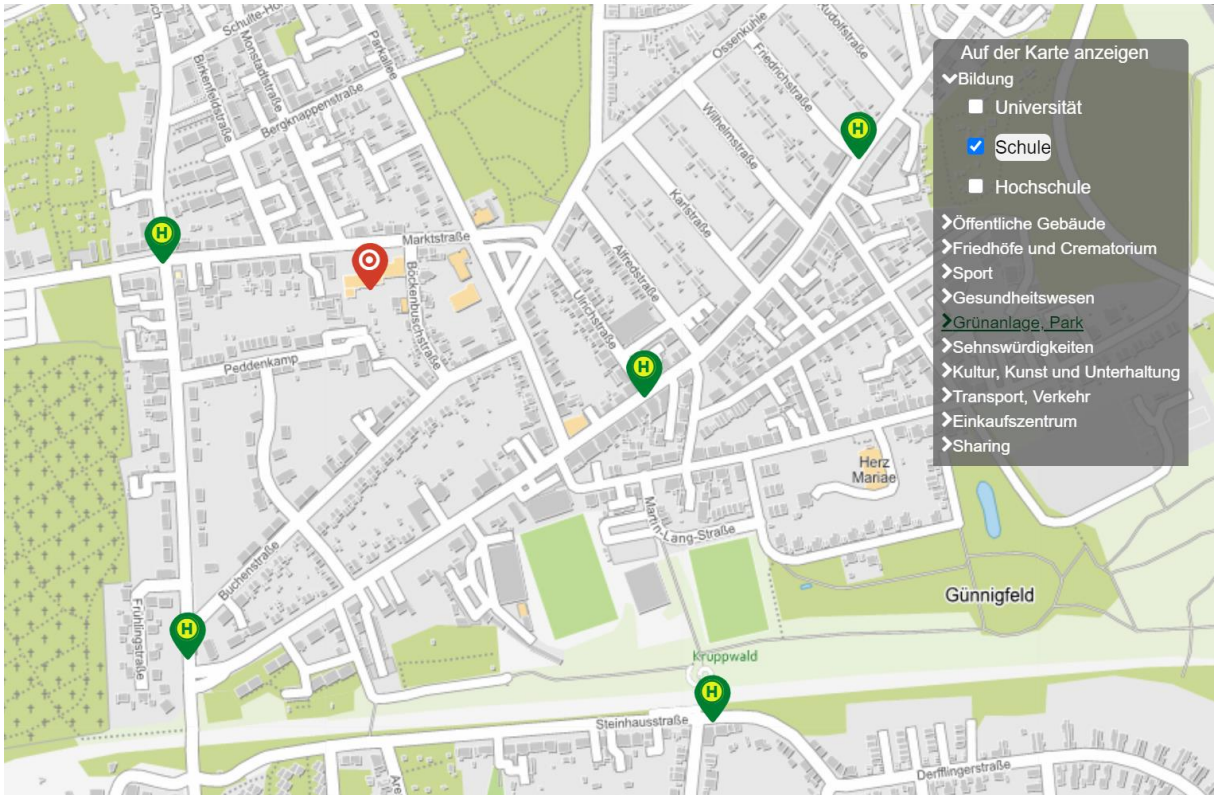

Abbildung 17 – Interaktive Karte mit Haltestellen und ausgewählten POIs

Aktuell dient der CoordInfo-Request auch dazu Detailinformation von Sharing-Objekten anzufragen. Siehe hierzu das Kapitel Anfrage von Detailinformation eines Sharing-Objekts.

14.1 Ansteuerung der Koordinatenanfrage

Die Koordinatenanfrage wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_COORD_REQUEST?HTTP-Parameter
```

Beispiele:

Anfrage von Punkt POIs mit der Umkreissuche:

```
http://server:port/virtuellesVerzeichnis/XML_COORD_REQUEST?
inclFilter=1&coord=7.075195:51.536086:WGS84[DD.DDDDD]&radius_1
=5000&type_1=POI_POINT
```

Anfrage von Sharing-Objekten aus dem Bediengebiet 1 des Betreibers mit der ID 1006 innerhalb einer Bounding Box:

```
http://server:port/virtuellesVerzeichnis/XML_COORD_REQUEST?XML
_COORD_REQUEST?outputFormat=rapidJSON&hideBannerInfo=1&vehSM=B
B&vehBB=BB&vehBBLU=11.1:48.6:WGS84[dd.ddddd]&vehBBRL=12.1:47.6
:WGS84[dd.ddddd]&vehPId1=1006&vehOpArId1=1
```

Anfrage von Infrastrukturelementen im Umkreis einer Koordinate:

```
http://server:port/virtuellesVerzeichnis/XML_COORD_REQUEST?coo
rd=3513309%3A755034%3ANBWT&inclFilter=1&type_1=INFRASTRUCTURE&
radius_1=452000
```

14.2 Obligatorische Parameter

Das Suchgebiet muss entweder durch eine Umkreissuche oder eine Bounding Box ausgewählt werden. Des Weiteren muss mindestens ein Punkt-Typ durch die Filter zur Auswahl der Punkt-Typen ausgewählt werden.

14.3 Umkreissuche

14.3.1 coord

Mit diesem Parameter wird die Zentrumskoordinate angegeben, die den Mittelpunkt der Suche nach den Punkten darstellt. Die Koordinate setzt sich zusammen aus der x-Koordinate, der y-Koordinate und dem Koordinatensystem (in der Regel `WGS84[dd.ddddd]`) jeweils getrennt durch einen Doppelpunkt.

Hinweis: Der Suchradius `radius_<Filterindex>` wird für jeden Punkt-Typ `type_<Filterindex>` separat angegeben (siehe Filter zur Einschränkung des Suchraums).

14.4 Bounding Box

Es ist möglich nur den Teil der Koordinatensequenz, der innerhalb einer Bounding Box liegt, auszugeben. Dazu sind folgende Parameter notwendig:

14.4.1 boundingBox

Dieser Parameter aktiviert die Bounding Box und benötigt keinen Wert. Sobald er angegeben wird, wird ausschließlich innerhalb des Rechtecks, dass durch die linke obere und rechte untere Ecke angegeben wird, gesucht. Die Ecken werden durch die Parameter `boundingBoxLU` und `boundingBoxRL` angegeben. Die Angabe ist unbedingt erforderlich.

14.4.2 boundingBoxLU = <x>:<y>:<Koordinatensystem>

Die mit diesem Parameter angegebene Koordinate stellt die linke obere Ecke (left, upper) einer Bounding Box dar. Der Wert des Parameters setzt sich aus der x-Koordinate, der y-Koordinate und dem Koordinatensystem (in der Regel `WGS84[dd.ddddd]`), jeweils getrennt durch einen Doppelpunkt, zusammen.

14.4.3 boundingBoxRL = <x>:<y>:<Koordinatensystem>

Die mit diesem Parameter angegebene Koordinate stellt die rechte untere Ecke (right, lower) einer Bounding Box dar. Der Wert des Parameters setzt sich aus der x-Koordinate, der y-Koordinate und dem Koordinatensystem (in der Regel `WGS84[dd.ddddd]`), jeweils getrennt durch einen Doppelpunkt, zusammen.

14.5 Filter zur Auswahl der Punkt-Typen

Für die Koordinatenanfrage stehen erweiterte Filter zur Verfügung. Um diese verwenden zu können, ist ihre Aktivierung durch den Parameter `inclFilter=1` notwendig.

Die Filter zeichnen sich durch ein Suffix `_<Filterindex>` aus. Auf diese Weise können verschiedene Filter miteinander kombiniert werden und auf einen Punkt-Typ (siehe `type_<Filterindex>)` angewendet werden. Kombinierte Filter haben das gleiche Suffix.

Beispiel: Durch die Angabe von Filtern ist es möglich nur ausgewählte Punkt-Typen, wie Haltestellen, anzufragen und diese gegebenenfalls durch ihre Zeichenklasse einzuschränken. Der Typ "Haltestelle" wird durch den Parameter `type_<Filterindex>=STOP` gewählt; die Zeichenklasse durch den Parameter `inclDrawClasses_<Filterindex>`. Die beiden Suffixe müssen identisch sein.

14.5.1 inclFilter = 1

Aktivierung der Filter zur Einschränkung des Suchraums bei der Ermittlung von Objekten. Auf diese Weise können Objekt nach bestimmten Kriterien gefiltert werden.

14.5.2 type_<Filterindex>

Mit diesem Parameter kann ein bestimmter Punkt-Typ ausgewählt werden. Mehrere Punkt-Typen können durch die mehrfache Verwendung dieses Parameters angefragt werden. Dabei wird ein anderer Index `<Filterindex>` vergeben. Folgende Typen sind möglich:

- `ANY` Alle Punkte
- `BUS_POINT` Steige
- `ENTRANCE` Eingänge
- `GIS_AREA` GIS-Gebiet
- `GIS_POINT` GIS-Punkt

- `INFRASTRUCTURE` Infrastrukturelemente
- `LINE` Linien, die über das Straßensegment der übergebenen Koordinate `coord` fahren
- `POI_AREA` Flächen-POIs (wichtige Flächen-Punkte)
- `POI_POINT` Punkt-POIs (wichtige Punkte)
- `STOP` Haltestellen
- `STREET` Straßen

Sollen weitere Filter (siehe Filter zur Einschränkung des Suchraums) auf einen Element-Typen angewendet werden, so haben diese das gleiche Suffix `_<Filterindex>`.

14.6 Filter zur Einschränkung des Suchraums

Die Filter zur Einschränkung des Suchraums funktionieren analog zu den Filtern zur Auswahl der Punkt-Typen. Auch hier gilt, dass die Filter-Parameter durch den Parameter `inclFilter = 1` aktiviert sein müssen.

14.6.1 radius_<Filterindex>

Mit diesem Parameter wird der Radius für die Umkreisssuche in Metern angegeben. Den Mittelpunkt der Suche stellt die Zentrumskoordinate `coord` dar.

14.7 Filter für Haltestellen

Einige Parameter benötigen das Suffix `_<Filterindex>`. Dies ist in Kapitel Filter zur Auswahl von Punkt-Typen beschrieben.

14.7.1 exclMOT_<Filterindex>

Mit diesem Parameter können ein oder mehrere Verkehrsmittel IDs (siehe Verkehrsmitteltypen) die bei der Suche nach Haltestellen vom Typ type_<Filterindex> ausgeschlossen werden. Haltestellen, die ausschließlich von den angegebenen Verkehrsmitteln bedient werden, werden in den Suchergebnissen nicht berücksichtigt. Mehrere Verkehrsmittel IDs werden durch einen Doppelpunkt getrennt angegeben.

14.7.2 fltrParkObj = 1

Bei der Suche nach Haltestellen werden ausschließlich solche berücksichtigt, die innerhalb von ausgezeichneten Parkzonen liegen.

14.7.3 inclMOT_<Filterindex>

Mit diesem Parameter können ein oder mehrere Verkehrsmittel IDs (siehe Verkehrsmitteltypen) die bei der Suche nach Haltestellen vom Typ type_<Filterindex> eingeschlossen werden. Damit werden ausschließlich Haltestellen, die von diesem Verkehrsmittel bedient werden, bei der Suche anach Haltestellen berücksichtigt. Mehrere Verkehrsmittel IDs werden durch einen Doppelpunkt getrennt angegeben.

14.8 Filter für POI

Einige Parameter benötigen das Suffix `_<Filterindex>`. Dies ist in Kapitel Filter zur Auswahl von Punkt-Typen beschrieben.

14.8.1 exclLayers_<Filterindex>

Mit diesem Parameter kann man GIS-Layer angeben, der nicht berücksichtigt werden soll. Dadurch reduziert sich die Suchzeit. Mehrere GIS-Layer können durch einen Doppelpunkt getrennt angegeben werden.

14.8.2 inclDrawClasses_<Filterindex>

Mit diesem Parameter kann eine Zeichenklasse mittels ihrer ID angegeben werden, die die gesuchten Punkte haben müssen. Als Wert wird der numerische Code der Zeichenklasse (siehe Handbuch zur Karten-Konfiguration) übergeben. Mehrere Zeichenklassen können durch Doppelpunkt getrennt angegeben werden.

Beispiel: `51:45:43:56:78:64:44:46:42:40:52:72:62:48:49:69:63`

14.8.3 purpose

Mit diesem Parameter kann die Suche nach wichtigen Punkten (POIs) auf Punkte mit einem bestimmten Verwendungszweck reduziert werden. Das heißt, dass bestimmte Gruppen von wichtigen Punkten separat behandelt werden können.

Hinweis: Durch den Verwendungszweck ist es möglich verschiedenen wichtigen Punkten verschiedene Konfigurationen zuzuweisen. Dies Erfolgt durch verschiedene Sektionen mit dem Namen des Verwendungszwecks in der Konfigurationsdatei des EFA IT-Kernels. Diese Funktionalität kann auch durch Einschränkung des Suchraums mittels einer Zeichenklasse abgebildet werden. Hierzu dient der Parameter `inclDrawClasses <Filterindex>`.

Beispiel: `purpose=7`

14.8.4 inclPOIH_<Filterindex>

Mit diesem Parameter können eine oder mehrere POI Hierarchien angegeben werden, die die gesuchten wichtigen Punkte (`type_<Filterindex>=POI_AREA` oder `type_<Filterindex>=POI_POINT`) haben müssen. Sollen mehrere Hierarchien angegeben, werden diese durch einen Doppelpunkt getrennt. Es werden auch die Punkte einer durch diesen Parameter angegebenen Hierarchie angefordert, wenn die Ausgabe der entsprechenden Hierarchie über die Konfigurationsdatei des EFAITKernels unterdrückt wird.

Beispiel: `inclPOIH_1=A:B:D`

14.9 **Filter für Parkobjekte**

Neben den hier beschriebenen Parametern ist es möglich, Echtzeitinformation zur Auslastung zu erhalten. Die Parameter hierfür sind für den ParkObject-Request dokumentiert.

Da die Anfrage der Echtzeit die Performance verschlechtert, sollte diese nur angefragt werden, wenn dies notwendig ist. Dies ist der Fall, wenn beispielsweise die Icons auf der Karte gemäß ihrer Auslastung farbig dargestellt werden sollen (siehe Abbildung 18).

Abbildung 18 - Parkobjekte mit Echtzeitauslastung

Beispiel:

Anfrage aller Parkhäuser und Parkplätze innerhalb des Kartenausschnitts:

```
http://server:port/virtuellesVerzeichnis/XSLT_COORD_REQUEST?parkingSearchMethod=BB&parkingBB=1&parkingBBLU=11.1:48.6:WGS84[dd.ddddd]&parkingBBRL=12.1:47.6:WGS84[dd.ddddd]&parkingObjectType=3
```

Anfrage der Auslastung für alle Parkobjekte innerhalb des Kartenausschnitts:

```
http://server:port/virtuellesVerzeichnis/XML_COORD_REQUEST?parkingSearchMethod=BB&parkingBB=1&parkingBBLU=11.1:48.6:WGS84[dd.ddddd]&parkingBBRL=12.1:47.6:WGS84[dd.ddddd]&parkObjInfo=1&parkObjInfoRealtime=1
```

14.9.1 **Obligatorische Parameter**

Für die Anfrage von Parkobjekten zur Anzeige auf einer interaktiven Karte sollte die Suche innerhalb einer Bounding Box verwendet werden. Folgende Parameter sind dafür notwendig:

14.9.1.1 parkingSearchMethod = BB

Mit diesem Parameter wird der Suchmodus auf die Suche von Parkobjekten innerhalb einer Bounding Box umgestellt.

14.9.1.2 parkingBB = 1

Aktiviert die Suche von Parkobjekten innerhalb einer Bounding Box.

14.9.1.3 parkingBBLU

Angabe der linken oberen Ecke der Bounding Box. Der Wert des Parameters setzt sich aus dem x- und dem y-Wert der Koordinate sowie dem Koordinatenformat (in der Regel `WGS84[dd.ddddd]`) zusammen. Die Einzelwerte werden durch einen Doppelpunkt getrennt angegeben.

14.9.1.4 parkingBBRL

Angabe der rechten unteren Ecke der Bounding Box. Der Wert des Parameters setzt sich aus dem x- und dem y-Wert der Koordinate sowie dem Koordinatenformat (in der Regel `WGS84[dd.ddddd]`) zusammen. Die Einzelwerte werden durch einen Doppelpunkt getrennt angegeben.

14.9.2 Optionale Parameter zur Filterung

14.9.2.1 parkingObjectType

Mit diesem Parameter kann nach bestimmten Objekt-Typen gefiltert werden. Folgende Werte sind möglich:

- `-1` - undefinierter Parkobjekt-Type
- `1` - Parkhaus
- `2` - Parkplatz
- `4` - einzelner Parkplatz
- `8` - Parkzone
- `32` - Ladestationen

Der Parameter funktioniert als Bitmaske. Um mehrere Objekt-Typen auszuwählen, werden deren Werte addiert.

Beispiel: Für Parkhäuser und Parkplätze ist der Wert 3 (= 1 + 2).

Anmerkung: Wert 16 ist veraltet und kann in der aktuellen Version des Interfaces nicht mehr verwendet werden.

14.9.2.2 parkingCharge

Mit diesem Parameter können für Parkobjekte Einschränkungen hinsichtlich des Preises vorgenommen werden. Es stehen folgende Werte zur Verfügung:

- `free` - ausschließlich kostenfreie Parkobjekte
- `pay` - ausschließlich kostenpflichtige Parkobjekte

Standardmäßig werden alle Parkobjekte berücksichtigt.

14.9.2.3 parkingBikeRideInclusion

Standardmäßig werden alle Fahrrad-Parkobjekte ausgegeben. Dabei spielt es keine Rolle, ob sich als Bike&Ride-Parkobjekt gekennzeichnet sind. Mit diesem Parameter können solche Objekte ausgeschlossen oder exklusiv ausgegeben werden. Folgende Werte sind möglich:

- **`INCLUDE`** - Bike&Ride-Objekte sind enthalten
- `EXCLUDE` - Bike&Ride-Objekte werden ausgeschlossen
- `EXCLUSIVE` - ausschließlich Bike&Ride-Objekte

14.9.2.4 parkingParkRideInclusion

Standardmäßig werden alle PKW-Parkobjekte ausgegeben. Dabei spielt es keine Rolle, ob sich als Park&Ride-Parkobjekt gekennzeichnet sind. Mit diesem Parameter können solche Objekte ausgeschlossen oder exklusiv ausgegeben werden. Folgende Werte sind möglich:

- **`INCLUDE`** - Park&Ride-Objekte sind enthalten
- `EXCLUDE` - Park &Ride-Objekte werden ausgeschlossen
- `EXCLUSIVE` - ausschließlich Park&Ride-Objekte

14.10 Filter für Sharing-Objekte

Zum Filtern der Sharing-Objekte nach Betreiber und Bediengebiet, können die für den OpArea-Request im Kapitel Filter beschriebenen Parameter `vehPId<Index>` und `opAreaId<Index>` verwendet werden.

14.11 Weitere optionale Parameter

14.11.1 deadline

Mit diesem Parameter kann ein Datum angegeben werden, dass die zu diesem Zeitpunkt gültigen Punktobjekte (zum Beispiel Haltestellen) ermittelt. Das Darum wird als vierstellige Jahreszahl, gefolgt von der zweistelligen Monatszahl und der zweistelligen Tageszahl angegeben. Wird dieser Parameter nicht verwendet, werden die Punktobjekte, die zu der aktuellen Server-Zeit gültig sind, ermittelt.

14.11.2 mapNameOutput

Mit diesem Parameter wird das Koordinatensystem (in der Regel `WGS84[dd.ddddd]`) des GIS Netzes angegeben. Der Parameter ist insbesondere dann sinnvoll, wenn das Koordinatensystem der Karte nicht mit dem Koordinatensystem, in dem die Daten vorliegen (GIS Netz), übereinstimmt. Entsprechen sich das Koordinatensystem der Karten und der Daten, kann der Parameter weggelassen werden. Aus Performancegründen ist es jedoch sinnvoll ihn immer zu übergeben, da so die Ermittlung des Koordinatensystems durch den EFAITKernel wegfallen kann.

Beispiel: `mapNameOutput=WGS84[dd.ddddd]`

14.11.3 max

Mit diesem Parameter wird die maximale Anzahl an Elementen angegeben, die mittels der Koordinatenanfrage ermittelt und ausgegeben werden sollen. Standardmäßig liegt keine Beschränkung der Anzahl vor.

14.12 Anfrage von Detailinformation eines Sharing-Objekts

Aktuell erfolgt die Anfrage von Detailinformation eines Sharing-Objekts (siehe Abbildung 19) über den CoordInfo-Request. In einer zukünftigen Version des Interfaces wird es dafür analog zum ParkObject-Request, zur Anfrage von Detailinformation eines Parkobjekts, einen eigenständigen Request geben.

Abbildung 19 - Detailinformation eines Car-Sharing-Objekts

Zu beachten ist, dass mit dem CoordInfo-Request ausschließlich Detailinformationen zu Fahrzeugen und Stationen ausgegeben werden können. Nicht aber zum

Betreiber. Soll die Ausgabe mit Betreiberinformation, wie zum Beispiel dem Betreibernamen oder dem Bediengebiet, angereichert werden, muss diese mit dem OpArea-Request angefragt werden.

Beispiel:

```
http://server:port/virtuellesVerzeichnis/XML_COORD_REQUEST?hideBannerInfo=1&coordReqType=VEHICLEDETAIL&vehDetail=1&vehSM=ID&SharingRefId=WDD1173421N698305:1006:1&vehSpec=1
```

14.12.1 Obligatorische Parameter

Um Detailinformation für ein Sharing-Objekt abzufragen, ist es notwendig den Anfragemodus des CoordInfo-Requests umzustellen und die Ausgabe der Details zu aktivieren. Zusätzlich muss das Sharing-Objekt identifizieren werden.

14.12.1.1 coordReqType=VEHICLEDETAIL

Stellt den Anfragemodus des CoordInfo-Request auf die Detailanfrage von Sharing-Objekten um.

14.12.1.2 SharingRefId

Dieser Parameter dient dazu das Sharing-Objekt zu identifizieren. Der Wert des Parameters setzt sich aus der ID, der Betreiber ID und der ID des Bediengebiets zusammen. Die Werte werden durch einen Doppelpunkt getrennt.

14.12.2 vehAtStation = 1

Aktiviert die Ausgabe von Fahrzeugen in Stationen. Er ist nur obligatorisch, wenn stationäre Fahrzeuge berücksichtigt werden sollen.

14.12.2.1 vehDetail = 1

Aktivierung der Ausgabe der Detailinformation eines einzelnen Fahrzeugs.

14.12.3 vehSM = ID

Schaltet den Such-Modus der Fahrzeugsuche des CoordInfo-Requests auf die Suche nach einem einzelnen Fahrzeug um.

14.12.4 vehSpec = 1

Aktiviert die Ausgabe von Detailinformation.

14.12.5 Optionale Parameter

Die folgenden Parameter dienen der Filterung oder der Ausgabe zusätzlicher Details.

14.12.5.1 vehAvail = 1

Aktiviert die Verfügbarkeit in der Ausgabe.

14.12.5.2 vehEngineType

Mit diesem Parameter können die Sharer-Objekte nach Antriebsart gefiltert werden. Folgende Werte sind möglich:

- `COMBUSTION`
- `ELECTRIC`
- `HYBRID` -

Der Parameter kann mehrfach verwendet werden, um mehrere Antriebsarten auszuwählen.

14.12.6 vehLocType<Index>

Dieser Parameter ermöglicht das Filtern nach stationsgebundenen (`CLASSIC` oder `STATION`) oder freien Sharing-Objekten (`FREEFLOATER` oder `POSITION`). Folgende Werte sind möglich:

- `CLASSIC`
- `FREEFLOATER`
- `STATION`
- `POSITION`

Der Parameter kann mehrfach verwendet werden.

14.12.7 vehSF

Mit diesem Parameter können die Objekte nach Sharer-Typ gefiltert werden. Folgende Werte sind möglich:

- `CAR`
- `BIKE`
- `MINIVEHICLE` (z.B. Scooter)
- `TAXI`

Der Parameter kann mehrfach verwendet werden.

15 GeoObject-Request

Im Folgenden wird die Anfrage von Linienverläufen mit dem GeoObject-Request erläutert. Der Linienverlauf wird als Koordinatenfolge ausgegeben. Die durchfahrenen Haltestellen können ebenfalls ermittelt. Linienverlauf und Haltestellen können beispielsweise auf einer Karte eingezeichnet werden (siehe Abbildung 20).

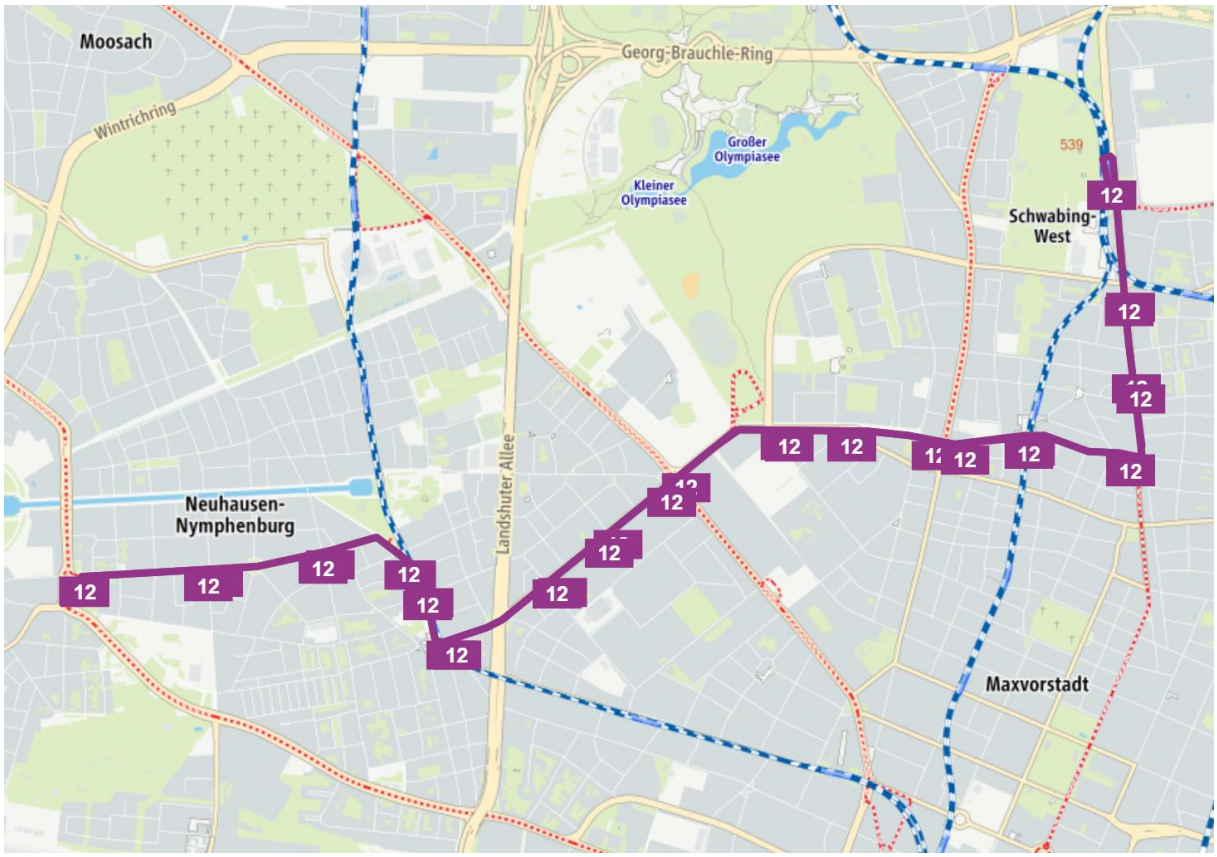

Abbildung 20 - Verlauf der Linie 12

15.1 Ansteuerung des GeoObject-Requests

Der GeoObject-Request wird mit HTTP-Parametern gesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_GEOOBJECT_REQUEST?HTTP-Parameter
```

15.2 Obligatorische Parameter

Für den GeoObject-Request ist nur die Angabe einer Linie zwingend notwendig. Der Parameter enthält gleichzeitig die Information, ob die durchfahrenen Haltestellen mit ausgegeben werden sollen oder nicht.

15.2.1 **line**

ID der Linie. Siehe auch Eingabe einer Linie.

15.3 **Optionale Parameter**

Die folgenden Parameter sind optional und stellen zusätzliche Funktionalität zur Verfügung:

15.3.1 **filterDate = <JJJJMMTT>**

Dieser Parameter gibt das Datum an, für das der Linienverlauf erstellt werden soll. Der Wert setzt sich zusammen aus der vierstelligen Angabe des Jahres, der zweistelligen Angabe des Monats und der zweistelligen Angabe des Tages.

Anmerkung: Die Angabe des Datums ist interessant, da die Linienverläufe beispielsweise an unterschiedlichen Wochentagen voneinander abweichen können. Denkbar wäre, dass die letzten beiden Haltestellen einer Linie sonntags nicht angefahren werden.

15.3.2 **filterStopID**

Durch diesen Filter ist es möglich nur die Teile eines Linienverlaufs auszugeben, die über bestimmte Haltestellen verlaufen. Die ID der Haltestelle wird als Wert des Parameters übergeben. Der Parameter kann mehrfach verwendet werden.

15.3.3 **lineReqType**

Typ der angefragten Linien. Folgende Werte sind möglich:

- 0 - alle Linien
- 2 - nur Linien mit Aushangfahrplan (STT)
- 4 - nur Linien mit Fahrplanbuchseiten (TTB)
- 8 - nur Linien mit Routen-Karten (ROB)
- 16 - nur Linien mit ZOB Haltestellen-Fahrplan (StationTT)

Dabei handelt es sich um eine Bitmaske. Zur Wahl mehrere Typen können die Werte addiert werden.

15.3.4 **vSL = 1**

Der Parametername ist eine Abkürzung für „verify serving lines“. Ist er aktiv, werden die bedienenden Linien aller durchfahrenen Haltestellen ausgegeben. Diese sind standardmäßig nicht Bestandteil der Ausgabe.

15.4 **Bounding Box**

Die Suche der Punkte kann innerhalb einer Boundin Box stattfinden. Hierzu werden die Parameter, die im Kapitel Bounding Box für den CoordInfo-Request beschrieben wurden, verwendet.

16 ParkObject-Request

Im Folgenden wird die Ansteuerung des ParkObject-Requests erläutert. Dieser wird zur Anfrage der Detailinformation eines Parkobjekts genutzt. Ein Parkobjekt kann ein Parkhaus, ein Parkplatz, ein Park&Ride-Parkplatz oder eine Parkzone sein. Sofern vorhanden, kann Echtzeitinformation zur Auslastung angefordert werden.

16.1 Ansteuerung des ParkObject-Requests

Der Parkobject-Request wird mit HTTP-Parametern gesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_PARKOJECT_REQUEST
?HTTP-Parameter
```

Abbildung 21 - Detailinformation eines Parkplatzes

Beispiel:

Anfrage von Detailinformation inklusive Echtzeitinformation zur Auslastung für das Parkobjekt mit der ID 1-751.

```
http://server:port/virtuellesVerzeichnis/XML_PARKOBJECT_REQUES
T?paID=1-
751&parkObjInfo=1&extendedParkingClasses=1&parkObjectInfoRealt
ime=1`
```

16.1.1 Obligatorische Parameter

16.1.1.1 extendedParkingClasses = 1

Aktiviert die strukturiere Ausgabe der Detailinformation.

16.1.1.2 parkObjInfo = 1

Aktiviert die Ausgabe der Detailinformation.

16.1.2 Optionale Parameter

16.1.2.1 parkObjInfoStatic = 1

Aktiviert die Ausgabe von statischen Informationen, wie der Anzahl der Parkplätze oder der Adresse des Parkobjekts.

16.2 Anfrage der Detailinformation eines Parkobjekts

Aus Performancegründen ist es sinnvoll Detailinformation nur zu einzelnen Parkobjekten anzufragen (siehe Abbildung 21).

16.2.1.1 paID

Mit diesem Parameter kann die ID eines Parkobjekts angegeben werden, um dieses auszuwählen. Soll eine als Parkobjekt ausgewiesene Haltestelle ausgewählt werden, wird stattdessen der Parameter `paIDStop` zur Wahl des Parkobjekts verwendet.

16.2.1.2 paIDStop

Dieser Parameter stellt eine Alternative zu `paID` dar. Er dient dazu als Parkobjekt ausgewiesene Haltestellen auszuwählen.

16.3 Echtzeitinformation zur Auslastung

Für die Anfrage von Echtzeitinformation muss die Anfrage von Detailinformation durch den Parameter `parkObjInfo = 1` aktiviert sein.

16.3.1 parkObjInfoRealtime = 1

Fügt Echtzeitinformation über die Auslastung der Parkobjekte hinzu.

17 OpArea-Request

Im Folgenden wird die Ansteuerung der Anfrage von Sharing-Anbietern, ihres Angebots und ihren Bediengebieten (OpArea-Request) erläutert. Bei den Sharing-Anbietern, kann es sich beispielsweise um Anbieter von Leih-Autos, -Fahrrädern oder -Scooter handeln.

Eine Übersicht, über alle Sharing-Anbieter und ihr Angebot erhält man durch eine Anfrage mit dem Parameter `opareaSM=ALL` (siehe Obligatorische Parameter). Die Betreiber können anhand ihrer ID und ihres Bediengebietes gefiltert werden. Dafür sind die im Kapitel Filter beschriebenen Parameter zuständig.

Detailinformation zu einzelnen Sharing-Objekten werden aktuell mit dem CoordInfo-Request angefragt. Siehe dazu Kapitel Anfrage von Detailinformation eines Sharing-Objekts.

17.1 Ansteuerung des OpArea-Request

Der OpArea-Request wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_OPAREA_REQUEST?HTTP-Parameter
```

Beispiele:

Anfrage einer Übersicht aller Car-Sharing Betreiber und ihrer Bediengebiete:

```
http://server:port/virtuellesVerzeichnis/XML_OPAREA_REQUEST?opareaSM=ALL&vehSF=CAR
```

Anfrage von Information zum Betreiber mit der ID 1006 und seinem Bediengebiet 1:

```
http://server:port/virtuellesVerzeichnis/XML_OPAREA_REQUEST?opareaSM=ALL&vehPId1=1006&opAreaId1=1
```

Anfrage der Geometrie des Bediengebiets 1 des Betreibers mit der ID 1006:

```
http://server:port/virtuellesVerzeichnis/XML_OPAREA_REQUEST?operatingZoneGeometry=1&opareaSM=ALL&vehPId1=1006&opAreaId1=1
```

17.2 Obligatorische Parameter

17.2.1 opareaSM

Mit diesem Parameter wird die Suchmethode ausgewählt. Er kann folgende Werte haben:

- `ALL` - alle Objekte
- `RADIAL` - alle Objekte innerhalb eines Radius

Bei der Umkreissuche gilt zu beachten, dass eine Zentrumskoodinate und ein Radius durch den Parameter `coordRadius` angegeben werden muss. Außerdem ist der Parameter `opASR=1` zwingend notwendig.

17.3 Umkreissuche

Die Umkreissuche wird mit dem Parameter `opareaSM=RADIAL` aktiviert.

17.3.1 coordRadius

Mit diesem Parameter werden die Zentrumskoordinate und der Radius für die radiale Suche nach Sharern angegeben. Der Wert setzt sich zusammen aus der x- und der y-Koordinate sowie dem Koordinatenformat (in der Regel `WGS84[dd.ddddd]`). Optional kann ein Radius in Metern angegeben werden. Falls dies nicht der Fall ist, beträgt dieser automatisch 500 m. Die Werte werden durch Doppelpunkt getrennt.

17.3.2 opASR = 1

Dieser Parameter ist aus programmablauftechnischen Gründen zur Aktivierung der Umkreissuche notwendig.

17.4 Filter

17.4.1 opAreaId<Index>

Mit diesem Parameter kann ein Bediengebiet eines Sharing-Anbieters explizit ausgewählt werden. Der Wert des Parameters ist die ID dieses Bediengebiets.

Da die Bediengebiete über keine global eindeutige ID verfügen, muss immer der entsprechende Betreiber durch den Parameter `vehPId<Index>` mit angegeben werden. Die beiden Indices `<Index>` der beiden Parameter entsprechen einander.

Durch mehrfache Verwendung des Parameters, können mehrere Bediengebiete ausgewählt werden.

17.4.2 vehPId<Index>

Mit diesem Parameter kann ein Sharing-Anbieter explizit über dessen ID gewählt werden. Der Index `<Index>` wird fortlaufend, beginnend mit 1, vergeben.

Durch mehrfache Verwendung kann nach mehreren Anbietern gleichzeitig gesucht werden.

17.4.3 vehSF

Mit diesem Parameter wird ein Fahrzeugtyp ausgewählt. Aktuell sind folgende Werte verfügbar:

- `BIKE` - Fahrrad
- `CAR` - PKW
- `MOTORCYCLE` - Motorrad

17.5 **Anfrage von Bediengebieten und abgedeckten Gebieten**

Bei nicht stationären Sharing-Fahrzeugen stellt sich die Frage nach dem Bediengebiet (siehe Abbildung 22). Als Suchmodus wird `operareaSM=ALL` verwendet.

Abbildung 22 - Das Bediengebiet eines Sharing-Objektes

Das Bediengebiet selbst wird über die im Kapitel Filter beschriebenen Parameter ausgewählt. Außerdem sind folgende Parameter notwendig:

17.5.1 **operatingZoneGeometry = 1**

Aktivierung der Ausgabe der umrandenden GIS-Koordinaten von Bediengebieten.

18 StopList-Rquest

Mit dem StopList-Request können alle Haltestellen des Verbundgebietes ausgegeben werden.

Bei großen Netzen empfiehlt sich eine Filterung durch die im Abschnitt Parameter zur Filterung beschriebenen Parameter. Ansonsten kann es zu langen Antwortzeiten oder zum Absturz des Browsers kommen. Zusätzlich gibt es Parameter, die die Ausgabe von ergänzender Information bewirken. Diese sind im Abschnitt Parameter zur Ausgabe zusätzlicher Information beschrieben.

18.1 Ansteuerung des StopList-Requests

Der StopList-Request wird mit HTTP-Parametern gesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_STOPLIST_RE-
QUEST?HTTP-Parameter
```

18.2 Parameter zur Filterung

18.2.1 stopListOMC

Mit diesem Parameter kann eine Gemeindekennziffer (GKZ bzw. OMC) angegeben werden. Es werden ausschließlich Haltestellen gefunden, die innerhalb der angegebenen Gemeinde liegen.

18.2.2 stopListPlaceId

Mit diesem Parameter kann die ID eines Ortes angegeben werden. Es werden ausschließlich Haltestellen gefunden, die in Orten mit der entsprechenden ID liegen. Sinnvoll ist die Kombination von GKZ und ID des Ortes.

18.2.3 stopListSubnetwork

Mit diesem Parameter kann ein Netz angegeben werden. Es werden ausschließlich Haltestellen gefunden, die von Verkehrsmitteln dieses Netzes bedient werden.

18.2.4 fromstop

Mit diesem Parameter kann eine Haltestellen ID angegeben werden, die die Untergrenze eines Intervalls darstellt. Es werden ausschließlich Haltestellen ausgegeben, die in dem Intervall liegen. Die Obergrenze des Intervalls wird durch den Parameter `tostop` angegeben.

18.2.5 tostop

Mit diesem Parameter kann eine Haltestellen ID angegeben werden, die die Obergrenze eines Intervalls darstellt. Es werden ausschließlich Haltestellen ausgegeben, die in dem Intervall liegen. Die Untergrenze des Intervalls wird durch den Parameter `fromstop` angegeben.

18.3 Parameter zur Ausgabe zusätzlicher Information

Bei der Ausgabe von Zusatzinformation muss beachtet werden, dass diese die Performance der Anfrage verschlechtert.

18.3.1 servingLines = 1

Dieser Parameter bewirkt, dass die bedienenden Linien für jede Haltestelle ausgegeben werden.

18.3.2 servingLinesMOTType = 1

Dieser Parameter bewirkt die Ausgabe des Hauptverkehrsmittels jeder Haltestelle. Er kann nicht in Kombination mit dem Parameter `servingLinesMOTTypes = 1` verwendet werden.

18.3.3 servingLinesMOTTypes = 1

Dieser Parameter bewirkt die Ausgabe aller Verkehrsmittel der Haltestellen. Die Verkehrsmittel werden pro Haltestelle als kommagetrennte Liste ausgegeben. Der Parameter kann nicht in Kombination mit dem Parameter `servingLinesMOTType = 1` verwendet werden.

18.3.4 tariffZones = 1

Dieser Parameter bewirkt, dass die Tarifzone für jede Haltestelle ausgegeben wird. Alternativzonen werden durch einen Schrägstrich getrennt mit ausgegeben.

19 AddInfo-Request

Mit dem AddInfo-Request werden aktuelle Meldungen angefragt. Mittels HTTP-Parametern können die Meldungen nach verschiedenen Kriterien gefiltert werden.

Werden keine Filterparameter angegeben, erfolgt die Ausgabe aller Meldungen. Dabei ist zu beachten, dass die Antwortzeiten je nach Menge der eingepflegten Meldungen extrem lange sein können und die Anfrage einen Browserabsturz bewirken kann. Daher empfiehlt es sich nicht, die Meldungen vollständig auszugeben, sondern beispielsweise nur die Meldungen des heutigen Datums.

19.1 Ansteuerung der Anfrage von aktuellen Meldungen

Der AddInfo-Request wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_ADDINFO_REQUEST?HTTP-Parameter
```

19.2 Datums-Filter

19.2.1 filterDateValid

Filtert nach Meldungen, die für einen bestimmten Tag gültig sind. Das Datum wird über die zweistellige Angabe des Tages, gefolgt von der zweistelligen Angabe des Monats und der vierstelligen Angabe des Jahres, angegeben. Als Separator dient das Minuszeichen. Der Parameter kann mehrfach verwendet werden.

19.2.2 filterDateValidComponentsActive = 1

Dieser Parameter ermöglicht eine Alternative zu `filterDateValid`. Tag, Monat und Jahr können getrennt über die Parameter `filterDateValidDay`, filterDate`ValidMonth` und `filterDateValidYear` eingegeben werden.

19.2.3 filterDateValidDay

Dieser Parameter ist Bestandteil des komponentenweisen Datumsfilters, der durch den Parameter `filterDateValidComponentsActive=1` aktiviert wird. Mit ihm gibt man den Tag (zweistellig) an, für den die Meldungen angezeigt werden sollen.

19.2.4 filterDateValidMonth

Dieser Parameter ist Bestandteil des komponentenweisen Datumsfilters, der durch den Parameter `filterDateValidComponentsActive=1` aktiviert wird. Mit ihm gibt man den Monat (zweistellig) an, für den die Meldungen angezeigt werden sollen.

19.2.5 **filterDateValidYear**

Dieser Parameter ist Bestandteil des komponentenweisen Datumsfilters, der durch den Parameter `filterDateValidComponentsActive=1` aktiviert wird. Mit ihm gibt man das Jahr (vierstellig) an, für das die Meldungen angezeigt werden sollen.

19.2.6 **filterPublicationStatus**

Ausschließlich aktuelle Meldungen (*`current`*) oder abgelaufene Meldungen (*`history`*) anzeigen.

19.2.7 **filterValidIntervalStart** und **filterValidIntervalEnd**

Diese Parameter geben ein Datumsintervall an. Es werden ausschließlich Meldungen ausgegeben, die innerhalb des angegebenen Intervalls liegen. Das Start- bzw. End-Datum wird über die zweistellige Angabe des Tages, gefolgt von der zweistelligen Angabe des Monats und der vierstelligen Angabe des Jahres, angegeben. Als Separator dient das Minuszeichen.

19.3 **Orts-Filter**

19.3.1 **filterOMC**

Filtert nach Meldungen, die bestimmte Orte betreffen. Der Wert des Filters ist eine Liste von Gemeindekennziffern (GKZ bzw. OMC), die durch einen Doppelpunkt voneinander getrennt werden. Auf diese Weise können beispielsweise Landkreise modelliert werden.

19.3.2 **filterOMC_PlaceID**

Filtert nach Meldungen, die einen bestimmten Ort betreffen. Der Ort wird durch die Gemeindekennziffer und eine ID des Ortes bestimmt. Die beiden Werte werden durch einen Doppelpunkt getrennt angegeben. Der Parameter kann mehrfach verwendet werden.

19.4 **Linien-, Haltestellen-, Betreiber- und Verkehrsmittel-Filter**

19.4.1 **filterLineNumberIntervalStart** und **filterLineNumberIntervalEnd**

Angabe eines Liniennummer-Intervalls. Es wird nach Meldungen gefiltert, die Linien betreffen, deren Liniennummern (entspricht den letzten 3 Stellen der DIV-Linie) in dem Intervall liegen. Der Wert der Parameter muss größer als Null sein und kann maximal 999 sein, außerdem muss der Wert von `filterLineNumberIntervalEnd` größer sein als der von `filterLineNumberIntervalStart`.

19.4.2 **filterMOTType**

Filtert nach Meldungen, die ein bestimmtes Verkehrsmittel betreffen. Zur Wahl mehrerer Verkehrsmittel, wird der Parameter mehrfach verwendet. Als Wert des

Parameters wird die Verkehrsmittel-ID angegeben. Eine Übersicht über die Verkehrsmittel ist im Anhang Verkehrsmitteltypen zu finden.

19.4.3 **filterPNLineDir**

Filtert nach Meldungen, die Linien betreffen, die den Kriterien „Teilnetz“, „DIVA Linie“ und „Richtung“ entsprechen. Die Kriterien werden durch Doppelpunkt getrennt angegeben. Der Parameter kann mehrfach verwendet werden.

19.4.4 **filterPNLineSub**

Filtert nach Meldungen, die Linien betreffen, die den Kriterien „Teilnetz“, „DIVA Linie“ und „Ergänzung“ entsprechen. Die Kriterien werden durch Doppelpunkt getrennt angegeben. Der Parameter kann mehrfach verwendet werden.

19.4.5 **itdLPxx_selLine**

Filtert nach Meldungen, die eine bestimmte DIVA-Linie betreffen. Als Wert des Parameters wird die DIVA-Linien-Nummer übergeben. Zur Wahl mehrerer DIVA-Linien, wird der Parameter mehrfach verwendet.

19.4.6 **itdLPxx_selOperator**

Filtert nach Meldungen, die die Linien eines bestimmten Betreibers betreffen. Als Wert des Parameters wird die Betreiber-Kennung übergeben. Zur Wahl mehrerer Betreiber, wird der Parameter mehrfach verwendet.

19.4.7 **itdLPxx_selStop**

Filtert nach Meldungen, die eine bestimmte Haltestelle betreffen. Als Wert wird die Haltestellen-ID übergeben. Zur Wahl mehrerer Haltestellen, wird der Parameter mehrfach verwendet.

19.4.8 **line**

Dieser Parameter wählt eine Linie aus. Er kann mehrfach übergeben werden. Sein Wert setzt sich zusammen aus dem Teilnetz, der DIVA Liniennummer, der Ergänzung, der Richtung und dem Fahrplanprojekt, getrennt durch Doppelpunkte. Für die Ergänzung gilt zu beachten, dass der Unterstrich "_" durch ein Leerzeichen " " ersetzt werden muss.

Mit dem Parameter `passedStops=1` können dann Meldungen zurückgegeben werden, die die durchfahrenen Haltestellen der ausgewählten Linien betreffen.

19.5 Filtern nach Meldungstypen und IDs

19.5.1 filterInfoID

Mit dem Parameter kann eine bestimmte Meldung anhand ihrer ID gesucht werden. Die ID wird als Wert des Parameters übergeben. Der Parameter kann zur Suche mehrerer Meldungen mehrfach verwendet werden.

19.5.2 filterInfoType

Filtert nach Meldungen eines bestimmten Typs. Für den ÖV sind folgende Meldungstypen definiert:

- `stopInfo`
- `stopBlocking`
- `lineInfo`
- `lineBlocking`
- `routeInfo`
- `routeBlocking`

Des Weiteren gibt es die Typen:

- `generalInfo`
- `bannerInfo`

Der Parameter kann mehrfach verwendet werden.

19.6 Filter nach Betreiber und Quelle

19.6.1 filterProviderCode

Filtert nach Meldungen, die von einem bestimmten Autor oder Anbieter eingegeben oder zur Verfügung gestellt wurden. Zur Wahl mehrerer Betreiber, wird der Parameter mehrfach verwendet. Als Wert wir der Code des Autors bzw. Anbieters übergeben.

19.6.2 filterSourceSystemName

Filtern nach Meldungen, die von dem als Wert übergebenen Quellsystem eingegeben wurden. Der Parameter kann mehrfach verwendet werden.

20 SystemInfo-Request

Im Folgenden wird das Anfragen von Systeminformation (SystemInfo-Request) erläutert. Es wird die verwendete Programmversion ausgegeben, sowie Information darüber welches Format die Daten haben und wann diese zuletzt aktualisiert wurden.

20.1 Ansteuerung der Anfrage von Systeminformation

Der SystemInfo-Request wird über HTTP-Parameter angesteuert. Die Antwort der Anfrage liegt in einer JSON-Struktur vor:

```
http://server:port/virtuellesVerzeichnis/XML_SYSTEMINFO_REQUEST?HTTP-Parameter
```

20.2 Obligatorische Parameter

Dieser Request hat keine obligatorischen Parameter. Ohne Parameter angefragt, gibt er Information über die verwendete EFA-Version, das Datenformat, den Zeitpunkt der Erstellung der Daten sowie über die Fahrplanperiode aus.

20.3 Optionale Parameter

20.3.1 validityPeriod = 1

Ausgabe aller verfügbaren Fahrplanperioden.

21 Anhang

21.1 Verkehrsmitteltypen

Belegung der Standard Verkehrsmittel-IDs:

ID	Verkehrsmittel
0	Zug
1	S-Bahn
2	U-Bahn
3	Stadtbahn
4	Straßenbahn
5	Stadtbus
6	Regionalbus
7	Schnellbus
8	Seil-/Zahnradbahn
10	AST/Rufbus
11	Schwebebahn
12	Flugzeug
13	Regionalzug (z.B. IRE, RE und RB)
14	Nationaler Zug (z.B. IR und D)
15	Internationale Zug (z.B. IC und EC)
16	Hochgeschwindigkeitszüge (z.B. ICE)
17	Schienenersatzverkehr
18	Schuttlezug
19	Bürgerbus

Kundenspezifische Verkehrsmittel können von der Standardbelegung abweichen.

21.2 **Objekttypen**

Bei der Suche nach Objekten ist manchmal die Angabe des Objekttyps notwendig. Bei der Punktverifikation ist diese optional, kann aber als zusätzliches Filterkriterium verwendet werden. Folgende Objekttypen sind möglich:

Objekttyp	**ID (Dezimalwert)**	**Beschreibung**
-	0	Kein Filter aktiv. Suche im kompletten Suchraum.
PLACE	1	Suche in allen Orten des abgedeckten GIS-Gebietes.
STOP	2	Suche innerhalb aller im abgedeckten GIS-Gebiet erfassten Haltestellen IDs und Alias-Namen für Haltestellen.
STREET	4	Suche innerhalb aller im abgedeckten GIS-Gebiet definierten Straßennamen.
ADDRESS	8	Suche innerhalb aller im abgedeckten GIS-Gebiet definierten Adressen.
CROSSING	16	Suche innerhalb aller im abgedeckten GIS-Gebiet erfassten Kreuzungen.
POI	32	Suche innerhalb aller im abgedeckten GIS-Gebiet definierten IDs und Alias-Namen wichtiger Punkte.
POSTCODE	64	Suche innerhalb aller im abgedeckten GIS-Gebiet erfassten Postleitzahlen.

Die IDs der Objekttypen sind so angelegt, dass sie als Bitmaske verwendet werden können. Auf diese Weise ist es möglich die Objekttypen beliebig zu kombinieren. Soll nach keinem speziellen Objekttyp gefiltert werden, muss 0 angegeben werden. Die Kombination von mehreren Objekttypen kann bei der Punktverifikation eingesetzt werden. Für die Suche nach Betreibern oder Linien macht sie keinen Sinn.

Beispiel: Die Objekttypen zur Punktverifikation werden beispielsweise als Wert des Parameters `anyObjFilter_<usage>` in Form einer Bitmaske zur Filterung des Suchraums übergeben. Soll eine Filterung nach Adressen, Straßen und Haltestellen erfolgen, muss der Filter den Wert 14 (8+4+2) haben.

22 Glossar

22.1 **HTTP-Parameter Makros**

HTTP-Parameter Makros werden in der Konfigurationsdatei des EFAControllers definiert. Dadurch sind sie kundenspezifisch. Sie fassen mehrere HTTP-Parameter in einem einzigen zusammen. Name und Wert des Makros können in der Konfigurationsdatei frei gewählt werden.

Die Verwendung von Makros bietet sich insbesondere dann an, wenn kundenspezifische Kriterien für eine Suchanfrage oder Berechnung vorliegen. Ein Beispiel sind die Kriterien für die Punktsuche. Diese werden in der Regel durch eine Kombination mehrerer Parameter abgebildet und als Makro abgelegt. Ändern sich die Kriterien, wird das Makro in der Konfigurationsdatei angepasst. Eine Anpassung in der Benutzeroberflächen ist nicht notwendig, was insbesondere dann von Vorteil ist, wenn die Kriterien für mehrere gelten. Ein weiterer Vorteil von Makros ist, dass durch die Zusammenfassung mehrerer HTTP-Parameter zu einem Parameter die Anzahl der mitgeschickten Parameter bzw. die URLs für die Requests kürzer werden.