GSLetterNeo vol.145

2020年8月

GTFS 、運行スケジュールをみる

松原 伸人 matubara@sra.co.jp

はじめに

GTFS (General Transit Feed Specification) は、鉄道やバスなど公共交通機関の時刻表や路線図に使用される情報の共通形式です。公共交通機関が公開したフィードを利用することで乗換案内などのアプリケーションを作成できます。今回は運行スケジュールが記載されているcalendar.txtとcalendar_dates.txtを見てみます。calendar.txtとcalendar_dates.txtの定義はGoogleのGTFSリファレンスに記載があります。

calendar.txt | リファレンス | 静的な交通機関 | Google Developers

> calendar_dates.txt では、特定の日付を指定して明示的にサービスを有効にしたり無効にしたりすることができます。2 つの方法で使用できます。
>
> - 推奨される使い方: calendar.txt とともに calendar_dates.txt を使用して、calendar.txt で定義されたデフォルトのサービス パターンの例外を定義します。通常は定期的なサービスとして運営し、特別なイベントや学校行事などのため特定の日に限定して変更が必要な場合に適しています。この場合、サービスの ID には、calendar.service_id を参照する calendar_dates.service_id を使用します。
> - もうひとつの使い方: calendar.txt を使用せず、サービスが利用できる各日付を calendar_dates.txt に指定します。これにより、サービスのかなりのバリエーションが可能になり、通常の毎週のスケジュールのないサービスに対応できます。この場合、service_id はこのファイルで指定する ID です。

たとえば平日ダイヤと土日祝日ダイヤで運行スケジュールが書いてある場合は、calendar.txtに平日ダイヤと土日ダイヤの2レコードが記載され、calendar_dates.txtに祝日の日数分のレコードが記載されていることと想定されます。現在運行している旅程の一覧を取得してみます。

データセットの読み込み

GTFSデータセットのファイル群を指定してデータセットを読み込みます。開発中のGTFS.jsのloadFromFilesで次のように書けます。

```
GTFS.loadFromFiles(files, (gtfs) => {
    // gtfs変数に読み込んだGTFSデータセットが入っています
    console.log(gtfs)
})
```

GTFS.js

calendar.txtはgtfs.calendarByService_idに、calendar_dates.txtはgtfs.calendar_datesByService_idにそれぞれ入ります。今回紹介するプログラムを試す場合は次のURLをwebブラウザで開いてください。

test-collect_service_enabled_trips.html

上記URLをwebブラウザで開き「GTFSデータフォルダを選択」の下の「ファイルを選択」をおしてファイルダイアログでGTFSデータセットフォルダを選択して「選択」を押します。すると上記のloadFromFilesが実行されるようになっています。以降で紹介するコードは全て同じHTML内に記載しています。webブラウザのwebインスペクタでご覧ください。

指定した日の旅程の運行状況をみる

旅程はgtfs.tripByRoute_idに入っています。各旅程レコードのservice_idフィールドに記載のIDから適用されている運行スケジュールが得られます。ここでは推奨されている使い方について、次のようにデータフィールドをチェックして、運行中か運行していないかをみていきます。

(1) サービスを確認する
　*calendar.txt*に*service_id*がなければ運行していない
(2) 運用期間を確認する
　現在の日付が運用期間内でなければ運行していない
(3) 曜日の運行を確認する
*(3-1) weekday*フィールドの値が*1*の場合
*(3-1-1) calendar_dates.txt*に現在の日付に対する記載がなければ運行中
*(3-1-2) exception_type*が*2*の場合は運行していない
*(3-2) weekday*フィールドの値が*0*の場合
*(3-2-1) calendar_dates.txt*に現在の日付に対する記載がなければ運行していない
*(3-2-2) exception_type*が*1*の場合は運行中

旅程の運行状態を出力するコードは次のように書けます。同じコードがtest-collect_service_enabled_trips.htmlのisTripServiceEnabledという名前の関数に書いてあります。isTripServiceEnabled関数は上記の(1)(2)(3)により運行中かどうか判定し、運行中の場合service: true、簡単な理由をmessageに記載して応えます。

```
// (1) サービスを確認する
let calendarData = gtfs.calendarByService_id[trip.service_id]
if (!calendarData) {
    return {
        service: false,
        message: `旅程: ${trip.trip_id}: 運行していない ... calendar.txt に無い ... service_id: ${trip.service_id}`
    }
}
// (2) 運用期間を確認する
let startTime = moment(calendarData.start_date, 'YYYYMMDD').toDate().getTime()
let endTime = moment(calendarData.end_date, 'YYYYMMDD').toDate().getTime()
let t = date.getTime()
if (t < startTime || endTime < t) {
    return {
        service: false,
        message: `旅程: ${trip.trip_id}: 運行していない ... 期間外 ... service_id: ${trip.service_id}`
    }
}
// (3) 曜日の運行を確認する
let dateString = moment(date).format('YYYYMMDD')
let weekday = GTFS.weekdayOfDate(date)
let weekdayField = calendarData[weekday]
let calendar_dates = gtfs.calendar_datesByService_id[trip.service_id]
// (3-1) weekdayフィールドの値が1の場合
if (weekdayField == '1') {
    // (3-1-1) calendar_dates.txtに現在の日付に対する記載がなければ運行中
    if (!calendar_dates) {
        return {
            service: true,
            message: `旅程: ${trip.trip_id}: 運行中 ... service_id: ${trip.service_id}`
        }
    }
    // (3-1-2) exception_typeが2の場合は運行していない
    let result = calendar_dates.filter(calendar_dateData => {
        return calendar_dateData.date == dateString && calendar_dateData.exception_type == '2'
    })
    if (result.length == 1) {
        return {
            service: false,
            message: `旅程: ${trip.trip_id}: 運行していない ... 運休日の指定あり ... service_id: ${trip.service_id}`
        }
```

```
    } else {
        return {
            service: true,
            message: `旅程: ${trip.trip_id}: 運行中 ... service_id: ${trip.service_id}`
        }
    }
// (3-2) weekdayフィールドの値が0の場合
} else if (weekdayField == '0') {
    // (3-2-1) calendar_dates.txtに現在の日付に対する記載がなければ運行していない
    if (!calendar_dates) {
        return {
            service: false,
            message: `旅程: ${trip.trip_id}: 運行していない ... service_id: ${trip.service_id}`
        }
    }
    // (3-2-2) exception_typeが1の場合は運行中
    let result = calendar_dates.filter(calendar_dateData => {
        return calendar_dateData.date == dateString && calendar_dateData.exception_type == '1'
    })
    if (result.length == 1) {
        return {
            service: true,
            message: `旅程: ${trip.trip_id}: 運行中 ... 運行日の指定あり service_id: ${trip.service_id}`
        }
    } else {
        if (!onlyservice)
            console.log()
        return {
            service: false,
            message: `旅程: ${trip.trip_id}: 運行していない ... service_id: ${trip.service_id}`
        }
    }
}
```

GTFS.weekdayOfDate(date)は、指定した日付に該当するフィールド名を応える関数で、上記のGTFS.jsに定義しています。

このHTMLの中では、output関数にデータセットと現在の日付を指定して、全ルート全旅程の運行状態を出力しています。たとえば、次の祝日の9月21日に運行している旅程を得る場合、webインスペクタ上で次のようにデータセットと日付を入力して実行し調べられます。

```
output(gtfsData, new Date('2020-09-21'))
```

outputを実行する時にの3つ目の引数にfalseを指定すると、運行しない旅程とその理由も出力します。

おわりに

今回はGTFSのtrips.txtとcalendar.txtとcalendar_dates.txtから指定の日付の運行状況をみる方法を紹介しました。推奨される方法とは違うcalendar_dates.txtだけで記載されているGTFSの場合も同じようにexception_typeの値が1かどうかで判断できると思います。

GSLetterNeo vol.145
発行日 2020年8月20日
発行者 株式会社 ＳＲＡ 先端技術研究所
編集者 土屋 正人
バックナンバー https://www.sra.co.jp/gsletter/

お問い合わせ
gsneo@sra.co.jp
〒171-8513 東京都豊島区南池袋2-32-8
株式会社SRA