

LA FORCE D'UN TOUT

ALSACE
CHAMPAGNE-ARDENNE
LORRAINE

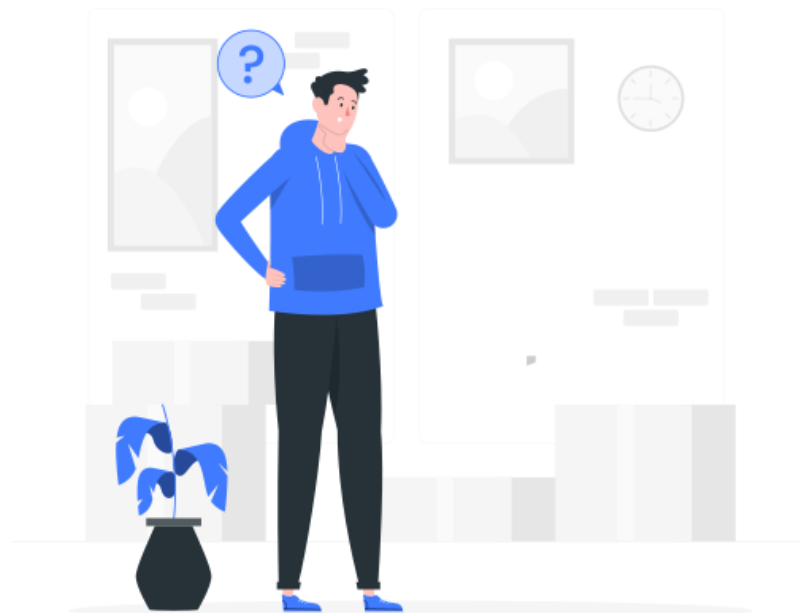


API et données ferroviaires avec R

La Région
Grand Est

Obtenir la liste des trains arrivés en gare de Strasbourg hier.

Obtenir la liste des trains arrivés en gare de Strasbourg hier.



Comment savoir si des données existent pour répondre à notre besoin ?

Obtenir la liste des trains arrivés en gare de Strasbourg hier.



Obtenir la liste des trains arrivés en gare de Strasbourg hier.

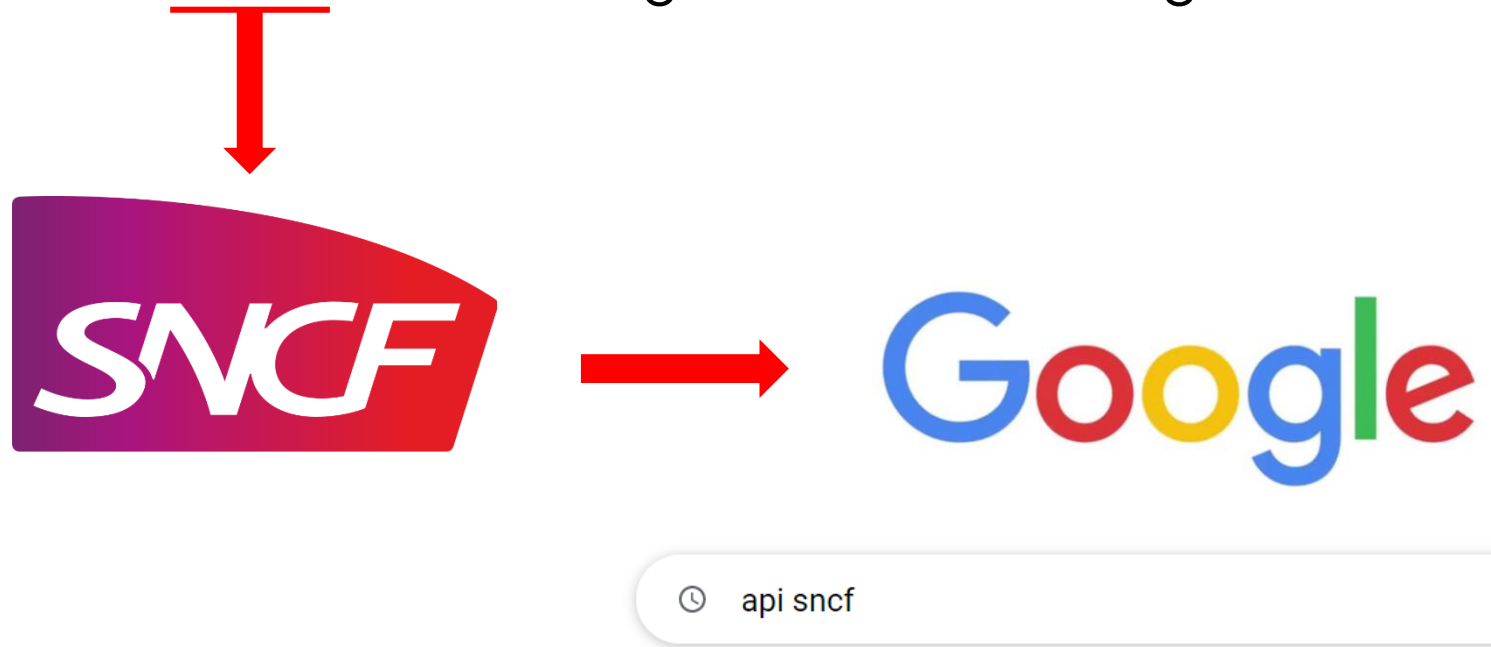


1) On regarde sur leur Open Data
(<https://ressources.data.sncf.com/pages/accueil>)

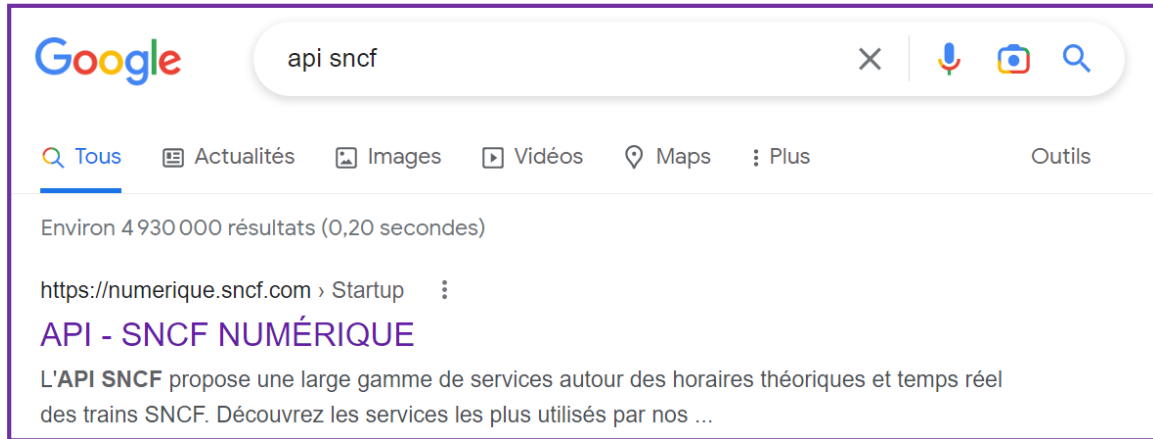
2) On vérifie si les données proposées permettent de répondre à notre besoin

3) On découvre qu'il existe une API

Obtenir la liste des trains arrivés en gare de Strasbourg hier.



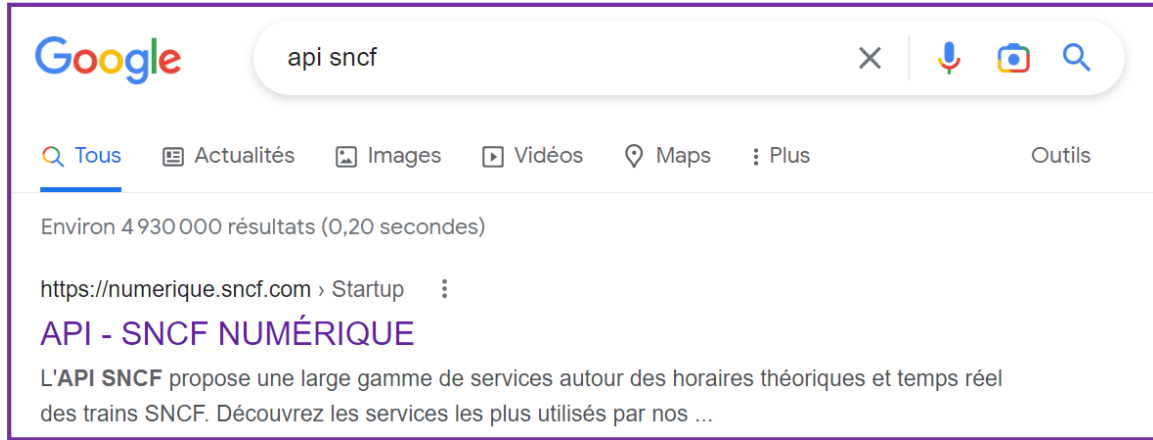
Trouver la source de données



The screenshot shows a Google search interface. The search bar contains the text "api sncf". Below the search bar, there are navigation links for "Tous", "Actualités", "Images", "Vidéos", "Maps", and "Outils". The search results indicate approximately 4,930,000 results found in 0.20 seconds. The first result is from the URL "https://numerique.sncf.com > Startup" and is titled "API - SNCF NUMÉRIQUE". The description of the result states: "L'API SNCF propose une large gamme de services autour des horaires théoriques et temps réel des trains SNCF. Découvrez les services les plus utilisés par nos ...".

1

Trouver la source de données



A screenshot of a Google search for "api sncf". The search bar contains "api sncf" and the Google logo is visible. Below the search bar, there are navigation options: "Tous", "Actualités", "Images", "Vidéos", "Maps", and "Outils". The search results show "Environ 4 930 000 résultats (0,20 secondes)". The first result is from "https://numerique.sncf.com > Startup" and is titled "API - SNCF NUMÉRIQUE". The description of the result states: "L'API SNCF propose une large gamme de services autour des horaires théoriques et temps réel des trains SNCF. Découvrez les services les plus utilisés par nos ...".

1



A screenshot of the SNCF NUMÉRIQUE website. The header includes the site name "SNCF NUMÉRIQUE" and navigation links: "Actus & agenda", "Startup", "Store", "Ressources", and "Ambition". The main content area has a breadcrumb trail "Accueil > Startup > API" and a large heading "API SNCF". Below the heading, there is a paragraph: "Accédez aux itinéraires et horaires temps réel des trains SNCF pour inventer les nouveaux services de mobilité." followed by another paragraph: "Inspirez-vous de plusieurs de nos partenaires et boostez vos services de mobilité avec l'API SNCF : comparateur de voyage, horaires temps réel, chatbot, information sur sites touristiques, robotique d'interaction et bien d'autres cas d'usage." At the bottom, there are two buttons: "Commencez à utiliser l'API" and "Consulter la documentation".

2

Trouver la source de données

A screenshot of a Google search for "api sncf". The search bar shows "api sncf" with a search icon. Below the search bar, there are navigation options: "Tous", "Actualités", "Images", "Vidéos", "Maps", and "Outils". The search results show "Environ 4 930 000 résultats (0,20 secondes)". The first result is "https://numerique.sncf.com > Startup > API - SNCF NUMÉRIQUE". The description for this result is "L'API SNCF propose une large gamme de services autour des horaires théoriques et temps réel des trains SNCF. Découvrez les services les plus utilisés par nos ...".

1

Demandez votre clé d'accès à l'API SNCF

Accédez aux itinéraires et horaires temps réel des trains SNCF pour inventer les nouveaux services de mobilité.

Avant de vous inscrire, nous vous conseillons de consulter la FAQ.

[Consultez la FAQ >](#)

Prénom*	Nom*
<input type="text"/>	<input type="text"/>
E-mail*	Organisation / Entreprise
<input type="text"/>	<input type="text"/>
Site Web	
<input type="text" value="https://"/>	

J'ai lu et j'accepte les conditions d'utilisation de l'API SNCF*

3

A screenshot of the SNCF NUMÉRIQUE website. The header includes "SNCF NUMÉRIQUE" and navigation links: "Actus & agenda", "Startup", "Store", "Ressources", and "Ambition". The breadcrumb trail is "Accueil > Startup > API". The main heading is "API SNCF". The text below reads: "Accédez aux itinéraires et horaires temps réel des trains SNCF pour inventer les nouveaux services de mobilité." and "Inspirez-vous de plusieurs de nos partenaires et boostez vos services de mobilité avec l'API SNCF : comparateur de voyage, horaires temps réel, chatbot, information sur sites touristiques, robotique d'interaction et bien d'autres cas d'usage." At the bottom, there are two buttons: "Commencez à utiliser l'API" and "Consulter la documentation".

2

Création d'un compte pour utiliser l'API

Demandez votre clé d'accès à l'API SNCF

Accédez aux itinéraires et horaires temps réel des trains SNCF pour inventer les nouveaux services de mobilité.

Avant de vous inscrire, nous vous conseillons de consulter la FAQ.

[Consultez la FAQ >](#)

Prénom*

Nom*

E-mail*

Organisation / Entreprise

Site Web

J'ai lu et j'accepte les conditions d'utilisation de l'API SNCF*

Envoyer

Pour les développeurs

Gratuit

- 150 000 requêtes/mois (5000 requêtes/jour)
- 20 requêtes/min pour les horaires temps réel de SNCF Transilien (Île-de-France) disponibles via l'API Transilien
- Support et documentation
- Créer et tester des requêtes : playground.navitia.io

Utiliser l'API

Pour les entreprises

Selon vos besoins

- Plan sur mesure
- Support prioritaire et documentation

Contactez-nous

Merci de vous être inscrit à l'API SNCF. Voici votre clé d'authentification :

5a65431g-b045-4e1d-5zaf-6f325377f068|  **Token = clé d'accès à l'API**

Comment vous authentifier à l'API ?

Accédez à l'API : <https://api.sncf.com/v1>

Username : copiez puis collez votre clé

Password : laissez le champ vide

ou copiez votre clé directement dans l'URL.

Comment utiliser l'API ?

Quelques exemples :

Le détail des modes de transport couvert par l'API

GET https://api.sncf.com/v1/coverage/sncf/commercial_modes

Itinéraires entre Paris et Lyon

GET

<https://api.sncf.com/v1/coverage/sncf/journeys?from=admin:fr:75056&to=admin:fr:69123&atetime=20221004T122308>

Prochains départs à Montparnasse

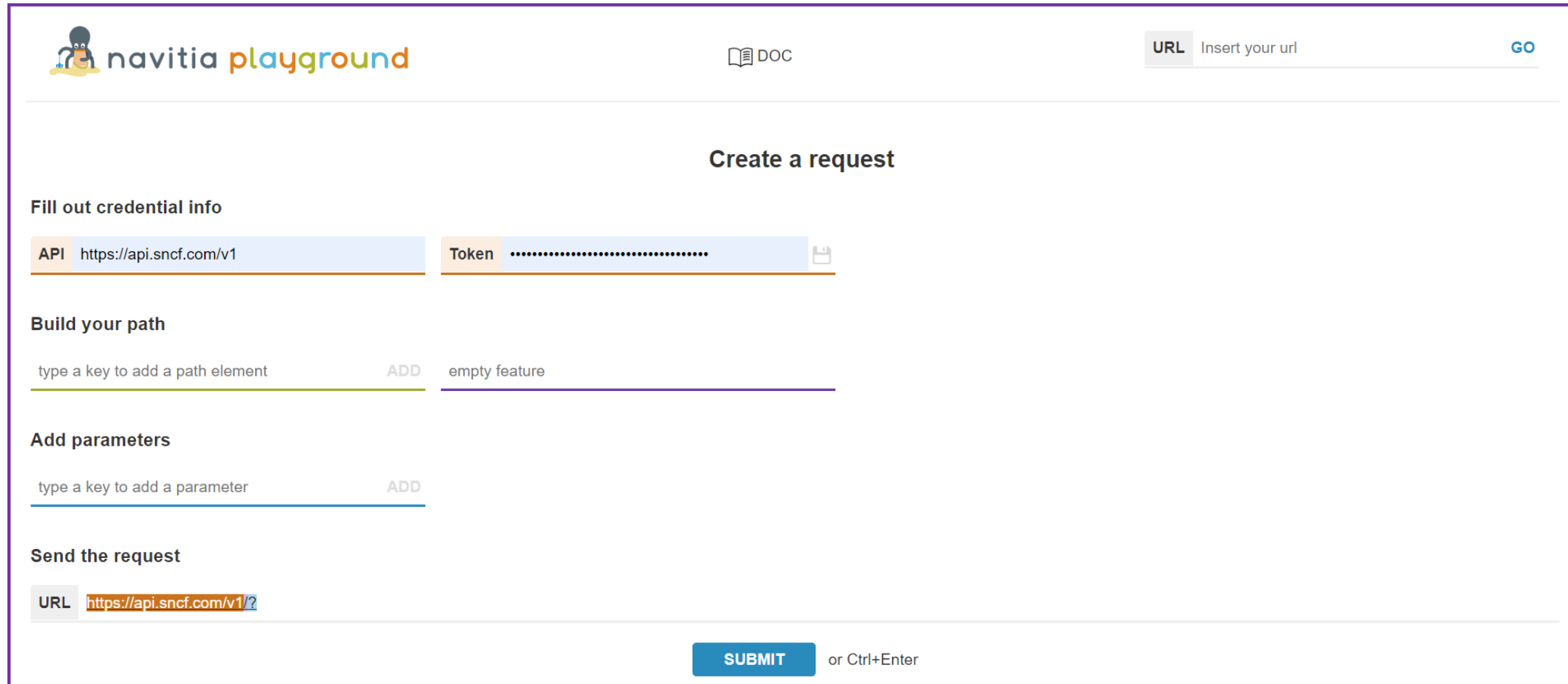
GET

https://api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87391003/departures?datetime=20221004T122308

- [Documentation](#)
- [Support](#)

L'équipe API SNCF

Soit on est pas à l'aise avec le code et on utilise le « Playground » disponible.
(<https://playground.navitia.io/play.html>)



The screenshot shows the Navitia Playground interface. At the top left is the logo 'navitia playground'. To the right is a 'DOC' icon and a 'URL' input field with the placeholder 'Insert your url' and a 'GO' button. The main heading is 'Create a request'. Below this, there are four sections: 'Fill out credential info' with 'API' (containing 'https://api.sncf.com/v1') and 'Token' (containing dots) fields; 'Build your path' with a 'type a key to add a path element' field and an 'ADD' button, and an 'empty feature' field; 'Add parameters' with a 'type a key to add a parameter' field and an 'ADD' button; and 'Send the request' with a 'URL' field containing 'https://api.sncf.com/v1/?'. At the bottom center is a 'SUBMIT' button with the text 'or Ctrl+Enter' next to it.

Soit on est à l'aise avec le code et on choisi son environnement de travail.

Dans les deux cas, il est important de **comprendre comment fonctionne l'API.**

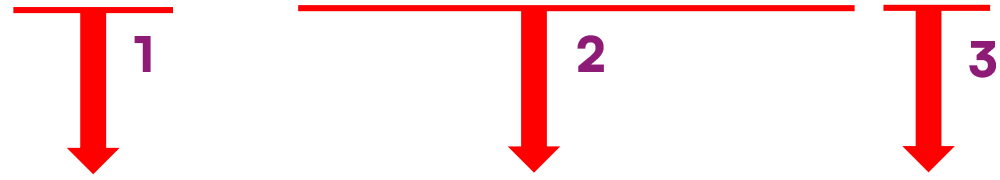
Sans consulter la documentation, il est impossible de créer un requête correcte.

Accès à la documentation : <https://doc.navitia.io/#getting-started>

Pour cet exemple, le logiciel R sera utilisé avec l'environnement R Studio.



Obtenir la liste des trains arrivés en gare de Strasbourg hier.



Notre requête doit spécifier ces trois éléments.

Construction de la requête :

"https://5a65431g-b045-4e1d-5zaf6f325377f068@api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87212027/arrivals?count=1000&from_datetime=20230130T000000&until_datetime=20230131T000000&"

Token

2

1

3

Token : Se place au milieu de l'adresse du serveur avec un @

<https://api.sncf.com/v1/> devient <https://5a65431g-b045-4e1d-5zaf6f325377f068@api.sncf.com/v1/>

1 — Arrivées : « arrivals » plus une spécification du nombre maximum de retours

[arrivals?count=1000](#)

2 — Gare : Paramètre « stop_areas/ » mentionné « stop_area:SNCF: » plus le code UIC de la gare
[stop_areas/stop_area:SNCF:87212027stop_area:SNCF:87212027](#)

3 — Date : Spécifié par les paramètres « from_datetime » et « until_datetime »

[&from_datetime=20230130T000000&until_datetime=20230131T000000&](#)

Dans la barre URL ► Réception d'un fichier JSON.

```
{
  "pagination": {
    "start_page": 0,
    "items_on_page": 201,
    "items_per_page": 1000,
    "total_result": 201
  },
  "links": [
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/stop_points/{stop_point.id}",
      "type": "stop_point",
      "rel": "stop_points",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/commercial_modes/{commercial_modes.id}",
      "type": "commercial_modes",
      "rel": "commercial_modes",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/stop_areas/{stop_area.id}",
      "type": "stop_area",
      "rel": "stop_areas",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/physical_modes/{physical_modes.id}",
      "type": "physical_modes",
      "rel": "physical_modes",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/disruptions/{disruptions.id}",
      "type": "disruptions",
      "rel": "disruptions",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/routes/{route.id}",
      "type": "route",
      "rel": "routes",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/physical_modes/{physical_mode.id}",
      "type": "physical_mode",
      "rel": "physical_modes",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/commercial_modes/{commercial_mode.id}",
      "type": "commercial_mode",
      "rel": "commercial_modes",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/disruptions/{disruption.id}",
      "type": "disruption",
      "rel": "disruptions",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/vehicle_journeys/{vehicle_journey.id}",
      "type": "vehicle_journey",
      "rel": "vehicle_journeys",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/lines/{line.id}",
      "type": "line",
      "rel": "lines",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/trips/{trip.id}",
      "type": "trip",
      "rel": "trips",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/networks/{network.id}",
      "type": "network",
      "rel": "networks",
      "templated": true
    },
    {
      "href": "https://api.sncf.com/v1/coverage/sncf/stop_areas/SNCF:87212027/arrivals?count=1000&from_datetime=20230108T000000&until_datetime=20230109T000000",
      "type": "first",
      "templated": false
    }
  ],
  "disruptions": [
    {
      "status": "past",
      "disruption_id": "085da438-b51b-411d-82df-16e9eb3d31",
      "severity": {
        "color": "#000000",
        "priority": 42,
        "name": "additional service",
        "effect": "ADDITIONAL_SERVICE",
        "impact_id": "085da438-b51b-411d-82df-16e9eb3d31",
        "application_periods": [
          {
            "begin": "20230108T103700",
            "end": "20230108T121000"
          }
        ],
        "updated_at": "20230109T115235",
        "uri": "085da438-b51b-411d-82df-16e9eb3d31",
        "impacted_objects": [
          {
            "impacted_stops": [
              {
                "amended_arrival_time": "103700",
                "stop_point": {
                  "name": "Niederbronn-les-Bains",
                  "links": [],
                  "coord": {
                    "lat": "48.952385",
                    "lon": "7.63409"
                  },
                  "label": "Niederbronn-les-Bains (Niederbronn-les-Bains)",
                  "equipments": []
                },
                "id": "stop_point:SNCF:87213256:Coach",
                "stop_time_effect": "added",
                "departure_status": "added",
                "is_detour": false,
                "amended_departure_time": "103700",
                "cause": "",
                "arrival_status": "added",
                "amended_arrival_time": "104500",
                "stop_point": {
                  "name": "Reichshoffen",
                  "links": [],
                  "coord": {
                    "lat": "48.930925",
                    "lon": "7.658072"
                  },
                  "label": "Reichshoffen (Reichshoffen)",
                  "equipments": []
                },
                "id": "stop_point:SNCF:87213249:Coach",
                "stop_time_effect": "added",
                "departure_status": "added",
                "is_detour": false,
                "amended_departure_time": "104500",
                "cause": "",
                "arrival_status": "added",
                "amended_arrival_time": "105200",
                "stop_point": {
                  "name": "Gundershoffen",
                  "links": [],
                  "coord": {
                    "lat": "48.905261",
                    "lon": "7.653535"
                  },
                  "label": "Gundershoffen (Gundershoffen)",
                  "equipments": []
                },
                "id": "stop_point:SNCF:87213223:Coach",
                "stop_time_effect": "added",
                "departure_status": "added",
                "is_detour": false,
                "amended_departure_time": "105200",
                "cause": "",
                "arrival_status": "added",
                "amended_arrival_time": "105900",
                "stop_point": {
                  "name": "Mertzwiller",
                  "links": [],
                  "coord": {
                    "lat": "48.872056",
                    "lon": "7.678244"
                  },
                  "label": "Mertzwiller (Mertzwiller)",
                  "equipments": []
                },
                "id": "stop_point:SNCF:87213207:Coach",
                "stop_time_effect": "added",
                "departure_status": "added",
                "is_detour": false,
                "amended_departure_time": "105900",
                "cause": "",
                "arrival_status": "added",
                "amended_arrival_time": "110900",
                "stop_point": {
                  "name": "Schweighouse-sur-Moder",
                  "links": [],
                  "coord": {
                    "lat": "48.828677",
                    "lon": "7.742564"
                  },
                  "label": "Schweighouse-sur-Moder (Schweighouse-sur-Moder)",
                  "equipments": []
                },
                "id": "stop_point:SNCF:87213108:Coach",
                "stop_time_effect": "added",
                "departure_status": "added",
                "is_detour": false,
                "amended_departure_time": "110900",
                "cause": "",
                "arrival_status": "added",
                "amended_arrival_time": "111800",
                "stop_point": {
                  "name": "Haguenau",
                  "links": [],
                  "coord": {
                    "lat": "48.813486",
                    "lon": "7.782757"
                  },
                  "label": "Haguenau (Haguenau)",
                  "equipments": []
                },
                "id": "stop_point:SNCF:87213058:Coach",
                "stop_time_effect": "added",
                "departure_status": "added",
                "is_detour": false,
                "amended_departure_time": "111800",
                "cause": "",
                "arrival_status": "added",
                "amended_arrival_time": "111800",
                "stop_point": {
                  "name": "Haguenau",
                  "links": [],
                  "coord": {
                    "lat": "48.813486",
                    "lon": "7.782757"
                  },
                  "label": "Haguenau (Haguenau)",
                  "equipments": []
                }
              ]
            }
          }
        ]
      }
    }
  ]
}
```

Requêter l'API via R


Pour manipuler des données dans R, il peut être pratique de faire appel à des packages dédiés.

Nous allons utiliser ici la collection de packages proposée au sein du Tidyverse. Il s'agit d'un ensemble de packages R interopérables abordant un très grand nombre d'opérations : import/export de données, visualisation, manipulation des tableaux de données, manipulation de variables, extraction de données du Web, programmation, ...

```
install.packages("tidyverse")  
library("tidyverse")
```



Dans R, l'objectif est de récupérer le fichier JSON et de le traiter pour sortir une liste propre.

Pour cela, nous allons utiliser le package JsonLite. 

On installe le package avec la fonction `install.packages()` puis on le charge avec la fonction `library()` :

```
install.packages("jsonlite")  
library("jsonlite")
```

Nous allons ensuite, grâce à la fonction `fromJSON()`, récupérer les données. On nomme les données « json_arrivees » :

```
json_arrivees <- fromJSON("https://5a65431g-b045-4e1d-  
5zaf6f325377f068@api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87212027/arrivals?count=1000  
&from_datetime=20230130T000000&until_datetime=20230131T000000")
```


Les données sont à présent chargées dans R.

On utilise la fonction `view()` pour voir les données :

`View(json_arrivees)`

On constate que les données sont présentées sous la forme d'une liste.

Dans la section « arrivals », on peut voir le nombre de trains.

C'est dans cette section là que nous devons aller chercher les informations qui nous intéressent.

Name	Type	Value
▼ json_arrivees	list [8]	List of length 8
▶ pagination	list [4]	List of length 4
▶ links	list [14 x 4] (S3: data.frame)	A data.frame with 14 rows and 4 columns
▶ disruptions	list [15 x 13] (S3: data.frame)	A data.frame with 15 rows and 13 columns
notes	list [0]	List of length 0
▶ arrivals	list [201 x 5] (S3: data.frame)	A data.frame with 201 rows and 5 columns
▶ feed_publishers	list [2 x 4] (S3: data.frame)	A data.frame with 2 rows and 4 columns
▶ context	list [2]	List of length 2
exceptions	list [0]	List of length 0

Requêter l'API via R

En recherchant dans les données, on se rend compte que les informations que l'on souhaite sont toutes présentes dans la section « arrivals ». Les données générales se situent dans le sous-groupe **display_informations**

Les données horaires se situent dans la section **stop_date_time**

- ▼ arrivals
 - ▶ display_informations
 - ▶ stop_point
 - ▶ route
 - ▶ links
 - ▶ stop_date_time

Pour **display_informations**

```
# Transformation des données en tableau
infos <- as.data.frame(as.list(json_arrivees$arrivals)$display_informations)
# Sélection des variables utiles et création d'un identifiant pour jointure
infos <- arriv_infos %>% select(Reseau = network, Type = physical_mode, Direction =
direction, Ligne = name, Num = trip_short_name) %>% mutate(id = rownames(arriv_infos))
```

Pour **stop_date_time**

```
# Transformation des données en tableau
horaire <- as.data.frame(as.list(json_arrivees$arrivals)$stop_date_time)
# Sélection des variables utiles et création d'un identifiant pour jointure
horaire <- arriv_hor %>% select(Arrivee = arrival_date_time) %>% mutate(id =
rownames(arriv_hor))
```

Requêter l'API via R

On fusionne les deux jeux de données grâce à la fonction `merge()`

```
# Jointure des données générales et des données horaires  
arrivees <- merge(infos, horaire, by = "id")
```

On obtient ceci :

id	Reseau	Type	Direction	Ligne	Num	Arrivee_reelle
1	TGV INOUI	Train grande vitesse	Montpellier Saint-Roch (Montpellier)	Montpellier Saint-Roch - Metz	5521	20230108T065500
2	TER	TER / Intercités	Strasbourg (Strasbourg)	Strasbourg - Offenbourg	87406	20230108T070600
3	TER	TER / Intercités	Strasbourg (Strasbourg)	Strasbourg - Haguenau	830507	20230108T072000
4	TER	TER / Intercités	Strasbourg (Strasbourg)	Strasbourg - Lauterbourg - Woerth	830747	20230108T074000
5	TER	TER / Intercités	Strasbourg (Strasbourg)	STRASBOURG - BÂLE	96206	20230108T074400
6	TER	TER / Intercités	Sélestat (Sélestat)	Sarrebouurg - Strasbourg - Sélestat	832009	20230108T080300
7	TER	TER / Intercités	Strasbourg (Strasbourg)	Strasbourg - Offenbourg	87404	20230108T080400
8	TER	TER / Intercités	Strasbourg (Strasbourg)	STRASBOURG-MULHOUSE	832352	20230108T080430
9	TGV INOUI	Train grande vitesse	Paris Est (Paris)	Colmar - Paris Est	2352	20230108T080900
10	TGV INOUI	Train grande vitesse	Paris Est (Paris)	Freiburg (Breisgau) Hbf - Paris Est	9408	20230108T081300
11	TER	TER / Intercités	Strasbourg (Strasbourg)	Strasbourg - Haguenau	830511	20230108T081830
12	TER	TER / Intercités	Strasbourg (Strasbourg)	STRASBOURG - SELESTAT Via MOLSHEIM	831711	20230108T082400
13	TER	TER / Intercités	Strasbourg (Strasbourg)	Strasbourg - Rothau	831879	20230108T083600
14	TER	TER / Intercités	Strasbourg (Strasbourg)	Strasbourg - Sarrebruck	86389	20230108T083700
15	TER	TER / Intercités	Strasbourg (Strasbourg)	STRASBOURG - BÂLE	96210	20230108T083900
16	DB SNCF	Train grande vitesse	Stuttgart Hbf (Stuttgart)	Stuttgart Hbf - Paris Est	9571	20230108T084100
17	TER	TER / Intercités	Strasbourg (Strasbourg)	STRASBOURG - WISSEMBOURG	830635	20230108T084300
18	DB SNCF	Train grande vitesse	Paris Est (Paris)	Francfort sur le Main - Paris Est	9568	20230108T084700
19	TER	TER / Intercités	Saverne (Saverne)	Sarrebouurg - Strasbourg - Sélestat	832012	20230108T084900
20	TGV INOUI	Train grande vitesse	Marseille Saint-Charles (Marseille)	Nice-Villa - Nancy	9877	20230108T090000

Requêter l'API via R

Nous obtenons un **tableau avec l'ensemble des informations utiles sur les arrivées des trains en gare de Strasbourg pour la journée d'hier** :

- Numéro de train
- Type (TER, TGV)
- Ligne
- Direction
- Heure d'arrivée

On peut à présent **exporter les données** sous la forme que l'on souhaite.

Exemple pour exporter les données sous la forme d'un CSV

```
# Ecriture du CSV
```

```
write.csv(arrivees, 'C:/Users/boucherec/Documents/Trains/arrivees_Strasbourg.csv')
```

Requêter l'API via R

```
install.packages("jsonlite")
library("jsonlite")
install.packages("tidyverse")
library("tidyverse")
install.packages("lubridate")
library("lubridate")

json_arrivees <- fromJSON("https://5a65431g-b045-4e1d-
5zaf6f325377f068@api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87212027/arrivals?count=1000&from_datetime=20230130T000000&until_datetime=20230131T000000")

View(json_arrivees)

# Transformation des données en tableau
arriv_infos <- as.data.frame(as.list(json_arrivees$arrivals)$display_informations)
# Sélection des variables utiles et création d'un identifiant pour jointure
infos <- arriv_infos %>% select(Reseau = network, Type = physical_mode, Direction = direction, Ligne = name, Num = trip_short_name) %>% mutate(id =
rownames(arriv_infos))

# Transformation des données en tableau
arriv_hor <- as.data.frame(as.list(json_arrivees$arrivals)$stop_date_time)
# Sélection des variables utiles et création d'un identifiant pour jointure
horaire <- arriv_hor %>% select(Arrivee = arrival_date_time) %>% mutate(id = rownames(arriv_hor))

# Jointure des données générales et des données horaires
arrivees <- merge(infos, horaire, by = "id")

# Ecriture du CSV
write.csv(arrivees, 'C:/Users/boucherec/Documents/Trains/arrivees_Strasbourg.csv')
```

Requêter l'API via R

```
install.packages("jsonlite")
library("jsonlite")
install.packages("tidyverse")
library("tidyverse")
install.packages("lubridate")
library("lubridate")
```

Appel des packages

```
json_arrivees <- fromJSON("https://5a65431g-b045-4e1d-5zaf6f325377f068@api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87212027/arrivals?count=1000&from_datetime=20230130T000000&until_datetime=20230131T000000")
```

```
View(json_arrivees)
```

```
# Transformation des données en tableau
```

```
arriv_infos <- as.data.frame(as.list(json_arrivees$arrivals)$display_informations)
```

```
# Sélection des variables utiles et création d'un identifiant pour jointure
```

```
infos <- arriv_infos %>% select(Reseau = network, Type = physical_mode, Direction = direction, Ligne = name, Num = trip_short_name) %>% mutate(id = rownames(arriv_infos))
```

```
# Transformation des données en tableau
```

```
arriv_hor <- as.data.frame(as.list(json_arrivees$arrivals)$stop_date_time)
```

```
# Sélection des variables utiles et création d'un identifiant pour jointure
```

```
horaire <- arriv_hor %>% select(Arrivee = arrival_date_time) %>% mutate(id = rownames(arriv_hor))
```

```
# Jointure des données générales et des données horaires
```

```
arrivees <- merge(infos, horaire, by = "id")
```

```
# Ecriture du CSV
```

```
write.csv(arrivees, 'C:/Users/boucherec/Documents/Trains/arrivees_Strasbourg.csv')
```

Requêter l'API via R

```
install.packages("jsonlite")
library("jsonlite")
install.packages("tidyverse")
library("tidyverse")
install.packages("lubridate")
library("lubridate")
```

Requête API

```
json_arrivees <- fromJSON("https://5a65431g-b045-4e1d-5zaf6f325377f068@api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87212027/arrivals?count=1000&from_datetime=20230130T000000&until_datetime=20230131T000000")
```

```
View(json_arrivees)
```

```
# Transformation des données en tableau
```

```
arriv_infos <- as.data.frame(as.list(json_arrivees$arrivals)$display_informations)
```

```
# Sélection des variables utiles et création d'un identifiant pour jointure
```

```
infos <- arriv_infos %>% select(Reseau = network, Type = physical_mode, Direction = direction, Ligne = name, Num = trip_short_name) %>% mutate(id = rownames(arriv_infos))
```

```
# Transformation des données en tableau
```

```
arriv_hor <- as.data.frame(as.list(json_arrivees$arrivals)$stop_date_time)
```

```
# Sélection des variables utiles et création d'un identifiant pour jointure
```

```
horaire <- arriv_hor %>% select(Arrivee = arrival_date_time) %>% mutate(id = rownames(arriv_hor))
```

```
# Jointure des données générales et des données horaires
```

```
arrivees <- merge(infos, horaire, by = "id")
```

```
# Ecriture du CSV
```

```
write.csv(arrivees, 'C:/Users/boucherec/Documents/Trains/arrivees_Strasbourg.csv')
```

Requêter l'API via R

```
install.packages("jsonlite")
library("jsonlite")
install.packages("tidyverse")
library("tidyverse")
install.packages("lubridate")
library("lubridate")
```

```
json_arrivees <- fromJSON("https://5a65431g-b045-4e1d-5zaf6f325377f068@api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87212027/arrivals?count=1000&from_datetime=20230130T000000&until_datetime=20230131T000000")
```

```
View(json_arrivees)
```

Transformation des données transmises par l'API en tableau propre et lisible

```
# Transformation des données en tableau
arriv_infos <- as.data.frame(as.list(json_arrivees$arrivals)$display_informations)
# Sélection des variables utiles et création d'un identifiant pour jointure
infos <- arriv_infos %>% select(Reseau = network, Type = physical_mode, Direction = direction, Ligne = name, Num = trip_short_name) %>% mutate(id = rownames(arriv_infos))

# Transformation des données en tableau
arriv_hor <- as.data.frame(as.list(json_arrivees$arrivals)$stop_date_time)
# Sélection des variables utiles et création d'un identifiant pour jointure
horaire <- arriv_hor %>% select(Arrivee = arrival_date_time) %>% mutate(id = rownames(arriv_hor))

# Jointure des données générales et des données horaires
arrivees <- merge(infos, horaire, by = "id")

# Ecriture du CSV
write.csv(arrivees, 'C:/Users/boucherec/Documents/Trains/arrivees_Strasbourg.csv')
```


Requêter l'API via R

```
install.packages("jsonlite")
library("jsonlite")
install.packages("tidyverse")
library("tidyverse")
install.packages("lubridate")
library("lubridate")

json_arrivees <- fromJSON("https://5a65431g-b045-4e1d-5zaf6f325377f068@api.sncf.com/v1/coverage/sncf/stop_areas/stop_area:SNCF:87212027/arrivals?count=1000&from_datetime=20230130T000000&until_datetime=20230131T000000")

View(json_arrivees)

# Transformation des données en tableau
arriv_infos <- as.data.frame(as.list(json_arrivees$arrivals)$display_informations)
# Sélection des variables utiles et création d'un identifiant pour jointure
infos <- arriv_infos %>% select(Reseau = network, Type = physical_mode, Direction = direction, Ligne = name, Num = trip_short_name) %>% mutate(id = rownames(arriv_infos))

# Transformation des données en tableau
arriv_hor <- as.data.frame(as.list(json_arrivees$arrivals)$stop_date_time)
# Sélection des variables utiles et création d'un identifiant pour jointure
horaire <- arriv_hor %>% select(Arrivee = arrival_date_time) %>% mutate(id = rownames(arriv_hor))

# Jointure des données générales et des données horaires
arrivees <- merge(infos, horaire, by = "id")

# Ecriture du CSV
write.csv(arrivees, 'C:/Users/boucherec/Documents/Trains/arrivees_Strasbourg.csv')
```

Export des données au format CSV

- Automatisation de la date grâce à des variables
 - From_datetime = hier à minuit
 - Until_datetime = aujourd'hui à minuit
- Génération de la requête pour plusieurs gares grâce à une liste
 - stop_area -> liste avec les codes UIC des gares souhaitées
- Adaptation du script pour les départs
Arrivals > Departures
- Consultation de la documentation pour découvrir l'ensemble des possibilités

La Région Grand Est et les AOM du territoire mettent à **disposition du public les données du Système d'Information Multimodale Fluo** et permettent leur réutilisation en vue de favoriser le développement de services à l'utilisateur et l'innovation sur le territoire.

Quatre possibilités d'intégration de ces données sont proposées :

- La mise à disposition du module de calcul d'itinéraires en marque grise
- La mise à disposition de modules intégrables (recherche horaire, recherche d'itinéraire, plans à proximité) en marque grise ou blanche
- La mise à disposition des autres données mobilité du référentiel et du calculateur d'itinéraires, accessibles par une API
- La mise à disposition des données d'offres théoriques des réseaux de transport public, sous la forme de fichier bruts (NEPTUNE, GTFS)

Infos : <https://www.fluo.eu/fr/developpeur-api/77>



LA FORCE D'UN TOUT

ALSACE
CHAMPAGNE-ARDENNE
LORRAINE



API et données ferroviaires avec R

La Région
Grand Est