OAG®
connecting the world of travel

schedules
essential data you can trust

analytics
advanced intelligence to drive your business

flightview
flight status information you can act on

OAG FV Flight Notifications for Business (FNB) Integration Guide

Table of Contents

1. Introduction

With OAG flightview's Flight Notifications for Business (FNB), a customer collects and registers a set of flights with the OAG flightview system on a rolling basis. FNB monitors these flights and sends notifications to the customer as flight updates occur.

OAG flightview's FNB is suited for customers who prefer to receive updates on a subset of flights they or their customers care about, rather than obtaining updates on all in-air or near-term fights and filtering this comprehensive list themselves (see OAG flightview's FV Data Batch and Changes Products for updates on all flights).

The flight alerts provided by FNB are based on OAG flightview's comprehensive real-time flight information, which contains data from hundreds of sources, globally. Such sources include schedule data, airline direct feeds, in-air position feeds or radar data, and airport feeds. Data from all of these sources is normalized and aggregated to provide customers with the best picture of what is going on in the skies for flights across the globe.

When flight updates occur, FNB pushes notifications to the customer via an HTTP(S) request with the notification information in JSON format. This information is comprised of 3 categories: (1) the event that triggered the notification, (2) complete and current information about the flight, and (3) all changes that have occurred to it (not just those associated with the notification). The customer pulls the appropriate information from the notification and passes it to its planning and operations internal systems or a notification engine that pushes alerts to their users/employees.

Flight notifications can be sent to a business customer who then forwards these alerts to their end users or OAG flightview can send the alerts to the end users on behalf of the business customer.

Flight Notifications to a business customer

FNB sends notifications for the flights that the business customer registers. The customer, in turn, pushes alerts to its users concerning the recent flight changes or uses this information to make operational adjustments to its own travel services.

Flight Notifications to end users

As of release 1.8, users can also directly receive email or SMS text alerts. Business customers will be set up the same way and FNB will send the email or SMS alerts on behalf of the customer.

2. OAG flightview Servers

The OAG flightview Servers support the FNB data API by receiving customer flight registration requests and sending notifications, both over the Internet. The registrations are received on a rolling basis via a REST (Representational State Transfer) API while the

notifications are sent using HTTP(S) protocol. Details of these APIs are provided later in this document.

The OAG flightview Servers are in a secure co-location facility with 24/7 support and 99% plus uptime. They are load-balanced to ensure the quick processing of flight registrations and delivery of notifications.

3. Registering Flights

To register flights, the customer shall use a simple REST (Representational State Transfer) API. This will be an HTTP POST request. The request will contain:

- customer authentication information,
- identifying information about a target flight (airline code, flight number, departure/arrival airport codes, scheduled departure/arrival date & time)
- an action for this flight (register/unregister).

More specifically, the request will consist of a base URL where some parameters are captured in this URL, plus some query parameters. Each of these shall be specified below.

3.1 Base URL

The base URL will be: https://fnb.flightview.com

3.2 POST Verb

Use HTTP POST when issuing the request.

3.3 URL REST Parameters

The following parameters will be added to this base URL:

1. **register/unregister:** indicates if the specified flight should be registered or unregistered
2. **departure/arrival:** indicates whether the identifying information for the flight is arrival or departure information
3. **{airport}:** 3-letter airport coded associated with the flights departure/arrival airport
4. **{carrier}:** 2-letter airline code
5. **{flight number}:** flight number
6. **{year}:** year associated with the scheduled departure/arrival time for the flight; as yyyy
7. **{month}:** month associated with the scheduled departure/arrival time for the flight; as mm
8. **{day}:** day associated with the scheduled departure/arrival time for the flight; as dd
9. **{time}:** (optional*) time associated with the scheduled departure/arrival for the flight; as hhmm. Time is local to the departure or arrival city. Hours are in 24 hour format.

*Note: If a scheduled time is specified, FNB will match to a flight based on this time. Specifying the time is important when there are multiple flights with the same flight number on the same date between the same city pair. Under such circumstances, if a time is not specified in the registration, then FNB will send notifications for both flights. In this case, your system will have to disambiguate between the two. Though this scenario doesn't happen too

often, it does occur and therefore, it is recommended that you specify the time as a parameter in the URL.

3.3 Authentication Parameters

The POST body will use standard JSON (JavaScript Object Notation) to specify the authentication parameters for receiving alerts. These parameters are the following:

1. appid - id that identifies the FNB application used by the customer

2. appkey - key that authenticates the customer and authorizes them to use this application

Sample post body:

{"RequestParameters":{"appid":"fnb.json.myapp","appkey":"secret"}}

3.4 Sample Request

Putting all the values above together, here is a sample request for registering flight AA1 departing from JFK on 11/15/2016:

https://fnb.flightview.com/register/departure/JFK/AA/1/2016/11/15

Post body:

{"RequestParameters":{"appid":"fnb.json.myapp","appkey":"secret"}}
Here is the same example specifying the departure time at 9:00am:

https://fnb.flightview.com/register/departure/JFK/AA/1/2016/11/15/0900

Post body:

{"RequestParameters":{"appid":"fnb.json.myapp","appkey":"secret"}}

Note: These are samples only with made up values

3.5 Additional Format for URL Parameters (supports registration by tail number)

In addition to the URL parameters and query format described in section 3.2, a new query parameter format has been added. This enables a customer to register general aviation flights for notifications. This new format can be used for both scheduled commercial flights as well as general aviation flights.

1. **register/unregister:** indicates if the specified flight should be registered or unregistered
2. **departure/arrival:** indicates whether the identifying information for the flight is arrival or departure information
3. **{airport}:** 3-letter airport coded associated with the flights departure/arrival airport
4. **ACID:** This is a constant to represent the aircraft ID that follows

5. **{aircraft identifier}:** flight number or tail number
6. **{year}:** year associated with the scheduled departure/arrival time for the flight; as yyyy
7. **{month}:** month associated with the scheduled departure/arrival time for the flight; as mm
8. **{day}:** day associated with the scheduled departure/arrival time for the flight; as dd
9. **{time}:** (optional*) time associated with the scheduled departure/arrival for the flight; as hhmm. Time is local to the departure or arrival city. Hours are in 24 hour format.

Here is an example to show the usage for a commercial scheduled flight, using airline and flight number.

https://fnb.flightview.com/register/departure/IAH/ACID/UA6072/2017/02/28

Post body:

{"RequestParameters":{"appid":"fnb.json.myapp","appkey":"secret"}}

Here is an example to show the usage for a GA flight, using tail number.

https://fnb.flightview.com/register/departure/TEB/ACID/N123XX/2017/02/28

Post body:

{"RequestParameters":{"appid":"fnb.json.myapp","appkey":"secret"}}

3.6 FNB Response

When the customer sends a request for a flight registration or unregistration, a response will be returned from FNB, with:

(1) an HTTP status code that indicates success/failure relative to receiving the request
(2) a JSON body that indicates success/failure for processing the request.

Potential errors include: the flight was not found, the syntax was invalid, or the customer authentication was invalid.

The following is an example of the JSON body for a successful registration:


```
{
  "RegistrationRequest": {
          "Action": "Register",
           "EndpointType": "arrival",
          "Airport": "EWR",
          "Airline": "UA",
          "FlightNumber": "49",
          "Year": "2016",
          "Month": "12",
          "Day": "19",
          "Time": "06:16:00",
          "Email": "",
          "EmailDisplayName": "",
           "LanguageCode": "en",
          "ContentType": "text",
          "Phone": "",
          "SmsCarrier": "",
          "TwitterHandle": "",
          "AppId": "fnb.json.example",
```

```
            "DeliveryUrl": "",
            "VerifyUrl": false,
            "VerifyUrlPostData": "",
            "VerifyUrlTimeoutSec": -1
  },
             "FlightIdentifiers": [
             {
                        "AirlineCode": "UA",
                        "FlightNumber": "49",
                        "DepartureAirportCode": "BOM",
                        "ArrivalAirportCode": "EWR",
                        "SchedDepartureLocal": "2016-12-19T00:10:00"
            }
            ],
            "Success": true,
             "Error": ""
}
```


The following is an example of the JSON for an unsuccessful registration because the flight is not found:


```
{"RegistrationRequest":
              {
                        "Action":"Register",
                        "EndpointType":"Departure",
                        "Airport":"ATL",
                        "Airline":"AA",
                        "FlightNumber":"1442",
                        "Year":"2016",
                        "Month":"12",
                        "Day":"11",
                        "Time":"14:22:00",
                        "AppId":"fnb.json.refimpl",
              },
              "FlightIdentifier":null,
              "Success":false,
              "Error":"Flight not Found"
}
```


Note: A flight should only be registered by a customer once. However, if a flight is registered again, FNB will send a standard response for successfully processing the request. Meanwhile, only 1 set of notifications will be sent for the flight (no matter how many times it is registered). A “now tracking” alert is sent for every registration.

Note: The software making the http request should always check the http response code. HTTP 200 (OK) indicates there were no system errors, but you could still have a "Flight not found" for example.

If you get anything other than a 200, there is an error that needs further attention.

For example: you will get an HTTP 401 as an authentication error, and HTTP 500 for an internal system error.

4. Best Practices for Registering Flights

To ensure that a customer's integration with FNB is most efficient, OAG flightview recommends some best practices.

1. It is recommended that you wait to register a flight until it is 2-5 days prior to scheduled departure. This minimizes the probability of a traveler's plans changing after his or her flight is registered, resulting in a registration followed by an unregistration followed by a new registration. Preventing this situation results in fewer API calls and less load on all systems.
2. It is recommended that you include the schedule time (for departure or arrival) for a flight when you register it so FNB can match to it and only push notifications on the flight you care about. If you do not include the scheduled time and there are multiple flights with the same flight number on the same date between the same city pair but different departure/arrival times, FNB will send notifications on all of them and your system will need to disambiguate them.
3. Although "now tracking" can be configured off, we strongly recommend to use this alert. It is a simple and excellent method to ensure the round-trip communication path is working.

5. Error Messages for Flight Registration Issues

As mentioned in Section 3, when you register a flight, FNB will send an HTTP(S) response. If the registration is unsuccessful there will be an error in the JSON body of the response. See example below:


```
{"RegistrationRequest":
        {
                "Action":"Register",
                "EndpointType":"Departure",
                "Airport":"ATL",
                "Airline":"AA",
                "FlightNumber":"1442",
                "Year":"2016",
                "Month":"12",
                "Day":"11",
                "Time":"14:22:00",
                "AppId":"fnb.json.refimpl",
        },
        "FlightIdentifier":null,
        "Success":false,
        "Error":"Flight not Found"
}
```


There are a few error messages that may be sent depending on the issue with the flight registration. These errors are listed in the table below. If you have further questions about them, please contact the OAG flightview team.

Error Message	Issue or Problem with Flight Registration
"Flight not Found"	Flight could not be found in FNB system. This could be due to a variety of reasons; some possibilities include: (1) the flight simply does not exist, (2) the flight has been recently added to the airline's schedule and is not yet in the FNB system, (3) the airline

	code, airport code or flight number specified in the URL is invalid.
“Invalid request syntax”	Required parameter in the URL is missing. Please review the required parameters listed above and ensure they are all included.
“Invalid appid/appkey combination”	Either the appid or appkey or both are invalid. Please check with the FNB team for your valid appid and appkey.
“Invalid year[month][day][time] format”	The format of the specified date parameter is invalid. Please check the format defined above for these parameters.

6. Auto-registering flights

There are two ways of registering flights – customer registers each flight as per the process described in section 3 above or registering all flights for an airline automatically. Auto-registration is the process of receiving alerts for all flights for an airline. The airlines are configured during setting up the customer instance and after that, the customer need not register any flight belonging to the airline, as per the process described in section 3 above. Making changes to the auto-registered airlines list (adding or removing airlines) require changes to the customer’s configuration and needs to be scheduled for a re-deployment.

7. Receiving Notifications for Registered Flights

(Note: This section describes the notification messages received by a business customer and not the end user)

When an event occurs that triggers a notification, you will receive an HTTP(S) request which contains the notification information in JSON format in the POST body. For security, some customers use HTTPS and restrict acceptance of the HTTPS request by FNB’s IP range. The customer will set up a Delivery URL and build a “receiver” web app hosted in your environment that will listen for this POST. The Delivery URL may be any legal URL. The “receiver” app can be in any programming environment and shall return an HTTP response.

Each notification is delivered at least once on a best-effort policy. If more than one notification is triggered for a given flight (multiple updates occurred), you shall receive a separate POST for each one.

When you receive an HTTP/HTTPS request, your “receiver” web app should:

1. Extract the flight identifier to determine the flight associated with the notification
2. Extract the Alert Type and Alert Sub-Type fields to determine the event that triggered the notification
3. Do one of the following, based on the event that triggered the notification:
   - Use the Alert Text field to send an alert directly to a customer, employee or application
   - Execute business logic or send custom alert text to a customer, employee or application
4. Send an HTTP response acknowledging receipt of the FNB HTTP/HTTPS request and an indication of whether your system successfully processed it or not.

For example, a travel agency might receive a notification that a traveller’s flight is cancelled, then rebook him/her on a different flight and send an email indicating the change. Or a cruise

operator may receive a notification that an inbound flight for a group of passengers is delayed and decide to delay their ship's departure from port until they arrive. Or a limousine service might receive a notification that a client's flight has arrived, and send an appropriate text message to a driver to pick him/her up at the appropriate baggage claim.

7.1 Specifying New Delivery URL

When your instance of FNB is set up, the Delivery URL where FNB should send your notifications will be included in the configuration. At any time you may use the API to change this Delivery URL via an HTTP or HTTPS request. This will be an HTTP GET request.This request will consist of a base URL plus some query parameters.

The base URL will be: http://fnb.flightview.com/updatedeliveryurl/

The authentication parameters need to be part of the POST request as described above:
1. **AppID={application ID}:** id that identifies the version of FNB used by the customer
2. **AppKey={application key}:** key that identifies the customer and authenticates them to use this version of FNB

Other parameters that need to be specified are:
3. **deliveryURL={delivery URL}:** customer's new external URL where FNB will send notifications
4. [optional] **VerifyURL={true/false}:** this parameter should be included and set to 'false' if the Delivery URL should not be verified by FNB (perhaps because it has not been fully set up yet); if this parameter is included and set to 'true' yet the customer's new Delivery URL does not respond when FNB tries to verify it, the Delivery URL will not be changed to the new one. Default value is false.

Putting all the values above together, here is a sample request for changing the Delivery URL:
https://fnb.flightview.com/updatedeliveryurl

Post body:

```
{
   "RequestParameters": {
      "appid": "fnb.json.myapp",
      "appkey": "secret",
      "deliveryurl": "http://testURL/FnbReceiver/default.aspx"
   }
}
```

When FNB has successfully processed a request to change your Delivery URL, it will send an "UpdatedDeliveryURL" notification confirming the change. The POST body in JSON will look like the following:

```
"Alert":
{
          "Type":"UpdatedDeliveryURL",
          "SubType":"",
          "Text":"DeliveryURL has been updated to <DeliveryURL>",
},
```

7.2 Notification Content in POST Body

The POST body will contain a number of fields in JSON format, including:

- the specific alert event that triggered the notification (e.g. "ETA change") and an alert message for this event (e.g. "Flight UA123 now arriving at 10:07 PM"),
- detailed information about the flight*
- a list of the changes associated with the flight.*

The amount of information in the POST body is extensive and enables you to create your own custom alert messages for your users that contains as much detail as you prefer.

***Note:** Section 8 of this document describes each field in the flight details and flight changes.

The following is an example of the JSON POST body for a notification associated with flight BA119:


```
{
	"FlightIdentifier":{
		"AirlineCode":"BA",
		"FlightNumber":"119",
		"DepartureAirportCode":"LHR",
		"ArrivalAirportCode":"BLR",
		"SchedDepartureLocal":"2016-11-16T14:00:00"
	},
	"Alert":{
		"Type":"StatusChange",
		"SubType":"Arrived",
		"Text":"Flight BA119: Status is now Arrived",
		"Reinstated":false
	},
	"FlightData":{
		"FvFlightId":"BA119LHRBLR11161400",
		"ProcessingTimeUtc":"2016-11-16T23:42:00Z",
		"AirlineCode":"BA",
		"FlightNumber":"119",
		"SchedDepartureAirportCode":"LHR",
		"SchedArrivalAirportCode":"BLR",
		"Status":"InGate",
		"ScheduleStatus":"",
		"DiversionStatus":"Original",
		"OperatingAirlineCode":"",
		"OperatingFlightNumber":"",
		"SchedDepartureLocal":"2016-11-16T14:00:00",
		"SchedArrivalLocal":"2016-11-17T05:00:00",
		"SchedDepartureUtc":"2016-11-16T14:00:00Z",
		"SchedArrivalUtc":"2016-11-16T23:30:00Z",
		"LatestDeparture":{
			"Accuracy":"Actual",
			"DateTimeUtc":"2016-11-16T14:00:00Z",
			"DateTimeLocal":"2016-11-16T14:00:00",
			"DateTimeType":"Gate",
			"SourceType":"AirlineAirport"
		},
		"LatestArrival":{
			"Accuracy":"Actual",
			"DateTimeUtc":"2016-11-16T23:37:00Z",
			"DateTimeLocal":"2016-11-17T05:07:00",
```

```
                        "DateTimeType":"Gate",
                        "SourceType":"AirlineAirport"
                },
                "DepartureTerminal":"5",
                "DepartureGate":"",
                "ArrivalTerminal":"",
                "ArrivalGate":"",
                "Baggage":"",
                "ServiceType":"J",
                "AircraftType":"777",
                "OptionalEquipment":"",
                "WeightClass":"",
                "AlternateDepartureAirportCode":"",
                "AlternateArrivalAirportCode":"",
                "DepAirportCountryId":"GB",
                "ArrAirportCountryId":"IN",
                "LegSequenceNumber":1,
                "RecoveryExists":false,
                "TailNumber":"",
                "Unscheduled":false,
                "RecoveryFlight":null
        },
        "FlightChangeData":{
                "HasAnyChanges":true,
                "StatusChanged":true,
                "PreviousStatus":"Landed",
                "DepartureTimeChanged":false,
                "PreviousDepartureTimeLocal":"",
                "PreviousDepartureTimeUtc":"",
                "ArrivalTimeChanged":true,
                "PreviousArrivalTimeLocal":"2016-11-17T05:09:00",
                "PreviousArrivalTimeUtc":"2016-11-16T23:39:00Z",
                "DepartureTerminalChanged":false,
                "PreviousDepartureTerminal":"",
                "DepartureGateChanged":false,
                "PreviousDepartureGate":"",
                "ArrivalTerminalChanged":false,
                "PreviousArrivalTerminal":"",
                "ArrivalGateChanged":false,
                "PreviousArrivalGate":"",
                "BaggageChanged":false,
                "PreviousBaggage":"",
                "IsNewToFlightWindow":false,
                "OtherChanged":false,
                "DiversionStatusChanged":false,
                "PreviousDiversionStatus":"Original"
        },
        "AlertIdentifier":"5cc8dfef-8799-4232-8791-b5002b0d19ce"
}
```


Note: UTC fields should be interpreted as UTC even if there is no trailing Z.

The information in the FlightIdentifier will remain constant. So, if the scheduled departure changes, you may still refer to the flight by the scheduled departure returned in the first alert you get for the flight, typically the “Now Tracking” or “New To Window” alert.

7.3 Customer Response

When FNB sends a request with the flight notification, you should return a response, with:
(1) an HTTP status code
(2) a JSON body that indicates success/failure for processing the request.

The HTTP status code sent back to FNB from the customer should be:

- 200 if the notification was accurately received or a business logic error occurred

non-200 if a system error was experienced with the notification
In the JSON body, there are two suggested, though *optional* fields:

AlertIdentifier: The GUID in the HTTP(S) request from FNB.

ResponseText: Free form text of your choosing, which will be logged in the FNB system. This free form text will often include information about the success or failure of processing the request.

For example:

```
{
        “AlertIdentifier”:  "fa954c6c-1f64-4084-8cb0-05629eb0c062"
        "ResponseText": "Processed without errors”
}
```

Again, neither of these fields is necessary to include in the JSON body.

8. Events that Trigger Notifications

FNB sends a notification to the customer for the following list of flight updates. If multiple updates occur to a flight at the same time, then the customer will receive a separate notification for each one. For each update the following information is provided in the table:

- Description
- Alert Type (as seen in JSON)
- Alert Sub-Type (as seen in JSON)
- Alert Text (as seen in JSON)

If there are additional updates/events that you would like to receive notifications for, please contact the OAG flightview team.

Description	Alert Type	Alert SubType	Alert Text*
Flight status changed to ‘Scheduled’	StatusChange	Scheduled	Flight <ACID>: Status is now Scheduled.
Flight status changed to ‘Departed’	StatusChange	Departed	Flight <ACID>: Status is now Departed. {Left Gate at <ActDep>}
Flight status changed to ‘In Air’	StatusChange	InAir	Flight <ACID>: Status is now In Air. {Takeoff at <ActOff>}
Flight status changed to ‘Landed’	StatusChange	Landed	Flight <ACID>: Status is now Landed. {Landed at <ActOn>}
Flight status changed to ‘Arrived’	StatusChange	Arrived	Flight <ACID>: Status is now Arrived. {At Gate at <ActArr>}

Flight status changed to 'Expected'	StatusChange	Expected	Flight <ACID>: Status is now Expected.
Flight status changed to 'Cancelled'	StatusChange	Cancelled	Flight <ACID>: Status is now Cancelled.
Flight status changed to 'Departed' & 'Diverted'	StatusChange	Departed-Diverted	Flight <ACID>: Diverted to <ARR>.
Flight status changed to 'Departed & 'Recovered'	StatusChange	Departed-Recovered	Flight <ACID>: Recovered flight is Departed.
Flight status changed to 'In Air' & 'Diverted'	StatusChange	InAir-Diverted	Flight <ACID>: Diverted to <ARR>.
Flight status changed to 'In Air' & 'Recovered'	StatusChange	InAir-Recovered	Flight <ACID>: Recovered flight is In Air.
Flight status changed to 'Arrived' & 'Diverted'	StatusChange	Arrived-Diverted	Flight <ACID>: Diverted flight Arrived at <ARR>.
Flight status changed to 'Arrived' & 'Recovered'	StatusChange	Arrived-Recovered	Flight <ACID>: Recovered flight arrived at <ARR>.
Flight status changed to 'Landed' & 'Diverted'	StatusChange	Landed-Diverted	Flight <ACID>: Diverted flight Landed at <ARR>.
Flight status changed to 'Landed' & 'Recovered'	StatusChange	Landed-Recovered	Flight <ACID>: Recovered flight Landed at <ARR>.
Flight status changed to 'Delayed' and new ETD (Est Time Dep)	StatusChange	Delayed-New-ETD	Flight <ACID>: Delayed - Departing at <EstDep>
Flight status changed to 'Delayed'	StatusChange	Delayed	Flight <ACID>: Status is now Delayed.
ETD (Est Time Dep) changed	ETDChange		Flight <ACID>: Now Departing at <EstDep>.
ETA (Est Time Arr) changed	ETAChange		Flight <ACID>: Now Arriving at <EstArr>.
Change in Departure Gate assignment	GateChange	Departure	Flight <ACID>: Departure Term-Gate is now <Term-Gate>.
Change in Arrival Gate assignment	GateChange	Arrival	Flight <ACID>: Arrival Term-Gate is now <Term-Gate>.
Change in Baggage assignment	BaggageChange		Flight <ACID>: Baggage Claim is now <Bags>.
Just registered and in Flight Window** (started monitoring/tracking)	NowTracking		Flight <ACID>: <DEP> to <ARR>, is now being tracked.
Entered Flight Window** (started monitoring/tracking)	NewToWindow	Tracking	Flight <ACID>: <DEP> to <ARR>, is now being tracked.
Entered Flight Window** (started monitoring/tracking) & Flight status is 'Cancelled'	NewToWindow	Cancelled	Flight <ACID>: <DEP> to <ARR>, is now Cancelled.
Pre-ETD confirmation X hrs prior to estimated departure (X is configurable)	PreETD	<# of minutes>	Flight <ACID>: Estimated to depart in approx <hrs> hr[s] [<mins> min[s]] at <ETD>.

Pre-Departure confirmation X hrs prior to sched departure (X is configurable)	PreDeparture	<# of minutes>	Flight <ACID>: Scheduled to depart in approx <hrs> hr[s] [<mins> min[s]] at <SchedDep>.
Flight registered by customer	FutureFlightRegistered		Flight <ACID>: <DEP> to <ARR> on <DepDate>, registered for notifications.
Scheduled departure time changed	RevisedSchedule		Flight <ACID>: Now Scheduled to depart at <EstDep>.
Flight unregistered by customer	Unregistered		Flight <ACID>: <DEP> to <ARR> on <DepDate>, Unregistered for notifications by client.
Updated Delivery URL	UpdatedDeliveryURL		The Delivery URL has been updated to <DeliveryURL>.

*** Note: The Alert Text is subject to change.**
** See Section 9 below for definition of 'Flight Window'.

8.1 Special Cases

There are some important events that should be called out to ensure they are treated correctly by your "receiver" web app.

1. **Registering a Cancelled Flight:** There may be times where you register a flight that has already been cancelled. If this happens, it is important that you are informed immediately so you can communicate the cancellation to your customer or your internal team. Therefore, after receiving the registration request, FNB will send you a notification for a status change event, Flight status changed to 'Cancelled'.

2. **Flight Reinstated after being Cancelled:** There may be times where a flight is cancelled by an airline and then reinstated. When a flight is reinstated, many airlines provide a status update for it with the reinstatement. Therefore, you will receive a notification for the flight for one of the above events, which will include the Reinstatement attribute in the alert object set to 'true'. ***Note:*** The Reinstatement attribute is always part of the alert object in the JSON, but most of the time it is set to 'false'.

8.2 Smart Notification Rules

Currently, in FNB there are smart notification rules relative to a small subset of flight updates that indicate when a notification should not be sent to the customer. This is to ensure that only actionable information is sent. For example, for many customers a 2 minute change to an estimated time of arrival is not particularly actionable because if they are sending alerts to travelers, they are unlikely to care about a 2 minute change and may be annoyed by receiving such an alert. The current set of Smart Notification Rules is listed below.

Estimated time of Departure (ETD)

- After a notification is sent for a 'delay' status with an initial ETD, only send another notification if the updated ETD has changed by more than 15 minutes as compared to the last ETD notification sent. For example, if the ETD changes by 2 minutes compared to the last ETD notification sent, do not send another notification. However, if the ETD

continues to change so the cumulative change is greater 15 minutes as compared to the last ETD notification sent, then send a new notification concerning the current ETD.

Estimated Time of Arrival (ETA)

- Only send a notification if the ETA has changed by more than 15 minutes as compared to:
  1. the scheduled arrival time (if no other ETA has been reported yet)
  2. the last ETA reported to the customer (in an earlier notification)

Terminal & Gate Assignments

- Send a notification for all terminal and gate assignment changes except the following cases:
  1. In some airports there are gate numbers that end in a letter; for example, 12A, 12B, 12C and 12D or 77 and 77A. In many cases these gate doors are next to one another or are really 1 gate door associated with multiple jetways. Such gate assignments are common for small commuter planes. Also, many times airline operators switch the gate/jetway one or more times just prior to the flight's departure so a flight's gate assignment may change from 12A to 12C to 12B in a short period of time. Since these gates are so close to one another and in many cases are simply different jetways associated with the same gate door, it is less than desirable to annoy travelers by sending them multiple alerts for these changes. Therefore, do not send a notification for such changes.
  2. In some large, busy airports the gate management system routinely reassigns gates for flights later in the day to accommodate flights currently delayed or facing some other unexpected issue. Because gate assignments often change one or more times a few hours to prior to scheduled departure, do not send a notification for a gate change that occurs more than Y minutes before the scheduled departure time, where Y is configurable (see Section 10 for customer configuration).
  3. Do not send a notification when the terminal and/or gate assignments change from a populated value to a blank value. In some cases these values strobe which can cause confusion if notifications are sent.
  4. Do not send a notification for a departure terminal or departure gate change after the flight has departed.
  5. Do not send a notification for any terminal or gate change after the flight has reached a terminal (concluding) status; these statuses include Cancelled, Arrived, Landed.

  Note: It is possible to turn off all Terminal & Gate Assignment smart rules so all changes are sent without exception (see Section 10 on customer configuration).

9. Flight Data and Flight Change Data in Notification

In this section the different fields associated with the FlightData and FlightChangeData objects in the notification body are described. The first table lists those associated with FlightData and the second table lists those associated with FlightChangeData. For FlightChangeData object, there is a pair of fields for each component of the flight that may have changed:

1. The first "Change" field is a flag that indicates when a change to this component has occurred. Possible values are: *true* or *false* where *true* means there is a change and *false* means there is no change.
2. The second "Previous" field is the previous value for this component. It will be blank if the value has not changed.

Flight Data Fields

Field Name (Column)	Description

FVFlightId	Unique identifier for the flight leg. FvFlightId field is not to be used as a unique identifier; it is for FV internal use only.
ProcessingTimeUtc	Time when the Batch output was created (UTC format)
AirlineCode	The 2 character IATA airline code
FlightNumber	The flight number assigned by the airline
SchedDepartureAirportCode	The 3 letter IATA airport code for the departure airport
SchedArrivalAirportCode	The 3 letter IATA airport code for the arrival airport
Status	Current status or “state” of the flight leg; possible values include: Unknown, Scheduled, NoTakeoffInfo, Proposed, Cancelled, Delayed, OutGate, InAir, Landed, InGate
ScheduleStatus	Current status or “state” of the flight leg relative to its schedule departure/arrival times; possible values include: Undefined, DepartureEarly, DepartureOnTime, DepartureDelay, ArrivalEarly, ArrivalOnTime, ArrivalDelay
DiversionStatus	Current status or “state” relative to scheduled arrival airport for the flight leg; possible values include: Original or <blank> (flight leg is on target for scheduled arrival airport), Diverted (flight leg diverted to alternate arrival airport instead of scheduled arrival airport), Recovery (flight leg for recovery from alternate airport to scheduled arrival airport)
OperatingAirlineCode	The 2 character IATA airline code for the airline that is operating the flight
OperatingFlightNumber	The flight number associated with the operating flight assigned by the operating airline
SchedDepartureLocal	The scheduled departure date & time assigned by the airline (in time zone local to the departure airport)
SchedArrivalLocal	The scheduled arrival date & time assigned by the airline (in time zone local to the arrival airport)
SchedDepartureUtc	The scheduled departure date & time assigned by the airline (UTC format)
SchedArrivalUtc	The scheduled arrival date & time assigned by the airline (UTC format)
LatestDeparture - Accuracy	Flag indicating whether the the *DateTimeLocal* & *DateTimeUtc* values are based on the schedule, estimated by the airline or OAG flightview’s algorithm, or actual - having occurred already; possible values are: Scheduled, Estimated, Actual
LatestDeparture - DateTimeUtc	The most recent departure date & time for the flight leg (UTC format)
LatestDeparture - DateTimeLocal	The most recent departure date & time for the flight leg (in time zone of the departure airport)
LatestDeparture - DateTimeType	Flag indicating whether the DateTimeLocal & DateTimeUtc values are associated with departing the Gate or with taking off from the Runway; possible values are: Gate, Runway

LatestDeparture - SourceType	The data source of the latest departure date & time in fields DateTimeLocal & DateTimeUtc; possible values are: ScheduleData, AirlineData, OAG flightview Data
LatestArrival - Accuracy	Flag indicating whether the the DateTimeLocal & DateTimeUtc values are based on the schedule, estimated by the airline or OAG flightview's algorithm, or actual - having occurred already; possible values are: Scheduled, Estimated, Actual
LatestArrival - DateTimeUtc	The most recent arrival date & time for the flight leg (UTC format)
LatestArrival - DateTimeLocal	The most recent arrival date & time for the flight leg (in time zone of the arrival airport)
LatestArrival - DateTimeType	Flag indicating whether the DateTimeLocal & DateTimeUtc values are associated with touchdown on the runway or arrival at the Gate; possible values are: Gate, Runway
LatestArrival - SourceType	The data source of the latest arrival date & time in fields DateTimeLocal & DateTimeUtc ; possible values are: ScheduleData, AirlineData, Flightview Data
DepartureTerminal	Terminal assignment for the flight leg at the departure airport
DepartureGate	Gate assignment for the flight leg at the departure airport
ArrivalTerminal	Terminal assignment for the flight leg at the arrival airport
ArrivalGate	Gate assignment for the flight leg at the arrival airport
Baggage	Baggage claim assignment for the flight leg at the arrival airport
ServiceType	The type of service provided by the flight leg, whether it is for passengers only, cargo only, mail only, passengers & cargo, cargo & mail; common values include: J (scheduled, normal service passenger), S (scheduled, shuttle mode passenger) – check with OAG flightview for more values
AircraftType	The type of aircraft that is flying the flight leg
OptionalEquipment	A classification of the AircraftType; possible values are: H (heavy), M (medium), L (light)
WeightClass	The weight class of the aircraft that is flying the flight leg
AlternateDepartureAirportCode	If DiversionStatus = Recovery, this is the alternate airport code the recovery leg departed from
AlternateArrivalAirportCode	If DiversionStatus = Diverted, this is the alternate airport code the flight has been diverted to
DepAirportCountryId	Country Id for the departure airport
ArrAirportCountryId	Country Id for the arrival airport
LegSequenceNumber	Many flight numbers have multiple legs and the sequence number indicates the order in which this Flight leg appear amongst all the legs; values are integral starting at 1
RecoveryExists	For a diverted flight leg, indicates whether a recovery flight leg has been established; possible values are: true, false

TailNumber	The unique tail number of the aircraft that is flying the flight leg
Unscheduled	Indicates that the flight was not in the schedule; possible values are: true, false
RecoveryFlight	An entire FlightData object (with all the fields listed in this table) for the recovery flight when the recover flight exists. If RecoveryExists field above has a value of 'true' then this object exists. When RecoveryExists field is 'false' then this object is 'null'.

Flight Data Change Fields

Field Name (Column)	Description
HasAnyChanges	Flag that indicates whether any change has occurred to the flight; possible values: true, false
StatusChanged	Flag that indicates there is a change in the Status value from that previously reported; possible values: true, false
PreviousStatus	Previous value of Status
DepartureTimeChanged	Flag that indicates there is a change in the Departure date & time value from that previously reported; possible values: true, false
PreviousDepartureTimeLocal	Previous value of Departure date & time (Local airport time) field
PreviousDepartureTimeUtc	Previous value of Departure date & time (UTC format) field
ArrivalTimeChanged	Flag that indicates there is a change in the Arrival date & time from that previously reported; possible values: true, false
PreviousArrivalTimeLocal	Previous value of Arrival date & time (Local airport time) field
PreviousArrivalTimeUtc	Previous value of Arrival date & time (UTC format) field
DepartureTerminalChanged	Flag that indicates there is a change in the Departure Terminal value from that previously reported; possible values: true, false
PreviousDepartureTerminal	Previous value of Departure Terminal field
DepartureGateChanged	Flag that indicates there is a change in the Departure Gate value from that previously reported; possible values: true, false
PreviousDepartureGate	Previous value of Departure Gate field
ArrivalTerminalChanged	Flag that indicates there is a change in the Arrival Terminal value from that previously reported; possible values: true, false
PreviousArrivalTerminal	Previous value of Arrival Terminal field
ArrivalGateChanged	Flag that indicates there is a change in the Arrival Gate value i from that previously reported; possible values: true, false
PreviousArrivalGate	Previous value of Arrival Gate field
BaggageChanged	Flag that indicates there is a change in the Baggage claim value from that previously reported; possible values: true, false
PreviousBaggage	Previous value of Baggage Claim

IsNewToFlightWindow	Flag that indicates the flight leg is new to the Flight Window; possible values: true, false. It should have no “Change” fields populated.
OtherChanged	Flag that indicates there is some other field, not listed in this table, has changed; possible values: true, false
DiversionStatusChanged	Flag that indicates there is a change in the Diversion Status value from that previously reported; possible values: true, false
PreviousDiversionStatus	Previous value of Diversion Status field

Note: UTC fields should be interpreted as UTC even if there is no trailing Z.

10. Flight Window for Monitoring Registered Flights

While customers often register flights before their scheduled departure dates, the flights are not monitored until their scheduled departure time is 24 hours from now (current time). When a flight’s scheduled departure time is 24 hours from now, FNB begins to monitor it for changes and send notifications on these changes. The flight continues to be monitored until an appropriate amount of time after it reaches a termination status (Cancelled, Landed, Arrived, or Past Flight). During this period, whenever a change occurs to the flight, a notification is sent to all the customers who registered it.

11. Registering for Email Notifications to End Users

Starting with release 1.8, FNB can send email alerts on behalf of business customers. Your application must first be configured by OAG for user notifications.

When issuing the request, the user's address and other content-related parameters must also be specified in the POST body JSON. In addition to the authentication parameters described above, the user parameters are:

1. **{email}:** end user’s email address to send the alerts to
2. **{emaildisplayname}:** The email receiver’s display name (in English only)
3. **{contenttype}:** email format. Options are text or html
4. **{language}:** 2 character ISO language code. Default is en. Other international languages that are supported include but not limited to German (de), French (fr) and Arabic (ar) etc.
5. acceptsterms choose "T" to confirm that you have informed the user that they accept legal terms about privacy, etc.

Putting all the values above together, here is a sample request for registering flight DL2747 departing from LGA on May 9th 2016 at 9:00am and to send email alerts to John Doe in HTML format and in English.

https://fnb.flightview.com/register/departure/LGA/DL/2747/2016/05/09/0900

Post body:

{"RequestParameters":{"appid":"fnb.json.myapp","appkey":"secret","email":"user@example.com","emaildisplayname":"Test User","languagecode":"en","contenttype":"text","acceptsterms":"T"}}

For integrating web components on the customer website to enable users to register for alerts, please contact an OAG flightview sales representative for more details.

11.1 Registering for SMS Notification for end users

Starting with release 1.8, OAG flightview can send SMS alerts on behalf of business customers.

When issuing the request, the user's address and other content-related parameters must also be specified in the POST body JSON. In addition to the authentication parameters described above, the user parameters are:

1. **{phone}:** end user's phone number to send the alerts to, starting with the country code
2. **{language}:** 2 character ISO language code. Default is en. Other international languages supported include German (de), French (fr) and Arabic (ar) etc.
3. **{SMSCarrier}**: SMS Vendor for delivering text messages.

12. Customer Configuration

There will be a configuration file for each FNB customer maintained by the OAG flightview team. This configuration file contains customer-specific information for using the API and configurations that drive how notifications are sent.

The customer-specific information is the following:

- **AppID** (for authentication)
- **AppKey** (for authentication)
- **Customer's Delivery URL** (where HTTP(S) requests with notifications are sent)

The customer-specific configurations that drive how notifications are sent are the following:

- **Suppressing Notifications for Events**
  In Section 7 there is a list of events for which notifications will be sent. Some customers do not want notifications on certain events because they do not find these events useful. Therefore, a customer may choose to suppress notifications for any of the events listed in Section 8.

- **Threshold for Sending Gate Changes**
  As described in Section 8.2, gate assignments at many airports often change in the hours prior to departure due to earlier flights arriving and departing off schedule. Therefore, notifications for gate changes that occur multiple hours prior to scheduled departure can generate a lot of noise. Given this phenomenon, Terminal/Gate Smart Notification rules prevent FNB from sending notifications on gate changes that occur more than Y minutes before the flight's scheduled departure. Y minutes is the

threshold that may be specified by the customer. Often this value is set to 180 or 240 minutes (3 or 4 hours).

- **Use of Terminal/Gate Notification Smart Rules**
  Some customers want to receive notifications on every Terminal/Gate change, no matter what it is or when it occurs. Therefore, it is possible for a customer to turn off the Terminal/Gate Notification Smart Rules described in Section 8.2, in which case ALL terminal/gate changes will be sent from the time the flight enters the Flight Window.

- **Pre-Departure Confirmation Notification Time Setting**
  The Pre-Departure Confirmation notification is a notification sent X minutes prior to the scheduled departure of a flight and contains all the current information about the flight. It is used as a reminder that the registered flight is scheduled to departure soon. The number of minutes, X, prior to scheduled departure when this Pre-Departure notification is sent, can be set by the customer.

12.1 Customer Configuration for Email Alerts

The following configurations can be done on a per-application basis by the OAG flightview team for each customer.

- **From email address** – This is a no-reply email address that can be set up to match the customer's email domain.

- **From email display name** – This is the display name that the email recipient will see. It can be configured to any English text, for example "<Customer name> Flight Status>".

- **Translated text for alerts** – The translated text for each of the alerts need to be set up for the languages that the customer chooses to use.

12.2 Customer Configuration for SMS Alerts

The following configurations can be done on a per-application basis by the OAG flightview team for each customer.

- **Alphanumeric sender ID** – This is the sender id for the SMS. Not all countries allow alphanumeric sender id though, in which case a numeric number is used.

- **Translated text for alerts** – The translated text for each of the alerts need to be set up for the languages that the customer chooses to use.