ODIGEO

THE ONLINE TRAVEL GROUP

B2B API integrator manual

Table of contents

1. Introduction

The purpose of this document is to describe and serve as reference manual for the Web Services B2B API for Odigeo platform. This API defines a series of operations and data objects to interact with the Odigeo booking service, starting with search, flight selection, passenger data submission, payment details submission, and ending the process with confirmation.

1.1. Prerequisites

The intended audience for this document is developers and technicians familiar with Web services technologies from the client-side perspective. Reader should be familiar with XML, WSDL, XSD and related technologies and languages. Depending on the client technology used, knowledge of Java (JAXB/JAXWS) or .NET or any Web services frameworks is expected.

2. Creating a basic WS client

The booking procedure is composed of several steps, and partners need to follow them in the correct order to finish a booking operation. Since the procedure implies several calls to different methods in the WS API, client applications must keep the HTTP session alive between different calls in order for the API to work properly. This goal is accomplished by sending back all cookies present in the HTTP responses to the server. Depending on the client technology this can be done in different ways. For example, in Java, using JAXWS, it would be something similar to setting a request context attribute before the Web Services interaction:

```
WService proxy = new WSService().getWSPort();
((BindingProvider)proxy).getRequestContext().put(BindingProvider.SESSION_MAINTAIN_PROPERTY, true);
```

All cookies issued by the server must be treated opaquely by the partner and sent back to the server on each request. A separate session has to be maintained per each user that wants to complete a booking with the API.

2.1. Cookies Management

The following actions must be performed to ensure the proper functioning of the different flow steps.

- Session cookies sent by the B2B API have to be maintained within the same session.

2.2. Webservice definition

The web service definition can be accessed by retrieving the WSDLs from the following URLs:

```
https://<host-environment>/b2b/ws/v1/B2bBooking?wsdl
```

The *v1* indicates that the version being accessed is version v1. There is a versioning mechanism that may keep multiple versions of the API running simultaneously. The version upgrade procedure is described in detail later on. Current version of the API is v1. Please note that for test purposes, the host is b2btest.odigeo.com. This environment needs to be used for testing API version upgrades, although they both operate with real data (i.e. confirmed bookings are considered final bookings as if they were performed in the regular website).
These WSDLs contain all the information needed to implement a client application: data objects definition, and exposed methods.
The service is explained in following WSDL file:

- B2bBooking. Provides needed operations to search for itineraries, create a shopping cart and complete the booking (to attach user data, payment data, and confirm).

The easiest way to implement a client application is to start by generating the client code, using a generation tool. In Java, this can be done with wsimport tool that is shipped with any recent JDK:

```
wsimport –s . https://<host-environment>/b2b/ws/v1/B2bBooking?wsdl
```

This command will generate a java package containing both generated sources and compiled classes that will serve as API client, including:

- Data objects involved in all operations, as described in the next sections.
- ObjectFactory used to create all data objects.
- Service interface that can be used to invoke the WS methods and receive the responses.

2.3. Version upgrade

This section describes the upgrade procedures that need to be carried out whenever the API is changed in such a way that is incompatible with previous versions. Incompatible changes include:

- Modification of operations signatures.
- Renaming of operations.
- Modification of data objects handled by operations.
- Deletion of operations.

Whenever one of these changes takes place, a new version of the API will be released, in a different URL that will include the version number, following the pattern:

```
https://<host-environment>/b2b/ws/<version_number>/B2bBooking?wsdl
```

The previous API version will remain alive for a period of time, and will eventually be removed later on. This period must be used by partner applications to upgrade partner implementations to new API version (that includes new functionalities, or improves existing ones). Upgrading procedure involves:

- Generation of client classes using generation tools (like wsimport in Java).
- Replacing of old client classes with generated ones.
- Adaptation of the rest of the application to the differences in the client APIs, which will usually cause it not to compile. Since the classes will be in the same package and have the same name (unless explicitly changed), only those classes that are related to the API's that have changed will need to be modified.

3. Distribution notifications

Distribution notifications' mechanism is used by B2B API to notify partners of the events occurred during booking flow.
B2B API partner can use distribution notifications mechanism to pick up offline booking changes and use getBookingById or getBookingByPartnerReference operations to get complete booking information.
The partners that want to receive a notification must provide an URL where notifications will be send. Each notification has an action and it's status.

- This URL must be provided by the API client beforehand. A GET Http Request will be done to the provided URL. A notification url will have the following parameters:

3.1. Notification url

- The notification url send by Odigeo has the following parameters:
  - bookingId. With the Odigeo booking id.
  - partnerReferenceId. With the partner reference id provided at confirmation time. It's optional because it's not manadatory to identify bookings using a partnerReferenceId.
  - actionStatus, a string containing the pair action/status separated by '_', every pair is separated by "-"

  Example:
  <partner_notification_url>?bookingId=12345678&actionStatus=ISSUED_0&
  partnerReferenceId=135BJD7789
- It's up to the partner processing this notification with some listener (servlet, php script …) and confirm that this notification has been received properly with an "ACK".

3.1.1. Actions notified

OFFLINE_CHANGES ("OFFCH"),
CREATION ("CREAT")
BOOKING_ISSUE ("BISS")

3.1.1.1. Creation notification

The notification action CREATION has the following status available:
- 0, CONTRACT
- 1, REQUEST

Contract: indicates that the booking has been closed with the provider.
Request: pending to be closed by batch processing or manual process.

3.1.1.2. GDS issuing notification

When a GDS reach a final status a BOOKING_ISSUE notification will be sent.
Possible statuses are:
- 0, ISSUED
- 1, DECLINED BY GDS
- 2, SENT TO MANUAL
- 3, CANCELLED

Issued: The booking is closed successfully.
DeclinedByGds: The GDS declined the booking.
SentTomanual: we got some error and the booking is sent to call center in order to manage it.

3.1.1.3. Offline changes notification

Eventually we can notify that a booking has changed but not affecting to the booking status, all of these notifications of changes offline will be included in the OFFLINE_CHANGES notification with a unique status = 0.

4. Environments

There are 3 environments where you have different endpoints. Each one has different purposes, and we explain them.

4.1. Mocking services environment

The aim of mocking services is to expose predefined reponses to partners. Giving certain requests the enpoint responds always the same way.

Environment: http://testingmocking1.qa.odigeo.com/b2bmock

WSDL: http://testingmocking1.qa.odigeo.com/b2bmock?wsdl

4.2. Testing environment

The aim of this environment is to expose B2B API in a testing environment for integration purposes. All bookings are fake in this environment.

Environment: https://b2bintegration-qa1.services.odigeo.com/b2b/ws/v1/B2bBooking

WSDL: https://b2bintegration-qa1.services.odigeo.com/b2b/ws/v1/B2bBooking?WSDL

4.3. Production environment

Environment: https://b2b.odigeo.com/b2b/ws/v1/B2bBooking

WSDL: https://b2b.odigeo.com/b2b/ws/v1/B2bBooking?WSDL

5. API reference

5.1. Booking service

The booking process is carried out by accessing the B2B API which is exposed by several methods (also called operations in the WS world). The parameters and return types for these methods can either be primitive types (such as Strings, Integers, Dates …) or beans, that is, objects with primitive or other bean properties (or collections of them). This chapter describes the available operations. Note that most of the complexity relies on the data objects involved on the operations, which represent the complex data structures required by the booking engine (passenger data, itineraries, search criteria …). All these methods are available through the B2bBooking wsdl.
Basic booking operation consists of searching (maybe multiple times), selecting an itinerary through the createShoppingCart method, attach passenger and buyer data through the addPersonalInfoToShoppingCart method, and confirm booking by invoking the confirmBooking method. The following diagram summarizes this workflow:

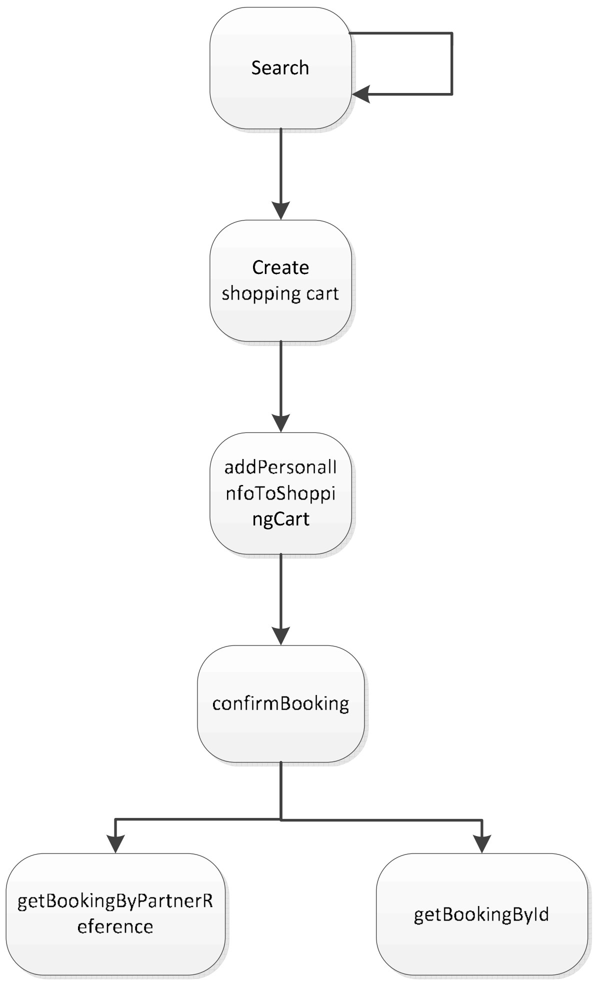

Figure 1 - B2B API operations diagram

There are also some additional optional communication ways related to the booking that happen outside the basic workflow. These are:

- Retrieving booking data by identifier, which can be performed any time after the booking is confirmed (see getBookingById and getBookingByPartnerReference for details).

- Through distribution notifications mechanism, the partner will receive notifications when a booking changes its status. To receive those notifications partner has to provide an URL that will be called whenever a booking is completed or changes its status. More information at the end of the document.

5.1.1. Search process

The first step in the booking process is searching for itineraries. The client may perform more than one search in the same session, but only the last one will be considered valid. A *searchId* parameter is included in the server response to uniquely identify active search, and will be used to match subsequent calls to read or selection methods. Clients must keep track of these unique identifiers.
All this is accomplished by using the following operation:

Return type	Name	Parameters	Description
SearchResponse	**search**	Credentials credentials, UserInfo userInfo, SearchRequest searchRequest	Performs an itinerary search request, and returns a searchResponse object that may contain results. If there is any error condition it returns a typed error message.

- SearchResponse **search** (Credentials credentials, UserInfo userInfo, SearchRequest searchRequest). Performs a search of flight itineraries for either a one-way, round-trip or multisegment request.
  - Receives a Credentials object, a UserInfo object and a SearchRequest object (all explained in next section).
  - Returns a SearchResponse object which is also described in next section.

5.1.1.1. Data objects in search process

Search methods use a series of complex objects to represent the status of the search, the search requests performed by the clients:

- **Credentials**. It contains the following fields
  - String partnerId: The partner identifier, e.g. “partner”.
  - String password: The partner password, e.g. “partner2014”.
- **UserInfo.** It contains the following properties:
  - String userAgent: The software that is acting on behalf of a user.
  - String realUserIp. The real user IP address.
- **SearchRequest.** Container for all the simple search request info:
  - ItinerarySearchRequest itinerarySearchRequest. This object contains the search request that determine the itinerary to be found features (see below).
- **ItinerarySearchRequest**. Represents a simple search criterion (including from one to six different segments) for a given set of departure and destination places and dates. It contains the following properties:
  - int maxSize. If search succeeds, a subset of itineraries will be returned. This parameter limits the maximum number of itineraries to be returned in this subset.
  - CabinClass cabinClass. Either BUSINESS, FIRST, PREMIUM_ECONOMY, ECONOMIC_DISCOUNTED or TOURIST, representing the type of cabin seat the user is asking for. If this parameter is not sent, all cabin classes are asked for. TOURIST includes PREMIUM_ECONOMY and ECONOMIC_DISCOUNTED.
  - int numAdults. Number of adult passengers.
  - int numChildren. Number of children between 2 and 12 years old. Sum of adults + children <= 9
  - int numInfants. Number of babies (under 2 years old). Minimum 0 maximum number of adults.
  - List<ItinerarySegmentRequest> segmentRequests. List of segments to be searched. The possible lengths of this list are:
    - One. For a one-way itinerary.

        - Two. If second segment destination and arrival places match destination and arrival for the first one respectively, it is what is commonly known as a round-trip itinerary. If locations are different, then it can be a completely independent segment, and in this case, it is named multisegment itinerary.
        - Three to six. For multisegment itineraries, with up to six different segments.

        Segments must be in correct date order.
    - Boolean directFlightsOnly. This flag indicates if the search will only include direct flights (flights with no transfers or technical stops).
- **ItinerarySegmentRequest**. Represents a single segment request data, containing:
    - ItineraryLocationRequest departure. Place the segment departs from.
    - ItineraryLocationRequest destination. Destination the segment arrives to.
    - Date date. Date on which the segment departure must take place. In the internal XML representation, it follows the ISO 8601 format: YYYY-MM-DD. For example, 2011-07-20, which is naturally mapped as a xs:date schema type. However, although xd:date allows for Timezone info to be specified, this information must not be set, in case it is specified, it would be ignored. Searches return always departure city timezone, so if departure iata airport it's MAD (Madrid) Timezone will be UTC/GMT +1 hour.
    - String time: Parameter can be used for indicating the exact time of departure for the segment. In the internal XML representation, it follows the format hhmm. For example, 1800.
    - Int timeWindow: Indicates the time window used for specific hours search. To use this parameter the time parameter has to be set. For example, if the time is set to 1800 and time window is set to 2, the results will include flights departing from 1600 to 2000. Results from the day after or before can be included if the time range specified is large enough.
- **ItineraryLocationRequest**. Represents a place from which a segment departs or arrives. It contains:
    - String iataCode. IATA code.
- **BaseRequest.** Information request included at any request. All requests inherit from this one. **BaseResponse**. It represents the information response that is included in all responses. All responses inherit from this one.
    - List<MessageResponse> messages. List of error and warning messages rose during the method execution.
- **MessageResponse**: Represents an API message. It's expected to be used for warnings and errors. It contains:
    - String code: Message code
    - String description: Message description
- **Preferences**. It contains:
    - String locale: Current locale.
    - String currency. Current currency.
- **PreferencesAwareResponse.** Represents the preferences status of the booking flow. Inherits from the object BaseResponse and adds the following properties:
    - Preferences preferences. Preferences for the current response.
- **SearchResult.** It represents the response for a search operation. Inherits from the object PreferencesAwareResponse and adds the following properties:
    - long searchId. Search identifier that represents this result set, and must be used in subsequent operations.
    - ItinerarySearchResults itinerarySearchResults. Explained below.
- **ItinerarySearchResults.** Represents a priced sorted list of itineraries, as a result of a search operation. It contains the following properties:
    - List<FareItinerary> itineraryResults. List of results. Each item is a FareItinerary object that represents a combination of flight routes. See details below.
    - ItinerariesLegend legend. Summary of the locations, carriers, segments and sections involved in this set of itineraries.
    - CabinClass cabinClass. Either BUSINESS, FIRST, PREMIUM_ECONOMY, ECONOMIC_DISCOUNTED or TOURIST, representing the type of cabin seat the user has asked for. ECONOMIC_DISCOUNTED AND PREMIUM_ECONOMY are considered as subtypes of the class TOURIST.
- **FareItinerary.** It represents an itinerary that fulfills the search requirements indicated by the user. It contains the following properties:

- String key. Unique itinerary identifier inside current search. It must be used if this itinerary is selected later on.
- BigDecimal totalPrice Total price for this itinerary. It’s the sum of all Fares
- PassengerTypeFare adultFare. Fare related to ADULT passengers.
- PassengerTypeFare childFare. Fare related to CHILD passengers.
- PassengerTypeFare infantFare. Fare related to INFANT passengers.
- List<Integer> firstSegments. List of segments for the first part of the route. In a one way itinerary, they will be the only present segments. To resolve the segment identifiers to Segment objects, refer to the ItinerariesLegend.
- List<Integer> secondSegments. List of segments for the second part of the route.
- List<Integer> thirdSegments. List of segments for the third part of the route.
- List<Integer> fourthSegments. List of segments for the fourth part of the route.
- List<Integer> fifthSegments. List of segments for the fifth part of the route.
- List<Integer> sixthSegments. List of segments for the sixth part of the route.
- List<String> firstSegmentsKeys. List of unique first segment keys inside this itinerary, in the same order as firstSegments, which must be used to identify this segment if selected later on.
- List<String> secondSegmentsKeys. Same as firstSegmentKeys but for second segment.
- List<String> thirdSegmentsKeys. Same as firstSegmentKeys but for third segment.
- List<String> forthSegmentsKeys. Same as firstSegmentKeys but for fouth segment.
- List<String> fifthSegmentsKeys. Same as firstSegmentKeys but for fifth segment.
- List<String> sixthSegmentsKeys. Same as firstSegmentKeys but for sixth segment.

* **PassengerTypeFare.** It represents combination of**:**
  - BigDecimal fare. Total fares. (Total price consists of total fares + total taxes)
  - BigDecimal tax. Total taxes.

  Both are considered for every passenger of a given type.
* **ItinerariesLegend**. Summary of the locations, segments and carriers involved in a set of itineraries, containing properties:
  - Collection<Carrier> carriers. Carriers involved, as defined before
  - Collection <Location> locations. Itinerary locations, with structure as defined above.
  - Collection<SegmentResult> segmentResults. Each SegmentResult object contains an identifier (id) and a Segment. Segment identifiers in FareItinerary objects can be resolved to Segments by matching the identifier with a SegmentResult found in this list.
  - Collection<SectionResult> sectionResults. Each SectionResult object contains an identifier (id) and a Section. Section identifiers in Segments can be resolved to Sections by looking up the id property in this list.
* **Segment.** A segment represents a single trip from a departure place to a destination, maybe containing stopovers, technical stops… The total number of stopovers including technical stops can be computed by adding the number of sections plus the number of technical stops in each section minus one. The segment object contains the following properties:
  - List<Integer> sections. List of section identifiers in the segment. Each segment is a single trip by plane, although it may contain technical stops, and these stops sometimes imply changing the plane in flight sections. A section identifier can be resolved to a Section by looking up in the ItinerariesLegend (see below).
  - Integer carrier. Carrier identifier for the airline. The full name of the company can be retrieved from the legend.
  - Long duration. Estimated journey duration in minutes.
  - Integer seats. Indicates the number of seats available for the segment, if known.
* **Section.** A section is a single trip inside a segment, without scales (but potentially technical stops). It contains:
  - String id. An identifier for the section, for flights is the flight number.
  - int from. Place the airplane departs from. This number refers to a location defined in the legend.
  - int to. Place the airplane arrives to. This number refers to a location defined in the legend.
  - Date departureDate. Date and time the aircraft departs from. Internal XML representation follows the standard ISO 8601 format CCYY-MM-DDThh:mm:ss, with implicit timezone in the departure place (from).

- Date arrivalDate. Date and time the aircraft arrives to. Internal XML representation follows the standard ISO 8601 format CCYY-MM-DDThh:mm:ss, with implicit timezone in the arrival place (to).
- String departureTerminal. Terminal the airport departs from.
- String arrivalTerminal. Terminal the airport arrives to.
- Integer carrier. Carrier identifier for the airline. The full name of the company can be retrieved from the legend.
- CabinClass cabinClass. Either BUSINESS, FIRST, PREMIUM_ECONOMY, ECONOMIC_DISCOUNTED or TOURIST, representing the type of cabin seat the section contains, if it differs from the user choice.
- List<TechnicalStop> technicalStops. List of known technical stops during this section.
- Int operatingCarrier. Carrier id that operates the flight, in case of codesharing (different from marketing carrier, indicated by “carrier” property). If empty, then it is not a codesharing flight.
- String aircraft. Type of plane that operates the route.
- BaggageAllowanceType baggageAllowanceType, enumeration for the baggage conditions specified by section. Possible values and meanings are: KG(Kilos), PO(Pounds), NI(Nil), SC(Special Charge), NP(Number of pieces), SZ(Size), VL(Value), WE(Weight). This value is only informed after selection in case the provider gives this information, not in the search results.
- Integer baggageAllowanceQuantity. Value of the baggage allowance conditions that refer to the baggage allowance type. This value is only informed after selection in case the provider gives this information, not in the search results.
- Long duration. Duration of the section in minutes.
- Boolean timesUndefined. This attribute will be true when the times are not defined for the segment (typically charter flights). Otherwise it will be null.
- FlightType flightType. It can have the following values:
  - ‘REG’ of regular.
  - ‘LC’ of lowcost.

- **Location.** Place (city, airport…) where an itinerary starts, ends, or has a stopover or technical stop. Contains:
  - String iataCode. IATA Code for the place, in case it exists.
  - String cityIataCode. IATA Code for the place.
  - String cityName. City name in current locale.
  - String countryCode. Country code.
  - String countryName; Country name in current locale.
  - LocationType locationType. Type of location represented: City
  - String name. Localized place name, for example, if it is an airport, the airport name.
- **Carrier.** Represents an airline company. It contains:
  - Integer id: Carrier identifier used in other objects.
  - String code. 2 letters standard IATA code for the carrier.
  - String name. Airline name.
- **TechnicalStop.** Represents an intermediate stop in a Section. It contains:
  - Date arrivalDate. Arrival date to the technical stop location. Internal XML representation follows the standard ISO 8601 format CCYY-MM-DDThh:mm:ss, with implicit timezone in the location the stop takes place.
  - Date departureDate. Departure date from the technical stop. Follows same date format as arrivalDate.
  - String equipmentType. Type of airplane involved.
  - int location. Location id where the plane stops. This id refers to a location defined in the legend. If the IATA code of the location is not recognized by the system (e.g a small airport that does not exist yet in the system) this field will be set to 0.

5.1.1.1. Request example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://b2b.odigeo.com/b2b/ws/v1">
  <soapenv:Header/>
  <soapenv:Body>
    <v1:search>
      <credentials partnerId="partner" password="partner2014"/>
      <userInfo userAgent="userAgent" realUserIp="10.92.15.15"/>
      <searchRequest>
        <itinerarySearchRequest maxSize="50" directFlightsOnly="true" cabinClass="TOURIST"
numAdults="1" numChildren="1" numInfants="1">
          <segmentRequests date="2014-06-24">
            <departure iataCode="BCN"/>
            <destination iataCode="PAR"/>
          </segmentRequests>
        </itinerarySearchRequest>
      </searchRequest>
    </v1:search>
  </soapenv:Body>
</soapenv:Envelope>
```

5.1.1.2. Response example

```
 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"/>
  <soap:Body>
    <ns2:searchResponse xmlns:ns2="http://b2b.odigeo.com/b2b/ws/v1">
      <searchResponse searchId="6673304719">
        <preferences currency="EUR" locale="en"/>
        <itinerarySearchResults cabinClass="TOURIST">
          <fareItineraryResults firstSegments="0 1 2 3 4 5 6 7 8 9" firstSegmentsKeys="0,IB5937
1,IB5240 2,IB5926 3,IB5393 4,IB5532 5,IB5276 6,IB5242 7,IB5854 8,IB5756 9,IB5930"
key="0,1A" totalPrice="216.47">
            <adultFare fare="48.17" tax="62.68"/>
            <childFare fare="20.55" tax="62.68"/>
            <infantFare fare="2.71" tax="19.68"/>
          </fareItineraryResults>
          <fareItineraryResults firstSegments="10 11 12 13" firstSegmentsKeys="0,VY8024
1,VY8008 2,VY8018 3,VY8020" key="1,1A" totalPrice="216.47">
            <adultFare fare="48.17" tax="62.68"/>
            <childFare fare="20.55" tax="62.68"/>
            <infantFare fare="2.71" tax="19.68"/>
          </fareItineraryResults>
          <fareItineraryResults firstSegments="14 15" firstSegmentsKeys="0,IB5999 1,IB5524"
key="2,1A" totalPrice="249.43">
            <adultFare fare="61.83" tax="62.68"/>
            <childFare fare="37.83" tax="62.68"/>
            <infantFare fare="4.73" tax="19.68"/>
          </fareItineraryResults>
          <fareItineraryResults firstSegments="16 17" firstSegmentsKeys="0,VY8010 1,VY8016"
key="3,1A" totalPrice="249.43">
            <adultFare fare="61.83" tax="62.68"/>
            <childFare fare="37.83" tax="62.68"/>
            <infantFare fare="4.73" tax="19.68"/>
          </fareItineraryResults>
          <fareItineraryResults firstSegments="18 19 20 21 22 23"
firstSegmentsKeys="0,VY8012 1,VY8242 2,VY8014 3,VY8244 4,VY8022 5,VY8028" key="4,1A"
totalPrice="289.58">
            <adultFare fare="137.34" tax="62.68"/>
            <childFare fare="4.80" tax="62.68"/>
            <infantFare fare="2.40" tax="19.68"/>
            </fareItineraryResults>
            <fareItineraryResults firstSegments="24 25 26 27 28 29"
firstSegmentsKeys="0,AF1449 1,AF1349 2,AF1649 3,AF1949 4,AF1049 5,AF1249" key="5,1A"
totalPrice="944.68">
              <adultFare fare="436.97" tax="60.68"/>
              <childFare fare="322.83" tax="60.68"/>
              <infantFare fare="43.84" tax="19.68"/>
            </fareItineraryResults>
            <fareItineraryResults firstSegments="30 31 32 33" firstSegmentsKeys="0,UX3249
1,UX3149 2,UX3649 3,UX3469" key="6,1A" totalPrice="1136.26">
              <adultFare fare="565.56" tax="60.68"/>
              <childFare fare="373.54" tax="60.68"/>
              <infantFare fare="56.12" tax="19.68"/>
            </fareItineraryResults>
            <fareItineraryResults firstSegments="34" firstSegmentsKeys="0,AF1149" key="7,1A"
totalPrice="1180.55">
              <adultFare fare="564.46" tax="60.68"/>
              <childFare fare="418.87" tax="60.68"/>
              <infantFare fare="56.18" tax="19.68"/>
            </fareItineraryResults>
            <fareItineraryResults firstSegments="35" firstSegmentsKeys="0,UX3471" key="8,1A"
totalPrice="1385.52">
              <adultFare fare="704.68" tax="60.68"/>
              <childFare fare="468.63" tax="60.68"/>
              <infantFare fare="71.17" tax="19.68"/>
            </fareItineraryResults>
            <fareItineraryResults firstSegments="36" firstSegmentsKeys="0,UX3495" key="9,1A"
totalPrice="1568.86">
              <adultFare fare="808.14" tax="60.68"/>
              <childFare fare="538.19" tax="60.68"/>
              <infantFare fare="81.49" tax="19.68"/>
            </fareItineraryResults>
            <fareItineraryResults firstSegments="37 38" firstSegmentsKeys="0,JL5318 1,JL5322"
key="10,1A" totalPrice="1646.11">
              <adultFare fare="857.61" tax="19.68"/>
              <childFare fare="642.83" tax="19.68"/>
              <infantFare fare="86.63" tax="19.68"/>
            </fareItineraryResults>
            <legend>
              <carriers code="IB" id="0" name="Iberia"/>
              <carriers code="VY" id="1" name="Vueling"/>
              <carriers code="AF" id="2" name="Air France"/>
              <carriers code="UX" id="3" name="Air Europa"/>
              <carriers code="JL" id="4" name="Japan Airlines"/>
              <locations cityIataCode="PAR" cityName="Paris" countryCode="FR"
countryName="France" iataCode="ORY" id="1391" locationType="AIRPORT" name="Orly"/>
              <locations cityIataCode="BCN" cityName="Barcelona" countryCode="ES"
countryName="Spain" iataCode="BCN" id="157" locationType="AIRPORT" name="El Prat"/>
              <locations cityIataCode="PAR" cityName="Paris" countryCode="FR"
countryName="France" iataCode="CDG" id="330" locationType="AIRPORT" name="Charles De
Gaulle"/>
              <sectionResults id="0">
                <section aircraft="320" arrivalDate="2014-06-24T08:45:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T07:00:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="5937"
operatingCarrier="1" to="1391"/>
              </sectionResults>
            <sectionResults id="1">
              <section aircraft="320" arrivalDate="2014-06-24T09:20:00" arrivalTerminal="3"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T07:30:00"
departureTerminal="1" duration="110" flightType="REG" from="157" id="5240"
operatingCarrier="1" to="330"/>
            </sectionResults>
            <sectionResults id="2">
              <section aircraft="320" arrivalDate="2014-06-24T10:15:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T08:35:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="5926"
operatingCarrier="1" to="1391"/>
            </sectionResults>
            <sectionResults id="3">
              <section aircraft="320" arrivalDate="2014-06-24T10:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T08:55:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="5393"
operatingCarrier="1" to="1391"/>
            </sectionResults>
            <sectionResults id="4">
              <section aircraft="320" arrivalDate="2014-06-24T16:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T15:00:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="5532"
operatingCarrier="1" to="1391"/>
            </sectionResults>
            <sectionResults id="5">
              <section aircraft="320" arrivalDate="2014-06-24T17:45:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T16:05:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="5276"
operatingCarrier="1" to="1391"/>
            </sectionResults>
            <sectionResults id="6">
              <section aircraft="320" arrivalDate="2014-06-24T18:50:00" arrivalTerminal="3"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T17:00:00"
departureTerminal="1" duration="110" flightType="REG" from="157" id="5242"
operatingCarrier="1" to="330"/>
            </sectionResults>
            <sectionResults id="7">
              <section aircraft="320" arrivalDate="2014-06-24T19:15:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T17:35:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="5854"
operatingCarrier="1" to="1391"/>
            </sectionResults>
            <sectionResults id="8">
              <section aircraft="320" arrivalDate="2014-06-24T21:10:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T19:25:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="5756"
operatingCarrier="1" to="1391"/>
            </sectionResults>
            <sectionResults id="9">
              <section aircraft="320" arrivalDate="2014-06-24T22:10:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T20:35:00"
departureTerminal="1" duration="95" flightType="REG" from="157" id="5930"
operatingCarrier="1" to="1391"/>
            </sectionResults>
            <sectionResults id="10">
              <section aircraft="320" arrivalDate="2014-06-24T10:15:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T08:35:00"
```

```
departureTerminal="1" duration="100" flightType="REG" from="157" id="8024" to="1391"/>
                </sectionResults>
                <sectionResults id="11">
                  <section aircraft="320" arrivalDate="2014-06-24T10:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T08:55:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="8008" to="1391"/>
                </sectionResults>
                <sectionResults id="12">
                  <section aircraft="320" arrivalDate="2014-06-24T16:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T15:00:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8018" to="1391"/>
                </sectionResults>
                <sectionResults id="13">
                  <section aircraft="320" arrivalDate="2014-06-24T19:15:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T17:35:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8020" to="1391"/>
                </sectionResults>
                <sectionResults id="14">
                  <section aircraft="320" arrivalDate="2014-06-24T07:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T06:00:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="5999"
operatingCarrier="1" to="1391"/>
                </sectionResults>
                <sectionResults id="15">
                  <section aircraft="320" arrivalDate="2014-06-24T12:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T10:55:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="5524"
operatingCarrier="1" to="1391"/>
                </sectionResults>
                <sectionResults id="16">
                  <section aircraft="320" arrivalDate="2014-06-24T07:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T06:00:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8010" to="1391"/>
                </sectionResults>
                <sectionResults id="17">
                  <section aircraft="320" arrivalDate="2014-06-24T12:40:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T10:55:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="8016" to="1391"/>
                </sectionResults>
                <sectionResults id="18">
                  <section aircraft="320" arrivalDate="2014-06-24T08:45:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T07:00:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="8012" to="1391"/>
                </sectionResults>
                <sectionResults id="19">
                  <section aircraft="320" arrivalDate="2014-06-24T09:20:00" arrivalTerminal="3"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T07:30:00"
departureTerminal="1" duration="110" flightType="REG" from="157" id="8242" to="330"/>
                </sectionResults>
                <sectionResults id="20">
                  <section aircraft="320" arrivalDate="2014-06-24T17:45:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T16:05:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8014" to="1391"/>
                </sectionResults>
                <sectionResults id="21">
                  <section aircraft="319" arrivalDate="2014-06-24T18:50:00" arrivalTerminal="3"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T17:00:00"
```

```
departureTerminal="1" duration="110" flightType="REG" from="157" id="8244" to="330"/>
            </sectionResults>
            <sectionResults id="22">
              <section aircraft="320" arrivalDate="2014-06-24T21:10:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T19:25:00"
departureTerminal="1" duration="105" flightType="REG" from="157" id="8022" to="1391"/>
            </sectionResults>
            <sectionResults id="23">
              <section aircraft="320" arrivalDate="2014-06-24T22:10:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="1" departureDate="2014-06-24T20:35:00"
departureTerminal="1" duration="95" flightType="REG" from="157" id="8028" to="1391"/>
            </sectionResults>
            <sectionResults id="24">
              <section aircraft="321" arrivalDate="2014-06-24T08:40:00" arrivalTerminal="2F"
cabinClass="PREMIUM_ECONOMY" carrier="2" departureDate="2014-06-24T06:45:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="1449" to="330"/>
            </sectionResults>
            <sectionResults id="25">
              <section aircraft="321" arrivalDate="2014-06-24T14:30:00" arrivalTerminal="2F"
cabinClass="PREMIUM_ECONOMY" carrier="2" departureDate="2014-06-24T12:35:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="1349" to="330"/>
            </sectionResults>
            <sectionResults id="26">
              <section aircraft="321" arrivalDate="2014-06-24T17:05:00" arrivalTerminal="2F"
cabinClass="PREMIUM_ECONOMY" carrier="2" departureDate="2014-06-24T15:10:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="1649" to="330"/>
            </sectionResults>
            <sectionResults id="27">
              <section aircraft="321" arrivalDate="2014-06-24T19:55:00" arrivalTerminal="2F"
cabinClass="PREMIUM_ECONOMY" carrier="2" departureDate="2014-06-24T18:00:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="1949" to="330"/>
            </sectionResults>
            <sectionResults id="28">
              <section aircraft="318" arrivalDate="2014-06-24T21:20:00" arrivalTerminal="2F"
cabinClass="PREMIUM_ECONOMY" carrier="2" departureDate="2014-06-24T19:25:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="1049" to="330"/>
            </sectionResults>
            <sectionResults id="29">
              <section aircraft="318" arrivalDate="2014-06-24T22:20:00" arrivalTerminal="2F"
cabinClass="PREMIUM_ECONOMY" carrier="2" departureDate="2014-06-24T20:25:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="1249" to="330"/>
            </sectionResults>
            <sectionResults id="30">
              <section aircraft="32S" arrivalDate="2014-06-24T08:40:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="3" departureDate="2014-06-24T06:45:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="3249"
operatingCarrier="2" to="330"/>
            </sectionResults>
            <sectionResults id="31">
              <section aircraft="32S" arrivalDate="2014-06-24T14:30:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="3" departureDate="2014-06-24T12:35:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="3149"
operatingCarrier="2" to="330"/>
            </sectionResults>
            <sectionResults id="32">
              <section aircraft="32S" arrivalDate="2014-06-24T17:05:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="3" departureDate="2014-06-24T15:10:00"
```

```
departureTerminal="1" duration="115" flightType="REG" from="157" id="3649"
operatingCarrier="2" to="330"/>
                </sectionResults>
                <sectionResults id="33">
                  <section aircraft="32S" arrivalDate="2014-06-24T21:20:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="3" departureDate="2014-06-24T19:25:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="3469"
operatingCarrier="2" to="330"/>
                </sectionResults>
                <sectionResults id="34">
                  <section aircraft="321" arrivalDate="2014-06-24T12:10:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="2" departureDate="2014-06-24T10:15:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="1149" to="330"/>
                </sectionResults>
                <sectionResults id="35">
                  <section aircraft="32S" arrivalDate="2014-06-24T22:20:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="3" departureDate="2014-06-24T20:25:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="3471"
operatingCarrier="2" to="330"/>
                </sectionResults>
                <sectionResults id="36">
                  <section aircraft="32S" arrivalDate="2014-06-24T12:10:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="3" departureDate="2014-06-24T10:15:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="3495"
operatingCarrier="2" to="330"/>
                </sectionResults>
                <sectionResults id="37">
                  <section aircraft="321" arrivalDate="2014-06-24T17:05:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="4" departureDate="2014-06-24T15:10:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="5318"
operatingCarrier="2" to="330"/>
                </sectionResults>
                <sectionResults id="38">
                  <section aircraft="321" arrivalDate="2014-06-24T19:55:00" arrivalTerminal="2F"
cabinClass="BUSINESS" carrier="4" departureDate="2014-06-24T18:00:00"
departureTerminal="1" duration="115" flightType="REG" from="157" id="5322"
operatingCarrier="2" to="330"/>
                </sectionResults>
                <segmentResults id="0">
                  <segment carrier="0" duration="105" sections="0"/>
                </segmentResults>
                <segmentResults id="1">
                  <segment carrier="0" duration="110" sections="1"/>
                </segmentResults>
                <segmentResults id="2">
                  <segment carrier="0" duration="100" sections="2"/>
                </segmentResults>
                <segmentResults id="3">
                  <segment carrier="0" duration="105" sections="3"/>
                </segmentResults>
                <segmentResults id="4">
                  <segment carrier="0" duration="100" sections="4"/>
                </segmentResults>
                <segmentResults id="5">
                  <segment carrier="0" duration="100" sections="5"/>
                </segmentResults>
                <segmentResults id="6">
```

```
    <segment carrier="0" duration="110" sections="6"/>
  </segmentResults>
  <segmentResults id="7">
    <segment carrier="0" duration="100" sections="7"/>
  </segmentResults>
  <segmentResults id="8">
    <segment carrier="0" duration="105" sections="8"/>
  </segmentResults>
  <segmentResults id="9">
    <segment carrier="0" duration="95" sections="9"/>
  </segmentResults>
  <segmentResults id="10">
    <segment carrier="1" duration="100" sections="10"/>
  </segmentResults>
  <segmentResults id="11">
    <segment carrier="1" duration="105" sections="11"/>
  </segmentResults>
  <segmentResults id="12">
    <segment carrier="1" duration="100" sections="12"/>
  </segmentResults>
  <segmentResults id="13">
    <segment carrier="1" duration="100" sections="13"/>
  </segmentResults>
  <segmentResults id="14">
    <segment carrier="0" duration="100" sections="14"/>
  </segmentResults>
  <segmentResults id="15">
    <segment carrier="0" duration="105" sections="15"/>
  </segmentResults>
  <segmentResults id="16">
    <segment carrier="1" duration="100" sections="16"/>
  </segmentResults>
  <segmentResults id="17">
    <segment carrier="1" duration="105" sections="17"/>
  </segmentResults>
  <segmentResults id="18">
    <segment carrier="1" duration="105" sections="18"/>
  </segmentResults>
  <segmentResults id="19">
    <segment carrier="1" duration="110" sections="19"/>
  </segmentResults>
  <segmentResults id="20">
    <segment carrier="1" duration="100" sections="20"/>
  </segmentResults>
  <segmentResults id="21">
    <segment carrier="1" duration="110" sections="21"/>
  </segmentResults>
  <segmentResults id="22">
    <segment carrier="1" duration="105" sections="22"/>
  </segmentResults>
  <segmentResults id="23">
    <segment carrier="1" duration="95" sections="23"/>
  </segmentResults>
  <segmentResults id="24">
    <segment carrier="2" duration="115" sections="24"/>
  </segmentResults>
  <segmentResults id="25">
```

```
              <segment carrier="2" duration="115" sections="25"/>
            </segmentResults>
            <segmentResults id="26">
              <segment carrier="2" duration="115" sections="26"/>
            </segmentResults>
            <segmentResults id="27">
              <segment carrier="2" duration="115" sections="27"/>
            </segmentResults>
            <segmentResults id="28">
              <segment carrier="2" duration="115" sections="28"/>
            </segmentResults>
            <segmentResults id="29">
              <segment carrier="2" duration="115" sections="29"/>
            </segmentResults>
            <segmentResults id="30">
              <segment carrier="3" duration="115" sections="30"/>
            </segmentResults>
            <segmentResults id="31">
              <segment carrier="3" duration="115" sections="31"/>
            </segmentResults>
            <segmentResults id="32">
              <segment carrier="3" duration="115" sections="32"/>
            </segmentResults>
            <segmentResults id="33">
              <segment carrier="3" duration="115" sections="33"/>
            </segmentResults>
            <segmentResults id="34">
              <segment carrier="2" duration="115" sections="34"/>
            </segmentResults>
            <segmentResults id="35">
              <segment carrier="3" duration="115" sections="35"/>
            </segmentResults>
            <segmentResults id="36">
              <segment carrier="3" duration="115" sections="36"/>
            </segmentResults>
            <segmentResults id="37">
              <segment carrier="4" duration="115" sections="37"/>
            </segmentResults>
            <segmentResults id="38">
              <segment carrier="4" duration="115" sections="38"/>
            </segmentResults>
          </legend>
        </itinerarySearchResults>
      </searchResponse>
    </ns2:searchResponse>
  </soap:Body>
</soap:Envelope>
```

5.1.2. Selection procedure (Shopping cart creation)

Once a search has been performed, user can select an itinerary to create the shopping cart through the operation *createShoppingCart*.

Return type	Name	Parameters	Description
ShoppingCartSummaryResponse	**createShoppingCart**	Credentials credentials, SelectionRequest	Selects an itinerary to be booked. If there is any error condition it

	selectionRequest	returns a typed error message.

- ShoppingCartSummaryResponse **createShoppingCart** (Credentials credentials, SelectionRequest selectionRequest). Selects an itinerary from the currently active search and constructs a shopping cart. The price of the selected itinerary may change after invoking this method. It is up to the partner to detect this possible price change and act accordingly if any action needs to be taken.
    - Receives a Credentials and a SelectionRequest object which is described in detail in next section.
    - Returns a ShoppingCartSummaryResponse object which is also described in next section.

5.1.2.1. Data objects in selection process

Selection requires a Credentials and a SelectionRequest object, and returns a ShoppingCartSummaryResponse object representing the status of the selection:

- **Credentials**. The same as defined above.
- **SelectionRequest**. It is a container for:
    - ItinerarySelectionRequest selection: Object that contains info about the itinerary that must be selected from a previous search.
    - long searchId. Search id to be selected. Must match last search in progress.
- **ItinerarySelectionRequest**. It represents an itinerary selection, containing the following properties:
    - String fareItineraryKey. Selected itinerary unique identifier inside current search. It must be equal to key attribute of selected FareItinerary element received in last SearchResponse.
    - String[] segmentKeys. Segment keys inside the selected itinerary, in the adequate order, being 0 the first choice for the first segment in the itinerary. For example, in a ROUND_TRIP itinerary, this array must contain two identifiers, indicating the first segment identifier in the outbound trip, and the second one for the inbound trip. These keys are read from the firstSegmentKeys, secondSegmentKeys, thirdSegmentKeys, fourthSegmentsKeys, fifthSegmentsKeys and sixthSegmentsKeys properties within the FareItinerary element. Note that different segment choices can be available for each trip, as same price can apply for different time schedules of different flights operated by same carrier.

- **ShoppingCartSummaryResponse**. Represents the result of the selection operation, merely a container for properties. It inherits from PreferencesAwareResponse.
    - ShoppingCart shoppingCart. Object that contains the itinerary summary, passengers, buyer, and other related information. More information below.
- **ShoppingCart**. Represents the items to be bought by the buyer, it is first populated with the selected itinerary, and filled in later on with passengers information, buyer data, payment status, etc. It contains the following properties:
    - long bookingId. Booking identifier that must be used for subsequent requests to identify this booking.
    - BigDecimal totalPrice. Price of all the items in the cart.
    - PassengerTypeFare adultFare. Fare related to ADULT passengers.
    - PassengerTypeFare childFare. Fare related to CHILD passengers.
    - PassengerTypeFare infantFare. Fare related to INFANT passengers.
    - BigDecimal baggageTotalFee. Total amount of the baggage fees included in this shopping cart. If passenger data (which includes baggage data) has not been submitted yet, the baggage fee is usually zero. Once baggage items are explicitly added per passenger, amount may increase, depending on the airline baggage policies.
    - ItineraryShoppingItem itineraryShoppingItem. Represents the itinerary item contained in the shopping cart.
    - BuyerInformationDescription requiredBuyerInformation: Describes the buyer information needed in order to confirm this reservation. Depending on the flight features, different data is required, and this object describes what is mandatory. This information will be available even if the buyer data has already been submitted. In case there is a data mismatch (i.e. there are missing required fields, and sending buyer data causes an exception to be

thrown, one could compare the required fields with the present values to detect where the discrepancy comes from):

- RequiredField needsName. Indicates if name information is required or optional.
- RequiredField needsLastNames. Indicates if last names information is required or optional.
- RequiredField needsMail. Indicates if mail information is required or optional.
- RequiredField needsIdentification. Indicates if identification information is required or optional (Passport, National document identifier…).
- RequiredField needsDateOfBirth. Indicates if birth date information is required or optional.
- RequiredField needsAddress. Indicates if address information is required or optional.
- RequiredField needsCityName. Indicates if city name information is required or optional.
- RequiredField needsStateName. Indicates if state name is required or optional.
- RequiredField needsZipCode. Indicates if zip code is required or optional.
- RequiredField needsPhoneNumber. Indicates if phone number is required or optional. A Phone object contains basically a country code and a number, as explained below.
- RequiredField needsAlternativePhoneNumber. Indicates if alternative phone number is required. A Phone object contains basically a country code and a number, as explained below.
- RequiredField needsCountryCode. Indicates if country code number is required. Country code following the ISO-3166 two letters format.

- RequiredField. Possible values for the required fields. These values are: MANDATORY and OPTIONAL.
- Set<FormOfIdentificationDescriptor> requiredFoidInformation. Describes the FormOfIdentification information needed in order to confirm this booking that has not been yet introduced. Depending on the carriers and suppliers, different FOID's can be required. This set will be available when some FOID information is missing.
- List<TravellerInformationDescription> requiredTravellerInformation. List of TravellerInformationDescription. More information below.
- Buyer buyer. More information regarding buyer below.
- List<Traveller> travellers. List of travellers information (personal data) for each passenger in the trip. It will not be available in the ShoppingCart until filled in by the buyer.
- List<FormOfIdentification> foids. Form of identification introduced in the shopping cart.

- **ItineraryShoppingItem**. Represents an itinerary shopping item. The basic properties are:
  - String id. Itinerary related identifier.
  - List<Integer> segments. Selected segment identifiers. Segment object structure is described above; in the search procedure, ids can be resolved to Segment objects using the legend attached to the itinerary.
  - List<FareComponent> fareComponents. List with information related with Fares
    - FareComponent: It has the following fields:
      - List<String> fareRules: Rules related with each fare.
      - String passengerType: Type of passenger
      - String origin: Origin related with this fare
      - String destination: Destination related with this fare.
  - ItinerariesLegend legend. Describes the locations, carriers, segments and sections present in the itinerary.
- **TravellerInformationDescription.** It describes the traveller information needed in order to confirm this reservation. Depending on the flight features, very different data is required, and this object describes what is necessary. The ShoppingCart will contain one of such descriptors for each passenger. This list will be available even if the passenger data has already been submitted. In case there is a data mismatch (i.e. there are missing required fields, and sending passenger data causes an exception to be thrown, one could compare the list of required fields with the present values to detect where the discrepancy comes from):

- TravellerType travellerType. One of ADULT, CHILD or INFANT, determining what is required for this type of passenger.
- BaggageConditions baggageConditions. Represent the baggage conditions available for this passenger, as described next.
- Set<IdentificationType> identificationTypes. One of PASSPORT, NATIONAL_ID_CARD representing the choice of type of identification that is allowed for this passenger. It may vary significantly, depending on whether or not the passenger is adult, or the flight being local or international. PASSPORT and NATIONAL_ID_CARD are available to any itinerary, NIE and NIF are only for Spanish bookings.
- RequiredField needsTitle. Set to true if title information is required, that is Mr, Ms or Mrs. See TravelerRequest object definition later on.
- RequiredField needsName. Indicates if name information is required or optional.
- RequiredField needsNationality. Indicates if nationality information is required or optional.
- RequiredField needsGender. Indicates if gender information is required or optional.
- RequiredField needsBirthDate. Indicates if birth date information is required or optional.
- RequiredField needsCountryOfResidence. Indicates if country of residence information is required or optional.
- RequiredField needsIdentification. Indicates if identification information is required or optional (Passport, National ID Card …).
- RequiredField needsIdentificationExpirationDate. Indicates if identification information expiration date information is required or optional.
- RequiredField needsIdentificationCountry. Indicates if identification country information is required or optional.
- Set<AirlineGroup> frequentFlyerAirlineGroups. Set of groups of airlines that the user may be asked for frequent flyer card numbers. AirlineGroup is a container for a Set of Airlines (Carriers). User may choose one frequent flyer program from each group.
- Set<TravellerMeal> availableMeals. Set of possible TravellerMeals offered.

```
<requiredTravellerInformation needsIdentificationCountry="OPTIONAL"
needsIdentificationExpirationDate="OPTIONAL"
needsIdentification="OPTIONAL" needsCountryOfResidence="OPTIONAL"
needsBirthDate="OPTIONAL" needsNationality="OPTIONAL"
needsGender="OPTIONAL" needsName="MANDATORY"
needsTitle="MANDATORY" travellerType="ADULT">
```

- **TravellerMeal**: Enumeration with the possible traveler meals values. These values are: STANDARD, KOSHER, HALAL, LOW_CHOLESTEROL, LOW_SODIUM, VEGAN, VEGETARIAN, BABY, CHILD, DIABETIC, GLUTEN_FREE and HINDU.
- **FormOfIdentification**. Represent a form of identification introduced. Contains the following attributes:
  - IdentificationType type. Form of identification type, possible values are NATIONAL_ID_CARD, PASSPORT.
  - String value. Form of identification value.
- **Buyer**. Buyer data, with the following properties:
  - String name. Buyer first name.
  - String lastNames. Buyer last names.
  - String mail. Buyer email.
  - Date dateOfBirth. Passenger birth date, following the ISO 8601 format: YYYY-MM-DD.
  - IdentificationType buyerIdentificationType. One of NATIONAL_ID_CARD, PASSPORT, depending on the type of identification.
  - String identification. Buyer identification value.
  - String address. Buyer address.
  - String cityName. Buyer city address.
  - String stateName. Buyer state address.
  - String zipCode. Buyer zipcode address.
  - Phone phoneNumber. First phone number.
  - Phone alternativePhoneNumber. Second phone number.

  - String countryCode. Buyer country of residence, following the ISO-3166 two letters format.
- **Traveller**. It contains the following fields.
  - TravellerType travellerType. Type of passenger, one of ADULT, CHILD or INFANT.
  - TravellerTitle titleName. Passenger title, one of MR, MRS, MS.
  - String name. Passenger name.
  - String middleName. Passenger middle name
  - String firstLastName. Passenger first last name.
  - String secondLastName. Passenger second last name.
  - TravellerGender travellerGender. Passenger gender, one of MALE, FEMALE.
  - Date dateOfBirth. Passenger birth date, following the ISO 8601 format: YYYY-MM-DD.
  - String nationalityCountryCode. Country code for the passenger nationality, following the ISO-3166 two letters format.
  - String countryCodeOfResidence. Country code for the passenger residence following the ISO-3166 two letters format.
  - IdentificationType identificationType. Type of identification, one of NATIONAL_ID_CARD or PASSPORT.
  - String identification. Identification value, that is, the passport number for example, if the chosen identification document was passport.
  - Date identificationExpirationDate. Expiration date for the identification document, following the ISO 8601 format: YYYY-MM-DD.
  - String identificationIssueCountryCode. Country code for the identification document issuer following the ISO-3166 two letters format.
- **FormOfIdentificationDescriptor**. Each form of identification descriptor contains a Set of possible IdentificationTypes for this descriptor. The possible values are NATIONAL_ID_CARD, PASSPORT.
- **Phone**. Represents a phone number, consisting of:
  - String countryCode. Country the number belongs to in two letters code.
  - String number. Telephone number, without country code.
- **BaggageDescriptor.** It represents the number of baggages or kilos quantity, or both of them.
  - Int pieces: num of baggages
  - Int kilos: num of kilos.
- **BaggageApplicationLevel**. Enumeration that has 3 possible values:
  - SECTION
  - SEGMENT
  - ITINERARY
- **BaggageConditions.** Baggage conditions represent the baggage fees that some airlines may charge to the passengers. It contains the following properties**:**
  - SelectableBaggageDescriptor baggageFees. List of selectable baggage descriptors available. Each BaggageFee object contains two properties:
    - BaggageDescriptor baggageDescriptor. Number of bags/kilos.
    - BigDecimal fee. Fee for the specified baggageDescriptor.
  - BaggageDescriptor baggageDescriptorIncludedInPrice. Number of baggage pieces/kilos already included in the price.
  - BaggageApplicationLevel: application for the baggage as defined before.
  - SegmentTypeIndex: only apply when the baggage application level is segment.

  For example, possible representations could be:

```
<baggageConditions baggageIncludedInPrice="0">
        <baggageFees fee="0.00" numBags="0"/>
        <baggageFees fee="11.00" numBags="1"/>
</baggageConditions>
```

In this case, only one baggage piece can be carried at an additional cost of 11 €.

```
<baggageConditions baggageIncludedInPrice="1">
        <baggageFees fee="0.00" numBags="0"/>
        <baggageFees fee="0.00" numBags="1"/>
           <baggageFees fee="10.00" numBags="2"/>
```

```
            <baggageFees fee="30.00" numBags="3"/>
</baggageConditions>
```

In this case, up to three baggage pieces can be carried. The first one is free, but one extra bag costs 10€, and two extra bags cost 30€

5.1.2.1. Request example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://b2b.odigeo.com/b2b/ws/v1">
  <soapenv:Header/>
  <soapenv:Body>
    <v1:createShoppingCart>
      <credentials partnerId="partner" password="partner2014"/>
      <selectionRequest searchId="${search#Response#//searchResponse/@searchId}">
        <itinerarySelection fareItineraryKey="1" segmentKeys="0"/>
      </selectionRequest>
    </v1:createShoppingCart>
  </soapenv:Body>
</soapenv:Envelope>
```

5.1.2.2. Response example

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"/>
  <soap:Body>
    <ns2:createShoppingCartResponse xmlns:ns2="http://b2b.odigeo.com/b2b/ws/v1">
      <createShoppingCart>
        <preferences currency="EUR" locale="en"/>
        <shoppingCart baggageTotalFee="0.00" bookingId="428505855" totalPrice="216.47">
          <adultFare fare="48.17" tax="62.68"/>
          <childFare fare="20.55" tax="62.68"/>
          <infantFare fare="2.71" tax="19.68"/>
          <itineraryShoppingItem id="0" segments="0">
            <legend>
              <carriers code="VY" id="0" name="Vueling"/>
              <locations cityIataCode="PAR" cityName="Paris" countryCode="FR"
countryName="France" iataCode="ORY" id="1391" locationType="AIRPORT" name="Orly"/>
              <locations cityIataCode="BCN" cityName="Barcelona" countryCode="ES"
countryName="Spain" iataCode="BCN" id="157" locationType="AIRPORT" name="El Prat"/>
              <sectionResults id="0">
                <section aircraft="320" arrivalDate="2014-06-24T10:15:00" arrivalTerminal="W"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T08:35:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8024" to="1391"/>
              </sectionResults>
              <segmentResults id="0">
                <segment carrier="0" duration="100" sections="0"/>
              </segmentResults>
            </legend>
          </itineraryShoppingItem>
          <requiredTravellerInformation needsBirthDate="OPTIONAL"
needsCountryOfResidence="OPTIONAL" needsGender="OPTIONAL"
needsIdentification="OPTIONAL" needsIdentificationCountry="OPTIONAL"
needsIdentificationExpirationDate="OPTIONAL" needsName="MANDATORY"
needsNationality="OPTIONAL" needsTitle="MANDATORY" travellerType="ADULT">
            <identificationTypes>NATIONAL_ID_CARD</identificationTypes>
            <identificationTypes>PASSPORT</identificationTypes>
            <baggageConditions baggageApplicationLevel="SEGMENT"
segmentTypeIndex="FIRST">
              <baggageDescriptorIncludedInPrice pieces="1"/>
            </baggageConditions>
            <availableMeals>HALAL</availableMeals>
            <availableMeals>DIABETIC</availableMeals>
            <availableMeals>HINDU</availableMeals>
            <availableMeals>STANDARD</availableMeals>
            <availableMeals>VEGETARIAN</availableMeals>
            <availableMeals>BABY</availableMeals>
            <availableMeals>LOW_CHOLESTEROL</availableMeals>
            <availableMeals>LOW_SODIUM</availableMeals>
            <availableMeals>KOSHER</availableMeals>
            <availableMeals>CHILD</availableMeals>
            <availableMeals>GLUTEN_FREE</availableMeals>
            <availableMeals>VEGAN</availableMeals>
          </requiredTravellerInformation>
          <requiredTravellerInformation needsBirthDate="MANDATORY"
needsCountryOfResidence="OPTIONAL" needsGender="OPTIONAL"
needsIdentification="OPTIONAL" needsIdentificationCountry="OPTIONAL"
needsIdentificationExpirationDate="OPTIONAL" needsName="MANDATORY"
needsNationality="OPTIONAL" needsTitle="MANDATORY" travellerType="INFANT">
            <identificationTypes>NATIONAL_ID_CARD</identificationTypes>
            <identificationTypes>PASSPORT</identificationTypes>
            <availableMeals>HALAL</availableMeals>
            <availableMeals>DIABETIC</availableMeals>
            <availableMeals>HINDU</availableMeals>
            <availableMeals>STANDARD</availableMeals>
            <availableMeals>VEGETARIAN</availableMeals>
            <availableMeals>BABY</availableMeals>
            <availableMeals>LOW_CHOLESTEROL</availableMeals>
            <availableMeals>LOW_SODIUM</availableMeals>
            <availableMeals>KOSHER</availableMeals>
            <availableMeals>CHILD</availableMeals>
            <availableMeals>GLUTEN_FREE</availableMeals>
            <availableMeals>VEGAN</availableMeals>
          </requiredTravellerInformation>
          <requiredTravellerInformation needsBirthDate="MANDATORY"
needsCountryOfResidence="OPTIONAL" needsGender="OPTIONAL"
needsIdentification="OPTIONAL" needsIdentificationCountry="OPTIONAL"
needsIdentificationExpirationDate="OPTIONAL" needsName="MANDATORY"
needsNationality="OPTIONAL" needsTitle="MANDATORY" travellerType="CHILD">
            <identificationTypes>NATIONAL_ID_CARD</identificationTypes>
            <identificationTypes>PASSPORT</identificationTypes>
            <baggageConditions baggageApplicationLevel="SEGMENT"
segmentTypeIndex="FIRST">
              <baggageDescriptorIncludedInPrice pieces="1"/>
            </baggageConditions>
            <availableMeals>HALAL</availableMeals>
            <availableMeals>DIABETIC</availableMeals>
            <availableMeals>HINDU</availableMeals>
            <availableMeals>STANDARD</availableMeals>
            <availableMeals>VEGETARIAN</availableMeals>
            <availableMeals>BABY</availableMeals>
            <availableMeals>LOW_CHOLESTEROL</availableMeals>
            <availableMeals>LOW_SODIUM</availableMeals>
            <availableMeals>KOSHER</availableMeals>
            <availableMeals>CHILD</availableMeals>
            <availableMeals>GLUTEN_FREE</availableMeals>
            <availableMeals>VEGAN</availableMeals>
          </requiredTravellerInformation>
          <requiredBuyerInformation needsAddress="MANDATORY"
needsAlternativePhoneNumber="OPTIONAL" needsCityName="MANDATORY"
needsCountryCode="MANDATORY" needsDateOfBirth="OPTIONAL"
needsIdentification="OPTIONAL" needsLastNames="MANDATORY" needsMail="MANDATORY"
needsName="MANDATORY" needsPhoneNumber="MANDATORY"
needsStateName="OPTIONAL" needsZipCode="MANDATORY">
            <identificationTypes>NATIONAL_ID_CARD</identificationTypes>
            <identificationTypes>PASSPORT</identificationTypes>
          </requiredBuyerInformation>
        </shoppingCart>
      </createShoppingCart>
    </ns2:createShoppingCartResponse>
  </soap:Body>
</soap:Envelope>
```

5.1.3. Passengers information

After selecting an itinerary and retrieving the ShoppingCartSummaryResponse object, a client knows which user data is required in order to continue the booking procedure. This ShoppingCartSummaryResponse object contains the shopping cart, with a list of TravellerInformationDescription objects that define which user data is expected, a summary of the itinerary to be booked, and buyer and passenger data, once submitted by the client:

Return type	Name	Parameters	Description
ShoppingCartSummaryResponse	**addPersonalInfoToShoppingCart**	Credentials credentials, PersonalInfoRequest personalInfo	Submits passenger data to be stored in the ShoppingCart. If there is any error condition it returns a typed error message.

- ShoppingCartSummaryResponse **addPersonalInfoToShoppingCart** (Credentials credentials, PersonalInfoRequest personalInfo). Submits passenger data related to a shopping cart. The selected itinerary price may change as a result of this method invocation. It is up to the partner to detect this event and if any special action needs to be taken, act accordingly.
  - Receives a PersonalInfoRequest object representing the passenger and buyer user data to be considered in the booking process.
  - Returns a ShoppingCartSummaryResponse object (see above for full description).

5.1.3.1. Data objects in passenger information submission

Submission of passenger data requires a Credentials and a PersonalInfoRequest object, and returns a ShoppingCartSummaryResponse object representing the status of the operation:

- **Credentials**. The same as defined above.
- **PersonalInfoRequest**. It contains the following properties:
  - long bookingId. Must be set to the ShoppingCart identifier returned by previous invocation to createShoppingCart.
  - List<TravellerRequest> traveller. List of Traveller objects. It must contain the same number and type of passengers that were requested in passenger search. They must also follow the same order that was indicated by the TravellerInformationDescription. Each object represents one passenger data.
  - BuyerRequest buyer. Buyer personal data.

- **BaggageSelection**. It contains the following properties.
  - BaggageDescriptor baggageDescriptor. The descriptor representing a baggage selection.
  - SegmentTypeIndex. The segment.
- **TravellerRequest**. Traveler user data, with the following properties. It must contain at least the required attributes as indicated in the TravellerInformationDescription:
  - String travellerType. ADULT, CHILD or INFANT.
  - String titleName. Traveler title, one of MR, MRS or MS.
  - String name. Passenger first name.
  - String middleName. Passenger middle name.
  - String firstLastName. Passenger first last name.
  - String secondLastName. Passenger second last name.
  - String gender. One of MALE or FEMALE.
  - Date dateOfBirth. Passenger birth date, following the ISO 8601 format: YYYY-MM-DD.
  - IdentificationType buyerIdentificationType. One of NATIONAL_ID_CARD, PASSPORT, depending on the type of identification.
  - String nationalityCountryCode. Country code where the passenger was born, following the ISO-3166 two letters format.
  - String countryCodeOfResidence. Country code where the passenger lives, following the ISO-3166 two letters format.
  - String identification. Identification number or code.
  - String identificationTypeName. Type of identification, PASSPORT or
  - NATIONAL_ID_CARD.
  - String mealName. Traveler meal name, STANDARD, KOSHER, HALAL,
  - LOW_CHOLESTEROL, LOW_SODIUM, VEGAN, VEGETARIAN, BABY, CHILD, DIABETIC, GLUTEN_FREE or HINDU
  - Date identificationExpirationDate. Identification expiration date, following the ISO 8601 format.
  - String identificationIssueContryCode. Country code where the identification document was issued, following the ISO-3166 two letters format.
  - String localityCodeOfResidence. When booking refers to a resident of a region subject to discount, this is the town code. If search did not refer to a location subject to resident fares, or the client did not enable resident search, this attribute can be safely ignored.
  - Sel<BaggageSelectionRequest> baggageSelectionRequest. Baggage selections for the traveler.
  - Set<FrequentFlyerCardCode> frequentFlyerAirlineCodes. List of frequent flyer codes.
- **FrequentFlyerCardCode.** It represents a frequent flyer card code supplied by the user. It contains two properties**:**
  - String carrierCode. Carrier code, for example, JK or IB
  - String passengerCardNumber. Passenger card number. It might be an alphanumeric code depending on the carrier, even if it is suffixed as “number”.
- **BuyerRequest**. Buyer user data, with the following properties:
  - String name. Buyer first name.
  - String lastNames. Buyer last names.
  - String mail. Buyer email.
  - String identification. Buyer identification code.
  - Date dateOfBirth. Passenger birth date, following the ISO 8601 format: YYYY-MM-DD.
  - IdentificationType buyerIdentificationType. One of NATIONAL_ID_CARD, PASSPORT, depending on the type of identification.
  - String buyerIdentificationTypeName. One of NATIONAL_ID_CARD, PASSPORT or NIF, depending on the type of identification.
  - String address. Buyer address.
  - String cityName. Buyer city address.
  - String stateName. Buyer state address.
  - String zipCode. Buyer zipcode address.
  - Phone phoneNumber. First phone number.
  - Phone alternativePhoneNumber. Second phone number.

   - String countryPhoneNumber. Country code for the first phone number, following the ISO-3166 two letters format.
   - String alternativeCountryPhoneNumber. Country code for the alternative phone number, following the ISO-3166 two letters format.
   - String countryCode. Buyer country of residence, following the ISO-3166 two letters format.
- **ShoppingCartSummaryResponse**. It is the same type of object returned by the selection method *createShoppingCart*. After *addPersonalInfoToShoppingCart* completes successfully, it will contain passenger and buyer data inside the shopping cart.

5.1.3.2. Request example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://b2b.odigeo.com/b2b/ws/v1">
  <soapenv:Header/>
  <soapenv:Body>
    <v1:addPersonalInfoToShoppingCart>
      <credentials partnerId="partner" password="partner2014"/>
      <personalInfo
bookingId="${createShoppingCart#Response#//createShoppingCart/shoppingCart/@bookingId}">
        <buyer name="nameBuyer" lastNames="test" mail="Buyer@Buyer.com"
buyerIdentificationType="PASSPORT" identification="1111111111" dateOfBirth="1987454-05-05"
address="test" cityName="test" stateName="test" zipCode="00050" countryCode="ES">
          <phoneNumber countryCode="ES" number="655252525"/>
          <alternativePhoneNumber countryCode="ES" number="655252525"/>
        </buyer>
        <travellers dateOfBirth="1987-05-05" identificationExpirationDate="2015-10-10"
travellerType="ADULT" titleName="MR" name="TESTA" firstLastName="TEST"
secondLastName="TEST" middleName="test" gender="MALE" nationalityCountryCode="ES"
countryCodeOfResidence="ES" identification="1111111111" identificationType="PASSPORT"
identificationIssueCountryCode="ES" numBags="0" mealName="STANDARD">
          <baggageSelection segmentTypeIndex="FIRST">
            <baggageDescriptor pieces="1"/>
          </baggageSelection>
        </travellers>
        <travellers dateOfBirth="2013-05-05" identificationExpirationDate="2015-10-10"
travellerType="INFANT" titleName="MR" name="TESTB" firstLastName="TEST"
secondLastName="TEST" middleName="TEST" gender="MALE" nationalityCountryCode="ES"
countryCodeOfResidence="ES" identification="1111111111" identificationType="PASSPORT"
identificationIssueCountryCode="ES" numBags="0" mealName="STANDARD"></travellers>
        <travellers dateOfBirth="2005-05-05" identificationExpirationDate="2015-10-10"
travellerType="CHILD" titleName="MR" name="TESTC" firstLastName="TEST"
secondLastName="TEST" middleName="TEST" gender="MALE" nationalityCountryCode="ES"
countryCodeOfResidence="ES" identification="1111111111" identificationType="PASSPORT"
identificationIssueCountryCode="ES" numBags="0" mealName="STANDARD"></travellers>
      </personalInfo>
    </v1:addPersonalInfoToShoppingCart>
  </soapenv:Body>
</soapenv:Envelope>
```

5.1.3.3. Response example

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"/>
  <soap:Body>
    <ns2:addPersonalInfoToShoppingCartResponse
```

```
xmlns:ns2="http://b2b.odigeo.com/b2b/ws/v1">
      <passengerDataResponse>
        <preferences currency="EUR" locale="en"/>
        <shoppingCart baggageTotalFee="0.00" bookingId="428505855" totalPrice="216.47">
          <adultFare fare="48.17" tax="62.68"/>
          <childFare fare="20.55" tax="62.68"/>
          <infantFare fare="2.71" tax="19.68"/>
          <itineraryShoppingItem id="0" segments="0">
            <legend>
              <carriers code="VY" id="0" name="Vueling"/>
              <locations cityIataCode="PAR" cityName="Paris" countryCode="FR"
countryName="France" iataCode="ORY" id="1391" locationType="AIRPORT" name="Orly"/>
              <locations cityIataCode="BCN" cityName="Barcelona" countryCode="ES"
countryName="Spain" iataCode="BCN" id="157" locationType="AIRPORT" name="El Prat"/>
              <sectionResults id="0">
                <section aircraft="320" arrivalDate="2014-06-24T10:15:00" arrivalTerminal="W"
baggageAllowanceQuantity="1" baggageAllowanceType="NP"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T08:35:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8024" to="1391"/>
              </sectionResults>
              <segmentResults id="0">
                <segment carrier="0" duration="100" sections="0"/>
              </segmentResults>
            </legend>
          </itineraryShoppingItem>
          <buyer address="test" buyerIdentificationType="PASSPORT" cityName="test"
countryCode="ES" dateOfBirth="1987454-05-05" identification="1111111111" lastNames="test"
mail="Buyer@Buyer.com" name="nameBuyer" stateName="test" zipCode="00050">
            <phoneNumber countryCode="ES" number="655252525"/>
            <alternativePhoneNumber countryCode="ES" number="655252525"/>
          </buyer>
          <travellers countryCodeOfResidence="ES" dateOfBirth="1987-05-05"
firstLastName="TEST" gender="MALE" identification="1111111111"
identificationExpirationDate="2015-10-10" identificationIssueCountryCode="ES"
identificationType="PASSPORT" mealName="STANDARD" middleName="test" name="TESTA"
nationalityCountryCode="ES" secondLastName="TEST" titleName="MR" travellerType="ADULT">
            <baggageSelection segmentTypeIndex="FIRST">
              <baggageDescriptor pieces="1"/>
            </baggageSelection>
          </travellers>
          <travellers countryCodeOfResidence="ES" dateOfBirth="2013-05-05"
firstLastName="TEST" gender="MALE" identification="1111111111"
identificationExpirationDate="2015-10-10" identificationIssueCountryCode="ES"
identificationType="PASSPORT" mealName="STANDARD" middleName="TEST" name="TESTB"
nationalityCountryCode="ES" secondLastName="TEST" titleName="MR"
travellerType="INFANT"/>
          <travellers countryCodeOfResidence="ES" dateOfBirth="2005-05-05"
firstLastName="TEST" gender="MALE" identification="1111111111"
identificationExpirationDate="2015-10-10" identificationIssueCountryCode="ES"
identificationType="PASSPORT" mealName="STANDARD" middleName="TEST" name="TESTC"
nationalityCountryCode="ES" secondLastName="TEST" titleName="MR" travellerType="CHILD"/>
        </shoppingCart>
      </passengerDataResponse>
    </ns2:addPersonalInfoToShoppingCartResponse>
  </soap:Body>
</soap:Envelope>
```

5.1.4. Payment details and confirmation

After searching, navigating, selecting an itinerary, and submitting passenger data, the trip may be purchased. To do so, payment information is required. The price may have changed after baggage information is added, and other products are acquired, and also depending on the payment method selected. Payment operation is:

Return type	Name	Parameters	Description
BookingResponse	**confirmBooking**	Credentials credentials, PaymentDataRequest paymentDataRequest, long bookingId, String partnerReferenceId	Submits payment data to be stored in the ShoppingCart, and confirms booking.

- BookingResponse **confirmBooking** (Credentials credentials, PaymentDataRequest paymentDataRequest, long bookingId, String partnerReferenceId). Confirms booking. Price may change, and it is up to the partner to detect it and validate new price with user. After that it should be needed to call confirmBooking again.
  - Receives a BookingRequest a container object that contains the payment method to use, a bookingId identifier that must come from the shopping cart in previous steps, and a partnerReferenceId that may be the booking reference in the partner's systems, and be used later on in order to retrieve the booking information.
  - Returns a BookingResponse object (see above for full description).

5.1.4.1. Data objects in payment details and confirmation

Submission of payment data requires a PaymentDataRequest object, booking id and external partner reference id. It returns a BookingResponse object representing the status of the operation.

- **Credentials**. The same as defined before.
- **PaymentDataRequest.** It contains the following properties:
  - CollectionMethodType collectionMethodType. Type of payment method.
    - PREPAY ('prepay')
    - CREDITCARD ('creditcard')
  - CreditCardCollectionDetailsParametersRequest creditCardRequest. Credit card user data.
  - PrePayParametersRequest prePayParametersRequest. Prepay container.
- long **bookingId**. Odigeo booking identifier that has been given in a previous step
- String **partnerReferenceId**. Reference identifier that partner wants to give to this booking.
- **CreditCardCollectionDetailsParametersRequest**. Contains the credit card payment information supplied by the customer:
  - String cardTypeCode. Type of credit card code.
  - String cardOwner. Card owner.
  - String cardNumber. Card number
  - String cardSecurityNumber. Card security number.
  - String cardExpirationMonth. Card expiration month in two digits format.
  - String cardExpirationYear. Card expiration year in two digits format.
- **PrePayParametersRequest** prepayParameterRequest: Container related with form of payment used when payment has been done previously through a bank account, deposit or another method agreed with Odigeo.
- **BookingResponse**. It represents the result of a booking operation, that inherits from PreferencesAwareResponse containing:
  - boolean processed. Flag indicating whether or not the booking operation was successful
  - BookingResponseStatus status. Indicator of status condition. Possible values are :
    - BOOKING_ERROR. Internal server error, not recoverable.
    - BOOKING_PAYMENT_RETRY. Confirmation should be retried, a recoverable error has happened, for example, payment data was invalid, but booking could

eventually be completed with different data (for example, a new credit card number).
    - BOOKING_CONFIRMED. Booking is confirmed.
    - BOOKING_REPRICING. Price has changed since last shown; user might need to be asked for confirmation. Partner should inform the customer about this condition, and eventually re-confirm payment data to commit the booking.
    - BROKEN_FLOW. Booking flow natural order has been altered (for example, confirmation happened before sending passenger data), and booking cannot be completed. This is a non-recoverable error.
- ShoppingCart shoppingCart. In case the booking was not processed yet, the shopping cart is presented. Depending on the error condition, payment data may need to be resubmitted.
- BookingProcessingSummaryResponse bookingProcessingSummaryResponse. In case the booking was not completely processed, this object represents the booking status as it is being processed.
    - BookingSummary bookingSummary. Booking features, including itinerary, passengers, etc. as described next.

5.1.4.2. Request example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://b2b.odigeo.com/b2b/ws/v1">
  <soapenv:Header/>
  <soapenv:Body>
    <v1:confirmBooking>
      <credentials partnerId=“partner” password=“partner2014”/>
      <bookingRequest collectionMethodType="PREPAY">
      </bookingRequest>
<bookingId>${addPersonalInfoToShoppingCart#Response#//passengerDataResponse/shoppingCart/
@bookingId}</bookingId>
      <partnerReferenceId>partner-ref001</partnerReferenceId>
    </v1:confirmBooking>
  </soapenv:Body>
</soapenv:Envelope>
```

5.1.4.3. Response example

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"/>
  <soap:Body>
    <ns2:confirmBookingResponse xmlns:ns2="http://b2b.odigeo.com/b2b/ws/v1">
      <confirmResponse>
        <status>BOOKING_CONFIRMED</status>
        <shoppingCart/>
        <bookingProcessingSummaryResponse>
          <processed>true</processed>
          <bookingSummary bookingId="428505855" bookingStatus="REQUEST"
partnerReferenceId="marketing-test-2" totalPrice="216.47">
            <adultFare fare="48.17" tax="62.68"/>
            <childFare fare="20.55" tax="62.68"/>
            <infantFare fare="2.71" tax="19.68"/>
            <buyer address="test" buyerIdentificationType="PASSPORT" cityName="test"
countryCode="ES" dateOfBirth="9346-05-04" identification="1111111111" lastNames="TEST"
mail="buyer@buyer.com" name="NAMEBUYER" stateName="test" zipCode="00050">
              <phoneNumber countryCode="ES" number="655252525"/>
              <alternativePhoneNumber countryCode="ES" number="655252525"/>
            </buyer>
            <travellerResponses countryCodeOfResidence="ES" dateOfBirth="1987-05-05"
firstLastName="TEST TEST" identification="1111111111" identificationExpirationDate="2015-10-10"
identificationIssueCountryCode="ES" identificationType="PASSPORT" mealName="STANDARD"
middleName="test" name="TESTA" nationalityCountryCode="ES" titleName="MR"
travellerType="ADULT">
              <baggageSelection segmentTypeIndex="FIRST">
                <baggageDescriptor kilos="0" pieces="1"/>
              </baggageSelection>
            </travellerResponses>
            <travellerResponses countryCodeOfResidence="ES" dateOfBirth="2013-05-05"
firstLastName="TEST TEST" identification="1111111111" identificationExpirationDate="2015-10-10"
identificationIssueCountryCode="ES" identificationType="PASSPORT" mealName="STANDARD"
middleName="TEST" name="TESTB" nationalityCountryCode="ES" titleName="MR"
travellerType="INFANT"/>
            <travellerResponses countryCodeOfResidence="ES" dateOfBirth="2005-05-05"
firstLastName="TEST TEST" identification="1111111111" identificationExpirationDate="2015-10-10"
identificationIssueCountryCode="ES" identificationType="PASSPORT" mealName="STANDARD"
middleName="TEST" name="TESTC" nationalityCountryCode="ES" titleName="MR"
travellerType="CHILD"/>
            <itineraryBookings>
              <bookings bookingStatus="REQUEST" numAdults="1" numChildren="1" numInfants="1"
pnr="6QW84Q" segments="0">
                <legend>
                  <carriers code="VY" id="0" name="Vueling"/>
                  <locations cityIataCode="PAR" cityName="Paris" countryCode="FR"
countryName="France" iataCode="ORY" id="1391" locationType="AIRPORT" name="Orly"/>
                  <locations cityIataCode="BCN" cityName="Barcelona" countryCode="ES"
countryName="Spain" iataCode="BCN" id="157" locationType="AIRPORT" name="El Prat"/>
                  <sectionResults id="0">
                    <section aircraft="320" arrivalDate="2014-06-24T10:15:00" arrivalTerminal="W"
baggageAllowanceQuantity="1" baggageAllowanceType="NP"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T08:35:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8024" to="1391"/>
                  </sectionResults>
                  <segmentResults id="0">
                    <segment carrier="0" duration="100" sections="0"/>
                  </segmentResults>
                </legend>
              </bookings>
            </itineraryBookings>
          </bookingSummary>
        </bookingProcessingSummaryResponse>
      </confirmResponse>
    </ns2:confirmBookingResponse>
  </soap:Body>
</soap:Envelope>
```

5.1.5. Post booking operations

Return type	Name	Parameters	Description
BookingSummaryResponse	**getBookingById**	Credentials credentials, long bookingId	Get booking by Odigeo identifier
BookingSummaryResponse	**getBookingByPartnerReference**	Credentials credentials, String partnerReferenceId	Get booking by partner reference identifier

- BookingSummaryResponse **getBookingById** (Credentials credentials, long bookingId). Retrieve booking information by identifier. Intended to be called when the booking is completed or the partner receives a notification of status changes, with no need for a pre-established session, in order to get a summarized description of the booking.
  - Receives the booking identifier, and a Credentials object.
  - Returns a BookingSummaryResponse object that represents the booking information.

- BookingSummaryResponse **getBookingByPartnerReference** (Credentials credentials, String partnerReferenceId). Retrieve booking information by partner reference. Intended to be called any time after the booking is completed, with no need for a pre-established session, in order to get a summarized description of the booking.
  - Receives the booking partner reference identifier, and a Credentials object (partnerId and password).
  - Returns a BookingSummaryResponse object that represents the booking information. If the partner uses unique identifiers for different bookings, it will contain at most one booking; otherwise there could be more than one.

5.1.5.1. Data objects in post-booking operations.

- **BookingSummaryResponse**. Container for booking information, containing a list of BookingSummary objects named bookingSummaries. Normally, only one item will be available, but requests of booking information using *getBookingByPartnerReference* could produce multiple results if the partner did not use unique identifiers all the time.
- **BookingSummary**. Represents a booking, with the following structure:
  - Long id. Booking Odigeo identifier.
  - String partnerBookingReference. Partner assigned reference.
  - BookingStatus. Booking status
    - CONTRACT status means the booking is completed and permanent.
    - REQUEST is a temporary status and means that there are some actions pending to be executed to the booking before reach a final status.
    - REJECTED status means that the booking could not be completed and was cancelled.
    - UNKNOWN status means that we are unable to determine whether a booking has been successfully done. For example, in some cases there might be a timeout on the provider but the booking can eventually be successful or fail.
  - Buyer buyer. Buyer information, as described above.
  - PassengerTypeFare adultFare. Fare related to ADULT passengers.
  - PassengerTypeFare childFare. Fare related to CHILD passengers.
  - PassengerTypeFare infantFare. Fare related to INFANT passengers.
  - BigDecimal totalPrice. Total price for all the items in the cart.
  - BigDecimal baggageFees. Cost related to baggages selected. If no baggage is selected this attribute is not returned
  - List<Traveller> travellers, following the same structure as defined above in the context of ShoppingCart information.
  - ItineraryBookings itineraryBookings. Contains a collection of itinerary bookings, each one describing an itinerary:
    - Set<ItineraryBooking> bookings
- **ItineraryBooking**. Describes an itinerary, including some provider information (such as PNR):
  - int numAdults. Number of adult passengers in the itinerary.
  - int numChildren. Number of children passengers in the itinerary.
  - int numInfants. Number of infant passengers in the itinerary.
  - String pnr. Reservation code that may be sent to the buyer and be used to identify this trip.
  - BookingStatus bookingStatus.

- List<Integer> segments. Segment id's that may be resolved using the legend to represent the itinerary.
- ItinerariesLegend legend. Legend that resolves carriers, segments, sections, locations and technical stops in order to determine the itinerary.

5.1.5.2. Request example

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="http://b2b.odigeo.com/b2b/ws/v1">
  <soapenv:Header/>
  <soapenv:Body>
    <v1:getBookingById>
      <credentials partnerId="partner" password="partner2014"/>
<bookingId>${addPersonalInfoToShoppingCart#Response#//passengerDataResponse/shoppingCart/
@bookingId}</bookingId>
    </v1:getBookingById>
  </soapenv:Body>
</soapenv:Envelope>
```

5.1.5.3. Response example

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"/>
  <soap:Body>
    <ns2:getBookingByIdResponse xmlns:ns2="http://b2b.odigeo.com/b2b/ws/v1">
      <bookingSummary>
        <preferences currency="EUR" locale="en"/>
        <bookingSummaries bookingId="428505855" bookingStatus="REQUEST"
partnerReferenceId="marketing-test-2" totalPrice="216.47">
          <adultFare fare="48.17" tax="62.68"/>
          <childFare fare="20.55" tax="62.68"/>
          <infantFare fare="2.71" tax="19.68"/>
          <buyer address="test" buyerIdentificationType="PASSPORT" cityName="test"
countryCode="ES" dateOfBirth="9346-05-04" identification="1111111111" lastNames="TEST"
mail="buyer@buyer.com" name="NAMEBUYER" stateName="test" zipCode="00050">
            <phoneNumber countryCode="ES" number="655252525"/>
            <alternativePhoneNumber countryCode="ES" number="655252525"/>
          </buyer>
          <travellerResponses countryCodeOfResidence="ES" dateOfBirth="1987-05-05"
firstLastName="TEST TEST" identification="1111111111" identificationExpirationDate="2015-10-10"
identificationIssueCountryCode="ES" identificationType="PASSPORT" mealName="STANDARD"
middleName="test" name="TESTA" nationalityCountryCode="ES" titleName="MR"
travellerType="ADULT">
            <baggageSelection segmentTypeIndex="FIRST">
              <baggageDescriptor kilos="0" pieces="1"/>
            </baggageSelection>
          </travellerResponses>
          <travellerResponses countryCodeOfResidence="ES" dateOfBirth="2013-05-05"
firstLastName="TEST TEST" identification="1111111111" identificationExpirationDate="2015-10-10"
identificationIssueCountryCode="ES" identificationType="PASSPORT" mealName="STANDARD"
middleName="TEST" name="TESTB" nationalityCountryCode="ES" titleName="MR"
travellerType="INFANT"/>
          <travellerResponses countryCodeOfResidence="ES" dateOfBirth="2005-05-05"
firstLastName="TEST TEST" identification="1111111111" identificationExpirationDate="2015-10-10"
identificationIssueCountryCode="ES" identificationType="PASSPORT" mealName="STANDARD"
middleName="TEST" name="TESTC" nationalityCountryCode="ES" titleName="MR"
```

```
travellerType="CHILD"/>
          <itineraryBookings>
            <bookings bookingStatus="REQUEST" numAdults="1" numChildren="1" numInfants="1"
pnr="6QW84Q" segments="0">
              <legend>
                <carriers code="VY" id="0" name="Vueling"/>
                <locations cityIataCode="PAR" cityName="Paris" countryCode="FR"
countryName="France" iataCode="ORY" id="1391" locationType="AIRPORT" name="Orly"/>
                <locations cityIataCode="BCN" cityName="Barcelona" countryCode="ES"
countryName="Spain" iataCode="BCN" id="157" locationType="AIRPORT" name="El Prat"/>
                <sectionResults id="0">
                  <section aircraft="320" arrivalDate="2014-06-24T10:15:00" arrivalTerminal="W"
baggageAllowanceQuantity="1" baggageAllowanceType="NP"
cabinClass="ECONOMIC_DISCOUNTED" carrier="0" departureDate="2014-06-24T08:35:00"
departureTerminal="1" duration="100" flightType="REG" from="157" id="8024" to="1391"/>
                </sectionResults>
                <segmentResults id="0">
                  <segment carrier="0" duration="100" sections="0"/>
                </segmentResults>
              </legend>
            </bookings>
          </itineraryBookings>
        </bookingSummaries>
      </bookingSummary>
    </ns2:getBookingByIdResponse>
  </soap:Body>
</soap:Envelope>
```