UOF - Endpoints

Table of contents:

- API endpoints
- Details regarding endpoints in the API
- Match bookings
- Special note on the tournament(s).xml endpoint
- Response codes
- Endpoint update frequency
    - Cashing remarks for implementation

API endpoints

The following table lists all available endpoints found in the self-service documentation here: https://iodocs.betradar.com/unifiedfeed

HTTP	Endpoint-path from **https://iodocs.betradar.com/**	Description
GET	`descriptions/(lang)/markets.xml`	Describes all currently available markets.
GET	`descriptions/(lang)/match_status.xml`	Describes all sports specific match status codes used during live matches in the odds_changes message. Translated to available languages.
GET	`descriptions/betstop_reasons.xml`	Describes all bet stop reasons.
GET	`descriptions/betting_status.xml`	Describes all betting statuses (in odds_changes.odds)
GET	`descriptions/producers.xml`	Describes all currently available producers and their ids.
GET	`descriptions/void_reasons.xml`	Describes all possible void reasons (in bet_settlement.market)
POST	`liveodds/booking-calendar/events/(id)/book`	POST on this endpoint to book a match in the booking-calendar
GET	`probabilities/(id)`	Retrieve the probabilities for all available markets for a sport-event. Typically used for cashout purposes.
GET	`probabilities/(id)/(market-id)`	Retrieve all the probabilities for all the market lines for the specified sport-event and market-id. Typically used for cashout purposes.
GET	`probabilities/(id)/(market-id)/(specifiers)`	Retrieve the probabilities for the specified market line.
POST	`(product)/stateful_messages/events/(id)/initiate_request`	Used request all stateful messages (bet_settlement, bet_cancel, etc.) sent for a particular sport event.
POST	`(product)/recovery/initate_request.xml?after=(timestamp)[&request_id=(x)][&node_id=(y)]`	Sends all current odds and historical stateful messages (bet_settlement, rollback_betsettlement, bet_cancel, rollback_betcancell) from the specified timestamp and onwards. An error is returned if the specified timestamp is too far in the past.

POST	`(product)/odds/events/(id)/initiate_request`	All current odds for all the markets for one single match/race. The event must be either not_started or live.
GET	`sports/(lang)/competitors/(id)/profile.xml`	Name and details about a competitor (team, race driver, tennis player etc).
GET	`sports/(lang)/fixtures/changes.xml`	Lists the ids of sport-events that have fixture changes the last 24 hours.
GET	`sports/(lang)/sport_events/(id)/fixture.xml`	Lists the fixture for a specified event (event = match, game, race, outright).
GET	`sports/(lang)/sport_events/(id)/timeline.xml`	Information about a sport event that is ongoing or completed. The actual endpoint is event-specific and the actual attributes varies slightly between sports.
GET	`sports/(lang)/sport_events/(id)/summary.xml`	Returns the results for a sport event. The <id> must use an urn-scheme. sr:match:<id> for match-ids. sr:stage:<id> for race results. sr:tournament:<id> for tournament results, etc.
GET	`sports/(lang)/players/(id)/profile.xml`	Name and details about a player. A player here is a player in a team. (A tennis player is a competitor)
GET	`sports/(lang)/schedules/(date)/schedule.xml`	Lists all sport events scheduled to start at a specified date (UTC). <date> is an ISODate (e.g. 2016-02-10).
GET	`sports/(lang)/schedules/live/schedule.xml`	Lists all currently live sport events (id, what teams, start time, etc.).
GET	`sports/(lang)/schedules/pre/schedule.xml?start=(x)&limit=(y)`	Lists almost all events we are offering prematch odds for. This endpoint can be used during early start-up to obtain almost all fixtures. This endpoint is one of the few that uses pagination.
GET	`sports/(lang)/sports.xml`	A list of all available sports
GET	`sports/(lang)/sports/(id)/categories.xml`	This lists available categories for a sport. Category is generic classification term BetRadar uses to at the highest level subclassify a particular sport (e.g. for Tennis the categories can be ATP-Tour, WTA-Tour, David Cup etc., for soccer the categories are the various countries)
GET	`sports/(lang)/sports/(id)/tournaments.xml`	A list of all available tournaments for the specified sport that have had matches in the past 7 days, or that have any matches scheduled in the future.
GET	`sports/(lang)/tournaments.xml`	A list of all available tournaments
GET	`sports/(lang)/tournaments/(id)/info.xml`	Provides basic information about a tournament (such as the dates of the tournament, type of tournament and competitors in the tournament)
GET	`sports/(lang)/tournaments/(id)/schedule.xml`	Provides all scheduled matches in the specified tournament.
GET	`sports/(lang)/venues/(id)/profile.xml`	Details about a venue.
GET	`users/whoami.xml`	Lists the callers bookmaker id, AMQP access point and when the provided access token will expire.

GET	`/descriptions/(language)/markets/(market_id)/variants/variant_urn)`	Get market descriptions and outcomes for markets from odds /settlements, containing "variant=<urn>".
GET	`/(product)/descriptions/(language)/markets/(market_id/variants/(variant_urn)`	Get market descriptions and outcomes for markets from odds /settlement, containing "variant=<urn>".

Note

There are 2 main endpoints to be used in association with *variants*;

a) /descriptions/(language)/markets/(market_id)/variants/variant_urn)

b) /(product)/descriptions/(language)/markets/(market_id/variants/(variant_urn)

The last endpoint (b) is used if the variant doesn't start with "sr:". However, the first endpoint redirects to the second endpoint, so if you set your code to "followredirect" you can always just use the first endpoint.

Details regarding endpoints in the API

- (lang) Is the requested language (note that the XML elements and attributes themselves are not translated, only some of the attribute values; e.g. team names, player names etc.).
- (producer) Can be live odds (Live Odds) or pre (normal prematch odds), Betpal, premium_cricket, vfl (virtual football), vto (virtual tennis), vbl (virtual basketball), wns (number betting). Additional producers will be added over time: (See the endpoint: https://api.betradar.com/v1/descriptions/producers.xml for a listing of currently available producers and their ids).

Start: Is the starting record (this is an index, not time).

Limit: Is how many records (sport_events) to return. Max for limit is 1000, so you can do; *start = 0&limit=1000*, then *start=1000&limit=1000* and so on, until you get no data.

The primary goal of this endpoint is to reduce the number of individual fixtures you have to read during a recovery.

Note

When using the *sports/(lang)/schedules/pre/schedule.xml?start=(x)&limit=(y)* endpoint in the table above, keep in mind that this is intended for a mostly stateless start-up of the client.

Match bookings

Note that in the daily schedule, all sport-events are listed with a “live-odds” attribute. This attribute lists the match-booking state and can be in the following states: ***booked*** (already booked), ***bookable*** (can be booked), ***buyable*** (need to be bought before it can be booked, contact your sales-representative), and ***unavailable*** (currently not available at all for live booking).

This is only for live odds where a customer can request booking of a particular match.

HTTP	Endpoint	Description
POST	`liveodds/booking-calendar/events/(id)/book`	POST on this Endpoint to put a match in the booking calendar.

XML example

```
<response response_code="OK">
    <action>Request for booking an event : sr:match:12345678 from
bookmaker: 1234 received</action>
    <message>OK.</message>
</response>
```

Special note on the tournament(s).xml endpoint

This endpoint lists all available tournaments where Sportradar provides coverage.

You can still receive information about a match that is associated with a tournament not on the returned list from the endpoints. This could for example happen with lower division soccer league matches where Sportradar does not provide full league coverage, but may do so only upon request from one or more bookmakers. Such tournaments are typically not listed, and Sportradar does not provide a tournament schedule either for these tournaments.

Therefor it is important to not assume that these endpoints will return every single tournament of interest, but rather all available tournaments where Sportradar provide coverage, and/or customers buy coverage for a particular match.

Response codes

The general http status codes are defined here https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

Here is how Status Codes are used in this API:

Code	Name	Description
200	Ok	When everything is ok and we return data directly in the http body
201	Created	For booking calendar when client successfully books a match
202	Accepted	The system accepted your request for odds recovery/statefull messages and will soon/immediately send it out on the feed
403	Forbidden	Token is missing or invalid.
404	Not found	Resource (event id or sport id etc) was not found, the system will not fulfill your request.
409	Conflict	Bookmaker has another odds recovery already in progress
503	Service unavailable	Returned when the underlying odds producers are temporarily down. Retry again soon.

Endpoint update frequency

Note

This cashing is done on the server side and does not need to be performed on the client side.

To avoid unnecessary re-computation of endpoints, the following endpoints are cached and will not be recomputed until the cache time expires:

- **Static**: 1 day - endpoint almost never changes. Example: List of sports
- **Long**: 1 hour - endpoint very rarely changes. Example: Schedules.
- **Medium**: 10 minutes - Endpoint affected by played matches, but not time critical: stats
- **Short**: 1 minutes - Endpoint affected by played matches, that needs quick update
- **Live**: 1 seconds - Endpoint needs real-time updates.

Endpoint	Type	Cache TTL
sports.xml	Static	1 day
venues/(id)/summary.xml	Static	1 day
sports/(id)/tournaments.xml	Long	1 hour
sports/(id)/categories.xml	Long	1 hour
players/(id)/profile.xml	Long	1 hour
schedules/(date)/schedule.xml	Long	1 hour
tournaments.xml	Long	1 hour
tournaments/(id)/info.xml	Long	1 hour
tournaments/(id)/schedule.xml	Long	1 hour
competitors/(id)/profile.xml	Medium	10 min
schedules/live/schedule.xml	Live	1 sec
sport_events/(id)/fixture.xml	Live	1 sec
sport_events/(id)/summary.xml	Live	1 sec
sport_events/(id)/timeline.xml	Live	1 sec

Cashing remarks for implementation

On the client side, it is possible to cache fixtures. If you do this, you have to invalidate a particular fixture when you receive a fixture_change message, for that fixture. 24 hours is a reasonable caching time.

Variant descriptions can be cached for 24 hours or more. The only reason a particular variant description would change is in case of a data entry error.

Back to top