Introduction

As developers and sports fans ourselves, we know how difficult it is to find a quality, affordable source for sports data.

**So we created one!**

We're committed to making **MySportsFeeds** the best, most accessible sports data API for everyone. If you've been searching for an easy-to-use, flexible, and affordable (or FREE) data source for your app, site, blog or research project, you've finally found it!

PERSONAL usage

If you believe your usage is personal, you can [contact us](https://www.mysportsfeeds.com/contact-us) or [message us on #Slack](https://mysportsfeeds-slackinvite.herokuapp.com/) with details. If approved, we'll adjust your account and you can create a PERSONAL API Key.

We still offer a base amount of access to CORE data feeds (schedules and scores) for the 2 most recent seasons. For anything beyond that, you'll need to purchase addons, which are extremely affordable (a few dollars per addon).

If you have any questions at all, you can send them our way at [help@mysportsfeeds.com](mailto:help@mysportsfeeds.com).

Join Us on #Slack

If you're a fan of **#Slack** like we are, you can also [join our Slack team](https://mysportsfeeds-slackinvite.herokuapp.com/) for more updates and announcements. Either way, we take great pride in our fast response time, so go ahead and time us!

API Keys

Access to the API is done via API Keys, which you can manage from your account page. This applies to all versions of the API, and you can access all versions with the same key.

Subscriptions

Once created, you can add or remove subscriptions for the available leagues. For each subscription, you can choose a plan type (**Non-Live**, **Live**, or **Near-Realtime**), and any addons you may need. The billing interface will walk you through selection of each.

Authentication

All versions of the API make use of the HTTP Basic authentication mechanism. It means you'll need to include a special "Authorization" HTTP header along with your HTTP request. The value for that header will relate to your API Key. Assuming you've created at least one API Key from your account page, here's what the required Authorization header will look like.

Authorization: Basic {encrypted\_api\_key\_credentials}

where:

| **Parameter** | **Description** |
| --- | --- |
| {encrypted\_api\_key\_credentials} | a value obtained by concatenating  api\_key\_token + ":" + password  and then encoding it with base64 encoding. You can [read more here](https://en.wikipedia.org/wiki/Basic_access_authentication).  An example for this in each language is included within the examples for each feed. Select one from the list to the left.    \*\*IMPORTANT\*\*  The password you use in your request depends on the version you're requesting.  For backwards-compatibility reasons, all v1.x requests will need to use your MySportsFeeds account password.  For v2.x requests you'll simply use the literal string "MYSPORTSFEEDS". Once v1.x is eventually phased out, all requests will use "MYSPORTSFEEDS" as the password. |

Why Basic Auth?

There is some debate over whether HTTP Basic should be used to secure APIs at all. We believe it's not only appropriate for our API, but also highly convenient. The reasons:

* it prevents accidentally exposing your API key in the URL. Instead, you can just copy/paste request URLs without worrying about including your credentials.
* it's an extremely simple authentication scheme, understood by all browsers. Meaning you don't need postman or other API tools to test your requests, if you don't want to.
* all requests are secured over SSL.
* all requests simply issue the HTTP GET method, there's no risk in causing any harm to the system, intentional or not.
* if an API key is compromised, it's extremely easy to delete the key and replace it with another.

Projections

Version 2.x of our API now features several projections feeds, available via the **PROJECTIONS** addon. In general, they return projections for:

* seasonal player stat totals
* daily/weekly player gamelogs
* daily/weekly DFS fantasy points (based on game-by-game projections)

Design Philosophy

It's important to note that the above projections are for players only, not teams or game outcomes (W, L, scores, etc).

Also important, is that the gamelogs projections do not consider the likelihood of whether a player will play in a given game or not. Instead, the projection only considers what their stats would be **\*IF\*** they played in that game.

Our very custom algorithm first calculates the projected performance for each player over the duration of the season, including the expected number of games played.

Then, it further projects their performance in every game for the season. During the season at regular intervals, the seasonal totals and game-by-game projections will be adjusted and recalculated. Game-by-game projections for played games will not be adjusted, since they are useful as a way for us to visualize our own accuracy. In this way, the projections will increase in accuracy as a season progresses and games are played.

Pricing

Since we want to provide everyone access our sports data, we've created two different sets of pricing tiers, depending on whether you represent a **COMMERCIAL** business or a **PERSONAL** user.

Personal Usage

To be considered for personal usage, your usage must be both private and personal. This can includes hobbyists, students, bloggers and researchers. For new subscribers, you'll automatically be assigned as a COMMERCIAL user unless you [Contact Us](https://www.mysportsfeeds.com/contact-us) to request PERSONAL access. When you apply, let us know how you consider it to be personal, and how our API will be used.

If we agree to assign you PERSONAL account status, by default you'll have free access to CORE data feeds only (schedule + scores).

Commercial Usage

This includes any individuals or businesses who charge for their service, including advertising. Even still, [our commercial pricing](https://www.mysportsfeeds.com/feed-pricing/) represents the absolute BEST value out there, as you've no doubt discovered yourself!

Available Formats

All feeds come in both **JSON** and **XML** formats. And some even have a **CSV** format that you can import directly into an Excel spreadsheet.

In the case of **XML**, you can also download the **XSD** definition for each feed. In fact, the XSD can also be used as a guideline for the content of the JSON feeds as well, since they're both based on the same overall content definitions.

Regardless which format you need, it's incredibly easy to specify in your requests. Meaning it's equally easy to change your mind.

PUSH Notifications

By far, making (or "PULLing") HTTP requests over our lightning-fast RESTful API is the standard way to retrieve data.

***Important:*** Our previous PUSH functionality was limited and is in need of a redesign. So for now, PULL-based requests are the only option.

Compression

The **MySportsFeeds** API serves up millions of requests per month. As you can imagine, that results in a LOT of bandwidth! So we'd greatly appreciate if every subscriber made an effort to ensure that they are enabling compression from their client code.  
  
Doing so saves roughly 90% of the bandwidth!  
  
All you really need to do is specify an additional header as follows.

Accept-Encoding: gzip

All of the code examples contained in these API docs will specify compression where needed. In most cases and libraries, this will be the default behaviour. However, particularly with cURL, that is NOT the default. Instead, you should supply the "--compress" option.

curl -X "GET" "https://www.mysportsfeeds.com/api/feed/pull/nfl/2016-2017-regular/scoreboard.json?fordate=20161211"

-u {username}:{password}

--compress

Conventions

The **MySportsFeeds** API has been designed with consistency in mind. So it won't matter whether you're making boxscore requests for live MLB games, looking at cumulative player stats for the NFL, pouring over active players for the NBA, or examining play-by-plays for the NHL. The feeds, URL and parameters, and content all follow a similar and predictable pattern.

What's a "Feed"?

Some would prefer to call them endpoints, and they'd be partly correct. Each feed is available via an endpoint, but they're actually more than that. Each feed represents a set of syndicated content (available in several formats), along with supported params and the ability to be PULLed or PUSHed. However, the terms endpoint and feed in our case can be used interchangably.

Team, Player, Venue and Game IDs

All IDs listed are globaly unique and consistent, and will not change from feed to feed.

Dates and Times

For v1.x feeds, dates use a format of YYYYMMDD, which assume an Eastern time zone (EST).

For v2.x feeds, dates use an ISO-8601 compatible syntax, so you can reliably use them from your favourite code libraries.

Stats Abbreviations

For feeds that allow optional team/player stats via teamstats or playerstats, you can also specify that **NONE** be included via specifying teamstats=none or playerstats=none.  
Otherwise, a list of comma-delimited, case-insensitive abbreviations can be specified. Note that this will include stats across all stat categories.

Backoff Delay

Some feeds involve heavier processing, which give them more "weight" and impose more demand on CPU and RAM resources. Such feeds will include a required backoff delay, measured in seconds. You will need to pace your requests for the same feed (and league) to satisfy this condition.  
  
**NOTE**:  
Game-specific feeds \*DO NOT\* require a backoff delay, since you will often need to access them individually for each game.

At the moment, only v2.x feeds are affected.

v2.x feeds including a **30 second** backoff include:

* Seasonal DFS
* Seasonal Player Gamelogs
* Seasonal Player Stats Projections
* Seasonal Game Lines
* Daily Futures

v2.x feeds including a **15 second** backoff include:

* Daily/Weekly Game Lines

v2.x feeds including a **5 second** backoff include:

* Daily/Weekly Player Gamelogs
* Daily/Weekly Team Gamelogs
* Daily/Weekly DFS
* Daily/Weekly Player Gamelogs Projections
* Daily/Weekly DFS Projections
* Player Injuries
* Injury History
* Seasonal Team Stats
* Seasonal Player Stats
* Players
* Seasonal Standings

Request Limit within a 1-minute interval

All requests will count towards a limit every minute. By default, a count of "1" will be used. If a backoff is also required for that same feed, the backoff seconds will also be added.  
  
So if a backoff of 5 seconds is required, the count will be increased by 5, meaning 5 + 1 = "6".  
  
At the end of each minute, if the number is above the defined allowable amount of:  
  
**100**  
  
Then a 429 response will be received.

After each minute, the counter will be reset and start over again.

Response Codes

Whenever you receive an empty response, or are otherwise experiencing unexpected behaviour, you should check the request's HTTP response code and reference the table below. The following are all possible HTTP response codes for any request to the API.

| **Code** | **Description** |
| --- | --- |
| **200** | Request was successful. |
| **204** | Feed content not found. There is no available content for the feed as of yet.  This is often the case when you only have a non-live subscription, and you're trying to access a game's feed while it's still live.  It's also expected when you request a boxscore or the play-by-play for a future game which hasn't started yet. |
| **304** | This response code coincides with the optional URL parameter:  &force=false  If you specify the above force parameter as "false", the API will check to see if the feed's content has changed since your last request. If the content hasn't changed, the API will return an empty response, and the response code will be 304. This is done as a way to save bandwidth, for those who are only expecting changes since the last request.  The default behaviour without force specified, is "true", which will return content regardless. It's then up to you to determine whether or not the content has actually changed.  Overall, this code indicates that feed content has not changed since your last request. For live feeds, this could also indicate that you haven't waited long enough since your last request based on your subscription's update frequency (1/3/5/10 minutes, or near-realtime). |
| **400** | Indicates that some part of the request is malformed. It can be caused by any one of the following:   * unrecognized format (other than csv, json, or xml) * unrecognized league (other than nfl, mlb, nba, or nhl) * unrecognized season (should follow convention of {start-year}-{end-year}-{season-type}) * unrecognized feed name, double-check spelling for typos * for game-specific feeds, specified game not found, double-check gameid URL parameter (should follow convention of {YYYYMMDD}-{away-team}-{home-team}) * for date-specific feeds, badly formatted date for fordate URL parameter, should follow convention of (YYYYMMDD) * for game-specific feeds, badly formatted game identifier (should follow convention of {YYYYMMDD}-{away-team}-{home-team}) * unrecognized stat abbreviations for the teamstats or playerstats URL parameters, double check typos. |
| **401** | Not authenticated. You need to authenticate each request using your HTTP BASIC-encoded MySportsFeeds username/password. See the **Authentication** section. |
| **403** | Not authorized. Indicates you're attempting to access a feed for the current in-progress season, without a proper subscription. For in-progress or upcoming seasons, you'll need to be subscribed. |
| **429** | Too Many Requests. In order to make the API as responsive as possible, we have to limit excessive overuse. See the Throttling section for the actual limits. |
| **500** | An internal error has occurred, and has been logged for further inspection. Feel free to let us know if you're receiving this response unexpectedly. |
| **499,502,503** | These errors can occur periodically, and usually means we're in the process of publishing system updates. It should resolve itself in a matter of a few seconds. If you're seeing this consistently, please contact us. |

Available Stats

**MySportsFeeds** is designed to support the periodic addition of new team and player stats as we mature and expand. While this documentation makes an effort to include all of the up-to-date stats in the sample responses, they may not be exhaustive. Therefore you should always defer to the live responses for a complete list of all supported stats.

In addition, the number of stats supported may vary from season to season, as we retroactively make changes to past seasons. In some cases, some stats may not even be relevant to past seasons due to each league's rule changes. Once again, you should always defer to the live responses for a complete list of all supported stats for a given season.

Seasonal Games (v2.1)

All games for a season including schedule, status and scores.

\*Addon Required\*: None (CORE)

URL Examples:

**CSV**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/games.csv

**JSON**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/games.json

**XML**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/games.xml

**XSD (for XML)**: https://api.mysportsfeeds.com/v2.1/xsd/nhl/seasonal\_games

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/**games**.{format}

with optional parameters:

**team={list-of-teams}** *(filter teams)*

**date={date-range}** *(filter on dates)*

**status={list-of-game-statuses}** *(filter game statuses)*

**sort={sort-specifier}** *(sort the feed's content)*

**offset={offset-specifier}** *(filter results starting at the given offset)*

**limit={limit-specifier}** *(limit the maximum # of results)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {format} | one of **csv**, **xml**, or **json** |
| date={date-or-range-of-dates} | (*optional*) a single date or a range of dates.  If this parameter is omitted, by default ALL dates are included.  There are several acceptable formats for a date. Each is case-insensitive.  Yesterday:  **yesterday**  Today:  **today**  Tomorrow:  **tomorrow**  Simple YYYYMMDD format:  **20170427**  Since a given date:  **since-yesterday** **since-20170426** **since-3-days-ago** **since-3-weeks-ago** **since-3-months-ago**  Until a given date:  **until-tomorrow** **until-20170501** **until-3-days-from-now** **until-3-weeks-from-now** **until-3-months-from-now**  Between two specified dates:  **from-yesterday-to-today** **from-20170426-to-tomorrow** **from-3-days-ago-to-3-days-from-now** |
| team={list-of-teams} | (*optional*) comma-delimited list of one or more teams.  If this parameter is omitted, by default ALL teams are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a team. Each is case-insensitive, and strips out any punctuation.  Team ID:  **24**  Team abbreviation:  **bos**  City and team name:  **boston-celtics** |
| status={list-of-game-statuses} | (*optional*) comma-delimited list of one or more game statuses.  If this parameter is omitted, by default ALL game statuses are included.  Values are case-insensitive, supported values are:  **unplayed** *(scheduled but not yet started)* **in-progress** *(currently underway)* **postgame-reviewing** *(game is over, but we're reviewing against official sources)* **final** *(game is final and reviewed)* |
| sort={sort-specifier} | (*optional*) Specifies a sorting order for the results. Format is:  {prefix}.{value}.{sort-order}  The {sort-order} is optional, and must be either "**A**" for Ascending, or "**D**" for descending.  Valid values for this feed are:  **game.starttime** |
| offset={offset-specifier} | (*optional*) Specifies the starting offset for the results. Defaults to 0. |
| limit={limit-specifier} | (*optional*) Specifies the maxmimum number of results. If not specified, all results are returned. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

The following describes each property of the feed's returned response. Static samples are no longer available without making a valid request to the API.

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ISO-8601 format in UTC timezone. |
| game.schedule | Contains the scheduled start date/time, along with references to each team and the venue. |
| game.schedule.id | Unique identifier for the game. |
| game.schedule.startTime | An ISO-8601 timestamp in UTC. |
| game.schedule.awayTeam, game.schedule.homeTeam, game.schedule.venue | Short-handed references for the teams and venue. You'll find more details for each in the references section at the bottom of the feed. |
| game.schedule.venueAllegiance | Which team has home-team advantage? Possible values are **AWAY**, **HOME**, and **NEUTRAL**. |
| game.schedule.scheduleStatus | Set to **NORMAL** if is on schedule as expected. Otherwise, set to **DELAYED** or **POSTPONED** accordingly. |
| game.schedule.originalStartTime | Will be **null** unless scheduleStatus is **POSTPONED**. In that case, this will represent the originally scheduled date/time, and startTime above will represent the new start date/time. |
| game.schedule.delayedOrPostponedReason | May be set to indicate the reason for a delay or postponement. Otherwise, set to **null**. |
| game.schedule.playedStatus | Possible values are:  **UNPLAYED** - not yet started, scheduled only.  **LIVE** - live and in progress.  **COMPLETED** - game is over and reviewed (final).  **COMPLETED\_PENDING\_REVIEW** - game is over, however we're reviewing all plays and stats, which takes roughly an hour. Afterwards it's set to COMPLETED. |
| game.score.currentPeriod | If game is in progress, indicates the period. Otherwise, set to **null**. |
| game.score.currentPeriodSecondsRemaining | If game is in progress, indicates the seconds remaining in the current period. Otherwise, set to **null**. |
| game.score.currentIntermission | If game is in progress and between periods, indicates the intermission. Otherwise, set to **null**. |
| game.score.awayScoreTotal, game.score.homeScoreTotal | If game is unplayed, set to **null**. Otherwise, indicates the toal score for both teams. |
| game.score.periods | An array of scoring for each period. |

These samples demonstrate how to make a PULL request for this feed in various languages. Includes authentication via **HTTP BASIC** with your API Key credentials, and specifying compression where necessary.

You'll just need to replace the parameters for {pull-url} and {apikey\_token} with appropriate values.

Or if you want an even more concrete example, [check out our wrapper libraries on Github](https://github.com/MySportsFeeds). If your favourite language isn't available as a wrapper library yet, it may soon be - let us know!

* [cURL](javascript:void(0);)
* [PHP](javascript:void(0);)
* [Python](javascript:void(0);)
* [Javascript](javascript:void(0);)
* [Node.js](javascript:void(0);)
* [Java](javascript:void(0);)
* [C#](javascript:void(0);)
* [Obj-C](javascript:void(0);)
* [Ruby](javascript:void(0);)

$.ajax

({

type: "GET",

url: {pull-url},

dataType: 'json',

async: false,

headers: {

"Authorization": "Basic " + btoa({apikey\_token} + ":" + MYSPORTSFEEDS)

},

data: '{ "comment" }',

success: function (){

alert('Thanks for your comment!');

}

});

OR NODE JS

*// request Request*

(function(callback) {{ '{' }}

'use strict';

const httpTransport = require('https');

const responseEncoding = 'utf8';

const httpOptions = {{ '{' }}

hostname: 'www.mysportsfeeds.com',

port: '443',

path: {{ '{pull-url}' }},

method: 'GET',

headers: {{ '{' }}"Authorization":"Basic " + btoa({{ '{apikey\_token}' }} + ":" + MYSPORTSFEEDS)}

{{ '}' }};

httpOptions.headers['User-Agent'] = 'node ' + process.version;

const request = httpTransport.request(httpOptions, (res) => {{ '{' }}

let responseBufs = [];

let responseStr = '';

res.on('data', (chunk) => {{ '{' }}

if (Buffer.isBuffer(chunk)) {{ '{' }}

responseBufs.push(chunk);

{{ '}' }}

else {{ '{' }}

responseStr = responseStr + chunk;

{{ '}' }}

{{ '}' }}).on('end', () => {{ '{' }}

responseStr = responseBufs.length > 0 ?

Buffer.concat(responseBufs).toString(responseEncoding) : responseStr;

callback(null, res.statusCode, res.headers, responseStr);

{{ '}' }});

{{ '}' }})

.setTimeout(0)

.on('error', (error) => {{ '{' }}

callback(error);

{{ '}' }});

request.write("")

request.end();

{{ '}' }})((error, statusCode, headers, body) => {{ '{' }}

console.log('ERROR:', error);

console.log('STATUS:', statusCode);

console.log('HEADERS:', JSON.stringify(headers));

console.log('BODY:', body);

{{ '}' }});

Daily Games (v2.1)

All games on a given date including schedule, status and scores.

\*Addon Required\*: None (CORE)

URL Examples:

**CSV**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/date/20251007/games.csv

**JSON**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/date/20251007/games.json

**XML**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/date/20251007/games.xml

**XSD (for XML)**: https://api.mysportsfeeds.com/v2.1/xsd/nhl/daily\_games

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/date/{date}/**games**.{format}

with optional parameters:

**team={list-of-teams}** *(filter teams)*

**status={list-of-game-statuses}** *(filter game statuses)*

**sort={sort-specifier}** *(sort the feed's content)*

**offset={offset-specifier}** *(filter results starting at the given offset)*

**limit={limit-specifier}** *(limit the maximum # of results)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {date} | a valid date in the form of **YYYYMMDD** |
| {format} | one of **csv**, **xml**, or **json** |
| team={list-of-teams} | (*optional*) comma-delimited list of one or more teams.  If this parameter is omitted, by default ALL teams are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a team. Each is case-insensitive, and strips out any punctuation.  Team ID:  **24**  Team abbreviation:  **bos**  City and team name:  **boston-celtics** |
| status={list-of-game-statuses} | (*optional*) comma-delimited list of one or more game statuses.  If this parameter is omitted, by default ALL game statuses are included.  Values are case-insensitive, supported values are:  **unplayed** *(scheduled but not yet started)* **in-progress** *(currently underway)* **postgame-reviewing** *(game is over, but we're reviewing against official sources)* **final** *(game is final and reviewed)* |
| sort={sort-specifier} | (*optional*) Specifies a sorting order for the results. Format is:  {prefix}.{value}.{sort-order}  The {sort-order} is optional, and must be either "**A**" for Ascending, or "**D**" for descending.  Valid values for this feed are:  **game.starttime** |
| offset={offset-specifier} | (*optional*) Specifies the starting offset for the results. Defaults to 0. |
| limit={limit-specifier} | (*optional*) Specifies the maxmimum number of results. If not specified, all results are returned. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

The following describes each property of the feed's returned response. Static samples are no longer available without making a valid request to the API.

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ISO-8601 format in UTC timezone. |
| game.schedule | Contains the scheduled start date/time, along with references to each team and the venue. |
| game.schedule.id | Unique identifier for the game. |
| game.schedule.startTime | An ISO-8601 timestamp in UTC. |
| game.schedule.awayTeam, game.schedule.homeTeam, game.schedule.venue | Short-handed references for the teams and venue. You'll find more details for each in the references section at the bottom of the feed. |
| game.schedule.venueAllegiance | Which team has home-team advantage? Possible values are **AWAY**, **HOME**, and **NEUTRAL**. |
| game.schedule.scheduleStatus | Set to **NORMAL** if is on schedule as expected. Otherwise, set to **DELAYED** or **POSTPONED** accordingly. |
| game.schedule.originalStartTime | Will be **null** unless scheduleStatus is **POSTPONED**. In that case, this will represent the originally scheduled date/time, and startTime above will represent the new start date/time. |
| game.schedule.delayedOrPostponedReason | May be set to indicate the reason for a delay or postponement. Otherwise, set to **null**. |
| game.schedule.playedStatus | Possible values are:  **UNPLAYED** - not yet started, scheduled only.  **LIVE** - live and in progress.  **COMPLETED** - game is over and reviewed (final).  **COMPLETED\_PENDING\_REVIEW** - game is over, however we're reviewing all plays and stats, which takes roughly an hour. Afterwards it's set to COMPLETED. |
| game.score.currentPeriod | If game is in progress, indicates the period. Otherwise, set to **null**. |
| game.score.currentPeriodSecondsRemaining | If game is in progress, indicates the seconds remaining in the current period. Otherwise, set to **null**. |
| game.score.currentIntermission | If game is in progress and between periods, indicates the intermission. Otherwise, set to **null**. |
| game.score.awayScoreTotal, game.score.homeScoreTotal | If game is unplayed, set to **null**. Otherwise, indicates the toal score for both teams. |
| game.score.periods | An array of scoring for each period. |

These samples demonstrate how to make a PULL request for this feed in various languages. Includes authentication via **HTTP BASIC** with your API Key credentials, and specifying compression where necessary.

You'll just need to replace the parameters for {pull-url} and {apikey\_token} with appropriate values.

Or if you want an even more concrete example, [check out our wrapper libraries on Github](https://github.com/MySportsFeeds). If your favourite language isn't available as a wrapper library yet, it may soon be - let us know!

* [cURL](javascript:void(0);)
* [PHP](javascript:void(0);)
* [Python](javascript:void(0);)
* [Javascript](javascript:void(0);)
* [Node.js](javascript:void(0);)
* [Java](javascript:void(0);)
* [C#](javascript:void(0);)
* [Obj-C](javascript:void(0);)
* [Ruby](javascript:void(0);)

*// request Request*

(function(callback) {{ '{' }}

'use strict';

const httpTransport = require('https');

const responseEncoding = 'utf8';

const httpOptions = {{ '{' }}

hostname: 'www.mysportsfeeds.com',

port: '443',

path: {{ '{pull-url}' }},

method: 'GET',

headers: {{ '{' }}"Authorization":"Basic " + btoa({{ '{apikey\_token}' }} + ":" + MYSPORTSFEEDS)}

{{ '}' }};

httpOptions.headers['User-Agent'] = 'node ' + process.version;

const request = httpTransport.request(httpOptions, (res) => {{ '{' }}

let responseBufs = [];

let responseStr = '';

res.on('data', (chunk) => {{ '{' }}

if (Buffer.isBuffer(chunk)) {{ '{' }}

responseBufs.push(chunk);

{{ '}' }}

else {{ '{' }}

responseStr = responseStr + chunk;

{{ '}' }}

{{ '}' }}).on('end', () => {{ '{' }}

responseStr = responseBufs.length > 0 ?

Buffer.concat(responseBufs).toString(responseEncoding) : responseStr;

callback(null, res.statusCode, res.headers, responseStr);

{{ '}' }});

{{ '}' }})

.setTimeout(0)

.on('error', (error) => {{ '{' }}

callback(error);

{{ '}' }});

request.write("")

request.end();

{{ '}' }})((error, statusCode, headers, body) => {{ '{' }}

console.log('ERROR:', error);

console.log('STATUS:', statusCode);

console.log('HEADERS:', JSON.stringify(headers));

console.log('BODY:', body);

{{ '}' }});

OR JAVASCRIPT

$.ajax

({

type: "GET",

url: {pull-url},

dataType: 'json',

async: false,

headers: {

"Authorization": "Basic " + btoa({apikey\_token} + ":" + MYSPORTSFEEDS)

},

data: '{ "comment" }',

success: function (){

alert('Thanks for your comment!');

}

});

[FAQ](https://www.mysportsfeeds.com/faq/)

[Member Rewards](https://www.mysportsfeeds.com/member-rewards/)

[Terms of Service](https://www.mysportsfeeds.com/terms-use/)

[Privacy Policy](https://www.mysportsfeeds.com/privacy-policy/)

[About Us](https://www.mysportsfeeds.com/about-us/)

[Affiliate Logos](https://www.mysportsfeeds.com/logos/)



© 2012-2025 Be A Contender Software Services Inc. | All Rights Reserved

MySportsFeeds is an unofficial and independent source of information not affiliated with the National Hockey League (NHL), National Football League (NFL), Major League Baseball (MLB), National Basketball Association (NBA), or any of their sub-affiliates.

This site uses cookies. Some of the cookies we use are essential for parts of the site to operate and have already bee

Current Season (v2.1)

Returns the current season and supported stats, for a given date.

\*Addon Required\*: None (CORE)

URL Examples:

**JSON**: https://api.mysportsfeeds.com/v2.1/pull/nhl/current\_season.json

**XML**: https://api.mysportsfeeds.com/v2.1/pull/nhl/current\_season.xml

**XSD (for XML)**: https://api.mysportsfeeds.com/v2.1/xsd/nhl/current\_season

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/**current\_season**.{format}

with optional parameters:

**date={date}** *(specify a date)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {format} | one of **csv**, **xml**, or **json** |
| date={date} | (*optional*) a single date as YYYYMMDD.  If this parameter is omitted, by default the current date is assumed. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ET timezone. In some cases this will be set to null when the feed has been craeted for the first time. This is not intended behaviour and will be corrected when time allows. |
| season | An array of applicable seasons for the given date (or today if no date provided). In almost all cases, only a single season will be returned. However there is potential for seasons to overlap, for example when a preseason game extends past the regular season start date. In that case, both the preseason and the regular season would be listed on the overlapping date. |
| season.details.name | The simple descriptive name of the season. |
| season.details.slug | This "slug" can be used throughout all other feeds where you need to specify the **{season-name}**. |
| season.details.startDate, season.details.endDate | The start and end date for the season, listed in YYYY-MM-DD format. |
| season.details.intervalType | Will be one of: **preseason**, **regular**, or **playoffs**. |
| season.supportedTeamStats | An array of team stats that are supported for the season. |
| season.supportedPlayerStats | An array of player stats that are supported for the season. |
| season.supportedTeamStats. teamStat.category, season.supportedPlayerStats. playerStat.category | Overall category name for the stat, each stat belongs to a category. |
| season.supportedTeamStats. teamStat.name, season.supportedPlayerStats. playerStat.name | General description of the stat. |
| season.supportedTeamStats. teamStat.abbreviation, season.supportedPlayerStats. playerStat.abbreviation | The stat abbreviations are particularly useful when you want to filter only specific stats in some feeds. They can be used within the **teamstats=** or **playerstats=** parameter for all feeds which support filtering of stats. Just specify the URL-encoded abbreviations as a comma-delimited list as demonstrated in the various examples for other feeds. |

Latest Updates (v2.1)

Lists all the latest update timestamps for each feed.

\*Addon Required\*: None (CORE)

URL Examples:

**CSV**:

**JSON**:

**XML**:

**XSD (for XML)**:

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/**latest\_updates**.{format}

with optional parameters:

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {format} | one of **csv**, **xml**, or **json** |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated, specified in ISO-8601 format. |
| feedUpdates | An array of entries for each feed and it's update status(es). |
| feedUpdates.feed | Identifiying details for the feed. |
| feedUpdates.lastUpdatedOn | A timestamp indicating when this feed was last modified, expressed in ISO-8601 format. |
| feedUpdates.forDate | If the feed supports multiple dates, each available date is listed here as an array of entries with timestamps for each date. |
| feedUpdates.forGame | If the feed supports multiple games, each available game is listed here as an array of game entries. |
| feedUpdates.forGame.game | Identifying details for the game. |

Seasonal Venues (v2.1)

Lists all venues used in a league's specific season.

\*Addon Required\*: None (CORE)

URL Examples:

**CSV**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/venues.csv

**JSON**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/venues.json

**XML**: https://api.mysportsfeeds.com/v2.1/pull/nhl/2025-2026-regular/venues.xml

**XSD (for XML)**: https://api.mysportsfeeds.com/v2.1/xsd/nhl/seasonal\_venues

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

The following describes each property of the feed's returned response. Static samples are no longer available without making a valid request to the API.

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ISO-8601 format in UTC timezone. |
| venues | An array of venues in use for the given season. |
| venues.venue | The details for the venue. |
| venues.homeTeam | Details for the team who calls the venue "home". |

Daily Player Gamelogs (v2.1)

All player game logs for a date including game and stats.

\*Addon Required\*: STATS

URL Examples:

**CSV**:

**JSON**:

**XML**:

**XSD (for XML)**:

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/date/{date}/**player\_gamelogs**.{format}

with optional parameters:

**team={list-of-teams}** *(filter teams)*

**player={list-of-players}** *(filter players)*

**position={list-of-positions}** *(filter player positions)*

**game={list-of-games}** *(filter on games)*

**stats={list-of-stats}** *(filter stats)*

**sort={sort-specifier}** *(sort the feed's content)*

**offset={offset-specifier}** *(filter results starting at the given offset)*

**limit={limit-specifier}** *(limit the maximum # of results)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {date} | a valid date in the form of **YYYYMMDD** |
| {format} | one of **csv**, **xml**, or **json** |
| game={game-identifier} | Must represent a valid game. Specify this value by concatenating  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation)  Optionally, in cases where multiple games are scheduled on the same day between the same teams (MLB double-headers in particular), you will also need to specify the index of the game based on start time. This is only required if there are multiple applicable games between the teams on the same day.  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation) + "-" (game index)  *Note:* If you specify a non-existent game, or an exact match cannot be otherwise found, then a 404 response code will be returned. Valid examples are:   * **20161221-BAL-DET** * **20161221-BAL-DET-1** * **20161221-BAL-DET-2** |
| team={list-of-teams} | (*optional*) comma-delimited list of one or more teams.  If this parameter is omitted, by default ALL teams are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a team. Each is case-insensitive, and strips out any punctuation.  Team ID:  **24**  Team abbreviation:  **bos**  City and team name:  **boston-celtics** |
| player={list-of-players} | (*optional*) comma-delimited list of one or more players.  If this parameter is omitted, by default ALL players are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a player. Each is case-insensitive, and strips out any punctuation.  Last name only:  **smith**  First and Last names:  **joe-smith**  In case several players with teh same name exist, you can also include the player ID:  **joe-smith-1234** |
| position={list-of-positions} | (*optional*) comma-delimited list of one or more player positions.  If this parameter is omitted, by default ALL positions are included.  The following are the acceptable formats for a position. Each is case-insensitive, and strips out any punctuation.  Position abbreviation:  **qb** |
| game={list-of-games} | (*optional*) comma-delimited list of one or more games.  If this parameter is omitted, by default ALL games are included.  Must represent a valid game. Specify this value by concatenating  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation)  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified. |
| stats={list-of-stats} | (*optional*) comma-delimited list with one or more stat abbreviations.  If this parameter is omitted, by default ALL stats are included. To include NO stats, set to **none**.  A full list of available stats and their abbreviations can be obtained by requesting the **Current Season** feed. |
| sort={sort-specifier} | (*optional*) Specifies a sorting order for the results. Format is:  {prefix}.{value}.{sort-order}  The {sort-order} is optional, and must be either "**A**" for Ascending, or "**D**" for descending.  Valid values for this feed are:  **game.starttime**  **player.lastname**  **player.age**  **player.birthplace**  **player.birthdate**  **player.height**  **player.weight**  **player.position**  **player.injury**  **player.team**  **player.number**  **team.city**  **team.name**  **team.abbr** |
| offset={offset-specifier} | (*optional*) Specifies the starting offset for the results. Defaults to 0. |
| limit={limit-specifier} | (*optional*) Specifies the maxmimum number of results. If not specified, all results are returned. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

The following describes each property of the feed's returned response. Static samples are no longer available without making a valid request to the API.

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ET timezone. In some cases this will be set to null when the feed has been craeted for the first time. This is not intended behaviour and will be corrected when time allows. |
| gamelog | An array of gamelog entries. |
| gamelog.game | A shorthand reference for the game. |
| gamelog.player | A shorthand reference for the player.  Possible positions are:   * **C** * **LW** * **RW** * **D** * **G** |
| gamelog.team | A shorthand reference to the player's team for the corresponding game. |
| gamelog.stats | A list of the player's stats for the specified game. If a list of comma-delimited stats abbreviations has been specified, only those applicable stats will be listed.  The stats listed here are not exhaustive, as suggested by the ellipses (**...**). In fact, the supported stats can potentially change from season to season. (Although that will mostly only be the case for older seasons)  For a complete list of supported stats for each season, use the **Current Season** feeed. |

Daily Team Gamelogs (v2.1)

All team game logs for a date including game and stats.

\*Addon Required\*: STATS

URL Examples:

**CSV**:

**JSON**:

**XML**:

**XSD (for XML)**:

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/date/{date}/**team\_gamelogs**.{format}

with optional parameters:

**team={list-of-teams}** *(filter teams)*

**game={list-of-games}** *(filter on games)*

**stats={list-of-stats}** *(filter stats)*

**sort={sort-specifier}** *(sort the feed's content)*

**offset={offset-specifier}** *(filter results starting at the given offset)*

**limit={limit-specifier}** *(limit the maximum # of results)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {date} | a valid date in the form of **YYYYMMDD** |
| {format} | one of **csv**, **xml**, or **json** |
| game={game-identifier} | Must represent a valid game. Specify this value by concatenating  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation)  Optionally, in cases where multiple games are scheduled on the same day between the same teams (MLB double-headers in particular), you will also need to specify the index of the game based on start time. This is only required if there are multiple applicable games between the teams on the same day.  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation) + "-" (game index)  *Note:* If you specify a non-existent game, or an exact match cannot be otherwise found, then a 404 response code will be returned. Valid examples are:   * **20161221-BAL-DET** * **20161221-BAL-DET-1** * **20161221-BAL-DET-2** |
| team={list-of-teams} | (*optional*) comma-delimited list of one or more teams.  If this parameter is omitted, by default ALL teams are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a team. Each is case-insensitive, and strips out any punctuation.  Team ID:  **24**  Team abbreviation:  **bos**  City and team name:  **boston-celtics** |
| game={list-of-games} | (*optional*) comma-delimited list of one or more games.  If this parameter is omitted, by default ALL games are included.  Must represent a valid game. Specify this value by concatenating  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation)  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified. |
| stats={list-of-stats} | (*optional*) comma-delimited list with one or more stat abbreviations.  If this parameter is omitted, by default ALL stats are included. To include NO stats, set to **none**.  A full list of available stats and their abbreviations can be obtained by requesting the **Current Season** feed. |
| sort={sort-specifier} | (*optional*) Specifies a sorting order for the results. Format is:  {prefix}.{value}.{sort-order}  The {sort-order} is optional, and must be either "**A**" for Ascending, or "**D**" for descending.  Valid values for this feed are:  **game.starttime**  **team.city**  **team.name**  **team.abbr** |
| offset={offset-specifier} | (*optional*) Specifies the starting offset for the results. Defaults to 0. |
| limit={limit-specifier} | (*optional*) Specifies the maxmimum number of results. If not specified, all results are returned. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ET timezone. In some cases this will be set to null when the feed has been created for the first time. This is not intended behaviour and will be corrected when time allows. |
| gamelog | An array of gamelog entries. |
| gamelog.game | A shorthand reference for the game. |
| gamelog.team | A shorthand reference to the player's team for the corresponding game. |
| gamelog.stats | A list of the player's stats for the specified game. If a list of comma-delimited stats abbreviations has been specified, only those applicable stats will be listed.  The stats listed here are not exhaustive, as suggested by the ellipses (**...**). In fact, the supported stats can potentially change from season to season. (Although that will mostly only be the case for older seasons)  For a complete list of supported stats for each season, use the **Current Season** feeed. |

Seasonal Team Stats (v2.1)

Lists each team along with their seasonal stats totals on a given day.

\*Addon Required\*: STATS

URL Examples:

**CSV**:

**JSON**:

**XML**:

**XSD (for XML)**:

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/**team\_stats\_totals**.{format}

with optional parameters:

**team={list-of-teams}** *(filter teams)*

**date={date-range}** *(filter on dates)*

**stats={list-of-stats}** *(filter stats)*

**sort={sort-specifier}** *(sort the feed's content)*

**offset={offset-specifier}** *(filter results starting at the given offset)*

**limit={limit-specifier}** *(limit the maximum # of results)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {format} | one of **csv**, **xml**, or **json** |
| date={date-or-range-of-dates} | (*optional*) a single date or a range of dates.  If this parameter is omitted, by default ALL dates are included.  There are several acceptable formats for a date. Each is case-insensitive.  Yesterday:  **yesterday**  Today:  **today**  Tomorrow:  **tomorrow**  Simple YYYYMMDD format:  **20170427**  Since a given date:  **since-yesterday** **since-20170426** **since-3-days-ago** **since-3-weeks-ago** **since-3-months-ago**  Until a given date:  **until-tomorrow** **until-20170501** **until-3-days-from-now** **until-3-weeks-from-now** **until-3-months-from-now**  Between two specified dates:  **from-yesterday-to-today** **from-20170426-to-tomorrow** **from-3-days-ago-to-3-days-from-now** |
| team={list-of-teams} | (*optional*) comma-delimited list of one or more teams.  If this parameter is omitted, by default ALL teams are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a team. Each is case-insensitive, and strips out any punctuation.  Team ID:  **24**  Team abbreviation:  **bos**  City and team name:  **boston-celtics** |
| stats={list-of-stats} | (*optional*) comma-delimited list with one or more stat abbreviations.  If this parameter is omitted, by default ALL stats are included. To include NO stats, set to **none**.  A full list of available stats and their abbreviations can be obtained by requesting the **Current Season** feed. |
| sort={sort-specifier} | (*optional*) Specifies a sorting order for the results. Format is:  {prefix}.{value}.{sort-order}  The {sort-order} is optional, and must be either "**A**" for Ascending, or "**D**" for descending.  Valid values for this feed are:  **team.city**  **team.name**  **team.abbr** |
| offset={offset-specifier} | (*optional*) Specifies the starting offset for the results. Defaults to 0. |
| limit={limit-specifier} | (*optional*) Specifies the maxmimum number of results. If not specified, all results are returned. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

Seasonal Team Stats (v2.1)

Lists each team along with their seasonal stats totals on a given day.

\*Addon Required\*: STATS

URL Examples:

**CSV**:

**JSON**:

**XML**:

**XSD (for XML)**:

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

The following describes each property of the feed's returned response. Static samples are no longer available without making a valid request to the API.

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ET timezone. In some cases this will be set to null when the feed has been created for the first time. This is not intended behaviour and will be corrected when time allows. |
| teamStatsTotals | An array of team stats totals entries. |
| teamStatsTotals.team | The team's details. |
| teamStatsTotals.stats | A list of the team's stats totals for the season. If a list of comma-delimited stats abbreviations has been specified, only those applicable stats will be listed.  The stats listed here are not exhaustive, as suggested by the ellipses (**...**). In fact, the supported stats can potentially change from season to season. (Although that will mostly only be the case for older seasons)  For a complete list of supported stats for each season, use the **Current Season** feeed. |

Seasonal Player Stats (v2.1)

Lists each player along with their seasonal stats totals on a given day.

\*Addon Required\*: STATS

URL Examples:

**CSV**:

**JSON**:

**XML**:

**XSD (for XML)**:

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/**player\_stats\_totals**.{format}

with optional parameters:

**player={list-of-players}** *(filter players)*

**position={list-of-positions}** *(filter player positions)*

**country={list-of-countries}** *(filter player countries of birth)*

**team={list-of-teams}** *(filter teams)*

**date={date-range}** *(filter on dates)*

**stats={list-of-stats}** *(filter stats)*

**sort={sort-specifier}** *(sort the feed's content)*

**offset={offset-specifier}** *(filter results starting at the given offset)*

**limit={limit-specifier}** *(limit the maximum # of results)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {format} | one of **csv**, **xml**, or **json** |
| date={date-or-range-of-dates} | (*optional*) a single date or a range of dates.  If this parameter is omitted, by default ALL dates are included.  There are several acceptable formats for a date. Each is case-insensitive.  Yesterday:  **yesterday**  Today:  **today**  Tomorrow:  **tomorrow**  Simple YYYYMMDD format:  **20170427**  Since a given date:  **since-yesterday** **since-20170426** **since-3-days-ago** **since-3-weeks-ago** **since-3-months-ago**  Until a given date:  **until-tomorrow** **until-20170501** **until-3-days-from-now** **until-3-weeks-from-now** **until-3-months-from-now**  Between two specified dates:  **from-yesterday-to-today** **from-20170426-to-tomorrow** **from-3-days-ago-to-3-days-from-now** |
| team={list-of-teams} | (*optional*) comma-delimited list of one or more teams.  If this parameter is omitted, by default ALL teams are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a team. Each is case-insensitive, and strips out any punctuation.  Team ID:  **24**  Team abbreviation:  **bos**  City and team name:  **boston-celtics** |
| player={list-of-players} | (*optional*) comma-delimited list of one or more players.  If this parameter is omitted, by default ALL players are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a player. Each is case-insensitive, and strips out any punctuation.  Last name only:  **smith**  First and Last names:  **joe-smith**  In case several players with teh same name exist, you can also include the player ID:  **joe-smith-1234** |
| position={list-of-positions} | (*optional*) comma-delimited list of one or more player positions.  If this parameter is omitted, by default ALL positions are included.  The following are the acceptable formats for a position. Each is case-insensitive, and strips out any punctuation.  Position abbreviation:  **qb** |
| country={list-of-countries} | (*optional*) comma-delimited list of one or more countries.  If this parameter is omitted, by default ALL countries are included.  Values are case-insensitive, and stripped of any punctuation.  Example:  **usa** |
| stats={list-of-stats} | (*optional*) comma-delimited list with one or more stat abbreviations.  If this parameter is omitted, by default ALL stats are included. To include NO stats, set to **none**.  A full list of available stats and their abbreviations can be obtained by requesting the **Current Season** feed. |
| sort={sort-specifier} | (*optional*) Specifies a sorting order for the results. Format is:  {prefix}.{value}.{sort-order}  The {sort-order} is optional, and must be either "**A**" for Ascending, or "**D**" for descending.  Valid values for this feed are:  **player.lastname**  **player.age**  **player.birthplace**  **player.birthdate**  **player.height**  **player.weight**  **player.position**  **player.injury**  **player.team**  **player.number**  **team.city**  **team.name**  **team.abbr** |
| offset={offset-specifier} | (*optional*) Specifies the starting offset for the results. Defaults to 0. |
| limit={limit-specifier} | (*optional*) Specifies the maxmimum number of results. If not specified, all results are returned. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

The following describes each property of the feed's returned response. Static samples are no longer available without making a valid request to the API.

{

"lastUpdatedOn": "2018-08-21T12:52:07.968Z",

"playerStatsTotals": [

{

"player": {

"id": 4419,

"firstName": "Justin",

"lastName": "Abdelkader",

"primaryPosition": "LW",

"jerseyNumber": 8,

"currentTeam": {

"id": 16,

"abbreviation": "DET"

},

"currentRosterStatus": "ROSTER",

"currentInjury": null,

"height": "6'2\"",

"weight": 214,

"birthDate": "1987-02-25",

"age": 31,

"birthCity": "Muskegon, MI",

"birthCountry": "USA",

"rookie": false,

"highSchool": null,

"college": null,

"handedness": {

"shoots": "L"

},

"officialImageSrc": null,

"socialMediaAccounts": [

{

"mediaType": "TWITTER",

"value": "justinabss"

}

]

},

"team": {

"id": 16,

"abbreviation": "DET"

},

"stats": {

"gamesPlayed": 75,

"scoring": {

"goals": 13,

"assists": 22,

"points": 35,

"hatTricks": 0,

"powerplayGoals": 4,

"powerplayAssists": 5,

"powerplayPoints": 9,

"shorthandedGoals": 0,

"shorthandedAssists": 0,

"shorthandedPoints": 0,

"gameWinningGoals": 0,

"gameTyingGoals": 0

},

"skating": {

"plusMinus": -11,

"shots": 110,

"shotPercentage": 11.8,

"blockedShots": 40,

"hits": 174,

"faceoffs": 97,

"faceoffWins": 47,

"faceoffLosses": 50,

"faceoffPercent": 48.5

},

"penalties": {

"penalties": 29,

"penaltyMinutes": 78

}

}

},

{

"player": {

"id": 9554,

"firstName": "Pontus",

"lastName": "Aberg",

"primaryPosition": "LW",

"jerseyNumber": 46,

"currentTeam": {

"id": 24,

"abbreviation": "EDM"

},

"currentRosterStatus": "ROSTER",

"currentInjury": null,

"height": "5'11\"",

"weight": 196,

"birthDate": "1993-09-22",

"age": 24,

"birthCity": "Stockholm",

"birthCountry": "Sweden",

"rookie": false,

"highSchool": null,

"college": null,

"handedness": {

"shoots": "R"

},

"officialImageSrc": null,

"socialMediaAccounts": []

},

"team": {

"id": 18,

"abbreviation": "NSH"

},

"stats": {

"gamesPlayed": 37,

"scoring": {

"goals": 2,

"assists": 6,

"points": 8,

"hatTricks": 0,

"powerplayGoals": 0,

"powerplayAssists": 0,

"powerplayPoints": 0,

"shorthandedGoals": 0,

"shorthandedAssists": 0,

"shorthandedPoints": 0,

"gameWinningGoals": 2,

"gameTyingGoals": 0

},

"skating": {

"plusMinus": 8,

"shots": 39,

"shotPercentage": 5.1,

"blockedShots": 7,

"hits": 16,

"faceoffs": 10,

"faceoffWins": 4,

"faceoffLosses": 6,

"faceoffPercent": 40

},

"penalties": {

"penalties": 4,

"penaltyMinutes": 8

}

}

},

{

"player": {

"id": 9554,

"firstName": "Pontus",

"lastName": "Aberg",

"primaryPosition": "LW",

"jerseyNumber": 46,

"currentTeam": {

"id": 24,

"abbreviation": "EDM"

},

"currentRosterStatus": "ROSTER",

"currentInjury": null,

"height": "5'11\"",

"weight": 196,

"birthDate": "1993-09-22",

"age": 24,

"birthCity": "Stockholm",

"birthCountry": "Sweden",

"rookie": false,

"highSchool": null,

"college": null,

"handedness": {

"shoots": "R"

},

"officialImageSrc": null,

"socialMediaAccounts": []

},

"team": {

"id": 24,

"abbreviation": "EDM"

},

"stats": {

"gamesPlayed": 16,

"scoring": {

"goals": 2,

"assists": 6,

"points": 8,

"hatTricks": 0,

"powerplayGoals": 0,

"powerplayAssists": 1,

"powerplayPoints": 1,

"shorthandedGoals": 0,

"shorthandedAssists": 0,

"shorthandedPoints": 0,

"gameWinningGoals": 1,

"gameTyingGoals": 0

},

"skating": {

"plusMinus": 1,

"shots": 31,

"shotPercentage": 6.5,

"blockedShots": 1,

"hits": 8,

"faceoffs": 2,

"faceoffWins": 0,

"faceoffLosses": 2,

"faceoffPercent": 0

},

"penalties": {

"penalties": 1,

"penaltyMinutes": 2

}

}

}

],

"references": {

"teamReferences": [

{

"id": 16,

"city": "Detroit",

"name": "Red Wings",

"abbreviation": "DET",

"homeVenue": {

"id": 145,

"name": "Little Caesars Arena"

},

"teamColoursHex": [],

"socialMediaAccounts": [],

"officialLogoImageSrc": null

},

{

"id": 18,

"city": "Nashville",

"name": "Predators",

"abbreviation": "NSH",

"homeVenue": {

"id": 15,

"name": "Bridgestone Arena"

},

"teamColoursHex": [],

"socialMediaAccounts": [],

"officialLogoImageSrc": null

},

{

"id": 24,

"city": "Edmonton",

"name": "Oilers",

"abbreviation": "EDM",

"homeVenue": {

"id": 22,

"name": "Rogers Place"

},

"teamColoursHex": [],

"socialMediaAccounts": [],

"officialLogoImageSrc": null

}

],

"playerReferences": [

{

"id": 4419,

"firstName": "Justin",

"lastName": "Abdelkader",

"primaryPosition": "LW",

"jerseyNumber": 8,

"currentTeam": {

"id": 16,

"abbreviation": "DET"

},

"currentRosterStatus": "ROSTER",

"currentInjury": null,

"height": "6'2\"",

"weight": 214,

"birthDate": "1987-02-25",

"age": 31,

"birthCity": "Muskegon, MI",

"birthCountry": "USA",

"rookie": false,

"highSchool": null,

"college": null,

"handedness": {

"shoots": "L"

},

"officialImageSrc": null,

"socialMediaAccounts": [

{

"mediaType": "TWITTER",

"value": "justinabss"

}

]

},

{

"id": 9554,

"firstName": "Pontus",

"lastName": "Aberg",

"primaryPosition": "LW",

"jerseyNumber": 46,

"currentTeam": {

"id": 24,

"abbreviation": "EDM"

},

"currentRosterStatus": "ROSTER",

"currentInjury": null,

"height": "5'11\"",

"weight": 196,

"birthDate": "1993-09-22",

"age": 24,

"birthCity": "Stockholm",

"birthCountry": "Sweden",

"rookie": false,

"highSchool": null,

"college": null,

"handedness": {

"shoots": "R"

},

"officialImageSrc": null,

"socialMediaAccounts": []

}

],

"playerStatReferences": [

{

"category": "Scoring",

"fullName": "goals",

"description": "Goals",

"abbreviation": "G",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "assists",

"description": "Assists",

"abbreviation": "A",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "points",

"description": "Points",

"abbreviation": "Pts",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "plusMinus",

"description": "Plus/Minus",

"abbreviation": "+/-",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "hatTricks",

"description": "Hat Tricks",

"abbreviation": "Hat",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "shots",

"description": "Shots",

"abbreviation": "Sh",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "shotPercentage",

"description": "Shot Percentage",

"abbreviation": "Sh%",

"type": "DECIMAL"

},

{

"category": "Penalties",

"fullName": "penalties",

"description": "Penalties",

"abbreviation": "Pn",

"type": "INTEGER"

},

{

"category": "Penalties",

"fullName": "penaltyMinutes",

"description": "Penalty Minutes",

"abbreviation": "PIM",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "powerplayGoals",

"description": "Powerplay Goals",

"abbreviation": "PPG",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "powerplayAssists",

"description": "Powerplay Assists",

"abbreviation": "PPA",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "powerplayPoints",

"description": "Powerplay Points",

"abbreviation": "PPPts",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "shorthandedGoals",

"description": "Shorthanded Goals",

"abbreviation": "SHG",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "shorthandedAssists",

"description": "Shorthanded Assists",

"abbreviation": "SHA",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "shorthandedPoints",

"description": "Shorthanded Points",

"abbreviation": "SHPts",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "gameWinningGoals",

"description": "Game Winning Goals",

"abbreviation": "GWG",

"type": "INTEGER"

},

{

"category": "Scoring",

"fullName": "gameTyingGoals",

"description": "Game Tying Goals",

"abbreviation": "GTG",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "hits",

"description": "Hits",

"abbreviation": "Ht",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "faceoffs",

"description": "Faceoffs",

"abbreviation": "FO",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "faceoffWins",

"description": "FaceoffWins",

"abbreviation": "F/O W",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "faceoffLosses",

"description": "FaceoffLosses",

"abbreviation": "F/O L",

"type": "INTEGER"

},

{

"category": "Skating",

"fullName": "faceoffPercent",

"description": "FaceoffPct",

"abbreviation": "F/O %",

"type": "DECIMAL"

},

{

"category": "Skating",

"fullName": "blockedShots",

"description": "Blocked Shots",

"abbreviation": "BS",

"type": "INTEGER"

}

]

}

}

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ET timezone. In some cases this will be set to null when the feed has been created for the first time. This is not intended behaviour and will be corrected when time allows. |
| playerStatsTotals | An array of player stats totals entries. |
| playerStatsTotals.player | The player's details. |
| playerStatsTotals.team | The player's team details. |
| playerStatsTotals.stats | A list of the player's stats totals for the season. If a list of comma-delimited stats abbreviations has been specified, only those applicable stats will be listed.  The stats listed here are not exhaustive, as suggested by the ellipses (**...**). In fact, the supported stats can potentially change from season to season. (Although that will mostly only be the case for older seasons)  For a complete list of supported stats for each season, use the **Current Season** feeed. |

Daily Game Lines (Odds) (v2.1)

All Game Lines (Odds) entries for a given date including line history.

\*Addon Required\*: ODDS

URL Examples:

**JSON**:

**XML**:

**XSD (for XML)**:

* [PULL URL Syntax](javascript:void(0);)
* [Response](javascript:void(0);)
* [Code Samples](javascript:void(0);)

Issue a HTTPS GET request to:

https://api.mysportsfeeds.com/v2.1/pull/nhl/{season}/date/{date}/**odds\_gamelines**.{format}

with optional parameters:

**game={list-of-games}** *(filter on games)*

**team={list-of-teams}** *(filter teams)*

**source={list-of-sources}** *(filter odds sources)*

**force={force-if-not-modified}** *(force content)*

where:

| **Parameter** | **Description** |
| --- | --- |
| {season} | a value obtained by concatenating  (season start year) + "-" + (season end year) + "-" + either "**regular**" or "**playoff**", depending on the season's type  *Note:* if the starting and ending years are the same, you can just specify the single year.  You can also use the special keywords **current**, **latest** or **upcoming**.  Using **current** relates to the current in-progress season (whether regular or playoff). If a current season is not in progress (offseason), you'll receive an empty response with response code **400**. Using **latest** will return the latest season (whether regular or playoff).  Note that **current** and **latest** will in fact apply to the same season if a season is in progress.  Using **upcoming** will only apply to a season which has been added, but has not yet actually started. This applies in cases where the upcoming season's game schedule is available, but there is no curent season underway (offseason, or between regular and playoff seasons). If no applicable season is available, an empty 400 response is returned.  Valid examples are:   * **2016-2017-regular** * **2016-playoff** * **current** * **latest** * **upcoming** |
| {date} | a valid date in the form of **YYYYMMDD** |
| {format} | one of **csv**, **xml**, or **json** |
| game={game-identifier} | Must represent a valid game. Specify this value by concatenating  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation)  Optionally, in cases where multiple games are scheduled on the same day between the same teams (MLB double-headers in particular), you will also need to specify the index of the game based on start time. This is only required if there are multiple applicable games between the teams on the same day.  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation) + "-" (game index)  *Note:* If you specify a non-existent game, or an exact match cannot be otherwise found, then a 404 response code will be returned. Valid examples are:   * **20161221-BAL-DET** * **20161221-BAL-DET-1** * **20161221-BAL-DET-2** |
| team={list-of-teams} | (*optional*) comma-delimited list of one or more teams.  If this parameter is omitted, by default ALL teams are included.  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified.  There are several acceptable formats for a team. Each is case-insensitive, and strips out any punctuation.  Team ID:  **24**  Team abbreviation:  **bos**  City and team name:  **boston-celtics** |
| source={list-of-sources} | (*optional*) comma-delimited list of one or more odds sources.  If this parameter is omitted, by default ALL sources are included.  Values are case-insensitive, and stripped of any punctuation.  The list of possible sources will vary and increase as we add support for additional sources. But any of the listed names in the feeds cn be used, such as:   * **bovada** * **888sports** * **betonline** |
| game={list-of-games} | (*optional*) comma-delimited list of one or more games.  If this parameter is omitted, by default ALL games are included.  Must represent a valid game. Specify this value by concatenating  (game date as YYYYMMDD) + "-" + (away team abbreviation) + "-" + (home team abbreviation)  In addition, for either the **player\_gamelogs** or **team\_gamelogs**, at least one player, team, or game MUST be specified. |
| force={force-if-not-modified} | (*optional*) Valid values are **true** or **false**.  Specifies whether to "force" content to be returned and avoid an empty 304 response. (See Response Codes section for more details)  Defaults to "true", which will always return the most up-to-date content based on your subscription level.  Specifying "false" will also avoid any throttling restrictions on your requests. See the Throttling section for more. |

Properties

| **Name** | **Details** |
| --- | --- |
| lastUpdatedOn | Indicates the time that this feed's content was generated. Specified in ISO-8601 format in UTC timezone. |
| gameLines | A list of all games along with their lines from various sources. |
| gameLines.game | A short summary of the game. A longer summary can be found in the **references.gameReferences** section near the bottom. |
| gameLines.lines | A list of the lines for the associated game, from various sources. |
| gameLines.lines.source | A summary of the sportsbook or vegas source for the lines. The number of supported sources will increase over time. |
| gameLines.lines.moneyLines, gameLines.lines.pointSpreads, gameLines.lines.overUnders, gameLines.lines.futures, gameLines.lines.props | A historical list of all known lines, futures and props from the associated game and source. If empty, no lines were available or known for the game and source.  **\*VERY IMPORTANT\***: additional entries are only made here if the line was detected to have moved from the previous line for the same source and the same game segment (FULL, FIRST\_QUARTER, FIRST\_HALF, etc). |
| \*.asOfTime | The time of the line capture, represented in ISO-8601 format. |
| \*.gameSegment | The specific segment of the game this line pertains to. Possible values are:   * **FULL** * **FIRST\_HALF** * **SECOND\_HALF** * **FIRST\_QUARTER** * **SECOND\_QUARTER** * **THIRD\_QUARTER** * **FOURTH\_QUARTER** * **FIRST\_FIVE\_INNINGS** * **LAST\_FOUR\_INNINGS** * **FIRST\_INNING** * **SECOND\_INNING** * **THIRD\_INNING** * **FOURTH\_INNING** * **FIFTH\_INNING** * **SIXTH\_INNING** * **SEVENTH\_INNING** * **EIGHTH\_INNING** * **NINETH\_INNING** * **FIRST\_PERIOD** * **SECOND\_PERIOD** * **THIRD\_PERIOD** * **REGULATION\_ONLY** |
| \*.awayLine, \*.homeLine, \*.drawLine, \*.overLine, \*.underLine |  |