TRUMEDIA

TruMedia

Baseball API Documentation

Website

Getting Started

Username: This is the email address associated with your TruMedia account.

Sitename: This is the name of your site. This is typically the first part of the URL of your site without the sport suffix (ex: sitename of *dodgers.trumedianetworks.com* is *dodgers*).

Temporary Token:

Temporary authentication tokens allow a URL to be authorized and accessed without login. In order to create a temporary token, a master token is required. The master token should remain private. Only the temporary token is exposed in the URL. Temporary authentication tokens will last for 24 hours by default, but can be requested to last for any duration up to 72 hours.

The two primary use cases for temporary authentication tokens are:

- To implement an SSO integration allowing users of your internal site to access the TruMedia site without being required to log in separately.
- To access the TruMedia APIs

Master Token:

These are currently provided by TruMedia.

Getting Started

Using your Username, Sitename, and Master Token, you can use the code below to create a Temporary Token:

Code Template - Paste In Your Username, Sitename, and Master Token (for Python)

```
import requests
import json
import pandas as pd
from io import StringIO
import urllib.parse
# Headers
headers = {"Content-Type": "application/json"}
# Data payload
data = {"username": "user@email.com", "sitename": "sitename", "token": "master token"}
# POST request
url = "https://api.trumedianetworks.com/v1/siteadmin/api/createTempPBToken"
res = requests.post(url, headers=headers, data=json.dumps(data))
# Parse response as JSON
token = res.json()
# Extract temp token
tok = token.get("pbTempToken")
print(tok)
```

If you need more information regarding your username, sitename, or master token, please contact mlbsupport@trumedianetworks.com or ncaasupport@trumedianetworks.com.

Pulling Data

A user can pull data by editing parts of the query URL. Below is the URL:

```
api_url = (f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/{dataFormat.csv}"
           f"?param1={param1}&param2={param2}&columns={columns_encoded}"
           f"{filters}{qualifications}{sort}&token={tok}").
```

Adjusting Data Format:

This will adjust the format that the data is presented, changing what each row represents, The options for dataFormat are `AllGames, AllTeams, TeamTotals, TeamSeasons, TeamGames, TeamPlays, PlayerTotals, PlayerSeasons, PlayerGames, PlayerPlays, GamePitches, GamePitchesTrackman (NCAA Only), and GameVideo.`

You can also change the file type to JSON or CSV. `To change the file type, reference the dataFormat section and change between` ***`dataFormat.csv`*** `and` ***`dataFormat.json`***.

Adding Parameters:

This can narrow the dataset to only what you want to see. In order to add parameters, you will need to declare the parameter and its settings before the query URL. In the query URL, you will need to add the filter and its contents into the URL. To see all available parameters, please visit this page 8.

Note: For every additional parameter, you will need to include an ampersand ("&") when listing them in the query URL.

Adding Columns: Using the native stat abbreviations, you can list the columns you want to pull in this format: `columns <- "[stat1],[stat2]".`

To see all available stats, please visit this the glossary on the TruMedia site:
https://*insert_your_site_url_here*.trumedianetworks.com/baseball/glossary?pd=%7B"activeTab"%3A"stats"%7D

Note: Make sure to use brackets around stat abbreviations and do not leave any spaces in column listing.

Pulling Data

A user can pull data by editing parts of the query URL. Below is the URL

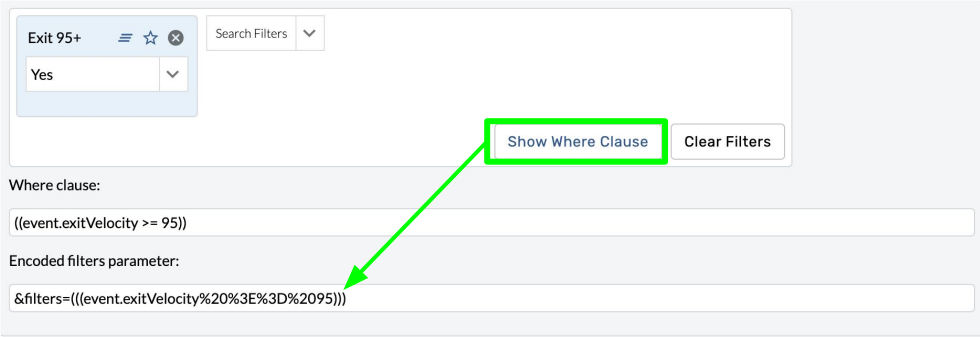
```
api_url = (f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/dataFormat.csv"
  f"?param1={param1}&param2={param2}&columns={columns_encoded}"
  f"{filters}{qualifications}{sort}&token={tok}")
```

Adding Filters:

To add filters to your query URL, you can use this helpful tool by visiting this tool. Put your site URL prefix into the red italicized section below:

https://*insert_your_site_url_here*.trumedianetworks.com/baseball/tools?pd=%7B"activeTab"%3A"queryTool"%7D

Click on the 'Search Filters' dropdown and search for the desired filters. Once complete, click on the 'Show Where Clause' button and copy the text from the second section into your filter declaration statement (ex: `filters = "&filters=(((event.exitVelocity%20%3E%3D%2095)))"`)



Pulling Data

A user can pull data by editing parts of the query URL. Below is the URL

```
api_url = (f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/dataFormat.csv"
      f"?param1={param1}&param2={param2}&columns={columns_encoded}"
      f"{filters}{qualifications}{sort}&token={tok}")
```

Adding Qualifications:

Optional qualification clause for limiting results. Applies to all queries except Plays queries. This parameter is essentially like using the Qualifications UI control on a player or team page. The URL string is `"&qualification=%5Bstat1%5D+%3E%3D+number"`. For example, if you wanted to set a qualification for the query results to only include players with 100 or more innings pitched, you would use the following as your qualifications declaration statement (printed before the query): `qualifications = "&qualification=%5BIP%5D+%3E%3D+100"`

Note: Qualifications will be applied before filters in the query statement. For example, if a user builds a query to find all whiff rates among pitchers with 100+ innings, the 100+ attempt qualification will be implemented first (narrowing the query to players with 100+ total innings) and the whiff rates with the applied filters will follow.

Adding Sort:

Optional order clause for sorting the results. This parameter is essentially like clicking on a stat column header in the UI to sort the report by that stat. The URL string is `"&sort=%5Bstat1%5D+ORDER"`. For example, if you wanted to sort the query results by passing attempts descending, you would use the following as your qualifications declaration statement (printed before the query): `sort <- "&sort=%5BIP%5D+DESC"`.

Complete Examples

Below is an example of pulling ***all games*** *from the 2025 season*:

```
#Authentification
Copy and paste code from  Slide 3
 #Setting parameters for the 2025 Season
seasonYear= 2025
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/AllGames.csv"
   f"?seasonYear={seasonYear}&token={tok}"
    )
data = pd.read_csv(api_url)
print(data.head(5))
```

Below is an example of pulling ***all teams*** *from the 2025 season*:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for the 2025 Season
seasonYear = 2025
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/AllTeams.csv"
   f"?seasonYear={seasonYear}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example of pulling ***team aggregated*** basic pitching stats (raw numeric values) for home games only during the 2024 regular season sorted by ERA ascending:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for the 2024 Regular Season
seasonYear = 2024
seasonType = "REG"
#Setting the columns
columns = "[IP],[ERA],[FIP],[WHIP],[K/9]"
columns_encoded = urllib.parse.quote(columns)
#Adding filter for home games (copied from results from  filter URL generation process)
filters = "&filters=((((team.game.home%20AND%20NOT%20game.venueFlipped)%20OR%20(team.game.away%20AND%20game.venueFlipped))))"
#Adding sort for ERA ascending
sort = "&sort=%5BERA%5D+ASC"
#Adding format parameter to retrieve raw figures
format = "RAW"
#Data Query
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/TeamTotals.csv"
   f"?seasonYear={seasonYear}&seasonType={seasonType}&columns={columns_encoded}"
   f"&format={format}{filters}{sort}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling basic pitching stats by team season with RISP in the 2025 regular season among teams with at least 100 innings pitched sorted by batting average descending:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for the 2025 Regular Season
seasonYear = 2025
seasonType = "REG"
#Setting the columns
columns = "[IP],[InZoneWhiff%|PIT],[Chase%|PIT],[FPStk%|PIT],[BA|PIT]"
columns_encoded = urllib.parse.quote(columns)
#Adding filter for RISP (copied from results from  filter URL generation process )
filters = "&filters=(((event.manOnSecond%20OR%20event.manOnThird)))"
#Adding qualification for teams with at least 100 IP
qualifications = "&qualification=%5BIP%5D+%3E%3D+100"
#Adding sort for batting average against descending
sort = "&sort=%5BBA|PIT%5D+DESC"
#Data Query
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/TeamSeasons.csv"
   f"?seasonYear={seasonYear}&seasonType={seasonType}&columns={columns_encoded}"
   f"{filters}{qualifications}{sort}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling basic team hitting stats for LSU by game for the 2023 season:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for the 2023 season
seasonYear = 2023
teamId = 5323
#Setting the columns
columns = "[PA],[H],[XBH],[BB],[HBP],[HR],[AVG],[SLG]"
# Encode columns
columns_encoded = urllib.parse.quote(columns, safe="[],%")
#Data Query
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/TeamGames.csv"
   f"?seasonYear={seasonYear}&teamId={teamId}&columns={columns_encoded}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling all home runs for LSU during the 2023 season including exit velocities:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for LSU for the 2023 season
seasonYear = 2023
teamId = 5323
statEvent = "[HR]"
#Setting the columns
columns = "event.exitVelocity"
# Encode statEvent and columns to handle special characters
statEvent_encoded = urllib.parse.quote(statEvent)
columns_encoded = urllib.parse.quote(columns)
#Data Query
api_url = (
    f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/TeamPlays.csv"
    f"?seasonYear={seasonYear}&teamId={teamId}&statEvent={statEvent_encoded}"
    f"&columns={columns_encoded}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling basic total player hitting stats at the D1 level against left-handed pitching in the 2025 regular season of players with at least 100 total plate appearances, sorted by number of home runs descending:

```
#Authentification
Copy and paste code from Slide 3
#Setting parameters for the 2025 Regular Season
seasonYear = 2025
seasonType = "REG"
#Setting the columns
columns = "[PA],[AVG],[SLG],[HR],[XBH]"
# Encode columns so special characters don't break the URL
columns_encoded = urllib.parse.quote(columns)
#Adding filter for left-handed pitchers and D1 teams (copied from results from  filter URL generation process )
filters = "&filters=(((season.seasonLevel%20IN%20('BBC'%2C'SFT')%20AND%20team.game.gameLeague%20%3D%20'D1'))%20AND%20 ((event
.pitcherHand%20%3D%20'L')))"
#Adding qualification for players with at least 100 plate appearances
qualifications = "&qualification=%5BPA%5D+%3E%3D+100"
#Adding sort for home runs descending
sort = "&sort=%5BHR%5D+DESC"
#Data Query
api_url = (
   f"https://api.trumedianetworks .com/v1/mlbapi/custom/baseball/DirectedQuery/PlayerTotals.csv"
   f"?seasonYear={seasonYear}&seasonType={seasonType}&columns={columns_encoded}"
   f"{filters}{qualifications}{sort}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling player season total batting stats for LSU players from the 2020 through 2023 regular seasons with at least a .400 slugging percentage:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for LSU in the Regular Season
seasonType = "REG"
teamId = 5323
#Setting the columns
columns = "[AB],[Barrel%],[BABIP],[ExitVel],[SLG]"
# Encode columns properly (leave brackets, commas, and % unencoded since API often expects them)
columns_encoded = urllib.parse.quote(columns)
#Adding filter for seasons from 2020 to 2023 (copied from results from  filter URL generation process )
filters = "&filters=((season.seasonYear%20%3E%3D%202020)%20AND%20(season.seasonYear%20%3C%3D%202023))"
#Adding qualification for players with at least a .400 slugging percentage
qualifications = "&qualification=%5BSLG%5D+%3E%3D+.400"
#Data Query
api_url = (
    f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/PlayerSeasons.csv"
    f"?seasonType={seasonType}&teamId={teamId}&columns={columns_encoded}"
    f"{qualifications}{filters}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling basic pitching stats for Paul Skenes for the 2023 season:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for Paul Skenes for the 2023 season
seasonYear = 2023
playerId = 1648798793
#Setting the columns
columns = "[IP],[ERA],[WHIP],[K|PIT],[BB|PIT],[FBVel]"
# URL-encode columns so special characters like | and [] don't break the URL
columns_encoded = urllib.parse.quote(columns)
#Data Query
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/PlayerGames.csv"
   f"?seasonYear={seasonYear}&playerId={playerId}&columns={columns_encoded}&token={tok}"
)
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling game video for the Princeton @ Miami game on February 23rd, 2025 (NCAA only) :

```
#Authentification
Copy and paste code from  Slide 3
#Choose the game you want
trackmanGameID = "20250223-UofMiami-1"
#Data Query
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/GamePitchesTrackman.csv"
   f"?trackmanGameId={trackmanGameId}&token={tok}")
data = pd.read_csv(api_url)
print(data.head(5))
```

Complete Examples

Below is an example pulling all pitches for the Princeton @ Miami game on February 23rd, 2025 with game ID:

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for the particular game
gameId = "253656885"
#Data Query
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/GamePitchesTrackman.csv"
   f"?gameId={gameID}&token={tok}")
data = pd.read_csv(api_url)
print(data.head(5))
```

Below is an example pulling all trackman data for the Princeton @ Miami game on February 23rd, 2025 with trackman game ID (Only applicable to NCAA):

```
#Authentification
Copy and paste code from  Slide 3
#Setting parameters for the particular game
trackmanGameID = "20250223-UofMiami-1"
#Data Query
api_url = (
   f"https://api.trumedianetworks.com/v1/mlbapi/custom/baseball/DirectedQuery/GamePitchesTrackman.csv"
   f"?trackmanGameId={trackmanGameID}&token={tok}")
data = pd.read_csv(api_url)
print(data.head(5))
```

Full List of Parameters

gameId: Satisfies requirement for all queries. Comma delimited list of NCAA game ids and MLB Game PK values. Max of 100 game ID's for team queries, 25 for player queries.

gameString: Satisfies requirement for all queries. Comma delimited list of MLB Game String values (format: gid_YYYY_MM_DD_HOMETM_AWAYTM_N). Max of 100 game strings for team queries, 25 for player queries.

seasonLevel: Satisfies requirement for Team Totals and Team Seasons queries. Level of play (MLB, AAA, AAX, AFA, AFX, ASX, ROK, AFL, MEX, WIN, NPB, KBO)

seasonYear: Satisfies requirement for Team Totals and Team Seasons queries. Year of season

- For single years: `seasonYear <- 2023;`
- For multiple years: `seasonYear <- "2021%2C2022%2C2023"`

gameType: Comma delimited list of game types (REG, WC, DS, LCS, WS, PRE) or (REG, PLY) for NCAA baseball.

teamId: Comma delimited list of up to 10 Team ID's (except for Team Plays query which takes only 1 team ID). Satisfies requirement for all team queries as well as the Player Totals and Players Seasons queries.

orgId: MLB Org ID. Satisfies requirement for Team Totals and Team Seasons queries.

opponentId: Comma delimited list of up to 10 Team ID's

playerId: Comma delimited list of up to 50 Player ID's (except for Player Plays query which takes only 1 player ID). Satisfies requirement for all player queries.

columns: Optional additional comma delimited columns of data to return (generally stats).

filters: These are additional parameters for the TruMedia query WHERE clause. They can be used to narrow the hand of the batter or pitcher, the date range, pitch type, and any other attribute you want to filter on. More information on this parameter can be found via the UI under Tools → Filter Syntax Tool.

statEvent: Required for the Team Plays and Player Plays query, this defines the stat for events being queried. This parameter is essentially like clicking on a stat in the UI to get the events that relate to that stat. See the example All home runs in the 2022 regular season, which uses [HR] for the statEvent parameter.

qualification: Optional qualification clause for limiting results. More information on this parameter can be found in the "TruMedia Data Service Filter Querying" document.

sort: Optional order clause for sorting the results. More information on this parameter can be found in the "TruMedia Data Service Filter Querying" document.

format: FORMATTED - formatted value (ex: '.123', '12%', etc); RAW - raw decimal values; MIXED - For JSON only, an array with both the formatted and raw values.