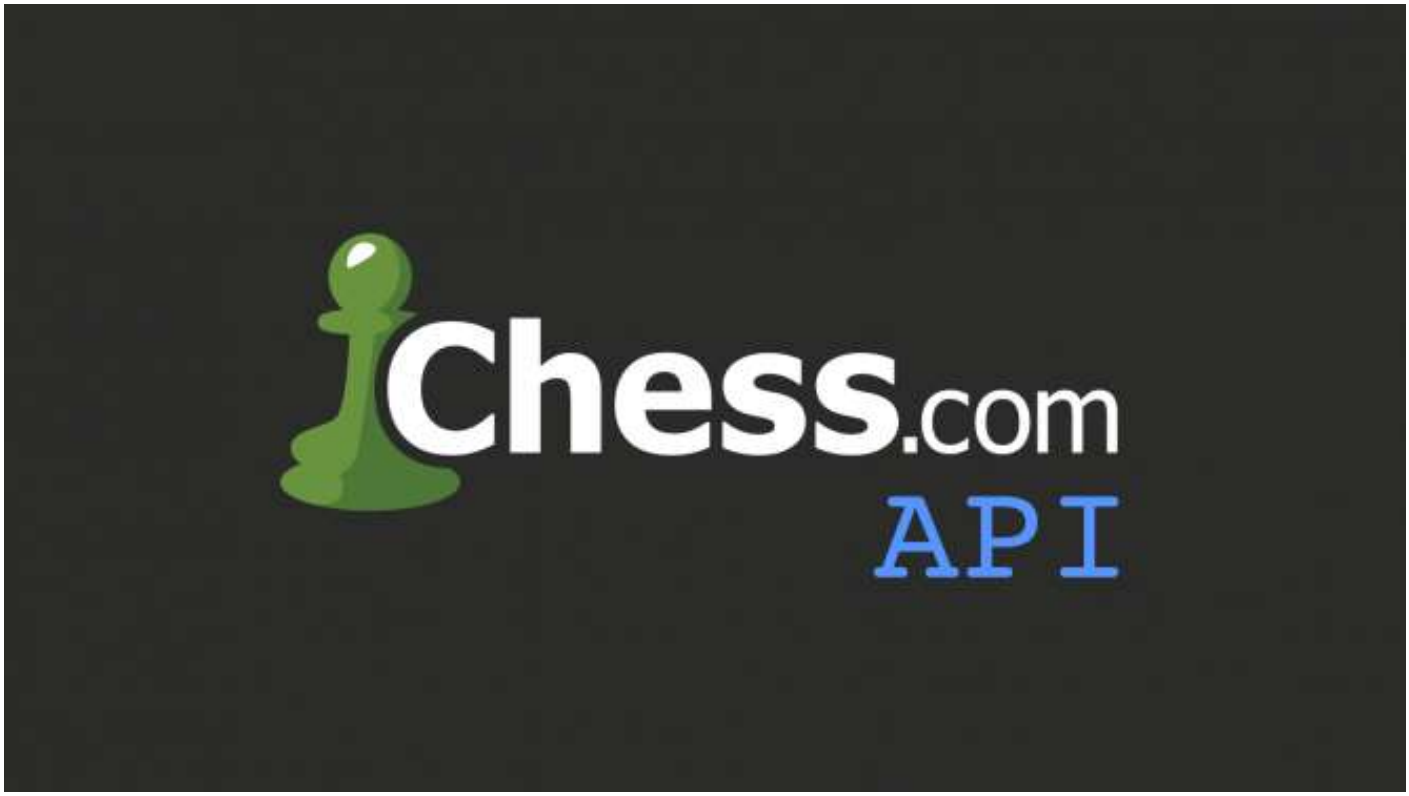
Announcements



Published-Data API

bcurtis
Updated: May 10, 2022, 8:59 AM | 180

The PubAPI is a read-only REST API that responds with JSON-LD data. Our goal is to re-package all currently public data from the website and make it available via the PubAPI. "Public Data" is information available to people who are not logged in, such as player data, game data, and club/tournament information. This excludes private information that is restricted to the logged in user, such as game chat and conditional moves.

This is read-only data. You cannot send game-moves or other commands to Chess.com from this system. If you wish to send commands, you will be interested in the Interactive API releasing later this year.

To use the PubAPI:

1. determine the data you want, and compose the URL for it based on the endpoint URL pattern

2. request that URL in your browser, program, Postman, cURL, or pigeon.
3. enjoy the JSON 🙂

This page is the documentation for the PubAPI. We will edit this page to make corrections and add new documentation as the API grows. We will date these changes and report them in the CHANGELOG so that you can tell what is fresh. Please keep the comments focused on the documentation itself, and use the Club Forums to discuss suggestions, bug reports, and how you use the data!

Table of Contents

- Tournaments:
  - Tournament
  - Tournament round
  - Tournament round's group
- Team matches:
  - Daily Team Match
  - Daily Team Match board
  - Live Team Match
  - Live Team Match board
- Countries:
  - Country Profile
  - List of players
  - List of clubs
- Daily puzzle:
  - Daily puzzle
  - Random daily puzzle
- Streamers:
  - Streamers
- Leaderboards:
  - Leaderboards
- Game results codes

General Use Features and Issues

The format for each endpoint will be different, and described in a release note about that endpoint. They all share the following features and issues. Even though you only need a URL to obtain the data, we advise that you read through and understand all of these before you start any significant work.

Cache invalidation: This endpoints refresh at most once every 12 hours

Not (Yet) Guaranteed Current

About 3% of players are still actively using the old "v2" website for some actions. When these players perform an action that modifies the data you are requesting, the data may be out of date when we deliver it to you. We expect to remove the option of using v2 in the coming months. Until

then, you can communicate to your users that data accuracy will be improved if they and their opponents are all using the new v3 website.

This issue does not apply to people using the mobile apps.

English

URL responses are the same for everyone, no matter who or where they are. This speeds up delivery, but also means that in cases where the data contain words (for example, reasons for game ending, error responses), these words will be in English.

Rate Limiting

Your serial access rate is unlimited. If you always wait to receive the response to your previous request before making your next request, then you should never encounter rate limiting.

However, if you make requests in parallel (for example, in a threaded application or a webserver handling multiple simultaneous requests), then some requests may be blocked depending on how much work it takes to fulfill your previous request. You should be prepared to accept a "`429 Too Many Requests`" response from our server for any non-serial request that you make.

In some cases, if we detect abnormal or suspicious activity, we may block your application entirely. If you supply a recognizable user-agent that contains contact information, then if we must block you application we will attempt to contact you to correct the problem.

How to get the PubAPI data

The PubAPI endpoints can be called, by any kind of client or browsers that supports the HTTP protocol.

So if you only want to see what happens when you issue a *request* to an endpoint you can open your browser and write in the address bar the endpoint you are interested in (for example the Profile endpoint.

The *response* is compressed, but there are many tools only that can render that response in a more readable format (gor example https://jsonformatter.curiousconcept.com).

If you have access to a Command Prompt/Terminal Window you can use the `curl` command that will give you more useful information.
For example the command "`curl -v https://api.chess.com/pub/player/hikaru`" will return, among with the response, a lot of useful data such as the `response code`, `Etag`, `date`, `last-modified` that are very useful to check if you are retrieving fresh or cached data.

If you want to play with cdata retrieved by `curl` you can use jq as was shown in these brilliant examples by **@pmaugeri**.

If you don't want to deal with command line there are many free tools online that provide an easier user interface. The ones that we use mostly in our team are Postman and Insomnia.

If you use Postman or Insomnia, **we have released a collection of endpoints** that you can download from HERE. After installing the collection you will find all the current endpoints that you can call by a click (and analyze the response, convert the call into a language of your choice etc.) . If you need instructions on how import collections, check this link for Postman or this link for Insomnia.

All time fields are expressed as "timestamp", which is a number calculated as the time measured in the number of **seconds** since the Unix Epoch (January 1 1970 00:00:00 GMT). All programming languages have functions to convert timestamps to an human-readable format, but if you want to try by hand you can use one of the many online converters.

HTTP Responses

Response codes are used to tell your client about the state of the data you requests.

- `200` = "enjoy your JSON"
- `301` = if the URL you requested is bad, but we know where it should be; your client should remember and correct this to use the new URL in future requests
- `304` = if your client supports "`ETag/If-None-Match`" or "`Last-Modified/If-Modified-Since`" caching headers and the data have not changed since the last request
- `404` = we try to tell you if the URL is malformed or the data requested is just not available (e.g., a username for a user that does not exist)
- `410` = we know for certain that no data will ever be available at the URL you requested; your client should not request this URL again
- `429` = we are refusing to interpret your request due to rate limits; see "Rate Limiting" above

JSON-LD

We provide "linked data" contexts for each of our JSON data formats. JSON-LD can be parsed as regular JSON (it is fully compatible), but we also provide an HTTP header called "`Link`" which contains a URL to the JSON-LD context for this data format. This helps the data describe its own meaning. You can read more about JSON-LD by reading the spec or searching Google in your language.

JSONP

In JSONP, or "JSON with padding," a "`callback`" parameter can be sent in the query string of any URL, and the value will be treated as the name of a function to call with the data as a parameter. This is useful if you want to point to the URL in a <script> tag's src value, instead of calling the URL programmatically in your code. As an example, compare these:

- https://api.chess.com/pub/player/erik
- https://api.chess.com/pub/player/erik?callback=myJavascriptFunction

JSONP function names containing non literal chars or longer than 200 chars will be removed from response.

Caching

Each response has "`ETag`" and "`Last-Modified`" headers. If your client supplies the proper header in the request to prove that the data have not changed since your previous request ("`If-None-Match`" and "`If-Modified-Since`", respectively), then you will receive a `304 Not Modified` response code, telling you that it is safe and correct to use your cached version.

If you make a request more frequently than the "`Cache-Control`" header's "max-age" value, then our CDN may respond to you directly with cached values. At refresh rates greater than specified in the "max-age" value, the CDN will validate the currency of its response before sending it to you. You may see headers indicating a cache "HIT", "MISS", "EXPIRED", or "REVALIDATED" for this purpose.

Please note: The endpoints refresh at most once every 24 hours, if not noted otherwise.

HTTP Compression & HTTP/2

If the request is made with an "`Accept-Encoding: gzip`" header, then the response will be gzip compressed for transfer if it is of a size where this saves time (generally, more than 200 bytes). With gzip'd transfers, you can save up to 80% of the download bandwidth for the same data.

HTTP/2 requests will get HTTP/2 responses. For the most part, this means header compression / binary transfer, and multiplexed responses for parallel downloads. However, see above-mentioned rate limits on parallel requests.

Endpoints

Player Profile

Description: Get additional details about a player in a game.

URL pattern: https://api.chess.com/pub/player/{username}

Data format:

```
{
  "@id": "URL", // the location of this profile (always self-referencing)
  "url": "URL", // the chess.com user's profile page (the username is displayed with the original letter case)
  "username": "string", // the username of this player
  "player_id": 41, // the non-changing Chess.com ID of this player
  "title": "string", // (optional) abbreviation of chess title, if any
  "status": "string", // account status: closed, closed:fair_play_violations, basic, premium, mod, staff
  "name": "string", // (optional) the personal first and last name
  "avatar": "URL", // (optional) URL of a 200x200 image
  "location": "string", // (optional) the city or location
  "country": "URL", // API location of this player's country's profile
  "joined": 1178556600, // timestamp of registration on Chess.com
  "last_online": 1500661803, // timestamp of the most recent login
  "followers": 17 // the number of players tracking this player's activity
  "is_streamer": "boolean", //if the member is a Chess.com streamer
  "twitch_url": "Twitch.tv URL",
  "fide": "integer" // FIDE rating
}
```

JSON-LD Context: https://api.chess.com/context/Player.jsonld

Example: https://api.chess.com/pub/player/erik

Note: the "player_id" is provided as a convenience to determine when a username has been changed. If you retrieve a Player Profile by the username-based URL linked from a Game or other object, and this new Player Profile has a "player_id" that matches a Profile you previously downloaded, then you can safely assume that this new Profile replaces the old, and all URLs with the previous username will now be found under the new username. This should be an extremely rare occurrence. This "player_id" will never change for a given account, however the future availability of this ID is not guaranteed.

Titled Players

Description: List of titled-player usernames.

URL pattern: https://api.chess.com/pub/titled/{title-abbrev}

Valid title abbreviations are: GM, WGM, IM, WIM, FM, WFM, NM, WNM, CM, WCM.

Data Format:

```
{
  "players": [
    /* array of usernames for players with this title */
  ]
}
```

JSON-LD contexts: https://api.chess.com/context/Players.jsonld

Example: https://api.chess.com/pub/titled/GM

Player Stats

Description: Get ratings, win/loss, and other stats about a player's game play, tactics, lessons and Puzzle Rush score.

URL pattern: https://api.chess.com/pub/player/{username}/stats

The response contains many "stats objects," each identified by the ratings-type from the game, composed of the rules code, an underscore, and the time-class code. If a person has not played a particular game type, no stats object will be present for it. Like so:

```
{
  "chess_daily": {
    /* stats object for games of rules "chess" and "daily" time-class */
  },
  "chess960_daily": {
    /* stats object for games of rules "chess960" and "daily" time-class */
  },
  "chess_blitz": {
    /* stats object for games of rules "chess" and "blitz" time-class */
  },
  "tactics": {
     "highest": {
```

```
            "rating": "integer",
            "date": "timestamp"
        },
        "lowest": {
            "rating": "integer",
            "date": "timestamp"
        }
    },
    "lessons":{
        "highest": {
            "rating": "integer",
            "date": "timestamp"
        },
        "lowest": {
            "rating": "integer",
            "date": "timestamp"
        }
    },
    "puzzle_rush": {
        "daily":{
            "total_attempts": "integer",
            "score": "integer"
         },
        "best": {
            "total_attempts": "integer",
            "score": "integer"
         }
    }
}
```

Each stats object will contain only information that has been collected and is not "default". For example, if a player has not won any games in a game type, the "best" stats will not be present; if they have not played in tournaments, then the tournaments stats will not be present. Not all stats are collected for all game types, and the available data may change from time to time as we gather

more information. Tactics, Lessons and Puzzle Rush stats may be missing, depending on player activity.

Data Format, each Game-Type:

```
{
  "last": { // the current stats
    "date": 1509709165, // timestamp of the last rated game finished
    "rating": 1642, // most-recent rating
    "rd": 58 // the Glicko "RD" value used to calculate ratings changes
  },
  "best": { // the best rating achieved by a win
    "date": 1256228875, // timestamp of the best-win game
    "rating": 2065, // highest rating achieved
    "game": "URL" // URL of the best-win game
  },
  "record": { // summary of all games played
    "win": 177,  // number of games won
    "loss": 124, // number of games lost
    "draw": 21,  // number of games drawn
    "time_per_move": 18799, // integer number of seconds per average move
    "timeout_percent": 9.99 // timeout percentage in the last 90 days
  },
  "tournament": { // summary of tournaments participated in
    "count": 20,   // number of tournaments joined
    "withdraw": 1, // number of tournaments withdrawn from
    "points": 39,  // total number of points earned in tournaments
    "highest_finish": 1 // best tournament place
  }
}
```

JSON-LD contexts: *in progress*

Example: https://api.chess.com/pub/player/erik/stats

Player online status

Description: Tells if an unser has been online in the last five minutes.

URL pattern: https://api.chess.com/pub/player/{username}/is-online

Data Format:

```
{
  "online": boolean, //true or false
}
```

JSON-LD contexts: https://api.chess.com/context/PlayerIsOnline.jsonld

Example: https://api.chess.com/pub/player/erik/is-online

Player Games

There are five endpoints for a player's games: to-move, current, list of available monthly archives, monthly archive, and downloadable-PGN monthly archive. The to-move, current games, and monthly archive lists use this wrapping data format:

```
{
  "games": [
    /* array of Game objects in ascending game-end-time order */
  ]
}
```

If no games were played that period (that month, or "now"), this is an empty array.

These game objects share these element definitions:

- `pgn`: the PGN representation of the game.
- `rules`: to indicate chess-variant play. Possible values are: "chess", "chess960", "bughouse", "kingofthehill", "threecheck", "crazyhouse"
- `time_class`: ratings-group speed of the game. Possible values are: "daily", "rapid", "blitz", "bullet".
- `time_control`: specific time control used in the game, in the PGN standard notation.
- `last_activity`: in currently-playing (daily) games, this timestamp represents the time of the last action affecting the game. This may be a move, draw offer, or even a deactivation of the chat window. Omitted in archives, where it is synonymous with the `end_time`.

Current Daily Chess

Description: Array of Daily Chess games that a player is currently playing.

URL pattern: https://api.chess.com/pub/player/{username}/games

Data Format:

```
{
  "games": [
    {
      "white": "string", // URL of the white player's profile
      "black": "string", // URL of the black player's profile
      "url": "string", // URL of this game
      "fen": "string", // current FEN
      "pgn": "string", // current PGN
      "turn": "black", // player to move
      "move_by": 1501765498, // timestamp of when the next move must be made
                             // this is "0" if the player-to-move is on
vacation
      "draw_offer": "black", // (optional) player who has made a draw offer
      "last_activity": 1509810789, // timestamp of the last activity on the
game
      "start_time": 1254438881, // timestamp of the game start (Daily Chess
only)
      "time_control": "string", // PGN-compliant time control
      "time_class": "string", // time-per-move grouping, used for ratings
      "rules": "string", // game variant information (e.g., "chess960")
      "tournament": "string", //URL pointing to tournament (if available),
      "match": "string" //URL pointing to team match (if available)
    }
  ]
}
```

JSON-LD contexts: https://api.chess.com/context/ChessGames.jsonld, https://api.chess.com/context/ChessGame.jsonld

Example: https://api.chess.com/pub/player/erik/games

To-Move Daily Chess

Description: Array of Daily Chess games where it is the player's turn to act.

URL pattern: https://api.chess.com/pub/player/{username}/games/to-move

Data Format:

```
{
  "games": [
    {
      "url": "string", // URL of this game
      "move_by": 1254438881, // timestamp of the when the move must be made by
                             // this is "0" if it is not this player's turn
      "draw_offer": true, // (optional) this player has received a draw offer
      "last_activity": 1509810789, // timestamp of the last activity on the
game
    }
  ]
}
```

Note: this list will at times include some games where it is not the player's turn, if a draw offer has been made. In these cases, the `move_by` value is "0", and the game is sorted to the top to represent possible immediate action before the other player moves. Sorting is normally based on how soon a move must be made by.

JSON-LD contexts: https://api.chess.com/context/ChessGames.jsonld, https://api.chess.com/context/ChessGame.jsonld

Example: https://api.chess.com/pub/player/erik/games/to-move

List of Monthly Archives

Description: Array of monthly archives available for this player.

URL pattern: https://api.chess.com/pub/player/{username}/games/archives

Data Format:

```
{
  "archives": [
```

```
    /* array of URLs for monthly archives in ascending chronological order */
  ]
}
```

JSON-LD contexts: https://api.chess.com/context/GameArchives.jsonld

Example: https://api.chess.com/pub/player/erik/games/archives

Complete Monthly Archives

Description: Array of Live and Daily Chess games that a player has finished.

URL pattern: https://api.chess.com/pub/player/{username}/games/{YYYY}/{MM}

- "YYYY" is the four digit year of the game-end
- "MM" is the two-digit month

Data Format, each Game:

```
{
  "white": { // details of the white-piece player:
    "username": "string", // the username
    "rating": 1492, // the player's rating after the game finished
    "result": "string", // see "Game results codes" section
    "@id": "string" // URL of this player's profile
  },
  "black": { // details of the black-piece player:
    "username": "string", // the username
    "rating": 1942, // the player's rating after the game finished
    "result": "string", // see "Game results codes" section
    "@id": "string" // URL of this player's profile
  },
  "accuracies": { // player's accuracies, if they were previously calculated
    "white": float,
    "black": float
  },
  "url": "string", // URL of this game
  "fen": "string", // final FEN
```

```
  "pgn": "string", // final PGN
  "start_time": 1254438881, // timestamp of the game start (Daily Chess only)
  "end_time": 1254670734, // timestamp of the game end
  "time_control": "string", // PGN-compliant time control
  "rules": "string", // game variant information (e.g., "chess960")
  "eco": "string", //URL pointing to ECO opening (if available),
  "tournament": "string", //URL pointing to tournament (if available),
  "match": "string", //URL pointing to team match (if available)
}
```

JSON-LD contexts: https://api.chess.com/context/ChessGames.jsonld, https://api.chess.com/context/ChessGame.jsonld

Example: https://api.chess.com/pub/player/erik/games/2009/10

Multi-Game PGN Download

Description: standard multi-game format PGN containing all games for a month.

URL pattern: https://api.chess.com/pub/player/{username}/games/{YYYY}/{MM}/pgn

Data Format: this download follows the PGN standard, and is not JSON-encoded. The response has two special headers worth noting if programmatically hitting this endpoint (browsers handle these automatically):

- `Content-Type: application/x-chess-pgn`
  This content type indicates the type of parser needed to understand the data
- `Content-Disposition: attachment; filename="ChessCom_username_YYYYMM.pgn"`
  This disposition indicates that browser should download, not display, the result, and it suggests a filename based on the source archive

Example: https://api.chess.com/pub/player/erik/games/2009/10/pgn

Player's Clubs

Description: List of clubs the player is a member of, with joined date and last activity date.

URL pattern: https://api.chess.com/pub/player/{username}/clubs

Data Format:

```
{
  "clubs": [
```

```
      {
         "@id": url, // URL of Club endpoint
         "name": string, // Club's name
         "last_activity": timestamp, //timestamp of last activity
         "icon": url, // Club's icon url
         "url": url, // Club's url
         "joined": timestamp  // Timestamp of when player joined the Club
      }
  ]
}
```

JSON-LD contexts: https://api.chess.com/context/PlayerClubs.jsonld

Example: https://api.chess.com/pub/player/erik/clubs

Player Matches

Description: List of Team matches the player has attended, is partecipating or is currently registered.

URL pattern: https://api.chess.com/pub/player/{username}/matches

Data Format:

```
{
    "finished": [
        /** List of matches */
        {
            "name": "Name of match",
            "url": "URL", // URL of match on web site
            "@id": "URL", // URL of PubAPI match endpoint
            "club": "URL", // URL of player's club endpoint
            "results": [
                "played_as_white": "win", //result of game played as white, see
"Game results codes" section
                "played_as_black": "win" //result of game played as black, see
"Game results codes" section
            ],
```

```
            "board": "URL", // URL of PubAPI match board endpoint
        }
    ],
    "in_progress": [
       /** List of matches */
        {
            "name": "Name of match",
            "url": "URL", // URL of match on web site
            "@id": "URL", // URL of PubAPI match endpoint
            "club": "URL", // URL of player's club endpoint
            "board": "URL", // URL of PubAPI match board endpoint
        }
    ],
    "registered": [
       /** List of matches */
        {
            "name": "Name of match",
            "url": "URL", // URL of match on web site
            "@id": "URL", // URL of PubAPI match endpoint
            "club": "URL", // URL of player's club endpoint
        }
    ]
  }
}
```

JSON-LD contexts: https://api.chess.com/context/PlayerMatches.jsonld

Example: https://api.chess.com/pub/player/erik/matches

Player's Tournaments

Description: List of tournaments the player is registered, is attending or has attended in the past.

URL pattern: https://api.chess.com/pub/player/{username}/tournaments

Data Format:

```
{
    "finished": [
        {
            "url": "URL", // link to the PubAPI URL of the tournament
            "@id": "URL", // link to the Web URL of the tournament
            "wins": 3, //number of wins
            "losses": 5, //number of losses
            "draws": 0, //number of draws
            "points_awarded": 0, //pints awarded
            "placement": 4, //placement
            "status": "eliminated", //final status of the player in this
tourmanent {winner, eliminated, withdrew, removed}
            "total_players": 5 //number of total players
        } /* array of finished tournaments */
    ],
    "in_progress": [
        {
            "url": "URL", // link to the PubAPI URL of the tournament
            "@id": "URL", // link to the Web URL of the tournament
            "status": "eliminated", //final status of the player in this
tourmanent {winner, eliminated, withdrew, removed}
        } /* array of in progress tournaments */
    ],
    "registered": [
        {
            "url": "URL", // link to the PubAPI URL of the tournament
            "@id": "URL", // link to the Web URL of the tournament
            "status": "invited" //current status of user {invited, registered}
        } /* array of registered tournaments */
    ]
}
```

JSON-LD contexts: https://api.chess.com/context/PlayerTournaments.jsonld

Example: https://api.chess.com/pub/player/erik/tournaments

Clubs

All club-based URLs use the club's "URL ID" to specify which club you want data for.

https://api.chess.com/pub/club/{url-ID}

The `url-ID` is the same as found in the URL for the club's web page on www.chess.com. For example, the url-ID of the Chess.com Developer's Club is `chess-com-developer-community`

Club Profile

Description: Get additional details about a club.

URL pattern: https://api.chess.com/pub/club/{url-ID}

Data format:

```
{
  "@id": "URL", // the location of this profile (always self-referencing)
  "name": "string", // the human-readable name of this club
  "club_id": 57796, // the non-changing Chess.com ID of this club
  "icon": "URL", // (optional) URL of a 200x200 image
  "country": "URL", // location of this club's country profile
  "average_daily_rating": 1376, //average daily rating
  "members_count": 54 //total members count
  "created": 1178556600, // timestamp of creation on Chess.com
  "last_activity": 1500661803, // timestamp of the most recent post, match, etc
  "visibility": "public", // whether the club is public or private
  "join_request": "URL", // location to submit a request to join this club
  "admin": [
    /* array of URLs to the player profiles for the admins of this club */
  ],
  "description": "string" // text description of the club
}
```

JSON-LD Context: https://api.chess.com/context/Club.jsonld

Example: https://api.chess.com/pub/club/chess-com-developer-community

Club Members

Description: List of club members (usernames and joined date timestamp), grouped by club-activity frequency. The club-activity is one of this actions:

- Viewing the club homepage
- Viewing the clubs news index or a specific news article (but not the notification message received that the news was published)
- Viewing the club's forums or a specific forum thread
- Changing their club settings, including modifying their membership; for admins, this includes inviting or authorizing new members
- Viewing the club's tournament, team match, or votechess lists
- Viewing club membership lists or running a related search, or viewing the leaderboards for the club

Please note: playing a club game is not counted as a club-activity.

URL pattern: https://api.chess.com/pub/club/{url-ID}/members

Cache invalidation: This endpoints refresh at most once every 12 hours

Data Format:

```
{
  "weekly": [
    {
        "username": "string", //username
        "joined": "integer",  //timestamp
    }
  ],
  "monthly": [
    {
        "username": "string", //username
        "joined": "integer",  //timestamp
    }
  ],
  "all_time": [
    {
        "username": "string", //username
```

```
        "joined": "integer",  //timestamp
    }
  ]
}
```

Note: these lists are only updated based on members joining, leaving, or being banned. A member listed as a monthly-active or "all_time" member may not be moved to the weekly-active list for up to 24 hours after posting, reading club news, or participating in club activities.

JSON-LD contexts: https://api.chess.com/context/ClubMembers.jsonld

Example: https://api.chess.com/pub/club/chess-com-developer-community/members

Club Matches

Description: List of daily and club matches, grouped by status (registered, in progress, finished).

URL pattern: https://api.chess.com/pub/club/{ID}/matches

```
{
    "finished": [
        /** List of matches */
        {
            "name": "match name", //the team match name
            "@id": "URL", // URL pointing to the team match endpoint
            "opponent": "https://api.chess-dev.com/pub/club/testing-teams", //
URL pointing to the opponent club endpoint
            "result": "win", // see "Game results codes" section
            "start_time": 1305324926, // timestamp of the match start
            "time_class": "daily"
        }
    ],
    "in_progress": [
        /** List of matches */
        {
            "name": "match name", //the team match name
            "@id": "URL", // URL pointing to the team match endpoint
            "opponent": "https://api.chess-dev.com/pub/club/testing-teams", //
```

```
URL pointing to the opponent club endpoint
            "start_time": 1305324926, // timestamp of the match start
            "time_class": "daily"
        }
    ],
    "registered": [
        /** List of matches */
        {
            "name": "match name", //the team match name
            "@id": "URL", // URL pointing to the team match endpoint
            "opponent": "https://api.chess-dev.com/pub/club/testing-teams", //
URL pointing to the opponent club endpoint
            "time_class": "daily"
        }
    ]
}
```

JSON-LD contexts: https://api.chess.com/context/ClubMatches.jsonld

Example: https://api.chess.com/pub/club/team-usa-southwest/matches

Tournaments

All tournaments-based URLs use the tournament's "URL ID" to specify which tournament you want data for.

https://api.chess.com/pub/tournament/{url-ID}

The `url-ID` is the same as found in the URL for the tournament's web page on www.chess.com. For example, the url-ID of the Chess.com Developer's Club is `-33rd-chesscom-quick-knockouts-1401-1600`

Tournament

Description: Get details about a daily, live and arena tournament.
URL pattern: https://api.chess.com/pub/tournament/{url-ID}
Data format:

```
{
    "name": "name",
```

```
    "url": "URL", //url to Web tournament's URL
    "description": "description",
    "creator": "username", //username of creator
    "status": "finished", //status of tournament {finished, in_progress,
registration}
    "finish_time": 1251846528, //timestamp of finish time, if tournament is
finished
    "settings": {
        "type": "round_robin",
        "rules": "string", // game variant information (e.g., "chess960")
        "time_class": "daily",
        "time_control": "1/259200",
        "is_rated": true,
        "is_official": false,
        "is_invite_only": false,
        "initial_group_size": 5,
        "user_advance_count": 1,
        "use_tiebreak": true,
        "allow_vacation": false,
        "winner_places": 1,
        "registered_user_count": 5,
        "games_per_opponent": 2,
        "total_rounds": 1,
        "concurrent_games_per_opponent": 1
    },
    "players": [
        /** List of tournament's rounds URL */
        {
            "username": "username",
            "status": "eliminated" //status of user
        }
    ],
    "rounds": [
        /** List of tournament's rounds URL */
```

```
    ]
}
```

JSON-LD Context: https://api.chess.com/context/Tournament.jsonld

Example: https://api.chess.com/pub/tournament/-33rd-chesscom-quick-knockouts-1401-1600

Tournament's round

Description: Get details about a tournament's round.

URL pattern: https://api.chess.com/pub/tournament/{url-ID}/{round}

Data format:

```
{
    "groups": [
        /** List of tournament's round groups */
    ],
    "players": [
        /** List of tournament's round players */
        {
            "username": "username",
            "is_advancing":false // {true, false}, only if this tournament is
completed
        }
    ]
}
```

JSON-LD Context: https://api.chess.com/context/TournamentRound.jsonld

Example: https://api.chess.com/pub/tournament/-33rd-chesscom-quick-knockouts-1401-1600/1

Tournament's round group

Description: Get details about a tournament's round group.

URL pattern: https://api.chess.com/pub/tournament/{url-ID}/{round}/{group}

Data format:

```
{
    "fair_play_removals":[
      //list of username accounts closed for fair play violation
    ],
```

```
    "games": [
      /** List of group's games */
      {
          "white": "string", // URL of the white player's profile
          "black": "string", // URL of the black player's profile
          "url": "string", // URL of this game
          "fen": "string", // current FEN
          "pgn": "string", // current PGN
          "turn": "black", // player to move
          "move_by": 1501765498, // timestamp of when the next move must be
made
                                 // this is "0" if the player-to-move is on
vacation
          "draw_offer": "black", // (optional) player who has made a draw offer
          "last_activity": 1509810789, // timestamp of the last activity on the
game
          "start_time": 1254438881, // timestamp of the game start (Daily Chess
only)
          "time_control": "string", // PGN-compliant time control
          "time_class": "string", // time-per-move grouping, used for ratings
          "rules": "string", // game variant information (e.g., "chess960")
          "eco": "string", //URL pointing to ECO opening (if available),
          "tournament": "string", //URL pointing to tournament (if available)
      }
    ],
    "players": [
      /** List of group's players */
        {
            "username": "username",
            "points": 2, //points earned by player in this group adjuested in
case of fair play recalculations)
            "tie_break": 6, //tie-break points by player earned in this group
            "is_advancing": false // {true, false}
        }
```

```
    ]
}
```

JSON-LD Context: https://api.chess.com/context/TournamentRoundGroup.jsonld
Example: https://api.chess.com/pub/tournament/-33rd-chesscom-quick-knockouts-1401-1600/1/1

Daily Team matches

All team matches-based URLs use the match "ID" to specify which match you want data for.

https://api.chess.com/pub/match/{ID}

The `ID` is the same as found in the URL for the team match web page on www.chess.com. For example, the ID WORLD LEAGUE Round 5: Romania vs USA Southwest is `12803`.

Team Match

Description: Get details about a team match and players playing that match. After the match is finished there will be a link to each player's stats endpoint, in order to get up-to-date information about the player.
URL pattern: https://api.chess.com/pub/match/{ID}
Data format of match in its registration phase:

```
{
    "name": string,
    "url": "URL", // the URL of the match on the website
    "description": string, // description
    "start_time" : timestamp, //manual or auto time start
    "settings":{
      "time_class": "daily",
      "time_control": string, // time control
      "initial_setup": ,
      "rules": string, // game variant information (e.g., "chess960")
      "min_team_players": integer, //minimum number of players per team
      "max_team_players": integer, //maximum number of players per team
      "min_required_games": integer, //minimum number of required games
      "min_rating": integer, //minimum rating of players to be admitted in this
match
```

```
      "max_rating": integer, //maximum rating of players to be admitted in this
match
      "autostart": boolean //if the match is set to automatically start
    },
    "status": "registration",
    "boards": integer, // number of boards
    "teams": {
        "team1": {
            "@id": "URL", // API URL for the club profile
            "url": "URL", // Web URL for the club profile
            "name": "string", // club name
            "score": score, // Team score (adjuested in case of fair play
recalculations)
            "players": [
                {
                    "username": "username",
                    "board": "url", // url of board
                    "rating": 1355, //rating of player
                    "rd": 25.12, //Glicko RD
                    "timeout_percent": 25.12, //timeout percentage in the last
90 days
                    "status": "basic" //status of user
                }
            ]
        },
        "team2": {
            "@id": "URL", // API URL for the club profile
            "url": "URL", // Web URL for the club profile
            "name": "string", // club name
            "score": score, // Team score (adjuested in case of fair play
recalculations)
            "players": [
                {
                    "username": "username",
```

```
                    "board": "url", // url of board
                    "rating": 1355, //rating of player
                    "timeout_percent": 25.21, //timeout percentage in the last
90 days
                    "status": "basic" //status of user
                }
            ]
        }
    }
}
```

Data format in others phases of match (in progress, finished):

```
{
    "name": string,
    "url": "URL", // the URL of the match on the website
    "description": string, // description
    "start_time" : timestamp, //manual or auto time start
    "settings":{
      "time_class": "string", // time-per-move grouping, used for ratings
      "time_control": string, // time control
      "rules": string, // game variant information (e.g., "chess960")
      "min_team_players":0, //minimum number of players per team
      "max_team_players":0, //maximum number of players per team
      "min_required_games":0, //minimum number of required games
      "autostart":false
    },
    "status": string, // {finished, in_progress},
    "boards": integer, // number of boards
    "teams": {
        "team1": {
            "@id": "URL", // API URL for the club profile
            "name": "string", // club name
```

```
            "score": score, // Team score (adjuested in case of fair play
 recalculations)
            "players": [
                {
                    "username": "username",
                    "board": "url", // url of board
                    "stats": "url", //url to player's stats
                    "played_as_white": "string", //result {win, lose, resign,
 etc.} of player when played as white (if game's finished)
                    "played_as_black": "string"  //result {win, lose, resign,
 etc.} of player when played as black (if game's finished)
                }
            ],
            "fair_play_removals":[
              //list of usernames that were closed during match
            ]
        },
        "team2": {
            "@id": "URL", // API URL for the club profile
            "name": "string", // club name
            "score": score, // Team score (adjuested in case of fair play
 recalculations)
            "players": [
                {
                    "username": "username",
                    "board": "url", // url of board
                    "stats": "url", //url to player's stats
                    "played_as_white": "string", //result {win, lose, resign,
 etc.} of player when played as white (if game's finished)
                    "played_as_black": "string"  //result {win, lose, resign,
 etc.} of player when played as black (if game's finished)
                }
            ],
            "fair_play_removals":[
```

```
            //list of usernames that were closed during match
        ]
    }
  }
}
```

Please note: we don't keep snapshots of user's statistics changes during matches, so after the registration phase you will able to get up-to-date statistics following the stats link of every player wich will lead to Player's stats endpoint.

JSON-LD Context: https://api.chess.com/context/Match.jsonld
Example: https://api.chess.com/pub/match/12803

Daily Team match board

Description: Get details about a team match board. Only in-progress or finished games will be included, so there may be one or two games in this list.
URL pattern: https://api.chess.com/pub/match/{ID}/{board}
Data format:

```
{
    "board_scores":{
      "player1": 0.5, // User score (adjuested in case of fair play
recalculations)
      "player2": 1.5 // User score (adjuested in case of fair play
recalculations)
    },
    "games": [
        {
            "white": { // details of the white-piece player:
              "username": "string", // the username
              "rating": 1492, // the player's rating after the game finished
              "result": "string", // if game's finished, see "Game results
codes" section
              "@id": "string", // URL of this player's profile
              "team": "url" // url to club's profile
            },
```

```
            "black": { // details of the black-piece player:
              "username": "string", // the username
              "rating": 1942, // the player's rating after the game finished

              "result": "string", // if game's finished, see "Game results
codes" section
              "@id": "string", // URL of this player's profile
              "team": "url" // url to club's profile
            },
            "accuracies": { // player's accuracies, if they were previously
calculated
              "white": float,
              "black": float
            }
            "url": "string", // URL of this game
            "fen": "string", // current FEN
            "pgn": "string", // final PGN, if game's finished
            "start_time": 1254438881, // timestamp of the game start (Daily
Chess only)
            "end_time": 1254670734, // timestamp of the game end, if game's
finished
            "time_control": "string", // PGN-compliant time control
            "time_class": "string", // time-per-move grouping, used for
ratings
            "rules": "string", // game variant information (e.g., "chess960")
            "eco": "string", //URL pointing to ECO opening (if available),
            "match": "string", //URL pointing to team match (if available)
        }
    ]
}
```

JSON-LD Context: https://api.chess.com/context/MatchBoard.jsonld

Example: https://api.chess.com/pub/match/12803/1

Live Team matches

All live team matches-based URLs use the match "ID" to specify which match you want data for.
https://api.chess.com/pub/match/{ID}

The ID is the same as found in the URL for the team match web page on www.chess.com. For example, the ID Friendly 5+2 is 5833.

Live Team Match

Description: Get details about a team match and players playing that match. After the match is finished there will be a link to each player's stats endpoint, in order to get up-to-date information about the player.

URL pattern: https://api.chess.com/pub/match/live{ID}

Data format of match whenn scheduled:


```
{
   "@id":"https://api.chess.com/pub/match/live/5861",
   "name":"Friendly 10|2 Rapid Open: Srbija Tim vs Team USA Live",
   "url":"https://www.chess.com/club/matches/live/5861",
   "start_time":1579988425,
   "status":"scheduled",
   "boards":0,
   "settings":{
      "rules":"chess",
      "time_class":"standard",
      "time_control":600,
      "time_increment":2,
      "min_team_players":1,
      "min_required_games":0,
      "autostart":false
   },
   "teams":{
      "team1":{
         "@id":"https://api.chess.com/pub/club/srbija-tim",
         "name":"Srbija Tim",
         "url":"https://www.chess.com/club/srbija-tim",
         "score":0,
```

```
            "players":[

            ],
            "fair_play_removals":[

            ]
         },
         "team2":{
            "@id":"https://api.chess.com/pub/club/team-usa-live",
            "name":"Team USA Live",
            "url":"https://www.chess.com/club/team-usa-live",
            "score":0,
            "players":[

            ],
            "fair_play_removals":[

            ]
         }
      }
   }
```


Data format when match is finished:


```
{
   "@id":"https://api.chess.com/pub/match/live/5833",
   "name":"Friendly 5+2",
   "url":"https://www.chess.com/club/matches/live/5833",
   "start_time":1579471260,
   "end_time":1579472487,
   "status":"finished",
   "boards":6,
   "settings":{
```

```
      "rules":"chess",
      "time_class":"blitz",
      "time_control":300,
      "time_increment":2,
      "min_team_players":2,
      "min_required_games":0,
      "autostart":false
   },
   "teams":{
      "team1":{
         "@id":"https://api.chess.com/pub/club/lynx-club",
         "name":"Lynx Club",
         "url":"https://www.chess.com/club/lynx-club",
         "score":7,
         "result":"win",
         "players":[
            {
               "username":"guilhermekk",
               "stats":"https://api.chess.com/pub/player/guilhermekk/stats",
               "status":"premium",
               "played_as_white":"win",
               "played_as_black":"resigned",
               "board":"https://api.chess.com/pub/match/5833/4"
            },
            {
               "username":"insan3_7",
               "stats":"https://api.chess.com/pub/player/insan3_7/stats",
               "status":"premium",
               "played_as_black":"checkmated",
               "played_as_white":"win",
               "board":"https://api.chess.com/pub/match/5833/1"
            },
            {
               "username":"jydra21",
```

```
            "stats":"https://api.chess.com/pub/player/jydra21/stats",
            "status":"basic",
            "played_as_black":"checkmated",
            "played_as_white":"insufficient",
            "board":"https://api.chess.com/pub/match/5833/5"
         },
         {
            "username":"luisen_17",
            "stats":"https://api.chess.com/pub/player/luisen_17/stats",
            "status":"basic",
            "played_as_white":"win",
            "played_as_black":"win",
            "board":"https://api.chess.com/pub/match/5833/2"
         },
         {
            "username":"over_9000_aka_max",

 "stats":"https://api.chess.com/pub/player/over_9000_aka_max/stats",
            "status":"premium",
            "played_as_white":"win",
            "played_as_black":"win",
            "board":"https://api.chess.com/pub/match/5833/6"
         },
         {
            "username":"rahara1988",
            "stats":"https://api.chess.com/pub/player/rahara1988/stats",
            "status":"premium",
            "played_as_black":"checkmated",
            "played_as_white":"agreed",
            "board":"https://api.chess.com/pub/match/5833/3"
         }
      ],
      "fair_play_removals":[
```

```
            ]
         },
         "team2":{
            "@id":"https://api.chess.com/pub/club/not-so-pro-chess-league",
            "name":"Not-So PRO Chess League",
            "url":"https://www.chess.com/club/not-so-pro-chess-league",
            "score":5,
            "result":"lose",
            "players":[
               {
                  "username":"25elevin",
                  "stats":"https://api.chess.com/pub/player/25elevin/stats",
                  "status":"premium",
                  "played_as_black":"timeout",
                  "played_as_white":"timeout",
                  "board":"https://api.chess.com/pub/match/5833/2"
               },
               {
                  "username":"alexkotow",
                  "stats":"https://api.chess.com/pub/player/alexkotow/stats",
                  "status":"basic",
                  "played_as_black":"resigned",
                  "played_as_white":"resigned",
                  "board":"https://api.chess.com/pub/match/5833/6"
               },
               {
                  "username":"checkmatemark04",

  "stats":"https://api.chess.com/pub/player/checkmatemark04/stats",
                  "status":"basic",
                  "played_as_white":"win",
                  "played_as_black":"agreed",
                  "board":"https://api.chess.com/pub/match/5833/3"
               },
```

```
            {
               "username":"sougataghosh",
               "stats":"https://api.chess.com/pub/player/sougataghosh/stats",
               "status":"premium",
               "played_as_white":"win",
               "played_as_black":"checkmated",
               "board":"https://api.chess.com/pub/match/5833/1"
            },
            {
               "username":"stompall",
               "stats":"https://api.chess.com/pub/player/stompall/stats",
               "status":"premium",
               "played_as_white":"win",
               "played_as_black":"insufficient",
               "board":"https://api.chess.com/pub/match/5833/5"
            },
            {
               "username":"vicountvonjames",

 "stats":"https://api.chess.com/pub/player/vicountvonjames/stats",
               "status":"basic",
               "played_as_black":"checkmated",
               "played_as_white":"win",
               "board":"https://api.chess.com/pub/match/5833/4"
            }
         ],
         "fair_play_removals":[

         ]
      }
   }
}
```

Please note: we don't keep snapshots of user's statistics changes during matches, so after the registration phase you will able to get up-to-date statistics following the stats link of every player wich will lead to Player's stats endpoint.

JSON-LD Context: https://api.chess.com/context/Match.jsonld

Example: https://api.chess.com/pub/match/live/5833

Live Team match board

Description: Get details about a team match board. Only in-progress or finished games will be included, so there may be one or two games in this list.

URL pattern: https://api.chess.com/pub/match/live/{ID}/{board}

Data format:

```
{
   "board_scores":{
      "stompall":1.5, // User score (adjuested in case of fair play
recalculations)
      "jydra21":0.5 // User score (adjuested in case of fair play
recalculations)
   },
   "games":[
      {
         "url":"https://www.chess.com/live/game/4415534343",
         "pgn":"PGN of the game",
         "time_control":"300+2",
         "end_time":1579471691,
         "rated":true,
         "fen":"r7/p4pN1/1pn4k/8/2bP3R/2P3R1/6PP/6K1 b - -",
         "time_class":"blitz",
         "rules":"chess",
         "white":{
            "rating":1351,
            "result":"win",
            "@id":"https://api.chess.com/pub/player/stompall",
            "username":"stompall"
```

```
         },
         "black":{
            "rating":1458,
            "result":"checkmated",
            "@id":"https://api.chess.com/pub/player/jydra21",
            "username":"JYDRA21"
         },
         "eco":"https://www.chess.com/openings/Kings-Gambit-Accepted-Modern-
Defense-4.exd5"
      },
      {
         "url":"https://www.chess.com/live/game/4415552847",
         "pgn":"Pgn of the game",
         "time_control":"300+2",
         "end_time":1579472482,
         "rated":true,
         "fen":"8/8/2k2K2/8/8/8/8/8 b - -",
         "time_class":"blitz",
         "rules":"chess",
         "white":{
            "rating":1456,
            "result":"insufficient",
            "@id":"https://api.chess.com/pub/player/jydra21",
            "username":"JYDRA21"
         },
         "black":{
            "rating":1353,
            "result":"insufficient",
            "@id":"https://api.chess.com/pub/player/stompall",
            "username":"stompall"
         },
         "eco":"https://www.chess.com/openings/Scotch-Game-Schmidt-Variation-
5.Nxc6-bxc6"
      }
```

```
    ]
}
```

JSON-LD Context: https://api.chess.com/context/MatchBoard.jsonld
Example: https://api.chess.com/pub/match/live/5833/5

Countries

All country-based URLs use the country's 2-character ISO 3166 code (capitalized) to specify which country you want data for.

https://api.chess.com/pub/country/{iso}

https://en.wikipedia.org/wiki/ISO_3166-1_alpha-2

Chess.com supports players and clubs identifying with some regions that are not recognized countries in this ISO list. These countries are identified with codes within the "user-assigned code" ranges. When possible, we tried to use codes that are commonly used in other applications. This is a list of these codes at Chess.com:

- XA : "Canary Islands"
- XB : "Basque Country"
- XC : "Catalonia"
- XE : "England"
- XG : "Galicia"
- XK : "Kosovo"
- XP : "Palestine"
- XS : "Scotland"
- XW : "Wales"
- XX : "International"

The presence or absence of any region on this list does not reflect any political opinion by Chess.com. We're just here to play chess. 🙂

Country Profile

Description: Get additional details about a country.
URL pattern: https://api.chess.com/pub/country/{iso}
Data format:

```
{
  "@id": "URL", // the location of this profile (always self-referencing)
  "name": "string", // the human-readable name of this country
  "code": "string" // the ISO-3166-1 2-character code
}
```

JSON-LD Context: https://api.chess.com/context/Country.jsonld
Example: https://api.chess.com/pub/country/IT

Country Players

Description: List of usernames for players who identify themselves as being in this country.
URL pattern: https://api.chess.com/pub/country/{iso}/players

Data Format:

```
{
  "players": [
    /* array of usernames for recently active players in this country */
  ]
}
```

Note: complete lists of all players are not available to download. By requesting this list once per day, you will obtain a list of all new registrants and all people who are currently active on Chess.com and identify themselves as being in (or associated with) this country. People who switch their country designation will be included in the next generation of the list.

Cache invalidation: This endpoints refresh at most once every 12 hours

JSON-LD Context: https://api.chess.com/context/CountryPlayers.jsonld
Example: https://api.chess.com/pub/country/IT/players

Country Clubs

Description: List of URLs for clubs identified as being in or associated with this country.
URL pattern: https://api.chess.com/pub/country/{iso}/clubs

Data Format:

```
{
  "clubs": [
    /* array of profile URLs for clubs in this country */
  ]
}
```

Note: the "country" of a club means different things to different clubs. In some cases, the clubs solicit membership from people in that country; in others, they attempt to meet for OTB chess and the country is where they do that; sometimes it indicates the language spoken; sometimes it's just the country of the founder and they haven't given it much thought.

JSON-LD Context: https://api.chess.com/context/CountryClubs.jsonld
Example: https://api.chess.com/pub/country/IT/clubs

Daily puzzle

Description: Information about the daily puzzle found in www.chess.com .
URL pattern: https://api.chess.com/pub/puzzle

Data Format:

```
{
   "title": "title", //the title of the daily puzzle
   "url": "URL", //url to daily puzzle in chess.com
   "publish_time": 1513584000, //the date of the published puzzle
   "fen": "FEN", //the FEN of the published puzzle
   "pgn": "PGN", //the PGN of the published puzzle, and yes, we know that thing you have noticed ;)
   "image":"the link to the image"
}
```

Note: if you are going to publish the Daily Puzzle somewhere please remember to give credits to Chess.com by means of a clearly visibile text link that points to the url of the puzzle page.

JSON-LD Context: https://api.chess.com/context/DailyPuzzle.jsonld
Example: https://api.chess.com/pub/puzzle

Random daily puzzle

Description: Information about a randomly picked daily puzzle.

URL pattern: https://api.chess.com/pub/puzzle/random

Data Format:

```
{
   "title": "title", //the title of the daily puzzle
   "url": "URL", //url to daily puzzle in chess.com
   "publish_time": 1513584000, //the date of the published puzzle
   "fen": "FEN", //the FEN of the published puzzle
   "pgn": "PGN", //the PGN of the published puzzle, and yes, we know that thing you have noticed ;)
   "image":"the link to the image"
}
```

Notes: the puzzle doesn't change every request but has some caching latency (around 15 seconds). If you are going to publish the Daily Puzzle somewhere please remember to give credits to Chess.com by means of a clearly visibile text link that points to the url of the puzzle page.

JSON-LD Context: https://api.chess.com/context/DailyPuzzle.jsonld

Example: https://api.chess.com/pub/puzzle/random

Streamers

Description: Information about Chess.com streamers.

URL pattern: https://api.chess.com/pub/streamers

Data Format:

```
{
   "streamers":[
      {
         "username": "string",
         "avatar": "URL",
         "twitch_url": "Twitch.tv URL",
         "url":"member url's"
      }
```

```
    ]
  }
```

Notes: the endpoint refreshes every 5 minutes.

Example: https://api.chess.com/pub/streamers

Leaderboards

Description: It displays information about top 50 player for daily and live games, tactics and lessons.

URL pattern: https://api.chess.com/pub/leaderboards

Data Format:

```
{
   "daily":[
      {
         "player_id": "integer",
         "@id": "URL",
         "url": "URL",
         "username": "string",
         "score": "integer",
         "rank": "integer" /* [1..50] */
      },
      [...]
   ],
   "daily960":[
      {
         "player_id": "integer",
         "@id": "URL",
         "url": "URL",
         "username": "string",
         "score": "integer",
         "rank": "integer" /* [1..50] */
      },
```

```
      [...]
   ],
   "live_rapid":[
      {
         "player_id": "integer",
         "@id": "URL",
         "url": "URL",
         "username": "string",
         "score": "integer",
         "rank": "integer" /* [1..50] */
      },
      [...]
   ],
   "live_blitz":[
      {
         "player_id": "integer",
         "@id": "URL",
         "url": "URL",
         "username": "string",
         "score": "integer",
         "rank": "integer" /* [1..50] */
      },
      [...]
   ],
   "live_bullet":[
      {
         "player_id": "integer",
         "@id": "URL",
         "url": "URL",
         "username": "string",
         "score": "integer",
         "rank": "integer" /* [1..50] */
      },
      [...]
```

```
    ],
    "live_bughouse":[
       {
          "player_id": "integer",
          "@id": "URL",
          "url": "URL",
          "username": "string",
          "score": "integer",
          "rank": "integer" /* [1..50] */
       },
       [...]
    ],
    "live_blitz960":[
       {
          "player_id": "integer",
          "@id": "URL",
          "url": "URL",
          "username": "string",
          "score": "integer",
          "rank": "integer" /* [1..50] */
       },
       [...]
    ],
    "live_threecheck":[
       {
          "player_id": "integer",
          "@id": "URL",
          "url": "URL",
          "username": "string",
          "score": "integer",
          "rank": "integer" /* [1..50] */
       },
       [...]
    ],
```

```
    "live_crazyhouse":[
       {
          "player_id": "integer",
          "@id": "URL",
          "url": "URL",
          "username": "string",
          "score": "integer",
          "rank": "integer" /* [1..50] */
       },
       [...]
    ],
    "live_kingofthehill":[
       {
          "player_id": "integer",
          "@id": "URL",
          "url": "URL",
          "username": "string",
          "score": "integer",
          "rank": "integer" /* [1..50] */
       },
       [...]
    ],
    "lessons":[
       {
          "player_id": "integer",
          "@id": "URL",
          "url": "URL",
          "username": "string",
          "score": "integer",
          "rank": "integer" /* [1..50] */
       },
       [...]
    ],
    "tactics":[
```

```
      {
         "player_id": "integer",
         "@id": "URL",
         "url": "URL",
         "username": "string",
         "score": "integer",
         "rank": "integer" /* [1..50] */
      },
      [...]
   ]
}
```

Notes: the endpoint refreshes when one of the leaderboards is updated.

Example: https://api.chess.com/pub/leaderboards

Game results codes

In the table below are listed all codes that are returned by game related endpoints.

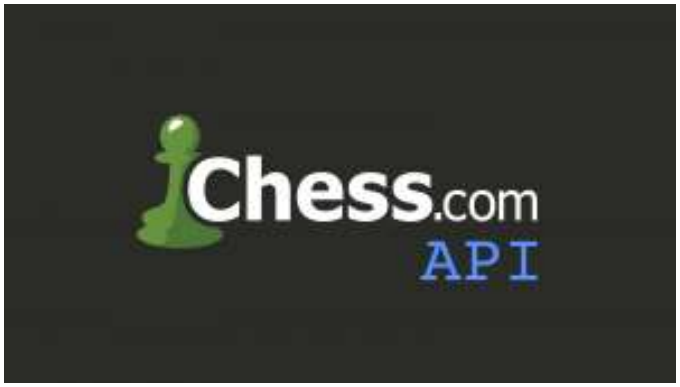
Code	Description
win	Win
checkmated	Checkmated
agreed	Draw agreed
repetition	Draw by repetition
timeout	Timeout
resigned	Resigned
stalemate	Stalemate
lose	Lose
insufficient	Insufficient material
50move	Draw by 50-move rule
abandoned	Abandoned

kingofthehill	Opponent king reached the hill
threecheck	Checked for the 3rd time
timevsinsufficient	Draw by timeout vs insufficient material
bughousepartnerlose	Bughouse partner lost

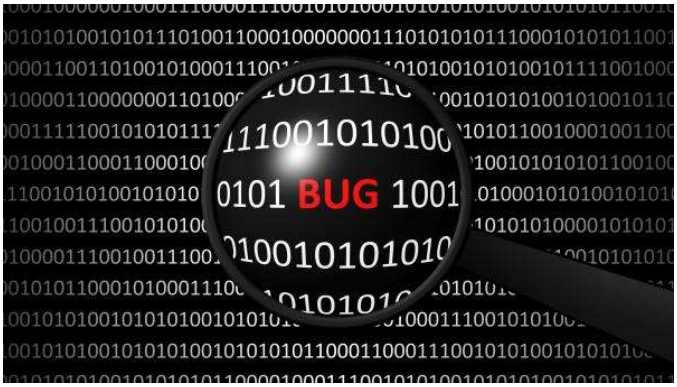
Please contribute bug reports, comments, and feature requests in the forums!

- Last reviewed and updated: 2022-05-10
- Comments: https://www.chess.com/news/view/published-data-api#comments
- Revisions: https://gist.github.com/andreij/0e3309200c0a6bb26308817a168203f3/revisions

More News



Breaking change: Player's clubs endpoint



Chess.com Bug Bounty Policy

Comments (180)

Sort: Newest

Ameya_Joshi14

3 days ago

0

Is there a way to access the puzzles rating? I am not able to find anything related to "rated puzzles" in the "documentation" if it is to be called so.

stephen_33

6 days ago

0

The failure to keep the API documentation up to date has been an *issue* for some time, not to mention the 'CHANGELOG' which was last updated 30/July/2018!

I may be biased but I think the API is a valuable resource that really needs proper investment.

* The impression of *neglect* isn't helped by the fact that the club owner (**@bcurtis**), hasn't been near the site since 2/April this year. He's usually my first port of call with questions and problems.

1 1

HighJebus

6 days ago

0

I don't know if it has been mentioned (just checked first pages), but some remarks:

- In the "Team Matches" section, inside each element of the "finished" array, "result" is not related to the "Game results codes", but to "win","draw" or "lose" (since these are results of a match, not a game).
- In the "Daily Team Match" section, some values are missing from this list:
  - Inside each team object:
    - The "fair_play_removals" array (empty) and the "locked" boolean. Both in "registration" phase.
    - The "url" string. In the other 2 phases.
    - The "result" string ("win", "draw" or "lose"). In "finished" phase.
  - The "end_time" number. In "finished" phase.
- The JSON-LD context for several endpoints is missing, the link header too.

stephen_33

11 days ago

0

I think what you're trying to achieve is way beyond anything this site offers. Since the API is in effect 'read only' I see no possible way you'd be able to accept a challenge even if an endpoint was provided.

That would involve the kind of interaction that the API just doesn't provide.

brownpanthera

11 days ago

0

> **stephen_33 wrote:**
>
> > **brownpanthera wrote:**
> >
> > > **brownpanthera wrote:**
>
> Not that I can think of in the case of one to one game challenges.
>
> In order to check match challenges are set up correctly in leagues I help run, I get around the problem by asking team admins to post the URL's of their challenges. Then I scrape the data I need to check from the

> challenge HTML document (they're downloadable). It works quite well but it's not ideal and any day now the site will change the challenge format and I'll have to tweak my code to deal with that.
>
> But you haven't described exactly what it is you're trying to do?

okay, it makes sense. I am trying to build a notification system. I play all my chess matches on my laptop, so I want to create something that allows me to accept challenges without opening chess dot com. I mean, even when I'm working or doing anything else, I want to receive notifications for new challenges. I think running the chess dot com in the background could achieve this, but I don't want to do that or just keep open chess dot com in the browser should also work but I don't want this also.

stephen_33

11 days ago

0

> **brownpanthera wrote:**
>
> > **brownpanthera wrote:**
> >
> > > **stephen_33**
> > > so is there any way to build something from scratch to get the data?

Not that I can think of in the case of one to one game challenges.

In order to check match challenges are set up correctly in leagues I help run, I get around the problem by asking team admins to post the URL's of their challenges. Then I scrape the data I need to check from the challenge HTML document (they're downloadable). It works quite well but it's not ideal and any day now the site will change the challenge format and I'll have to tweak my code to deal with that.

But you haven't described exactly what it is you're trying to do?

brownpanthera

11 days ago

0

> **brownpanthera wrote:**
>
> > **stephen_33**
> > so is there any way to build something from scratch to get the data?

stephen_33

12 days ago

1

There're no endpoints for challenges of any kind which is a pity because I could really use one for team match challenges but they don't exist either.

All endpoints are cached and are only intended for data that *persists* or is permanent. Challenges are strictly temporary which is why they're not included. Much the same applies to Live games I think.

brownpanthera

13 days ago 0

is there any way to get the challenge endpoint? I mean when a user gives a challenge to their friends or anyone?

ImperfectAge

Apr 16, 2023 0

There seems be a new country code in use - "XT" for Chinese Taipei - for example in this user's profile: **@gingernoobgod**. This code is not specified under Countries on this page: https://www.chess.com/announcements/view/published-data-api#pubapi-endpoint-country

chiopra

Mar 13, 2023 0

Is there no endpoint for players' solved puzzles? 😕

stephen_33

Mar 9, 2023 0

Endpoints for individual games? No there isn't. Nor is there any way to retrieve a game PGN given its web game_id - you have to do it the long way round, starting with a player's monthly archive.

brownpanthera

Mar 9, 2023 0

is there any way to get 1v1 data? I don't see any param for that

sanamirza348

Feb 22, 2023 0

evad90 wrote:

Small free app to have an overview of your chess matches.

https://chess-for-fun.vercel.app/ Hey I wanna connect with you may I get any resource

evad90

Feb 19, 2023 2

Small free app to have an overview of your chess matches.

https://chess-for-fun.vercel.app/

Zach-F

Feb 14, 2023

0

thanks

davideffe

Jan 31, 2023

1

Is there a function to retrieve my library collections games ?

Shades25

Dec 27, 2022

2

Love the API! Was curious if there was any update on the "Interactive API releasing later this year".

stephen_33

Dec 4, 2022

0

But the player id given in a player's endpoint is unique to this site, so isn't it just as useful as the uuid?

https://api.chess.com/pub/player/misterbrons

"player_id":18950690

Since we can't access data by means of the id alone, it's not of much use other than to confirm a change of name when we know both old and new names.

MisterBrons

Dec 4, 2022

0

Any tips on the most efficient way to get the uuid of a user? https://api.chess.com/pub/player/ doesn't expose the uuid. The Complete Monthly Archives does but it doesn't feel the most efficient way.

I want to work with uuid's to account for username changes.

Login or Join