ChatGPT

Implementing The Odds API in a Local Application

Overview of The Odds API

The Odds API is a third-party sports betting odds aggregator that provides real-time odds data from numerous major sportsbooks around the world. It **scrapes and compiles odds** for dozens of sports and leagues, covering **over 70 sports and 40+ bookmakers globally** [1] [2]. The API returns data in **JSON format** and supports both decimal and American odds formats [3] for easy integration into applications. Its coverage includes all popular betting markets – moneylines (head-to-head), point spreads, over/unders (totals), futures (outrights), and even an expanding selection of **player props and alternate lines** for certain sports [4].

Why use The Odds API? It is widely regarded as one of the best odds data providers for developers due to its **broad coverage**, **ease of use**, and **cost-effective plans** [5] [6]. The service has been operating since 2017 and is known for reliability [7]. Unlike building your own scraper for each bookmaker, The Odds API offers a single unified feed of odds from sources like DraftKings, FanDuel, BetMGM, Caesars, Unibet, Betfair, and many more across the US, UK, Europe, and Australia [2]. This saves developers significant effort in gathering and updating odds data. It also provides timely updates for **in-play (live)** games and upcoming matches [8], ensuring you can get near real-time line movements.

Getting Started: Account Setup and API Key

To use The Odds API, you first need to **sign up for an API key**. Visit the The Odds API website and create an account – upon subscribing (even to the free tier), you will receive your unique API key via email [9]. The service offers a **free starter plan** (great for development/testing) which provides **500 API requests per month** at no cost [5] [10]. This free tier includes all sports and most bookmakers, but excludes certain features like historical odds [11] [12]. For higher usage needs, paid plans start around $30/month for 20,000 monthly requests and scale up to millions of calls, with higher tiers unlocking **historical odds data** access [13] [10].

After obtaining your API key, you'll use it in each request to authenticate. The API expects the key as a query parameter `apiKey=YOUR_KEY` on all endpoint URLs. (Alternatively, some clients can send it as a header, but the simplest method is including it in the URL.) All requests are made to the base host `https://api.the-odds-api.com` [14]. For example, a basic request URL looks like:

```
https://api.the-odds-api.com/v4/sports?apiKey=YOUR_API_KEY
```

Once you have your key, you can start calling the various endpoints described below. In a local application, you would typically use an HTTP client (like `requests` in Python or `axios/fetch` in Node.js) to send GET requests to these endpoints and then parse the JSON response.

API Endpoints and Capabilities

The Odds API is a RESTful service with straightforward endpoints to retrieve sports, events, and odds data. Below are the key API capabilities and how to use each one:

- **List Available Sports:** The `/v4/sports` endpoint returns a list of all sports/leagues covered by the API, each with a unique sport key. This lets you discover the exact keys to use for subsequent odds queries. **Request:** `GET /v4/sports?apiKey=YOUR_KEY`. By default it returns only sports that are currently "active" (in-season), but you can include `all=true` to get both in-season and out-of-season sports [15]. This endpoint is free to call and **does not consume your request quota** [16] [17]. Each sport object includes fields like `key` (e.g. `"soccer_epl"`), sport group, league title, description, whether it's active, and if outright markets are available [18] [19]. For example, a portion of the response looks like:

```
[
  {
    "key": "americanfootball_nfl",
    "group": "American Football",
    "title": "NFL",
    "description": "US Football",
    "active": true,
    "has_outrights": false
  },
  {
    "key": "americanfootball_nfl_super_bowl_winner",
    "group": "American Football",
    "title": "NFL Super Bowl Winner",
    "description": "Super Bowl Winner 2024",
    "active": true,
    "has_outrights": true
  },
  ...
]
```

In this example, `"americanfootball_nfl"` is the sport key for NFL games, whereas `"americanfootball_nfl_super_bowl_winner"` is a special sport key for the NFL futures (outright winner market) [19] [20]. You would use these keys in the following endpoints.

- **List Upcoming Events (Games):** To get a schedule of upcoming games (and any games currently in-play) for a given sport, use the `/v4/sports/{sport}/events` endpoint. This returns a list of events with basic info: each event's unique `id`, the `home_team` and `away_team` names, and the `commence_time` (start time) [21] [22]. **Request:** `GET /v4/sports/{sport_key}/events?apiKey=YOUR_KEY`. For example, to list upcoming NFL games:

```
GET https://api.the-odds-api.com/v4/sports/americanfootball_nfl/events?
apiKey=YOUR_KEY
```

This would return an array of NFL games with their IDs and teams [22]. Odds are **not** included in this particular response (just the match info) [21] [23]. Like the sports list, calling the events endpoint does **not consume quota** (free to call) [21] [23]. It's useful for caching event IDs or building a schedule without using up your request credits. You can filter this endpoint by date range if needed (there are optional `commenceTimeFrom` and `commenceTimeTo` parameters to restrict to events in a certain time window) [24] [25]. The special sport key `"upcoming"` can also be used here (and in odds queries) – using `/sports/upcoming/events` will return a mixed list of any live games plus the next 8 upcoming games across *all* sports [26].

- **Retrieve Odds for Events:** This is the core feature – getting the odds from various bookmakers for upcoming or live games. Use the `/v4/sports/{sport}/odds` endpoint to fetch odds for all events in a given sport (optionally filtered by market or bookmaker). **Request:** `GET /v4/sports/{sport_key}/odds?apiKey=YOUR_KEY&regions={regions}&markets={markets}&oddsFormat={format}` (plus other optional filters). For example:

```
GET https://api.the-odds-api.com/v4/sports/americanfootball_nfl/odds?
apiKey=YOUR_KEY&regions=us&markets=h2h,spreads&oddsFormat=american
```

This would retrieve the moneyline (`h2h`) and point spread odds for each upcoming NFL game, from **bookmakers in the US region** (like DraftKings, FanDuel, Caesars, etc.), with odds in American format [27] [28]. You can specify one or multiple regions – available region codes include `us` (United States), `us2` (an alternate set of US books), `uk` (United Kingdom), `eu` (Europe), `au` (Australia), etc. [28]. The `markets` parameter lets you choose which betting markets to include; by default it returns only `h2h` (moneyline) if not specified [29]. You can request multiple markets by comma-separating them (e.g. `markets=h2h,spreads,totals`). Supported market keys include:

- `h2h` – head-to-head moneyline odds (default)
- `spreads` – point spread odds (with point values and odds)
- `totals` – over/under totals
- `outrights` – futures (championship winner odds, etc., for sports that have `has_outrights=true`)

Each additional market you request will add to the data volume **and** the request cost, so only ask for what you need. (Specifically, each market *per region* counts as 1 credit towards your monthly quota [29] [30]. For instance, querying `h2h,spreads` in one region counts as 2 requests from your quota in total.) You can also filter to specific bookmakers using the `bookmakers` parameter (a comma-separated list of bookie keys), which overrides the region setting if used [31]. This is useful if you only care about odds from, say, a specific sportsbook.

Response: The odds endpoint returns an array of game objects, each containing the `id`, teams, start time, and a list of bookmakers with their odds. Each bookmaker entry has its own `key` (identifier like `"draftkings"`), a `title` (human name, e.g. "DraftKings"), a timestamp of last odds update, and a list of `markets` with odds. For example, a single NFL game in the response might look like:

```
{
  "id": "bda33adca828c09dc3cac3a856aef176",
  "sport_key": "americanfootball_nfl",
  "commence_time": "2025-12-01T18:00:00Z",
  "home_team": "Tampa Bay Buccaneers",
  "away_team": "Dallas Cowboys",
  "bookmakers": [
    {
      "key": "draftkings",
      "title": "DraftKings",
      "last_update": "2025-12-01T14:30:00Z",
      "markets": [
        {
          "key": "h2h",
          "outcomes": [
            { "name": "Dallas Cowboys", "price": +240 },
            { "name": "Tampa Bay Buccaneers", "price": -303 }
          ]
        },
        {
          "key": "spreads",
          "outcomes": [
            { "name": "Dallas Cowboys", "point": 6.5, "price": -110 },
            { "name": "Tampa Bay Buccaneers", "point": -6.5, "price": -110 }
          ]
        }
      ]
    },
    {
      "key": "fanduel",
      "title": "FanDuel",
      "last_update": "2025-12-01T14:31:00Z",
      "markets": [ ... ]
    },
    ...
  ]
}
```

In the above (illustrative) snippet, for the moneyline (`h2h`) market DraftKings offers +240 on Dallas and -303 on Tampa Bay [32] [33], and for the point spread (`spreads`) both teams are at -110 with Dallas +6.5 points and Tampa -6.5 [34] [35]. The API will include one object per bookmaker that has odds available. Note that for betting exchanges like Betfair, the API even provides separate lay odds markets (denoted by keys like `h2h_lay`) automatically alongside the back odds [36].

You can request odds in either `decimal` or `american` format using the `oddsFormat` parameter (defaults to decimal if not specified) [37]. There are also optional flags such as `dateFormat` (ISO 8601 vs Unix timestamp for times) and flags to include extra info: `includeLinks=true` will provide direct affiliate/deeplinks to the bookmaker's event page or betslip, if available [38]; `includeSids=true` will include the source/bookmaker's internal IDs for teams and markets (useful if you need to cross-reference or build your own links) [39]; and `includeBetLimits=true` can show betting limits for exchange markets [40]. In most simple use cases, you won't need those, but they're there for advanced integrations.

Usage Cost: Unlike the sports or events list, the odds endpoint **does consume your usage credits**. As mentioned, the cost is proportional to the number of markets and regions you request. For example, fetching 1 market from 1 region costs 1 credit, while 3 markets from 2 regions would cost 3×2 = 6 credits [41] [42]. The API responses include helpful headers `x-requests-remaining` and `x-requests-used` so you can track your quota usage in real-time [43]. Also note that if your query returns no data (e.g., no events found), it **does not deduct** any credits [44] [45]. The free plan's 500 credits can go quickly if you request many markets frequently, so plan accordingly (you might fetch only the essential markets, or only call the API at certain intervals). If you hit the rate limit (HTTP 429), the guidance is simply to slow down – space out your requests a bit more [46] (The Odds API protects against bursts, but normal usage with modest pacing shouldn't be an issue).

- **Odds for a Specific Event:** If you need **more detailed odds for one particular game** (for example, to get a wide array of prop bets or alternate lines for that game), The Odds API provides a `/v4/sports/{sport}/events/{eventId}/odds` endpoint. This returns odds for a single specified event (game) and allows **any available market** to be requested via the `markets` parameter [47] [48]. In contrast, the main `/odds` endpoint is limited to the major markets (h2h, spreads, totals, outrights) for efficiency. The event-specific odds endpoint is useful if you want something like player prop odds, alternate spreads/totals, or period-by-period lines for one game. **Request:** `GET /v4/sports/{sport_key}/events/{eventId}/odds?apiKey=YOUR_KEY&regions={regions}&markets={markets}&oddsFormat={format}`. The required `eventId` is the game's ID from the earlier **events list** endpoint [48] [49]. You can list multiple market keys in `markets` (including exotic ones like `"player_pass_tds"` for player touchdowns, etc., as long as the bookmaker offers them) [47] [50]. The output structure is similar to the normal odds response but scoped to one game. The documentation suggests using this endpoint only when necessary, since it can return a lot of data (and thus is restricted to one event at a time) [51] [52]. **Cost:** The usage cost here is also based on markets × regions (same rate of 1 credit per region-market) [53] [42]. For most common needs (moneyline/spread/total), the main multi-event odds endpoint is more efficient, but event-specific odds let you dig into all markets on a single game on demand.

- **Listing Available Markets for an Event:** Not sure which market keys exist for a given game? There's a handy endpoint `/v4/sports/{sport}/events/{eventId}/markets` that returns just the list of betting markets offered for that event (per bookmaker). This can be useful to programmatically discover what prop or alternate markets are available before deciding to fetch odds. **Request:** `GET /v4/sports/{sport_key}/events/{eventId}/markets?apiKey=YOUR_KEY&regions={regions}` (you can filter by region or specific bookmakers similarly to the odds endpoints) [54] [55]. **Response:** It returns a JSON object with the event's basic info (id, teams, time) and each bookmaker, but instead of odds, each bookmaker entry contains a list of market keys and the last update time for each market [56] [57]. For example, for a given MLB baseball game, the response might show that FanDuel offers markets like `"alternate_spreads"`, `"batter_home_runs"`, `"first_inning_total"` etc., each with a last_update timestamp [57] [58] – indicating those markets have odds available via the event odds endpoint. This endpoint costs 1 credit per call [59], but it's lighter than pulling all odds. You might use it to dynamically display a menu of available bet types for a selected game in your app.

- **List Teams/Participants:** The API also lets you fetch a list of **participants** (teams or players) for a given sport using `/v4/sports/{sport}/participants`. For team sports, this returns all the team names and an associated participant `id` for each [60] [61]. For individual sports like tennis, it would list player names. This can help if you need IDs to map team names or just want to verify the official names as used by the API. **Request:** `GET /v4/sports/{sport_key}/participants?apiKey=YOUR_KEY`. For example, `/v4/sports/americanfootball_nfl/participants` would return the full list of NFL team names with their IDs [61] [62]. This endpoint likely does **not consume quota** (the docs don't list a cost, implying it's treated as reference data). Note that it doesn't list individual players on each team – it's only the top-level participants (teams or competitors) in the league [63].

- **Historical Odds Data:** One standout feature (for paid plans) is access to historical odds. The Odds API can provide **snapshots of odds at past timestamps**, which is valuable for analysis of line movement or for modeling. The historical endpoints require a paid subscription and use a similar structure under `/v4/historical/`. Key historical endpoints include:

- **Historical Odds Snapshot:** `GET /v4/historical/sports/{sport}/odds?date={timestamp}...` – Returns the odds **snapshot at a specific date/time** for all games in a sport [64] [65]. You provide a `date` (in ISO8601 format, UTC) and the API returns the closest available snapshot at or before that time [66]. The data payload mirrors the live `/odds` response (list of games with odds), but it's wrapped with metadata like the exact snapshot timestamp and links to previous/next snapshot times [67] [68]. Historical data is recorded at regular intervals (every 5 or 10 minutes) and available going back to mid-2020 [69]. This is great for retrieving closing lines (by using a timestamp just before game start) or tracking how odds changed over time. **Cost:** Because of the large data involved, historical odds calls cost **10 credits per region per market** [70] [71] – so they are ten times the cost of a live odds request.

- **Historical Events:** `GET /v4/historical/sports/{sport}/events?date={timestamp}...` – Similar to the live events endpoint but for the past. It lists what events were present at the given timestamp (with their IDs, teams, and start times) [72] [73]. This is useful to find an event's ID from a past date so you can query its odds. For example, if you want the odds for last year's Super Bowl, you'd first call historical events around that date to get the event ID, then use the historical event odds endpoint. **Cost:** ~1 credit per call (and free if it returns no data) [74] [75].

- **Historical Odds for a Specific Event:** `GET /v4/historical/sports/{sport}/events/{eventId}/odds?date={timestamp}&markets=...` – This gives the odds for one event at a specified historical time [76] [77]. It's the historical analog of the event odds endpoint, allowing any market. For example, you could pull all player prop closing odds for a specific game by using the game's ID and the timestamp of game start. The cost is similar to historical odds – about 10 credits per region/market (the documentation notes 1 market 1 region = 10 credits) [78]. These historical endpoints are only available on paid plans [69] [74], so on the free tier you won't be able to use them.

(The Odds API also offers a separate Scores & Results API for final scores, and even a premade Odds Widget for websites [79], but those are outside the core odds API and might require additional subscription. For fetching game results, you would either use their scores API or an alternative data source.)

Using the API in Your Application

To implement The Odds API locally, you'll integrate by making HTTP GET requests to these endpoints and processing the JSON data. Here's a general step-by-step approach:

1. **Choose your target data and endpoint:** Determine what you need – e.g., upcoming odds for all NBA games (use `/sports/basketball_nba/odds`), or maybe just one game's props (use `/events/{id}/odds`), etc. Use the **sport key** from the sports list for the league you're interested in [80] [81]. For broad coverage of live/upcoming games across sports, you can even use the `upcoming` pseudo-sport to get a snapshot of a few upcoming games in all sports at once [26].

2. **Build the request URL with parameters:** Start with the base URL `https://api.the-odds-api.com/v4/` + the endpoint path. Include your `apiKey` in the query string. Add required params like `regions` (e.g. `regions=us` for U.S. books) and specify any optional ones like `markets`, `oddsFormat`, etc. For example, to get English Premier League soccer odds in decimal format you might do:

   ```
   GET https://api.the-odds-api.com/v4/sports/soccer_epl/odds?
   apiKey=YOUR_KEY&regions=uk&markets=h2h,totals&oddsFormat=decimal
   ```

   This would return moneyline and over/under odds from UK bookmakers for all upcoming EPL matches. The documentation provides example request formats for each endpoint (as well as a Postman collection) to help you form the URLs [82] [83].

3. **Make the HTTP request:** Using your programming language of choice, send a GET request to the URL. In Python you might use the `requests` library, in JavaScript fetch or Axios, etc. No special authentication headers are needed beyond the API key parameter. The Odds API servers will respond with a JSON payload. Ensure you handle potential HTTP errors – e.g., a 401/403 if your API key is invalid, or 429 if you hit rate limits (in which case just wait and retry more slowly) [84] [46].

4. **Parse the JSON response:** The response will be JSON data. For example, if you requested odds, you will get an array of events as shown earlier, which you can loop through in code. Extract the pieces you need – e.g., event names (team names), odds values, etc. The structure is mostly self-explanatory (bookmakers -> markets -> outcomes). Be mindful that some fields like `price` (odds) might be integers or floats depending on format (American odds can be integer like -110, decimal odds might be 1.91 etc.). If you requested American odds, positive values are underdogs and negative are favorites as usual.

5. **Use the data in your app:** With the parsed data, you can display odds on your interface, store them in a database, run calculations (like finding the best odds across bookies for arbitrage, etc.), or trigger alerts when lines move. The API's update frequency is quite high – odds are updated **every**

few minutes or even seconds for in-play markets (the exact interval can vary by sport and book; The Odds API notes that snapshots are as frequent as 5 minutes apart for historical data and real-time feed is continuously updated [85] [86]). The included `last_update` timestamp for each bookmaker tells you how recent that line is [87] [88].

6. **Monitor usage:** Check the HTTP headers in each response – `x-requests-remaining` will tell you how many calls you have left in your current monthly quota [43]. This is especially important on the free plan. If you approach the limit, you might want to throttle your requests or upgrade the plan. Also, design your app to avoid unnecessary calls (for instance, you might cache the list of sports which doesn't change often, or use the `eventIds` filter to fetch odds for only games of interest instead of all games) [89]. The API also doesn't charge for empty responses, so filtering by IDs or time range can be efficient if only some games matter to you [44] [45].

In practice, using The Odds API is very straightforward. For example, in Python you could do something like:

```python
import requests
url = "https://api.the-odds-api.com/v4/sports/basketball_nba/odds"
params = {
    "apiKey": "YOUR_API_KEY",
    "regions": "us",
    "markets": "h2h,spreads",
    "oddsFormat": "decimal"
}
response = requests.get(url, params=params)
odds_data = response.json()  # parse JSON into Python data structures
```

And that's it – `odds_data` would then be a list of NBA games with odds from US bookmakers. The Odds API provides official code samples for Python and Node.js as well [90] [91], but the above is essentially all that's needed to connect.

Considerations and Other Providers

The Odds API is a robust choice for odds data, especially given its free tier and wide coverage [5] [6]. If you require features beyond its scope, there are other odds data providers to consider. For instance, **SportsDataIO** offers an odds API as part of a broader sports data suite (including scores, stats, etc.) [92] [93], and it has a free trial (though the free trial data may be partially scrambled) [94]. **OddsJam** and **Unabated** are other services known for real-time odds and tools (e.g. for arbitrage or positive EV betting), but they tend to be enterprise-level (OddsJam has no free tier, Unabated starts around $500/month) [95] [96]. **SportRadar** and **Betradar** have odds feeds too, usually aimed at large businesses. Each provider has different pricing and integration complexity – for a solo developer or small project, The Odds API is often the most accessible and cost-effective choice. It provides the essential live odds, futures, and (with paid plans) historical line data needed for most applications [97] [98].

In summary, integrating The Odds API into your local application involves: getting your API key, calling the relevant endpoints for the sports and markets you need, and then handling the JSON results. The API is well-documented and developer-friendly [99], making it straightforward to get everything from current game lines to future odds in your app. With this in place, you can build features like odds comparison across sportsbooks, monitoring line movements, or powering betting dashboards – all without maintaining your own scrapers. It's simply about leveraging the right endpoint for the data you want, and The Odds API has you covered for most odds-related needs. Good luck with your implementation!

Sources: The Odds API official documentation and website [100] [28] [27] [21], and a comparison of odds API providers [5] [6] which highlights The Odds API's features and free tier.

---

[1] [2] [3] [4] [7] [8] [9] [11] [12] [79] [99] [100] Sports Odds API | The Odds API
https://the-odds-api.com/

[5] [6] [10] [13] [92] [93] [95] [96] [97] [98] Comparing the Best Odds API Providers - Sports Game Odds - Odds API and Data
https://sportsgameodds.com/comparing-odds-api-providers/

[14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [67] [68] [69] [70] [71] [72] [73] [74] [75] [76] [77] [78] [80] [81] [84] [85] [86] [87] [88] [89] [90] Odds API Documentation V4 | The Odds API
https://the-odds-api.com/liveapi/guides/v4/

[82] [83] The Odds API | Documentation | Postman API Network
https://www.postman.com/odds-api/the-odds-api-workspace/documentation/my4qrii/the-odds-api

[91] Odds API Code Samples V4 | The Odds API
https://the-odds-api.com/liveapi/guides/v4/samples.html

[94] Frequently Asked Questions - SportsDataIO
https://sportsdata.io/developers/faq