



API REFERENCE | PUBLIC

Live in-play pricing

A client guide for discovering available matches, authenticating a WebSocket, subscribing to one or more matches, and receiving live prices.

Public contract

This guide defines match discovery, default markets, and WebSocket subscription for live tennis pricing.

Version 1.0 | Public contract

01

What this guide covers

Use the in-play API to find available matches and use the returned match ID when subscribing. You only work with the public match information and pricing events needed for integration.

Your boundary

The endpoint names and payloads in this document define your public contract. Keep the API key private.

The client flow is:

```
GET live/upcoming matches
or
GET matches for a UTC date
  |
  v
User selects a match and receives its match ID
  |
  v
User opens the WebSocket
  |
  v
User authenticates with the API key
  |
  v
User subscribes using that match ID
  |
  v
User receives a market snapshot and price updates
```

02

Discover matches

All discovery requests use the API key in the X-API-Key header. Dates are interpreted as UTC calendar dates.

Method	Path	Purpose
GET	/v2/inplay/matches?status=live	Matches currently live
GET	/v2/inplay/matches?status=upcoming	Upcoming matches
GET	/v2/inplay/matches?date=YYYY-MM-DD	Upcoming, live, completed, and cancelled matches for a UTC date

```
curl --request GET 'https://marketengine.statsmatics.com/v2/inplay/matches?status=live' \
--header 'X-API-Key: tpa_live_your_key'
```

The response lists the matches available to you. It includes the match ID and the display information needed to choose the correct match.

```
{
  "matches": [{
    "match_id": "194136",
    "players": {
      "p1": { "name": "Rigete Te" },
      "p2": { "name": "Bernard Tomic" }
    },
    "status": "live", "gender": "men",
    "draw": "singles", "format": "B03",
    "surface": "hard", "tournament": "Hangzhou",
    "scheduled_time": "2026-09-22T09:00:00.000Z"
  }]
}
```

Match ID

Use the match_id returned by discovery when you subscribe.

03

Choose a match

Choose a live or upcoming singles match by player names, tournament, and date. Retain the returned match_id, for example 194136.

Field	Client use
match_id	Use this exact string in the WebSocket subscribe frame.
players.p1 / players.p2	Display the player names.
status / draw	Choose a live or upcoming singles match.
format / gender	Select the correct client workflow.
surface / tournament	Display match context.

04

Connect and authenticate

Open one WebSocket connection. A single connection can hold multiple match subscriptions.

```
wss://marketengine.statsmatics.com/v2/inplay/ws
```

Send the API key as the first JSON frame:

```
{
  "type": "authenticate",
  "request_id": "auth-1",
  "api_key": "<api-key>"
}
```

```
{
  "protocol": "live.v1",
  "type": "ack",
  "request_id": "auth-1",
  "status": "authenticated"
}
```

API key handling

Keep the API key private. Never put it in a URL, browser source, WebSocket subprotocol, or log.

Authentication is separate from selecting a match. First choose a match, then authenticate, then subscribe.

05

Subscribe and receive prices

The default subscription returns MONEYLINE, HANDICAP, and TOTAL markets together: match Moneyline, applicable set Moneylines, seven Handicap lines, and seven Total lines. The main Handicap and Total lines are closest to 50/50, with three whole-game steps on each side. One priced state produces one markets array.

```
{
  "type": "subscribe",
  "match_id": "194136"
}
```

```
{
  "protocol": "live.v1",
  "type": "ack",
  "match_id": "194136",
  "status": "pending"
}
```

The acknowledgement can be pending while the first price is prepared. The first market-bearing message is a snapshot. This excerpt shows one line of each market type; the full markets array also includes six more Handicap lines, six more Total lines, and applicable set Moneylines:

```
{
  "protocol": "live.v1",
  "type": "snapshot",
  "match_id": "194136",
  "players": [
    {"position": "PLAYER_1", "name": "Rigele Te"},
    {"position": "PLAYER_2", "name": "Bernard Tomic"}
  ],
  "status": "live",
  "calibration_status": "SOLVED",
  "price_timestamp_utc": "2026-09-22T10:15:30.123Z",
  "markets": [
    {
      "type": "MONEYLINE", "period": "MATCH",
      "selections": [
        {"selection": "PLAYER_1", "line": null, "name": "Rigele Te", "model_probability": 0.413, "decimal_price": 2.421},
        {"selection": "PLAYER_2", "line": null, "name": "Bernard Tomic", "model_probability": 0.587, "decimal_price": 1.704}
      ]
    },
    {
      "type": "HANDICAP", "period": "MATCH",
      "selections": [
        {"selection": "PLAYER_1", "line": -2.5, "name": "Rigele Te", "model_probability": 0.524, "decimal_price": 1.908},
        {"selection": "PLAYER_2", "line": 2.5, "name": "Bernard Tomic", "model_probability": 0.476, "decimal_price": 2.101}
      ]
    },
    {
      "type": "TOTAL", "period": "MATCH",
      "selections": [
        {"selection": "OVER", "line": 22.5, "name": "Over 22.5", "model_probability": 0.548, "decimal_price": 1.825},
        {"selection": "UNDER", "line": 22.5, "name": "Under 22.5", "model_probability": 0.452, "decimal_price": 2.212}
      ]
    }
  ]
}
```

Future-set Moneylines are conditional on that set being played. Completed or impossible sets are not priced.

Market updates and connection recovery

When the priced state changes, the service sends one prices message containing the complete markets array. The excerpt below shows representative entries from that one frame.

```
{
  "protocol": "live.v1",
  "type": "prices",
  "match_id": "194136",
  "status": "live",
  "calibration_status": "SOLVED",
  "price_timestamp_utc": "2026-09-22T10:15:36.847Z",
  "markets": [
    {
      "type": "MONEYLINE", "period": "MATCH",
      "selections": [
        {"selection": "PLAYER_1", "line": null, "name": "Rigele Te", "model_probability": 0.438, "decimal_price": 2.283},
        {"selection": "PLAYER_2", "line": null, "name": "Bernard Tomic", "model_probability": 0.562, "decimal_price": 1.779}
      ]
    },
    {
      "type": "HANDICAP", "period": "MATCH",
      "selections": [
        {"selection": "PLAYER_1", "line": -2.5, "name": "Rigele Te", "model_probability": 0.547, "decimal_price": 1.828},
        {"selection": "PLAYER_2", "line": 2.5, "name": "Bernard Tomic", "model_probability": 0.453, "decimal_price": 2.208}
      ]
    },
    {
      "type": "TOTAL", "period": "MATCH",
      "selections": [
        {"selection": "OVER", "line": 22.5, "name": "Over 22.5", "model_probability": 0.571, "decimal_price": 1.751},
        {"selection": "UNDER", "line": 22.5, "name": "Under 22.5", "model_probability": 0.429, "decimal_price": 2.331}
      ]
    }
  ]
}
```

Multiple matches

Authenticate once, then send one subscribe frame per match. Each frame needs only the match ID returned by discovery. Route incoming prices by match_id.

```
{"type": "subscribe", "match_id": "194136"}
{"type": "subscribe", "match_id": "194137"}
{"type": "subscribe", "match_id": "194138"}
```

If the connection closes, use this simple flow:

1. Reconnect to the WebSocket.
2. Authenticate with your API key.
3. Send the same subscription requests again.
4. Continue from the fresh match snapshot.

Condition	Client action
Invalid or revoked key	Obtain a valid API key and authenticate again.
Match unavailable	Refresh the match list and choose an available match.
Service temporarily unavailable	Wait and continue when the service sends the next current price.
Match completed	Stop consuming prices for that match and retain the final update.
Socket closes	Reconnect, authenticate, and send the same match subscriptions again.

Recovery is handled by the service

The service handles feed recovery, state management, and the latest snapshot. You only reconnect the WebSocket and send the same match subscriptions again.

07

Integration checklist

- Use your API key.
- Call the live, upcoming, or UTC-date match endpoint before subscribing.
- Show the match ID and player names so the selection is explicit.
- Use the `match_id` returned by discovery; do not invent or transform it.
- Connect to the WebSocket and authenticate before sending subscription commands.
- Send one subscribe frame per selected match. Each frame needs only type and `match_id`.
- Use `match_id` in incoming events to route updates for multiple matches.
- If the connection closes, reconnect, authenticate, and send the same match subscriptions again.
- Stop at completed and keep the final update for your records.

Service behavior

Feed recovery and live connection state are handled by the service. You consume current snapshots, prices, statuses, and completion updates.

For support, provide `match_id`, event type, HTTP status when applicable, and UTC timestamp. Never send the API key.