BROADCASTERS

A score bug and a live feed for your stream

Put a live pickleball score on OBS or vMix with a score bug page and a data feed. The organizer gives you a key and two addresses; here is what to do with them.

Updated 24 October 2026 3 min read

What you get

The organizer makes you a **feed key**. You get two addresses:

- the **score bug page**, for OBS;
- the **data address**, for vMix and anything that reads a web address.

A key reads scores and nothing else. It ends on a date the organizer chose, and they can take it back at any time. Treat the address like a password for that event's scores.

A key can be for **one court**, which always shows whatever match is on it; for **one match**, pinned; or for the **whole event**.

OBS: the score bug page

1. Add a **Browser** source.
2. Paste the **score bug page** address.
3. Width 480, height 400 is plenty. The background is already transparent.

The serving side has a badge with the server's number: **1st** or **2nd** server in doubles. The first serve of a game reads **Opening serve**. The rallyxrally mark sits in the footer.

Options go after the key in the address: `?hold=30` keeps a finished match up for 30 seconds (60 if not said), and `?scale=1.5` makes it bigger. It shows nothing while nothing is on the court, and it never shows a match that was typed in after play.

vMix, and anything that reads a web address

Use the **data address** as a JSON data source and refresh it every **1 second**. Faster is wasted, because answers are cached for 2 seconds.

```
{
  "v": 1,
  "asOf": "2026-10-24T10:15:02.000Z",
  "scope": "court",
  "target": "Court 1",
  "match": {
    "category": "Men's Open",
    "court": "Court 1",
    "status": "live",
    "game": 2,
    "gameTo": 11,
    "call": "5-3-2",
    "a": { "name": "Side A", "points": 5, "games": 1 },
    "b": { "name": "Side B", "points": 3, "games": 0 },
    "serving": "a",
    "server": 2,
    "serve": "2nd",
    "played": [{ "a": 11, "b": 8 }],
    "winner": null,
    "capture": "live",
    "lastPointAt": "2026-10-24T10:15:00.000Z"
  },
  "matches": []
}
```


- `match` is set for a **court** or **match** key, and is null while nothing is on that court. An **event** key leaves it null and fills `matches` with every match in play, newest first.
- `status` is `ready` (started, nothing played), `live` or `complete`. `call` is the score as the referee calls it: serving side, receiving side, server number. It is null once the match is decided.
- `serving` is the side serving, `a` or `b`. `server` is 1 or 2 in side-out doubles. `serve` says it in words: `opening`, `1st` or `2nd`. **`opening` is the first serve of a game.** That team starts as the second server, so `server` is 2 at 0–0, and it is not a second chance. Use `serve` and do not work it out from `server`. All three are null when there is no second server, or the match is decided.
- `a` and `b` read **Side A** and **Side B** unless the organizer chose to show names.
- `capture` says how the match was recorded. **`entered` means it was typed in after play**, so do not show it as live. The score bug page already refuses to.
- `v` is the shape of the data. A different shape means a new `v` at a new address, never a change to this one.
- A key that does not work (wrong, ended or revoked) gets `401`, without saying which.

Scoreboards

Controllers such as Daktronics take data over a serial port or a UDP socket on the venue network, which a web service cannot reach. That needs a small program at the venue that reads the data address and writes what the controller expects. It is not built yet. If your venue has one, tell the organizer which controller it is.
