

Put an email signup form inside a tweet.

How X player cards let people subscribe without leaving the post, and a copy-paste prompt your AI coding agent can use to build one for your newsletter.

**Andrew Warner** @AndrewWarner

Want the guide? Drop your email right here 📧

FREE PDF

Get the guide.

Send it

games.codeshiftagent.com

By **Andrew Warner**

How a form gets inside a post

When you paste a link into a post, X's crawler (`Twitterbot`) reads the page's meta tags. If they declare a *player card*, X can show an HTTPS page of yours in an iframe inside the post. Usually that's a video player. It can just as well be a signup form.

The tags on your share page

<code>twitter:card</code>	<code>player</code>
<code>twitter:player</code>	HTTPS URL of the form page that loads inside the iframe
<code>twitter:player:width / height</code>	Iframe size in pixels, e.g. 640 × 480
<code>twitter:image</code>	Preview shown before the form loads. Make it a picture of the form.
<code>twitter:title / description</code>	Text shown with the card: say what they get
<code>og:*</code>	Open Graph equivalents for other sites

Where this came from

Pieter Levels (@levelsio) booted Windows XP inside a post this way (his player URL is `pieter.com/?embed=1` at 640×480). I copied the technique for a tiny platformer, *Pip's Run*, which played inside my post. That proved it works on our domain. The obvious next step was the useful version: a form that grows a newsletter.

Three URLs

```
https://example.com/join/x      # paste this in the post (tags only)
https://example.com/join/play   # the form X loads in the iframe
https://example.com/join/card.png # preview image
```

X's tags use `name="twitter:..."` ; Open Graph uses `property="og:..."` . The tags must be in the raw HTML, not added by JavaScript.

Making the form work in the iframe

02

A form inside someone else's page has quirks. These are the ones that mattered for us.

1. It loads on tap

Readers see your preview image first. One tap loads the form. So the image should look like the form, with the promise in big type.

2. Taps work. Keyboard focus needs help.

In our game, keystrokes stayed with X's page instead of reaching the iframe, while taps and clicks always got through. For a form that means:

- On `pointerdown`, `touchstart` and `click` on the input, call `window.focus()` and then `input.focus()`.
- Don't call `preventDefault()` on pointer events. That can stop focus moving into the iframe.
- Use `type="email"`, `inputmode="email"` and `autocomplete="email"`, so phones show the right keyboard and offer autofill.
- Font size of 16px or more on the input, so iPhones don't zoom in.

3. Always have a fallback

Some apps or browsers may not let people type in an embedded frame. Add a small link, "Open the form in a new tab", that calls `window.open()` on your form page. Nobody gets stuck.

4. Fit 640×480 and phones

One headline, one line of copy, one field, one button. Center it, and let the button wrap under the input on narrow screens.

5. Don't block framing

Serve over HTTPS. Don't send `X-Frame-Options` or a restrictive CSP `frame-ancestors`. An allowlist of only x.com and twitter.com can break if X frames the page from another origin.

Connecting it to your newsletter

The form posts to your own server, which talks to your email provider. Here beehiiv is the worked example; any provider with a "create subscriber" API works the same way.

```
POST /join/api/subscribe          # your Worker
1. reject if honeypot filled or email invalid
2. rate-limit by IP
3. POST https://api.beehiiv.com/v2/publications/{PUB_ID}/subscriptions
   Authorization: Bearer {BEEHIIV_API_KEY}
   { "email": "...", "reactivate_existing": false,
     "send_welcome_email": true, "utm_source": "x",
     "utm_medium": "tweet-card", "referring_site": "x.com" }
4. 2xx → { ok: true } → thank-you screen
```

Keep the key on the server

Store the API key and publication ID as Worker secrets. The page never sees them. If they're missing, return a friendly "not open yet" error and log it. Never fake a success.

Basic abuse protection

- **Honeypot:** a hidden field humans never fill. Bots often do.
- **Rate limit:** a few attempts per IP per few minutes.
- **Validate** the email server-side, not just in the browser.

Deliver the lead magnet on the spot

Right after a successful signup, show a thank-you screen inside the post and start the PDF download automatically, with a download button as a fallback. No email setup required.

Caching and the domain caveat

X caches card data for a while (guides commonly cite up to about 7 days). Change the title or image? Share the URL with a new query like `?v=2`. Changes to the form page itself show up right away.

X historically required approval for player-card domains, and its Card Validator no longer shows previews. Our new subdomain rendered as a player card, but treat that as "worked for us," not a guarantee. Test by pasting your URL into a draft post without publishing.

For your agent: the prompt

Paste into Claude Code, Codex, Cursor or any coding agent with access to your Cloudflare account. Fill in the {BRACES}.

Build an email signup form that runs INSIDE a post on X via a player card, as one Cloudflare Worker on {DOMAIN}. Newsletter provider: {PROVIDER} (worked example below is beehiiv; adapt the API call).

ROUTES

- GET /join/x Card page, static HTML. In <head>:
 - <meta name="twitter:card" content="player">
 - <meta name="twitter:title" content="{TITLE}">
 - <meta name="twitter:description" content="{WHAT THEY GET}">
 - <meta name="twitter:image" content="https://{DOMAIN}/join/card.png">
 - <meta name="twitter:player" content="https://{DOMAIN}/join/play">
 - <meta name="twitter:player:width" content="640">
 - <meta name="twitter:player:height" content="480">
 - + og:title, og:description, og:image, og:url.
 Body: script redirecting humans to /join/play.
- GET /join/play The form, designed for 640x480 and phones: headline, one line of copy, email input (type=email, inputmode=email, autocomplete=email, font-size >= 16px), hidden honeypot input "company", submit button, inline error text, and a thank-you state that auto-starts the /join/guide.pdf download (download button as fallback).
 - On pointerdown/touchstart/click of the input: window.focus() then input.focus(). Never preventDefault on pointer events.
 - Add an "Open the form in a new tab" link using window.open().
- GET /join/card.png Screenshot of the form at 640x480 (2x).
- GET /join/guide.pdf The lead magnet.
- POST /join/api/subscribe JSON {email, company}:
 - 400 if honeypot filled or email invalid (validate server-side).
 - 429 if more than 5 attempts per CF-Connecting-IP in 10 minutes.
 - If env.API_KEY / env.PUBLICATION_ID missing: console.error and return 503 {ok:false, message:"Signups aren't open just yet"}.
 - Else (beehiiv): POST https://api.beehiiv.com/v2/publications/{env.PUBLICATION_ID}/subscriptions
 - Authorization: Bearer \${env.API_KEY}
 - {email, reactivate_existing:false, send_welcome_email:true, utm_source:"x", utm_medium:"tweet-card", referring_site:"x.com"}
 - 2xx → {ok:true}. Else log status + body, return 502 friendly msg.
- No X-Frame-Options, no CSP frame-ancestors. HTTPS only.

SECRETS wrangler secret put API_KEY ; wrangler secret put PUBLICATION_ID

TEST

1. curl -A "Twitterbot/1.0" https://{DOMAIN}/join/x → tags present.
2. Local page on another origin iframes /join/play at 640x480. Headless browser: click the parent page first, then click the input, confirm it's focused, type an email, submit, screenshot each state.
3. Honeypot filled → rejected. Bad email → rejected. 6th try → 429.
4. One real signup → appears in {PROVIDER}; the thank-you screen auto-starts the PDF download. Then delete the test subscriber.
5. Paste /join/x into a draft post on X (don't publish) to check it.

THANKS FOR READING

Now let people subscribe without leaving the post.

See it live: the form that sent you this guide is at games.codeshiftagent.com/join.

— Andrew Warner