macOS companion app

Canvas

The macOS app embeds an agent-controlled **Canvas panel** using `WKWebView`. It is a lightweight visual workspace for HTML/CSS/JS, A2UI, and small interactive UI surfaces.

Where Canvas lives

Canvas state is stored under Application Support:

- `~/Library/Application Support/OpenClaw/canvas/<session>/...`

The Canvas panel serves those files via a **custom URL scheme**:

- `openclaw-canvas://<session>/<path>`

Examples:

- `openclaw-canvas://main/` → `<canvasRoot>/main/index.html`
- `openclaw-canvas://main/assets/app.css` → `<canvasRoot>/main/assets/app.css`
- `openclaw-canvas://main/widgets/todo/` → `<canvasRoot>/main/widgets/todo/index.html`

If no `index.html` exists at the root, the app shows a **built-in scaffold page**.

Panel behavior

- Borderless, resizable panel anchored near the menu bar (or mouse cursor).
- Remembers size/position per session.
- Auto-reloads when local canvas files change.
- Only one Canvas panel is visible at a time (session is switched as needed).

Canvas can be disabled from Settings → **Allow Canvas**. When disabled, canvas node commands return `CANVAS_DISABLED`.

Agent API surface

Canvas is exposed via the **Gateway WebSocket**, so the agent can:

- show/hide the panel
- navigate to a path or URL
- evaluate JavaScript
- capture a snapshot image

CLI examples:

```
openclaw nodes canvas present --node <id>
openclaw nodes canvas navigate --node <id> --url "/"
openclaw nodes canvas eval --node <id> --js "document.title"
openclaw nodes canvas snapshot --node <id>
```

Notes:

- `canvas.navigate` accepts **local canvas paths**, `http(s)` URLs, and `file://` URLs.
- If you pass `"/"`, the Canvas shows the local scaffold or `index.html`.

A2UI in Canvas

A2UI is hosted by the Gateway canvas host and rendered inside the Canvas panel. When the Gateway advertises a Canvas host, the macOS app auto-navigates to the A2UI host page on first open.

Default A2UI host URL:

```
http://<gateway-host>:18789/__openclaw__/a2ui/
```

A2UI commands (v0.8)

Canvas currently accepts **A2UI v0.8** server→client messages:

- `beginRendering`
- `surfaceUpdate`
- `dataModelUpdate`
- `deleteSurface`

`createSurface` (v0.9) is not supported.

CLI example:

```
cat > /tmp/a2ui-v0.8.jsonl <<'EOFA2'
{"surfaceUpdate":{"surfaceId":"main","components":[{"id":"root","component":{"Colu
{"beginRendering":{"surfaceId":"main","root":"root"}}
EOFA2

openclaw nodes canvas a2ui push --jsonl /tmp/a2ui-v0.8.jsonl --node <id>
```

Quick smoke:

```
openclaw nodes canvas a2ui push --node <id> --text "Hello from A2UI"
```

Triggering agent runs from Canvas

Canvas can trigger new agent runs via deep links:

```
openclaw://agent?...
```

Example (in JS):

```
window.location.href = "openclaw://agent?message=Review%20this%20desi
```

The app prompts for confirmation unless a valid key is provided.

Security notes

- Canvas scheme blocks directory traversal; files must live under the session root.
- Local Canvas content uses a custom scheme (no loopback server required).
- External `http(s)` URLs are allowed only when explicitly navigated.

‹ WebChat Gateway Lifecycle ›