

شیت پایین (وبویوی داخل بازی)

شیت کشویی که همه‌ی صفحه‌های بازی را داخل بازی نشان می‌دهد. از پایین بالا می‌آید، با کشیدن، زدن بیرون یا دکمه‌ی خود صفحه بسته می‌شود و پیام‌های صفحه را به بازی می‌رساند.

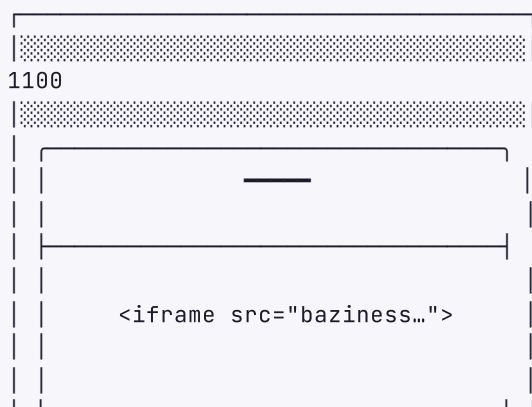
چرا شیت

همه‌ی صفحه‌های بازی (ورود، پروفایل، فروشگاه جایزه، پیشنهاد بازی) داخل بازی باز می‌شوند، نه در تب جدید. ظرف پیشنهادی یک شیت پایین است: روی بازی تیره شده از پایین بالا می‌آید، کمی پایین‌تر از بالای صفحه می‌ایستد و لبه‌ی بالایی گرد با یک دستگیره دارد. بازیکن به سه روش می‌تواند آن را ببندد و خود صفحه هم می‌تواند.

امتحان‌ش کنید:

اجزا

Text



← `rgba(8,6,4,.5) · z-index` لایه‌ی تیره (زدن = بستن)

1100

← شیت: ارتفاع ۹۲٪، حداکثر عرض 560px

22px گوشه‌ها 22px 0 0

← تنها جای کشیدن - 26px ناحیه‌ی دستگیره

← `load` تا رویداد `spinner!` `sandbox!` بدون

رفتارهای لازم

1. باز شدن: شیت را زیر صفحه بکشید و در فریم بعد به بالا متحرک کنید (حدود ۳۲۰ میلی‌ثانیه، `ease-out`).

2. سه راه بستن:

- کشیدن دستگیره به پایین بیشتر از `(ارتفاع شیت 28%, 150px min)`؛ کشیدن کوتاه‌تر شیت را به جایش برمی‌گرداند
- زدن روی ناحیه‌ی تیره‌ی بالای شیت
- صفحه پیام `{ "type": "close" }` بفرستد

3. فقط دستگیره کشیدنی است. صفحه لمس‌های خودش را می‌بلعد، پس کشیدنی که داخل صفحه شروع شود شیت را تکان نمی‌دهد.
4. روی `iframe sandbox` نگذارید. صفحه‌ی ورود به کوکی‌ها، پنجره‌های بازشو (برای کد) و `redirect` خودش نیاز دارد. `clipboard-read; clipboard-write; payment` را مجاز کنید.
5. تا بارگذاری صفحه یک `spinner` نشان دهید.
6. پیام‌ها را فقط از `origin` خود صفحه بپذیرید. هم JSON رشته‌ای و هم شیء ساده را بخوانید و `type` رشته‌ای را الزامی کنید.
7. تمیز ببندید: به پایین متحرک کنید، بعد فقط یک بار جمعش کنید: فریم را به `about:blank` ببرید، گوش دادن به پیام‌ها را قطع کنید و بعد «بسته شد» را اعلام کنید. اگر در این فاصله شیت دوباره باز شده، کاری نکنید.
8. بالای همه چیز باشد، اسپلش هم: صفحه‌ی ورود ممکن است از اسپلش باز شود.
9. اندازه‌اش مثل ستون بازی باشد: در گوشی تمام عرض، در صفحه‌های پهن حداکثر حدود ۵۶۰ پیکسل و وسط‌چین.

وب: React، TypeScript و CSS

TypeScript (web)

```
// bazinesswebview.ts - the sheet, framework-free.
export interface BazinessMessage { type: string; data?: unknown }

export class BazinessSheet {
  readonly el: HTMLElement;
  private sheet: HTMLElement;
  private frame: HTMLIFrameElement;
  private spinner: HTMLElement;
  private allowedOrigin: string | null = null;
  private isOpen = false;
  private dragging = false;
  private dragStartY = 0;
  private dragDelta = 0;
  private sheetHeight = 1;

  onMessage: ((m: BazinessMessage) => void) | null = null;
  onClose: (() => void) | null = null;

  constructor() {
    this.frame = Object.assign(document.createElement("iframe"), { className: "baziness-frame" });
    this.frame.setAttribute("allow", "clipboard-read; clipboard-write; payment"); // no sandbox!
    this.spinner = Object.assign(document.createElement("div"), { className: "baziness-spinner" });
    const grabberZone = document.createElement("div");
    grabberZone.className = "baziness-grabber-zone";
    grabberZone.innerHTML = '<div class="baziness-grabber"></div>';
    this.sheet = document.createElement("div");
    this.sheet.className = "baziness-sheet";
    this.sheet.append(grabberZone, this.spinner, this.frame);
    this.el = document.createElement("div");
    this.el.className = "overlay baziness-webview hidden";
    this.el.append(this.sheet);
```

```

document.body.append(this.el);

this.frame.addEventListener("load", () => this.spinner.classList.add("hidden"));
grabberZone.addEventListener("pointerdown", (e) => this.dragStart(e));
grabberZone.addEventListener("pointermove", (e) => this.dragMove(e));
grabberZone.addEventListener("pointerup", () => this.dragEnd());
grabberZone.addEventListener("pointercancel", () => this.dragEnd());
this.el.addEventListener("pointerdown", (e) => { if (e.target === this.el) this.close();
}); // tap outside
}

open(url: string, cropBottomPx = 0): void {
    if (this.isOpen) return;
    this.isOpen = true;
    this.frame.style.marginBottom = cropBottomPx > 0 ? `${-cropBottomPx}px` : "";
    this.allowedOrigin = new URL(url).origin;
    this.spinner.classList.remove("hidden");
    this.el.classList.remove("hidden");
    this.sheet.style.transition = "none";
    this.sheet.style.transform = "translateY(100%)";
    this.frame.src = url;
    window.addEventListener("message", this.onWindowMessage);
    requestAnimationFrame(() => {
        this.sheet.style.transition = "";
        this.sheet.style.transform = "translateY(0)";
    });
}

close(): void {
    if (!this.isOpen) return;
    this.isOpen = false;
    this.dragging = false;
    window.removeEventListener("message", this.onWindowMessage);
    this.sheet.style.transition = "";
    this.sheet.style.transform = "translateY(100%)";
    let torn = false;
    const teardown = () => {
        if (torn || this.isOpen) return; // already torn down, or reopened meanwhile
        torn = true;
        this.sheet.removeEventListener("transitionend", teardown);
        this.el.classList.add("hidden");
        this.frame.src = "about:blank";
        this.frame.style.marginBottom = "";
        this.sheet.style.transform = "";
        this.allowedOrigin = null;
        this.onClosed?.();
    };
    this.sheet.addEventListener("transitionend", teardown);
    window.setTimeout(teardown, 400);
}

private dragStart(e: PointerEvent) {
    this.dragging = true;
    this.dragStartY = e.clientY;
    this.dragDelta = 0;
    this.sheetHeight = this.sheet.getBoundingClientRect().height || 1;
    this.sheet.style.transition = "none";
    (e.currentTarget as HTMLElement).setPointerCapture(e.pointerId);
}

private dragMove(e: PointerEvent) {

```

```

    if (!this.dragging) return;
    this.dragDelta = Math.max(0, e.clientY - this.dragStartY); // downward only
    this.sheet.style.transform = `translateY(${this.dragDelta}px)`;
  }
  private dragEnd() {
    if (!this.dragging) return;
    this.dragging = false;
    this.sheet.style.transition = "";
    if (this.dragDelta > Math.min(150, this.sheetHeight * 0.28)) this.close();
    else this.sheet.style.transform = "translateY(0)"; // spring back
  }

  private onWindowMessage = (event: MessageEvent) => {
    if (this.allowedOrigin && event.origin !== this.allowedOrigin) return;
    let value: unknown = event.data;
    if (typeof value === "string") { try { value = JSON.parse(value); } catch { return; } }
    if (value && typeof value === "object" && typeof (value as BazinessMessage).type ===
"string") {
      this.onMessage?.(value as BazinessMessage);
    }
  };
}

```

React

```

// BazinessSheet.tsx – the React version. Mount once at the app root; baziness.host drives
it.
import { useEffect, useRef, useState } from "react";
import { createPortal } from "react-dom";
import { baziness } from "../platform/baziness";

export function BazinessSheet() {
  const [url, setUrl] = useState<string | null>(null);
  const [shown, setShown] = useState(false);
  const [loading, setLoading] = useState(true);
  const [drag, setDrag] = useState(0);
  const dragStart = useRef<number | null>(null);

  useEffect(() => {
    baziness.host = {
      open: (next) => {
        setLoading(true);
        setUrl(next);
        requestAnimationFrame(() => requestAnimationFrame(() => setShown(true))); // slide up
      },
      close: () => setShown(false),
    };
    return () => { baziness.host = null; };
  }, []);

  useEffect(() => { // messages, from the page's origin
    only
    if (!url) return;
    const origin = new URL(url).origin;
    const onMessage = (e: MessageEvent) => {
      if (e.origin !== origin) return;
      const msg = parse(e.data); // see "Handling page messages"
      if (msg) baziness.handleMessage(msg);
    }
  }, [url]);
}

```

```

};
window.addEventListener("message", onMessage);
return () => window.removeEventListener("message", onMessage);
}, [url]);

const onTransitionEnd = () => {
  if (!shown && url) { // finished sliding down → tear down
once
    setUrl(null);
    setDrag(0);
    baziness.handleClosed();
    measureViewport(); // the phone may have moved its nav
bar
  }
};

if (!url) return null;
return createPortal(
  <div className={`overlay baziness-overlay ${shown ? "is-open" : ""}`}
    onPointerDown={(e) => e.target === e.currentTarget && setShown(false)}>
    <div className="baziness-sheet"
      style={{ transform: shown ? `translateY(${drag}px)` : "translateY(100%)",
        transition: dragStart.current !== null ? "none" : undefined }}
      onTransitionEnd={onTransitionEnd}>
      <div className="baziness-grabber-zone"
        onPointerDown={(e) => { dragStart.current = e.clientY;
e.currentTarget.setPointerCapture(e.pointerId); }}
        onPointerMove={(e) => dragStart.current !== null && setDrag(Math.max(0,
e.clientY - dragStart.current))}
        onPointerUp={() => { dragStart.current = null; if (drag > 140) setShown(false);
else setDrag(0); }}>
        <span className="baziness-grabber" />
      </div>
      {loading && <div className="baziness-loading"><Spinner /></div>}
      <iframe className="baziness-frame" src={url} title="بازینس"
        allow="clipboard-read; clipboard-write; payment" onLoad={() =>
setLoading(false)} />
      </div>
    </div>,
    document.getElementById("app-frame") ?? document.body,
  );
}

```

CSS

```

/* Above the splash (z 1000) so it shows during the first-time login choice. */
.overlay.baziness-webview {
  position: fixed;
  inset: 0;
  z-index: 1100;
  display: flex;
  justify-content: center;
  align-items: flex-end; /* dock the sheet to the bottom */
  background: rgba(8, 6, 4, 0.5); /* the dimmed, tap-to-close area */
}
.overlay.baziness-webview.hidden { display: none; }

.baziness-sheet {
  position: relative;

```

```

width: min(560px, 100%);          /* full width on phones, capped on wide screens */
height: 92%;                      /* stops short of the top */
display: flex;
flex-direction: column;
background: #fff;
border-radius: 22px 22px 0 0;
overflow: hidden;                 /* also clips the cropped bottom bar */
box-shadow: 0 -10px 34px rgba(0, 0, 0, 0.42);
transform: translateY(100%);
transition: transform 320ms cubic-bezier(0.22, 0.61, 0.36, 1);
will-change: transform;
}

/* The only draggable area – the iframe captures its own pointer events. */
.baziness-grabber-zone {
  flex: 0 0 auto;
  height: 26px;
  display: flex;
  align-items: center;
  justify-content: center;
  cursor: grab;
  touch-action: none;
  background: #fff;
}

.baziness-grabber {
  width: 44px;
  height: 5px;
  border-radius: 999px;
  background: rgba(0, 0, 0, 0.22);
}

.baziness-frame {
  flex: 1 1 auto;
  width: 100%;
  min-height: 0;
  border: 0;
  background: #fff;
}

.baziness-spinner {
  position: absolute;
  top: 50%;
  left: 50%;
  width: 42px;
  height: 42px;
  margin: -21px 0 0 -21px;
  border-radius: 50%;
  border: 4px solid rgba(120, 90, 46, 0.25);
  border-top-color: #b4824a;
  animation: spin 0.9s linear infinite;
}

.baziness-spinner.hidden { display: none; }
@keyframes spin { to { transform: rotate(360deg); } }

```

کامپوننت `BazinessWebView` در **کد کلاینت** ← **Cocos Creator** همین کار را با یک پری فب `WebView` انجام می‌دهد: لایه‌ی بارگذاری تا لود شدن صفحه، دکمه‌ی بستنی که بعد از چند ثانیه ظاهر می‌شود، و انیمیشن‌های لغزیدن `Open` / `Close`.

Unity

Unity وب‌ویوی داخلی ندارد و به پلاگین نیاز دارید. این طرح از [gree/unity-webview](https://github.com/gree/unity-webview) استفاده می‌کند: صفحه ۸٪ پایین‌تر از بالای صفحه شروع می‌شود، دستگیره یک المان UI یونیتی بالای آن است و پیام‌های صفحه با یک پل کوچک جاوااسکریپتی به `#C` می‌رسند.

C# (Unity)

```
// BazinessSheet.cs - a bottom sheet around a WebView plugin (gree/unity-webview).
using System;
using UnityEngine;
using UnityEngine.EventSystems;

public class BazinessSheet : MonoBehaviour, IDragHandler, IEndDragHandler // put this on
the grabber UI
{
    [SerializeField] RectTransform grabber; // a 26px bar drawn by Unity above the web
    view
    [SerializeField] GameObject dimmer; // full-screen button behind; onClick →
    Close()
    WebViewObject web;
    Action<string> onMessage;
    Action onClosed;
    float topMargin, drag;

    public void Open(string url, Action<string> onMessage, Action onClosed = null)
    {
        this.onMessage = onMessage;
        this.onClosed = onClosed;
        topMargin = Screen.height * 0.08f; // the sheet stops 8% short of the top
        drag = 0;
        web = new GameObject("BazinessWebView").AddComponent<WebViewObject>();
        web.Init(
            cb: msg => this.onMessage?.Invoke(msg), // messages forwarded by the bridge
            below
            ld: _ => web.EvaluateJS(
                "window.addEventListener('message',function(e){" +
                "Unity.call(typeof e.data==='string'?e.data:JSON.stringify(e.data));});"));
        Layout();
        web.LoadURL(url);
        web.SetVisibility(true);
        dimmer.SetActive(true);
        grabber.gameObject.SetActive(true);
    }

    public void OnDrag(PointerEventData e) { drag = Mathf.Max(0, drag - e.delta.y); Layout();
```

```

}

public void OnEndDrag(PointerEventData e)
{
    if (drag > Mathf.Min(150, (Screen.height - topMargin) * 0.28f)) Close();
    else { drag = 0; Layout(); } // spring back
}

public void Close()
{
    if (web == null) return;
    Destroy(web.gameObject);
    web = null;
    dimmer.SetActive(false);
    grabber.gameObject.SetActive(false);
    onClose?.Invoke();
}

void Layout()
{
    int top = (int)(topMargin + drag + grabber.rect.height);
    web.SetMargins(0, top, 0, 0);
    grabber.anchoredPosition = new Vector2(0, -(topMargin + drag));
}
}

```

اشتباه‌های رایج

اشتباه	نتیجه
روی <code>sandbox</code> <code>iframe</code>	ورود گیر می‌کند: پنجره‌های بازشو و کوکی‌ها مسدود می‌شوند
کل شیت کشیدنی است	کشیدن داخل صفحه هم صفحه را اسکرول می‌کند و هم شیت را تکان می‌دهد، یا هیچ چیز تکان نمی‌خورد
بررسی <code>origin</code> برای پیام‌ها نیست	پیامی از جای دیگر به جای پیام صفحه گرفته می‌شود
«بسته شد» دو بار اعلام می‌شود (پایان انیمیشن + زمان‌سنج)	ورودی که در انتظار است دو بار تمام می‌شود؛ بسته‌ی شروع دو بار باز می‌شود
شیت زیر اسپلش است	صفحه‌ی ورود نادیده پشت اسپلش باز می‌شود
فریم به <code>about:blank</code> برنمی‌گردد	صفحه‌ی قبلی در پس‌زمینه به اجرا ادامه می‌دهد