好的，这是基于你目前**最新、最完美**的代码版本（支持单文件拖拽、PDF/图片/视频全兼容、全局老板键、伪装模式）整理的完整开发与使用文档。

你可以将此内容保存为 README.md 或直接作为项目的说明书。

# 🐟 摸鱼神器 (MoYu Player) - 开发文档

**伪装名称**：邮箱助手 (Outlook Assistant)

**核心功能**：无边框隐蔽播放、全能格式支持、全局老板键、窗口置顶。

## 1. 环境准备 (Prerequisites)

在编译之前，请确保你的电脑安装了以下环境：

1. **Go 语言环境**: [下载 Go](https://go.dev/dl/) (建议版本 1.20+)。
2. **GCC 编译器 (MinGW)**:
   * 由于使用了 webview 库 (CGO)，Windows 必须安装 GCC。

本地安装包

## 2. 项目目录结构

请建立一个文件夹，确保文件结构如下：

Plaintext

moyu\_player/  
├── go.mod # Go 模块文件  
├── go.sum # 依赖校验文件  
├── main.go # 后端核心代码 (系统API、热键、窗口管理)  
├── index.html # 前端界面代码 (UI、播放逻辑、拖拽处理)  
├── icon.ico # 伪装图标 (如 Outlook 或 设置齿轮图标)  
└── rsrc.syso # (运行 rsrc 命令后生成的资源文件)

## 3. 核心代码

### 3.1 初始化项目

在目录中运行终端：

Bash

go mod init moyu\_player  
go get [github.com/webview/webview\_go](http://github.com/webview/webview_go)  
go install github.com/akavel/rsrc@latest

### 3.2 后端代码 (main.go)

*功能：无边框实现、全局热键(Alt+Q)监听、窗口置顶/拖拽/显隐控制。*

Go

package main  
  
import (  
 "embed"  
 "fmt"  
 "log"  
 "net"  
 "net/http"  
 "os"  
 "strings"  
 "syscall"  
 "time"  
  
 webview "github.com/webview/webview\_go"  
)  
  
//go:embed index.html  
var content embed.FS  
  
// --- Windows API 定义 ---  
var (  
 user32 = syscall.NewLazyDLL("user32.dll")  
 procGetWindowLong = user32.NewProc("GetWindowLongW")  
 procSetWindowLong = user32.NewProc("SetWindowLongW")  
 procSetWindowPos = user32.NewProc("SetWindowPos")  
 procSendMessage = user32.NewProc("SendMessageW")  
 procReleaseCapture = user32.NewProc("ReleaseCapture")  
 procShowWindow = user32.NewProc("ShowWindow")  
 procGetAsyncKeyState = user32.NewProc("GetAsyncKeyState")  
 procSetForegroundWindow = user32.NewProc("SetForegroundWindow")  
)  
  
const (  
 GWL\_STYLE = -16  
 WS\_CAPTION = 0x00C00000  
 WS\_THICKFRAME = 0x00040000  
 WM\_SYSCOMMAND = 0x0112  
 SC\_MINIMIZE = 0xF020  
 SC\_DRAG\_MOVE = 0xF012  
  
 SW\_HIDE = 0  
 SW\_RESTORE = 9  
  
 VK\_MENU = 0x12 // Alt 键  
 VK\_Q = 0x51 // Q 键  
)  
  
type PlayerBridge struct {  
 w webview.WebView  
 isVisible bool  
}  
  
// WinMove 调用系统原生拖拽  
func (p \*PlayerBridge) WinMove() {  
 hwnd := p.w.Window()  
 procReleaseCapture.Call()  
 procSendMessage.Call(uintptr(hwnd), uintptr(WM\_SYSCOMMAND), uintptr(SC\_DRAG\_MOVE), 0)  
}  
  
// WinMin 最小化  
func (p \*PlayerBridge) WinMin() {  
 hwnd := p.w.Window()  
 procSendMessage.Call(uintptr(hwnd), uintptr(WM\_SYSCOMMAND), uintptr(SC\_MINIMIZE), 0)  
}  
  
// WinClose 关闭程序  
func (p \*PlayerBridge) WinClose() {  
 p.w.Terminate()  
}  
  
// SetAlwaysOnTop 设置窗口置顶  
func (p \*PlayerBridge) SetAlwaysOnTop(isTop bool) {  
 hwnd := p.w.Window()  
 hwndTopMost := -1  
 hwndNoTopMost := -2  
 targetOrder := hwndNoTopMost  
 if isTop {  
 targetOrder = hwndTopMost  
 }  
 // 0x0003 = SWP\_NOMOVE | SWP\_NOSIZE  
 procSetWindowPos.Call(uintptr(hwnd), uintptr(targetOrder), 0, 0, 0, 0, 0x0003)  
}  
  
// setFrameless 修改无边框样式  
func (p \*PlayerBridge) setFrameless(frameless bool) {  
 hwnd := p.w.Window()  
 index := GWL\_STYLE  
 style, \_, \_ := procGetWindowLong.Call(uintptr(hwnd), uintptr(index))  
 newStyle := int32(style)  
  
 if frameless {  
 newStyle = newStyle &^ (WS\_CAPTION | WS\_THICKFRAME)  
 } else {  
 newStyle = newStyle | WS\_CAPTION | WS\_THICKFRAME  
 }  
 procSetWindowLong.Call(uintptr(hwnd), uintptr(index), uintptr(newStyle))  
 // 刷新窗口状态  
 procSetWindowPos.Call(uintptr(hwnd), 0, 0, 0, 0, 0, 0x0020|0x0001|0x0002|0x0004)  
}  
  
// ToggleMode 切换 摸鱼模式/正常模式  
func (p \*PlayerBridge) ToggleMode(isMini bool) {  
 if isMini {  
 p.setFrameless(true)  
 p.w.SetSize(320, 180, webview.HintNone) // 极简尺寸  
 } else {  
 p.setFrameless(false)  
 p.w.SetSize(800, 600, webview.HintNone) // 正常尺寸  
 }  
}  
  
// ToggleVisibility 老板键逻辑 (双向开关)  
func (p \*PlayerBridge) ToggleVisibility() {  
 hwnd := p.w.Window()  
 if p.isVisible {  
 // 隐藏  
 procShowWindow.Call(uintptr(hwnd), uintptr(SW\_HIDE))  
 p.isVisible = false  
 } else {  
 // 恢复并强制前台  
 procShowWindow.Call(uintptr(hwnd), uintptr(SW\_RESTORE))  
 procSetForegroundWindow.Call(uintptr(hwnd))  
 p.isVisible = true  
 }  
}  
  
func (p \*PlayerBridge) Log(msg string) {  
 fmt.Println("Frontend:", msg)  
}  
  
func main() {  
 // 1. 启动本地静态文件服务器 (解决跨域和资源加载)  
 listener, err := net.Listen("tcp", "127.0.0.1:0")  
 if err != nil {  
 log.Fatal(err)  
 }  
 port := listener.Addr().(\*net.TCPAddr).Port  
 dir, \_ := os.Getwd()  
 fileHandler := http.FileServer(http.Dir(dir))  
 go func() { http.Serve(listener, fileHandler) }()  
  
 fmt.Printf("Server started at http://127.0.0.1:%d\n", port)  
  
 // 2. 初始化 WebView  
 w := webview.New(true)  
 defer w.Destroy()  
  
 // --- 核心伪装：设置标题 ---  
 w.SetTitle("邮箱助手")   
 w.SetSize(800, 600, webview.HintNone)  
  
 bridge := &PlayerBridge{  
 w: w,  
 isVisible: true,  
 }  
  
 // 3. 绑定 Go 函数给前端 JS 调用  
 w.Bind("toggleMode", bridge.ToggleMode)  
 w.Bind("goLog", bridge.Log)  
 w.Bind("winMin", bridge.WinMin)  
 w.Bind("winClose", bridge.WinClose)  
 w.Bind("setTop", bridge.SetAlwaysOnTop)  
 w.Bind("winMove", bridge.WinMove)  
 w.Bind("bossKey", bridge.ToggleVisibility)  
  
 // 4. 注入 HTML 并替换端口  
 htmlContent, \_ := content.ReadFile("index.html")  
 finalHTML := strings.Replace(string(htmlContent), "{{PORT}}", fmt.Sprintf("%d", port), -1)  
 w.SetHtml(finalHTML)  
  
 // 5. 启动全局键盘监听协程 (Alt + Q)  
 go func() {  
 for {  
 // 50ms 检测一次  
 time.Sleep(50 \* time.Millisecond)  
  
 // 检查 Alt(VK\_MENU) 和 Q(VK\_Q)  
 altState, \_, \_ := procGetAsyncKeyState.Call(uintptr(VK\_MENU))  
 qState, \_, \_ := procGetAsyncKeyState.Call(uintptr(VK\_Q))  
  
 // 0x8000 代表键被按下  
 if (altState&0x8000 != 0) && (qState&0x8000 != 0) {  
 // 在主线程触发切换  
 w.Dispatch(func() {  
 bridge.ToggleVisibility()  
 })  
 // 防抖动：等待 300ms  
 time.Sleep(300 \* time.Millisecond)  
 }  
 }  
 }()  
  
 w.Run()  
}

### 3.3 前端代码 (index.html)

*功能：全格式拖拽支持、智能UI缩放、顶部隐藏式控制栏。*

HTML

<!DOCTYPE **html**>  
<html lang="zh">  
<head>  
 <meta charset="UTF-8">  
 <script src="https://cdn.jsdelivr.net/npm/hls.js@1"></script>  
 <style>  
 body { margin: 0; background: #000; overflow: hidden; display: flex; flex-direction: column; height: 100vh; }  
   
 /\* 内容容器：居中显示所有媒体 \*/  
 .content-container {  
 flex: 1; width: 100%; height: 100%; position: relative;  
 overflow: hidden; display: flex; justify-content: center; align-items: center;  
 }  
 /\* 默认隐藏，激活时显示 \*/  
 video, img, iframe { width: 100%; height: 100%; display: none; border: none; }  
 video, img { object-fit: contain; } /\* 保持比例 \*/  
 .active { display: block !important; }  
  
 /\* 顶部感应区 \*/  
 .top-hover-zone {  
 position: absolute; top: 0; left: 0; width: 100%; height: 80px; z-index: 200;  
 display: flex; flex-direction: column;  
 }  
 /\* 控制栏：平时隐藏，鼠标移入显示 \*/  
 .controls {  
 width: 100%; background: rgba(0,0,0,0.85); display: flex; align-items: center;   
 box-sizing: border-box;   
 font-size: clamp(6px, 2vw, 16px); /\* 智能缩放字号 \*/  
 padding: 0.5em 0.6em;   
 opacity: 0; transform: translateY(-100%); transition: all 0.3s ease;  
 -webkit-app-region: drag;   
 }  
 .top-hover-zone:hover .controls { opacity: 1; transform: translateY(0); }  
   
 /\* 摸鱼模式下布局调整 \*/  
 .controls.mini input { display: none !important; }  
 .controls.mini { justify-content: center; gap: 0.6em; }  
   
 input {   
 flex: 1; font-size: 1em; margin-right: 0.6em; outline: none; padding: 0.3em 0.5em;   
 background: #333; color: white; border: 1px solid #555; border-radius: 0.3em;  
 -webkit-app-region: no-drag;   
 }  
 button {   
 font-size: 1em; cursor: pointer; margin-right: 0; padding: 0.3em 0.5em;   
 background: #444; color: white; border: 1px solid #666; border-radius: 0.3em;  
 -webkit-app-region: no-drag; display: flex; align-items: center; justify-content: center;  
 min-width: 2em; height: 2em;   
 }  
 button:hover { background: #666; }  
   
 #btnBoss { color: #aaa; border-color: #888; }  
 #btnBoss:hover { color: #fff; background: #666; }  
  
 .controls > \* { margin-right: 0.4em; }  
 .controls.mini > \* { margin-right: 0; }   
   
 /\* 拖拽手柄 \*/  
 .btn-move {  
 -webkit-app-region: no-drag; display: inline-flex; justify-content: center; align-items: center;   
 width: 2em; height: 2em; font-size: 1em; cursor: move; box-sizing: border-box;  
 background: #2b5c82; color: white; border: 1px solid #3e7ca8; border-radius: 0.3em;  
 -webkit-user-select: none; user-select: none;  
 }  
 .btn-move:hover { background: #3e7ca8; }  
   
 #btnTop { font-size: 1em; padding: 0.3em 0.5em; }  
 .win-controls { display: flex; align-items: center; gap: 0.4em; margin-left: auto; }  
 .controls.mini .win-controls { margin-left: 0; }  
 .btn-close { background: #c00; border-color: #a00; }  
 </style>  
</head>  
<body>  
 <div class="top-hover-zone">  
 <div class="controls" id="ctrlBar">  
 <input type="text" id="urlInput" placeholder="支持 视频 / 图片 / PDF 拖拽">  
 <button id="btnBoss" onclick="toggleBoss()" title="老板键 (Alt+Q)">👻</button>  
 <button id="btnTop" onclick="toggleTop()" title="置顶">📌</button>  
 <button onclick="playMedia()" title="播放/刷新">▶</button>  
 <button onclick="toggleSize()" title="切换模式">📺</button>  
   
 <div class="win-controls" id="winBtns" style="display:none;">  
 <div class="btn-move" onmousedown="startDrag()" title="拖动">✥</div>  
 <button onclick="winMin()">-</button>  
 <button class="btn-close" onclick="winClose()">×</button>  
 </div>  
 </div>  
 </div>  
  
 <div class="content-container">  
 <video id="vPlayer" controls autoplay></video>  
 <img id="imgViewer" alt="Image Viewer">  
 <iframe id="pdfViewer"></iframe>  
 </div>  
  
 <script>  
 const PORT = "{{PORT}}";  
 const video = document.getElementById('vPlayer');  
 const img = document.getElementById('imgViewer');  
 const pdf = document.getElementById('pdfViewer');  
 const input = document.getElementById('urlInput');  
 const winBtns = document.getElementById('winBtns');  
 const ctrlBar = document.getElementById('ctrlBar');  
   
 let isMini = false;  
 let isTop = false;  
 let hls = null;  
 let currentType = 'video';  
  
 function playMedia() {  
 let val = input.value;  
 if (!val) return;  
 if (!val.startsWith('http') && !val.startsWith('blob:')) val = `http://127.0.0.1:${PORT}/${val}`;  
 loadResource(val);  
 }  
  
 // 智能资源加载  
 function loadResource(src) {  
 video.pause();  
 [video, img, pdf].forEach(el => el.classList.remove('active'));  
   
 const cleanSrc = src.split('?')[0].toLowerCase();  
 if (/\.(jpg|jpeg|png|gif|webp|svg)$/.test(cleanSrc)) {  
 img.src = src; img.classList.add('active');  
 currentType = 'image';  
 } else if (cleanSrc.endsWith('.pdf')) {  
 pdf.src = src; pdf.classList.add('active');  
 currentType = 'pdf';  
 } else {  
 video.classList.add('active');  
 currentType = 'video';  
 if (hls) { hls.destroy(); hls = null; }  
 if (cleanSrc.includes('.m3u8') && Hls.isSupported()) {  
 hls = new Hls(); hls.loadSource(src); hls.attachMedia(video);  
 hls.on(Hls.Events.MANIFEST\_PARSED, () => video.play());  
 } else {  
 video.src = src; video.play();  
 }  
 }  
 }  
  
 function startDrag() { window.winMove(); }  
 function toggleBoss() { window.bossKey(); }  
   
 function toggleSize() {  
 isMini = !isMini;  
 toggleMode(isMini);   
 ctrlBar.classList.toggle('mini', isMini);  
 winBtns.style.display = isMini ? 'flex' : 'none';  
 document.querySelector('.top-hover-zone').style.height = isMini ? '100%' : '80px';  
 }  
   
 function toggleTop() {  
 isTop = !isTop; window.setTop(isTop);  
 const btn = document.getElementById('btnTop');  
 btn.style.borderColor = isTop ? '#2ea44f' : '#666';  
 btn.style.background = isTop ? '#2ea44f' : '#444';  
 }  
  
 // 拖拽监听  
 document.body.addEventListener('dragover', e => e.preventDefault());  
 document.body.addEventListener('drop', e => {  
 e.preventDefault();  
 if (e.dataTransfer.files.length > 0) {  
 const file = e.dataTransfer.files[0];  
 input.value = file.name;  
 loadResource(URL.createObjectURL(file));  
 }  
 });  
   
 // 双击暂停 (仅视频)  
 document.body.addEventListener('dblclick', (e) => {  
 if (['BUTTON', 'INPUT'].includes(e.target.tagName) || e.target.closest('.controls')) return;  
 if (currentType === 'video') video.paused ? video.play() : video.pause();  
 });  
 </script>  
</body>  
</html>

## 4. 伪装图标与编译打包

为了达到极致的伪装效果，我们需要替换 exe 图标并隐藏控制台黑框。

### 第一步：准备图标

1. 找到一个 .ico 图标文件（例如 Outlook 图标、系统齿轮图标）。
2. 将其重命名为 icon.ico，放在项目根目录下。

### 第二步：生成资源文件

在项目目录下打开终端，执行：

Bash

rsrc -ico icon.ico -o rsrc.syso

*(如果提示找不到 rsrc，请确认已执行 go install github.com/akavel/rsrc@latest)*

### 第三步：最终编译 (去黑框)

执行以下命令生成最终的可执行文件。注意 -H windowsgui 参数，它负责隐藏黑色的命令行窗口。

Bash

go build -ldflags "-H windowsgui" -o "OutlookAssistant.exe"

## 5. 使用指南 (User Guide)

双击 OutlookAssistant.exe 运行。

|  |  |  |
| --- | --- | --- |
| **场景** | **操作** | **效果** |
| **正常模式** | 启动程序 | 800x600 窗口，伪装标题为“邮箱助手”。 |
| **摸鱼模式** | 点击顶部栏 📺 图标 | 窗口变为 **320x180 无边框**，UI自动变小，极其隐蔽。 |
| **移动窗口** | 按住蓝色 ✥ 按钮 | 仅在摸鱼模式下显示，按住它即可随意拖动窗口。 |
| **播放内容** | **拖拽文件**到窗口 | 支持 .mp4, .m3u8 (视频), .jpg/png (图片), .pdf (文档)。 |
| **老板键** | 按键盘 **Alt + Q** | **按一下**：窗口瞬间彻底消失（任务栏也不见）。  **再按一下**：窗口瞬间弹回并置顶。 |
| **窗口置顶** | 点击 📌 按钮 | 绿色为开启。无论怎么点其他软件，播放器永远在最上面。 |

祝你摸鱼愉快！如有需要更新功能，修改代码后重复 **第4步** 编译即可。