# Multi-Station Radio Player: Comprehensive Documentation

This document provides a detailed guide on the structure, configuration, and customization of a multi-station radio player built with HTML, CSS, and JavaScript. This player dynamically fetches song information and offers the flexibility to use a local API or a pre-configured web-based API.

## 1. Overview

This radio player offers a user-friendly interface for enjoying online radio stations. It allows for the addition of multiple stations, each with its own live stream, song information, social media links, and more. Station configuration is done directly within the HTML, simplifying the customization process.

## 2. File Structure

- **`index.html`:** Contains the main HTML for the player, including:
  - Visual structure and interactive elements.
  - Station configurations within a `<script>` tag.
- **`js/main.js`:** Houses the JavaScript code that powers the player's functionality, including:
  - Audio player management.
  - Dynamic fetching and updating of song information (using local or web API).
  - Rendering the station list.
  - Control of interface elements like buttons, menus, and modals.
- **`api.php`:** (Optional) PHP script acting as a local API to extract metadata from radio streams, providing song information.
- **`css/main.min.css`:** Defines the visual styling of the player, including layout, colors, and typography.
- **`custom.css`:** Allows for adding custom styles.
- **`assets/`:** Folder to store images, icons, and other visual assets.

## 3. Detailed Configuration

### 3.1. Configuring Stations (`index.html`)

Radio stations are configured within a `<script>` block in the `index.html` file, defining the `window.streams.stations` object. Each station is an object with the following properties:

| Property | Description |
| --- | --- |
| `name` | Station name displayed on the interface. |
| `hash` | Unique identifier for the station. |
| `description` | Short description of the station. |
| `logo` | Path to the station logo image file. |
| `album` | Path to a default “album” image to display before the actual cover art is loaded. |
| `cover` | Path to the currently playing song’s cover art. |
| `api` | **(Optional)** URL of the local API (`api.php`) configured to fetch station information. This should include the `stream_url` as a parameter. If left blank, the script will use the pre-configured web API within `js/main.js`. |
| `stream_url` | URL of the station’s audio stream. |
| `tv_url` | URL of the station’s live video stream (optional). |
| `server` | Defines the music platform (“spotify” or “itunes”) used to fetch additional info (if the corresponding API is in use). |
| `program` | Object containing information about the current program (optional). |

| Property | Description |
| --- | --- |
| `social` | Object with links to the station's social media profiles (optional). |
| `apps` | Object with links to download the station's apps (optional). |

**Configuration Example:**

```
<script>
window.streams = {
    timeRefresh: 10000, // Refresh time in milliseconds
    stations: [
        {
            name: "Example FM",
            hash: "examplefm",
            description: "The best music!",
            logo: "assets/examplefm_logo.png",
            album: "assets/default_album.jpg",
            cover: "assets/default_album.jpg",
            api: "api.php?url=https://example.com/stream", //
        Local API (optional)
            stream_url: "https://example.com/stream",
            server: "itunes", // Use iTunes API
            social: {
                facebook: "https://facebook.com/examplefm",
                instagram: "https://instagram.com/examplefm"
            }
        },
        // ... more stations
    ]
};
</script>
```

### 3.2. Local API (Optional)

If you choose to use the local API (`api.php`), follow these instructions to set it up:

- **Configuration:**
  - In the `api.php` file, the `$allowedUrls` variable should list all allowed stream URLs.

- **Functionality:**
  - `getMp3StreamTitle()`: Extracts the song title from the stream metadata.
  - `extractArtistAndSong()`: Separates artist and song title.
  - `getAlbumArt()`: Fetches album art (currently set up to use the iTunes API).
  - `updateHistory()`: Maintains a history of played songs.

**Note:**

If the `api` field is left blank in the station configuration, `js/main.js` will default to the pre-configured web API. Make sure the web API you are using is functioning and correctly set up within the JavaScript code.

## 4. Customization, Interface, Interaction, and Publication

The sections regarding:

- **Customizing visual styles** (`css/main.min.css` and `custom.css`)
- **Using custom images and icons** (`assets/`)
- **User interface elements** (header, station selector, history, etc.)
- **User navigation and interaction**
- **Publishing the player to a web server**

### 4.1. Key Elements

- **Header:** Displays the station logo and buttons for accessing the history, station list, and mobile menu.
- **Player Section:** Contains the album art, song information (artist and title), playback controls (play/pause, next/previous station), and volume control.
- **Visualizer:** A simple audio visualizer that responds dynamically to the music.
- **Off-Canvas Sidebar:**
  - **Station List:** Displays all available stations with thumbnails.
  - **History:** Shows a history of recently played songs.
- **Lyrics Modal:** Displays the lyrics of the currently playing song (if available through the Vagalume API).

### 4.2. Navigation

- **Station Selection:** Click on a station in the station list to begin playback.
- **Song History:** Access the history through the button in the header.
- **Song Lyrics:** Click the “Lyrics” button to open the lyrics modal.
- **Mobile Menu:** The menu button in the header provides access to the same functionality on mobile devices.

## 5. Customization

### 5.1. Visual Styles (`css/main.min.css` and `custom.css`)

- Colors, fonts, spacing, element sizes, and other visual properties can be customized by editing the CSS rules.

### 5.2. Images and Icons (`assets/`)

- Replace the default images in the `assets` folder with your own to customize the station logo, album art, and icons.

## 6. Publication

1. Make sure the local API (`api.php`), if used, is configured correctly and accessible on your server.
2. Upload all files and folders (HTML, CSS, JavaScript, PHP, images) to your web server.

## 7. Additional Considerations

- **Copyright:** Ensure that you have the rights to use all images, music, and other content used in your radio player.
- **Stream Metadata:** The accuracy of song information is dependent on the quality of the metadata provided by the radio station's stream.

# Reproductor de Radio Multi-Estación: Documentación Completa

Este documento proporciona una guía detallada sobre la estructura, configuración y personalización de un reproductor de radio multi-estación construido con HTML, CSS y JavaScript. Este reproductor obtiene dinámicamente información de las canciones y ofrece la flexibilidad de usar una API local o una API web preconfigurada.

## 1. Descripción General

Este reproductor de radio ofrece una interfaz fácil de usar para disfrutar de emisoras de radio online. Permite agregar múltiples estaciones, cada una con su propia transmisión en vivo, información de canciones, enlaces a redes sociales y más. La configuración de las estaciones se realiza directamente en el HTML, simplificando el proceso de personalización.

## 2. Estructura de Archivos

- **`index.html`:** Contiene el HTML principal del reproductor, incluyendo:
  - Estructura visual y elementos interactivos.

    - Configuraciones de las estaciones dentro de una etiqueta `<script>`.
- **`js/main.js`:** Alberga el código JavaScript que impulsa la funcionalidad del reproductor, incluyendo:
    - Gestión del reproductor de audio.
    - Obtención y actualización dinámica de información de canciones (usando la API local o web).
    - Renderizado de la lista de estaciones.
    - Control de elementos de la interfaz como botones, menús y modales.
- **`api.php`:** (Opcional) Script PHP que actúa como una API local para extraer metadatos de las transmisiones de radio, proporcionando información de las canciones.
- **`css/main.min.css`:** Define el estilo visual del reproductor, incluyendo el diseño, los colores y la tipografía.
- **`custom.css`:** Permite añadir estilos personalizados.
- **`assets/`:** Carpeta para almacenar imágenes, iconos y otros recursos visuales.

## 3. Configuración Detallada

### 3.1. Configuración de Estaciones (`index.html`)

Las estaciones de radio se configuran dentro de un bloque `<script>` en el archivo `index.html`, definiendo el objeto `window.streams.stations`. Cada estación es un objeto con las siguientes propiedades:

| **Propiedad** | **Descripción** |
|---|---|
| `name` | Nombre de la estación que se muestra en la interfaz. |
| `hash` | Identificador único para la estación. |
| `description` | Breve descripción de la estación. |
| `logo` | Ruta al archivo de imagen del logotipo de la estación. |
| `album` | Ruta a una imagen de “álbum” predeterminada para mostrar antes de que se cargue la carátula real. |

| Propiedad | Descripción |
| --- | --- |
| `cover` | Ruta a la carátula de la canción que se está reproduciendo actualmente. |
| `api` | **(Opcional)** URL de la API local (`api.php`) configurada para obtener información de la estación. Debe incluir la `stream_url` como parámetro. Si se deja en blanco, el script utilizará la API web preconfigurada en `js/main.js`. |
| `stream_url` | URL del flujo de audio de la estación. |
| `tv_url` | URL del flujo de vídeo en directo de la estación (opcional). |
| `server` | Define la plataforma de música (“spotify” o “itunes”) que se utilizará para obtener información adicional (si la API correspondiente está en uso). |
| `program` | Objeto que contiene información sobre el programa actual (opcional). |
| `social` | Objeto con enlaces a los perfiles de redes sociales de la estación (opcional). |
| `apps` | Objeto con enlaces para descargar las aplicaciones de la estación (opcional). |

**Ejemplo de Configuración:**

```
<script>
window.streams = {
```

```
    timeRefresh: 10000, // Tiempo de actualización en
        milisegundos
    stations: [
        {
            name: "Ejemplo FM",
            hash: "ejemplofm",
            description: "¡La mejor música!",
            logo: "assets/ejemplofm_logo.png",
            album: "assets/default_album.jpg",
            cover: "assets/default_album.jpg",
            api: "api.php?url=https://example.com/
        stream", // API local (opcional)
            stream_url: "https://example.com/stream",
            server: "itunes", // Usar API de iTunes
            social: {
                facebook: "https://facebook.com/ejemplofm",
                instagram: "https://instagram.com/ejemplofm"
            }
        },
        // ... más estaciones
    ]
};
</script>
```

### 3.2. API Local (Opcional)

Si decide utilizar la API local (`api.php`), siga estas instrucciones para configurarla:

- **Configuración:**
  - En el archivo `api.php`, la variable `$allowedUrls` debe enumerar todas las URL de transmisión permitidas.
- **Funcionalidad:**
  - `getMp3StreamTitle()`: Extrae el título de la canción de los metadatos de la transmisión.
  - `extractArtistAndSong()`: Separa el artista y el título de la canción.
  - `getAlbumArt()`: Obtiene la carátula del álbum (actualmente configurado para utilizar la API de iTunes).
  - `updateHistory()`: Mantiene un historial de las canciones reproducidas.

**Nota:**

Si el campo `api` se deja en blanco en la configuración de la estación, `js/main.js` utilizará la API web preconfigurada. Asegúrese de que la API web

que está utilizando funciona correctamente y está configurada correctamente en el código JavaScript.

## 4. Interfaz de Usuario e Interacción

### 4.1. Elementos Clave

- **Encabezado:** Muestra el logotipo de la estación y botones para acceder al historial, la lista de estaciones y el menú móvil.
- **Sección del Reproductor:** Contiene la carátula del álbum, información de la canción (artista y título), controles de reproducción (reproducir/pausar, siguiente/anterior estación) y control de volumen.
- **Visualizador:** Un visualizador de audio simple que responde dinámicamente a la música.
- **Barra Lateral Fuera del Lienzo:**
  - **Lista de Estaciones:** Muestra todas las estaciones disponibles con miniaturas.
  - **Historial:** Muestra un historial de las canciones reproducidas recientemente.
- **Modal de Letras:** Muestra la letra de la canción que se está reproduciendo actualmente (si está disponible a través de la API de Vagalume).

### 4.2. Navegación

- **Selección de Estación:** Haga clic en una estación de la lista de estaciones para iniciar la reproducción.
- **Historial de Canciones:** Acceda al historial a través del botón del encabezado.
- **Letras de Canciones:** Haga clic en el botón “Letras” para abrir el modal de letras.
- **Menú Móvil:** El botón de menú del encabezado proporciona acceso a la misma funcionalidad en dispositivos móviles.

## 5. Personalización

### 5.1. Estilos Visuales (`css/main.min.css` y `custom.css`)

- Los colores, las fuentes, el espaciado, los tamaños de los elementos y otras propiedades visuales se pueden personalizar editando las reglas CSS.

### 5.2. Imágenes e Iconos (`assets/`)

- Sustituya las imágenes predeterminadas de la carpeta `assets` por las suyas propias para personalizar el logotipo de la estación, las carátulas de los álbumes y los iconos.

## 6. Publicación

1. Asegúrese de que la API local (`api.php`), si se utiliza, está configurada correctamente y es accesible en su servidor.
2. Suba todos los archivos y carpetas (HTML, CSS, JavaScript, PHP, imágenes) a su servidor web.

## 7. Consideraciones Adicionales

- **Derechos de Autor:** Asegúrese de tener los derechos para utilizar todas las imágenes, la música y demás contenido utilizado en su reproductor de radio.
- **Metadatos de la Transmisión:** La precisión de la información de las canciones depende de la calidad de los metadatos proporcionados por la transmisión de la emisora de radio.

# Documentação Abrangente do Reprodutor de Rádio Multi-Estações

Este documento detalha a estrutura, configuração e personalização de um reprodutor de rádio multi-estações construído com HTML, CSS e JavaScript. O reprodutor é capaz de buscar informações de músicas dinamicamente e oferece a flexibilidade de usar uma API local ou uma versão web pré-configurada.

## 1. Visão Geral

Este reprodutor de rádio proporciona uma interface amigável para os usuários desfrutarem de rádios online. Ele permite a adição de múltiplas estações, cada uma com sua própria transmissão ao vivo, informações de músicas, links de mídia social e muito mais. A configuração das estações é realizada diretamente no HTML, simplificando o processo de personalização.

## 2. Estrutura de Arquivos

- **`index.html`:** Contém o HTML principal do player, incluindo:
  - Estrutura visual e elementos interativos.
  - Configurações das estações dentro da tag `<script>`.
- **`js/main.js`:** Contém o código JavaScript que controla a funcionalidade do reprodutor, incluindo:
  - Gerenciamento do player de áudio.
  - Busca e atualização dinâmica de informações de músicas (usando API local ou web).
  - Renderização da lista de estações.
  - Controle de elementos da interface, como botões, menus e modais.
- **`api.php`:** (Opcional) Script PHP que atua como uma API local para extrair metadados de transmissões de rádio, fornecendo informações de músicas.

- **`css/main.min.css`:** Define o estilo visual do reprodutor, incluindo layout, cores e tipografia.
- **`custom.css`:** Permite a adição de estilos personalizados.
- **`assets/`:** Pasta para armazenar imagens, ícones e outros recursos visuais.

## 3. Configuração Detalhada

### 3.1. Configuração das Estações (`index.html`)

As estações de rádio são configuradas dentro de um bloco `<script>` no arquivo `index.html`, definindo o objeto `window.streams.stations`. Cada estação é um objeto com as seguintes propriedades:

| **Propriedade** | **Descrição** |
|---|---|
| `name` | Nome da estação, exibido na interface. |
| `hash` | Identificador único da estação. |
| `description` | Breve descrição da estação. |
| `logo` | Caminho para o arquivo de imagem do logotipo da estação. |
| `album` | Caminho para uma imagem padrão de “álbum” a ser exibida antes da capa real ser carregada. |
| `cover` | Caminho para a imagem de capa da música atual. |
| `api` | **(Opcional)** URL da API local (`api.php`) configurada para buscar informações da estação. Deve incluir a `stream_url` como parâmetro. Se este campo for deixado em branco, o script usará a API web pré-configurada em `js/main.js`. |

| Propriedade | Descrição |
| --- | --- |
| `stream_url` | URL do stream de áudio da estação. |
| `tv_url` | URL do stream de vídeo ao vivo da estação (opcional). |
| `server` | Define a plataforma de música (“spotify” ou “itunes”) usada para buscar informações adicionais (se a API correspondente estiver em uso). |
| `program` | Objeto contendo informações do programa em andamento (opcional). |
| `social` | Objeto com links para as redes sociais da estação (opcional). |
| `apps` | Objeto com links para download dos aplicativos da estação (opcional). |

**Exemplo de Configuração:**

```
<script>
window.streams = {
    timeRefresh: 10000, // Tempo de atualização em milissegundos
    stations: [
        {
            name: "Exemplo FM",
            hash: "exemplofm",
            description: "A melhor música!",
            logo: "assets/exemplofm_logo.png",
            album: "assets/default_album.jpg",
            cover: "assets/default_album.jpg",
            api: "api.php?url=https://exemplo.com/
        stream", // API local (opcional)
            stream_url: "https://exemplo.com/stream",
            server: "itunes", // Usar API do iTunes
            social: {
```

```
                facebook: "https://facebook.com/exemplofm",
                instagram: "https://instagram.com/exemplofm"
            }
        },
        // ... outras estações
    ]
};
</script>
```

### 3.2. API Local (Opcional)

Se você optar por usar a API local (`api.php`), siga as instruções abaixo para configurá-la:

- **Configuração:**
  - No arquivo `api.php`, a variável `$allowedUrls` deve conter todas as URLs de stream autorizadas.
- **Funcionalidades:**
  - `getMp3StreamTitle()`: Extrai o título da música dos metadados do stream.
  - `extractArtistAndSong()`: Separa artista e título da música.
  - `getAlbumArt()`: Busca a capa do álbum (atualmente configurado para usar a API do iTunes).
  - `updateHistory()`: Mantém um histórico das músicas tocadas.

**Observação:**

Se o campo `api` for deixado em branco na configuração da estação, o `js/main.js` usará a API web pré-configurada. Assegure-se de que a API web que você está utilizando esteja funcionando e configurada corretamente no código JavaScript.

## 4. Personalização, Interface, Interação e Publicação

As seções sobre:

- **Personalização de estilos visuais** (`css/main.min.css` e `custom.css`)
- **Uso de imagens e ícones personalizados** (`assets/`)
- **Elementos da interface do usuário** (cabeçalho, seletor de estação, histórico, etc.)
- **Navegação e interação do usuário**
- **Processo de publicação do reprodutor em um servidor web**

### 4.1. Elementos Principais

- **Cabeçalho:** Exibe o logotipo da estação e botões para acessar o histórico, lista de estações e menu mobile.

- **Seção do Player:** Contém a arte da capa do álbum, informações da música (artista e título), controles de reprodução (reproduzir/pausar, avançar/retroceder estação) e controle de volume.
- **Visualizador:** Um visualizador de áudio simples que responde dinamicamente à música.
- **Menu Lateral (Offcanvas):**
  - **Lista de Estações:** Mostra todas as estações disponíveis com miniaturas.
  - **Histórico:** Exibe um histórico das músicas tocadas recentemente.
- **Modal de Letras:** Exibe a letra da música (se disponível através da API do Vagalume).

### 4.2. Navegação

- **Seleção de Estação:** Clique em uma estação na lista de estações para iniciar a reprodução.
- **Histórico de Músicas:** Acesse o histórico através do botão no cabeçalho.
- **Letras da Música:** Clique no botão “Lyrics” para abrir o modal de letras.
- **Menu Mobile:** O menu no cabeçalho oferece acesso às mesmas funcionalidades em dispositivos móveis.

## 5. Personalização

### 5.1. Estilos Visuais (`css/main.min.css` e `custom.css`)

- As cores, fontes, espaçamento, tamanhos dos elementos e outras propriedades visuais podem ser personalizadas editando as regras CSS.

### 5.2. Imagens e Ícones (`assets/`)

- Substitua as imagens padrão na pasta `assets` por suas próprias imagens para personalizar o logotipo da estação, capas de álbuns e ícones.

## 6. Publicação

1. Certifique-se de que a API local (`api.php`) esteja configurada corretamente e acessível no seu servidor.
2. Faça o upload de todos os arquivos e pastas (HTML, CSS, JavaScript, PHP, imagens) para o seu servidor web.

## 7. Considerações Adicionais

- **Direitos Autorais:** Assegure-se de ter os direitos de uso para todas as imagens, músicas e outros conteúdos utilizados no reprodutor.
- **Metadados do Stream:** A precisão das informações da música depende da qualidade dos metadados fornecidos pela estação de rádio.