H5 Games Ads

H5 Games Ads let you grow your earnings by showing ads in your HTML5 (H5) games. H5 games can run in web browsers and also in WebViews on native apps. H5 Games Ads use the new H5 Games Ad Placement API. After you integrate the API into your H5 games, we show ads at the best moments for your users.

Benefits

H5 Games Ads offer:

High-performing formats

- **Interstitials**: Full-screen ads that are displayed at natural breaks in your game, such as between levels. Users can choose to either click these ads or return to your game.
- **Rewarded**: Ads that users can choose to interact with in exchange for in-game rewards.

Cross-platform support

- H5 Games Ads support embedding your game into a web page directly, as well as iframing your game into a different site.
- H5 Games Ads also support app integration with AdMob, so that you can monetize your game with app ads when your H5 game is hosted in the webview of a mobile app (Android support only).

With H5 Games Ads, your games will be able to show ads on your site, on a webpage, or within an app WebView — all from a single API.

Guides

Follow this **Setup** guide to get started using our extension.

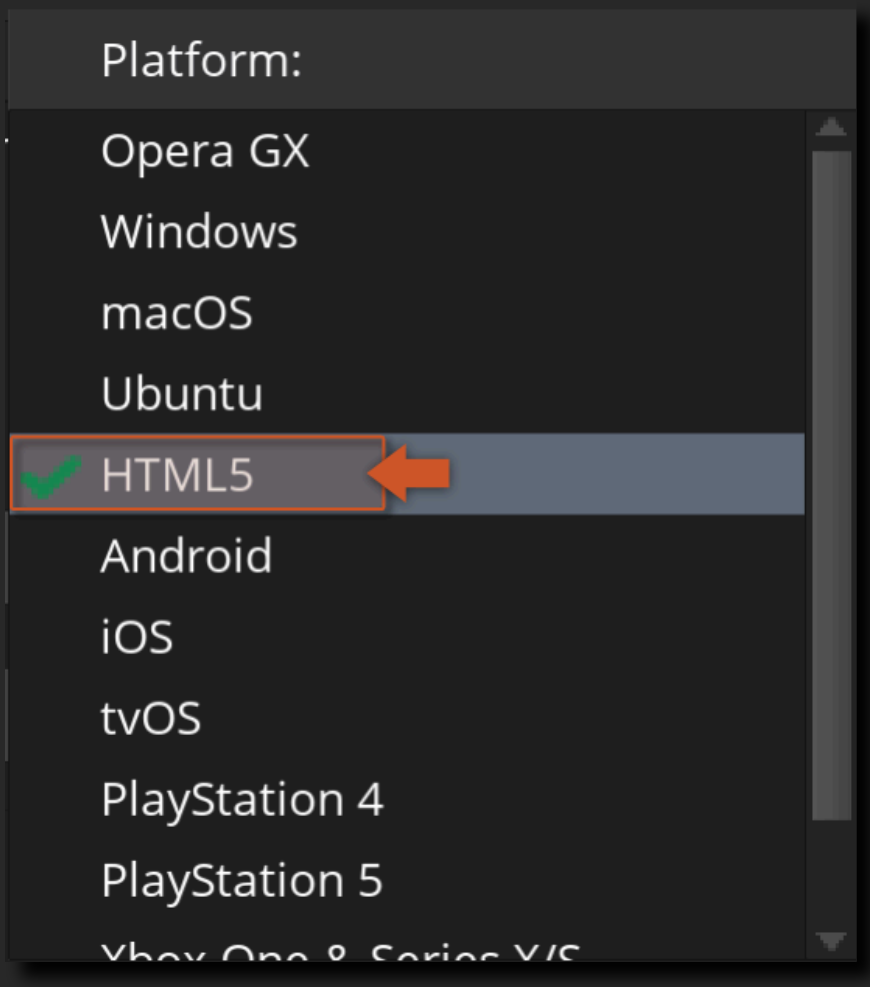
Functions

These functions are provided to interact with the H5 Games Ads API:

- H5GamesAds_Interstitial
- H5GamesAds_RewardVideo
- H5GamesAds_Sound

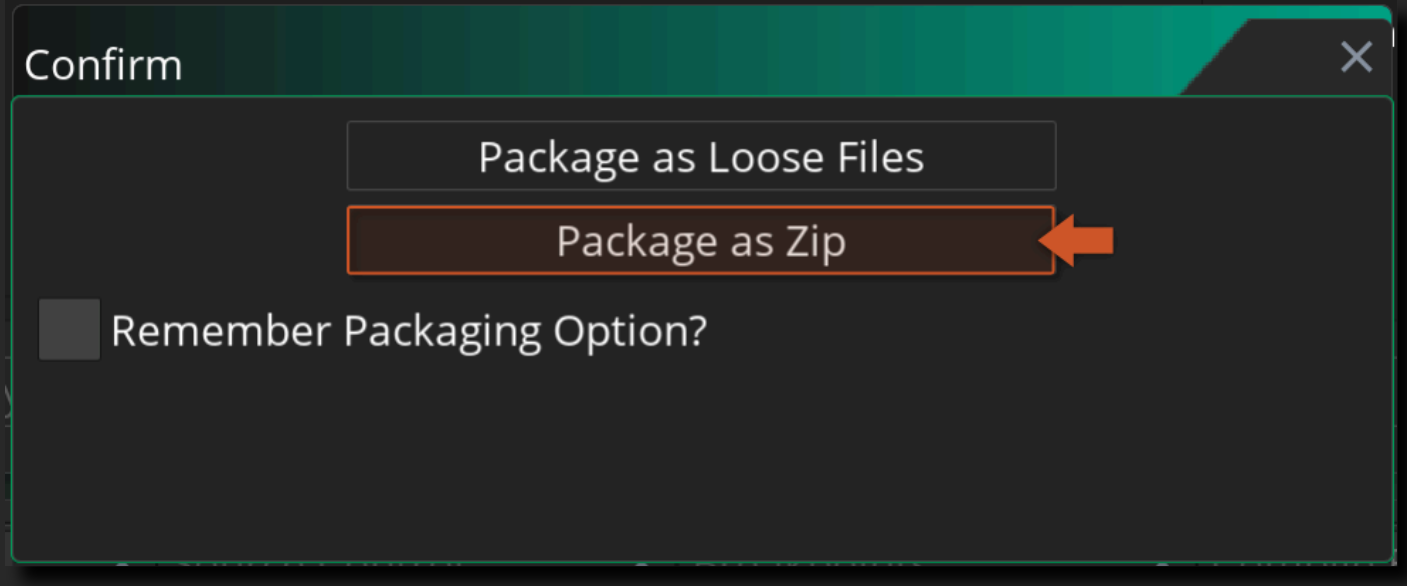
Setup

1. Select HTML5 export

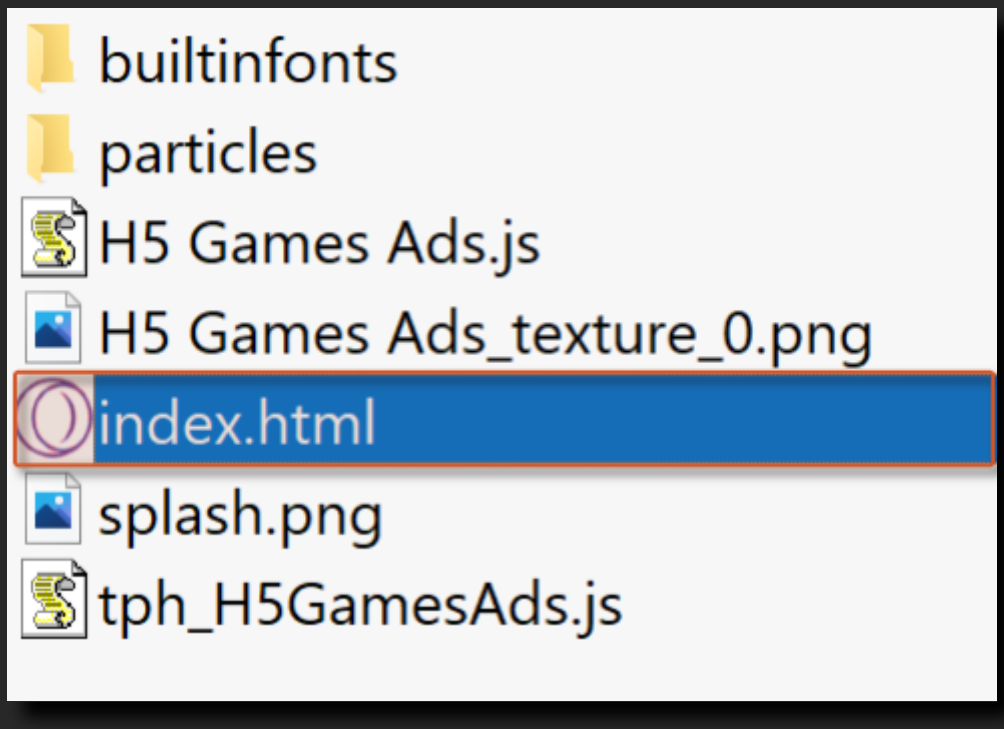


2. Build your project from within the IDE.

3. Click the `Package as Zip` option and select an exporting path.



4. Locate the exported zip file, unpack it and move the `index.html` to file to the included files of your project.



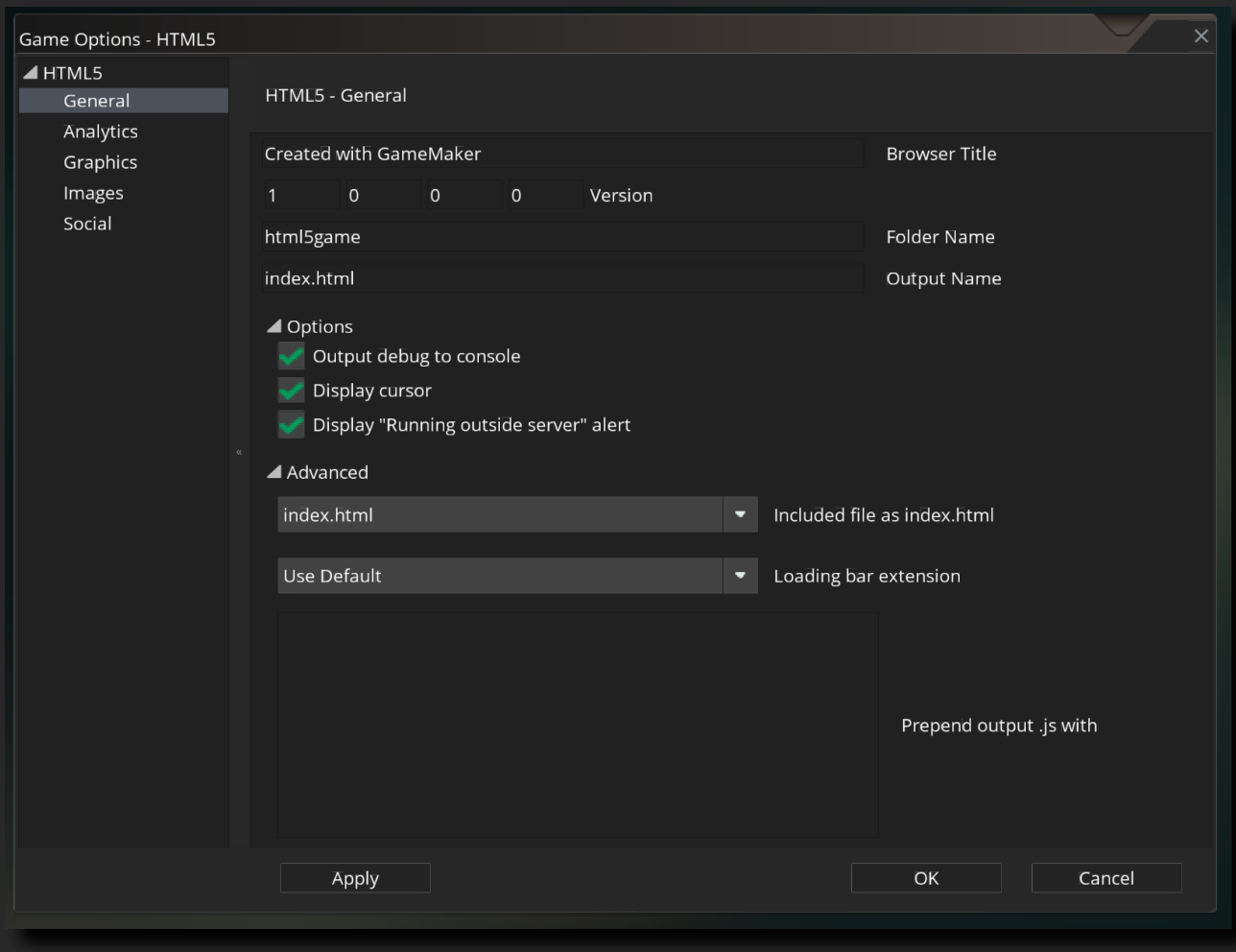
5. Open the index and paste the following code between `</head>` and `<body>`:

```
    <script async
        data-ad-frequency-hint="30s"
        data-adbreak-test="on"
        src="https://pagead2.googlesyndication.com/pagead/js/
adsbygoogle.js?client=APPID"
        crossorigin="anonymous">
    </script>

    <script>
       window.adsbygoogle = window.adsbygoogle || [];
       var adBreak = adConfig = function(o) {adsbygoogle.push(o);}
       adConfig({preloadAdBreaks: 'on'});
    </script>
```

NOTE Replace "APPID" with your actual application ID.

6. Go to `Game Options -> Platform Settings -> HTML5 -> Advanced` and select the new index.html:



7. Your project is now ready to use the H5 Game Ads extension.

H5GamesAds_Interstitial

This function defines a placement where an ad *could* show—an *opportunity* to show an ad. Whether an ad *actually* shows will depend on a number of things, including:

- The type of ad placement that you declared
    - Is this ad at the start of the game? Between levels? At a moment when the player has paused the game? etc.
- Whether a suitable ad exists for the current player, that is:
    - relevant to them
    - consistent with their data privacy and consent settings
- How many ads the player has seen recently
- The control settings (e.g. ad frequency) you have configured for this game:
    - either as hints in the tag OR
    - within AdSense (note the controls available in AdSense will evolve over time)

The precise type that you use is important. As it can directly affect the kind of ad experience that will be shown (an interstitial versus a rewarded ad). But it can also have other less visible effects on behavior of the API. It's important that you specify the placement type correctly so the API can work optimally.

Placement Type	Description
`'start'`	Your game has loaded, the UI is visible and sound is enabled, the player can interact with the game, but the game play has not started yet.
`'pause'`	The player pauses the game.
`'next'`	The player navigates to the next level.
`'browse'`	The player explores options outside of the game play.

NOTE it's important to realize that a call to H5GamesAds_Interstitial may not show an ad at all. It simply declares a place where an ad *could* be shown.

Syntax:

```
H5GamesAds_Interstitial(placement_type);
```

Argument	Type	Description
placement_type	string	State of your game when the ad will be shown.

Returns:

```
undefined
```

Triggers:

```
Asynchronous Social Event
```

async_load Contents		
Key	**Type**	**Description**
type	string	One of the following `"H5GamesAds_Interstitial_BreakDone"` - is always triggered; `"H5GamesAds_Interstitial_BeforeAd"` - only triggered if ad is shown; `"H5GamesAds_Interstitial_AfterAd"` - only triggered if ad is shown.
breakType	string	The type of break (passed to the function); see above.
breakStatus	string	The status of the ad; `"noAdPreloaded"`, `"viewed"`, `"frequencyCapped"`
breakFormat	string	The format of the ad; `"interstitial"`

Extended Example:

```
H5GamesAds_Interstitial("next");
```

With this function call we are defining the placement of an interstitial ad. The callback event can now be caught using the **Social Async Event**, with this example code:

```
switch(async_load[?"type"])
{
    case "H5GamesAds_Interstitial_BreakDone":
        var breakType = async_load[?"breakType"] // "start", "pause", "next",
"browse"
        var breakStatus = async_load[?"breakStatus"] // "noAdPreloaded", "viewed",
"frequencyCapped"
        var breakFormat = async_load[?"breakFormat"] // "interstitial"
    break

    case "H5GamesAds_Interstitial_BeforeAd":
    break

    case "H5GamesAds_Interstitial_AfterAd":
    break
}
```

The function call triggers a bunch of events for each phase it is on. The event type `"H5GamesAds_Interstitial_BreakDone"` is always triggered even if the ad was not displayed and contains additional information.

H5GamesAds_RewardVideo

This function defines a reward video placement, for requesting and displaying a reward ad.

NOTE It's important to realize that a call H5GamesAds_RewardVideo to may not show an ad at all. It simply declares a place where an ad *could* be shown.

NOTE If no video can be loaded this can result in an interstitial ad being shown.

Syntax:

```
H5GamesAds_RewardVideo();
```

Returns:

```
undefined
```

Triggers:

```
Asynchronous Social Event
```

async_load Contents		
Key	**Type**	**Description**
type	string	One of the following `"H5GamesAds_RewardVideo_BreakDone"` - is always triggered; `"H5GamesAds_RewardVideo_BeforeAd"` - only triggered if ad is shown; `"H5GamesAds_RewardVideo_AfterAd"` - only triggered if ad is shown; `"H5GamesAds_RewardVideo_BeforeReward"`; `"H5GamesAds_RewardVideo_AdDismissed"` - only triggered if ad was dismissed; `"H5GamesAds_RewardVideo_AdViewed"` - only

		triggered if ad was watched completly (should reward).
breakType	string	The type of break (passed to the function); see above.
breakStatus	string	The status of the ad; `"noAdPreloaded"`, `"viewed"`, `"frequencyCapped"`
breakFormat	string	The format of the ad; `"interstitial"`, `"reward"`

Extended Example:

```
H5GamesAds_RewardVideo();
```

With this function call we are defining the placement of a reward video ad. The callback event can now be caught using the **Social Async Event**, with this example code:

```
switch(async_load[?"type"])
{
    case "H5GamesAds_RewardVideo_BreakDone":
        var breakType = async_load[?"breakType"]//"reward"
        var breakStatus = async_load[?"breakStatus"] // "noAdPreloaded", "viewed",
"frequencyCapped"
        var breakFormat = async_load[?"breakFormat"] // "reward", "interstitial"
    break

    case "H5GamesAds_RewardVideo_BeforeAd":
    break

    case "H5GamesAds_RewardVideo_AfterAd":
    break

    case "H5GamesAds_RewardVideo_BeforeReward":
    break

    case "H5GamesAds_RewardVideo_AdDismissed":
    break

    case "H5GamesAds_RewardVideo_AdViewed":
       show_debug_message("Video Reward ->  Reward")
    break
}
```

The function call triggers a bunch of events for each phase it is on. The event type `"H5GamesAds_RewardVideo_BreakDone"` is always triggered even if the ad was not displayed and contains additional information.

H5GamesAds_Sound

Enable or disable sound in the advertisements. The default value is sound on.

Syntax:

```
H5GamesAds_Sound(sound);
```

Argument	Type	Description
sound	bool	Whether the ads should play sound. true: sound; false: mute;

Returns:

```
N/A
```

Example:

```
H5GamesAds_Sound(false);
```

The code sample above will mute the future ads being played.