SWAG API DEVELOPERS GUIDE

Include the following files:

```
<script type="text/javascript"
src="https://swagapi.shockwave.com/dist/swag-api.js">
<link rel="stylesheet" type="text/css"
href="https://swagapi.shockwave.com/dist/swag-api.css">
```

Score Configuration

Each type of score for your game can be configured individually. Expressed in JSON, a score configuration is in the following format:

```
{
  game: String,
  name: String,
  level_key: String,
  value_name: String,
  value_type: String,
  value_formatter: String,
  order: Number,
  reverse: Boolean,
  mode: String
}
```

attribute	required	default	type	description
game	y	-	String	api key of your game
name	y	-	String	Display name for the score *eg. 'Level 1'*
level_key	y	-	String	reference key for this level *eg. 'level1'*
value_name	n	-	String	Name for the score values *eg. 'Fastest Time'*
value_type	n	"number"	String	The type of value (number, time)
value_formatter	n	"default"	String	The type of formatter to use for this value (see Formatters)
order	n	-	Number	The display order of this level
reverse	n	false	Boolean	if true, minimum scores value are used for api score calculations
mode	n	"default"	String	Scores with mode of `first` will only display the first score for a day in the leaderboards

These configurations are loaded into our highscore system to facilitate the specific needs of each type of score for your game.

Formatters

Define a value_formatter in score configuration or as a parameter in api methods to format values.

Number Formatters

No special number formatters are currently available. Let us know if there is a format you'd like to see.

Time Formatters

formatter	example output
default	00:01:05.5
shortDuration	1m 5.5s
longDuration	1 minute, 5.5 seconds
seconds	65.5s
ms	65500

Connecting to the API

SWAGPI will be accessible as a global

```
var api = SWAGAPI.getInstance({
  wrapper: wrapper,
  api_key: '5c6c3c056917a692f96f9651',
  theme: 'shockwave',
  debug: true
});
```

API options:

option	type	description
wrapper	domElement	the domElement containing the game
api_key	String	unique identifier for the game
theme	String	set the theme for api ui elements
debug	boolean	enable debug console messages

The client must use the `startSession` method to start using the api and wait for the promise to resolve or the SESSION_READY event before using any other api calls.

Using promise:

```
return api.startSession()
  .then(function() {
      //do stuff
  });
```

Using event listener:

```
api.on('SESSION_READY', function() {
  //do stuff
});
```

Using the API

Session Methods

method	parameters	description	method result
startSession	-	Used start an api session. The client must wait for the promise to resolve or the SESSION_READY event before using any other api calls.	Promise
startGame	-	Call this method before the player starts a game "session"	Promise
endGame	options	Call this method at the end of a player game "session"	Promise

End game options

The endGame method accepts an options object which enables developers to submit custom metrics for a game session.

Example use cases:

Single level runner game (without win condition):

```
{ feet: 134 }
```

Puzzle with countdown timer:

```
{ success: true }
```

Monster shooter with multiple types, tracking which enemy killed the player:

```
{ werewolf: 4, gargoyle: 2, ghoul: 12, killedBy: "ghoul" }
```

Start and End Game Usage

The startGame and endGame methods are intended to be used at the start and end of a game session.

This will enable game developers to view game metrics like average session time and number of play sessions in the developer dashboard.

The API may display UI elements when these methods are invoked such as interstitial ads or site promotions. When the display of these UI elements has completed, the promise will resolve.

Example use case of startGame:

```
- Player uses a start button
- startGame() method is invoked and returns a promise
- when the promise is resolved, game play starts
```

Example use case of endGame:

```
- Player is defeated in a game
- endGame() method is invoked and returns a promise
- when the promise is resolved, game displays a retry button
```

Score Methods

method	parameters	description	method result
getScoreCategories	-	Returns a json array of highscore categories associated with this game	Promise, resolves json
getScores	see getScores options	Returns a json array of scores based on the options objects	Promise, resolves json
postScore	level_key, value, options	Post the score `value` for the `level_key` for the current user.	Promise
postDailyScore	day, level_key, value	Post the score `value` for the `level_key` and `day` for the current user.	Promise

Score Methods Options

postScore options

The following options are available:

option	type	description
confirmation	boolean	If this option is included, api will display a confirmation dialog after the score is posted

example:

```
api.postScore('level_1', 400, { confirmation: true });
```

getScores options

The following options are available:

option	required	description
level_key	yes	The level to retrieve scores for.
type	no	String, *default:* `"standard"`, *values:* `"standard", "weekly"`
period	no	String, *default:* `"alltime"`, *values:* `"daily", "weekly", "monthly", "alltime"`
current_user	no	boolean, *default:* `false`. If true, only get scores for current user
target_date	no	epoch time or ISO Date string in format YYYY-MM-DD. Use this date as the base for date ranges eg. "daily", "weekly"
use_daily	no	boolean, use score 'day' instead of score post date
value_formatter	no	Format scores using specified formatter

Example (scores this week on level1 for the current user):

```
return api.getScores({
  type: 'weekly',
  level_key: 'level1',
  current_user: true
})
  .then(function(scores) {
    //do something
  });
```

UI Methods

method	parameters	description	method result
showDialog	type	display a dialog of type `scores`, `dailyscores` `achievements` or `weeklyscores` (see dialog options for more information)	-
showAd	-	Displays an ad	Promise

showDialog options

example:

```
api.showDialog('scores', {
    title: 'Best Scores',
    level_key: 'level1',
    period: 'alltime',
    value_formatter: ''
});
```

The following options are available:

option	description
title	Overrides the title in the dialog
level_key	Sets the default level_key in the select
period	Sets the default period in the select
value_formatter	Overrides the formatter used in the score config

Achievement Methods

method	parameters	description	method result
getAchievementCategories	-	Return a json array of achievements associated with this game	Promise, resolves json
postAchievement	achievement_key	Post an achievement `achievement_key` for the current user	Promise
getUserAchievements	-	Return a list of all achievements by the current user for this game	Promise, resolves json

Data store Methods

method	parameters	description	method result
postDatastore	key, value	Post a `value` to `key`. If 'key' exists for this user, it will be overwritten.	Promise
getUserDatastore	-	Returns a json array of all data store objects associated with this user	Promise

Misc Methods

method	parameters	description	method result
isSubscriber	-	returns true if the current user is a subscriber	Promise, resolves Boolean
hasDailyScore	level_key	returns true if the current user has submitted a score today	Promise, resolves Boolean
getCurrentDay	-	returns the current "day" used by the api (PST timezone). format: `{"day":"2019-05-09"}`. If the url parameters `day`, `month`, and `year` are present, this method will return this date rather than the current date. eg. `day=04&month=07&year=19`	Promise, resolves json

Branding Methods

method	parameters	description	method result
getBrandingLogo	-	returns an HTMLImageElement of the appropriate site logo	Promise, resolves HTMLImageElement
getBrandingLogoUrl	-	returns the url of the appropriate site logo	Promise, resolves a string
SWAGAPI.showBrandingAnimation	elementid, callback	displays a branding animation in the provided element id	Promise

The method `SWAGAPI.showBrandingAnimation` can be used to display the branding animation before a game. Note this is a static method so it can be used independently of the API instance.

example:

```
SWAGAPI.showBrandingAnimation('game')
  .then(function() {
    //display the game
  });
```

example (using callback):

```
SWAGAPI.showBrandingAnimation('game', function() {
  //display the game
});
```

External Component Integration

method	parameters	description	method result
postExternalMessage	message	send a message to an external component	-

example:

```
api.postExternalMessage({
    type: 'lobby',
    data: [
      { username: 'user1'},
      { username: 'user2'},
      ...
    ]
});
```

message types:

Leaderboard

```
{
  type: 'leaderboard',
  data: [
    { username: 'user1', points: 300},
    { username: 'user1', points: 300},
    ...
  ]
}
```

Lobby

```
{
    type: 'lobby',
    data: [
      { username: 'user1'},
      { username: 'user2'},
      ...
    ]
}
```

Waiting

```
{
    type: 'waiting'
}
```

API Events:

event	description
SESSION_READY	The api is ready to use
ERROR	An api error has occurred
DIALOG_CLOSED	The active dialog has closed

Demo

There is a simple demo of the api at:

https://swagapi.shockwave.com/demo.html (view source)

Developing and Using Locally

For now you will need to edit your etc\hosts file.

On Windows you can find at c:windows\system32\drivers\etc\host

and add the following line

```
127.0.0.1            local.shockwave.com
```

Install http-server

```
npm install -g http-server
```

Launch a local web server where your index.html is

```
http-server -p 8888
```

launch your game with

```
http://local.shockwave.com:8888
```

Notes:

- You can either be logged into shockwave.com or as a guest when testing.
- To get an API key or anything else, please contact your support at Addicting Games.

VERSION 2.0.4