A BEAUTIFUL REPLACEMENT FOR JAVASCRIPT'S "ALERT"

SO... WHAT DOES IT DO?

Here's a comparison of a standard error message. The first one uses the built-in **alert**-function, while the second is using **sweetAlert**.

Normal alert

Show error message

CODE:

```
alert("Oops... Something went wrong!");
```

Show error message

CODE:

```
sweetAlert("Oops...", "Something went
wrong!", "error");
```

Pretty cool huh? SweetAlert automatically centers itself on the page and looks great no matter if you're using a desktop computer, mobile or tablet. It's even highly customizeable, as you can see below!

MORE EXAMPLES

In these examples, we're using the shorthand function **swal** to call sweetAlert.

A basic message

Try me!

```
swal("Here's a message!")
```

A title with a text under

Try me!

```
swal("Here's a message!", "It's pretty,
isn't it?")
```

A success message!

Try me!

```
swal("Good job!", "You clicked the
button!", "success")
```

A warning message, with a function attached to the "Confirm"-button...

Try me!

```
swal({
  title: "Are you sure?",
  text: "You will not be able to recover
this imaginary file!",
  type: "warning",
  showCancelButton: true,
  confirmButtonColor: "#DD6B55",
  confirmButtonText: "Yes, delete it!",
  closeOnConfirm: false
},
function(){
  swal("Deleted!", "Your imaginary file has
been deleted.", "success");
});
```

... and by passing a parameter, you can execute something else for "Cancel".

Try me!

```
swal({
  title: "Are you sure?",
  text: "You will not be able to recover
this imaginary file!",
  type: "warning",
  showCancelButton: true,
  confirmButtonColor: "#DD6B55",
  confirmButtonText: "Yes, delete it!",
  cancelButtonText: "No, cancel plx!",
  closeOnConfirm: false,
  closeOnCancel: false
},
function(isConfirm){
  if (isConfirm) {
    swal("Deleted!", "Your imaginary file
has been deleted.", "success");
  } else {
    swal("Cancelled", "Your imaginary file
is safe :)", "error");
  }
});
```

A message with a custom icon

Try me!

```
swal({
  title: "Sweet!",
  text: "Here's a custom image.",
  imageUrl: "images/thumbs-up.jpg"
});
```

An HTML message

Try me!

```
swal({
  title: "HTML <small>Title</small>!",
  text: "A custom <span
style="color:#F8BB86">html<span> message.",
  html: true
});
```

A message with auto close timer

Try me!

```
swal({
  title: "Auto close alert!",
  text: "I will close in 2 seconds.",
  timer: 2000,
  showConfirmButton: false
});
```

A replacement for the "prompt" function

Try me!

```
swal({
  title: "An input!",
  text: "Write something interesting:",
  type: "input",
  showCancelButton: true,
  closeOnConfirm: false,
  animation: "slide-from-top",
  inputPlaceholder: "Write something"
},
function(inputValue){
  if (inputValue === false) return false;

  if (inputValue === "") {
    swal.showInputError("You need to write
something!");
    return false
  }

  swal("Nice!", "You wrote: " + inputValue,
"success");
});
```

With a loader (for AJAX request for example)

Try me!

```
swal({
  title: "Ajax request example",
  text: "Submit to run ajax request",
  type: "info",
  showCancelButton: true,
  closeOnConfirm: false,
  showLoaderOnConfirm: true,
},
function(){
  setTimeout(function(){
    swal("Ajax request finished!");
  }, 2000);
});
```

You can also change the theme of SweetAlert!

Try me!

```
<link rel="stylesheet" type="text/css"
href="dist/sweetalert.css">
<link rel="stylesheet" type="text/css"
href="themes/twitter.css">
```

DOWNLOAD & INSTALL

Method 1: Install through bower:

```
$ bower install sweetalert
```

Method 2: Install through NPM:

```
$ npm install sweetalert
```

Method 3: Download the sweetAlert **CSS** and **JavaScript** files.

Download files

1. Initialize the plugin by referencing the necessary files:

```
<script src="dist/sweetalert.min.js"></script>
<link rel="stylesheet" type="text/css" href="dist/sweetalert.css">
```

2. Call the sweetAlert-function after the page has loaded

```
swal({
  title: "Error!",
  text: "Here's my error message!",
  type: "error",
  confirmButtonText: "Cool"
});
```

CONFIGURATION

Here are the keys that you can use if you pass an object into sweetAlert:

Argument	Description
title	The title of the modal. It can either be added to the object under the key "title" or passed as the first parameter of the function.
text	A description for the modal. It can either be added to the object under the key "text" or passed as the second parameter of the function.
type	The type of the modal. SweetAlert comes with 4 built-in types which will show a corresponding icon animation: "**warning**", "**error**", "**success**" and "**info"**. You can also set it as "**input**" to get a prompt modal. It can either be put in the object under the key "type" or passed as the third parameter of the function.
allowEscapeKey	If set to **true**, the user can dismiss the modal by pressing the Escape key.

customClass
A custom CSS class for the modal. It can be added to the object under the key "customClass".

allowOutsideClick
If set to **true**, the user can dismiss the modal by clicking outside it.

showCancelButton
If set to **true**, a "Cancel"-button will be shown, which the user can click on to dismiss the modal.

showConfirmButton
If set to **false**, the "OK/Confirm"-button will be hidden. Make sure you set a timer or set allowOutsideClick to true when using this, in order not to annoy the user.

confirmButtonText
Use this to change the text on the "Confirm"-button. If showCancelButton is set as true, the confirm button will automatically show "Confirm" instead of "OK".

confirmButtonColor
Use this to change the background color of the "Confirm"-button (must be a HEX value).

cancelButtonText
Use this to change the text on the "Cancel"-button.

closeOnConfirm
Set to *false* if you want the modal to stay open even if the user presses the "Confirm"-button. This is especially useful if the function attached to the "Confirm"-button is another SweetAlert.

closeOnCancel
Same as closeOnConfirm, but for the cancel button.

imageUrl
Add a customized icon for the modal. Should contain a string with the path to the image.

imageSize
If imageUrl is set, you can specify imageSize to describes how big you want the icon to be in px. Pass in a string with two values separated by an "x". The first value is the width, the second is the height.

timer
Auto close timer of the modal. Set in ms (milliseconds).

html
If set to true, will not escape title and text parameters. (Set to false if you're worried about XSS attacks.)

animation
If set to **false**, the modal's animation will be disabled. Possible (string) values : **pop** (default when animation set to true), **slide-from-top**, **slide-from-bottom**

inputType
Change the type of the input field when using **type: "input"** (this can be useful if you want users to type in their password for example).

inputPlaceholder	When using the input-type, you can specify a placeholder to help the user.
inputValue	Specify a default text value that you want your input to show when using **type: "input"**
showLoaderOnConfirm	Set to **true** to disable the buttons and show that something is loading.

METHODS

SweetAlert also comes with some simple methods that you can call:

Function	Description
setDefaults	If you end up using a lot of the same settings when calling SweetAlert, you can use setDefaults at the start of your program to set them once and for all!
close	Close the currently open SweetAlert programatically.
showInputError	Show an error message after validating the input field, if the user's data is bad
enableButtons, disableButtons	Disable or enable the user to click on the cancel and confirm buttons.

CONTRIBUTE

SweetAlert was created by **Tristan Edwards**, you can follow him on **Twitter** or **Dribbble** for updates and other cool projects!

Feel free to fork SweetAlert on **GitHub** if you have any features that you want to add!

- 2016