The immediate function pattern is a syntax that enables you to execute a function as soon as it is defined.

(function () {

alert('watch out!');

}());

• You define a function using a function expression. (A function declaration won’t work.)

• You add a set of parentheses at the end, which causes the function to be executed immediately.

• You wrap the whole function in parentheses (required only if you don’t assign the function to a variable).

[复制代码](javascript:void(0);)

(function () {

var days = ['Sun', 'Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat'],

today = new Date(),

msg = 'Today is ' + days[today.getDay()] + ', ' + today.getDate();

alert(msg);

}()); // "Today is Fri, 13"

[复制代码](javascript:void(0);)

If  this  code  weren’t  wrapped  in  an  immediate  function,  then  the  variables  days, today, and msg would all be global variables, leftovers from the initialization code.

**Parameters of an Immediate Function**

[复制代码](javascript:void(0);)

// prints:

// I met Joe Black on Fri Aug 13 2010 23:26:59 GMT-0800 (PST)

(function (who, when) {

console.log("I met " + who + " on " + when);

}("Joe Black", new Date()));

[复制代码](javascript:void(0);)

Commonly, the global object is passed as an argument to the immediate function so that it’s accessible inside of the function without having to use window.

(function (global) {

// access the global object via `global`

}(this));

Shouldn’t pass too many parameters to an immediate function, because it could quickly become a burden to constantly scroll to the top and to the bottom of the function to understand how it works.

**Returned Values from Immediate Functions**

var result = (function () {

return 2 + 2;

}());

use the scope of the immediate function to privately store some data, specific to the inner function you return.

[复制代码](javascript:void(0);)

var getResult = (function () {

var res = 2 + 2;

return function () {

return res;

};

}());

[复制代码](javascript:void(0);)

Used for Defining object properties

[复制代码](javascript:void(0);)

var o = {

message: (function () {

var who = "me",

what = "call";

return what + " " + who;

}()),

getMsg: function () {

return this.message;

}

};

// usage

o.getMsg(); // "call me"

o.message; // "call me"

[复制代码](javascript:void(0);)

**Benefits and Usage**

It helps you wrap an amount of work you want to do without leaving any global variables behind. All the variables you define will be local to the self-invoking functions and you don’t have to worry about polluting the global space with temporary variables.

Enables you to wrap individual features into self-contained modules.

[复制代码](javascript:void(0);)

// module1 defined in module1.js

(function () {

// all the module 1 code ...

}());

[复制代码](javascript:void(0);)