(function(global, factory){

if(….){……

}else{

factory(global);

}

}(typeof window !== "undefined" ? window : this, function( window, noGlobal ) {

……….

} ) );

global == typeof window !== "undefined" ? window : this

factory(global) == function( window, noGlobal ) {

……….

} )

function(window,noGlobal){

var

version = "1.11.0",

jQuery = function( selector, context ) { //创建一个实例对象无需使用new语法

return new jQuery.fn.init( selector, context );//以工厂模式返回一个对象

},

rtrim = /^[\s\uFEFF\xA0]+|[\s\uFEFF\xA0]+$/g,

rmsPrefix = /^-ms-/,

rdashAlpha = /-([\da-z])/gi,

fcamelCase = function( all, letter ) {

return letter.toUpperCase();

};

jQuery.fn = jQuery.prototype = {

//把jQuery原型赋值给jQuery.fn，通过它访问jQuery原型对象的属性和方法

//红色部分是jQuery原型对象的属性和方法

jquery: version,

constructor: jQuery, //字面量方式创建一个新的对象

selector: "",

length: 0,

toArray: function() {

return slice.call( this );

},

get: function( num ) {

return num != null ?

( num < 0 ? this[ num + this.length ] : this[ num ] ) :

slice.call( this );

},

pushStack: function( elems ) {

var ret = jQuery.merge( this.constructor(), elems );

ret.prevObject = this;

ret.context = this.context;

return ret;

},

each: function( callback, args ) {

return jQuery.each( this, callback, args );

},

map: function( callback ) {

return this.pushStack( jQuery.map(this, function( elem, i ) {

return callback.call( elem, i, elem );

}));

},

slice: function() {

return this.pushStack( slice.apply( this, arguments ) );

},

first: function() {

return this.eq( 0 );

},

last: function() {

return this.eq( -1 );

},

eq: function( i ) {

var len = this.length,

j = +i + ( i < 0 ? len : 0 );

return this.pushStack( j >= 0 && j < len ? [ this[j] ] : [] );

},

end: function() {

return this.prevObject || this.constructor(null);

},

push: push,

sort: deletedIds.sort,

splice: deletedIds.splice

};

}

