* Jump To … +  
    
  [<<< back to documentation](http://docs.google.com/index.html)  [base.js](http://docs.google.com/base.html)   [constraint.js](http://docs.google.com/constraint.html)   [defaults.js](http://docs.google.com/defaults.html)   [factory.js](http://docs.google.com/factory.html)   [field.js](http://docs.google.com/field.html)   [form.js](http://docs.google.com/form.html)   [main.js](http://docs.google.com/main.html)   [multiple.js](http://docs.google.com/multiple.html)   [pubsub.js](http://docs.google.com/pubsub.html)   [remote.js](http://docs.google.com/remote.html)   [ui.js](http://docs.google.com/ui.html)   [utils.js](http://docs.google.com/utils.html)   [validator.js](http://docs.google.com/validator.html)   [validator\_registry.js](http://docs.google.com/validator_registry.html)

# main.js

* [¶](#gjdgxs)  
  import $ from 'jquery';  
  import Utils from './utils';  
  import Defaults from './defaults';  
  import Base from './base';  
  import ValidatorRegistry from './validator\_registry';  
  import UI from './ui';  
  import Form from './form';  
  import Field from './field';  
  import Multiple from './multiple';  
  import Factory from './factory';  
    
  var vernums = $.fn.jquery.split('.');  
  if (parseInt(vernums[0]) <= 1 && parseInt(vernums[1]) < 8) {  
   throw "The loaded version of jQuery is too old. Please upgrade to 1.8.x or better.";  
  }  
  if (!vernums.forEach) {  
   Utils.warn('Parsley requires ES5 to run properly. Please include https://github.com/es-shims/es5-shim');  
  }
* [¶](#30j0zll)  
  Inherit on, off & trigger to Parsley:  
  var Parsley = Object.assign(new Base(), {  
   element: document,  
   $element: $(document),  
   actualizeOptions: null,  
   \_resetOptions: null,  
   Factory: Factory,  
   version: '@@version'  
   });
* [¶](#1fob9te)  
  Supplement Field and Form with Base This way, the constructors will have access to those methods  
  Object.assign(Field.prototype, UI.Field, Base.prototype);  
  Object.assign(Form.prototype, UI.Form, Base.prototype);
* [¶](#3znysh7)  
  Inherit actualizeOptions and \_resetOptions:  
  Object.assign(Factory.prototype, Base.prototype);

### [¶](#2et92p0) jQuery API $('.elem').parsley(options) or $('.elem').psly(options) $.fn.parsley = $.fn.psly = function (options) { if (this.length > 1) { var instances = []; this.each(function () { instances.push($(this).parsley(options)); }); return instances; }

* [¶](#tyjcwt)  
  Return undefined if applied to non existing DOM element  
   if (this.length == 0) {  
   return;  
   }  
    
   return new Factory(this[0], options);  
  };

### [¶](#3dy6vkm) Field and Form extension Ensure the extension is now defined if it wasn’t previously if ('undefined' === typeof window.ParsleyExtend) window.ParsleyExtend = {};

### [¶](#1t3h5sf) Parsley config Inherit from ParsleyDefault, and copy over any existing values Parsley.options = Object.assign(Utils.objectCreate(Defaults), window.ParsleyConfig); window.ParsleyConfig = Parsley.options; // Old way of accessing global options

### [¶](#4d34og8) Globals window.Parsley = window.psly = Parsley; Parsley.Utils = Utils; window.ParsleyUtils = {}; $.each(Utils, (key, value) => { if ('function' === typeof value) { window.ParsleyUtils[key] = (...args) => { Utils.warnOnce('Accessing `window.ParsleyUtils` is deprecated. Use `window.Parsley.Utils` instead.'); return Utils[key](...args); }; } });

### [¶](#2s8eyo1) Define methods that forward to the registry, and deprecate all access except through window.Parsley var registry = window.Parsley.\_validatorRegistry = new ValidatorRegistry(window.ParsleyConfig.validators, window.ParsleyConfig.i18n); window.ParsleyValidator = {}; $.each('setLocale addCatalog addMessage addMessages getErrorMessage formatMessage addValidator updateValidator removeValidator hasValidator'.split(' '), function (i, method) { window.Parsley[method] = (...args) => registry[method](...args); window.ParsleyValidator[method] = function () { Utils.warnOnce(`Accessing the method '${method}' through Validator is deprecated. Simply call 'window.Parsley.${method}(...)'`); return window.Parsley[method](...arguments); }; });

### [¶](#17dp8vu) UI Deprecated global object window.Parsley.UI = UI; window.ParsleyUI = { removeError: function (instance, name, doNotUpdateClass) { var updateClass = true !== doNotUpdateClass; Utils.warnOnce(`Accessing UI is deprecated. Call 'removeError' on the instance directly. Please comment in issue 1073 as to your need to call this method.`); return instance.removeError(name, {updateClass}); }, getErrorsMessages: function (instance) { Utils.warnOnce(`Accessing UI is deprecated. Call 'getErrorsMessages' on the instance directly.`); return instance.getErrorsMessages(); } }; $.each('addError updateError'.split(' '), function (i, method) { window.ParsleyUI[method] = function (instance, name, message, assert, doNotUpdateClass) { var updateClass = true !== doNotUpdateClass; Utils.warnOnce(`Accessing UI is deprecated. Call '${method}' on the instance directly. Please comment in issue 1073 as to your need to call this method.`); return instance[method](name, {message, assert, updateClass}); }; });

### [¶](#3rdcrjn) PARSLEY auto-binding Prevent it by setting ParsleyConfig.autoBind to false if (false !== window.ParsleyConfig.autoBind) { $(function () {

* [¶](#26in1rg)  
  Works only on data-parsley-validate.  
   if ($('[data-parsley-validate]').length)  
   $('[data-parsley-validate]').parsley();  
   });  
  }  
    
  export default Parsley;