* 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)

# factory.js

* [¶](#gjdgxs)  
  import $ from 'jquery';  
  import Utils from './utils';  
  import Base from './base';  
  import Form from './form';  
  import Field from './field';  
  import Multiple from './multiple';  
    
  var Factory = function (element, options, parsleyFormInstance) {  
   this.element = element;  
   this.$element = $(element);
* [¶](#30j0zll)  
  If the element has already been bound, returns its saved Parsley instance  
   var savedparsleyFormInstance = this.$element.data('Parsley');  
   if (savedparsleyFormInstance) {
* [¶](#1fob9te)  
  If the saved instance has been bound without a Form parent and there is one given in this call, add it  
   if ('undefined' !== typeof parsleyFormInstance && savedparsleyFormInstance.parent === window.Parsley) {  
   savedparsleyFormInstance.parent = parsleyFormInstance;  
   savedparsleyFormInstance.\_resetOptions(savedparsleyFormInstance.options);  
   }  
    
   if ('object' === typeof options) {  
   Object.assign(savedparsleyFormInstance.options, options);  
   }  
    
   return savedparsleyFormInstance;  
   }
* [¶](#3znysh7)  
  Parsley must be instantiated with a DOM element or jQuery $element  
   if (!this.$element.length)  
   throw new Error('You must bind Parsley on an existing element.');  
    
   if ('undefined' !== typeof parsleyFormInstance && 'Form' !== parsleyFormInstance.\_\_class\_\_)  
   throw new Error('Parent instance must be a Form instance');  
    
   this.parent = parsleyFormInstance || window.Parsley;  
   return this.init(options);  
  };  
    
  Factory.prototype = {  
   init: function (options) {  
   this.\_\_class\_\_ = 'Parsley';  
   this.\_\_version\_\_ = '@@version';  
   this.\_\_id\_\_ = Utils.generateID();
* [¶](#2et92p0)  
  Pre-compute options  
   this.\_resetOptions(options);
* [¶](#tyjcwt)  
  A Form instance is obviously a <form> element but also every node that is not an input and has the data-parsley-validate attribute  
   if (this.element.nodeName === 'FORM' || (Utils.checkAttr(this.element, this.options.namespace, 'validate') && !this.$element.is(this.options.inputs)))  
   return this.bind('parsleyForm');
* [¶](#3dy6vkm)  
  Every other element is bound as a Field or FieldMultiple  
   return this.isMultiple() ? this.handleMultiple() : this.bind('parsleyField');  
   },  
    
   isMultiple: function () {  
   var type = Utils.getType(this.element);  
   return ((type === 'radio' || type === 'checkbox') ||  
   (this.element.nodeName === 'SELECT' && null !== this.element.getAttribute('multiple')));  
   },
* [¶](#1t3h5sf)  
  Multiples fields are a real nightmare :( Maybe some refactoring would be appreciated here…  
   handleMultiple: function () {  
   var name;  
   var multiple;  
   var parsleyMultipleInstance;
* [¶](#4d34og8)  
  Handle multiple name  
   this.options.multiple = this.options.multiple ||  
   (name = this.element.getAttribute('name')) ||  
   this.element.getAttribute('id');
* [¶](#2s8eyo1)  
  Special select multiple input  
   if (this.element.nodeName === 'SELECT' && null !== this.element.getAttribute('multiple')) {  
   this.options.multiple = this.options.multiple || this.\_\_id\_\_;  
   return this.bind('parsleyFieldMultiple');
* [¶](#17dp8vu)  
  Else for radio / checkboxes, we need a name or data-parsley-multiple to properly bind it  
   } else if (!this.options.multiple) {  
   Utils.warn('To be bound by Parsley, a radio, a checkbox and a multiple select input must have either a name or a multiple option.', this.$element);  
   return this;  
   }
* [¶](#3rdcrjn)  
  Remove special chars  
   this.options.multiple = this.options.multiple.replace(/(:|\.|\[|\]|\{|\}|\$)/g, '');
* [¶](#26in1rg)  
  Add proper data-parsley-multiple to siblings if we have a valid multiple name  
   if (name) {  
   $('input[name="' + name + '"]').each((i, input) => {  
   var type = Utils.getType(input);  
   if ((type === 'radio' || type === 'checkbox'))  
   input.setAttribute(this.options.namespace + 'multiple', this.options.multiple);  
   });  
   }
* [¶](#lnxbz9)  
  Check here if we don’t already have a related multiple instance saved  
   var $previouslyRelated = this.\_findRelated();  
   for (var i = 0; i < $previouslyRelated.length; i++) {  
   parsleyMultipleInstance = $($previouslyRelated.get(i)).data('Parsley');  
   if ('undefined' !== typeof parsleyMultipleInstance) {  
    
   if (!this.$element.data('FieldMultiple')) {  
   parsleyMultipleInstance.addElement(this.$element);  
   }  
    
   break;  
   }  
   }
* [¶](#35nkun2)  
  Create a secret Field instance for every multiple field. It will be stored in data('FieldMultiple') And will be useful later to access classic Field stuff while being in a FieldMultiple instance  
   this.bind('parsleyField', true);  
    
   return parsleyMultipleInstance || this.bind('parsleyFieldMultiple');  
   },
* [¶](#1ksv4uv)  
  Return proper Form, Field or FieldMultiple  
   bind: function (type, doNotStore) {  
   var parsleyInstance;  
    
   switch (type) {  
   case 'parsleyForm':  
   parsleyInstance = $.extend(  
   new Form(this.element, this.domOptions, this.options),  
   new Base(),  
   window.ParsleyExtend  
   ).\_bindFields();  
   break;  
   case 'parsleyField':  
   parsleyInstance = $.extend(  
   new Field(this.element, this.domOptions, this.options, this.parent),  
   new Base(),  
   window.ParsleyExtend  
   );  
   break;  
   case 'parsleyFieldMultiple':  
   parsleyInstance = $.extend(  
   new Field(this.element, this.domOptions, this.options, this.parent),  
   new Multiple(),  
   new Base(),  
   window.ParsleyExtend  
   ).\_init();  
   break;  
   default:  
   throw new Error(type + 'is not a supported Parsley type');  
   }  
    
   if (this.options.multiple)  
   Utils.setAttr(this.element, this.options.namespace, 'multiple', this.options.multiple);  
    
   if ('undefined' !== typeof doNotStore) {  
   this.$element.data('FieldMultiple', parsleyInstance);  
    
   return parsleyInstance;  
   }
* [¶](#44sinio)  
  Store the freshly bound instance in a DOM element for later access using jQuery data()  
   this.$element.data('Parsley', parsleyInstance);
* [¶](#2jxsxqh)  
  Tell the world we have a new Form or Field instance!  
   parsleyInstance.\_actualizeTriggers();  
   parsleyInstance.\_trigger('init');  
    
   return parsleyInstance;  
   }  
  };  
    
  export default Factory;