* 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)

# defaults.js

* [¶](#gjdgxs)  
  All these options could be overriden and specified directly in DOM using data-parsley- default DOM-API eg: inputs can be set in DOM using data-parsley-inputs="input, textarea" eg: data-parsley-stop-on-first-failing-constraint="false"  
  var Defaults = {

### [¶](#30j0zll) General

* [¶](#1fob9te)  
  Default data-namespace for DOM API  
   namespace: 'data-parsley-',
* [¶](#3znysh7)  
  Supported inputs by default  
   inputs: 'input, textarea, select',
* [¶](#2et92p0)  
  Excluded inputs by default  
   excluded: 'input[type=button], input[type=submit], input[type=reset], input[type=hidden]',
* [¶](#tyjcwt)  
  Stop validating field on highest priority failing constraint  
   priorityEnabled: true,

### [¶](#3dy6vkm) Field only

* [¶](#1t3h5sf)  
  identifier used to group together inputs (e.g. radio buttons…)  
   multiple: null,
* [¶](#4d34og8)  
  identifier (or array of identifiers) used to validate only a select group of inputs  
   group: null,

### [¶](#2s8eyo1) UI Enable\Disable error messages uiEnabled: true,

* [¶](#17dp8vu)  
  Key events threshold before validation  
   validationThreshold: 3,
* [¶](#3rdcrjn)  
  Focused field on form validation error. ‘first’|’last’|’none’  
   focus: 'first',
* [¶](#26in1rg)  
  event(s) that will trigger validation before first failure. eg: input…  
   trigger: false,
* [¶](#lnxbz9)  
  event(s) that will trigger validation after first failure.  
   triggerAfterFailure: 'input',
* [¶](#35nkun2)  
  Class that would be added on every failing validation Parsley field  
   errorClass: 'parsley-error',
* [¶](#1ksv4uv)  
  Same for success validation  
   successClass: 'parsley-success',
* [¶](#44sinio)  
  Return the $element that will receive these above success or error classes Could also be (and given directly from DOM) a valid selector like '#div'  
   classHandler: function (Field) {},
* [¶](#2jxsxqh)  
  Return the $element where errors will be appended Could also be (and given directly from DOM) a valid selector like '#div'  
   errorsContainer: function (Field) {},
* [¶](#z337ya)  
  ul elem that would receive errors’ list  
   errorsWrapper: '<ul class="parsley-errors-list"></ul>',
* [¶](#3j2qqm3)  
  li elem that would receive error message  
   errorTemplate: '<li></li>'  
  };  
    
  export default Defaults;