Form validation with jQuery

Documentation

Validate forms like you've never validated before!

"But doesn't jQuery make it easy to write your own validation plugin?"
Sure, but there are still a lot of subtleties to take care of: You need a standard library of validation methods (such as emails, URLs, credit card numbers). You need to place error messages in the DOM and show and hide them when appropriate. You want to react to more than just a submit event, like keyup and blur.
You may need different ways to specify validation rules according to the server-side enviroment you are using on different projects. And after all, you don't want to reinvent the wheel, do you?

"But aren't there already a ton of validation plugins out there?"
Right, there are a lot of non-jQuery-based solutions (which you'd avoid since you found jQuery) and some jQuery-based solutions. This particular one is one of the oldest jQuery plugins (started in July 2006) and has proved itself in projects all around the world. There is also an article discussing how this plugin fits the bill of the should-be validation solution.

Not convinced? Have a look at this example:

```
<form class="cmxform" id="commentForm" method="get" action="">
  <fieldset>
    <legend>Please provide your name, email address (won't be published) and a comment</
    <p>
      <label for="cname">Name (required, at least 2 characters)</label>
      <input id="cname" name="name" minlength="2" type="text" required>
    </p>
    <p>
      <label for="cemail">E-Mail (required)</label>
      <input id="cemail" type="email" name="email" required>
    </p>
    <p>
      <label for="curl">URL (optional)</label>
      <input id="curl" type="url" name="url">
    </p>
    <p>
```

```
      <label for="ccomment">Your comment (required)</label>
      <textarea id="ccomment" name="comment" required></textarea>
    </p>
    <p>
      <input class="submit" type="submit" value="Submit">
    </p>
  </fieldset>
</form>
<script>
$("#commentForm").validate();
</script>
```

Isn't that nice and easy?

A single line of jQuery to select the form and apply the validation plugin, plus a few annotations on each element to specify the validation rules.

Of course that isn't the only way to specify rules. You also don't have to rely on those default messages, but they come in handy when starting to setup validation for a form.

A few things to look out for when playing around with the demo

- After trying to submit an invalid form, the first invalid element is focused, allowing the user to correct the field. If another invalid field – that wasn't the first one – was focused before submit, that field is focused instead, allowing the user to start at the bottom if he or she prefers.
- Before a field is marked as invalid, the validation is lazy: Before submitting the form for the first time, the user can tab through fields without getting annoying messages – they won't get bugged before having the chance to actually enter a correct value
- Once a field is marked invalid, it is eagerly validated: As soon as the user has entered the necessary value, the error message is removed
- If the user enters something in a non-marked field, and tabs/clicks away from it (blur the field), it is validated – obviously the user had the intention to enter something, but failed to enter the correct value

That behaviour can be irritating when clicking through demos of the validation plugin – it is designed for an unobtrusive user experience, annoying the user as little as possible with unnecessary error messages. So when you try out other demos, try to react like one of your users would, and see if the behaviour is better then. If not, please let me know about any ideas you may have for improvements!

API Documentation

You're probably looking for

Options for the validate() method

If not, read on.

Throughout the documentation, two terms are used very often, so it's important that you know their meaning in the context of the validation plugin:

- **method**: A validation method implements the logic to validate an element, like an email method that checks for the right format of a text input's value. A set of standard methods is available, and it is easy to write your own.
- **rule**: A validation rule associates an element with a validation method, like "validate input with name "primary-mail" with methods "required" and "email".

Plugin methods

This library adds three jQuery plugin methods, the main entry point being the `validate` method:

- `validate()` – Validates the selected form.
- `valid()` – Checks whether the selected form or selected elements are valid.
- `rules()` – Read, add and remove rules for an element.

Custom selectors

This library also extends jQuery with three custom selectors:

- `:blank` – Selects all elements with a blank value.
- `:filled` – Selects all elements with a filled value.
- `:unchecked` – Selects all elements that are unchecked.

Validator

The validate method returns a Validator object that has a few public methods that you can use to trigger validation programmatically or change the contents of the form. The validator object has more methods, but only those documented here are intended for usage.

- `Validator.form()` – Validates the form.
- `Validator.element()` – Validates a single element.
- `Validator.resetForm()` – Resets the controlled form.
- `Validator.showErrors()` – Show the specified messages.
- `Validator.numberOfInvalids()` – Returns the number of invalid fields.
- `Validator.destroy()` – Destroys this instance of validator.

There are a few static methods on the validator object:

- `jQuery.validator.addMethod()` – Add a custom validation method.
- `jQuery.validator.format()` – Replaces {n} placeholders with arguments.
- `jQuery.validator.setDefaults()` – Modify default settings for validation.
- `jQuery.validator.addClassRules()` – Add a compound class method.

List of built-in Validation methods

A set of standard validation methods is provided:

- `required` – Makes the element required.
- `remote` – Requests a resource to check the element for validity.
- `minlength` – Makes the element require a given minimum length.
- `maxlength` – Makes the element require a given maximum length.
- `rangelength` – Makes the element require a given value range.
- `min` – Makes the element require a given minimum.
- `max` – Makes the element require a given maximum.
- `range` – Makes the element require a given value range.
- `step` – Makes the element require a given step.
- `email` – Makes the element require a valid email
- `url` – Makes the element require a valid url
- `date` – Makes the element require a date.
- `dateISO` – Makes the element require an ISO date.
- `number` – Makes the element require a decimal number.
- `digits` – Makes the element require digits only.
- `equalTo` – Requires the element to be the same as another one

Some more methods are provided as add-ons, and are currently included in additional-methods.js in the download package. Not all of them are documented here:

- `accept` – Makes a file upload accept only specified mime-types.
- `creditcard` – Makes the element require a credit card number.
- `extension` – Makes the element require a certain file extension.
- `phoneUS` – Validate for valid US phone number.
- `require_from_group` – Ensures a given number of fields in a group are complete.

You can find the source code for all additional methods in the GitHub repository.

It is possible to re-define the implementation of the built-in rules using the `$.validator.methods` property

General Guidelines

The General Guidelines section provides detailed discussion of the design and ideas behind the plugin, explaining why certain things are as they are. It covers the features in more detail than the API documentation, which just briefly explains the various methods and options available.

If you've decided to use the validation plugin in your application and want to get to know it better, it is recommended that you read the guidelines.

Fields with complex names (brackets, dots)

When you have a name attribute like user[name], make sure to put the name in quotes. More details in the General Guidelines.

Too much recursion

Another common problem occurs with this code:

```
$("#myform").validate({
 submitHandler: function(form) {
   // some other code
   // maybe disabling submit button
   // then:
   $(form).submit();
 }
});
```

This results in a too-much-recursion error: `$(form).submit()` triggers another round of validation, resulting in another call to submitHandler, and voila, recursion. Replace that with form.submit(), which triggers the native submit event instead and not the validation.

So the correct code looks slightly different:

```
$("#myform").validate({
 submitHandler: function(form) {
   form.submit();
 }
});
```

Demos

The Marketo sign-up form

The Marketo sign-up form, step 2

Based on an old version of the marketo.com sign-up form. The custom validation was once replaced with this plugin. Thanks to Glen Lipka for contributing it!

Notable features of the demo:

- Customized message display: No messages displayed for the required method, only for typing-errors (like wrong email format); A summary is displayed at the top ("You missed 12 fields. They have been highlighted below.")
- Remote validation of email field. Try to enter eg. glen@marketo.com
- Integration with masked-input plugin, see Zip and Phone fields and Credit Card Number on step 2
- A custom method for making the billing address on step 2 optional when "Same as Company Address" is checked
- A custom method for checking the password: Checks that the password contains at least one number and one character and that it is at least 6 characters long. If the user blurs the field with an invalid value, the input is emptied and gets focus again.

The Remember The Milk sign-up form

The sign-up form from rememberthemilk.com (based on an older version). The custom validation was replaced using this plugin. Thanks to RTM for contributing!

Notable features of the demo:

- Custom message display, based on the original table layout, using success option to display a checkmark for valid fields
- Remote validation of username, to check if it is already taken (try "Peter", "asdf" or "George")

A multipart "buy&sell a house" form

Contributed by Michael Evangelista, showing a multipart form for buying and selling houses.

Notable features of the demo:

- Multipart, implemented using the jQuery UI accordion and a custom method to check if an element is on the current page when validated
- Integration with masked-input plugin, see Phone and Zip fields

Using remote validation to help with captchas

Features remote validation for helping the user to fill out captchas.

Notable features of the demo:

- Remote validation to check if the user entered the correct captcha, without forcing him to submit the form first

---