jQuery

Mask Plugin

A jQuery Plugin to make masks on form fields and html elements.

Documentation

Basic Usage Examples

```
$(document).ready(function(){
 $('.date').mask('00/00/0000');
 $('.time').mask('00:00:00');
 $('.date_time').mask('00/00/0000 00:00:00');
 $('.cep').mask('00000-000');
 $('.phone').mask('0000-0000');
 $('.phone_with_ddd').mask('(00) 0000-0000');
 $('.phone_us').mask('(000) 000-0000');
 $('.mixed').mask('AAA 000-S0S');
 $('.cpf').mask('000.000.000-00', {reverse: true});
 $('.money').mask('000.000.000.000.000,00', {reverse: true});
 $('.money2').mask("#.##0,00", {reverse: true});
 $('.ip_address').mask('0ZZ.0ZZ.0ZZ.0ZZ', {
  translation: {
   'Z': {
    pattern: /[0-9]/, optional: true
   }
  }
 });
 $('.ip_address').mask('099.099.099.099');
 $('.percent').mask('##0,00%', {reverse: true});
 $('.clear-if-not-match').mask("00/00/0000", {clearIfNotMatch: true});
 $('.placeholder').mask("00/00/0000", {placeholder: "__/__/____"});
 $('.fallback').mask("00r00r0000", {
   translation: {
    'r': {
     pattern: /[\/]/,
     fallback: '/'
    },
    placeholder: "__/__/____"
   }
  });
 $('.selectonfocus').mask("00/00/0000", {selectOnFocus: true});
);
```

Callback Examples

```
var options = {
 onComplete: function(cep) {
  alert('CEP Completed!:' + cep);
 },
 onKeyPress: function(cep, event, currentField, options){
  console.log('An key was pressed!:', cep, ' event: ', event,
           'currentField: ', currentField, ' options: ', options);
 },
 onChange: function(cep){
  console.log('cep changed! ', cep);
 },
 onInvalid: function(val, e, f, invalid, options){
  var error = invalid[0];
  console.log ("Digit: ", error.v, " is invalid for the position: ", error.p, ". We expect something like: ", error.e);
 }
};

$('.cep_with_callback').mask('00000-000', options);
```

On-the-fly mask change

```
var options =  {onKeyPress: function(cep, e, field, options){
  var masks = ['00000-000', '0-00-00-00'];
    mask = (cep.length>7) ? masks[1] : masks[0];
  $('.crazy_cep').mask(mask, options);
}};

$('.crazy_cep').mask('00000-000', options);
```

Mask as a function

```
var SPMaskBehavior = function (val) {
  return val.replace(/\D/g, '').length === 11 ? '(00) 00000-0000' : '(00) 0000-00009';
},
spOptions = {
  onKeyPress: function(val, e, field, options) {
      field.mask(SPMaskBehavior.apply({}, arguments), options);
    }
};

$('.sp_celphones').mask(SPMaskBehavior, spOptions);
```

Using HTML Notation Exemples

To get your mask applied with the data-mask attribute just use it as the same way you use with the $.mask function.

```
<input type="text" name="field-name" data-mask="00/00/0000" />
```

Activating a reversible mask

```
<input type="text" name="field-name" data-mask="00/00/0000" data-mask-reverse="true" />
```

Using clearIfNotMatch option

```
<input type="text" name="field-name" data-mask="00/00/0000" data-mask-clearifnotmatch="true" />
```

Using selectOnFocus option

```
<input type="text" name="field-name" data-mask="00/00/0000" data-mask-selectonfocus="true" />
```

Translation

Teach to jQuery Mask Plugin how to apply your mask:

```
// now the digit 0 on your mask pattern will be interpreted
// as valid characters like 0,1,2,3,4,5,6,7,8,9 and *
```

```
$('.your-field').mask('00/00/0000', {'translation': {0: {pattern: /[0-9*]/}}});
```

By default, jQuery Mask Plugin only reconizes the logical digit A (Numbers and Letters) and S (A-Za-z) but you can extend or modify this behaviour by telling to jQuery Mask Plugin how to interpret those logical digits.

```
$('.your-field').mask('AA/SS/YYYY', {'translation': {
                        A: {pattern: /[A-Za-z0-9]/},
                        S: {pattern: /[A-Za-z]/},
                        Y: {pattern: /[0-9]/}
                      }
                });
```

Now jQuery Mask Plugin knows the logic digit Y and you can create your own pattern.

Optional digits

You can also tell to jQuery Mask which digit is optional, to create a IP mask for example:

```
// way 1
$('.ip_address').mask('099.099.099.099');
// way 2
$('.ip_address').mask('0ZZ.0ZZ.0ZZ.0ZZ', {translation:  {'Z': {pattern: /[0-9]/, optional: true}}});
```

Now, all **Z** digits in your masks is optional.

Recursive digits

With jQuery Mask Plugin you can also define recursive patterns inside your mask:

```
$('.money_example').mask('#.##0,00', {reverse: true});
```

With example above the mask will be placed from the right to the left (that's why reverse:true is defined). As soon as you start typing, a "0,00" will be applied followed by repeating recursively the following pattern "#.##". The result could be something like: 1.234.567,890.

You can also use that kind of feature to define what kind of data could be typed inside of a field:

```
$('.example').mask('0#');
```

Now only numbers will be allowed inside your form field.

Fallback digits

When a user types a invalid char for the current position the plugin will replace it by its fallback instead of erasing them.

```
$('.field_with_fallback').mask("00r00r0000", {
  translation: {
    'r': {
      pattern: /[\/]/,
      fallback: '/'
    },
    placeholder: "__/__/____"
  }
});
```

Removing the mask

```
$('.date').unmask();
```

Getting the unmasked typed value

```
$('.date').cleanVal();
```

Customization

jQuery Mask Plugin has a few default options that you can overwrite as you like:

Field Options

```
var custom_options = {
  byPassKeys: [8, 9, 37, 38, 39, 40],
  translation: {
            '0': {pattern: /\d/},
            '9': {pattern: /\d/, optional: true},
            '#': {pattern: /\d/, recursive: true},
            'A': {pattern: /[a-zA-Z0-9]/},
            'S': {pattern: /[a-zA-Z]/}
        };
};
```

byPassKeys list of keyboard's keyCode that you want to be ignored when it was pressed.
translation object with all digits that should be interpreted as a special chars and its regex representation.

Public Methods

```
/**
 * Applies the mask to the matching selector elements.
 *
 * @selector elements to be masked.
 * @mask should be a string or a function.
 * @options should be an object.
 **/
$(selector).mask(mask [, options]);

/**
 * Seek and destroy.
 *
 * @selector elements to be masked.
 **/
$(selector).unmask();

/**
 * Gets the value of the field without the mask.
 *
 * @selector elements to be masked.
 **/
$(selector).cleanVal();

/**
 * Applies the mask to the matching selector elements.
 *
 * @selector optional: elements to be masked. The default behaviour it's to lookup for all elements with data-mask attribute.
 **/
$.applyDataMask([selector]);
```

Global Options

```
// nonInput: elements we consider nonInput
// dataMask: we mask data-mask elements by default
// watchInputs: watch for dynamically added inputs by default
// watchDataMask: by default we disabled the watcher for dynamically added data-mask elements by default (performance reasons)
```

```
$.jMaskGlobals = {
  maskElements: 'input,td,span,div',
  dataMaskAttr: '*[data-mask]',
  dataMask: true,
  watchInterval: 300,
  watchInputs: true,
  watchDataMask: false,
  byPassKeys: [9, 16, 17, 18, 36, 37, 38, 39, 40, 91],
  translation: {
    '0': {pattern: /\d/},
    '9': {pattern: /\d/, optional: true},
    '#': {pattern: /\d/, recursive: true},
    'A': {pattern: /[a-zA-Z0-9]/},
    'S': {pattern: /[a-zA-Z]/}
  }
};
```