# data-driven-components Documentation

***Release master***

**Feb 17, 2018**

# Contents

CHAPTER 1

# Introduction

Data Driven Components (briefly DDC) is a javascript jquery plugin that simplify data handling component building through easy and flexible options definitions. The goal with this approach would be give a way to build Single-Page Application in a while provided that the database has already been designed and built.

CHAPTER 2

# Requirements

The following libraries are required for DDC to function properly: - jQuery - Bootstrap 3 - Font Awesome - Datatables - Bootstrap-combobox - Bootstrap-datepicker

CHAPTER 3

# Installation

Just put the library with his dependancies in your index.html page as described in the docs folder.

```
<script src="bootstrap.data-driven-components.js"></script>
```

CHAPTER 4

# Demo

https://codicepulito.github.io/data-driven-components

CHAPTER 5

# Documentation

CHAPTER 6

# Global

## 6.1 ddcVersion()

Return [semver][http://semver.org/{]}(http://semver.org/) compatible version number

**Returns**: `String`, Actual version

## 6.2 ddcClearAll(except)

Empty all root nodes except those passed in parameter arrays

## 6.3 Example

```
$('#root').ddcClearAll(['navbar1'])
```

**Parameters**

**except**: `Array`, Array of elements to not empty

**Returns**: `void`

### 6.3.1 ddcDatatable(parameters)

## 6.4 Append a datatable - http://www.datatables.net/

## 6.5 Copyright (c) 2008-2015 SpryMedia Limited

**Parameters**

**parameters**: `object`, Object with elements required to generate the html snippet: - datatableId: valid html5 id attribute; see https://www.w3.org/TR/html5/dom.html#the-id-attribute - ajax: asyncronous function call options - ajax.jsend: set the [jsend][https://labs.omniti.com/labs/jsend{]}(https://labs.omniti.com/labs/jsend) compatibility - ajax.responseDataKey: if ajax.jsend is false, set the object key contains data - ajax.url: a valid url address - buttons: array that defines the buttons that will appear in the document to the end user as documented at https://datatables.net/reference/option/buttons.buttons - dom: String that define the table control elements to appear on the page and in what order as documented at https://datatables.net/reference/option/dom - onClick: function callback called on row's item clicked - panel: string that define the title of a bootstrap panel to wrap into - pageLength: integer Number of rows to display on a single page when using pagination as documented at https://datatables.net/reference/option/pageLength - priorityColumns: array of elements to set visibility priority to the columns, telling Responsive which columns it should remove before others; see https://datatables.net/extensions/responsive/priority - response: dataset response object in [jsend][https://labs.omniti.com/labs/jsend{]}(https://labs.omniti.com/labs/jsend) format with optional schema (columns info)

**Returns**: `void`,

## 6.6 Example 1: Datatable with manual data

```
$('#root').ddcDatatable({
   datatableId: 'datatable1',
   response: {
     data: [
       {
           "id": 1,
           "name": "Leanne Graham",
           "username": "Bret",
           "email": "Sincere@april.biz",
           "phone": "1-770-736-8031 x56442",
           "website": "hildegard.org",
           "edit": "<center><button id=\"1\"></button></center>"
       },
       {
           "id": 2,
           "name": "Ervin Howell",
           "username": "Antonette",
           "email": "Shanna@melissa.tv",
           "phone": "010-692-6593 x09125",
           "website": "anastasia.net",
           "edit": "<center><button id=\"2\"></button></center>"
       }
     ]
   },
   buttons: [],
   priorityColumns: {name: 1, username: 2, email: 3},
   onClick: datatable1Click
```

(continues on next page)

(continued from previous page)

```
})

// callback function
function datatable1Click(this) {
 var id = $(this).attr('id')
}
```

## 6.7 Example 2: Datatable with ajax remote data

```
$('#root').ddcDatatable({
   datatableId: 'datatable1',
   ajax: {
     url: 'https://randomuser.me/api/?results=20',
     responseDataKey: 'results',
     jsend: false
   },
   response: null,
   buttons: [],
   priorityColumns: {name: 1, username: 2, email: 3},
   onClick: datatable1Click
})

// callback function
function datatable1Click(this) {
 var id = $(this).attr('id')
}
```

### 6.7.1 ddcForm(parameters)

Append a bootstrap form with inputs and input-group-addon

**Parameters**

**parameters**: `object`, Object with elements required to generate the html snippet: - formId: valid html5 id attribute; see https://www.w3.org/TR/html5/dom.html#the-id-attribute - ajax: asyncronous function call options - ajax.jsend: set the [jsend][https://labs.omniti.com/labs/jsend{}](https://labs.omniti.com/labs/jsend) compatibility - ajax.responseDataKey: if ajax.jsend is false, set the object key contains data - ajax.url: a valid url address - buttons: array of objects [button0, button1, . . . , buttonN] - button0.name: string representing the html button label - button0.class: valid html class attribute; see https://www.w3.org/TR/html5/dom.html#classes - button0.id: valid html5 id attribute; see https://www.w3.org/TR/html5/dom.html#the-id-attribute - button0.onClick: function callback called on button clicked - datepicker: Datepicker options; see https://bootstrap-datepicker.readthedocs.io/en/stable/options.html) - fields: array of objects [field0, field1, . . . , fieldN] - field0.addon: optional array of elements - field0.addon.icon: string without "fa" representing the span class (require [Font Awesome][http://fontawesome.io/{}](http://fontawesome.io/)) - field0.addon.onClick: function callback called on addon span clicked - field0.class: optional string representing one or more html class attribute see https://www.w3.org/TR/html5/dom.html#classes - field0.name: string representing the html input label also used as id after removing the spaces and concatenated with formId [formId-field0.name] - field0.readonly: boolean - if true make field readonly - field0.type: data type [string|bool|lookup|datepicker] - override schema.fields.native_type (lookup require [bootstrap-combobox][https://github.com/danielfarrell/bootstrap-combobox{}](https://github.com/danielfarrell/bootstrap-combobox)) (datepicker require [bootstrap-datepicker][https://github.com/uxsolutions/bootstrap-datepicker{}](https://github.com/uxsolutions/bootstrap-datepicker)) - modal: optional string render the form in modal with the specified title -

panel: string that define the title of a bootstrap panel to wrap into - response: dataset response object in jsend format with optional schema (ex. PHP PDO getColumnMeta)

**Returns**: `void`,

## 6.8 Example 1: Form with manual data

```
$('#root').ddcForm({
  formId: 'form2',
  title: 'Form',
  panel: 'Form with manual data',
  datepicker: {
    autoclose: 'true',
    language: 'it',
    format: 'yyyy-mm-dd'
  },
  response: {
      data: [
          {
            field1: 'value1',
            field2: 'value2',
            field3: true,
            field4: '2017-01-01'
          }
      ],
      schema: {
          fields: [
            {name: "field1", native_type: "varchar"},
            {name: "field2", native_type: "varchar"},
            {name: "field3", native_type: "bool"},
            {name: "field4", native_type: "date"}
          ]
      }
  },
  fields: [
      {
        name: "field1",
        class: 'col-4',
        type: "lookup",
        data: [
          { value: '001', text: 'lookupform1' },
          { value: '002', text: 'lookupform2' }
        ]
      },
      {name: "field2", class: 'col-4', addon: { icon: 'reply', onClick: form1Click }},
      {name: "field3", class: 'col-4'},
      {name: "field4", class: 'col-4', type: 'datepicker'}
  ],
  buttons: [
      { name: "Cancel", class: "btn btn-default" },
      { name: "Add", class: "btn btn-primary", id: 'addForm2Send', onClick:␣
↪addFormSend }
  ]
})

// callback function for button
```

(continues on next page)

(continued from previous page)

```
function addFormSend(parameters) {
    console.log(parameters)
}

// callback function for addon
function form1Click(this) {
    var id = $(this).attr('id')
    console.log(id)
}
```

## 6.9 Example 2: Form with lookup ajax remote data

```
$('#root').ddcForm({
  formId: 'form1',
  panel: 'Form with ajax remote data',
  response: null,
  fields: [
    {
      name: "field1",
      type: "lookup",
      url: 'https://raw.githubusercontent.com/codicepulito/data-driven-components/
↪master/test/json/jsendLookup.json'
    },
    {name: "field2", type: "string"},
    {name: "field3", type: "bool"}
  ],
  buttons: [
    { name: "Cancel", class: "btn btn-default" },
    { name: "Add", class: "btn btn-primary", id: 'addForm1Send', onClick: addFormSend␣
↪}
  ]
})

// callback function for button
function addFormSend(parameters) {
    console.log(parameters)
}
```

### 6.9.1 ddcLocale(locale)

Get or set a language locale

**Parameters**

**locale**: `string`, Optional language locale setter

**Returns**: `Array`, Actual country code and language locale

## 6.10 Example

```
$('#root').ddcLocale('it')
```

### 6.10.1 ddcModal(modalId, title, message, buttons)

Append a bootstrap modal with title and message

**Parameters**

**modalId**: `string`, A valid html5 id attribute; see https://www.w3.org/TR/html5/dom.html#the-id-attribute

**title**: `string`, The modal title

**message**: `string`, The modal body contains the message

**buttons**: `Array`, array of objects [button0, button1, ..., buttonN] - button0.class: valid html class attribute; see https://www.w3.org/TR/html5/dom.html#classes - button0.data: string value usable in callback - button0.id: valid html5 id attribute; see https://www.w3.org/TR/html5/dom.html#the-id-attribute - button0.name: string representing the html button label - button0.onClick: function callback called on button clicked

**Returns**: `void`,

## 6.11 Example

```
$('#root').ddcModal('modal1', 'Modal Title', 'This is a message.');
$('#modal1').modal('show');
```

## 6.12 Example with buttons

```
// callback functions
function addModalSend(value) {
  console.log(value)
}

$('#root').ddcModal('modal1', 'Modal Title', 'This is a message.', [
 { name: "Cancel", class: "btn btn-default" },
 { name: "Add", class: "btn btn-primary", data: 'myValue', id: 'addModalSend',␣
↪onClick: addModalSend }
]);
$('#modal1').modal('show');
```

### 6.12.1 ddcNavbar(parameters)

Append a bootstrap navbar menu with items and dropdown sub-items

**Parameters**

**parameters**: `object`, Object with elements required to generate the html snippet: - navbarId: valid html5 id attribute; see https://www.w3.org/TR/html5/dom.html#the-id-attribute - items: array of objects [item0, item1, ..., itemN] - item0.id: null if it has submenu or valid html5 id attribute - item0.name: null as separator or string representing

the html value of item visible to the user - item0.submenu: optional array of items object [subitem0, subitem1, . . . , subitemN] - item0.onClick: function callback called on item/subitem click

**Returns**: `void`,

## 6.13 Example

```
// callback functions
function navbar1Click(id) {
  $('#root').ddcModal('modal1', 'Navbar Click', 'Navbar subitem 1 clicked.');
  $('#modal1').modal('show');
}

function navbar2Click(id) {
  $('#root').ddcModal('modal1', 'Navbar Click', 'Navbar subitem 2 clicked.');
  $('#modal1').modal('show');
}

function navbar3Click(id) {
  $('#root').ddcModal('modal1', 'Navbar Click', 'Navbar item 3 clicked.');
  $('#modal1').modal('show');
}

$(document).ready(function() {
  $('#root').ddcNavbar({
    navbarId: 'navbar1',                // id attribute
    items: [
      {
        id: null,                       // id attribute
        name: "Item 1",                 // html value visible to the user
        submenu: [
          { id: 1, name: "Subitem 1", onClick: navbar1Click},
          { id: null, name: null },     // separator
          { id: 2, name: "Subitem 2", onClick: navbar2Click}
        ]
      },
      { id: 3, name: "Item 3", onClick: navbar3Click},
    ]
  })
})
```