

What is GironaForms?

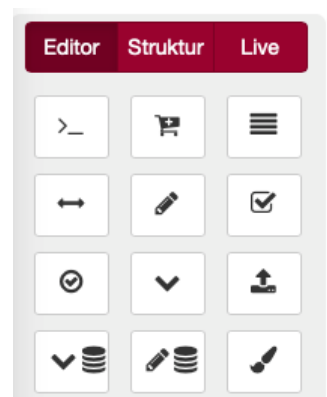
It's a very simple form editor, designed for creating forms fast and simple and always responsive. To add an element just drag one of the symbol and drop it into the place you want to insert it.













The document is structured in rows and columns.
There can up to be 12 columns in one row.
The number of rows is not limited.

For this we are using Bootstrap.
Actually we simply have 1 JS-Class which is for editing and showing the form.

We're offer the following elements

```
C_Option.prototype.TYPE_TEXT = "text";
C_Option.prototype.TYPE_INT = "int";
C_Option.prototype.TYPE_BIGTEXT = "bigtext";
C_Option.prototype.TYPE_BIGPLAINTEXT = „bigplaintext";
C_Option.prototype.TYPE_INPUTTYPE = "inputtype";
C_Option.prototype.TYPE_INPUTBOOL = "inputbool";
C_Option.prototype.TYPE_MULTIOPTION = "radiooption";
C_Option.prototype.TYPE_SELECTOPTION = "selectoption";
C_Option.prototype.TYPE_ADDRESSTYPESELECT = "adresstypeselect";
```



	Row		Textarea (WYSIWYG)
	Column		Input-Field (Text, email),
	Line (<hr/>)		Checkbox
	Radio-Button		Dropdown
	Dropdown JSON		File-Upload
	Input JSON Suggest		

Editor:

The screenshot shows the Girona Editor interface. At the top, there is a header bar with the 'girona Editor' logo and the title '_Grabmalantrag_Ibbenbueren'. Below the header is a menu bar with 'Datei', 'Navigation', and 'Hilfe'. A toolbar with various icons is located below the menu bar. A blue box labeled '1' highlights the main form area. A blue arrow labeled '3.1 Drag' points to the toolbar. A blue arrow labeled '3.2 Drop' points to the form area. A blue circle labeled '2' is in the top right corner. The form itself is titled 'Musterformular' and contains several input fields for user information, including 'Name der/des Inhaberin/Inhabers', 'Beauftragter Gewerbebetrieb', 'Straße/Haus-Nr.', 'Fachfirma Name', 'Fachfirma Vorname', 'Postleitzahl/Wohnort', 'user.plz', and 'user.ort'. At the bottom, there is a section for 'Antrag auf Errichtung/Veränderung eines Grabmal/einer Grabeinfassung' with a dropdown for 'Grabmal - GM1' and a section for 'Friedhof - FNR' with a dropdown for 'Abteilung' and a section for 'Grab-Nr.' with a dropdown for 'Grabart - GRABART'.

This is the editor in edit mode!

- (1) shows the formpagewrapper. Within this `<div>` everything is generated by the main.js
- (2) Are the available modes: „Edit“, „Structure“ and „Preview“. The mode „Live“ is for viewing and filling the form. The button „Live“ is not available here“
- (3) To add elements just drag an icon from the bar and drop it within the formpage-wrapper.

Options for input

If you click on an edit field in edit mode the Setting or Options for that input will open.
Here are more Parameters like label, placeholder, dataname etc.

The screenshot shows a web form titled "Musterformular" from "ibb Ibbenbüren Das Hoch im Münsterland". A modal window titled "Einstellungen für input" is open, displaying configuration options for an input field. The modal includes the following settings:

- Titel:** Name der/des Inhaberin/Inhabers
- Platzhalter:** Name der/des Inhaberin/Inhabers
- Datenname:** Name der/des Inhaberin/Inhabers
- Titel Anzeigen:** Anzeigen
- Inputfeld verstecken:** Nein
- Eingabetyp:** Text

The dataname is very important. This defines the key which is used later to read the data from this field.

main.js

The main.js contains the logic for building, viewing and editing GironaForms.

Usage:

At first we need to initialize the formpage.

```
let myForm = new FormPage("", $("#formpage"));  
myForm.setMode(FormPage.prototype.MODE_LIVE);
```

With *setMode* you can change the mode to:

MODE_EDIT
MODE_PREVIEW
MODE_LIVE
MODE_STRUCTURE

Load existing Forms:

Next we initialize the formloader, which we need to load the form from the JSON file.

```
var fl = new FormLoader();  
let myForm = fl.loadVersion(JSON.parse(FORMFILE), $("#formpage"));
```

Load data to form

If you want to fill the form with existing data you can use this:

```
myForm.setFormData(newData);  
myForm.update();
```

Helper stuff

To initialize date picker etc use this:

```
$('body').updatePolyfill();  
  
$(".datepicker").datepicker({  
  language: "de",  
  format: 'dd.mm.yyyy'  
});
```

Structure of main.js

```
function FormEvents(formPage)
function IdGenerator()
function BasicElement(parent, $ele)
function FormLoader()
function FormPage(name, $id_wrapper)
function Row(formpage, $ele)
function Column(row, $ele)
function Component
```

Simple structure:

Row contains **Columns**
Columns contains **Component**

A component can be:

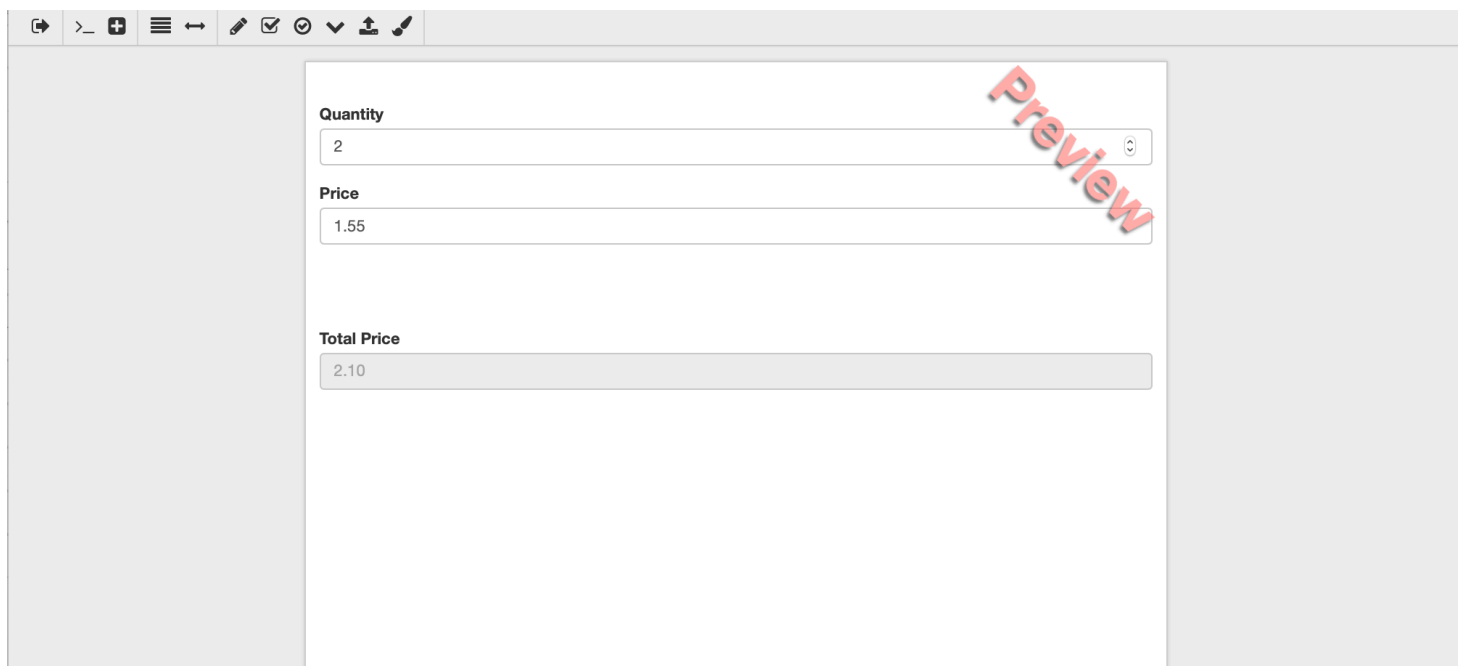
```
function C_Input(column, $ele)
function C_SuggestInput(column, $ele)
function C_AddressInput(column, $ele)
function C_Dropzone(column, $ele)
function C_Checkbox(column, $ele)
function C_Radio(column, $ele)
function C_Select(column, $ele)
function C_Signature(column, $ele)
function C_URLSelect(column, $ele)
```

Your task

We want to add a new component named „Formular“.
This should be a disabled textfield which can calculate with other data in the form.

You have to add **C_FormularInput**

Example shown below!



The screenshot shows a software development interface with a toolbar at the top. The main area displays a form preview with a large red "Preview" watermark. The form contains three input fields: "Quantity" with the value "2", "Price" with the value "1.55", and "Total Price" with the value "2.10". The "Total Price" field is highlighted with a grey background, indicating it is a calculated field.

Field	Value
Quantity	2
Price	1.55
Total Price	2.10