ACTION FORM

User Manual

Version 3.5.0, Released Jul 18, 2014

www.dnnsharp.com

Contents

What is Action Form?

Action Form is the most powerful, flexible, secure, fast and easy-to-use tool that enables the creation of amazing web forms. The web is all about interaction, and web forms are an essential part of that. Forms created by Action Form are completely customizable, giving you the power to make any type of form you can imagine, e.g. surveys, contact forms, widgets and lead generation forms etc. Moreover, it takes just minutes to set up even the most complex forms for your website, with no programming and technical experience required.

Besides, you will benefit from our excellent support and quick responses for implementing your feature requests.

Why Action Form?

Action Form is the fastest DNN Forms solution for controlling access to portal resources, for collecting data from users before granting them access, for updating profiles, building registration and log in forms, for controlling exactly how the data is pushed into the database using SQL Queries and for much more.

Once you begin collecting results, it provides functionality of powerful functions covering workflows, marketing, sharing, downloading security and tracking. Using Action Form makes possible to carry out a real-time interaction through a variety of fields.

With regular updates, enjoy even faster and more lightweight builder, providing you all the modern features.

Key Features

- Easy build complex forms
- Fast and powerful functions
- Fully customized yet Simple layout and design
- Secure log in and registration
- Easy to administrate and moderate
- 1 Year of Free Updates
- Efficient database access management
- And much more

Support

Our support staff is friendly and always available to help you. Engage us on our community platform.

You can download latest version of Action Form from http://www.dnnsharp.com/dnn/modules/action-form-builder. While on that page, you can also check the changelog to see what's new.

Getting Started

How to Use Action From

Basically, you can have an Action Form module fully functional simply by going to the Manage Form main page, chose a predefined template form (Contact, Log in, Registration, Subscribe to MailChimp) and click the Back button. This add on the page a form with default settings, or you can start with a new form from scratch by selecting the Blank form template, so that you can configure it according to your needs.

With each upgrade we include new cool features. We also have complied a list with tutorials in order to help you better manage the module, click here to see them.

How to Install Action Form

You can download the latest version from Action Form Download Page. While you are on that page, you can also check the changelog to see what's new.

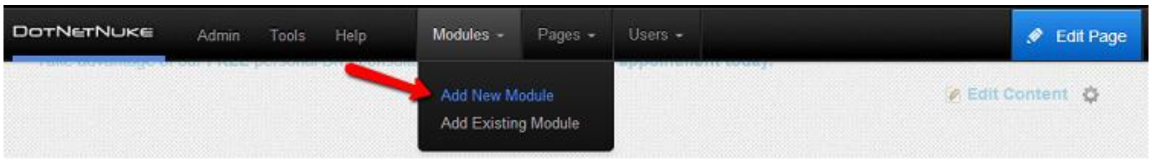
In order to install Action Form module on your DNN website, you can follow the steps below:

1. Begin by Logging into your DNN site with host account;
2. From the Host Menu option select Extensions;
3. On the Extensions module menu, click on Install Extension Wizard button;
4. On the Upload New Extension Package page, click on Choose File button;
5. Select the Action Form package you previously downloaded on your hard drive, then click on Open;
6. Click on Next, accept the license, continue through the install. If all goes well, you will get a list of messages with "EndJob Installation Successful" at the bottom. When all is finished, click on Return button.

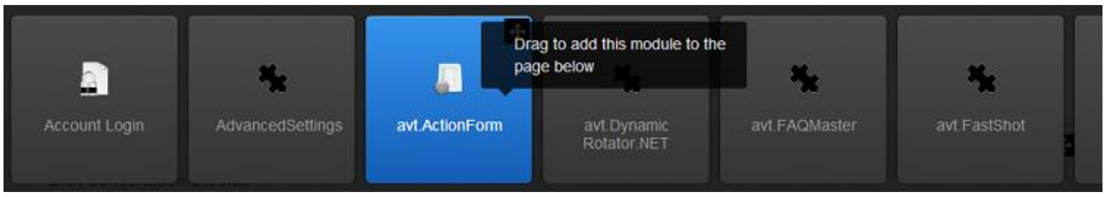
And that's pretty much all about the installation, the next step is to activate the module from production or if you try a demo, you have to unlock the trial for 30 days - these two options are displayed as a warning message at the bottom of the module after it was added on a page.
Add Action Form Module to your page

In the steps below, we are going to show you how to add a predefined contact form, don't forget that Action Form is configurable, so you have the ability to create many other more complex forms.

1. When the page you work on is opened, select the Modules option from the top menu, click on the Add New Module option.

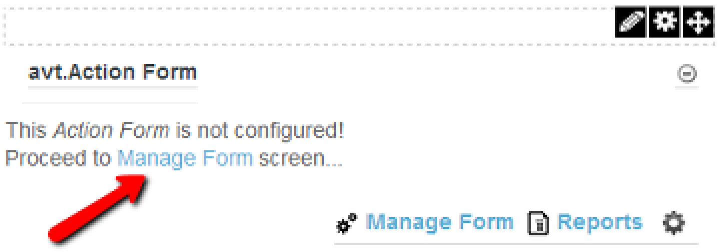


2. Find avt.ActionForm and with Drag&Drop option add the module to your page.



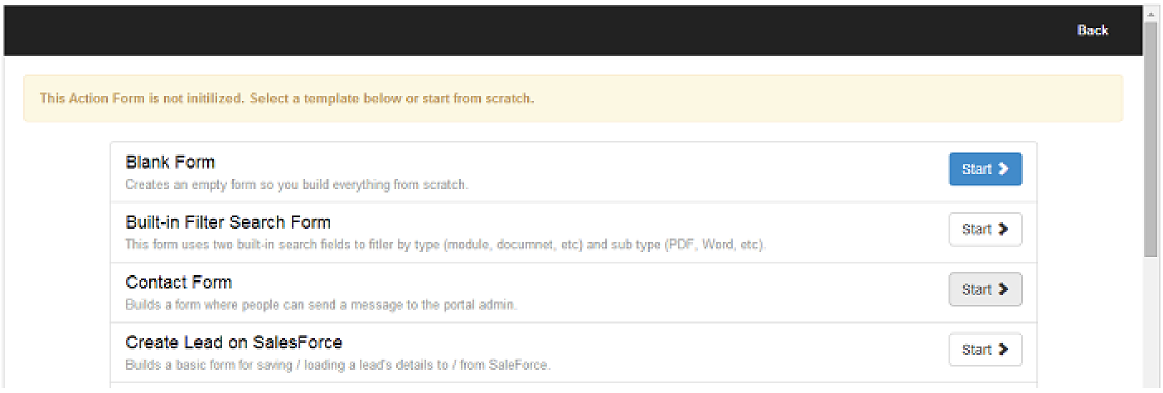
3. Now that you have installed Action Form Module on you page, you will need to configure it by clicking on Manage Form option.

Important: Note that you will need to unlock the 30 Day trial or Activate for Production before you are able to see this screen.


avt.Action Form

This *Action Form* is not configured!
Proceed to Manage Form screen...

4. In this example we are going to create a standard Contact Form - when you are on the Manage Form page, click on Start button displayed on the Contact Form template. You will be redirected to the form settings screen.


Back

This Action Form is not initilized. Select a template below or start from scratch.

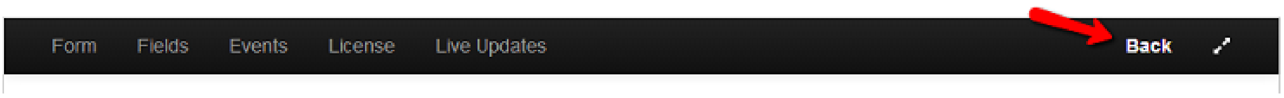
Blank Form
Creates an empty form so you build everything from scratch.
Start

Built-in Filter Search Form
This form uses two built-in search fields to filter by type (module, documnet, etc) and sub type (PDF, Word, etc).
Start

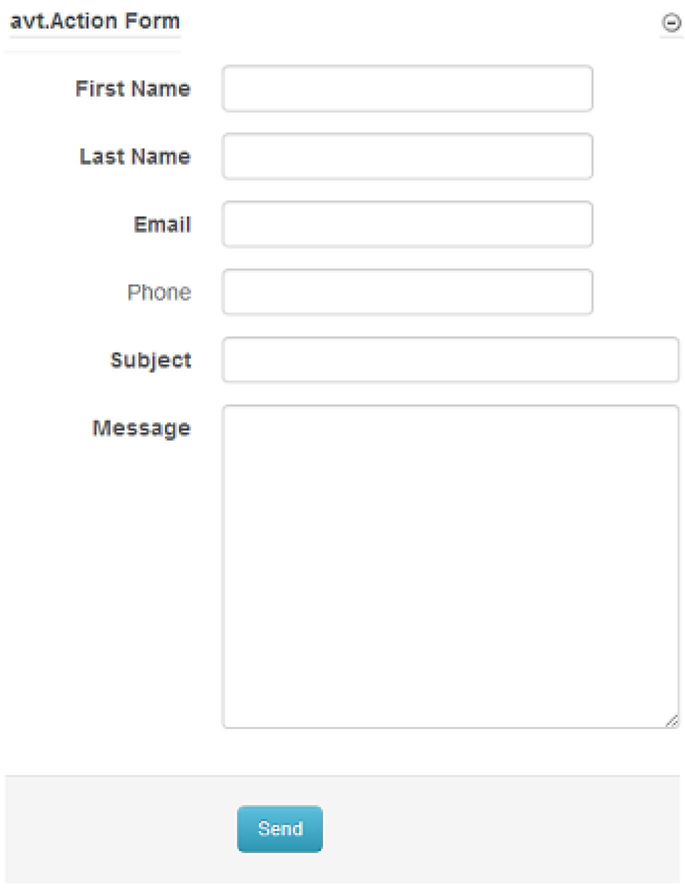
Contact Form
Builds a form where people can send a message to the portal admin.
Start

Create Lead on SalesForce
Builds a basic form for saving / loading a lead's details to / from SaleForce.
Start

5. Just by clicking on the Back button, you will have a standard contact form, like in the example below.

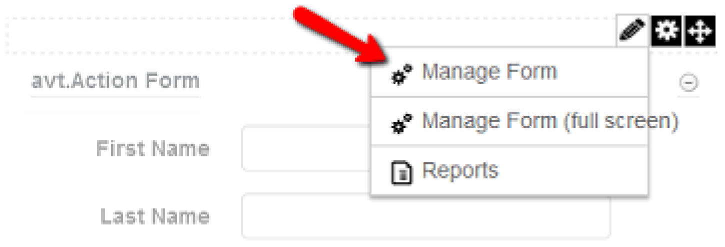


Standard Contact Form example

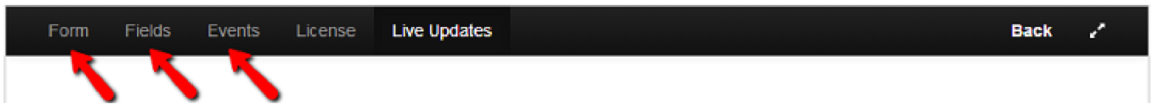


Configure Action Form Module

To manage the Action Form you've created, click on the Manage menu, then on the Manage Form link - this opens the Action Form main configuration page.



In the top menu bar, you will find 3 tabs: Form (Form Settings), Fields, Events. These are the 3 steps to configure and manage the form.

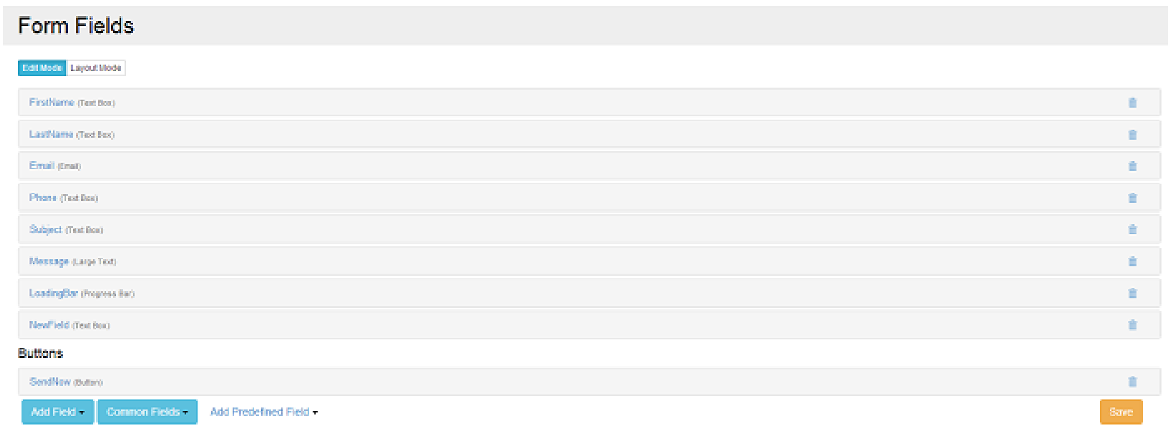


Action Form General Settings

The Action Form settings, highlighted on the image below, allows you to configure the layout and appearance of the form. These settings are accessible on the Form by going to the manage page and enlarge the General Settings list. Our Action Form module is highly configurable so it can be set on every website with different designs. More info on how to change the look of the forms can be found under Appearance and Form Layout sections.

Action Form Fields

In this section, you can change the settings for your form fields. It allows you to customize the form fields and adjust various options such as UI settings, validations, etc. You can add, remove, and reorder fields according to your needs. After you finish setting up the fields, click on Save button. More details about how fields can be configured, can be found under the Form Fields section.

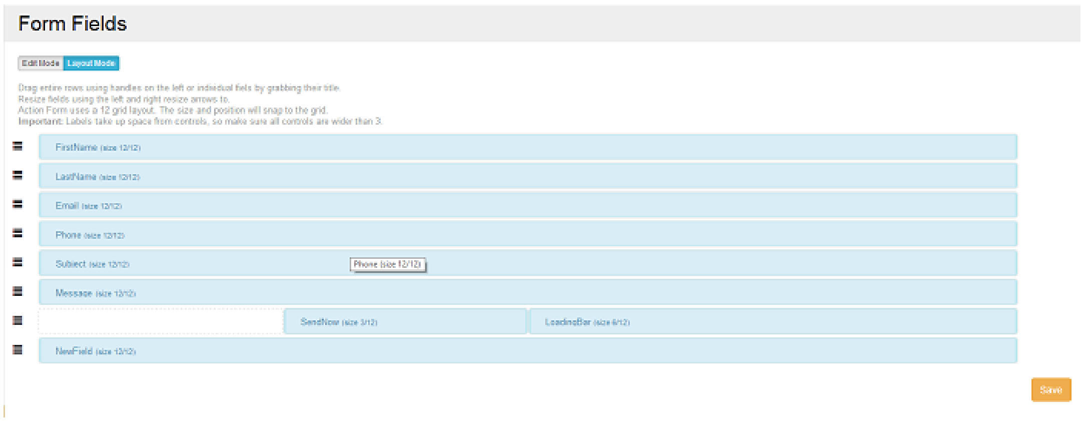


FirstName (Text Box)
LastName (Text Box)
Email (Email)
Phone (Text Box)
Subject (Text Box)
Message (Large Text)
LoadingBar (Progress Bar)
NewField (Text Box)

Buttons

SendNow (Button)

Also, if you click on the Layout Mode button, it will show you a panel from where you can configure the fields' layout, you can change their order, size, or even place them on colons. More details about this option can be found in the Form Layout section.

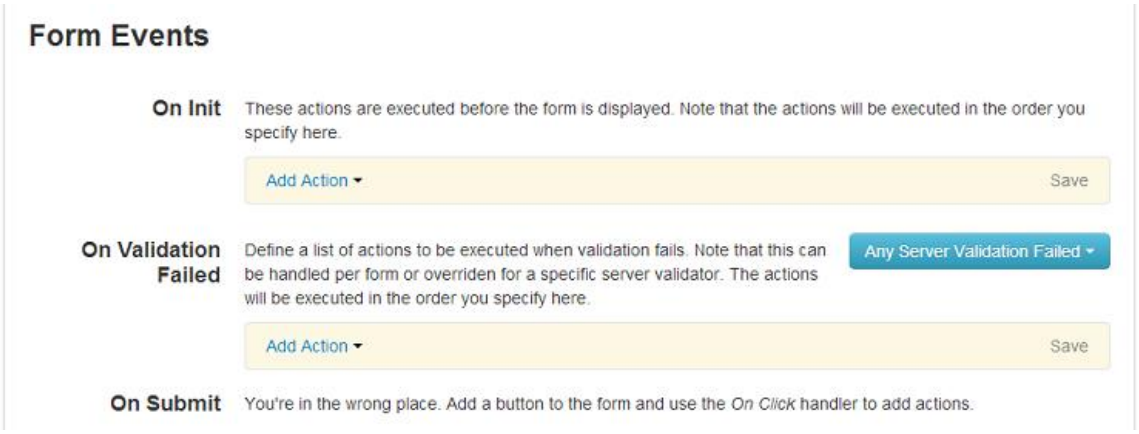

Form Fields

Drag entire rows using handles on the left or individual fiels by grabbing their title.
Resize fields using the left and right resize arrows to.
Action Form uses a 12 grid layout. The size and position will snap to the grid.
Important: Labels take up space from controls, so make sure all controls are wider than 3.

FirstName (size 12/12)
LastName (size 12/12)
Email (size 12/12)
Phone (size 12/12)
Subject (size 12/12)
Message (size 12/12)
SendNow (size 3/12) LoadingBar (size 6/12)
NewField (size 12/12)

Action Form Events

The next step to configure the Action Form is to set up the Events. It determines what happens when the form is successfully submitted by the user. Each Event has different actions, each offering specific functions such as email notifications, user registration, SQL queries and so on. More details about these options can be found under the Form Events section.


Form Events

On Init These actions are executed before the form is displayed. Note that the actions will be executed in the order you specify here.

On Validation Failed Define a list of actions to be executed when validation fails. Note that this can be handled per form or overriden for a specific server validator. The actions will be executed in the order you specify here.

On Submit You're in the wrong place. Add a button to the form and use the *On Click* handler to add actions.

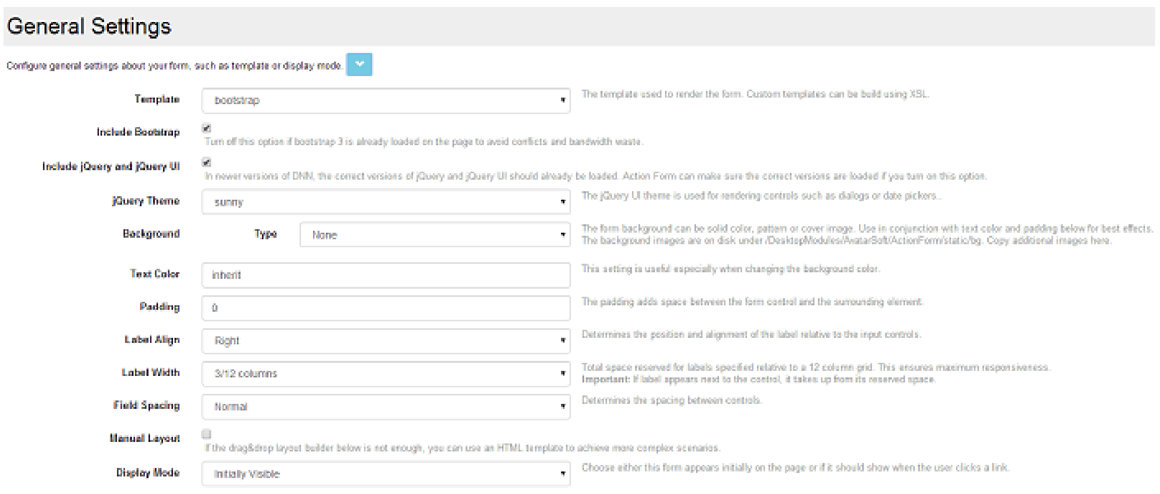
Appearance

General Settings Options

The forms, among other things, are frequently one of the main communication path between your visitors and your website. Feedback is always important, this is why is absolutely necessary to make sure that the forms you are using are easy to understand and intuitive for any user.

Your forms do not need to be boring, you can be sure that they are attractive and very effective if you use Action Form module. To get noticed, you need to come up with something unique and interesting - colors, position, size of the form are often used to reach interesting designs. Luckily, Action Form delivers this out of the box.

The General Settings, along with the Layout settings, give you the ability to set up the forms exactly the way you want so they match with your website design and to also be attractive.



General Settings

Configure general settings about your form, such as template or display mode.

Template	bootstrap	The template used to render the form. Custom templates can be build using XSL.
Include Bootstrap	✓	Turn off this option if bootstrap 3 is already loaded on the page to avoid conflicts and bandwidth waste.
Include jQuery and jQuery UI	✓	In newer versions of DNN, the correct versions of jQuery and jQuery UI should already be loaded. Action Form can make sure the correct versions are loaded if you turn on this option.
jQuery Theme	sunny	The jQuery UI theme is used for rendering controls such as dialogs or date pickers..
Background	Type: None	The form background can be solid color, pattern or cover image. Use in conjunction with text color and padding below for best effects. The background images are on disk under /DesktopModules/AvatarSoft/ActionForm/static/bg. Copy additional images here.
Text Color	inherit	This setting is useful especially when changing the background color.
Padding	0	The padding adds space between the form control and the surrounding element.
Label Align	Right	Determines the position and alignment of the label relative to the input controls.
Label Width	3/12 columns	Total space reserved for labels specified relative to a 12 column grid. This ensures maximum responsiveness. **Important:** If label appears next to the control, it takes up from its reserved space.
Field Spacing	Normal	Determines the spacing between controls.
Manual Layout	☐	If the drag&drop layout builder below is not enough, you can use an HTML template to achieve more complex scenarios.
Display Mode	Initially Visible	Choose either this form appears initially on the page or if it should show when the user clicks a link.

Bellow we'll list all the general settings options which can be found on the forms with a short description which will cover the functionality of each option - these settings can be used in order to configure the form at the level of template of display.

Template

The default template of the forms is the Bootstrap one which has the purpose of rendering the form - more on this option can be found on the Template page.

Include Bootstrap and jQuery UI

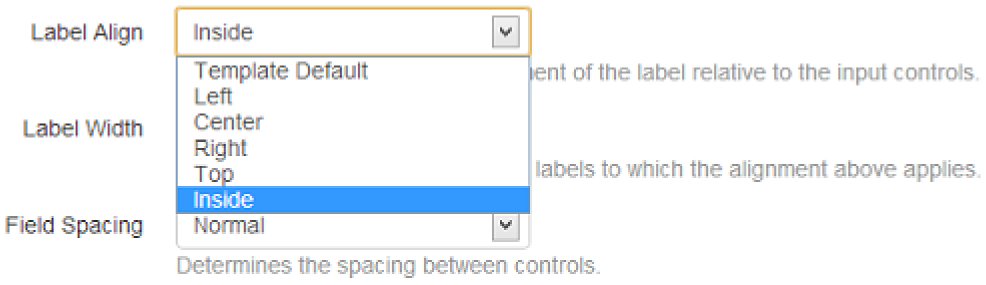
Include Bootstrap and Include jQuery and jQuery UI options are checked by default and have the role of customize the layout of the forms accordingly. They can be used along with the jQuery Theme option which will give a different appearance to the forms. Click here to read more about jQuery Theme option.

Background

The form background can be solid color, pattern or cover image and these options are best used with the Text Color options which will give a nice effect to the form if you don't want to remain to the standard default options. The usability of these settings is that you don't need to write code for each form when wanting to customize it, you just have to use these drop down lists from where you can select the option which suits you. Among the themes, background color, text color, there's also the possibly to change the padding - you can enlarge the space between the forms and the surrounding elements.

Label Align

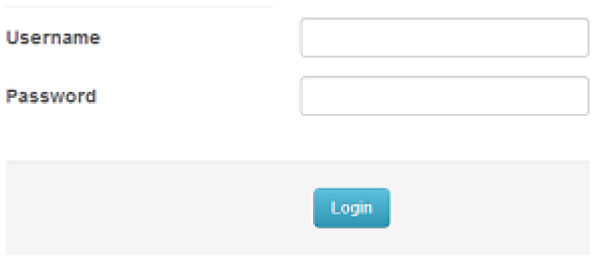
The Label Align property allows you to align the labels on the action forms. The labels are the title names for the text boxes, for example "Last Name" and "First Name". You can position the labels to the Left, Right, Center, Top, Inside. Note that the position is displayed on the space allocated in the Label Width. To select a position for your labels, select from the Label Align drop down box and specify the position.


Label Align: Inside

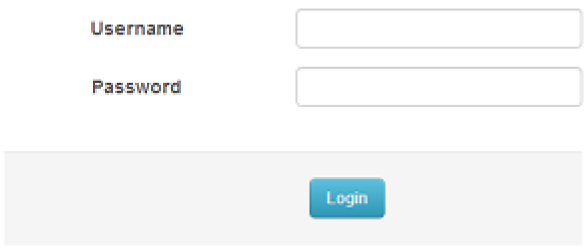
…ent of the label relative to the input controls.
Label Width
…labels to which the alignment above applies.
Field Spacing: Normal
Determines the spacing between controls.

Below are several images which exemplify the front end layout after setting different Label Alignments on the forms:

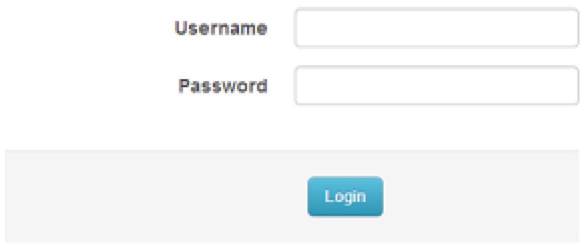
Align Label to Left



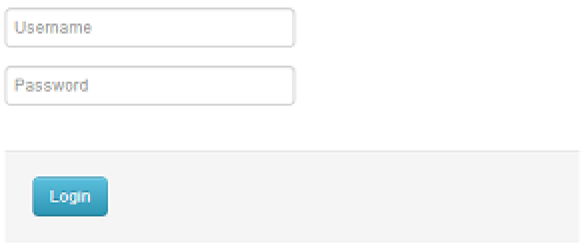
Align Label to Center



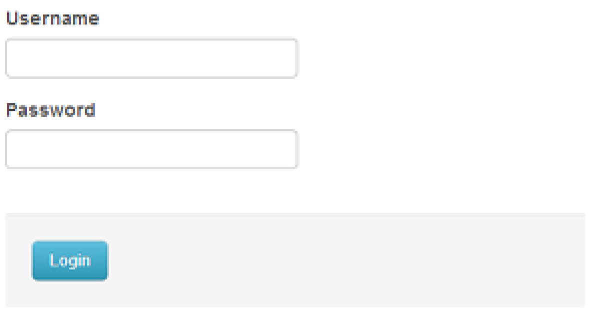
Align Label to Right



Align Label Inside

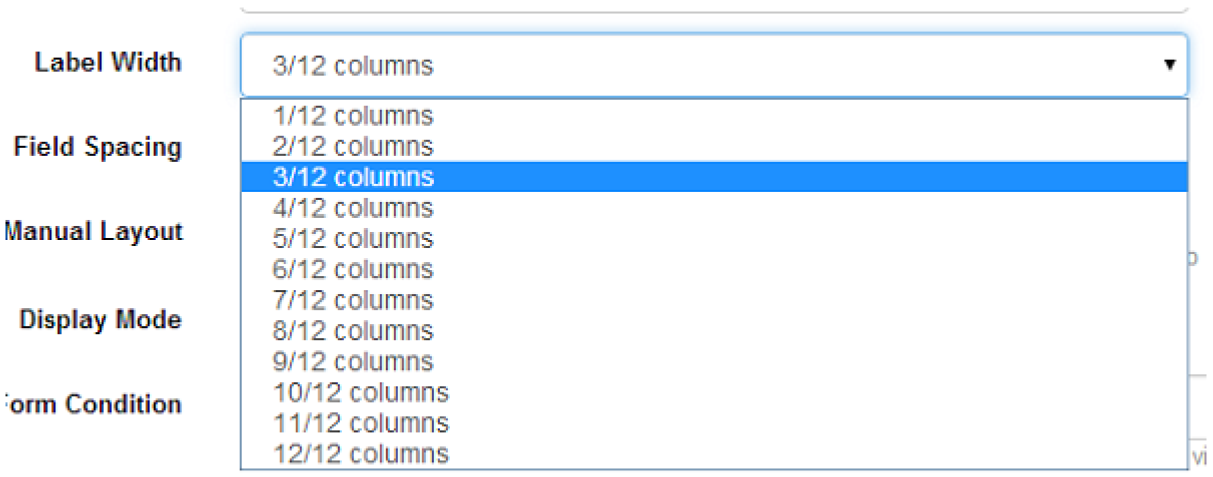


Align Label to Top



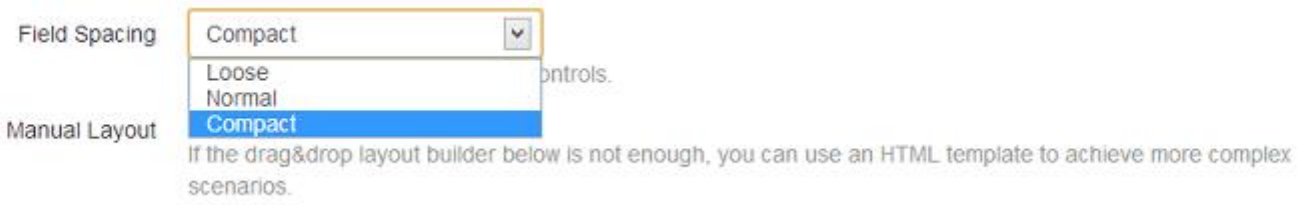
Label Width

The Label Width property allows you to specify the width of the label columns on the action form. It controls the space allocated for labels, so it can display the labels closer or farther from the controls. This option ensures the responsive layout according to each setting. Make sure that you select an option which provides sufficient space, does not overlap the labels or insert a wide space between the labels and the text box.


Label Width: 3/12 columns

Field Spacing

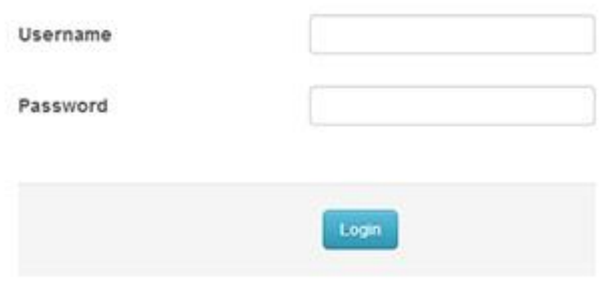
The Field Spacing determines the spacing between controls. You can chose from 3 options: Loose, Normal, Compact. This is a good option to be used when you have a big complex form because you need to save some space.


Field Spacing: Compact

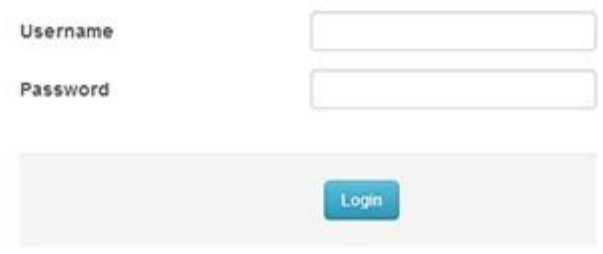
If the drag&drop layout builder below is not enough, you can use an HTML template to achieve more complex scenarios.

Here are listed several layouts where the Field Spacing options were used:

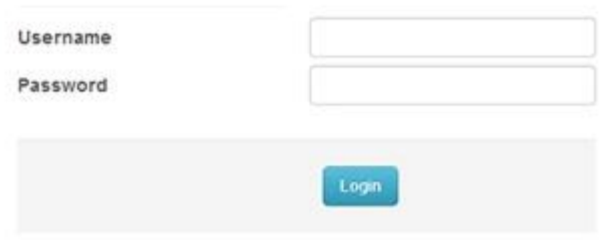
Field Spacing - Loose



Field Spacing - Normal



Field Spacing – Compact



Manual Layout

To implement more complex layout scenarios, you have the Manual Layout option which provides you with an HTML template which can be edited according to your needs in order to achieve a nicer form. When editing the HTML template you can also use Tokens for the fields you want to use. More on this option can be found on the Form Layout page.

Display Mode

This option is used to determine either if the form will be displayed on the page or if it will get displayed when the user clicks a link. More on this option can be found on the Display Mode page in this documentation.

Show Form Condition

In this box you can provide a Boolean server expression or you can use Tokens to determine if the form is visible for certain users (note that for admins the form is always visible in edit mode).

Sidebar Options

Left and Right Sidebar HTML options let you insert text which will be displayed accordingly on the form, on mobile devices, the text gets displayed on top of the form.

Initialization Scripts

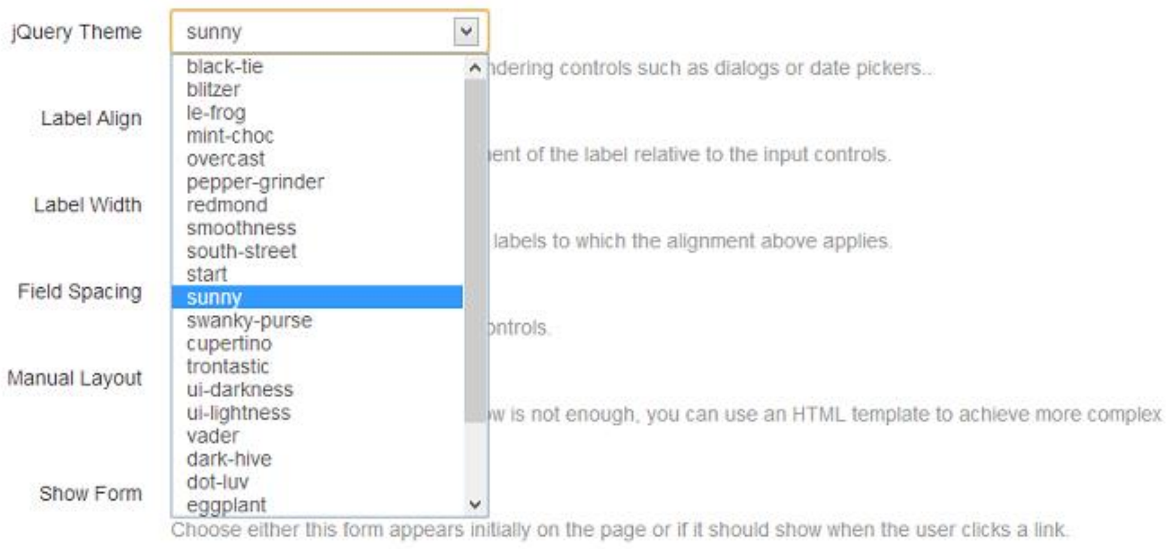
This option lets you include more scripts on the page along with the form - click here to read more on this option.

Show Tool tips

When enabled, this option will show the short description of the fields, if there is any, when the user hovers over the field Label. If a short description for the Title field is added, for example, a Name field could have a short description like "Please enter your Name.", when hovering the title of the field, this description will be displayed.

jQuery Theme

The jQuery Theme property is used for rendering controls such as dialogs or date pickers. Each theme gives a different appearance, some themes apply simple changes while others are prominent, so when you select a theme, ensure that it blends with your page and website design. Use the drop-down menu to specify a theme for your Action Form controls and then click on the save button.



jQuery Theme: sunny

black-tie, blitzer, le-frog, mint-choc, overcast, pepper-grinder, redmond, smoothness, south-street, start, sunny, swanky-purse, cupertino, trontastic, ui-darkness, ui-lightness, vader, dark-hive, dot-luv, eggplant

…dering controls such as dialogs or date pickers..

Label Align — …ent of the label relative to the input controls.

Label Width — … labels to which the alignment above applies.

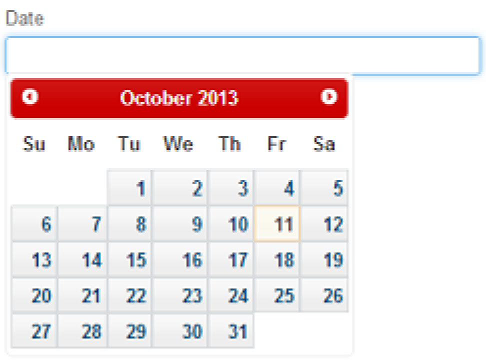
Field Spacing — …ontrols.

Manual Layout — …w is not enough, you can use an HTML template to achieve more complex

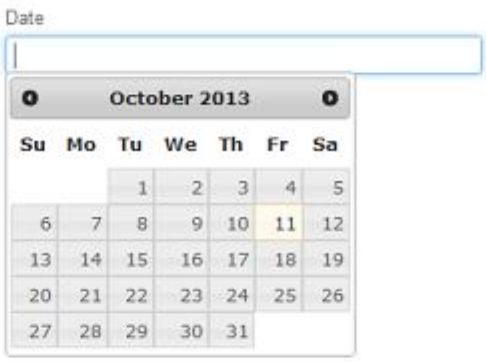
Show Form — Choose either this form appears initially on the page or if it should show when the user clicks a link.

Here are some examples applied on a calendar field:

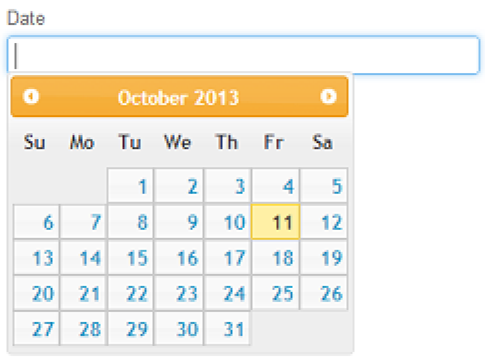
Blitzer



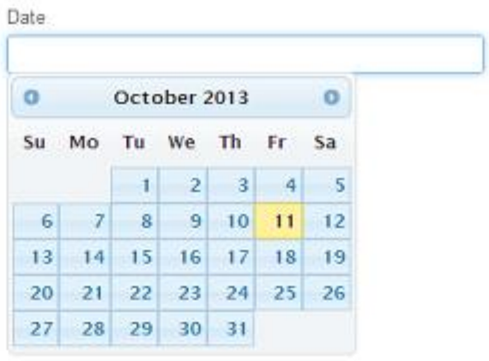
Smoothness



UI-Lightness



Cupertino



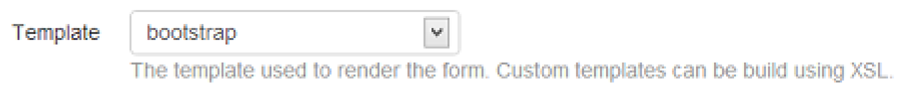
Building Custom Themes

Rolling your own theme is also possible. Create the theme using http://jqueryui.com/themeroller/, then copy it to folder \DesktopModules\AvatarSoft\ActionForm\templates\jQuery\yourthemename. Also check the other themes in the folder and make sure you follow the same naming convention for jquery-ui.css. After you complete these steps the theme will also be displayed in the jQuery Theme drop down list.

Template

As default we've set up the Bootstrap template because of some cool features that bootstrap offers: clean, fluid grid layout; responsive design; custom form elements; typography; JavaScript interaction; cross-browser compatibility, etc. This template is fully customizable, you have a full control over it, but you will need a web developer to make some changes.

Action Form produces an XML file which is used by the XSL file to define the form structure and to render the form fields - the bootstrap template it's based on this XSL file. This XML template could be visible just by putting Action Form in Debug Mode.

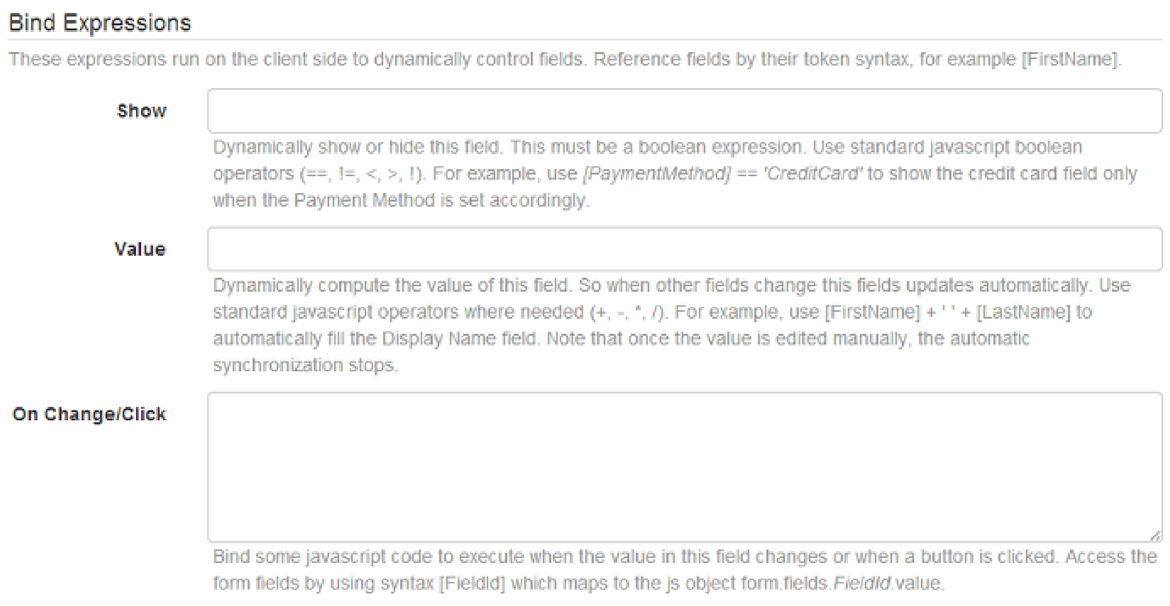

Template: bootstrap
The template used to render the form. Custom templates can be build using XSL.

Following this path: \DesktopModules\AvatarSoft\ActionForm\templates\Form, you will find the bootstrap template. You can edit this default template or just make a new one with copy/paste and renaming it so you can differentiate them. After you make these changes, your custom template will appear in the drop-down selection.

Bindings (aka Dynamic Forms)

This feature provides the ability to build dynamic forms through the use of bindings. A binding is an expression made of field values as they exist on client side when the expression is evaluated, and which affects the form state on client side, thus creating forms that dynamically change based on user input. More specifically, there are expressions that can control field visibility or the value.

Behind the scenes, Action Form uses the popular Angular JS library, but we've implemented a layer that would provide the more familiar syntax which is the field name between square brackets (eq [FieldId]).



Bind Expressions

These expressions run on the client side to dynamically control fields. Reference fields by their token syntax, for example [FirstName].

Show

Dynamically show or hide this field. This must be a boolean expression. Use standard javascript boolean operators (==, !=, <, >, !). For example, use *[PaymentMethod] == 'CreditCard'* to show the credit card field only when the Payment Method is set accordingly.

Value

Dynamically compute the value of this field. So when other fields change this fields updates automatically. Use standard javascript operators where needed (+, -, *, /). For example, use [FirstName] + ' ' + [LastName] to automatically fill the Display Name field. Note that once the value is edited manually, the automatic synchronization stops.

On Change/Click

Bind some javascript code to execute when the value in this field changes or when a button is clicked. Access the form fields by using syntax [FieldId] which maps to the js object form.fields.*FieldId*.value.

More on Expressions

These are JavaScript flavor expressions which are evaluated by Action Form to determine if a form field is visible or to auto generate a value. So the first thing you need to know about these expressions is that they are of two kinds: Boolean expressions and Value expressions. A Boolean expression evaluates to true or false and it's used by the Show binding. A value expression returns something that is stored in a field value.

These expressions are implemented on top of Angular JS data binding mechanism. Besides field value, you can invoke any JS function. For example, show a Continue button when the zip code length is 5 using this expression: [ZipCode].length == 5. Or, show it if a field is a number: !isNaN([Age]). The ability to use the JS Api is what gives limitless potential to Action Form binding expressions!

Don't let yourself be fooled by the square brackets syntax. This is just a way to reference fields. But they're still JavaScript expressions, Action Form will replace the references with constructs like form.fields.<fieldId>.value before evaluating as JavaScript. So all specific rules apply. You need to use quotes (single or double) around text, concatenate using the plus operator, compare using the Boolean operators and so on. But you don't need to be a guru in JavaScript to use bindings. In 50% of the cases all that you'll use is the == operator to show or hide fields and the plus operator to concatenate strings the set the value in another field.

Inside expressions you're limited to using operators only. You are not allowed to use control structures, to declare variables, functions and so on. You can however use the On Click/Change binding where you have full syntax.

Using Tokens

We get this often on support: "Why can't I bind using [User:FirstName]?". The answer is simple: these tokens live on the server side. We manage to fool some with this answer, but others are more alert and come with this clever question: "Ok, but what about [FormField] syntax? It works in binding expressions and pretty much everywhere else". Well, like mentioned earlier, we did extra work to implement this layer on top of Angular JS to translate [FormFileld] from tokens to angular specific code like form.fields.FormFieldId.value. But we've only done it for form fields. Invoking all tokens just to export them to JS would be overkill.

Solution 1: Hidden Fields

But that doesn't mean you can't do it. It's just that you have to do the work yourself. One way to do it is with Hidden Fields. Add one such field to your form and input the token in the value field. Now you have it on client side and can use it in expressions via the hidden field, pretty much like any other field.

Solution 2: Initialization Script

Another way to do it is to export the tokens as JavaScript variables via the Initialization Scripts section in Action Form. For example, write something like:

```
this.myUsername = "[User:Username]";.
```

Notice the using of this keyword. It refers to the Angular JS scope. Every time you use a variable in an expression, it's fetched from this scope. So now that we've placed myUsername in the scope, we can later use it in an expression just by its name. So one field could be made visible with the following expression: myUsername == 'host'.

Some very important side notes on this technique. You can also "import" global variables into Action Form. So writing something like this.myPage = g_tabId; would bring the value of a global variable to be used inside the expressions. And you can also place functions in this. For example,

```
this.myFunc = function(args) {
   // do something
};
```

And then, just invoke the function from an expression just as you'd invoke any function in JavaScript.

Visibility

This binding dynamically shows or hides fields based on a Boolean expression. Access field values by their name between square brackets and use standard JavaScript Boolean operators (==, !=, <, >, !). For example, use [PaymentMethod] == 'CreditCard' to show the credit card field only when the Payment Method is set accordingly. More complex expressions are possible by using the logical and operator (&&) and the or operator (||).

Dynamically show or hide this field. This must be a boolean expression. Use standard javascript boolean operators (==, !=, <, >, !). For example, use *[PaymentMethod]* == *'CreditCard'* to show the credit card field only when the Payment Method is set accordingly.

Here are a few different scenarios that use this binding.

- **Based on a check box**

Because a check box is already a Boolean, the expression can be as simple as [CheckboxFieldId].
This is equivalent to: [CheckboxFieldId] == true

- **Based on a radio button**

The default radio box is called Yes/No. So the expression must check the field value against a string like this:

[RadioFieldId] == 'Yes'

- **Based on Multiple Choice fields**

The multiple choice field allows administrators to provide a list of items. The expression need to compare against the item value. You have to pay attention to this only when using the pipe syntax to provide both a text and a value for an item (for example Item Text|value here). Here's an example: [MultpleChoiceFieldID] == 'Some Value'.

- **Complex Expressions**

So far you've seen some simple examples. Often you'll need to take into account more that one field. For example, to show a button after a user has ticked the Agreed to Terms check box and has written his or her name in the signature field, use this expression:

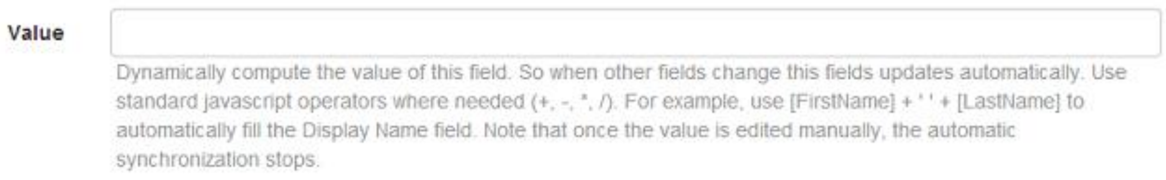
[AgreedToTerms] && [Signature] == [Name]

If you need to join more expressions, also use round brackets to group conditions in order the define priorities. For example, [AgreedToTerms] && ([Signature] == [Name] || [SignLaterAtYourOffice])

Value

This binding is used to auto-populate a field based on values in other fields. Let's take the most natural example in DNN:

[FirstName] + ' ' + [LastName]


Value

Dynamically compute the value of this field. So when other fields change this fields updates automatically. Use standard javascript operators where needed (+, -, *, /). For example, use [FirstName] + ' ' + [LastName] to automatically fill the Display Name field. Note that once the value is edited manually, the automatic synchronization stops.

This is the expression that you would bind to the Display Name field. It simply concatenates the First Name with Last Name and separate them with a white space. The field will update automatically every time first name or last name fields change. But there is a catch. Once the user "touches" the Display Name field the value is no longer updated automatically. Currently there is no way to reactivate the binding once that happens. If you do need to do this, then take a look at creating the binding manually in the On Click/Change section further down on this page.

You've already seen the concatenation operator in the example above. Here are more usages of this feature:

- **Conditional binding**

Here's a classic example with two fields, call them Source and Target, that need to be synchronized when a check box called Sync is ticked. For this purpose, you can use the conditional operator from JavaScript and write the following expression in the Target field:

[Sync] ? [Source] : "".

This simply says "If Sync is checked, then use the value from Source field, otherwise use nothing". There is a video on this example at youtu.be/iOjGys1NVmg and also some files that you can find attached to this page.

- **Parse values**

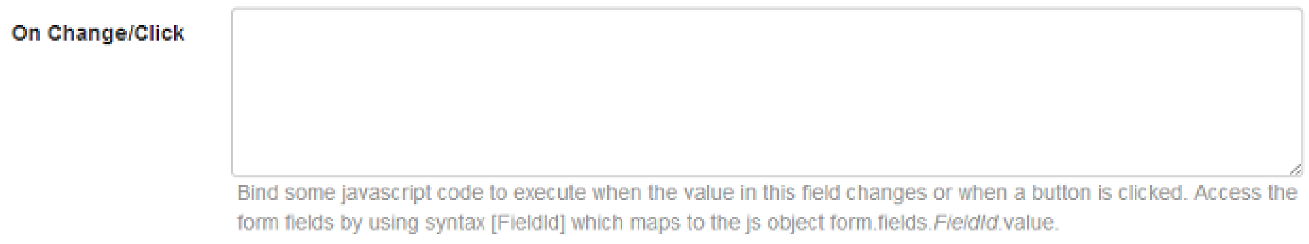
Think a common example with ID numbers which sometimes contain the date of birth. Why not parse this and populate the date of birth field automatically? With an ID like 20010110xxxxxxxx, where we have YYYYMMDD at the beginning, we can extract the date using the JavaScript subtr function. So, to the date time field that accepts the MM-DD-YYYY format, bind the following expression:

[IdField].substr(4, 2) + '-' + [IdField].substr(6, 2) + '-' + [IdField].substr(0, 4)

All this does is take parts of the ID fields and concatenate them together but in a different order and separated by a hyphen.

On Click/Change

This binding is used to bind some code to be executed when a button is clicked or a field value changes. This allows for more advanced scenarios compared to the other bindings and you can use the full JavaScript capabilities, including conditions and loops and so on. In this context, this refers to the field that raised the event, more specifically to the jQuery object built with this DOM object.


On Change/Click

Bind some javascript code to execute when the value in this field changes or when a button is clicked. Access the form fields by using syntax [FieldId] which maps to the js object form.fields.*FieldId*.value.

Here are some common scenarios:

- **Advanced bindings**

The example in the previous section can be rewritten like:

```
if ([Sync] == 'Yes')
    [Target] = [Source];
```

This is somehow cleaner code and it also bypasses the Action Form mechanism to break the binding after the Target field is "touched". It also allows for more advanced processing so values can be compared, iterated, parsed and so on before being bound again to a field. There is a video on this example at youtu.be/iOjGys1NVmg and also some files that you can find attached to this page.

- **Fetch data from Web Service**

Now this gets interesting! Let's say we have two fields, Source and Target. We want to update the Target with the value from a web service whenever Source changes. To make the request we can use the $http service from Angular JS. This is already included by Action Form, we can just use it in the On Click/Change binding of the Source field:

```
$http({method: 'GET', url: '/someUrl'})
    .success(function(data) {
        [Target] = data;
    });
```

More services are available in Angular JS, but not all of them are exposed by Action Form. Write to us below in comments or via email if you need more services.

- **Custom bindings**

In a future version we'll add more bindings to control CSS classes, to disable fields and so on. But until then you can achieve all of these manually via this On Click/Change binding. Consider the following example:

```
if ([Sync])
    $(this).addClass('field-synched');
```

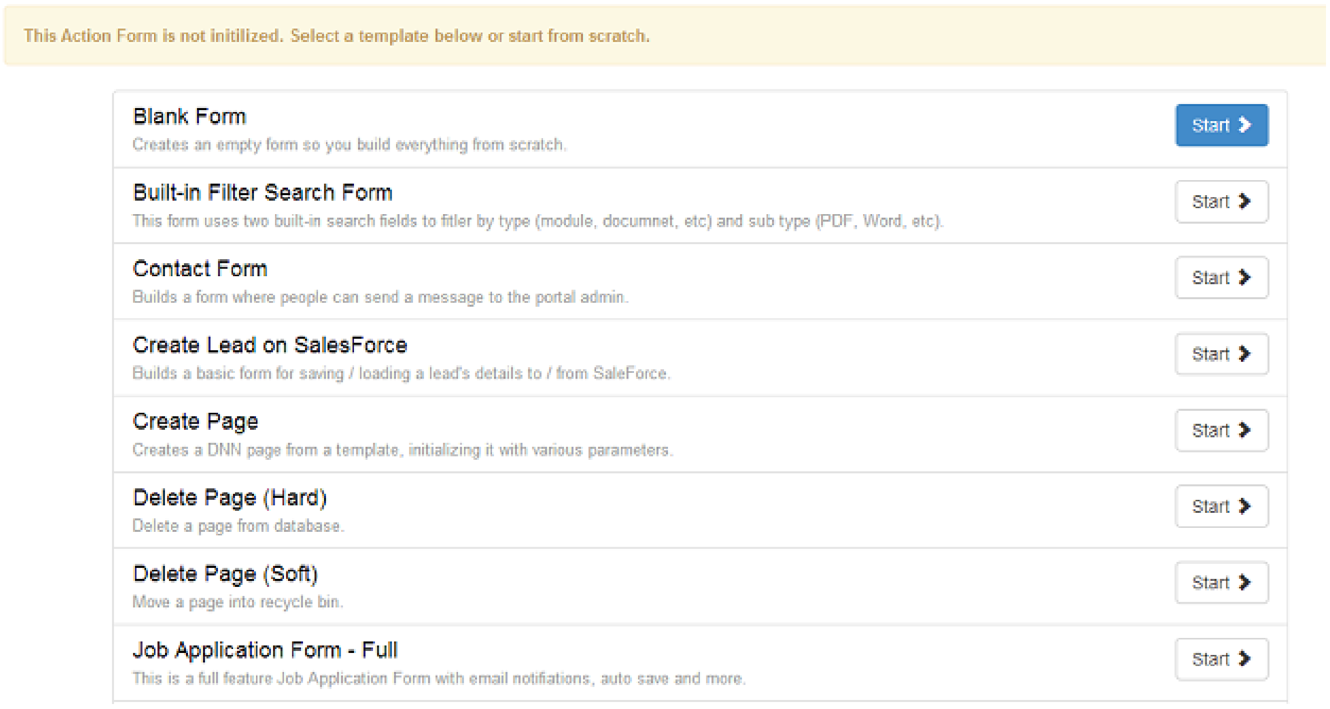
What this does, it adds a class to a field when it changed and the Sync check box is ticked. Note the use of this in this context refers to the form field that generated the change event.

Creating a New Form

Easy-to-use, Action Form will help you to generate and manage various forms in your DNN portal in less than 1 minute. Contact form, mailing list, survey form, application form, event registration form, are just some of hundreds of forms that this single module can help you to create and control. We've spent a lot of time making the interface friendly and intuitive. Using a simple AJAX admin console it takes just minutes to set up even most complex forms with no programming and technical experience required.

Check out the Getting Started guide to find more details on how to add Action Form to your page. You can have an Action Form module fully functional simply by going to the Manage Form main page. Action Form comes with several predefined form templates: Contact Form, Log in Form, Registration Form, Subscribe to MailChimp etc. You can start just by using one of this predefined templates or you can start a new one from scratch, so that you can configure it just the way you need it. You can also modify the default templates to suite your needs.

To use a default form template or creating a new one, click on the Start button.



This Action Form is not initilized. Select a template below or start from scratch.

Blank Form
Creates an empty form so you build everything from scratch.

Built-in Filter Search Form
This form uses two built-in search fields to fitler by type (module, documnet, etc) and sub type (PDF, Word, etc).

Contact Form
Builds a form where people can send a message to the portal admin.

Create Lead on SalesForce
Builds a basic form for saving / loading a lead's details to / from SaleForce.

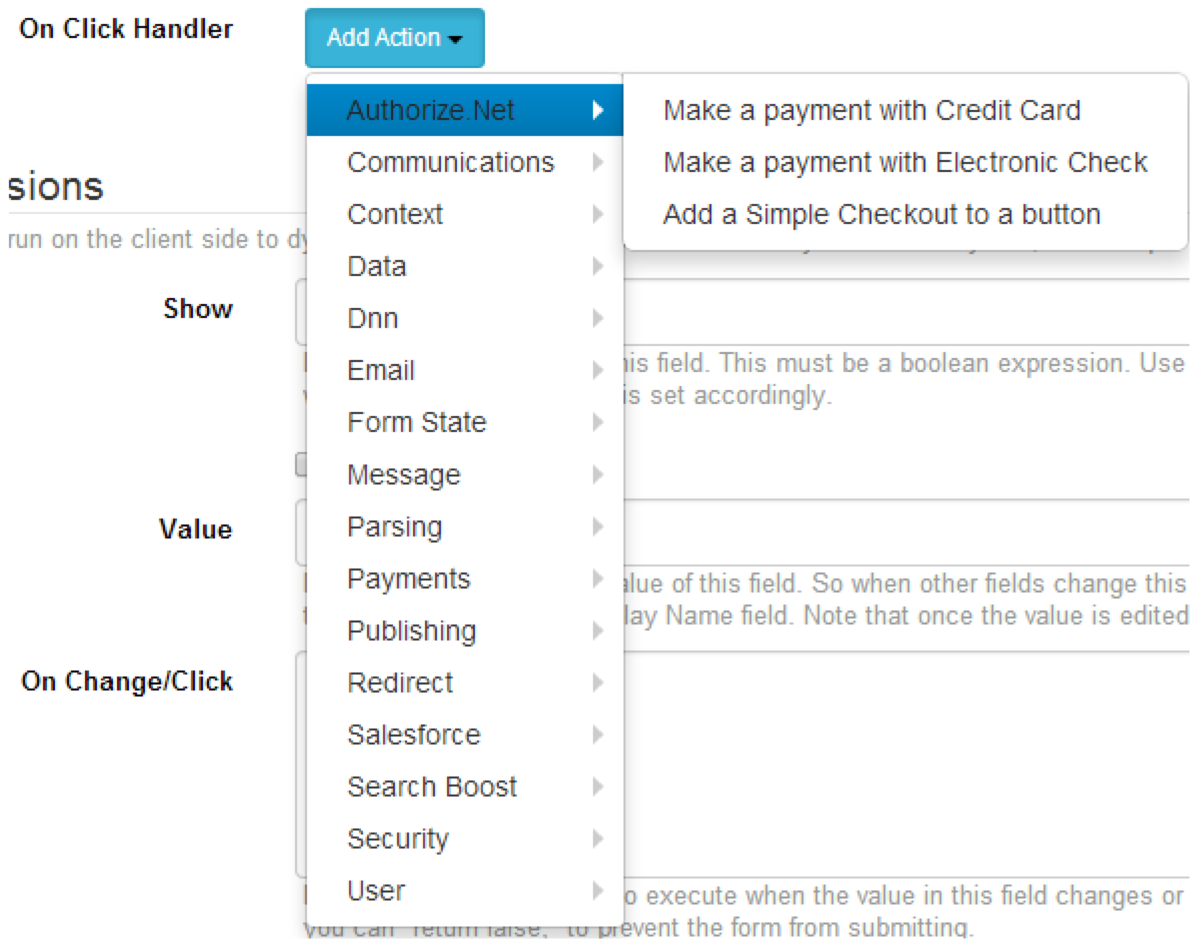
Create Page
Creates a DNN page from a template, initializing it with various parameters.

Delete Page (Hard)
Delete a page from database.

Delete Page (Soft)
Move a page into recycle bin.

Job Application Form - Full
This is a full feature Job Application Form with email notifiations, auto save and more.

Note: After installing the Action Form add-on modules, they will be displayed as Actions on the button fields in the On Click Handler list:



run on the client side to dy

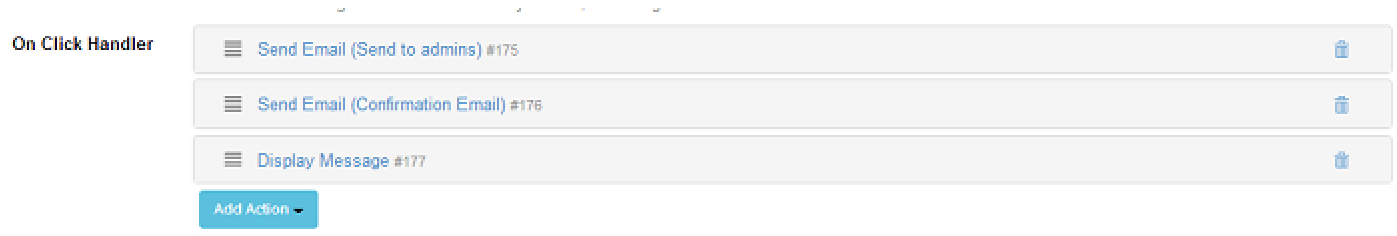
is field. This must be a boolean expression. Use
is set accordingly.

alue of this field. So when other fields change this
lay Name field. Note that once the value is edited

o execute when the value in this field changes or
you can "return false;" to prevent the form from submitting.

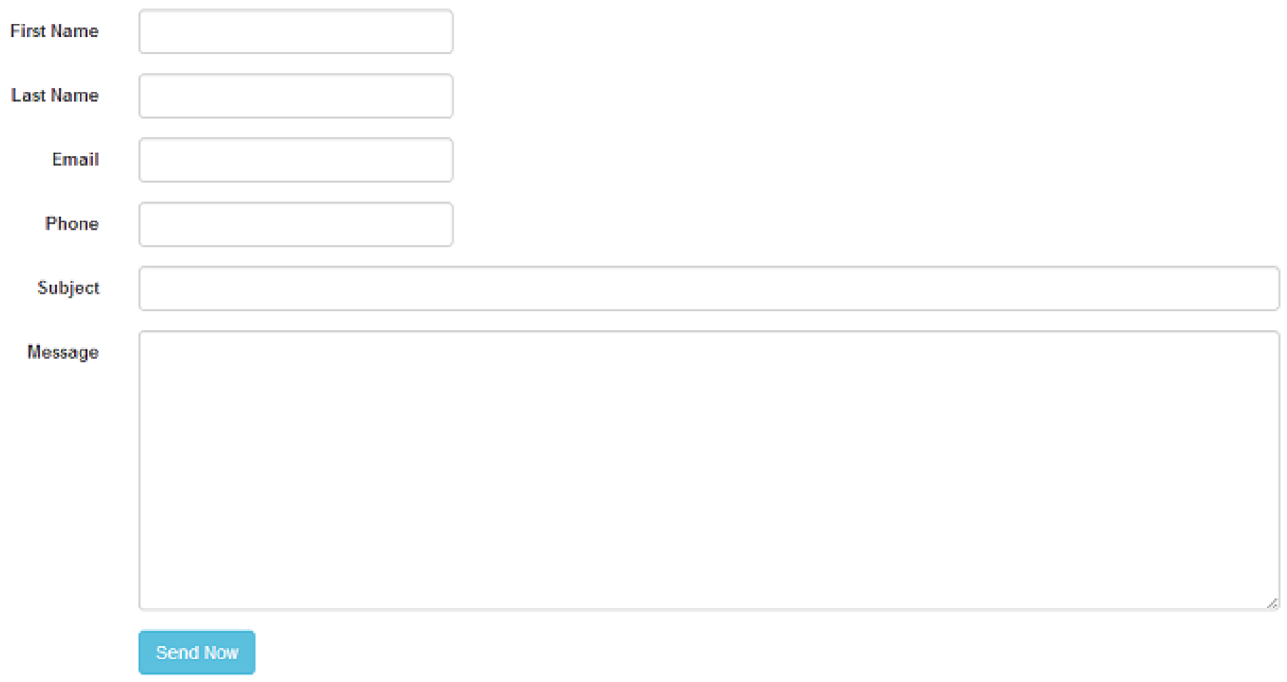
Contact Form

The Contact Form allows you to implement a feedback form to a web page or a post in no time. It is an easy form, it builds a form where people can send a message to the portal admin. It doesn't require any additional settings, though there are some available options. All you need is to activate the form from the first panel and check the email settings - where the messages are sent - to a user specific email address or any email address. Check out this video to see how to set up the email so that you don't encounter any problems when the form is submitted.


On Click Handler
Send Email (Send to admins) #175
Send Email (Confirmation Email) #176
Display Message #177

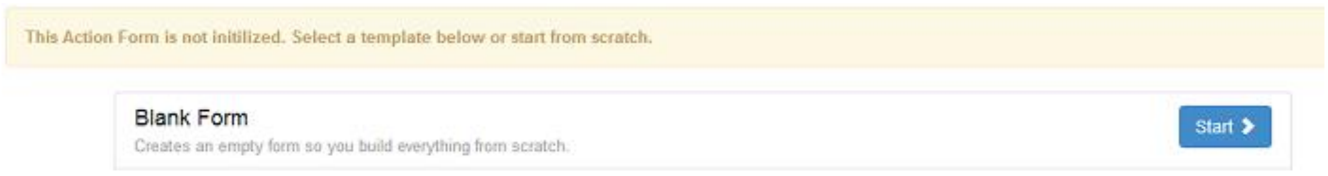
This default template can also be customized if you think you'll need some additional fields, delete some of the existing ones or change the UI by inserting some CSS Styles. If you click on the Layout Mode button, you will have the ability to rearrange the fields' order. More info regarding this feature can be found on the Form Layout page.

And here's how the Contact form template looks as final product:



Create From Scratch

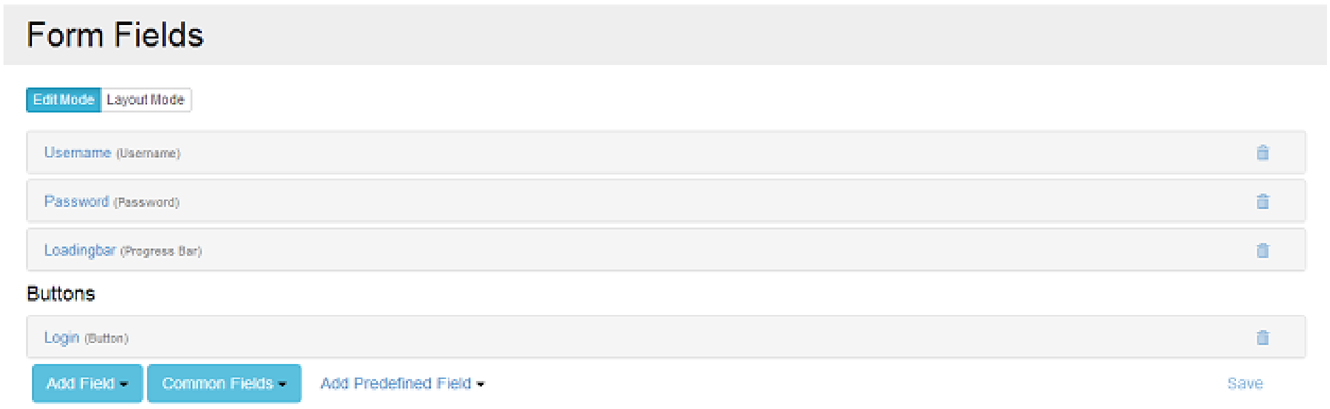
When you want to create a form from scratch, these is the Blank Form template option which let's you add any type of field(s) that you need. By simply adding the Action Form module on the page and going to manage form page, you have to click on start button displayed for the Blank template.


This Action Form is not initilized. Select a template below or start from scratch.

Blank Form
Creates an empty form so you build everything from scratch.

Login Form

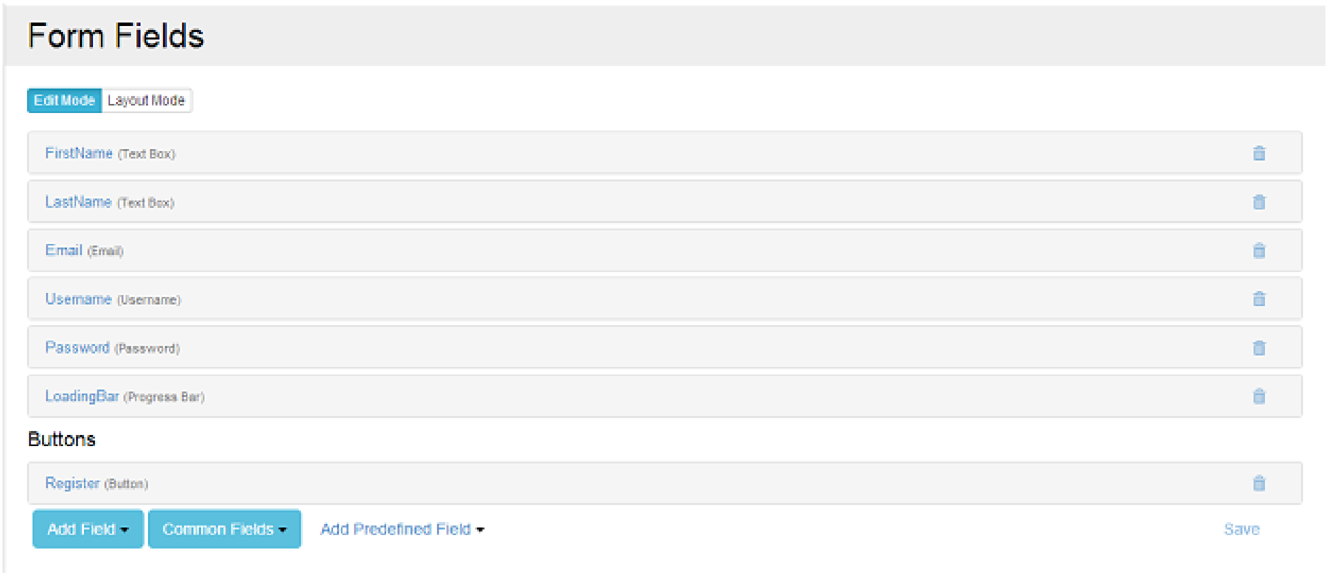
Using a log in form for authentication is a common and flexible method for handling authentication. The predefined Login Form is a simple template with username and password fields which have the purpose to log the user in the account. Pretty much every aspect of the form log in can be customized. This predefined form template is a simple form that uses two fields, Username and Password and on button click handler there is a User Login action.


Form Fields

Username (Username)
Password (Password)
Loadingbar (Progress Bar)
Buttons
Login (Button)

Registration Form

Action Form comes with a Registration Form template already created for you. This offers a smoother user experience for those looking to sign up for a website - if a user doesn't have an account but would like to create one, this will bring him to the registration page. Some surveys suggest that websites with shorter registration forms have larger lists of new users on a daily basis. This is an explanation for why we have created a smaller registration form for you to use. The Registration form template creates a basic form which collects the Name, Email, User and Password, creates a new user account and sends a standard registration email to users. Like all the other forms, the registration form is also customizable, so you can add, remove, add CSS styles, reorder the fields.



Form Fields

FirstName (Text Box)
LastName (Text Box)
Email (Email)
Username (Username)
Password (Password)
LoadingBar (Progress Bar)

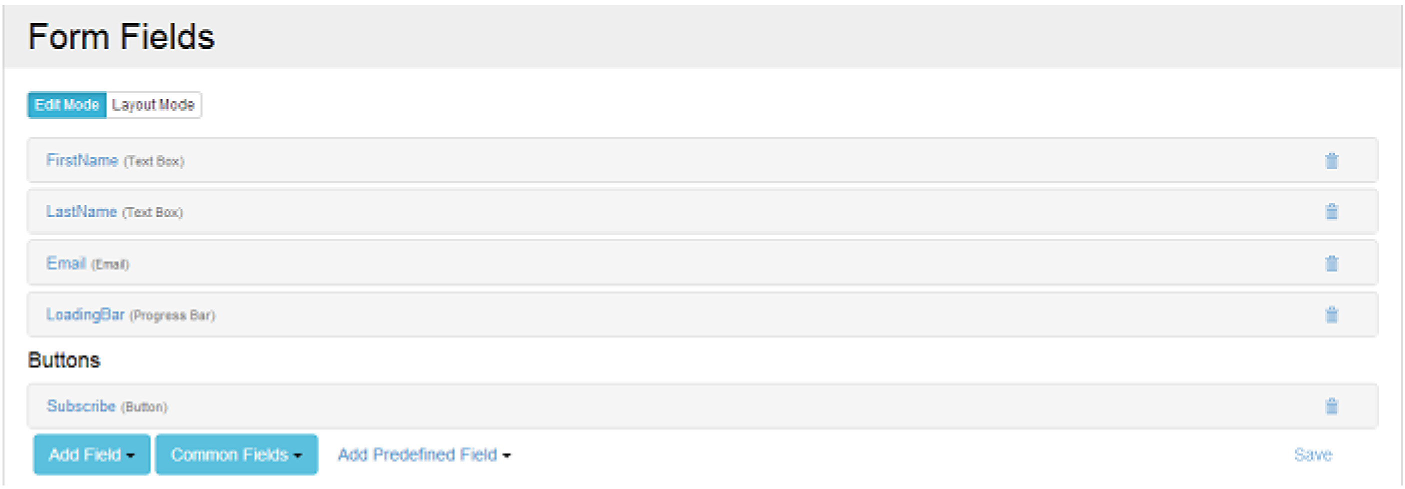
Buttons

Register (Button)

Subscribe to MailChimp

The Subscribe to MailChimp predefined form allows you to quickly and easily add a signup form on your site for your MailChimp list. This form subscribes an email address to a Mailchimp list along with the First and Last Name. Before starting to use it, make sure to configure your API Key (go to Account > Extra > API Keys) and List Name (it has to be exactly as it appears in MailChimp) under the Subscribe button. Follow this link to find out more about MailChimp API Key.

Action Form requires an API key to successfully connect to a MailChimp account and transfer subscriber information and other data. Click here to see a video tutorial where we use the Subscribe to MailChimp template form.



Form Fields

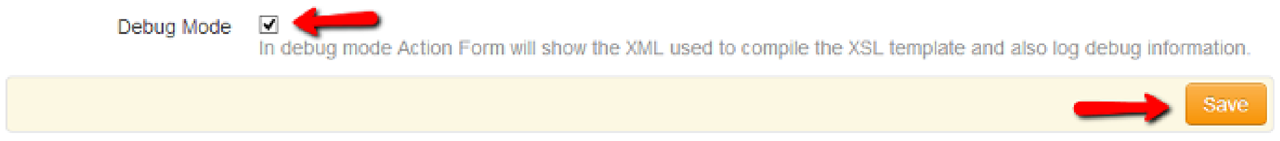
FirstName (Text Box)
LastName (Text Box)
Email (Email)
LoadingBar (Progress Bar)

Buttons

Subscribe (Button)

Debugging

In Action Form, you have the option to put the form in Debug Mode so you can trace any anomaly that is bothering you. We set up this option so that you can easily see the XML file with the data structure of the form. This info is very useful when you create your own template or when you try to customize the default templates. It is also very useful to track any mistakes you've made when you've created a new form. You just have to check the box for the Debug Mode option and click on the Save button.



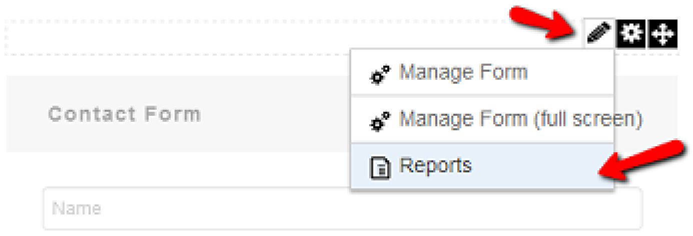
Debug Mode ✓
In debug mode Action Form will show the XML used to compile the XSL template and also log debug information.

For front end you can use browser web development tools like Firebug that facilitates the debugging, editing, and monitoring of any website's CSS, HTML, DOM, XHR, and JavaScript.

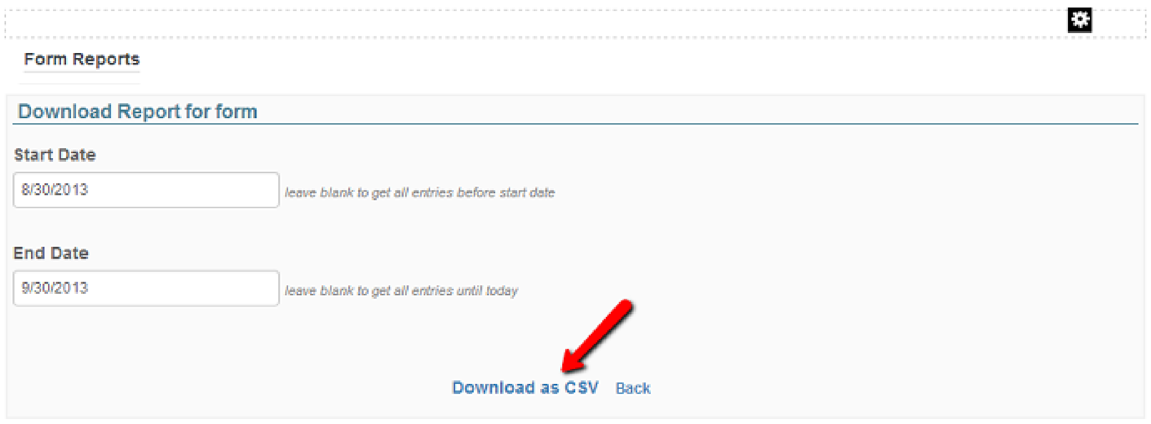
Debugging is also a methodical process of finding and reducing the number of bugs or defects. Normally the first step in debugging is to attempt to reproduce the problem. If you are unlucky and you're encountering problems, you can generate some reports and send the files to us via email (or setup a place where we can download the reports if the files are too big) and we'll do our best to find a solution. When submitting debugging information to us for investigation, please make sure to provide the following:

- Steps to reproduce the error, if possible.
- Admin access to your site - we need this to reproduce the issue by ourselves on your environment because sometimes, an issue is dependent on the environment settings and there is a high change to not reproduce the issue on our side, so isolating it on your environment is the best choice if you need a fix for a problem.
- When an error occurs, go to Admin -> Event Viewer and locate the error message along with the Stack Trace; for the error to be relevant, the Stack Trace must contain the word ActionForm.
- We also need you to provide us with the DNN version where you encounter the problem.
- If possible, export the form settings/content so we can see the exact settings you have made on the form. You can export these settings following the following steps: click on the manage menu -> settings -> export content -> select the destination where you want the file to be exported. Usually, the exported file can be found in the root folder of your site.
- Send us the Report in order to review it. You can do this by exporting the CSV Report. See the images below:

Click on Reports option



After selecting the start and end date, click on Download as CSV


Form Reports

Download Report for form

Start Date	8/30/2013	leave blank to get all entries before start date
End Date	9/30/2013	leave blank to get all entries until today

Display Mode

Usually, when a page contains the Action Form module, the created form is displayed as the page loads. This is the standard and most common way to display the forms. But sometimes you would want to display the form in a popup page when the user clicks on a link, or just display some text and redirect the user to a different page where the actual submission of the form will be made.

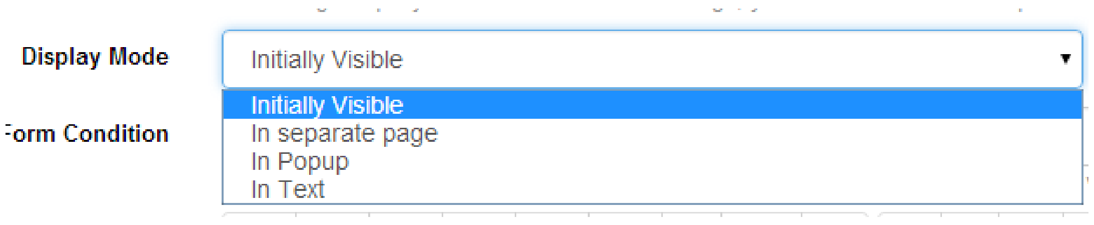
All these options are possible on Action Form module and are configured from the Display Mode option.

Initialization Options

- **Initially Visible** - this option displays a static form.
- **In separate page** - selecting this option and linking the text to a web page will make the form to open in a separate page - when you click on the hypelink, you'll be redirected to another web page.
- **In Text** - displays the form on the same page it was added after you click on the hyperlink.
- **In Popup** - displays the form in a popup page.

You will need to specify one of these options to define how you want the form to appear when the user clicks on a link. By default, the form uses the first option - Initially Visible, so the form is displayed as a static form when the page loads.

You can access these options by clicking on the drop down box on the Display Mode option (see the image below).

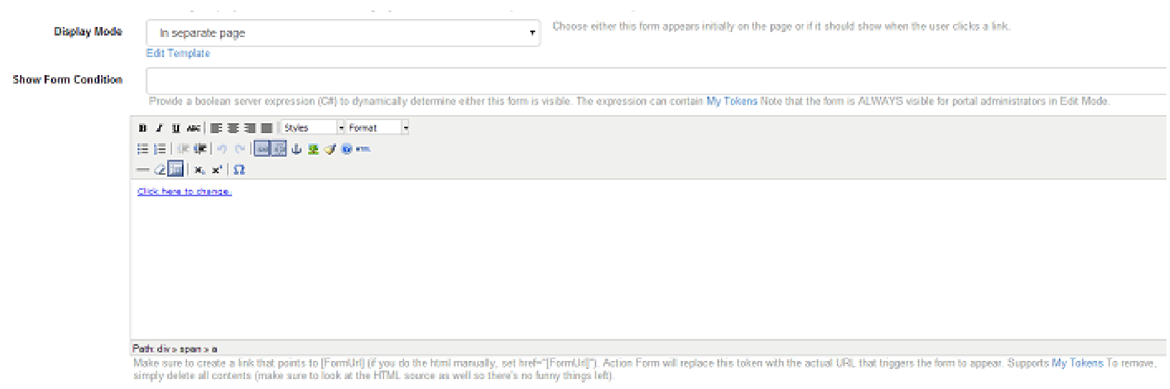


Initially Visible

This option is used renders the form while the page loads, this is the default display mode. To see how this option works, you can test it compared to the other Display Modes which open the form in another web page or replaces the text with the actual form when a link is clicked.
Select Initially Visible from the drop down box.
When you are done, click on Save.
When the form is completely saved, click on Back button and while the Manage Action Form page closes, you are returned to the main page.

In separate Page

The In Separate Page option will instruct Action Form to redirect to a new page for actual page submission. You can create a text that is the main content, which also contains a special link that will get replaced with the actual URL where the form is located.



Display Mode: In separate page

Edit Template

Choose either this form appears initially on the page or if it should show when the user clicks a link.

Show Form Condition:

Provide a boolean server expression (C#) to dynamically determine either this form is visible. The expression can contain My Tokens Note that the form is ALWAYS visible for portal administrators in Edit Mode.

Click here to change.

Path: div > span > a

Make sure to create a link that points to [FormUrl] (if you do the html manually, set href="[FormUrl]"). Action Form will replace this token with the actual URL that triggers the form to appear. Supports My Tokens To remove, simply delete all contents (make sure to look at the HTML source as well so there's no funny things left).

1. Select the In Separate Page option from the Show Form drop down box.
2. Click on Edit Template

- The drop down box opens the text box editor. It shows a sample text on which an URL is linked - this will make the redirect to the form when the user clicks on the hyperlink.
- You can edit this text and replace it with a description that best describes your action form. Make sure you keep the target link to [FormUrl] and if you do the HTML manually, set href="[FormUrl]". Action Form will replace this token with the actual URL that triggers the form to appear. When you are done, click on Save button.

In Popup

The Popup option creates a modal JavaScript dialog box. It opens the form when the text link from the page is clicked and it also allows you to specify the width and the height of the dialog box.

The popup option includes the same steps as the In Text and Separate Page options. The only difference is made by the specifications for the width and height of the dialog box. The results are also similar to the In Text and Separate Page options, except that the popup effect and the size can vary for the form.

In text

The In Text option comes with a default text, which can be changed, and which contains a link that when is clicked, the text hides and the form gets displayed instead of the text. This option could be very useful for when you have a page that is populated with a lot of content. In this way you can save space and make the page look great for the users.

The form on which the In Text option is applied, behaves quite similar to Separate Page option. The only difference is the target location, the Separate Page option opens the form on a new page, but the In Text option opens the form on the same page. The steps necessary for setting the In Text option on a form are the same steps described and listed for the Separate Page option.

Extensibility

Action Form features an open configuration architecture where new functionality can be added just from the configuration files. This makes it a very powerful framework as well. As part of the architecture we've addressed a few hot topics that frameworks usually have:

- **Extensibility.** Action Form implements a decoupled interface where components (such as fields, actions, validations, templates) are created at run time via reflection based on the XML configuration.
- **Continuous Updates.** Custom components are not added directly to the main configuration file that comes with Action Form.

Custom Actions

Action Form is build on an open configuration architecture, that makes it very configurable. We have developed it in this way because we know how important is for you to have the ability to customize it with new functionalities which suits you needs.

Custom Predefined Forms

Action Form comes with a few predefined forms such as the Contact form, registration, login forms. You can easily create new predefined forms so you don't have to start from scratch every time you create a new Action Form. But for now, this needs to be done manually starting from an export you do from Module Actions > Export Content.

Extensions

DNN Authorize.Net Add-on

Connect your website or business application to the Authorize.Net® Payment Gateway for processing online payments!

DNN Authorize.Net Add-on allows you to Accept Payments Online in your DNN plaform. You can now accept credit cards* and electronic checks** through your Authorize.Net Payment Gateway account.

*All major credit cards: Visa®, MasterCard®, American Express®, Discover®, Diner's Club, JCB + Signature Debit Cards
**Authorize.Net eCheck.Net®

Authorize.Net is a registered trademark of CyberSource, a Visa company.

Make a payment with Credit Card, CC

This type of payment allows one to make a payment with Credit Card through Authorize.Net's Payment Gateway, using Authorize.Net's Advanced Integration Method (AIM).

The security of an AIM transaction is ensured through a 128-bit Secure Sockets Layer (SSL) connection between the merchant’s Web server and the Authorize.Net Payment Gateway.

There are different credit card transaction types supported by the payment gateway and they each have specific field requirements: Authorization and Capture, Authorization Only, Prior Authorization and Capture, etc.

Make sure you already read the Authorize.Net manual prior to use different credit card transaction types and specific fields: http://www.authorize.net/support/AIM_guide.pdf

For your convenience, DNN Authorize.Net Add-on already contains a template to get you started, named 'Pay With Credit Card, Authorize.Net'.

It is a form that will do an Authorization and Capture transaction. This is the most common type of credit card transaction and is the default payment gateway transaction type. The amount is sent for authorization, and if approved, is automatically submitted for settlement.

Requirements

- The merchant must have a merchant bank account that allows Internet transactions.
- The merchant must have an e-commerce (Card Not Present) Authorize.Net Payment Gateway account.
- The merchant must have a valid Secure Sockets Layer (SSL) certificate and their Web site must be capable of initiating both client- and server-side SSL connections.

- The merchant must be able to store payment gateway account data securely (for example, API Login ID, or Transaction Key).
- The merchant must have Action form module installed.

Getting started

Step 1 - Sign Up and Activate an Authorize.Net Account.

You will need both a merchant account and Authorize.Net Payment Gateway to accept credit cards.
You can sign up here: https://www.authorize.net/signupnow/

Step 2 - Get the API Login ID and Transaction Key from Authorize.Net.

These keys will authenticate requests to the Authorize.Net Payment Gateway.
Get the id and the key if don't already have them. You will find them in your ACCOUNT -> Settings -> Security Settings -> General Security Settings -> API Login ID and Transaction Key.

Your API Login ID and Transaction Key are unique pieces of information specifically associated with your payment gateway account. However, the API login ID and Transaction Key are NOT used for logging into the Merchant Interface. These two values are only required when setting up an Internet connection between your Web site and the payment gateway. They are used by the payment gateway to authenticate that you are authorized to submit Web site transactions.

Step 3 - Setup Action Form

Add an Action Form module to your page, go to Manage Form and select a Authorize.Net template to get started.
Set your API Login ID and Transaction Key.
Set the description and total ammount for your item.
Add any extra fields that you need for your particular setup.

Settings Reference

API Login ID
Required. API Login ID. This key will authenticate requests to the payment gateway.

Transaction Key
Required. Transaction Key. This key will authenticate requests to the payment gateway.

Credit Card Transaction Type
Credit Card Transaction Type. It can be one of the following:
AUTH_CAPTURE, AUTH_ONLY, PRIOR_AUTH_CAPTURE, CAPTURE_ONLY, CREDIT, VOID, or you can use any other future development Authorize.Net types using the Expression button at the end of the dropdwon list.
If the value submitted does not match a supported value, the transaction is rejected. If this field is not submitted or the value is blank, the payment gateway will process the transaction as an AUTH_CAPTURE.

Go Live
Enable this option to switch to Live Mode. By default, unchecked, Test Mode, the transaction will be posted to the Authorize.Net's test server for developer accounts: https://test.authorize.net/gateway/transact.dll.
Make a few test transactions before going live!

Payment description
Tells what the transaction is about, for example a service name. This will appear in statements, receipts, etc.

Transaction's currency
For your convenience there is a dropdown list with a few preset currencies. You can at any time change it to other currency using the the Expression button at the end of the dropdwon list. The currencies that a merchant can accept through Authorize.Net are determined by their payment processor. Read the documentation or contact Authorize.Net to check which of the currencies are set for your account.

Amount - total to pay
The total Amount to pay in the selected currency. Can contain other context tokens, for example [TotalAmmount], and My Tokens.

Credit Card Number
Which of the fields in this form should be used as Credit Card Number

Credit Card CCV
Which of the fields in this form should be used as Credit Card CCV. This field is required if the merchant would like to use the Card Code Verification (CCV) security feature.

Expiration Month
Which of the fields in this form should be used as The customer’s credit card expiration Month

Expiration Year
Which of the fields in this form should be used as The customer’s credit card expiration Year

Note: x_exp_date Value: The customer’s credit card expiration date will be sent as a concatenation of the Expiration Month and Expiration Year fields.
Formats accepted by Authorize.Net: MMYY, MM/YY,MM-YY, MMYYYY, MM/YYYY, MM-YYYY
Set your fields accordingly.

First Name
Which of the fields in this form should be used as First Name

Last Name
Which of the fields in this form should be used as Last Name

Address
Which of the fields in this form should be used as Address

City

Which of the fields in this form should be used as City

State
Which of the fields in this form should be used as State

Country
Which of the fields in this form should be used as Country

Postal Code
Which of the fields in this form should be used as Postal Code/ ZIP Code.

Fields
Select which extra data to pass to Authorize.Net. Map Authorize.Net's Fields to Action Form Fields or Expressions. Additional data to pass to Authorize.Net. Make sure you read the AIM manual.
Some common used data could be perhaps x_invoice_num, x_email, x_cust_id, with values from some tokens.

Output Authorize.Net Response Code Token Name
Optionally provide a token name where to store the Authorize.Net Response Code generated by the transaction. For example, store Authorize.Net Response Code that is needed later in another action.

Output Authorize.Net Response Reason Code Token Name
Optionally provide a token name where to store the Authorize.Net Response Reason Code generated by the transaction. For example, store Authorize.Net Response Reason Code that is needed later in another action. (useful purpose: send it to an admin)

Output Authorize.Net Response Reason Text Token Name
Optionally provide a token name where to store the Authorize.Net Response Reason Text generated by the transaction. For example, store Authorize.Net Response Reason Text that is needed later in another action.

Output Authorize.Net Response Authorization Code Token Name
Optionally provide a token name where to store the Authorize.Net Response Authorization Code generated by the transaction. The authorization or approval code. For example, store Authorize.Net Response Authorization Code that is needed later in another action.

Output Authorize.Net Response Transaction ID Token Name
Optionally provide a token name where to store the Authorize.Net Response Transaction ID generated by the transaction. The payment gateway-assigned identification number for the transaction. For example, store Authorize.Net Response Transaction ID that is needed later in another action.

On Approved
Define a list of actions that should execute when this action's result is Approved.

On Declined
Define a list of actions that should execute when this action's result is Declined.

On Error

Define a list of actions that should execute when this action's result is Error.

On Held For Review
Define a list of actions that should execute when this action's result is Held For Review.

Make a payment with Electronic Check, eCheck

This type of payment allows one to submit electronic check transactions through an Authorize.Net Card Not Present Payment Gateway account, using Authorize.Net's Advanced Integration Method (AIM).

eCheck.Net® is Authorize.Net's exclusive electronic check processing solution. eCheck.Net enables Web merchants already processing credit card transactions through the Authorize.Net Payment Gateway to offer their customers an additional payment option.

The security of an AIM transaction is ensured through a 128-bit Secure Sockets Layer (SSL) connection between the merchant's Web server and the Authorize.Net Payment Gateway.

There are different electronic check transaction types supported by the payment gateway and they each have specific field requirements: Authorization and Capture, Authorization Only, Prior Authorization and Capture, etc.
Make sure you already read the Authorize.Net manuals prior to use different electronic check transaction types and specific fields.

There are currently six eCheck.Net transaction types supported by the Authorize.Net Payment Gateway.

- **Accounts Receivable Conversion (ARC)**

This transaction type is a one-time charge against a customer's checking account. ARC allows merchants to collect payments received in the mail or left in a drop-box, and convert them to an electronic payment.

- **Back Office Conversion (BOC)**

This transaction type is a one-time charge against a customer's checking account. BOC allows merchants to collect a check written at a point of sale (checkout counter, manned bill payment location, service call location) and convert it to an ACH debit during back office processing.

- **Cash Concentration or Disbursement (CCD)**

This transaction type is a one-time or recurring charge or refund against a business checking account. CCD transactions are fund transfers to or from a corporate entity.

- **Internet-Initiated Entry (WEB)**

This transaction type is a one-time or recurring charge against a consumer checking or savings account and for which payment authorization was obtained from the customer via the Internet.

- **Prearranged Payment and Deposit Entry (PPD)**

This transaction type is a one-time or recurring charge or refund against a consumer checking or savings account. PPD transactions may only be originated when payment and deposit terms between the merchant and the customer are prearranged.

- **Telephone-Initiated Entry (TEL)**

This transaction type is a one-time charge against a consumer checking or savings account that was originated by telephone. TEL transactions can only be originated when an existing relationship between the merchant and the customer exists; or if no relationship exists, the customer must initiate the telephone call to the merchant.

For your convenience, DNN Authorize.Net Add-on contains four templates to get you started, named: 'Pay With eCheck ARC Authorize.Net','Pay With eCheck PPD Authorize.Net','Pay With eCheck TEL Authorize.Net' and 'Pay With eCheck WEB Authorize.Net'.

They are forms that will do an Authorization and Capture transaction. This is the most common type of transaction and is the default payment gateway transaction type.

Requirements

- The merchant must have a merchant bank account that allows Internet transactions.
- The merchant must have an e-commerce (Card Not Present) Authorize.Net Payment Gateway account.
- The merchant must have a valid Secure Sockets Layer (SSL) certificate and their Web site must be capable of initiating both client- and server-side SSL connections.
- The merchant must be able to store payment gateway account data securely (for example, API Login ID, or Transaction Key).
- The merchant has completed the eCheck.Net application and underwriting process with Authorize.Net and is enabled to process eCheck.Net transactions.
- The merchant must have Action form module installed.

Getting started

Step 1 - Sign Up and Activate an Authorize.Net Account.

You will need both a merchant account and Authorize.Net Payment Gateway to accept credit cards.
You can sign up here: https://www.authorize.net/signupnow/

Step 2 - Get the API Login ID and Transaction Key from Authorize.Net.

These keys will authenticate requests to the Authorize.Net Payment Gateway.
Get the id and the key if don't already have them. You will find them in your ACCOUNT -> Settings -> Security Settings -> General Security Settings -> API Login ID and Transaction Key.

Your API Login ID and Transaction Key are unique pieces of information specifically associated with your payment gateway account. However, the API login ID and Transaction Key are NOT used for logging into the Merchant Interface. These two values are only required when setting up an Internet connection between your Web site and the payment gateway. They are used by the payment gateway to authenticate that you are authorized to submit Web site transactions.

Step 3 - Setup Action Form

Add an Action Form module to your page, go to Manage Form and select a Authorize.Net template to get started. Set your API Login ID and Transaction Key. Set the description and total ammount for your item. Add any extra fields that you need for your particular setup.

Settings Reference

API Login ID
Required. API Login ID. This key will authenticate requests to the payment gateway.

Transaction Key
Required. Transaction Key. This key will authenticate requests to the payment gateway.

eCheck Transaction Type
eCheck.Net Transaction Type. It can be one of the following:
AUTH_CAPTURE, AUTH_ONLY, PRIOR_AUTH_CAPTURE, CAPTURE_ONLY, CREDIT, VOID, or you can use any other future development Authorize.Net types using the Expression button at the end of the dropdwon list.
If the value submitted does not match a supported value, the transaction is rejected. If this field is not submitted or the value is blank, the payment gateway will process the transaction as an AUTH_CAPTURE.

Go Live
Enable this option to switch to Live Mode. By default, unchecked, Test Mode, the transaction will be posted to the Authorize.Net's test server for developer accounts: https://test.authorize.net/gateway/transact.dll.
Make a few test transactions before going live!

Payment description
Tells what the transaction is about, for example a service name. This will appear in statements, receipts, etc.

Transaction's currency
For your convenience there is a dropdown list with a few preset currencies. You can at any time change it to other currency using the the Expression button at the end of the dropdwon list. The currencies that a merchant can accept through Authorize.Net are determined by their payment processor. Read the documentation or contact Authorize.Net to check which of the currencies are set for your account.

Amount - total to pay
The total Amount to pay in the selected currency. Can contain other context tokens, for example [TotalAmmount], and My Tokens.

Bank Aba Code
Which of the fields in this form should be used as Bank Aba Code. The valid routing number of the customer's bank.

Bank Acct Num

Which of the fields in this form should be used as Bank Acct Num. The customer's valid bank account number

Bank Acct Type
Which of the fields in this form should be used as Bank Acct Type. The type of bank account.

Bank Name
Which of the fields in this form should be used as Bank Name. The name of the bank that holds the customer's account.

Bank Acct Name
Which of the fields in this form should be used as Bank Acct Name. The name associated with the bank account.

eCheck Type
ARC - Accounts Receivable Conversion
BOC - Back Office Conversion
CCD - Cash Concentration or Disbursement
PPD - Prearranged Payment and Deposit Entry
TEL - Telephone-Initiated Entry
WEB - Internet-Initiated Entry
The type of electronic check transaction.

Bank Check Number
Which of the fields in this form should be used as Bank Check Number. The check number on the customer's paper check. Required only when x_echeck_type=ARC or BOC.

First Name
Which of the fields in this form should be used as First Name

Last Name
Which of the fields in this form should be used as Last Name

Address
Which of the fields in this form should be used as Address

City
Which of the fields in this form should be used as City

State
Which of the fields in this form should be used as State

Country
Which of the fields in this form should be used as Country

Postal Code
Which of the fields in this form should be used as Postal Code/ ZIP Code.

Fields
Select which extra data to pass to Authorize.Net. Map Authorize.Net's Fields to Action Form Fields or Expressions. Additional data to pass to Authorize.Net. Make sure you read the AIM manual.
Some common used data could be perhaps x_invoice_num, x_email, x_cust_id, with values from some tokens.

Output Authorize.Net Response Code Token Name
Optionally provide a token name where to store the Authorize.Net Response Code generated by the transaction. For example, store Authorize.Net Response Code that is needed later in another action.

Output Authorize.Net Response Reason Code Token Name
Optionally provide a token name where to store the Authorize.Net Response Reason Code generated by the transaction. For example, store Authorize.Net Response Reason Code that is needed later in another action. (useful purpose: send it to an admin)

Output Authorize.Net Response Reason Text Token Name
Optionally provide a token name where to store the Authorize.Net Response Reason Text generated by the transaction. For example, store Authorize.Net Response Reason Text that is needed later in another action.

Output Authorize.Net Response Authorization Code Token Name
Optionally provide a token name where to store the Authorize.Net Response Authorization Code generated by the transaction. The authorization or approval code. For example, store Authorize.Net Response Authorization Code that is needed later in another action.

Output Authorize.Net Response Transaction ID Token Name
Optionally provide a token name where to store the Authorize.Net Response Transaction ID generated by the transaction. The payment gateway-assigned identification number for the transaction. For example, store Authorize.Net Response Transaction ID that is needed later in another action.

On Approved
Define a list of actions that should execute when this action's result is Approved.

On Declined
Define a list of actions that should execute when this action's result is Declined.

On Error
Define a list of actions that should execute when this action's result is Error.

On Held For Review
Define a list of actions that should execute when this action's result is Held For Review.

Simple Checkout

Simple Checkout helps you create "Buy Now" and "Donate" buttons for your Web site.

Simple Checkout is a perfect solution for organizations that rely on donations and specialty merchants that typically sell one item, either a product or a service, (in any quantity) per order.

Choose from classic Action Form Buttons, customize them to your own liking to include text or picture of your choosing.

How Simple Checkout Works

To generate the buttons, you simply enter applicable information such as an item description and price into the Authorize.Net Merchant Interface. HTML code is then generated automatically, which you can copy and paste into your Action Form, On Click Handler Action for your newly created button. When customers click your Simple Checkout button, they are taken to Authorize.Net's secure, hosted payment form to enter their payment information, along with any other required information.

Because sensitive card data is collected using Authorize.Net's secure servers, Simple Checkout can help simplify your compliance with the Payment Card Industry (PCI) Data Security Standard, https://www.pcisecuritystandards.org/security_standards/.

Configuring Your Buy Product Now, Buy Service Now or Donate Buttons

1. In Action Form, you can choose between adding a Button or an Image Button, configuring them to have some custom text or image.

2. All the other specific settings, are configured on Authorize.Net Merchant Interface. You only need to copy and paste the generated code to your On Click Handler Action, Button Code parameter.

On Authorize.Net Merchant Interface, you can customize several settings for each item:

- Item ID and Description

Each item can be assigned a unique item ID and description, which is displayed on your order page, as well as the payment form and receipt page.

- Suggested Donation Amounts

Nonprofit organizations can specify suggested donation amounts.

- Shipping Methods

If shipping is required, you can configure up to 10 shipping methods (e.g. Ground, Overnight, Two Day, etc.), with ranges and costs (e.g. 1-5 items = $4.95, 6-10 items = $6.95, etc.) for each method.

- Maximum Per Order

You can specify the maximum quantity of each item, per order, that a customer may purchase.

Getting started

Step 1 - Sign Up and Activate an Authorize.Net Account.

You will need both a merchant account and Authorize.Net Payment Gateway to accept credit cards. You can sign up here: https://www.authorize.net/signupnow/

Step 2 - Setup your items in Authorize.Net

Go to your account's Simple Checkout tab, and add the desired items. Currentlly, Authorize.Net provides three types of items: Donation, Product and Service.

Once the item is created, make sure you Copy all the Donate/Product/Service Button Code. That will be something along: <form ... </form>

Step 3 - Setup Action Form

Add an Action Form module to your page. Add a Button or Picture Button to your Form. Customize to your liking the text or the picture, then Add Action -> 'Add a Simple Checkout to a button' to On Click Handler of the newly created button. Paste the HTML code provided on Authorize.Net for the item in the Button Code field.

Settings Reference

- **Button Code**

In this field you will Paste the HTML code provided on Authorize.Net for the item as is. The addon will take care of the rest of the processing.

DNN Campaign Monitor Add-on

This extension provides the ability to subscribe to a Campaign Monitor list from Action Form. Typically, this is a Subscribe to Newsletter form (provided as a template in this extension), but you have the ability to use this integration as part of any form - for example as part of registration.

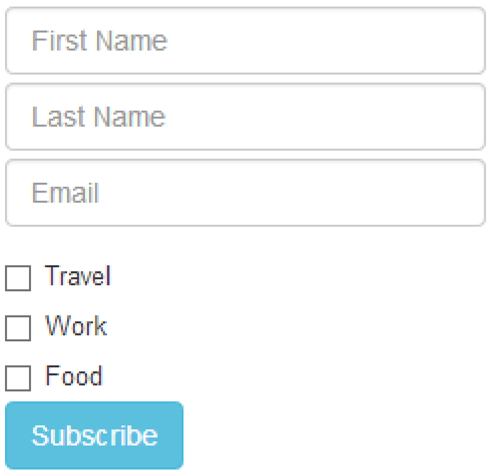
Getting Started

The following steps will guide you on installing and setting up the Campaign Monitor add-on.

If you don't already have it, you can download Action Form for DNN trial from our website. You will not be able to install the Campaign Monitor add-on if you don't have Action Form. The DNN installer will display a relevant error. Also, make sure you have at least version 03.02.59 of Action Form.

Once you have Action Form, you can proceed to install the Campaign Monitor extension that you've downloaded from DNN Store after your purchased it. Install it from Host > Extensions just as you would do with any other extension.

Now that you have installed all these, let's start adding the Campaign Monitor form. At the end you will have a form to subscribe like the one below (note that you can personalize it just the way you want it or you can just create one from scratch).

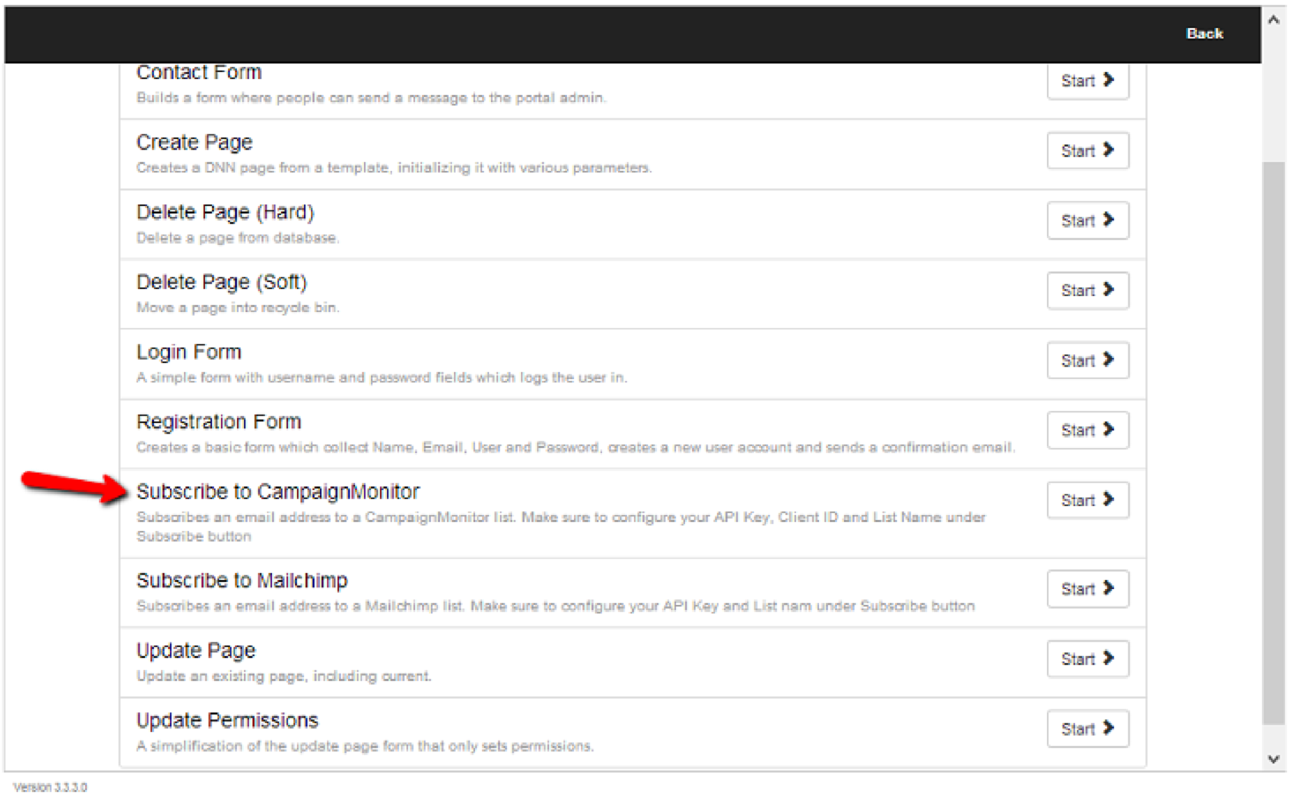


1. Select the page where you want to have your Campaign Monitor form.

2. Select in the top menu Modules > Add New Module and add the avt.ActionForm module to your page.

3. Now that you have installed Action Form on your page, select Manage Form.

4. In the start panel chose the Subscribe to CampaignMonitor template. Press the Start button and you will have the form installed on your page. Check the Configuration Options section below to see what needs to be configured before the form is actually functional.



Contact Form
Builds a form where people can send a message to the portal admin.
Create Page
Creates a DNN page from a template, initializing it with various parameters.
Delete Page (Hard)
Delete a page from database.
Delete Page (Soft)
Move a page into recycle bin.
Login Form
A simple form with username and password fields which logs the user in.
Registration Form
Creates a basic form which collect Name, Email, User and Password, creates a new user account and sends a confirmation email.
Subscribe to CampaignMonitor
Subscribes an email address to a CampaignMonitor list. Make sure to configure your API Key, Client ID and List Name under Subscribe button
Subscribe to Mailchimp
Subscribes an email address to a Mailchimp list. Make sure to configure your API Key and List nam under Subscribe button
Update Page
Update an existing page, including current.
Update Permissions
A simplification of the update page form that only sets permissions.

Configuration Options

Before you start building your list, you need to make some settings on the form.

In the Action Form settings panel, go to:

1. Interests (field) > Items and fill in your own interests categories.

2. Form Fields > Buttons > Subscribe (button) > On Click Handler (section) > Subscribe To Campaign Monitor (action)

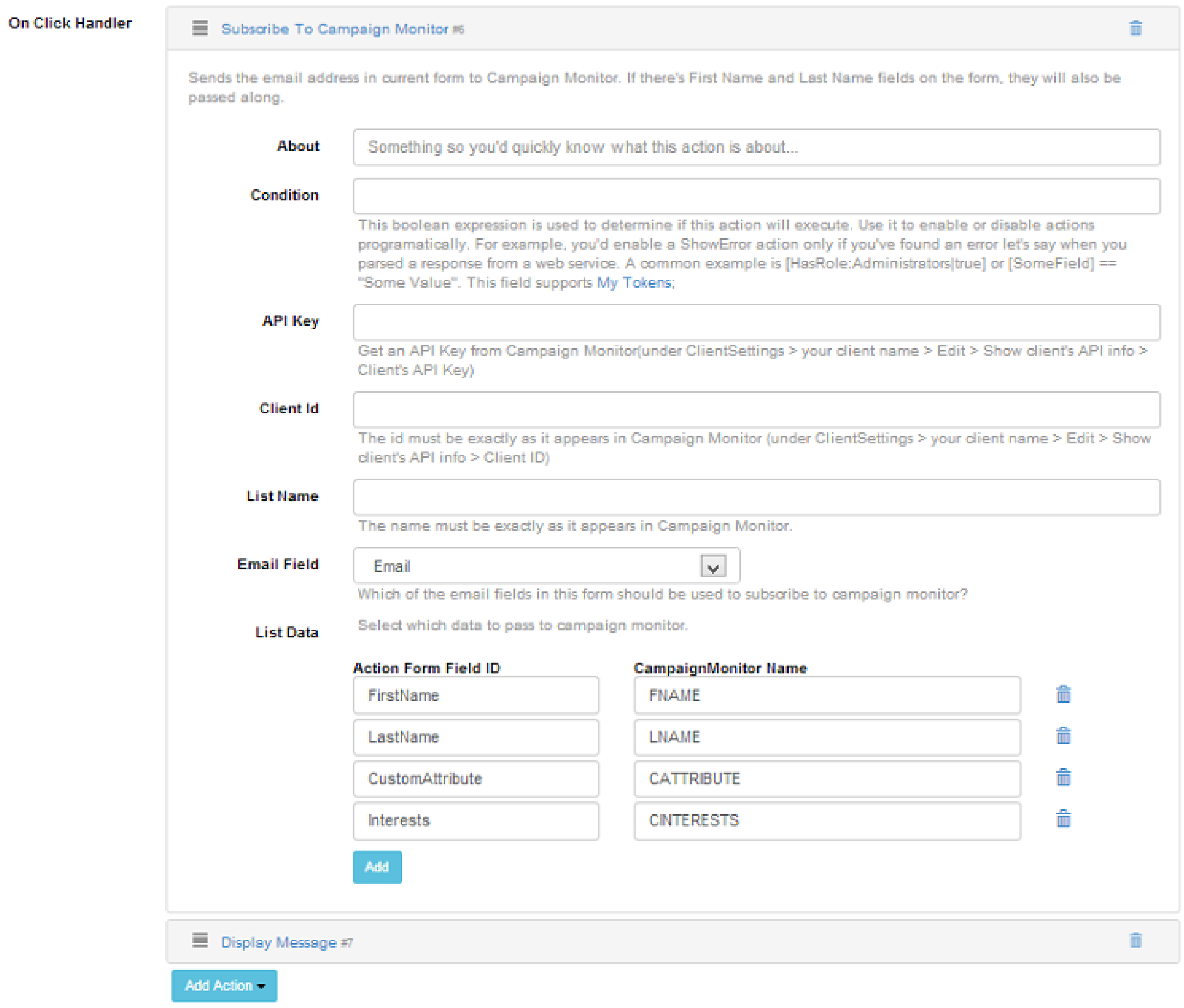
Fill in your Campaign Monitor details:

- API Key - you can find it under Client Settings > your client name > Edit > Show client's API info > Client's API Key.
- Client Id - must be exactly as it appears in Campaign Monitor under Client Settings > your client name > Edit > Show client's API info > Client ID.
- List Name - must be exactly as it appears in Campaign Monitor.

Also check these settings:

- For the Email Field select the appropriate email field to be used to subscribe to Campaign Monitor.
- If you decide to personalize the form with your custom fields, make sure to map them with the ones in Campaign Monitor. You can do this in the List Data section. These settings are used to pass correctly all data collected to Campaign Monitor.

After checking all settings, press the Save button and Back. That's all. Now you're ready to start building your email list.



On Click Handler

Subscribe To Campaign Monitor #6

Sends the email address in current form to Campaign Monitor. If there's First Name and Last Name fields on the form, they will also be passed along.

About: Something so you'd quickly know what this action is about...

Condition: This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;

API Key: Get an API Key from Campaign Monitor(under ClientSettings > your client name > Edit > Show client's API info > Client's API Key)

Client Id: The id must be exactly as it appears in Campaign Monitor (under ClientSettings > your client name > Edit > Show client's API info > Client ID)

List Name: The name must be exactly as it appears in Campaign Monitor.

Email Field: Email
Which of the email fields in this form should be used to subscribe to campaign monitor?

List Data: Select which data to pass to campaign monitor.

Action Form Field ID	CampaignMonitor Name
FirstName	FNAME
LastName	LNAME
CustomAttribute	CATTRIBUTE
Interests	CINTERESTS

This predefined form will display a notification message after he sign up. You can modify it to suits your needs on the On Click Handler > Display message.

You can also add and new action to send a notification email to subscriber. You can do this by clicking on the Add Action button > Email > Send Email. Check this section **Send Emails with Action Form** for more info about setting up the form to send emails.

DNN Clickatell Add-on

Getting started

In order to use the Clickatell gateway you need a Clickatell account and at least one registered connection (API sub-product instance) between your application and the Clickatell gateway. Each connection method is known as a sub-product. You can follow these steps to get started:

Step 1 - Sign up for an account with Clickatell

If you already have a Clickatell Central account, proceed to Step 2 for instructions on how to edit an API connection on your account.

If you do not already have a Clickatell Central account, you need to register for one. You can do so at https://www.clickatell.com/register/?productid=1 by signing up for the Developers' Central product. On sign up for you will be given a username, password and a client ID. Please note that the client ID is not the same thing as the API ID, which we will setup later. If you registered for the trial, you will have received 10 free credits with which to test the service. Messages sent with these credits contain a canned (pre-populated) message. You can test the API using these credits or purchase credits to start sending your own customised messages.

Step 2 - Setup Clickatell

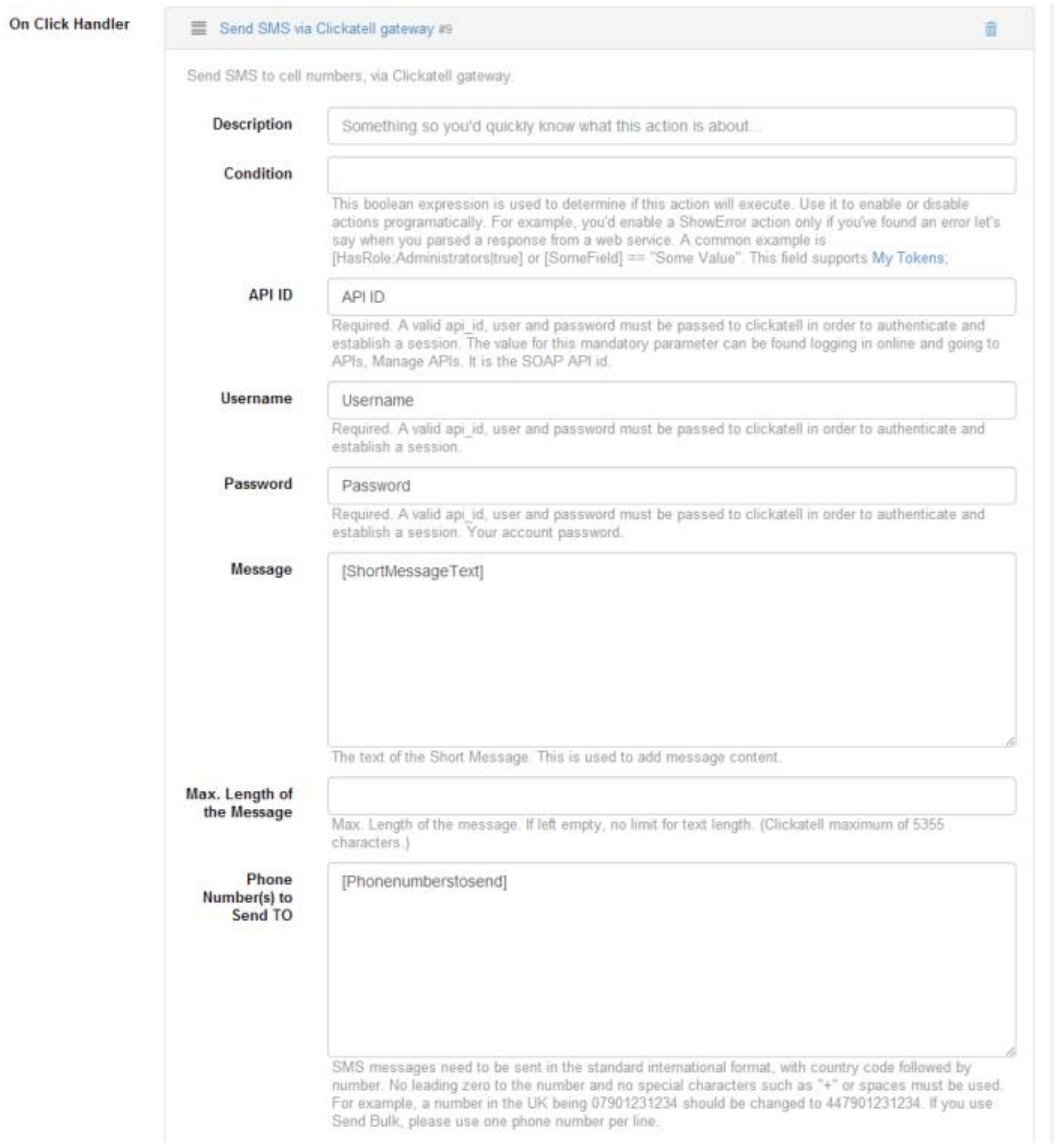
Action Form integrates with Clickatell using SOAP API. This needs to be setup in Clickatell Developers' Central. Login at https://www.clickatell.com/login/ and add the SOAP API to your account from APIs > Set up a new API > Add SOAP API. Complete all the required fields to configure your API. After successfully adding a connection, a confirmation message will be displayed with a unique API ID and information on how to get started. It displays the API connection parameters and authentication details. These are required when connecting to the Clickatell gateway from Action Form to send a message.
You will need this API's ID for Action Form. (not the SOAP API name). You can double check this API ID in the main menu: APIs > Manage APIs.

Step 3 - Setup Action Form

Once you have the Clickatell username, password and the SOAP API ID, you can proceed to setup Action Form. The Clickatell integration is provided as an add-on that can be purchased from DNN Store. So you will need to already have Action Form on the server before installing the Clickatell add-on. You can install both packages from Host > Extensions.

In this guide we will create a simple form to send an SMS message to a single phone number supplied in a text box. Start by adding a new Action Form module to the desired page. At the initial configuration step, for demo purposes, we will choose a Blank Form. It creates an empty form so you build everything from scratch. For our example we will need a text box to collect the phone number and a textarea for the message. Add these using the Add Field button in Action Form - Manage Screen.

Next, add a button which will push the message to the Clickatell API. The action can be found under Add Action > Communications > Send SMS via Clickatell gateway. Here, you will use your username, password from Step 1 and the SOAP API ID from step 2 from above. Finally, we will need to wire the phone number and the message fields to the respective settings under the Clickatell action. We do this by referencing the form fields using their token syntax. Assuming the fields are named Phone Number and Message, the tokens that go under the Clickatell settings are [PhoneNumber] and [Message]. If configured correctly, it should look like in the following screenshot:



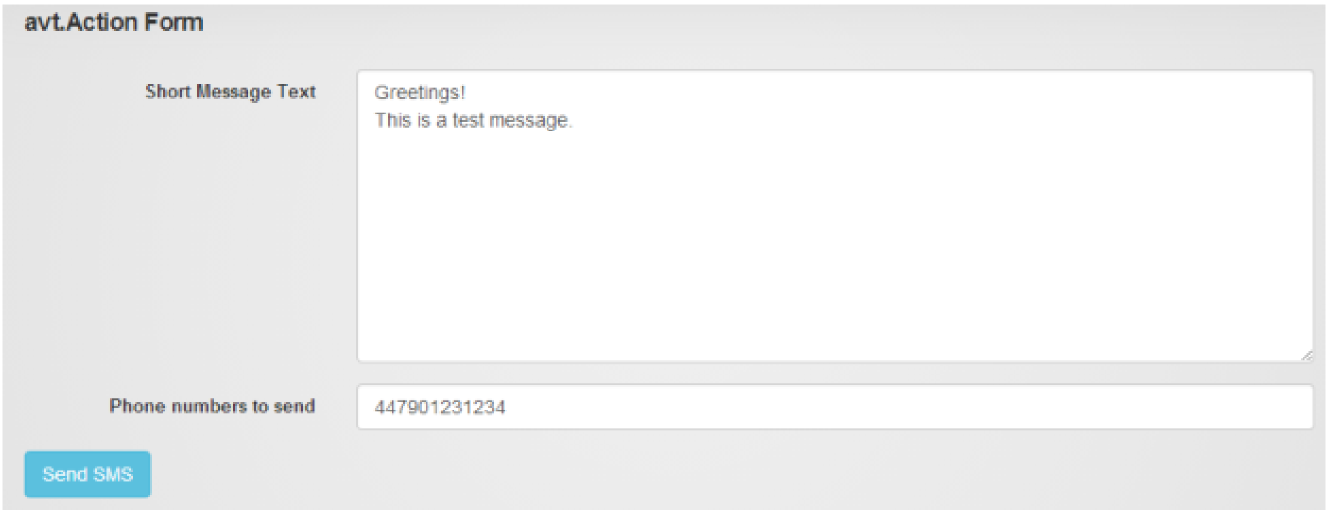
On Click Handler

Send SMS via Clickatell gateway #9

Send SMS to cell numbers, via Clickatell gateway.

Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;
API ID	API ID Required. A valid api_id, user and password must be passed to clickatell in order to authenticate and establish a session. The value for this mandatory parameter can be found logging in online and going to APIs, Manage APIs. It is the SOAP API id.
Username	Username Required. A valid api_id, user and password must be passed to clickatell in order to authenticate and establish a session.
Password	Password Required. A valid api_id, user and password must be passed to clickatell in order to authenticate and establish a session. Your account password.
Message	[ShortMessageText] The text of the Short Message. This is used to add message content.
Max. Length of the Message	Max. Length of the message. If left empty, no limit for text length. (Clickatell maximum of 5355 characters.)
Phone Number(s) to Send TO	[Phonenumberstosend] SMS messages need to be sent in the standard international format, with country code followed by number. No leading zero to the number and no special characters such as "+" or spaces must be used. For example, a number in the UK being 07901231234 should be changed to 447901231234. If you use Send Bulk, please use one phone number per line.

Save settings, go back to the front end. The form should look similar to


avt.Action Form

Short Message Text	Greetings! This is a test message.
Phone numbers to send	447901231234

Input your phone number, the message and click the button to have Action Form push the message to Clickatell and ultimately to your phone.

Settings Reference

This integration requires a valid Clickatell account and an SOAP API endpoint. Fill this in the first 3 fields labelled API ID, username and password. Note that the API ID that is created by default when you open the Clickatell account is a REST API ID, which will not work with current integration. Follow instructions at step 2 above for setting up a SOAP API ID.

The Clickatell integration allows the following configuration options:

Message
This can be a text message, or it can be a token that references the actual message. The token can be either a form field, in which case it will pull whatever value was submitted with form, or it can be a My Token. Note that an SMS is limited to 160 chars. If the message is larger, it will be split automatically by Clickatell, and may be merged back into a single message depending on the delivery network and the receiver phone.

Max. Length of the Message
This parameter is optional and allows administrators to limit the size of the messages that can be sent (for example, content submitted by users or content derived from user tokens). If set, it must be an integer. If left empty, there is no limit imposed by the administrator. Be aware that there is a limit though imposed by Clickatell, which is 5355 characters (SMS gateway technical limits). The extension will truncate the text to this Clickatell maximum, even if field is left empty.

Phone Number(s) to Send TO
SMS messages need to be sent in the standard international format, with country code followed by number. No leading zero to the number and no special characters such as "+" or spaces must be used. For example, a number in the UK being 07901231234 should be changed to 447901231234. You can send to more that one number by placing one phone number per line. The phone numbers can be

hardcoded or can come from tokens - either form tokens such as [PhoneNumber] or DNN/My Tokens such as [Profile:Cell].

DNN Constant Contact Add-on

This extension provides the ability to subscribe to a Constant Contact list from Action Form. Typically, this is a Subscribe to Newsletter form (provided as a template in this extension), but you have the ability to use this integration as part of any form - for example as part of registration.

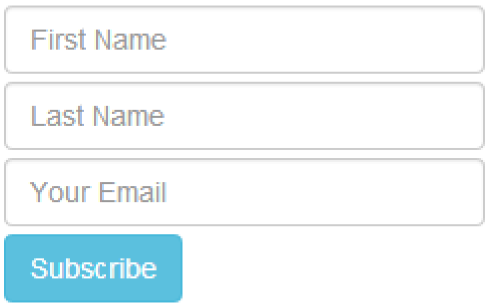
Getting Started

The following steps will guide you on installing and setting up the Constant Contact add-on.

If you don't already have it, you can download Action Form for DNN from our website. You will not be able to install the Constant Contact add-on if you don't have Action Form. The DNN installer will display a relevant error. Also, make sure you have at least version 03.03.20 of Action Form.

Once you have Action Form, you can proceed to install the Constant Contact extension that you've downloaded from DNN Store after your purchased it. Install it from Host > Extensions just as you would do with any other extension.

Now that you have installed all these, let's start adding the Constant Contact form. At the end you will have a form to subscribe like the one below (note that you can personalize it just the way you want it or you can just create one from scratch).

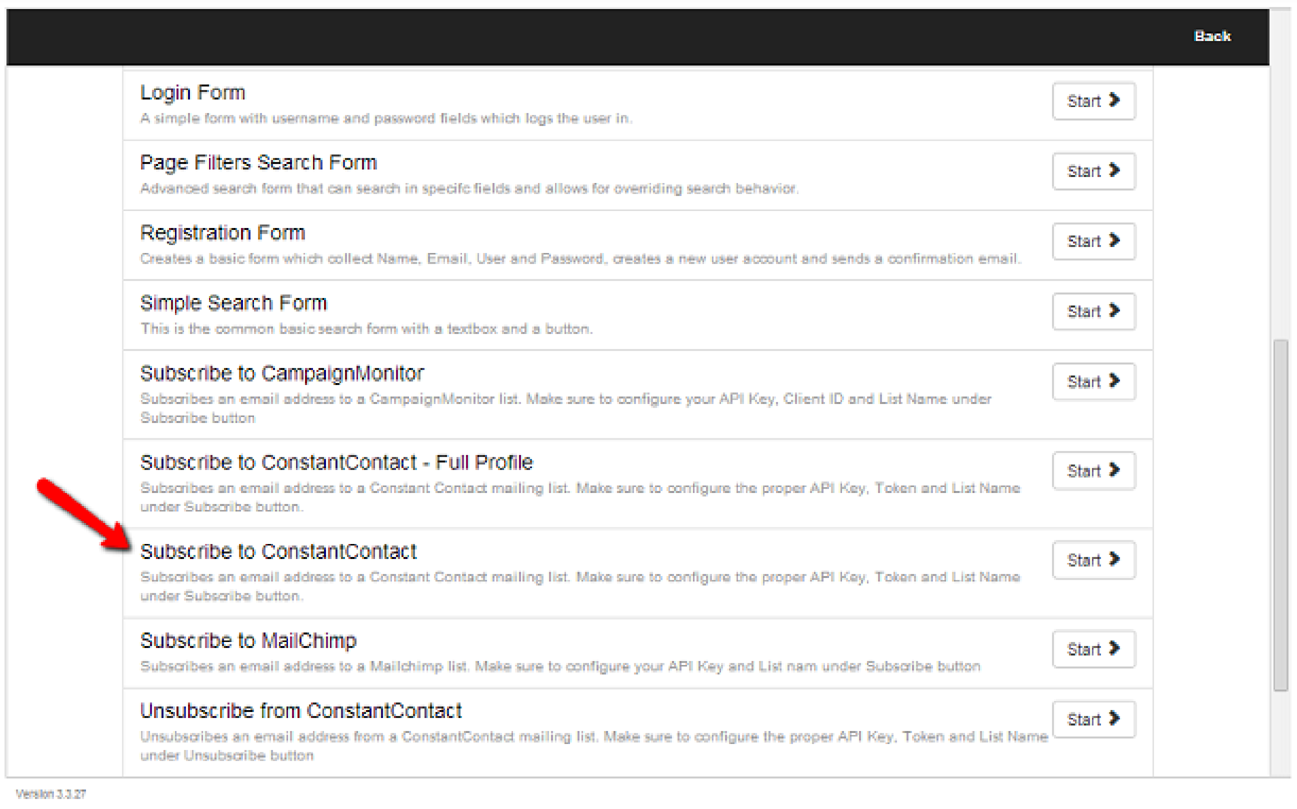


1. Select the page where you want to have your Constant Contact form.

2. Select in the top menu Modules > Add New Module and add the avt.ActionForm module to your page.

3. Now that you have installed Action Form on your page, select Manage Form.

4. In the start panel chose the Subscribe to Constant Contact template. Press the Start button and you will have the form installed on your page. Check the Configuration Options section below to see what needs to be configured before the form is actually functional.



Login Form
A simple form with username and password fields which logs the user in.

Page Filters Search Form
Advanced search form that can search in specifc fields and allows for overriding search behavior.

Registration Form
Creates a basic form which collect Name, Email, User and Password, creates a new user account and sends a confirmation email.

Simple Search Form
This is the common basic search form with a textbox and a button.

Subscribe to CampaignMonitor
Subscribes an email address to a CampaignMonitor list. Make sure to configure your API Key, Client ID and List Name under Subscribe button

Subscribe to ConstantContact - Full Profile
Subscribes an email address to a Constant Contact mailing list. Make sure to configure the proper API Key, Token and List Name under Subscribe button.

Subscribe to ConstantContact
Subscribes an email address to a Constant Contact mailing list. Make sure to configure the proper API Key, Token and List Name under Subscribe button.

Subscribe to MailChimp
Subscribes an email address to a Mailchimp list. Make sure to configure your API Key and List nam under Subscribe button

Unsubscribe from ConstantContact
Unsubscribes an email address from a ConstantContact mailing list. Make sure to configure the proper API Key, Token and List Name under Unsubscribe button

Version 3.3.27

Configuration Options

Before you start building your list, you need to make some settings on the form.

In the Action Form settings panel, go to:

1. Interests (field) > Items and fill in your own interests categories.

2. Form Fields > Buttons > Subscribe (button) > On Click Handler (section) > Subscribe To Constant Contact (action)

Fill in your Constant Contact details:

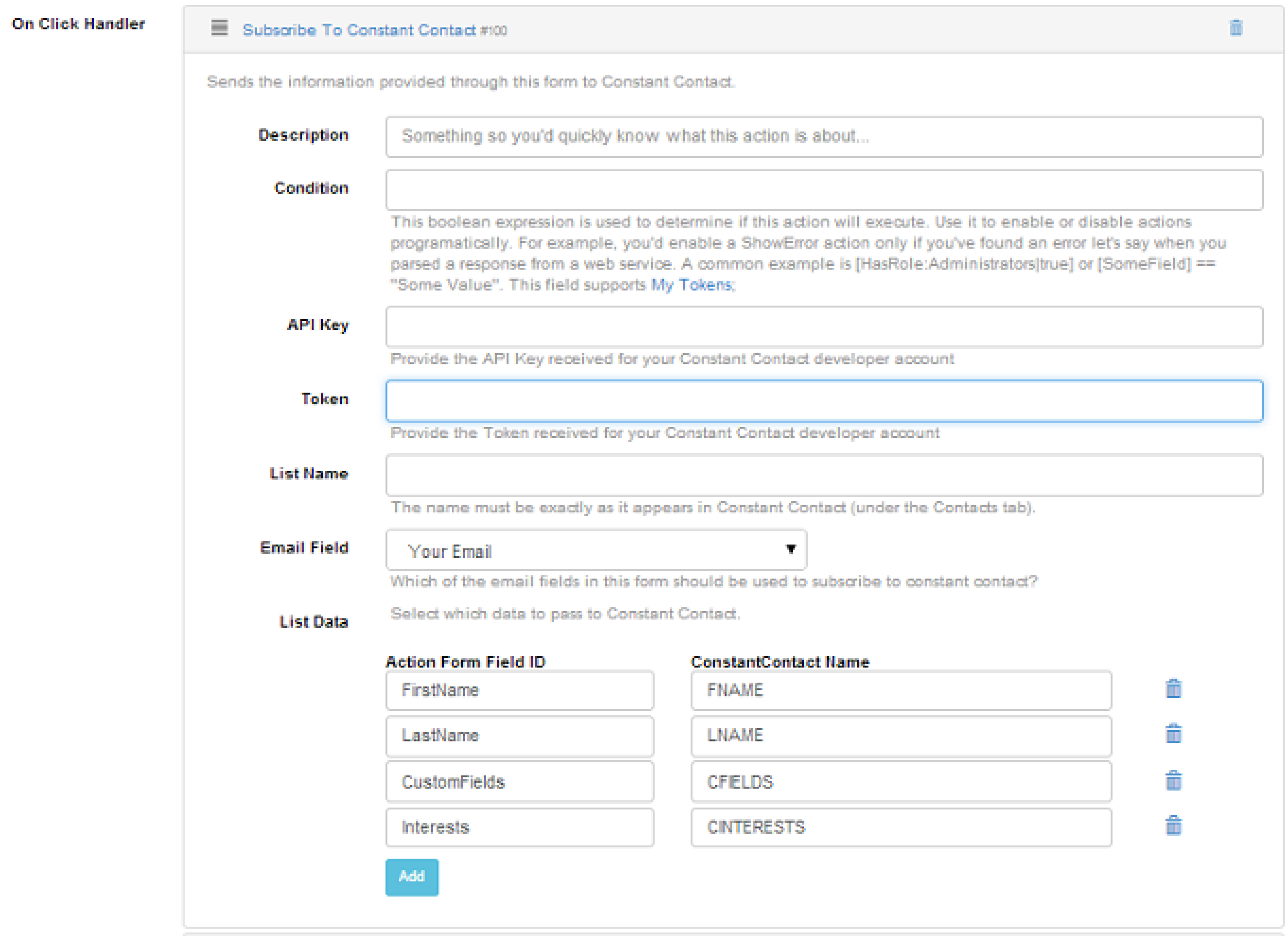
- API Key - received for your developer account. Details here http://developer.constantcontact.com/api-keys.html

- Token - received for your developer account.
- List Name - must be exactly as it appears in Constant Contact. Login here https://login.constantcontact.com/login/

Also check these settings:

- For the Email Field select the appropriate email field to be used to subscribe to Constant Contact.
- If you decide to personalize the form with your custom fields, make sure to map them with the ones in Constant Contact. You can do this in the List Data section. These settings are used to pass correctly all data collected to Constant Contact.

After checking all settings, press the Save button and Back. That's all. Now you're ready to start building your email list.



On Click Handler

Subscribe To Constant Contact #100

Sends the information provided through this form to Constant Contact.

Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens.
API Key	Provide the API Key received for your Constant Contact developer account
Token	Provide the Token received for your Constant Contact developer account
List Name	The name must be exactly as it appears in Constant Contact (under the Contacts tab).
Email Field	Your Email — Which of the email fields in this form should be used to subscribe to constant contact?
List Data	Select which data to pass to Constant Contact.

Action Form Field ID	ConstantContact Name
FirstName	FNAME
LastName	LNAME
CustomFields	CFIELDS
Interests	CINTERESTS

This predefined form will display a notification message after he sign up. You can modify it to suits your needs on the On Click Handler > Display message.

You can also add and new action to send a notification email to subscriber. You can do this by clicking on the Add Action button > Email > Send Email. Check this section Send Emails with Action Form for more info about setting up the form to send emails.

DNN PDF Generator Add-on

Getting started

Step 1 - Install Action Form if you don't have it installed yet.

Step 2 - Install the Add On*

*Note that the add on will install locally an executable that will need appropriate permissions to run on your server. If your system administrator does not allow this, you can use the External URL field (http://your-web-server/api/GeneratePDF) in conjunction with Generate Pdf Web API portal.

Step 3 - Configure the Action

How to generate Pdf documents directly from the DNN portal:

- Create a form with some text boxes and a button.
- Add an action of type Generate Pdf to the button.
- In the Generate Pdf action you can define an HTML template, using tokens, DNN or My Tokens, texts from your form text boxes, etc.
- Add a name for the PDF file to generate.
- Add the Generated PDF File Destination Path.
- There are a few options to select for your pdf: Orientation,Paper Size and Color.
- Optionally provide token names that can be used in next actions down the stack, for Absolute URL, Relative URL, Physical Path for the Generated PDF.
- Leave External URL empty for default local generation.

Settings Reference

- **HTML Code**

Can contain form tokens (for example [Email]) and My Tokens.

- **PDF Name**

The name of the PDF file to generate. If left empty, a GUID will be generated for name. Illegal characters will be cut out from the name.

- **Generated File Destination**

The destination directory on your portal.

- **Orientation Landscape**

Set orientation to Landscape. Default, unchecked, is Portrait.

- **Paper Size**

Set paper size to: A4, Letter, etc.

- **Grayscale**

If checked, PDF will be generated in Grayscale, otherwise will keep the html colors.

- **Store Absolute URL**

Optionally provide a token name where to Store Absolute URL. The token can be used in next actions down the stack.

- **Store Relative URL**

Optionally provide a token name where to Store Relative URL. The token can be used in next actions down the stack.

- **Store Physical Path**

Optionally provide a token name where to Store Physical Path. The token can be used in next actions down the stack.
Useful for example to send the generated Pdf file in an email attachment.

- **External URL**

When this is present, Action Form POSTs the HTML to this URL. Useful when pdf generation can't be executed on server due to permission restrictions.

There is a Generate Pdf Web API application available for download, to install it on your server of choice.

DNN Job Application Add-on

The following steps will guide you on installing and setting up the Job Application Form.

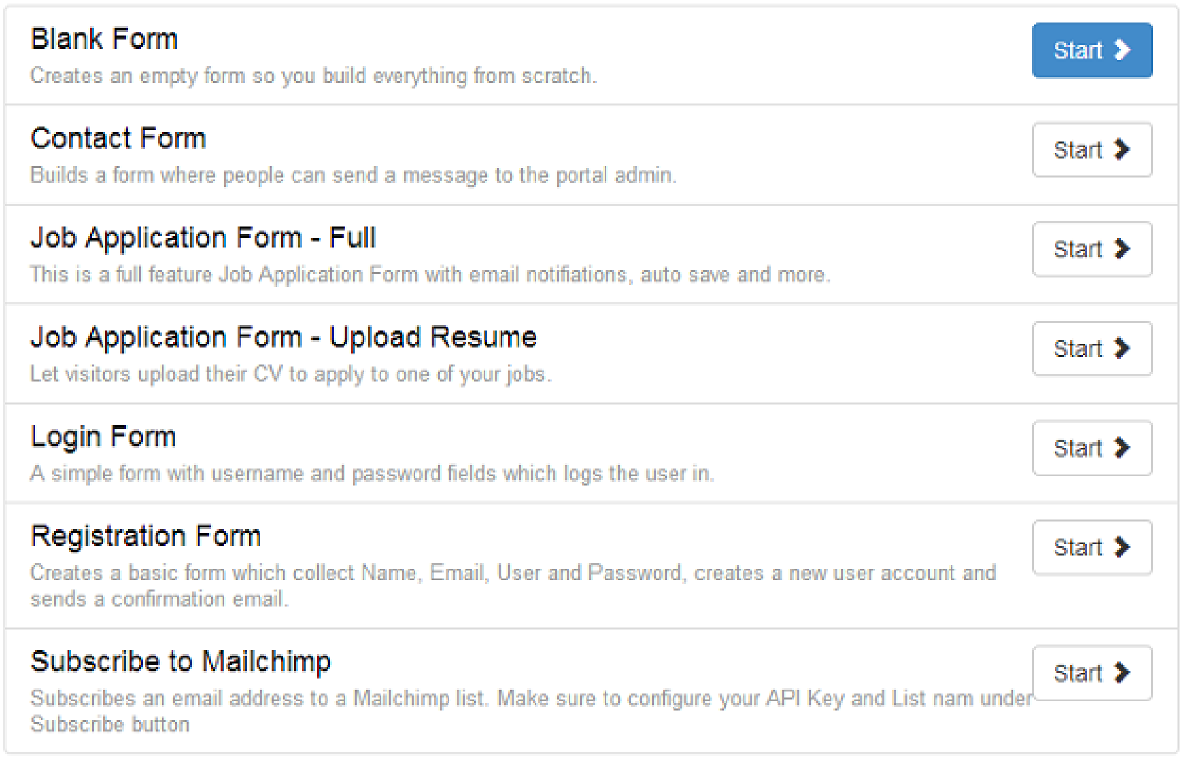
If you don't already have it, you can download Action Form for DNN trial from our website. You will not be able to install the Job Application add-on if you don't have Action Form. The DNN installer will display a relevant error. Also, make sure you have at least version 03.02.59 of Action Form.

Once you have Action Form, you can proceed to install the Job Application extension that you've downloaded from DNN Store after your purchased it. Install it from Host > Extensions just as you would do with any other extension.

Now that you have installed all these, let's start adding the Job Application form.

- Select the page where you want to have your Job Application Form
- Select in the top menu Modules > Add New Module and add the avt.ActionForm module to your page
- Now that you have installed Action Form on your page, select Manage Form

- In the start panel you can chose now which form would you like to add to your page: the short version (for uploading a Resume) or the long version to completed online. Press the Start button and you will have the form installed on your page.



Blank Form
Creates an empty form so you build everything from scratch.

Contact Form
Builds a form where people can send a message to the portal admin.

Job Application Form - Full
This is a full feature Job Application Form with email notifiations, auto save and more.

Job Application Form - Upload Resume
Let visitors upload their CV to apply to one of your jobs.

Login Form
A simple form with username and password fields which logs the user in.

Registration Form
Creates a basic form which collect Name, Email, User and Password, creates a new user account and sends a confirmation email.

Subscribe to Mailchimp
Subscribes an email address to a Mailchimp list. Make sure to configure your API Key and List nam under Subscribe button

- Before you start accept Resumes, you need to make some settings on the form. Press the Manage Form button and let's make these settings:
- Select the Upload Your Photo field and set the folder where you want the photo to be uploaded on the Upload to Folder section;
- Select the Job you apply for field and fill in the Items section with the jobs available;
- For the Annexes, select each field, File 1, File 2, File 3, File 4 and set the folder where you want the files to be uploaded on the Upload to Folder section;

- Check to see if the form is set up correctly to send emails (select button Ready? Apply for Dream Job > on On Click Handler section select the actions Send Email). If you encountered any issues, make sure to read more on how to set up Action Form to send emails.

That's all. Now you're ready to receive applicants for your available jobs.

Note that you can customize all the fields in the form, you can add new ones that suites your needs. But if you do this, also make the changes on the email templates that are delivered to you and your applicants. You can modify this templates on the Ready? Apply for Dream Job button > On Click Handler section > Send Email actions > Body section.

DNN MailChimp Add-on

This extension provides the ability to subscribe to a MailChimp list from Action Form. Typically, this is a Subscribe to Newsletter form (provided as a template in this extension), but you have the ability to use this integration as part of any form - for example as part of registration.

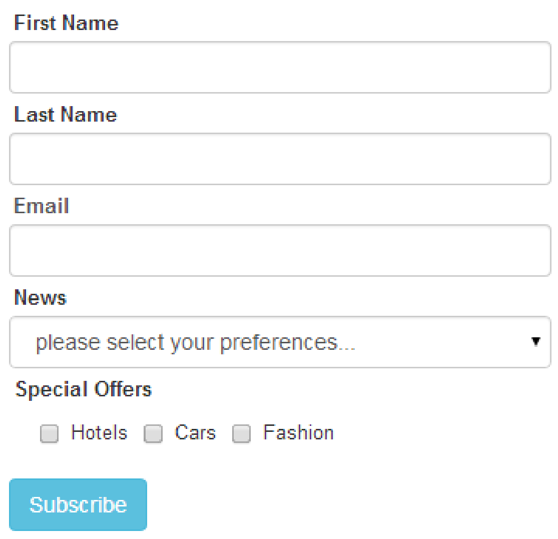
Getting Started

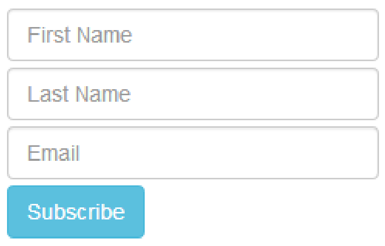
The following steps will guide you on installing and setting up the DNN MailChimp add-on.

If you don't already have it, you can download Action Form for DNN trial from our website. You will not be able to install the DNN MailChimp add-on if you don't have Action Form. The DNN installer will display a relevant error. Also, make sure you have at least version 03.02.59 of Action Form.

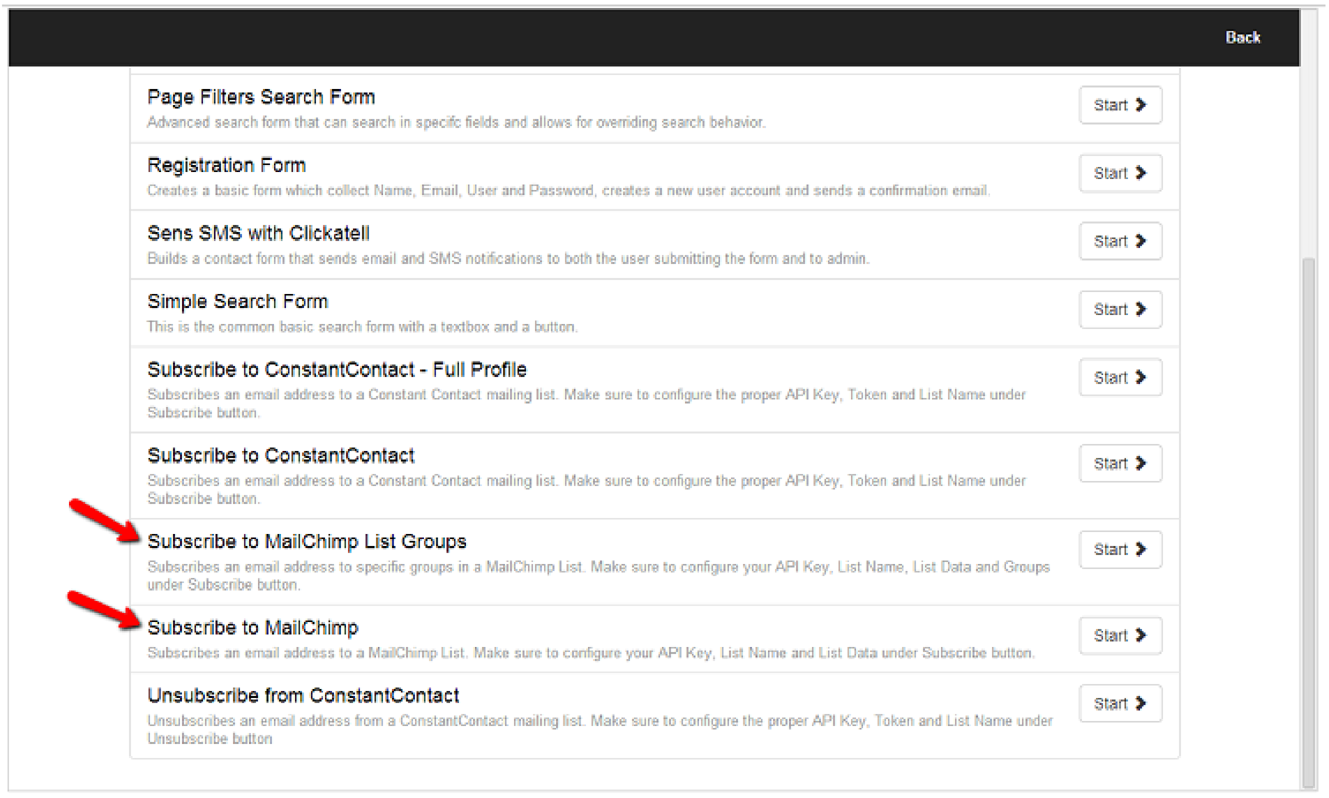
Once you have Action Form, you can proceed to install the DNN MailChimp extension that you've downloaded from DNN Store after your purchased it. Install it from Host > Extensions just as you would do with any other extension.

Now that you have installed all these, let's start adding one of the DNN MailChimp form. At the end you will have a form to subscribe like the ones below (note that you can personalize them just the way you want it or you can just create one from scratch).





- Select the page where you want to have your DNN MailChimp form.
- Select in the top menu Modules > Add New Module and add the avt.ActionForm module to your page.
- Now that you have installed Action Form on your page, select Manage Form.
- In the start panel chose the Subscribe to MailChimp or Subscribe to MailChimp List Groups. Press the Start button and you will have the form installed on your page. Check the Configuration Options section below to see what needs to be configured before the form is actually functional.



Page Filters Search Form Advanced search form that can search in specific fields and allows for overriding search behavior.	Start
Registration Form Creates a basic form which collect Name, Email, User and Password, creates a new user account and sends a confirmation email.	Start
Sens SMS with Clickatell Builds a contact form that sends email and SMS notifications to both the user submitting the form and to admin.	Start
Simple Search Form This is the common basic search form with a textbox and a button.	Start
Subscribe to ConstantContact - Full Profile Subscribes an email address to a Constant Contact mailing list. Make sure to configure the proper API Key, Token and List Name under Subscribe button.	Start
Subscribe to ConstantContact Subscribes an email address to a Constant Contact mailing list. Make sure to configure the proper API Key, Token and List Name under Subscribe button.	Start
Subscribe to MailChimp List Groups Subscribes an email address to specific groups in a MailChimp List. Make sure to configure your API Key, List Name, List Data and Groups under Subscribe button.	Start
Subscribe to MailChimp Subscribes an email address to a MailChimp List. Make sure to configure your API Key, List Name and List Data under Subscribe button.	Start
Unsubscribe from ConstantContact Unsubscribes an email address from a ConstantContact mailing list. Make sure to configure the proper API Key, Token and List Name under Unsubscribe button.	Start

Configuration Options

Before you start building your list, you need to make some settings on the form.

In the Action Form settings panel, go to Form Fields > Buttons > Subscribe (button) > On Click Handler (section) > Subscribe To MailChimp (action). Fill in your MailChimp details:

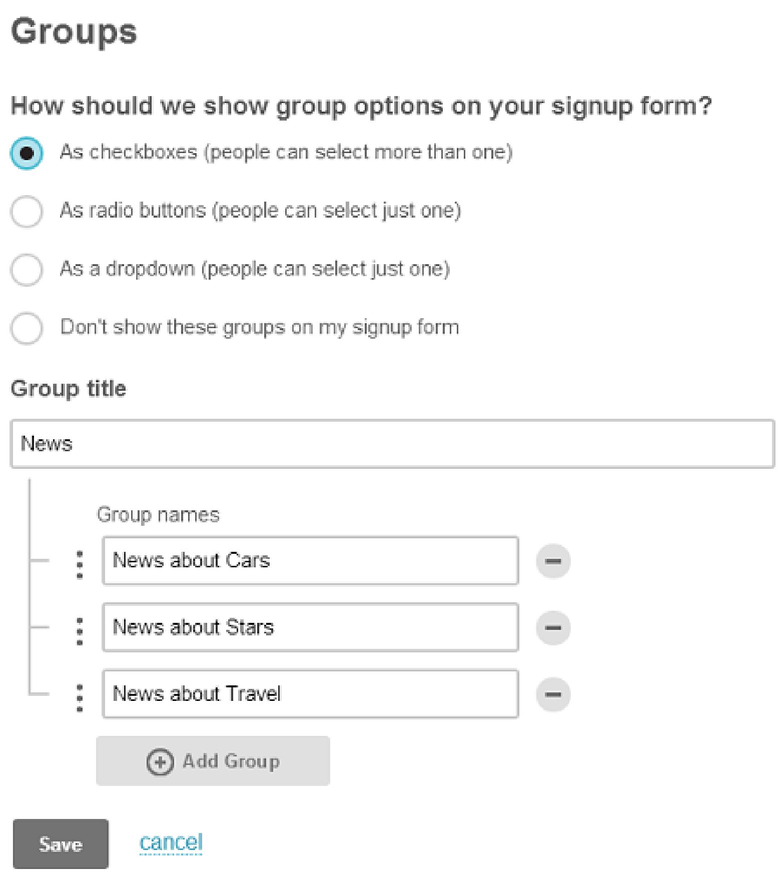
- API Key - you can find it under Account > Extra > API Keys.
- List Name - must be exactly as it appears in MailChimp.

Also check these settings:

- For the Email Field select the appropriate email field to be used to subscribe to MailChimp.
- If you decide to personalize the form with your custom fields, make sure to map them with the ones in MailChimp. You can do this in the List Data section. These settings are used to pass correctly all data collected to MailChimp.
- Groups (check screenshots below) - check these settings only if you want to subscribe new users to some specific groups in a MailChimp list. Note that:

- Group Title must be exactly as you have it in MailChimp.

- Group Names are the fields ID's created in Action Form. Enclose them in square brackets with no spaces. In Action Form, for the Groups listing you need to use the multiple choice field types. Also be sure that the Groups you add in Action Form matches exactly with the ones in MailChimp. For ex., as you see in the "Subscribe to Mailchimp List Groups" standard form, we used a multiple choice with dropdown field type for the "News"; in the "Items" section we've added "News about Cars", "News about Stars", "News about Travel" which matches exactly to the groups created in MailChimp.



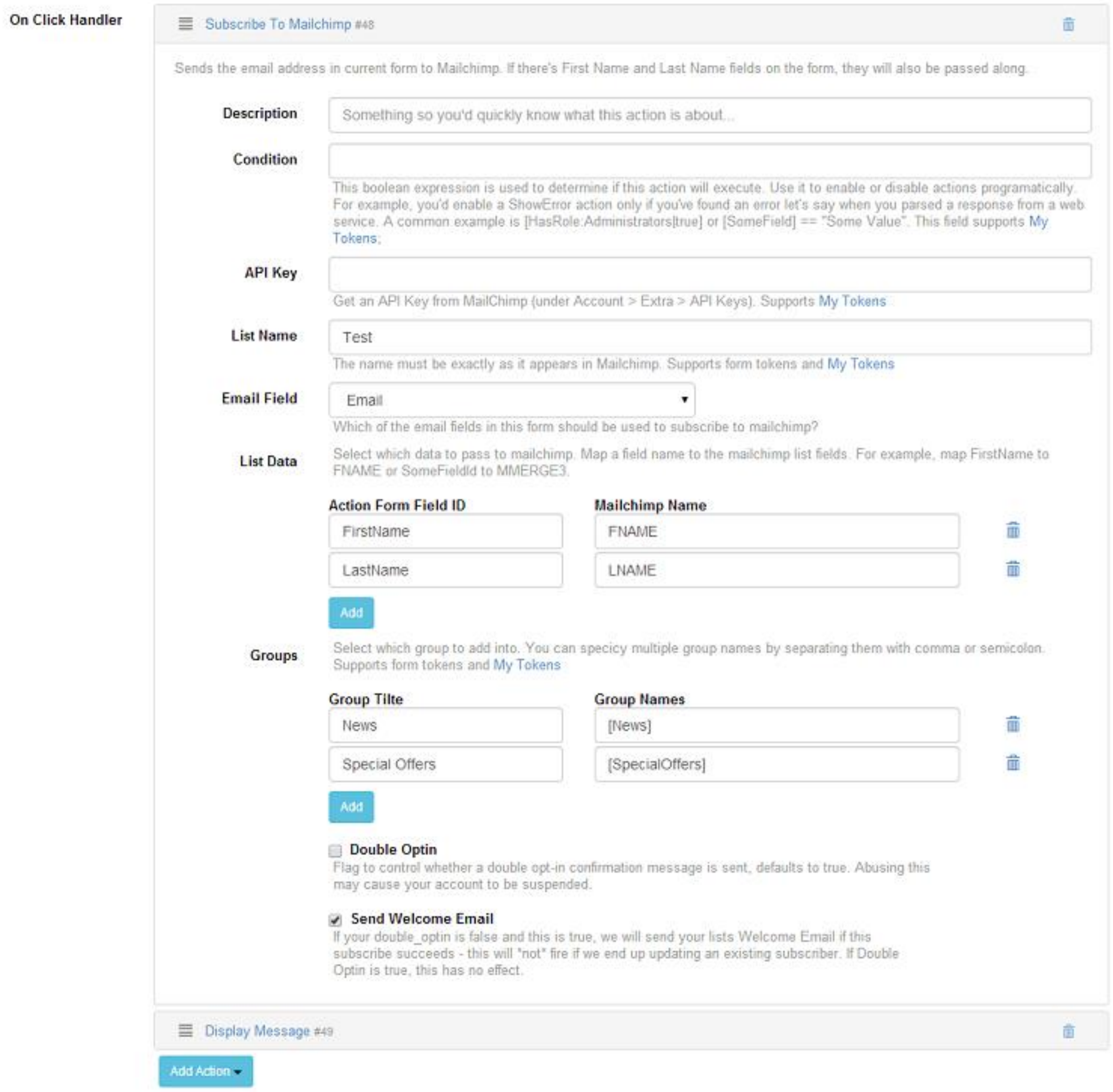
Groups

How should we show group options on your signup form?

- [✓ As checkboxes (people can select more than one)]
- As radio buttons (people can select just one)
- As a dropdown (people can select just one)
- Don't show these groups on my signup form

Group title: News

Group names:
- News about Cars
- News about Stars
- News about Travel



On Click Handler

Subscribe To Mailchimp #48

Sends the email address in current form to Mailchimp. If there's First Name and Last Name fields on the form, they will also be passed along.

Description: Something so you'd quickly know what this action is about...

Condition:
This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens.

API Key:
Get an API Key from MailChimp (under Account > Extra > API Keys). Supports My Tokens

List Name: Test
The name must be exactly as it appears in Mailchimp. Supports form tokens and My Tokens

Email Field: Email
Which of the email fields in this form should be used to subscribe to mailchimp?

List Data
Select which data to pass to mailchimp. Map a field name to the mailchimp list fields. For example, map FirstName to FNAME or SomeFieldId to MMERGE3.

Action Form Field ID	Mailchimp Name
FirstName	FNAME
LastName	LNAME

Groups
Select which group to add into. You can speciy multiple group names by separating them with comma or semicolon. Supports form tokens and My Tokens

Group Tilte	Group Names
News	[News]
Special Offers	[SpecialOffers]

☐ **Double Optin**
Flag to control whether a double opt-in confirmation message is sent, defaults to true. Abusing this may cause your account to be suspended.

✓ **Send Welcome Email**
If your double_optin is false and this is true, we will send your lists Welcome Email if this subscribe succeeds - this will *not* fire if we end up updating an existing subscriber. If Double Optin is true, this has no effect.

Display Message #49

After checking all settings, press the Save button and Back. That's all. Now you're ready to start building your email list.

This predefined form will display a notification message after he sign up. You can modify it to suits your needs on the On Click Handler > Display message.

You can also add and new action to send a notification email to subscriber. You can do this by clicking on the Add Action button > Email > Send Email. Check this section Send Emails with Action Form for more info about setting up the form to send emails.

My Action Form Uniqueizer Add-on

My Action Forms Uniqueizer is an add-on for DNN Sharp's Action Form DNN module and an accompanying DNN module to administer the generated unique data.

Installation

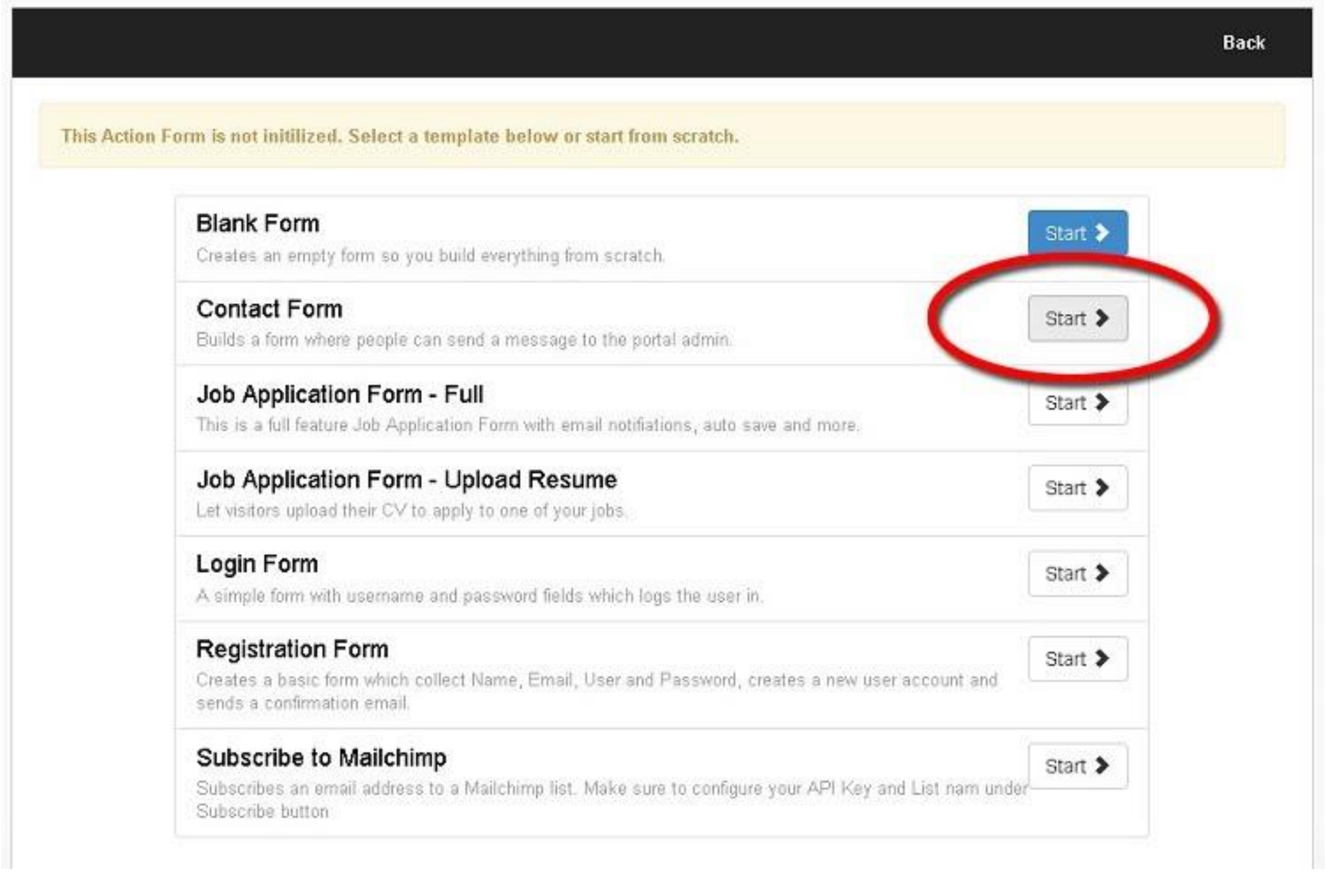
Log into your DNN installation as Administrator or SuperUser and install MyActionFormsUniqueizer as you would any other DNN module. In your list of installed modules, you will get an entry called MyActionFormsUniqueizer and inside your ActionForm you will get another Action.

General

If you want to do a sweepstake on your DNN website using ActionForm you will want to keep participants from submitting multiple times. This is where MyActionFormsUniqueizer comes in.

A simple Sweepstake

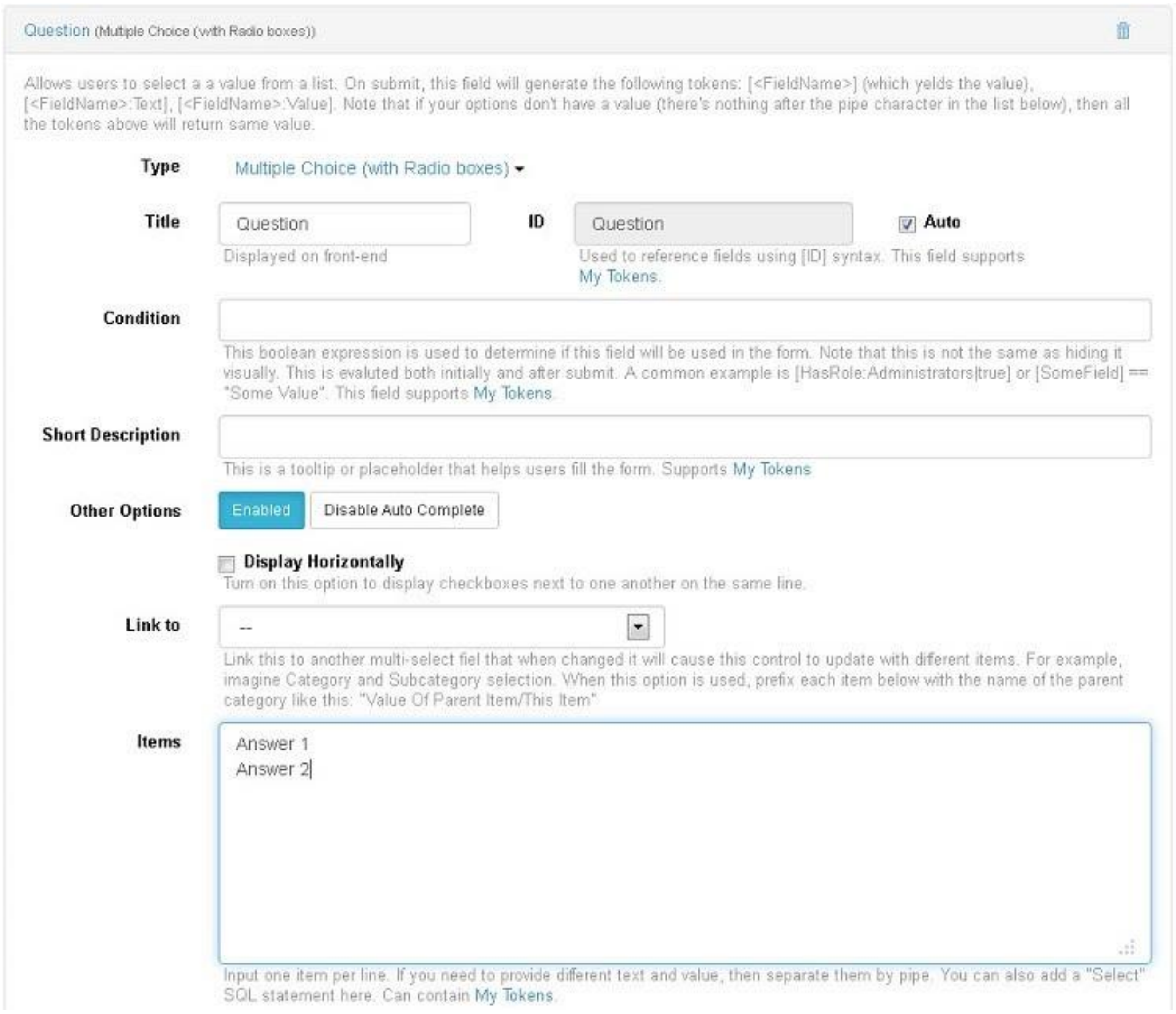
Let's try it out. Add an ActionForm module to a page on your site.
Give your module a meaningful title, here "Sweepstake". Select "Manage Form" from the module popup. And select for instance the "Contact Form" template, or any other predefined or custom template, that seems appropriate.



This Action Form is not initilized. Select a template below or start from scratch.

Blank Form Creates an empty form so you build everything from scratch.	Start
Contact Form Builds a form where people can send a message to the portal admin.	Start
Job Application Form - Full This is a full feature Job Application Form with email notifiations, auto save and more.	Start
Job Application Form - Upload Resume Let visitors upload their CV to apply to one of your jobs.	Start
Login Form A simple form with username and password fields which logs the user in.	Start
Registration Form Creates a basic form which collect Name, Email, User and Password, creates a new user account and sends a confirmation email.	Start
Subscribe to Mailchimp Subscribes an email address to a Mailchimp list. Make sure to configure your API Key and List nam under Subscribe button	Start

Now you add a field for the question:


Question (Multiple Choice (with Radio boxes))

Allows users to select a a value from a list. On submit, this field will generate the following tokens: [<FieldName>] (which yelds the value), [<FieldName>:Text], [<FieldName>:Value]. Note that if your options don't have a value (there's nothing after the pipe character in the list below), then all the tokens above will return same value.

Type: Multiple Choice (with Radio boxes)

Title: Question
Displayed on front-end

ID: Question [✓ Auto]
Used to reference fields using [ID] syntax. This field supports My Tokens.

Condition:
This boolean expression is used to determine if this field will be used in the form. Note that this is not the same as hiding it visually. This is evaluted both initially and after submit. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens.

Short Description:
This is a tooltip or placeholder that helps users fill the form. Supports My Tokens

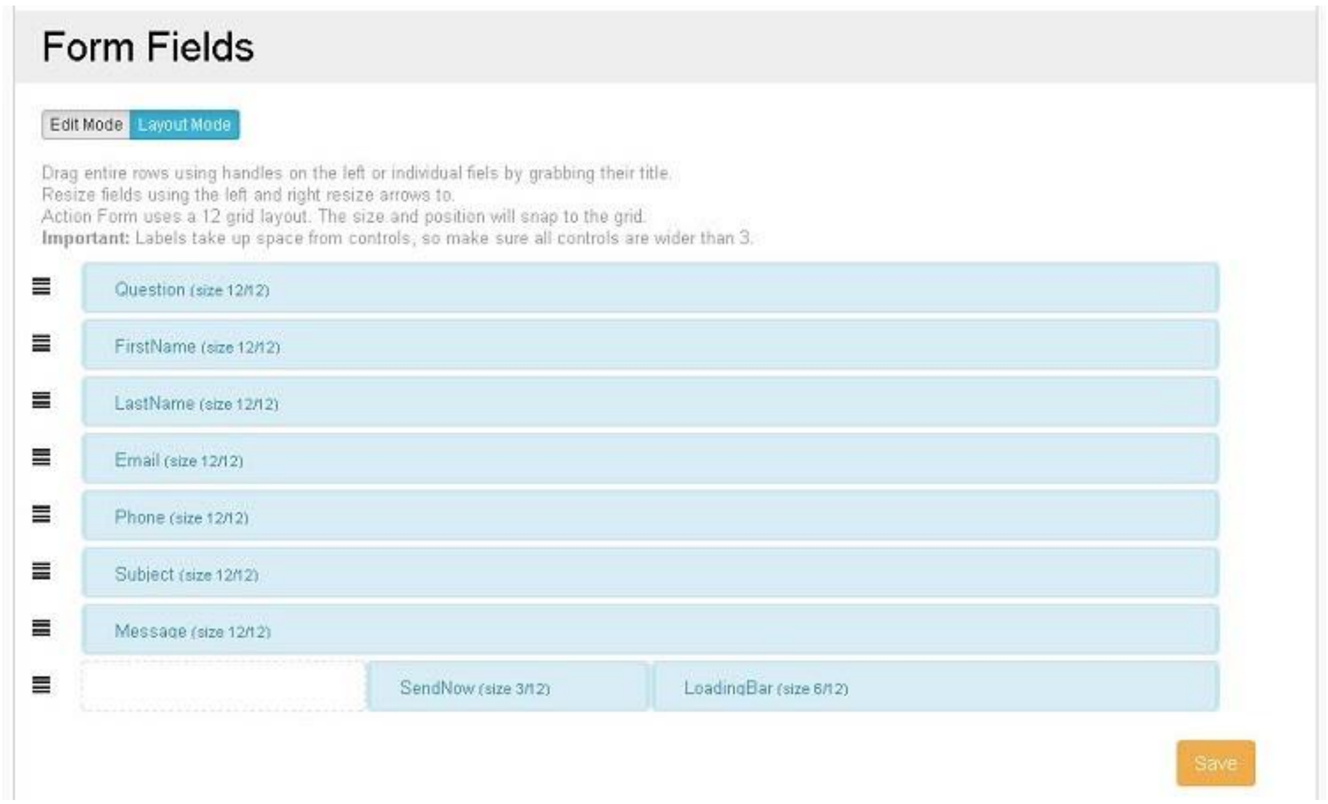
Other Options: [✓ Enabled] Disable Auto Complete

Display Horizontally
Turn on this option to display checkboxes next to one another on the same line.

Link to: --
Link this to another multi-select fiel that when changed it will cause this control to update with diferent items. For example, imagine Category and Subcategory selection. When this option is used, prefix each item below with the name of the parent category like this: "Value Of Parent Item/This Item"

Items:
Answer 1
Answer 2
Input one item per line. If you need to provide different text and value, then separate them by pipe. You can also add a "Select" SQL statement here. Can contain My Tokens.

I used a “Multiple Choice (with Radio boxes)” type of field. And write a question and some answers. Make the field “Reqiured” and move it to the top of all the fields.
Select “Layout Mode” and drag and drop the field. Now save your changes. Now add the validator that prevents users from submitting multiple times.


Form Fields

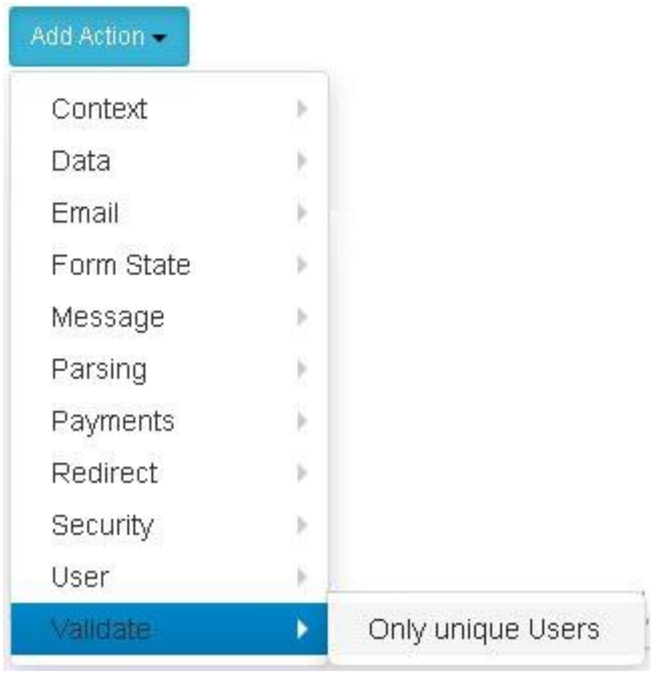
Edit Mode | Layout Mode

Drag entire rows using handles on the left or individual fiels by grabbing their title.
Resize fields using the left and right resize arrows to.
Action Form uses a 12 grid layout. The size and position will snap to the grid.
Important: Labels take up space from controls, so make sure all controls are wider than 3.

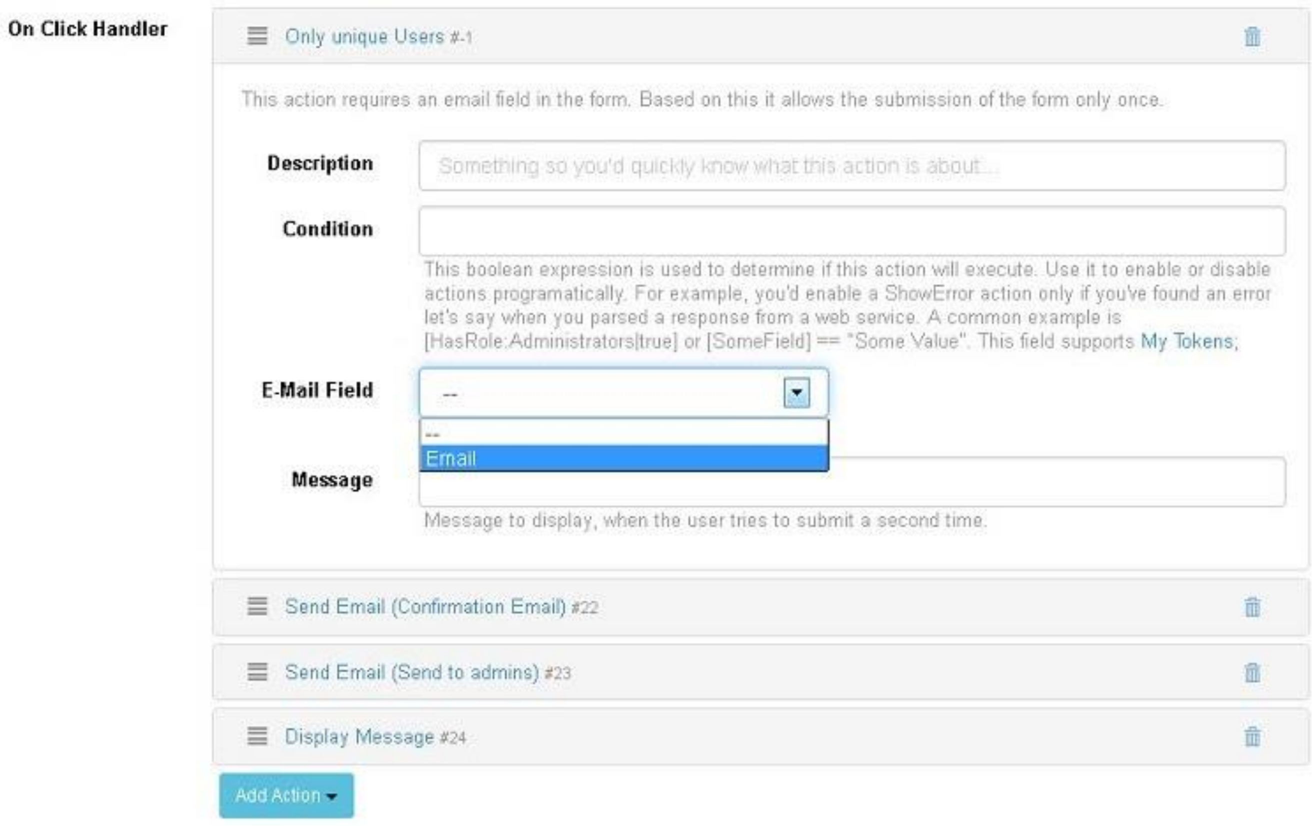
Question (size 12/12)
FirstName (size 12/12)
LastName (size 12/12)
Email (size 12/12)
Phone (size 12/12)
Subject (size 12/12)
Message (size 12/12)
SendNow (size 3/12) | LoadingBar (size 6/12)

Save

Back in “Edit Mode” expand the “SendNow” button and add a new action.



Under “Validate” add “Only unique Users” and move it to the top, so that it is the first of the actions that are triggered by “SendNow”, because you will want to prevent other actions from running, when the user submits a second time.



On Click Handler

Only unique Users #-1

This action requires an email field in the form. Based on this it allows the submission of the form only once.

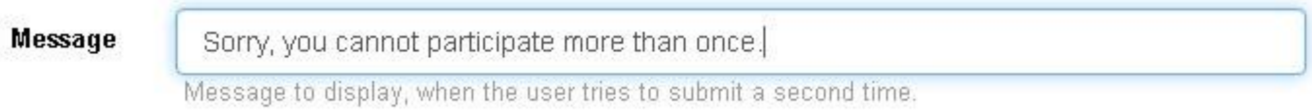
Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;
E-Mail Field	-- (options: --, Email)
Message	Message to display, when the user tries to submit a second time.

Send Email (Confirmation Email) #22

Send Email (Send to admins) #23

Display Message #24

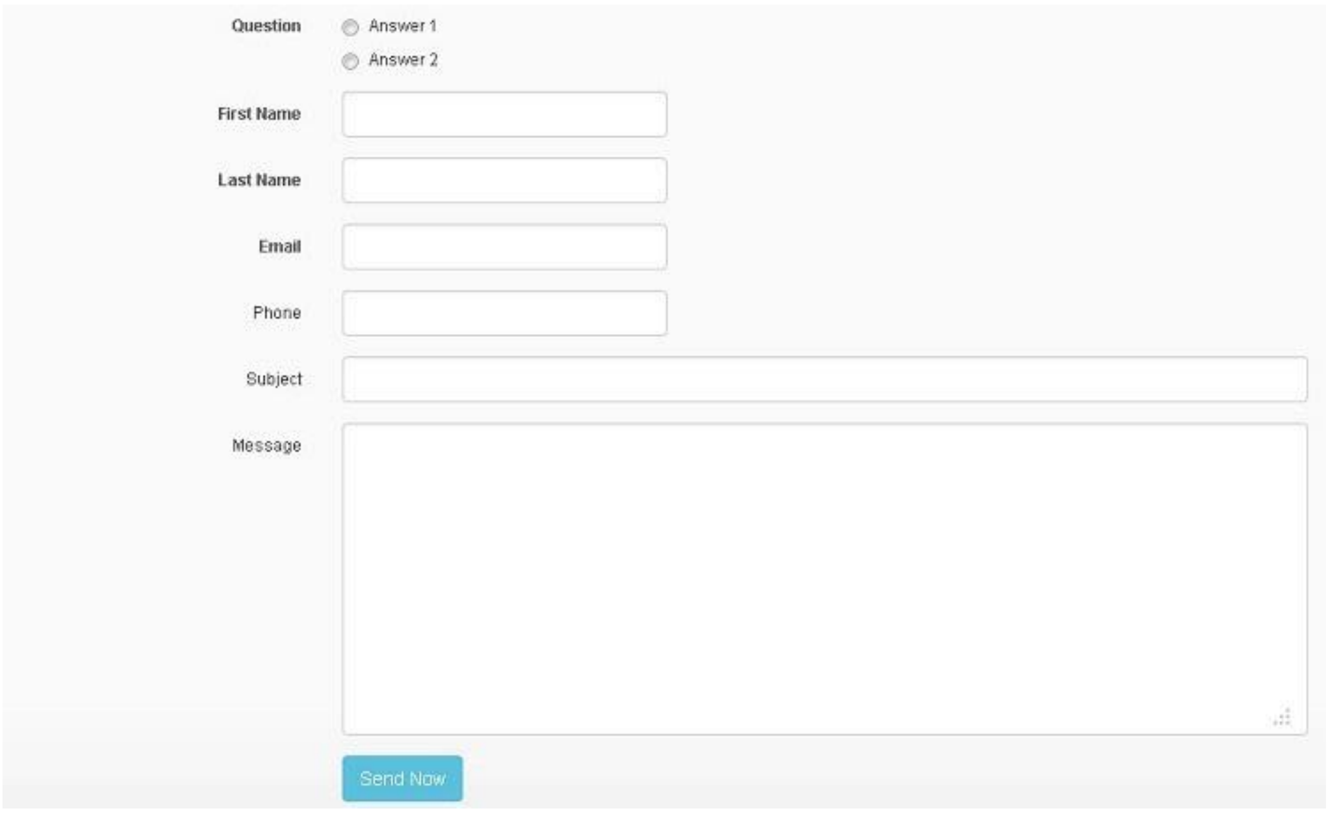
The “Only unique Users” action needs two fields to be filled out. First you have to select witch field contains the e-mail address, that the action uses to identify the user. And second you write the message that is displayed, when the user is submitting a second time.



Message: Sorry, you cannot participate more than once.

Message to display, when the user tries to submit a second time.

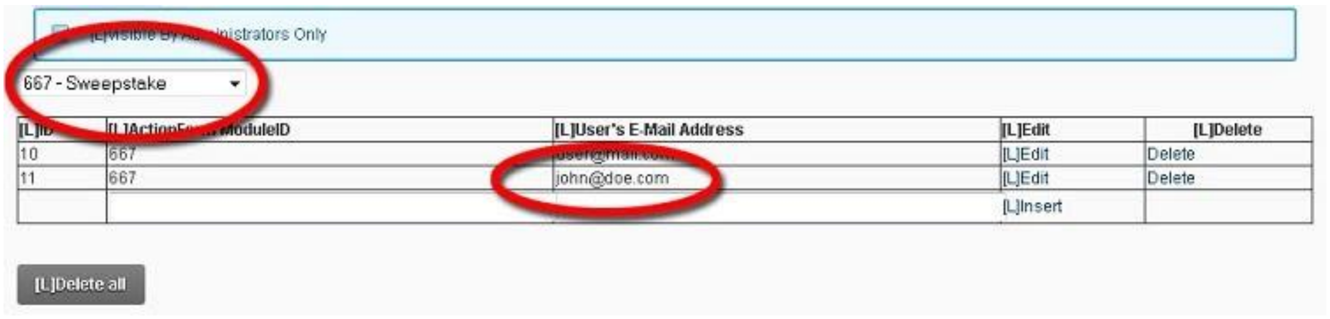
The final Action Form looks like this:



The Admin Module

The admin module is named "MyActionFormsUniqueizer" and you can add it to the page like any other module.

The module should not be visible to the sites users but only for administrators. So you either put it on a hidden page or set the permissions of the module to be only displayed for administrators. The module can be used to administrate all instances of ActionForm with MyActionFormsUniqueizer add-in.



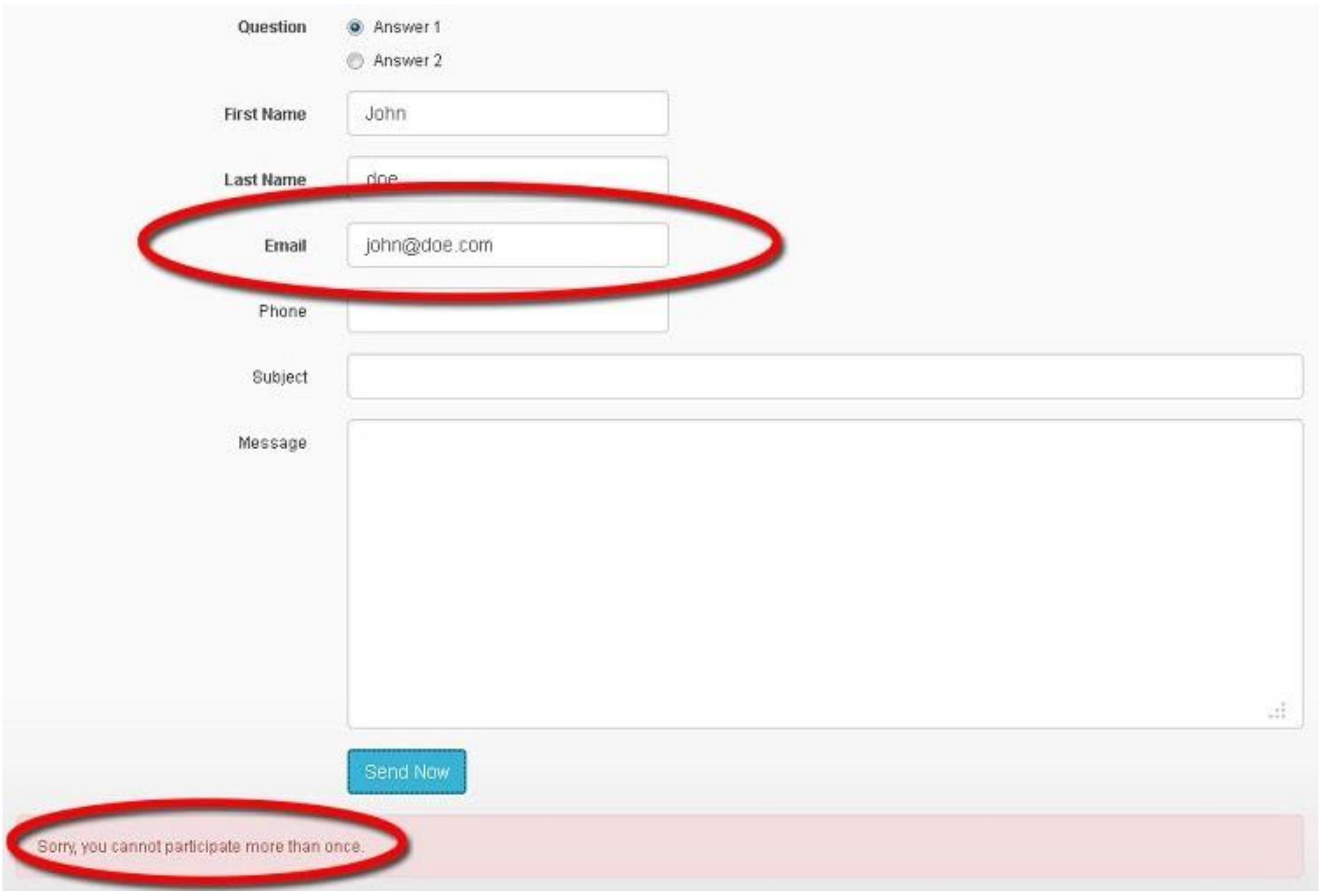
[L]Visible By Administrators Only

667 - Sweepstake

[L]ID	[L]ActionForm ModuleID	[L]User's E-Mail Address	[L]Edit	[L]Delete
10	667	user@mail.com	[L]Edit	Delete
11	667	john@doe.com	[L]Edit	Delete
			[L]Insert	

[L]Delete all

In the upper left corner of the module (first red ring) is a dropdown, where the module-id and the title of a given ActionForm module is displayed, in this case "Sweepstake". The second red ring denotes a particular user that has submitted his answer to the sweepstake once. If he tries it a second time, the form looks like this:



Question	(● Answer 1) ○ Answer 2
First Name	John
Last Name	doe
Email	john@doe.com
Phone	
Subject	
Message	

Sorry, you cannot participate more than once.

The users e-mail address is found in the table and therefore the message that you have entered gets displayed and any subsequent action is not performed.

DNN Salesforce Add-on

Getting started

This Add-on provides a 2 way integration between our modules and Sales Force. It can push data to Salesforce, such as leads, contacts or any other entity, and it can also query Salesforce using SOQL - which is an SQL-like query language provided by Salesforce. Once data is retrieved, it can be stored in tokens and used in other actions down the stack.

Salesforce's customer relationship management (CRM) service is broken down into several broad categories: Sales Cloud, Service Cloud, Data Cloud (including Jigsaw), Collaboration Cloud (including Chatter) and Custom Cloud (including Force.com). Salesforce users can configure their CRM application at the "platform" level. In the Salesforce system, there are tabs such as "Contacts", "Reports", "Accounts", etc. Each tab contains associated information. For example, "Contacts" has standard fields like First Name, Last Name, and Email.

In addition to the web interface, Salesforce.com offers a SOAP Web service API that enables integration with other systems. Here is were you, the DNN admin come into play! Now users can add entities and run queries, not only at the "platform" level, but from DNN platform too! An administrator can add this extension in Action Form, set up a few things, and his DNN users are good to go! This is how Action From communicates with Salesforce.

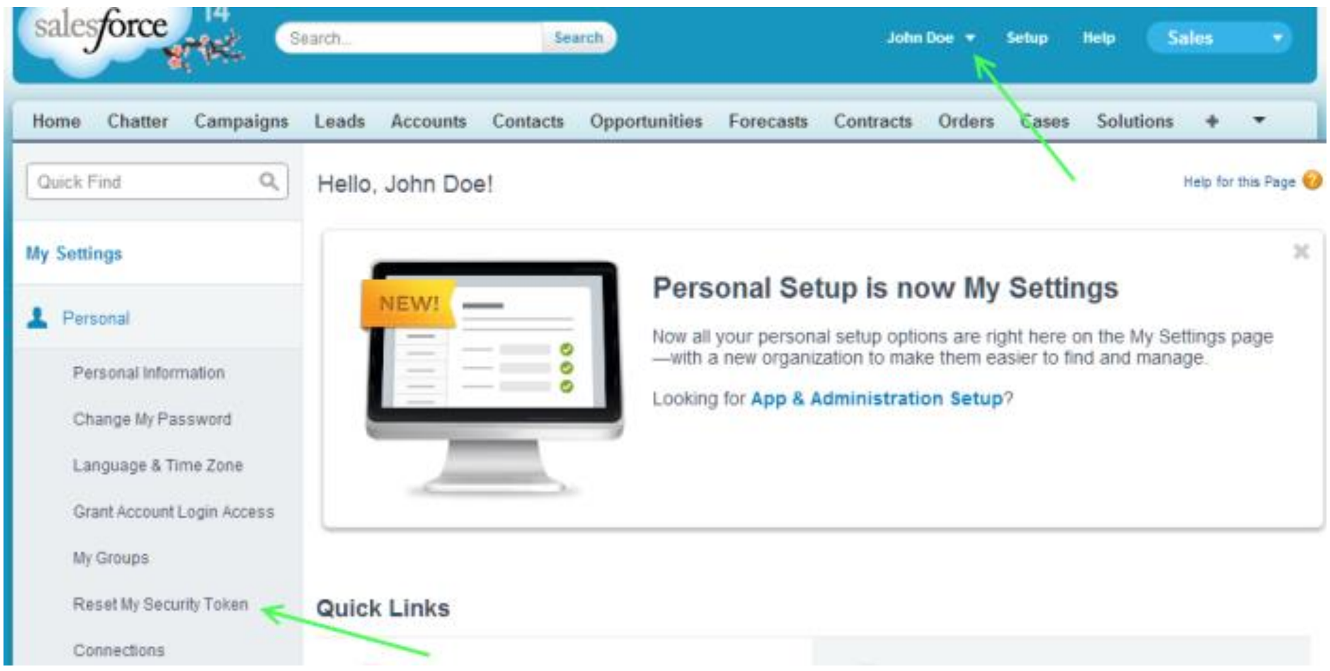
Step 1 - Sign up for an account with Salesforce

There are a few steps one has to take before starting to set up Action Form. First thing to do would be to have an account on Salesforce, with access to the SOAP API. This seems to depend on which type of subscription you have with Salesforce. You can also start with a new developer account which gets access to the API implicitly. You can open such an account at developer.salesforce.com. Or, if you already have a Salesforce account login at login.salesforce.com.

Note that you will need to activate the account, by clicking on a link you should receive in an activation email for your free Developer Edition account. Once the account is active, you will need a security token.

Step 2 - Get a Security Token from Salesforce

To get the security token, you will have to reset the security token, which is not so intuitive to do. In the upper right corner: <Your User Name> > My Settings. Then, on the left side: My Settings > Personal > Reset security token. After resetting it, a new security token will be sent to the account's email.



Step 3 - Setup Action Form

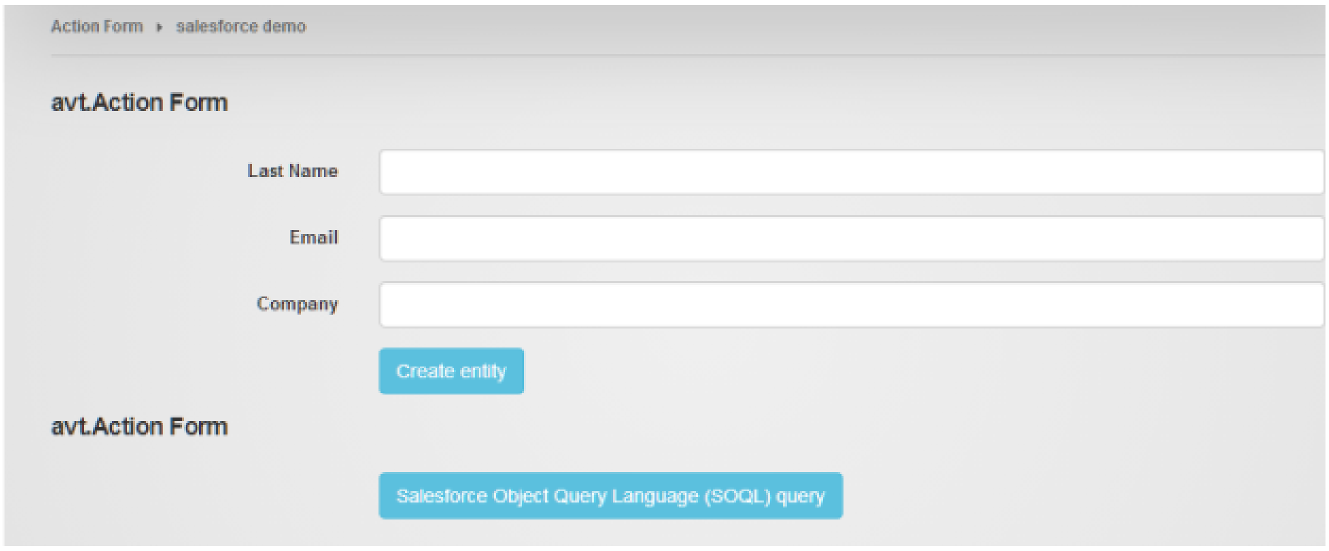
Once you have the Salesforce username, password and security token, you can proceed to setup Action Form. The Salesforce integration is provided as an add-on that can be purchased from DNN Store. So you will need to already have Action Form on the server before installing the Salesforce add-on. You can install both packages from Host > Extensions.

In this guide we will create two simple forms: one to create a Lead and one to query some data from Salesforce.

Create a Lead form

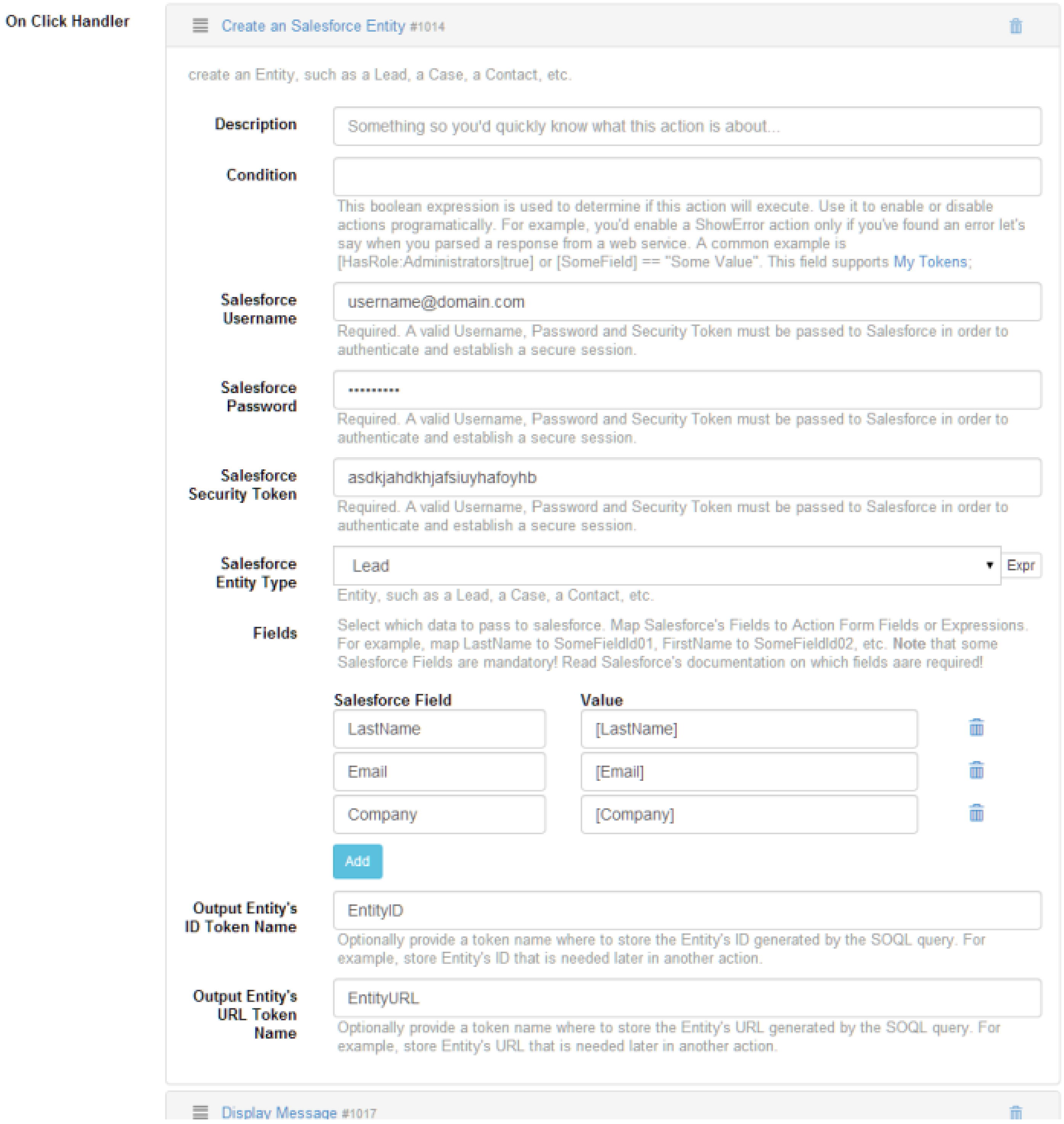
Start by adding a new Action Form module to the desired page. At the initial configuration step, for demo purposes, we will choose a Blank Form. It creates an empty form so you build everything from scratch. For our example we will add three text boxes to collect the Last Name, the Email and the Company for the new Lead. Add these using the Add Field button in Action Form - Manage Screen. Also add a button to perform the desired action.

Note that different entities have different requirements in terms of required fields. Make sure to read the documentation.


Action Form ▸ salesforce demo
avt.Action Form
Last Name
Email
Company
Create entity
avt.Action Form
Salesforce Object Query Language (SOQL) query

Add two actions on the On Click handler:

- a Create an Salesforce Entity action, for actually creating the entity



On Click Handler

Create an Salesforce Entity #1014

create an Entity, such as a Lead, a Case, a Contact, etc.

Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;
Salesforce Username	username@domain.com Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.
Salesforce Password	········· Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.
Salesforce Security Token	asdkjahdkhjafsiuyhafoyhb Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.
Salesforce Entity Type	Lead ▾ Expr Entity, such as a Lead, a Case, a Contact, etc.
Fields	Select which data to pass to salesforce. Map Salesforce's Fields to Action Form Fields or Expressions. For example, map LastName to SomeFieldId01, FirstName to SomeFieldId02, etc. Note that some Salesforce Fields are mandatory! Read Salesforce's documentation on which fields aare required!

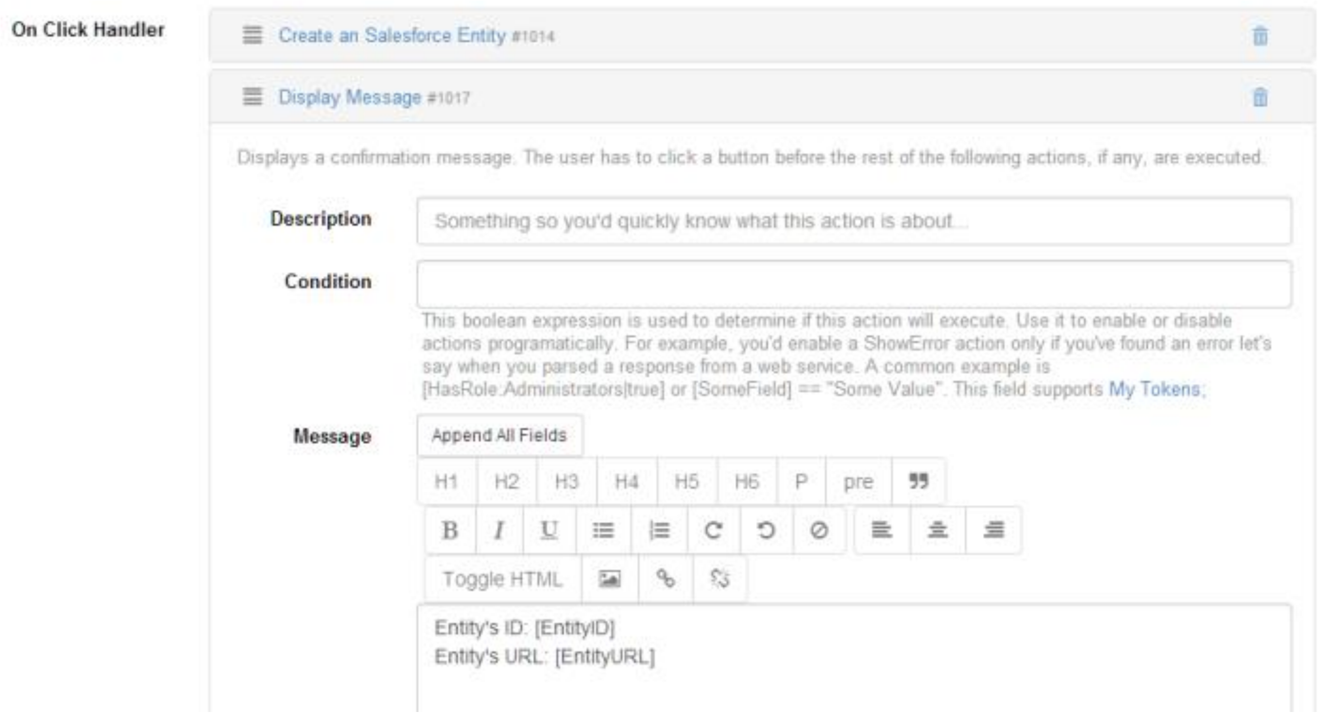
Salesforce Field	Value
LastName	[LastName]
Email	[Email]
Company	[Company]

Add

Output Entity's ID Token Name	EntityID Optionally provide a token name where to store the Entity's ID generated by the SOQL query. For example, store Entity's ID that is needed later in another action.
Output Entity's URL Token Name	EntityURL Optionally provide a token name where to store the Entity's URL generated by the SOQL query. For example, store Entity's URL that is needed later in another action.

Display Message #1017

- and a Display message action, for displaying the newly created entity's id and URL. You can pass these tokens for later use in other actions.



Displays a confirmation message. The user has to click a button before the rest of the following actions, if any, are executed.

Description: Something so you'd quickly know what this action is about...

Condition:

This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;

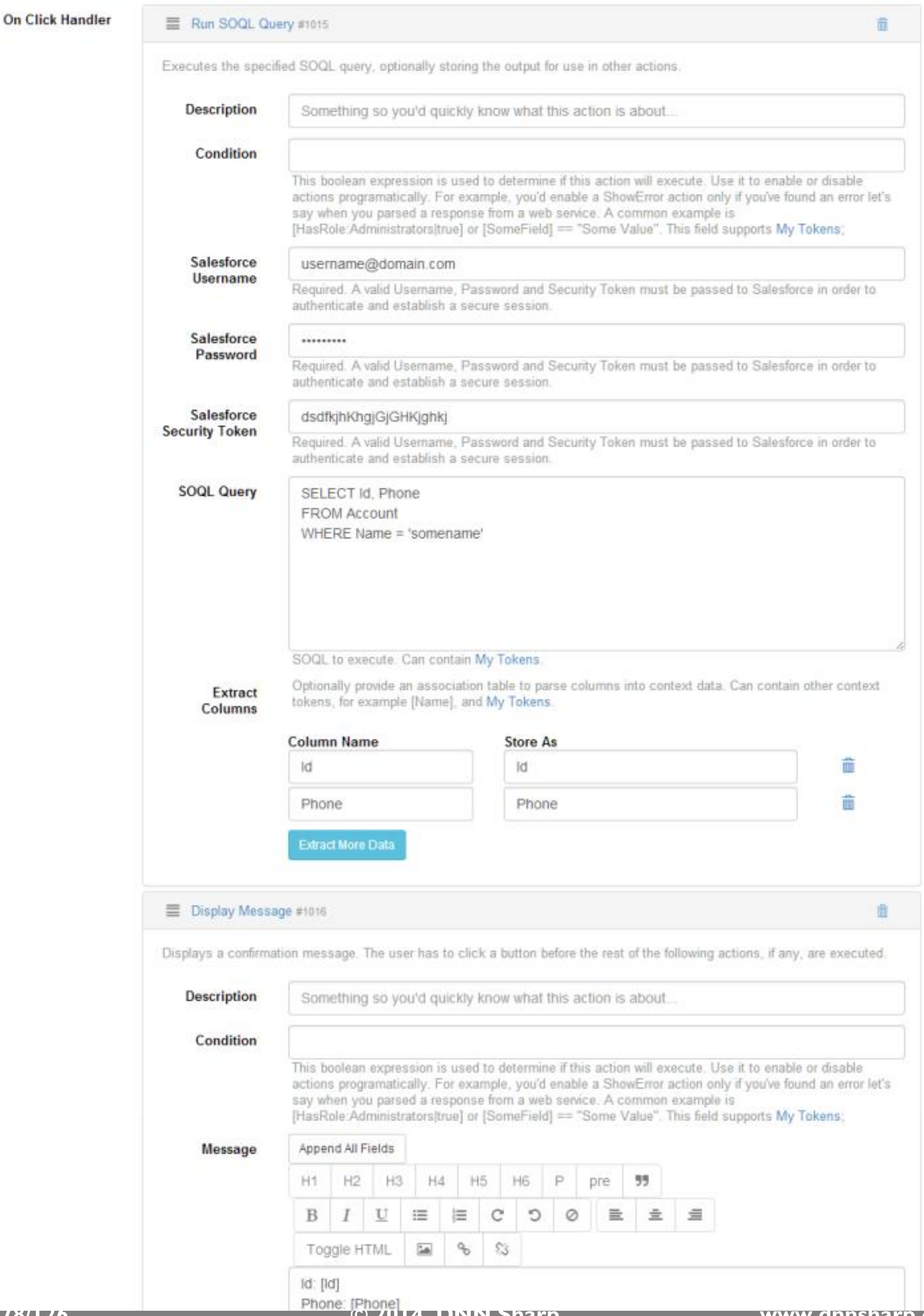
Message:

Entity's ID: [EntityID]
Entity's URL: [EntityURL]

Query Salesforce form

Start by adding a new Action Form module to the desired page. At the initial configuration step, for demo purposes, we will choose a Blank Form. It creates an empty form so you build everything from scratch. For our example we will need a button to perform the desired action. Add this using the Add Field button in Action Form - Manage Screen.

SOQL is very similar to SQL. Make sure to read the documentation at http://www.salesforce.com/us/developer/docs/soql_sosl/index_Left.htm. The example below returns the Id and Phone of an Account for which we know the name.



On Click Handler

Run SOQL Query #1015

Executes the specified SOQL query, optionally storing the output for use in other actions.

Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;
Salesforce Username	username@domain.com Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.
Salesforce Password	••••••••• Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.
Salesforce Security Token	dsdfkjhKhgjGjGHKjghkj Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.

SOQL Query

```
SELECT Id, Phone
FROM Account
WHERE Name = 'somename'
```

SOQL to execute. Can contain My Tokens.

Extract Columns

Optionally provide an association table to parse columns into context data. Can contain other context tokens, for example [Name], and My Tokens.

Column Name	Store As
Id	Id
Phone	Phone

Extract More Data

Display Message #1016

Displays a confirmation message. The user has to click a button before the rest of the following actions, if any, are executed.

Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;

Message

Id: [Id]
Phone: [Phone]

Settings Reference

- **Salesforce Username**

Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.

- **Salesforce Password**

Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.

- **Salesforce Security Token**

Required. A valid Username, Password and Security Token must be passed to Salesforce in order to authenticate and establish a secure session.

- **Salesforce Entity Type**

The entity type intended to create. There is a dropdown list with the most common type entities, Lead, Case, Contact, Account, but you can always select a different entity by clicking on the Expr button at the end of the dropdown list, and entering manually your own type of entity.
You can see a list of standard objects and their standard fields in the Salesforce documentation.

- **Fields**

Select which data to pass to salesforce. Map Salesforce's Fields to Action Form Fields or Expressions. For example, map LastName to SomeFieldId01, FirstName to SomeFieldId02, etc. Note that some Salesforce Fields are mandatory! Read Salesforce's documentation on which fields are required!
You can see a list of standard objects and their standard fields in the Salesforce documentation. Lookout for the required fields! Different object have different required fields.

- **Output Entity's ID Token Name**

Optionally provide a token name where to store the newly created Entity's ID. For example, store Entity's ID that is needed later in another action.

- **Output Entity's URL Token Name**

Optionally provide a token name where to store the newly created Entity's URL. For example, store Entity's URL that is needed later in another action, like redirect, etc.

- **SOQL Query**

Salesforce Object Query Language (SOQL) query to execute. Can contain My Tokens. For more info on SOQL, check Force.com SOQL Reference.

- **Extract Columns**

Optionally provide an association table to parse columns into context data. Can contain other context tokens, for example [Name], and My Tokens.

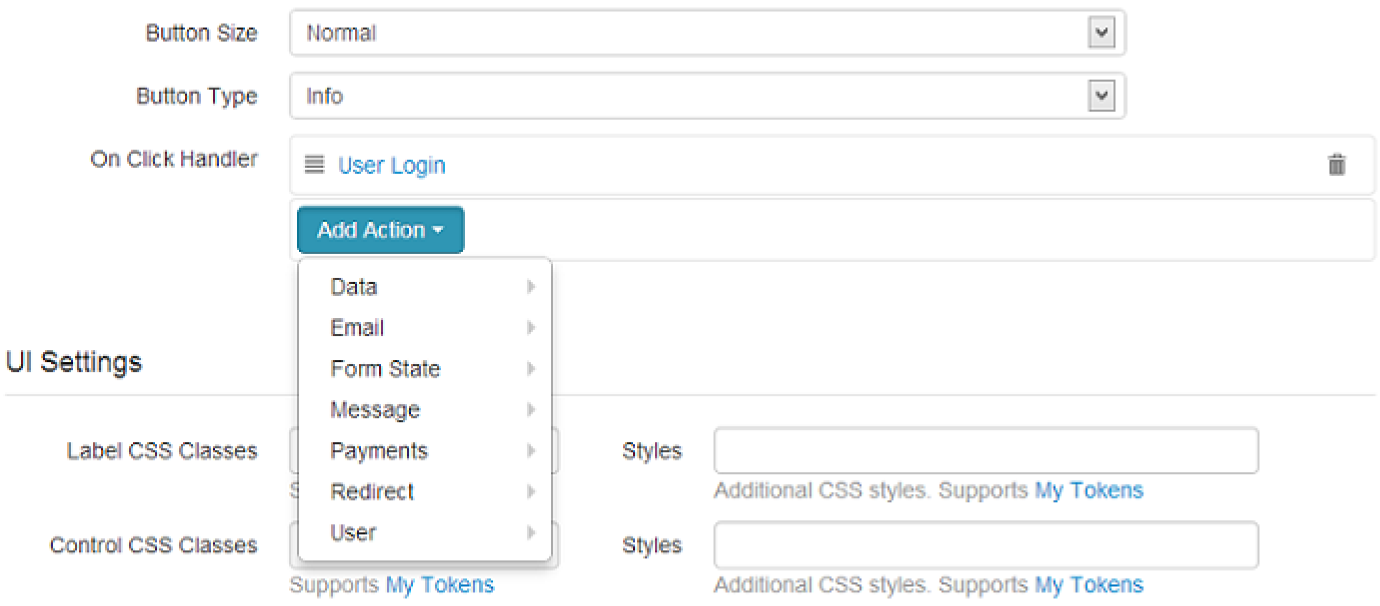
Form Actions

Action Form comes with a dozen different action types and the order in which they are added to the list matters a lot for a few different reasons. First of all, Action Form provides a shared context where Actions can store their data. This means that actions at the top can provide data for other actions down the stack to use.

In Action Form you can set Actions in three places:

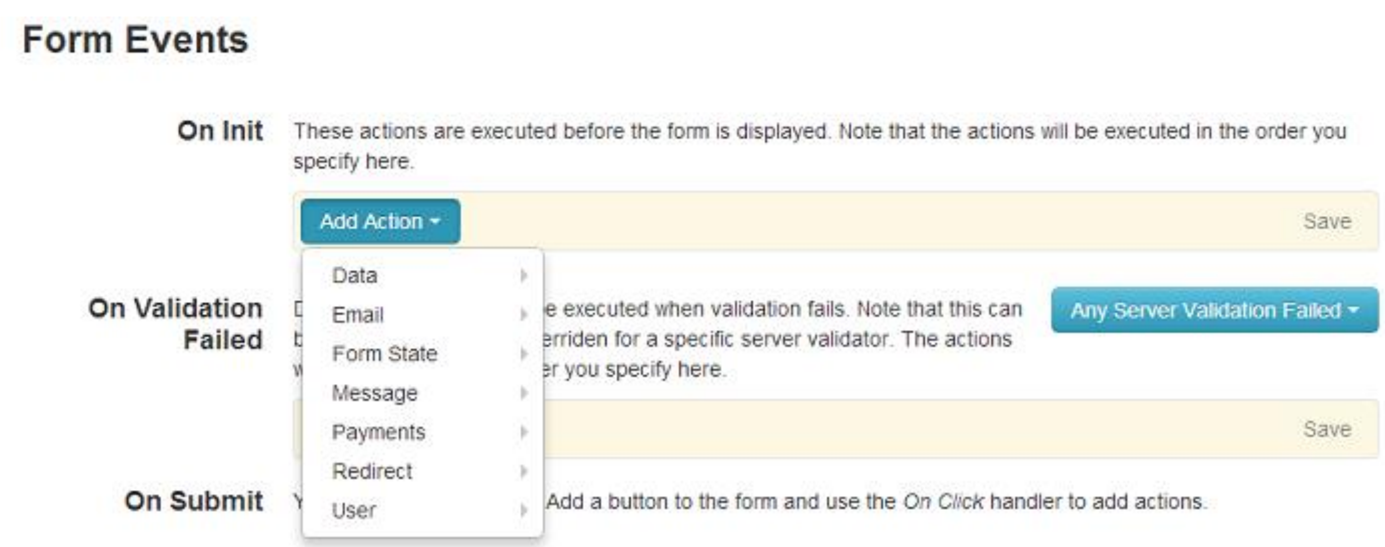
1. Set actions on the submit button.

Note that before you can set actions, you need to add a button field for your form.



Button Size	Normal
Button Type	Info
On Click Handler	User Login

2. Set actions On Init Event


Form Events

On Init These actions are executed before the form is displayed. Note that the actions will be executed in the order you specify here.

On Validation Failed …e executed when validation fails. Note that this can …erriden for a specific server validator. The actions …er you specify here.

On Submit …Add a button to the form and use the *On Click* handler to add actions.

3. Set actions On Validation Failed Event

Check out the specific sections for more details about each action that you can set in Action Form.

Context

The context is like a container where actions can read and write data. The Context action has two options: Inject Form Data and Load User.

The Inject Form Data action lets you manually add and write data in the form. This action is very useful, for example, when you want to add data which is specific to the button when you click on it or for when you want to append data that you don't want exposed through hidden fields. Note that this is only valid for current action list, it does not persist until next list of actions executes, for example on submit. Each piece of data in the context is stored under a name and can be accessed through tokens using that name. Use this action to load data that other actions down the stack need.

The values inserted into context can be constant text or tokens, including tokens created with My Tokens. Loading over a name that already exists will overwrite the existing data. If a field with the same name exists, data is also loaded into the form.

Load User Action

A form runs in the context of a user, it could be the current user performing the submit or it could be the result of other actions. This lets you manually set a user. Note that this is only valid for current action list - it does not persist until next list of actions executes, for example, on submit.

Data

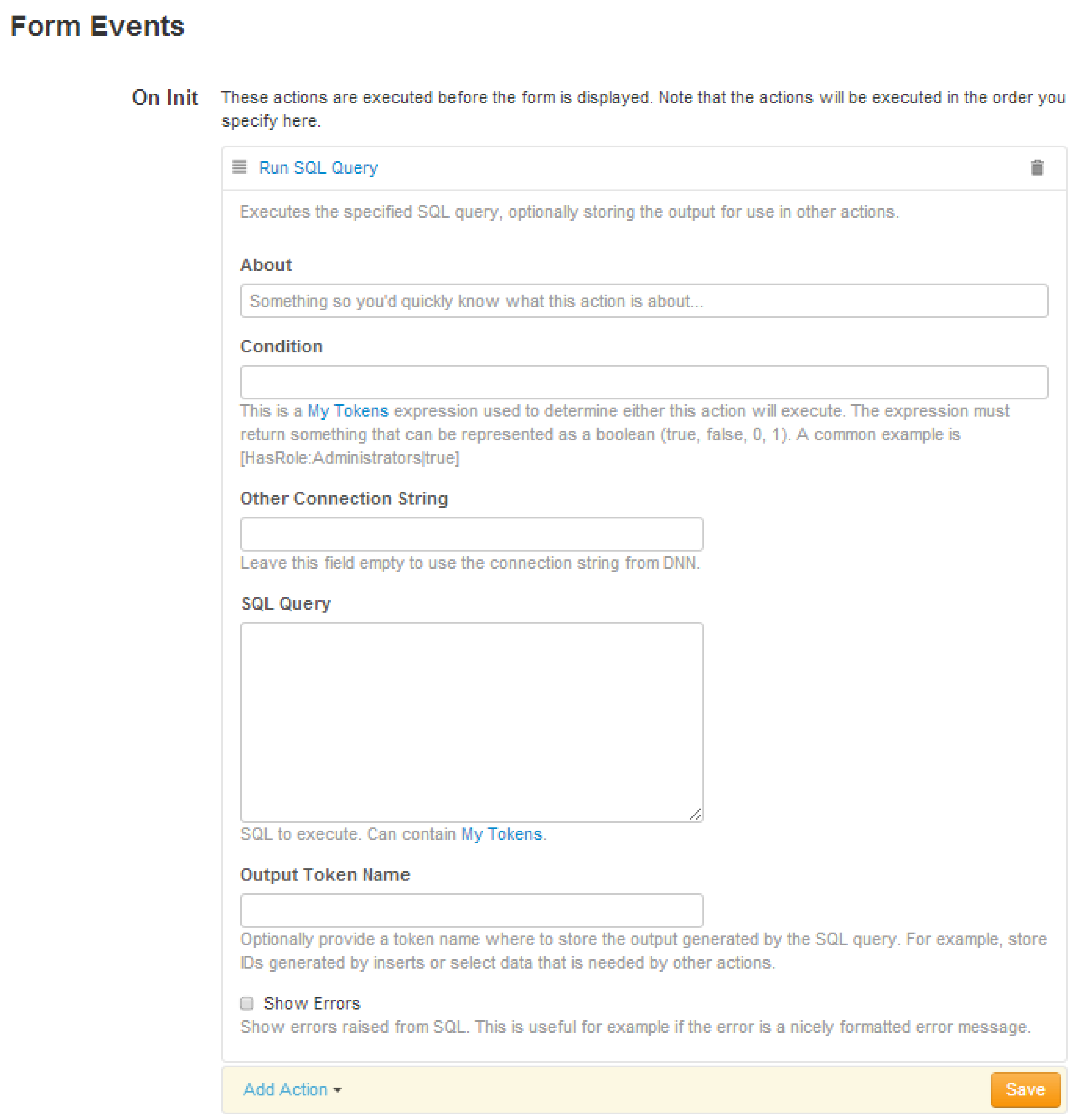
Run SQL Query

This action executes an SQL statement, optionally capturing the output. The SQL runs in the context of the DNN database, but there are plans to extend it to also allow a connection string or a connection string name that will make it possible to run in other databases as well – not restricted to SQL Server either. This action supports context tokens and My Tokens inside the SQL query.

Here are some common scenarios when you would use this action:

- Use an UPDATE statement to calculate some statistics for large databases on an interval, instead of calculating them on every call.
- Execute an SELECT statement to retrieve data to be used with other actions down the stack.
- Flush old temporary data using a DELETE statement.
- Execute a stored procedure that calculates commissions paid through a referral program.

Currently, only one field can be captured from an SQL action, so make sure that your query returns the data you need on first column of the first row. We may extend this in the future to be able to store multiple columns. As a workaround, either create multiple SQL actions, or produce the final text output directly from the SQL query. For example, if you need the full name of a user, you can use something like SELECT FirstName + ' ' + LastName from Users where UserId = [UserId].



On Init These actions are executed before the form is displayed. Note that the actions will be executed in the order you specify here.

Executes the specified SQL query, optionally storing the output for use in other actions.

About

Something so you'd quickly know what this action is about...

Condition

This is a My Tokens expression used to determine either this action will execute. The expression must return something that can be represented as a boolean (true, false, 0, 1). A common example is [HasRole:Administrators|true]

Other Connection String

Leave this field empty to use the connection string from DNN.

SQL Query

SQL to execute. Can contain My Tokens.

Output Token Name

Optionally provide a token name where to store the output generated by the SQL query. For example, store IDs generated by inserts or select data that is needed by other actions.

☐ Show Errors

Show errors raised from SQL. This is useful for example if the error is a nicely formatted error message.

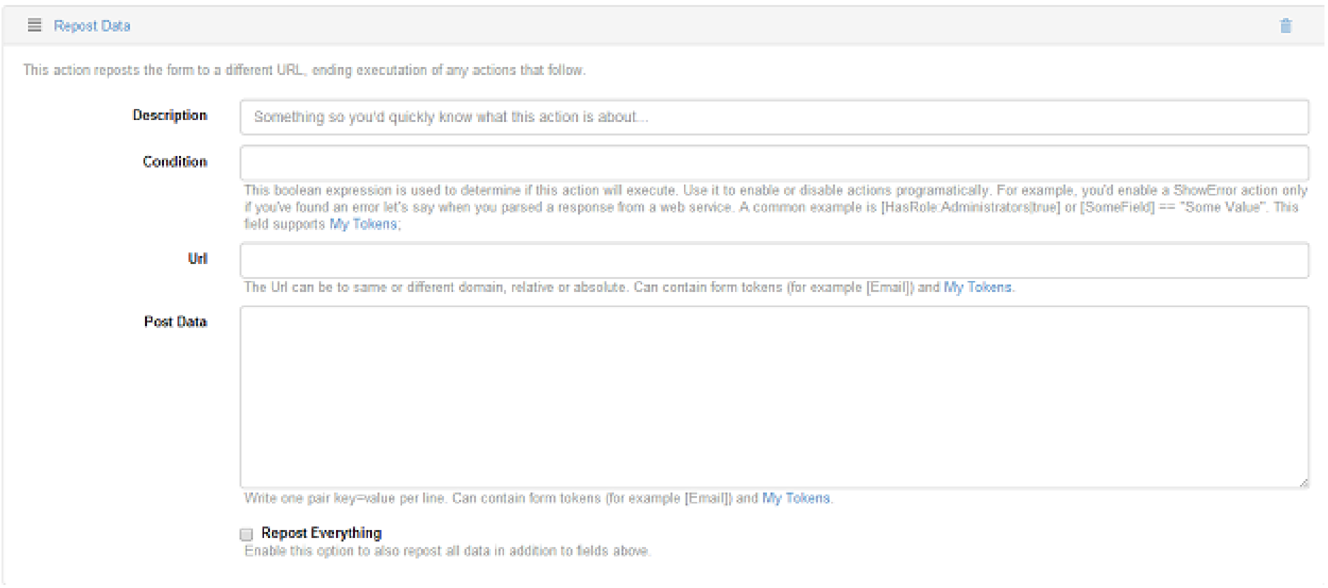
Server Request

This action has the purpose of posting the form data to a different server, you can use the URL field to determine the location for where the data will be posted, in this box you can also use tokens, and there's another useful box, the Post Data box, where you can put key = value pairs on separate lines and you can also add additional HTTP headers, in case you need it, and you can specify the HTTP method by selecting it from the "Choose an HTTP Method" drop down list.

Repost Data

This action reposts the form to a different URL, ending execution of any actions that follow. This action can be used to make an HTTP request to a different server, optionally sending data. Often, this means invoking a web service. The following fields can be configured:

- **URL.** This represents the URL to make the request to. A common mistake is to forget to include the protocol. For example www.domain.com/webservice is wrong. Instead, use http://www.domain.com/websservice. Optionally, append the query string directly to the URL after the question mark. For examplehttp://www.domain.com/websservice?q=test&p=1. This field supports context tokens and My Tokens.
- **POST Data**. This is data to send to the URL using POST operation. Put key=value pairs, each on a separate line. It's also possible to post whole messages, for example and XML (that SOAP-like services expect) by simply putting the XML without any lines. This field supports context tokens and My Tokens.


Repost Data

This action reposts the form to a different URL, ending executation of any actions that follow.

Description: Something so you'd quickly know what this action is about...

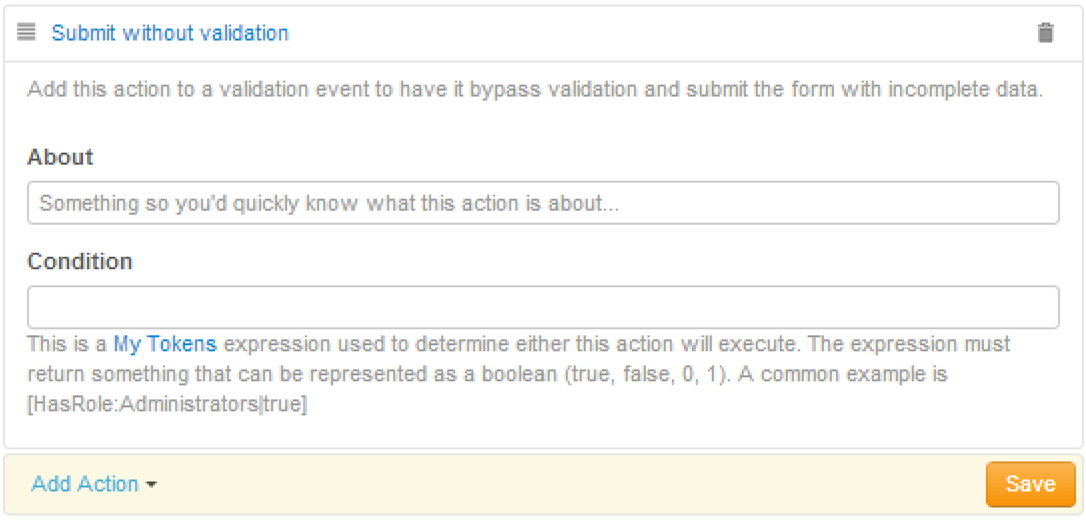
Condition:
This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;

Url:
The Url can be to same or different domain, relative or absolute. Can contain form tokens (for example [Email]) and My Tokens.

Post Data:
Write one pair key=value per line. Can contain form tokens (for example [Email]) and My Tokens.

☐ Repost Everything
Enable this option to also repost all data in addition to fields above.

Submit Without Validation



Add this action to a validation event to have it bypass validation and submit the form with incomplete data.

About

Something so you'd quickly know what this action is about...

Condition

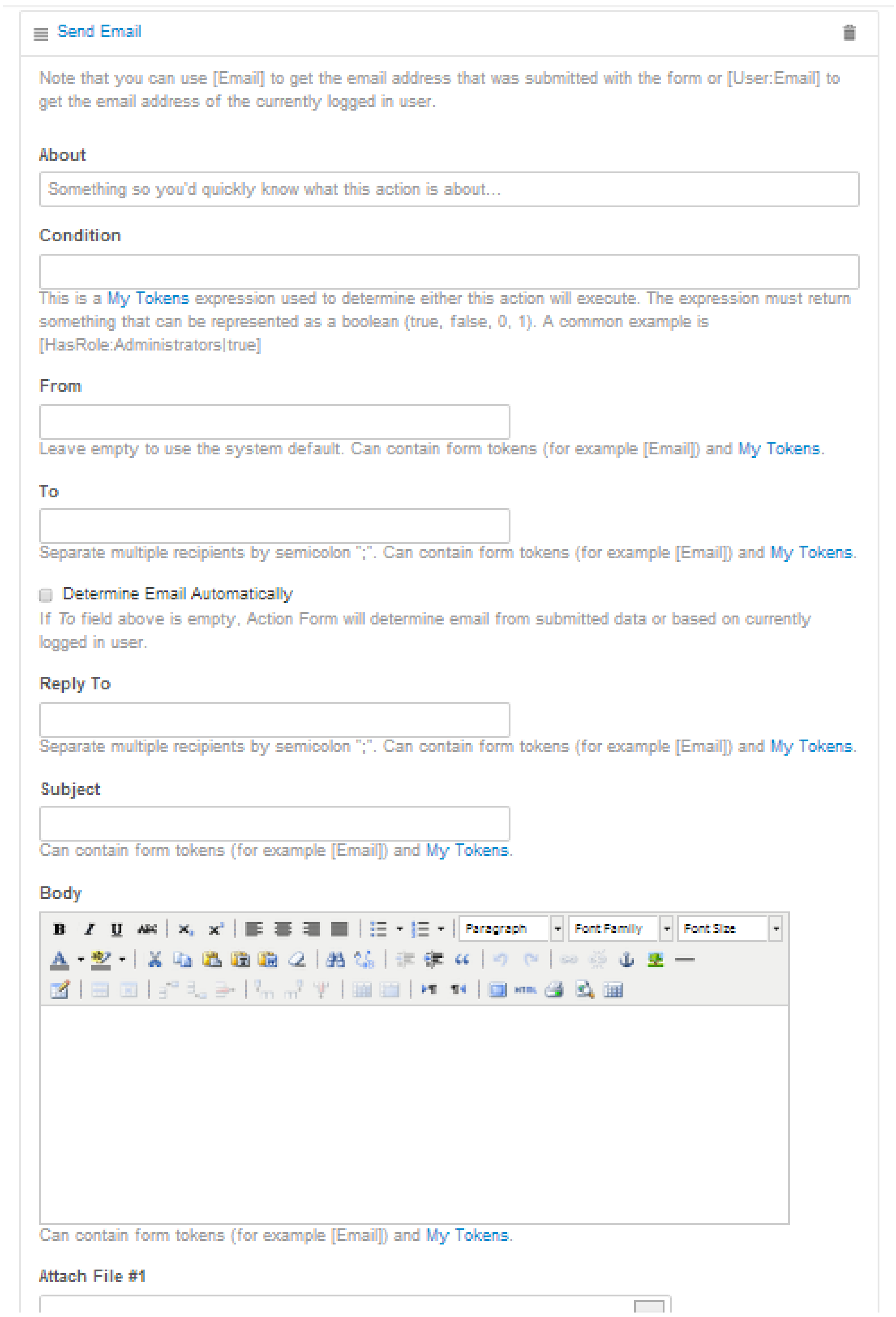
This is a My Tokens expression used to determine either this action will execute. The expression must return something that can be represented as a boolean (true, false, 0, 1). A common example is [HasRole:Administrators|true]

Email

Send Email

This action does exactly what it says. It sends an email using the SMTP server that is defined in DNN Host Settings. The following fields can be configured:

- **From.** This is the sender of the email as it will appear to the recipient. Leave empty to use the system default – if a portal context is set, the portal administrator email is used; otherwise, the email defined in Host Settings is used. This field supports context tokens and My Tokens.
- **To.** Determines who will receive the email. Separate multiple email addresses with semicolon. If the Determine Automatically option is used, then Sharp Scheduler tries to infer the email from the Context User. This field supports context tokens and My Tokens.
- **Reply To.** When the recipient of the email replies to this email, the To field fills with this email instead of the original From. This only makes sense when sending emails to non-admin users that are likely to reply to those emails. For example, it can be a trial reminder email. Separate multiple email addresses with semicolon. This field supports context tokens and My Tokens.
- **Subject.** For best experience, choose something not too short but not too long either. This field supports context tokens and My Tokens.
- **Body.** This is the email content. If you need custom data then use other actions, for example the Run SQL action, to fetch it and store it in the job context, then access it through the context tokens. This field also supports My Tokens.
- **Attachments.** This action allows you to attach up to 5 email attachments to the email. If a Portal Context is specified, then the file picker will show files that belong to that portal, otherwise, files from the Host folder are listed.



Send Email

Note that you can use [Email] to get the email address that was submitted with the form or [User:Email] to get the email address of the currently logged in user.

About

Something so you'd quickly know what this action is about...

Condition

This is a My Tokens expression used to determine either this action will execute. The expression must return something that can be represented as a boolean (true, false, 0, 1). A common example is [HasRole:Administrators|true]

From

Leave empty to use the system default. Can contain form tokens (for example [Email]) and My Tokens.

To

Separate multiple recipients by semicolon ";". Can contain form tokens (for example [Email]) and My Tokens.

☐ Determine Email Automatically

If *To* field above is empty, Action Form will determine email from submitted data or based on currently logged in user.

Reply To

Separate multiple recipients by semicolon ";". Can contain form tokens (for example [Email]) and My Tokens.

Subject

Can contain form tokens (for example [Email]) and My Tokens.

Body

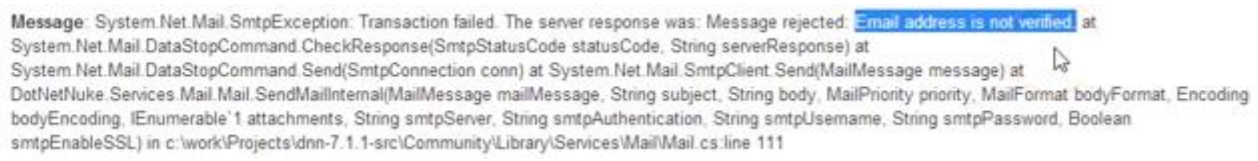
Can contain form tokens (for example [Email]) and My Tokens.

Attach File #1

Troubleshooting

Before starting everything, test that the form is working perfectly. Go to Admin > Event Viewer to see if you get any errors. If the form doesn't work as it should be, you need to verify some settings.

1. Make sure that the SMTP settings are configured properly. You can find this in Host Settings. Follow this path: Host > Host Settings > Advanced Settings Test that the Username and Password used to connect to the Server are correct. You can do this by clicking on the Test SMTP Settings button.
2. Make sure that the email address used to send the form is allowed to send emails. Action Form uses the system default (site email) which is the Administrator's email. Some servers like Amazon need the email address used to send the form to be verified. Only specific email addresses are allowed to send emails. In Event Viewer, you might get the next error:



```
Message: System.Net.Mail.SmtpException: Transaction failed. The server response was: Message rejected: Email address is not verified. at
System.Net.Mail.DataStopCommand.CheckResponse(SmtpStatusCode statusCode, String serverResponse) at
System.Net.Mail.DataStopCommand.Send(SmtpConnection conn) at System.Net.Mail.SmtpClient.Send(MailMessage message) at
DotNetNuke.Services.Mail.Mail.SendMailInternal(MailMessage mailMessage, String subject, String body, MailPriority priority, MailFormat bodyFormat, Encoding
bodyEncoding, IEnumerable`1 attachments, String smtpServer, String smtpAuthentication, String smtpUsername, String smtpPassword, Boolean
smtpEnableSSL) in c:\work\Projects\dnn-7.1.1-src\Community\Library\Services\Mail\Mail.cs:line 111
```

So, if in Event Viewer you get the Email address not verified error, you need to set up in the From field the Host email address. The host email could be found under the Host > Host Settings > Basic Settings. Follow this path to set-up the email address: Manage Action Form > Fields > Send (submit button) > Send Email (action) -> Click on the Save button and now your Action Form should work as expected.

It is not difficult to set up Action Form but there are a few aspects and details to look after which are difficult to debug if you don't know them or if you don't know where to look. In the scenario from above, it was simple because the server told us that he rejected the email and why. In other cases, the server will not tell you the reason, so you will have to figure it out by yourself or sometimes it will not even tell you that it was rejected. It will just show the successfully submitted message but the recipient will not get it, probably the email just remains blocked somewhere in a pick-up folder for example. If the server doesn't do its job, the email will remain in that pick-up folder and no error will get reported. Once again, if there is an error, Action Form will log it in Admin > Event Viewer.

Subscribe to MailChimp

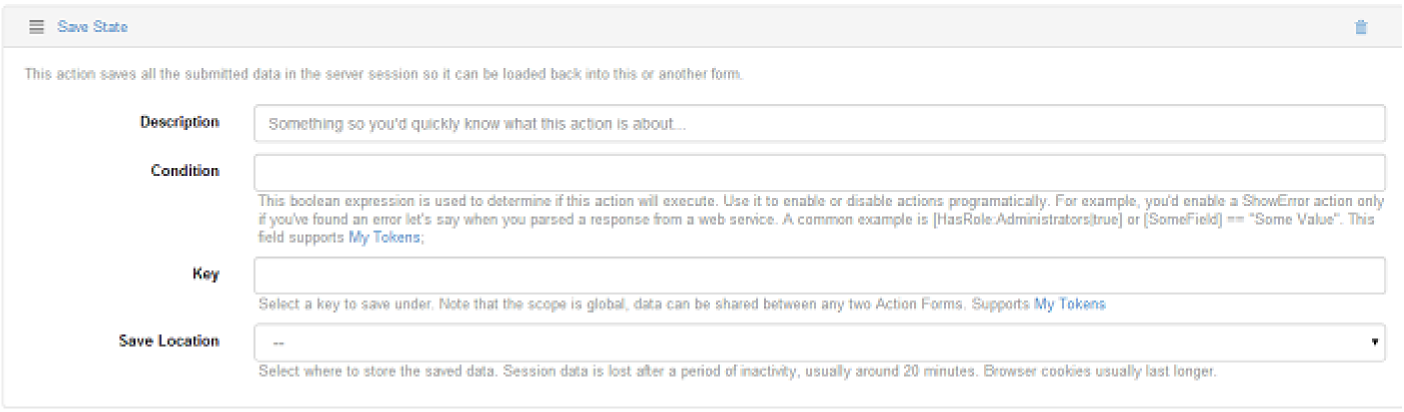
You can set up this action to a button so when the form is submitted it subscribes an email address to a Mailchimp list. The following fields need to be configured so that the form will subscribe the email address to your list:

- **API Key.** Login to MailChimp and go to Account > Extra > API Keys. Follow this link to find out more about MailChimp API Key. Action Form requires an API key to successfully connect to a MailChimp account and transfer subscriber information and other data to your list.
- **List Name.** The list name has to be exactly as it appears in your MailChimp account.

Form State

Save State

This action saves all the submitted data in the server session so it can be loaded back into this or another form. You need to use a key, in the key field, under which the data will be saved, and you also need to specify where the saved data will be stored in the Save Location, either by selecting the Server Session or the Browser Cookies. Keep in mind that the server data is lost after a period of inactivity, usually after 20 minutes, browser cookies usually last longer.


Save State

This action saves all the submitted data in the server session so it can be loaded back into this or another form.

Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;
Key	Select a key to save under. Note that the scope is global, data can be shared between any two Action Forms. Supports My Tokens
Save Location	-- Select where to store the saved data. Session data is lost after a period of inactivity, usually around 20 minutes. Browser cookies usually last longer.

Auto Save State

Add this action to "On Init" event to auto save form state in browser cookies. Load it back later by using Load State action, normally also placed in the initialization event.

Clear State

This action has the purpose of clearing the previously saved state, for example, if you have a contact form and on its button you've set the Save state on server session action, to not keep the values that the user fills in the boxes, you can use the Clear state action so that the form refreshes and clears all the data which was inserted in the boxes.

Load State

As it says, from the label, this action will load the previously saved state in the form.

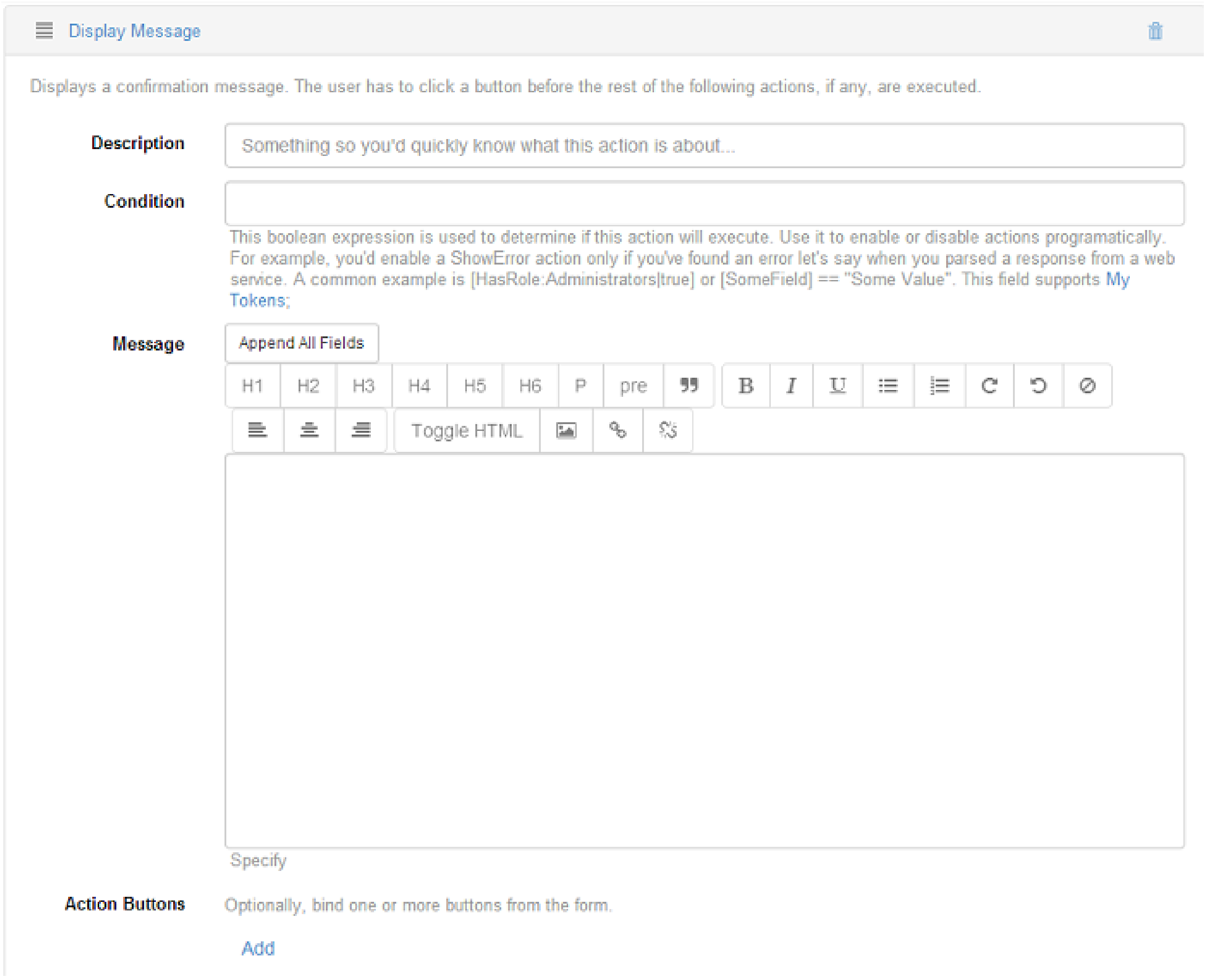
Message

Action Form allows you to set actions to display confirmation and error messages after the user submits the form, with these actions, you can use tokens, for example [Name] and My Tokens to apply a confirmation message in the Message text box. Every form should have a confirmation message in order to inform the user whether the form data was successfully sent or if there were any errors.

Display Message

This action does exactly what it says. It displays a message after the form is submitted. What you can configure:

- **Message.** In this text box you can input your message that will be displayed after the form is submitted. Example: "Thank you for contacting us. Your message was successfully sent." This field also supports My Tokens. This option is also one of the simplest and fastest way for debugging. Populate it with tokens and see if a form is working correctly.
- **Action Buttons.** Optionally, you can bind one or more buttons from the form. For example, you can add a button and set an action on it to redirect to another page or to download a file.



Displays a confirmation message. The user has to click a button before the rest of the following actions, if any, are executed.

Description	Something so you'd quickly know what this action is about...
Condition	This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;
Message	Specify
Action Buttons	Optionally, bind one or more buttons from the form.

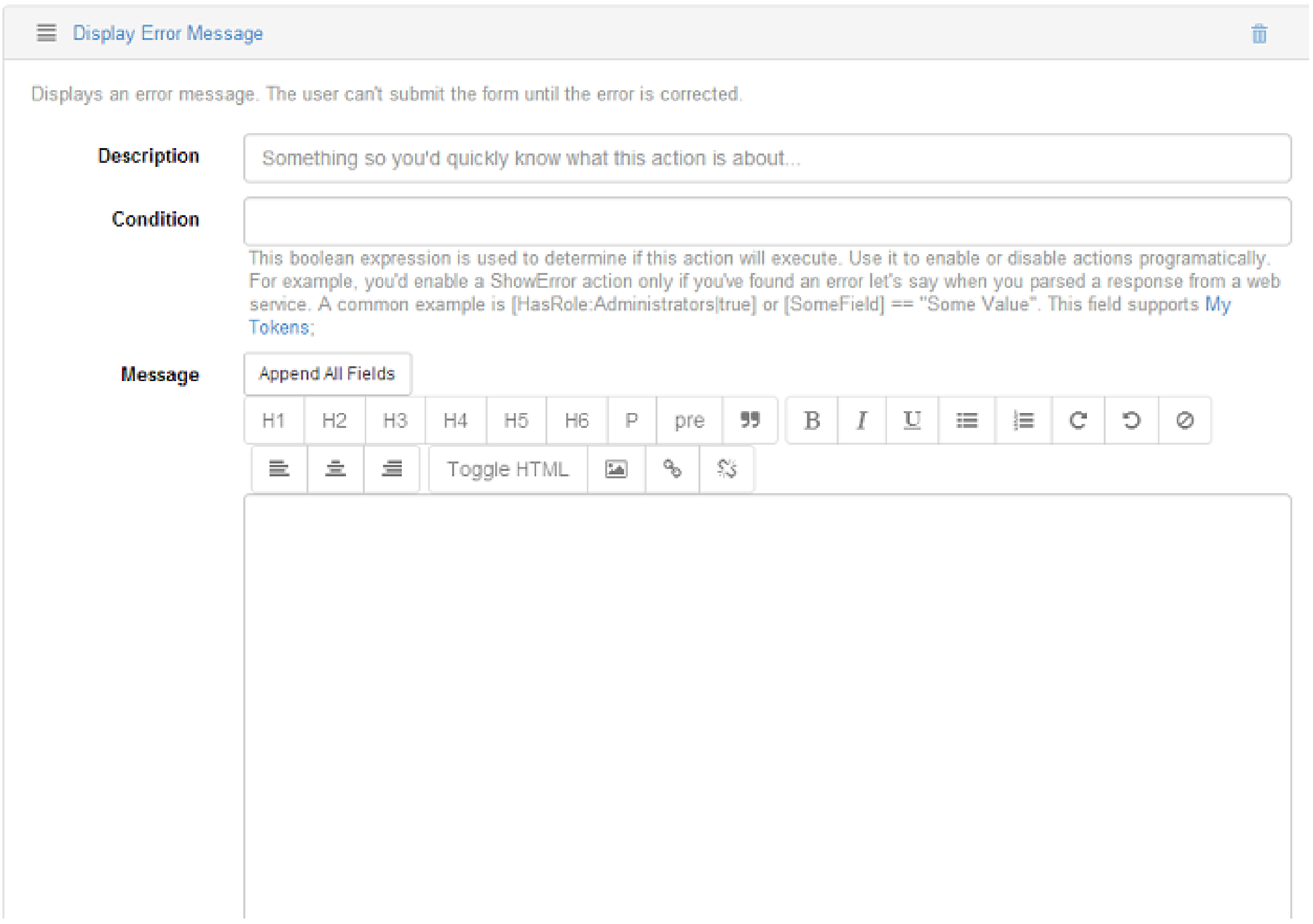
Update Form Data (AJAX)

This action sends current context back to the browser, updating the form fields.

Display Error Message

This action displays a message if there are some errors while the user is trying to submit the form. It's similar to Display Message action except that you cannot add a button. This action, along with Display Message, can also be used on the section "On Validation Failed". Here's what you can configure:

- **Message.** In this text box you can input you message that will be displayed if there are errors when the form is submitted. Example: "Errors occurred while sending the form. Please correct the errors and try again." This field also supports My Tokens.



Displays an error message. The user can't submit the form until the error is corrected.

Something so you'd quickly know what this action is about...

This boolean expression is used to determine if this action will execute. Use it to enable or disable actions programatically. For example, you'd enable a ShowError action only if you've found an error let's say when you parsed a response from a web service. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens;

Payments

Collect PayPal Payment

Action Form allows you to set a PayPal payment requirement when the user submits the form. To set this requirement you will need to add the Collect Paypal Payment action. It allows you to set PayPal payments by submitting your Sandbox Account Email, Live Account Email, Payment Amount, Payment Frequency, and Currency preference. You can also apply tokens to the email addresses, item title and payment amount fields.

The Collect Paypal Payment action allows you to set the requirements before giving the user access to specific resources or rights. When the users submit the form, they are redirected to the PayPal website, where they are able to complete the payment process. The user is required to log in in order to complete the payment process, but if he does not have a Paypal account, he is required to create one.

- **Sandbox Account Email**

The Sandbox Account Email field is used for entering your PayPal Sandbox email address. This is the email address you used when you've set up your PayPal Sandbox account. Paypal Sandbox is a testing environment that allows you to test transactions without monetary transactions. You are required to enter a valid email address. If you do not enter a valid email you will get an error message.

- **Live Account Email**

The Live Account Email field is used for entering your Live PayPal email address. This is the email address you've used to open your Paypal account. If you do not enter a valid email you will get an error message.

- **Recurring Payment**

The Recurring drop down menu includes a list of ongoing payment options:
Select None to require a recurring payment.
Select Monthly to specify a monthly recurring payment.
Select Yearly to specify a yearly recurring payment.
Item Title
The Item Title field is used for entering the title of the product or service.

- **Currency Code**

The Currency Code drop down menu is used for specifying your currency preference, for example U.S Dollars. The drop down menu includes a list of currency preferences, displayed in the list below.

Select U.S. Dollars (USD) to specify U.S Dollars as the currency preference.
Select Euros (EUR) to specify Euros as the currency preference.
Select Canadian Dollars (CAD) to specify Canadian dollars as the currency preference.
Select Pounds Sterling (GBP) to specify English pounds as the currency preference.
Select Australian Dollar (AUD) to specify Australian dollars as the currency preference.

- **Amount**

The Amount field is used for entering the amount for the product or service. You are required to enter a valid number. If you enter letters or symbols you will get an error message. Do not enter the dollar ($) symbol, this is invalid and will return an error message. Only numbers and decimal points are allowed, for example 5.55.

- **Cancel Page**

The Cancel Page drop down menu includes a list of pages for redirecting the user when Cancel is selected. The list of pages in the Cancel Page menu depends on the pages available in your site.
Pending Payment Page
The Pending Payment Page drop down menu includes a list of pages for redirecting the user when the payment is pending. The list of pages in the Cancel Page menu depends on the pages available in your site.

- **Generate Unique Order ID**

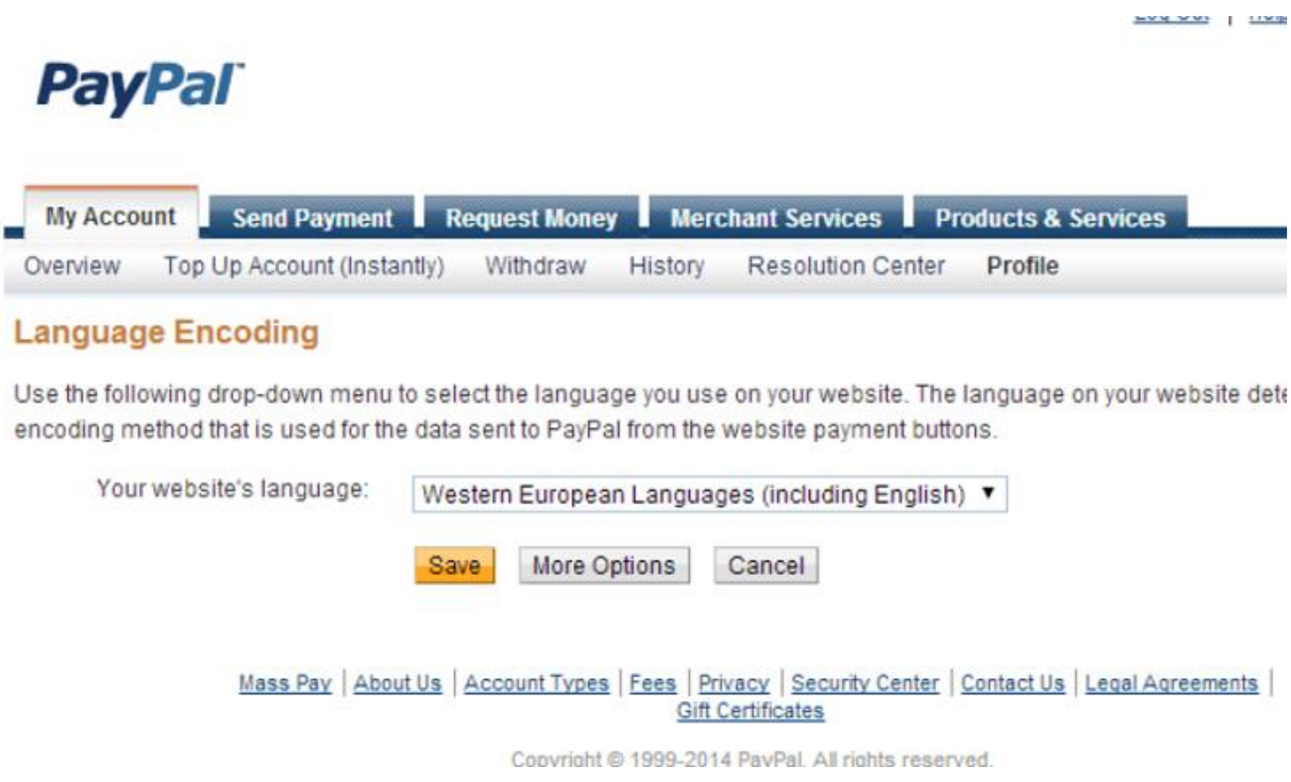
If you check this option box, an unique ID will be generated for each order the users makes.

Payments without PayPal account

By default, PayPal requires customers that they either already have a PayPal account or create one during the checkout process. This option can be configured from PayPal so customers can buy without having a PayPal account, by using a credit card for example.
More information can be found here

Troubleshooting

If you are unlucky and encountered this error: "Data does not match input character set or default encoding. For more information, please contact the merchant." you should check the encoding language on your PayPal Account.
Go into your PayPal account > Profile > My Selling Tools (in right menu) > PayPal button encoding language (a link at the bottom) and see what's displayed there. Your settings for website's language should look like in the image below. Set it this way and the error should be gone.


My Account
Send Payment
Request Money
Merchant Services
Products & Services
Overview
Top Up Account (Instantly)
Withdraw
History
Resolution Center
Profile

Language Encoding

Use the following drop-down menu to select the language you use on your website. The language on your website dete
encoding method that is used for the data sent to PayPal from the website payment buttons.

Your website's language: Western European Languages (including English)

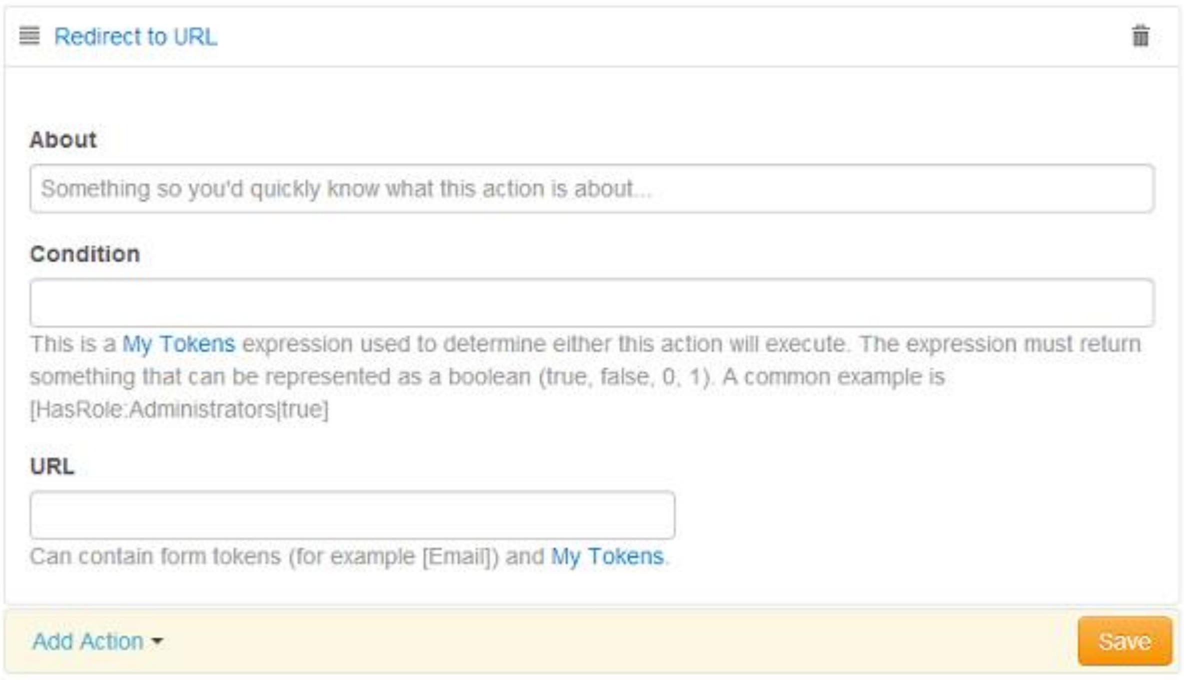
Save
More Options
Cancel
Mass Pay | About Us | Account Types | Fees | Privacy | Security Center | Contact Us | Legal Agreements |
Gift Certificates
Copyright © 1999-2014 PayPal. All rights reserved.

Redirect

Action Form allows you to redirect the user to a specified location after the user submits the form. This action requires that you specify a redirect location. You can redirect the user to an URL, another page on the portal or to a location from where he can download a file from your portal.

Redirect to URL

This option redirects the user to an external link after the user submits the form. Input your URL in the URL field - here you have also the ability to use tokens.


Redirect to URL

About

Something so you'd quickly know what this action is about...

Condition

This is a My Tokens expression used to determine either this action will execute. The expression must return something that can be represented as a boolean (true, false, 0, 1). A common example is [HasRole:Administrators|true]

URL

Can contain form tokens (for example [Email]) and My Tokens.

Redirect to Portal Page

This option redirects the user to another page in the portal after the user submits the form. Chose a portal page from the drop down menu, on the Page field.

Send File for Download

This option redirects the user to a file in the portal to be downloaded after the user submits the form. Before you set up this action, you will need to upload your file to your portal folder. After this, you will find the file just by selecting it from the drop down list on the File field.

User

User Login

Action Form allows you to require the user to be logged in to submit the Action Form, this a case when you have a multi-step form. You can also use Action Form as an Login module, it's very simple just by using the predefined Login Form template - this is a simple form with username and password fields which logs the user in. Note that the form must contain the Username and Password fields in order for the User Login action applied on the button to work.

User Registration

Action form allows you to specify an action for creating user account when the user submits the form. To apply this action, the form must have at least an email address field. The Action Form sets the email address as the username. This option is only applicable for new users. User Registration Options:

- **Email as Username**

Select this option to create the user account using the email address provided. This option will use the email address if you do not have the username field on the form. If you do not want to use the email address for the user name you will need to provide the username field for your form.

- **Generate random password**

Select this option to generate a random password. This generates a random password if you do not provide a password field on the Action Form. If you do not want to generate a random password you will need to create a password field for your form.

- **Send standard DNN registration email**

Select this option to send the standard DNN registration email. If you do not want to send DNN registration email and would like a custom email template, you will need to set-up Email actions.

- **Login if user already exists**

Select this option if the user exists in the system. The user will be logged in if the user authentication matches.

Update User Profile

Action Form allows you to set up some actions so that the users can update their profile information. It is a very useful option when your site has an intranet section. Update User Profile options:

- **Allow password update**

For this option to work, you need to set-up a password field for you Action From. This option is useful when a user would like to update his password.

- **Also update Display Name with**

For this option to work, you need to set-up a First Name and Last Name field for you Action From. This option will give the user the ability to change his display name with one of the two options: First Name and Last Name.

A practical example would be to populate the profile property from a registration form when a new user is created and to do this you need to add on the button the Update Profile action which will match every field or data you have on your form with a profile property, this includes data that you load with SQL action, Inject Data actions or others, the match is done by title and ID.

Grant User Role

Action form allows you to assign additional roles to an user as well as an expiration date per each role. Grant User Role options:

- **Role**

This option allows you to grant a security role - all the user roles created in admin page is displayed in the drop down list from where you can select.

- **Other Role Names**

This option give you the possibility to input a role name and if you specify multiple role names, separate them by comma. It also accepts DNN tokens and My Tokens.

- **Role expiration (days)**

In this filed you can set a period to determine after how many days the role expires. It you leave it blank, it will never expire.

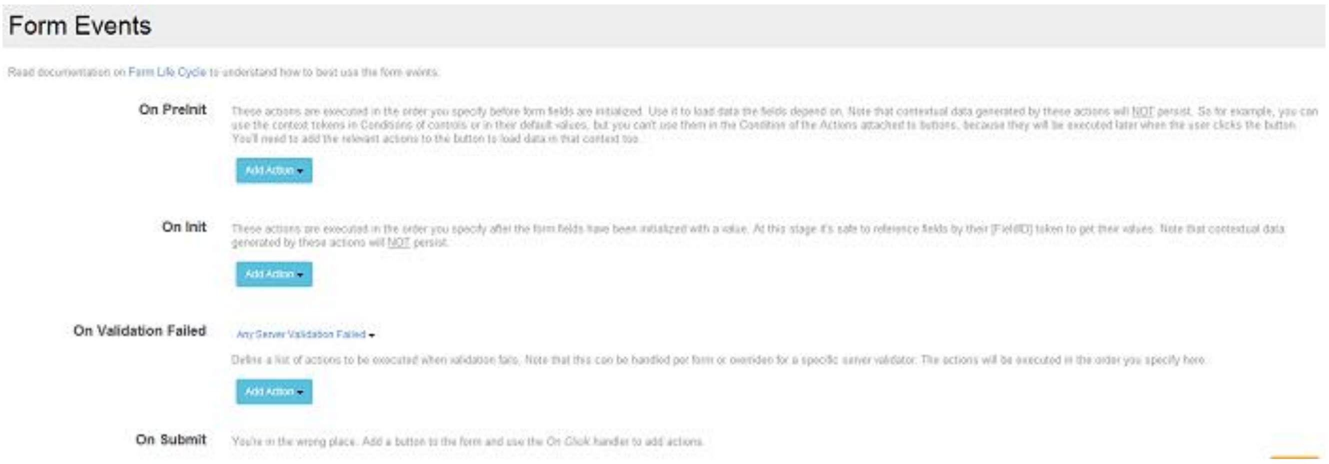
Revoke User Role

This action can be used to cancel a security role from a specific user. From the Role drop down list you have the possibility to select the role you want to revoke (this data is fetched from the Admin > Security Roles page).

Form Events

We created the Lifecycle section in this user guide with the purpose of explaining the process of actions added on the form fields and to provide some understanding regarding the life cycle of Action Form.

To sum up, there are three Form Events, On Preinit, On Init and On Validation Failed, besides them, as action event can also be considered the On Click Handlers from the buttons.



Form Events

Read documentation on Form Life Cycle to understand how to best use the form events.

On Preinit These actions are executed in the order you specify before form fields are initialized. Use it to load data the fields depend on. Note that contextual data generated by these actions will NOT persist. So for example, you can use the context tokens in Conditions of controls or in their default values, but you can't use them in the Condition of the Actions attached to buttons, because they will be executed later when the user clicks the button. You'll need to add the relevant actions to the button to load data in that context too.

On Init These actions are executed in the order you specify after the form fields have been initialized with a value. At this stage it's safe to reference fields by their [FieldID] token to get their values. Note that contextual data generated by these actions will NOT persist.

On Validation Failed Any Server Validation Failed

Define a list of actions to be executed when validation fails. Note that this can be handled per form or overriden for a specific server validator. The actions will be executed in the order you specify here.

On Submit You're in the wrong place. Add a button to the form and use the *On Click* handler to add actions.

When an action is set on Preinit Event, they are executed in the order you specify before form fields get initialized. Use this event to load data on which the fields depend on. Note that contextual data generated by these actions will NOT persist. So for example, you can use the context tokens in Conditions of controls or in their default values, but you can't use them in the Condition of the Actions attached to buttons, because they will be executed later when the user clicks the button. You'll need to add the relevant actions to the button to load data in that context too.

If there are actions set on Init Event, they get executed in the order you specify after the form fields have been initialized with a value. At this stage it's safe to reference fields by their [FieldID] token to get their values. Note that contextual data generated by these actions will NOT persist.

The On Validation Failed event defines a list of actions to be executed when validation fails. Note that this can be handled per form or overridden for a specific server validator. The actions will be executed in the order you specify here. Click here to see a tutorial on an older version of Action Form where we provide some samples of how this event works.

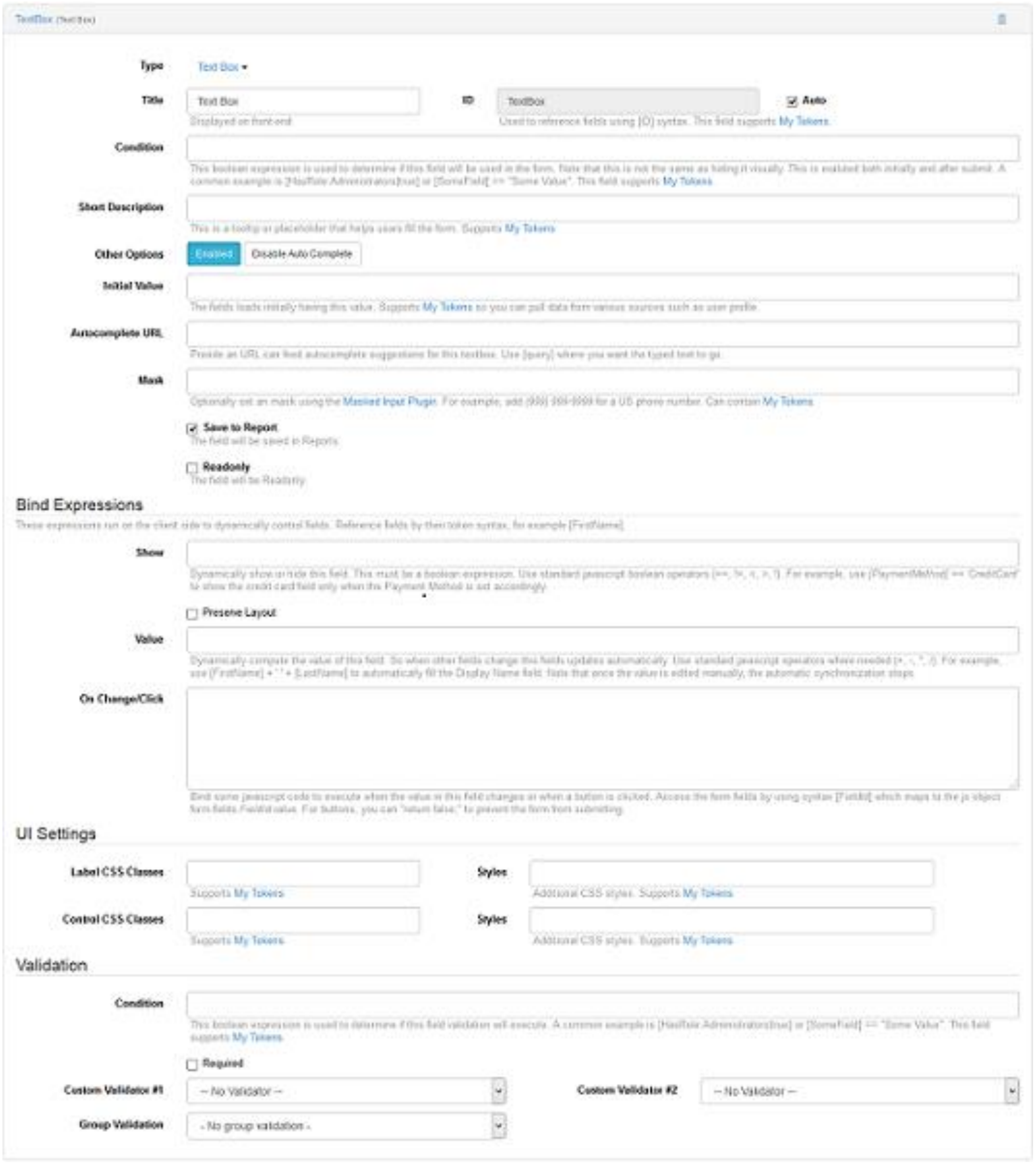
And regarding the last event, On Submit, this event is used by adding actions on the button fields, on the click handlers - for more details, read the Form Actions sub-pages where we've listed the actions which can be set on the buttons.

Form Fields

Form fields are used to actually create the form. You can add, remove, enable, disable and customize these form fields. You can also reorder them with drag&drop in the Layout Mode. To remove form fields, you have to select one or more fields, then click the Remove button at the right and then click on the Save button. You can expand the fields just by clicking on the label, this will open the settings panel where you can customize the UI.

The main configuration options are the Title and Type fields. Basically, the Title is the label of the the form. The Type field is a drop down list which specifies the input type required from the user, for example Text, Email, Check box, etc.

You can customize the form fields with the General, UI Settings and Validations options. The General options allow you to select the type of the field, name, assign values to the title, etc. The UI Settings option allows you to apply CSS styles and the Validation option allow you to set the filed as required and to put specific validations on it. See the screenshot below (this is an example of Text Box type of field). For additional information about fields, please check each field page.



TextBox (Text Box)

Type: Text Box ▾

Title: Text Box — Displayed on front-end

ID: TextBox ☑ Auto — Used to reference fields using [ID] syntax. This field supports My Tokens.

Condition: — This boolean expression is used to determine if this field will be used in the form. Note that this is not the same as hiding it visually. This is evaluated both initially and after submit. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens.

Short Description: — This is a tooltip or placeholder that helps users fill the form. Supports My Tokens.

Other Options: Enabled | Disable Auto Complete

Initial Value: — The fields loads initially having this value. Supports My Tokens so you can pull data from various sources such as user profile.

Autocomplete URL: — Provide an URL can feed autocomplete suggestions for this textbox. Use [query] where you want the typed text to go.

Mask: — Optionally set an mask using the Masked Input Plugin. For example, add (999) 999-9999 for a US phone number. Can contain My Tokens.

☑ Save to Report — The field will be saved in Reports.

☐ Readonly — The field will be Readonly.

Bind Expressions

These expressions run on the client side to dynamically control fields. Reference fields by their token syntax, for example [FirstName].

Show: — Dynamically show or hide this field. This must be a boolean expression. Use standard javascript boolean operators (==, !=, <, >, !). For example, use [PaymentMethod] == CreditCard to show the credit card field only when the Payment Method is set accordingly.

☐ Preserve Layout

Value: — Dynamically compute the value of this field. So when other fields change this fields updates automatically. Use standard javascript operators where needed (+, -, *, /). For example, use [FirstName] + ' ' + [LastName] to automatically fill the Display Name field. Note that once the value is edited manually, the automatic synchronization stops.

On Change/Click: — Bind some javascript code to execute when the value in this field changes or when a button is clicked. Access the form fields by using syntax [FieldId] which maps to the js object form.fields.FieldId.value. For buttons, you can "return false;" to prevent the form from submitting.

UI Settings

Label CSS Classes: — Supports My Tokens | Styles: — Additional CSS styles. Supports My Tokens

Control CSS Classes: — Supports My Tokens | Styles: — Additional CSS styles. Supports My Tokens

Validation

Condition: — This boolean expression is used to determine if this field validation will execute. A common example is [HasRole:Administrators|true] or [SomeField] == "Some Value". This field supports My Tokens.

☐ Required

Custom Validator #1: -- No Validator -- | Custom Validator #2: -- No Validator --

Group Validation: - No group validation -

Most settings in the screenshot above are found in all field types. You will find described the specific settings, on each field page.

General Settings

- **Form field Type**

Action Form supports various field types, from simple text boxes to complex controls such as date pickers. The type determines how users interact with the form to submit data. Field types are specified in the Type drop down box of the Fields section and also in the drop down list on the Add Field button. There are several field type categories like: Address, Buttons, Date & Time, DNN, Files, Hidden Data, Image, Multiple choice, Security, Statistics, Text, User. A description of each category is specified in the list below.

Address
Buttons
Date & Time
DNN (DotNetNuke)
Files
Hidden Data
Multiple choice
Security
Image
Statistics
Text
User

- **Field Title**

The Title field is for specifying the name for the field - it's displayed on front end as a label for the field. Example: First Name, Email, Message, etc.

- **Field ID**

Action Form gives you the ability to automate the field ID using the Auto check box. It is responsible for storing the Title name. You can assign a new ID for the Title or keep the assigned ID in the field. The assigned ID is auto-generated based on the text entered for the Title. This option supports My Tokens.

- **Field Short Description**

This option allows you to enter a short description for the Title field. It is a tooltip or placeholder that will help users to fill in the form. For example, for a Name field, a short description could be: "Please enter your Name." This option supports My Tokens.

- **Other Options**

This section just gives you the ability to enable/disable the Field and the Auto Complete.

- **Initial Value**

This option allows you to insert Tokens. The token inserted here, is applied to the Title field. For example, [User:LastName] token is entered on the Field Initial Value to populate the user's last name in the text field on the Action Form. This option also supports My Tokens.

- **Autocomplete URL**

This option can feed auto-completed suggestions for this text box, here you can use {query} where you want the typed text to go.

- **Mask**

Optionally you can set a mask on a field, for example, if you want a certain type of Phone field, you can put a mask like (999) 999-9999 or a certain date type like 99/99/9999, or a time field 99:99:99, percentage 9,99%, amount 999.999,99$. Please keep in mind that on mobile devices, the fields which have mask formatting are not recognized.

UI Settings

The UI Settings allow you to insert and apply CSS Styles and Classes to individual or multiple text fields. Action Form gives you the ability to customize the form exactly the way you want so it perfectly suites you website design. Action Form allows you to use the Label CSS Classes and Control CSS Classes properties to apply CSS styles to the forms. CSS classes are predefined classes created by DNN and should reference the relevant css file.

Label CSS Classes - allows you to apply multiple CSS classes to labels on the form.

Control CSS Classes - allows you to apply multiple CSS classes to controls.

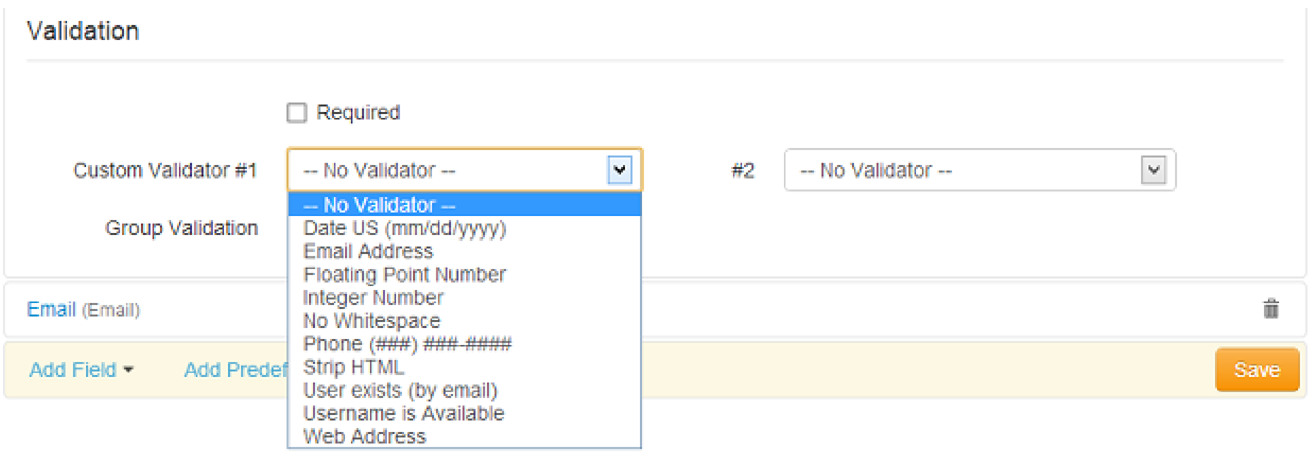
Label CSS Styles property allows you to input and apply CSS Styles to labels.

Validation

The Validations properties includes the Required check box option, two Custom Validation boxes and the Group Validation drop down menu. They allow you to set validations on the fields.

Required check box option.
It allows you to set a field as required in order for the user to be informed that it's a mandatory field and the form will not be submitted without info filled in that specific field. If the user does not complete the field, the form returns the standard "required" error message.

Custom Validators #1 & #2
Custom Validation #1 and #2 specifies formats for validating text fields on the action form. You can set up to two validations. If you want additional validators you can add them to the /DesktopModules/AvatarSoft/ActionForm/Validators folder.


Validation

☐ Required

Custom Validator #1: -- No Validator --   #2: -- No Validator --

Group Validation

-- No Validator --
Date US (mm/dd/yyyy)
Email Address
Floating Point Number
Integer Number
No Whitespace
Phone (###) ###-####
Strip HTML
User exists (by email)
Username is Available
Web Address

Email (Email)

Add Field ▾   Add Predef   Save

- Date US format allows you to set the field to accept inserting the date in the US format mm/dd/yyyy.
- Email Address format allows you to set the field to accept only email addresses.
- Floating Point Number format allows you to set the field to accept only decimal numbers.

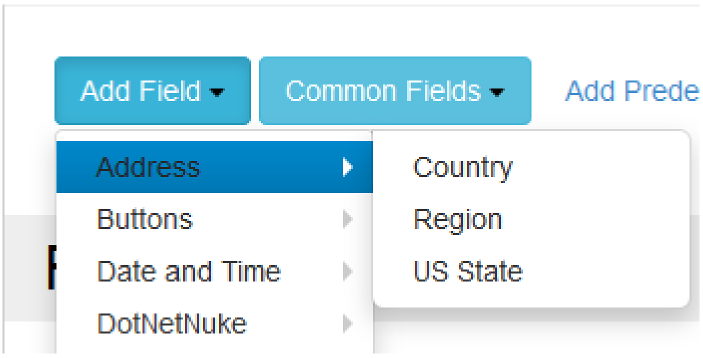
- Integer Number format allows you to set the field to accept only integers.
- No Whitespace format allows you to set the field to not accept white spaces.
- Phone format allows you to set the field to accept phone numbers, using the (###) ###-#### format.
- Strip HTML format allows you to set the field to accept HTML. This will strip the HTML tags and submit only the text.
- User exists validator allows you to display a message to inform the user who is trying to register that the email address used is already in the database.
- Username is Available validator allows you to display a message when the username is available.
- Web Address format allows you to set the field to accept only web addresses.

Group Validation

From the drop down list you can chose one of the group validations. Click here to see a tutorial about group validation on Action Form.

Address

The Address field helps you capture address information on a form and contains three options, Country, Region, US State which allow the users to select the needed values which are populated in these lists. Keep in mind that Regions is strictly dependent on the Country field, so if you want the Region list to be populated with the correct values, you have to link the Country Field option to the previously defined Country.



The first field is a drop down for Country selection, the second field Region can be a drop down from where you can select the Region (if in the Country drop down list is set the Country field previously created) or it can be a text box where you can manually specify the region.

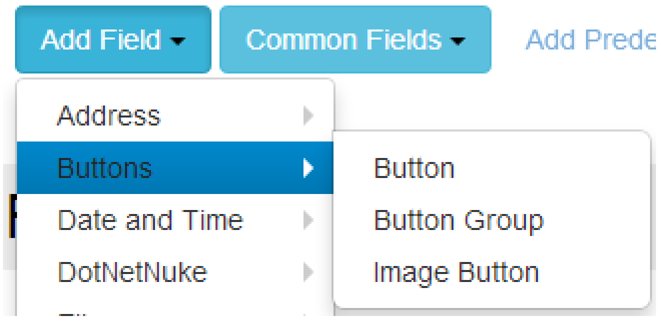
Region

In the Region field there is a specific label, named Country Field, this label links the region to the country. Therefore, when a country is selected in the Region field, there will appear a drop down list with all available regions for the selected Country. For example, if you select Canada as country, the region will display a drop down list which contains all the states from Canada. If you select France as country, the region will display as a text box because an this country there are no regions defined.

Buttons

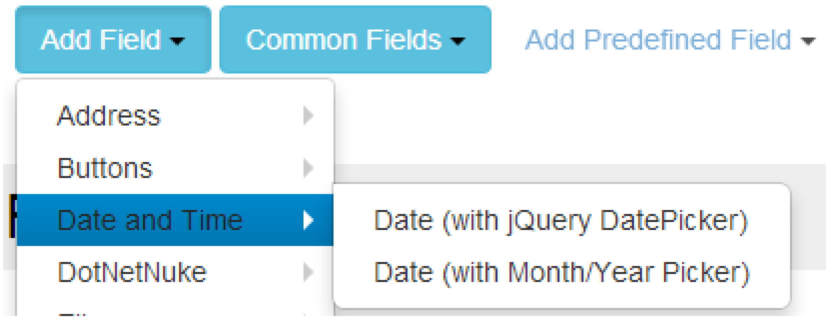
Button filed has three choices: simple Button, Button Group and Image button. The Button field differs somehow from the other fields due to the actions which can be set on the button by selecting them from the On Click Handler list. More about these actions can be found on the From Actions sub-pages. As additional and different setting options on the button fields, you have the size, type and align drop down lists from where you can customize the display of the button (e.g. a small button which can be displayed on the right side of the page with Danger icon on it). Like the other types of fields, the Button fields, have the UI Settings section where you can customize the CSS classes and the Bind Expressions section where you can set them to run on the client side to dynamically control the fields.

The Button Group field can be used when you have multiple buttons and need to align them together in order to have a nice effect on the front end, and the Image Button field is used along with the Image URL box where you can place the image link address - this option is usually used with a redirect action.



Date & Time

The Date field has two available options, the jQuery date picker and the simple standard Month/Year picker. You can customize the date picker using the jQuery Theme option from the General Settings section or by styling the CSS classes from the UI Settings section.



Optionally, you can provide a JSON object with other options to pass into the jQuery Date Picker control.

For example:

1. To disable the weekends from the date picker, write in the Other Options for jQuery UI Datepicker (JSON) field:

```
{
 beforeShowDay: $.datepicker.noWeekends
}
```

2. To enable days beginning with the system date:

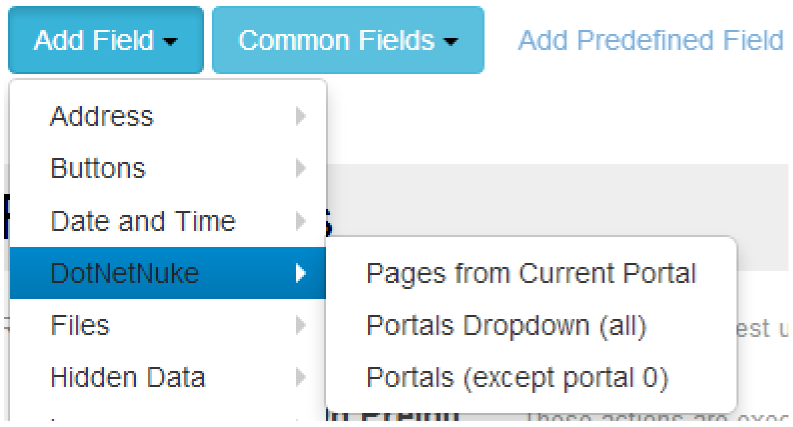
```
{
minDate: 0
}
```

3. To enable days between the system date and a certain date (e.g. until the 25th of August 2014):

```
{
minDate: -0,
maxDate: '08/25/2014'
}
```

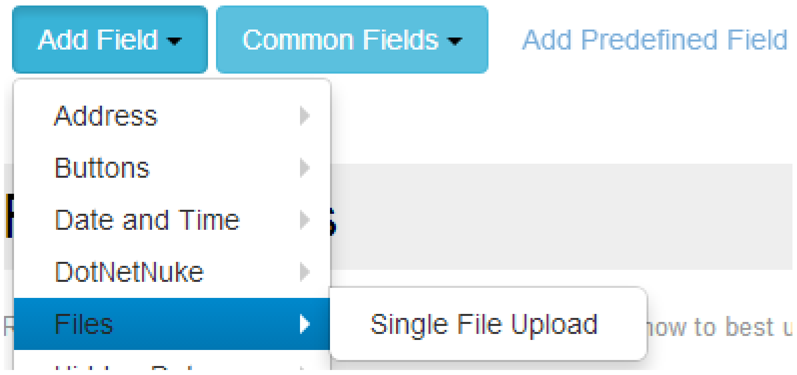
DNN

The DNN field is usually used for its three options with the purpose of loading pages from the DNN Portals - there are three options to use: Pages from Current Portal, Portals Dropdown (all) and Portals (except portal 0) and as additional and recommended field there is the Initially Checked option, used to determine which item is initially selected. It also supports My Tokens so you can pull data from various sources such as user profile.



Files

Using the Single File Upload field on a form gives you the possibility to create tokens on submit action and to stream the files into database. Here you can define the folder where the user will upload the file by setting it from the Upload to Folder drop down list, and you can also use the Handle Duplicates option for the cases where the users upload the same file more than once.



Hidden Data

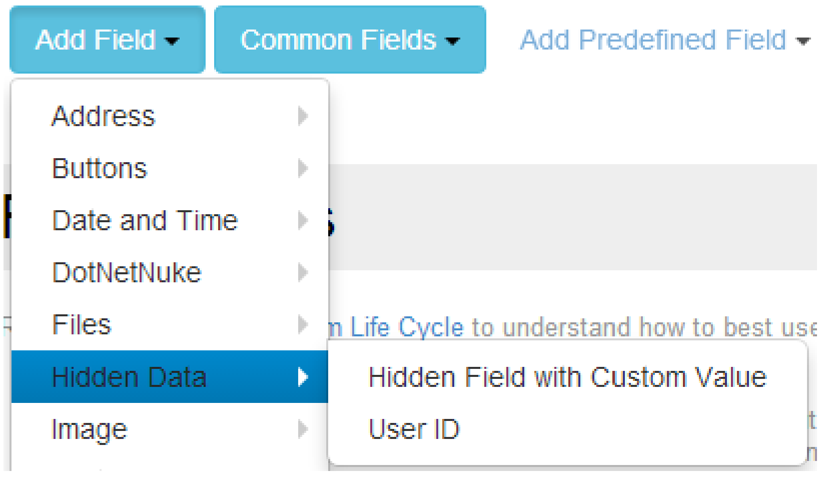
The two Hidden Data field options - the Hidden Field with Custom Value and the User ID fields - the hidden fields do not show on the page, therefore the visitor can't type anything into a hidden field, which leads to the purpose of the field, to submit information that is not entered by the visitor.

Hidden fields allow you to store "hidden" information within a form. These fields are not displayed by the client. However, if the user selects the "View Source" option in the browser, the entire form is visible, including the hidden fields. Hidden fields are therefore meant just for passing information to and from forms transparently.

This is an example of how the two hidden fields are stored and displayed in the source of the page:

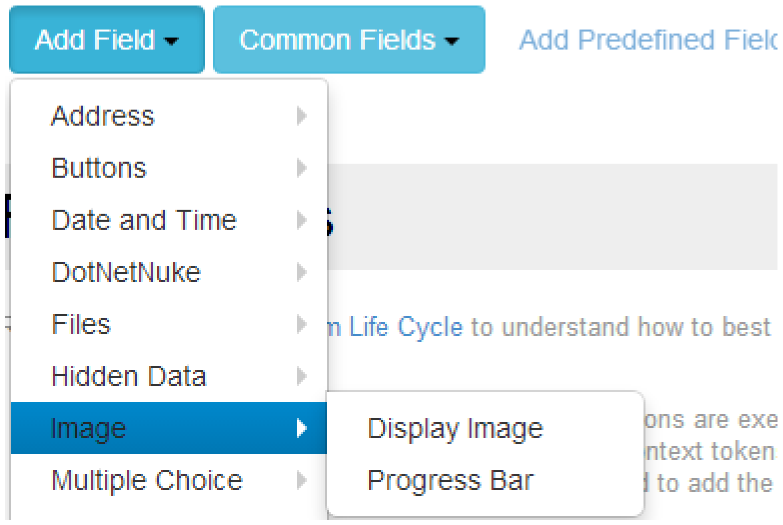
<div class="form-group"><input type="hidden" name="dnn1420HiddenFieldwithCustomValue" data-fieldid="420" data-af-field="HiddenFieldwithCustomValue" value="Image" data-ng-model="form.fields.HiddenFieldwithCustomValue.value" data-val="{{ form.fields.HiddenFieldwithCustomValue.value }}">

</div><div class="form-group"><input type="hidden" name="dnn1420UserID" data-fieldid="421" data-af-field="UserID" value="3" data-ng-model="form.fields.UserID.value" data-val="{{ form.fields.UserID.value }}"></div><div class="form-group"><div class=" col-sm-12"><img src="/example.jpg" class="">



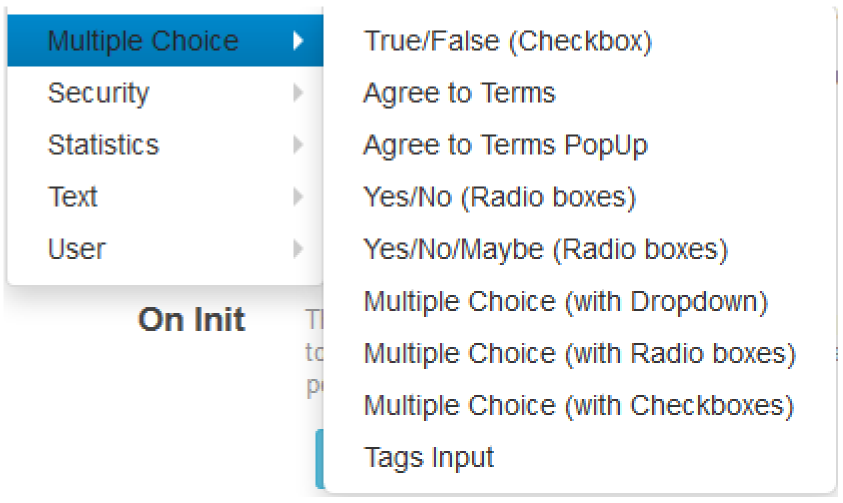
Image

There are two image options available on Action Form, the Display Image field option and the Progress Bar option which will be displayed after the user clicks on a submit button. For the Display Image option, you can select an already uploaded image from the DNN portal by selecting it from the Image drop down list, or you can provide an image URL in the Image URL box. The Progress Bar field is used along with the Image option where you can define the loading animation which will display on front at submit action.



Multiple choice

With this option you can create different choice fields like check boxes, radio boxes, drop down lists etc.



Multiple Choice (with Drop Down)

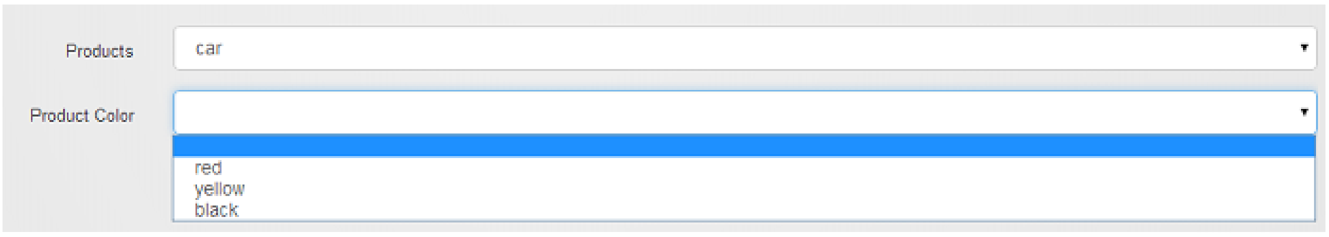
The Multiple Choice with Drop Down option is used to create a list of items for a user to choose from. The Drop Down field allows the users to select an option from a given list.

Let's create an example where you create a drop down list called Products, in this list you have to add some Items, use the Items box and list several items, each item to be added per line (e.g. car, laptop etc). If you want, let's say, to add different colors/sizes for each item, you can create another drop down list which has to be linked to the Products drop down and when the user selects a product from the list, the second drop down calls only the colors available for that specific product only.

Here's how you can do it: in the second drop down list set on "Link to" option the "Products" field, then, in "Items" box there's a certain syntax which has to be correctly used to make the form work. You have to prefix the item with the name of the parent (in our case: "car") and separate the parent from the child using slash ("/"), for example if you want to display red, yellow and black for the product car, in the Items box you have to set:

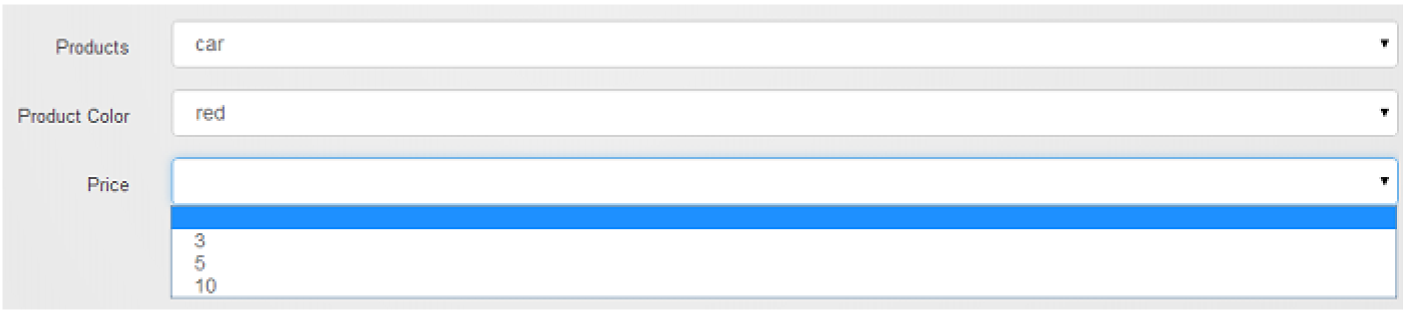
car/yellow
car/red
car/black

and here's how it will be displayed on front when the user selects car from the first drop down list:



And if you want to add a subcategory for the second drop down list (Product Color), then, on the third drop down list, let's name it Price, you have to link it to the second drop down - Product Color and in the Items box you have to use the syntax as follows:

car/red/3
car/red/5
car/red/10



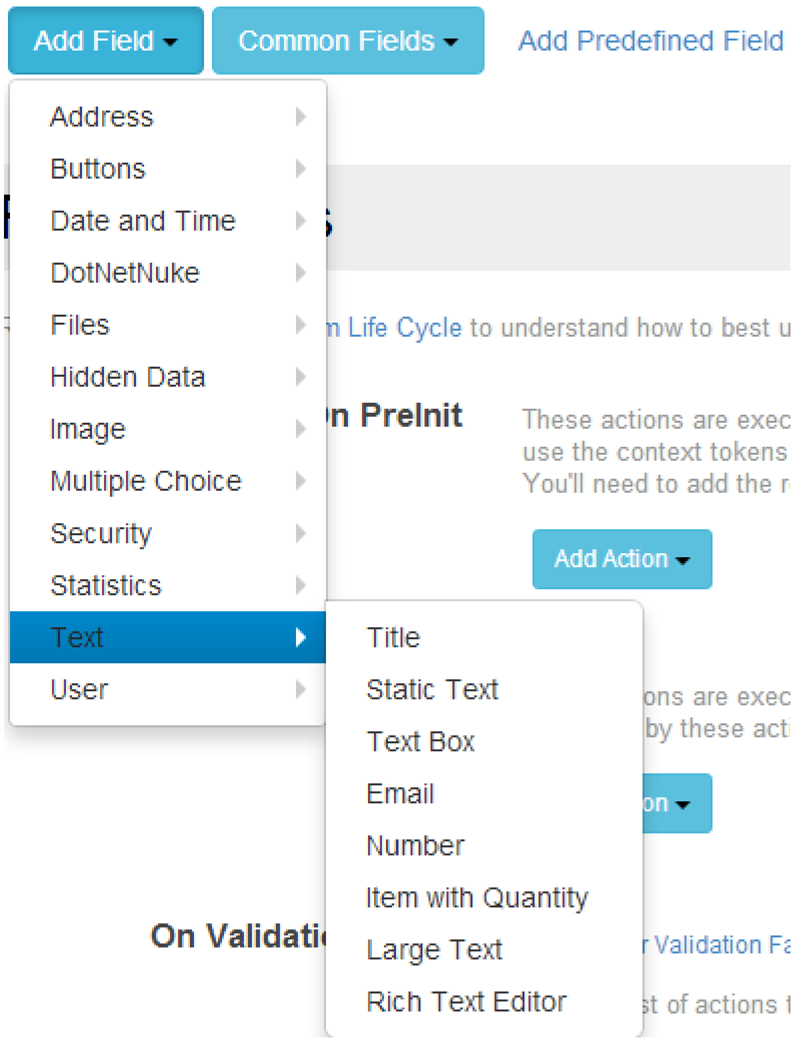
Another example of a drop down option would be to populate a drop down with a list of users from DNN and use the ID as the value displayed. Here's how you can do it:

1. put this query in the drop down Items field: select Displayname, UserId from Users (Note how the first column is the text, and the second column is the value)

2. or use this query in the items field: select Firstname+' '+Lastname, UserID from Users,

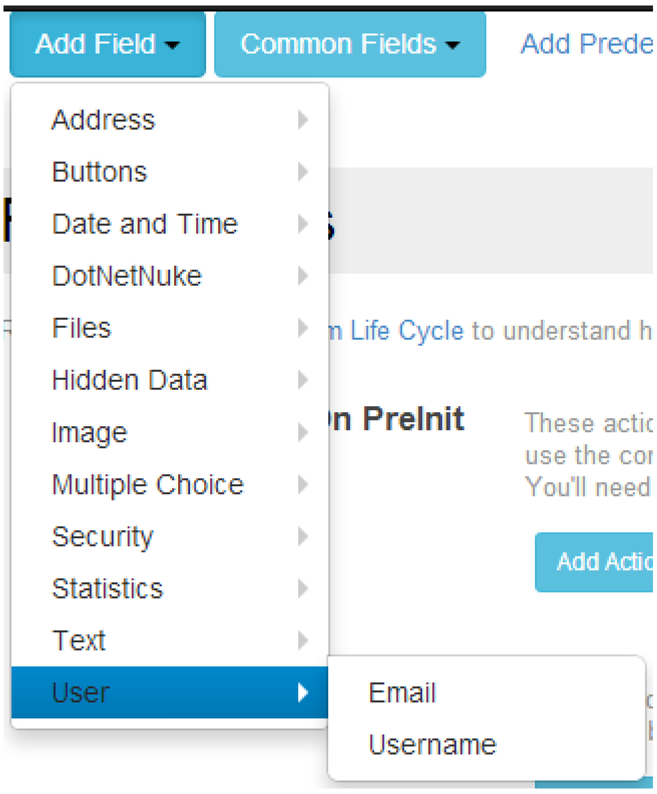
Text

The most used field in creating forms, Text field has several options useful options like: Title, Static Text, Text Box, Email, Number, Item with Quantity, Large Text, Rich Text Editor. A note to make here is on the Email field where only valid emails can be entered, which means that the correct email syntax is accepted (both an '@' and '.' character are required), to make sure the email address will be a correct one you can also apply an email validation on this field.



User

The last field is the User one where there are two available options, the Email and the Username, they are usually used in a registration form along with a password field. You can also use validations on these fields in order to improve the accuracy of the data filled by the users.


Add Field
Common Fields
Add Prede
Address
Buttons
Date and Time
DotNetNuke
Files
Hidden Data
Image
Multiple Choice
Security
Statistics
Text
User
Email
Username
n Life Cycle to understand h
n PreInit
These actic
use the cor
You'll need
Add Actio

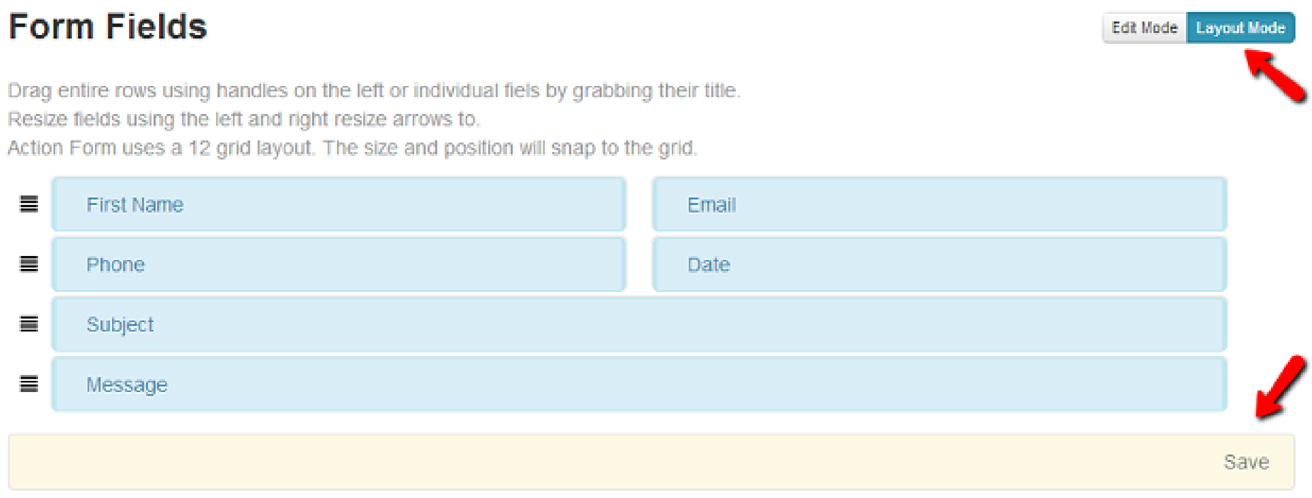
Form Layout

We have been working really hard to make it as easy as possible also packed with cool features, that's why in Action Form you have the ability to change the layout of your forms in two ways:

- Drag&Drop Layout (Layout Mode) - which is simple, easy and fast.
- Manual Layout - gives you the possibility to create some complex forms.

Layout Mode

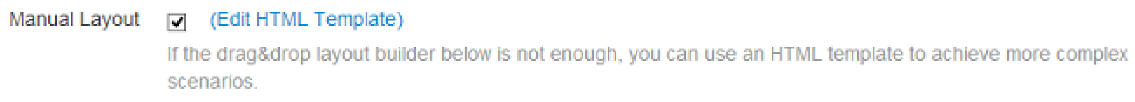
Selecting and editing in Layout Mode is the simplest and fastest way for you to achieve high complex forms with multiple fields. Using drag&drop you can move the fields up&down, left&right so it matches your needs.


Form Fields

Drag entire rows using handles on the left or individual fiels by grabbing their title.
Resize fields using the left and right resize arrows to.
Action Form uses a 12 grid layout. The size and position will snap to the grid.

Manual Layout

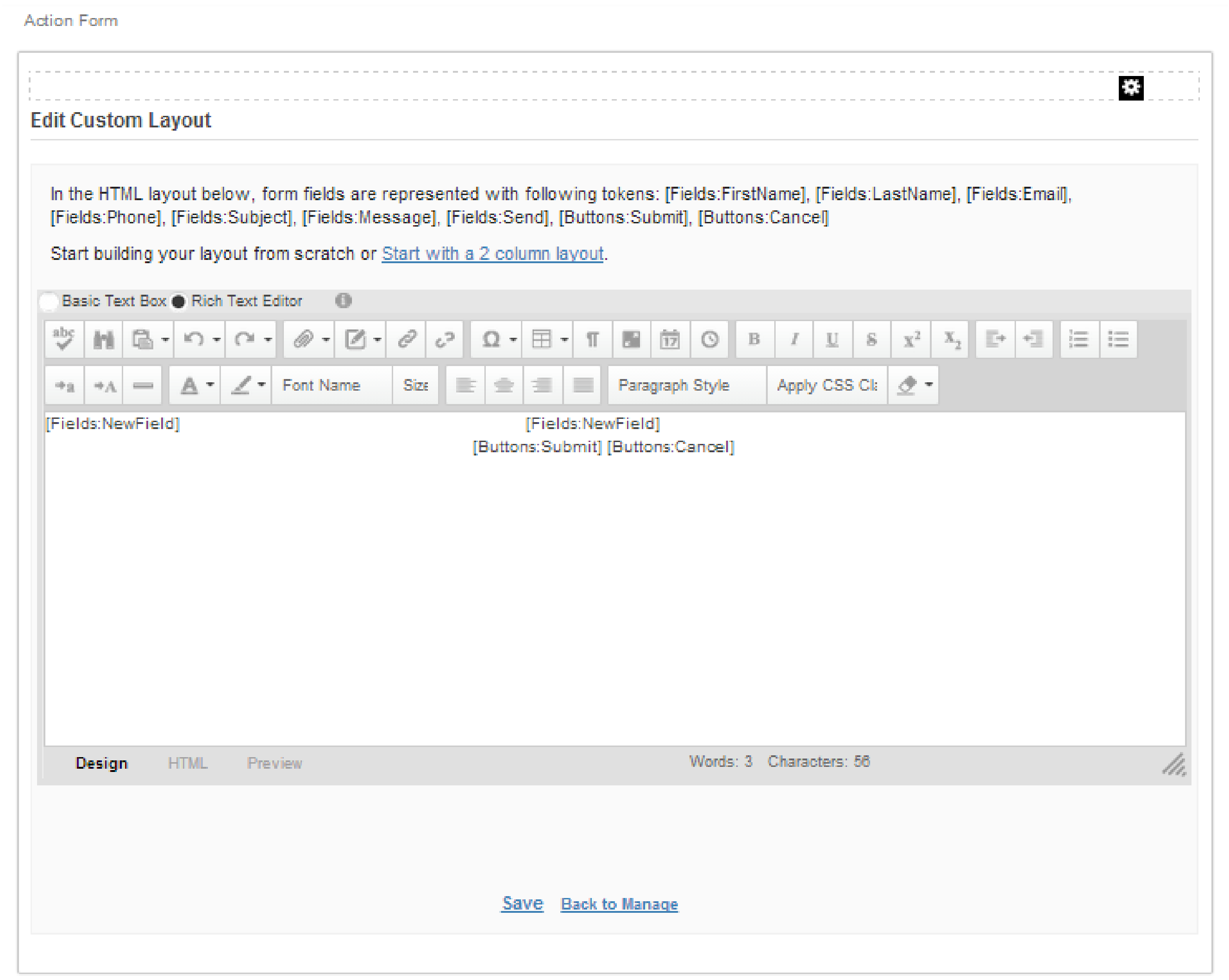
If the drag&drop layout builder is not enough, you can use an HTML template to achieve more complex scenarios. If you are working with complex forms such as insurance applications, medical claims, government transactions, this option give you the ability to create the exact form that suites you.


Manual Layout ☑ (Edit HTML Template)
If the drag&drop layout builder below is not enough, you can use an HTML template to achieve more complex scenarios.

To edit the HTML Template, check mark the Manual Layout option and then click on Edit HTML Template.

In the HTML layout, form fields are represented with the following tokens: [Fields:FirstName], [Fields:LastName], [Fields:Email], [Fields:Phone], [Fields:Subject], [Fields:Message], [Fields:Send], [Buttons:Submit], [Buttons:Cancel]. Action Form has integrated My Tokens so you can have more options available to chose from.

You can start building a new layout from scratch or start with a predefined 2 column layout. After you are done, click the Save button and your form is ready.


Action Form

Edit Custom Layout

In the HTML layout below, form fields are represented with following tokens: [Fields:FirstName], [Fields:LastName], [Fields:Email], [Fields:Phone], [Fields:Subject], [Fields:Message], [Fields:Send], [Buttons:Submit], [Buttons:Cancel]

Start building your layout from scratch or Start with a 2 column layout.

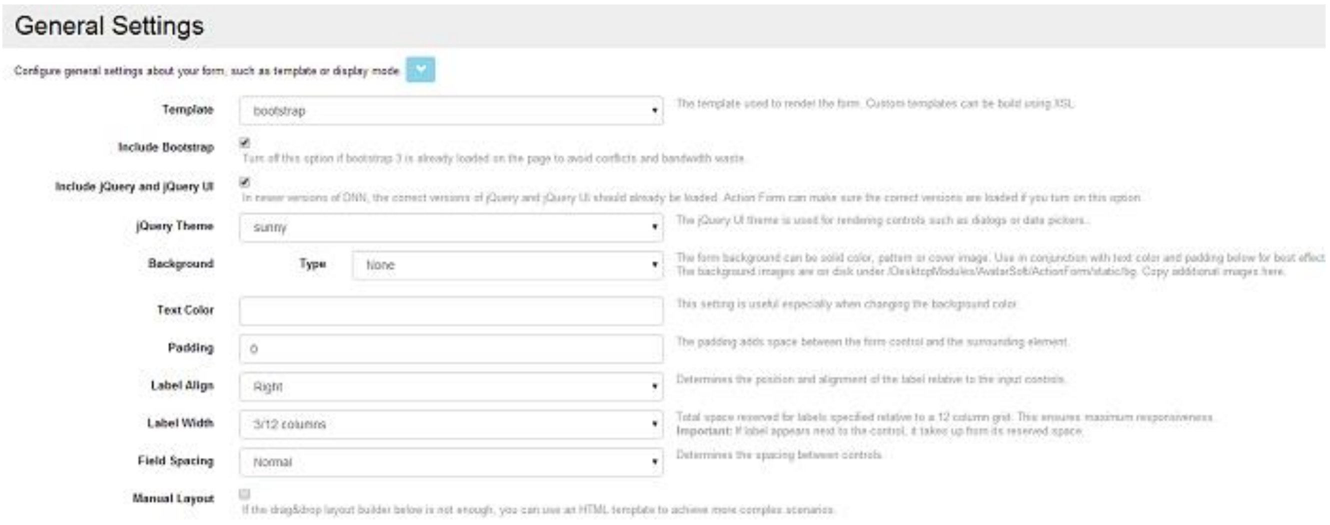
Basic Text Box [✓ Rich Text Editor]

```
[Fields:NewField]                                   [Fields:NewField]
                                          [Buttons:Submit] [Buttons:Cancel]
```

Words: 3 Characters: 56

Form Settings

The Action Form settings, highlighted on the image below, allows you to configure the layout and appearance of the form. These settings are managed in the General Settings section. Our Action Form module is highly configurable so it can be set on every website with different designs. More info on how to change the look & feel of forms, you will find under Appearance and Form Layout sections.



General Settings

Configure general settings about your form, such as template or display mode

Template	bootstrap	The template used to render the form. Custom templates can be build using XSL.
Include Bootstrap	✓	Turn off this option if bootstrap 3 is already loaded on the page to avoid conflicts and bandwidth waste.
Include jQuery and jQuery UI	✓	In newer versions of DNN, the correct versions of jQuery and jQuery UI should already be loaded. Action Form can make sure the correct versions are loaded if you turn on this option.
jQuery Theme	sunny	The jQuery UI theme is used for rendering controls such as dialogs or data pickers..
Background	Type: None	The form background can be solid color, pattern or cover image. Use in conjunction with text color and padding below for best effect. The background images are on disk under /DesktopModules/AvatarSoft/ActionForm/static/bg. Copy additional images here.
Text Color		This setting is useful especially when changing the background color.
Padding	0	The padding adds space between the form control and the surrounding element.
Label Align	Right	Determines the position and alignment of the label relative to the input controls.
Label Width	3/12 columns	Total space reserved for labels specified relative to a 12 column grid. This ensures maximum responsiveness. **Important:** If label appears next to the control, it takes up from its reserved space.
Field Spacing	Normal	Determines the spacing between controls.
Manual Layout	☐	If the drag&drop layout builder below is not enough, you can use an HTML template to achieve more complex scenarios.

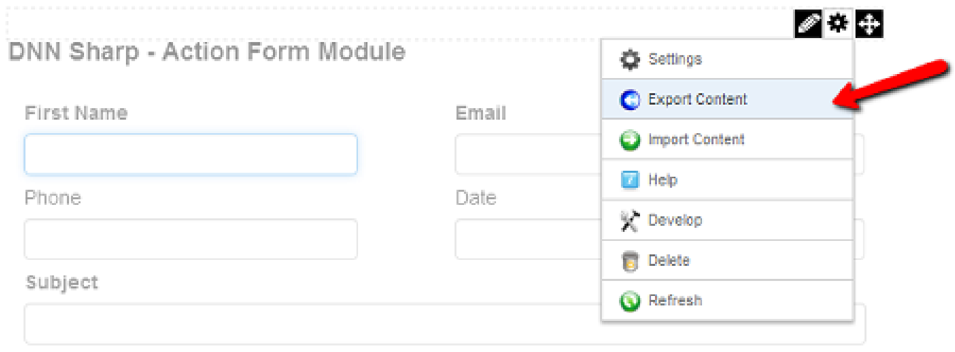
Import / Export Settings

The Action Form settings are saved in the database so you have the ability to manual change them. You can find these settings following this path: \Your-Database\Tables\dbo.avtActionForm_FormSettings. After you make some changes, you will need to Clear Cache before these changes are visible on the front-end.
Note that we are using the default export/import option in DNN 7 version.

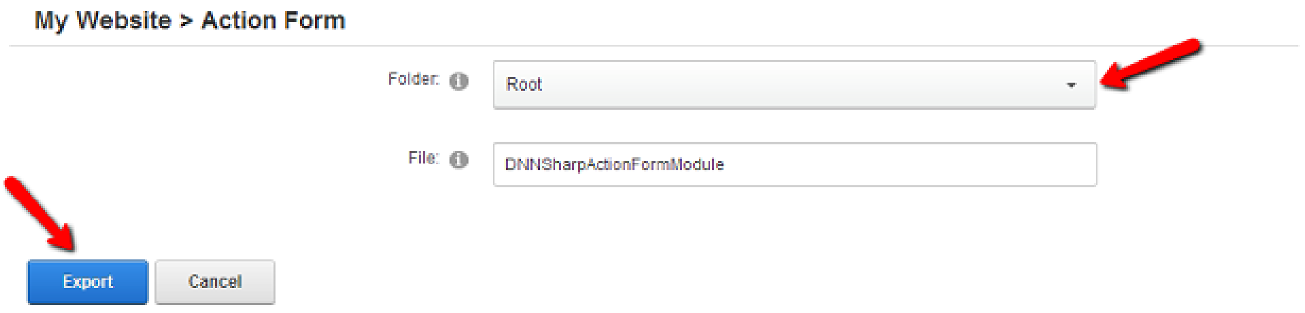
Save / Export Settings

If you have a big form that has various fields, radio buttons, drop-downs, date pickers etc. and wondering which is the best strategy to save these settings and the content to an external file that can be loaded at a later time or to another website, this is the way:

1. Move your mouse over the Module Settings button and then click on Export Content. Note that you have to be on Edit Mode to see all of these settings. A pop-up window will open. This option can also be used as a backup tool.



2. Select the Root folder where the .XML file with your settings/content will be saved. You can also change the File Name if you have more forms to backup; in this way backups can be differentiated.



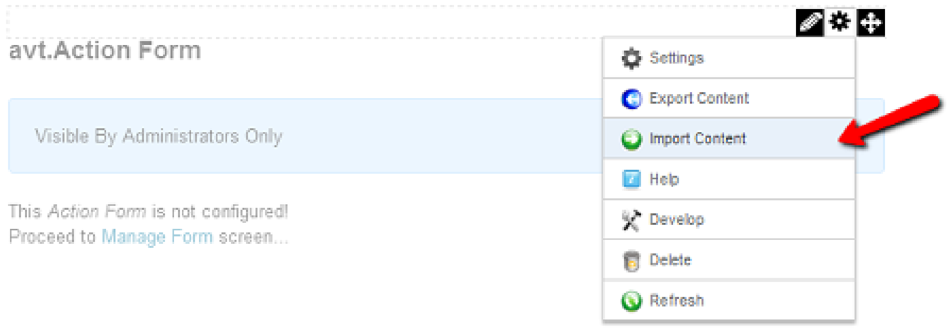
3. Finally, hit the Export button. That's all, your settings and content are saved.

Import Settings

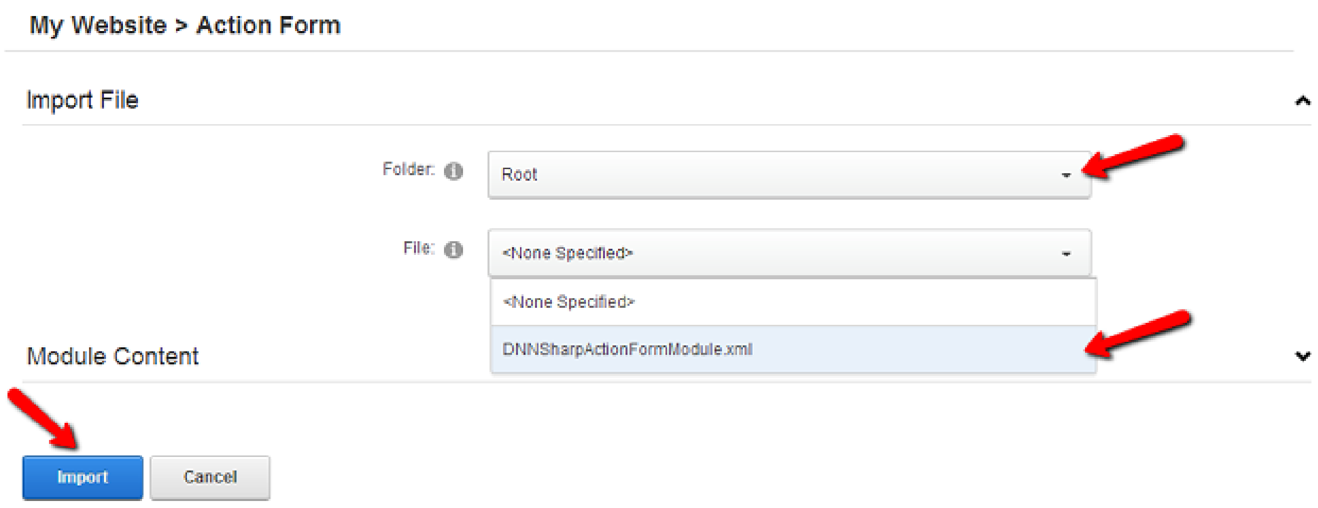
Importing settings can be done almost like when exporting settings with just a few differences. This option gives you the ability to move a form to another page, where you can modify it or to another website. It is a time saver option.

Importing on the same website:

1. Move your mouse over the Module Settings button and then click on Import Content. A pop-up window will open.


avt.Action Form
Visible By Administrators Only
This Action Form is not configured!
Proceed to Manage Form screen...
Settings
Export Content
Import Content
Help
Develop
Delete
Refresh

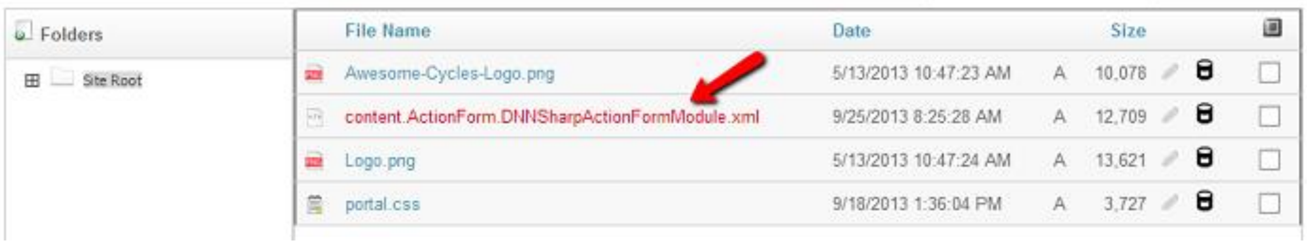
2. Select the Root folder where the .XML file with your settings/content was saved.



3. Next, hit the Import button. After this, you will have a new form with the exact settings saved before.

Importing on another website:

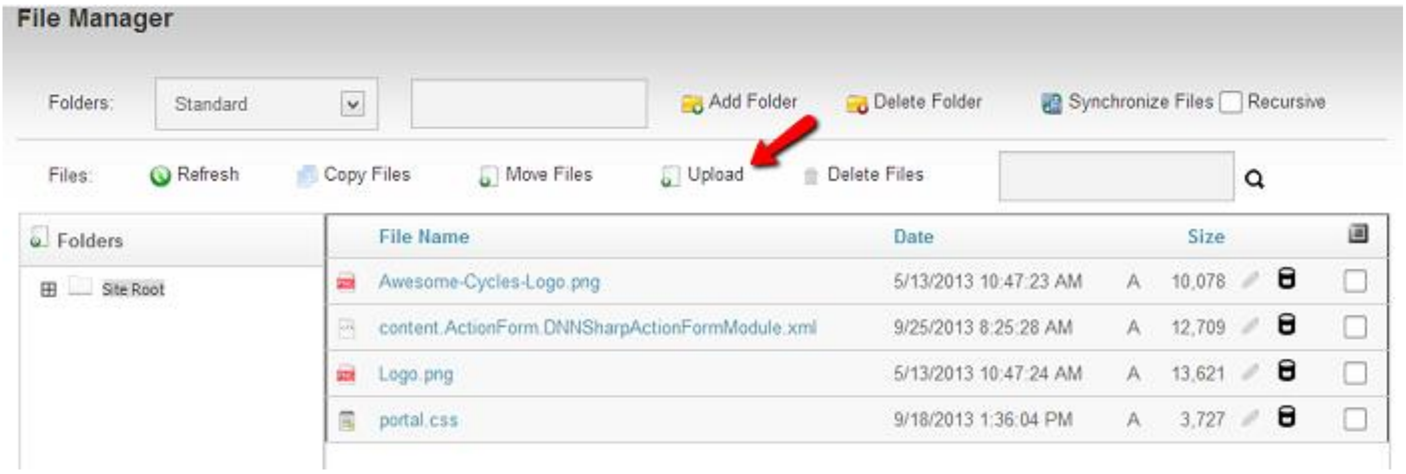
Before applying the saved settings to another website, you need to download the Action Form XML file to your computer and the upload it to the other location.
You can download the XML file just by clicking on it, like in the image below.



File Name	Date		Size
Awesome-Cycles-Logo.png	5/13/2013 10:47:23 AM	A	10,078
content.ActionForm.DNNSharpActionFormModule.xml	9/25/2013 8:25:28 AM	A	12,709
Logo.png	5/13/2013 10:47:24 AM	A	13,621
portal.css	9/18/2013 1:36:04 PM	A	3,727

After downloading the XML file, follow the steps below to upload it to another website.

1. Go to Admin -> File Manager and click on Upload button. You will be redirected to the upload page.



File Manager

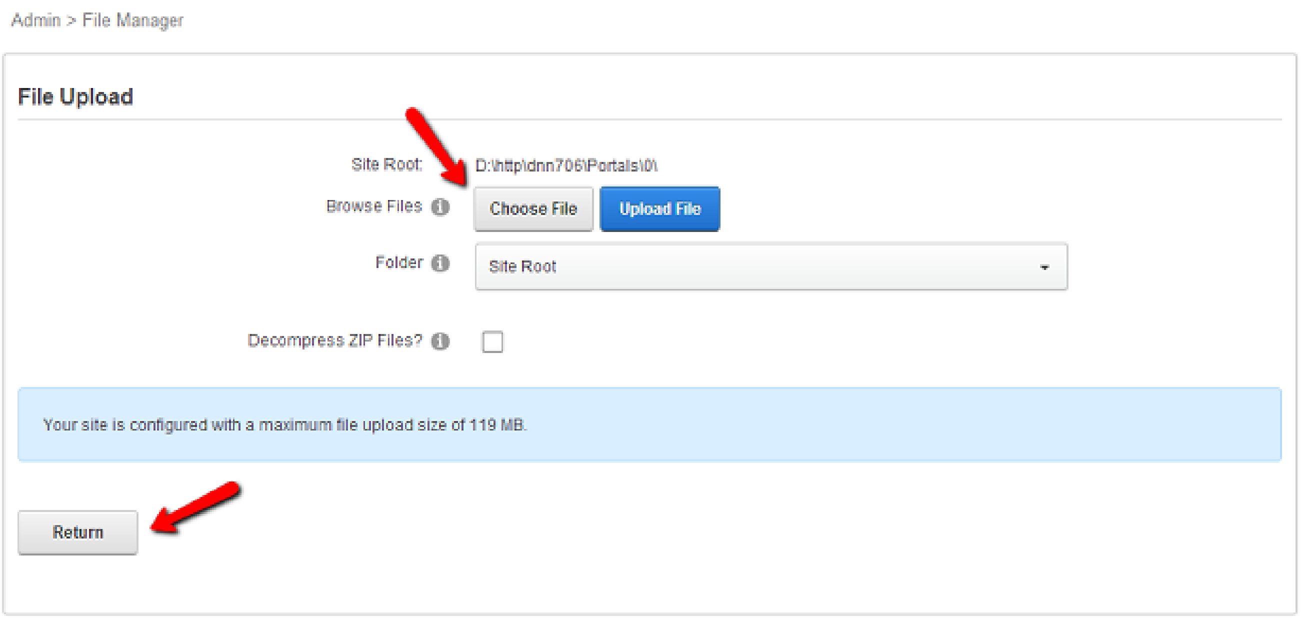
Folders: Standard | Add Folder | Delete Folder | Synchronize Files | Recursive

Files: Refresh | Copy Files | Move Files | Upload | Delete Files

Folders: Site Root

File Name	Date		Size
Awesome-Cycles-Logo.png	5/13/2013 10:47:23 AM	A	10,078
content.ActionForm.DNNSharpActionFormModule.xml	9/25/2013 8:25:28 AM	A	12,709
Logo.png	5/13/2013 10:47:24 AM	A	13,621
portal.css	9/18/2013 1:36:04 PM	A	3,727

2. Click on Choose button -> Select the file you want to upload and click on Upload File button. After this click on Return button so that you go back to the File Manager section.



Admin > File Manager

File Upload

Site Root: D:\http\dnn706\Portals\0\

Browse Files: Choose File | Upload File

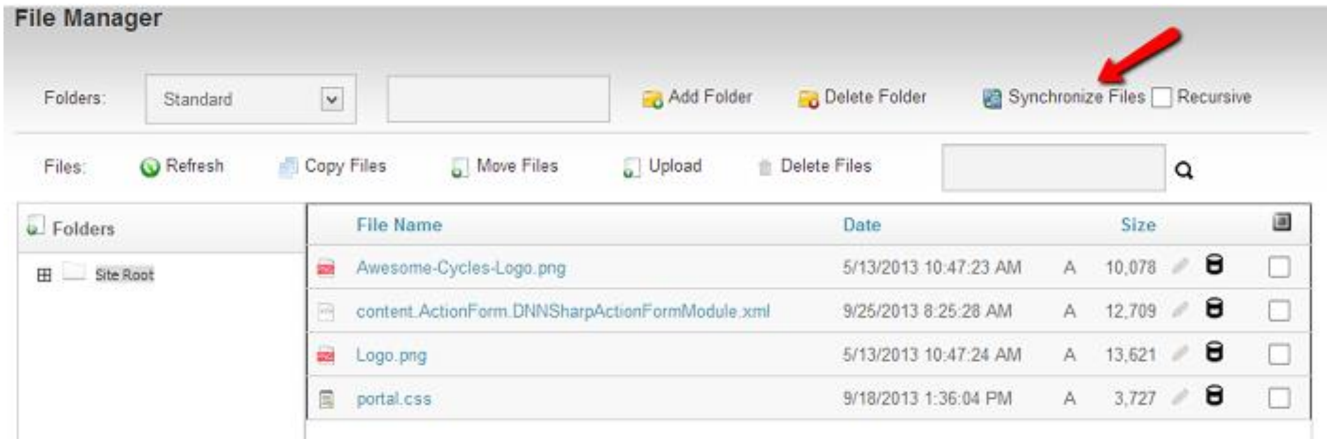
Folder: Site Root

Decompress ZIP Files? ☐

Your site is configured with a maximum file upload size of 119 MB.

Return

3. Now you will need to click on the Synchronize Files button so that the uploaded file to appear in the Import Content drop-down list.


File Manager

Folders: Standard | Add Folder | Delete Folder | Synchronize Files | Recursive

Files: Refresh | Copy Files | Move Files | Upload | Delete Files

Folders	File Name	Date		Size
Site Root	Awesome-Cycles-Logo.png	5/13/2013 10:47:23 AM	A	10,078
	content.ActionForm.DNNSharpActionFormModule.xml	9/25/2013 8:25:28 AM	A	12,709
	Logo.png	5/13/2013 10:47:24 AM	A	13,621
	portal.css	9/18/2013 1:36:04 PM	A	3,727

4. Next follow the steps from the above section, Importing on the same website.

Initialization Scripts

Action Form comes with tons of features, but sooner or later you'll need to do something directly to the controls. For example, maybe you want to have a jQuery plugin run on a textbox or you want to disable some fields. This is very easy to achieve using the Initialization Script option under General Settings. The simplest thing you can do, is iterate everything to see what information is available. You can do it with this script:

```
$.each(form.fields, function(i, field) {
   console.log(field);
});
```

Note that the info goes into the Console tab, so press F12 in your browser. The most useful information in the field object is the "id" property, which holds the ID of the HTML element. Therefore, you can can get a reference to it using javascript:

```
var el = document.getElementById(form.fields.SomeFieldName.id);
```

or jQuery:

```
var $el = $('#' + form.fields.SomeFieldName.id);
```

Once you have that, you can do DOM manipulation, install plugins and so on. For example, one could hide a textbox by using $el.hide(); If you're trying to integrate a 3rd party jQuery plugin, you'll first have to include it. Currently you can't do this with Action Form, so you'll have to add it to the DNN skin or the Page Header under Page Settings. After you do that, you can simply call the plugin on the jquery element. For example, the following script would provide a password complexity message using the complexify plugin:

```
$el.complexify(options, callback(valid, complexity){
    alert("Password complexity: " + complexity);
});
```

Other things to keep in mind is that this field also supports My Tokens. Note that if you use form tokens in this context they access the value of each field. So for example, [SomeFormField] is equivalent to form.fields.SomeFormField.value. One useful trick to know is that values can also be changed at this stage by assigning values to them. The following 2 are equivalent:
form.fields.SomeFieldName.value = 'Something new';
[SomeFieldName] = 'Something new';

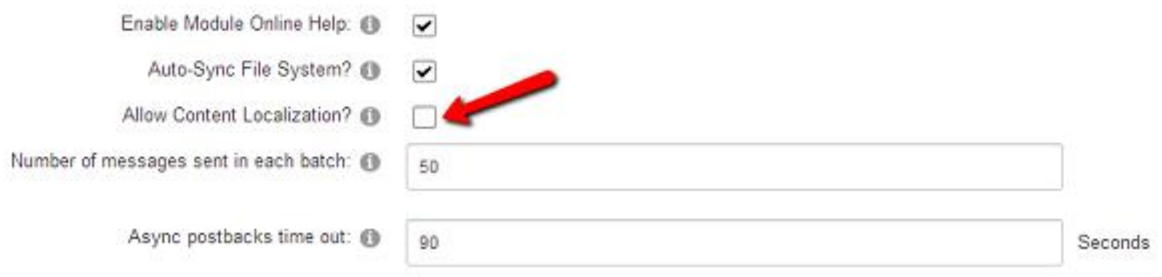
Localization

People all over the world treat the Internet as their main location for information and services. These people do not speak the same language so localization has become one of the primary tools for business global expansion. Doing this well will boost the company's ability to connect with those visiting the site, not to mention, foster a healthy relationship with potential customers.

Enabling Content Localization in DNN 7

Action Form can be easily translated. Before starting to add languages, you need to apply one setting: allow the site administrators to enable content localization for their site. Note that this is the standard procedure in DNN 7. You can do this by following the steps bellow:

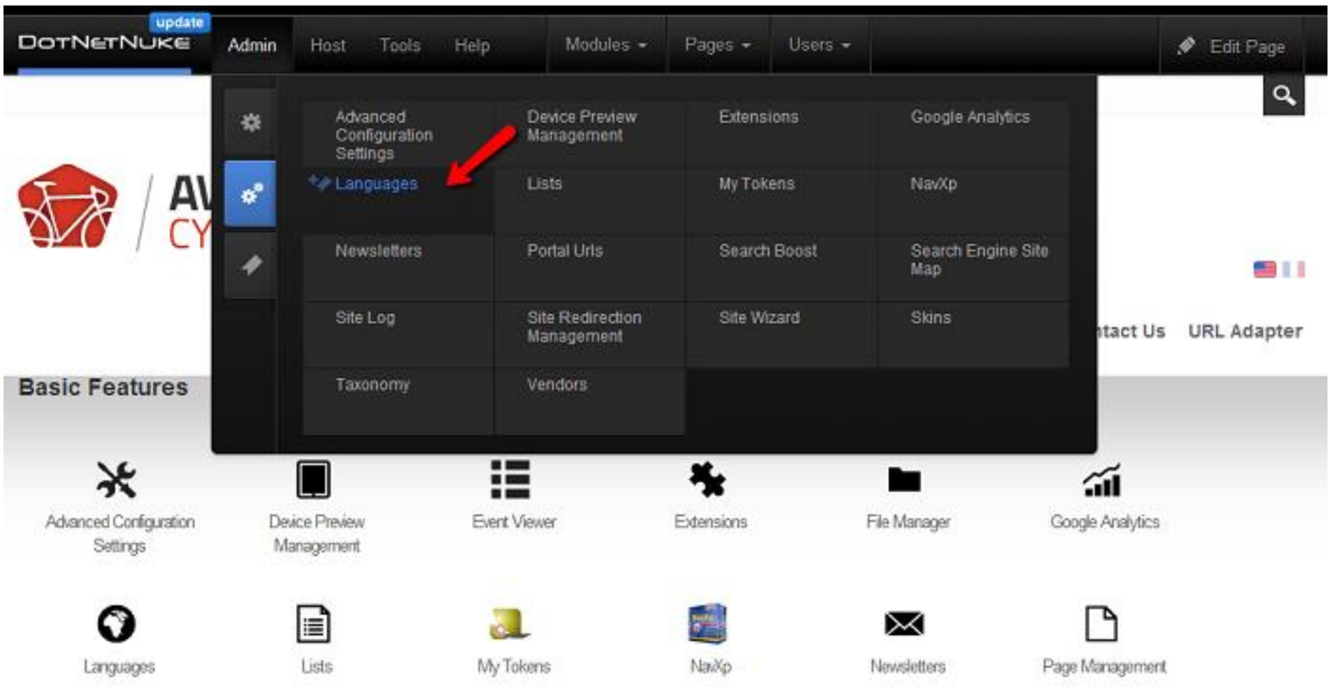
- Go to Host -> Host Settings -> Other Settings and put a check mark next to Allow Content Localization.



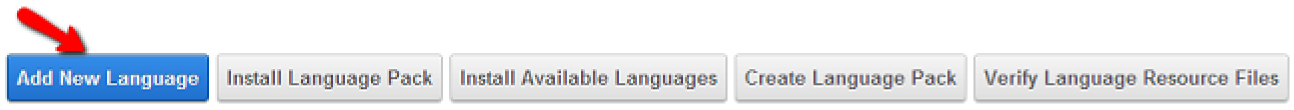
Enable Module Online Help:	✓
Auto-Sync File System?	✓
Allow Content Localization?	☐
Number of messages sent in each batch:	50
Async postbacks time out:	90 Seconds

Now let's add some languages to your site and translate the Action Form module.

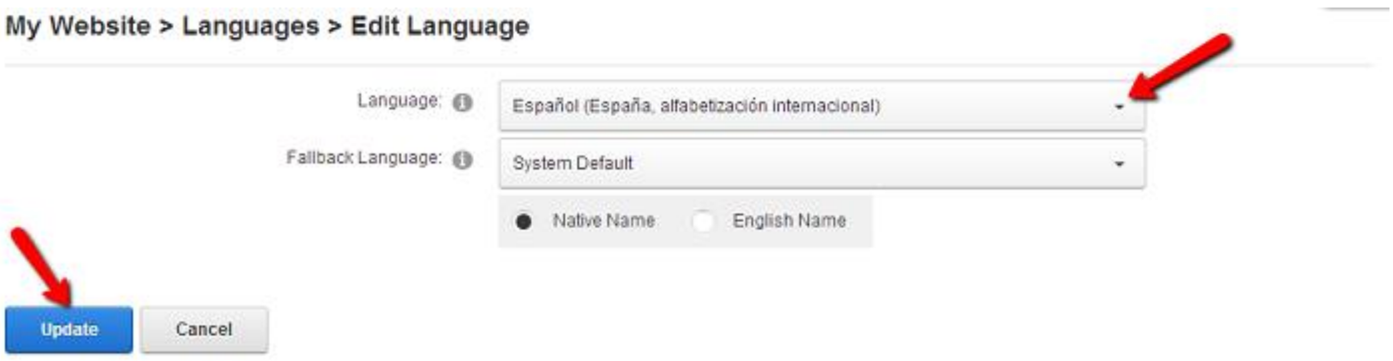
1. Go to Admin -> Advanced Settings -> Languages.



2. In this section you have the ability to add new languages and edit them. Before that, click on the Enable Content Localization button. As an example we are going to install the Spanish language. Click on the Add New Language button.



3. Select the language from the drop-down list and then click on the Update button.


My Website > Languages > Edit Language

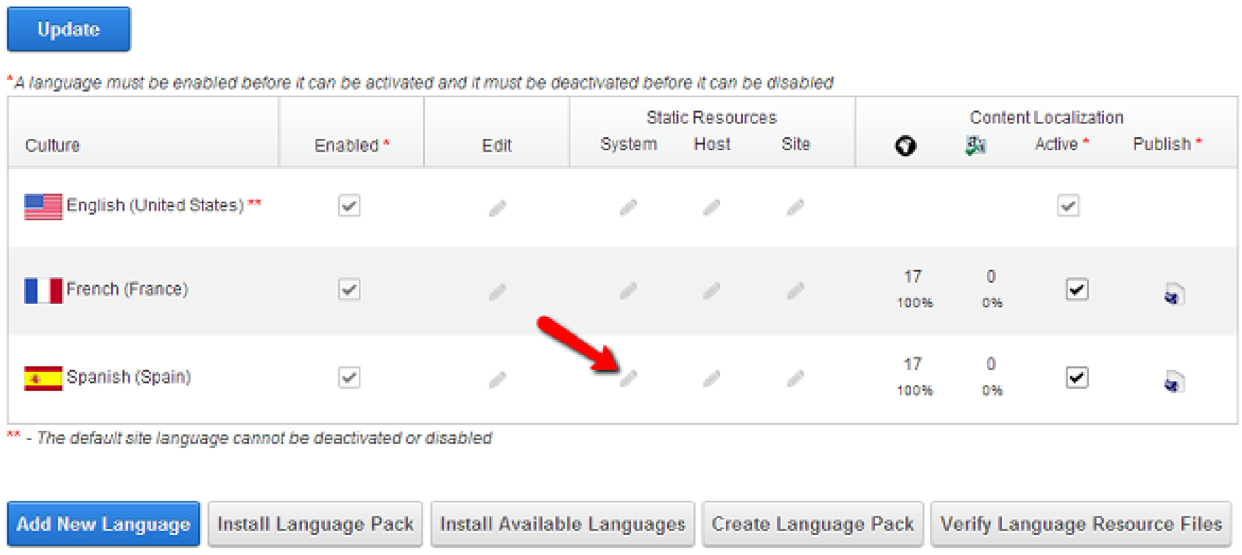
Language: Español (España, alfabetización internacional)

Fallback Language: System Default

[✓ Native Name] English Name

Editing Localization

1. Now we have installed the Spanish language. Put a check mark on the Content Localization - Active column to activate the language. Next click on the Edit button for the System column. A pop-up window will open with the Language Editor section.

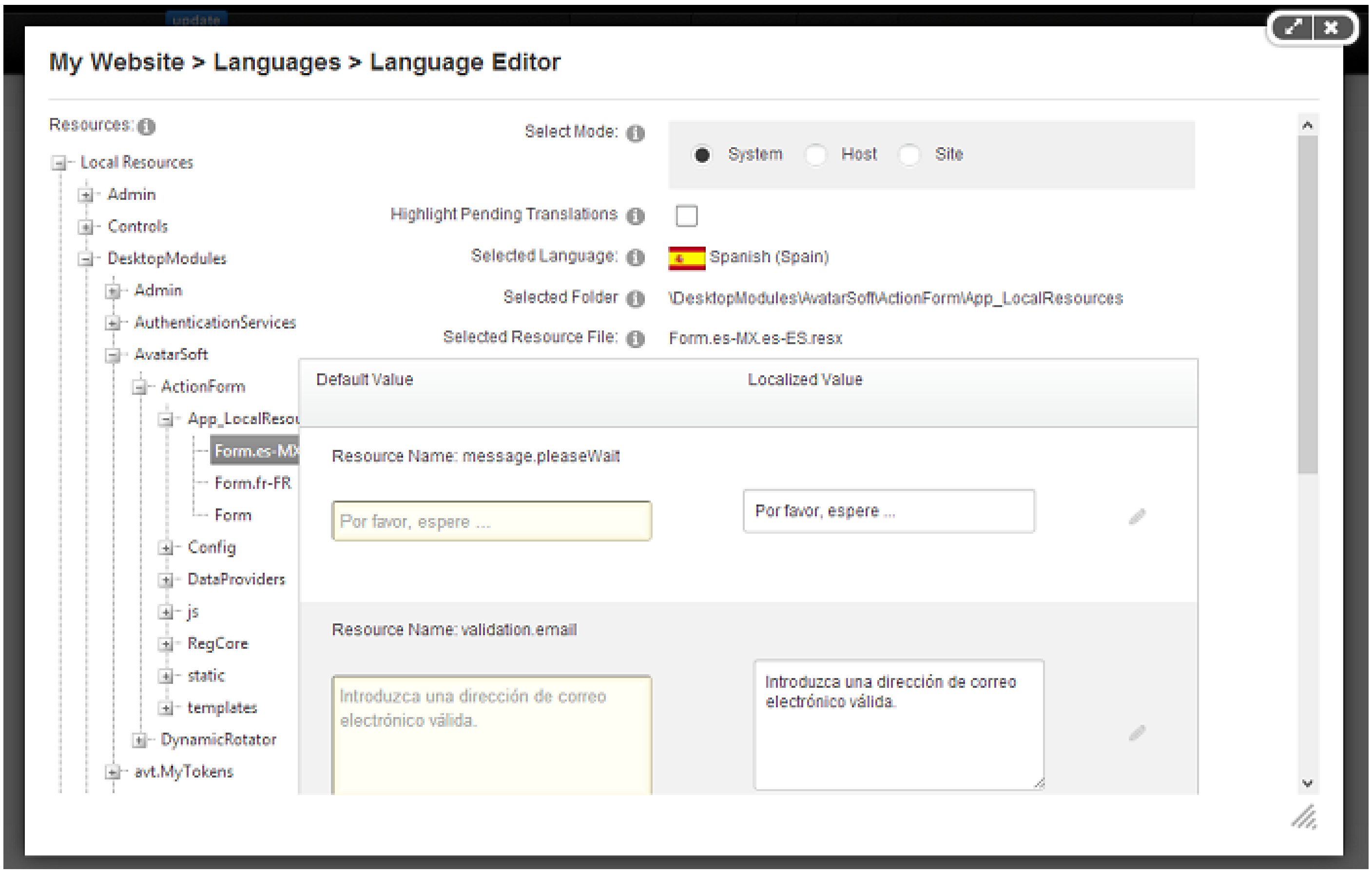


*A language must be enabled before it can be activated and it must be deactivated before it can be disabled

Culture	Enabled *	Edit	Static Resources System	Static Resources Host	Static Resources Site	Content Localization	Content Localization	Content Localization Active *	Content Localization Publish *
English (United States) **	✓	✎	✎	✎	✎			✓	
French (France)	✓	✎	✎	✎	✎	17 100%	0 0%	✓	
Spanish (Spain)	✓	✎	✎	✎	✎	17 100%	0 0%	✓	

** - The default site language cannot be deactivated or disabled

2. Select from the left menu Local Resources -> Desktop Modules -> AvatarSoft -> ActionForm -> App_LocalResources -> Form. In this section we can edit the Spanish language of the Action Form module.



My Website > Languages > Language Editor

Resources:

- Local Resources
  - Admin
  - Controls
  - DesktopModules
    - Admin
    - AuthenticationServices
    - AvatarSoft
      - ActionForm
        - App_LocalResou
          - Form.es-MX
          - Form.fr-FR
          - Form
        - Config
        - DataProviders
        - js
        - RegCore
        - static
        - templates
      - DynamicRotator
    - avt.MyTokens

Select Mode: [✓ System] Host Site

Highlight Pending Translations: ☐

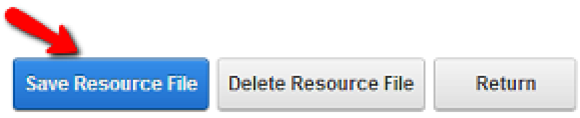
Selected Language: Spanish (Spain)

Selected Folder: \DesktopModules\AvatarSoft\ActionForm\App_LocalResources

Selected Resource File: Form.es-MX.es-ES.resx

Default Value	Localized Value
Resource Name: message.pleaseWait	
Por favor, espere ...	Por favor, espere ...
Resource Name: validation.email	
Introduzca una dirección de correo electrónico válida.	Introduzca una dirección de correo electrónico válida.

3. After you finish with the translation, click on Save Resource File button to save the changes. Now your form is translated.



What can be Localized?

You can localize all the front-end like validation messages, custom validations, the bootstrap template, also the standard templates (Contact Form, Login Form, Registration Form, Subscribe to Mailchimp). Note that for custom validators, the name must be like this: validation."the-validator-name". ex.: validation.url. You can localize the standard form templates (Contact Form, Login Form, Registration Form, Subscribe to Mailchimp) just by copy/paste and renaming the .config file.

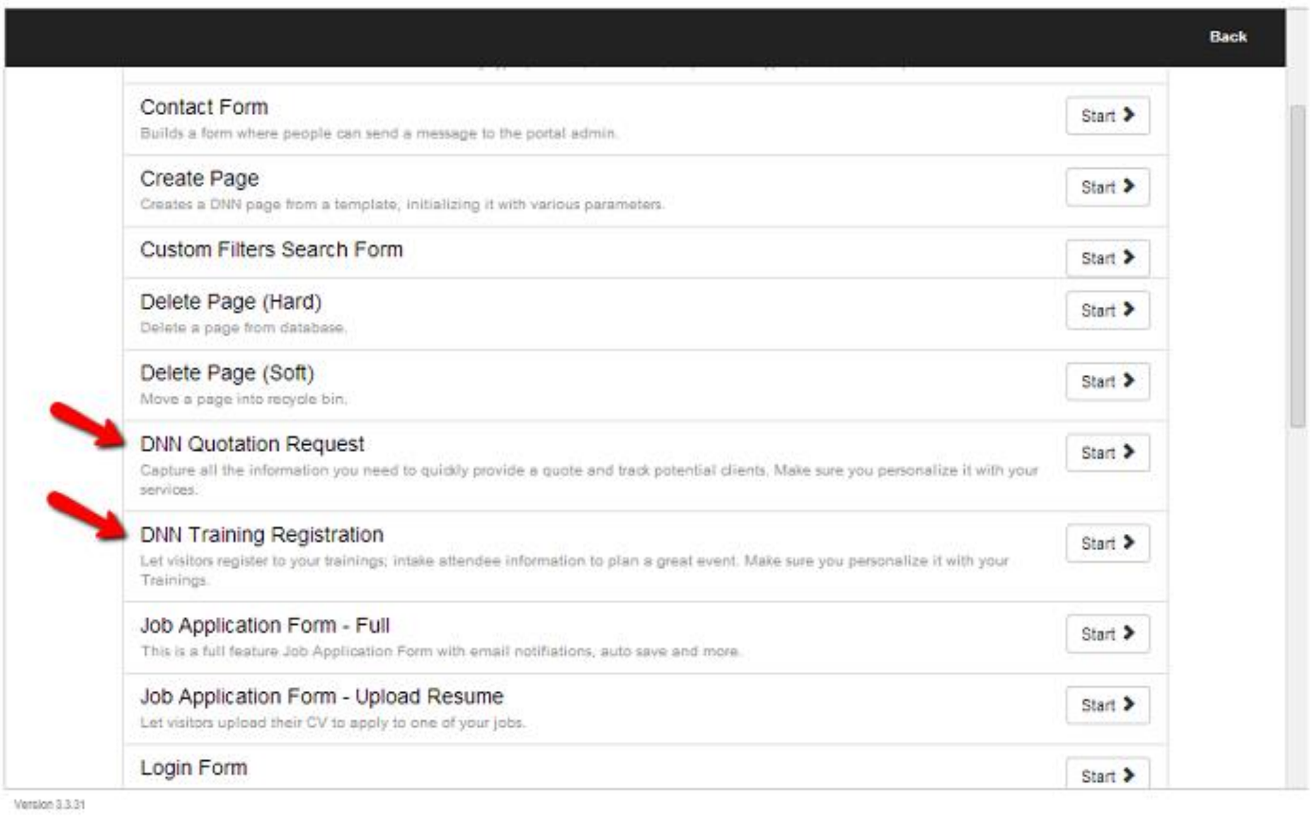
Ex.: In this example we will create and translate the Spanish language. Follow this path: LocalResources\DesktopModules\AvatarSoft\ActionForm\Config\Templates\ and copy/paste the Contact Form.config file; rename it into Contact Form es_ES.config. Translate the Contact Form es_ES.config file. Now when you will create a standard Contact Form, you'll have the possibility to chose the language in which the form is created.

Free Resources

Take advantage of Action Form's free library. All of these materials are free for download (find attached at the end of this page).

How you can use them?

All you have to do is to upload the template into the Templates folder of Action Form. This is the path where you need to upload the templates: DesktopModules\AvatarSoft\ActionForm\Config\Templates\. After this, you will have them in the control panel list of templates every time you add a new Action Form module. Hit the Start button and configure/personalize this form. You can delete or add new fields in order to suites your needs. Also make sure you set-up correctly actions on button; don't worry it's simple.



Contact Form
Builds a form where people can send a message to the portal admin.

Create Page
Creates a DNN page from a template, initializing it with various parameters.

Custom Filters Search Form

Delete Page (Hard)
Delete a page from database.

Delete Page (Soft)
Move a page into recycle bin.

DNN Quotation Request
Capture all the information you need to quickly provide a quote and track potential clients. Make sure you personalize it with your services.

DNN Training Registration
Let visitors register to your trainings; intake attendee information to plan a great event. Make sure you personalize it with your Trainings.

Job Application Form - Full
This is a full feature Job Application Form with email notifiations, auto save and more.

Job Application Form - Upload Resume
Let visitors upload their CV to apply to one of your jobs.

Login Form

Version 3.3.31

What you need to configure / check on these Action Form templates:

1. Quotation Request form template:

- Project Details section - personalize with your services
- Actions on button - send emails (if you need more info check this section Send Emails with Action Form)

2. Training Registration form template:

- Training Sign-up and Selection section - personalize with your trainings and prices
- Important notes about this Training
- Payment
- Actions on button - send emails (if you need more info check this section Send Emails with Action Form) and PayPal payments (Payments)

3. Customer Satisfaction Survey:

- Set it how you want it to be displayed (pop-up, separate page, etc.)
- Personalize with your company Name and Logo.
- Actions on button - send emails (if you need more info check this section Send Emails with Action Form)

4. Customer Satisfaction Specific Survey:

- Set it how you want it to be displayed (pop-up, separate page, etc.)
- Personalize with your company Name and Logo.
- Actions on button - send emails (if you need more info check this section Send Emails with Action Form)

5. Tech Support Satisfaction Survey:

- Set it how you want it to be displayed (pop-up, separate page, etc.)
- Personalize with your company Name and Logo.
- Actions on button - send emails (if you need more info check this section Send Emails with Action Form)

Lifecycle

Action Form starts out simple, but as soon as you start adding actions and manipulating data, it can get quite complex. Without understanding the life cycle of how Action Form operates with data, you're down to trial and error, which can be a reason of frustration. This page demystifies this processes.

Form Initialization

When the form first loads, data can be preloaded into the form using many different settings. It's very important to understand the order in which these apply, so you can plan best architecture.

- **A new blank form context is created.**

The context is just a repository where all data lives. Values of the form fields are automatically loaded into context, but you can also load data manually using actions such as Execute SQL or Inject Data.

- **PreInit actions are executed.**

These are a stack of actions defined under Form Events section. At this stage, load data into context that you need to use inside the form fields (for example in default values or in the list of values of a dropdown).

- **Load Default Values**

At this stage, Action Form injects the form tokens into the context, initializing them with the default values. If during the PreInit phase you've loaded data with a name the same as a form field, it will get overwritten by the field default value. If you want that to stick, move the action in the Init event.

- **Load existing entry**

If a parameter named entry is present in query string, Action Form will try to find a matching entry in the internal reports table. If it finds one, it loads it's value into the interface, overwriting all values that were computed so far. This is how the Action Grid integration works.

- **Init actions are executed**

This is the final stage of initialization. Anything you inject into the context at this point will overwrite existing data. For example, add an SQL Action and match columns to field names.

Tokens

Tokens are a mechanism to pull dynamic data inside a static template. Think of the classical example, "Hello [FirstName], DNN Sharp loves you". Here, [FirstName] is a token that would get replaced with the actual first name of a person, for example, of the person receiving an email.

Action Form supports a few types of tokens in various contexts.

- **Form Data**

The value of any form fields can be references using a token, which is its ID between square brackets. For example, if the ID of a form field is FirstName, then the token is [FirstName]. Note that the ID and the token can't contain spaces or punctuation.

- **Utility Tokens**

[_ReportKey] - unique ID that identifies the current submission. This also powers edit functionality in Action Form. Whenever a entry parameter is present in query string, Action Form tries to match the associated report entry and load it into the form.
[_Referrer] - holds the URL from which the user arrived to the form
[_UserIp] - holds the IP of the user
[_FormUrl] - the URL where the form lives
[_FormUrlRelative] - same as above, only it's a relative URL
[_EditUrl] - is the URL to edit current submission at a later time
[_IsNew] - true when the form is submitted for the first time
[_IsEdit] - true when the form is edited (for example using the [_EditUrl] token or Action Grid).

- **My Tokens**

My Tokens comes with dozens of new predefined tokens such as Session, Server, Query String or Post Data access. It also allows to create custom tokens from database queries, HTTP Requests or files.

Validations

Client Side Validation

Client side form validation is essential because it saves time and bandwidth. It gives you more options to point out to the users where they are filling the form in a wrong way. This doesn't mean that you don't need server side validation; server side validation is implicit and cannot be deactivated. It is possible that the people who visit your website to use an old browser or they have JavaScript disabled, which will break client side only validation. Client side and server side validations complement each other and they should not be used independently.

Client Side Validation is good to be used because of this two main reasons:

- It is a fast form validation; if the user is filling the form in a wrong way, the alarm is activated when transmitting the form.
- It set-ups to display only one error at a time and focus on the wrong field; this way you help and ensure that the user is filling in correctly all the form fields.

In Action Form we've set-up an option to enable or disable the client side validation. Why? So that you can treat the Event on Validation Failed or maybe you want to implement your own validation logic.

Support

- Our support staff is friendly and always available to help you. Click here to ASK US.
- Or contact us at: support@dnnsharp.com

Useful Links

- Request support on our support forum.
- Purchase a new license from DNN Store.
- More info about Action Form at http://www.dnnsharp.com/dnn/modules/action-form-builder
-
- Latest version of this documentation at http://action-form.dnnsharp.com/