Kentico allows you to program your own action steps, which users can then incorporate into automation processes. You need to:

1. Write the action's code in a custom class.
2. Register the class as a new action in the system.

Writing custom actions

1. Create a new class in your Kentico web project. You can:
   - Define the action step in **App_Code** and load the class via the API.
   - Create a new assembly (Class library) and include the class there. You must add the appropriate references to both the assembly and the main Kentico project.
2. Set the class to inherit from:
   - **CMS.Automation.AutomationAction** - for general automation actions
   - **CMS.ContactManagement.ContactAutomationAction** - for actions that work with the contacts being handled by the process
3. Override the *Execute()* method from the base class.

```
public override void Execute()
{
}
```

Write the code that you want the action to run into the *Execute* method's body.

Working with action parameters

You can use parameters to allow users to modify the behavior of your custom action. Users can edit the values of parameters when configuring specific instances of the action step in the workflow designer. If you want your action step to use parameters, you need to define the parameters as fields when registering the action in the system.

To access the values of parameters in your code, use the *GetResolvedParameter<ParameterType>* method. The method accepts the following parameters:

- **parameter name** - the exact **Field name** set for the parameter in the system.
- **default value** - the value returned if the parameter is empty for the given action step.

Example

```
GetResolvedParameter<string>("MessageText", string.Empty);
```

Registering actions in the system

Once you create the class containing the required logic, you need to register the custom action:

1. Log in to the Kentico administration interface and open the **Marketing automation** application.
2. Switch to the **Actions** tab.
3. Click **New action**.
4. Fill in the following fields:
   - **Display name** - the name of the action step used in the user interface.

- **Action provider - Assembly name** - specify the name of the library where the action is implemented. Select *(custom classes)* for classes defined in the application's App_Code (or Old_App_Code) folder.
- **Action provider - Class name** - specify the full name of the class that contains the action's code (including namespaces).

5. Click **Save**.
6. If the action has any parameters, switch to the **Parameters** tabs and define the form fields through which users can configure the action.

Users can now place your custom action into the automation process designer.