Sublogin Documentation

PDF Manual

MODULE

sublogin

LAST UPDATED

2026-03-29

Extending Frontend Form Fields

Sublogin Documentation · /sublogin/features/extending-frontend-form-fields

This guide explains how to add or adjust fields on the **Sublogin frontend create/edit form** (Customer Account area) by extending the field definition returned by `getFormFields()`.

When To Use This

Use this if you need:

- custom, customer-specific fields on the Sublogin form
- additional toggles/selectors that depend on your business logic

This approach is useful to avoid forking the Sublogin module.

Field Reference (What The Form Can Show)

The exact form fields can vary by configuration and installed add-ons. Common base fields are:

- **Customer** (Admin-only): selects the main customer account the Sublogin belongs to.
- **Store**: the store view the Sublogin is created for (can affect store context and emails).
- **Prefix**: only shown if prefixes are enabled in the Magento customer configuration.
- **Firstname / Lastname**
- **Email**: must be unique across the whole system (customers and sublogins).
- **Password**: set manually or auto-generate (UI-dependent).
- **Shared Customer Addresses**: shown when address assignment is restricted; lets you choose which main-account addresses the Sublogin can use.

Other common (optional) fields/sections:

- **Active/Enabled**
- **Send Backend Mails** (notifications)
- **Can Create Sublogins**
- **Expire Date**
- **Payment/Shipping Method Restrictions** (if enabled)

Add-ons may add additional fields (examples):

- Roles & Permissions: role/group assignment
- Budget Management: budget configuration
- Order Approval: approval flags/rules
- Catalog Visibility: restriction type and assignments

Extending `getFormFields()` Via Plugin

You can extend the frontend form fields by creating a Magento plugin for:

- `MageB2B\\Sublogin\\Block\\Sublogin\\Edit::getFormFields()`

1. Define The Plugin (di.xml)

Create `etc/frontend/di.xml` in your custom module:

```
<?xml version="1.0"?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="urn:magento:framework:ObjectManager/etc/conf
ig.xsd">
    <type name="MageB2B\Sublogin\Block\Sublogin\Edit">
        <plugin name="company_module_sublogin_form_fields"
type="Company\Module\Plugin\MageB2B\Sublogin\Block\Sublogin\Edit\AddCustomF
ield" />
    </type>
</config>
```

2. Implement `afterGetFormFields()`

Create the plugin class:

```
<?php

declare(strict_types=1);

namespace Company\Module\Plugin\MageB2B\Sublogin\Block\Sublogin\Edit;

use MageB2B\Sublogin\Block\Sublogin\Edit;

class AddCustomField
{
    public function afterGetFormFields(Edit $subject, array $result): array
    {
        $customFields = [
            [
                'name' => 'yes_no_dropdown',
                'label' => __('Yes No Dropdown'),
                'type' => 'dropdown',
                'default' => 1,
                'options' => [
                    ['value' => 1, 'label' => __('Yes')],
                    ['value' => 0, 'label' => __('No')],
                ],
                'visible' => true,
                'position' => 1400,
            ],
        ];

        return array_merge($result, $customFields);
    }
}
```

Field Definition Keys

The field array typically uses keys like:

- `name`: unique field identifier (used as input name)

- `label`: displayed label (use `__()` for translation)
- `type`: renderer/type used by the form (e.g. `text`, `password`, `dropdown`)
- `default`: default value
- `options`: for select/dropdown fields (array of `value`/`label`)
- `visible`: whether the field is shown
- `position`: sorting position (higher usually means later)

Important Notes

- This hook changes what is **rendered** on the form. How a new field is **validated, saved, and loaded** depends on your implementation and where you persist the value.
- If you are unsure which field definition keys/types are supported in your installation, inspect the existing `$result` structure returned by `getFormFields()` first, then mirror the same patterns for your custom field.