Advanced Conditional Logic for RSForm! Pro — Complete Examples Guide

This guide walks through every major feature with step-by-step setup instructions. Each example specifies exactly which form fields to create first — including field type, name, caption, and options — followed by the Advanced Conditional Logic field configuration.

How to Read These Examples

Every example follows this structure:

Step 1 — Create the form fields Listed in the order they should appear in the form. For each field:

- **Type** — the RSForm! Pro field type to select when adding the field
- **Name** — the internal identifier (used in conditions and actions)
- **Caption** — the label shown to the user
- **Options / Settings** — values, defaults, and other configuration

Step 2 — Add the Advanced Conditional Logic field This is our plugin's field. Add it after the fields it references. Configure the CONDITION tab and ACTIONS tabs as shown.

---

1. SUBMISSION COUNT

Example 1.1 — Early Bird Discount

Scenario: Conference registration. The first 100 registrants pay €299; everyone after pays €499. A badge message appears for early registrants.

Step 1 — Create these form fields first

Field 1

- Type: **Text Box**
- Name: `ticket_price`
- Caption: `Ticket Price`
- Default Value: *(leave empty — the condition will set it)*
- Read Only: Yes *(so the user cannot edit it manually)*

Field 2

- Type: **Text Box**

- Name: `discount_code`
- Caption: `Discount Code`
- Default Value: *(empty)*
- Read Only: Yes

Field 3

- Type: **Free Text** *(display-only, no input)*
- Name: `early_bird_badge`
- Caption: *(empty)*
- Text: *(leave empty — the condition will set it)*

Step 2 — Add the Advanced Conditional Logic field

- Type: **Advanced Conditional Logic**
- Name: `cond_early_bird`
- Caption: *(empty)*

CONDITION tab:

Setting	Value
Condition Type	Submission count
Operator	less than
Value	100

ACTIONS — IF TRUE *(fewer than 100 submissions)*

Target field	Action	Value
`ticket_price`	Set value	299
`discount_code`	Set value	EARLY100
`early_bird_badge`	Set HTML	`<strong>Early bird discount: 40% off!</strong>`
`early_bird_badge`	Show	—

ACTIONS — IF FALSE *(100 or more submissions)*

Target field	Action	Value
`ticket_price`	Set value	499
`discount_code`	Clear value	—

Target field	Action	Value
`early_bird_badge`	Hide	—

What happens: When the form loads, the plugin checks the current submission count. If fewer than 100 have been submitted, it sets the price to 299, shows the discount code and badge. Otherwise it sets 499 and hides the badge. The user sees the correct price immediately on page load — no interaction needed.

Example 1.2 — Progressive Capacity Messages

Scenario: Workshop with 100 spots. The availability message changes as the workshop fills up. A waitlist email field appears only when full.

Step 1 — Create these form fields first

Field 1

- Type: **Free Text**
- Name: `capacity_notice`
- Caption: *(empty)*
- Text: *(empty — the condition sets it)*

Field 2

- Type: **Text Box**
- Name: `waitlist_email`
- Caption: `Waitlist Email Address`
- *(This field is hidden initially — the condition shows it when full)*

Step 2 — Add three Advanced Conditional Logic fields

Conditional Field 1 — Available (0–79 registrants)

- Name: `cond_capacity_ok`

CONDITION tab:

Setting	Value
Condition Type	Submission count
Operator	less than
Value	80

ACTIONS — IF TRUE

Target field	Action	Value
`capacity_notice`	Set HTML	`<strong>Plenty of spots available!</strong>`
`capacity_notice`	Set class	`alert alert-success`
`waitlist_email`	Hide	—
`waitlist_email`	Optional	—

Conditional Field 2 — Almost Full (80–99)

- Name: `cond_capacity_warning`

CONDITION tab:

Setting	Value
Condition Type	Combined
Logic operator	AND
Rule 1	Submission count — greater or equal — 80
Rule 2	Submission count — less than — 100

ACTIONS — IF TRUE

Target field	Action	Value
`capacity_notice`	Set HTML	`<strong>Only a few spots left!</strong>`
`capacity_notice`	Set class	`alert alert-warning`

Conditional Field 3 — Full (100+)

- Name: `cond_capacity_full`

CONDITION tab:

Setting	Value
Condition Type	Submission count
Operator	greater or equal
Value	100

ACTIONS — IF TRUE

Target field	Action	Value
`capacity_notice`	Set HTML	`<strong>Workshop is full.</strong> Join our waitlist below.`
`capacity_notice`	Set class	`alert alert-danger`
`waitlist_email`	Show	—
`waitlist_email`	Required	—

---

2. FIELD VALUE

Example 2.1 — VIP Ticket: Show Extra Fields

Scenario: Event registration. When VIP is selected, additional company and job title fields appear and become required.

Step 1 — Create these form fields first

Field 1

- Type: **Select List**
- Name: `ticket_type`
- Caption: `Ticket Type`
- Options *(one per line in RSForm! field editor)*:

```
Standard|Standard
VIP|VIP
```

- Default: Standard

Field 2

- Type: **Text Box**
- Name: `company_name`
- Caption: `Company Name`
- *(Will be hidden initially by the condition)*

Field 3

- Type: **Text Box**
- Name: `job_title`
- Caption: `Job Title`

- *(Will be hidden initially by the condition)*

Field 4

- Type: **Free Text**
- Name: `vip_info`
- Caption: *(empty)*
- Text: *(empty — condition sets it)*
- *(Will be hidden initially by the condition)*

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_vip`

CONDITION tab:

Setting	Value
Condition Type	Field value
Field	ticket_type
Operator	equals
Value	VIP *(select from the dropdown — do not type)*

ACTIONS — IF TRUE

Target field	Action	Value
`company_name`	Show	—
`company_name`	Required	—
`job_title`	Show	—
`job_title`	Required	—
`vip_info`	Set HTML	`<strong>VIP includes:</strong> Reserved seating · Networking dinner · Gift`
`vip_info`	Show	—

ACTIONS — IF FALSE

Target field	Action	Value
`company_name`	Hide	—
`company_name`	Optional	—
`job_title`	Hide	—

Target field	Action	Value
`job_title`	Optional	—
`vip_info`	Hide	—

Note: Because the IF FALSE list contains `hide` actions, the plugin automatically hides these fields on page load — before JavaScript runs. There is no flash of unwanted content.

Example 2.2 — Checkbox: Dietary Preferences

Scenario: Conference catering. When "Vegetarian" is checked, a menu note appears. When "Gluten-free" is checked, an allergy details field appears and becomes required.

Step 1 — Create these form fields first

Field 1

- Type: **Checkbox Group**
- Name: `dietary`
- Caption: `Dietary Requirements`
- Options:

```
Standard|No special requirements
Vegetarian|Vegetarian
Vegan|Vegan
Gluten-free|Gluten-free
Dairy-free|Dairy-free
```

Field 2

- Type: **Free Text**
- Name: `veggie_note`
- Caption: *(empty)*
- Text: *(empty — condition sets it)*

Field 3

- Type: **Text Box**
- Name: `allergy_details`
- Caption: `Please describe your allergy in detail`

Step 2 — Add two Advanced Conditional Logic fields

Conditional Field 1 — Vegetarian note

- Name: `cond_vegetarian`

CONDITION tab:

Setting	Value
Condition Type	Field value
Field	dietary
Operator	contains
Value	Vegetarian *(select from dropdown)*

ACTIONS — IF TRUE

Target field	Action	Value
`veggie_note`	Set value	Our vegetarian menu is fully certified.
`veggie_note`	Show	—

ACTIONS — IF FALSE

Target field	Action	Value
`veggie_note`	Hide	—

Why `contains` and not `equals`? When a user checks both "Vegetarian" and "Gluten-free", RSForm! stores the value as `"Vegetarian,Gluten-free"`. The `equals` operator would look for an exact match of the full string and fail. The `contains` operator correctly checks whether "Vegetarian" appears anywhere in that string.

Conditional Field 2 — Gluten-free details

- Name: `cond_gluten`

CONDITION tab:

Setting	Value
Condition Type	Field value
Field	dietary
Operator	contains
Value	Gluten-free *(select from dropdown)*

ACTIONS — IF TRUE

Target field	Action	Value
`allergy_details`	Show	—

Target field	Action	Value
`allergy_details`	Required	—

ACTIONS — IF FALSE

Target field	Action	Value
`allergy_details`	Hide	—
`allergy_details`	Optional	—

Example 2.3 — Radio Group: Payment Method

Scenario: Order form. Different input fields appear depending on which payment method the user selects.

Step 1 — Create these form fields first

Field 1

- Type: **Radio Group**
- Name: `payment_method`
- Caption: `Payment Method`
- Options:

```
credit_card|Credit Card
bank_transfer|Bank Transfer
paypal|PayPal
```

- Default: credit_card

Field 2

- Type: **Text Box**
- Name: `card_number`
- Caption: `Card Number`

Field 3

- Type: **Text Box**
- Name: `bank_account`
- Caption: `Bank Account Number`

Field 4

- Type: **Text Box**

- Name: `paypal_email`
- Caption: `PayPal Email Address`

Step 2 — Add three Advanced Conditional Logic fields

Conditional Field 1 — Credit Card

- Name: `cond_credit_card`

CONDITION tab:

Setting	**Value**
Condition Type	Field value
Field	payment_method
Operator	equals
Value	credit_card *(select from dropdown)*

ACTIONS — IF TRUE

Target field	**Action**	**Value**
`card_number`	Show	—
`card_number`	Required	—
`bank_account`	Hide	—
`bank_account`	Optional	—
`paypal_email`	Hide	—
`paypal_email`	Optional	—

ACTIONS — IF FALSE

Target field	**Action**	**Value**
`card_number`	Hide	—
`card_number`	Optional	—

Repeat the same pattern for `bank_transfer` and `paypal` — each gets its own conditional field that shows the relevant input and hides the others.

Why not `contains` for radio? Radio always stores a single value. Use `equals` / `not_equals`.

Example 2.4 — Dynamic Option List

Scenario: Service booking. Premium clients see only afternoon slots; basic clients see all slots.

Step 1 — Create these form fields first

Field 1

- Type: **Select List**
- Name: `service_type`
- Caption: `Service Type`
- Options:

```
basic|Basic
premium|Premium
```

Field 2

- Type: **Select List**
- Name: `time_slot`
- Caption: `Preferred Time`
- Options *(add all slots — the condition removes the ones not available)*:

```
08:00|08:00
10:00|10:00
12:00|12:00
14:00|14:00
16:00|16:00
```

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_time_slots`

CONDITION tab:

Setting	**Value**
Condition Type	Field value
Field	service_type
Operator	equals
Value	premium *(select from dropdown)*

ACTIONS — IF TRUE *(Premium: morning slots removed)*

Target field	**Action**	**Value**
`time_slot`	Remove option	08:00 *(select from dropdown)*

Target field	Action	Value
`time_slot`	Remove option	10:00
`time_slot`	Remove option	12:00

ACTIONS — IF FALSE *(Basic: morning slots restored)*

Target field	Action	Value
`time_slot`	Add option	`08:00|08:00`
`time_slot`	Add option	`10:00|10:00`
`time_slot`	Add option	`12:00|12:00`

Add option format: `value|Display Label`. Add option checks for duplicates automatically — it is safe to re-apply. Always add all slots to the field initially and use `remove_option` to hide them — this gives you full control over ordering.

Example 2.5 — Disable and Hide Individual Options

Scenario: Delivery options. The "Express" option is disabled when stock is low (controlled via a hidden field). The "Pickup" option is hidden for international shipping addresses.

Step 1 — Create these form fields first

Field 1

- Type: **Select List**
- Name: `shipping_country`
- Caption: `Country`
- Options: *(your country list)*

Field 2

- Type: **Hidden**
- Name: `stock_level`
- Caption: *(empty)*
- Default Value: `ok` *(set to `low` programmatically when stock is low)*

Field 3

- Type: **Select List**
- Name: `delivery_method`
- Caption: `Delivery Method`

- Options:

```
standard|Standard (3–5 days)
express|Express (next day)
pickup|Store Pickup
```

Step 2 — Add two Advanced Conditional Logic fields

Conditional Field 1 — Disable Express when stock is low

- Name: `cond_express_stock`

CONDITION tab:

Setting	**Value**
Condition Type	Field value
Field	stock_level
Operator	equals
Value	low

ACTIONS — IF TRUE

Target field	**Action**	**Value**
`delivery_method`	Disable option	express *(select from dropdown)*

ACTIONS — IF FALSE

Target field	**Action**	**Value**
`delivery_method`	Enable option	express

Conditional Field 2 — Hide Pickup for international orders

- Name: `cond_pickup_visibility`

CONDITION tab:

Setting	**Value**
Condition Type	Field value
Field	shipping_country
Operator	not equals
Value	HU *(or your local country code)*

ACTIONS — IF TRUE

Target field	Action	Value
`delivery_method`	Hide option	pickup

ACTIONS — IF FALSE

Target field	Action	Value
`delivery_method`	Show option	pickup

Disable vs Hide: `disable_option` keeps the option visible but greyed out — the user can see it is unavailable. `hide_option` removes it from the list entirely. Choose based on whether you want users to know the option exists.

Example 2.6 — Checkbox Pre-selection

Scenario: Newsletter signup. When "VIP" membership is selected, the "Premium Updates" checkbox is automatically checked and "Basic Updates" is unchecked.

Step 1 — Create these form fields first

Field 1

- Type: **Select List**
- Name: `membership_type`
- Caption: `Membership Type`
- Options:

```
standard|Standard
vip|VIP
```

Field 2

- Type: **Checkbox Group**
- Name: `newsletter_prefs`
- Caption: `Newsletter Preferences`
- Options:

```
basic|Basic Updates
premium|Premium Updates
events|Event Invitations
```

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_newsletter_prefs`

CONDITION tab:

Setting	Value
Condition Type	Field value
Field	membership_type
Operator	equals
Value	vip *(select from dropdown)*

ACTIONS — IF TRUE

Target field	Action	Value
`newsletter_prefs`	Check option	premium *(select from dropdown)*
`newsletter_prefs`	Uncheck option	basic

ACTIONS — IF FALSE

Target field	Action	Value
`newsletter_prefs`	Uncheck option	premium
`newsletter_prefs`	Check option	basic

Example 2.7 — Select Pre-selection

Scenario: Order form. When "Corporate" account type is chosen, the billing period automatically switches to "Annual".

Step 1 — Create these form fields first

Field 1

- Type: **Select List**
- Name: `account_type`
- Caption: `Account Type`
- Options:

```
personal|Personal
corporate|Corporate
```

Field 2

- Type: **Select List**
- Name: `billing_period`
- Caption: `Billing Period`
- Options:

```
monthly|Monthly
annual|Annual (save 20%)
```

- Default: monthly

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_billing_period`

CONDITION tab:

Setting	Value
Condition Type	Field value
Field	account_type
Operator	equals
Value	corporate *(select from dropdown)*

ACTIONS — IF TRUE

Target field	Action	Value
`billing_period`	Select option	annual *(select from dropdown)*

ACTIONS — IF FALSE

Target field	Action	Value
`billing_period`	Select option	monthly

Example 2.8 — Regex: Input Format Validation

Scenario: Application form. The postal code field must be exactly 4 digits. A message field shows instant feedback as the user types.

Step 1 — Create these form fields first

Field 1

- Type: **Text Box**

- Name: `postal_code`
- Caption: `Postal Code`
- Validation: *(optional — the condition gives real-time visual feedback, RSForm! validation handles submit-time enforcement)*

Field 2

- Type: **Free Text**
- Name: `postal_code_hint`
- Caption: *(empty)*
- Text: *(empty — condition sets it)*

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_postal_format`

CONDITION tab:

Setting	Value
Condition Type	Field value
Field	postal_code
Operator	matches regex
Value	`^[0-9]{4}$`

ADVANCED tab:

Setting	Value
Real-time Update	Yes *(re-evaluates on every keystroke)*

ACTIONS — IF TRUE *(format is valid)*

Target field	Action	Value
`postal_code_hint`	Set HTML	`<span style="color:green">✓ Valid postal code</span>`
`postal_code_hint`	Show	—

ACTIONS — IF FALSE *(format is invalid or empty)*

Target field	Action	Value
`postal_code_hint`	Set HTML	`<span`

Target field	Action	Value
		`style="color:#999">Please enter a 4-digit postal code</span>`
`postal_code_hint`	Show	—

How regex evaluation works: With Real-time Update enabled, the condition re-evaluates on every keystroke. The user sees instant green/grey feedback as they type — without submitting the form. The pattern `^[0-9]{4}$` means: start of string (`^`), exactly four digits (`[0-9]{4}`), end of string (`$`). Write patterns without delimiters — the plugin adds them automatically.

Other useful patterns:

Purpose	Pattern
Hungarian postal code	`^[0-9]{4}$`
Phone number (flexible)	`^\+?[0-9\s\-]{8,}$`
Basic email format	`^[^@\s]+@[^@\s]+\.[^@\s]+$`
Letters only (with accents)	`^[a-zA-ZáéíóöőúüűÁÉÍÓÖŐÚÜŰ\s]+$`
Coupon code (A–Z, 0–9, 6–10 chars)	`^[A-Z0-9]{6,10}$`

---

3. TIME BASED

Example 3.1 — Three-Tier Promotional Pricing

Scenario: Product launch with three price tiers: Super Early Bird (before March 1), Early Bird (March–April), Regular (after April 1).

Step 1 — Create these form fields first

Field 1

- Type: **Text Box**
- Name: `price`
- Caption: `Price`
- Read Only: Yes

Field 2

- Type: **Free Text**
- Name: `promo_badge`

- Caption: *(empty)*
- Text: *(empty)*

Step 2 — Add three Advanced Conditional Logic fields

Use https://www.unixtimestamp.com/ to convert dates, or use the built-in datetime picker — it converts automatically.

Conditional Field 1 — Super Early Bird (before March 1)

- Name: `cond_super_early`

CONDITION tab:

Setting	Value
Condition Type	Time based
Operator	less than
Value	1740787200 *(2026-03-01 00:00 UTC — use datetime picker)*

ACTIONS — IF TRUE

Target field	Action	Value
`price`	Set value	799
`promo_badge`	Set HTML	`<strong>SUPER EARLY BIRD: 60% off!</strong>`
`promo_badge`	Set class	`alert alert-danger`
`promo_badge`	Show	—

Conditional Field 2 — Early Bird (March 1 – April 1)

- Name: `cond_early_bird`

CONDITION tab:

Setting	Value
Condition Type	Combined
Logic operator	AND
Rule 1	Time based — greater or equal — 1740787200
Rule 2	Time based — less than — 1743465600

ACTIONS — IF TRUE

Target field	Action	Value
`price`	Set value	1199
`promo_badge`	Set HTML	`<strong>EARLY BIRD: 40% off!</strong>`
`promo_badge`	Set class	`alert alert-warning`
`promo_badge`	Show	—

Conditional Field 3 — Regular Price (after April 1)

- Name: `cond_regular_price`

CONDITION tab:

Setting	Value
Condition Type	Time based
Operator	greater or equal
Value	1743465600 *(2026-04-01 00:00 UTC)*

ACTIONS — IF TRUE

Target field	Action	Value
`price`	Set value	1999
`promo_badge`	Set HTML	`Regular price — promotional period has ended.`
`promo_badge`	Set class	`alert alert-info`
`promo_badge`	Show	—

---

4. USER BASED

Example 4.1 — Pre-fill Fields for Logged-in Users

Scenario: Contact form. Logged-in users do not need to type their name and email — those fields are hidden and a greeting appears instead.

Step 1 — Create these form fields first

Field 1

- Type: **Text Box**

- Name: `sender_name`
- Caption: `Your Name`
- Required: Yes

Field 2

- Type: **Text Box**
- Name: `sender_email`
- Caption: `Your Email`
- Required: Yes

Field 3

- Type: **Free Text**
- Name: `logged_in_note`
- Caption: *(empty)*
- Text: *(empty)*

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_logged_in`

CONDITION tab:

Setting	Value
Condition Type	User based
User Condition	is logged in
(no value needed)	

ACTIONS — IF TRUE *(user is logged in)*

Target field	Action	Value
`sender_name`	Hide	—
`sender_name`	Optional	—
`sender_email`	Hide	—
`sender_email`	Optional	—
`logged_in_note`	Set value	Sending as your registered account.
`logged_in_note`	Show	—

ACTIONS — IF FALSE *(guest user)*

Target field	Action	Value
`sender_name`	Show	—
`sender_name`	Required	—
`sender_email`	Show	—
`sender_email`	Required	—
`logged_in_note`	Hide	—

Example 4.2 — Member Discount by User Group

Scenario: Product order. Members of Joomla user group 8 (Gold Members) see an exclusive discount message and receive a promo code.

Step 1 — Create these form fields first

Field 1

- Type: **Free Text**
- Name: `discount_badge`
- Caption: *(empty)*
- Text: *(empty)*

Field 2

- Type: **Hidden**
- Name: `promo_code`
- Caption: *(empty)*
- Default Value: *(empty)*

To find the correct user group ID: Joomla Admin → Users → Groups. The number in the ID column is what you enter in the Value field.

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_gold_member`

CONDITION tab:

Setting	Value
Condition Type	User based
User Condition	member of user group

Setting	Value
Operator	equals
Value	8 *(your Joomla user group ID)*

ACTIONS — IF TRUE

Target field	Action	Value
`discount_badge`	Set HTML	`<strong>Gold Member: 25% discount applied.</strong>`
`discount_badge`	Show	—
`promo_code`	Set value	GOLD25

ACTIONS — IF FALSE

Target field	Action	Value
`discount_badge`	Hide	—
`promo_code`	Clear value	—

---

5. COMBINED CONDITIONS (AND / OR)

Example 5.1 — VIP + Early Registrant: Exclusive Offer

Scenario: Conference. An exclusive offer appears only if the visitor chose VIP AND is among the first 50 registrants. Both conditions must be true.

Step 1 — Create these form fields first

Field 1

- Type: **Select List**
- Name: `ticket_type`
- Caption: `Ticket Type`
- Options:

```
standard|Standard
vip|VIP
```

Field 2

- Type: **Free Text**

- Name: `exclusive_offer`
- Caption: *(empty)*
- Text: *(empty)*

Field 3

- Type: **Free Text**
- Name: `bonus_gift`
- Caption: *(empty)*
- Text: `You will receive a bonus gift at the venue.`

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_vip_early`

CONDITION tab:

Setting	Value
Condition Type	Combined
Logic operator	AND
Rule 1	Submission count — less than — 50
Rule 2	Field value — ticket_type — equals — vip

ACTIONS — IF TRUE

Target field	Action	Value
`exclusive_offer`	Set HTML	`<strong>Exclusive offer: first 50 VIP registrants only!</strong>`
`exclusive_offer`	Set class	`alert alert-success`
`exclusive_offer`	Show	—
`bonus_gift`	Show	—

ACTIONS — IF FALSE

Target field	Action	Value
`exclusive_offer`	Hide	—
`bonus_gift`	Hide	—

Example 5.2 — Free Shipping (OR Logic)

Scenario: Webshop. Free shipping applies if the order total is over €500 OR the customer is a loyalty member. Either condition alone is sufficient.

Step 1 — Create these form fields first

Field 1

- Type: **Text Box**
- Name: `order_total`
- Caption: `Order Total (€)`

Field 2

- Type: **Select List**
- Name: `loyalty_member`
- Caption: `Loyalty Member`
- Options:

```
no|No
yes|Yes
```

- Default: no

Field 3

- Type: **Free Text**
- Name: `shipping_badge`
- Caption: *(empty)*
- Text: *(empty)*

Field 4

- Type: **Hidden**
- Name: `shipping_fee`
- Caption: *(empty)*
- Default Value: `4.99`

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_free_shipping`

CONDITION tab:

Setting	Value
Condition Type	Combined
Logic operator	OR
Rule 1	Field value — order_total — greater than — 500
Rule 2	Field value — loyalty_member — equals — yes

ACTIONS — IF TRUE

Target field	Action	Value
`shipping_fee`	Set value	0.00
`shipping_badge`	Set HTML	`FREE SHIPPING`
`shipping_badge`	Set class	`alert alert-success`
`shipping_badge`	Show	—

ACTIONS — IF FALSE

Target field	Action	Value
`shipping_fee`	Set value	4.99
`shipping_badge`	Hide	—

Example 5.3 — Scholarship: Time Limit AND Capacity

Scenario: Scholarship application. The form closes when the deadline passes OR when 50 applications are received — whichever comes first. Two separate conditional fields each handle one closure trigger.

Step 1 — Create these form fields first

Field 1

- Type: **Free Text**
- Name: `closed_notice`
- Caption: *(empty)*
- Text: *(empty)*

Field 2 *(the form's submit button — RSForm! adds this automatically; use its name here)*

- Name: `submit` *(check the actual name in your form)*

Step 2 — Add two Advanced Conditional Logic fields

Conditional Field 1 — Deadline

- Name: `cond_deadline`

CONDITION tab:

Setting	Value
Condition Type	Time based
Operator	greater or equal
Value	1748649600 *(2026-05-31 00:00 UTC — use datetime picker)*

ACTIONS — IF TRUE

Target field	Action	Value
`closed_notice`	Set HTML	`Application deadline has passed.`
`closed_notice`	Show	—
`submit`	Disable	—

Conditional Field 2 — Capacity

- Name: `cond_capacity`

CONDITION tab:

Setting	Value
Condition Type	Submission count
Operator	greater or equal
Value	50

ACTIONS — IF TRUE

Target field	Action	Value
`closed_notice`	Set HTML	`All 50 positions have been filled.`
`closed_notice`	Show	—
`submit`	Disable	—

Example 5.4 — Member Early Access (Logged-in + Time Window)

Scenario: Product launch. Registered members can place orders 48 hours before the public launch. Both conditions must be true simultaneously.

Step 1 — Create these form fields first

Field 1

- Type: **Free Text**
- Name: `access_banner`
- Caption: *(empty)*
- Text: *(empty)*

Field 2

- Type: **Submit Button**
- Name: `order_button`
- Caption: `Place Order`

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_member_early_access`

CONDITION tab:

Setting	Value
Condition Type	Combined
Logic operator	AND
Rule 1	User based — is logged in
Rule 2	Time based — less than — 1748649600 *(public launch time)*

ACTIONS — IF TRUE

Target field	Action	Value
`access_banner`	Set HTML	`<strong>Member Early Access</strong> — order now, 48 hours before public launch.`
`access_banner`	Show	—
`order_button`	Show	—

ACTIONS — IF FALSE

Target field	Action	Value
`access_banner`	Hide	—
`order_button`	Hide	—

---

6. CALCULATE

Example 6.1 — Automatic VAT Calculation

Scenario: Order form. The total price including 27% VAT is calculated automatically from quantity and unit price.

Step 1 — Create these form fields first

Field 1

- Type: **Text Box**
- Name: `quantity`
- Caption: `Quantity`
- Default Value: `1`

Field 2

- Type: **Text Box**
- Name: `unit_price`
- Caption: `Unit Price (€)`
- Read Only: Yes *(set by another condition or manually)*

Field 3

- Type: **Text Box**
- Name: `total_price`
- Caption: `Total Price incl. VAT (€)`
- Read Only: Yes

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_calc_total`

CONDITION tab:

Setting	Value
Condition Type	Submission count
Operator	greater or equal
Value	0 *(always true — runs on every page load)*

ADVANCED tab:

Setting	Value
Real-time Update	Yes *(recalculates as the user types)*

ACTIONS — IF TRUE

Target field	Action	Value
`total_price`	Calculate	`{quantity} * {unit_price} * 1.27`

Reference other fields using `{fieldname}`. Supported operators: `+`, `-`, `*`, `/`, `(`, `)`. The plugin uses a safe custom math parser — `eval()` is never called.

---

7. SET HTML AND CSS CLASSES

Example 7.1 — Styled Delivery Info Block

Scenario: Order form. A styled info block shows estimated delivery time and changes appearance based on the selected delivery method.

Step 1 — Create these form fields first

Field 1

- Type: **Select List**
- Name: `delivery_method`
- Caption: `Delivery Method`
- Options:

```
standard|Standard
express|Express
same_day|Same Day
```

Field 2

- Type: **Free Text**
- Name: `delivery_info`
- Caption: *(empty)*
- Text: *(empty)*

Step 2 — Add the Advanced Conditional Logic field

- Name: `cond_delivery_info`

CONDITION tab:

Setting	Value
Condition Type	Field value
Field	delivery_method
Operator	equals
Value	express *(select from dropdown)*

ACTIONS — IF TRUE

Target field	Action	Value
`delivery_info`	Set HTML	`<strong>Express Delivery</strong>  Next business day · Order before 2 PM.`
`delivery_info`	Set class	`alert alert-warning`
`delivery_info`	Show	—

ACTIONS — IF FALSE

Target field	Action	Value
`delivery_info`	Set HTML	`<strong>Standard Delivery</strong>  3–5 business days · Free over €50.`
`delivery_info`	Set class	`alert alert-info`
`delivery_info`	Show	—

`set_class` vs `add_class`: Use `set_class` to replace the colour variant entirely (e.g. switch from `alert-info` to `alert-warning`). Use `add_class` only to append an extra modifier without removing existing classes.

`set_html` note: `set_html` only works on **Free Text** and **Preview** fields — display-only

elements. It replaces the inner HTML with your content, allowing bold, line breaks, and links. It has no effect on input fields.

Quick Reference

Operator Reference

Operator	Meaning	Use with
`equals`	Exact match	Text, select, radio
`not_equals`	Does not match	Text, select, radio
`greater_than`	Numerically greater	Numbers, timestamps
`less_than`	Numerically less	Numbers, timestamps
`greater_or_equal`	Greater or equal	Numbers, timestamps
`less_or_equal`	Less or equal	Numbers, timestamps
`contains`	String contains value	Checkbox (multi-value), text
`not_contains`	String does not contain	Checkbox, text
`is_empty`	Value is blank	Any field
`is_not_empty`	Value is not blank	Any field
`matches_regex`	Matches regex pattern	Text, textarea
`not_matches_regex`	Does not match pattern	Text, textarea

Action Reference

Action	Available on	Effect
`show` / `hide`	All fields	Toggles visibility
`enable` / `disable`	Input fields, buttons	Toggles interactivity
`required` / `optional`	Input fields	Toggles validation
`set_value`	Input fields	Sets the value
`calculate`	Text Box	Sets value from math expression
`clear_value`	Input fields	Empties the value
`set_html`	Free Text, Preview	Replaces inner HTML
`set_class`	All fields	Replaces all CSS classes
`add_class`	All fields	Appends a CSS class
`remove_class`	All fields	Removes one CSS class
`add_option`	Select, Radio, Checkbox	Adds a new option
`remove_option`	Select, Radio, Checkbox	Removes an option by value

Action	Available on	Effect
`show_option` / `hide_option`	Select, Radio, Checkbox	Shows or hides one option
`enable_option` / `disable_option`	Select, Radio, Checkbox	Enables or disables one option
`check_option` / `uncheck_option`	Checkbox	Checks or unchecks one option
`select_option`	Select, Radio	Selects one option, clears others

© 2026 László Blahó — extension@blaho.hu — https://extensions.blaho.hu