Abilities in the real world

How to make your plugin usable by the assistants your clients already use.



Mark Jansen

[WCNL]

12 November 2026



The attendees of WCNL 2025 · Photo: Jeroen Rotty

Within two years, your clients will manage their WordPress site solely from their AI assistant.



Hi, I’m Mark

Messing with the web since 1998, getting paid for it since 2011, and lately building with the Abilities API for the agentic web.

Grill enthusiast, drummer in Headline and klootschieter (yes, ask me).







Mark Jansen

Freelance WordPress and Laravel developer

There’s this new big thing

You might have heard about it already.

"It's huge."

"More profound than electricity or fire."

"The most disruptive force in history."

"The iPhone moment."

"As fundamental as the creation of the microprocessor."

Artificial intelligence

For us normies



I don’t think it’s intelligent.

When I say AI, I mean the assistant you already use

Claude

ChatGPT

Gemini

Copilot

Le Chat

Perplexity

Grok

Cursor

How most people use it

The assistant gives advice, and you carry it over to the site by hand.



Taking WordPress into the agentic era

If WordPress falls behind, people move on.

Eighteen months

May 2025
AI Team
Automattic, Google, 10up

July 2025
AI Building Blocks
Canonical first, Core when ready

December 2025
WordPress 6.9
Abilities API in core

May 2026
WordPress 7.0
WP AI Client in core

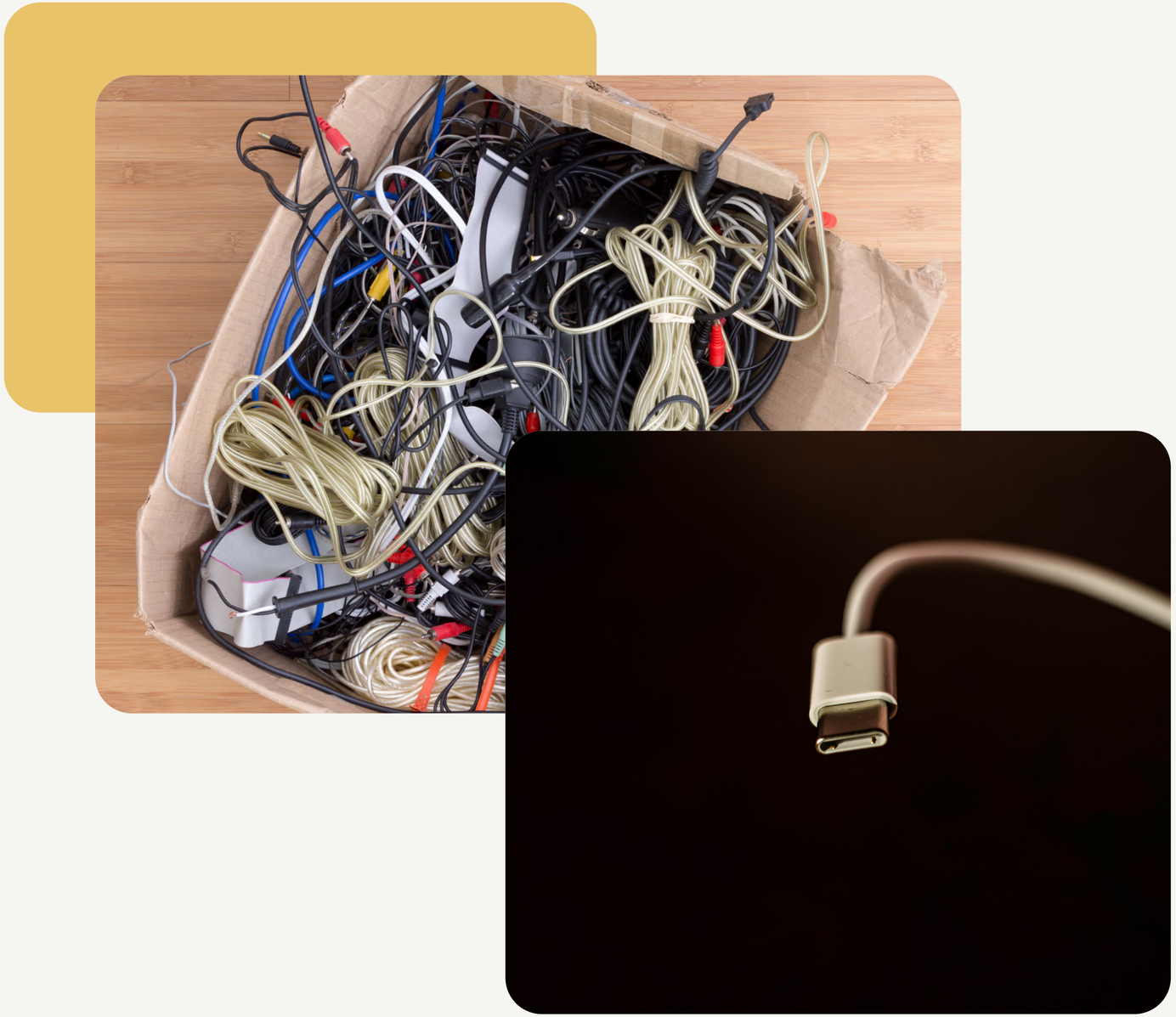
August 2026
WordPress 7.1
Lifecycle hooks, and a setting to share via MCP

October 2026
MCP Adapter
Canonical plugin on WordPress.org

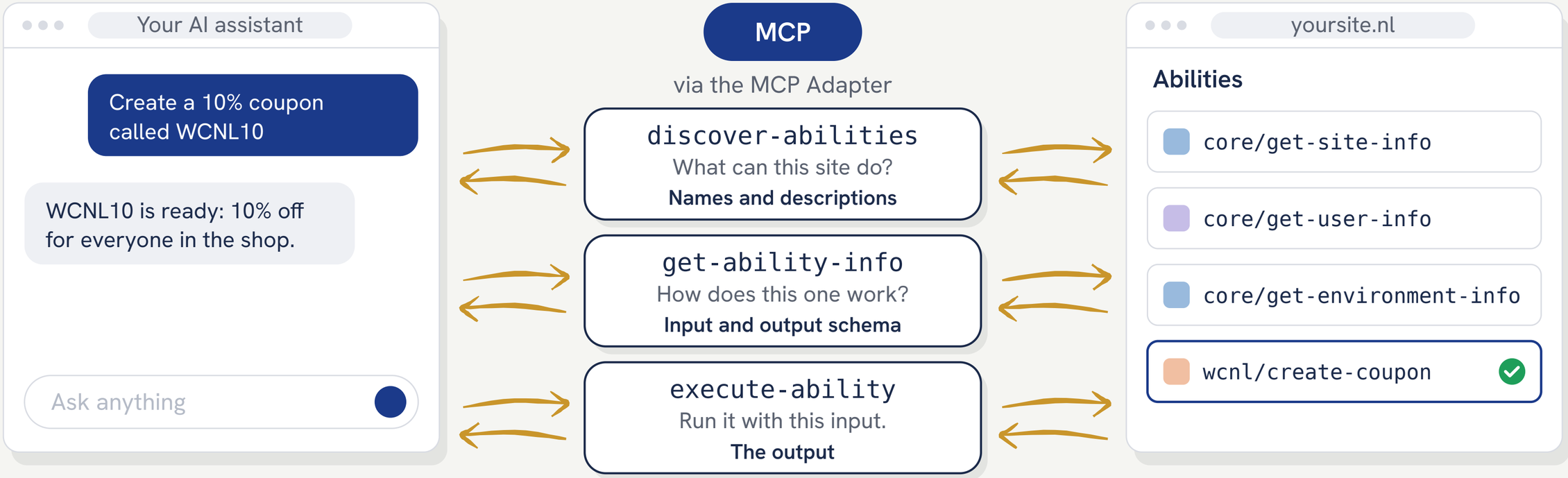
December 2026
WordPress 7.2
Secrets API, more core abilities

MCP: the USB-C of AI

One plug for every assistant. The MCP Adapter, an official WordPress.org plugin, turns abilities into MCP tools.



How does it work?



The Abilities API

1. Abilities explained
2. Creating your own
3. Getting your clients ready

Goedemiddag, Mark

Waarmee kan ik je vandaag helpen?

Project of map

Opus 5.5 Gemiddeld

Automatisch

Recente artefacten



Part 1

Abilities explained

What is an ability?

A named, self-describing unit of functionality with input, output and permissions.

`wcnl/create-coupon`

What it does	Creates a WooCommerce coupon with a percentage discount. Use when someone asks for a new discount code.
What it needs	code: text, at least 3 characters percent: a whole number from 1 to 100
What it returns	The id and code of the new coupon
Who can use it	Shop managers

How is it different from REST or WP-CLI?

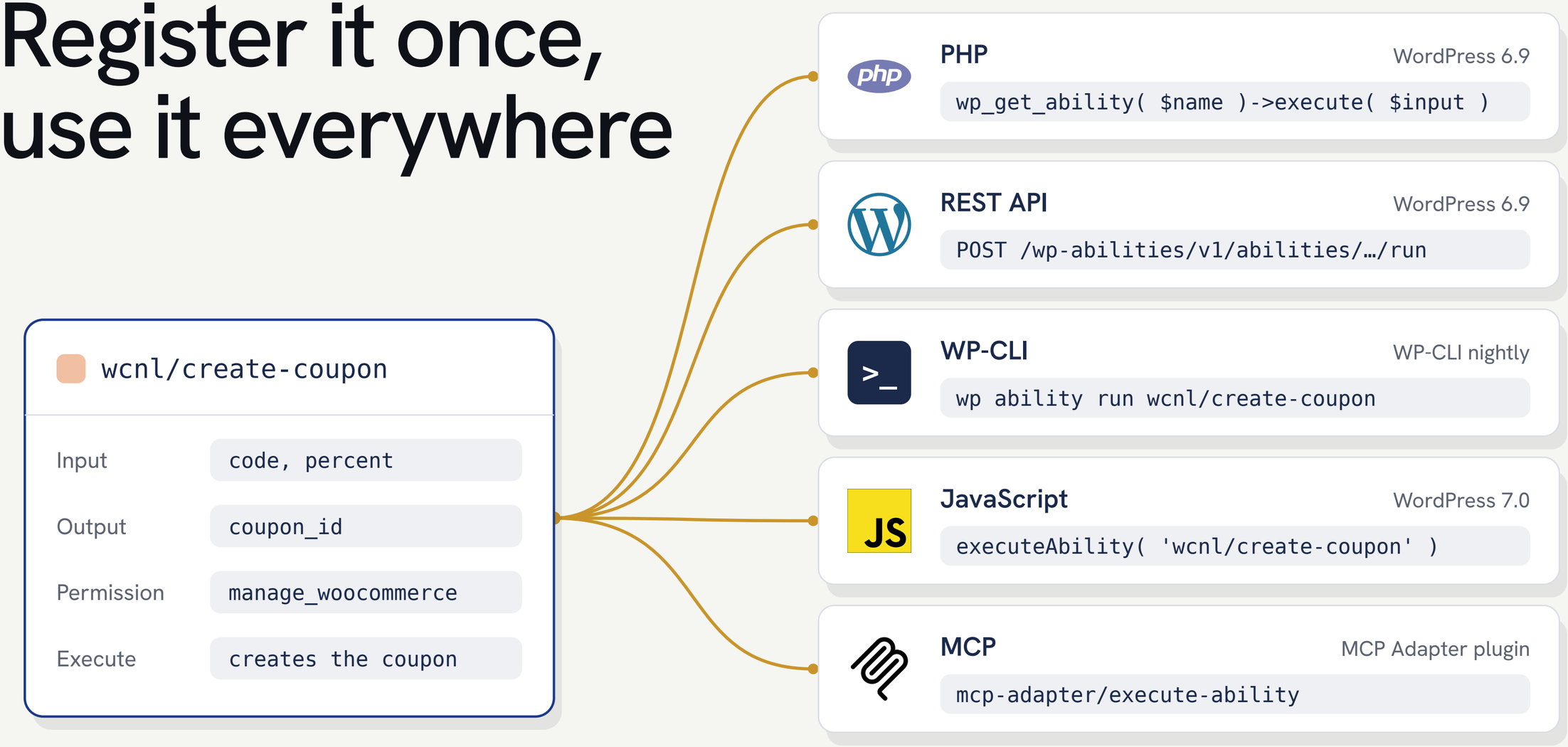
	REST endpoint	WP-CLI command	Ability
Input	Checked if you add args	Synopsis only	Enforced by the schema
Output	Not checked	Not checked	Enforced by the schema
Permission	A notice if you forget it	Up to your code	No registration without it
Other channels	Build them yourself	Build them yourself	Come with the registration
An assistant	Needs a wrapper	Needs shell access	Uses it directly via MCP

Abilities aren’t for you.

They’re for your users, clients and customers.

You write them, your clients use them.

Register it once, use it everywhere





Part 2

Creating your own

First, a category

Required, and it has to exist before the ability does.

wcnl-abilities.php

```php
add_action( 'wp_abilities_api_categories_init', function () {
  wp_register_ability_category( 'wcnl', [
    'label'       => 'WordCamp Nederland demo',
    'description' => 'Abilities from the WCNL 2026 demo plugin.',
  ] );
} );
```

wcnl/create-coupon, line by line

wcnl-abilities.php

```php
add_action( 'wp_abilities_api_init', function () {
  wp_register_ability( 'wcnl/create-coupon', [
    'label'       => 'Create coupon',
    'description' => 'Creates a WooCommerce coupon with a percentage discount. '
                   . 'Use when someone asks for a new discount code.',
    'category'    => 'wcnl',
    'input_schema'  => [
      'type'       => 'object',
      'properties' => [
        'code'    => [ 'type' => 'string', 'minLength' => 3 ],
        'percent' => [ 'type' => 'integer', 'minimum' => 1, 'maximum' => 100 ],
      ],
      'required'             => [ 'code', 'percent' ],
      'additionalProperties' => false,
    ],
    'output_schema' => [
      'type'       => 'object',
      'properties' => [ 'id' => [ 'type' => 'integer' ], 'code' => [ 'type' => 'string' ] ],
    ],
```

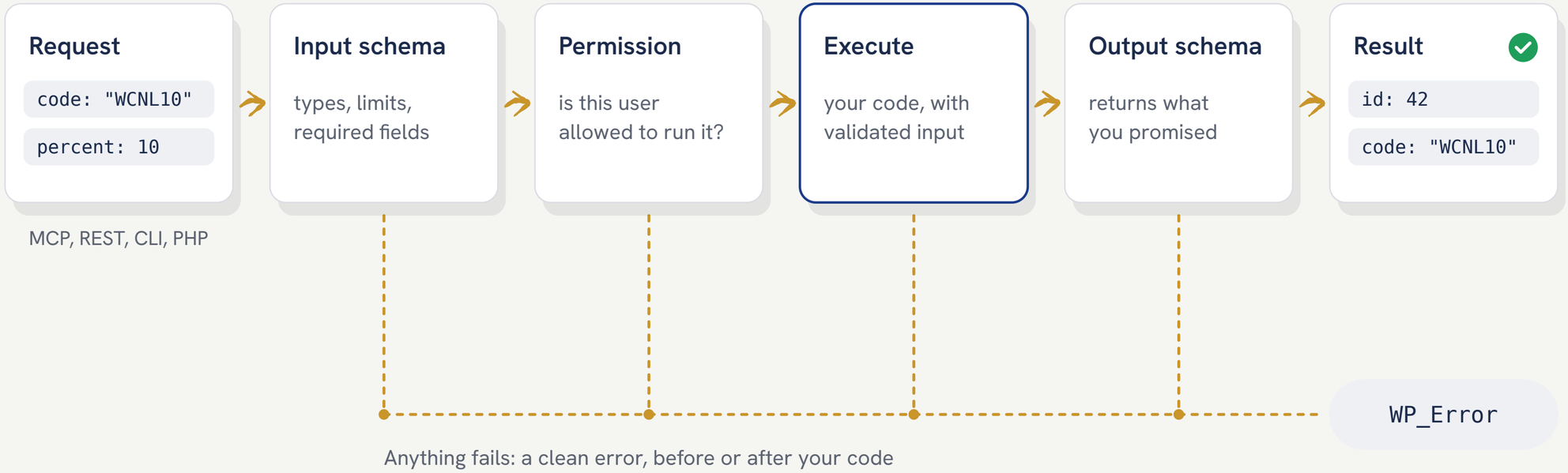
wcnl/create-coupon, line by line

wcnl-abilities.php

```
    'permission_callback' => fn( $input ) => current_user_can( 'manage_woocommerce' ),
    'execute_callback'    => function ( $input ) {
      if ( wc_get_coupon_id_by_code( $input['code'] ) ) {
        return new WP_Error( 'coupon_exists', 'That code is already in use.' );
      }
      $coupon = new WC_Coupon();
      $coupon->set_code( $input['code'] );
      $coupon->set_discount_type( 'percent' );
      $coupon->set_amount( $input['percent'] );
      $coupon->save();
      return [ 'id' => $coupon->get_id(), 'code' => $coupon->get_code() ];
    },
    'meta' => [
      'annotations' => [ 'readonly' => false, 'destructive' => false, 'idempotent' => false ],
      'public'      => true, // 6.9 and 7.0: 'show_in_rest' => true, 'mcp' => [ 'public' => true ]
    ],
  ] );
} );
```

What WordPress does for you

Every failure comes back as a `WP_Error`, your callback only ever sees validated data, and over REST the same chain runs.



Nine hooks, in order

Start	`wp_ability_invoked`	action	Audit log, including failed attempts
	`wp_pre_execute_ability`	filter	Caching, rate limits, approval, test mocks
Input	`wp_ability_normalize_input`	filter	Defaults, caller context
	`wp_ability_validate_input`	filter	Site rules: discount never above 30%
Permission	`wp_ability_permission_result`	filter	Extra rules per site or role
Execute	`wp_before_execute_ability`	action	Start a timer, log context
	`execute_callback`	your code	What the ability does
	`wp_ability_execute_result`	filter	Strip fields, mask personal data
Output	`wp_ability_validate_output`	filter	Your own checks after the schema
Done	`wp_after_execute_ability`	action	Notifications, metrics, Slack

Test it without a model

The CLI runs the same chain as the assistant, without the guesswork.

Not in a stable WP-CLI release yet: run `wp cli update --nightly`

terminal

```
wp ability list
wp ability get wcnl/create-coupon
wp ability run wcnl/create-coupon \
  --input='{"code":"WCNL10","percent":10}' \
  --user=ai-editor
```

Design rules

From building them myself, and from WordPress’s own agent skills.

- One read ability per question, with filters in the input schema
- One write ability per change, done in a single call
- If you can’t describe it in one sentence, it’s two abilities
- Return names and context, and keep lists short
- Errors that tell the assistant what to fix
- No personal data in or out: no email addresses or phone numbers

Part 3

Getting your clients ready

Who’s already in

The more plugins that use Abilities, the more powerful WordPress becomes. **Jason Adams** WordPress AI Team, November 2025

WooCommerce

Yoast

Elementor

ACF

Gravity Forms

WPForms

WP Rocket

Akismet

WP Mail SMTP
a wpforms product

Easy Digital Downloads

Bricks

MainWP

Search & Filter

Koko Analytics

Albert

And more (November 2026)

What your clients will ask

“Clear the cache for the homepage.”

`site/clear-cache`

“Redirect /old-campaign to the homepage.”

`seo/add-redirect`

“Send order 1234 to the ERP again.”

`shop/resync-order`

“Change our opening hours for Christmas.”

`site/set-opening-hours`

Do it safely

An assistant works as a logged-in user, with everything that user is allowed to do.

- A separate user with a limited role
- Start with read-only abilities, add writes later
- Mark destructive abilities as destructive
- Never send personal data to the model

What’s next

Aimed at WordPress 7.3, with foundations in 7.2.

1. More abilities in core
2. MCP built into core
3. Connect an assistant by logging in (OAuth)
4. Abilities in the Command Palette

Bonus

Five minutes of live coding.

Within two years, your clients will manage their WordPress site solely from their AI assistant.

Let's make sure their site is ready.

Monday morning

1. Install the MCP Adapter, and Albert (shameless plug)
2. Try it on a local or staging site
3. Register your own ability, read-only first
4. Build on it and make something great

Sources

"Just tell WordPress what your plugin can do, and let the agent handle the rest."

Nik McLaughlin

- **Abilities API**
  developer.wordpress.org/apis/abilities-api
- **MCP Adapter**
  wordpress.org/plugins/mcp-adapter
- **WordPress AI Team**
  make.wordpress.org/ai
- **Nik McLaughlin, The Importance of the Abilities API & Abilities API for Humans (WCUS 2026)**
  keepwebsitesweird.pika.page
- **PR #1168, Plugin Directory: Detect the abilities a plugin registers**
  github.com/WordPress/wordpress.org/pull/1168
- **Albert, Connect AI Assistants to WordPress**
  wordpress.org/plugins/albert-ai-butler

Thanks



Questions? Come and find me.

yourmark.nl

Slides: yourmark.nl/wcnl2026

LinkedIn
Mark Jansen

(Twitter)
@mark_jansen