Smart Watering with Rachio – Control4 Driver

The **Smart Watering with Rachio** driver enables Control4 to integrate seamlessly with Rachio irrigation controllers and Rachio Smart Hose Timers.

Updating

To ensure customers have the latest updates including bug fixes, we recommend you enable the Driver Central **Automatic Updates** option. If you choose to update the driver manually, add the correct driver files to Composer Pro following the **Installing the Driver in Composer Pro** steps below.

> ⚠️ As of version 1.4.0 (build 104000), the driver filenames have been changed. This affects the driver update process for existing users.

The driver filenames have been updated to:

- `smart_watering.c4z`
- `smart_watering_valve.c4z`

In previous versions, the filenames were:

- `rachio_smart_watering.c4z`
- `rachio_smart_hose_timer.c4z`

To maintain backward compatibility, existing installations using the legacy filenames will continue to receive updates via the Driver Central automatic update service until version 2.0.0 is released.

However, the driver download package now includes only the updated filenames to ensure new installations use the current naming convention and avoid confusion.

Configuration

Prerequisites

Before installing the driver:

1. Ensure the customer has an account at https://app.rach.io or via the mobile app.
2. Rachio devices (e.g., irrigation controllers or Smart Hose Timers) must already be added to the customer's Rachio account.
3. Obtain the Rachio API Key:
   - Via https://app.rach.io: Go to **Account Settings**, then click **GET API KEY**.
   - In the mobile app: Tap the **Profile** icon, then select **API Key Copy**.

> 💡 For the most up-to-date API key instructions, refer to Rachio’s support article.

Installing the Driver in Composer Pro

1. Unzip the driver bundle on the computer running Composer Pro.
2. In Composer Pro, go to **Driver > Add or Update Driver or Agent**.
3. Add the `rachio_smart_watering.c4z` driver first.
4. Repeat the process to add the `smart_watering_valve.c4z` driver.

> ⚠️ **Both drivers must be added in Composer Pro before proceeding.** The Smart Watering Valve driver is used for both sprinkler valves and Smart Hose Timer valves.

Initial Setup

1. In the **System Design** view, search for **Smart Watering with Rachio**.
2. Drag and drop **Smart Watering with Rachio** to add the driver to the project.
3. In the **Properties** tab, enter the API key in the `API Key` field and click **Set**.

When configured correctly:

- `API Status` will show **Last API request was successful**.
- `User ID` will be populated.

Adding Devices

To create Control4 devices for each Rachio sprinkler zone and Smart Hose Timer:

1. In the **Actions** property, choose **Add Devices** and click **Set**.

> ⏱ Device creation may take a minute or two depending on the number of devices in the Rachio account.

- A corresponding driver for each active sprinkler zone and Smart Hose Timer will be created in the current room.
- Devices are automatically connected to the appropriate bindings.
- Disabled zones will not be added.
- You may move devices to other rooms in Composer as needed.

To add newly enabled zones or newly added devices later, simply run **Add Devices** again.

Manually Adding Devices (Optional)

If you'd prefer to manually add and connect devices:

1. In the **Actions** tab, run **Create Bindings**.
2. Add a **Smart Watering Valve** driver to the project.
3. In the **Connections** view, link one of the proxy bindings from the Rachio driver to the new valve device.
4. Repeat for each desired sprinkler zone or Smart Hose Timer.

Using the Smart Watering Valve

Refer to the documentation included with the **Smart Watering Valve** driver for usage details, including how to integrate with Control4 Navigator and keypad buttons.

Checking Hub Status (for Smart Hose Timers)

The Rachio Smart Hose Timer uses a hub to communicate with the cloud. To check the hub status:

1. In the **Actions** tab, select **Check Smart Hose Timer Hub Status** and click **Set**.
2. Use the `Hub Status` dropdown to view a list of hubs with their serial numbers and current status (**Online** or **Offline**).

The `Hub Status Last Updated` property shows the timestamp of the last manual or automatic hub status check.

Driver Properties

Property	Description
`Cloud Status`	Shows if the driver license is activated, expired, or if updates are available.
`Automatic Updates`	Automatically installs updates when set to **On**.
`Driver Version`	Displays the current driver version.
`API Status`	Status of the last API call (e.g., success or HTTP error).
`API Key`	Rachio API key for the customer's account.
`Actions`	Driver configuration and status update actions.
`Log Level`	Logging verbosity: **Debug** for detailed info, **Trace** for raw HTTP data.
`Poll Time Seconds`	Interval (in seconds) between external state checks.
`User ID`	Rachio User ID used for identifying API requests.
`Hub Status`	List of Smart Hose Timer hubs and their statuses (Online/Offline).
`Hub Status Last Updated`	Timestamp of the last hub status check.
`Main API Limit`	Remaining API calls for standard Rachio controllers (daily limit: 3,500).
`Smart Hose API Limit`	Remaining API calls for Smart Hose Timers (daily limit: 3,500).

Programming Integration

Events

- **When a Rachio Hub goes offline**
- **When a Rachio Hub comes back online**
- **When `MAIN_API_REQUESTS_REMAINING` changes**
- **When `SMART_HOSE_API_REQUESTS_REMAINING` changes**

Commands

- **Check Hub Status –** Useful for triggering regular hub checks via scheduled programming.

Variables

These variables can be monitored to prevent exceeding Rachio API call limits:

- `MAIN_API_REQUESTS_REMAINING`: Remaining API calls for standard irrigation devices.
- `SMART_HOSE_API_REQUESTS_REMAINING`: Remaining API calls for Smart Hose Timers.

Conditionals

- **If `MAIN_API_REQUESTS_REMAINING` is < 100**
- **If `SMART_HOSE_API_REQUESTS_REMAINING` is < 100**

Support

If you encounter any issues:

1. Contact Driver Central Support.
2. If the issue cannot be resolved, it will be escalated to **Blessing Innovations LLC**.

For feature requests or bug reports, please open an issue on GitLab: 👉 Submit an Issue