Uchuva ArcGIS MCP — Documentation

Compiled from repository sources. Generated: 2026-05-10

This document bundles the user-facing and technical documentation for the MCP server, REST client, Windows installer, ArcPy bridge, and ArcGIS Pro toolbox.

Project overview

Source: `README.md`

Uchuva ArcGIS Integration

Pieces for **ArcGIS Portal / REST**, **ArcGIS Pro / ArcPy**, and an **MCP server** for Claude and Cursor.

Layout

Folder	Contents
`arcgis-core`	Portal / feature layer REST client, auth helpers (`@uchuva/arcgis-core`).
`mcp-arcgis`	MCP stdio server: tools such as `portal_search_items`, `query_features`, optional edits, ArcPy bridge.
`arcpy-bridge`	Python scripts run with ArcGIS Pro's Python.
`pro-addin`	Python toolbox (`.pyt`) to export project context.
`docs`	Install notes, MCP contracts, Cursor/Claude configuration.

Quick start

```
npm install
npm run build
npm run test
```

Configure MCP per `docs/cursor-claude-mcp.md` and `docs/INSTALL.md`.

Windows installer (.exe)

For a bundled executable (no `node` on the target PC):

```
npm install
npm run installer:windows
```

For a **graphical Setup.exe** (Inno Setup), after installing Inno:

```
npm run installer:gui
```

Details: `installer/README.md` and `installer/setup/README-GUI_INSTALLER.md`.

Esri licensing

`arcpy`, ArcGIS Pro, and Esri services remain under their own licenses; this repo is an OSS integration shell and does not ship Esri proprietary binaries.

Installation

Source: `docs/INSTALL.md`

Setup

```
cd Uchuva_ArcGIS
npm install
npm run build
```

Windows .exe

```
npm install
npm run installer:windows
```

Produces `installer/out/UchuvaArcGIS-Mcp.exe` and copies `arcpy-bridge` next to it. In Cursor/Claude use that path as the MCP `command` (no `node`). See `installer/README.md`.

Graphical installer (Inno): with Inno 6 installed, `npm run installer:gui` builds the Setup automatically; see `installer/setup/README-GUI_INSTALLER.md`.

MCP server (`mcp-arcgis`)

Runnable as: `node mcp-arcgis/dist/cli.js` (or the `mcp-arcgis` bin after `npm run build`).

Environment variables

Variable	Description
`ARCHGIS_PORTAL_URL`	Portal root URL (e.g. `https://www.arcgis.com` or Enterprise). No trailing slash.
`ARCHGIS_TOKEN`	Optional portal token (preferred over username/password when set).
`ARCHGIS_USERNAME` / `ARCHGIS_PASSWORD`	Token generation via `/sharing/rest/generateToken`.
`ARCHGIS_CAPABILITY_READ`	If `false`, disables read tools. Default: reads enabled.
`ARCHGIS_CAPABILITY_WRITE`	Must be `true` to apply edits or mutating GP without `dryRun`. Default: `false`.
`ARCHGIS_GP_ALLOWLIST`	Comma/semicolon-separated URL prefixes allowed for `run_geoprocessing`.
`ARCHGIS_AUDIT_LOG`	Path to NDJSON audit log for writes/GP.
`ARCHGIS_PRO_PYTHON`	Absolute path to ArcGIS Pro `python.exe` (ArcPy bridge).
`ARCHGIS_QUERY_MAX`	Max page size for `query_features` (default `5000`).
`ARCHGIS_CONTRACTS_OVERRIDE`	Optional path to a `.md` file replacing the embedded MCP contracts resource.

See `.env.example` at the repo root.

ArcPy bridge

`arcpy-bridge/README.md` — scripts must run with ArcGIS Pro's Python.

ArcGIS Pro toolbox

`pro-addin/uchuva_context_tools.pyt` — register the folder in ArcGIS Pro or copy into your project.

Cursor and Claude (MCP configuration)

Source: `docs/cursor-claude-mcp.md`

Integration uses the **stdio MCP server** (`mcp-arcgis`). It exposes tools for ArcGIS Portal/Enterprise REST and optional ArcPy via ArcGIS Pro's Python.

Option A: Windows .exe (easiest to ship)

After `npm run installer:windows` or the Inno installer, use **`UchuvaArcGIS-Mcp.exe`**. Keep **`arcpy-bridge`** beside the exe. In MCP config set `command` to that path (no `node`).

Full template: installer/setup/cursor-mcp.example.json — only change the absolute `command` path (use `/` in JSON paths).

Option B: Node from source

After `npm run build`, entrypoint is `mcp-arcgis/dist/cli.js` with `command: "node"` and the path in `args`.

Where Cursor reads MCP

Cursor merges MCP from:

Scope	Typical path (Windows)
User	`%USERPROFILE%\.cursor\mcp.json`
Project	`<repo>\.cursor\mcp.json`

The root JSON must be a single object: **`{ "mcpServers": { ... } }`**. Pasting a second nested `mcpServers` block or invalid JSON breaks MCP.

Example (development with Node)

```
{
  "mcpServers": {
    "uchuva-arcgis": {
      "command": "node",
      "args": ["C:/path/to/Uchuva_ArcGIS/mcp-arcgis/dist/cli.js"],
      "env": {
        "ARCHGIS_PORTAL_URL": "https://www.arcgis.com"
      }
    }
  }
}
```

Do not leave placeholder values like `"<optional>"` in real configs; omit keys or use real values.

After TypeScript changes, run `npm run build` in the repo.

Project rules (`.cursor/rules`) can require `confirm` for mutating tools.

If MCP “breaks” Cursor or shows errors

Common causes on Windows:

1. **Invalid JSON** — bad quotes/commas, duplicate `mcpServers`, or single `\` in paths (use `\\` or `/`).
2. **Wrong path** to the exe or `cli.js`.
3. **SmartScreen** blocking the first run of an unsigned exe.

Recovery: quit Cursor, rename `mcp.json` to `mcp.json.bak` (user and/or project), restart Cursor, recreate from `installer/setup/cursor-mcp.example.json`.

Claude Desktop

Windows graphical installer: if you leave the **“Register MCP in Claude Desktop”** task enabled, the installer merges the `uchuva-arcgis` entry into `%AppData%\Claude\claude_desktop_config.json` (with a backup of the previous file). Fully quit Claude from the tray and reopen it.

Manual: same JSON shape as below in `claude_desktop_config.json` per Anthropic MCP docs.

Contracts resource

The server exposes `uchuva://docs/contracts-tools` (Markdown embedded in the binary, or override with `ARCHGIS_CONTRACTS_OVERRIDE`).

MCP tool contracts

Source: `docs/contracts/tools.md`

Uchuva ArcGIS MCP tool contracts

All tools return structured JSON. Write operations (`create_feature`, `update_feature`, `run_geoprocessing`) require explicit `confirm: true` plus the `write` capability enabled via environment variables.

Capability environment variables

Variable	Role
`ARCHGIS_CAPABILITY_READ`	Defaults to enabled. If `false`, read tools do not execute.
`ARCHGIS_CAPABILITY_WRITE`	Defaults to `false`. Must be `true` for `create_*`, `update_*`, and mutating GP.

Portal

`portal_health`

Input: `{}`

Output: `{ portalUrl, isPortal, ssl, currentVersion?, authConfigured, tokenExpiryIso? }`

`portal_search_items`

Search Portal items (REST `/sharing/rest/search`).

Input:

```
{
  "query": "type:\"Feature Service\" owner:someuser",
  "num": 20,
  "start": 1,
  "sortField": "modified",
  "sortOrder": "desc"
}
```

Output: `{ total?, nextStart?, results[] }`

`describe_layer`

Input:

```
{
  "serviceUrl": "https://services.../arcgis/rest/services/name/FeatureServer",
  "layerId": 0,
  "includeExtent": true
}
```

Output: Layer metadata (fields, geometry types, `capabilities`, optional summarized stats).

`query_features`

Paged query. Hard limit: `maxRecordCount` parameter (default 500; max 5000 with care).

Input:

```
{
  "layerUrl": "https://.../FeatureServer/0",
  "where": "1=1",
  "outFields": "OBJECTID,NAME",
  "returnGeometry": false,
  "resultRecordCount": 100,
  "resultOffset": 0,
  "orderByFields": "",
  "returnCountOnly": false
}
```

Output: Standard feature layer `query` response (GeoJSON / features as returned by the service).

`summarize_layer_fields`

Field statistics via REST `/query` and `outStatistics` when supported.

Input:

```
{
  "layerUrl": "...",
  "where": "1=1",
  "statistics": [{ "statisticType": "count", "onStatisticField": "OBJECTID", "outStatisticFieldName": "cour
}
```

`create_feature`

Input:

```
{
  "layerUrl": "...",
  "confirm": false,
  "dryRun": true,
  "features": [{ "geometry": {...}, "attributes": {...} }]
}
```

`dryRun: true` validates only; `confirm: true` with `dryRun: false` applies when `ARCHGIS_CAPABILITY_WRITE=true`.

`update_feature`

Same pattern as create; uses an `updates` array.

`run_geoprocessing`

Only `taskEndpointUrl` values allowed by CSV prefix list `ARCHGIS_GP_ALLOWLIST`.

MCP input:

```
{
  "confirm": false,
  "dryRun": true,
  "taskEndpointUrl": "https://.../GPServer/TaskName/submitJob",
  "inputs": { "Input_Features": "...", "distance": "10" }
}
```

`inputs` must be string values (complex values as JSON strings if the service expects that). With `dryRun: true`, `ARCHGIS_CAPABILITY_WRITE` is not required.

`execute_arcpy_script`

Runs an allowlisted script using ArcGIS Pro's Python (`ARCHGIS_PRO_PYTHON`). Does not run arbitrary code.

Input:

```
{
  "script": "describe_project",
  "args": {}
}
```

See `arcpy-bridge/README.md`.

Windows MCP bundle (installer)

Source: `installer/README.md`

Windows MCP bundle (executable)

Produces **`installer/out/UchuvaArcGIS-Mcp.exe`** with a bundled Node runtime so Cursor / Claude Desktop users do not need `node` on PATH on the target machine.

Requirements (build machine only)

- Windows
- Node.js 20+ and `npm`

Create the .exe

From the repository root:

```
npm install
npm run installer:windows
```

This generates **`installer/out/UchuvaArcGIS-Mcp.exe`** and **`installer/out/arcpy-bridge/`** (scripts must stay **next to** the .exe).

Graphical installer (Setup wizard)

After installing Inno Setup 6:

```
npm run installer:gui
```

That packages the MCP and compiles `setup/UchuvaArcGIS-Mcp.iss`. Details: `setup/README-GUI_INSTALLER.md`. Output: **`dist-setup/`** with a versioned name like `UchuvaArcGIS-Mcp-Setup-0.1.0.1.exe` (`installer/setup-version.json` bumps after each successful compile).

The Setup wizard can **merge** the `uchuva-arcgis` entry into Claude Desktop's `%AppData%\Claude\claude_desktop_config.json` (optional task, on by default). Uninstall removes that stanza when it still points at the installed exe.

Cursor / Claude configuration

Use the exe as the MCP `command` (no `node`):

```
"mcpServers": {
  "uchuva-arcgis": {
    "command": "C:/full/path/.../installer/out/UchuvaArcGIS-Mcp.exe",
    "args": [],
    "env": {
      "ARCHGIS_PORTAL_URL": "https://www.arcgis.com"
    }
  }
}
```

Use docs/cursor-claude-mcp.md for portal URLs and tokens.

If MCP errors appear after editing JSON, see **Recovery** in docs/cursor-claude-mcp.md.

Limitations vs. running from source

- Windows x64 oriented.
- ArcPy still needs `ARCHGIS_PRO_PYTHON` for `execute_arcpy_script`; ArcGIS Pro is not bundled.
- Optional: `ARCHGIS_CONTRACTS_OVERRIDE` path to a custom contracts `.md` file.

Quick alternative (no bundled exe)

Run **`node …/mcp-arcgis/dist/cli.js`** as documented in the main README.

Graphical installer (Inno Setup)

Source: `installer/setup/README-GUI_INSTALLER.md`

Graphical installer (Inno Setup)

Builds a **Setup.exe** wizard (Next / Finish) using Inno Setup 6.

Steps

1) Generate `installer/out/`

From the repository root:

```
npm install
npm run installer:windows
```

2) Build the graphical installer (automatic)

```
npm run installer:gui
```

That runs `installer:windows` and then finds **ISCC.exe** (Inno Setup 6) from typical paths, `where ISCC`, or environment variables:

- `ISCC` — full path to `ISCC.exe`
- `INNO_SETUP_DIR` — folder containing Inno (`INNO_SETUP_DIR\ISCC.exe`)

2b) Build manually with Inno GUI

Option A — Inno GUI: open `installer/setup/UchuvaArcGIS-Mcp.iss` in Inno Setup Compiler and **Build → Compile**.

Option B — command line (adjust path to `ISCC.exe`):

```
cd installer\setup
& "C:\Program Files (x86)\Inno Setup 6\ISCC.exe" UchuvaArcGIS-Mcp.iss
```

3) Output

`npm run installer:gui` bumps the **build** number in `installer/setup-version.json` only after a **successful** compile. Version format: `major.minor.patch.build` (e.g. `0.1.0.7`).

The Setup file is written to **`dist-setup/UchuvaArcGIS-Mcp-Setup-<version>.exe`** (for example `dist-setup/UchuvaArcGIS-Mcp-Setup-0.1.0.7.exe`).

Compiling `UchuvaArcGIS-Mcp.iss` from Inno IDE or `ISCC` **without** `/D` uses the script defaults (`0.1.0.0` and base name without a suffix).

What the Setup does

- Per-user install (no admin): `%LocalAppData%\Programs\Uchuva\ArcGIS-MCP\`
- Copies `UchuvaArcGIS-Mcp.exe` and `arcpy-bridge\`
- Optional desktop shortcut
- Optional post-install: open `README_CURSOR.txt`

Enterprise alternatives

- **WiX** + signed MSI
- **Advanced Installer** / **InstallShield** (commercial)

ArcPy bridge

Source: `arcpy-bridge/README.md`

ArcPy bridge (ArcGIS Pro)

These scripts **`import arcpy`** and must be run with **ArcGIS Pro's** Python environment, not a generic Python install.

Configuration

Set **`ARCHGIS_PRO_PYTHON`** to the `python.exe` of the `arcgispro-py3` environment.

Example on Windows:

```
C:\Program Files\ArcGIS\Pro\bin\Python\envs\arcgispro-py3\python.exe
```

From MCP, the `execute_arcpy_script` tool only allows known script names (allowlist).

Scripts

Script	args	Output
`describe_project`	`project_path` (`.aprx`)	Lists map names
`list_layouts`	`project_path`	Lists layout names

Each script prints a single JSON object on stdout.

MCP

MCP tool: `execute_arcpy_script` with `script` and optional `args`.

ArcGIS Pro Python toolbox

Source: `pro-addin/README.md`

Python toolbox (`.pyt`) as a lightweight alternative to a full .NET ArcGIS Pro SDK add-in.

Usage

1. In ArcGIS Pro: **Insert** → **Toolbox** → **Add Toolbox** and point to the `pro-addin` folder (or copy `uchuva_context_tools.pyt` into your project).
2. Run **Export project summary (.aprx)** to write a JSON file with map and layout names for use as context in Cursor (no geometries).

This toolbox does not call cloud LLMs; MCP integration stays in Cursor/Claude Desktop.

Environment template (.env.example)

Source: `.env.example`

```
# Copy to .env or export before launching MCP. Public portal search without a token may return limited resu

ARCHGIS_PORTAL_URL=https://www.arcgis.com
# ARCHGIS_TOKEN=

# Optional: generateToken against PORTAL_URL
# ARCHGIS_USERNAME=
# ARCHGIS_PASSWORD=

ARCHGIS_CAPABILITY_READ=true
ARCHGIS_CAPABILITY_WRITE=false

# Comma/semicolon-separated URL prefixes for run_geoprocessing
# ARCHGIS_GP_ALLOWLIST=

# Optional write-audit NDJSON log
# ARCHGIS_AUDIT_LOG=C:/Temp/uchuva-arcgis-audit.ndjson

# ArcGIS Pro Python (execute_arcpy_script only)
# ARCHGIS_PRO_PYTHON=C:/Program Files/ArcGIS/Pro/bin/Python/envs/arcgispro-py3/python.exe

# Inno Setup (Windows build: npm run installer:gui)
# ISCC=C:/Program Files (x86)/Inno Setup 6/ISCC.exe
# INNO_SETUP_DIR=C:/Program Files (x86)/Inno Setup 6
```

Example MCP config (installer)

Source: `installer/setup/cursor-mcp.example.json`

```
{
  "mcpServers": {
    "uchuva-arcgis": {
      "command": "C:/FULL/PATH/TO/UchuvaArcGIS-Mcp.exe",
      "args": [],
      "env": {
        "ARCHGIS_PORTAL_URL": "https://www.arcgis.com"
      }
    }
  }
}
```

Installer: info before install (plain text)

Source: `installer/setup/INFO_BEFORE_INSTALL.txt`

```
This wizard will install:

  - The Uchuva ArcGIS MCP server (bundled Node runtime)
  - Optional ArcPy helper scripts in the arcpy-bridge folder

It does NOT include ArcGIS Pro, ArcGIS Desktop, or ArcGIS Online accounts.

By default, the installer merges the MCP entry into Claude Desktop's claude_desktop_config.json so you only

Installation can run as a normal user (no administrator rights required).
```

Installer: post-install notes for Cursor / Claude (README_CURSOR.txt)

Source: `installer/setup/README_CURSOR.txt`

```
================================================================
  Uchuva ArcGIS MCP — after installation (Cursor / Claude)
================================================================

Installed MCP executable (typical per-user install):

  %LocalAppData%\Programs\Uchuva\ArcGIS-MCP\UchuvaArcGIS-Mcp.exe

================================================================
  Claude Desktop
================================================================

If you kept the install task "Register MCP in Claude Desktop", the installer
merged this path into every known Claude config location, including:

  • %AppData%\Claude\claude_desktop_config.json  (classic installer)

  • %LocalAppData%\Packages\Claude_*\LocalCache\Roaming\Claude\claude_desktop_config.json
    (Microsoft Store / packaged Claude)

A timestamped backup is created beside each file when replacing or merging.

• Fully quit Claude Desktop (system tray Exit), then start it again.
• If the MCP tools do not appear: Help → Troubleshooting → Enable Developer Mode,
  then Settings → Developer → Reload MCP Configuration (depends on app version).

To re-run the merge manually (e.g. after moving the install folder):

  powershell -NoProfile -ExecutionPolicy Bypass ^
    -File "%LocalAppData%\Programs\Uchuva\ArcGIS-MCP\configure-claude-desktop-mcp.ps1" ^
    -InstallDir "%LocalAppData%\Programs\Uchuva\ArcGIS-MCP"

================================================================
  Cursor MCP — valid JSON
================================================================

1) Common paths:

   GLOBAL:  %USERPROFILE%\.cursor\mcp.json
   PROJECT: <your-repo>\.cursor\mcp.json

2) IMPORTANT:

   • The file must be valid JSON: a single root object { ... }
   • Do not paste a second nested "mcpServers" block.
   • Use forward slashes in "command" paths, even on Windows:

       "command": "C:/Users/YourName/AppData/Local/Programs/Uchuva/ArcGIS-MCP/UchuvaArcGIS-Mcp.exe"

   • Full template: installer/setup/cursor-mcp.example.json
     (edit only the "command" path to match your machine)

================================================================
  If Cursor MCP breaks or shows errors
================================================================

1) Quit Cursor.
2) Temporarily rename:
```

```
     %USERPROFILE%\.cursor\mcp.json  -> mcp.json.bak

     and if applicable:

     <project>\.cursor\mcp.json  -> mcp.json.bak

3) Restart Cursor and recreate valid JSON from the example file.

================================================================

Typical env vars in that JSON ("env"):

  ARCHGIS_PORTAL_URL

  ARCHGIS_TOKEN   (or ARCHGIS_USERNAME / ARCHGIS_PASSWORD)

  Optional ArcPy:

  ARCHGIS_PRO_PYTHON=C:/Program Files/ArcGIS/Pro/bin/Python/envs/arcgispro-py3/python.exe

================================================================
```

- Does not install ArcGIS Pro.
- Keep the arcpy-bridge folder next to the MCP exe (the installer copies it).