RESO UPI Service Container

Table of Contents

Introduction

This is a container-based service for the RESO UPI (Universal Property Identifier). It can create or validate a RESO UPI using a RESTful API. It can also be used to discover information about states, counties, and towns from text-based hints.

The container is preconfigured with the following properties:

Property	Description	Setting
host	IP Address of the server	"0.0.0.0"
name	Used for Docker commands	reso-upi-service-container
port	Port serviced by the server	8081
withTLS	Server is secure (HTTPS)	false

The container is not configured to support TLS requests and is intended for internal use, not as a publically-available service.

Installation

The Docker container service must be installed on the computer that the API Service container will be installed on. There are many guides on the internet that discuss Docker installation and a good place to start would be the installation tools provided by the distributor.

The following script can be used to install the API Server as a container-based application once Docker is installed.

```
> ./scripts/build_container.sh
```

Container Operations

Scripts located in the `scripts` directory of the distribution can simplify Docker use.

- build_container.sh
- monitor_container.sh
- start_container.sh
- stop_container.sh
- teardown_container.sh

The scripts were developed using a bash shell. Refer to the script code for details regarding Docker invocation.

Build or Rebuild the Container

This builds and starts the container from an image in the repository:

```
> ./scripts/build_container.sh
```

Starting and Stopping the Container

Once build, a container can be started with the follwing:

```
> ./scripts/start_container.sh
```

The container can be stopped with the following:

```
> ./scripts/stop_container.sh
```

You should only build the container once, and then use the start and stop processes to control the container.

Monitor the Container

To see a running output of the logs for the container, execute the following:

```
> docker logs -f reso-upi-service-container
```

To see a snapshot of the logs for the container, execute the following:

```
> docker logs reso-upi-service-container
```

Removing the Container

This cleans up the docker environment:

```
> ./scripts/remove_container.sh
```

View the Container

The following command allows you to see the containers that are running.

```
> docker ps -a
```

Log into a Container

If you need to log into the service container, use the following command.

```
> docker exec -it reso-upi-service-container /bin/bash
```

API Methods
Scenario #1: parse a upi with seperate state and county

{"success":true,"upi":"US-50013-50650-666777888-R-N"}

Scenario #2: parse a upi with fips

{"success":true,"upi":"US-50013-50650-666777888-R-N"}

```
### Parse a UPI

The following script demonstrates parsing a UPI with the `/upi/parse` method.
```

~~~
```sh
./scripts/parse_upi.sh
============
 RESO UPI Service

 API Method: upi/parse
============

Sending to API Server at: http://localhost:8081
---
Scenario #1: parse a upi
---
Parcing UPI US-50013-50650-666777888-R-N
{"county_code":"013","county_name":"Grand Isle County","country_code":"US","country_name":"U
~~~

Discover UPI Information

The `Discover` capability can be useful when constructing parameters for building UPIs because it creates codes that can be used to build UPIs.

If you use the name of an area (such as a county or town) to discover UPI information, the service will look for matches across all states, counties, and subcounties. Each result contains codes that can be used to build a UPI. Only one of the results si what you were looking for.

The structure of each hint includes fields for the country_code, fips_code, and subcountry_code of the candidate hint. Here is an example of hints received when searching for "Kane".

```
{
...
{"country_code":"US","fips_code":"17089","subcounty_code":"60365","hint_description":"Level=
...
}
```

The country_code, fips_code, and subcountry_code for the hint is received as well as a field named 'hint_descriotion'. The hint_descriotion indicates that the match "Kane" occurred on the county level. The system matched the stored county name of "Kane County".

The following script demonstrates discovering UPI with the `/upi/discover` method.

```
./scripts/discover.sh
============
 RESO UPI Service

 API Method: upi/discover
============
```

```
Sending to API Server at: http://localhost:8081
---
Scenario #1: Return geographic hints API Server
---
Retreiving hints for: Kane
[{"country_code":"US","fips_code":"17061","subcounty_code":"38882","hint_description":"Level

---
Scenario #2: Return geographic hints API Server
---
Retreiving hints for: Charleston
[{"country_code":"US","fips_code":"17029","subcounty_code":"12580","hint_description":"Level
```

Get Build Version

Execute the following script to return all facts.

```
> ./scripts/getServerVersion.sh
============
 RESO UPI Service

 API Method: version
============

Sending to API Server at: http://localhost:8081
---
Scenario #1: Return the build version of the API Server
---
Version: build20210317
```

Notes
