MCP Gateway and Registry Code Walkthrough

compiled by D. Gueorguiev, 9/10/2025

# Available MCP Servers

currenttime

fininfo

mcpgw

realserverfaketools

Let us look first at the simplest server currenttime

## Inside MCP Server currenttime

This server provides an interface to get the current time in a specified timezone using the timeapi.io API.

We start with the imports and few helper functions and boilerplate code:

[mcp-gateway-registry/servers/currenttime/server.py](https://github.com/agentic-community/mcp-gateway-registry/blob/main/servers/currenttime/server.py)

import os

import argparse

import logging

from mcp.server.fastmcp import FastMCP

from pydantic import Field

from typing import Annotated

# Configure logging

logging.basicConfig(

level=logging.INFO,

format='%(asctime)s,p%(process)s,{%(filename)s:%(lineno)d},%(levelname)s,%(message)s'

)

logger = logging.getLogger(\_\_name\_\_)

def parse\_arguments():

"""Parse command line arguments with defaults matching environment variables."""

parser = argparse.ArgumentParser(description="Current Time MCP Server")

parser.add\_argument(

"--port",

type=str,

default=os.environ.get("MCP\_SERVER\_LISTEN\_PORT", "8000"),

help="Port for the MCP server to listen on (default: 8000)",

)

parser.add\_argument(

"--transport",

type=str,

default=os.environ.get("MCP\_TRANSPORT", "streamable-http"),

choices=["sse", "streamable-http"],

help="Transport type for the MCP server (default: streamable-http)",

)

return parser.parse\_args()

We instantiate the helper function to parse the arguments supplied with the command line invocation of the module [mcp-gateway-registry/servers/currenttime/server.py](https://github.com/agentic-community/mcp-gateway-registry/blob/main/servers/currenttime/server.py)

Then we log in the parsed arguments and instantiate the FastMCP server:

# Parse arguments at module level to make them available

args = parse\_arguments()

# Log parsed arguments for debugging

logger.info(f"Parsed arguments - port: {args.port}, transport: {args.transport}")

logger.info(f"Environment variables - MCP\_TRANSPORT: {os.environ.get('MCP\_TRANSPORT', 'NOT SET')}, MCP\_SERVER\_LISTEN\_PORT: {os.environ.get('MCP\_SERVER\_LISTEN\_PORT', 'NOT SET')}")

# Initialize FastMCP server

mcp = FastMCP("CurrentTimeAPI", host="0.0.0.0", port=int(args.port))

mcp.settings.mount\_path = "/currenttime"

A function which generates a prompt which will be discoverable by a client is the decorated system\_prompt\_for\_agent:

@mcp.prompt()

def system\_prompt\_for\_agent(location: str) -> str:

"""

Generates a system prompt for an AI Agent that wants to use the current\_time MCP server.

This function creates a specialized prompt for an AI agent that wants to determine the

current time in a specific timezone.

The prompt instructs an model to provide the name of a timezone closest to the current

location provided by the user so that the timezone name (such as America/New\_York,

Africa/Cairo etc.) can be passed as an input to the tools provided by the current\_time MCP

server.

Args:

location (str): The location of the user, which will be used to determine the timezone.

Returns:

str: A formatted system prompt for the AI Agent.

"""

system\_prompt = f"""

You are an expert AI agent that wants to use the current\_time MCP server. You will be provided with the user's location as input.

You will need to determine the name of the timezone closest to the current location provided by the user so that the timezone name (such as America/New\_York, Africa/Cairo etc.)

can be passed as an input to the tools provided by the current\_time MCP server.

The user's location is: {location}

"""

return system\_prompt

# References

[1]