## Overview

This module handles the configuration management for the code monitoring tool. It defines a MonitorConfig class that loads settings from a YAML file (.monitor\_config.yaml by default). These settings determine which file extensions to monitor and which file or directory patterns to ignore. A global singleton instance, CONFIG, is created and exported for easy access to these settings throughout the application.

## Classes

### MonitorConfig

A class to load and store configuration settings from a YAML file.

The MonitorConfig class is responsible for encapsulating the application’s configuration. When an object of this class is created, it initializes with default settings and then attempts to override them by reading a specified YAML configuration file. If the configuration file does not exist, the default values are used.

The two main configuration parameters are: - ignore\_patterns: A list of glob patterns for files and directories to be ignored by the monitor. - file\_extensions: A list of file extensions (e.g., .py, .md) that the monitor should track.

#### \_\_init\_\_(config\_path='.monitor\_config.yaml')

Initializes the MonitorConfig instance.

This constructor sets up the initial state with default values. It then checks if the file at config\_path exists. If it does, the method opens and parses the YAML file, updating the instance’s attributes with the values found in the file. Keys not present in the YAML file will retain their default values.

* **Parameters:**
  + config\_path (str, optional): The path to the YAML configuration file. Defaults to '.monitor\_config.yaml'.
* **Returns:**
  + None
* **Usage Example:**

An example of a .monitor\_config.yaml file:

# .monitor\_config.yaml  
# List of glob patterns to ignore  
ignore\_patterns:  
 - "\*/.venv/\*"  
 - "\*/\_\_pycache\_\_/\*"  
 - "\*.tmp"  
 - "docs/\_build/\*"  
  
# List of file extensions to monitor  
file\_extensions\_to\_check:  
 - ".py"  
 - ".md"  
 - ".yaml"

When MonitorConfig is instantiated, it will load these values.

# Assuming the .monitor\_config.yaml above exists  
config = MonitorConfig()  
  
print(config.ignore\_patterns)  
# Output: ['\*/.venv/\*', '\*/\_\_pycache\_\_/\*', '\*.tmp', 'docs/\_build/\*']  
  
print(config.file\_extensions)  
# Output: ['.py', '.md', '.yaml']

## Global Instances

### CONFIG

A global, pre-initialized instance of the MonitorConfig class.

* **Type:** MonitorConfig
* **Description:** This singleton instance is created when the config.py module is first imported. It automatically attempts to load configuration from .monitor\_config.yaml in the current working directory. It is the intended way for other parts of the application to access configuration settings.
* **Usage Example:**

from code\_monitor.config import CONFIG  
  
# Use the global CONFIG instance to access settings  
def should\_ignore\_file(filepath):  
 # Access ignore\_patterns directly from the global config  
 for pattern in CONFIG.ignore\_patterns:  
 if fnmatch.fnmatch(filepath, pattern):  
 return True  
 return False  
  
def is\_valid\_extension(filepath):  
 # Access file\_extensions directly from the global config  
 return any(filepath.endswith(ext) for ext in CONFIG.file\_extensions)

yaml github: token: “your\_github\_personal\_access\_token” owner: “github\_username\_or\_org” repo: “repository\_name”

monitor: check\_interval: 60 commit\_check\_depth: 5 target\_branch: “main”

actions: run\_command: “echo ‘New commit found!’”

python result = sub\_num(10, 3) print(result) # Expected output: 7

python result = sub\_num(10, 3) print(result) # Expected output: 7