Logger Config Component

FirstLoggerConfigDTO

SecondLoggerConfigDTO

NoLoggerConfig

«Enum»
FileMode
+READ: str
+OVERWRITE: str
+APPEND: str

«dataclass(frozen=True)»
LoggerConfigDTO
+file_path: str
+message_template: Template
+file_mode: FileMode
+max_file_size_by_byte: int

Logger Config Component

Logger Subject Component

«interface»
LoggerObserverInterface
+update(log_entry: LogEntryDTO)

-observer's

«interface»
LoggerSubjectInterface
+register_logger_observer(new_observer: LoggerObserverInterface): bool
+remove_logger_observer(removable_observer: LoggerObserverInterface): bool
+notify_logger_observers(log_entry: LogEntryDTO): bool

Logger Subject Component

LogEntry Component

LogEntryFactory
«@staticmethod»+create_new_log_entry(level: str, msg_text: str, context: str): LogEntryDTO

InfoLogEntry   +create
WarningLogEntry   +create
ErrorLogEntry   +create
CriticalLogEntry   +create
DebugLogEntry   +create
TraceLogEntry   +create
NewLevelLogEntry   +create

«dataclass(frozen=True)»
LogEntryDTO
+message_text: str
+context: str
+created_at: datetime
+get_level(): str

LogEntry Component

Boundary for Logging

BaseLogger Component

«metaclass=ABCMeta»
BaseLogger
-perform_file_explorer: FileExplorerInterfaceStrategy
-logger_config: LoggerConfigDTO
+process_log_msg(log_entry: LogEntryDTO): bool
+set_new_config(new_config: LoggerConfigDTO)
+set_new_perform_file_explorer(new_file_explorer: FileExplorerInterfaceStrategy)
#read_log_entry(log_entry: LogEntryDTO): Dict[str, str]
#flush_log_msg(data: Dict[str, str]): bool

ConcreteBaseLoggerObserver

ConcreteBaseLogger

BaseLogger Component

-contains

File Explorer Component

«interface»
FileExplorerInterfaceStrategy
+create_file(path: str, file_name: str): bool
+create_dir(path: str, dir_name: str): bool
+check_path_is_exists(path: str): bool
+read_from_file(path: str): str
+overwrite_file(path: str, content: str): bool
+append_to_file(path: str, content: str): bool

WindowsFileExplorer   LinuxFileExplorer   NoFileExplorer

File Explorer Component

Boundary for OS Interaction

Abstract DataBase Component

«metaclass=ABCMeta»
DataBase
-logger_observers_list: LoggerObserverInterface[0..*]
+query_param_placeholder: str = "?"
+log_entry_factory: LogEntryFactory
+deconstruct_database_and_components()
+__del__()

Boundary DataBase Core

Abstract DataBase Component

Single Connection DataBase Component

«metaclass=ABCMeta»
SingleConnectionDataBase
#perform_connection_manager: SingleConnectionManager
#config: Dict[str, Any]
#transaction_manager: TransactionManager
+set_new_connection_config(new_config: Dict[str, Any])
+set_new_connection_manager(new_manager: SingleConnectionManager)
+set_new_transaction_manager(new_manager: TransactionManager)

ConcreteSingleConnectionDataBase1

ConcreteSingleConnectionDataBase2

Single Connection DataBase Component

Pool Connection DataBase Component

«metaclass=ABCMeta»
PoolConnectionDataBase
#perform_connection_manager: PoolConnectionManager
#config: PoolConfigDTO
#transaction_manager: TransactionManager
+set_new_pool_config(new_config: PoolConfigDTO)
+set_new_pool_manager(new_manager: PoolConnectionManager)
+set_new_transaction_manager(new_manager: TransactionManager)

ConcretePoolConnectionDataBase1

ConcretePoolConnectionDataBase2

Pool Connection DataBase Component

Pool Config Component

PoolConfigDTO
-connection_config: Dict[str, Any]
+pool_size: int
-validate_fields_data()
+set_new_connection_config(new_config: Dict[str, Any])

PostgreSQLPoolConfigDTO

MySQLPoolConfigDTO

Pool Config Component

Boundary for Connection Configuration

Boundary for Query Core

Query Interface Component

«interface»
QueryInterface
+execute_query_no_returns(query: str, *params)
+execute_query_returns_one(query: str, *params): str
+execute_query_returns_many(query: str, returns_count: int, *params): Tuple[str, ...]
+execute_query_returns_all(query: str, *params): Tuple[str, ...]

Query Interface Component

Boundary for Query Core

Transaction Manager Component

«Enum»
TransactionState
+INITIALIZED: str
+ACTIVE: str
+COMMITED: str
+ROLLED_BACK: str

«metaclass=ABCMeta»
TransactionManager
-state: TransactionState
-isolation_level: IsolationLevel
-current_connection
-query_param_placeholder: str
+begin(connection, query_param_placeholder: str)
+execute_in_active_transaction(query: str, *params)
+commit()
+rollback()
-apply_isolation_level(isolation_level: IsolationLevel)

«Enum»
IsolationLevel
+READ_UNCOMMITTED: str
+READ_COMMITTED: str
+REPEATABLE_READ: str
+SERIALIZABLE: str

MySQLTransactionManager

PostgreSQLTransactionManager

Transaction Manager Component

Pool Manager Component

MySQLPoolConnectionAdapter

PostgreSQLPoolConnectionAdapter

PoolConnectionManager
-active_pool: PoolConnectionInterface
+initialize_new_pool(pool_config: PoolConfigDTO)
+reinitialize_active_pool(pool_config: PoolConfigDTO)
-deconstruct_active_pool()
+read_active_pool_status()
+get_connection_from_active_pool()
+return_connection_to_active_pool()

«interface»
PoolConnectionInterface

NoPoolConnectionManager

Pool Manager Component

Single Manager Component

SingleConnectionManager
-connection_adapter: SingleConnectionInterface
+initialize_new_connection(connection_config: Dict[str, Any])
+reinitialize_connection(connection_config: Dict[str, Any])
-deconstruct_connection()
+read_connection_status()
+get_connection()

«interface»
SingleConnectionInterface
+close()
+commit()
+connect()
+get_cursor()
+reconnect()

NoSingleConnectionManager

PostgreSQLSingleConnectionAdapter
-adapted_connection

MySQLSingleConnectionAdapter
-adapted_connection

Single Manager Component

Boundary for DBMS Interaction

Main Component

«Enum»
DBMS
+MySQL: str
+PostgreSQL: str
+SQLite: str

«Enum»
ConnectionMode
+Single: str
+Pool: str

DataBaseFactory
-baked_database: DataBase
-baked_connection_interface
-baked_transaction_manager
-baked_pool_config = None
+create_database(dbms_type: DBMS, connection_mode: ConnectionMode, connection_config: Dict[str, Any], loggers: LoggerObserverInterface[0..*] = None): DataBase
-bake_database_by_dbms_type()
-bake_connection_interface_by_connection_mode()
-bake_transaction_manager()
-set_baked_connection_interface_to_database()
-set_baked_transaction_manager_to_database()
-set_loggers_to_database_observers_list()
-if_mode_is_pool_bake_pool_config()
-set_baked_pool_config_to_connection_interface()

Main Component

Boundary for API Use