

SecuIntegrator 24 V2

公用模組需求文件

|  |  |  |
| --- | --- | --- |
| **撰寫人**  **Brian Chuang** |  |  |

|  |
| --- |
| 目錄 |
| [壹、 LogManager 3](#_Toc179743056)  [一、 需求分析 3](#_Toc179743057)  [二、 非功能需求 3](#_Toc179743058)  [三、 可擴展性需求 4](#_Toc179743059)  [四、 介面需求 4](#_Toc179743060)  [五、 測試案例 6](#_Toc179743061)  [貳、 TaskScheduleManager 10](#_Toc179743062) |

# LogManager

## 需求分析

透過需求分析定義 LogManager 的核心功能，例如：

* + 1. 記錄不同等級(Error、Warning、Information)的日誌。
    2. 註冊事件記錄日誌時通知外部。
    3. 設定保留天數並清理過期日誌。
    4. 文字檔日誌要有可讀性，使用Json格式，要記錄時間戳記，並需顯示結構分明並加入換行。
    5. 儲存路徑邏輯要可以支援置換。
    6. 儲存檔名生成邏輯要可以支援置換。

這些需求會轉化成測試案例，並依此開發。

## 非功能需求

* + 1. 效能
       1. 日誌記錄操作應輕量並且不影響主要應用程式的效能。
       2. 清理過程應避免長時間佔用系統資源。
    2. 可測試性

支援 xUnit 及 Moq

## 可擴展性需求

* + - 1. 可置換項目一律定義 Interface，提供使用者未來可以實作自己的邏輯傳入，如未特別傳入則使用原有內部邏輯處理。
      2. 預留其它文字檔案格式支援。
      3. 預留支援資料庫。

## 介面需求

* + 1. 日誌最大長度設定
       1. 類型：屬性。
       2. 說明：允許設定每條日誌的最大字數限制，防止日誌過長影響效能或超出儲存體限制。
       3. 用途：限制日誌訊息的長度，保證儲存日誌時的效能和可讀性。
    2. 日誌保留天數設定
       1. 類型：屬性。
       2. 說明：允許設定日誌檔案保留的最大天數，過期日誌會自動清理。
       3. 用途：用來控制日誌檔案的儲存天數，避免檔案過多佔用儲存空間。
    3. 寫入日誌
       1. 類型：方法。
       2. 說明：將日誌寫入到指定的儲存媒介中，支援非同步寫入，並根據日誌等級處理不同類型的日誌訊息。
       3. 用途：用來記錄應用程式中的事件和錯誤資訊，並以設定好的格式和路徑儲存。
    4. 日誌事件
       1. 類型：事件。
       2. 說明：用來通知外部模組當有日誌新增時觸發。例如，當 LogManager 記錄一筆日誌後，會透過此事件通知訂閱者，外部系統可根據此事件更新顯示或進行後續操作。
       3. 用途：當有新的日誌被寫入時，觸發此事件通知。
    5. 路徑命名類別
       1. 類型：介面。
       2. 說明：此類別負責生成日誌的儲存路徑，可以根據外部配置或參數來替換儲存路徑，支持不同環境下日誌存放位置的變更。
       3. 用途：用來生成日誌檔案的儲存位置。
    6. 檔案命名類別
       1. 類型：介面。
       2. 說明：負責生成日誌檔案的命名格式，允許自定義檔名（如：包含日期、時間、應用程式名稱等…）。
       3. 用途：用來產生日誌檔案的命名。
    7. 日誌寫入類別
       1. 類型：介面。
       2. 說明：此類別負責將日誌內容寫入檔案中，支持非同步操作，並且應具有錯誤處理能力，確保即使在寫入錯誤時系統不會崩潰。
       3. 用途：負責將日誌內容寫入到指定的儲存體中。
    8. 日誌清理類別
       1. 類型：介面。
       2. 說明：負責根據設定的保留天數，定期清理過期的日誌檔案，應避免在清理過程中佔用過多資源。
       3. 用途：清理過期日誌，確保日誌檔案數量不會無限制增長。

## 測試案例

* + 1. 日誌記錄功能測試
       1. 測試目標：驗證 LogManager 是否能正確記錄不同等級的日誌，並確保日誌內容符合可讀性要求。
       2. 測試條件：
          1. LogManager 配置完成，並設定為使用 JSON 格式記錄日誌。
          2. 將不同等級的日誌（例如：Error、Warning、Information）寫入日誌系統。
       3. 驗證點：
          1. 每條日誌應包含時間戳記、日誌等級、訊息內容，並格式化為可讀的 JSON。
          2. 應驗證日誌內容結構是否分明並帶有換行符號，符合 JSON 格式的可讀性需求。
    2. 日誌事件通知測試
       1. 測試目標：確認 LogManager 在記錄日誌時是否能正確觸發日誌更新事件。
       2. 測試條件：
          1. 訂閱 LogManager 的日誌更新事件。
          2. 當一筆新的日誌被寫入時，檢查事件是否被觸發。
       3. 驗證點：

事件應能被正確觸發，並將新寫入的日誌內容傳遞給事件的訂閱者。

* + 1. 日誌最大長度限制測試
       1. 測試目標：驗證 LogManager 是否能根據設定的最大長度限制截斷過長的日誌訊息。
       2. 測試條件：
          1. 將 LogManager 配置為允許的最大日誌長度（例如 500 字）。
          2. 寫入一條超過最大長度的日誌訊息。
       3. 驗證點：

日誌訊息應被截斷至指定長度，並正確記錄至檔案。

* + 1. 日誌保留天數測試
       1. 測試目標：確認 LogManager 能根據設定的保留天數自動清理過期日誌。
       2. 測試條件：
          1. 設定 LogManager 的日誌保留天數（例如 5 天）。
          2. 日誌系統中存在多筆超過保留天數的日誌。
       3. 驗證點：

執行日誌清理操作後，過期的日誌應被正確刪除，保留天數內的日誌應正常保存。

* + 1. 日誌路徑和檔名生成測試
       1. 測試目標：驗證 LogManager 是否能根據外部配置的儲存路徑和檔名生成邏輯，正確生成並儲存日誌。
       2. 測試條件：
          1. 設定自定義的日誌儲存路徑和檔名格式。
          2. 寫入日誌並檢查儲存檔案的位置與檔名是否符合預期。
       3. 驗證點：

應生成正確的日誌儲存路徑和檔名，日誌應保存在指定位置且符合命名規則（如包含日期、時間戳記等）。

* + 1. 非同步寫入日誌測試
       1. 測試目標：確認 LogManager 能夠在不影響應用程式效能的情況下，非同步寫入日誌。
       2. 測試條件：
          1. 寫入多筆日誌，並模擬高負載操作。
          2. 測量寫入過程中的應用效能和資源佔用。
       3. 驗證點：

日誌應能在異步情況下正常寫入，並且應用程式的效能不應顯著下降。

* + 1. 手動清理日誌測試
       1. 測試目標：驗證 LogManager 的手動清理功能是否能有效清除過期日誌。
       2. 測試條件：
          1. 存在多筆過期日誌，且部分日誌在保留期限內。
          2. 呼叫手動清理方法。
       3. 驗證點：

過期日誌應被正確刪除，保留期限內的日誌應被正常保留。

# TaskScheduleManager