Система логування – система структурування інформації та запису виконаних дій, необхідних для дії підсистем та користувачів бази даних

Роль – структурування та передача інформації всередині підсистеми

**Юзкейси системи логування**

**Створення log-файлу**

Діючі особи:

1. Підистема логування
2. Підсистема-клієнт, яка надсилає дані

Мета: створити log-файл для його передачі в базу даних чи подальшого редагування

Основний сценарій:

1. Підсистема-клієнт надсилає дані для системи логування
2. Підистема логування формує log-файли із сирцевих даних

Код у PlantUML:

@startuml

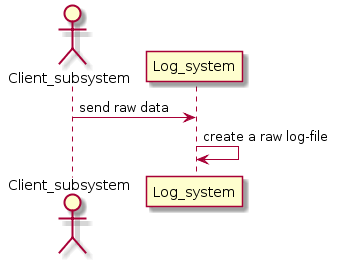
actor Client\_subsystem

Client\_subsystem -> Log\_system : send raw data

Log\_system -> Log\_system : create a raw log-file

@enduml

Діаграма у PlantUML:



**Вивести історію роботи бази даних**

Діючі особи:

1. Підсистема логування
2. Підсистема-клієнт, яка вимагає дані
3. Підсистема Database

Мета: вивести історію дій у базі даних, не виводячи в повній мірі всю інформацію про кожен log-файл

Основний сценарій:

1. Підсистема-клієнт надсилає підсистемі логування часові рамки до історії дій в базі даних
2. Підсистема логування надсилає в базу даних запит на отримання інформації
3. Підсистема логування вибирає основну інформацію з отриманих log-файлів та надсилає її клієнту

Код у PlantUML:

@startuml

actor Client\_subsystem

Client\_subsystem -> Log\_subsystem : send history request

activate Log\_subsystem

Log\_subsystem -> Database : send request

activate Database

alt data available

ref over Database : Database::extract\_files()

Database -> Log\_subsystem : give data

ref over Log\_subsystem : Log\_subsystem::process\_data()

Log\_subsystem -> Client\_subsystem : return history

else data unavailable

Database --> Log\_subsystem : history access unsuccessful

destroy Database

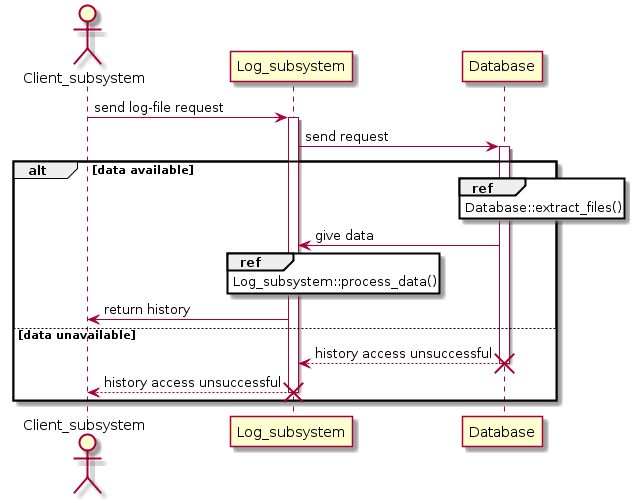
Log\_subsystem --> Client\_subsystem : history access unsuccessful

destroy Log\_subsystem

end

@enduml

Діаграма у PlantUML:



**Оновлення лог-файлу**

Діючі особи:

1. Підсистема-клієнт, яка хоче оновити лог-файл
2. Система логування
3. Підсистема Database

Мета: оновити log-файл, якщо відбулось щоси важливе

Основний сценарій:

1. Підсистема-клієнт надсилає запит на оновлення конкретної інформації у конкретному лог-файлі
2. Система логування надсилає запит до Database на оновлення інформації
3. Database оновлює необхідні дані

Код у PlantUML:

@startuml

actor Client\_subsystem

Client\_subsystem -> Log\_system : send data and file to update

activate Log\_system

Log\_system -> Database : update information

activate Database

alt update success

ref over Database : Database::update\_file()

Database --> Log\_system : edit success

Log\_system --> Client\_subsystem : edit success

else update unsuccess

Database --> Log\_system : edit unsuccess

destroy Database

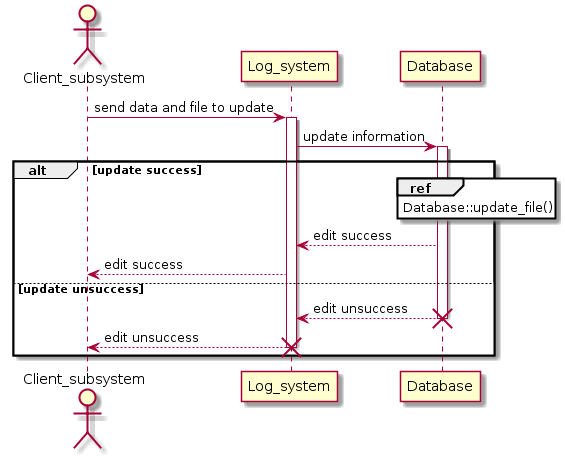
Log\_system --> Client\_subsystem : edit unsuccess

destroy Log\_system

end

@enduml

Діаграма у PlantUML:

****

**Обробити звичайний log-файл**

Діючі особи:

1. Підистема логування
2. Підсистема-клієнт, яка надсилає дані
3. Підсистема Database

Мета: записати log-файл у базу даних

Основний сценарій:

1. За допомогою першого юзкейсу згенерувати log-файл
2. Накласти на згенерований log-файл прапори належності log-файла до звичайного типу
3. Надіслати запит у Database на додавання нового log-файлу до бази даних

Код у PlantUML:

@startuml

actor Client\_subsystem

Client\_subsystem -> Log\_subsystem : send raw data

ref over Log\_subsystem : Log\_subsystem::create\_raw\_log\_file()

Log\_subsystem -> Log\_subsystem : set ordinary log file flags

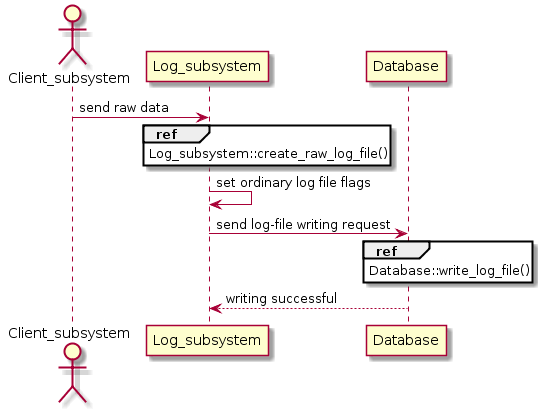
Log\_subsystem -> Database : send log-file writing request

ref over Database : Database::write\_log\_file()

Database --> Log\_subsystem : writing successful

@enduml

Діаграма у PlantUML:

****

**Обробити повідомлення про помилку**

Діючі особи:

1. Підистема логування
2. Підсистема-клієнт, яка надсилає дані
3. Підсистема Database

Мета: у разі виникнення помилки швидко відреагувати на неї та записати необхідну інформацію про неї

Основний сценарій:

1. За допомогою першого юзкейсу згенерувати log-файл
2. Накласти на згенерований log-файл прапори належності log-файла до типу помилки
3. Надіслати запит у Database на додавання нового log-файлу до бази даних

Код у PlantUML:

@startuml

actor Client\_subsystem

Client\_subsystem -> Log\_subsystem : send raw data

ref over Log\_subsystem : Log\_subsystem::create\_raw\_log\_file()

Log\_subsystem -> Log\_subsystem : set error log file flags

Log\_subsystem -> Database : send log file to database

activate Database

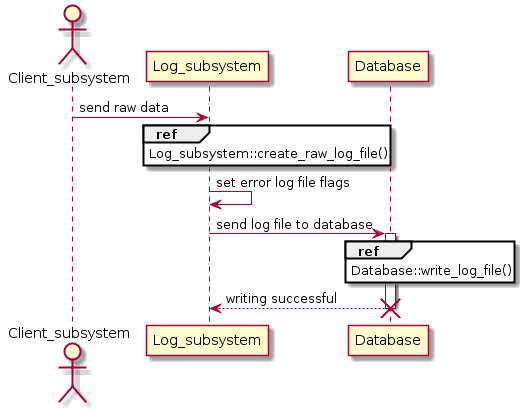
ref over Database : Database::write\_log\_file()

Database --> Log\_subsystem : writing successful

destroy Database

@enduml

Діаграма у PlantUML:

****

**Згенерувати попередження про можливу помилку/небезпечні дії**

Діючі особи:

1. Система логування
2. Підсистема Database
3. Підсистема GUI

Мета: запис у базу даних інформації про небезпечну дію в системі

Основний сценарій:

1. За допомогою першого юзкейсу згенерувати log-файл
2. Накласти на згенерований log-файл прапори належності log-файла до типу попередження
3. Надіслати запит у Database на додавання нового log-файлу до бази даних

Код у PlantUML:

@startuml

actor Client\_subsystem

Client\_subsystem -> Log\_subsystem : send raw data

ref over Log\_subsystem : Log\_subsystem::create\_raw\_log\_file()

Log\_subsystem -> Log\_subsystem : set warning log file flags

Log\_subsystem -> Database : send log file to database

activate Database

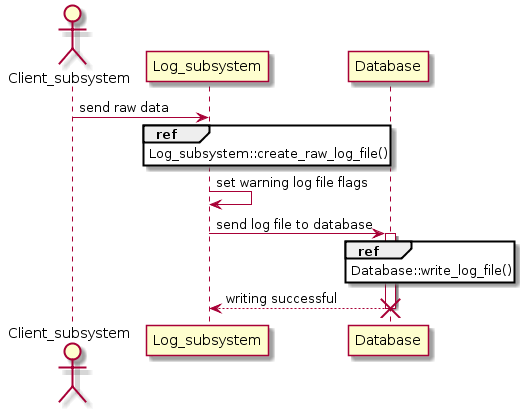
ref over Database : Database::write\_log\_file()

Database --> Log\_subsystem : writing successful

destroy Database

@enduml

Діаграма у PlantUML:

****

**Обробити критичну помилку**

Діючі особи:

1. Система логування
2. Підсистема Database
3. Підсистема GUI

Мета: запис у базу даних інформації про критичну помилку дію в системі

Основний сценарій:

1. За допомогою першого юзкейсу згенерувати log-файл
2. Накласти на згенерований log-файл прапори належності log-файла до типу критичної помилки
3. Надіслати запит у Database на додавання нового log-файлу до бази даних

Код у PlantUML:

@startuml

actor Client\_subsystem

Client\_subsystem -> Log\_subsystem : send raw data

ref over Log\_subsystem : Log\_subsystem::create\_raw\_log\_file()

Log\_subsystem -> Log\_subsystem : set critical error log file flags

Log\_subsystem -> Database : send log file to database

activate Database

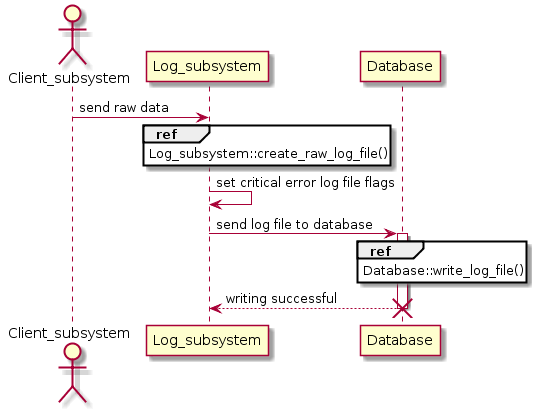
ref over Database : Database::write\_log\_file()

Database --> Log\_subsystem : writing successful

destroy Database

@enduml

Діаграма у PlantUML:

****