**Отчёт по лабораторной работе: Реализация паттерна "Загрузка по требованию" (Lazy Load)**

**1. Описание проблемы предметной области**

В предметной области управления новостным контентом часто возникает необходимость работы с большими объёмами данных, таких как текст статей, изображения или мультимедиа. В приложении новостного ридера пользователь взаимодействует с интерфейсом, где отображается список заголовков новостей, а содержимое конкретной статьи загружается при её выборе. Основная проблема заключается в том, что загрузка содержимого всех новостей при запуске приложения приводит к значительным временным затратам и избыточному использованию ресурсов памяти, особенно если пользователь просматривает только часть материалов.

Целью данной лабораторной работы является оптимизация процесса загрузки содержимого новостей, чтобы минимизировать задержки при запуске программы и сократить потребление ресурсов, загружая данные только по мере необходимости.

**2. Решение: использование паттерна "Загрузка по требованию"**

Паттерн "Загрузка по требованию" (Lazy Load) применяется для отложенной инициализации объектов, что позволяет загружать данные только в момент их фактического использования. В разработанном приложении — новостном ридере на C# с использованием Windows Forms — паттерн реализован следующим образом:

* **Архитектура приложения**: Приложение состоит из трёх основных классов:
  + Form1 — главная форма, отвечающая за пользовательский интерфейс (список заголовков в ListView и отображение содержимого в WebBrowser).
  + NewsLibrary — класс, управляющий коллекцией новостей.
  + NewsItem — класс, представляющий отдельную новость с заголовком, именем файла и содержимым.
* **Реализация паттерна**:  
  В классе NewsItem содержимое новости (текст HTML-файла) загружается только при вызове метода GetContent(). Для этого используются два поля:
  + content: String — хранит содержимое новости после загрузки.
  + isLoaded: Boolean — флаг, указывающий, было ли содержимое загружено.  
    Метод GetContent() проверяет значение isLoaded. Если isLoaded = false, содержимое загружается из файла, флаг устанавливается в true, и содержимое сохраняется в поле content. Последующие вызовы метода возвращают уже загруженное содержимое без обращения к файлу.

Пример кода реализации:

public string GetContent()

{

if (!isLoaded)

{

Console.WriteLine($"Загрузка содержимого из {Filename}...");

Thread.Sleep(2000); // Имитация задержки

try

{

content = File.ReadAllText(Filename);

}

catch (Exception ex)

{

content = $"<html><body><h1>Ошибка загрузки</h1><p>{ex.Message}</p></body></html>";

}

isLoaded = true;

Console.WriteLine($"Содержимое загружено из {Filename}.");

}

return content;

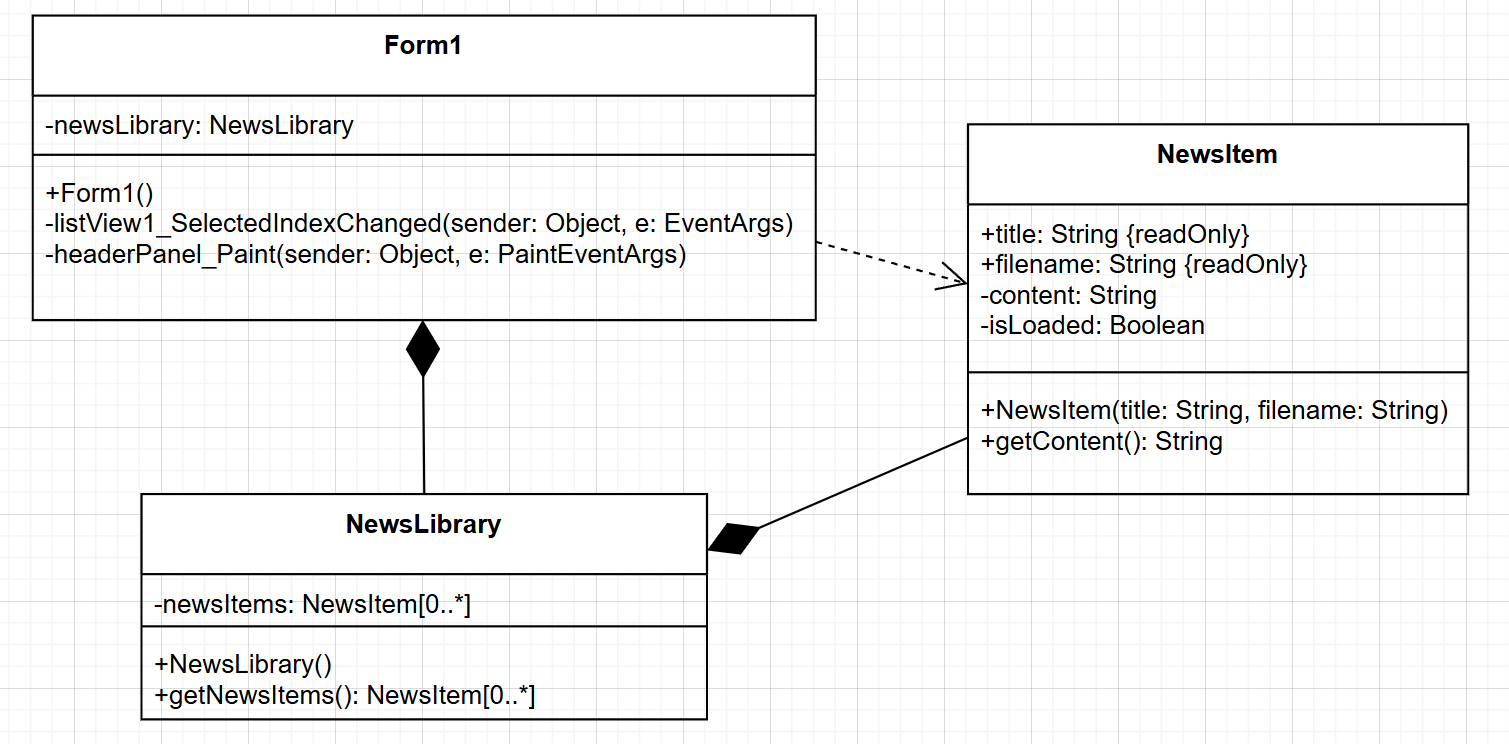
}

* **Сравнение с версией без паттерна**:  
  В версии без паттерна содержимое всех новостей загружается в конструкторе NewsItem при создании объектов, что приводит к задержке при запуске приложения (особенно заметной при большом количестве новостей или медленном доступе к файлам).

Паттерн Lazy Load позволяет загружать содержимое только для выбранной пользователем новости, что снижает время запуска приложения и потребление памяти.

**3. Диаграмма классов**

Для визуализации архитектуры приложения с применением паттерна "Загрузка по требованию" разработана UML-диаграмма классов, представленная на рисунке 1.

****  
*Рисунок 1 – Архитектура приложения с паттерном "Загрузка по требованию"*

* **Связи между классами**:
  + **Form1 → NewsLibrary**: Композиция. Form1 создаёт и владеет экземпляром NewsLibrary, управляя его жизненным циклом.
  + **NewsLibrary → NewsItem**: Композиция. NewsLibrary создаёт и владеет коллекцией NewsItem.
  + **Form1 → NewsItem**: Зависимость. Form1 вызывает метод GetContent() объекта NewsItem, но не владеет им.

Диаграмма отражает использование паттерна Lazy Load в классе NewsItem, где поле isLoaded и метод GetContent() обеспечивают отложенную загрузку содержимого.

**4. Вывод**

Внедрение паттерна "Загрузка по требованию" в новостной ридер существенно повлияло на производительность и эффективность программы:

* **Снижение времени запуска**: В версии с паттерном содержимое новостей не загружается при старте приложения, что сокращает начальную задержку. Например, при использовании трёх новостей с имитацией двухсекундной задержки загрузки общее время запуска сократилось с 6 секунд (в версии без паттерна) до менее чем 1 секунды.
* **Экономия ресурсов**: Память выделяется только для содержимого выбранной новости, что особенно важно при работе с большим количеством данных или медленным доступом к файлам.
* **Гибкость**: Паттерн позволяет легко масштабировать приложение, добавляя новые новости без увеличения времени запуска.

В то же время версия без паттерна демонстрирует значительные задержки при инициализации, так как все файлы загружаются сразу, что неэффективно при редком просмотре всех новостей.

Паттерн Lazy Load оказался эффективным решением для оптимизации загрузки данных в приложении, улучшив пользовательский опыт и снизив требования к ресурсам.