

Assignment-04



Session: 2021 – 2025

Submitted by:

Muhammad Yaqoob 2021-CS-118

Submitted to:

Dr Syed Khaldoon Khurshid

Department of Computer Science

University of Engineering and Technology UET Lahore

Table of Contents

1	Structure Guided Browsing.....	3
1.1	Introduction.....	3
1.2	Steps in Development	3
1.2.1	Choosing an application.....	3
1.2.2	Defining the Content Hierarchy	3
1.2.3	Create a Visual Content Map	3
1.2.4	Implement Navigation Elements.....	4
1.2.5	Implement Interactive Features.....	4
1.3	Code to Structure Guided Browsing	5
1.3.1	Defining the Content Hierarchy	5
1.3.2	Creating a Visual Content Map	6
1.3.3	Implementing Navigation Elements.....	7
1.3.4	Implementing Interactive Features.....	7
1.4	Conclusion	8
2	Hypertext Model.....	9
2.1	Introduction.....	9
2.2	Objectives	9
2.3	Steps in Development	9
2.3.1	Choosing an application.....	9
2.3.2	Defining Nodes and Content.....	9
2.3.3	Creating Links.....	9
2.3.4	Designing User Interface	9
2.3.5	Implementing Hyperlinks	9
2.3.6	Testing Navigation	10
2.3.7	Enhancing User Experience	10
2.4	Conclusion	10

1 Structure Guided Browsing

1.1 Introduction

This project implements Structure Guided Browsing for a content management system (CMS). The goal is to provide users with an intuitive way to explore a hierarchical structure of topics and categories, enabling efficient access to information. The implementation emphasizes interactivity and user-friendly navigation.

1.2 Steps in Development

1.2.1 Choosing an application

Let's implement Structure Guided Browsing for a content management system (CMS) where articles are organized into categories and topics. For the similarity, I have just added the titles of the articles. I can add the content and title as both.

The CMS organizes content into categories and topics. This structure is hierarchical, making it suitable for guided browsing. The user starts at the root node and navigates through levels to access specific information.

1.2.2 Defining the Content Hierarchy

The content hierarchy is represented using a nested Python dictionary. Each key represents a category or topic, and its value can be:

- A dictionary: Indicates subcategories or subtopics.
- A list: Represents terminal nodes with specific items or concepts.

Here is the example:

```
content_hierarchy = {  
    "Global Knowledge": {  
        "Technology": {  
            "Software Development": {  
                "Programming": {  
                    "Languages": ["Python", "Java", "C++"],  
                    "Paradigms": ["Object-Oriented", "Functional"]  
                }  
            }  
        }  
    }  
}
```

1.2.3 Create a Visual Content Map

I have generated a visual map of the hierarchy. It indents sublevels for clarity, helping users visualize the structure.

Here is the example:

- Global Knowledge
 - Technology
 - Software Development
 - Programming
 - Languages
 - Python
 - Java
 - Paradigms
 - Object-Oriented
 - Functional

1.2.4 Implement Navigation Elements

Key features include:

- Dynamic Navigation: Users choose options to drill down into subcategories.
- Backtracking: Users can return to a previous level using the 'back' command.
- Error Handling: Invalid inputs prompt re-selection.

Here is the example working of the navigation elements:

You are in: Global Knowledge

Select an option to navigate further or type 'back' to go back.

1. Technology
2. Science
3. Arts & Humanities

You are in: Science

Select an option to navigate further or type 'back' to go back.

1. Natural Sciences
2. Applied Sciences

You are in: Global Knowledge

Select an option to navigate further or type 'back' to go back.

1. Technology
2. Science
3. Arts & Humanities

1.2.5 Implement Interactive Features

For the interactive feature, I have implemented the collapsible sections. Where users can open article of their interest.

Collapsible sections simulate dynamic exploration by allowing users to expand or collapse categories interactively.

- Users decide whether to expand specific categories.

[+] Technology

Do you want to expand 'Technology'? (y/n): y

[+] Software Development

Do you want to expand 'Software Development'? (y/n): n

1.3 Code to Structure Guided Browsing

1.3.1 Defining the Content Hierarchy

```
# Define the content hierarchy as a dictionary
content_hierarchy = {
    "Global Knowledge": {
        "Technology": {
            "Software Development": {
                "Programming": {
                    "Languages": ["Python", "Java", "JavaScript", "C++", "Rust"],
                    "Paradigms": ["Object-Oriented", "Functional", "Procedural"]
                },
                "Web Development": {
                    "Frontend": ["HTML", "CSS", "JavaScript Frameworks"],
                    "Backend": ["Node.js", "Django", "Flask", "Ruby on Rails"],
                    "DevOps": ["CI/CD", "Docker", "Kubernetes"]
                },
                "Mobile Development": {
                    "Platforms": ["Android", "iOS", "Cross-Platform"],
                    "Languages": ["Java", "Swift", "Flutter", "React Native"]
                }
            },
            "Hardware": {
                "Embedded Systems": ["Arduino", "Raspberry Pi"],
                "IoT": ["Smart Homes", "Wearable Devices"]
            }
        },
        "Science": {
            "Natural Sciences": {
                "Physics": {
                    "Branches": ["Quantum Mechanics", "Relativity", "Thermodynamics"],
                    "Subfields": ["Astrophysics", "Nuclear Physics", "Optics"]
                },
                "Biology": {
                    "Branches": ["Genetics", "Evolution", "Ecology"],
                    "Subfields": ["Molecular Biology", "Microbiology", "Zoology"]
                }
            },
            "Applied Sciences": {
                "Engineering": {
                    "Disciplines": ["Mechanical", "Electrical", "Civil", "Software Engineering"],
                    "Technologies": ["Robotics", "Automation", "Renewable Energy"]
                }
            }
        }
    }
}
```

```

    },
    "Medicine": {
        "Fields": ["Cardiology", "Neurology", "Oncology"],
        "Technologies": ["Imaging", "Telemedicine", "Genomic Medicine"]
    }
},
"Arts & Humanities": {
    "Visual Arts": {
        "Painting": {
            "Styles": ["Impressionism", "Surrealism", "Abstract", "Realism"],
            "Famous Artists": ["Monet", "Picasso", "Van Gogh", "Dali"]
        },
        "Sculpture": {
            "Styles": ["Classical", "Modern", "Abstract"],
            "Famous Sculptors": ["Michelangelo", "Rodin", "Henry Moore"]
        }
    },
    "Music": {
        "Genres": ["Classical", "Jazz", "Rock", "Electronic"],
        "Famous Composers": ["Beethoven", "Mozart", "Miles Davis", "John Coltrane"]
    },
    "Literature": {
        "Genres": ["Fiction", "Non-Fiction", "Poetry", "Drama"],
        "Famous Authors": ["Shakespeare", "Tolstoy", "Hemingway", "Maya Angelou"]
    }
}
}

```

1.3.2 Creating a Visual Content Map

```

# Function to display the visual content map
def display_content_map(hierarchy, indent=0):
    for key, value in hierarchy.items():
        print("  " * indent + f"- {key}")
        if isinstance(value, dict):
            display_content_map(value, indent + 1)
        elif isinstance(value, list):
            for item in value:
                print("    " * (indent + 1) + f"- {item}")

print("Visual Content Map:")
display_content_map(content_hierarchy)

```

1.3.3 Implementing Navigation Elements

```
# Function to navigate the content hierarchy
def navigate_content(hierarchy, path="Global Knowledge"):
    if path not in hierarchy:
        print("Invalid path. Returning to Home.")
        return

    current_level = hierarchy[path]
    while True:
        print(f"\nYou are in: {path}")
        print("Select an option to navigate further or type 'back' to go back.")
        if isinstance(current_level, dict):
            options = list(current_level.keys())
        elif isinstance(current_level, list):
            print("You have reached the end of the content.")
            return

        for i, option in enumerate(options, 1):
            print(f"{i}. {option}")

        choice = input("Choose an option (number) or 'back': ").strip().lower()
        if choice == 'back':
            return

        try:
            selected_option = options[int(choice) - 1]
            navigate_content(current_level, selected_option)
        except (IndexError, ValueError):
            print("Invalid choice. Please select a valid option.")

navigate_content(content_hierarchy)
```

1.3.4 Implementing Interactive Features

```
# Function to simulate collapsible sections
def interactive_content_viewer(hierarchy, level=0):
    for key, value in hierarchy.items():
        if isinstance(value, dict):
            print("    " * level + f"[+] {key}")
            if input(f"Do you want to expand '{key}'? (y/n): ").strip().lower() == 'y':
                interactive_content_viewer(value, level + 1)
        elif isinstance(value, list):
            for item in value:
                print("    " * (level + 1) + f"- {item}")

interactive_content_viewer(content_hierarchy)
```

1.4 Conclusion

This implementation successfully demonstrates Structure Guided Browsing for a CMS. By combining hierarchical representation, interactive features, and user-friendly navigation, it provides a robust framework for content management. Future improvements can focus on extending interactivity and enhancing user experience.

Key Features includes:

- Content Visualization:
 - The hierarchy is presented in an organized format, aiding comprehension.
 - Indentation clearly indicates nested levels.
- Interactive Navigation:
 - Enables dynamic exploration of content.
 - Backtracking ensures user control over navigation.
- Collapsible Sections:
 - Offers flexibility to explore or hide content based on user preference.

2 Hypertext Model

2.1 Introduction

The objective of this assignment is to design and implement a hypertext model in the form of an e-book application. This application will allow users to navigate through chapters and sections as interconnected nodes. The assignment uses Python for its simplicity and support for structured data handling.

2.2 Objectives

- **Dynamic Navigation:** Users can explore chapters through interconnected links.
- **Backtracking:** Users can return to the previous chapter with ease.
- **Interactive CLI:** Offers a user-friendly way to navigate an e-book.

2.3 Steps in Development

2.3.1 Choosing an application

I choose an e-book as the application to implement the Hypertext Model. The e-book will have chapters and sections as nodes, allowing the user to navigate between them.

2.3.2 Defining Nodes and Content

I defined the nodes as chapters and sections within each chapter. Each chapter will act as a "node" containing content, and nodes will be linked together.

The e-book is structured as a dictionary where:

- Keys represent chapter IDs.
- Values include the title, content, and links to other chapters.

Example of chapter structure

```
1: {  
    "title": "Introduction",  
    "content": "Welcome to this ebook. This is the introduction chapter.",  
    "links": [2] # Link to Chapter 1  
},
```

2.3.3 Creating Links

Links are defined as an array of chapter IDs, representing navigable chapters. For example, Chapter 1 links to Chapter 2, forming a bidirectional connection.

2.3.4 Designing User Interface

The user interface is a simple command-line interaction. Users input commands to navigate between chapters. If they want to go back, they can type 'B'.

A command-line interface (CLI) was implemented. Key features include:

- Displaying the chapter title and content.
- Listing available navigation options.
- Accepting user input for navigation.

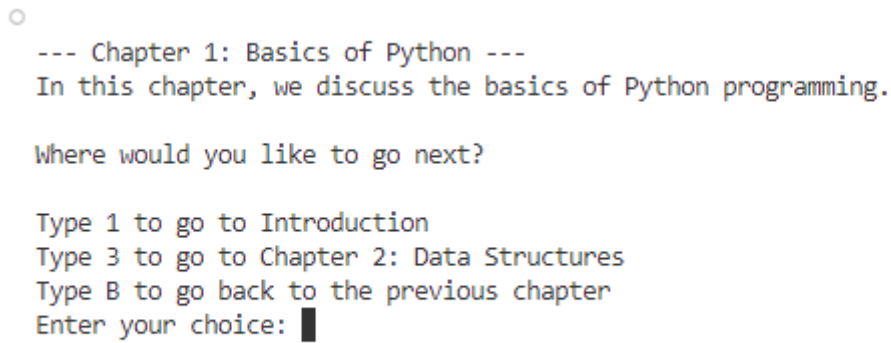
2.3.5 Implementing Hyperlinks

The navigation logic is embedded in the `display_chapter()` function, simulating hyperlinks. Users can:

- Enter a chapter ID to navigate.
- Use the "B" option to return to the previous chapter.

2.3.6 Testing Navigation

The application begins at the "Introduction" chapter. Users navigate interactively, and the system validates inputs to ensure seamless transitions between chapters.



```
--- Chapter 1: Basics of Python ---
In this chapter, we discuss the basics of Python programming.

Where would you like to go next?

Type 1 to go to Introduction
Type 3 to go to Chapter 2: Data Structures
Type B to go back to the previous chapter
Enter your choice: █
```

Figure 1: Testing Navigation

2.3.7 Enhancing User Experience

To improve usability:

- The "Back" feature was added, enabling users to return to the previous chapter.
- The `clear_screen()` function ensures a clean interface for each chapter display.

2.4 Conclusion

The hypertext-based e-book application effectively demonstrates the concept of interconnected nodes. By implementing features like hyperlinks and backtracking, it provides an interactive reading experience. Further enhancements can expand its functionality and user appeal.