

CLI Comeback !

Développez des applications CLI avec .NET



Tidjani Belmansour

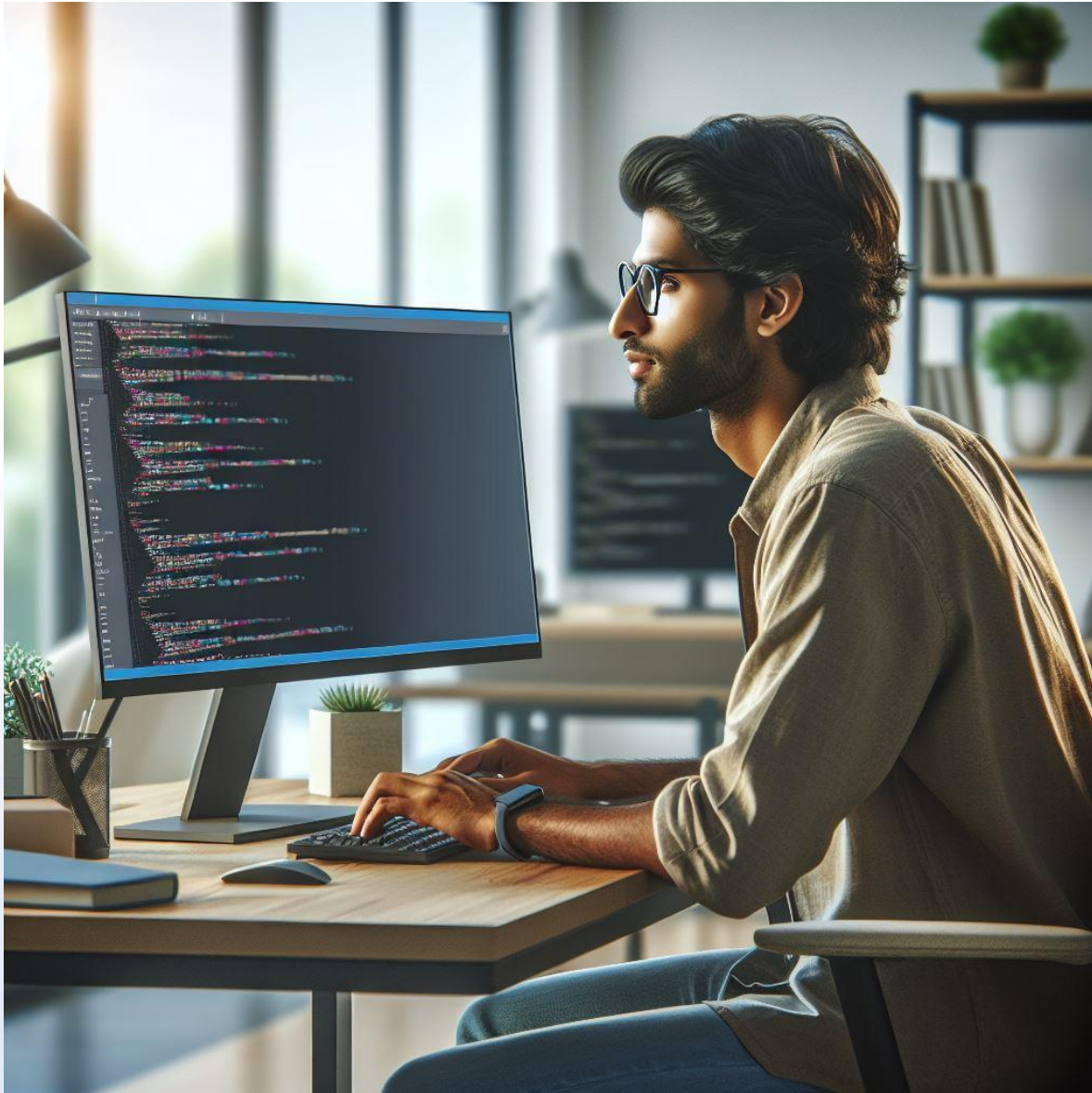
ConFoo.CA

DEVELOPER CONFERENCE

Montréal, Canada, 26-28 Février 2025



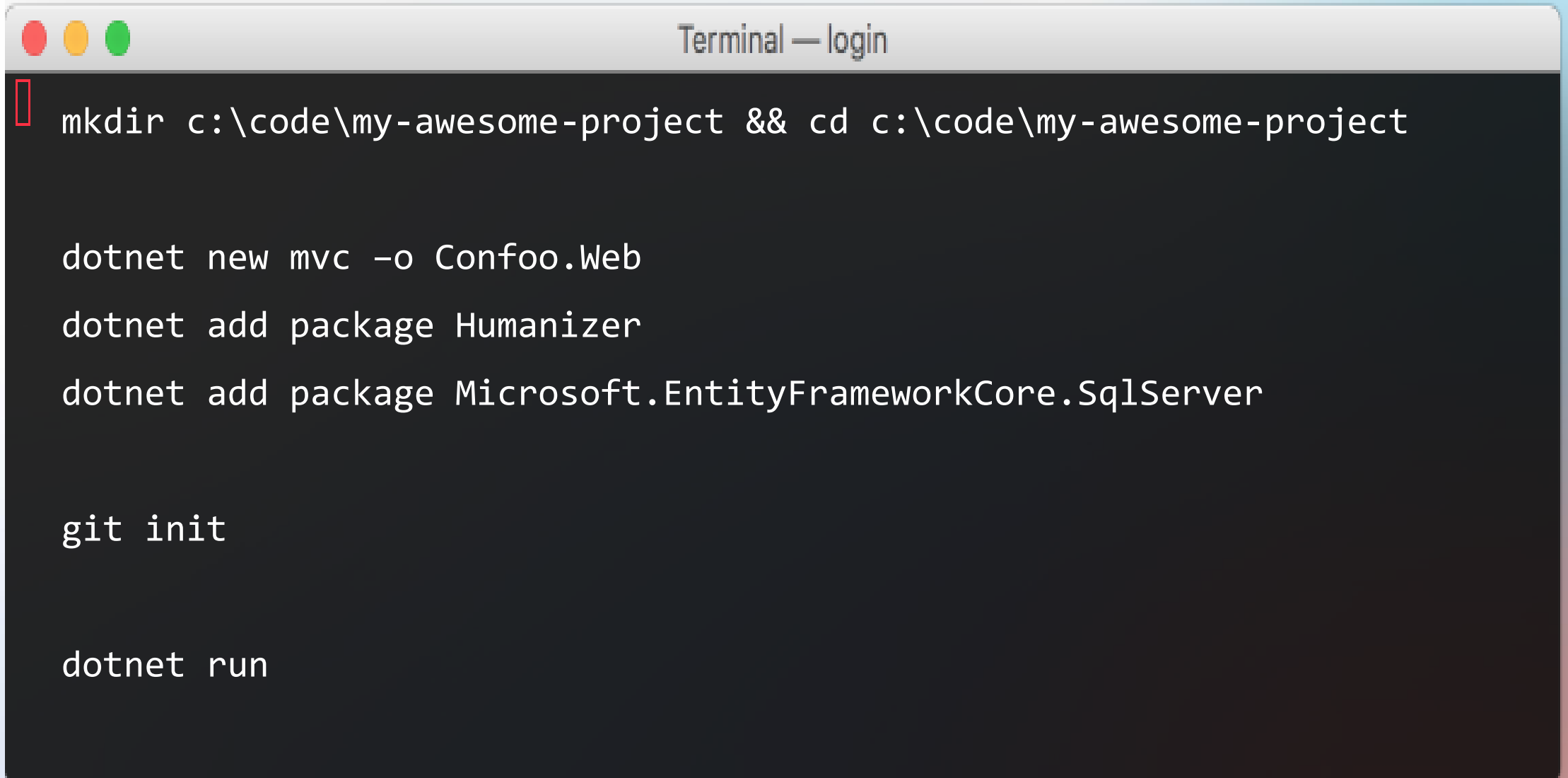
Développeur



IT Pro



Rôle == Développeur



```
Terminal — login

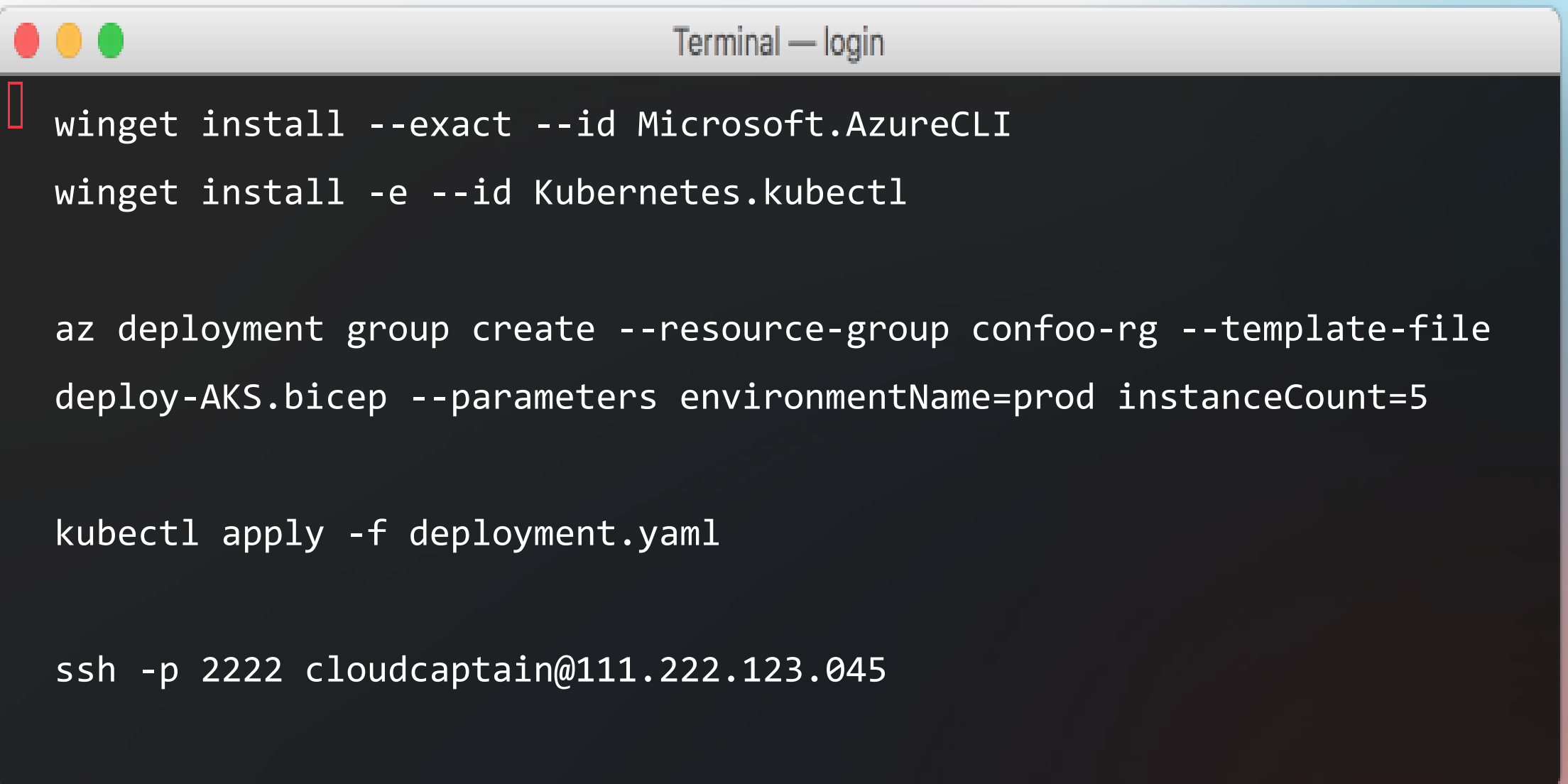
mkdir c:\code\my-awesome-project && cd c:\code\my-awesome-project

dotnet new mvc -o Confoo.Web
dotnet add package Humanizer
dotnet add package Microsoft.EntityFrameworkCore.SqlServer

git init

dotnet run
```


Rôle == IT Pro



```
Terminal — login

winget install --exact --id Microsoft.AzureCLI
winget install -e --id Kubernetes.kubectl

az deployment group create --resource-group confoo-rg --template-file
deploy-AKS.bicep --parameters environmentName=prod instanceCount=5

kubectl apply -f deployment.yaml

ssh -p 2222 cloudcaptain@111.222.123.045
```

Salut, mon nom est ...

Tidjani Belmansour



Directeur, Architecte Cloud @ Cofomo (<https://www.cofomo.com>)



Microsoft Azure MVP



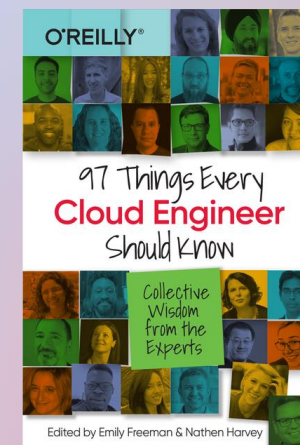
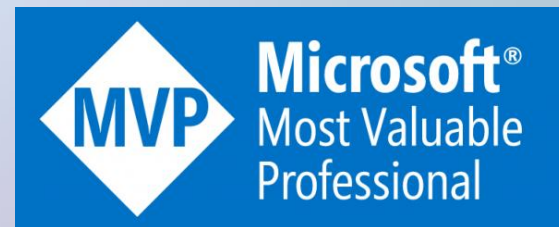
Co-organisateur « Communauté Azure de Québec »
(<https://meetup.com/azureqc>)



/in/tidjani-belmansour



<https://espacenuagic.com> | <https://dev.to/tidjani>



Pourquoi parler d'applications CLI en 2025 ?!

Une application CLI nous permet de ...

- ❖ Garder le **focus** sur la tâche à accomplir
- ❖ **Automatiser** les tâches récurrentes ou à faible valeur ajoutée
- ❖ **Optimiser** la consommation des ressources du système
- ❖ Réaliser des **tâches plus complexes** que via l'interface graphique

Mais on a déjà des GUI ...

Application avec GUI

- ❖ Contrôle limité
- ❖ Consomme plus de ressources
- ❖ Moins de possibilités d'automatisation
- ❖ Les GUI changent souvent

Application CLI

- ❖ Contrôle avancé
- ❖ Consomme moins de ressources
- ❖ Plus de possibilités d'automatisation
- ❖ Les CLI changent moins souvent

Une application CLI: C'est pas juste une application console, ça ?!

Toute application CLI **est** une application console

Mais...

Toute application console **n'est pas** une application CLI

Application CLI *vs* Application console

Application CLI

- ❖ Interaction riche avec l'utilisateur
- ❖ Conçue spécifiquement pour interagir avec un système ou un service particulier

Application console

- ❖ Interaction limitée avec l'utilisateur
- ❖ Exécute une tâche unique

Applications CLI: Pourquoi en .NET ?

Build anything with a unified platform

.NET



Cloud



Web



Desktop



Mobile



Gaming



IoT



AI



Visual Studio



Visual Studio
Code



CLI



GitHub
Copilot



Windows



Linux



macOS



NuGet



GitHub



.NET
Aspire



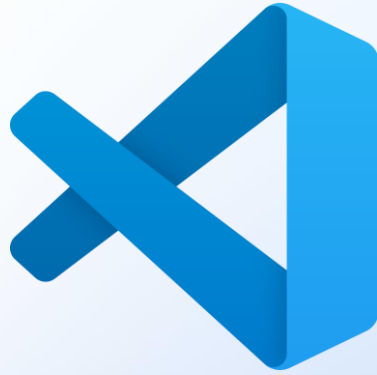
Components, tools,
library vendors

Tools

Operating systems

Ecosystem

Vos applications favorites offrent aussi un CLI !



Même celles-ci !!



<https://github.com/kardolus/chatgpt-cli>

<https://github.com/marcolardera/chatgpt-cli>



<https://github.com/gautamkrishnar/socli>

OK...

Par où je commence ?



Tout commence par une application console ...

```
dotnet new console -n bookmarkr -o bookmarkr --use-program-main
```

```
dotnet run Packt Publishing
```

Program.cs X

Program.cs

```
1 namespace helloConsole;
2
3 class Program
4 {
5     static void Main(string[] args)
6     {
7         Console.WriteLine($"Hello, {args[0]} {args[1]}!");
8     }
9 }
10
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

- PS C:\code\helloConsole> dotnet run Packt Publishing
Hello, Packt Publishing!
- PS C:\code\helloConsole> █

Anatomie d'une application CLI

```
bookmarkr link add --name 'Confoo' --url 'https://confoo.ca/' --category 'Developer Conference'
```

Anatomie d'une application CLI

bookmarkr link add --name 'Confoo' --url 'https://confoo.ca/' --category 'Developer Conference'

Anatomie d'une application CLI

```
bookmarkr link add --name 'Confoo' --url 'https://confoo.ca/' --category 'Developer Conference'
```

Anatomie d'une application CLI

```
bookmarkr link add --name 'Confoo' --url 'https://confoo.ca/' --category 'Developer Conference'
```


Anatomie d'une application CLI

```
bookmarkr link add --name 'Confoo' --url 'https://confoo.ca/' --category 'Developer Conference'
```

Anatomie d'une application CLI

```
bookmarkr link add -n 'Confoo' -u 'https://confoo.ca/' -C 'Developer Conference'
```

**Ça ne m'a pas l'air
simple de gérer tout ça
depuis les arguments
d'une application
console...**

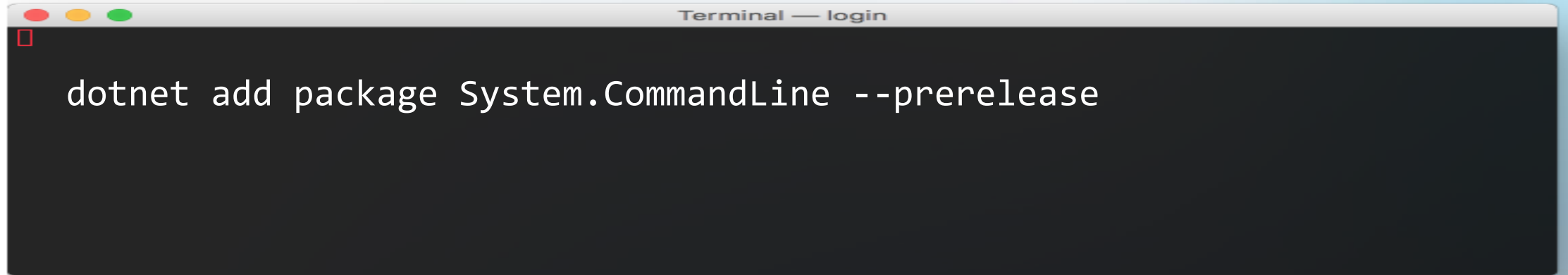


From console to CLI

Hello, `System.CommandLine` !

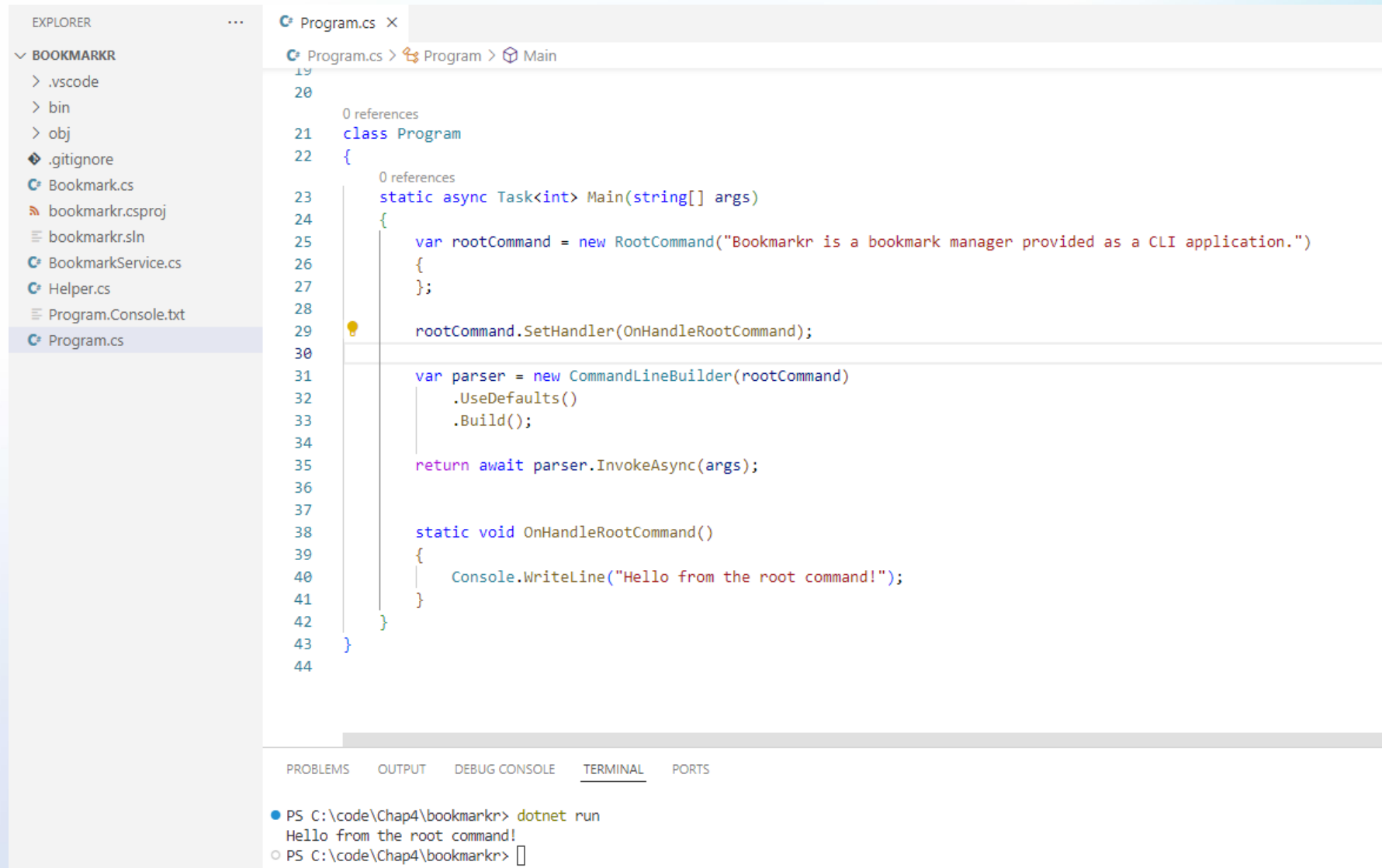
- ❖ Projet développé par Microsoft et la communauté
- ❖ Projet supporté par la .NET Foundation
- ❖ Ensemble riche de fonctionnalités
- ❖ Support de conventions spécifiques aux plateformes (ex. insensibilité à la casse sous Windows)

Ajouter la librairie à notre projet

A terminal window with a dark background and a light gray title bar. The title bar contains the text "Terminal — login" and three colored window control buttons (red, yellow, green) on the left. A red cursor icon is visible on the left side of the terminal area. The command "dotnet add package System.CommandLine --prerelease" is entered in white text.

```
Terminal — login  
dotnet add package System.CommandLine --prerelease
```


Créer notre première commande (root command)



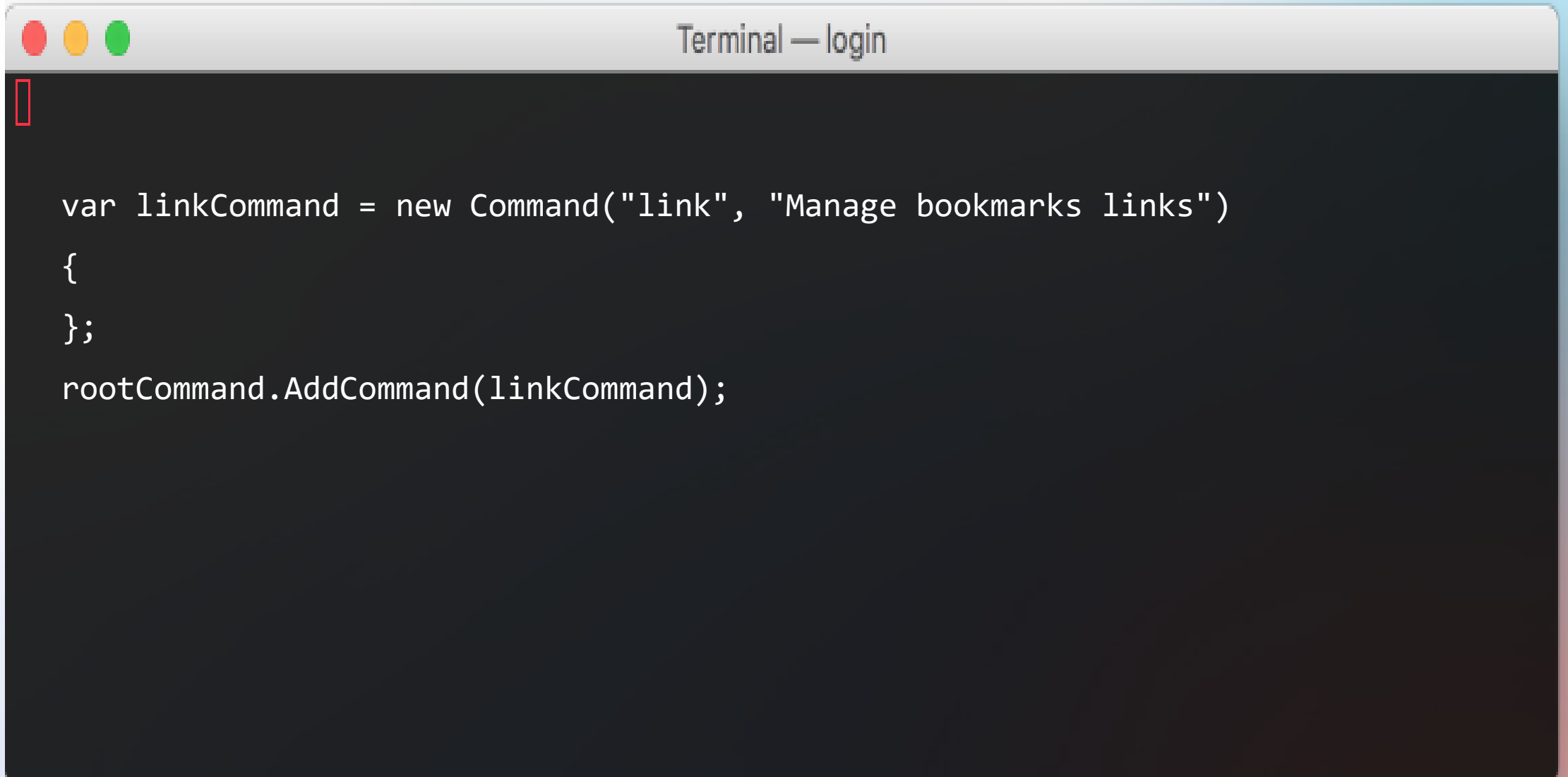
The screenshot displays the Visual Studio Code interface with a C# project named 'BOOKMARKR'. The Explorer sidebar on the left shows the project structure, including files like .vscode, bin, obj, .gitignore, Bookmark.cs, bookmarkr.csproj, bookmarkr.sln, BookmarkService.cs, Helper.cs, Program.Console.txt, and Program.cs. The Program.cs file is open in the editor, showing the following code:

```
19
20
21 class Program
22 {
23     0 references
24     static async Task<int> Main(string[] args)
25     {
26         var rootCommand = new RootCommand("Bookmarkr is a bookmark manager provided as a CLI application.")
27         {
28         };
29         rootCommand.SetHandler(OnHandleRootCommand);
30
31         var parser = new CommandLineBuilder(rootCommand)
32             .UseDefaults()
33             .Build();
34
35         return await parser.InvokeAsync(args);
36
37         static void OnHandleRootCommand()
38         {
39             Console.WriteLine("Hello from the root command!");
40         }
41     }
42 }
43
44
```

At the bottom of the window, the TERMINAL tab is active, showing the execution of the application:

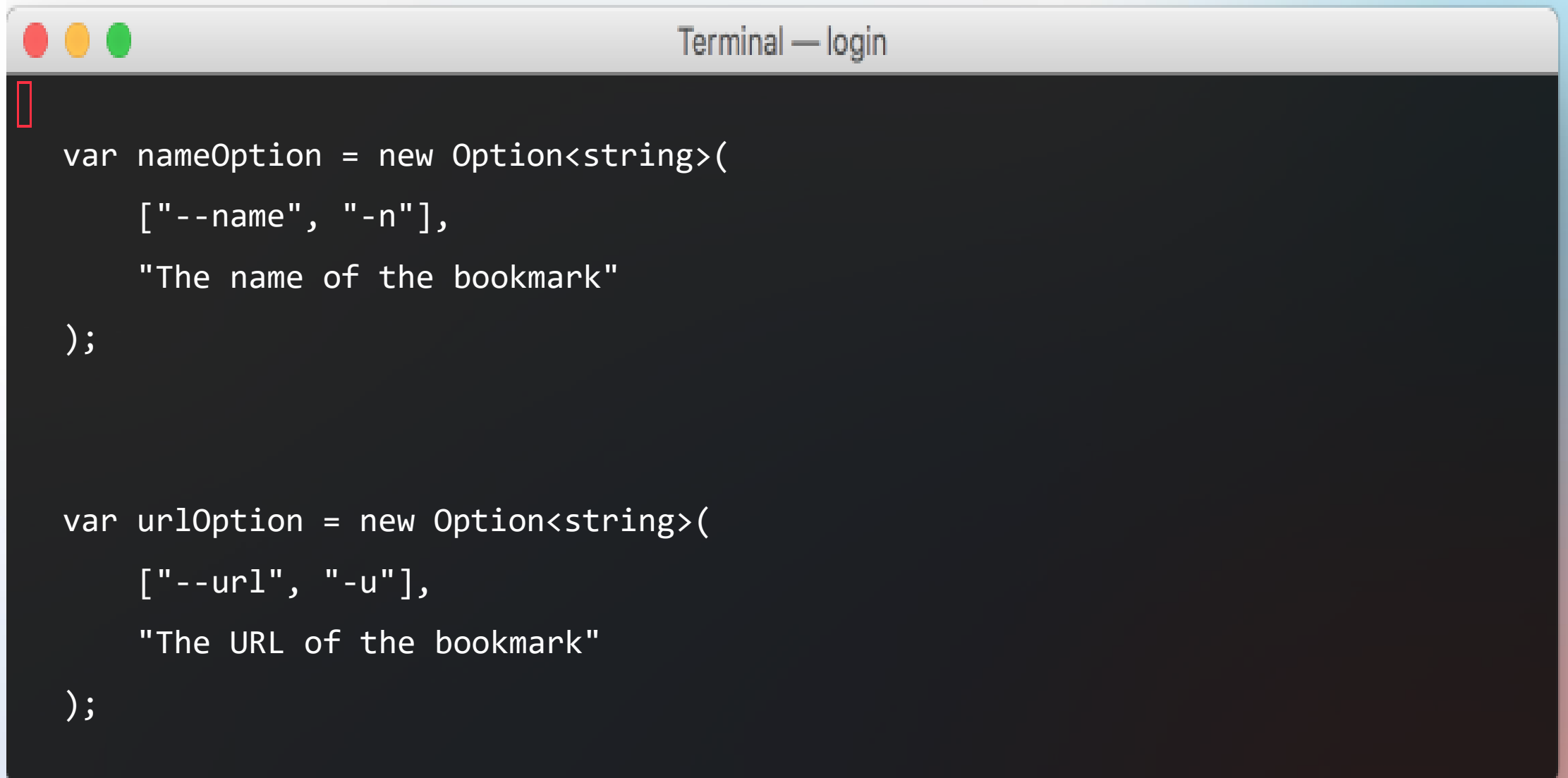
```
PS C:\code\Chap4\bookmarkr> dotnet run
Hello from the root command!
PS C:\code\Chap4\bookmarkr> 
```

Ajouter une 2^e commande

A screenshot of a macOS-style terminal window. The title bar is light gray and contains the text "Terminal — login" on the right and three colored window control buttons (red, yellow, green) on the left. The terminal area has a dark gray background. A red cursor is positioned at the start of the first line of code. The code is written in a light gray monospaced font and consists of three lines: a variable declaration, an opening curly brace, and a method call.

```
var linkCommand = new Command("link", "Manage bookmarks links")  
{  
};  
rootCommand.AddCommand(linkCommand);
```

Ajouter des paramètres aux commandes

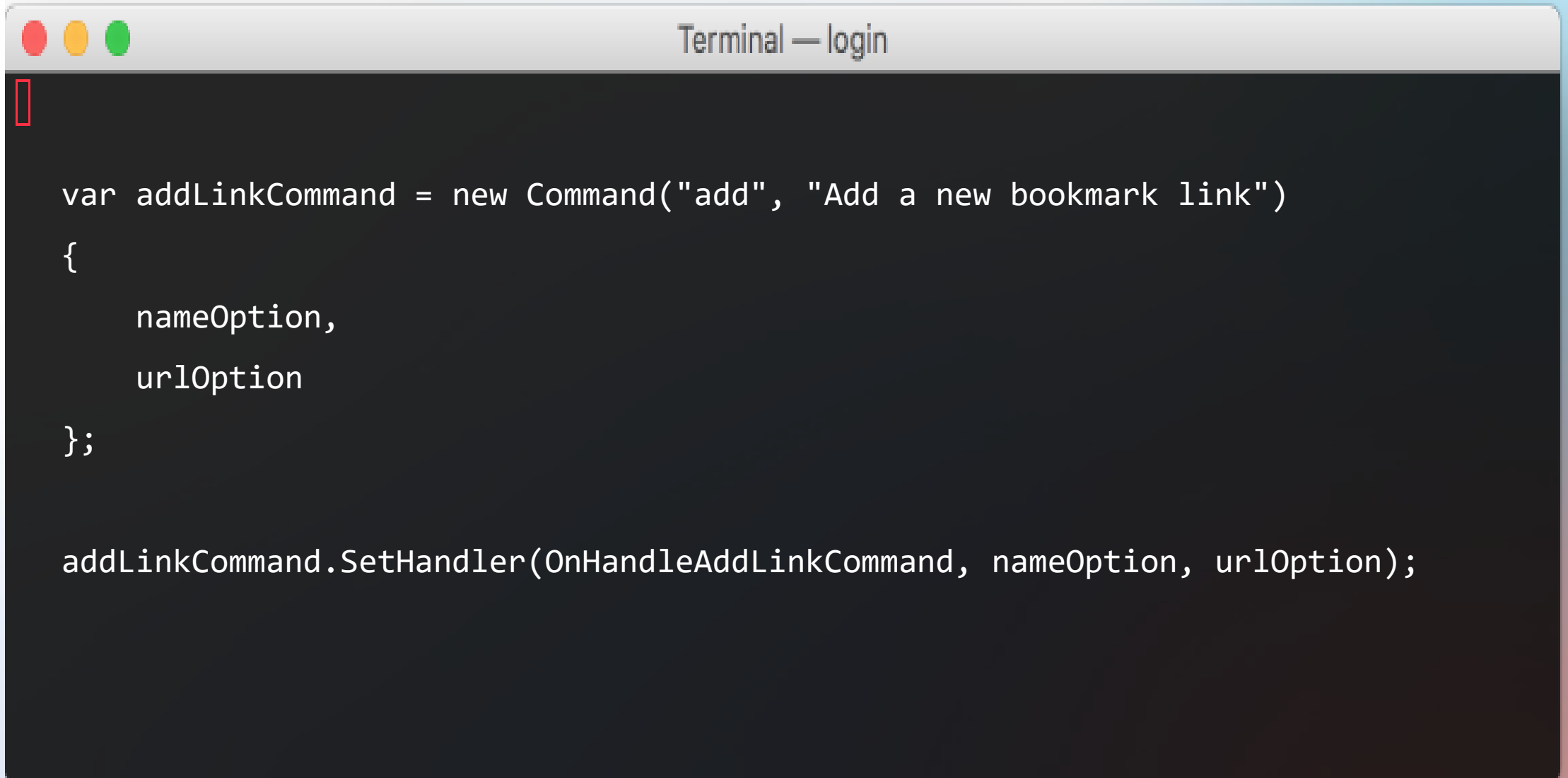


```
Terminal — login

var nameOption = new Option<string>(
    ["--name", "-n"],
    "The name of the bookmark"
);

var urlOption = new Option<string>(
    ["--url", "-u"],
    "The URL of the bookmark"
);
```

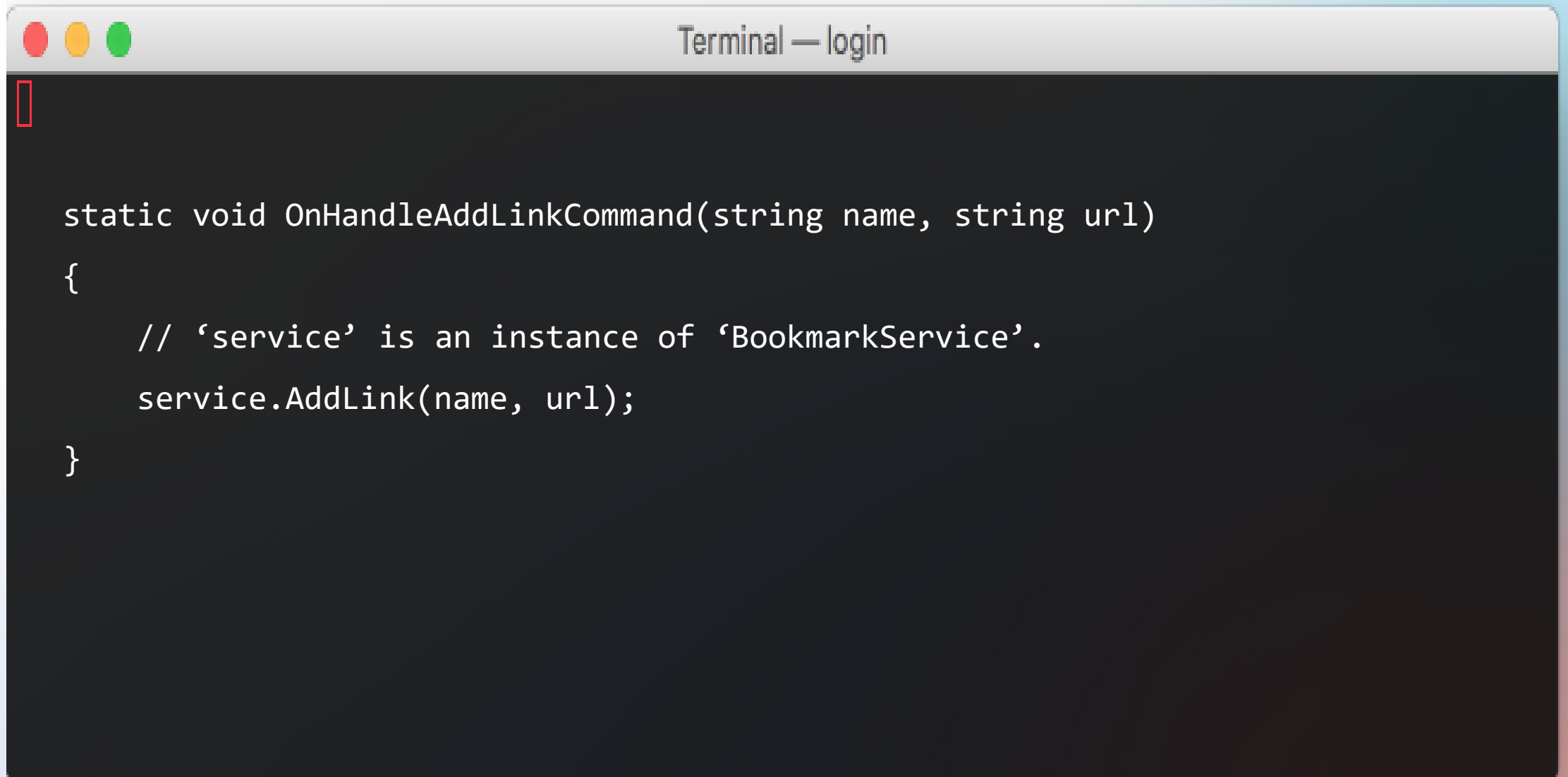
Ajouter une sous-commande

A screenshot of a macOS-style terminal window with a title bar containing three colored window control buttons (red, yellow, green) and the text "Terminal — login". The terminal has a dark background and displays C# code. A red cursor is positioned at the start of the first line of code.

```
var addLinkCommand = new Command("add", "Add a new bookmark link")
{
    nameOption,
    urlOption
};

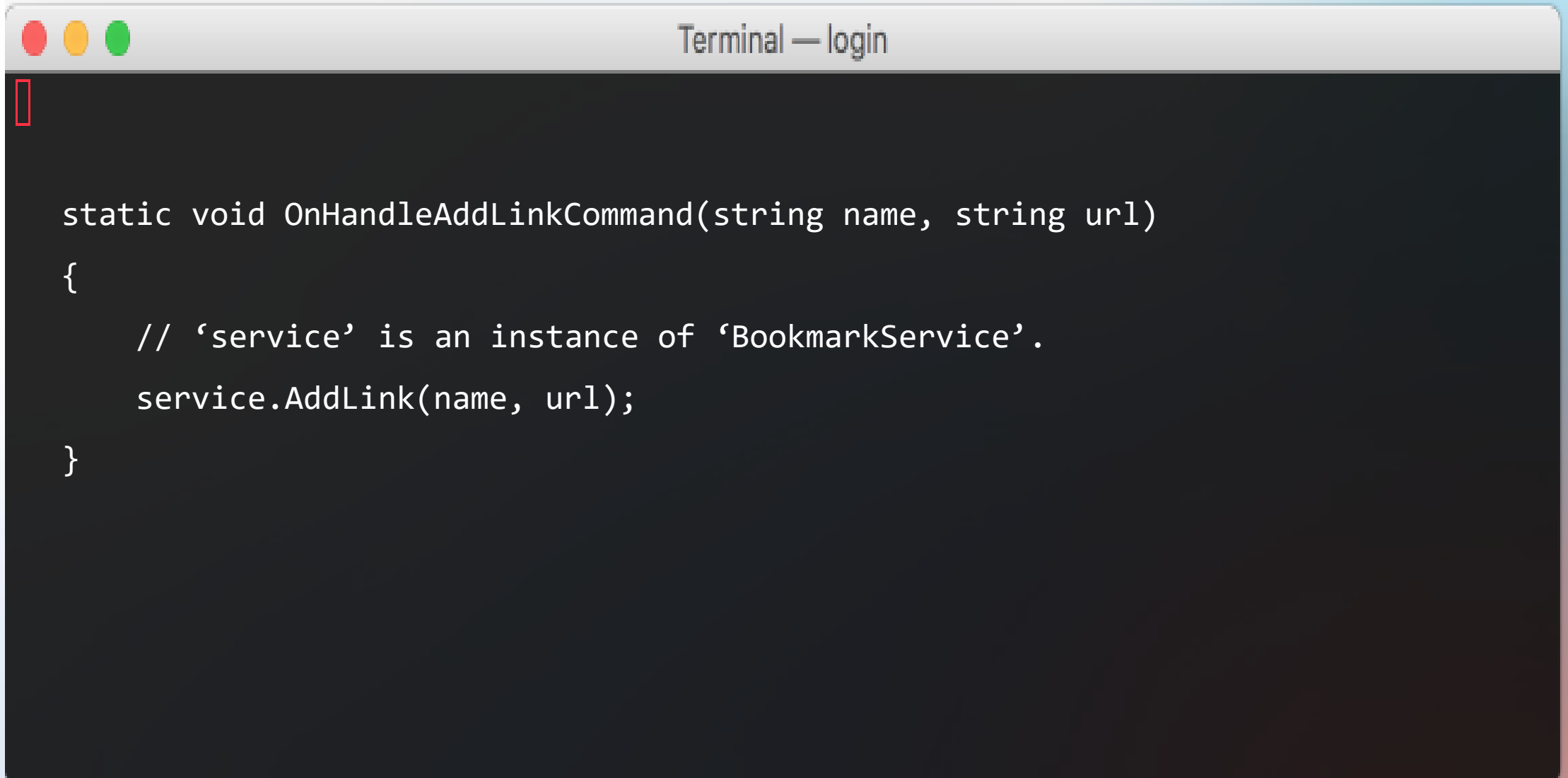
addLinkCommand.SetHandler(OnHandleAddLinkCommand, nameOption, urlOption);
```

Ajouter le "command handler"

A terminal window with a title bar that says "Terminal — login". The window has three colored window control buttons (red, yellow, green) on the left. The main area is dark gray and contains C# code. A red cursor is at the start of the first line.

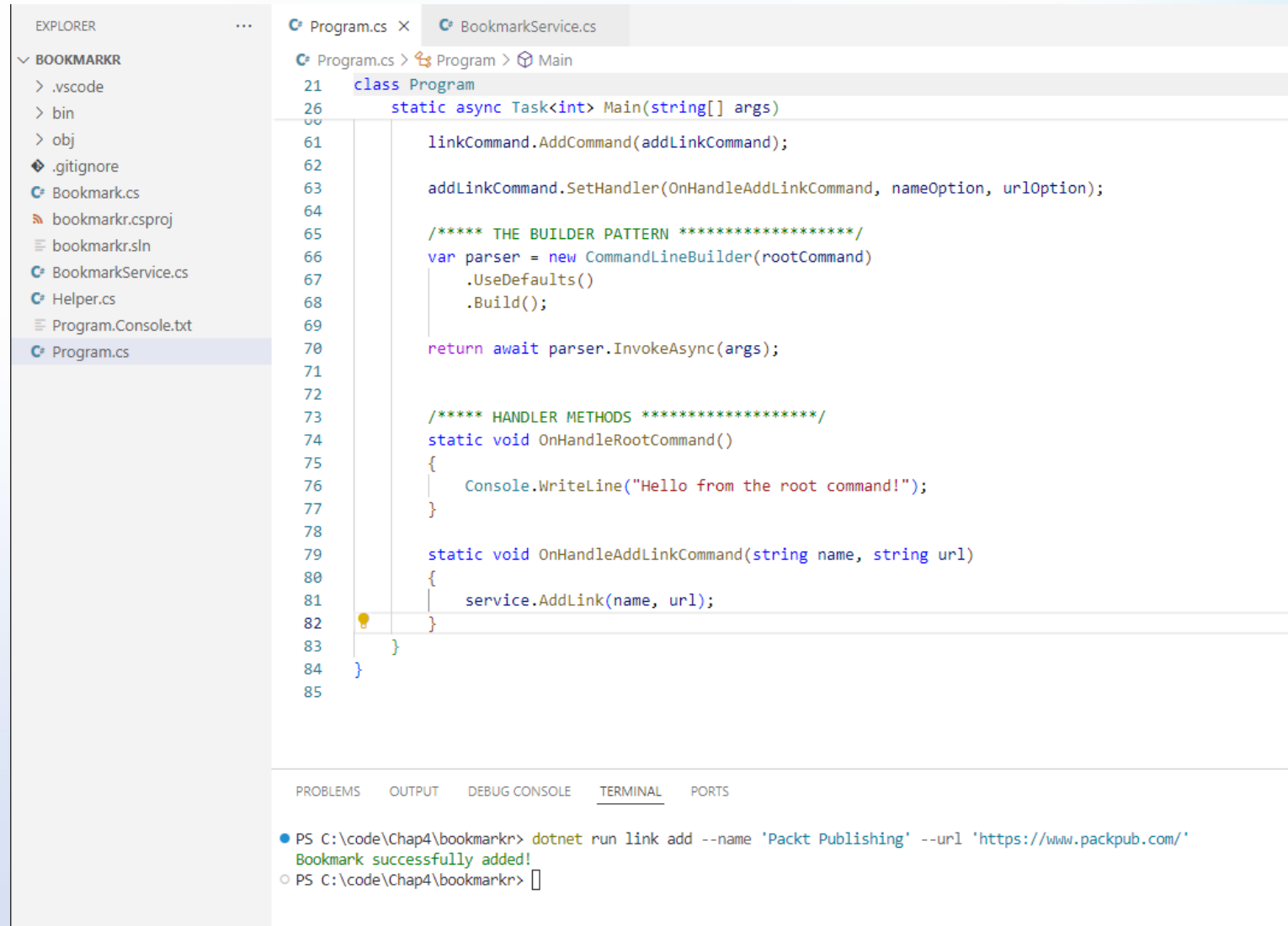
```
static void OnHandleAddLinkCommand(string name, string url)
{
    // 'service' is an instance of 'BookmarkService'.
    service.AddLink(name, url);
}
```


Ajouter le "command handler"

A screenshot of a macOS-style terminal window. The title bar is light gray and contains the text "Terminal — login" on the right and three colored window control buttons (red, yellow, green) on the left. The terminal area has a dark gray background. A red rectangular cursor is positioned at the start of the first line of code. The code is written in a light gray monospaced font and defines a static method `OnHandleAddLinkCommand` that takes `string name` and `string url` as parameters. Inside the method, there is a comment `// 'service' is an instance of 'BookmarkService'.` followed by the call `service.AddLink(name, url);`.

```
static void OnHandleAddLinkCommand(string name, string url)
{
    // 'service' is an instance of 'BookmarkService'.
    service.AddLink(name, url);
}
```

Exécuter notre commande



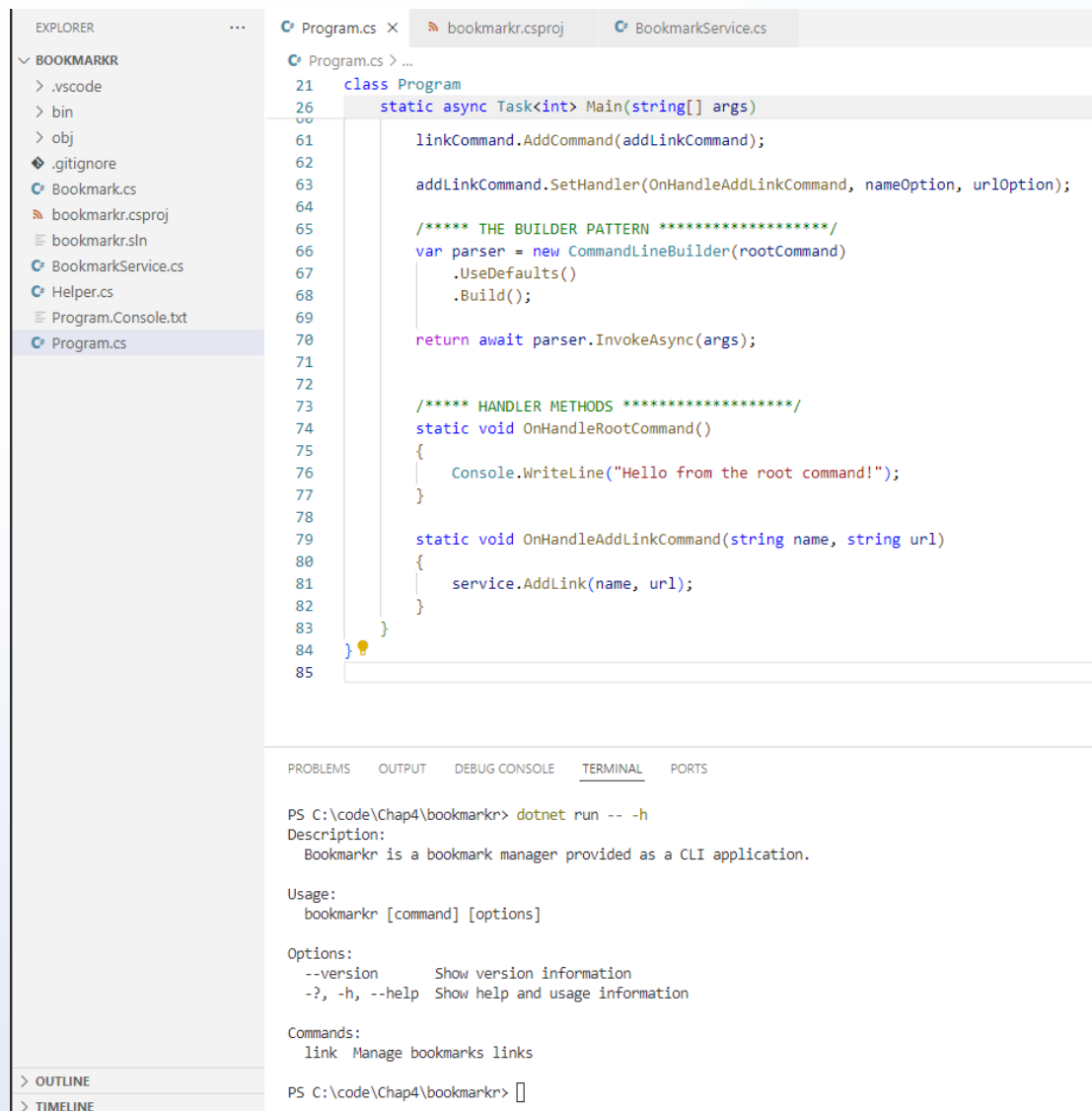
The screenshot displays the Visual Studio Code interface. On the left, the Explorer pane shows a project named 'BOOKMARKR' with files like '.vscode', 'bin', 'obj', '.gitignore', 'Bookmark.cs', 'bookmarkr.csproj', 'bookmarkr.sln', 'BookmarkService.cs', 'Helper.cs', 'Program.Console.txt', and 'Program.cs'. The 'Program.cs' file is selected and open in the editor. The editor shows the following C# code:

```
21 class Program
26 static async Task<int> Main(string[] args)
61     linkCommand.AddCommand(addLinkCommand);
62
63     addLinkCommand.SetHandler(OnHandleAddLinkCommand, nameOption, urlOption);
64
65     /**** THE BUILDER PATTERN *****/
66     var parser = new CommandLineBuilder(rootCommand)
67         .UseDefaults()
68         .Build();
69
70     return await parser.InvokeAsync(args);
71
72
73     /**** HANDLER METHODS *****/
74     static void OnHandleRootCommand()
75     {
76         Console.WriteLine("Hello from the root command!");
77     }
78
79     static void OnHandleAddLinkCommand(string name, string url)
80     {
81         service.AddLink(name, url);
82     }
83 }
84
85
```

At the bottom of the window, the TERMINAL pane is active, showing the command prompt output:

```
PS C:\code\Chap4\bookmarkr> dotnet run link add --name 'Packt Publishing' --url 'https://www.packpub.com/'
Bookmark successfully added!
PS C:\code\Chap4\bookmarkr>
```

Help !



EXPLORER

- BOOKMARKR
 - .vscode
 - bin
 - obj
 - .gitignore
 - Bookmark.cs
 - bookmarkr.csproj
 - bookmarkr.sln
 - BookmarkService.cs
 - Helper.cs
 - Program.Console.txt
 - Program.cs

```
21 class Program
26 static async Task<int> Main(string[] args)
61 linkCommand.AddCommand(addLinkCommand);
62
63 addLinkCommand.SetHandler(OnHandleAddLinkCommand, nameOption, urlOption);
64
65 /***** THE BUILDER PATTERN *****/
66 var parser = new CommandLineBuilder(rootCommand)
67     .UseDefaults()
68     .Build();
69
70 return await parser.InvokeAsync(args);
71
72
73 /***** HANDLER METHODS *****/
74 static void OnHandleRootCommand()
75 {
76     Console.WriteLine("Hello from the root command!");
77 }
78
79 static void OnHandleAddLinkCommand(string name, string url)
80 {
81     service.AddLink(name, url);
82 }
83
84 }
85
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

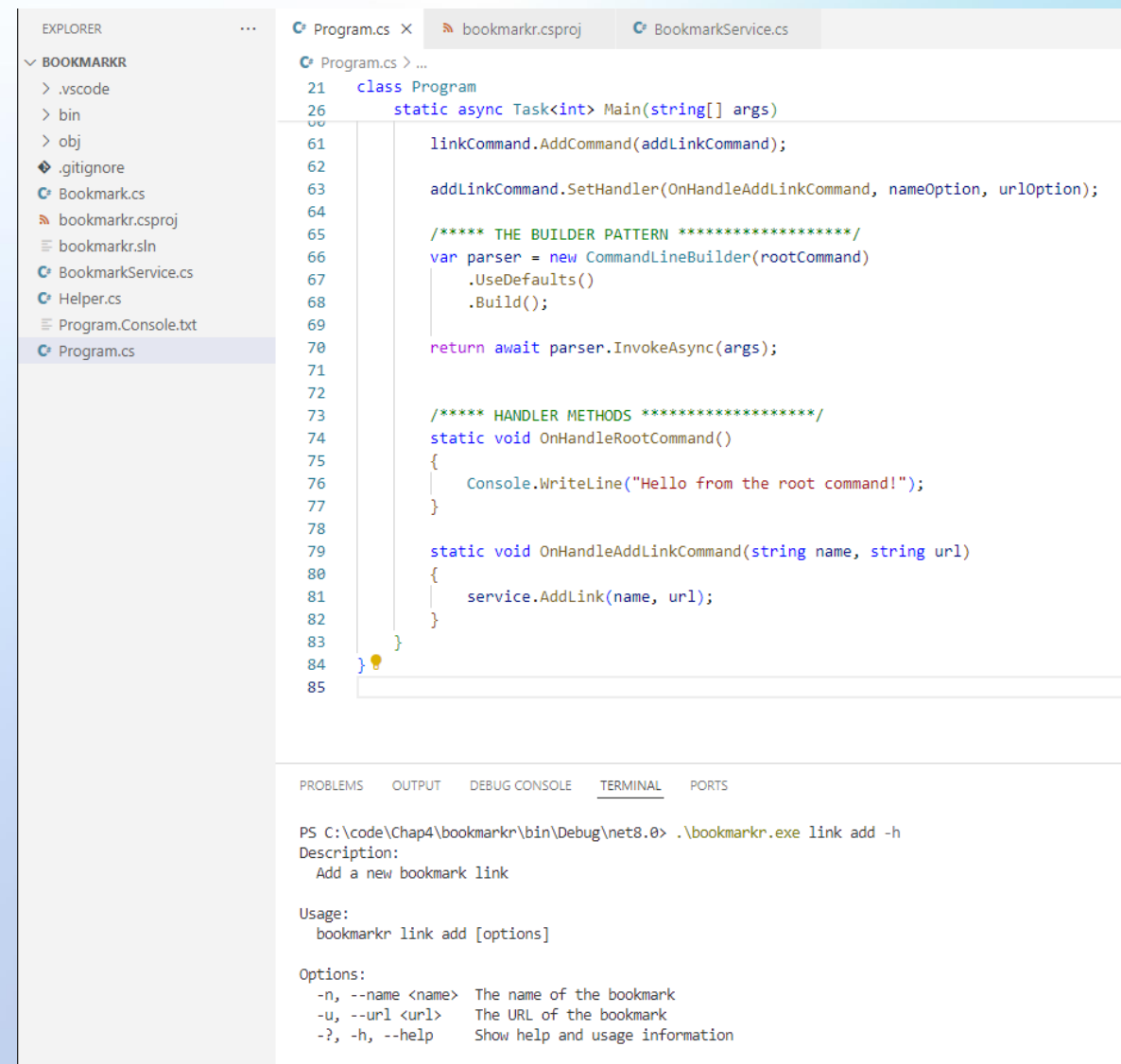
PS C:\code\Chap4\bookmarkr> dotnet run -- -h
Description:
Bookmarkr is a bookmark manager provided as a CLI application.

Usage:
bookmarkr [command] [options]

Options:
--version Show version information
-?, -h, --help Show help and usage information

Commands:
link Manage bookmarks links

PS C:\code\Chap4\bookmarkr>



EXPLORER

- BOOKMARKR
 - .vscode
 - bin
 - obj
 - .gitignore
 - Bookmark.cs
 - bookmarkr.csproj
 - bookmarkr.sln
 - BookmarkService.cs
 - Helper.cs
 - Program.Console.txt
 - Program.cs

```
21 class Program
26 static async Task<int> Main(string[] args)
61 linkCommand.AddCommand(addLinkCommand);
62
63 addLinkCommand.SetHandler(OnHandleAddLinkCommand, nameOption, urlOption);
64
65 /***** THE BUILDER PATTERN *****/
66 var parser = new CommandLineBuilder(rootCommand)
67     .UseDefaults()
68     .Build();
69
70 return await parser.InvokeAsync(args);
71
72
73 /***** HANDLER METHODS *****/
74 static void OnHandleRootCommand()
75 {
76     Console.WriteLine("Hello from the root command!");
77 }
78
79 static void OnHandleAddLinkCommand(string name, string url)
80 {
81     service.AddLink(name, url);
82 }
83
84 }
85
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\code\Chap4\bookmarkr\bin\Debug\net8.0> .\bookmarkr.exe link add -h
Description:
Add a new bookmark link

Usage:
bookmarkr link add [options]

Options:
-n, --name <name> The name of the bookmark
-u, --url <url> The URL of the bookmark
-?, -h, --help Show help and usage information

▼ Chapter09/bookmarkr

> .vscode

▼ Commands

▼ Export

ExportCommand.cs

> Import

> Interactive

▼ Link

> Add

LinkCommand.cs

> Sync

▼ ServiceAgents/BookmarkrSyncrServiceAgent

BookmarkrSyncrServiceAgent.cs

IBookmarkrSyncrServiceAgent.cs

▼ Services/BookmarkService

BookmarkService.cs

IBookmarkService.cs

Modularité & Extensibilité

<https://github.com/PacktPublishing/Building-CLI-Applications-with-C-Sharp-and-.NET/blob/main/Chapter07/bookmarkr/Program.cs>

Modularité & Extensibilité

Terminal — login

```
public class LinkCommand : Command
{
    #region Properties
    private readonly IBookmarkService _service;
    #endregion

    #region Constructor
    public LinkCommand(IBookmarkService service, string name, string? description = null)
        : base(name, description)
    {
        _service = service;
        AddCommand(new LinkAddCommand(_service, "add", "Add a new bookmark link"));
    }
    #endregion

    #region Options
    #endregion

    #region Handler method
    #endregion
}
```

Modularité & Extensibilité

Terminal — login

```
class Program
{
    static async Task<int> Main(string[] args)
    {
        // ...
        /** CONFIGURE DEPENDENCY INJECTION FOR THE IBookmarkService **/
        var host = Host.CreateDefaultBuilder(args)
            .ConfigureServices((hostContext, services) =>
            {
                // Register your services here
                services.AddSingleton<IBookmarkService, BookmarkService>();
            })
            .Build();

        _service = host.Services.GetRequiredService<IBookmarkService>();

        /** REGISTER SUBCOMMANDS OF THE ROOT COMMAND **/
        rootCommand.AddCommand(new ExportCommand(_service, "export", "Exports all bookmarks to a file"));

        /** THE BUILDER PATTERN **/
        // code removed for brevity.
    }
}
```

Packaging et déploiement

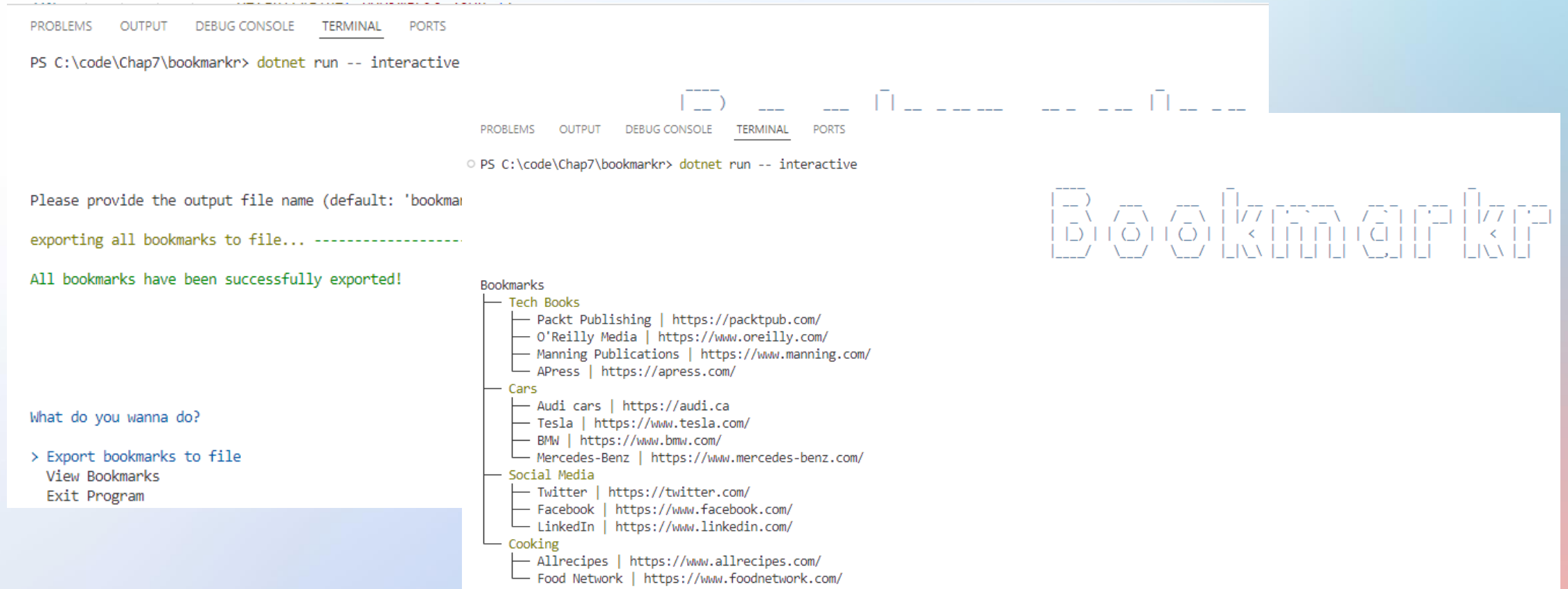


apt-get

Oui, mais ça manque
de panache un CLI ...



Du graphisme dans les CLI ? Pas de problème !



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\code\Chap7\bookmarkr> dotnet run -- interactive

Please provide the output file name (default: 'bookmarkr')
exporting all bookmarks to file... -----
All bookmarks have been successfully exported!

What do you wanna do?
> Export bookmarks to file
  View Bookmarks
  Exit Program
```

```
Bookmarks
├── Tech Books
│   ├── Packt Publishing | https://packtpub.com/
│   ├── O'Reilly Media | https://www.oreilly.com/
│   ├── Manning Publications | https://www.manning.com/
│   └── APress | https://apress.com/
├── Cars
│   ├── Audi cars | https://audi.ca
│   ├── Tesla | https://www.tesla.com/
│   ├── BMW | https://www.bmw.com/
│   └── Mercedes-Benz | https://www.mercedes-benz.com/
├── Social Media
│   ├── Twitter | https://twitter.com/
│   ├── Facebook | https://www.facebook.com/
│   └── LinkedIn | https://www.linkedin.com/
└── Cooking
    ├── Allrecipes | https://www.allrecipes.com/
    └── Food Network | https://www.foodnetwork.com/
```

Bookmarkr

Du graphisme dans les CLI ? Pas de problème !

<https://spectreconsole.net/>

Spectre.Console Features

Colors

- ✓ 2-bit color
- ✓ 3-bit color
- ✓ 4-bit color
- ✓ 8-bit color
- ✓ Truecolor (16.7 million)
- ✓ Automatic color conversion



OS Windows macOS Linux

Styles All ansi styles: bold, dim, italic, underline, strikethrough, reverse, and even blink.

Text Word wrap text. Justify left, center or right.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque in metus sed sapien ultricies pretium a at justo. Maecenas luctus velit et auctor maximus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque in metus sed sapien ultricies pretium a at justo. Maecenas luctus velit et auctor maximus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque in metus sed sapien ultricies pretium a at justo. Maecenas luctus velit et auctor maximus.

Markup Spectre.Console supports a simple bbcode like markup for color, style, and emoji! 🍌 🍌 🍌 🍌

Tables and Trees

Foo	Bar
Baz	<div>Overview</div> <div>Files<ul style="list-style-type: none">src<ul style="list-style-type: none">foo<ul style="list-style-type: none">bar.csbaz<ul style="list-style-type: none">qux<ul style="list-style-type: none">corgi.txtwaldo.xml</div> <div>3 Files, 225 KiB</div>
Qux	Corgi

Charts



■ C# 82% ■ PowerShell 13% ■ Bash 5%



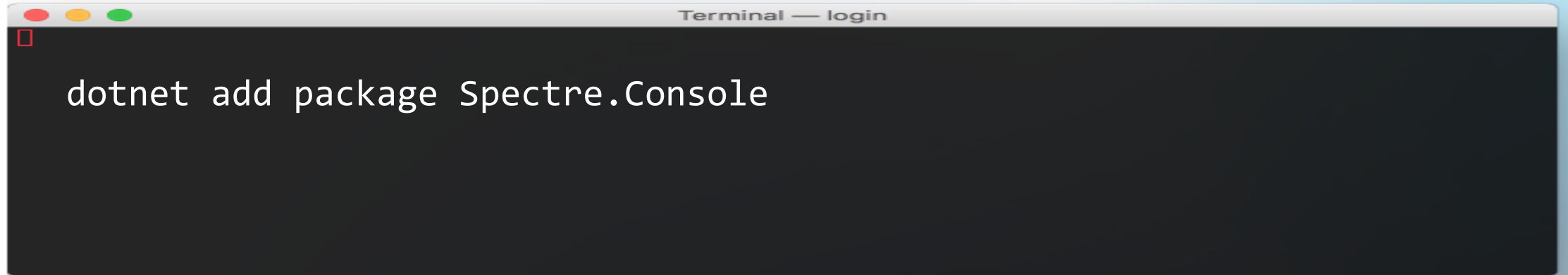
Apple 32
Oranges 13
Bananas 22

Exceptions

```
System.InvalidOperationException: Something went very wrong!  
at Demo.ExceptionGenerator.SomeOperationGoingWrong() in  
C:\Users\Patrik\Source\github\patriksvensson\spectre.console\examples\Console\Demo\  
ExceptionGenerator.cs:28  
at Demo.ExceptionGenerator.SomeOperation() in  
C:\Users\Patrik\Source\github\patriksvensson\spectre.console\examples\Console\Demo\  
ExceptionGenerator.cs:23  
at Demo.ExceptionGenerator.GenerateException() in  
C:\Users\Patrik\Source\github\patriksvensson\spectre.console\examples\Console\Demo\  
ExceptionGenerator.cs:12
```

+ Much more! Tables, Grids, Trees, Progress bars, Status, Bar charts, Calendars, Figlet, Images, Text prompts, List boxes, Separators, Pretty exceptions, Canvas, CLI parsing

Ajouter la librairie à notre projet

A terminal window with a dark background and a light gray title bar. The title bar contains the text "Terminal — login" and three colored window control buttons (red, yellow, green) on the left. A small red cursor icon is visible on the left side of the terminal. The command "dotnet add package Spectre.Console" is entered in white text.

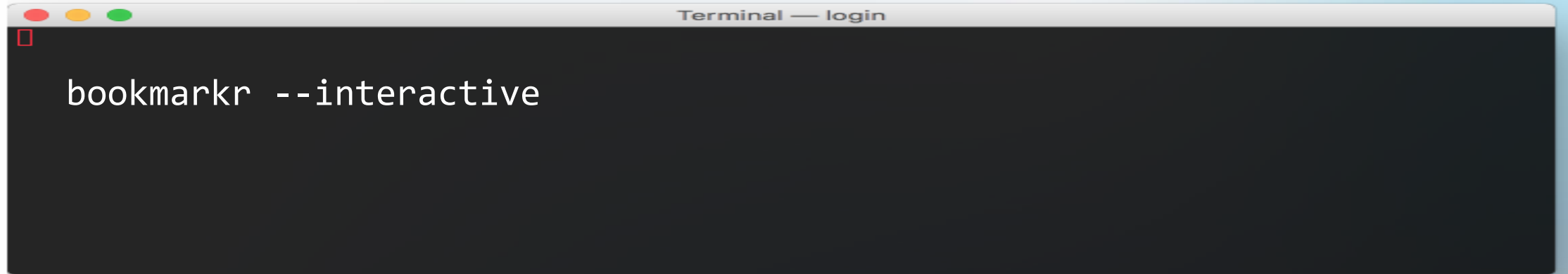
```
dotnet add package Spectre.Console
```

Pourquoi Spectre.Console ?

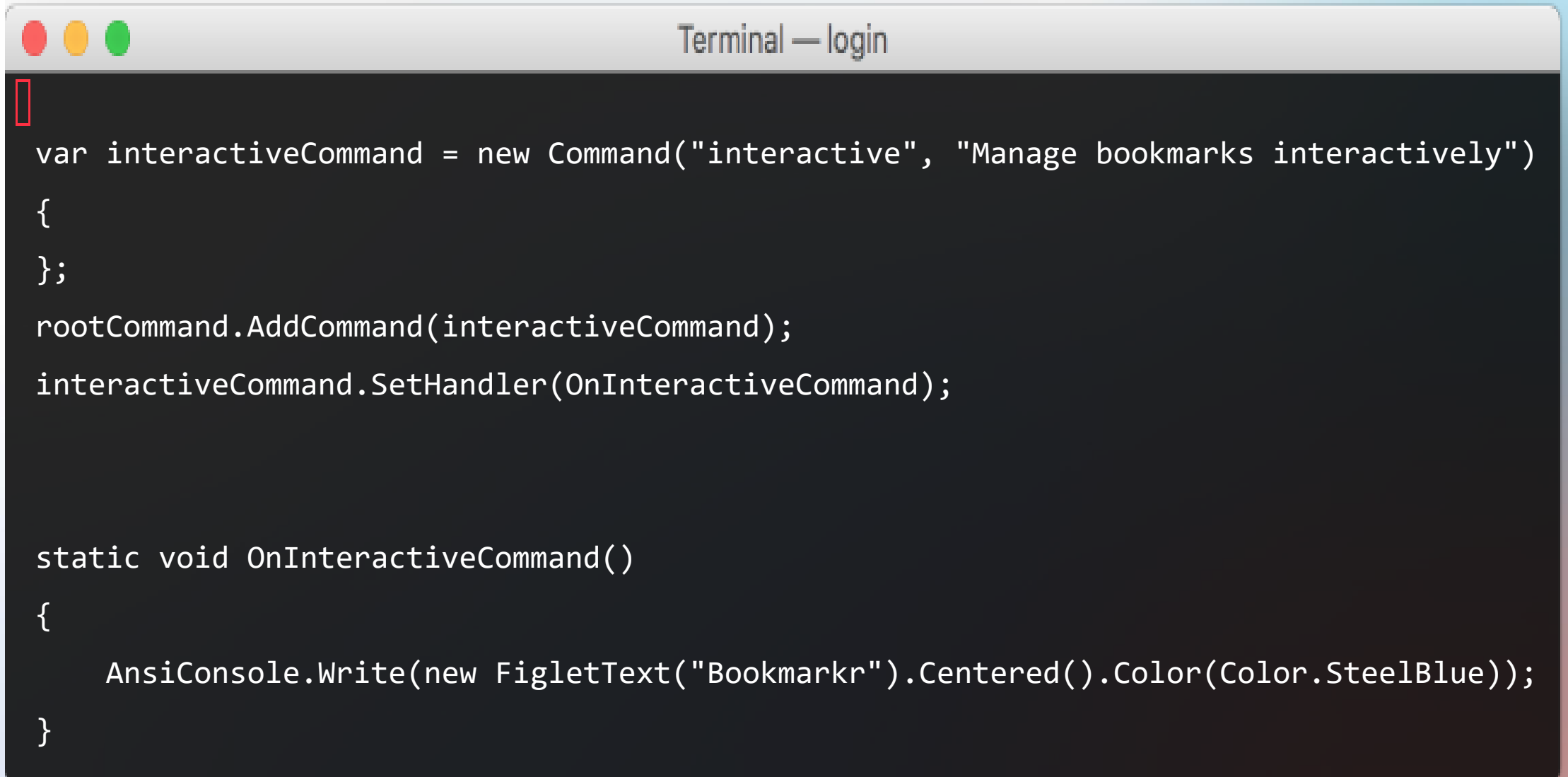
Une librairie complète !

- ❖ Intuitive et facile à prendre en main
- ❖ Liste impressionnantes de contrôles graphiques
- ❖ Détection des capacités du terminal et fallback lorsque requis
- ❖ *Test-Friendly* (IAnsiConsole, ...)

Ajoutons du graphisme à notre CLI !



Ajoutons un FIGlet à notre CLI !



```
Terminal — login

var interactiveCommand = new Command("interactive", "Manage bookmarks interactively")
{
};

rootCommand.AddCommand(interactiveCommand);
interactiveCommand.SetHandler(OnInteractiveCommand);

static void OnInteractiveCommand()
{
    AnsiConsole.Write(new FigletText("Bookmarkr").Centered().Color(Color.SteelBlue));
}
```

Ajoutons un FIGlet à notre CLI !

```
280
281 ✓
282 {
283     AnsiConsole.Write(new FigletText("Bookmark").Centered().Color(Color.SteelBlue));
284 }
285 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

● PS C:\code\Chap7\bookmarkr> dotnet run -- interactive

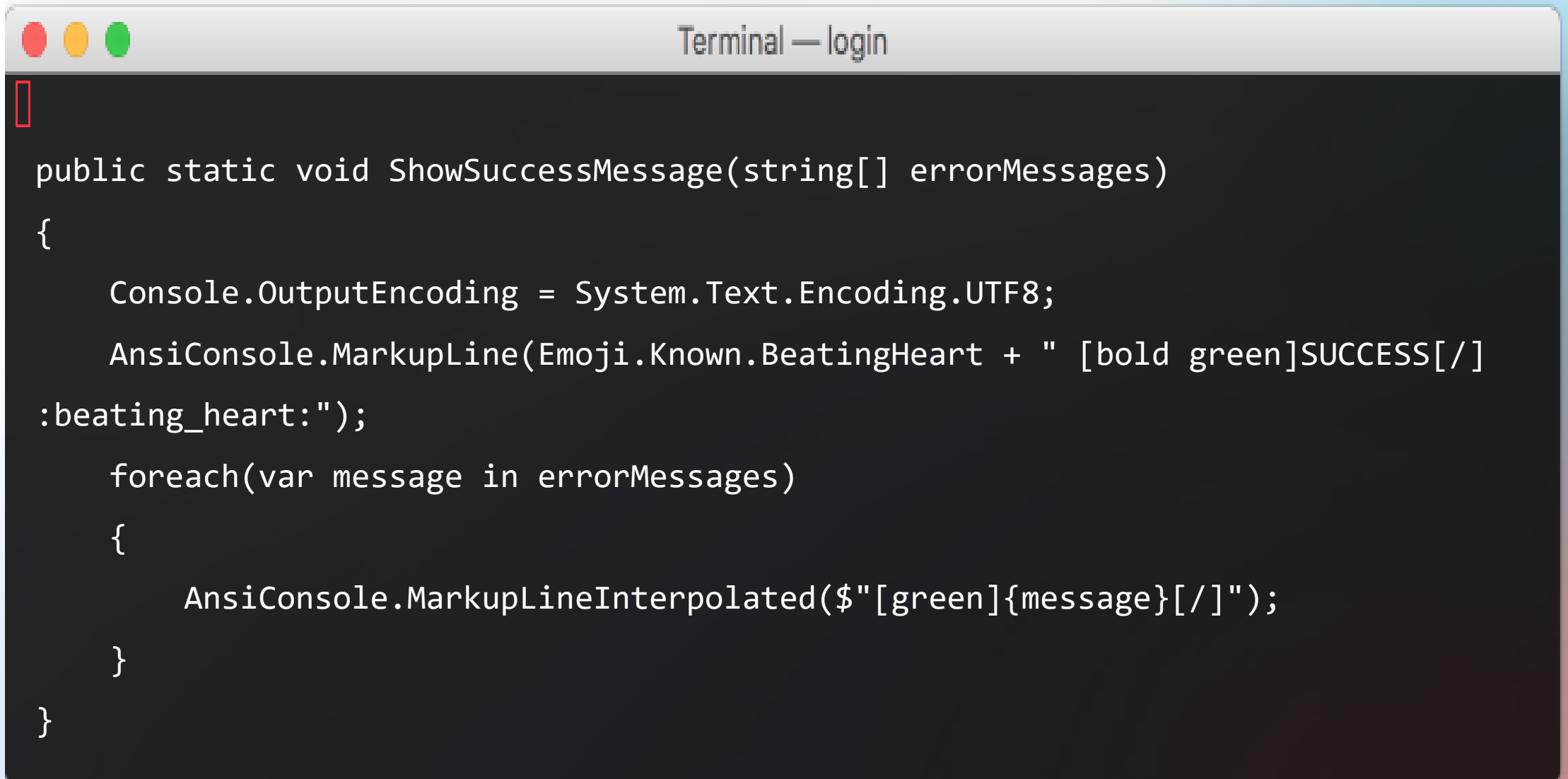
Bookmarkr

Performing shutdown tasks...

○ PS C:\code\Chap7\bookmarkr>

○ PS C:\code\Chap7\bookmarkr>

Ajoutons des emojis à notre CLI !

A terminal window with a title bar that says "Terminal — login". The window has three colored window control buttons (red, yellow, green) on the left. The main area is dark grey and contains C# code. A red cursor is at the start of the first line.

```
public static void ShowSuccessMessage(string[] errorMessages)
{
    Console.OutputEncoding = System.Text.Encoding.UTF8;
    AnsiConsole.MarkupLine(Emoji.Known.BeatingHeart + " [bold green]SUCCESS[/]
:beating_heart:");
    foreach(var message in errorMessages)
    {
        AnsiConsole.MarkupLineInterpolated($"[green]{message}[/]");
    }
}
```

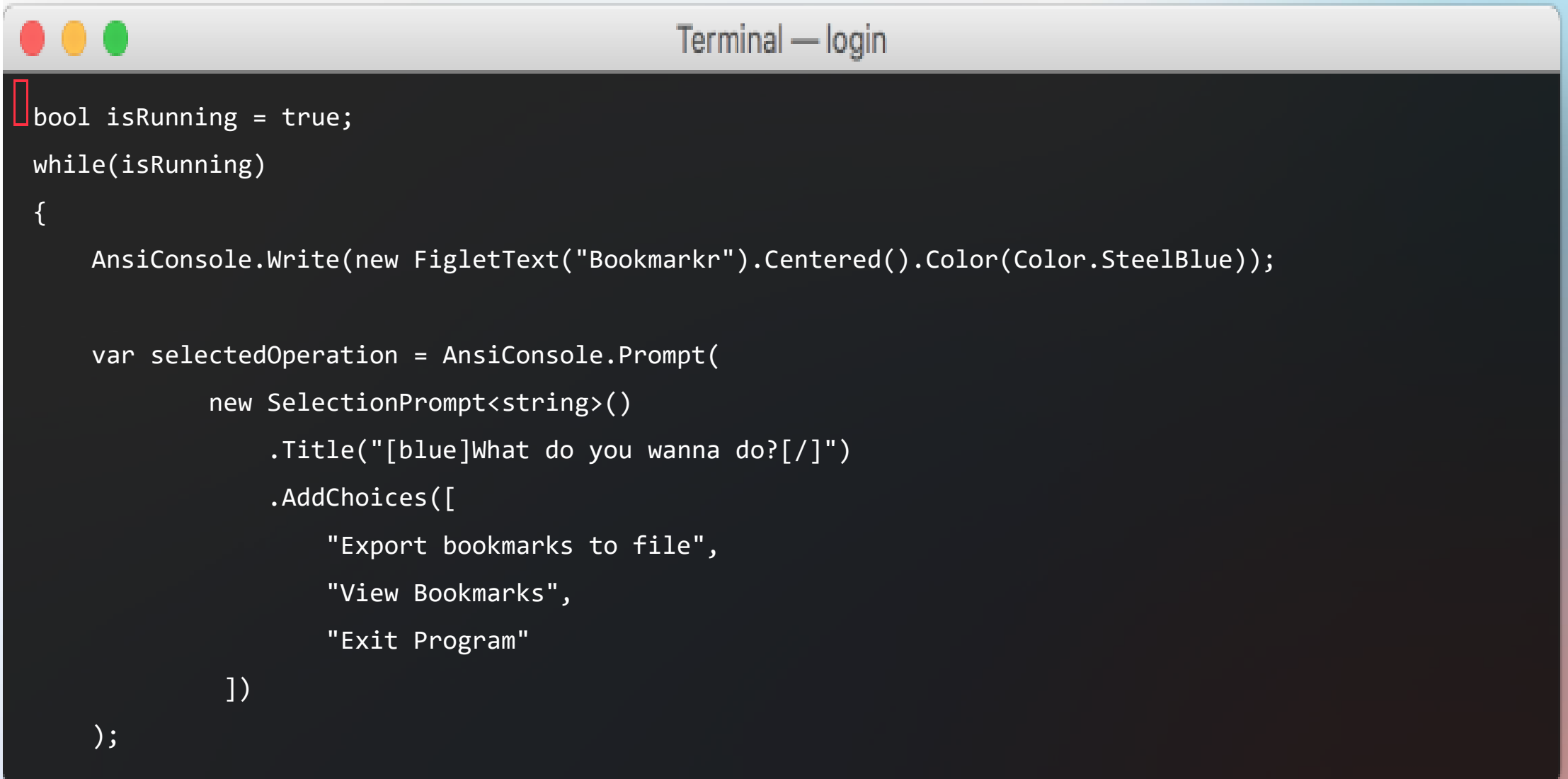

Ajoutons des emojis à notre CLI !

```
212      /***** HANDLER METHODS *****/
213      static void OnHandleRootCommand()
214      {
215          Helper.ShowSuccessMessage(["This is an example of a success message.",
216          "Well done! You have done great. This message is only to let you know that everything went according to the plan."]);
217          Console.WriteLine("Hello !!");
218      }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● PS C:\code\Chap7\bookmarkr> dotnet run
♥️ SUCCESS ♥️
This is an example of a success message.
Well done! You have done great. This message is only to let you know that everything went according to the plan.
Hello !!
Performing shutdown tasks...
○ PS C:\code\Chap7\bookmarkr> █
```

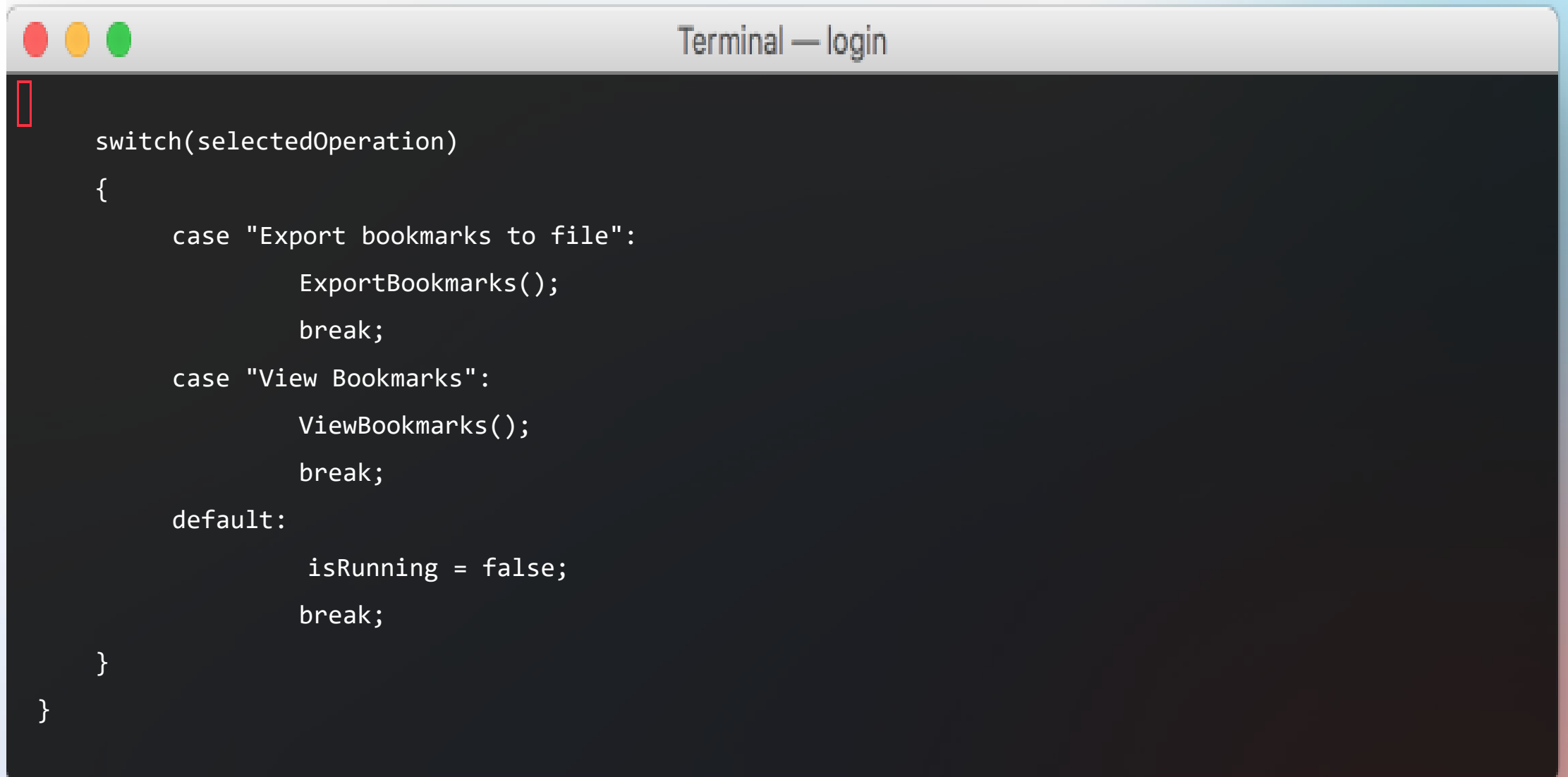
Ajoutons une liste de choix à notre CLI !

A screenshot of a macOS Terminal window titled "Terminal — login". The window has a dark background and a light gray title bar with three colored window control buttons (red, yellow, green) on the left. The code is written in C# and is part of a loop that runs while a boolean variable 'isRunning' is true. Inside the loop, it first writes a blue-colored "Bookmarkr" header to the console. Then, it prompts the user for a selection using the 'AnsiConsole.Prompt' method. The prompt is titled "[blue]What do you wanna do?[/]" and offers three choices: "Export bookmarks to file", "View Bookmarks", and "Exit Program". The code is currently at the end of the prompt method call, with a closing parenthesis and semicolon to be added.

```
bool isRunning = true;
while(isRunning)
{
    AnsiConsole.Write(new FigletText("Bookmarkr").Centered().Color(Color.SteelBlue));

    var selectedOperation = AnsiConsole.Prompt(
        new SelectionPrompt<string>()
            .Title("[blue]What do you wanna do?[/]")
            .AddChoices([
                "Export bookmarks to file",
                "View Bookmarks",
                "Exit Program"
            ])
    );
};
```

Ajoutons une liste de choix à notre CLI !

A terminal window titled "Terminal — login" with a dark background. The window contains a switch statement in code. A red cursor is positioned at the start of the first line of the switch statement. The code is as follows:

```
switch(selectedOperation)
{
    case "Export bookmarks to file":
        ExportBookmarks();
        break;
    case "View Bookmarks":
        ViewBookmarks();
        break;
    default:
        isRunning = false;
        break;
}
```

Ajoutons une liste de choix à notre CLI !



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
○ PS C:\code\Chap7\bookmarkr> dotnet run -- interactive

Bookmarkr

What do you wanna do?
> Export bookmarks to file
View Bookmarks
Exit Program
```

Alors ...

To CLI or not to CLI ?

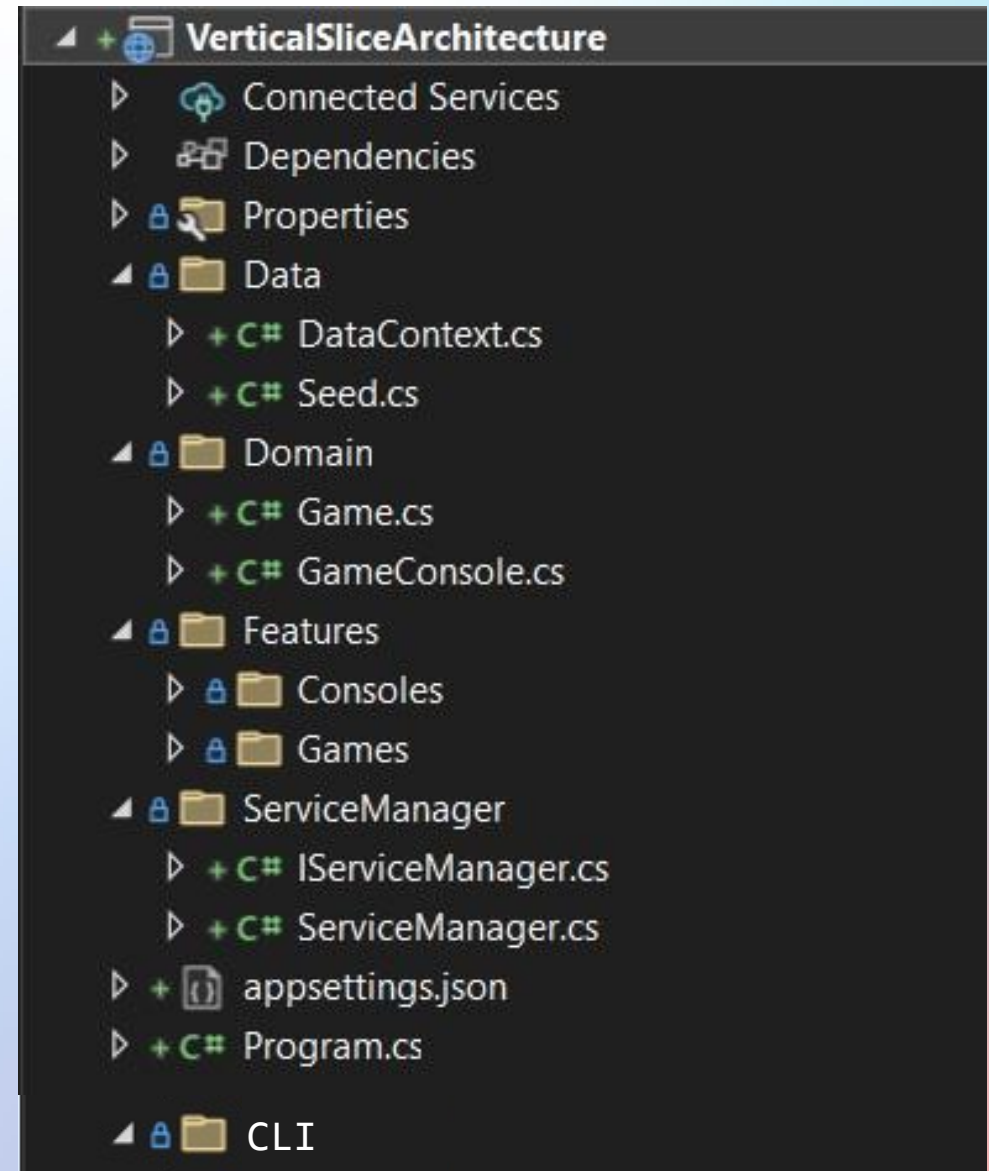


To CLI or not to CLI?

Vous n'avez pas à choisir !

- ❖ Dans certains contextes, une application CLI fait du sens
- ❖ Dans d'autres, pas du tout !

Offrez-en donc la possibilité à vos utilisateurs



Et bien plus...

- ❖ Working with external APIs & Services
- ❖ Error handling & Logging
- ❖ Testing CLI Applications
- ❖ Packaging & Deployment
- ❖ Performance optimization & Tuning
- ❖ Security Considerations
- ❖ ...



<https://packt.link/E7ySG>



Tidjani Belmansour

Merci !

Passez nous voir au kiosque de Cofomo

Mon LinkedIn



ConFoo.CA
DEVELOPER CONFERENCE

Appréciation

