Table of content

Introduction

In this section, you will find internal and external example projects that use the ConsoleAppVisuals library. Feel free to explore them as you like and use them as a reference for your own projects.

Each one will guide you through the setup and execution of the project, as well as provide a brief explanation of the purpose of the project.

Contributing

If you use the ConsoleAppVisuals library in your project and would like to share it with the community, feel free to reach us at morgan@kodelab.fr.

Before you do, make sure to do the following:

- Read the contribution guidelines
- Use the latest or recent version of the library
- Ensure that your code is clean and understandable (documentation is not mandatory but appreciated)
- Capture a GIF or Snapshot demo of your project
- Add a license to your project
- [OPTIONAL]: Follow the template provided here.

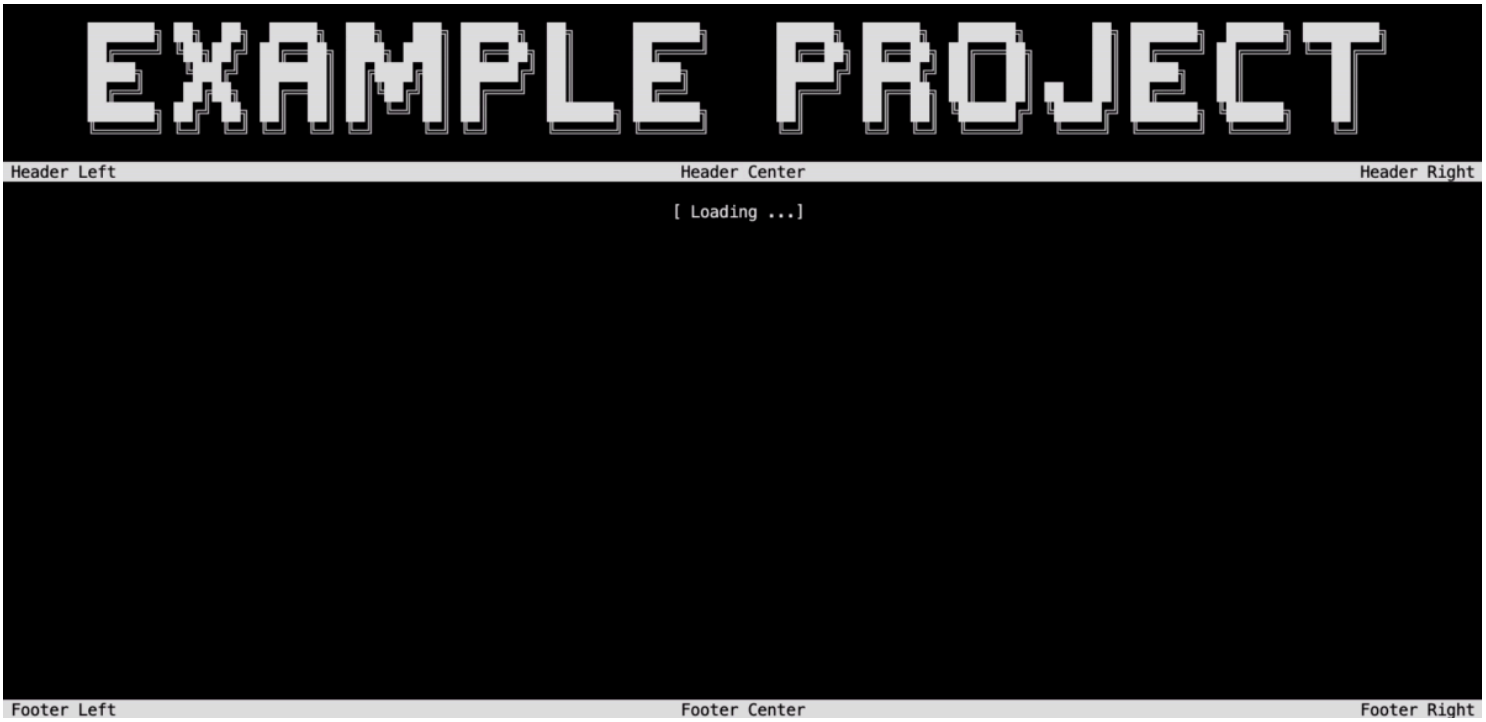
Have a question, give a feedback or found a bug? Feel free to open an issue or start a discussion on the GitHub repository.

Presentation

Author	Size	Library version	Source files
MorganKryze	medium	latest	link

Introduction

The Presentation project serves the purpose to demonstrate the capabilities of the library gathering the major features in a single application.



Features covered

The project covers the following features:

- General: Change `Title` font, update elements, menus management
- PassiveElements: `Title`, `Header` & `Footer`, `Matrix`, `TableView`
- InteractiveElements: `ScrollingMenu`, `Dialog`, `Prompt`, `IntSelector`, `TableSelector`
- AnimatedElements: `FakeLoadingBar`, `LoadingBar`
- Inspector PassiveElement: `ElementDashboard`, `ElementsList`

Build & Run

Install

To clone the project, run the following command:

```
git clone https://github.com/MorganKryze/ConsoleAppVisuals.git
```

Or alternatively, download the project as a zip file from the repository.

Setup

Versions

The project is built with the latest version of the library.

The project is built with the `net9.0` and `net8.0` target framework. Some changes should be applied to the project to make it compatible with the `net6.0` or `net7.0` target frameworks.

Check your local .NET SDK version by running the following command:

```
dotnet --version
```

Build

To build the project, open a terminal in the project directory (`./ConsoleAppVisuals`).

Then move to the working directory:

```
cd examples/Presentation
```

Finally, run the following command:

```
dotnet build
```

Run

If the build is successful, run the following command:

```
dotnet run
```

The application should start and display the same result as in the demonstration video.

Have a question, give a feedback or found a bug? Feel free to open an issue or start a discussion on the GitHub repository.

Password Manager

Author	Size	Library version	Source files
MorganKryze	medium	latest	link

Introduction

The PasswordManager example is a simple console application that allows the user to manage a list of passwords (non-secure). The user can add, remove, and update passwords. The application uses the `Prompt` element to hide the password characters when the user enters them.

Features covered

The project covers the following features:

- General: use of `Prompt` properties (secrecy), update of password list, menus management
- PassiveElements: `Title`, `Header` & `Footer`, `TableView`
- InteractiveElements: `ScrollingMenu`, `Dialog`, `Prompt`, `TableSelector`
- AnimatedElements: `FakeLoadingBar`

Build & Run

Install

To clone the project, run the following command:

```
git clone https://github.com/MorganKryze/ConsoleAppVisuals.git
```

Or alternatively, download the project as a zip file from the repository.

Setup

Versions

The project is built with the latest version of the library.

The project is built with the `net9.0` and `net8.0` target framework. Some changes should be applied to the project to make it compatible with the `net6.0` or `net7.0` target frameworks.

Check your local .NET SDK version by running the following command:

```
dotnet --version
```

Build

To build the project, open a terminal in the project directory (`./ConsoleAppVisuals`).

Then move to the working directory:

```
cd examples/PasswordManager
```

Finally, run the following command:

```
dotnet build
```

Run

If the build is successful, run the following command:

```
dotnet run
```

The application should start and display the same result as in the demonstration video.

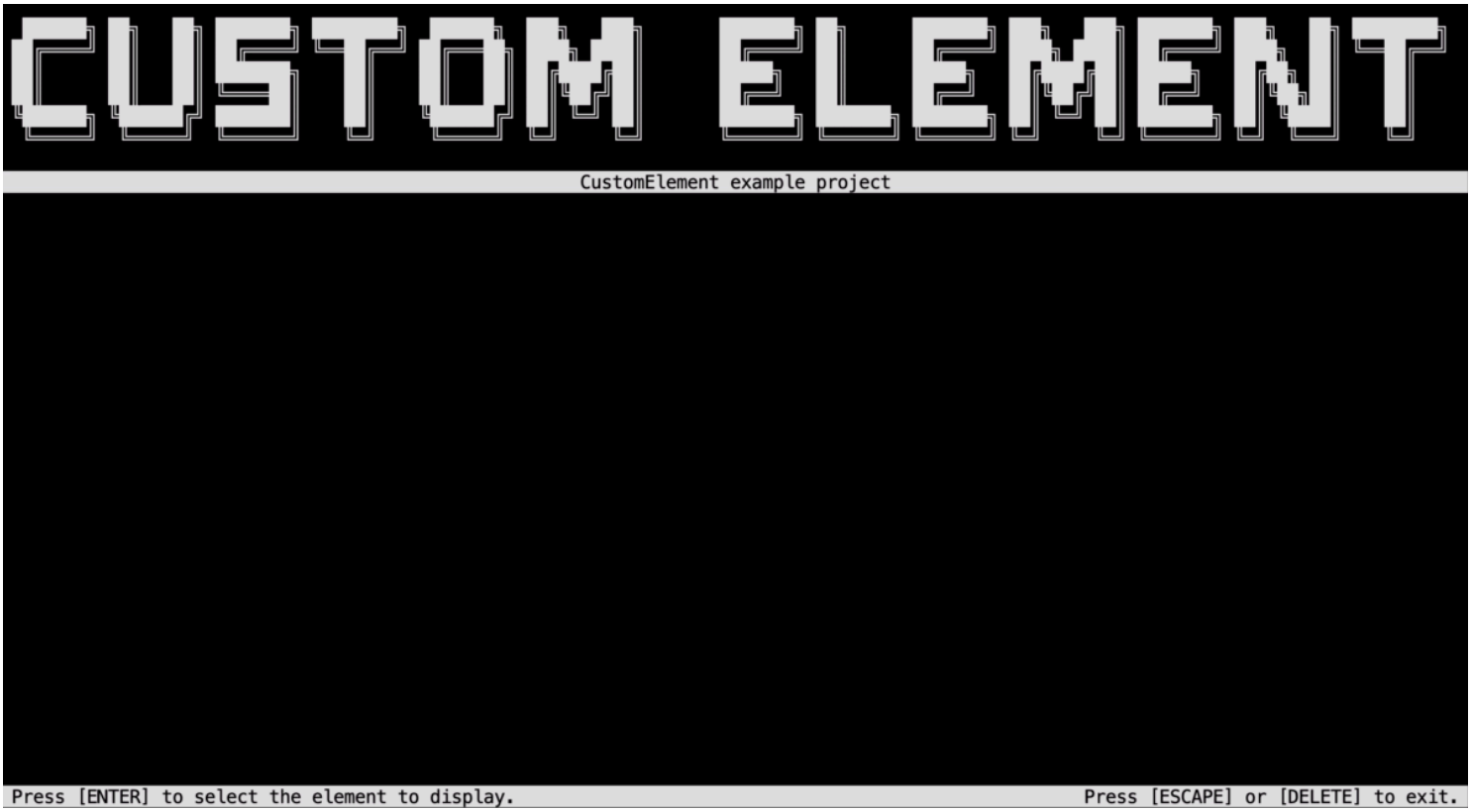
Have a question, give a feedback or found a bug? Feel free to open an issue or start a discussion on the GitHub repository.

Custom Element

Author	Size	Library version	Source files
MorganKryze	small	latest	link

Introduction

The CustomElement example demonstrates the use of the library to create a custom element (Passive or Interactive) that can be used in a Console application.



Features covered

The project covers the following features:

- General: Create and use PassiveElement and InteractiveElement
- PassiveElements: `Title`, `Header` & `Footer`
- InteractiveElements: `ScrollingMenu`

Build & Run

Install

To clone the project, run the following command:

```
git clone https://github.com/MorganKryze/ConsoleAppVisuals.git
```

Or alternatively, download the project as a zip file from the repository.

Setup

Versions

The project is built with the latest version of the library.

The project is built with the `net9.0` and `net8.0` target framework. Some changes should be applied to the project to make it compatible with the `net6.0` or `net7.0` target frameworks.

Check your local .NET SDK version by running the following command:

```
dotnet --version
```

Build

To build the project, open a terminal in the project directory (`./ConsoleAppVisuals`).

Then move to the working directory:

```
cd examples/CustomElement
```

Finally, run the following command:

```
dotnet build
```

Run

If the build is successful, run the following command:

```
dotnet run
```

The application should start and display the same result as in the demonstration video.

Have a question, give a feedback or found a bug? Feel free to open an issue or start a discussion on the GitHub repository.

Custom Font

Author	Size	Library version	Source files
MorganKryze	small	latest	link

Introduction

The CustomFont example demonstrates the use of the library to create a custom font that can be used in a Console application.

Features covered

The project covers the following features:

- General: Create and use custom fonts
- PassiveElements: `Title`, `Header` & `Footer`
- InteractiveElements: `ScrollingMenu`, `Dialog`

Build & Run

Install

To clone the project, run the following command:

```
git clone https://github.com/MorganKryze/ConsoleAppVisuals.git
```

Or alternatively, download the project as a zip file from the repository.

Setup

Versions

The project is built with the latest version of the library.

The project is built with the `net9.0` and `net8.0` target framework. Some changes should be applied to the project to make it compatible with the `net6.0` or `net7.0` target frameworks.

Check your local .NET SDK version by running the following command:

```
dotnet --version
```

Build

To build the project, open a terminal in the project directory (`./ConsoleAppVisuals`).

Then move to the working directory:

```
cd examples/CustomFont
```

Finally, run the following command:

```
dotnet build
```

Run

If the build is successful, run the following command:

```
dotnet run
```

The application should start and display the same result as in the demonstration video.

Have a question, give a feedback or found a bug? Feel free to open an issue or start a discussion on the GitHub repository.

BankingExplorer

Author	Size	Library version	Source files
MorganKryze	medium	3.3.0	link

Introduction

The BankingExplorer project is a simple console application that demonstrates the use of the library to create a banking assistant. You may store expenses and incomes, archive your data and view different accounts.

Features covered

The project covers the following features:

- General: Create and use PassiveElement and InteractiveElement
- PassiveElements: `Title`, `Header` & `Footer`, `EmbedText` (formerly Interactive)
- InteractiveElements: `ScrollingMenu`, `Prompt`, `TableSelector`
- AnimatedElements: `FakeLoadingBar`

Build & Run

Install

To clone the project, run the following command:

```
git clone https://github.com/MorganKryze/BankingExplorer.git
```

Or alternatively, download the project as a zip file from the repository.

Setup

Versions

The project is built with the latest version of the library.

The project is built with the `net9.0` and `net8.0` target framework. Some changes should be applied to the project to make it compatible with the `net6.0` or `net7.0` target frameworks.

Check your local .NET SDK version by running the following command:

```
dotnet --version
```

Build

To build the project, open a terminal in the project directory (`./BankingExplorer`).

Then move to the working directory:

```
cd src/BankingExplorer
```

Finally, run the following command:

```
dotnet build
```

Run

If the build is successful, run the following command:

```
dotnet run
```

The application should start and display the same result as in the demonstration video.

License

```
MIT License

Copyright (c) 2023 MorganKryze
```

Have a question, give a feedback or found a bug? Feel free to open an issue or start a discussion on the GitHub repository.

Boggle : a word game

Author	Size	Library version	Source files
Eliott A. Roussille	big	v3.5.4	Boggle

Introduction

Boggle is a word game where players try to find as many words as possible from a grid of letters.

Features covered

The project covers the following features:

- PassiveElements: `Text`, `EmbedText`, `Title`, `Header` & `Footer`, `Matrix`
- InteractiveElements: `Prompt`, `IntSelector`
- AnimatedElements: `FakeLoadingBar`

Build & Run

Install

To clone the project, run the following command:

```
git clone https://github.com/aust-1/Boggle.git
```

Or alternatively, download the project as a zip file from the repository.

Setup

Versions

The project is built with the latest version of the library.

The project is built with the `net9.0` and `net8.0` target framework. Some changes should be applied to the project to make it compatible with the `net6.0` or `net7.0` target frameworks.

Check your local .NET SDK version by running the following command:

```
dotnet --version
```

Build

To build the project, open a terminal in the project directory (`./src/Boggle`).

Then move to the working directory:

```
cd src/Boggle
```

Finally, run the following command:

```
dotnet build
```

Run

If the build is successful, run the following command:

```
dotnet run
```

The application should start and display the same result as in the demonstration video.

License

```
MIT License

Copyright (c) 2025 Austin
```

Have a question, give a feedback or found a bug? Feel free to open an issue or start a discussion on the GitHub repository.