# Getting Started Clone the repo

Clone the repo to the local machine.

```
git clone https://github.com/benpollarduk/adventure-framework.git
```

## Hello World

```
// create the player. this is the character the user plays as
var player = new PlayableCharacter("Dave", "A young boy on a quest to find the
meaning of life.");
/// create region maker. the region maker simplifies creating in game regions. a
region contains a series of rooms
var regionMaker = new RegionMaker("Mountain", "An imposing volcano just East
of town.")
    // add a room to the region at position x 0, y 0, z 0
    [0, 0, 0] = new Room("Cavern", "A dark cavern set in to the base of
the mountain.")
};
// create overworld maker. the overworld maker simplifies creating in game
overworlds, an overworld contains a series or regions
var overworldMaker = new OverworldMaker("Daves World", "An ancient
kingdom.", regionMaker);
// create callback for generating games
var gameCreator = Game Create("The Life Of Dave",
    "Dave awakes to find himself in a cavern...",
    "A very low budget adventure.",
    x => overworldMaker.Make(),
    () => player,
    x => CompletionCheckResult.NotComplete);
// begin the execution of the game
Game.Execute(gameCreator);
```

# Example game

The quickest way to start getting to grips with the structure of BP.AdventureFramework is by taking a look at the examples. An example game is provided in the

<u>BP.AdventureFramework.Examples</u> directory and have been designed with the aim of showcasing the various features.

# Running the examples

The example applications can be used to execute the example BP.AdventureFramework game and demonstrate the core principals of the framework. Set the **BP.AdventureFramweork.Examples** project as the start up project and build and run to start the application.

# Namespace BP.AdventureFramework. Assets

### Classes

#### ConditionalDescription

Represents a conditional description of an object.

#### **Description**

Represents a description of an object.

#### **ExaminableObject**

Represents an object that can be examined.

#### **ExaminationResult**

Represents the result of an examination.

#### <u>Identifier</u>

Provides a class that can be used as an identifier.

#### <u>Item</u>

Represents an item that can be used within the game.

### **Structs**

#### **Size**

Represents a size.

## **Interfaces**

#### **IExaminable**

Represents any object that is examinable.

#### **IPlayerVisible**

Represents any object that is visible to a player.

# **Delegates**

#### ExaminationCallback

Represents the callback for examinations.