C# Adventure Game Tutorial

Part 1: Designing the game

Traditional text adventure games run in a console, and require the user to type in text to progress in the game:

```
Mr. Sprinkle's Great Escape!
            o_.
         .-'   '.
       .'      _-'-''--o
     J    ,'``  _       >.
   .-',___,)____)___),-'
```

```
Introduction:
You close your eyes, readying yourself for a 2 hour slumber.
'Tomorrow is the day!' You think to yourself.
'I, Mr. Sprinkles the 3rd, will make my escape!'
And with a wiggle of your tail, you fall asleep.

*NOTE*: At any time, type 'help' or 'hint' if you need aid.

Would you like to begin (Y/y)?
```

(Screenshot from http://www.lincolnli.com/text-adventure-game---c.html)

Modern adventure games have 3D graphics and need high-end processors to run, taking up many GB of disk space.

This tutorial is a compromise between the two, using a Windows form, buttons, listboxes and textboxes. It is also possible to include some simple 2D graphics to enhance it.

There are a number of aspects of an adventure game that need to be taken into consideration:

1. The player: Name, strength, health, inventory etc.
2. The world: locations, description, entrances etc.
3. Items in the world, the player's interaction with them and player's inventory.
4. Enemies and Weapons if required

The game will get increasingly difficult to manage as it grows in size and complexity.

The best way to manage it is to put code in different areas.

Take for example the player's inventory:

- The inventory is a list of items carried by the player
- Each item has properties, e.g. name, description, weight, size etc.

We already have variables that can store simple information such as age in years (int), height in metres (float) and name (string).

What we really need is a variable that can hold **more** information, so we could have an 'Item' variable that had name, description, weight, size included in it or a 'player' variable that contained name, strength, health etc.

Luckily we already have one – the 'class'

The class goes even further:

In a 'Player' class, you could even store a list of 'Item' s .

This would be the player's inventory!

This tutorial makes extensive use of classes.

Step 1- Design the form:

Start Visual Studio 2019

1) Create a new project.
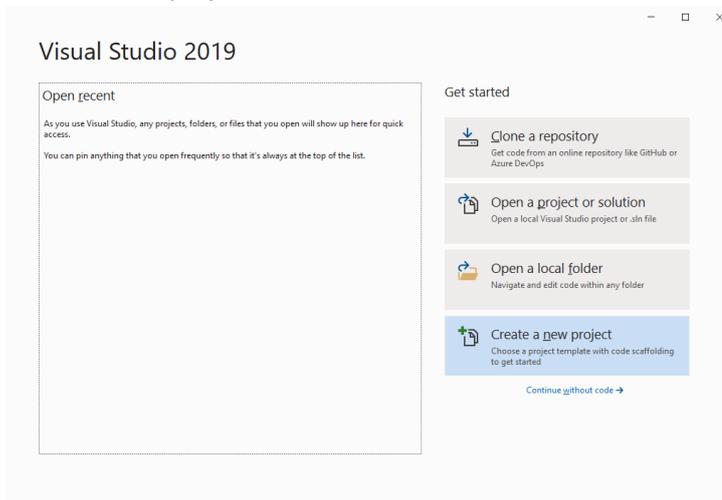
2) Choose C#; Windows; Desktop as indicated. Click on Windows Forms App (.Net Framework)
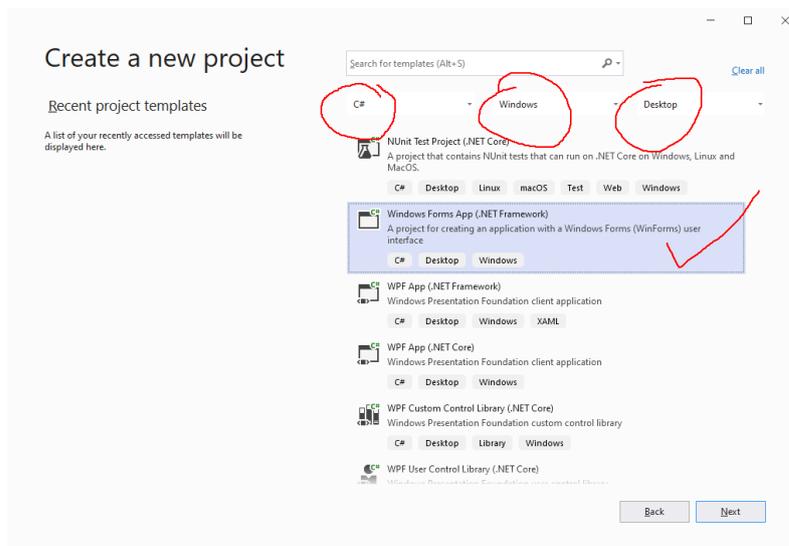
3) Click 'Next' Check the folder it will be created in and browse to your preferred location
4) Name it 'Text Adventure Tutorial'. (I have added Template as this is just for an empty GUI)
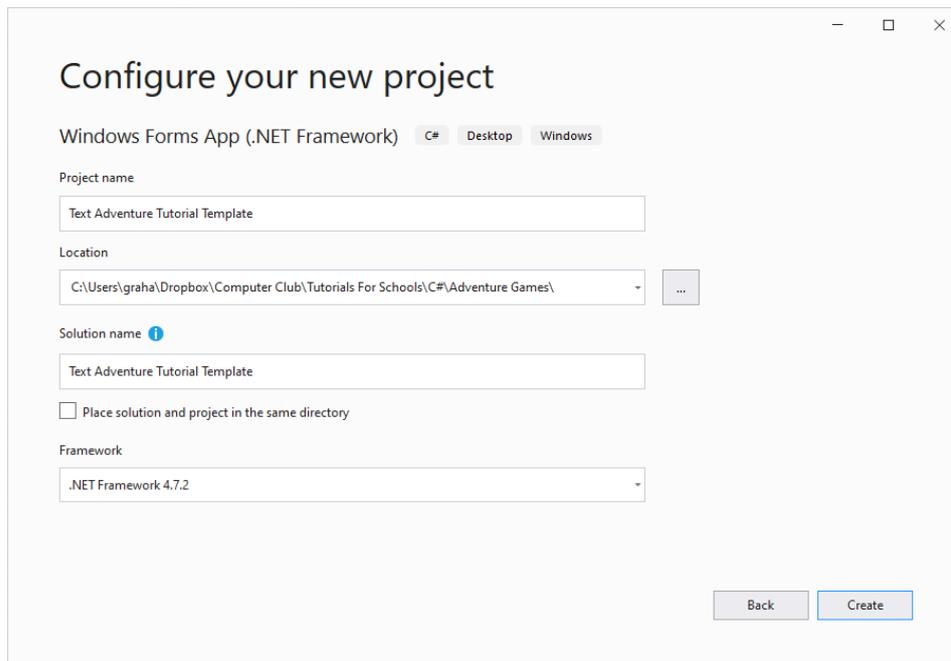
5) Click 'Create'
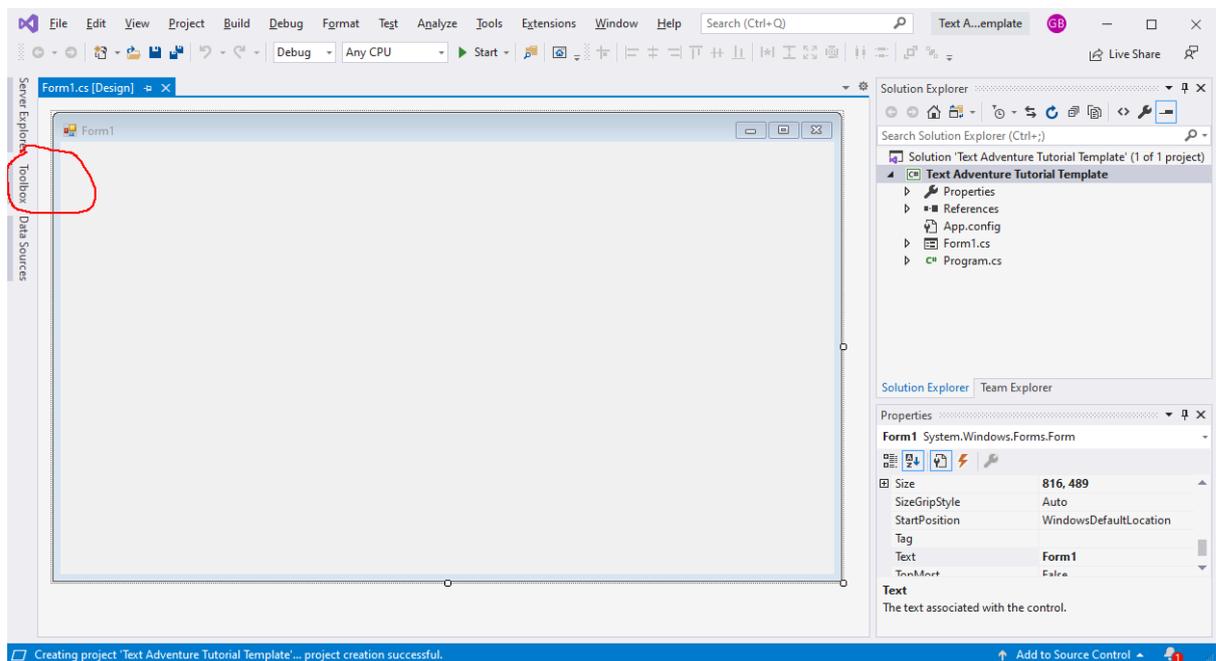
1) On the far left side there are 3 vertical tabs, with one of them labelled "Toolbox" Click on it to show the Toolbox

2) Click on the 'pin' image to lock the Toolbox in place

Drag the window out to increase it's size. If you look in the Properties pane on the right for the 'Size' property, it will change to reflect the new size.

Many school monitors are small, low resolution and 4:3 aspect ratio, so the size of the window you can drag out could be quite low.

Aim for 800 x 600, and type these numbers directly into the Size property if necessary. This will create scrollbars to allow you to work on different parts of the window.
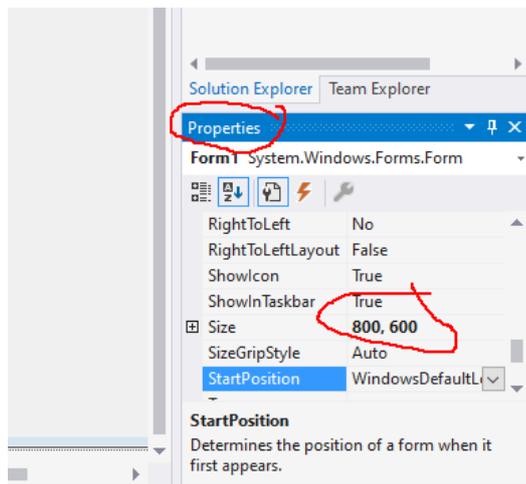
Traditional Text Adventure games require the user to type text in such as 'go north' or 'take torch', after giving a description of the current location.

For game movement , four buttons labelled N, E, S, W will be used, and each button will be enabled or disabled depending on the exits available in each location. This overcomes the text-based:

'go north'

'You cannot go that way'

This game will still have a text description, but items available will be displayed in a listbox, so the user can select an item and examine or take it.

Game Window Design
Game output label:

Click on the label button in the Toolbox pane and draw a label in the top left corner.

Make it about half the width and height of the window. It will instantly drop to a small default size.

In the Properties window, select Autosize and set it to False :
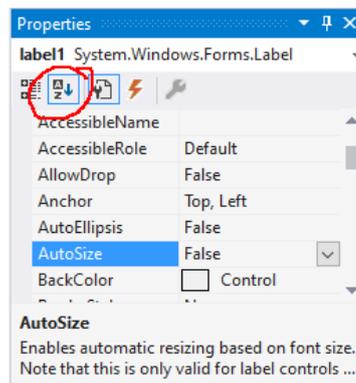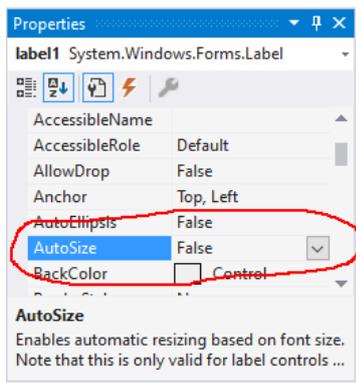
Go to the Properties pane, and click on the AZ| button

Slide the scrollbar to the top so you can see (Name) **label1.**

Change the name to **lblGame**

Change BorderStyle to **FixedSingle**

Change Font to **Consolas Regular 12**

Change BackColor to **Black**

Change ForeColor to **White**

Change Size to **360,340**

Change Text to **(Empty)**
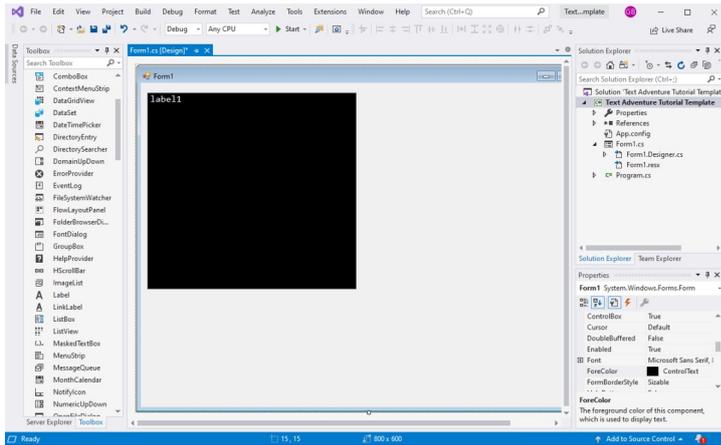
Location ListBox

To display a list of available items in each location use a ListBox control.

From the Toolbox click on ListBox and place it under the Label 'lblGame'. Leave enough space under it to place a couple of buttons.

Change the following ListBox properties:

Name	lstLocation
SelectionMode	One
Size	360, 132

Add a label above the ListBox:

Name	lblLocationItems
Text	Items in
TextAlign	MiddleCenter
AutoSize	False
BorderStyle	FixedSingle

Add two buttons underneath the ListBox

Name	Text
btnExamine	Examine
btnTake	Take

Completed Left half of the window:
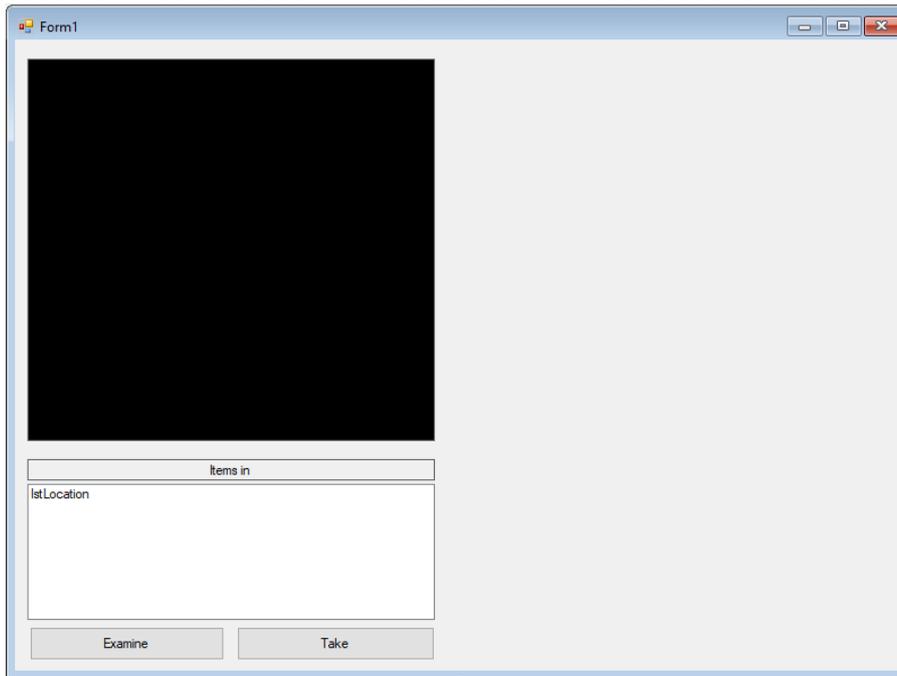
Player's Inventory ListBox

Use another ListBox to display the player's inventory. It can be positioned anywhere, but to keep it neat for coders with OCD, place it next to the Location ListBox and make it the same size. Copy the lstLocation listbox and rename may be easier

Set the properties:

Name	lstInventory
SelectionMode	One
Size	360, 132

Add a label above it with the Text property set to "Inventory:" Copy the lblLocationItems label and rename if you want.

Inventory Buttons

Add two buttons under the inventory ListBox:

Name	Text
btnHold	Hold selected item
btnExamineInventory	Examine

Form1

Items in

lstLocation

Examine | Take

Inventory

lstInventory

Hold selectedd item | Examine
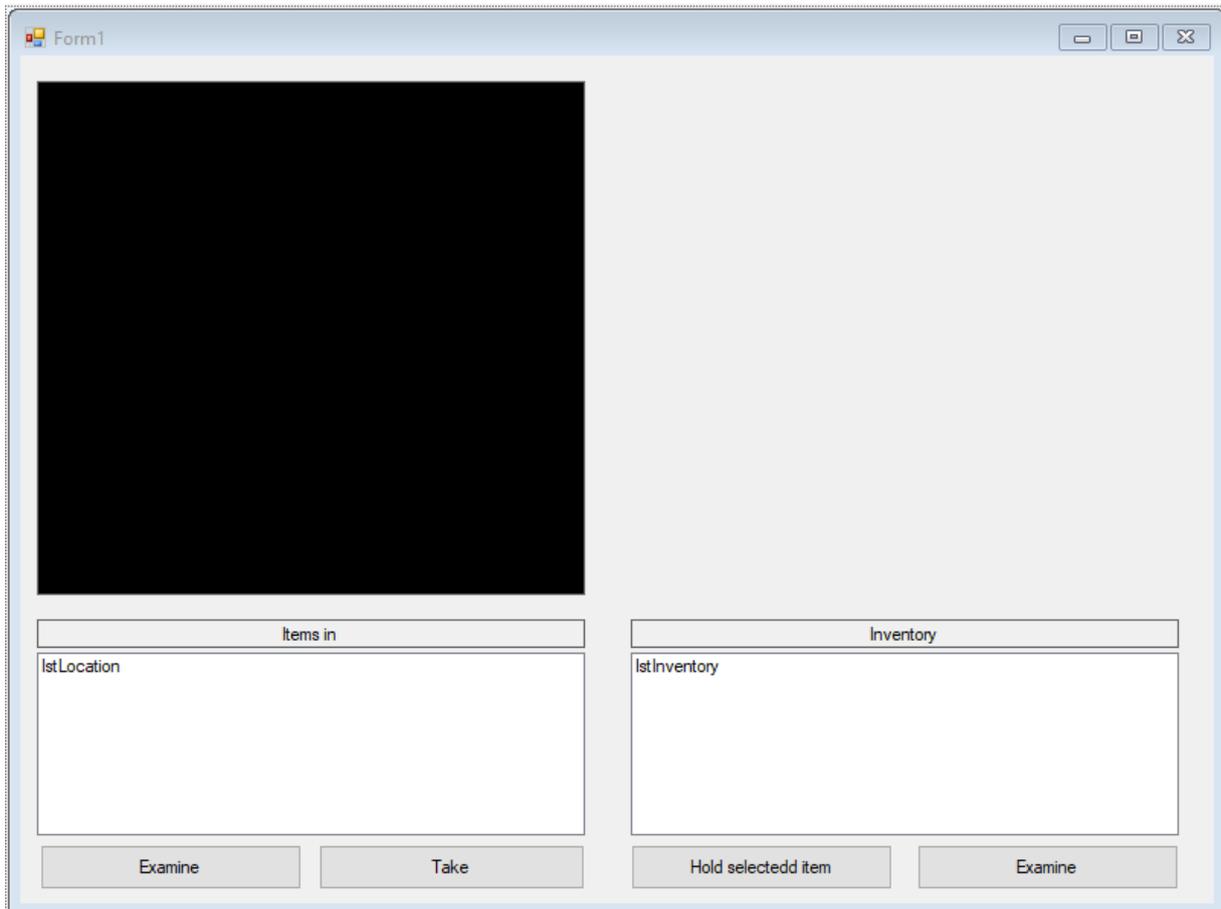
Movement Buttons

Draw a square button in the remaining space

| Name	btnNorth
Text	N
Font	<your choice> Size 36
Size	100,100

Select the button and Copy it (Ctrl-C , or Right-Click → Copy)

Paste 3 copies

Move them into a pattern to suit your playing preferences

Now change the Name and Text of each copy to these values:

btnEast	E
btnSouth	S
btnWest	W

The font will already match the original button
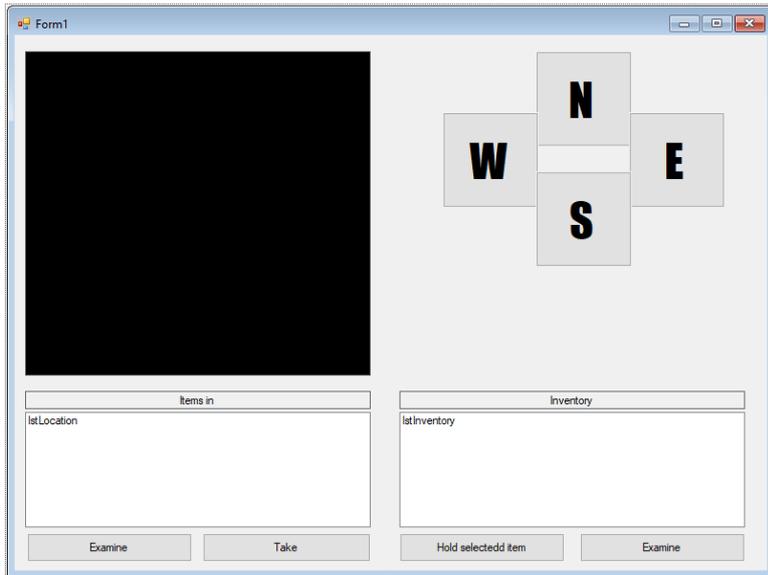
The final area to be designed represents the Player's hand, with a label showing what is being held, and buttons to control dropping the item to the ground, returning it to the inventory, and of course using it.

Add a label:

Name	lblHand
BorderStyle	FixedSingle
Text	<none>
TextAlign	MiddleCenter

Add three more buttons with the following Name/Text:

btnDrop	Drop
btnReturn	Return to inventory
btnUse	Use

That's it! Design complete:
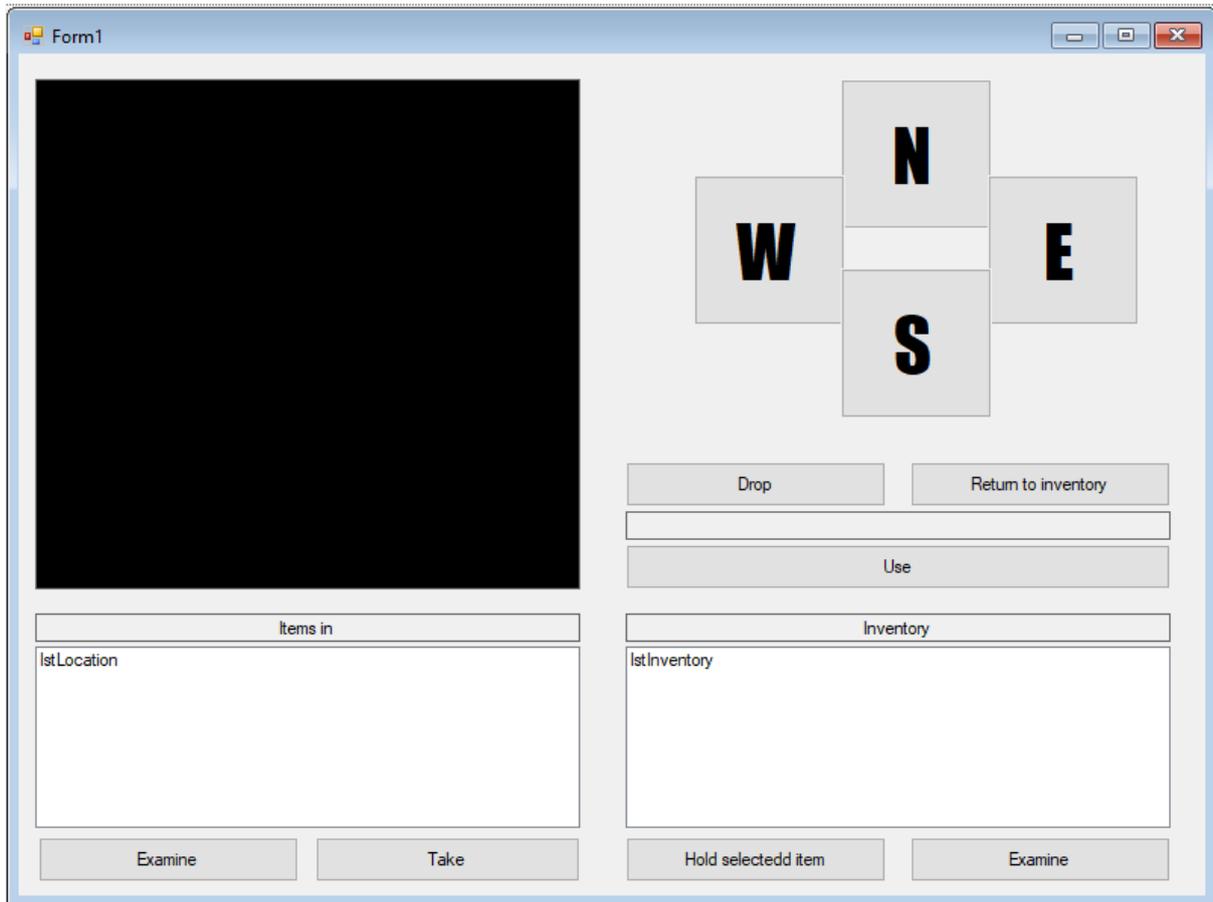
You can change the Form1 Text property to "Text Adventure Game" or similar so it looks a bit better, but this will be changed in code in due course.

Now it's time to get coding…