You want to create a game, or just want to learn how to program in C#, sometimes starting simply is the best way to go, so here's a step by step guide to creating a card game.

This set of tutorials will walk you through creating a simple card game. The example is blackjack but the same principles apply to most card games so you should be able to create almost any card game once you've mastered the basics.

For an example you can try out the simple poker game I created using the same principles. Starting with the relatively simple rules of blackjack, you'll learn the most common C# programming practices and techniques. Then, if you want, you can improve the game, adding more features perhaps network play for multiplayer or card games with more complex rules.

What will I learn?

The focus of these lessons is to show you the most common things you'll need to do to create a C# program – such as creating classes, handling user inputs, and doing common calculations.

What do I need to have to create this?

The examples will be created using visual studio fortunately Microsoft has made this available for free at: https://www.visualstudio.com/

Download the community version and it will be all you need to create the code for this project.

You're also going to need images of cards, a quick google search should find you some or you can create them from scratch. I used Photoshop to create mine. A great free alternative is the gimp: http://www.gimp.org/ or you could just make some simple cards in any graphics program even paint.

tutorial_intro_html_m6c9c7c61 Card drawn in MS paint

save your cards as jpegs 200x270pixels and name them as follows 01_clubs.jpg, 02_clubs.jpg… etc

Once you have your cards and have downloaded and installed visual studio it's time to create your project.

Choose file ->new -> project

For this project we'll use Visual C# and windows Forms

Give your project a name (I just called mine Blackjack) and click ok.

Now you should be viewing your newly created form, currently called Form1

You can edit this in the properties field

I've chosen 1920 by 1080 as I want this project to fit a standard HD screen

And I renamed it CaptnemoBlackJack. In the text field (This name doesn't really matter so pick whatever you prefer)

But left the (Name) field as Form1 this is the field that our code can use to identify the form so we'll leave this as is.

I've also gone ahead and changed the BackColor to ForestGreen which gives us a card table look.

Now we need some place to display our cards, since the most cards we should have on the table in blackjack is 11 (four 2's plus three 3's plus four 1's =21)

We'll need 11 PictureBoxes, drag the picturebox from the toolbox (left hand side) onto the form you can repeat this or copy and paste till you have 11 picture boxes.

Should look something like this. Next We'll need a way to display messages to the user so add a RichTextBox make its size 300,100

We'll also need a few buttons so drag a button from the left hand side resize it and change the text to Deal, then repeat for hit and stick

Ok, we've got a basic card table now it's time to write some code. As I already mentioned this will be C#, previously I had only written code in C and C++ but fortunately C# isn't too different. I'm going to try and follow object oriented programing here and card games lend themselves quite well to this. In Visual Studio each class(object definition) is a separate file. The first thing we'll define is a card. In the solution explorer (should be to the right hand side)

Right click on the project name, choose add->class

Name this **Card.cs** and click ok

You'll see the editor has created your class and added some basic code should look like this

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace blackjack
{
  class Card
  {
  }
}
```

Every card will have the following attributes: suit, value, card number

We also want to define what a 'blank' card will be when a new card is created

So now our class should look like this:

```
 public class Card
  {
    public string suit { set; get; }
    public int value { set; get; }
    public int card_number { set; get; }
    public Card()
    {
      value = -1;
      suit = "";
      card_number = -1;
    }
  }          //END Card class
```

Notice I changed it to a public class so it'll be available to the rest of our project,

I've also added a comment at the end so I can clearly see where that class ends.

Our next object that we'll need is a whole deck of cards. So once more right click on the

Project name in the solution Explorer choose add->class this time name it Deck.cs and click ok.

Here's the code we'll put there, try reading through it, I'll go through and explain it afterward.

Deck.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace blackjack
{
    public class Deck
    {
        public List<Card> cards { set; get; } //deck is list of cards
        // ******************************
        public Deck()   //create a deck
        {
            cards = new List<Card>();
            loaddeck();
        }
        //******************************
        public void loaddeck()
        {
            cards.Clear();
            int cardnum = 1;
            for (int i = 1; i < 5; i++) // 4 suits
            {
                for (int j = 1; j < 14; j++)
// 13 cards each (total 52 cards)
                {
                Card currentcard = new Card();
 //create a new card
                    //assign suit
          currentcard.card_number = cardnum;
                    if (i == 1)
                currentcard.suit = "spades";
                    if (i == 2)
                currentcard.suit = "clubs";
                    if (i == 3)
                currentcard.suit = "hearts";
                    if (i == 4)
               currentcard.suit = "diamonds";
                    // assign value
               currentcard.value = j;
                    // put in deck
               cards.Add(currentcard);
 // adding card to deck
                    //next card
                    cardnum++;
                }
            }
        }
        // ************************
        public Card FindCard(int cardnum)
        {
            foreach (Card a_card in cards)
            {
         if (a_card.card_number == cardnum)
                {
                    return a_card;
                }
            }
            return null;
        }
        // *************
    } // end deck class
}
```

So the first thing you should notice is that a Deck object is going to contain a List which is a type of array

This list will consist of objects of the class we previously made 'Card' and the list will be called cards.

NOTE: C# like C and C++ is case specific so Card is not the same as card, but to make it more readable since the list holds many (52) cards I have named it by the plural.

Next we have a Constructor function that creates a deck, it assigns a new list to cards then calls the function loaddeck. The loaddeck function has a pair of nested loops, one that counts through the four suits and one that goes through the values 1 through 13. Each time the card number is incremented so card number 1 is the ace of spades, card number 14 is the ace of clubs etc.

Finally we have a function that returns a card object from the deck when given a card number.

You'll notice at this point the deck is in order we could go ahead and shuffle it now but instead I'm going to randomly pick cards from the deck as we need them which will have the same effect. So what we need next is a random number, computers aren't good at random. There are a few different ways of creating these, many use the system clock as a 'seed' to give different results. I'm going to use one that uses a cryptography library to create the seed. So once more create a new class, I'll call this one RandomNumber.cs. The code for it should look like this:

RandomNumber.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace blackjack
{
    public class RandomNumber
    {
        private static readonly
System.Security.Cryptography.RNGCryptoServicePro
vider _seed = new
System.Security.Cryptography.RNGCryptoServicePro
vider();
      public static int NumberBetween(int minimum,
int maximum)
      {
         byte[] randomNumber = new byte[1];
         _seed.GetBytes(randomNumber);
         double asciiValue =
Convert.ToDouble(randomNumber[0]);
         double multiplier = Math.Max(0, (asciiValue /
255d) - 0.00000000001d);
         // adding one to the range, to allow for
rounding
         int range = maximum - minimum + 1;
         double randomValue = Math.Floor(multiplier
* range); // rounds to ensure within range
         return (int)(minimum + randomValue); // adds
minvalue to randomvalue(0 to max)
      }
   } // END RandomNumber
}
```

Now we can take that random number and use it to draw cards from our deck. We'll place these cards in a new object I'll call Hand. Yep, you guessed it, create a new class and name this one Hand.cs Just like a deck, a hand will need a list structure to hold our cards, I'm also going to add a number_of_cards which will be the initial amount of cards dealt (2) and we can increment this as more cards are added to the hand. We also have score to store the value of the hand and a string called result, this will give us some output to display to the player based on the current score.

Hand.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace blackjack
{
   public class Hand
   {
      public List<Card> cards { set; get; }
      public int number_of_cards { set; get; }
      public int score { set; get; }
      public string result { set; get; }
      // *******************************
      public Hand(int numcards) //constructor creates
list and places empty cards in it
      {
         number_of_cards = numcards; //number of
cards in a hand
         cards = new List<Card>();    // list to hold
cards
         Card emptycard = new Card(); // blank card
         for (int i = 0; i < numcards; i++)
         {
            cards.Add(emptycard);      // adding blank
cards to list
         }
      }

//****************************************
      public void add_card(Deck currentdeck, int
number_cards_in_a_hand) // picks random card
from deck and puts it in one of the blank card spots
created
      {
         bool added = false;
         int pickedcard = 0;
         if (currentdeck.cards.Count <= 2)
         {
            currentdeck.loaddeck();
            //clear_hand();
         }
         pickedcard =
RandomNumber.NumberBetween(2,
currentdeck.cards.Count - 1);
         Card currentcard =
currentdeck.cards.ElementAt(pickedcard);
         Card tobereplaced = new Card();

         while (!added)
         {
            foreach (Card temp in cards)
            {
                if (temp.suit == "")
//place to put new card in
                    tobereplaced = temp;
            }
            if (tobereplaced != null)
            {
                cards.Remove(tobereplaced);
                cards.Add(currentcard);
                added = true;
                currentdeck.cards.Remove(currentcard);
            }
        }
    }
//*****************************
    public void deal_cards(Deck currentdeck, int
numcards)
    {
        numcards = cards.Count;
        for (int i = 0; i < numcards; i++)
        {
            add_card(currentdeck, numcards);
        }
    }
  // *************************************
    public void evaluate_hand()
    {
        score=0;
       foreach (Card temp in cards)
        {
            if (temp.value < 10)
                score = score + temp.value;
            else
                score = score + 10; // assigning face cards
a value of 10
        }
        // adjust for aces
        foreach (Card temp in cards)
        {
            if (temp.value==1 && score+10<=21)
            score = score + 10; // We haven't gone bust
yet so Ace is scored at 11
        }
        if (score>21)
        {
            result = "Sorry you bust";
        }
        else
        {
            result = "You have ";
        }
    }
  } //End hand
}
```

This class is a little longer than the previous one as I have included functions to help deal with the hand objects. First though we have the usual constructor function which creates a list of cards and inserts blank cards. Next we have the function that uses the RandomNumber class to generate a random number and use this to choose a card from the deck, which is then placed in the list by the add_card function. The add_card function is called by the deal_card function which manages how many cards are added. Finally we have the evaluate_hand function which calculates the score for a hand, adjusting aces to be 1 or eleven as necessary and setting the result string to either bust or the current score.

Ok, now we have the objects we'll need lets put them into action. From the Solution Explorer choose Form1.cs and We'll be adding the following elements, a current deck the players hand, the dealers hand and the number of cards to deal initially. We'll also setup how the form is going to look at the start of the game. Here's the code, read through it and the comments should explain it.

Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace blackjack
```

```
{
    public partial class Form1 : Form
    {
        public Deck currentdeck { get; set; }
        public Hand player_hand { get; set; }
        public Hand dealer_hand { get; set; }
        public int numcards { get; set; }
        public Form1()
        {
            numcards = 2; //deal 2 cards initially
            currentdeck = new Deck();
 //create new deck of cards
            player_hand = new Hand(numcards); //create
hand for player
            dealer_hand = new Hand(numcards); //
create hand for dealer
            InitializeComponent();
            button1.Visible = true;
 // deal button is visible
            button2.Visible = false;
// hit button hidden
            button3.Visible = false;
 // stick button hidden
            pictureBox1.Visible = true;
 //showing spot for first card
            pictureBox2.Visible = false;
 //hiding the rest
            pictureBox3.Visible = false;
pictureBox4.Visible = false;            pictureBox5.Visible
= false;            pictureBox6.Visible = false;
pictureBox7.Visible = false;            pictureBox8.Visible
= false;            pictureBox9.Visible = false;
pictureBox10.Visible = false;
            pictureBox11.Visible = false;
pictureBox12.Visible = false;
             richTextBox1.Text = "Welcome to CaptNemo
BlackJack" + Environment.NewLine;
            richTextBox1.Text += "Press Deal to begin" +
Environment.NewLine;
        } //END Form1
    } //End partial Class
}
```

All right, now you actually have something to try out. Up at the top of the screen there's a Start button. Click this and your program will be compiled and started. You should have something like this:

So let's make something happen when the player clicks the deal button. Close the running program then select the Form1.cs[Design] tab at the top of the project window

Double click on the deal button and you'll be taken back to the Form1.cs tab but now some code has been added creating a function that will run when the deal button is clicked. Currently this is blank and should look like this:

```
private void button1_Click(object sender, EventArgs
e)
        {
        }
```

Now we can add some code to display the players cards

In between the { } add display_hand(player_hand); This won't do anything yet though as we haven't created the display_hand function, so after our new function add this:

```
 // *** Display current hand ***
        public void display_hand(Hand playerhand)
        {
            int count = 0;
            string currentcard_picture = "";

            if (playerhand.cards != null && playerhand !=
null)
                foreach (Card currentcard in
playerhand.cards)
                {
                    if (currentcard != null && currentcard.suit !=
"" )
                    {
                        if (currentcard.value < 10)
                            currentcard_picture = "_0" +
currentcard.value.ToString() + "_" + currentcard.suit;
                if (currentcard.value == 10)
currentcard_picture = "_" +
currentcard.value.ToString() + "_" + currentcard.suit;
                    if (currentcard.value == 11)
                    currentcard_picture = "J" + "_" +
currentcard.suit;
                   if (currentcard.value == 12)
                currentcard_picture = "Q" + "_" +
currentcard.suit;
                 if (currentcard.value >= 13)
                  currentcard_picture = "K" + "_" +
currentcard.suit;
                    }
                    else
                 currentcard_picture = "space";
                    if (count == 0)
                    {
   System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox1.Image = myImage;
        pictureBox1.SizeMode =
PictureBoxSizeMode.StretchImage;
                    }
                    if (count == 1)
                    {
    System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox2.Image = myImage;
                    }
                    if (count == 2)
                    {
     System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox3.Image = myImage;
                    }
                    if (count == 3)
                    {
    System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox4.Image = myImage;
                    }
                    if (count == 4)
                    {
    System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox5.Image = myImage;
                    }
                    if (count == 5)
                    {
     System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox6.Image = myImage;
                    }
                    if (count == 6)
                    {
     System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox7.Image = myImage;
                    }
                    if (count == 7)
                    {
    System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox8.Image = myImage;
                    }
                    if (count == 8)
                    {
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                        pictureBox5.Image = myImage;
                    }
                    if (count == 9)
                    {
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                        Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox10.Image = myImage;
                }
                if (count == 10)
                {
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox11.Image = myImage;
                }
                if (count == 11)
                {
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox12.Image = myImage;
                }
                count++;
            }
        } // end display hand
```

This still won't do anything though as we have one more important step. Remember those cards you made at the beginning of all this? We need to import those to our project. At the top of the screen click on PROJECT ->blackjack properties. Then in the left hand column select Resources, along the top of this section is a tab Add Resource, click the small black dropdown arrow on its right and choose 'Add existing file..' then navigate to where you saved your card images select them and click open. You'll notice the names displayed are different from the file names, visual studio adds an underscore _ since resource names shouldn't start with a number and doesn't display the file extension. Not to worry though as our display hand function already takes that into account. Now go back and change the button1_Click function to this:

```
  private void button1_Click(object sender, EventArgs
e)
        {
            player_hand.deal_cards(currentdeck,
numcards);
            display_hand(player_hand);
            richTextBox1.Text = player_hand.result +
Environment.NewLine;
        }
```

Go ahead and try running your project again, click the deal button you should get output like this:

You'll notice that even though the deal function adds two cards to the players hand we're only seeing one. This is because we set all the picture boxes after the first one to hidden.

So in the display hand function look for the section following if (count ==1), this is checking for a second card, 0 being the first. Now change it to look like this:

```
if (count == 1)
                {
                    pictureBox2.Visible = true; //showing
second card
   System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox2.Image = myImage;
                    pictureBox2.SizeMode =
PictureBoxSizeMode.StretchImage;
                }
```

The line pictureBox2.visible= true; shows the second card, also note the line pictureBox2.SizeMode = PictureBoxSizeMode.StretchImage; which ensures our card image is stretched to fit the picturebox correctly. This addition should be done to all the card display sections 0 through11 now our display cards function should look like this:

```
// *** Display current hand *******************
        public void display_hand(Hand playerhand)
        {
            int count = 0;
            string currentcard_picture = "";
            DateTime t = DateTime.Now;
            if (playerhand.cards != null && playerhand !=
null)
                foreach (Card currentcard in
playerhand.cards)
```

```
            {
                if (currentcard != null && currentcard.suit
!= "" )
                {
                    if (currentcard.value < 10)
            currentcard_picture = "_0" +
currentcard.value.ToString() + "_" + currentcard.suit;
                    if (currentcard.value == 10)
            currentcard_picture = "_" +
currentcard.value.ToString() + "_" + currentcard.suit;
                    if (currentcard.value == 11)
                        currentcard_picture = "J" + "_" +
currentcard.suit;
                    if (currentcard.value == 12)
                        currentcard_picture = "Q" + "_" +
currentcard.suit;
                    if (currentcard.value >= 13)
                        currentcard_picture = "K" + "_" +
currentcard.suit;
                }
                else
                    currentcard_picture = "space";
                if (count == 0)
                {
                    pictureBox1.Visible = true; //showing
first card
    System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox1.Image = myImage;
                    pictureBox1.SizeMode =
PictureBoxSizeMode.StretchImage;
                }
                if (count == 1)
                {
                    pictureBox2.Visible = true; //showing
second card
    System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox2.Image = myImage;
                    pictureBox2.SizeMode =
PictureBoxSizeMode.StretchImage;
                }
                if (count == 2)
                {
                    pictureBox3.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox3.Image = myImage;
                    pictureBox3.SizeMode =
PictureBoxSizeMode.StretchImage;
                }
                if (count == 3)
                {
                    pictureBox4.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox4.Image = myImage;
                    pictureBox4.SizeMode =
PictureBoxSizeMode.StretchImage;
                }
                if (count == 4)
                {
                    pictureBox5.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox5.Image = myImage;
                    pictureBox5.SizeMode =
PictureBoxSizeMode.StretchImage;
                }
                if (count == 5)
                {
                    pictureBox6.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                    Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                    pictureBox6.Image = myImage;
                    pictureBox6.SizeMode =
PictureBoxSizeMode.StretchImage;
                }
                if (count == 6)
                {
```

```
                pictureBox7.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                pictureBox7.Image = myImage;
                pictureBox7.SizeMode =
PictureBoxSizeMode.StretchImage;
            }
            if (count == 7)
            {
                pictureBox8.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                pictureBox8.Image = myImage;
                pictureBox8.SizeMode =
PictureBoxSizeMode.StretchImage;
            }
            if (count == 8)
            {
                pictureBox9.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                pictureBox9.Image = myImage;
                pictureBox9.SizeMode =
PictureBoxSizeMode.StretchImage;
            }
            if (count == 9)
            {
                pictureBox10.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                pictureBox10.Image = myImage;
                pictureBox10.SizeMode =
PictureBoxSizeMode.StretchImage;
            }
            if (count == 10)
            {
                pictureBox11.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                pictureBox11.Image = myImage;
                pictureBox11.SizeMode =
PictureBoxSizeMode.StretchImage;
            }
            if (count == 11)
            {
                pictureBox12.Visible = true;
 System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
                pictureBox12.Image = myImage;
                pictureBox12.SizeMode =
PictureBoxSizeMode.StretchImage;
            }
          count++;
        }
    } // end display hand
```

Go ahead and run your project and press deal, you should now see two cards, you'll notice though, that the deal button is still displayed, if you click it you'll continue to receive two cards each time which are displayed until all our picture boxes are filled. Let's go ahead and fix that now. In the button1_Click function (our deal button) we need to add a line to hide the deal button, then we need to make the hit and stay buttons visible, You may have also noticed the result is not showing in the textbox yet, this is because we haven't called the evaluate function yet so we may as well do that now, here's the updated function:

```
private void button1_Click(object sender, EventArgs
e) //Deal button
    {
  player_hand.deal_cards(currentdeck, numcards);
      display_hand(player_hand);
      player_hand.evaluate_hand();
      richTextBox1.Text = player_hand.result +
Environment.NewLine;
      button1.Visible = false;
      button2.Visible = true;
      button3.Visible = true;
      }
```

Now if you run the project you should see two cards when the deal button is clicked and their combined value displayed in the textbox. The hit and stick button still don't do anything though so let's take care of that next. Going back to the Form1.cs[Design] and double click on the hit button. This will create the button2_Click function. Add code so it looks like the following:

```
private void button2_Click(object sender, EventArgs
e) // Hit function
      {
         player_hand.add_card(currentdeck, 1);
            display_hand(player_hand);
            player_hand.evaluate_hand();
            richTextBox1.Text = player_hand.result +
Environment.NewLine;
            button1.Visible = false;
            button2.Visible = true;
            button3.Visible = true;
      }
```

Depending on the order you made your picture boxes the cards are probably showing up beneath the previous dealt ones, we can fix this with pictureBox.BringToFront(); add this to each of the sections in the display cards function and they'll each move to the front as they are dealt.

Here's how it looks for the first card , repeat for the rest:

```
  if (count == 0)
                 {
                pictureBox1.Visible = true;
//showing first card
   System.Resources.ResourceManager rm =
blackjack.Properties.Resources.ResourceManager;
                Bitmap myImage =
(Bitmap)rm.GetObject(currentcard_picture);
          pictureBox1.Image = myImage;
pictureBox1.BringToFront();
pictureBox1.SizeMode =
pictureBoxSizeMode.StretchImage;
                 }
```

Ok, the cards continue to be dealt even after we've bust so lets add a end_game function, this goes in Form1.cs you can add it right after the button2 function.

```
private void end_game(Hand player_hand, Hand
dealer_hand)
        {
            dealer_hand.deal_cards(currentdeck,
numcards); //deal cards to dealer
            dealer_hand.evaluate_hand();
            while(dealer_hand.score<15) //dealer sticks
on 15 or higher
            {
                dealer_hand.add_card(currentdeck, 1);
                dealer_hand.evaluate_hand();
            }
            if (player_hand.score > 21)
            {
                richTextBox1.Text = "You bust better luck
next time." + Environment.NewLine;
            }
            if (dealer_hand.score > 21)
            {
richTextBox1.Text = "Dealer has " +
dealer_hand.score + ", congratulations you win ."
+Environment.NewLine;
            }
            else
            {
                if ((player_hand.score > dealer_hand.score)
&& (player_hand.score <= 21))
                {
richTextBox1.Text = "Dealer has " +
dealer_hand.score + ", congratulations you win ." +
Environment.NewLine;
                }
                if(player_hand.score ==21)
                {
       richTextBox1.Text = "Congratulations
BLACKJACK, you win ." + Environment.NewLine;
                }
            }
            button1.Visible = true;
 // deal button is visible
            button2.Visible = false;
 // hit button hidden
          button3.Visible = false;
// stick button hidden
        }
```

By now most of this should be fairly familiar. Firstly we give the dealer his cards, note he sticks at 15 or higher, this can be changed to match whichever rules you'd prefer. We then compare the dealer hand and player hand to see who won, and display an appropriate message to the player. We'll need to call this function when the stick button is clicked so double click on the stick button to generate a button3_Click function and add code so it looks like this:

```
    private void button3_Click(object sender, 
EventArgs e) // stick button
        {
          end_game(player_hand, dealer_hand);
        }
```

We'll need to call the end_game function from another place as well, in the button2_Click function that is called when the player clicks hit, we'll check if the player goes bust and if so call end_game.

```
if (player_hand.score >= 21)
          {
              end_game(player_hand, dealer_hand);
          }
```

Almost done now, the end_game function displays the result of the current hand and reveals the deal button, so the final step we'll make here is to reset the hands when the deal button is clicked. The new deal function should look like this:

```
  private void button1_Click(object sender, EventArgs 
e) //Deal button
        {
          //initialise components
          numcards = 2; //deal 2 cards initially
          currentdeck = new Deck(); //create new deck 
of cards
          player_hand = new Hand(numcards); //create 
hand for player
          dealer_hand = new Hand(numcards); // 
create hand for dealer
          button1.Visible = true;
 // deal button is visible
          button2.Visible = false;
// hit button hidden
          button3.Visible = false;
// stick button hidden
          pictureBox1.Visible = true;
//showing spot for first card
          pictureBox2.Visible = false;
//showing second card
          pictureBox3.Visible = false;
//hiding the rest
          pictureBox4.Visible = false;
          pictureBox5.Visible = false;
          pictureBox6.Visible = false;
          pictureBox7.Visible = false;
          pictureBox8.Visible = false;
          pictureBox9.Visible = false;
          pictureBox10.Visible = false;
          pictureBox11.Visible = false;
          pictureBox12.Visible = false;
          richTextBox1.Text = "Welcome to CaptNemo 
BlackJack" + Environment.NewLine;
          richTextBox1.Text += "Press Deal to begin" + 
Environment.NewLine;
          player_hand.deal_cards(currentdeck, 
numcards);
          display_hand(player_hand);
          player_hand.evaluate_hand();
         richTextBox1.Text = player_hand.result + 
Environment.NewLine;
          button1.Visible = false;
          button2.Visible = true;
          button3.Visible = true;
        }
```

There you go a working black jack game, some ideas for improvement, add a bet button, award the player a starting amount of money and keep score. Adding multiple player options etc. I'll leave those for you though. Please feel free to leave any questions or comments. If you make a great card game I'd love to hear about it.