**Inventory Management**

Patrick Benjamin, Julie Chiu, Richard Nguyen

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 InventoryManagement

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

//Initial lists.

SortedList<string, decimal> foodList = new SortedList<string, decimal>

{ ["Lunch Meat"] = 7, ["Bread"] = 6, ["Oranges"] = 12, ["Carrots"] = 4, ["Twinkies"] = 40 };

SortedList<string, decimal> weaponList = new SortedList<string, decimal>

{ ["Steak Knife"] = 4, ["Shotgun"] = 2, ["Hand Grenade"] = 20, ["Laser Gun"] = 2 };

SortedList<string, decimal> firstaidList = new SortedList<string, decimal>

{ ["Bandage"] = 10, ["Rubbing Alchohol"] = 4, ["Hand Sanitizer"] = 2, ["Gauze"] = 5 };

//Initializing strings for our master list/show button

string aidString = "";

string foodString = "";

string weaponString = "";

private void radioButton3\_CheckedChanged(object sender, EventArgs e)

{

UpdateList(firstaidList); //Updates list box when selected.

}

private void UpdateList(SortedList<string, decimal> list)

{

listBox1.Items.Clear(); //Clears Listbox

foreach (KeyValuePair<string, decimal> item in list)

{

listBox1.Items.Add(item.Key); //Populates listbox with the Keys of a given list.

}

}

private void button1\_Click(object sender, EventArgs e) //Take button

{

string listItem = listBox1.GetItemText(listBox1.SelectedItem); //Identifies what is selected in the listbox and converts the name to a string.

//Subtracts the count using the subtract method.

subtract(nudAmount.Value, listItem, foodList); //Uses subtract method with the given values.

subtract(nudAmount.Value, listItem, weaponList);

subtract(nudAmount.Value, listItem, firstaidList);

nudAmount.Value = 0; //Resets the NumericUpDown button to zero.

}

private void rdoFood\_CheckedChanged(object sender, EventArgs e)

{

UpdateList(foodList); //Updates list box when selected.

}

private void rdoWeapons\_CheckedChanged(object sender, EventArgs e)

{

UpdateList(weaponList); //Updates list box when selected.

}

private string MasterList(SortedList<string,decimal> list, string name)

{

name = "";

foreach (KeyValuePair<string, decimal> item in list)

{

name += item.Key + "\t" + item.Value + "\n"; //Gets sorted lists Keys and Values and puts them into a string.

}

return name;

}

private void btnShow\_Click(object sender, EventArgs e) // show entire inventory

{

foodString = MasterList(foodList, foodString); //Uses MasterList method and puts result into a string.

aidString = MasterList(firstaidList, aidString); //Uses MasterList method and puts result into a string.

weaponString = MasterList(weaponList, weaponString); //Uses MasterList method and puts result into a string.

string bigList = "Food:\n\n" + foodString + "\n"+"First Aid:\n\n"+ aidString + "\n"+ "Weapons:\n\n" + weaponString; //Puts everything into a neat, bigger list.

MessageBox.Show(bigList,"Master List"); //Displays final list.

}

private bool subtract(decimal value,string item, SortedList<string, decimal> list) // subtract quantity from inventory

{

if (list.ContainsKey(item)) //Checks if the item selected in listbox is actually in it.

{

list[item] -= value; //Subtracts the value of the Key item.

return true;

}

else

return false;

}

private bool AddValue(decimal value, string item, SortedList<string, decimal> list) // add quantity to inventory

{

if (list.ContainsKey(item)) //Checks if the item selected in listbox is actually in the list.

{

int itemIndex = list.IndexOfKey(item);

list[item] += value; //Adds the value of the Key item.

return true;

}

else

return false;

}

private void btnAdd\_Click(object sender, EventArgs e)

{

string listItem = listBox1.GetItemText(listBox1.SelectedItem);

//Uses AddValue method with given values.

AddValue(nudAmount.Value, listItem, foodList);

AddValue(nudAmount.Value, listItem, weaponList);

AddValue(nudAmount.Value, listItem, firstaidList);

nudAmount.Value = 0;

}

private void btnItemAdd\_Click(object sender, EventArgs e) // add item to list

{

if (txtAddItem.Text == "") //Checks for empty string and shows the message box if there's no input.

{

MessageBox.Show("Please enter an item to add.");

txtAddItem.Focus();

}

else //Otherwise...

{

string message = "Are you sure you want to add this item to the selected category?"; //Confirms whether you are sure you want to add item.

DialogResult button = MessageBox.Show(message, "Are you sure?", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (button == DialogResult.Yes)

{

if (rdoFood.Checked) //Makes sure which radio button is checked.

{

foodList.Add(txtAddItem.Text, 0); // Then adds item to the appropriate list.

UpdateList(foodList);

}

else if (rdoWeapons.Checked)

{

weaponList.Add(txtAddItem.Text, 0);

UpdateList(weaponList);

}

else if (radioButton3.Checked)

{

firstaidList.Add(txtAddItem.Text, 0);

UpdateList(firstaidList);

}

}

}

}

private void btnExit\_Click(object sender, EventArgs e)

{

this.Close();

}

private void txtAddItem\_TextChanged(object sender, EventArgs e)

{

}

private void nudAmount\_ValueChanged(object sender, EventArgs e)

{

}

private void btnClear\_Click(object sender, EventArgs e)

{ //Clears selected item from list.

string message = "Are you sure you want to remove this item from the selected category?"; //Confirms whether are you sure you want to remove item.

DialogResult button = MessageBox.Show(message, "Are you sure?", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (button == DialogResult.Yes) //If yes is selected...

{

string listItem = listBox1.GetItemText(listBox1.SelectedItem); //Gets Item and puts it into a string.

if (rdoFood.Checked)

{

foodList.Remove(listItem); //Removes item.

UpdateList(foodList); //Updates listbox immediately.

}

else if (rdoWeapons.Checked)

{

weaponList.Remove(listItem);

UpdateList(weaponList);

}

else if (radioButton3.Checked)

{

firstaidList.Remove(listItem);

UpdateList(firstaidList);

}

}

}

private void listBox1\_SelectedIndexChanged(object sender, EventArgs e)

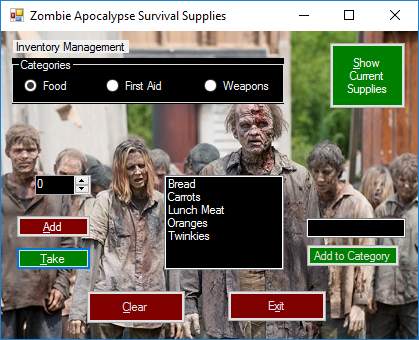
{

}

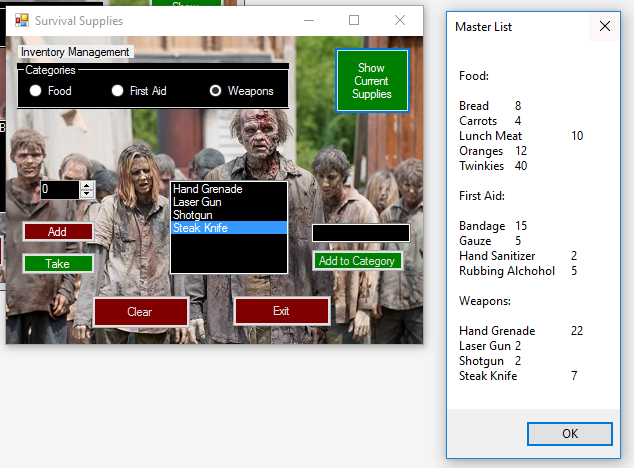
}

}

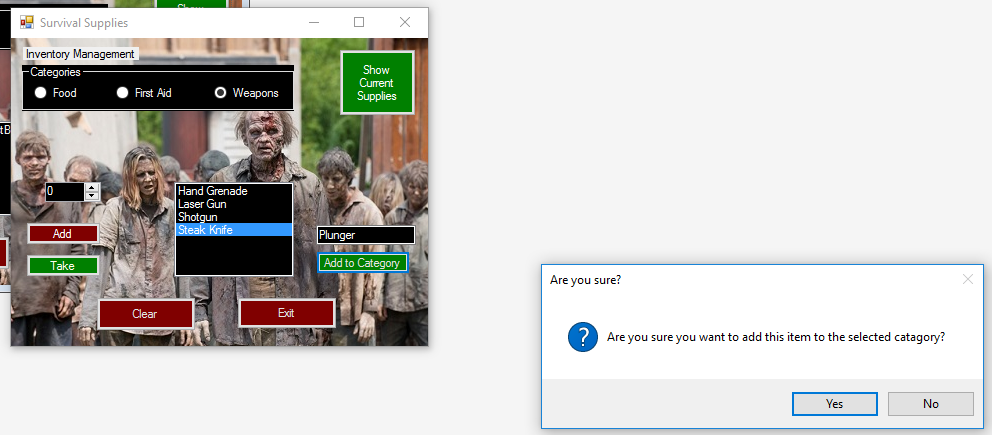
App:



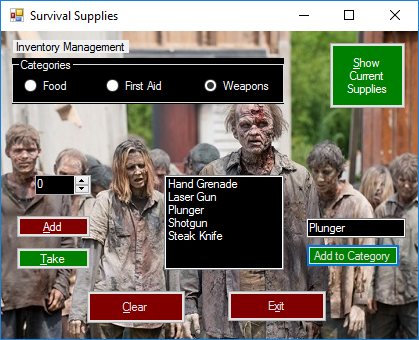
Selected items where inventory changed from supply list:



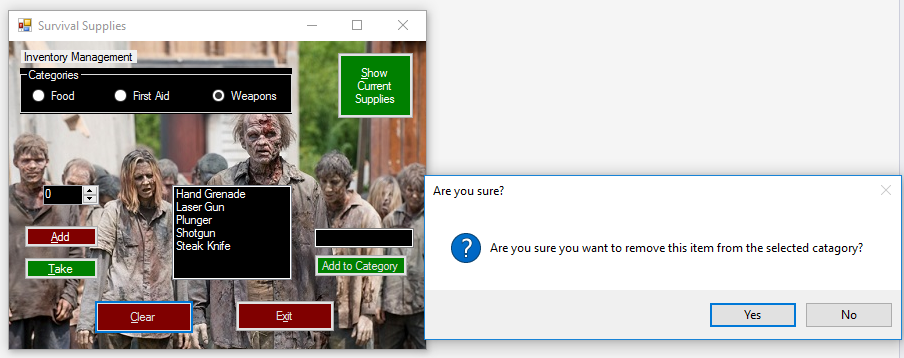
Added item to Weapons category and message box appears asking for confirmation:



Item added successfully:



Message box confirming removing an item from the list:



Data validation requiring user input:

