public Form1(string com_selected) // Standard Form Initialization
// ################################################################################################################

public void Form1_Shown(object sender, EventArgs e) // Set up defaults at startup when form is first shown
// ################################################################################################################

private void initDCC(string com_port) // Sets up communication with the COM port for railroad commands
// ################################################################################################################

public void SetupListViews() // Set up defaults for the listviews in the program
// ################################################################################################################

private void SetListViewProperties(ListView listview) // Sets the default settings of a listview for this program
// ################################################################################################################

public void CenterListView(ListView listview) // Centers the items in a listview
// ################################################################################################################

public void SetupColors() // Set up default colors that the user can select from the file
// ################################################################################################################

public void SetupTrackParts() // Set up default track parts to be used on track layout
// ################################################################################################################

public void SetupTurnouts() // Set up default turnout numbers and scheme for track layout
// ################################################################################################################

public void SetupAIUs() // Set up default aiu address from a file given - if not there, use the default and get first AIU status read
private byte[] SetupTurnoutCommand(int turnout_id, byte direction)
  // Sets up the command for the turnout to then later send to the track
private byte[] SetupForwardCommand(int train_id, int speed)
  // Sets up the command for moving a train forward to then later send to the track
private byte[] SetupBackwardCommand(int train_id, int speed)
  // Sets up the command for moving a train backward to then later send to the track

public void SetupBeginningLabels(Label label, int i)
  // Set up beginning labels colors and text
public void SetupLaterLabels(Label label)
  // Set up later labels colors
public void SetLabelAttributes()
  // Set up defaults for the labels and what labels go with what pieces
public void DefaultImages()
  // Set up default images with certain pictureBoxes
public void ShowTurnoutImages(int turnout_id, PictureBox bottom, PictureBox top)
  // Set up turnout images with certain images pending attributes
public void UpdateLabels(Label label, string item)
  // Standard delegate to use to update labels on main thread
public void HandleClientConnect(IAsyncResult sync_result)
// Connection to client initiated
// - New connection attempt has been made, process and see if a train is available. Set up sockets, etc.

public void HandleDatafromClient(Socket HolderSocket)
// Information received from Client
// - Bytes of data are now in the stream of which are sent from the client, callback was issued and this function
  is now called which handles the data

public void HandleDataReceivedfromClient(IAsyncResult sync_result)
// Information received from Client - now handle the data
// - Bytes of data are now in the stream of which are sent from the client, callback was issued and this function
  is now called which handles the data

private void ProcessClientRequest(string[] data)
// Function used if the server is sending a message to the client - not included right now

public void ServerSendResponse(IAsyncResult sync_result)
// Function used if the server is sending a message to the client - not included right now

private void timer1_Tick(object sender, EventArgs e)
// Utilize a timer to check AIU's to see where trains are moving on the track

private void timer2_Tick(object sender, EventArgs e)
// Utilize a timer to check blocked trains and send updates to clients

private void CommandDCC(int item, byte[] command)
// Sends commands to the railroad and uses a monitor to make sure commands are sent out one at a time

private void pictureBox4_Click(object sender, EventArgs e)
// User has clicked the upper Turnout 0
private void pictureBox3_Click(object sender, EventArgs e)
// User has clicked the lower Turnout 0
private void pictureBox6_Click(object sender, EventArgs e)
// User has clicked the upper Turnout 1
private void pictureBox5_Click(object sender, EventArgs e)
// User has clicked the lower Turnout 1
private void pictureBox8_Click(object sender, EventArgs e)
// User has clicked the upper Turnout 2
private void pictureBox7_Click(object sender, EventArgs e)
// User has clicked the lower Turnout 2
private void pictureBox10_Click(object sender, EventArgs e)
// User has clicked the upper Turnout 3
private void pictureBox9_Click(object sender, EventArgs e)
// User has clicked the lower Turnout 3
private void pictureBox14_Click(object sender, EventArgs e)
// User has clicked the upper Turnout 4
private void pictureBox13_Click(object sender, EventArgs e)
// User has clicked the lower Turnout 4
private void pictureBox12_Click(object sender, EventArgs e)
// User has clicked the upper Turnout 5
private void pictureBox11_Click(object sender, EventArgs e)
// User has clicked the lower Turnout 5
public void CheckTurnout(int turnout_id, string location, PictureBox bottom, PictureBox top)
// Main function for checking and calling to switch turnouts if applicable
private void pictureBox72_Click(object sender, EventArgs e)
// User clicked that they want to add a train - attempt to add it according to selected

private void pictureBox76_Click(object sender, EventArgs e)
// Remove trains 'button' clicked - remove selected train

public int GetTrainLocation(string train_id)
// Gets the trains current location

private void DeallocateRequestTrack(int train_location)
// Deallocate the current 'Requested' track at this current train location

private void DeallocateTrack(int train_location)
// Deallocate the current 'Track ID' track at this current train location

private void GetRequestedTrack(int train_location, int current_track, ref string track_piece, ref string next_sensor)
// Get the requested track and next sensor train should be going

private void RequestNextTrack(string train_id)
// Figures out what the next track is that the train is moving and attempts to gain ownership

private void SensorHitChange(string train_id)
// Train hit their next expected sensor - change structure information and images

private void CheckAIU()
// Gets the current AIU status from the 'Blackbox' to see if any sensors were hit
private void ChangeandRequestAIU(int sensor_id)
// Make sensor visible and look through Next Sensor Id’s for Trains to see if it matches

private void ChangeAIUImages(int[] value, int aiu_nbr, int counter)
// Logic to check AIU current status with default to change images to show 'hits'

private void pictureBox72_MouseHover(object sender, EventArgs e)
// Mouse is over the Add Train button, change to highlighted

private void pictureBox76_MouseHover(object sender, EventArgs e)
// Mouse is over the Remove button, change to highlighted

private void pictureBox72_MouseLeave(object sender, EventArgs e)
// Mouse has left the Add Train button, change to unhighlighted

private void pictureBox76_MouseLeave(object sender, EventArgs e)
// Mouse has left the Remove button, change to unhighlighted

private void CloseSocket(Socket TempSocket)
// User has disconnected - open up the train for another user to use

private void Form1_FormClosing(object sender, FormClosingEventArgs e)
// Form is closing - close any potentially open sockets

private void Form1_FormClosing(object sender, FormClosingEventArgs e)
// Form is closing - close any potentially open sockets
```csharp
private void button1_Click(object sender, EventArgs e)
// 'Save Logs' button has been clicked - display saveFileDialog and save

private void button7_Click(object sender, EventArgs e)
// Do a simple 'Print Preview' to print out at least the first page of the log information
// > Modified from an example of Dr. Blahnik's

private void pictureBox64_Click(object sender, EventArgs e)
// Expand to show logs 'button' clicked - expand or limit

public void WritetoLogs(string loginfo)
// Standard delegate to use to write logs to listBox1 on main thread

private void pd_PrintPage(object sender, PrintPageEventArgs ev)
// Populate print preview for printing on the outputs

private void button8_Click(object sender, EventArgs e)
// Standard print setup dialog that allows a user to set default printer, etc.
```