RoboShovel Coding Convention

- All methods will start with a capital letter. Standard java convention starts with a lower case letter so we will use this to quickly differentiate between our custom functions and the standard API.
- All local variables will start with a lowercase letter.
- All global variables will start with an underscore followed by a lowercase letter.
- All variables and functions must be labeled well enough to understand its use without even looking at implementation.
- All variables which are a compound of multiple words must capitalize all the compounded words. Some examples are:

```
private final String _someGuysName = "Some Guy";
public int myAge = 20;
private bool done = false;
```

- All interfaces must be named with the prefix "I" for easy interface identification for example:

```
/**************************************************************
 * The interface controller for the capabilities of a command.
 *
 * @author Akram Kassay
 **************************************************************/
public interface ICommand
{
    /*****************************************************************************
     * The logic for this command to execute.
     *
     * @param params any parameter that you would like to pass as an argument
     *****************************************************************************/
     public void Execute(object params);

    /*****************************************************************************
     * The logic required to determine whether or not the command can execute.
     *
     * @param params any parameter that you would like to pass as an argument
     * @return a boolean indicating whether or not the command is allowed to execute
     *****************************************************************************/
     public bool CanExecute(object params);
}
```

- All if, while or for statements will utilize curly braces despite whether or not the internal content is one line. The only exception to this rule is inline notation of code for example:

```
return IsConnected() ? NetworkStatus.Success : NetworkStatus.Disconnected;
```

- All curly braces will appear on their own separate line for example:

```
if (IsConnected())
{
    return NetworkStatus.Success;
}
else
{
    return NetworkStatus.Disconnected;
}
```

- Try to use the "final" modifier as much as possible for safer code. It prevents further modifications being done to the class, method or variable and makes the code a bit safer.
- All locks (mutexes) utilized for multithreading MUST end with the word "lock". Some examples are:

```
private final _ioStreamLock = new ReentrantLock();
private final modelInfoLock = new ReentrantLock();
private final lock = new ReentrantLock();
```

- All thread declarations must end with the word "thread". Some examples are:

```
private Thread eventRegistrationThread = new Thread("EventRegistration");
private Thread ioCommThread = new Thread(ioRunnable, "IOCommunications");
```

- All classes which possess properties require modifier methods, these getter and setter functions must have either prefixes "Get" or "Set", for example:

```
/*************************************************************
 * Definition of a person for this random example.
 *
 * @author Akram Kassay
 *************************************************************/
public final class Person
{
	/** The age of the person **/
	private int _age;

	/*****************************************************************************
	 * Create a person.
	 *
	 * @param age the age of the person
	 *****************************************************************************/
	public void Person(int age)
	{
		_age = age;
	}
	/*****************************************************************************
	 * Gets the age of the person.
	 *
	 * @return the age of person
	 *****************************************************************************/
	public int GetAge()
	{
		Return _age;
	}
	/*****************************************************************************
	 * Sets the age of the person.
	 *
	 * @param age the age of the person
	 *****************************************************************************/
	public void SetAge(int age)
	{
		_age = age;
	}
}
```

- Inline interfaces are NOT allowed for the final code. They can be done during debugging and testing but must eventually be moved into their own class. An example of an inline interface is:

```
button.addOnClickListener(new OnClickListener()
{
	public void OnClick(View v, bool pressed)
	{
		_networkConnected = pressed;
	}
});
```

Instead do this below:

```
/*************************************************************
 * Handles network connect button.
 *
 * @author Akram Kassay
 *************************************************************/
public final class NetworkConnectButtonListener implements OnClickListener
{
```

```
	/*************************************************************
	 * Actions to perform when the network connect button is pressed.
	 *
	 * @param v the button which was pressed on the UI
	 * @param pressed the state of the button (down = true, up = false)
	 *************************************************************/
	public void onClick(View v)
	{
		_networkConnected = v.isPressed();
	}
}
```

Then back in main…

```
button.addOnClickListener(new NetworkConnectButtonListener());
```