Merhaba arkadaşlar

Android telefonumuz ve arduino kullanarak yaptığımız rc arabayı sizlerle paylaşmak istiyorum.Aşama aşama arabayı nasıl yaptığımı neler kullandığımı inceleyebilirsiniz.

## Kullandığımız Malzemeler:

120x60mm 4mm Şaft Aparatlı Teker x  4

Arduino x 1

HC05 Bluetooth-Serial Modül x 1

12V 37mm 700Rpm Redüktörlü DC Motor x 4

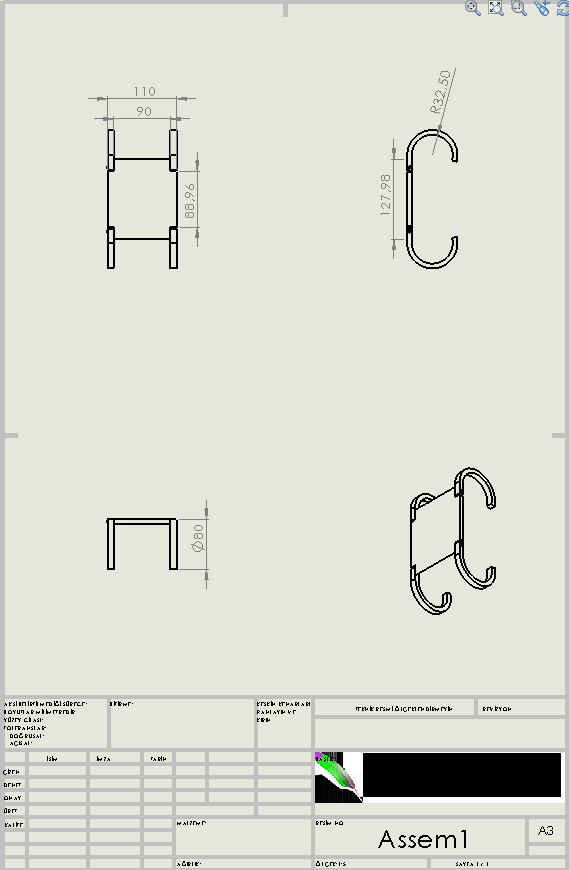
L298N Voltaj Regulatörlü Çift Motor Sürücü Kartı x 2

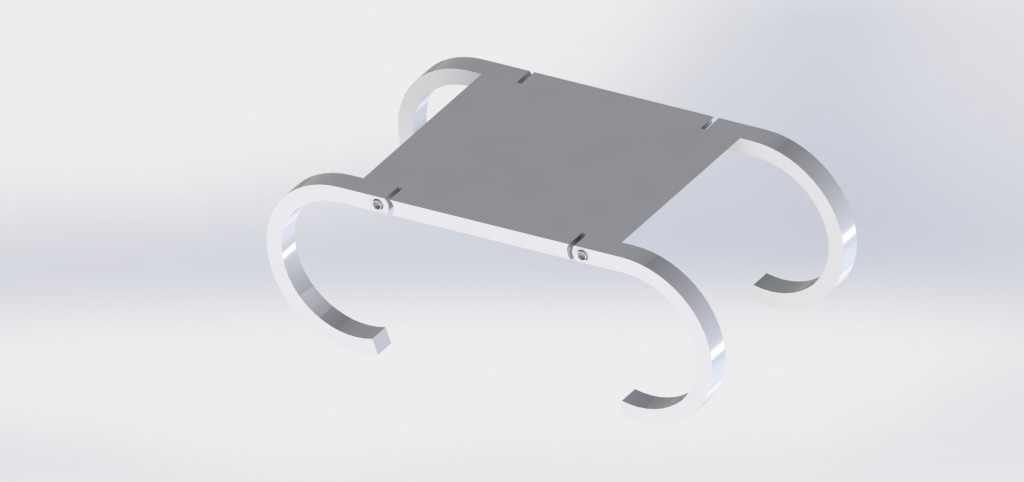
12v Lipo Pil x 1

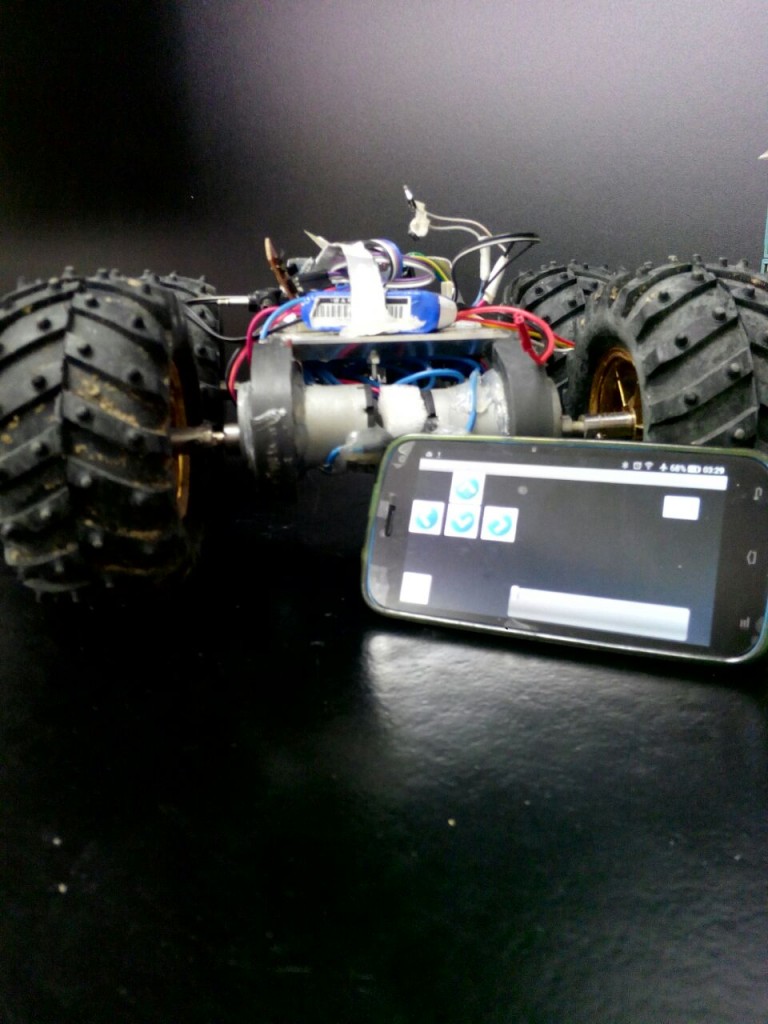
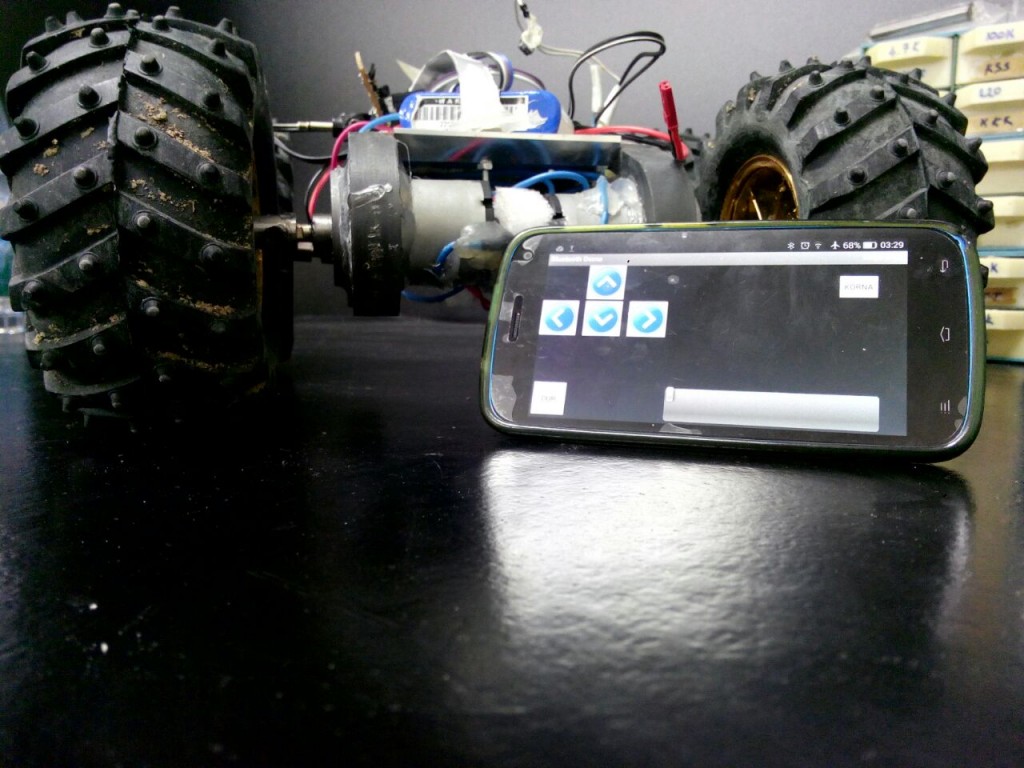
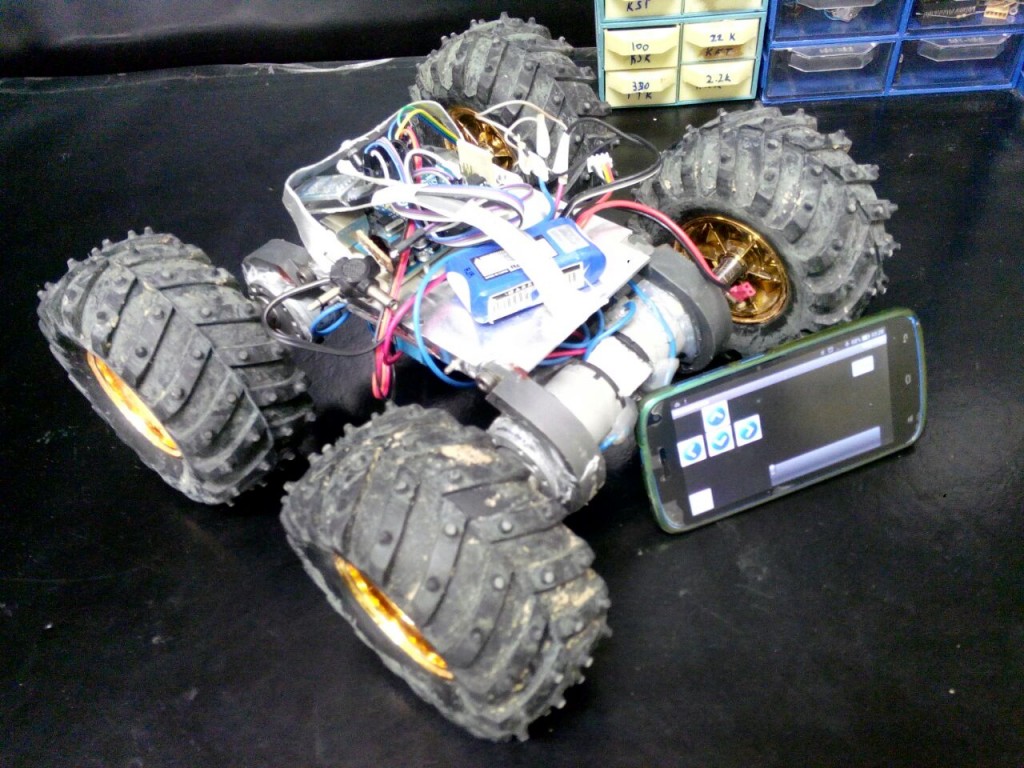
Erkek-Dişi M-M Jumper Kablo (40'lı satılıyor fiyatı 8tl arası)

 Ana gövde (Şase) size kalmış.İsterseniz ben aşağıda SolidWorks Çizimini paylaştım.

Arkadaşlar cihazın kontrol yazılımları geçmeden mekanik tasarımından ve nasıl yaptığıma dair ufak bir bilgi ve görsel paylaşım yapayım.Ana şaseyi bir arkadaşım özel olarak cnc'de üretmişti .Siz istediğiniz şekilde şase tasarımı yapabilirsiniz.Şase tasarımında dikkat etmeniz gereken yer motorların ve motor sürücülerin yerleşimine uygun bir şase yapmanız.Şasemizin SolidWorks çizimine buradan ulaşabilirsiniz.

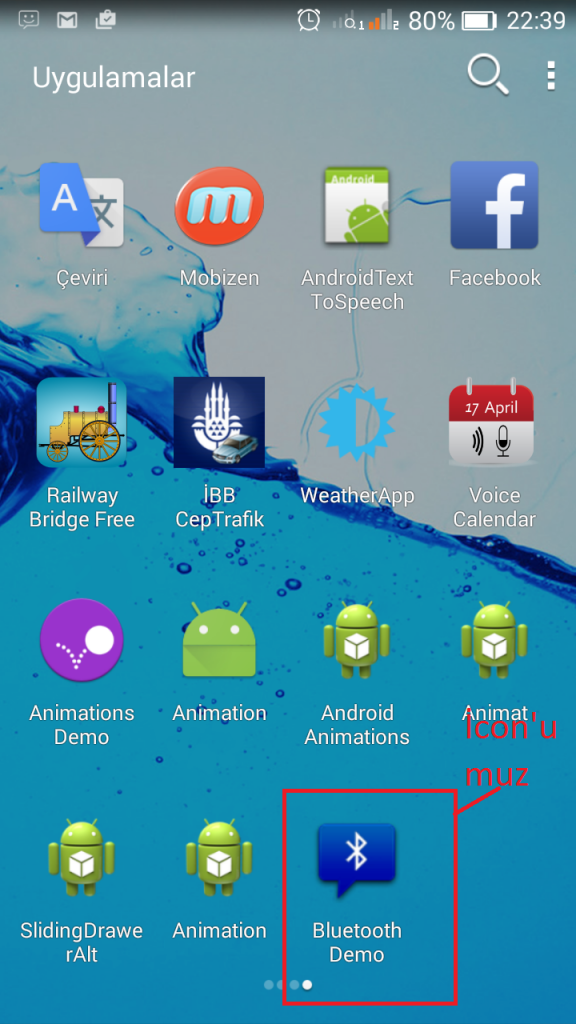
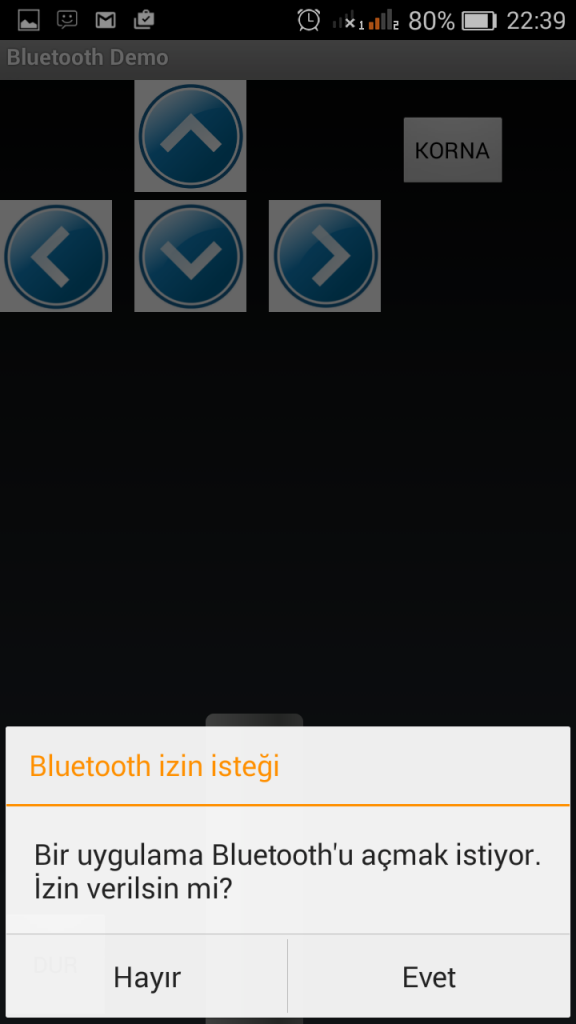
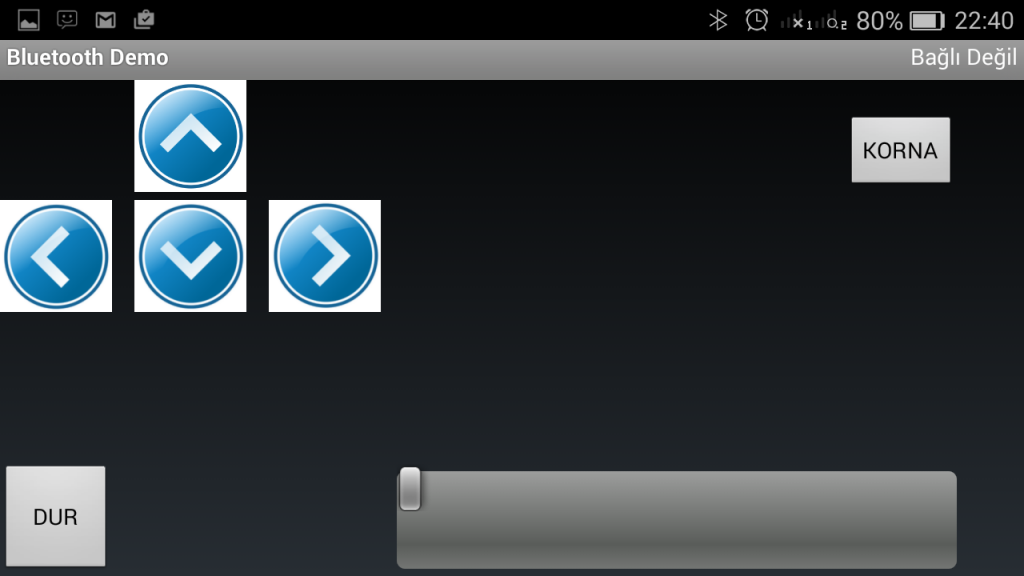
[](http://stackcuriosity.com/wp-content/uploads/2015/02/teknikresim.png)

[](http://stackcuriosity.com/wp-content/uploads/2015/02/2.jpg)

[](http://stackcuriosity.com/wp-content/uploads/2015/02/IMG-20150211-WA0000.jpg)[/](http://stackcuriosity.com/wp-content/uploads/2015/02/IMG-20150211-WA0001.jpg)[](http://stackcuriosity.com/wp-content/uploads/2015/02/IMG-20150211-WA0002.jpg)[](http://stackcuriosity.com/wp-content/uploads/2015/02/IMG-20150211-WA0003.jpg)

## 

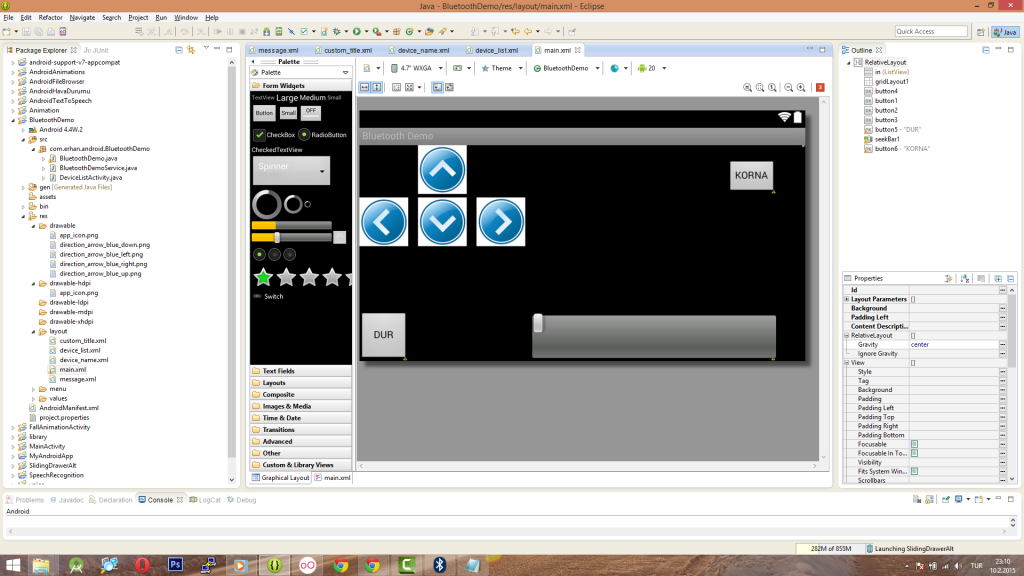
## Uygulama Görselleri:

[](http://stackcuriosity.com/wp-content/uploads/2015/02/Screenshot_2015-02-10-22-39-39.png)[](http://stackcuriosity.com/wp-content/uploads/2015/02/Screenshot_2015-02-10-22-39-50.png)[](http://stackcuriosity.com/wp-content/uploads/2015/02/Screenshot_2015-02-10-22-40-04.png)

Kısaca Android arayüzümüzü anlatacak olursam. Dur butonu(hız verdikden sonra elinizi hızdan çekseniz bile aracın aldığı ivme ile belirli bir süre devam etmesini engellemek için koydum) ,Korna(arabamıza bir tane buzzer koydum:) korna amaçlı güzelde:) ),hız kontrol seekbar'ı(Aracımızın hızı arttırmak ve azaltmak için).Sağ-sol-ileri-geri( yön tuşları ). Arayüzümüzün hepsi bunlardan ibaret.

Önce android kısmından başlayalım:

Uygulamamızın eclipse dosya yapısı ve uygulamanın android arayüz tasarımı resimdeki gibi.

[](http://stackcuriosity.com/wp-content/uploads/2015/02/eclipse.png)

### ****BluetoothDemo.java****

Arkadaşlar bu dosyada çalışmaya ilk başladığımdan  itibaren programlama aşamalarım var.Bilerek silmedim önce motorları kontrol etmeden ledleri yakıp söndürneye çalıştım.Sizde yeni başlıyorsanız tavsiye ederim.Gerçi bu yazıyı tamamen uyguladığınız taktirde sorunsuz birşekilde android ile bluetooth üzerinden rc araba kontrol etmeyi başarabilirsiniz.

/\*

\* Copyright (C) 2009 The Android Open Source Project

\*/

package com.stackcuriosity.android.BluetoothDemo;

import android.annotation.SuppressLint;

import android.app.Activity;

import android.bluetooth.BluetoothAdapter;

import android.bluetooth.BluetoothDevice;

import android.content.Intent;

import android.os.Bundle;

import android.os.Handler;

import android.os.Message;

import android.util.Log;

//import android.util.Log;

//import android.view.KeyEvent;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.MotionEvent;

import android.view.View;

import android.view.View.OnLongClickListener;

import android.view.View.OnTouchListener;

import android.view.Window;

import android.view.View.OnClickListener;

//import android.view.inputmethod.EditorInfo;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.EditText;

//import android.widget.EditText;

import android.widget.ListView;

import android.widget.ProgressBar;

import android.widget.RadioButton;

import android.widget.SeekBar;

import android.widget.SeekBar.OnSeekBarChangeListener;

import android.widget.TextView;

import android.widget.Toast;

import android.widget.ToggleButton;

/\*\*

\* This is the main Activity that displays the current chat session.

\*/

public class BluetoothDemo extends Activity {

// Debugging

//private static final String TAG = "BluetoothDemo";

//private static final boolean D = true;

//BluetoothChatService Handler dan gönderilen mesaj tipleri

public static final int MESSAGE\_STATE\_CHANGE = 1;

public static final int MESSAGE\_READ = 2;

public static final int MESSAGE\_WRITE = 3;

public static final int MESSAGE\_DEVICE\_NAME = 4;

public static final int MESSAGE\_TOAST = 5;

// BluetoothChatService Handler'dan Anahtar kelimelerin alınması için

public static final String DEVICE\_NAME = "device\_name";

public static final String TOAST = "toast";

// Intent kodları

private static final int REQUEST\_CONNECT\_DEVICE = 1;

private static final int REQUEST\_ENABLE\_BT = 2;

// Layout ekran görünümü için

private TextView mTitle;

private ListView mConversationView;

//private EditText mOutEditText;

//private Button mSendButton;

private SeekBar seekbar;

private EditText edt;

private Button SagButon;

private Button SolButon;

private Button IleriButon;

private Button GeriButon;

private Button AcilStop;

private Button Korna;

/\* private RadioButton Led1;

private RadioButton Led2;

private ProgressBar sicaklik;

private ToggleButton durum;

private TextView isidegeri;\*/

/\*private boolean LED1check=false;

private boolean LED2check=false;

private byte[] veri = new byte[1]; //1 byte veri gönderme değişkeni

\*/

//Bağlı aygıtın adı

private String mConnectedDeviceName = null;

// Array adapter'ı bluetooghtla iletişim için kullanıyoruz.

private ArrayAdapter<String> mConversationArrayAdapter;

// String buffer for outgoing messages

//private StringBuffer mOutStringBuffer;

// Local Bluetooth adapter

private BluetoothAdapter mBluetoothAdapter = null;

// Bluetoothla iletşim servisi tanımlama

private BluetoothDemoService mChatService = null;

@SuppressLint("NewApi")

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

//if(D) Log.e(TAG, "+++ ON CREATE +++");

//layout pencereli kurmak için

requestWindowFeature(Window.FEATURE\_CUSTOM\_TITLE);

setContentView(R.layout.main);

getWindow().setFeatureInt(Window.FEATURE\_CUSTOM\_TITLE, R.layout.custom\_title);

// text view ımızı tanıttık

mTitle = (TextView) findViewById(R.id.title\_left\_text);

mTitle.setText(R.string.app\_name);

// Takılı Bluetooth adapter'ımızı aldık.

mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();

// Adaptör null ise o zaman, Bluetooth desteklenmiyordur.

if (mBluetoothAdapter == null) {

Toast.makeText(this, "Bluetooth mevcut değil", Toast.LENGTH\_LONG).show();

finish();

return;

}

}

@SuppressLint("NewApi")

@Override

public void onStart() {

super.onStart();

//if(D) Log.e(TAG, "++ ON START ++");

if (!mBluetoothAdapter.isEnabled()) {

Intent enableIntent = new Intent(BluetoothAdapter.ACTION\_REQUEST\_ENABLE);

startActivityForResult(enableIntent, REQUEST\_ENABLE\_BT);

// Otherwise, setup the chat session

} else {

if (mChatService == null) setupChat();

}

}

@Override

public synchronized void onResume() {

super.onResume();

try {

//Log.i("zms", "buluetooth bağlandıı");

// Performing this check in onResume() covers the case in which BT was

// not enabled during onStart(), so we were paused to enable it...

// onResume() will be called when ACTION\_REQUEST\_ENABLE activity returns.

if (mChatService != null) {

// Only if the state is STATE\_NONE, do we know that we haven't started already

if (mChatService.getState() == BluetoothDemoService.STATE\_NONE) {

// Start the Bluetooth chat services

mChatService.start();

}

}

} catch (Exception e) {

// TODO: handle exception

}

}

private void setupChat() {

try {

//Log.d(TAG, "setupChat()");

// konuşma thread için dizi adaptörünü başlatmak

mConversationArrayAdapter = new ArrayAdapter<String>(this, R.layout.message);

mConversationView = (ListView) findViewById(R.id.in);

mConversationView.setAdapter(mConversationArrayAdapter);

seekbar = (SeekBar) findViewById(R.id.seekBar1);

seekbar.setMax(255);

seekbar.setRotation(0);

//seekbar.setScrollX(50);

seekbar.setOnSeekBarChangeListener(new OnSeekBarChangeListener() {

public void onStopTrackingTouch(SeekBar seekBar) {

// TODO Auto-generated method stub

}

public void onStartTrackingTouch(SeekBar seekBar) {

// TODO Auto-generated method stub

}

public void onProgressChanged(SeekBar seekBar, int progress,

boolean fromUser) {

// TODO Auto-generated method stub

//deger.setText("SeekBar Değeri : "+progress);

Log.i("progress değeri", ""+progress);

sendMessage(""+progress);

}

});

//Buton1'e basıldığında LED1 in durumuna göre LED1 yakılır veya söndürülür.

SagButon = (Button) findViewById(R.id.button2);

SagButon.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

/\*if(LED1check){

veri[0] &= ~(1);}

else

{veri[0] |= 1;}

sendData(veri);\*/

sendMessage("801");

seekbar.setProgress(0);

//sendMessage("0");

Toast.makeText(getApplicationContext(), "SAĞA DÖNN", Toast.LENGTH\_LONG).show();

}

});

//Buton2'ye basıldığında LED2 in durumuna göre LED2 yakılır veya söndürülür.

SolButon = (Button) findViewById(R.id.button1);

SolButon.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

/\*if(LED2check){

veri[0] &= ~(0);}

else

{veri[0] |= 0;}

sendData(veri);\*/

sendMessage("802");

seekbar.setProgress(0);

//sendMessage("0");

Toast.makeText(getApplicationContext(), "SOLA DÖNN", Toast.LENGTH\_LONG).show();

}

});

IleriButon = (Button) findViewById(R.id.button4);

IleriButon.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

// TODO Auto-generated method stub

sendMessage("803");

seekbar.setProgress(0);

//sendMessage("0");

Toast.makeText(getApplicationContext(), "İLERİ MARŞŞ", Toast.LENGTH\_LONG).show();

}

});

GeriButon = (Button)findViewById(R.id.button3);

GeriButon.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

sendMessage("804");

seekbar.setProgress(0);

//sendMessage("0");

Toast.makeText(getApplicationContext(), "GERİ DÖN LAAA", Toast.LENGTH\_LONG).show();

}

});

AcilStop =(Button)findViewById(R.id.button5);

AcilStop.setOnTouchListener(new OnTouchListener() {

public boolean onTouch(View v, MotionEvent event) {

if (event.getAction() == MotionEvent.ACTION\_DOWN ) {

sendMessage("808");

return true;

}

else if (event.getAction() == MotionEvent.ACTION\_UP ) {

sendMessage("805");

return true;

}

return false;

}

});

/\* AcilStop.setOnLongClickListener(new OnLongClickListener() {

public boolean onLongClick(View v) {

sendMessage("808");

return false;

}

});

AcilStop.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

// TODO Auto-generated method stub

sendMessage("805");

seekbar.setProgress(0);

Toast.makeText(getApplicationContext(), "DURSANA LANN", Toast.LENGTH\_SHORT).show();

}

});\*/

Korna=(Button)findViewById(R.id.button6);

Korna.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

// TODO Auto-generated method stub

sendMessage("806");

Toast.makeText(getApplicationContext(), "GAÇIN ULANNNN",Toast.LENGTH\_SHORT).show();

sendMessage("807");

}

});

// Bluetooth bağlantıları gerçekleştirmek için BluetoothChatService başlatılmalı

mChatService = new BluetoothDemoService(this, mHandler);

// Initialize the buffer for outgoing messages

//mOutStringBuffer = new StringBuffer("");

} catch (Exception e) {

// TODO: handle exception

}

}

@Override

public synchronized void onPause() {

super.onPause();

//if(D) Log.e(TAG, "- ON PAUSE -");

}

@Override

public void onStop() {

super.onStop();

//if(D) Log.e(TAG, "-- ON STOP --");

}

@Override

public void onDestroy() {

super.onDestroy();

//Bluetooth iletişim servislerini durdurmak için

if (mChatService != null) mChatService.stop();

//if(D) Log.e(TAG, "--- ON DESTROY ---");

}

@SuppressLint("NewApi")

private void ensureDiscoverable() {

if (mBluetoothAdapter.getScanMode() !=

BluetoothAdapter.SCAN\_MODE\_CONNECTABLE\_DISCOVERABLE) {

Intent discoverableIntent = new Intent(BluetoothAdapter.ACTION\_REQUEST\_DISCOVERABLE);

discoverableIntent.putExtra(BluetoothAdapter.EXTRA\_DISCOVERABLE\_DURATION, 300);

startActivity(discoverableIntent);

}

}

private void sendData(byte[] send){

if (mChatService.getState() != BluetoothDemoService.STATE\_CONNECTED) {

Toast.makeText(this, R.string.not\_connected, Toast.LENGTH\_SHORT).show();

return;

}

if(send.length>0)

mChatService.write(send);

}

/\*\*

\* Sends a message.

\* @param message A string of text to send.

\*/

private void sendMessage(String message) {

// Check that we're actually connected before trying anything

if (mChatService.getState() != BluetoothDemoService.STATE\_CONNECTED) {

Toast.makeText(this, R.string.not\_connected, Toast.LENGTH\_SHORT).show();

return;

}

// Check that there's actually something to send

if (message.length() > 0) {

// Get the message bytes and tell the BluetoothChatService to write

byte[] send = message.getBytes();

mChatService.write(send);

}

}

//Handler, BluetoothChatService den geri dönen bilgileri yakalar.

private final Handler mHandler = new Handler() {

@Override

public void handleMessage(Message msg) {

switch (msg.what) {

case MESSAGE\_STATE\_CHANGE:

//if(D) Log.i(TAG, "MESSAGE\_STATE\_CHANGE: " + msg.arg1);

switch (msg.arg1) {

case BluetoothDemoService.STATE\_CONNECTED:

mTitle.setText(R.string.title\_connected\_to);

mTitle.append(mConnectedDeviceName);

mConversationArrayAdapter.clear();

break;

case BluetoothDemoService.STATE\_CONNECTING:

mTitle.setText(R.string.title\_connecting);

break;

case BluetoothDemoService.STATE\_LISTEN:

case BluetoothDemoService.STATE\_NONE:

mTitle.setText(R.string.title\_not\_connected);

break;

}

break;

/\*case MESSAGE\_WRITE:

byte[] writeBuf = (byte[]) msg.obj;

//Toast.makeText(getApplicationContext(), "Bilgi gönderildi",Toast.LENGTH\_SHORT).show();

// construct a string from the buffer

String writeMessage = new String(writeBuf);

mConversationArrayAdapter.add("Me: " + writeMessage);

break;\*/

/\*case MESSAGE\_READ:

byte[] readBuf = (byte[]) msg.obj;\*/

/\*Led1 = (RadioButton) findViewById(R.id.LED1);

Led2 = (RadioButton) findViewById(R.id.LED2);

sicaklik = (ProgressBar) findViewById(R.id.progressBar1);

durum = (ToggleButton) findViewById(R.id.toggleButton1);

isidegeri = (TextView) findViewById(R.id.sicaklik);\*/

//Gelen verinin 8. biti 1 ise veri LED ve Buton durum bilgisi olarak değerlendirilir.

/\* if((readBuf[0] & 0x80) == 0x80){

//Gelen verinin 1. bitini durumuna göre LED1 göstergesi düzenlenir.

if((readBuf[0] & 0x01) == 0x01){

Led1.setChecked(true);

LED1check=true;}

else {

Led1.setChecked(false);

LED1check=false;}

//Gelen verinin 2. bitinin durumuna göre LED2 göstergesi düzenlenir.

if((readBuf[0] & 0x02)==0x02){

Led2.setChecked(true);

LED2check=true;}

else{

Led2.setChecked(false);

LED2check=false;}

//Gelen verinin 3. bitinin durumuna göre Buton durum göstergesi düzenlenir.

if((readBuf[0] & 0x04)==0x04)

durum.setChecked(true);

else

durum.setChecked(false);}

//Gelen verinin 8. biti sıfır ise gelen veri sıcaklık bilgisi olarak değerlendirilir.

else {

if(readBuf[0]>100) readBuf[0]=100; //Sıcaklık değeri max 100 dereceyi aşmıcak şekilde ayarlanır.

//Sıcaklık bilgisi görüntülenir.

sicaklik.setProgress((int)readBuf[0]);

isidegeri.setText("Sıcaklık="+readBuf[0]+"°");}

break;\*/

case MESSAGE\_DEVICE\_NAME:

// save the connected device's name

mConnectedDeviceName = msg.getData().getString(DEVICE\_NAME);

Toast.makeText(getApplicationContext(), "Connected to "

+ mConnectedDeviceName, Toast.LENGTH\_SHORT).show();

break;

case MESSAGE\_TOAST:

Toast.makeText(getApplicationContext(), msg.getData().getString(TOAST),

Toast.LENGTH\_SHORT).show();

break;

}

}

};

@SuppressLint("NewApi")

public void onActivityResult(int requestCode, int resultCode, Intent data) {

//if(D) Log.d(TAG, "onActivityResult " + resultCode);

switch (requestCode) {

case REQUEST\_CONNECT\_DEVICE:

// When DeviceListActivity returns with a device to connect

if (resultCode == Activity.RESULT\_OK) {

// Get the device MAC address

String address = data.getExtras()

.getString(DeviceListActivity.EXTRA\_DEVICE\_ADDRESS);

// Get the BLuetoothDevice object

BluetoothDevice device = mBluetoothAdapter.getRemoteDevice(address);

// Attempt to connect to the device

mChatService.connect(device);

}

break;

case REQUEST\_ENABLE\_BT:

//

if (resultCode == Activity.RESULT\_OK) {

// Bluetooth artık etkin

setupChat();

} else {

//Log.d(TAG, "BT not enabled");

Toast.makeText(this, R.string.bt\_not\_enabled\_leaving, Toast.LENGTH\_SHORT).show();

finish();

}

}

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.option\_menu, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case R.id.scan:

//Aygıtları görmek için DeviceListActivity başlatın ve tarama yaptırın

Intent serverIntent = new Intent(this, DeviceListActivity.class);

startActivityForResult(serverIntent, REQUEST\_CONNECT\_DEVICE);

return true;

case R.id.discoverable:

//Bu cihaz başkaları tarafından görülebilir olduğundan emin olun

ensureDiscoverable();

return true;

}

return false;

}

}

### ****BluetoothDemoService.java****

/\*

\* Copyright (C) 2009 The Android Open Source Project

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

package com.stackcuriosity.android.BluetoothDemo;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.util.UUID;

import android.bluetooth.BluetoothAdapter;

import android.bluetooth.BluetoothDevice;

import android.bluetooth.BluetoothServerSocket;

import android.bluetooth.BluetoothSocket;

import android.content.Context;

import android.os.Bundle;

import android.os.Handler;

import android.os.Message;

//import android.util.Log;

/\*\*

\* This class does all the work for setting up and managing Bluetooth

\* connections with other devices. It has a thread that listens for

\* incoming connections, a thread for connecting with a device, and a

\* thread for performing data transmissions when connected.

\*/

public class BluetoothDemoService {

// Debugging

//private static final String TAG = "BluetoothChatService";

//private static final boolean D = true;

// Sunucu soketi oluştururken SDP kayıdı için Ad

private static final String NAME = "BluetoothRcCar";

// Bu uygulama için benzersiz UUID'si

private static final UUID MY\_UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

// Member fields

private final BluetoothAdapter mAdapter;

private final Handler mHandler;

private AcceptThread mAcceptThread;

private ConnectThread mConnectThread;

private ConnectedThread mConnectedThread;

private int mState;

// Constants that indicate the current connection state

public static final int STATE\_NONE = 0; //bağlantı boşta

public static final int STATE\_LISTEN = 1; // gelen bağlantılar için dinleme

public static final int STATE\_CONNECTING = 2; // Bağlantı kurulurken

public static final int STATE\_CONNECTED = 3; // Şimdi bluetooth cihazına bağlandı

/\*\*

\* Constructor. Prepares a new BluetoothChat session.

\* @param context The UI Activity Context

\* @param handler A Handler to send messages back to the UI Activity

\*/

public BluetoothDemoService(Context context, Handler handler) {

mAdapter = BluetoothAdapter.getDefaultAdapter();

mState = STATE\_NONE;

mHandler = handler;

}

/\*\*

\* Set the current state of the chat connection

\* @param state An integer defining the current connection state

\*/

private synchronized void setState(int state) {

//if (D) Log.d(TAG, "setState() " + mState + " -> " + state);

mState = state;

//UI Etkinliği güncelleme böylece Handler için yeni durum ataması yapılır

mHandler.obtainMessage(BluetoothDemo.MESSAGE\_STATE\_CHANGE, state, -1).sendToTarget();

}

/\*\*

\* Return the current connection state. \*/

public synchronized int getState() {

return mState;

}

/\*\*

\* Start the chat service. Specifically start AcceptThread to begin a

\* session in listening (server) mode. Called by the Activity onResume() \*/

public synchronized void start() {

//if (D) Log.d(TAG, "start");

//Bir bağlantı yapılırken başka bağlantıyı iptal etmek için

if (mConnectThread != null) {mConnectThread.cancel(); mConnectThread = null;}

// Şu anda bir bağlantı çalıştıran herhangi bir parçacığı iptal için

if (mConnectedThread != null) {mConnectedThread.cancel(); mConnectedThread = null;}

// BluetoothServerSocket i dinlemek için thread başlangıcı

if (mAcceptThread == null) {

mAcceptThread = new AcceptThread();

mAcceptThread.start();

}

setState(STATE\_LISTEN);

}

/\*\*

\* Start the ConnectThread to initiate a connection to a remote device.

\* @param device The BluetoothDevice to connect

\*/

public synchronized void connect(BluetoothDevice device) {

//if (D) Log.d(TAG, "connect to: " + device);

// Cancel any thread attempting to make a connection

if (mState == STATE\_CONNECTING) {

if (mConnectThread != null) {mConnectThread.cancel(); mConnectThread = null;}

}

// Cancel any thread currently running a connection

if (mConnectedThread != null) {mConnectedThread.cancel(); mConnectedThread = null;}

// Start the thread to connect with the given device

mConnectThread = new ConnectThread(device);

mConnectThread.start();

setState(STATE\_CONNECTING);

}

/\*\*

\* Start the ConnectedThread to begin managing a Bluetooth connection

\* @param socket The BluetoothSocket on which the connection was made

\* @param device The BluetoothDevice that has been connected

\*/

public synchronized void connected(BluetoothSocket socket, BluetoothDevice device) {

//if (D) Log.d(TAG, "connected");

// Bağlantıyı tamamladı thread iptal

if (mConnectThread != null) {mConnectThread.cancel(); mConnectThread = null;}

// Şu anda bir bağlantı çalıştıran herhangi bir parçacığı iptal

if (mConnectedThread != null) {mConnectedThread.cancel(); mConnectedThread = null;}

// Cancel the accept thread because we only want to connect to one device

if (mAcceptThread != null) {mAcceptThread.cancel(); mAcceptThread = null;}

// Bağlantıyı yönetmek ve yayınlar gerçekleştirmek için thread başlatın

mConnectedThread = new ConnectedThread(socket);

mConnectedThread.start();

// Bağlı aygıtın adını gönderin

Message msg = mHandler.obtainMessage(BluetoothDemo.MESSAGE\_DEVICE\_NAME);

Bundle bundle = new Bundle();

bundle.putString(BluetoothDemo.DEVICE\_NAME, device.getName());

msg.setData(bundle);

mHandler.sendMessage(msg);

setState(STATE\_CONNECTED);

}

/\*\*

\* Stop all threads

\*/

public synchronized void stop() {

//if (D) Log.d(TAG, "stop");

if (mConnectThread != null) {mConnectThread.cancel(); mConnectThread = null;}

if (mConnectedThread != null) {mConnectedThread.cancel(); mConnectedThread = null;}

if (mAcceptThread != null) {mAcceptThread.cancel(); mAcceptThread = null;}

setState(STATE\_NONE);

}

/\*\*

\* Write to the ConnectedThread in an unsynchronized manner

\* @param out The bytes to write

\* @see ConnectedThread#write(byte[])

\*/

public void write(byte[] out) {

// Create temporary object

ConnectedThread r;

//

synchronized (this) {

if (mState != STATE\_CONNECTED) return;

r = mConnectedThread;

}

//

r.write(out);

}

/\*\*

\* Indicate that the connection attempt failed and notify the UI Activity.

\*/

private void connectionFailed() {

setState(STATE\_LISTEN);

// Send a failure message back to the Activity

Message msg = mHandler.obtainMessage(BluetoothDemo.MESSAGE\_TOAST);

Bundle bundle = new Bundle();

bundle.putString(BluetoothDemo.TOAST, "Unable to connect device");

msg.setData(bundle);

mHandler.sendMessage(msg);

}

/\*\*

\* Indicate that the connection was lost and notify the UI Activity.

\*/

private void connectionLost() {

setState(STATE\_LISTEN);

// Send a failure message back to the Activity

Message msg = mHandler.obtainMessage(BluetoothDemo.MESSAGE\_TOAST);

Bundle bundle = new Bundle();

bundle.putString(BluetoothDemo.TOAST, "Device connection was lost");

msg.setData(bundle);

mHandler.sendMessage(msg);

}

/\*\*

\* This thread runs while listening for incoming connections. It behaves

\* like a server-side client. It runs until a connection is accepted

\* (or until cancelled).

\*/

private class AcceptThread extends Thread {

// The local server socket

private final BluetoothServerSocket mmServerSocket;

public AcceptThread() {

BluetoothServerSocket tmp = null;

// Create a new listening server socket

try {

tmp = mAdapter.listenUsingRfcommWithServiceRecord(NAME, MY\_UUID);

} catch (IOException e) {

//Log.e(TAG, "listen() failed", e);

}

mmServerSocket = tmp;

}

public void run() {

//if (D) Log.d(TAG, "BEGIN mAcceptThread" + this);

setName("AcceptThread");

BluetoothSocket socket = null;

// Listen to the server socket if we're not connected

while (mState != STATE\_CONNECTED) {

try {

// This is a blocking call and will only return on a

// successful connection or an exception

socket = mmServerSocket.accept();

} catch (IOException e) {

//Log.e(TAG, "accept() failed", e);

break;

}

// If a connection was accepted

if (socket != null) {

synchronized (BluetoothDemoService.this) {

switch (mState) {

case STATE\_LISTEN:

case STATE\_CONNECTING:

// Situation normal. Start the connected thread.

connected(socket, socket.getRemoteDevice());

break;

case STATE\_NONE:

case STATE\_CONNECTED:

// Either not ready or already connected. Terminate new socket.

try {

socket.close();

} catch (IOException e) {

//Log.e(TAG, "Could not close unwanted socket", e);

}

break;

}

}

}

}

//if (D) Log.i(TAG, "END mAcceptThread");

}

public void cancel() {

//if (D) Log.d(TAG, "cancel " + this);

try {

mmServerSocket.close();

} catch (IOException e) {

//Log.e(TAG, "close() of server failed", e);

}

}

}

/\*\*

\* This thread runs while attempting to make an outgoing connection

\* with a device. It runs straight through; the connection either

\* succeeds or fails.

\*/

private class ConnectThread extends Thread {

private final BluetoothSocket mmSocket;

private final BluetoothDevice mmDevice;

public ConnectThread(BluetoothDevice device) {

mmDevice = device;

BluetoothSocket tmp = null;

// Get a BluetoothSocket for a connection with the

// given BluetoothDevice

try {

tmp = device.createRfcommSocketToServiceRecord(MY\_UUID);

} catch (IOException e) {

//Log.e(TAG, "create() failed", e);

}

mmSocket = tmp;

}

public void run() {

//Log.i(TAG, "BEGIN mConnectThread");

setName("ConnectThread");

// Always cancel discovery because it will slow down a connection

mAdapter.cancelDiscovery();

// Make a connection to the BluetoothSocket

try {

// This is a blocking call and will only return on a

// successful connection or an exception

mmSocket.connect();

} catch (IOException e) {

connectionFailed();

// Close the socket

try {

mmSocket.close();

} catch (IOException e2) {

//Log.e(TAG, "unable to close() socket during connection failure", e2);

}

// Start the service over to restart listening mode

BluetoothDemoService.this.start();

return;

}

// Reset the ConnectThread because we're done

synchronized (BluetoothDemoService.this) {

mConnectThread = null;

}

// Start the connected thread

connected(mmSocket, mmDevice);

}

public void cancel() {

try {

mmSocket.close();

} catch (IOException e) {

//Log.e(TAG, "close() of connect socket failed", e);

}

}

}

/\*\*

\* This thread runs during a connection with a remote device.

\* It handles all incoming and outgoing transmissions.

\*/

private class ConnectedThread extends Thread {

private final BluetoothSocket mmSocket;

private final InputStream mmInStream;

private final OutputStream mmOutStream;

public ConnectedThread(BluetoothSocket socket) {

//Log.d(TAG, "create ConnectedThread");

mmSocket = socket;

InputStream tmpIn = null;

OutputStream tmpOut = null;

// Get the BluetoothSocket input and output streams

try {

tmpIn = socket.getInputStream();

tmpOut = socket.getOutputStream();

} catch (IOException e) {

//Log.e(TAG, "temp sockets not created", e);

}

mmInStream = tmpIn;

mmOutStream = tmpOut;

}

public void run() {

//Log.i(TAG, "BEGIN mConnectedThread");

byte[] buffer = new byte[1024];

int bytes;

// Keep listening to the InputStream while connected

while (true) {

try {

// Read from the InputStream

bytes = mmInStream.read(buffer);

// Send the obtained bytes to the UI Activity

mHandler.obtainMessage(BluetoothDemo.MESSAGE\_READ, bytes, -1, buffer)

.sendToTarget();

} catch (IOException e) {

//Log.e(TAG, "disconnected", e);

connectionLost();

break;

}

}

}

/\*\*

\* Write to the connected OutStream.

\* @param buffer The bytes to write

\*/

public void write(byte[] buffer) {

try {

mmOutStream.write(buffer);

// Share the sent message back to the UI Activity

mHandler.obtainMessage(BluetoothDemo.MESSAGE\_WRITE, -1, -1, buffer)

.sendToTarget();

} catch (IOException e) {

//Log.e(TAG, "Exception during write", e);

}

}

public void cancel() {

try {

mmSocket.close();

} catch (IOException e) {

//Log.e(TAG, "close() of connect socket failed", e);

}

}

}

}

### ****DeviceListActivity.java****

/\*

\* Copyright (C) 2009 The Android Open Source Project

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

package com.stackcuriosity.android.BluetoothDemo;

import java.util.Set;

import android.app.Activity;

import android.bluetooth.BluetoothAdapter;

import android.bluetooth.BluetoothDevice;

import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;

import android.content.IntentFilter;

import android.os.Bundle;

//import android.util.Log;

import android.view.View;

import android.view.Window;

import android.view.View.OnClickListener;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.ListView;

import android.widget.TextView;

import android.widget.AdapterView.OnItemClickListener;

/\*\*

\* This Activity appears as a dialog. It lists any paired devices and

\* devices detected in the area after discovery. When a device is chosen

\* by the user, the MAC address of the device is sent back to the parent

\* Activity in the result Intent.

\*/

public class DeviceListActivity extends Activity {

// Debugging

//private static final String TAG = "DeviceListActivity";

//private static final boolean D = true;

// Return Intent extra

public static String EXTRA\_DEVICE\_ADDRESS = "device\_address";

// Member fields

private BluetoothAdapter mBtAdapter;

private ArrayAdapter<String> mPairedDevicesArrayAdapter;

private ArrayAdapter<String> mNewDevicesArrayAdapter;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

// layout ekran atamaları

requestWindowFeature(Window.FEATURE\_INDETERMINATE\_PROGRESS);

setContentView(R.layout.device\_list);

//

setResult(Activity.RESULT\_CANCELED);

// Cihaz aramayı gerçekleştirmek için

Button scanButton = (Button) findViewById(R.id.button\_scan);

scanButton.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

doDiscovery();

v.setVisibility(View.GONE);

}

});

// Initialize array adapters. One for already paired devices and

// one for newly discovered devices

mPairedDevicesArrayAdapter = new ArrayAdapter<String>(this, R.layout.device\_name);

mNewDevicesArrayAdapter = new ArrayAdapter<String>(this, R.layout.device\_name);

// arama ve eşleştirilmiş cihazları listelemek için

ListView pairedListView = (ListView) findViewById(R.id.paired\_devices);

pairedListView.setAdapter(mPairedDevicesArrayAdapter);

pairedListView.setOnItemClickListener(mDeviceClickListener);

ListView newDevicesListView = (ListView) findViewById(R.id.new\_devices);

newDevicesListView.setAdapter(mNewDevicesArrayAdapter);

newDevicesListView.setOnItemClickListener(mDeviceClickListener);

// Cihaz bulunduğu zman kaydetmek için

IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION\_FOUND);

this.registerReceiver(mReceiver, filter);

filter = new IntentFilter(BluetoothAdapter.ACTION\_DISCOVERY\_FINISHED);

this.registerReceiver(mReceiver, filter);

mBtAdapter = BluetoothAdapter.getDefaultAdapter();

Set<BluetoothDevice> pairedDevices = mBtAdapter.getBondedDevices();

//Eşleştirilmiş cihazlar varsa, ArrayAdapter'a her birini eklemek için

if (pairedDevices.size() > 0) {

findViewById(R.id.title\_paired\_devices).setVisibility(View.VISIBLE);

for (BluetoothDevice device : pairedDevices) {

mPairedDevicesArrayAdapter.add(device.getName() + "\n" + device.getAddress());

}

} else {

String noDevices = getResources().getText(R.string.none\_paired).toString();

mPairedDevicesArrayAdapter.add(noDevices);

}

}

@Override

protected void onDestroy() {

super.onDestroy();

if (mBtAdapter != null) {

mBtAdapter.cancelDiscovery();

}

this.unregisterReceiver(mReceiver);

}

/\*\*

\* Start device discover with the BluetoothAdapter

\*/

private void doDiscovery() {

//if (D) Log.d(TAG, "doDiscovery()");

setProgressBarIndeterminateVisibility(true);

setTitle(R.string.scanning);

findViewById(R.id.title\_new\_devices).setVisibility(View.VISIBLE);

if (mBtAdapter.isDiscovering()) {

mBtAdapter.cancelDiscovery();

}

mBtAdapter.startDiscovery();

}

//listview de listelediğimiz cihazlar için tıklama dinleyici

private OnItemClickListener mDeviceClickListener = new OnItemClickListener() {

public void onItemClick(AdapterView<?> av, View v, int arg2, long arg3) {

// Cancel discovery because it's costly and we're about to connect

mBtAdapter.cancelDiscovery();

String info = ((TextView) v).getText().toString();

String address = info.substring(info.length() - 17);

Intent intent = new Intent();

intent.putExtra(EXTRA\_DEVICE\_ADDRESS, address);

setResult(Activity.RESULT\_OK, intent);

finish();

}

};

private final BroadcastReceiver mReceiver = new BroadcastReceiver() {

@Override

public void onReceive(Context context, Intent intent) {

String action = intent.getAction();

if (BluetoothDevice.ACTION\_FOUND.equals(action)) {

BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA\_DEVICE);

if (device.getBondState() != BluetoothDevice.BOND\_BONDED) {

mNewDevicesArrayAdapter.add(device.getName() + "\n" + device.getAddress());

}

//arama tamamlandığında Activite başlığını değiştirmek için

} else if (BluetoothAdapter.ACTION\_DISCOVERY\_FINISHED.equals(action)) {

setProgressBarIndeterminateVisibility(false);

setTitle(R.string.select\_device);

if (mNewDevicesArrayAdapter.getCount() == 0) {

String noDevices = getResources().getText(R.string.none\_found).toString();

mNewDevicesArrayAdapter.add(noDevices);

}

}

}

};

}

## ****Ekran arayüz tasarımı:Layout dosyamız:****

### ****custom\_title.xml****

<?xml version="1.0" encoding="utf-8"?>

<!-- Copyright (C) 2009 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center\_vertical"

>

<TextView android:id="@+id/title\_left\_text"

android:layout\_alignParentLeft="true"

android:ellipsize="end"

android:singleLine="true"

style="?android:attr/windowTitleStyle"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:layout\_weight="1"

/>

<TextView android:id="@+id/title\_right\_text"

android:layout\_alignParentRight="true"

android:ellipsize="end"

android:singleLine="true"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:textColor="#fff"

android:layout\_weight="1"

/>

</RelativeLayout>

### ****device\_list.xml****

<?xml version="1.0" encoding="utf-8"?>

<!-- Copyright (C) 2009 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

>

<TextView android:id="@+id/title\_paired\_devices"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="@string/title\_paired\_devices"

android:visibility="gone"

android:background="#666"

android:textColor="#fff"

android:paddingLeft="5dp"

/>

<ListView android:id="@+id/paired\_devices"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:stackFromBottom="true"

android:layout\_weight="1"

/>

<TextView android:id="@+id/title\_new\_devices"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="@string/title\_other\_devices"

android:visibility="gone"

android:background="#666"

android:textColor="#fff"

android:paddingLeft="5dp"

/>

<ListView android:id="@+id/new\_devices"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:stackFromBottom="true"

android:layout\_weight="2"

/>

<Button android:id="@+id/button\_scan"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="@string/button\_scan"

/>

</LinearLayout>

### ****device\_name.xml****

<?xml version="1.0" encoding="utf-8"?>

<!-- Copyright (C) 2009 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<TextView xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textSize="18sp"

android:padding="5dp"

/>

### ****main.xml****

<?xml version="1.0" encoding="utf-8"?>

<!--

Copyright (C) 2009 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="vertical" >

<ListView

android:id="@+id/in"

android:layout\_width="match\_parent"

android:layout\_height="5dp"

android:layout\_alignParentTop="true"

android:stackFromBottom="true"

android:transcriptMode="alwaysScroll" >

</ListView>

<GridLayout

android:id="@+id/gridLayout1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentLeft="true"

android:layout\_below="@+id/in"

android:layout\_marginTop="70dp"

android:columnCount="1" >

</GridLayout>

<Button

android:id="@+id/button4"

android:layout\_width="70dp"

android:layout\_height="70dp"

android:layout\_alignParentTop="true"

android:layout\_toLeftOf="@+id/button2"

android:text=""

android:background="@drawable/direction\_arrow\_blue\_up" />

<Button

android:id="@+id/button1"

android:layout\_width="70dp"

android:layout\_height="70dp"

android:layout\_alignTop="@+id/gridLayout1"

android:layout\_toRightOf="@+id/gridLayout1"

android:text=""

android:background="@drawable/direction\_arrow\_blue\_left" />

<Button

android:id="@+id/button2"

android:layout\_width="70dp"

android:layout\_height="70dp"

android:layout\_alignBaseline="@+id/button1"

android:layout\_alignBottom="@+id/button1"

android:layout\_marginLeft="14dp"

android:layout\_toRightOf="@+id/button3"

android:text=""

android:background="@drawable/direction\_arrow\_blue\_right" />

<Button

android:id="@+id/button3"

android:layout\_width="70dp"

android:layout\_height="70dp"

android:layout\_below="@+id/gridLayout1"

android:layout\_marginLeft="14dp"

android:layout\_toRightOf="@+id/button1"

android:text=""

android:background="@drawable/direction\_arrow\_blue\_down" />

<Button

android:id="@+id/button5"

android:layout\_width="70dp"

android:layout\_height="70dp"

android:layout\_alignParentBottom="true"

android:layout\_alignParentLeft="true"

android:text="DUR" />

<SeekBar

android:id="@+id/seekBar1"

android:layout\_width="350dp"

android:layout\_height="70dp"

android:layout\_alignParentBottom="true"

android:layout\_alignParentRight="true"

android:layout\_marginRight="42dp"

android:layout\_weight="1"

android:rotation="270"

/>

<Button

android:id="@+id/button6"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignBottom="@+id/button4"

android:layout\_alignRight="@+id/seekBar1"

android:text="KORNA" />

</RelativeLayout>

### ****message.xml****

<?xml version="1.0" encoding="utf-8"?>

<!-- Copyright (C) 2009 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<TextView xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:textSize="18sp"

android:padding="5dp"

/>

Son olarak buda values klasörümüzde tanımladığımız isimler

### ****strings.xml****

<?xml version="1.0" encoding="utf-8"?>

<!-- Copyright (C) 2009 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<resources>

<string name="app\_name">Bluetooth Demo</string>

<!-- BluetoothChat -->

<string name="send">Send</string>

<string name="not\_connected">Cihaza Bağlı değilsiniz.</string>

<string name="bt\_not\_enabled\_leaving">Bluetooth açık değil, uygulamadan çıkılıyor...</string>

<string name="title\_connecting">Bağlanıyor...</string>

<string name="title\_connected\_to">Bağlandı: </string>

<string name="title\_not\_connected">Bağlı Değil</string>

<!-- DeviceListActivity -->

<string name="scanning">Cihazlar taranıyor...</string>

<string name="select\_device">Bağlanacak cihazı seçin</string>

<string name="none\_paired">Eşleşilmiş Cihaz Yok</string>

<string name="none\_found">Cihaz Bulunamadı</string>

<string name="title\_paired\_devices">Eşleşilmiş Cihazlar</string>

<string name="title\_other\_devices">Diğer Uygun Cihazlar</string>

<string name="button\_scan">Cihaz Tara</string>

<!-- Options Menu -->

<string name="connect">Cihaza Bağlan</string>

<string name="discoverable">Görünür Yap</string>

</resources>

### ****AndroidManifest.xml****

<?xml version="1.0" encoding="utf-8"?>

<!-- Copyright (C) 2009 The Android Open Source Project

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

-->

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

package="com.stackcuriosity.android.BluetoothDemo"

android:versionName="1.0">

<uses-sdk

android:minSdkVersion="14"

android:targetSdkVersion="21"

/>

<uses-permission android:name="android.permission.BLUETOOTH\_ADMIN" />

<uses-permission android:name="android.permission.BLUETOOTH" />

<application android:label="@string/app\_name"

android:icon="@drawable/app\_icon"

>

<activity android:name=".BluetoothDemo"

android:label="@string/app\_name"

android:configChanges="orientation|keyboardHidden"

android:screenOrientation="landscape"

android:theme="@android:style/Theme"

>

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

<activity android:name=".DeviceListActivity"

android:label="@string/select\_device"

android:theme="@android:style/Theme.Dialog"

android:configChanges="orientation|keyboardHidden" />

</application>

</manifest>

Evet arkadaşlar android arayüzü bukadar.Burada sadece sizin için eksik olarak butonlara hareketleri göstermesi bakımından sağ-sol-ileri-geri yön tuşları şeklinde bir .png resim koymuştum.Atadığımız bu görsel resimler yok.Bunları siz kendi zevkinize göre ayarlarsınız.İstediğiniz şekilde butonlara resim ataya bilirsiniz.

Arduino ya geçecek olursak

## Arduino.cc

#include <SoftwareSerial.h>// import the serial library

//Genotronex adında SoftwareSerial özel komutu kullanarak yazılımsal bir seri port haberleşme ayarlanıyor.

//RX pini 10'uncu TX pinide 11 pin olarak ayarlıyorz

SoftwareSerial Genotronex(10,11); // RX, TX

//Değişkenlerinimizi tanımladık

//Arkadaşlar arduino uno gibi düşük ram'a sahip cihazlarda programlama

//yaparken en az ram kullanımı önemli olduğunda bellekte yer tutacak değişkenlerimizi

//asgari boyutta tanımlamalıyız.

boolean ileri=false;

boolean geri=false;

boolean sag=false;

boolean sol=false;

boolean dur=false;

boolean korna=false;

boolean sonDurum=false;

boolean aDur=false;

int enson;

int hiz=0;

int dizi[30];

int i=0;

int x=0;

String stringOne;

void setup() {

Serial.begin(9600);

Genotronex.begin(9600);

Genotronex.println("zafer");

//pinlerin durumlarını çıkış olarak atadık

pinMode(5,OUTPUT);

pinMode(6,OUTPUT);

pinMode(8,OUTPUT);

pinMode(3,OUTPUT);

pinMode(4,OUTPUT);

pinMode(7,OUTPUT);

pinMode(12, OUTPUT);

//pinlerimizin hepsini başlangıcta lojic 0 olarak ayarladık

digitalWrite(8,0);

digitalWrite(3,0);

digitalWrite(4,0);

digitalWrite(7,0);

digitalWrite(12,0);

}

void loop() {

i=0;

stringOne = "";

//Seri porttan okudumuz dataları diziye attık

//-48 yapmamızın sebebi ASCII kodları decimal'e çevirmek için kullandık

while (Genotronex.available()){

dizi[i]=Genotronex.read()-48;

i++;

}

//Dizinin her bir elemanını string bir değişkenimizin içerisinde yan yana birleştirdik.

for(int j=0;j<i && i>0;j++){

stringOne += dizi[j];

}

while(!Genotronex.available()){

Serial.println(stringOne);

//Telefonumuz üzerinden gelen dataları filtrelemek için her datanın öncesinde sırayla 8 ve 0 rakamlarını gönderdik.

//filtrelememizdeki amaç parazit olarak gelen tek haneli rakamları engellemek.

//Devamındaki komutlar gelen datalara göre rc-arabaya yön verme komutlarıdır.

if(stringOne=="803")

{

ileri= true;

sag=false;

sol=false;

geri=false;

dur=false;

aDur=false;

hiz=0;

}

else if(stringOne=="801"){

ileri= false;

sag=true;

sol=false;

geri=false;

dur=false;

aDur=false;

}

else if(stringOne=="802"){

ileri= false;

sag=false;

sol=true;

geri=false;

dur=false;

aDur=false;

}

else if(stringOne=="804"){

ileri= false;

sag=false;

sol=false;

geri=true;

dur=false;

aDur=false;

}

else if(stringOne=="805"){

ileri= false;

sag=false;

sol=false;

geri=false;

dur=true;

aDur=false;

}

else if(stringOne=="808"){

ileri= false;

sag=false;

sol=false;

geri=false;

dur=false;

aDur=true;

}

else if(stringOne=="806"){

digitalWrite(12, !digitalRead(12));

delay(20);

}

else if(stringOne.toInt()<=255)

{

hiz=stringOne.toInt();

Serial.println(hiz);

}

if(ileri==true&&geri==false&&sag==false&&sol==false&&dur==false){

digitalWrite(8,1);

digitalWrite(3,0);

analogWrite(5,hiz);

digitalWrite(4,1);

digitalWrite(7,0);

analogWrite(6,hiz);

enson=0;

sonDurum=true;

}

else if(ileri==false&&geri==true&&sag==false&&sol==false&&dur==false){

digitalWrite(8,0);

digitalWrite(3,1);

analogWrite(5,hiz);

digitalWrite(4,0);

digitalWrite(7,1);

analogWrite(6,hiz);

enson=1;

sonDurum=true;

}

else if(ileri==false&&geri==false&&sag==true&&sol==false&&dur==false){

digitalWrite(8,1);

digitalWrite(3,0);

analogWrite(5,hiz);

digitalWrite(4,0);

digitalWrite(7,1);

analogWrite(6,hiz);

enson=2;

}

else if(ileri==false&&geri==false&&sag==false&&sol==true&&dur==false){

digitalWrite(8,0);

digitalWrite(3,1);

analogWrite(5,hiz);

digitalWrite(4,1);

digitalWrite(7,0);

analogWrite(6,hiz);

enson=3;

}

else if(aDur==true){

if(enson==0&&sonDurum==true){

digitalWrite(8,0);

digitalWrite(3,1);

analogWrite(5,200);

digitalWrite(4,0);

digitalWrite(7,1);

analogWrite(6,200);

delay(50);

sonDurum=false;

}

else if(enson==1&&sonDurum==true){

digitalWrite(8,1);

digitalWrite(3,0);

analogWrite(5,200);

digitalWrite(4,1);

digitalWrite(7,0);

analogWrite(6,200);

delay(50);

sonDurum=false;

}

}

else if(dur==true){

Serial.println("durdaaaa");

digitalWrite(8,0);

digitalWrite(3,0);

analogWrite(5,0);

digitalWrite(4,0);

digitalWrite(7,0);

analogWrite(6,0);

}

}

}