

# Arduino Bluetoothシールド/ Arduino Bluetoothライブラリ/ Android Bluetoothライブラリ 説明書

## Arduino Bluetoothシールド

使用Bluetoothモジュール:FB155BC

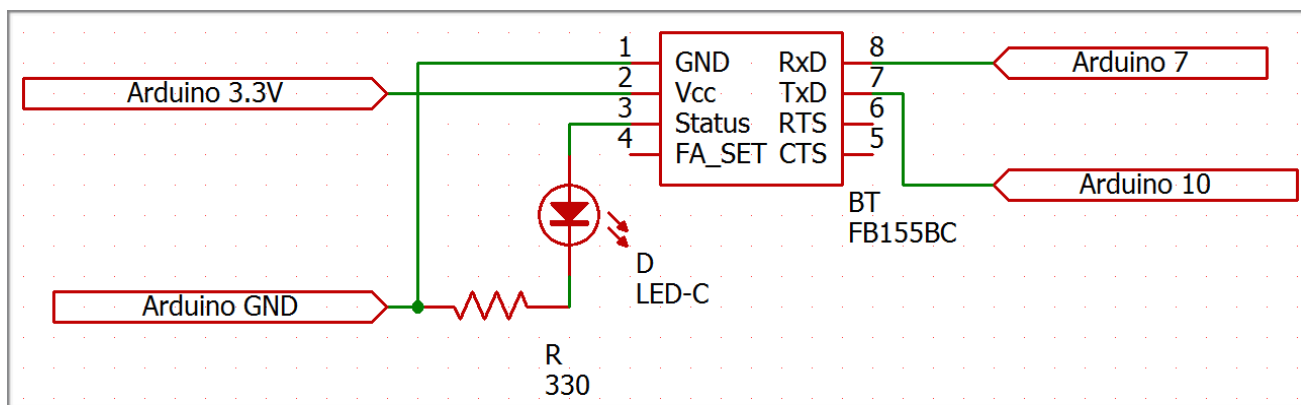


図 1 回路図

Arduino Bluetoothライブラリをそのまま使用するのであれば、図 1 の通りのピン配置で設計、制作する。ライブラリを改変する、または使用しないのであれば、ピン配置は適当に変えることもできる。

# Arduino Bluetoothライブラリ 利用方法

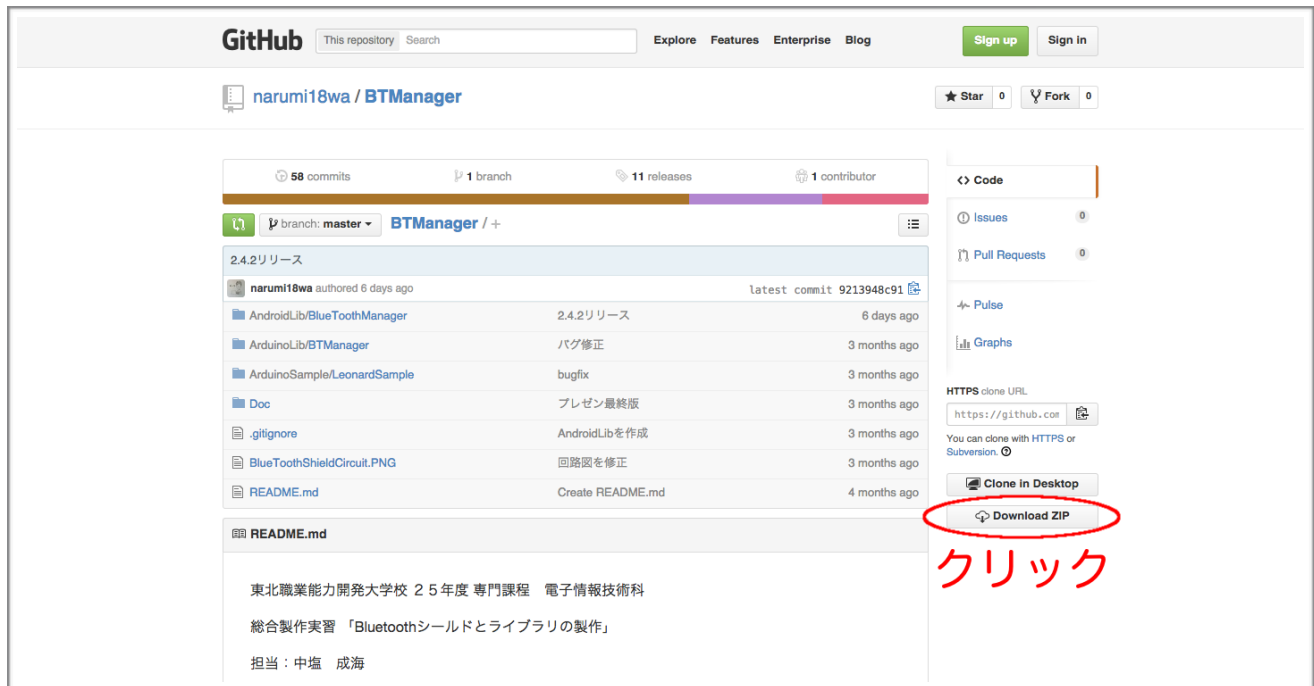


図2 全てダウンロード

図2に従い、全てのファイルをZIPでダウンロードします。

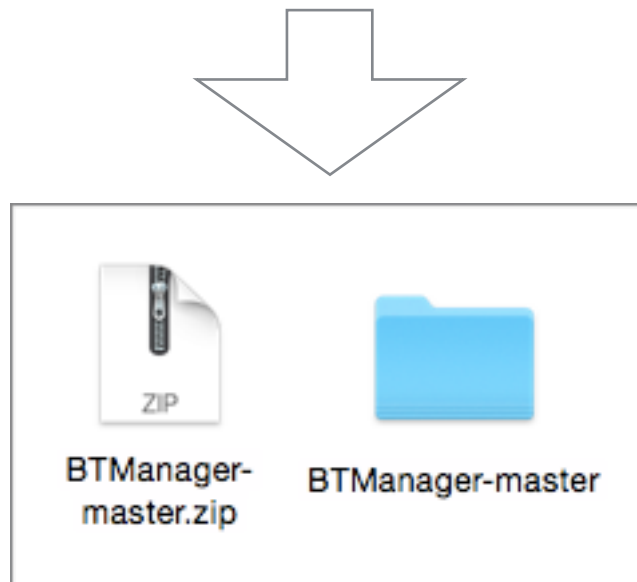


図3 展開してください

図3のようにわかり易い場所に展開してください。

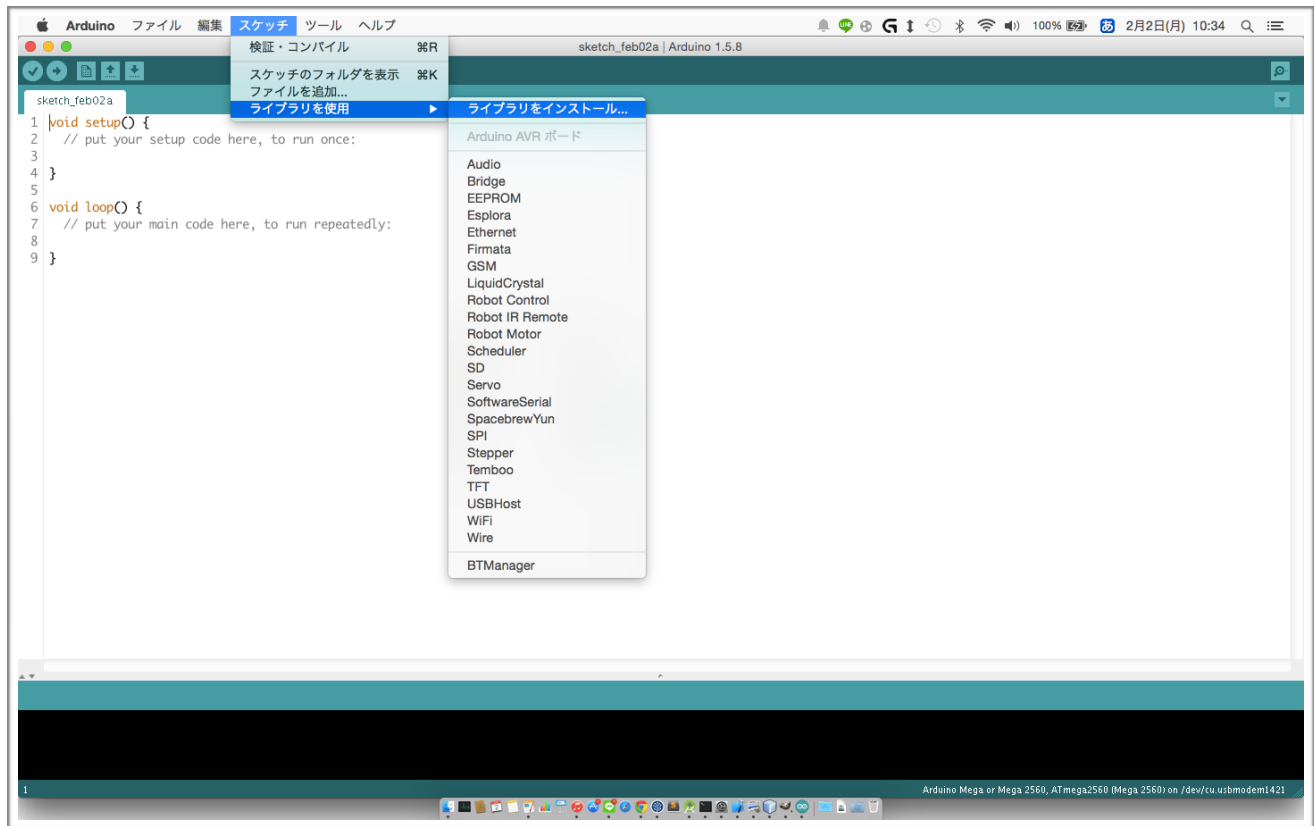
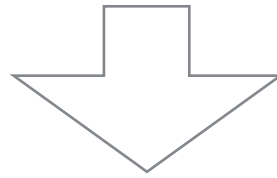


図4 ライブラリをインストールをクリック

図4のように、ArduinoIDEを起動し、ツールバーからライブラリをインストール…をクリックしてください。

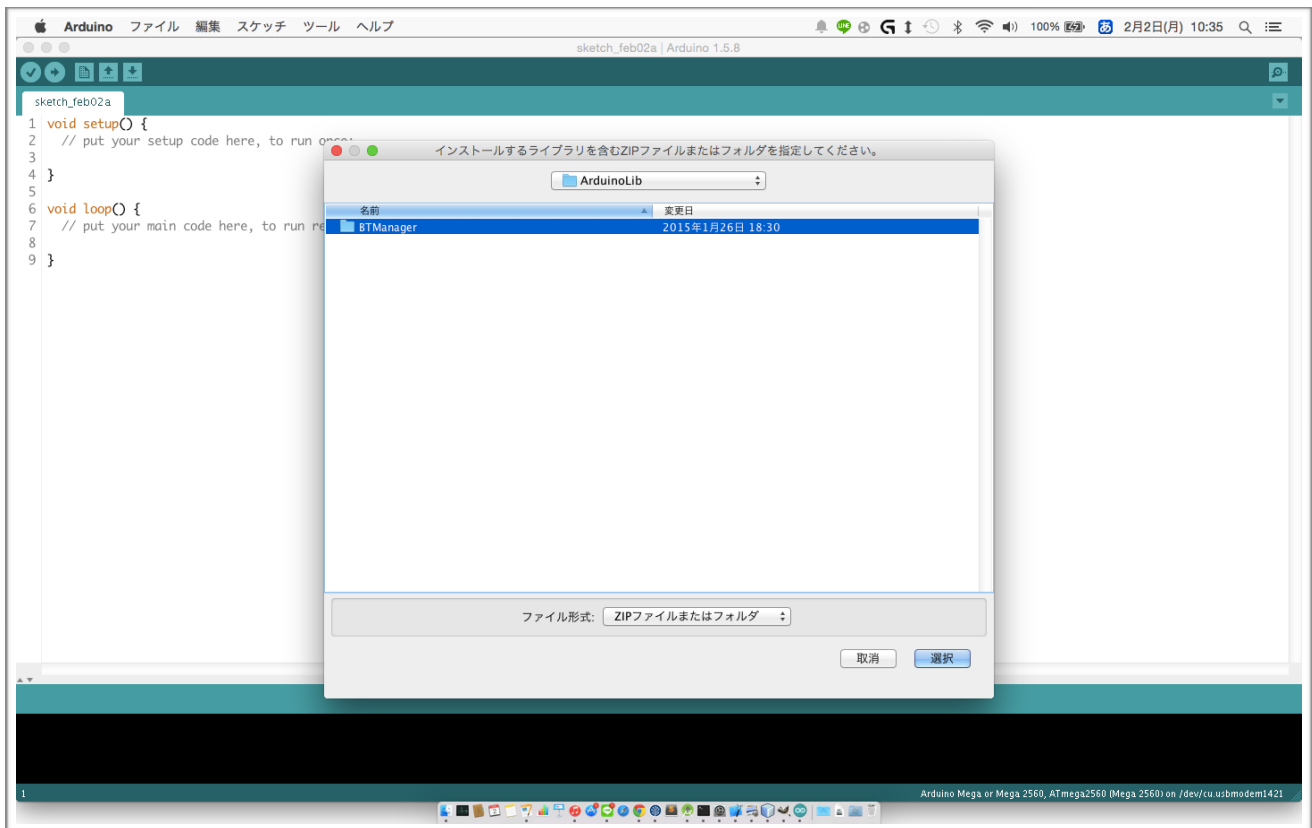
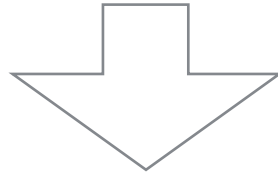


図5 BTManagerを開く

図5のように展開したファイルの中の(展開したフォルダ)/BTManager-master/ArduinoLib/BTManagerを開く

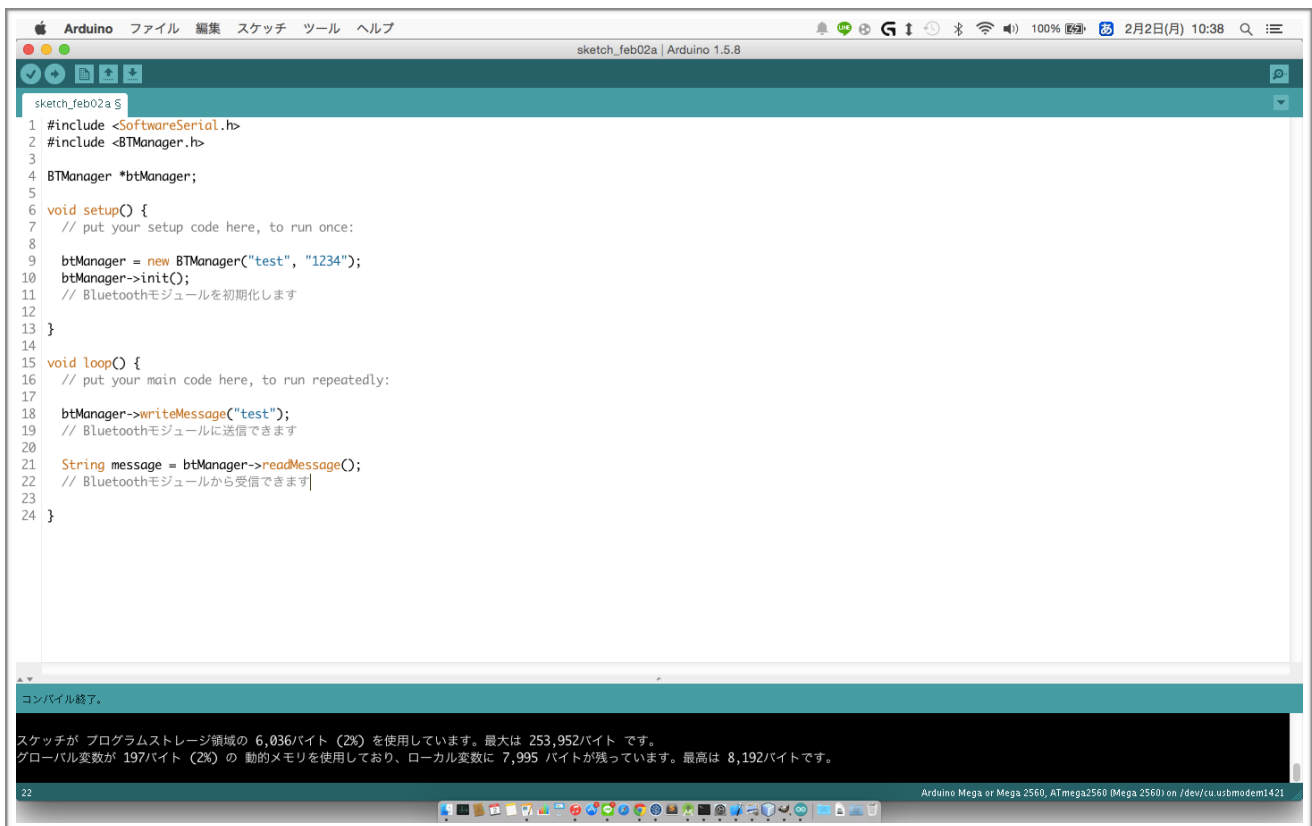
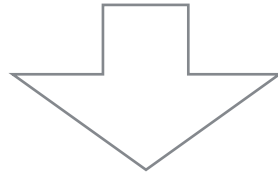


図 6 必要最低限のコード

図 6 を参考にBTManagerを動くように設定する。loop内の実装は例であって、好きに実装して良い。

## 仕様

Sketchからアクセスできるメソッドは以下の通りの仕様となっている

BTManager(String btName, String btPassWord);

コンストラクタ。btNameは”他の端末から見える名前”、btPassWordは”パスワード”をそれぞれ設定できる。

void init();

Bluetoothモジュールをコンストラクタの設定で初期化する。

String readMessage();

Bluetoothモジュールに送信されているメッセージを読む。

void writeMessage(String message);

Bluetoothモジュールからメッセージを送信する。

# Android Bluetoothライブラリ 利用方法

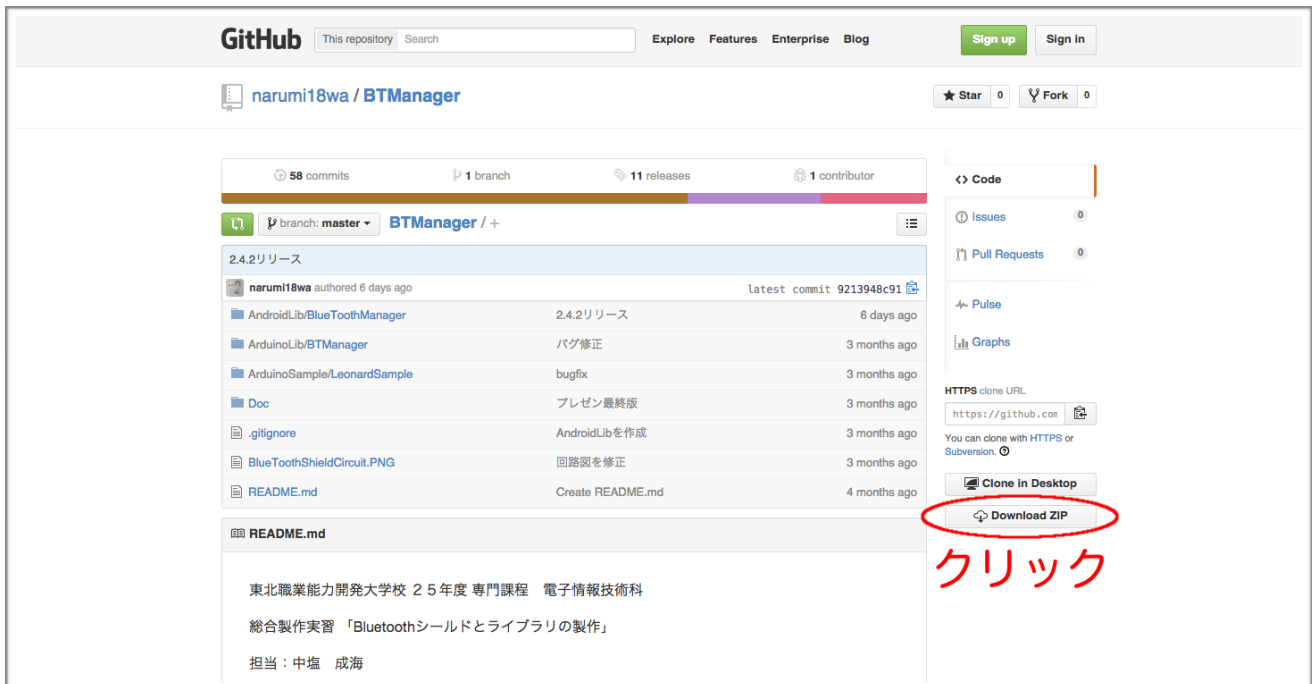


図7 全てダウンロード

図7 に従い、全てのファイルをZIPでダウンロードします。

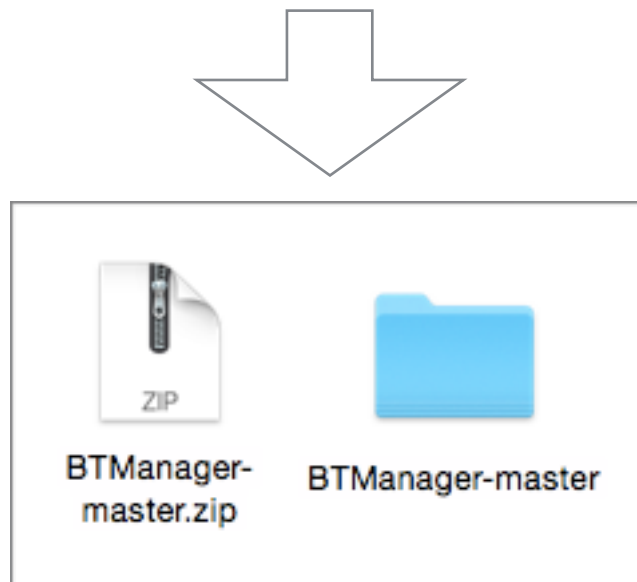


図8 展開してください

図8 のようにわかり易い場所に展開してください。

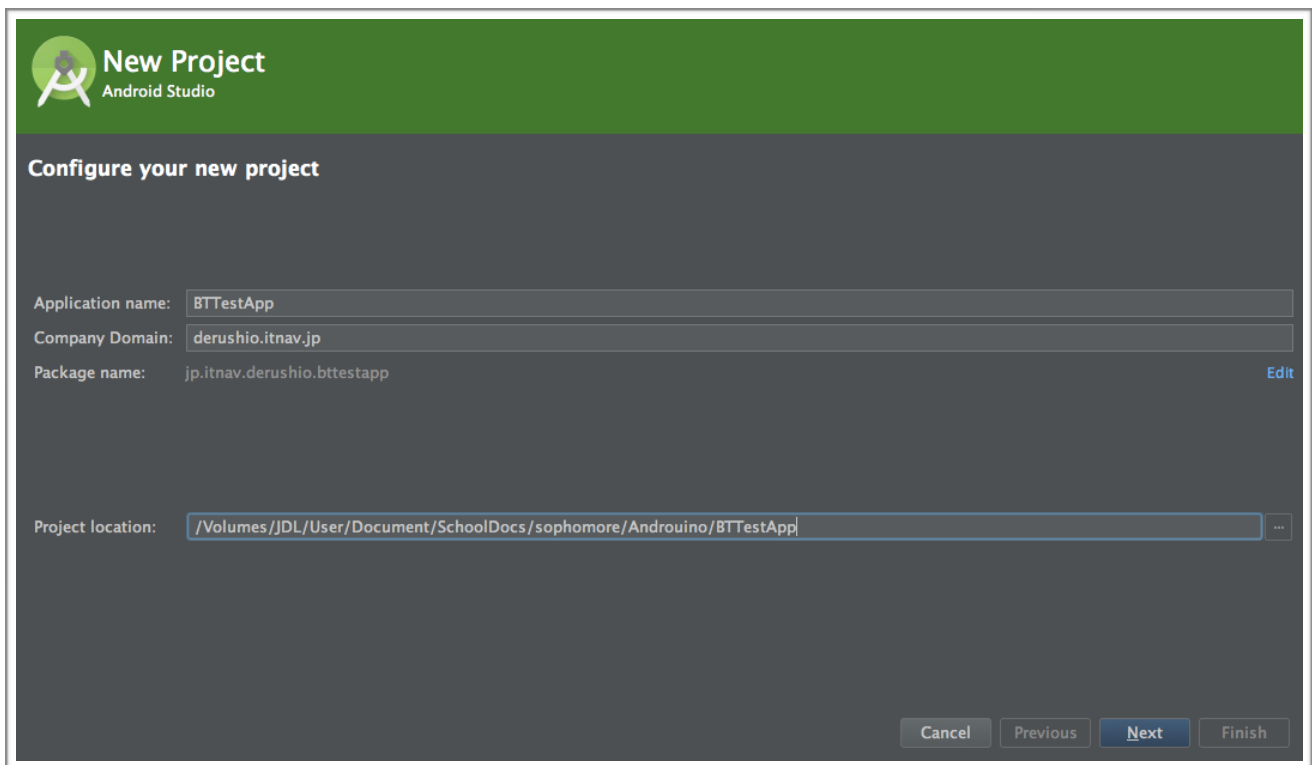
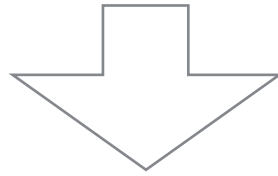


図9 プロジェクトを作成してください

あなたの作りたいプロジェクトを作成してください。まず、テスト用のプロジェクトを作成しても良いかもしれません。

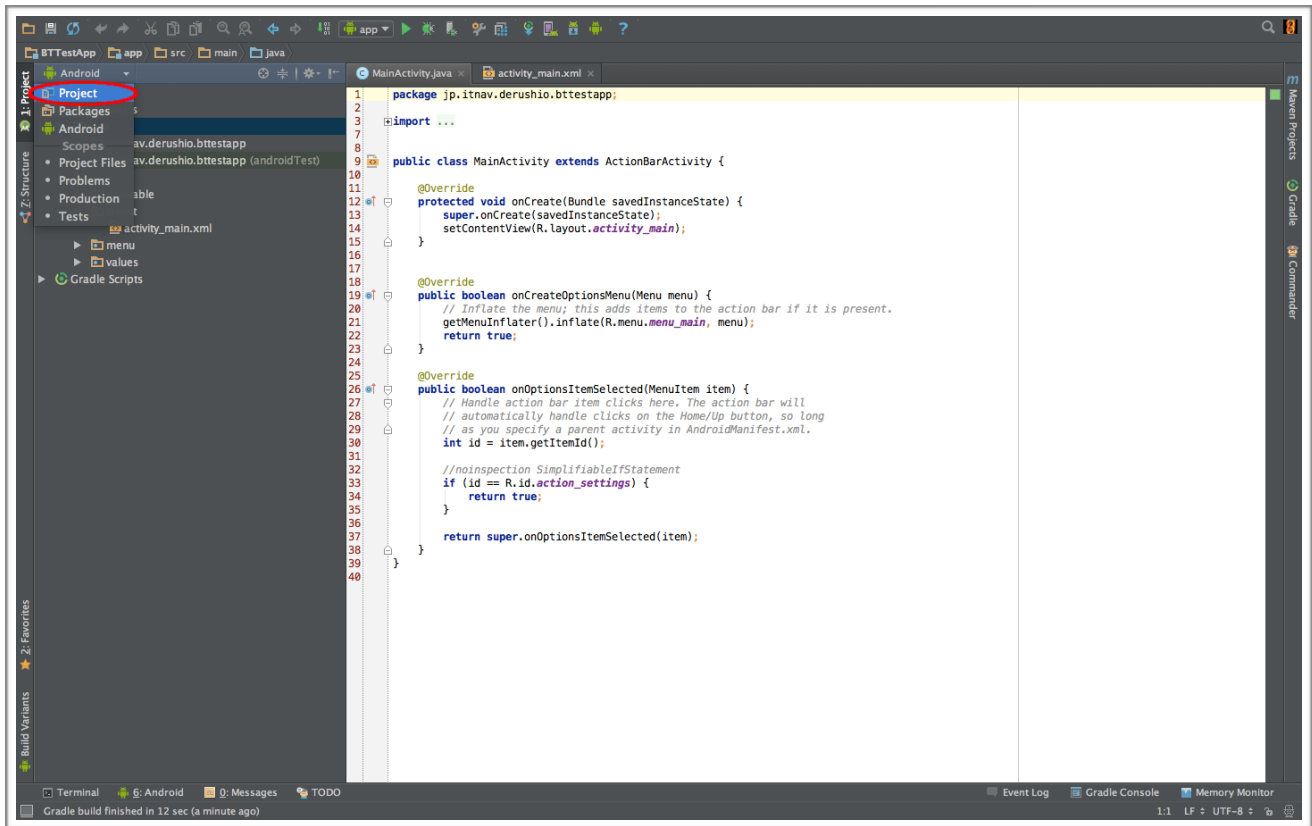
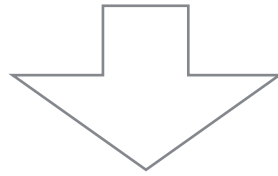
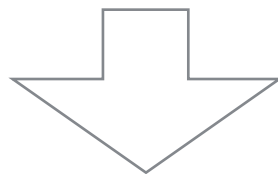


図 10 フォルダツリーを表示

フォルダツリーを表示させるために、**左下**のメニュー（□マーク）から Project を選び、プルダウンメニューから、Project を選択します。すでに Project になっている場合はそのままで大丈夫です。



フォルダ	書類	フォルダ	フォルダ	フォルダ	デベロッパ
BTManager-master	README.md	BlueToothManager	app	release	BlueToothManager-1.0.jar
その他			gradle	src	BlueToothManager-1.1.jar
BTManager-master.zip	フォルダ		デベロッパ	デベロッパ	BlueToothManager-2.2.jar
	AndroidLib		BlueToothManager.iml	app.iml	BlueToothManager-2.3.1.jar
	ArduinoLib		build.gradle	build.gradle	BlueToothManager-2.3.jar
	ArduinoSample		gradlew	proguard-rules.pro	BlueToothManager-2.4.1.jar
	Doc		settings.gradle		BlueToothManager-2.4.2.jar
	イメージ		その他		BlueToothManager-2.4.jar
	BlueToothShieldCircuit.PNG		gradle.properties		
			gradlew.bat		

図 11 jar ファイルをコピー

図 5 のように展開したファイルの中の(展開したフォルダ)/BTManager-master/AndroidLib/BlueToothManager/app/release/BlueToothManager-2.4.2.jar（最新版）をコピーしてください。



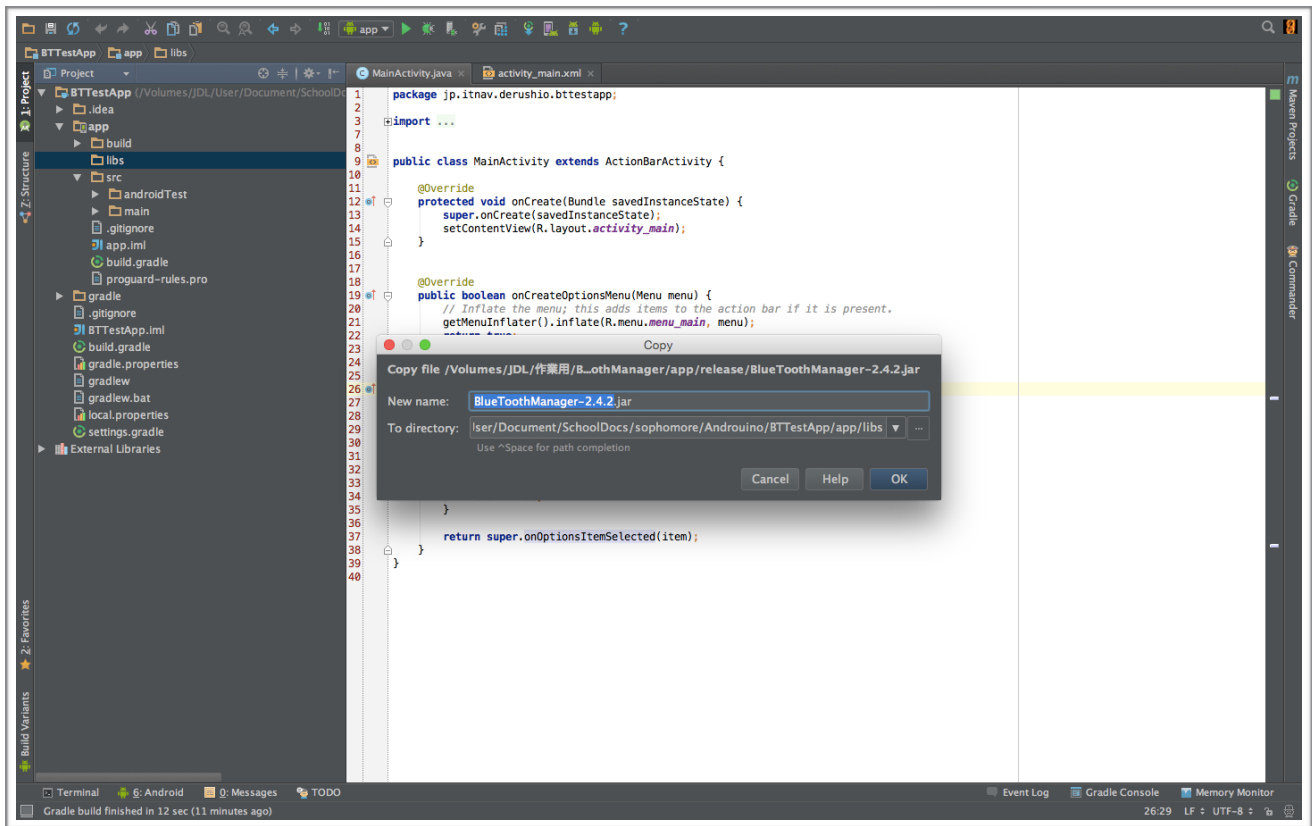
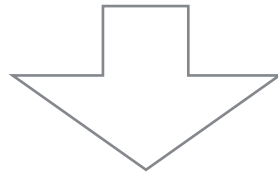


図 1 2 ペーストしましょう

Android Studioの左下メニュー → ProjectからProject内のlibフォルダ（図 1 2 参照）を選択し、“Ctrl + v”でペーストしましょう。

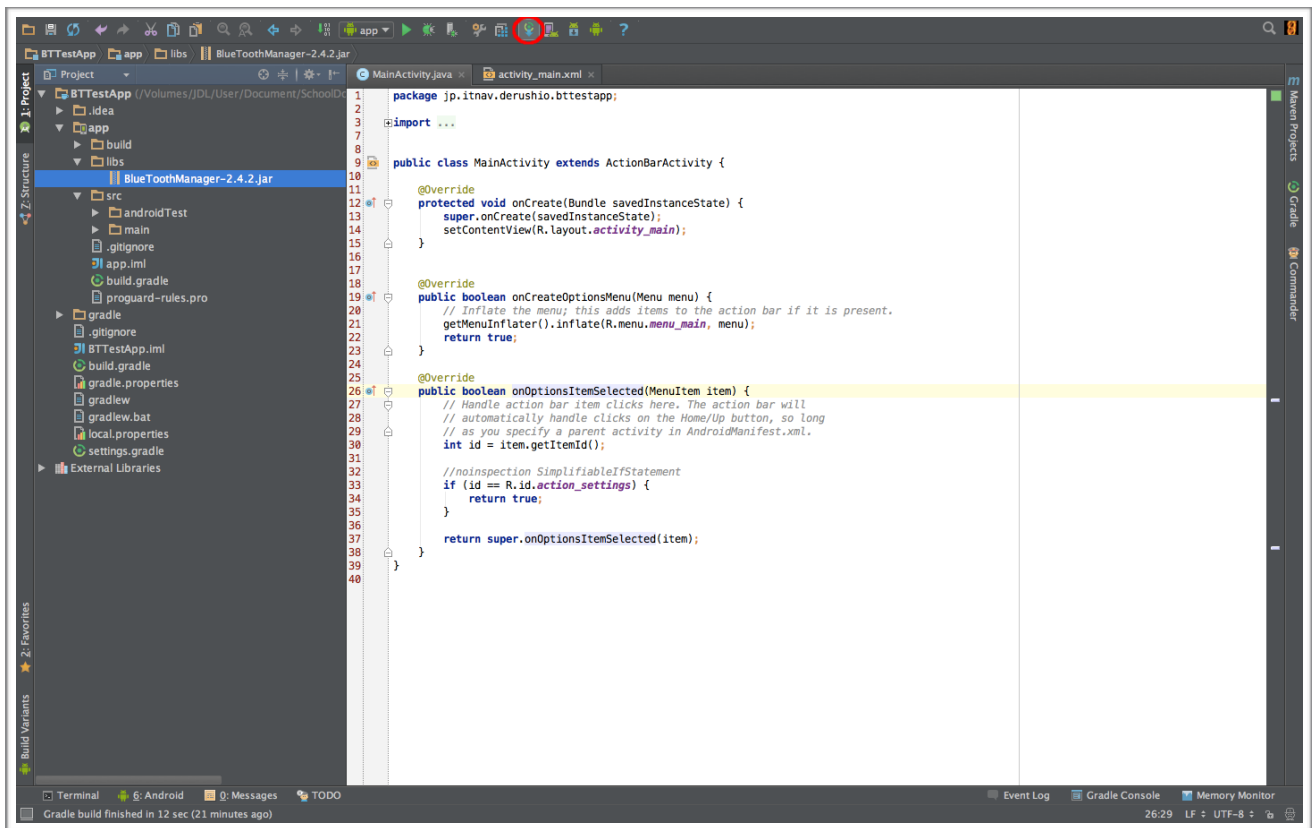
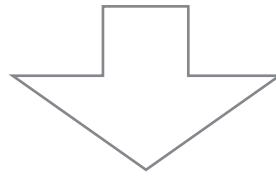


図 1 3 リビルドしよう

図 1 3 のようにクロームと ↓ が組み合わさったようなアイコンをクリックしてリビルドしましょう。

以上でライブラリを使う準備は整いました。しかし、実装方法については慣れていなければわからない部分が多いでしょう。そこで、**サンプルプログラム(BluetoothManagerSample)**をzip内に入れておいたので、それを参考にしながら実装していきましょう。

## MainActivity.java(何かにコピーして見ることを推奨)

```
public class MainActivity extends BluetoothManagedActivity {
    // BluetoothManagedActivityを継承している

    private LinearLayout pairedDeviceList;
    // ペアリングしたBluetoothDeviceを表示するLinearLayout

    private TextView textViewReadMessage;
    // 受信したメッセージを表示するTextView
    private EditText editTextWriteMessage;
    // 送信するメッセージを設定するEditText

    private TimerHandler timerHandler;
    private boolean isTimerHandlerStarted = false;
    // タイマー

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        timerHandler = new TimerHandler();

        pairedDeviceList = (LinearLayout) findViewById(R.id.pairedDeviceList);
        textViewReadMessage = (TextView) findViewById(R.id.textViewReadMessage);
        editTextWriteMessage = (EditText) findViewById(R.id.editTextWriteMessage);
        // 画面された画面と変数を関連付け

        Set<BluetoothDevice> pairedDevices = getPairedDevices();
        // ペアリングされているデバイスを取得
        for (final BluetoothDevice bluetoothDevice : pairedDevices) {
            Button button = new Button(this);
            // ボタンをインスタンス
            button.setText(bluetoothDevice.getName() + "\n" + bluetoothDevice.getAddress());
            button.setLayoutParams(new LinearLayout.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT, ViewGroup.LayoutParams.WRAP_CONTENT));
            button.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    setTargetDevice(bluetoothDevice);
                    connectDevice();
                    readMessageStart(100);
                    timerHandler.start(1000);
                    isTimerHandlerStarted = true;
                    // クリックされたら、クリックしたボタンに対応するBluetoothDeviceに接続、同時にメッセージの受信をスタートする。
                }
            });
            pairedDeviceList.addView(button);
        }

        @Override
        protected void onPause() {
            super.onPause();
            if (isTimerHandlerStarted) {
                timerHandler.timerStop();
            }
        }
        // 画面が止まった時にタイマーもストップ

        @Override
        protected void onResume() {
            super.onResume();
            if (isTimerHandlerStarted) {
                timerHandler.timerStart();
            }
        }
        // 画面が始まった時にタイマーをスタート

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
            // Inflate the menu; this adds items to the action bar if it is present.
            getMenuInflater().inflate(R.menu.menu_main, menu);
            return true;
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
            // Handle action bar item clicks here. The action bar will
            // automatically handle clicks on the Home/Up button, so long
            // as you specify a parent activity in AndroidManifest.xml.
            int id = item.getItemId();

            //noinspection SimplifiableIfStatement
            if (id == R.id.action_settings) {
                return true;
            }

            return super.onOptionsItemSelected(item);
        }

        public void writeMessage(View v) {
            editTextWriteMessage.clearFocus();
            writeMessage(editTextWriteMessage.getText().toString());
            // メッセージを送信する
        }

        private class TimerHandler extends Handler {
            // タイマーを定義するclass
            private boolean isTick = false;
            // タイマーが動いているか
            private long delayMillisec = 1000;
            // 待ち時間の長さ

            @Override
            public void handleMessage(Message msg) {
                super.handleMessage(msg);

                if (isDeviceConnected()) {
                    textViewReadMessage.setText("受信：" + readMessage().get(0));
                    // メッセージを受信する。
                    // メッセージはArrayListの受信時間順に帰ってくるので、get(0)で最新データが取れる。
                }

                if (isTick) {
                    sleep();
                }
            }

            public void timerStart(long delayMillisec) {
                this.delayMillisec = delayMillisec;
                isTick = true;
                sleep();
            }

            public void timerStart() {
                timerStart(delayMillisec);
            }
            // リスタート用のOverLoad

            public void timerStop() {
                isTick = false;
            }

            private void sleep() {
                removeMessages(0);
                sendMessageDelayed(obtainMessage(0), delayMillisec);
            }
        }
    }
}
```

## 上記コードで継承している自作クラス

### BluetoothManagedActivity要点

- protected void onCreate(android.os.Bundle savedInstanceState);
  - Bluetoothとタイマーを初期化します
- protected void onResume();
  - Bluetoothデバイスと接続しなおします
- protected void onPause();
  - Bluetoothデバイスと一旦切断します
- protected Set<BluetoothDevice> getParedDevices();
  - ペアリングされているBluetoothデバイス郡を取得します
- protected void setTargetDevice(BluetoothDevice targetDevice);
  - 接続するBluetoothデバイスをターゲットします
- protected BluetoothDevice getTargetDevice();
  - ターゲットされているBluetoothデバイスを取得します
- protected boolean isDeviceConnected();
  - Bluetoothデバイスに今つながっているか取得します
- protected void connectDevice();
  - Bluetoothデバイスに接続します
- protected void disconnectDevices();
  - Bluetoothデバイスから切断します
- protected void readMessageStart(long delayMillisec);
  - delayMillisecの秒数ごとにBluetoothからメッセージを受信し、メールボックスに蓄積します。
- protected void readMessageStop();
  - Bluetoothからメッセージを受信することを取りやめます

- `protected void writeMessage(String message);`
  - Bluetoothからメッセージを送信します
- `protected ArrayList<String> readMessage();`
  - `readMessageStart(long delayMilliSec)`で蓄積されているメールボックスを取得します。0が一番新しく、100件まで蓄積されます。
- `protected void setOnConnect(Handler onConnect);`
  - 接続時のハンドラを設定します。
  - `message.what`について
    - 0は成功
    - -1はSocketが見つからない
    - -2はBluetoothデバイスが見つからない
    - -3はそんなデバイスとペアリングしていない
- `protected void setOnDisconnect(Handler onDisconnect);`
  - 切断時のハンドラを設定します
  - `message.what`について
    - 0は成功
    - -1はBluetoothデバイスが見つからない

activity\_main.xml(何かにコピーして見ることを推奨)