



# 目次

<b>1. 初めての起動</b>	<b>3</b>
(1) アプリケーション起動	3
(2) Arduino IDE 自動検出	4
(3) 自動検出できなかった場合 (開始できない場合)	5
(4) チュートリアル	6
(5) 画面構成	8
(6) RGB形式について	9
<b>2. ダークテーマを作ってみよう</b>	<b>10</b>
(1) 色の入れ替え	10
(2) 基本16色から選択	13
(3) カラーピッカー (HSL形式) による色選択	15
(4) 直接指定 (RGB形式) による色選択	16
(5) コピー & 貼り付けによる設定	17
(6) 候補 (他のキー) から選択して設定	19
(7) 履歴 (最近使った色) から選択	21
(8) Arduino IDE 更新 (テーマファイル書き込み)	22
(9) その他の色の変更	24
<b>3. テーマファイル各種設定</b>	<b>25</b>
(1) フォントの設定「人気の高いフォント」	25
(2) フォントの設定「一覧より選択」	26
(3) TABキーの動作「半角スペースで埋めるか」設定	27
(4) インデントの幅 (スペース数) の設定	27
(5) その他のキー設定	28
(6) 外観のキーを変更する場合	28
<b>4. その他の機能</b>	<b>29</b>
(1) 編集データのファイル保存と読み取り	29
(2) 色の置換	30
(3) 「元に戻す」と初期化	31
(4) 全体の初期化	31
(5) その他の機能	32
(6) アップデート	33
(7) アンインストール	35

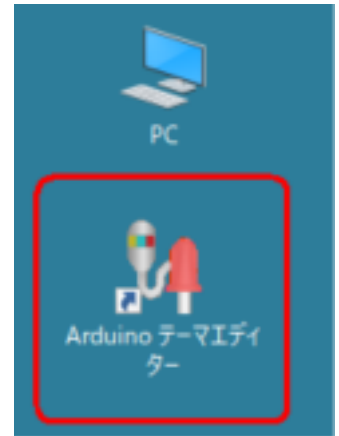
# 1. 初めての起動

アプリケーションを初めて起動した場合の初期設定を行います。

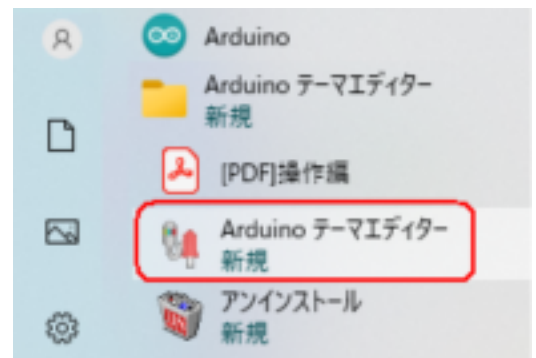
## (1) アプリケーション起動

デスクトップ上に作成されたアイコンをクリック  
(またはダブルクリック) します。

ショートカットを作成していない場合は  
インストール先フォルダから実行ファイル  
「ArduiThemeEditor.exe」を実行します。



または、スタートメニューから本ソフトウェアを  
選択します。



## (2) Arduino IDE 自動検出

アプリケーションが起動します。  
同時に、Arduino IDE のインストール先を自動検出します。



テーマファイルは3種類あります。

- preferences.txt (ユーザーのローカルデータに自動設定)
- theme.txt (色やフォントを設定/インストール先フォルダ内)
- default.xml (一部の色を設定/インストール先フォルダ内)

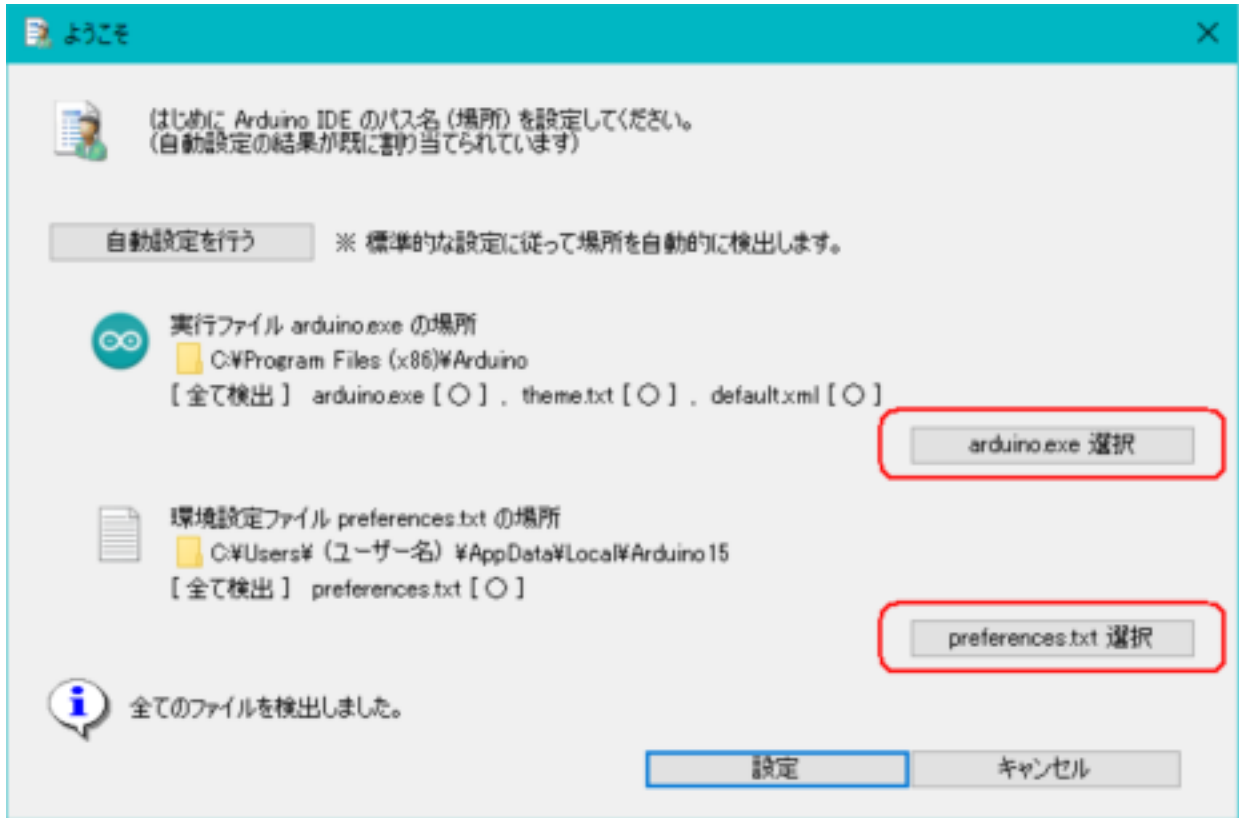
Arduino IDE の実行ファイルとテーマファイルが全てが検出された場合、ボタン①「開始」を押して(4)へ進みます。

テーマファイルが検出されなかった場合、フォルダの場所を指定する必要があります。(Arduino IDE のインストール先を変更している場合、検出できない場合があります) ボタン②「Arduino 設定画面を起動」を押して(3)へ進みます。



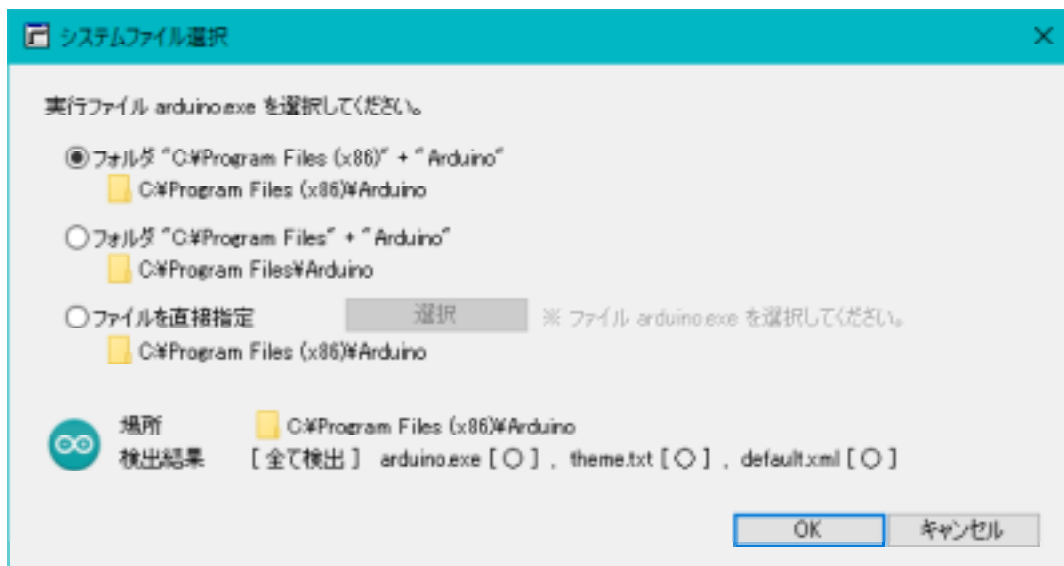
### (3) 自動検出できなかった場合（開始できない場合）

以下の設定画面（環境設定）が表示されます。



検出されていないファイルについて、ボタン「選択」を押します。

フォルダ選択画面が表示されます。

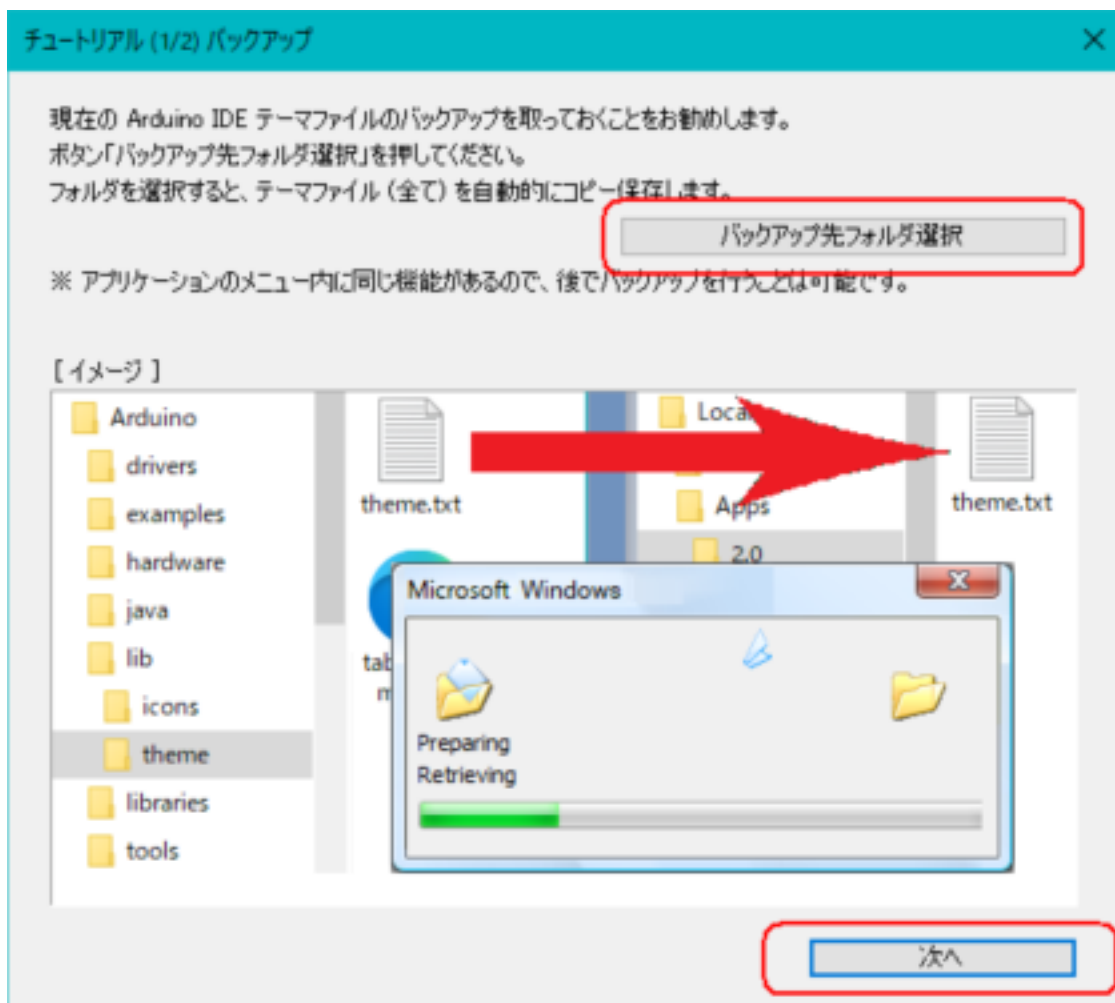


フォルダの候補が表示されます。  
検出結果を見ながらフォルダを選択してボタン「OK」を押します。

全てのテーマファイルが検出されたらボタン「設定」を押して（2）に戻ります。

## (4) チュートリアル

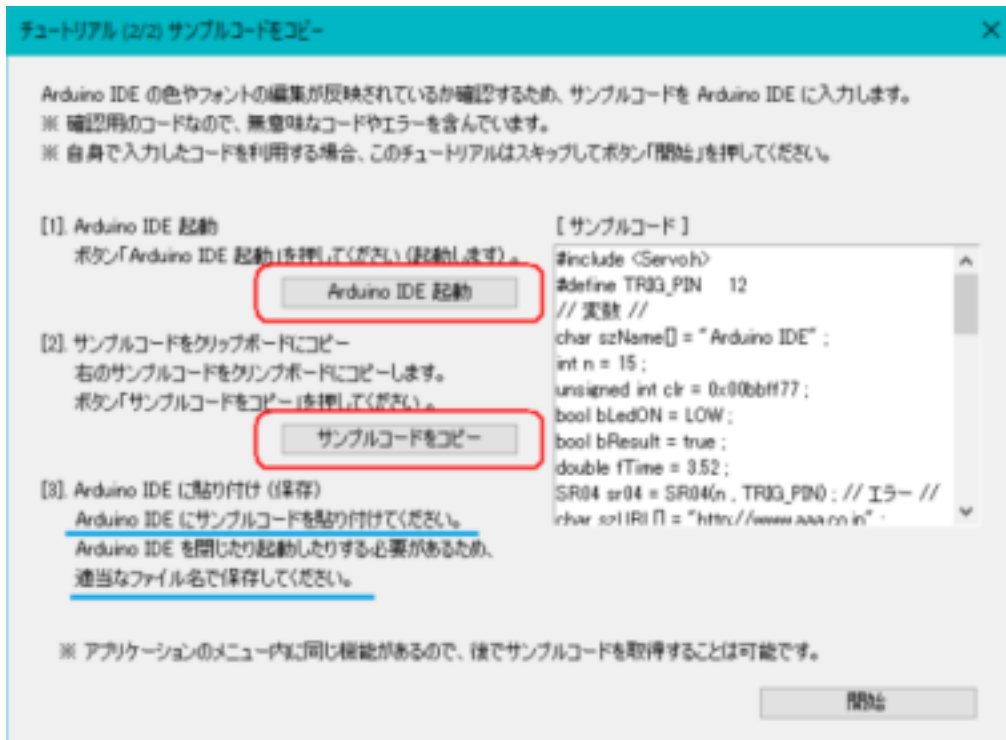
チュートリアル（1 / 2）では、現在のテーマファイルのバックアップを取ります。  
（不要な場合はスキップすることができます）



バックアップを実行する場合は、ボタン「バックアップ先フォルダ選択」を押してフォルダを選択します。  
（選択すると自動でバックアップが行われます）

ボタン「次へ」を押します。

チュートリアル（2 / 2）では、サンプルコードを Arduino IDE に貼り付けます。  
（不要な場合はスキップすることができます）

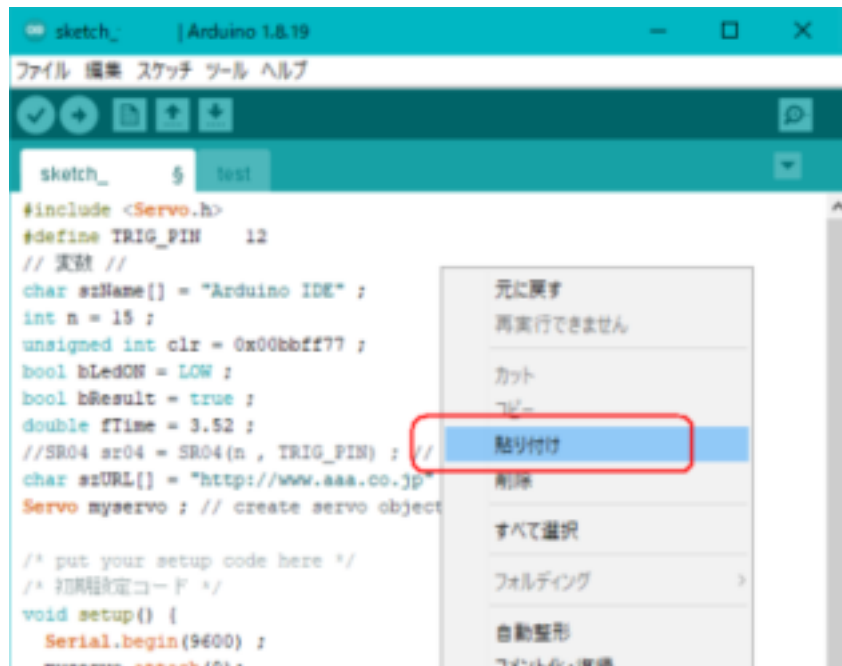


以下の手順[1]～[3]に従います。

[1] ボタン「Arduino IDE 起動」を押します。

[2] ボタン「サンプルコードをコピー」を押します。

[3] 起動した Arduino IDE のエディター領域で右クリックして「貼り付け」を選択します。

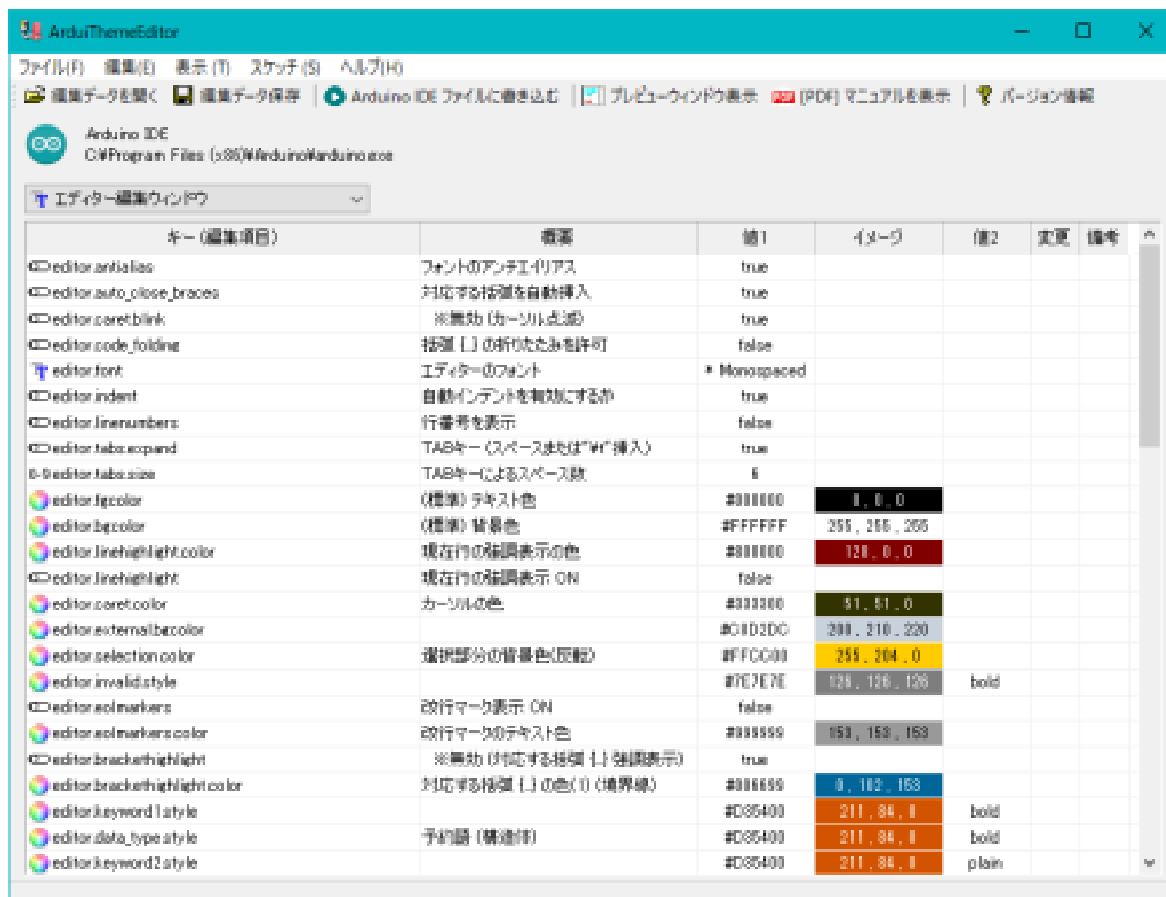


サンプルコードが貼り付けられたら適当なファイル名でスケッチファイルを保存します。  
Arduino IDE は一旦終了します。

ボタン「開始」を押します。

## (5) 画面構成

編集画面が表示されます。



プレビュー画面が別ウィンドウで表示されます。



## (6) RGB形式について

色は「光の三原色」である**Red(赤)**、**Green(緑)**、**Blue(青)**で表現されます。  
アプリケーションでは以下の2通りの方法で表示されます。

RGB(**r** , **g** , **b**)

r , g , b は 0 ~ 2 5 5 の範囲で10進数で表記

#**rrggbb**

r , g , b は 0 ~ Fの範囲で16進数で表記

例1 白

RGB(255, 255, 255) #FFFFFF (または#FFF)

例2 黒

RGB(0, 0, 0) #000000 (または#000)

例3



RGB(255, 128, 64) #FF8040

例4



RGB(0, 128, 255) #0080FF

以上の流れでアプリケーション起動から初期設定まで行います。

## 2. ダークテーマを作ってみよう

ダークテーマの作成を例に色やフォント等の編集方法を解説します。  
以下のようにテキスト色を白（#FFFFFF）、背景色を黒（#000000）に変更します。



```
プレビューウィンドウ
操作(S)
マイコンボードに書き込む
sketch_servo sample_0 sample_1
#include <Servo.h>
#define TRIG_PIN 12
// 変数 //
char szName[] = "Arduino IDE";
int n = 15;
unsigned int clr = 0x00bbff77;
bool bLedON = LOW;
bool bResult = true;
double fTime = 3.52;
SR04 sr04 = SR04(n, TRIG_PIN); // エラー //
char szURL[] = "http://www.aaa.co.jp";
Servo myservo; // create servo object to control a servo //

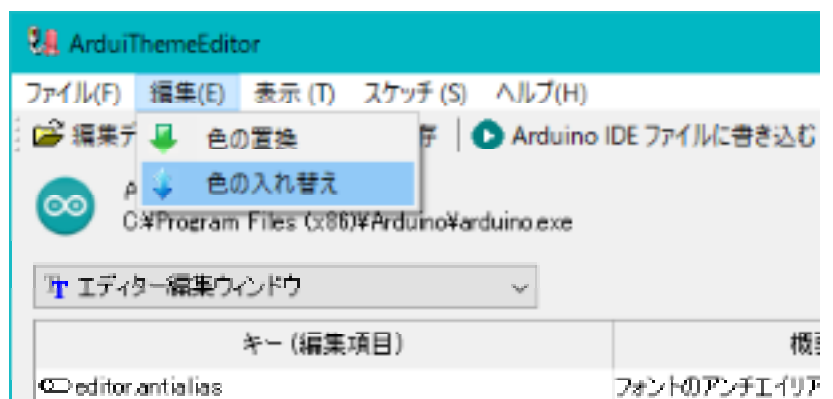
/* put your setup code here */
/* 初期設定コード */
void setup() {
```

### (1) 色の入れ替え

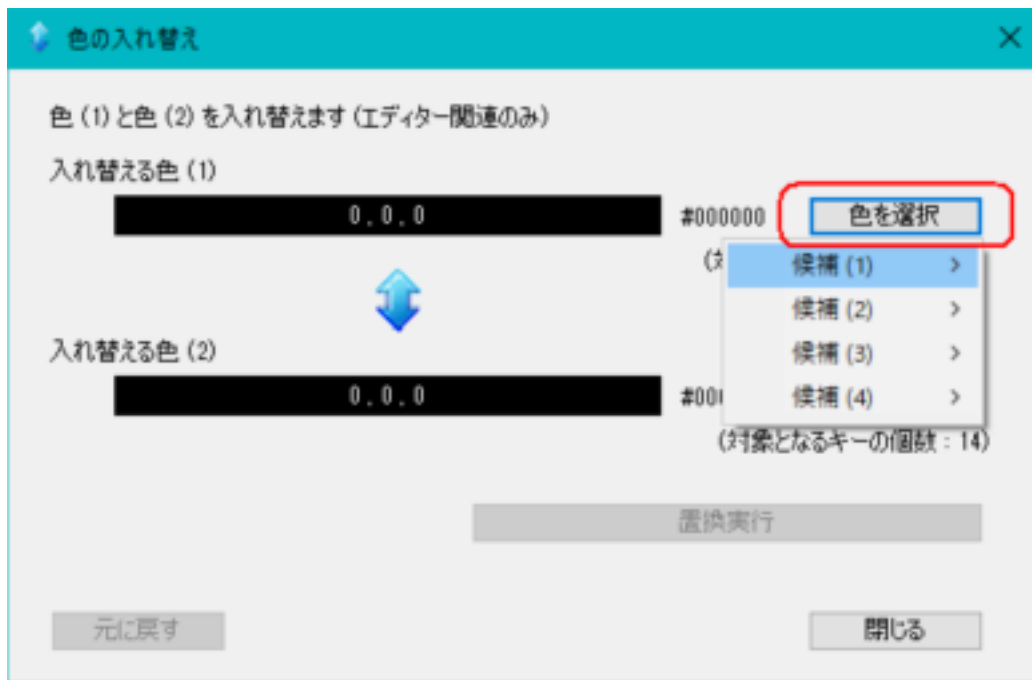
2つの色（白と黒）を指定して全て入れ替えます。

- 「メニュー」
- 「編集」
- 「色の入れ替え」

を選択します。



入れ替える色（1）のボタン「色を選択」を押します。



候補（1）～（4）には、他のキー（編集項目）の色が表示されます。白と黒を入れ替えるので、白を選びます。



(例) #FFFFFF (標準) 背景色 editor.bgcolor

入れ替える色 (1) (2) を選択してからボタン「置換実行」を押します。



ボタン「閉じる」を押します。

一覧表のキー「テキスト色」「背景色」などが全て入れ替わっています。

<input type="checkbox"/> editor.linenumbers	行番号を表示	false	
<input type="checkbox"/> editor.tabs.expand	TABキー (スペースまたは" <code>\t</code> "挿入)	true	
0-9 editor.tabs.size	TABキーによるスペース数	2	
<input type="checkbox"/> editor.fgcolor	(標準) テキスト色	#FFFFFF	255, 255, 255
<input type="checkbox"/> editor.bgcolor	(標準) 背景色	#000000	0, 0, 0
<input type="checkbox"/> editor.linehighlight.color	現在行の強調表示の色	#E2E2E2	226, 226, 226
<input type="checkbox"/> editor.linehighlight	現在行の強調表示 ON	false	

キー (編集項目)	概要	変更後
editor.fgcolor	(標準) テキスト色	255, 255, 255
editor.bgcolor	(標準) 背景色	0, 0, 0
<matchedBracket bg	対応する括弧 {...} の色(2) (内部)	0, 0, 0
<style token "LITERAL_NUMBER_DECIMAL_INT"	数字 (整数)	255, 255, 255
<style token "LITERAL_NUMBER_FLOAT"	数字 (実数/小数点)	255, 255, 255
<style token "LITERAL_NUMBER_HEXADECIMAL"	数字 (16進数 0xFF..)	255, 255, 255
<style token "SEPARATOR"	括弧 () {} のテキスト色	255, 255, 255



プレビューウィンドウも変更されています。



## (2) 基本16色から選択

演算記号 (+ - \* / = . , ; 等) を白に変更します。

キー (編集項目)	概要	変更後
editor.operator.style	演算記号 (+ - = , . : 等)	255, 255, 255

上記のキーを選択します。

キー (編集項目)	概要	値1	イメージ
editor.preprocessor.style	予約語 (#define 等)	#5E6D03	94, 109, 3
editor.url.style	※無効 (URLのテキスト色)	#0000FF	0, 0, 255
editor.operator.style	演算記号 (+ - = , . : 等)	#434F54	67, 79, 84
editor.label.style	※無効 (予約語 case, goto)	#7E7E7E	126, 126, 126
editor.comment1.style	コメント行 //	#434F54	67, 79, 84
editor.comment2.style	コメント範囲 /* ... */	#95A5A6	149, 165, 166

右クリックによりポップアップメニューを開きます。

メニュー → 「基本16色」 → RGB(255, 255, 255) (#FFFFFF)

キー (編集項目)	概要	値1	イメージ	値2	変更
editor.literal_string_double_...	char型 "... 内の文字列	#005C5F	0, 92, 95	plain	
editor.preprocessor.style	予約語 (#define 等)	#5E6D03	94, 109, 3	plain	
editor.url.style	※無効 (URLのテキスト色)	#0000FF	0, 0, 255	underlined	
editor.operator.style	演算記号 (+ - = . : 等)	#434F54	67, 79, 84	plain	
editor.label.style	※無効 (予約語 case, goto)	#7E7E7E			
editor.comment1.style		0, 0, 0			
editor.comment2.style		128, 128, 128			
<matchedBracket bg		192, 192, 192			
<style token "IDENTIFIER"		255, 255, 255	#FFFFFF	white	
<style token "DATA_TYPE"		128, 0, 0	#800000	maroon	
<style token "FUNCTION"		255, 0, 0	#FF0000	red	
<style token "VARIABLE"		128, 0, 128	#800080	purple	
<style token "RESERVED"		255, 0, 255	#FF00FF	fuchsia	
<style token "RESERVED"		0, 128, 0	#008000	green	
<style token "PREPROCESSOR"		0, 255, 0	#00FF00	lime	
<style token "ANNOTATION"		128, 128, 0	#808000	olive	
<style token "COMMENT"		255, 255, 0	#FFFF00	yellow	
<style token "COMMENT"		0, 0, 255	#0000FF	blue	
<style token "COMMENT"		0, 0, 128	#000080	navy	
<style token "COMMENT"		0, 128, 128	#008080	teal	
<style token "LITERAL"		0, 255, 255	#00FFFF	aqua	

色を設定

基本16色 >

色を直接指定 (RGB)

候補 (1) >

候補 (2) >

候補 (3) >

候補 (4) >

文字装飾 >

コピー

貼り付け

最近使った色 >

初期状態に戻す >

元に戻す

演算記号の色が「白」に変更されます。

キー (編集項目)	概要	値1	イメージ
editor.url.style	※無効 (URLのテキスト色)	#0000FF	0, 0, 255
editor.operator.style	演算記号 (+ - = . : 等)	#FFFFFF	255, 255, 255
editor.label.style	※無効 (予約語 case, goto)	#7E7E7E	128, 128, 128

### (3) カラーピッカー (HSL形式) による色選択

C++関数、if、for などの予約語の色をカラーピッカーにより変更します。

キー (編集項目)	概要	変更後
editor.reserved_word.style	予約語 (c++関数, if, for 等)	17, 204, 238

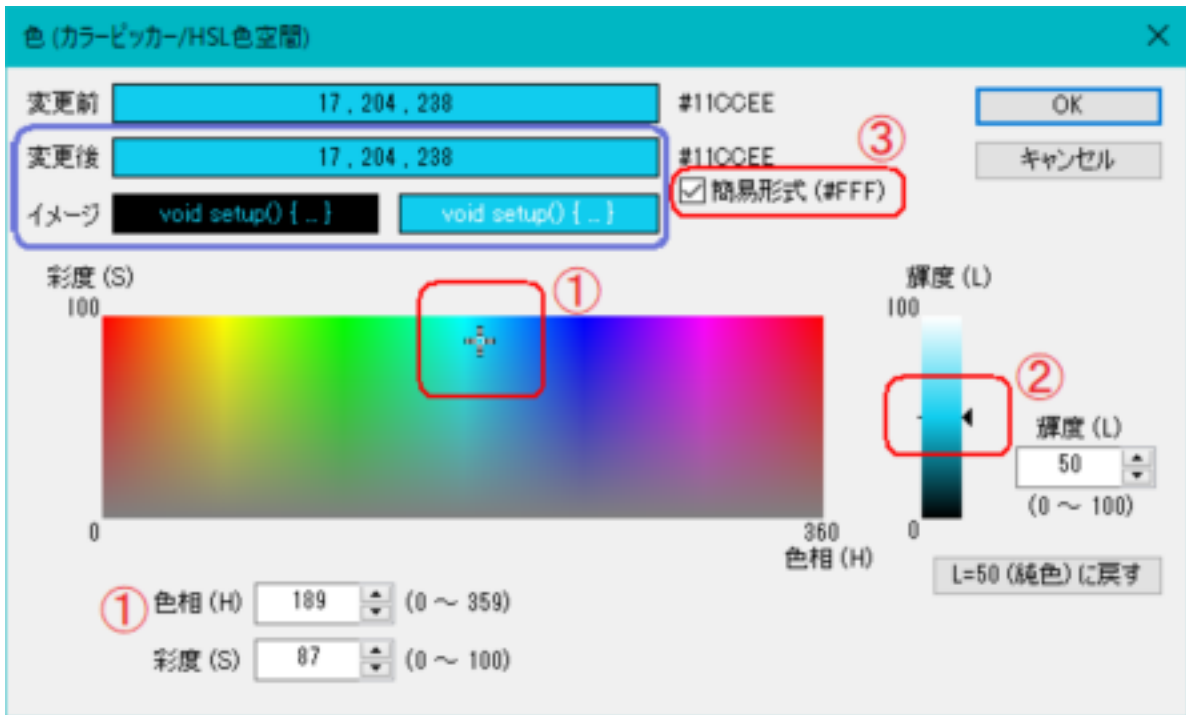
キーを選択して右クリックによりポップアップメニューを開きます。

メニュー → 「色を設定」

キー (編集項目)	概要	値1	イメージ	値2
editor.keyword2.style		#D35400	211, 84, 0	plain
editor.function.style	予約語 (メンバ関数, 操作関..	#D35400	211, 84, 0	plain
editor.keyword3.style		#5E6D03	94, 109, 3	plain
editor.reserved_word.style	予約語 (c++関数, if, for 等)	#5E6D03	94, 109, 3	plain
editor.literal1.style				plain
editor.literal2.style				plain
editor.variable.style				plain
editor.reserved_word_2.style	予約語 (型 void, int, long 等)			plain
editor.literal_boolean.style	予約語 (true/false)			plain
editor.literal_char.style				plain
editor.literal_string_double	char型 " " 内の文字列			plain

色を設定  
基本16色 >  
色を直接指定 (RGB)  
候補 (1) >  
候補 (2) >

HSL色空間により色を選択します。



初めに②の輝度Lを50に戻します。

輝度L (0~100) は明るさを意味し、L=50の場合が純色です。




色相Hと彩度Sは①をマウスクリック（または数値入力）することにより決定します。

輝度Lは②をマウスクリック（または数値入力）することにより決定します。

(例) ①H=189 , S=87    ②L=50    ③簡易形式をチェック  
RGB(17, 204, 238) (#11CCEE)

自動的にRGB形式に変換されます。

色を選択したらボタン「OK」を押します。

キー (編集項目)	概要	値1	イメージ
 editor.keyword3.style		#5E6D08	94, 109, 3
 editor.reserved_word.style	予約語 (c++関数, if, for 等)	#11CCEE	17, 204, 238
 editor.literal1.style		#00979C	0, 151, 156

#### (4) 直接指定 (RGB形式) による色選択

void、int、long などの予約語の色をRGB形式により変更します。

キー (編集項目)	概要	変更後
editor.reserved_word_2.style	予約語 (型 void, int, long 等)	68, 153, 238

キーを選択して右クリックによりポップアップメニューを開きます。

メニュー → 「色を直接指定(RGB)」

キー (編集項目)	概要	値1	イメージ	値2
 editor.reserved_word.style	予約語 (c++関数, if, for 等)	#11CCEE	17, 204, 238	plain
 editor.literal1.style		#00979C	0, 151, 156	plain
 editor.literal2.style		#00979C	0, 151, 156	plain
 editor.variable.style		#00979C	0, 151, 156	plain
 editor.reserved_word_2.style	予約語 (型 void, int, long 等)	#00979C	0, 151, 156	plain
 editor.literal_boolean.style	予約語 (true/false)			plain
 editor.literal_char.style				plain
 editor.literal_string_double_...	char型 "... 内の文字列			plain
 editor.preprocessor.style	予約語 (#define 等)			plain
 editor.url.style	※無効 (URLのテキスト色)			derlined
 editor.operator.style	演算記号 (+ - = . : 等)			plain
 editor.label.style	※無効 (予約語 case, goto)			bold
 editor.comment1.style	コメント行 //			plain

 色を設定

基本16色 >

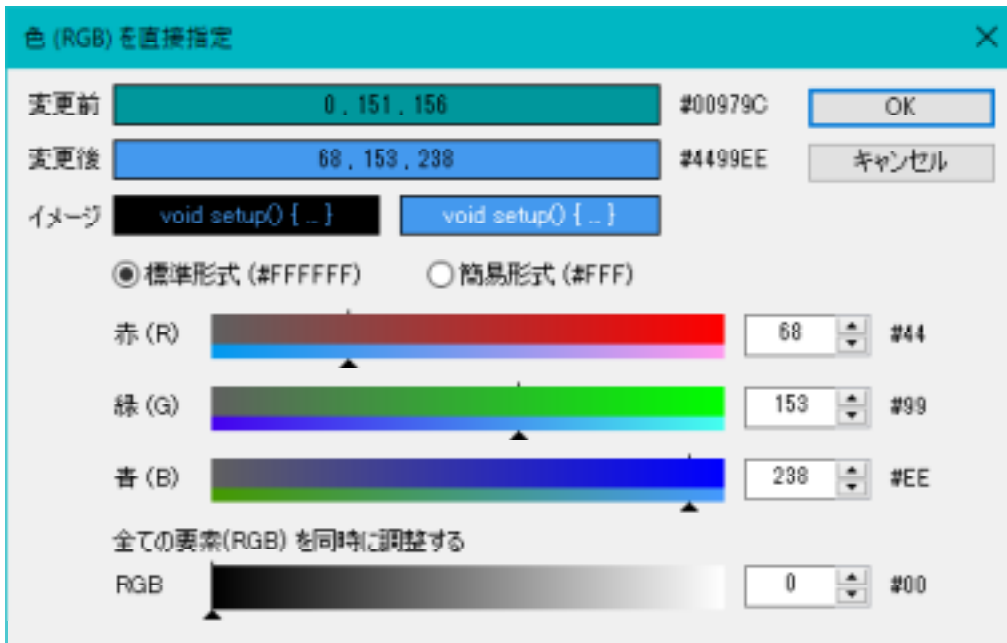
 色を直接指定 (RGB)

候補 (1) >

候補 (2) >

候補 (3) >

RGB形式により色を選択します。



(例) RGB(68, 153, 238) (#4499EE)

色を選択したらボタン「OK」を押します。

キー (編集項目)	概要	値1	イメージ
editor.variable.style		#00979C	0, 151, 156
editor.reserved_word_2.style	予約語 (型 void, int, long 等)	#4499EE	68, 153, 238
editor.literal_boolean.style	予約語 (true/false)	#00979C	0, 151, 156

## (5) コピー&貼り付けによる設定

予約語 (true/false) の色を設定します。

キー (編集項目)	概要	変更後
editor.literal_boolean.style	予約語 (true/false)	68, 153, 238

コピー元 : editor.reserved\_word\_2.style / 予約語 (型 void, int, long 等)

貼り付け先 : editor.literal\_boolean.style / 予約語 (true/false)

キー (編集項目)	概要	値1	イメージ
editor.variable.style		#00979C	0, 151, 156
editor.reserved_word_2.style	予約語 (型 void, int, long 等)	#4499EE	68, 153, 238
editor.literal_boolean.style	予約語 (true/false)	#00979C	0, 151, 156
editor.literal_char.style		#00979C	0, 151, 156

メニュー

キー "editor.reserved\_word\_2.style / 予約語 (型 void, int, long 等)" を選択して右クリックします。

メニュー → 「コピー」

キー (編集項目)	概要	値1	イメージ	値2
editor.variable.style		#00979C	0, 151, 156	plain
editor.reserved_word_2.style	予約語 (型 void, int, long 等)	#4499EE	68, 153, 238	plain
editor.literal_boolean.style	予約語 (true/false)			plain
editor.literal_char.style				plain
editor.literal_string_double...	char型 "... 内の文字列			plain
editor.preprocessor.style	予約語 (#define 等)			plain
editor.url.style	※無効 (URLのテキスト色)			derlined
editor.operator.style	演算記号 (+ =, .. 等)			plain
editor.label.style	※無効 (予約語 case, goto)			bold
editor.comment1.style	コメント行 //			plain
editor.comment2.style	コメント範囲 /* .. */			plain
<matchedBracket be	対応する括弧 [...] の色(2) (内..			
<style token "IDENTIFIER"				
<style token "DATA_TYPE"				bold
<style token "FUNCTION"				
<style token "VARIABLE"				
<style token "RESERVED_...				bold
<style token "RESERVED_...				bold
<style token "PREPROCE...				
<style token "ANNOTATIO...				

色を設定

基本16色 >

色を直接指定 (RGB)

候補 (1) >

候補 (2) >

候補 (3) >

候補 (4) >

文字装飾 >

コピー

貼り付け

最近使った色 >

初期状態に戻す >

元に戻す

キー "editor.literal\_boolean.style / 予約語 (true/false)" を選択して右クリックします。

メニュー → 「貼り付け」

キー (編集項目)	概要	値1	イメージ	値2
editor.variable.style		#00979C	0, 151, 156	plain
editor.reserved_word_2.style	予約語 (型 void, int, long 等)	#4499EE	68, 153, 238	plain
editor.literal_boolean.style	予約語 (true/false)	#00979C	0, 151, 156	plain
editor.literal_char.style				plain
editor.literal_string_double...	char型 "... 内の文字列			plain
editor.preprocessor.style	予約語 (#define 等)			plain
editor.url.style	※無効 (URLのテキスト色)			derlined
editor.operator.style	演算記号 (+ =, .. 等)			plain
editor.label.style	※無効 (予約語 case, goto)			bold
editor.comment1.style	コメント行 //			plain
editor.comment2.style	コメント範囲 /* .. */			plain
<matchedBracket be	対応する括弧 [...] の色(2) (内..			
<style token "IDENTIFIER"				
<style token "DATA_TYPE"				bold
<style token "FUNCTION"				
<style token "VARIABLE"				
<style token "RESERVED_...				bold
<style token "RESERVED_...				bold
<style token "PREPROCE...				
<style token "ANNOTATIO...				
<style token "COMMENT_...				

色を設定

基本16色 >

色を直接指定 (RGB)

候補 (1) >

候補 (2) >

候補 (3) >

候補 (4) >

文字装飾 >

コピー

貼り付け

最近使った色 >

初期状態に戻す >

元に戻す

貼り付けを実行すると色が上書きされます。

キー (編集項目)	概要	値1	イメージ	値2	変更
editor.variable.style		#00979C	0, 151, 156	plain	
editor.reserved_word_2.style	予約語 (型 void, int, long 等)	#4499EE	68, 153, 238	plain	(*)
editor.literal_boolean.style	予約語 (true/false)	#4499EE	68, 153, 238	plain	(*)
editor.literal_char.style		#00979C	0, 151, 156	plain	

## (6) 候補 (他のキー) から選択して設定

初めに以下のキーの色を緑に変更します。

**カラーピッカーまたはRGB直接指定で色を変更してください。**

キー (編集項目)	概要	変更後
editor.comment1.style	コメント行 //	0, 214, 0

変更後は以下のようになります。

editor.label.style	※無効 (予約語 case, goto)	#7E7E7E	126, 126, 126	bold
editor.comment1.style	コメント行 //	#00D600	0, 214, 0	変更済み
editor.comment2.style	コメント範囲 /* ... */	#95A5A6	149, 165, 166	plain

次に以下のキーの設定を「候補から選択」によって行います。

キー (編集項目)	概要	変更後
editor.comment2.style	コメント範囲 /* ... */	0, 214, 0

キーを選択して右クリックします。

メニュー → 「候補(1)」

キー (編集項目)	概要	値1	イメージ	値2
editor.operator.style	演算記号 (+ - = . : 等)	#FFFFFF	255, 255, 255	plain
editor.label.style	※無効 (予約語 case, goto)	#7E7E7E	126, 126, 126	bold
editor.comment1.style	コメント行 //	#00D600	0, 214, 0	plain
editor.comment2.style	コメント範囲 /* ... */	#95A5A6	149, 165, 166	plain
<matchedBracket bg	対応する括弧 [ ] の色(2) (内...			
<style token "IDENTIFIER"				bold
<style token "DATA_TYPE"				bold
<style token "FUNCTION"				bold
<style token "VARIABLE"				bold
<style token "RESERVED_...				bold
<style token "RESERVED_...				bold
<style token "PREPROCE...				
<style token "ANNOTATIO...				

色を設定

- 基本16色 >
- 色を直接指定 (RGB) >
- 候補 (1) >
- 候補 (2) >
- 候補 (3) >
- 候補 (4) >



メニュー

- 「候補(1)」
- 「コメント行 // editor.comment1.style」



選択すると以下のように色が変更されます。

キー (編集項目)	概要	値1	イメージ	値2
editor.operator.style	演算記号 (+ - = , : 等)	#FFFFFF	255, 255, 255	plain
editor.label.style	※無効 (予約語 case, goto)	#7E7E7E	126, 126, 126	bold
editor.comment1.style	コメント行 //	#00D600	0, 214, 0	plain
editor.comment2.style	コメント範囲 /* .. */	#00D600	0, 214, 0	plain
<matchedBracket be	対応する括弧 [ ] の色(2) (内..	#000000	0, 0, 0	
<style token "IDENTIFIER"		#FFFFFF	255, 255, 255	



## (7) 履歴 (最近使った色) から選択

カーソルの色を設定します。

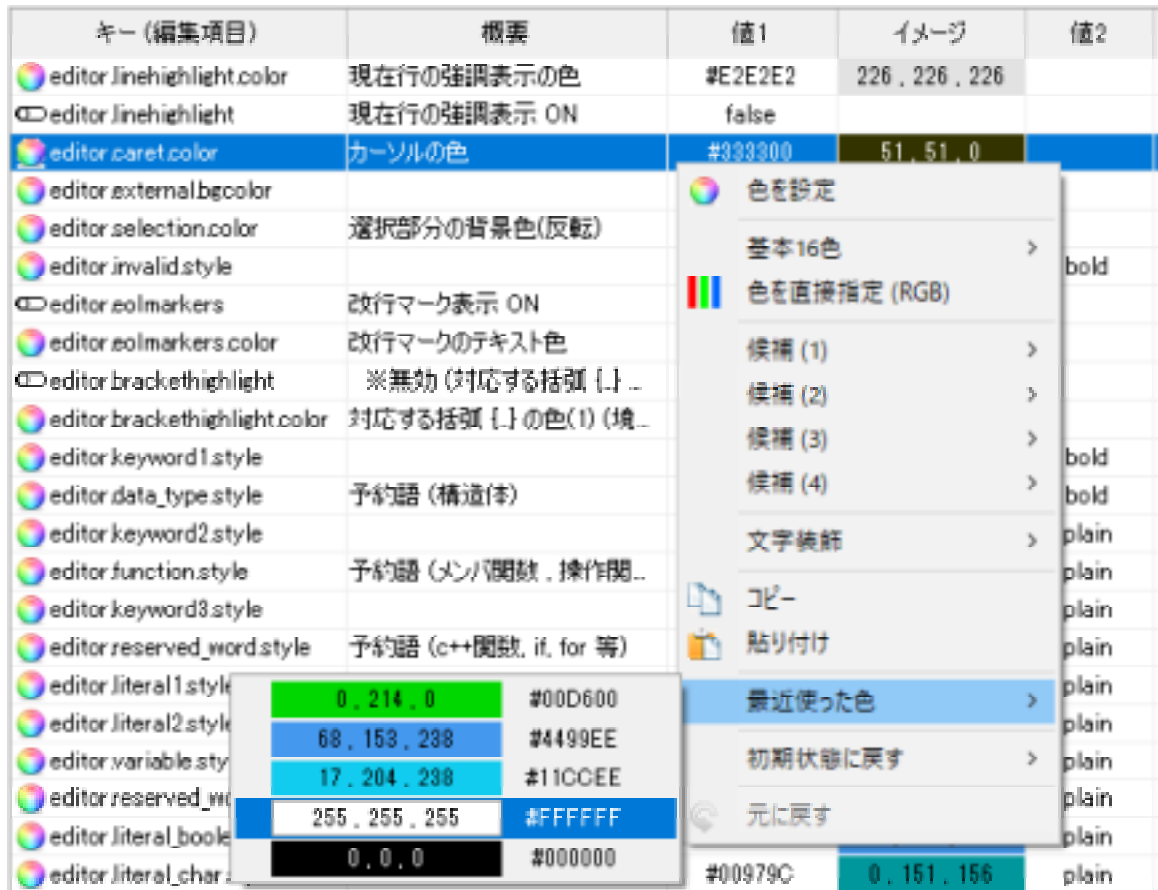
キー (編集項目)	概要	変更後
editor.caret.color	カーソルの色	255, 255, 255

キーを選択して右クリックします。

### メニュー「表示」

→ 「最近使った色」

→ 「255, 255, 255 #FFFFFF」 ※ 白色



キー (編集項目)	概要	値1	イメージ	値2	
editor.linehighlight.color	現在行の強調表示の色	#E2E2E2	226, 226, 226		
editor.linehighlight	現在行の強調表示 ON	false			
editor.caret.color	カーソルの色	#333300	51, 51, 0		
editor.externalbgcolor					
editor.selection.color	選択部分の背景色(反転)				
editor.invalid.style				bold	
editor.eolmarkers	改行マーク表示 ON				
editor.eolmarkers.color	改行マークのテキスト色				
editor.brackethighlight	※無効 (対応する括弧 [...] ..				
editor.brackethighlight.color	対応する括弧 [...] の色(1) (境...				
editor.keyword1.style				bold	
editor.data_type.style	予約語 (構造体)			bold	
editor.keyword2.style				plain	
editor.function.style	予約語 (メンバ関数, 操作関...			plain	
editor.keyword3.style				plain	
editor.reserved_word.style	予約語 (c++関数, if, for 等)			plain	
editor.literal1.style		0, 214, 0	#00D600	plain	
editor.literal2.style		68, 153, 238	#4499EE	plain	
editor.variable.style		17, 204, 238	#11CCEE	plain	
editor.reserved_mi		255, 255, 255	#FFFFFF	plain	
editor.literal_boole		0, 0, 0	#000000	plain	
editor.literal_char			#00979C	0, 151, 156	plain

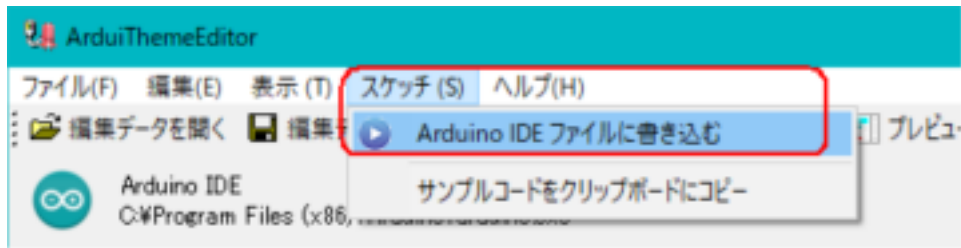
選択すると以下のように色が変更されます。

キー (編集項目)	概要	値1	イメージ
editor.linehighlight	現在行の強調表示 ON	false	
editor.caret.color	カーソルの色	#FFFFFF	255, 255, 255
editor.externalbgcolor		#C8D2DC	200, 210, 220

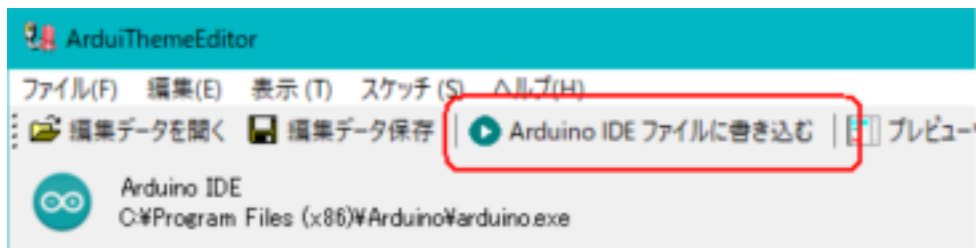
## (8) Arduino IDE 更新 (テーマファイル書き込み)

今まで編集した内容を Arduino IDE に反映させていきます。

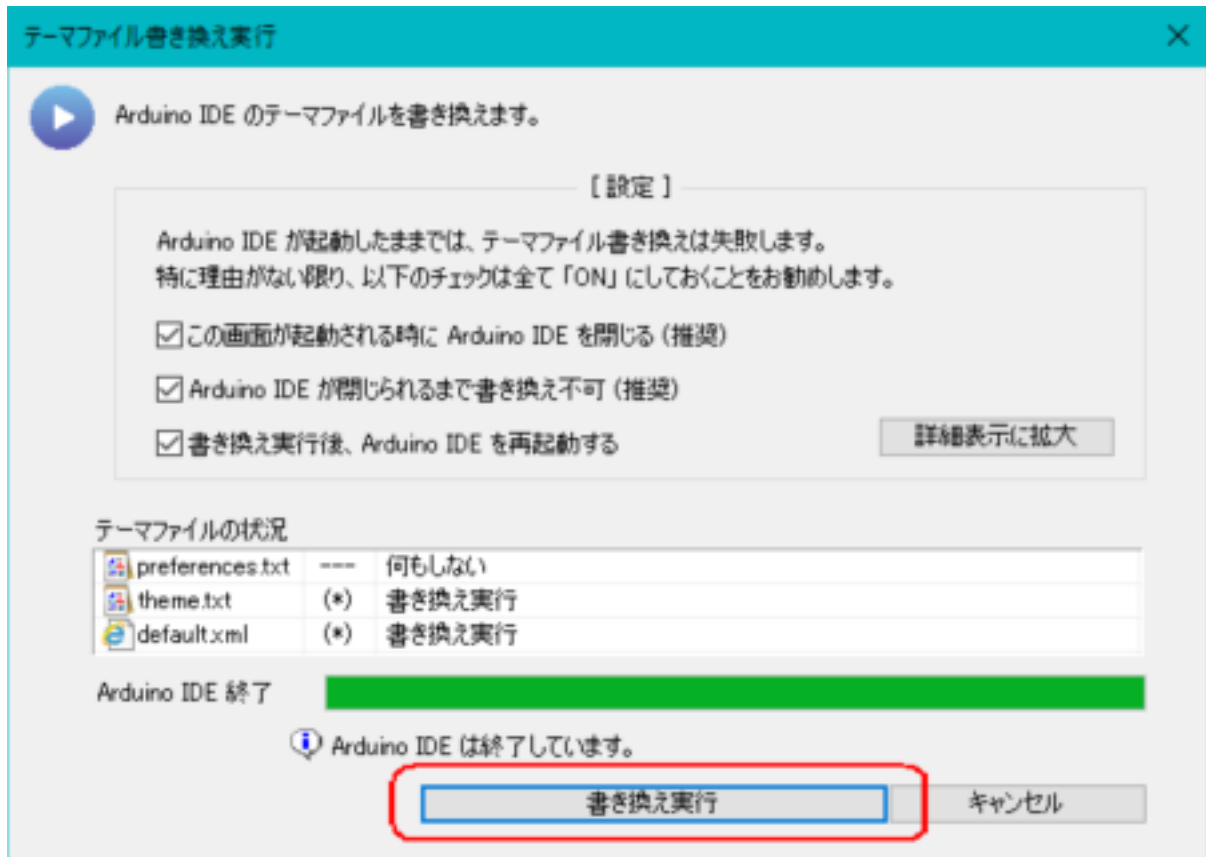
メニュー「スケッチ」 → 「Arduino IDE ファイルに書き込む」



または以下のツールバーボタンを押します。



「テーマファイル書き換え実行」画面が表示されます。



ボタン「書き換え実行」を押します。

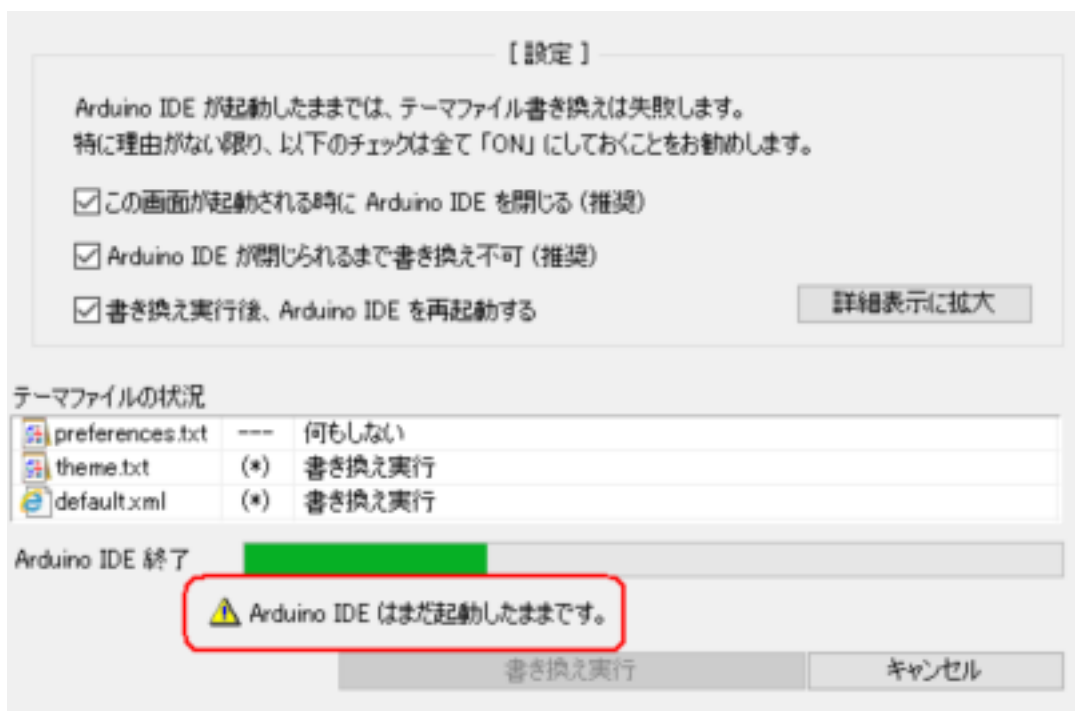
ボタンが押せない状態の場合、次ページの後半へ進んでください。

テーマファイルが書き換えられて Arduino IDE が起動します。



※ Arduino IDE

※ ボタン「書き換え実行」が押せない場合



Arduino IDE が起動している場合、テーマファイルを書き換えることができません。

Arduino IDEを全て終了して下さい。

## (9) その他の色の変更

以下のキーの色を設定してください。

キー (編集項目)	概要	変更後
editor.linehighlight.color	現在行の強調表示の色 <b>editor.linehighlight=trueの場合のみ有効</b>	226, 226, 226
editor.selection.color	選択部分の背景色(反転)	153, 153, 153
editor.data_type.style	予約語 (構造体)	211, 84, 0
editor.function.style	予約語 (メンバ関数, 操作関連)	221, 136, 34
editor.literal_string_double_quote.style	char型 "... " 内の文字列	238, 221, 17
editor.preprocessor.style	予約語 (#define 等)	238, 85, 221

対応する括弧の強調表示を消去します。



```
{
// if ... //
if (k < 3) {
digitalRead();
myservo.write();
}
// switch - case //
switch(k)
{
```

キー (編集項目)	概要	変更後
editor.brackethighlight.color	対応する括弧 {...} の色(1) (境界線)	0, 0, 0
<matchedBracket bg	対応する括弧 {...} の色(2) (内部)	0, 0, 0

※editor.brackethighlight = false にしても表示されるため、背景色と同じ色にして見えないようにします。

以上の流れでダークテーマの色設定が行われました。  
第3章ではフォントやタブなどの設定を行います。

# 3. テーマファイル各種設定

Arduino IDE の機能（フォントやタブなど）の設定方法を解説します。

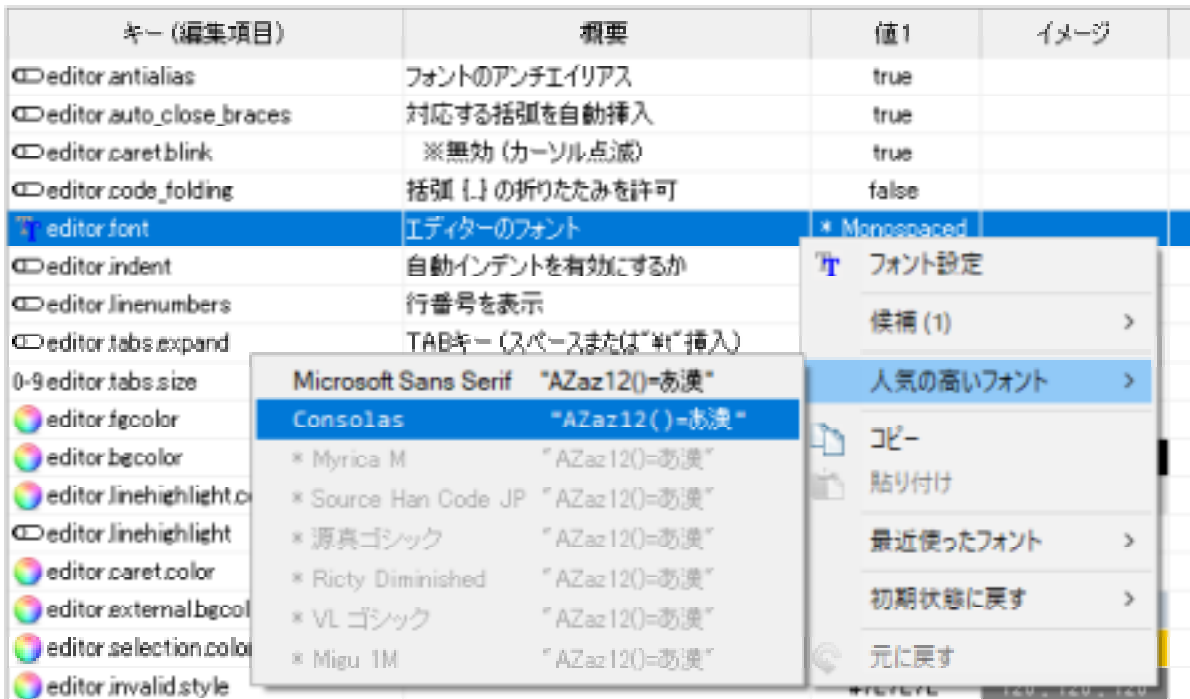
## (1) フォントの設定「人気の高いフォント」

エディター領域のフォントを変更します。

キー（編集項目）	概要	変更後
editor.font	エディターのフォント	Consolas

キーを選択して右クリックします。

メニュー → 「人気の高いフォント」 → 「Consolas」



選択すると以下のようにフォントが設定されます。

editor.code_folding	括弧【】の折りたたみを許可	false	
editor.font	エディターのフォント	Consolas	Consolas
editor.indent	自動インデントを有効にするか	true	

「人気の高いフォント」は「プログラミングに適しているフォント」として有名なフォントを集めて表示しています。

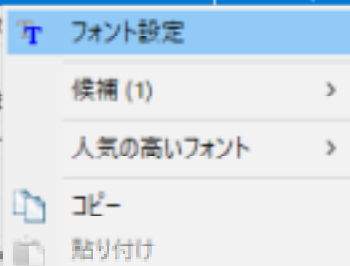
フォント名の前に "\*" が表示されている場合、Windowsシステム内にフォントが存在しないことを意味しています。

## (2) フォントの設定「一覧より選択」

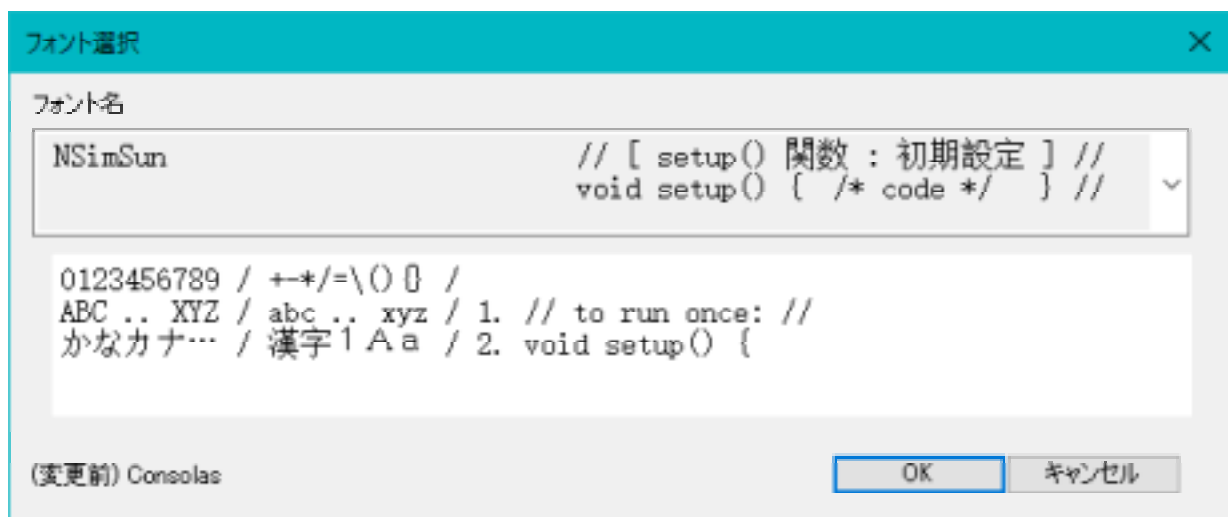
(1) 同様にキーを選択して右クリックします。

メニュー → 「フォント設定」

キー (編集項目)	概要	値1	イメージ
<input type="checkbox"/> editor.aliases	フォントのアンチエイリアス	true	
<input type="checkbox"/> editor.auto_close_braces	対応する括弧を自動挿入	true	
<input type="checkbox"/> editor.caret_blink	※無効 (カーソル点滅)	true	
<input type="checkbox"/> editor.code_folding	括弧 [ ] の折りたたみを許可	false	
<input checked="" type="checkbox"/> editor.font	エディターのフォント	* Monospaced	
<input type="checkbox"/> editor.indent	自動インデントを有効		
<input type="checkbox"/> editor.linenumbers	行番号を表示		
<input type="checkbox"/> editor.tabs.expand	TABキー (スペースま		
0-9 editor.tabs.size	TABキーによるスベ		
<input type="checkbox"/> editor.fgcolor	(標準) テキスト色		255, 255, 255
<input type="checkbox"/> editor.bgcolor	(標準) 背景色		0, 0, 0
<input type="checkbox"/> editor.linehighlight.color	現在行の強調表示		226, 226, 226



フォント設定画面が表示されます。



フォントを選択したらボタン「OK」を押します。

フォントサイズは Arduino IDE の「環境設定」において変更することができます。

### (3) TABキーの動作「半角スペースで埋めるか」設定

TAB キーに動作「半角スペースで埋めるか」を選択します。

キー (編集項目)	概要	変更後
editor.tabs.expand	TABキー (スペースまたは" <code>%t</code> "挿入)	false

キーを選択して右クリックします。

メニュー → 「false 設定」

キー (編集項目)	概要	値1	イメージ
editor.code_folding	括弧 [ ] の折りたたみを許可	false	
editor.font	エディターのフォント	Consolas	Consolas
editor.indent	自動インデントを有効にするか	true	
editor.linenumbers	行番号を表示	false	
editor.tabs.expand	TABキー (スペースまたは" <code>%t</code> "挿入)	false	
0-9 editor.tabs.size	TABキーによるスペース数		
editor.fgcolor	(標準) テキスト色	255, 255, 255	
editor.becolor	(標準) 背景色	0, 0, 0	
editor.linehighlight.color	現在行の強調表示の色	226, 226, 226	
editor.linehighlight	現在行の強調表示の色		
editor.caret.color	カーソルの色	255, 255, 255	

キー「editor.tabs.expand」は、行頭のインデントについて「タブではなく半角スペースを埋めるか？」という動作設定を定義しています。  
true = 半角スペースで埋める  
false = タブのまま使用する (推奨)

### (4) インデントの幅 (スペース数) の設定

TAB キーによるインデントの幅を入力します。

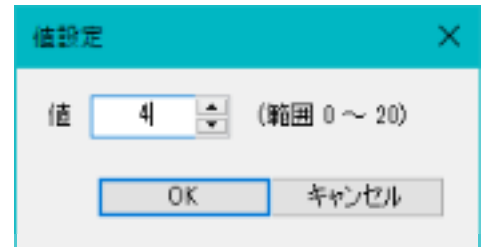
キー (編集項目)	概要	変更後
editor.tabs.size	TABキーによるスペース数	4または2

キーを選択して右クリックします。

メニュー → 「数値設定」

キー (編集項目)	概要	値1	イメージ
editor.linenumbers	行番号を表示	false	
editor.tabs.expand	TABキー (スペースまたは" <code>%t</code> "挿入)	false	
0-9 editor.tabs.size	TABキーによるスペース数	2	
editor.fgcolor	(標準) テキスト色	255, 255, 255	
editor.becolor	(標準) 背景色	0, 0, 0	
editor.linehighlight.color	現在行の強調表示の色	226, 226, 226	
editor.linehighlight	現在行の強調表示の色		
editor.caret.color	カーソルの色	#FFFFFF	255, 255, 255

値を入力してボタン「OK」を押します。



## (5) その他のキー設定

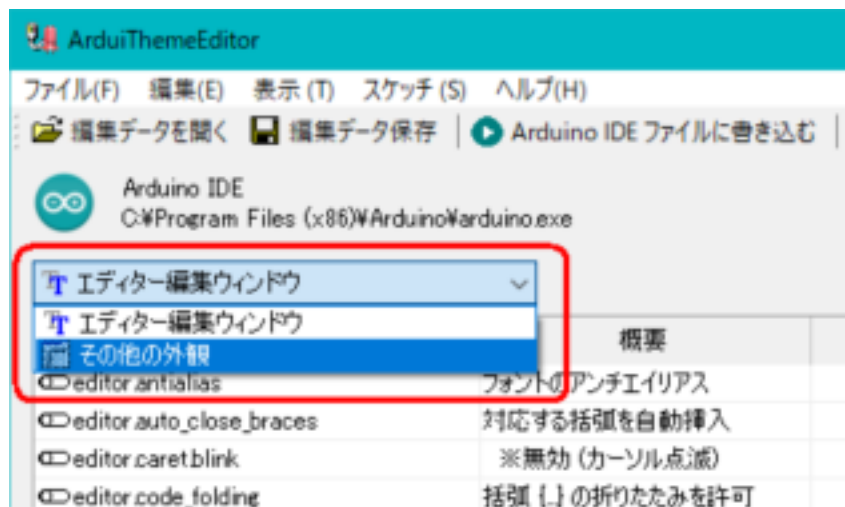
以下のキーを設定してください。

キー (編集項目)	概要	値
editor.auto_close_braces	対応する括弧を自動挿入	true/false
editor.linenumbers	行番号を表示	true/false
editor.linehighlight	現在行の強調表示 ON	true/false
editor.eolmarkers	改行マーク表示 ON	true/false

※これらのキーは自由に設定してください。

## (6) 外観のキーを変更する場合

Arduino IDE の外観 (ツールバーやエラーメッセージ欄など) を編集したい場合は一覧表を切り替えます。



メニュー「Arduino IDE ファイルに書き込む」を実行して、テーマファイルを書き換えてください。

以上の流れでテーマファイルの設定は完了しました。



# 4. その他の機能

アプリケーションの機能を解説します。

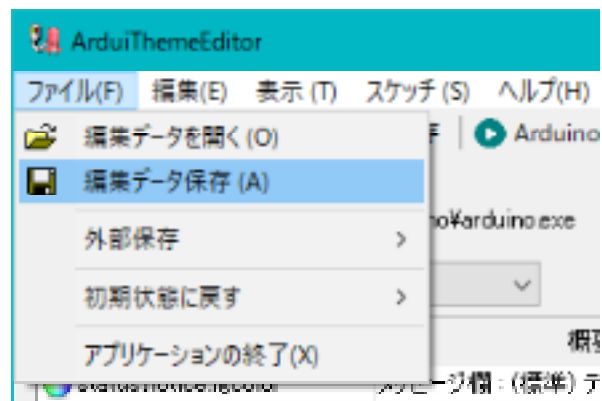
## (1) 編集データのファイル保存と読み取り

現在の編集内容を専用ファイル（拡張子 .at9）としてファイル保存します。

### メニュー「ファイル」

→ 「編集データ保存」

を選択します。



または、ツールバーボタン

### 「編集データ保存」

を押します。



説明画面が表示されます。



**操作の説明**

**メニュー「編集データ保存」読み取り**  
現在の編集内容をファイル形式「\*.at9」で保存したり読み込んだりします。  
(ファイル名を選択します)

値1	イメージ	値2
#333300	51, 51, 0	
#08D2DC	200, 210, 220	
#FF0000	255, 204, 0	
#7E7E7E	128, 128, 128	bold
false		
#999999	153, 153, 153	
true		
#006699	0, 102, 153	
#085400	211, 84, 0	
#085400	211, 84, 0	
#085400	211, 84, 0	plain
#085400	211, 84, 0	italic

Local  
JidentityService  
Apps  
2.0  
9NW...W1Y  
mydata.at9

**メニュー「Arduino テーマファイルの出力」**  
現在の編集内容からテーマファイルを外部に書き出して保存します。  
(ファイル名を選択します)

値1	イメージ	値2
#333300	51, 51, 0	
#08D2DC	200, 210, 220	
#FF0000	255, 204, 0	
#7E7E7E	128, 128, 128	bold

Local  
JidentityService  
Apps  
2.0  
theme.txt

他メニュー「テーマファイルの出力」「テーマファイルのバックアップ」と混同しないように説明画面が毎回表示されます。

保存先フォルダとファイル名を指定するとキーの内容がファイル保存されます。

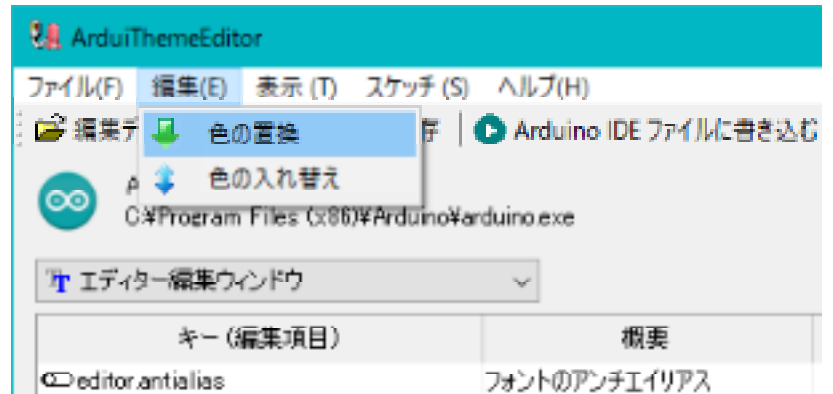
読み取る場合は「編集データ読み取り」を選びます。保存したファイルを指定することでキーの内容を復元することができます。

## (2) 色の置換

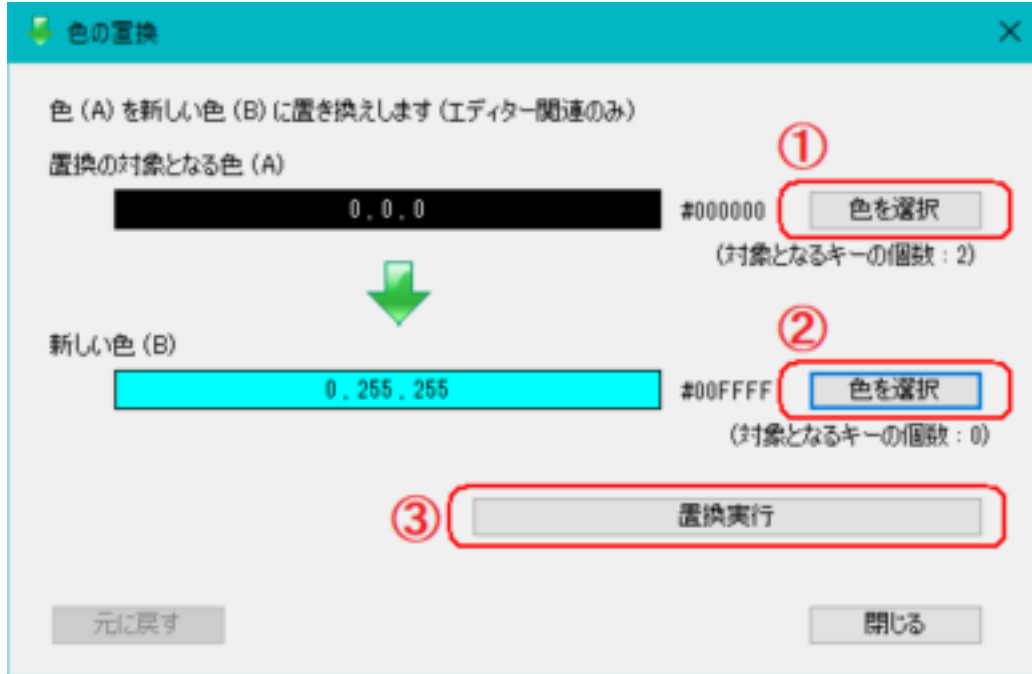
指定した色を他の色に置き換えます。

「メニュー」  
→ 「編集」  
→ 「色の置換」

を選択します。



置換の対象となる色 (A) を新しい色 (B) に置き換えます。  
※ 外観 (エディター以外のキー) は置換されません。



①②のボタン「色を選択」により色を選択します。

③のボタン「置換実行」を押すことにより色が置き換えられます。

### (3) 「元に戻す」と初期化

各キーを変更した場合、直前の値を記録します。

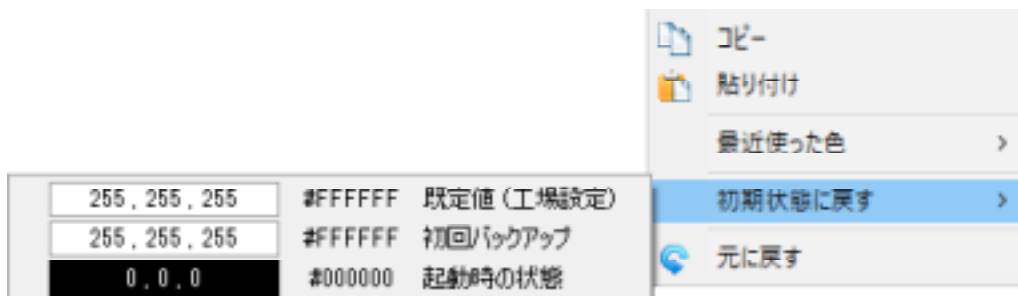
元に戻す場合は、キーを選択して右クリックします。

メニュー → 「元に戻す」



初期状態に戻したい場合、同様にキーを選択して右クリックします。

メニュー → 「初期状態に戻す」



値を選択します。

### (4) 全体の初期化

全てのキーを初期状態に戻す場合はメニューから選択します。

「メニュー」  
→ 「ファイル」  
→ 「初期状態に戻す」



#### 既定値 (工場設定)

Arduino IDE をインストールした状態

#### 初回バックアップ復元

アプリケーションをインストールした直後の状態

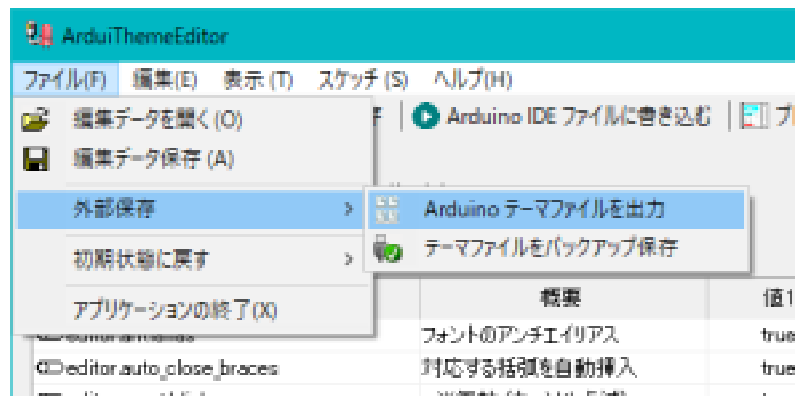
#### 現在の状態に戻す

アプリケーションを起動した時の状態

## (5) その他の機能

テーマファイルをパソコン上に保存する場合はメニューから選択します。

「メニュー」  
→ 「ファイル」  
→ 「外部保存」



### Arduino テーマファイルを出力

現在の編集内容からテーマファイルを復元してパソコン上に保存  
(フォルダを指定することにより3種類のテーマファイルが作成されます)

### テーマファイルをバックアップ保存

現在 Arduino IDE で使用されているテーマファイルをパソコン上にコピー保存  
(フォルダを指定することにより3種類のテーマファイルがコピー複製されます)

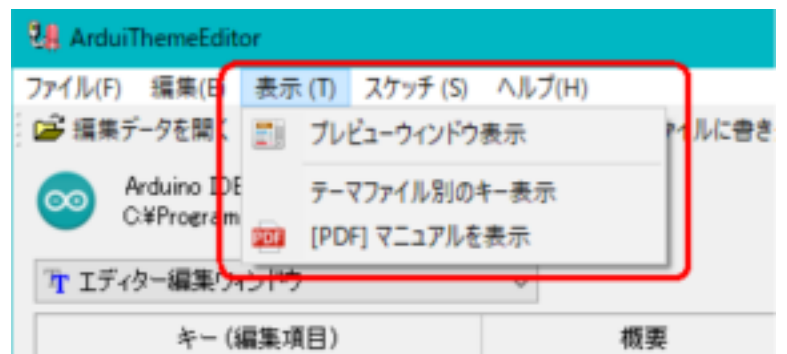
メニュー「表示」の各項目の概要は以下の通りです。

### プレビューウィンドウ表示

閉じてしまったプレビュー  
ウィンドウを再表示

### テーマファイル別のキー表示

キー全体をファイル別に分類  
して表示します。 5  
(別ウィンドウで表示)

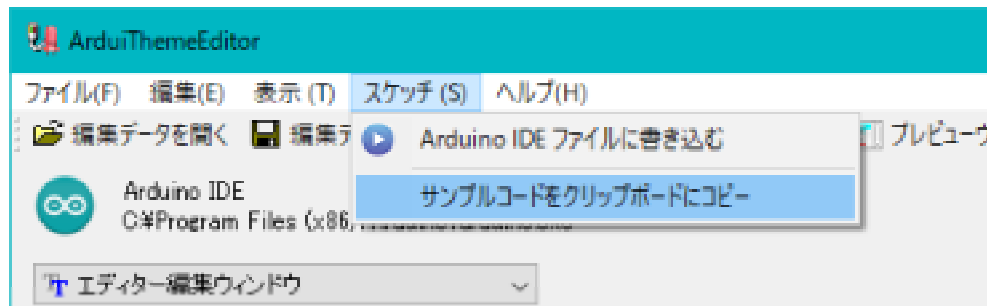


### [PDF] マニュアル表示

アプリケーション付属の本マニュアルを表示します。

サンプルコードをクリップボードにコピーする場合はメニューから選択します。

メニュー → 「スケッチ」 → 「サンプルコードをクリップボードにコピー」

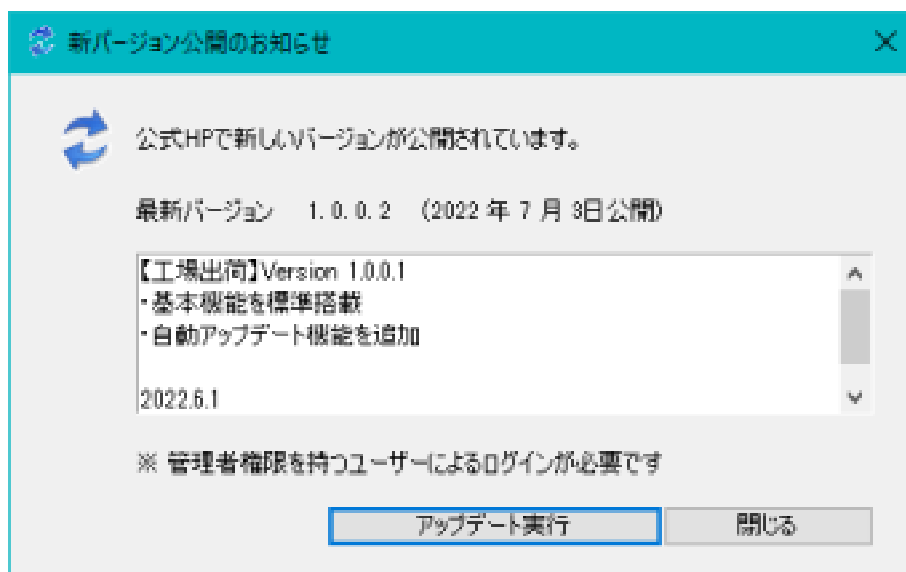


## (6) アップデート

アプリケーションの新しいバージョンがホームページ上で公開された場合、自動的にアップデートを行います。  
(インストーラを再びダウンロードする必要はありません)

アプリケーションは、起動時に公式ホームページにアクセスして、新しいバージョンが公開されているかチェックを行います。

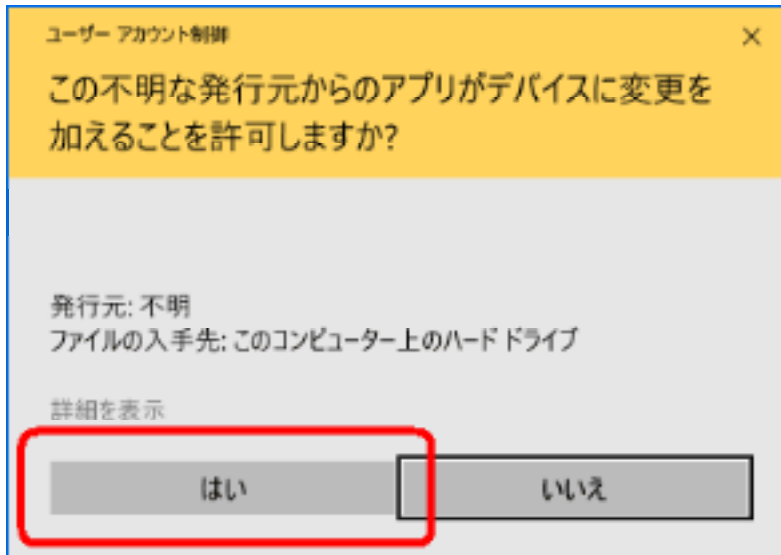
新しいバージョンが公開されている場合、以下のメッセージが表示されます。



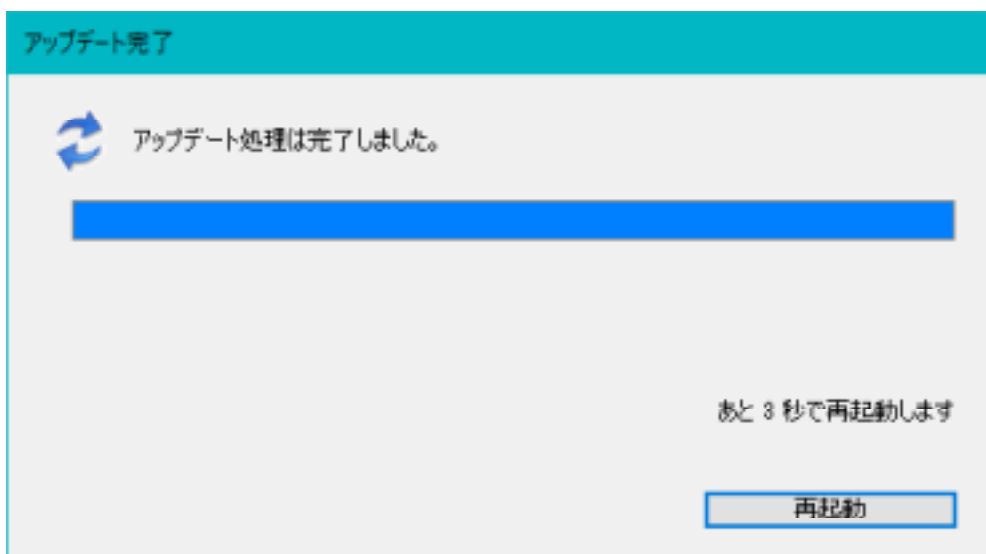
以下の点に注意してください。

- ・ 管理者権限でのログイン（パスワード入力）が求められます。
- ・ 他のアプリケーションは全て閉じてください。

ボタン「アップデート実行」を押します。



ボタン「はい」をクリックします。



アップデートが自動的に終わりました。

## (7) アンインストール

Windowsスタートメニュー  
→ 「Arduino テーマエディター」  
→ 「アンインストール」

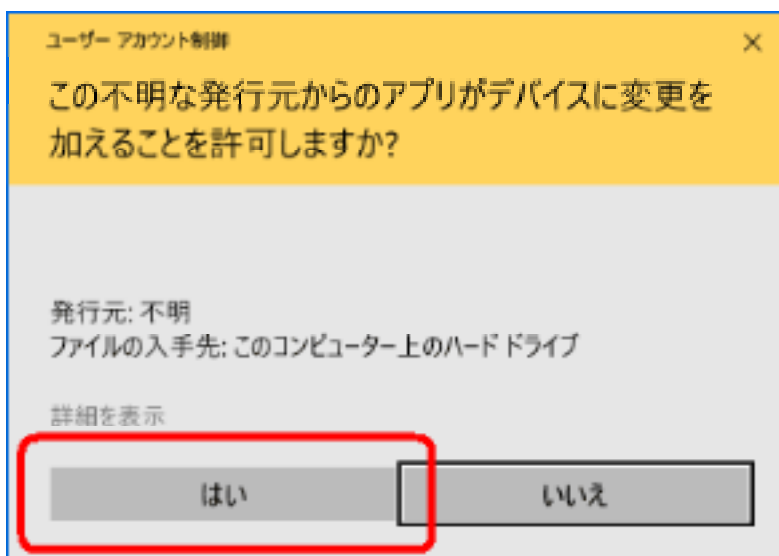
をクリックします。



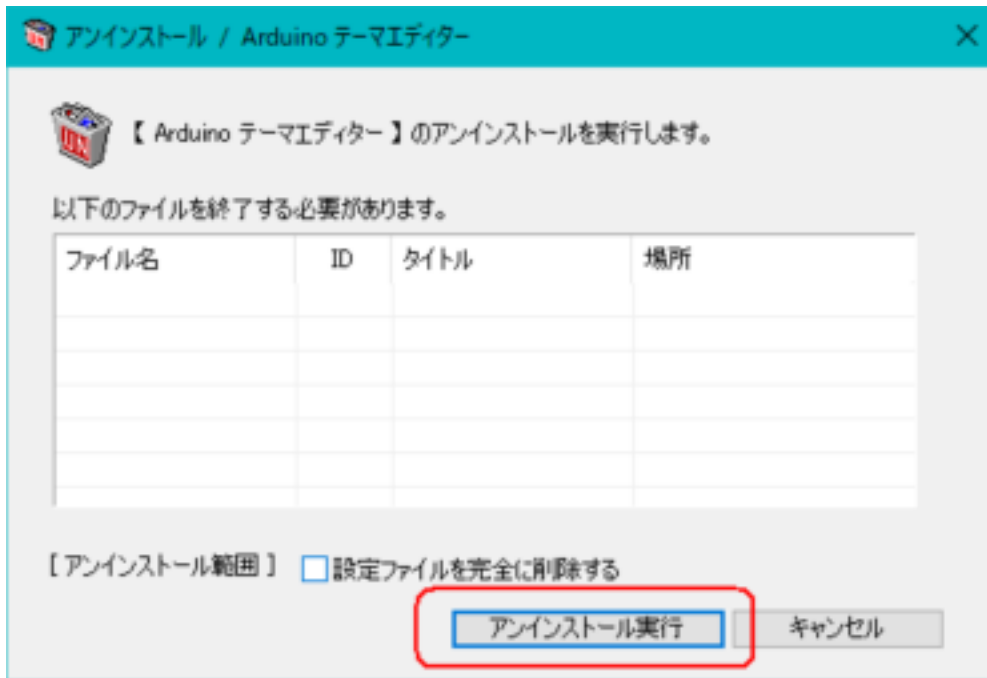
ボタン「アンインストール実行画面を起動」を押します。

以下の点に注意してください。

- ・ 管理者権限でのログイン（パスワード入力）が求められます。
- ・ 他のアプリケーションは全て閉じてください。



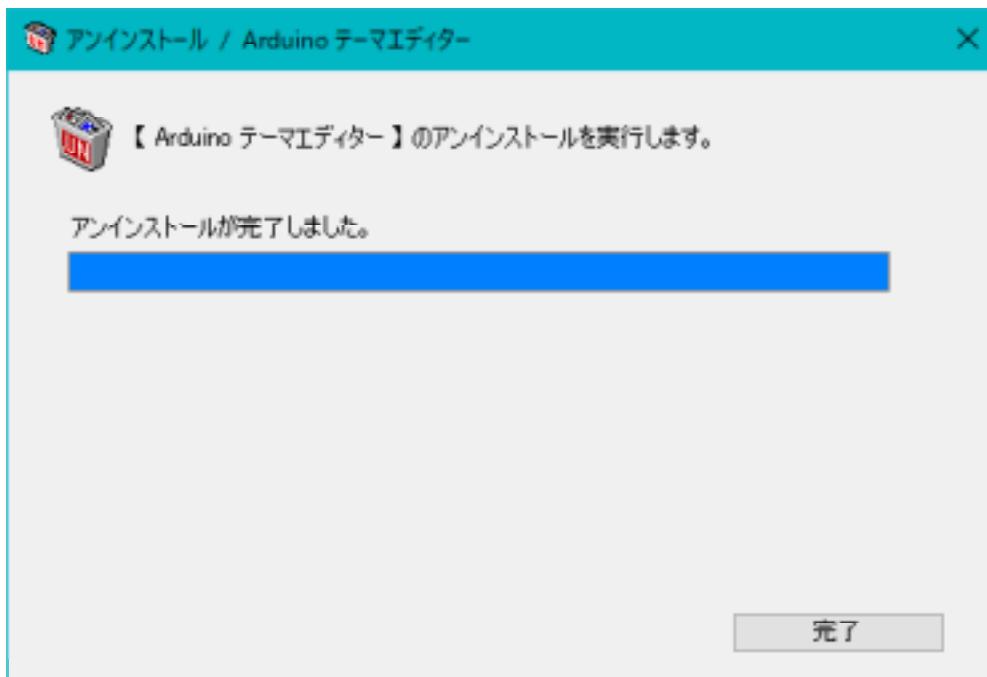
ボタン「はい」をクリックします。



今後再インストールする予定がなく設定ファイルを完全削除したい場合は、チェックボックス「設定ファイルを完全に削除する」をONにしてください。

ボタン「アンインストール実行」を押します。

アンインストールが行われます。





## 注意点

---

本書の内容（写真、画像、文章等）の一部または全てを無断で転載・複製することは禁止されています。

画像には著作権があります。

本書の内容は、利用者の同意・事前確認を得ることなく変更することがあります。

利用者が本書および本書に関連するコンテンツ、リンク先サイトにおける一切のサービス等をご利用されたことに起因または関連して生じた一切の損害（間接的であると直接的であるとを問わない）について、当社は責任を負わないものとします。

## マニュアルの構成

---

マニュアルは以下の分冊で構成されています。各分冊の概要は以下の通りです。

（[こちら](#)で記載されたマニュアルが本書です）

- ・ 準備編（インストールガイド）  
インストールの流れを説明しています。
- ・ 操作編（チュートリアル）  
操作方法について説明しています。

2022年6月1日 初版

Copyright (C) 2022 YoYonTek Developer