

# プログラミング概論

第14回 2023年12月20日

JavaScriptによるアプリ開発

# 今回の授業内容

- JavaScriptによるアプリの開発
  - ピアノアプリの開発（+作業時間の計測）
- AppInventorで改めてピアノをつくって**作業時間**を計測
- JavaScriptとAppInventorで太鼓をつくって**入力時間**を比較

今回計測したタイムは次のレポートを書く際に必要になります。必ず記録し無くさないように注意すること。

# 作業時間と入力時間の違い

- 作業時間とは
  - 入力開始から完成までにかかった時間
  - 入力し終わるまでではなく、修正時間も含む
  - AppInventorの場合はデザイナー画面での作業も含む
- 入力時間とは
  - 特定の箇所を入力するのににかかった時間
  - AppInventorの場合はブロックの組み立てにかかった時間（デザイナー画面の作業は含まない）
  - 修正時間は含まない

# JAVASCRIPTによるアプリの開発

# 開発から実行までの流れ

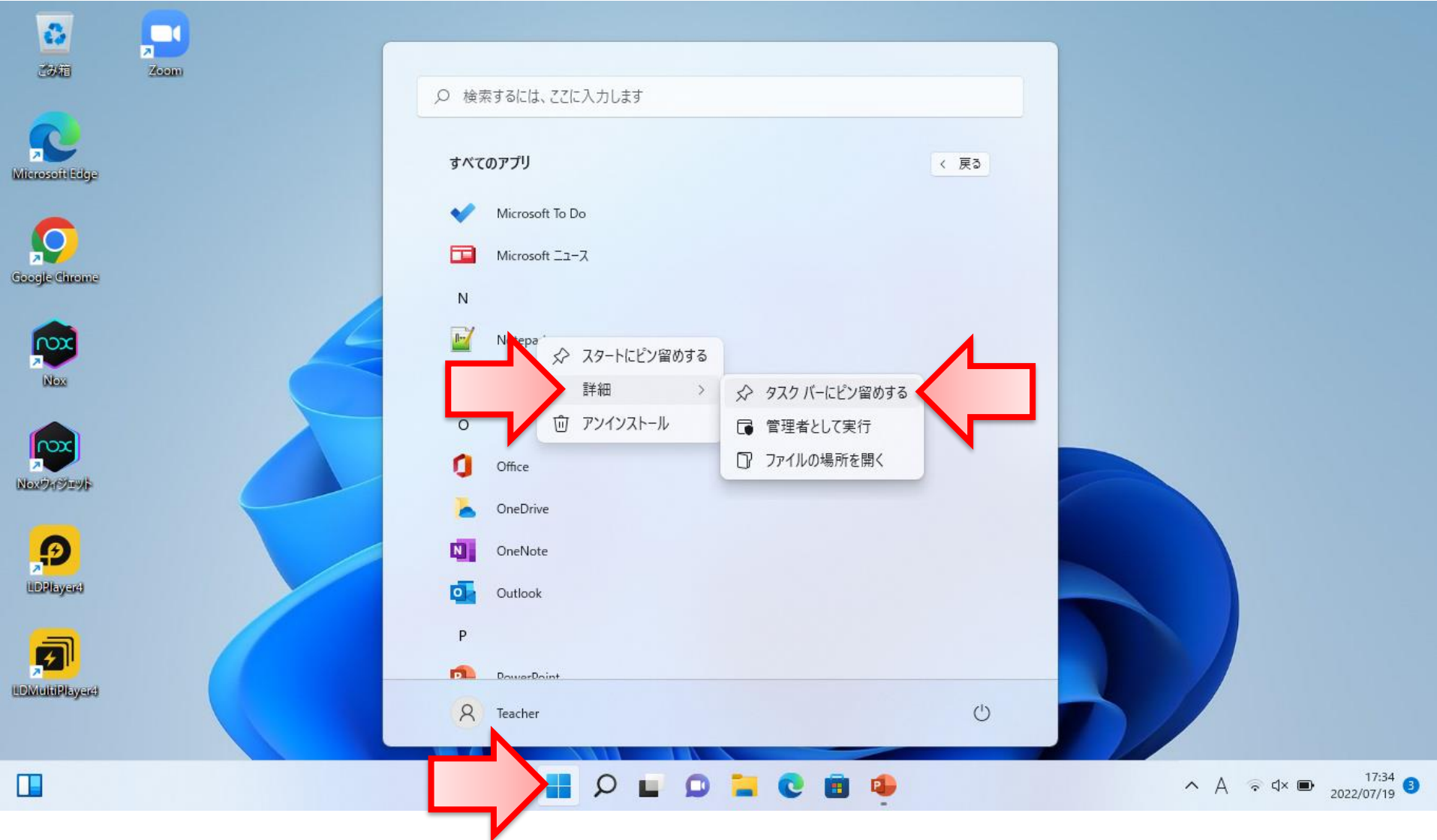
1. Notepad++を起動する
2. プログラムを慎重に入力する  
& 入力間違いがないか何度も見直す
3. 保存する
4. アップロードする
5. 自分のスマートフォンで実行する

今回はウェブアプリなのでiPhoneでも動きます！

# 今回の作業の進め方の注意

- まずはピアノアプリをJavaScriptで作成する
- ピアノアプリは作業時間を計測するため作業を中断するときにはストップウォッチを一時停止する
- ピアノアプリが正しく動くまでは太鼓アプリの作成に取りかからない

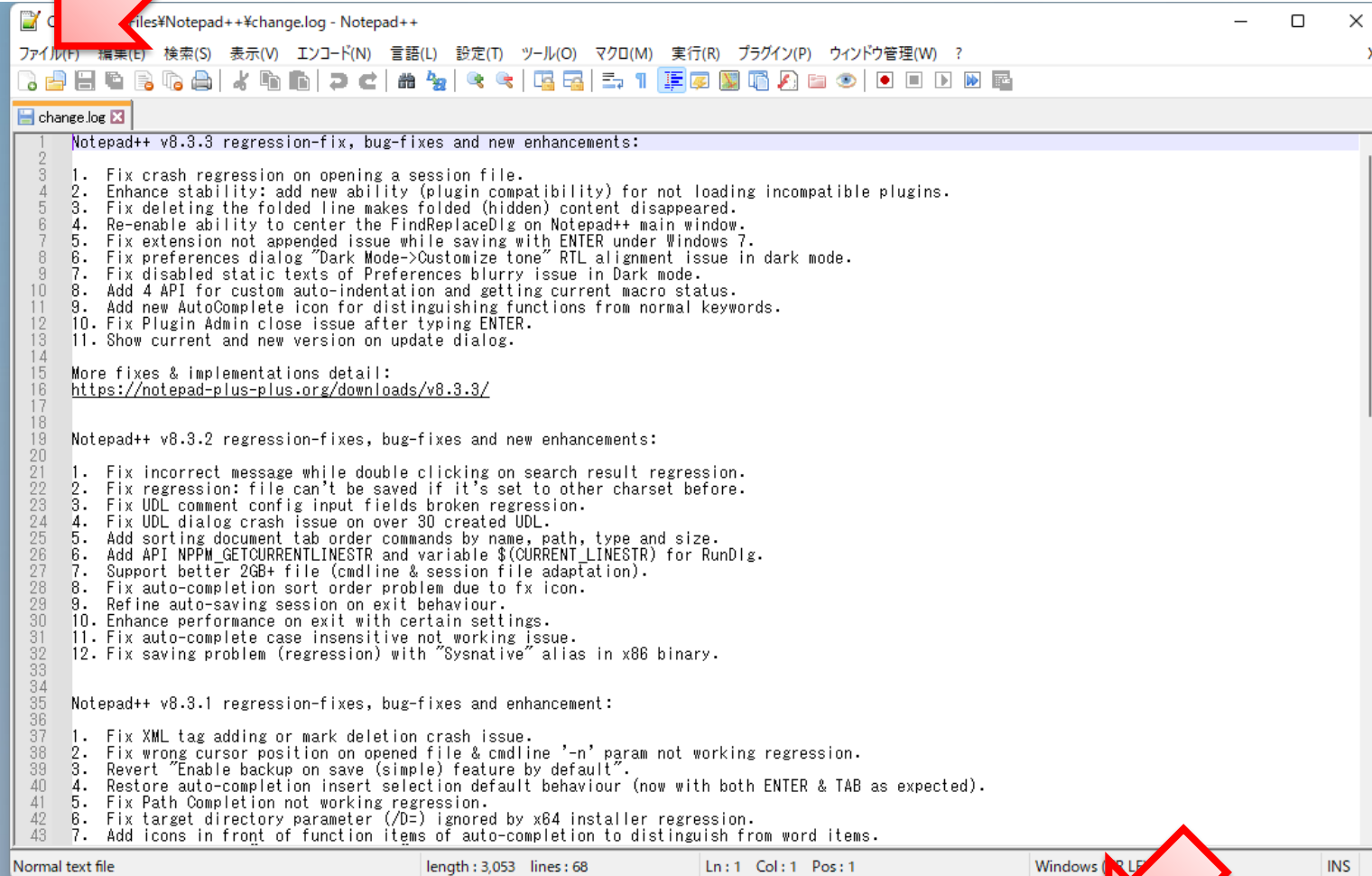
# Notepad++をタスクバーにピン留めする



スタートメニュー → [すべてのアプリ] → [詳細] → [タスクバーにピン留めする]

# Notepad++を起動する

[ファイル] → [新規作成]

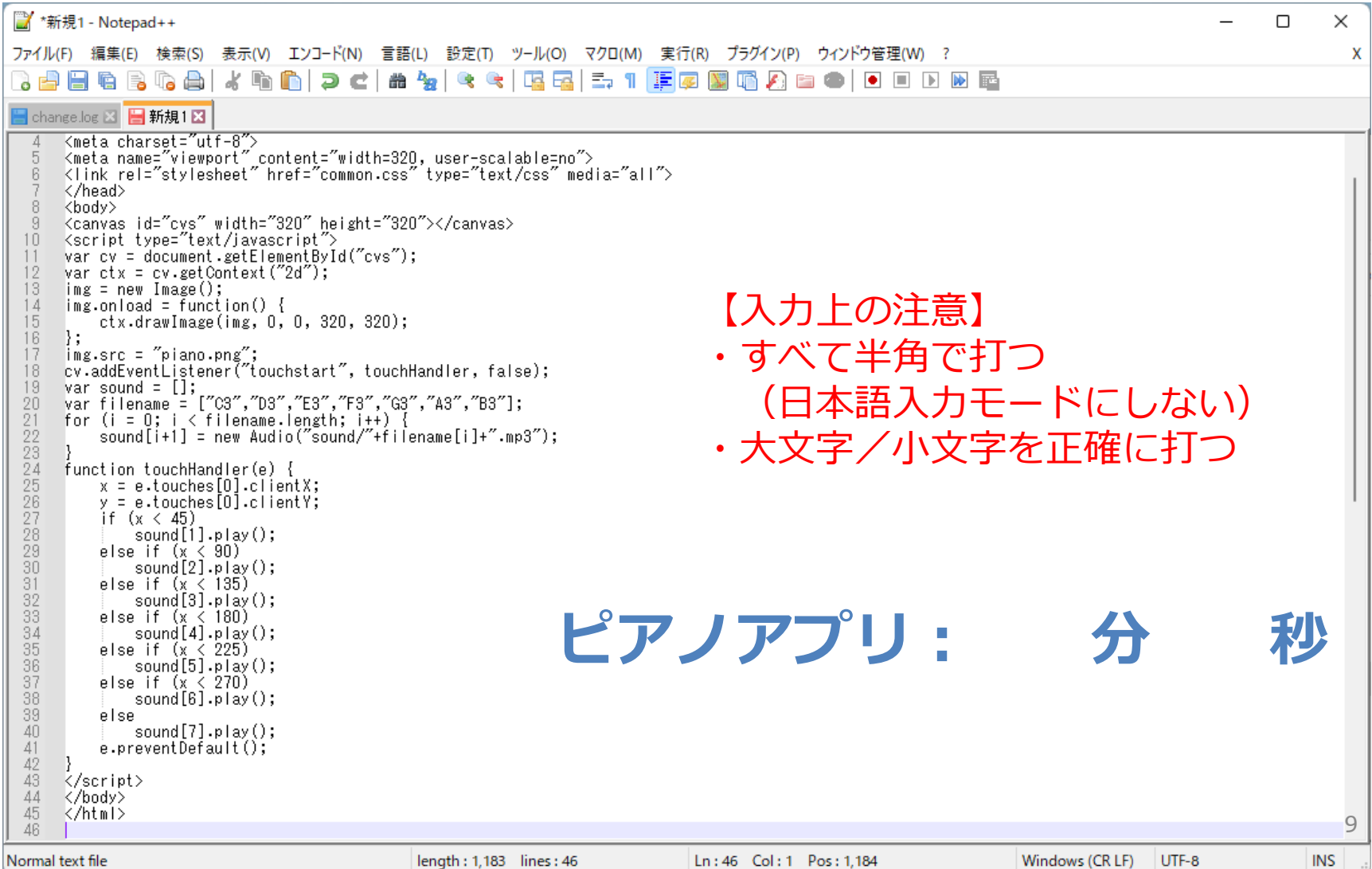


```
1 Notepad++ v8.3.3 regression-fix, bug-fixes and new enhancements:
2
3 1. Fix crash regression on opening a session file.
4 2. Enhance stability: add new ability (plugin compatibility) for not loading incompatible plugins.
5 3. Fix deleting the folded line makes folded (hidden) content disappeared.
6 4. Re-enable ability to center the FindReplaceDlg on Notepad++ main window.
7 5. Fix extension not appended issue while saving with ENTER under Windows 7.
8 6. Fix preferences dialog "Dark Mode->Customize tone" RTL alignment issue in dark mode.
9 7. Fix disabled static texts of Preferences blurry issue in Dark mode.
10 8. Add 4 API for custom auto-indentation and getting current macro status.
11 9. Add new AutoComplete icon for distinguishing functions from normal keywords.
12 10. Fix Plugin Admin close issue after typing ENTER.
13 11. Show current and new version on update dialog.
14
15 More fixes & implementations detail:
16 https://notepad-plus-plus.org/downloads/v8.3.3/
17
18 Notepad++ v8.3.2 regression-fixes, bug-fixes and new enhancements:
19
20
21 1. Fix incorrect message while double clicking on search result regression.
22 2. Fix regression: file can't be saved if it's set to other charset before.
23 3. Fix UDL comment config input fields broken regression.
24 4. Fix UDL dialog crash issue on over 30 created UDL.
25 5. Add sorting document tab order commands by name, path, type and size.
26 6. Add API NPPM_GETCURRENTLINESTR and variable $(CURRENT_LINESTR) for RunDlg.
27 7. Support better 2GB+ file (cmdline & session file adaptation).
28 8. Fix auto-completion sort order problem due to fx icon.
29 9. Refine auto-saving session on exit behaviour.
30 10. Enhance performance on exit with certain settings.
31 11. Fix auto-complete case insensitive not working issue.
32 12. Fix saving problem (regression) with "Sysnative" alias in x86 binary.
33
34 Notepad++ v8.3.1 regression-fixes, bug-fixes and enhancement:
35
36
37 1. Fix XML tag adding or mark deletion crash issue.
38 2. Fix wrong cursor position on opened file & cmdline '-n' param not working regression.
39 3. Revert "Enable backup on save (simple) feature by default".
40 4. Restore auto-completion insert selection default behaviour (now with both ENTER & TAB as expected).
41 5. Fix Path Completion not working regression.
42 6. Fix target directory parameter (/D=) ignored by x64 installer regression.
43 7. Add icons in front of function items of auto-completion to distinguish from word items.
```



# プログラムを慎重に入力する

## 入力開始から完成までにかかった時間を計測する (ストップウォッチのウェブサイトやスマホアプリ等を利用して)



```
4 <meta charset="utf-8">
5 <meta name="viewport" content="width=320, user-scalable=no">
6 <link rel="stylesheet" href="common.css" type="text/css" media="all">
7 </head>
8 <body>
9 <canvas id="cvs" width="320" height="320"></canvas>
10 <script type="text/javascript">
11   var cv = document.getElementById("cvs");
12   var ctx = cv.getContext("2d");
13   img = new Image();
14   img.onload = function() {
15     ctx.drawImage(img, 0, 0, 320, 320);
16   };
17   img.src = "piano.png";
18   cv.addEventListener("touchstart", touchHandler, false);
19   var sound = [];
20   var filename = ["C3", "D3", "E3", "F3", "G3", "A3", "B3"];
21   for (i = 0; i < filename.length; i++) {
22     sound[i+1] = new Audio("sound/"+filename[i]+".mp3");
23   }
24   function touchHandler(e) {
25     x = e.touches[0].clientX;
26     y = e.touches[0].clientY;
27     if (x < 45)
28       sound[1].play();
29     else if (x < 90)
30       sound[2].play();
31     else if (x < 135)
32       sound[3].play();
33     else if (x < 180)
34       sound[4].play();
35     else if (x < 225)
36       sound[5].play();
37     else if (x < 270)
38       sound[6].play();
39     else
40       sound[7].play();
41     e.preventDefault();
42   }
43 </script>
44 </body>
45 </html>
46
```

**【入力上の注意】**

- すべて半角で打つ  
(日本語入力モードにしない)
- 大文字／小文字を正確に打つ

**ピアノアプリ : 分 秒**

Normal text file | length: 1,183 lines: 46 | Ln: 46 Col: 1 Pos: 1,184 | Windows (CR LF) UTF-8 INS

# 保存する

[ファイル] → [上書き保存]

The screenshot shows a Notepad++ window with the File menu open. The 'Save' option (上書き保存) is highlighted in blue. A red arrow points from the text above to the File menu, and another red arrow points from the File menu to the 'Save' option. The window title is '新 ad+'. The menu items are as follows:

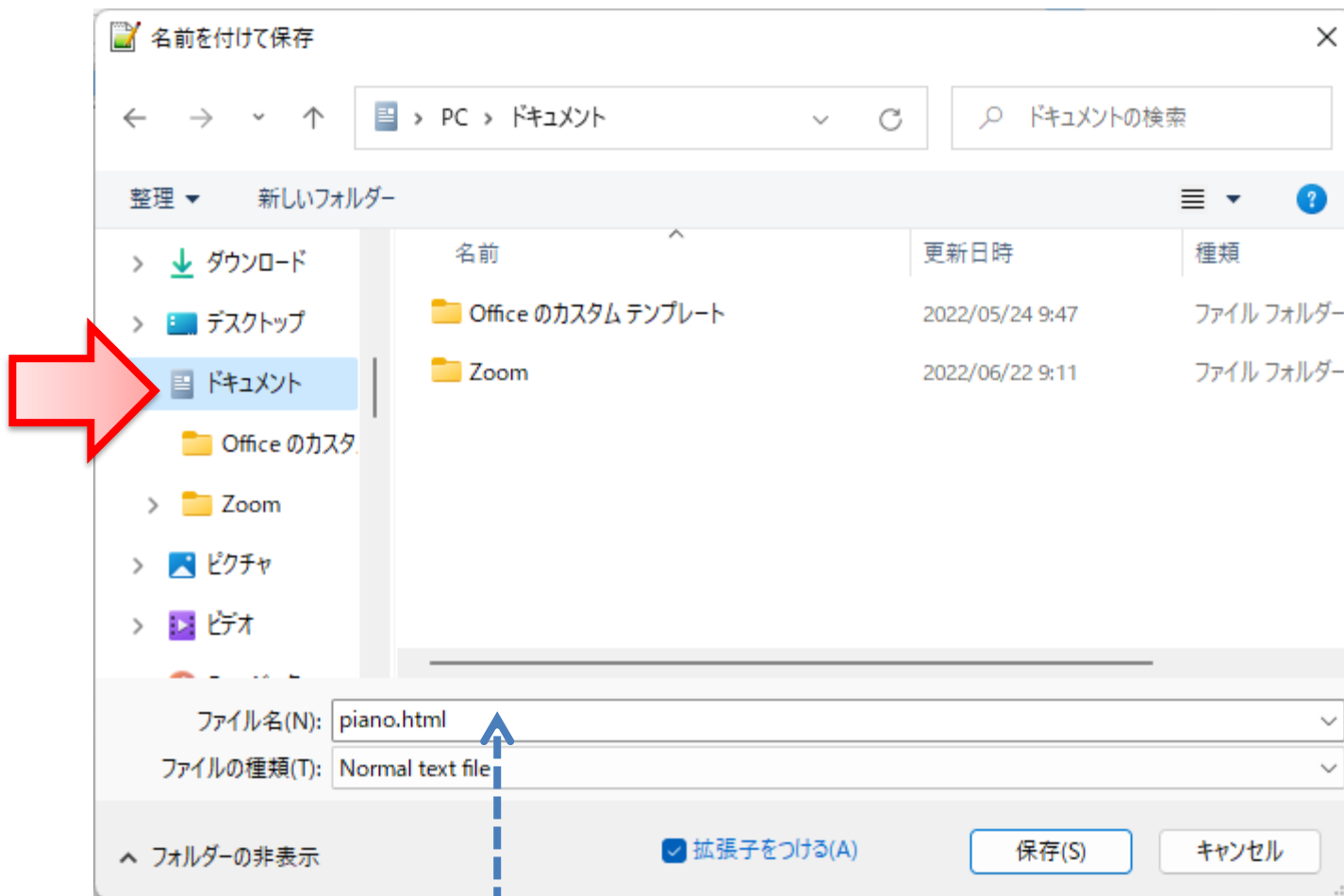
- 新規作成(N) Ctrl+N
- 開く(O)... Ctrl+O
- このファイルのあるフォルダーを開く(F) >
- 既定のアプリで開く(D)
- フォルダーをワークスペースに追加(W)...
- 再読み込み(L) Ctrl+R
- 上書き保存(S) Ctrl+S**
- 名前を付けて保存(A)... Ctrl+Alt+S
- 複製を別名で保存(Y)...
- すべて保存(E) Ctrl+Shift+S
- 名前の変更(R)...
- 閉じる(C) Ctrl+W
- すべて閉じる(E) Ctrl+Shift+W
- 他のファイルを閉じる(M) >
- ごみ箱に移動(B)
- セッションを読み込む(I)...
- セッションを保存(I)...
- 印刷(P)... Ctrl+P
- 今すぐ印刷(W)
- 終了(X) Alt+F4

The main text area contains the following code:

```
<!--scalable=no-->  
<!--text/css" media="all">  
  
</canvas>  
  
ler, false);  
3", "B8");  
e[i]+".mp3");
```

The status bar at the bottom shows: Normal text file, length: 1,183, lines: 46, Ln: 46, Col: 1, Pos: 1,184, Windows (CR LF), UTF-8, INS.

# 保存する



ピアノアプリ : piano.html  
太鼓アプリ : taiko.html

# アップロードする

## レポートについて

- [レポートサンプル](#)
- [優秀レポート紹介](#) new!
- [よくある質問](#)

## 課題の提出サイト

- [課題1](#) 楽器アプリの制作 (〆切: 11月28日23:59)
- [課題2](#) 的当てゲームの制作 (〆切: 12月19日23:59)
- [課題3](#) 天気予報アプリの制作 (〆切: 1月9日23:59)
- [JavaScript実行用](#)

## フリー素材へのリンク

- [いらすとや](#) (おみくじの検索結果)
- [音楽研究所](#) (森のくまさん)

## 素材

- ピアノ



# アップロードする

The image shows a web browser window with the URL `poppy.fwex.tohoku-gakuin.ac.jp/progintro/upload_js.html`. The page title is "プログラミング概論" and the content is "JavaScript実行用". The form contains the following fields:

- 学生番号:
- パスワード:
- HTMLファイル (.html):  選択されていません
- 

The file explorer window shows the "Documents" folder with the following files and folders:

名前	更新日時	種類
Officeのカスタムテンプレート	2022/05/24 9:47	ファイル フォルダー
Zoom	2023/11/27 15:11	ファイル フォルダー
piano	2022/07/19 17:43	Chrome HTML D

The file name is "piano" and the type is "HTML Document". The "開く(O)" button is highlighted.

# 打ち込みミスの有無を確認

```
poppy.fwex.tohoku-gakuin.ac.jp x +
← → C 保護されていない通信 | poppy.fwex.tohoku-gakuin.ac.jp/progintro/
OK!: <!DOCTYPE html>
OK!: <html>
OK!: <head>
OK!: <meta charset="utf-8">
OK!: <meta name="viewport" content="width=320, user-scalable=no">
OK!: <link rel="stylesheet" href="common.css" type="text/css" media="all">
OK!: </head>
OK!: <body>
OK!: <canvas id="cvs" width="320" height="320"></canvas>
OK!: <script type="text/javascript">
OK!: var cv = document.getElementById("cvs");
OK!: var ctx = cv.getContext("2d");
OK!: img = new Image();
OK!: img.onload = function() {
OK!:   ctx.drawImage(img, 0, 0, 320, 320);
OK!: };
OK!: img.src = "piano.png";
OK!: cv.addEventListener("touchstart", touchHandler, false);
OK!: var sound = [];
OK!: var filename = ["C3", "D3", "E3", "F3", "G3", "A3", "B3"];
OK!: for (i = 0; i < filename.length; i++) {
OK!:   sound[i+1] = new Audio("sound/"+filename[i]+".mp3");
OK!: }
OK!: function touchHandler(e) {
OK!:   x = e.touches[0].clientX;
OK!:   y = e.touches[0].clientY;
OK!:   if (x < 45)
OK!:     sound[1].play();
OK!:   else if (x < 90)
OK!:     sound[2].play();
OK!:   else if (x < 135)
OK!:     sound[3].play();
OK!:   else if (x < 180)
OK!:     sound[4].play();
OK!:   else if (x < 225)
OK!:     sound[5].play();
OK!:   else if (x < 270)
OK!:     sound[6].play();
OK!:   else
OK!:     sound[7].play();
OK!:   e.preventDefault();
OK!: }
OK!: </script>
OK!: </body>
OK!: </html>
```

すべての行がOK!に  
なっているか確認する

アップロードが完了しました。

```
poppy.fwex.tohoku-gakuin.ac.jp x +
← → C 保護されていない通信 | poppy.fwex.tohoku-gakuin.ac.jp/progintro/
OK!: <!DOCTYPE html>
OK!: <html>
OK!: <head>
OK!: <meta charset="utf-8">
OK!: <meta name="viewport" content="width=320, user-scalable=no">
OK!: <link rel="stylesheet" href="common.css" type="text/css" media="all">
OK!: </head>
OK!: <body>
OK!: <canvas id="cvs" width="320" height="320"></canvas>
OK!: <script type="text/javascript">
OK!: var cv = document.getElementById("cvs");
OK!: var ctx = cv.getContext("2d");
OK!: img = new Image();
OK!: img.onload = function() {
OK!:   ctx.drawImage(img, 0, 0, 320, 320);
Alert!:
OK!: };
OK!: img.src = "piano.png";
OK!: cv.addEventListener("touchstart", touchHandler, false);
OK!: var sound = [];
OK!: var filename = ["C3", "D3", "E3", "F3", "G3", "A3", "B3"];
OK!: for (i = 0; i < filename.length; i++) {
OK!:   sound[i+1] = new Audio("sound/"+filename[i]+".mp3");
OK!: }
OK!: function touchHandler(e) {
OK!:   x = e.touches[0].clientX;
OK!:   y = e.touches[0].clientY;
OK!:   if (x < 45)
OK!:     sound[1].play();
OK!:   else if (x < 90)
OK!:     sound[2].play();
OK!:   else if (x < 135)
OK!:     sound[3].play();
OK!:   else if (x < 180)
OK!:     sound[4].play();
OK!:   else if (x < 225)
OK!:     sound[5].play();
OK!:   else if (x < 270)
OK!:     sound[6].play();
OK!:   else
OK!:     sound[7].play();
OK!:   e.preventDefault();
OK!: }
OK!: </script>
OK!: </body>
OK!: </html>
```

Alert!の行があったら  
打ち込みミスがある  
可能性が高い

アップロードが完了しました。

# 自分のスマートフォンで実行する

<http://poppy.fwex.tohoku-gakuin.ac.jp/progintro/js/230710000.html>



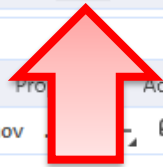
↑  
自分の学生番号  
に変更する

起動してすぐは反応が鈍くても、しばらく  
いじっているうちに改善する可能性があります

# もしうまく動かなかったら

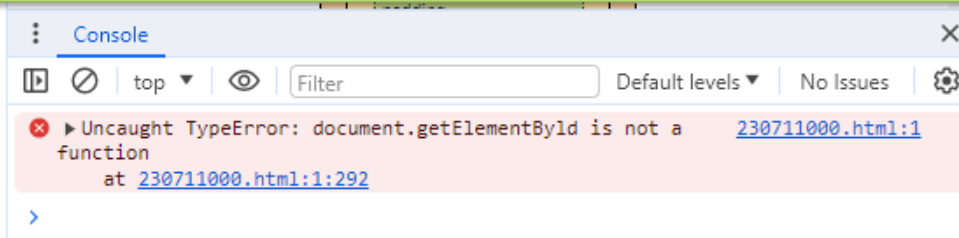


③



④

- ① パソコンでChromeを立ち上げ、アドレスバーに自分のアプリのURLを入力。  
`http://poppy.fwex.tohoku-gakuin.ac.jp/progintro/js/23071XXXX.html`
- ② fnキーを押しながらf12キーを押す。
- ③ リロードする
- ④ エラーがあると赤い×が表示されているはずなので、そこをクリック



- ⑤ ここに出てくる情報を参考に、入力間違いを探す

→ この場合だったら `getElementById` がおかしい 16



**APPINVENTORで改めてピアノを  
つくって作業時間を計測**

# 改めて最初から作り直して時間を計測する

ai2.appinventor.mit.edu/?locale=en#6031483960295424

Gakki\_test Screen1 Add Screen ... Remove Screen Designer Blocks

**Palette**

**User Interface**

- Button
- CheckBox
- DatePicker
- Image
- Label
- ListPicker
- ListView
- Notifier
- PasswordTextBox
- Slider
- Spinner
- TextBox
- TimePicker
- WebView

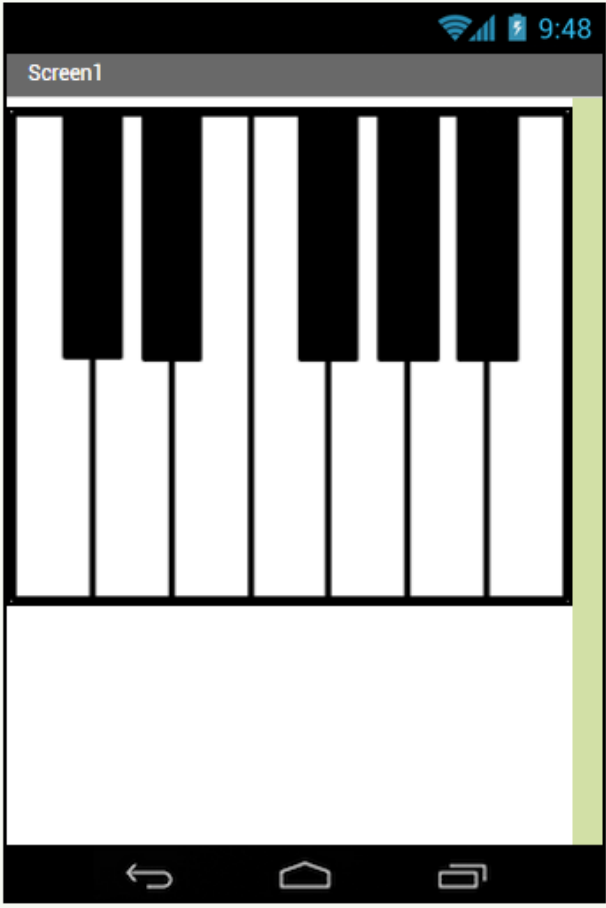
**Layout**

**Media**

**Viewer**

Display hidden components in Viewer

Check to see Preview on Tablet size.



**Components**

- Screen1
  - Canvas1
  - Sound1
  - Sound2
  - Sound3
  - Sound4
  - Sound5
  - Sound6
  - Sound7

Rename Delete

**Properties**

Sound7

MinimumInterval (ms)

500

Source

p\_B3.mp3...

# プロジェクトを新規作成してから正しく動くことを確認するまでにかかった時間

```
when Canvas1 TouchDown
  x y
  do
    if get x < 45
      then call Sound1 .Play
    else if get x < 90
      then call Sound2 .Play
    else if get x < 135
      then call Sound3 .Play
    else if get x < 180
      then call Sound4 .Play
    else if get x < 225
      then call Sound5 .Play
    else if get x < 270
      then call Sound6 .Play
    else call Sound7 .Play
```

分 秒

# JAVASCRIPTとAPPINVENTORで 太鼓をつくって入力時間を比較

# 入力時間の比較を行う

- 最初から最後までではなく、一部の「文字入力」と「ブロック入力」時間のみの比較を行う  
(動作確認時間や修正時間を含まない)

# デザイナー画面での作業時間は含まない

ai2.appinventor.mit.edu/?locale=en#4663554056388608

Designer Blocks

Palette

User Interface

- Button
- CheckBox
- DatePicker
- Image
- Label
- ListPicker
- ListView
- Notifier
- PasswordTextBox
- Slider
- Spinner
- TextBox
- TimePicker
- WebView

Layout

Media

Viewer

Display hidden components in Viewer

Check to see Preview on Tablet size.

Screen1

Components

- Screen1
  - Canvas1
  - Sound1
  - Sound2

Properties

Sound2

MinimumInterval (ms)

500

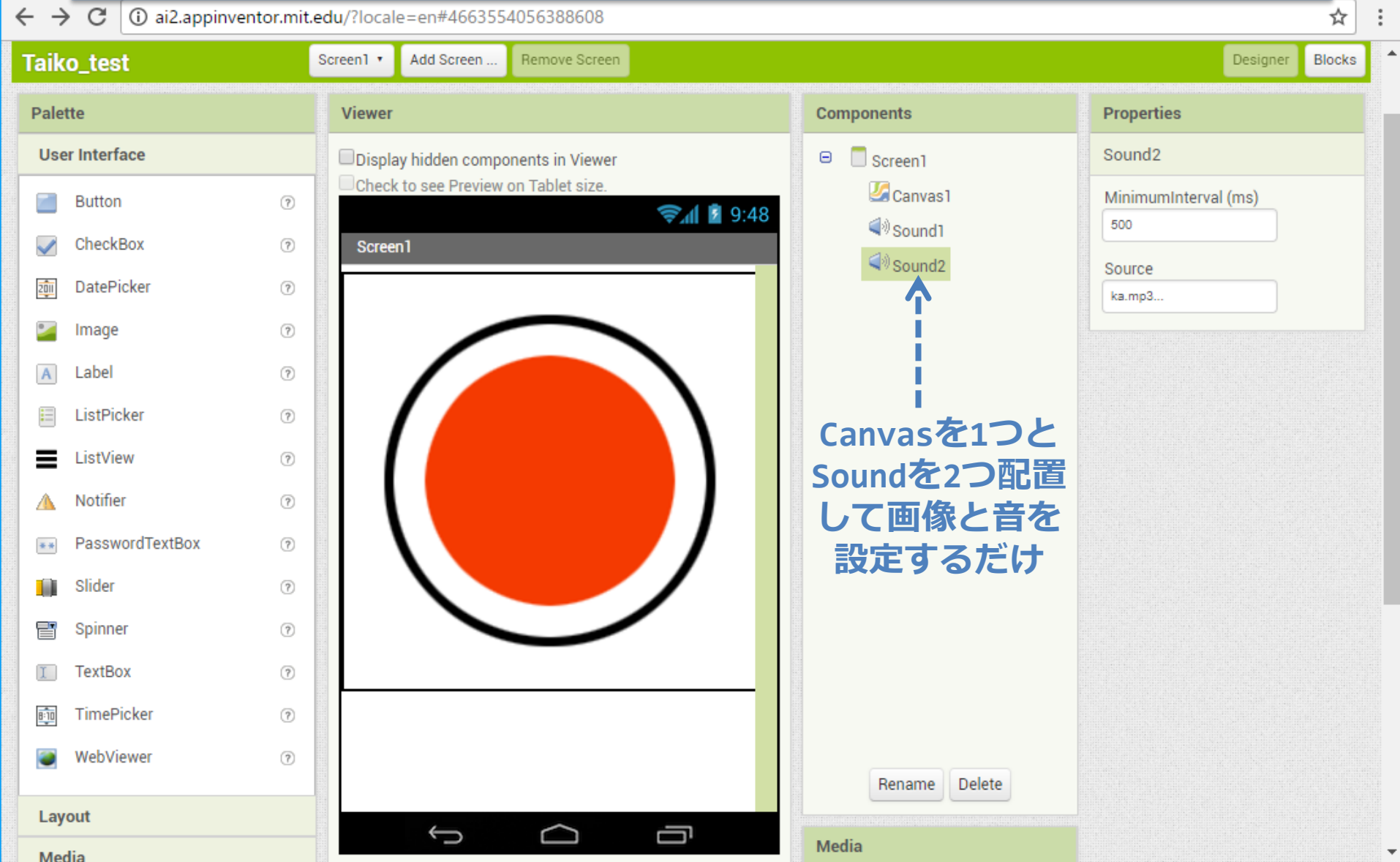
Source

ka.mp3...

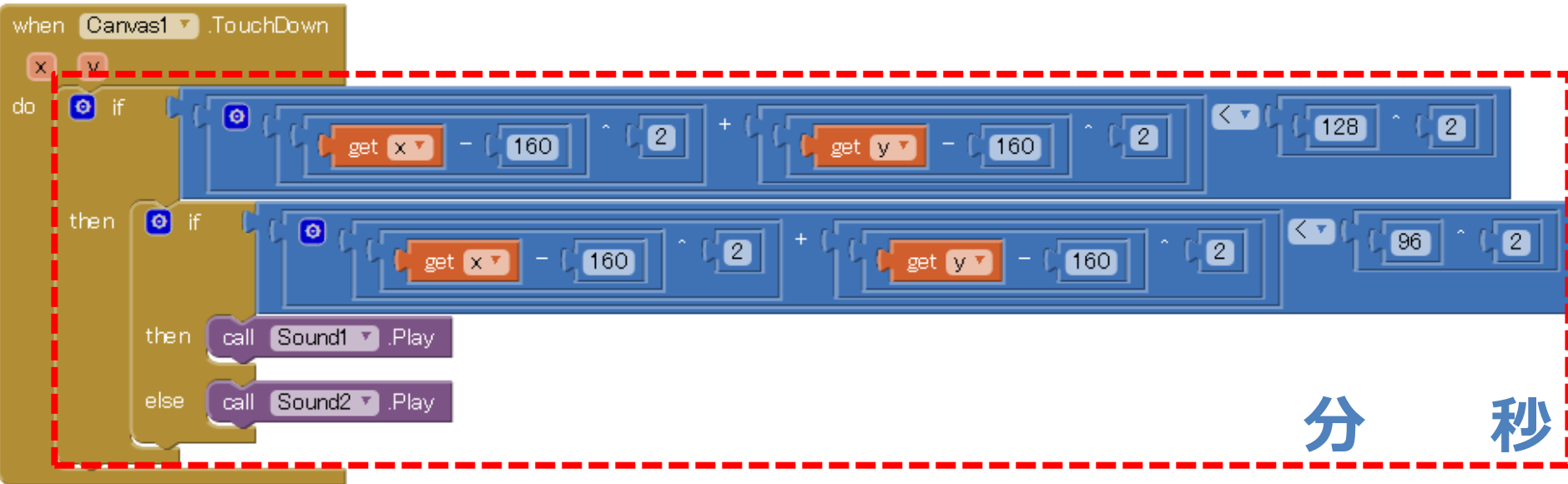
Canvasを1つとSoundを2つ配置して画像と音を設定するだけ

Rename Delete

Media



# 点線内の入力時間をそれぞれ計測する



The image shows a Scratch script for a touch event on Canvas1. The script starts with a 'when Canvas1 TouchDown' block. Inside a 'do' block, there is an 'if' block that checks a mathematical condition:  $(\text{get } x - 160)^2 + (\text{get } y - 160)^2 < 128^2$ . If this condition is true, another 'if' block checks a second condition:  $(\text{get } x - 160)^2 + (\text{get } y - 160)^2 < 96^2$ . If the second condition is true, it calls 'Sound1.Play'. If the second condition is false, it calls 'Sound2.Play'. A red dashed box highlights the two nested 'if' blocks, and the characters '分' (minutes) and '秒' (seconds) are placed to the right of the box.

```
function touchHandler(e) {  
  x = e.touches[0].clientX;  
  y = e.touches[0].clientY;  
  if ((x - 160) * (x - 160) + (y - 160) * (y - 160) < 128 * 128)  
    if ((x - 160) * (x - 160) + (y - 160) * (y - 160) < 96 * 96)  
      sound[1].play();  
    else  
      sound[2].play();  
  e.preventDefault();  
}
```

分 秒

# 授業中に完成しなかったら

- タイムの計測中に授業の終了時刻になってしまったら、タイマーを止め、ここまでのタイムを記録する。時間があるときに作業の続きをおこない、タイムを合算すればよい。
- 授業中にすべての作業が終わらなかった場合は、次回の授業までに作業の続きをおこない、タイムを計測して記録しておくこと。