



付録 4 マウスでお絵かき

	PC	And	iOS
Firefox	○	×	×
Chrome	○	×	×

3.2 で作ったお絵かきアプリはスマホ（タッチ）専用で、PC（マウス）では動きません。マウスでお絵かきするコードは以下です。タッチイベントをマウスイベントに置き換えることと、座標を取得する部分が少し違うだけです。両方のコードを共存させやすいように関数名もマウス用に変えておきました。興味があったら試してみましょう。

参考：マウスでお絵かき [index.html](#)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset='utf-8'>
    <meta name='viewport' content='width=device-width,initial-scale=1'>
    <title>3.2s マウスでお絵かき</title>
    <style>
      #canvas {
        /* Canvas に枠をつける CSS */
        border: solid 1px #000000;
      }
    </style>
  </head>
  <body>
    <!-- Canvas (画用紙・お絵かきセット) を置く -->
    <canvas width='300' height='400' id='canvas'></canvas>
    <script>
      // Canvas を使うためのお決まりの 2 行
      var canvas = document.getElementById('canvas');      // お絵かきセットを得て
      var context = canvas.getContext('2d');                // その筆を得る
      // マウスで描くための 3 個のイベント（開始、描画、終了）
      canvas.addEventListener('mousedown', drawStartM);    // マウスダウン時
      canvas.addEventListener('mousemove', drawM);          // マウスが動いた時
      canvas.addEventListener('mouseup', drawEndM);         // マウスアップ時
      // 各種変数の宣言
      var touching = false;                                // マウスダウン中か否かの変数
      var startX = 0;                                       // 描画開始の X 座標
      var startY = 0;                                       // 描画開始の Y 座標
      var penColor = '#000000';                             // 筆の色
      var penWidth = 5;                                     // 筆の太さ
      // 描画開始の関数 drawStartM (マウス用)
      function drawStartM(event) {
        touching = true;
        // マウスボタンが押された座標（開始座標）を得る
        startX = event.clientX - canvas.offsetLeft;
        startY = event.clientY - canvas.offsetTop;
      }
    </script>
  </body>
</html>
```



```
// 描画の関数 drawM ( マウス用 )
function drawM(event) {
    if (touching == true) { // マウスダウン中なら描画
        // マウスが動いた後の座標 ( 終了座標 ) を得る
        var endX = event.clientX - canvas.offsetLeft;
        var endY = event.clientY - canvas.offsetTop;
        // 開始座標から終了座標に向かって線を引く
        context.beginPath();
        context.moveTo(startX, startY); // 開始座標から
        context.lineTo(endX, endY); // 終了座標へ
        context.strokeStyle = penColor; // 筆の色
        context.lineWidth = penWidth; // 筆の太さ
        context.lineCap = 'round'; // 線の端を丸める
        context.stroke(); // 線を引く
        // 今の終了座標を次の開始座標にする
        startX = endX;
        startY = endY;
    }
}
// 描画終了の関数 drawEndM ( マウス用 )
function drawEndM(event) {
    touching = false;
}
</script>
</body>
</html>
```

付録 5 テキスト読み上げエンジンの設定

3.7 の音声合成アプリを Android スマホで実行するには、日本語に対応した「テキスト読み上げエンジン (TextToSpeech; TTS)」をインストールしておく必要があります。

iOS の場合は TTS のインストールは不要です。

「Play ストア」からインストール

-  **Play ストア** を開きます。
- 検索欄に「Google テキスト読み上げ」と入力して検索します。
-  **インストール** をタップすれば OK です。

※ 「Google テキスト読み上げ」以外の TTS もあります。例えば KDDI の「N2 TTS」、AQUEST の「AquesTalk TTS」などです。特に「N2 TTS」はより滑らかな合成ができて、音声データ (声) もたくさん選べます。Google 製の声に飽きたら試してみましょう。

読み上げエンジンの設定

- **設定** → **言語と入力** → **テキスト読み上げの出力** とたどります。
- 「優先するエンジン」で「●Google テキスト読み上げ」を選びます。
- 「Google テキスト読み上げ」の右にある歯車アイコンをタップ → **音声データをインストール** → **日本語 (日本)** をタップします。
- 「音声セット 1」の右側に下向きの矢印 (ダウンロードボタン) があれば、タップして音声データをインストールします。ゴミ箱のアイコンであれば、音声データは既にインストール済です。
- 「優先するエンジン」のページに戻り、「サンプルを再生」をタップして 『こちらが日本語音声合成のサンプルです』と発声されたら、設定完了です。