徐　丙鉄
近畿大学工学部情報学科

制作するアプリ

五線譜上をクリックして作曲し，演奏できるアプリ

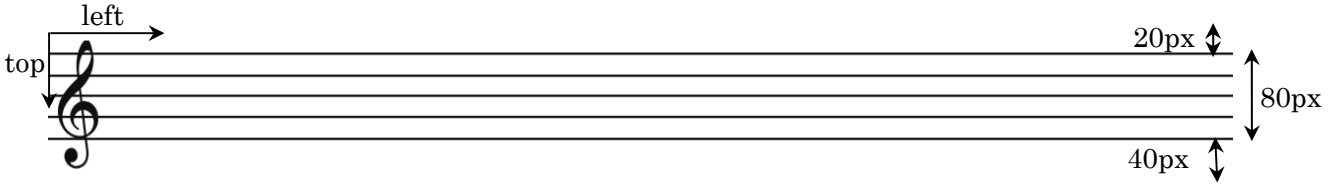
１．準備

五線譜の画像，音符の画像，サウンドファイル，作業用 html ファイルは圧縮してネット上に置いています。以下のアドレスにアクセスして，デスクトップに解凍して下さい。

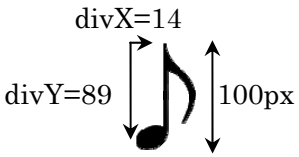
`http://buturi.heteml.jp/js/SoundMusicScore.zip`

imags フォルダ

5senfu.png　1800px×140px，5 線の間隔 20px



onpu8.png　55px×100px



オタマジャクシの●の中心は
音符画像の左上頂点から，
右へ 14px，下へ 89px

sounds フォルダ

A0.wav, A1.wav, A2.wav, ・・・ , A6.wav
B0.wav, B1.wav, B2.wav, ・・・ , B6.wav
・　　・　　・　　・　　・
G0.wav, G1.wav, G2.wav, ・・・ , G6.wav

soundMusicScore.html

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8" />
<title>Sound Music Score</title>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js">
</script>
<style>
html, body{ margin: 0; padding: 0}

#score{
        margin: 100px 10px; padding: 0;
        width: 1800px; height: 140px;
        background-image: url("./imgs/5senfu.png");
}
```

```
#playBtn, #clearBtn{
      width: 80px; height: 30px; margin: 10px;
}

</style>
<script>
//--- 五線譜上の音符の位置 onkai と サウンドデータの対応付け ----------------------
// onkai  0, 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19
// code  A4,G4,F4,E4,D4,C4,B3,A3,G3,F3,E3,D3,C3,B2,A2,G2,F2,E2,D2,C2
var soundCode = ['A4','G4','F4','E4','D4','C4','B3','A3','G3','F3',
                    'E3','D3','C3','B2','A2','G2','F2','E2','D2','C2'];
</script>
</head>
<body>

<div id="score"></div>

<button id="playBtn">演奏</button><button id="clearBtn">クリア</button>

</body>
</html>
```

２．jQuery：ブラウザ用の JavaScript ライブラリで，DOM 要素の取得が CSS のセレクタ形式で便利

（1） DOM tree が構築されたタイミング（DOMContentLoaded）で JavaScript を実行する

```
$(function(){
    //ここに実行したいプログラムを書く
});
```

（2）DOM 要素の取得

```
$("#score")          ⇔    document.getElementById("score")
```

注意：取得された要素は jQuery オブジェクトで，DOM 要素に様々な機能が追加されている。

```
$(".onpu")           ⇔    document.getElementsByClassName("onpu")
```

（3）イベント処理

```
//五線譜がクリックされたら，クリックした座標（page 座標）を警告ウインドウで表示
$("#score").click( function(e){
    alert( "x="+ e.pageX + ", y=" + e.pageY );
});
```

なお，jQuery イベントの座標 pageX，pageY はドキュメントの左上が原点である。

ステップ１．五線譜をクリックしたら，その座標をコンソールに出力

Chrome
F12 キーを押すと Developer Tools が起動する。Developer Tools のメニューから Console をクリック

今後は、この枠内の部分のソースを表示

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8" />
<title>Sound Music Score</title>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
<style>
html,body{ margin: 0; padding: 0}

#score{
        margin: 100px 10px; padding: 0;
        width: 1800px; height: 140px;
        background-image: url("./imgs/5senfu.png");
}

#playBtn, #clearBtn{
        width: 80px; height: 30px; margin: 10px;
}

</style>
<script>
//--- 五線譜上の音符の位置 onkai と サウンドデータの対応付け ----------------------
// onkai  0, 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19
// code  A4,G4,F4,E4,D4,C4,B3,A3,G3,F3,E3,D3,C3,B2,A2,G2,F2,E2,D2,C2
var soundCode = ['A4','G4','F4','E4','D4','C4','B3','A3','G3','F3',
                 'E3','D3','C3','B2','A2','G2','F2','E2','D2','C2'];

$(function(){

        //五線譜をクリックしたら，その座標をコンソールに出力
        $("#score").click( function(e){
                console.log("x=" + e.pageX + ", y=" + e.pageY );
        });

});
</script>
</head>
<body>
<div id="score"></div>
<button id="playBtn">演奏</button><button id="clearBtn">クリア</button>
</body>
</html>
```

この部分を追加

A

ステップ２．五線譜をクリックすると，その位置に音符を表示する

ステップ２a.　音符を HTML で記述し表示する

（以下，関連する部分のみ）

```
<style>
#score{
        margin: 100px 10px; padding: 0;
        width: 1800px; height: 140px;
        background-image: url("./imgs/5senfu.png");
        position: relative;
}
</style>
</head>
<body>

<div id="score">
<div class="onpu" style="position: absolute; left: 200px; top: 0px;">
<img src="./imgs/onpu8.png" height="100">
</div>
</div>

</body>
```

onpuDiv

CSS の position プロパティ（static, relative, absolute, fixed）

position: absolute
位置指定有効な最も近い世代の祖先要素（親ボックス）からの相対位置。位置は祖先要素からの top, left, right, bottom 等で指定する。上記の例では，#score が親ボックスで，.onpu は#score の左上を原点とする座標で，位置を指定する。

ステップ２b. 上記 HTML をプログラムで表現し，クリック event の座標で left と top を指定する

```
var onpuDiv = $("<div/>").attr({class: "onpu"});
onpuDiv.html(  '<img src="./imgs/onpu8.png" height="100">' );
onpuDiv.css( { position: "absolute", left: (e.pageX - 10),
                        top: (e.pageY - 100) } ).appendTo("#score");
```

上記コードを前のページの [A] に挿入

・JavaScript では，文字列を囲むのはシングルクオテーション(')でもダブルクオテーション(")でもよい。クオテーションが入れ子になるときは，上記の 2 行目を参考にする。
・JavaScript の基本データ型：number, string, boolean
・JavaScript での配列と連想配列

配列	var array = [“近大”, 50];	array[0];	array[1];
連想配列（オブジェクト）	var person = { name: “近大”, age: 50 };	person.name;	person[“name”]

ステップ２c. クリックした位置がオタマジャクシの●の中心になるように調整

```
var x0 = 10;        var y0 = 100;
var divX = 14;      var divY = 89;

//五線譜がクリックされたらその位置にオタマジャクシの●がくるように音符を追加
$("#score").click( function(e){
      console.log("x=" + e.pageX + ", y=" + e.pageY );

      var onpuDiv = $("<div/>").attr({class: "onpu"});
      onpuDiv.html(   '<img src="./imgs/onpu8.png" height="100">' );
      onpuDiv.css( { position: "absolute", left: (e.pageX - x0 - divX),
                     top: (e.pageY - y0 - divY) } ).appendTo("#score" );

});
```

ステップ２d. オタマジャクシの●が五線の適切な高さになるように調整（高さの量子化）

```
var x0 = 10;        var y0 = 100;
var divX = 14;      var divY = 89;
var dy = 10;        //五線の間隔 20px なので，音程の 1 度は 10px

//五線譜がクリックされたら音符を追加
$("#score").click( function(e){
        console.log("x="+ e.pageX + ", y=" + e.pageY );
        var x = e.pageX;  var y = e.pageY;
        y = y – y0;
        var onkai = Math.round( y/dy );     y = onkai*dy;
                        console.log( "onkai=" + onkai + " ,y=" + y );

        var onpuDiv = $("<div/>").attr({class: "onpu" });
                onpuDiv.html(   '<img src="./imgs/onpu8.png" height="100">' );
                onpuDiv.css( { position:"absolute", left: (e.pageX - x0 - divX),
                                top: (y   - divY) } ).appendTo("#score");
});
```

ステップ２e. オタマジャクシを横に一定間隔に配置されるように調整

```
var x0 = 10;        var y0 = 100;
var divX = 14;      var divY = 89;
var dx = 55;        var dy = 10;

//五線譜がクリックされたら音符を追加
$("#score").click( function(e){
```

```
            console.log("x="+ e.pageX + ", y=" + e.pageY );
            var x = e.pageX;  var y = e.pageY;
            x = x – x0;          y = y – y0;
            var n = Math.round( x/dx );          x = n*dx;
                  console.log( "n=" + n + " ,x=" + x );
            var onkai = Math.round( y/dy ); y = onkai*dy;
                  console.log( "onkai=" + onkai + " ,y=" + y );

            var onpuDiv = $("<div/>").attr({class: "onpu"});
                  onpuDiv.html(   '<img src="./imgs/onpu8.png" height="100">' );
            onpuDiv.css( { position:"absolute", left: (x - divX),
                        top: (y - divY) } ).appendTo("#score");
});
```

ステップ２f.　n でその音の再生タイミングを指定し，onkai で再生 sound ファイルを指定するので，onpuDiv に n の値と onkai の値を持たせる。さらに，オタマジャクシの●付近に box 領域を設定し，そこをクリックすると音符が削除されるようにする。

```
<style>
.onpu span{
      display: block;
      position: absolute;
      color: blue;
}

/* n 拍目 */
.onpu .n{
      left: 4px;
      top: 10px;
}

/* onkai 音の高さ */
.onpu .onkai{
      left: 4px;      top: 25px;
}

/* オタマジャクシの● に重ねてクリック領域を設定，テスト段階ではボーダーを表示 */
.clickArea{
      border: 1px solid red;
      width: 55px;    height: 20px;
      left: -14px;    top: 76px;
}
</style>
```

```
<script>
        var onpuDiv = $("<div/>").attr({class: "onpu"});
        onpuDiv.html(   '<img src="imgs/onpu8.png" height="100">'
                        + '<span class="clickArea">S</span>'
                        + '<span class="n">' + n + '</span>'
                        + '<span class="onkai">' + onkai + '</span>' );

        //onpuDiv 内の span.clickArea がクリックされると onpuDiv が消去される
                $("span.clickArea").on("click", function (e) {
                        e.stopPropagation(); $(this).parent().remove();
                })

</script>
```

ステップ２g. clear ボタン(音符の削除)の実装

```
//クリアボタンがクリックされたら音符削除
$("#clear").click(function(){

        $(".onpu").each( function(){
                $(this).remove();
         });

});
```

ステップ３．play ボタン（演奏）の実装

ステップ３a. サウンドファイルの再生方法

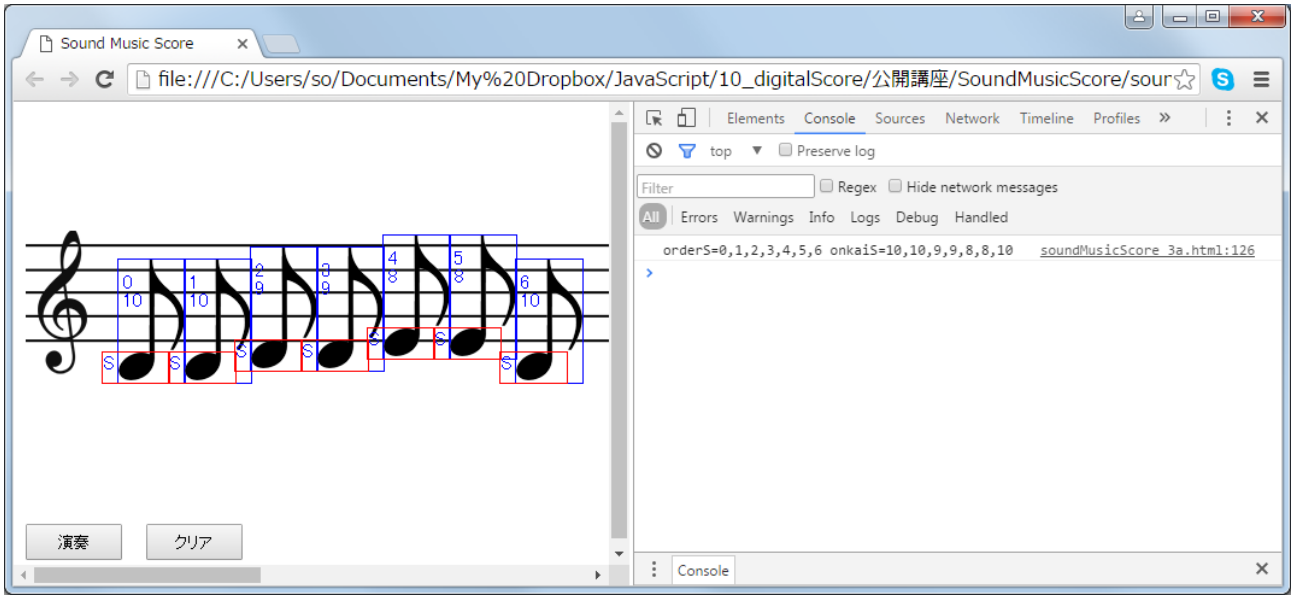
手順:onkai に対応するサウンドファイルの url を指定して Audio オブジェクトを生成し,n 拍目に play を指示する

```
//--- onkai と サウンドデータの対応付け ------------------------------------------
// onkai  0, 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19
// code  A4,G4,F4,E4,D4,C4,B3,A3,G3,F3,E3,D3,C3,B2,A2,G2,F2,E2,D2,C2
var soundCode = ['A4','G4','F4','E4','D4','C4','B3','A3','G3','F3',
                 'E3','D3','C3','B2','A2','G2','F2','E2','D2','C2'];

function playOnkai(onkai, n){       //onkai= 10, n=5
    var url = "./sounds/" + soundCode[onkai] + ".wav";
    var audioObj = new Audio(url);
    var delayTime = 300+500*n ;     // 演奏開始までの待ち時間(300ms, 1 拍 500ms)
    setTimeout( function(){ audioObj.play(); }, delayTime);
}

playOnkai(10, 3);
```

ステップ３ｂ．採譜機能の実装

orderS: その音符が何拍目かを表す n の配列
onkaiS: 音符の音階（0,1,2,3,・・・）onkai の配列

```
var orderS = []; var onkaiS = [];
//演奏ボタンがクリックされたら採譜する
$("#playBtn").click( function(){
        orderS = []; onkaiS = [];       //リセット
        //採譜
        $(".onpu").each(function(){
                orderS.push( $(this).children(".n").text() );
                onkaiS.push( $(this).children(".onkai").text() );
         });
        console.log("orderS=" + orderS + " onkaiS=" + onkaiS);
});
```



ステップ３c．演奏機能の実装

```
<style>
/* テスト段階では onpuDiv のボーダーを確認のために表示 */
.onpu{
        border: 1px solid blue;
}

/* onpuDiv の子孫の span 要素を block 要素にし，絶対座標で位置を決める，文字色は青 */
.onpu span{
        display: block;
        position: absolute;
        color: blue;
```

```
}

/* n 拍目 */
.onpu .n{
        left: 4px;      top: 10px;
}

/* onkai 音の高さ */
.onpu .onkai{
        left: 4px;      top: 25px;
}

/* オタマジャクシの● に重ねてクリック領域を設定，テスト段階ではボーダーを表示 */
.clickArea{
        border: 1px solid red;
        width: 55px;    height: 25px;
        left: -14px;    top: 76px;
}
</style>
<script>
//--- onkai と サウンドデータの対応付け ----------------------
// onkai  0, 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19
// code  A4,G4,F4,E4,D4,C4,B3,A3,G3,F3,E3,D3,C3,B2,A2,G2,F2,E2,D2,C2
var soundCode = ['A4','G4','F4','E4','D4','C4','B3','A3','G3','F3',
                                'E3','D3','C3','B2','A2','G2','F2','E2','D2','C2'];

var x0 = 100; var y0 = 100;     //五線譜の基準点のページ座標
var dx = 55;  var dy = 10;      //dx：水平方向の量子化単位，dy：垂直方向の量子化単位

//クリック event を引数にして音符画像を含む onpuDiv を追加する関数の定義
function putOnpu( event ){              // event:クリックイベント
        var x = event.pageX;    var y = event.pageY;

if( x > x0 ){   //クリックした x 座標が x0 以上のときに，音符を追加する
        var divX = 14; var divY = 89;   //onpuDiv の表示位置合わせのパラメータ

// 五線譜上の音の量子化した位置，
// n：左端からの音符の量子化した位置で何拍目かを表す
// onkai：五線譜上の音の量子化した高さで，0 は高いファ，3 が高いド
        var n = Math.round((x - x0)/dx);        x = x0 + n*dx;
        var onkai = Math.round((y - y0)/dy);    y = y0 + onkai*dy;

//オタマジャクシのイメージと n と onkai を値として持つ div
// <div class="onpu" name=3><img src="./imgs/8bu.png" height="100">
// <span class="clickArea">S</span><span class="n">n</span><span class="onkai">onkai</span>
```

```
// </div>
        var onpuDiv = $("<div/>").attr({ class: "onpu", name: (n) });
        onpuDiv.html(   '<img src="./imgs/onpu8.png" height="100">'
                      + '<span class="clickArea">S</span>'
                      + '<span class="n">' + n + '</span>'
                      + '<span class="onkai">' + onkai + '</span>'  );

onpuDiv.css( { position: "absolute", left: (x-divX), top: (y-divY) } ).appendTo("#score");

//onpuDiv はクリックされると消去される。なお, 動的に追加した要素に対するイベント処理は on
        onpuDiv.on("click", "span.clickArea", function (e) {
                e.stopPropagation();
                $(this).parent().remove();
        })
}       //end if --------------

}

//--- 演奏 ----------------------------------------
//指定した onkai を n 拍後に鳴らす
function playOnkai(onkai, n){           //例：onkai = 10, n = 2

        var url = "./sounds/" + soundCode[onkai] + ".wav";      //例 url = "./sounds/C3.wav"
        var audioObj = new Audio(url);
        var delayTime = 300 + 500*n;            // 演奏開始までの待ち時間 300 ms, 1 拍 500ms

        var filter = 'name="' + n +'"';         //filter = 'name="2"'
        //class が onpu の要素で name 属性が n である要素(onpuDiv)
        var onpuDiv = $( ".onpu["+ filter + "]" );
        //alert( onpuDiv.attr("name") );
        setTimeout(
                function(){
                        //サウンドが再生されるときに onpuDiv の枠線を描く
                        onpuDiv.css( "border" , "solid 1px red" );
                        audioObj.play();
                        //100ms 後に枠線を削除する
                        setTimeout( function(){ onpuDiv.css( "border" , "none" );} , 100);
                },
                delayTime
        );
}
//--- 採譜データを元に music を演奏する ------------------------
function playMusic( orderS, onkaiS ){

        for( var i = 0; i < orderS.length; i++ ){
```

```
                        var n = orderS[i];
                        var onkai = onkaiS[i];
                        playOnkai(onkai, n);
        }

}

$(function(){

        //五線譜がクリックされたら音符を追加
        $("#score").click( function(e){
                putOnpu(e);
        });

        //clear ボタンをクリックすると全ての音符を削除
        $("#clearBtn").click(function(){

                $(".onpu").each(function(){
                        $(this).remove();
                });

        });

        var orderS = []; var onkaiS = [];

        //演奏ボタンがクリックされたら採譜する
        $("#playBtn").click( function(){
                orderS = []; onkaiS = [];
                //採譜
                $(".onpu").each(function(){
                        orderS.push( $(this).children(".n").text() );
                        onkaiS.push( $(this).children(".onkai").text() );
                 });
                console.log("orderS=" + orderS + " onkaiS=" + onkaiS);
                playMusic( orderS, onkaiS );
        });

});
</script>
</head>
<body>
<div id="score"></div>
<button id="playBtn">演奏</button><button id="clearBtn">クリア</button>
</body>
</html>
```

完成版

```
<!DOCTYPE html>
<html lang="ja">
<head>
<meta charset="UTF-8" />
<title>Sound Music Score</title>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
<style>
html,body{ margin: 0; padding: 0}

#score{
        margin: 100px 10px;     padding: 0;
        width: 1800px;          height: 140px;
        background-image: url("imgs/5senfu.png");
        position: relative;
}

/* onpuDiv の子孫の span 要素を block 要素にし，絶対座標で位置を決める，文字色は青 */
.onpu span{
        display: block;
        position: absolute;
        color: blue;
}

/* n 拍目 */
.onpu .n{
        left: 4px;      top: 10px;
}

/* onkai 音の高さ */
.onpu .onkai{
        left: 4px;      top: 25px;
}

/* オタマジャクシの● に重ねてクリック領域を設定，テスト段階ではボーダーを表示 */
.onpu .clickArea{
        border: 1px solid red;
        width: 55px;    height: 25px;
        left: -14px;    top: 76px;
}

/* 完成版では透明にして見えなくする */
.onpu span{
        opacity: 0;
}
```

```
#playBtn, #clearBtn{
        width: 80px;    height: 30px;   margin: 10px;
}
</style>
<script>
//--- onkai と サウンドデータの対応付け ----------------------
// onkai  0, 1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16,17,18,19
// code  A4,G4,F4,E4,D4,C4,B3,A3,G3,F3,E3,D3,C3,B2,A2,G2,F2,E2,D2,C2
var soundCode = ['A4','G4','F4','E4','D4','C4','B3','A3','G3','F3','E3','D3','C3',
                                                'B2','A2','G2','F2','E2','D2','C2'];

var x0 = 10; var y0 = 100;      //五線譜の基準点のページ座標
var XX0 = 80;                   //ト音記号を避けて音符を追加
var dx = 55;  var dy = 10;      //dx：水平方向の量子化単位，dy：垂直方向の量子化単位

//クリック event を引数にして
//音符画像と音の高さ onkai と n 拍目の情報を含む onpuDiv を追加する関数の定義 -------------
function putOnpu( event ){              // event:クリックイベント
        var x = event.pageX;    var y = event.pageY;

  if( x > XX0 ){        //ト音記号を避ける。x が 80 以上のときに，音符を追加する -------
        x = x - x0;     y = y - y0;
        var divX = 14;   var divY = 89; //onpuDiv の表示位置合わせのパラメータ

// 五線譜上の音の量子化した位置
// n：左端からの音符の量子化した位置で何拍目かを表す
// onkai：五線譜上の音の量子化した高さで，0 は高いラ，3 が高いミ
        var n = Math.round( (x-XX0)/dx );               x = XX0 + n*dx;
        var onkai = Math.round( y/dy );                 y = onkai*dy;

//オタマジャクシのイメージと n と onkai を値として持つ div
// <div class="onpu" name= n><img src="./imgs/8bu.png" height="100">
// <span class="clickArea">S</span><span class="n">n</span><span class="onkai">onkai</span>
// </div>
        var onpuDiv = $("<div/>").attr({ class:"onpu", name: (n) });
        onpuDiv.html(   '<img src="./imgs/onpu8.png" height="100">'
                      + '<span class="clickArea">S</span>'
                      + '<span class="n">' + n + '</span>'
                      + '<span class="onkai">' + onkai + '</span>'  );

        onpuDiv.css( { position: "absolute", left: (x-divX),
                                            top: (y-divY) } ).appendTo("#score");

//onpuDiv は .clickArea がクリックされると消去される。 動的に追加した要素に対するイベント登録は on
```

```
        onpuDiv.on("click", "span.clickArea", function (e) {

                e.stopPropagation();    //イベントのバブリングを止める
                $(this).parent().remove();

        })
  }     //end if ------------------------------------------------------------

}

//--- 演奏 ----------------------------------------
//指定した onkai を n 拍後に鳴らす関数
function playOnkai(onkai, n){           //例：onkai = 10, n = 2

        var url = "./sounds/" + soundCode[onkai] + ".wav";      //例 url = sounds/C3.wav
        var audioObj = new Audio(url);
        var delayTime = 300 + 500*n;            // 演奏開始までの待ち時間 300 ms, 1 拍 500ms

        var filter = 'name="' + n +'"';         //filter = 'name="2"'
        //class が onpu の要素で name 属性が n である要素(onpuDiv)
        var onpuDiv = $( ".onpu["+ filter + "]" );
        //alert( onpuDiv.attr("name") );
        setTimeout(
                function(){
                        //サウンドが再生されるときに onpuDiv の枠線を描く
                        onpuDiv.css( {border: "solid 1px red"} );
                        audioObj.play();
                        //100ms 後に枠線を削除する
                        setTimeout( function(){ onpuDiv.css( {border: "none"} );} , 100);
                },
                delayTime
        );
}

//--- 採譜データを元に演奏する ------------------------
function playMusic( orderS, onkaiS ){

        for( var i = 0; i < orderS.length; i++ ){
                        var n = orderS[i];
                        var onkai = onkaiS[i];
                        playOnkai(onkai, n);
        }

}
```

```
$(function(){

	//五線譜がクリックされたら音符を追加
	$("#score").click( function(e){
		putOnpu(e);
	});

	//clear ボタンをクリックすると全ての音符を削除
	$("#clearBtn").click(function(){

		$(".onpu").each(function(){
			$(this).remove();
		});

	});

	//採譜データを納める配列変数
	var orderS = []; var onkaiS = [];

	//演奏ボタンがクリックされたら採譜する
	$("#playBtn").click( function(){
		orderS = []; onkaiS = [];	//初期化

		//採譜
		$(".onpu").each( function(){
			orderS.push( $(this).children(".n").text() );
			onkaiS.push( $(this).children(".onkai").text() );
		 });
		//開発時確認用
		console.log( "orderS=" + orderS + " onkaiS=" + onkaiS);

		playMusic( orderS, onkaiS );
	});

});

</script>
</head>
<body>
<div id="score"></div>
<button id="playBtn">演奏</button><button id="clearBtn">クリア</button>
</body>
</html>
```