



INSTITUTO TECNOLÓGICO DE AERONÁUTICA

**PROJETO GRAMÁTICA MUSICAL
CTC – 34**

Dylan Nakandakari Sugimoto
Gabriel Adriano de Melo

São José dos Campos
20/11/2017

Projeto Gramática Musical

O que é?

Esse é um projeto que pretende implementar uma gramática de fácil aprendizado e escrita na forma de um aplicativo web para que os usuários possam utilizá-lo para, escrevendo músicas usando a gramática, obter o desenho da partitura da música escrita.

Metologia

Para isso, escolheu-se utilizar a linguagem javascript devido à portabilidade para várias plataformas (notebook, celulares, tablets, etc). Assim, escolhida a linguagem procurou-se bibliotecas que já fizessem a parte de desenhar uma partitura e que estivessem acopladas de alguma forma com o MIDI.js para que o aplicativo desse projeto pudesse disponibilizar uma amostra sonora da entrada do usuário. Assim, encontrou-se a biblioteca abc.js que realiza tanto a parte de desenho da partitura quanto a parte de gerar um arquivo de mídia sonora.[1]

Modificou-se uma demo[2] da biblioteca para integrar a gramática do projeto, ou seja, modificou-se essa demo para que a entrada do usuário seja na gramática desse projeto e a saída fosse o desenho da partitura e o arquivo de mídia sonora. Para isso, teve-se que traduzir a gramática desse projeto para o abc notation de forma que pudéssemos usar demo para gerar o desenho e o arquivo sonoro. Essa tradução utiliza o PEG.js [3] que é uma biblioteca para escrever o parser de gramáticas. Por fim, realizou-se a integração do PEG.js com o abc.js resultando no aplicativo desejado nesse projeto.

Resumidamente, definiu-se uma gramática utilizando-se o PEG.js, gerando o parser, que construía uma estrutura de dados com as listas das notas musicais. Programou-se um escritor para a notação ABC, utilizando-se a estrutura de dados gerada anteriormente. Integrou-se o framework abc.js para renderizar as notas musicais.

Descrição da Linguagem

A linguagem definida teve como objetivo principal ser a mais simples possível, para possibilitar o fácil uso por iniciantes musicais. A sua escrita tende a imitar a fonética das próprias notas, por meio da técnica do solfejo, que consiste na leitura da partitura cantando cada nota, de acordo com a sua duração e altura.

Assim, as notas são escritas por seus próprios nomes (sem acentos) do, re, mi, fa, sol, la, si, e podem ter seu tempo dobrado utilizando-se doo, ree, mii, faa, sool, laa, sii, pausa. Também pode-se especificar o tempo de cada nota por uma fração após a nota. As notas podem ser deslocadas uma oitava acima escrevendo-se em maiúsculo. Um exemplo simples da sua utilização está demonstrado na Figura 1, com um texto de fácil leitura da música dó ré mi fá.

Na interface do usuário, observada nas Figuras 1 e 2 há uma caixa de texto para a escrita da música e vários formatos de saída, um texto em abc, uma imagem da partitura renderizada, além de opções de escutar e baixar os arquivos de música gerados.



Figura 1: Demonstração da escrita da música.

Há comandos especiais para aumentar ou diminuir o tempo padrão das notas, respectivamente “maisLento” e “maisRápido”, que dobram ou diminuem o tempo padrão da nota pela metade, a partir de todas as próximas notas. Além do controle sobre o tempo, pode-se mudar a oitava padrão das notas com os comandos “sobeOitava” e “desceOitava”. Observa-se a utilização desses comandos na Figura 2.

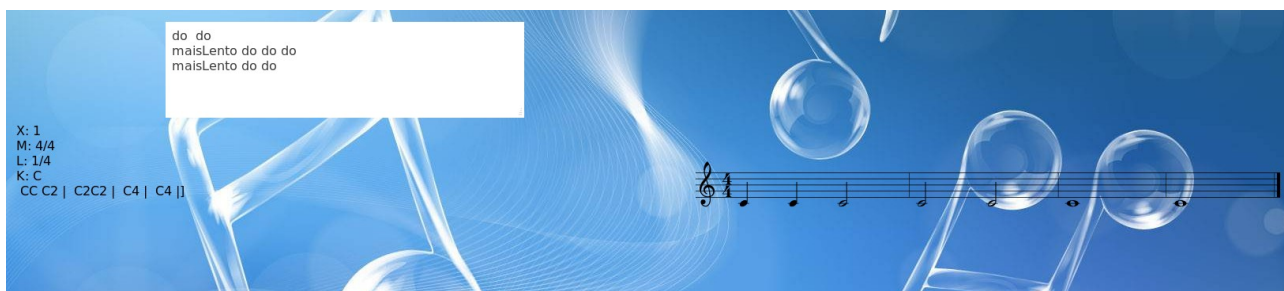


Figura 2: Utilização do comando maisLento.

Pode-se definir o tempo individual de cada nota por meio de uma fração após a nota. Também pode-se utilizar sustenidos e bemois por meio dos simbolos # e b após a nota, antes da fração, sem espaços. Assim, exemplos de notas válidas seriam *do# solb Faa3/2 Sol#1/2 Re/2*. No caso de frações do tipo *1/x*, pode-se omitir o *1*, ficando apenas com */x*, como no caso de *Re/2*.

A música também apresenta um cabeçalho, com elementos opcionais, que devem ser definidos na seguinte ordem: Titulo, Compositor, Tempo, Nota, Velocidade, Escala, Juntos. Eles devem ser escritos exatamente como expresso acima e devem ser seguidos de dois pontos. Os atributos Titulo e Compositor aceitam qualquer cadeia. Os atributos Tempo e Nota precisam de uma fração, que indicam o tempo de cada compasso e o tempo padrão de uma nota. A velocidade é expressa por meio de uma fração que indica a nota e um sinal de igual precedido de um inteiro, que indica batidas por minuto (bpm dessa nota). A escala segue a notação de acordes A, B, C, D O atributo Juntos indica quais instrumentos devem ser desenhados juntos em um mesmo pentagrama, deve-se utilizar parentesis para definir o agrupamento. Utilizando-se *(1 2) (3)*, os dois primeiros instrumentos seriam desenhador juntos em um mesmo pentagrama, enquanto o terceiro seria desenhado em outro pentagrama.

Os intrumentos definidos por padrão são o piano, o violino, o celo, a flauta e a bateria. Outros instrumentos não reconhecidos serão mapeados para o piano, mas a extensão é fácil para a adição de novos instrumentos. Utiliza-se a diretiva *Intrumento:* para criar um novo pentagrama para a voz. Um exemplo de utilização extensiva desses comandos está disponível na Figura 3.

Titulo: Canon em Ré Maior
Compositor: Johann Pachelbel
Tempo: 4/4
Nota: 1/4
Velocidade: 1/4=45
Escala: D

Instrumento: Violino
pausa8
Fa Mi Re Do si la si Do Re Do si la sol fa sol mi
maisRapido re fa la sol fa re fa mi re desceOitava si sobeOitava re la sol si la sol
fa re mi Do Re Fa La la si sol la fa re Re Re3/2 maisRapido Do
Re Do Re re do la mi fa re Re Do si la Fa La Si Sol Fa Mi Sol Fa Mi Re Do si la sol fa mi sol fa mi

Instrumento: Violino
pausa16
Fa Mi Re Do si la si Do Re Do si la sol fa sol mi
maisRapido re fa la sol fa re fa mi re desceOitava si sobeOitava re la sol si la sol
fa re mi Do Re Fa La la si sol la fa re Re Re3/2 maisRapido Do

Instrumento: Violino
Re/4 maisRapido fa/2 maisRapido la Re Do mi la Do si Re Fa si la do fa la sol desceOitava si sobeOitava re sol la re fa la si re
sol si Do mi la Do
re fa la Re Do mi la Do si Re Fa si la do fa la sol desceOitava si sobeOitava re sol la re fa la si re sol si Do mi la Do
re fa la Re Do mi la Do si Re Fa si la do fa la sol desceOitava si sobeOitava re sol la re fa la si re sol si Do mi la maisLento Do/2
maisLento
Fa Mi Re Do si la si Do Re Do si la sol fa sol mi
maisRapido re fa la sol fa re fa mi re desceOitava si sobeOitava re la sol si la sol

Instrumento: Celo
Re la si fa sol re sol la
Re la si fa sol re sol la
Re la si fa sol re sol la
Re la si fa sol re sol la
Re la si fa sol re sol la
Re la si fa sol re sol la

X: 1
T: Canon em Ré Maior

Canon em Ré Maior
Johann Pachelbel

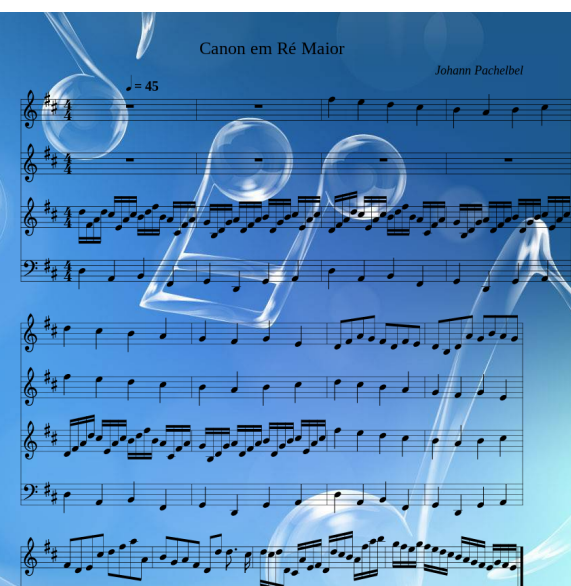


Figura 3: Demonstração completa da escrita de uma música.

Estrutura de Dados

A estrutura de dados utiliza-se da agregação entre objetos, seja por meio de atributos ou por meio de uma lista de outros objetos.

No nível mais alto, há o objeto devolvido pelo parse. Esse objeto contém um objeto de cabeçalho e uma lista de vozes. O objeto de cabeçalho, contém os atributos do cabeçalho.

As vozes que formam a lista de vozes representam um instrumento. Ela contém o identificador do instrumento e uma lista de notas. Cada nota tem a sua altura, a sua oitava, a sua duração e o seu acidente, dados necessários para identificar as notas. Como a sua execução e escrita é sequencial, a lista de notas é suficiente para a sua representação.

Há símbolos especiais que também podem ser armazenados dentro da lista de notas, como as pausas ou outros diretivas diretas do abc, desde que não contenham dois pontos.

Assim como cada voz (instrumento) é armazenado separadamente, há liberdade para a sua edição sem que haja interferência no outro instrumento.

Escritor abc

Para escrever a estrutura de dados para ABC, é necessário percorrê-la inteiramente e sequencialmente. Cada instrumento pode ser percorrido paralelamente.

Para percorrer uma lista de notas, é necessário contar o tempo de cada nota, de tal forma que a nota se encaixe perfeitamente no compasso. Assim, é feita a divisão de notas que tenham uma duração maior do que a do compasso, sendo necessário uni-las em diferentes compassos. Também é realizado o agrupamento entre colcheias e semicolcheias por 4.

São realizadas as transposições de oitavas armazenadas na nota além de serem aplicados os acidentes e símbolos especiais para cada nota.

O cálculo do número de compasso por cada linha também deve ser levado em consideração, para que a partitura não seja apenas uma grande linha de compasso. Assim, o número de compassos por linha foi padronizado por 4, embora esse valor possa ser ajustado como parâmetro no código.

Escreve-se primeiro o cabeçalho, no formato de abc, em seguida, sequencialmente, são escritas as traduções de cada voz, até que ao final haja a representação completa da estrutura de objetos em uma String no formato abc.

O resultado final é integrado com o framework abc.js, que realiza a renderização e a execução da música no formato abc. Como não é possível ouvir diferentes instrumentos pelo navegador, também disponibiliza-se os links para baixar um arquivo abc e outro midi.

Referência Bibliográfica

- [1] Disponível em: <https://abcjs.net/#what>
- [2] Disponível em: <https://abcjs.net/abcjs-editor.html>
- [3] Disponível em: <https://pegjs.org/>
- [4] Disponível em: https://rawgit.com/Gabru/CTC34_Gramatica-Musical/master/notaMusical.html

Anexo

Código disponível em https://github.com/Gabru/CTC34_Gramatica-Musical

```
{
var tg = 1;
var og = 0;
}

Dynota = _ cabecalho:Cabecalho _ vozes:Vozes _ {return {cabecalho, vozes};}

Cabecalho = t:Titulo? _ c:Compositor? _ m:Tempo? _ l:TempoNota? _ v:Velocidade?
_ k:Escala? _ d:Diretiva?

{let a = {}; if(t!=null) a.T=t; if(c!=null) a.C=c; if(m!=null) a.M=m; if(l!=
null) a.L=l; if(v!=null) a.Q=v;

if(k!=null) a.K=k; if(d!=null) a["%%score"]=d; return a;}

Titulo = "Titulo:" _ titulo:([^\n]*) [\n] {return titulo.join("");}

Compositor = "Compositor:" _ compositor:([^\n]*) [\n] {return
compositor.join("");}

Tempo = "Tempo:" _ s:([^\n]*) [\n] {return s.join("");}

TempoNota = "Nota:" _ s:([^\n]*) [\n] {return s.join("");}

Velocidade = "Velocidade:" _ s:([^\n]*) [\n] {return s.join("");}

Escala = "Escala:" _ s:([^\n]*) [\n] {return s.join("");}

Diretiva = "Juntos:" s:([^\n]*) [\n] {return s.join("");}

Vozes = p:VozPrincipal v:Voz* {let r = [p]; return r.concat(v);}

VozPrincipal = _ nomeVoz:IdVoz? _ notas:Pentagrama {return {nomeVoz, notas};}

Voz = _ nomeVoz:IdVoz _ notas:Pentagrama {return {nomeVoz, notas};}
```

```
IdVoz = "Instrumento:" _ s:([^\n]*) [\n] {tg=1; og=0; return s.join("");}
```

Pentagrama

```
= Simbolo*
```

Simbolo

```
= _ n:Nota aci:Acidente? temp:Temporizacao? _ Comando?
```

```
{if (aci!=null) n["acidente"]=(aci); if (temp!=null) n["tempo"]*=temp; return n}
```

Comando =

```
"maisLento" {tg*=2; return null;}  
/"maisRapido" {tg/=2; return null;}  
/"sobeOitava" {og++; return null;}  
/"desceOitava" {og--; return null;}
```

Acidente

```
= "#" / "##" / "b" / "bb"
```

Temporizacao

```
= "/" val:Inteiro {return 1/val;} / Fracao
```

Nota "nota"

```
= NotaNormalAumentada  
/ NotaOitavaAcimaAumentada  
/ NotaNormal  
/ NotaOitavaAcima  
/ "pausa" {return {nota:"pausa", oitava:og, tempo:tg}}  
/ s:([^\t\n\r:]*) [\t\n\r] {return {nota:s.join(""), especial:true};}
```

NotaNormal

```
= ("do" / "re" / "mi" / "fa" / "sol" / "la" / "si") {return {nota:text(),  
oitava:og, tempo:tg};}
```

NotaNormalAumentada

```
= n:("doo" / "ree" / "mii" / "faa" / "sool" / "laa" / "sii") {return {nota:
```

```
(n[0]+n.slice(2)), oitava:og, tempo:2*tg}};
```

NotaOitavaAcima

```
= n:("Do" / "Re" / "Mi"/ "Fa" / "Sol" / "La"/ "Si") {return  
{nota:text().toLowerCase(), oitava:og+1, tempo:tg}};
```

NotaOitavaAcimaAumentada

```
= n:("Doo" / "Ree" / "Mii"/ "Faa" / "Sool" / "Laa"/ "Sii")  
{return {nota:(n[0]+n.slice(2)).toLowerCase(), oitava:og+1, tempo:2*tg}};
```

Inteiro "inteiro"

```
= [0-9]+ { return parseInt(text(), 10); }
```

Fracao "Fração"

```
= num:Inteiro "/"? den:Inteiro? {let resp = num; if(den!=null) resp/=den;  
return resp}
```

_ "espaços brancos"

```
= [ \t\n\r]*
```

```

1 {
2   var tg = 1;
3   var og = 0;
4 }
5 Dynota = _ cabecalho: Cabecalho _ vozes: Vozes _ {return {cabecalho, vozes};}
6
7 Cabecalho = t: Titulo? _ c: Compositor? _ m: Tempo? _ l: TempoNota? _ v: Velocidade? _ k: Escala? _ d: Diretiva?
8 {let a = {}; if(t!=null) a.T=t; if(c!=null) a.C=c; if(m!=null) a.M=m; if(l!=null) a.L=l; if(v!=null) a.Q=v;
9 if(k!=null) a.K=k; if(d!=null) a["%%score"]=d; return a;}
10
11 Titulo = "Titulo:" _ titulo: ([^\\n]*) [\\n] {return titulo.join("");}
12
13 Compositor = "Compositor:" _ compositor: ([^\\n]*) [\\n] {return compositor.join("");}
14
15 Tempo = "Tempo:" _ s: ([^\\n]*) [\\n] {return s.join("");}
16
17 TempoNota = "Nota:" _ s: ([^\\n]*) [\\n] {return s.join("");}
18
19 Velocidade = "Velocidade:" _ s: ([^\\n]*) [\\n] {return s.join("");}
20
21 Escala = "Escala:" _ s: ([^\\n]*) [\\n] {return s.join("");}
22
23 Diretiva = "Juntos:" s: ([^\\n]*) [\\n] {return s.join("");}
24
25 Vozes = p: VozPrincipal v: Voz* {let r = [p]; return r.concat(v);}
26
27 VozPrincipal = _ nomeVoz: IdVoz? _ notas: Pentagrama {return {nomeVoz, notas};}
28 Voz = _ nomeVoz: IdVoz _ notas: Pentagrama {return {nomeVoz, notas};}
29
30 IdVoz = "Instrumento:" _ s: ([^\\n]*) [\\n] {tg=1; og=0; return s.join("");}
31
32 Pentagrama
33   = Simbolo*
34
35 Simbolo
36   = _ n: Nota aci: Acidente? temp: Temporizacao? _ Comando?
37   {if (aci!=null) n["acidente"]=(aci); if (temp!=null) n["tempo"]*=temp; return n}
38
39 Comando =
40   "maisLento" {tg*=2; return null;}
41
42   /"maisRapido" {tg/=2; return null;}
43   /"sobeOitava" {og++; return null;}
44   /"desceOitava" {og--; return null;}
45
46 Acidente
47   = "#" / "##" / "b" / "bb"
48
49 Temporizacao
50   = "/" val: Inteiro {return 1/val;} / Fracao
51
52 Nota "nota"
53   = NotaNormalAumentada
54   / NotaOitavaAcimaAumentada
55   / NotaNormal
56   / NotaOitavaAcima
57   / "pausa" {return {nota: "pausa", oitava: og, tempo: tg}}
58   / s: ([^ \\t\\n\\r:]* ) [ \\t\\n\\r ] {return {nota: s.join(""), especial: true};}
59
60 NotaNormal
61   = ("do" / "re" / "mi" / "fa" / "sol" / "la" / "si") {return {nota: text(), oitava: og, tempo: tg};}
62
63 NotaNormalAumentada
64   = n: ("doo" / "ree" / "mii" / "faa" / "sool" / "laa" / "sii") {return {nota: (n[0]+n.slice(2)), oitava: og, tempo: 2*tg};}
65
66 NotaOitavaAcima
67   = n: ("Do" / "Re" / "Mi" / "Fa" / "Sol" / "La" / "Si") {return {nota: text().toLowerCase(), oitava: og+1, tempo: tg};}
68
69 NotaOitavaAcimaAumentada
70   = n: ("Doo" / "Ree" / "Mii" / "Faa" / "Sool" / "Laa" / "Sii")
71   {return {nota: (n[0]+n.slice(2)).toLowerCase(), oitava: og+1, tempo: 2*tg};}
72
73 Inteiro "inteiro"
74   = [0-9]+ { return parseInt(text(), 10); }
75
76 Fracao "Fração"
77   = num: Inteiro "/"? den: Inteiro? {let resp = num; if(den!=null) resp/=den; return resp}
78
79 _ "espaços brancos"
80   = [ \\t\\n\\r]*
81

```