Music-Mike

Husam Abdul-Kafi
System Architect
(hsa2126)

Harvey Wu
Language Guru
(hw2473)

Lakshmi Bodapati
Team Manager
(lmb2254)

Kaitlin Pet
Tester
(khp2106)

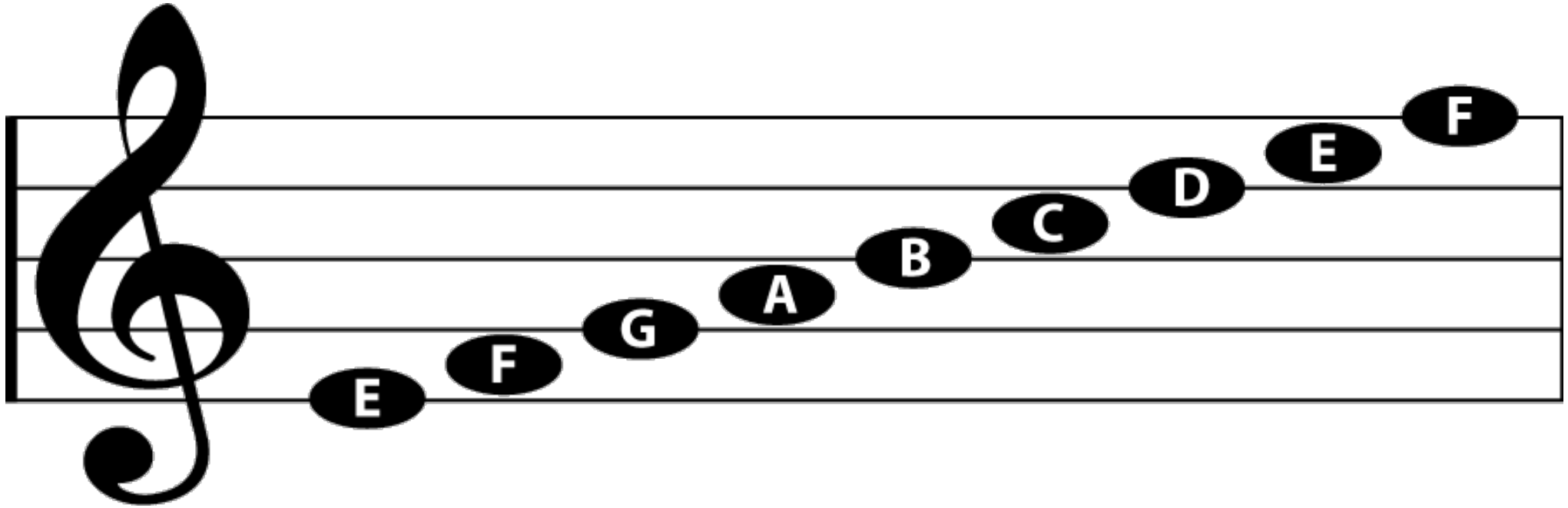
Motivation

- Write a strongly-typed language for expressing musical concepts intuitively for musicians
- Build music in terms of tunes, modes and rhythms independently

Looks good AND sounds good

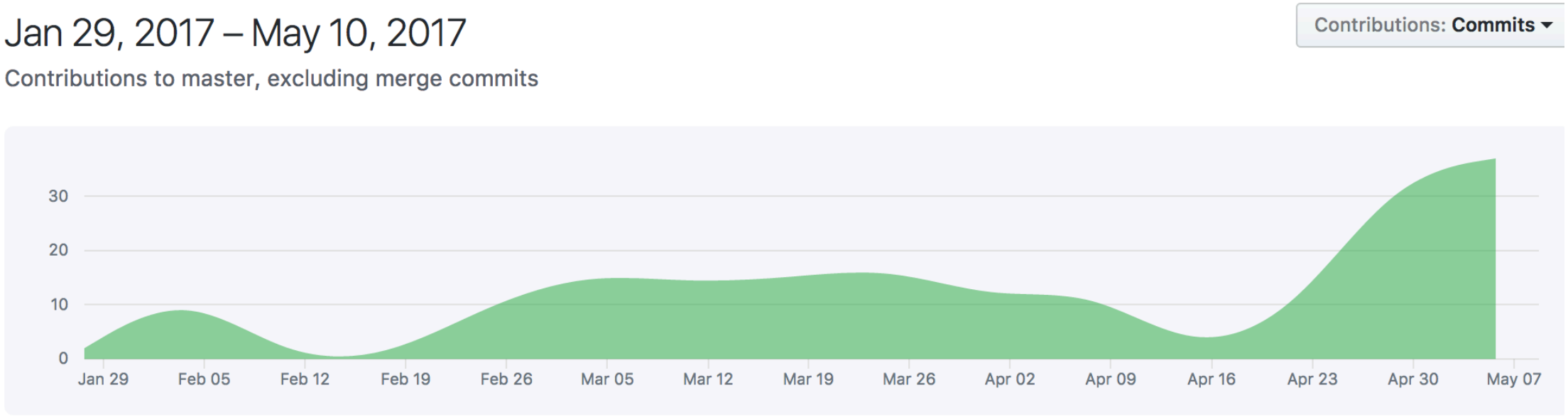
Introduction

- Western music notated on a staff with notes given a duration based on the symbol and pitch based on location on the staff.
- Most programming music libraries are unintuitive and complicated
- **music-mike** – create music based on varied manipulations of patterns in a modal system



Project Planning

- Agile Software Development using Github issues
- Feature based rather than module based work distribution
- Weekly deadlines



Overview and Features

- Context-Sensitive Scanner
- Type Inference
- Immutable Data Types
- Polymorphic Functions
- Modal-based music manipulation and expression
- Easy-to-use musical programming language
- Output music to multiple channels with ease

Syntax

Basic Functionality

```
pitch_list= p:[ ^1|^5 3 2|9 3# 4 5];
mode=[1 3 5 6 8 10 12];
rhythm_list=r:[ .1 .5 1. .05 1. .05];
start_note=60;
Synth(pitch_list rhythm_list mode start_note );
```

Function Declaration & Type Inference

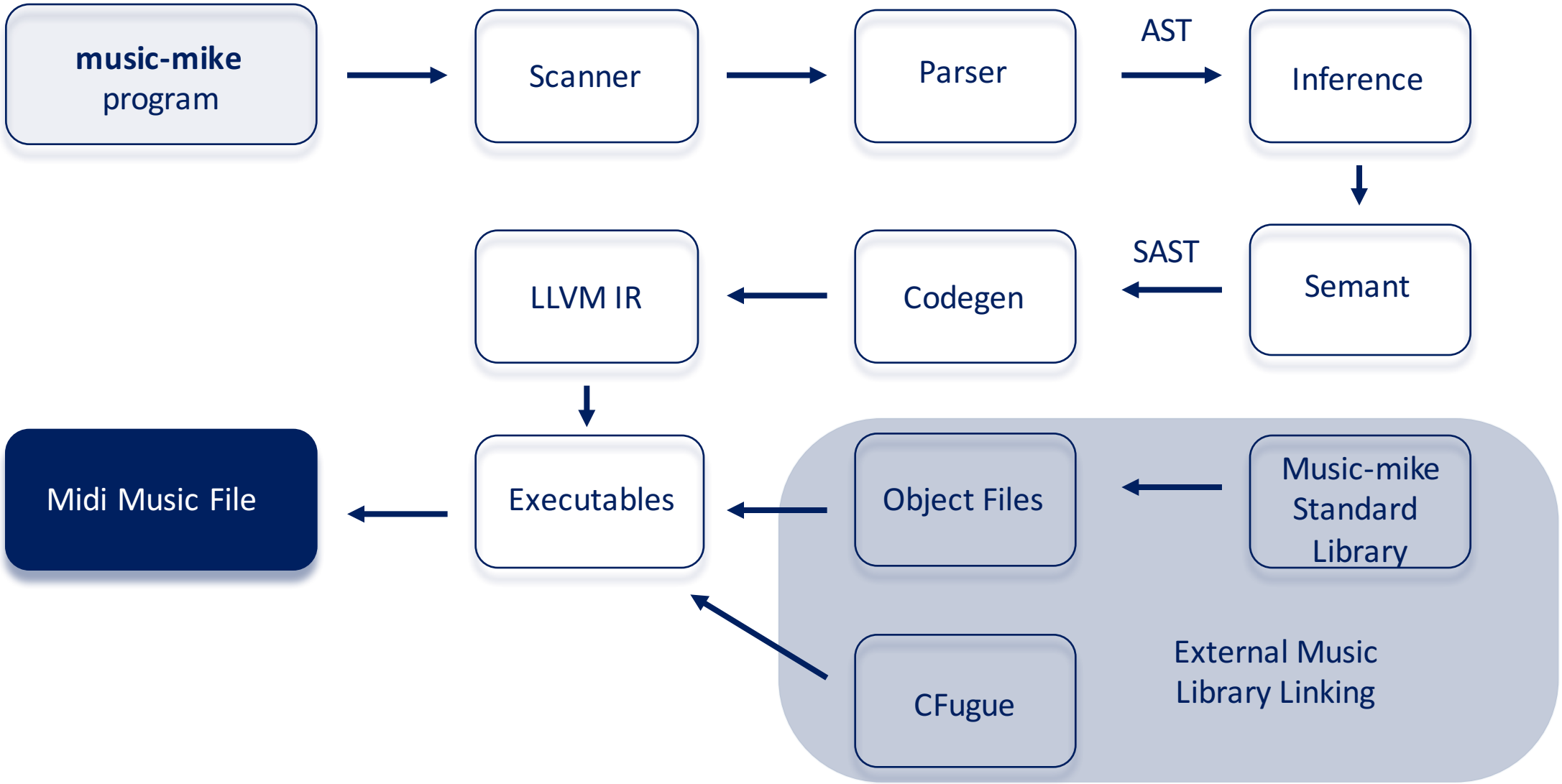
```
def Name x y z =
  {
    Printint(10);
    Printstr("hello");
    a = 5 + 10;
    x*a+z-y;
  };
Printint(Name(1 2 3));
```

```
y = 6;
def Fun x = x;
z = Fun(y);
Printint(z);
```

Complex Example

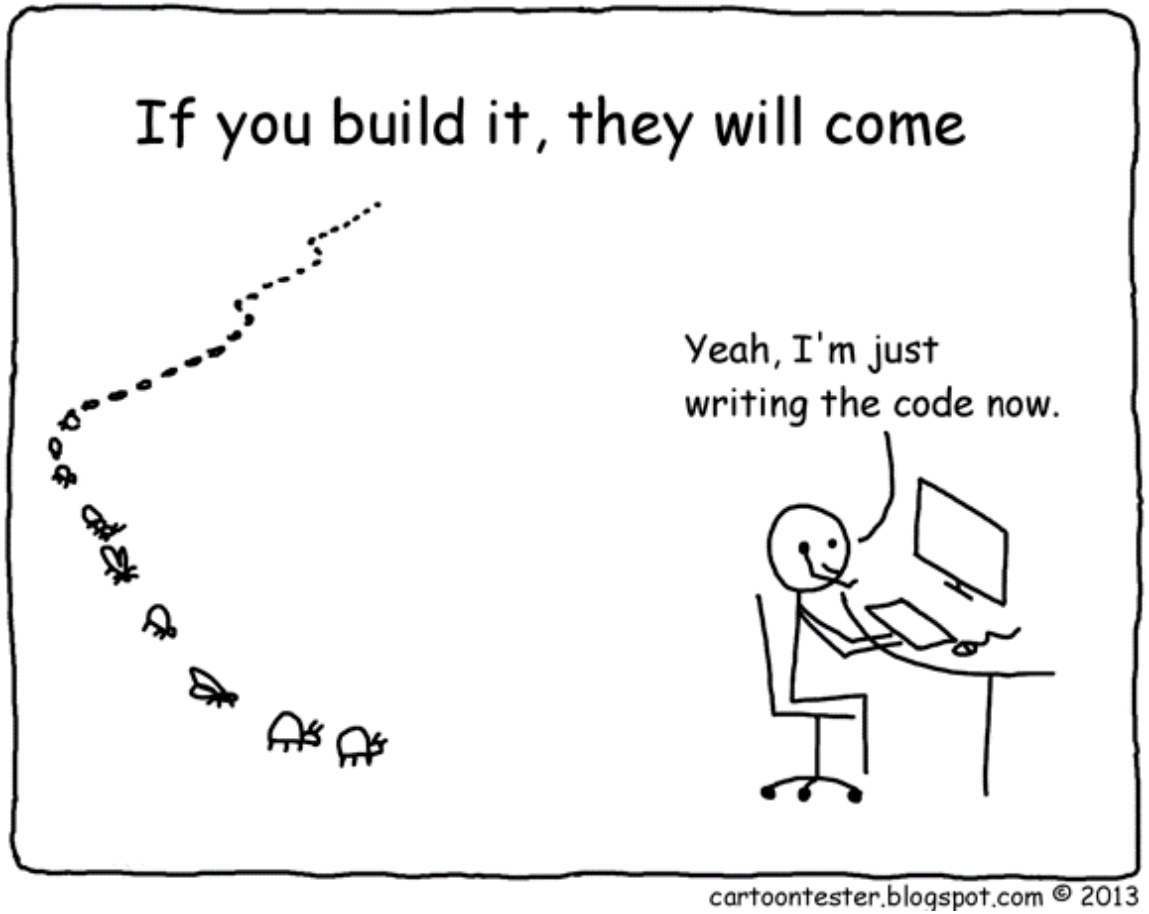
```
minor1 = [11 13 14 16 18 20 22 23];
minor2 = [11 13 15 16 18 20 22];
theme = p:[0 1 3 5 8 7 8 7b 8 6 8 6b 8 5 8 7 8 4# 6 3 2 3 4# 2 1];
r1 = r:[0. s s s s s s s s s s s s s s s s s s s s s s s s s s s];
r2 = r:[6. s s s s s s s s s s s s s s s s s s s s s s s s s s s];

counter = p:[0 5 6 5 4 3 2 4 3 2 1 v7 1 2 v7 1];
r3 = r:[0. s s s s s s s 1.25 s s s s s s s];
arp = p:[0 v5 v7 2 4 v7 2 4 6 5 4 6 5 4 3 2 1 v7 v6 v5 3];
r4 = r:[s s s s s s s s s s s s s s s s s s s s s s s 1.25];

rone = Synth(theme r1 minor1 50 1);
rtwo = Synth(counter r3 minor1 (50 + 14) 1);
lone = Synth(theme r2 minor1 33 2);

song = Merge(rone lone);

Make_midi(song "bach.midi");
```

Compiler Architecture



Testing

- Fail Testing
- Assignment
- Blocks
- Types
- Control Flow
- Printing
- Functions



Demo