

Beyond Middleware: Thinking like a Programmer

Ben Houge Associate Professor Berklee College of Music

GAME DEVELOPERS CONFERENCE March 14–18, 2016 · Expo: March 16–18, 2016 #GDC16



"I'm not a great programmer, but I can get the computer to do the job that I ask it to."

—John Chowning Inventor of FM Synthesis Founder of CCRMA, Stanford University



Overview

I. Programming DefinedII. Games and AffordancesIII. Programmer Use CasesIV. Programming the Future



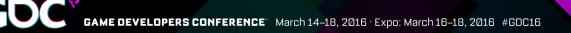
I. Programming Defined

- •What Is Programming?
- •Programming Concepts in Game Audio



What Is Programming?

- •Computer Code Is a List of Instructions
- •An Algorithm Is a Set of Rules
- •Programming Is Logical, Systematic Thinking



Programming Concepts in Game Audio

- •Modularity (Functions, Encapsulation)
- •Economy (Writing DRY Code)
- •Multiplicity (OOP)
- •Parameterization (Inheritance)
- •Responsiveness (Encapsulation)



II. Games and Affordances

- •Game Engines and Middleware
- •Things a Programmer Can Do



Game Engines and Middleware

- •A Game Is a Specific Kind of Program
- •What a Game Engine Does
- •A Set of Affordances
- •Question Your Materials



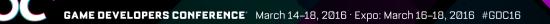
Things a Programmer Can Do

- •Develop an Engine
- •Play Sounds
- •Mix Sounds
- Modify Sounds
- Sequence Sounds
- Report/Debug/Optimize Sounds
- •Develop Offline Tools



III. Programmer Use Cases

- •Throwing Sounds over the Wall
- •Data in an External Resource
- •Implementing Middleware
- •New Deployment Mechanisms
- •Be Your Own Programmer



#1: Throwing Sounds over the Wall

- •Leisure Suit Larry 7: Love for Sail! (1996)
- •Everything Done in Code
- Programmer Required for all Changes
- •Little Control or Iteration



#2: Data in an External Resource

- •King's Quest: Mask of Eternity (1998)
- •Data vs. Metadata
- •Simple Text Files
- •Reboot Required to Test
- •Empowerment: No Programmer Required!



#3: Implement Middleware

- Johnny Drama (1999-2001, unreleased)
- Microsoft's DirectMusic
- •Visual Basic Scripting
- •Programmer Calls Methods, Passes Data
- •Two Sides: Tool vs. Runtime
- •Analogous to Wwise/FMOD



#4: New Deployment Mechanisms

- •Tom Clancy's EndWar (2008)
- Prototype Audio Behaviors in Max/MSP
- •System Design = A Kind of Composition
- •Work w/ Programmer to Integrate and Test
- •New Tools to Manage New Data



#5: Be Your Own Programmer

- •Responding to Dancers' Movements
- •Manipulating Live Audio Streams
- •Sonifying Data
- Real-Time Soundtrack for a 5-Course Meal
- Prototyping Game Music Behaviors



Demo: Please Be Seated

- •Dance Collab w/ New Movement Collective
- •Performed in Valencia, Winchester, London
- •Real-Time Control over Parameters
 - •Intensity
 - •Musical Scale (in Just Intonation!)
 - •Number of Elements
 - Position



IV. Programming the Future

•Towards Future Innovation



Towards Future Innovation

- •Put Yourself in a Programmer's Shoes
- Programming Problems Are Compositional Problems
- •Inventing New Paradigms and Platforms
- •All Music Is Game Music



Q & A

Ben Houge Associate Professor Berklee College of Music bhouge@berklee.edu @AleaBoy

GAME DEVELOPERS CONFERENCE March 14–18, 2016 · Expo: March 16–18, 2016 #GDC16



Appendix A: Programming Terms and Examples



A Variable

// an integer (no fractional component)
coinCount = 56

// a floating-point number (can have a fractional part) playerHealth = 0.89

// a boolean value (true or false)
// named for 19th c. English mathematician George Boole
hasTalkedToWizard = true

// these are comments, BTW



An Expression

// can be evaluated to result in a single value
3 + 11
(x + y) / 2
playerHealth > 0



A Statement

// describes an action to be carried out // (i.e., something changes) playerHealth = 1.0; meaningOfLifeUniverseEverything = 42; x = x + 4; print("Health is %f", playerHealth); c++; // same as c=c+1, good name for a new language



A Conditional Statement

```
// controls the flow of a program
if (playerHealth <= 0) {
    playDeathAnimation();
    print("You lose.");
} else {
    playVictoryDance();
    print("You win!");
}</pre>
```



An Iterative Loop

```
// repeat an action for a certain number of times
int enemyCount = 12;
while (enemyCount > 0) {
    spawnEnemy();
    enemyCount = enemyCount - 1;
}
```



A Function (or Method)

```
// a sequence of operations to perform a specific task
int addTwoIntegers(int x, int y) {
    int sum;
    sum = x + y;
    return sum;
}
```

```
// later perform the task using the function's name
int mySum = addTwoIntegers(4, 3);
```



A Class

```
// a way of encapsulating values and functionality
// a blueprint for objects (00P)
class Rectangle {
    int width, height;
public:
    void set_values (int,int);
    int area() {return width*height;}
};
```



Appendix B: Overview of Common Programming Languages



Common Programming Languages

- •C++ vs. C (but not C+)
- •JavaScript (for web, Unity)
- •Java (Processing/Arduino)
- •C# (particularly in Unity)
- •Python (for scripting)
- •Swift or Objective C (for iOS/OSX development)
- •PHP or Ruby on Rails (for web servers)



Music-Specific Languages

- Csound
- •Max/MSP
- •Pure Data
- SuperCollider
- •Common Lisp Music