PSAM 5600 Section E CRN 4598
Spring Semester 2017
Wednesdays 7:00PM – 9:40PM

Currents: Coding with Cinder

Week 1: Course Introduction / Introduction to Cinder

Instructors
Luobin Wang (luobin@newschool.edu)
Weili Shi (weili@newschool.edu)

Welcome to Currents: Coding with Cinder!

Here we take creative coding seriously.

Who are we?

Luobin Wang
Developer, Potion
MFA DT ’16
http://peterobbin.me
luobin@newschool.edu

Weili Shi

Developer, Bluecadet

MFA DT ’16

http://shi-weili.com

weili@newschool.edu

Who are you?

Tell us about your background, programming experience, and interested areas!

What is Cinder

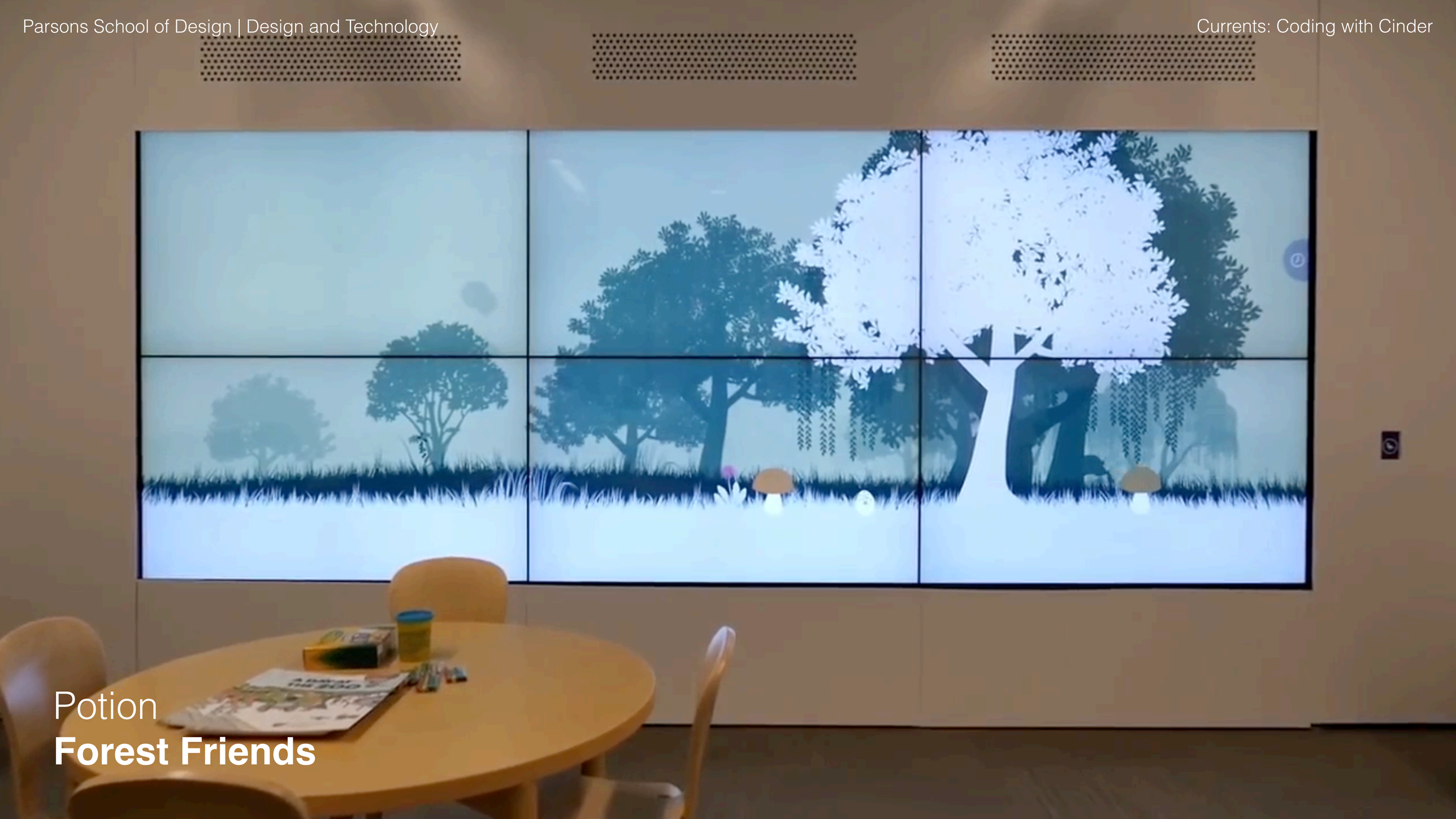
Cinder is an open source creative coding library written in C++. It provides a toolset similar to openFrameworks, but exposes more infrastructures of the C++ programming language and the computer, which makes it more powerful and stable.

What can Cinder do?

Simply speaking, Cinder provides a framework for building interactive multimedia applications. It is a wrapper of OpenGL. It also provides a united cross-platform programming interface to functionalities like graphics, audio, video, computational geometry, and input/output.

Bluecadet

Van Gogh’s Bedrooms



Potion
Forest Friends

Robert Hodgin et al.
Taxi, Taxi!

Red Paper Heart

The Good Deed Machine

Why teach Cinder?

While Cinder is the creative coding framework of choice in many digital agencies, it is rarely taught in schools. We want to fill the gap, and prepare you for professional creative coding work.

Why learn Cinder?

Cinder encourages good programming/ software engineering practices, which will empower your to accomplish comprehensive programming goals.

This is an intermediate-level course.

We expect you to be comfortable with writing interactive applications using at least one creative coding platform (openFrameworks, Processing, etc.)

Learning to code *is* challenging, and rewarding.

It is true to all of us.

We put even efforts into two things:

1. **Luobin** Introduces Cinder functionalities.
2. In support of Cinder programming, **Weili** provides related knowledge of C++, computer science, and software engineering.

Course outline

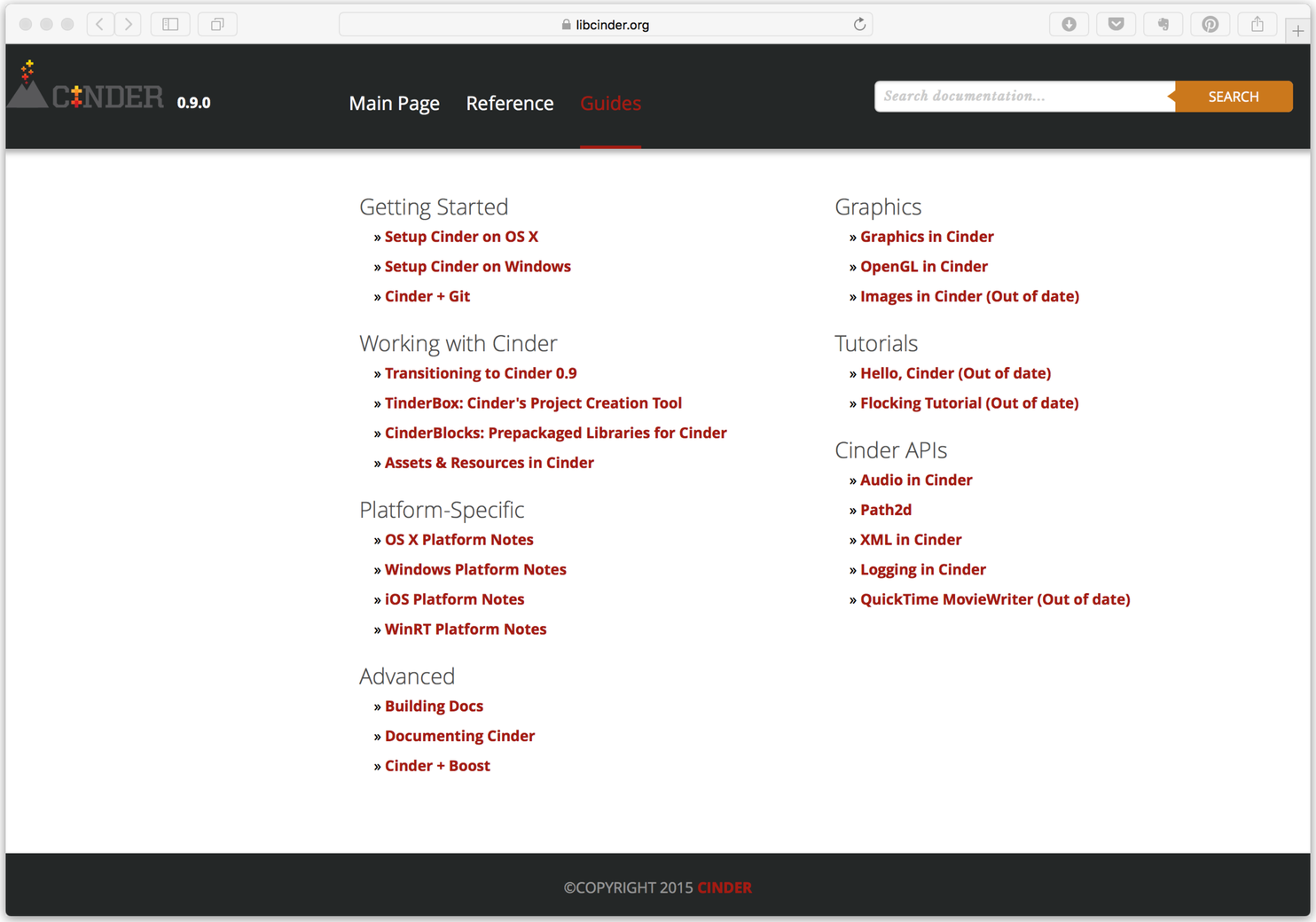
WEEK 1	*01/25*	Course introduction / introduction to Cinder
WEEK 2	*02/01*	C++ fundamentals recap / texture and shader
WEEK 3	*02/08*	Object oriented programming / particle system
WEEK 4	*02/15*	Memory management / particle system revisited
WEEK 5	*02/22*	Animation with timeline / version control with GitHub
WEEK 6	*03/01*	Video & audio playback / input & output
WEEK 7	*03/08*	Building / debugging / libraries / Cinder blocks
WEEK 8	*03/15*	Guest lecture 1 (Ritesh Lala from Potion)
WEEK 9	*03/22*	<Spring break / no class>
WEEK 10	*03/29*	poScene scene graph / MVC design pattern / final project kick-off
WEEK 11	*04/05*	poScene components / touch event handling
WEEK 12	*04/12*	poScene components continued / software architecture & project management
WEEK 13	*04/19*	Final project workshop 1
WEEK 14	*04/26*	Guest lecture 2 (Ben Bojko from Bluecadet)
WEEK 15	*05/03*	Final project workshop 2
WEEK 16	*05/10*	Final project presentation

Assignments & Grading

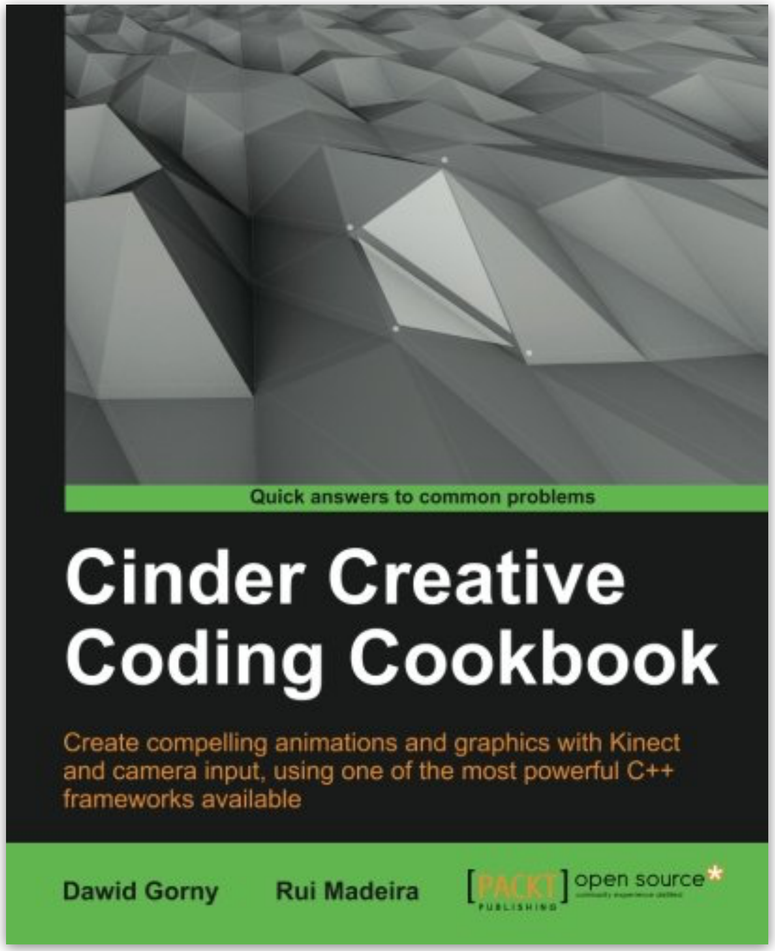
1. Attendance/participation 20%
2. Weekly assignments (individual) 40%
3. Final project (group) 40%

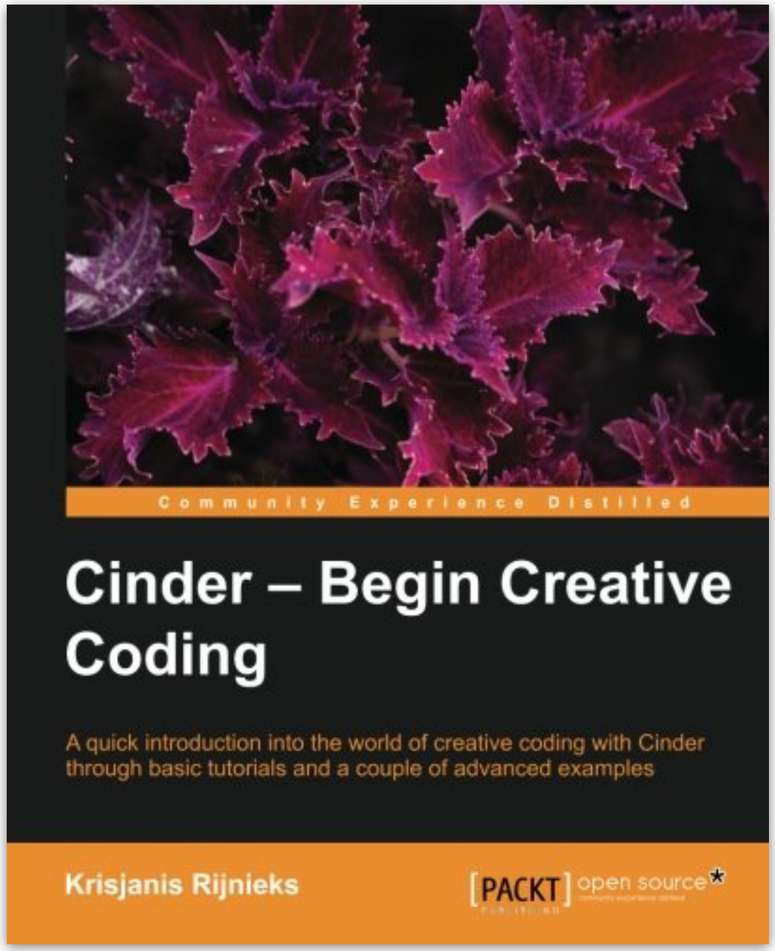
Recommended Readings

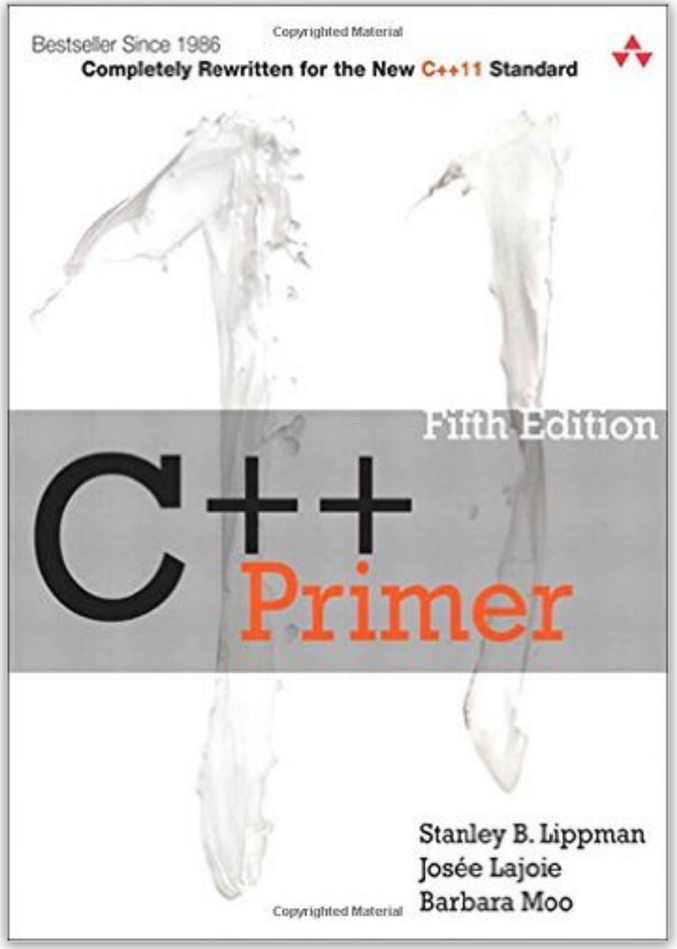
To name a few. For the curious minds.

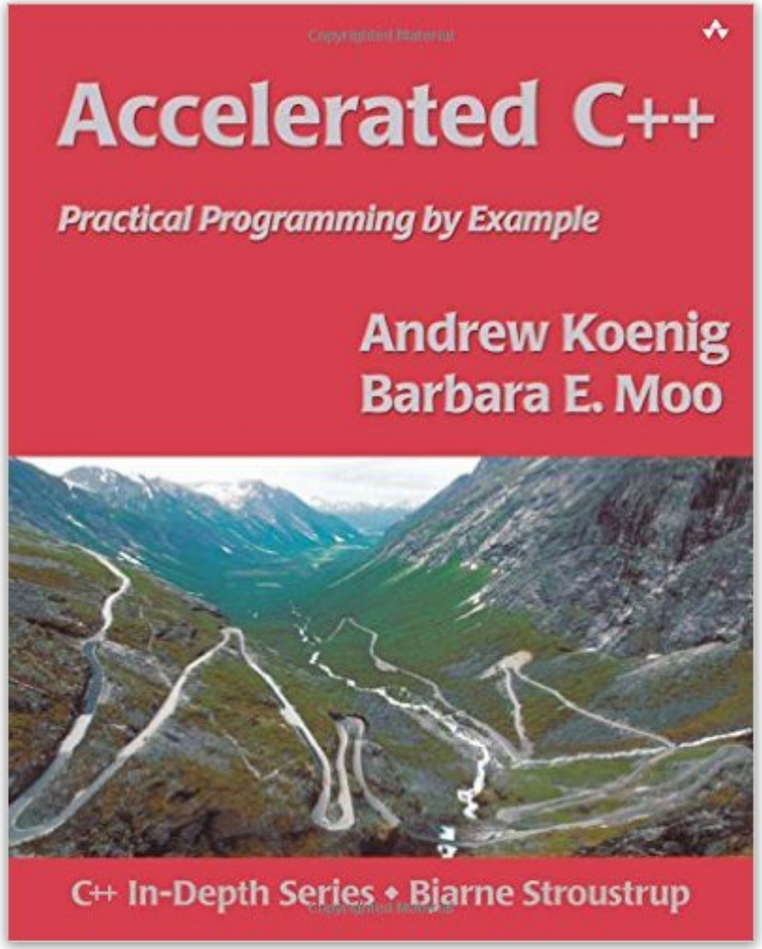


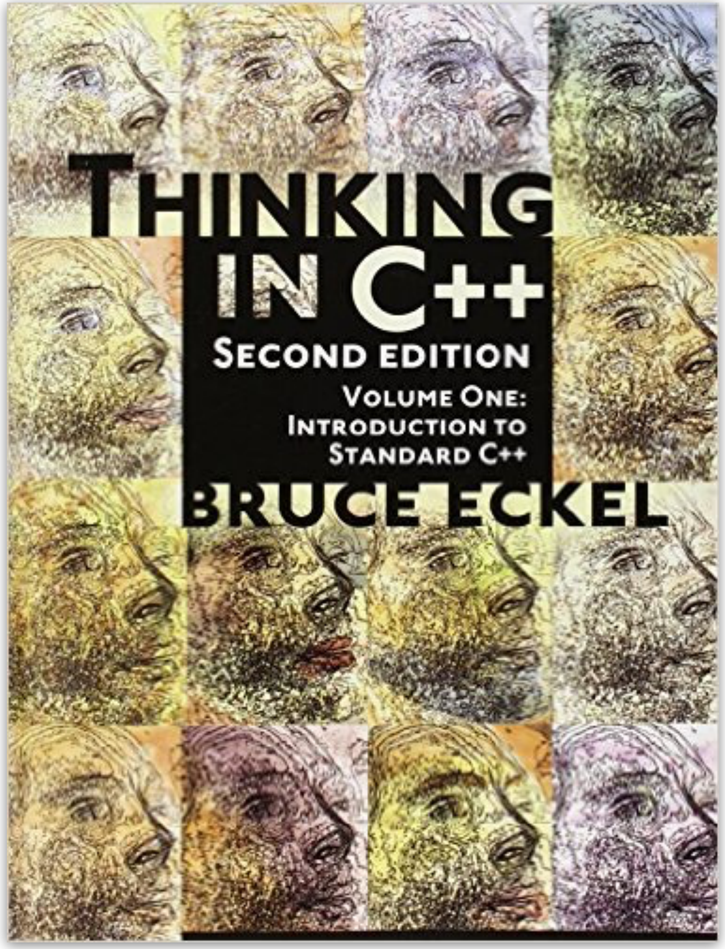
Official Cinder guides

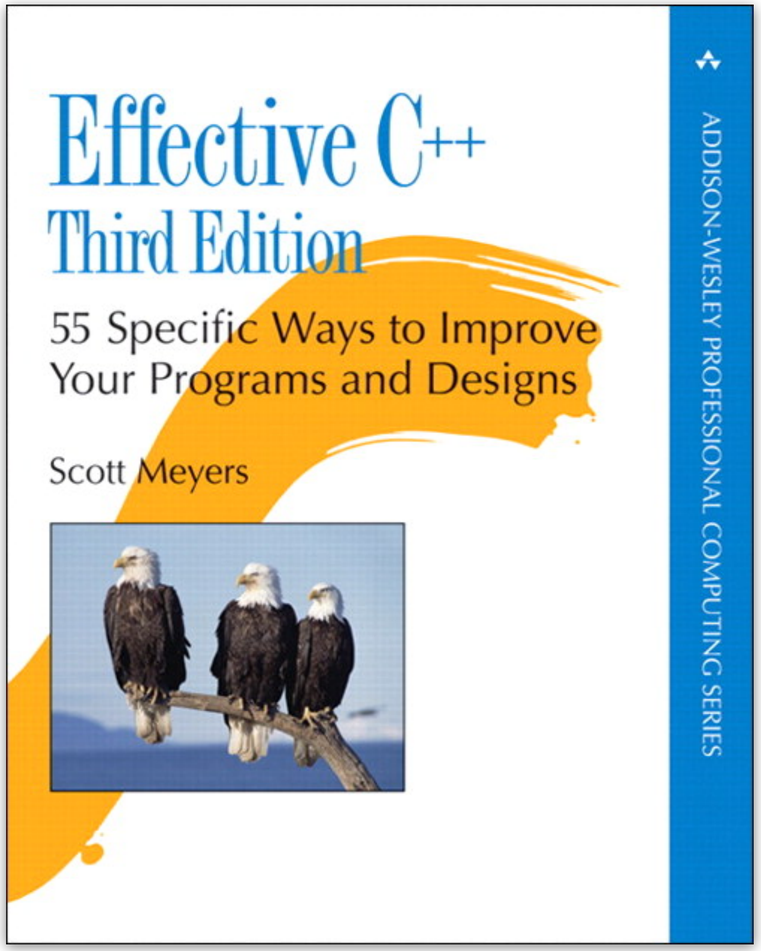

Quick answers to common problems
Cinder Creative Coding Cookbook
Create compelling animations and graphics with Kinect and camera input, using one of the most powerful C++ frameworks available
Dawid Gorny
Rui Madeira
[PACKT] open source
PUBLISHING

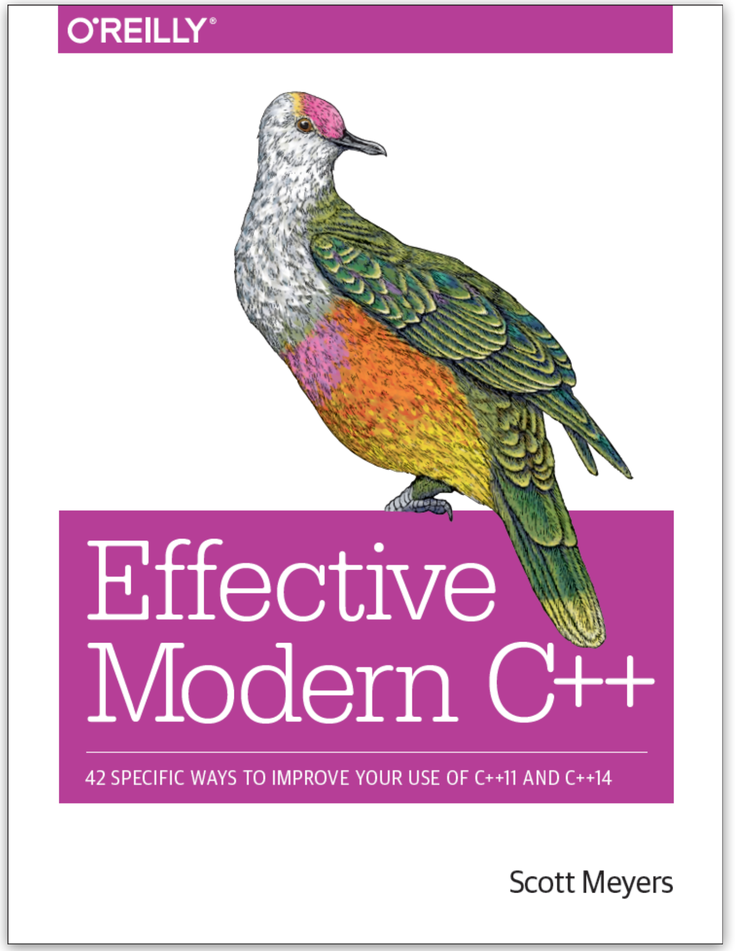

Community Experience Distilled
Cinder – Begin Creative Coding
A quick introduction into the world of creative coding with Cinder through basic tutorials and a couple of advanced examples
Krisjanis Rijnieks
[PACKT] open source

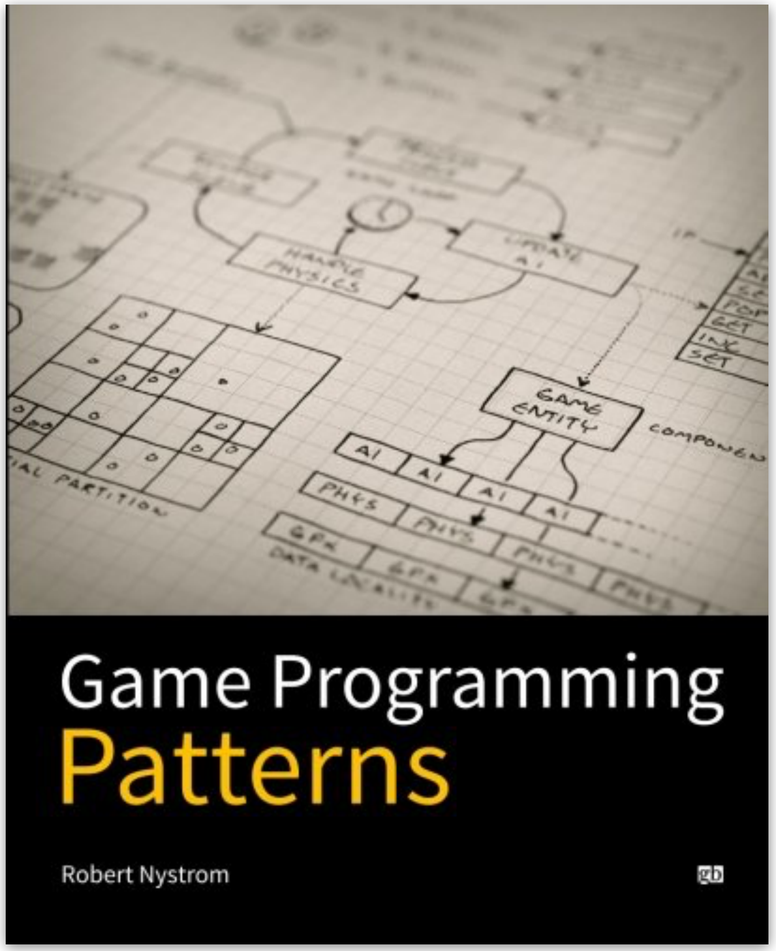

Bestseller Since 1986
Copyrighted Material
Completely Rewritten for the New C++11 Standard
Fifth Edition
C++ Primer
Stanley B. Lippman
Josée Lajoie
Barbara Moo
Copyrighted Material

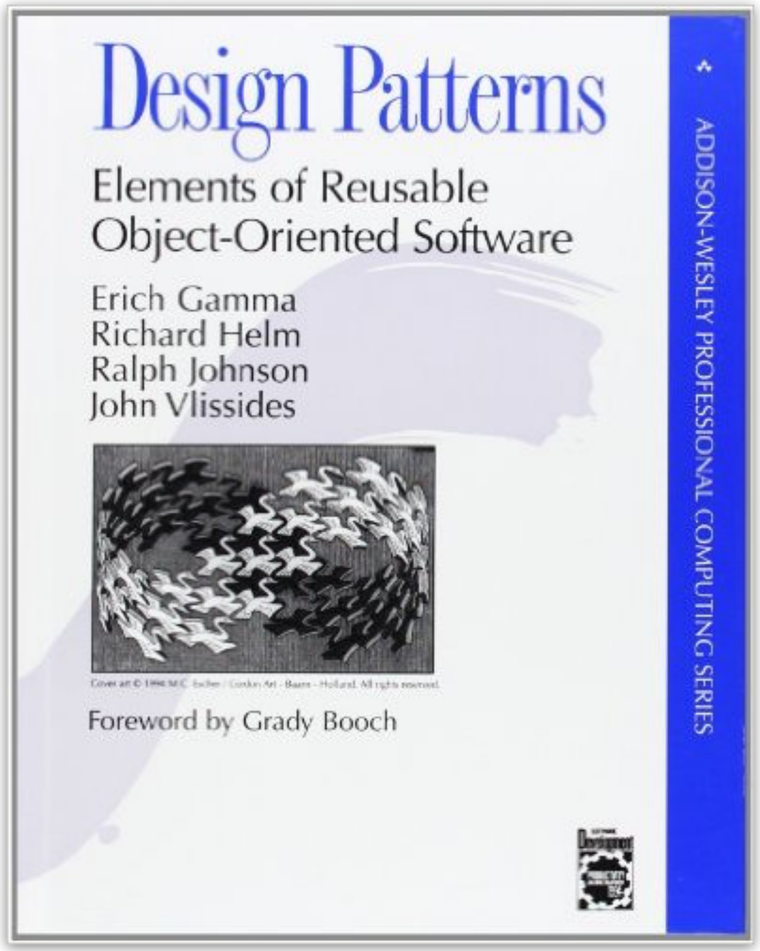

Accelerated C++
Practical Programming by Example
Andrew Koenig
Barbara E. Moo
C++ In-Depth Series • Bjarne Stroustrup

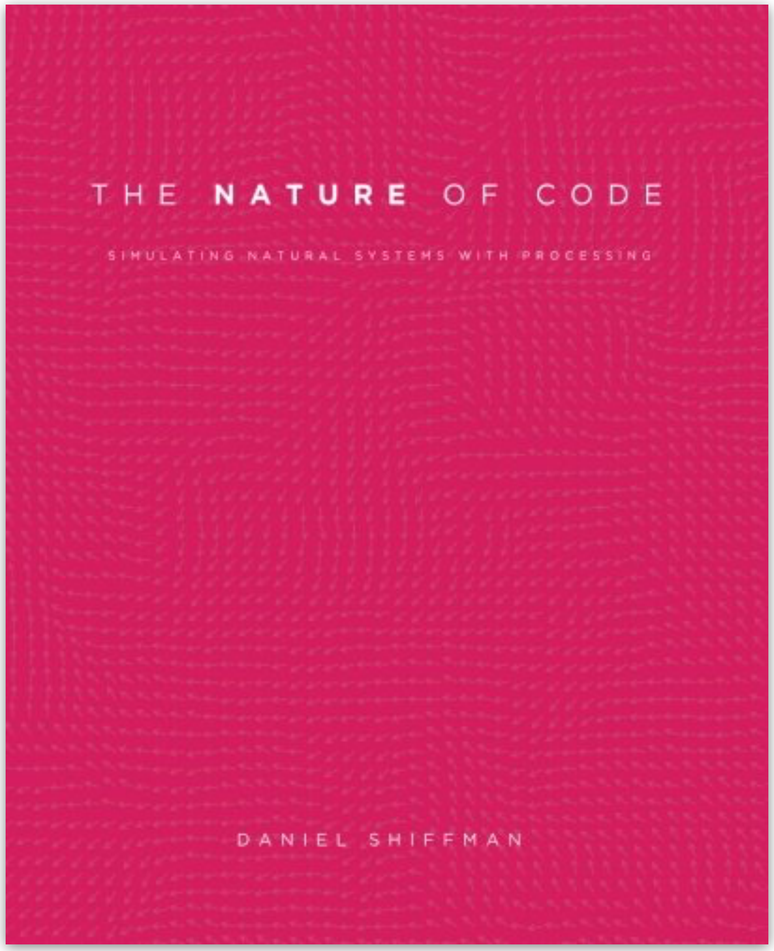

THINKING IN C++
SECOND EDITION
VOLUME ONE:
INTRODUCTION TO
STANDARD C++
BRUCE ECKEL


Effective C++
Third Edition
55 Specific Ways to Improve Your Programs and Designs
Scott Meyers
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

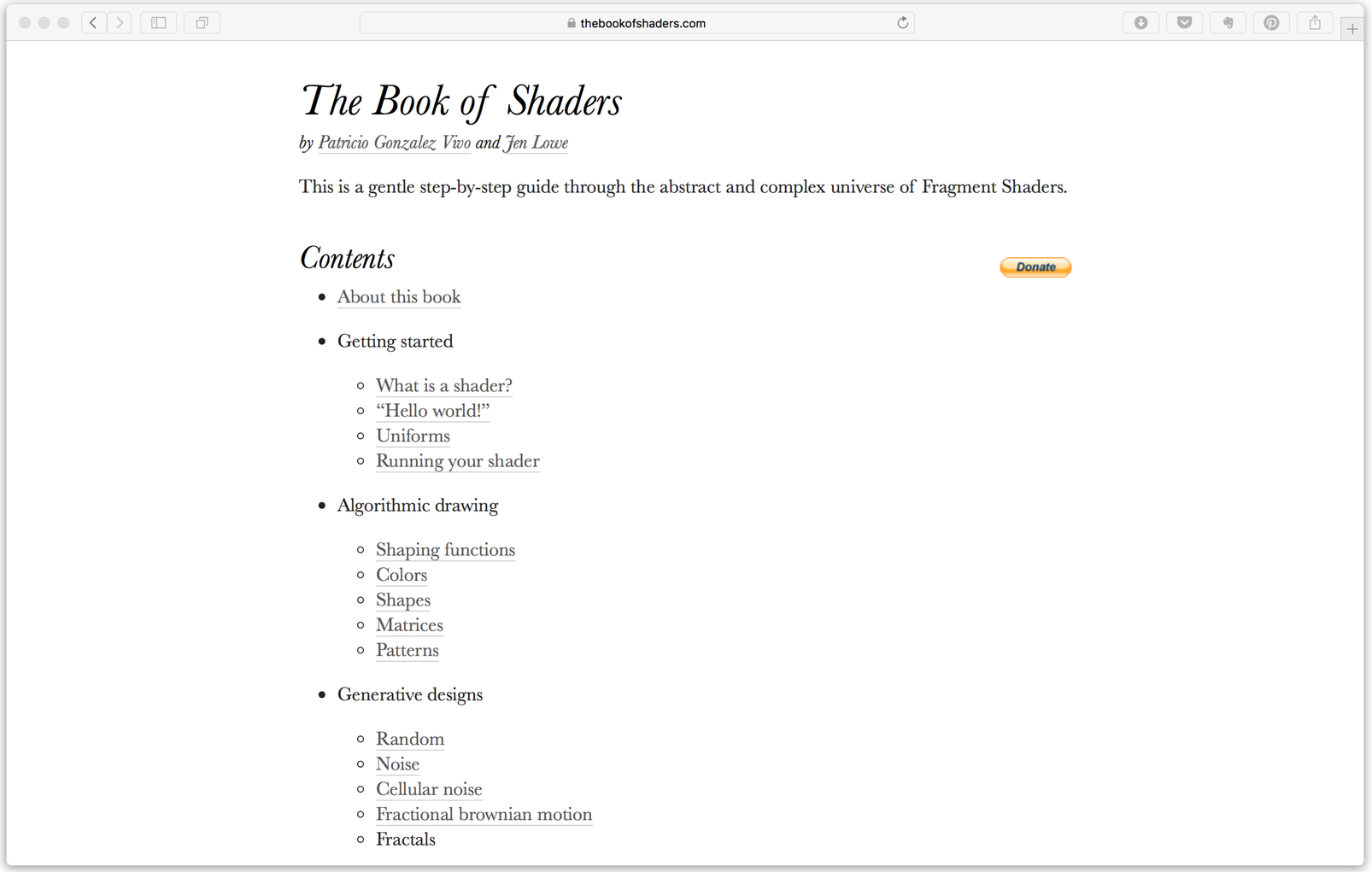

O'REILLY
Effective Modern C++
42 SPECIFIC WAYS TO IMPROVE YOUR USE OF C++11 AND C++14
Scott Meyers


Game Programming
Patterns
Robert Nystrom


Design Patterns
Elements of Reusable
Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES



The Nature of Code



The Book of Shaders

Survey

Are you a Mac user or a Windows user?

Getting started

1. Get your favorite IDE installed (Xcode or Visual Studio)

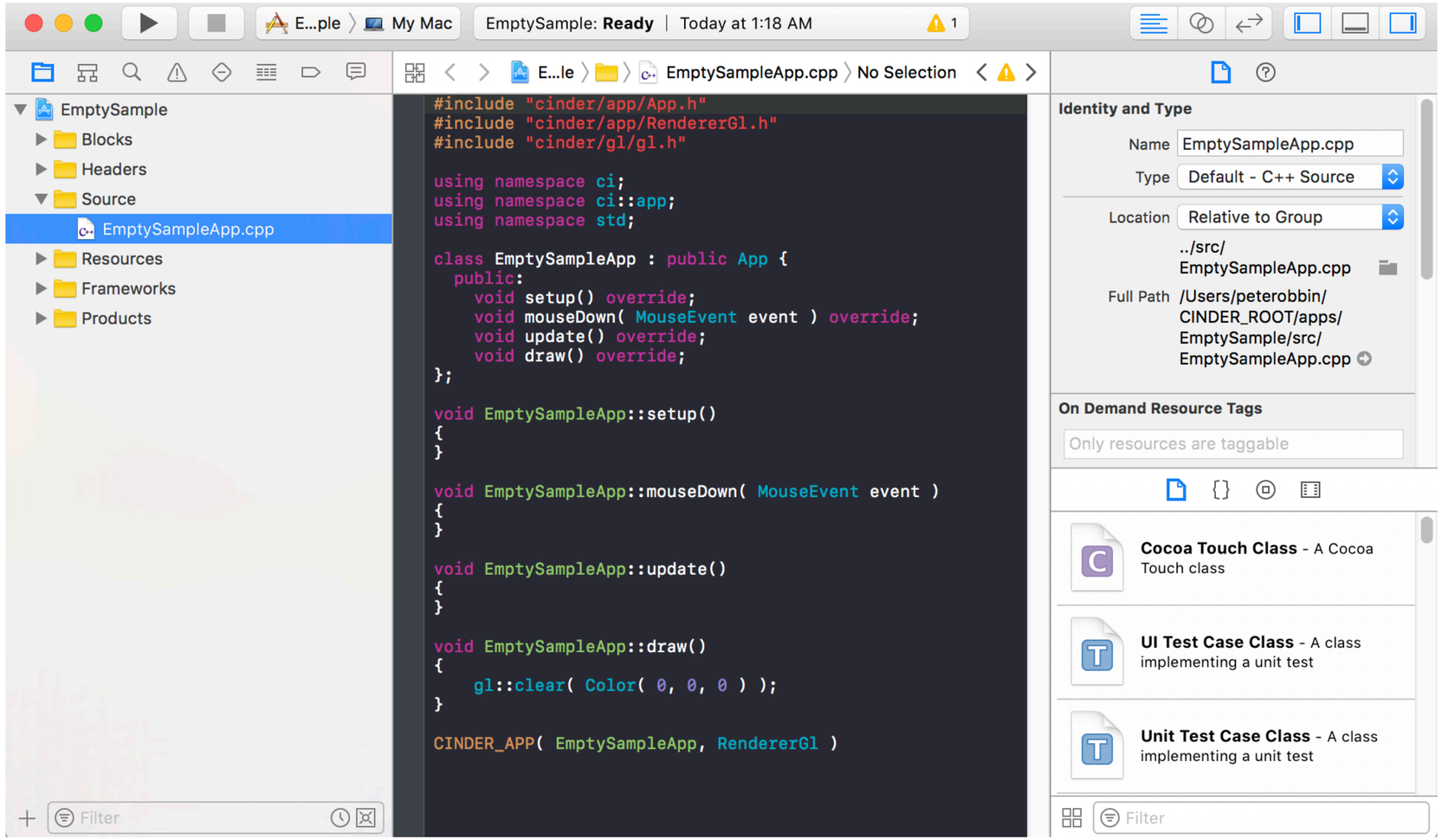
2. Download Latest Cinder from https://libcinder.org

3. Build Cinder

Using TinderBox

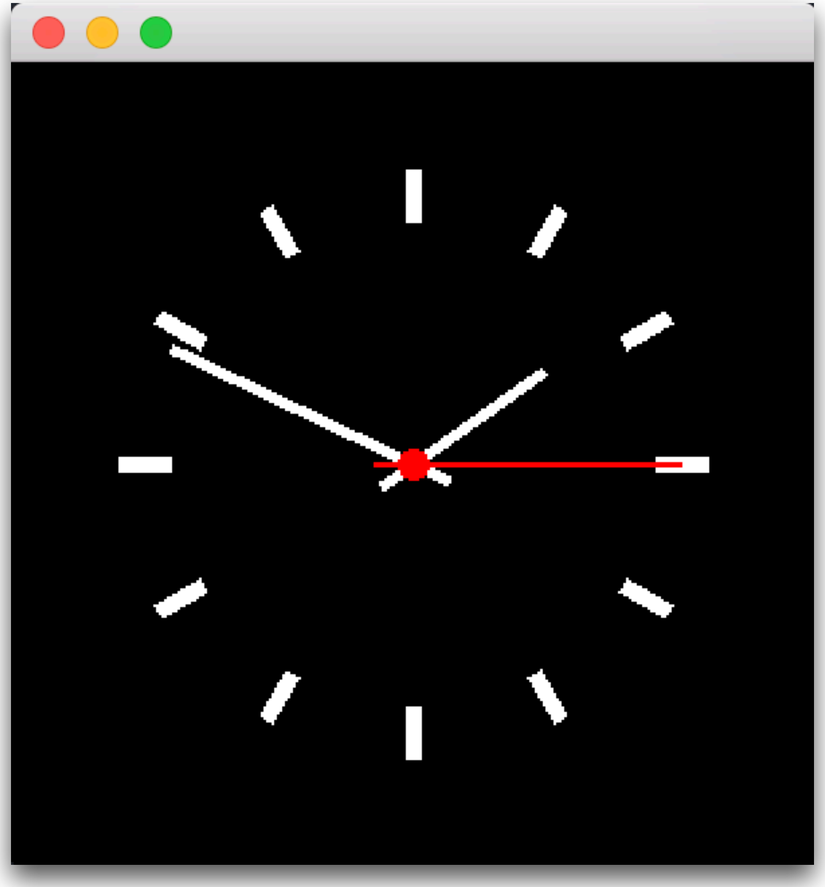
1. **Specify your project name (don’t just type in random names)**
2. **Specify the location of where you want to store your project**
3. **Select your target IDE to build project files (you can do both)**
4. **Only create a git repository if you know what you’re doing**
5. **We are not using any Cinder Blocks for now**

Your first Cinder project



Paul Houx's analog clock example

https://github.com/paulhoux/Cinder-Samples/tree/master/AnalogClock



Namespaces

Namespaces provide a method for preventing name conflicts in large projects. Symbols declared inside a namespace block are placed in a named **scope** that prevents them from being mistaken for identically-named symbols in other scopes.

http://en.cppreference.com/w/cpp/language/namespace

Using Namespace

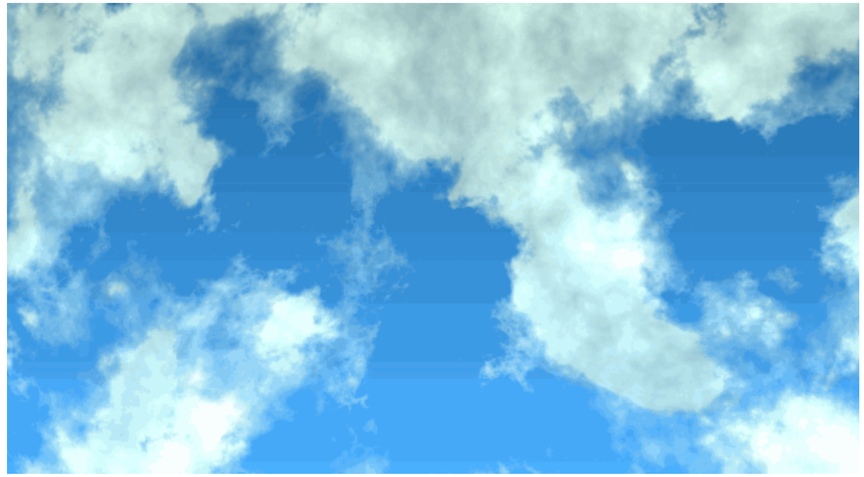
Using-directive (effective until the end of the scope): **using namespace std;**

Or

Specify the namespace when using it: **std::string;**

http://en.cppreference.com/w/cpp/language/namespace

ci::gl::



A group of functions under cinder’s openGL namespace, which uses the power of your graphic card.

ci:: or cinder::

Cinder namespace. Check Cinder documentation for namespaces and functions: https://libcinder.org/docs/

std::

C++ standard namespace. All the entities (variables, types, constants, and functions) of the standard C++ library are declared within the std namespace.

http://www.cplusplus.com/doc/tutorial/namespaces/

ci::vec2(x, y) is also glm::vec2(x, y)

Store values from a vector, or just two numbers you want to put together. GLM stands for OpenGL Mathematics.

gl::clear

Clean up everything that is on your canvas.

gl::pushModelView()

gl::popModelView()

Wrap translation inside push and pop to ensure that changes will only be applied inside.

gl::translate(glm::vec2)

Translate the origin point of a canvas to another location.

ci::Color(r, g, b)

A cinder object which stores RGB value.

gl::color(r, g, b)

gl::color(ci::Color(r, g, b))

Change your current drawing color.

gl::drawSolidRect()

gl::drawSolidCircle()

Draw basic shape using gl function.

std::time(nullptr)

Returns the current calendar time encoded as a **std::time_t** object, and also stores it in the object pointed to by arg, unless arg is a null pointer.

std::localtime(const time_t *)

```
auto t = std::time( nullptr ); // getting an time object
auto d = std::localtime(&t); // get calendar time from time object
```

Homework:

1. Create a analog digit (non interactive)

2. Create a analog meter that responds to mouse or keyboard input (if you hold key longer the meter should go higher)