

# Overview of Design Patterns in C++

**Douglas C. Schmidt**

**[d.schmidt@vanderbilt.edu](mailto:d.schmidt@vanderbilt.edu)**

**[www.dre.vanderbilt.edu/~schmidt](http://www.dre.vanderbilt.edu/~schmidt)**



**Professor of Computer Science**

**Institute for Software  
Integrated Systems**

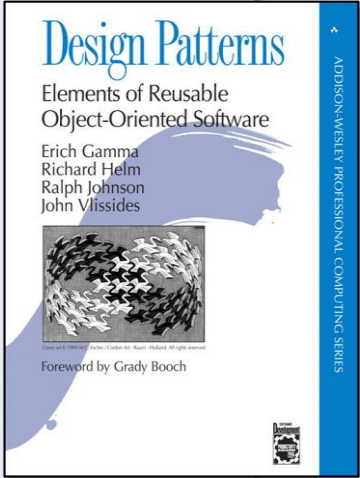

**Vanderbilt University  
Nashville, Tennessee, USA**



# Learning Objectives in this Lesson

- Know what topics we'll cover

	Creational	Structural	Behavioral
Class	Factory Method ✓	Adapter ✓ (class)	Interpreter ✓ Template Method ✓
Object	Abstract Factory ✓ Builder ✓ Prototype Singleton ✓	Adapter ✓ (object) Bridge ✓ Composite ✓ Decorator ✓ Flyweight Façade Proxy	Chain of Responsibility Command ✓ Iterator ✓ Mediator Memento Observer ✓ State ✓ Strategy ✓ Visitor ✓



# Learning Objectives in this Lesson

---

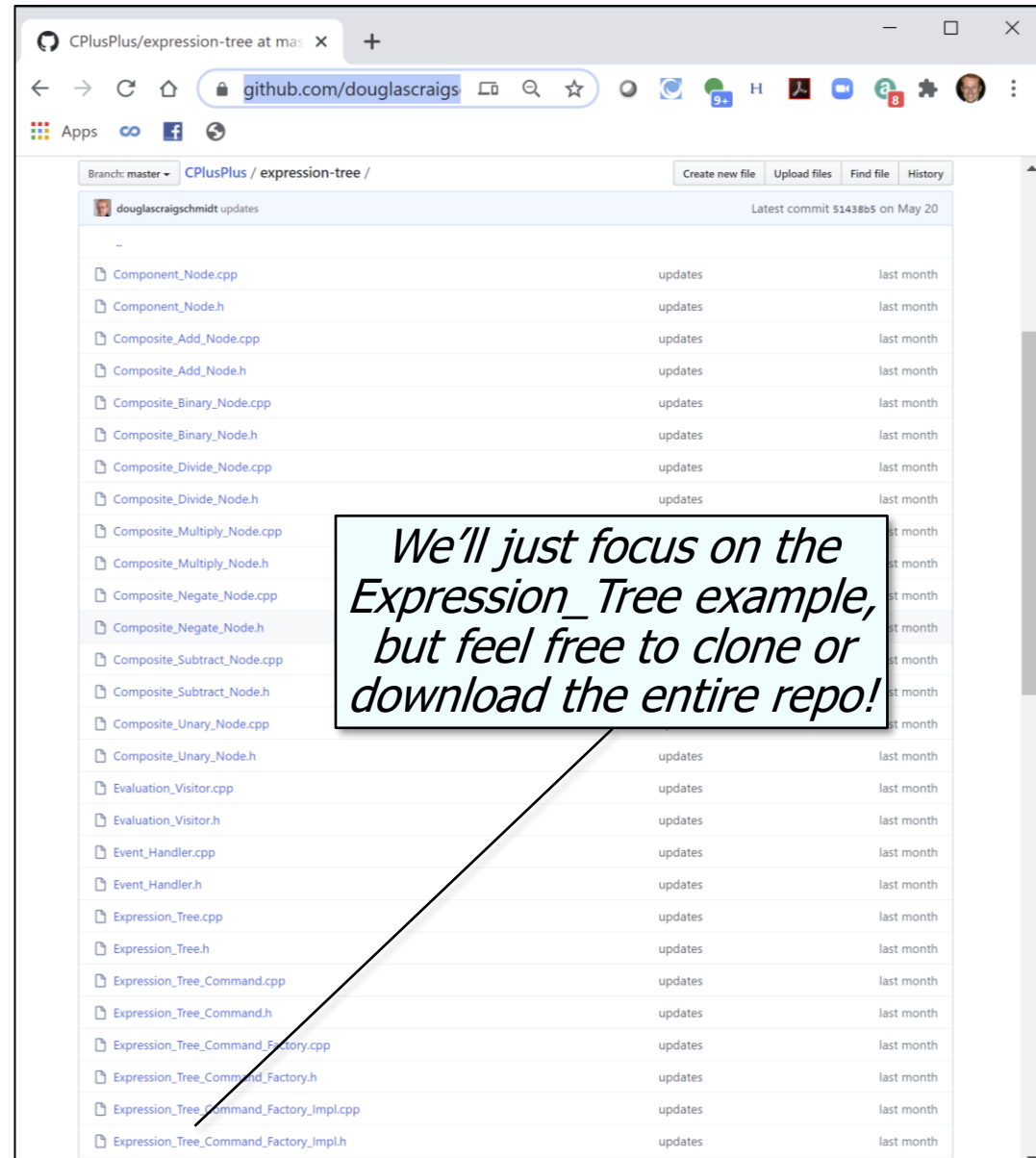
- Know what topics we'll cover
- Be aware of other digital learning resources



# Learning Objectives in this Lesson

- Know what topics we'll cover
- Be aware of other digital learning resources
- Be able to locate examples of the C++ case study app

USE THE  
SOURCE LUKE!



See [github.com/douglasraigschmidt/CPlusPlus/tree/master/expression-tree](https://github.com/douglasraigschmidt/CPlusPlus/tree/master/expression-tree)

---

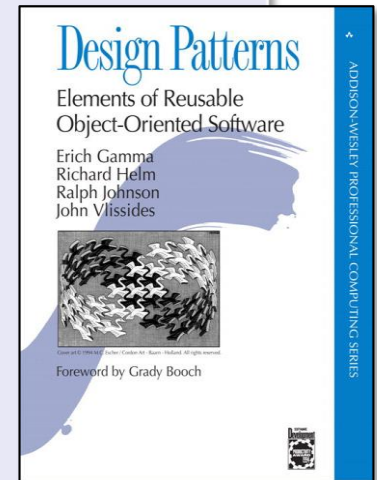
# Overview of this Part of the Course

---

# Overview of this Part of the Course

- We focus on programming “Gang-of-Four” (GoF) design patterns in C++, e.g.

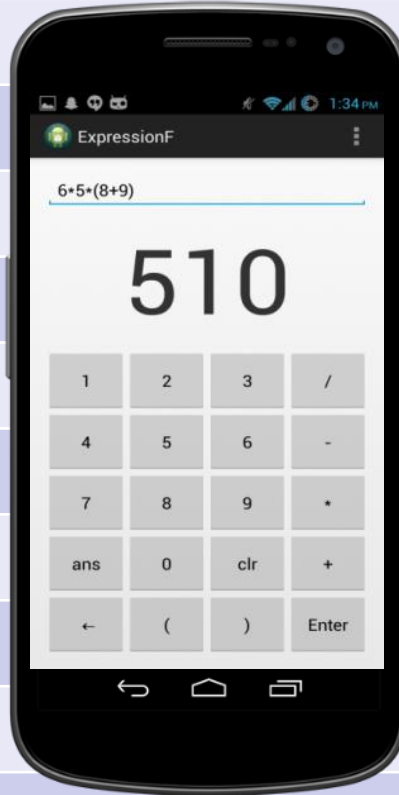
	Creational	Structural	Behavioral
Class	Factory Method ✓	Adapter ✓ (class)	Interpreter ✓ Template Method ✓
Object	Abstract Factory ✓ Builder ✓ Prototype Singleton ✓	Adapter ✓ (object) Bridge ✓ Composite ✓ Decorator ✓ Flyweight Façade Proxy	Chain of Responsibility Command ✓ Iterator ✓ Mediator Memento Observer ✓ State ✓ Strategy ✓ Visitor ✓



# Overview of this Part of the Course

- We apply many GoF patterns in the context of a case study app

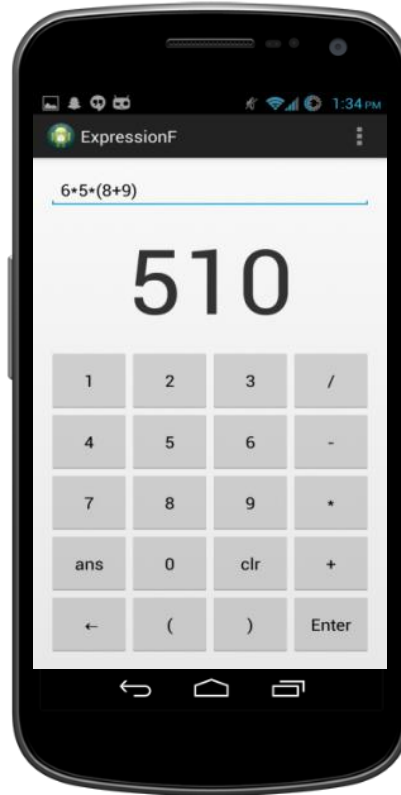
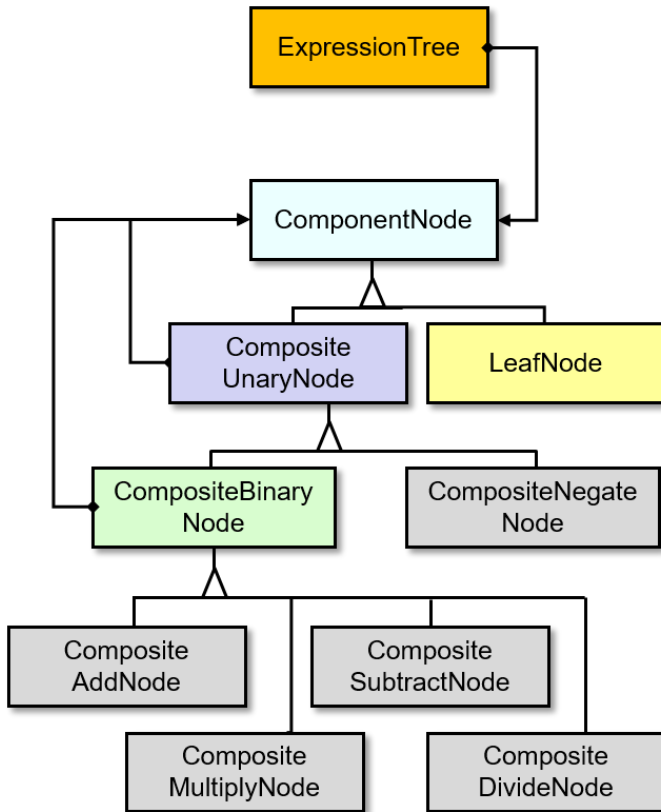
Design Problem	Pattern
Non-extensible & error-prone designs	Composite
Minimizing impact of variability	Bridge
Inflexible expression input processing	Interpreter
Inflexible interpreter output	Builder
Scattered request implementations	Command
Inflexible creation of variabilities	Factory Method
Inflexible expression tree traversal	Iterator
Obtrusive behavior changes	Strategy
Non-extensible tree operations	Visitor
Incorrect user request ordering	State
Non-extensible operating modes	Template Method
Minimizing global variable liabilities	Singleton



See [github.com/douglasraigschmidt/CPlusPlus/tree/master/expression-tree](https://github.com/douglasraigschmidt/CPlusPlus/tree/master/expression-tree)

# Overview of this Part of the Course

- This course focuses on both pattern-oriented design & implementation topics



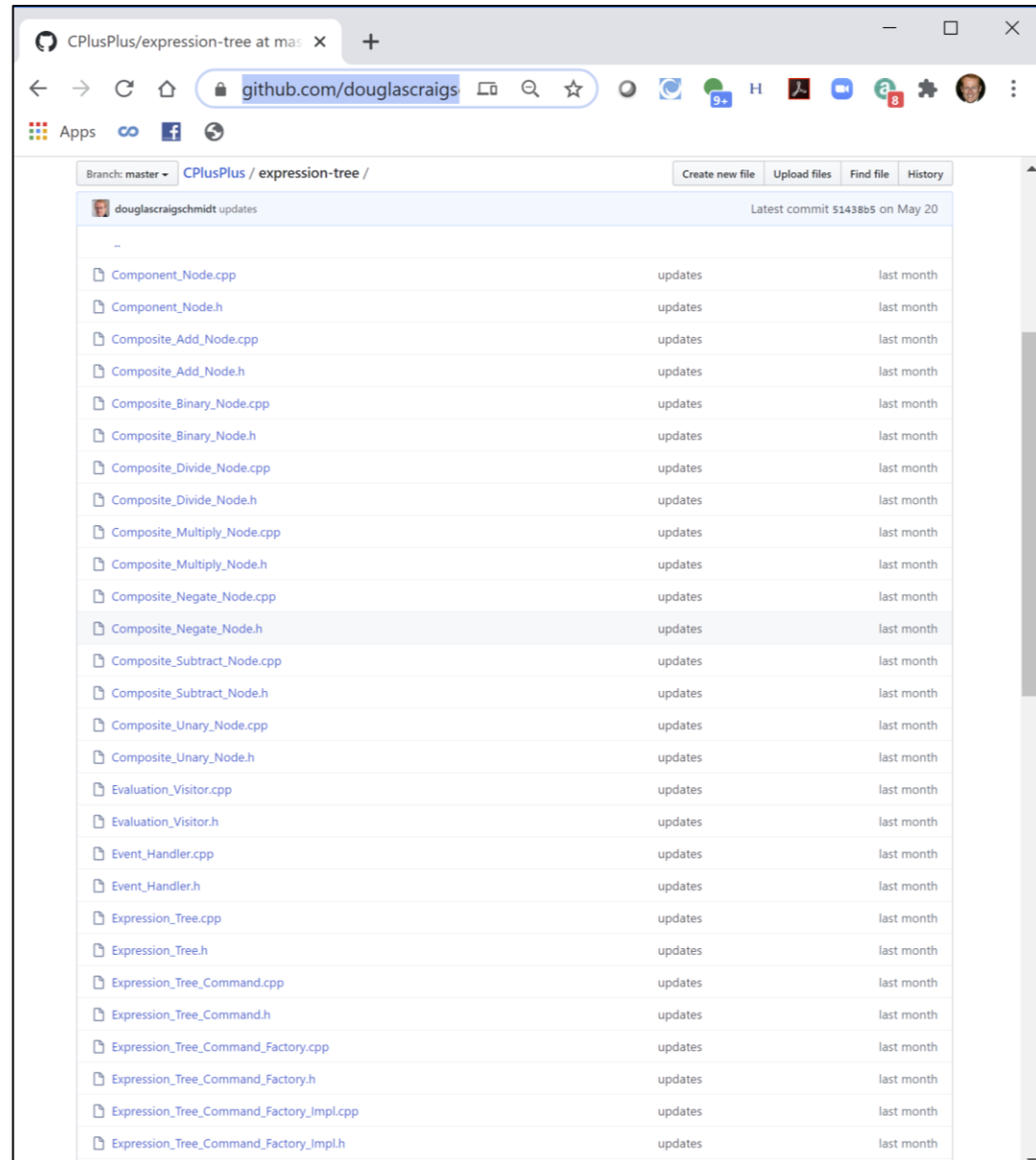
```
Expression_Tree
    expr_tree = ...;
Visitor print_visitor
    = ...;
```

```
for (auto iter = tree.begin
      (order);
      iter != tree.end
      (order);
      ++iter)
    (*iter).accept
    (print_visitor);
```



# Overview of this Part of the Course

- The C++ expression tree processing app we cover is available on github



See [github.com/douglasraigschmidt/CPlusPlus/tree/master/expression-tree](https://github.com/douglasraigschmidt/CPlusPlus/tree/master/expression-tree)


---

# Other Digital Learning Resources

---


# Other Digital Learning Resources

- See my website for many more videos & screencasts related to programming with patterns, frameworks, C++, Java, etc.



**Digital Learning Offerings**

**Douglas C. Schmidt** ([d.schmidt@vanderbilt.edu](mailto:d.schmidt@vanderbilt.edu))  
Associate Chair of [Computer Science and Engineering](#),  
[Professor](#) of Computer Science, and Senior Researcher  
in the [Institute for Software Integrated Systems \(ISIS\)](#)  
at [Vanderbilt University](#)



**O'Reilly LiveTraining Courses**

- Programming with Java 8 Lambdas and Streams
  - [January 9th, 2018, 9:00am-12:00pm central time](#)
  - [February 1st, 2018, 9:00am-12:00pm central time](#)
  - March 1st, 2018, 9:00am-12:00pm central time
- Scalable Programming with Java 8 Parallel Streams
  - [January 10th, 2018, 11:00am-3:00pm central time](#)
  - February 6th, 2018, 11:00am-3:00pm central time
  - March 6th, 2018, 11:00am-3:00pm central time
- Reactive Programming with Java 8 Completable Futures
  - [January 12th, 2018, 10:00am-1:00pm central time](#)
  - [February 13th, 2018, 10:00am-2:00pm central time](#)
  - March 13th, 2018, 10:00am-2:00pm central time

**[Pearson LiveLessons](#) Courses**

- [Java Concurrency](#)
- [Design Patterns in Java](#)

**[Coursera MOOCs](#)**

- [Android App Development](#) Coursera Specialization
- [Pattern-Oriented Software Architecture](#) (POSA)

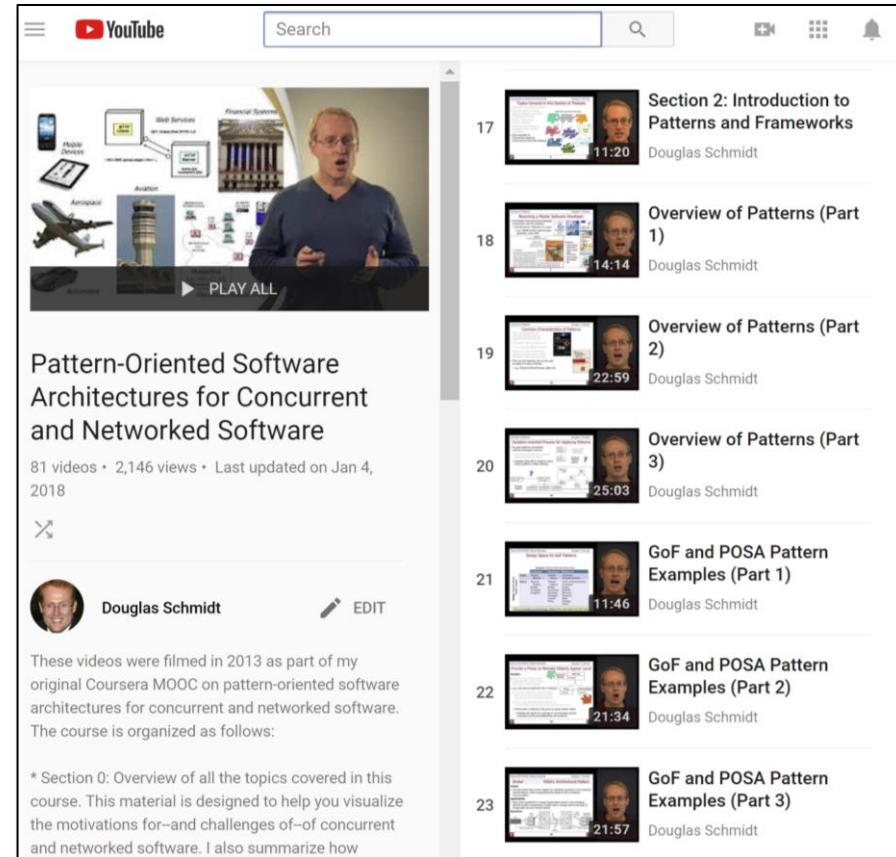
**[Vanderbilt University Courses](#)**

- [Playlist](#) from my [YouTube Channel](#) videos from [CS 891: Introduction to Concurrent and Parallel Java Programming with Android](#)
- [Playlist](#) from my [YouTube Channel](#) videos from [CS 892: Concurrent Java Programming with Android](#)
- [Playlist](#) from my [YouTube Channel](#) videos from [CS 251: Intermediate Software Design with Java](#)
- [Playlist](#) from my [YouTube Channel](#) videos from [CS 282: Concurrent Java Network Programming in Android](#)
- [Playlist](#) from my [YouTube Channel](#) videos from [CS 251: Intermediate Software Design with C++](#)
- [Playlist](#) from my [YouTube Channel](#) videos from [CS 282: Systems Programming for Android](#)

See [www.dre.vanderbilt.edu/~schmidt/DigitalLearning](http://www.dre.vanderbilt.edu/~schmidt/DigitalLearning)

# Other Digital Learning Resources

- See my website for many more videos & screencasts related to programming with patterns, frameworks, C++, Java, etc.
- Videos from my MOOC “Pattern-Oriented Software Architecture” are relevant!

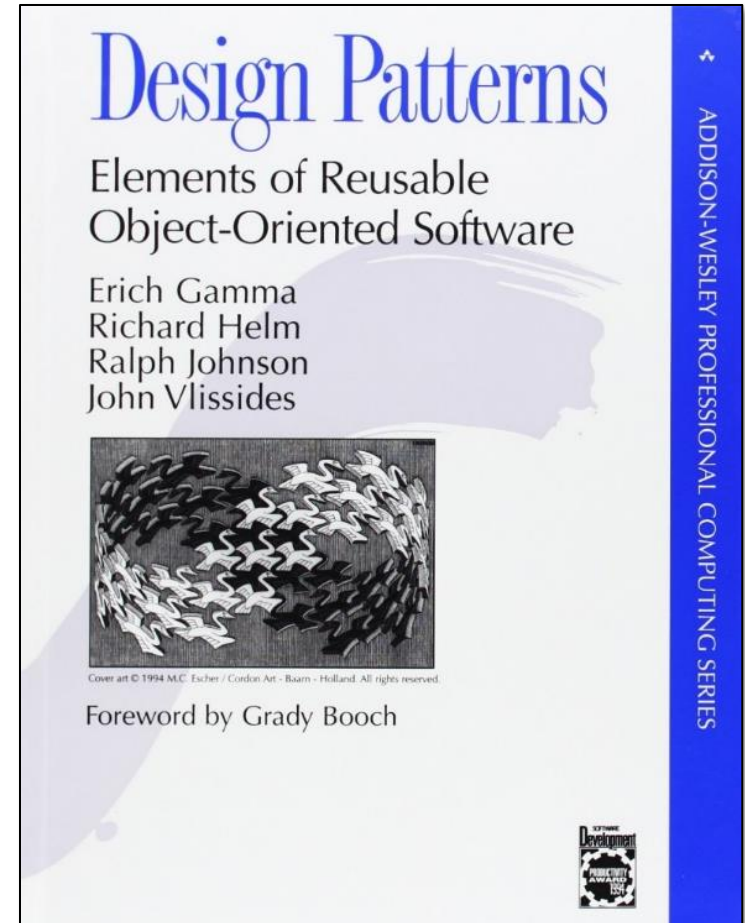


The screenshot shows a YouTube playlist page. The main video player displays a man (Douglas Schmidt) speaking, with a 'PLAY ALL' button below it. The video title is 'Pattern-Oriented Software Architectures for Concurrent and Networked Software'. Below the title, it says '81 videos • 2,146 views • Last updated on Jan 4, 2018'. The channel name 'Douglas Schmidt' is visible. A description follows: 'These videos were filmed in 2013 as part of my original Coursera MOOC on pattern-oriented software architectures for concurrent and networked software. The course is organized as follows: \* Section 0: Overview of all the topics covered in this course. This material is designed to help you visualize the motivations for—and challenges of—of concurrent and networked software. I also summarize how'. To the right, a list of videos is shown, including 'Section 2: Introduction to Patterns and Frameworks', 'Overview of Patterns (Part 1)', 'Overview of Patterns (Part 2)', 'Overview of Patterns (Part 3)', 'GoF and POSA Pattern Examples (Part 1)', 'GoF and POSA Pattern Examples (Part 2)', and 'GoF and POSA Pattern Examples (Part 3)'.

See [www.youtube.com/playlist?list=PLZ9NgFYEMxp6CHE-QQ040tIDILNcBqJnc](https://www.youtube.com/playlist?list=PLZ9NgFYEMxp6CHE-QQ040tIDILNcBqJnc)

# Other Digital Learning Resources

- The book *Design Patterns: Elements of Reusable Object-Oriented Software* describes all the Gang-of-Four (GoF) patterns in detail



See [en.wikipedia.org/wiki/Design\\_Patterns](https://en.wikipedia.org/wiki/Design_Patterns)

# Other Digital Learning Resources

- The “POSA” books contain good sources of material on other types of patterns & pattern relationships



See [www.dre.vanderbilt.edu/~schmidt/POSA](http://www.dre.vanderbilt.edu/~schmidt/POSA)

---

# End of Course Overview

---