The Singleton Pattern
Structure & Functionality

Douglas C. Schmidt
Learning Objectives in This Lesson

- Recognize how the `Singleton` pattern can be applied to centralize access to global resources.
- Understand the structure & functionality of the `Singleton` pattern.

```java
If (uniqueInstance == null)
    uniqueInstance = new Singleton();
return uniqueInstance;
```
Structure & Functionality of the Singleton Pattern

Douglas C. Schmidt
Intent
• Ensure a class has only one instance & provide a global point of access

See en.wikipedia.org/wiki/Singleton_pattern
Applicability

- When there must be exactly one instance of a class & it must be accessible from a well-known access point.

**Singleton**

**GoF Object Creational**

**Reactor**
- run_event_loop()
- end_event_loop()
- instance()

**ET_Event_Handler**
- handle_input()
- prompt_user()
- receive_input()
- make_command()
- execute_command()

**Verbose_Mode**
- ET_Event_Handler
- prompt_user()
- make_command()

**Succinct_Mode**
- ET_Event_Handler
- prompt_user()
- make_command()
Applicability

• When there must be exactly one instance of a class & it must be accessible from a well-known access point

• When the sole instance should be extensible by subclassing & clients should be able to use an extended instance without modifying any code
If (uniqueInstance == null)
uniqueInstance = new Singleton();
return uniqueInstance;
If (uniqueInstance == null)
  uniqueInstance = new Singleton();
  return uniqueInstance;
Performing a “first-time-in check”

If (uniqueInstance == null)
uniqueInstance = new Singleton();
return uniqueInstance;

See [en.wikipedia.org/wiki/Lazy_initialization](en.wikipedia.org/wiki/Lazy_initialization)