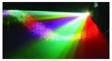
Supplement to
The Design and Implementation of Multimedia Software

The Singleton Pattern

Prof. David Bernstein

James Madison University

users.cs.jmu.edu/bernstdh



Motivation

- Some applications require exactly one instance of a class:

  Windowing systems with one event queue

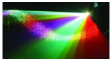
  Word processors with one menu bar (for all documents)

- Unfortunately, constructors can be called repeatedly:

  In one method

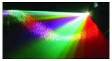
  In one class

  Across multiple classes

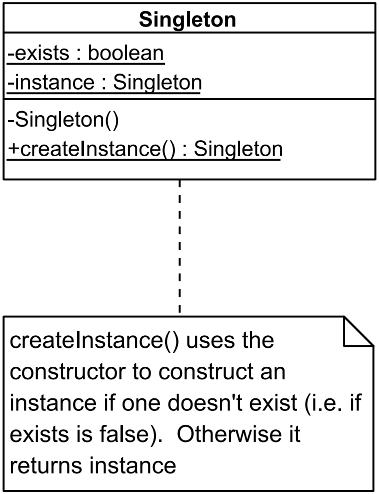


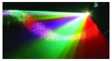
Requirements of a Singleton

- Create and return an instance of itself if one doesn't exist
- Return the existing instance if one does exist



One Implementation of the Singleton Pattern

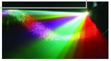

Singleton
-exists : boolean
-instance : Singleton
-Singleton()
+createInstance() : Singleton
createInstance() uses the constructor to construct an instance if one doesn't exist (i.e. if exists is false). Otherwise it returns instance



Example - A FileViewer

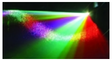
```
public class FileViewer
{

    private static boolean      exists = false;
    private static FileViewer   instance;

    private FileViewer()
    {


        exists = true;
    }

}
```

```
    public static FileViewer createInstance()
    {
       if (!exists) instance = new FileViewer();
       return instance;

    }
```



Example - Using a `FileViewer`

```
public void valueChanged(ListSelectionEvent lse)
{
    FileViewer   fv;
    String       fn;

    fn = (String)list.getSelectedValue();

    fv = FileViewer.createInstance();
    fv.load(fn);
}
```



Thread Safety

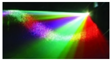
- A Potential Problem:

  If the `createInstance()` method might be called by multiple threads, problems can arise

- Resolutions:

  Make `createInstance()` synchronized

  Use eager initialization (i.e., instantiate `instance` when it is declared)



Example - Using Eager Instantiation

```
private static FileViewer   instance = new FileViewer();
```

```
public static FileViewer createInstance()
{
    return instance;
}
```