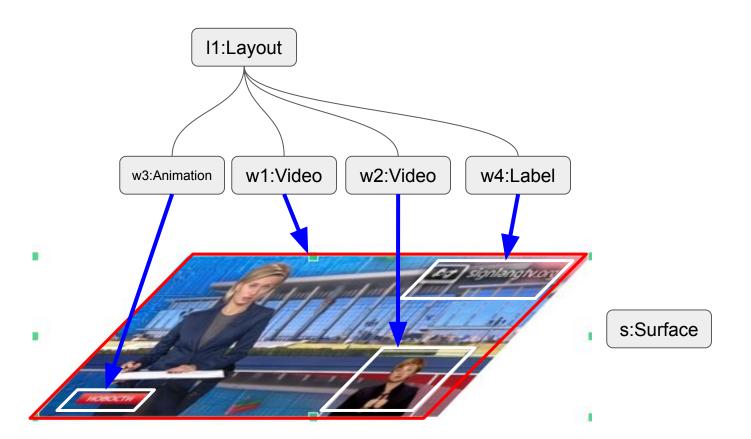
WATERMARK **VIDEO LOWER THIRD**

TICKER







s:Surface



...60...



1 seg

- Es un objeto complejo con dos Graphic Context para hacer "bit blit"
- Los widgets deben dibujar en la misma superficie pero tambien otras partes del "sistema"
 - o Por ejemplo para agregar mensajes como
 - "Verifique su conexion"
 - "Factura Impaga"
 - Por ejemplo para controlar fps o cache de memoria

Singleton

Intención:

 Asegurar que una clase solo tiene una instancia y provee una manera uniforme de acceder a ella.



Detalles de Implementación (Java)

- Constructor privado para restringir la creación de instancias de la clase de otras clases.
- Variable estática privada de la misma clase que es la única instancia de la clase.
 - Eager Initialization
 - Static block con try/catch
 - Lazy Initialization
- Método estático público que devuelve la instancia de la clase, este es el punto de acceso global para que el mundo exterior obtenga la instancia de la clase singleton.
 - Lugar para implementar Lazy Initialization

```
public class Surface {
         private static Surface instance;
         private static Frame[] frames = new Frame[2];
         private int current_idx;
         private Surface(){
             //frames[1] = new Frame("4k");
             //frames[2] = new Frame("4k");
             //current idx = 1;
 8
 9
10
         static {
             try{instance = new Surface();
11
             }catch (Exception e){
12
                 throw new RuntimeException("Unable to create Surface");
13
14
15
16
         public static Surface getInstance(){return instance;}
17
         public void bitblit(Bitmap bitmap){
18
             //blit bitmap onto the current frame
19
             11 . . .
20
21
22
23
```

```
Eager Initialization + Static block
```

```
public static synchronize Surface getInstance(){
   if(instance == null){
      instance = new Surface();
   }
   return instance;}
```

Lazy Initialization

Singleton vale la pena?

Si!

- Permite tener una única instancia de Surface
- Cualquier método de cualquier objeto se puede acceder al Singleton

No!

- Es dificil de implementar el ciclo de vida del Singleton
 - Como/cuando se resetea?
- Genera acoplamiento con la clase y su instancia

Wrap Up

- Patrones Creacionales tratan con problemas de "construir/instanciar" modelos de objetos complejos (productos)
 - Los patrones se describen con <u>roles</u>
- En FactoryMethod una jerarquía de Creators cada uno define métodos que crean productos
- En Builder se distingue un Director y una jerarquía Builders. El Director pide partes, un builder provee esas partes y eventualmente el producto completo
- En Singleton una clase se asegura que solo existe una instancia
- GoF define otros Patrones Creacionales
 - Abstract Factory
 - Prototype