

# Caciocavallo

## Portable GUI backends



Mario Torre  
Roman Kennke  
2009-02-02 12:52:63.654

# Caciocavallo

**Portable GUI backends**  
(or, how we became famous, but no one ever cared)



Mario Torre  
Roman Kennke  
2009-02-02 12:52:63.654

# Overview

- History and background
- AWT peer overview
- Classpath and OpenJDK Implementation overview
- Caciocavallo architecture
- Demo and Examples



# One year ago



## **What is it?**

- A very delicious kind of south italian cheese
- A portable AWT peer implementation
- Quite cool

# History Fluff

- Previous implementation in GNU Classpath (aka Swing peers)
- 1 week before Fosdem 2008: prototype planning
- Fosdem 2008: Broken knife.
- Finished coding
- April 2008: Started OpenJDK Challenge
- May 2008: Another broken knife
- August 2008: Finished OpenJDK Challenge with Bronze

# OpenJDK Challenge

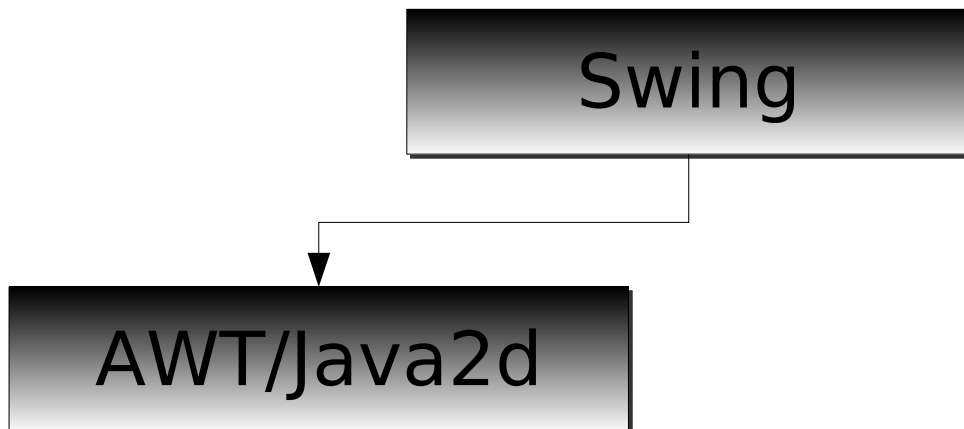
- Original goal of project: Analyse and fix OpenJDK AWT & Swing code wrt external Toolkits
- This part is finished and mostly merged into OpenJDK7 (except FontManager)
- New goal: Create portable AWT toolkit based on Swing widgets

# General AWT peer overview

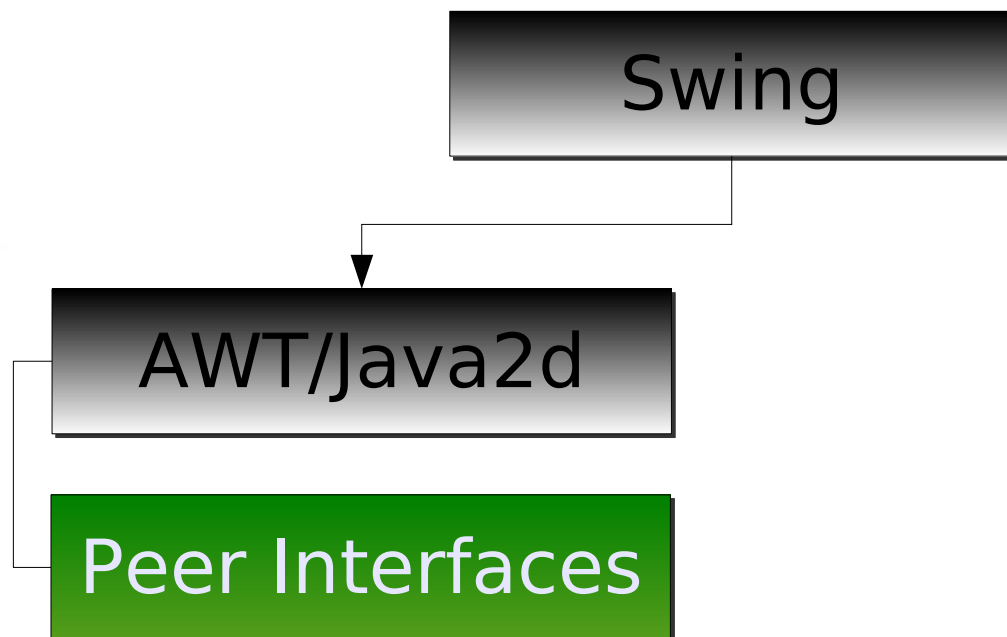
# General AWT peer overview

Swing

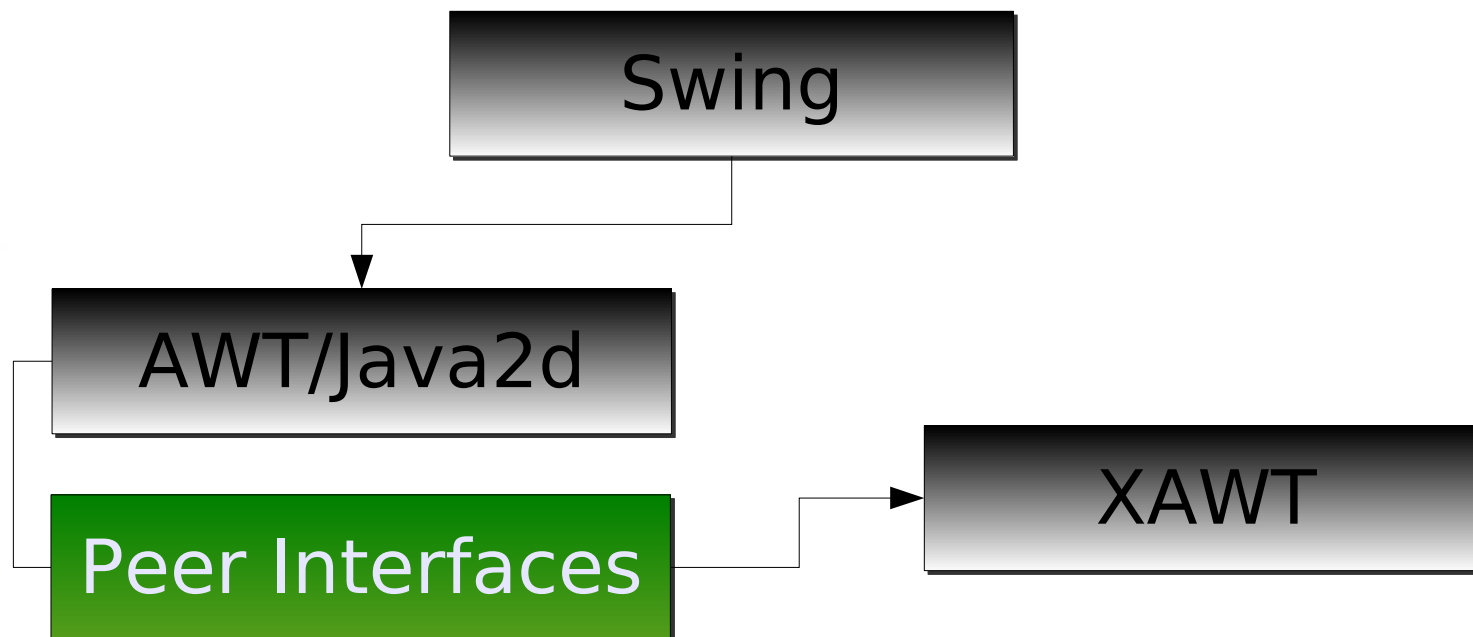
# General AWT peer overview



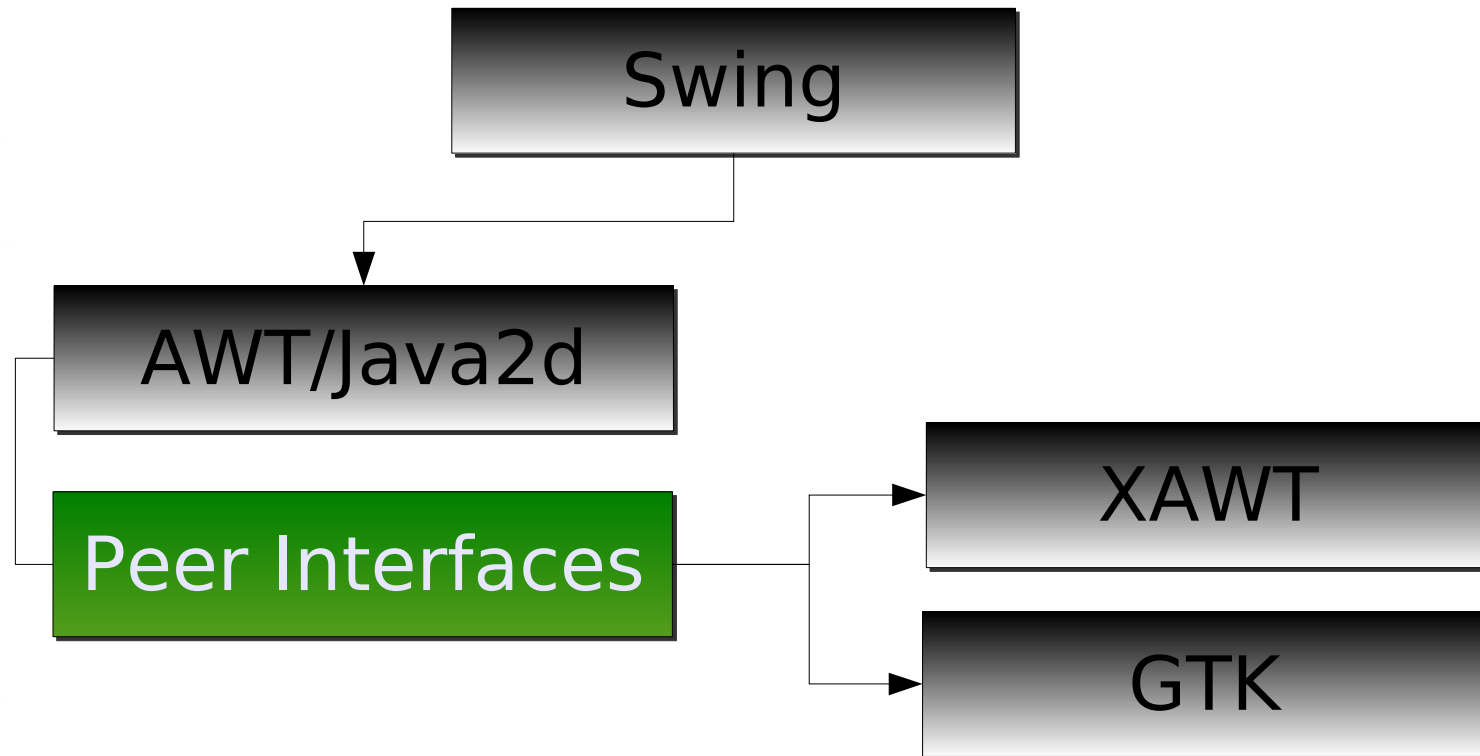
# General AWT peer overview



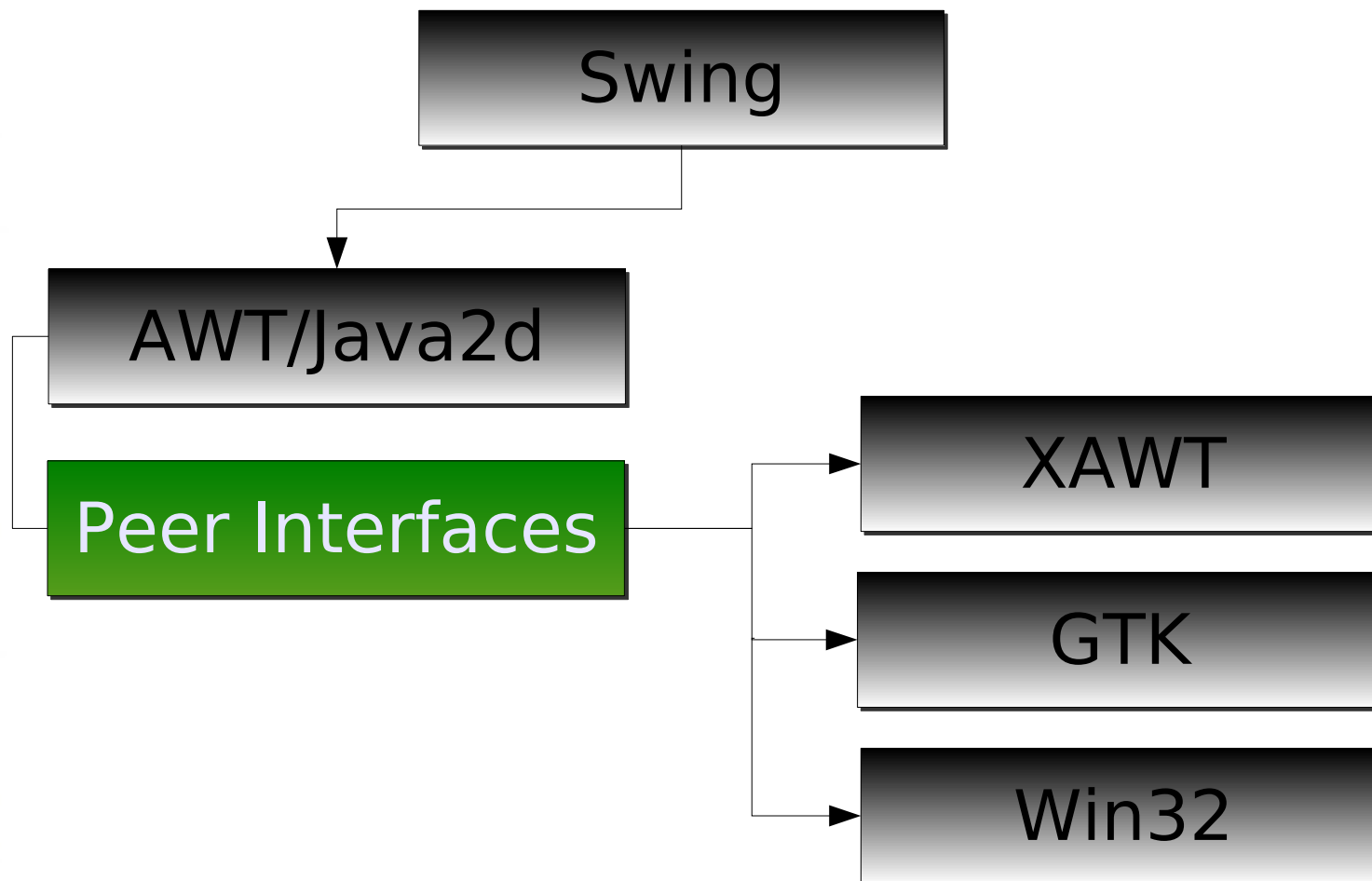
# General AWT peer overview



# General AWT peer overview



# General AWT peer overview



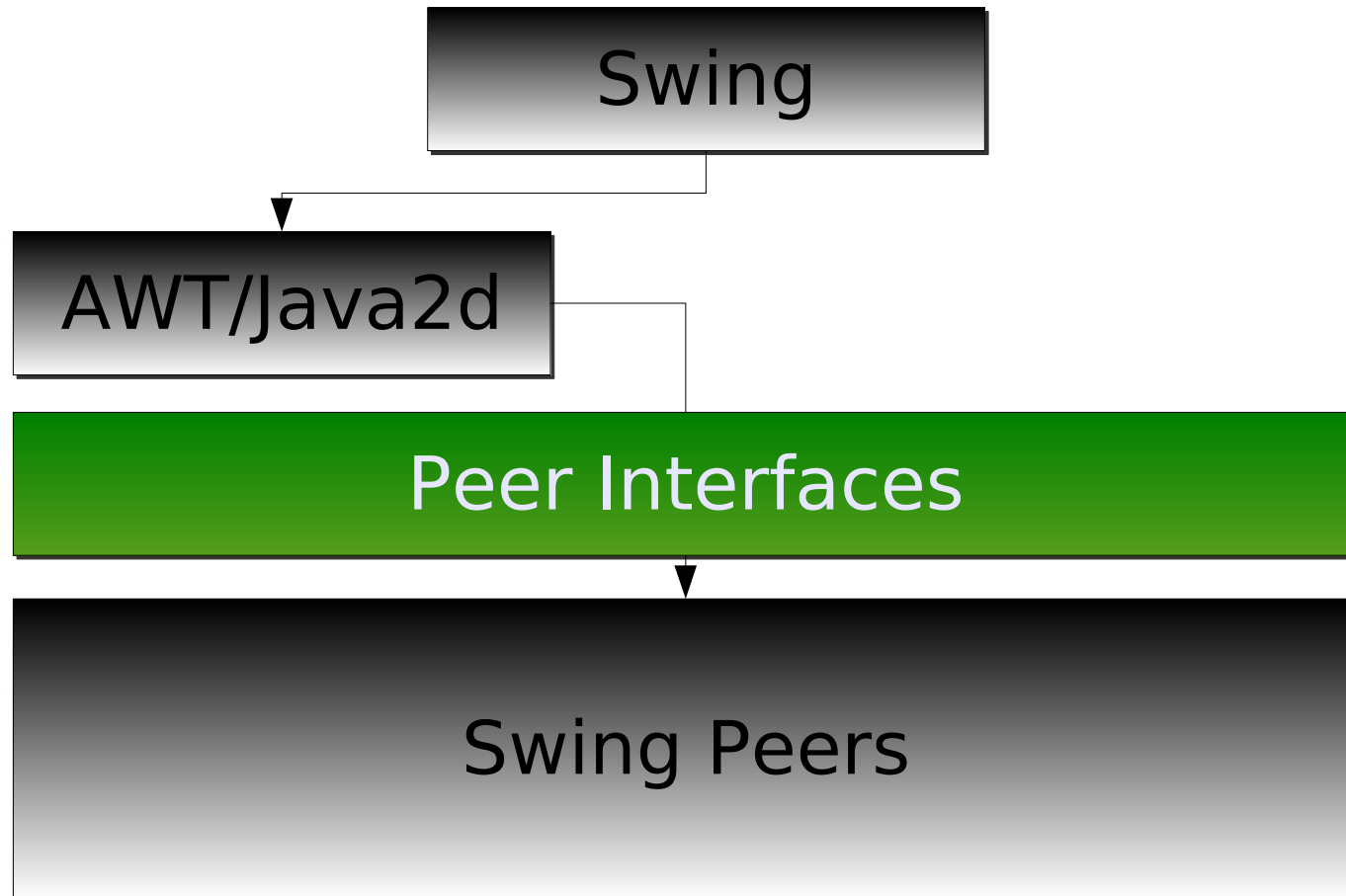
# Heavyweight vs lightweight

- Heavyweight: each widget has its own native window
- Lightweight: only toplevel windows have native window
- AWT widgets are supposed to behave heavyweight

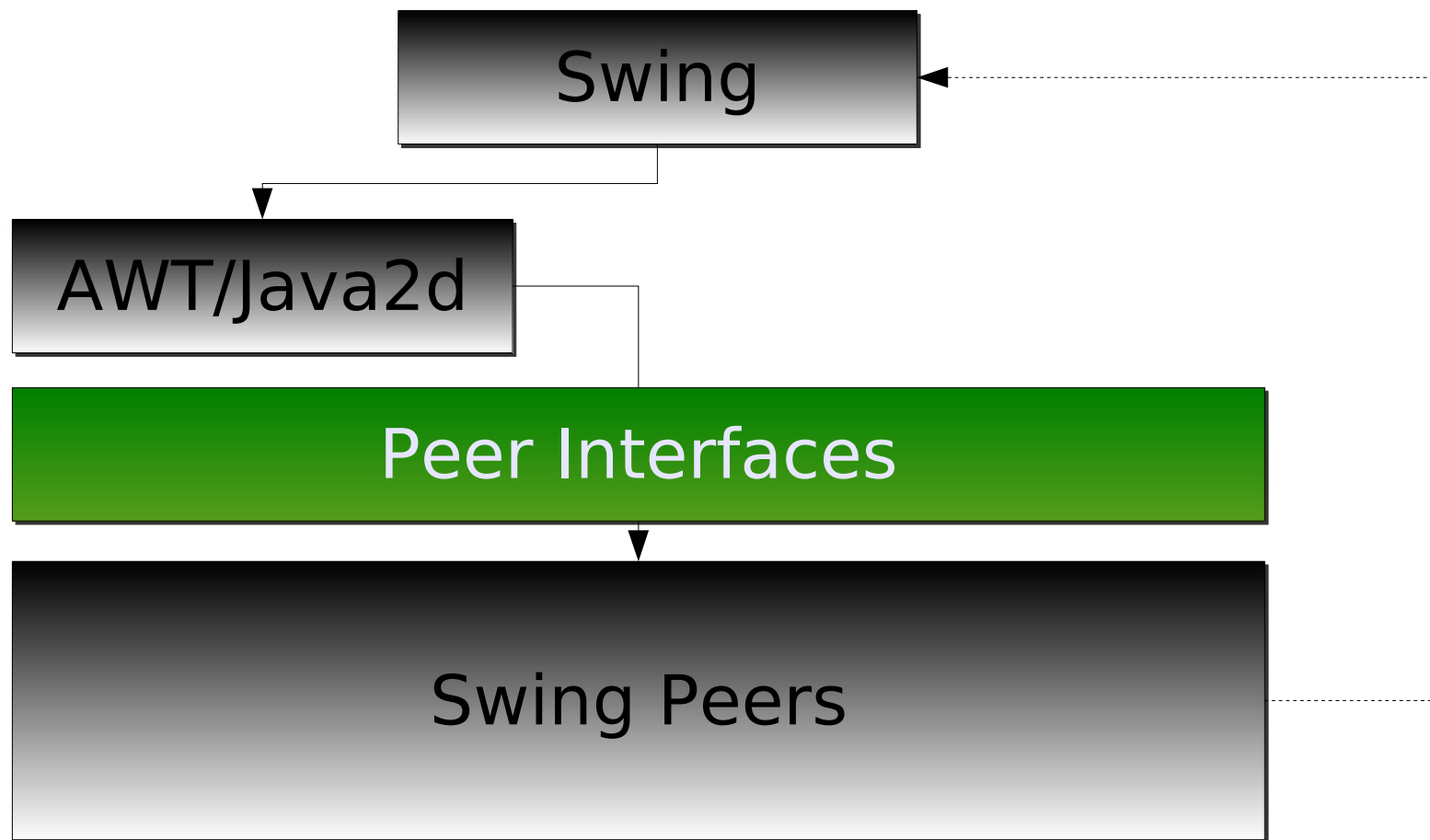
# Related implementations

- ClasspathSwing Peers
- OpenJDK
- Caciocavallo (based on concepts of borh)

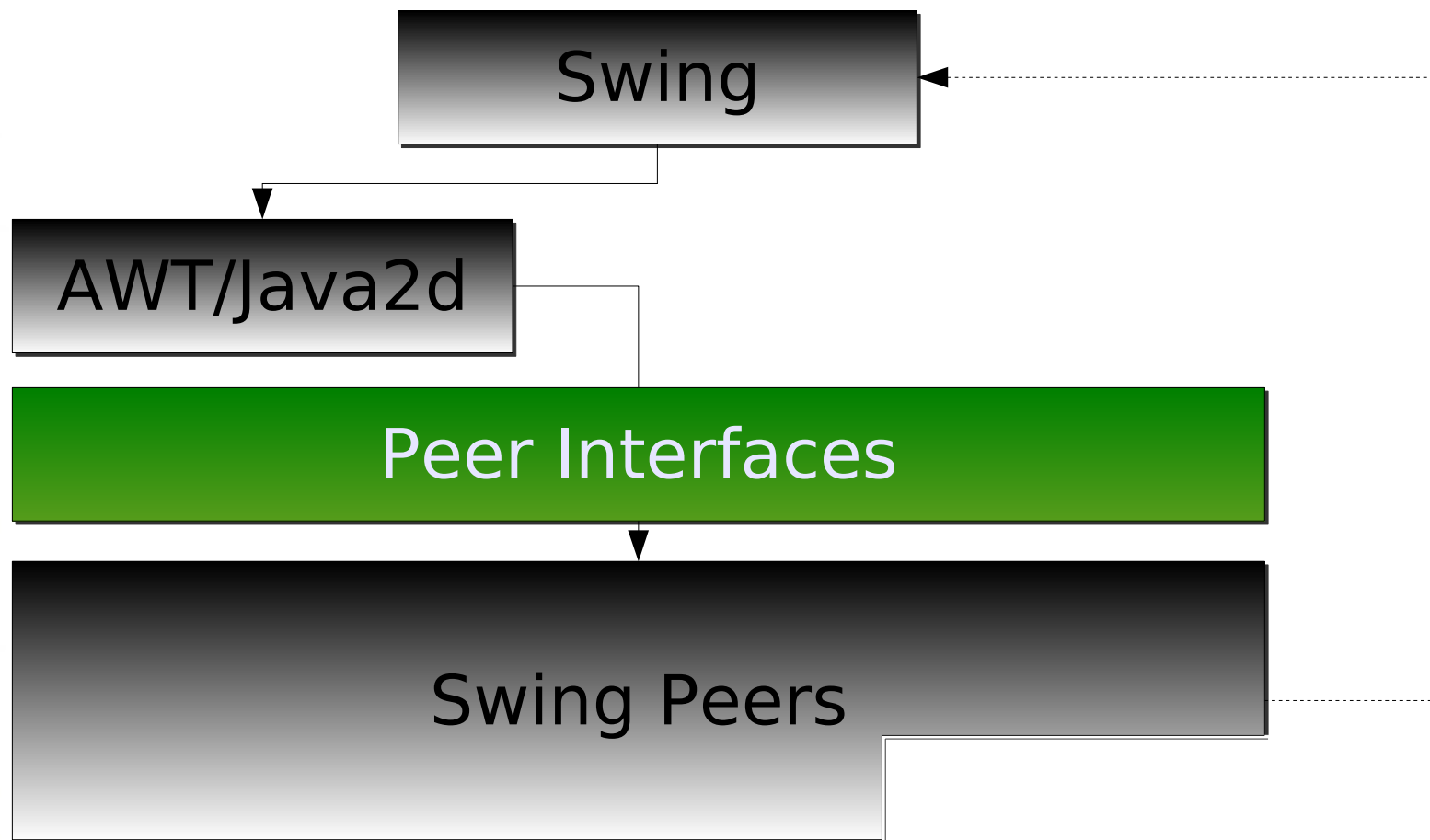
# Classpath Swing peers



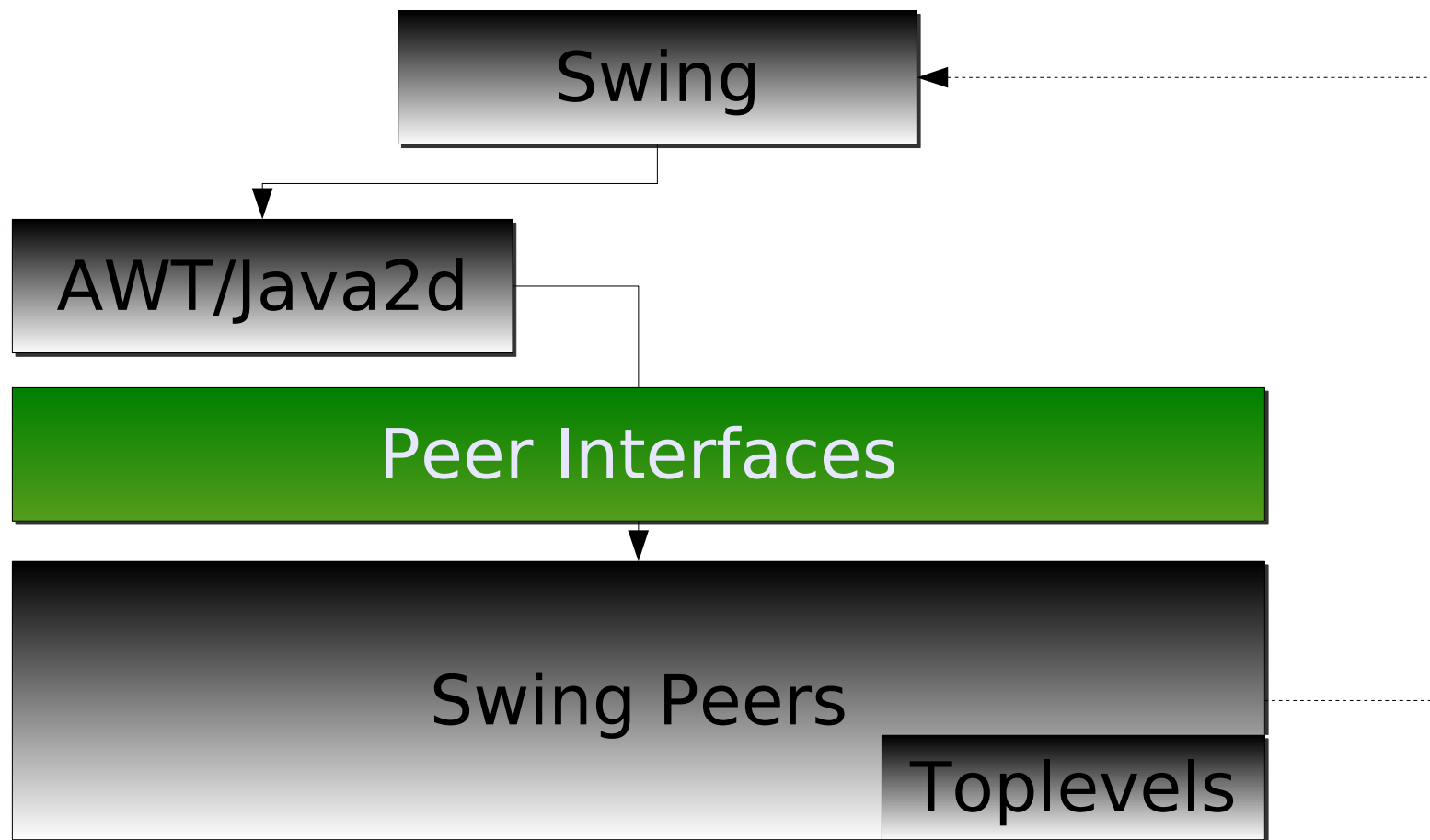
# Classpath Swing peers



# Classpath Swing peers



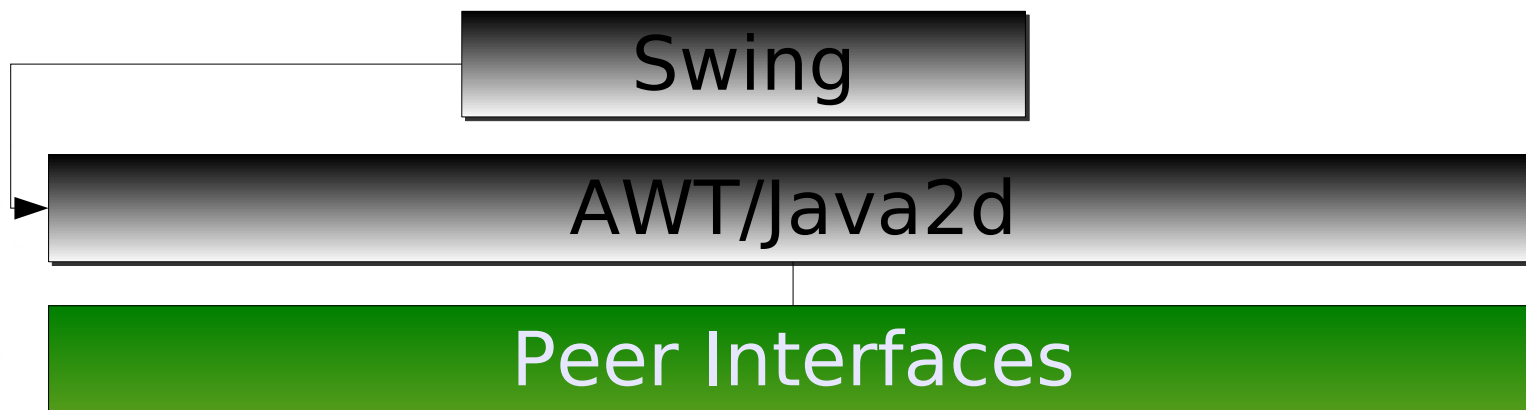
# Classpath Swing peers



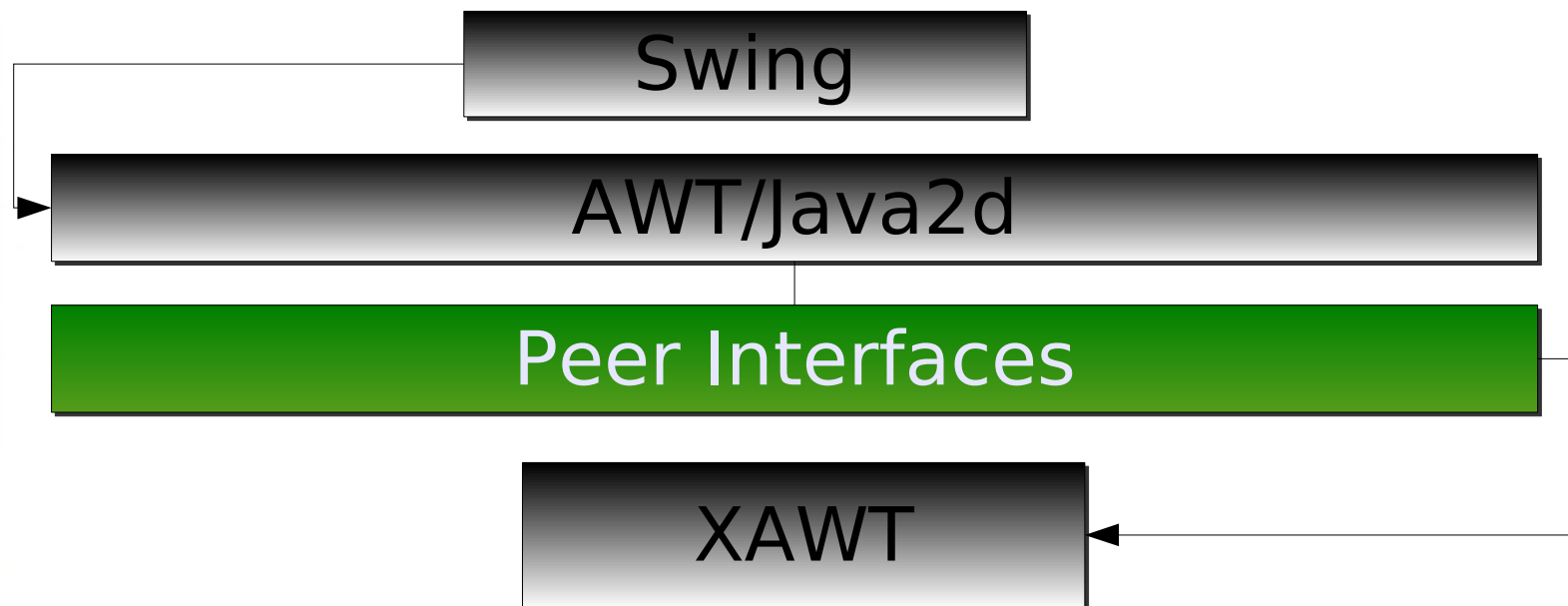
# Classpath peers

- Advantages:
  - Portable
- Disadvantages:
  - Lightweight
  - Not straightforward to implement on actual target

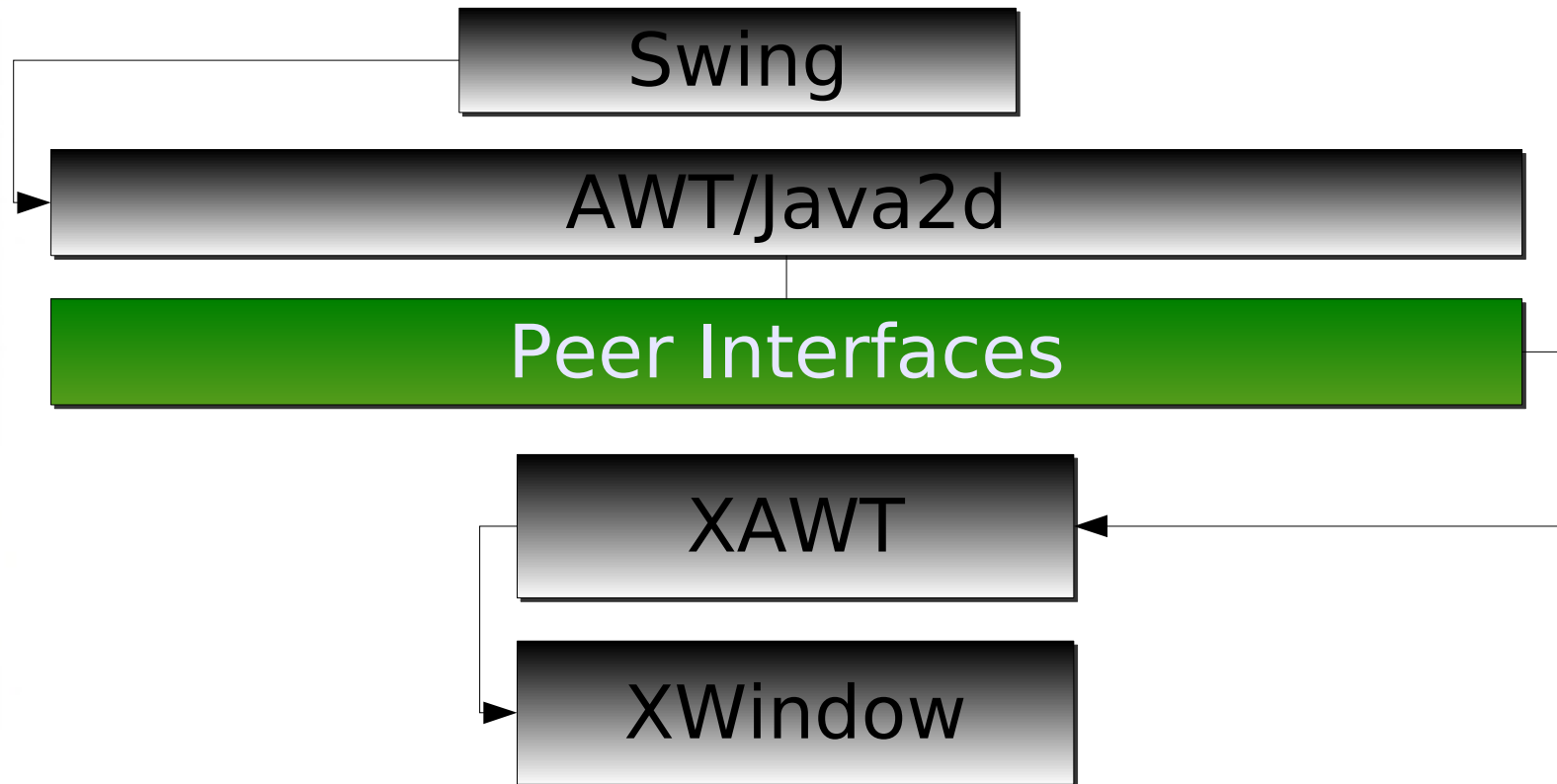
# OpenJDK XAWT peers



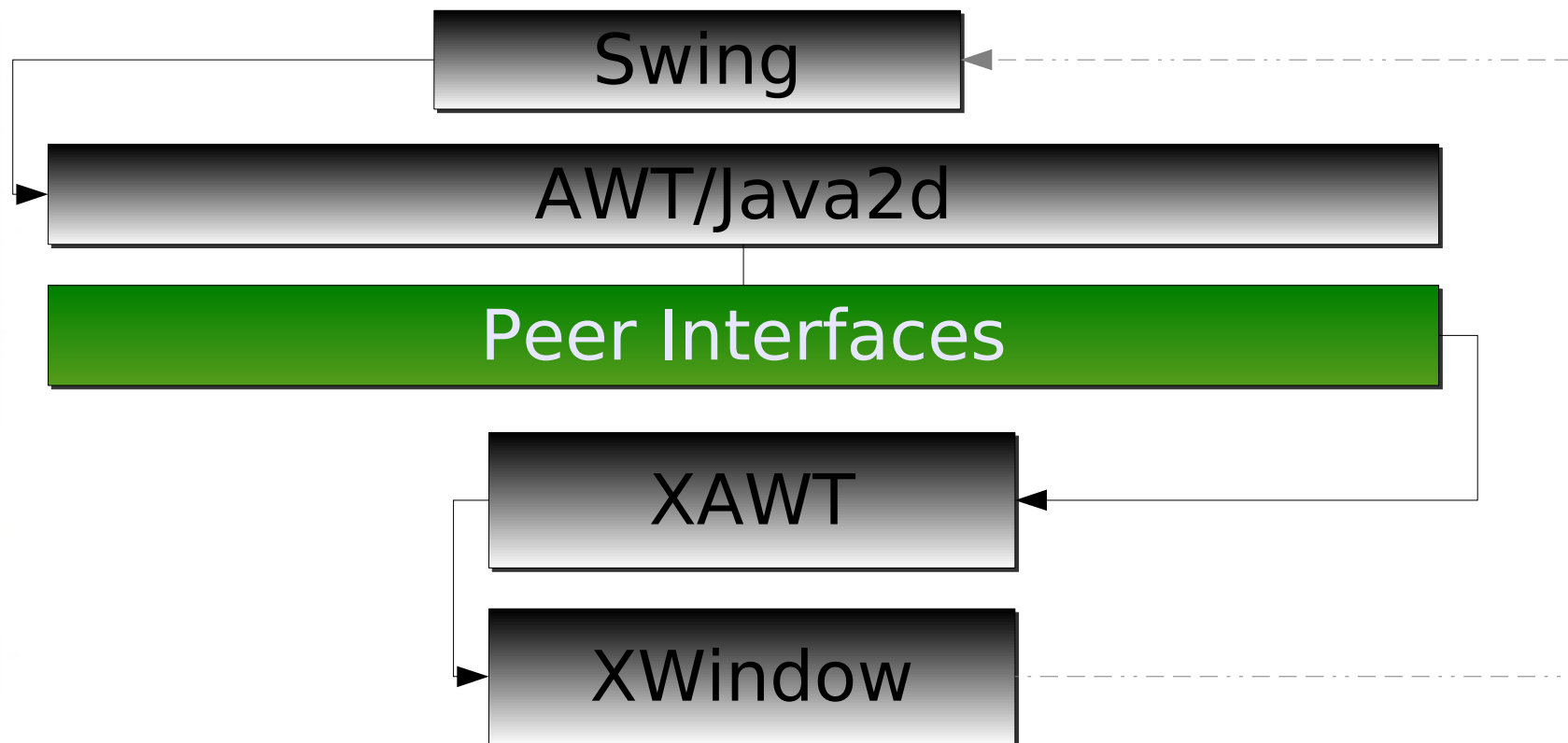
# OpenJDK XAWT peers



# OpenJDK XAWT peers



# OpenJDK XAWT peers



# OpenJDK XAWT peers

- Advantages:
  - Heavyweight
  - No other native dependencies other than X11
- Disadvantages:
  - Tied to X11

## Why Caciocavallo?

- Some platforms have no (suitable) widget implementation at all
- Porting to new platforms should be as easy as possible
- Fast prototyping should be always possible

# Caciocavallo

Classpath Swing peers + OpenJDK XAWT peers

=

Caciocavallo



# Caciocavallo

Classpath Swing peers + OpenJDK XAWT peers

=

Caciocavallo

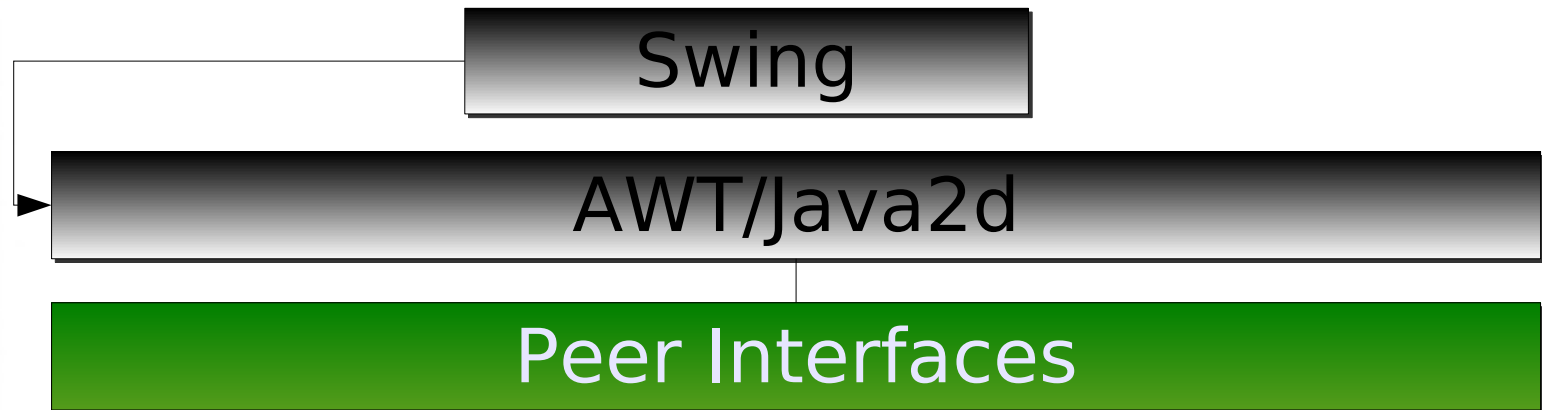


Note: no cheese was harmed while doing this presentation... but maybe it was before

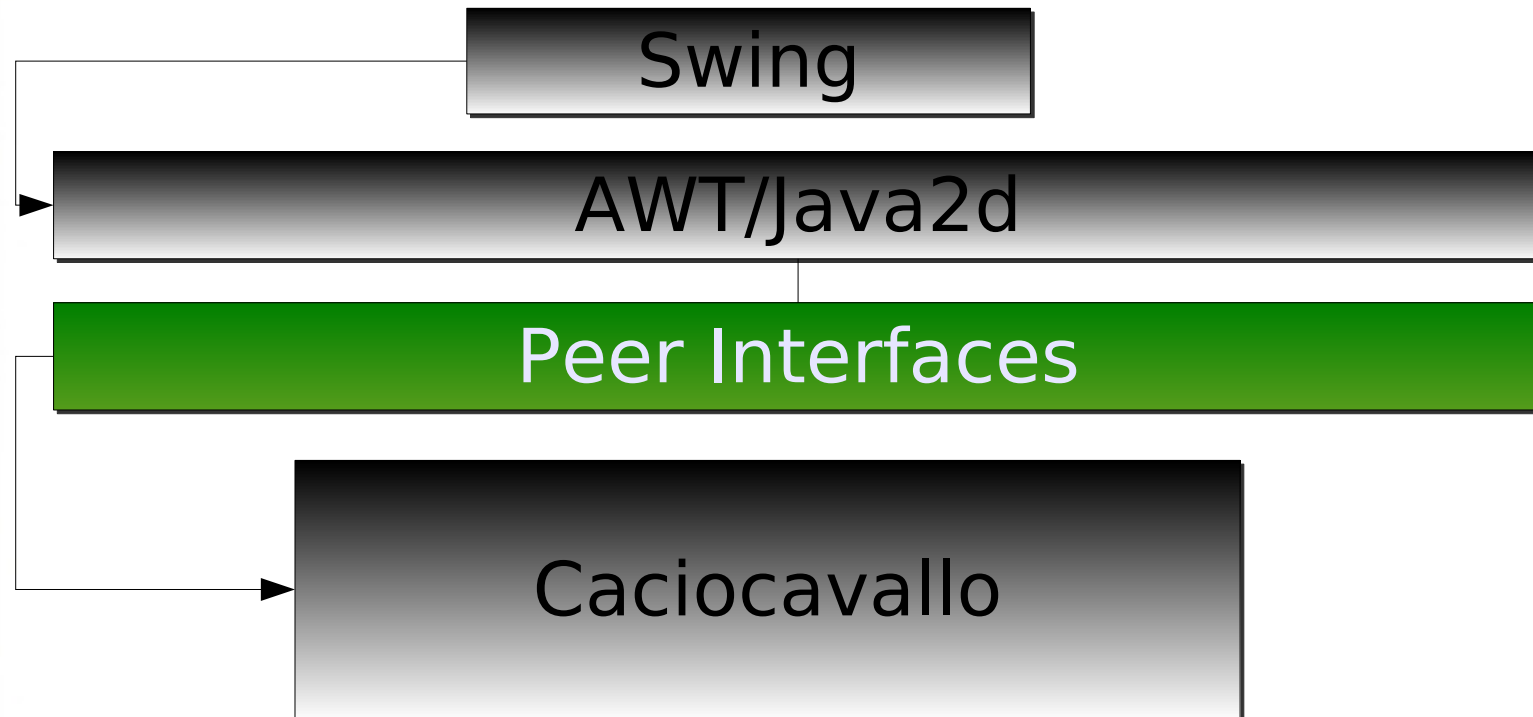
## How?

- Let Swing do drawing and logic (idea from GNU Classpath's Swing peers)
- Let each widget have its own native window (idea from XAWT peers)
- Separate things through interfaces

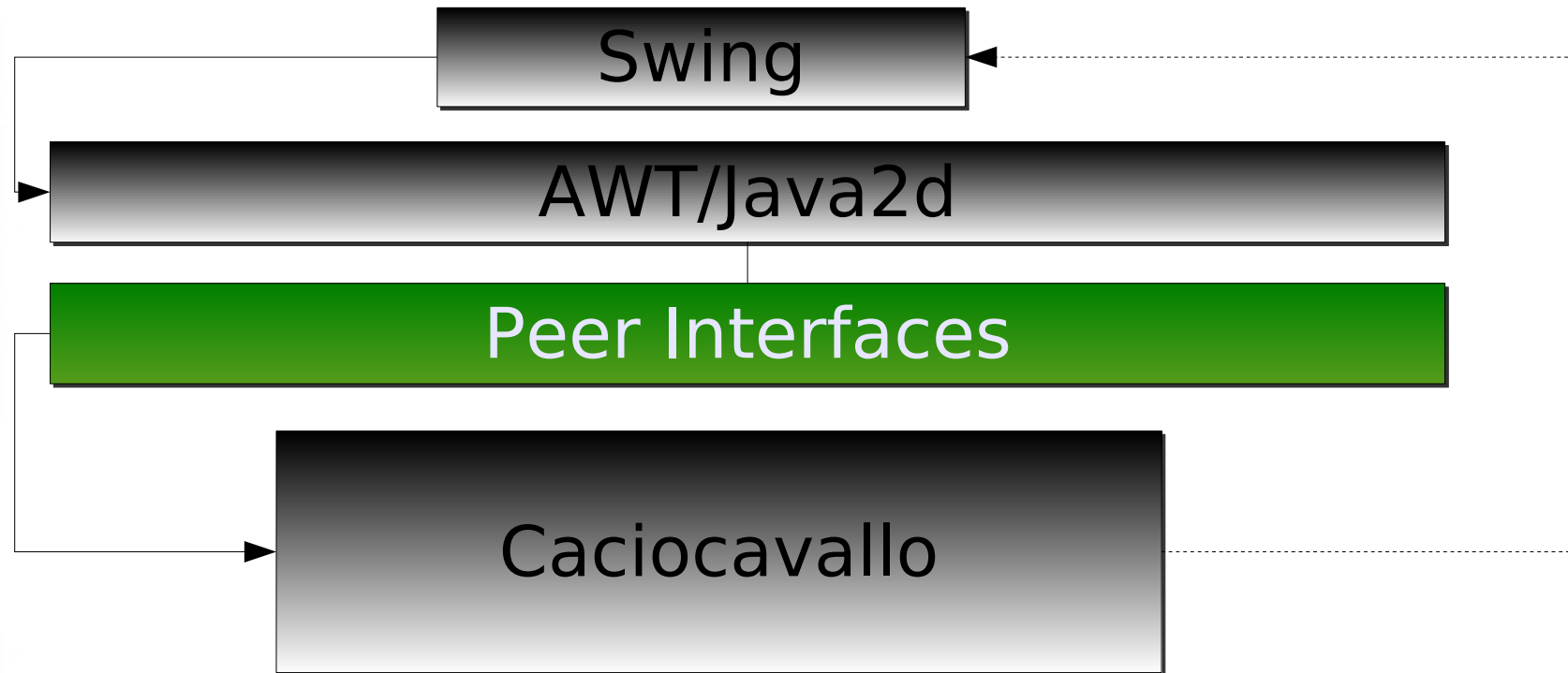
# Architecture overview



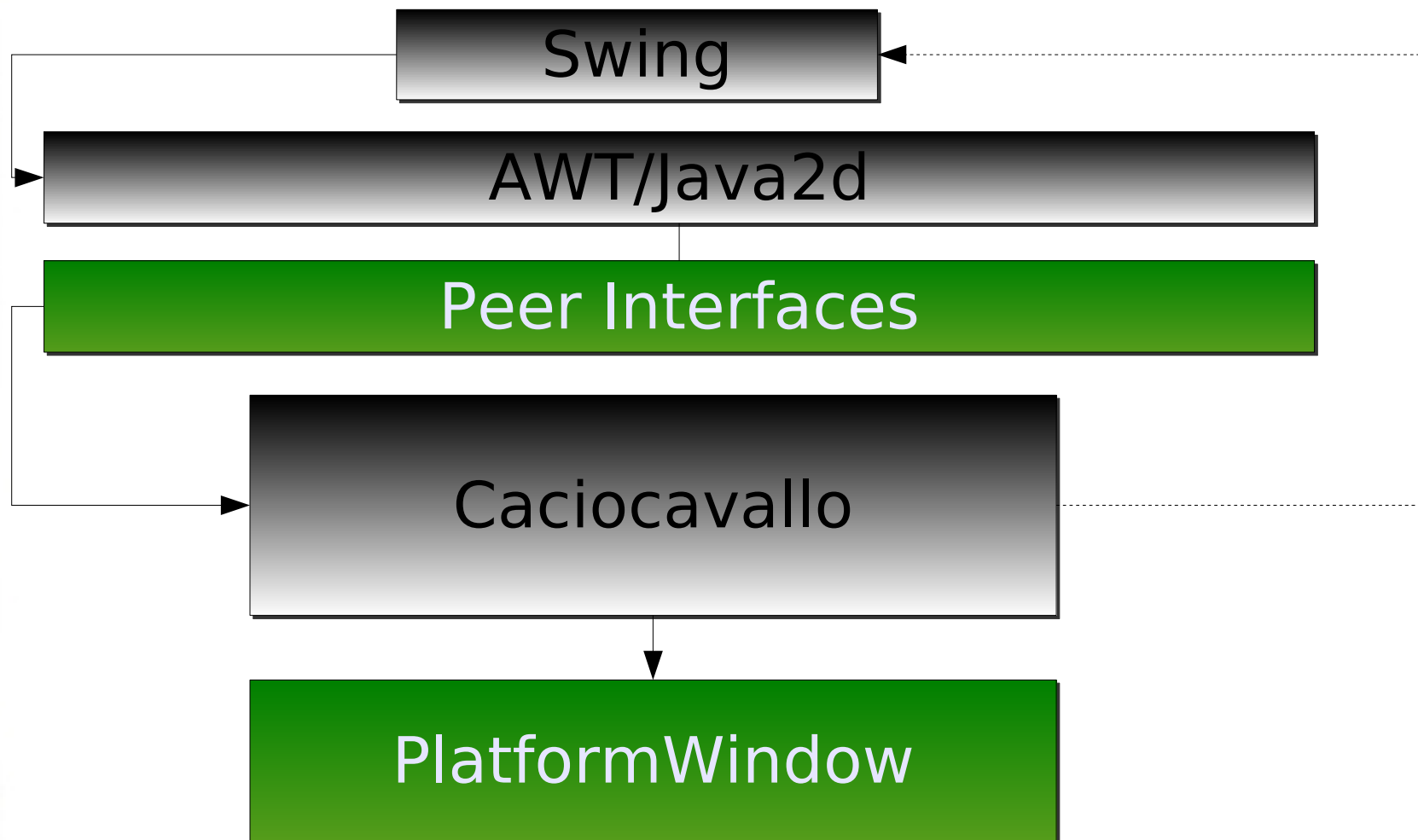
# Architecture overview



# Architecture overview



# General AWT peer overview



# PlatformWindow

- PlatformWindow defines all basic windowing behaviour
- Implementations only need to implement PlatformWindow
- Much smaller interface than the peers
- ~20 methods vs. >100 methods

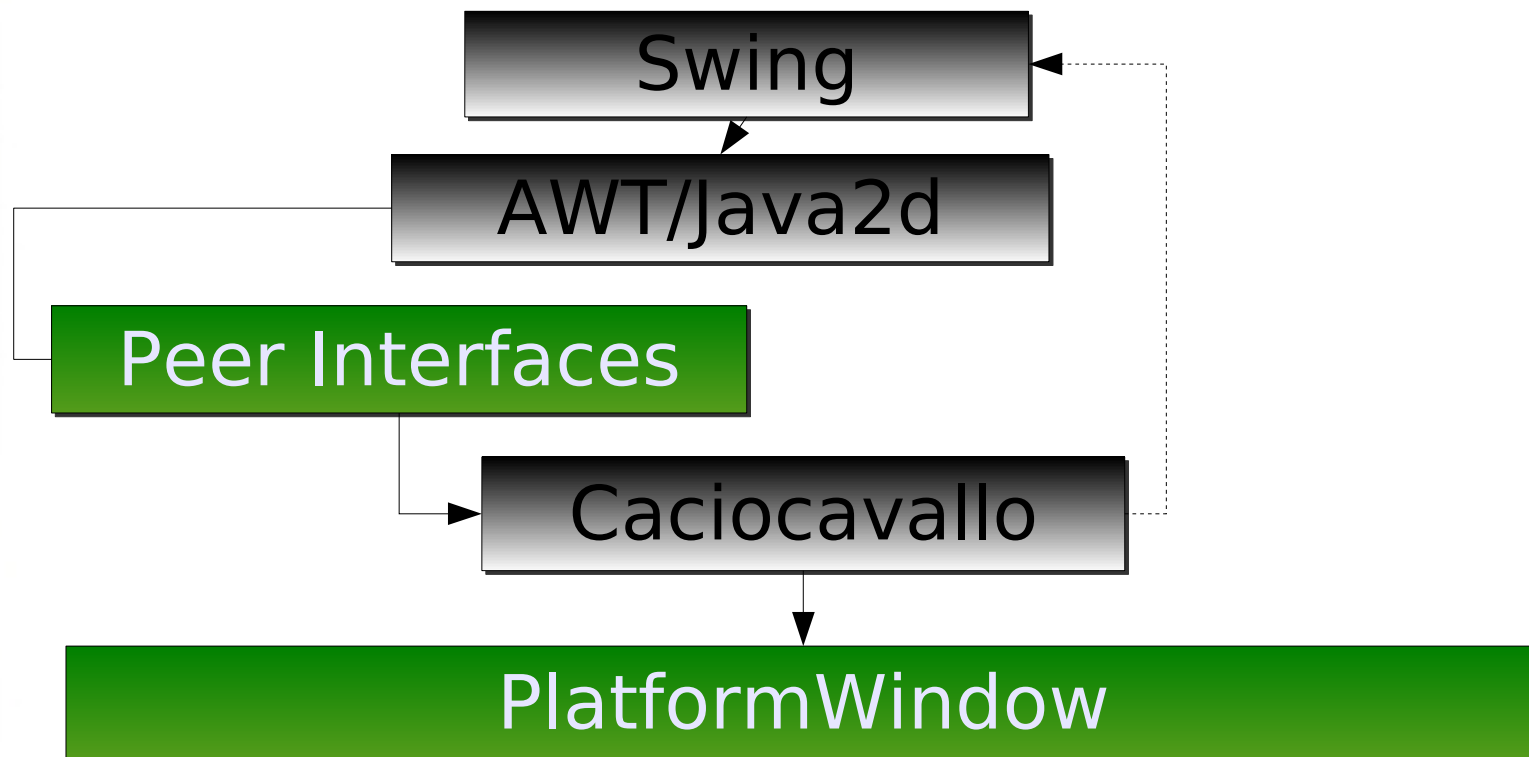
# PlatformWindow

- `void setVisible(boolean v);`
- `void setBounds(int x, int y, int w, int h);`
- `Graphics2D getGraphics();`
- etc.

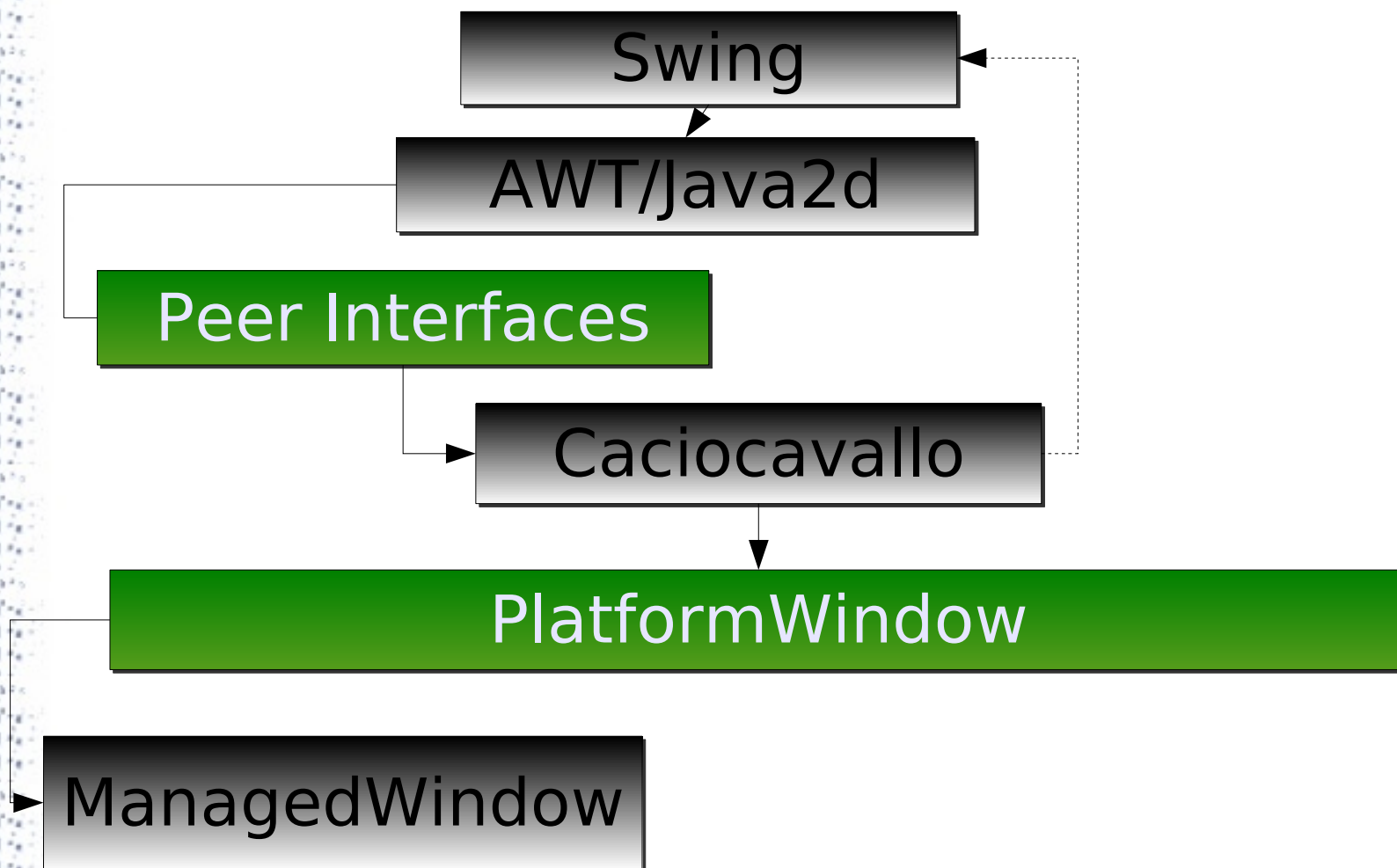
# PlatformWindow

- What if the actual target platform doesn't even support windows?
- Think: framebuffers, embedded targets, etc.

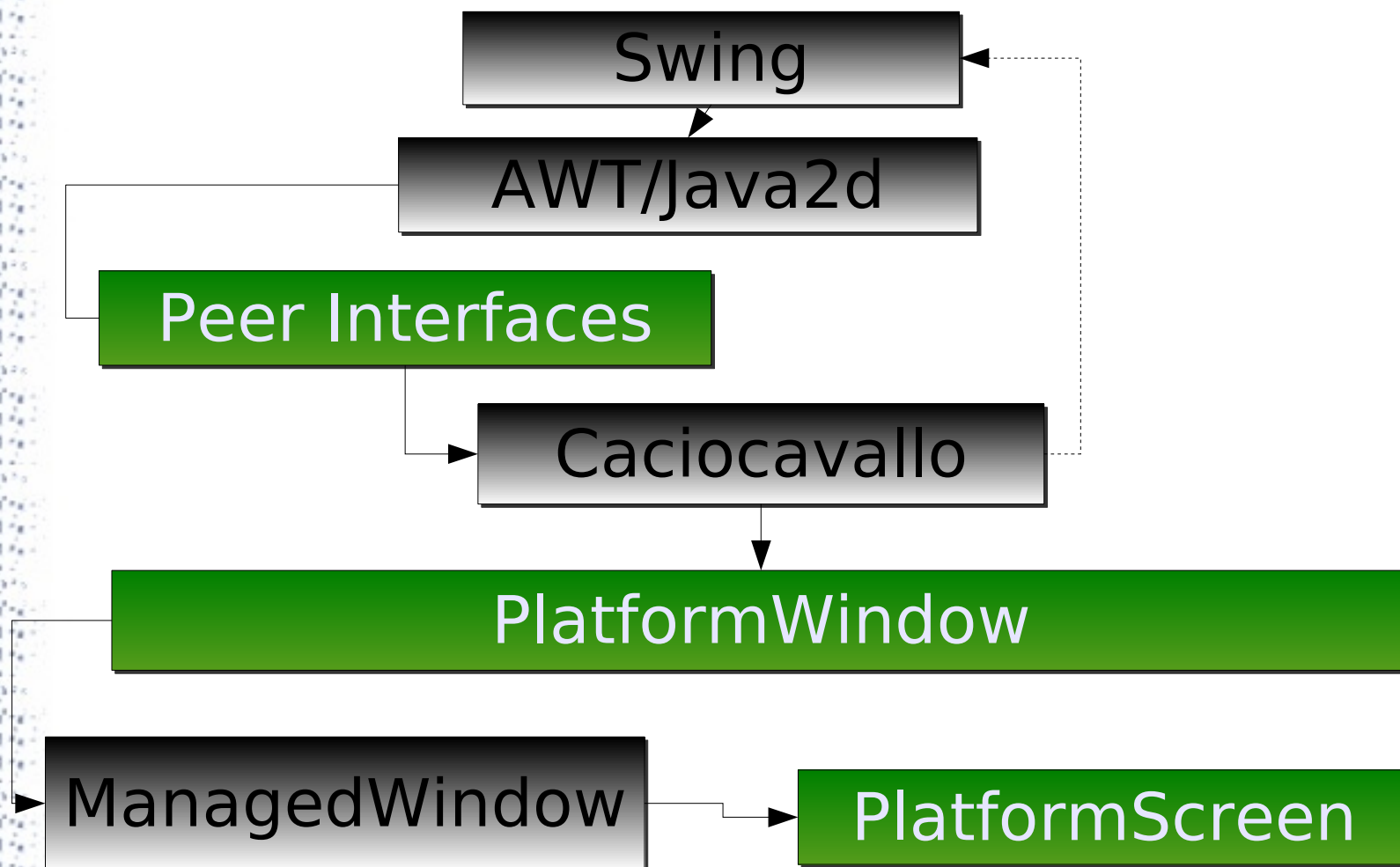
# Architecture overview



# Architecture overview



# Architecture overview



# PlatformScreen

- Subset of PlatformWindow
- Defines a container for managed windows
- Very small: ~5 methods
- Allow very fast prototyping

# Advantages of Caciocavallo

- Small
- Delegates all the fluff to who has to care!
- Allow very fast prototyping
- Very flexible

# Use Cases and Examples

- VxWorks WindML Toolkit
- DirectFB Toolkit
- VxWorks MiniX OpenGL Toolkit
- QT 4
- Aqua

## **Where we are, and where we go**

- Basic architecture is fixed
- Not complete... yet
- Completing remaining widgets



**Questions?**

The background of the slide is a collage of images. At the top, there is a checkered racing flag, a person's hands typing on a laptop, and a gear. Below these, a large, vertical stream of white data points or code characters flows down the left side of the slide. The overall color scheme is blue and white.

**Authographs anyone?**

# **Authographs anyone?**

We can provide our phone numbers too :)



**Thank you...**

The background of the slide is a collage of images. At the top, there is a checkered racing flag, a person's hands typing on a laptop keyboard, and a gear. In the upper left, an airplane is flying. The bottom half of the slide is a white background with a vertical strip of blue and white digital data on the left side.

**Thank you...**

... for sleeping

The background of the slide is a collage of images. At the top left, there is a white airplane flying against a blue sky. In the center, a person's hands are shown typing on a laptop keyboard. To the right, there are several interlocking gears. The bottom half of the slide is filled with a dense, vertical stream of small, white, pixelated characters, resembling a digital data stream or a waterfall of code.

**Thank you...**

... for sleeping

That's all folks...

The background of the slide is a collage of images. At the top, there's a checkered racing flag, a person's hands typing on a laptop, and a gear. Below these, a stream of white data points or code characters flows down the left side of the slide. In the top left, there's a small image of an airplane.

**Thank you...**

... for sleeping

That's all folks... but at least this year we didn't complain