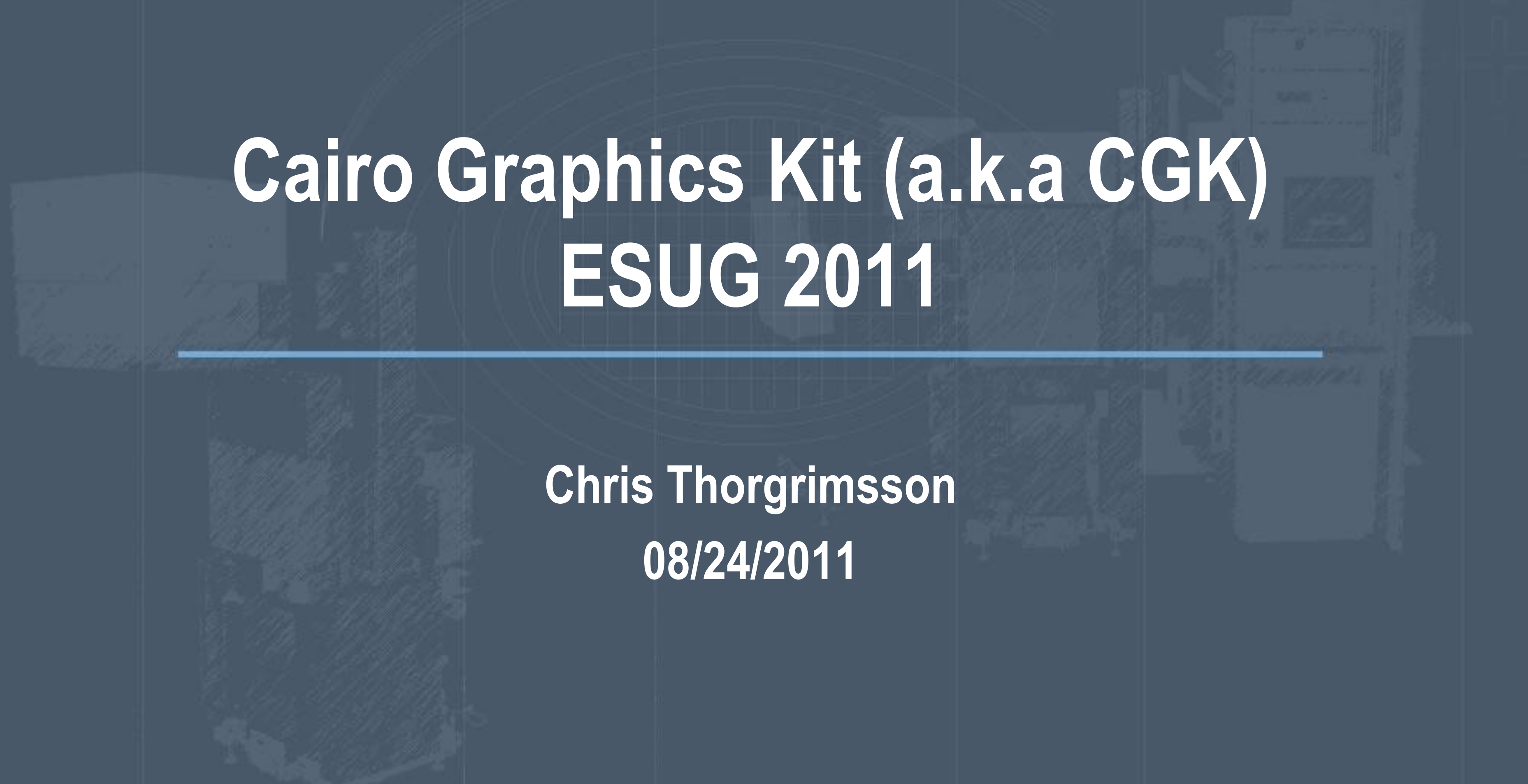


Cairo Graphics Kit (a.k.a CGK)
ESUG 2011

Chris Thorgrimsson

08/24/2011

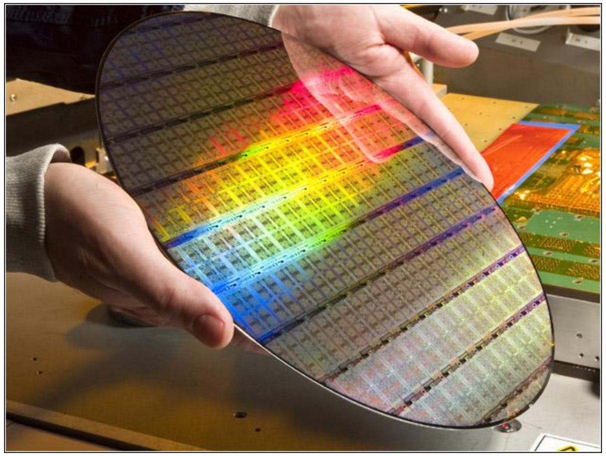
Background

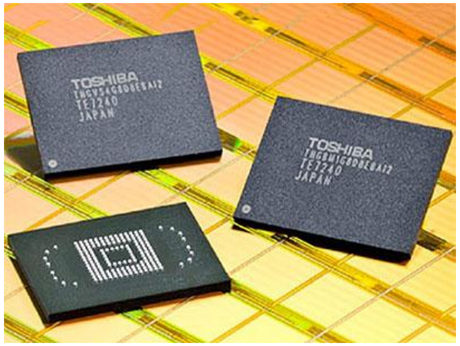
- Who am I ?

Chris Thorgrimsson
Senior Staff Software Engineer
Lam Research Corporation
chris.thorgrimsson@lamrc.com

- What does Lam Research do?
  - Major supplier of wafer fabrication equipment and services to the worldwide semiconductor industry
    - IC components inside cell phones, computers, tablets…etc are manufactured using our equipment
    - VisualWorks Smalltalk is at the heart of our equipment control system

Lam
RESEARCH
2300



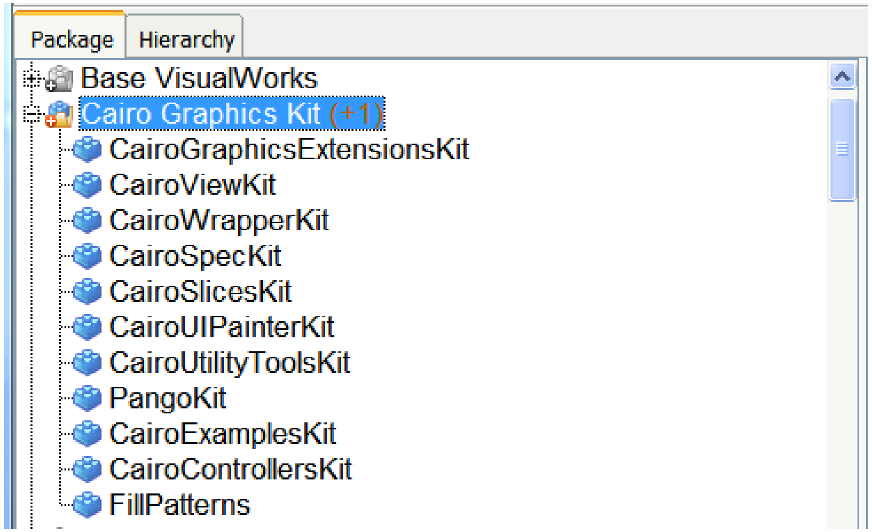

TOSHIBA
JAPAN
TOSHIBA
JAPAN

What is cairo?

- cairo is an open source 2D graphics library with support for multiple output targets.
  - Written in C
  - Usually compiled as an external library (e.g. DLL)
  - Currently supported output targets include
    - X Window System (Linux)
    - Quartz (Mac)
    - Win32 (Windows)
    - image buffers (Targets an in-memory image buffer)
    - PostScript (Generates a PostScript file, suitable for high-quality print output)
    - PDF (Generates a vectorized PDF file, suitable for high-quality print output)
    - SVG file output (Generates a “Scalable Vector Graphics file)
- Cincom created a Smalltalk language binding to the cairo C library
  - Utilizes Cincom’s “DLL and C Connect” technology
  - Published as a package called “CairoGraphics” in the Cincom public STORE

What is the CGK?

- It's a collection of packages that further enhance the *CairoGraphics* and *Pango* packages already provided by Cincom
- It provides cairo based:
  - Views
  - Wrappers
  - Controllers
  - Extensions
  - Utilities
  - Examples



Base VisualWorks
Cairo Graphics Kit (+1)
CairoGraphicsExtensionsKit
CairoViewKit
CairoWrapperKit
CairoSpecKit
CairoSlicesKit
CairoUIPainterKit
CairoUtilityToolsKit
PangoKit
CairoExamplesKit
CairoControllersKit
FillPatterns

- It's a working example of what you can do with VisualWorks, cairo and Pango
- It's a free bundle licensed under LGPL for use in VisualWorks 7.7
  - An MIT license is also in the works
  - With a bit of work and some side effects, it also works in VW 7.4

CGK Origins…

- What started it all?
  - The animation technique used by our older framework couldn’t meet the requirements of some new equipment that was being designed
  - The new system would require an animation to have two functions that VisualWorks couldn’t handle on its own
    - I would need to rotate a semi transparent PNG image in real time
    - I would need decent anti-aliasing after the image was rotated

CGK Origins…

- What other requirements contributed to its evolution?
  - Other developers wanted to use cairo capabilities, but not at the binding level
  - Developers wanted to maintain a familiar VisualWorks experience
    - Use tools like UI Builder / Painter
      - WYSIWYG
      - Incorporate familiar functions from vector/pixel based editing programs
    - MVC
    - GUIs derived from ApplicationModel
  - I wanted to easily share my work with the VisualWorks community
    - Initially it started out as an example repository
    - www.mycairographics.com

Staying with the UI Painter

- It let us break down the process of creating an animation overview into the same process as developing a normal VisualWorks GUI. ○ ● windowSpec
  - It provided the WYSIWYG functions they wanted
- It had “some” of the features found in graphic editing tools I was familiar with so it was the logical place for my own tool creations.
  - New slices were needed to provided support for the various cairo based views ○

Staying with MVC

- In the beginning, it was more like MV than MVC
- CairoPNGImageView, PangoMarkupLabelView, and CairoLabeledVisualView don't have a runtime controller
  - CairoVisualRegion doesn't either, but it's not a really a View and has no model
- CairoActionButtonView and CairoDendrogramView have controllers, but they don't exploit any specific behavior of cairo
- Most work on controllers has revolved around extending / subclassing existing UIPainter controllers
  - I also include new trackers as part of the controller work

Staying with Wrappers (a.k.a Decorators)

- If the widgets were going to be VisualWorks “proper”, then they would need to work with most of the usual VisualWorks wrappers .
  - Some were extended and some were subclassed.
- The wrapping process works in the same as a VisualWorks visual component
  - It’s part of the component spec and is done during UI building
- Cairo based components interact with their wrappers more than their VisualWorks counterparts.
  - SpecWraper
  - TransformWrapper
  - LayoutWrapper

Cairo Based Views (CairoSimpleView)

- Any cairo based view can always get to its spec wrapper
  - The spec hold the visual components DNA and in the CGK, I modify the DNA a lot.
- Any cairo based view can always get to its transform wrapper.
  - The transform wrapper is an integral part of the CGK and gives most Views their ability to scale, rotate, translate etc…
- Any cairo based view or subclass of VisualPart implements a “double dispatch” approach to the *displayOn:* method
  - Cairo based components may be handed an instance of ScreenGraphicsContext or CiaroContext and therefore, need to know how to render themselves on either context
  - Using the visual hierarchy browser is an example of when a component may get either type of context object ● AOTA

The Affine Transform and its role in the CGK

- Its so important, it got its own wrapper
- The transform wrapper is at the heart of the CGK
- It provides a CGK View with a full range of transformation capabilities
  - A view can be rotated, scaled or translated either during runtime or during canvas editing
- Transform wrappers also act as a composite container, allowing it to apply a single transform to multiple components
  - Transform wrappers can also be nested within each other.
- Transform wrappers also understand how to translate Point objects in to or out of their coordinate space no matter how deeply nested. ○
  - globalToLocal:
  - localToGlobal:
- Controllers that interact with transform wrappers don't need to worry about mouse point translation, the wrapper will take care of this form them
- The transform wrapper maintains its own damage repair policy ● CarrierArm

Conclusion

The Good and the Bad about cairo and the CGK

- The Good
  - We've been able the achieve our goal of integrating more advanced graphics capabilities into our software while staying with VisualWorks
  - We can continue to use GUI development tools and methodologies we're familiar with
  - Cairo has a fantastic community behind it. It is constantly evolving and continues to offer cutting edge features
- The Bad
  - It's a binding to the outside world. External DLL bindings make our developers nervous
    - Stay within the VM is usually our motto
    - The first sign of trouble with the VM usually involves me defending cairo
  - Most Smalltalk developers aren't graphics guys. Looking pretty usually isn't a high priority in Smalltalk
    - I respectfully disagree!
  - Using pre-built cairo binaries can tricky.
    - I ended up building my own for Win32.

What's next for the CGK?

- Views
  - VectorShape
  - CalloutBox
  - Gauge
  - Slider
  - Progress Bars
  - Radio Buttons
  - Graphing
    - Dendrogram ● Dendrogram
    - Line, Bar, Pie etc…
- UI Painter Slices
  - Replacing the Color and Pattern Slice with a new Fill/Stroke slice
    - Color, Pattern, or Gradient selection in one unified slice
  - Whatever would be specific to the above Views
- AppModelView
  - My answer to the difficulties of creating a VisualWorks "proper" widget.

What would I like to see from Cincom

- Just one thing
  - The UI Painter and its tools are starting to show their age! It’s time for an overhaul.
    - Simple…right?
    - Talk to me later, I have some ideas…..for a small fee of course ;-)
- Ok, maybe one more thing
  - Can you fix this kind of stuff?

ScrollerButtonView

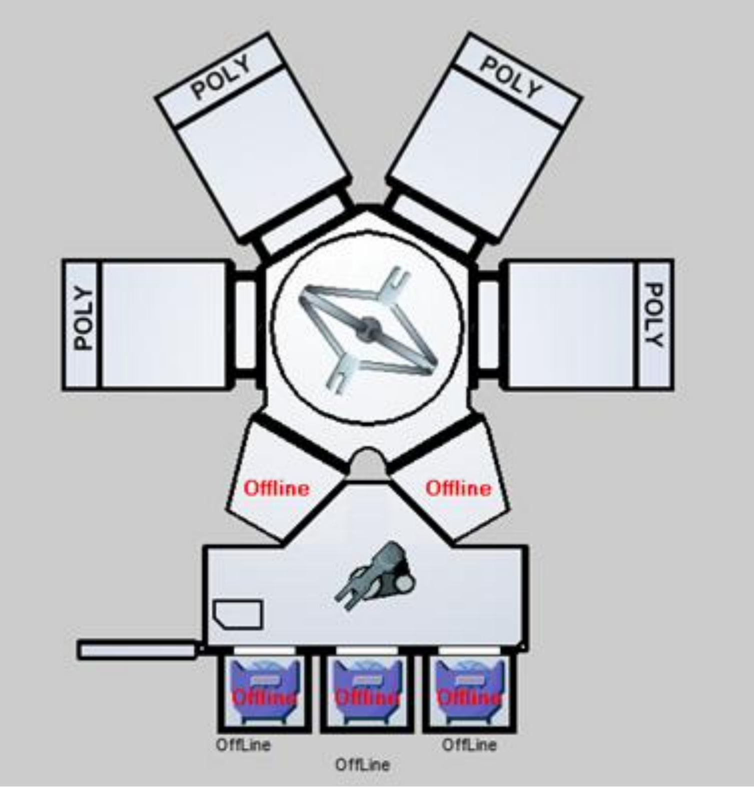
```
outerContainer
^self container notNil
   ifTrue: [self container container notNil
      ifTrue: [self container container container notNil
        ifTrue: [self container container container container notNil
          ifTrue: [self container container container container container]]]]
```

FAST to Customer Solutions™

Lam RESEARCH

Supplemental Slides

Animation The Old Way



- Images are added or removed from a CompositePart at runtime
  - All possible images reside in a Dictionary that's maintained by the application model
  - Events drive the animation
  - The sequence below would animate an arm extension, retraction and rotation event sequence

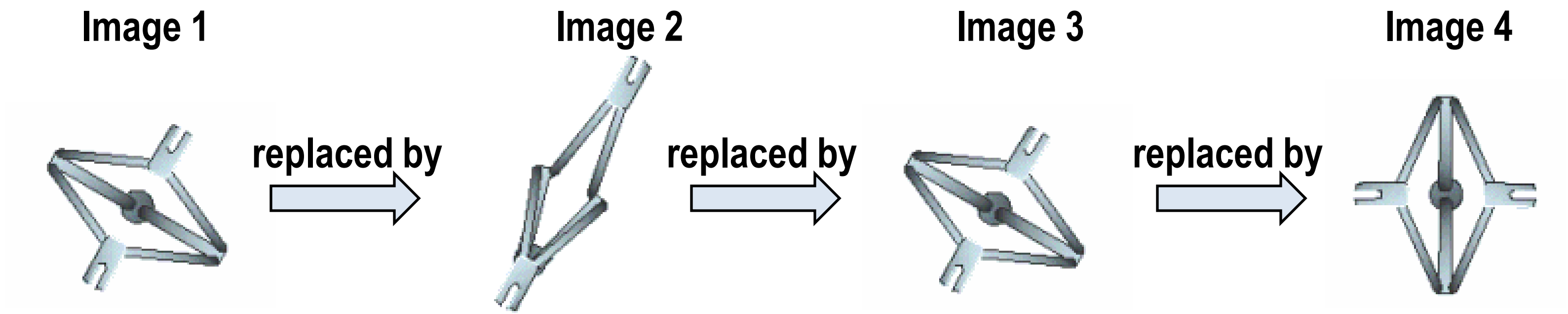
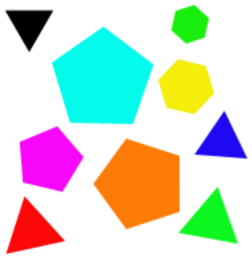


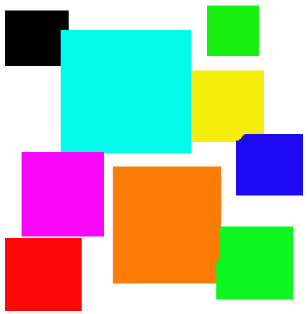
Image 1 replaced by Image 2 replaced by Image 3 replaced by Image 4

Dealing with PNG Files…VisualWorks versus cairo

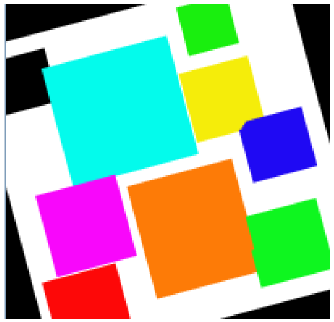
Test PNG Image



VisualWorks (7.4)

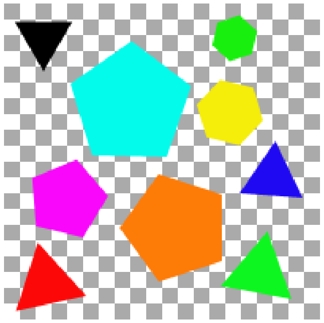


The result after the PNGImageReader reads the file.

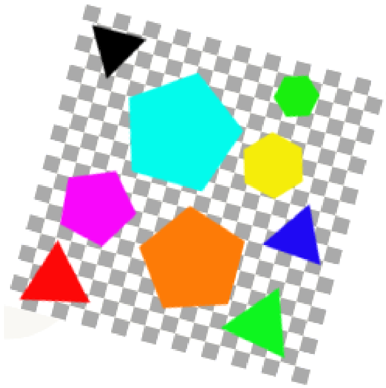


The result after a 15 degree rotation.

cairo



The result after using cairo to render the image



Using a cairo affine transform to rotate the image