
ORACLE®

JavaOne™

ORACLE®

Introduction To JavaFX Scenic View

Jonathan Giles
Consulting Member of Technical Staff
Java Client Group
September, 2016

Java Your (Next)

JavaOne ORACLE

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

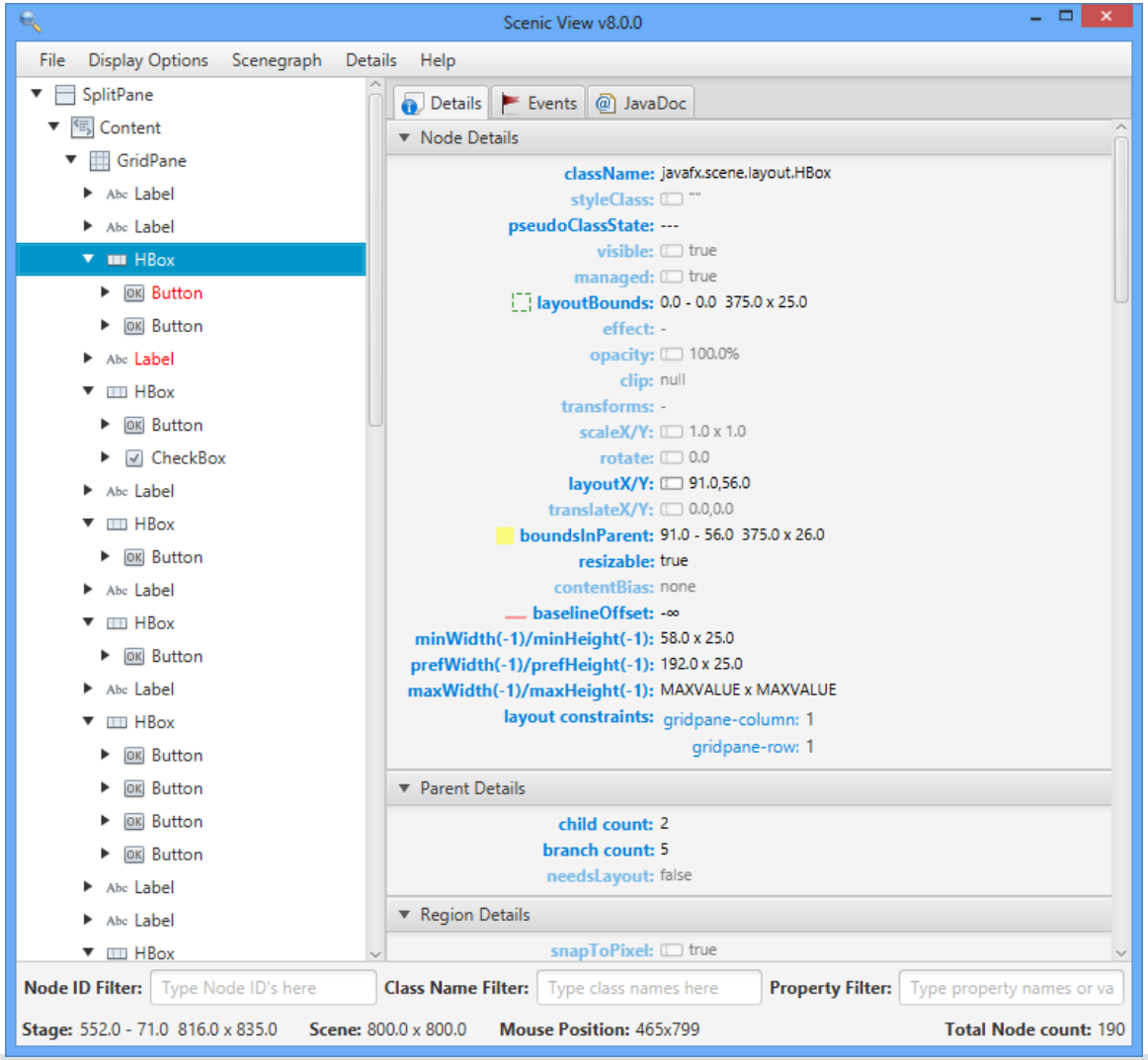
Warning: This is almost last years BOF.

If you attended last year, you may not want to stick around…

Note: I've condensed the BOF this year.

We should be out of here in 20 minutes…

Scenic View in a Nutshell



Scenic View in a Nutshell

- Scenic View is a free JavaFX scenegraph analyser.

- I develop Scenic View,
  - when time permits (or when a conference is coming up!)
  - it is not my job (but it is very useful in my day job)!

- Download and find out more about Scenic View here: http://www.scenic-view.org

Scenic View

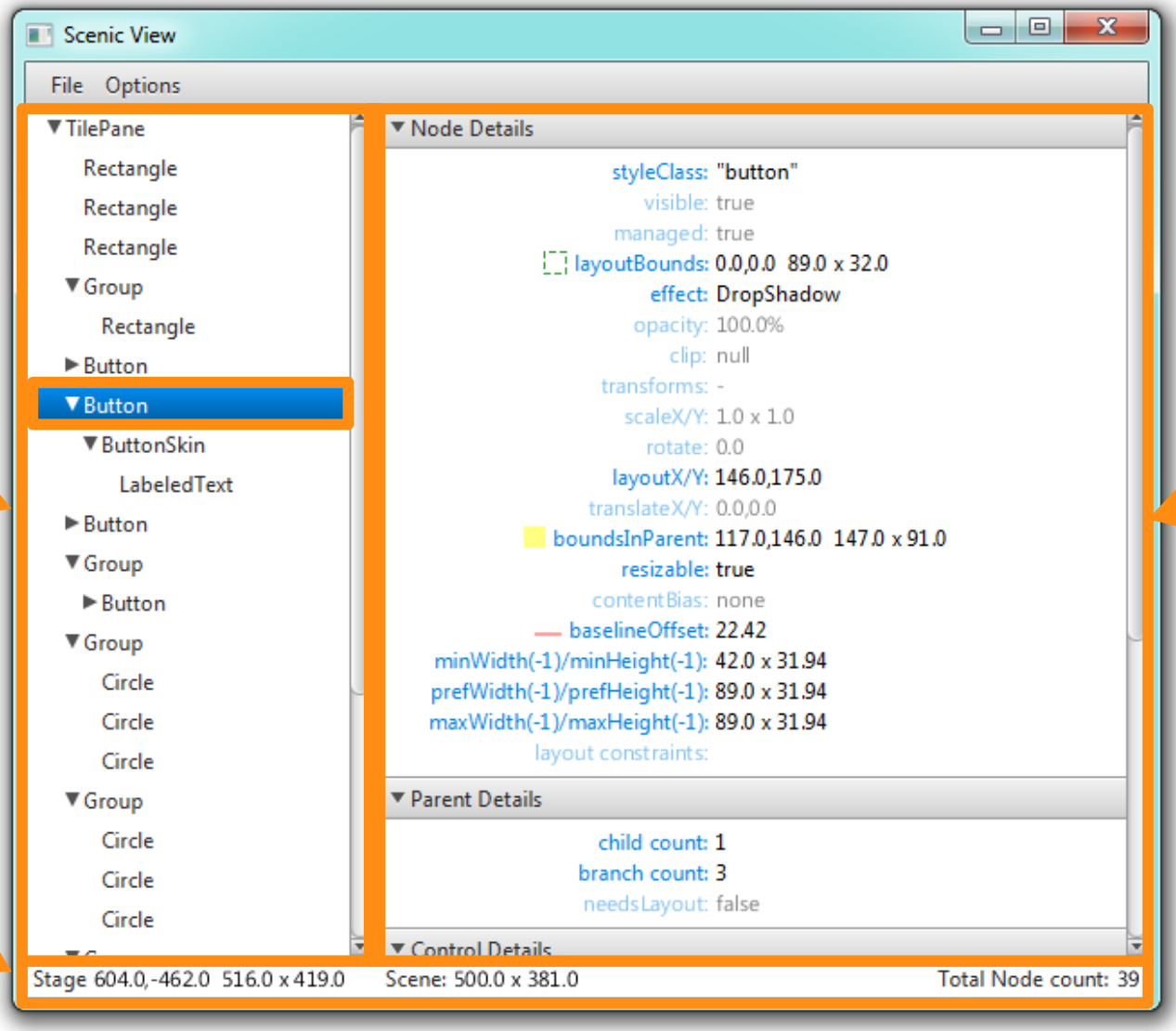
- What is Scenic View?
  - Originally built by Amy Fowler for diagnosing runtime issues with UI layout
  - It was really simple to use:
    just add `ScenicView.show(scene)` in your code

- I took Amy's code and polished the UI considerably before the first public release of Scenic View on May 6, 2012
- It looked a little like this:

Scenic View Pre-1.0.0

Tree showing scenegraph structure of running application

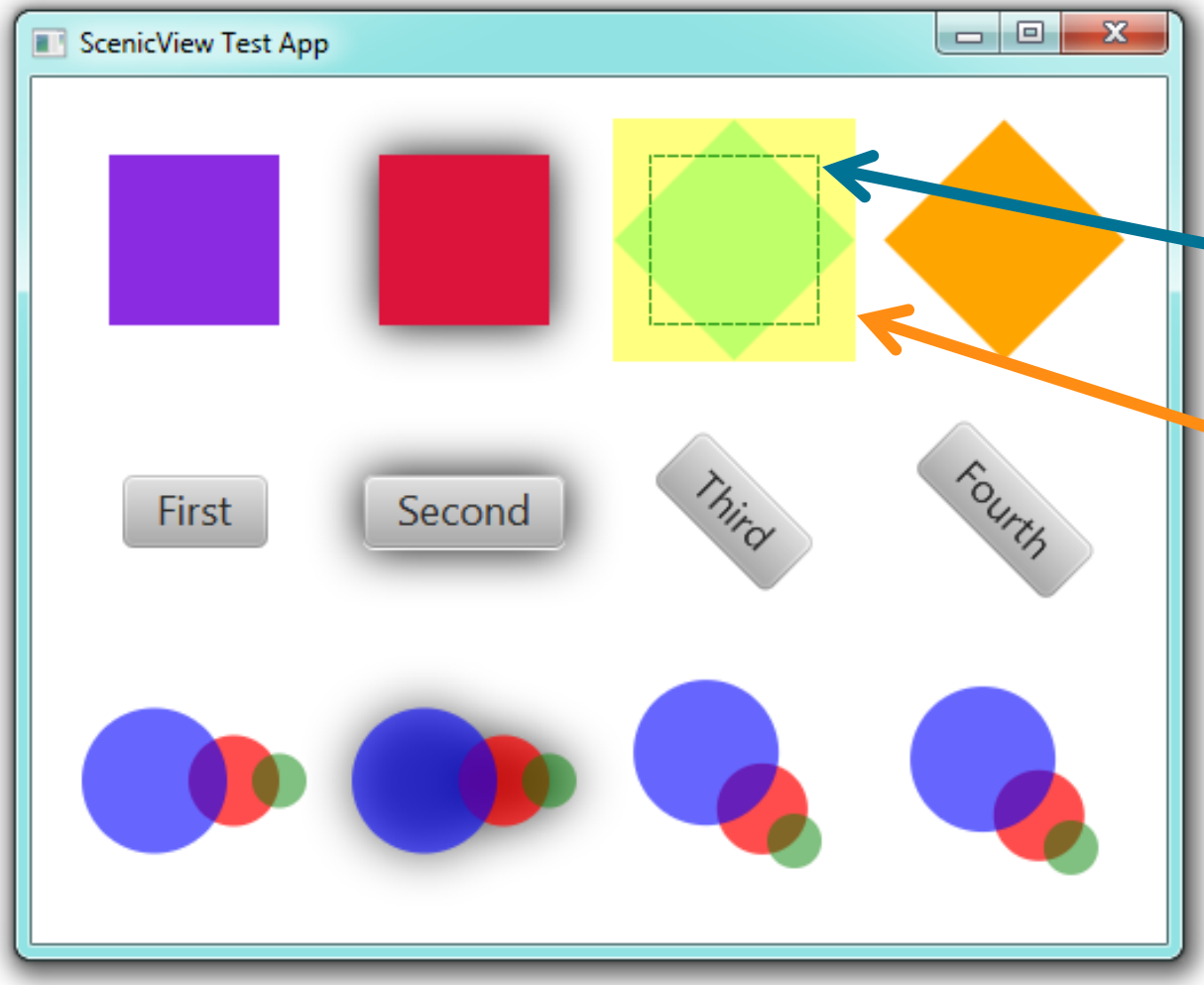
Application overview



The most important properties for the selected node

Who knows the difference between
‘layout bounds’ and ‘bounds in parent’?

Scenic View



Scenic View can also draw overlays in the application it is observing.

The green dashed rectangle shows the layoutBounds,

and the yellow filled rectangle shows the boundsInParent.

This can be very useful for debugging.

Scenic View

- The response to the public release of Scenic View was extremely positive.
- Most feedback was of the form: “I love it, but it needs to do X”
- X included:
  - Live editing ✓
  - Filtering ✓
  - Selecting nodes by clicking in the UI ✓
  - Event tracing ✓

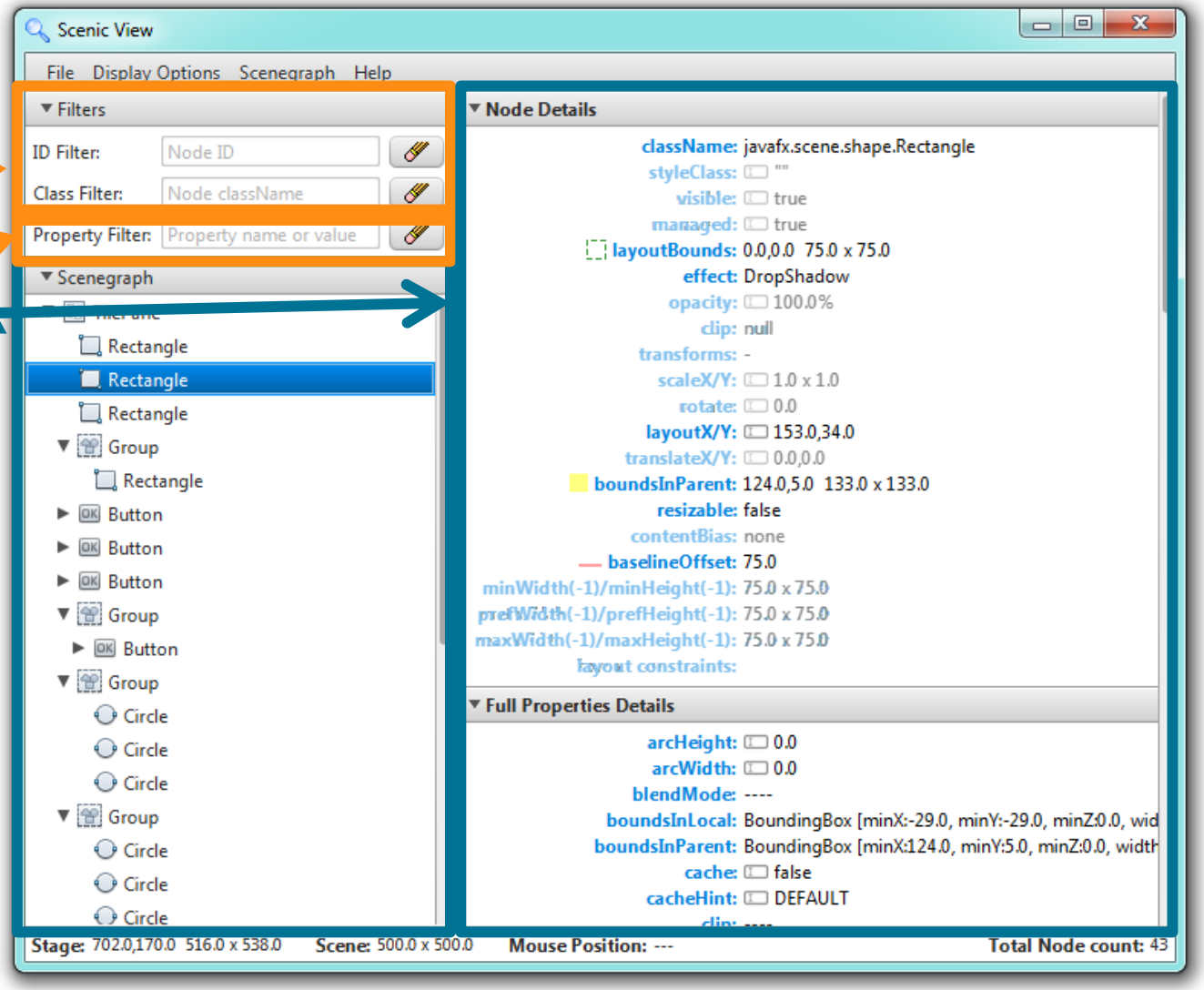
Scenic View 1.0.0

- Ander Ruiz contacted me after the first release, and had a number of ideas.
- Even better, he was keen to help program them!
- Together we made available Scenic View 1.0.0 on June 4, 2012
- The first versioned release of Scenic View
- It looked a little like this:

Scenic View 1.0.0

Ability to filter the tree area by typing in an ID or class name

Ability to filter the details area by typing in a property name



The Problem with Scenic View 1.0.0

- The major complaint Ander and I heard from users:
  - People did not want to have to modify their code by adding `ScenicView.show(scene)`.

- This proved an interesting (and complex) problem to resolve!
- We needed a way to connect to applications at runtime without any modification of their code.

- We settled on two solutions

Solution 1: Java Agents

- The Java agent API allows for an external library to be called when an application starts.
- Simply add the following when starting your application:
  `-javaagent:ScenicView.jar`
- Scenic View will start when your application starts
- It will automatically discover all stages in your application

- Best approach: in your IDE have two ‘run’ profiles, one with Scenic View enabled and the other without

Solution 2: Java Attach API

- Java provides the Attach API to discover running Java applications
- We use this to install a small socket server into your application at runtime, through which Scenic View can communicate

- This means that Scenic View can discover all running JavaFX applications and you don’t need to do anything!
- To use this solution, simply start Scenic View directly and it’ll start in this mode

Scenic View 1.1.0

- Scenic View 1.1.0 was released on August 14th, 2012 after much testing and user feedback
- This release required a massive amount of reworking and foundation building.

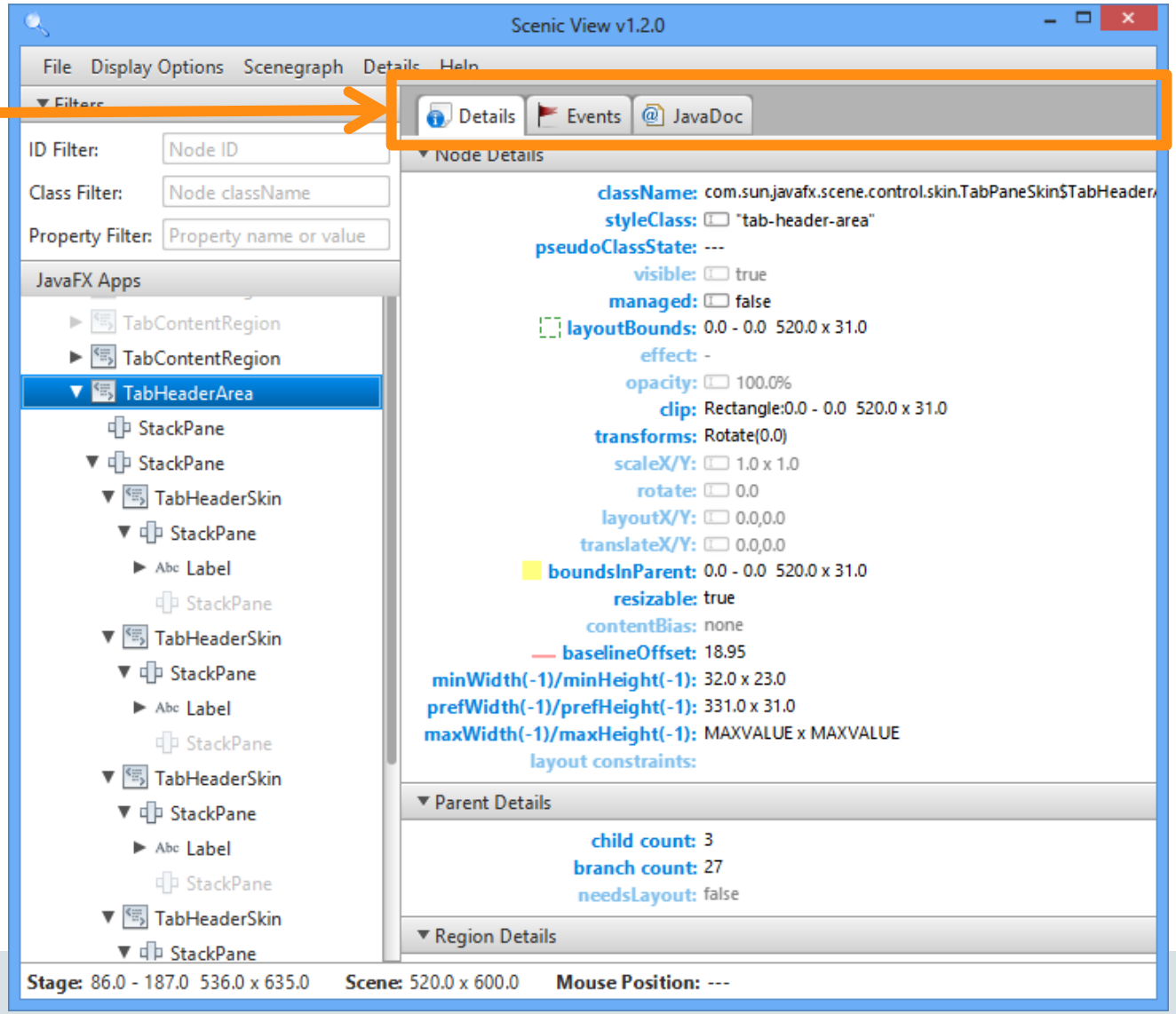
- We were incredibly relieved to get this working on Windows, Mac OS and Linux!
- However, things are never perfect, and Scenic View 1.1.1 was released on August 16th, 2012
- This improved our ability to debug peoples issues.

Scenic View 1.2.0

- Still, we knew there was more to do, so we carried on and released Scenic View 1.2.0 on September 25th, 2012.
- This release included:
  - Event tracing support
  - JavaDoc browsing support
  - Streamlined menus (context menus)
  - Bug fixes!

Scenic View 1.2.0

Tabbed area for new functionality



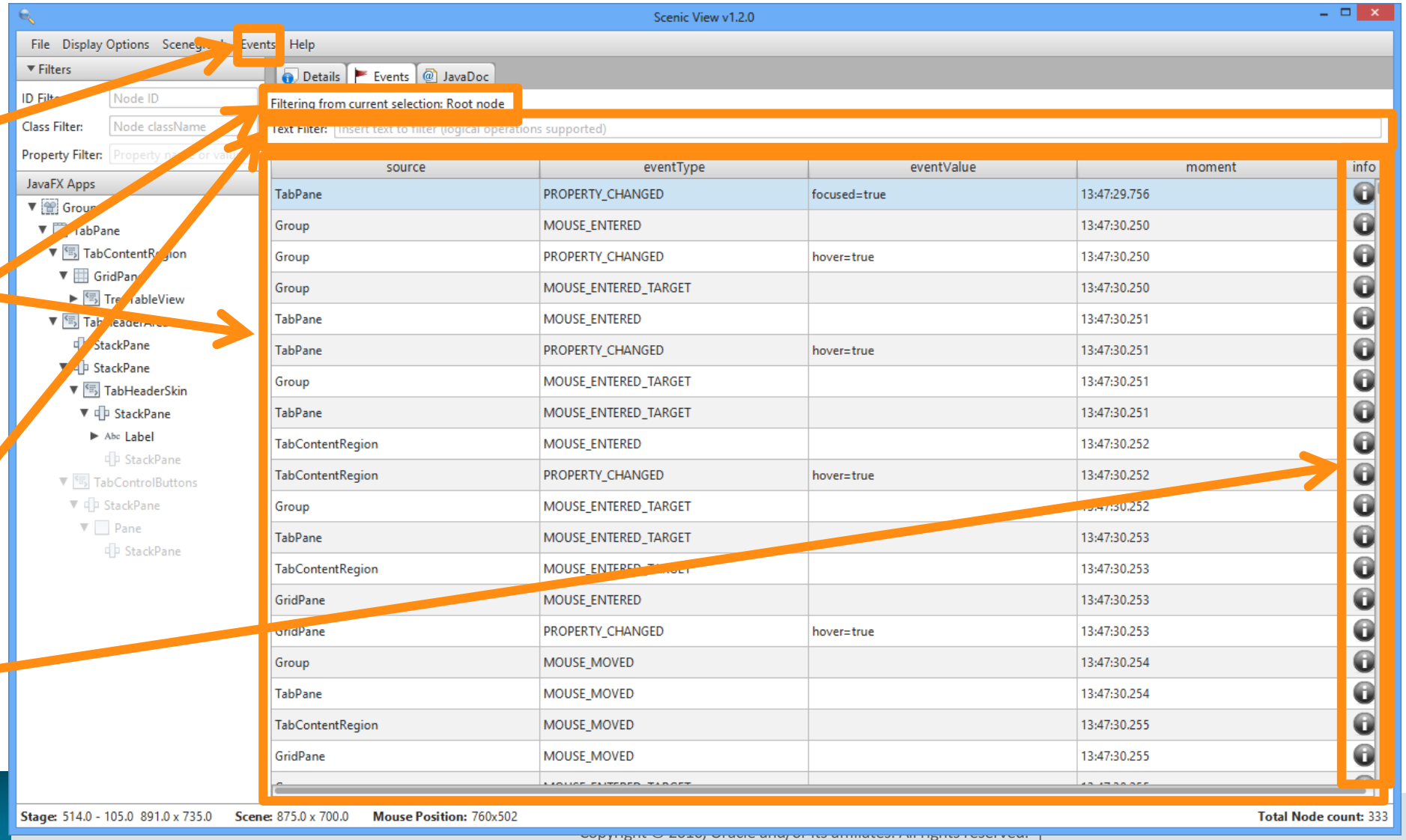
Scenic View 1.2.0: Event Tracing

Event tracing is enabled from the Events menu

Tabs show all recording events from selected node down

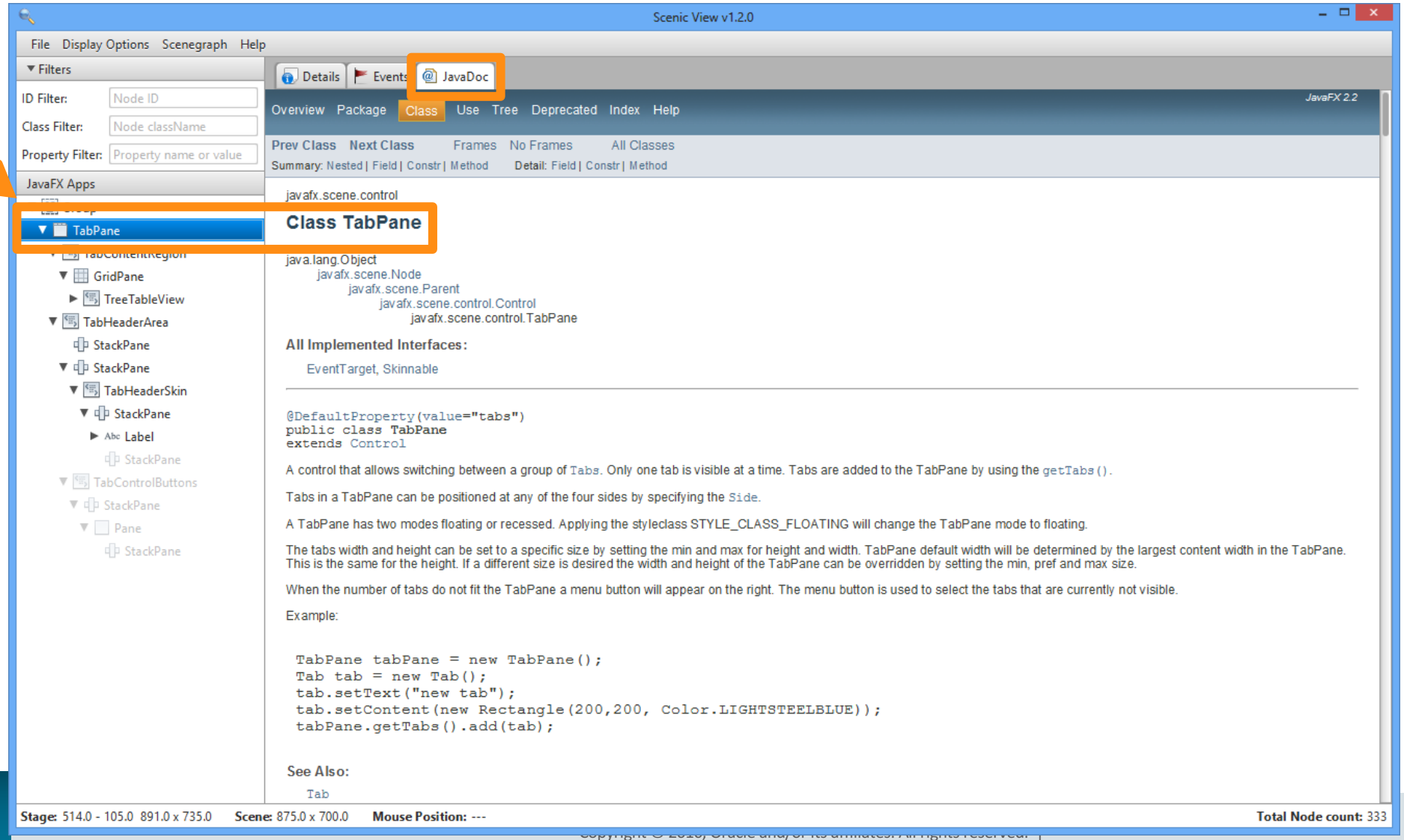
Events can be searched using boolean statements

Click the info button for the entire stacktrace



Scenic View 1.2.0: JavaDoc Browser

Browser shows JavaDoc for currently selected node



Scenic View 1.3.0

- Scenic View 1.3.0 was released on November 12, 2012
- What was new?
  - Massive performance gains
  - Animation tracer
  - Improved CSS support
  - Version update checking
  - Mac native menubar integration
  - Bug fixes and miscellaneous improvements

Beyond Scenic View 1.3.0

- Scenic View 1.3.0 was the last release with support for JavaFX 2.x.
- Unfortunately at this stage Ander had to drop out due to work commitments

Scenic View 8.0.0

- Released and open sourced at JavaOne 2014

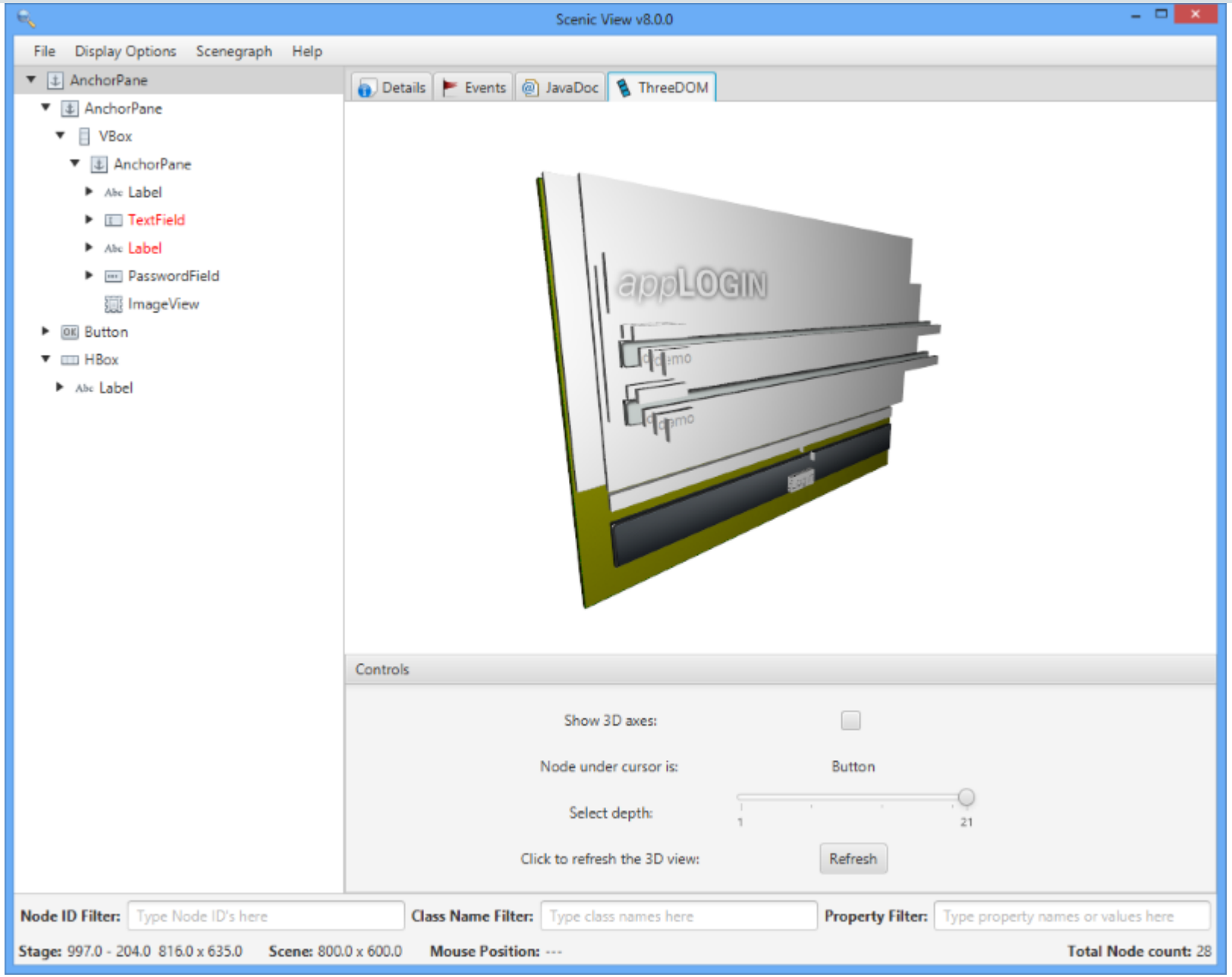
- GPL licensed
- Code repository is open here: https://bitbucket.org/scenicview/scenic-view

Scenic View 8.6.0

- Released September 2nd, 2015
- Primarily consists of two very cool community contributions :
  - ThreeDOM
  - CSSFX

ThreeDOM

3D ‘explosion’ view of user interface



CSSFX

- Ability to edit / save CSS files and have them be dynamically reloaded at runtime without needing to restart application.

Scenic View 9.0.0

- Released August 30th, 2016
- At present this is simply a branch of the main branch with minimal set of patches to work in JDK 9

Scenic View Demo

Getting Started
- In Six Simple Steps!

1. The following software is required:
   - Mercurial
   - Gradle
   - JDK 8

2. You'll need a Bitbucket account
   - Accounts are free from http://bitbucket.org

Getting Started
- In Six Simple Steps!

3. Fork the repo.

   Go here to create your own fork

   – https://bitbucket.org/scenicview/scenic-view/fork

4. Clone your fork:

   – hg clone https://<username>@bitbucket.org/<username>/<forkname>

   – e.g.

   • hg clone https://jonathangiles@bitbucket.org/jonathangiles/scenic-view

Getting Started
- In Six Simple Steps!

Build your clone. From clone root directory, run:

- gradle clean assemble

Run your clone:

- gradle run

The Future of Scenic View

- What else is there left to do?
- Should I pack my bags and stop developing Scenic View now?
- Is there a feature you'd love to have?
  - Tell me!
  - Email me and let me know at jonathan.giles@oracle.com

- Some ideas:
  - Pulse logger support (pulse duration, time since last pulse, etc)
  - Less bugs, faster, better UI, etc

The Future of Scenic View

- Even better – please join in and help me to develop it!
- Fork the project on bitbucket and do pull requests

Thanks for Attending!

It’s Discussion Time!

How to contact me:
jonathan@jonathangiles.net
@JonathanGiles

JavaOne™

ORACLE®


ORACLE®