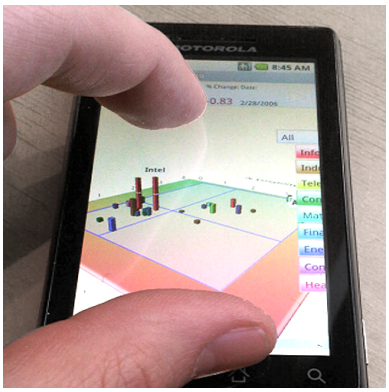
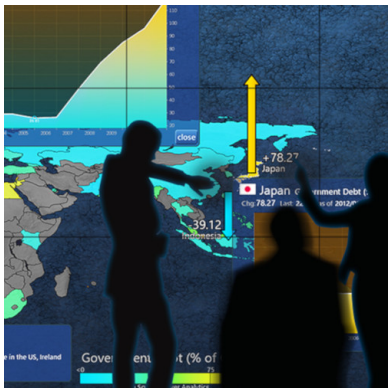




04J®

FOR CROSS-PLATFORM DATA VISUALIZATION SOLUTIONS USING JAVA

TECHNICAL WHITEPAPER

uncharted

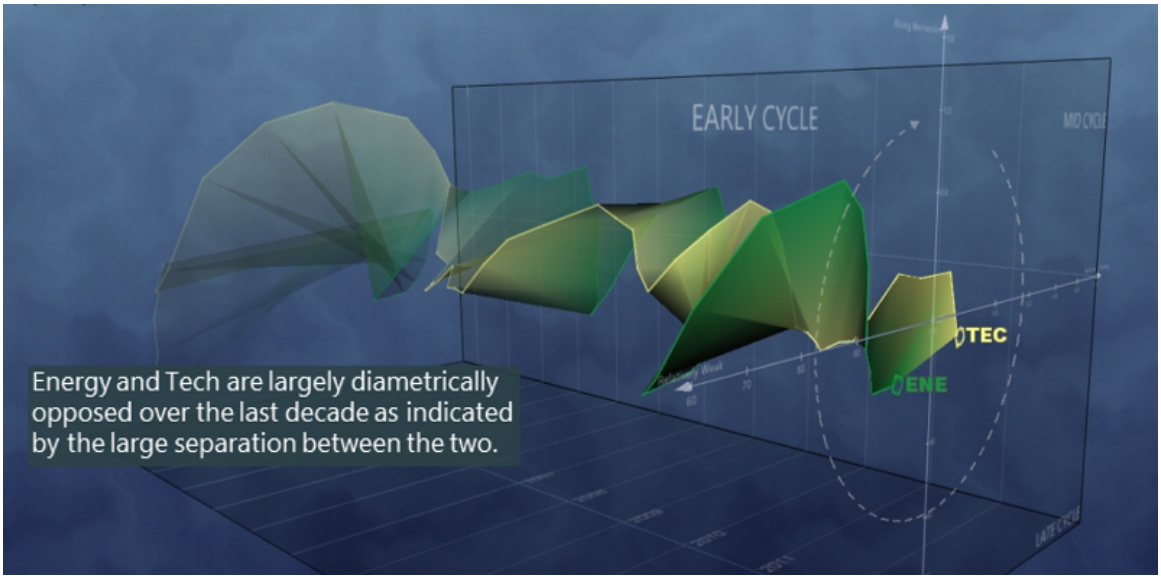


Figure 1: Large scale visualizations use O4J to provide the big picture overview and all the details on tap for applications ranging from command centers to strategic planning to interactive *data story* walls.

Introduction

Uncharted® (www.uncharted.software) is the leading provider of innovative business visualization software and professional services for Fortune 500 companies and third-party software providers. Our innovative, multi-dimensional, interactive user interfaces enable our client base to achieve far more comprehension and rapid understanding of their complex data.

The principals at Uncharted have a long and distinguished history of designing and developing high value, business visualization solutions. In addition to providing consulting services to our clients, Uncharted has persistently sought out new opportunities to develop our own software toolkits to help facilitate custom business data visualization solutions where existing solutions are simply not enough.

O4J is a proven software toolkit for business data visualization. With O4J, fully interactive 3D visualizations can be delivered to Windows, Mac, Linux, Android and Blackberry devices. O4J is in use today by thousands of users in production visualizations ranging from tiny highly interactive visualizations for mobile devices to rich visualizations of complex data for viewing across many screens or massive video walls.

O4J provides a wealth of features and visualization techniques for making effective information visualizations as outlined in this paper.

Figure 2: Financial firms use O4J for real-time market data. Mobile visualization provides the quick "at-a-glance" comprehension needed when on the move.

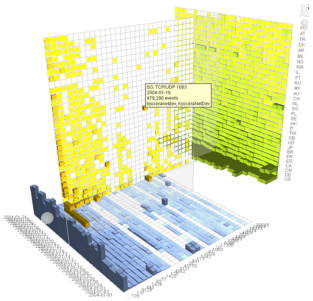


Figure 3: O4J is used for Network Security: large amounts of intrusion events can be visually depicted and explored providing insight into anomalies and causes around security incidents.

O4J

The O4J software toolkit (SDK) is a cross-platform 2D and 3D visualization platform offering deployment options from smart phones, to lightweight web applets, to desktop applications, to fully immersive environments and site installations. Unlike other libraries, O4J scales all the way down to mobile devices or web pages without hardware graphics support, or all the way up to a distributed cluster-rendered immersive environment, all with a single API.

The O4J architecture is adaptive and opportunistic, taking advantage of capabilities of the platform discovered at run-time. Run time graphics acceleration is available on Windows, Mac OS X, Linux, Android and select Blackberry devices. When hardware assisted graphics acceleration is unavailable, O4J uses its optimized pure Java software renderer, eliminating the glitches that can occur when encountering poor graphics drivers in a deployed environment.

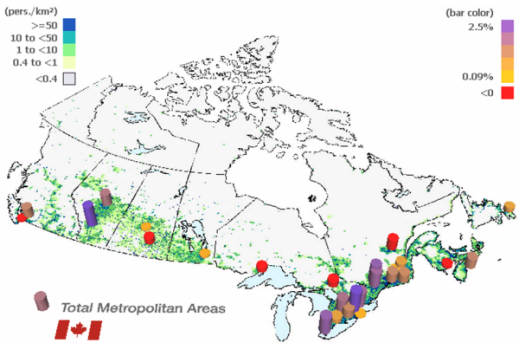


Figure 4: Simple visualization integrating a population density map with major urban centers. Animation, layered visuals and tooltips enable easy exploration of many variables at once.

Technical capabilities of the O4J library include:

- A real-time 3D rendering engine.
- Cross-platform portability across mobile and desktop devices, including support for touch and mouse interactions.
- Either pure Java, software-only rendering or hardware acceleration.
- A server-side deployment option leveraging pure Java for a robust 3D solution on standard servers.
- A large library with hundreds of Java classes, designed for creative applications, extensibility, power and ease of use, managing graphic and device complexities automatically.
- Advanced features such as high-performance scene and streaming scene change serialization, dynamic label deconfliction, scalable native and foreign language text rendering, and automatic animation tweening.

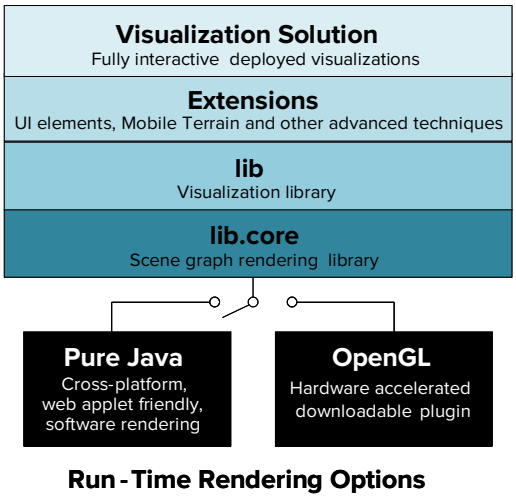


Figure 5: O4J architecture.

Benefits

O4J provides a simple flexible interface for creating rich interactive visualization solutions. The library allows the developer to focus on the business solution without getting bogged down in the details of low level device and graphics manipulation. This ease of use is combined with the flexibility to develop truly customized solutions which would not be available from ready-made charts.

Flexibility: O4J provides a simple consistent interface that can be used to create a wide range of visualizations. For example, a simple 3D bar chart can be made with four instantiations:

- A VisualSet2D of Cubes
- A VisualSet2D of Lines
- Two VisualSets of Text

With only these three classes and the associated methods, complete flexibility is provided - flexibility that cannot be achieved with a monolithic component class with hundreds of methods.

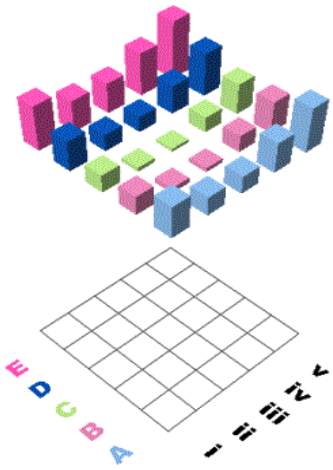


Figure 6: With O4J, a bar chart can be created with four instantiations of three classes. The same instances can also be used to make a scatterplot.

Cross-Platform and Ease of Development: Low-level graphics alternatives, such as DirectX, OpenGL or WebGL are hardware specific, making it difficult to use for broad deployments or cross-platform deployments. Further, the use of low-level graphics requires additional specialized programming effort, such as rendering optimization, scene management and viewer navigation.

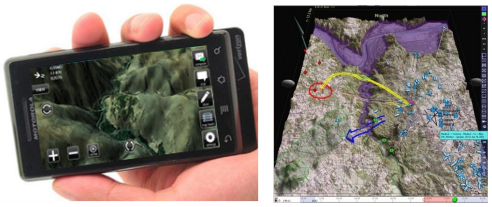


Figure 7: This 3D logistics visualization has been designed to work for different users and tasks ranging from large video walls (e.g. status), desktop (e.g. analysis) and mobile devices (e.g. updates).

2D graphics libraries, such as those based on Flash, SVG or HTML5, do not provide true 3D capabilities; and typically are not scalable to data-driven interfaces requiring the management of thousands of data elements. Other plug-in alternatives may offer 3D, but typically do not offer cross-platform support nor optimizations for mobile devices, again making it difficult to use for broad deployment.

O4J is an effective solution for 3D interactive visualization in desktop, mobile and server environments providing programming flexibility, deployment flexibility and cross-platform portability.

Easy Data Management: With O4J, data is kept entirely separate from the visual elements. Using O4J means data is not repeated multiple times throughout the application, nor does the data need to be specially transformed to fit the specific format required by the visualization. Data can be managed centrally and it is easy to attach and update multiple visualization elements to the data. Furthermore, O4J does not prescribe any pre-set data model or data format, giving the developer flexibility to use the most appropriate data model for the data. Visualizations built using O4J can leverage the latest approaches to managing and interacting with data from traditional relational databases, to tight integration with real-time data feeds and complex event processing, to advanced big data systems such as Hadoop and MapReduce.

Easy Interactions: O4J makes it easy to extend and enhance visualizations with rich interactions, through events including mouse and touch events to enable features such as tooltips, pinch-zoom and drill-down.

Easy Animations: O4J simplifies the creation of smooth transitions and complex animations within visualizations. Simple transitions are created by defining an end state and letting the library automatically animate the changes. More complex animations can be defined using keyframes to describe the changes.

Easy Graphics Management: O4J facilitates the management of detailed visualizations and the creation of re-usable visual components with techniques such as Layouts, VisualSets, Grouping and object oriented programming techniques which promotes ease-of-programmability, performance and scalability.

Features

O4J contains a multitude of data-visualization-specific classes for creating, positioning and manipulating graphical objects (or groups of objects) in a 3D scene. But it is not limited to 3D. With O4J, developers can also create pure 2D or mixed 2D/3D data visualizations.

One of the primary advantages of using the library is that no low-level graphics programming theory is required. Object creation, visual techniques, interaction handling, scene rendering and viewer navigation are all managed by the library, providing the ability for developers to create interactive visualization applications with significantly greater ease and speed.

A key advantage of O4J is that a single API can be used to develop for desktop, web and mobile platforms using a single core API, with only two simple class selections required to integrate the application into different environments.

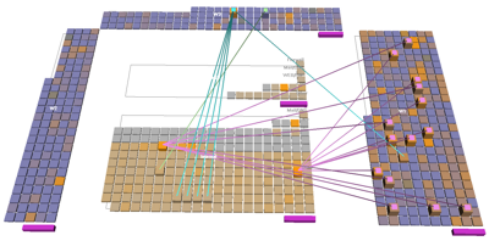


Figure 8: Node and link application, illustrating the relationships between thousands of individual items.

Data-source Neutral

O4J is not constrained to a single data source. Everything from text and XML files, to Microsoft Excel spreadsheets and any database can be accessed via standard Java data connection methods. This open data architecture ensures connectivity to critical legacy and pre-calculated data as well as real-time data feeds and allows for tight integration directly into pre-existing Java applications.

Viewer

At the center of the O4J API is the Viewer class, which is the root client description of all aspects of the scene to be rendered. The viewer treats devices and interactions abstractly, providing a single consistent interface for drawing and interacting with the scene.

The Viewer manages high performance rendering processes that efficiently update only those parts of the scene that need updating and redraw the scene only when needed. Multi-threading is used to take advantage of multi-core processors. Within the viewer:

- **Vizlet**: is responsible for the abstracting the device specific application runtime context, including life cycle, resource, and license management as well as access to the device event queue.
- **Canvas**: abstracts the device specific surface (window, image or texture) being drawn to.
- **Viewer**: is responsible for coordinating user updates to the scene and viewpoints with rendering to a canvas.
- **Renderer**: is responsible for drawing the scene onto the canvas. Renderers include Uncharted's Pure Java Software renderer, as well as OpenGL and OpenGL ES renderers for both Android and Blackberry. Renderers can be specified by the client or selected automatically based on an evaluation of the device context at runtime.

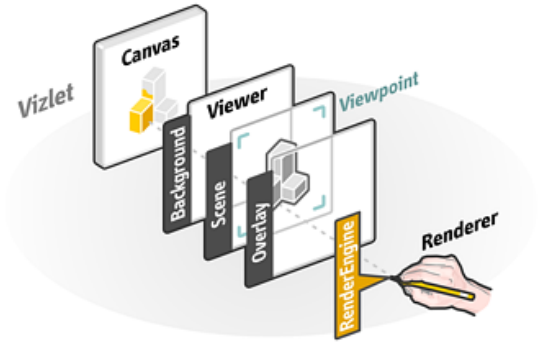


Figure 9: Viewer classes abstract devices, interaction and rendering technologies so that the developer only needs to manage the scene.

Viewer Interaction

The viewer has built-in navigation controls that permit simplified and intuitive interaction with the scene, including panning, zooming and rotating. By clicking and dragging on the scene, for example, the user can manipulate the viewpoint to highlight and view any position for visual drilldown and analysis. Viewer navigation can also be controlled via Viewpoint objects, allowing the developer to programmatically set up their own custom viewpoints.

Shapes

O4J provides different shape types that can be used as a template for single or multiple visual elements. These templates include:

- Cube
- Line
- Sphere
- Cone
- Radial Arcs
- Text
- Quad
- Polygon
- Cylinder
- Point
- Arrow
- Screen Sprite
- Scene Text (for in-scene 3D text)
- Screen Text (for 2D overlay text)
- Face (for surfaces)
- Glyph (for arbitrary shapes)

Conversion from Java2D Shapes is also supported.

Visuals

Visuals are abstract representations of graphic objects in the class library. Since any Shape can be applied to a visual, changing its appearance simply involves applying a different Shape to the instance. To accommodate multiple instances of a graphical object, the library has **VisualSet** and **VisualSet2D** classes that simplify and optimize the creation of multiple instances of a similar shape.

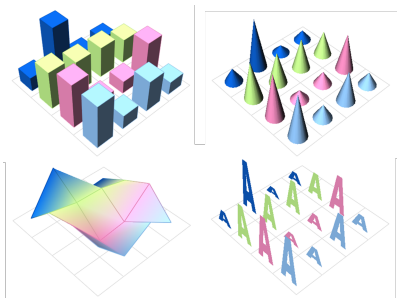


Figure 10: A VisualSet2D (4 x4 elements) set to various shapes, each set to the same height and color values.

Visual layouts

A Visual represents one or more instances of a similar piece of geometry, arranged according to a user-specified layout. There are two layout manager classes in the class library to help manage regularly structured placement of graphical objects:

- **Row layout** - for incrementally placing along a single dimension e.g. one row of bars.
- **Grid layout** - for row and column placement, e.g. as in a table or bar chart.

A visual object can also be placed arbitrarily anywhere in 3D space if managed layout is not required, e.g. in a network diagram or on a map.

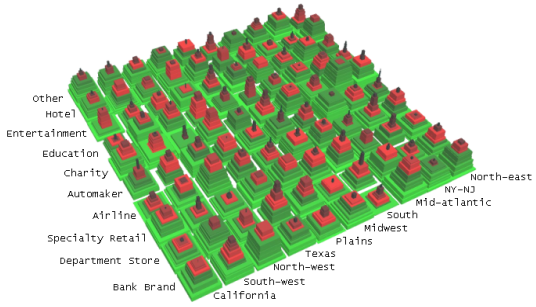


Figure 11. A grid based layout is used in this novel credit risk visualization showing credit quality in stepped pyramids.

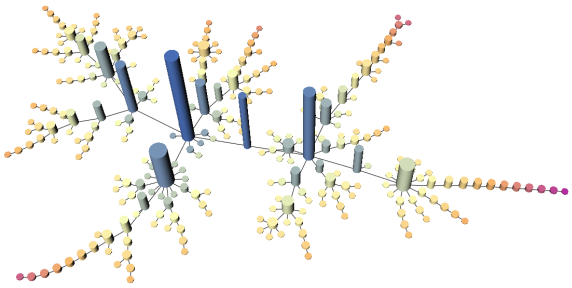


Figure 12. A spring-based layout is used for this visualization of a social network.

Groups

Multiple graphic objects can be placed in a **Group** container. This is a non-visible scene component that can contain multiple other scene components. This facilitates object oriented visual design, by enabling all the sub-components of a complex object to be managed as a single entity. For example, the developer can create a 'ColorBarGroup' object that contains all the visual elements and interactions that are necessary to display a bar graph (title, axis labels, bar height, bar color, grid lines, border, etc.). All items contained within a group are translated, rotated and scaled according to the containing group.

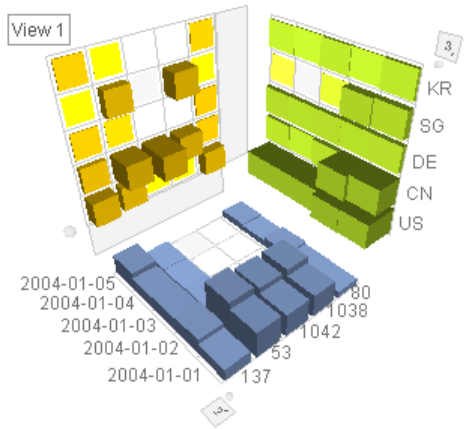


Figure 13: All the elements on each plane have been grouped together into a common class; helping with both clarity of the visualization and simplified management of the 3D scene.

Text, Labels and Annotations

While many 3D environments (e.g. games) do not require high quality text, it is critically important in visualizations for labeling and annotations. Text support includes access to all system typefaces (at all sizes and styles) and also has full Unicode support for extended characters and bi-directional foreign text, such as Hebrew or Arabic. True Type fonts may be embedded and deployed with an application, removing the necessity for the font to be pre-installed on a system.

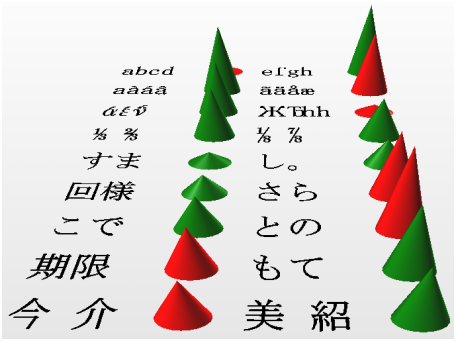


Figure 14: Supports Unicode and bi-directional text.

Text placement includes various layout options:

- **In-Scene** placement: e.g. oriented in the 3D scene so that there is an unambiguous alignment between the text and associated shapes.
- **In-Screen** placement: e.g. displayed head-on but anchored in the 3D scene, effective for annotations.
- **Overlay** placement: e.g. displayed head-on in a consistent 2D locations, effective for legends and filters. Use the Underlay for large titles and watermarks.

Figure 15: Supports text aligned to 3D scene or front facing.

Classes are provided for adaptive label placement in complex scenes, such that labels maintain readability, are clearly associated with its corresponding shape, and do not overlap.

Lighting and Materials

In 3D scenes, it is important to get the right lighting and look. O4J provides a quick high-level interface for setting up lights and materials.

The interface provides directional lights at various orientations and intensities and materials, which are properties of surfaces that can make them appear to be solid, metallic, translucent or transparent. It also provides high-level control for features such as fog/fade; clipping planes, bold silhouettes, glows and simulated drop shadows and reflections.

Figure 16: Bold silhouettes around the outside of the boxes, but not between them, visually pop them out from the background.

Textures

Textures provide the ability to use images in the scene; for example, as underlying maps, floor-plans or diagrams. The library supports textures in all java supported image formats including JPEG, GIF, PNG and more. Textures can be stored and additionally manipulated in memory, for example, image processed to lighten or darken or combine multiple images together. A viewer may be configured to render to a texture displayed in another viewer with one call, in order to paint inks on a terrain, for instance.

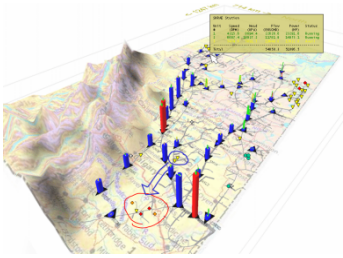


Figure 17: Maps and terrain provide context in this pipeline network. End-user mark-up has been converted into another texture, overlaid on the first.

While textures cannot be applied to lines, the O4J lines support a range of features, including wide lines, parallel lines, multicolor lines and dash patterns.

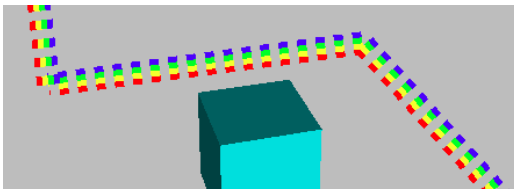


Figure 18: An O4J dashed, wide, multi-color line.

Animation

An easy-to-use animation framework provides animated transitions between geometry states. Animated changes in geometry provide the user with a transition that draws attention to what is changing without the abrut loss of context that occurs when changes are immediate. Animations can be triggered simply by providing the viewer with a time range to tween the geometry changes, e.g. over the next second; or full keyframe based animations can be set up for specific scene elements.

Interaction Events

Using O4J, graphics can respond to user events (tap/click, hover, double-tap/click, drag, pinch, etc.) for interactions such as filtering, navigation and selection. This makes any object in the scene available for use as a controller for other objects or as a trigger for events and interaction. For example, tapping on a label could be used as a drilldown to filter out only data related to items similar to the label.

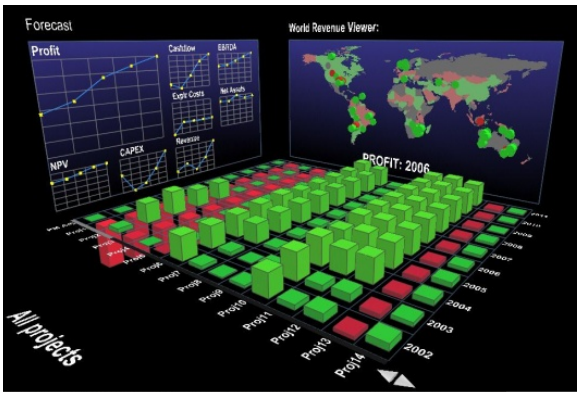


Figure 19: Users can interact with this business model, clicking and dragging to adjust factors such as capital expenditures, sales forecast and inventory and see the results update immediately.

Data-rich Tooltips

Tooltips are 2D graphics that are presented in the overlay (on top of the 3D scene), typically when the user hovers the mouse or taps on a piece of geometry. These can be very effective for presenting additional drill-down details on graphical representations of data. Tooltips can contain any combination of text, 2D graphics or even 3D scenes.

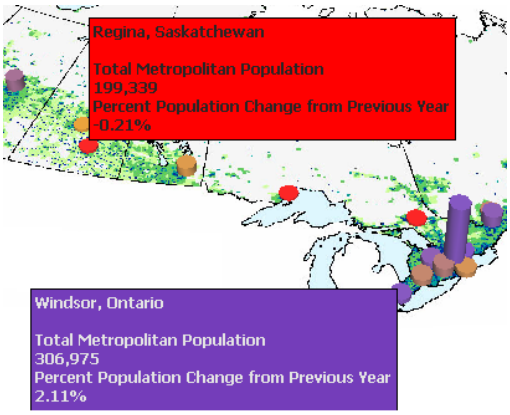


Figure 20: Data rich tooltips, displayed with the same color-coding as the data objects.

Screen Sprites

O4J supports screen sprites, which draw an image as an icon at a location in the 3D scene very similar to how screen text is drawn flat to the screen and at a constant size. These can be used for highly scalable point clouds, or to help visualize complex elements as an icon.

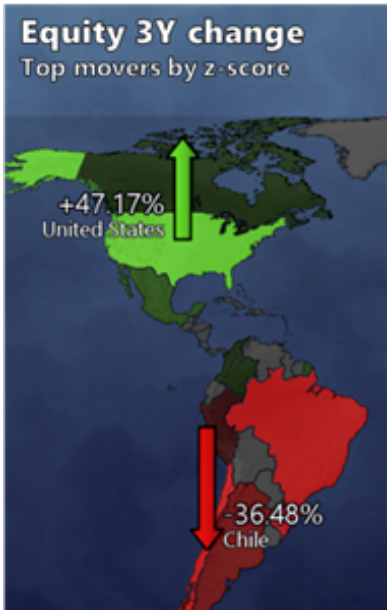


Figure 21: Screen sprites draw at a constant size for perfect readability from any 3D viewpoint.

High Resolution Output

Output capabilities have been implemented to facilitate generating snapshots of the scene as an image, which in turn could be embedded into another document (e.g. PDF) or sent to the printer. By using a tiled multi-pass rendering strategy under the hood, output can be generated by the library at a much higher resolution than the current screen display, e.g. for high quality poster size output.

Extension Libraries

2D Cross-Platform UI Widget Library

Creating cross-platform applications typically require the developer to port traditional interface items – such as labels, buttons, slider bars – to the native UI library for each platform. Uncharted provides a touch-friendly cross-platform user interface toolkit, similar to what one might find on a smart phone. It is designed to work in the Overlay layer and facilitate a "write once" development of UI layer components for cross-platform deployment.

Skins are supported for customizing the size and appearance of widgets for various devices and contexts. A client may choose from a predefined skin, or define and supply their own using only xml configuration and a few image resources (no code required).

Multi-screen and Stereo Displays

While a single desktop computer can support 4 or even 8 monitors, there are situations requiring more screens (e.g. video walls, megapixel walls) and/or coordination of multiple angled screens (e.g. immersive environments such as a CAVE) that require multiple computers all interacting with the same 3D scene. O4J provides an extension library that coordinates real-time distributed rendering across any number of networked computers, using O4J's serialization and asymmetric frustum capabilities coupled with a multicast scene update architecture.

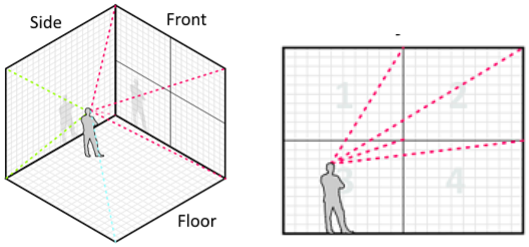


Figure 22: Multiple screens, whether flat or angled, can be configured into a single seamless 3D viewing environment. O4J automatically sets up the asymmetric viewing frustums, synchronizes scenes, and coordinates all interactions across all computers.

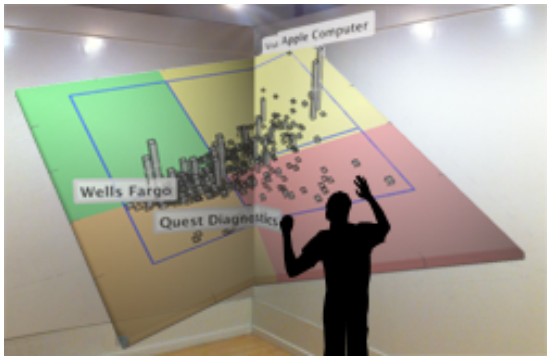


Figure 23: Two projectors each provide half of the scene: O4J adjusts the perspective properly to make a singular scene.

The end-user sees and interacts with single continuous scene. The API allows the developer to provide dynamic updates to each of the screen locations (and gaps if any), dynamic viewpoints (e.g. head tracking, game devices) and stereoscopic rendering (i.e. support for 3D glasses).

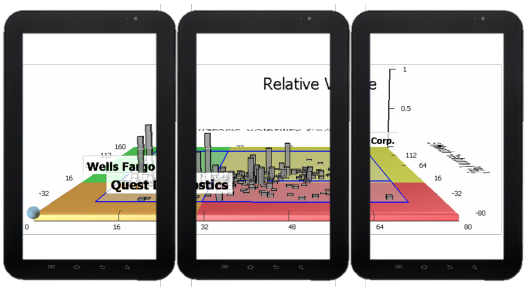
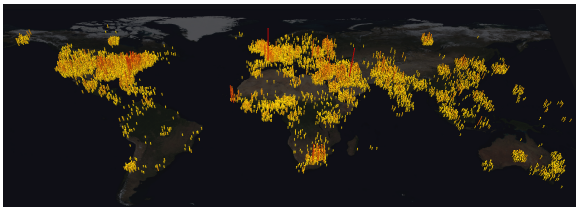


Figure 24: A single tablet visualization can be expanded across multiple tablets in a collaborative application.

In addition to display walls of any size, and stereoscopic immersive environments, multiple mobile devices can also be coordinated into a single view.

Mobile Terrain

Mobile Terrain is a mapping (GIS) and collaboration framework optimized for mobile deployments. Contact us for more information.



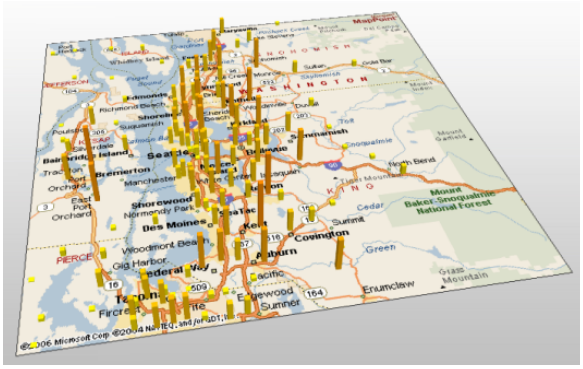


Figure 25: Mobile Terrain provides mapping functionality for embedding maps such as satellite imagery or map tiles into scenes and calibrating additional markers and graphics onto the maps.

Cross-Platform Deployment

O4J supports deployment to mobile devices with Android or Java ME, extending the same powerful cross-platform desktop capabilities to handheld devices using a single API. 32 and 64 bit Variants of Windows, Linux, and Mac OS X are also supported.

Android

Deployment to Google's Android mobile device platform is supported, with OpenGL ES based graphics acceleration.

Blackberry

Graphics acceleration for Blackberry is provided, for those Blackberry devices that support OpenGL ES.

Java ME

Deployment to base level Java ME is supported for older mobile and Blackberry devices, without requirements for specialized and rare JSR's.

Windows, Mac OS X, Linux

Support for desktop includes 32 or 64 bit OpenGL based rendering for Windows, Mac OS X and Linux, in addition to O4J Pure Java Software 3D rendering. The library selects and loads the appropriate platform specific classes and native support automatically at runtime, meaning the same application bundle may be shipped for all desktop variants, simplifying deployment.

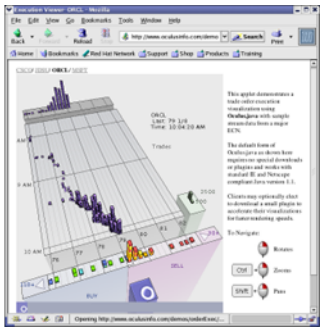


Figure 26: The same visualization as an Android application and on Red Hat Linux.

100% Pure Java

All functionality in O4J toolkit is available in 100% pure Java. Pure java provides the maximum potential for cross-platform portability. O4J has been used by customers in cross platform environments with no modifications. O4J has been architected for high performance, even when running in pure java mode.

Hardware Acceleration

Most users wish to take full advantage of accelerated video cards to take advantage of the best performance available. O4J provides acceleration extensions for Microsoft Windows, Linux, Mac and Android that will create a seamless bridge to any OpenGL accelerated video graphics at runtime, where available. And because it's seamless, the developer doesn't need to write any special code for hardware acceleration. O4J does it for you.

Optimization and Performance

The desire to push the limits of scene complexity within the tight restrictions of mobile devices has prompted a concerted effort to improve library performance even further. In addition to the multiple paths required for rendering to different devices, other forks have

been added to take advantage of advanced graphics card capabilities at runtime if discovered. The multi-threaded architecture has also been further tuned for fast setup of textures and geometry for OpenGL ES as well as traditional OpenGL.

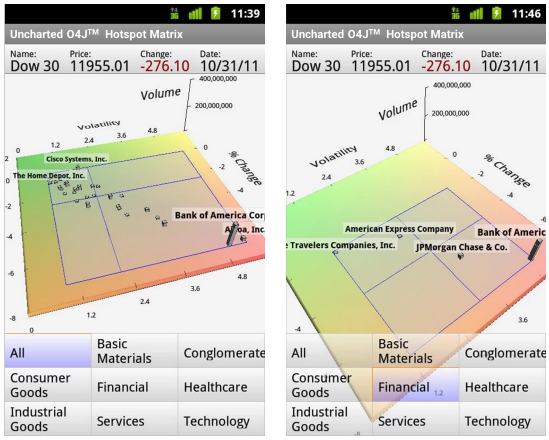


Figure 27: This stock market visualization animates at sub-second intervals on Android devices - demo version available on Google Play.

Small Footprint

The need to have quick downloads and minimize the use of limited device memory has resulted in a small memory footprint. The core O4J library is under 1 Mb, and visualizations can be developed that are as small as a 400kb jar file.

2D Cross-Platform UI Widget Library

As introduced in sections above, Uncharted provides a touch-friendly cross-platform user interface toolkit designed to work in the Overlay layer and facilitate a “write once” development of UI layer components for cross-platform deployment.

Deployment

O4J Visualizations can be deployed in various scenarios, including:

- **Java Applications**: are effective for large scale desktop applications or stand-alone mobile apps. All the visualization software runs on the client device.
- **Java Applets**: access is as simple as viewing a web page.
- **Zero Client**: The visualizations can reside server side as a service that generates static images (e.g. PNG files), for consumption by reports (e.g. PDF) or for dynamic composition into web pages that utilize JavaScript to allow for interactions such as clicking for selection, filtering and drill-down.

Note that it may be desirable to create an O4J visualization which is deployed using more than one of the above strategies, such as a trade tracking visualization deployed on a web page for a wide audience and as an application for an internal fraud analysis group.

Conclusion

The O4J software toolkit is a state-of-the-art class library that allows custom multi-dimensional business data visualization solutions to be delivered across a wide range of platforms.

With the power and ease of use provided by Uncharted's toolkit, 2D and 3D data visualization solutions can now be deployed to many users across platforms at very low cost. Additionally, O4J was engineered to be data-source neutral - ensuring connectivity to critical legacy and pre-calculated data as well as real-time data feeds and integration within other applications and their pre-established data models.

The power inherent in multi-dimensional user interfaces, allows for far more comprehension and better understanding of complex data. This enhanced interaction adds up to huge revenue potential and considerable business value.

Professional Services

Uncharted provides services above O4J to assist our customers get maximum value from visualization. Uncharted is a full-service company: in addition to training and knowledge transfer, Uncharted professional services help our customers create the right visual designs to meet their business objectives and user capabilities; design application architectures to integrate within the enterprise environment; and develop prototypes, pilots and production class visualizations.

Contacts

For more information on O4J and other Uncharted 3D visualization toolkits, and our visualization services, please email us at **info@uncharted.software** or contact:

Richard Brath

Managing Partner, Commercial Innovations
Telephone: (416) 203-3003, ext. 242
richard.brath@uncharted.software

David Jonker

Senior Partner, CEO
Telephone: (416) 203-3003, ext. 223
david.jonker@uncharted.software

Office Location

Uncharted Software Inc.
2 Berkeley Street, Suite 600
Toronto, Ontario M5A 4J5

Telephone: (416) 203-3003
Facsimile: (416) 203-0646

www.uncharted.software