# Streaming Graphics

Andy Norton, SPSS
Lee Wilkinson, SPSS

# Dancer

- Dancer is an SPSS project for streaming and animated graphics
- Dancer is an *extensible* framework, written in Java.
  - Domains, functions, statistics, elements, renderers

# Overview

- Table
  - Provides access to data values
- Frame
  - Coordinate system for displaying elements
- Element
  - Represents a data source within a frame

# Demonstration

- Table
  - Random walk for each point
- Frame
  - 3D
  - With color ramp from dark-green to yellow
- Elements
  - Point
  - Surface (Loess smooth)

# Agenda

- Tables
- Frames
- Elements

# Tables

- Data streams are mapped to tables
  - *Rows* represent objects
  - *Variables* represent attributes
  - *Cells* contain the values
- Tables may have any implementation that can determine
  - What rows exist?
  - What is the current value of a cell?
- May be local or remote.

# Domains

- Domains
  - declare a set of possible values
  - may include values not currently in the data.
  - may be extended to include new values.
- Domains are used to
  - Validate that any data values can be handled
  - Adapt views to best display the data.

# Event Notification

- How do we know where changes have occurred?
  - Constant variables
  - Always-changing variables
  - Event notification
- Pseudo-variable _exists_ identifies presence of row

# Agenda

- Tables
- Frames
- Elements

# Frames

- Frames establish the rules by which values are represented.
- These rules map
  - From data value
  - To aesthetic value
- Frames display guides (axes and legends) explaining these rules

# Legends

- Legends display mappings
  from data values to aesthetic values.
- Legends in Dancer belong to the frame,
  and thus apply to all elements.
- Legends are also controllers, allowing you to
  change mapping rules on the fly.
- User can dynamically show and hide legends.

# Facets

- Each facet is an independent filtered view of the data
  - Facets need not cover the data.
  - Facets need not be disjoint.
- Facet displays may be different sizes
  - Allows for foreground and comparison
- User can dynamically show and hide facets.

# Continuity: The Problem

- *Continuity* is an issue whenever one property is automatically assigned, based upon another changeable property.
- Unnecessary change distracts the user.
- Implication:

  Don’t optimize each rendition separately.

  Trade off optimality for continuity.

# Stability: Example

- Example
  - Ticks are determined from bounds

    bounds = [ 0, 100]

    ticks = {0, 20, 40, 60, 80, 100}
  - If bounds change, the new ticks should be related to the old.

    bounds = [ 50, 150]

    ticks = {60, 80, 100, 120, 140}
- Other examples
  - tick values, assignment of symbols and colors, sort order, text formats, page layout

# Stability: Solutions

- Maintain same pattern for extensions
  - For example, same interval and offset for ticks.

    Old {0, 20, 40, 60, 80, 100}

    New {60, 80, 100, 120, 140}
- Be liberal for additions
  - There may be more additions to come.
- Be conservative for deletions.
  - The deleted value may reappear soon.

# Agenda

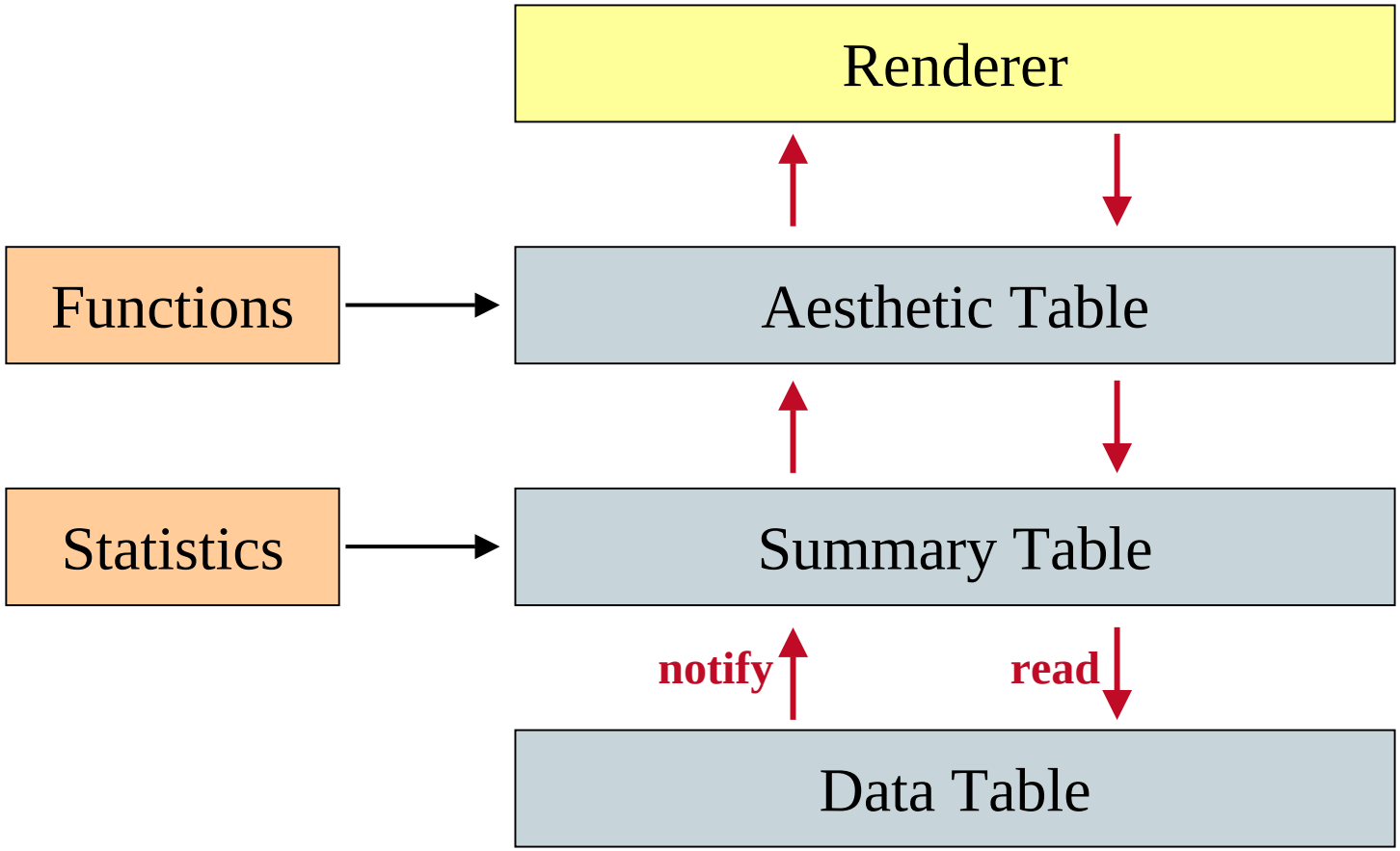
- Tables
- Frames
- Elements

# Elements

- Elements in the same frame may be based upon different tables.
- Elements refer to the frame for aesthetic functions.
- Elements may be dynamically added and removed.
- Current elements:

  bar, link, point, surface, path.

# Functional stack



# Read on Render

- Mark dirty on changes
  - Fast
  - Coalesces repeated changes
- Read on render
  - Up to date
  - Only read when needed
  - If overloaded, reads less often

# Multi-threaded

- Notification thread
  - Minimize load
- Rendering thread
  - Set maximum frame rate to keep resources available.
    For example, 10-40 fps.
- GUI Thread
  - Respond quickly to GUI activity.

# Relativity issues

- The Problem
  - Values can change during rendering.
  - Not just outdated, but systematically off

# Relativity issues

- The Problem
  - Values can change during rendering.
  - Not just outdated, but systematically off

# Relativity issues

- The Problem
  - Values can change during rendering.
  - Not just outdated, but systematically off

- Solutions
  - Read all data before rendering.
  - Alternative:  estimate data as of given time
    - Interpolate from last frame
    - Maintain a sliding window of past data

# Demonstration

- Weather from Humboldt, Greenland (78º N)
- Greenland Climate Network (GC-Net)
  http://cires.colorado.edu/steffen/gc-net/gc-net.html

# Demonstration

- NASDAQ stock trades for Microsoft (MSFT)
- April 2, 2002
- Last 3 minutes of trading

# Contact Information


- Andy Norton
- SPSS, Inc.
- 233 S. Wacker Dr, Chicago IL 60606
- 312-651-3154
- anorton@spss.com