Illustration Types in Plone

2005 North American Plone Symposium
New Orleans, LA, July 22, 2005

Jeff Pittman
Dept. of Earth and Space Sciences,
Lamar University
Beaumont, Texas
ess.lamar.edu

geojeff.org for software releases

Illustration Types in Plone

- There is a common need for custom types for generation of images and SVG and PDF documents. Aren't you a visual learner or thinker? Say yes!
- This effort provides a “dev pack” of example types to use as go-bys.
- Examples include SmileyFace, TeachingSchedule, EarthMoonSun, GeologicalTimeScale...

My Life as an Illustrator

"No Pain, No Gain"
In the Steady March of … Progress

Rapidograph
(late 1970s-1980s)

Zipatone,
Transfer Lettering
(early 1980s)

Dot Matrix and,
FancyFont™,
Pixels
(mid to late 1980s)

Programmable Graphics

Specialized Software
for custom generation
of ready-made graphics.

CorelDraw™
Illustrator™
Canvas™, etc.
for
Vector Graphics
(late 1980s to Now)

First with Java and the JAI,
Now Python, SVG, PIL, ReportLab
(mid 1990s to Now)

It is easy to forget the power we have now
to do things that were previously not feasible.

SVG (Scalable Vector Graphics)

- 2D Vector, Raster Graphics, Text
- W3C Standards
- XML-based
- Interactivity, scripting
- Better browser support coming

For SVG 1.2:

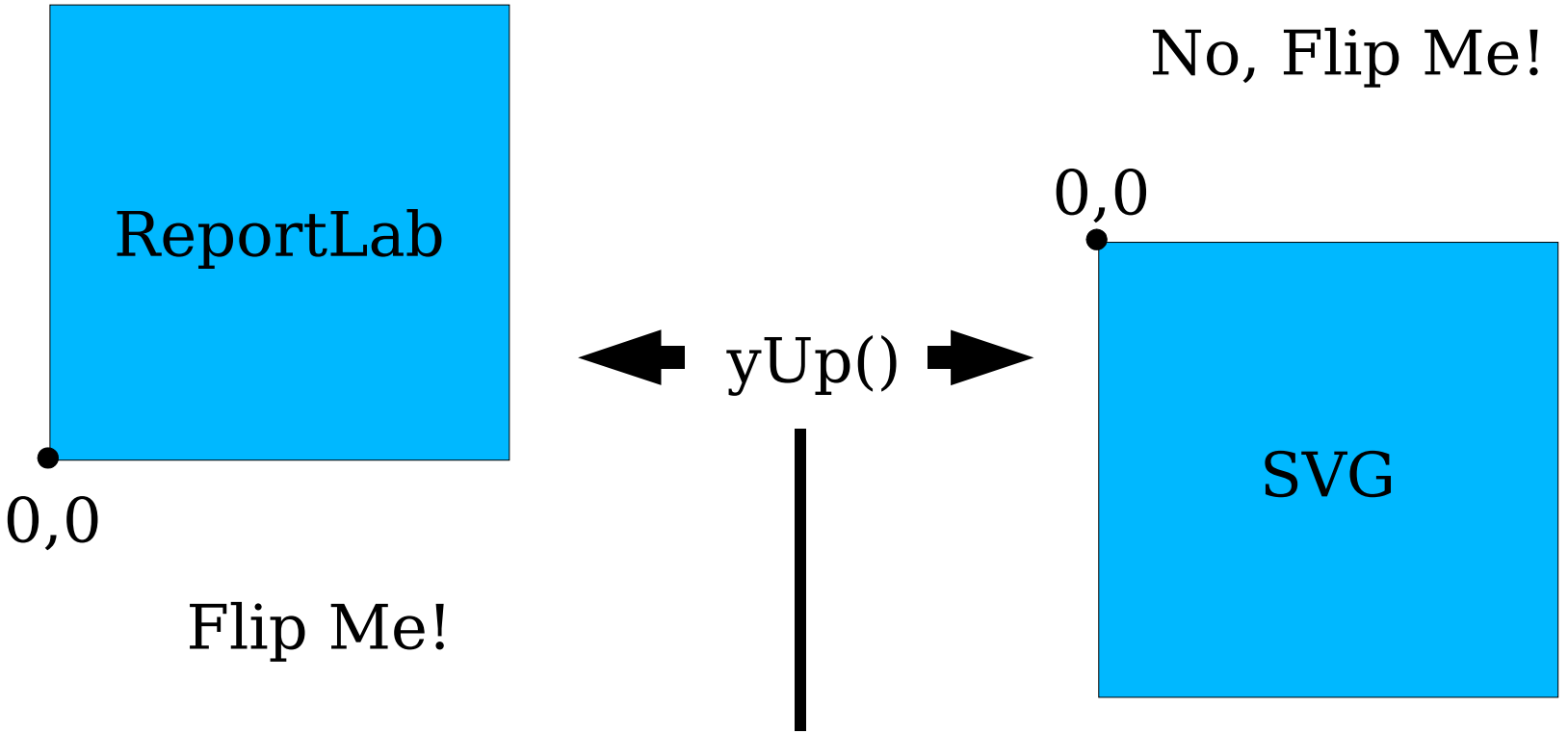
- Xml Binding Language
- Text Wrapping
- Editable Text
- Audio/Video
- Transitions
- Multipage Support
- Streaming, Timing, Vector Effects, Better Scripting

See report by Andreas Neumann, carto.net

SVG and **ReportLab**

have different Y-Axis Origins.

Hey, Variety is the Spice of Life.



Use a simple method to flip y-values,
which returns (height - y-value).

ReportLab generation

```
drawing.add(Circle(self.x_offset+(s/2), self.y_offset+(s/2), s/2,
                   fillColor=self.fillColor, strokeColor=self.strokeColor,
                   strokeWidth=max(s/38.,2.)))

for i in (1,2):
    drawing.add(Ellipse(self.x_offset+(s/3)*i,self.y_offset+(s/3)*2, s/30, s/10,
                        fillColor=self.fillColor, strokeColor = self.strokeColor,
                        strokeWidth=max(s/38.,2.)))
# ...
# some math to get the x,y points for the curve of the smile
# ...

smile = PolyLine(pointslist,
                         fillColor = self.strokeColor,
                         strokeColor = self.strokeColor,
                         strokeWidth = max(s/38.,2.))

drawing.add(smile)
```

For ReportLab, you use Python calls to add drawing primitives to a ReportLab Drawing instance, and it is a simple matter of effectively using the well-documented API to make your custom drawings.

SVG generation

For SVG generation, a list of strings is made. It is a bit persnickety, as you need to escape quotes, do 'printf' type parameter substitution.

```
smiley_svg = \
   ["    <circle cx=\"%d\" cy=\"%d\" r=\"%d\" " %\
        (self.x_offset+(s/2),yUp(self.y_offset+(s/2)), s/2),
    "style=\"fill:%s;stroke:%s;stroke-width:%d\" />\n" %\
        (colorstr(self.fillColor),colorstr(self.strokeColor),self.strokeWidth)]

for i in (1,2):
   smiley_svg += ["    <ellipse cx=\"%d\" cy=\"%d\" rx=\"%d\" ry=\"%d\" " %\
                       (self.x_offset+(s/3)*i,yUp(self.y_offset+(s/3)*2), s/30, s/10),
                       "style=\"stroke:%s; stroke-width:20\" />\n" %\
                       colorstr(self.strokeColor)]
# ...
# some math to get the x,y points for the curve of the smile (calls yUp())
# ...

smiley_svg += ["<polyline points=\"%s\" fill=\"none\"
                         style=\"stroke:%s;stroke-width:%d\"/>\n" %\
                     (self.getPointsListString(pointslist),
                     colorstr(self.strokeColor),max(s/38.,self.strokeWidth))]

return smiley_svg
```

SmileyFace Code

(in Zope Products directory)

Main Archetypes code for creating folder, data input widgets, and the main controlling class.

This contains the actual drawing code for direct SVG and for creating a ReportLab drawing.

This page template has code for showing a list of generated items.

Extensions
    Install.py
CHANGELOG.txt
README.txt
version.txt
__init__.py
config.py
SmileyFace.py
FrecklesValidator.py
drawing.py
utils.py
sample_data.py
skins
    SmileyFace
        SmileyFace_view.pt
        SmileyFace.css.dtml
        smileyfaceicon.gif

Design and Description

- Archetypes-based – uses standard widgets with customization for input validation of CSV data.
- Folderish – goal is generation of one or more targets (images, PDF document, or SVG document), which will be held in a folder. The view shows a list of generated items, to be clicked for viewing. A custom view is also easy.

Design and Description, cont.

- Generation – for SVG, examples are shown for emitting directly; all other targets are generated with ReportLab (jpg, png, PDF, etc.). ReportLab can also be used to generate SVG.
- Custom SVG – offers functionality for interactivity using Ecmascript. Examples are given for Y-axis flipping and basic graphics generation.
- Colors – Nice (time scale uses colors from usgs.gov).

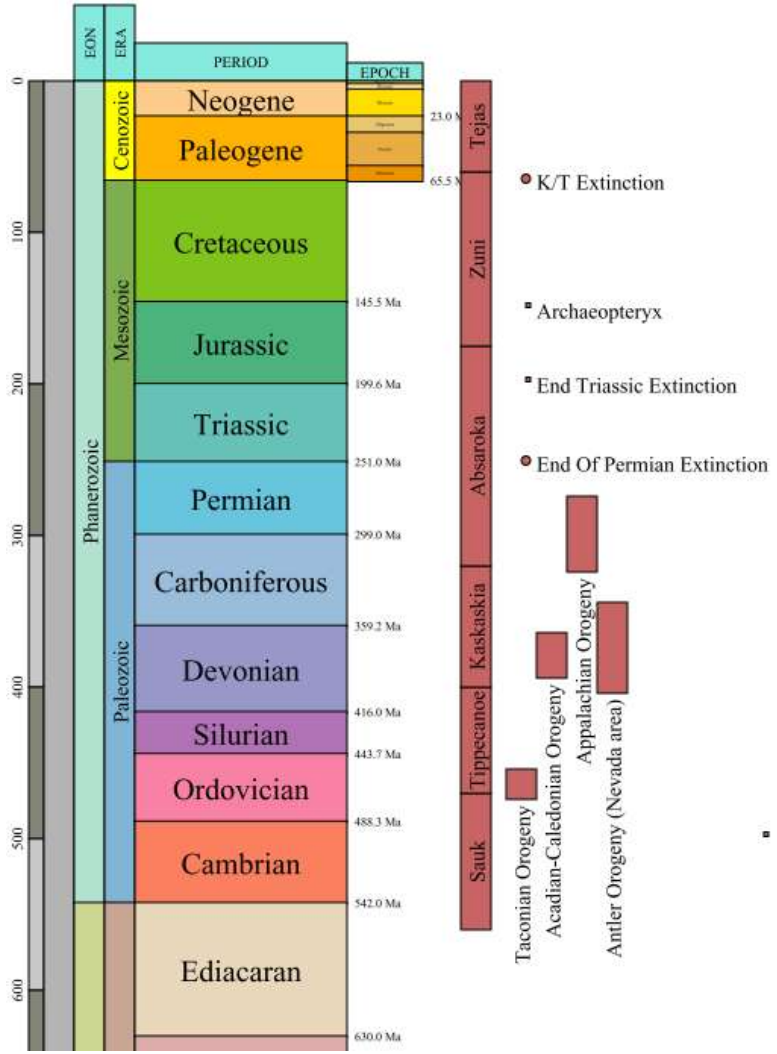
Design and Description, cont.

- Editing – graphics targets are regenerated after data are modified; old versions are simply replaced; if versioning needed, copy.
- Scope, scale – there will be only one, or perhaps as many as 15 or 20 instances – so, robustness is assumed (SVG is compact, if size is a concern).
- Charts – other products handle charts, but this system could be useful.

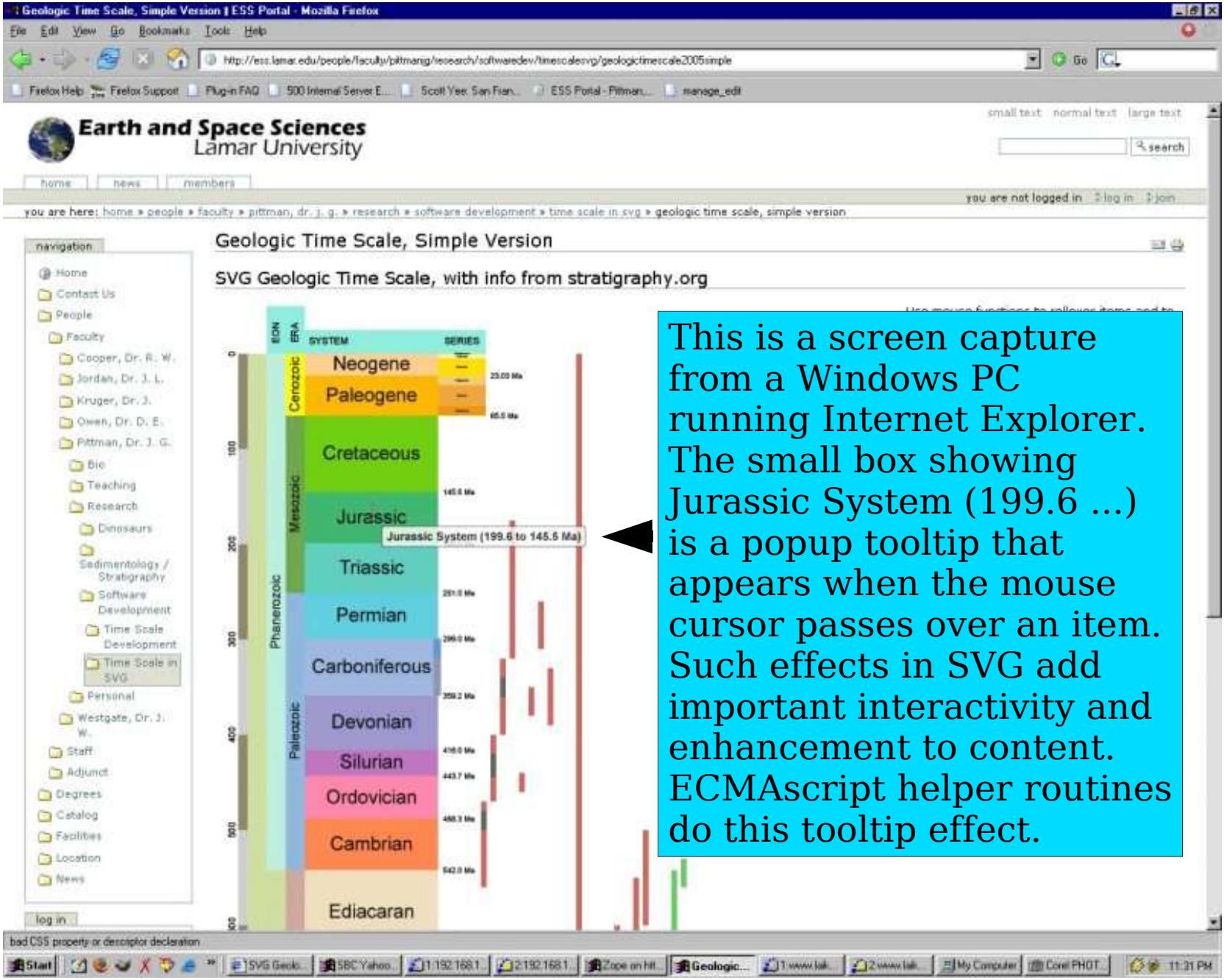
Design Questions

- When to use CSS and when to "draw"?
- CSV data input – pragmatic? efficient? Popular with users?
- DevPack or Individual "Products"?
  - On plone.org, which is better?
- Best practices? -- Discovery of the obvious? Violation of the obvious?
- Tricks? – e.g., CSS overflow:scroll?

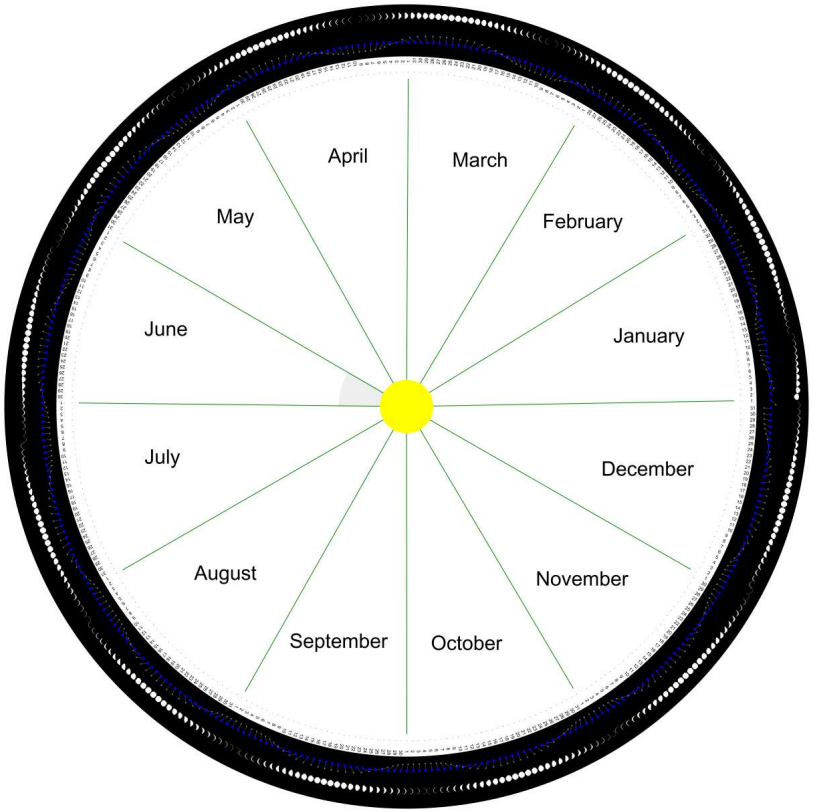
GeologicTimeScale (The Inspiration)



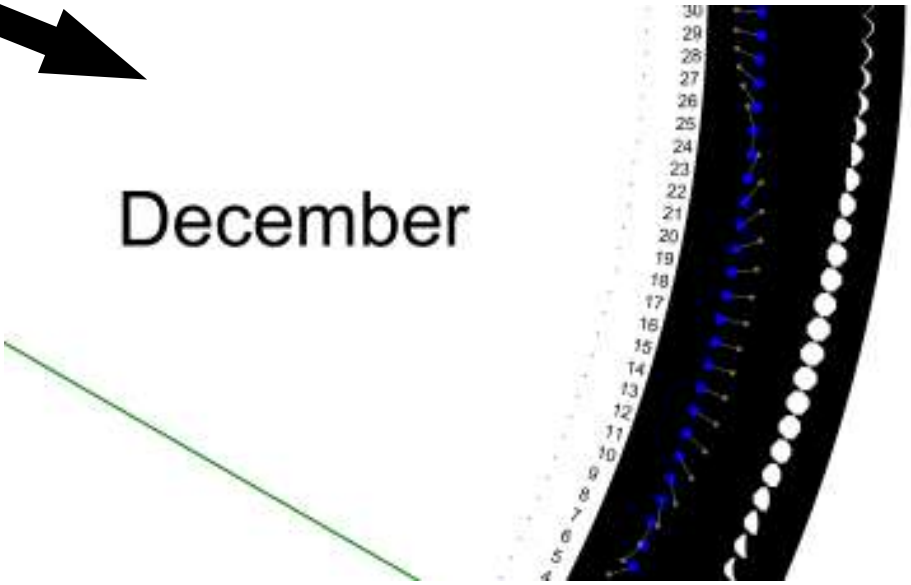
- Data from stratigraphy.org
- Standard colors from usgs.gov
- Simplified and Complete Versions
- point-in-time events and range events (the bars)
- Examples: Cretaceous birds, Eocene Mammals of Wyoming, Cephalopoda...
- Rollover text effects in SVG


Geologic Time Scale, Simple Version | ESS Portal - Mozilla Firefox
Earth and Space Sciences
Lamar University
Geologic Time Scale, Simple Version
SVG Geologic Time Scale, with info from stratigraphy.org
EON
ERA
SYSTEM
SERIES
Phanerozoic
Cenozoic
Mesozoic
Paleozoic
Neogene
Paleogene
Cretaceous
Jurassic
Triassic
Permian
Carboniferous
Devonian
Silurian
Ordovician
Cambrian
Ediacaran
Jurassic System (199.6 to 145.5 Ma)
This is a screen capture from a Windows PC running Internet Explorer. The small box showing Jurassic System (199.6 ...) is a popup tooltip that appears when the mouse cursor passes over an item. Such effects in SVG add important interactivity and enhancement to content. ECMAscript helper routines do this tooltip effect.

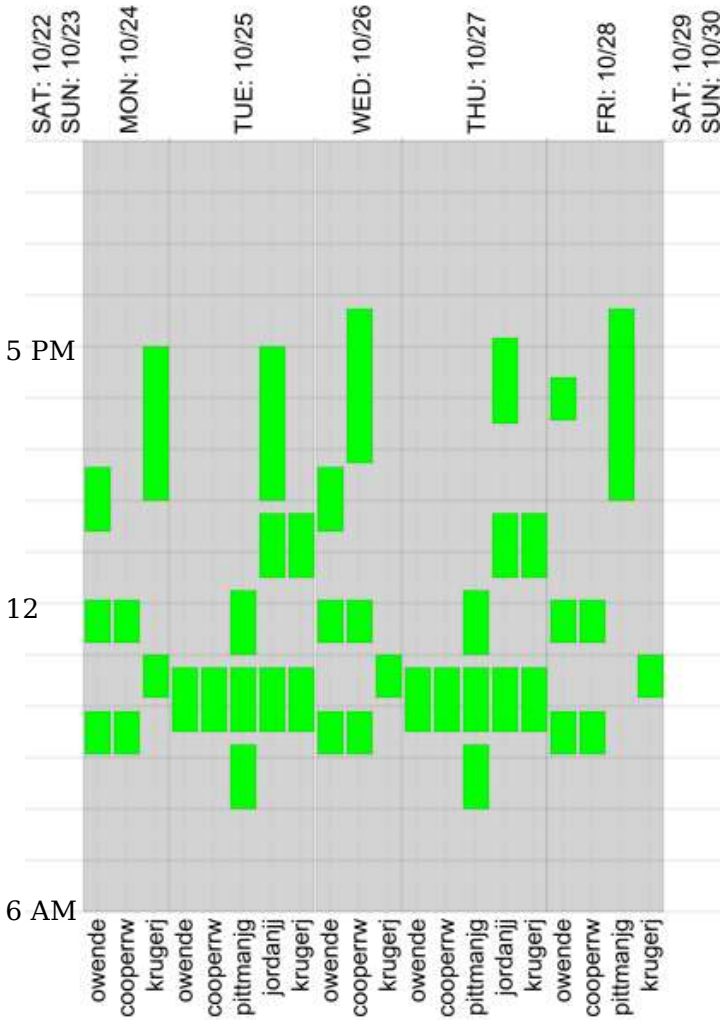
EarthMoonSun (A “Solar” Calendar)



- Phases of the moon are the focus; shows current day or month



TeachingSchedule



- Data is for recurring events (college classes) for each teacher.
- The plot produced is very wide, if run for a whole semester.
- A column for each day shows classes, identified by time and teacher.
- This is just a snippet from a very wide drawing.

More Illustration Types

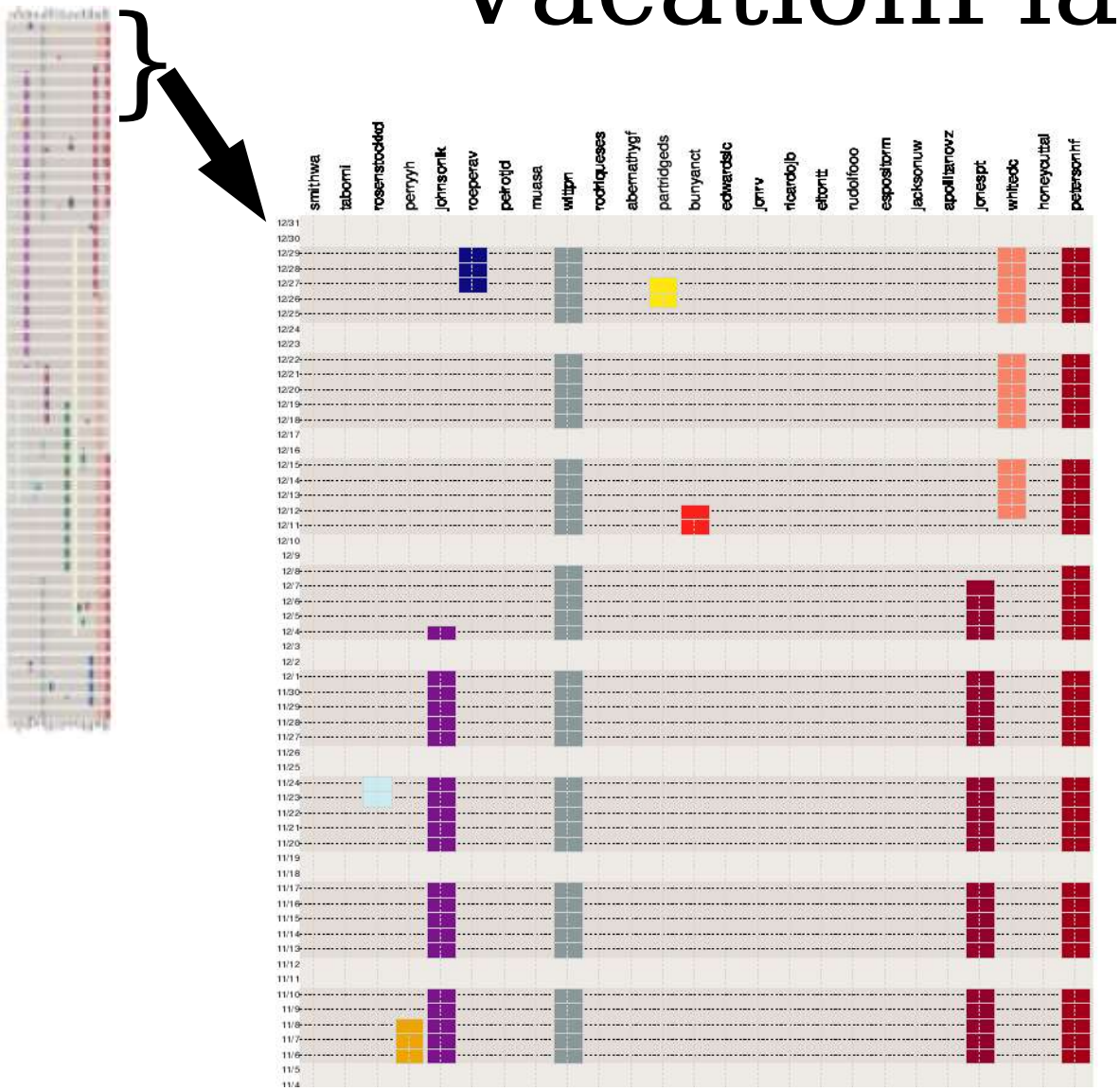
VacationPlanner

- design can closely follow TeachingSchedule, with CVS data for vacations to include person info, start day, day_portion (all, morning, afternoon), duration, etc.

- This might be a case where use of a custom type for Vacation would allow users to add their own Vacation instances to the folder. The calendar plot-generating code would read the data from these, instead of CSV data entered in a Lines field. Or users add their own Vacation instances to their home folders, etc.

- There are “horizontal” and “vertical” versions of the calendar plot.

VacationPlanner



- Vertical layout is “geological” and, in this case, seems to be appropriate (most recent is on top).

- Vacation days are specified as CSV lines with data for person, start day, duration, ½ day, full day, etc.

- I don't get vacations, so why do I care? :)

More Illustration Types

- Evolutionary Trees (would use drawtree.py by Rick Ree)
  - Data: nested groups for drawing branching “tree” diagrams to show evolutionary relationships.
- BookAnnotation (needs programming)
  - Data: page and line number for words and topics needing description.
  - Small format pages for printing.
  - Displays for interactive sequential scrolling and searching for specific pages.

242

Cordillera

Mr. Low

Gregory Bay

Sarmiento

Port Famine

243

their tattered clothes had been burnt by sleeping so near their fires.

Mount Tarn

244

death-like scene of desolation

246

Winter's Bark

Dr. Hooker

cryptogamic

BookAnnotation

- **mock-up of multipage panel for Charles Darwin's 'Voyage of the Beagle'**
- **shows 8 pages (as example)**
- **would have popup mouseover tooltips for terms and topics**
- **A three-wide, nine-deep portrait style layout would work better for online viewing on a Plone page**

247

true creeper (Certhia familiaris)

Harpalidae

Heteromidiae

Succinia

Pyr

Kerguelen Land

249

Flustacea

the miserable lord of this miserable land

250

Magdalen Channel

The inanimate works of nature – rock, ice, snow, wind, and water – all warring with each other, yet combined against man – here reined in absolute sovereignity.

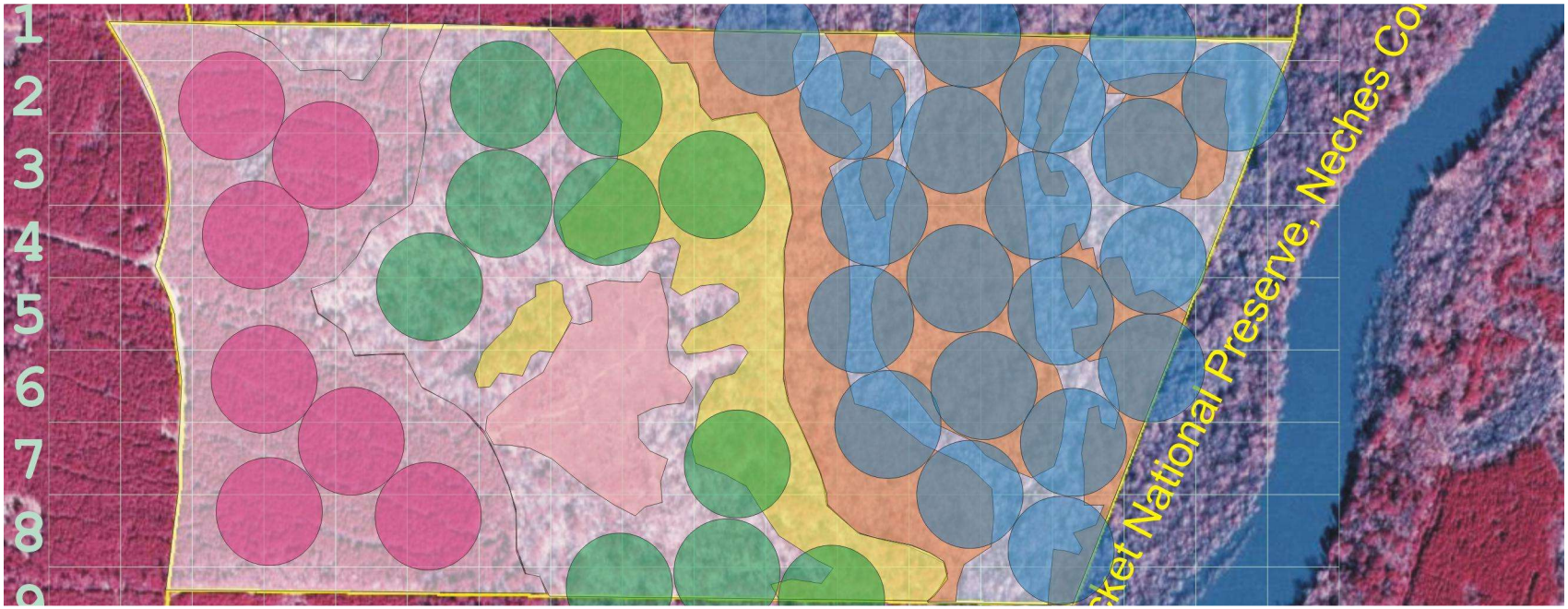
relief

More Illustration Types

- Custom Maps (See carto.net for SVG)
  - Using PIL and SVG, aerial photos and infrared images are cropped and chopped for generation of SVG-based maps for scientific surveys.
  - Regular scientific surveys often need maps for showing a random data collection scheme generated by software.
  - Generation/storage in Plone would offer:
    - Archiving, Organization, Centralization
    - Monitoring by collaborators/mentors
    - Incorporation of Results for Presentation

My Ecology Research Project – SVG, PIL

Plone as Repository, as Hub for Collaboration



This is a portion of a map showing an infrared image of an East Texas forest area, where an ecological study is being done. Using Python, PIL, and SVG, maps are produced for each field session, for showing locations of randomly selected data collection sites. Instances of this map would be created several times each month, for each field excursion, and would be saved in an archive folder for scientific data validation purposes, and for providing a means for real-time monitoring of the project by collaborators and managers. There would, of course, be many other benefits and uses in Plone.

Conclusions

- Archetypes-based illustration content types offer a useful means of taking user input and creating graphical plots and drawings. Basic drawing “harness” is complete -- now for:
  - evenness of browser support
  - effective cross-platform (browser) design
  - exploring enhanced user interactivity via SVG, CSS.
- Thanks to Plone Community
- Thanks to creators of SVG, PIL, ReportLab, Python, etc. tutorials and code

Follow-Up Questions/Answers

- Availability: Check geojeff.org which will always have an up-to-date listing and status of projects. Individual products will be added to products area of plone.org as they are approved and published.
- Could this system be used for very large diagrams with many parts? Yes, in fact, in considering the SVG-based interactive mapping applications in carto.net examples, use of this system seems appropriate. Small images that are part of the UI, multiple SVG files, and multiple ECMAscript files could be maintained by such a folderish AT-based Mapping controller application.

Follow-Up Q/A, continued

- CSV input: A participant commented that use of ExternalEditor could work well, and also that use of a spreadsheet by “normal” users, along with exporting to CSV and then pasting into the Archetypes lines field widget, would help make CSV input viable.
- Could TeachingSchedule or VacationPlanner be tied to events? (a question asked in the elevator after the talk) Yes, in the main product class holding the AT schema and code, an events folder could be read to rebuild graphical representations of the events, showing time conflicts, firing conflict reports, finding open slots for an event, etc.

Follow-Up Q/A, continued

- SmileyFace, boosted?: A symposium attendee asked if I had seen the use of a smiley face as a display to show the state of 10 variables. I found the following page -- http://members.aol.com/DBBoles/bc36b.html -- which describes reseach on the topic, and has links to articles. I am not sure if this is the project the attendee was recalling, but suspect that it is, because he mentioned 10 variables. Perhaps SmileyFace could be modified to present a dynamically changing SVG smiley face, controlled by ECMAscript helper routines – maybe this would get some really smart people to use Plone Zone :)