VisualEyes Tutorial

PLEASE NOTE

This tutorial is still under construction.

About this tutorial

This tutorial will introduce you to working with VisualEyes by walking you through your own project: "After the Greenwich Tea Party: New Jersey During the American Revolution." In Chapter One, you will create a new project designed to plot the battles and skirmishes of the Revolutionary War in New Jersey. In Chapter Two, you will learn how to edit your project and add features. In Chapter Three, you will learn how to create paths and dots. In Chapter Four, you will learn how to import data into your project using .xml files. And in Chapter Five, you will learn how to present the data in infoBoxes (short for "information boxes") that can interact with the viewer.

Chapter One

In this chapter, you will learn the basics of VisualEyes:

- Projects, Views, Resources, Displays, and Controls
- Introduction to the VisEdit Tool
- You will learn how to:

1. Create a new project
2. Understand VisEdit
3. Understand elements and sub-elements
4. Edit a project with VisEdit
5. Save a project with a new name

Understanding VisualEyes: Projects, Views, Resources, Displays, and Controls

Before you begin creating a new VisualEyes project, you'll find it helpful to take a look at an existing one.

- Point your browser to http://www.viseyes.org/show/?base=jt. After a few moments, the Jefferson's Travels visualization will load.
- Take a few minutes to examine the screen.

[Insert screen shot]

The entire set of features you see on the screen – the set of three tabs, the timeline, any additional instructions, text and images – make up the *project*. Each tab represents a *view*, a set of features designed to present one aspect of the material. The information that goes into the view – the maps, text, images, and data – are called the *resources*. And the building blocks that allow you to interact with the resources – such as the *map overview*, *timeline*, and *control panel* – are called the *displays* and *controls*.

The first tab, Jefferson's Travels, gives a geospatial representation of Jefferson's travels in England in 1786. The background map shows southern England in the 18th century. A *slider* on the left side allows you to zoom in and out. When you zoom in, the *map overview*, in yellow, allows you to keep track of where you are on the large map.

To track Jefferson's travels across the map, you can press the Play button [image] on the *timeline* at the bottom of the project.

- Click on the Play button to watch the path of Jefferson's travels. A small portrait of Jefferson traces his ***path*** across the map. That ***path*** is linked to the ***timeline***, so that Jefferson's travels can be pinpointed very precisely in time as well as space. Both ***path*** and ***timeline*** are linked to an ***infoBox*** showing Jefferson's memoranda for each date.

Note that the map switches at two points, to a map of 18th century London. This allows viewers to chart Jefferson's travels more closely. You can use the *speed control* [image] to speed up and slow down the timeline.

Explore other features of the Jefferson's Travels view:

- Stop the movie by pressing the *pause button* [image].
- Use the timeline slider to position the timeline on April 8, 1776
- Click on an *information button* [insert image]. A ***docviewer*** appears, providing images and text about the location. Click the

arrow buttons to move through the pages, and the close button [image] when you are finished.
- Click on a building *icon* [image]. This brings up an ***infobox*** that links to an external web page. Click the close button when you are done.

Next, take a look at the control panel, on the right of the view. This allows the user to display, or hide, other resources included in the view.

- Look under Geography. When you first load the Jefferson's Travels view, the "Show Memorandum" box is checked. Click on it to remove the check, and to hide Jefferson's memoranda.
- Click on "Show Graph" to display a graph of Jefferson's purchases, and on "Show Overview" under Map Features to remove the map overview.

The next feature on the control panel, "Show People Map," demonstrates a valuable feature of VisualEyes, a *concept map*. It allows viewers to explore networks of interrelated people, places, or things.

- Click on "Show People Map" to display Jefferson's social network while he was in England.
- At the center is Jefferson himself. Click on the image to see an infoBox with a brief description of his travels in the 1780s.
- Click on the other people in the people map to view their information. You can close each infoBox by clicking on its close button. Note that the lines connecting each person indicates whether he/she was linked to Jefferson through diplomatic, social, or commercial connections.
- When done, click on the close button to close the concept map and return to the main Jefferson's Travels view.

> **WARNING!** Do not attempt to get back to the main view by clicking on the Back button of your browser. That will close Jefferson's Travels completely.

The last feature of the control panel is a Search box. This is an advanced feature of VisualEyes, and is described in the VisualEyes Manual.

Tools: VisEdit

Now that you have explored Jefferson's Travels, you are ready to create your own VisualEyes project. You will do that using *VisEdit*, an editing tool specifically designed for VisualEyes. With VisEdit, you can create projects without having to work with the underlying software code.

- Point your browser to http://www.viseyes.org/edit.htm. The VisEdit screen will load. At the bottom, you will see a gray box saying "Please log in"
- In the Username text box, leave the default user name, guest
- Do not alter or delete the password (hidden by asterisks) in the password box.

Loading the Sample Project

Go to the File menu, and select the "Load By ID" item and type: "*samples"* with no quotes. This will load the samples project described in detail bellow.

- Take a few minutes to examine the screen.

[Insert screen shot]

At the top of the screen is a menu bar with four options: File, Edit, Tools, and Show. Your username, samples, appears on the right of the menu bar.

- Click on each to see the menu bar options. You'll learn what they do later.

Below the menu bar, there are three panels. The one on the left lists each of the ***element***s you create for your project. When you first start working on it, there is only one element in it: ***project***.

- Click on the word **PROJECT**.
- Take a few minutes to examine the screen.

In the upper panel on the right, you will see *attributes* associated with the element ***project***. The name of the attribute is listed in the *Tag* column. Choices you can make regarding the attribute are listed in the *Choices* column. Values the attribute can have are listed in the *Values* column.

If you want to create an account, click on the link below the UVa logo and fill in the information requested. Then, instead of using "guest" and "blank", use your new username and password. Using your email address is an easy way to get a memorable password.

When you begin a new project, attributes come with present values. These preset values are known as the *defaults*. As you can see on VisEdit, the default value for the attribute *title*, associated with the ***project*** element, is "My Project." The ***project*** element has another default attribute, *lastSaved*. Right now, that attribute is blank – that is, it has no value, because it has not been saved yet. The upper panel on the right will always show you the attributes for the element highlighted on the left.

In the lower panel on the right, you will see information about the element ***project***. It describes the element, and it also tells you the attributes – *title* and *lastSaved*, that the element ***project*** can have. The panel on the lower right will always provide additional information for working with the element highlighted on the left.

The Attribute and Information panels are *context-sensitive*: that is, they display information relevant to whichever element is highlighted.

Understanding VisEdit XML

VisualEyes projects are created using a program called *XML*. Although it is not necessary for you to work with the underlying XML code, you will find it helpful to understand how XML code is organized. Programmers think of XML as branching from the top down, through a kind of flow chart. The top level, often represented as an oval, is called a *node*. These nodes have lower-level branches, generally represented as smaller ovals. The lower levels *nest* within the nodes. In VisEdit, each element is a node. Each *attribute* nests within its element and modifies it. Each "value" nests within the *attribute*, and gives precise instructions for how the element should be modified. In this manual, ***elements*** will be indicated by bold italics, *attributes* in italics, and "values" in quotation marks.

As you get more experience with VisualEyes, you may decide to edit the XML directly, rather than using VisEdit. If you wish to work directly with XML, you will find it helpful to use a program like Dreamweaver.

Understanding Elements and Sub-Elements

Many elements, like ***project***, can hold other elements. You can think of ***project*** as the main folder, and the elements it holds as subfolders. You can see the subfolders by clicking on the arrow to the left of ***project***.

- Click on the arrow. A list of sub-elements appears: ***frame***, ***textformat***, ***tab***, and ***view***.
- Click on each of the sub-elements. As you do, the set of attributes associated with that element appears in the Attributes panel. The material presented in the Information panel changes as well.
- Take a moment to click on the ***view*** element. Note that it has one attribute, *title*, with the value "My View".

What do all of these Attributes do? To find out, you can preview what you have created so far.

- Click on the Save & Preview button on the bottom right of the Attribute panel. In a few moments, your project will appear.

[Insert screen shot here]

What you see is your Project as you have defined it, a very, very empty version of the Greenwich Tea Party project. The ***frame*** sub-element sets the size of the box that holds the project. Everything else you add to it will be contained within the ***frame***. The ***textformat*** element defines the size and type of the text. The ***tab*** element defines the colors for the tab at the top of the frame. And the ***view*** element defines the title of the view. Right now, you have only one view – with the *title* "My View" – but as you remember from Jefferson's Travels, a project can have multiple views. You access each view by clicking on its tab.

- Click on the Return to Editing button at the top of the screen to return to VisEdit.

WARNING! Do not attempt to get back to the main view by clicking on the Back button of your browser. That will close your project completely, and you will have to sign in again to VisEdit.

There are a few more aspects of the VisEdit screen to explore before you move on.

The Add new element list box at the bottom of the element panel will always display the available sub-elements for a highlighted element.

- Click on ***project*** in the element panel if it is not already highlighted.

You can find the list of available sub-elements by clicking on the Add new element list box at the bottom of the element panel.

- Click on the Add new element list box. You will a list of the available sub-elements for ***project***: ***frame***, ***logo***, ***tab***, ***textformat***, and (when you scroll down) ***view***.

Some of the sub-elements have sub-elements of their own.

- Click on the ***frame*** sub-element, then on the Add new element list box. You will see that it is empty: that is, you can not add any sub-elements to ***frame***.
- Do the same for the ***textformat*** and ***tab*** sub-elements. They do not have any sub-elements, either.
- Click on the ***view*** sub-element, then on the Add new element list box. As you can see, there are many available sub-elements for ***view***. A view presents a set of features like maps, graphs, and information boxes, so the list of available sub-elements allows you to specify the features to include in that view.

There is a similar list box at the bottom of the Attribute panel, showing the available attributes for the element highlighted on the left. You can see how the list boxes work by adding an attribute to ***project***.

- Click on the ***project*** element in the Element panel.
- Click on the Add new attribute list box. You will see a list of the following attributes: *title*, *lastSaved*, *version*. The first two appear automatically. Note that there is now a date and time next to *lastSaved*: it is the date and time you clicked on the Save and Edit button.

You have the option to add the version attribute.

- Click on *version* from the Add new attribute list box. The attribute *version* appears in the Tag column.

For now, you don't need it. To remove an attribute, click on it to highlight it, then click on Remove Attribute.

- Click on *version* if it is not already highlighted, then on Remove Attribute.

You follow the same procedure to add and remove elements from the element panel.

- From the Add new element list box, click on ***tab*** to add it to the element panel
- Click on Remove to remove it.

Editing a Project

When you start a new project, it is created with default elements and attributes. You can customize the elements to fit your own ideas. You will practice this by editing the view title, the words that appear on the project tabs. As you saw, the current *title* for ***view*** is "My View," but that is not very descriptive. Follow these instructions to replace it with the *title* "Map View".

- Click on ***view*** in the Element Panel to highlight it (if the only element you see is ***project***, then click on the arrow to the left to display the ***project*** sub-elements, then click on ***view***).
- In the Attribute panel, you should see the tag *title*, with the value "My View".
- Click on "My View" to highlight it. The value becomes editable text, which you can change to "Map View".
- Click anywhere outside the text to remove the highlight.
- Click on the Save & Preview button. The *title* for ***view*** appears as "Map View".

What if you wanted a more descriptive title? Something like After the Greenwich Tea Party?

- Click on the Return to Editing button.
- Click on ***view*** in the Element panel, and edit the *title* to "After the Greenwich Tea Party"
- Click on Save & Preview to see the changes.

[Insert screen shot]

Oh dear. That didn't work out very well. Most of the new title was chopped off. Why?

- Click on the Return to Editing button to investigate the problem.

The reason why the new title doesn't appear is that the tab is too small to hold it. The ***tab*** element is created with default settings. You can see the settings by looking at the ***tab*** attributes.

- Click on ***tab*** in the Element panel to highlight it, and then look at its attributes in the Attribute panel. Right now, none appear.
- But if you look at the Information panel, you can see that it has a number of pre-set attributes.

These include *hgt*, for setting the height in pixels, *offCol*, for setting the color of a tab that has not been clicked on, and *onCol*, for setting the color of a tab that has been clicked on. (Tabs change color to indicate the *active view*, that is, the view displayed "on top" of the screen.) And among the other attributes, you will find *wid*, for setting the width of the tab in pixels.

- Look at the default (specified as *def*). It is set at 100, meaning the tab's default setting is 100 pixels wide.

To change the width of the tab, you will specify the attribute *wid*, with a value of "200" pixels.

- Click on the Add new attribute list box, and scroll down until you see *wid*.
- Click on *wid* to add it to the Attribute panel. The default value is "100".
- Click on "100" in the value column to select it, and edit it to "200".
- Click on Save & Preview. Now the title displays properly.
- Click on Return to Editing to get back to the VisEdit screen.

Saving a Project with a New Name

Your project is saved whenever you click on Save & Preview, but the file is overwritten every time you save it. There is no "undo" button, so if you make changes, but don't like them, you will have to painstakingly undo them step by step. Sometimes you may not realize there is a problem until you have made many changes, which can make the "undo" steps very time-consuming.

One solution to this is to save your project a number of times along the way, with names that will help you keep track of each stage. For this manual, you will be saving your work with a new name for each

chapter. That way, if you make a mistake, you can always go back to the project as it stood at the end of the previous chapter.

You will end this chapter by saving your project with the name Chapter 1.

- Click on File from the menu bar, then click on Save as… from the menu. A text box appears with the instructions, Type new title to save project as…
- Type Chapter 1, and click on OK.
- Click on ***project*** from the Element panel if it is not already highlighted. In the Attribute panel, the title has been changed to Chapter 1.

Your previous project, My Project, still exists.

- Click on File, then on Load, to see a list of your projects displayed in the Information box. You should see both My Project and Chapter 1.

Congratulations! You have learned how to work with VisualEyes and created your first project. You have learned how to create and edit elements and attributes using VisEdit, and how to Save & Preview your project. You have also learned how to save your project with a new name to protect your change.

Chapter Two

> This chapter is in the process of being rewritten to reflect changes in VisualEyes.

In this chapter, you will learn how to add components to your VisualEyes project, including how to:

1. Load a file
2. Create a new view
3. Delete a view
4. Add a map
5. Understand external resources
6. Understand **GLUE**
7. Add components to a view
8. Add zoom control
9. Use VisEdit to modify components

10. Add a timeline

Loading a file

Once you have created and saved a project, it remains available for you to load and resume work on it. Files you create in VisualEyes are not stored on your personal computer, but rather on a *server*, a computer based at the University of Virginia. You access the files through your browser. [You will learn methods for keeping back-up copies of your work on your own computer in a later chapter.]

- Point your browser to the VisEdit screen and log in, as you did for the last chapter.

Your project should be loaded exactly as you left it at the end of the last chapter. All your elements should be available in the Element panel, and at the right of the menu bar you should see your project information: SAMPLES [or the username you specified]| Chapter 1, followed by the five-digit *ID number* that is a unique identifier for that project

As mentioned in the last chapter, it is good practice to save your project frequently in manageable sections, using an easy-to-remember name for each section. You will therefore save the project now, with the name Chapter 2. That way you will retain our Chapter 1 file as a backup, in case you need it.

- Click on File, then Save as…, and save the project with the name Chapter 2. The file is saved and the new file name appears on the right of the menu bar.

What if you wanted to reload your Chapter 1 file?

- Click on File, then Load project. A list of projects appears in the Information panel.
- Click on the ID number for the Chapter 1 project. Chapter 1 is re-loaded.

For this chapter, you will be working with the Chapter 2 file, retaining the Chapter 1 file as a backup.

- Click on File, then Load project, and click on the ID number for the Chapter 2 file to load it.

Using Wizards

VisEdit comes with a number of built-in *wizards* to help you carry out common tasks in VisualEyes. Wizards automate specific tasks, like creating views and adding map resources.

NEW CONTENT NEEDED HERE

Deleting a view

The easiest way to delete a view is from the main VisEdit screen. **Be careful, though! Deleting a view will delete all the resources, data, and displays associated with it!**

Since you have not created any features for "Third View", you can safely delete it.

- Click on VIEW: Third View in the Element panel to highlight it.
- Click on the Remove button at the bottom of the panel. The view is removed.

Adding a Map

Now that you have set up your views, it is time to add features to begin to make them work. First, you will use the Wizards feature to add a map of New Jersey to your view. VisualEyes considers maps to be *image resources*, so you access the map wizard by clicking on Resources from the Wizard chart panel.

- Click on Wizard View to access the Wizard screen.
- Click on Resources under View 1 ("After the Greenwich Tea Party"). Resource wizards for View 1 are displayed in the panel on the left.
- Click on Add image resource and **GLUE**. The Image resource wizard is displayed.
- [Insert screen shot]

Understanding External Resources

In order to add a map to a view, you will find it helpful to understand how VisualEyes works with *external resources*, that is, resources held in files created outside the software. These can be images, documents, or numbers, and they can be in a variety of file formats, such as .jpg

(a kind of image file) or spreadsheets. To work with these files, you must first make sure that they are available online, since VisualEyes can't read them on your computer. Later on you will discuss ways of putting your own files online so VisualEyes can find them. For this section, though, you will use an image file that is already online, a map of southern New Jersey from the early 19th century. You'll need its full URL: http://burkeandhare.com/NJsouth.jpg .

Understanding GLUE

VisualEyes uses a specific kind of script to identify external resources and add them to projects. This kind of script is called ***GLUE***, short for General Language to Unite Events. **GLUE** appears in VisEdit as an element, and it is the heart of making interactive visualizations. **GLUE** elements can do a wide range of things. Their most common functions are:

1. To cause resources, such as images, paths, and charts, to show up on the screen, automatically or on command.
   # To connect the data resources to viewers, through features such as display tables, popup windows, charts, and data-driven maps.

For now, you don't need the technical details of how **GLUE** works. All you need to know is the basic idea behind it: You first identify the map you want to display by giving it a unique ID, and then you instruct VisualEyes to **GLUE** it to the ***view*** in which you want it to appear. Note that you do not glue a map to an entire ***project***, but only to a specific ***view***. That means that if you want to use it in more than one view, you must identify it and *glue* it to each one separately.

Fortunately, using the Add image resources wizard makes this task simple. Start by adding the map of southern England specified above to View 1. You should be at Step 1 of the wizard, Add resource.

- Click on 2.Type ID. A text box appears, prompting you for the ID for your map resource. The ID should be a name that you will remember, and that you only use for this map. As with other wizards, the script will appear in the box on the bottom right.
- Type mapNJ.
- Click on 3.Type URL, and enter the URL: http://burkeandhare.com/NJsouth.jpg (you may have to delete the existing prompt, http://)

- Click on 4. Add **GLUE**. There is nothing for you to enter here: the wizard will add the **GLUE**.
- Click 5. Show at start?

Images can be displayed either when the program is first loaded, or else when triggered by a particular event, like clicking a button or reaching a certain date on the timeline. In this case, you want the map to be displayed when the program is loaded, so you can leave the default setting, which is "true." This means "Yes, I do want the map to show up when the program is first started."

- Click on 6. Done. You have finished the wizard.
- Click the Add! You are returned to the main Wizards screen.

Take a few minutes to see what the changes look like in the main VisEdit screen.

- Click on Main Tree View to return to the main VisEdit screen. Note that there is now a folder icon next to your first ***view***, and a small right-pointing arrow. This indicates that the view has sub-elements.
- Click on the arrow to see the sub-elements, ***image*** and **GLUE**. Note that ***image*** has the attributes *src*, *id*, and *type*. The value for *src* is the URL you specified, http://burkeandhare.com/NJsouth.jpg , and the value for *id* is "mapNJ". The attributes for GLUE are *init*, with a value of "true", and *from*, with a value of "mapNJ".
- Click on Save & Preview to check the results. The map appears on the "After the Greenwich Tea Party" tab.
- Click Return to editing to return to the main VisEdit screen.

Adding a Zoom Control

Now that you have a map of southern New Jersey, wouldn't it be useful to be able to zoom into it? You can add a zoom control by using a View wizard.

- Click on Wizards View to display the Wizard screen.
- Click on the View oval for After the Greenwich Tea Party to display available View wizards.
- Click on Add a zoom control from the list on the left. The Zoom control wizard is displayed. You are positioned at the first step.
- Click on 2.Type left. A text box appears, prompting you for the position, in *pixels*, of the left edge of the zoom control,

calculated from the left edge of the project frame. The default value is "30". Since that is a good place for the zoom control, leave the number unchanged.

- Click on 3.Type top. This text box prompts you for the position, in pixels, of the top edge of the zoom control, calculated from the top of the project frame. Again, the default value is "30". Leave it unchanged.
- Click on 3.Done, then on the Add! button.
- Click on Save & Preview. The zoom control appears at the upper left corner of the map
- Click on Return to editing.

Adding a Timeline

The final feature you will add to your view is a timeline. It is one of the most important controls in VisualEyes, allowing you to create dynamic presentation of material linked to specific dates. Again, you will be using a View wizard.

- Click on Wizards View to display the Wizard screen.
- Click on the View oval for "After the Greenwich Tea Party" to display available View wizards.
- Click on Add a Timeline. The Timeline wizard is displayed. As always, you are at the first step.
- Click on 2. Type start. The Type a starting date text box is displayed.
- Enter 12/22/1774

This is the format for December 22, 1774, the date of the Greenwich Tea Party, considered the first conflict between patriots and British authorities in New Jersey.

- Click on 3. Type end. The Type an ending date text box is displayed.
- Enter 04/02/1783

This is the format for April 2, 1783, when Captain John Stewart shot the notorious Loyalist Captain John Bacon, near Tuckerton, considered the last skirmish of the war in New Jersey.

- Click on 4. Show dates? The default it "true," which means the dates will be shown, so leave it as is.
- Click on 5. Tick length. This is the length, in pixels, of the tick marks for the timeline. The default is 8 pixels, so leave it as is.

- Click on 6. Add frame. This allows you to position the timeline. The default is to insert a frame, so leave it as is.
- Click on 7. Type width. The default is 600 pixels, so leave it as is.
- Click on 8. Type height. The default is 100 pixels, so leave it as is.
- Click on 9. Type left position. The default is 100 pixels, so leave it as is.
- Click on 10. Type top position. The default is 550 pixels, so leave it as is.
- Click on 11. Pick bar color. The default is gray, but you are going to change it to blue.
- Click on the gray box. A palette will appear.
- Click on any of the blue boxes. The bar color is now blue.
- Click on 12. Done, then on the Add! button.
- Click on Save & Preview to view the project.

Well, the timeline is there, but the wizard did not do exactly what you'd expected. Only the years are showing, not the days or months!

Modifying a feature

- Click on Return to Editing.

To make the dates display properly, you must add an attribute called *dateFormat* to the timeline.

- Click on the arrow next to VIEW: After the Greenwich Tea Party, to display the sub-elements.
- Click on the sub-element ***timeline***. You can see all the attributes you created with the wizard: *majorTick*, *showValues*, *max*, and *min* in the Attribute panel.
- Click on the Add new attribute list box, and then click on *dateFormat*. The attribute is added.

[Insert screen shot]

The default value is listed as "yr", which explains why timeline only displayed the year 1786.

- Click on the list box under Choices to see the other possibilities.
- Click on the last choice, "mo,dy,yr". The value changes to "mo,dy,yr".
- Click on Save & Preview.

That's better! Now the dates are properly displayed in month, day, and year format.

- Click on Return to Editing. You have completed Chapter Two.

Congratulations! You have successfully used wizards to add a map, zoom control, and timeline to your view. You have also learned how to navigate between wizards and the main VisEdit screen. Finally, you have been introduced to glue and scripts. You will learn more about all these features in the next chapter.

Chapter Three

In this chapter, you will learn how to create paths to display events that change over time and space:

- How to:
  - Understand paths and ***dot***s
  - Understand screen redraw
  - Create a simple ***path*** with the ***Path*** wizard
  - Positioning ***dot***s using x and y coordinates
  - Edit a ***path*** in VisEdit
  - Understand the time attribute
  - Understand the date attribute
  - Use Copy and Paste in VisEdit

Understanding Paths and *Dot*s

In this chapter, you will learn how to create a *path*, constructed from ***dot***s that are linked to a specific point on the timeline, and on the map. The ***path*** allows you to display Thomas Jefferson's route as he traveled through southern England in the spring of 1786.
In order to understand how paths work, you will start by taking a closer look at the Jefferson's Travels visualization.

- Point your browser to http://www.viseyes.org/show/?base=jt. After a few moments, the Jefferson's Travels visualization will load.
- Use the timeline slider to position the timeline on April 8, 1786.

As you can see, VisualEyes has drawn an orange line on the map, connected by icons in the shape of buildings. This is a very easy and intuitive way to display the route Jefferson traveled during this period, with the icons representing his stops along the way. The VisualEyes

element that defines the route is called the *path*. It is drawn by defining a set of ***dot****s*, indicated by the icons.

- Move the slider to April 30, 1786.

Again, it is very easy to understand the display. The orange ***path*** represents Jefferson's travels from the time he arrived in England until he left two months later. The ***dot***s show his stops along the way. What's going on behind the scenes to make this work?

Understanding Screen Redraw

VisualEyes projects are highly interactive. For that reason, the screen constantly needs to be redrawn to reflect changes implemented by the user. Each time you click on the screen and drag the map, scroll the timeline, or move the zoom control, you send a command to VisualEyes to adjust itself to your response. This is called a *screen redraw*, and it is one of the most valuable features of VisualEyes.
Some of the instructions for the screen redraw are built into the program. Others you can add yourself. For example, when you added the map to your view, you specified that one of the attributes for the GLUE element, *init*, was set to a value of "true". The GLUE tells VisualEyes to load the map as soon as the project loaded, rather than waiting for a command from the user.
When you create paths and ***dot***s to represent change over time, you create a different kind of GLUE element, one that links a specific kind of display – a line and/or a ***dot*** – with a specific date on the timeline. Pressing the Play button sets the timeline in motion. Behind the scenes, the GLUE goes to work. It's as though the **GLUE** was following a set of instructions: "Attention VisualEyes! When the timeline reaches the specified point, draw the ***path*** and/or ***dot*** on the screen." The instructions continue – for as many paths and/or ***dot***s as you've specified – until the timeline reaches the end.
We'll discuss some of the technical aspects of timeline control as we go along. For now, you will use a combination of wizards and VisEdit to create a simple ***path*** showing some of the movements of American troops in New Jersey during the Revolutionary War.
Start out by loading VisEdit for your project.

- Point your browser to the VisEdit screen and log in.
- Click on File, then Save as…, and save the project with the name Chapter 3.

Creating a Simple *Path* with the *Path* Wizard

There are several different ways to provide VisualEyes with the information needed to draw a path. You will start out with the simplest, the ***Path*** wizard.

- Click on Wizards View.
- Click on Displays, under View: After the Greenwich Tea Party. Available wizards for Displays are listed on the left.
- Click on Add a Path. The ***Path*** wizard is displayed, and you are set at Step 1.

[Insert screen shot]

- Click on 2.Add ID. You are prompted for a unique ID to use for this path.
- Enter pathPatriots.
- Click on 3. Pick line color, and leave the default, red.
- Click on 4. Type line width, and leave the default, 4 pixels.
- Click on 5. Tween lines? This is asking whether the ***path*** should draw lines between the ***dot***s. The default is "true," so leave it as is.
- Click on 6. Add ***dot***. This does not require any input.
- Click on 7. ***Dot*** style, and click on the list box to see the possible icon types. The default is bar; you can leave it as is.
- Click on 8. ***Dot*** size, and leave the default, 20 pixels.
- Click on 9. Add ***dot***, to add a second ***dot***. You do not have to define the style or size; they use the same definition as the first ***dot***.
- Click on 10. Add **GLUE**. Paths, like maps, are first defined, then "glued" to the view.
- Click on 11. Done, then on Add!.
- Click on Save & Preview. You can see the first ***dot*** in the upper left corner of the map.
- Press the play button. As the timeline moves, the ***path*** will extend from the first ***dot*** to the second.

This ***path*** may work perfectly, but it is not very useful. The ***path*** starts somewhere in Pennsylvania and ends before it ever reaches New Jersey. How can you edit the ***path*** so it follows the actual route of Revolutionary War battles?

Positioning *Dot*s using X, Y Coordinates

In order to display actual places on the map, you must specify the position of the ***dot***s on the map. You do this by specifying the x and y

coordinates for the map location. The map, like other images, is made up of pixels, and each pixel has a unique x (horizontal) and y (vertical) location.
When you created your ***dot***s with the ***path*** wizard, the first was given the default x, y setting of (100,100), corresponding to the point 100 pixels counting from the left edge (the x coordinate) and 100 pixels down from the top edge (the y coordinate). The second ***dot*** had the default x, y setting of 400,400, corresponding to the point 400 pixels from the left edge and 400 down from the top edge.
You can see the x,y coordinates for any point on the map by holding down the Alt key while clicking on the point. The coordinates will appear in the lower right of the project, just under the play button of the timeline.

[Insert screen shot]

Use this technique to view the coordinates for ***dot***s you've created.

- You should still be previewing the project. While holding down the Alt button, click on the first ***dot***. In the lower right, you will see something close to Screen 98,97.

The bar is 20 pixels wide and is centered on the specified point, so the coordinate values will be somewhere between 90 and 110.
Now, do the same for the second ***dot***.

- Hold down Alt while clicking on the second ***dot***. In the lower right, you will see something close to Screen 398, 405.

Again, the actual values will range between 390 and 410, because the bar is 20 pixels wide.
To edit the ***dot*** information to match the actual movements of the troops during the war, you will modify their coordinates to match those for Greenwich – site of the Greenwich Tea Party, considered the first action by New Jersey patriots against the British – and for Cape May, site of the casualty on New Jersey soil.

- Find Greenwich on the map. It is on the Delaware River in Cumberland County. You may have to drag the map and zoom in to see it. Hold down Alt and click on the city. The coordinates displayed should be close to 565, 790.
- Now, find Cape May on the map, on the southern tip of New Jersey. Hold down Alt and click on it. The coordinates displayed should be close to 890, 1210.

Note these coordinates. The first is the *x* coordinate, and the second is the *y* coordinate. You will need them to modify the path.

- Click on Return to Editing.

Editing a *Path* in VisEdit

As you saw in the last chapter, you can use VisEdit to modify a component you have created with a wizard.

- Click on the arrow next to PATH:pathPatriots (in the Element panel) to display the two ***dot***s.
- Click on the first ***dot*** in the Element panel. Its attributes are displayed in the Attribute panel.

Note that the values for *x* and *y* are each "100". You want to change that to the coordinates for Greenwich

- Edit the *x* and *y* values: *x*="565" and *y*="790". **Warning! The *x* value will be underneath the *y* value: make sure you enter the values correctly.**

Since you will be creating a number of ***dot***s, you'll find it helpful to identify them. Give this ***dot*** the *id* "Greenwich".

- Click on Add new attribute, scroll down until you see *id*, then click on it.
- Enter the *id* value "Greenwich".
- Click on the second ***dot***. Its attributes appear in the Attribute panel. Note that the *id* "Greenwich" now appears next to the first ***dot*** in the element panel.

Now, edit the coordinates, and create an *id*, for the second ***dot***.

- Edit the *x* and *y* values: *x*= "1530" and *y*= "940". **Warning! Remember that the *x* value will be underneath the *y* value: make sure you enter the values correctly.**
- Create an *id* attribute, and enter the *id* "Cape May".
- Click on any element. The *id* "Cape May" appears next to the second ***dot*** in the Element panel.
- Click on Save & Preview.
- Click on the Play button.

This is better, but still not perfect. The ***path*** draws correctly, but the section of the map with the first ***dot*** is hidden by the frame.

Much better! The ***dot***s appear correctly on the map.
The location of the ***dot***s is now correct, but their timing is not. The Greenwich Tea Party took place on December 22, 1774, but the skirmish at Cape May took place on June 29, 1776, not in 1783. You can correct this with a little additional editing.

- Click on Return to Editing to return to VisEdit.

Understanding the Time Attribute

- Click on the Greenwich ***dot*** in the Element panel to display its attributes. Note that it has an attribute, *time*, that has a value of "0" (zero).
- Click on the Cape May ***dot***. Note that it, too, has the attribute *time*, with a value of "1".

When you create a ***path*** using the ***Path*** wizard, the default setting displays the first ***dot*** when the project is loaded, and the second ***dot*** when the timeline reaches the end. That is, the default **GLUE** for the ***path*** is something like: "Attention VisualEyes! When the timeline is at the very beginning, display the first ***dot***, and begin drawing the ***path*** line. When it reaches the very end, complete the ***path*** line and display the second ***dot***."
This instruction is translated into language VisualEyes can understand through the *time* attribute. When defining a ***dot***, giving the *time* attribute the value of "0" (zero) will display the ***dot*** when the timeline is at the beginning. Giving the *time* attribute the value of "1" will display a ***dot*** when the timeline reaches the end.

Understanding the Date Attribute

You can make the ***path*** more useful for displaying time-dependent data by linking the ***dot***s to specific dates, rather than to the beginning and end of the timeline. You do this by using the *date* attribute, which links the ***dot*** to a specific date on the timeline.
First, remove the *time* attribute.

- Click on the Greenwich ***dot*** in the Element panel.
- Click on *time*, then on Remove attribute. The attribute is removed.
- Click on the Cape May ***dot***, then on the *time* attribute.

- Click on Remove attribute to remove it.

Next, add the date attribute.

- Click on the Greenwich ***dot***.
- Click on Add new attribute, then on *date*.

The values for the date attribute should follow the same format as you used for the timeline. The Greenwich Tea Party occurred on December 22, 1774, so you enter the date as 12/22/1774.

- Click on the Value column for *date*, and enter "12/22/1774".

The Cape May skirmish, in which Richard Wickes was killed, thus becoming the first casualty on New Jersey soil, took place on June 29, 1776, so you can add that date to the Cape May ***dot***'s attributes.

- Click on the Cape May ***dot***.
- Add the attribute *date*.
- Click on the Value column for *date*, and enter 6/29/1776.
- Save & Preview
- Click the Play button. The ***path*** should display correctly.
- Click on Return to Editing.

Using Copy and Paste in VisEdit

So far, so good. You have defined the ***path*** for New Jersey battles and defined the first two stops along the way. But how can you add more? The easiest way to add more ***dot***s is simply to copy and paste the one you have. Once the first ***dot*** is defined, all subsequent ***dot***s have the same attributes unless you choose to change them. You can make copies of your second (Cape May) ***dot***, then edit the *id* and coordinates for each.
For this project, you need a total of 8 ***dot***s to display the most memorable battles and skirmishes of southern New Jersey. You'll do this by first, creating 3 copies of the Cape May ***dot***, and then modifying them to correspond to the actual locations of the hostilities.

- Click on the Cape May ***dot*** in the Element panel.
- Click on Edit from the menu bar, then on Copy.
- Click on Edit, then on Paste. The ***dot*** is copied.

- Follow the same steps until you have a total of 8 ***dot***s, one for Greenwich and 7 for Cape May.

- Click on the third ***dot*** (the second Cape May ***dot***).
- Edit the attributes so that the *id*= "Egg Harbor", *x*= "1140" and *y*= "890", *date*= "4/26/1777".
- Edit the rest of the ***dot***s with the data from the following table:

id	*x*	*y*	*date*
Greenwich (already created)	565	790	12/22/1774
Cape May (already created)	1530	940	6/29/1776
Egg Harbor (already created)	1140	890	4/26/1777
Chestnut Neck	1180	750	10/6/1778
Absecon	1240	860	10/1/1779
Tom's River	1350	320	5/14/1780
Barnegat Beach	1365	450	10/26/1782
Tuckerton	1290	670	4/3/1783

When the ***dot***s are created, you are ready to view the project.

- Click on Save & Preview, and then on the Play button. The ***path*** should follow the military action in southern New Jersey during the Revolutionary War.
- Click on Return to Editing.

Congratulations! You have successfully created and edited your path.

Chapter Four

In this chapter, you will learn how to enhance the ways viewers can interact with your maps and data. You will learn how to add a magnifying glass to zoom into your map, and how to add a control panel to your view. You will also learn advanced techniques for working with data files created with spreadsheets.

You will learn how to:

1. Understand widgets
2. Add a Magnifier
3. Add a Control Panel
4. Add an element to the Control Panel
5. Modify the Control Panel
6. Link a ***path*** to the Control Panel
7. Work with external data files
8. Understand data files

9. Import external data files into VisualEyes
10. Add an external data file to a view
11. Fill a ***path*** using ***Dot***fill

Understanding Widgets

VisualEyes allows you to add a number of *widgets* to your project. A widget is specialized tool that performs a particular task. They are highly visual and interactive, and can make the project more interesting for the user.

To see how they work, you will modify your second view to display a map of the entire length of New Jersey for the Revolutionary War period. You will then create a widget called a Magnifier, which can be dragged across the map.

As usual, start out by loading VisEdit for your project.

- Point your browser to the VisEdit screen and log in.
- Click on File, then Save as…, and save the project with the name Chapter 4.

Rename the Second View tab to All New Jersey Battles.

- Click on Second View in the Element panel, and rename the *title* attribute to "All New Jersey Battles".
- Click on Save & Preview to view the new tab. Then, click on Return to Editing.

The next step is to add the map. For this view, you'll use the map located at http://burkeandhare.com/njFullx.jpg|http://burkeandhare.com/njFullx.jpg.

- From the Wizards panel, click on Add Image, then on the Next step button
- Add the *id* "mapNJfull", then click on Next step
- Add the URL: http://burkeandhare.com/njFullx.jpg, then click on Next step
- Click on Next step again to add the **GLUE**
- Click on Next step again to choose the default *init*= "true", which enables the map to show up when the project is loaded.
- Click on Next step, then Add, to add the map. It will appear in the view as IMAGE:mapNJfull.

The **GLUE** also appears in the view. Since you'll be using **GLUE** elements a number of times in this view, you'll find it helpful to give it an *id*.

- Click on **GLUE** in the Element panel.
- From the Add new attributes list box, select *id*.
- Enter "showMapNJfull", then click on **GLUE** again. The *id* will appear.
- Click on Save & Preview to see your project.
- Click on the All New Jersey tab to see the map.

When the view loads, the northern half of the state of New Jersey is displayed.
[insert screen shot]

You can drag the map to see the southern part of the state.

- Practice dragging the map to see the northern and southern sections. When done, click Return to editing.

Adding a Magnifier

A Magnifier is a visual tool, shaped like a magnifying glass, for zooming into a section of an image. It makes working with images more interesting and interactive. The easiest way to add a magnifier is to use the wizard.

- Click on VIEW: All New Jersey Battles if it is not already highlighted.
- From the Wizards panel, scroll down until you see Add Magnifier.
- Click on Add Magnifier to begin the wizard. The Add Magnifier wizard appears

[insert screen shot]

- The first prompt is "Add magnifier glass widget". Click on Next step to continue.
- You are prompted for the magnifer *id*. The default is "myMagnifier", so it will appear when you click on the text box. Click on Next step to accept the *id* and continue.
- You are prompted for the URL of the image. In this case, you want to magnify the image you are already using, so enter http://burkeandhare.com/njFullx.jpg and click on Next step.

Note that the Magnifier widget does not actually "magnify" the image. Instead, it replaces the original image with a larger one, so as to create the illusion of a magnifying glass. It is as though you have created a window from the original map through to the related section of a larger map underneath. The Magnifier can be used in other ways, for example, to replace a modern map of a region with a historical map, or a black and white image with a related one in color.
The next step prompts you to choose the initial zoom factor from a list.

- Enter 2 and click on Next step.

The next set of prompts allow you to modify the appearance of the magnifying glass, but for now, you will keep the defaults.

- Click on Next step through step 12.
- When the Done prompt appears, click on the Add button. The Magnifier widget will be added.

You can see the Magnifier widget in the Element panel.

- Click on the arrow by VIEW:All New Jersey to see the elements beneath it. You can see that the Magnifier widget has been added.
- Click on the **GLUE** element underneath it. You'll want to give it a name.
- Give the **GLUE** the *id* "showMagnifier".
- Click on Save & Preview, then on the All New Jersey tab. The Magnifier appears.
- Practice dragging the Magnifier around the map.

You'll see that it will only magnify the top part of the map: if you drag the map so that the southern part of New Jersey is displayed, the magnifying glass continues to display the zoomed-in image of the northern part of the state. Again, that's because the Magnifier creates a window from the original image through to the zoomed-in one you specified. Though the original image can be dragged around the screen, the zoomed-in one (visible in the magnifying glass) cannot.

- Click on Return to editing.

Adding a Control Panel

For the next part of the project, we will add a ***path*** with New Jersey battles. We would like to use the same map of the entire state. One

option is to create a new view and use the same map for the background. There is, though, a more efficient way to use our resources: we can create a *control panel* to organize our material, and present different types of information within the same view.
You can create a control panel with a wizard.

- Click on VIEW:All New Jersey if it is not already highlighted.
- From the Wizard panel, click on Add control panel.
- Follow the steps of the wizard in the usual way, clicking on Next Step to select the defaults.
- Specify the *title* as New Jersey Map.
- At the Done prompt, click the Add button. The control panel is created.
- Click on Save & Preview, and on the All New Jersey tab to see it.

[insert screen shot]

Your control panel has been created, but it does not yet do anything. Now we'll add the attributes to make it work.

- Click on Return to editing to return to the Tree view, and click on CONTROLPANEL in the Element panel to show its related elements and attributes.

The control panel wizard creates the structure for the control panel, but you have to add *element* elements to create the actual controls. For this project, you will create two check boxes. These boxes work as simple on/off switches. When checked, they are on, and they display the specified information. When unchecked, they hide the specified information.

Adding an element to a control panel

The first element you add will control whether or not the Magnifier is displayed.

- Make sure CONTROLPANEL is highlighted in the Element panel, and click on the Add new element list box to display the options.
- Click on element. ELEMENT appears in the Element panel.
- In the Attribute panel, click on *title*, and enter "Show Magnifier".
- Click on *type*, and select "checkbox" from the list box.

The two attributes you just added control the appearance of the element. It will be displayed as a check box, with the title Show

Magnifier. The next attribute you add will control what the check box actually does. It is a **GLUE** attribute, which will run a specified **GLUE** script. The script you want to run is the **GLUE** you've already created: showMagnifier, the **GLUE** that tells VisualEyes to display the Magnifier.

- From the Add new attribute list box, select **GLUE**. The **GLUE** attribute provides a list box with the **GLUE** scripts you have created.
- Select showMagnifier.

The **GLUE** attribute you have just added tells VisualEyes to display the Magnifier when the check box in the control panel is checked. For it to work properly, you will have to adjust the showMagnifier **GLUE** slightly.

- From the Element panel, click on **GLUE**:showMagnifier to display its attributes.

In the Attributes panel, you can see that *init* is set to "true". That means that the **GLUE** will automatically run when the project is loaded. You want to modify it so that it will only run when the check box in the control panel is checked.

- Click on *init* in the Attribute panel, then on the list box in the Choices column.
- Select "false".

Now the "showMagnifier" **GLUE** will not run automatically. Instead, it will only run when the **GLUE** is called, that is, when the checkbox in the control panel is checked.

- Click on Save & Preview, and then on the All New Jersey tab. As you can see, the Magnifier no longer appears in the view.
- Click on the Show Magnifier check box in the control panel. The Magnifier appears.

Modifying the Control Panel

The control panel does what it is supposed to, but it could look a little bit prettier. Why not change the opacity, so that you can see the map beneath?

- Click on Return to editing, then on FRAME under CONTROLPANEL in the Element panel.

The attribute that controls the opacity of the control panel is *alpha*. When it is set to "100", the feature is fully opaque. Try changing it to "75".

- Click on the *alpha* attribute, and change the value to "75".
- Click on Save & Preview, then on the All New Jersey tab to see the change. The control panel is partially transparent.

There are other ways you can modify a Control Panel.You can adjust the appearance of the control panel by changing the color of the background and text, and its height and width. You can also adjust where it appears in the view. You can see the attributes that control its overall appearance by selecting FRAME in the Element panel. You can see the attributes that control the font size and appearance by selecting TEXTFORMAT in the Element panel.

Congratulations! You have successfully added a widget to your project!

- Click on File, then on Save to save your work.

Adding a *Path* to the Control Panel

We can now move on to the next part of our project: adding a ***path*** of Revolutionary War battles in New Jersey. You already know the basic steps for adding a path. You first create it and define its attributes, then add ***dot***s and define their attributes. Finally, you create a **GLUE** to show the path. You can have the ***path*** display when you click a checkbox on the control panel by linking to its **GLUE**, just as you did for the Magnifier.
Begin by defining the ***path*** itself.

- Click on VIEW:All New Jersey Battles if it is not already highlighted.
- From the Wizard panel, select Add Path.
- Follow the steps for the wizard, giving the ***path*** the *id* "pathBattles" (step 2).
- You can keep the default for the line color.
- When prompted for the width (*wid*), enter "0".
- When prompted for Tween lines between ***dot***s, select "false" (step 5). That means that you do not want lines to be drawn between the ***dot***s.
- Keep the default for the first ***dot*** (step 6). You'll change it later.
- Select "star" for the ***dot*** *style* (step 7)
- Enter "40" when prompted for the ***dot*** *size*.

- Add the second ***dot*** and the **GLUE**, and click on Add.

You'll want to preview the ***path*** to make sure it's working right. But first, make a few changes to make the ***dot***s easier to see.

- Edit the first ***dot***'s attributes so that *x*= "500" and *y*= "500".
- Remove the *time* attribute.

The *time* attribute links the ***dot*** to a specific point on the timeline. Since you are not working with a timeline, you don't need it right now.

- Edit the second ***dot***'s attributes so that *x*= "800" and *y*= "800".
- Remove the *time* attribute.
- Save & Preview. The two ***dot***s should appear as you specified them.
- Return to editing.

The next step is to give the path's **GLUE** a name, so that you can call it from the control panel checkbox.

- Click on **GLUE** if it is not already highlighted.
- From the Attribute panel, give it the *id* "showPath".
- Select *init* from the Attribute panel, and select "false". That means that the ***path*** won't show up when the project is initially loaded. Instead, it will wait until it is called – in this case, by checking on a checkbox in the control panel.

You have now set up the ***PATH*** "pathBattles", and the **GLUE** "showPath" that will allow that ***path*** to be displayed. The next step is to create a checkbox on the control panel.

- Click on CONTROLPANEL from the Element panel, and select element from the Add new element list box.
- Enter "Show Battles" for the *title*, and select "checkbox" for the *type*.
- Select **GLUE** from the Add new attribute list box, and choose "showPath".
- Save & Preview. The ***path*** will appear when you click the Show Battles checkbox.
- Return to editing.

You have used the ***dot***s to make sure that your ***path*** displays properly, but you won't need them again.

- Remove the two ***DOT***s from the Element panel. Be sure not to remove the ***PATH*** or **GLUE**!

Working with External Data Files

Last time you created a path, you added each ***dot*** individually. This time, we will use an *external data file*, together with the *fill****dot*** script, to add the ***dot***s to the path. An external data file can be created in another program, like an Excel spreadsheet. This can be very helpful for projects in which an assistant enters the data. In order to make the best use of this feature, you'll find it helpful to understand how VisualEyes processes data files.

Understanding Data Files

In many software programs, data is organized into units, often called *records*. Each record contains certain pieces of information, called *variables*. A very familiar kind of data organized in this way is an address book. Each person in the address book is a separate record. Each piece of information about the person – name, address, cell number – is a variable. In many programs, including Excel spreadsheets, records and variables are organized into tables of rows and columns, as in the example below:

Name	Address	Cell
Smith, John	104 East Lansing St, Aberdeen, MD	401-345-8726
Roberts, Mary	43 South Omaha Ct, Briggs, CA	323-969-3871
Tucker, Bill	8 North Chestnut Ave, Mt Oliver, PA	412-123-7538
Damato, Jane	9372 West Broad St, East Troy, WI	262-456-0892

In this table, the names of the variables are "Name", "Address", and "Cell". The data is organized according to people, and each person is a separate record. Each person has distinct values for each variable, and those values are entered into the appropriate column for that variable.

When working with VisualEyes, you can organize the data for each ***dot*** in the same way. Each ***dot*** is a separate record, and ***dot***'s attributes are its variables. With that in mind, we can take another look at the data we entered for PATH:pathPatriots:

id	*x*	*y*	*date*
Greenwich	565	790	12/22/1774

Cape May	1530	940	6/29/1776
Egg Harbor	1140	890	4/26/1777
Chestnut Neck	1180	750	10/6/1778
Absecon	1240	860	10/1/1779
Tom's River	1350	320	5/14/1780
Barnegat Beach	1365	450	10/26/1782
Tuckerton	1290	670	4/3/1783

When you created each ***dot***, you created a record, like a row in an Excel spreadsheet. Each *attribute* – *id*, *x*, *y*, and *date* – is a variable, and the "value" you entered for each – for example, "Greenwich", "Cape Map", or "Egg Harbor" for *id*, and "565", "1530", or "1140" for *x* – provides information to VisualEyes on how to define the ***dot*** and locate it in time and space.
For the ***path*** you've just created, pathPatriots, you will be adding the following ***dot***s. Take a few moments to look at the data set in the table

id	***lab***	***x***	***y***	***style***	***wid***	***col***	***glue***	***year***	***mark***
Absecon	Absecon	1208	1885	star	40	0x660000	showBattles	1779	0
Amboy	Amboy	1200	829	star	40	0x660000	showBattles	1776	1
Barnegat	Barnegat	1325	1490	star	40	0x660000	showBattles	1782	2
Bordentown	Bordentown	925	1160	star	40	0x660000	showBattles	1778	3
Brigantine	Brigantine	1240	1798	star	40	0x660000	showBattles	1775	4
Cape May	Cape May	865	2205	star	40	0x660000	showBattles	1776	5
Camden	Camden	620	1320	star	40	0x660000	showBattles	1778	6
Cranberry	Cranberry	1093	988	star	40	0x660000	showBattles	1777	7
Delaware Bay	Delaware Bay	585	2090	star	40	0x660000	showBattles	1780	8
Egg Harbor	Egg Harbor	1070	1915	star	40	0x660000	showBattles	1779	9
Elizabethtown	Elizabethtown	1295	670	star	40	0x660000	showBattles	1779	10
Hackensack	Hackensack	1400	480	star	40	0x660000	showBattles	1777	11
Haddonfield	Haddonfield	645	1385	star	40	0x660000	showBattles	1778	12
Manasquan	Manasquan	1430	1220	star	40	0x660000	showBattles	1780	13
Monmouth	Monmouth	1260	1000	star	40	0x660000	showBattles	1778	14
Morristown	Morristown	1128	550	star	40	0x660000	showBattles	1777	15
Mt. Holly	Mt. Holly	857	1270	star	40	0x660000	showBattles	1776	16
New Brunswick	New Brunswick	1148	835	star	40	0x660000	showBattles	1777	17

Newark	Newark	1330	600	star	40	0x660000	showBattles	1780	18
Paramus	Paramus	1370	370	star	40	0x660000	showBattles	1780	19
Paulus Hook	Paulus Hook	1350	655	star	40	0x660000	showBattles	1779	20
Princeton	Princeton	960	960	star	40	0x660000	showBattles	1777	21
Red Bank	Red Bank	1417	980	star	40	0x660000	showBattles	1780	22
Sandy Hook	Sandy Hook	1450	870	star	40	0x660000	showBattles	1782	23
Shrewsbury	Shrewsbury	1155	1135	star	40	0x660000	showBattles	1779	24
Somerset	Somerset	990	760	star	40	0x660000	showBattles	1779	25
Spanktown	Spanktown	1244	730	star	40	0x660000	showBattles	1781	26
Toms River	Toms River	1347	1362	star	40	0x660000	showBattles	1778	27
Trenton	Trenton	880	1065	star	40	0x660000	showBattles	1776	28
Tuckerton	Tuckerton	1250	1693	star	40	0x660000	showBattles	1783	29
Woodbridge	Woodbridge	1245	765	star	40	0x660000	showBattles	1780	30

The information for each ***dot*** is displayed in a separate row, with the value for each variable in a separate column. Most of the variables provide VisualEyes with the attributes it needs to define each ***dot***, and you have seen many of them before. These are:

id The id for the ***dot***

x Location of the ***dot*** on the map, using the x,y coordinates

y Location of the ***dot*** on the map, using the x,y coordinates

style Symbol used to define the ***dot***. In the data table above, the value is "star"

wid Width of the ***dot*** in pixels. In the data set above, the value is "40"

col Color of the ***dot***. In the data set above, the value is "0x660000"

The data table also includes some new attributes for you to work with:

lab Label for the ***dot***. You can use this label to identify the ***dot*** to the viewer. It can be the same as, or different from, the id.

glue This *glue* allows you to link each ***dot*** to a VisualEyes feature or component. When the viewer interacts with the ***dot*** – by clicking on it, for example – the glue will "call" another part of the program. You'll use the *glue* attribute to connect each ***dot*** to an infobox in the next chapter.

The final two variables in the data table, *year* and *mark*, have no special meaning to VisualEyes. Instead, they add additional

information about each record, which will be helpful in displaying the data. You'll use them to display information in infoboxes in the next chapter.

year	The year that the battle took place. Note that we are NOT calling this variable *date*. That's because the attribute *date* has special meaning in VisualEyes: it connects each ***dot*** to a point on the timeline. Since we're not using a timeline for this part of the project, we don't want to confuse VisualEyes by creating a *date* attribute.
mark	A unique number that belongs to one, and only one, record. We'll discuss its purpose in the next chapter.

Importing External Data Files into VisualEyes

CONTENT TK

Adding an External Data File

Now that you understand the data, it is time to add it to the All New Jersey view as a ***resource***.

- Click on VIEW:All New Jersey if it is not already highlighted.
- Click on Add New Element from the Element panel, and select Resource.
- In the Attribute panel, enter *id* as "dataBattles"
- For *type*, select "xml"
- From the Add new attribute list box, select *src*. Enter the data file "/data/19362-njbattlesdata.xml"
- From the Add new attribute list box, select *preload*, and select "true". This ensures that the data loads when you load the view.

Filling a *Path* with a Data File

You have added the New Jersey battles data to the project. The next step is to use the data to fill up the ***path*** with ***dot***s. To understand how to do that, imagine that you have a research assistant working for you, and that you would like that research assistant to create each ***dot*** individually, as you did for the VIEW:After the Greenwich Tea Party. You might say give him/her instructions like: "Please take the information for each of these ***dot***s contained in this data source, and create the ***dot***s within this path. Please add each of the ***dot***s in the

order in which they appear in the original data set." These instructions contain three important pieces of information:

1. What you want done (fill a ***path*** with ***dot***s from a data source)
2. Where the information for the ***dot***s should come from (the specified data source)
3. Where the ***dot***s should go (into the specified path).

From these instructions, your research assistant would know exactly how you wanted to create the ***dot***s for the path.
You can give the same instructions to VisualEyes, as long as you use the right language. The language you use is a *script*, contained within a **GLUE** element. If you could speak to VisualEyes in English, you might say: "Please take the information for each of these ***dot***s from the specified data set, and add it to the specified path." To speak to VisualEyes in a language the program can understand, you make use of the ***dot****fill* command. To use ***dot***fill, all you need is the *id* of the path, and the *id* of the data you are using to fill create the ***dot***s. In this case, the ***path*** *id* is pathBattles, and the data *id* is dataBattles. The script will therefore look like this:

dotfill(pathBattles,dataBattles)

1. ***dot*****fill** tells VisualEyes what you want done (create ***dot***s from a data source)
2. **pathBattles** tells VisualEyes where the information for the ***dot***s should go (the specified path)
3. **dataBattles** tells VisualEyes where the information for the ***dot***s should come from (the specified data source)

Please note that there should not be any spaces in the script before or after parentheses or commas.

To enter it, you create a new **GLUE** element.

- Click on VIEW:All New Jersey Battles if it is not already highlighted.
- From the Add new element list box, select glue.
- From the Attribute panel, give the **GLUE** the *id* "fillBattles".
- Add the attributes *init* and *once*, and set them both to "true". This will direct VisualEyes to fill the ***path*** with the specified data only once, when the data is loaded.
- Delete "Insert script code here" from the lower panel, and insert the script ***dot***fill(pathBattles,dataBattles).

- Save & Preview, and click on the All New Jersey Battles tab.
- Click the checkbox for New Jersey Battles. The ***dot***s will appear.

[insert screen shot]

Congratulations! You have successfully used an external data file to add ***dot***s to a path, and you have controlled the display of the ***path*** with the Control Panel.

Chapter Five

In this chapter, you will learn how to create an infobox that appears whenever a ***dot*** is clicked. The infobox will display the location and year of the battle associated with each ***dot***. You will also learn more about how VisualEyes links data, as well as more about working with scripts.

You will learn how to:

1. Create a simple Infobox attached to a ***dot*** (use apples, pears)
2. Create showBattles **GLUE** to show it
3. Link data using special variables (explain about $$click, and how to connect it to mark)
4. Explain about query, list, and status, and do a status query [query($battles,dataBattles,lab+year+mark,mark EQ $$click,) status($battles)]
5. Adjust it to just be two variables, lab+year.
6. Go back to infobox, change apples, pears to $$1,$$2
7. Add replaceword to showBattles
8. Work with advanced scripts
9. Understand special characters in infoboxes
10. Understand **GLUE** and mark variables

First go round:

```
showBattles: $$1, $$2, $$3 in infobox, and
query($battles,dataBattles,lab+year+mark,mark EQ $$click,)
status($$click)
```

Save & Preview to make sure it is working properly.
Then remove $$3, and remove mark from variable list, and add:

```
 replaceword(infoBattles,$battles)
```

Creating an InfoBox

An infobox is one of the most versatile features of VisualEyes. It can be set up to appear when clicked, like a pop-up box on a webpage. For this project, we will set up an infobox to appear when each of the New Jersey battle ***dot***s is clicked. The infobox will display the location and date for each battle.
You can use a wizard to create the infobox.

- Make sure VIEW:All New Jersey Battles is highlighted, and select Infobox from the Wizards panel.

Follow the prompts for each step:

- Give the infobox the *id* infoBattles
- Specify a *solid* tail (the tail connects the box to the ***dot***)
- For the *text* to display in the box, type "Apples, Pears, Bananas". You'll change it later.
- For *position* of box, select "west".
- Give the infobox a FRAME *width* of 150 pixels.
- Give the box a FRAME *height* of 75 pixels.
- Leave the corner radius and background color at the defaults.
- Add the **GLUE** for the infobox, and give it the *id* "showBattles".
- Click Add to finish the wizard.
- Click on INFOBOX:infoBattles in the Element panel to see what you've added.

[insert screen shot]

You can see the specifications you've created for your infobox. The attributes specify how the box will look. The text "Apples, Pears, Bananas" is what will appear in the box.

You can also see the **GLUE** you've created to display the infobox, showBattles. You may remember that is the name of the **GLUE** attribute included in the dataBattles xml file. By giving the **GLUE** to display the infobox that *id*, you are giving a command to VisualEyes: You are saying, "Whenever any of the ***dot***s in PATH:pathBattles is clicked, display INFOBOX:infoBattles".
You can see how it works by viewing your project.

- Click on Save & Preview, and on the All New Jersey Battles view.
- Click on Show Battles in the Control Panel to display the path.
- Click on any of the ***dot***s. The infobox is displayed.

[insert screen shot]
The infobox as you have set it up will appear when any of the ***dot***s are displayed. If you want to change the background color or the size of the box, you can change the attributes of its ***frame***. It will always display the same text, "Apples, Pears, Bananas", because that is what we specified in the wizard.

Displaying Variables in an Infobox

Of course, we don't want the infobox to display a list of fruit. It's worthwhile thinking at this point exactly what we do want. We want a viewer to be able to click on each ***dot***, and have the infobox display the location and year of the battle associated with that ***dot***. It will make it easier to work with VisualEyes if we separate this task into a number of steps:

1. We click on a ***dot***, which calls a **GLUE** to show the infobox.
2. We want VisualEyes to go through our list of ***dot***s, and find the data associated with the one we clicked.
3. We want VisualEyes to look in that data to find the location of the ***dot*** – which we included as the variable *lab* (for label), and the year, which we included as the variable *year*.
4. Once VisualEyes has found that information, we want it displayed to as the text for the infobox (instead of "Apples, Pears, Bananas").
5. And we want VisualEyes to do exactly the same thing for each ***dot*** we click, each time locating the data associated with that ***dot*** (and only that ***dot***).

One way to do this would be to create a separate infobox for each ***dot***, and a separate **GLUE** to call each infobox. But that would be a lot of work: for our New Jersey Battles data, we would have to create over 30 infoboxes and **GLUE** elements, and enter the data for each. A more efficient way of doing it would be to create just one infobox, and ask VisualEyes to change the information in it, depending on which ***dot*** is clicked.

Using a Script with an Infobox

We can ask VisualEyes to do all these things by including a *script* in the **GLUE** that displays the infobox, **GLUE**:showBattles.

- Click on **GLUE**:showBattles in the Element panel to highlight it, then add the attribute *script*.

- Enter the first line of the script exactly as shown (no spaces between parentheses or commas, and be sure to include the final comma before the close parenthesis):

```
query($battles,dataBattles,lab+year+mark,mark EQ $$click,)
```

- Then, enter the second line exactly as shown (again, no spaces):

```
status($battles)
```

- Click on Save & Preview, then go to the New Jersey Battles view.
- Click on Show Battles in the Control Panel, and on any of the ***dot***s.

The infobox itself will look the same, since you have not changed its appearance. But if you look down at the lower left corner of the screen, you'll see three pieces of information: the location, year, and mark associated with each ***dot***. The information changes to correspond to each ***dot*** clicked.

How did that happen? To go over it, return to editing, and click on **GLUE**:showBattles to take a closer look at the script.
The first line of script sets up a *query* to select the data. The command "query($battles,dataBattles,lab+year+mark,mark EQ $$click,)" gives VisualEyes the following instructions, in order from left to right:

1. Create a new *list*, a container to put data in, called $battles [**query($battles,**]
2. Take the information from the data resource used to define each ***dot***, "XML:dataBattles" [query($battles,**dataBattles,**]
3. Go through all the data records in order, and select the information for three of the variables: lab, year, mark, in that order. Put that information in the list $battles. If the list $battles were printed, it would look like a series of rows, one per ***dot***, with three columns, for lab, year, and mark. [query($battles,dataBattles,**lab+year+mark,**]
4. For each ***dot***, compare the value of the dataset variable, mark, with the value of the variable VisualEyes creates to keep track of the order of ***dot***s in each path, called $$click. If the value of mark is equal to the value of the variable $$click, then display the data for that ***dot***. [query($battles,dataBattles,lab+year+mark,**mark EQ $$click,)**]

Using $$click to Select Data

VisualEyes creates the variable $$click whenever a ***path*** is created. The value of $$click is numbered consecutively, starting at 0 with the first ***dot*** in the path, and continuing on through all the ***dot***s in that path. The ***dot***s are ordered by the way they appear in the View (if they were entered individually) or by the way they appeared in the data file. In PATH:pathBattles, the first ***dot*** is Absecon, so VisualEyes assigned the value of $$click to "0". The last ***dot*** is Woodbridge, and VisualEyes assigned the value of $$click to "30". VisualEyes assigns each ***dot*** in the ***path*** one, and only one, value for $$click.
When we created the dataset for New Jersey battles for pathBattles, we created the variable mark as an id number, a unique identifier for each record. Each record has one, and only one value for mark. We assigned the values to go in the same order, so the value of mark for Absecon is "0", and the value of mark for Woodbridge is "30". That means that, for any given path, there can only be one value of $$click that matches any value of mark.
Many businesses use a similar technique to match customer id, held in one database, with order information, held in another.

Using Status() to Display Data

The next line of the script, status($battles), tells VisualEyes to display the current contents of the list named $battles). Remember that the current contents have been set by the query, so the list $battles will only ever contain the values for the ***dot*** that has been clicked. If no ***dot*** is clicked, the list is empty. VisualEyes is set up so that status is always displayed in the lower left corner.

You can also use the status command to display the current value for $$click.

- Edit the script so that the status command reads: status($$click).
- Save & Preview, and check the results by clicking on the New Jersey Battles ***dot***s. The value for $$click should appear in the lower left corner. You can check that the ***dot*** at Absecon appears as 0, and the ***dot*** at Woodbridge as 30.
- Return to editing.
  Using ReplaceWord to fill an Infobox from a List
  The last step for our infobox is to display the data for each ***dot*** in the box. Just as we used the script fill***dot***s to fill the ***path*** with data for each ***dot***, we can use the script replaceword to fill the infobox with the lab and year variables for each ***dot***.

Once again, we can divide the necessary tasks into steps we want VisualEyes to take:

1. We want VisualEyes to get the lab and year values for each ***dot*** (already accomplished with the query, above).
2. We want VisualEyes to replace the text in INFOBOX:infoBattles with the lab and year values. This can be subdivided into two steps:
    1. In INFOBOX: infoBattles, replace the text "Apples, Pears, Bananas" – which VisualEyes interprets as constant text, that is, text that should appear all the time – with "$$1,$$2,$$3" – which VisualEyes interprets as *placeholders* for three words appearing in sequence
    2. Use the replaceword command to replace the placeholders with the data for each ***dot***.

Begin by editing the infobox:

- Click on INFOBOX:infoBattles in the Element panel to highlight it.
- Delete Apples, Pears, Bananas from the lower panel, and replace it with the placeholders: $$1,$$2,$$3
- Click on **GLUE**:showBattles in the Element panel
- Add another line of script, typed exactly as below, with no spaces:

```
replaceword(infoBattles,$battles)
```

This tells VisualEyes to replace the placeholders ($$1,$$2,$$3) in INFOBOX:infoBattles with the values for the three variables (lab, year, mark, in that order) contained in the list $battles.

- Save & Preview.
- Go to New Jersey Battles view and click on the ***dot***s to make sure the infoboxes are working. When clicked, each infobox should display the location (lab), year, and mark for each battle (***dot***). If you look at the lower left corner, you'll see the value for $$click for each ***dot***.

Now that you've checked your data, you don't need to display the mark values.

- Return to editing
- Click on INFOBOX:infoBattles, and remove the third placeholder. The text should read: $$1,$$2

- Save & Preview. The infobox will only display the location and date.

[insert screen shot]

Congratulations! You have created and edited an infobox associated with ***dot***s on a path, and have increased your understanding of scripts in VisualEyes.