

Collision Diagram Wand

Pd' Programming, Inc.

Introduction.....	3
Magic Wands Suite.....	3
Collision Diagram Wand	3
Interfaces.....	3
Wand XML.....	3
SVG (Scalable Vector Graphics).....	3
PdWandSVGControl.....	4
WandXML – Collision Diagram	5
Input.....	5
Document Structure (input)	5
General.....	5
Command.....	6
Command / Output.....	6
Command / Output / DataList / Field	6
Command / Data / Query	7
Command / Data / Query / Parameter.....	7
Command / Data / VirtualFunctions	7
Command / Renderer / Primitives / Objects	7
Command / Renderer / SymbolTable	7
Command / Renderer / Schematic	8
DiagramSettings / Global.....	8
Sample WandXML file.....	8
Output	10
Document structure (output).....	10
General.....	10
OutputResults.....	10
OutputResults / CrashRecord / Field	10
Sample WandXML response	11
Configuration	13
Data Access.....	13
Virtual functions	13
Object table.....	13
Schematics	13
Schematics	13

Introduction

Magic Wands Suite

The Magic Wands Suite is a collection of components designed to assist software developers creating advanced data analysis programs. The Suite provides tools to generate reports, diagrams, charts, and to manage uniquely challenging data manipulation situations.

The components in the Magic Wands Suite were built using the technology and experience from the many years of writing and supporting the Intersection Magic products.

Collision Diagram Wand

The Collision Diagram Wand is a component that generates collision diagrams for graphical analysis of crash data records. The Diagram Wand uses an XML stream to define the settings to be used when generating the diagram. An XML stream is also used to return the diagram content and any other information such as errors that may have occurred during processing.

Interfaces

- ActiveX – for conventional Windows desktop applications
- COM+ – for conventional Windows applications where server-side processing is desired.
- Web Service – For networked applications including .Net applications where server-side processing is desired.
- VCL – For Borland Delphi & C++ Builder applications.

Wand XML

Wand XML is the data format used for making calls to, and receiving a response from most Magic Wand applications. Wand XML is an evolving specification. The process looks like this:

The application calling the Wand creates an XML document that follows the Wand XML specification. The document contains all the information needed to generate the desired collision diagram. This XML content is passed to the wand which uses the information to generate the diagram. The Wand then generates a new XML document, also following the Wand XML specification, that contains the results. These results may be as simple as the output filename, as complex as an embedded XML document, or a list of error messages describing why the process failed.

SVG (Scalable Vector Graphics)

SVG is a W3C XML specification for creating two dimensional graphics such as collision diagrams. SVG is an ideal format for Wands graphical output, for a number of reasons:

- It is an open format
- It is a platform independent format
- As an XML implementation, it can be displayed, searched and manipulated with standard XML tools.
- It provides functionality for animation, interaction and dynamic manipulation using the built-in ECMA scripting language.
- Viewers are available for many platforms including web browsers.

There are a number of SVG viewing and editing implementations available including applications for Windows, Java, Macintosh, etc. The most popular implementation on the Windows platform is currently the Adobe SVG Viewer v3.

The Adobe SVG viewer was designed as a web browser plug-in and does a wonderful job in that context. As an ActiveX plug-in it can also be called by other stand-alone applications written in most modern programming libraries. However, the standard Adobe SVG Viewer ActiveX control does not easily expose the type of functionality needed by stand-alone applications.

PdWandSVGControl

In order to be able to write an application that uses the Adobe SVG Viewer, Pd' Programming created the PdWandSVGControl. This control is an ActiveX "wrapper" that provides a more usable interface to the Adobe ActiveX control.

The PdWandSVGControl provides simple programmatic access to the following types of functionality:

- Loading of SVG documents located in a file
- Loading of SVG documents located in memory
- Event handlers to detect mouse movement related to the diagram graphics
- Event handlers to detect mouse clicks and drags related to diagram graphics
- Functions to obtain the source for all or part of an SVG document
- Functions to convert between screen and diagram coordinate systems
- Functions to reference diagram graphics using programmer defined id strings, such as case id numbers.
- Function to print graphic image to a printer DC (HDC or handle).

WandXML – Collision Diagram

Input

Document Structure (input)

```

<WandXML>
  <General>
    ... basic version and time/date information
  </General>
  <Command>
    <Action>CollisionDiagram</Action>
    <Output>
      ... variables such as filenames, types, etc.
    </Output>
    <Data>
      ... variables for connecting to and querying crash data
      ... also includes references to virtual functions
    </Data>
    <Renderer>
      ... information used for rendering diagrams
      ... references to graphic primitives, symbol tables
      ... references to schematics
    </Renderer>
    <DiagramSettings>
      ... variables used for collision diagrams
    </DiagramSettings>
  </Command>
</WandXML>

```

General

ExpectedVersion	Text	Expected Wand version.	Wand version number that the application is written against. * A warning will be reported if this value doesn't match the current wand version.
CallingTimeStamp	Date [time]	Time when call was prepared	<i>Optional</i> This value is passed back in the return xml.
CallingProgram	Text	Name of program calling the wand.	<i>Optional</i> identifier useful for debugging and process tracking
CallingVersion	Number	Version of the calling program	<i>Optional</i> information
UniqueID	Text	Application data	<i>Optional</i> data that will be passed back in the return xml.

Command

Action	Text	Wand action to perform	For collision diagram, this value should be: "CollisionDiagram"
--------	------	------------------------	--

Command / Output

FileName	Text	Name of output file if applicable	If file output is requested, this parameter tells the wand where to write the file.
Method	Text	Means of outputting the data	This value may be either "file" or "inline". File: Write the content to this file name. Inline: Content should be included in the xml content of the output XML.
UseJavaScript	Boolean	Determines if output will include ECMA (JavaScript) code.	This value should be "True" if the SVG diagram will be presented in a web browser with interactivity provided by Javascript. This value should be "False" if the SVG diagram will be used in the PdWandSVGControl ActiveX component.
ADOxmlFN	Text	If exists, will output an xml dataset to this file.	This value should be a filename where an xml version of the data included in the diagram will be written. The format will be that of the ADO dataset SaveToFile() function using pFXML as the format.

Command / Output / DataList / Field

Name	Text	Optional, name of field to include in resulting data list section.	The field name should be as it would appear as a result of the query. (i.e. should not include table prefix and should include any field name assignment using "as")
------	------	--	--

Command / Data / Query

ConnectionString	Text	ADO connection string.	See appendix for examples. A utility is included for creating these strings.
UserId	Text	Login ID for this connection.	User name / login / etc. (note: if no security, try using "admin")
Password	Text	Password that matches UserId	Password for this connection. If no login, or using default "admin", leave password blank.
SQL	Text	SQL statement for retrieving the desired crash data.	The resulting table should be one row per crash with all the required columns for generating a diagram.

Command / Data / Query / Parameters / Parameter

Name	Text	SQL parameter name used in the SQL query	Optional means of changing data values in the SQL query.
Value	Text	The value of the SQL parameter	Text representation of the type specified in the "type" value.
Type	Text	Type of the parameter	"TEXT" or "NUMBER" or "DATE"

Command / Data / VirtualFunctions / VirtualFunction

SourceFN	Text	Name of virtual functions file	This will generally be a physical file name, but may also be a URL
SourceName	Text	Name of virtual functions section	Must be "default"

Command / Renderer / Primitives

SourceFN	Text	Name of graphic primitives file	Path and name of "objects.im"
SourceName	Text		Must be "default"

Command / Renderer / SymbolTable

SourceFN	Text	Name of symbol table file	Path and name of symbol table file
SourceName	Text		Must be "default"

Command / Renderer / Schematic

SourceFN	Text	Name of schematics file	Path and name of schematics file
SourceName	Text		Default is "int_4/normal"

DiagramSettings / Global

Name	Text	Name of diagram	Used for display
Spacing	Number	Default distance between crashes	Default is 50.0
PenWidth	Number	Default thickness of line graphics	Default is 1.0

Sample WandXML file

```

<?xml version="1.0"?>
<WandXML xmlns:wandxml="http://pdmagic.com/wands/wandxml/2.5">
  <General>
    <ExpectedVersion>2.5.0.2</ExpectedVersion>
    <CallingTimeStamp>5/20/2003 12:01:00 AM</CallingTimeStamp>
    <CallingProgram>DiagramWandHarness.exe</CallingProgram>
    <CallingVersion>1.0</CallingVersion>
    <UniqueID>Call1002</UniqueID>
  </General>
  <Command>
    <Action>CollisionDiagram</Action>
    <Output>
      <FileName>C:\Program
files\PdMagic\MagicWands\DiagramWand\samples\temp\output.svg</FileName>
      <Method>file</Method>
      <UseJavaScript>True</UseJavaScript>
      <ADOxmlFN></ADOxmlFN>
      <DataList>
        <Field>
          <Name>CaseID</Name>
        </Field>
        <Field>
          <Name>CollisionDate</Name>
        </Field>
      </DataList>
    </Output>
    <Data>
      <Query>
        <ConnectionString>Provider=Microsoft.Jet.OLEDB.4.0;
Data Source=C:\Program Files\PdMagic\MagicWands\DiagramWand\samples\csv\data;
Extended Properties="text;HDR=NO;FMT=Delimited"</ConnectionString>
        <UserId>admin</UserId>
        <Password></Password>
        <SQL>
SELECT c.CaseId, c.CollisionDate, c.PrimaryRoad, c.SecondaryRoad, c.Distance,
c.Direction, c.TypeOfCollision, c.CollisionSeverity, p1.DirectionOfTravel as
DirectionOfTravel_One, p1.MovementPrecedingCollision as
MovementPrecedingCollision_one, p2.DirectionOfTravel as DirectionOfTravel_Two,
p2.MovementPrecedingCollision as MovementPrecedingCollision_Two,
p3.DirectionOfTravel as DirectionOfTravel_Three, p3.MovementPrecedingCollision
as MovementPrecedingCollision_Three
FROM (( collision.txt c
LEFT JOIN party.txt AS p1 ON (( c.caseid=p1.caseid ) and ( p1.partynumber = 1
)))

```

```

LEFT JOIN party.txt AS p2 ON (( c.caseid=p2.caseid ) and ( p2.partynumber = 2
))
LEFT JOIN party.txt AS p3 ON (( c.caseid=p3.caseid ) and ( p3.partynumber = 3
))
WHERE ((PrimaryRoad = :PrimaryStreetA AND SecondaryRoad = :CrossStreetA) OR
(PrimaryRoad = :PrimaryStreetB AND SecondaryRoad = :CrossStreetB)) AND
(CollisionDate >= :FirstDate AND CollisionDate &lt;= :LastDate) AND Distance >=
:MinDistance
ORDER BY c.Caseid
</SQL>

```

```

<Parameters>
  <Parameter>
    <Name>FirstDate</Name>
    <Value>01/01/1998</Value>
    <DataType>DATE</DataType>
  </Parameter>
  <Parameter>
    <Name>LastDate</Name>
    <Value>12/31/2002</Value>
    <DataType>DATE</DataType>
  </Parameter>
  <Parameter>
    <Name>PrimaryStreetA</Name>
    <Value>VENTURA RD</Value>
    <DataType>TEXT</DataType>
  </Parameter>
  <Parameter>
    <Name>PrimaryStreetB</Name>
    <Value>WAGON WHEEL RD</Value>
    <DataType>TEXT</DataType>
  </Parameter>
  <Parameter>
    <Name>CrossStreetA</Name>
    <Value>WAGON WHEEL RD</Value>
    <DataType>TEXT</DataType>
  </Parameter>
  <Parameter>
    <Name>CrossStreetB</Name>
    <Value>VENTURA RD</Value>
    <DataType>TEXT</DataType>
  </Parameter>
  <Parameter>
    <Name>MinDistance</Name>
    <Value>1</Value>
    <DataType>NUMBER</DataType>
  </Parameter>
</Parameters>

```

```
</Query>
```

```
<VirtualFunctions>
```

```
<VirtualFunction>
```

```
<SourceFN>C:\Program
```

```
Files\PdMagic\MagicWands\DiagramWand\samples\csv\config\VirtualFunctions.xml
```

```
</SourceFN>
```

```
<SourceName>default</SourceName>
```

```
</VirtualFunction>
```

```
</VirtualFunctions>
```

```
</Data>
```

```
<Renderer>
```

```
<Primitives>
```

```
<SourceFN>C:\Program
```

```
files\PdMagic\MagicWands\DiagramWand\samples\csv\graphics\objects.im</SourceFN>
```

```
<SourceName>default</SourceName>
```

```
</Primitives>
```

```

    <Schematic>
      <SourceFN>c:\Program
files\PdMagic\MagicWands\DiagramWand\samples\csv\config\Schematics.xml</SourceFN
N>
      <SourceName>Int_4/Normal</SourceName>
    </Schematic>
    <SymbolTable>
      <SourceFN>C:\Program
files\PdMagic\MagicWands\DiagramWand\samples\csv\config\ObjectMap.xml</SourceFN
>
      <SourceName>default</SourceName>
    </SymbolTable>
  </Renderer>
  <DiagramSettings>
    <Global>
      <Name>untitled</Name>
      <PenWidth>1</PenWidth>
      <Spacing>50</Spacing>
    </Global>
  </DiagramSettings>
</Command>
</WandXML>

```

Output

Document structure (output)

General

Version	Text	Wand version.	Version of this Wand
TimeStamp	Date Time	Time when call was completed	This time stamp is set as the wand completes processing.
[User Data]	?	User data from input	All of the optional fields including CallingProgram, CallingTimeStamp, UniqueId, etc. are included in the output.

OutputResults

FileName	Text	Filename	If the Output/Method in the input request was "file", this value will contain that file name.
Svg	Text	SVG document	If the Output/Method in the input request was "inline", this tag will contain the entire SVG document.

OutputResults / Data / Records / Record / Field

Name	Text	Name of field.	Name of field as exists in resulting table.
Type	Text	Type of field.	Field type. Possible values

			are: "text", "number", "date", "boolean", "unknown".
Value	Text	Field value represented as a string.	Field value is converted to a string type and returned here.

Errors

Error	Text	Error message	<p>Each message describes a step in the program that caused an unrecoverable error. This error list will often resemble a program "stack" when the error occurred.</p> <p>The "class" attribute is often helpful to Pd' Programming support staff when determining the module in which the error occurred.</p>
-------	------	---------------	--

Warnings

Warning	Text	Warning message	<p>Each message describes an occurrence in the program where data values or other information may lead to an error, or invalid output. Unlike "Errors" above, warnings need not be related in any way.</p> <p>The "class" attribute is often helpful to Pd' Programming support staff when determining the module in which the warning.</p>
---------	------	-----------------	---

Sample WandXML response

```
<?xml version="1.0"?>
```

```
<WandXML>
```

```
  <General>
```

```
    <Version>2.5.0.3</Version>
```

```
    <ExpectedVersion>2.5.0.2</ExpectedVersion>
```

```
    <TimeStamp>1/2/2004 4:33:08 PM</TimeStamp>
```

```
    <CallingTimeStamp>5/20/2003 12:01:00 AM</CallingTimeStamp>
```

```
    <CallingProgram>DiagramWandHarness.exe</CallingProgram>
```

```
    <CallingVersion>1.0</CallingVersion>
```

```
    <UniqueID>Call1002</UniqueID>
```

```
</General>
<OutputResults>
  <FileName>C:\Program
files\PdMagic\MagicWands\DiagramWand\samples\temp\output.svg</FileName>
  <svg/>
  <Data>
    <FieldList/>
    <Records>
      <Record>
        <Field>
          <Name>CaseId</Name>
          <Value>5604010330083504560</Value>
        </Field>
        <Field>
          <Name>CollisionDate</Name>
          <Value>3/30/2001</Value>
        </Field>
      </Record>
      ... (more records as needed)
      <Record>
        <Field>
          <Name>CaseId</Name>
          <Value>5604010403171804531</Value>
        </Field>
        <Field>
          <Name>CollisionDate</Name>
          <Value>4/3/2001</Value>
        </Field>
      </Record>
    </Records>
  </Data>
</OutputResults>
<Errors/>
<Warnings>
  <Warning class="Data">No value for Query.Password</Warning>
</Warnings>
</WandXML>
```

Configuration

Data Access

Collision Diagram SVG Format		
Diagram settings		User stored information
Schematics	Object table	Graphic primitives
Virtual functions		

Virtual functions					
Pass-thru fields		Program defined fields		User defined fields	
Result Table (flat – 1 crash per row)					
SQL Query + parameters					
ADO + Connection String					
ASCII	dBase	MS Access	Misc ODBC	MS SQL Server	Oracle 8i or 9i

Virtual functions

Object table

Schematics

Schematics