

# Justine User's Manual

Stephen R. Lee

---

Los Alamos National Laboratory

Applied Theoretical and Computational Physics Division

Computational Science Methods Group

Mail Stop F645

*srlee@lanl.gov*

LA-UR-95- 3304

## Preface

(Read this at least once)

## Table of Contents

---

*Last Modified Thu Sep 7 13:35:26 MDT 1995, Stephen R. Lee*

*srlee@lanl.gov*

---

## Table of Contents

### Chapter 1: Overview of Justine

*Index*

**Required Concepts**

**The Main Window**

**The 3-D Window**

**The 2-D Windows**

### Chapter 2: Quick Start

#### DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

**MASTER**



## **DISCLAIMER**

**Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.**

*Index*

**Steps to Constructing a Problem**  
**Building Geometry**  
**Creating and Assigning Materials**  
**Setting LARAMIE Options**  
**Executing the Transport Code of Choice**  
**Summary**

**Chapter 3:Defining Geometries**

*Index*

**Bodies v.s. Surfaces**  
**Creating Objects Using Property Sheets**  
**Creating Objects by Dragging**  
**Creating Objects, Shortcuts**  
**Visualizing Geometries, 3-D**  
**Visualizing Geometries, 2-D**  
**Editing Options and Scrolling Lists**  
**The Edit Menu (Main Window)**  
**Cutter Objects**  
**The Global View Menu (Main Window)**  
**2-D Global View Options**  
**Special Object Property Sheets**  
**Summary**

**Chapter 4: Creating Cells**

**Chapter 5: Building Materials**

**Chapter 6: LARAMIE Code Specific Options**

**Chapter 7: Automatic Variance and Time of Analysis Reduction (AVATAR)**

**Chapter 8: Justine Power Features**

**Chapter 9: Justine Help Package**

**Chapter 10: Brief Tutorial Examples**

MCNP  
DANTSYS  
AVATAR  
X3D

**Appendix A: Interacting with Motif**

**Appendix B: The Resources File**

**Appendix C: The Restart File Format**

**Appendix D: Frequently Asked Questions**

**Global Index**

---

*Last Modified Thu Sep 7 13:35:26 MDT 1995, Stephen R. Lee*

srlee@lanl.gov

---

## Preface

This online manual was written in Hyper-text Markup Language (HTML) and was meant to be viewable using any web browser, as well as Justine's internal help package. However, due to the size of the document and the fact that it was tuned to work best with Netscape, it is suggested that the reader use that to view the document.

The table of contents contains links to the chapters of the manual as well as major topics within those chapters. Each chapter also has its own index, which allows the reader to look for specific topics within that chapter at a level of detail greater than that provided in the table of contents. The global index is a collection of these chapter indices. *These indices are not available yet.*

While HTML offers a variety of advantages, it also has the disadvantage of not allowing some of the more powerful document formatting features offered in conventional publishing software. Therefore, some typical publication conventions are not followed in this document.

This document may seem rather "wordy" for a hypertext document. That is due to the fact that this is a user's *manual*, and meant to be a stand-alone document that is printable and readable outside the execution context of Justine. While some of the same words in the manual are included in the context-sensitive help features of Justine, the two are quite distinct from one another.

**This document is still in progress and not all pieces may yet be available.**

It is assumed that the viewing software is capable of producing PostScript files for printing, if desired. If not, contact the author for a PostScript file.

Justine is a trademark of the Regents of the University of California, Los Alamos National Laboratory.

---

*Last Modified Thu Sep 7 13:35:26 MDT 1995, Stephen R. Lee*

srlee@lanl.gov

---

# Chapter 1: Overview of Justine

---

## Introduction

Justine is the graphical user interface to the Los Alamos Radiation Modeling Interactive Environment (LARAMIE). It provides LARAMIE customers with a powerful, robust, easy-to-use, WYSIWYG interface that facilitates geometry construction and problem specification.

It is assumed that the reader is familiar with LARAMIE, and the transport codes available, i.e., MCNPTM and DANTSYS. No attempt is made in this manual to describe these codes in detail. Information about LARAMIE, DANTSYS, and MCNP are available elsewhere<sup>1,2,3</sup>. It is also assumed that the reader is familiar with the Unix operating system and with Motif widgets and their look and feel<sup>4</sup>. However, a brief description of Motif and how one interacts with it can be found in Appendix A.

Additional information about Justine is available from our home page:  
<http://www-xdiv.lanl.gov/XCM/Justine>.

---

## Required Concepts

To enable a user to visualize generated geometries, Justine uses four views of the user's geometry: A three-dimensional (3-D) wire-frame or shaded polygonal view, and three two-dimensional (2-D) slices through the user's 3-D geometry. In Figure 1-1, we see a 3-D wire-frame image of two simple spheres. In Figure 1-2, we see the three 2-D cut planes through the geometry. Each plane is a slice in the XY, XZ, and YZ planes. As can be seen in Figure 1-1, the 3-D axes are colored. The x-axis is red, the y-axis is blue, and the z-axis is green. In Figure 1-2, the borders of the slice planes are also colored. This serves as a visual reminder as to which axis is being looked down. For example, the border of the XY slice window is green, indicating that the user is looking down the z-axis.

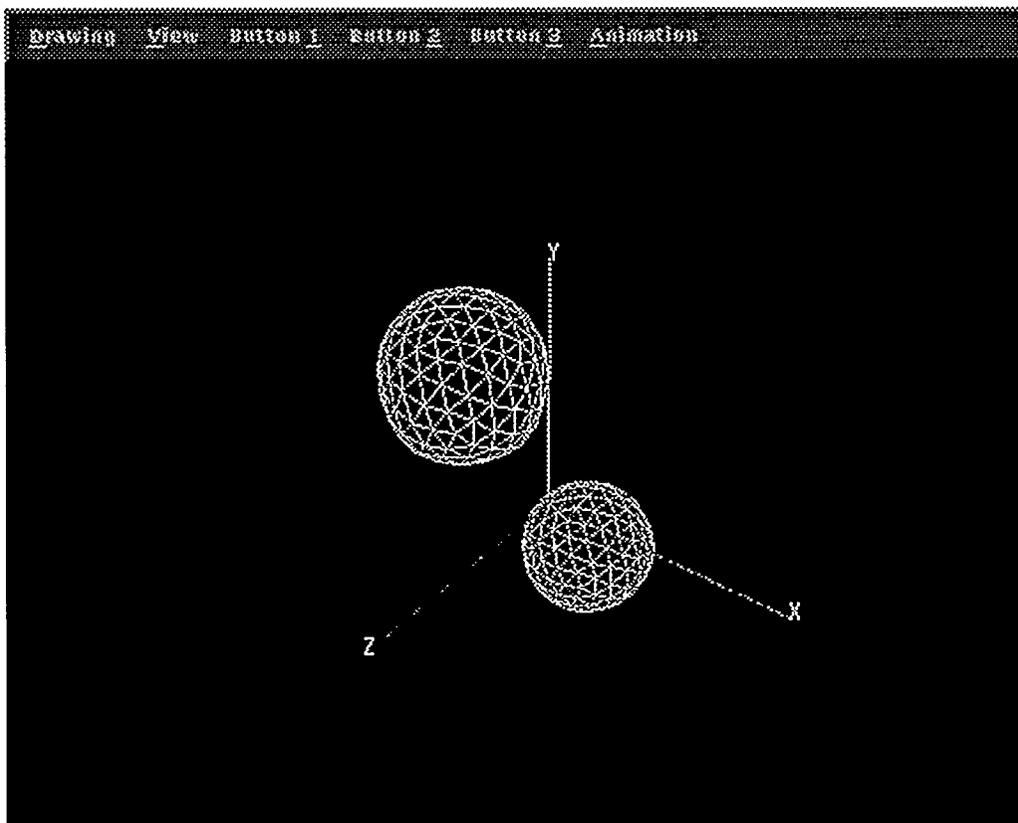


Figure 1.1: Wireframe view of two simple spheres.

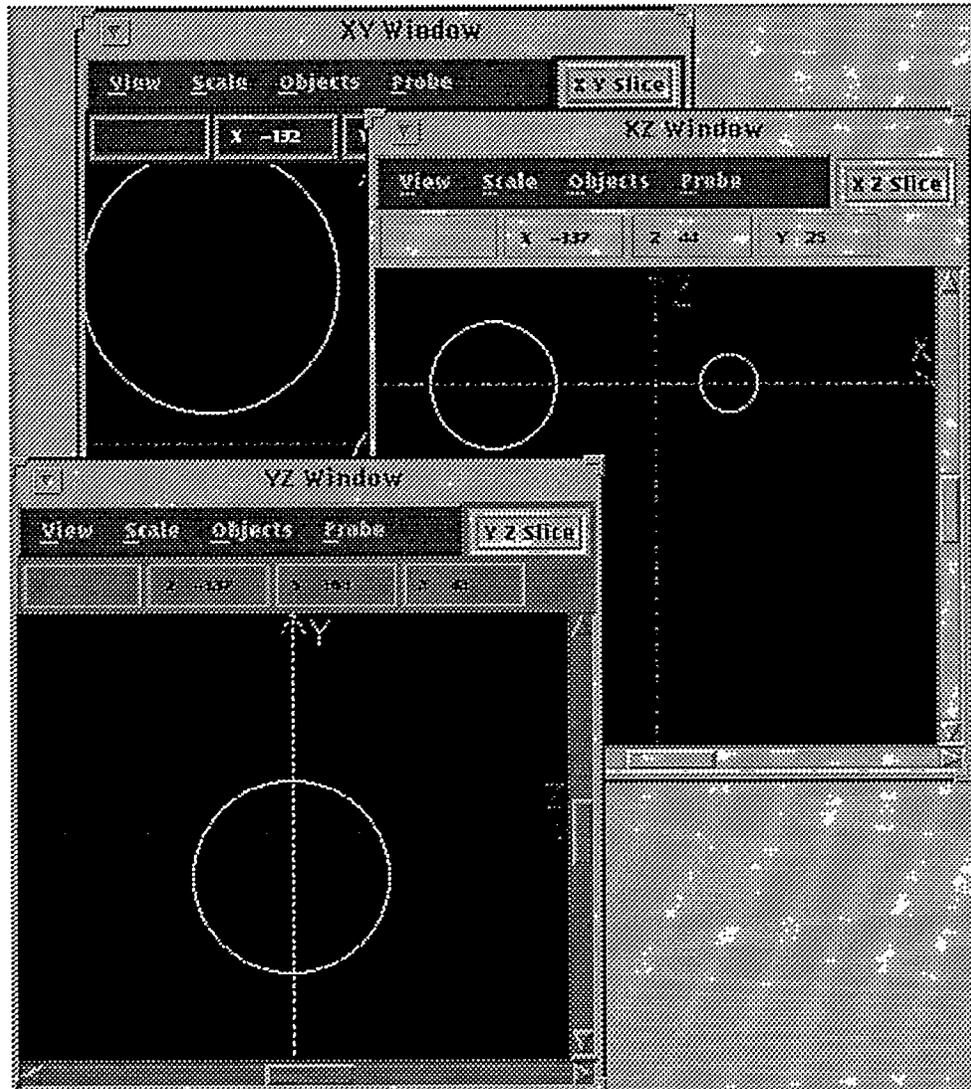


Figure 1.2: Cut plane views of the 3-D geometry.

With Justine, user's build geometries using 3-D bodies and 2-D surface. Bodies and surfaces may be created in two ways: Using property sheets or by dragging the object in one of the 2-D windows. This will be described in detail later. For now, just understand that there are two ways in which a user may realize the available objects in Justine. Throughout the manual, I will refer to bodies and surfaces as objects when it is not important to distinguish between 2-D surfaces and 3-D bodies.

One uses these objects to build complex geometries. For example, if the geometry desired was a box with a cylindrical hole in it, one would realize a box body and a cylinder body in the desired locations and sizes, then subtract the cylinder from the box using the combinatorial solid geometric operations available in Justine. The important thing to understand here is that one creates, or realizes, primitive objects and surfaces in Justine and uses CSG operations to create more complicated objects. The manner in which one accomplishes this will be described in Chapters 3 and 4.

## Representation of Geometry

Justine represents the 3-D objects in either wire-frame or shaded mode. Visually and internally, the 3-D

geometry is represented and rendered as a series of polygons. The wire-frame images are simply the outline of the polygons that lie on the surface. They have no effect on the transport codes in LARAMIE, as this information is never passed to these codes. The polygons are for visualization purposes only. When the objects are shaded, the faceted appearance of the objects is again due to the polygonal representation of the geometries.

In addition to the polygonal shading, ray-trace imaging is also available. This is slower than polygonal shading, but the quality of the image is better as no faceting occurs. Justine offers three modes for ray-tracing images: coarse, fine, and super fine.

The 2-D window geometries are rendered as outlines of the intersection of the 3-D objects and the slice plane. When these objects are shaded (as cells), the shading is accomplished with scan lines.

---

## Justine's Windows

When one executes Justine, five windows are launched. Three 2-D slice view windows, one 3-D window, and Justine's main window, with a menu bar and body primitive icons. The four geometry rendering windows are iconified by default on start-up. Each of these windows will now be discussed in more detail.

### The Main Window

Justine's main window consists of a menu bar and body primitive icons (Figure 1-3). The majority of user control over Justine, how the software acts and what it does, is obtained through this main window. Each item on the menu bar will now be described in more detail.

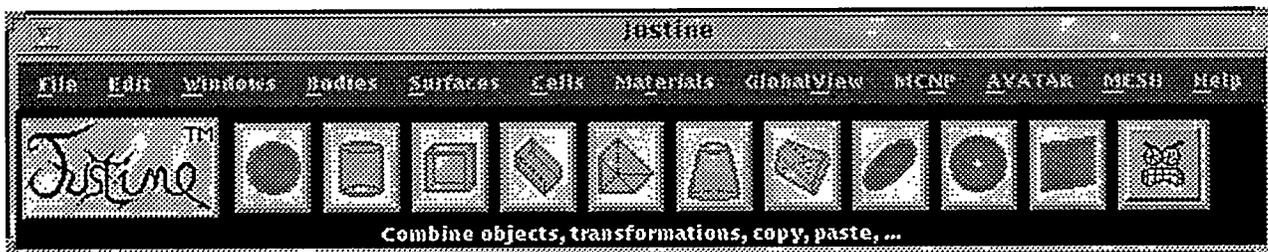


Figure 1.3: Justine main window.

### File

The File menu item is used to manage Justine input and output. Through this menu, one may read and write Justine restart files, MCNP input files, Sabrina input files, Justine composite library files, material composite files, DANTSYS input files, and so on. Each individual menu entry will now be explained in more detail.

#### 1. Load

This sub-menu allows a user to load a variety of files into Justine. Current menu entries allow:

- Justine Restart
- IGES File
- Justine Library
- Material File
- MCNP Data File
- Sabrina
- MCNP Geometry
- MCNP Surfaces
- MCNP Input Deck
- PTRAC File
- WWINP File

These files will be discussed in detail later. Briefly, a Justine restart file is a user generated file that saves the current Justine state. All geometries, parameters, materials, and so on are stored in this file enabling the user to quit and restart Justine without losing work. *Generally speaking, it is a very good idea to create Justine restart files at major milestones of problem generation.* The MCNP data file refers to the separate file containing MCNP information that is generated when the Justine restart file is written. This file may be read in separately to apply MCNP specific setting to different problem setups. Library files, on the other hand, allow a user to bring in previously created geometries into a current problem. A material file is one that contains user defined or pre-defined composite materials for repeated use in a variety of problem definitions. Finally, can also read in a variety of different pieces of an existing MCNP input file, including just geometry (which includes cell definitions), just surface information (which excludes cells), and the entire deck, which includes surfaces, cells, materials, and other MCNP parameters. The remainder of the files will be discussed later.

In each case, the file selection widget is presented to the user when the read is requested. This widget, seen in Figure 1-4, allows you to navigate files and directories and select the desired file for reading. Justine has pre-defined "filters" that vary depending on the type of file read that was requested. These filters are:

- \*.restart -- Justine restart files
- \*.lib -- Justine library files
- \*.mcn -- MCNP input files
- wwinp\* -- WWINP files
- \*.mat -- Material files

Naturally you may have files that you wish to read that do not match these naming conventions. The filter text field at the top of the file selection box allows you to alter the filter to anything you desire.

Moving through your directories is accomplished by selecting the desired directory as it appears in the left most scrolling list. You select the directory by double-clicking it with the left mouse button. The ".." entry in the list indicates the previous directory. Therefore, you would select that to move up in your directory hierarchy. You may also enter the name of the directory directly in the filter text-field.

Files appear in the right-most scrolling list. Selecting a file in this list causes the file selection widget to disappear, and that file to be read into Justine. Once again, double clicking the file selects the file for read and begins the read operation. You may enter the file, with path name if desired, in the lower text field, then select the "Ok" button. As with all of Justine's pop-up's, the "Cancel" button will simply

cancel the entire read operation.

## *2. Save, Save As, and TTY Echo*

This submenu allows a user to write a file in a variety of formats. These formats include:

- Justine restart and library files
- MCNP input files
- Sabrina input files
- THREEDANT input files
- FRAC input files
- Polygon files

The first save is accomplished using the "Save As" menu item. Future saves may then be done using the "Save" feature. This allows rapid updates to the same files as the problem definition proceeds. The "TTY Echo" submenu is related to the "Save As" feature, as it allows the user to view the input file on the screen, or tty as Justine refers to it, without actually generating the file. This can be useful to simply do a "sanity check" and reassure yourself that the MCNP, Sabrina, THREEDANT, or FRAC input file looks as you expect it to look.

Just as with the Load submenu, the Save As submenu will bring up the file selection widget. The operation in this case is the same as previously described.



Figure 1.4: File selection widget.

### 3. Reset

This menu item completely clears Justine's memory. This is equivalent to quitting Justine and restarting it. A dialog box appears when this is selected, giving the user the opportunity to cancel the operation. By selecting "Disregard Changes," Justine's memory is reset. **This operation will completely erase all data, and is not reversible!**

### 4. Quit

This will exit Justine. Selecting Justine's logo will also terminate Justine. Anything that has not been saved to a restart file will be lost.

### Edit

The edit menu contains a variety of items relating to the manipulation of geometry in Justine. Most of these items will be described in detail in later sections. However, in summary, menu items under this heading will bring up property sheets for objects, cells, and other options, and will enable or disable

cutter objects. All of these advanced features will be discussed in following sections.

## **Windows**

This menu allows a user to raise or iconify all or any combination of the 2-D and 3- D geometry windows in Justine. This saves you from having to search your work space on your platform for the iconified windows.

## **Bodies**

This menu contains all editing functions for all body primitives in Justine. Some detail is given below.

### **1.Properties**

This brings up the property sheet for entering exact body (or surface) descriptions. For example, the property sheet allows you to enter a sphere of exactly 12.5 cm in radius, centered at (10,12,13,5). As will be seen later, a single property sheet can be used to enter quantities for all of Justine's body primitives and surfaces.

### **2.Nest**

This brings up a property sheet that allows a user to nest any number of bodies within another body.

### **3.RPP, Arbitrary Box, Wedge, etc.**

These submenus allow a user to bring up a property sheet for each of these body primitives. Since the same property sheet is used for all body and surface objects, selecting these items from the menu bring up the property sheet in the correct mode for the object selected.

In addition, the icons shown on the main window depict images of some of the individual bodies and surfaces available in Justine. Selecting these icons will also produce the property sheet for that object.

### **4.Draw in 2-D**

This submenu allows a user to *drag out* the bodies in the 2-D window(s), without having to type in exact quantities. The submenu has a cascading menu attached to it, which allows the user to specify different restrictions on the creation of the object. For example, a general sphere can be generated, or a sphere centered on the origin, or centered on any of the three coordinate axes may also be generated. When one of these items is selected from the menu, the mouse becomes armed for dragging, and one or more of the 2-D windows will become *active* (depending on the primitive requested). Since it will be important to select the correct 2-D slice, the user is given a visual cue as to which slice is "selectable" given the primitive selected (see Figure 1-5).

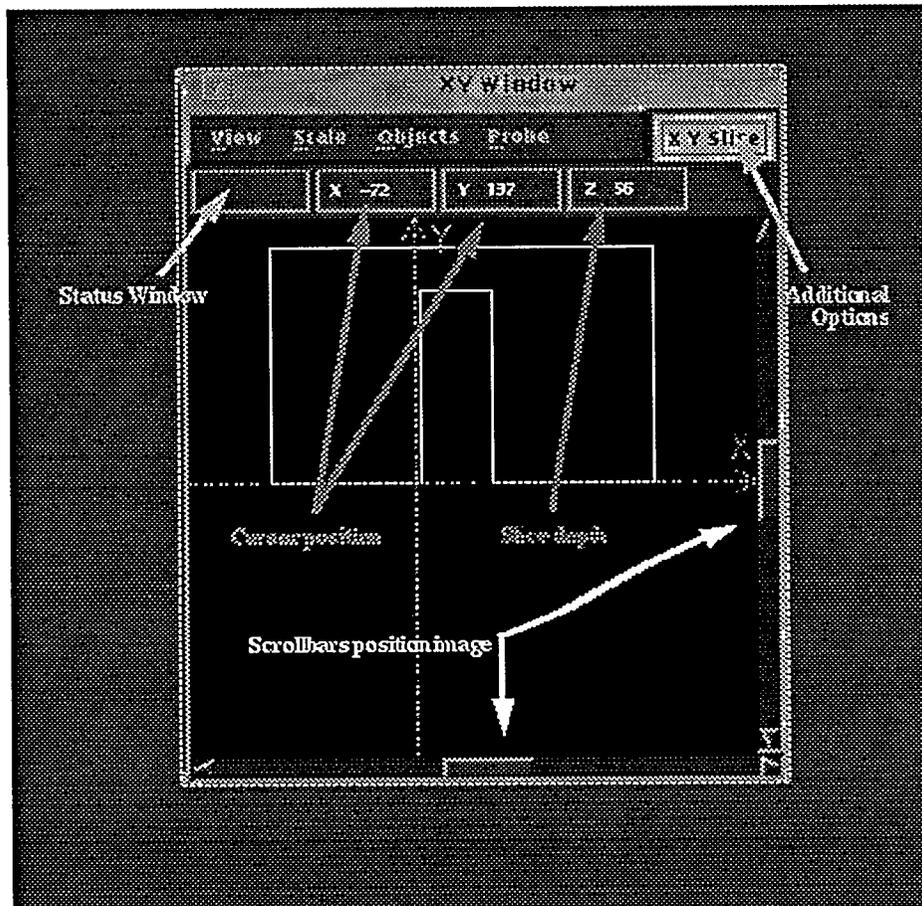


Figure 1.5: 2-D window.

### 5. Show Geometry List

This option will display a list of all bodies and surfaces generated by the user. This will become important later as the user will need to select different objects for different operations. The objects are listed by name and Justine ID number. The names may have been entered by the user, or the Justine default names may appear (depending on what the user did).

### Surfaces

Entries here mirror those for Bodies. Functionality is identical.

### Cells

This menu contains functions for generating and modifying MCNP cells (discussed later).

### Materials

This menu contains functions for generating and modifying materials (discussed later).

### Global View

As the name implies, this menu controls the general viewing parameters for Justine. Centering of the view port on different objects (as selected from the lists), showing and hiding different objects or sets of objects, and coloring the objects are all modifiable through this menu selection. The "2D Options" submenu allows the same modifications to be made to the 2-D slice views.

## **MCNP**

This menu item contains all MCNP specific items, such as tallies, sources, and so on. Generally speaking, each submenu item here brings up a separate property sheet tailored for that particular option. All MCNP options are described later.

## **AVATAR**

AVATAR stands for Automatic Variance And Time of Analysis Reduction. It is a special package available through Justine in which THREEEDANT is used to generate variance reduction information for MCNP. This menu item controls the AVATAR process which is described later.

## **Mesh**

Mesh generation is described later.

## **Help**

This menu contains all Justine help package functions. These include sending mail to Justine developers using the gripe facility, loading the main window help package, and inquiring about technical support. All of Justine's online help, as this user manual itself, are hyper-text based.

---

## **The 3-D Window**

As we have seen in Figure 1-1, the 3-D window is used to render the user's geometry in three dimensions.

The user may interactively rotate, zoom, and pan their geometries using the mouse. Justine's rotational axes are visual axes of rotation in the user's reference frame. The zoom and pan reference point is the origin.

Rotations are an intuitive "arcball" implementation. Holding down the left mouse button in the 3-D window will arm the mouse for rotation operations. The cursor changes to two arrows, indicating that the mouse is rotation-enabled. The default mode is unconstrained rotations about the origin of the coordinate system. This allows a user to have the effect of "dragging" one part of the geometry to another part of the screen (e.g., moving the z-axis up). The best way to get a feel for the rotations in Justine is to simply experiment with them.

The middle mouse button controls the location of the object(s) within the window, that is, panning is controlled by this button. When the middle mouse button is selected, the cursor changes to a crossing

arrow pattern, indicating that the mouse is armed for pan. Motions with the mouse correspond directly with geometry translations.

The right mouse button controls zooming. When this mouse button is selected, the cursor reverts to a magnifying glass icon, indicating that the mouse is armed for zooming. Motion of the mouse to the right enlarges the geometry, motion to the left reduces the geometry.

A close inspection of Figure 1-1 reveals a menu bar at the top of the 3-D window. This bar allows one to control many aspects of the visualization of the geometry in the 3-D window. Through this menu you may, for example, shade geometry and change default mouse actions.

## **Drawing**

This menu allows the user to control how the geometry is rendered.

### **1. *Shade***

This submenu has an attached cascading menu that allows users to select geometry shading options. Polygon shading is generally the best choice, since it is fast and provides excellent rendering of the 3-D geometry as a solid object. Ray tracing is also offered in this submenu.

### **2. *Wireframe***

This is a selectable option which disables shading and leaves the geometry rendered in wireframe mode. Wireframe rendering is always done by drawing polygon edges on the surface of the object.

### **3. *Backcull/No Culling***

This refers to hidden line removal. When the Backcull option is selected (which is the default) polygons that are on the "back" of the rendered object are not visible. If the no culling option is selected, all surface polygons are rendered, including those that are on the backside of an object in its current orientation. Generally speaking, culling hidden polygons makes the images much cleaner and easier to visualize.

### **4. *Double Buffer/No Double Buffer***

This refers to the rendering quality of the geometry during rotations. No Double Buffer is the default, because it is fastest. However, the geometry will have a "flashing effect" as it is rotated. To remove this effect, Double Buffering should be enabled. However, rotations will be slower and, depending on the complexity of the geometry, the interactivity with the mouse will be reduced.

### **5. *Show/Hide Axis***

This toggles the coordinate axes on and off. The default is to show the axes.

### **6. *Set/Use Light Source, Etch, Shadowing, etc.***

These options all enhance, to different degrees, the quality and realism of ray traced images in the 3-D

window.

### ***7.Save Picture***

This entry contains a sub-menu that allows the user to save either the window or the image (as defined by the world box for the geometry) as a GIF file or a PostScript file. The PostScript implementation allows the user to set various additional parameters that affect how the image will be printed, including the size of the image on the paper, the position, portrait or landscape, different media types, and so on.

### ***8.Polygonized CSG***

CSG stands for combinatorial solid geometry. This menu item allows the user to ask Justine to not compute CSG polygons and not render the CSG images in the 3- D window. This will be described in more detail later. The default is to turn on CSG polygons to allow all user generated geometries to be viewed.

## **View**

This menu controls what is seen in the 3-D window. In addition to centering on a given selection (as selected in one of the lists), one can look down any one of the coordinate axes or edit specific angles of rotation and orientation. This menu also allows the user to specify specific field of vision parameters, in common *mm* format.

## **Button 1, 2, and 3**

These menus allow the user to redefine the default mouse actions and constrain the motion of the mouse. For example, any of the three mouse buttons (left, middle, right) can be redefined to do any of the three functions (rotate, pan, and zoom). In addition, the motion of the mouse can be constrained to cause a specific action, as in constraining the rotation to take place only in the azimuthal, elevation, or twist planes as opposed to unconstrained rotations as is the default.

In addition, the mouse buttons may be used to select objects in the 3-D window and edit, hide, or delete them. This is often useful, particularly to toggle the display of objects off, as one can merely point and click, and "peel away" layers of geometry like the skin from an onion.

## **Animation**

This menu allows the user to initiate simple animation about the x,y,z, or arbitrary line of rotation. In addition, this menu contains the "Movie" submenu which allows the user to generate and edit complicated animation sequences and splice them together smoothly for the creation of a smooth geometry animation sequence.

## **The 2-D Windows**

As can be seen in Figure 1-2, the 2-D windows also have menu bars. Once again, these are used to control the visualization options for that specific window. From Figure 1-5, we see that the image is panned inside the 2-D window using scroll bars. Each of the 2-D windows is independently resizable

and zoomable (the zooming function will be discussed momentarily). In addition, the "Status Window" is used to give the user visual clues to what actions are expected next in a sequence, and where the actions should take place. For example, if a user should request to drag out a sphere centered on the x-axis, the status window would show "SPH" in each of the applicable slices (in this case, the XZ and XY slices). In addition, the three other "status windows" show the current position of the cursor. This is very useful to exactly position geometries and gain other geometry and position information.

## **View**

This menu allows the user to manipulate how geometries are displayed in the 2-D slice window. One can center the view on the origin or selected object(s), change the depth of the slice plane, add cross hairs to the cursor, and so on. Changing the slice depth means to change the position of the plane perpendicular to the corresponding axis (e.g., the Z axis for the XY slice), thereby altering the view of the geometry in that window. Once this menu item is selected, the left mouse button is armed for changing the depth. The user then moves into one of the other slices and drags a new cut plane. As the cut plane is moved, the view changes in the corresponding slice.

Finally, one can also save the window or image to GIF or PostScript file, just as with the 3-D window from the View menu.

## **Scale**

This submenu contains a variety of preset scaling factors, by which the geometry can be enlarged or reduced. The default scale is 1/1, and all three 2-D windows have the same scaling (locked synchronization) as a default. As is discussed below, this can be changed.

In addition to the predefined scaling factors, arbitrary scaling is also allowed. Selection of this arms the mouse to allow the user to draw a small bounding box about the region of interest, which is then scaled to fill the 2-D window in its current size.

## **Objects**

This menu has a variety of powerful submenu functions that allow the user to copy, move, resize, and paste bodies, surfaces, or cells. In addition, the visualization aspects of the cell may also be controlled in this submenu. All of these functions are accessible from the Justine main window, but accessing them here is more convenient.

## **Probe**

This feature allows the user to probe the geometry for a variety of physical and computational parameters. One can examine material assignments to cells, generated mesh quantities, and a host of additional problem queries.

## **Slice Button**

On each of the 2-D windows, a button labeled "XY Slice", "XZ Slice", or "YZ Slice" appears. This button provides access to more detailed and powerful control over the 2-D windows. The overall extents

of the windows may be changed (independently), the scale may be changed, and the synchronization of zooming may be altered.

---

## Conventions

Throughout this manual, certain conventions are used To *select* something means to place the mouse over whatever is being selected, and depress the left mouse button over it. This applies to geometry objects, buttons, menus, scrollbars, and so on.

---

*Last Modified Thu Sep 7 13:35:26 MDT 1995, Stephen R. Lee*

srlee@lanl.gov

---

# Chapter 2: Quick Start

---

## Introduction

This chapter briefly describes the process for setting up and running a problem using Justine. It is intended to be a brief tutorial to get a user started right away with Justine. However, it is important to read subsequent chapters to fully understand the process.

The process steps are given below, and discussion about each step is provided. This chapter will serve to get you started right away running your problems with Justine.

It should be noted that many of Justine's detailed features including AVATAR, are beyond the scope of a quick start. These details are found in subsequent chapters.

---

## Process Steps

As you might imagine, the following are steps in the process of setting up and executing a problem using Justine:

1. Define the geometry
2. Create the cells
3. Define and assign materials
4. Set LARAMIE options
5. Execute the transport code of choice

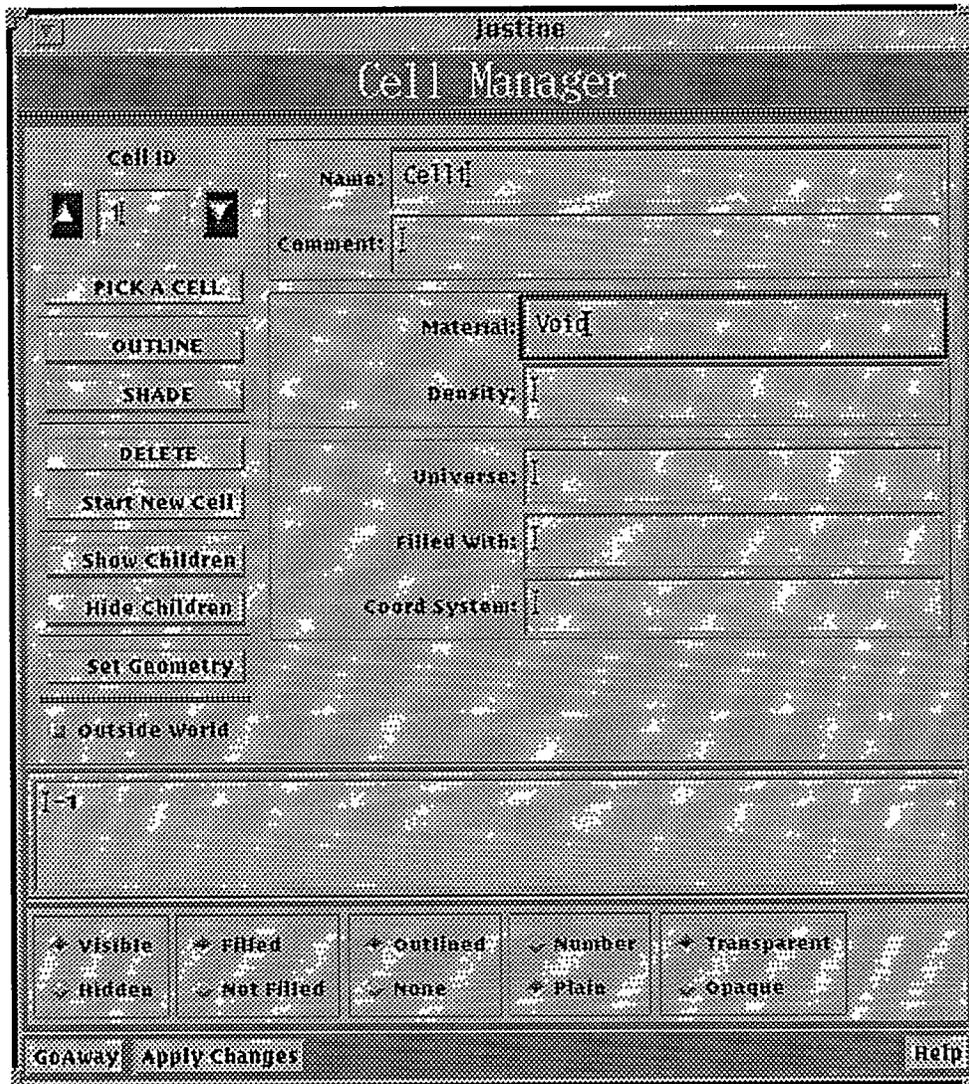


Figure 2.4: Cell manager property sheet.

## Changing the Slice Depth

It may be necessary to change the slice depth during the cell creation process. Obviously, the slice planes in their default locations may miss some of the geometry depending on the arrangement of the bodies and surfaces. Before you can pick the bodies, surfaces, and senses of the cells, *all* of the cell boundaries must be visible in at least one of the 2-D slices.

To change the depth of a given slice, select the "View" menu in the 2-D slice whose depth you wish to change. Then, in one of the other slices, drag the plane around (which is visible when you press and hold the left mouse button in one of the other slices) until the desired position is obtained. When you release the left mouse button, the new depth is in effect.

---

## Creating and Assigning Materials

The "material" step in the process is really a two step process. One must define the materials in the problem and then assign them to cells.

## Defining Materials

The definition of materials in the problem is accomplished with the material definition property sheet. Figure 2.5 shows this property sheet. The scrolling list in the upper left hand corner of the property sheet lists the elements directly from the XSDIR file in *alphabetical* order. The scrolling list to the right of that shows the individual cross section evaluations available for that element. The menu above the scrolling list allows you to select continuous, discrete, and so on, types of cross sections. The default value from the XSDIR file is always selected the first entry in the list, and only those evaluations available for the given element are selectable.

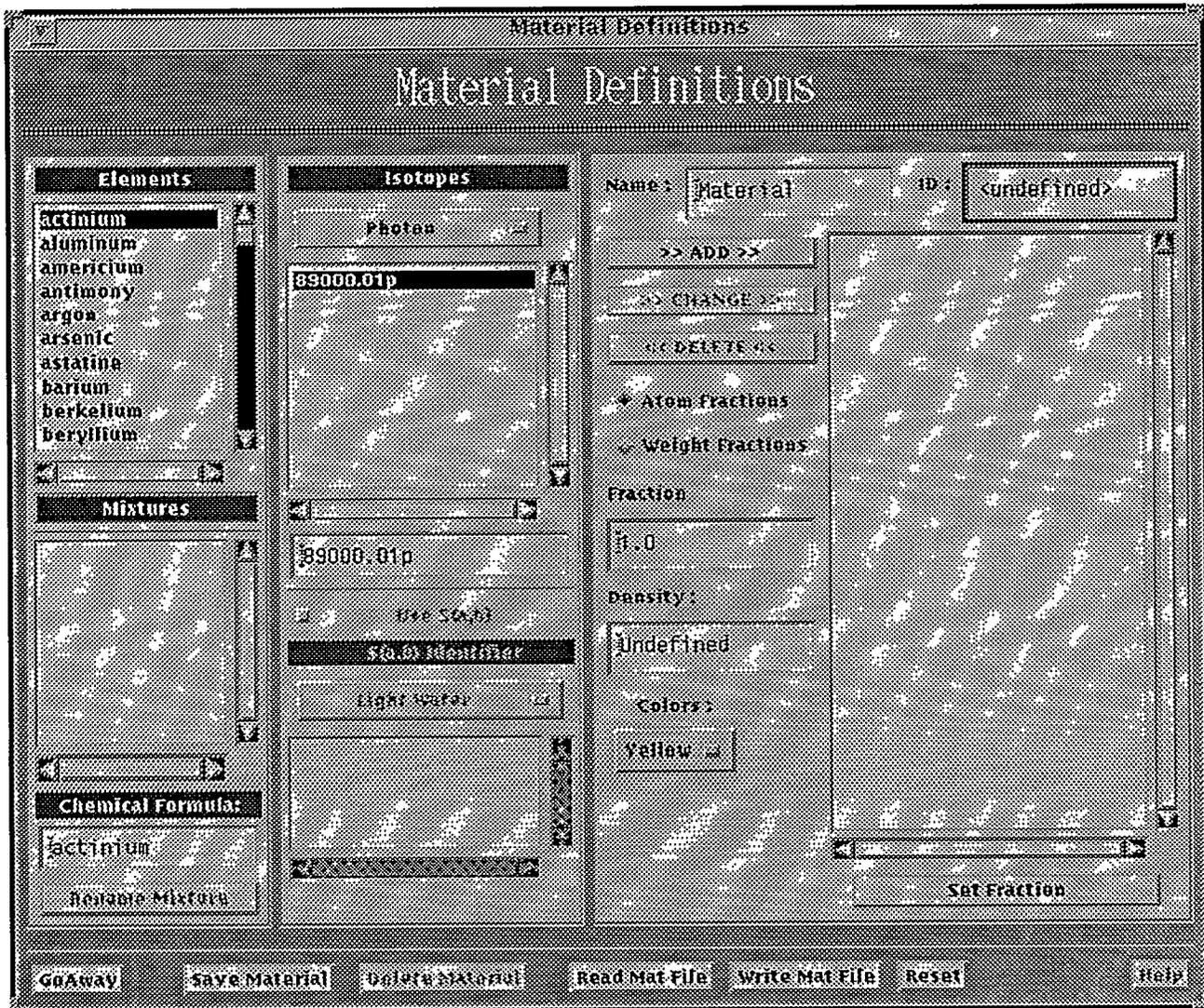


Figure 2.5: Material definition property sheet.

To create a material, select the desired element and the cross section evaluation, enter a fraction, specify weight or atomic (atomic is the default), give the material a name, and select the "Add" button. This will add the cross section and fraction to the scrolling list on the right. Continue this procedure until the

material is defined. Once complete, select the "Save Material" button and continue until all materials have been defined.

It is undoubtedly clear from the figure that a great many other things can be done with this property sheet. However, these more advanced features are expanded in another chapter.

---

## Assigning Materials to Cells

Assigning materials to cells is a very simple process. Once you have built your materials, display their names in a scrolling list by selecting "Show Material List" from the "Materials" menu item. This list will display the names of all materials you have defined. You must also list the cells you have defined by selecting "Show Cell List" from the "Cell" menu item. Once these two lists are up, simply select the material and the cell(s) to assign that material to, then select the "Assign Material" button under the material scroll list. You may select more than one cell by holding down the left mouse button and dragging the mouse over the desired cells. It is during this phase of the process that you will appreciate the nice descriptive cell names you provided Justine when you built the cells, as these names appear in the cell list and make it very easy to assign the correct materials to the correct cells.

Once you have the two lists up and have selected a material and the cell(s) to which you want to assign that material, select the "Assign Material" button on the cell list. This will bring up another popup that will ask for the density of the material in question. Enter the appropriate density and then "Apply." If you have selected more than one cell for this material, you may assign a different density to each cell by selecting "Apply," which will apply the entered density to the first selected cell, then prompt you for the density in the next cell, and so on. If you want to assign the same density to all cells with this material, simply select "Apply to All."

---

## Setting LARAMIE Options

Currently, LARAMIE options are split into several menus. The MCNP menu item on the Justine main window contains all available MCNP code options. The AVA-TAR and MESH menu items contain some setting relevant to THREEDANT. For the purposes of the quick start, only MCNP options will be discussed.

A variety of MCNP options are available. These include:

- Fixed Source
- Criticality Source
- Source Distributions
- Histories
- Tallies
- Variance Reduction
- Cutoffs
- Mode

- Physics
- Run Time Specs
- Particle Tracks

Most of the items in this menu lead to property sheets for the individual MCNP code options. In a quick start chapter like this, it is impossible to know what MCNP options should be stressed. However, the tally property sheet will be discussed in brief detail, as tallies are always needed and the other property sheets are similar in style to the tally property sheet.

### Tallies

The tally menu item actually contains a cascading menu that allows the edit of tallies as well as the creation of DXTRAN spheres and multipliers. The "Define/Edit Tallies" property sheet is shown in Figure 2.6. As an inspection of the figure reveals, most tally options are available from this property sheet. The tally type is selectable in the widget in the upper right-hand corner. Depending on the type of tally, a set of cells or surfaces may need to be selected from scrolling lists and added as single object or a union of objects to create the tally region. Energy and time bins are also available.

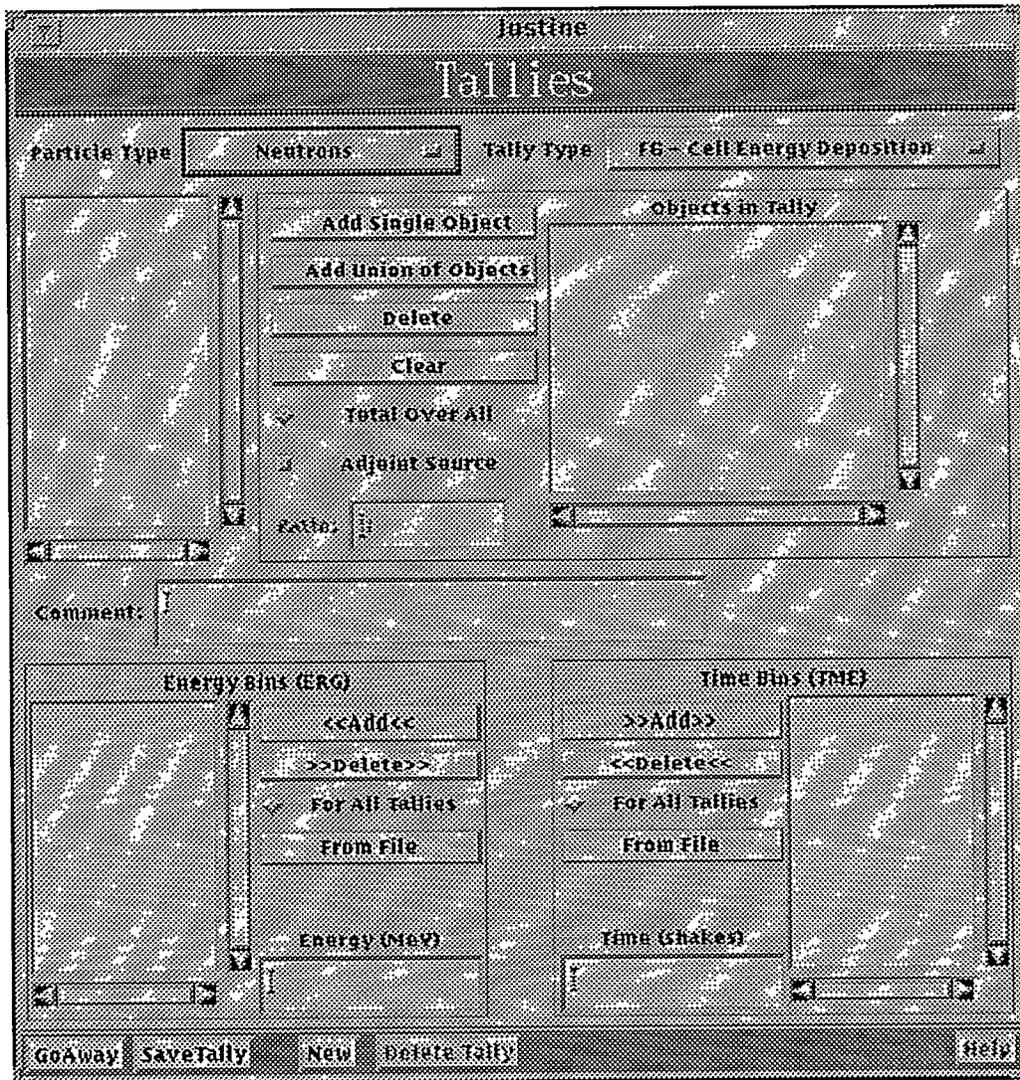


Figure 2.6: Tally edit and definition property sheet

### Additional Cards

At this time, not all MCNP data cards are available in Justine. For example, none of the MPLOTT cards have been implemented. To facilitate the complete use of MCNP from Justine, the "Type in data card" menu item under the MCNP main menu item brings up a small property sheet in which a user may simply enter the MCNP card(s) of choice. Once the remainder of the data cards and visualization features of MCNP have been incorporated into Justine, this menu item will vanish.

---

## Executing the Transport Code of Choice

You may write an MCNP input file and then execute MCNP as separate steps or do both at the selection of one item. To write an MCNP input file, select "File", then "Save As", then "MCNP" from the "File" menu on the main window. The file selection dialog box will appear, allowing you to store the file with any name in any directory. Then, you can run MCNP.

Alternatively, you can generate the file and execute MCNP by selecting "Run MCNP" from the MCNP menu on the Justine main window. If you have not already generated an input file, Justine will generate one given the current problem parameters and spawn an MCNP process.

---

## Summary

Chapter 3 and 4 discuss the different ways to realize geometries and create cells, and the different means at your disposal to visualize your geometries and cells. In Chapter 5, the material interface is discussed in detail. Chapters 6 and 7 cover LARAMIE specific options in great detail, including AVATAR. Chapter 8 highlights Justine's power features. Chapter 9 describes Justine's help package, and Chapter 10 provides the user with some tutorial examples on the use Justine.

As you can see, there are many things that have not been covered in this chapter that will greatly enhance your productivity with Justine, and exercise Justine's more powerful and subtle features. If you want to be able to use Justine to its full potential you should read the user manual in its entirety.

---

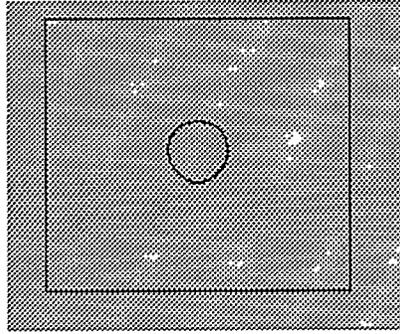
*Last Modified Thu Sep 7 13:35:26 MDT 1995, Stephen R. Lee*

srlee@lanl.gov

---

## Chapter 3: Defining Geometries

---



**Figure 3.1:** Slice down the Z-axis of a box with a cylindrical hole removed.

---

## Creating Objects

As mentioned previously, there are two ways in which object primitives can be created in Justine. A user may bring up a *property sheet* for a particular object, and enter exact quantities for various coefficients, radii, position, etc., or the user may select the object to be generated and drag the object out using the mouse in one or more of the 2-D slice windows.

### Property Sheets

Generally speaking, property sheets offer the user the most flexibility and the most power for setting up real problems. The object property sheets, regardless of the type of object, all have the same look and feel. Figure 3.2 shows the property sheet for a sphere body.

# Introduction

In this chapter, the details of creating geometries with Justine is explored. Surfaces and bodies are defined, the different ways in which to create geometries are explained, and various editing commands are explored.

---

## Bodies v.s. Surfaces

Justine is a unique geometry editor, allowing the user to generate complex geometries using both solid bodies and surfaces. A *body primitive* is any object that can be visualized as a 3-D solid object in the real world. *Primitive* means a subset of objects that can be thought of as "building blocks" for more complex geometries. An example of a body primitive is a right circular cylinder. Such a primitive has physical properties such as a finite length and radius, and can be thought of as a *quanta* of solid objects, upon which more complex objects may be constructed (like a soda can, for example).

By contrast, a *surface primitive* is a more abstract mathematical object, that may or may not exist in the real world. For example, an infinite cylindrical surface does not exist in the real world. However, it does exist as a quanta of surfaces, and can be used to construct more complex objects. For example, using two plane surfaces and one cylindrical surface (all of which are infinite in extent), one may construct the cylindrical solid object which, as a truncated 3-D object, does exist in reality. Justine allows users to build geometries using both of these object quanta.

## Surfaces or Bodies?

The decision on which type of primitive to use depends on several things. First and foremost, it depends on your mindset. For example, many users are used to thinking of geometries in the traditional MCNP sense, that is, as surfaces. So, when they need to build a "box", for example, they will construct one using six plane surfaces. However, Justine allows one to do this with one simple property sheet, or one simple drag operation. There is no need to constantly deal with a plethora of surfaces when constructing geometries.

It is important to remember that the algorithm that generates polygons for 3-D rendering in Justine is body based. Therefore, it is usually easiest to think of your geometry in terms of real objects, or bodies, that you can then translate into object primitives and construct your geometry using Justine. Surfaces should be used only when necessary to complete the geometry construct. For example, if you want to generate a geometry such as depicted in Figure 3.1, you could do so using surfaces. It would take six planes to generate the box, and one cylindrical surface to generate the hole in the box. However, it would only take two bodies to produce the same geometry (a right parallel piped and a cylinder). It is much easier to do the latter than the former and let Justine translate the bodies into surfaces automatically for MCNP.

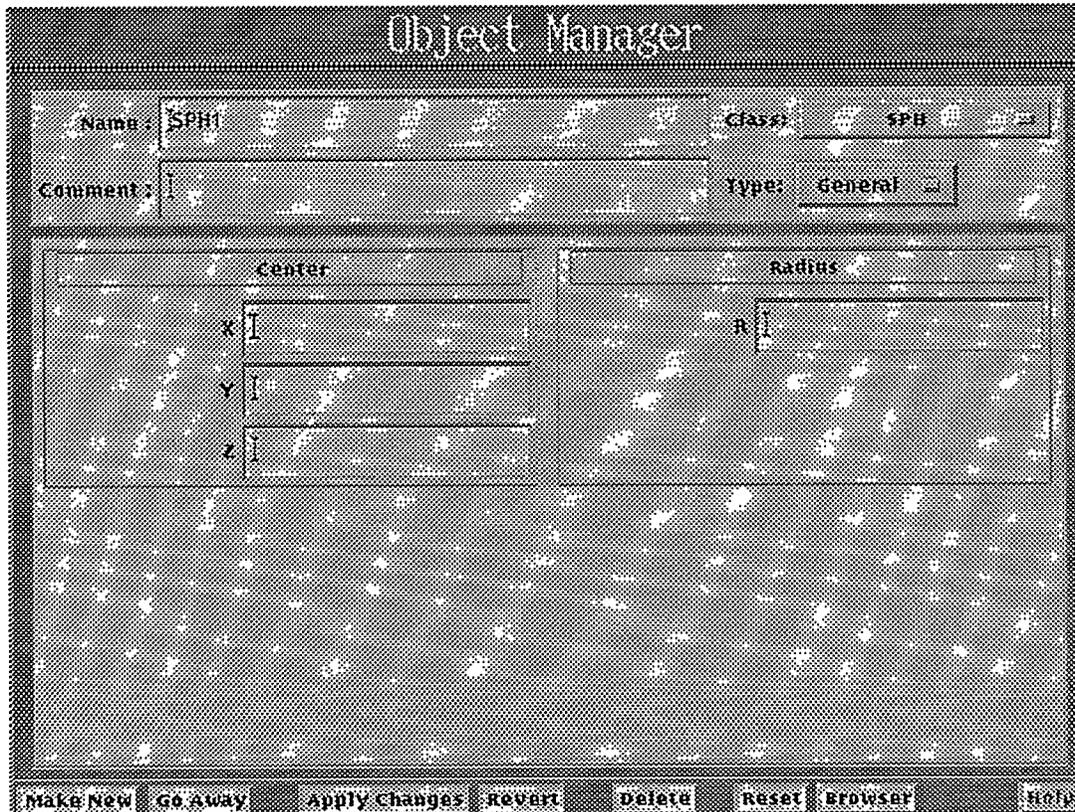
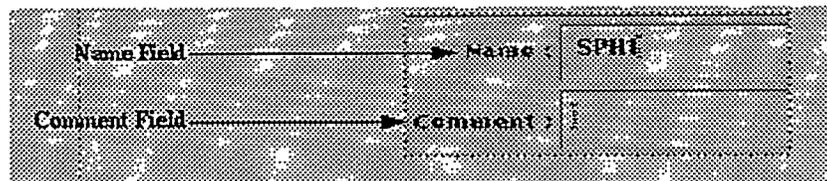


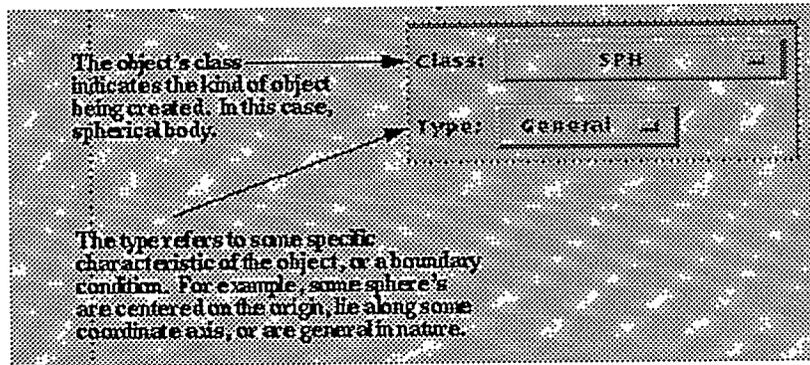
Figure 3.2: Sphere body property sheet. Note the class and type are selectable.

As can be seen from Figure 3.2, the property sheet has a variety of different components.



Justine picks a default name for the object, in this case, SPH1. This name indicates the type of object (SPH is a sphere body) and a number. As the user creates more spheres, Justine automatically increases the number so that there are no duplicate names (which are not allowed). In addition to the default name, the user is also allowed to enter any name that is desired in this field. As will become apparent later, it is often useful to give object primitives descriptive names for future reference. For example, if this geometry were part of a box with a spherical hole in the center, the user might choose to name this object "Hole." If the user does choose to use non-default names, no duplicate names are allowed. If the user does use a duplicate name, Justine will append a number to the end of the duplicate name. The object will be created, but the user will be warned of the duplication. The user is responsible for changing the altered name, if desired.

The comment field allows the user to tag this piece of the geometry with some sort of descriptive comment. It's use is optional.



In this region of the property sheet, the *class*, or kind, of object are given as well as the *type*, or limiting characteristics of the object. Each of these widgets are actually menus, and allow multiple options from which to select. The class menu selections are shown in Figure 3.3. A definition for each of the classes of objects can be found in Appendix D: Primitives.

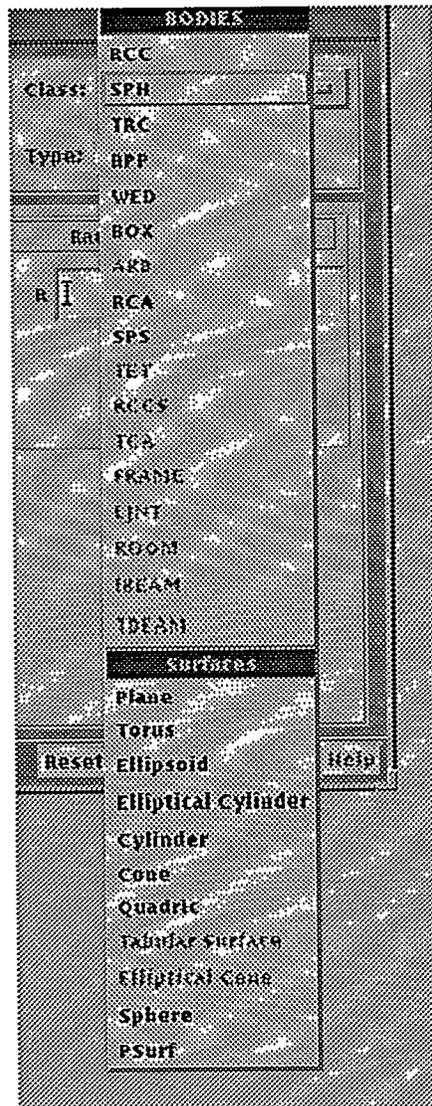


Figure 3.3: Object classes.

The type of object is a way of specifying a specific "bound" on the object within the class of the object. This menu will allow a user to constrain the creation of the object in some meaningful way. For example, for the sphere, the types of spheres allowed are general, on the origin, on the X-axis, on the Y-Axis, or on the Z-axis. However, for a cylindrical body, the constraints include all of the above plus parallel to the X, Y, or Z axis. These "degrees-of-freedom" for the different bodies or surfaces are reflected in the menu options provided to the user. That is, only those applicable to that particular object class are presented to the user in the "Type" menu.

As Figure 3.1 reveals, directly beneath these menus and fields is an area that is largely blank. In the sphere property sheet, there are two "groups" of text fields in this area, labeled "Center" and "Radius." Clearly this is where the user would enter the coordinates for the center of the general sphere, and the radius of the sphere. Since the type is "General," the user is expected to enter the coordinates for the center of the sphere, which can lie anywhere in 3-D space. However, if the type were "On Origin," only the radius field would appear since there is no need to enter the center coordinates.

It should therefore be clear that this large area is used to display the appropriate text fields for the class and type of object selected. For example, the general right circular cylinder would display text fields for the coordinates of the base, the height vector, and the radius.

As with all Justine's text fields, a blank entry indicates a NULL string or zero, depending on the type of input. Justine does include error checks so that if a user requests a sphere with zero radius (for example), Justine warns the user and provides the opportunity to correct the error. In addition, checking on the parity of input is also done. If a user enters characters where numbers are expected, Justine once again will warn the user. Finally, there are a variety of buttons at the bottom of the property sheet. These buttons will essentially "update" Justine's internal data structure. Each will be described in detail.

#### **Make New**

This button tells Justine to construct a new object based on the entries in the current property sheet. When this button is selected, Justine polls all of the data entry fields on the property sheet, checks them for any errors or inconsistencies, and constructs the object. Depending on the visualization parameters set (discussed below under "Browser") the object will be drawn in all four graphics windows in some color.

In addition, the "Make New" button will allow a user to generate copies of previously created objects. For example, if a user just generated a sphere, called SPH1, of radius 10.25 centered on the origin, and wanted another sphere of exactly the same diameter located at (100,5.2,12.5), the appropriate fields can be modified, and the user can simply select the "Make New" button to generate the object. However, remember that the name must be changed so that no duplicate object names occur.

#### **Go Away**

This button, when pressed, pops the property sheet down. If any of the changes on the property sheet have not been propagated to Justine's internal data structures (via the "Make New", "Apply Changes", or "Delete" operations), they will be lost.

#### **Apply Changes**

This has the same effect as the "Make New" button, with one important exception. This button allows a user to *modify* and *replace* an existing object. The name can be changed, the parameters altered, and so on. This button tells Justine "replace the old object with these modifications."

### Revert

This selection returns the values and state of the property sheet to the last state and values. This is useful if a user has started to edit an existing object, but decides to abort the operation and start over with the same object. "Revert" accomplishes this.

### Delete

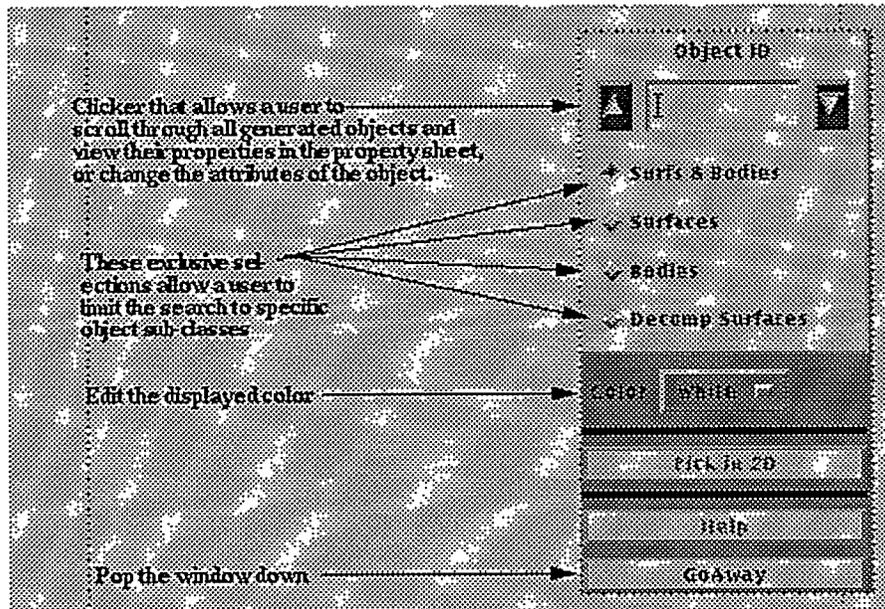
This button, when selected, will delete the object from Justine's data structure entirely. **This is not a reversible operation.**

### Reset

This button will reset the property sheet to its default state. All un-stored changes will be lost.

### Browser

The browser button pops up another window that allows the user to browse through generated geometries.



The "Object ID" is actually a number that Justine assigns to each object that the user creates. The user generally does not need to worry about this ID, as it is merely used to reference Justine's data structure to pull the information for the object into the property sheet. As a user scrolls through the object ID list (done using the up or down arrow *clickers*), the property changes to reflect all of the information

available for that particular object.

The exclusive selections will limit the search of objects based on the sub-class of the object. The default is to search for both surfaces and bodies. However, the user may search through surfaces only or bodies only by selecting one of those choices. The "Decomp surfaces" selection is a special sub-class. Since MCNP is a surface-based algorithm, all input to MCNP must be in the form of surfaces. Therefore, Justine must translate, or *decompose*, any bodies used into surfaces for use by MCNP. As body objects are generated, Justine maintains a list of surfaces that comprise each body. If a user wishes to view parameters (as generated by Justine) for these decomposition surfaces, this selection will allow it. Generally speaking, the user need not worry about such details.

## Using Property Sheets to Create Objects

As explained in the first chapter, the property sheets themselves may be "popped-up" in essentially three ways:

- By selecting the object icon on Justine's main window (see Figure 1.3);
- By selecting the object name from the "Bodies" or "Surfaces" submenu on Justine's main window;
- By changing the "Class" and "Type" menu selections on the property sheet itself.

The latter is not really a way of popping up a property sheet, but is a way to avoid popping up additional sheets once one is available.

One may also bring up multiple property sheets by selecting different icons or object names from the menus without popping previous sheets down.

Once the property sheet(s) is up, all one does to generate an object is fill in all appropriate fields in the property sheet. When the "Apply Changes" button is selected, the object is generated, polygons are created, object information is stored in Justine's data structure, and the object is drawn in all four windows. In the 2-D windows, it may or may not be visible depending on the slice depth (see below). The property sheet does not pop down at this point, since the user may wish to enter additional parameters for more objects.

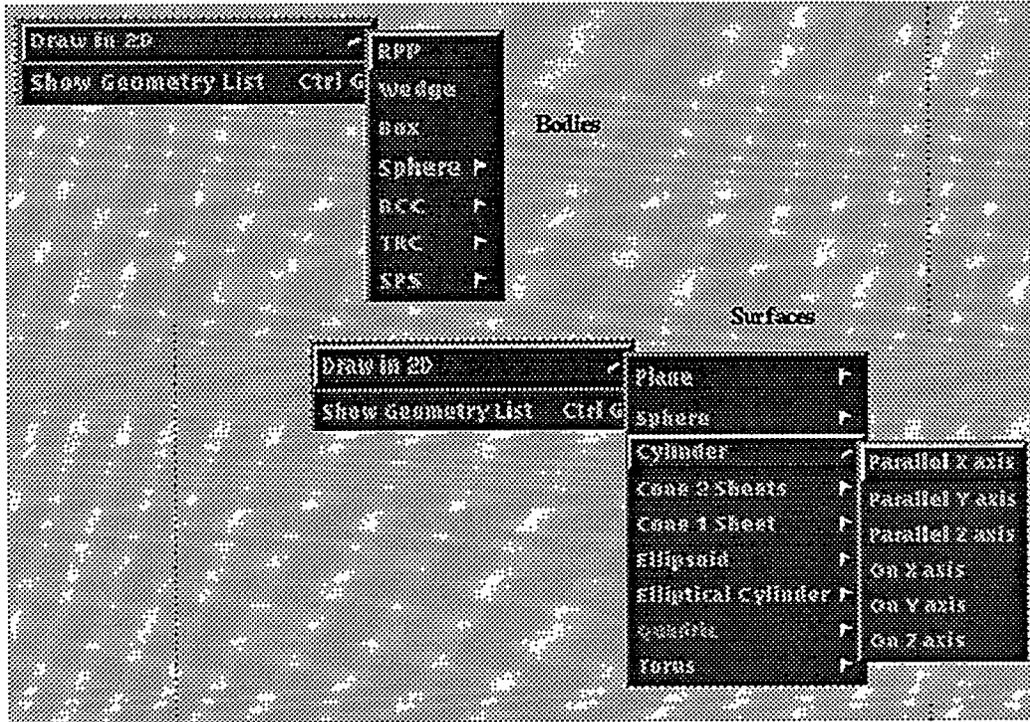
In each of the fields, you may enter exact numbers or text (depending on what is expected). Justine does check to see that actual numbers are entered where they are expected, and will warn you, with an error popup, if an error of this nature is made. Similarly, if a critical dimension or coefficient is omitted, Justine will also warn you (like a zero radius sphere). Remember that all empty fields are considered zero (if numbers are expected) or NULL strings.

## Creating Objects by Mouse Dragging

As mentioned earlier, Justine does allow the creation of objects by dragging them out with the mouse in one or more of the 2-D windows. Depending on the class of the object selected, it may be necessary to drag in more than one 2-D window.

Under the "Bodies" and "Surfaces" menu items on the Justine main window, a submenu titled "Draw in 2D" appears. This is a cascading menu that allows the user to select the class and type of the object to be

drawn.

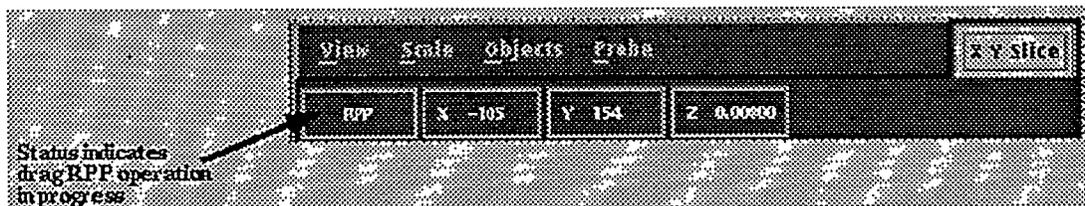


Note that in the case of surfaces, the submenu for selecting the type of surface is also shown.

Once the object class (RPP, Plane, whatever) and type (on origin, parallel to X-axis, whatever) is chosen, the mouse is then *armed* for dragging the object out in one or more of the 2-D windows.

Which 2-D window and how many windows will be required to generate the object is of course a function of the object class and type. For example, for a general sphere, a single drag in a single 2-D window will define the object. However, for an RPP, mouse operations in at least two windows will be required. To illustrate this, let's consider an example.

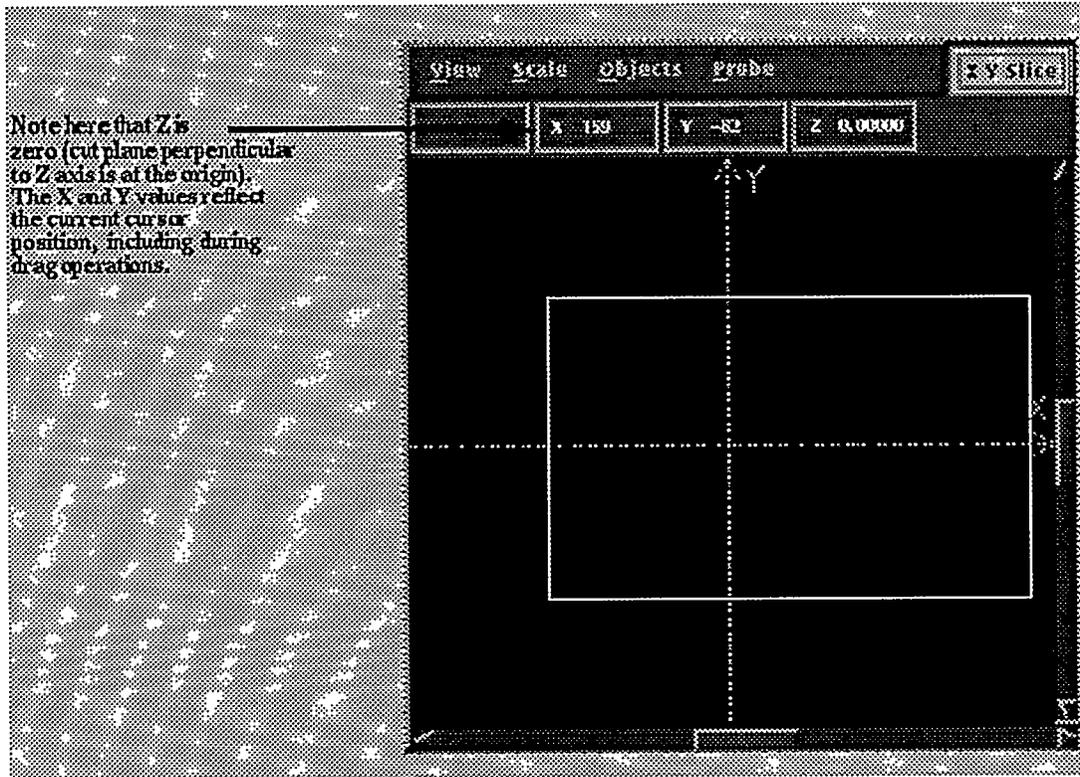
Suppose that the user has selected "RPP" from the body "Draw in 2D" menu. After such an operation, the mouse is armed for dragging and the status window in the 2-D slices all show "RPP":



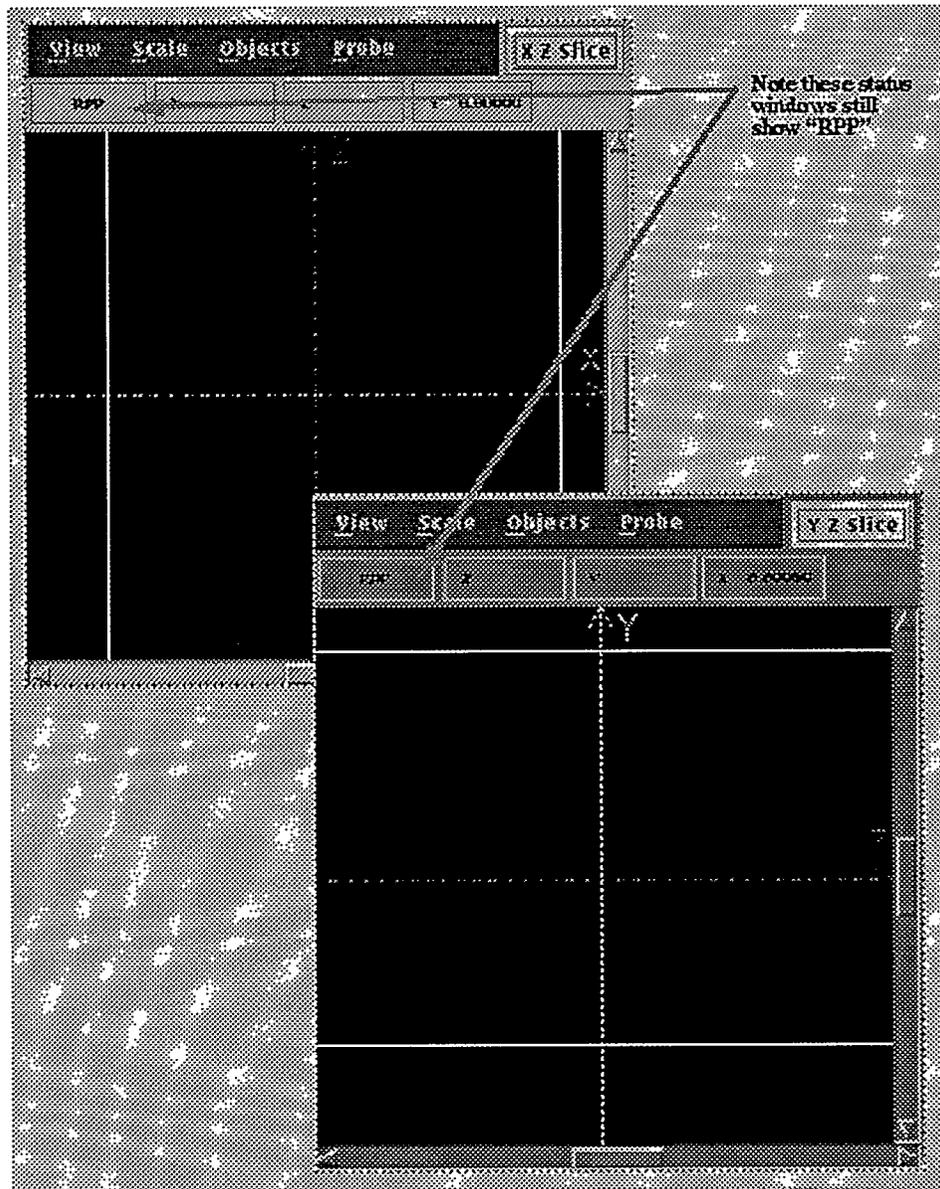
This appears in all three 2-D slice windows (although only the XY slice is shown here). Since it appears in all three windows, it is telling the user that a drag operation to generate an RPP is expected in *any* of the three windows.

Let's say that the user decided to begin the drag operation in the XY slice. To drag an object, depress the left mouse button while in the desired 2-D window and move the mouse. A dashed-line outline of the

object in question appears. The X and Y (in this example) fields above the 2-D window reflect the X and Y coordinates of the mouse as the drag occurs. Completion of this operation occurs when the left mouse button is released. Now our 2-D XY Slice Window (the top part of it) might look like this:

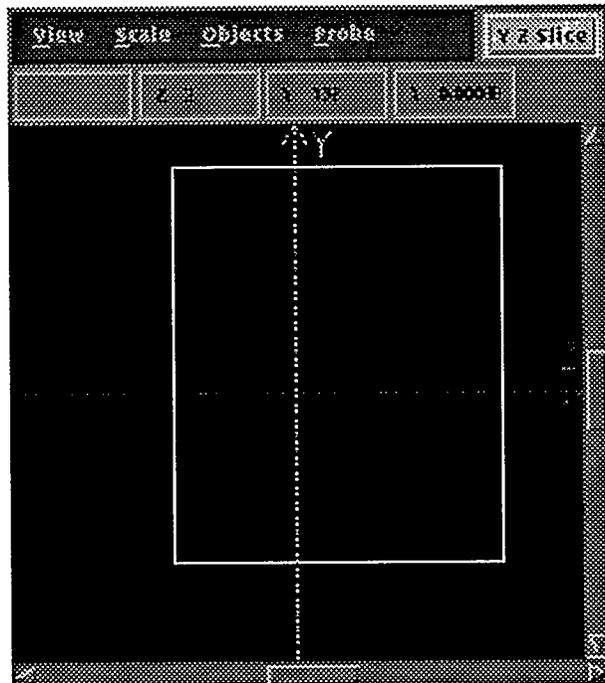


Here we see our partially completed geometry. However, the box is not complete. The extent of the box along the Z axis still needs to be defined. Our other 2-D slices look like this:



Justine is waiting for the user to complete the specification of the box with the mouse. In this case, the status windows are indicating that the user may complete the specification (in this case, the extent along the Z axis) in either the YZ or XZ slice. The operation is the same as dragging the box out, namely, using the left mouse button and dragging the "sides" of the box (which appear as a single dashed line). The box sides are placed independently.

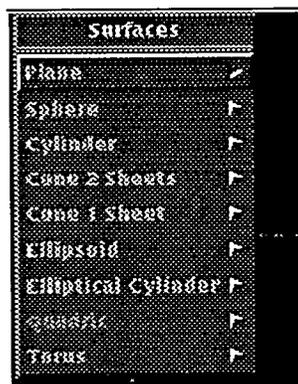
Once this is complete, the box has been completely specified:



This process is similar for all other objects. Some cases will require action in more than one window to drag and specify the object, others will not. Common sense combined with the status windows in the 2-D slices will guide you.

### Shortcuts

There are some handy shortcuts that will help you create objects with the mouse faster. On Motif 1.2 and higher systems, a pop-up menu is available directly from any of the 2-D slice windows. This menu (shown in Figure 3.4) is accessed by holding down the shift key and depressing the left mouse button with the mouse a 2-D drawing area.



**Figure 3.4:** Surface creation pop-up menu. This menu is available directly from the 2-D slice windows.

This menu allows the user to select surfaces for creation through the drag mechanism directly from the 2-D window. This saves having to go up to the Justine main window to arm the mouse for surface dragging.

In addition, there is a handy trick that allows a user to create *multiple instances* of the *same type* and

class of object without having to repeatedly access the menu at all. If, for example, multiple spheres in different locations were desired, one could select the general sphere surface from the above menu, then drag the spheres out with the *middle* mouse button. Using this button keeps the mouse armed for surface dragging. However, it is important to remember to drag the *last* sphere out with the *left* mouse button, or unexpected results may occur. In addition, Motif 1.2 has a well known bug which causes the Justine client to hang if the right mouse button is used applied in the 2-D windows. Therefore, you should avoid that error.

### Why Drag?

Dragging geometries out offers a quick, convenient way to create object primitives. However, the downside is that there is no convenient way to get exact quantities (as in a sphere of radius 127.317). Some users prefer to drag out rough geometric components, then go back with the property sheet and the browser and enter exact quantities. Ultimately, the choice is left to the user. Both capabilities are available.

---

## Visualizing Geometries

In this section, the details of geometry visualization are explained. Most of these concepts were introduced in the first two chapters, but not in significant detail. Here, the details of 2-D and 3-D visualization are presented.

### Three-Dimensional Visualization

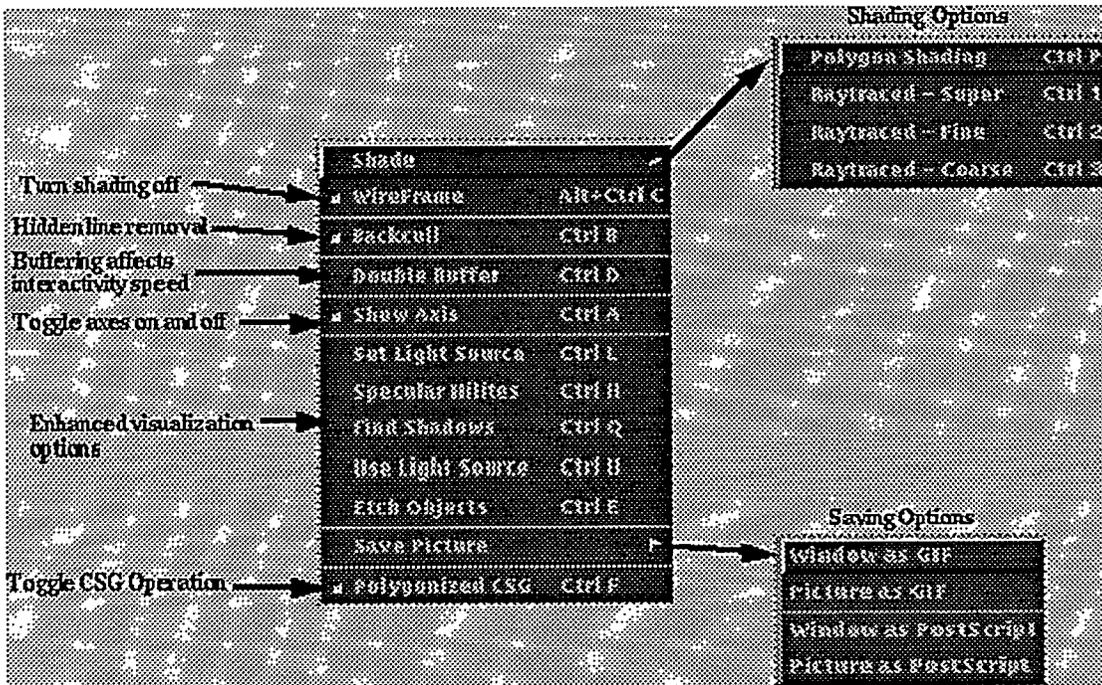
The 3-D window offers a variety of visualization options to the Justine user which are available from the menu bar at the top of the window.



These individual menu items will now be discussed in more detail.

#### **Drawing** Drawing

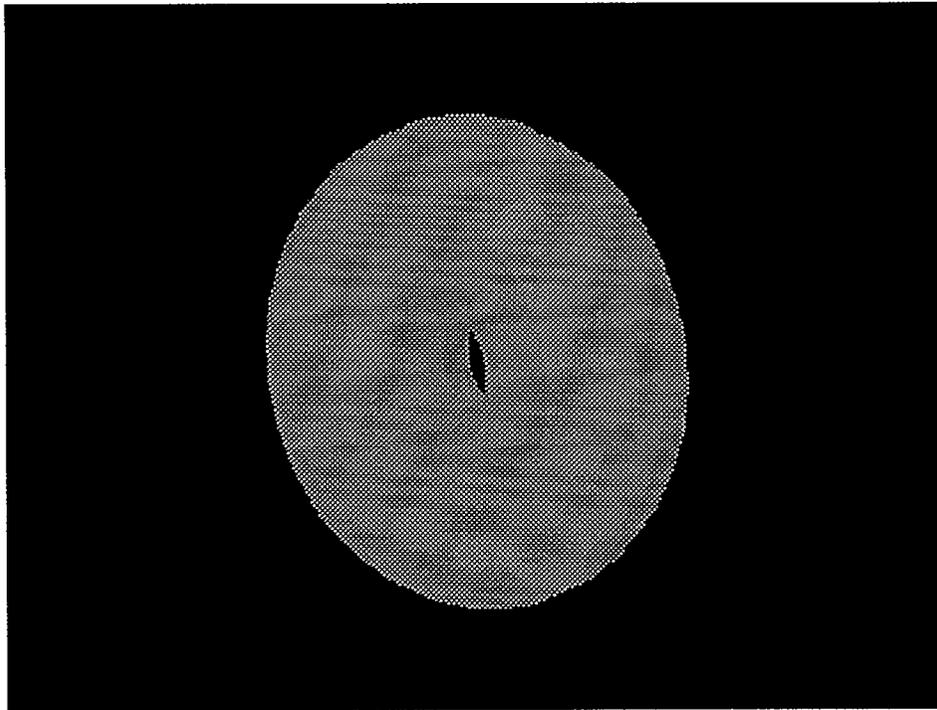
The drawing menu has the following options:



Justine's default values are shown. That is, by default, Justine has 3-D shading disabled, backculling on, double buffering disabled, axes visible, and polygonized combinatorial solid geometry enabled.

### Shading/Wireframe

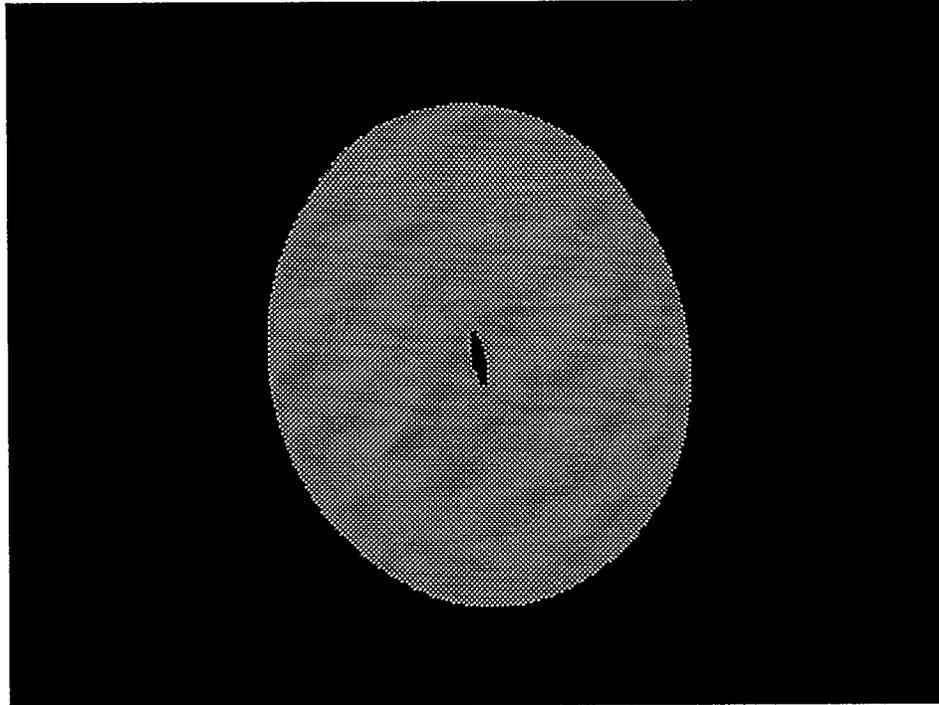
When wireframe is selected (as shown above, which is the Justine default), all geometries are rendered in wireframe mode in 3-D. Chapter 1 discusses the difference between wireframe and polygon rendering. In addition to rendering with polygons, three ray-tracing options are also provided. Ray tracing has the advantage of rendering a more realistic looking objects, devoid of the facetting that is seen as a result of the polygonalization process, but has the disadvantage of being slower. As a result, three different resolutions, or "granularity", of ray tracing are provided. Figures 3.5 - 3.8 show four shaded images of a toroidal surface. Figure 3.5 is a shaded polygonal representation, and Figures 3.6-3.8 are ray-traced images, each traced with the three different ray-tracing resolutions (best to worst). As can be seen, the lower the resolution, the more edge blurring occurs. However, this object represents a worst case scenario.



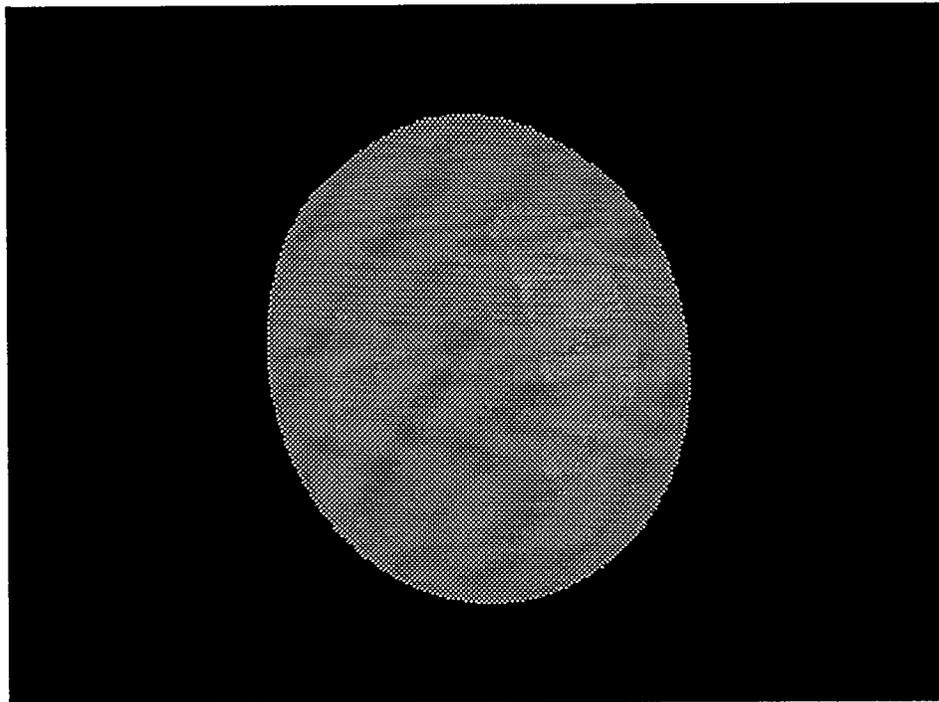
**Figure 3.5:** Polygonal shading of a toroidal surface.



**Figure 3.6:** Ray-traced rendering of a toroidal surface. Shown is the highest resolution mode ("Super").

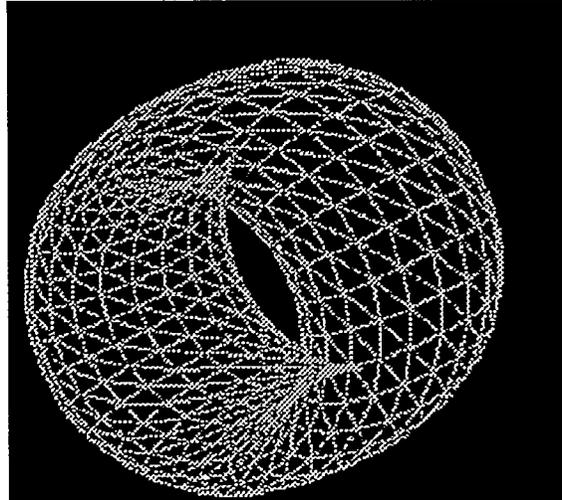


**Figure 3.7:** Ray-traced rendering of a toroidal surface. Shown is medium resolution mode ("Fine"). Note the increase in edge blurring.



**Figure 3.8:** Ray-traced rendering of a toroidal surface. Shown is low resolution mode ("Coarse"). Note the edges are very difficult to see.

Polygonal shading is always faster than ray tracing and, generally speaking, is more than adequate for visualization needs. Ray tracing is often used to generate higher-quality images for presentations and papers. The same object in wireframe mode is shown below in Figure 3.9.

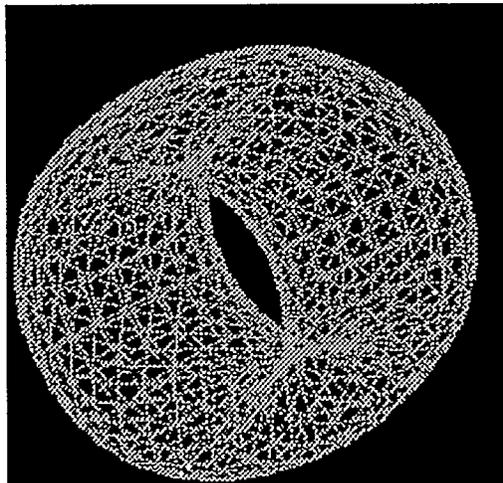


**Figure 3.9:** Wireframe rendering of a toroidal surface. In this image, the edges of the polygons are rendered, with backcull enabled.

The 3-D image always reverts to wireframe during rotation, zoom, and translation operations.

#### *Backcull/No Culling*

Justine has the ability to do some hidden line removal in the wireframe polygon rendered images. Justine accomplishes this by "culling" or not displaying any polygons face away from the viewport. This has the effect of enhancing the depth cue of the object, which is why it is the default in Justine. However, one can disable this. In the image above, Culling is enabled. In Figure 3.10, no culling is done. Note the increase in clutter in the image, and some loss of depth cue information as compared to Figure 3.9.



**Figure 3.10:** Wireframe rendering of a toroidal surface. Here, backculling is disabled.

#### *Buffering*

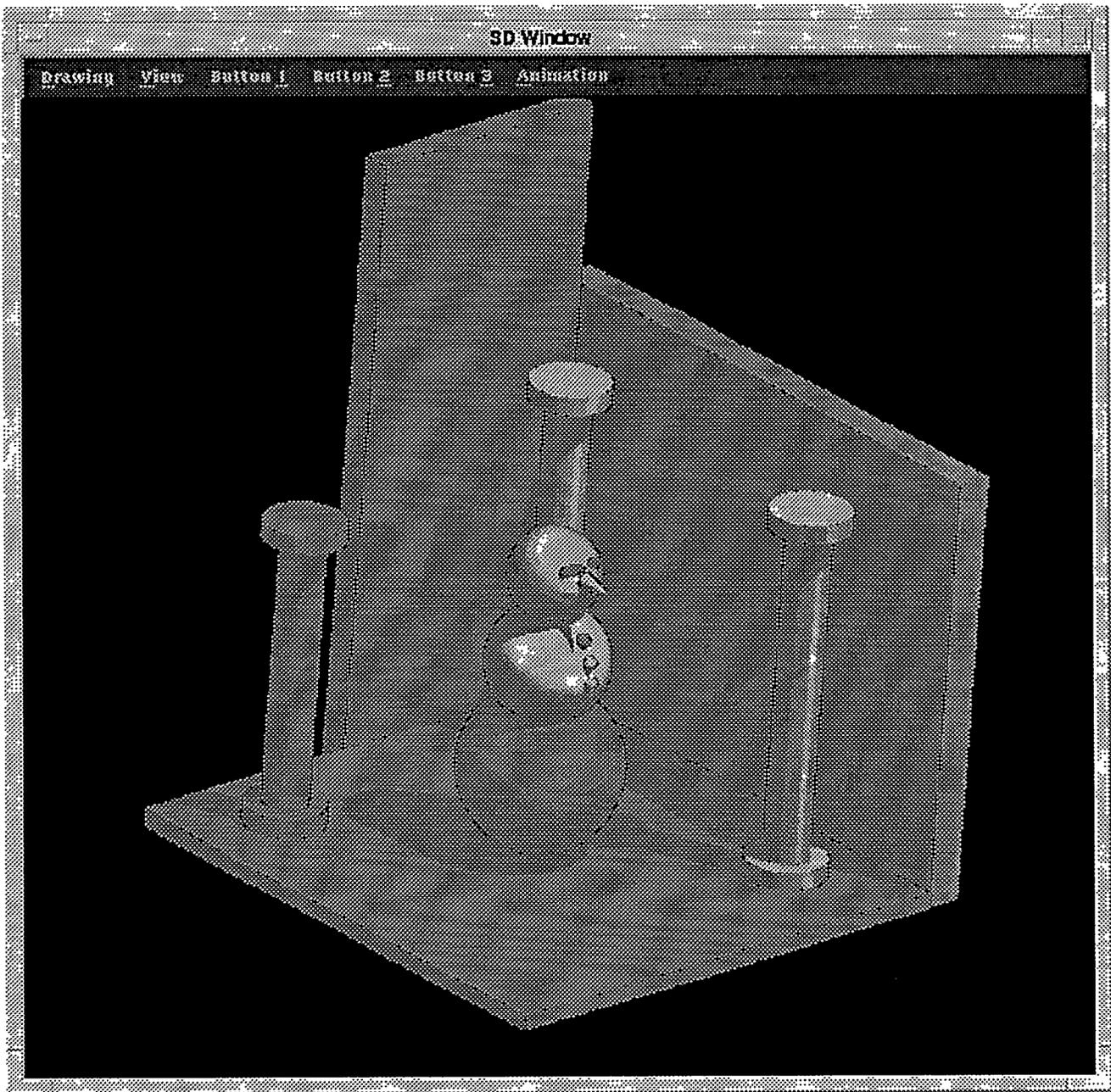
Double buffering is a technique that allows the storage of the next buffered image as the current buffered image is cleared. This makes for a smooth transition from one image to another as an object is

rotated (for example). When double buffering is disabled (Justine's default), the rotation is more interactive (i.e., faster), but the image flashes during the operation due to the fact that the screen is being refreshed directly, as the image is rotated, without regard for clearing the old one first.

Due to interactivity speed, double buffering is disabled by default in Justine. However, for certain situations, such as animation, the user may wish to enable it. In addition, some platforms are much faster than others. If the image seems overly jumpy, try enabling double buffering and see if the interactivity of the platform is sufficient to allow the use of the feature.

### *Light Sources, Shadows, Etching*

These advanced features apply to ray-traced images in Justine. They allow you to set sources for lighting, include shadowing effects, and etch the outlines of rendered objects. All of these things contribute to the realism of the rendered images. Unfortunately, they all also increase the rendering time. Figure 3.11 shows an image with a light source set and shadowing and etching enabled.



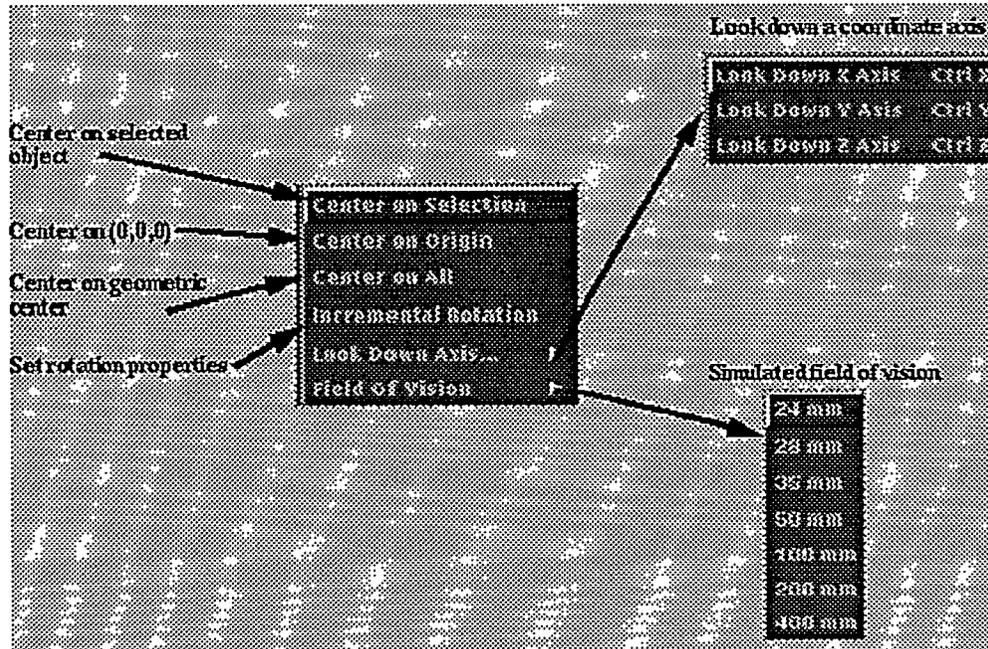
**Figure 3.11:** Ray-traced image with shadows, etching, and light source set.

### *Polygonized CSG*

This option refers to the algorithm that Justine uses to combine solid objects (and surfaces). This is loosely referred to as combinatorial solid geometry or CSG. When one makes cells, which is the topic of the next chapter, one combines primitive objects to form more complex objects. Justine breaks these complex sets into binary operations on pairs of primitives, such as box minus cylinder. In doing so, for rendering purposes, additional CSG polygons are generated for the new object (such as a box with a cylindrical hole in it). This can be a time consuming process. If this is disabled, no 3-D image of generated cells will be made, thereby negating the value of Justine's three-dimensional viewing features. By default, Justine generates the CSG polygons.

## View

The view menu has the following options, each of which affect the visualization of the geometry.



### *Centering/Looks*

All of these options center the viewport on something, or orient the coordinate axes in some particular direction. Centering on the origin does just that, regardless of the current position of the viewport center with respect to the rest of the geometry, when this is selected, the viewport center and the origin of the coordinate axes are immediately made to coincide. Similarly with the geometric center and the viewport center when "Center on All" is selected.

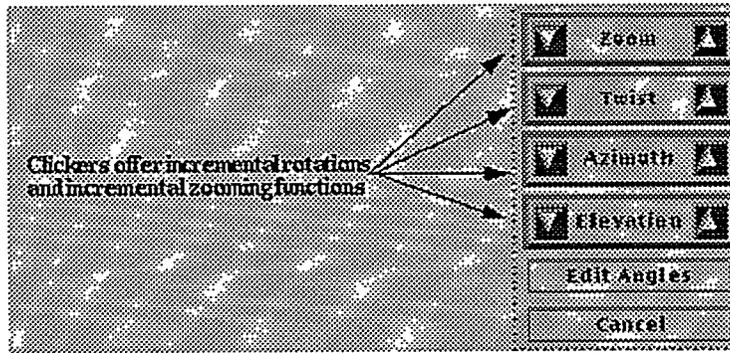
Looking down one of the coordinate axis provides a handy mechanism for immediately rotating the 3-D view to look similar to one of the 2-D slices, without having to go through the mouse manipulations to get there.

Centering on selection allows one to center the view on a particularly selected object or cell. Selection of the object or cell is made from one of the scrolling lists that is discussed below.

Finally, selecting one of the field of vision values will simulate that particular field of vision with respect to the geometry. These are roughly equivalent to photographic values and provide quick, handy mechanisms for zooming in and out.

### *Incremental Rotation*

This menu item pops up another property sheet, shown below.

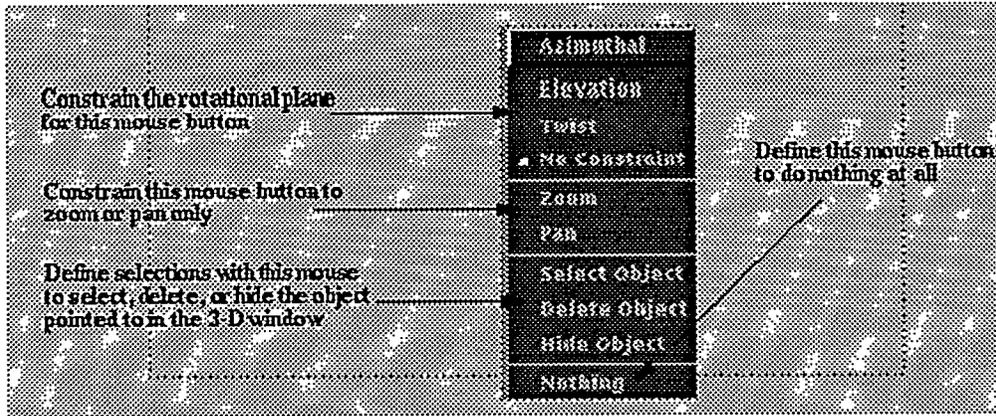


With the mouse, smooth, controllable rotations are provided. However, it is sometimes useful to step through small angles of rotation or zooming factors piecemeal as well. These clickers, when selected, will rotate the geometry about different viewport axes or zoom the geometry by small increments with each successive click. The increments are selectable through the "Edit Angles" button, which brings up the property sheet shown below.

Zoom:	Twist:	Azimuth:	Elevation:
<input type="text" value="1"/>	<input type="text" value="175"/>	<input type="text" value="0"/>	<input type="text" value="14"/>
<input type="text" value="1"/>	<input type="text" value="1"/>	<input type="text" value="0"/>	<input type="text" value="1"/>
<input type="text" value="0.00531587"/>	<input type="text" value="0.121759"/>	<input type="text" value="0.991601"/>	<input type="text" value="0.0432943"/>
<input type="checkbox"/> X	<input type="checkbox"/> Y	<input type="checkbox"/> Z	<input type="checkbox"/> R
<input type="button" value="Apply"/>	<input type="button" value="Cancel"/>	<input type="button" value="Edit Increment"/>	

Using this sheet and the dial buttons (or the text fields), one can set explicit values for the rotations and zooms. The definitions for azimuth, twist, and elevation are given in the next section.

**Button 1** **Button 2** **Button 3** **Button 1, Button 2, and Button 3** The purpose of these menus is to allow the user to alter the function of the mouse buttons as defined by Justine. For each button, these menu items are identical:



### Default Mouse Settings

By default, the mouse is designed as follows in Justine:

Table1: Default Mouse Definitions

Mouse Button/Action	Function
Hold Left and Drag	Arbitrary Rotation
Hold Middle and Drag	Arbitrary Pan
Hold Right and Drag	Zoom

However, Justine allows the user to redefine any or all of the mouse buttons using the above menu. The menu is identical for each button, owing to the ability to assign any of the functions to any of the mouse buttons.

### Constraining Rotations

The default setting for the left mouse button is to not constrain the rotation, that is, to allow free rotations within the "arcball" implemented rotational algorithm. However, the user may constrain the plane of rotation to be along one of the *viewport rotational axes* by selecting one of these options. The viewport axes are a set of orthogonal axes in the user's reference frame.

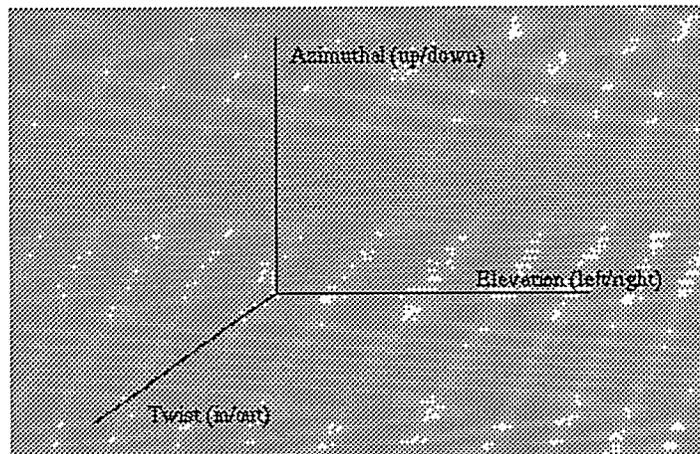


Figure 3.12: Viewport coordinate axes.

Figure 3.12 depicts the fixed coordinate system in the user's reference frame that serve as the viewport coordinates. Constraining mouse rotations to be in the azimuthal plane means that all rotations of the geometry are made about the azimuthal axis above. The azimuthal axis runs in the up/down direction in the user's reference frame. When this constraint is applied, all mouse motion with the button pressed that has been assigned this constraint rotates the geometry about this axis. Similarly for the other axes.

### *Unconstrained Rotations*

Justine uses the Arcball[REF] technique to accomplish unconstrained rotation. This is an implementation of mouse rotations that applies mathematical theory to the interface rotations. Since mouse motion has but two degrees of freedom, position pairs are used as the ends of an arc along some arbitrary spherical surface. As the mouse is moved, its position is used to compute the arc. This in turn causes the object to move with the mouse along the computed arc. The result is an intuitive rotational scheme in which users may effectively drag different parts of the geometry to different points in the viewport, and easily rotate and move the image around in a straightforward way.

Whenever the mouse button is depressed that has been defined to be the rotation operation (constrained or unconstrained), the mouse pointer reverts to a double arrow, serving to remind the user of the functional mode that the mouse is in.

### *Constraining Action*

The remainder of the options in the menu deal with constraining the action of the mouse. Panning is accomplished by holding down the middle mouse button and moving the mouse. The geometry follows the mouse motion intuitively. With zooming, motion to the right while holding down the right mouse button makes the image larger, motion to the left makes the image smaller.

As with rotations, the mouse pointer reverts to different bitmap images when selected for zooming and panning. For zooming, the mouse pointer reverts to a small magnifying glass. For panning, the mouse refers to two double headed crossing arrows.

In addition to altering the mouse button functions so that any of them can rotate, pan, or zoom, one can also increase the flexibility of Justine considerably and enhance the action of the mouse by defining one or more of the mouse buttons to delete an object, hide an object, or select an object. In doing so, a user may point to a 3-D object on the window, select it, and Justine will take appropriate action on that selected object. To determine which object is being indicated in 3-D, a ray trace is done from the clicked point in the viewport to the first 3-D surface that is encountered. If the mouse action is to delete the object, once the mouse is clicked, that object is removed from Justine's internal data structures (this is not a reversible operation). If the mouse action is to hide the object, the object is hidden as soon as it is selected. This is a nice feature, as it allows a user to "peel away" outer layers of a geometry, as one might do with the skin of an onion. Finally, if the mouse action is to select the object, that object is selected for some other action once the mouse is clicked (other action might include making it a cell, editing its properties, and so on).

For selection, the mouse pointer reverts to an "eye" icon. For deletion, the mouse pointer reverts to an "X", and for hiding, the mouse pointer reverts to a small chicken icon.

Finally, one can redefine individual mouse buttons to have no action whatsoever. For this definition, when the mouse button is selected nothing happens, other than the mouse pointer reverts to a "skull and crossbones" icon.

## Animation

Justine offers a variety of animation options.



Animation about the X, Y, or Z axis is simply animation about the elevation, azimuthal, or twist viewport axes. That is, the geometry is rotated about one of these visual axes. Custom animation refers to animation about an arbitrary axis specified by the user (through the origin). When this is chosen, the user is expected to drag a rotational axis out with the left mouse button in the 3-D window. As the user drags the axis, a dashed line appears. When the user releases the mouse, signifying the completed definition of the axis, the line disappears and the geometry begins to rotate about this axis.

In general, when animating, it is best to enable double buffering (as described above). This ensures a smooth animation sequence. However, with double buffering enabled, the dashed line defining the axis for custom animation will not be seen when the user drags it out. This is a result of the double buffering algorithm.

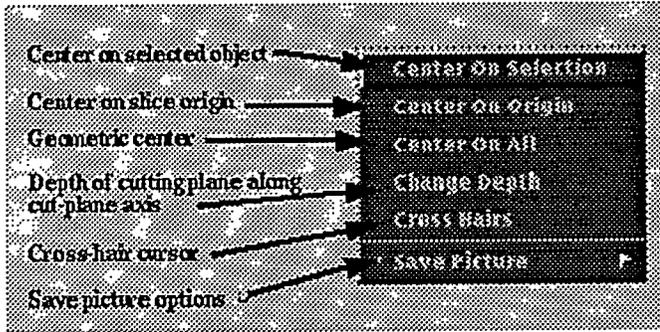
The generation of movies with Justine is covered in Chapter 8, Justine Power Features.

## Two-Dimensional Visualization

As with the 3-D window, the 2-D windows offer a variety of visualization options to the Justine user. The 2-D windows have a menu bar, status fields, and a button to specify specific options for that particular 2-D window. Each of these items will now be discussed in more detail.

## View

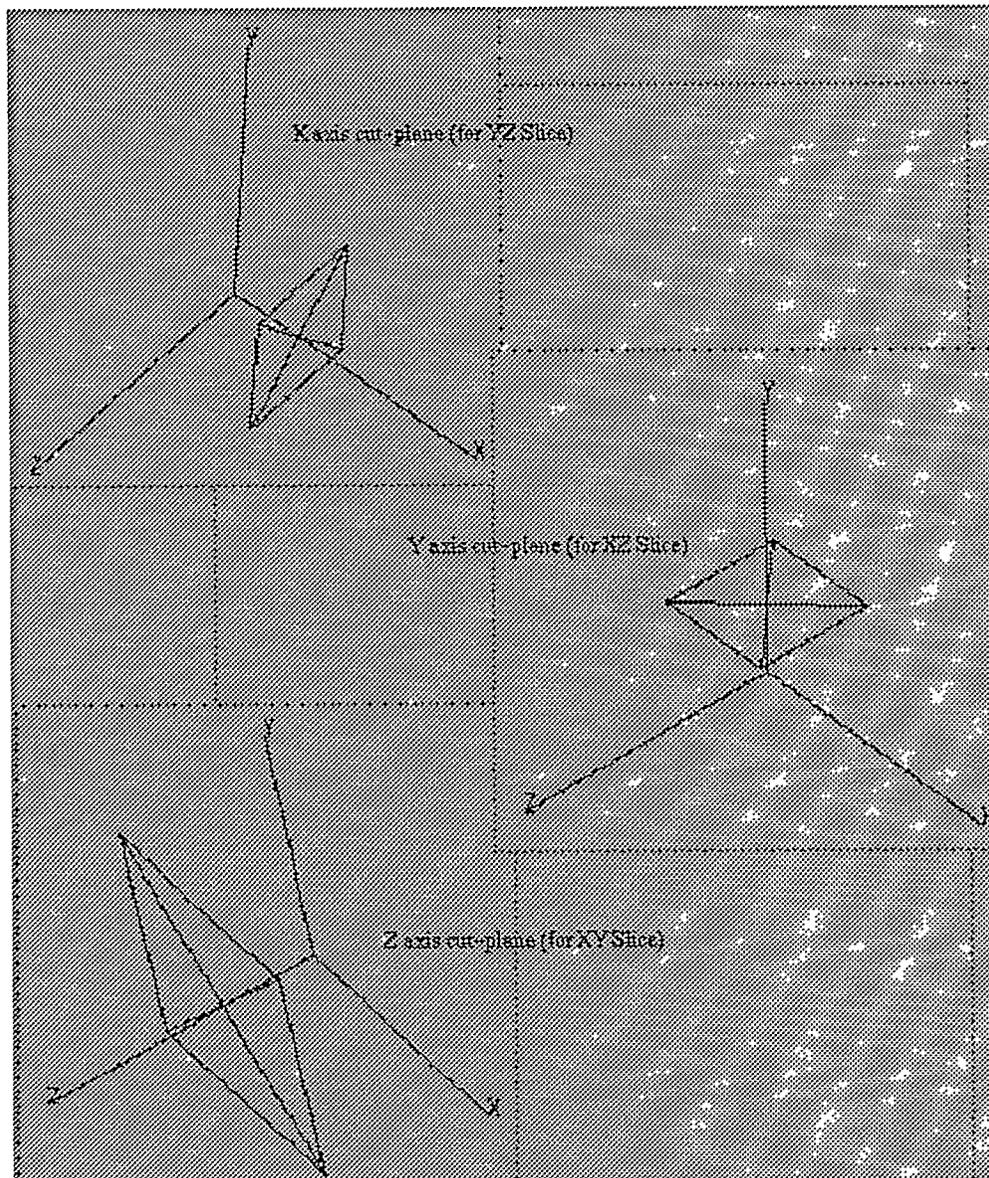
As in the 3-D window, the view menu options affect the visualization of the geometry.



The 2-D centering functions work identically to their 3-D counterparts. Similarly for the save picture options.

### *Change Depth*

Each of the 2-D windows is a *slice* in one of the coordinate planes. That is, in each window, the geometry is drawn as the intersection of the *cut-plane* perpendicular to one of the coordinate axes and the 3-D geometry. The "XY Slice" renders the intersection of the geometry with a cut-plane perpendicular to the Z axis, the "XZ Slice" renders the intersection of the geometry with a cut-plane perpendicular to the Y axis, and the "YZ Slice" renders the intersection of the geometry with a cut-plane perpendicular to the X axis. Figure 3.13 illustrates each of the cut planes and their relationship to the coordinate axes.



**Figure 3.13:** Cut planes and their relationship to the coordinate axes and the slice windows.

From Figure 3.13, we see that by changing the "slice depth" in one of the 2-D views we are really moving the appropriate cut-plane along one of the coordinate axis. Changing the depth is often required in order to visualize different parts of the geometry located in different places along the cut-plane axis.

To change the slice depth in one of the slice views, select the slice depth menu item in that slice. Then, move to one of the other slices and drag, with the left mouse button, the plane to a new location. The user is prompted for this action in the status window of the other two slice views. If, for example, a user selected "Change Depth" in the XY Slice, the status window of the other two windows would both say "XYSlice" indicating that in either of those two windows, the mouse is armed to alter the depth of the slicing plane along the Z axis (in this example). As the user drags the plane in one of the other windows, a dashed line representing the cut plane moves along with the mouse. The coordinates for the plane's position along the cut plane axis is also displayed. In addition, as the plane is moved, the window in to which the depth applies view changes, as the plane intersects the geometry in different areas.

## Cross Hairs

This simple option replaces the default mouse pointer in the 2-D windows, which is the system default mouse arrow for most functions, with a cross hair that covers the entire slice view from top to bottom and left to right. This allows for accurate placement of objects and determining extent of cells, among other things.

## Scale

This menu item allows the user to select from a variety of zooming factors, to reduce or enlarge the geometry, or to enlarge the geometry bounded by an arbitrary box. This menu is shown below:

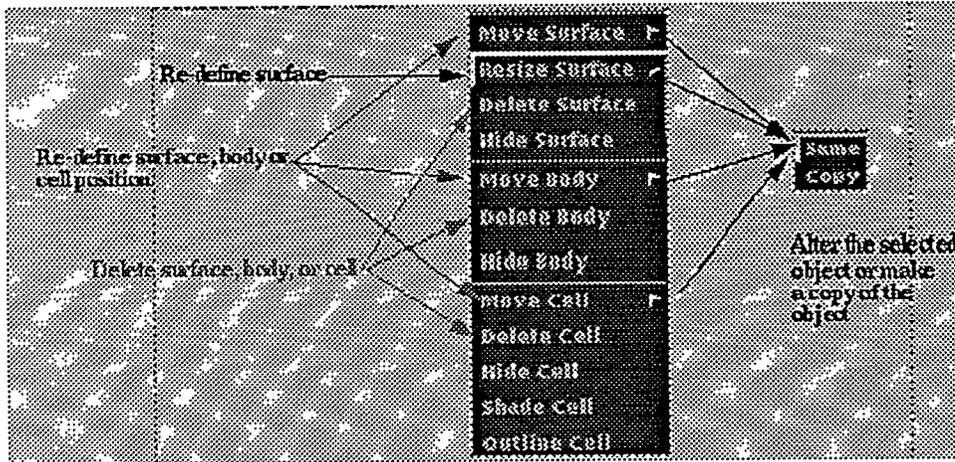
LARGER	
1/1	Alt 1
2/1	Alt 2
4/1	Alt 4
6/1	Alt 6
8/1	Alt 8
12/1	Alt 12
50/1	Alt 5
SMALLER	
1/2	Ctrl 2
1/4	Ctrl 4
1/6	Ctrl 6
1/8	Ctrl 8
1/12	Ctrl 12
Arbitrary	

The numbers indicate an approximate scaling factor to enlarge or reduce the image. When one of these is selected, the mouse is armed to select the new viewport center for viewing with the new scale. Once the mouse is positioned at the desired point in the geometry, and the left mouse button is clicked, the new scale will take effect with that point at the center of the viewport.

Similarly, the "Arbitrary" menu selection allows a user to draw an arbitrary box around some portion of the geometry. When this is selected, the mouse is armed to drag a box. As the left mouse button is held down and the mouse is moved, a dashed outline of the box appears in the 2-D window. When the mouse is released, scaling occurs based on the extent of the box drawn.

## Objects

This menu item provides a variety of geometry and cell editing and visualization features in the 2-D window.



### *Moving Objects*

One can move surfaces or bodies (moving cells is not yet implemented) in the 2-D windows by selecting the appropriate "Move" class, then moving the mouse into the 2-D window, clicking and holding the left mouse button on the desired object, and by dragging it to its new location. One can move the *same* object, meaning just translate it from one place to another, or once can move a *copy* of the object, meaning to leave the old one where it is, make a copy of it, and move that one to a new location. This provides a fast mechanism for making a copy of an object and placing it somewhere without having to use the property sheets.

### *Resizing Surfaces*

Selecting this option will arm the mouse to resize the surface selected. Again, a copy operation may be done. One can only resize surfaces at this time. No cell or body resizing is allowed. As with most operations, resizing the surface occurs when the user depresses and holds the left mouse button over the desired surface, and drags the new one out. During the drag operation, the old surface is rendered unchanged, but the new outline is shown in dashed line form.

### *Deleting*

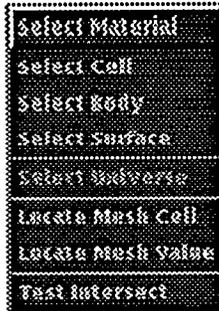
As expected, this option arms the mouse for removing the body, surface, or cell from the Justine data structure. **Once selected and deleted, this operation is not reversible.**

### *Hide, Shade, Outline Cells*

This menu controls the visualization of generated cells. Selecting any of these and then selecting the desired cell will have the stated visualization effect on the cell in all three cut-pane views.

### **Probe** Probe

The probe menu item allows the user to query the geometry, mesh, and problem space for a variety of physical and computational attributes. The menu is shown below:



### *Select Material, Cell, Body, Surface*

These options will highlight and provide information on either the material selected, the cell, body, or surface selected. In the case of the latter three, this can be used in conjunction with the property sheets to retrieve complete information on the object. For materials, the material selected (if any) is reported to the script window (i.e., the window in which Justine was run).

### *Locate Mesh Cell, Mesh Value*

When these options are selected, the value of the mesh or the cell contained in a particular mesh cell is reported to the script window when selected.

### *Test Intersect*

This option will report all surfaces intersected from the point on the slice window where the mouse was selected to a point on the other side of the window (in a straight line).



### **2-D Slice Parameters**

In all three windows, a button appears in the upper right-hand corner of the window, and is labeled "XY Slice", "XZ Slice", or "YZ Slice" depending on the cut-plane window viewed. This button brings up a property sheet pertaining to that window only that allows the controlling of various slice parameters. The features provided in the popup property sheet are explained in detail in Chapter 8, "Justine Power Features."

---

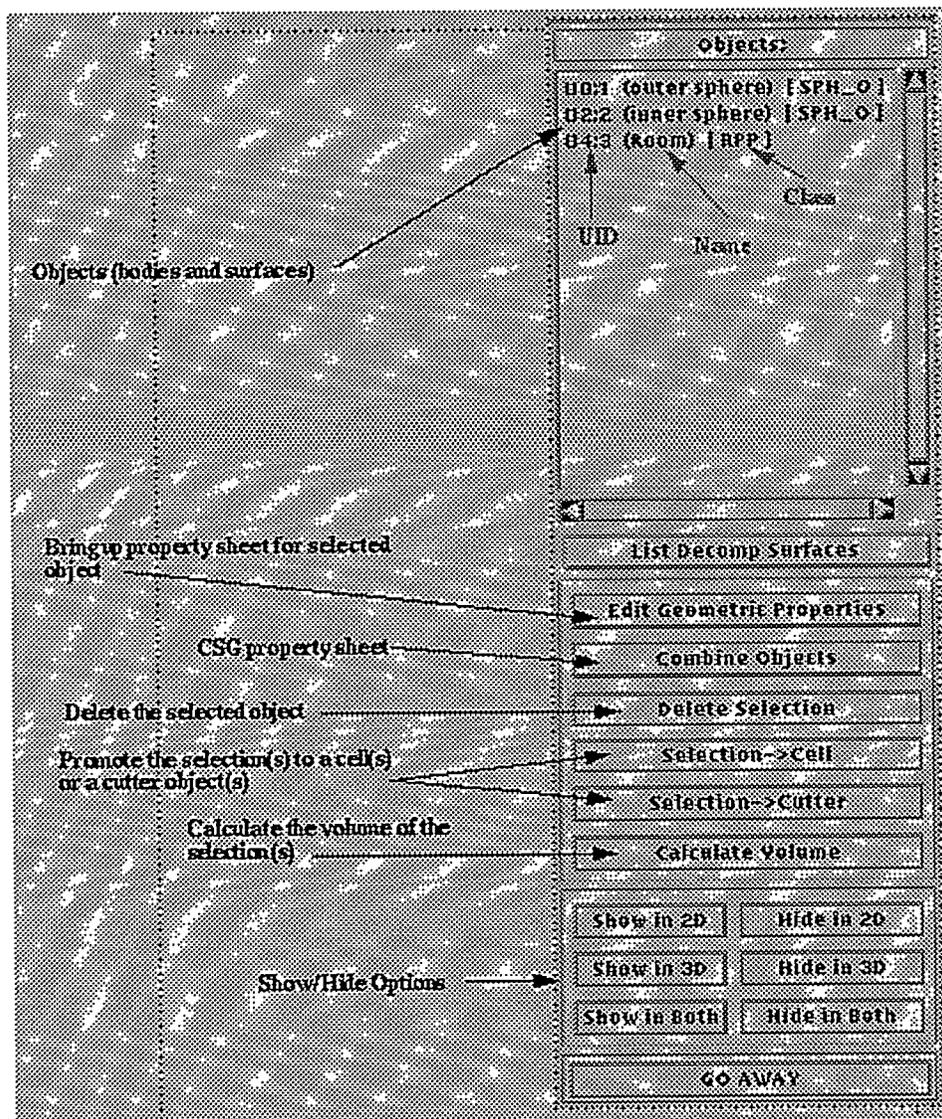
## **Other Editing and Viewing Options**

We have seen some of Justine's powerful editing and visualization options, in the form of modifying existing objects, copying them, deleting them, and so on, both with mouse based and property sheet based operations. However, there are additional editing and visualization details and options that have yet to be discussed.

### **Scrolling Lists**

Justine provides different scrolling lists of objects, cells, materials, and so on, which allow selections for

a variety of operations. An example of such a scrolling list for geometry objects is shown below.



The scrolling lists are used to select items for further editing. The object scrolling list is shown above. This list contains all user-generated objects (bodies and surfaces). The objects are listed by their name (default or user defined), their class (sphere, or wedge, etc.), and by their user ID numbers (UID). Justine maintains a separate list of ID's. One is Justine's internal number for the object, which appears first in the list, and the other is the corresponding MCNP surface number, which appears second. For example, from the figure, the line U0:1 (outer sphere) [SPH\_O] indicates that the class of this object is a sphere body (SPH\_O), it's name is "outer sphere," the MCNP surface id number is 1, and the Justine UID is 0.

Generally speaking, the user only examines the name of the object to reference it. The UID and class are typically not important from the standpoint of the scrolling list.

In general, the selection list works in this manner. A user may select one of the items in the list by pointing to it with the mouse and selecting the left mouse button, or the user may select multiple

contiguous items by holding down the mouse button and dragging the mouse over the list. If multiple non-contiguous items are desired, hold down the control key while depressing the left mouse button on each desired item. This will allow multiple selections on non-contiguous items.

Once the item(s) have been selected, the user may then apply a variety of operations to them. It should be noted that for some editing options, multiple selections are not appropriate. In such cases, the first selected is used.

### **Edit Geometric Properties**

This button pops up the property sheet for the selected item. This allows rapid access of different objects without having to resort to scrolling with the property sheet or selecting with the mouse.

### **Combine Objects**

This will pop up the CSG property sheet, with the first selected in the "Object A" slot and the second selected in the "Object B slot" (described later).

### **Delete Selected Object**

This will delete all objects currently selected in this scrolling list. This operation is not reversible.

### **Promotion**

Promoting a selection, or selections, to cutter bodies or cells is discussed later.

### **Show/Hide Options**

These buttons will all show or hide the selected object(s) as desired. This is a quick way of visually navigating the geometry.

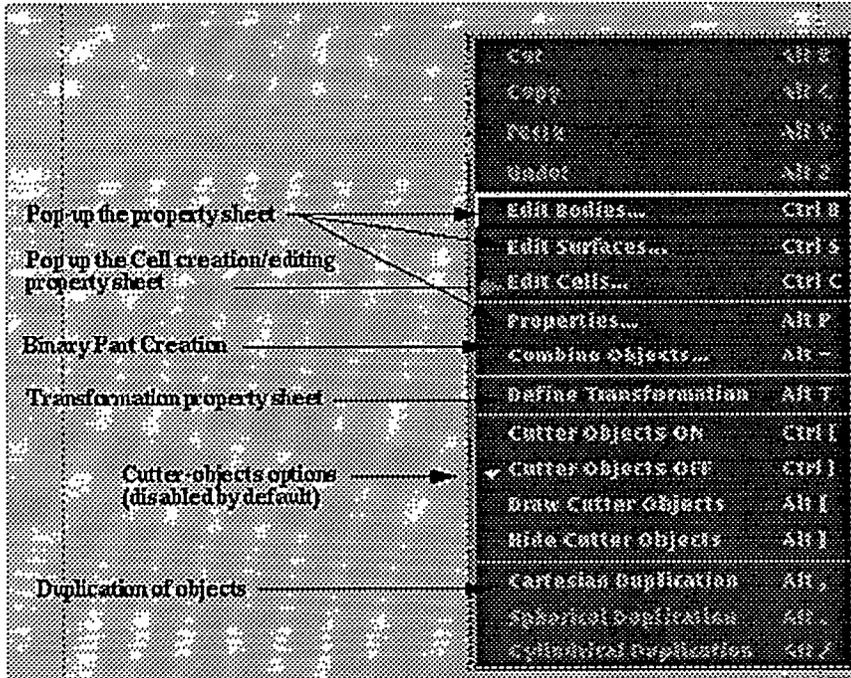
### **Calculate Volume**

This option will tell Justine to compute the volume of the selected object based on the polygons that are used to render the object in three dimensions. Surfaces are clipped by the world box.

The scrolling lists are accessible from different menu items. In the case of the scrolling list described above, for objects, it is available from the *Bodies* or *Surfaces* menus, and is listed in each as "Show Geometry List."

### **The Edit Menu**

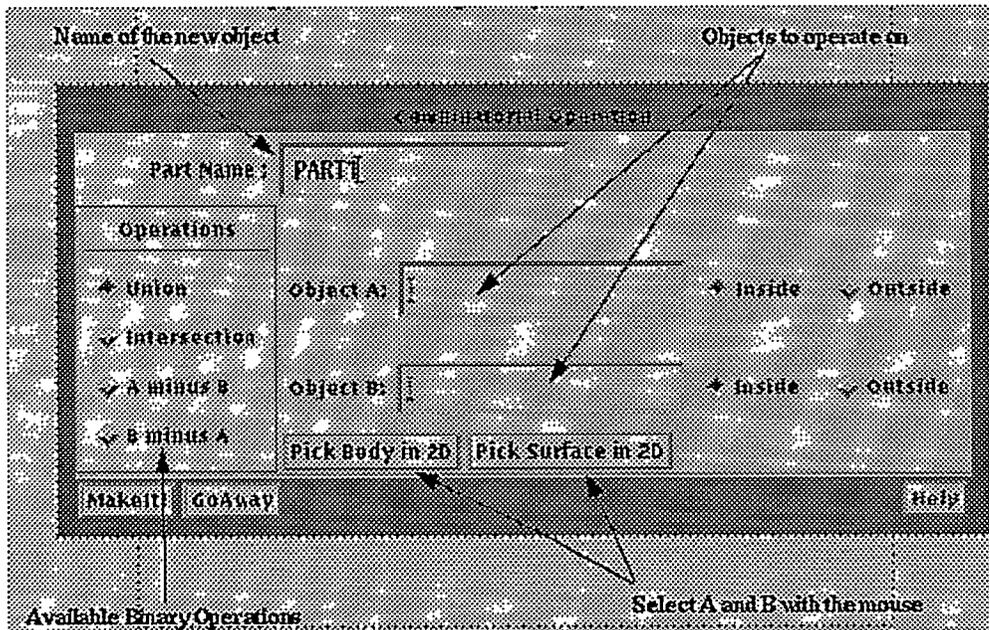
The edit menu provides a variety of additional editing functions that have not been discussed.



In addition to popping-up the property sheets, as previously discussed, several additional functions are provided through this menu. One thing to notice is that on this menu there are *three ways* to bring up essentially the same property sheet, that is, the property sheet that allows one to enter exact quantities for different objects. Discussion of the "Edit Cells" item is deferred until Chapter 4, "Creating Cells."

### Combine Objects

As previously explained, Justine's editing algorithm is based on the binary combinations of solid objects (CSG). This includes the combinations of objects that are not solids, such as surfaces. During the cell creation process, which is discussed in the next chapter, users must select different object and the sense of the cell before Justine will do the required CSG operations to define the cell. No other action is required on the part of the user. However, Justine does provide a mechanism for getting directly at the CSG operations themselves, namely, taking one object minus another, or unioning two objects together, and so on. The "Combine Objects" menu item brings up the following property sheet. The concept of inside and outside is clear for all but the plane. For the plane, the inside is rendered pink and the outside is simply the color for the plane object (in the 3-D window):

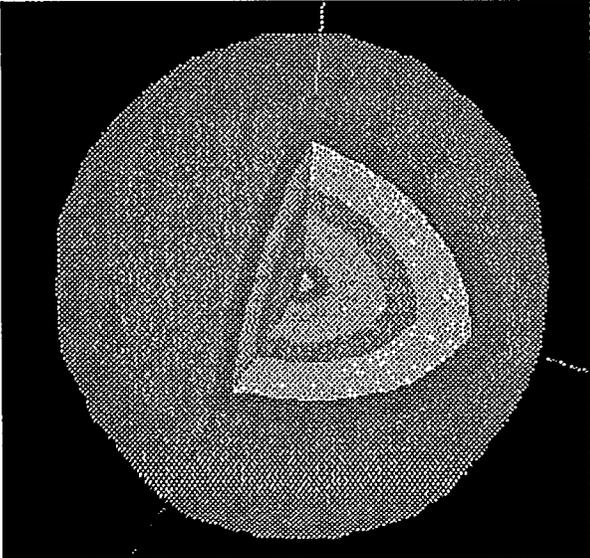


Justine refers to objects generated in this manner as *parts*. That is, if one takes a box primitive and, say, subtracts a cylinder primitive from it, the resulting box with a cylindrical hole removed is called a part in Justine. The default name for the resulting part is provided by Justine ("PART1" above), or the user may enter a different name. Object A and Object B may be provided in three ways. The user may enter the name of the objects to be operated on, the user may select the objects from a list (discussed earlier), or the user may pick the objects with the mouse in one of the 2-D slice windows. The latter is accomplished by selecting one of the buttons on the above property sheet, then moving into a 2-D window and selecting the desired object. This causes the name of the selected object to appear in one of the object slots on the above property sheet. One then selects the desired operation for these objects from the exclusive list on the left. Selecting "Make It!" will cause Justine to generat

### Cutter Objects

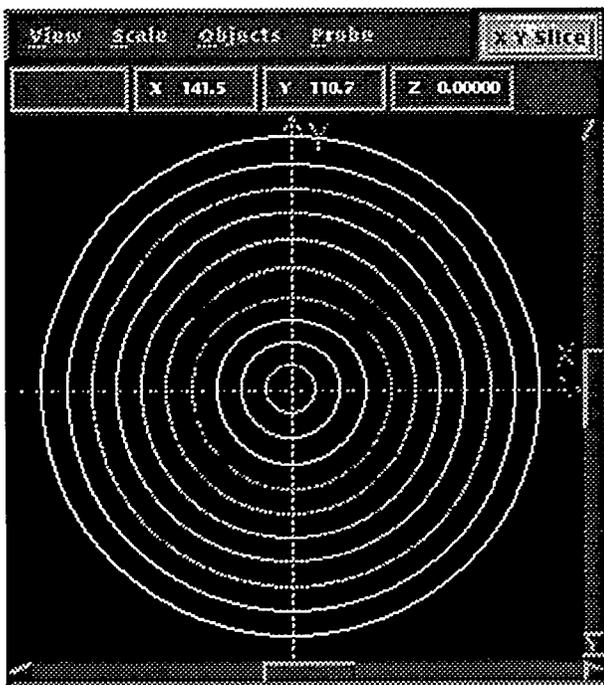
Cutter objects are a special class of body or surface. Cutter objects are used to cut away parts of the geometry to make other parts visible, but do not themselves comprise the geometry in question.

For example, suppose that a problem consisted of many spheres nested together as we see below.

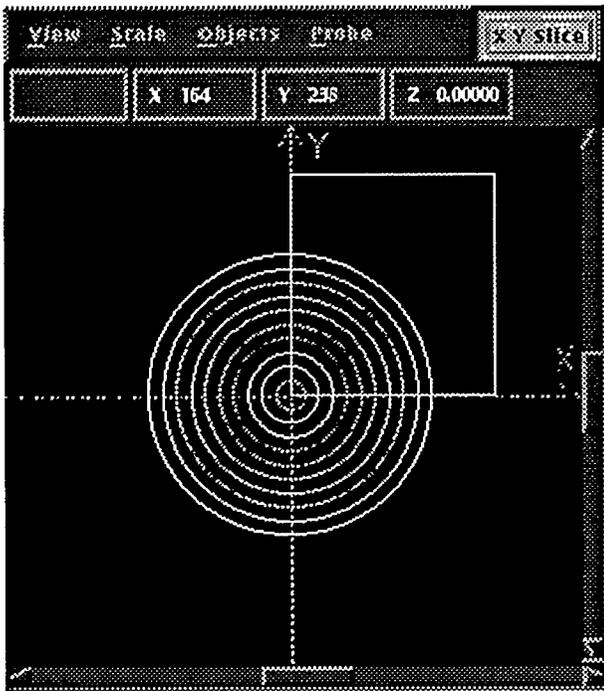


As can be seen, there are several nested spheres in this geometry. However, if it were not for the octant removed from the geometry, a shaded representation would reveal a solid (red) sphere. Cutter objects allow a user to cut into the geometry and remove sections for visualization purposes. There are three steps to cutter object generation; the cutter object must be realized; promoted to cutter class; cutter objects must be enabled so that they will be constructed by Justine.

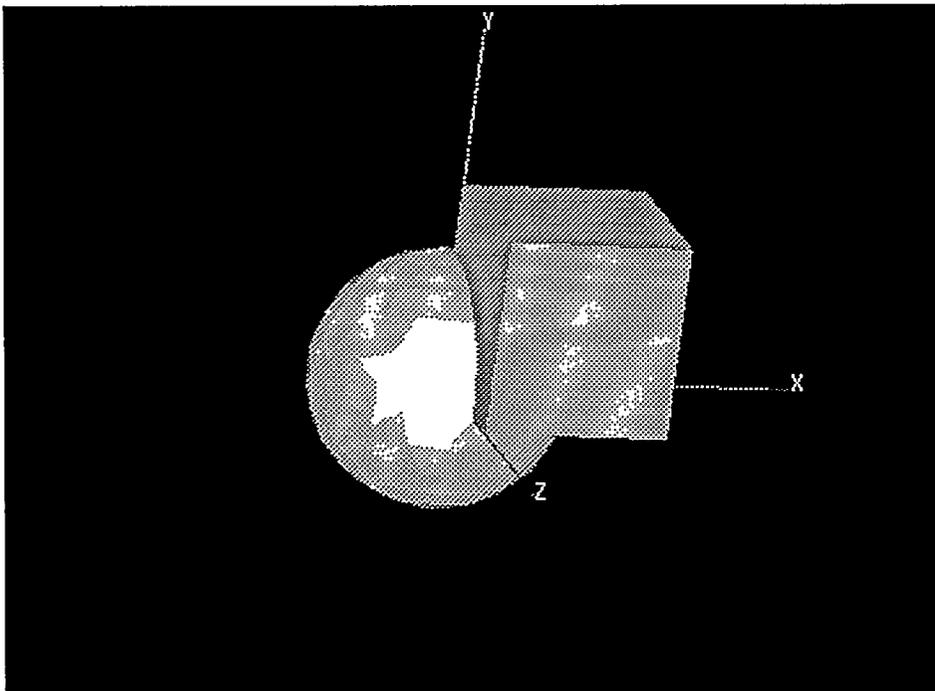
Consider the example above. Suppose that we have generated nine nested spheres and wish to cut an octant out of them. Examine the slice below:



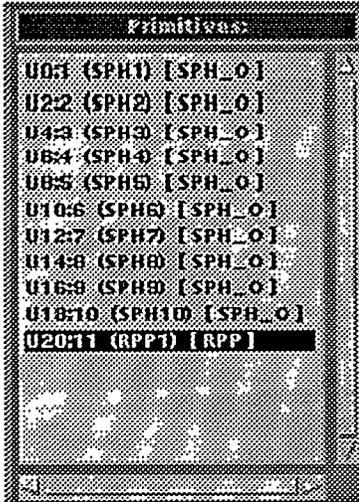
The simplest way to remove an octant is to draw an appropriate object out in 2-D. In this case, we would select RPP under the "Draw in 2D" sub-menu beneath the "Bodies" menu item on the main window. We would then draw out the RPP shown below.



In 3-D, we now have



Assuming we allowed Justine to use default names for all objects, our list now looks like this:



Our last generated object was the object that we wish to use as a cutter object. In the above figure, we have highlighted it. Now, we select "Selection -> Cutter" which promotes this object to a cutter body. Upon selection of this item, the box disappears from all four geometry rendering windows. This is due to the default settings in Justine.

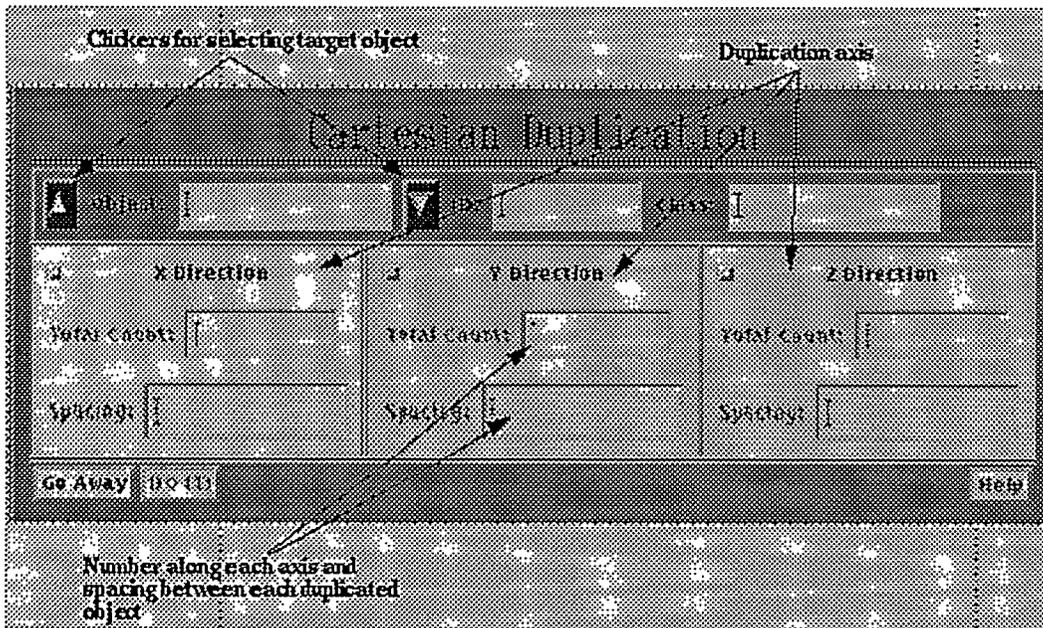
At this point, all we have done to our geometry is generate another object (an RPP in this case) and promoted it to a special status (cutter body). With cutter objects disabled, nothing further happens. However, if the "Cutter Objects ON" is selected from the "Edit" menu, Justine will generate the polygons for the new combination, and display the geometry with the cutter object removed, as depicted when we began this section.

Once the cutter objects have been enabled, and the polygons are generated, one can toggle the cut-view on and off using the "Cutter Objects ON/OFF" settings. The first time cutter objects are requested, Justine must generate CSG polygons which may take some time. After that, however, rendering is quick as images are simply toggled back and forth.

Finally, by default, Justine does not render the cutter objects themselves when they are enabled. However, if a user wishes to view the cutter object along with the cut-rendered geometry, selecting "Draw Cutter Objects" enables this feature. "Hide Cutter Objects" disables this feature, and is the default in Justine.

### **Cartesian Duplication**

This selection is a powerful tool, allowing a user to easily generate a repeated pattern of objects in a geometry. When selected, this option brings up the following property sheet:

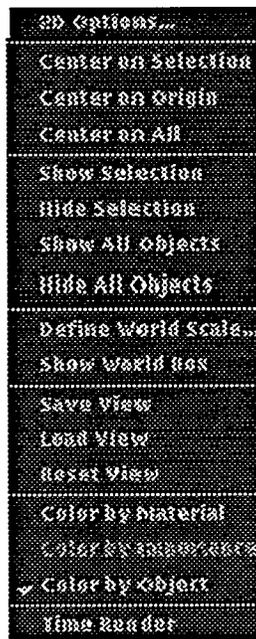


This simple property sheet allows a user to quickly construct a lattice of cylinders inside a box, for example. As the property sheet indicates, duplication can be done independently along any combination of the coordinate axes.



## The Global View Menu

The global view menu, shown in Figure 3.14, allows the user to control visualization aspects of Justine in all three of the geometry views. Many of these options rely on the user selecting items from Justine's object scrolling list discussed earlier. All items referring to "selections" work in this manner. Some brief comments on some of the more obvious options follow.



**Figure 3.14:** The Global View menu items. These items control the visualization of objects in Justine.

**Center on Selection/Origin/All** This will center the 3-D viewport on the selected item, the origin, or the geometric problem center, respectively.

**Show/Hide Selection** This will toggle the selected item(s) on or off in all four geometry rendering windows.

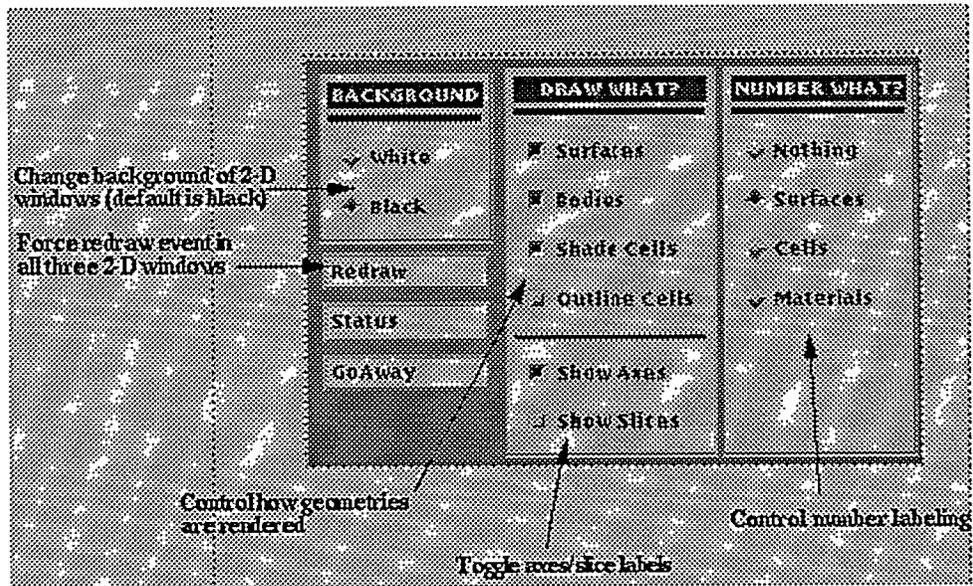
**Show/Hide All** This option will show or hide all available objects in all four rendering windows.

**Color by Material/Object** By default, Justine colors each generated object and cell using an internal color list (which can be changed by the user). This is coloring by object, and is the default in Justine. However, when colors are assigned to other problem attributes, such as materials, Justine can also color the shaded, wireframe, or ray-traced objects by such an attribute. As other attributes are added, these selection options will grow.

**Time Render** When enabled, this reports Justine's rendering time.

## 2D Options

This item brings up a property sheet that controls many aspects of the 2-D rendering of objects and cells. This property sheet controls what information is presented to the user in the 2-D windows, and how that information is presented. Modifications made using this property sheet affect all three 2-D windows.



### *Redraw*

This selection forces a redraw event in all three 2-D windows. Depending on the options selected in this property sheet, it is sometimes necessary to force Justine to redraw the windows immediately. Generally, it is a good idea to select this button before leaving this property sheet.

### **Draw What?**

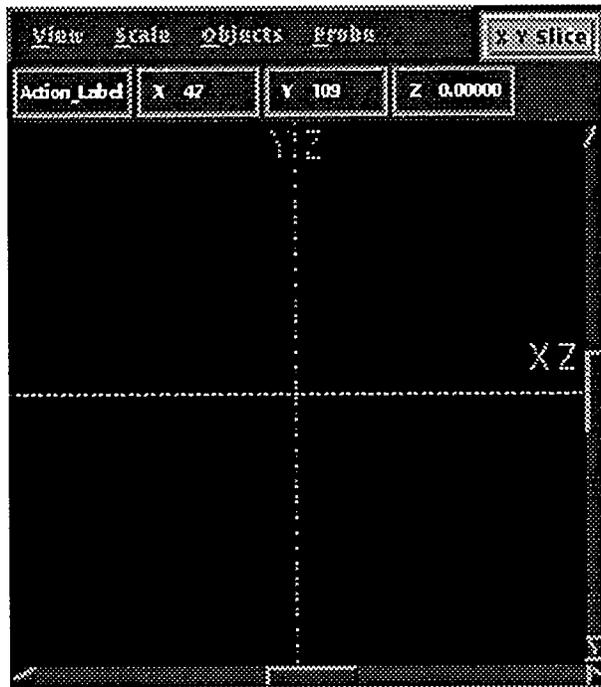
This region of the property sheet controls how objects and cells are rendered in the 2-D windows, and how the axes are drawn. The default settings draw surfaces, bodies, shade all cells, and show the axes. If, for example, surface drawing is disabled, any generated surfaces will not be rendered in the 2-D windows. Similarly for bodies.

### *Shade/Outline Cells*

By default, all created cells (see next chapter) are shaded in the 2-D windows. However, a user may specify that the cells be outlined rather than shaded, thus making the cells transparent in the 2-D windows. This will be discussed in more detail in the next chapter.

### *Show Axes/Slices*

The default is to display the labeled in each of the 2-D windows. However, it is also possible to display the axes labeled with the coordinate axis *plus* the slice axis. This is accomplished by disabling "Show Axis" and enabling "Show Slices." Doing so produces the following (in the XY Slice as an example):

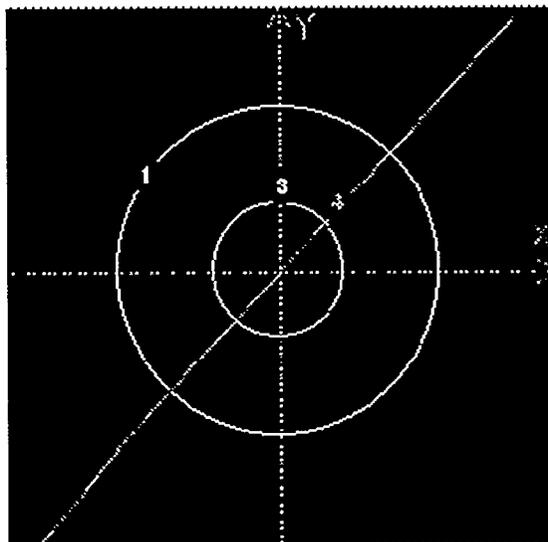


Note that each axis is now labeled with both the coordinate axis label (X or Y) and the slice axis (Z).

### Number What?

These selections control what is numbered in the 2-D windows. Justine's default is to number surfaces only. However, numbering can also be enabled for materials or cells, or no numbering at all can be specified.

Numbering surfaces, cells, and/or materials is sometimes useful for setting additional MCNP options (for example) or for those long-time MCNP users that are used to thinking of geometries in terms of numbered surfaces and cells. An example of numbered surfaces (XY Slice only) are shown below:

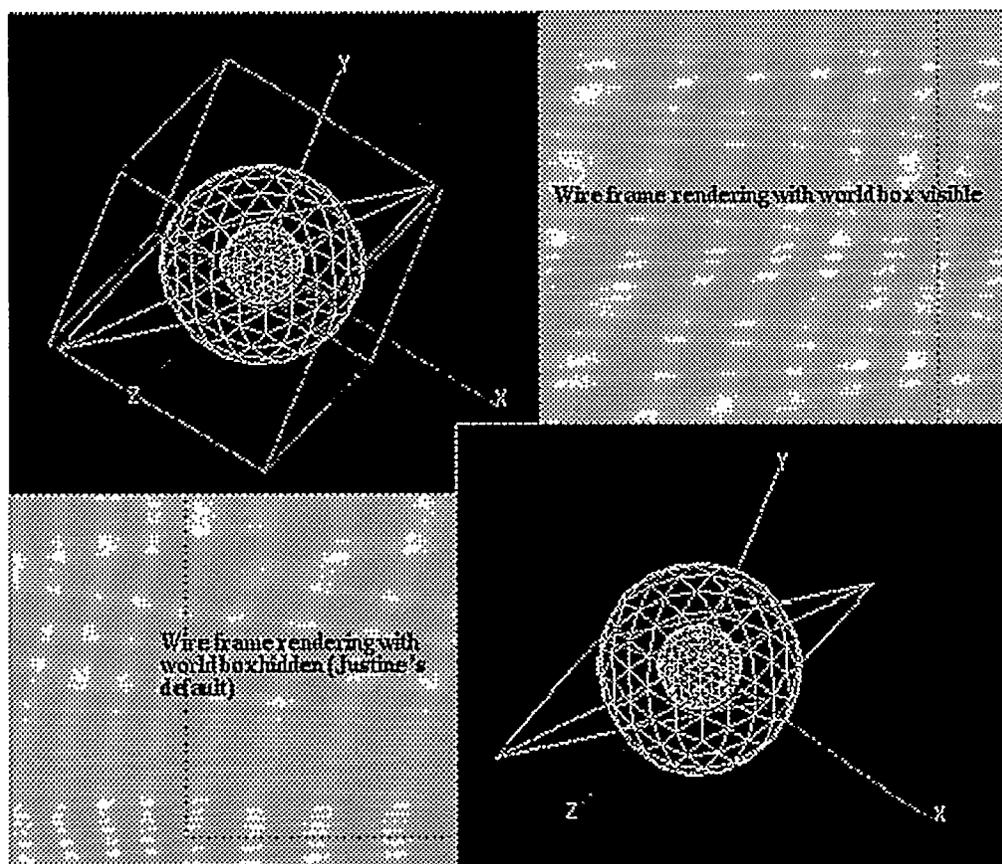


Here we see three surfaces each labeled with their surface numbers (the reason the numbers are 1, 3, and 4 is that one surface, surface 2, is not visible in this slice).

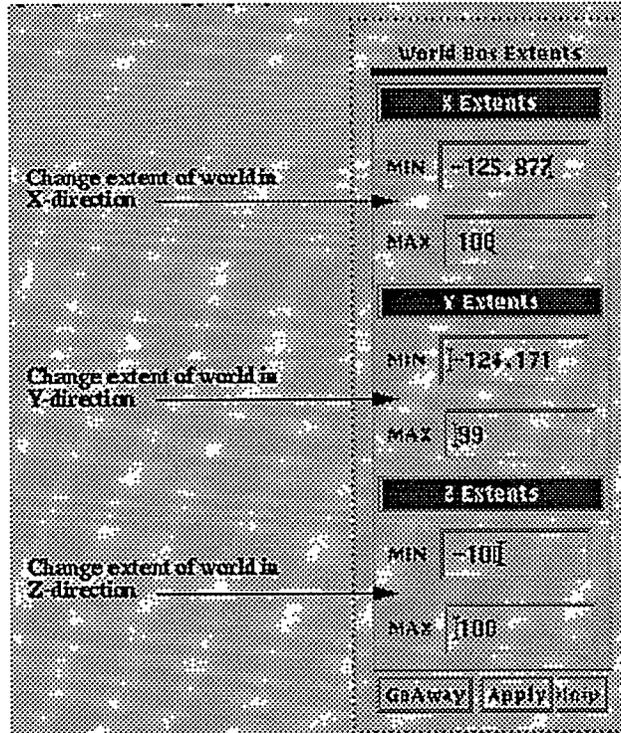
## World Scale/ World Box

In Justine, infinite surfaces are clipped to an imaginary *world box*, which is larger than the extent of any object or generated cells in the geometry. If the user wishes to visualize this imaginary box, selecting "Show World Box" will do so. This menu selection reverts to "Hide World Box" once selected to allow the user to toggle the box back off.

Below, a wireframe geometry consisting of two spheres and a single plane is shown both with the world box on and off. Notice that the plane is clipped to the world box boundary. This is how Justine handles infinite surface rendering.



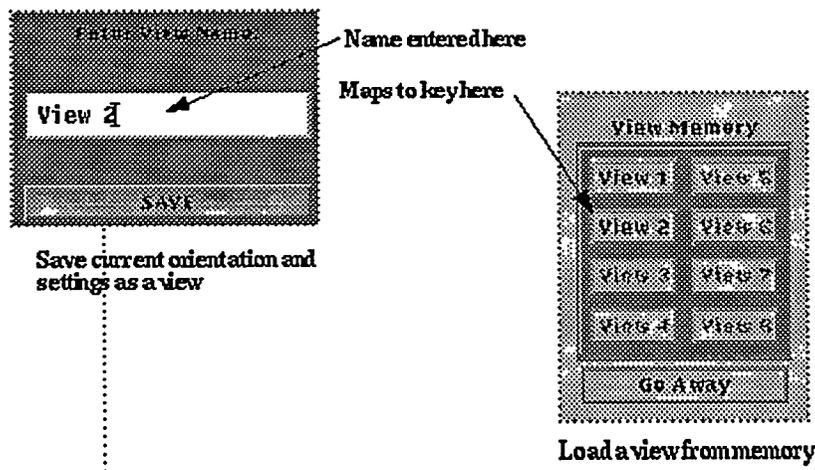
In addition to toggling the world box on and off, the user is also provided a means to change the size of the world box, that is, change the size of the "world" for a particular problem. This is accomplished from the "Define World Scale" menu item, which brings up the following property sheet:



Generally speaking, the size of the world box should be left as Justine calculates it. However, there are special circumstances when the user needs to alter the computed size (for some old MCNP input files, for example).

## Saving and Restoring Views

A view refers to a settings that apply to the visualization of geometries, including orientation and zooming factors. Justine allows the user to save and restore up to eight different views of a geometry. The "Save View" and "Load View" property sheets are given below:



The user places the geometry in the desired orientation, with the desired rendering options enabled, and then selects "Save View." The user is then presented with the left property sheet above, and may enter a name for the view or accept the default name (for example, one might orient a particular geometry to

look down some feature, and then save the view as "Down Pipe"), then save the view. The name appears in a selectable button on the sheet that appears when "Load View" is selected. Selecting these buttons immediately places the geometry in the orientation saved. Simply selecting different buttons toggles between the saved views.

"Restore View" clears the view memory and allows eight new views to be saved. This operation is not reversible.

## Special Object Property Sheets

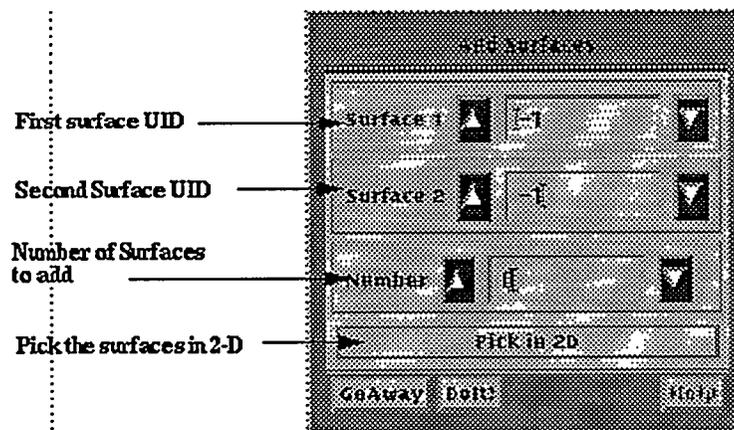
Justine offers additional special purpose property sheets for the generation of specialized geometries and configurations.

### Spherical Shell/Cylinder Annulus

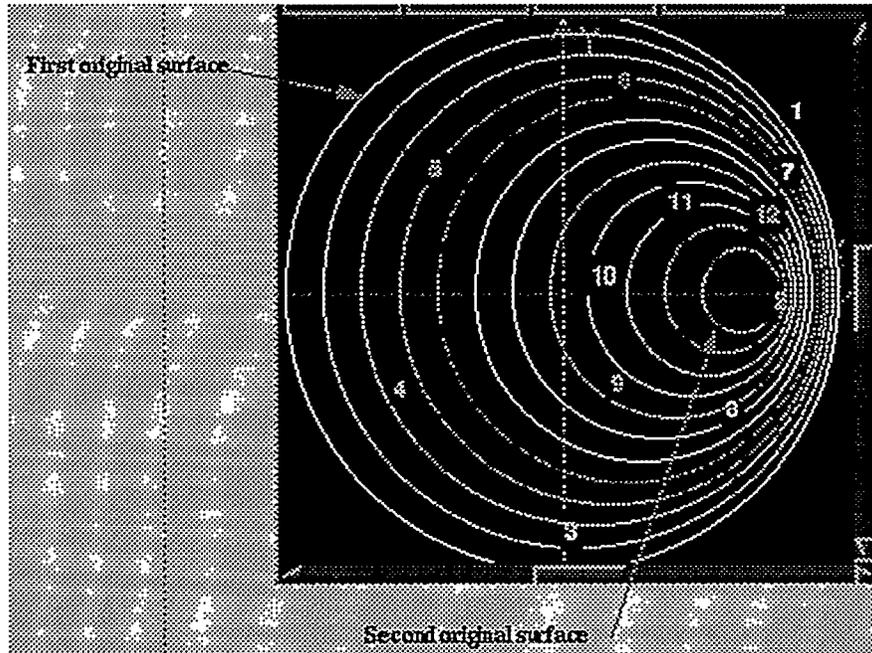
Selecting one of these options (under "Bodies") brings up a property sheet that looks very much like either the sphere body or cylinder body property sheets (depending on which is selected) with one important addition. The user is allowed to enter both an inner and outer radius, to create the wall thickness between the two bodies.

### Adding Surfaces

The "Add Surface" menu item (under "Surfaces") allows a user to add an arbitrary number of surfaces of the same type between two existing surfaces.



The user selects two surfaces of the same sub-class (like two spheres, two ellipsoids, etc.) by either clicking or entering the UID for the surface (from the scrolling list) or by picking the surfaces in 2-D with the mouse. Then, the user enters the number of surfaces to generate. When "Do It!" is selected, a linear interpolation between the closest points on the two surfaces is done, and equidistant surfaces are added between them. The interpolation will adjust the parameters for the surface accordingly so that equidistant new surfaces are generated. An example of such a generation is shown below.



Here we see that we have added ten new surfaces between the two original surfaces.

---

## Summary

In this chapter we have explored most of Justine's powerful geometry visualization tools. Some of the features discussed here will be elaborated on in the Power Features chapter.

---

*Last Modified Thu Sep 7 13:35:26 MDT 1995, Stephen R. Lee*

srlee@lanl.gov

---