Chapter 9.
Examples

This chapter provides two examples of the save file format. The first example shows how to read a record in a save file. The second example illustrates subtypes and references.

There are two types of save files denoted by their extension: save as text (.sat) and save as binary (.sab). The data elements of a .sat and a .sab are identical, except that sab files store it in a binary form. For the sake of clarity, the examples in this chapter refer to a .sat file, because these files can be viewed and understood with a simple text editor.

Save File Example

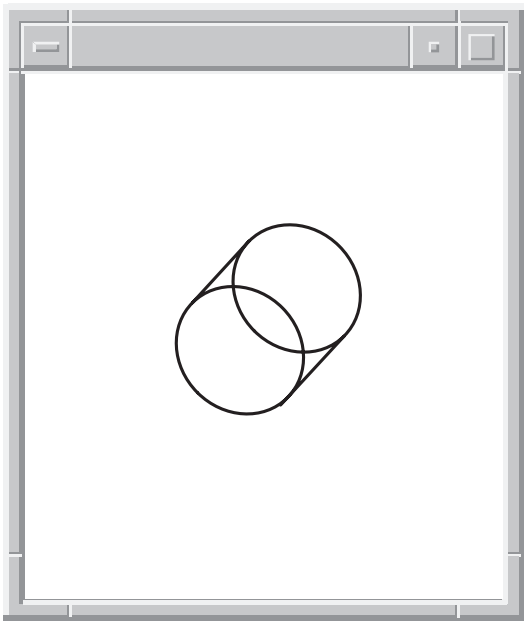
The following simple example shows a .sat file with topology and geometry. The first two lines are the header, followed by the entity records, and finally the end marker. Optional sequence numbers were included.

Record lines “–0” and “–1” are explained in more detail following the full example to show exactly what each item is and where to locate that information in this manual. In explaining these two records in the tables, each piece of information is looked at in relation to the data element itself, the subsection in the manual where it is located, and a brief description of the data. The records start from entity in the center and use spaces as separators between each data element.

The .sat example file was created using Scheme, which displayed a simple solid cylinder. The option to create sequence numbers in the save file was turned on.

```
# ACIS Scheme Commands

(option:set
    ”sequence_save_files” #t)
(solid:cylinder
    (position 0 0 0)
    (position 0 20 0)
    10)
(part:save ”exp_cyl”)
```

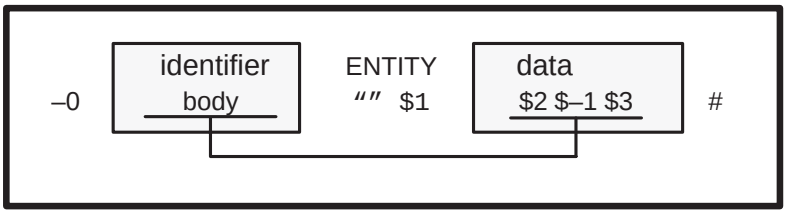


```
200 0 1 0
23 ACIS Test Harness – 2.0 8 ACIS 2.0 24 Mon Feb 12 13:59:03 1996 25.4
1e–06 1e–10
–0 body $1 $2 $–1 $3 #
–1 display_attribute–st–attrib $–1 $4 $–1 $0 1 #
–2 lump $–1 $–1 $5 $0 #
–3 transform $–1 1 0 0 0 0 –1 0 1 0 0 10 0 1 rotate no_reflect
    no_shear #
–4 rgb_color–st–attrib $–1 $6 $1 $0 0 1 0 #
–5 shell $–1 $–1 $–1 $7 $–1 $2 #
–6 id_attribute–st–attrib $–1 $–1 $4 $0 1 #
–7 face $–1 $8 $9 $5 $–1 $10 forward single #
–8 face $–1 $11 $12 $5 $–1 $13 forward single #
–9 loop $–1 $14 $15 $7 #
–10 cone–surface $–1 0 0 0 0 0 1 10 0 0 1 I I 0 1 forward I I I I #
–11 face $–1 $–1 $16 $5 $–1 $17 forward single #
–12 loop $–1 $–1 $18 $8 #
–13 plane–surface $–1 0 0 –10 0 0 –1 –1 0 0 forward_v I I I I #
–14 loop $–1 $–1 $19 $7 #
–15 coedge $–1 $15 $15 $18 $20 1 $9 $–1 #
–16 loop $–1 $–1 $21 $11 #
–17 plane–surface $–1 0 0 10 0 0 1 1 0 0 forward_v I I I I #
–18 coedge $–1 $18 $18 $15 $20 0 $12 $–1 #
–19 coedge $–1 $19 $19 $21 $22 1 $14 $–1 #
–20 edge $–1 $23 $23 $18 $24 forward #
```

```
–21 coedge $–1 $21 $21 $19 $22 0 $16 $–1 #
–22 edge $–1 $25 $25 $21 $26 forward #
–23 vertex $–1 $20 $27 #
–24 ellipse–curve $–1 0 0 –10 0 0 –1 10 0 0 1 I I #
–25 vertex $–1 $22 $28 #
–26 ellipse–curve $–1 0 0 10 0 0 1 10 0 0 1 I I #
–27 point $–1 10 0 –10 #
–28 point $–1 10 0 10 #
End–of–ACIS–data
```

The first two lines of the file are the header. The third line, shown below, describes a body entity.

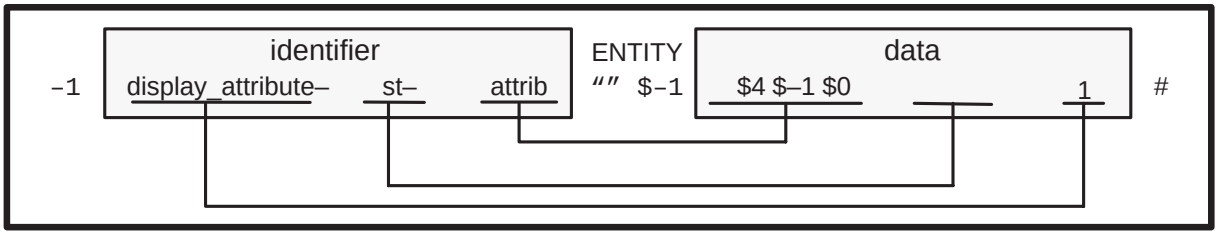
```
–0 body $1 $2 $–1 $3 #
```



Code Output	Save identifier:	Description
–0		Sequence number 0; first data record of file.
body	**body**	ident – (BODY derived from ENTITY)
$1	**ENTITY**	ENTITY data – Attribute $rec_num
$2	**body**	BODY data – Lump $rec_num
$–1	**body**	BODY data – Wire $rec_num
$3	**body**	BODY data – Transform $rec_num
#		Terminator

The fourth line of the file, shown below, describes a display attribute entity.

```
–1 display_attribute–st–attrib $–1 $4 $–1 $0 1 #
```



Code Output	Save identifier:	Description
`–1`	**ENTITY** (implied)	Sequence number 1; address for attribute pointer called out in sequence number 0. The double quotation marks (“”) are meant to signify that ENTITY is implied.
`display_attribute–`	**display_attribute**	save identifier (display_attrib derived from ATTRIB_ST)
`st–`	**st**	save identifier (st derived from ATTRIB)
`attrib`	**attrib**	save identifier (attrib derived from ENTITY)
`$–1`	**ENTITY**	ENTITY data – Attribute $rec_num
`$4`	**attrib**	ATTRIB data – Next attribute $rec_num
`$–1`	**attrib**	ATTRIB data – Prev. attribute $rec_num
`$0`	**attrib**	ATTRIB data – Owner $rec_num
`1`	**display_attribute**	DISPLAY_ATTRIB data – Display revision
`#`		Terminator

Subtypes and References Example

As a more graphical illustration of how subtyping and referencing work in a save file, several Scheme commands and a resulting .sat file are presented. The Scheme commands generate a cylinder which has one end bounded by a spline face. Moreover, another cylinder intersects the cylinder along the spline face, forming a cylindrical groove in the spline face. The spline surface is used in the definition of several spline curves in this example.


```
# ACIS Scheme Commands

(set:options
    ”sequence_save_files” #t)
(solid:cylinder
    (position 0 0 0)
    (position 0 0 20)
    10)
;; result [entity 1]
(solid:wiggle 20 20 40 1 1 1 1)
;; result [entity 2]
(solid:intersect
    (entity 1)(entity 2))
(solid:cylinder
    (position –20 0 20)
    (position 20 0 20)
    3)
;; result [entity 3]
(solid:subtract
    (entity 1)(entity 3))
(part:save ”exm_ref”)
```

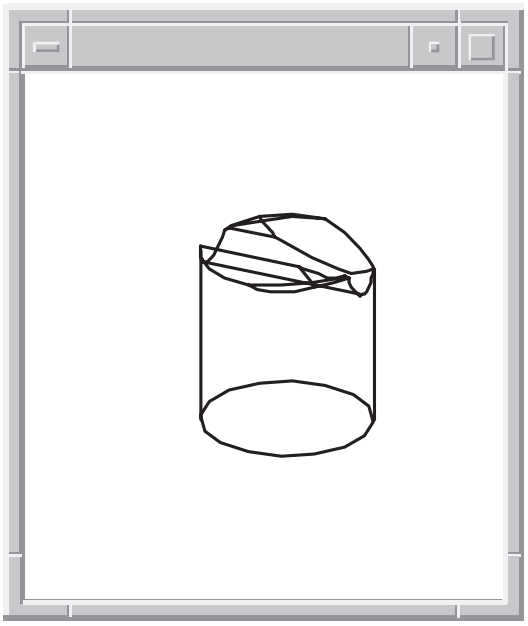


Figure 9-1. ACIS .sat File Example

The following example is taken from the file exm_ref.sat created in the above Scheme code. It does not list all sequence numbers from 0 to 171, but rather only a portion of them. The highlighted entries for sequence numbers 17, 96, and 118 are explained in more detail following this listing.

```
200 0 1 0
–0 body $1 $2 $–1 $3 #
–1 display_attribute–st–attrib $–1 $4 $–1 $0 3 #
```

```
–2 lump $–1 $–1 $5 $0 #
–3 transform $–1 1 0 0 0 1 0 0 0 1 0 0 10 1
       rotate no_reflect no_shear #
–4 rgb_color–st–attrib $–1 $6 $1 $0 0 1 0 #
–5 shell $–1 $–1 $–1 $7 $–1 $2 #
–6 id_attribute–st–attrib $–1 $–1 $4 $0 35 #
–7 face $–1 $8 $9 $5 $–1 $10 reversed single #
–8 face $–1 $11 $12 $5 $–1 $13 forward single #
–9 loop $–1 $–1 $14 $7 #
–10 cone–surface $–1 0 0 10 1 0 0 0 0 –3 1 I I 0 1 forward I I I I
#
–11 face $–1 $15 $16 $5 $–1 $17 forward single #
–12 loop $–1 $–1 $18 $8 #
–13 plane–surface $–1 0 0 10 0 0 1 1 0 0 forward_v I I I I #
–14 coedge $–1 $19 $20 $21 $22 1 $9 $–1 #
–15 face $–1 $23 $24 $5 $–1 $25 forward single #
–16 loop $–1 $–1 $26 $11 #
–17 spline–surface $–1 forward { exactsur nubs 3 3 open
       open none none 3 2
    .
    .   [Data not shown; this is subtype number 0]
    .
    } I I I I #
–18 coedge $–1 $27 $28 $29 $30 0 $12 $–1 #
    .
    .   [Sequence entries 19 through 94 not shown]
    .
–95 vertex $–1 $93 $138 #
–96 intcurve–curve $–1 forward { surfintcur nubs 3 open 4
    .
    .   [Data not shown; this is subtype number 8]
    .
    spline forward { exactsur nubs 3 3 open open none none 3 3
    .
    .   [Data not shown; this is subtype number 9]
    .
    } I I I I
    plane 0 0 10 0 0 1 1 0 0 forward_v I I I I
    .
    .   [Data not shown]
    .
    nullbs
    F 0 F 4.1671539659088168 } I I #
–97 coedge $–1 $64 $54 $134 $139 1 $65 $–1 #
    .
```

```
    .   [Sequence entries 98 through 116 not shown]
    .
–117 vertex $–1 $73 $152 #
–118 intcurve–curve $–1 forward { surfintcur nubs 3 open 4
    .
    .   [Data not shown; this is subtype number 10
    .   and makes a reference to subtype number 8]
    spline forward { ref 8 } I I I I
    .
    .   [Data not shown]
    .
    F 3.2425629808368615 F 7.4005854616279674 } I I #
–119 vertex $–1 $115 $153 #
    .
    .   [Sequence entries 120 through 168 not shown]
    .
–169 vertex $–1 $164 $171 #
–170 ellipse–curve $–1 0 0 –10 0 0 –1 10 0 0 1 I I #
–171 point $–1 0 10 –10 #
End–of–ACIS–data
```

The first occurrence of a subtype definition for exactsur is on sequence number 17, and is numbered 0 in the subtype index table. This is explained in the following table.

Code Output	Save identifier:	Description
17		Sequence number 17.
spline–	**spline**	save identifier (spline derived from SURFACE).
surface	**surface**	save identifier (surface derived from ENTITY).
$–1	**ENTITY**	ENTITY data – Attribute $rec_num.
forward	**spline data**	Parameter defining the sense direction for the spline curve.
{	**subtype**	Start of the subtype definition. In this particular case, the subtype index number is 0.
exactsur	**exact_spl_sur**	Subtype name is exactsur and uses data element spl_sur.
	spl_sur	Data element, which has a bs3 surface and a real used as the fit tolerance.

Code Output	Save identifier:	Description
nubs	**bs3_surface**	Ident: used as part of B-spline curve definition.
3	**bs3_surface**	integer, *u*-degree.
3	**bs3_surface**	integer, *v*-degree.
open	**bs3_surface**	ident: *u*-closure identifier.
open	**bs3_surface**	ident: *v*-closure identifier.
none	**bs3_surface**	ident: *u*-singularity.
none	**bs3_surface**	ident: *v*-singularity.
3	**bs3_surface**	ident: number of *u*-knots.
2	**bs3_surface**	ident: number of *v*-knots.
[Data not shown]	**bs3_surface**	Data that was left out related to the *u*-knot values, their paired multiplicity, the *v*-knot values, their paired multiplicity, associated *x*, *y*, *z* coordinates, and a fit tolerance.
}	**subtype**	End of the subtype definition. In this particular case, the subtype index number is 0.
I I		interval – *u*-parameter range.
I I		interval – *v*-parameter range.
#		Terminator for sequence number –17.

The listing before the above table also has an example of a reference. Sequence number –118 has the text "`{ ref 8 }`" embedded within its information. This is a direct reference to subtype object 8. (This references the ninth subtype definition within the file, because the subtype index numbering begins with 0.)

The subtype object with the index number 8 is defined on sequence number 96. Thus, that definition of a surfintcur is used as part of sequence number 118. Note that the reference to subtype 8 in sequence number 118 actually occurs during the definition of another subtype. Moreover, nested subtype definitions are also supported, as is seen within the subtype definition on sequence number 96.