

# Engineering Sketch Pad (ESP)



## User Training Session 8 Selection & Attribution

**John F. Dannenhoffer, III**

[john@geocentrictech.com](mailto:john@geocentrictech.com)

Geocentric Technologies LLC

updated for v1.28

- Purpose and Types of Attributes
- Setting Attributes
- Viewing Attributes: `DisplayFilter`
- Selecting Entities
- Attributes That are Automatically Set
- `Csystem`
- Editing Attributes: `UDPRIM editAttr`
- Homework Exercise

- Attributes are meta-data that can be used to tag any entity
- Attributes can be applied to:
  - Bodys
  - Faces
  - Edges
  - Nodes
- Attributes can be:
  - one or more integers (reserved for internal use)
  - one or more floating-point numbers
  - a character string

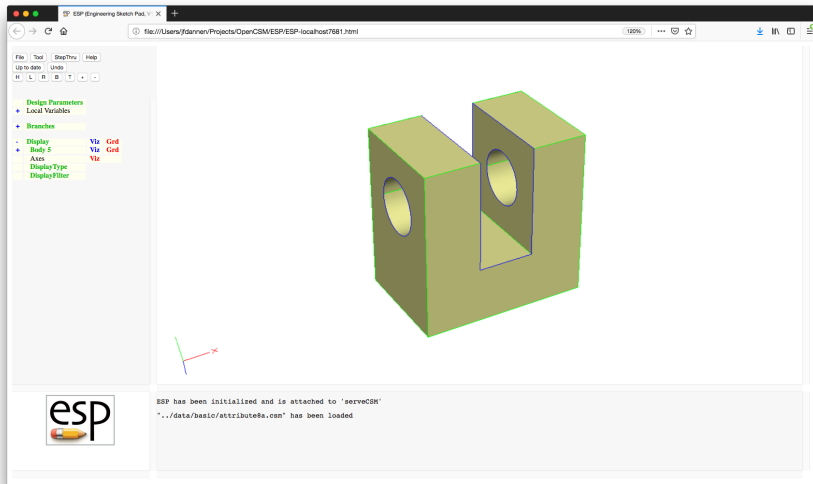
- Attributes can be associated with any Branch that produces a Body or the **SELECT** statement
- Attributes are defined by an **ATTRIBUTE** statement
- Take care when starting an Attribute name with a period (which is reserved for **EGADS**) or an underscore (which is reserved for **OpenCSM**)
- If the first character of the value is a dollar-sign, then the Attribute will contain a character string
- Otherwise the Attribute will contain one or more real (double) values
  - if the value is the name of a multi-valued Parameter, then the Attribute will be multi-valued
  - if the value is a semi-colon-separated list of expressions, then the Attribute will be multi-valued
  - otherwise the Attribute will be a single real (double)

- Global Attributes are set with an **ATTRIBUTE** statement before the first Body is created
- Attributes can be set for a Body (and all newly-created Faces) with an **ATTRIBUTE** statement following the Branch that created the Body
- Attribute can be set on any entity(s) by putting an **ATTRIBUTE** statement following a **SELECT** statement
- Best practice is to set the Attributes as soon as the Body is created (for example via a primitive or grown Body command)



# Attribute Example (1)

## Whole configuration





# Attribute Example (2)

.csm file

```
ATTRIBUTE density 2710 # global attribute
```

```
BOX      0 0 0 3 3 2
```

```
ATTRIBUTE tag $block
```

```
BOX      1 1 0 1 2 2
```

```
ATTRIBUTE tag $slot
```

```
SUBTRACT
```

```
CYLINDER -1 2 1 4 2 1 1/2
```

```
ATTRIBUTE tag $hole
```

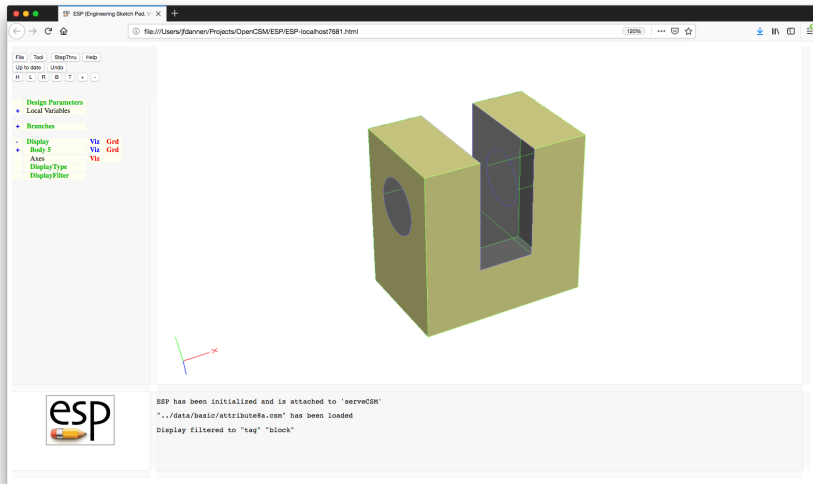
```
SUBTRACT
```

```
END
```



# Attribute Example (3)

DisplayFilter to tag block

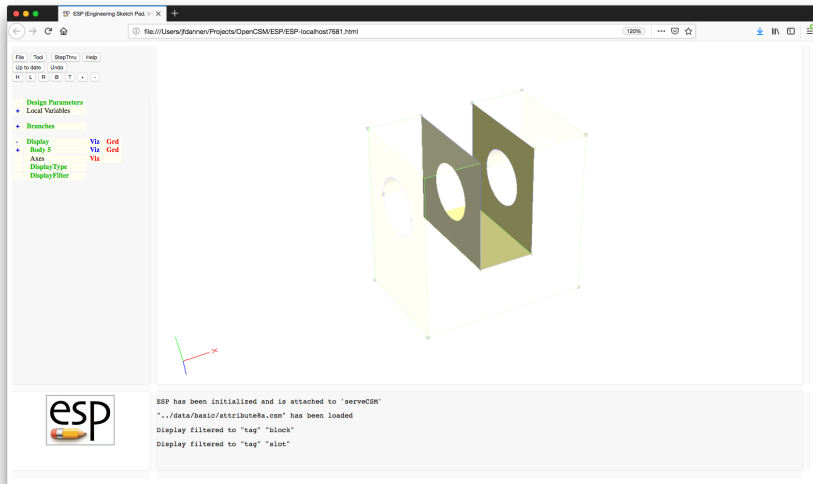






# Attribute Example (4)

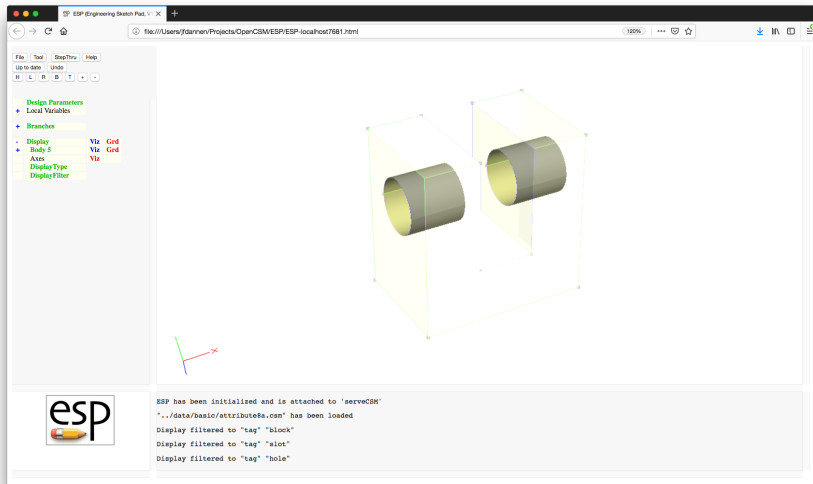
DisplayFilter to tag slot





# Attribute Example (5)

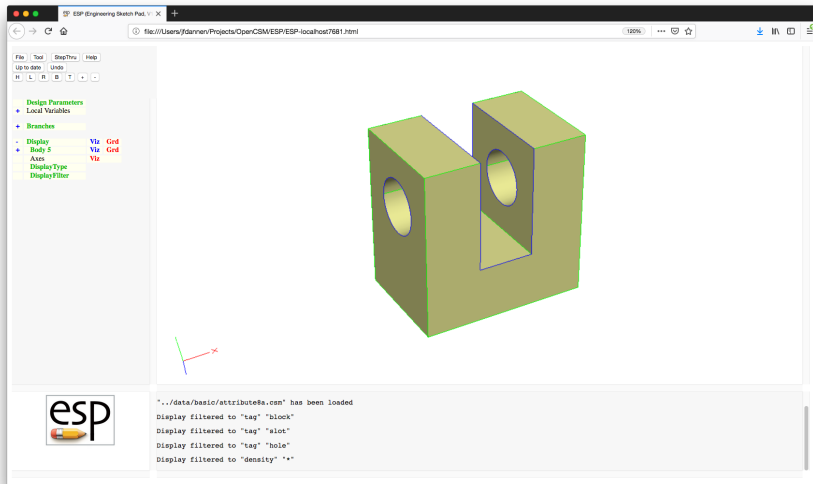
DisplayFilter to tag hole





# Attribute Example (6)

DisplayFilter to density \*

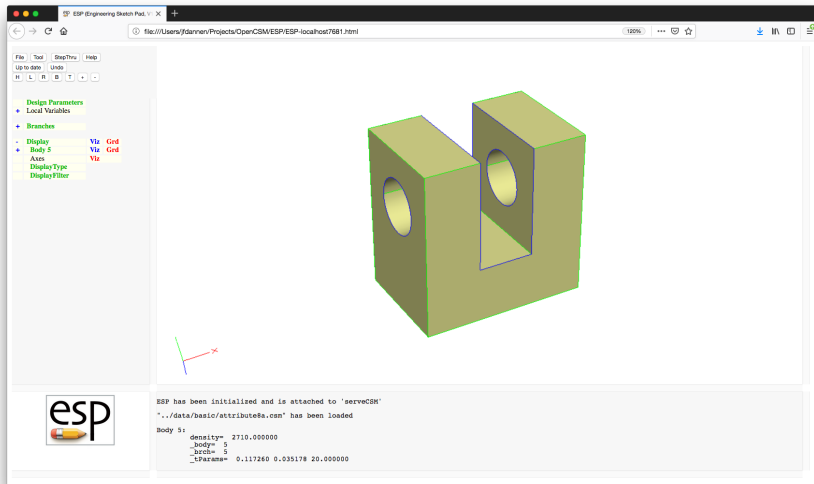


- Attributes can be viewed in **ESP** in three ways:
  - pressing the mouse in the Tree Window when cursor is over the Body name
  - pressing the **^** or **6** key when pointing to a Face, Edge, or Node in the Graphics Window
  - using the **Display Filter** option (at the bottom of the Tree Window)



# Attribute Example (6)

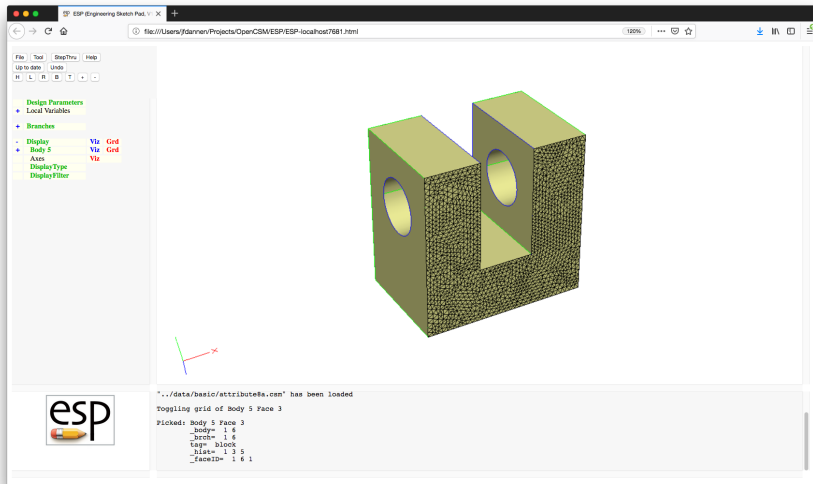
After pressing **Body 5** in TreeWindow

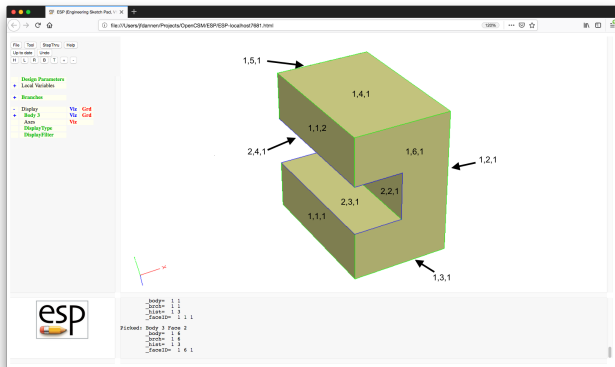




# Attribute Example (6)

After pressing  $\wedge$  on Face with Grid





BOX	0	0	0	2	3	3
BOX	-1	1	-1	2	1	5
SUBTRACT						

- FaceID is generated by the Body in which the Face first exists
- EdgeID is generated based upon the ibody/iford of its two adjoining Faces
  - Edge on bottom of front Face has EdgeID 1, 3, 1, 6, 1
  - Edge at bottom of slot & front Face has EdgeID 1, 6, 2, 3, 1

- The **SELECT** statement stores its values in:
  - **@seltype**
    - -1 if only a Body is selected
    - 0 if one or more Nodes are selected
    - 1 if one or more Edges are selected
    - 2 if one or more Faces are selected
  - **@selbody** contains the number of the Body selected
  - **@sellist** contains the list of the Nodes, or Edges, or Faces selected within **@selbody**
- **SELECT** statement keywords can be UPPERCASE, lowercase, but not MixedCase



- `SELECT BODY` — selects last Body created
- `SELECT BODY ibody` — selects Body `ibody`
- `SELECT BODY -n` — selects the  $n^{\text{th}}$  Body from the top of the Stack
- `SELECT BODY $attrName1 attrValue1 ...` — selects the last Body that matches all the given Attributes

- `SELECT FACE` — selects all Faces in selected Body
- `SELECT FACE iface` — selects Face `iface` in selected Body
  - using this is considered a bad practice since Face numbering may change depending on the version of `OpenCASCADE` that is being used
- `SELECT FACE ibody1 iford1 iseq=1` — selects the Face that has the indicated `ibody1/iford1`
  - as each Face is created, it is marked with the Body in which it was created and the face-order in that Body. This is the preferred technique.
- ...

- ...
- `SELECT FACE xmin xmax ymin ymax zmin zmax`
  - if `xmin < xmax`, selects all Faces fully between `xmin` and `xmax`
  - if `xmin > xmax`, selects all Faces partially between `xmin` and `xmax`
  - same applies to `ymin`, `ymax`, `zmin`, and `zmax`
- `SELECT FACE x x y y z z` — selects Face whose center is closest to `x,y,z`
- `SELECT FACE -1 ibody1` — selects the Faces that are in common with a Face in `ibody1`
- `SELECT FACE -2 ibody1` — selects Faces that are within SolidBody `ibody1`
- `SELECT FACE $attrName1 attrValue1 ...` — selects the Faces that matches all the given Attributes

- A zero (0) can be used as a wildcard for either `ibody1`, `iford1`, or `iseq`
- The star (\*) can be used as a wildcard for either `attrName` or `attrValue`

```
BOX 0 0 0 1 1 1
```

```
SELECT FACE 1 1
```

```
    ATTRIBUTE x $min
```

```
SELECT FACE 1 2
```

```
    ATTRIBUTE x $max
```

```
SELECT FACE 1 3
```

```
    ATTRIBUTE y $min
```

```
SELECT FACE 1 4
```

```
    ATTRIBUTE y $max
```

```
SELECT FACE 1 5
```

```
    ATTRIBUTE z $min
```

```
SELECT FACE $x $min      # @sellist=1
SELECT FACE $x $*        # @sellist=1;2
SELECT FACE $* $min      # @sellist=1;3;5
SELECT FACE $* $*        # @sellist=1;2;3;4;5;6
```

- `SELECT EDGE` — selects all Edges in selected Body
- `SELECT EDGE iedge` — selects Edge `iedge` in selected Body
  - using this is considered a bad practice since Edge numbering may change depending on the version of `OpenCASCADE` that is being used
- `SELECT EDGE ibody1 iford1 ibody2 iford2 iseq=1` — selects the Edge that has the indicated `ibody1/iford1`
  - as each Edge is created, it is marked with the `ibody/iford` of the Faces that adjoin it. This is the preferred technique.
- ...

- ...
- `SELECT EDGE xmin xmax ymin ymax zmin zmax`
  - if `xmin < xmax`, selects all Edges fully between `xmin` and `xmax`
  - if `xmin > xmax`, selects all Edges partially between `xmin` and `xmax`
  - same applies to `ymin`, `ymax`, `zmin`, and `zmax`
- `SELECT EDGE xmid ymid zmid` — selects the Edge whose midpoint is closest to the given coordinates
- `SELECT EDGE -1 ibody1` — selects the Edges that are in common with a Edge in `ibody1`
- `SELECT EDGE -2 ibody1` — selects Edges that are within SolidBody `ibody1`
- `SELECT EDGE $attrName1 attrValue1 ...` — selects the Edges that matches all the given Attributes

- **SELECT NODE** — selects all Nodes in selected Body
- **SELECT NODE inode** — selects Node **inode** in selected Body
  - using this is considered a bad practice since Node numbering may change depending on the version of **OpenCASCADE** that is being used
- **SELECT NODE x y z** — selects the Nodes closest to the given coordinates
- **SELECT NODE -1 ibody1** — selects the Nodes that are in common with a Node in **ibody1**
- **SELECT NODE -2 ibody1** — selects Nodes that are within **SolidBody ibody1**
- **SELECT NODE \$attrName1 attrValue1 ...** — selects the Nodes that matches all the given Attributes

- `SELECT ADD ...` adds entities to the select list
- `SELECT SUB ...` subtracts (removes) entities from the select list
- `SELECT NOT` selects all entities not currently selected and unselects those that were selected
- `SELECT SORT key` sorts the select list by one of the keys: `$xmin`, `$ymin`, `$zmin`, `$xmax`, `$ymax`, `$zmax`, `$xcg`, `$ycg`, `$zcg`, `$area`, or `$length`



```
ATTRIBUTE density 2710                                # global attribute
```

```
BOX          0  0  0  3  3  2
```

```
ATTRIBUTE tag $block
```

```
BOX          1  1  0  1  2  2
```

```
ATTRIBUTE tag $slot
```

```
SUBTRACT
```

```
CYLINDER -1  2  1  4  2  1  1/2
```

```
ATTRIBUTE tag $hole
```

```
SUBTRACT
```

```
#-----
```

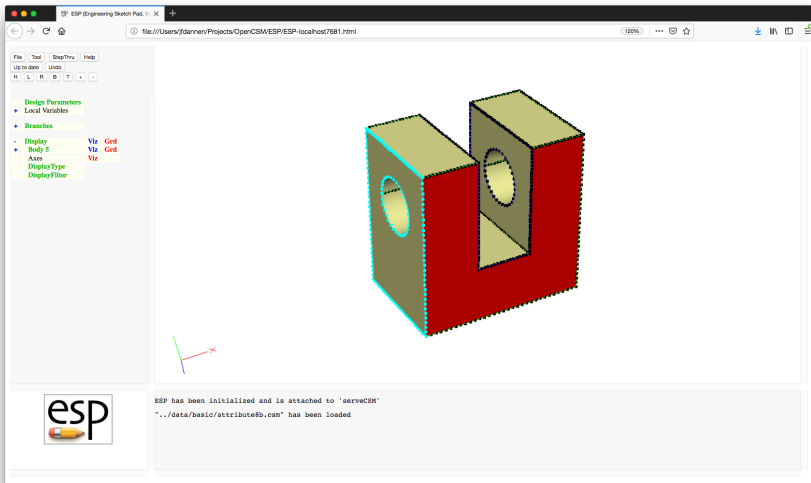
```
SELECT      FACE    1 6 1                                # select by FaceID
```

```
ATTRIBUTE _color $red
```

```
SELECT      EDGE    -.1 0.1  -.1 3.1  -.1 2.1  # select by bbox
```

```
ATTRIBUTE _color $cyan
```

```
ATTRIBUTE _gcolor $cyan
```





# Attributes Automatically Set to Bodys

<code>_body</code>	Body index (bias-1)
<code>_brch</code>	Branch index (bias-1)
<code>_tParams</code>	specified tessellation parameters: maximum side length, maximum specified sag, maximum angle
<code>_csys_*</code>	arguments when CSYSTEM was defined
<code>&lt;any&gt;</code>	all global attributes
<code>&lt;any&gt;</code>	all attributes associated with Branch that created Body
<code>&lt;any&gt;</code>	all attributes associated with "select \$body" statement



# Attributes Automatically Set to Faces (1)

`_body` non-unique 2-tuple associated with first Face creation  
[0] Body index in which Face first existed (bias-1)  
[1] face-order associated with creation (see above)

`_brch` non-unique even-numbered list associated with Branches  
that are active when the Face is created (most  
recent Branch is listed first)  
[2\*i ] Branch index (bias-1)  
[2\*i+1] (see below)

Branches that contribute to brch attribute are

- primitive (for which brch[2\*i+1] is face-order)
- udprim.udc (for which brch[2\*i+1] is 1)
- grown (for which brch[2\*i+1] is face-order)
- applied (for which brch[2\*i+1] is face-order)
- sketch (for which brch[2\*i+1] is Sketch primitive if  
making WIRE)
- patbeg (for which brch[2\*i+1] is pattern index)
- recall (for which brch[2\*i+1] is 1)
- restore (for which brch[2\*i+1] is Body number stored)

`_faceID`      unique 3-tuple that is assigned automatically  
    `[0]`      `body[0]`  
    `[1]`      `body[1]`  
    `[2]`      sequence number

if multiple Faces have same `_faceID[0]` and `_faceID[1]`,  
then the sequence number is defined based upon the  
first rule that applies:

- \* Face with smaller `xcg` has lower sequence number
- \* Face with smaller `ycg` has lower sequence number
- \* Face with smaller `zcg` has lower sequence number
- \* Face with smaller area has lower sequence number

`_hist`      list of Bodys that contained this Face (oldest to newest)

`_tParams`    specified tessellation parameters: maximum side length,  
              maximum specified sag, maximum angle

`<any>`      all attributes associated with Branch that first created Face

`<any>`      all attributes associated with "SELECT \$face" statement

```

_body      non-unique 2-tuple associated with first Edge creation
  [0]      Body index in which Edge first existed (bias-1)
  [1]      100 * min(body[1][ileft],body[1][irite])
           + max(body[1][ileft],body[1][irite])
           (or -3 if non-manifold)

_edgeID     unique 5-tuple that is assigned automatically
  [0]      _faceID[0] of Face 1 (or 0 if non-manifold)
  [1]      _faceID[1] of Face 1 (or 0 if non-manifold)
  [2]      _faceID[0] of Face 2 (or 0 if non-manifold)
  [3]      _faceID[1] of Face 2 (or 0 if non-manifold)
  [4]      sequence number

...

```

...

```
_edgeID[0]/[1] swapped with edge[2]/[3]
  100*_edgeID[0]+_edgeID[1] > 100*_edgeID[2]+_edgeID[3]
if multiple Edges have same _edgeID[0], _edgeID[1],
  _edgeID[2], and _edgeID[3], then the sequence number
  is defined based upon the first rule that applies:
  * Edge with smaller xcg      has lower sequence number
  * Edge with smaller ycg      has lower sequence number
  * Edge with smaller zcg      has lower sequence number
  * Edge with smaller length has lower sequence number
```

**\_nface**      number of incident Faces

**\_tParams**    specified tessellation parameters: maximum side length,  
                 maximum specified sag, maximum angle

**<any>**        all attributes associated with "select \$edge" statement



# Attributes Automatically Set to Nodes

<code>_nodeID</code>	unique integer
<code>_nedge</code>	number of incident Edges
<code>&lt;any&gt;</code>	all attributes associated with "select \$node" statement





# Special User-defined Attributes for Bodys

- `_makeQuads` to make quads on all Faces in Body
- `_name` string used in ESP interface for a Body
- `_stlColor` color to use for all Faces in an .stl file
- `_newSeqnum` if it exists, sequence numbering (in `_faceID` and `_edgeID`) is done using a scheme that is less susceptible to geometry changes
- `_oldPersist` if it exists, the old Edge Attribute persistence algorithm is used (and therefore some Attributes erroneously may not persist)



# Special User-defined Attributes for Faces

<code>_color</code>	color of front of Face in ESP either R,G,B in three 0-1 reals or \$red, \$lred, \$green, \$lgreen, \$blue, \$lblue, \$yellow, \$magenta, \$cyan, \$white, or \$black
<code>_bcolor</code>	color of back of Face in ESP (see <code>_color</code> )
<code>_gcolor</code>	color of grid of Face in ESP (see <code>_color</code> )
<code>_viz</code>	if set to \$off, then Face is initially not shown
<code>_grd</code>	if set to \$on, the grid on Face is initially shown
<code>_trn</code>	if set to \$on, then Face is initially shown transparent
<code>_makeQuads</code>	to make quads for this Face
<code>_stlColor</code>	color to use for this Face in an .stl file



# Special User-defined Attributes for Edges

<code>_color</code>	color of Edge in ESP either R,G,B in three 0-1 reals or \$red, \$lred, \$green, \$lgreen, \$blue, \$lblue, \$yellow, \$magenta, \$cyan, \$white, or \$black
<code>_gcolor</code>	color of grid of Edge in ESP (see <code>_color</code> )
<code>_viz</code>	if set to \$off, then Edge is initially not shown
<code>_grd</code>	if set to \$on, then grid on Edge is initially shown
<code>_ori</code>	if set to \$on, then orientation is initially shown for Edge



# Special User-defined Attributes for Nodes

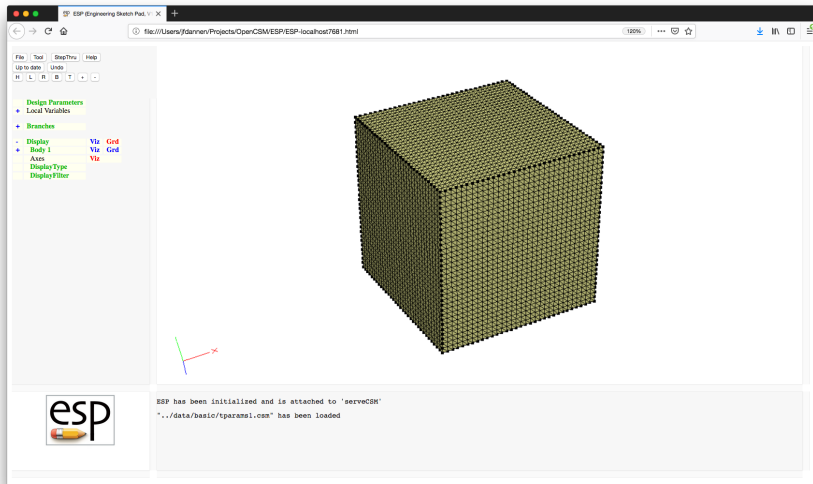
`_color`      color of Node in ESP  
              either R,G,B in three 0-1 reals  
              or \$red, \$lred, \$green, \$lgreen, \$blue, \$lblue,  
              \$yellow, \$magenta, \$cyan, \$white, or \$black

`_viz`        if set to \$off, then Edge is initially not shown

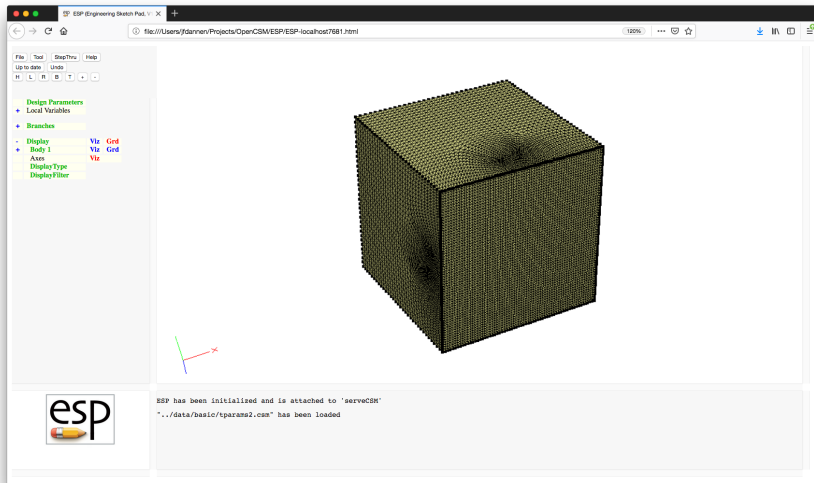


# Tessellation Parameters (1)

Default tessellation has `_tParams = 0.043; 0.013; 20`

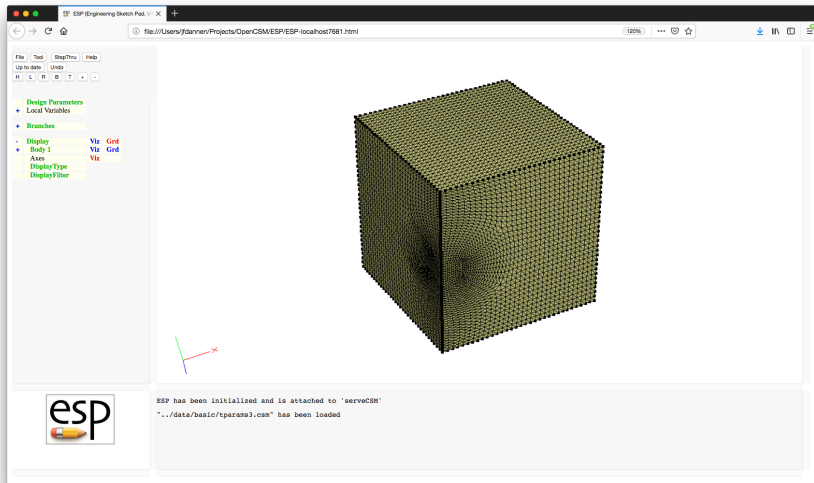


```
SELECT      FACE    1    6
ATTRIBUTE .tParams "0.02; 0.013; 20"
```



# Tessellation Parameters (1)

```
SELECT      EDGE  1   1   1   6
ATTRIBUTE .tParams "0.02; 0.013; 20"
```



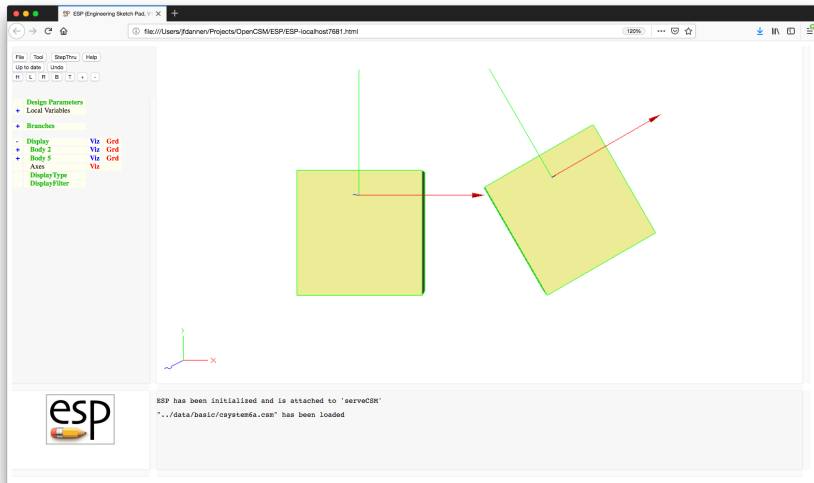
- Csystems (coordinate systems) are generated by the **CSYSTEM** statement and are applied to the Body on the top of the Stack
- Csystems are treated in many ways like Attributes
  - Csystem names must not be the same as an Attribute name
  - Csystems are found in **ESP** in same place as Attributes
- Csystems are transformed along with any transformations that are applied to their Body



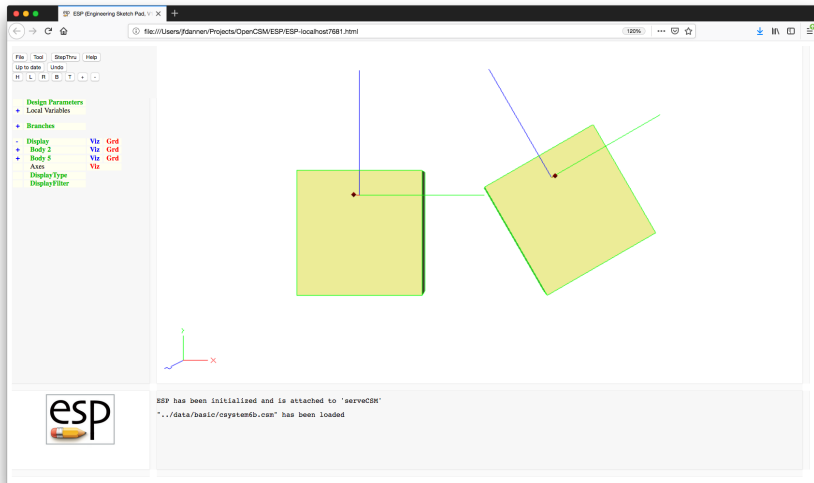
- Format of the CSYSTEM statement is:
  - If argument to CSYSTEM contains 9 entries:  
`{x0, y0, z0, dx1, dy1, dz1, dx2, dy2, dz2}`  
origin is at (x0,y0,z0)  
dirn1 is in (dx1,dy1,dz1) direction  
dirn2 is in (dx2,dy2,dz2) direction
  - If argument to CSYSTEM contains 5 entries and first is positive:  
`{+iface, ubar0, vbar0, du2, dv2}`  
origin is at normalized (ubar0,vbar0) in iface  
dirn1 is normal to Face  
dirn2 is in (du2,dv2) direction

- Format of the **CSYSTEM** statement is:
  - If argument to **CSYSTEM** contains 5 entries and first is negative:  
`{-iedge, tbar, dx2, dy2, dz2}`  
origin is at normalized (tbar) in iedge  
dirn1 is tangent to Edge  
dirn2 is part of (dx2,dy2,dz2) that is  
orthogonal to dirn1
  - If argument to **CSYSTEM** contains 7 entries:  
`{inode, dx1, dy1, dz1, dx2, dy2, dz2}`  
origin is at Node inode  
dirn1 is in (dx1,dy1,dz1) direction  
dirn2 is part of (dx1,dy2,dz2) that is  
orthogonal to dirn1

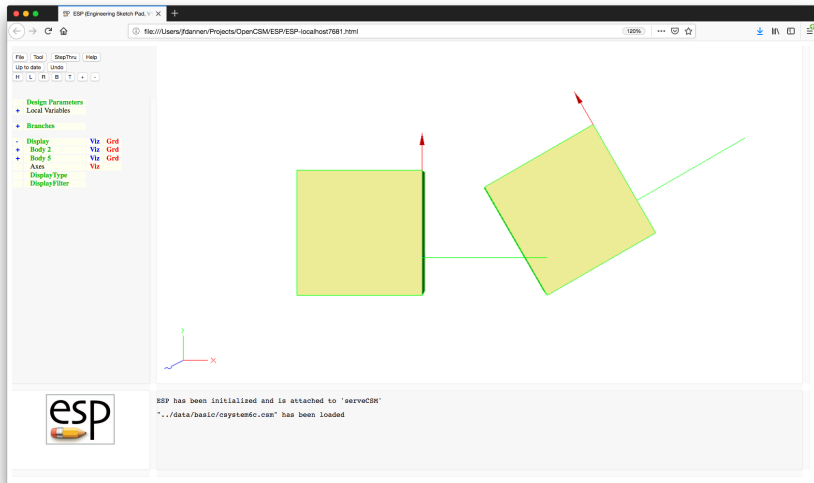
```
CSYSTEM cs1 "0.5; 0.8; 1.1;    1;0;0;    0;1;0"
```



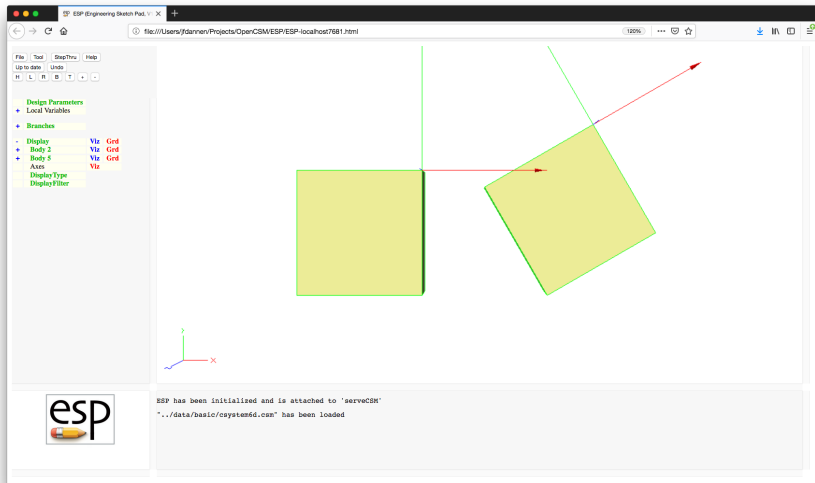
```
CSYSTEM cs1 "+6;    0.5; 0.8;    1;0"
```



```
CSYSTEM cs1 "-6; 0.3; 1;0;0"
```



```
CSYSTEM cs1 "7;    1;0;0;    0;1;0"
```





# Attribute Editor (1)

- Best practice is to set Attributes when entity is first created
- If not possible, the `editAttr` UDF is available to set Attributes based upon the Attributes of an entity's neighbors

- Statements in the attribute editor can be one of:
  - `NODE`      `<selector> <attrName1=attrValue1> ...`
  - `EDGE`      `<selector> <attrName1=attrValue1> ...`
  - `FACE`      `<selector> <attrName1=attrValue1> ...`
  - `AND`      `<selector> <attrName1=attrValue1> ...`
  - `ANDNOT` `<selector> <attrName1=attrValue1> ...`
  - `SET`                      `<attrName1=attrValue1> ...`
- Keywords can either be specified in lowercase or UPPERCASE (but not MixedCase)
- `<selector>` can be one of HAS, ADJ2NODE, ADJ2EDGE or ADJ2FACE



- Typical block of code looks like:

```
NODE ADJ2FACE tagType=spar tagIndex=1
AND  ADJ2FACE tagType=lower
AND  ADJ2EDGE tagType=root
SET                      capsConstraint=pointConstraint1
```

- Patterns can be used with PATBEG and PATEND

```
# SolidBody
BOX      0 -1 -1  3  2  2
ATTRIBUTE type $OML
SELECT   FACE  @nbody  1
ATTRIBUTE face $xmin
SELECT   FACE  @nbody  2
ATTRIBUTE face $xmax
SELECT   FACE  @nbody  3
ATTRIBUTE face $ymin
SELECT   FACE  @nbody  4
ATTRIBUTE face $ymax
SELECT   FACE  @nbody  5
ATTRIBUTE face $zmin
SELECT   FACE  @nbody  6
ATTRIBUTE face $zmax
STORE    SolidBody
```

```
# get bounding box of SolidBody
RESTORE    SolidBody
SET        xmin    @xmin
SET        xmax    @xmax
SET        ymin    @ymin
SET        ymax    @ymax
SET        zmin    @zmin
SET        zmax    @zmax
STORE      .
```

```
# Waffle (centered on SolidBody)
UDPRIM    waffle    filename <<    depth zmax-zmin+2
  POINT   A AT    xmin-1  (ymin+ymax)/2
  POINT   B AT    xmax+1  (ymin+ymax)/2
  LINE    AB  A   B   type=symmetry

  PATBEG   i   3
    POINT   C AT    xmin+i/4*(xmax-xmin) ymin-1
    POINT   D AT    xmin+i/4*(xmax-xmin) ymax+1
    LINE     .   C   D   bulkhead=!val2str(i,0)
  PATEND

>>
TRANSLATE 0  0  zmin-1
STORE      Waffle
```

```
# score the SolidBody by the Waffle and extract Faces
```

```
RESTORE    SolidBody
```

```
RESTORE    Waffle
```

```
SUBTRACT
```

```
EXTRACT    0
```

```
# generate the internal structure
```

```
RESTORE    SolidBody
```

```
RESTORE    Waffle
```

```
INTERSECT
```

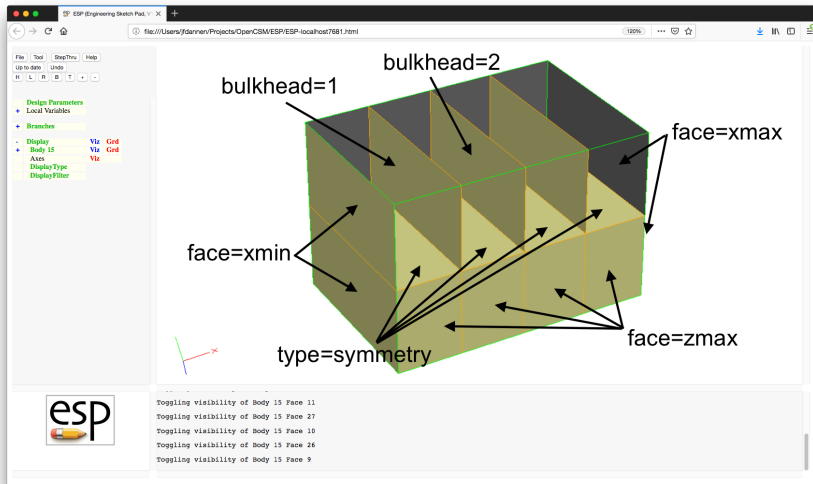
```
# put them together
```

```
UNION
```



# EditAttr Example (5)

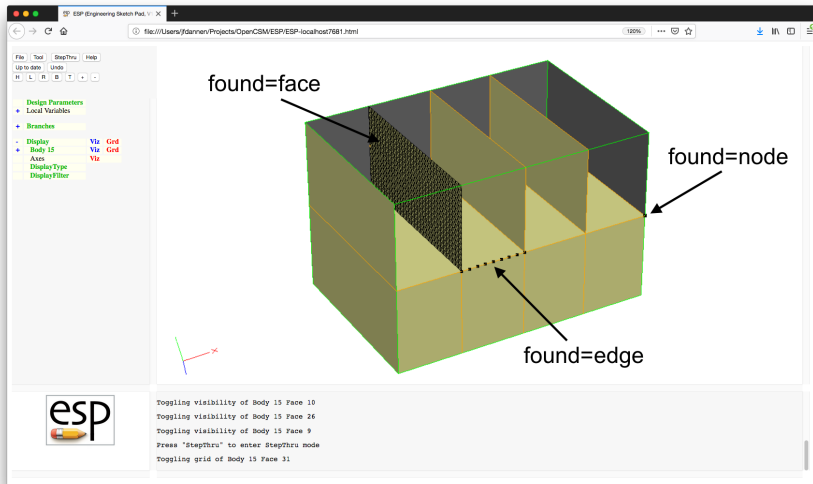
Some Faces not shown for clarity





# EditAttr Example (6)

Attributes that we want to define



```
# select Face on bulkhead=1 but top half
UDPRIM      editAttr  filename  <<
    FACE    HAS        bulkhead=1
    AND     ADJ2FACE   face=ymax
    SET                                           found=face
>>
```



```
# select Edge on OML seam between bulkheads 1 and 2
UDPRIM      editAttr  filename  <<
  NODE      ADJ2FACE  bulkhead=1
  SET                               bulkhead=1
  NODE      ADJ2FACE  bulkhead=2
  SET                               bulkhead=2
  EDGE      ADJ2FACE  face=zmax
  AND       ADJ2FACE  type=symmetry
  AND       ADJ2NODE  bulkhead=1
  AND       ADJ2NODE  bulkhead=2
  SET                               found=edge
  NODE      HAS       bulkhead=1
  SET                               bulkhead=
  NODE      HAS       bulkhead=2
  SET                               bulkhead=
>>
```

```
# select Node on OML seam at the outlet
UDPRIM      editAttr  filename  <<
  NODE      ADJ2FACE  face=xmax
  AND       ADJ2FACE  face=zmax
  AND       ADJ2FACE  type=symmetry
  SET
  >>
```

- Using  
`$ESP_ROOT/training/ESP/exercises/session08/wingStruct.csm`
  - put the Attribute `LoadPoint=leftTip` on the Node that is at the intersection of the forward spar, wing tip, and upper skin on the left wing
  - for the skin panels on the right wing that are between the first and second rib, make their color red and their grid white
  - make the Edges blue that are between two red panels