Engineering Sketch Pad (ESP)

Training Session 8
Selection & Attribution

John F. Dannenhoffer, III
jfdannen@syr.edu
Syracuse University

Bob Haimes
haimes@mit.edu
Massachusetts Institute of Technology

updated for v1.19

Overview

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

Attribute Overview

- 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

Attribute Rules

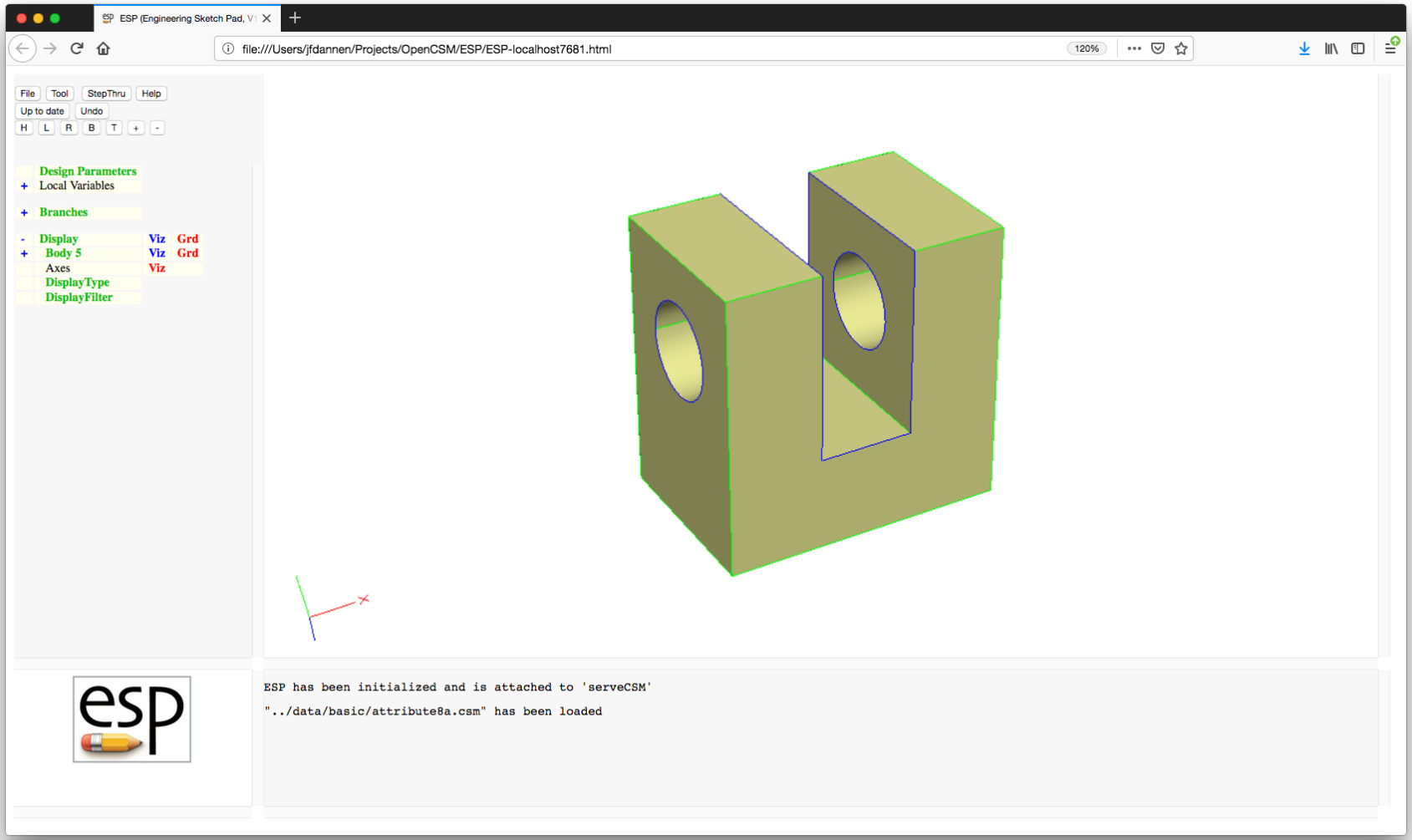
- Attributes can be associated with any Branch that produces a Body or the `SELECT` statement
- Attributes are defined by an `ATTRIBUTE` statement
- Take care when starting an Attribute name with a period (which is reserved for `EGADS`) or an underscore (which is reserved for `OpenCSM`)
- If the first 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)

Setting Attributes

- 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



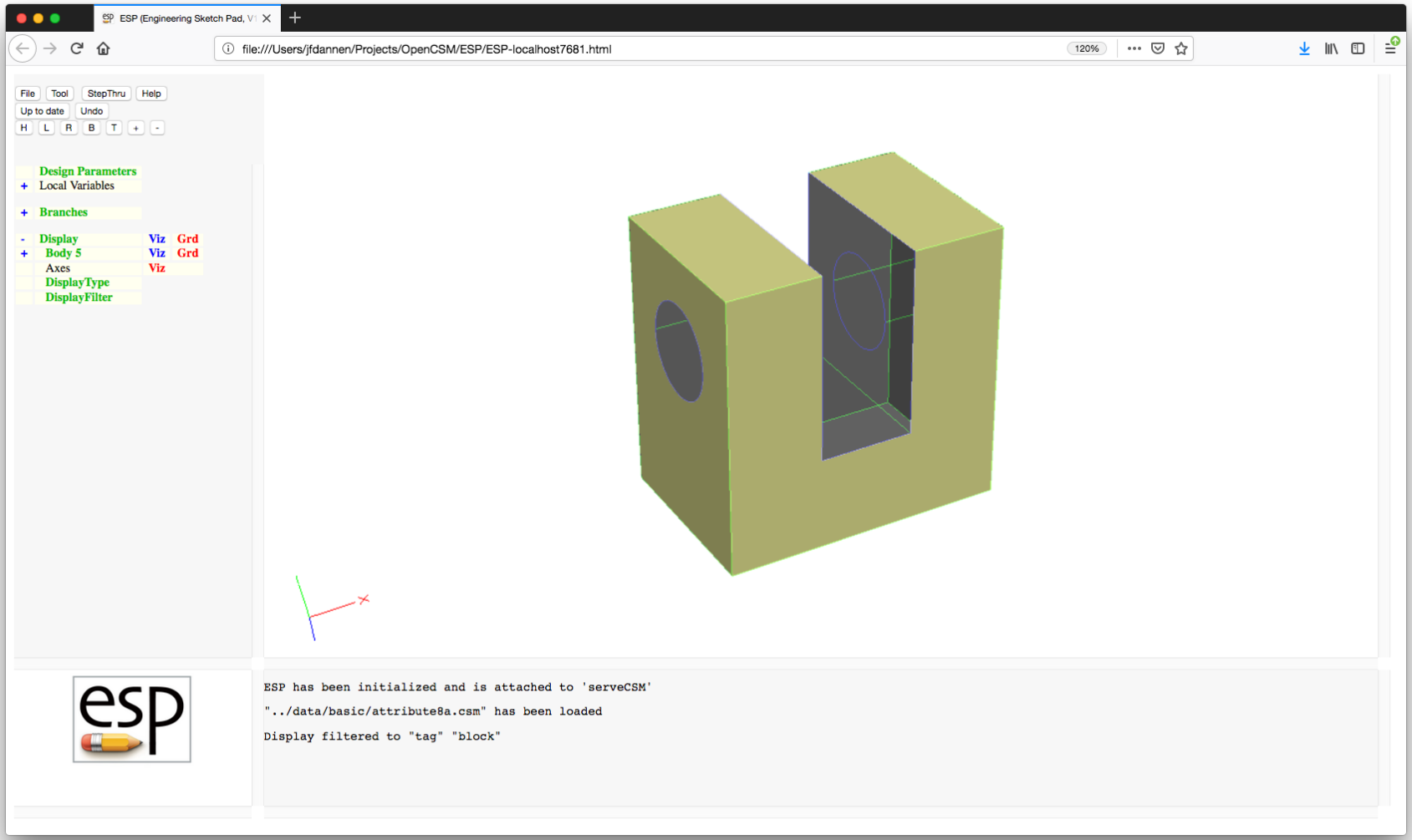
Attribute Example (2)

.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
```

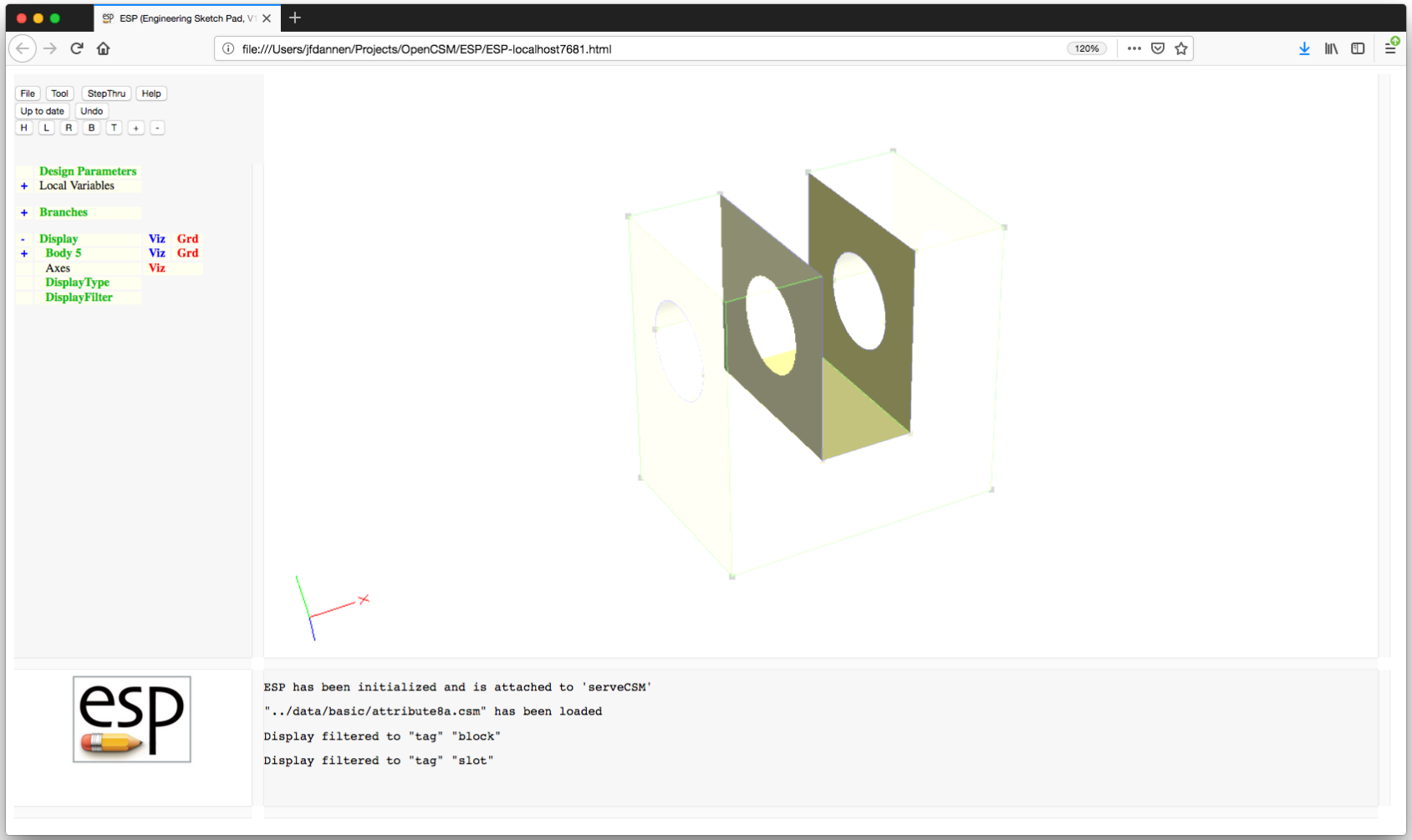
Attribute Example (3)

DisplayFilter to `tag block`



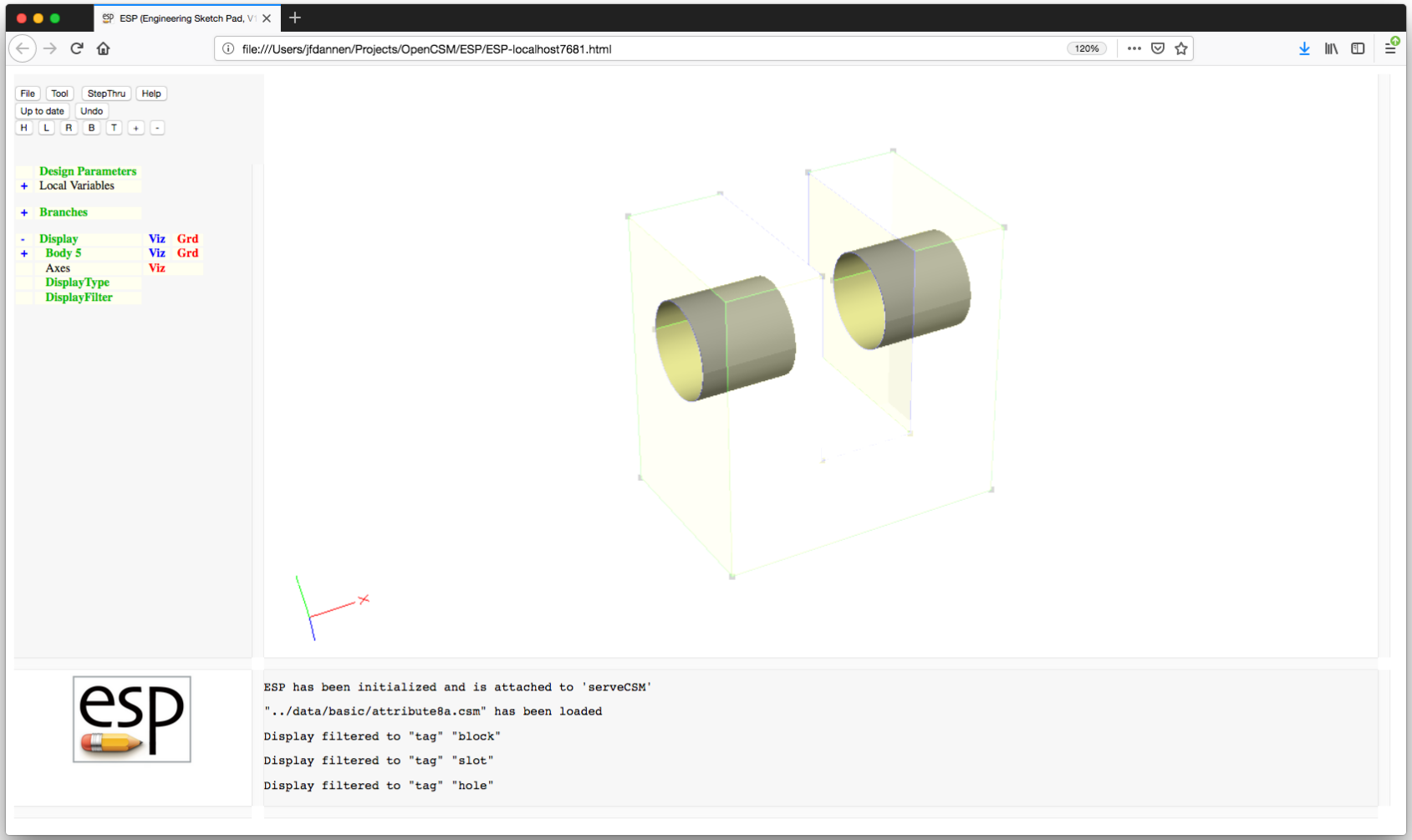
Attribute Example (4)

DisplayFilter to `tag slot`



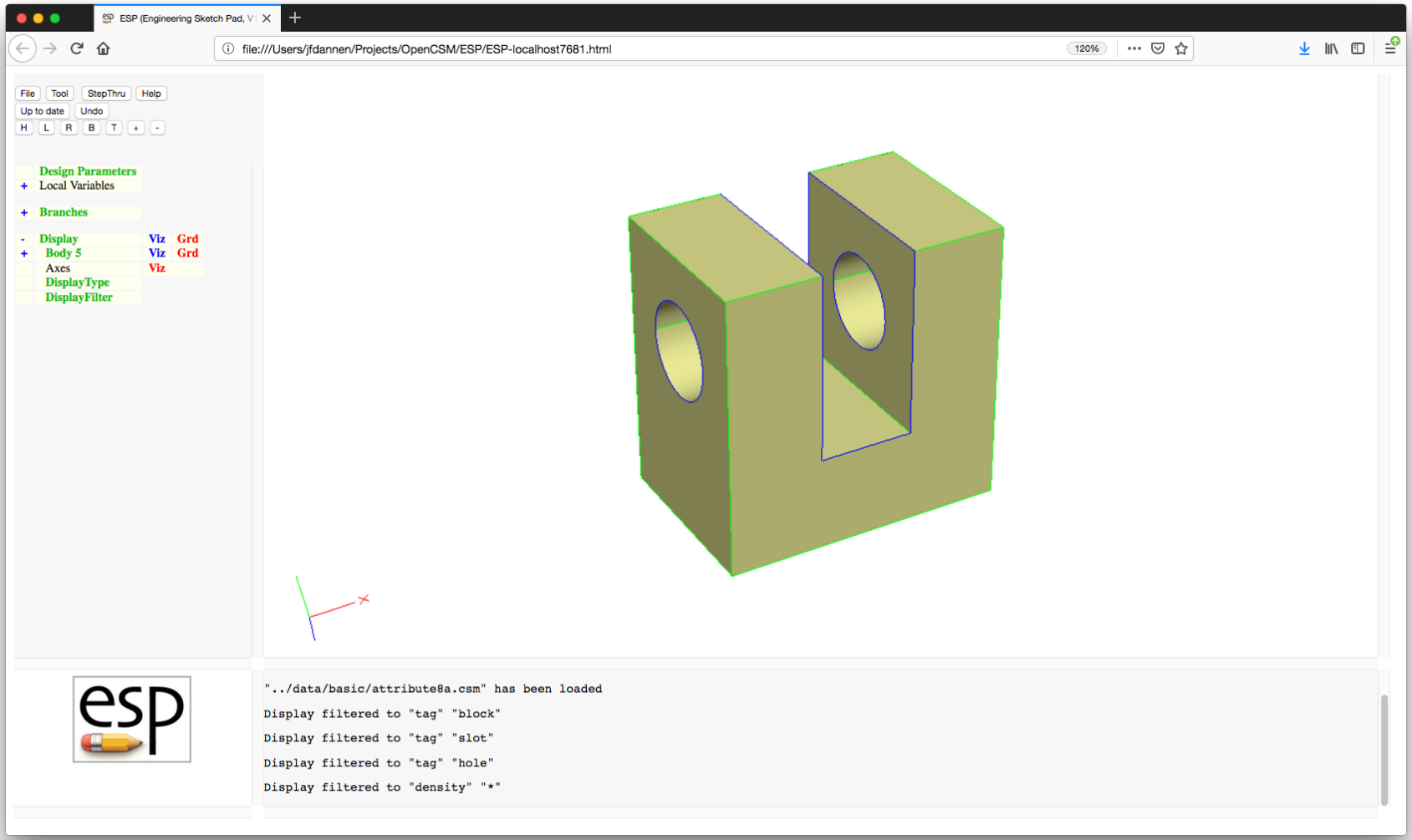
Attribute Example (5)

DisplayFilter to `tag hole`



Attribute Example (6)

DisplayFilter to `density *`

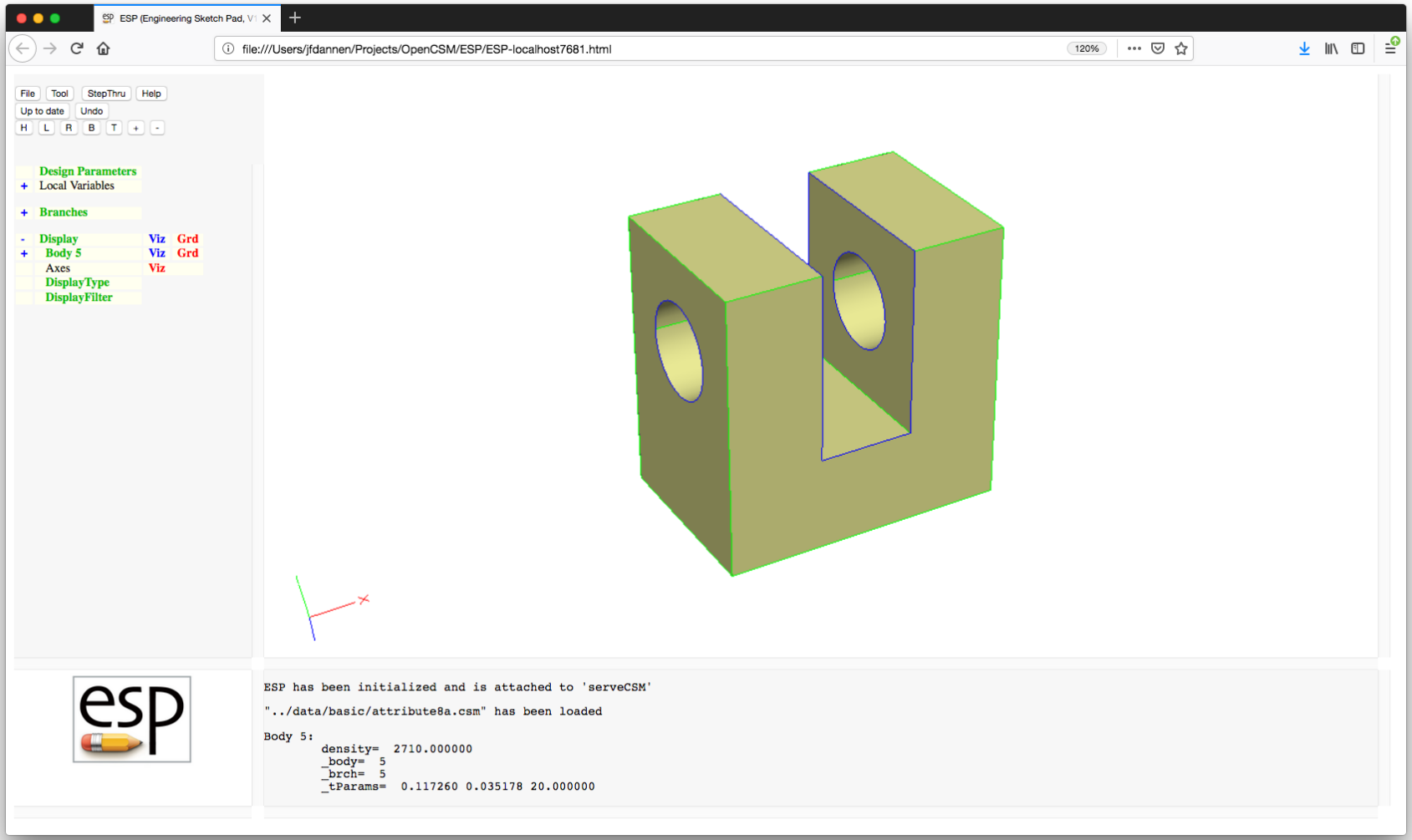


Viewing Attributes

- 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)

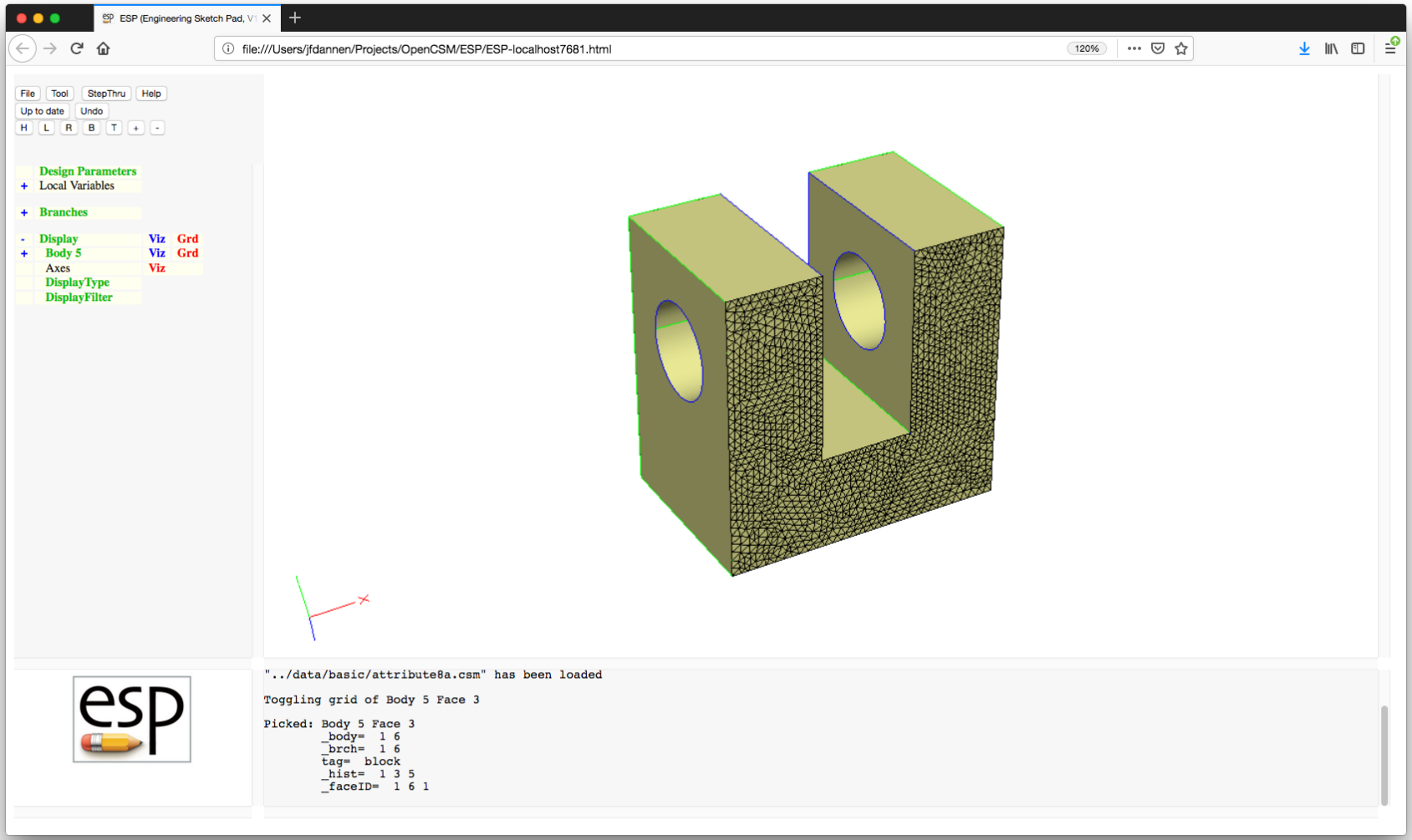
Attribute Example (6)

After pressing **Body 5** in TreeWindow

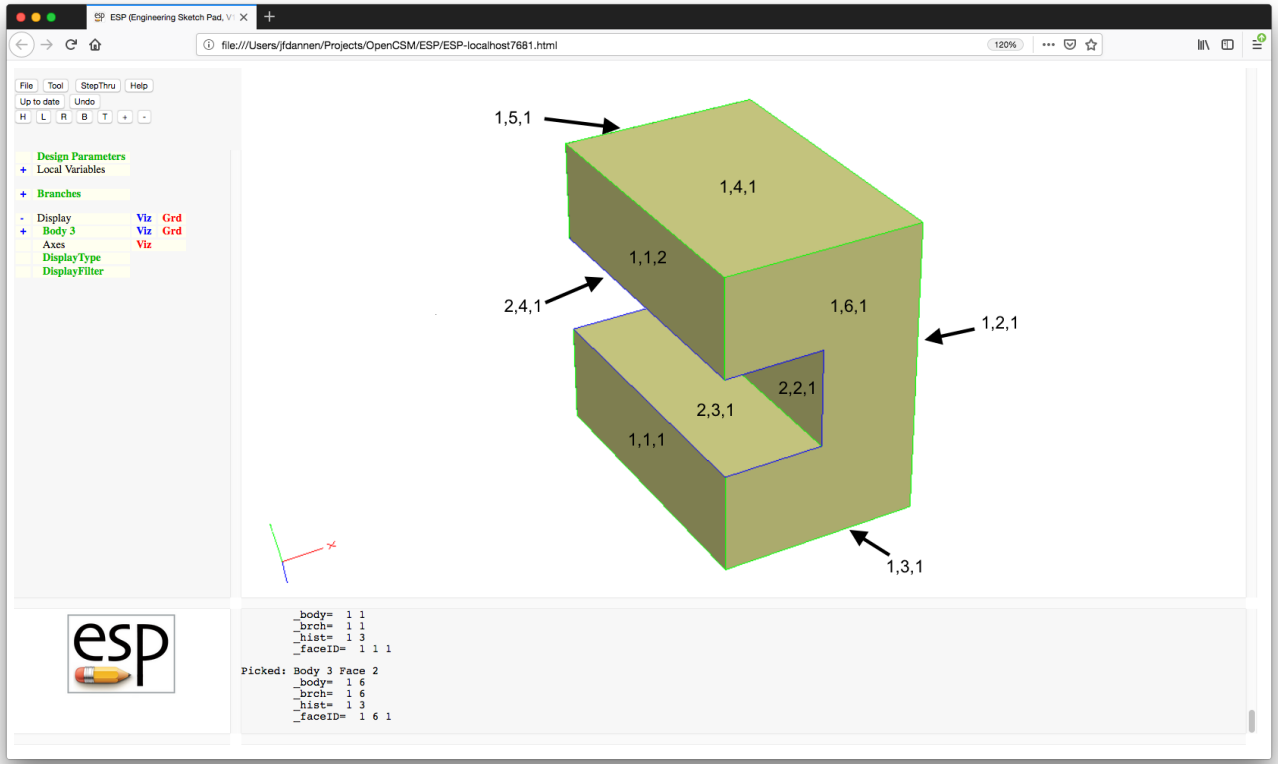


Attribute Example (6)

After pressing ∧ on Face with Grid



ibody/iford Refresher




```
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

SELECT Statement

- 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`

Selecting a Body

- `SELECT BODY` — selects last Body created
- `SELECT BODY ibody` — selects Body `ibody`
- `SELECT BODY -n` – selects the nth Body from the top of the Stack
- `SELECT BODY $attrName1 attrValue1 ...` — selects the last Body that matches all the given Attributes

Selecting Face(s)

- `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` — selects the Faces totally within the specified bounding box
- `SELECT FACE $attrName1 attrValue1 ...` – selects the Faces that matches all the given Attributes

Selecting Edge(s)

- `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` — selects the Edges totally within the specified bounding box
- `SELECT EDGE xmid ymid zmid` — selects the Edge whose midpoint is closest to the given coordinates
- `SELECT EDGE $attrName1 attrValue1 ...` – selects the Edges that matches all the given Attributes

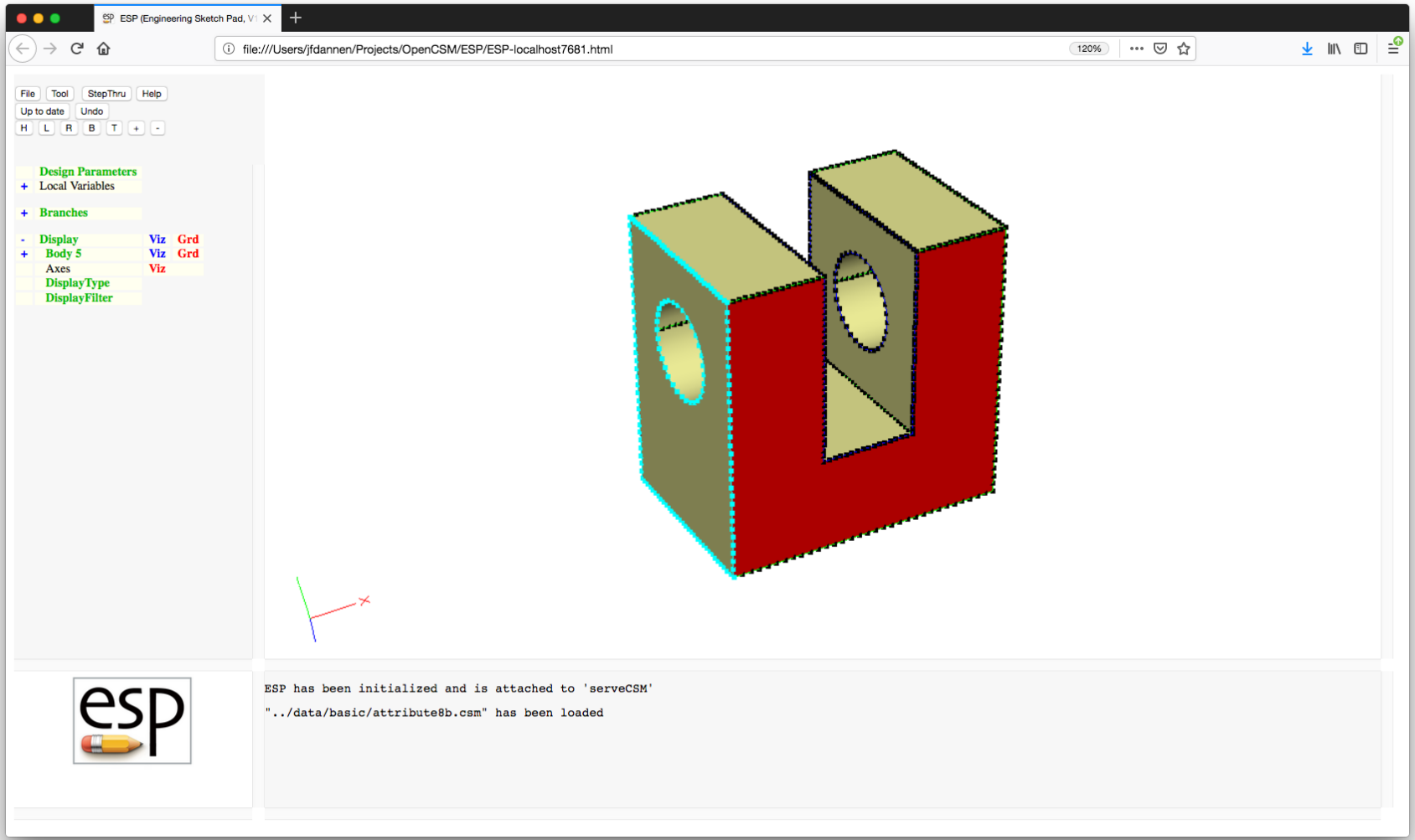
Selecting Node(s)

- `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 $attrName1 attrValue1 ...` – selects the Nodes that matches all the given Attributes

Selection Example (1)

```
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
```

Selection Example (2)



Modifying the Selection List

- Use `SELECT ADD ...` to add Faces, Edges, or Nodes to the selection list
- Use `SELECT SUB ...` to remove Faces, Edges, or Nodes from the selection List
- Both of these option use the selection type from the previous selection

- Use `SELECT SORT $key` to sort `@sellist` based upon `$xmin`, `$ymin`, `$zmin`, `$xmax`, `$ymax`, `$zmax`, `$xcg`, `$ycg`, `$zcg`, `$length` (if Edges), or `$area` (if Faces)

Attributes Automatically Set to Bodys

```
_body       Body index (bias-1)

_brch       Branch index (bias-1)

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

_csys_*     arguments when CSYSTEM was defined

<any>       all global attributes

<any>       all attributes associated with Branch that created Body

<any>       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)
```

Attributes Automatically Set to Faces (2)

```
_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
```

Attributes Automatically Set to Edges (1)

```
_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

...
```

Attributes Automatically Set to Edges (2)

```
...
            _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

```
_nodeID     unique integer

_nedge      number of incident Edges

<any>       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
```

Special User-defined Attributes for Faces

```
_color      color of front of Face in ESP
            either R,G,B in three 0-1 reals
            or $red, $green, $blue, $yellow, $magenta,
            $cyan, $white, or $black

_bcolor     color of back of Face in ESP (see _color)

_gcolor     color of grid of Face in ESP (see _color)

_makeQuads  to make quads for this Face

_stlColor   color to use for this Face in an .stl file
```

Special User-defined Attributes for Edges

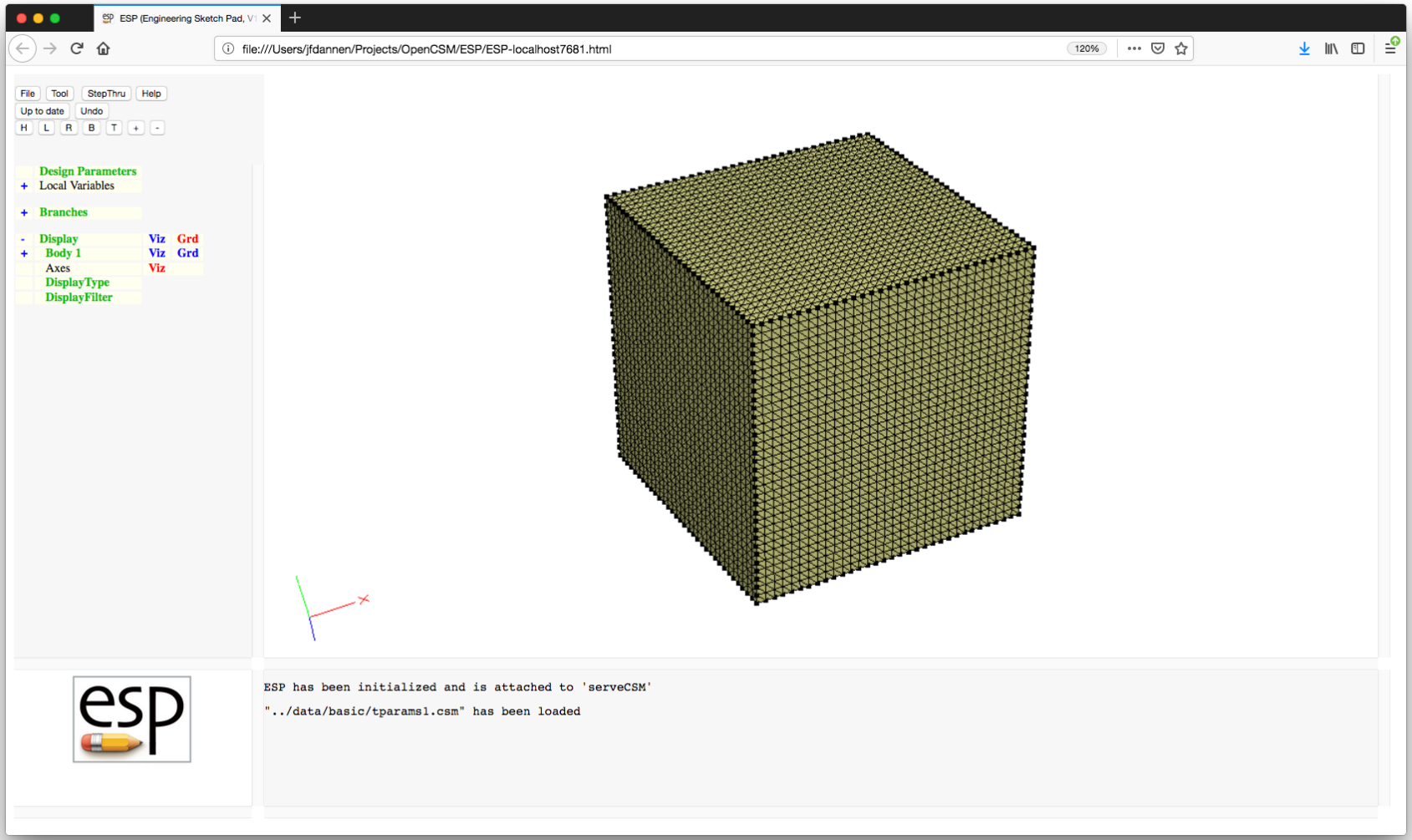
```
_color      color of front of Edge in ESP
            either R,G,B in three 0-1 reals
            or $red, $green, $blue, $yellow, $magenta,
            $cyan, $white, or $black

_gcolor     color of grid of Edge in ESP (see _color)
```

Special User-defined Attributes for Nodes

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

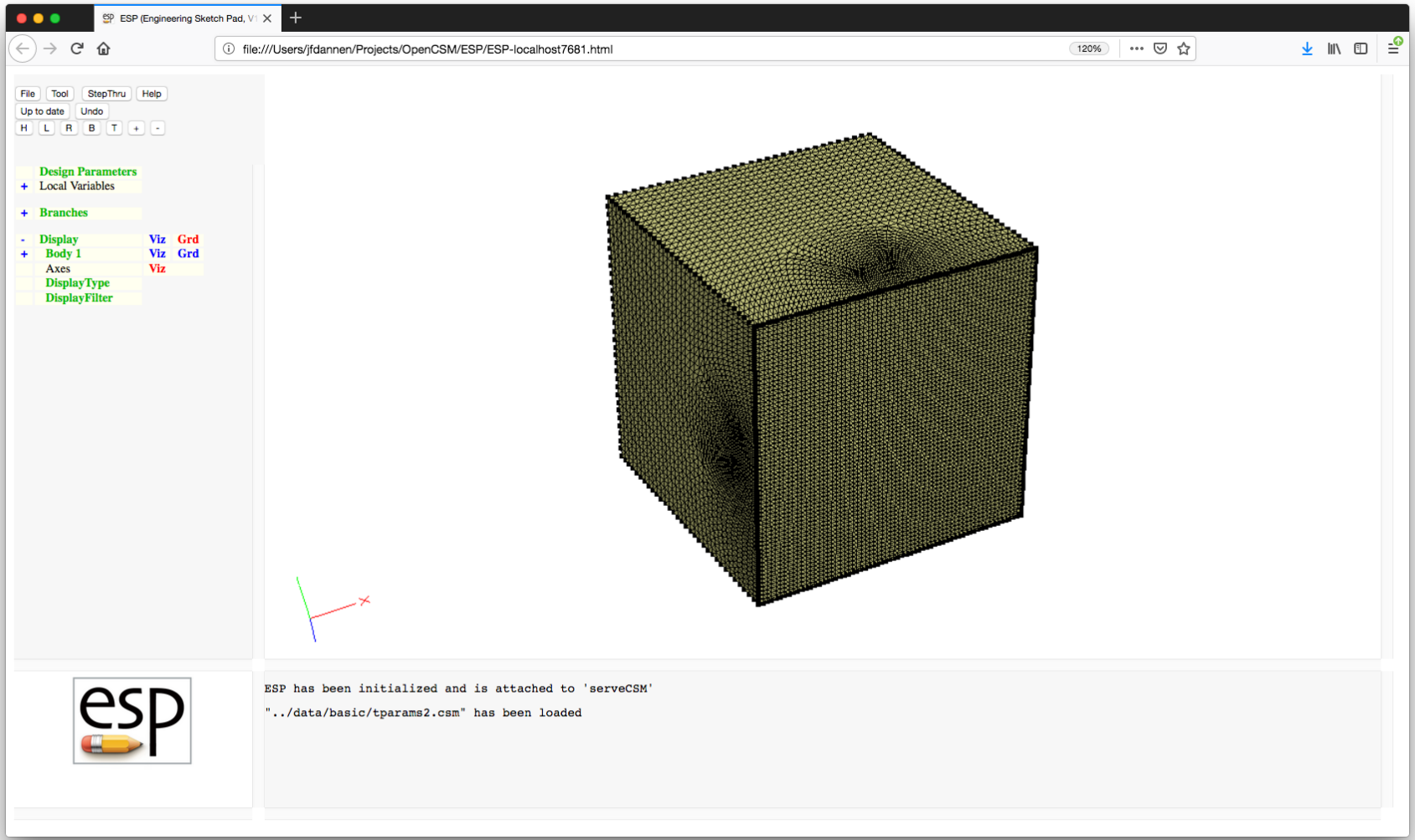
Tessellation Parameters (1)

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



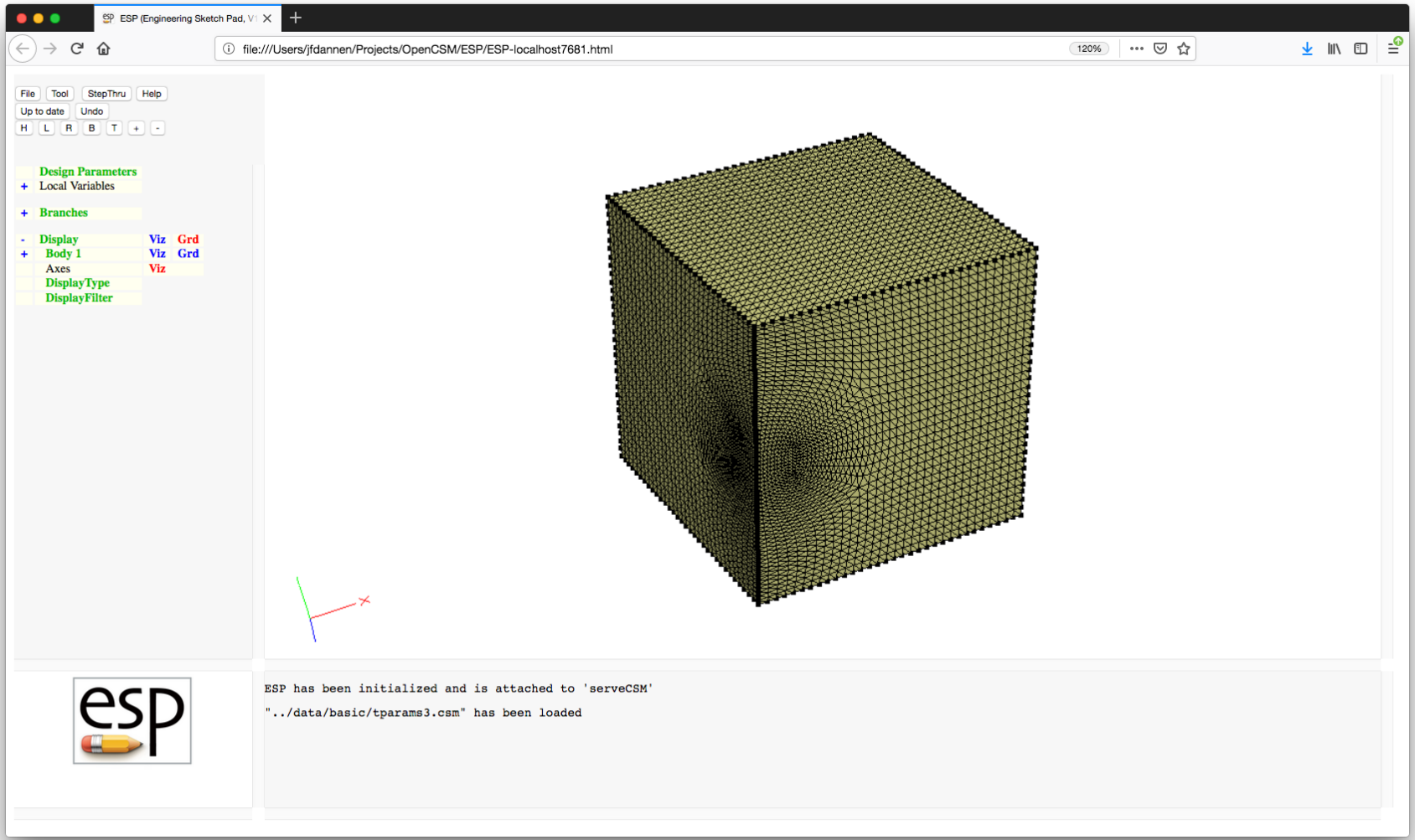
Tessellation Parameters (2)

```
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"
```



Csystem rules (1)

- 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

Csystem rules (2)

- 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
```

Csystem rules (3)

- Format of the `CSYSTEM` statement is:
  - If argument to `CSYSTEM` contains 5 entries and first is negative:

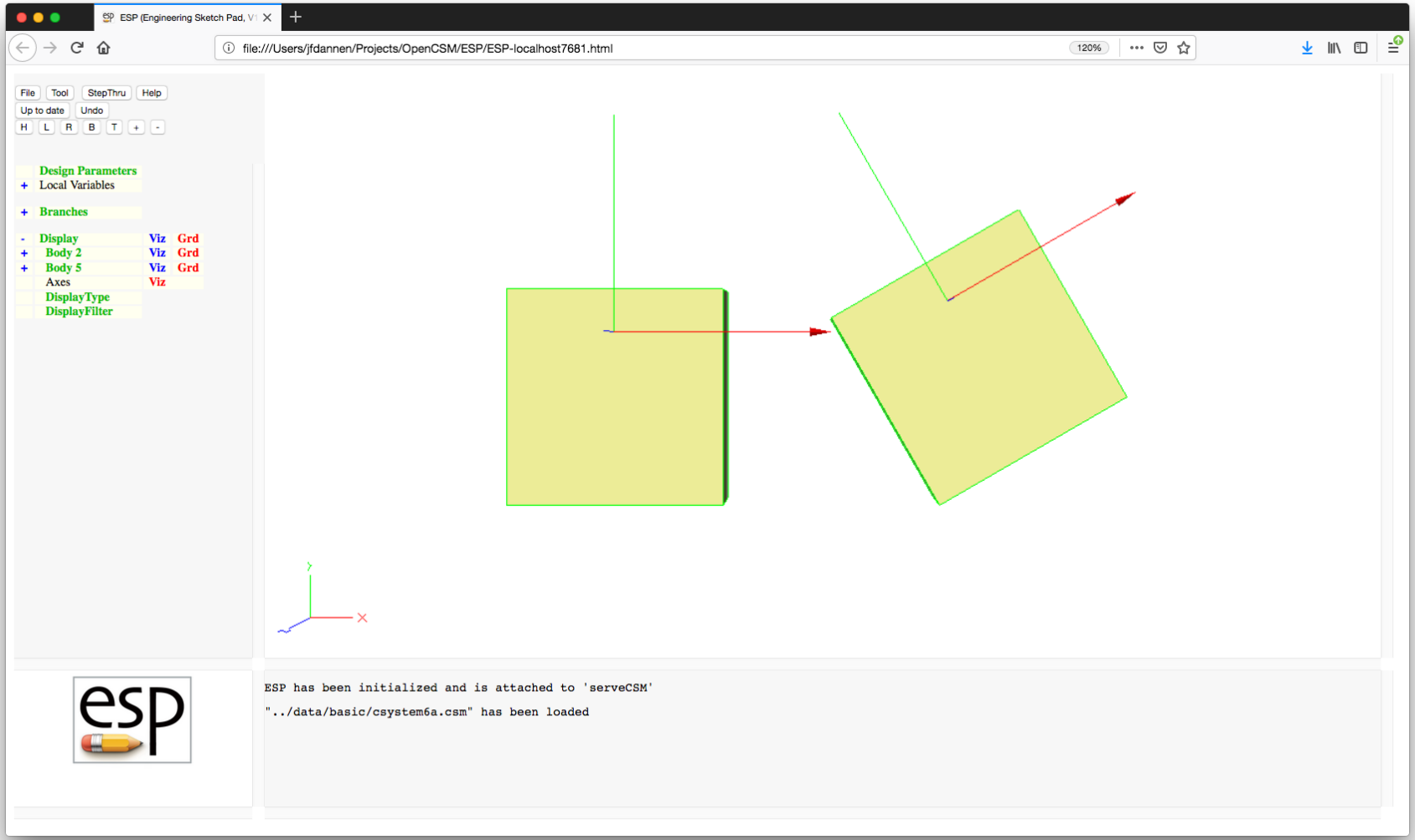
```
{-iedge, tbar, dx2, dy2, dz2}
origin is at normalized (tbar) in iedge
dirn1  is tangent to Edge
dirn2  is part of (dx2,dy2,dz2) that is
                orthogonal to dirn1
```

  - 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
```

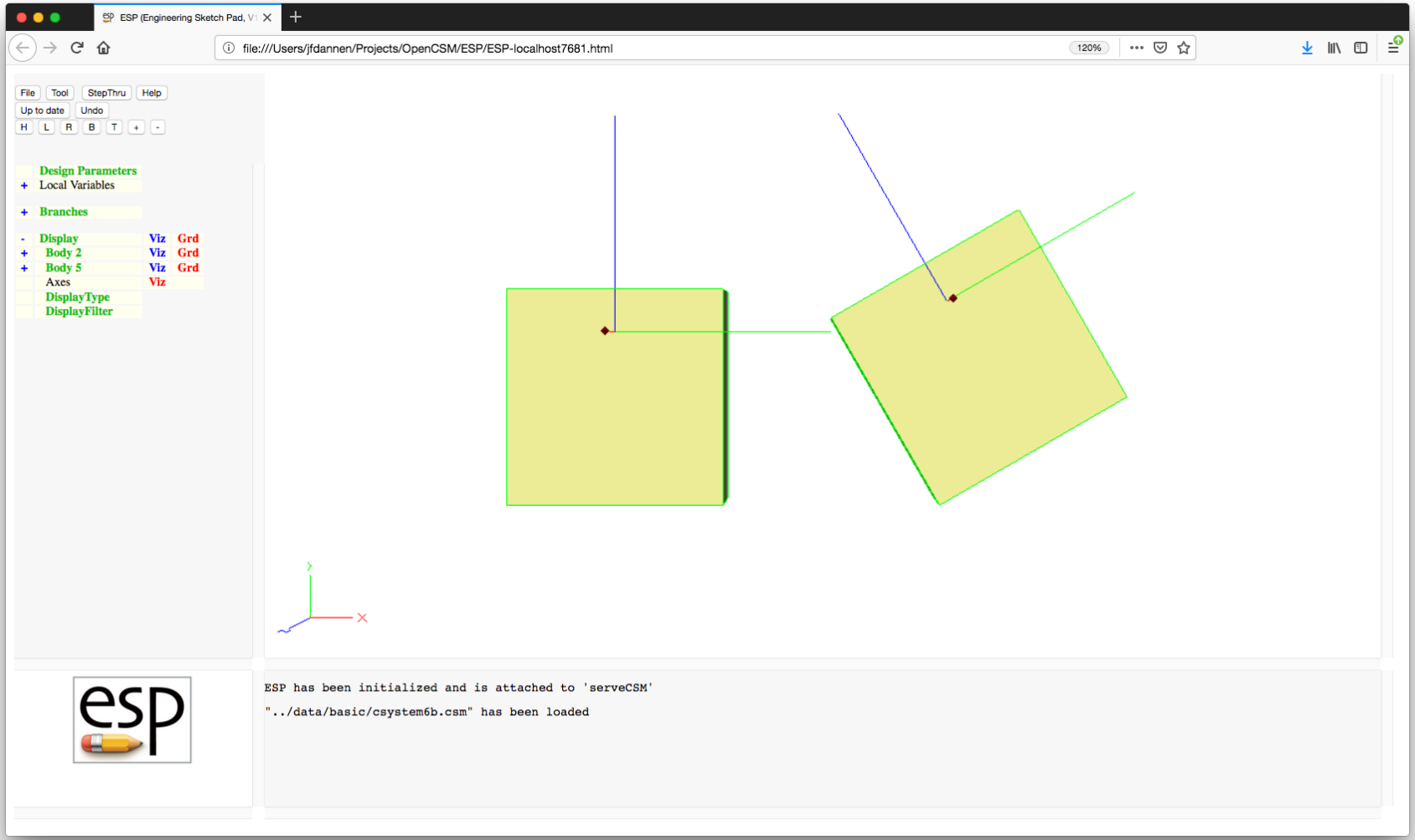
Csystem Defined with respect to Body

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



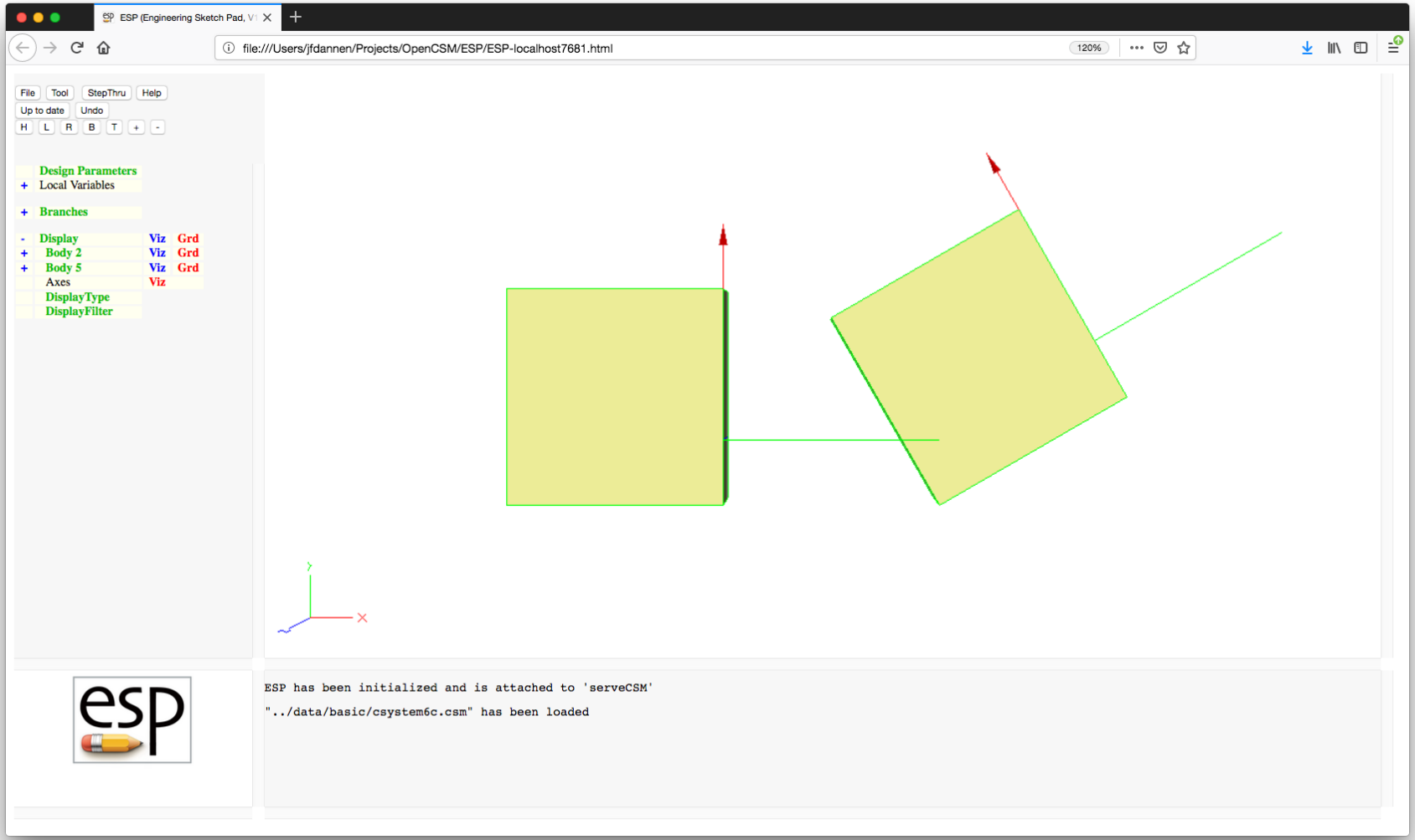
Csystem Defined with respect to Face 6

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



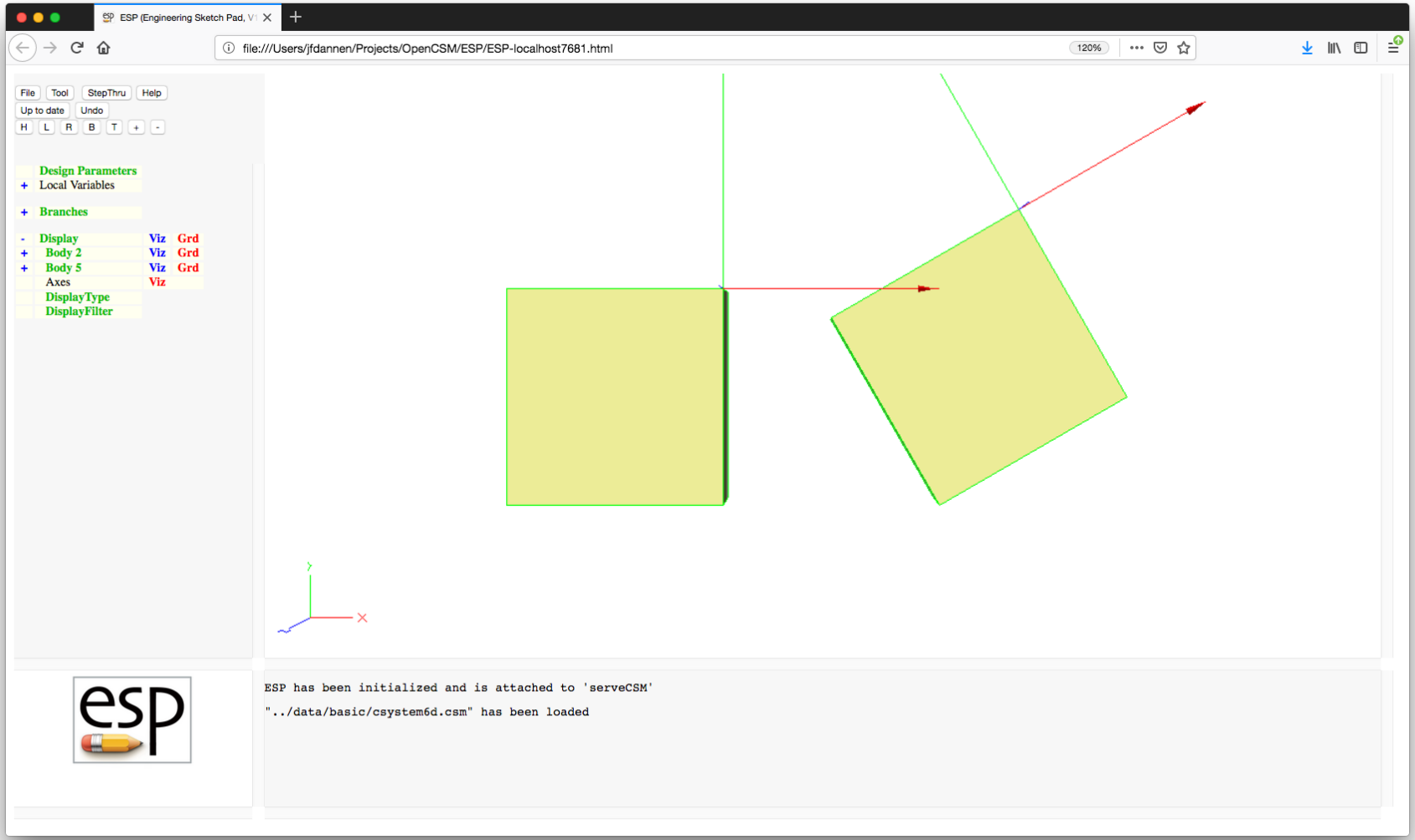
Csystem Defined with respect to Edge 6

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



Csystem Defined with respect to Node 7

```
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

Attribute Editor (2)

- 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
- `<selector>` can be one of `HAS`, `ADJ2NODE`, `ADJ2EDGE` or `ADJ2FACE`

Attribute Editor (3)

- 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`

EditAttr Example (1)

```
# 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
```

EditAttr Example (2)

```
# 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     .
```

EditAttr Example (3)

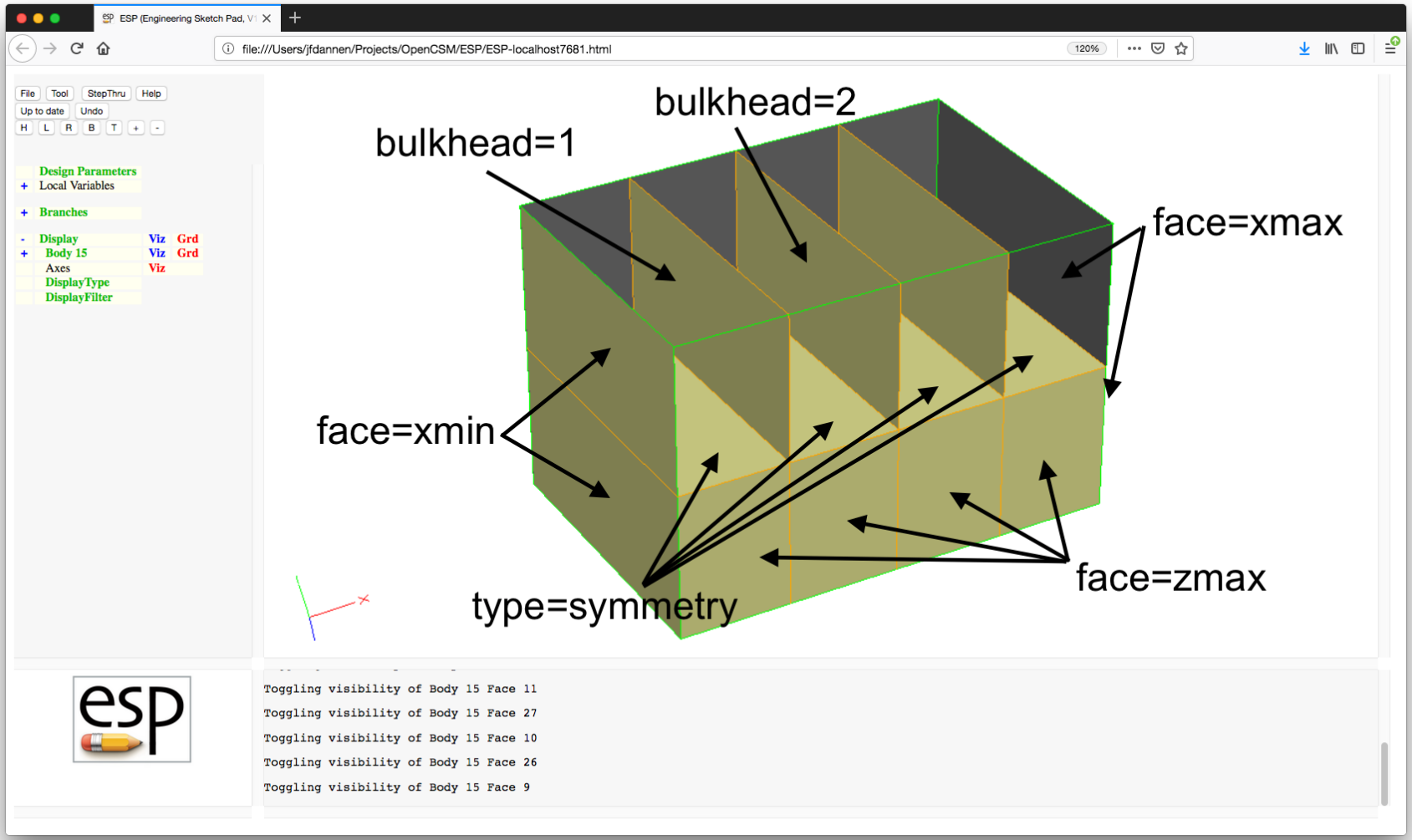
```
# Waffle (centered on SolidBody)
UDPRIM    waffle   filename <<   depth zmax-zmin+2
   POINT  A AT  xmin-1  (ymin+ymax)/2
   POINT  B AT  xmax+1  (ymin+ymax)/2
   LINE   AB  A  B  type=symmetry

   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
```

EditAttr Example (4)

```
# 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
```

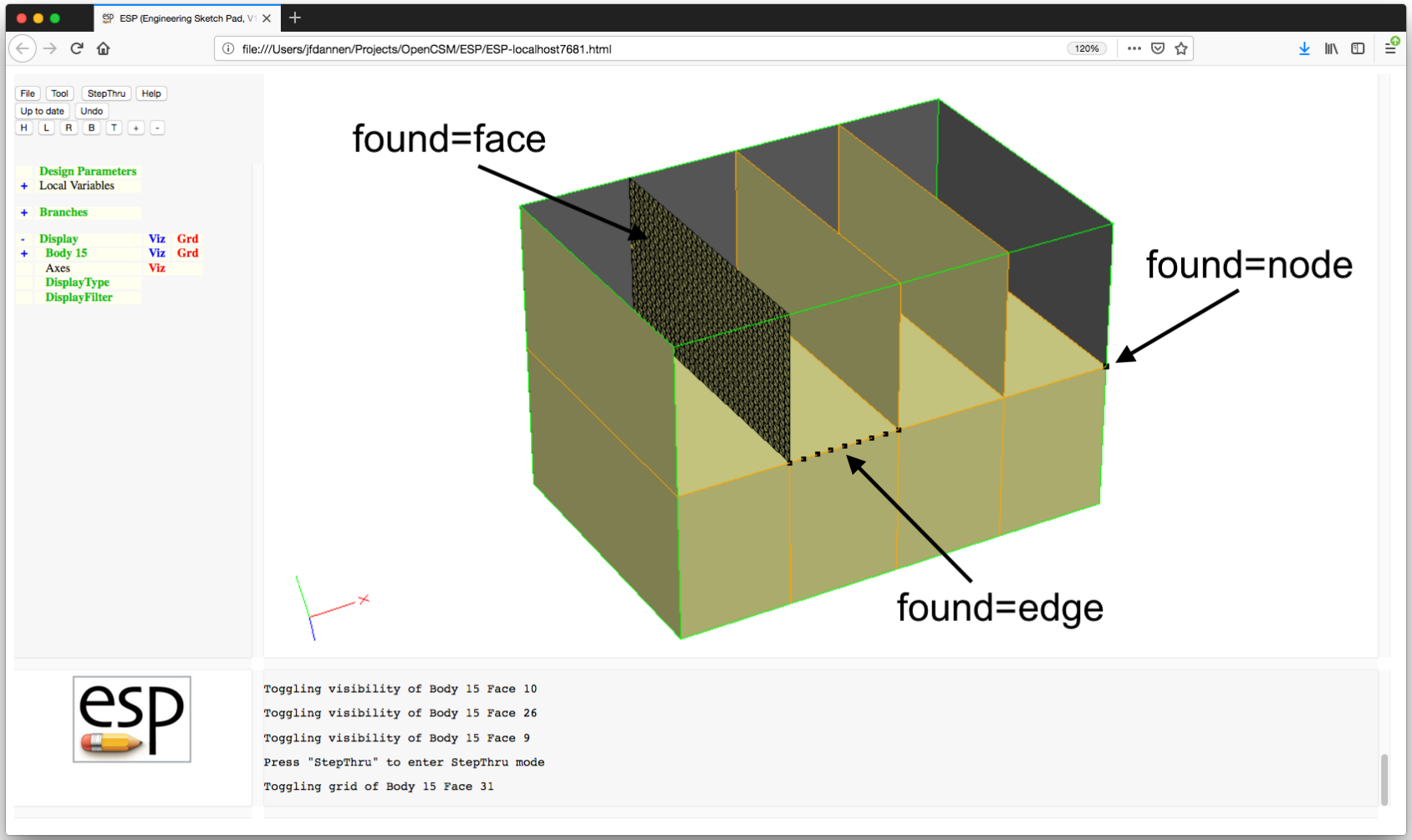
EditAttr Example (5)

Some Faces not shown for clarity



EditAttr Example (6)

Attributes that we want to define



EditAttr Example (7)

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

EditAttr Example (8)

```
# 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=
>>
```

EditAttr Example (9)

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

Homework Exercise

- Using `$ESP_ROOT/training/ESP/data/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 rite 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