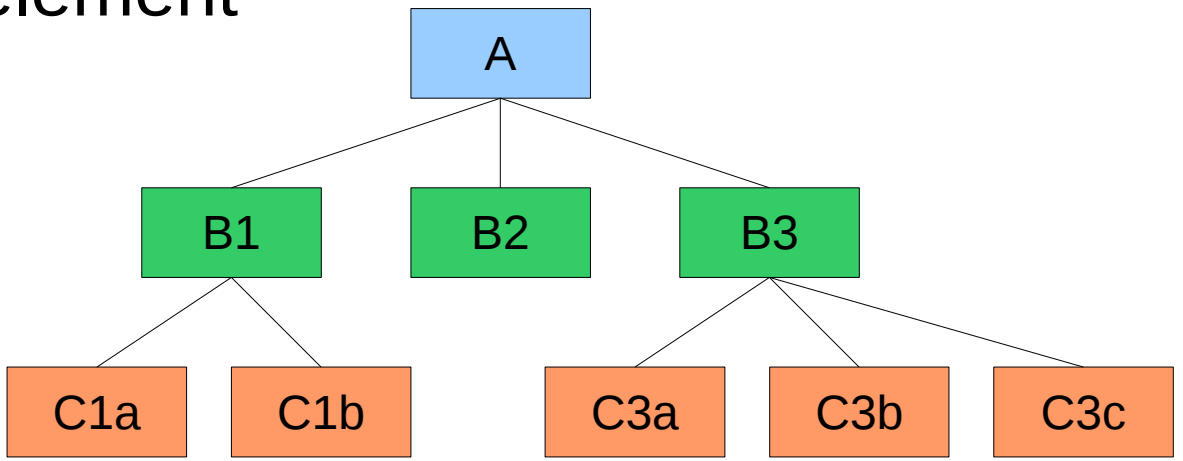
JEOD Training

Reference Frame Tree
Mass Tree

Tree Concept

- See Chapters D and E in course_text.pdf
- Main concepts:
  - One Root per tree
  - One parent per element
    - Except Root
  - Multiple children
  - Multiple siblings

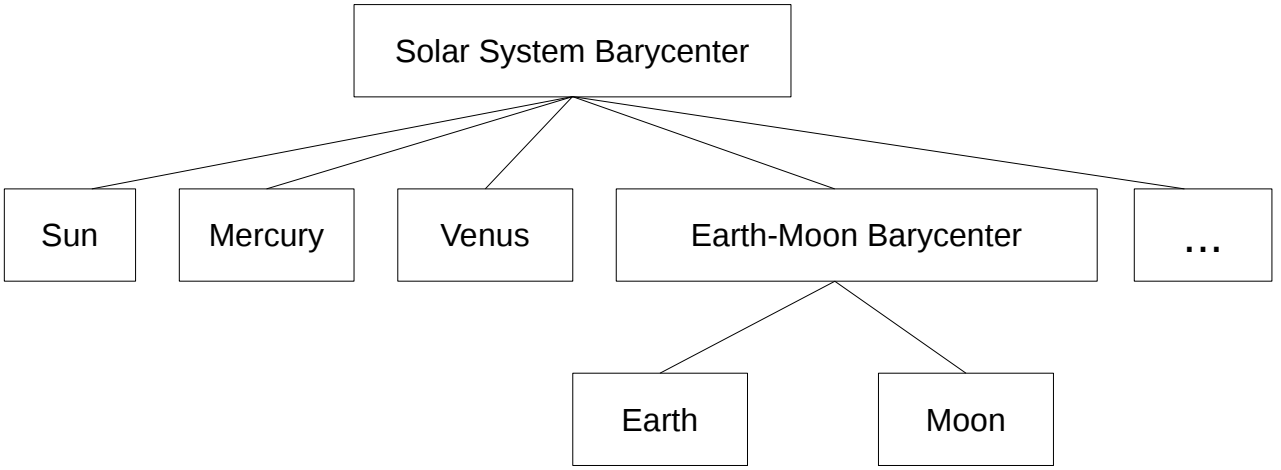


Reference Frame Tree

- One reference frame tree per sim
- Reference Frames have states
  - States are known relative to parent
    - position, velocity
    - ang_vel_this, T_parent_this, Q_parent_this
      - Quaternions are always left-handed transformation quaternions
  - States can be obtained relative to any other frame by traversing the tree (automatic tools available)
- Reference Frames may be picked up and moved
- Ephemerides auto-added
- User-added

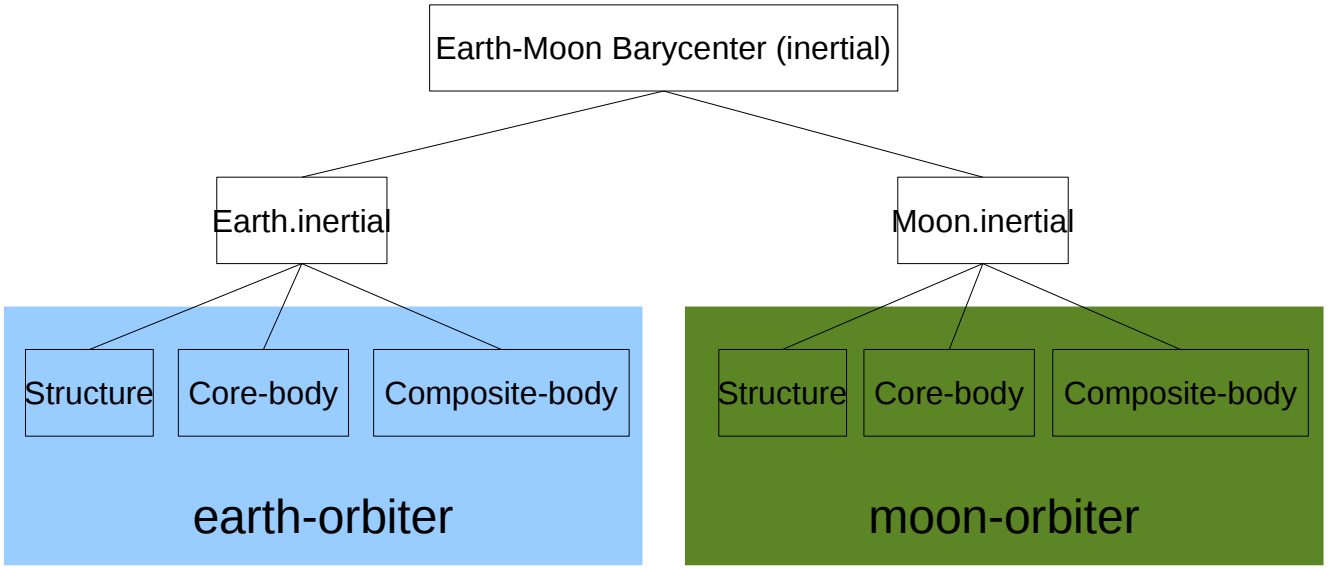
Ephemerides

- Uses DE405, DE421 or SPICE.
  - No DE430 (yet)
- Inertial frames may be used as integration frames
  - “<planet>.inertial”



Extending the Tree

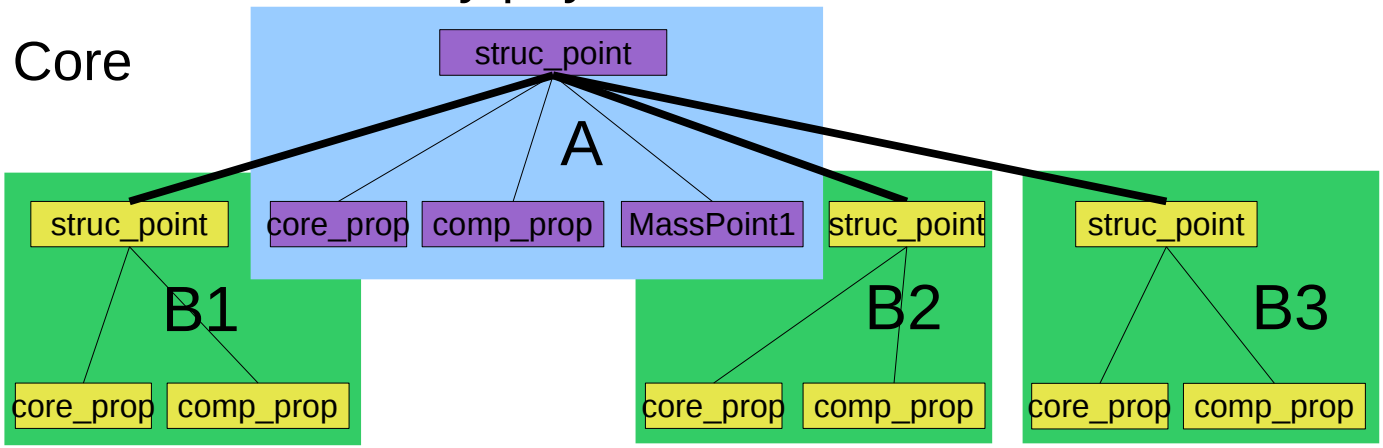
- Vehicles are assigned an “integration frame”
  - Vehicle frames have states relative to this frame
  - Vehicle frames are added as children of this frame



- Additional frames added as needed.
  - More on this later

Mass Tree

- One mass tree per contiguous group of bodies
  - Hierarchy, what is attached ***to*** what
  - Attachments abstract; not necessarily physical
  - Composite – vs. - Core



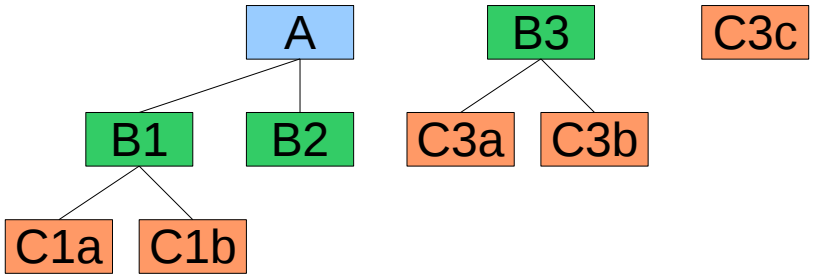
- Each body has:
  - Structure point (*MassBasicPoint*),
  - Core properties (*MassProperties*),
  - Composite properties (*MassProperties*),
  - Mass Points (*MassPoint*, optional)

Adding Points of Interest

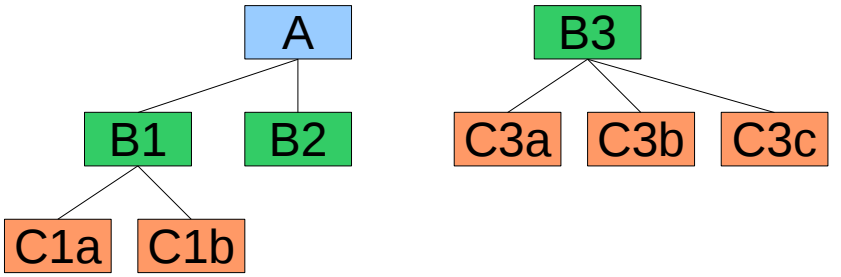
- Points of interest (*MassPoint*) can be added to a *MassBody* with the *MassBodyInit* body-action.
- Inheritance Hierarchy:
  - *MassPoint* is a *MassBasicPoint* with:
    - Name
  - *MassBasicPoint* (e.g. *structure_point*) is a *MassPointState*
  - *MassPointState* has:
    - Position
    - Orientation
  - *MassProperties* (e.g. *core_properties*, *composite_properties*) is a *MassBasicPoint* with:
    - Mass
    - Inertia tensor

Manipulating the Mass Tree

- Attachments

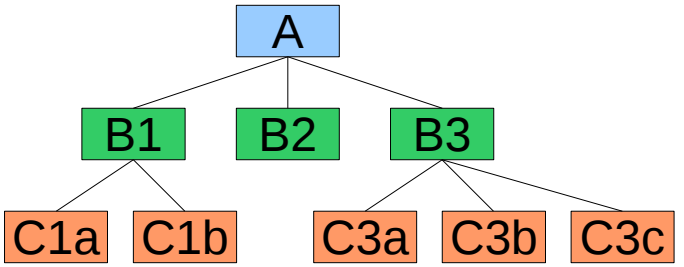


  - C3c to B3:



  - B3 to A or C3* to A:

    (visual - abstract nature)



Reference Frame Tree and Mass Tree

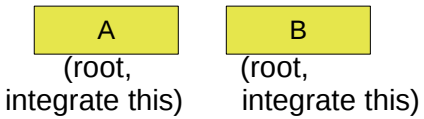
- Concepts:
  - composite_body, composite_properties
  - core_body, core_properties
  - structure, structure_point
  - DynBody, MassBody
- DynBody is a MassBody.
  - sub-frames associated with mass-points
  - composite_body frame:
    - Has same position as composite_properties
      - composite_body is known relative to inertial frame
      - composite_properties is known relative to structural
    - Has same orientation as composite_properties
      - Likewise
      - Thus inertial position and orientation of structural is defined
  - Etc.

Adding Reference Frames (and Mass Points)

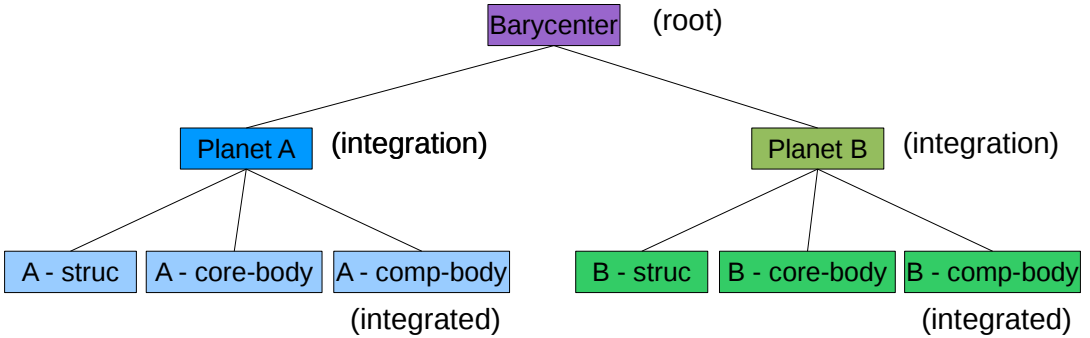
- Simplest method to add a reference frame is to add a mass-point to a *DynBody* via the Body-action *MassBodyInit*
  - Creates a new *BodyRefFrame*
  - Adds new frame as a child of *DynBody's* integration frame
  - Adds the new frame to the Dynamics Manager for maintenance
- Alternative:
  - Create a frame
  - Add it to the tree (add_frame_to_tree)
  - Add it to the Dynamics Manager (add_frame)
  - Subscribe to it
- Recover a MassPoint with *find_mass_point*
- Recover a BodyRefFrame with *find_vehicle_point*

Response of Reference Frame Tree to Manipulation of Mass Tree

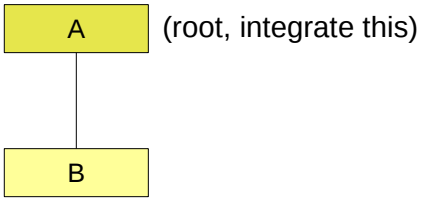
Mass Tree

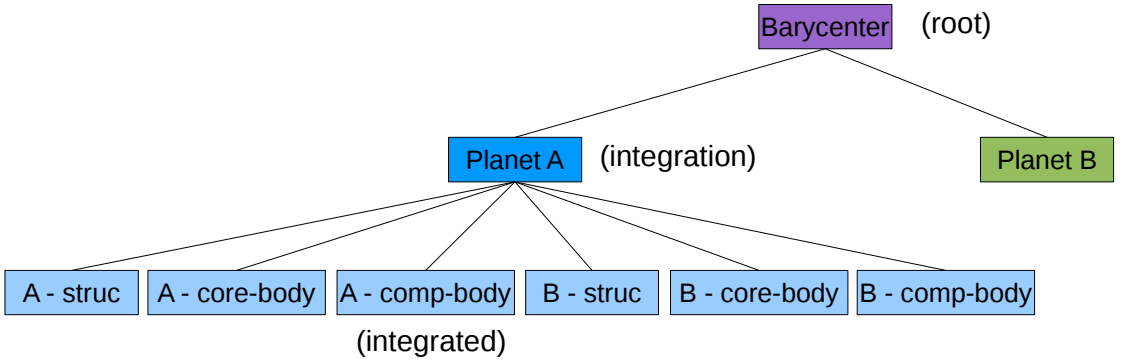


Reference Frame Tree



- Attach the bodies:
- Frames of attached bodies are siblings





Traversing the Reference Frame Tree

- frame.compute_position_from(source-frame, 3-array)
- frame.compute_relative_state(source-frame, RefFrameState)

- e.g.

```
vehicle2.dyn_body.find_vehicle_point("vehicle2_point").compute_relative_state(
                                                    vehicle.dyn_body.core_body,
                                                    states.body_point_frame_state)
```

- Note – *RefFrameState* output on *compute_relative_state* is the class used to describe the *state* of a *RefFrame*; it is not a *RefFrame*.