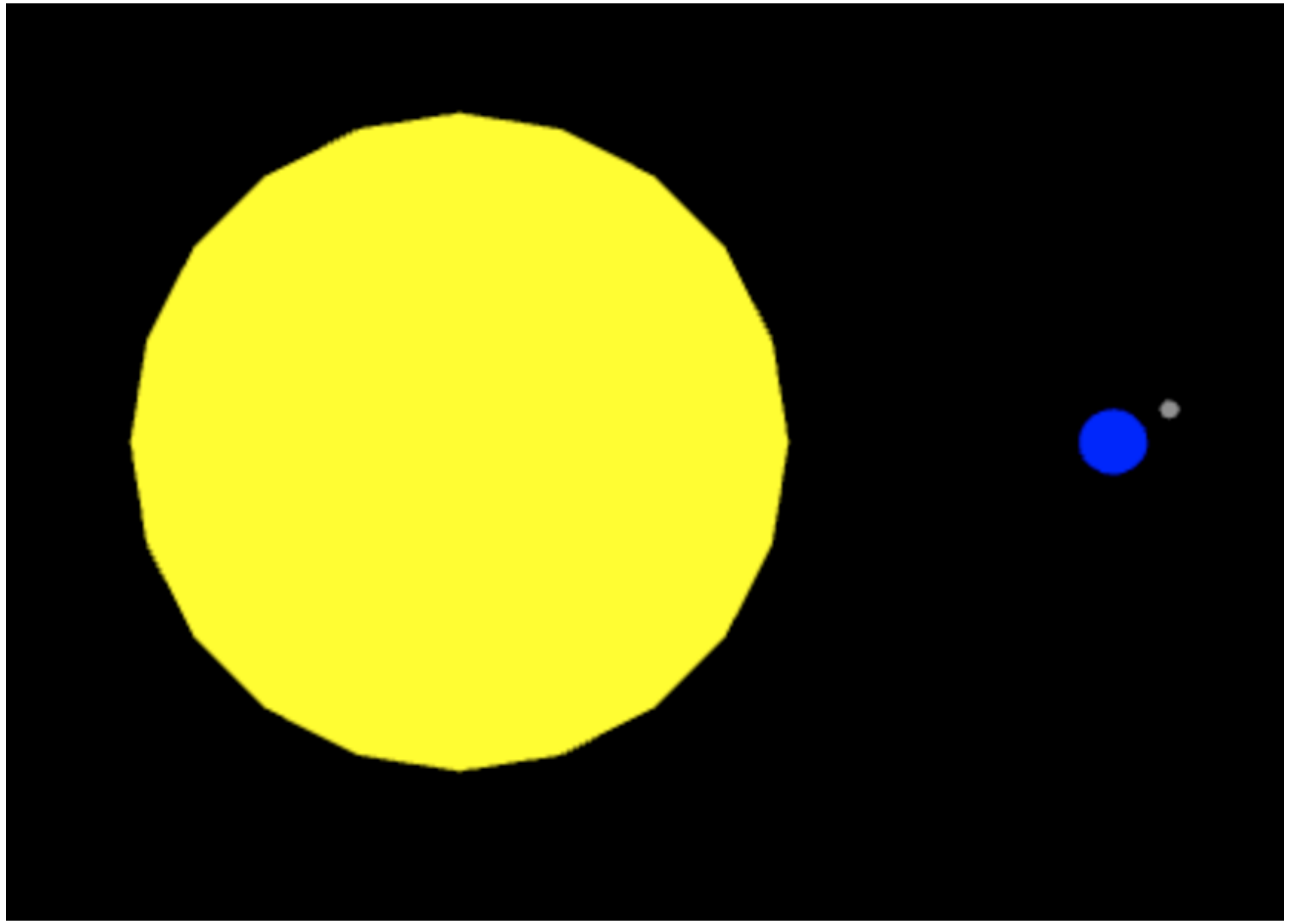
Solar System Example

CS 385 - Class 9
22 February 2022

Solar System using Matrix Stacks

Recall Our Simple Solar System

- We need to render three objects
  - Sun, Earth, and Moon
- The Sun is the center of the solar system, so everyone's dependent on its position
- The Moon's dependent on the position of the Earth
- Also, everyone's dependent on the viewer
  - which we specifying using the *viewing transformation*



MatrixStack.js

- Our JavaScript matrix stack implementation
  - stored in **MatrixStack.js**
- In your HTML file, include a reference to the JavaScript file
  - make sure to use the path correct
- And since **MatrixStack.js** relies on **MV.js**, don't forget to include it as well

```
<!DOCTYPE html>
<html>
<head>
  <script src="MV.js"></script>
  <script src="MatrixStack.js"></script>
  ...
</head>
<body>
  ...
</body>
</html>
```

Getting Started

- All our our matrix manipulation will occur in `render()`

❶ We'll start by creating a new MatrixStack

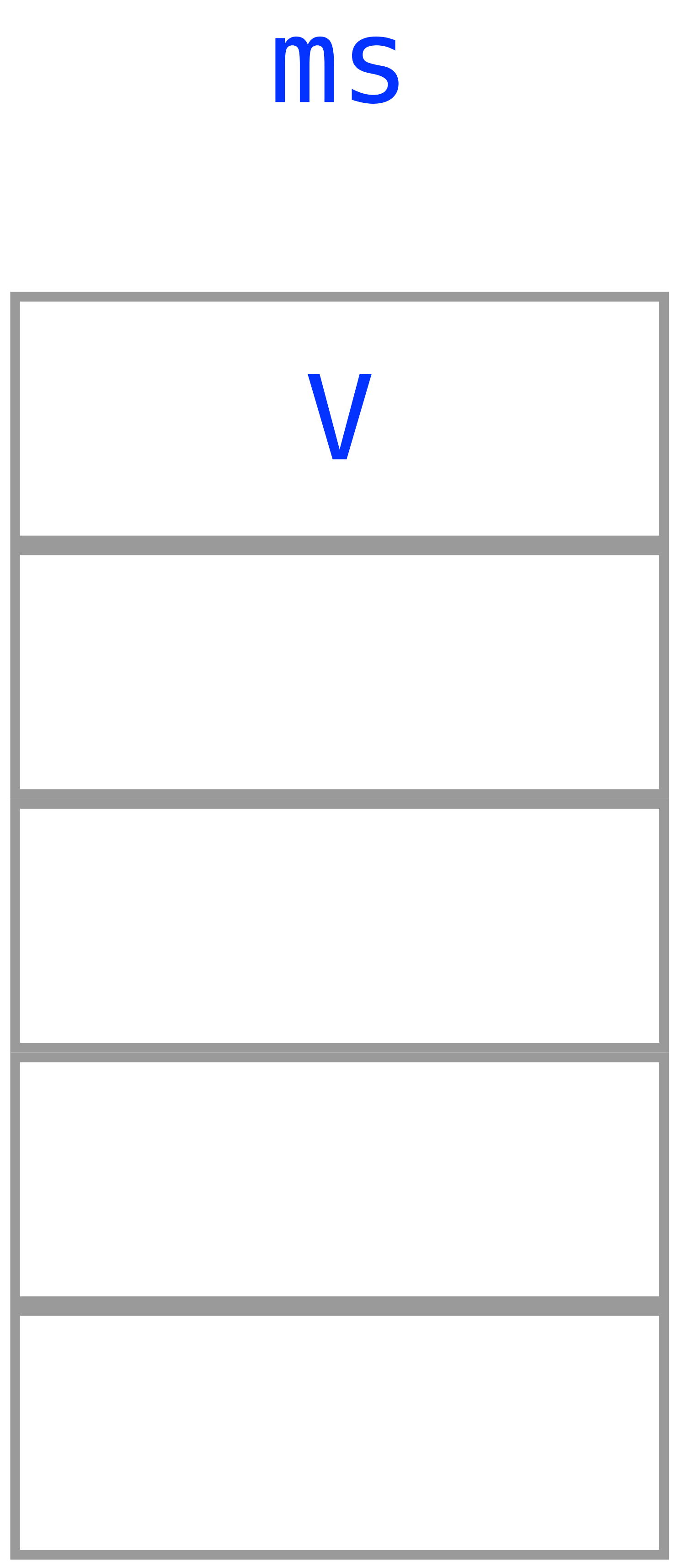
❷ Initializing it to the viewing transform

  - we'll use the *simplest viewing transform* (see Class 6's notes)

```
function render() {
  gl.clear( ... );

❶ ms = new MatrixStack();

❷ var V = translate(0.0, 0.0, -0.5*(near +far))
  ms.load(V);

  ...
}
```

Matrix Stack State

- After that operation, our matrix stack has one matrix V as the current matrix



ms

V

Rendering the Sun

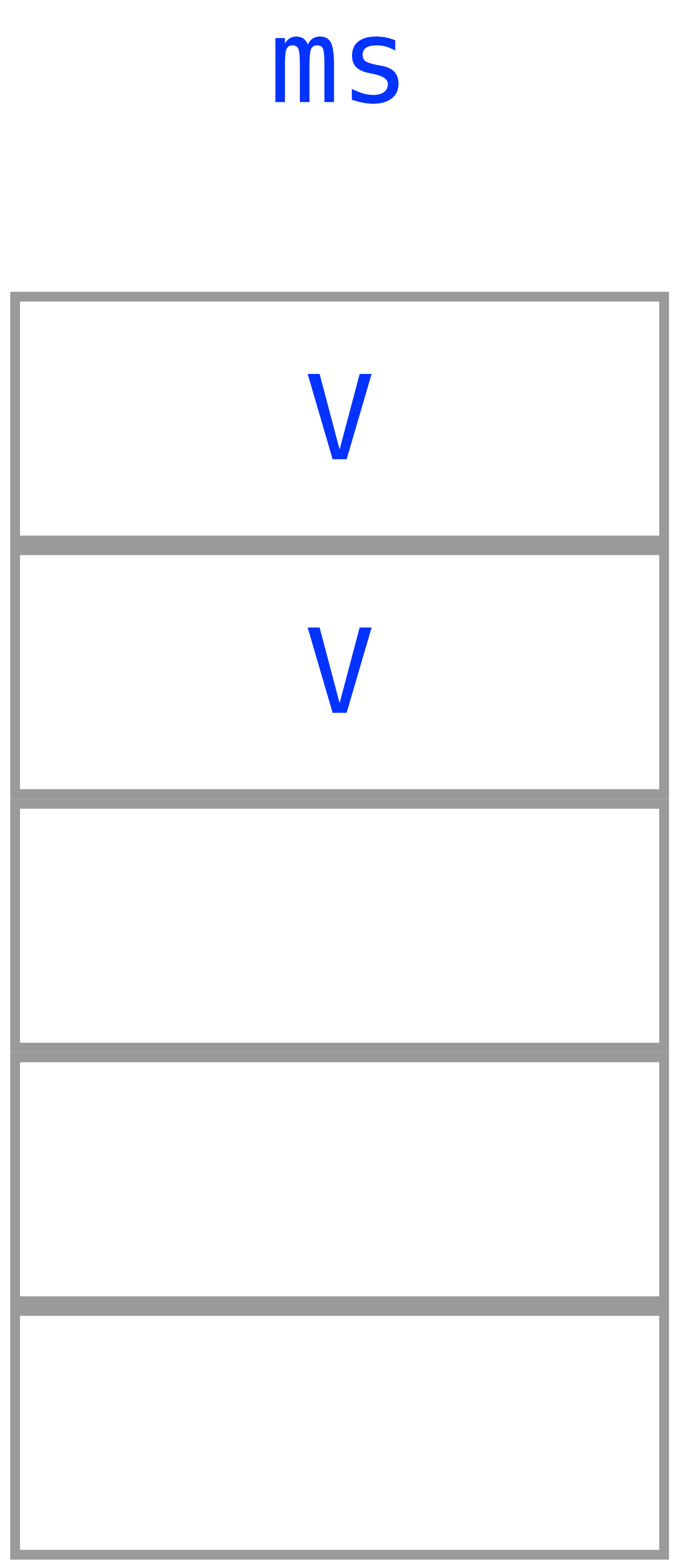
- Since the Sun's size doesn't affect the rendering of any other planets, we isolate it within a push/pop pair

```
function render() {
  gl.clear( ... );

  ms = new MatrixStack();

  var V = translate(0.0, 0.0, -0.5*(near +far))
  ms.load(V);

   ms.push();
   ms.scale(Sun.radius);
   ... // set up other parameters required to draw Sun
   Sun.draw();
   ms.pop();
}
```

Matrix Stack State

- After the Sun's push operation, the matrix stack looks like

```
ms.push()
```



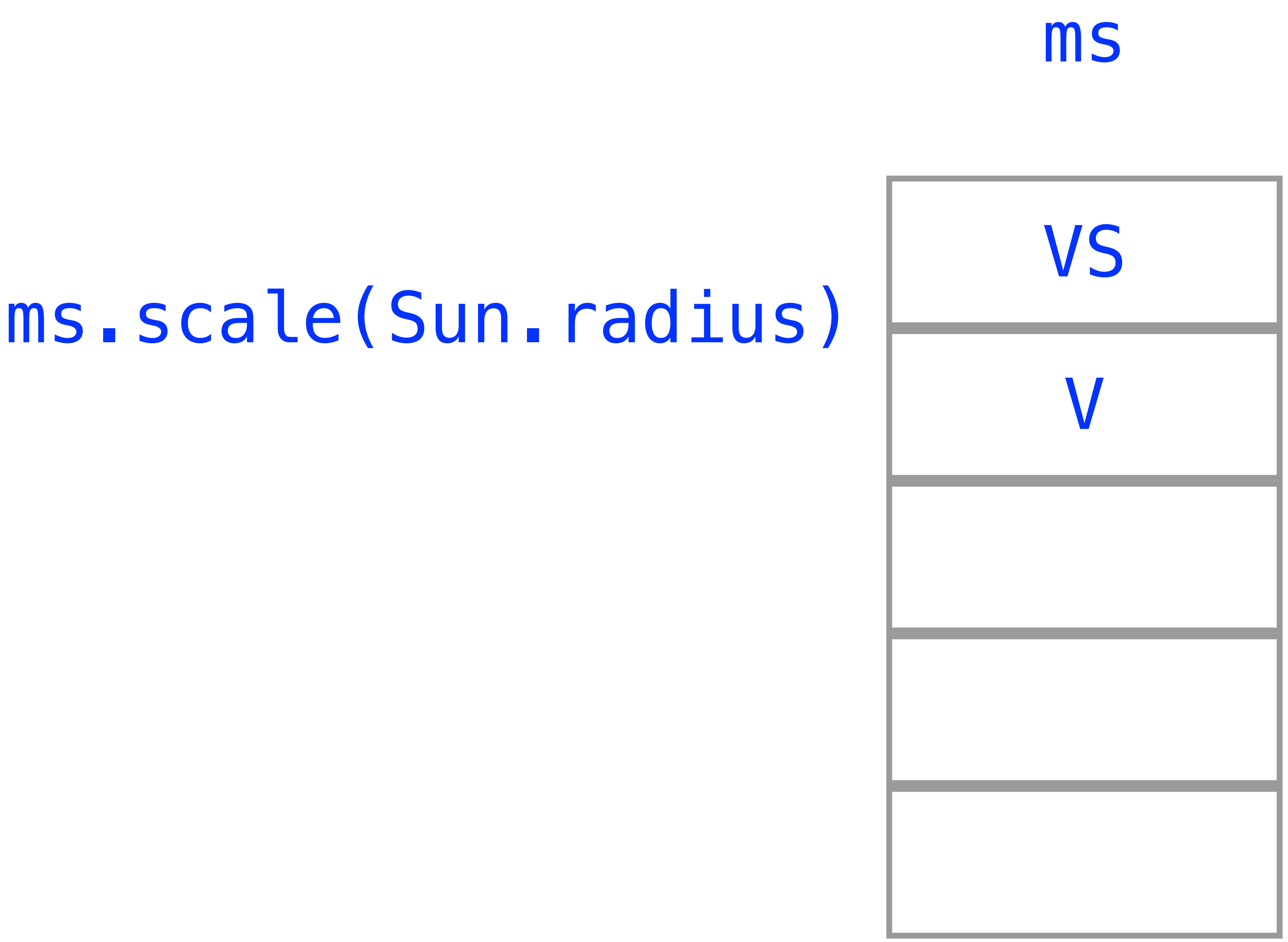
ms

V

V

Matrix Stack State

- Scaling the Sun results in the following state



ms

ms.scale(Sun.radius)

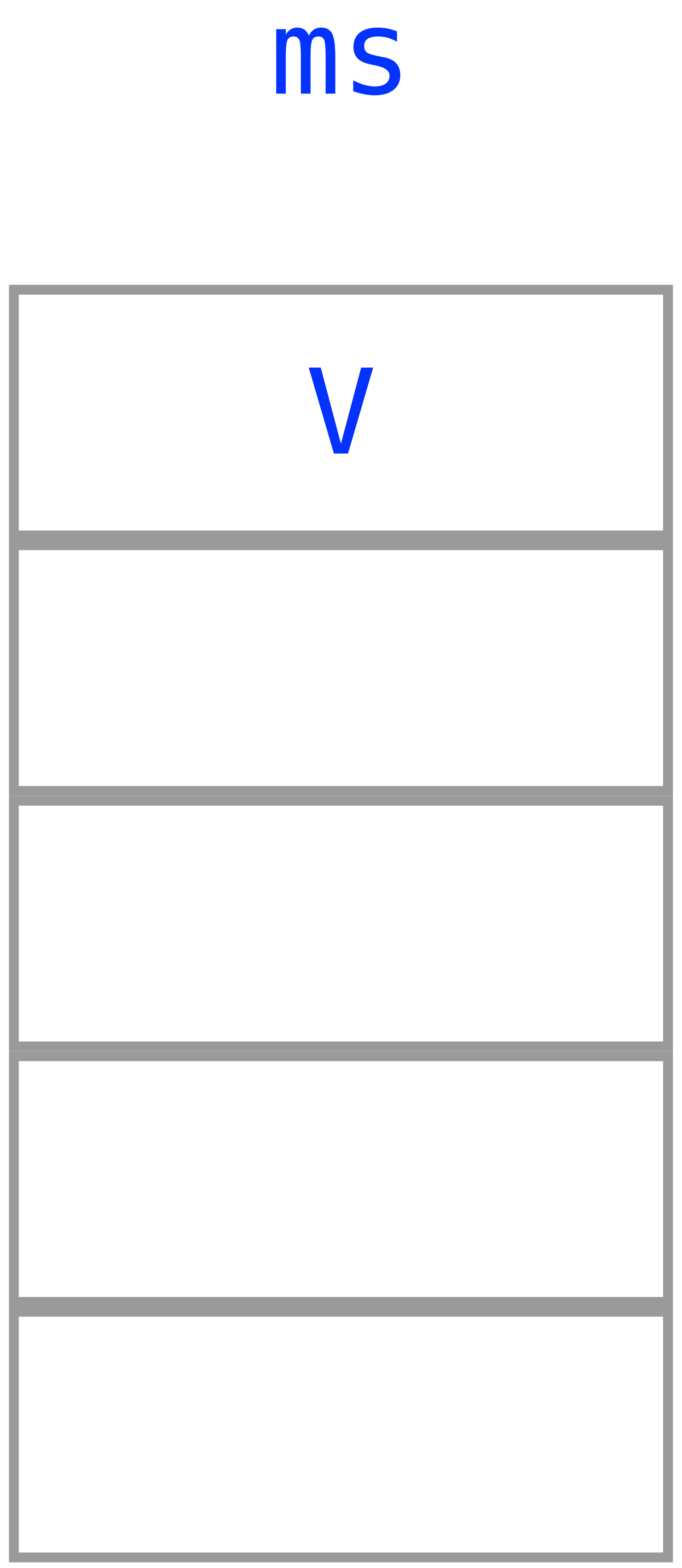
VS

V

Matrix Stack State

- After rendering the Sun, pop returns the stack to its pre-Sun state

```
ms.pop()
```



Rendering the Earth

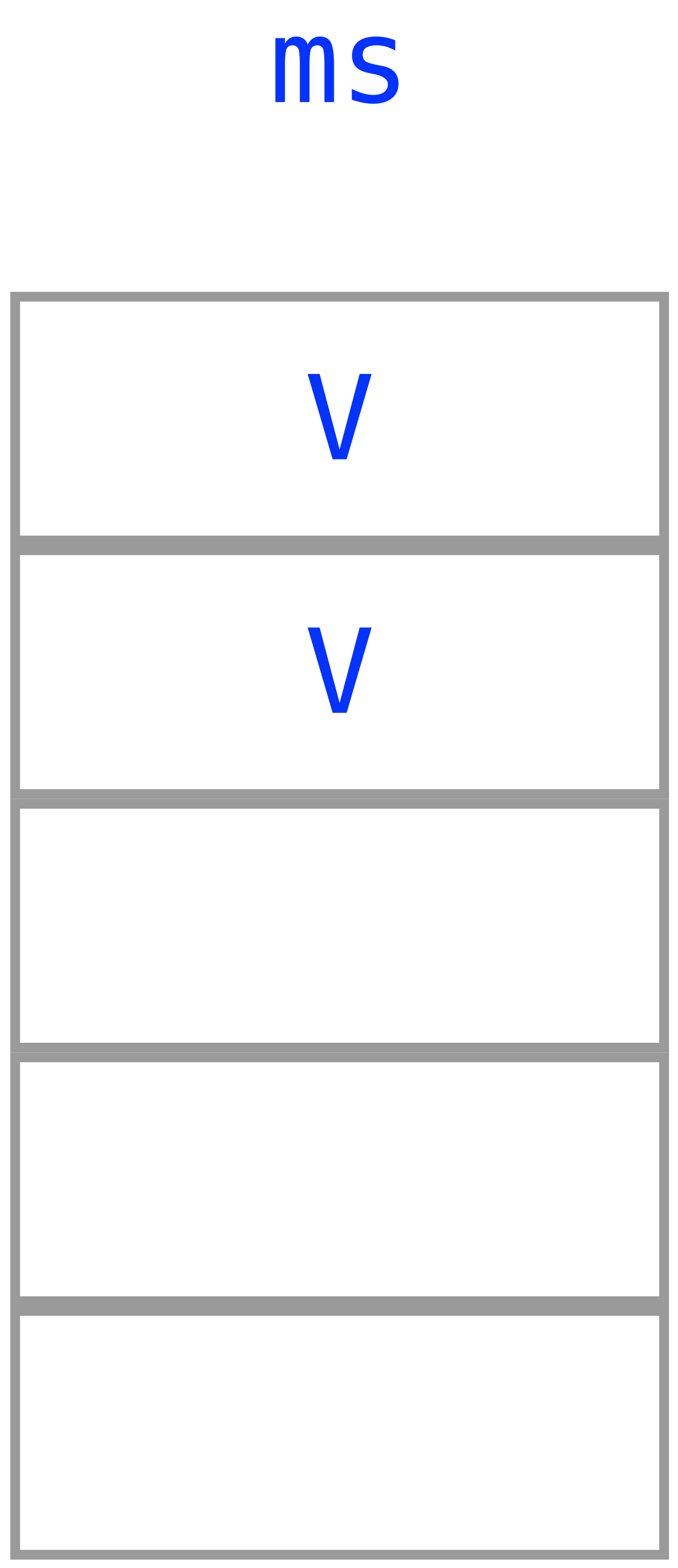
- The Earth requires several transformations:
  - positioning it appropriately with respect to the sun
    - this includes accounting for the distance from the sun
    - and its rotation around the sun (i.e., the day of the year)
  - scaling it to the appropriate size
  - incorporating a rotation to represent its day

```
function render() {
  gl.clear( ... );

  ms = new MatrixStack();

  var V = translate(0.0, 0.0, -0.5*(near +far))
  ms.load(V);

  ms.push();
  ms.scale(Sun.radius);
  ... // set up other parameters required to draw Sun
  Sun.draw();
  ms.pop();

  ms.push();
  ms.rotate(year, axis);
  ms.translate(distance, 0, 0);
  ms.rotate(day, axis);
  ms.scale(Earth.radius);
  Earth.draw();
  ms.pop();
}
```

Matrix Stack State

- Again, after the Earth's push operation, the matrix stack looks like

`ms.push()`



ms

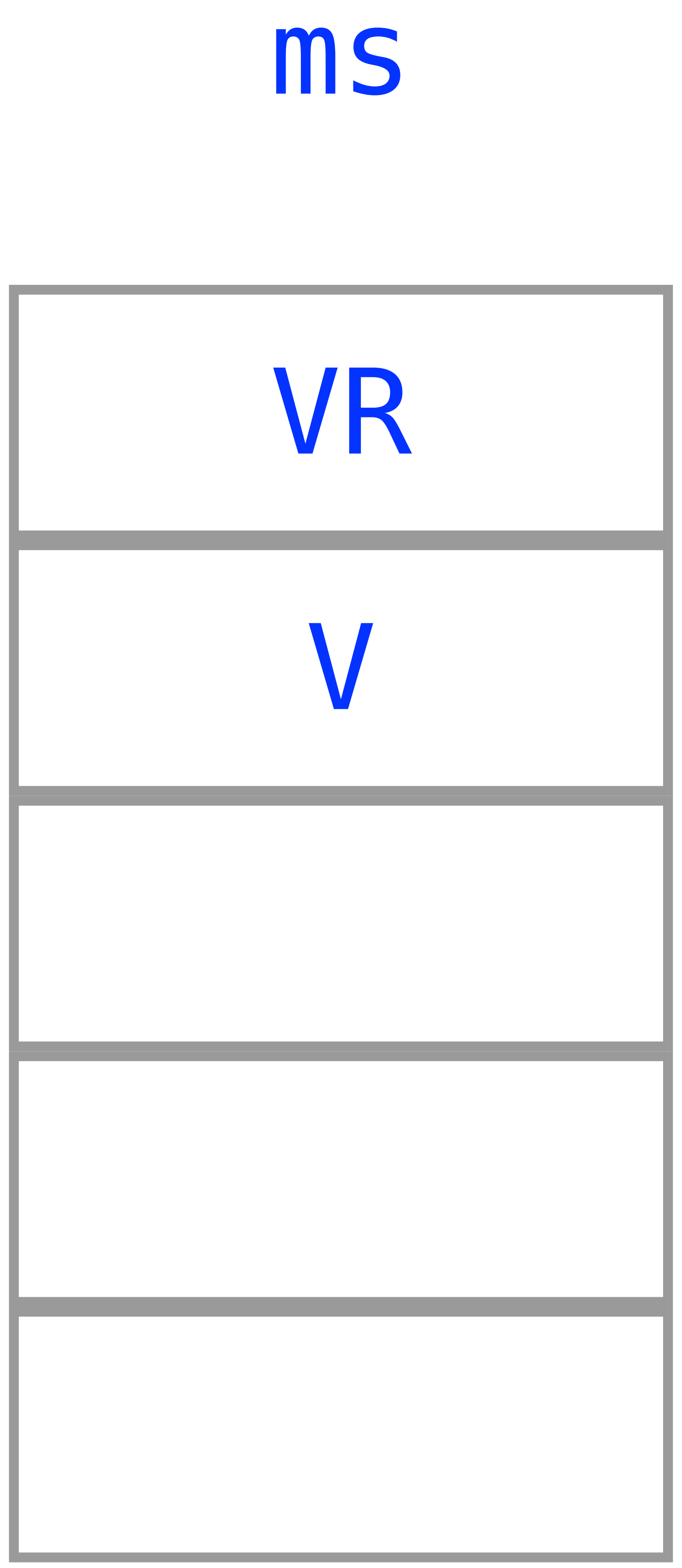
V

V

Matrix Stack State

- Rotating the coordinate system to take into account the Earth's day-of-the-year position yields

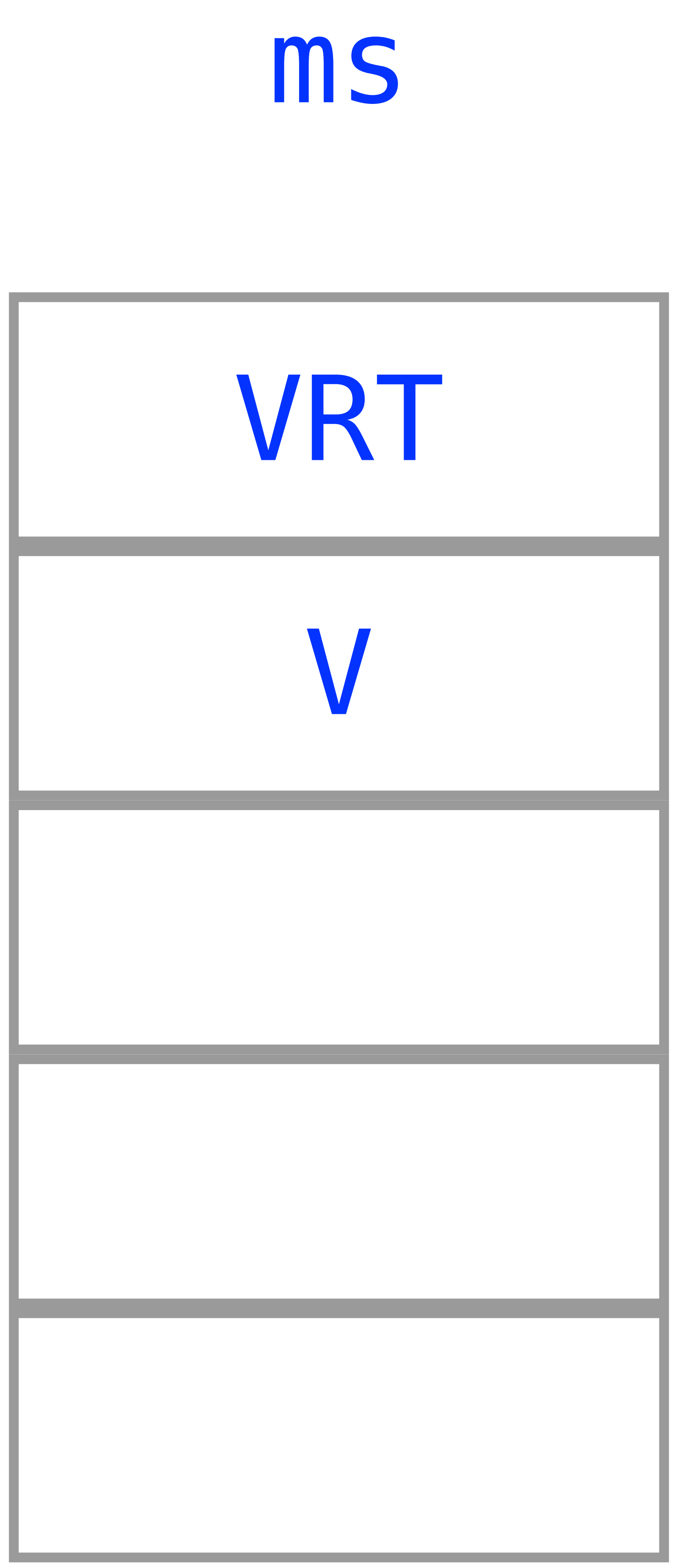
```
ms.rotate(year, axis)
```



Matrix Stack State

- Next, we move the Earth's coordinate system to its appropriate distance from the Sun

```
ms.translate(distance, 0, 0)
```



ms

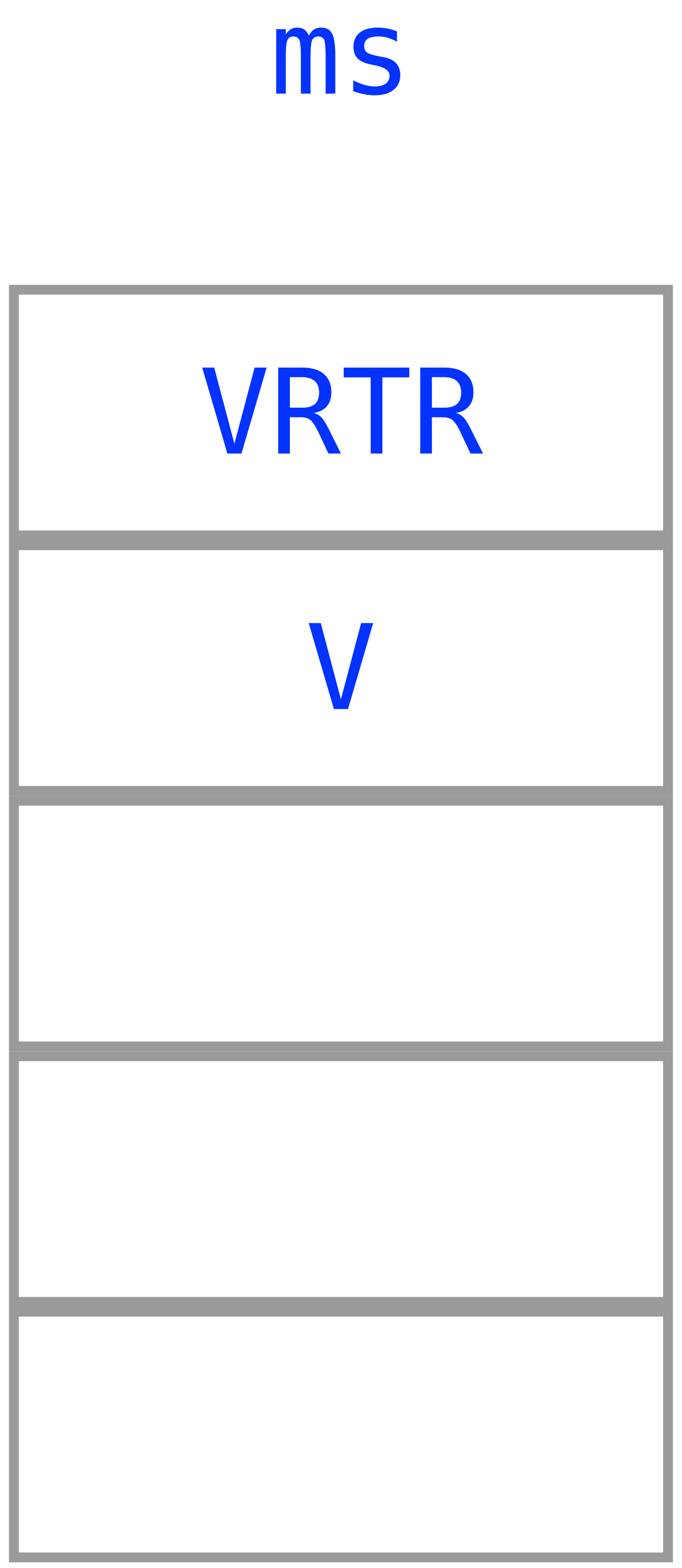
VRT

V

Matrix Stack State

- Next, we rotate the Earth's coordinate system to take into account the Earth's day rotation

```
ms.rotate(day, axis)
```



ms

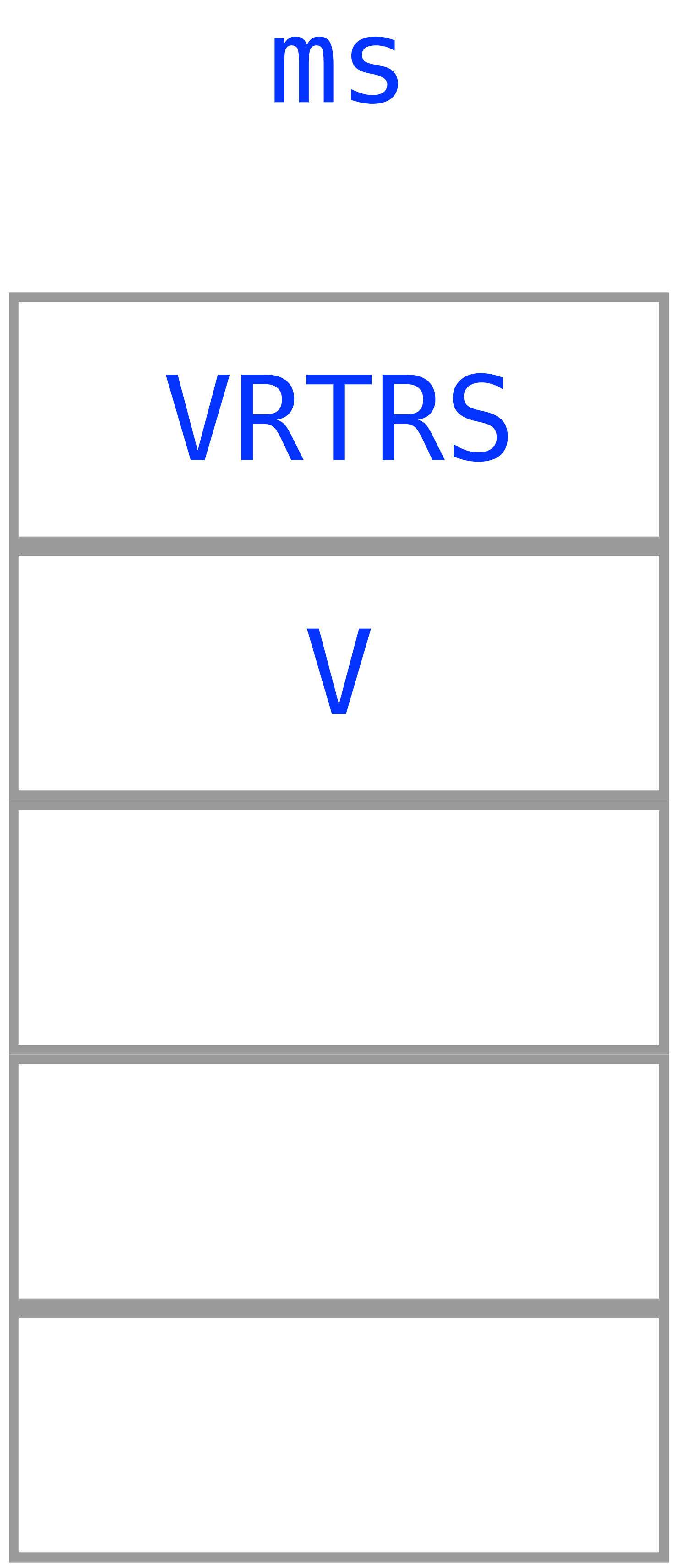
VRTR

V

Matrix Stack State

- Just as with the Sun, we then scale the coordinate system to the appropriate size to match the Earth's radius

```
ms.scale(Earth.radius)
```



ms

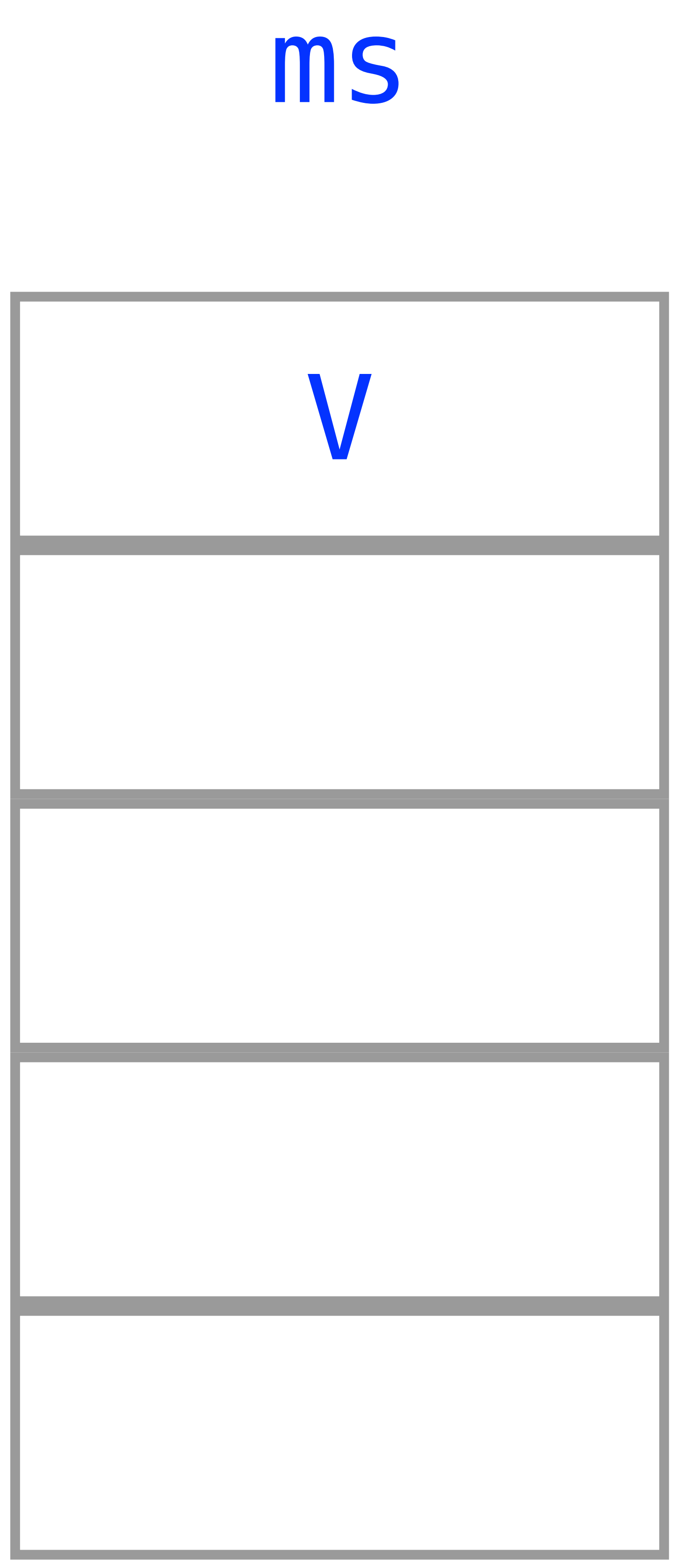
VRTRS

V

Matrix Stack State

- After rendering the Earth, pop returns the stack to its pre-Sun state

```
ms.pop()
```



ms

V

Rendering the Moon

- Like the Earth, the Moon requires several transformations:
    - positioning it appropriately with respect to the Earth
        - this includes accounting for the distance from the Earth
        - and its rotation around the Earth (i.e., the day of the month)
    - scaling it to the appropriate size
    - incorporating a rotation to represent its day
- However, some of the Earth's transformations affect the Moon, and others don't

```
function render() {
  gl.clear( ... );

  ms = new MatrixStack();

  var V = translate(0.0, 0.0, -0.5*(near +far))
  ms.load(V);

  ms.push();
  ms.scale(Sun.radius);
  ... // set up other parameters required to draw Sun
  Sun.draw();
  ms.pop();

   ms.push();
   ms.rotate(year, axis);
   ms.translate(Earth.distance, 0, 0);
   ms.rotate(day, axis);
   ms.scale(Earth.radius);
   Earth.draw();
   ms.pop();
}
```

Dependent Tranformations

- Which of the Earth's transformations affect the Moon?
  - the year rotation — since that controls the Earth's position relative to the Sun
  - the Earth's distance from the Sun — which also controls the Moon's position
- However, the Earth's daily rotation (day), nor its size affect the Moon
- So, we need to introduce some additional matrix stack pushes and pops to isolate transformations

```
function render() {
  gl.clear( ... );

  ms = new MatrixStack();

  var V = translate(0.0, 0.0, -0.5*(near +far))
  ms.load(V);

  ms.push();
  ms.scale(Sun.radius);
  ... // set up other parameters required to draw Sun
  Sun.draw();
  ms.pop();

  ms.push();
  ms.rotate(year, axis);
  ms.translate(Earth.distance, 0, 0);
  ms.rotate(day, axis);
  ms.scale(Earth.radius);
  Earth.draw();
  ms.pop();
}
```

Dependent Tranformations

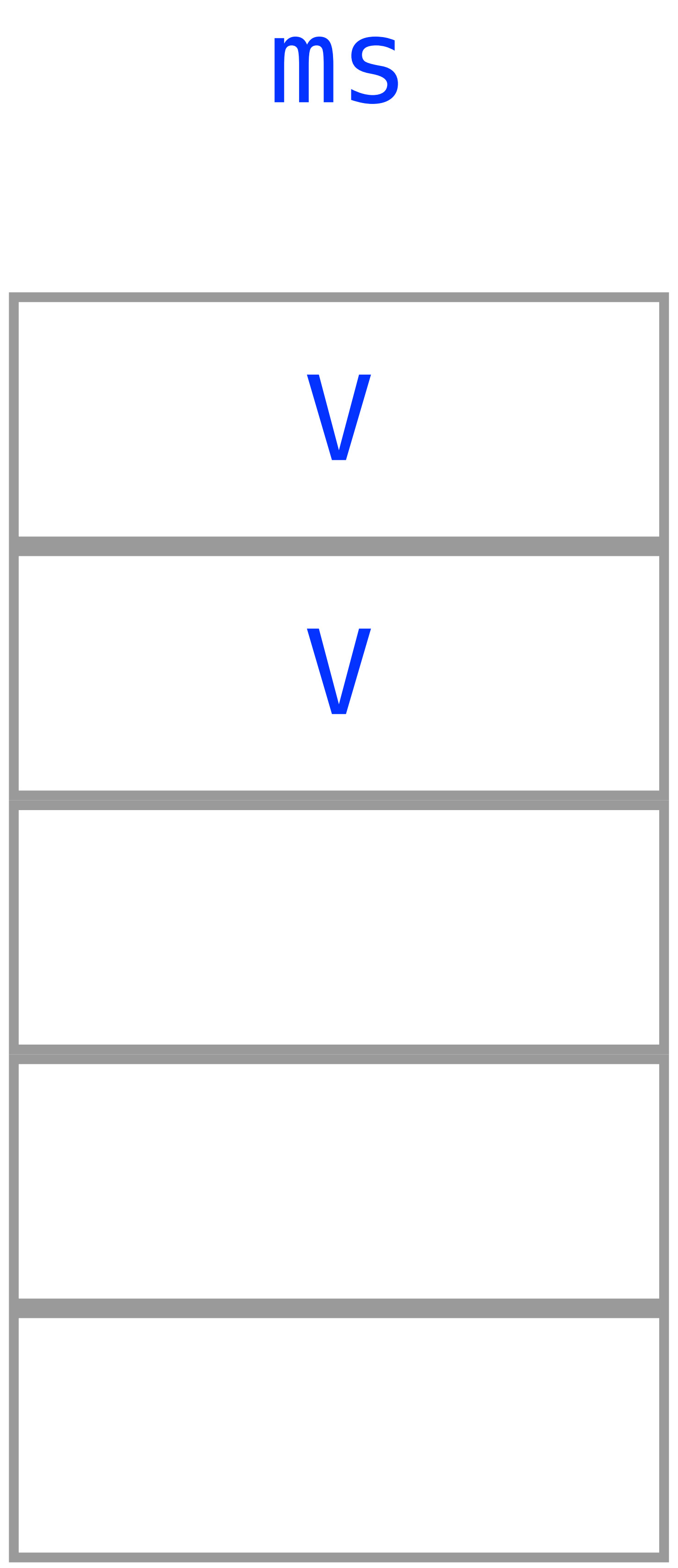
- This modifies how our matrix stack looks, as shown on the following slides

```
function render() {
  gl.clear( ... );

  ms = new MatrixStack();

  var V = translate(0.0, 0.0, -0.5*(near +far))
  ms.load(V);

  ms.push();
  ms.scale(Sun.radius);
  ... // set up other parameters required to draw Sun
  Sun.draw();
  ms.pop();

  ms.push();
  ms.rotate(year, axis);
  ms.translate(Earth.distance, 0, 0);
  ms.push();
  ms.rotate(day, axis);
  ms.scale(Earth.radius);
  Earth.draw();
  ms.pop();
  ms.pop();
}
```

Matrix Stack State

- Starting over, after the Earth's push operation, the matrix stack looks like

```
ms.push()
```



ms

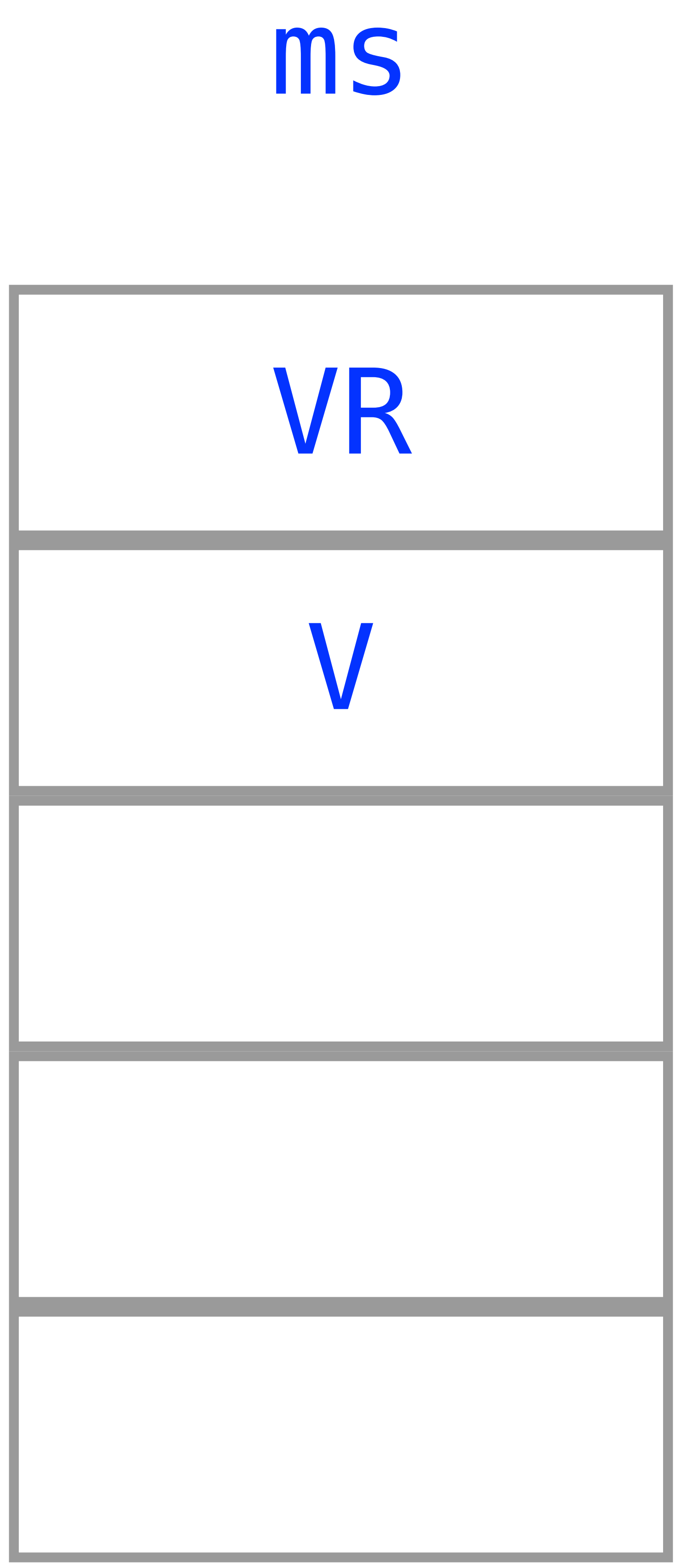
V

V

Matrix Stack State

- Rotating the coordinate system to take into account the Earth's day-of-the-year position yields

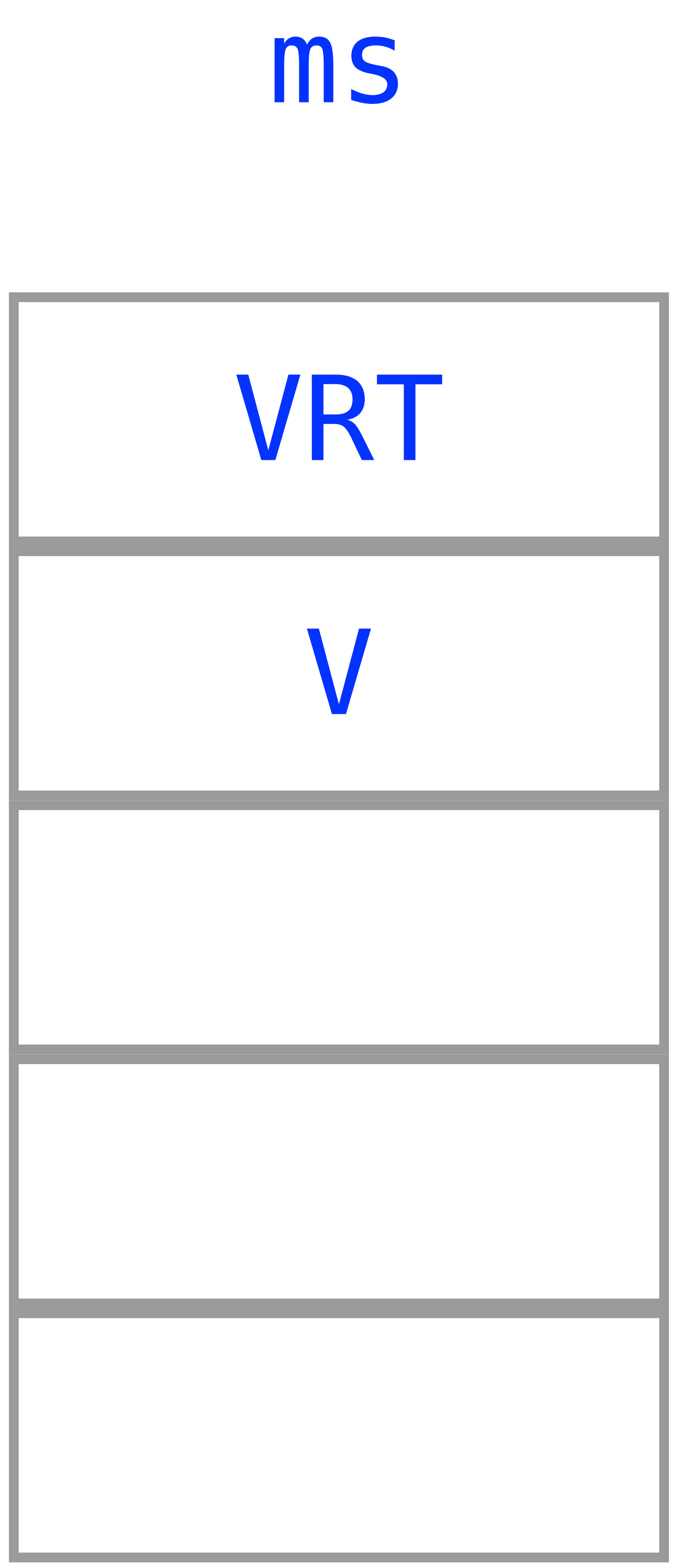
```
ms.rotate(year, axis)
```



Matrix Stack State

- Next, we move the Earth's coordinate system to its appropriate distance from the Sun

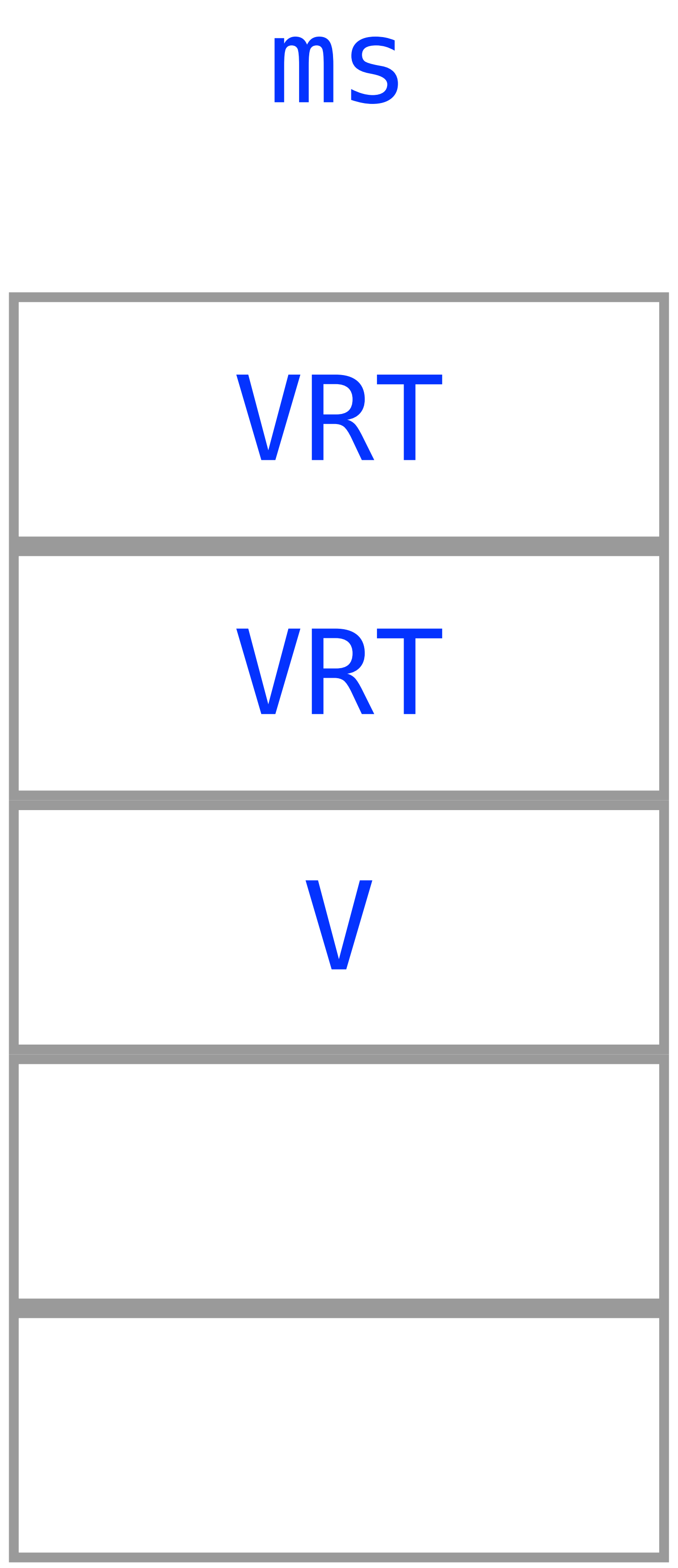
```
ms.translate(distance, 0, 0)
```



Matrix Stack State

- To isolate some of the Earth's transformations from other transforms, we add in the additional matrix stack push

```
ms.push()
```



ms

VRT

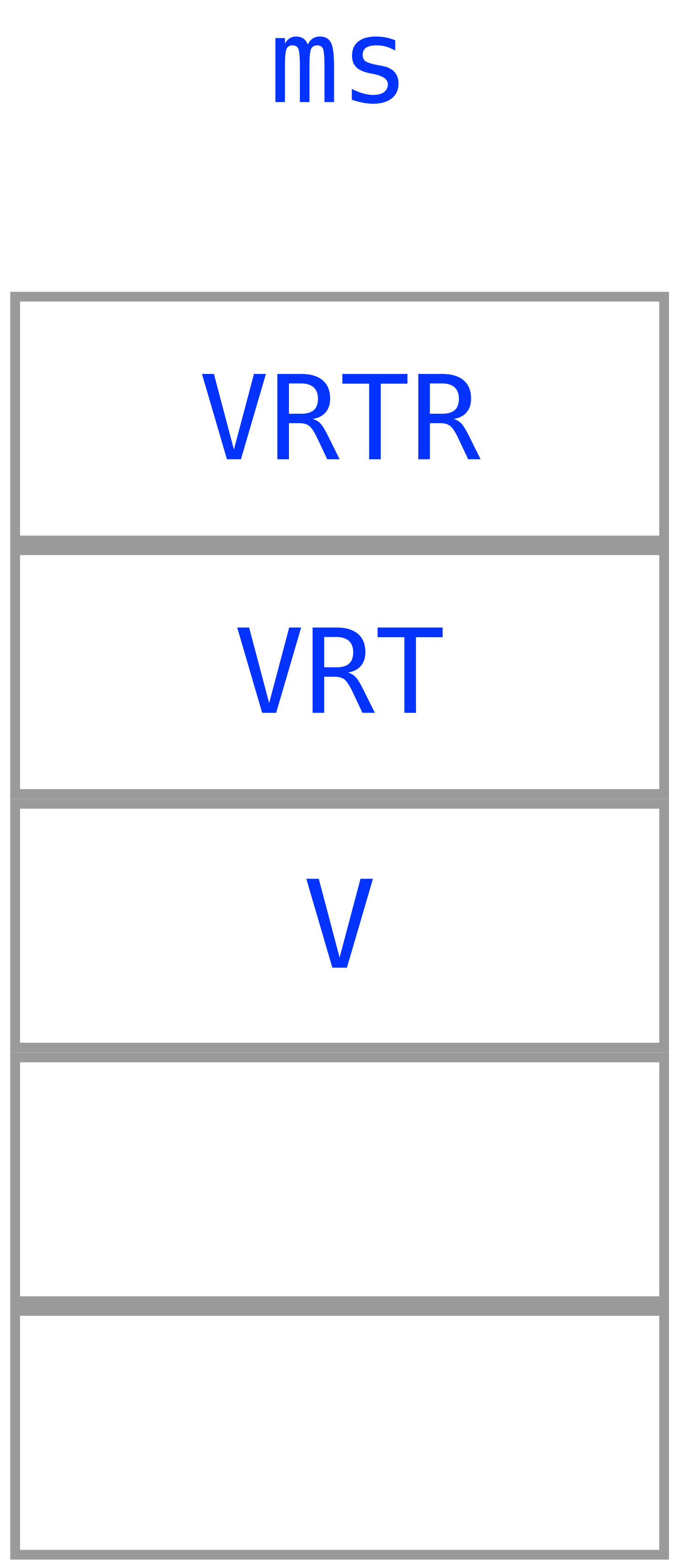
VRT

V

Matrix Stack State

- Continuing with our new matrix stack, we rotate the Earth's coordinate system to take into account the Earth's day rotation

```
ms.rotate(day, axis)
```



ms

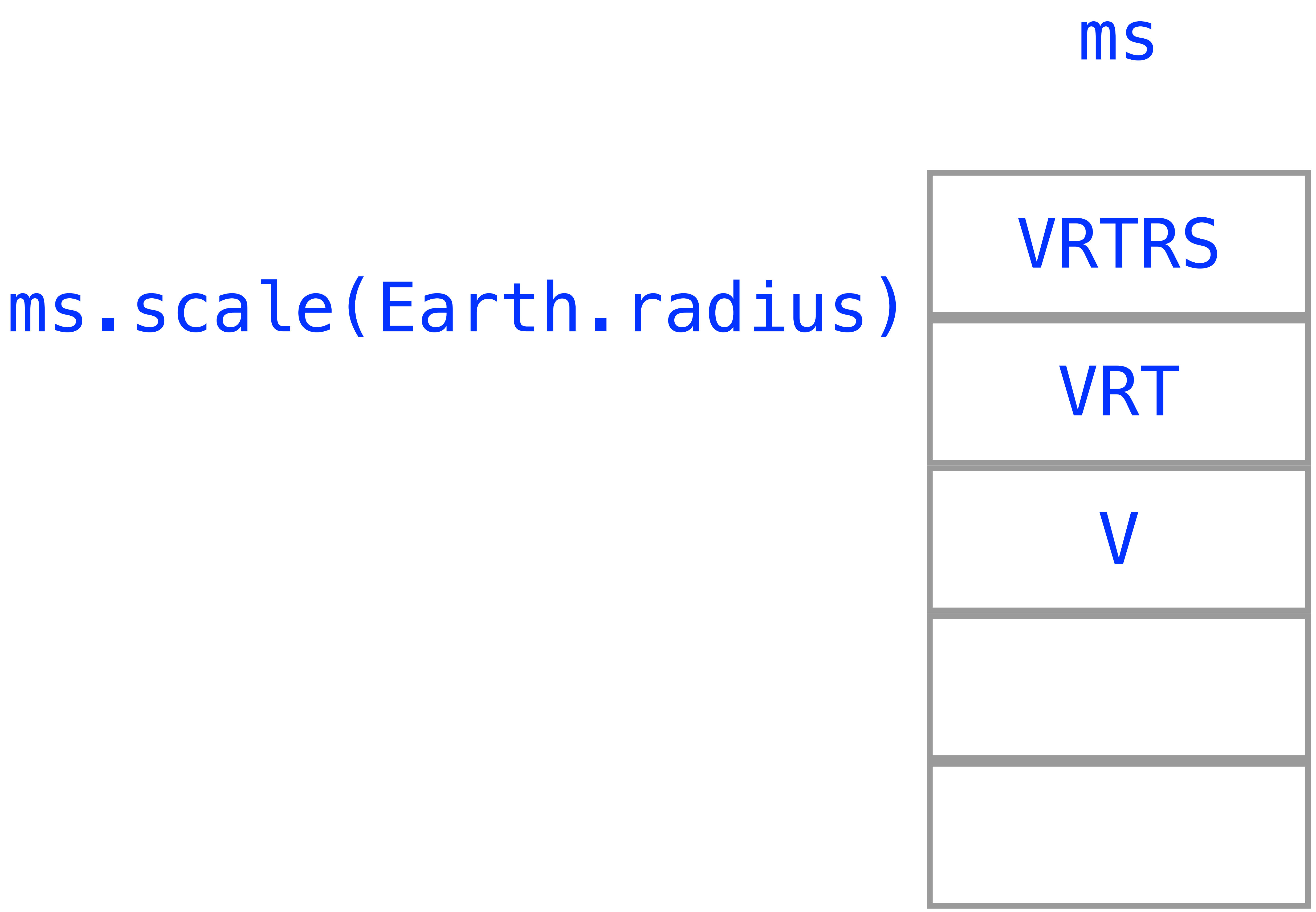
VRTR

VRT

V

Matrix Stack State

- Just as with the Sun, we then scale the coordinate system to the appropriate size to match the Earth's radius



ms

`ms.scale(Earth.radius)`

VRTRS

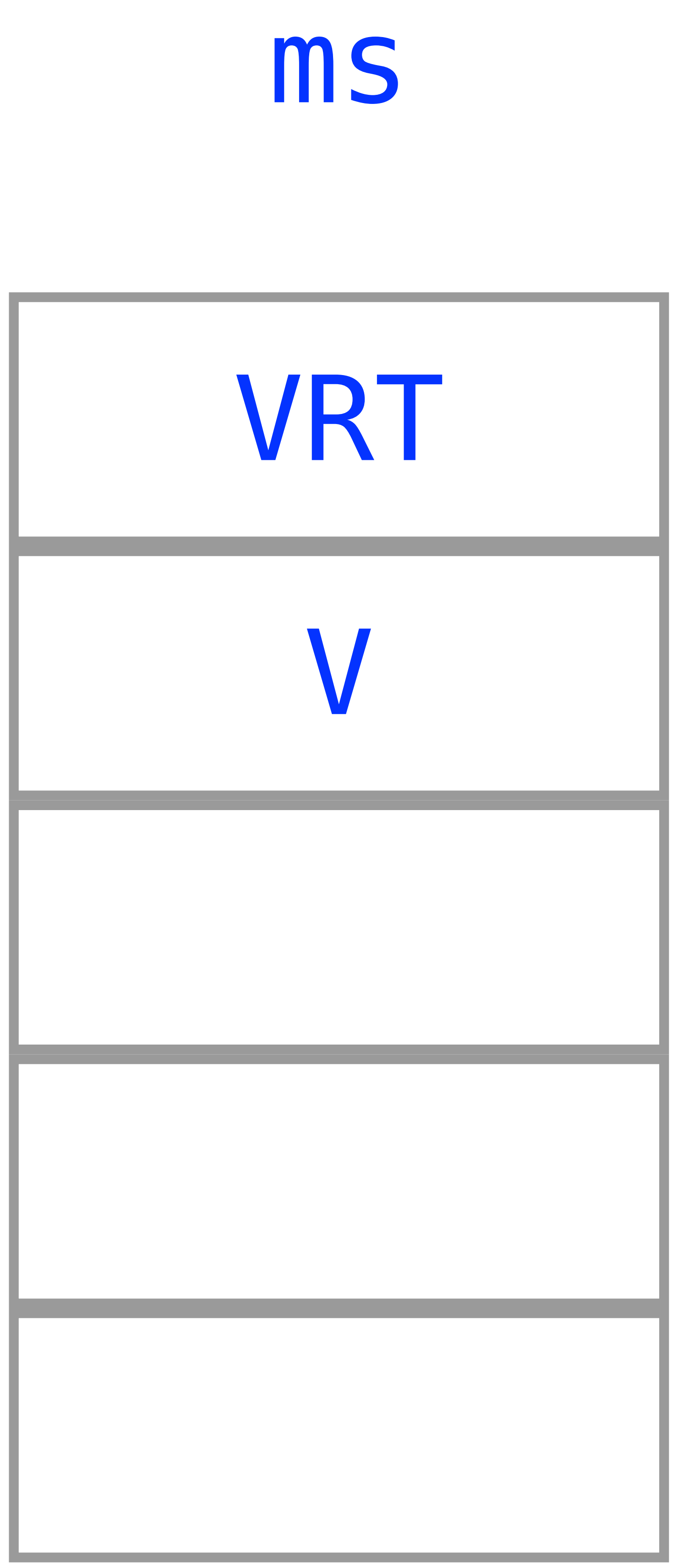
VRT

V

Matrix Stack State

- After rendering the Earth, pop returns the stack to its pre-Earth state, which we can augment for the Moon's transformations

```
ms.pop()
```



ms

VRT

V

Rendering the Moon

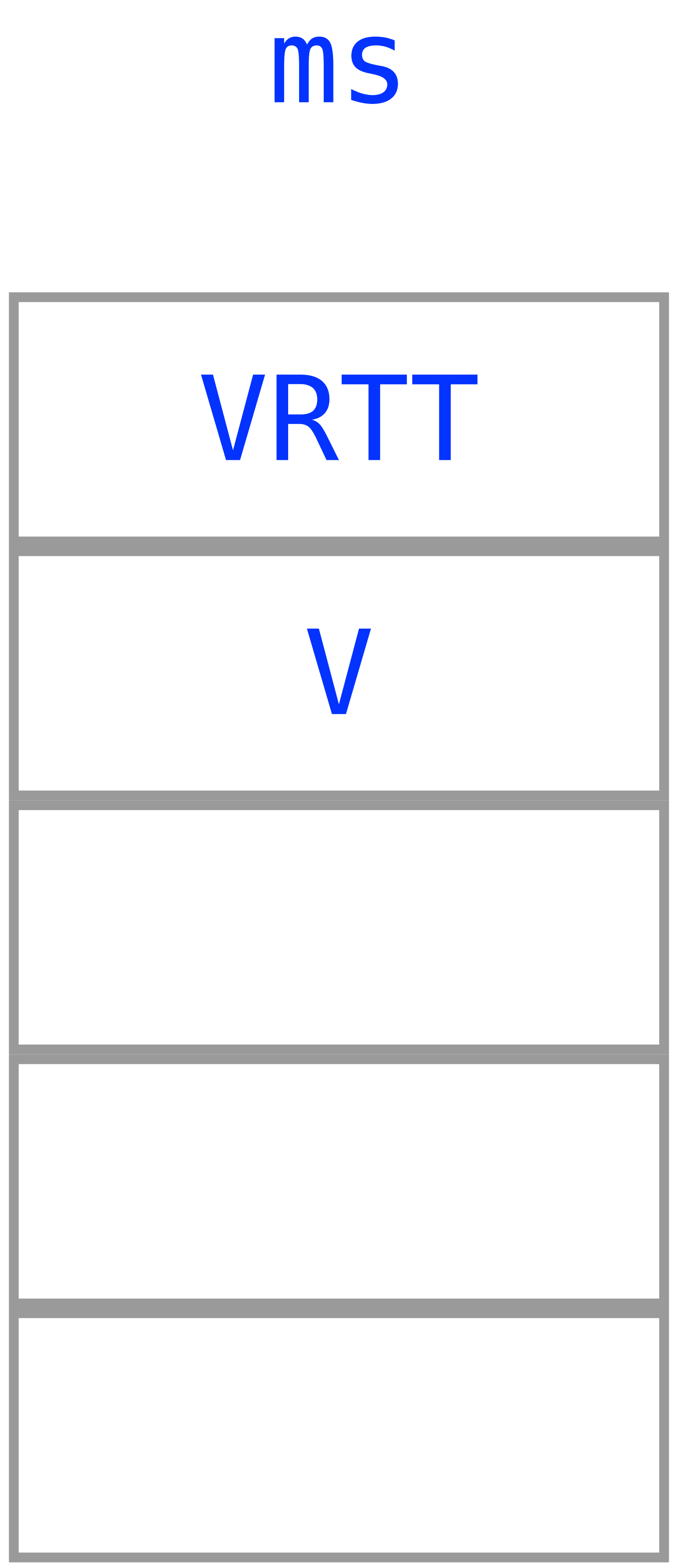
- To associate the Earth's transformations that apply to the Moon, we merely render while they're still on the matrix stack

```
function render() {
  gl.clear( ... );

  ms = new MatrixStack();

  var V = translate(0.0, 0.0, -0.5*(near +far))
  ms.load(V);

  ms.push();
  ms.scale(Sun.radius);
  ... // set up other parameters required to draw Sun
  Sun.draw();
  ms.pop();

  ms.push();
  ms.rotate(year, axis);
  ms.translate(Earth.distance, 0, 0);
  ms.push();
  ms.rotate(day, axis);
  ms.scale(Earth.radius);
  Earth.draw();
  ms.pop();
  ms.translate(Moon.distance, 0, 0);
  ms.scale(Moon.radius);
  Moon.draw();
  ms.pop();
}
```

Matrix Stack State

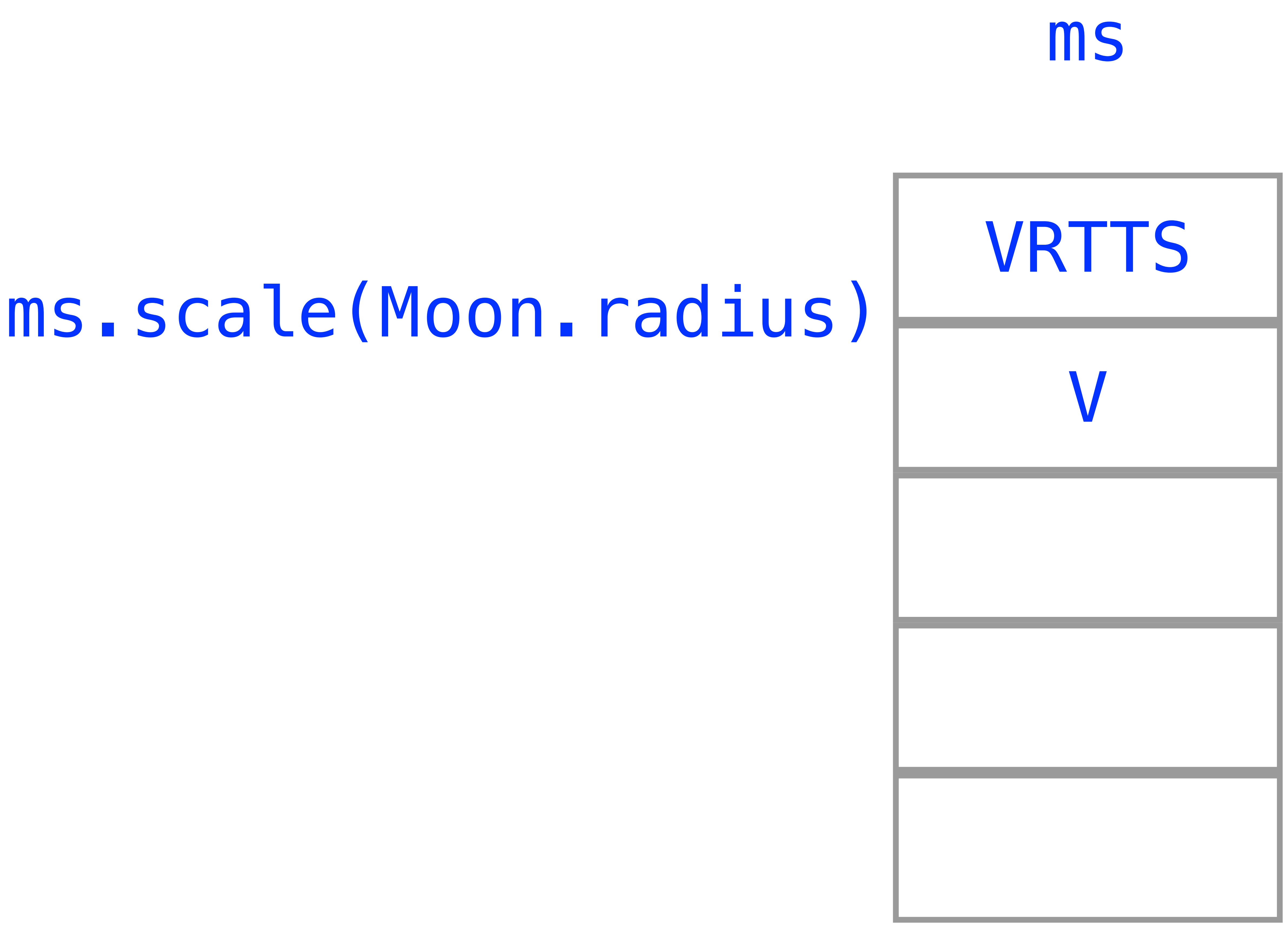
- We can now add our Moon's transformations

```
ms.translate(Moon.distance, 0, 0)
```



Matrix Stack State

- Including its scale



ms

ms.scale(Moon.radius)

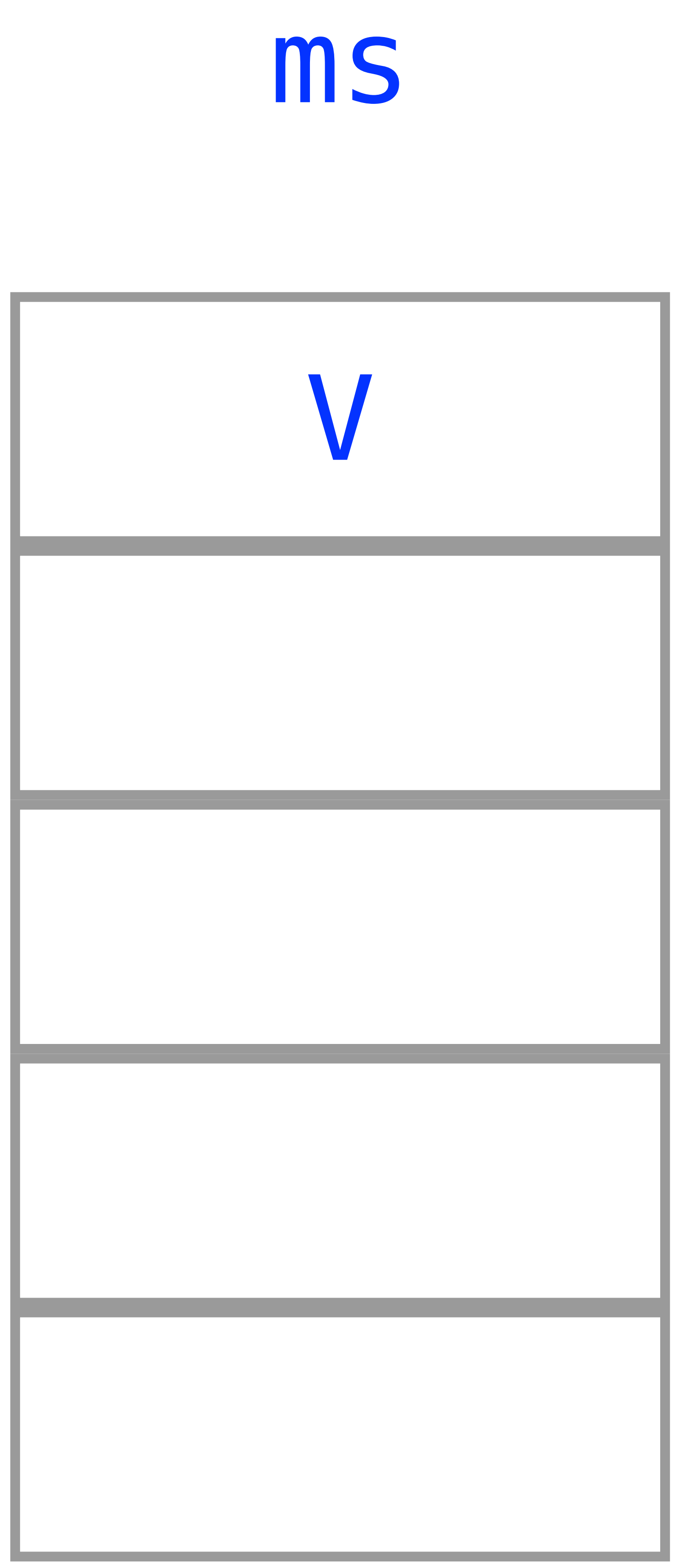
VRTTS

V

Matrix Stack State

- And finally, we pop to restore the stack

```
ms.pop()
```

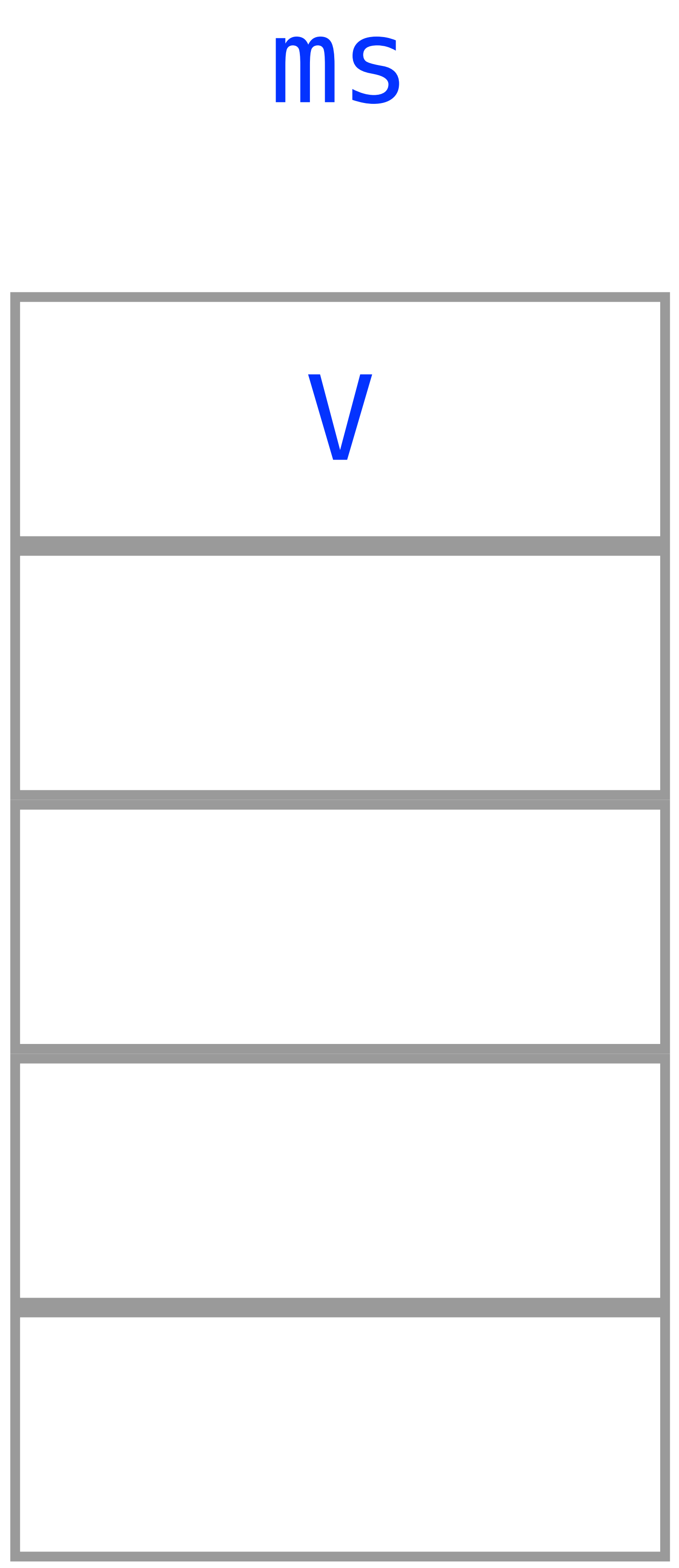


ms

V

Something to think about

- Why didn't we do a push and pop around the Moon like we did for the Earth?



ms

V