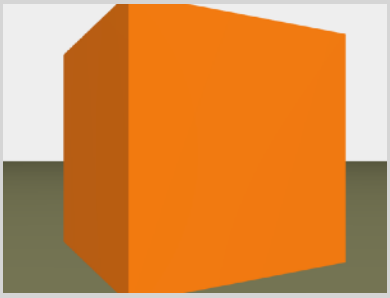
THREE.js 카메라 이해하기

fov : 카메라 각도
망원 - 28도이하 (각도가 작고 시야가 좁다)
일반 - 43도 정도
광원 - 63도 (각도가 크고 시야가 넓다)



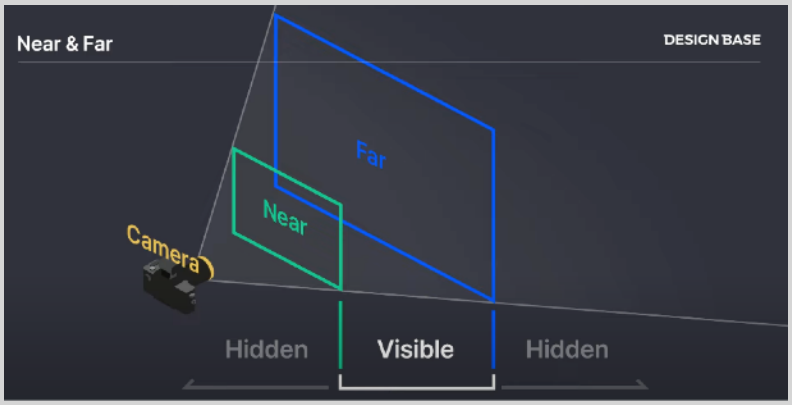
43도

28도

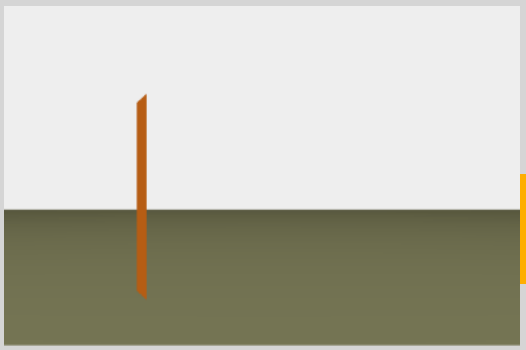
80도

aspect : 종횡비 (가로 세로 비율)
- 윈도우 viewport로 기준을 잡는다.

near / far
near : 카메라시점이 시작하는 구간
far : 카메라시점이 끝나는 구간



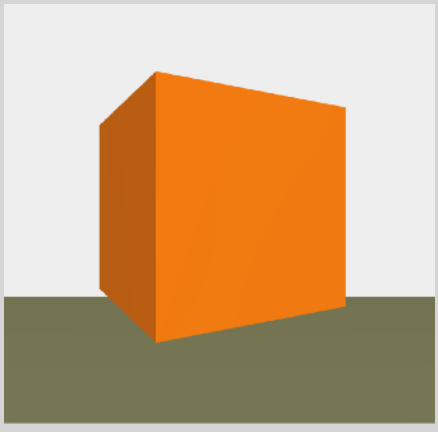
near : 0.5



near : 1 (카메라가 피사체와 가까워지면서 앞 부분은 잘린 모습을 볼 수 있다.)

** near의 숫자가 커질 수록 앞 쪽이 잘린다.
피사체와 더 가까워 진다고 생각할 수 있다.

far:1000



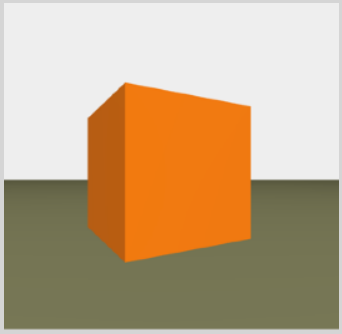
far:2 (카메라의 끝 영역이 좁아지면서 뒷면의 땅이 점점 안보이는 것을 볼 수 있다.)

** far의 숫자가 작아질 수록 뒤가 잘린다.
피사체와 더 가까워 진다고 생각할 수 있다.

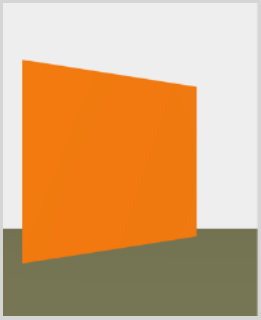
카메라 시점

```
camera.position.x = 0
camera.position.y = 0
camera.position.z = 1
camera.position.set(0, 0, 1);
```

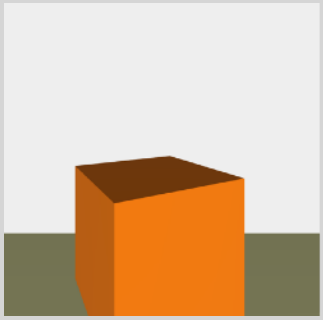
위의 세 줄을 마지막 줄의 set으로 합쳐서 코드를 작성할 수 있다.



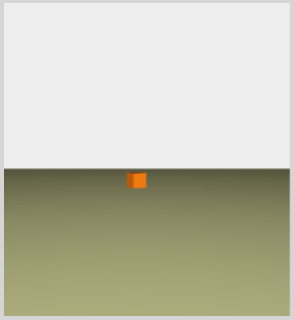
(0,0,1)



(0.4, 0 ,1)

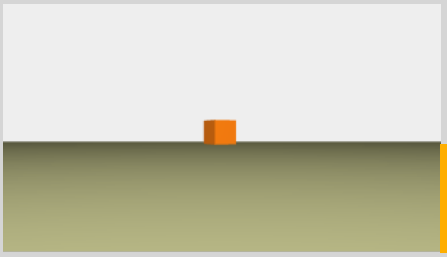


(0, 1 ,1)



(0, 1 ,10)

**카메라의 위치 값을 잘못 설정하면 에러가 난것 처럼 아무것도 안보일 수 있으니 주의하자.
'0의 값은 중앙이라는 뜻을 가진다'라고 생각하면 좋다.



(0,0,10) y축을 0으로 했을때 높낮이 없이 멀리 화면이 나오는 것을 확인 할 수 있다.

카메라가 바라보는 곳

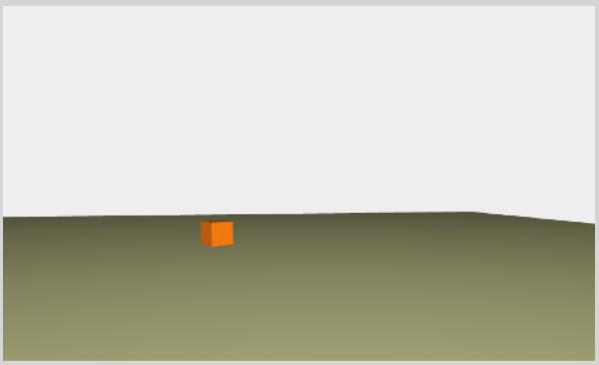
```
camera.lookAt(new THREE.Vector3(0.5, 0, 0));
```

**카메라가 피사체를 비추는 방향은 position
카메라가 바라보고 있는 방향 lookAt

둘의 차이점은 position의 경우 각도에 따라 피사체의 다른면 또는 여러 각도에서 볼 수 있고,
lookAt의 경우 position에 의해 그렇게 보이는 피사체를 일정하게 바라보는 방향이다.

예시)

position(0,0,1)
lookAt(0.5, 0.5 ,1)
- 카메라가 바라보는 각도가 다른 것을 확인할 수 있다.



position(0,1,10)
lookAt(2, 0 ,0)
-position에 의해 멀리서 렌더링이 되었지만 카메라가 바라보는 각도는 x축 2로 인해 땅이 오른쪽으로 치우친 모습을 볼 수 있다.