**Modify the NFT image onchain**

**SVG NFT Flipping the Mood**

With the assurance that our tokenURI function is returning a correctly formatting string, for both our tokenURI itself *and* our imageURI, I think we're ready to make this NFT dynamic!

Because our SVGs are on-chain, this affords us the ability to easily swap between them by calling a function. Let's write that function now.

Our first consideration should be that *only the owner* of an NFT should be able to flip its mood. We can use the \_isApprovedOrOwner function, included within the ERC721 standard to verify this before our flipMood function execution.

function flipMood(uint256 tokenId) public {

if(!\_isApprovedOrOwner(msg.sender, tokenId)){

revert MoodNFT\_\_CantFlipMoodIfNotOwner();

}

}

Remember to create our new custom error at the start of the contract! `error MoodNFT\_\_CantFlipMoodIfNotOwner();`.

From here, we'll just check if it NFT is happy, and if so, make it sad, otherwise we'll make it happy. This will flip the NFT's mood regardless of it's current mood.

With openzeppelin version 5.0.0 `\_isApprovedOrOwner` was removed in favor of a new `\_isAuthorized` function. However, this function has a downside.

\_WARNING: This function assumes that\_ \_`owner`\_ \_is the actual owner of\_ \_`tokenId`\_ \_and does not verify this assumption.\_

A more elegant solution is to check for approval and for the owner of the token separately. Therfore, we can use `getApproved()` and `ownerOf()` from openzeppelin contract in `ERC721.sol`.

function flipMood(uint256 tokenId) public view {

if(getApproved(tokenId) != msg.sender && ownerOf(tokenId) != msg.sender){

revert MoodNFT\_\_CantFlipMoodIfNotOwner();

}

if(s\_tokenIdToMood[tokenId] == Mood.HAPPY){

s\_tokenIdToMood[tokenId] == Mood.SAD;

} else{

s\_tokenIdToMood[tokenId] == Mood.HAPPY;

}

}

**Wrap Up**

This was easy enough, we've now got a function which an owner can use to flip the mood of their NFT. Wonderful!

In the next lesson we'll walkthrough the creation of our deployment script!