# 344.https://stackoverflow.com/questions/71506308/how-to-call-mint-on-erc721-without-emit-transfer

**T:**How to call \_mint() on erc721 without emit Transfer

**Q:**I've read here that it is possible to mint 2^256 nfts in a single transaction. I've tried to achieve this by directly assigning \_owners and \_balances mappings but ofc these are private variables so i can't change them. I tried making an \_mint() override but that also didn't work. How does this process work?

2 **Answer**

**A1:**For simplification, let's do a 10k NFTs scenario.  
  
It's not about invoking a single mint() function 10k times, rather than building your contract logic in a way that allows setting up a range of valid IDs.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Using the MFS part of IPFS, you can upload multiple files into a folder using the same directory ID and actual file names. Example:  
  
https://ipfs.io/ipfs/<dir\_id\_abc>/1.jsonhttps://ipfs.io/ipfs/<dir\_id\_abc>/2.jsonhttps://ipfs.io/ipfs/<dir\_id\_abc>/3.jsonetc...  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
These metadata files contain links to the images.  
  
Your contract can then implement a custom function that shadows an authorized address as an owner of the NFT if both following conditions are met:  
  
 ● The ID is in a valid range (in our case 1-10k)  
  
 ● The NFT is not owned by anybody else (i.e. it's owned by the default address 0x0)  
  
function \_exists(uint256 tokenId) override internal view returns (bool) { if (tokenId >= 1 && tokenId <= 10000) { return true; } return super.\_exists(tokenId);}function ownerOf(uint256 tokenId) override public view returns (address) { address owner = \_owners[tokenId]; // The ID is in a valid range (in our case 1-10k) // The NFT is not owned by anybody else (i.e. it's owned by the default address 0x0) if (tokenId >= 1 && tokenId <= 10000 && owner == address(0x0)) { // shadows an authorized address as an owner return address(0x123); } return super.ownerOf(tokenId);}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
The tokenURI() function then validates the token existence (using the \_exists() function) and returns the final URI concatenated from the base URI (https://ipfs.io/ipfs/<dir\_id\_abc>/), the ID, and the .json suffix.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Mind that this approach does not work on the OpenZeppelin implementation, as their \_owners property is private and not readable from child contracts. But you can take this snippet as an inspiration for a custom implementation that allows simulating an arbitrary default owner of 10k (or even 2^256) tokens.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**C1:**While on the topic of OpenZeppelin, do they decide the standard of the ERC721 tokens? Meaning, do you have to implement their contracts in order to be listed in opensea for example?

**C2:**@EduardoRobles The ERC-721 standard has gone through several stages of reviews and now is finalized and cannot be changed anymore... OpenZeppelin is "just" a company that publishes their implementation following this standard as an opensource library, but they cannot change the standard at the moment... So assuming that OpenSea accepts and can work with any contract following the standard (including these that don't emit the Transfer() event while minting during contract creation), you can use other implementations (other than OpenZeppelin) as well.

**C3:**@EduardoRobles But the thing is - OpenSea and other marketplaces might not recognize these tokens minted without the Transfer() event, even though it's in accordance with the standard. It depends on the implementation of each of these marketplaces.

**A2:**Tbh I don't know how that could be possible without paying ungodly amounts of gas. Why are you trying to mint that many tokens? Are you trying to get all the NFTs in a collection? If so, you'll have to pay the gas costs for every mint regardless.

**C1:**They do state on that post that it's possible to mint them under 200k gas. I'm not trying to mint 2^256 anyway, And it's not about saving money but time.

**C2:**Right but that post doesn't get into the specifics of how that would happen in the code. So I'd ask him for more details on how that happens via DM. Best of luck!

**C3:**Yeah, I just sent a reply asking for some demo. I trust him since he created the hardhat-deploy library, he probably knows what's he's doing. There has to be a trick to it

**C4:**For sure keep me updated I'd love to know how too.