# 300.https://stackoverflow.com/questions/71731951/how-to-remove-zeros-in-tokenid-in-solidity

**T:**How to remove zeros in tokenID in Solidity?

**Q:**I'm learning Solidity, there's a problem that keeps me busy for days  
  
Basically, I'm going to mint a bunch of NFTs.  
  
The images are 1.png, 2.png, 3.png  
  
The corresponding metadata has the same number as image, so it's 1.json, 2.json, 3.json etc.  
  
And the link to the metadata in Pinata isipfs://blahblahblah/{id}.json  
  
However, due to the clients interpretation, on OpenSea the URI becomesipfs://blahblahblah/00000000000000000000000000000000000000000000000000000000000000xx.json  
  
Which means:  
  
"1.json" becomes 0000000000000000000000000000000000000000000000000000000000000001.json  
  
"2.json" becomes 0000000000000000000000000000000000000000000000000000000000000002.json....  
  
As a result, OpenSea can't get the data of the NFT, because the link is not correct  
  
Is there a way to convert the tokenID to strip off all the zeros, and leave only the number?  
  
Thank you so much  
  
Here's the code  
  
// SPDX-License-Identifier: MITpragma solidity >=0.4.22 <0.9.0;import '@openzeppelin/contracts/token/ERC1155/ERC1155.sol';import '@openzeppelin/contracts/access/Ownable.sol';contract ArtCollectibleV2 is Ownable, ERC1155 {// Base URIstring private baseURI;string public name;constructor() ERC1155( 'ipfs://blahblahblah/{id}.json' ){ setName('Collection Name'); }function setURI(string memory \_newuri) public onlyOwner { \_setURI(\_newuri);}function setName(string memory \_name) public onlyOwner { name = \_name;}function mintBatch(uint256[] memory ids, uint256[] memory amounts) public onlyOwner{ \_mintBatch(msg.sender, ids, amounts, '');}function mint(uint256 id, uint256 amount) public onlyOwner { \_mint(msg.sender, id, amount, '');}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**C1:**can you share related code of your contract?

**C2:**Sure, I just updated the full code of the contract

1 **Answer**

**A1:**In your imported ERC1155.sol contract, there is a virtual function that returns the uri of the token:  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
 /\*\* \* @dev See {IERC1155MetadataURI-uri}. \* \* This implementation returns the same URI for \*all\* token types. It relies \* on the token type ID substitution mechanism \* https://eips.ethereum.org/EIPS/eip-1155#metadata[defined in the EIP]. \* \* Clients calling this function must replace the `\{id\}` substring with the \* actual token type ID. \*/ function uri(uint256) public view virtual override returns (string memory) { return \_uri; }  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
You can override it and prefix 00..00 to the token id to be compatible with the OpenSea  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

**C1:**Ah, I see. Do you mean removing the prefix 00..00 to be compatible with OpenSea ? Is it possible if you can show me the code on how to do that? I think of trimming the id, but I'm still new to Solidity, and it seems Solidity doesn't support it yet.

**C2:**Due to the clients interpretation can you explain it more?

**C3:**" By this mechanism, any occurrence of the {id} substring in either the URI or any of the amounts in the JSON file at said URI will be replaced by clients with the token type ID. For example, the token-cdn-domain{id}.json URI would be interpreted by clients as token-cdn-domain/… for token type ID 0x4cce0. " Source: docs.openzeppelin.com/contracts/3.x/api/token/…

**C4:**so, due to the intepretation, my metadata URL will be changed automatically from ipfs://blahblahblah/1.json To ipfs://blahblahblah/0000000000000000000000000000000000000000000000000000000000000001.json And because there's no such file in the IPFS, which means my NFT will have no metadata