# 9.https://stackoverflow.com/questions/69302320/erc721-transfer-caller-is-not-owner-nor-approved

**T:**ERC721: transfer caller is not owner nor approved

**Q:**I have a nftToken Contract that mints token to msg.sender, then I have a function in a market contract that transfers the nft from owner to market contract. However, I am getting an error that says: ERC721: transfer caller is not owner nor approved.  
  
here is my nftContract (nft) function snippet:  
  
function createToken(string memory tokenURI) public returns (uint) { \_tokenIds.increment(); uint256 newItemId = \_tokenIds.current(); \_mint(msg.sender, newItemId); \_setTokenURI(newItemId, tokenURI); setApprovalForAll(contractAddress, true); return newItemId;}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
here is my market code (stripeMarket Contract) function snippet:  
  
function createItem( address nftContract, uint256 tokenId ) public payable{ address \_owner = IERC721(nftContract).ownerOf(tokenId); IERC721(nftContract).transferFrom(\_owner, address(this),tokenId); IERC721(nftContract).approve(address(this),tokenId); }  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
and here I am trying to call it from the frontend with web3:  
  
const getItems=async()=>{ await contracts.nft.methods.createToken("https://i.ytimg.com/vi/nYxGhQYi0s4/maxresdefault.jpg").send({from: accounts[0]}); const owners = await contracts.nft.methods.ownerOf(1).call({from:accounts[0]}); await contracts.stripeMarket.methods.createItem(contracts.nft.\_address,1).send({from: {owners}}); }  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
But I am getting the error:  
  
ERC721: transfer caller is not owner nor approved.

1 **Answer**

**A1:**When the nftContract executes the setApprovalForAll(contractAddress, true), it allows the contractAddress (the Market contract) to operate all of the nftContract's tokens.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
But the newly minted token is owned by the msg.sender - not by the nftContract. So the approval does not apply to this token.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Depending on your use case, you can  
  
 ● Mint the new token to the nftContract (instead of the msg.sender) so that the Market contract is allowed to operate it. Or mint it to the Market contract directly.  
// the owner is the `nftContract`  
\_mint(address(this), newItemId);  
  
// the Market contract is allowed to operate the `nftContract`'s tokens  
setApprovalForAll(contractAddress, true);  
  
  
  
 ● Have the msg.sender (the token owner) execute approve(marketAddress, tokenId) on the nftContract before executing the createItem().  
This will give the Market contract approval to operate this particular token owned by the msg.sender. (Assuming it's the same address as the \_owner - otherwise it will fail.)