The reason the __getattr__ method is called only after looking for the attribute in the object's dictionary is because the __getattr__ method follows the MRO of the object. Therefore, when getting an attribute from the object using getattr, it will first look into the attribute dictionary of the object, and then give the user the opportunity to handle the missing attribute from the dictionary of the object, and then continue into the rest of the MRO by then looking at the base class of the object. This is not the case for the __setattr__ and __delattr__ methods, as these methods only set and delete attributes within the object itself, and therefore don't go through the MRO of the object. Therefore, the setattr and delattr dunders will already only look into the attribute dictionary of the object, and then give the user the opportunity to handle missing attributes.