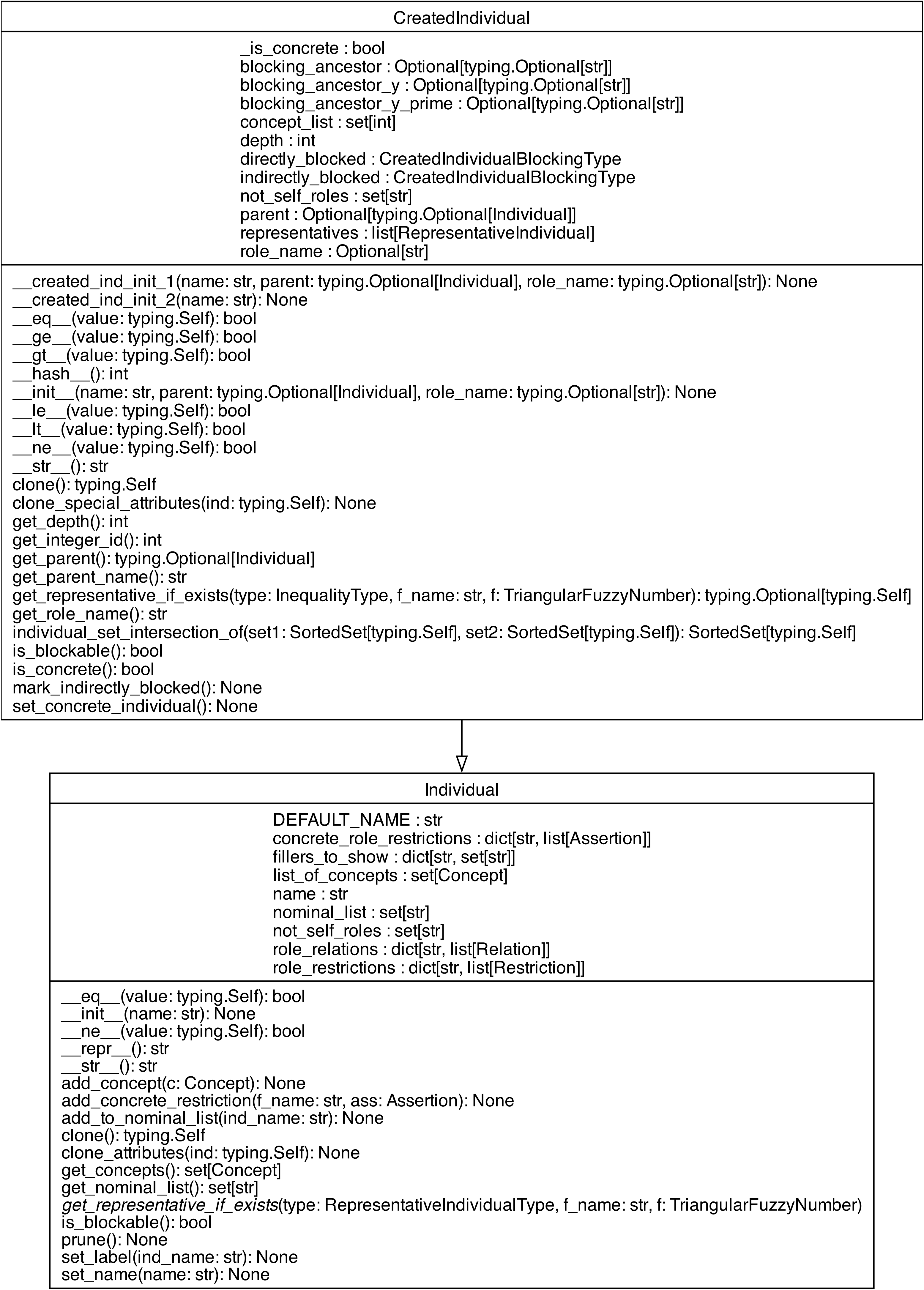
CreatedIndividual
_is_concrete : bool
blocking_ancestor : Optional[typing.Optional[str]]
blocking_ancestor_y : Optional[typing.Optional[str]]
blocking_ancestor_y_prime : Optional[typing.Optional[str]]
concept_list : set[int]
depth : int
directly_blocked : CreatedIndividualBlockingType
indirectly_blocked : CreatedIndividualBlockingType
not_self_roles : set[str]
parent : Optional[typing.Optional[Individual]]
representatives : list[RepresentativeIndividual]
role_name : Optional[str]
__created_ind_init_1(name: str, parent: typing.Optional[Individual], role_name: typing.Optional[str]): None
__created_ind_init_2(name: str): None
__eq__(value: typing.Self): bool
__ge__(value: typing.Self): bool
__gt__(value: typing.Self): bool
__hash__(): int
__init__(name: str, parent: typing.Optional[Individual], role_name: typing.Optional[str]): None
__le__(value: typing.Self): bool
__lt__(value: typing.Self): bool
__ne__(value: typing.Self): bool
__str__(): str
clone(): typing.Self
clone_special_attributes(ind: typing.Self): None
get_depth(): int
get_integer_id(): int
get_parent(): typing.Optional[Individual]
get_parent_name(): str
get_representative_if_exists(type: InequalityType, f_name: str, f: TriangularFuzzyNumber): typing.Optional[typing.Self]
get_role_name(): str
individual_set_intersection_of(set1: SortedSet[typing.Self], set2: SortedSet[typing.Self]): SortedSet[typing.Self]
is_blockable(): bool
is_concrete(): bool
mark_indirectly_blocked(): None
set_concrete_individual(): None
Individual
DEFAULT_NAME : str
concrete_role_restrictions : dict[str, list[Assertion]]
fillers_to_show : dict[str, set[str]]
list_of_concepts : set[Concept]
name : str
nominal_list : set[str]
not_self_roles : set[str]
role_relations : dict[str, list[Relation]]
role_restrictions : dict[str, list[Restriction]]
__eq__(value: typing.Self): bool
__init__(name: str): None
__ne__(value: typing.Self): bool
__repr__(): str
__str__(): str
add_concept(c: Concept): None
add_concrete_restriction(f_name: str, ass: Assertion): None
add_to_nominal_list(ind_name: str): None
clone(): typing.Self
clone_attributes(ind: typing.Self): None
get_concepts(): set[Concept]
get_nominal_list(): set[str]
get_representative_if_exists(type: RepresentativeIndividualType, f_name: str, f: TriangularFuzzyNumber)
is_blockable(): bool
prune(): None
set_label(ind_name: str): None
set_name(name: str): None