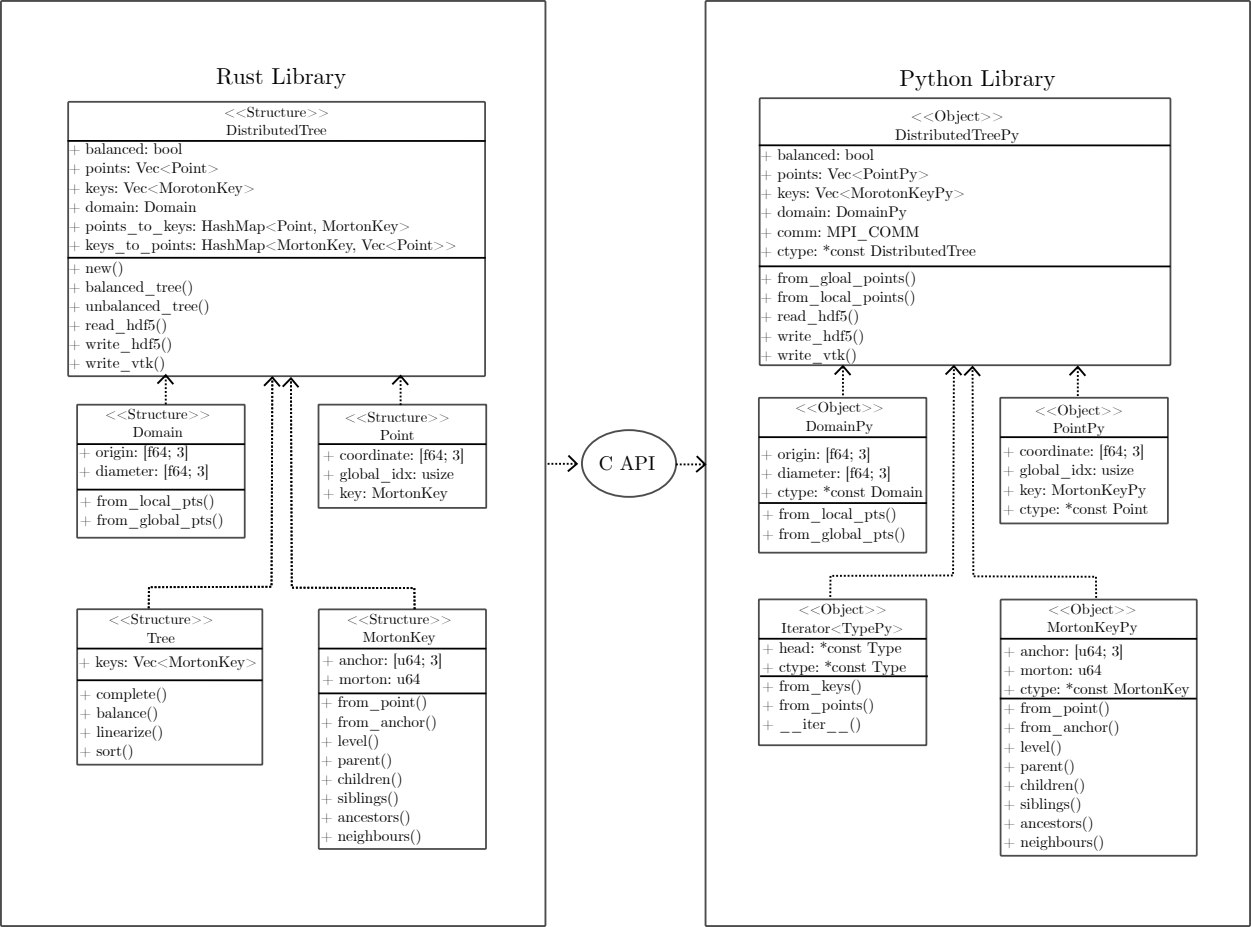
Rust Library
<<Structure>>
DistributedTree
+ balanced: bool
+ points: Vec<Point>
+ keys: Vec<MorotonKey>
+ domain: Domain
+ points_to_keys: HashMap<Point, MortonKey>
+ keys_to_points: HashMap<MortonKey, Vec<Point>>
+ new()
+ balanced_tree()
+ unbalanced_tree()
+ read_hdf5()
+ write_hdf5()
+ write_vtk()
<<Structure>>
Domain
+ origin: [f64; 3]
+ diameter: [f64; 3]
+ from_local_pts()
+ from_global_pts()
<<Structure>>
Point
+ coordinate: [f64; 3]
+ global_idx: usize
+ key: MortonKey
<<Structure>>
Tree
+ keys: Vec<MortonKey>
+ complete()
+ balance()
+ linearize()
+ sort()
<<Structure>>
MortonKey
+ anchor: [u64; 3]
+ morton: u64
+ from_point()
+ from_anchor()
+ level()
+ parent()
+ children()
+ siblings()
+ ancestors()
+ neighbours()
C API
Python Library
<<Object>>
DistributedTreePy
+ balanced: bool
+ points: Vec<PointPy>
+ keys: Vec<MorotonKeyPy>
+ domain: DomainPy
+ comm: MPI_COMM
+ ctype: *const DistributedTree
+ from_gloal_points()
+ from_local_points()
+ read_hdf5()
+ write_hdf5()
+ write_vtk()
<<Object>>
DomainPy
+ origin: [f64; 3]
+ diameter: [f64; 3]
+ ctype: *const Domain
+ from_local_pts()
+ from_global_pts()
<<Object>>
PointPy
+ coordinate: [f64; 3]
+ global_idx: usize
+ key: MortonKeyPy
+ ctype: *const Point
<<Object>>
Iterator<TypePy>
+ head: *const Type
+ ctype: *const Type
+ from_keys()
+ from_points()
+ __iter__()
<<Object>>
MortonKeyPy
+ anchor: [u64; 3]
+ morton: u64
+ ctype: *const MortonKey
+ from_point()
+ from_anchor()
+ level()
+ parent()
+ children()
+ siblings()
+ ancestors()
+ neighbours()