1. **BaseDocument** – Any class that descends from this object has access to storing, Tags, Links and ability to have References.

Example Usage:

public class Car : BaseDocument

{

public int Year { get; set; } = 2016;

public string Make { get; set; } = "Chevrolet";

public string Model { get; set; } = "Camaro";

}

1. **RepositoryBase<TBaseDocument>** - Allows saving, retrieving or updating of all documents.

Example Usage:

var rep = new RepositoryBase<Car>();

var car = rep.Find(id);

var chevies = rep.AsQueryable().Where(x => x.Make == "Chevrolet").ToList();

rep.Upsert(car);

rep.RemoveOne(car);

car.SaveToDatabase()

1. ReferenceObject<TBaseDocument> - Used as a lightweight object to get around circular references and allowing for turning back into BaseDocument.

Example Usage:

public class Car : BaseDocument

{

public int Year { get; set; } = 2016;

public string Make { get; set; } = "Chevrolet";

public string Model { get; set; } = "Camaro";

public CarRef AsCarRef()

{

return new CarRef(this);

}

}

public class CarRef : ReferenceObject<Car>

{

// This will automatically get resolved

public int Year { get; set; }

public Car AsCar()

{

return ToBaseDocument();

}

public CarRef(Car document) : base(document)

{

}

}

1. **RefrenceList<BaseDocument, ReferenceObject>** - Ability to work with Reference objects easier with some cool tricks.

Example Usage:

public class CarLot

{

public string Name { get; set; } = "Parkway Chevrolet";

public ReferenceList<Car,CarRef> Cars { get; set; } = new ReferenceList<CarCarRef>();

}

Useful methods:

ResyncWithDatabase(List<TReferenceObject> documents)

AddListOfReferenceObjects(List<TReferenceObject> documents)

AddReferenceObject(TReferenceObject document)

RemoveDocumentRef(TReferenceObject document)

RefreshAll() // Updates any values from real document

List<TBaseDocument> AsListOfBaseDocuments()

1. Document Linking – Save a Link to any other BaseDocument

Example Usage:

var repCars = new RepositoryBase<Car>();

var camaro = repCars.AsQueryable().Single(x => x.Model == "Camaro");

var repHumans = new RepositoryBase<Human>();

var human = repHumans.AsQueryable().Single(x => x.Name == "Marc Pike");

camaro.CreateLink(human); // Both human and Camaro will get a link to each other

camaro.CreateLinkOneWay(human); // Only Camaro has link to human

camaro.RemoveLink(human);

Other methods:

RemoveLinkForTypeAndId(Type type, string id)

RemoveLinkOneWay(BaseDocument attachedDoc)

RemoveAllLinks()

RemoveAllLinksForType(Type type)

List<Link> GetAllLinks()

List<Link> GetAllLinksForType(string typeFullName)

List<Link> GetAllLinksForType(Type type)

1. **LinkResolver** – Use this to get back the Linked BaseDocuments

Example Usage:

var humans = new LinkResolver<CHuman>(car).GetAllLinkedDocuments();

Other useful method:

new LinkResolver<CHuman>(car).RemoveAllLinksForThisType()

new LinkResolver<CHuman>(car).GetAllLinks()

1. **Tag properties** - Allow for additional information to any BaseDocument

Example Usage:

var repCars = new RepositoryBase<Car>();

var camaro = repCars.AsQueryable().Single(x => x.Model == "Camaro");

camaro.SetTagValue("Color", "Yellow");

camaro.SetTagValue("Engine", "LT1");

camaro.SetTagValue("EngineType", "V8");

var color = camaro.GetTagValue("Color", defaultValue: "Black");

1. Version – All BaseDocument classes have a Version

Example Usage:

camaro.Version.Version = "2.1";

camaro.Version.Comments = "Upgraded for performance";

camaro.Version.ExecutedOn = DateTime.Now;

1. C# Method of Resolving Model Changes:

NOTE:

All BaseDocument objects support the ISupportInitialize interface

which means they have a BeginInit()/EndEnit() virtual methods

All BaseDocument objects have a reference to BsonExtraElements

How to handle Model changes:

public override void EndInit()

{

If (Version.Version = “1.0”) {

object **nameValue**;

if (ExtraElements.TryGetValue("FrstName", out **nameValue**))

{

var firstName = (string)**nameValue**;

ExtraElements.Remove("FrstName");

FirstName = firstName;

}

if (ExtraElements.TryGetValue("LstName", out **nameValue**))

{

var lastName = (string)**nameValue**;

ExtraElements.Remove("LstName");

LastName = lastName;

}

Version.Version = “1.1”;

Version.Comments = “Fieldname changes”;

Version.ExecutedOn = DateTime.Now;

SaveDatabase();

}

}