# DBContext property

private XxxxxxContext context { get; set; }

# LINQ methods to create or run query expressions

Where(expression)-filters entity by expression

OrderBy(expression)-orders by the expression

FirstOrDefault(expression)-returns first instance matching expression, null if none found

ToList()-returns objects in List<T>

# Method of DbSet<Entity> class gets entity by id

Find(id)-returns first instance of entity of id, null if none found

# Add LINQ namespace import directive

Using System.Linq;

# Build a query expression

IQueryable<Xxxxxx> query = context.Xxxxxx.OrderBy(m => m.*Property*);

# Execute a query expression

IQueryable<Xxxxxx> xxxxxx = query.ToList();

# Build and execute a query expression

var xxxxxx = context.Xxxxxx.OrderBy(m => m.*Property*).ToList();

# Build a query expression chaining LINQ methods

var xxxxxx = context.Xxxxxx.Where(m => m.*Property* > value).OrderBy(m => m.*Property*);

# Build a query expression multi-line

IQueryable<Xxxxxx> query = context.Xxxxxx;

query = query.Where( m => m. *Property* > value);

query = query.Where( m => m. *Property* < value);

query = query.OrderBy( m => m. *Property*);

# Various query executions to find id

int id = *value*;

var xxxxx = context.Xxxxxx.Where(m => m.XxxxxxId == id).FirstOrderDefault();

var xxxxx = context.Xxxxxx. FirstOrderDefault (m => m.XxxxxxId == id);

var xxxxx = context.Xxxxxx.Find(id);