Getting Started (Saturday, January 16, 2021)

This guide compliments the sample programs, the User's Guide and the API reference provided on our site. Developers should review this in order to better understand how to quickly build a VelocityDB-integrated application.

We made a getting started video as well. To see it click here.

Getting assemblies

The simplest way to get started is by using the nuget package.

You can download the entire product package setup from the VelocityDB website. After you download it, run it to install it on your PC.

Add a reference to VelocityDB.dll

If you used the nuget package then you don't need to do this step, installing the nuget automatically adds this reference. Otherwise right click on the References folder of your project in Visual Studio's Solution Explorer and select Add Reference.... Navigate to the folder where VelocityDB was installed, by default it is in C:\Program Files (x86)\VelocityDb.

- ✓ VelocityDB.dll

If you target platform is Linux, add a reference to VelocityDBMono.dll instead

Add a reference to System.Transactions assembly

This is not a NuGet, it is an assembly that is part of the standard .NET Framework.

Add required using Statements

As a minimum you will need the following using statements:

```
using VelocityDb;
using VelocityDb.Session;
```

Define storable classes

```
public class Employee : OptimizedPersistable  // This base class contains the object identifier and some helpful properties and functions.
  {
    Company m_company;
    string m_firstName;
    string m_lastName;
    int m_age;
    DateTime m_hireDate;
    string m_city;

    public Company Employer
    {
      get
      {
        return m_company;
      }
      set
      {
        Update(); // required for VelocityDB to know you want to persistently update an object and its possible indices
        m_company = value;
    }
  }

  public string LastName
  {
    get
    {
      return m_lastName;
    }
    set
    {
      Update();
      m_lastName = value;
    }
  }
  public string FirstName
  {
    get
    {
      return m_firstName;
    }
    set
    {
      Update();
      m_firstName = value;
    }
  }
  public int Age
  {
    get
    {
      return m_age;
    }
    set
    {
      Update();
      m_age = value;
    }
  }
  public DateTime HireDate
  {
    get
    {
      return m_hireDate;
    }
    set
    {
      Update();
      m_hireDate = value;
    }
  }
  public string City
  {
    get
    {
      return m_city;
    }
    set
    {
      Update();
      m_city = value;
    }
  }
}

public class Company : OptimizedPersistable // This base class contains the object identifier and some helpful properties and functions.
{
  string m_name;
  string m_address;
  string m_phone;
  public string Name
  {
    get
    {
      return m_name;
    }
    set
    {
      Update(); // required for VelocityDB to know you want to persistently update an object and its possible indices
      m_name = value;
    }
  }
  public string Address
  {
    get
    {
      return m_address;
    }
    set
    {
      Update();
      m_address = value;
    }
  }
  public string Phone
  {
    get
    {
      return m_phone;
    }
    set
    {
      Update();
      m_phone = value;
    }
  }
```

Open/Create a database folder and store objects

```
class QuickStart
{
  static readonly string systemDir = "QuickStart"; // appended to SessionBase.BaseDatabasePath

  static int Main(string[] args)
  {
```

Create a session object within a using block so session is finalized after using it.

```
using (SessionNoServer session = new SessionNoServer(systemDir))
{
  Console.WriteLine("Running with databases in directory: " + session.SystemDirectory);
```

Always add a try catch block around update transactions

```
try
{
  session.BeginUpdate();
  Company company = new Company();
  company.Name = "MyCompany";
  session.Persist(company);
  Employee employee1 = new Employee();
  employee1.Employer = company;
  employee1.FirstName = "John";
  employee1.LastName = "Walter";
  session.Persist(employee1);
  session.Commit();
}
catch (Exception ex)
{
  Trace.WriteLine(ex.Message);
```

Always abort transaction if you get an exception

```
    session.Abort();
  }
}
Retrieve();
return 0;
```

Retrieve objects from database

```
static void Retrieve()
{
  using (SessionNoServer session = new SessionNoServer(systemDir))
  {
    session.BeginRead();
    // get all companies from database
    IEnumerable<Company> companies = session.AllObjects<Company>();

    // get all employees that FirstName contains "John"
    IEnumerable<Employee> employees = session.AllObjects<Employee>();
    var query = from Employee emp in employees
                where emp.FirstName.Contains("John")
                select emp;

    foreach (Employee emp in query)
    {
      //do something with employee
      emp.ToString();
    }

    // get all Employees that are over 30 years old and are from Berlin:
    var query2 = from Employee emp in employees
                 where emp.Age > 30 && emp.City == "Berlin"
                 select emp;

    foreach (Employee emp in query2)
    {
      //do something with employee
      emp.ToString();
    }
    // get all Employees that work at "MyCompany" company:
    var query3 = from Employee emp in employees
                 where emp.Employer.Name == "MyCompany"
                 select new { emp.FirstName, emp.LastName };
    session.Commit();
  }
}
```

For more examples, download our setup installer and take a look at our many sample projects here or in your installed VelocityDB (see %USERPROFILE%\My Documents\VelocityDB\VelocityDB.sln)