Entity Framework

Code-First Migrations

Lesson Objectives

- Migration in EF 6 Code-First
- Automated Migration
- Code-Based Migration
- Rollback Migration

CODE-FIRST MIGRATION

Migration in EF Code-First

- When you don’t have permission to drop database
- When you have important data, so you cannot drop database
- Migration: movement from one part of something to another.
  - ✓ Add or Remove entity from DbContext
  - ✓ Add or Remove or Update entity property

Automated Migration

- **HOW:**
  - ✓ PM> enable-migrations –EnableAutomaticMigration:$true
  - ✓ Set the database initializer to MigrateDatabaseToLatestVersion
- **PROS**:
  - ✓ Don't have to process database migration manually for each change
- **CONS**:
  - ✓ You may loss data in the corresponding column or table

Code-Based Migration

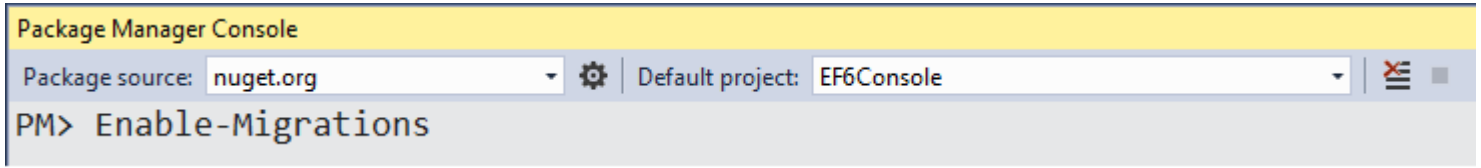
- Provides more control on the migration
- Allows to configure additional things
  - ✓ setting a default value of a column,
  - ✓ configure a computed column
  - ✓ ….
- You may consider to backup database first, before run any migration command

Code-Based Migration

- Process to migrate:
  - ✓ Step 1.0: Enable-Migrations:
    - Enables the migration in the project by creating a Configuration class.
  - ✓ Step 2.x: Add-Migration:
    - Creates a new migration class as per specified name with the Up() and Down() methods.
  - ✓ Step 3.x: Update-Database:
    - Executes the last migration file created by the Add-Migration command and applies changes to the database schema.

Code-Based Migration

- Step 1.0: Enable-Migrations:
  - ✓ Enables the migration in the project by creating a Configuration class.
  - ✓ Make sure that the default project is the project contains context class



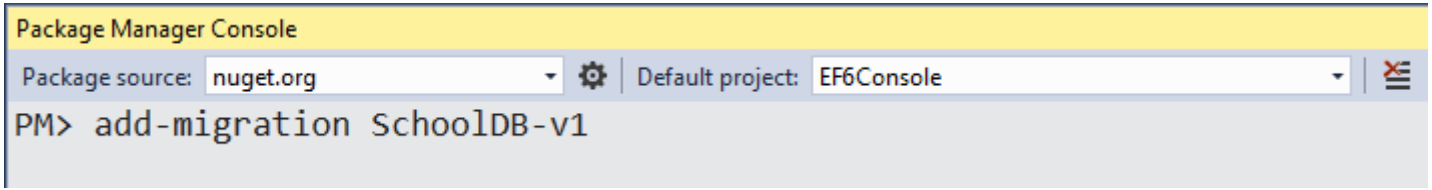
Code-Based Migration

- The command will create the Configuration class derived from DbMigrationsConfiguration with **AutomaticMigrationsEnabled = false**

```
public class SchoolContext: DbContext
{
    public SchoolDBContext(): base("SchoolDB")
    {
        Database.SetInitializer(new MigrateDatabaseToLatestVersion<SchoolDBContext, EF6Console.Migrations.Configuration>());
    }

    public DbSet<Student> Students { get; set; }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {

    }
}
```

Code-Based Migration

- ✓ Step 2.x: Add-Migration:
  - Creates a new migration class as per specified name with the Up() and Down() methods.
- ✓ Step 3.x: Update-Database:
  - Executes the last migration file created by the Add-Migration command and applies changes to the database schema.



Rollback Migration

- Rollback to a previous migration due to an error in the current migration or wanting to rewind and start over
- PM> update-database -TargetMigration:<name without timestamp>
- Notices:
  - ✓ The migration cannot rollback deleted data
  - ✓ The migrations which were rolled back were **not deleted**.
  - ✓ The next time perform an update-database command, the migrations will be re-executed.
  - ✓ To complete delete, we need to delete the migration from project entirely

Multiple DbContext

- Multiple DbContext was first introduced in Entity Framework 6.0.
- Multiple context classes may belong to a single database or two different databases.

Multiple DbContext

- Rules: always specify context for the command
- Syntax:

  -ContextTypeName <DbContext-Name-with-Namespaces> MigrationsDirectory:<Migrations-Directory-Name>

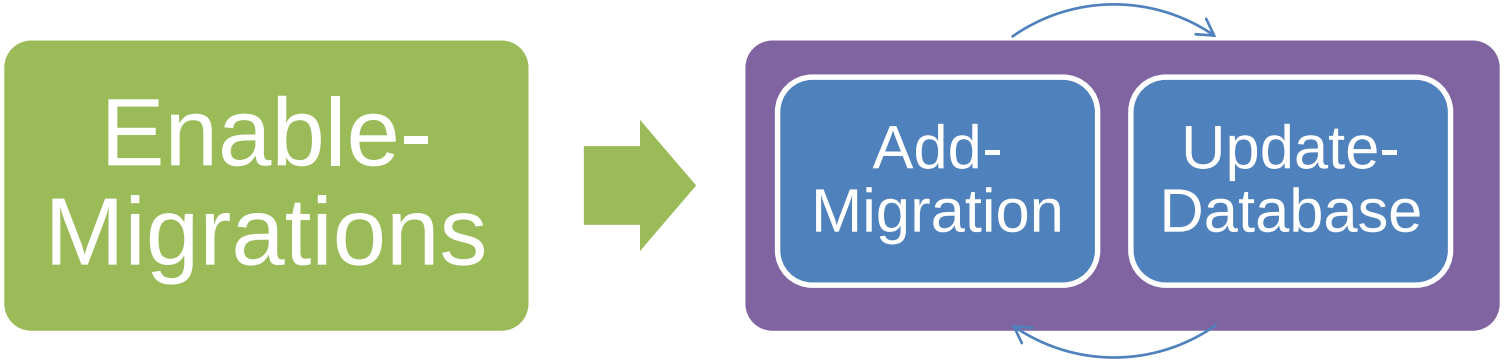
- Example:

```
PM→ enable-migrations -ContextTypeName:EFCodeFirstDemo.MyStudentContext
```

References

- https://coding.abel.nu/2012/03/ef-migrations-command-reference/
- https://dzone.com/articles/ef-migrations-command

Summary



Thank you