Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

Synopse mORMot framework

An Open Source Client-Server ORM/SOA framework

https://synopse.info

http://mormot.net

Contributors

Alan Chate

Alexander (sha)

Alexander (volax)

AlexPirate

Alfred Glaenzer (alf)

Andre Heider (dhewg)

Antoine Simard (AntoineGS)

Arnaud Bouchez

ASiwon

Aweste

Bas Schouten

BigStar

BugsDigger

Cheemeng

CoMPi

Damien (ddemars)

Darian Miller

Daniel Kuettner

David Mead (MDW)

Delphinium (louisyeow)

DigDiver

Dominikcz

EgorovAlex

Emanuele (lele9)

Eric Grange

Esmond

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

Esteban Martin (EMartin)

Eugene Ilyin

Eva Freimann (EVaF)

FeelAirSlow

F-Vicente

Goran Despalatovic (gigo)

Jean-Baptiste Roussia (jbroussia)

Joe (jokusoft)

Johan Bontes

Jordi Tudela

Kevin Chen

Lagodny

Leon Oosthuizen

Macc2010

Maciej Izak (hnb)

Marcos Douglas B. Santos (mdbs99)

Mario Moretti

Marius Maximus (mariuszekpl)

Martin Eckes

Martin Suer

Mapes

Matkov

Maxim Masiutin

Mazinsw

MChaos

Miab3

Michael (EgonHugeist)

Michalis Kamburelis

MilesYou

Mingda

Mr Yang (ysair)

Nicolas Marchand (MC)

Nortg

Nzsolt

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

Oleg Tretyakov

Ondrej (reddwarf)

Pavel Mashlyakovskii (mpv)

Pierre le Riche

RalfS

Richard6688

Rik (rvk)

Sabbiolina

Sanyin

Sinisa (sinisav)

Sllimr7139

SSoftPro

Stefan (itSDS)

Svetozar Belic (transmogrifix)

Transmogrifix

Uian2000

Vaclav

Vadim Orel

Willo vd Merwe

Win2014

Wloochacz

Wolfgang Ehrhardt

Yoanq

Ysair

Zed

[See below if you upgrade from 1.17 revision]

Synopse mORMot is an Open Source Client-Server ORM SOA MVC framework for Delphi 6 up to the latest Delphi and FPC revisions, targeting Windows/Linux for servers, and any platform for clients (including mobile or AJAX).

The main features of mORMot are therefore:

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

- ORM/ODM: objects persistence on almost any database (SQL or NoSQL);
- SOA: organize your business logic into REST services;
- Clients: consume your data or services from any platform, via ORM/SOA APIs;
- Web MVC: publish your ORM/SOA process as responsive Web Applications.

With local or remote access, via an auto-configuring Client-Server REST design.

Due to its modular design, switch from a Client-Server architecture over HTTP, named pipes or GDI messages into a stand-alone application is just a matter of mORMot classes initialization.
For instance, the very same executable can even be running stand-alone, as a server, as a service, or a client, depending on some run-time parameters!

Emphasizing simplicity, speed and versatility, mORMot is a incredibly well documented Open Source project easy enough to add basic ORM or Client-Server features to simple applications for hobbyists, or let experienced users develop scaling and strong service-based projects for their customers, with the advantages of native code and easy-to-deploy solutions, reducing deployment cost and increasing ROI.

It provides an Open Source self-sufficient set of units (even Delphi starter edition is enough) for creating any application, from a stand-alone solution up to the most complex Domain-Driven Design (DDD):

- Presentation layer featuring MVC UI generation with i18n and reporting (with pdf export) for rich Delphi clients, MVC web clients (with logic-less Mustache templates) or rich AJAX clients (via native JSON/REST access);

- Application layer implementing Service Oriented Architecture via interface-based services (like WCF) and Client-Server ORM (including method-based services) - following a RESTful model using JSON over several communication protocols (e.g. HTTP/1.1);

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

- Domain Model layer handling all the needed business logic in plain Delphi objects, including high-level managed types like dynamic arrays or records for Value Objects, dedicated classes for Entities or Aggregates, and variant storage with late-binding for dynamic documents;

- Data persistence infrastructure layer with ORM operations on direct Oracle, MS SQL, OleDB, ODBC, ZEOS/ZDBC access or any TDataSet provider (e.g. FireDAC/AnyDAC, UniDAC, NexusDB, BDE...), with a powerful SQLite3 kernel, and optional SQL access if needed, with amazing performance and advanced features like Array DML, auto-generating SQL for SQLite3, Oracle, Jet/MSAccess, MS SQL, Firebird, DB2, PostgreSQL, MySQL and NexusDB - and alternative high-speed MongoDB NoSQL database access for ODM persistence;

- Cross-Cutting infrastructure layers for handling data filtering and validation, security (e.g. Windows authentication or any custom model), caching, logging and testing (framework uses test-driven approach and features interface stubbing and mocking).

With mORMot, ORM/ODM is not used only for data persistence of objects (like in other implementations), but as part of a global n-Tier, Service Oriented Architecture (SOA), ready to implement Domain-Driven solutions. This framework is not an ORM on which a transmission layer has been added, like almost everything existing in Delphi, C# or Java: this is a full Client-Server ORM/SOA from the ground up.
This really makes the difference.

The business logic of your applications will be easily exposed as Services, and will be accessible from light clients (written in Delphi or any other mean, including AJAX).
The SpiderMonkey JavaScript engine has been integrated on the server side and can be used to define business rules or any process (including MVC web rendering) - just like node.js, but with a multi-threaded core, and the full power of our optimized Delphi libraries at hand.

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

The framework Core is non-visual: you will get everything you need in a consistent set of classes to be used from code. In order to let you focus on your business, using mORMot's KISS/DRY/SOC/YAGNI/TDD and Convention Over Configuration patterns. But you have also some UI units available (including screen auto-generation, reporting and ribbon GUI), and you can use it from any RAD, web, or AJAX clients (via JavaScript or Smart Mobile Studio).

No dependency is needed on the client side (no DB driver, or third-party runtime): it is able to connect via standard HTTP, even through a corporate proxy or a VPN. Rich Delphi clients can be deployed just by copying and running a stand-alone small executable, with no installation process. Stream can be encrypted via HTTS or with proven SHA/AES-256. Endpoints are configured automatically for each published interface on both server and client sides, and creating a load-balancing proxy is a matter of one method call.
Speed and scalability has been implemented from the ground up: a genuine optimized multi-threaded core let a single server handle more than 50,000 concurrent clients, faster than DataSnap, WCF or node.js, and our rich SOA design is able to implement both vertical and horizontal scalable hosting, using recognized enterprise-level SQL or NoSQL databases for storage.

Even if mORMot will be more easily used in a project designed from scratch, it fits very well the purpose of evolving any existing Delphi project, or creating the server side part of an AJAX application.

Licensed under a disjunctive tri-license giving you the choice of one of the three following sets of free software/open source licensing terms:
- Mozilla Public License, version 1.1 or later;
- GNU General Public License, version 2.0 or later;
- GNU Lesser General Public License, version 2.1 or later.

This allows the use of our code in as wide a variety of software projects as possible, while still maintaining copy-left on code we wrote.

Main project page:
http://mORMot.net

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

Documentation:

https://synopse.info/files/html/Synopse%20mORMot%20Framework%20SAD%201.18.html

Installation:

https://synopse.info/files/html/Synopse%20mORMot%20Framework%20SAD%201.18.html#TITL_113

FAQ:

https://synopse.info/files/html/Synopse%20mORMot%20Framework%20SAD%201.18.html#TITL_123

How to get the source:

https://synopse.info/fossil/wiki?name=Get+the+source

A forum is dedicated to support:

https://synopse.info

A blog is available:

http://blog.synopse.info

Issues and feature requests can be posted (take a look at the forums
and latest unstable version first!):

https://synopse.info/fossil/reportlist

You can also monitor/fork our projects on GitHub:

https://github.com/synopse/mORMot

You may also install it as a Delphinus package: Delphinus-Support

Don't forget to download the documentation (available online or as pdf files,
created by our SynProject tool).
In particular, you should take a look at all general introduction chapters
of the SAD document. It will cover all key-concepts and code modelling
used by the framework.
A developer guide is included in this SAD document, in its 2nd part. You'll

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

get good practice guidance, presentation of the ORM/SOA approach and other underlying concepts.

Feel free to contribute by posting enhancements and patches to this quickly evolving project.

Enjoy!

Some units (e.g. SynPdf, SynGdiPlus, SynBigTable, SynCommons, SynCrypto, SynDB*, SynSQLite3, SynMongoDB, SynMustache, SynSM, mORMotReport) are used by mORMot, but do not require the whole framework to be linked.
That is, you can use e.g. only PDF generation, SynDB fast database access, a static-linked SQLite3 engine, direct MongoDB access, Mustache templates, SpiderMonkey JavaSCript engine, code-generated reports, or the TDocVariant, TDynArray, TSynLog classes of SynCommons, without using the main mORMot units and features (ORM, Client-Server, services, UI).

Some of those units can even be compiled with Delphi 5 (e.g. SynPdf, SynDB).

Quick Steps when upgrading from a previous 1.17 revision:

1) Note that some units where renamed, and some breaking changes introduced by some enhanced features, therefore a direct update is not possible

2) Erase or rename your whole previous #\Lib directory

3) Download latest 1.18 revision files as stated just above

4) Change your references to mORMot units:
- Add in your uses clause SynLog.pas and/or SynTests.pas if needed;
- Rename in your uses clause any SQLite3Commons reference into mORmot;
- Rename in your uses clause any SQLite3 reference into mORMotSQLite3;

Wagner Hard- & Software
Georg-Schumann-Straße 294
04159 Leipzig
www.systemhaus-whs.de

- Rename in your uses clause any other SQlite3* reference into mORMot*;
- Add in one uses clause a link to SynSQLite3Static (for Win32).

5) Consult the units headers about 1.18 for breaking changes, mainly:
- TSQLRecord.ID: TID primary key, TIDDynArray, and TRecordReference are now Int64;
- Renamed Iso8601 low-level structure as TTimeLogBits;
- TJSONSerializerCustomReader/Writer callbacks changed;
- TSQLRestServerCallBackParams replaced by TSQLRestServerURIContext class;
- TSQLRestServerStatic* classes renamed as TSQLRestStorage*;
- rmJSON* enums replaced by TSQLRestRoutingREST/JSON_RPC classes;
- Changed '¤' into '~' character for mORMoti18n language files.