

TIME LIVE SOURCE CODE

INTRODUCTION AND IMPLEMENTATION DETAILS

OVERVIEW

The TimeLive provides enterprise-level architecture for simplified development of web based applications using the Microsoft .NET framework. The architecture consists of N-Tier architecture

N-Tier architecture produces a far more flexible and scalable than client/server environment. N-Tier architecture has a presentation layer and three separate layers - a business logic layer and a data access logic layer and a database layer. The next section discusses each of these layers in detail.

1-TIMELIVE (N-TIER APPLICATION IMPLEMENTATION)

N-tier architecture is typically thus defined by the following layers:

PRESENTATION LAYER:

This is a front-end component, which is responsible for portable presentation logic. TIMELIVE presentation logic layer consists of standard ASP.NET web forms, web controls, standard asp.net validations etc. This layer works with results/output of the business logic layer and transforms the results into something usable and readable by the end user.

BUSINESS LOGIC LAYER:

Allows users to share and control business isolating it from the other layers of the application. The business functions between the presentation layer and data access logic layers, sending the client's data requests to the database layer through the data access layer.

DATA ACCESS LOGIC LAYER:

Provides access to the database by executing a set of SQL statements or stored procedures. This is where you will write generic methods to interface with your data.

DATABASE LAYER:

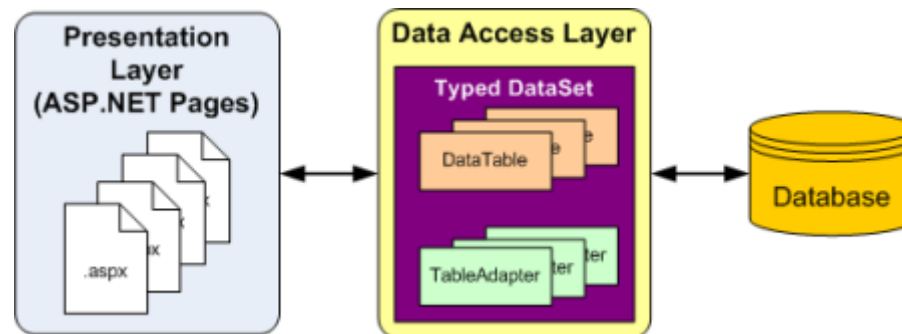
Make up of RDBMS database component such as SQL Server that provides the mechanism of store and retrieve data.

2-DNA ARCHITECTURE

If an alternative definition of DNA was brought to layman terms it would be scalability. Microsoft invented this concept where developing applications in function specific layers or tiers that an application could be scaled endlessly and maintained with low total cost of ownership. These tiers include Presentation, Business, Data Access, and Data Storage. The splitting of tier creates boundaries and allows encapsulation of specific functionality from tiers that should not have knowledge of other tiers.

3-DATA ACCESS LAYER USING TABLEADAPTER CONFIGURATION WIZARD

Traditionally the process you employ to create data access layer classes is a manual process, meaning that you first create a class and then add the appropriate methods to it. With the introduction of Visual Studio 2005, Microsoft has introduced a new TableAdapter Configuration Wizard that makes creating a data access logic layer class a breezy experience. Using this wizard, you can create a data access logic layer component without having to write a single line of code. This increases the productivity of the developers to a great extent. Once you create those classes, you can consume them exactly the same way you consume built-in objects. Before looking at an example, let us briefly review what a [TableAdapter] is. A TableAdapter connects to a database, executes queries, or stored procedures against a database, and fills a DataTable with the data returned by the query or stored procedure. In addition to filling existing data tables with data, TableAdapters can return new data tables filled with data. The TableAdapter Configuration Wizard allows you to create and edit TableAdapters in strongly typed datasets.



4-TIMELIVE IMPLEMENTATION HIGH LEVEL OVERVIEW

Some of the important characteristics of the TIMELIVE application are as follows:

- The data access layer classes are generated using the new TableAdapter Configuration Wizard, which enables you to create data access layer classes without writing a single line of code.
- ASP.NET Web forms in the user interface layer are generated using master pages, providing a consistent look and feel for the entire application.
- Web forms utilize ObjectDataSource control to directly bind the output of the middle tier methods to data bound controls such as a GridView control.
- Web forms also take advantage of caching of database contents to increase the performance and throughput of the web application. This is implemented using TimeLive own custom caching framework.
- Custom Membership Provider class in order to provide TIMELIVE own custom authentication implementation.
- Microsoft Active Directory Membership Provider implementation with very easily switching between between database authentication and Active Directory Authentication mode.
- System settings password protected area for changing system setting like SMTP settings, database connection related settings etc.
- Database driven ASP.Net SiteMap Provider implementation in order to provide customizable Role based Security in application.
- UI controls are based on free Ajax framework, free controls and free import/export framework in order to provide cost effective solution.
- All 3rd party controls are 100% managed control.
- Copy/Paste deployment with full support of deploying TimeLive on shared hosting environment.
- Selectable culture (configurable in application preferences)
- Selectable UI Language (English, Italian, Chinese, Dutch, German, French). All strings are retrieving from .net resource files.
- Full customizable reports implementation

- Export / Import to CSV / Quickbooks (IIF)
- Fully Microsoft AJAX framework implementation.

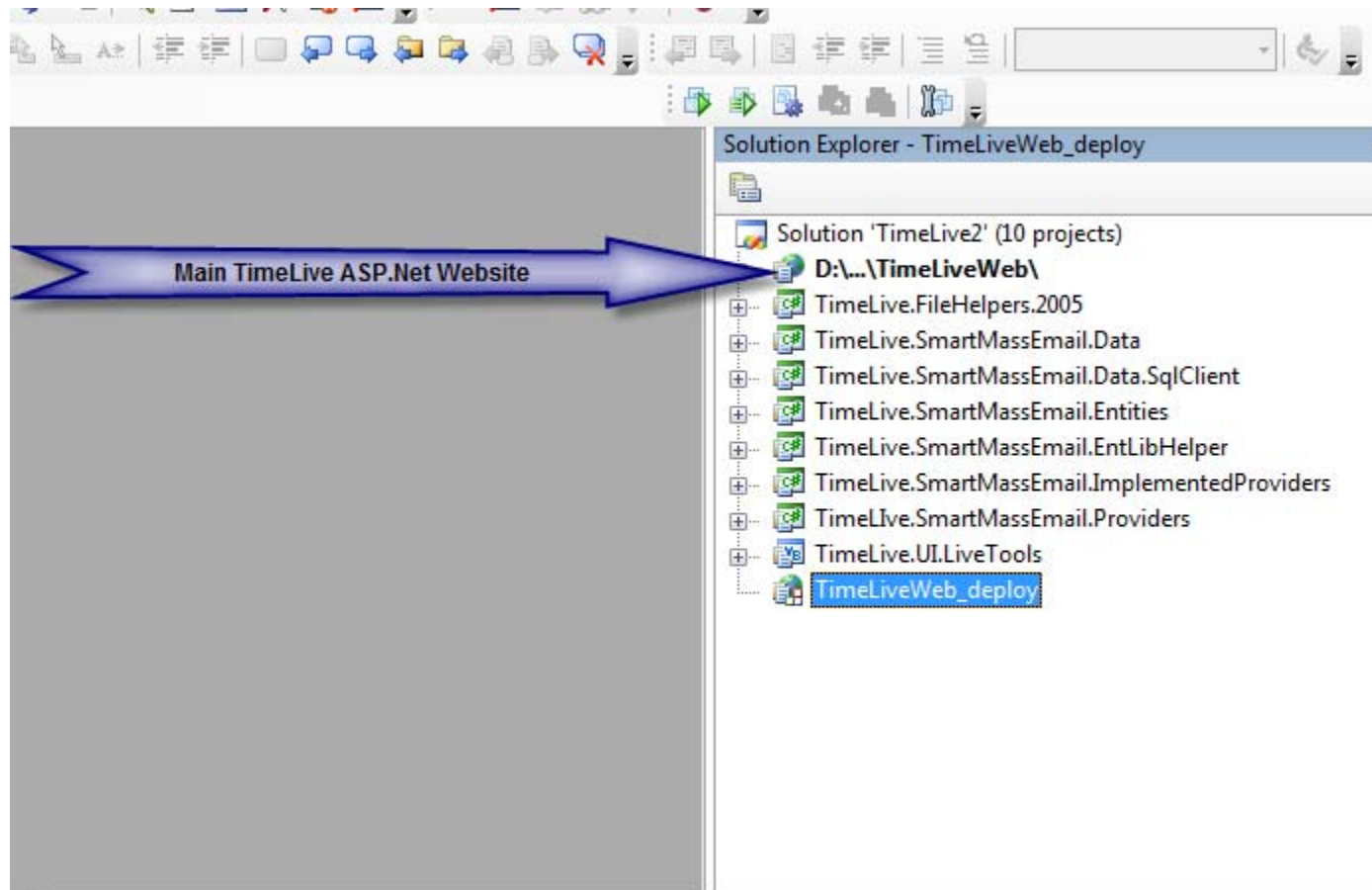
5- TIMELIVE SOURCE CODE (QUICK START)

1. Download and install Eworld UI 2.0.6. <http://www.eworldui.net/Download.aspx>
2. Download and install "Web Deployment Project" <http://msdn.microsoft.com/en-us/asp.net/aa336619.aspx>
3. Extract source code in any folder.
4. Launch TimeLive solution file "TimeLive2.sln" in Visual Studio 2005 or 2008.
5. Click on start button in Visual Studio to launch TimeLive from source code.
6. Complete database setup page and create your new TimeLive database.
7. Fill account add form and click on add to add a new account.
8. Click on "Go to login page" to navigate to login page.
9. Login with email id and password which you just use in registration in TimeLive.

6-TIMELIVE SOURCE CODE (SOLUTION AND PROJECTS)

TIMELIVE source code comprises of main TimeLiveWeb asp.net website with different dependent projects. Following are the functional TimeLive projects.

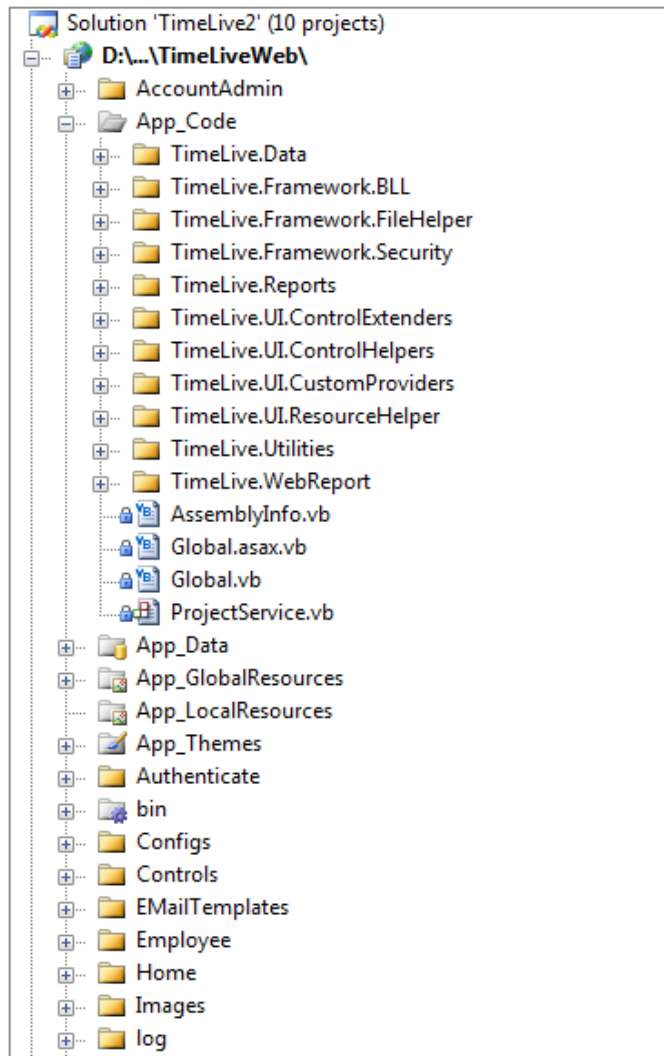
1. TimeLiveWeb (Main asp.net TimeLive Website)



2. TimeLive.FileHelper.2005 (Open source FileHelper project (<http://www.filehelpers.com/>))
3. TimeLive.SmartMassEmail.Data (Project of open source mass email framework)
<http://www.codeproject.com/KB/IP/smartmassemail.aspx>
4. TimeLive.SmartMassEmail.Data.SqlClient (Project of open source mass email framework)
<http://www.codeproject.com/KB/IP/smartmassemail.aspx>
5. TimeLive.SmartMassEmail.Entities (Project of open source mass email framework)
<http://www.codeproject.com/KB/IP/smartmassemail.aspx>
6. TimeLive.SmartMassEmail.EntLibHelper (Project of open source mass email framework)
<http://www.codeproject.com/KB/IP/smartmassemail.aspx>
7. TimeLive.SmartMassEmail.ImplementedProviders
<http://www.codeproject.com/KB/IP/smartmassemail.aspx>
8. (Project of open source mass email framework)
<http://www.codeproject.com/KB/IP/smartmassemail.aspx>
9. TimeLive.SmartMassEmail.ImplementedProviders
<http://www.codeproject.com/KB/IP/smartmassemail.aspx>
10. TimeLive.UI.LiveTools (ASP.Net Gridview extenders implementation)

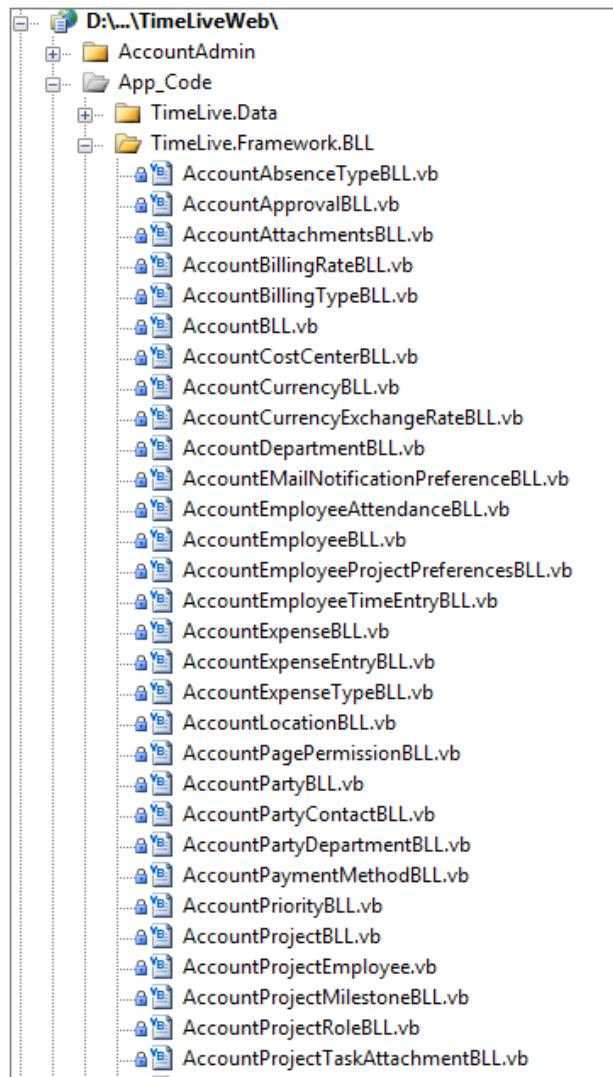
7-TIMELIVE WEB FOLDER STRUCTURE

TIMELIVE Website comprises of few built-in asp.net folders like “App_Code”, “App_Data” with application related folders like AccountAdmin, ProjectAdmin etc.



8-TIMELIVE WEB (BUSINESS LAYER CLASSES)

TimeLive presentation layer call these business layer class functions for database and non-database related operations. Every business layer class contains CRUD operation functions (Create, Read, Update, Delete) for its own table as well as provides business rules related to that particular business entity.



9-DATABASE LAYER:

These are the core TIMELIVE database tables and their corresponding business layer class.

Table Name	BLL class	Description
Account	AccountBLL	TimeLive multiple account management. AccountId differentiate multiple organization data in tables. [Account] with its 1:1 table [AccountPreferences] store single account primary information and preferences.
AccountEmployee	AccountEmployeeBLL	Employee comprises users in TimeLive. An employee table consists of all user information like id, personal information, user preferences, encrypted password etc.
AccountParty	AccountPartyBLL	AccountParty store all customer related information including contact information of customer.
AccountProject	AccountProjectBLL	AccountProject store every customer's project defined in TIMELIVE application. This table store project name, status, project estimated times, estimated cost, project status etc.
AccountProjectEmployee	AccountProjectEmployeeBLL	AccountProjectEmployee table store employee ids of all project members with their [Project based billing rate] and [Project based employee rate].
AccountProjectTask	AccountProjectTaskBLL	AccountProjectTask store all tasks of a project. This table also store [All Project Task] field in order to create generalized tasks between projects. It also store billable flag, parent task id etc of a particular task.

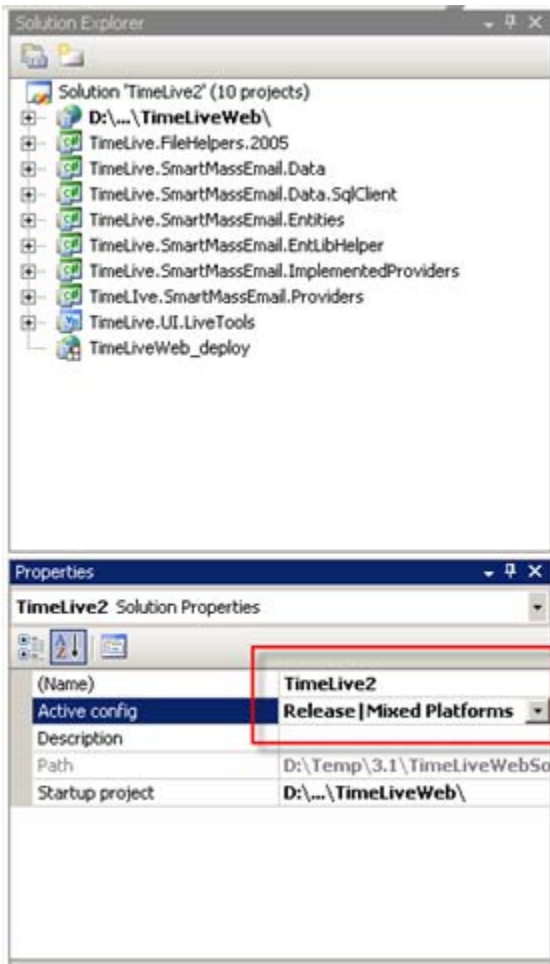
AccountProjectTaskEmployee	AccountProjectTaskEmployeeBLL	Store all employee ids who are assigned in a particular task.
AccountProjectRole	AccountProjectRoleBLL	Store role based billing rate in case of role based billing rate setup in project.
AccountEmployeeTimeEntry	AccountEmployeeTimeEntryBLL	AccountEmployeeTimeEntry table comprises of time entry records in TIMELIVE system. Every time entry record comprise of date, project, task, hours, billing rate, employee rate, billing rate currency, exchange rate, approval status flag etc.
AccountEmployeeTimeEntryApproval	AccountEmployeeTimeEntryApprovalBLL	Store all approval / rejection records of a time entry. This table store approver activities with approver's comments.
AccountExpense	AccountExpenseBLL	Expense code list
AccountExpenseEntry	AccountExpenseEntryBLL	Store employee based expense entry records of TimeLive. It store information like reimbursement, billable, currency, exchange rate, approval status etc.
AccountExpenseEntryApproval	AccountExpenseEntryApprovalBLL	Store history of approval / rejection records with approver's comments.
Account Currency	AccountCurrencyBLL	Store all applicable currencies for an account
AccountCurrencyExchangeRate	AccountCurrencyExchangeRateBLL	Store history based exchange rate of currencies.
AccountPagePermission	AccountPagePermissionBLL	Store Role based permission of pages
AccountProjectTaskAttachment	AccountProjectTaskAttachmentBLL	Project task attachments
AccountProjectTaskComments	AccountProjectTaskCommentsBLL	Project task comments
AccountProjectTaskEmployee	AccountProjectTaskEmployeeBLL	Project task employees

AccountWorkingDay	AccountWorkingDayBLL	Working day setup
Audit	AuditBLL	Audit trail of task
EmailMessage	EmailEngineBLL	Main email message table and its classes for sending email on different activities.

10-TIMELIVE DEPLOYMENT:

TimeLive source code comes with one pre-configured “Web Deployment Project” in TimeLive solution. Using [Web Deployment Project], developers can generate TimeLive deployment files. These deployment files can be published to your already running TimeLive application. You can also create your own installer if you want to distribute your customized TimeLive application using installer.

How to generate TimeLive deployment files:



1. Open web.config file and change “debug” to false of compilation tag.
`<compilation debug="false" defaultLanguage="vb">`
2. Select “TimeLive2” solution in solution explorer and press [F4] to open property of TimeLive solution.
3. Select [Release | Mixed Platform] in “Active Config” property.
4. Right click on TimeLiveWeb_deploy project and click on “Build” to start generating deployment files.
5. Deployment project will generate TimeLive deployment files in “TimeLive2\TimeLiveWeb_deploy\Release” folder.

How to create TimeLive installer for distributing TimeLive using your own installers?

<http://www.wisdombay.com/articles/article000014.htm>