2014

Templar
Feature Document

The purpose of this document is to:

- Understand new features in framework.
- This document is intended for Developers.

Sangram Chavan
Tavisca Solutions Pvt. Ltd
3/10/2014

If you have comments / queries about this documentation, email them to:

templar@tavisca.com

Table of Contents

Reduced number of framework assemblies to ease deployment

Templar now has only 5 required assemblies apart from third party assemblies namely

For Widget Development:
Tavisca.Templar.Contract.dll, Tavisca.Frameworks.Spectrum.dll, Tavisca.Templar.Extensions.dll

Templar Framework:
Tavisca.Templar.dll, Tavisca.Templar.EmbeddedResourcePathProvider.dll.

Going forward upgrading for new features and fixes will become very easy.

New database mode support for Themes

There is no more a need for pushing theme content to every deployment machine. With this feature you can store themes in a database instead of the file system with full separation between design and live mode, which will be single point of deployment for multi-server deployment scenarios. Migrating from File System Mode to Database Mode is also very easy.

Note: Database mode doesn't work in backward compatibility mode. So you can't work with file system mode side by side with database mode.
Database Theme support requires Distributed Cache Solution e.g. AppFabric caching for performance reasons.

New improved file manager

New improved web based file manager for viewing theme content, Web File Manager works similar to file manager on your computer. Supports manipulations with existing files/folders can be done through the context menu or even with shortcuts. New File manager supports listing of files & folder, editing of text content based files, preview of files, creating zip files, uploading & downloading of files, easy moving or copying of files across folders.

Note: File editing is being disabled in case of Backward Compatibility mode i.e. when Themes are store on the file system.

Site level resources

Site level content can now be stored & accessed depending upon the access control. Content that varies with site such as xml configuration files, documents like FAQ's, Privacy policy etc… can now be stored with the site.

External JavaScript support

In Templar v1.7 you can add external JavaScript both Site & Page Level; every page level JavaScript takes precedence over site level. You can arrange order for JavaScript. Default JQuery can be overridden from external path, but order for jQuery & Key Name can't be changed. It has to be the first one. If jQuery Path is empty Templar Framework will insert its own jQuery.
Site-Resources can also be added as scripts.

Enhanced versioning features

Templar v1.7 now comes with entire Site Versioning, Page versioning, Global Culture & Global Theme Versioning support. Using previous version data you can see difference between current data using diff manager and can even update the data.

Web Performance Optimization

Templar now has a better way of CSS combining & compression. Also for performance reasons Templar now adds Cache Header for all dynamic resources that it serves. To enable optimization change value of key Templar.EnableWebOptimization in AppSetting to "Y", to disable it change value to "N", by default it is "Y".

When Themes are Database & CSS files are modified from UI File manager in order to reflect changes, you are recommended to clear WebOptimizationCache to force to re-bundling of CSS.

Enhanced Site Templates

Site Templates are now enhanced and include site level culture, resources and themes. Entire Site can now be re-created just using the template without any dependency.

Note: Global cultures and themes are not part of the site template and have to be moved independently if used. Site template refers them by their name.

Composite Widget & Enhanced Content Placeholder Widget

Introduction of Composite widget allows placing other widgets inside it, which helps in creating complex layouts & placing widget side by side in same panels. Content Placeholder widget has been improved to have tied relation with child widgets contained in it. So when it is dragged or dropped to another location all children are also moved with it.

Basic collaboration support

Notifications on page listing and designer appear whenever the same page is accessed and modified by some other user simultaneously.

Support for page level chat on designer to send simple message.

Sync site from site or template locally or over the web

Sites can now be updated from template or any other site on same deployment or remote deployment over the web provided the user has required access.

Global resources allow separate download\upload of content.

We now allow global Themes and Globalization data to be downloaded separately. This can be used to update corresponding data on a different Templar deployment.

Support for widget client event mapping

Framework now support JavaScript event mapping along with legacy way of broadcasting of JavaScript event.

Framework support for distributed caching & UI for cache management

Support for Distributed Cache has been added. UI cache management includes separate site specific cache with ability to view content corresponding to entries; remove individual entries or clear entire site specific cache without affecting other site caches.

Improved testability and standardized site data access

Added TemplarContext as a standard way of accessing Environment Data, Session, Culture Resources, Culture list and SEO parameters(Header's,footers & metadata) .
Enhanced testability for services and widgets with ability to create mock TemplarContext.

Page designer enhancements

a. New drag drop implementation on designer using jQuery has made UI more interactive.
b. Added Widgets search option on designer page.
c. One click collapse/expand all widgets is now possible
d. Ability for displaying widget help on designer page
e. Settings display also allows modal full page view of settings
f. Designer full screen mode
g. Page action notification (small loader) in design page menu bar

Templar Context now available in - aspx and ashx page created in Templar.

We now allow Templar settings to be accesses in aspx and ashx pages also.

Support for new Handlers to perform pre-post actions

Multiple handlers can now be added at site level to perform pre & post actions in execution life cycle of page, something similar to modules.
E.g. Logging ambient data whenever required.

Templar deployment service

Allows site\pages\templates\global culture\global theme - download, update via a service
Can be used to create deployment automation infrastructure

Miscellaneous Templar Admin UI usability enhancements

a. Templar User passwords are now md5 hashed.
b. Consistency changes - grids\search\paging fonts etc.
c. Tab reorganization
d. Removed Sandbox
e. Added Report - Exception \ Cache \ Feedback \ Scheduled Pages View
f. Favicon support for themes
g. Simplified options
    I. Site settings display changed with addition of SEO and Site Resources
    II. Create site options reduced with default creation from template
    III. Page settings default options reduced with option for advanced settings
    IV. All list editing made popup based with wide space for editing.
h. Multi selection with delete allowed in template listing.
i. Global resource download\upload.
j. Multi user login and theme changes allowed.
k. Added comment support in content for globalization
l. Page theme opt out support - Enable Theming key in page data
m. Exception reports accessible via the Templar UI
n. Widget preview enhancements - support for theme and event testing
o. Feedback window on Templar Admin UI
p. Templar default live site markup is now XHTML 1.0 Strict compliant
q. Doc-type switching support is added to enable HTML 5 site\page development in Templar SEO related client code can be injected on every page of site
r. Templar Admin UI now allows enabling/disabling of User.
s. Updated jQuery version to v1.7.2

t. Send Email Option /link is now configuration driven
u. Minor Schema Changes added Constraint on Page Table for unique Page url within same site
v. Added delete artifact API in Templar Deployment Service
w. Added download API for site resource, theme and globalization data in Site Management Service.

Support for Custom Layouts

Templar now supports new type of layout called “Custom Layout” along with existing 5 legacy layouts. In case of custom layout you can design and integrate any HTML markup with placeholders as your new layout.

Custom Layout allows defining complete HMTL markup along with placeholders for Widgets & substitution of Templar variables. Custom Layout can be dynamically or statically set by Pre-execute handler by setting markup string or Site Resource Key in page setting named ‘PageLayout’. Set Page Setting Key PageLayout’s value to ‘SiteResourceUrl(<siteresourcekey>)’.

E.g. SiteResourceUrl(HomePageLayout).

If key is missing or value is empty then layout is picked from applied theme. It means Theme should have folder named ‘Layouts’ which will hold HTML layout for individual Templar pages & Site Level default layout.

Custom Layouts can be defined at 3 levels.

1. Site level default layout.
2. Master Page Level Layout.
3. Normal / Content Page Level Layout.

Layout of page is always picked in reverse order of hierarchy.

Consider following Scenario
Site with URL: A Theme: B
Master Page with relative URL: /pages/masterpages/master
Content Page with relative URL: /pages/contentpages/login
Normal Page with relative URL: /home

1. Normal Page is accessed.
   When home page is accessed then framework looks for layout file at location.
   ”/Uploads/A/Themes/B/Layouts/home.html”.

   If the file is missing, then framework looks for site default page layout.
   “/Uploads/A/Themes/B/Layouts/ DefaultLayout.html”.

   If this file is also missing framework doesn’t render page properly & exception is logged in background.

2. Content Page is accessed
   When login page is accessed then framework looks for layout file at location.
   ”/Uploads/A/Themes/B/Layouts/pages/contentpages/login.html”.

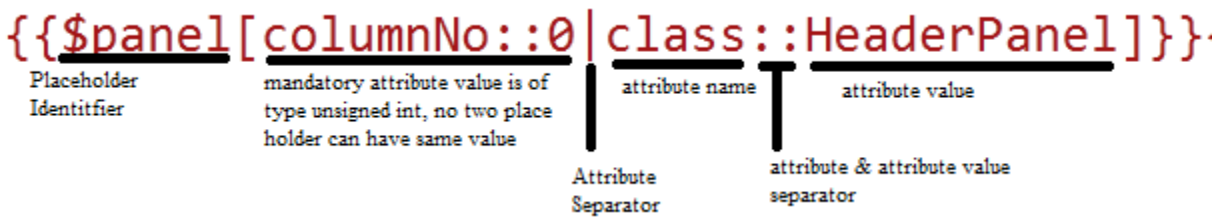
   If the file is missing, then framework looks for master page layout.
   ”/Uploads/A/Themes/B/Layouts/pages/masterpages/master.html”.

   If the file is missing, then framework looks for site default page layout.
   “/Uploads/A/Themes/B/Layouts/ DefaultLayout.html”.

   If this file is also missing framework doesn’t render page properly & exception is logged in background.

Layout Markup Substitution

1. Defining Placeholders for Widget - {{$panel[columnNo::0|attribute::attributeValue|...]}}



2. Resolving Templar Page Url - [$TemplarUrl(~<*templarpagerelativeurl*>)$].

3. Adding Images/Css/other files in them - [$ThemeUrl (<*filepath w.r.t. theme*>)$].

4. Adding Site Resource Url’s - [$SiteResourceUrl (<*siteresourcekey*>)$].

5. Adding Site Setting Value – [$SiteSettings(<*groupname*>,<*key*>)$].

6. Adding Page Setting Value – [$PageSetting(<*key*>)$].

7. Adding Globalization Resource Value –[#<*GlobalizationResourceKey*>#].

Sample Layout HTML File

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
      <title>Travel - Home</title>
      <meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
   </head>
<body>
   <div class="bodyCss">
```

```
      <div id="header">
         <a href="index.html" class="logo">
            <img src="[$ThemeUrl(images/logo.jpg)$]" alt="" height="90" width="274">
         </a>
         <ul id="menu">
            <li><a href="[$TemplarUrl(~/home)$]">Home Page</a></li>
            <li><a href="[$TemplarUrl(~/travelabout)$]">About Company</a></li>
         </ul>
      </div>
      {{$panel[columnNo::0|id::contentPanel]}}
      {{$panel[columnNo::1|id::footer]}}
   </div>
</body>
</html>
```

Serving Static HTML Pages

Static html as put in a layout can be served as-is without any markup changes on Templar Page's url. Custom Layouts set by pre-execute handler or from site resource or from Theme can directly serve as HTML page to browser by setting key named "OnlyRenderPageLayout" in PageSetting to "Y"

In this case apart from Placeholder substitution, all other substitution takes place, and final string content is sent back to browser.

Include Default Scripts to Page

Default behavior of Templar is to add all default scripts to the page. In case default scripts are not required set "DisableDefaultScripts" in PageSetting to "Y". Default value is "N"

Custom Authentication Provider Integration

Templar now supports integrating custom authentication provider. This feature enables plugging in third party authentication providers like Active Directory or custom authentication service for controlling access to Templar admin application.

To implement Custom Authentication Provider you need to implement following interfaces

Member Description

1. FirstName : First Name for Authenticated User
2. LastName: Last Name for Authenticated User
3. LoginName : Login Username of Authenticated User
4. RoleId : This allows changing the Role of the User

While he/she is logging in to Templar. For ex: If LDAP (Lightweight Directory Access Protocol) is used as an Authentication method, there can be cases where users' groups are changed. And in such case, it is desired that the Role also gets changed accordingly in Templar too. Hence, using this property cases like these can be handled.

Member Description

1. AllowUserRoleModification : Allow role modification or not is driven by a flag. Any of the roles can be assigned to a Templar user by using RoleId property of 'ITemplarUser' interface except 'SuperUser' role. When Templar authenticates any particular user, it checks whether user is trying to gain SuperUser role. This role can be acquired only by one user at a time, which gives all the rights. 'ValidateUser' method of UserAuthenticationProvider has been changed to accommodate all these conditions.
2. Authenticate : Authenticate user with given user name and password, return true & populate out parameter authenticatedUser, return false & set null to out parameter authenticatedUser.
3. CreateDisabledTemplarUserIfNotPresent: if true, authenticated user in created in Templar if the user was not previously registered but in disabled state. If false registers newly authenticated user as normal user.
4. Name: Name of Provider. This will be used in error message response on authentication failure.

Templar Default Application Configuration

Key Name	Purpose
Templar.IsFirstUse	Use to setup new development environment(Database) Default : ”Y”
Templar.Cache.CacheName	Name of the Templar Main Cache Default: “TemplarCache”
Templar.Cache.TemplarDesignCacheName	Name of the Templar Design Cache Default: “TemplarDesignCache”
Templar.Cache.TemplarLiveCacheName	Name of the Templar Live Cache Default: “TemplarLiveCache”
Templar.Cache.Provider	Name of class inheriting from abstract class System.Runtime.Caching.ObjectCache. Few Implementation are provided in Tavisca.Templar.Extensions 1. Wrapper around Dictionary Tavisca.Templar.Caching.InMemoryCacheImpl 2. Wrapper around HttpCache Tavisca.Templar.Caching.HttpRuntimeCacheImpl 3. Wrapper for AffFabric Tavisca.Templar.Caching.AppFabricCacheImpl Default: Tavisca.Templar.Caching.HttpRuntimeCacheImpl
Templar.Cache.EnableCacheMetaUpdate	Enable Cache Metadata update Default : “Y”
Templar.Cache.EnableAsyncCacheMetaUpdate	Cache access stats are updated asynchronously when value is Y else synchronous updation takes place Default : “Y”
Templar.EnablePortBasedSSLDetection	Templar allow port based SSL detection when this key has value “Y” else portbased detection is disabled. Default : “N”
Templar.SecurePorts	When port based SSL detection is enabled this key holds the comma separated port numbers on which secured page has to be served. Default : “443”
Templar.SaveViewStateInSession	Decreases size of page by storing viewstate in session when value is “Y”, else viewstate is persisted on page in Hidden Field Default: “Y”
Templar.AllowThumbnailsInFileManager	Show thumbnails in FileManager Default : “N”
Templar.AllowThemeEdit	Allow editing of css & text based resources in themes Default : ”N”
Templar.FileSystem	Theme Storage support is decided by this value, Theme can be stored either on WindowsFileSystem or in Database [WindowsFileSystem | DatabaseFileSystem]

	Default : “WindowsFileSystem”
Templar.EnableWebOptimization	Combining CSS for Preview & Live view from Themes Default : “Y”
Templar.ContentPlaceHolderWidgetName	Name of ContentPlaceHolder Widget Default : “ContentPlaceHolder”
Templar.CompositeWidgetName	Name of Composite Widget Default : “CompositeWidget”
Templar.ExcludeRoutes	File Extensions to be excluded from routing & let ASP.NET handle it Default: “htm|html|aspx|gif|jpg|png|axd|ico|svc|ashx|xml|txt|css|swf|js|php”
Templar.SuppressJavascriptErrors	Suppress JavaScript errors on Client Side & log it as Exception in Application Errors Default : “N”
Templar.EnableFeedback	Show feedback window in Admin Panel Default : “Y”
Templar.RestrictedSiteUrls	Names that are restricted for siteurl Default : “templar|resources”
Templar.EnableLogging	Log Key not found exception if not found in Globalization Culture Default: “N”
Templar.RestrictedPageUrls	Names that are restricted for pageurl Default : “design|preview|live|templar|services|handlers|templaradmin|defaultlayout”
Templar.ShowMachineName	On Live & Preview page bydefault machine name is visisble to locate response origin machine, if value is Y this information is available as tooltip at top left corner of browser Default: “N”
Templar.EnablePageScheduling	Enable page scheduling feature Default : “N”
Templar.PageSchedulerPollingInterval	Page Scheduler Polling Interval in Minutes Default : “5”
Templar.ShowSendEmailOption	Show mail sending option in Admin UI Default : “Y”
Templar.EnableCaching	On demand Caching of Site & Page Entities (i.e. Configurations) Default: “Y”
Templar.ApplicationVersion	This key suffix as querystring to all javascript url & css when included by framework, it is useful for for forcing static content update on client by changing its value.
Templar.TemplarContextLocatorService	Name of class implementing interface ITemplarContextLocatorService Value can be overridden to create MockContext for Testing Widgets locally
Templar.EnableAuditLogging	Enable/disable Templar audit logging feature Default : “Y”

Templar.FeedbackEncryptionKey	This key is used for decrypts the encrypted email info.
Templar.FeedbackSettings	This is used for decrypts the encrypted email info. Value is encrypted email-info.

Templar Default Page Level Configuration

Key Name	Purpose
DisableDefaultScripts	Use to exclude default scripts being added to page. Default : ”N”