ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

Search projects

Project Information

Project feeds

Code license
Apache License 2.0

Labels
application, ajax, web, orm, serialization, reflection, dependency, interpreter, Services, ffead, Messaging, Application, Server, injection, IOC

Members
sumeet.chhetri@gmail.com

Featured

Downloads
apache_mod_ffeadcpp-src-v1.0.zip
ffead-cpp-wiki.pdf
ffead-server-freebsd-src-v1.6.zip
ffead-server-freebsd-x86-bin-v1.6.zip
ffead-server-solaris-src-v1.6.zip
ffead-server-solaris-x64-bin-v1.6.zip
ffead-server-unix-src-v1.6.zip
ffead-server-unix-x64-bin-v1.6.zip
ffead-server-unix-x86-bin-v1.6.zip
ffead-server-winx-cygwin-bin-v1.6.zip
ffead-server-winx-cygwin-src-v1.6.zip
Show all »

Wiki pages
ExampleRestController
FreeBSDInstallationDirections
InstallationDirections
QuickStartGuide
WindowsCygwinInstallationDirections
Show all »

Links

Groups
General Discussion

Introduction

The framework is developed for rapid development of Enterprise application on the C++ platform. It is a c++ web framework, c++ application framework, c++ rest framework, c++ security framework and c++ soap framework all bundled into one. It consists of the following and is currently implemented for **LINUX/FREEBSD/WINDOWS(Cygwin)/SOLARIS**. It is the **first and only** C++ Application framework to provide **non-intrusive Dependency Injection and Business Driven Component Logic and POCO based Development**. Most of the features are controlled by configuration files.

Features

- Easy to use **View Framework**
- **SSL Support** available
- **Web Server** (Multi process, Multi Threaded – EPOLL based)
- Inbuilt Authentication handlers and **OAUTH** support
- Configuration driven URL mapping
- **Dependency Injection** (constructor and setter injection)
- **ORM** library (currently implemented for MySQL, Integrated with STL, Table mappings through configuration files –> One – Many, Many – Many ,One – One. The ORM Can be easily extended to other Databases).
- **SOAP Integration** (Web Service implementation through configuration file –> Methods in a C++ header file exposed as Web-Services)
- **REST** Controller framework (pretty URL's)
- **AJAX** Integration (Using property based configuration - On the lines of DWR for Java –> Just define C++ header files and Methods will be exposed as AJAX calls)
- **EJB styled Beans** (Remote and Local Interfaces exposed -> C++ files have the services, and the methods to be exposed are defined in a configuration file)
- **Universal Object type** for C++ (intelligent pointer - no need of extending any class – identifies the object type)
- XML based **Serialization** (Limited -> only for single level /no nesting of objects - header files required)
- **Reflection** (Limited –> header files required)
- **Dynamic C++ Pages** (Mix HTML and C++ code to produce run time views without web server restart)
- **Template Engine** and **Dynamic Views** generated from C++ objects.
- **Controller Pattern** (Implement controllers mapped with URL patterns to define custom behaviors)
- **Request/Response Filters** (Implement a chain of custom Filters for Pre/Post processing of request/response)
- **Thread Pool** Implementation
- **C++ Interpreter** (Limited support)
- **Rule based WEB** Behavior(idea can be expanded to other areas application wide)
- XML Parser (DOM Styled)
- Database Connection Pooling
- **Internationalization** support
- Utilities such as Timer, Logging, Property Files etc.
- File Upload
- FTP Server
- **ROLE based Security** features for Method access (Web Services, AJAX calls, C++ Bean service calls, Server URL’s)
- **Module** for Integration of the framework with **Apache Web Server**
- **Method Invoker** Server for cross platform Method Invocation (A daemon per language/platform approach – XML based Serialization/De- Serialization)

Also hosted at https://github.com/sumeetchhetri/ffead-cpp

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

Search projects

Search Current pages **for** Search

1 - 39 of 39

PageName ▼	Summary + Labels ▼	Changed ▼	ChangedBy ▼	...
ExampleRestController	An Example Rest Controller Implementation restcontroller implementation Featured	moments ago	sumeet.chhetri@gmail.com	
SolarisInstallationDirections	Installing ffead-server-solaris on Solaris	8 minutes ago	sumeet.chhetri@gmail.com	
FreeBSDInstallationDirections	Installing ffead-server-freebsd on FreeBSD Featured	9 minutes ago	sumeet.chhetri@gmail.com	
InstallationDirections	Installing ffead-server on GNU/Linux Featured	10 minutes ago	sumeet.chhetri@gmail.com	
WindowsCygwinInstallationDirections	Installing ffead-server on Windows and Cygwin Featured	11 minutes ago	sumeet.chhetri@gmail.com	
ExampleThreadPoolUsage	Example Thread Pool Usage cpp thread pool scheduled priority direct ffead	16 minutes ago	sumeet.chhetri@gmail.com	
TestSerialization	Example Serialization Usage cpp serialization support ffead	18 minutes ago	sumeet.chhetri@gmail.com	
ServerProperties	The properties for the Application Server. ffead cpp server property configuration	20 minutes ago	sumeet.chhetri@gmail.com	
NewWebApp	Create a new application in minutes	23 minutes ago	sumeet.chhetri@gmail.com	
ApplicationLevelConfig	Configure Applications ffead cpp application configuration controller template dynamic view dview filter	24 minutes ago	sumeet.chhetri@gmail.com	
AjaxConfig	Configuration for Ajax support ajax configuration cpp ffead	28 minutes ago	sumeet.chhetri@gmail.com	
QuickStartGuide	Create a new application in minutes Featured	29 minutes ago	sumeet.chhetri@gmail.com	
ExampleComponent	Example Component File ffead cpp example implementation component service business logic	Jul 2012	sumeet.chhetri@gmail.com	
ExampledcpFile	An Example .dcp file ffead cpp dcp dynamic page	Jul 2012	sumeet.chhetri@gmail.com	
ExampletpeFile	An Example .tpe file ffead cpp template example file	Jul 2012	sumeet.chhetri@gmail.com	
MessagingConfig	Configuration file for Messaging Support ffead cpp messaging configuration topic queue	Jul 2012	sumeet.chhetri@gmail.com	
CibernateConfig	Configuration for Cibernate (ORM) ffead cpp cibernate orm configuration hasmany hasone relation object table	Jul 2012	sumeet.chhetri@gmail.com	
LoggerProperties	Properties for Application level Logging ffead cpp logger configuration	Jul 2012	sumeet.chhetri@gmail.com	
FilterConfig	Configuration file for Filters cpp web filter configuration request response ffead	Jul 2012	sumeet.chhetri@gmail.com	
SecurityConfig	Configuration for Security cpp web security rolebased	Jul 2012	sumeet.chhetri@gmail.com	
ExampleWebService	An Example Web Service Implementation ffead cpp web service wsdl example implementation	Jun 2012	sumeet.chhetri@gmail.com	
ExampleController	An Example Controller Implementation ffead cpp example controller implementation	Jun 2012	sumeet.chhetri@gmail.com	
WebServicesConfig	Configuration for Web-Services ffead cpp example implementation web service wsdl	Jun 2012	sumeet.chhetri@gmail.com	
FViews	FFEAD Views	Jul 2011	sumeet.chhetri@gmail.com	
ExampleFilter	Implementation files for Content Filters ffead cpp filter request response pre post processing content example	Apr 2011	sumeet.chhetri@gmail.com	
TestReflection	Example Reflection Usage cpp reflection support ffead	Aug 2010	sumeet.chhetri@gmail.com	
DependencyInjection	Dependency Injection in FFEAD cpp dependency injection setter constructor interface ffead	Aug 2010	sumeet.chhetri@gmail.com	
SetterInjection	Implementation files for Setter Injection cpp setter injection ffead	Aug 2010	sumeet.chhetri@gmail.com	
ConstructorInjection	Implementation files for Constructor Injection cpp constructor injection ffead	Aug 2010	sumeet.chhetri@gmail.com	
InterfaceInjection	Implementation files for Interface Injection cpp interface injection ffead	Aug 2010	sumeet.chhetri@gmail.com	
ExampleTemplateImpl	An Example Template Implementation ffead cpp template example implementation	Aug 2010	sumeet.chhetri@gmail.com	
ExampleAJAXService	An Example AJAX Service Implementation ffead cpp ajax service example implementation object to javascript mapping	Aug 2010	sumeet.chhetri@gmail.com	
ExampleComponentServices	Example Component Service Implementations ffead cpp component service implementation example	Aug 2010	sumeet.chhetri@gmail.com	

TestComponent	Example Component Usage ffead cpp business component remote function call logic example	Aug 2010	sumeet.chhetri@gmail.com
TestCppInterpreter	Example Cpp Interpreter Usage ffead cpp interpreter eval	Aug 2010	sumeet.chhetri@gmail.com
TestCibernate	Example ORM Usage ffead cpp cibernate orm example object relational mapping	Aug 2010	sumeet.chhetri@gmail.com
ExampleDynamicViewImpl	An Example DynamicView Implementation ffead cpp dynamic view dview example implementation	Aug 2010	sumeet.chhetri@gmail.com
BootstrapDependencyInjection	Bootstrapping Dependency Injection ffead cpp dependency injection bootstrap example	Aug 2010	sumeet.chhetri@gmail.com
ExampleDBTablesAndObjects	An Example Implementation of DB Tables and Objects ffead cpp db tables object mappings cibernate orm	Aug 2010	sumeet.chhetri@gmail.com

1 - 39 of 39

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

InstallationDirections
Installing ffead-server on GNU/Linux
Featured, Phase-Deploy

Updated Today (10 minutes ago) by sumeet.chhetri@gmail.com

GNU/Linux based Installation

- Follow **Windows/Cygwin Install** for directions on how to install on Windows and Cygwin
- Download the tar file and extract it to the location of your choice.
- Install **Unix-ODBC** and required odbc files for **MySQL** dependency
- Install **Open-SSL** as well
- **JSON-SPIRIT** source and header files are already included in the source code distribution
- Also included **prototype.js** in source for AJAX support
- Go to the ffead-server/Release or ffead-server/Debug depending on whether you need to debug the server code
- Open terminal and type "make all" and "make build-apps" to build the server and the default applications provided
- This will create the distribution folder named ffead-server inside ffead-server/Release or ffead-server/Debug folders accordingly
- Type ./server.sh when inside the ffead-server folder to start the application server
- A default application is already provided for your reference inside the ffead-server/web folder, this application is served at urlpath /
- 2 other applications are provided, urlpath /flexApp and /oauthApp
- To compile the default application shared library go to the ffead-server/Release/ffead-server/web/default/src/Debug folder and run "make all"
- Copy the libdefault library to the ffead-server/Release/ffead-server/lib folder
- Place your application shared library inside the ffead-server/Release/ffead-server/lib or ffead-server/Debug/ffead-server/lib folder.
- Place your web application specific files inside the ffead-server/Release/ffead-server/web folder inside a folder with your application name

► Sign in to add a comment

FreeBSDInstallationDirections

Installing ffead-server-freebsd on FreeBSD

Featured

Updated Today (9 minutes ago) by sumeet.chhetri@gmail.com

FreeBSD based Installation

- Follow **Windows/Cygwin Install** for directions on how to install on Windows and Cygwin
- Download the tar file and extract it to the location of your choice.
- Install **Unix-ODBC** and required odbc files for **MySQL** dependency
- Install **Open-SSL** as well
- **JSON-SPIRIT** source and header files are already included in the source code distribution
- Also included **prototype.js** in source for AJAX support
- Go to the ffead-server-freebsd/Release or ffead-server-freebsd/Debug depending on whether you need to debug the server code
- Open terminal and type "gmake all" and "gmake build-apps" to build the server and the default applications provided
- This will create the distribution folder named ffead-server inside ffead-server/Release or ffead-server/Debug folders accordingly
- Type ./server.sh when inside the ffead-server folder to start the application server
- A default application is already provided for your reference inside the ffead-server/web folder, this application is served at urlpath /
- 2 other applications are provided, urlpath /flexApp and /oauthApp
- To compile the default application shared library go to the ffead-server/Release/ffead-server/web/default/src/Debug folder and run "gmake all"
- Copy the libdefault library to the ffead-server/Release/ffead-server/lib folder
- Place your application shared library inside the ffead-server/Release/ffead-server/lib or ffead-server/Debug/ffead-server/lib folder.
- Place your web application specific files inside the ffead-server/Release/ffead-server/web folder inside a folder with your application name
- Remember all installation files(makefile,subdir.mk) assume the ports are installed to the /usr/local/lib directory, in case you choose custom settings during installation of dependencies then make sure to change all files to include the -L/usr/local/lib to -L/location/of your/choice

► Sign in to add a comment

SolarisInstallationDirections
Installing ffead-server-solaris on Solaris

Updated Today (8 minutes ago) by sumeet.chhetri@gmail.com

Solaris based Installation

- Follow **Windows/Cygwin Install** for directions on how to install on Windows and Cygwin
- Download the tar file and extract it to the location of your choice.
- Install the following using the commands mentioned below

System Header files

```
sudo pkg install system/header
```

Install OpenCSW and edit PATH variable for CSW

```
sudo pkgadd -d http://get.opencsw.org/now vi ~/.profile
```

> - change the PATH to add /opt/csw/bin before the /usr/bin part - PATH should now look like /opt/csw/bin:/usr/bin:/usr/share/bin

Install GCC/UnixODBC and SSL devel and libraries

```
sudo pkgutil -y -i gcc4core gcc4g++ libstdc++6 unixodbc unixodbc_dev libssl1_0_0 libssl_dev subversion(optional)
```

- JSON-SPIRIT source and header files are already included in the source code distribution
- Also included prototype.js in source for AJAX support
- Go to the ffead-server-solaris/Release or ffead-server-solaris/Debug depending on whether you need to debug the server code
- Open terminal and type "gmake all" and "gmake build-apps" to build the server and the default applications provided
- This will create the distribution folder named ffead-server inside ffead-server/Release or ffead-server/Debug folders accordingly
- Type ./server.sh when inside the ffead-server folder to start the application server
- A default application is already provided for your reference inside the ffead-server/web folder, this application is served at urlpath /
- 2 other applications are provided, urlpath /flexApp and /oauthApp
- To compile the default application shared library go to the ffead-server/Release/ffead-server/web/default/src/Debug folder and run "gmake all"
- Copy the libdefault library to the ffead-server/Release/ffead-server/lib folder
- Place your application shared library inside the ffead-server/Release/ffead-server/lib or ffead-server/Debug/ffead-server/lib folder.
- Place your web application specific files inside the ffead-server/Release/ffead-server/web folder inside a folder with your application name

WindowsCygwinInstallationDirections

Installing ffead-server on Windows and Cygwin

Featured, Phase-Deploy

Updated Today (11 minutes ago) by sumeet.chhetri@gmail.com

Installation On Windows with Cygwin

- Download Cygwin setup.exe and follow instruction to install Cygwin, proceed with default settings.
- After Cygwin is successfully installed, you can open a cygwin terminal to check whether Cygwin works.
- Run the setup.exe to install further dependencies whenever required.
- Following are the dependencies,
  1. gcc
  2. g++
  3. autoconf
  4. automake
  5. boost libraries and devel
  6. openssl libraries and devel (use Cygwin_SSL_UnixOdbc_libs.zip from downloads section for version 1.0)
  7. unixODBC libraries and devel (not found in Cygwin repository, use Cygwin_SSL_UnixOdbc_libs.zip from download section)
  8. bison,yacc,flex,gdb,readline,binutils (for building and installing openssl and unixODBC)
- Copy bin/gcc-3 or bin/gcc-4 and rename it ot gcc (the already existing link file gcc sometimes gives permission errors)
- You can build and install openssl and unixodbc and copy the required library files to the cygwin lib folder. The already built libraries are present in the Cygwin_SSL_UnixOdbc_libs.zip file which can be copied to the lib folder directly.
- Download the latest release tar file and extract it to the location of your choice under Cygwin installation.
- Make sure **boost** libraries are installed on your system
- Install **Unix-ODBC** and required odbc files for **MySQL** dependency
- Install **Open-SSL** as well
- **JSON-SPIRIT** source and header files are already included in the source code distribution
- Also included **prototype.js** in source for AJAX support
- Go to the ffead-server/Release or ffead-server/Debug depending on whether you need to debug the server code
- Open terminal and type "make all" and "make build-apps" to build the server and the default applications provided
- This will create the distribution folder named ffead-server inside ffead-server/Release or ffead-server/Debug folders accordingly
- Type ./server.sh when inside the ffead-server folder to start the application server
- A default application is already provided for your reference inside the ffead-server/web folder, this application is served at urlpath /
- 2 other applications are provided, urlpath /flexApp and /oauthApp
- To compile the default application shared library go to the ffead-server/Release/ffead-server/web/default/src/Debug folder and run "make all"
- Copy the libdefault library to the ffead-server/Release/ffead-server/lib folder
- Place your application shared library inside the ffead-server/Release/ffead-server/lib or ffead-server/Debug/ffead-server/lib folder.
- Place your web application specific files inside the ffead-server/Release/ffead-server/web folder inside a folder with your application name

► Sign in to add a comment

QuickStartGuide

Create a new application in minutes

Featured, Phase-Implementation

Updated Today (29 minutes ago) by sumeet.chhetri@gmail.com

Quick Start Guide

BINARY FILE

- Download the ffead-server-{os}-bin-v{version}.zip file and extract it to the location of your choice.
- Visit the Installation Directions page for more information
- Go the folder ffead-server.
- Modify ffead-server/resources/server.prop to set the port, default application, number of server processes and other essential attributes Server-Properties
- Modify ffead-server/resources/log.properties file to enable custom logging for the applications Logger-Properties
- chmod 777 ffead-server/server.sh
- Execute ./server.sh start the Web Server
- Enter http://localhost:port/index.html and Watch the magic!!!

SOURCE FILE

- Download the ffead-server-v{version}.zip file and extract it to the location of your choice.
- Visit the Installation Directions page for more information
- Go the folder ffead-server/Debug or ffead-server/Release folder.
- chmod 777 makeAll.sh
- Execute the ./makeAll.sh file, it will compile and build the complete project and all required web applications
- Go to the ffead-server/Debug/ffead-server or ffead-server/Release/ffead-server folder
- Modify ffead-server/resources/server.prop to set the port, default application, number of server processes and other essential attributes Server-Properties
- Modify ffead-server/resources/log.properties file to enable custom logging for the applications Logger-Properties
- chmod 777 ffead-server/server.sh
- Execute ./server.sh start the Web Server
- Enter http://localhost:port/index.html and Watch the magic!!!

► Sign in to add a comment

ServerProperties

The properties for the Application Server.

ffead, cpp, server, property, configuration

Updated Today (20 minutes ago) by sumeet.chhetri@gmail.com

Server Properties

```
# The default Application name
DEF_PATH=default

#The Server listening port number
PORT_NO=8080

#The Component Invoker listeneing port number
CMP_PORT=

#The Messaging Daemon listening port number
MESS_PORT=

#The Method Invoker listening port number
MI_PORT=

#Enable SSL?
SSL_ENAB=false

#The mime types supported
SUPP_MIMES=*

#Is Streaming supported?
STREAM_ENAB=false

#Is Authorization required
AUTH_ENAB=false

#Number of Server processes
NUM_PROC=4

#Is AJAX available?
AJAX_ENAB=true

#Are Controllers enabled?
CONT_ENAB=true

#Support Dynamic C++ Pages
DCP_ENAB=true

#Is Template Engine enabled?
TPE_ENAB=true

#Are Dynamic Views enabled?
DVIW_ENAB=true

#The type of Threading strategy
THRD_PREQ=true

#The Thread Pool size for Pooled implementations
THRD_PSIZ=30

#The DB connection Pool Size
DB_CONN_POOL_SIZE=

#Where to store the session state Browser cookies/Server side file
SESS_STATE=server

#When to timeout the session
SESS_TIME_OUT=3600

#Compilers are generally disabled on production deployments
#Start server with DEV_MODE=true on Development servers
#and move the code to Production changing the flag DEV_MODE=false
#The code would not be generated at run-time on server restart now
DEV_MODE=true

#If you want to bind the server to a particulat IP
IP_ADDR=
```

► Sign in to add a comment

LoggerProperties

Properties for Application level Logging

ffead, cpp, logger, configuration

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Logger Properties

```
#The Logger Mode FILE/CONSOLE
MODE=FILE

#The Logger Level INFO/DEBUG/ERROR
LEVEL=INFO

#The File Path in case MODE is set to FILE
FILEPATH=/home/sumeet/server/server.log

#The Format of the Date in the Logger
DATEFMT=dd/mm/yyyy hh:mi:ss
```

► Sign in to add a comment

NewWebApp

Create a new application in minutes

Updated Today (23 minutes ago) by sumeet.chhetri@gmail.com

Add new web application

- Download the tar file and extract it to the location of your choice.
- Visit the Installation Directions page for more information
- Go the folder ffead-server-unix/Debug/ffead-server.
- Create a new folder in ffead-server/web i.e, appname
- Modify ffead-server/resources/ffead-server.prop to set the port, default application, number of server processes and other essential attributes Server-Properties
- Create ffead-server/resources/log.properties file to enable custom logging for the applications Logger-Properties
- Create new folders /lib /components /config /dcp /include under your newly created application folder (appname)
  - ffead-server/web/appname
    1. /lib
    2. /include
    3. /dcp
    4. /config
    5. /components
- Drop all your application level header files intended for Serialization and Reflection enabled support in the /appname/include folder. All files targeted for Web-Services, Ajax, Database mapping, Components, Controllers, Templates, Views etc should have their header definition files present in the /appname/include folder.
- Drop the shared library of your application inside the /appname/lib folder.
- Create/Drop your html/tpe files directly in the /appname folder Example-tpe-File
- Create/Drop your dcp files in the /appname/dcp folder Example-dcp-File
- Create /appname/config/cibernate.xml for ORM support Cibernate-Configuration
- Create /appname/config/application.xml for REST full services,Dynamic C++ pages,Templates and Dynamic view support Application-Level-Configuration
- Create /appname/config/afc.properties for Ajax Support Ajax-Configuration
- Create /appname/config/ws.xml for Web-Service Support Web-Services-Configuration
- For Business Entities or Business Driven Beans create your custom *.cmp files and drop them inside the /appname/components folder Example-Component
- For Dependency Injection create a file named /appname/config/depInj.xml Example depInj.xml
- Create /appname/config/messaging.xml for Messaging support Messaging-Configuration
- Create a new HTML page named index.html and place it in your /appname folder
- Restart the Web Server
- Enter http://localhost:port/appname/ and Watch the magic!!!

► Sign in to add a comment

ApplicationLevelConfig

Configure Applications

ffead, cpp, application, configuration, controller, template, dynamic, view, dview, filter

Updated Today (24 minutes ago) by sumeet.chhetri@gmail.com

Application Level Configuration

```
<app>
        <controllers>
                <controller class="DefaultController" url="*.action"/>
                <controller class="DefaultController" url="*.do"/>
                <controller from="*.yourext" to="*.html"/>
        </controllers>
        <authhandlers>
                <authhandler provider="file:users" url="*.authenticate"/>
                <authhandler provider="class:DefaultOAUTHController" url="*.oauth"/>
        </authhandlers>
        <templates>
                <template class="DefTemp" file="test.tpe"/>
        </templates>
        <dviews>
                <dview class="Dview" path="test.view"/>
        </dviews>
        <filters>
                <filter class="DefaultIOFilter" type="in"/>
                <filter class="DefaultIOFilter" type="out"/>
        </filters>
        <security>
                <login-handler provider="file:users" url="login.html"/>
                <welcome file="index.html"/>
                <!--login-handler provider="class:DefaultLoginHandler"/-->
                <!--login-handler provider="db:DefaultLoginHandler"/-->
                <secure path="*" role="ROLE_ANONYMOUS"/>
                <secure path="/rest/*" role="ROLE_USER"/>
        </security>
        <restcontrollers>
                <restcontroller class="DefaultRestController" urlpath="/rest/path/" name="rest1">
                        <restfunction name="addNumbers" alias="add/{1}/{2}" meth="GET">
                                <param type="int" name="1" from="path"/>
                                <param type="int" name="2" from="path"/>
                        </restfunction>
                </restcontroller>
                <restcontroller class="DefaultRestController" urlpath="/rest/reqparam/" name="rest1">
                        <restfunction name="addNumbers" alias="add" meth="GET">
                                <param type="int" name="1" from="reqparam"/>
                                <param type="int" name="2" from="reqparam"/>
                        </restfunction>
                </restcontroller>
                <restcontroller class="DefaultRestController" urlpath="/rest/postparam/" name="rest1">
                        <restfunction name="addNumbers" alias="add" meth="POST">
                                <param type="int" name="1" from="postparam"/>
                                <param type="int" name="2" from="postparam"/>
                        </restfunction>
                </restcontroller>
                <restcontroller class="DefaultRestController" urlpath="/rest/header/" name="rest1">
                        <restfunction name="addNumbers" alias="add" meth="GET">
                                <param type="int" name="1" from="header"/>
                                <param type="int" name="2" from="header"/>
                        </restfunction>
                </restcontroller>
                <restcontroller class="DefaultRestController" urlpath="/rest/path1/" name="rest2">
                        <restfunction name="addNumbers" alias="+/{1}/{2}" meth="GET">
                                <param type="int" name="1" from="path"/>
                                <param type="int" name="2" from="path"/>
                        </restfunction>
                </restcontroller>
                <restcontroller class="DefaultRestController" name="rest3">
                        <restfunction name="addNumbers" alias="ad/{1}/{2}" meth="GET">
                                <param type="int" name="1" from="path"/>
                                <param type="int" name="2" from="path"/>
                        </restfunction>
                </restcontroller>
                <restcontroller class="DefaultRestController">
                        <restfunction name="power" meth="GET" baseUrl="/rest/controller/base{1}/power/exp{2}">
                                <param type="int" name="1" from="path"/>
                                <param type="int" name="2" from="path"/>
                        </restfunction>
                </restcontroller>
                <restcontroller class="DefaultRestController">
                        <restfunction name="addNumbers" meth="GET" alias="addNumbers/{1}/{2}">
                <param type="int" name="1" from="path"/>
                <param type="int" name="2" from="path"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restvec">
            <restfunction name="testVector" alias="tstvec" meth="POST">
                <param type="vector&lt;int&gt;" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restvecobj">
            <restfunction name="testVectorObject" alias="tstvecobj" meth="POST" icontentType="application/json">
                <param type="vector&lt;TestMany&gt;" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restobj">
            <restfunction name="testObject" alias="tstobj" meth="POST" icontentType="application/json" ocontentType="application/json">
                <param type="TestMany" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restvecobj">
            <restfunction name="testVectorObject" alias="tstvecobj.xml" meth="POST" icontentType="application/xml">
                <param type="vector&lt;TestMany&gt;" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restobj">
            <restfunction name="testObject" alias="tstobj.xml" meth="POST" icontentType="application/xml">
                <param type="TestMany" from="body"/>
            </restfunction>
        </restcontroller>
    </restcontrollers>
</app>
```

► Sign in to add a comment

Terms - Privacy - Project Hosting Help

Powered by Google Project Hosting

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

SecurityConfig
Configuration for Security
cpp, web, security, rolebased

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Security Configuration

```
<!--The list of security parameters in the application-->
<app>
  <security>
        <!--Provides a handler for authenticating and authorizing access to users-->
        <login-handler provider="file:users" url="login.html"/>
        <!--login-handler provider="class:DefaultLoginHandler"/-->
        <!--login-handler provider="db:DefaultLoginHandler"/-->
        <!--A welcome page shown right after successful authentication/authorization-->
        <welcome file="index.html"/>
        <!--All pages in the application can be accessed anonymously-->
        <secure path="*" role="ROLE_ANONYMOUS"/>
        <!--The /rest/* path of the application can only be accessed by valid users of the application-->
        <secure path="/rest/*" role="ROLE_USER"/>
    </security>
</app>
```

FilterConfig

Configuration file for Filters

cpp, web, filter, configuration, request, response, ffead

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Filter Configuration

```
<!--The list of filters configured in the application-->
<app>
   <filters>
      <!--A filter configured for request level filtering and the first filter in the request chain
will server all request patterns-->
      <filter type="request" class="ExampleRequestFilter"/>
      <!--A filter configured for response level filtering and the first filter in the response chain
will server only *.htm-->
      <filter type="response" class="ExampleResponseFilter" url="*.htm"/>
   </filters>
</app>
```

► Sign in to add a comment

ExampleFilter

Implementation files for Content Filters

ffead, cpp, filter, request, response, pre, post, processing, content, example

Updated Apr 7, 2011 by sumeet.chhetri@gmail.com

```
ExampleRequestFilter.cpp
#include "Filter.h"

class ExampleRequestFilter : public Filter
{
   void doInputFilter(HttpRequest *request)
   {
      //Pre/Post Processing of request
      request.set....
   }
};


ExampleResponseFilter.cpp
#include "Filter.h"

class ExampleResponseFilter : public Filter
{
   void doOutputFilter(HttpResponse *response)
   {
      //Pre/Post Processing of response
      response.set....
   }
};


application.xml
<app>
      <filters>
            <filter class="ExampleRequestFilter" type="in"/>
            <filter class="ExampleResponseFilter" type="out"/>
      </filters>
</app>
```

► Sign in to add a comment

CibernateConfig

Configuration for Cibernate (ORM)

ffead, cpp, cibernate, orm, configuration, hasmany, hasone, relation, object, table

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Cibernate ORM Configuration

```
<cibernate>
    <config>
        <!--The DB user name-->
        <uid>DB user Name</uid>
        <!--The DB user password-->
        <pwd>DB user password</pwd>
        <!--The DB Data Source name-->
        <dsn>Data Source Name</dsn>
        <!--The DB connection pool size-->
        <pool-size>5</pool-size>
    </config>
    <tables>
        <!--The Table to Objcet Mapping-->
        <table name="test4" class="Test4">
            <col dbf="dat" obf="date"></col>
            <col dbf="datt" obf="datt"></col>
            <col dbf="datm" obf="dattm"></col>
        </table>
        <table name="person" class="Person">
            <hasMany fk="person_id" pk="id" field="interests" relClass="Person_Interests" relfk="interest_id" relpk="id">Interest</hasMany>
            <col dbf="id" obf="id"></col>
            <col dbf="name" obf="name"></col>
            <col dbf="age" obf="age"></col>
            <hasOne fk="life_id" pk="id" field="life" lazy="true">Life</hasOne>
        </table>
        <table name="interest" class="Interest">
            <hasMany fk="interest_id" pk="id" field="persons" relClass="Person_Interests" relfk="person_id" relpk="id">Person</hasMany>
            <col dbf="id" obf="id"></col>
            <col dbf="desc" obf="desc"></col>
            <col dbf="type" obf="type"></col>
        </table>
        <table name="person_interests" class="Person_Interests">
            <col dbf="person_id" obf="person_id"></col>
            <col dbf="interest_id" obf="interest_id"></col>
        </table>
        <table name="life clas="Life">
            <col dbf="life_id" obf="life_id"></col>
            <col dbf="desc" obf="desc"></col>
            <col dbf="type" obf="type"></col>
        </table>
    </tables>
</cibernate>
```

► Sign in to add a comment

ExampleDBTablesAndObjects

An Example Implementation of DB Tables and Objects

ffead, cpp, db, tables, object, mappings, cibernate, orm

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

Person.h

```
#ifndef PERSON_H_
#define PERSON_H_
#include "string"
using namespace std;

class Person {
        int id;
        int age;
        string name;
public:
        Person();
        virtual ~Person();
        int getId() const;
        void setAge(int);
        int getAge() const;
        void setId(int);
        string getName() const;
        void setName(string);
        bool operator<(Person t) const;
};

#endif /* PERSON_H_ */

create table person (id int,int age,name varchar(255));
```

Life.h

```
#ifndef LIFE_H_
#define LIFE_H_
#include "string"
using namespace std;

class Life {
        int id;
        strin type,desc;
public:
        Life();
        virtual ~Life();
        int getId() const;
        void setId(int id);
        string getType() const;
        void setType(string);
        string getDesc() const;
        void setDesc(string);
};

#endif /* LIFE_H_ */

create table life (id int,desc varchar(255),type varchar(50));
```

Interest.h

```
#ifndef INTEREST_H_
#define INTEREST_H_
#include "string"
using namespace std;

class Interest {
        int id;
        strin type,desc;
public:
        Interest();
        virtual ~Interest();
        int getId() const;
        void setId(int id);
        string getType() const;
        void setType(string);
        string getDesc() const;
        void setDesc(string);
};

#endif /* INTEREST_H_ */

create table interest (id int,desc varchar(255),type varchar(50));
```

PersonInterest.h

```
#ifndef PERSONINTEREST_H_
#define PERSONINTEREST_H_
#include "string"
using namespace std;

class PersonInterest {
        int person_id;
        int interest_id;
public:
        PersonInterest();
        virtual ~PersonInterest();
        int getPersonId() const;
        void setPersonId(int personId);
        int getInterestId() const;
        void setInterestId(int interestId);
};

#endif /* PERSONINTEREST_H_ */

create table person_interest(person_id int,interest_id int);
```

Test.h

```
#ifndef TEST_H_
#define TEST_H_
#include "string"
using namespace std;

class Test{
        int id;
        int age;
        string name;
public:
        Test();
        virtual ~Test();
        int getId() const;
        void setId(int);
        string getName() const;
        void setName(string);
        bool operator<(Test t) const;
};

#endif /* TEST_H_ */

create table test (id int,name varchar(255));



= {{{Test4.h}}} =
{{{
#ifndef TEST4_H_
#define TEST4_H_
#include "Date.h"
#include "BinaryData.h"

class Test4 {
        Date date;
        Date datt;
        Date dattm;
        BinaryData binar;
public:
        Test4();
        virtual ~Test4();
        Date getDate() const;
        void setDate(Date date);
        Date getDatt() const;
        void setDatt(Date datt);
        Date getDattm() const;
        void setDattm(Date dattm);
        BinaryData getBinar() const;
        void setBinar(BinaryData binar);
};
#endif /* TEST4_H_ */

create table test4 (date date,datt datetime,dattm timestamp,binar blob);
}}}
```

► Sign in to add a comment

TestCibernate

Example ORM Usage

ffead, cpp, cibernate, orm, example, object, relational, mapping

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

```
#include "Cibernate.h"
#include "Test.h"
#include "Timer.h"
#include "Object.h"

int main()
{
        Cibernate chib("MySQL-test","sumeet","sumeet");
        int i=2,j=4,k;
        Object oi;
        oi << i;
        Object oj;
        oj << j;
        Object ok;
        ok << k;
        chib.addParam("i","in",oi);
        chib.addParam("j","inout",oj);
        chib.addParam("k","out",ok);
        Timer tim;
        tim.start();
        chib.procedureCall("func1");
        vector<Test> tec = chib.getARAC<Test>();
        int id1=1;
        string name1 = "sumeet";
        oi << id1;
        oj << name1;
        chib.addParam("id",oi);
        chib.addParam("name",oj);
        tec = chib.getARACW<Test>();
        Test tec1 = chib.getOR<Test>(1);
        tec1 = chib.getOR<Test>(2);
        tec1 = chib.getOR<Test>(3);
        vector<int> tec2 = chib.getAROC<int>("id");
        vector<string> tec3 = chib.getAROC<string>("name");
        id1 = 4;
        name1 = "amit1233";
        oi << id1;
        oj << name1;
        chib.addParam1("id",oi);
        chib.addParam1("name",oj);
        int wid1 = 4;
        oi << wid1;

        Test tr;
        tr.setId(5);
        tr.setName("sumit");
        vector<string> cols;
        cols.push_back("id");cols.push_back("name");
        chib.insertORSC<Test>(tr,cols);
        tr.setId(6);
        chib.insertORAC<Test>(tr);
        tec.clear();
        Test *tp = new Test;
        tp->setId(7);
        tec.push_back(*tp);
        tp = new Test;
        tp->setId(8);
        tec.push_back(*tp);
        chib.bulkInsertRAC<Test>(tec);
        cols.erase(cols.begin()+1);
        string ns = "";
        Object on;
        on << ns;
        chib.addParam("name",on);
        //chib.getARSCW<Test>(cols);
        //chib.getARSC<Test>(cols);
        tp->setName("kriss");
        id1 = 8;
        oi << id1;
        chib.addParam("id",oi);
        //chib.updateRsAC<Test>(*tp);
        tim.end();
        cout << "\ndone" << flush;
        return 1;
}
```

► Sign in to add a comment

AjaxConfig

Configuration for Ajax support

ajax, configuration, cpp, ffead

Updated Today (28 minutes ago) by sumeet.chhetri@gmail.com

Ajax Configuration

```
#The Interfaces or Methods exposed
INTF=Expose
```

► Sign in to add a comment

ExampleAJAXService

An Example AJAX Service Implementation

ffead, cpp, ajax, service, example, implementation, object, to, javascript, mapping

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

ExampleAJAXService.h

```
#ifndef ExampleAJAXService_H_
#define ExampleAJAXService_H_
#include "PropFileReader.h"
#include "YObject.h"

class ExampleAJAXService{
public:
        ExampleAJAXService();
        virtual ~ExampleAJAXService();
        YObject sayHello(string,int,float);
        string sayHello1(string,int,float);
        YObject sayHello2(YObject,int,float);
};

#endif /* ExampleAJAXService_H_ */
```

ExampleAJAXService.cpp

```
ExampleAJAXService::ExampleAJAXService()
{}
ExampleAJAXService::~ExampleAJAXService()
{}
YObject ExampleAJAXService::sayHello(string j,int i,float c)
{
        YObject yobj;
        yobj.i = i;
        yobj.j = j;
        yobj.c = c;
        return yobj;
}
string ExampleAJAXService::sayHello1(string j,int i,float c)
{
        return "Hi There";
}
YObject ExampleAJAXService::sayHello2(YObject arg,int i,float j)
{
        YObject yobj;
        yobj = arg;
        return yobj;
}
```

► Sign in to add a comment

MessagingConfig

Configuration file for Messaging Support

ffead, cpp, messaging, configuration, topic, queue

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Messaging Configuration

```
<messaging>
   <service>
      <destination type="Queue" name="myQ"></destination>
      <url>localhost:8000</url>
   <service>
   <service>
      <destination type="Topic" name="myT"></destination>
      <url>localhost:8001</url>
   <service>
</messaging>
```

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

Search projects

Project Home Downloads **Wiki** Issues Source

Search Current pages for Search

WebServicesConfig

Configuration for Web-Services

ffead, cpp, example, implementation, web, service, wsdl

Updated Jun 23, 2012 by sumeet.chhetri@gmail.com

ws.xml

```
<web-services>
    <web-service name="testing" class="Testing">
        <test1 outname="result"/>
        <test2 outname="result"/>
        <test3 outname="result"/>
        <test4 outname="result"/>
    </web-service>
</web-services>
```

► Sign in to add a comment

ExampleWebService

An Example Web Service Implementation

ffead, cpp, web, service, wsdl, example, implementation

Updated Jun 30, 2012 by sumeet.chhetri@gmail.com

Testing.h

```
#ifndef Testing_H_
#define Testing_H_

class Testing {
public:
	Testing();
	virtual ~Testing();
	void test1(string);
	string test2();
	void test3(Test);
	Test test4(string);
};
#endif /* Testing_H_ */
```

Testing.cpp

```
#include "Testing.h"

Testing::Testing() {
	// TODO Auto-generated constructor stub

}

Testing::~Testing() {
	// TODO Auto-generated destructor stub
}

void Testing::test1(string in)
{
	cout << "in Webservice Req for test1 --\n" << in << flush;
}
string Testing::test2()
{
	cout << "in Webservice Req for test2 --\n" <<  flush;
	return "success";
}
void Testing::test3(Test t)
{
	cout << "in Webservice Req for test3 --\n" << t.getName() << flush;
}
Test Testing::test4(string in)
{
	Test g;
	g.setId(1);
	g.setName("Ffead-cpp");
	cout << "in Webservice Req for test4 --\n" << in << flush;
	return g;
}
```

Config for web-service in ws.xml

```
<web-services>
	<!--Custom implementation of a web-service, with methods-->
	<web-service name="testing" class="Testing">
		<test1 outname="result"/>
		<test2 outname="result"/>
		<test3 outname="result"/>
		<test4 outname="result"/>
	</web-service>
</web-services>
```

ExampleController

An Example Controller Implementation

ffead, cpp, example, controller, implementation

Updated Jun 30, 2012 by sumeet.chhetri@gmail.com

ExampleController.h

```
#ifndef EXAMPLECONTROLLER_H_
#define EXAMPLECONTROLLER_H_
#include <iostream>
#include "Controller.h"

class ExampleController: public Controller{
public:
	ExampleController();
	virtual ~ExampleController();
	HttpResponse service(HttpRequest);
};
#endif /* EXAMPLECONTROLLER_H_ */
```

ExampleController.cpp

```
ExampleController::ExampleController()
{}
ExampleController::~ExampleController()
{}
HttpResponse ExampleController::service(HttpRequest request)
{
	/*Play with the request*/
	HttpResponse res;
	/*Modify response*/
	return res;
}
```

Config for controller in application.xml

```
<controllers>
	<!--Custom controller handling url patterns-->
	<controller class="DefaultController" url="*.action"/>
	<controller class="DefaultController" url="*.do"/>

	<!--Internal controller handling extension conversions-->
	<controller from="*.yourext" to="*.html"/>
</controllers>
```

► Sign in to add a comment

ExampleRestController

An Example Rest Controller Implementation

restcontroller, implementation, Featured

Updated Today (moments ago) by sumeet.chhetri@gmail.com

DefaultRestController.h

```
#ifndef DEFAULTRESTCONTROLLER_H_
#define DEFAULTRESTCONTROLLER_H_

#include "RestController.h"
#include <math.h>
#include <iostream>
#include "vector"
#include "TestMany.h"

class DefaultRestController: public RestController {
public:
        DefaultRestController();
        virtual ~DefaultRestController();
        void addNumbers(int,int);
        void power(int,int);
        void testVector(vector<int>);
        void testObject(TestMany);
        void testVectorObject(vector<TestMany> param);
};

#endif /* DEFAULTRESTCONTROLLER_H_ */
```

DefaultRestController.cpp

```
#include "DefaultRestController.h"

DefaultRestController::DefaultRestController() {
        // TODO Auto-generated constructor stub

}

DefaultRestController::~DefaultRestController() {
        // TODO Auto-generated destructor stub
}

void DefaultRestController::addNumbers(int a, int b)
{
        int c = a + b;
        response->setHTTPResponseStatus(HTTPResponseStatus::Ok);
        response->setContent_type(ContentTypes::CONTENT_TYPE_TEXT_PLAIN);
        response->setContent_str(CastUtil::lexical_cast<string>(a) + " + " + CastUtil::lexical_cast<string>(b) + " = " +
                        CastUtil::lexical_cast<string>(c));
        cout << "Processed input request inside DefaultRestController..." << endl;
}

void DefaultRestController::power(int base, int exponent)
{
        int c = pow((double)base, (double)exponent);
        response->setHTTPResponseStatus(HTTPResponseStatus::Ok);
        response->setContent_type(ContentTypes::CONTENT_TYPE_TEXT_PLAIN);
        response->setContent_str(CastUtil::lexical_cast<string>(base) + " ^ " + CastUtil::lexical_cast<string>(exponent) + " = " +
                        CastUtil::lexical_cast<string>(c));
        cout << "Processed input request inside DefaultRestController..." << endl;
}

void DefaultRestController::testVector(vector<int> param)
{
        string temvec = "vector[";
        for (int var = 0; var < param.size(); ++var) {
                temvec += CastUtil::lexical_cast<string>(param.at(var));
                if(var!=param.size()-1)
                        temvec += ",";
        }
        temvec += "]";
        response->setHTTPResponseStatus(HTTPResponseStatus::Ok);
        response->setContent_type(ContentTypes::CONTENT_TYPE_TEXT_PLAIN);
        response->setContent_str(temvec);
        cout << "Processed input request inside DefaultRestController..." << endl;
}

void DefaultRestController::testObject(TestMany testMany)
```

```
{
    buildResponse(HTTPResponseStatus::Ok, "TestMany", &testMany);
    cout << "Processed input request inside DefaultRestController..." << endl;
}


void DefaultRestController::testVectorObject(vector<TestMany> param)
{
    buildResponseVector(HTTPResponseStatus::Ok, "TestMany", &param);
    cout << "Processed input request inside DefaultRestController..." << endl;
}
```

Config for rest service in application.xml

```
<restcontrollers>
        <restcontroller class="DefaultRestController" urlpath="/rest/path/" name="rest1">
            <restfunction name="addNumbers" alias="add/{1}/{2}" meth="GET">
                <param type="int" name="1" from="path"/>
                <param type="int" name="2" from="path"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" urlpath="/rest/reqparam/" name="rest1">
            <restfunction name="addNumbers" alias="add" meth="GET">
                <param type="int" name="1" from="reqparam"/>
                <param type="int" name="2" from="reqparam"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" urlpath="/rest/postparam/" name="rest1">
            <restfunction name="addNumbers" alias="add" meth="POST">
                <param type="int" name="1" from="postparam"/>
                <param type="int" name="2" from="postparam"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" urlpath="/rest/header/" name="rest1">
            <restfunction name="addNumbers" alias="add" meth="GET">
                <param type="int" name="1" from="header"/>
                <param type="int" name="2" from="header"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" urlpath="/rest/path1/" name="rest2">
            <restfunction name="addNumbers" alias="+/{1}/{2}" meth="GET">
                <param type="int" name="1" from="path"/>
                <param type="int" name="2" from="path"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="rest3">
            <restfunction name="addNumbers" alias="ad/{1}/{2}" meth="GET">
                <param type="int" name="1" from="path"/>
                <param type="int" name="2" from="path"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController">
            <restfunction name="power" meth="GET" baseUrl="/rest/controller/base{1}/power/exp{2}">
                <param type="int" name="1" from="path"/>
                <param type="int" name="2" from="path"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController">
            <restfunction name="addNumbers" meth="GET" alias="addNumbers/{1}/{2}">
                <param type="int" name="1" from="path"/>
                <param type="int" name="2" from="path"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restvec">
            <restfunction name="testVector" alias="tstvec" meth="POST">
                <param type="vector&lt;int&gt;" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restvecobj">
            <restfunction name="testVectorObject" alias="tstvecobj" meth="POST" icontentType="application/json">
                <param type="vector&lt;TestMany&gt;" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restobj">
            <restfunction name="testObject" alias="tstobj" meth="POST" icontentType="application/json">
                <param type="TestMany" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restvecobj">
            <restfunction name="testVectorObject" alias="tstvecobj.xml" meth="POST" icontentType="application/xml">
                <param type="vector&lt;TestMany&gt;" from="body"/>
            </restfunction>
        </restcontroller>
        <restcontroller class="DefaultRestController" name="restobj">
            <restfunction name="testObject" alias="tstobj.xml" meth="POST" icontentType="application/xml">
                <param type="TestMany" from="body"/>
            </restfunction>
        </restcontroller>
    </restcontrollers>
```

► Sign in to add a comment

FViews

FFEAD Views

Updated Jul 15, 2011 by sumeet.chhetri@gmail.com

FFEAD View or **FView** is a strategy where major view handling code is generated by the Server, all you need to do is just write plain HTML pages and define a corresponding **fviews.xml** file, it also provides easy javascript ajax methods for DOM events. Arguments to be sent to the server and callback can also be specified.

Every page in the XML file has a backing Page (**TestPage**) class for handling the DOM AJAX events.

You can define Form backing beans in the configuration file (fview.xml) and define your Bean (**TestForm**) class and a controller (**TestFormController**) responsible for handling the Submit action of the form. The Data from the HTML form is automagically converted to the bean class and fed to the controller onSubmit method.

The only point to be noted is that the URL in the action attribute in the form should end with **.form** extension which should also be the case in the xml config file. Also note the mapping of the form input parameter names to the **TestForm** class properties. All javascript functions can be written in the functions tag inside the page element.

HTML Page (test.html)

```
<html>
        <head>
        </head>
        <body>
                <p id="para">sdfsadsa</p>
                <input type="text" value="sdfsa" id="text"/>
                <a id="link" href="#">Testing</a>
                <form id="tform" action="/default/test.form" method="get" >
                        TextField: <input type="text" value="" name="txtField"/>
                        NumField: <input type="text" value="" name="numField"/>
                        ChooseField: <select name="selField"><option value="1">One</option><option value="2">Two</option></select>
                        <input type="submit" value="Submit"/>
                </form>
        </body>
</html>
```

Example fview.xml

```
<fview>
        <page htm="test.html" class="TestPage">
                <event eid="text" type="onclick" func="textonclick" args="1,document.getElementById('link').innerText,'Hello'" cb="alert(response.responseTe
                <event eid="link" type="onclick" func="linkonclick" cb="document.getElementById('para').innerHTML=response;alert(response.responseText)"/
                <functions>
                        <![CDATA[
                                function test()
                                {
                                        alert("Hello />");
                                }
                        ]]>
                </functions>
                <form name="test.form" bean="TestForm" controller="TestFormController">
                        <field name="txtField" prop="txt"/>
                        <field name="numField" prop="num"/>
                        <field name="selField" prop="che"/>
                </form>
        </page>
</fview>
```

TestPage

HEADER FILE

```
#ifndef TESTPAGE_H_
#define TESTPAGE_H_
#include "string"
using namespace std;
class TestPage {
public:
        TestPage();
        virtual ~TestPage();
        string textonclick(int,string,string);
        int linkonclick();
};

#endif /* TESTPAGE_H_ */
```

CPP FILE

```
#include "TestPage.h"
TestPage::TestPage() {
        // TODO Auto-generated constructor stub

}

TestPage::~TestPage() {
        // TODO Auto-generated destructor stub
}

string TestPage::textonclick(int a,string b,string c)
{
        return "Test Successfull" + b + c;
}

int TestPage::linkonclick()
{
        return 12345;
}
```

TestForm

HEADER FILE

```
#ifndef TESTFORM_H_
#define TESTFORM_H_
#include "string"
using namespace std;

class TestForm {
        int num;
        string txt;
        string che;
public:
        TestForm();
        virtual ~TestForm();
        string getChe() const;
    int getNum() const;
    string getTxt() const;
    void setChe(string che);
    void setNum(int num);
    void setTxt(string txt);
};

#endif /* TESTFORM_H_ */
```

CPP FILE

```
#include "TestForm.h"
TestForm::TestForm() {
        // TODO Auto-generated constructor stub

}

string TestForm::getChe() const
{
    return che;
}

int TestForm::getNum() const
{
    return num;
}

string TestForm::getTxt() const
{
    return txt;
}

void TestForm::setChe(string che)
{
    this->che = che;
}

void TestForm::setNum(int num)
{
    this->num = num;
}

void TestForm::setTxt(string txt)
{
    this->txt = txt;
}

TestForm::~TestForm() {
        // TODO Auto-generated destructor stub
}
```

TestFormController

HEADER FILE

```
#ifndef TESTFORMCONTROLLER_H_
#define TESTFORMCONTROLLER_H_
#include "HttpResponse.h"
#include "TestForm.h"
#include "iostream"

class TestFormController {
public:
	TestFormController();
	virtual ~TestFormController();
	void onSubmit(void*,HttpResponse*);
};

#endif /* TESTFORMCONTROLLER_H_ */
```

CPP FILE

```
#include "TestFormController.h"

TestFormController::TestFormController() {
	// TODO Auto-generated constructor stub

}

TestFormController::~TestFormController() {
	// TODO Auto-generated destructor stub
}

void TestFormController::onSubmit(void* vform,HttpResponse* res)
{
	TestForm* form = (TestForm*)vform;
	res->setStatusCode("200");
	res->setStatusMsg("OK");
	res->setContent_type("text/plain");
	res->setContent_str(form->getTxt()+form->getChe());
	cout << form->getTxt()+form->getChe() << "inside TestFormController" << endl;
}
```

► Sign in to add a comment

ExampletpeFile

An Example .tpe file

ffead, cpp, template, example, file

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Example Template file

```
<html>
  <head>
    <script src="someScript.js"></script>
  </head>
  <body>
    <h1>Hi ${user}!!</h1>
    Enter Your age: <input type="text"/>
    <input type="submit"/>
  </body>
</html>
```

► Sign in to add a comment

ExampleTemplateImpl

An Example Template Implementation

ffead, cpp, template, example, implementation

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

ExampleTemplate.h

```
#ifndef EXAMPLETEMPLATE_H_
#define EXAMPLETEMPLATE_H_
#include "TemplateHandler.h"

class ExampleTemplate: public TemplateHandler {
public:
        ExampleTemplate();
        virtual ~ExampleTemplate();
        Context getContext();
};
#endif /* EXAMPLETEMPLATE_H_ */
```

ExampleTemplate.cpp

```
ExampleTemplate::ExampleTemplate()
{}
ExampleTemplate::~ExampleTemplate()
{}
Context ExampleTemplate::getContext()
{
        Context cntxt;
        /*Add template variables to Context*/
        return cntxt;
}
```

► Sign in to add a comment

ExampledcpFile

An Example .dcp file

ffead, cpp, dcp, dynamic, page

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Example Dynamic C++ Page

```
<DCPH>
#include "string"
#include <iostream>
using namespace std;
</DCPH>
<html>
<head>
</head>
<body>
<input type="text"/>
lsdflsdkfjsdlflk
<DCPB>
string h;
h = "Hello World!!";
</DCPB>
<import>/home/sumeet/server/web/default/dcp/testheader.dcp</import>
<input type="submit"/>
<input type="text"/>

<DCPB>
//Goes to the server console
cout << h << flush;
</DCPB>

<DCPB>
for(int i=0;i<10;i++)
{
</DCPB>
<input type="text"/>
<DCPB>
}
</DCPB>
<DCPF>
void printscrn()
{
       //Goes to the server console
       cout << "Hello World from function!!" << flush;
}
</DCPF>

</body>
</html>
```

Comment by nawe...@gmail.com, Jan 6, 2012

Ciao

► Sign in to add a comment

ExampleComponent

Example Component File

ffead, cpp, example, implementation, component, service, business, logic

Updated Jul 30, 2012 by sumeet.chhetri@gmail.com

Example Component Configuration

```
#The Component Name, Should be unique
@CMP_NAME=TEST_BEAN

#The Component Description
@CMP_DESC=Test Bean

#Is The Component available Through Ajax
@AJAX_AVAIL=true

#Is the Component exposed as a Web-Service
@WEBS_AVAIL=true

#Can the Component be additionally invoked using MI
@MINV_AVAIL=true

#Is AUTO Trnsaction enabled?
@AUTO_TRANS=true

#The Threading Strategy
@THRD_PER_REQ=false

#The Thread Pool size
@THRD_POOL_NUM=20

#The Authorization source
@AUTH_FROM=database

#The Details of the Auth Source
@AUTH_DETS=@DB

#Is Authorization required for All Services?
@AUTH_ALL=false

#The User Groups that can access the Services
@USR_GRP_ALWD=UG_1,UG_2,UG_3

#The Users that are Blocked.
@BLOCK_USERS=user1,user2

#The Protocols allowed
@PROTO_ALWD=tcp,http,udp

#The DB connection Pool Size
@DB_CONN_POOL_NUM=10

#The Auth Connection Source
@AUTH_CONN_SRC=test_dsn

#The Auth Source User Name
@AUTH_USR_NAME=test

#The Auth Source User Password
@AUTH_USR_PASS=test

#The Auth Source Address
@AUTH_ADD=

#Are Sessions allowed
@SESSION=false

#The Service Details
#@NAME is the Service Name
#@USR_GRP_ALWD are the User Groups allowed to access the Component
#@SIGNATURE is the Service signature
#@ARGS are the arguments required for the Service
#SRV_RET is the Service Return type
@SERVICE1=@NAME(myFirstService) @USR_GRP_ALWD(UG_1,UG_2) @SIGNATURE(Service1.service1) @ARGS(void) @SRV_RET(string)
@SERVICE2=@NAME(mySecondService) @BLOCK_USERS(user7) @SIGNATURE(Service2.service2) @ARGS(string) @SRV_RET(string)
```

► Sign in to add a comment

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

ExampleComponentServices

Example Component Service Implementations

ffead, cpp, component, service, implementation, example

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

Service1.h

```
#ifndef SERVICE1_H_
#define SERVICE1_H_
#include "string"
#include "ServiceInt.h"
using namespace std;

class Service1 :public ServiceInt{
public:
	Service1();
	virtual ~Service1();
	string service1();
};

#endif /* SERVICE1_H_ */
```

Service2.h

```
#ifndef SERVICE2_H_
#define SERVICE2_H_
#include "string"
#include "ServiceInt.h"
using namespace std;
class Service2 : public ServiceInt{
public:
	Service2();
	virtual ~Service2();
	string service2(string);
};

#endif /* SERVICE2_H_ */
```

► Sign in to add a comment

TestComponent

Example Component Usage

ffead, cpp, business, component, remote, function, call, logic, example

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

TestComponent.cpp

```
/*
Client code can use a Remote Bean to invoke Business Logic
Local Applications/Modules can use the Local Bean instance
Trying to get Local Bean instance from client code will throw exception
*/
#include "BeanContext.h"
#include "Component_TEST_BEAN_Remote.h"
#include "Component_TEST_BEAN.h"

int main()
{
	/*Declare the BeanContext with the remote component listening host and port*/
	BeanContext cntxt("localhost",7001);

	/*Get the Remote Bean Instance*/
	Component_TEST_BEAN_Remote *remote = (Component_TEST_BEAN_Remote*)cntxt.lookup("TEST_BEAN");

	string a ="Hello Business Logic!!";
	if(remote!=NULL)
	{
		/*Invoke the remote method on the Bean*/
		cout << remote->mySecondService(a) << flush;
	}

	/*The Below line should throw exception*/
	Component_TEST_BEAN local;
	return 1;
}
```

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

ExampleDynamicViewImpl

An Example DynamicView Implementation

ffead, cpp, dynamic, view, dview, example, implementation

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

ExampleDynamicView.h

```
#ifndef EXAMPLEDynamicView_H_
#define EXAMPLEDynamicView_H_
#include "DynamicView.h"

class ExampleDynamicView: public DynamicViewHandler {
public:
        ExampleDynamicView();
        virtual ~ExampleDynamicView();
        Document getDocument();
};
#endif /* EXAMPLEDynamicView_H_ */
```

ExampleDynamicView.cpp

```
ExampleDynamicView::ExampleDynamicView()
{}
ExampleDynamicView::~ExampleDynamicView()
{}
Document ExampleDynamicView::getDocument()
{
        Document doc;
        /*Create a Document object*/
        return doc;
}
```

DependencyInjection

Dependency Injection in FFEAD

cpp, dependency, injection, setter, constructor, interface, ffead

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

depInj.xml

```
<beans>
   <!--Define a new Bean of type TestBeanProp and inject objects as properties
      Setter Injection
   -->
   <bean name="testBeanProp" class="TestBeanProp" injectAs="prop">
      <!--Inject bean with name dependencyBean1-->
      <inject bean="dependencyBean1">
      <!--Inject bean with name dependencyBean2-->
      <inject bean="dependencyBean2">
      <!--Inject bean with type DependencyBean3-->
      <inject name="dependencyBean3" class="DependencyBean3">
      <!--Inject a string with value-->
      <inject name="strProp" inbuilt="string" value="Hello ">
   </bean>
   <!--Define a new Bean of type DependencyBean1-->
   <bean name="dependencyBean1" class="DependencyBean1"/>
   <!--Define a new Bean of type DependencyBean2-->
   <bean name="dependencyBean2" class="DependencyBean2"/>


   <!--Define a new Bean of type TestBeanCons and inject objects as constructor args
      Constructor Injection
   -->
   <bean name="testBeanCons" class="TestBeanCons" injectAs="cons">
      <!--Inject an integer value-->
      <inject name="intProp" inbuilt="int" value="1234">
      <!--Inject a boolean value-->
      <inject name="boolProp" inbuilt="bool" value="true">
      <!--Inject bean with type DependencyBean4-->
      <inject name="dependencyBean4" class="DependencyBean4">
   </bean>


   <!--Define a new Bean of type TestBeanIntf and inject objects as compatible interfaces
      Interface Injection
   -->
   <bean name="testBeanIntf" class="TestBeanIntf" injectAs="intf">
      <!--Inject bean with interface type DepDependencyBean1-->
      <inject intfType="DepDependencyIntf1">
      <!--Inject bean with interface type DepDependencyBean2-->
      <inject intfType="DepDependencyIntf2">
   </bean>
   <!--Define a new Bean of type DepDependencyBean1Impl which implements DepDependencyIntf1 interface-->
   <bean name="dependencyIntfImpl1" class="DepDependencyBean1Impl"/>
   <!--Define a new Bean of type DepDependencyBean2Impl which implements DepDependencyIntf2 interface-->
   <bean name="dependencyIntfImpl2" class="DepDependencyBean2Impl"/>
</beans>
```

► Sign in to add a comment

SetterInjection

Implementation files for Setter Injection

cpp, setter, injection, ffead

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

```
class TestBeanProp
{
private:
   string *strProp;
   DependencyBean1 *dependencyBean1;
   DependencyBean2 *dependencyBean2;
   DependencyBean3 *dependencyBean3;
public:
   void setStrProp(string *strProp)
   {
      this->strProp = strProp;
   }
   string* getStrProp()
   {
      return this->strProp;
   }

   void setDependencyBean1(DependencyBean1 *dependencyBean1)
   {
      this->dependencyBean1 = dependencyBean1;
   }
   DependencyBean1* getDependencyBean1()
   {
      return this->dependencyBean1;
   }

   void setDependencyBean2(DependencyBean2 *dependencyBean2)
   {
      this->dependencyBean2 = dependencyBean2;
   }
   DependencyBean2* getDependencyBean2()
   {
      return this->dependencyBean2;
   }

   void setDependencyBean3(DependencyBean3 *dependencyBean3)
   {
      this->dependencyBean3 = dependencyBean3;
   }
   DependencyBean3* getDependencyBean3()
   {
      return this->dependencyBean3;
   }

   void print()
   {
      cout << *(this->strProp) << fflush;
      this->dependencyBean1->print();
      this->dependencyBean2->print();
      this->dependencyBean3->print();
   }
};

class DependencyBean1
{
public:
   void print()
   {
      cout << "Wo" << fflush;
   };
};

class DependencyBean2
{
public:
   void print()
   {
      cout << "rl" << fflush;
   };
};

class DependencyBean3
{
public:
   void print()
```

```
  {
    cout << "d!!" << fflush;
  };
};
```

► Sign in to add a comment

ConstructorInjection

Implementation files for Constructor Injection

cpp, constructor, injection, ffead

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

```
class TestBeanCons
{
private:
    int *intProp;
    bool *boolProp;
    DependencyBean4 *dependencyBean4;
public:
    TestBeanCons(int *intProp,bool *boolProp,DependencyBean4 *dependencyBean4)
    {
        this->intProp = intProp;
        this->boolProp = boolProp;
        this->dependencyBean4 = dependencyBean4;
    }

    int* getIntProp()
    {
        return this->intProp;
    }

    bool* getBoolProp()
    {
        return this->boolProp;
    }

    DependencyBean4* getDependencyBean4()
    {
        return this->dependencyBean4;
    }

    void print()
    {
        if(*(this->boolProp))
        {
            this->dependencyBean4->print();
            cout << *(this->intProp) << fflush;
        }
    }
};

class DependencyBean4
{
public:
    void print()
    {
        cout << "Hello World " << fflush;
    }
};
```

► Sign in to add a comment

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

Search projects

InterfaceInjection

Implementation files for Interface Injection

cpp, interface, injection, ffead

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

```
class TestBeanIntf
{
private:
   DependencyIntf1 *dependencyIntf1;
   DependencyIntf2 *dependencyIntf2;
public:
   void DependencyIntf1(DependencyIntf1 *dependencyIntf1)
   {
      this->dependencyIntf1 = dependencyIntf1;
   }
   DependencyIntf1* getDependencyIntf1()
   {
      return this->dependencyIntf1;
   }

   void DependencyIntf2(DependencyIntf2 *dependencyIntf2)
   {
      this->dependencyIntf2 = dependencyIntf2;
   }
   DependencyIntf2* getDependencyIntf2()
   {
      return this->dependencyIntf2;
   }

   public void print()
   {
      this->getDependencyIntf1()->print1();
      this->getDependencyIntf2()->print2();
   }
};

class DependencyIntf1
{
public:
   virtual void print1()=0;
};

class DependencyIntf2
{
public:
   virtual void print2()=0;
};

class DepDependencyBean1Impl : public DependencyIntf2
{
public:
   void print1()
   {
      cout <<"Hello " << fflush;
   }
};

class DepDependencyBean2Impl : public DependencyIntf2
{
public:
   void print2()
   {
      cout "World!!" << fflush;
   }
};
```

► Sign in to add a comment

BootstrapDependencyInjection

Bootstrapping Dependency Injection

ffead, cpp, dependency, injection, bootstrap, example

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

```
#include "FFEADContext.h"
#include "TestBeanProp.h"
#include "TestBeanCons.h"
#include "TestBeanIntf.h"

int main()
{
   FFEADContext *cntxt = new FFEADContext("/path/to/depInj.xml");

   //Get the testBeanProp from the container
   TestBeanProp *testBeanProp = (TestBeanProp*)cntxt->getBean("testBeanProp");
   testBeanProp->print();

   //Get the testBeancons from the container
   TestBeanCons *testBeanCons = (TestBeanCons*)cntxt->getBean("testBeanCons");
   testBeanCons->print();

   //Get the testBeanPIntf from the container
   TestBeanIntf *testBeanIntf = (TestBeanIntf*)cntxt->getBean("testBeanIntf");
   testBeanIntf->print();

   //Clean up resources
   cntxt->clear();
   return 1;
}

/*
The output of the following program would be
Hello World!!Hello World 1234Hello World!!
*/
```

► Sign in to add a comment

My favorites ▼ | Sign in

ffead-cpp

c++ framework, c++ web framework, c++ application framework, c++ security framework, c++ rest framework, c++ soap framework, Framework for Enterprise Application Development, c++ web sites,c++ web applications, c++ driven web development - c++

Search projects

Project Home Downloads **Wiki** Issues Source

Search Current pages for Search

ExampleThreadPoolUsage

Example Thread Pool Usage

cpp, thread, pool, scheduled, priority, direct, ffead

Updated Today (16 minutes ago) by sumeet.chhetri@gmail.com

ThreadPoolTest.cpp

```
using namespace std;
#include "ThreadPool.h"
#include "CastUtil.h"

class MyTask : public Task
{
        float j;
public:
        MyTask(float j){this->j = j;}
        ~MyTask(){}
        void run()
        {
                cout << "Task run " << j << "\n"<< flush;
        }
        string toString()
        {
                return ("Task No "+CastUtil::lexical_cast<string>(j));
        }
};

void testDirectExecution()
{
        /*Declare a Thread Pool with Min 2 and Max 5 Threads*/
        ThreadPool pool(2,5,true);
        /*Create a Task*/
        mytask task1(1);
        mytask task2(2);
        mytask task3(3);
        mytask task4(4);
        mytask task5(5);
        /*Execute The Task*/
        pool.execute(task1);
        pool.execute(task2);
        pool.execute(task3);
        pool.execute(task4);
        pool.execute(task5);
        /*Wait for completion of all Tasks*/
        pool.joinAll();
}

void testPrioritizedExecution()
{
        /*Declare a Thread Pool with Min 2 and Max 5 Threads, with Low 1 and 4 High Priority*/
        ThreadPool pool(2,5,1,4,true);
        /*Create a Task*/
        mytask task1(1);
        mytask task2(2);
        mytask task3(3);
        mytask task4(4);
        mytask task5(5);
        /*Execute The Tasks on priority*/
        pool.execute(task1,2);
        pool.execute(task2,4);
        pool.execute(task3,4);
        pool.execute(task4,1);
        pool.execute(task5,4);
        /*Wait for completion of all Tasks*/
        pool.joinAll();
}

void testScheduledExecution()
{
        /*Declare a Thread Pool with Min 2 and Max 5 Threads*/
        ThreadPool pool(2,5,true);
        /*Create a Task*/
        mytask task1(1);
        mytask task2(2);
        mytask task3(3);
        mytask task4(4);
        mytask task5(5);
        /*Schedule a task to be executed after the defined delay period*/
        pool.schedule(task1,10,TimeUnit::MILLISECONDS);
        pool.schedule(task2,10,TimeUnit::SECONDS);
        pool.schedule(task3,1,TimeUnit::HOURS);
        pool.schedule(task4,10,TimeUnit::DAYS);
        pool.schedule(task5,110,TimeUnit::MILLISECONDS);
        /*Wait for completion of all Tasks*/
        pool.joinAll();
}

void testDirectScheduledExecution()
{
        /*Declare a Thread Pool with Min 2 and Max 5 Threads*/
        ThreadPool pool(2,5,true);
        mytask task1(1);
        mytask task2(2);
        mytask task3(3);
        mytask task4(4);
        mytask task5(5);
        /*Schedule a task to be executed after the defined delay period*/
        pool.schedule(task1,10,TimeUnit::MILLISECONDS);
        pool.schedule(task2,10,TimeUnit::SECONDS);
        /*Execute the task*/
        pool.execute(task3);
        pool.schedule(task4,10,TimeUnit::DAYS);
        pool.execute(task5);
        /*Wait for completion of all Tasks*/
        pool.joinAll();
}


int main()
{
        /*Test the Direct Thread Pooling mechanism*/
        testDirectExecution();
        /*Test the Scheduled Thread Pooling mechanism*/
        testScheduledExecution();
        /*Test the Priority Driven Thread Pooling mechanism*/
        testPrioritizedExecution();
        /*Test the Mixed Thread Pooling mechanism*/
        testDirectScheduledExecution();
        return 0;
}
```

TestCppInterpreter

Example Cpp Interpreter Usage

ffead, cpp, interpreter, eval

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

TestCppInterpreter.cpp

```
#include "CppInterpreter.h"

int main()
{
	/*Create Interpreter Instance*/
	CppInterpreter cpi;

	/*Declare Local variables*/
	int a = 0,b=10;
	string b = "hello!!";

	/*Bind the desired variables to the Interpreter*/
	cpi.bind<int>("a",a);
	cpi.bind<int>("b",b);

	/*Evaluate the Code String*/
	cpi.eval("while(a<15){a+=3;if(b<50){b+=10;}}");
	cpi.eval("while(a<15){a+=3;}");
	cpi.eval("int y=2;string h=\"fsdfsdfsdfsdf\";for(a=20;a>0;a--){b+=10;y++;}");
	cpi.eval("if(a==2){a=8;}else if(b==11){a=90;}else while(a<50){a++;}");
	cpi.eval("a=10-2+8-6;");

	/*Display the Modified variables*/
	cout << a << flush;
	cout << "\n" << flush;
	cout << b << flush;
	cout << "\n" << flush;
	return 1;
}
```

TestReflection

Example Reflection Usage

cpp, reflection, support, ffead

Updated Aug 16, 2010 by sumeet.chhetri@gmail.com

```
#include "Reflector"
#include "Test.h"

int main()
{
	Reflector reflector;
	ClassInfo clas = reflector.getClassInfo("Test");
	args argus;
	argus.push_back("int");
	vals valus;
	int ids = 1;
	valus.push_back(&ids);
	Method meth = clas.getMethod("setId",argus);
	reflector.invokeMethod<void*>(clas.getInstance(),meth,valus);
	argus.clear();
	meth = clas.getMethod("getId",argus);
	int id = reflector.invokeMethod<int>(clas.getInstance(),meth,valus);
	cout << id << flush;cout << "\n" << flush;
	Field fld = clas.getField("id");
	void* idp = reflector.getField(clas.getInstance(),fld);
	Test *p = new Test;
	cout << p->getId() << flush;cout << "\n" << flush;
	cout << reflector.instanceOf(clas.getInstance(),"Test") << flush;
	bool fl;
	cout << "static::" << Object::instanceOf(*p,"Test") << "\n" << flush;
	cout << reflector.instanceOf(reflector,"Test") << flush;
	cout << reflector.instanceOf(clas,"Test") << flush;
}
```

► Sign in to add a comment

TestSerialization

Example Serialization Usage

cpp, serialization, support, ffead

Updated Today (18 minutes ago) by sumeet.chhetri@gmail.com

```
#include <iostream>
#include "string"
#include "TestMany.h"
#include "TestForm.h"
#include "TestSTLs.h"
#include "Serialize.h"
#include "XMLSerialize.h"
#include "JSONSerialize.h"

using namespace std;

int main() {

    TestMany tm;
    tm.t.setId(1);
    tm.t.setName("test");

    tm.y = 2;

    tm.vd.push_back(0.1);
    tm.vd.push_back(1.1);
    tm.vd.push_back(2.1);
    tm.vd.push_back(3.1);
    tm.vd.push_back(4.1);

    tm.vi.push_back(0);
    tm.vi.push_back(1);
    tm.vi.push_back(2);
    tm.vi.push_back(3);
    tm.vi.push_back(4);

    tm.vl.push_back(0);
    tm.vl.push_back(1);
    tm.vl.push_back(2);
    tm.vl.push_back(3);
    tm.vl.push_back(4);

    tm.vs.push_back("1");
    tm.vs.push_back("2");
    tm.vs.push_back("3");
    tm.vs.push_back("4");
    tm.vs.push_back("5");

    YObject yo;
    yo.i = 1;
    yo.j = "1";
    yo.c = 1.1;
    tm.vyo.push_back(yo);
    yo.i = 2;
    yo.j = "2";
    yo.c = 1.2;
    tm.vyo.push_back(yo);

    Serialize ser;
    string binstr = ser.serialize<TestMany>(tm);

    TestMany tmn = ser.unserialize<TestMany>(binstr);
    cout << tmn.vi.size()<<tmn.vl.size()<<tmn.vd.size()<<tmn.vs.size()<<tmn.vyo.size()<<tmn.t.getId()<<tmn.t.getName()<<tm.y << endl;

    binstr = ser.serializeUnknown(&tm, "TestMany");
    TestMany* tmp = (TestMany*)ser.unSerializeUnknown(binstr, "TestMany");
    tmn = *tmp;
    cout << tmn.vi.size()<<tmn.vl.size()<<tmn.vd.size()<<tmn.vs.size()<<tmn.vyo.size()<<tmn.t.getId()<<tmn.t.getName()<<tm.y << endl;

    XMLSerialize xser;
    binstr = xser.serialize<TestMany>(tm);

    tmn = xser.unserialize<TestMany>(binstr);
    cout << tmn.vi.size()<<tmn.vl.size()<<tmn.vd.size()<<tmn.vs.size()<<tmn.vyo.size()<<tmn.t.getId()<<tmn.t.getName()<<tm.y << endl;

    binstr = xser.serializeUnknown(&tm, "TestMany");
    tmp = (TestMany*)xser.unSerializeUnknown(binstr, "TestMany");
    tmn = *tmp;
    cout << tmn.vi.size()<<tmn.vl.size()<<tmn.vd.size()<<tmn.vs.size()<<tmn.vyo.size()<<tmn.t.getId()<<tmn.t.getName()<<tm.y << endl;
```

```
	JSONSerialize jser;
	binstr = jser.serialize<TestMany>(tm);

	tmn = jser.unserialize<TestMany>(binstr);
	cout << tmn.vi.size()<<tmn.vl.size()<<tmn.vd.size()<<tmn.vs.size()<<tmn.vyo.size()<<tmn.t.getId()<<tmn.t.getName()<<tm.y << endl;

	binstr = jser.serializeUnknown(&tm, "TestMany");
	tmp = (TestMany*)jser.unSerializeUnknown(binstr, "TestMany");
	tmn = *tmp;
	cout << tmn.vi.size()<<tmn.vl.size()<<tmn.vd.size()<<tmn.vs.size()<<tmn.vyo.size()<<tmn.t.getId()<<tmn.t.getName()<<tm.y << endl;


	TestSTLs stls;
	stls.vli.push_back(1);
	stls.vlsh.push_back(1);
	stls.vll.push_back(1);
	stls.vld.push_back(1.1);
	stls.vlb.push_back(true);
	stls.vls.push_back("1");
	stls.vlyo.push_back(yo);

	stls.vvi.push_back(1);
	stls.vvsh.push_back(1);
	stls.vvl.push_back(1);
	stls.vvd.push_back(1.1);
	stls.vvb.push_back(true);
	stls.vvs.push_back("1");
	stls.vvyo.push_back(yo);

	Test tst;
	tst.setId(1);
	tst.setName("1");

	stls.vsi.insert(1);
	stls.vssh.insert(1);
	stls.vsl.insert(1);
	stls.vsd.insert(1.1);
	stls.vss.insert("1");
	stls.vsyo.insert(tst);

	stls.vmsi.insert(1);
	stls.vmssh.insert(1);
	stls.vmsl.insert(1);
	stls.vmsd.insert(1.1);
	stls.vmss.insert("1");
	stls.vmsyo.insert(tst);

	stls.vdi.push_back(1);
	stls.vdsh.push_back(1);
	stls.vdl.push_back(1);
	stls.vdd.push_back(1.1);
	stls.vdb.push_back(true);
	stls.vds.push_back("1");
	stls.vdyo.push_back(yo);

	stls.vqi.push(1);
	stls.vqsh.push(1);
	stls.vql.push(1);
	stls.vqd.push(1.1);
	stls.vqb.push(true);
	stls.vqs.push("1");
	stls.vqyo.push(yo);

	stls.vpppli = new list<int>;
	stls.vpppli->push_back(1);

	binstr = ser.serialize<TestSTLs>(stls);
	cout << stls.toString() << endl;

	TestSTLs stlsn = ser.unserialize<TestSTLs>(binstr);
	cout << stlsn.toString() << endl;

	binstr = xser.serialize<TestSTLs>(stls);
	cout << stlsn.toString() << endl;

	stlsn = xser.unserialize<TestSTLs>(binstr);
	cout << stlsn.toString() << endl;

	binstr = jser.serialize<TestSTLs>(stls);
	cout << stlsn.toString() << endl;

	stlsn = jser.unserialize<TestSTLs>(binstr);
	cout << stlsn.toString() << endl;
}
```

► Sign in to add a comment