Technical Blog

Contents

[Reusable master template 1](#_Toc417856604)

[How this word document template is created 1](#_Toc417856605)

[How to learn java 2](#_Toc417856606)

[Installing java 2](#_Toc417856607)

[Helloworld program and Example to demonstrate logical thinking 2](#_Toc417856608)

[IDE 2](#_Toc417856609)

[Class , Method, variable , constructor , statements 2](#_Toc417856610)

[Datatypes & Variables 2](#_Toc417856611)

[Variable & Datatypes 2](#_Toc417856612)

[Constructor 3](#_Toc417856613)

[Unit testing - Terminology 3](#_Toc417856614)

[Velocity framework - Template 3](#_Toc417856615)

[Velocity Framework - Architecture 3](#_Toc417856616)

[Reusable master template 4](#_Toc417856617)

[Reusable master template 4](#_Toc417856618)

# Reusable master template

This is reusable master template that can be copied and reused for this entire document. All content in this document need to be as per this template.

# How this word document template is created

Reference >> Table of contents to generate table of contents.

Home >> small arrow under change styles to select headings which are used for table content generation.

<https://www.youtube.com/watch?v=awRxezHJpSM>

# How to learn java

# Installing java

# Helloworld program and Example to demonstrate logical thinking

# IDE

# Class , Method, variable , constructor , statements

# Datatypes & Variables

# Variable & Datatypes

**Resource**

@Path(“\shgh\sdjbb\”) – forward slash at beginning and end are optional @Path(“/user/{id}”) @Path(“/user/{id : regular exp}”)

**Interface**

@GET @POST @PUT @DELETE

**@produces** – based on Accept header

**@consumes** – based on content type header

Based on above four parameters relevant method is called

# Constructor

* Normal forms (Use and Discard) – **Explore**
* Design your database to split
* SQL query performance – **Explore**
* View / Materialized view – **Explore**
* On java side - **Explore**

# Unit testing - Terminology

Workers , Managers , SUT , DOC , Direct input , Direct output , Indirect input , Indirect output , state testing , interaction testing

# Velocity framework - Template

There are two symbols used in velocity template - $ #

$ - **Place holder** for data. Data gets pulled from velocity context object

# - **Directives** which have special purpose in template

#set($name=”anil”) - Put value in velocity context

#if( $value > 5 )

bigger

#elseif( $value < 5 )

smaller

#else

just right

#end

#foreach($item in $itemList)

#end

# Velocity Framework - Architecture

There are three components that work together to form velocity framework

* Velocity engine
* Template – File or string
* Velocity context – Holds data that need to be used by template

**Template as file:**

* Create velocity engine and call init method.
* Create velocity context object and put data in it.
* Call getTemplate method on velocity engine to get template
* Call merge method on template with velocity context and string writer

**Template as string:**

* Velocity.init
* Create velocity context
* Velocity.evaluate

[Tutorial](http://www.javaworld.com/article/2075966/core-java/start-up-the-velocity-template-engine.html)

[Velocity Examples](https://github.com/asakala/git/tree/master/Velocity)(https://github.com/asakala/git/tree/master/Velocity)

# Reusable master template

This is reusable master template that can be copied and reused for this entire document. All content in this document need to be as per this template.

# Reusable master template

This is reusable master template that can be copied and reused for this entire document. All content in this document need to be as per this template.