



# Jama Connect® Basic Training Exercises

## Table of Contents

|                                               |          |
|-----------------------------------------------|----------|
| <b>Overview .....</b>                         | <b>2</b> |
| <b>Required Homework Materials: .....</b>     | <b>2</b> |
| <b>Mockup Template Instructions .....</b>     | <b>2</b> |
| <b>Basic Training Homework Prompts.....</b>   | <b>3</b> |
| <b>Title Page Exercises .....</b>             | <b>3</b> |
| Exercise 1 – Title Page Structure.....        | 3        |
| Exercise 2 –Title Content .....               | 3        |
| Exercise 3 – Subtitle Content .....           | 3        |
| Exercise 4 – Report Run Author .....          | 3        |
| Exercise 5 – Embedded Logo Image .....        | 3        |
| <b>Content Page(s) Exercises .....</b>        | <b>4</b> |
| Exercise 1 – Container Types .....            | 4        |
| Exercise 2 – Text Items.....                  | 4        |
| Exercise 3 – Custom Item Type.....            | 4        |
| Exercise 4 – Unmapped Item Type Handling..... | 4        |
| <b>Item Type Macros Exercises .....</b>       | <b>5</b> |
| Exercise 1 – Format Containers Macro .....    | 5        |
| Exercise 2 – Format Text Macro .....          | 5        |
| Exercise 3 – Format Item Macro .....          | 5        |

# Overview

The purpose of these exercises is to allow you a chance to start building experience with Jama custom Velocity report development. This document will provide you with a collection of prompts that correspond to topics covered in the Part 1 slides / training video. Below you will find a list of the required homework materials that will be utilized to work through the exercises.

## Required Homework Materials:

1. Sublime Text Editor & provided Velocity library package or a text editor of your choice.
2. Basic\_Training\_Velocity\_Template.vm
3. Basic\_Training\_Mockup\_Example.docx
4. Basic\_Training\_Homework\_Output\_Example.docx
5. Jama Connect® instance
  - a. Note that to install your custom report under Admin > Organization > Reports to your Jama Connect® instance you will need to either user Admin permissions or a user with said permissions.

## Mockup Template Instructions

As outlined in the Part 1 materials, you will utilize the provided **Basic\_Training\_Mockup\_Example.docx** to extract embedded image definitions, style classes, and HTML document structure content to apply to your **Basic\_Training\_Velocity\_Template.vm** file. To begin that process, please follow the defined instructions below.

1. Open the **Basic\_Training\_Mockup\_Example.docx** file in Microsoft Word and perform a 'Save As' function setting the file format to .mht (Single File Webpage).
2. Open the created .mht file in your preferred text editor.
  - a. You may set your language library to Velocity (HTML) if using the Sublime editor and provided package to see syntax highlighting.
3. Perform a 'Find & Replace' actions with the Regex option enabled on the .mht following the defined replacement patterns outlined 'Tidy up the Velocity File' slide 12 from Part 1.
  - a. This will adjust the files readability for your exercise purposes and overall Velocity custom report development.
4. You will utilize this converted Mock up template file in the following Basic Training Velocity exercises defined below.

# Basic Training Homework Prompts

You will modify the provided **Basic\_Training\_Velocity\_Template.vm** for these code exercises. We recommend installing and running the report after each exercise to verify you changes step by step. You can utilize the provided **Basic\_Training\_Homework\_Output\_Example.docx** for comparison but note that it does not need to be an exact match.

*If you get stuck on any exercise, feel free to move to the following exercise as we will be reviewing the code solutions in following training video.*

## Title Page Exercises

For these exercises you will be working in the title page section denoted in your homework file with the label “## <ENTER TITLE PAGE INFORMATION HERE>”.

### Exercise 1 – Title Page Structure

Extract title page definition from .mht mockup file and place it into the proper section in your Velocity homework file denoted by the “## <ENTER TITLE PAGE INFORMATION HERE>” placeholder.

*Hint: it is recommended to clean away any excess or non-essential HTML / CSS content generated by Word to reduce code bulk and maintain code cleanliness for your report.*

### Exercise 2 –Title Content

Extract the current project name value and apply it as your document title page title.

*Hint: review slide ‘Jama Velocity API’ specifically regarding Velocity global variables.*

### Exercise 3 – Subtitle Content

When installing your report, apply a report criteria string parameter in the installation menu. When a user inputs a value into this parameter when running the report, this should appear as the title page subtitle.

*Hint: review the slide on ‘Report Parameters’.*

### Exercise 4 – Report Run Author

Using the Velocity API to extract the current report runner to inject their full name following the Author label.

*Hint: review slide on ‘Jama Velocity API’ specifically regarding Velocity API primary entry points known as ‘Sources’ (\$userSource).*

### Exercise 5 – Embedded Logo Image

Apply the provided Jama Connect® logo image as an embedded static image definition using the content found in your .mht mockup file. Ensure you modify the content properly for both the base64 image definition and the vml embedded image reference.

*Hint: review the slide on ‘Embedded Static Images’.*

## Content Page(s) Exercises

For these exercises you will be working in the output document list section with the label “## <ENTER ITEM FORMATTING HERE>”. This is where the primary #forloop that cycles through your exported items is used to create your output content.

### Exercise 1 – Container Types

For any exported item type of container Component [CMP], Set, [SET], or Folder [FLD], have the report output the item name in a heading 1 tag styling <h1>.

### Exercise 2 – Text Items

For any exported item type of Text [TXT] have the report output the item product id <documentKey> in static square '[' ]' braces followed by the item name in a heading 2 tag styling <h2>.

### Exercise 3 – Custom Item Type

Using an item type of your choice, have the report output the item name in heading 3 tag styling <h3>. Of which please follow it with a custom field of your choice that is not a rich text field (ex: <description>) where you provide a static field label followed by its value.

*Hint: review the slide on 'Fetching Generic Fields vs Custom Fields'.*

### Exercise 4 – Unmapped Item Type Handling

For any item type not mapped by a case defined above, define the following handling in the #else block definition. This is to ensure that every item sent to the report to run is still output in the generated file regardless of type.

In which you will output the items global id in static curly braces '{ }' followed by the item name. However, you should apply bold and italic formatting to only the item name content.

## Item Type Macros Exercises

For these exercises you will be creating your own item type handling macros with the label “<CUSTOM MACROS: INSERT/BUILD CUSTOM MACROS HERE>”. Here you will find pre-built sections to place each of your macros for the following exercises.

### Exercise 1 – Format Containers Macro

For your container item types, you will define a new item type handling macro called **#formatContainer()**.

This macro will hold your original previously defined handling for your container item types within the \$documentList #foreach loop cycle.

*Hint: ensure you properly provide the appropriate macro arguments needed for your item handling. If you need further guidance, you can review the ‘Velocity Content Macros’ slides.*

### Exercise 2 – Format Text Macro

For your text item types, you will define a new item type handling macro called **#formatText()**.

Like the previous exercise you will have this macro hold your original previously defined handling for your container item types within the \$documentList #foreach loop cycle.

### Exercise 3 – Format Item Macro

For your #else case for unmapped item types you will define a new item type handling macro called **#formatItem()**.

This macro will hold your original previously defined handling for your container item types within the \$documentList #foreach loop cycle. However, in addition following the original content you will also handle the items relationship items.

Create two static labels called ‘Downstream Items’ & ‘Upstream Items’ followed by a bullet list of the corresponding related items. Each related item should present the related items Product ID <documentKey> in static square ‘[ ]’ braces followed by the item name as a bullet list entry.