

TMM Development Setup

Prerequisites: Your computer needs to be set up per the Development Support: Getting Started page on pitawiki (Corpnet).

Checking out TMM

TMM can be found in our SVN server at <https://devslo112/repos/corp/programs/icode/tmm>

Usually you'll just want to check out the entire trunk into Eclipse. TMM is comprised of several module projects for the client, server, etc. If you have the trunk checked out, you can import projects as needed. It also makes it easier to update everything at once.

Setting up the Server

Import

In Eclipse, find the `tmm-webapp` folder in the `tmm` trunk project you checked out above. Right click on it and select "Import...". In the resulting dialog, choose "Existing Maven Projects" and then Next, Finish. This will create a new eclipse project for `tmm-webapp` and configure it automatically using information from the `maven pom.xml` file.

Note: *You can also check out the entire project if you need access to both the client and server code. Check out the main trunk of `tmm`. This will take a while. Then right click on the `tmm-webapp` folder inside this newly checked out project and select "Import." Select Maven-> Import as Maven project. After clicking finish, you will have two project- the main project that has everything, and a web-app project that will run on the server. In the rest of this document, everything is says to do to the `tmm-webapp` folder, do to the second project you built.*

Configure Tomcat

These instructions assume that you're using the Sysdeo Tomcat plugin for eclipse which is available in the devnet update site. The plugin requires some initial configuration which can be found on the Getting Started pitawiki page.

First you need to configure `tmm-webapp` as a tomcat project:

1. Open the project properties (right click the `tmm-webapp` folder and select properties) and go to the Tomcat properties page.
2. Check the "Is a Tomcat Project" checkbox.
3. Enter `tmm` for context name.
4. Enter `/target/tmm` for "Subdirectory to set as web application root"
5. Press OK to close the settings dialog.

You need to add another JVM parameter to tomcat to activate the correct spring profiles in TMM:

1. Open the workspace preferences dialog and go to Tomcat -> JVM Settings
2. Add a new JVM Parameter with the text
`-Dspring.profiles.active=dev,oracle`. If you're developing with postgres, you can change this to `dev,postgresql`.

Next, you need to build the webapp with Maven:

1. Right click the project and select Run as -> Maven Build... This will open a new run configuration window.
2. For "Goals" enter `clean install`.
3. Click Run. Maven will now build the project. This should take a few minutes.
4. In the future, you can reuse the same run configuration.

Now right click the project and select Tomcat Project -> Update Context Definition. This will create an xml file in your tomcat conf folder pointing to tmm-webapp. You can start tomcat with the toolbar button and tmm will start automatically.

Cassify

todo HowToCassify

Railcar Lib

In 6.0.5 the railcar libraries were moved to a separate jar file (`tmm-railcarlibs.jar`). The corresponding smaller version for dev is now here: [📄 tmm-railcarlibs-dev.jar](#).

Setup / layer files

Also in 6.0.5 the terminal definition / layer files were pulled out of the tmm-webapp source tree into there own separate project. This means a developer running Setup will not have a terminal to load unless:

- They build `deploy-motsu` (or anyother `deploy- project`) and point tomcat to that when they need to run Setup.
- They provide a `tmm-layers.jar` in `tomcat/lib` with the terminal they want to load.

It would be sweet if Setup accepted multiple layers jars, but doesn't currently if both jars try to define the same layer. It's a simple mod, but not done yet¹.

Setting up the Client


Using Flash Builder

Before importing the project:

1. Open Flash Builder preferences and go to General -> Workspace -> Linked Resources
2. Add a new Path Variable with the name `TMM_WEBAPP_DIR` pointing to the

target/tmm folder in your tmm-webapp project.

In Flash Builder, right click in Project/Package Explorer and select "Import...". In the dialog choose "Existing Maven Projects". Click Next and then browse to the location of the tmm/tmm-client folder that you checked out earlier. Finish the wizard to import the project. It will be automatically configured using information from the maven pom.xml file. The build directory will be set to TMM_WEBAPP_DIR.

Setting up run/debug: Click the arrow next to Run (or right click the project and select Run As) and select Run Configurations. Click the 'Web Application' icon with the Flex logo in it. click the 'New' icon at the top to create a new configuration. This will allow you to enter the configuration details. Under 'Project' make sure tmm-client is there (if not, browse to it). Un-check the "Use default" under "URL or path to launch" and change the URL to "  https://localhost:8443/tmm/index.html". Click "Apply". Now, when you select "Run" or "Debug", this configuration will be the default and should launch the web app.

Using the Flash Builder plug-in in Eclipse

Do the same thing as above, but do it in Eclipse.

1. 2013-5-9 (1)

DevelopmentSetup (last edited 2013-05-20 19:42:14 by jnetzley)