1 ProductionRun.java

1.1 Naming Conventions

1. All class names, interface names, method names, class variables, method variable, and constant used are meaningful.

2. The only one one-character variable in this ProductionRun.java file is
   GenericEntityException e
   And it is used as a parameter for the exception, for catching the statement. Therefore it is used in the loop, and it is temporary "throwaway" variable.

3. Class name of this ProductionRun.java file is:
   ProductionRun
   Therefore, it is with the first letter of each word in capitalized.

4. There is no ant defined interface within the ProductionRun.java file.

5. The methods of this file are as follow:
   exist()
   getGenericValue()
   store()
   getProductionProduced()
   getQuantity()
   setQuantity()
   getEstimatedStartDate()
   setEstimatedStartDate()
   getEstimatedCompletionDate()
   setEstimatedCompletionDate()
   recalculateEstimatedCompletionDate()
   getProductionRunName()
   setProductionRunName()
   getDescription()
   setDescription()
   getCurrentStatus()
   getProductionRunComponents()
   getProductionRunRoutingTasks()
   getLastProductionRunRoutingTask()
   clearRoutingTasksList()
   getEstimatedTaskTime()

isUpdateCompletionDate()
All the names of methods are verbs, and with the first letter of each addition work capitalized.

6. All of the class variables are as follow:
productionRun
productionRunProduct
productionProduced
quantity
estimatedStartDate
estimatedCompletionDate
productionRunName
description
currentStatus
productionRunRoutingTasks
dispatcher
There is no class variable with underscore, but all of the variables are lowercase first letter, and others with first letter capitalized.

7. There are two constant:
But "module" and "resource" should be modified to "MODULE"

```
public static final String module = ProductionRun.class.getName();
```

```
public static final String resource = "ManufacturingUiLabels";
```

and "RESOURCE".

1.2 Indention

For all indention, in the file ProductionRun.java, it adopts the convention of four space and done so consistently.
For indention, there is no tab used.

1.3 Braces

Bracing style adopted for entire class is Kernighan and Ritchie style.
For all body of all if-else,while,do-while,try-catch and for, the curly braces are used also for only one statement.

1.4 File Organization

1. In this file, for all of the sections, there is a blank line to separate from each others.

2. In this file, there is few line exceed 80 characters.

3. In this file, there is no line exceed 120 characters.

1.5 Wrapping Line

Every expressions in the ProducationRun.java file fit on a single line, so the convention is valid.

1.6 Comments

In this file, all of the comments are used to adequately explain what the class, interface, methods, and blocks of code are doing. What is more, there are also some comments in the method, in order to explain the detail.

1.7 Java Source Files

1. In this file, contains only one single public class.

2. In this file, this public class is this first class in the file.

3. There is no external program interface.

4. The javadoc is completed.

1.8 Package and Import Statements

The package statements are in the first non-comment statement.
And the Import statements follow with the package statements.

1.9 Class and Interface Declarations

The class declarations are in the correct order. And there is no interface. Which is:

1. class documentation comment.
2. class statement.
3. class (static) variable.
4. instance variable.
5. constructors.
6. methods.

The methods are grouped by the functionality.

2 ViewerServletRequest.java

2.1 Initialization and Declaration

1. All variables and class members are declared with correct type and the right visibility.
2. All variables are declared in the proper scope.
3. There is no call for the constructor.
4. There is no object references which is used.
5. All variables are initialized where they are declared.
6. All declarations appear at the beginning of block, some exceptions are declared after some instructions.

2.2 Method Calls

1. All of the parameters are presented in the correct order.
2. There is only one method: getParameter(String name), and it is been call correctly.
3. All method return type is correct

2.3 Arrays

There is no array used in this file.

2.4 Object Comparison

There are only two comparison in this ViewerServletRequest.java file, and they are used correctly.

2.5 Output Format

The class return always the desired output. The error message is managed in the classes of exception, so from this class we can not argue on the comprehensiveness.

2.6 Computation, Comparisons and Assignments

1. In this ViewerServletRequest.java file, we do not have long and complex arithmetic expressions.
2. All of the comparison and Boolean operators are correct.
3. For the throw-catch expression, the error condition is actually legitimate
4. The code does not contain any explicit and implicit type conversions.

2.7 time work

6h checklist