

# Evaluation Criteria for The Ricoh and Java™ Developer Challenge 2008/9 - Innovating Technology

First Name:  
 Last Name:  
 University:  
 Country:  
 Name Professor:

**Minimum requirements of the format in which to submit your application (in English):**

1. One-page summary (text and screen-shot) of the application in text format.
2. Presentation of the application from a marketing perspective in presentation format
3. Technical documentation document in text format. The following topics should be included: implicit code documentation (structure, variable names, etc), JavaDoc, programmers manual, installation instructions, product description.
4. Optional user/administration manual in text format

<b>MARKETING</b>	<b>50.00% Points (1 - 10) (10 is max.)</b>
<b>Product Concept</b>	<insert>
Simple ("The Elevator Pitch")	
Trendy	
Unique	
Hi-Tech	
Implementation	
Re-usability	
<b>Product Design</b>	<insert>
User Interface Design	
Work flow	
<b>Documentation</b>	<insert>
Quality of marketing documentation supplied	
<b>Presentation Skills (*if applicable)</b>	<insert>

<b>TECHNICAL</b>	<b>50.00% Points (1 - 10) (10 is max.)</b>
<b>Coding Skills</b>	<insert>
<b>Is the code clean and structured</b> - Hard coded values (magic numbers) or text messages should be avoided. This makes it easier to maintain the code and helps to localize the application. Did the developers use overwhelmingly large methods or did they do some refactoring to simplify the code?	
<b>Complexity of the application</b> - Is it difficult to create such an application or is it a simple task?	
<b>Object Oriented Programming</b> - Does the developer stick to the object oriented principles? Java by nature forces object oriented development especially when using GUI widgets. But what about the code of the business logic, does it follow the object oriented approach?	
<b>Design Patterns used</b> - Patterns help to get a more clear design by providing a general repeatable solution to commonly appearing problems.	
<b>Extra Stuff</b> - Such as demo- or simulation mode for applications that normally would require a more complex environment (which cannot be set up), default input material helps to use the application as is.	
<b>Technical Documentation</b>	<insert>
<b>UML</b> We would expect some documents explaining the model(s) created from the conceptual perspective: - Use case diagrams - Deployment/Component Diagrams showing - in case of distributed applications - which component(s) run(s) on client and which one(s) reside(s) on the server, together with their communication interfaces.  - In case of more complex interactions between such components, we would like to see some sequence diagrams explaining the timing of message transfer.	
<b>Object Oriented Design</b> - Also from the conceptual perspective some UML diagrams would help to see the object oriented approach by providing high level class diagrams.	
<b>Using technical terms</b>	
<b>Design Pattern used</b> - Design patterns are a finite catalogue of solutions for application development problems. Patterns describe a problem and suggest design proposals from the best practice. Such proposals are widely used and have their own names. This helps to understand the concept.	
<b>JavaDoc</b>	
<b>In-Line Comments</b> - Source code lines that are difficult for others to understand should be accompanied with comments. Alibi comments for simple code lines do not add any value. In those cases where complex code requires further textual explanation, in-line comments are a must.	

<b>Marketing Score</b>	
<b>Total Score</b>	

<b>Technical Score</b>	
------------------------	--