CONFIDENTIAL

PROJECT PERSONNEL ONLY

Sample Project

Jakarta Server Pages Coding Convention

Version 2.0

September 30, 2024

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| No. | Version | Date of change(s) | Category | Area(s) changed | | Details of change(s) | Personnel |
| Page | Item |
| 1 | 1.0 | 2018/8/24 | New | - | - | (First version created) | TIS |
| 2 | 2.0 | 2024/9/30 | Change | 3 | 1.5 | Describes the reading to Jakarta EE specification. | TIS Ichikawa |
| 7 | 3.1.1 | Change JSTL URI to Jakarta EE namespace. | TIS Ichikawa |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

（Cover）　 -

（Change History）　 -

（Contents）　 -

[1　Introduction 1](#_Toc169878733)

[1.1　Particulars of this document (purpose of coding convention, etc.) 1](#_Toc169878734)

[1.2　Role of development team members and rules for applying this convention 1](#_Toc169878735)

[1.3　Use of static analysis tools 1](#_Toc169878736)

[1.4　Format of rule boxes in chapter 2 onward 2](#_Toc169878737)

[1.5　Regarding the specification name of Jakarta EE in this Document 3](#_Toc169878738)

[2　Configuration rules 5](#_Toc169878739)

[2.1　Directory configuration 5](#_Toc169878740)

[2.1.1　Configure file directories so that the language and feature are clearly indicated 6](#_Toc169878741)

[3　Coding tips: Quality 7](#_Toc169878742)

[3.1　Escaping special characters 7](#_Toc169878743)

[3.1.1 Do not use tag libraries that cannot be used 7](#_Toc169878744)

[3.1.2 Do not use scriptlets 8](#_Toc169878745)

[3.1.3 Do not use EL expressions in areas other than tag libraries 8](#_Toc169878746)

[3.1.4 Do not use dynamic items in script areas 9](#_Toc169878747)

[3.2　Output of unnecessary information 10](#_Toc169878748)

[3.2.1 Do not use HTML comments 10](#_Toc169878749)

[3.3　Take maintenance into account during implementation 11](#_Toc169878750)

[3.3.1 Do not write constants with a control structure 11](#_Toc169878751)

[3.4　 Compliance with HTML standards 12](#_Toc169878752)

[3.4.1 Do not use elements that are not recommended for HTML4.01 12](#_Toc169878753)

[3.4.2 Write tags and attributes in lower case 13](#_Toc169878754)

[4　Coding tips: common practices 14](#_Toc169878755)

[4.1　Fundamentals of coding 14](#_Toc169878756)

[4.1.1 Always close HTML tags 14](#_Toc169878757)

[4.1.2 Indents should be 4 single-byte blank spaces 15](#_Toc169878758)

[5　Coding tips: Nablarch 16](#_Toc169878759)

[5.1　Specifying URIs 16](#_Toc169878760)

[5.1.1 Use a relative path from the context root when specifying URIs 16](#_Toc169878761)

|  |  |  |
| --- | --- | --- |
| 1　Introduction | |  |
|  | This document explains JSP coding convention to be used when developing screens using Nablarch’s web application framework. |  |

|  |  |  |
| --- | --- | --- |
| 1.1　Particulars of this document (purpose of coding convention, etc.) | |  |
|  | This document explains JSP coding convention that application programmers need to follow in all individual projects for developing applications using the Nablarch Application Framework. |  |

|  |  |  |
| --- | --- | --- |
| 1.2　Role of development team members and rules for applying this convention | |  |
|  | Application programmers need to implement JSP according to the JSP coding convention explained in this document. One of the conditions in inspections conducted at the time of delivery is that there must be no violations of this convention.  Consult an architect if a business requirement makes it impossible to comply with the convention. |  |

|  |  |  |
| --- | --- | --- |
| 1.3　Use of static analysis tools | |  |
|  | Use static analysis tools to complete a check of all of these points before requesting a code review.  Static analysis tools for Nablarch have been released together with this manual.  The tool indicated in the notes of each section should be used to check the points covered in that section. |  |

|  |  |  |
| --- | --- | --- |
| 1.4　Format of rule boxes in chapter 2 onward | |  |
| ◇    ◇  ◇      ◇ | **Heading** section (grayed out)  ・Rule code: Symbols used as identifiers for rules. Also used for linking with static analysis tools (provided separately).  ・Rule (summary): A somewhat brief summary. Refer to the description for details.  **Application** section (left pane, blank)  ・Example category: (Acceptable) if the code is correct, (Unacceptable) if a rule is violated.  ・Importance: “**IMPORTANT**” is used to indicate convention of particular importance (cases in which a bug will occur instantly if the convention is violated).  ・Other: Notes are included where necessary.  **Description** section (left pane, body text)  ・Description: Contains descriptions, points of caution, etc. for the rule.  ・Example (in box): Shows an example of code for the purposes of the rule. “\*” indicates a violation, “>” indicates relevant lines.  **Notes** section (right pane)  ・Related tools: JSP static analysis tool and/or HTML check tool  ・Supplementary information: Additional information on the point being explained. |  |

|  |  |  |
| --- | --- | --- |
| 1.5　Regarding the specification name of Jakarta EE in this Document | |  |
| ◇ | **。Unless otherwise specified, the acronyms used in this document refer to the Jakarta EE specifications as follows**   |  |  |  | | --- | --- | --- | | No. | Acronyms | Jakarta EE specifications | | 1 | JSF | Jakarta Faces | | 2 | JASPIC | Jakarta Authentication | | 3 | JACC | Jakarta Authorization | | 4 | JMS | Jakarta Messaging | | 5 | JPA | Jakarta Persistence | | 6 | JTA | Jakarta Transactions | | 7 | jBatch / JSR352 | Jakarta Batch | | 8 | JCA | Jakarta Connectors | | 9 | JAF | Jakarta Activation | | 10 | EL | Jakarta Expression Language | | 11 | EJB | Jakarta Enterprise Beans | | 12 | JAXB | Jakarta XML Binding | | 13 | JSON-B | Jakarta JSON Binding | | 14 | JSON-P | Jakarta JSON Processing | | 15 | JSP | Jakarta Server Pages | | 16 | JAX-WS | Jakarta XML Web Services | | 17 | JAX-RS | Jakarta RESTful Web Services | | 18 | JSTL | Jakarta Standard Tag Library | | 19 | CDI | Jakarta Contexts and Dependency Injection | |  |

|  |  |  |
| --- | --- | --- |
| **PQP-SNT-03 Do not use EL expressions in areas other than tag libraries　　　　　　　　Heading** section | |  |
| **Application** section | Do not use EL expressions in areas other than tag libraries.  (This may cause omission of escapes.)　　　　　　　　　　　　　　**Description** section | JSP static analysis tool  **Notes** section |
| (Unacceptable) | 01: ${user.userName} |  |
| (Acceptable) | 01: <c:out value="${user.userName}"/> |  |

|  |  |  |
| --- | --- | --- |
| 2　Configuration rules | |  |
|  | This section defines the directory configuration to be used for JSP files. |  |

|  |  |  |
| --- | --- | --- |
| 2.1　Directory configuration | |  |
|  | Decide on a directory configuration that enables easy handling of internationalization, features and JSP files. |  |

|  |  |  |
| --- | --- | --- |
| 2.1.1　Configure file directories so that the language and feature are clearly indicated | |  |
|  | Place JSP files directly inside the feature directory.  Place international JSP files inside the feature directory directly inside the language directory.  Directory configuration   |  | | --- | | [Document root]  └([Language directory (directory name: language name)])  └[Feature directory (directory name: name of feature)]  └JSP files | |  |
| (Acceptable) | International, language: ”ja”, name of feature：”project”   |  | | --- | | [Document root]  └ ja  └ project  └ create.jsp | |  |
| (Acceptable) | Not international, name of feature: ”project”   |  | | --- | | [Document root]  └ project  └ create.jsp | |  |

|  |  |  |
| --- | --- | --- |
| 3　Coding tips: Quality | |  |
|  | This section provides some tips to improve quality in JSP coding. |  |

|  |  |  |
| --- | --- | --- |
| 3.1　Escaping special characters | |  |
|  | Various libraries and JSP features can be used in JSP programs, but these may not perform escapes. This convention prohibits the implementation of any element that may cause omission of escapes. |  |

|  |  |  |
| --- | --- | --- |
| 3.1.1 Do not use tag libraries that cannot be used | |  |
| **IMPORTANT** | The only libraries that can be used are the tag library provided by Nablarch and the standard tag library (JSTL).  Usage of other tag libraries is prohibited as they may cause omission of escapes.  There is some duplication between the tag library provided by Nablarch and the standard tag library (JSTL). The tag library provided by Nablarch should generally be used in these cases.  Tag libraries that can be used   |  |  |  | | --- | --- | --- | | Prefix | URI | Description | | ｎ | http://tis.co.jp/nablarch | Tag library provided by Nablarch | | c | jakarta.tags.core | JSTL | | JSP static analysis tool |

|  |  |  |
| --- | --- | --- |
| 3.1.2 Do not use scriptlets | |  |
| **IMPORTANT** | Do not use scriptlets. Use the tag libraries instead.  　(This is because scriptlets may cause omission of escapes.) | JSP static analysis tool |
| (Unacceptable) | 01: <% String s = request.getAttribute("userName"); %> |  |
| (Acceptable) | 01: <n:write name="userName"/> |  |
| (Acceptable) | 01: <c:out value="${user.userName}"/> |  |

|  |  |  |
| --- | --- | --- |
| 3.1.3 Do not use EL expressions in areas other than tag libraries | |  |
| **IMPORTANT** | Do not use EL expressions in areas other than tag libraries.  (This may cause omission of escapes.) | JSP static analysis tool |
| (Unacceptable) | 01: ${user.userName} |  |
| (Acceptable) | 01: <c:out value="${user.userName}"/> |  |

|  |  |  |
| --- | --- | --- |
| 3.1.4 Do not use dynamic items in script areas | |  |
| **IMPORTANT** | Do not directly write dynamic items\*1 in script areas\*2.  (This may cause omission of escapes.)  If using dynamic items, output the tags in HTML and then reference the elements from the script.  \*1 EL expressions and n:write tags.  \*2 The body of script tags, event attributes (onclick, onmouseover, etc.), items starting with “javascript:” in attributes. | JSP static analysis tool |
| (Unacceptable) | 01: <n:text name="form.users[${status.id}]" onmouseover="popup(${status.id})" /> |  |
| (Acceptable) | 01: <%-- Acquire ${status.id} section from this in popup functions. --%>  02: <n:text id="text01\_${status.id}" name="form.users[${status.id}]" onmouseover="popup(this)" /> |  |

|  |  |  |
| --- | --- | --- |
| 3.2　Output of unnecessary information | |  |
|  | Implement code so that information that is not necessary for HTML is not output. |  |

|  |  |  |
| --- | --- | --- |
| 3.2.1 Do not use HTML comments | |  |
| **IMPORTANT** | Do not use HTML comments. (Use JSP comments.)  Otherwise, comments are output to HTML, which may cause information to be leaked. | JSP static analysis tool |
| (Unacceptable) | 01: <!—- Main content --> |  |
| (Acceptable) | 01: <%-- Main content --%> |  |

|  |  |  |
| --- | --- | --- |
| 3.3　Take maintenance into account during implementation | |  |
|  | Take maintenance into account when coding with JSP. |  |

|  |  |  |
| --- | --- | --- |
| 3.3.1 Do not write constants with a control structure | |  |
|  | In cases such as JSP code that controls whether to show or hide screen input items based on the user’s level, the user’s level needs to be checked using the judgment logic of control syntax.  Do not write the user level as a constant in that judgment logic, as it results in laborious maintenance work.  Implement and use a method to judge user level in objects where the user level is kept. |  |
| (Unacceptable) | \* 01: <c:if test="${userLevelCode == '01'}">  02: <n:text name="compName" size="65" maxlength="50" />  03: </c:if> |  |
| (Acceptable) | 01: <c:choose>  02: <c:when test="${userLevelCode.compUser}">  03: <td>  04: <n:text name="compName" size="65" maxlength="50" />  05: </td>  06: </c:when>  07: </c:choose> | It is expected that a judgment method (isCompUser) will be implemented in objects where the user level is kept. |

|  |  |  |
| --- | --- | --- |
| 3.4　 Compliance with HTML standards | |  |
|  | Use HTML that complies with standards. |  |

|  |  |  |
| --- | --- | --- |
| 3.4.1 Do not use elements that are not recommended for HTML4.01 | |  |
| **IMPORTANT** | Do not use tags or elements that are not recommended in HTML4.01.  Examples of tags that are not recommended  　<center>  <font>  Example of an element that is not recommended  <h1 align=”center”>  Refer to the links below for lists of tags and elements that are recommended and those that are not recommended.  Article: http://www.w3.org/TR/html401/index/elements.html  Article: http://www.w3.org/TR/html401/index/attributes.html | Requires customization.  Modify according to the version of HTML defined in the user interface standards.  HTML check tool |
| (Unacceptable) | \* 01: <center>  02: <table class="insertTbl">  03: <tr>  04: <th>User ID<n:forInputPage><span class="requiredMark">\*</span></n:forInputPage></th>  05: <td align="left">  06: <n:text name="systemAccount.userId" size="15" maxlength="10" />  07: <n:forInputPage>  08: (Maximum 10 single-byte alphanumeric characters)  09: <br />  10: <n:error name="systemAccount.userId" />  11: </n:forInputPage>  12: </td>  13: </tr>  14: </table>  \* 15: </center> |  |

|  |  |  |
| --- | --- | --- |
| 3.4.2 Write tags and attributes in lower case | |  |
|  | All tags and attributes in HTML must be written in lower case. |  |
| (Acceptable) | 01: <table class="insertTbl">  02: <tr>  03: <th>User ID<n:forInputPage><span class="requiredMark">\*</span></n:forInputPage></th>  04: <td align="left">  05: <n:text name="systemAccount.userId" size="15" maxlength="10" />  06: <n:forInputPage>  07: (Maximum 10 single-byte alphanumeric characters)  08: <br />  09: <n:error name="systemAccount.userId" />  10: </n:forInputPage>  11: </td>  12: </tr>  13: </table> |  |
| (Unacceptable) | 01: <TABLE CLASS="insertTbl">  02: <TR>  03: <TH>User ID<n:forInputPage><SPAN CLASS="requiredMark">\*</SPAN></n:forInputPage></TH>  04: <TD align="left">  05: <n:text name="systemAccount.userId" size="15" maxlength="10" />  06: <n:forInputPage>  07: (Maximum 10 single-byte alphanumeric characters)  08: <br />  09: <n:error name="systemAccount.userId" />  10: </n:forInputPage>  11: </TD>  12: </TR>  13: </TABLE> |  |

|  |  |  |
| --- | --- | --- |
| 4　Coding tips: common practices | |  |
|  | This chapter explains various common practices that programmers should be aware of but which team members who are inexperienced in Java programming may not know. |  |

|  |  |  |
| --- | --- | --- |
| 4.1　Fundamentals of coding | |  |
|  | This section explains fundamental practices in coding. |  |

|  |  |  |
| --- | --- | --- |
| 4.1.1 Always close HTML tags | |  |
| **IMPORTANT** | Forgetting to close tags not only decreases readability, it may cause elements not to be displayed properly in some browsers.  To ensure quality, be sure to close tags. | HTML check tool |
| (Unacceptable) | 01: <tr>  02: <th>User ID<n:forInputPage><span class="requiredMark">\*</span></n:forInputPage></th>  03: <td align="left">  04: <n:text name="systemAccount.userId" size="15" maxlength="10" />  05: <n:forInputPage>  06: (Maximum 10 single-byte alphanumeric characters)  07: <br />  08: <n:error name="systemAccount.userId" />  09: </n:forInputPage> |  |
| (Acceptable) | 01: <tr>  02: <th>User ID<n:forInputPage><span class="requiredMark">\*</span></n:forInputPage></th>  03: <td align="left">  04: <n:text name="systemAccount.userId" size="15" maxlength="10" />  05: <n:forInputPage>  06: (Maximum 10 single-byte alphanumeric characters)  07: <br />  08: <n:error name="systemAccount.userId" />  09: </n:forInputPage>  10: </td>  11: </tr> |  |

|  |  |  |
| --- | --- | --- |
| 4.1.2 Indents should be 4 single-byte blank spaces | |  |
|  | Recommended: Indents should be the width of 4 single-byte characters, and must be created with single-byte blank spaces. Using tabs is prohibited. |  |
| (Acceptable) | 01: <tr>  02: <th>User ID<n:forInputPage><span class="requiredMark">\*</span></n:forInputPage></th>  03: <td align="left">  04: <n:text name="systemAccount.userId" size="15" maxlength="10" />  05: <n:forInputPage>  06: (Maximum 10 single-byte alphanumeric characters)  07: <br />  08: <n:error name="systemAccount.userId" />  09: </n:forInputPage>  10: </td>  11: </tr> |  |

|  |  |  |
| --- | --- | --- |
| 5　Coding tips: Nablarch | |  |
|  | This chapter explains points to be aware of when writing JSP code with the Nablarch web application. |  |

|  |  |  |
| --- | --- | --- |
| 5.1　Specifying URIs | |  |
|  | Nablarch has custom tags for specifying URIs. This section explains points to be aware of when specifying URIs. |  |

|  |  |  |
| --- | --- | --- |
| 5.1.1 Use a relative path from the context root when specifying URIs | |  |
|  | When specifying URIs for form submissions from buttons or links, write the relative path from the context root. |  |
| (Acceptable) | 01: <n:submit cssClass="mainBtn" type="submit" name="search"  02: uri="/action/ss11AC/W11AC01Action/RW11AC0101"  03: value="search">  04: </n:submit> |  |