https://docs.oracle.com/cd/B14099_19/web.1012/b14014/keydev.htm#i1005631

Static Includes Versus Dynamic Includes

The `include` directive, described in "Directives", makes a copy of the included page and copies it into a JSP page (the "including page") during translation. This is known as a static include (or translate-time include) and uses the following syntax:

```
<%@ include file="/jsp/userinfopage.jsp" %>
```

The `jsp:include` tag, described in "Standard Actions: JSP Tags", dynamically includes output from the included page within the output of the including page during execution. This is known as a dynamic include (or runtime include) and uses the following syntax:

```
<jsp:include page="/jsp/userinfopage.jsp" flush="true" />
```

For those familiar with C syntax, a static include is comparable to a `#include` statement. A dynamic include is similar to a function call. They are both useful, but serve different purposes.

Note:

You can use static includes and dynamic includes only between pages in the same servlet context.

Logistics of Static Includes

A static include increases the size of the generated code for the including JSP page. It is as though the text of the included page is physically copied into the including page, at the point of the `include` directive, during translation. If a page is included multiple times within an including page, multiple copies are made.

A JSP page that is statically included is not required to be an independent, translatable entity. It simply consists of text that will be copied into the including page. The including page, with the included text copied in, must then be translatable. And, in fact, the including page does not have to be translatable prior to having the included page copied into it. A sequence of statically included pages can be fragments unable to stand on their own.

Logistics of Dynamic Includes

A dynamic include does not significantly increase the size of the generated code for the including page, although method calls, such as to the request dispatcher, will be added. The dynamic include results in runtime processing being switched from the including page to the included page, as opposed to the text of the included page being physically copied into the including page.

A dynamic include does increase processing overhead, with the necessity of the additional call to the request dispatcher.

A page that is dynamically included must be an independent entity, able to be translated and executed on its own. Likewise, the including page must be independent as well, able to be translated and executed without the dynamic include.

Advantages, Disadvantages, and Typical Uses of Dynamic and Static Includes

Static includes affect page size; dynamic includes affect processing overhead. Static includes avoid the overhead of the request dispatcher that a dynamic include necessitates, but may be problematic where large files are involved. (The service method of the generated page implementation class has a 64 KB size limit. See "Workarounds for Large Static Content or Significant Tag Library Usage".)

Overuse of static includes can also make debugging your JSP pages difficult, making it harder to trace program execution. Avoid subtle interdependencies between your statically included pages.

Static includes are typically used to include small files whose content is used repeatedly in multiple JSP pages. For example:

- Statically include a logo or copyright message at the top or bottom of each page in your application.
- Statically include a page with declarations or directives, such as imports of Java classes, that are required in multiple pages.
- Statically include a central "status checker" page from each page of your application. (See "Use of a Central Checker Page".)

Dynamic includes are useful for modular programming. You may have a page that sometimes executes on its own but sometimes is used to generate some of the output of other pages. Dynamically included pages can be reused in multiple including pages without increasing the size of the including pages.