**Jakarta Server Faces** (**JSF**; formerly JavaServer Faces) is a [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) specification for building [component](https://en.wikipedia.org/wiki/Software_component)-based [user interfaces](https://en.wikipedia.org/wiki/User_interface) for [web applications](https://en.wikipedia.org/wiki/Web_application)[[1]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-1) and was formalized as a standard through the [Java Community Process](https://en.wikipedia.org/wiki/Java_Community_Process) being part of the [Java Platform, Enterprise Edition](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition). It is also a [MVC](https://en.wikipedia.org/wiki/Model,_view,_controller) [web framework](https://en.wikipedia.org/wiki/Web_framework) that simplifies construction of [user interfaces](https://en.wikipedia.org/wiki/User_interface) (UI) for server-based applications by using reusable UI components in a page.[[2]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-2)

JSF 2 uses [Facelets](https://en.wikipedia.org/wiki/Facelets) as its default templating system. Other view technologies such as [XUL](https://en.wikipedia.org/wiki/XUL) or plain Java[[3]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-3) can also be employed. In contrast, JSF 1.x uses [JavaServer Pages](https://en.wikipedia.org/wiki/JavaServer_Pages) (JSP) as its default templating system.

History[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=1)]

|  |  |
| --- | --- |
| [[icon]](https://en.wikipedia.org/wiki/File:Wiki_letter_w_cropped.svg) | This section **needs expansion**. You can help by [adding to it](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=). *(August 2013)* |

In 2001, the original Java Specification Request (JSR) for the technology that ultimately became JavaServer Faces proposed developing a package with the name javax.servlet.ui[[4]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-4)

By June 2001, [*JavaWorld*](https://en.wikipedia.org/wiki/JavaWorld) would report on Amy Fowler's team's design of "the JavaServer Faces API" (aka "Moonwalk") – "an application framework for creating Web-based user interfaces".[[5]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-5)

**Versions**[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=2)]

* JSF 2.3 (2017-03-28) – Major features: search Expressions, extensionless URLs, bean validation for complete classes, push communication using [WebSocket](https://en.wikipedia.org/wiki/WebSocket), enhanced integration with CDI.[[6]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-6)
* JSF 2.2 (2013-05-21) – Introduced new concepts like stateless views, page flow and the ability to create portable resource contracts.[[7]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-7)
* JSF 2.1 (2010-11-22) – Maintenance release 2 of JSF 2.0. Only a very minor number of specification changes.[[8]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-8)[[9]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-9)
* JSF 2.0 (2009-07-01) – Major release for ease of use, enhanced functionality, and performance. Coincides with [Java EE](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition) 6.
* JSF 1.2 (2006-05-11) – Many improvements to core systems and APIs. Coincides with [Java EE](https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition) 5. Initial adoption into Java EE.
* JSF 1.1 (2004-05-27) – Bug-fix release. No specification changes.
* JSF 1.0 (2004-03-11) – Initial specification released.

How it works[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=3)]

Based on a component-driven [UI](https://en.wikipedia.org/wiki/User_interface) design-model, JavaServer Faces uses XML files called **view templates** or [Facelets](https://en.wikipedia.org/wiki/Facelets) views. The FacesServlet processes requests, loads the appropriate view template, builds a component tree, processes events, and renders the response (typically in the HTML language) to the client. The state of UI components and other objects of scope interest is saved at the end of each request in a process called **stateSaving** (note: *transient* **true**), and restored upon next creation of that view. Either the client or the server side can save objects and states.

JSF and Ajax[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=4)]

JSF is often used together with [Ajax](https://en.wikipedia.org/wiki/Ajax_(programming)), a [Rich Internet application](https://en.wikipedia.org/wiki/Rich_Internet_application) development technique. Ajax is a combination of web development techniques and technologies that make it possible to create rich user interfaces. The user interface components in [Mojarra](https://en.wikipedia.org/wiki/Project_Mojarra) (the JSF [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation)[[10]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-10)) and [Apache MyFaces](https://en.wikipedia.org/wiki/Apache_MyFaces) were originally developed for HTML only, and Ajax had to be added via JavaScript. This has changed, however:

Because JSF supports multiple output formats, Ajax-enabled components can easily be added to enrich JSF-based user interfaces. The JSF 2.0 specification provides built-in support for Ajax by standardizing the Ajax request lifecycle and providing simple development interfaces to Ajax events, allowing any event triggered by the client to go through proper validation, conversion, and finally method invocation, before returning the result to the browser via an XML DOM update.

JSF 2 includes support for [graceful degradation](https://en.wikipedia.org/wiki/Graceful_degradation) when [JavaScript](https://en.wikipedia.org/wiki/JavaScript) is disabled in the browser.

Ajax-enabled components and frameworks[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=5)]

The following companies and projects offer Ajax-based JSF frameworks or component libraries:

* [Apache MyFaces](https://en.wikipedia.org/wiki/Apache_MyFaces) – The [Apache Foundation](https://en.wikipedia.org/wiki/Apache_Foundation) JSF implementation with Ajax components
* Backbase *Enterprise Ajax – JSF Edition* – [Ajax framework](https://en.wikipedia.org/wiki/Ajax_framework)
* BootsFaces *Open source JSF Framework based on Bootstrap*
* [IBM Notes](https://en.wikipedia.org/wiki/IBM_Notes) – [XPages](https://en.wikipedia.org/wiki/XPages)
* [ICEfaces](https://en.wikipedia.org/wiki/ICEfaces) – open-source, Java JSF extension framework and [rich](https://en.wikipedia.org/wiki/Rich_Internet_application) components, Ajax without JavaScript
* [JBoss](https://en.wikipedia.org/wiki/JBoss) [*RichFaces*](https://en.wikipedia.org/wiki/RichFaces) (derived from and replaces [*Ajax4jsf*](https://en.wikipedia.org/wiki/Ajax4jsf)) – Ajax-enabled JSF components for layout, file upload, forms, inputs and many other features. It reached its end-of-life in June 2016.
* [OmniFaces](https://en.wikipedia.org/wiki/OmniFaces) – open-source JSF utility library
* [OpenFaces](https://github.com/OpenFaces-org/OpenFaces) – Ajax framework with JSF components
* [Oracle](https://en.wikipedia.org/wiki/Oracle_Corporation) *ADF Faces Rich Client* – [Oracle Application Development Framework](https://en.wikipedia.org/wiki/Oracle_Application_Development_Framework)
* [PrimeFaces](https://en.wikipedia.org/wiki/PrimeFaces) – Ajax framework with JSF components
* [Sun](https://en.wikipedia.org/wiki/Sun_Microsystems) Java BluePrints AJAX components
* [ZK](https://en.wikipedia.org/wiki/ZK_Framework) – Ajax framework with JSF components

Latest developments[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=6)]

[Facelets](https://en.wikipedia.org/wiki/Facelets) (which was designed specifically for Java Server Faces) was adopted as the official view technology for JSF 2.0. This eliminates the life-cycle conflicts that existed with JSP, forcing workarounds by Java developers.[[11]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-11) Facelets allows easy component/tag creation using XML markup instead of Java code, the chief complaint against JSF 1.x.

The new JSF developments also provide wide accessibility to [Java 5 annotations](https://en.wikipedia.org/wiki/Java_annotation) such as @ManagedBean, @ManagedProperty and @FacesComponent that removes the need for faces-config.xml in all cases except framework extension. Navigation has been simplified, removing the need for faces-config.xml navigation cases. Page transitions can be invoked simply by passing the name of the desired View/Facelet.

Addition of Partial State Saving and DOM updates are part of the built-in standardized [Ajax](https://en.wikipedia.org/wiki/Ajax_(programming)) support.

The latest JSF release has built-in support for handling resources like images, CSS and Javascript, allowing artifacts to be included with component libraries, separated into JAR files, or simply co-located into a consistent place within the Web application. This includes logical naming and versioning of resources.

JSF 2.0 also includes a number of other changes like adding support for events, separate development, staging, and production modes, similar to RAILS\_ENV in [Ruby on Rails](https://en.wikipedia.org/wiki/Ruby_on_Rails), and significantly expanding the standard set of components.

Criticisms[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=7)]

In their January 2014 "Technology Radar" publication, [ThoughtWorks](https://en.wikipedia.org/wiki/ThoughtWorks) wrote:[[12]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-12)

We continue to see teams run into trouble using JSF -- JavaServer Faces -- and are recommending you avoid this technology. Teams seem to choose JSF because it is a JEE standard without really evaluating whether the programming model suits them. We think JSF is flawed because it tries to abstract away [HTML](https://en.wikipedia.org/wiki/HTML), [CSS](https://en.wikipedia.org/wiki/CSS) and [HTTP](https://en.wikipedia.org/wiki/HTTP), exactly the reverse of what modern [web frameworks](https://en.wikipedia.org/wiki/Web_framework) do. JSF, like ASP.NET webforms, attempts to create [statefulness](https://en.wikipedia.org/wiki/Statefulness) on top of the stateless protocol HTTP and ends up causing a whole host of problems involving shared server-side state. We are aware of the improvements in JSF 2.0, but think the model is fundamentally broken. We recommend teams use simple frameworks and embrace and understand web technologies including HTTP, HTML and CSS.

In the article published November 2014 in the DZone website, titled "Why You Should Avoid JSF", Jens Schauder wrote:[[13]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-13)

Facelets, the preferred presentation technology of JSF looks at first sight like an ordinary templating technology like the good old JSP or Thymeleaf. But if you look closer the horror becomes obvious. In the same place where you structure your HTML, you also place the logic what parts of the UI should get updated on an action. A clear violation of the [separation of concerns](https://en.wikipedia.org/wiki/Separation_of_concerns) principle in my book. Even better is the immediate attribute which changes the server side life cycle! And if this isn’t enough it does it in different ways depending on what tag you use it on. You can’t make stuff like this up.

**Rebuttal to criticisms**[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=8)]

In February 2014, Çağatay Çivici (PrimeFaces Lead) responded to ThoughtWorks criticisms in a post titled: JSF is not what you’ve been told anymore. Çivici argues that improvements in JSF over the years offer many features that embrace modern web development, providing the option to write your own JavaScript, HTML, and CSS. Also regarding state, Çağatay wrote:

JSF is a stateful framework by nature and state makes web applications easy to develop with. With improved state management techniques introduced in JSF 2.0+ (e.g. stateless mode, partial state saving), JSF can scale as well.[[14]](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_note-14)

References[[edit](https://en.wikipedia.org/w/index.php?title=Jakarta_Server_Faces&action=edit&section=9)]

* 1. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-1) [JavaServer Faces Technology](http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html)
  2. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-2) [*"JSF 2.0 Tutorial"*](http://www.mkyong.com/tutorials/jsf-2-0-tutorials/)*. mkyong. 2010-12-12. Retrieved 2017-04-28. JavaServer Faces (JSF) 2.0, is an MVC web framework which focus on simplifies building user interfaces (comes with 100+ ready UI tags) for Java web application and make reusable UI component easy to implement.*
  3. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-3) [NoVDL: Write your JSF views in pure Java](https://rogerkeays.com/novdl-write-your-jsf-views-in-pure-java)
  4. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-4) [*"JSR 127: JavaServer Faces"*](https://www.jcp.org/en/jsr/detail?id=127)*. Java Community process. Oracle Corporation. 2014. Retrieved 2014-08-05. 2.6 Is there a proposed package name for the API Specification? (i.e., javapi.something, org.something, etc.) [:] javax.servlet.ui*
  5. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-5) *Armstrong, Eric (June 6, 2001).*[*"Java Web services: What's not to like?"*](https://www.infoworld.com/article/2075969/java-web-services--what-s-not-to-like-.html)*.*[*JavaWorld*](https://en.wikipedia.org/wiki/JavaWorld)*. Retrieved 2020-07-27. The JavaServer Faces API (aka Moonwalk) promises to provide an elegant solution for implementing interactive functionality on incompatible browsers. […] Designed by a team led by Amy Fowler, Sun's AWT and Swing architect, the JavaServer Faces API will provide a collection of GUI tools that will run on common browsers using standard HTML.*
  6. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-6) *Tijms, Arjan.*[*"What's new in JSF 2.3?"*](https://arjan-tijms.omnifaces.org/p/jsf-23.html)*. Musings of a Java EE developer. Retrieved 2020-07-27.*
  7. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-7) [JSF 2.2 (JSR-344) is final |techscouting through the java news](https://blog.oio.de/2013/04/16/jsf-2-2-jsr-344-is-final/). Blog.oio.de. Retrieved on 2020-07-27.
  8. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-8) [JSR 314 JavaServer Faces 2.1 JSF 2.1 |techscouting through the java news](https://blog.oio.de/2010/11/24/jsr-000314-javaservertm-faces-2-1/). Blog.oio.de. Retrieved on 2020-07-27.
  9. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-9) *Bosch, Andy (2010-11-29).*[*"Was ist neu in JSF 2.1"*](http://it-republik.de/jaxenter/news/Was-ist-neu-in-JSF-2.1-057653.html)*(in German). it-republik.de. Retrieved 2013-02-19.*
  10. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-10) *Ryan Lubke (5 December 2007).*[*"Project Mojarra - the JSF RI gets a code name"*](https://blogs.oracle.com/rlubke/entry/project_mojarra_the_jsf_ri)*.*
  11. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-11) *Bergsten, Hans.*[*"Improving JSF by dumping JSP"*](http://onjava.com/pub/a/onjava/2004/06/09/jsf.html)*. O'Reilly. Retrieved 18 August 2011.*
  12. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-12) [*"Technology Radar"*](https://web.archive.org/web/20160407102412/http:/thoughtworks.fileburst.com/assets/technology-radar-jan-2014-en.pdf)*(PDF). ThoughtWorks. January 2014. p. 12. Archived from*[*the original*](http://thoughtworks.fileburst.com/assets/technology-radar-jan-2014-en.pdf)*(PDF) on 2016-04-07. Retrieved 2014-01-30.*
  13. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-13) *Jens Schauder (November 2014).*[*"Why You Should Avoid JSF"*](https://dzone.com/articles/why-you-should-avoid-jsf)*. DZone.*
  14. [**^**](https://en.wikipedia.org/wiki/Jakarta_Server_Faces#cite_ref-14) [*"JSF is not what you've been told anymore"*](https://www.primefaces.org/jsf-is-not-what-youve-been-told-anymore/)*. PrimeFaces. 2014-02-12. Retrieved 2020-11-24.*